Conformación tridimensional y reconocimiento molecular de biopolímeros. Aplicación de RMN multidimensional y desarrollo de metodología de cálculo y estructural.

Memoria de Tesis Doctoral.
Daniel Monleón Salvadó
Dpto. Química Física
Universitat de València

Los Dres. **Bernardo Celda Muñoz** y **Roberto Tejero Toquero**, profesores titulares del Departamento de Química Física de la Facultad de Ciencias Químicas de la Universitat de València,

CERTIFICAN:

que D. Daniel Monleón Salvadó ha realizado bajo su dirección en este laboratorio la memoria que presenta con el título de: "Conformación Tridimensional y reconocimiento molecular de biopolímeros. Aplicación de RMN multidimensional y desarrollo de metodología de cálculo y estructural"

Valencia, Octubre de 1998

Fdo. Dr Bernardo Celda Muñoz Fdo. Dr. Roberto Tejero Toquero.

A mi madre

A mis hermanos

Índice General

Capítulo 1

Introducción General

Desde que se descubrió la existencia del átomo, la química ha tenido uno de sus principales intereses en comprender la ordenación y estructuración de los mismos a nivel microscópico y la influencia de éstas en las propiedades macroscópicas. La ordenación de los enlaces, así como las posibles orientaciones espaciales de los diferentes átomos constitutivos de una molécula tienen una importancia crucial en las propiedades tanto químicas como físicas del compuesto. Es por ello, que la química estructural es una de las disciplinas más pujantes en nuestros días.

1.1 Estructura y función en biopolímeros

La importancia de la disposición de los átomos constitutivos de una molécula en el espacio aumenta de interés cuando dicha molécula puede presentar una determinada función biológica, en especial aquellas de impacto médico o medioambiental. Entre este tipo de moléculas cabe destacar por su abundancia e importancia a las proteínas y a los ácidos nucleicos. Virtualmente, cada propiedad que caracteriza a un organismo vivo viene de alguna manera determinada por las proteínas y por los ácidos nucléicos. Los ácidos nucléicos codifican la información genética y expresan la misma utilizando casi exclusivamente otras proteínas. Por otra parte, los ácidos nucléicos contienen la información necesaria para la producción de las proteínas resultando así un sistema cooperativo que es esencial para el desarrollo de la vida.

Así, las proteínas y los ácidos nucléicos son los responsables de la expresión y transmisión de la información biológica. Como componentes

esenciales del correcto funcionamiento de la mayoría de procesos fundamentales en biología molecular, una propiedad muy importante en ellas es la especificidad de su función. Para algunas proteínas, la especificidad de su acción está definida de un modo tan estricto que un pequeño cambio en la molécula ligando puede conducir a una disminución importante en la asociación de ambas moléculas, en otras palabras alterar el proceso de reconocimiento molecular.

Considerando la estructura de biopolímeros a nivel atómico es importante tener presente la escala de tiempo del método seleccionado para su observación. En macromoléculas hay diferencias, no siempre despreciables, entre la estructura particular instantánea, suponiendo la observación en una escala de tiempos inferior a los movimientos vibracionales de la molécula, y la estructura promedio que se observa en un experimento de rayos X, por ejemplo, cuyo resultado final es el promedio de la observación durante un período de tiempo más o menos elevado y en un estado cristalino compacto. Estas diferencias son mucho más importantes cuando hablamos de estructuras en disolución, que por otra parte, la mayoría de los casos presentan mayor interés bioquímico teniendo en cuenta que es un medio más próximo al fisiológico. Fué precisamente el elevado nivel de detalle de las estructuras determinadas mediante rayos X lo que llevó durante mucho tiempo a aceptar la creencia de que las proteínas eran unos cuerpos rígidos y que los átomos permanecían en posiciones fijas durante la mayoría del tiempo. Simultáneamente, los procesos de reconocimiento molecular eran descritos desde un punto de vista estático, *llave-cerradura*, para las interacciones enzima-sustrato. Sin embargo, hoy en día el hecho de que los átomos de una molécula, desde el átomo de hidrógeno a la proteína más gigantesca, están en incesante movimiento a temperatura ambiente queda fuera de toda duda. Aunque, debido al carácter dinámico de las estructuras moleculares es bastante improbable que en un biopolímero con gran cantidad de modos de vibración posibles la estructura promedio sea adoptada en ningún momento, no es menos interesante el conocimiento de dicha estructura teniendo siempre presente las limitaciones que le impone su carácter de *promedio*.

Las estructuras obtenidas mediante RMN sin embargo, presentan la ventaja de que nuncca se dan como estructura promedio sino como una familia de estructuras que satisfacen las restricciones estructurales experimentales obtenidas de los espectros del mejor modo posible. De este modo, es posible evaluar de manera indirecta la flexibilidad de cada segmento de la proteína, diferenciando las zonas mejor definidas respecto a las peor de-

finidas. En principio, una región flexible como un giro ha de tener menos distancias cortas fijas que una rígida como pueda ser una hoja β. Esto llevará a una peor definición de las coordenadas finales en la región flexible. Sin embargo, esto no es siempre así, ya que el error experimental, unido a la posible interconversión entre diferentes conformaciones hace que las restricciones introducidas correspondan a una estructura promedio. La interpretación de las coordenadas de la estructura promedio resulta complejo sobre todo en proteínas en disolución.

En numerosas ocasiones, las dificultades experimentales para obtener muestras en las condiciones requeridas para aplicar una determinada técnica hace imposible la determinación experimental de la estructura de una proteína ya sea promedio o instantánea. Esta limitación pone de manifiesto la importancia de los métodos teóricos de determinación de estructuras. De la cantidad de proteínas descubiertas en la actualidad, tan sólo una mínima parte ha sido caracterizada estructuralmente. La incapacidad de obtener muestras en condiciones adecuadas para la determinación estructural experimental unida a la gran cantidad de tiempo que en la actualidad se requiere para la determinación de estructuras de proteínas hacen que dicha relación, aunque aumentando a velocidad relativa importante, permanezca en valores muy pequeños. Una metodología teórica que permite la determinación de estructuras de modo fiable según diferentes factores es la modelización por homología. Dicha técnica se basa en la necesaria similitud estructural que debe existir entre proteínas homólogas. Basándonos en esta similitud se puede extraer información de proteínas homólogas caracterizadas estructuralmente para obtener modelos teóricos de proteínas cuya estructura aún no ha sido determinada experimentalmente. De este modo, las estructuras de proteínas determinadas se podría aumentar en un orden de magnitud.

1.2 Dinámica de biopolímeros

Tanto la estructura como las fluctuaciones de los átomos constitutivos de una proteína son los responsables de muchas de las funciones que desempeñan. Es por ello de gran interés el conocimiento de la estructura de una proteína cuando se pretende realizar un estudio exhaustivo de sus funciones así como la identificación de las partes más flexibles y los centros de mayor fluctuación, que suelen estar asociados a los centros activos y en general a los procesos de reconocimiento molecular, previos y fundamentales

para cualquier tipo de actividad, química y/o biológica. Dicha asociación se corresponde a la necesidad en muchas ocasiones de reordenaciones estructurales a nivel local de los átomos del centro activo para que se pueda producir la actividad o función de la proteína y/o del ácido nucléico, como puede ser una transferencia electrónica o una oxidación.

Aunque las simulaciones teóricas de la dinámica molecular han aportado nuevas luces sobre el problema de la determinación de la movilidad intramolecular de proteínas, el apoyo de los datos experimentales siempre resulta necesario debido sobre todo a algunas interacciones típicas de moléculas en disolución. Tal es el caso de aquellas interacciones con una fuerte contribucción entrópica a la energía libre, como pueda ser la solvatación y en general las interacciones con el disolvente. La introducción de dichas contribuciones en las funciones de potencial que se suelen utilizar en los cálculos de dinámica molecular dista mucho de ser trivial y, aunque en numerosos estudios teóricos se ha introducido el disolvente de modo explícito mediante cajas de agua con carácter periódico y aproximaciones similares, la parametrización de la molécula de agua dentro de los campos de fuerza habituales difícilmente contiene términos que consideren de modo completo la importante contribución entrópica de dichas interacciones. Por ello, los métodos experimentales que permiten deducir información sobre la dinámica de las proteínas en disolución cobran mayor interés en los estudios estructurales día a día.

Tal es el caso de los estudios de dinámica de proteínas mediante la observación y análisis de los procesos de relajación. La relajación es un proceso físico de recuperación de posiciones de equilibrio que depende en gran medida de las condiciones físicas que rodean a un determinado núcleo. Del análisis detallado de dichos procesos se puede extraer información estructural sobre el entorno de un determinado núcleo, así como información dinámica de las interacciones con los núcleos más cercanos. Estas interacciones suele estar dominadas por la más importante que es el enlace químico con su vecino. De este modo, mediante técnicas de relajación se puede conocer información sobre la dinámica de un enlace particular dentro de una proteína. Sin embargo no es esta la única información sobre dinámica que se puede extraer de los procesos de relajación. Una contribución importante a estos procesos es el llamado intercambio químico asociado a la interconversión de un determinado segmento de la molécula entre varias conformaciones (habitualmente dos). Así, mediante el estudio de los procesos de relajación que sufren los núcleos de una determinada proteína podemos delimitar las zonas más flexibles y la velocidad de los

movimentos asociados a diferente enlaces, asi como las regiones en que se producen cambios conformacionales importantes de un modo cualitativo y semicuantitativo.

1.3 Refinamiento de estructuras de biopolímeros obtenidas mediante RMN

Aunque la determinación experimental de la estructura secundaria de una proteína sea relativamente sencillo de modo cualitativo mediante, por ejemplo, la búsqueda de determinados patrones en los espectros de RMN, en muchas ocasiones el interés de la estructura de una determinada proteína se centra en el conocimiento de la conformación preferida de una determinada cadena lateral o en la conformación local de un determinado segmento de la proteína a escala de angstroms. La determinación de estructuras mediante rayos X proporciona ese nivel de detalle en la mayoría de ocasiones. Sin embargo, el interés del conocimiento de las estructuras de biopolímeros en disolución hace que las posibles metodologías orientadas al refinamiento de estructuras de RMN cobren un interés creciente dentro del panorama de la bioquímica estructural.

La determinación de estructuras mediante resonancia magnética nuclear se basa fundamentalmente en el efecto nuclear Overhauser (NOE). La observación de una resonancia NOE entre dos protones en una proteína se corresponde a una distancia interprotónica inferior a 5 Å y calibrable según una escala de intensidades. La obtención de más y mejor calibrados NOEs es el primer paso en el refinamiento de una estructura por RMN. Sin embargo, el número de NOEs observables en el espectro NOESY (Espectroscopía Nuclear Overhauser) está limitado por el solapamiento entre resonancias y por factores experimentales asociados a la potencia de los equipos utilizados. Además, el elevadísimo número de grados de libertad teóricos de una macromolécula hace que las restricciones impuestas por las distancias derivadas de los espectros NOE no sean siempre suficientes para determinar la conformación preferida de cierto residuo o el plegamiento de un segmento de la proteína en un giro de modo inequívoco.

Es bien sabido que los ángulos diedros limitan de modo más eficaz que las distancias el espacio conformacional accesible a un determinado conjunto de átomos [1, 127]. Esto es debido a que, aunque tanto los distancias como los ángulos de enlace propios de la estructura covalente de una molécula se

consideren grados de libertad desde el punto de vista teórico, los ángulos diedros rotables son en la práctica los que sufren mayores variaciones y de hecho aportan la mayor contribución en las variaciones conformacionales de un polímero, sino la única. Por ello, la determinación de los valores del mayor número posible de ángulos diedros en una proteína es fundamental en el refinamiento de la estructura de la misma. En concreto, la determinación de los ángulos que implican a la orientación de las cadenas laterales χ_1 resulta de particular interés en el análisis de la actividad de algunos residuos dentro de una proteína, ya que son los que están directamente relacionados con la superficie molecular.

Por otra parte, la limitada aplicabilidad de las restricciones de distancias en el cálculo de estructuras de ácidos nucleicos hacen fundamental la determinación de los ángulos diedros en el proceso. La pequeña diferencia entre las distancias características de una conformación u otra en las conformaciones típicas de ADN (A-ADN, B-ADN y Z-ADN) hace que únicamente con la información extraída de los espectros NOESY sea particulamente complicado deducir la conformación tridimensional del ADN con detalle.

1.4 Bases de datos estructurales

Como ya hemos visto, la obtención de información estructural susceptible de ser utilizada en el refinamiento de estructuras tridimensionales de biopolímeros es un objetivo fundamental de un gran número de estudios bioquímicos. Sin embargo, las necesidades actuales de la bioquímica no exigen únicamente precisión en los resultados estructurales. Cada vez es más urgente la resolución de más estructuras de proteínas por lo que la velocidad del proceso se convierte en un factor de gran interés a la hora de seleccionar la técnica más adecuada.

La determinación precisa y rápida de la estructura de proteínas se ha convertido de este modo en uno de los objetivos fundamentales de la bioquímica, biología molecular y biofísica actuales. Esta necesidad surge sobre todo en los dos últimos años como respuesta a la acelerada evolución de los proyectos de secuenciación y análisis del material genético conocidos como proyectos GENOMA de las distintas especies en curso. Particularmente atractivo es el proyecto GENOMA humano por obvias y directas implicaciones, tanto a nivel biomédico como social, pudiéndose considerar una inflexión cualitativa importante en la evolución científica y social.

La posibilidad de secuenciación completa de todos los genes de un organismo, y como se ha indicado especialmente el humano, permite y posibilita, *a priori*, entre otras importantes aplicaciones, el diagnóstico de todo un conjunto de enfermedades de origen genético. No obstante, el listado de nucleótidos recogidos en una librería de un GENOMA no resulta suficiente. Más bien es el punto de partida para la comprensión total de todos los procesos bioquímicos que dicho gen o genes implican. En otras palabras, la secuencia de pares de bases obtenida en un proyecto GENOMA permite la posibilidad de deducir la estructura primaria de todas aquellas proteínas que codifica, pero sus posibilidades quedan prácticamente reducidas a dicha deducción. Concretamente, en la mayoría de los genes hasta ahora secuenciados, la proporción de proteínas propuestas sin una función conocida resulta muy elevada, a veces superior al 50 %.

Lógicamente, desde el punto de vista biológico, es imprescindible conocer la función de todas y cada una de las proteínas expresadas en un organismo para entender tanto su actividad y funcionamiento normal como las irregularidades que pueda presentar. Por tanto, el paso lógico siguiente a la secuenciación genética completa es la obtención de la función de todas y cada una de las proteínas codificadas en cada gen. Desafortunadamente, hoy en día no se dispone del conocimiento científico suficiente como para inferir de una manera directa la función o funciones de una proteína a partir de su secuencia primaria.

Sin embargo, sí que parece estar bien comprobada la relación directa entre estructura y función. Por consiguiente, la determinación de la estructura espacial de proteínas es la etapa intermedia entre la secuencia genética (secuencia primaria de una proteína) y la elucidación de la función de las mismas. En otras palabras, una vez demostrada la eficacia y el potencial del GENOMA secuencial aplicado a diversos y diferentes organismos, la siguiente etapa cuyos inicios se están gestando en la actualidad, es el GENOMA ESTRUCTURAL o PROTEOMA. Entre las consecuencia obvias directas del PROTEOMA aplicado a plantas, microorganismos y organismos superiores, entre ellos el humano, se puede considerar las medioambientales y terapéuticas. Básicamente se podría resumir de la siguiente manera, el proyecto GENOMA implica *diágnostico* mientras que el PROTEOMA implicará *terapia*. Además, la determinación de la estructura tridimensional de todas las proteínas de un gen o conjunto de genes de un organismo aportará la información imprescindible necesaria para el entendimiento global de todos los mecanismos bioquímicos involucrados en su funcionamiento.

De nuevo surge un problema, aunque en este caso más bien de carácter tecnológico, la determinación de la conformación tridimensional de proteínas no resulta inmediata, requiere el esfuerzo experimental de su producción y purificación, y la utilización de las técnicas estructurales precisas y adecuadas, en concreto RMN y Rayos X. En los dos últimos años se vienen realizando esfuerzos significativos para la automatización de ambas técnicas, con el objetivo de obtener una estructura diaria. No obstante, este esfuerzo es insuficiente para determinar la conformación de miles de secuencias primarias que anualmente se depositan en los bancos de datos. Esto implica que además de las herramientas experimentales, deben desarrollarse aquellas de carácter teórico que permitan determinar la conformación espacial de proteínas. Actualmente está bien reconocido que la predicción de la estructura tridimensional de proteínas mediante homología resulta una de las fuentes estructurales teóricas más fiables y rápidas. Desde el punto de vista conformacional dos son las aproximaciones al problema estructural: i) determinación del plegamiento global de la proteína; y ii) elucidación precisa de la configuración tridimensional de la misma. La determinación del plegamiento global posibilita una rápida evaluación de la función o posibles funciones de una proteína desconocida, mediante la comparación directa con homólogas en el espacio funcional de proteínas. No obstante, la correlación *plegamiento-función* no resulta directa en una importante proporción de los casos, por lo que un segundo nivel de análisis lo constituye la determinación, lo más precisa posible, de la estructura espacial de las proteínas de función desconocida. Las aportaciones que se pueden realizar en esta área van desde la implementación de las metodologías experimentales, por ejemplo nuevas secuencias de pulsos en RMN, al desarrollo y mejora de nuevos programas dedicados al perfeccionamiento de la determinación espacial de proteínas, y biopolímeros en general, tanto en su aplicación directa a los datos experimentales como en la predicción teórica por homología.

1.5 Objetivos de la Tesis

El objetivo principal de la Tesis ha sido la exploración, el análisis y el perfeccionamiento de las técnicas más habituales utilizadas en la determinación de la estructura y dinámica de biopolímeros en disolución. En concreto, se han utilizado métodos teóricos y experimentales para obtener las estructuras de dos biopolímeros entre los grupos más numerosos de ellos: proteínas y ácidos nucleicos.

La obtención de la estructura de una proteína de interés bioquímico por sus peculiaridades cinéticas mediante dos técnicas diferentes ha sido uno de los objetivos principales en la Tesis. La aplicación de las técnicas de modelización por homología y la realización de un análisis exhaustivo sobre los factores que afectan a la calidad del método permitirá aportar la obtención de mejores estructuras de proteínas mediante el uso de esta técnica. La determinación de la estructura de la Plastocianina *Synechocystis* mediante un método teórico como es la modelización por homología y otro experimental como es la RMN permitirá evaluar el grado de convergencia entre ambas y sopesar la validez de las técnicas teóricas como medio de ampliación de las estructuras de proteínas caracterizadas.

La aplicación del formalismo libre de modelo de Lipari-Szabo al análisis de los procesos de relajación en RMN mediante la confección de un programa escrito en C permitirá aportar mayor objetividad al estudio de la dinámica de proteína mediante técnicas de RMN. El estudio de sistemas cuya dinámica ha sido analizada previamente mediante el uso de otros programas diferentes y la comparación con los resultados obtenidos permitirá conocer la validez tanto de la aproximación empleada como del algoritmo programado. El análisis de la dinámica de una proteína tan bien estudiada como el wt-BPTI (Inhibidor Básico de la Tripsina de Páncreas Bovina en forma nativa) frente a uno de sus mutantes constituye un ejemplo de análisis exhaustivo de la dinámica de una proteína mediante relajación en RMN.

El desarrollo de las rutinas aplicadas en el programa HYPER para la obtención de restricciones de ángulos diedros en la cadena principal y en las cadenas laterales mediante la utilización de información de NOEs y constantes de acoplamiento escalar ${}^3J$ constituye un ejemplo de la dirección a seguir en el refinamiento de estructuras de RMN. La novedosa metodología utilizada en la determinación del ángulo diedro χ_1 mediante el análisis de intensidades TOCSY (espectroscopía de correlación total) frente al tiempo de mezcla del experimento, incluso para residuos sin dos protones metilénicos como las treoninas y su aplicación a una proteína cuya estructura ya ha sido determinada, permite evaluar la importancia del desarrollo de metodologías similares en el refinamiento de estructuras por RMN.

Por último, la determinación de la estructura de un oligonucleótido en disolución mediante RMN utilizando las técnicas de simulación de picos DQF-COSY (Espectroscopía de Correlación Escalar con Filtro de Doble Cuanto) y su comparación con la estructura determinada mediante rayos X, confirma el interés que presentan las estructuras en disolución de bio-

polímeros y las sustanciales diferencias que pueden presentar frente a las estructuras cristalinas determinadas mediante difracción de rayos X. Asimismo, la estructura del d(CCGCGG)$_2$ es un ejemplo de la influencia de la secuencia nucleotídica en la conformación local de cadenas de ADN.

Capítulo 2

Homología de la Plastocianina
Synechocistys

El plegamiento tridimensional de una proteína está determinado por la secuencia de aminoácidos del polipéptido. Normalmente, es imposible generar la estructura 3D directamente de una secuencia. Es necesario aportar información adicional como pueden ser datos experimentales obtenidos de diferentes técnicas como puedan ser la cristalografía de Rayos X o la Resonancia Magnética Nuclear. Sin embargo, estas técnicas están a menudo limitadas a proteínas y otras macromoléculas capaces de ser obtenidas en las cantidades apreciables de acuerdo a las necesidades específicas de cada una de ellas.

Existen numerosas bases de datos de información biológica que pueden aportar información adicional a la generación de estructuras de proteínas. Las bases de datos estructurales, como pueda ser el Banco de Datos de Proteínas Brookhaven (PDB), contienen información adicional a las estructuras tridimensionales provenientes de diferentes técnicas. Estos datos son susceptibles de ser utilizados para predecir estructuras 3D de secuencias con estructuras desconocidas mediante la modelización por homología.

2.1 Introducción a la modelización por homología

2.1.1 Generalidades

El término de homología es una inferencia de la teoría de la evolución basada en el estudio comparado de similaridad y significado biológico. Dos se-

cuencias que son homólogas tienen una relación evolutiva proveniente de un común antecesor. El término de similaridad es una medida del parecido entre dos secuencias y, aunque posteriormente veremos que no es un factor absoluto sobre la calidad de la homología estructural, puede ser tomada como una escala relativa de las propiedades seleccionadas para ser comparadas. Es muy frecuente observar que proteínas con funciones biológicas homólogas en diferentes organismos presentan una elevada similaridad secuencial, aunque fundamentalmente se observa que mantienen la estructura tridimensional.

En el contexto de la biología molecular, homología y similaridad son conceptos que se utilizan frecuentemente para aportar nuevos datos sobre posibles funciones de los productos de nuevos genes. Una base de datos de secuencias puede ser comparada con una secuencia específica con función desconocida buscando similitudes con genes o proteínas conocidos. Aunque esta posibilidad solo es aplicable a los casos de proteínas con funciones conocidas, permite aumentar la velocidad de análisis del genoma humano en casi un orden de magnitud. Aunque en numerosas ocasiones la similaridad secuencial no supere el 30 % de los residuos, se ha comprobado que una baja identidad secuencial puede dar lugar igualmente a una alta conservación en la estructura tridimensional de las proteínas.

En la modelización por homología, se utilizan estructuras de proteínas determinadas experimentalmente para predecir las conformaciones de otras proteínas con secuencias de aminoácidos similares. Esto es posible gracias a que pequeños cambios en la secuencia conducen normalmente a pequeños cambios en la estructura 3D [2, 3]. La precisión de los modelos de proteínas obtenidos mediante esta aproximación es comparable a la de modelos calculados mediante otros métodos teóricos de similar alcance [26]. La modelización por homología produce estructuras con valores de desviación cuadrática media de hasta 1 Å para secuencias con la suficiente similaridad respecto a estructuras 3D conocidas [20, 26]. Sin embargo, la modelización por homología no es tan precisa como puedan ser las técnicas experimentales de Rayos X y Resonancia Magnética Nuclear que pueden determinar una estructura de proteína con un resolución equivalente de hasta 0.3 y 0.5 Å respectivamente.

Por otra parte, como se puede deducir de la aplicación del método en sí, la modelización por homología tiene una importante limitación que consiste en la identidad secuencial necesaria para justificar la homología. Por lo tanto, los cálculos de modelización por homología están restringidos a

Técnica	Resolución equivalente (Å)	Peculiaridades Peculiaridades
RMN	1.3	Proteínas de pequeño tamaño posible tratamiento en disolución, no es necesario cristalizar, posible automatización
Rayos X	0.83	Requiere muestras cristalinas, alta resolución, posibles efectos de empaquetamiento en estructura
Modeliz. Homología	2.2	Requiere encontrar proteína homóloga con estructura determinada, requiere identidad secuencial alta, no hay datos experimentales.

Tabla 2.1: Cuadro comparativo de las diferentes técnicas para resolución equivalente de estructuras tridimensionales de proteínas.

secuencias con similaridad a proteínas de estructura 3D conocida. Sin embargo, como un 28 % de las secuencias conocidas tienen al menos un 25 % de identidad secuencial con alguna proteína de estructura conocida, se ha realizado una estimación según la cual el número de estructuras conocidas puede crecer en un orden de magnitud respecto a las determinadas experimentalmente mediante esta aproximación. Esta relación es susceptible de ser aumentada a medida que aumente el porcentaje de estructuras determinadas experimentalmente de diferentes familias. No obstante, el número de conformaciones similares que se han encontrado entre las estructuras determinadas sugiere que sólo un limitado número de conformaciones plegadas son usadas por todas las proteínas, del orden de 1000, 100 de las cuales ya han sido determinadas.

En muchos de los casos en que la secuencia diverge considerablemente, la dificultad de predecir la estructura tridimensional reside en reconocer el patrón de homología. En estos casos, suele buscarse el apoyo de alineaciones múltiples y predicciones de estructura secundarias para reconocer la homología [14]. De este modo, se puede evaluar los residuos que son cruciales para esa estructura y se puede concretar que en la alineación se conserve el aminoácido o por lo menos el tipo de aminoácido. El resto de aminoácidos puede no ser especificado, pero debe haber un número mínimo y máximo de residuos entre residuos cruciales para asegurar la fiabilidad de la predicción. De este modo se puede asegurar que el fundamento de la homología, que no es otro que la conservación de ciertos rasgos comunes a una familia de proteínas, se respetará en los posteriores cálculos.

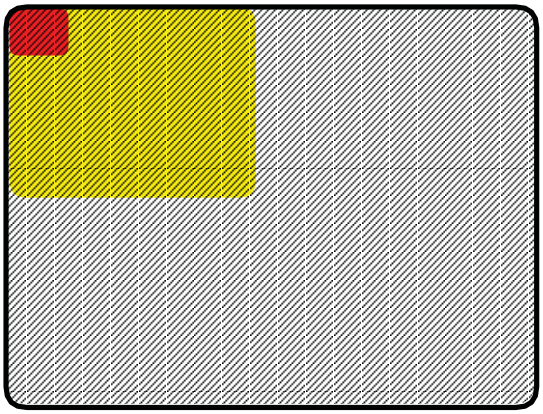


Figura 2.1: Esquema representando los porcentajes de estructuras conocidas y estructuras susceptibles de ser determinadas mediante modelización por homología.

Existen varios métodos para determinar las coordenadas atómicas de una estructura problema a partir de otra de estructura conocida [14, 20, 26]. Sin embargo, cualquiera que sea el método de homología empleado para calcular la estructura de la proteína problema, será necesario siempre basarlo en una alineación. Por ello y por lo visto en los párrafos anteriores, la elección de una buena estructura plantilla y de una correcta alineación con su secuencia es el paso crucial en cualquier cálculo estructural mediante modelización por homología.

2.1.2 Alineaciones

La alineación entre dos secuencias de proteínas consiste simplemente en realizar una serie de translaciones, inserciones y deleciones en el vector de secuencia de una de ellas para satisfacer un criterio de emparejamiento determinado. Usualmente, este criterio consiste en maximizar la identidad secuencial entre ambas secuencias, aunque se puede combinar con otros criterios de similar índole. Si la alineación implica dos únicas secuencias como son la secuencia problema y la secuencia plantilla, la alineación que proporciona la máxima identidad secuencial no resulta excesivamente complicada de encontrar. Sin embargo, en algunas ocasiones la alineación con mayor identidad secuencial no es necesariamente la más adecuada pa-

ra realizar la homología con el método elegido. Hay que tener presente que los algoritmos pensados para buscar una máxima identidad secuencial en una alineación se rigen por un criterio puramente estadístico [16] y que la similitud entre dos proteínas homólogas está basada en criterios estructurales y funcionales. Normalmente, la identidad secuencial en aquellos residuos conservados en una familia de proteínas permanecerá en la alineación de máxima identidad secuencial, pero no siempre ha de ocurrir así.

Un factor muy importante a la hora de estudiar la calidad de una alineación orientada a cálculos de modelización por homología es la situación de las posibles inserciones y deleciones que surgen de la alineación. Estos huecos suponen de por sí alteraciones en la estructura natural de cualquier proteína y por lo tanto se ha de intentar que sean pequeños y poco numerosos. Por ello en los algoritmos más típicos de alineación de secuencias se asocia la aparición de los mismos a funciones de penalización [17]. Por otra parte, si es inevitable la aparición de los huecos, es importante situarlos en aquellas regiones de la proteína que presenten una mayor flexibilidad estructural para adaptarse a la nueva situación anómala que fuerza su presencia. Un hueco presente en una zona de estructura secundaria definida como puede ser una hoja β o una α-hélice tenderá, por la perturbación que en si contiene, a producir la deformación de la misma. Sin embargo, el mismo hueco situado en un giro tiene muchas más posibilidades de no ocasionar una gran alteración en la estructura global de la proteína [126].

Poplar	--IDVLLGADDGSLAFVPSEFSISPGEKIVFKNNAGFPHNIVFDEDSIP
Synechocistis	ANATVKMGSDSGALVFEPSTVTIKAGEEVKWVNNKLSPHNIVFAAD---
French	--LEVLLGSGDGSLVFVPSEFSVPSGEKIVFKNNAGFPHNVVFDEDEIP
Parsley	--AEVKLGSDDGGLVFSPSSFTVAAGEKITFKNNAGFPHNIVFDEDEVP

Poplar	MSGVDASKISMSEEDLLNAKGETFEVALSNKGEYSFYCSPHQGAGMVGKVT
Synechocistis	LGVDADTAAKLSHKGLAFAAGESFTSTFTEPGTYTYYCEPHRGAGMVGKVV
French	MAGVDAVKISMPEEELLNAPGETYVVTLDTKGTYSFYCSPHQGAGMVGKVT
Parsley	PAGVNAEKISQEYLNGAGETYEVTLTEKGTYKFYCEPHAGAGMKGEVTVN

Tabla 2.2: Alineaciones entre diferentes plastocianinas utilizadas como plantillas. Las identidades secuenciales en las diferentes plastocianinas respecto a la $Synechocistys$ son 41.8 % ($Poplar$), 40.8 % $French$) y 52 % ($Parsley$).

Otro aspecto a tener en cuenta es la conservación del tipo de aminoácido en aquellos casos en que la identidad se pierde. Si es posible alinear tam-

bién basicidad y acidez en los residuos, la homología obtenida de este modo será bastante más completa que si olvidamos este tipo de factores.

El método convencional de localizar alineaciones de secuencias con un alto porcentaje de identidad secuencial es el descrito por Needlemann y Wunch [4]. Este método alinea dos secuencias maximizando el número de pares de residuos alineados. Para ello se aplica una función alineación que consiste en contabilizar todos los pares de residuos alineados y agregar una penalización opcional por la introducción de huecos en una de las secuencias.

La alineación elegida para realizar los cálculos de modelización por homología debería observar, por lo tanto, los siguientes rasgos:

- La alineación debería contener el mínimo posible número de huecos en ambas secuencias manteniendo un grado de identidad secuencial adecuado.
- Los huecos inevitables que aparezcan deberían estar localizados en zonas de poca rigidez estructural, es decir, alejados de zonas de estructura secundaria definida.
- Los huecos no deberían exceder una longitud de 3 residuos ya que dicho tamaño supone demasiada deformación estructural en las coordenadas a trasladar.
- En la medida de lo posible, se debería intentar conservar la alineación entre residuos de características similares como puedan ser acidez y basicidad o carácter aromático, entre otras.

2.1.3 Diferentes métodos de modelizar por homología

Todos los métodos de modelización mediante homología parten de una alineación entre la secuencia de la proteína problema y la secuencia de la proteína plantilla. La principal diferencia entre los diversos métodos que se pueden aplicar a la modelización por homología reside en el modo en que es calculado el modelo de estructura tridimensional a partir de una determinada alineación.

El método más antiguo y uno de los más utilizados es el acoplamiento de la estructura a un modelo rígido [18]. Dicho método construye el modelo partiendo de una pequeña cantidad de zonas internas conservadas,

giros y cadenas laterales, que son obtenidas de diferentes estructuras relacionadas con la proteína problema. Este entramado de regiones proviene de compatibilizar estos cuerpos rígidos en el marco estructural de la proteína definido por las posiciones de los carbonos α en las zonas conservadas del plegamiento. Las regiones rígidas tomadas de diferentes proteínas de estructura conocida relacionadas con el problema se superponen mediante un método de mínimos cuadrados. La secuencia de la proteína problema es alineada entonces con el consenso de secuencias de las proteínas de las que se han tomado los segmentos rígidos y a partir de la alineación y los segmentos rígidos se construye la proteína problema. Este método tiene una fiabilidad aceptable con alineaciones cuya identidad secuencial es mayor de 40 %. Sin embargo, la exactitud de la predicción decrece rápidamente al disminuir la identidad secuencial.

Otra familia de métodos, llamada genéricamente modelización por ajuste de segmentos [19], se basa en utilizar las posiciones aproximadas de los átomos conservados de las plantillas para calcular las coordenadas del resto de átomos. Esto se consigue mediante el uso de una base de datos de pequeños segmentos de estructuras de proteínas, clasificados mediante criterios energéticos, geométricos o una combinación de ambos. Las posiciones atómicas que no han podido ser establecidas a partir de la base de datos se generan mediante técnicas de búsqueda conformacional o mediante minimizaciones y dinámicas moleculares de dichos fragmentos en el marco de la estructura ya determinada, según el tamaño de los segmentos implicados.

El tercer grupo de métodos de modelización por homología es el que vamos a usar en el presente estudio. Estos métodos son conocidos genéricamente como modelización mediante safisfacción de restricciones espaciales. Como el propio nombre indica, el método consiste en obtener un conjunto de restricciones espaciales [20] de la proteína plantilla utilizando la alineación con la proteína problema. En estos métodos se puede utilizar información de muy variado origen sobre la proteína problema. Por ello es quizás la más prometedora de las tres posibilidades planteadas para la modelización por homología y por esta razón es la que se ha usado en este estudio.

Por su naturaleza, los métodos de modelización por homología mediante satisfacción de restricciones espaciales pueden ser aplicados según diferentes modalidades de cálculo. La aproximación más inmediata para la determinación de la estructura es la de la geometría de distancias para

construir todas las posiciones atómicas a partir de las restricciones espaciales derivadas de la proteína plantilla [5]. Otros métodos han utilizado la dinámica molecular para satisfacer las restricciones espaciales en la cadena principal de la proteína. En otros casos se han construido las coordenadas completas del esqueleto carbonado de la proteína a partir de las posiciones de los C_α de la plantilla. En dichos casos se puede utilizar la posición de los C_α combinados con los algoritmos de construcción de cadenas laterales y giros [8].

En este estudio se han utilizado dos aproximaciones al mismo método que han permitido evaluar tanto la calidad del mismo como la autoconsistencia de las aproximaciones empleadas. En primer lugar se ha utilizado una interesante aproximación que combina la geometría de distancias y la dinámica molecular con una función de densidad de probabilidad relacionada con una base de datos de alineaciones. Esta aproximación, utilizada en el programa MODELLER [14], ha sido probada en diferentes estudios estructurales y ha producido resultados satisfactorios y se puede considerar una aproximación consolidada. También se ha utilizado una aproximación de dinámica molecular combinada con búsqueda conformacional utilizando el programa CONGEN [22, 26] y aplicando conocidos protocolos de templado simulado restringido [48] que también ha sido comprobada en diferentes moléculas y que ha producido estructuras depositadas en el PDB [9].

2.1.4 Modelización por homología mediante satisfacción de restricciones espaciales

El método de modelización por homología mediante la satisfacción de restricciones espaciales requiere como paso inicial la obtención de restricciones espaciales que aplicar a la simulación molecular de la proteína problema en cuestión. Estas restricciones espaciales pueden ser de diferente tipo y origen. En su mayoría es conveniente que procedan de la proteína plantilla utilizada en la homología pero no es necesario que todas provengan de ella. Restricciones espaciales pueden ser una gran variedad de restricciones, desde las típicas restricciones de distancias equivalente a los NOEs de Resonancia Magnética Nuclear hasta restricciones tan difíciles de evaluar como la accesibilidad al disolvente [20]. En este estudio se han utilizado restricciones de distancias y diedros con ambos protocolos. Sin embargo, en el protocolo del programa MODELLER se incluyen restricciones de origen estadístico que permiten acelerar el cálculo del modelo final en gran

medida.

El paso siguiente a establecer que restricciones van a ser aplicadas a nuestro cálculo consiste en definir la función objetivo en la que se van a introducir dichas restricciones. Esta función objetivo será posteriormente optimizada y con ello obtendremos un modelo que satisfaga las restricciones espaciales derivadas de la plantilla homóloga a la estructura problema. La función objetivo puede tener diferentes formas y componentes pero básicamente ha de incluir la información topológica correspondiente a los aminoácidos constitutivos de la proteína y la información espacial derivada de las restricciones extraida de la estructura plantilla.

La optimización de la función objetivo construida con toda la información disponible es el siguiente paso en el proceso de obtención del modelo. Para ello se pueden utilizar diferentes métodos como puedan ser la geometría de distancias, la dinámica molecular o el templado simulado. Según la función objetivo planteada será más conveniente uno u otro método. En nuestro estudio se ha utilizado en ambos casos el templado simulado [26, 14]. Este método consiste en aplicar progresivamente temperatura a una estructura inicial utilizando una función objetivo en la que todos los factores han sido reescalados hasta alcanzar una alta temperatura. De este modo los átomos tienen casi total libertad de movimiento y al mismo tiempo una energía cinética que les permite superar barreras energéticas de otro modo insuperables. Una vez alcanzada la temperatura requerida, se aumenta progresivamente el peso de los factores de la función objetivo hasta alcanzar los valores de equilibrio dando prioridad a las restricciones espaciales introducidas. Cuando el sistema se equilibra en las condiciones de equilibrio a una alta temperatura se disminuye esta progresivamente hasta alcanzar la temperatura ambiente en la que se empiezan a recoger estructuras. Este método permite superar altas barreras de potencial y explorar el espacio conformacional de un modo más exhaustivo que con una simple dinámica molecular a temperatura ambiente.

Las estructuras obtenidas con este método se analizan en función del número de violaciones de las restricciones, de la energía de las mismas y de las interacciones de van der Waals presentes en la estructura final.

2.1.5 Las Plastocianinas

La plastocianina es una pequeña proteína de cobre cuya función en la cadena fotosintética es la catálisis de la transferencia de un electrón desde el

citocromo f en el complejo b_6f a el P700$^+$ en el fotosistema I [29]. En base a sus propiedades espectroscópicas se suele decir que es una proteína azul (ϵ $\sim$ 4700 cm^{-1} a 600 nm). La estructura de la plastocianina de $Poplar$ fué la primera determinada en 1978 [10] . La proteína está compuesta de ocho hojas β formando un barril β con un átomo de cobre enlazado por las cadenas laterales de dos histidinas, una cisteína y una metionina (figura 2.2).La superficie en esta región está compuesta casi exclusivamente por residuos hidrofóbicos y a dicha región se le suele llamar "parche hidrofóbico"[29]. Otra región en la que se concentra cierta carga negativa debida a las cadenas laterales desprotonadas de glutámico y aspártico se suele denominar "parche negativo". Se han identificado dos vías de transferencia electrónica desde y hacia el átomo de cobre en los trabajos originales [10]. Una vía adyacente a la HIS87, que es el único ligando del cobre accesible al disolvente, y una vía remota a través de la TYR83 en la parte este de la molécula [126].

Se puede considerar que las plastocianinas tienen tres funciones principales. La primera consiste en mantener el sitio del cobre de modo que el potencial de reducción de la proteína permanezca situado entre el de su reductor fisiológico (el citocromo f) y su oxidante (P700$^+$) [29]. La segunda consiste en proporcionar lugares de enlace específicos a sus complejantes fisiológicos en la reacción de transferencia. Por último, debe poseer la estructuración necesaria para la correcta transferencia electrónica controlada desde y hacia el sitio donde se encuentra coordinado el cobre.

Desde un punto de vista evolutivo es interesante hacer constar que, aunque la plastocianina es la única transportadora redox en las plantas superiores, puede ser reemplazada por el citocromo c_6 en un número determinado de cianobacterias y algas verdes. Hasta 1996 no se ha obtenido la estructura de ninguna plastocianina de organismo procariótico. En este año se obtuvo la estructura de la plastocianina de $Anabaena\ variabilis$ [11]. Dicha estructura se diferencia de las de otras algas y plantas superiores en varios aspectos. En primer lugar, la alineación de secuencias con el resto de plastocianinas muestra que el número de aminoácidos conservados disminuye notablemente respecto a las Plastocianinas de plantas superiores (ver tabla 2.2). Por otra parte, la plastocianina de $Anabaena$ es una de las de secuencia más larga que se conocen. En tercer lugar, se ha encontrado en dicha estructura que la plastocianina de $Anabaena$ tiene una constante de intercambio mucho más alta que el resto de plastocianinas debido a la ausencia de aminoácidos cargados negativamente en el lugar de acceso remoto al Cu [11].

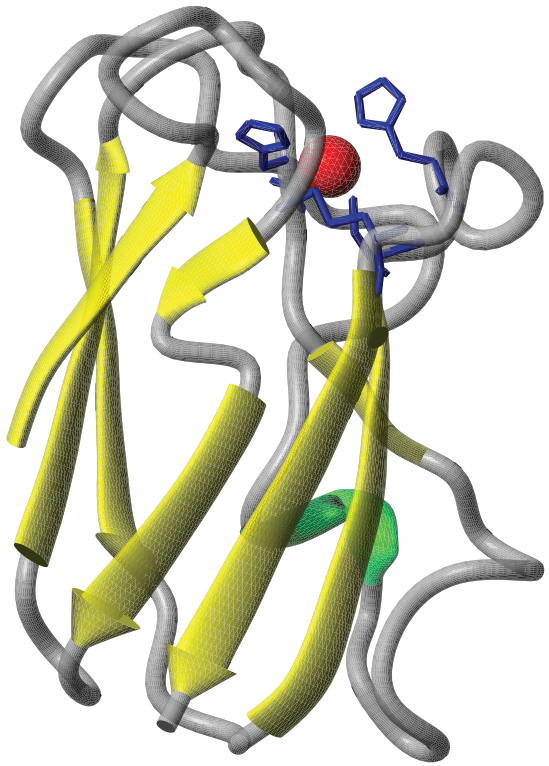


Figura 2.2: Estructura de la Plastocianina *Poplar* representativa de la estructura general de las Plastocianinas mostrando los residuos coordinados al átomo de Cobre. El lugar del Cobre está situado al "norte" de la proteína. El plegamiento y la ordenación de las cadenas laterales en torno al átomo de Cobre se conserva de un modo casi idéntico en todas las Plastocianinas.

Todo esto unido a las diferencias en mecanismo de reacción [50] desarrolladas en el punto siguiente hacen muy interesante el estudio de una plastocianina como la *Synechocistys* que, conservando una gran cantidad de residuos respecto al resto de plastocianinas, sin embargo presenta un comportamiento más similar a la de la plastocianina *Anabaena*. Asi pues, el estudio estructural de la Plastocianina *Synechocistys* puede proporcionar nuevas luces sobre las diferencias observadas entre las plastocianinas de plantas superiores y las de cianobacterias.

2.1.6 Cinética de reacción de las Plastocianinas

La interacción entre la plastocianina y el fotosistema I ha sido ampliamente estudiado en una gran variedad de organismos [6, 7]. Experimentos de cinética de alta velocidad han permitido comprobar la existencia de dos diferentes fases de reducción en las plantas superiores y algunas algas eucarióticas [50]. Existe una fase cinética llamada rápida ($t_{1/2}$ = 12 μs en la espinaca) cuya aportación a la amplitud total parece ser de un 40 % y que ha sido interpretada como correspondiente a la transferencia de electrones desde la plastocianina en situación estrechamente enlazada al fotosistema I hacia el P700 fotooxidado [12]. También se ha comprobado la existencia de una segunda fase cinética ($t_{1/2}$ = 110 μs) que puede implicar otro paso de reacción asociado al segundo lugar de enlace de la plastocianina en el centro de reacción. Esta explicación es susceptible de ser comprobada bajo la luz que puede aportar un modelo tridimensional de la estructura de la plastocianina y su distribución electrostática.

Algunos autores han asociado esta última fase a la reorientación de algunas cadenas laterales para optimizar la transferencia electrónica y previa a la misma [12]. La transferencia electrónica entre las dos proteínas con cargas opuestas implicadas en esta reacción en las células eucarióticas sólo parece alcanzarse cuando ocurre un reajuste adicional dentro del propio complejo de colisión formado.

Recientes estudios cinéticos llevados a cabo mediante análisis de laser-flash [13] han proporcionados interesantes resultados sobre el posible mecanismo de transferencia electrónica en las plastocianinas y el origen evolutivo del mismo. El estudio comparativo de la cinética de organismos evolutivamente diferentes ha permitido concluir que, cuando se realiza la transferencia hacia el fotosistema I de espinaca, la llamada fase rápida solo se observa con las proteínas ácidas de plantas superiores o algas verdes pero

Organismo	Mecanismo	K_{14}	K_{12}	K_{24}	K_{23}	K_{34}
Poplar	III		$4.2 \cdot 10^8$		$1.3 \cdot 10^4$	$6.3 \cdot 10^4$
Monoraphidium	III		$4.5 \cdot 10^4$		$1.0 \cdot 10^4$	$3.9 \cdot 10^4$
Annabaenna	II		$1.7 \cdot 10^6$	$0.7 \cdot 10^3$		
Synechocistys	I	$2.1 \cdot 10^6$				

Tabla 2.3: Tabla de constantes cinéticas para las reacciones de las plastocianinas de los diferentes organismos frente al PSI de espinaca. Los mecanismo suponen colisión orientada en el caso I, mecanismo mínimo de dos pasos en el caso II con formación de complejo intermedio y transferencia electrónica, y reestructuración del complejo como paso previo a la transferencia en el mecanismo III [13].

no con las proteínas básicas o moderadamente ácidas de las cianobacterias. En el caso concreto de la plastocianina *Synechocistys*, el mecanismo de reacción propuesto para la transferencia electrónica es simplemente mediante colisión, sin mediar en ningun momento un complejo intermedio de transición.

Esta diferencia en los mecanismos de reacción puede provenir de diferencias estructurales tanto en las regiones hidrofóbicas como hidrofílicas, aunque la posible formación o no de un complejo intermedio de transición hace necesario el análisis de potencial electrostático de las plastocianinas con diferentes mecanismos. Para arrojar nuevas luces sobre las posibles causas de dichas diferencias, se ha realizado un estudio estructural teórico mediante modelización por homología y un estudio electrostático sobre los resultados obtenidos.

2.2 Materiales y Métodos

Todos los cálculos realizados han sido llevados a cabo en estaciones de trabajo Silicon Graphics. En concreto la máquina utilizada ha sido una Power Indigo 2, con un procesador R8000 y 128 Megabytes de memoria principal. Los programas utilizados han sido obtenidos mediante FTP de los correspondientes servidores o han sido desarrollados en nuestro propio laboratorio y son de acceso gratuito para uso académico aunque el MODELLER [14] puede ser adquirido como modulo independiente del paquete de utilidades para modelización molecular Quanta de Molecular Simulations Inc [47]. El programa CONGEN [22] puede ser adquirido gratuitamente poniéndose en contacto con su autor Dr. Robert E. Bruccoleri. Las direcciones de ftp

desde las cuales se pueden obtener los programas y su petición de licencia académica son las siguientes:

- MODELLER: ftp://guitar.rockefeller.edu/pub/modeller
- CONGEN: Pedir al autor mediante correo electrónico.
- PDBSTAT, homología: ftp://hyper.quifis.uv.es/software

Para los dibujos y las gráficas se han utilizado varios programas de visualización molecular como son el Rasmol 2.6 [163] y el MOLMOL 2.5 [31] también de libre distribución. Las gráficas se han realizado con la herramienta de la compañía de programas informáticos de libre distribución GNU gnuplot [160]. El tratamiento de datos se ha realizado en su mayoría con comandos estándar del sistema UNIX (*awk, sed, grep*, etc.) y con el programa desarrollado en nuestro laboratorio PDBSTAT (sin publicar)..

2.2.1 El programa MODELLER

El programa MODELLER modela estructuras tridimensionales de proteínas mediante la satisfacción de restricciones espaciales. El programa necesita como información inicial las restricciones en la estructura espacial de la secuencia de aminoácidos a ser modelada. El resultado es una estructura 3D que satisface dichas restricciones en la medida de lo posible al tiempo que cumple los requisitos impuestos por la topología habitual para los aminoácidos extraida del programa CHARMM [24].

El uso más habitual del programa es el de modelización por homología. El programa permite realizar la modelización de un modo casi automático. Para ello el usuario proporciona una alineación de la secuencia a ser modelada con secuencias de estructura conocida y el MODELLER automáticamente genera las restricciones espaciales derivadas de la alineación, las aplica a la secuencia problema y genera un modelo 3D de la proteína de estructura desconocida. Sin embargo, el MODELLER es capaz de trabajar con restricciones de muy diferente origen como puedan ser restricciones derivadas de experimentos de Resonancia Magnética Nuclear, reglas de empaquetamiento de estructura secundaria, espectroscopía de fluorescencia, etc. Las restricciones que puede manejar el MODELLER también son de muy diferente origen. Pueden ser distancias, ángulos, ángulos diedros y pares de ángulos diedros definidos por átomos o pseudoátomos. El modelo 3D obtenido surge de la optimización de una función de densidad

de probabilidad (*pdf*) en el espacio cartesiano utilizando técnicas de gradientes conjugados y dinámica molecular con templado simulado.

El protocolo general del MODELLER es bastante similar a cualquier proceso de templado simulado con la salvedad de que la función a optimizar no es una función puramente energética. Esta función contiene además de la típica información de distancias de enlace, ángulos de enlace y diedros, información probabilística extraida de una base de datos interna del MODELLER sobre distancias entre los átomos de N y O en la cadena principal, accesibilidad al disolvente, la clase de cadena lateral y su orientación, restricciones estereoquímicas, etc [14]. De este modo, examinando en la base de datos alineaciones con estructuras ya determinadas, se incluye información estructural adicional a la disponible en la propia plantilla utilizada y se acelera la convergencia.

El primer paso consiste en extraer las restricciones espaciales de la estructura plantilla. En el MODELLER las restricciones espaciales incluyen distancias entre átomos de la cadena principal, distancias entre átomos de cadenas laterales, ángulos diedros y pares de ángulos diedros. Sin embargo, el número de restricciones que por defecto extrae el MODELLER de la plantilla es muy elevado y produce modelos 3D con muy poca dispersión con lo que la exploración del espacio conformacional accesible no parece muy exhaustiva. Para comprobar el nivel de restricciones necesario para obtener un resultado homólogo y al mismo tiempo una dispersión que permita evaluar la búsqueda de soluciones, se han realizado cálculos variando el protocolo de extracción de restricciones [126].

La estrategia de optimización utilizada en el programa MODELLER ha sido bastante similar a cualquier estrategia de templado simulado. Se calcularon 5 modelos de homología para cada nivel de restricciones y plantilla diferente. En el modelo inicial a refinar, se utilizaron las topologías de aminoácidos sin hidrógenos del CHARMM y se transladaron las coordenadas de los átomos homólogos de la plantilla aplicando una desviación aleatoria con un máximo de 10 Å [126]. Así, tenemos un modelo de la proteína con sólo átomos pesados compatible con el protocolo del MODELLER, ya que el método de extracción de restricciones del mismo solo necesita átomos pesados. A este modelo inicial se le aplican las restricciones de un modo progresivo al tiempo que se realizan cálculos de minimización mediante el método de gradientes conjugados. Al mismo tiempo que se aumenta el número y alcance de las restricciones incluídas en el cálculo, se reescalan las interacciones de van der Waals y no enlazantes desde un valor inicial

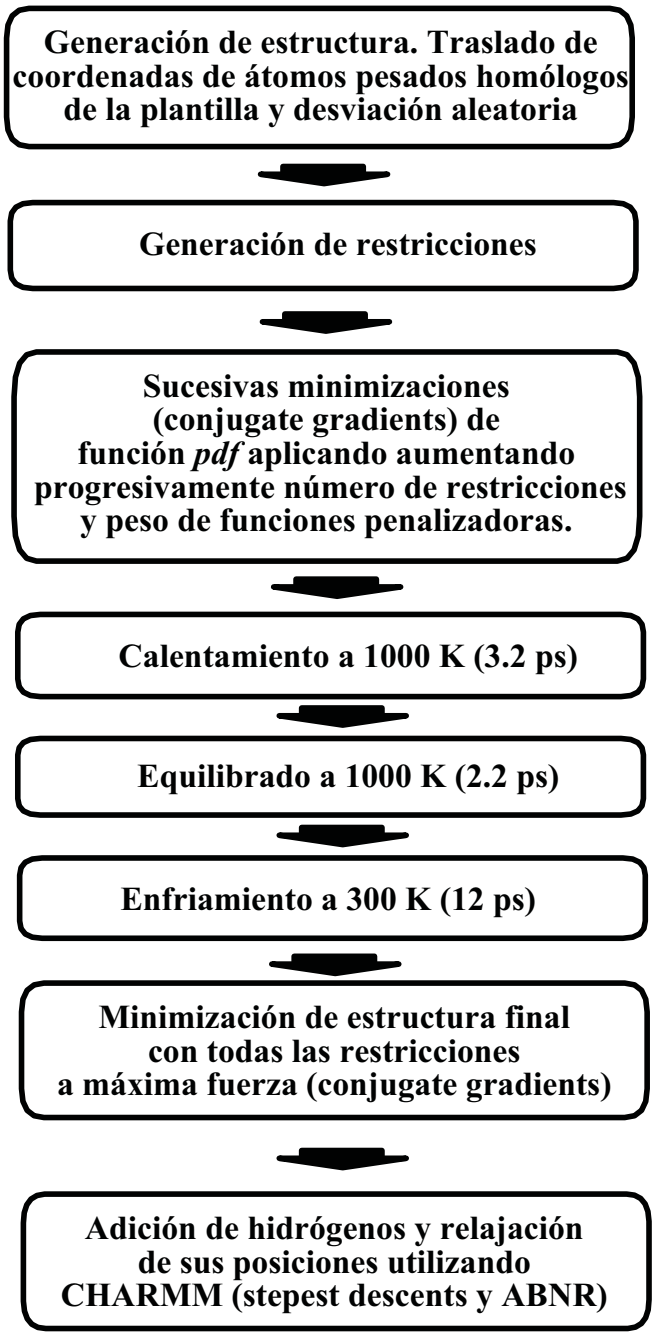


Figura 2.3: Protocolo general de cálculo de modelización por homología en el programa MODELLER.

muy pequeño hasta su valor normal. De este modo las barreras energéticas de la hipersuperficie de potencial debidas a estas interacciones son más fácilmente superables por el sistema. Cuando todas las restricciones han sido añadidas y todas las interacciones han alcanzado su valor normal, se realiza una minimización más profunda para relajar todos aquellos *puntos calientes* que el proceso de adición de restricciones ha podido generar. Entre cada dos minimizaciones tan solo se refinaron los contactos de van der Waals, excepto en el caso de la *Anabaena* en el cual, por tener la plantilla numerosas violaciones de tipo estereoquímico se realizaron refinamientos adicionales en este sentido. A continuación, se llevaron a cabo cálculos de dinámica molecular con todas las restricciones en un ciclo que comprendía una etapa de calentamiento progresivo hasta 1000 K en 3.2 ps, una etapa de equilibrado a dicha temperatura de 2.2 ps y un lento enfriamiento hasta 300 K en 12 ps. Es interesante resaltar que en dicho ciclo de dinámica molecular, aunque todas las restricciones han sido incluídas, no lo han sido de igual modo. El MODELLER tiene la posibilidad de trabajar con las llamadas restricciones dinámicas. Dichas restricciones son variables en el tiempo de dinámica dependiendo de la capacidad del sistema para satisfacerlas sin un gran recargo energético. Así, las restricciones que representen un recargo energético excesivo son incluídas en este grupo y reescaladas para no distorsionar el efecto del resto de restricciones.

Utilizando este protocolo se han obtenido 5 modelos de homología sin hidrógenos para cada nivel de restricciones y cada plantilla. A estos modelos se les han adicionado mediante el programa de cálculo CHARMM los hidrógenos y se han relajado sus posiciones mediante una minimización implicando únicamente esos átomos. La energía final de la molécula completa ha sido calculada mediante CHARMM. De este modo, la velocidad del cálculo ha sido mejorada manteniendo la calidad de los resultados.

2.2.2 Homología utilizando el programa CONGEN

El proceso es automático utilizando programas desarrollados en nuestro laboratorio [21] CONGEN (CONformational GENerator) es un programa de gran flexibilidad que utilizando funciones de energía empíricas y técnicas de búsqueda conformacional puede modelar estructuras tridimensionales de proteínas. El programa permite leer o construir estructuras de proteínas, realizar minimizaciones de energía, cálculos de dinámica molecular, búsquedas conformacionales sistemáticas, análisis de propiedades dinámicas, etc [23]. El programa es un descendiente del programa CHARMM [24] en

su versión 1.6 al que se le ha añadido la capacidad de búsqueda conformacional entre otras técnicas [22, 23].

El método utilizado en los cálculos de modelización por homología utilizando CONGEN se puede considerar un híbrido entre templado simulado utilizando dinámica molecular restringida y búsqueda conformacional sistemática [26]. La selección de átomos homólogos entre proteínas en la alineación es similar a la del MODELLER aunque presenta modificaciones de gran interés. Sólo los átomos pesados pueden ser considerados homólogos. Si los residuos son idénticos, todos los átomos pesados del residuo son considerados homólogos. Si difieren, sólo aquellos átomos pesados con el mismo tipo de átomo en el campo de fuerzas e idéntica hibridación son definidos como homólogos [26].

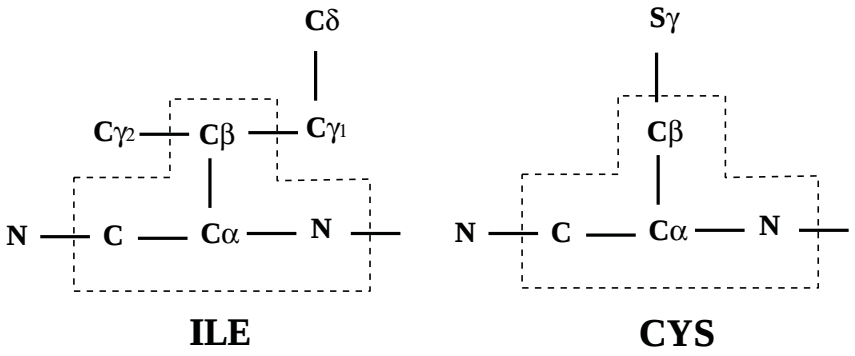


Figura 2.4: Representación de la selección de átomos homólogos genérica utilizada en el protocolo para el programa CONGEN.

A continuación, las coordenadas atómicas de la estructura plantilla son utilizadas para derivar restricciones de distancias entre los átomos definidos como homólogos. Un subconjunto de estas restricciones de distancia son seleccionadas aleatoriamente y utilizadas para crear restricciones de homología con límites superior e inferior. El número de restricciones seleccionadas es equivalente a las restricciones de RMN necesarias para obtener una estructura de alta resolución. Los límites superior e inferior de las restricciones se calculan añadiendo un 10 % a la distancia exacta en la estructura plantilla [48].

El paso siguiente es aplicar a la secuencia problema una dinámica molecular restringida a alta temperatura según el protocolo de templado simulado. La función objetivo contiene la energía conformacional propia de la molécula y la energía residual correspondiente a las violaciones de

las restricciones de homología según implementación especial y propia de CONGEN. La primera viene definida por los parámetros y topologías del campo de fuerzas CHARMM utilizando una constante dieléctrica dependiente de la distancia igual a $1 \cdot r$ para simular el disolvente. Las estructuras de partida son de conformación totalmente extendida y cada cálculo difiere únicamente en las velocidades iniciales aplicadas a los átomos en la dinámica definidas por una semilla aleatoria [48].

El proceso de templado comprende dos etapas fundamentales: el templado de aportaciones de cada interacción individual y el templado de temperatura. El templado de aportaciones de cada interacción individual no es otra cosa que ir reescalando gradualmente las contribuciones individuales del término especial de restricciones en la función objetivo desde un valor muy pequeño hasta el valor final de 100 Kcal / mol en sucesivas etapas de dinámica molecular a 1000 K. A continuación, se realiza el templado de temperatura que consiste en ir enfriando progresivamente el sistema desde los 1000 K a los que se ha realizado la primera etapa hasta los 300 K de temperatura ambiente en incrementos que van desde 100 K hasta 5 K. Los incrementos decrecen progresivamente a medida que se aproximan a la temperatura final para simular un decaimiento exponencial y favorecer una distribución entrópica adecuada. Una vez alcanzada la temperatura final, se equilibra el sistema en la misma mediante una dinámica molecular restringida de 5 ps. El procedimiento completo consta de 21 pasos de duración variable (10 pasos de 4 ps cada uno y 11 pasos de 3 ps cada uno) para el primer templado y 12 pasos de 1 ps cada uno en el segundo templado. La última etapa consiste en 10 ps de dinámica molecular restringida a 300 K. La estructura final procede de recoger las estructuras de los ultimos 3 ps de dinámica molecular y promediarlas. A esta estructura se le aplica una minimización de energía con restricciones. A lo largo de todo el proceso se mantiene plano y en conformación $trans$ el enlace peptídico de todos los residuos mediante la aplicación de una función de restricción armónica con fondo plano y una constante de fuerza de 200 Kcal / mol-grado y mínimo en $180^o \pm 2^o$.

La búsqueda conformacional se aplica en la resolución de aquellos tramos de la secuencia con deleciones y en la determinación de cadenas laterales. La búsqueda es realizada de modo iterativo sobre todos los ángulos χ de cadena lateral que implican átomos no homólogos hasta que la energía converja. A los tramos en que hay deleciones se aplica como restricción adicional la distancia extremo-extremo del segmento condicionada por los residuos que rodean a la deleción. El método de búsqueda conformacio-

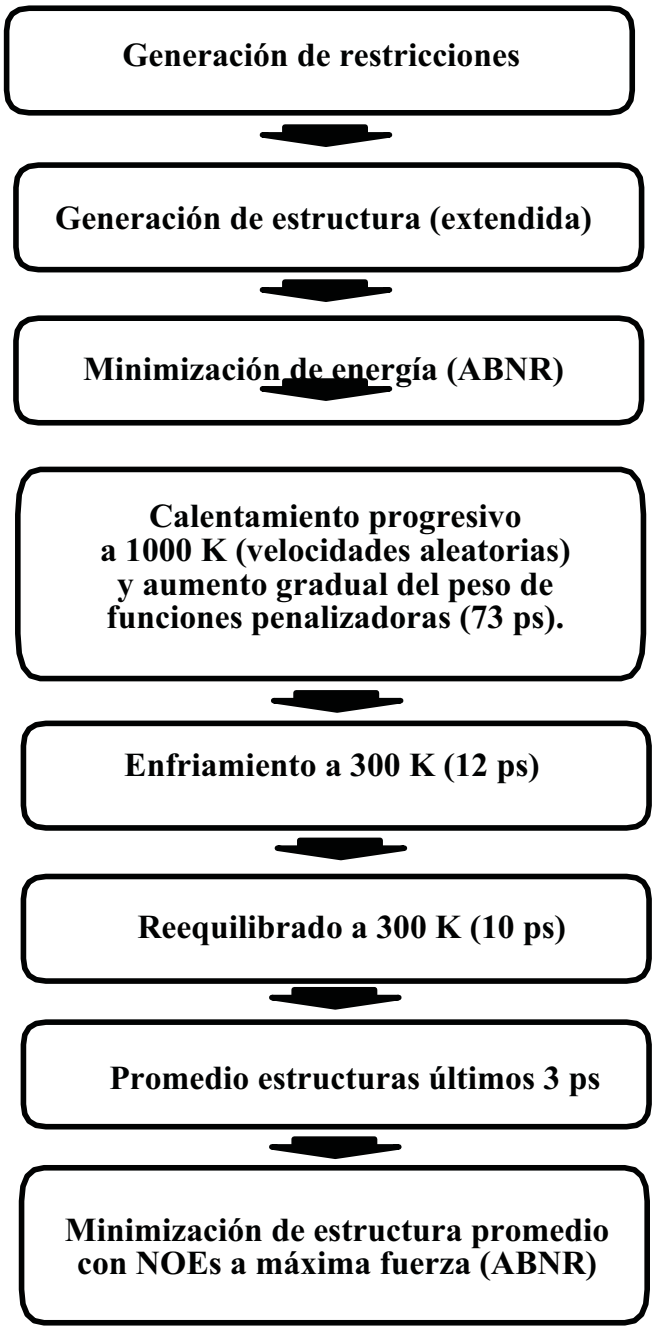


Figura 2.5: Protocolo utilizado para la homología con el programa CONGEN para la modelización por homología [26].

nal contenido en el CONGEN ha sido ensayado para pequeños péptidos obteniéndose resultados acordes con la información experimental [27, 28].

Este protocolo requiere bastante más tiempo de computación que el del apartado anterior. Por ello, sólo se ha utilizado la alineación que se ha comprobado óptima para la modelización mediante el análisis de alineaciones realizado con el MODELLER.

2.2.3 Estructuras plantilla

Se han utilizado cuatro estructuras plantilla de diferentes plastocianinas con una identidad secuencial en las diferentes alineaciones con respecto a la secuencia de *Synechocistys* de más del 35 % en todos los casos. Las plantillas utilizadas, además de proceder de determinaciones estructurales por diferentes técnicas, también tienen diferente nivel de resolución. Así, mientras la estructura de la plastocianina *Poplar* ha sido determinada a una resolución de 1.8 Å y mediante técnicas de Rayos X [10], el resto han sido determinadas por RMN y presentan una resolución aceptable aunque no tan alta. La única excepción es la plastocianina *Anabaena* que presenta una resolución muy baja y que por ello ha sido utilizada en un solo nivel de restricciones únicamente [11].

Con el programa MODELLER se han utilizado todas las plantillas y se han ensayado varios niveles de restricción variando la distancia máxima para tomar restricciones de la estructura plantilla. Se han probado en las estructuras plantilla de mejor resolución tres niveles diferentes tomando restricciones menores de 6, 8 y 10 Å respectivamente [126]. En el resto de estructuras plantilla solo se han tomado los niveles extremos, es decir, 6 y 10 Å, debido a que estos niveles proporcionan por una parte, la libertad suficiente para que el sistema se acomode a sus propias restricciones y por otra, el nivel de restricción suficiente para asegurar el plegado correcto de la proteína y la convergencia de los cálculos. Además, para la plantilla de más resolución, se han realizado los cálculos sobre dos alineaciones diferentes con un nivel similar de identidad secuencial pero diferente localización y tamaño de huecos para comprobar la influencia de los mismos.

Con el programa CONGEN, debido a que el protocolo utilizado requiere un mayor tiempo de computación, solo se ha ensayado la plantilla y alineación que ha proporcionado mejores resultados con el programa MODELLER. De este modo, se ha intentado comprobar la autoconsistencia de la aproximación de homología y de los dos protocolos empleados.

2.3 Resultados y discusión

2.3.1 Estructuras obtenidas

Se han obtenido 55 estructuras con el programa MODELLER y 10 estructuras con el programa CONGEN distribuidas como se puede apreciar en la tabla 2.5. Las energías obtenidas se encuentran dentro de los límites esperados para estructuras de calidad aceptable ($-10 \cdot Kcal \cdot mol^{-1} residuo^{-1}$) [48]. Dichas estructuras presentan una diversidad importante en las energías obtenidas según alineaciones, plantillas y nivel de restricciones [126]. Sin embargo, como se puede observar en la tabla 2.4, el plegamiento global de los modelos de $Synechocistys$ de diferentes plantillas, alineaciones, nivel de restricciones y protocolos es bastante similar. Esto es fácilmente explicable teniendo en cuenta el hecho de que la identidad secuencial es en todos los casos superior al 40 % y que los átomos de la cadena principal son, en la mayoría de los casos, homólogos sean o no idénticos los residuos, por lo que se pueden tomar restricciones de los mismos.

$\alpha - Helix$		Align. and Restr.	$\beta - Sheet$							
53-55	87-89	**Poplar I** $D \leq 6.0$ **Å**	3-7	16-17	20-23	27-33	42-43	68-73	77-82	92-97
	87-89	**Poplar I** $D \leq 8.0$ **Å**	4-7	16-17	20-21	28-33	42-43	68-72	77-82	92-97
	87-89	**Poplar I** $D \leq 10.0$ **Å**	3-7	16-17	20-23	27-33	42-43	68-73	77-82	92-97
51-54	87-89	**Poplar II** $D \leq 6.0$ **Å**	3-7	16-17	20-23	27-33	42-43	68-73	77-82	92-97
51-54	84-89	**Poplar II** $D \leq 8.0$ **Å**	3-7	16-17	20-23	27-33	42-43	68-73	77-82	92-97
51-54	84-87	**Poplar II** $D \leq 10.0$ **Å**	3-7	16-17	20-23	27-33	42-43	68-73	77-82	92-97
51-54	84-89	**French** $D \leq 10.0$ **Å**	4-7	16-17	22-23	27-33	42-43	68-73	77-82	92-97
51-54	84-89	**Parsley** $D \leq 10.0$ **Å**	5-7	16-17	20-23	29-33	41-43	68,71	77-82	92-97
51-54	87-89	**Anabaena** $D \leq 10.0$ **Å**	4-6		21-23	29-33	42-43		77-82	92-97
52-54	84-87	**Poplar X-Rays**	4-7	16-17	20-23	27-33	42-43	68-73	77-82	92-97
52-56	84-86	**Triple Mutante**	3-7	16-17	20-23	28-33	41-43	68-72	77-82	92-97

Tabla 2.4: Esquema de estructura secundaria de los modelos de plastocianina $Synechocistys$ obtenidos por modelización y de estructuras experimentales de otras plastocianinas.

El número de restricciones de distancia y diedros con -ln(pdf) superior a 3, que es el valor estándar recomendado por los autores del programa MODELLER, puede ser utilizado para evaluar la calidad de los cálculos de homología realizados con el mismo. En el caso de nuestros cálculos, al haber modificado el protocolo estándar del mismo, cabe esperar una desviación ligeramente superior. Para los tres niveles de restricciones y las dos alineaciones utilizadas con la plantilla de $Poplar$ el número de violaciones

con -ln(pdf) superior a 3 es siempre inferior al 5 % y en el caso de restricciones de distancia es inferior al 1 % lo cual indica un excelente grado de convergencia en los cálculos realizados. En los cálculos realizados con el programa CONGEN, los valores de energía residual de restricciones y de van der Waals se encuentra dentro de los valores aceptables propuestos por los autores [22].

2.3.2 Análisis de la metodología

Influencia de la plantilla y la alineación

La plastocianina $Poplar$ ha sido elegida como primera plantilla no solo debido a que fué la primera plastocianina cuya estructura fué determinada, sino también porque ha sido estructura de referencia en abundantes estudios sobre otras plastocianinas [30]. En los cálculos realizados con el programa MODELLER, se han utilizado dos posibles alineaciones, I y II en la tabla 2.2 con identidades secuenciales muy similares (42 % para la alineación I y 44 % para la alineación II). De este modo se puede evaluar los resultados obtenidos para una misma plantilla en función de la alineación y la localización de los huecos. En la alineación I, hay un único hueco que abarca tres residuos (SER45, ILE46 y PRO47 en la plastocianina $Poplar$) localizados en una región sin estructura secundaria definida y que parece ser un giro en la secuencia de $Poplar$. Sin embargo, en la alineación II, los huecos son dos. Existe un hueco de un residuo (ASP51) y uno de dos residuos (ILE55 y SER56 en $Poplar$) respectivamente, localizados en una zona de tendencia α helicoidal. El efecto de esta situación de los huecos se puede observar en las energías obtenidas para dichas alineaciones en la tabla 2.5. Se puede comprobar que la alineación con un solo hueco y situado en zonas sin estructura secundaria produce resultados de menor energía en general que aquella en que los huecos se encuentran en zonas de estructura secundaria definida. Esto es fácilmente explicable en función de la reorganización estructural que supone la presencia de un hueco en una alineación de secuencias. La ausencia de un residuo en una posición en la secuencia fuerza a los residuos vecinos a adaptar su conformación a las necesidades del hueco. Por ello, si éste se sitúa en una zona de mayor flexibilidad conformacional como pueda ser un giro, el coste energético conformacional de acoplamiento es inferior. Sin embargo, la diferencia en las alineaciones I y II no es suficientemente grande como para producir resultados energéticos muy diferentes y tan solo se puede apreciar una tendencia a menores energías

en la alineación I [126].

	$D \leq 6.0$ Å	$D \leq 8.0$ Å	$D \leq 10.0$ Å
Poplar Alin I	-6236	-5628	-6404
	-6055	-6101	-6250
	-5928	-5890	-5810
	-6167	-5906	-6165
	-5887	-5798	-6167
Poplar Alin II	-5411	-5927	-5912
	-5464	-5888	-5912
	-5504	-5814	-6002
	-5768	-5897	-5958
	-5596	-5917	-6087
French	-5445		-5887
	5397		-5282
	-4508		-5173
	865		-4874
	331		-4925
Parsley	-5466		-5411
	2385		-5075
	-2650		-4508
			-2806
			-4466
Anabaena	-5794		
	-5787		
	-5653		
	-5217		
	-4876		

Tabla 2.5: Tabla de Energías (en KJ mol^{-1}) de las estructuras de Plastocianina *Synechocistys* obtenidas por homología con diferentes plantillas y a diferente nivel de restricciones.

Por otra parte, se aprecia una ligera tendencia a disminuir la energía a medida que el nivel de restricciones aumenta. Dicha tendencia parece confirmar que un número más elevado de restricciones no solo ayuda a obtener el plegamiento sino que colabora en la determinación de la orientación de cadenas laterales en sus conformaciones de menor energía. Sin embargo, el número de restricciones no ha de ser tan elevado que obligue al sistema a adquirir conformaciones que no son propiamente suyas. Un número excesivo de restricciones podría forzar a la proteína problema a adquirir conformaciones propias de la plantilla en lugares en que la conformación de mínima energía no coincida. Por ello, el nivel adecuado de restricciones parece ser aquel que permite una cierta libertad conformacional al sistema

para alcanzar su propio mínimo al tiempo que se asegura el cumplimiento del objetivo del cálculo, es decir, la obtención de la estructura de una proteína que de por sí mantiene una homología estructural. El nivel de restricciones parece tener pues, un límite inferior a partir del cual las restricciones no son suficientes para garantizar una convergencia en los cálculos. Se han realizado cálculos adicionales tomando como distancia máxima 4 Å,y se ha utilizado la alineación I de la plantilla de *Poplar* y se han obtenido valores de energía promedio sobre -3000 KJ mol^{-1}, muy superiores a los del nivel de restricciones inmediatamente superior. Así pues, se ha considerado el intervalo de restricciones utilizado como el idóneo ya que para una plantilla de calidad estructural adecuada se han obtenido resultados energéticamente favorables en todo el intervalo.

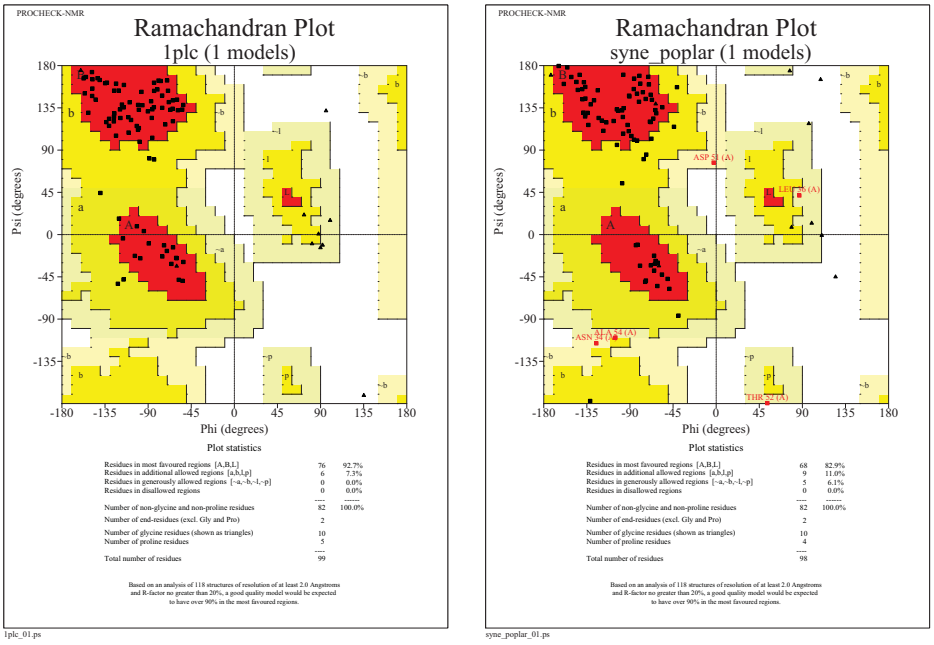


Figura 2.6: Diagramas de *Ramachandran* elaborados con el programa Procheck con las zonas coloreadas por preferencia energética para la plastocianina *Poplar* y *Synechocistys* por homología. Los residuos representados como Δ son glicinas. Los residuos en rojo están en zonas desfavorables energéticamente.

Observando los valores de RMSD (desviación cuadrática media de las coordenadas, ver tabla 2.7) respecto a la estructura promedio de cada conjunto de estructuras, se puede apreciar la dispersión en todos los resultados obtenidos. Dentro de una misma plantilla no se puede apreciar una tendencia clara, aunque se aprecia una dispersión similar para el mismo nivel de

restricciones y que esta disminuye al aumentar el número de restricciones, como cabía esperar. Cuando la calidad de la estructura plantilla disminuye la dispersión, en general aumenta, debido a que las restricciones derivadas pueden no conducir a las situaciones energéticamente más favorables y por lo tanto, cuando el propio método intenta corregir estos *puntos calientes* parece dispersar la solución en su exploración del espacio conformacional.

en %	**Proteína Plantilla**	**Modelo de Homología**	**Identidad Secuencial**
Poplar Alin I	48.7	46.3	41.8
Poplar Alin II	48.8	46.5	43.9
Parsley	35.3	37.3	52.0
French	45.0	50.0	40.8
Anabaena	–	–	53.1

Tabla 2.6: Porcentajes de identidad secuencial en zonas de estructura secundaria definida según esta se encuentre en la plastocianina plantilla o en el resultado de modelización.

Una vez analizado el efecto de la alineación sobre una misma plantilla con la plastocianina $Poplar$, vamos a ver el efecto de que tiene la calidad y alineación de la plantilla sobre el cálculo. En primer lugar, vamos a analizar la influencia de la calidad estructural de la plantilla en los cálculos realizados. Se puede utilizar un criterio estereoquímico para establecer la calidad estructural de un determinado modelo. El programa PROCHECK es uno de los más utilizados con este fin y es de los más reconocidos para comprobar la calidad estereoquímica de modelos de estructura de proteínas. Se ha utilizado este programa sobre todas las estructuras utilizadas como plantilla para la homología con $Synechocistys$ asi como con la estructura resultado de homología de mínima energía. Con la excepción de la plastocianina $Anabaena$, las estructuras de plantilla han mantenido el 90 % de los residuos en las zonas más favorables energéticamente en el diagrama de Ramachandran lo que implica un nivel de resolución de al menos 2.0 Å y un factor R no mayor de 20 %. En el caso de la $Anabaena$, el porcentaje de residuos en las zonas más favorables ha resultado alrededor del 60 %. Para la plastocianina $Poplar$ por el contrario, dicho porcentaje se ha situado por encima del 92 % indicando una estructura de una calidad notable. Para la estructura de menor energía resultado de los cálculos de homología, el procentaje de residuos en las zonas más favorables ha sido de un 83 %, porcentaje que se puede asociar a un nivel de resolución de 2.5 a 3.0 Å. Los diagramas de Ramachandran correspondientes a la plastocianina $Poplar$ y

al modelo de menor energía de homología pueden verse en la figura 2.6.

	$D \leq 6.0$ Å	$D \leq 8.0$ Å	$D \leq 10.0$ Å
Poplar Alin I	0.874	0.345	0.384
	0.498	0.268	0.130
	0.854	0.484	0.148
	0.747	0.435	0.132
	0.645	0.373	0.175
Poplar Alin II	0.614	0.800	0.094
	0.833	0.507	0.094
	0.680	0.511	0.196
	0.697	0.593	0.185
	0.522	0.506	0.134
French	0.968		0.231
	0.662		0.109
	0.868		0.161
	1.014		0.157
	0.791		0.181
Parsley	0.810		0.273
	0.614		0.211
	0.653		0.243
			0.590
			0.282
Anabaena			0.715
			0.484
			0.638
			0.693
			0.741

Tabla 2.7: Valores de RMSD entre cada estructura y el promedio para cada nivel de restricciones y alineación para los átomos de la cadena principal de los residuos con parámetro de orden de conservación de ϕ/ψ mayor de 0.9 en el mismo grupo.

Se puede observar que las plantillas con mejor calidad estereoquímica producen modelos bastante más favorables energéticamente que el resto. Por otra parte, cabría esperar que aquellas plantillas que presentarán un nivel de identidad secuencial más elevado con la secuencia problema produjeran resultados mejores que aquellas plantillas con menor identidad secuencial. Sin embargo, se puede observar esta tendencia invertida en los casos de las plastocianinas $French$ y $Parsley$. La plastocianina $Parsley$ tiene un 51 % de identidad secuencial 2.6 respecto a la plastocianina $Synechocistys$ (la más alta despues de la $Anabaena$) y sin embargo sus resultados son más desfavorables energéticamente que los de la plastocianina $French$ que tan solo tiene un 40 % o que la $Poplar$ que tiene un

44 %. Incluso los valores de RMSD son superiores en el caso de la plastocianina $Parsley$ indicando una mayor dispersión en los resultados. En el caso de la plastocianina $Poplar$ la altísima calidad de la estructura plantilla podría justificar esas notables diferencias, pero en el caso de la plastocianina $French$ no, ya que ambas plastocianinas, $French$ y $Parsley$ tienen una resolución similar y han sido determinadas por RMN. Este resultado puede ser explicado por el hecho de que la identidad secuencial entre la plastocianina $Parsley$ y la $Synechocistys$ decrece notablemente cuando se consideran solo las zonas de estructura secundaria definida mientras que para las plastocianinas $Poplar$ y $French$ este mismo porcentaje aumenta como puede apreciarse en la tabla 2.6. Esta circunstancia parece derivarse del hecho de que, las restricciones que no hayan sido tomadas de zonas de estructura secundaria definida, provienen de regiones de la proteína plantilla de cierta flexibilidad y se supone que se aplican a zonas de similares características. La aplicación de estas restricciones puede producir por lo tanto en algunos casos, cierta rigidez estructural en zonas que no deberían tenerla y que quizás tengan una conformación de mínima energía bastante diferente a la de la estructura de plantilla. Por ello, es razonable suponer que la localización de un exceso de átomos homólogos en zonas de mayor flexibilidad puede suponer una penalización energética en el resultado final de la homología [126].

Análisis de la influencia del protocolo

Por último, y para analizar otro factor importante en la predicción de estructuras tridimensionales mediante homología, se han comparado los resultados obtenidos con el programa CONGEN y los obtenidos con el programa MODELLER. Se han realizado cálculos con el programa CONGEN únicamente con la plastocianina $Poplar$ como plantilla y la alineación I ya que se ha comprobado con el MODELLER que es la que mejores resultados proporciona. Se puede comprobar en la tabla 2.8 que las energías obtenidas por ambos protocolos son muy similares, asi como la estructura secundaria (ver tabla 2.4). En dichas tablas se representa un subconjunto de diez estructuras con los valores menores de energía conformacional y energía residual de restricciones. La convergencia de dichas estructuras se puede comprobar en los valores de RMSD presentes en la tabla 2.8. Estos valores son bastante pequeños para las regiones mejor conservadas como puede verse en la tabla. Tanto las energías residuales de restricciones como las energías de van der Waals son bajas y muy similares a los obtenidos en

otros cálculos del mismo estilo con CONGEN y templado simulado por dinámica molecular restringida.

Estructura	Energía	RMSD
1	-6227	0.234
2	-5156	0.335
3	-6288	0.281
4	-6163	0.282
5	-6311	0.224
6	-5006	0.307
7	-6119	0.303
8	-5976	0.250
9	-5510	0.290
10	-6180	0.284

Tabla 2.8: Valores de Energías (kJ/mol) y RMSD (Å) para las 10 estructuras seleccionadas del cálculo de homología con CONGEN tomando como plantilla la plastocianina *Poplar* y la alineación I.

En la figura 2.7 se puede apreciar que la conformación final del esqueleto tanto de las 10 mejores estructuras obtenidas con el protocolo de CONGEN como las obtenidas mediante el MODELLER son muy similares. Esto no es sorprendente teniendo en cuenta el alto grado de homología entre la plantilla y la proteína problema. Sin embargo, se puede reconocer el efecto del disolvente en las estructuras obtenidas mediante el programa CONGEN que está ausente en las de MODELLER. Dicho efecto consiste en un ligero aumento del volumen global de la proteína. Estas últimas han sido calculadas en vacío y por eso no consideran dicho efecto.

2.3.3 Análisis estructural del resultado de homología

Descripción de la estructura

La conformación global de la estructura de plastocianina *Synechocistys* predicha por modelización por homología es muy similar al resto de plastocianinas cuyas estructuras han sido determinadas. El modelo adopta una conformación de barril β construida por hebras β entre los residuos 3 a 7, 16 a 17, 20 a 23, 27 a 33, 42 a 43, 68 a 73, 77 a 82 y 92 a 97, con un pequeño tramo de tendencia helicoidal entre los residuos 53 y 55 más cercano a una hélice 3_{10} que a una α-hélice. Un análisis inicial de los residuos 87 a 89 parece indicar una posible tendencia helicoidal. Sin embargo, los ángulos

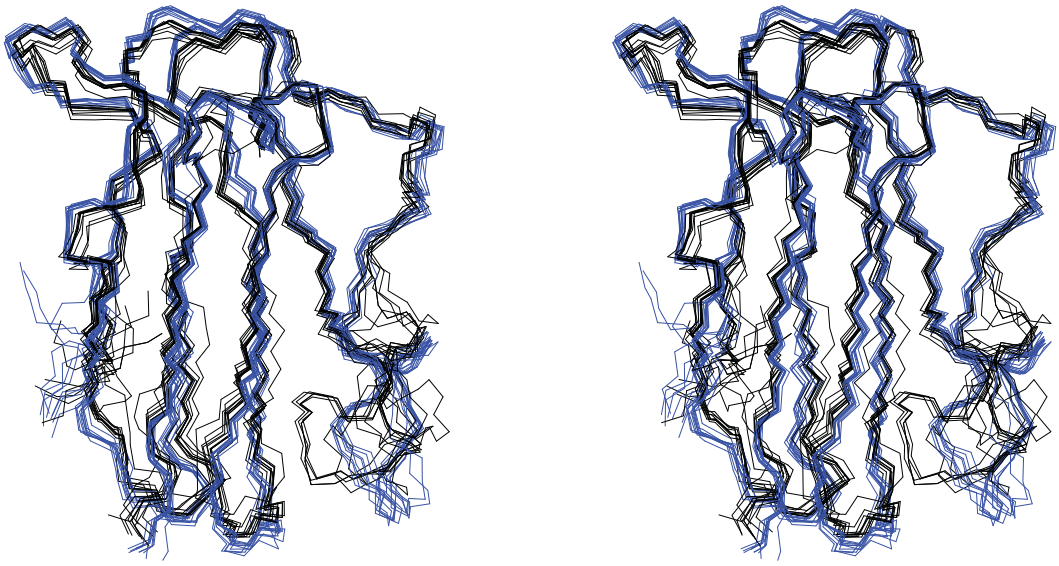


Figura 2.7: Superposición estereográfica del esqueleto carbonado de las 20 estructuras de menor energía para la plastocianina *Synechocistys* mediante MODELLER (negras) y CONGEN (grises).

diedros del esqueleto carbonado y los puentes de hidrógeno indican que la estructura secundaria en esta región está mejor descrita como una serie de giros β de tipo I unidos por puentes de hidrógeno. En particular, existe un alto grado de coincidencia entre la estructura secundaria de la plastocianina *Synechocistys* modelada y las estructuras determinadas por RMN para otra plastocianina de cianobacteria como es la *Anabaena* como puede apreciarse en la tabla 2.4. La diferencia más significativa reside en el pequeño tramo de tendencia helicoidal entre los residuos 50 y 60. Dicho tramo es bastante más largo en las estructuras de RMN que en los modelos de homología de la *Synechocistys* y que en otras plastocianinas. Sin embargo, el esquema general de puentes de hidrógeno de la plastocianina *Anabaena* parece indicar que esta es una zona relativamente flexible de hélice lo cual puede ser bastante difícil de predecir de plantillas como las utilizadas en la homología. El esquema general de puentes de hidrógeno de las zonas de hoja β es el mismo en los modelos de homología que en las estructuras de *Poplar* y *French* reflejando la similaridad existente en la estructura supersecundaria.

Como en el caso de *Scenedemus Obliquus*, el TRP31 y la TYR81 (en la numeración de *Synechocistys*) son de particular interés ya que ambas han sido sustituidas por PHE en las plantas superiores (PHE29 y PHE82 en *Poplar* [10]). Mientras en la estructura de RMN de la *Scenedemus* [73],

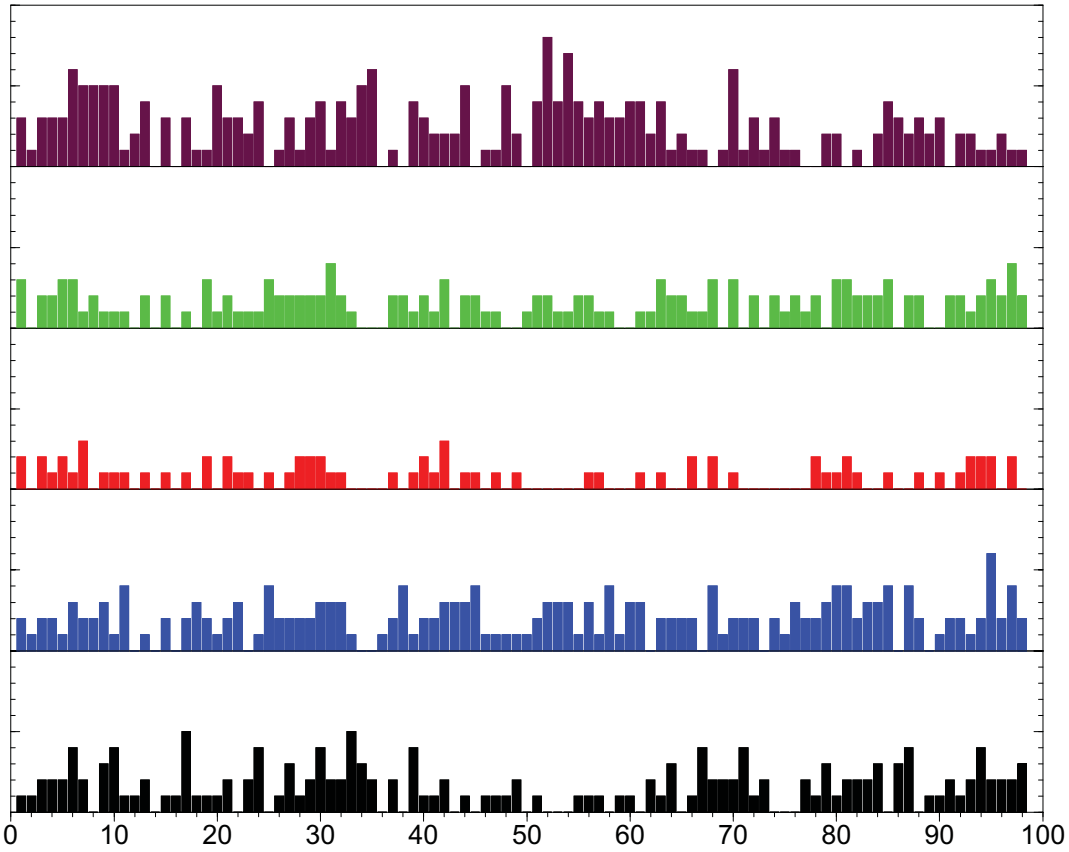


Figura 2.8: Gráfica de la distribución de puentes de hidrógeno en las estructuras de plastocianina utilizadas como plantilla y en la estructura de mínima energía obtenida por homología para la Plastocianina *Synechocistys* (de arriba a abajo, *Anabaena*, *French*, *Parsley*, *Poplar* y *Synechocistys* por homología).

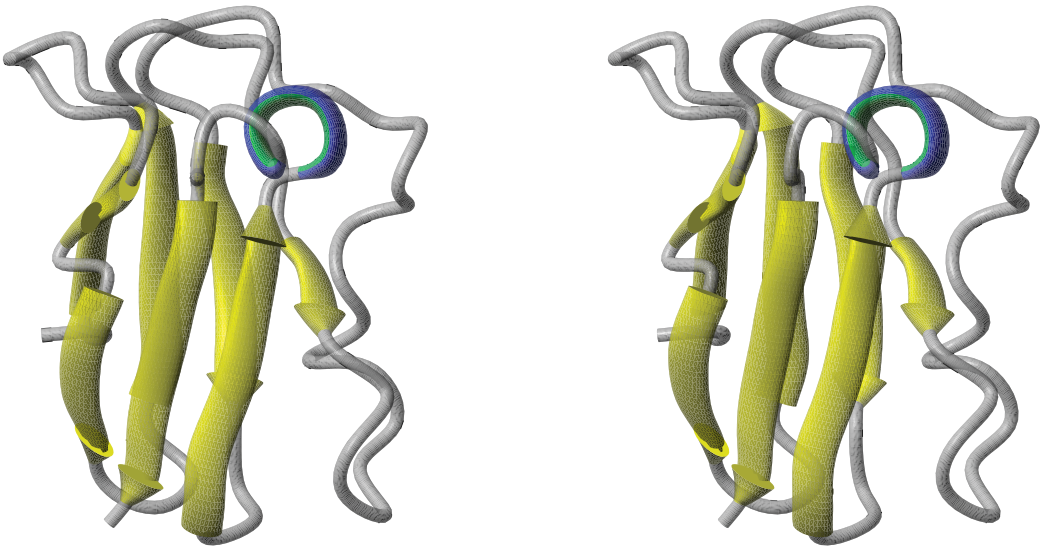


Figura 2.9: Representación estereográfica de la estructura tridimensional de mínima energía entre todos los modelos de homología para la plastocianina *Synechocistys* con indicación de las zonas de estructura secundaria.

el oxígeno fenólico y el NH indólico del TRP29 están dentro de los límites de distancia para formar puente de hidrógeno (lo que puede compensar la penalización energética que supone la introducción de un grupo hidróxilo polar con la TYR en la región hidrofóbica de la proteína), esta misma distancia en los modelos de homología de *Synechocistys* es bastante mayor, en torno a los 6 Å para más del 90 % de las estructuras. La presencia del par TRP31-TYR81 en las plastocianina *Synechocistys* y TRP29-TYR80 en las plastocianinas *Scenecdemus* y *Chlorella fusca* es un ejemplo de mutación complementaria, que por otra parte, no se encuentra en la plastocianina *Anabaena*.

Sitio de enlace al Cu

El lugar donde se encuentra coordinado el átomo de Cu está localizado en la región “norte” de la molécula y consiste en un átomo de cobre coordinado a los residuos CYS83, MET91, HIS39 y HIS86. La coordinación del Cu es dependiente del pH y adopta una coordinación casi tetraédrica o trigonal, esta última con los residuos HIS39, CYS83 y MET91. Además de los aminoácidos coordinados al cobre , hay varios residuos que están conservados en las plastocianinas con secuencia conocida, que parecen desempeñar una

función estructural en el sitio del cobre, como pueden ser la PHE16 en contacto de van der Waals con la cadena lateral de la MET91.

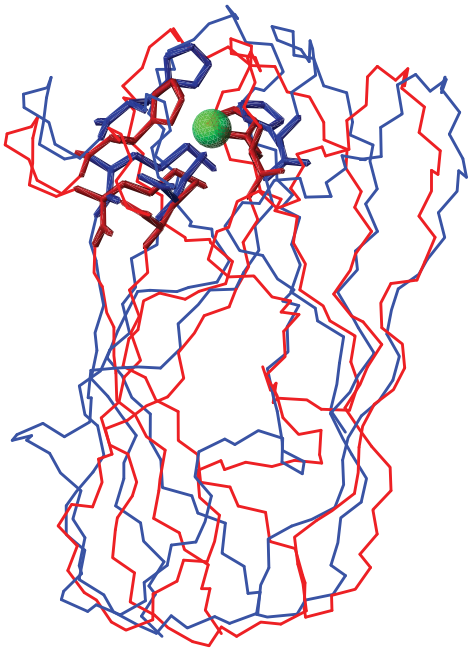


Figura 2.10: Superposición de las estructuras de plastocianina *Poplar* detrminada por Rayos X (Azul) y plastocianina *Synechocistys* por modelización por homología (Rojo) con el atomo de cobre en la posición de Rayos X (verde). Se han representado las cadenas laterales de los residuos CYS83, MET91, HIS39 e HIS86 en trazo más grueso.

En general, no se observan cambios estructurales importantes en el estado cristalino de la apoplastocianina después de la eliminación del átomo de cobre, sugiriendo que la posición de los residuos enlazados al cobre es independiente de la presencia del metal. Sin embargo, se pueden apreciar algunos cambios conformacionales menores: la cadena lateral de la HIS86 ha sido girada 180^o sobre su enlace C_β–$C\gamma$ de modo que el átomo de $N^{\delta 1}$ ligado al cobre se expone directamente al disolvente en ausencia del metal, sugiriendo que dicho residuo actúa como puerta rotacional para la entrada de cobre en su lugar de coordinación como se puede apreciar en la figura 2.11. Por otra parte, la PRO38, que es adyacente a la HIS39 y está muy conservada en las secuencias de plastocianinas conocidas, sufre un pequeño cambio conformacional de C^γ-exo a C^δ-endo.

La conformación global del centro del cobre en la estructura de plastocianina *Synechocistys* de mínima energía modelada por homología es

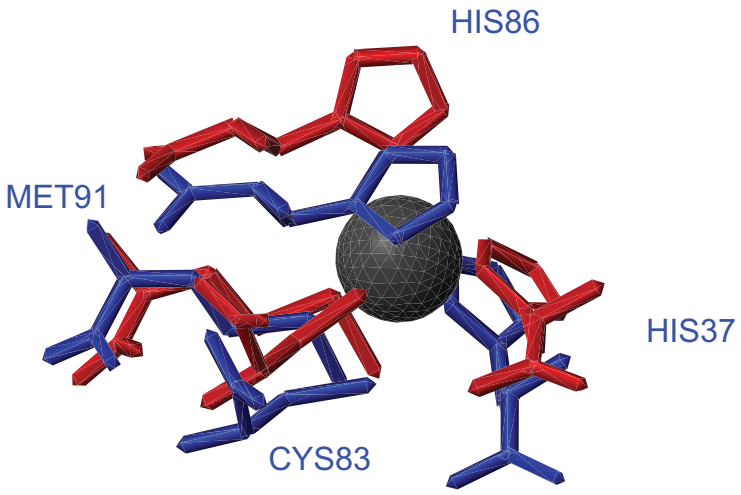


Figura 2.11: Ampliación de las zonas del Cobre en las plastocianinas de *Poplar*(rojo) y *Synechocistys*(negro). Se han representado las cadenas laterales de los residuos CYS83, MET91, HIS39 e HIS86.

bastante similar a la del resto de plastocianinas de estructura conocida. El cambio en la orientación del anillo de la HIS86 de la estructura predicha de mínima energía respecto a la estructura de Rayos X de la plastocinina *Poplar* es de especial interés. Los valores del ángulo diedro C_β–C_γ de la HIS86 son 125^o para la plastocianina *Poplar* y 80.9^o para el modelo predicho por homología para *Synechocistys*, en el cual el anillo de la histidina rota para exponer el átomo de $N^{\delta 1}$ al disolvente. Pequeños cambios conformacionales se pueden observar también en la PRO36 entre la estructura de Rayos X de *Poplar* y la predicha por homología de *Synechocistys* aunque la diferencia en los valores de ángulo diedro difieren en menos de 15^o. Como en *Poplar* y en otras plastocianinas, el PHE16 en la estructura de *Synechocistys* predicha está en contacto de van der Waals con la cadena lateral de la MET91, la cual puede ayudar a la estabilización de la posición del residuo en el centro del cobre. Por otra parte, como ocurre en la estructura de Rayos X de la plastocianina *Poplar*, las cadenas laterales de los residuos bien conservados en la familia de plastocianinas LEU14 y ASN33 en la estructura de mínima energía de la homología de *Synechocistys* están en contacto de van der Waals con las cadenas laterales de la HIS86 y la HIS39 respectivamente, lo que puede contribuir a mantener la conformación del centro de cobre en las diferentes plastocianinas. Además, la cadena lateral de la ASN40 en la estructura de mínima energía de *Synechocistys* da lugar a un puente de hidrógeno (a una distancia aproximada de 3.0 Å) entre el

nitrógeno peptídico del residuo GLU84 y el oxígeno carbonílico del residuo HIS58, quizás estabilizando la posición de la CYS83. Esquemas conformacionales similares pueden ser observados en la estructura de Rayos X de la plastocianina *Poplar* para los residuos LEU12, ASN31, HIS37, ASN38, GLU59, SER85 e HIS87.

Sitios de interacción

La región hidrofóbica en las plastocianinas esta situada en la parte "norte"de la proteína y rodea a la HIS86, que es el único residuo expuesto al disolvente de los ligados al átomo de cobre y uno de los lugares más favorables para la transferencia electrónica (ver figura 2.12). La región hidrofóbica en las plastocianinas de células eucarióticas está constituida por al menos ocho residuos (LEU12, ALA33, GLY34, PHE35, PRO36, PRO86, GLY89 y ALA90) [29] de los cuales tan sólo cinco están conservados en la plastocianina *Synechocistys* (LEU14, PRO38, PRO85, GLY88 y ALA89), con el resto de residuos ALA33, PHE35 y GLY34 sustituidos por LYS35, SER37 y LEU36 en la plastocianina *Poplar*. La naturaleza de la región hidrofóbica puede ser diferente por lo tanto, en las proteínas de cianobacterias respecto a las de células eucarióticas. Se pueden observar similares sustituciones a las apuntadas en la plastocianina de *Synechocistys* en la plastocianina de *Anabaena*, en la cual la GLY34 es reemplazada por VAL36 y la PHE35 por PRO37, mientras que los otros residuos de la región hidrofóbica se conservan respecto a la *Synechocistys*. Sin embargo, como ocurre en el resto de plastocianinas de estructura conocida, los residuos de 35 a 38 en la estructura de mínima energía de los modelos de homología para la plastocianina *Synechocistys*, se encuentran en una zona de giro entre las hebras β 3 y 4 y proporcionan una pequeña región hidrofóbica de forma marcadamente plana. Las cadenas laterales del único aminoácido cargado en este área, la LYS35, se sitúa a lo largo de la superficie de la molécula en dirección opuesta a la HIS86. La conservación de esta pequeña región hidrofóbica sugiere que la planaridad de esta superficie es un factor importante para la interacción de la plastocianina con el sustrato con el que reacciona, que parece utilizar dicha región hidrofóbica.

Se han propuesto dos posibles lugares de interacción con el sustrato de reacción de las plastocianinas y dos posibles vías de transferencia del electrón. En el lugar I (ver figura 2.12), la transferencia electrónica ocurriría vía un mecanismo redox de esfera exterior a través del residuo HIS86. El lugar II consiste en la TYR82 y, en las plastocianinas de plantas superiores,

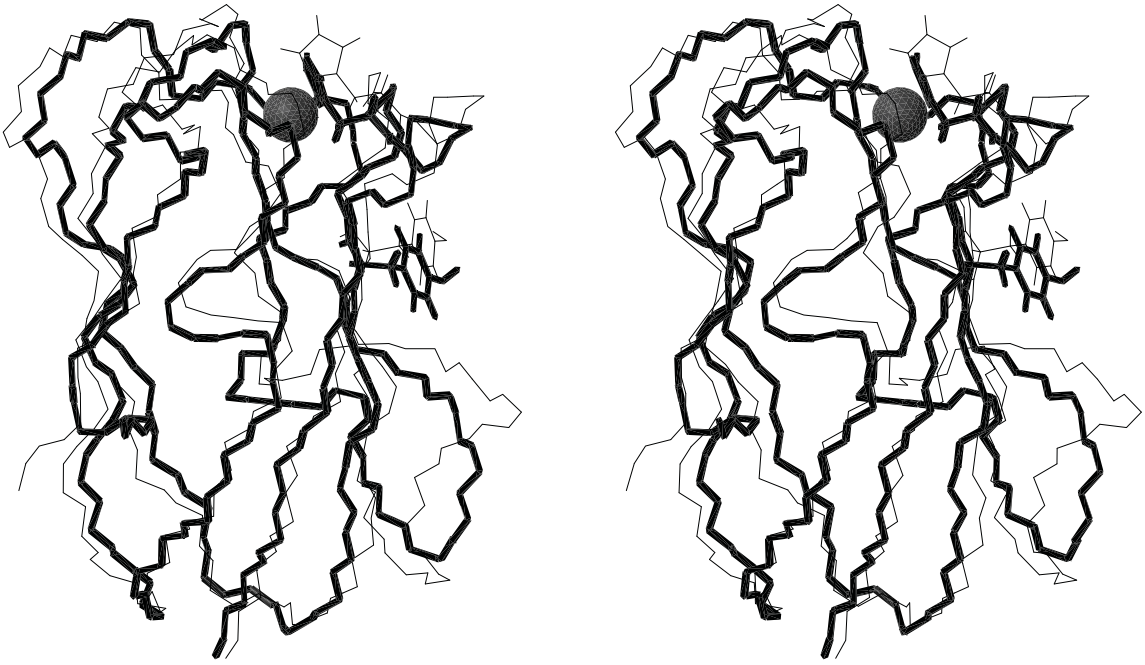


Figura 2.12: Vista estereográfica de una superposición de la estructura de mínima energía de la plastocianina *Synechocistys* (trazo fino) y la estructura de plastocianina *Poplar* de rayos X (trazo grueso) indicando los residuos fundamentales en las dos vías propuestas para la transferencia electrónica.

Residuo	χ_1 French Bean	χ_1 *Poplar*	Homol.
F16(F14)	-78	-74	-70
E17(V15)	-176	-175.5	-65
K24(S22)	30	-42	-177
E28(K26)	51	-167	-65
K30(V28)	179	-172	-175
W31(F29)	-66	-71	-63
H39(H37)	-56	-76	-67
K59(E60)	-169	-55	-67
S68(T69)	45	50	-67
F69(F70)	172	180	166
S71(V72)	64	60	64
E73(L74)	-56	-55	-68
E75(T76)	-42	-33	-68
Y79(Y80)	-68	-65	-64
Y82(Y83)	61	44	62
C83(C84)	164	170	-173
E98(N99)	-69	52	-72

Tabla 2.9: Tabla comparativa de los valores de χ_1 de los residuos conservados en los sitios de interacción en las plastocianinas utilizadas como plantillas y de la estructura de mínima energía por homología de la Plastocianina *Synechocistys*.

las zonas circundantes cargadas negativamente, las cuales están constituidas por las cadenas laterales ácidas de los residuos 42 a 45 y 59 a 61. En las plastocianinas de células procarióticas, los residuos en las posiciones 58 a 60 [11] han sido eliminados respecto a las proteínas de plantas superiores, resultando en un ligero cambio en la forma de la región negativa correspondiente, que puede ser parcialmente compensada por la presencia de cargas negativas en los residuos de las posiciones 53, 58 (ASP) y 85 (GLU). Una situación similar se ha observado en la plastocianina de *Parsley* [30], donde la ausencia de residuos ácidos en las posiciones 59 a 61 aparece compensada por la presencia de los residuos ácidos GLU53, GLU85 y GLU95. Sin embargo, el análisis de la secuencia de las plastocianinas de cianobacterias como *Anabaena* y *Synechocistys* muestra que los residuos en la región negativa de las plastocianinas de células eucarióticas se encuentra sin carga en estos casos, y no hay aparición de posibles complejos de transferencia electrónica cationicos.

Así, parece ser que en la evolución desde las algas azules-verdes a las

algas verdes y plantas superiores, la región ácida de las plastocianinas se ha ido progresivamente convirtiendo en un rasgo estructural distintivo que probablemente juega un papel importante en los procesos de reconocimiento y unión a los sustratos de posible transferencia electrónica en medio fisiológico, el fotosistema I y el citocromo f. La interacción de las plastocianinas de plantas con ambos sustratos ha sido planteada como interacción electrostática, en razón a las posibles fuerzas atractivas entre las regiones negativas en la proteína de cobre y ciertos residuos positivos presentes en los dos sustratos redox mencionados. Sin embargo, este no parece ser el caso de la plastocianina $Synechocistys$. Los estudios cinéticos mencionados en la sección 2.1.6 indican que el mecanismo de reacción no implica la formación de un complejo intermedio relativamente estable para el caso de la plastocianina $Synechocistys$ mientras si lo hace para las plastocianinas de plantas superiores [13].

Estudios recientes de mutagénesis han mostrado que se requiere la presencia de un residuo aromático en la posición 83 (82 en la numeración de $Synechocistys$ para que la transferencia electrónica sea eficiente [29]. En las plantas superiores y algunas algas, así como en la plastocianina de $Synechocistys$, hay en esa posición una anillo aromático de tirosina. En prácticamente todas las estructuras predichas por homología para la plastocianina de $Synechocistys$, independientemente de la plantilla y el número y nivel de restricciones, la TYR82 muestra una orientación definida, a pesar de su localización en la superficie y su exposición al disolvente (ver figura 2.12). Es interesante resaltar que el factor-B cristalográfico de los átomos de la cadena lateral de este residuo en la estructura de rayos X de la plastocianina de $Poplar$ es muy bajo, indicando que la posición de los mismos está muy bien definida, así como que la orientación es la misma que la medida en las estructuras determinadas por RMN de $Parsley$ y $French$ y que el valor de RMSD de los mismos es también bajo. Se puede deducir pues, de esta similitud en todas las plastocianinas que la orientación de esta cadena lateral puede tener un significado funcional importante relacionado con el posible papel a jugar por dicho residuo en el proceso de transferencia electrónica hacia y desde los sustratos redox y / o en la formación del complejo con ellos.

2.3.4 Comparación con un triple mutante

Aunque la estructura de la Plastocianina $Synechocistys$ no haya podido ser determinada por cristalografía de Rayos X debido a la imposibilidad

de obtener monocristales estables de la misma, un triple mutante si que ha proporcionado los monocristales necesarios para obtener su estructura mediante esta técnica. El triple mutante de la plastocianina *Synechocistys* presenta las siguientes mutaciones: A44D, D49P y A62L. Es de suponer, basándonos en el principio básico de la homología que un cambio tan pequeño en la secuencia no ha de implicar cambios estructurales apreciables. Sin embargo, hay que recordar que la estructura de plastocianina *Synechocistys* modelada por homología no ha sido modelada teniendo en cuenta los posibles efectos de empaquetamiento que pueden darse en la estructura cristalina. Además, aunque las secuencias del mutante y la proteína silvestre difieran en sólo tres residuos, la primera es capaz de producir monocristales y la segunda no, por lo que alguna diferencia estructural o de otro tipo ha de haber entre ambas. No obstante, a pesar de que pueden existir algunas diferencias entre ambas estructuras, la del triple mutante puede considerarse un punto de referencia experimental para comprobar los resultados obtenidos mediante modelización por homología. Dicha estructura presenta una calidad estereoquímica notable indicando una estructura de alta resolución como puede apreciarse en la figura 2.13. La energía de esta estructura de rayos X, una vez añadidos los hidrógenos y relajadas sus posiciones mediante el mismo protocolo aplicado a las estructuras modeladas con el programa MODELLER, ha resultado ser de -6548 KJ/mol lo cual indica una estructura de energía ligeramente inferior a las obtenidas por homología.

La conformación global de la plastocianina *Synechocistys* modelada por homología y la del triple mutante son bastante coincidente como puede apreciarse en la tabla 2.4. El tramo de α-hélice de los residuos 87 a 89 en el modelo de mínima energía de la plastocianina *Synechocistys* modelada parece desplazarse en la estructura del triple mutante hacia los residuos 84 a 86 más acorde con el resto de estructuras determinadas por rayos X como pueda ser la de la plastocianina *Poplar*. Sin embargo, algunas estructuras de homología también han presentado el tramo de α-hélice en esa región en los mismos residuos que el triple mutante. También se ha observado que el tramo con tendencia helicoidal en las plastocianinas *Synechocistys* modeladas en los residuos 53 a 55 se presenta como α-hélice en la estructura del triple mutante de los residuos 52 a 56. La distribución de las hojas β no presenta casi variación entre ambas estructuras. Las pequeñas diferenecias en las regiones de α-hélice pueden deberse a pequeñas reordenaciones por efecto del empaquetamiento. El esquema general de puentes de hidrógeno del triple mutante en las zonas de hoja β es el mismo que para la

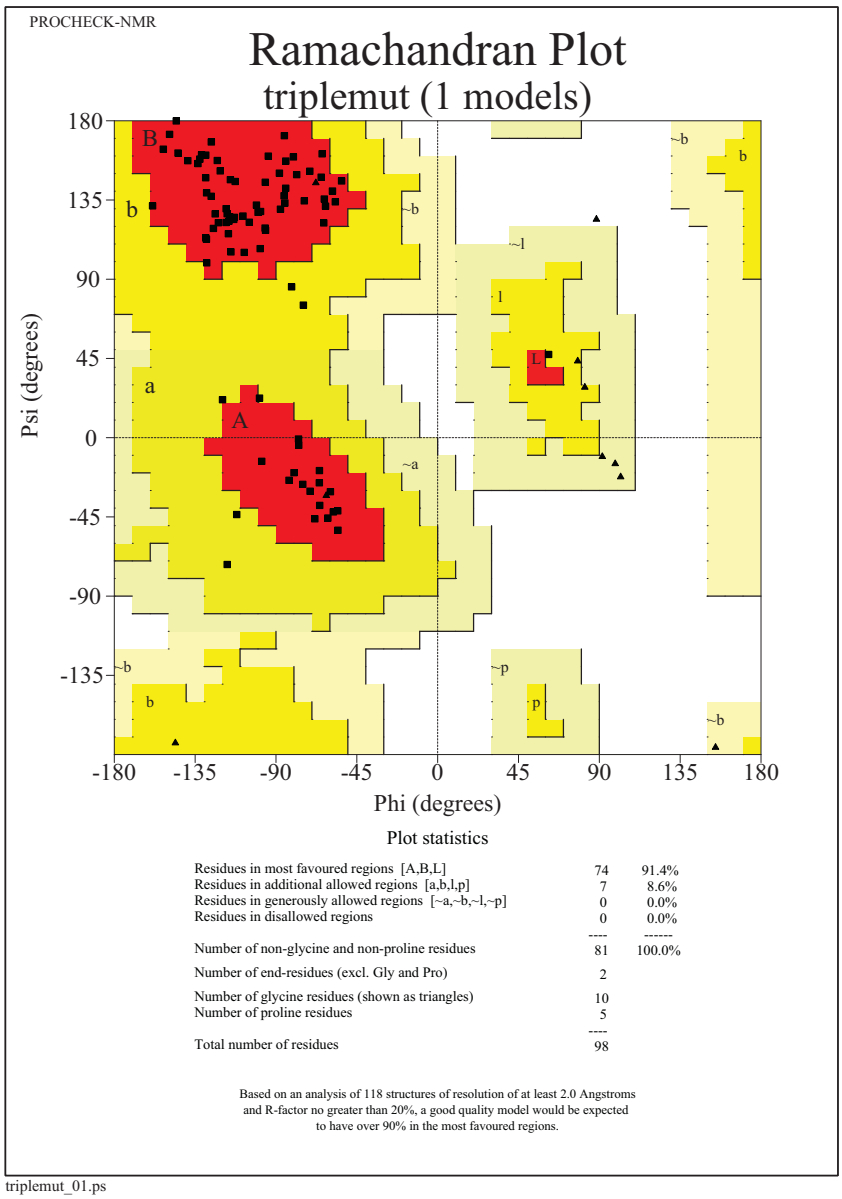


Figura 2.13: Diagrama de Ramachandran de la estructura de Rayos X del triple mutante de plastocianina *Synechocistys* indicando las zonas más favorables energéticamente y los porcentajes de residuos en cada una.

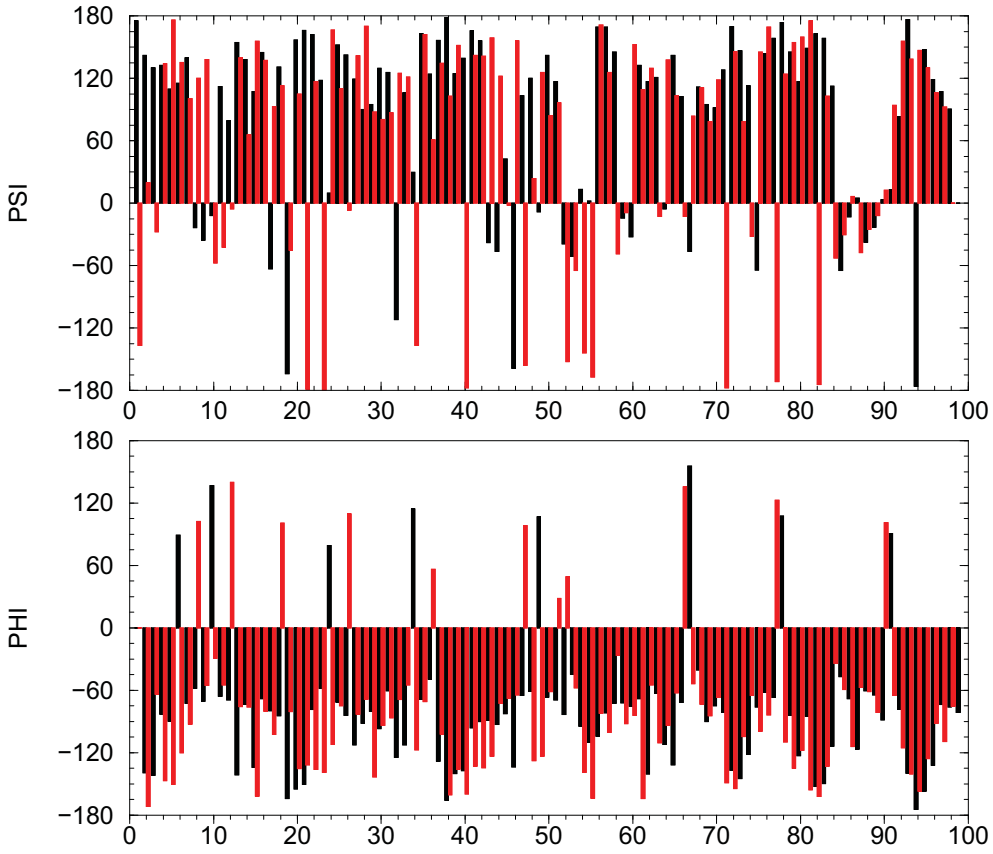


Figura 2.14: Gráfica comparativa de los valores de phi y psi entre la estructura de mínima energía de homología y la estructura de Rayos X del triple mutante. Negro: Triple Mutante, Rojo: Modelo de Homología.

plastocianina *Synechocistys* modelada de mínima energía. Asi pues, la estructura de plastocianina *Synechocistys* modelada posee un plegamiento acorde con la estructura del triple mutante determinada por rayos X.

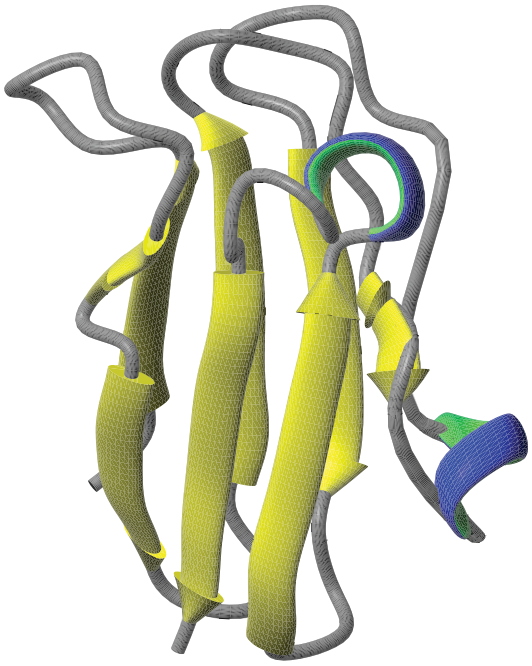


Figura 2.15: Representación de la estructura tridimensional del triple mutante de plastocianina *Synechocistys*.

Haciendo análisis similares al realizado en la sección anterior sobre diferentes peculiaridades estructurales propias de las plastocianinas, se puede comprobar que la estructura del triplemutante no difiere excesivamente del resto de plastocianinas y que es muy similar a la de la plastocianina *Synechocistys* modelada por homología. Por ejemplo, la distancia entre el protón NH indólico del TRP31 y el OH fenólico de la TYR81 es de 4.2 Å y por lo tanto no entra dentro de los límites de puente de hidrógeno como ocurre en el caso de plastocianinas de plantas superiores aunque es bastante más corta que la misma distancia en *Synechocistys* que era de 6 Å.

Otra región de especial interés es el lugar de coordinación del cobre (ver figura 2.16). Esta región está muy bien conservada y la coincidencia entre la estructura del triple mutante y la de la plastocianina *Synechocistys* modelada por homología es muy alta. De hecho, la única diferencia apreciable es la conformación de la cadena lateral de la HIS86. La estructura

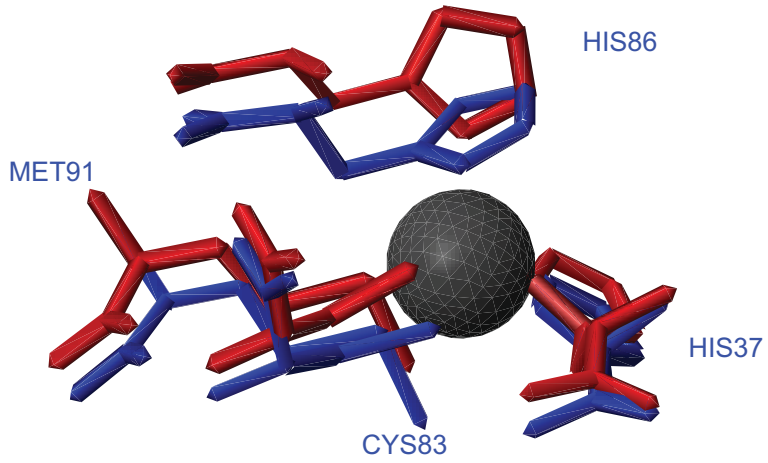


Figura 2.16: Ampliación del lugar de coordinación del cobre en el triple mutante de plastocianina *Synechocistys* (rojo) y en la estructura modelada por homología (negro). Se pueden ver los residuos HIS86, HIS37, CYS83 y MET91.

de rayos X del triple mutante contiene al átomo de cobre y por ello esta histidina se encuentra en la conformación que permite la coordinación de su $N^{\delta 1}$ al mismo, mientras que las estructuras modeladas corresponden a la apoproteína y por lo tanto, como ya se comentó en la sección anterior, la puerta rotacional se encuentra "cerrada". El resto de residuos implicados directa o indirectamente en la región de coordinación al cobre presentan características estructurales altamente coincidentes para ambas estructuras.

En la representación de las estructuras superpuestas (ver figura 2.17) se puede apreciar que las diferencias más significativas se encuentran en las regiones de mayor flexibilidad. Se ha calculado la desviación cuadrática media de las coordenadas para los átomos de las zonas con residuos cuyo parámetro de orden es superior a 0.95. Dicha desviación cuadrática media ha resultado ser de 0.61 para los átomos de la cadena principal y de 0.95 para los átomos pesados. La mayoría de los átomos seleccionados se encontraban en las regiones de hebra β. Esto indica que en aquellas regiones con estructura secundaria definida, la estructura es muy similar y que las diferencias estructurales se sitúan en las zonas de más flexibilidad. De hecho, en el giro que hay a continuación de una de las mutaciones (P49D) hay un cambio conformacional importante que parece favorecer la formación de la α-hélice en los residuos 52 a 56 en el triple mutante. Parece pues, que la presencia de la PRO en la posición 49 en la plastocianina *Synechocistys* obliga a la región adyacente a una conformación que posiblemente, además del efecto ya comentado sobre la α-hélice de 52 a 56, dificulte en alguna ma-

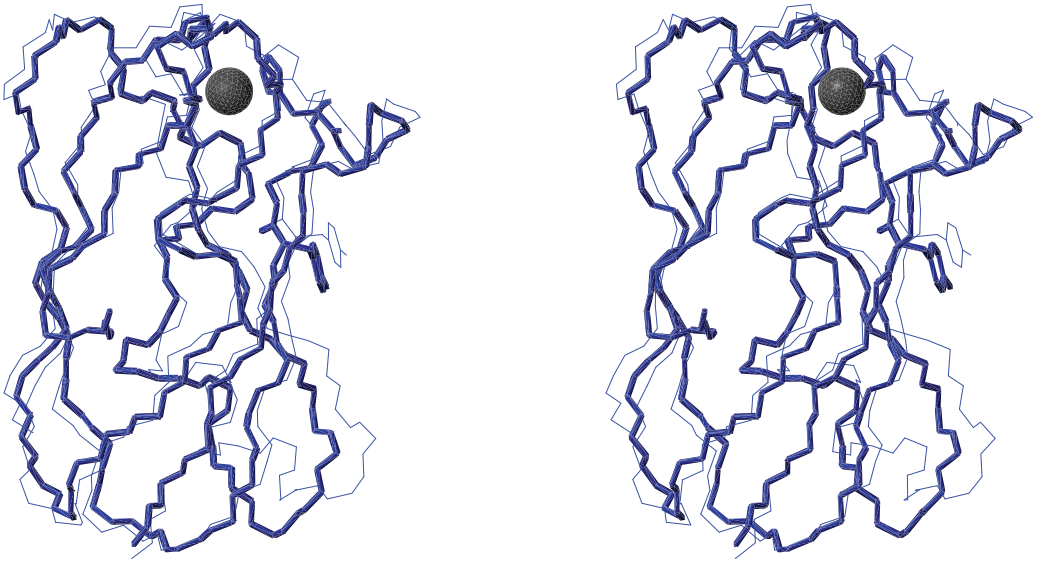


Figura 2.17: Superposición de las estructuras de triple mutante de plastocianina *Synechocistys* (trazo fino) y modelo de mínima energía de homología de plastocianina *Synechocistys* (trazo grueso). Se han representado las cadenas laterales más significativas de los dos sitios de interacción propuestos en la bibliografía.

nera el empaquetamiento requerido para la formación de monocristales.

2.3.5 Análisis de potencial electrostático

La conformación predicha por homología para la plastocianina *Synechocistys* es bastante similar a la de otras plastocianinas, y su calidad, verificada intrínsecamente mediante el análisis energético y estereoquímico de la misma y extrínsecamente mediante un análisis estructural detallado y la comparación con estructuras cristalinas de otras plastocianinas, puede ser considerada como de estructura de media resolución (2.5 - 3.0 Å) [69]. No hay diferencias conformacionales especiales entre la estructura predicha para la plastocianina *Synechocistys* y la estructura de otras plastocianinas que pueda explicar el diferente mecanismo cinético de reacción con los sustratos redox correspondientes (citocromo f y fotosistema I). Sin embargo, hay importantes modificaciones en las características de algunos residuos implicados en la región que rodea a la TYR82. El principal aspecto en el que residen estas diferencias consiste en sus características electrostáticas, ácidas (negativas) para las plastocianinas de plantas superiores y algas, y básicas (positivas) o neutras para las plastocianinas de cianobacterias como

Synechocistys.

Por ello, para entender el diferente mecanismo cinético entre las diferentes plastocianinas, se ha estudiado el potencial electrostático [164]. Se han calculado las superficies de potencial electrostático para las estructuras de las plastocianinas de *Synechocistys*, *Poplar*, *French* y *Parsley* utilizando el módulo incluido en el programa MOLMOL [31]. Este programa calcula la superficie de van der Waals de la molécula en estudio y colorea según el criterio del potencial electrostático en dicha superficie (GRASP [162]), cada punto de la misma. La figura 2.18 muestra la superficie de potencial electrostático de la estructura de rayos X de la plastocianina *Poplar* y la de la estructura de mínima energía predicha por homología para la plastocianina *Synechocistys*. Como cabría esperar, la superficie de potencial electrostático de la estructura predicha para la plastocianina *Synechocistys* presenta importantes diferencias con respecto a la de plastocianinas de plantas superiores. La mayoría de esas diferencias corresponden a la región que rodea a la TYR82. De las dos regiones negativas, de los residuos 42 a 45 y 59 a 61, presentes en las plastocianinas de plantas superiores debajo y encima de la TYR82 respectivamente, la primera es la única parcialmente compensada en la estructura predicha para *Synechocistys*. La ausencia de residuos ácidos en las posiciones 44 y 45 en *Synechocistys* parece ser compensado por la presencia de los residuos ASP49 y ASP51. Sin embargo, la segunda región negativa no está presente en la estructura predicha para la plastocianina *Synechocistys*. De hecho, en su lugar aparece una pequeña región positiva sobre la TYR82 constituida por LYS55, LYS59 y ARG87 (SER56, GLU60 y GLN88 en la plastocianina *Poplar*). Una situación similar se puede apreciar en la región negativa 59 a 61 de la plastocianina *Parsley*. Sin embargo, en dicha estructura, la ausencia de residuos ácidos en las posiciones 57 y 58 se ve compensada por la presencia de los residuos ácidos GLU83 y GLU93 y el potencial electrostático en las regiones rodeando a la TYR80 es bastante similar al de las plastocianinas *French* y *Poplar* (ver figura 2.19).

En la zona "sur" de la estructura predicha para *Synechocistys* y la estructura de Rayos X de *Poplar* parece haber otra importante diferencia en el potencial electrostático. La estructura predicha para la plastocianina *Synechocistys* muestra una región de potencial electrostático negativo en ese área que podría provenir principalmente de los residuos ácidos GLU84 y GLU98 (SER85 y ASN99 en la secuencia de *Poplar*), mientras que en la estructura de la plastocianina *Poplar* dicha región se muestra neutra o cargada ligeramente positiva debido a la cadena lateral de la LYS77 (PRO76

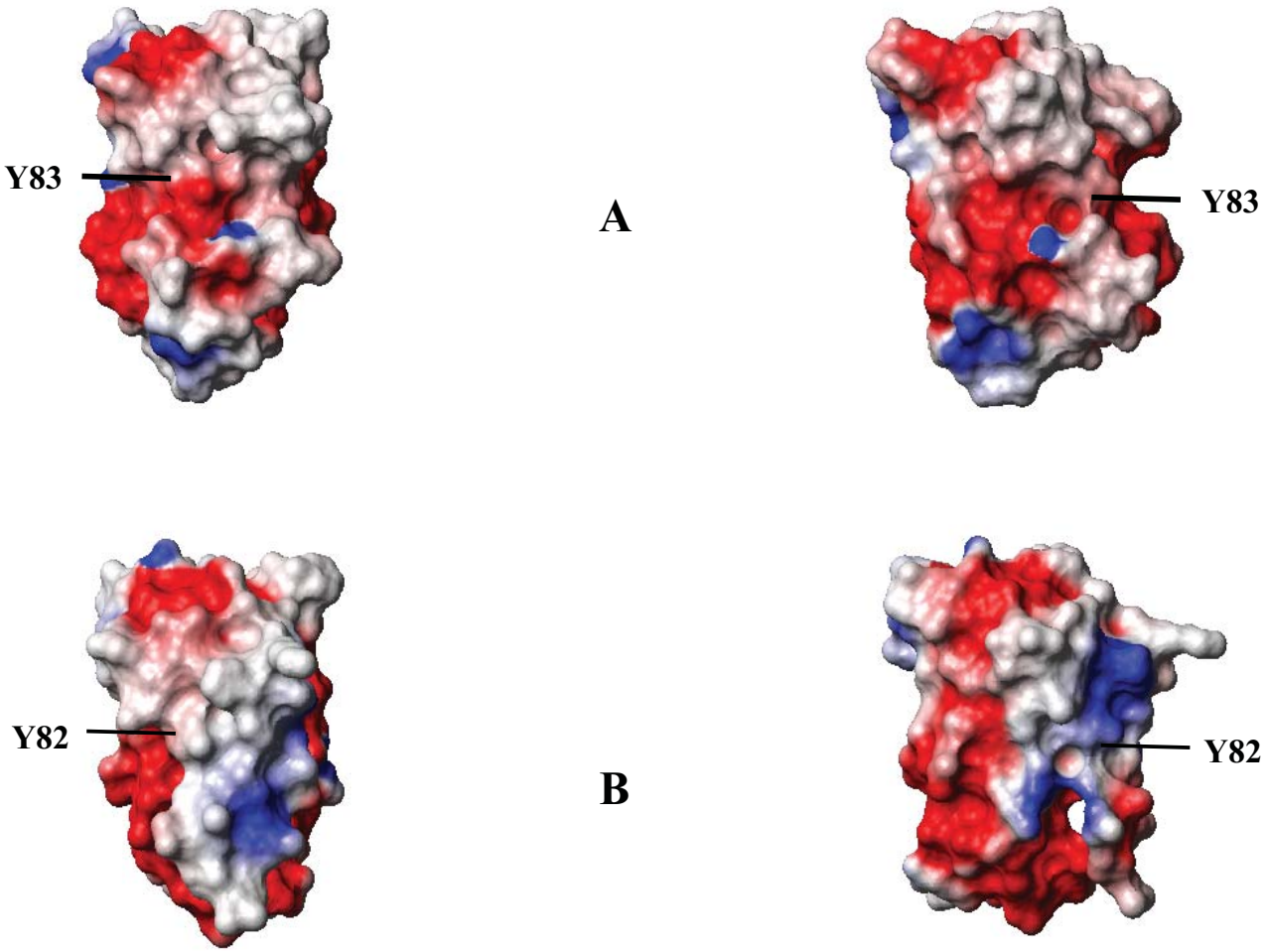


Figura 2.18: Potencial electrostático en la superficie de van der Waals de las estructuras de plastocianina *Poplar* determinada por Rayos X (a) y plastocianina *Synechocistys* predicha por homología (b) calculada por el programa MOLMOL. Las imágenes de la izquierda representan una visión de la zona "este" de la proteína y las de la derecha la orientación típica de las plastocianina.

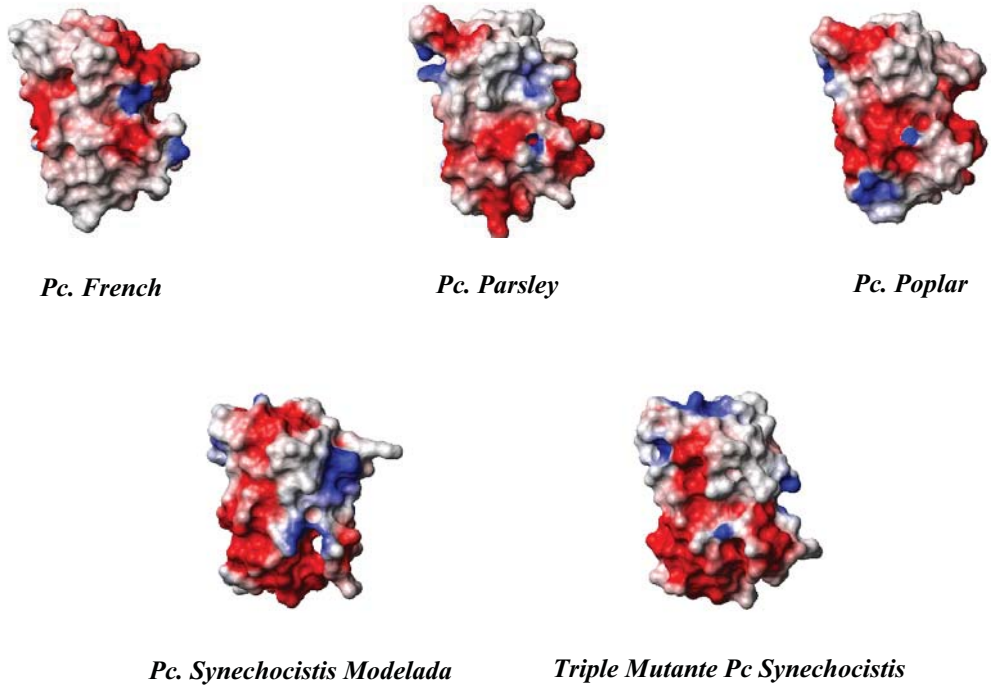


Figura 2.19: Potencial electrostático en la superficie de van der Waals de las estructuras de plastocianina $Poplar$ determinada por Rayos X (a), plastocianina $French$ determinada por RMN (c), plastocianina $Parsley$ determinada por RMN (c) y plastocianina $Synechocistys$ predicha por homología (d) calculada por el programa MOLMOL.

en la secuencia de $Synechocistys$). La diferente orientación de la LYS94 en las estructuras estudiadas de $Poplar$ y $Synechocistys$ produce diferencias importantes en los potenciales electrostáticos. Mientras en la estructura de $Synechocistys$ esta cadena lateral está extendida originando un potencial electrostático positivo mayor, en $Poplar$ está plegada hacia el interior.

El cambio de una alanina por la LYS35 en la zona hidrofóbica no introduce una variación especial en el potencial electrostático circundante. La cadena lateral básica de la LYS35 es opuesta a la de la región hidrofóbica en la que se encuentra y parece una protuberancia positiva en el conjunto de la superficie de potencial electrostático.

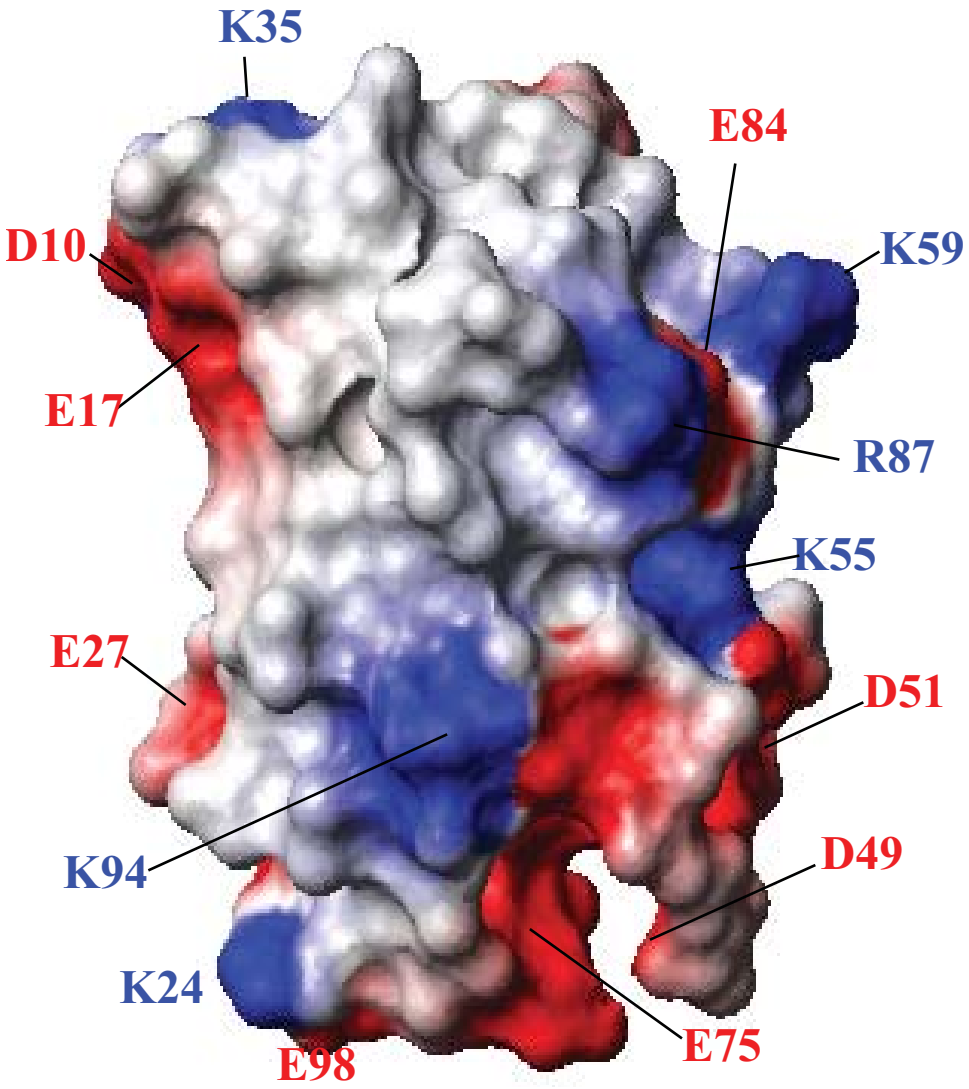


Figura 2.20: Potencial electrostático en la superficie de van der Waals de la estructura de plastocianina *Synechocistys* predicha por homología calculada por el programa MOLMOL. Se han indicado mediante flechas los residuos más relevantes en dicha distribución de potencial.

2.4 Conclusiones

El plegamiento general de la estructura predicha para la plastocianina $Synechocistys$ es similar a la del resto de plastocianinas. Las energías obtenidas para los modelos predichos por homología y los análisis estereoquímicos realizados sobre los mismos nos permiten concluir que la calidad de las estructuras obtenidas por homología son bastante satisfactorias y por lo tanto, dichas estructuras son susceptibles de ser utilizadas como una buena aproximación a la estructura real de la proteína.

Entre los diferentes factores que afectan a la predicción por homología de una estructura de proteína, la identidad secuencial en las zonas de estructura secundaria definida ha resultado ser uno de los más importantes afectando en gran medida tanto a la convergencia y buena marcha de los cálculos como a las energías finales y calidad estereoquímica de las mismas.

El conjunto completo de estructuras predichas mediante cálculos de homología, independientemente del protocolo empleado, el nivel y número de restricciones, la plantilla utilizada y la alineación propuesta, ha conducido a un modelo global único de plegamiento y estructura tridimensional, por lo que dicho modelo se puede considerar consistente con la suposición inicial de homología.

No se han encontrado diferencias sustanciales entre la estructura predicha para la plastocianina $Synechocistys$ mediante homología y las estructuras del resto de plastocianinas de estructura conocida, que permitan explicar satisfactoriamente la diferencia de mecanismos cinéticos observados experimentalmente.

El estudio de potencial electrostático de la estructura predicha por homología y su comparación con los potenciales electrostáticos de las plastocianinas de estructura conocida ha permitido encontrar interesantes diferencias que permiten explicar la formación de un complejo intermedio en la mayoría de reacciones de transferencia en las plastocianina y su no formación en el caso de la plastocianina $Synechocistys$. Estas diferencias pueden justificar el diferente mecanismo cinético de reacción de la misma.

Capítulo 3

Estructura tridimensional de la Plastocianina *Synechocistys* mediante Resonancia Magnética Nuclear

3.1 Introducción

La estructura tridimensional de una proteína puede ser determinada mediante Resonancia Magnética Nuclear de ^{1}H básicamente debido a que los átomos de hidrógeno que se encuentren a una distancia de 5 Å o inferior de cualquier otro darán un pico cruzado NOE (Efecto Nuclear Overhauser) en el espectro NOESY de la proteína, con una intensidad aproximadamente proporcional a la inversa de la sexta potencia de la distancia entre ellos. Hay otros factores que afectan a la magnitud del Efecto Nuclear Overhauser, por lo que dicha proporcionalidad sólo es aproximada y normalmente se trabaja con intervalos de distancias más que con distancias absolutas. Por ello, se suele medir cada pico NOESY y asociarlo a unos determinados límites superior e inferior para las distancias. El método tradicional es clasificar los picos NOESY en fuertes, medios y débiles y asociarlos a los límites superiores 2.5, 3.5 y 5 Å respectivamente. El límite inferior se suele asociar a la suma de los radios de van der Waals. No obstante, la aplicación conjunta de secuencias de pulso mejor implementadas y métodos de matrices de relajación inversa [81] pueden permitir determinar con mayor precisión el intervalo de distancia asociada al volumen correspondiente a

un pico NOESY concreto, así como aumentar el número de intervalos de distancias a partir de un calibrado volumen NOESY $vs.$ distancia.

Por otra parte, de las constantes de acoplamiento ${}^3J$ medidas en los espectros bidimensionales se puede extraer información precisa sobre los valores de los ángulos diedros que implican a los núcleos acoplados [166, 127]. Mediante la relación de Karplus se puede calcular el valor de dicho ángulo diedro en función de la constante de acoplamiento ${}^3J$ y una serie de constantes dependientes de los núcleos implicados.

La presencia e intensidad o ausencia de NOEs entre grupos próximos en la estructura covalente pueden ayudar a discernir la estructura secundaria de la proteína, mientras que el plegamiento y la estructura terciaria se puede elucidar mediante el análisis de los picos NOEs entre átomos lejanos en la estructura covalente y próximos en el espacio. Los picos NOEs son pues, esenciales en la determinación de la estructura de una proteína en disolución mediante Resonancia Magnética Nuclear. La información contenida en los picos NOE relativa a distancias unida a la información sobre ángulos dihedros que se puede extraer de las constantes de acoplamiento hacen a la RMN una técnica fundamental en la determinación de estructuras de biopolímeros en disolución.

La Resonancia Magnética Nuclear aporta información adicional a la estructura como pueda ser la flexibilidad de la proteína en disolución o información sobre la dinámica de la misma proteína (ver capítulo 4). Algunas conformaciones alternativas a la de mínima energía o datos sobre movilidad de grupos pueden ser observados a menudo en los espectros de Resonancia Magnética Nuclear de un modo directo. Incluso la velocidad de rotación de determinados grupos puede ser medida mediante los experimentos adecuados. Sin embargo, aunque la RMN aporte una gran cantidad de información dinámica, la inclusión de la flexibilidad en los cálculos de estruturas de proteínas no es inmediata ni mucho menos.

3.1.1 Introducción a la Resonancia Magnética Nuclear

En los experimentos de RMN de alta resolución, la muestra es situada en el seno de un campo magnético estático B_0 y sometida a la acción de uno o varios pulsos de radiofrecuencia (rf), B_1,B_2... La magnetización macroscópica M de la muestra se sitúa paralela al campo B_0 bajo su influencia. Al aplicar B_1 en una dirección perpendicular a B_0 en el eje x, se ejerce un momento sobre M que tiende a rotarlo en torno a x. Un pulso de radio-

frecuencia de una duración elegida para que la rotación de M en torno a x sea de 90^o situará a m en el plano xy perpendicular a B_0. La magnetización M se encuentra ahora en el plano xy pero en ausencia de B_1 el sistema tiende a a retomar el equilibrio termodinámico con la magnetización M paralela a B_0. En este proceso, M es susceptible de ser descompuesta en dos componentes paralela y perpendicular a B_0 que serán denominadas magnetización longitudinal y transversal respectivamente. La magnetización transversal precesionará en torno a z en el plano xy bajo la influencia de B_0 a la frecuencia de resonancia (o frecuencia de Larmor) ν_0. Esta magnetización transversal, en su recuperación del estado de equilibrio termodinámico, decaerá con el tiempo. En los experimentos de RMN, la magnetización transversal es la que genera las corrientes inducidas susceptibles de ser recogidas por el detector. Esto unido al decaimiento ya expuesto hacen que la medida directa de un experiemnto de RMN, sea un decaimiento libre de la inducción (FID) en función del tiempo que mediante transformada de Fourier es capaz de proporcionarnos una señal en función de frecuencias.

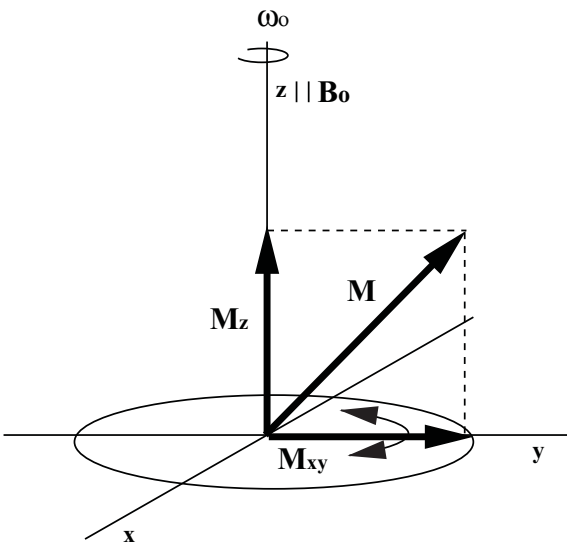


Figura 3.1: Representación vectorial de la magnetización M y sus componentes en el eje z y en el plano xy precesionando a la frecuencia de Larmor.

Lógicamente, el campo aparente que verá cada núcleo particular de la

muestra será el aplicado B_0 más la influencia de su entorno químico (efectos de apantallamientos, efectos de acoplamiento, etc.) que harán que la frecuencia de resonancia de cada núcleo sea diferente y que podamos identificar cada núcleo por su señal correspondiente. Así, cada núcleo posee unas características magnéticas diferenciadas que hacen que los parámetros medidos mediante RMN nos permita diferenciarlos.

El aspecto que más influye en la señal de RMN, en una primera aproximación será el desplazamiento químico (δ). Este define la situación de la línea de resonancia en el eje de radiofrecuencias. Se mide respecto a la señal correspondiente a un compuesto de referencia. Lógicamente, la frecuencia de la señal y por lo tanto su diferencia con una señal de referencia es proporcional al campo aplicado. Por ello, los desplazamientos químicos se suelen dar en unidades relativas independientes del campo. En concreto, se suele usar la unidad partes por millón (ppm). El desplazamiento químico está relacionado básicamente con la estructura química de la molécula en estudio.

La presencia de un núcleo a pocos enlaces del núcleo de interés puede producir acoplamiento con él produciendo un desdoblamiento de los estados de espín y por lo tanto de la señal. Este acoplamiento a través de enlace o escalar depende de una constante de acoplamiento, ${}^3J$, que es independiente del campo aplicado y que se suele medir en hertzios (Hz). Es el responsable de la estructura fina de las líneas individuales de resonancia.

Hay otro acoplamiento que no ocurre a través de los enlaces y que es de crucial importancia en la determinación de estructuras mediante Resonancia Magnética Nuclear. El conocido efecto nuclear Overhauser es la variación en intensidad de un pico de RMN cuando se irradia sobre otra línea de resonancia del espectro en lo que se conoce como experiemnto de doble irradiación. Este efecto de variación de intensidad es debido a interacciones dipolares entre los diferentes núcleos (acoplamiento dipolar o a través del espacio) y está relacionado con la inversa de la sexta potencia de la distancia que separa los dipolos. Se suele expresar este efecto en porcentaje sobre la intensidad sin perturbar de la línea de resonancia en cuestión

Experimentos 2D

Un experimento como el descrito en la sección anterior es llamado un experimento monodimensional de RMN. Este tipo de experimentos proporciona información sobre los desplazamientos químicos y acoplamiento escalar

de las resonancias individuales del espectro. Para obtener información adicional sobre conectividades o acoplamientos vectoriales que proporcionan las bases para estudios conformacionales o asignaciones secuenciales es necesario realizar experimentos de multiple irradiación. Estos requieren una irradiación selectiva de una resonancia particular con un pulso de radiofrecuencia B_2 y la observación del efecto en el resto del espectro. En el caso de los experimentos NOE, la irradiación selectiva es previa al pulso de adquisición pero en el caso de desacoplamiento de espín es simultánea a la adquisición. Estas técnicas tienen una aplicabilidad muy limitada en el caso de moléculas con un elevado número de núcleos como puedan ser las proteínas o los ácidos nucleicos. Aunque la resolución de los espectros pueda ser aumentada utilizando diferentes métodos, la selectividad del pulso en espectros densamente poblados de resonancias es bastante difícil. Además, cada espectro 1D de doble irradiación sólo aporta información sobre las conectividades de un núcleo por lo que el análisis de toda la molécula requeriría un elevadísimo número de espectros.

Las técnicas de RMN en 2D solucionan estas dificultades de un modo natural [130]. La segunda dimensión de tiempo en los experimentos 2D aparece de la siguiente manera. De un modo análogo a la RMN en 1D, el FID es recogido durante el tiempo de detección t_2 después de la aplicación del pulso de detección. Sin embargo, si un pulso no selectivo de radiofrecuencia es aplicado previamente al pulso de observación, la influencia de este pulso en el FID recogido en t_2 depende de la separación entre este pulso y el pulso de observación. Esta separación es el periodo de evolución t_1 y mediante variaciones incrementales del mismo puede generar una segunda dimensión en el tiempo. Para cada valor de t_1 se recoge una FID por lo que obtenemos una matriz de datos $S(t_1, t_2)$ que mediante una transformada de Fourier 2D produce el espectro 2D de frecuencias $S(\omega_1, \omega_2)$.

En general podemos distinguir cuatro intervalos de tiempo en un experimento de RMN-2D [37]. Vamos a analizar estos cuatro intervalos y a evaluar su influencia en el resultado final del experimento.

- *Periodo de preparación*. Es el tiempo durante el cual aplicamos los pulsos que situarán al sistema en el estado previo a la evolución. Este estado será un estado de no-equilibrio coherente. El objetivo de este periodo de tiempo es, pues, crear la coherencia, es decir, que la mayoría de los espínes se encuentren en la misma fase relativa a otros. En el experimento más simple consiste en un único pulso. No obstante, puede consistir en secuencias de pulsos extremadamente complejas

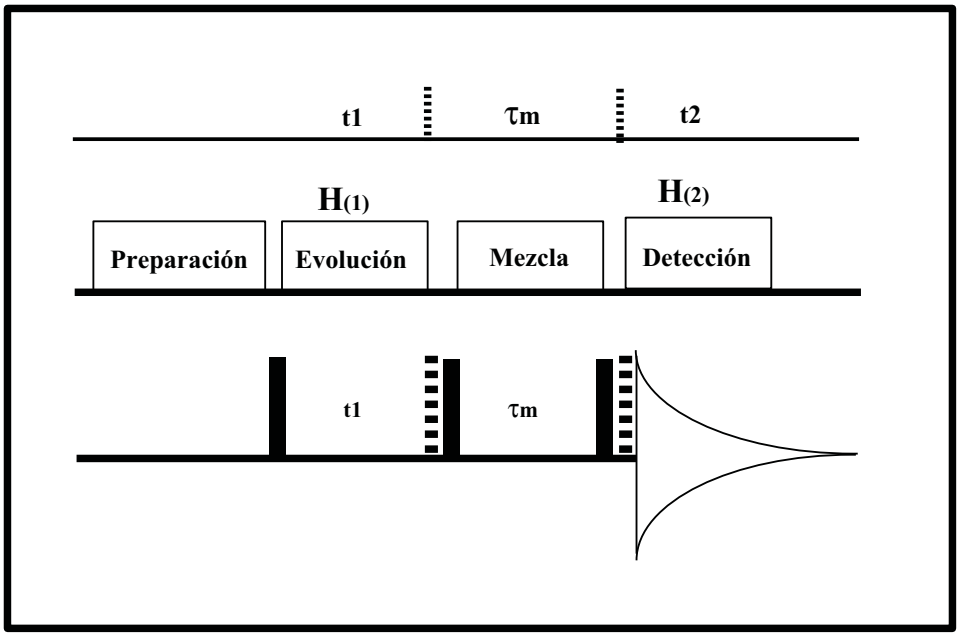


Figura 3.2: Esquema general de una secuencia de pulsos de un experimento RMN-2D. Se pueden apreciar los diferentes intervalos de tiempo.

que impliquen transferencia de polarización, excitaciones de coherencia múltiple cuántica o transferencia de polarización a núcleos de menor sensibilidad. La duración del periodo de preparación suele ser fija para una determinada secuencia de pulsos y se suele llamar τ_p.

- *Periodo de evolución*. El periodo de evolución t_1 es el responsable de la segunda dimensión. Durante este periodo de tiempo, el sistema evoluciona libremente en su recuperación del equilibrio desde el estado en que le situó el periodo de preparación. La coherencia evoluciona y al final de este periodo el sistema se encontrará en un estado determinado por el operador hamiltoniano H^1 efectivo durante t_1.

- *Periodo de mezcla.* Este periodo es el que nos permite extraer la información requerida de muestra. El diseño de una secuencia de pulsos tiene en el periodo de mezcla uno de sus herramientas más poderosas. Este periodo puede incluir uno a varios pulsos de radiofrecuencia y tiempos de espera (delays). Durante este periodo, la coherencia es transferida entre los espines. Por ejemplo, en los experimentos de autocorrelación el proceso de mezcla determina que pares de frecuencia $S(\omega_1, \omega_2)$ tienen intensidad no nula.

- *Periodo de detección*. Es exclusivamente durante este periodo cuando se recoge señal del espectrómetro. Durante el periodo de detección el sistema evoluciona bajo el operador hamiltoniano H^2 efectivo durante t_2 y la FID resultante es recogida y almacenada como una señal de t_2, $s^*(t_2)$. Tendremos así una cantidad de FIDs igual a la cantidad de incrementos de t_1 aplicados al experimento.

Los tiempos de preparación, mezcla y detección son en principio constantes en cada FID y la única diferencia entre ellas es el periodo de evolución t_1. La señal obtenida global será pues $s(t_1, t_2)$ que mediante transformación de Fourier 2D se convertirá en $S(\omega_1, \omega_2)$. Las frecuencias de precesión durante los periodos de evolución y detección, bajo H^1 y H^2 respectivamente determinarán las coordenadas de los picos resultantes.

La información extraída de los experimentos de RMN 2D nos permiten evaluar la magnitud tanto del acoplamiento escalar como del dipolar (acoplamiento J y efecto NOE respectivamente). Esta información nos permite realizar la asignación de resonancias y la extracción de información estructural en forma de restricciones de ángulos diedros y distancias interatómicas, como veremos en la sección 3.2.3. Los experimentos más utilizados para ello son los de correlación COSY, en alguna de sus variantes y el NOESY.

A continuación se describirá brevemente los experimentos bidimensionales homonucleares principales utilizados para la determinación de la estructura de la Plastocianina $Synechocistys$.

DQF-COSY

El COSY ha sido históricamente el primer experimento de RMN 2D [32, 33]. Es también el más simple de todos pues consiste únicamente en dos pulsos de 90^o separados por un periodo de evolución t_1. El periodo de mezcla viene determinado, pues, por un pulso de 90^o separando evolución y detección. El periodo de preparación es simplemente un lapso de tiempo función de T_1 que permita la total relajación del sistema después de la FID anterior. Con él se pueden observar los procesos de transferencia de coherencia debidos a acoplamiento escalar. El COSY produce entonces, mapas de correlación que muestran las conectividades por acoplamiento escalar espín-espín y nos da información sobre los núcleos que se encuentran próximos en la estructura covalente (es decir, a pocos enlaces unos de otros...). A la secuencia de pulsos estandar del COSY se le pueden aplicar

filtros cuánticos que permiten seleccionar el orden de coherencia a detectar en el espectro. Uno de los filtros más utilizados es el doble filtro cuántico que selecciona las coherencias a través de tres enlaces como máximo en el experimento DQF-COSY [34, 35, 36], por lo que se filtran los singletes.

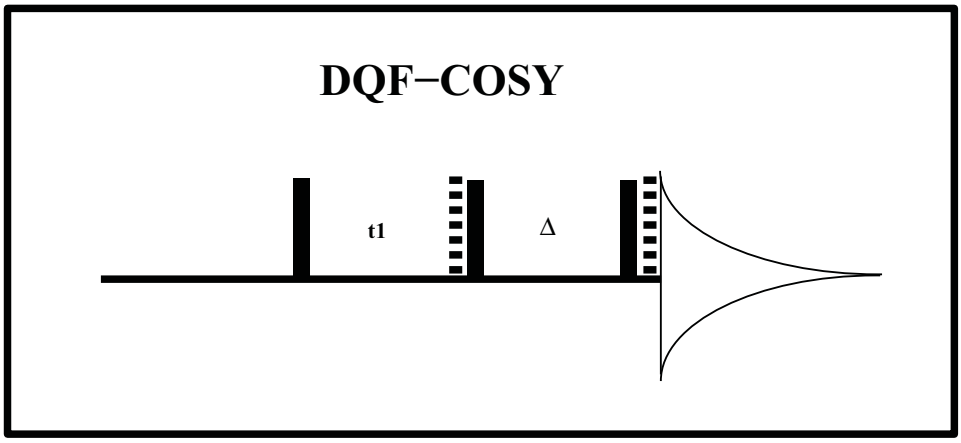


Figura 3.3: Secuencia de pulsos de el experimento DQF-COSY.

La representación vectorial del efecto de la secuencia de un DQFCOSY resulta prácticamente imposible debido precisamente a la generación de la coherencia de doble cuanto que le da nombre. En el experimento DQF-COSY, los dos primeros pulsos tienen el mismo efecto que en un COSY normal, pero el tercero reconvierte la coherencia bicuántica en magnetización transversa observable que aparecerá en antifase. El intervalo de tiempo Δ ha de ser tan corto como sea posible (habitualmente es del orden de los 3 μs).

En un COSY-90 normal los picos diagonales suelen ser de elevada intensidad, debido a la alta eficacia de la correlación de una frecuencia consigo misma. Los picos cruzados próximos a la diagonal entre protones con frecuencias de resonancia próximas entre si aparecerán distorsionados por el efecto de los picos diagonales. Es más, debido a que los picos diagonales aparecen en fase y los picos cruzados aparecen en antifase el efecto puede ser sustractivo y el pico puede desaparecer del espectro.

El DQF-COSY subsana estos inconvenientes mediante el filtro que elimina todas aquellas contribuciones al espectro que no hayan pasado por un estado bicuántico durante el periodo Δ. Los picos de la diagonal no proceden de estados en antifase por lo que su intensidad se ve reducida considerablemente respecto al espectro sin el filtro doble cuántico. Aun-

que los picos de la diagonal no desaparecen completamente del espectro se ven amortiguados en gran medida y los picos cercanos a ella se ven más nítidos.

NOESY

El NOESY [131] es otro experimento de crucial importancia en la aplicación de RMN a la determinación de estructuras y conformaciones. Como su nombre indica, está diseñado para evaluar el efecto NOE entre los diferentes núcleos. Sin embargo, a diferencia de lo que ocurre en los experimentos análogos de 1D, en el NOESY no se utiliza irradiación selectiva. El periodo de preparación consiste al igual que en el COSY en un lapso de tiempo dependiente de T_1. El periodo de mezcla, sin embargo, es ligeramente diferente. Este periodo consiste en dos pulsos de 90^o separados por un tiempo de mezcla fijo τ_m que determinará la intensidad del efecto NOE.

La secuencia de pulsos del experimento NOESY parece idéntica a la del DQF-COSY y, sin embargo, la información obtenida con él es totalmente diferente. La diferencia entre ambos experimentos no está pues en la secuencia de pulsos si no en la detección. Cuando excitamos un sistema de varios espines mediante dos o más pulsos, el número de respuestas que el sistema efectúa es muy numeroso. Cada respuesta corresponde a una coherencia y en la etapa de detección es necesario realizar una selección de la coherencia que deseamos observar. Tradicionalmente, esto se ha realizado mediante el llamado ciclado de fases que permite seleccionar una ruta de transferencia de coherencia sobre el resto. En la siguiente sección explicaremos en más detalle como se aplican estos ciclos de fases en la práctica.

En el experimento NOESY se selecciona pues, la ruta de transferencia de coherencia en que la magnetización regresa al eje Z sobre otras posibles rutas entre las que se encuentra la del experimento DQF-COSY. Además, el intervalo de tiempo Δ es superior ya que es del orden de milisegundos. De hecho, durante este período se produce la relajación dipolar entre los protones y por lo tanto su duración depende en realidad de los valores de T_1 para los protones implicados. La duración del tiempo de mezcla Δ debe ser lo bastante larga como para que pueda tener lugar la relajación dipolar, lo que depende de T_1, pero lo bastante corta como para que la relajación transversal que también tiene lugar simultaneamente y es bastante más rápida que la longitudinal, no haga la señal obtenida nula. Por ello, el experimento NOESY es bastante útil en la aplicación a moléculas de un tamaño relativamente grande, cuyos protones tienen valores de T_1 pequeños,

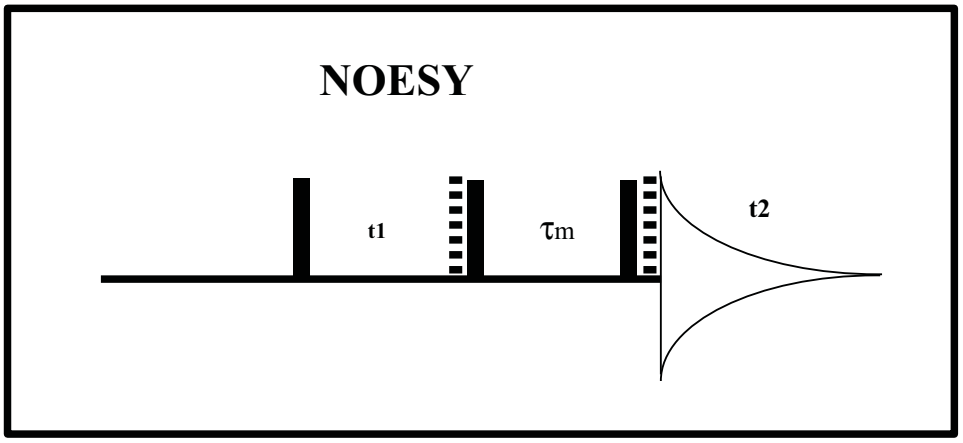


Figura 3.4: Secuencia de pulsos de el experiemnto NOESY.

y no lo es tanto en moléculas orgánicas de pequeño tamaño cuyos protones pueden tener valores de T_1 del orden de los segundos. Esto es debido a que el tiempo de correlación global τ_m de las moléculas pequeñas es bastante pequeño debido a su menor tamaño y su mayor movilidad global. La relación entre el valor de T_1 y τ_m puede observarse en la ecuación 4.2 (capítulo 4). Para este último tipo de moléculas se utiliza a menudo el ROESY, experimento en el cual se bloquean los espines y la relajación dipolar puede ser observada.

Así pues, los picos cruzados en los mapas de correlación NOESY manifiestan los acoplamientos dipolo-dipolo entre núcleos próximos en el espacio. El límite de distancia a la que este efecto NOE es observable en los experimentos de RMN es 5 Å aproximadamente ya que el efecto disminuye con la sexta potencia de la distancia. Con esta información y la obtenida en los experimentos DQF-COSY, se dispone, en principio, de una serie de datos estructurales que permiten, en la mayoría de los casos, evaluar la conformación espacial o conformaciones espaciales de una molécula.

Ciclos de fases

En experimentos de Resonancia Magnética Nuclear es necesario utilizar los llamados ciclos de fases para seleccionar la ruta de transferencia de coherencia elegida sobre el resto de rutas que se producen al aplicar más de un pulso a un sistema multiespín. En un ciclo de fases se modifican las fases de los pulsos y del detector de acuerdo a un esquema regular en sucesivas

repeticiones del experimento en las cuales la única modificación son dichas fases. La adición final de un número determinado de experimentos compensa unas coherencias y anula otras de modo que el resultado final es la coherencia escogida enfatizada.

Cada coherencia ofrece un comportamiento diferente a estos ciclos de fases. De este modo sólo coherencias que han seguido determinadas rutas antes de llegar al detector son sumadas coherentemente y no se anulan en el experimento. Un ejemplo muy simple es el siguiente. Se aplica un pulso con una relación de fase determinada entre el transmisor y el detector, de modo que en el analizador se suman las señales producidas. Si la fase del transmisor se modifica en π radianes y se repite el experimento, tras lo cual se añade la señal producida a la señal anterior, esta señal se ve reforzada como muestra la figura 3.5. Además, de este modo las señales de dispersión debido a la electrónica se cancelan. La secuencia de pulsos (uno sólo en este caso) se repite exactamente igual variando únicamente la fase.

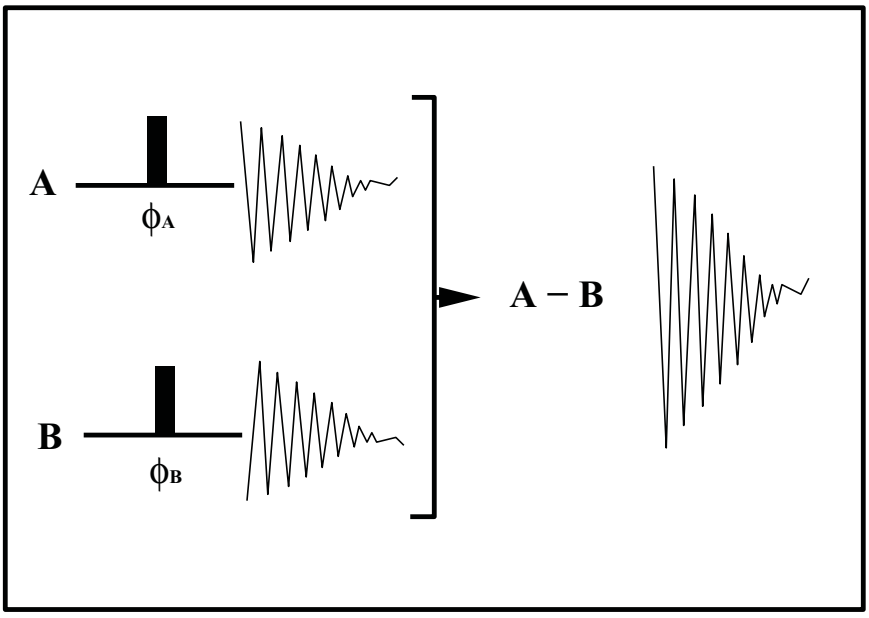


Figura 3.5: Ejemplo de adición de señales en un experimento de un sólo pulso.

En pulsos de mayor complejidad, modificando las fases de los pulsos y del receptor en sucesivos experimentos, se consigue seleccionar una señal correspondiente a una ruta de transferencia de coherencia sobre el resto. Es muy importante que el experimento se repita en las mismas condiciones exactas salvo la fase para que la cancelación que se busca se de en un grado

importante. Si esto no ocurre así se produce ruido en t_1 que dificulta el análisis de espectros.

Un problema que plantea el ciclado de fases es que pueden llegar a ser extremadamente largos para secuencias con muchas rutas de transferencia de coherencia diferentes. Por ejemplo, hay secuencias que requieren la adición de hasta 64 experimentos diferentes cada uno con su combinación de fases en los pulsos. Esta cantidad de experimentos se debería hacer para cada uno de los valores de t_1 del experimento y por lo tanto, para un experimento de 1024x2048 puntos, por ejemplo, se debería repetir 1024 cada ciclo de fases de 64 experimentos.

Una alternativa actual al ciclado de fases para seleccionar la ruta de transferencia de coherencia deseada es la utilización de gradientes de campo. Mediante la utilización de pulsos de gradiente de campo se puede seleccionar una ruta de transferencia de coherencia sobre otras, aplicándolos en los momentos adecuados y con una duración determinada.

Gradientes de campo

Los gradientes de campo magnético consisten en la aplicación de pulsos de campo magnético durante los cuales se hace el campo principal B_0 inhomogéneo variable en la muestra [132]. El método se basa en el efecto de desfase que los gradientes de campo magnético ejercen sobre las diferentes coherencias. De este modo, los desfases que se aplicaban antes a los pulsos para completar un ciclado de fases mediante adición, se pueden aplicar directamente y en un único experimento con los gradientes de campo magnético.

Al aplicar un gradiente de campo magnético de intensidad G_1 y duración g_1 a una muestra se produce un efecto desfasante sobre todo el conjunto de la muestra. Este desfase puede ser corregido para determinadas coherencias, por ejemplo inmediatamente después de un determinado pulso, mediante la aplicación de otro gradiente de campo magnético de intensidad G_2 y duración g_2 si para dichas coherencias se cumple que el desfase global inducido es nulo independientemente de la posición de dichos espínes en el espacio. Para las coherencias que el desfase total no sea nulo, se obtiene al final un conjunto de coherencias desfasadas que se anulan entre ellas y no darán señal observable en el detector.

Se consigue por lo tanto, eliminar el ciclado de fases necesario para seleccionar una determinada ruta de transferencia de coherencia y los experi-

mentos disminuyen su duración, al mismo tiempo que, al poder acumular más cantidad de espectros, aumentan su intensidad y resolución.

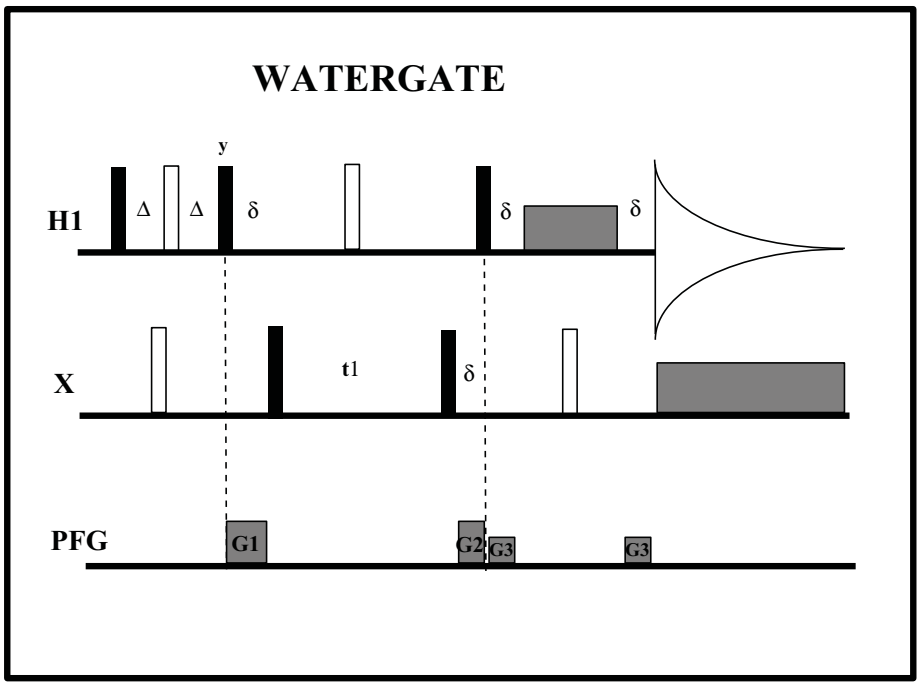


Figura 3.6: Secuencia de pulsos del experimento WATERGATE donde se puede observar la aplicación de gradientes de campo.

Sin embargo, la selección de rutas de transferencia de coherencia no es la única aplicación de los gradientes de campo magnético en la Resonancia Magnética Nuclear. Mediante la correcta aplicación de pulsos de gradientes de campo magnético en determinadas secuencias de pulsos se pueden conseguir efectos adicionales de gran interés como puede ser la disminución de la señal del agua. Este es el caso de la secuencia de pulsos WATERGATE (ver figura 3.6).

3.1.2 Asignación secuencial

La estrategia general utilizada para asignar las resonancias de los protones de una proteína de pequeño tamaño ha sido descrita en detalle en la bibliografía [37]. En la práctica la completa asignación de las resonancias de una proteína requiere un proceso iterativo en el que los pasos desarrollados en esta sección se repiten una y otra vez hasta alcanzar un conjunto de resultados consistente. Los pasos más habituales en la asignación de las

resonancias homonucleares de una proteína son los siguientes:

1. Se realiza la identificación de los sistemas de espín de cada aminoácido para los protones no lábiles de la molécula utilizando para ello espectros realizados en D_2O mediante la interpretación de las conectividades $^1H\ -^1 H$ en la medida de lo posible. En un sistema de espín se puede recorrer toda la secuencia de protones mediante los picos cruzados de acoplamiento escalar del DQF-COSY. Los espectros realizados en D_2O no presentan señales para los protones lábiles expuestos simplificando la interpretación de los mapas de conectividades.

2. Mediante los espectros obtenidos en H_2O de la proteína se completa la identificación de los sistemas de espín realizada en el paso anterior siguiendo la misma estrategia para el espectro completo y utilizando la información obtenida. Una vez terminada la identificación de los sistemas de espín completos para los protones lábiles y no lábiles en la medida de lo posible estamos en condiciones de proceder a la asignación secuencial.

3. Se identifican los sistemas de espín adyacentes mediante la observación de las señales de acoplamiento dipolar en el espectro NOESY. Se buscan e interpretan señales de conectividades NOE secuenciales $H_\alpha - H_N$ o $H_N - H_N$, incluso $H_\beta - H_N$, en el espectro y se realiza una asignación secuencial preliminar.

4. Mediante los pasos 1 a 3 se pretende obtener segmentos de la secuencia de la proteína lo suficientemente largos para que sean unívocos y que nos permitan engarzar los sistemas de espín asignados sobre la secuencia de aminoácidos de la proteína. Las asignaciones específicas dentro de la secuencia de aminoácidos se obtienen entonces, superponiendo los segmentos de secuencia encontrados mediante las conectividades observadas en los espectros de RMN y los correspondientes segmentos de la secuencia de aminoácidos de la proteína.

En la figura 3.7 se puede observar la estructura covalente de un aminoácido, una alanina concretamente, y el sistema de espín identificable mediante acoplamiento escalar y las conectividades NOE que permitirán realizar la asignación secuencial. Aplicando la información obtenida de este modo a todos los protones de una proteína se puede obtener, en principio, la asignación específica secuencial completa de todas las resonancias de un espectro.

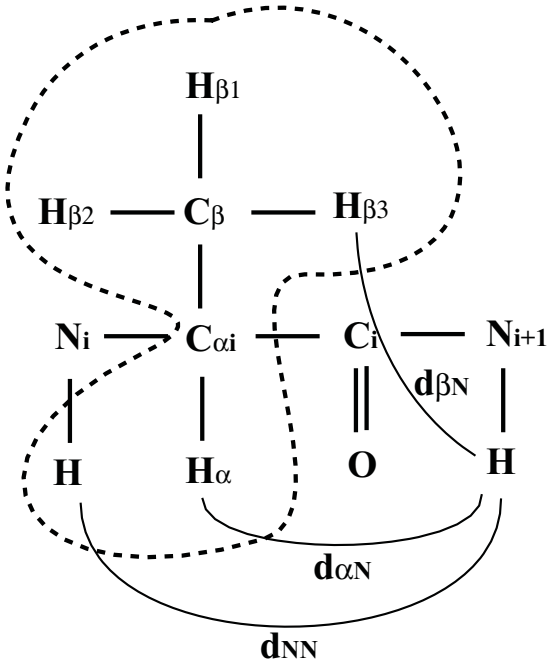


Figura 3.7: Sistema de espín de una alanina y conectividades NOE que permiten asignar secuencialmente el residuo.

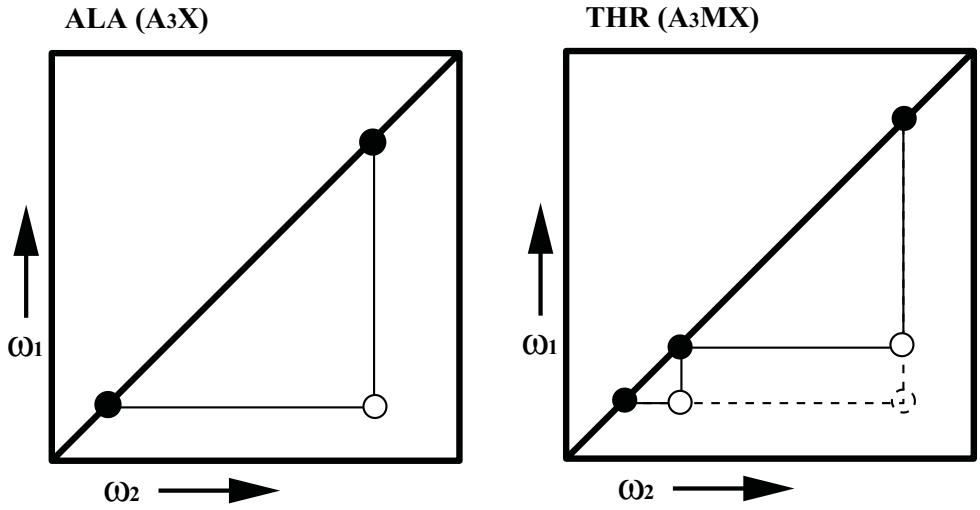


Figura 3.8: Patrones geométricos de la alanina y la treonina. La comparación de estos patrones genéricos con los obtenidos en nuestro espectro permitirán la adecuada identificación de los sistemas de espín.

El primer paso en la identificación de los sistemas de espín presentes en un espectro es la distinción entre los diferentes tipos de sistemas presentes. Los 20 aminoácidos que constituyen una proteína natural pueden identificarse según el patrón geométrico típico de sus conectividades COSY. Podemos distinguir fácilmente, por ejemplo, una glicina de una alanina por la presencia en el segundo caso de una resonancia en la zona de los protones β que no aparece en la glicina (ver figura 3.8). También podemos distinguir una histidina de una lisina por la aparición de resonancias en la zona aromática para la primera. Pautas de asignación similares unidas a la experiencia recogida en la bibliografía sobre desplazamientos químicos y patrones geométricos [37] posibilitan la asignación de las resonancias de un sistema de espín y la identificación del mismo. La organización geométrica de los picos cruzados en los espectros DQF-COSY aportan una valiosisima información en esta identificación.

Una vez identificados los sistemas de espín, es necesario obtener la información que nos permita realizar la asignación secuencial de dichos sistemas de espín. Esta información se encuentra en las regiones del espectro NOESY recogido en H_2O que contienen los picos cruzados $H_N - H_\alpha$, $H_N - H_N$ y $H_N - H_\beta$ secuenciales. Sólo es posible obtener la información necesaria para realizar la asignación secuencial del espectro recogido en H_2O ya que las tres conectividades secuenciales esenciales para la asignación contienen al menos un protón potencialmente intercambiable con el disolvente y por lo tanto en D_2O no se observarían sus señales. Además, las condiciones de pH y temperatura del medio han de ser las adecuadas para que el intercambio con el disolvente sea lo suficientemente lento como para que se puedan observar las resonancias de los protones intercambiables. Sin embargo, muchas veces es conveniente obtener espectros en D_2O, mucho menos poblados de picos, que en caso de posibles solapamientos en H_2O o para resolver ciertos conflictos, pueden arrojar nuevas luces al problema.

En un sistema con la asignación de resonancias completada, se puede seguir el esqueleto carbonado completo mediante los picos cruzados en el COSY y en el NOESY utilizando alternativamente los picos del COSY $H_{Ni} - H_{\alpha i}$ y los picos del NOESY $H_{\alpha i} - H_{Ni+1}$. De este modo, todas las resonancias del sistema estarían asignadas y se podría saber a que interacción corresponde cada pico cruzado de cada espectro.

Lógicamente, la naturaleza del método de asignación empleado con espectros homonucleares de protón limita su aplicación a sistemas de tamaño

moderado ya que la presencia de una cantidad excesiva de señales hace irresoluble la asignación por la gran cantidad de solapamientos presentes. Dichos solapamientos, sin embargo, se pueden resolver de modo relativamente sencillo mediante la utilización de espectroscopía heteronuclear y espectroscopía de RMN-3D cuyo principio básico es el mismo que el de la RMN-2D. Para ello, es necesario disponer de muestras marcadas en ^{15}N o ^{13}C o en ambos isótopos ya que la abundancia natural de los mismos dificulta bastante la obtención de señales apreciables cuando la concentración de muestra es pequeña como es el caso de los biopolímeros. Con muestras marcadas, es posible realizar la asignación específica secuencial completa a través de la cadena principal exclusivamente mediante el acoplamiento escalar. Es posible, mediante la adquisición de espectros de correlación heteronuclear, detectar las resonancias correspondientes al acoplamiento entre el ^{15}N y el protón amídicos y el ^{13}C α y entre este y el ^{13}C de carbonilo y de esta manera continuar a través de la cadena principal a lo largo de toda la secuencia [38].

Una vez la asignación secuencial ha sido realizada, el análisis cualitativo de los espectros COSY y NOESY nos pueden proporcionar información estructural de interés. La aparición de ciertos patrones de parámetros de RMN en la cadena polipeptídica es indicativa de estructuras secundarias particulares. La información así obtenida puede ser un punto de partida para posteriores análisis estructurales de carácter más cuantitativo como pueda ser la calibración de los picos cruzados NOE y su aplicación a geometría de distancias. De la observación de ciertos picos NOESY secuenciales se puede establecer la estructura secundaria de un tramo de la secuencia [37]. Por ejemplo, en las regiones de α-hélice suelen aparecer picos cruzados NOESY correspondiente a una distancia particular como es la distancia entre el protón $H_{\alpha i}$ y el $H_{N\,i+3}$. En las regiones de zona de hoja β los picos cruzados entre $H_{\alpha i}$ y H_{Ni} son especialmente intensos, correspondientes a una distancia interprotónica inferior a 3 Å. Otro ejemplo puede ser la comparación de los desplazamientos químicos de los protones α y los carbonos $^{13}C_{\alpha}$ en cada residuo con el desplazamiento del protón o ^{13}C equivalente en una proteína en ovillo estadístico. La aparición aislada de estos patrones nos está indicando la presencia de un elemento de estructura secundaria en el aminoácido en cuestión. El análisis del esquema de puentes de hidrógeno extraidos de el estudio de los protones H_N no intercambiables también es un elemento de gran interés a la hora de determinar la posible estructura secundaria y el posible esquema de plegamiento de una proteína, así como una fuente de restricciones para el cálculo de estructuras. El análisis con-

junto de dichos patrones y la información de desplazamientos químicos y de la secuencia en estudio nos llevará a conocer mejor las tendencias estructurales de nuestra proteína. Así, este tipo de patrones unido al conocimiento estructural de las conformaciones de estructura secundaria típicas llevan a la identificación de las zonas de estructura secundaria definida en una proteína.

Con toda la información estructural posible extraida de los espectros y el conjunto de distancias interprotónicas evaluadas a partir de los picos cruzados del espectro NOESY, se procede al cálculo a nivel atómico de la estructura tridimensional de la proteína. Existen varios métodos para utilizar dicha información, junto a la información que aporta la estructura covalente de la molécula, y extraer la estructura tridimensional de la proteína. En las siguientes secciones analizaremos las más utilizadas que han sido, por otra parte, las empleadas en el presente estudio.

3.1.3 Geometría de distancias. Matriz métrica y función objetivo variable.

Una vez obtenidas las restricciones espaciales correspondientes a la estructura de nuestra muestra, es necesario aplicarles el correspondiente algoritmo que las transforme en una estructura tridimensional. El primer paso en la generación de estructuras de RMN es el uso de las restricciones de distancias y ángulos diedros en el algoritmo conocido como geometría de distancias [39, 40]. La geometría de distancias implica tomar las restricciones experimentales determinadas por RMN junto a las restricciones derivadas de la estructura covalente, y crear una matriz de distancias interatómicas entre todos los átomos que sea consistente con la información aportada. El conjunto de distancias obtenidas en un espacio de distancias es proyectado posteriormente en el espacio cartesiano. El punto de partida del algoritmo no supone una estructura de partida por lo que en numerosas ocasiones, se utilizan distancias generadas aleatoriamente entre las que se incluyen las restricciones obtenidas. El proceso contiene esencialmente tres etapas:

- Los límites iniciales para las distancias entre los átomos son determinadas mediante triangulación a partir de las restricciones experimentales y aquellas restricciones derivadas de la estructura covalente del sistema. Debido a que, en la mayoría de los casos, las restricciones experimentales son incompletas e imprecisas, se suele recalcular un nuevo conjunto de límites para las restricciones mediante suavizado

de restricciones. Esto implica la selección de los intervalos más pequeños de distancias consistentes con la desigualdades triangulares. La desigualdad triangular es un axioma básico de la geometría que implica todas las distancias posibles entre tres puntos espaciales. En concreto para tres puntos i,j y k cualesquiera:

$$u_{ij} \leq u_{ik} + u_{jk}; l_{ij} \geq l_{ik} - l_{jk} \tag{3.1}$$

- Los valores de las distancias que se encuentran dentro de los límites obtenidos mediante el suavizado de límites se generan aleatoriamente y se calculan las coordenadas atómicas que mejor representan estos límites. Esto se llama "embedding".

- Las desviaciones de las coordenadas de los límites de distancias, así como las asignaciones estereoespecíficas, son minimizadas. Esta etapa se conoce como optimización.

Una dificultad con el método de la matriz métrica de geometría de distancias es que únicamente con la distancia no podemos definir la quiralidad de la estructura y por lo tanto se pueden obtener imágenes especulares (globales o locales) de la estructura real. Para solventar esto en los resultados finales, se suele hacer una comprobación de quiralidad sobre toda la molécula y sólo aquellas soluciones con la quiralidad correcta son utilizadas en los pasos siguientes. Un problema adicional asociado a este método es que no analiza de modo eficaz el espacio conformacional consistente con las restricciones experimentales y la estereoquímica. Como resultado de ello, los valores de la distribución de las desviaciones cuadráticas medias de las coordenadas (RMSD) de una serie de estructuras calculadas mediante este método tiende a ser infravalorada, particularmente en las regiones pobremente definidas por los datos experimentales.

En los métodos que implican minimización en el espacio de ángulos de torsión [42], las distancias de enlace y los ángulos de enlace son fijados durante el proceso y sólo son variados los ángulos de torsión. Para asegurar que el plegamiento es el adecuado, las restricciones se aplican de modo gradual durante el cálculo. Esto se consigue de diferentes modos, de forma que las distancias entre residuos cada vez más lejanos en la secuencia se incorporan al cálculo en sucesivos ciclos de minimización. En general, las estructuras obtenidas por estos métodos tienden a ser de alta energía y deben ser sometidas a métodos de minimización de energía que produ-

cen pequeños cambios en las coordenadas con apreciables mejoras en los valores de la energía.

3.1.4 Minimización de energía

La minimización de energía es habitualmente utilizada para encontrar un mínimo local en una hipersuperficie de energía potencial para una determinada estructura. Entre los algoritmos de minimización comúnmente empleados [43] se encuentran el de "steepest descent", gradientes conjugados y adaptacionales de Newton-Raphson. Todos estos asumen que la hipersuperficie de potencial se puede aproximar a un armónico en la cercanía del mínimo. Aunque la velocidad de descenso de energía por el método de "steepest descent" es inicialmente más alta, este método es no convergente y a menudo produce grandes oscilaciones en torno al mínimo. El método de gradientes conjugados [44] se aproxima al mínimo más rápidamente y resulta en un punto de mínima energía convergente. Sin embargo, si partimos de una estructura muy pobre, es decir, muy lejana del mínimo, el método es inestable numéricamente.

Cada iteración k de "steepest descent" se desarrolla en tres etapas básicas:

- se elige una dirección de descenso representada por un vector unitario, s_k, de $3N$ dimensiones
- se optimiza un tamaño de paso, especificado por el escalar λ_k
- se aplica el paso de descenso de acuerdo a la relación:

$$x_k = x_{k-1} + \lambda_k s_k \tag{3.2}$$

La dirección del desplazamiento s_k en este método es paralela al gradiente negativo del potencial.

El método de gradientes conjugados es un método que conjuga los gradientes de los pasos k y $k-1$. Así, aunque en el primer paso la dirección de búsqueda s_1 sea la misma que la utilizada en el método de "steepest descent", en los siguientes la dirección viene determinada por:

$$s_k = -g_k + b_k S_{k-1} \tag{3.3}$$

donde g_k es el gradiente en el paso k, y b_k es un escalar determinado por $|g_k|^2/|g_{k-1}|^2$, es decir, la relación entre los cuadrados del gradiente actual y el inmediatamente anterior. De este método, la dirección de descenso

no puede cambiar de sentido totalmente como ocurría en el caso de "steepest descent" y es posible la convergencia. El método, por lo tanto, es más eficiente.

El método de Newton-Raphson se basa en asumir que, en la región más próxima al mínimo la energía depende cuadráticamente de las variables independientes. De este modo, se puede aproximar la expresión de la energía a una parábola y se puede calcular el vértice de la parábola en un sólo paso mediante la primera y segunda derivada de la función en cualquier punto. Para superficies de energía cuadráticas, por lo tanto, no hace falta ninguna iteración para calcular el mínimo local. Para superficies no cuadráticas pero monotónicas se suele aplicar una de las múltiples modificaciones de este método. Este método no se suele utilizar en su forma pura por dos razones fundamentales. En primer lugar, el carácter no cuadrático de las funciones de potencial empleadas en mecánica y dinámica molecular hacen muy dificultosa la aplicación del método. En segundo lugar, el elevadísimo número de variables de las funciones de potencial hacen muy costoso el cálculo de las segundas derivadas de la función en un punto. Para solventar este problema se suele utilizar el método de bases adoptadas de Newton-Raphson (ABNR) [45] o métodos de bases truncadas.

En general, se suele aplicar como estrategia general una minimización de "steepest descent" de pocos pasos para relajar el sistema y situarlo en un buen punto de partida para métodos más eficaces como gradientes conjugados o ABNR de los cuales se aplica ya un buen número de iteraciones para asegurar la convergencia. En estas minimizaciones se pueden incluir las restricciones experimentales como parte del potencial mediante funciones armónicas o de otro tipo.

3.1.5 Dinámica molecular restringida y templado simulado

Sin embargo, utilizando los métodos de minimización de energía, es inevitable encontrar mínimos locales en un elevado porcentaje de ocasiones lo que puede conducir a resultados no correctos. Un método de evitar esto es la aplicación de técnicas de Dinámica Molecular Restringida [46]. Esta técnica implica incluir las restricciones espaciales derivadas de los espectros de RMN en la función potencial que simula el comportamiento de la molécula y aplicar dinámica molecular a dicha función. La dinámica molecular se basa en la resolución de las ecuaciones de Newton de movimiento:

$$F_i = m_i a_i \tag{3.4}$$

donde F_i es la fuerza sobre el átomo i, y m_i y a_i su masa y aceleración respectivamente. La fuerza sobre el átomo i puede ser calculada directamente a partir de la derivada del potencial en i respecto a sus coordenadas (r_i). Asi, la expresión 3.4 puede ser expresada explícitamente en forma diferencial:

$$-\frac{dV}{dr_i} = m_i \frac{d^2 r_i}{dt_i^2} \tag{3.5}$$

Así, con la expresión adecuada del potencial y conociendo las masas se puede resolver la ecuación diferencial y obtener las posiciones futuras en un tiempo t_i. Sin embargo, debido a la complejidad de la función potencial V, esta ecuación sólo se puede resolver de modo aproximado. La función V depende de las coordenadas de los átomos a considerar en el potencial $V = V(r_1, r_2, r_3, \cdots, r_N)$. La temperatura del sistema está relacionada con las velocidades atómicas por mecánica estadística:

$$\frac{3N}{2} k_B T = \sum Ni = 1 \frac{1}{2} m_i V_i^2 \tag{3.6}$$

donde k_B es la constante de Boltzmann, m_i y v_i son la masa y velocidad del átomo i y N es el número de átomos a considerar en la función de potencial. En una simulación a energía constante, la temperatura varía debido al intercambio entre energía cinética y potencial de los átomos. Si se mantiene constante la temperatura, las velocidades atómicas han de ser ajustadas de acuerdo a dicho criterio. Si la presión se mantiene constante, se debe permitir al volumen fluctuar mediante el reescalamiento de las distancias interatómicas.

El potencial global se define normalmente como una suma de contribuciones individuales agrupadas según el tipo de interacción que describen. Una forma habitual de escribir los potenciales moleculares es la siguiente:

$$v_{total} = V_{enlaces} + V_{angulos} + V_{diedros} + V_{vdw} + V_{coulomb} + V_{RMN} \tag{3.7}$$

donde $V_{enlaces}$, $V_{angulos}$ y $V_{diedros}$ mantienen las longitudes de enlace, los ángulos de enlace y los ángulos diedros en torno a su valor de equilibrio. Estos subpotenciales surgen de los modos de vibración que puede

presentar una molécula de más de tres átomos (vibración, flexión y torsión) sin contar la contribución de traslación y rotación. Los cinco primeros elementos del potencial tienen un carácter empírico y son los términos que describen las interacciones físicas a las que están sometidos los átomos mientras que el último incluye en cierta manera toda la información estructural recogida sobre la proteína mediante los experimentos de RMN. Esta información puede contener valores de ángulos diedros, constantes de acoplamiento, distancias interprotónicas, etc. Sin embargo, no representa una fuerza física real sobre ningún átomo de la molécula. El modo en que este subpotencial de RMN describe las restricciones introducidas varía de un método a otro, incluso de un programa de cálculo a otro, y es el responsable de la adecuación de la estructura final a la información experimental introducida. Un modo muy habitual de describirlas es mediante potenciales armónicos deformados de fondo plano ya que de este modo se consigue que mientras el parámetro estructural en cuestión se encuentre dentro de los límites impuestos la contribución a la función global de potencial sea nula.

Aunque se puede utilizar para este tipo de cálculos una estructura arbitraria, como por ejemplo, la estructura totalmente extendida [48] (todos los ángulos diedros iguales a 180^o), en la práctica se utilizan a menudo estructuras obtenidas mediante minimización o mediante geometría de distancias. Gracias a la energía cinética aportada a la proteína durante la simulación de dinámica, el problema de los mínimos locales se puede evitar de un modo relativamente simple. De todos modos, la hipersuperficie de potencial es demasiado extensa como para asegurar en ningún momento que los mínimos alcanzados estén siquiera muy próximos al mínimo global, aunque el método explora mejor el espacio conformacional accesible.

El templado simulado implica el aumento de temperatura del sistema y el reescalamiento de las constantes de fuerza de los parámetros estructurales para dar un mayor peso relativo a las restricciones experimentales [49, 15]. A continuación, se procede a enfriar el sistema y reescalar todas las constantes a sus valores originales de modo progresivo y muy lento. De este modo la exploración del espacio conformacional es mayor ya que por una parte la mayor velocidad de los átomos hace que exploren posiciones más diferenciadas y por otra parte, el reescalamiento de las constantes permite a los átomos moverse más libremente y evitar impedimentos en la exploración como puedan ser repulsiones de van der Waals excesivas. El enfriado lento y progresivo permite al sistema localizar un mínimo más general que en el caso de un enfriado rápido o de una simple dinámica molecular res-

tringida.

3.2 Métodos experimentales

3.2.1 Preparación de la muestra

La plastocianina $Synechocistys$ fué aislada de la cianobacteria *Synechocistys* sp PCC 6803 y fué purificada como se describe en la bibliografía [50]. La pureza de la plastocianina fué comprobada mediante PAGE y espectroscopía Ultravioleta visible. La plastocianina / Cu(II) purificada tiene una relación de absorbancias A_{275}/A_{597} de 2.2. La concentración de plastocianina fué determinada espectrofotométricamente utilizando un coeficiente de extinción molar de $\epsilon_{597} = 4.5 \cdot 10^3 M^{-1} cm^{-1}$ [50]. Las muestras para los experimentos de Resonancia Magnética Nuclear bidimensional tuvieron una concentración de 4 a 5 mM en proteína, la cual se disolvió en una disolución 0.5 mM de tricina / KOH en D_2O y en otra disolución al 95 % en H_2O y al 5 % en D_2O a pH 5.4 para los experimentos en agua pesada y agua respectivamente. La plastocianina / Cu(II) fué reducida en el tubo de RMN en el que se iban a realizar los experimentos mediante ditionito de sodio en exceso bajo atmósfera inerte de Argón. El tubo se selló a continuación con un sello de goma aislante.

3.2.2 Experimentos de RMN realizados

Los espectros de RMN fueron obtenidos en un espectrómetro Unity de Varian a 400 MHz. Se acumularon diferentes espectros DQF-COSY en D_2O y en H_2O. En la tabla 3.1 se pueden observar todos los experimentos que se realizaron. También se realizaron experimentos TOCSY a diferentes tiempos de mezcla en los que se usó MLEV-17. Los tiempos de mezcla fueron de 15 a 90 ms. Los experimentos NOESY realizados también se obtuvieron a diferentes temperaturas y tiempos de mezcla como se puede apreciar en la tabla 3.1. Todos los experimentos en H_2O fueron adquiridos bajo presaturación de la señal del agua y en el caso de los experimentos en D_2O con presaturación de la señal residual de la resonancia de HDO. Los tiempos de reciclado para los experimentos NOESY, COSY y TOCSY estuvo en un intervalo de 600 a 900 ms. El número de acumulaciones varió de 32 a 192. El número de puntos en la dimensión t_2 se mantuvo en el intervalo de 2048 a 4096 y el número de puntos en la dimensión t_1 en el intervalo de 256 a

Campo (MHz)	Experimento	Temp.	ω_1	ω_2	Tiempo de Mezcla	Disolvente
400	DQF-COSY	30	6000	6000		D_2O
	DQF-COSY	30	6000	6000		H_2O
	DQ-COSY	30	6000	24000		H_2O
	TOCSY	30	6000	12000	15,45,90	D_2O
	TOCSY	20,30	6000	12000	15,45,90	H_2O
	NOESY	30,40	6000	12000	50,90,150,225	D_2O
	NOESY	20,30,40	6000	12000	50,90,150,225	H_2O
600	DQF-COSY	30	6000	6000		H_2O
	TOCSY	30	6000	12000	15,90	H_2O
	NOESY	30	6000	12000	50,150	H_2O

Tabla 3.1: Tabla de experimentos de RMN realizados sobre la muestra de Plastocianina. *Synechocistys*.

512. Los espectros NOESY y TOCSY fueron transformados utilizando funciones ventana de campana seno cuadrado desplazadas 45^o o 60^o en ambas dimensiones. El número de puntos utilizado para el procesado de los espectros fue de 2048 o 4096 en ambas dimensiones y se utilizó rellenado de ceros para aumentar la resolución digital. Los espectros DQF-COSY fueron transformados utilizando una función ventana de seno cuadrado sin desplazar y se utilizó rellenado de ceros hasta alcanzar un número d epuntos de $2K$x$2K$.

Para resolver la asignación de algunas resonancias solapadas se realizaron experimentos adicionales en un espectrómetro de mayor campo. Se obtuvieron espectros DQF-COSY, TOCSY y NOESY a diferentes tiempos d emezcla en H_2O a 30^o de temperatura en un espectrómetro Brucker AMX 600 a 600 MHz de campo. Todos los espectros se adquirieron en modo sensitivo a la fase con incrementos de fase proporcionales al tiempo (TPPI)[51] para la detección de cuadratura en t_1. Los experimentos TOCSY fueron recogidos con una secuencia de mezclado “Clean CITY”. Para los experimentos NOESY y TOCSY se utilizó la secuencia de filtrado de señal de agua WATERGATE [52]. En los experimentos DQF-COSY, se utilizó irradiación selectiva para eliminar la señal del disolvente durante el periodo de espera de relajación.

3.2.3 Restricciones estructurales

Se obtuvieron distancias interprotónicas de los espectros NOESY realizados. La asignación de las resonancias de los espectros NOESY se llevó a cabo mediante un proceso iterativo de obtención de restricciones y comparación con estructuras de primera generación. En la primera etapa de dicha asignación se obtuvieron de modo completamente manual 632 NOEs unívocos, es decir, sin ambigüedades posibles, que se utilizaron para la obtención de estructuras de primera generación. De estos NOEs, 450 fueron de larga distancia, 113 secuenciales y 69 intraresiduo. Las ambigüedades iniciales se resolvieron mediante el análisis de las estructuras de baja resolución de primera generación. Las intensidades de los picos cruzados NOE se determinó mediante medición del volumen de pico estimado con el modulo *peak-picking* del programa GIFA [57]. Dicho volumen se obtiene por comparación con una región del espectro vacía para la evaluación del ruido. El calibrado de los picos cruzados NOE constó de dos etapas. En la primera de ellas, todos los picos obtenidos se asociaron a restricciones con límite inferior 1.8 Å y límite superior 5.0 Å en general y 5.5 Å para los pseudoátomos. En la etapa inicial sólo se incluyeron las restricciones interesiduo asociadas a pseudoátomos correspondientes a intensidades medias y fuertes de modo que 5.5 Å incluye las correciones habituales [53] de modo conservador. El límite superior de distancia correspondiente a la restricción media entre pseudoátomos con la máxima correción prevista [53] es inferior a 5.5 Å. De este modo, en la fase inicial la imprecisión asociada a los pseudoátomos se ha incluido evitando posibles sobreestimaciones. Las estructuras de primera generación obtenidas con dicho conjunto de restricciones permitieron evaluar los residuos de hoja β. A partir de dicho conjunto de residuos se estableció un criterio de distinción entre picos fuertes, medios y débiles. Los valores de intensidad utilizados para dicha separación se escogieron de modo que más del 85 % de los picos secuenciales $d_{\alpha N}$ en los residuos en hoja β fueran clasificados fuertes y más del 85 % del resto de picos secuenciales $d_{\alpha N}$ fueran clasificados como medios o fuertes [53]. Los límites superiores de las restricciones de distancia asociadas a los picos NOE fuertes, medios y débiles fueron 3.00 Å, 4.00 Å y 5.00 Å respectivamente. El límite inferior fue en todos los casos de 1.8 Å. Los límites superiores de las restricciones asociadas a los pseudoátomos en las etapas finales se determinaron mediante la asignación automática utilizando el programa ASNO [41] que contiene las correcciones de distancia asociadas a los grupos metilo y grupos metileno [53]. En los casos de solapamientos se aplicaron las restricciones correspondientes a picos cruzados débiles.

Residuo	$H_{\beta 2}$	$H_{\beta 3}$	$I_{H\alpha\beta 2}/I_{H\alpha\beta 3}/$	${}^3J_{\alpha\beta 2}/{}^3J_{\alpha\beta 3}$ F.B.	${}^3J_{\alpha\beta 2}/{}^3J_{\alpha\beta 3}$ Ps.
F16(F14)	2.42	3.04	3.90	3.72	4.22
E17(V15)	1.62	1.02	1.80	–	1.98
K24(S22)	1.40	1.76	3.20	–[a]	–[b]
E28(K26)	1.66	1.76	1.90	–	–[b]
K30(V28)	1.70	1.50	3.30	–[a]	–[b]
W31(F29)	3.00	3.10	2.03	4.43	5.20
H39(H37)	2.90	3.65	2.72	3.30	3.00
K59(E60)	1.74	1.62	0.51	0.46	0.23
S68(T69)	3.70	3.57	1.05	–[a]	–[b]
F69(F70)	2.51	1.80	3.10	0.60	1.71
S71(V72)	3.50	3.83	4.05	–[a]	–[b]
E73(L74)	2.78	2.42	¿3	3.56	2.8
E75(T76)	1.88	2.04	2.21	–	–
Y79(Y80)	3.32	3.42	2.61	6.05	3.67
Y82(Y83)	3.50	3.70	4.60	–[b]	3.87
C83(C84)	3.42	2.95	0.40	1.43	1.65
E98(N99)	1.90	1.96	2.91	–[b]	–[b]

Tabla 3.2: Tabla de asignación estereoespecífica, relación entre intensidades $I_{H\alpha\beta 2}/I_{H\alpha\beta 3}/$ y comparación de la relación de constantes de acoplamiento ${}^3J_{\alpha\beta}$ con otras plastocianinas determinadas por RMN. [a]:; [b]: .

Se obtuvieron restricciones de ángulos diedros ϕ mediante el análisis de la estructura hiperfina de los picos cruzados DQF-COSY correspondientes al acoplamiento escalar $H_N - H_\alpha$. Se aplicaron restricciones sobre los ángulos de torsión ϕ, $-180 < \phi < -80$ para los residuos con un valor de ${}^3J_{\alpha N}$ superior a 8.5 Hz [54]. No se observaron valores de ${}^3J_{\alpha N}$ inferiores a 5.5 Hz por lo que no se aplicaron más restricciones sobre el ángulo de torsión ϕ. Se obtuvieron también restricciones de angulo de torsión de cadena lateral χ_1 como se puede observar en la tablas 3.2 y 3.3 en la sección 3.4.3 mediante la aplicación de la metodología desarrollada en el capítulo 5 para la asignación estereoespecífica.

La determinación de la conformación del enlace peptídico de las prolinas PRO18, PRO38, PRO76 y PRO85 se realizó mediante los NOE entre los protones H_α y H_N del residuo anterior y los protones H_α y H_δ de la prolina correspondiente [37]. En todos los casos salvo en la PRO38 se observaron los NOE $H_{Ni-1} - H_{\delta i}$ indicativos de una conformación trans de la prolina.

Residuo	Conformación χ_1	χ_1 T.P.	χ_1 F.B.	χ_1 *Poplar*	Homol.
F16(F14)	g^-	-72	-78	-74	-70
E17(V15)	g^-	-179	-176	-175.5	-65
K24(S22)	g^-	-67	30	-42	-177
E28(K26)	g^-	59	51	-167	-65
K30(V28)	g^-	-173	179	-172	-175
W31(F29)	g^-	-70	-66	-71	-63
H39(H37)	g^-	-54	-56	-76	-67
D51(A52)	g^-	-86	–	–	-168
K59(E60)	t	-70	-169	-55	-67
S68(T69)	g^+	68	45	50	-67
62					
F69(F70)	g^-	-77	172	180	166
S71(V72)	g^-	-71	64	60	64
E73(L74)	g^-	-63	-56	-55	-68
E75(T76)	g^-	-166	-42	-33	-68
Y79(Y80)	g^-	-61	-68	-65	-64
Y82(Y83)	g^-	62	61	44	62
C83(C84)	t	174	164	170	-173
E98(N99)	g^-	-175	-69	52	-72

Tabla 3.3: Tabla de asignación estereoespecífica. Comparación de los valores de χ_1 en otras plastocianinas con la conformación determinada mediante asignación estereoespecífica.

	$H_{Ni-1} - H_{\alpha i}$	$H_{\alpha i-1} - H_{\alpha i}$	$H_{Ni-1} - H_{\delta i}$	$H_{\alpha i-1} - H_{\delta i}$	Conformación
PRO18	M	-	D	D?	cis
PRO38	D	-	-	-	cis
PRO76	D	-	F	F	trans
PRO85	-	-	M	M	trans

Tabla 3.4: Tabla de picos diferenciadores de la conformación cis y trans del enlace peptídico de las prolinas. D: débil; M: medio; F: fuerte. ? Indica posible ambigüedad en la intensidad por solapamiento.

Los picos $H_{\alpha i-1} - H_{\alpha i}$ no se observaron en ninguno de los casos, aunque en las prolinas PRO38 y PRO18 no sea necesariamente indicativo de su no existencia debido a que la señal se ve solapada por la diagonal. En el caso de la PRO38 se observó un NOE débil entre $H_{Ni-1} - H_{\delta i}$ indicativo de una conformación cis. En el caso de la PRO18 se observó una conectividad NOE $H_{Ni-1} - H_{\delta i}$ de intensidad media indicador de conformación cis. En el caso de esta PRO18 se asignó conformación cis tras comparación de los NOE diferenciadores. Dicha asignación se ve apoyada por la observación, aunque muy débil, del pico cruzado NOESY $H_{\alpha}GLU17 - H_{N}SER19$ que está ausente en conformaciones trans de prolinas. Así, la conformación de las prolinas fue asignada tal y como se ve en la tabla 3.4. La conformación de las PRO18 y PRO38, que por otra parte son prolinas conservadas en un gran número de plastocianinas, fue asignada cis manteniéndose la conformación de las plastocianinas homólogas [55, 56].

Las restricciones estructurales incluídas se pueden observar en la tabla 3.5 en la que se ve también la clasificación de las mismas según su separación en número de residuos en intraresiduos, secuenciales (1 residuo de separación) y a larga distancia (más de un residuo de separación). También se puede apreciar que las restricciones asociadas a los puentes de hidrógeno mostrados en la tabla 3.6 corresponden a los protones de intercambio lento extraídos de la comparación entre los espectros en D_2O y H_2O.

Aunque no se incluyeron las restricciones de distancia asociadas al ión de cobre en los cáculos de las estructuras de RMN, se realizó un posterior cálculo de minimización de energía sobre la estructura de mínima energía de RMN introduciendo el átomo de cobre y las restricciones sobre el mismo derivadas de estudios realizados sobre la geometría de coordinación de dicho átomo en la plastocianina. Se asumió en dicho cálculo que el átomo de cobre está coordinado a los mismos grupos dadores que en el resto de plastocianinas, que son, δN de la HIS39 e HIS86, γS de la CYS83 y δS de la MET91 [10, 55, 30]. De estructura de mínima energía de RMN y

Distancias	Manual Inicial	ASNO [41]	Manual refinamiento
Total	632	1172	1398
Intraresiduo	69	310	409
Secuenciales	113	232	264
Larga distancia	450	630	725
P. de hidrógeno	42	42	42
Dihedros			
Cadena lateral χ_1	–	–	17
Cadena principal ϕ	–	39	39
Cadena principal ω	98	98	98

Tabla 3.5: Restricciones utilizadas en el cálculo de estructura de la Plastocianina $Synechocystis$ mediante RMN.

se le aplicaron 1000 pasos de minimización de energía manteniendo las siguientes restricciones estructurales relativas al cobre: $Cu - N_\delta HIS39$ (2.16 $\pm$ 0.2 Å), $Cu - S_\gamma CYS83$ (2.15 $\pm$ 0.2 Å), $Cu - N_\delta HIS86$ (2.37 $\pm$ 0.2 Å) y $Cu - S_\delta MET91$ (2.88 $\pm$ 0.2 Å). La estructura resultante no presentó ninguna diferencia significativa respecto a la original de mínima energía de RMN. Las distancias finales obtenidas fueron $Cu - N_\delta HIS39$ 2.22 Å, $Cu - S_\gamma CYS83$ 2.32 Å, $Cu - N_\delta HIS86$ 2.51 Å y $Cu - S_\delta MET91$ 3.01 Å. No hubo por lo tanto violaciones y las restricciones fueron satisfechas.

El valor de desviación cuadrática media de las coordenadas de los átomos pesados de los residuos implicados en la coordinación entre la estructura minimizada con cobre y la de mínima energía de RMN fue de 0.08 Å indicando muy poca variación de la estructura y que las restricciones de RMN derivadas experimentalmente son compatibles con las de coordinación al cobre. Además, los valores de desplazamiento químico de la HIS39, por ejemplo, apoya la coordinación de este residuo al átomo de cobre. Por otra parte, los ángulos de equilibrio para la coordinación del cobre a sus ligandos fueron bastante similares a los descritos en la bibliografía. Por ejemplo, el ángulo $X - Cu - X$, con X igual a S o a N presenta un valor de 123^o para $S_\gamma CYS83 - Cu - N_\delta HIS39$ bastante próximo a los 110^o descritos en la bibliografía (constante de equilibrio de 10 $kcalmol^{-1}rad^{-2}$). El ángulo $Cu - S_\gamma CYS83 - C_\beta CYS83$ con 132^o muestra también muy poca diferencia respecto al valor de equilibrio de 120^o para $Cu - S - C$ (constante de equilibrio de 50 $kcalmol^{-1}rad^{-2}$) al igual que el ángulo $Cu - N_\delta HIS86 - C_\gamma HIS86$ con 132^o muy próximo al valor de equilibrio de 127^o para $Cu - N - C$ (50

5 VAL HN – 30 LYS OC	6 LYS HN – 17 GLU OC
7 MET HN – 32 VAL OC	8 GLY HN – 15 VAL OC
9 SER HN – 13 ALA OC	* 12 GLY HN – 9 SER OC
15 VAL HN – 13 ALA OC	17 GLU HN – 6 LYS OC
21 VAL HN – 94 LYS OC	23 ILE HN – 96 VAL OC
26 GLY HN – 73 PHE OC	* 27 GLU HN – 24 LYS OC
29 VAL HN – 71 SER OC	30 LYS HN – 3 ALA OC
31 TRP HN – 69 PHE OC	32 VAL HN – 5 VAL OC
33 ASN HN – 67 GLU OC	34 ASN HN – 7 MET OC
35 LYS HN – 10 ASP OC	42 VAL HN – 82 TYR OC
44 ALA HN – 80 THR OC	* 56 LEU HN – 41 ILE OC
60 GLY HN – 57 SER OC	* 62 ALA HN – 39 HIS OC
66 GLY HN – 33 ASN OC	69 PHE HN – 31 TRP OC
68 SER HN – 32 VAL OC	77 GLY HN – 97 VAL OC
79 TYR HN – 95 VAL OC	81 TYR HN – 93 GLY OC
83 CYS HN – 91 MET OC	* 89 ALA HN – 86 HIS OC
90 GLY HN – 86 HIS OC	* 90 GLY HN – 87 ARG OC
91 MET HN – 86 HIS OC	* 92 VAL HN – 90 GLY OC
93 GLY HN – 81 TYR OC	94 LYS HN – 19 SER OC
95 VAL HN – 79 TYR OC	96 VAL HN – 21 VAL OC
97 VAL HN – 77 GLY OC	98 GLU HN – 23 ILE OC

Tabla 3.6: Tabla de puentes de hidrógeno en la Plastocianina $Synechocystis$ asociados con intercambio lento observado en los experimentos en D_2O. Los puentes de hidrógeno que no están asociados a la estructura de hoja β están marcados con *.

$kcalmol^{-1}rad^{-2}$).

3.3 Métodos computacionales

3.3.1 Programa de Procesado de datos de RMN. Gifa.

El programa GIFA [57] es un programa de diferentes aplicaciones a datos de RMN. Ha sido desarrollado en diferentes grupos dedicados a la Resonancia Magnética Nuclear en Francia. Está diseñado para la transformación, visualización y análisis de conjuntos de datos de RMN en 1D, 2D y 3D. El programa incluye todos los elementos clásicos de procesado, visualización, impresión y análisis de datos de RMN como pueden ser diferentes funciones ventana aplicables, lectura de los formatos de datos más utilizados (Brucker-UX, Varian, Texto), transformaciones de Fourier y de Hilbert directas, inversa, complejas y reales, procesado por máxima entropía, predicción lineal y una gran cantidad de operaciones fácilmente configurables. Además se pueden crear funciones adicionales de un modo sencillo e intuitivo.

En el procesado de datos se ha utilizado para el procesado de los espectros DQF-COSY una función seno cuadrado sin desfase inicial junto a una función gaussiana de coeficiente 10 y 4 en las dos dimensiones (GM1 4 y GM2 10 en la correspondiente función del GIFA) y se ha rellenado de ceros hasta alcanzar un número de puntos de 8192x2048. Para los espectros NOESY y TOCSY se aplicaron funciones ventana seno cuadrado con un desfase de 30^o y 45^o jutno a una función gaussina de coeficiente 10 en t_2 y 2 en t_1 (GM1 2 y GM2 10 en la correspondiente función del GIFA) y un rellenado de ceros hasta alcanzar un número de puntos de 4096x2048. El análisis y asignación de los picos DQF-COSY, NOESY y TOCSY se realizó utilizando el programa paralelo XMCINDY desarrollado por el mismo grupo de trabajo que el programa GIFA. La recolección de picos y medida de volumenes de picos NOESY para el calibrado de distancias interprotónicas y la medida de volumenes de picos TOCSY para la asignación estereoespecífica se realizó con el módulo “assignment” del programa GIFA.

3.3.2 Geometría de distancias. Programa DIANA.

El programa DIANA [60], acrónimo de *DIstance geometry Algorithm for NMR Applications,* es uno de los programas de mayor tradición en el cálculo de

estructuras de biopolímeros mediante geometría de distancias. El objetivo del programa es el cálculo eficiente de conformaciones de biomacromoléculas, mediante la utilización de distancias interatómicas y ángulos diedros que pueden ser obtenidos de medidas de RMN. El algoritmo está basado en la minimización de una función objetivo variable, en la que los grados de libertad son los ángulos diedros susceptibles de rotación en la macromolécula (espacio de los ángulos de torsión).

Las librerías del programa contienen información estructural y topológica sobre los aminoácidos naturales y sobre los nucleótidos naturales junto a algunos residuos sintéticos de uso frecuente. Esta información comprende distancias de enlace, ángulos de enlace, geometría ideal e información adicional para el desarrollo del cálculo.

El programa necesita como punto de partida un fichero que contenga la secuencia de aminoácidos o nucleótidos y ficheros en los que se encuentren las restricciones estructurales a introducir en el cálculo. En el fichero de comandos es necesario indicarle al programa si se va a utilizar una estrategía de minimización estándar o si se va a modificar y en este último caso donde se encuentran los parámetros de minimización modificados. Mediante un número aleatorio introducido por el usuario el programa generará estructuras al azar y de un modo progresivo irá introduciendo las restricciones estructurales impuestas en la función objetivo e irá minimizando esta en cada iteración. Las estructuras finales serán simplemente las estructuras generadas aleatoriamente que satisfagan en la medida de lo posible las restricciones impuestas en la función de minimización.

Una estrategia bastante utilizada en la geometría de distancias es la estrategia conocida como REDAC (*REdundant Dihedral Angle Constraints*) [61]. Esta estrategia consiste en la utilización de restricciones de ángulos diedros obtenidas de los cálculos preliminares redundantes con las restricciones estructurales impuestas. En las estructuras preliminares obtenidas mediante DIANA, pueden existir ciertos patrones en los ángulos diedros que se repitan a lo largo de un número determinado de estructuras. Tal es el caso, por ejemplo, de un dihedro ϕ que tome un valor dentro de un intervalo de 20^o en torno a un promedio en 10 de las 20 estructuras obtenidas. Se puede considerar, que esta distribución de ese ángulo en concreto puede corresponder a un determinado patrón estructural y por lo tanto se puede incluir en sucesivos cálculos como restricción adicional. Así, si se realizan varios cálculos consecutivos y en cada uno se utilizan este tipo de restricciones extraidas del cálculo anterior, las estructuras obtenidas mostrarán

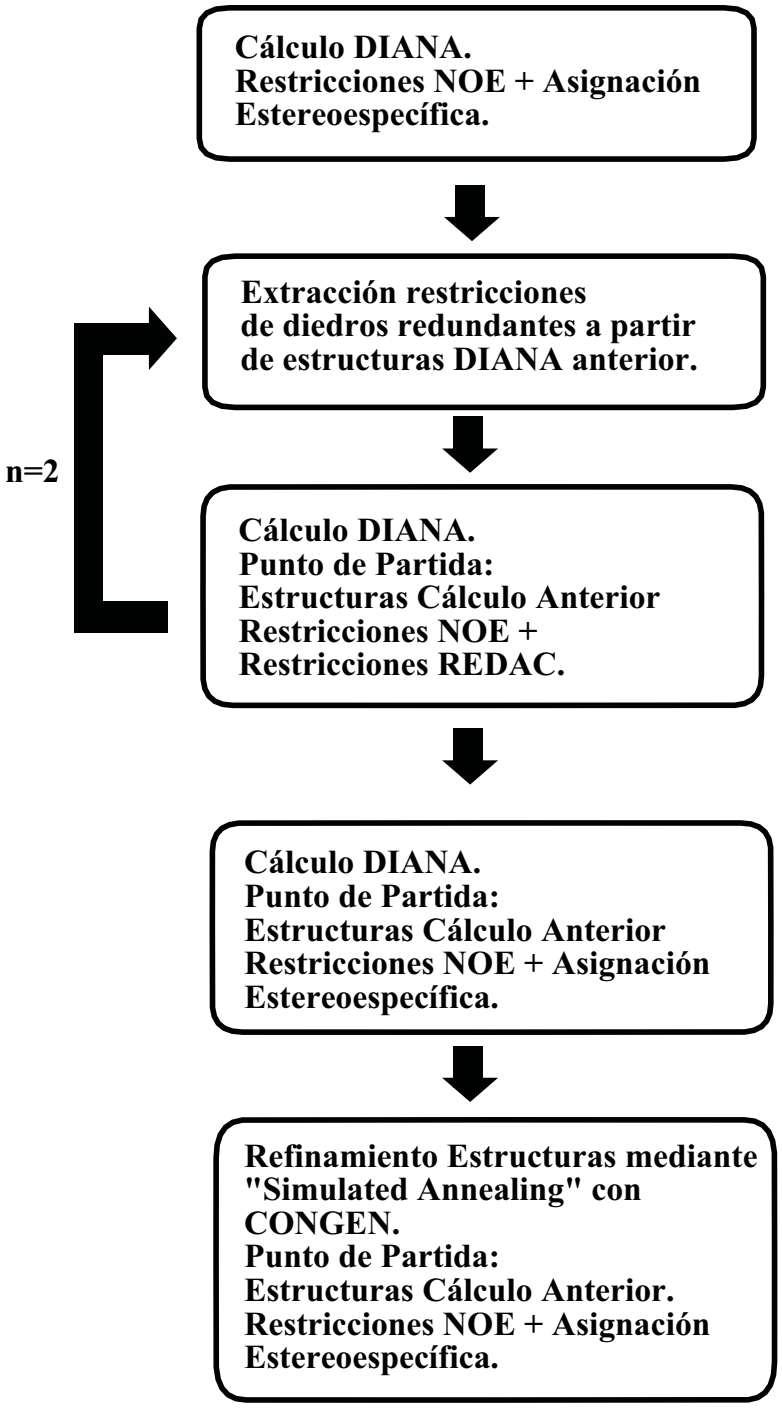


Figura 3.9: Estrategia utilizada en los cálculos de estructuras de plastocianina *Synechocistys* mediante RMN.

un grado de convergencia mayor. Para facilitar el proceso, en cada cálculo se toman como estructuras de partida, las estructuras finales del cálculo anterior en lugar de estructuras aleatorias. De este modo se puede generar un conjunto de estructuras convergentes y compatibles con las restricciones impuestas que se puede utilizar como punto de partida para un cálculo en el que únicamente se apliquen las restricciones experimentales. Gracias a la convergencia del punto de partida del cálculo, la convergencia en las estructuras finales es más probable.

En los cálculos realizados en este estudio se ha utilizado una estrategia REDAC [61] de cuatro etapas tal y como se describe en la figura 3.9. Se han realizado hasta 5 cálculos diferentes con un número de restricciones creciente para la asignación automática de NOEs mediante el programa ASNO (*ASsign NOes*) [41] (ver tabla 3.5. El primer cálculo de REDAC se realizó utilizando los NOEs extraídos del espectro NOESY mediante asignación manual. Estos NOEs no presentaron problemas de solapamiento o posible doble asignación por lo tanto se pueden considerar unívocos. A partir de ahí, las estructuras obtenidas en cada nivel de restricciones se utilizaron para la asignación adicional de nuevos NOEs, resolviendo las posibles ambiguedades mediante comparación de las distancias teóricas de cada NOE y las medidas en las estructuras de primera generación. El programa ASNO realiza esta comparación de modo automático y proporciona un listado de NOEs compatibles con las estructuras utilizadas como plantilla, los desplazamientos químicos y los picos cruzados observados.

3.3.3 Refinamiento de estructuras. Templado simulado.

Aunque las estructuras calculadas mediante el programa DIANA satisfacen en la medida de lo posible las restricciones estructurales impuestas, es necesario utilizar métodos que realicen una mayor exploración del espacio conformacional para asegurar un resultado óptimo. Además, los únicos grados de libertad en los cálculos de geometría de distancias con DIANA son los ángulos diedros [60]. Es necesario pues acudir a métodos en que se aumente el número de variables a ángulos de enlace y distancias de enlace y que al mismo tiempo, utilice funciones objetivo de carácter más físico como puede ser la función potencial. Uno de los métodos más utilizados es el templado simulado mediante dinámica molecular restringida. Utilizando como punto de partida las estructuras obtenidas mediante geometría de distancias aceleramos el cálculo de estructuras refinadas y ayudamos a la convergencia final del resultado.

El programa utilizado para los cálculos de templado simulado mediante dinámica molecular restringida ha sido el CONGEN [22] (ver página 27). Se ha introducido una constante dieléctrica dependiente de la distancia para simular el disolvente. La estrategia utilizada ha sido prácticamente la misma que la utilizada en los cálculos de homología del capítulo anterior sin la aportación de los métodos de búsqueda conformacional para las regiones poco definidas (ver 2.5). Se han introducido las modificaciones adecuadas para calcular estructuras a partir de conformaciones plegadas ya y no totalmente extendidas [48]. Además, el número de restricciones es considerablemente menor ya que han sido obtenidas por métodos experimentales. Sobre cada estructura final obtenida de geometría de distancias se ha aplicado un ciclo completo de templado simulado.

La fuerza aplicada a las restricciones ha sido de 1 en las unidades estándar del programa DIANA para la geometría de distancias. En el templado simulado, se ha utilizado la posibilidad que ofrece el programa CONGEN para variar la forma de la función de restricción NOE y se ha ido convirtiendo en una función de pozo más estrecho progresivamente en el templado simulado, al tiempo que se aumentaba su fuerza hasta un máximo de 100 $Kcal/mol$ Å^2. En los cálculos de templado simulado se ha limitado la fuerza máxima a 600 $Kcal/mol$ Å^2. La fuerza aplicada a los ángulos diedros ha sido de 100 $Kcal/mol^o$ para los diedros ϕ y χ_1 y de 500 $Kcal/mol^o$ para los diedros ω con el objetivo de mantener la planaridad del enlace peptídico a altas temperaturas.

3.4 Resultados y discusión

3.4.1 Asignación secuencial

La plastocianina de $Synechocistys$ sp. PCC 6803 contiene 98 aminoácidos. Los sistemas de espín de los diferentes aminoácidos fueron clasificados e identificados usando los 3 primeros pasos de la estrategia estándar en 4 pasos descrita en la sección 3.1.2. La asignación de los sistemas de espín brevemente se puede resumir en dos etapas principales. La primera es la asignación de las conectividades primarias $H_N - H_\alpha$ en H_2O mediante el espectro DQF-COSY. La segunda implica la asignación de los sistemas de espín con desplazamientos químicos característicos y patrones geométricos particulares en el espectro TOCSY.

Se encontró un total de 85 picos cruzados $H_N - H_\alpha$ en la región carac-

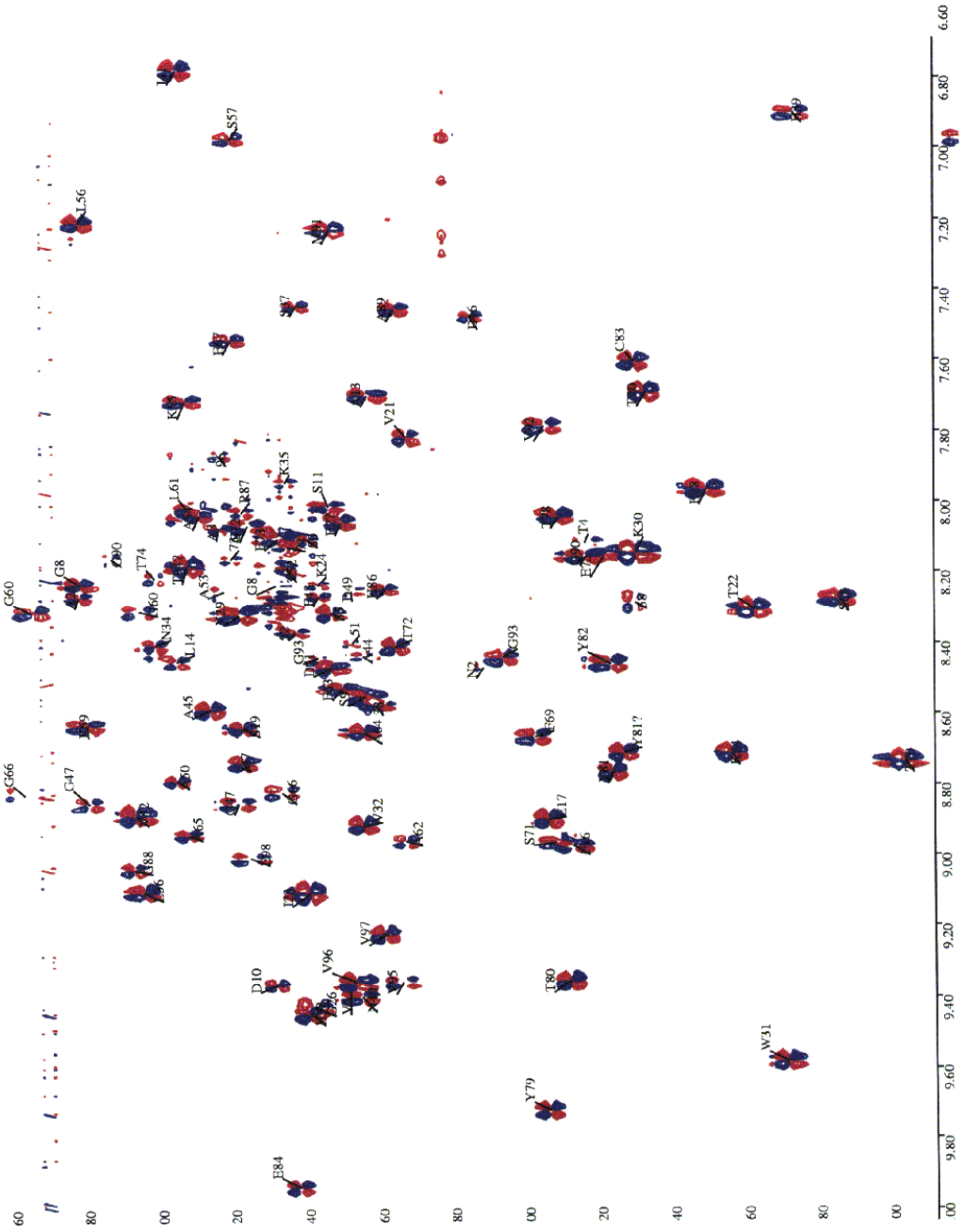


Figura 3.10: Zona característica del espectro DQF-COSY correspondiente a los picos cruzados $H_N - H_\alpha$.

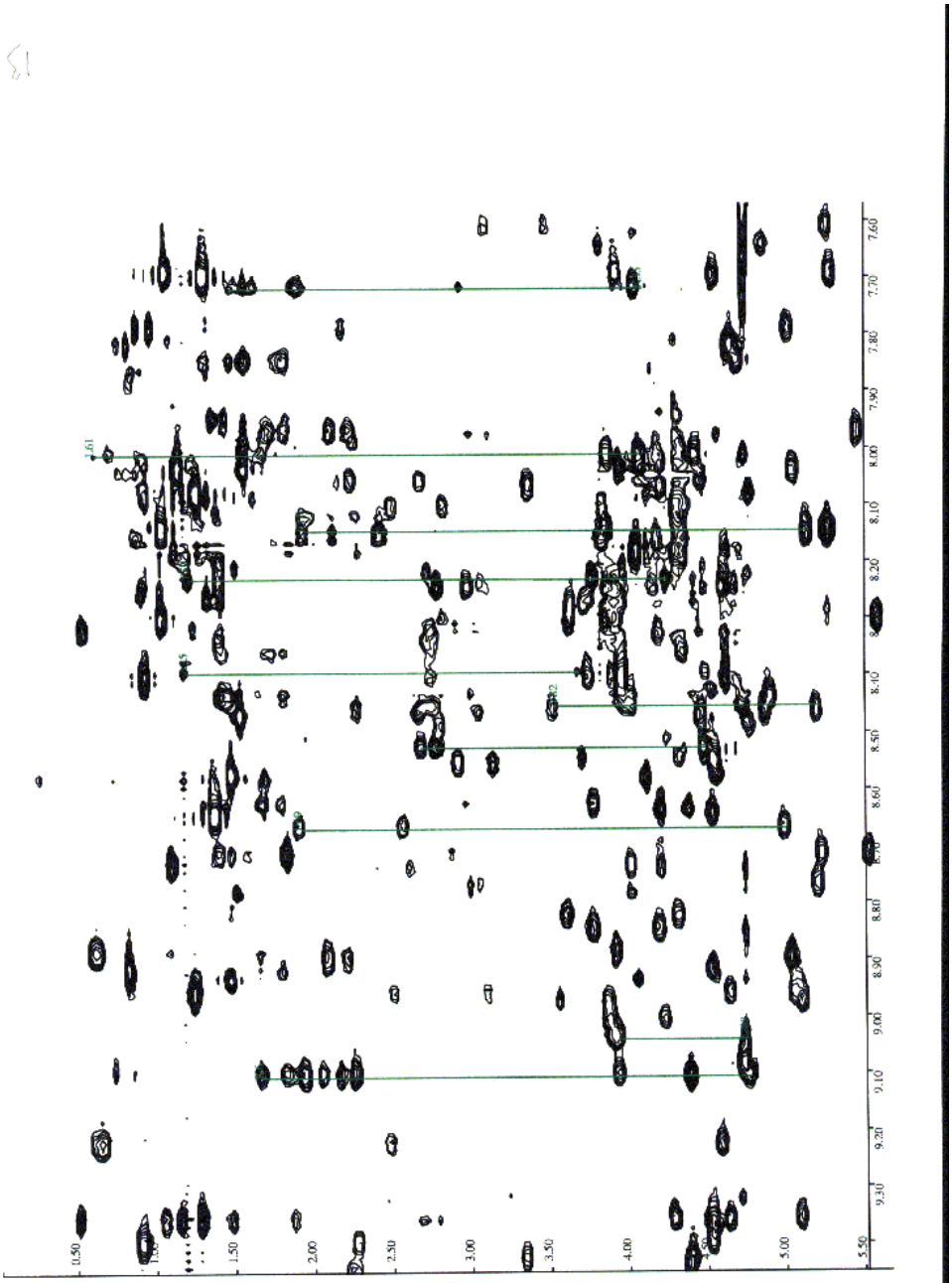


Figura 3.11: Zona característica del espectro TOCSY correspondiente a los picos cruzados $H_N - H_\alpha$ y $H_N - H_{Largos}$.

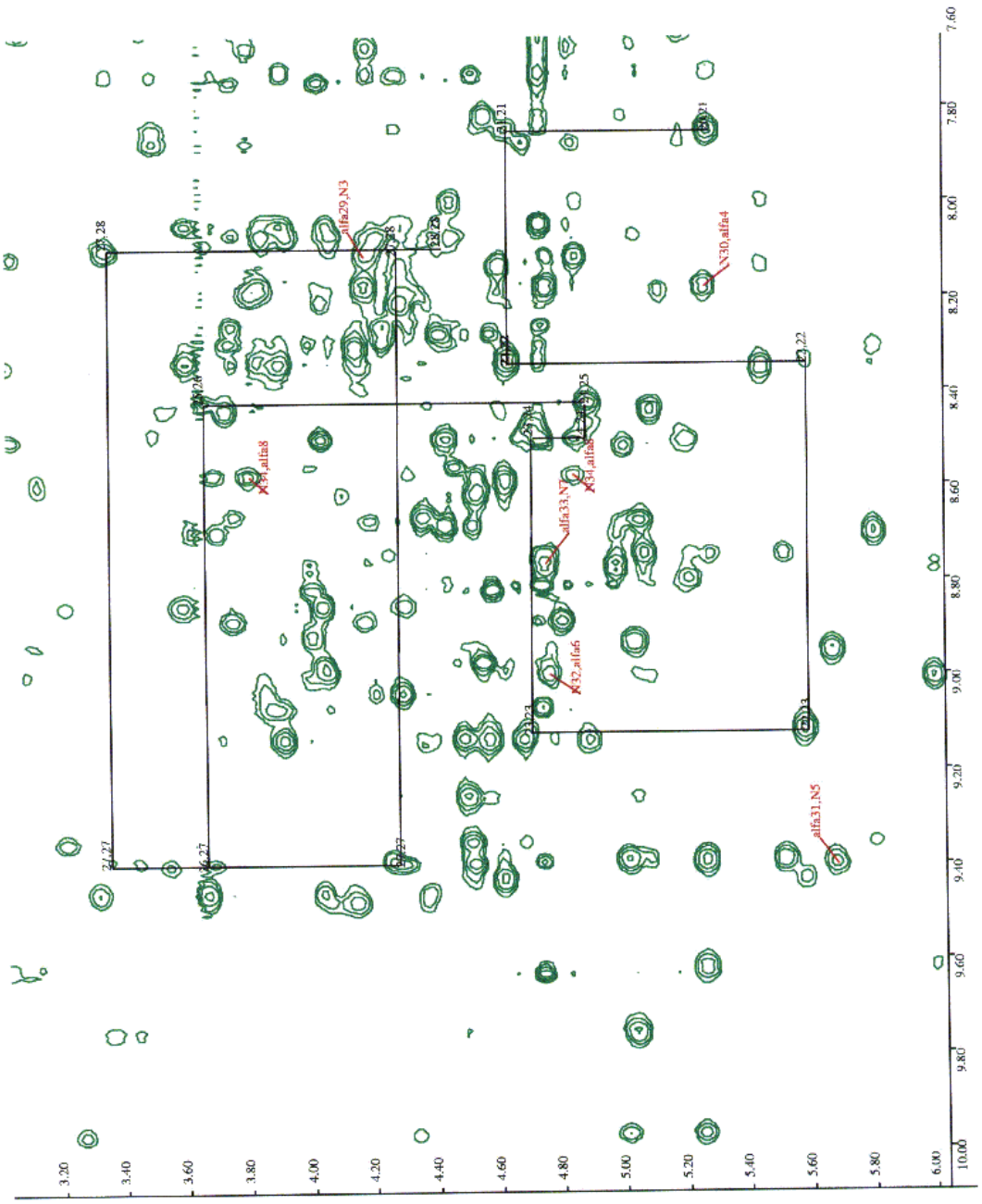


Figura 3.12: Zona característica del espectro NOESY correspondiente a los picos cruzados $H_N - H_\alpha$.

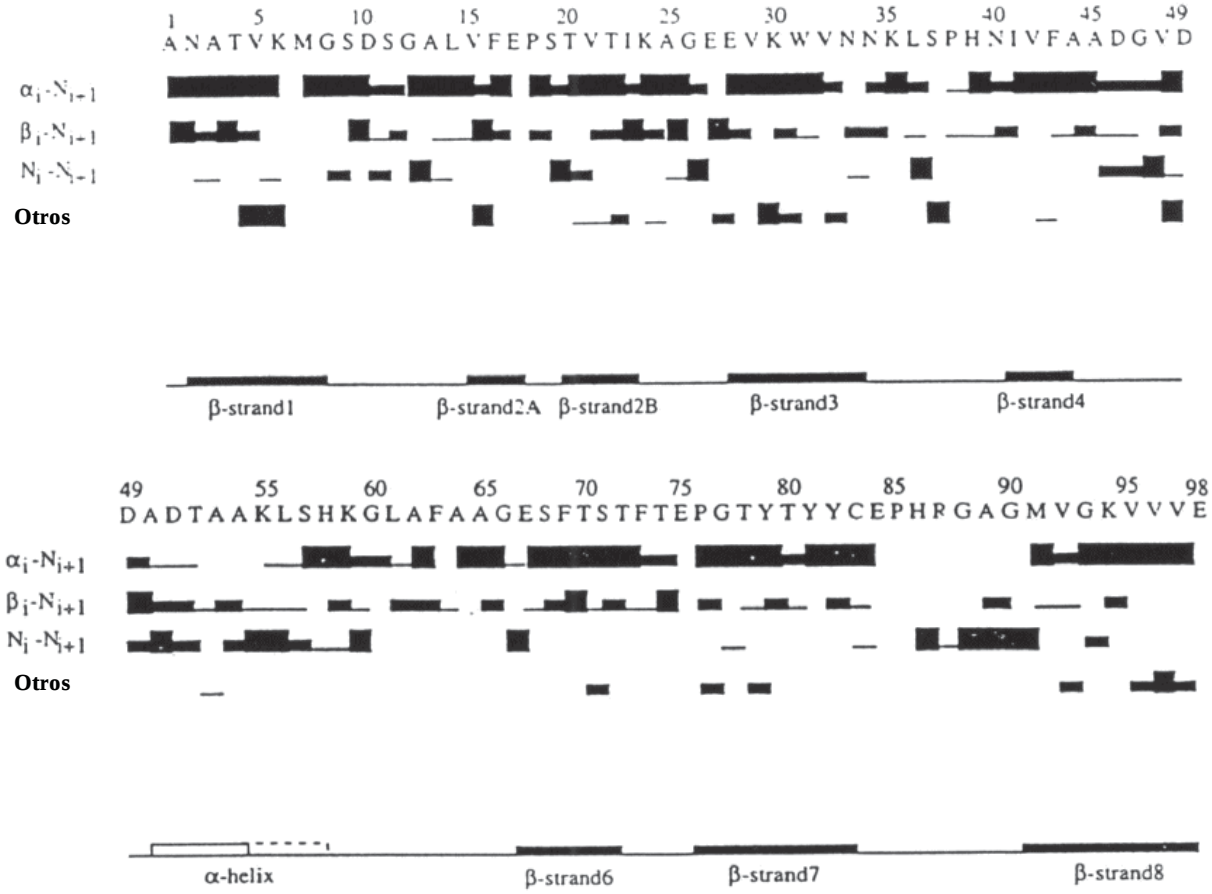


Figura 3.13: Representación esquemática de las conectividades utilizadas para la asignación secuencial de sistemas de espín y para la asignación preliminar de estructura secundaria de la Plastocianina *Synechocistys*.

terísticas de $H_N - H_\alpha$ del espectro DQF-COSY recogido a 30^oC en H_2O (ver figura 3.10). La plastocianina $Synechocistys$ tiene 4 prolinas y 10 glicinas por lo que se deberían observar 103 picos cruzados $H_N - H_\alpha$ en dicha región. La diferencia entre ambos valores se debe esencialmente a solapamientos de picos cruzados en el espectro. Basándonos en los patrones geométricos encontrados en el espectro TOCSY y en los espectros DQF-COSY, los residuos se pueden clasificar en los grupos siguientes: 49 residuos con un patrón de desplazamientos químicos unívoco (10 glicinas, 13 alaninas, 9 treoninas, 11 valinas 4 leucinas y 2 isoleucinas), 28 residuos con un patrón de desplazamientos químicos correspondiente a un sistema de espín AMX (7 serinas, 1 cisteína, 4 aspárticos, 5 fenilalaninas, 3 tirosinas, 3 histidinas y 1 triptófano), 17 residuos de cadena larga (7 glutámicos, 7 lisinas, 2 metioninas y 1 arginina) y finalmente 4 prolinas que no poseen protón en la cadena principal. El análisis de las regiones $H_N - H_\alpha$ y $H_N - CadenaLateral$ del espectro TOCSY (ver figura 3.11) muestra la presencia de 90 sistemas de espín de los 93 residuos observables. Las ambigüedades en la asignación de los sistemas de espín fueron resueltas mediante un análisis minucioso de los patrones de conectividades secuenciales en el espectro NOESY (ver figura 3.12) y mediante comparación de dichos patrones con las estructuras tridimensionales de primera generación obtenidas con geometría de distancias.

Los sistemas de espín correspondientes a las prolinas fueron asignados mediante el espectro NOESY, donde las conectividades se establecieron usando los NOEs secuenciales entre los protones α del aminoácido precedente y el $\delta - CH_2$ de la prolina ($H_{\alpha i-1} - H_{\delta i}$) y entre los protones α de la prolina y el protón amida de la cadena principal del residuo siguiente ($H_{\alpha i} - H_{Ni+1}$). Las conectividades NOEs utilizadas para la asignación de los residuos PRO18, PRO38, PRO76 y PRO85 se pueden observar en la figura 3.13.

Los sistemas de espín correspondientes a los anillos de los residuos con carácter aromático fueron asignados mediante la identificación de las estructura en multiplete de los picos cruzados en el DQF-COSY y en el TOCSY así como mediante la interpretación de los picos cruzados en el NOESY. La asignación de las resonancias de protones aromáticos del único triptófano presente en la secuencia fué confirmada por la existencia de picos cruzados NOE entre los protones amina de la cadena lateral y los dos grupos CH, δ_1 y δ_2. Además, se detectaron picos cruzados NOE entre el protón $H_{\delta 1}$ y el protón H_β.

La asignación de los protones amida intercambiables de las cadenas la-

terales de las asparaginas, $H_{N\delta}$ se realizó mediante los picos cruzados NOESY con sus correspondientes protones H_β.

Residuo	H_N	H_α	H_β	Otros
ALA1		4.02	1.50	
ASN2	8.44	4.82	3.02	7.52,6.82 (δ)
ALA3	8.05	4.11	1.20	
THR4	8.15	5.21	3.76	0.97 (γ_1)
VAL5	9.34	4.45	1.42	1.10,0.45 (γ)
LYS6	9.07	4.75	1.92	1.62 (γ), 1.82 (δ), 2.92 (ϵ)
MET7	8.71	4.16	1.78	2.58 (γ)
GLY8	7.63	4.80,3.81		
SER9	8.49	4.48	4.30,3.65	
ASP10	9.37	4.23	2.66,2.72	
SER11	8.03	4.35	4.01,3.82	
GLY12	8.04	4.15,3.30		
ALA13	7.68	4.48	1.25	
LEU14	8.46	4.04	1.48,1.60	1.08 (γ), 0.14,0.67 (δ)
VAL15	7.36	4.78	2.18	0.80,0.58 (γ)
PHE16	8.87	5.08	3.04,2.42	6.90 (δ), 6.97 (ϵ) 7.52 (ζ)
GLU17	8.86	4.97	1.62,2.02	2.12 (γ)
PRO18		5.02	2.24,2.58	2.02,1.62 (γ), 3.74,3.94 (δ)
SER19	8.63	4.15	4.33	
THR20	7.66	5.24	3.87	1.03 (γ)
VAL21	7.80	4.58	1.85	0.76,0.70 (γ)
THR22	8.27	5.55	3.82	1.00 (γ)
ILE23	9.07	4.70	2.22	1.02 (γ_1), 0.40 (γ_2), 0.72 (δ)
LYS24	8.48	4.86	1.40,1.48	1.60 (γ), 1.76 (δ), 2.95 (ϵ)
ALA25	8.38	3.66	1.13	
GLY26	9.41	4.32,3.27		
GLU27	8.04	4.42	2.19	2.70 (γ)
GLU28	7.90	5.38	1.76,1.66	2.06,1.99 (γ)
VAL29	8.30	4.12	1.18	0.48,-0.62 (γ)
LYS30	8.11	5.23	1.70	1.50 (γ), 1.90 (δ), 2.78 (ϵ)
TRP31	9.57	5.63	3.00,3.10	6.65 (δ), 8.99 (ϵ_{N1}), 7.38 (ϵ_3), 6.12 (η_2), 7.15 (η_3), 6.97 (η_2)
VAL32	8.92	4.48	1.75	0.78 (γ)
ASN33	9.31	4.75	3.20,2.46	6.78,5.76 (δ_N)
ASN34	8.49	3.99	2.25	
LYS35	7.85	4.58	1.55,1.42	1.28 (γ), 1.74 (δ), 2.98 (ϵ)
LEU36	9.19	3.85	1.80	1.62 (γ)
SER37	7.33	4.00	3.93	

Continua en la página siguiente

Continua de la página anterior

Residuo	H_N	H_α	H_β	Otros
PRO38		4.00	1.80	1.24,1.50 (γ), 2.42,2.64 (δ)
HIS39	6.80	5.81	3.56,2.50	7.38 (δ_2), 6.38 (ϵ_1), 11.56 (ϵ_{N2})
ASN40	9.46	4.52	2.64	
ILE41	6.73	3.98	0.99	0.50,0.28 (γ_1), -0.68 (γ_2), -0.15 (δ)
VAL42	8.93	3.85	1.08	0.61,0.56 (γ)
PHE43	8.25	4.38	3.00,2.82	7.12 (δ), 6.95 (ϵ_1), 7.02 (ζ_3)
ALA44	8.44	4.38	1.39	
ALA45	8.61	4.48	1.33	
ASP46	7.48	4.78	2.52,2.68	
GLY47	8.81	4.12,3.70		
VAL48	7.19	4.38	2.09	0.94,0.98 (γ)
ASP49	8.23	4.53	2.72,2.91	
ALA50	8.76	3.95	1.46	
ASP51	8.50	4.42	2.72,2.62	
THR52	8.21	4.20	3.70	1.16 (γ)
ALA53	8.26	4.10	1.27	
ALA54	8.02	4.03	1.52	
LYS55	7.72	3.98	1.85	1.46 (γ), 1.56 (δ), 2.88 (ϵ)
LEU56	7.19	3.69	1.36	0.90,0.18 (γ), -0.02,-0.38 (δ)
SER57	6.97	4.10,4.05		
HIS58	9.45	4.32	2.20	6.32 (δ_1), 8.71 (ϵ_1)
LYS59	8.56	3.71	1.74,1.62	1.32 (γ), 1.62 (δ), 2.88 (ϵ)
GLY60	8.27	3.92,3.56		
LEU61	7.98	4.04	1.70,1.74	1.60 (γ), 0.89,0.58 (δ)
ALA62	8.97	4.57	1.18	
PHE63	8.56	4.46	3.06,2.91	7.24 (δ), 7.29 (ϵ), 7.29 (ζ)
ALA64	8.71	4.52	1.44	
ALA65	8.90	4.00	1.42	
GLY66	8.81	4.28,3.55		
GLU67	7.56	4.11	2.12,1.94	2.35,2.20 (γ)
SER68	8.24	5.76	3.70,3.57	
PHE69	8.65	4.93	1.80,2.51	6.97 (δ), 7.11 (ϵ), 7.27 (ζ)
THR70	8.69	5.97	3.96	1.05 (γ)
SER71	8.94	5.04	3.50,3.83	
THR72	8.36	4.56	3.68	0.89 (γ)
PHE73	8.10	4.24	2.78,2.42	6.32 (δ), 6.51 (ϵ), 7.10 (ζ)
THR74	8.17	3.98	3.82	1.08 (γ)
GLU75	8.11	5.07	1.88,2.04	2.30 (γ)
PRO76		4.30	2.15,2.35	1.94,1.92 (γ), 3.78,3.82 (δ)
GLY77	8.99	4.17,3.83		

Continua en la página siguiente

Continua de la página anterior

Residuo	H_N	H_α	H_β	Otros
THR78	8.02	5.00	3.89	1.10 (γ)
TYR79	9.69	4.97	3.42,3.32	7.08 (δ), 6.62 (ϵ)
THR80	9.31	5.03	4.20	1.23 (γ)
TYR81	8.68	5.15	3.08,2.96	6.64 (δ), 6.55 (ϵ)
TYR82	8.50	5.15	3.50,3.70	7.15 (δ), 6.94 (ϵ)
CYS83	7.60	5.22	3.42,2.95	
GLU84	9.90	4.28	2.32	2.55 (γ)
PRO85		5.12	1.50,1.64	1.10 (γ), 3.24 (δ)
HIS86	8.24	4.57	2.76	
ARG87	8.02	4.08	1.56,1.12	
GLY88	9.05	3.87,4.37		
ALA89	7.45	4.60	1.60	
GLY90	8.15	5.10,3.85		
MET91	7.19	4.51	2.15,2.55	1.53 (γ)
VAL92	7.76	4.95	2.10	0.84,0.94 (γ)
GLY93	8.48	5.05,4.25		
LYS94	8.67	5.48	1.78	1.38 (γ), 1.42 (δ), 2.82 (ϵ)
VAL95	9.32	4.57	1.80	0.98,-0.21 (γ)
VAL96	9.36	4.45	2.21	0.86 (γ)
VAL97	9.19	4.55	2.40	0.58,0.62 (γ)
GLU98	9.05	4.32	1.90,1.96	2.10,2.20 (γ)

Tabla 3.7: Tabla de asignación de resonancias para la plastocianina $Synechocistys$.

La asignación secuencial específica de los sistemas de espín identificados se llevó a cabo utilizando las conectividades NOE entre los protones H_α, H_β y H_N y las amidas de los residuos adyacentes ($H_{\alpha i} - H_{Ni+1}$,$H_{\beta i} - H_{Ni+1}$ y $H_{Ni} - H_{Ni+1}$) [62, 63]. Las conectividades secuenciales observadas están esquematizadas en la figura 3.13 y las asignaciones de resonancias obtenidas están tabuladas en la tabla 3.7. Se recogieron espectros NOESY a diferentes temperaturas para evitar problemas de solapamiento de resonancias. Otras conectividades secuenciales de interés como $H_{\gamma i} - H_{Ni+1}$ fueron utilizadas como soporte adicional a la asignación secuencial principal. Los residuos ASN34, HIS86 y ARG87 fueron asignados en etapas posteriores del proceso con la ayuda de las estructuras preliminares obtendidas por geometría de distancias.

3.4.2 Estructura secundaria

Se realizó un análisis de conectividades NOE interresiduo para la caracterización de los elementos de estructura secundaria presentes en la Plastocianina $Synechocistys$. De acuerdo a este análisis se identificaron diferentes segmentos de hoja β y un pequeño segmento de α-hélice. La Plastocianina $Synechocistys$ en disolución contiene ocho hebras β, un segmento de α-hélice, cinco giros inversos y dos bucles. Ocho segmentos polipeptídicos extendidos fueron identificados mediante el análisis de intensas conectividades NOE interhebra antiparalela $H_{\alpha i} - H_{\alpha j}$ y de conectividades NOE medias interhebra $H_{Ni-1} - H_{Nj+1}$, $H_{Ni+1} - H_{Nj-1}$, $H_{\alpha i} - H_{Nj+1}$ y $H_{Ni+1} - H_{\alpha j}$ donde i y j corresponden a aminoácidos en hebras opuestas. La presencia de dos hebras β de estructura secundaria paralelas se analiza mediante las conectividades NOE medias $H_{\alpha i-1} - H_{Nj-1}$ y $H_{Ni} - H_{\alpha j}$ y las conectividades NOE débiles $H_{Ni} - H_{Nj+1}$ y $H_{Ni} - H_{Nj-1}$. Por otra parte, fuertes conectividades dipolares $H_{Ni} - H_{Ni+1}$ y la existencia de picos cruzados $H_{\alpha i} - H_{Ni+3}$ en la región de los residuos ALA50 a ALA54 indican la presencia de un motivo α-helicoidal en esa zona. Los valores de -1 para el índice de desplazamiento químico α (ver figura 3.14) de los aminoácidos ASP49 a LYS59 parecen sugerir que este motivo α-helicoidal podría extenderse hasta el residuo HIS58, aunque esto no se ve confirmado en los mapas de correlación NOESY debido a los solapamientos en la región $H_\alpha - H_N$. Además, la no observación de picos cruzados DQF-COSY con valores de ${}^3J_{N\alpha}$ inferiores a 5.5 Hz parece apoyar la idea de que, aunque la posible tendencia de dicha región sea helicoidal como se deduce de algunas conectividades aisladas y de los valores de desplazamiento químico, la estructura real pueda corresponder a una α-hélice de elevada movilidad y limitada estabilidad.

Los datos NOESY que han permitido esta asignación de estructura secundara pueden verse esquematizados en la figura 3.15 para las hebras β. Los datos de estructura secundaria derivados de la información extraida de los experimentos de RMN indican que la plastocianina $Synechocistys$ contiene dos hojas β, numeradas I y II. La hoja β I, comprende cuatro hebras β con un total de 22 aminoácidos, incluyendo el segmento N-terminal ALA3 a GLY8. La segunda hoja β contiene cuatro hebras β e incluyen el segmento C-terminal MET91 a GLU98. Los elementos de estructura secundaria tale como hebras β paralelas y antiparalelas se ven confirmadas por la observación de protones amida de intercambio lento (ver figura 3.14) y por los desplazamientos químicos de los protones α.

Figura 3.14: Índice de desplazamiento químico para los protones α de la Plastocianina *Synechocistys* respecto a los residuos aislados. También se puede observar los protones amida intercambiables indicativos de puentes de hidrógeno. $-1, \Delta\delta \leq -0.1ppm$; $0, -0.1ppm \leq \Delta\delta \leq 0.1ppm$; $+1, -0.1ppm \leq \Delta\delta$.

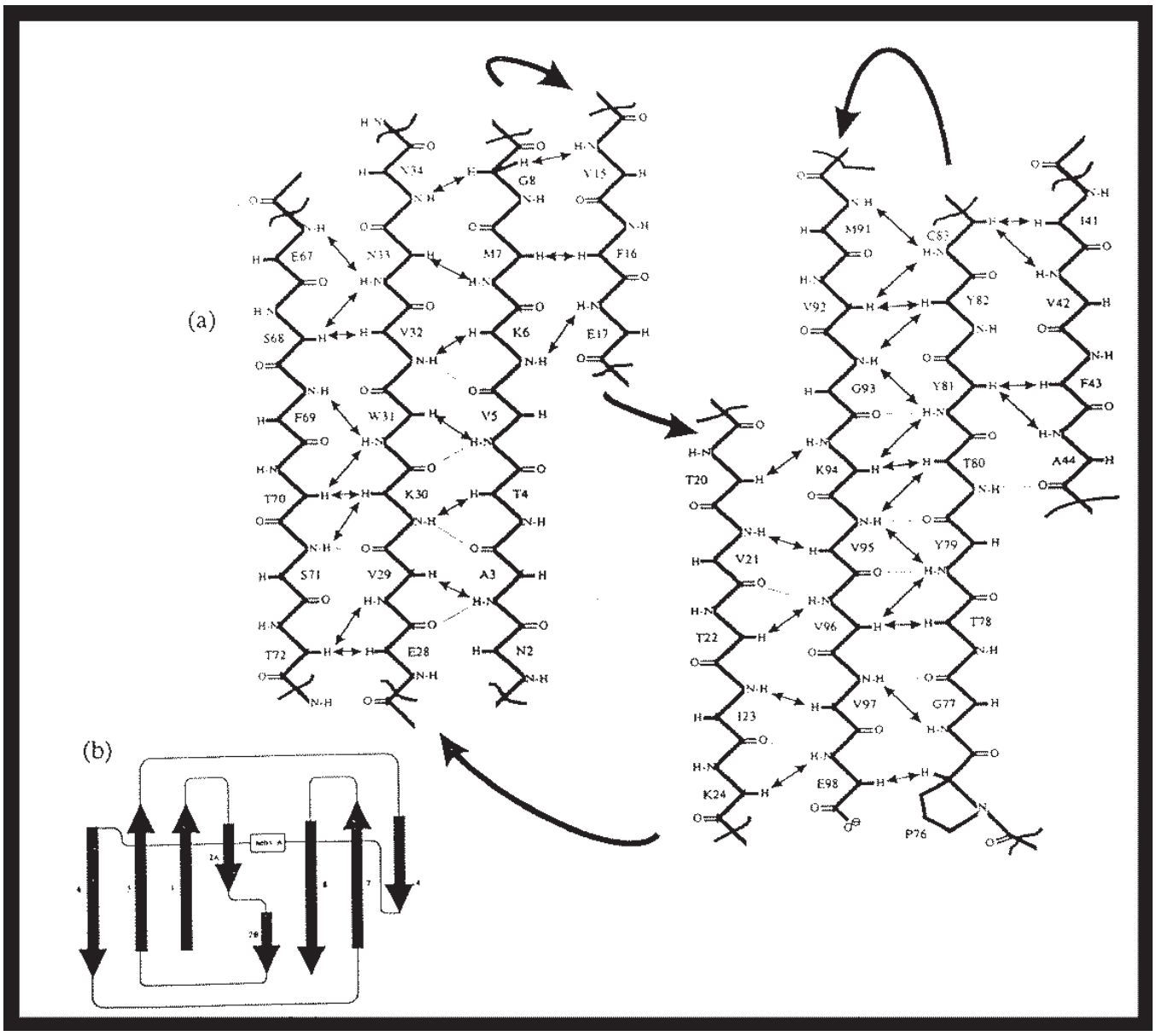


Figura 3.15: Esquema de conectividades NOE entre las hebras β de la plastocianina *Synechocistys* mostrando los puentes de hidrógeno correspondientes a los protones de intercambio lento determinados mediante comparación de los espectros en D_2O y H_2O.

3.4.3 Conformacíón de cadenas laterales

Se han obtenido valores de ángulo de torsión χ_1 y asignaciones estereoespecíficas para los protones metilénicos β mediante las constantes de acoplamiento ${}^3J_{\alpha\beta}$ y NOEs intraresiduo e interresiduo, implicando a los protones amida H_N, H_α y H_β (ver tabla 3.2). La relación relativa entre los valores de ${}^3J_{\alpha\beta 2}$ y ${}^3J_{\alpha\beta 3}$ se determinó como la relación entre intensidades de los picos cruzados $H_\alpha - H_{\beta 2}$ y $H_\alpha - H_{\beta 3}$, $I_{\alpha\beta 2}$ y $I_{\alpha\beta 3}$ respectivamente, en el espectro TOCSY a tiempo de mezcla corto como se puede apreciar en la figura 3.16 [64, 65] en la cual la transferencia de magnetización de H_α a $H_\beta 2$ y a $H_\beta 3$ es directamente proporcional a los valores de ${}^3J_{\alpha\beta 2}$ y ${}^3J_{\alpha\beta 3}$ respectivamente [66]. Mediante este análisis cualitativo se ha realizado una asignación estereoespecífica preliminar para 20 pares β metilénicos no degenerados en la plastocianina $Synechocistys$. Cuando los valores de ${}^3J_{\alpha\beta 2}/{}^3J_{\alpha\beta 3} \geq 2$ o ${}^3J_{\alpha\beta 2}/{}^3J_{\alpha\beta 3} \leq 0.5$ se selecciona una posición rotacional preferida para la cadena lateral entre $\chi_1 = +60(g^+)$, $\chi_1 = 180(t)$ y $\chi_1 = -60(g^-)$. La combinación de los valores de $I_{\alpha\beta 2}$ y $I_{\alpha\beta 3}$ así como los picos cruzados NOE intra e interresiduo a tiempo de mezcla corto (50 ms), permite la identificación de la proquiralidad de los protones β metilénicos. En los casos en que $0.5 \leq^3 J_{\alpha\beta 2}/{}^3J_{\alpha\beta 3} \leq 2$, se pueden esperar dos situaciones conformacionales diferentes. Cuando ambas constantes de acoplamiento ${}^3J_{\alpha\beta}$ tienen valores comprendidos entre 5 y 9 Hz, indicativos de un promedio de movimiento, se supone que existe una mezcla de poblaciones de rotámeros sobre el enlace $C_\alpha - C_\beta$ y ni la asignación estereoespecífica ni el valor del ángulo de torsión χ_1 puede ser determinado. Este es el caso, por ejemplo, de los protones β metilénicos del residuo ASP49. En los casos en que ambas constantes de acoplamiento ${}^3J_{\alpha\beta}$ tengan valores inferiores a 5 Hz, se supone un único rotámero con $\chi_1 = +60(g^+)$ que se confirma con las conectividades NOE correspondientes. Un ejemplo de rotámero de este tipo es el residuo SER68.

3.4.4 Calidad de las estructuras calculadas

Se calcularon un total de 50 estructuras finales de las que se ha escogido un subconjunto de 29 con valor de E_{NOE} inferior a 20 Kcal / mol. La superposición de las 29 estructuras escogidas se puede observar en la figura 3.18. No se obtuvieron violaciones de NOE superiores a 0.25 Å en ninguna de las 29 estructuras seleccionadas. En la tabla 3.8 se puede observar la poca dispersión existente entre las estructuras seleccionadas indicando un nivel

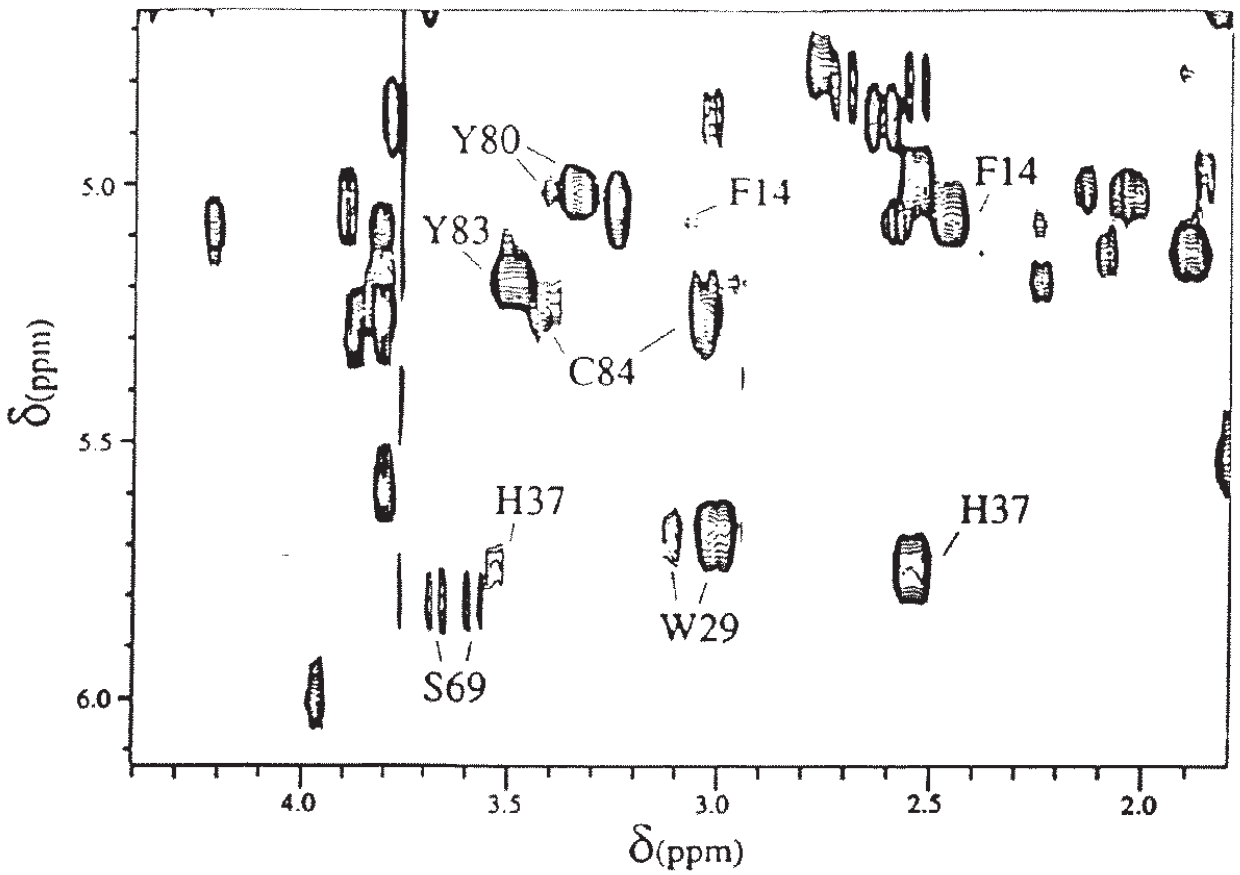


Figura 3.16: Región $H_\alpha - H_\beta$ del espectro TOCSY a 15 ms de tiempo de mezcla utilizada para la asignación estereoespecífica.

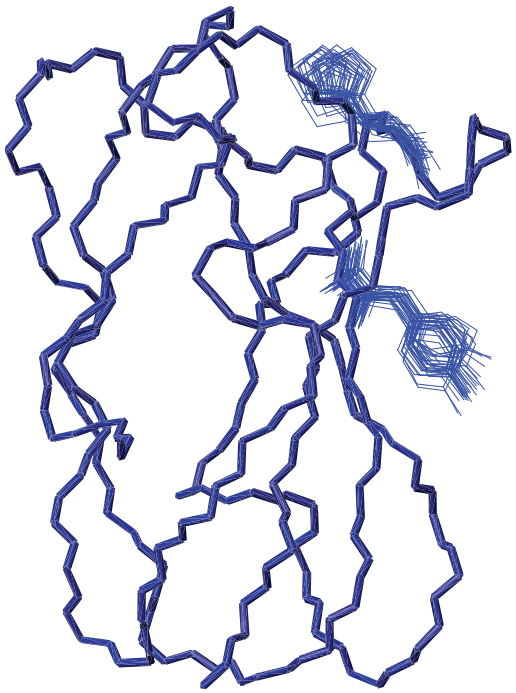


Figura 3.17: Cadena principal de la Plastocianina $Synechocystis$ en disolución mostrando las conformaciones de las cadenas laterales de los residuos implicados en la transferencia electrónica. Se puede observar la poca dispersión en sus conformaciones.

	Esqueleto carbonado	Átomos pesados
Regiones hoja β	0.63 ± 0.19	1.36 ± 0.34
Todos los residuos	0.97 ± 0.22	1.60 ± 0.30

Tabla 3.8: Desviaciones cuadráticas medias promedio en Å de las 29 estructuras seleccionadas respecto a la estructura promedio de la Plastocianina $Synechocystis$ en disolución.

de resolución de estructura bueno. En la figura 3.19 se puede apreciar que la distribución de restricciones de distancia a lo largo de la secuencia se corresponde en gran medida con los valores de desviación cuadrática media de las coordenadas atómicas en las estructuras obtenidas respecto al promedio. Las regiones con menos restricciones NOE presentan una mayor dispersión en las estructuras calculadas como se puede observar en las figuras 3.19 y 3.18. Dichas zonas coinciden con giros y bucles en la proteína, es decir, con las zonas de mayor flexibilidad y por ello menos definidas.

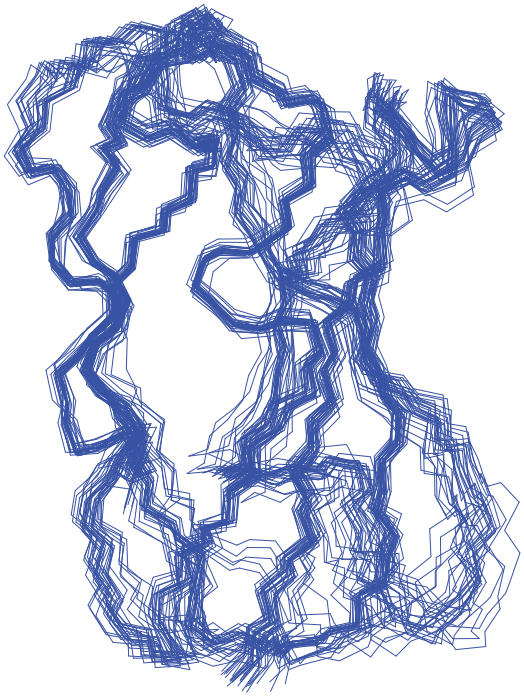


Figura 3.18: Superposición de la cadena principal de las 29 estructuras seleccionadas de la Plastocianina $Synechocystis$ en disolución. Se ha superpuesto solo los átomos de la cadena principal de las regiones en hoja β.

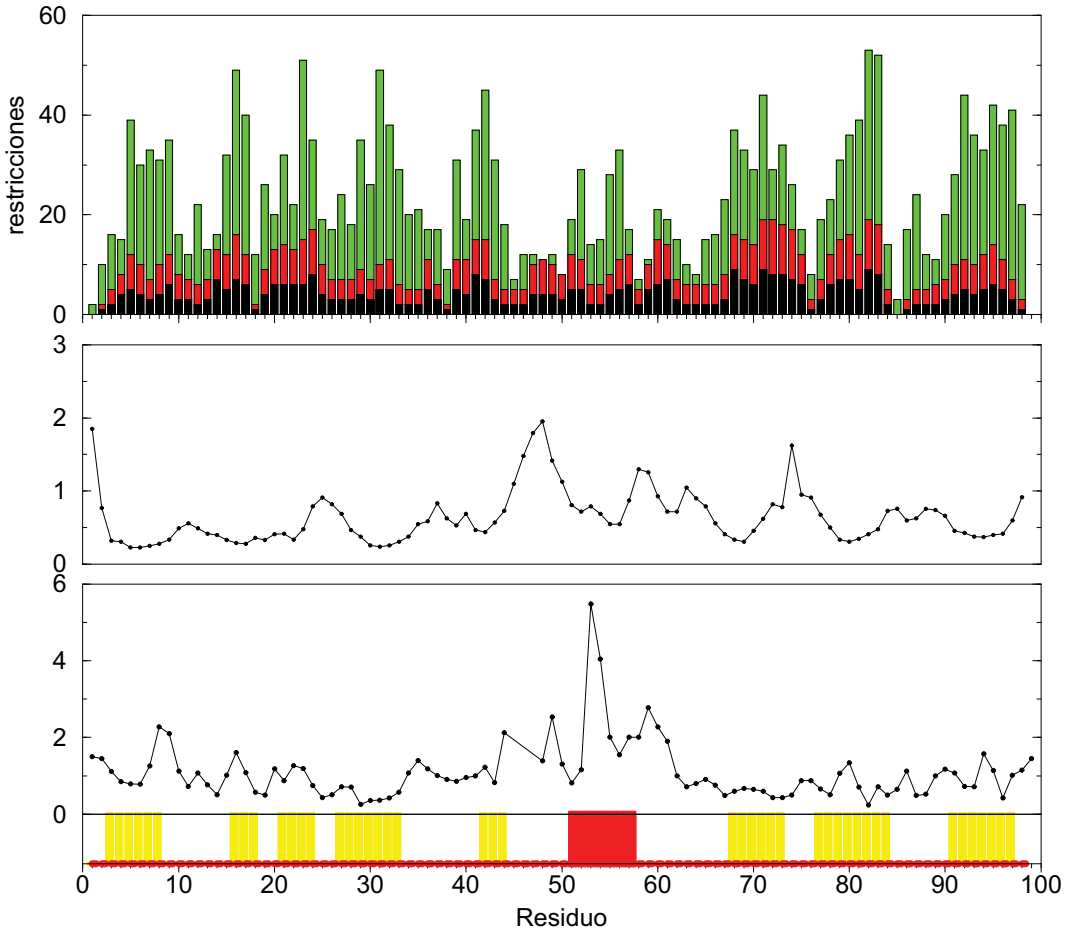


Figura 3.19: Representación frente a la secuencia de Plastocianina $Synechocistys$ del número de restricciones (arriba) intraresiduo (negro), secuenciales (rojo) y de larga distancia (verde), de la desviación cuadrática media de la familia de estructuras respecto al promedio para los átomos pesados (medio) y de la desviación cuadrática media de la estructura promedio de RMN frente a la estructura de Rayos X de $Poplar$ (abajo). Se muestra en color amarillo las regiones de hoja β y en rojo las de α hélice.

En la tabla 3.9 se pueden observar los valores de energía conformacional, de van der Waals y de NOE de las estructuras seleccionadas. Dichos valores indican una buena convergencia entre las estructuras calculadas y los datos experimentales ya que los valores de desviación de las restricciones NOE así como la energía de penalización por violaciones son bastante bajas. Los valores de energía de van der Waals indican que las estructuras tienen muy buenas interacciones de no enlace.

Un criterio muy utilizado para la evaluación de la calidad de una estructura determinada es la calidad estereoquímica [67]. El PROCHECK [68] es un programa muy utilizado y que ya ha sido presentado en el capítulo 2 para la evaluación de los modelos obtenidos por homología para la Plastocianina *Synechocystis* [126]. Las estructuras calculadas presentaron un 68 % global de residuos en las zonas más favorables del diagrama de Ramachandran como se puede observar en la figura 3.20(a). El número de residuos en zonas desfavorables es inferior a 1 por estructura indicando una calidad estereoquímica bastante buena. La resolución equivalente según diferentes criterios se puede observar en la figura 3.20(b) y se comprueba que la estructura obtenida es de una resolución media (2.5 -3.0 Å) [69, 70] superior a la estructura de la única plastocianina de cianobacteria resuelta con anterioridad [11] (50 % en zonas favorables y baja resolución).

Se realizó una comprobación adicional sobre la calidad de las estructuras obtenidas mediante el programa PDBSTAT [161] desarrollado en nuestro laboratorio para el análisis y cálculo de estadísticas de estructuras de proteínas. Se comprobó que la planaridad de los ángulos ω se mantiene dentro de los límites aceptables. Los residuos cuyos ángulos diedros ω se establecieron como cis en las restricciones de diedros fueron cis en todas las estructuras. Algunos residuos en las estructuras parecen situarse en zonas no favorables del diagrama de Ramachandran, tal y como indican los análisis realizados tanto con PDBSTAT como con PROCHECK. Sin embargo, la mayoría se sitúan en zonas favorables energéticamente y muy pocos pueden ser encontrados en zonas claramente prohibidas. Sin embargo, ésta situación es normal en estructuras con elevado porcentaje en hebras β como la de la plastocianina.

3.4.5 Puentes de Hidrógeno

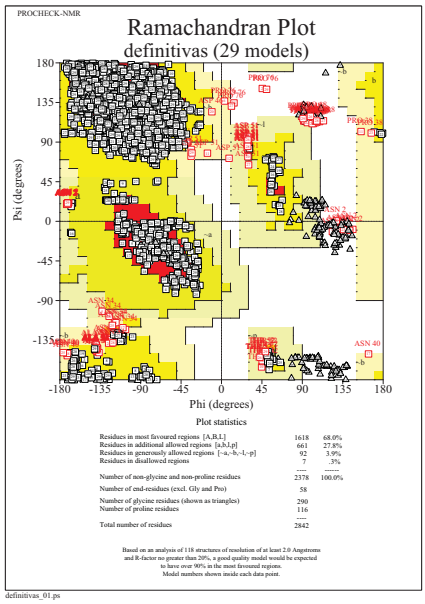
Las estructuras obtenidas para la Plastocianina *Synechocistys* en disolución se han caracterizado por la presencia de un número elevado de puen-

Energía Conformacional	Energía VdW	Energía NOE	Rmsd NOE (Å)
-1753.3	-555.6	10.6	0.0102
-1805.1	-550.6	4.1	0.0085
-1754.9	-557.4	10.6	0.0103
-1700.4	-539.2	17.7	0.0109
-1715.8	-536.2	13.9	0.0103
-1721.8	-533.3	5.0	0.0083
-1798.1	-527.1	18.8	0.0104
-1753.5	-557.8	11.1	0.0137
-1770.7	-551.5	4.6	0.0092
-1788.8	-552.6	8.6	0.0095
-1814.2	-563.1	18.7	0.0132
-1841.6	-535.2	17.0	0.0111
-1822.6	-546.9	8.3	0.0092
-1787.5	-545.6	6.2	0.0084
-1690.6	-552.3	6.5	0.0094
-1788.1	-547.8	5.9	0.0088
-1728.8	-545.9	10.3	0.0130
-1797.8	-547.8	7.7	0.0099
-1641.6	-534.4	8.4	0.0093
-1757.7	-576.4	9.4	0.0105
-1737.9	-552.5	13.5	0.0092
-1806.3	-550.9	17.3	0.0102
-1739.0	-554.2	19.8	0.0165
-1711.3	-541.4	17.1	0.0107
-1745.8	-542.2	13.2	0.0100
-1684.5	-525.9	10.2	0.0106
-1807.2	-529.0	7.6	0.0100
-1744.8	-556.1	6.6	0.0091
-1765.1	-550.4	10.1	0.0100

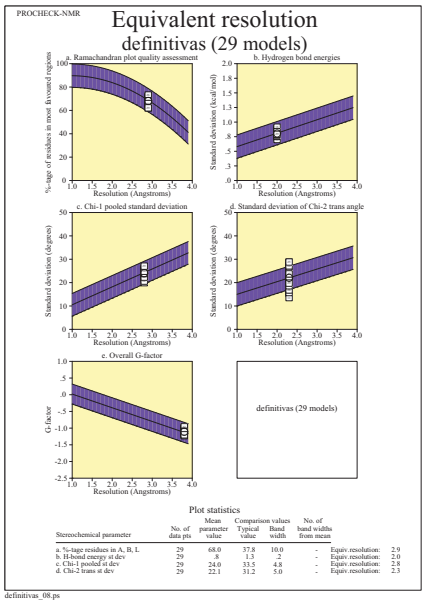
Tabla 3.9: Tabla de energías conformacional, de van der Waals y de NOE (en Kcal / mol) para las 29 estructuras seleccionadas de RMN medidas con el campo de fuerzas CHARMM.

	Media	Max.	Min.
E_{total}	-1757.75	-1841.59	-1641.61
E_{NOE}	10.99	4.07	19.79
E_{VdW}	-546.87	-576.42	-525.88
NOE (Å)	0.0103	0.0083	0.0165

Tabla 3.10: Tabla resumen de las estadísticas energéticas y estructurales de las estructuras seleccionadas.



(a) Diagrama de Ramachandran mostrando las zonas más favorables y el porcentage de residuos en ellas para las 29 estructuras seleccionadas de la Plastocianina *Synechocystis* en disolución



(b) Resolución equivalente según diferentes criterios de las 29 estructuras seleccionadas de la Plastocianina *Synechocystis* en disolución tal y como la calcula el programa PROCHECK

tes de hidrógeno. En los espectros de RMN se han observado 42 protones amida de intercambio lento de los cuales 30 se han identificado como protones correspondientes a los puentes de hidrógeno de las hojas β. Los 12 restantes se han asociado a puentes de hidrógeno en regiones de bucles o giros. En la tabla 3.6 se pueden observar todos los puentes de hidrógeno asociados a protones de intercambio lento. Todos estos puentes de hidrógeno han sido introducidos como restriccione en los cálculos y por lo tanto se han observado en todas las estructuras calculadas. El puente de hidrógeno SER9 $H_N \cdots OC$ ALA13 aparece en la mayoría de las otras plastocianinas como uno de los que no son de hoja β.

Se han encontrado en al menos 10 estructuras algunos puentes de hidrógeno que implican protones amida que no han sido clasificados como de intercambio lento. Estos puentes de hidrógeno son LYS30 $H_N \cdots OC$ ALA3, ASP51 $H_N \cdots OC$ ASP49, GLU67 $H_N \cdots OC$ ALA64, HIS86 $H_N \cdots OC$ CYS83 y LYS94 $H_N \cdots OC$ PRO18. De estos puentes de hidrógeno, el GLU67 $H_N \cdots OC$ ALA64 es uno de los más conservados en el resto de plastocianinas fuera de los de hoja β. El HIS86 $H_N \cdots OC$ CYS83 , por otra parte, además de ser uno de los más conservados en las plastocianinas y en las estructuras calculadas (aparece en 24 de las 29 obtenidas), es fundamental en la conformación tridimensional del lugar de coordinación del cobre.

Además de los puentes de hidrógeno mostrados en la tabla 3.6, una cantidad elevada de puentes de hidrógeno entre cadenas laterales ha podido ser determinado en un número variable de estructuras. Algunos de ellos se han repetido en más de la mitad de las estructuras y en la mayoría de ellos (6 de los 7 observados) el número de estructuras en que aparecen es superior a 17. Estos puentes de hidrógeno son ASN2 $H_{\delta 2} \cdots OC$ ASN2, SER9 $H_\gamma \cdots O_{\epsilon 2}$ GLU17, ASP10 $H_N \cdots O_{\epsilon 1}$ GLU17, ASN33 $H_{\delta 2} \cdots OC$ ALA64, SER68 $H_\gamma \cdots OC$ GLY66, THR70 $H_{\gamma 1} \cdots OC$ SER68 y SER71 $H_\gamma \cdots OC$ THR70. De ellos el ASN33 $H_{\delta 2} \cdots OC$ ALA64 es uno de los puentes de hidrógeno más conservados en todas las plastocianinas. Otros puentes de hidrógeno presentes en otras plastocianinas han sido observados en las estructuras calculadas aunque en menor medida, dando a entender que la diferencia estructural es pequeña. Estos son , por ejemplo, ASN33 $H_{\delta 2} \cdots OC$ ALA62 (7 estructuras), HIS39 $H_{\epsilon NH} \cdots OC$ ASN33 5 estructuras), TYR79 $H_{\eta OH} \cdots OC$ GLU75 (9 estructuras), SER57 $H_{\delta OH} \cdots OC$ ALA53 (6 estructuras calculadas) y THR80 $H_{\gamma 1} \cdots O_{\delta 1,2}$ ASP46 (8 estructuras).

3.4.6 Análisis de la estructura

La conformación global de la plastocianina $Synechocistys$ reducida, tal y como se puede ver en la figura 3.20, es muy parecida a la de otras plastocianinas cuya estructura ha podido ser determinada [55, 72, 73, 30, 11]. Así, la molécula se puede describir como una estructura en barril β compuesta por dos hojas β compuestas en total por ocho hebras β. Los pares de hebras están conectadas por giros y bucles , y las dos hojas β están separadas por un núcleo hidrófobico interno. La hoja β I contiene cuatro hebras β está formadas por los residuos de THR4 a MET7, de PHE16 a GLU17 de GLU27 a ASN33 y SER68 a PHE73. La hoja β II también consta de cuatro hebras β que están formadas por los residuos de de VAL21 a ILE23, de VAL42 a PHE43, de SER68 a PHE73, de GLY77 a TYR82 y de VAL92 a VAL97. El criterio de selección de los residuos que son hebra β es el habitual de que los valores de ϕ y ψ se encuentren en los intervalos de -180^o a -80^o y de 40^o a 180^o respectivamente.

En algunas plastocianinas es posible observar una región α helicoidal en la zona de los residuos 54 a 59 [126]. Sin embargo, en la Plastocianina *Synechocystis,* aunque se han observado algunas conectividades NOE que podrían indicar la presencia de una α hélice en dicha región, estas conectividades no presentan continuidad en todo el tramo. Por otra parte, la ausencia de un patrón de puentes de hidrógeno característico de α hélice tanto en las estructuras obtenidas como en los posibles protones de intercambio lento, apoyan la idea de que, aunque dicha región tenga cierta tendencia a tomar conformaciones próximas a la α hélice, como asi se aprecia en los valores de los ángulos diedros ϕ y ψ en los residuos implicados, la flexibilidad de ese tramo de la proteína haga que la presencia de una α hélice estable se vea cuanto menos dificultada.

La región de los residuos de GLU84 a GLY88 presenta también conformaciones próximas a la α hélice. En esta región, debido a que está muy próxima al lugar de coordinación del cobre y a que se encuentra rodeada por dos hebras β relativamente largas y muy bien definidas, la flexibilidad se supone más reducida. Sin embargo, el análisis de los ángulos diedros y la presencia de tan sólo uno de los puentes de hidrógeno que deberían observarse en caso de ser una α hélice, indica que esa región se puede describir mejor como dos giros β consecutivos. Esto se ha observado en las regiones equivalentes de otras plastocianinas como son la Plastocianina $Poplar$, la Plastocianina de $French\ Bean$ o la de $Parsley$ [30, 11, 10, 126].

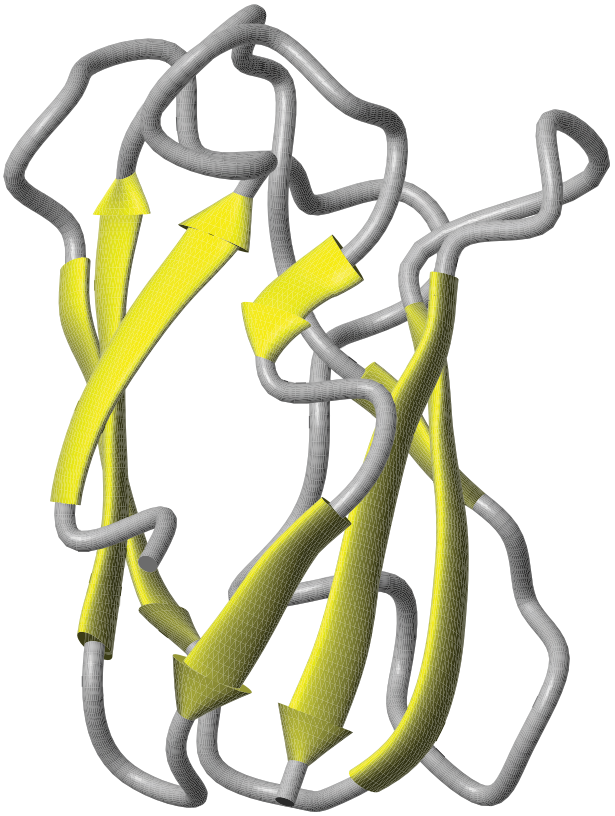


Figura 3.20: Representación de la estructura promedio de la Plastocianina $Synechocystis$ mostrando los elementos de estructura secundaria.

Figura 3.21: Representación estereográfica de la cadena principal de la Plastocianina *Synechocystis* en disolución.

3.4.7 Comparación con otras estructuras de Plastocianinas

Como se ha podido comprobar, la Plastocianina $Synechocystis$ en disolución presenta una estructura de plegamiento global muy similar a la de otras plastocianinas. En particular, la longitud y localización de los elementos de estructura secundaria como hojas β, la conformación de las prolinas conservadas, la orientación de las cadenas laterales más conservadas en la superficie y la red global de puentes de hidrógeno son casi idénticas a las de plastocianinas muy bien caracterizadas estructuralmente.

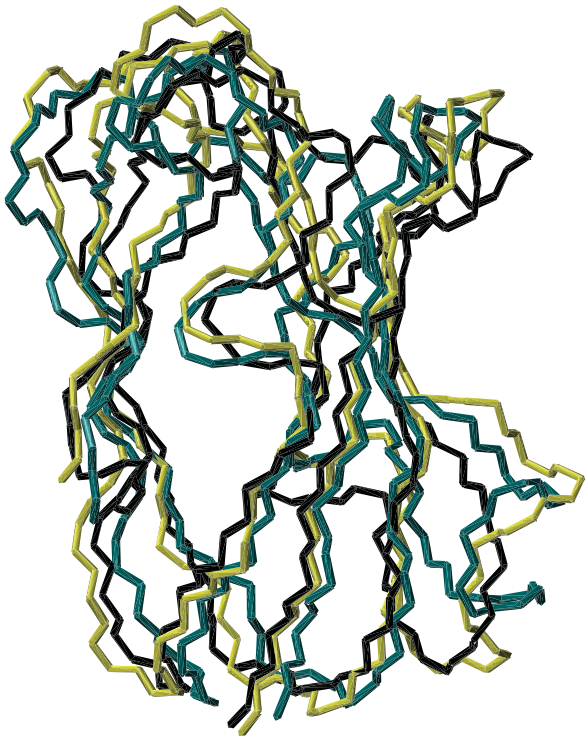


Figura 3.22: Superposición de las estructuras de Plastocianinas $Poplar$ (amarillo), $Parsley$ (verde), $Anabaena$ (rojo) y $Synechocystis$ (negro).

En concreto, una de las regiones más diferenciada entre una plastocianina muy bien caracterizada como es la Plastocianina $Poplar$ y la plastocianina $Synechocystis$ se centra en la región entre los residuos 54 a 58 como se puede apreciar en la figura 3.19. Esta región es la que presenta menos similitudes entre las plastocianinas caracterizadas siendo en algunas una región de α hélice y en otras una serie de giros consecutivos. En el caso de la plastocianina $Synechocystis$ se podría describir como un corto tramo de hélice 3_{10} de tan sólo tres residuos enlazada con dos giros. Esta

región situada en la cara "este" de la proteína. La diferencia entre otras plastocianinas y la plastocianina $Synechocystis$ en esta región va más allá de valores de desviación cuadrática media grandes. Mientras que el tramo 58 a 61 en las plastocianinas de plantas superiores consiste en un giro β, este motivo de estructura secundaria está ausente en la plastocianina $Synechocystis$ al igual que en la plastocianina $Anabaena$. Tanto la plastocianina $Synechocystis$ como la $Anabaena$, ambas procedentes de cianobacterias, se da la desaparición de este giro β presente en plantas superiores y ausente en algas verdes. Sin embargo, en algas verdes la desaparición está asociada a la ausencia de los residuos 57 y 58 respecto a las plastocianinas de plantas superiores, mientras que en las plastocianinas $Synechocystis$ y $Anabaena$ estos residuos están conservados respecto a plantas superiores. Sin embargo, mientras que en la plastocianina $Anabaena$ la desaparición del giro β conlleva un desplazamiento de la cadena polipeptídica pasando la SER58 de la $Anabaena$ a la posición equivalente a SER56 en $Poplar$, esta circunstancia no se da en la plastocianina $Synechocystis$. Esto puede ser debido a que el puente de hidrógeno estabilizador SER60 $H_{\delta OH} \cdots OC$ LYS57 es posible en la $Anabaena$ gracias a la mayor separación entre residuos en la secuencia, mientras en la $Synechocystis$ la separación es únicamente de dos residuos. Por ello, en la plastocianina $Synechocystis$ se observa un puente de hidrógeno más conservado en otras plastocianinas (SER57 $H_{\delta OH} \cdots OC$ ALA53 en 6 estructuras calculadas) en lugar del observado en la plastocinana $Anabaena$.

A pesar de las diferencias en la región "este" de la proteína, la conservación de una gran mayoría de rasgos estructurales respecto a otras plastocianinas es la tónica dominante en la estructura en disolución de la plastocianina $Synechocystis$. En el caso del esquema de puentes de hidrógenos hemos podido comprobar que no sólo se conserva el patrón de puentes de hidrógeno de las hojas β si no que además, un numeroso grupo de puentes de hidrógeno fuera de las hojas β e incluso en cadenas laterales se conserva respecto a otras plastocianinas. Lo mismo ocurre con las conformaciones de cadenas laterales determinadas mediante asignación estereoespecífica o medidas en las estructuras calculadas. Esta similaridad se puede observar tanto en las zonas protegidas de la región interna hidrofóbica a la que pertenecen las cadenas laterales de los residuos PHE16, TRP31, PHE73, TYR79 y CYS83, como las cadenas laterales expuestas al disolvente como son LYS59, SER68, GLU75 y TYR82, tal y como se puede apreciar en la tabla 3.3. Se puede comprobar que, como tónica general entre las plastocianinas tabuladas, la similaridad es mayor en los residuos del núcleo hidrófobico que en los

residuos expuestos al disolvente. Esto es así también para la plastocianina $Synechocystis$. Un residuo de particular importancia es la TYR82 que se encuentra en la vía de transferencia electrónica hacia el cobre. Esta cadena lateral parece ser diferente al resto de plastocianinas. En la estructura de RMN se observa como una cadena lateral g^- con valores de χ_1 entre -38^o y -76^o bastante bien definida, como se puede apreciar en la figura 3.17, y en el resto de plastocianinas aparece como g^+ con valores de χ_1 de 62^o, 61^o y 44^o para las plastocianinas $Synechocystis$ modelada por homología, Judía Verde y $Poplar$ respectivamente. Sin embargo, en la estructura de la plastocianina $Parsley$ en disolución determinada por RMN la conformación de esta cadena lateral es g^-, como en la plastocianina $Synechocystis$ y la relación entre las constantes de acoplamiento $^3J_{\alpha\beta}$ es bastante similar a la obtenida para la plastocianina $Synechocystis$ (ver tabla 3.2). En general, los residuos cuya conformación de cadena lateral se ha determinado mediante asignación estreoespecífica son bastante similares a al menos una de las plastocianinas ya caracterizadas.

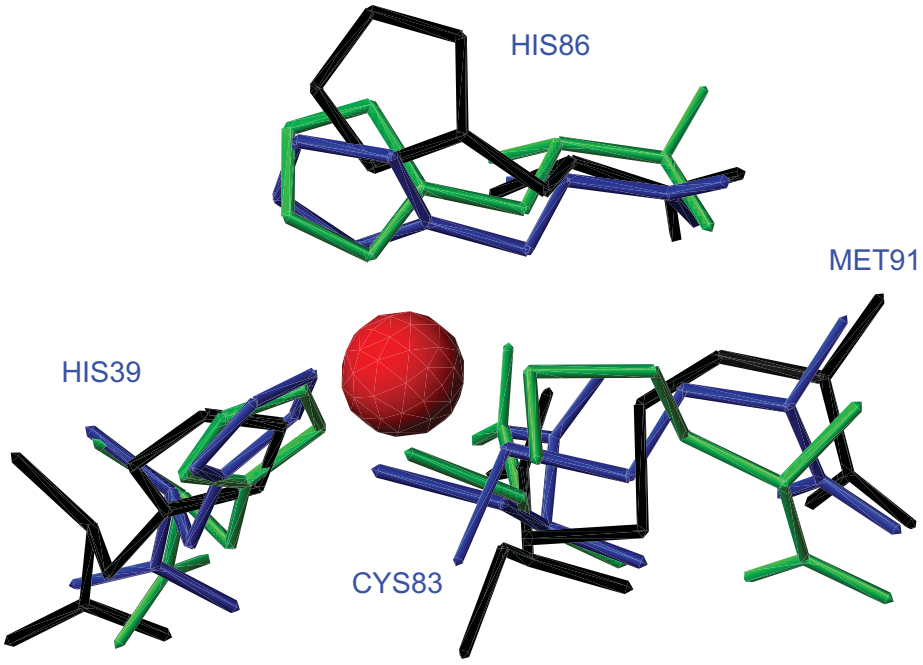


Figura 3.23: Ampliación del lugar de coordinación del cobre y superposición con la estructura de homología de la plastocianina $Synechocystis$ (azul), $Anabaena$ (verde) y $Synechocystis$ en disolución mediante RMN (negro).

El sitio de coordinación del cobre es otro de los lugares mejor conservados en las plastocianas. Como se puede apreciar en la figura 3.23, las dife-

rencias en la estructura de dicha región entre la plastocianina $Anabaena$ y la plastocianina $Synechocystis$ es muy pequeña. La coordinación mantenida es tetraédrica como corresponde al pH 7.0 utilizado en los experimentos realizados. La PHE16 está en contacto de van der Waals con la MET91, como se puede comprobar por la corta distancia entre algunos de sus átomos (N de PHE16 a S_δ de MET91 menos de 3 Å), lo que puede estabilizar la posición del residuo en su coordinación al cobre. Los mismo ocurre entre la HIS86 y la LEU14 (menos de 2.5 Å entre los protones β) y entre la HIS39 y la ASN33 (menos de 3 Å entre el O δ de la ASN33 y el N δ_1 de la HIS39) siendo todos ellos residuos fuertemente conservados en todas las plastocianinas. Esta posible estabilización adicional del sitio de coordinación del cobre a través de las interacciones de van der Waals con residuos conservados puede ser un motivo adicional a la conservación de dichos residuos en la secuencia general de las plastocianinas. Por otra parte, el protón δ_1 de la HIS58 presenta un puente de hidrógeno con los O ϵ_1 y ϵ_2 en 8 de las estructuras calculadas pudiendo ser motivo de estabilización adicional de la posición de la CYS83. Se han observado patrones similares de estabilización en la estructura cristalina de la plastocianina $Poplar$ para varios residuos más.

La plastocianina $Annabaena$ es la más parecida desde el punto de vista evolutivo a la plastocianina $Synechocystis$. Es por ello interesante realizar una comparación individualizada entre ellos y comprobar que ambas poseen estructuras bastante similares. En la figura 3.24 se puede observar la superposición de la familia de estructuras seleccionadas para la plastocianina $Synechocystis$ y la familia de estructuras depositada en el PDB para la plastocianina $Anabaena$. A pesar de que la dispersión en el caso de la $Anabaena$ es mayor y que el tamaño de proteína es tambien bastante superior a la de la $Synechocystis$, ambas estructuras son muy similares. Las mayores diferencias se sitúan en la zona de los bucles donde, la plastocianina $Anabaena$ tiene cuatro residuos adicionales sobre el resto de plastocianinas.

Por otra parte, la comparación de la estructura obtenida mediante RMN con la propuesta en el capítulo 2 obtenida mediante modelización por homología es necesaria para comprobar la calidad de las predicciones realizadas partiendo de un nivel de homología funcional elevado con el resto de plastocianinas, así como para analizar las posibles limitaciones del método. Se ha comparado la estructura de mínima energía de las obtenidas por RMN y la de mínima energía modelizada por homología para su comparación. Como se puede apreciar en la figura 3.25, la similitud entre ambas estructuras a nivel de cadena principal es muy elevada. Asi-

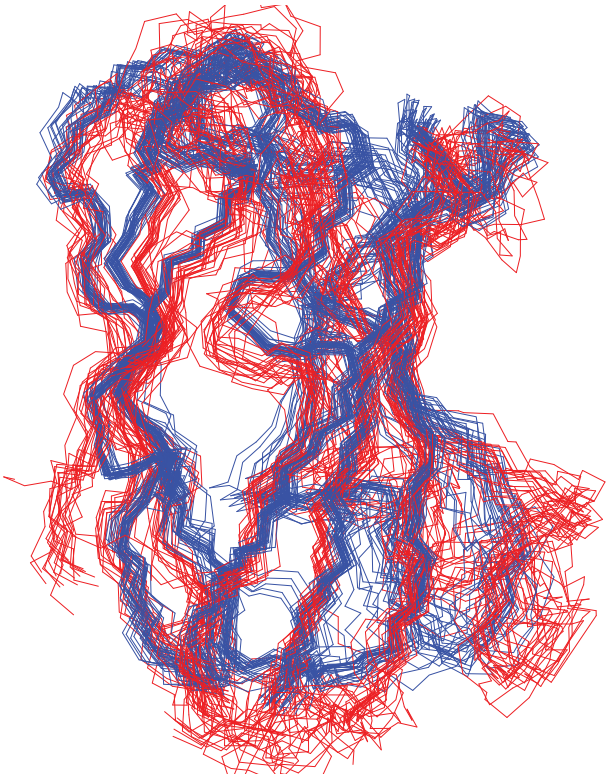


Figura 3.24: Superposición de las cadenas principales de las familias de estructuras determinadas por RMN para la plastocianina *Synechocystis* (azul) y la Plastocianina *Anabaena* (rojo) ambas procedentes de cianobacterias.

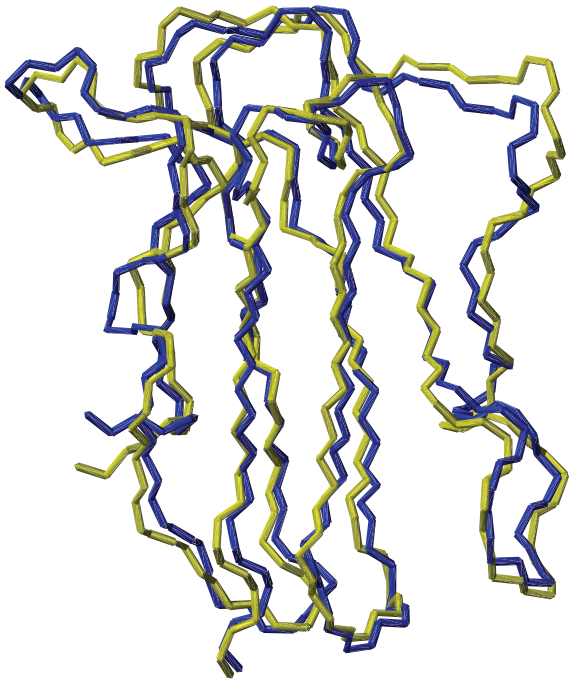


Figura 3.25: Superposición de la cadena principal de la estructura de mínima energía obtenida mediante modelización por homología (amarillo) y mediante RMN (azul) para la plastocianina *Synechocystis*.

mismo, los valores de desviación cuadrática media para los átomos de la cadena principal (2.49 Å) entre la estructura de mínima energía de RMN y la estructura de mínima energía obtenida mediante modelización por homología son lo suficientemente bajos como para considerar la modelización por homología una técnica de gran utilidad en la predicción de estructuras tridimensionales.

3.4.8 Análisis de potencial electrostático

Una vez obtenida la estructura tridimensional de la plastocianina *Synechocystis* en disolución y habiendo comprobado que, efectivamente las diferencias estructurales no son capaces de justificar una diferencia en el mecanismo cinético como la observada experimentalmente, se realizó el análisis de potencial electrostático sobre la estructura en disolución y se comparo el resultado con otras plastocianinas. Como ya se comentó en el capítulo 2, la principal diferencia electrostática entre la plastocianina $Synechocystis$ y las plastocianinas de plantas superiores reside en la región que rodea la TYR82. Estos residuos son eminentemente ácidos en las plastocianinas de plantas superiores mientras que en el caso de plastocianinas de cianobacterias como la plastocianina $Synechocystis$ o la plastocianina $Anabaena$ son de un marcado carácter básico o neutro. En la figura 3.26 se pueden observar las cadenas que acompañan a la TYR82 y su carácter positivo, tanto por el tipo de residuo que son como por el fuerte potencial positivo (azul) que se ve en la distribución de potencial en la superficie de van der Waals.

La distribución de potencial electrostático sobre la superficie de van der Waals se ha calculado del modo descrito en el capitulo 2. Se puede observar que el carácter positivo de la superficie obtenida es más fuerte que el que presentaba la esructura de homología. Esto es debido a la orientación de las cadenas de los residuos ASP49 y ASP51 que compensaban parcialmente la ausencia de los residuos ácidos en las posiciones 44 y 45. En la estructura de RMN, dicha región no está próxima al pequeño segmento en conformación de α hélice observada en las estructuras obtenidas por homología lo que da mayor flexibilidad a dicho tramo y hace que las cadenas laterales largas con cargas se orienten de modo diferente. Como se puede observar en la figura 3.26, el carácter ácido presente en las plantas superiores para esta región se ha perdido casi completamente en favor de una zona mucho más positiva.

La región que comprende los residuos del 59 al 61 es una zona típicamente negativa en las plantas superiores y que aquí se ha convertido en

Figura 3.26: Potencial electrostático en la superficie de van der Waals de la Plastocianina *Synechocystis* en disolución, tal como la representa el modulo GRASP del programa MOLMOL. Rojo: negativo, Azul: positivo, Blanco: Neutro.

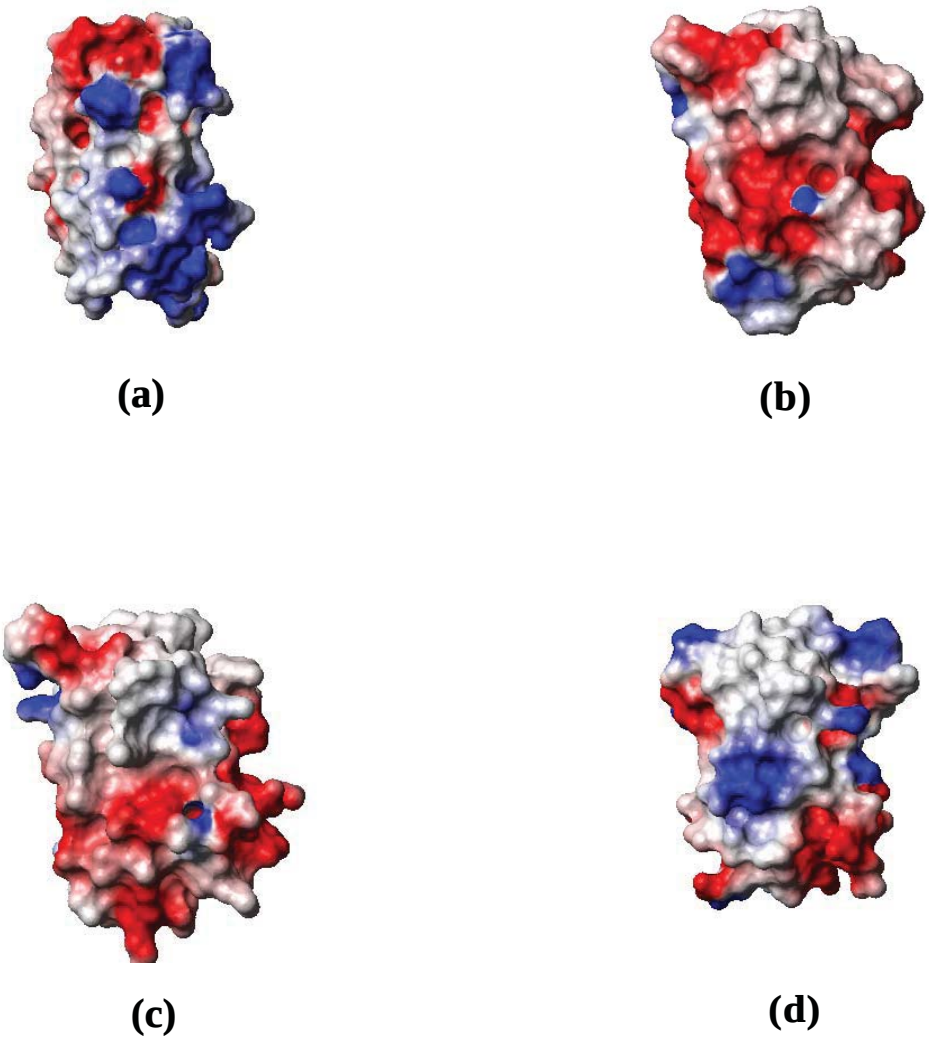


Figura 3.27: Comparación entre los potenciales electrostáticos en la superficie de van der Waals para las plastocianinas de a) $Anabaena$, b) $Poplar$, c) $Parsley$ y d) $Synechocystis$.

positiva, como ya se había observado en las estructuras obtenidas por homología. Esta región, que está situada encima de la TYR82, responsable de la transferencia electrónica hacia el cobre, está constituida por los residuos LYS55, LYS59 y ARG87. Estos residuos se repliegan más sobre la superficie de la proteína en la estructura determinada por RMN que en la de homología permitiendo la neutralización total de la pequeña región negativa correspondiente al GLU84 y produciendo un incremento sustancial de la carga positiva en dicha zona. Así, las orientaciones de las cadenas laterales de la plastocianina *Synechocystis* en disolución hacen que la superficie de potencial electrostático presente mayores diferencias con las plastocianinas de plantas superiores y, por el contrario, mayores similitudes con la plastocianina *Anabaena*, como puede observarse en la figura 3.27 que es otra plastocianina de cianobacteria.

3.5 Conclusiones

Se ha obtenido la estructura tridimensional de la Plastocianina *Synechocystis* en disolución mediante la aplicación de restricciones de distancias derivadas de experimentos de Resonancia Magnética Nuclear bidimensionales. La estructura obtenida tiene un plegamiento global similar al del resto de plastocianinas bien caracterizadas. Incluso una gran parte de las cadenas laterales de los residuos conservados mantienen su conformación respecto a la de plastocianinas homólogas, así como la conformación de las prolinas conservadas y el esquema general de puentes de hidrógeno.

No se han encontrado diferencias estructurales significativas entre la estructura de la plastocianina *Synechocystis* y las plastocianinas superiores que justifiquen un comportamiento cinético diferente y un mecanismo de reacción distinto. Las diferencias estructurales con el resto de plastocianinas se sitúan en la región de los residuos 58 a 65. Sin embargo, está región es la menos conservada y entre las propias plantas superiores se detectan las mayores diferencias en dicho segmento.

Se han encontrado similaridades entre la estructura de la plastocianina *Synechocystis* y la de la única plastocianina de cianobacteria resuelta hasta ahora. Algunos puentes de hidrógeno estabilizadores observados en la plastocianina *Anabaena* están presentes en la plastocianina *Synechocystis* así como la orientación global del bucle de los residuos 60 a 65 como se aprecia en las figuras de superposición.

Las diferencias fundamentales entre las plastocianinas de cianobacteria y las de plantas superiores se encuentra en la distribución de potencial electrostático, siendo ésta bastante similar entre la plastocianina $Anabaena$ y la plastocianina $Synechocystis$ y bastante diferente a la de plantas superiores. La predominancia de zonas de potencial negativo en las plastocianinas superiores favorece la formación de un complejo estable con el citocromo f, mientras que el fuerte carácter positivo de las plastocianinas $Anabaena$ y $Synechocystis$ dificulta la formación del complejo de transición provocando la necesidad de un mecanismo simple de colisión para la transferencia electrónica.

Capítulo 4

Estudio de la dinámica de una proteína mediante RMN. Análisis de la relajación.

4.1 Introducción.

La mayoría de los esfuerzos dirigidos a buscar una explicación de determinada función protéica o determinado mecanismo de acción se han basado en una visión estática de las estructuras de las proteínas. El modelo *llave-cerradura* se ha utilizado para explicar la especificidad mostrada por una enzima en su unión a un sustrato particular y la eficacia en las reacciones bioquímicas derivadas de la misma. Sin embargo, este modelo presupone una rigidez estructural que dista mucho de ser cierta. En aquellos casos en que la morfología de la proteína cambiaba en el transcurso de la reacción, se solía asociar este fenómeno a transiciones bruscas entre dos estructuras estáticas. Evidentemente, el término entrópico debe jugar y juega un papel fundamental en la variación de energía libre implicada en los procesos de reconocimiento molecular.

4.1.1 Introducción a Dinámica de Proteínas

En los últimos años se ha aceptado que los átomos de las moléculas están en constante movimiento. Así pues, la estructura observada mediante técnicas de cristalografía de Rayos X, por ejemplo, constituye una estructura promedio de la proteína [74]. Dicha estructura resulta de especial interés, ya que

suele ser el punto de partida de cualquier intento de elaborar descripciones más ajustadas de la actividad de la proteína. Sin embargo, hemos de tener presente las limitaciones que una estructura rígida conlleva y por lo tanto, en la medida de lo posible, es interesante y en bastantes ocasiones crítico conocer las propiedades dinámicas. La función de una proteína está intrínsecamente unida a los movimientos a nivel atómico [76]. Por ejemplo, el proceso de reconocimiento que implica las interacciones entre un inhibidor y su receptor, un antígeno y su anticuerpo, o una enzima y su substrato necesita la flexibilidad de la proteína para permitir a ambos un reconocimiento eficaz, rápido y selectivo y para adaptarse a su molécula complejante. Es necesario que cada uno de los componentes del proceso sea capaz de adoptar la conformación requerida para llevar a cabo su función. Incluso reacciones tan simples como el enlace de oxígeno o monóxido de carbono a las mioglobina implica un número elevado de subestados conformacionales [75].

Los cambios conformacionales que pueden tener lugar son de una gran variedad. Pueden ocurrir desde pequeños cambios conformacionales en cadenas laterales localizadas hasta movimientos relativos de unas hélices respecto a otras, pasando por movimientos de balanceo de ramas de la molécula o segmentos diferenciados. Al igual que ocurre en la mayoría de polímeros, la cadena polipeptídica que constituye la proteína es flexible por la cantidad de grados de libertad asociados a cada uno de los enlaces covalentes con posibilidad de rotación. La libertad de rotación de un enlace determinado permite que una porción de molécula se mueva respecto a otra. Cuando esto ocurre, al giro de la cadena principal acompaña el movimiento de las cadenas laterales, que a su vez están unidas por enlaces que permiten la rotación, lo que da lugar a una gran flexibilidad global. La flexibilidad de la cadena principal de una proteína y de sus cadenas laterales es lo que le permite plegarse en su estructura nativa [77]. Sin embargo, aun encontrándose plegada la proteína, la energía térmica correspondiente a las velocidades atómicas a temperatura ambiente es suficiente para permitir la rotación en torno a algunos enlaces y producir una cantidad considerable de fluctuaciones atómicas. A pesar de que en ocasiones, la proximidad de unos átomos respecto a otros restrinja en gran medida las fluctuaciones atómicas posibles, la suma de pequeños movimientos locales puede producir desplazamientos coordinados a gran escala de porciones de la proteína unas respecto a otras.

Según la escala de tiempo analizada, la dinámica de una proteína presenta unas características u otras. En intervalos de tiempo cortos, del or-

den de décimas de nanosegundos, las fluctuaciones de mayor contribución son los movimientos locales, como puedan ser las rotaciones de los grupos peptídicos y las cadenas laterales de unos 20 a 60 grados alrededor de los enlaces que unen el grupo con la cadena polipeptídica. Este movimiento corresponde a la variación del ángulo diedro llamado χ_1 y suelen presentar muy poca coherencia. Los grupos en movimiento colisionan con mucha frecuencia contra los grupos vecinos y el desplazamiento se ve interrumpido y el movimiento del grupo en el entorno de la proteína se asemeja al de una molécula en el seno de un líquido. Las colisiones impiden un movimiento continuado y se produce una trayectoria errática que corresponde al movimiento browniano de difusión [78]..

En intervalos de tiempo de duración superior, las principales características del movimiento corresponden más a las de los sólidos. Las fuerzas que mantienen a la proteína plegada impiden que los átomos se alejen excesivamente de sus posiciones promedio con lo que las fluctuaciones se asemejan más a las oscilaciones en torno a una posición de equilibrio propias de los átomos en un sólido [79]. Sin embargo, los movimientos propios de un sólido suelen ser de carácter más general que las oscilaciones locales. Es obvio que un átomo en particular no puede alejarse de su posición de equilibrio a menos que lo hagan también sus vecinos, ya que en caso contrario las colisiones con ellos se lo impedirían. Por ello, los grandes movimientos de porciones considerables de las proteínas (movimientos de dominios, por ejemplo) se componen en realidad de una gran cantidad de movimientos coordinados a nivel atómico. De este modo, el movimiento mantiene su similaridad con el movimiento de un sólido debido a la coordinación de las fluctuaciones de los átomos.

Se puede deducir pues, que aunque la estructura de una molécula es de gran interés para la comprensión de su actividad química, el estudio de su dinámica, primero a nivel atómico y luego a un nivel superior, es imprescindible para el completo conocimiento del sistema. De hecho, la actividad química en proteínas se encuentra en la mayoría de ocasiones localizada en la superficie de la molécula, es decir, en la parte de mayor movilidad de la misma [90].

La RMN es una técnica de gran aplicación en el estudio de procesos dinámicos en intervalos de tiempo tanto superiores a segundos como inferiores a nanosegundos. El margen de aplicación de la misma, por lo tanto, es el más amplio de todas las técnicas conocidas. Los movimientos más rápidos, como puedan ser las fluctuaciones atómicas en torno a su posición

de equilibrio, se pueden estudiar mediante el análisis de los parámetros de relajación [80].

Aunque en el estudio de la estructura de proteínas, la mayoría de información obtenida por RMN está basada en el protón, ^1H, los heteronúcleos, como son el ^{15}N y ^{13}C, aportan una información valiosísima en el estudio de los fenómenos de relajación y por lo tanto de la dinámica del sistema en estudio [82, 83].

4.1.2 Aplicación de la Relajación en RMN a dinámica de proteínas

Introducción a Relajación en RMN

Los tratamientos teóricos de la espectroscopía óptica ponen un especial enfásis en el papel que juega la emisión espontánea producto de la relajación desde el estado excitado al estado fundamental de una molécula. La probabilidad por unidad de tiempo, W, para las transiciones de un estado superior a otro inferior de energía en un dipolo magnético aislado mediante emisión espontánea de un fotón de energía $\Delta E = \hbar\omega$ viene dada por:

$$W = \frac{16\pi^3\gamma^2\hbar}{3\lambda^3} \tag{4.1}$$

Para un protón con una frecuencia de Larmor de 500 MHz, W es aproximadamente igual a $10^{-21}s^{-1}$. Así, la emisión espontánea es un mecanismo de relajación completamente despreciable en la Resonancia Magnética Nuclear. La emisión de fotones tanto espontánea como estimulada es importante en la espectroscopía óptica debido a las frecuencias de trabajo de las mismas, pero en la espectroscopía de Resonancia Magnética Nuclear las frecuencias son varios órdenes de magnitud inferiores y la contribución de la emisión es completamente despreciable.

La relajación del espín nuclear es consecuencia del acoplamiento de los sistemas de espín con sus entorno. Los niveles de energía del entorno se pueden suponer como un quasicontínuo con poblaciones descritas mediante una distribución de Boltzmann. Además, se puede suponer que el entorno se encuentra en equilibrio térmico siempre. El entorno modifica el campo magnético local en las posiciones del núcleo y de ese modo acopla débilmente con el sistema de espín. Los movimientos brownianos rotacionales de las moléculas en soluciones líquidas produce campos magnéticos

locales dependientes del tiempo que pueden ser descompuestos mediante análisis de Fourier en una superposición de campos magnéticos armónicos variables con diferentes frecuencias. Además, estos campos magnéticos pueden ser descompuestos en una componente paralela y otra perpendicular al campo magnético principal.

Las componentes transversales de los campos magnéticos estocásticos son responsables de las contribuciones no adiabáticas a la relajación. Si el espectro de Fourier de los campos magnéticos fluctuantes tranversales en la posición de un determinado núcleo contiene componentes de frecuencias correspondientes a diferencias de energía entre estados propios del sistema de espín, la transición entre estados propios puede ocurrir. En ese caso, la transición del sistema de espín de un estado de energía a otro viene acompañado de una transición en el entorno en sentido inverso para conservación de la energía. Debido a que en el entorno los niveles más poblados en el equilibrio térmico son los más bajos en energía, la transición más probable es de un nivel inferior a uno superior por lo que en el sistema de espín la más probable es la inversa. Así, el intercambio de energía entre el entorno y el sistema de espín lleva a este último al equilibrio térmico con el entorno y a las poblaciones del estado estacionario a la distribución de Boltzmann. Además, las transiciones entre estados estacionarios producidas por procesos no adiabáticos reducen los tiempos de vida de estos estados e introducen incertidumbre en las energías de los estados de espín nuclear a través del principio de incertidumbre de Heisenberg. El resultado de todo esto es una variación aleatoria de las frecuencias de Larmor de los espines y una pérdida de la coherencia de fase entre espines con el tiempo. Se puede concluir pues, que las fluctuaciones no adiabáticas entre estados resulta en un equilibrio térmico de las poblaciones de los estados de espín y un decaimiento de las coherencias fuera de la diagonal.

Los campos fluctuantes paralelos al campo magnético principal son responsables de las contribuciones adiabáticas a la relajación. Estos campos fluctuantes generan variaciones en el campo magnético total en la dirección z y consecuentemente en las energías en los estados de espín nuclear. Por lo tanto, los procesos adiabáticos producen una variación aleatoria de las frecuencias de Larmor de los espines. Con el tiempo, además, los espines pierden gradualmente la coherencia de fase y los elementos fuera de la diagonal de la matriz de densidad decaen a cero. Las poblaciones de los estados no se ven alteradas y no hay intercambio de energía entre el sistema de espín y el entorno porque las transiciones entre estados estacionarios no tienen lugar.

En la aproximación mas simple a la relajación en Resonancia Magnética Nuclear, la relajación en sistemas de espín aislados se caracteriza mediante las ecuaciones de Bloch en dos constantes de relajación de primer orden: la constante de relajación espín-entorno o longitudinal R_1, y la constante de relajación espín-espín o transversal, R_2. La relajación espín-entorno describe la recuperación de la magnetización longitudinal hacia el equilibrio térmico o, en su equivalente, el regreso de las poblaciones de los estados de energía del sistema de espín (elementos diagonales del operador de densidad) a la distribución de Boltzmann del equilibrio. La constante de relajación espín-espín describe el decaimiento de la magnetización transversal a cero, o lo que es lo mismo, el decaimiento de las coherencias de simple-cuanto transversales (elementos fuera de la diagonal de la matriz de densidad). Los procesos no adiabáticos contribuyen a ambas constantes mientras que los adiabáticos sólo contribuyen a la relajación espín-espín debido a que no producen cambio en las poblaciones de los estados estacionarios.

Modelo Lipari-Szabo

Un número muy elevado de interacciones físicas dan lugar a hamiltonianos estocásticos capaces de influir en la relajación de espín. En los estudios realizados en este trabajo sólo se han tenido en cuenta las interacciones magnética dipolar intramolecular, desplazamiento químico anisotrópico (CSA), acoplamientos escalar e intercambio químico. Para núcleos de espín $1/2$ en moléculas diamagnéticas los mecanismos de mayor contribución son el magnético dipolar y CSA. El acoplamiento cuadrupolar sólo es importante para núcleos de espín superior a $1/2$ [84].

Las constantes de relajación para los núcleos en proteínas depende de un número de factores muy elevado, incluyendo tiempos de correlación rotacional global, movimientos internos, disposiciones geométricas de los núcleos y las fuerzas relativas de los mecanismos de relajación aplicables. Así pues, los datos obtenidos mediante la medida de las magnitudes influídas por los procesos de relajación contienen información sobre la naturaleza de los movimientos internos que ocurren en los sistemas medidos [85]. Las determinaciones de las velocidades de relajación de ^{15}N o ^{13}C es particularmente útil para la obtención de información dinámica ya que la relajación de estos núcleos de espín $1/2$ está dominada por la interacción dipolar con los protones a los que están enlazados directamente mucho más que por el CSA.

En el caso de medidas de relajación en proteínas marcadas con ^{15}N, por ejemplo, las expresiones de las velocidades de relajación y del incremento de efecto NOE heteronuclear producido por la relajación serían las siguientes [86, 87]:

$$R_1 = \frac{d^2}{4}\left[J(\omega_H - \omega_N) + 3J(\omega_N) + 6J(\omega_H + \omega_N)\right] + \frac{\Delta^2\omega_N^2}{3}J(\omega_N) \quad (4.2)$$

$$\begin{aligned} R_2 = \ & \tfrac{d^2}{8}\left[4J(0) + J(\omega_H - \omega_N) + 3J(\omega_N) + 6J(\omega_H) + 6J(\omega_H + +\omega_N)\right] + \\ & + \tfrac{\Delta^2\omega_N^2}{18}\left[4J(0) + 3J(\omega_N)\right] + R_{ex} \end{aligned} \quad (4.3)$$

$$HNOE = 1 + \frac{\gamma_H\left[6J(\omega_H + \omega_N) - J(\omega_H - \omega_N)\right]}{\gamma_N\left[J(\omega_H - \omega_N) + \frac{3+4\Delta^2\omega_N^2}{3d^2}J(\omega_N) + 6J(\omega_H + \omega_N)\right]} \quad (4.4)$$

donde la función de densidad espectral, J, es la transformada de Fourier en coseno de la función de correlación total [88],

$$J(\omega) = 2\int_0^\infty \cos(\omega t)C(t)dt \quad (4.5)$$

y γ_H y γ_N son las relaciones giromagnéticas para el protón y el nitrógeno respectivamente, ω_H y ω_N son las frecuencias de Larmor para ^1H y ^{15}N, Δ el valor de la anisotropía de desplazamiento químico $\Delta = \delta_{||} - \delta_\perp$ ($\delta_{||}$ y $\delta_\perp$ son las componentes paralela y perpendicular del tensor de desplazamiento químico), y la constante d viene dada por:

$$d = \frac{\gamma_H\gamma_N\hbar}{r_{NH}^3}\frac{\mu_0}{4\pi} \quad (4.6)$$

donde $\hbar$ es la constante de Planck dividida por 2π, r_{NH} es la distancia de enlace amida $^{15}N -^1 H$ y μ_0 la permeabilidad del vacío.

Asumiendo que el movimiento global de la molécula es isotrópico, la función de correlación total se puede desglosar en factores como una función de correlación global y una función de correlación para los movimientos internos [89]. La función de correlación global sólo dependería del tiempo de correlacion total de la molécula τ_m y de la constante de difusión rotacional.

Suponiendo, por otra parte, que la función de correlación interna $C_I(t)$ decae a tiempos largos hacia un valor límite o valor de equilibrio y que lo hace de un modo exponencial se puede enunciar una expresión normalizada para dicha función. Dicha expresión contiene dos parámetros que caracterizan tanto el equilibrio de la función como su decaimiento.

$$C_I(t) = S^2 + (1 - S^2)e^{-\frac{t}{\tau_e}} \tag{4.7}$$

donde S^2, el parámetro de orden, es el valor de la función en el equilibrio (es decir, $C_I(\infty)$) y se puede demostrar que desarrollando $C_I(t)$ como el segundo polinomio de Legendre representa una medida de la restricción espacial del movimiento interno pudiendo variar entre 0 y 1 (0 para movilidad total, todas las orientaciones son posibles, 1 para restricción total, sólo una orientación es accesible). El otro parámetro τ_e, el tiempo de correlación efectivo, representa la velocidad del movimiento interno como cualquier tiempo efectivo.

Con estas suposiciones, Lipari y Szabo [89, 90] desarrollaron un formalismo libre de modelo en el que la densidad espectral se calcula según las expresiones anteriores dando por resultado,

$$J(\omega) = \frac{2}{5}\left[\frac{S^2\tau_m}{1 + (\omega\tau_m)^2} + \frac{(1 - S^2)\tau}{1 + (\omega\tau)^2}\right] \tag{4.8}$$

donde

$$\frac{1}{\tau} = \frac{1}{\tau_e} + \frac{1}{\tau_m} \tag{4.9}$$

En algunos casos, sin embargo, el movimiento interno del vector en cuestión no es posible describirlo mediante una escala de tiempo única. Esto quiere decir que la dinámica propia del vector necesita más de un tiempo efectivo de correlación (y por lo tanto más de un parámetro de orden) debido a que dicho movimiento más parece una composición de dos o más movimientos a diferentes escalas de tiempos (uno rápido y uno lento , por ejemplo). En estos casos, la expresión de $C_I(t)$ resulta ligeramente más complicada y por lo tanto la expresión de $J(\omega)$ también. En la mayoría de estos casos, basta con dos escalas de tiempos para definir correctamente el movimiento interno. Por lo tanto, factorizando el parámetro de orden S^2 en dos nuevos parámetros S_s^2 y S_f^2 correspondientes a un movimiento lento y otro rápido y haciendo lo mismo con τ_e en τ_s y τ_f tendremos descrito

el movimiento interno. Es posible, además simplificar la aproximación teniendo en cuenta que el τ_f será muy inferior respecto al τ_s. La expresión final de la función de densidad espectral quedará [91]

$$J(\omega) = \frac{2}{5}\left[\frac{S^2\tau_m}{1+(\omega\tau_m)^2} + \frac{S_f^2(1-S_s^2)\tau'}{1+(\omega\tau')^2}\right] \tag{4.10}$$

donde

$$S^2 = S_f^2 S_s^2 \tag{4.11}$$

y

$$\frac{1}{\tau'} = \frac{1}{\tau_s} + \frac{1}{\tau_m} \tag{4.12}$$

Como se puede comprobar, aplicando estas expresiones de $J(\omega)$ a las expresiones de R_1,R_2 y $HNOE$ según los casos y una vez conocido el valor de τ_m para la molécula en cuestión, tenemos una relación directa entre los parámetros de formalismo libre de modelo de Lipari-Szabo y las medidas experimentales de relajación. De este modo podemos evaluar la dinámica interna de una molécula mediante medidas de Resonancia Magnética Nuclear.

Una vez determinado τ_m para una molécula, R_1, R_2 y $HNOE$ pueden ser directamente calculados si los parámetros de orden y tiempos efectivos son conocidos. Sin embargo, la relación no es lineal por lo que el cálculo inverso de los parámetros de orden y tiempos efectivos de correlación a partir de las medidas de relajación no es directo ni puede realizarse de modo analítico.

La mayoría de métodos de obtención de los parámetros del formalismo libre de modelo de Lipari-Szabo, se basan en la minimización de una función de error que contiene las medidas experimentales y los parámetros teóricos [92, 109]. Este método presenta varios inconvenientes entre los que destacan dos por la importancia en la interpretación del resultado final.

En primer lugar, la selección del modelo que mejor describe el movimiento interno la ha de realizar de un modo subjetivo el usuario de los programas diseñados para ello en base a los resultados de minimización de la función de error. Debido al elevado carácter iterativo del proceso en

los primeros pasos no es fácil distinguir en muchas ocasiones entre los resultados de un modelo u otro. Es evidente que un error en estos primeros pasos pueden distorsionar el resultado final, no sólo del residuo implicado si no de todos los demás debido a que el cálculo de τ_m depende de los modelos seleccionados para cada residuo.

Por otra parte, el método de minimización produce como resultado el conjunto de parámetros para cada residuo y su modelo seleccionado cuya función de error es mínima. Sin embargo, la precisión de este resultado dista mucho de poder ser calculada de modo directo. De hecho, en la mayoría de los casos es preciso recurrir a complejos cálculos estadísticos que pueden mediatizar en cierto modo la incertidumbre de los resultados.

Debido a estas limitaciones, a las que se ha unido la necesidad de idear un método automático y objetivo de análisis de medidas de relajación, el objetivo del presente capítulo ha sido plantear un algoritmo de cálculo de los parámetros dinámicos del formalismo libre de modelo de Lipari-Szabo y su correspondiente traducción al lenguaje de programación C. La explicación del método desarrollado así como el análisis de resultados tanto simulados como experimentales y su comparación con los resultados obtenidos con programas similares y con otros formalismos ocupará las secciones siguientes. Como proteína ejemplo para la aplicación del nuevo programa de evaluación de los parámetros de la dinámica molecular se ha elegido el BPTI (Inhibidor Básico de la Tripsina de Pancreas Bovino).

4.1.3 Inhibidor Básico de la Tripsina de Páncreas Bovino

El inhibidor básico de la tripsina de páncreas bovina (wt-BPTI) es una pequeña proteína que contiene 58 aminoácidos. Ha sido utilizada en numerosos estudios teóricos y experimentales a causa de su gran estabilidad [93] y a que a pesar de ser de pequeño tamaño contiene muchos de los rasgos esenciales de proteínas mayores, además de que ha sido ampliamente estudiada por RMN. Se pueden encontrar tres estructuras de alta resolución resultas por cristalografía de Rayos X [94] y una de alta resolución determinada por RMN [95]. Las propiedades dinámicas de varios análogos al wt-BPTI también han sido estudiadas intensamente[90]. En estos estudios previos, algunas de las resonancias de protón en los segmentos polipeptídicos de los residuos 14 a 18 y 36 a 41 se ven ensanchadas a 36^o [96] y se pueden obtener muy pocos NOEs para estos protones. De los estudios de relajación con ^{13}C y ^{15}N, se ha podido elucidar el intercambio entre multiples conformaciones

a partir de los cortos tiempos de relajación de las resonancias de carbono α y nitrógeno amídico de las cisteína 14 y 38. La observación de la interconversión entre las dos conformaciones en el mutante [G36S]-BPTI y en el wt-BPTI se puede deducir un equilibrio dinámico entre dos confórmeros con diferente quiralidad en las cisteínas 14 y 38.

Debido al elevado número de estudios realizados sobre la dinámica de esta proteína y a que un análisis previo de los procesos de relajación y la dinámica del esqueleto carbonado de la proteína nativa ha sido realizado en nuestro grupo, se ha abordado el análisis de la dinámica intramolecular del wt-BPTI y del mutante [C30V,C51A]-BPTI mediante el nuevo método de búsqueda en rejilla. El mutante [C30V,C51A]-BPTI presenta peculiaridades que hacen muy interesante el estudio de su dinámica. Por una parte, a pesar de haber mutado únicamente dos residuos y que la estructura tridimensional se mantiene prácticamente idéntica a la del wt-BPTI, la estabilidad térmica del mismo es muy diferente a la de la proteína nativa. Debido a que no se han observado diferencias estructurales que permitan explicar esta distinta estabilidad térmica y que las mutaciones no son suficientes para producir una importante variación en la distribución electrostática de la proteína, el análisis de la dinámica intramolecular del mutante puede arrojar nuevas luces sobre el problema. Por otra parte, las mutaciones afectan a un puente disulfuro presente en la proteína nativa que en el mutante desaparecen. Es lógico suponer que la desaparición del mismo podría implicar un aumento en la flexibilidad de la proteína y por lo tanto en la movilidad de la misma.

4.2 Métodos

4.2.1 Medida de los parámetros de relajación

Las muestras fueron preparadas en disolución acuosa con concentraciones de 2mM para la proteína nativa enriquecida en ^{15}N y de 2.8 mM para el mutante [C30V,C51A]-BPTI también enriquecida en ^{15}N. Ambas muestran se tamponaron a un pH de 4.6 ± 0.1. El procedimiento de enriquecimiento isotópico utilizado fué el general descrito en la bibliografía [97]. La purificación fué realizada siguiendo el procedimiento general propuesto en la bibliografía [98]. La proteína se disolvió en una mezcla al 90 % en H_2O y al 10 % en D_2O. Todos los experimentos de RMN se realizaron en un espectrómetro Varian de 500 MHz a 309 K.

Las secuencias de pulsos utilizadas para medir los tiempos de relajación T_1 y T_2 del ^{15}N, y los NOE heteronucleares $^1H-^{15}N$ se pueden observar en la figura 4.1. Mediante el uso de desacoplamiento de protón con GARP se eliminaron los efectos de relajación cruzada entre los mecanismos de relajación dipolar y de anisotropía de desplazamiento químico (CSA) durante el tiempo de relajación T de los experimentos T_1 [99]. En las medidas de T_2, la secuencia CPMG espín-eco fué modificada para minimizar los efectos de correlación cruzada [100, 92]. Se aplicó un tiempo de espera de $2t$ entre pulsos sucesivos de 180^o de ^{15}N durante el CPMG ajustado a 768 μs ($t = 384\mu s$ y se aplicó un tiempo de espera t' igual a $2t$ + anchura de pulso de ^1H de 180^o. Se aplicaron pulsos de 180^o de protón cada 6.7 ms sobre el pico del cuarto eco de espín del CPMG para invertir el espín de ^1H rápidamente en comparación al decaimiento más rápido del doblete ^{15}N.

Los NOEs heteronucleares $^1H-^{15}N$ fueron medidos utilizando la secuencia de pulsos HNOE para gradiente de pulsos descrita en la bibliografía [102] . Se utilizó un tiempo de espera de reciclado largo (15 s, aproximadamente $5T_1$ del agua) para obtener la relajación completa de la magnetización del disolvente entre cada experimento, evitando los efectos de transferencia-saturación que resultan del agua parcialmente relajada. La saturación del protón amida durante el periodo NOE fué obtenida mediante el desacoplador GARP [99] para 3.7 segundos.

El cálculo de los parámetros de relajación T_1,T_2 y $HNOE$ se realizó mediante el ajuste de los datos experimentales de intensidad a diferentes tiempos de espera a las expresiones clásicas de la intensidad en función del tiempo. La intensidad se obtuvó midiendo el volumen de los picos cruzados en los espectros de correlación $^1H-^{15}N$. Las expresiones utilizadas en los ajustes fueron las siguientes:

$$I(t) = I_\infty - (I_\infty - I_0)e^{-t/T_1} \tag{4.13}$$

$$I(t) = I_0 e^{-t/T_2} \tag{4.14}$$

$$HNOE = I_{sat}/I_{eq} \tag{4.15}$$

donde I_0 e I_∞ son las intensidades inicial y final respectivamente y en sus correspondientes experimentos, T_1 y T_2 son los tiempos de relajación longitudinal y transversal respectivamente, $HNOE$ es el heteronoe y I_{sat} e I_{eq} son las intensidades del heteronoe con saturación y sin saturación. Sobre cada expresión se realizó un ajuste no lineal de mínimos cuadrados para los datos experimentales y se obtuvó el promedio de los resultados.

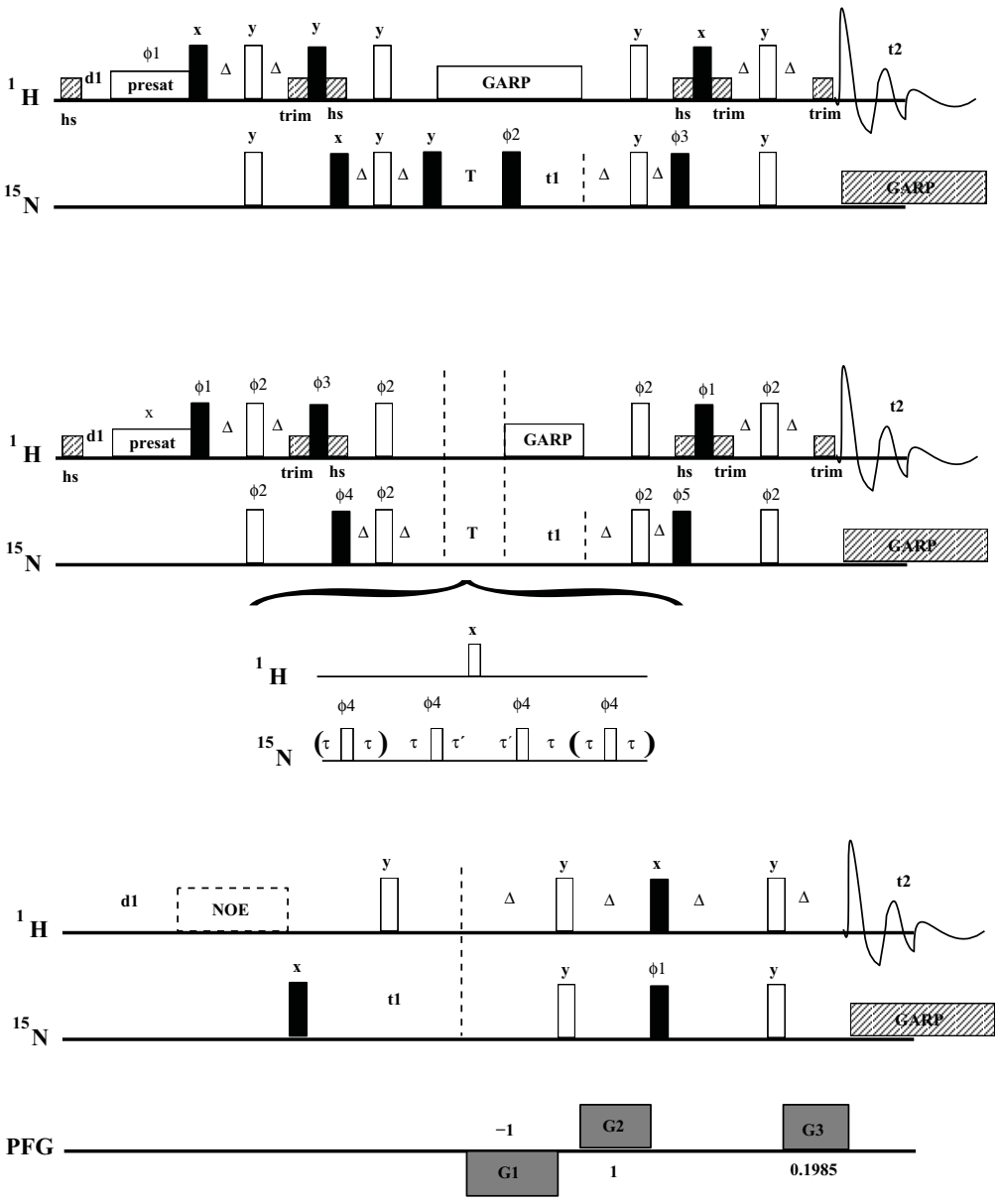


Figura 4.1: Secuencia de pulsos utilizada en la madida de los parámetros de relajación.

4.2.2 El método de minimización. Programa MODELFREE.

El método habitual utilizado en el análisis de medidas de relajación para obtener los parámetros dinámicos de proteínas ha sido el método de minimización MODELFREE desarrollado por A.Palmer [103]. Este método se basa en la minimización de una determinada función de error variando los parámetros dinámicos a calcular. La estrategia utilizada viene reflejada en el diagrama de flujo de la figura 4.2.

- Paso 1. Se toma un valor inicial del tiempo de correlación global $\tau_{m_i nit}$ estimado a partir de la relación promedio $< R_2/R_1 >$ para todos los residuos de la proteína.

- Paso 2. Para cada sistema de espín $^{15}N -^{1} H$, se ajustan los valores de R_1, R_2 y $HNOE$ a cada uno de los modelos de la tabla 4.1. El valor de $\tau_{m_i nit}$ se fija y se optimizan los parámetros de orden y los tiempos efectivos de correlación mediante minimización de la función objetivo:

$$< SSE >= \frac{(R_1 - R_1^*)^2}{(\Delta R_1)^2} + \frac{(R_2 - R_2^*)^2}{(\Delta R_2)^2} + \frac{(HNOE - HNOE^*)^2}{(\Delta HNOE)^2} \quad (4.16)$$

  donde R_1,R_2 y $HNOE$ son los valores experimentales de los parámetros de relajación, ΔR_1,ΔR_2 y $\Delta HNOE$ son las incertidumbres de los mismos y R_1^*, R_2^* y $HNOE^*$ son los correspondientes valores teóricos calculados para cada modelo de la tabla 4.1.

- Paso 3. Se escoge como modelo de cada residuo aquel que de un valor de función objetivo mínima para los parámetos de relajación de ese residuo. Cuando el resultado de diferentes modelos sea similar, se escoge el más simple sobre los más complejos. Este criterio tiene su justificación en los valores de S_f^2 en los modelos 5 y 6 y en R_{ex} en los modelos 4 y 6. Cuando el valor de S_f^2 en un modelo utilizando dos escalas de tiempo es mayor que 0.95, los modelos 2 y 4 son seleccionados sobre 5 y 6 ya que la escala de tiempo rápida está muy restrigida sobre la lenta y se puede suponer que $S^2 \sim S_s^2$. De un modo similar, cuando el valor de R_{ex} es inferior a 0.1 Hz la contribución del intercambio químico se puede considerar despreciable y los modelos 2 y 5 son preferidos sobre los 4 y 6 respectivamente.

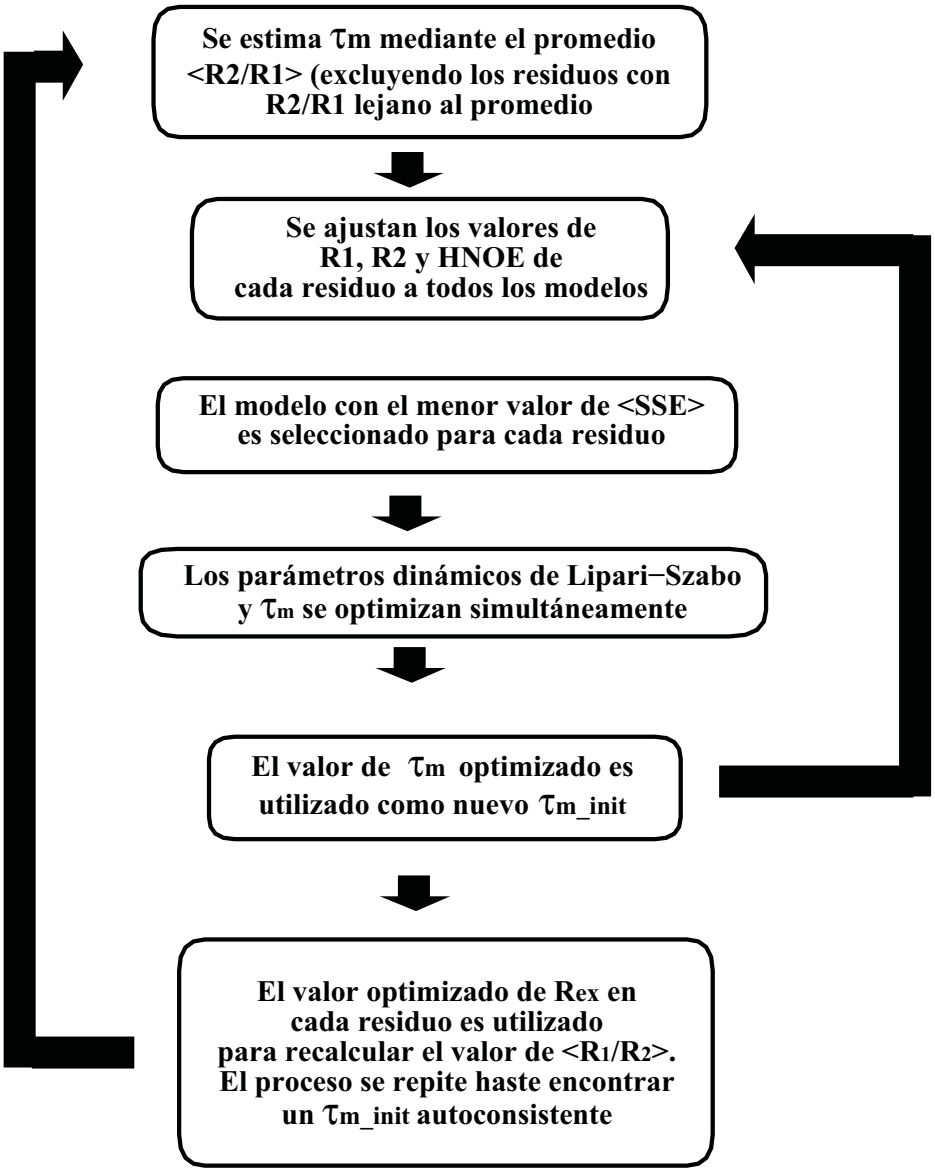


Figura 4.2: Diagrama de flujo del método de cálculo de los parámetros dinámicos de Lipari-Szabo y de selección del modelo utilizado por el programa MODEL-FREE.

Modelo	Función de densidad espectral	Parámetros
1	$J(\omega) = \frac{2}{5}\left[\frac{S^2\tau_m}{1+(\omega\tau_m)^2}\right]$	S^2
2	$J(\omega) = \frac{2}{5}\left[\frac{S^2\tau_m}{1+(\omega\tau_m)^2} + \frac{(1-S^2)\tau}{1+(\omega\tau)^2}\right]$	S^2,τ_e
3	$J(\omega) = \frac{2}{5}\left[\frac{S^2\tau_m}{1+(\omega\tau_m)^2}\right]$	S^2,R_{ex}
4	$J(\omega) = \frac{2}{5}\left[\frac{S^2\tau_m}{1+(\omega\tau_m)^2} + \frac{(1-S^2)\tau}{1+(\omega\tau)^2}\right]$	S^2,τ_e,R_{ex}
5	$J(\omega) = \frac{2}{5}\left[\frac{S^2\tau_m}{1+(\omega\tau_m)^2} + \frac{S_f^2(1-S_s^2)\tau'}{1+(\omega\tau')^2}\right]$	S_f^2,S_s^2,τ_s
6	$J(\omega) = \frac{2}{5}\left[\frac{S^2\tau_m}{1+(\omega\tau_m)^2} + \frac{S_f^2(1-S_s^2)\tau'}{1+(\omega\tau')^2}\right]$	S_f^2,S_s^2,τ_s,R_{ex}

Tabla 4.1: Tabla de modelos posibles de densidad espectral y expresiones de parámetros de relajación. En aquellos modelos que presentan como parámetro optimizable R_{ex},$R_2(experimental) = R_2 + R_{ex}$.

- Paso 4. Una vez seleccionado el modelo que mejor se ajusta a cada sistema de espín $^{15}N\ -^1\ H$, se optimizan los valores de τ_m y de los parámetros dinámicos internos de todos los residuos simultáneamente. Para la optimización de τ_m se añade las restricciones de que $0 \leq S^2 \leq 1, 0 \leq \tau_e \leq \tau_m$ y $0 \leq R_{ex}$ y se utiliza una rutina de interpolación cuadrática al mismo tiempo que S^2, τ_e y R_{ex} se optimizan para cada residuo mediante el algoritmo de Levenburg-Marquardt [58]. El valor de τ_m obtenido se denomina τ_{m_opt}.

- Paso 5. El proceso entero desde los pasos 2 al 4 es repetido utilizando τ_{m_opt} como valor inicial de τ_{m_init}.

- Paso 6. Cuando el valor de τ_m no varía se habrán identificado los residuos con un valor considerable de R_{ex} que se excluirán del cálculo de un nuevo τ_{m_init} a partir del promedio $< R_1/R_2 >$ de los residuos con R_{ex} muy pequeño. Con dicho valor de τ_{m_init} se repite el proceso entero desde los pasos 1 al 5.

Las incertidumbres de los valores obtenidos para los parámetros dinámicos se estiman mediante simulaciones de Monte Carlo, utilizando los métodos desarrollados por Palmer. Básicamente, se calculan valores de R_1^{bc}, R_2^{bc} y $HNOE^{bc}$ a partir de los valores de S^2, S_f^2, S_s^2, τ_e y R_{ex} estimados. Se

calculan distribuciones gaussianas con desviaciones estándar iguales a los errores experimentales para los parámetros de relajación con valores promedio R_1^{bc}, R_2^{bc} y $HNOE^{bc}$. Quinientos valores de R_1, R_2 y $HNOE$ se generan aleatoriamente de estas distribuciones y utilizados para obtener 500 conjuntos de parámetros dinámicos de Lipari-Szabo [103]. La distribución obtenida de parámetros dinámicos se considera adecuada para evaluar las incertidumbres si las desviaciones de los parámetros dinámicos S^2, τ_e y R_{ex} son inferiores al 15 %, 30 % y 10 % respectivamente. Para cada parámetro dinámico la dispersión se define como la diferencia promedio entre el valor obtenido mediante optimización y la media de la distribución simulada, dividida por el valor obtenido por optimización. Para movimientos complejos (modelos 5 y 6 en la tabla 4.1), los valores de S^2 son calculados como el producto de S_f^2 y S_s^2. Por lo tanto, las incertidumbres se podrán calcular mediante propagación de errores como

$$\delta_r = \left[(\delta_{rf}/S_f^2) + (\delta_{rs}/S_s^2)\right] S^2 \tag{4.17}$$

donde δ_{rf} y δ_{rs} son las desviaciones estándar en S_f^2 y S_s^2 respectivamente obtenidas de la simulación.

4.2.3 Aproximación gráfica

De acuerdo a toda medida experimental, las medidas de relajación pueden expresarse como $R_1 \pm \Delta R_1$, por ejemplo, correspondiendo a la medida y su incertidumbre. La esencia del procedimiento gráfico que vamos a desarrollar se basa en esta expresión. Básicamente se trata de calcular las superficies correspondientes a cada medida de relajación como $R = f(S^2, \tau_e)$ y superponer las proyecciones de las mismas para los valores de la medida de relajación permitida por $x \pm \Delta x$. Para una medida de relajación determinada con su error estimado, el area en el plano $S^2 - \tau_e$ entre las lineas de contorno $R_1 + \Delta R_1$ y $R_1 - \Delta R_1$ contiene todos los valores (S^2, τ_e) permitidos según la medida de R_1. Aplicando el mismo criterio a R_2 y $HNOE$ y hallando la intersección de las tres áreas tendremos determinada la región de valores (S^2, τ_e) permitidos según las medidas experimentales aplicadas [105].

Este método presenta la innegable ventaja, frente al método tradicional de minimización de una función de error, de que las incertidumbres se obtienen de un modo directo y no hace falta recurrir a métodos matemáticos susceptibles de producir artefactos. Además, se ha comprobado median-

te este método que las regiones permitidas no son en la inmensa mayoría de los casos rectangulares lo cual implica que las incertidumbres de los parámetros dinámicos no son independientes entre si. Se puede apreciar la forma de la zona permitida y como varía la incertidumbre en S^2 y τ_e según sus propios valores en la figura 4.3.

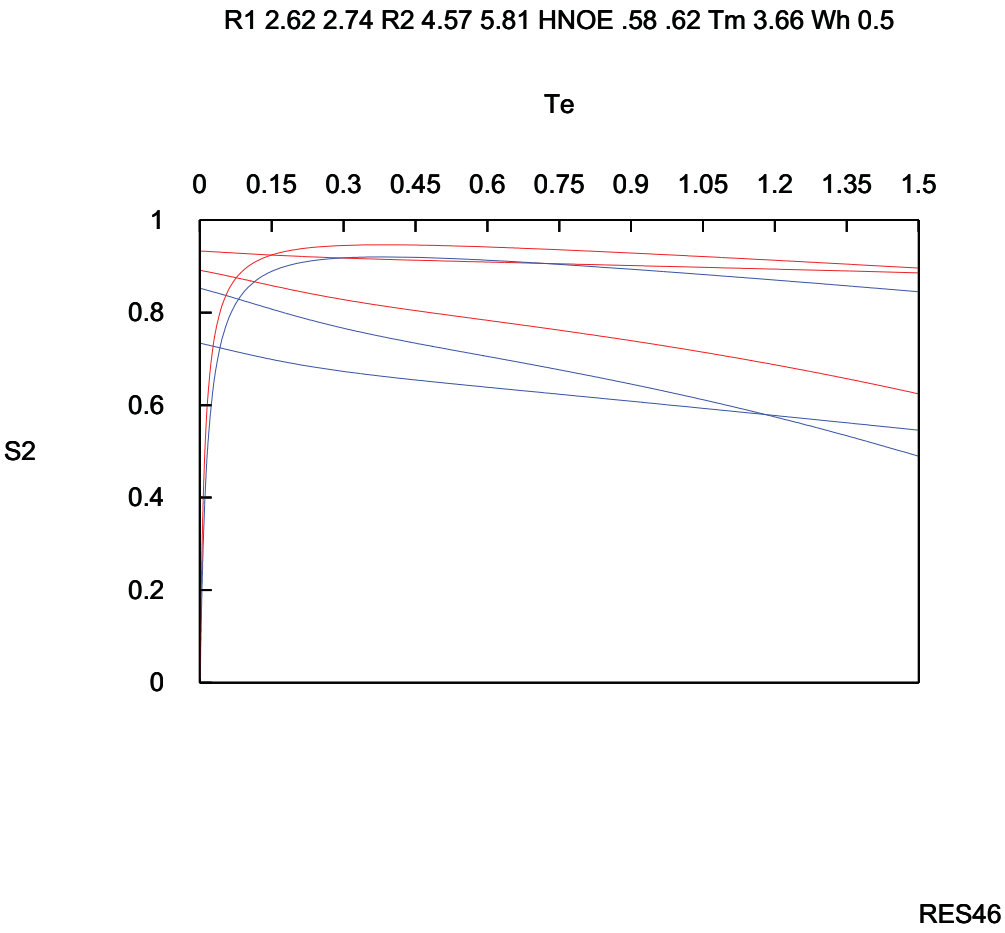


Figura 4.3: Visualización de la zona permitida para un aminoácido con valores de $R_1 = 2.68 \pm 0.6$, $R_2 = 5.81 \pm 0.62$ y $HNOE = 0.60 \pm 0.02$ para $\tau_m = 3.66ns$. Obsérvese la forma de la región permitida.

En un primer exámen, este método puede parecer excesivamente costoso debido a la necesidad de generar las gráficas correspondientes a los mapas de contornos. Sin embargo, se han programado una seria de macros de UNIX que utilizando el programa GNUPLOT [160] han permitido obtener dichas gráficas de modo semiautomático a partir de pocas indicaciones iniciales. Estas macros son capaces de producir gráficas para residuos hasta

el modelo 4 de la tabla 4.1 en formato *postscript*[59] de modo que el usuario pueda visualizar una a una en pantalla o imprimirlas directamente para mayor comodidad. El conjunto de macros descrito recibe el nombre de GRAPHIREX y se encuentra disponible en su totalidad en el apéndice D.

Obviamente, el método gráfico requiere realizar cortes a diferentes niveles cuando el número de grados de libertad exigido por el modelo supera el valor de 2.

4.2.4 Búsqueda en rejilla

El método de búsqueda en rejilla es una extensión directa del método gráfico descrito en la sección anterior. Simplemente es la aplicación de un algoritmo de barrido de las variables para la búsqueda de los valores que satisfacen el criterio expuesto de modo gráfico en un mapa de contornos. Se ha confeccionado un programa escrito en C para el cálculo de los parámetros dinámicos y valores de tiempos de correlación globales y locales utilizando los algoritmos propuestos cuyo nombre es SEARCHEL.

Para un valor dado de τ_m, R_1 puede ser directamente calculado si se proporcionan valores de S^2 y τ_e como ya hemos visto en la introducción teórica suponiendo un movimiento en una escala de tiempos única. La búsqueda en rejilla implica la exploración del espacio de parámetros de Lipari-Szabo calculando los valores teóricos de R_1, R_2 y $HNOE$ para cada conjunto de parámetros dinámicos y su comparación con los valores teóricos y su incertidumbre. En otras palabras, el método consiste básicamente en el cálculo de R_1, R_2 y $HNOE$ en un enrejillado de valores de los parámetros de Lipari-Szabo para cada punto del enrejillado. Para una medida dada de relajación con su incertidumbre estimada, sólo aquellos parámetros dinámicos con valores calculados de R_1, R_2 y $HNOE$ dentro de los límites definidos por $R_1 \pm \Delta R_1$, $R_2 \pm \Delta R_2$ y $HNOE \pm \Delta HNOE$ respectivamente, serán considerados valores permitidos.

Usando esta aproximación, se ha desarrollado un método automático de cálculo del tiempo de correlación global τ_m y los parámetros dinámicos. Las medidas de relajación de cada residuo son ajustadas a las expresiones de R_1, R_2 y $HNOE$ según los modelos de la tabla 4.1. El modelo escogido es aquel que proporcione solución permitida con el menor número de grados de libertad posibles. Según las expresiones de la tabla 4.1, es obvio que cuando un modelo proporciona solución válida en la búsqueda, los modelos que son simples adiciones de grados de libertad también han de dar

solución. Por ello, el modelo escogido es el más simple que proporciona un espacio permitido.

Asumiendo que los valores de τ_e son muy pequeños y que no existe intercambio químico ($R_{ex} = 0$), la relación R_2/R_1 es esencialmente independiente de S^2 y por lo tanto puede ser utilizada para evaluar el tiempo de correlación global, τ_m [101]. Normalmente, estas aproximaciones funcionan mejor en residuos que se ajustan al modelo 2. Para la evaluación del tiempo global de correlación se ha utilizado la siguiente estrategia:

- Paso 1. Se propone un valor inicial de τ_{m_init} calculado a partir del promedio $< R_2/R_1 >$ excluyendo aquellos valores con una relación R_2/R_1 fuera de un determinado intervalo en torno al promedio. El cálculo de este promedio se realiza de modo iterativo hasta que no se excluya ningún residuo respecto al paso anterior y el τ_{m-init} sea constante.

- Paso 2. El valor de τ_{m-init} calculado se utiliza para la determinación de aquellos residuos que tienen solución permitida según las expresiones del modelo 2. Estos residuos se etiquetan como residuos de modelo 2.

- Paso 3. Se calcula mediante barrido el intervalo de valores de τ_m que proporcionan solución según el modelo 2 en cada uno de los residuos etiquetados de modelo 2 en el paso anterior. El intervalo común de τ_m que da soluciones en los modelos etiquetados como modelo 2 se determina como la intersección entre todos los intervalos individuales de τ_m calculados para los residuos de modelo 2.

- Paso 4. Se determina un nuevo conjunto de residuos de modelo 2 utilizando el intervalo global de τ_m calculado en el paso 3. Este conjunto de residuos de modelo 2 es comparado con el utilizado en el paso 3 para la determinación del intervalo de τ_m global. Si ambos conjuntos coinciden, se ha alcanzado la consistencia y el cálculo contínua aceptando el intervalo de τ_m como el correcto. Si ambos conjuntos no coinciden, se calcula un nuevo τ_{m-init} utilizando el promedio $< R_2/R_1 >$ de los residuos del nuevo conjunto de modelos 2 y se vuelve al paso 2.

- Paso 5. El intervalo de τ_m obtenido se considera autoconsistente con los residuos que proporcionan solución permitida para las expresiones del modelo 2 de la tabla 4.1. En este modelo no se incluye la influencia del intercambio químico.

La fiabilidad del valor final de τ_m extraido de este cálculo depende en gran medida del número y porcentaje de residuos que satisfagan el modelo 2 utilizados en el cálculo y la cantidad de intercambio químico presente en la molécula siendo, por lo tanto, una primera indicación de la escala de tiempos de la movilidad intramolecular.

El intervalo de τ_m calculado se utiliza a continuación en la determinación del modelo más simple que satisfaga las medias de relajación y en el cálculo de los parámetros dinámicos permitidos de cada residuo. Para cada residuo, se escoge el modelo más simple con solución como el modelo que describe su dinámica. El criterio de esta selección está basado en la siguiente suposición: si un modelo simple puede dar conjuntos de parámetros dinámicos que cumplan las restricciones impuestas por las medidas de relajación, entonces los más complicados también lo harán para los mismos valores, aunque en sentido inverso esto no ocurra. Para evitar complejidad innecesaria en los modelos, seleccionamos el más simple asumiendo que los valores adicionales obtenidos en modelos más complejos serán despreciables por estar próximos a cero. El residuo cuyo movimiento es explicado exclusivamente por un modelo complejo no proporcionará solución en uno más simple.

Un aspecto muy importante en el hecho de que sólo se den como soluciones datos compatibles con las medidas experimentales es que cuando en un residuo, las medidas indiquen que no hay solución compatible, el residuo es descartado de cálculos posteriores. En los métodos de minimización, esto no ocurre y el residuo se incluye con el error correspondiente en los cálculos iterativos de τ_m aportando, por consiguiente, el correspondiente error a todo el proceso.

En las soluciones obtenidas por este método, se puede apreciar que los espacios permitidos de los parámetros dinámicos no son rectangulares como debiera ser en el caso de incertidumbres independientes, como ya habiamos observado en el método gráfico. Sin embargo, la relación entre las incertidumbres de los diferentes parámetros de Lipari-Szabo dista mucho de ser simple. En los resultados que proporciona el programa desarrollado, solo aparecen los valores máximos, mínimos y promedios de los intervalos de parámetros dinámicos calculados como si el espacio permitido si fuera rectangular. Sin embargo, aunque el usuario puede representar en cualquier momento dicho espacio mediante el método gráfico (ayudado de las soluciones obtenidas con el programa SEARCHEL) o con cualquier programa gráfico y el fichero de soluciones, el listado también proporciona un par

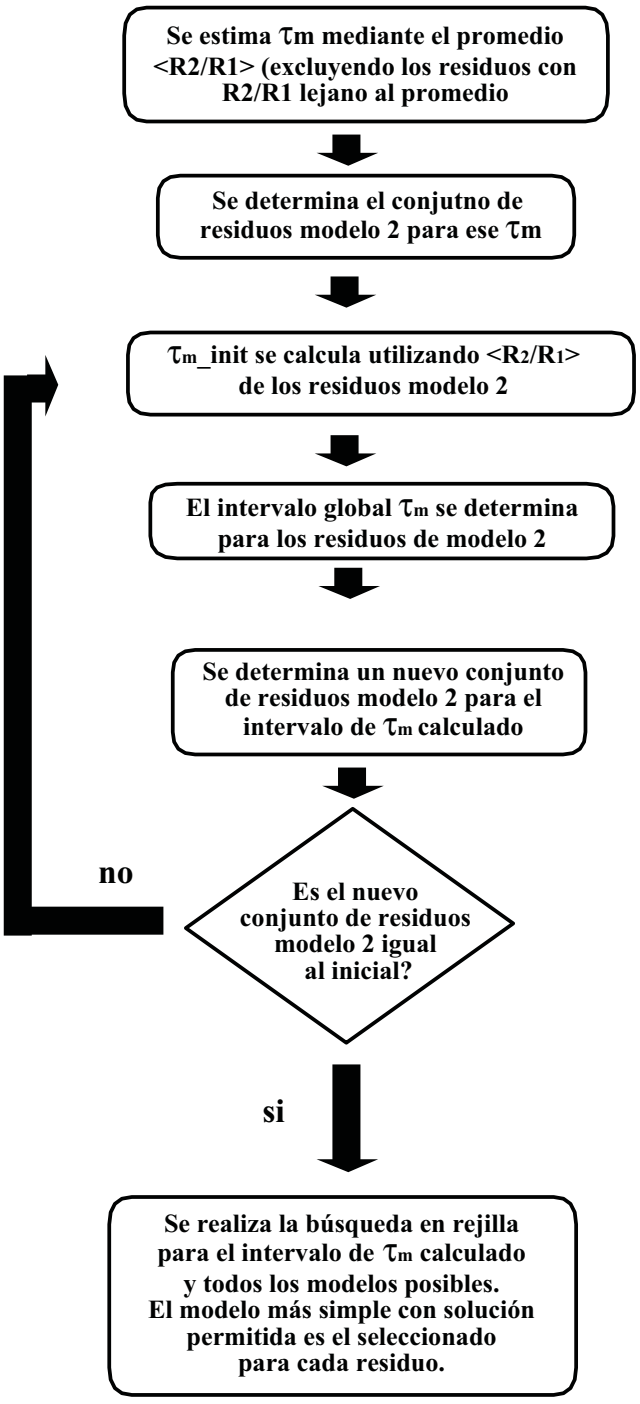


Figura 4.4: Diagrama de flujo del algoritmo de calculo de τ_m.

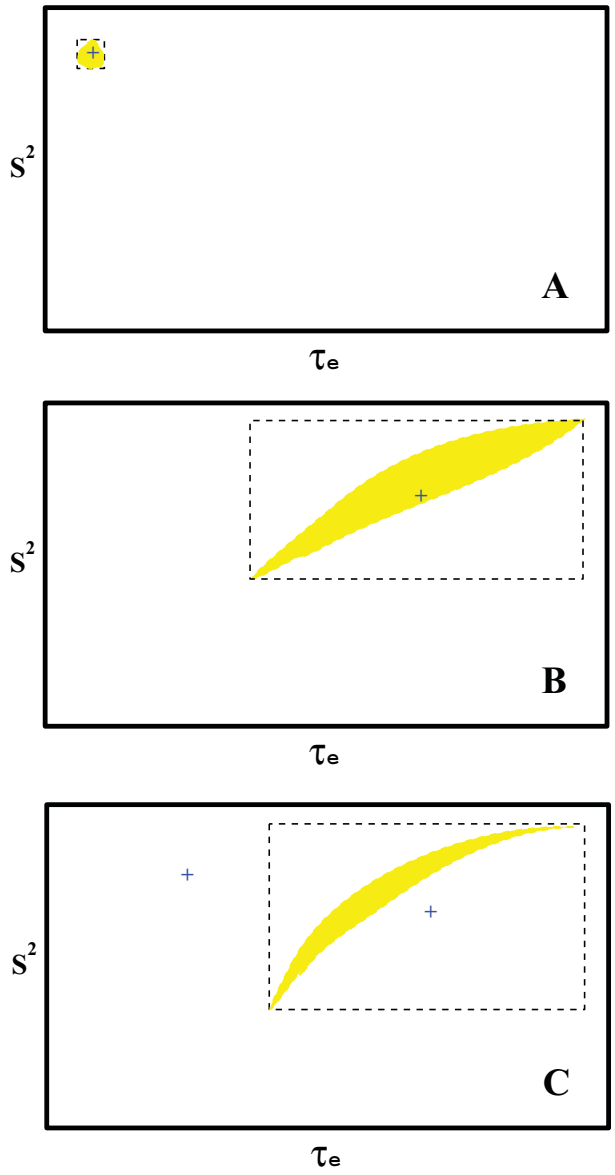


Figura 4.5: Ejemplo de las diferentes situaciones posibles en la forma de soluciones obtenidas mediante SEARCHEL. A) $\gamma \simeq 0.80$, tipo I; B) $\gamma \simeq 0.20$, tipo I; C) $\gamma \simeq 0.20$, tipo II.

de variables adicionales que pueden proporcionar una idea sobre la forma de la solución obtenida. Estas variables han sido denominadas tipo de solución y Gamma de solución. El tipo de solución es sólo un valor booleano que indica si el conjunto de valores promedio de la solución es un conjunto permitido o no (tipo I en la figura 4.5.A y 4.5.B, tipo II en la figura 4.5.C). La Gamma de solución es la relación entre el volumen permitido y el volumen de la relación rectangular. Cuanto más próxima a 1 sea el valor de Gamma más rectangular es la solución obtenida y menor es la dependencia entre las incertidumbres.

4.2.5 Análisis de densidad espectral reducida

La relajación cruzada dipolo-dipolo heteronuclear para los núcleos de ^1H y ^{15}N puede expresarse mediante la siguiente igualdad [106]:

$$R_{HN} = \frac{d^2}{4}\left[6J(\omega_H + \omega_N) + J(\omega_H - \omega_N)\right] \tag{4.18}$$

donde d tiene el expresado en la ecuación 4.6. Utilizando esta expresión y considerando que, para las proteínas el valor de la densidad espectral a frecuencia 0 es varios ordenes de magnitud superior a los valores a frecuencias más altas, se puede hacer una simplificación en las ecuaciones 4.2, 4.3 y 4.18 que hace posible la resolución algebraica de $J(0)$ y $J(\omega_N)$, es decir de las densidades espectrales reducidas. Suponiendo que $J(\omega_H) \simeq J(\omega_H - \omega_N) \simeq J(\omega_H + \omega_N)$, se puede deducir que,

$$J(0) \simeq \frac{3}{2(3d+c)}\left[-\frac{1}{2}R_1 + R_2 - \frac{3}{5}R_{HN}\right] \tag{4.19}$$

$$J(\omega_N) \simeq \frac{1}{(3d+c)}\left[R_1 - \frac{7}{5}R_{HN}\right] \tag{4.20}$$

$$J(\omega_H) \simeq \frac{1}{5d}R_{HN} \tag{4.21}$$

Sabiendo que la relajación cruzada dipolo-dipolo heteronuclear se puede relacionar con el $HNOE$ mediante,

$$HNOE = \frac{I_{sat} - I_{eq}}{I_{eq}} = \frac{\gamma_H}{\gamma_N}\frac{R_{HN}}{R_1} \tag{4.22}$$

y que al aparecer R_2 siempre como sumando la contribución de R_{ex} a la misma no afecta a las igualdades expuestas, se pueden deducir los valores de $J(0)$, de $J(\omega_H)$ y $J(\omega_N)$ una vez conocidos los valores de R_1, R_2 y $HNOE$.

Para los diferentes vectores NH en la misma proteína, el área bajo la curva de $J(\omega)$ permanece constante debido a que la energía de los campos magnéticos locales causantes de la relajación poseen un valor constante sobre el conjunto global [87]. Así, los valores más pequeños de $J(0)$ implican valores mayores a frecuencias más altas, indicando que los vectores NH se reorientan más rápidamente. Por lo tanto, los residuos cuyo valor de $J(0)$ esté por debajo del promedio con la correspondiente corrección de su desviación estándar, poseen movimientos internos rápidos [90]. Asimismo, los residuos con valores de $J(0)$ superiores al promedio más su desviación estándar poseen movimientos internos lentos.

Recientemente se ha propuesto que $J(\omega_N)$ y $J(\omega_H)$ están linealmente correlacionados con sus correspondientes valores de $J(0)$ mediante una relación lineal [107],

$$J(\omega_{N,H}) = \alpha J(0) + \beta \tag{4.23}$$

donde α y β son la pendiente y la ordenada en el origen del ajuste lineal correspondiente. El mayor error en los valores de $J(\omega_H)$ que en $J(\omega_N)$ hacen que el ajuste lineal de este último sea más preciso y fiable que el del primero. Utilizando la pendiente del ajuste y con la siguiente expresión [107],

$$2\alpha\omega_N^2\tau_m^3 + 5\beta\omega_N^2\tau_m^2 + 2(\alpha - 1)\tau_m + 5\beta = 0 \tag{4.24}$$

se puede determinar el tiempo de correlación global τ_m de la molécula.

4.3 Resultados y discusión

Los datos de R_1, R_2 y $HNOE$ utilizados tanto en el análisis de la dinámica del wt-BPTI como en el del mutante [C30V,C51A]-BPTI han sido extraídos de la tesis doctoral de María José Arnau [90] y de otros trabajos previos [108]. El análisis realizado con el programa MODELFREE de A.Palmer [103] también se ha extraido de la misma fuente. Los datos de R_1, R_2 y $HNOE$ del h-TGF-α han sido extraídos de la bibliografía [109].

4.3.1 Resultados de datos simulados

Como comprobación del método utilizado y del programa desarrollado,se realizó un cálculo de parámetros dinámicos a partir de datos simulados. Para ello, se utilizó la selección de modelos y el conjunto de parámetros dinámicos de Lipari-Szabo disponibles para el [C30V,C51A]-BPTI calculados mediante el programa MODELFREE [103] en estudios anteriores [90]. Utilizando dichos parámetros, las expresiones correspondientes a los modelos seleccionados para cada residuo y el valor de τ_m determinado para la molécula, se calcularon los valores de R_1, R_2 y $HNOE$ mediante el comando *simul* del programa SEARCHEL. Una vez obtenidos los valores teóricos de los parámetros de relajación, se procedió a aplicar el protocolo estándar de cálculo del programa SEARCHEL con sus valores de búsqueda y convergencia por defecto. Se les aplicó a estos valores un 1 % de error máximo con el objetivo de dificultar la búsqueda y comprobar si el programa es capaz de encontrar soluciones de pequeño tamaño.

El objetivo de esta simulación no es otro que el de comprobar que los valores obtenidos como resultado de la búsqueda en rejilla realizada por el programa SEARCHEL son autoconsistentes con las expresiones de cada uno de los modelos y que el criterio de selección del modelo es suficiente para garantizar un adecuado cálculo de los parámetros dinámicos y del valor de τ_m.

En la figura 4.6 se pueden apreciar las diferencias entre los valores obtenidos para el cálculo de los parámetros dinámicos de Lipari-Szabo y los utilizados en la simulación de los parámetros de relajación. Se puede comprobar que en la mayoría de los residuos las diferencias son nulas al nivel de precisión utilizado. La selección del modelo realizada por el programa SEARCHEL, por otra parte, coincide en casi todos los casos con el modelo propuesto para cada residuo como se puede comprobar en la tabla 4.2. Los residuos que difieren en la selección del modelo hecha por el SEARCHEL y el modelo introducido como origen, corresponden a valores del tiempo de correlación efectivo muy bajos y por lo tanto, el SEARCHEL ha encontrado soluciones con valor nulo y el modelo seleccionado ha sido aquel en que se despreciaba dicho tiempo. En los casos en que el cálculo de SEARCHEL proporciona una selección de modelo de mayor complejidad que la introducida, los datos de origen utilizados para el cálculo de los parámetros de relajación, son un subconjunto de la solución más amplia propuesta por SEARCHEL. Tal es el caso de los residuos GLY12 y LYS26.

Los resultados obtenidos nos permiten afirmar que el programa es ca-

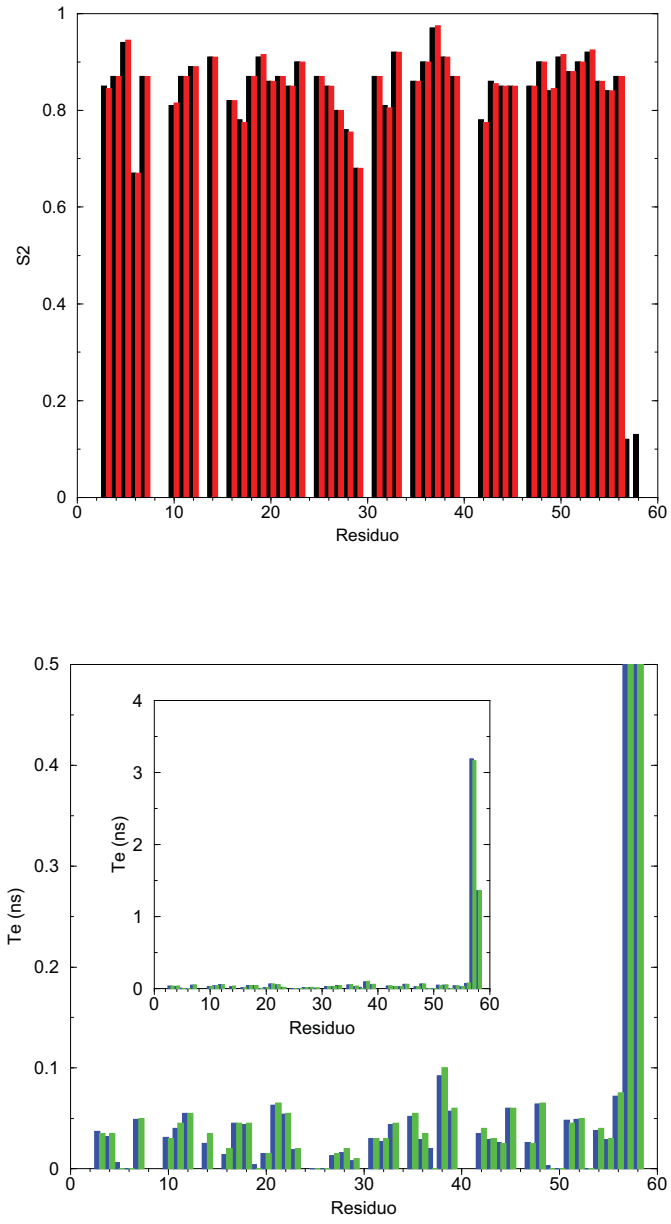


Figura 4.6: Comparación de los valores de los parámetros dinámicos de Lipari-Szabo para los resultados obtenidos de parámetros de relajación simulados (S^2 negro y τ_e azul) y los utilizados para la simulación (S^2 en rojo y τ_e en verde) con datos dinámicos del [C30V,C51A]-BPTI.

Residuo	Modelo A	Modelo B
ASP3	2	2
PHE4	2	2
CYS5	1	2
LEU6	3	3
GLU7	2	2
TYR10	2	2
THR11	2	2
GLY12	4	2
CYS14	2	4
ALA16	2	4
ARG17	2	2
ILE18	1	2
ILE19	2	2
ARG20	2	2
TYR21	2	2
PHE22	2	2
TYR23	1	2
ALA25	1	2
LYS26	4	2
ALA27	4	4
GLY28	4	4
LEU29	4	4
GLN31	2	2

Residuo	Modelo A	Modelo B
THR32	2	2
PHE33	2	2
TYR35	2	2
GLY36	4	4
GLY37	3	4
CYS38	4	4
ARG39	4	4
ARG42	4	4
ASN43	2	2
ASN44	2	2
PHE45	2	2
SER47	2	2
ALA48	2	2
GLU49	3	4
ASP50	1	2
ALA51	2	2
MET52	2	2
ARG53	1	2
THR54	4	4
CYS55	2	2
GLY56	2	2
GLY57	5	5
ALA58	5	5

Tabla 4.2: Tabla de modelos seleccionados para los resultados obtenidos de parámetros de relajación simulados y los utilizados para la simulación con datos dinámicos del [C30V,C51A]-BPTI. Modelo A: datos de origen. Modelo B: datos simulados.

paz de reproducir datos de manera autoconsistente. Las soluciones obtenidas, por otra parte, tienen una precisión adecuada a los datos utilizados para la simulación y, por ejemplo el τ_m, está definido en un intervalo mínimo dentro de los límites del procedimiento de búsqueda.

4.3.2 Resultados para el h-TGF-α

El h-TGF-α (TGF) es una pequeña proteína bien caracterizada por RMN, cuya dinámica ha sido estudiada utilizando el formalismo de modelo libre de Lipari-Szabo tal como propone Palmer [103, 109]. Los resultados de dicho análisis parecen indicar que esta proteína posee una dinámica de gran complejidad con numerosos residuos implicados en dinámicas de varias escalas temporales. Por otra parte, los resultados obtenidos con el programa MODELFREE parecen establecer movimientos imprecisos en un gran número de residuos, indicando en estos casos que, tanto la exactitud de los resultados como la selección del modelo puede estar sujeta a ciertos errores. Cuando, por ejemplo, el valor del tiempo de correlación efectivo τ_e tiene una incertidumbre del orden de su propio valor, es necesario preguntarse que sentido físico tiene una solución asi. Una solución de estas características indica que la dinámica de ese residuo en cuestión no corresponde a una o varias escalas de tiempo si no que la escala de tiempo está sujeta a variación total dentro de los límites impuestos por la correlación total de la molécula. Por ello, es preciso comprobar que la precisión de dichos resultados corresponde realmente a soluciones concretas del sistema y no a artefactos del método. En otras palabras, esta proteína puede considerarse un ejemplo significativo para el análisis comparativo entre MODELFREE y SEARCHEL.

Nombre	S^2	τ_e	S_f^2	S_s^2	R_{ex}
ASN6	0.650 ± 0.190	1.335 ± 1.235	–	–	–
ASP7	0.620 ± 0.050	0.060 ± 0.010	–	–	2.000 ± 1.100
CYS8	0.690 ± 0.030	0.165 ± 0.075	–	–	1.000 ± 0.500
THR13	0.870 ± 0.030	2.080 ± 2.030	–	–	1.150 ± 0.450
PHE15	0.765 ± 0.175	2.085 ± 2.025	–	–	1.900 ± 1.000
CYS16	0.765 ± 0.175	0.180 ± 0.060	–	–	2.200 ± 1.200
PHE17	0.860 ± 0.020	0.145 ± 0.045	–	–	–
HIS18	0.830 ± 0.010	1.200 ± 1.100	–	–	–
GLY19	0.855 ± 0.075	0.830 ± 0.760	–	–	–
THR20	0.895 ± 0.045	2.065 ± 2.045	–	–	–

Continua en la página siguiente

Continua de la página anterior

Nombre	S^2	τ_e	S_f^2	S_s^2	R_{ex}
CYS21	0.695 ± 0.255	2.090 ± 2.020	–	–	3.950 ± 1.550
ARG22	0.805 ± 0.145	0.120 ± 0.040	–	–	4.800 ± 1.000
PHE23	0.845 ± 0.035	1.955 ± 1.855	–	–	2.700 ± 0.900
LEU24	0.575 ± 0.315	2.120 ± 1.990	–	–	3.450 ± 1.350
VAL25	0.485 ± 0.375	1.455 ± 1.315	–	–	3.450 ± 1.150
GLN26	0.725 ± 0.155	2.100 ± 2.010	–	–	–
GLU27	0.425 ± 0.425	1.155 ± 1.035	–	–	1.750 ± 1.150
ASP28	0.750 ± 0.120	2.025 ± 1.855	–	–	–
LYS29	0.540 ± 0.320	0.055 ± 0.015	–	–	2.950 ± 1.050
ALA31	0.800 ± 0.050	1.560 ± 1.520	–	–	–
CYS32	0.795 ± 0.145	2.090 ± 2.020	–	–	1.600 ± 1.100
VAL33	0.710 ± 0.230	2.080 ± 2.030	–	–	2.350 ± 1.150
CYS34	0.725 ± 0.245	2.190 ± 0.300	–	–	–
HIS35	0.775 ± 0.035	2.080 ± 2.030	–	–	–
SER36	0.770 ± 0.180	2.085 ± 2.025	–	–	–
TYR38	0.780 ± 0.160	0.055 ± 0.005	–	–	–
VAL39	0.815 ± 0.005	0.080 ± 0.030	–	–	3.450 ± 0.850
GLY40	0.820 ± 0.040	1.970 ± 1.770	–	–	1.400 ± 0.700
CYS43	0.815 ± 0.125	3.305 ± 0.755	–	–	–
GLU44	0.860 ± 0.060	0.395 ± 0.095	–	–	–
HIS45	0.930 ± 0.030	0.105 ± 0.025	–	–	–
ALA46	0.860 ± 0.010	0.140 ± 0.010	–	–	–
ASP47	0.785 ± 0.005	0.190 ± 0.000	–	–	–
LEU48	0.500 ± 0.030	0.710 ± 0.580	–	–	–
LEU49	0.175 ± 0.175	0.520 ± 0.360	–	–	–
ALA50	0.093 ± 0.030	0.040 ± 0.010	0.665 ± 0.125	0.140 ± 0.140	–

Tabla 4.3: Tabla de parámetros dinámicos para el TGF calculados con SEARCHEL.

Debido a la naturaleza del método de cálculo propuesto en este trabajo, la incertidumbre de los resultados obtenidos es función únicamente de las medidas experimentales. De este modo, la posibilidad de artefactos matemáticos que perturben la precisión de los resultados se elimina íntrinsecamente.

Se utilizó como método de búsqueda el protocolo estándar del programa SEARCHEL con los parámetros de búsqueda por defecto. Los resultados conseguidos fueron, en general, bastante coincidentes con los descritos en la bibliografía obtenidos mediante el programa MODELFREE. En particular, la selección del modelo coincidió prácticamente en una gran parte de los casos como se puede comprobar en la tabla 4.5. Se puede comprobar

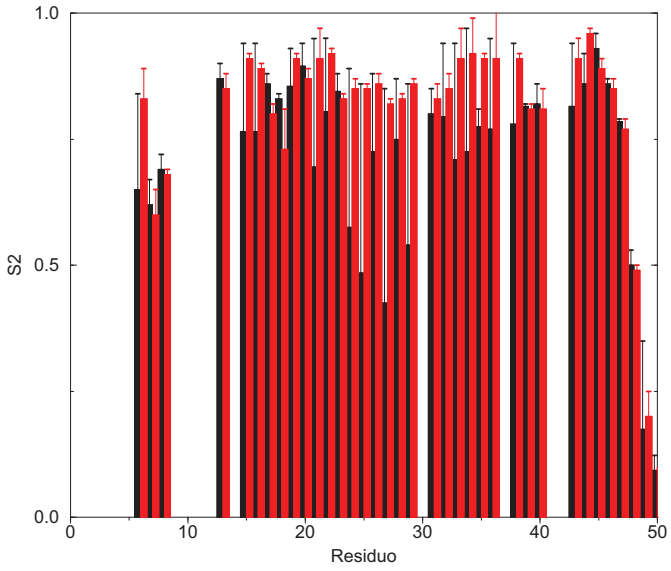


Figura 4.7: Comparación del parámetro dinamico de Lipari-Szabo S^2 para TGF según MODELFREE (rojo) y SEARCHEL (negro).

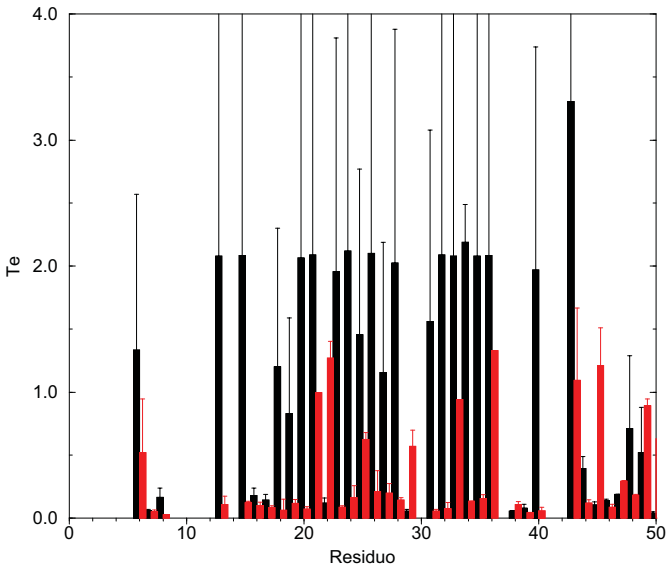


Figura 4.8: Comparación del parámetro dinamico de Lipari-Szabo τ_e para TGF según MODELFREE (azul) y SEARCHEL (verde).

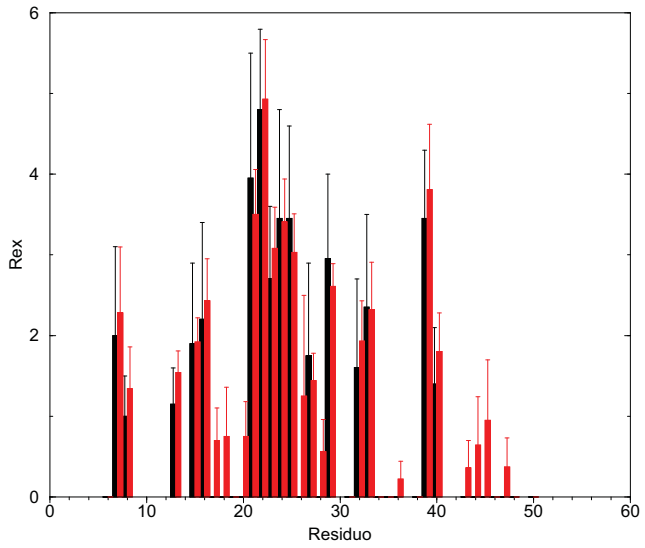


Figura 4.9: Comparación del parámetro dinamico de Lipari-Szabo R_{ex} para TGF según MODELFREE (rojo) y SEARCHEL (negro).

Ciclo	Num.Res. Modelo 2	$< R_2/R_1 >$	$\tau_{m_{init}}$	intervalo de τ_m
0	14	2.332	4.443	
1	18	2.174	4.130	4.08 - 4.58
2	16	2.154	4.149	4.10 - 4.30
3	17	2.127	4.042	3.94 - 4.24
4	19	2.122	4.032	3.78 - 4.13
5	18	2.112	4.012	3.81 - 4.16
6	19	4.032	4.032	3.78 - 4.13

Finalización por ciclo periódico.

Tabla 4.4: Tabla de tiempos de correlación global utilizados en el cálculo iterativo del intervalo de τ_m para el cálculo realizado sobre el TGF.

que las diferencias se centran en residuos con modelo 4 en la aproximación del MODELFREE que en la aproximación de SEARCHEL resultan de modelo 2. La mayoría de estos residuos tienen un valor de R_{ex} en las soluciones del MODELFREE despreciable En los valores de los parámetros dinámicos de Lipari-Szabo, se puede comprobar que aunque el valor promedio no sea coincidente en todos los residuos, los valores reportados en la biblografía se encuentran dentro de los intervalos indicados como solución en los resultados del programa SEARCHEL. En las figuras 4.7,4.8 y 4.9 se pueden comparar los parámetros dinámicos de Lipari-Szabo calculados por ambos métodos.

El valor de τ_m requiere un análisis particular (ver tabla 4.4). Aunque dicho valor no sea coincidente, hay que indicar que el valor óptimo de 3.76 ns reportado en la bibliografía se encuentra cercano al intervalo marcado por la búsqueda en rejilla (3.78 ns a 4.13 ns). Además, el intervalo de τ_m obtenido en el último ciclo de convergencia de SEARCHEL coincide plenamente con el intervalo de los tres valores de τ_m utilizados en el análisis del TGF de 4.17 ns a 3.76 ns [109]. Valores de τ_m inferiores a 3.76 resultan insatisfactorios para el análisis. Como ya se ha explicado en la sección de métodos, el intervalo de valores de τ_m obtenido mediante el programa SEARCHEL es el que es autoconsistente con aquellos modelos que no aportan contribución de intercambio químico a la relajación. Por ello, la solución es más descriptiva de un estado dinámico que si la solución es simplemente un valor con su incertidumbre. Hemos de recordar que la proteína siempre está en una situación dinámica y por lo tanto sus características físicas tanto globales como locales varían con el tiempo. Un intervalo de valores

Residuo	Modelo A	Modelo B
ASN6	2	2
ASP7	4	4
CYS8	4	4
THR13	4	4
PHE15	4	4
CYS16	4	4
PHE17	2	4
HIS18	2	4
GLY19	2	2
THR20	2	4
CYS21	4	4
ARG22	4	4
PHE23	4	4
LEU24	4	4
VAL25	4	4
GLN26	2	4
GLU27	4	4
ASP28	2	4

Residuo	Modelo A	Modelo B
LYS29	4	4
ALA31	2	2
CYS32	4	4
VAL33	4	4
CYS34	2	2
HIS35	2	2
SER36	2	4
TYR38	2	2
VAL39	4	4
GLY40	4	4
CYS43	2	4
GLU44	2	4
HIS45	2	4
ALA46	2	2
ASP47	2	5
LEU48	2	2
LEU49	2	5
ALA50	5	5

Tabla 4.5: Tabla de comparación de los modelos seleccionados según los programas SEARCHEL (Modelo A) y MODELFREE (Modelo B).

de τ_m representa mejor está realidad que un único valor. El hecho de que el valor obtenido mediante el método de minimización se encuentre dentro del intervalo calculado mediante el SEARCHEL nos indica que la solución del MODELFREE puede ser un caso particular de la solución más amplia proporcionada por el método de búsqueda en rejilla.

Se puede observar también que, aunque el número de residuos cuyo comportamiento sigue las expresiones del modelo 2 expuestas en la tabla 4.5 es bastante bajo, el valor de τ_m obtenido no dista mucho del dado en la bibliografía. Esto no nos indica que ambos sean estrictamente correctos ya que en ambos casos la selección de los modelos es crítica en el cálculo del valor de τ_m. Sin embargo, en el método propuesto utilizando el programa MODELFREE aplicado a los datos bibliográficos, la corrección del intercambio químico se aplica descartando los valores de R_2/R_1 de aquellos residuos con una fuerte contribución de R_{ex}, con lo que la selección del modelo es crucial en el cálculo. En el programa MODELFREE, al depender esta selección del criterio del usuario, los resultados pueden variar en dife-

rentes cálculos y por lo tanto, el valor de τ_m está sujeto a cierta subjetividad. En el programa SEARCHEL, dicha subjetividad es eliminada al seleccionar el modelo según un criterio constante a lo largo de todo el cálculo. Esto, unido al hecho de que en todo momento se trabaja con los intervalos de variables compatibles con los datos experimentales y no con las variables de mínimo error, aseguran que el resultado final será solución a los parámetros dinámicos de Lipari-Szabo consistentes con las medidas.

Concretamente, para un total de 25 enlaces amida, el mejor modelo para la descripción de su movilidad es coincidente entre el cálculo utilizando MODELFREE y el τ_m óptimo (3.76 ns) y la evaluación usando SEARCHEL y el intervalo final de τ_m (4.13 ns a 3.78 ns). Sin embargo, para los restantes 11 enlaces amida, la utilización de SEARCHEL con el intervalo optimizado de τ_m proporcionaba una descripción más simplificada de la movilidad interna. Por ejemplo, los residuos ASP47 y LAU49 se ajustan mejor a procesos de relajación multiexponencial (modelo 5) cuando se evalúa τ_m a partir del procedimiento óptimo descrito en la bibliografía [108] usando MODELFREE. En cambio, los mismos datos se ajustan mejor a un modelo de exponencial sencilla (modelo 2) cuando se usa el intervalo de τ_m idóneo en el sexto ciclo del cálculo realizado con SEARCHEL. Resultados similares se obtuvieron para los residuos PHE17, HIS18, THR20, LEU26, ASP28, SER36, LYS43, GLU44 y HIS45, para los que el mejor modelo se simplifica desde el más complejo de exponencial sencilla (modelo 4) con un valor único de τ_m de 3.76 ns más MODELFREE al de exponencial única (modelo 2) al utilizar SEARCHEL y el intervalo óptimo de τ_m (4.13 a 3.78 ns). Mientras que la aplicación del MODELFREE al ajuste del modelo más adecuado a la dinámica interna del TGF exigía la inclusión del parámetro de intercambio químico, R_{ex}, en 25 grupos amida (modelos 4 o 5), sólo 17 son necesarios según el formalismo utilizado en SEARCHEL aqui desarrollado. Es importante resaltar que la diferenecia se centra fundamentalmente en aquellos residuos para los que $R_{ex} \leq 1HZ$, cercano al límite experimental, y en donde dicha contribución puede considerarse en su límite de interpretación física. Concretamente 6 han sido los residuos (THR20, SER36, LYS43, GLU44, HIS45, ASP47 y LEU49) para los que la convergencia de SEARCHEL describe la movilidad del enlace amida mediante un modelo sencillo de exponencial simple (modelo 2). Un caso particular lo constituye el residuo ASP47, cuyos parámetros dinámicos se describen de manera individual [108] como un ejemplo de modelo complejo de movilidad. En cualquiera de los τ_m utilizados exige un ajuste a un modelo dinámico muy complejo, modelos 5 y 6, incluyendo valores de R_{ex} muy pequeños ($R_{ex} \leq 0.4Hz$). Sin embargo, la

aplicación del formalismo incluido en SEARCHEL permite una descripción de la dinámica del enlace amida del ASP47 mediante un modelo sencillo de simple exponencial (modelo 2), al tiempo que el valor de S^2 (0.785) es similar al óptimo según MODELFREE (0.77) con una incertidumbre inferior, 0.005 y 0.01 respectivamente. Además, el valor de τ_e resulta del mismo orden de magnitud, 190 ps SEARCHEL (ver tabla 4.3) y 330 ps MODELFREE con incertidumbres muy pequeñas. Este residuo puede ser considerado como un ejemplo tipo de la capacidad del formalismo desarrollado a través del programa SEARCHEL para un análisis más objetivo de la dinámica de proteínas mediante el modelo de Lipari-Szabo (ver figura 4.10).

Por otra parte, el análisis realizado mediante MODELFREE proporciona un elevado número de grupos amida con valores de S^2 entre 0.8 y 0.95 unidades con errores del orden de 0.05 [108] lo que podría ser interpretado como indicativo de una dinámica limitada en el TGF. Sin embargo, los valores promedio de S^2 obtenidos de SEARCHEL indican que sólo 14 de los 36 grupos amida medidos se encuentran entre 0.8 y 0.95, lo que estaría de acuerdo con una visión más dinámica de dicha proteína. Concretamente, un 39 % de los enlaces amida en TGF presentan valores de S^2 inferiores a 0.8. Simultáneamente, dichos grupos muestran un ensanchamiento apreciable de las resonancias de ^{15}N por intercambio químico indicativo de interconversiones entre múltiples conformaciones de la proteína a escala de microsegundos.

Como ya se ha indicado, los valores del parámetro de orden evaluado mediante MODELFREE se encuentran, en general, dentro del intervalo correspondiente de cálculo por SEARCHEL. No obstante, en algunos casos los valores de S^2 según MODELFREE se encuentran en el límite superior del intervalo propuesto por SEARCHEL, por lo que la interpretación de la movilidad puede ser parcialmente diferente. Veamos a continuación algunas de las diferenecias más significativas. El ASN6 es el último residuo de la primera hebra β, constituida sólo por 2 aminoácidos, PHE5 y ASN6, de la tercera triple hoja β del subdoninio amino terminal. A pesar de formar parte de una hoja β antiparalela, el ASN6 está localizado en la zona N-terminal de la proteína por lo que, como en la mayoría de los casos, presenta una movilidad relativamente importante, lo que sería coherente con el valor experimental del $HNOE$ (0.44), bastante inferior al valor medio en toda la proteína. Sin embargo, el valor de S^2 según el cálculo de MODELFREE usando el valor óptimo de τ_m es 0.83, relacionado habitualmente con una movilidad bastante restringida. Contrariamente, según SEARCHEL el valor de S^2 es de 0.650 para el residuo ASN6 lo que sería

más acorde con la localización espacial de dicho residuo y los valores de los parámetros de relajación de ^{15}N, particularmente el $HNOE$. Respecto a la movilidad asociada a la inversión de la cadena correspondiente a los residuos PHE17 e HIS18, según MODELFREE (con valores de S^2 de 0.71 a 0.86 y de 0.73 a 0.75 para ambos residuos respectivamente) parece trasladarse a los aminoácidos PHE15 y CYS16 con una dinámica mucho más compleja, incluyendo valores de R_{ex} próximos a los 2 Hz. Además, hay que destacar que con la excepción del residuo ASP47, las incertidumbres para PHE17 (0.02) y HIS18 (0.01) son las más pequeñas de toda la proteína indicando una excelente descripción de la movilidad de dichos residuos en el intervalo de τ_m optimizado. Por otro lado, PHE15 y CYS16 se encuentran localizados al final de una zona no ordenada de la proteina y en uno de los lugares en los que hay un equilibrio conformacional en la escala de microsegundos que implica al puente disulfuro CYS16-CYS32. Además, la interpretación física de valores de S^2 superiores a 0.90 en conjunción con movimientos que necesitan de un modelo complejo de exponencial única (modelo 4) resulta relativamente complicada. No obstante, aunque la metodología de SEARCHEL no elimina los valores de S^2 superiores a 0.85, el valor promedio de S^2 y su intervalo de incertidumbre (0.175) pueden ser indicativos de un comportamiento dinámico no sencillo, relacionado con una movilidad importante. Un análisis similar se podría realizar para los aminoácidos CYS21, LEU24, VAL25, LEU26, GLU27, ASP28, LYS29, CYS32 y VAL33. Aqui merecería una especial mención los resultados relativos a los residuos VAL25, GLU26, GLU27 y ASP28, que forman parte de la horquilla que conecta las dos hebras β, de GLY19 a LEU24 y de LYS29 a LYS34, de la triple hoja β de la parte amino terminal. Obviamente, estos 4 aminoácidos se encuentran en una parte estructural dotada con una movilidad relativamente importante, como asi queda reflejado en los valores de $HNOE$ [108]. Además, también están implicados procesos de cambio conformacional en la escala de microsegundos, específicamente el residuo GLU27. Sin embargo, según los resultados de la aplicación de MODELFREE, para todos ellos el valor de S^2 es superior a 0.8 unidades, es decir la interpretación dinámica directa según Lipari-Szabo, sin un conocimiento de su localización espacial, indicaría que el segmento VAL25 a ASP28 presenta una movilidad restringida. Por el contrario, los resultados conseguidos mediante el programa SEARCHEL pondrían de manifiesto una movilidad interna rápida en la escala de subnanosegundos o incluso superior, lo que sería mucho más coherente con su posición estructural. De nuevo la aplicación de SEARCHEL no sólo permite el análisis más objetivo de los datos de relajación, sino que posibilita una mejor interpretación de

los mismos respecto a la estructura tridimensional de la proteína. De manera similar, los aminoácidos HIS35 y SER36 que forman parte de un giro de tipo II y que incluye al aminoácido *bisagra*, SER36, entre los dos subdominios en los que se constituye el TGF, presenta valores de S^2 inferiores a 0.8 según SEARCHEL mientras que según MODELFREE los valores de S^2 son muy elevados y superiores a 0.9. En otras palabras, la interpretación física de los resultados obtenidos con MODELFREE, indicaría una zona relativamente rígida, en aparente contradicción con la localización espacial en una zona relativamente flexible de la molécula, punto de unión entre dos subdominios, N y C-terminales.

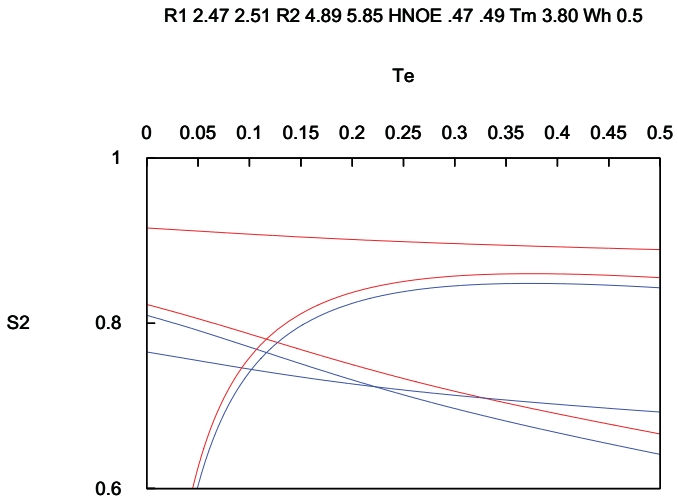


Figura 4.10: Solución gráfica calculada por GRAPHIREX para el residuo ASP47 del TGF con τ_m de 3.80 y modelo 2. Se puede apreciar la forma casi rectangular de la solución.

Desde el punto de vista metodológico, merece la pena destacar que SEARCHEL obtiene una solución convergente según modelo 2 para el residuo SER36, residuo que por otro lado, requiere de un modelo más complejo (modelo 4) según MODELFREE con la inclusión de un valor de R_{ex} de 0.22, que, tal como ya se ha comentado, se encuentra en los límites de error experimental y de una interpretación física adecuada, además de que el error en S^2 es uno de los mayores (0.3). En otras palabras, al igual que para el tramo secuencial LYS43 a ASP46, SEARCHEL encuentra soluciones homogéneas para los parámetros de relajación mediante la aplicación de un modelo exponencial simple, sin necesidad de recurrir a la inclusión de valores de R_{ex} inferiores a 1 HZ, y la utilización de modelos más complejos, que requieren al menos tres variables para sólo tres datos experimentales.

ASP472_TGF

La dinámica interna del resto de residuos de TGF queda descrita de manera similar mediante MODELFREE y SEARCHEL. No obstante, aunque la interpretación de la movilidad a través de los valores de S^2 de los residuos de la parte carboxiterminal resulta coincidente entre la aplicación de MODELFREE y SEARCHEL, sólo resulta necesaria la aplicación de un modelo exponencial múltiple (modelo 5) a los datos del residuo ALA50 según SEARCHEL (ver tabla 4.3), por lo que el análisis metodológico de los datos de relajación se simplifica otra vez. Los resultados de la dinámica molecular de TGF según SEARCHEL son coherentes con el equilibrio dinámico en el enlace entre la primera hebra β (PHE5 a ASN6) del núcleo de hoja β con la segunda hebra β (GLY19 a VAL25) a pH 7.0, propuesto en función de los datos de NOE y de intercambio de deuterio.

Asimismo, el movimiento de *bisagra* entre los dos subdominios de TGF queda reflejado por el hecho de bastantes residuos (PHE15, CYS16, CYS32, VAL33, VAL39 y GLY40) en la interfacie de dichos dominios, presentan un ensanchamiento de resonancias de ^{15}N consistente con una dinámica en la escala de microsegundos.

Finalmente, el tercer mecanismo de transición conformacional en TGF, la isomerización de puentes disulfuro [108] también puede ser explicado a través de los resultados de la dinámica según SEARCHEL. Prácticamente, una gran proporción de residuos que flanquean a los puentes disulfuro CIS8 / CYS21 y CIS16 / CIS32 (ver tabla4.3) muestran un ensanchamienso de línea por intercambio químico importante.

Desde el punto de vista de reconocimiento molecular, las zonas propuestas como epítopos para enlace con receptor [110] son PHE15 a HIS18 y VAL39 a GLU44. De acuerdo con los resultados de movilidad interna según SEARCHEL, tanto los residuos PHE15 y CIS16 como VAL39 y GLY40 son zonas de dinámica significativa, con valores de S^2 inferiores a 0.8 e intercambio químico, R_{ex} superiores a 1.5 Hz. En otras palabras, que los probables cambios en dichos movimientos internos pueden aportar una importante contribución a la energía libre del enlace entre el receptor de EGF y TGF.

4.3.3 wt-BPTI y [C30V,C51A]-BPTI

Datos iniciales

Al programa SEARCHEL se le introdujo como datos iniciales únicamente los datos de R_1, R_2 y $HNOE$ de las proteínas problema. El algoritmo propuesto permite con esa única información y sin la aportación de semillas iniciales que puedan mediatizar la búsqueda, la obtención de los valores de los parámetros dinámicos del formalismo libre de modelo de Lipari-Szabo y el valor del tiempo de correlación global τ_m. Los datos de relajación utilizados en el cálculo tanto para el wt-BPTI como para el mutante [C30V,C51A]-BPTI pueden observarse en las tablas 4.6 y 4.7 respectivamente [111].

Residuo	R_1	R_2	$HNOE$
ASP3	2.78±0.02	4.77±0.05	0.51±0.06
PHE4	2.90±0.03	4.46±0.08	0.55±0.09
CYS5	2.99±0.01	4.57±0.07	0.58±0.10
LEU6	2.96±0.03	4.07±0.08	0.49±0.05
GLU7	2.76±0.03	3.95±0.01	0.49±0.03
TYR10	2.63±0.03	4.05±0.09	0.51±0.08
THR11	2.80±0.01	4.56±0.14	0.52±0.08
GLY12	2.82±0.04	4.30±0.09	0.58±0.11
CYS14	2.68±0.03	5.75±0.03	0.63±0.04
LYS15	2.59±0.04	6.08±0.07	0.53±0.10
ALA16	2.66±0.01	3.96±0.03	0.53±0.10
ARG17	2.55±0.02	3.76±0.06	0.53±0.07
ILE18	2.11±0.03	4.97±0.23	0.48±0.06
ILE19	2.68±0.02	3.91±0.02	0.51±0.05
ARG20	2.72±0.04	3.89±0.06	0.48±0.05
TYR21	2.86±0.01	3.97±0.05	0.52±0.06
PHE22	2.71±0.03	4.05±0.02	0.52±0.06
TYR23	2.82±0.02	4.07±0.02	0.55±0.12
ASN24	2.83±0.06	4.20±0.08	0.53±0.04
ALA25	2.65±0.03	4.20±0.10	0.55±0.08
LYS26	2.63±0.01	4.31±0.01	0.58±0.07
ALA27	2.63±0.01	3.80±0.07	0.55±0.05

Continua en la página siguiente

Continua de la página anterior

Residuo	R_1	R_2	$HNOE$
GLY28	2.63±0.05	4.12±0.10	0.57±0.09
LEU29	2.77±0.01	4.00±0.02	0.54±0.12
CYS30	2.64±0.01	3.78±0.01	0.49±0.06
GLN31	2.83±0.07	3.97±0.09	0.51±0.04
THR32	2.53±0.01	3.73±0.01	0.52±0.06
PHE33	2.81±0.05	4.11±0.02	0.54±0.08
VAL34	2.68±0.02	3.83±0.01	0.50±0.13
TYR35	2.82±0.03	4.10±0.01	0.50±0.07
GLY36	2.83±0.06	4.44±0.08	0.54±0.10
GLY37	2.51±0.13	4.70±0.07	0.61±0.05
CYS38	2.80±0.02	5.03±0.01	0.54±0.07
ARG39	2.77±0.12	5.11±0.38	0.54±0.05
ALA40	2.70±0.04	4.90±0.03	0.56±0.07
ARG42	2.63±0.04	3.93±0.09	0.50±0.05
ASN43	2.77±0.01	4.15±0.01	0.54±0.07
ASN44	2.80±0.02	3.96±0.03	0.53±0.09
PHE45	2.71±0.04	3.86±0.12	0.55±0.08
LYS46	2.95±0.01	4.27±0.06	0.52±0.08
SER47	2.69±0.03	3.99±0.05	0.52±0.08
ALA48	2.72±0.03	4.33±0.05	0.53±0.07
GLU49	2.89±0.02	4.38±0.07	0.57±0.09
ASP50	2.94±0.01	4.67±0.07	0.56±0.09
CYS51	2.88±0.06	4.25±0.12	0.55±0.07
MET52	2.95±0.01	4.42±0.08	0.54±0.08
ARG53	2.80±0.02	4.69±0.04	0.59±0.08
THR54	2.74±0.06	4.28±0.01	0.59±0.09
CYS55	2.74±0.01	4.15±0.07	0.54±0.12
GLY56	2.72±0.03	4.12±0.07	0.50±0.09
GLY57	2.05±0.03	2.81±0.13	0.18±0.03
ALA58	1.54±0.02	2.58±0.09	-0.23±0.03

Tabla 4.6: Datos de relajación del wt-BPTI.

Residuo	R_1	R_2	$HNOE$
ASP3	2.71±0.06	4.22±0.29	0.55±0.02
PHE4	2.79±0.19	4.25±0.30	0.56±0.07

Continua en la página siguiente

Continua de la página anterior

Residuo	R_1	R_2	$HNOE$
CYS5	3.04±0.19	4.71±0.05	0.60±0.01
LEU6	2.08±0.01	4.04±0.60	0.67±0.18
GLU7	2.74±0.04	4.25±0.22	0.54±0.03
TYR10	2.59±0.13	4.03±0.14	0.54±0.05
THR11	2.77±0.12	4.36±0.05	0.55±0.02
GLY12	2.79±0.07	4.50±0.22	0.55±0.01
CYS14	2.85±0.15	5.70±0.31	0.58±0.01
ALA16	2.57±0.07	4.23±0.12	0.57±0.01
ARG17	2.49±0.05	3.92±0.27	0.51±0.01
ILE18	2.75±0.06	4.31±0.37	0.55±0.03
ILE19	2.82±0.01	4.08±0.42	0.61±0.12
ARG20	2.71±0.07	4.25±0.24	0.58±0.02
TYR21	2.74±0.06	4.41±0.25	0.53±0.03
PHE22	2.70±0.24	4.30±0.26	0.53±0.03
TYR23	2.80±0.03	4.33±0.34	0.58±0.01
ALA25	2.69±0.13	4.40±0.22	0.60±0.03
LYS26	2.65±0.09	4.17±0.19	0.65±0.13
ALA27	2.51±0.16	4.18±0.18	0.57±0.01
GLY28	2.37±0.15	4.15±0.26	0.58±0.01
LEU29	2.12±0.29	4.14±0.13	0.57±0.19
GLN31	2.71±0.05	4.36±0.25	0.56±0.01
THR32	2.56±0.04	3.97±0.44	0.55±0.04
PHE33	2.89±0.02	4.40±0.21	0.57±0.03
TYR35	2.71±0.10	4.27±0.29	0.53±0.07
GLY36	2.80±0.08	4.82±0.32	0.57±0.01
GLY37	3.03±0.18	5.14±0.05	0.59±0.05
CYS38	2.87±0.07	5.58±0.11	0.53±0.04
ARG39	2.74±0.05	7.35±0.04	0.53±0.04
ARG42	2.46±0.09	4.14±0.15	0.52±0.04
ASN43	2.71±0.02	4.39±0.32	0.56±0.01
ASN44	2.68±0.08	4.23±0.27	0.56±0.03
PHE45	2.73±0.06	4.13±0.19	0.52±0.01
LYS46	4.18±2.97	4.70±0.08	0.61±0.05
SER47	2.73±0.17	4.25±0.20	0.56±0.01
ALA48	2.93±0.12	4.40±0.19	0.55±0.01
GLU49	2.61±0.08	4.39±0.20	0.59±0.02

Continua en la página siguiente

Continua de la página anterior			
Residuo	R_1	R_2	$HNOE$
ASP50	2.83±0.09	4.58±0.10	0.62±0.02
CYS51	2.90±0.18	4.41±0.12	0.55±0.02
MET52	2.82±0.24	4.55±0.29	0.56±0.02
ARG53	2.85±0.12	4.62±0.16	0.61±0.01
THR54	2.70±0.29	4.50±0.10	0.55±0.02
CYS55	2.65±0.02	4.20±0.31	0.56±0.04
GLY56	2.75±0.04	4.20±0.36	0.52±0.01
GLY57	2.11±0.11	2.78±0.17	0.21±0.04
ALA58	1.38±0.15	1.76±0.23	-0.29±0.04

Tabla 4.7: Datos de relajación del mutante $C30V, C51A$-BPTI.

Protocolo de búsqueda con SEARCHEL

A ambos conjuntos de datos se les aplico la misma secuencia de comandos internos de SEARCHEL. Dicha secuencia de comandos implica los siguientes pasos:

- Comando **calctm**. Cálculo de τ_m con los valores de tamaño de enrejillado y criterios de convergencia definidos por defecto en el programa (ver apéndice A). No se toma límite inferior en la precisión de las medidas. El límite inferior en la precisión de las medidas es una precaución necesaria en la búsqueda en rejilla. Esto quiere decir que, en el caso de que una medida tenga una precisión relativa inferior a un valor indicado por el usuario (1 % habitualmente) el programa cambia los valores de incertidumbre de esta medida por el mínimo permitido. De este modo evitamos que el programa busque soluciones de tamaño inferior al enrejillado.

- Comando **calctree**. Cálculo de los parámetros dinámicos de Lipari-Szabo y selección del modelo. El límite de precisión en este paso esta puesto en un 1 % y se utilizan los límites de τ_m derivados del paso anterior. Los resultados obtenidos se almacenan del siguiente modo. Los parámetros de relajación que no han producido una solución en ninguno de los modelos ensayados se almacenan en un nuevo fichero para un posterior análisis. Los parámetros dinámicos de Lipari-Szabo para cada residuo según el modelo seleccionado se almacena en fiche-

Ciclo	Num.Res. Modelo 2	$< R_2/R_1 >$	$\tau_{m_{init}}$	intervalo de τ_m
0	14	1.527	2.47	
1	15	1.521	2.46	2.41 - 2.51
2	17	1.517	2.44	2.39 - 2.49
3	18	1.518	2.43	2.38 - 2.48
4	20	1.513	2.43	2.38 - 2.48
5	20	1.513	2.43	2.38 - 2.48

Tabla 4.8: Tabla de tiempos de correlación global utilizados en el cálculo iterativo del intervalo de τ_m para el wt-BPTI.

Ciclo	Num.Res. Modelo 2	$< R_2/R_1 >$	$\tau_{m_{init}}$	intervalo de τ_m
0	35	1.578	2.61	
1	36	1.576	2.60	2.55 - 2.70
2	37	1.574	2.59	2.54 - 2.69
3	37	1.574	2.59	2.54 - 2.69

Tabla 4.9: Tabla de tiempos de correlación global utilizados en el cálculo iterativo del intervalo de τ_m para el [C30V,C51A]-BPTI.

ros individuales para una posterior representación gráfica mediante el programa gnuplot. Hay tambien un resumen de resultados para la realización de tablas y gráficas globales.

- Comando **init**. Se inicializan los tamaños de enrejillado a valores más precisos al tiempo que se aumentan los límites de exploración para analizar los residuos que no han dado solución con los parámetros por defecto. Los valores utilizados se pueden ver en el fichero *generoso* (ver apéndice A).

- Comando **calctree**. Se vuelve a realizar la búsqueda en rejilla de los parámetros dinámicos de Lipari-Szabo exclusivamente para los residuos que no proporcionan solución con los parámetros por defecto utilizando criterios menos estrictos de convergencia y aumentando el límite de precisión al 2 %.

El protocolo estándar descrito permite que la selección del modelo la realice el programa utilizando el comando calctree en lugar de calcall. Este último comando realiza la búsqueda en rejilla para todos los modelos del 1 al 5 para todos los residuos. El modelo 6 de la tabla 4.1 no se utiliza en este programa debido a que siendo el número de grados de libertad superior al

número de restricciones aplicables las soluciones son de un tamaño excesivo para hacer válida cualquier interpretación. La solución a este problema sería aumentar el número de restricciones mediante medidas a diferentes campos magnéticos, por ejemplo. Calculando todos los modelos mediante calcall, se permite al usuario elegir el modelo de modo que, aunque en principio el programa está diseñado para un análisis automático de las soluciones, el análisis manual no está descartado. Una vez el usuario hubiera elegido el modelo, el comando *calcone* calcularía las soluciones para el modelo escogido.

Resultados con SEARCHEL

Los resultados de los parámetros dinámicos de Lipari-Szabo utilizando el protocolo automático descrito anteriormente se pueden observar en las tablas 4.10 y 4.11.

Nombre	S^2	S_f^2	S_s^2	τ_e	R_{ex}	Gamma	Tipo
ASP3	0.905 ± 0.005	–	–	–	0.500 ± 0.100	0.278	I
PHE4	0.945 ± 0.005	–	–	–	–	0.833	I
CYS5	0.975 ± 0.005	–	–	–	–	0.667	I
LEU6	–	–	–	–	–	0	II
GLU7	–	0.675 ± 0.235	0.920 ± 0.020	4570 ± 2910	–	0.034	II
TYR10	0.855 ± 0.005	–	–	–	–	0.833	I
THR11	0.910 ± 0.010	–	–	–	0.250 ± 0.150	0.444	I
GLY12	0.920 ± 0.010	–	–	–	–	0.778	I
CYS14	0.895 ± 0.025	–	–	-265 ± 235	1.600 ± 0.100	0.046	II
LYS15	0.845 ± 0.015	–	–	–	2.100 ± 0.100	0.333	I
ALA16	0.855 ± 0.015	–	–	45 ± 45	–	0.092	II
ARG17	0.820 ± 0	–	–	25 ± 5	–	0.333	II
ILE18	0.690 ± 0	–	–	–	1.800 ± 0.200	0.333	II
ILE19	–	0.875 ± 0.035	0.895 ± 0.005	1940 ± 480	–	0.062	II
ARG20	–	0.820 ± 0.030	0.915 ± 0.005	2035 ± 365	–	0.053	II
TYR21	–	–	–	–	–	0	II
PHE22	0.880 ± 0	–	–	–	–	0.333	II
TYR23	–	0.810 ± 0.030	0.955 ± 0.005	2115 ± 315	–	0.05	II
ASN24	0.910 ± 0.010	–	–	–	–	0.444	I
ALA25	0.870 ± 0.010	–	–	20 ± 20	–	0.2	II
LYS26	0.855 ± 0.005	–	–	–	0.250 ± 0.050	0.417	II
ALA27	–	–	–	–	–	0	II
GLY28	0.860 ± 0.010	–	–	–	–	0.444	I
LEU29	–	0.805 ± 0.045	0.945 ± 0.005	2085 ± 345	–	0.063	II
CYS30	–	–	–	–	–	0	II
GLN31	–	–	–	–	–	0	II
THR32	–	0.895 ± 0.075	0.900 ± 0.070	1265 ± 1215	–	0.107	II
PHE33	0.890 ± 0	–	–	70 ± 0	–	0.333	II
VAL34	–	0.720 ± 0.070	0.935 ± 0.015	2060 ± 420	–	0.092	II

Continua en la página siguiente

Continua de la página anterior

Nombre	S^2	S_f^2	S_s^2	τ_e	R_{ex}	Gamma	Tipo
TYR35	–	0.825 ± 0.065	0.955 ± 0.015	1915 ± 565	–	0.102	II
GLY36	0.930 ± 0.010	–	–	–	–	0.333	II
GLY37	0.815 ± 0.035	–	–	–	0.800 ± 0.200	0.1	II
CYS38	0.910 ± 0.010	–	–	–	0.750 ± 0.050	0.333	II
ARG39	0.905 ± 0.035	–	–	–	0.900 ± 0.500	0.655	I
ALA40	0.880 ± 0.010	–	–	–	0.800 ± 0.100	0.296	I
ARG42	0.860 ± 0.010	–	–	–	–	0.333	II
ASN43	0.900 ± 0	–	–	–	–	0.333	II
ASN44	–	–	–	–	–	0	II
PHE45	–	0.900 ± 0.020	0.900 ± 0	1835 ± 545	–	0.055	II
LYS46	0.835 ± 0.035	–	–	1995 ± 435	–	0.081	II
SER47	0.870 ± 0	–	–	–	–	0.333	II
ALA48	0.885 ± 0.005	–	–	–	0.200 ± 0.100	0.333	I
GLU49	0.940 ± 0.010	–	–	–	–	0.556	I
ASP50	0.960 ± 0.010	–	–	–	0.200 ± 0.100	0.296	I
CYS51	0.930 ± 0.010	–	–	–	–	0.444	I
MET52	0.965 ± 0.005	–	–	–	–	0.5	II
ARG53	0.910 ± 0.010	–	–	–	0.450 ± 0.050	0.333	I
THR54	0.905 ± 0.005	–	–	–	–	0.333	II
CYS55	0.895 ± 0.005	–	–	–	–	0.667	I
GLY56	0.885 ± 0.005	–	–	–	–	0.833	I
GLY57	–	0.535 ± 0.235	0.775 ± 0.025	1475 ± 1005	–	0.218	II
ALA58	–	–	–	–	–	0	II

Tabla 4.10: Tabla de parámetros dinámicos de Lipari-Szabo según han sido calculados por SEARCHEL para el wt-BPTI. También han sido tabulados los parámetros especiales Gamma y tipo.

Nombre	S^2	S_f^2	S_s^2	τ_e	R_{ex}	Gamma	Tipo
ASP3	0.865 ± 0.025	–	–	35 ± 25	–	0.389	I
PHE4	0.885 ± 0.045	–	–	–	–	0.825	I
CYS5	0.940 ± 0.010	–	–	–	–	0.333	II
LEU6	0.705 ± 0.005	–	–	-60 ± 10	–	0.083	II
GLU7	0.870 ± 0.020	–	–	45 ± 35	–	0.381	I
TYR10	0.830 ± 0.030	–	–	–	–	0.393	I
THR11	0.875 ± 0.025	–	–	35 ± 25	–	0.222	I
GLY12	0.895 ± 0.025	–	–	45 ± 25	–	0.285	I
CYS14	0.920 ± 0.040	–	–	–	1.200 ± 0.500	0.28	I
ALA16	0.850 ± 0	–	–	–	–	0.25	II
ARG17	0.785 ± 0.015	–	–	40 ± 10	–	0.292	I
ILE18	0.885 ± 0.015	–	–	–	–	0.25	II
ILE19	0.910 ± 0.010	–	–	–	–	0.333	I
ARG20	0.870 ± 0.020	–	–	–	–	0.9	I
TYR21	0.875 ± 0.025	–	–	60 ± 40	–	0.444	I
PHE22	0.865 ± 0.065	–	–	85 ± 75	–	0.295	II
TYR23	0.905 ± 0.005	–	–	–	–	0.5	I
ALA25	0.870 ± 0.040	–	–	–	–	0.667	I
LYS26	0.855 ± 0.025	–	–	–	–	0.875	I
ALA27	0.845 ± 0.015	–	–	–	–	0.25	II
GLY28	0.805 ± 0.005	–	–	–	–	0.375	I

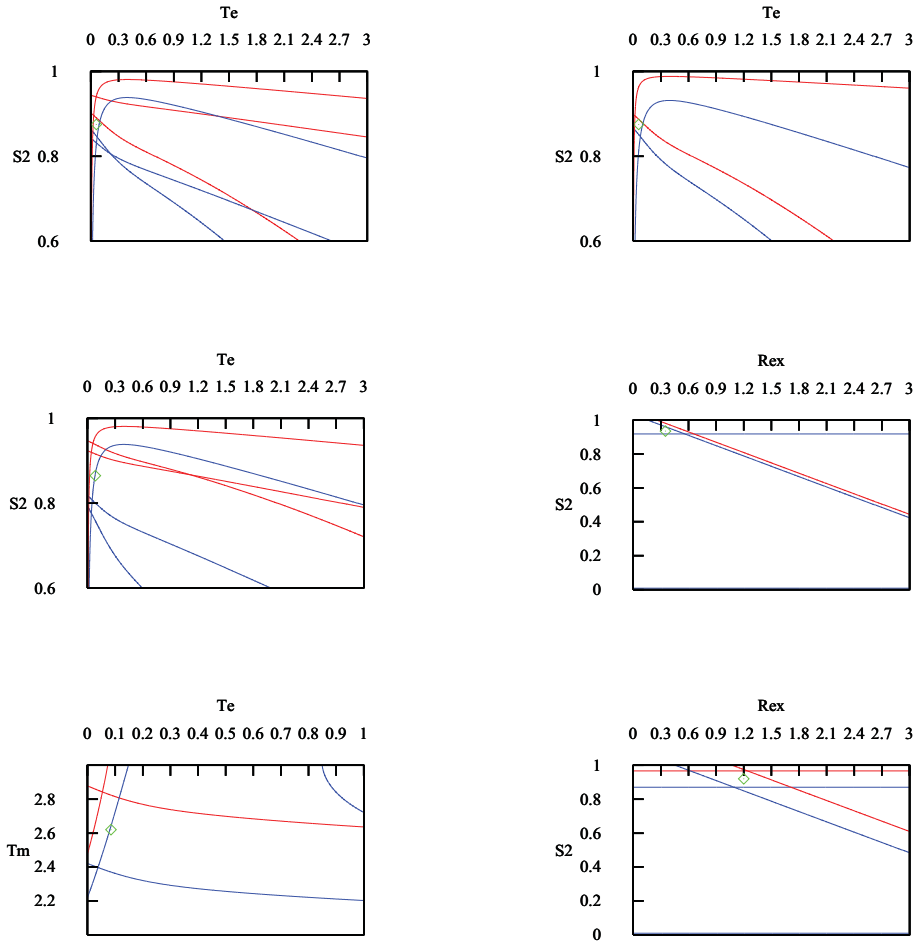
Continua en la página siguiente

Continua de la página anterior

Nombre	S^2	S_f^2	S_s^2	τ_e	R_{ex}	Gamma	Tipo
LEU29	0.680 ± 0.090	–	–	–	0.750 ± 0.550	0.205	I
GLN31	0.870 ± 0.020	–	–	25 ± 15	–	0.263	I
THR32	0.830 ± 0.010	–	–	–	–	0.417	I
PHE33	0.935 ± 0.005	–	–	–	–	0.375	II
TYR35	0.870 ± 0.030	–	–	–	–	0.929	I
GLY36	0.930 ± 0	–	–	–	–	0.25	II
GLY37	0.960 ± 0.040	–	–	–	0.350 ± 0.250	0.171	II
CYS38	0.910 ± 0.030	–	–	120 ± 110	1.050 ± 0.250	0.155	I
ARG39	0.875 ± 0.025	–	–	60 ± 50	3 ± 0.200	0.117	I
ARG42	0.795 ± 0.015	–	–	40 ± 20	–	0.225	II
ASN43	0.870 ± 0.010	–	–	25 ± 15	–	0.271	I
ASN44	0.865 ± 0.025	–	–	–	–	0.417	I
PHE45	0.865 ± 0.015	–	–	50 ± 10	–	0.271	II
LYS46	0.950 ± 0.030	–	–	–	–	0.464	I
SER47	0.865 ± 0.045	–	–	25 ± 15	–	0.319	I
ALA48	0.920 ± 0.020	–	–	45 ± 25	–	0.283	I
GLU49	0.850 ± 0.010	–	–	–	–	0.5	I
ASP50	0.930 ± 0.030	–	–	-255 ± 245	–	0.104	II
CYS51	0.895 ± 0.035	–	–	40 ± 30	–	0.371	I
MET52	0.930 ± 0.050	–	–	–	–	0.25	II
ARG53	0.930 ± 0.040	–	–	-255 ± 245	–	0.067	II
THR54	0.905 ± 0.035	–	–	55 ± 45	–	0.309	I
CYS55	0.855 ± 0.005	–	–	–	–	0.875	I
GLY56	0.875 ± 0.015	–	–	65 ± 15	–	0.328	I
GLY57	–	0.445 ± 0.245	0.795 ± 0.055	1790 ± 900	–	0.57	II
ALA58	–	0.265 ± 0.265	0.750 ± 0.210	990 ± 900	–	0.062	II

Tabla 4.11: Tabla de parámetros dinámicos de Lipari-Szabo según han sido calculados por SEARCHEL para el $C30V, C51A$-BPTI. También han sido tabulados los parámetros especiales Gamma y tipo.

Se puede apreciar a primera vista en dichas tablas una dinámica más compleja para el caso de la dinámica del wt-BPTI ya que el número de residuos con soluciones correspondientes a las expresiones del modelo 4 de la tabla de modelos 4.1 es mayor en este caso. Por otra parte, los tiempos de correlación global τ_m también son ligeramente diferentes. La imprecisión en los enlaces de movimiento lento (valores de τ_e elevados) es bastante elevada. Esto, unido a la forma no rectangular de las soluciones obtenidas en el método por búsqueda en rejilla, indica que la dinámica de dichos residuos está gobernada por diferentes factores en las diferentes escalas de tiempo. Es fácil determinar que residuos tienen una solución más o menos cuadrada observando el valor de gamma y el tipo de solución obtenida. En la figura 4.3 se pueden apreciar algunos ejemplos de las diferentes situaciones posibles descritas en la sección de métodos 4.5. Como se puede comprobar, la forma de la solución corresponde bastante bien con los valores reportados para gamma y el tipo de solución.



A) TYR21 BPTI−m γ = 0.444, type I
B) PHE22 BPTI−m γ = 0.295, type II
C) PHE22 BPTI−m γ= 0.295, type II
D) ARG39 BPTI−m γ = 0.117, type I
E) GLY37 BPTI−m γ = 0.171, type II
F) ASP50 BPTI−m γ = 0.107, type II

Figura 4.11: Ejemplos de soluciones gráficas tal y como las muestra GRAPHIREX para algunos residuos representativos del $C30V, C51A$-BPTI.

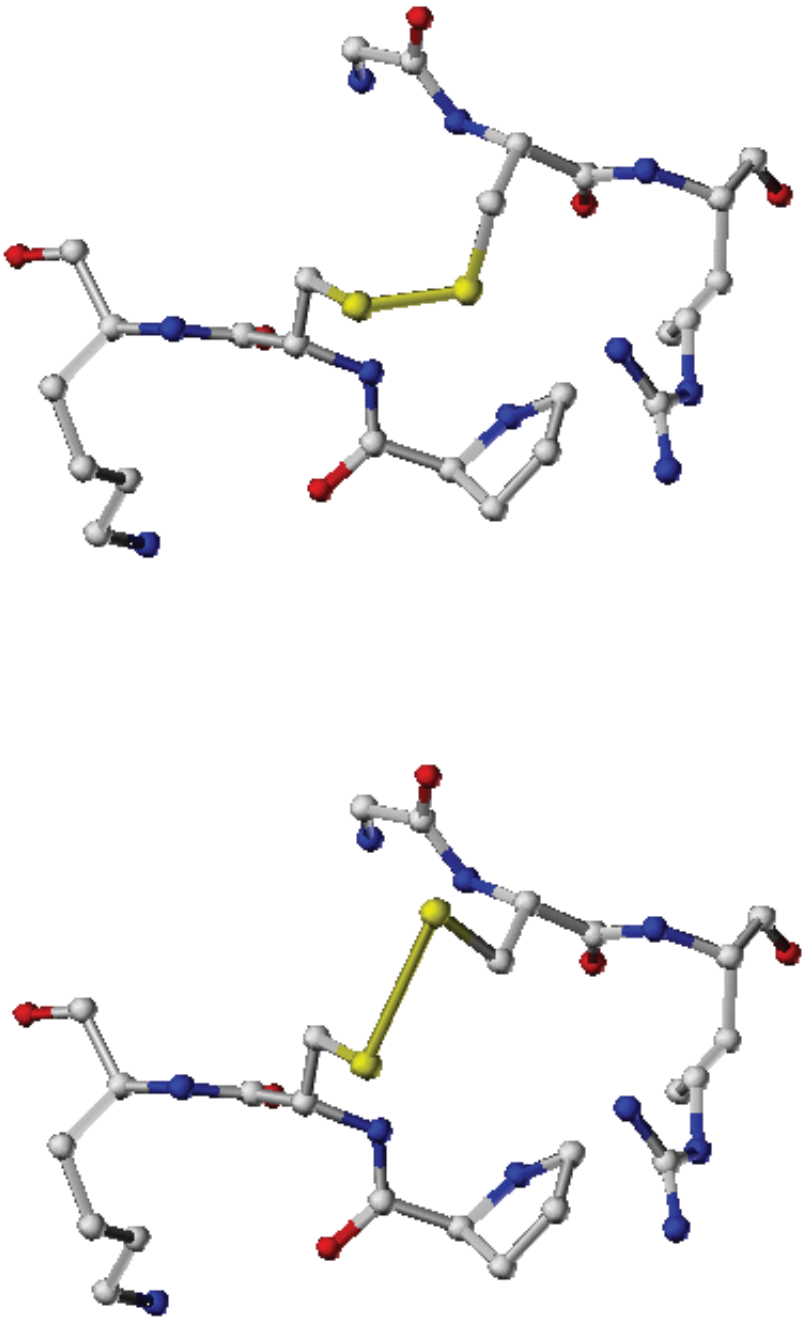


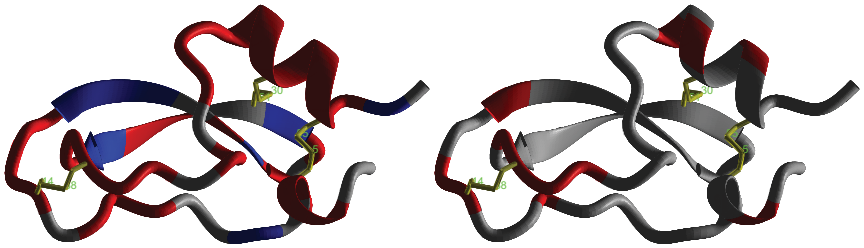
Figura 4.12: Conformaciones adoptadas por los puentes disulfuro en wt-BPTI (arriba) y $C30V, C51A$-BPTI (abajo).

Las regiones próximas al puente disulfuro desaparecido con la mutación son las que presentan mayores diferencias entre el wt-BPTI y el [C30V,-C51A]-BPTI. En concreto, se puede apreciar que los residuos en torno a la CYS31 del wt-BPTI poseen dinámicas de alta complejidad (modelo 5 para los que se obtiene solución o no se encuentra solución) mientras que los mismos residuos en el mutante presentan dinámicas simples de modelo 2 o modelo 3. Esto parece indicar la desaparición de la escala de tiempos lenta en torno al puente disulfuro. Se aprecia pues un incremento en la velocidad de movimiento en el mutante en la región donde antes estaba el puente disulfuro. Esta modificación no parece afectar en gran medida a la rigidez del enlace, contrariamente a lo que cabría esperar, ya que los valores de S^2 no varían excesivamente de una proteína a la otra. El intercambio químico se ve modificado en el mutante disminuyendo en la región de la CYS51 mutada (ALA51 en el mutante). En los residuos para los que se ha encontrado solución, el wt-BPTI tiene un intercambio químico mayor que el mutante. Sin embargo, esta disminución en el R_{ex} no se observa en la región de la CYS30 (VAL30 en el mutante), donde el intercambio químico no presenta grandes variaciones de una proteína a otra.

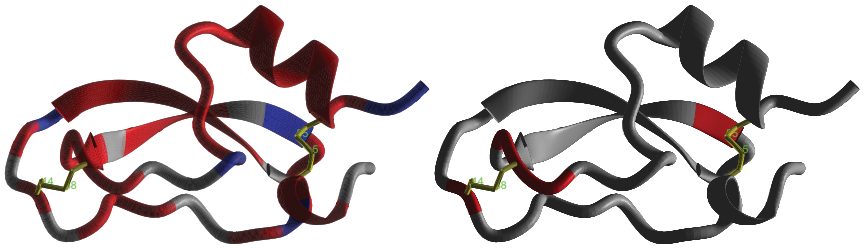
Las zonas terminales conservan bastante bien sus propiedades dinámicas. Se puede comprobar que el extremo C-terminal mantiene una dinámica compleja en ambos casos si bien en el caso del wt-BPTI, la ausencia de solución para la ALA58 parece indicar una dinámica más compleja con posible intercambio químico que en el mutante no se observa.

Algunas de las soluciones propuestas para determinados residuos del [C30V,C51A]-BPTI carecen de sentido físico. Tal es el caso de los residuos LEU6, ASP50 y ARG53 cuyos valores de τ_e son negativos aunque en el caso de los dos últimos el valor extremo es tan próximo a 0 que se podrían considerar residuos correspondientes al modelo 1. El caso de la LEU6 es bastante peculiar ya que en el wt-BPTI es uno de los residuos para los que no se encuentra solución ni siquiera explorando las regiones sin sentido físico. Es posible que en ambos casos, mutante y proteína nativa, este residuo se comporte según una dinámica complicada aunque de diferentes características. Las diferencias entre ambos residuos también se pueden observar en los modelos asignados a los residuos adyacentes.

La LEU6 se encuentra al final de una pequeña región α-helicoidal que comprende los residuos del 3 al 7. La LEU6, según la estructura de Rayos X del wt-BPTI, tiene su cadena lateral totalmente expuesta al disolvente en la cara de la proteína con menor número de elementos de estructura secun-



(a) wt-BPTI,S^2 (b) wt-BPTI,R_{ex}



(c) $C30V, C51A$-BPTI,S^2 (d) $C30V, C51A$-BPTI,R_{ex}

Figura 4.13: Representación tridimensional del mutante de BPTI y el BPTI nativo coloreada segun los valores de R_{ex} y el S^2. $S^2 \geq 0.8$, rojo. $S^2 \leq 0.8$, azul. $R_{ex} \geq 3.0$, rojo. $R_{ex} \leq 3.0$, azul.

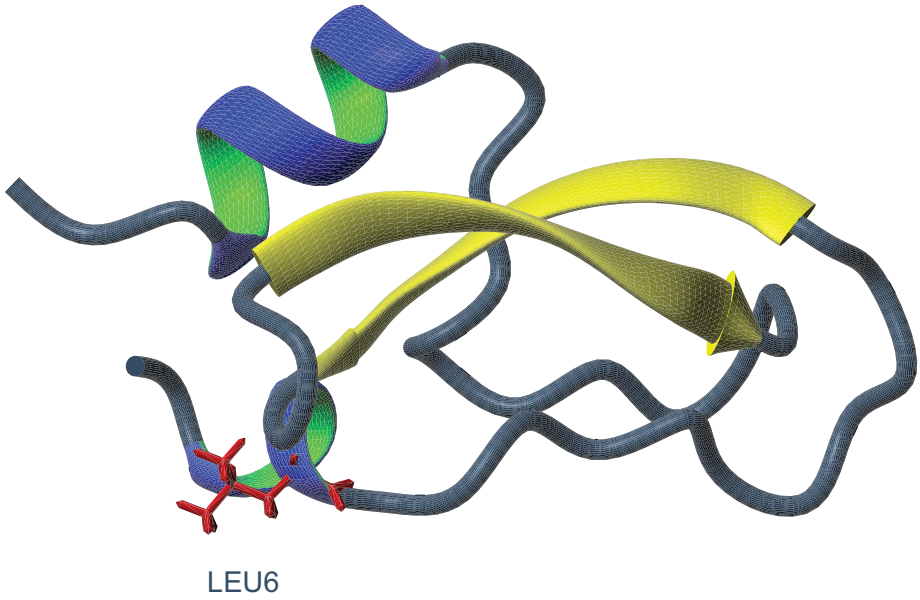


Figura 4.14: Representación tridimensional de la estructura obbtenida por Rayos X para el wt-BPTI indicando la posición del residuo LEU6 con su cadena lateral en verde.

daria definida (ver figura 4.14) y por lo tanto en la región que *a priori* a de resultar de mayor movilidad. Sin embargo, el hecho de que este en el único elemento de cierta estructuración en ese lado de la proteína combinado a estar expuesto al disolvente y en una zona relativamente dinámica puede contribuir a que este residuo (y en general, los que le rodean en la α-hélice) posean una dinámica relativamente compleja.

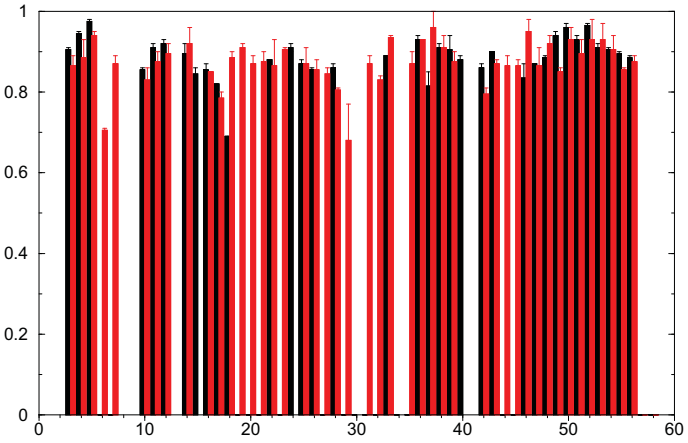


Figura 4.15: S^2 frente a secuencia para wt-BPTI (negro) y $C30V, C51A$-BPTI.

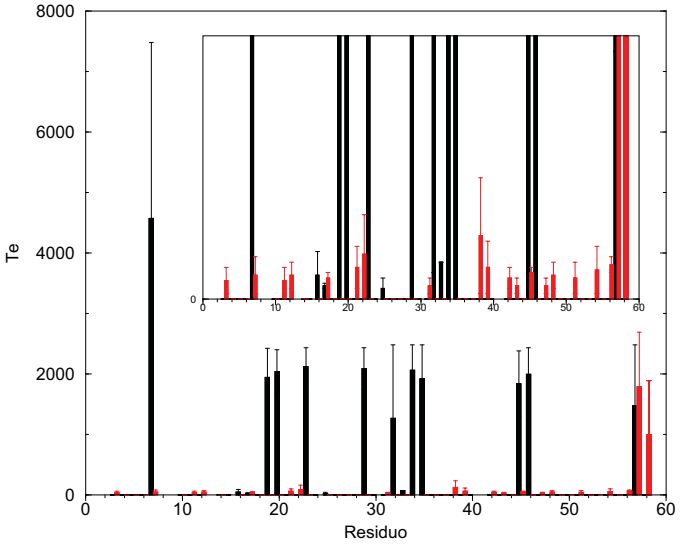


Figura 4.16: τ_e frente a secuencia para wt-BPTI (negro) y $C30V, C51A$-BPTI.

Un aspecto importante a resaltar es que las diferencias entre la dinámica de los residuos de ambas proteínas no se localizan exclusivamente en las

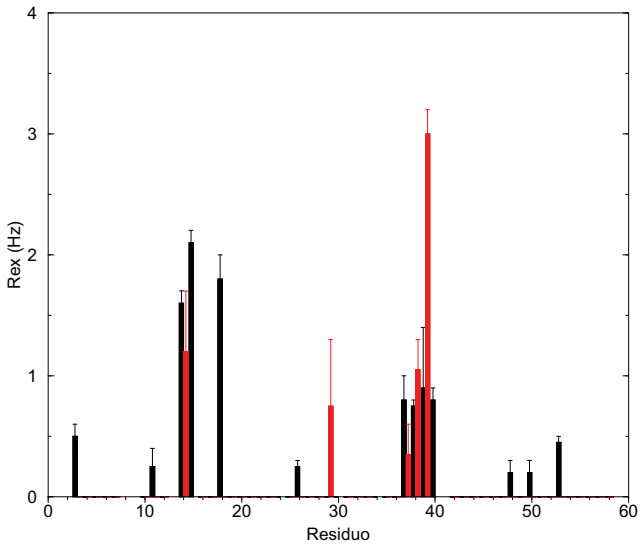


Figura 4.17: R_{ex} frente a secuencia para wt-BPTI (negro) y $C30V, C51A$-BPTI.

zonas de mutación. Aunque en estas zonas es donde se observan las mayores diferencias, también es posible observar variaciones en los parámetros dinámicos y en los modelos seleccionados para los residuos de las zonas correspondientes a los otros puentes disulfuros. En particular, la zona en torno a la CYS5 y la región entre la CYS14 y la mutación en la CYS30 se ven bastante afectadas como puede apreciarse en la figura 4.13. Esto puede deberse en parte a la contribución de los puentes disulfuro a los procesos de intercambio químico en las regiones circundantes a los mismos. Por otra parte, la desaparición de un puente disulfuro puede tensionar más los otros dos para compensar su desaparición. La posición en el espacio del puente disulfuro desaparecido y las conformaciones de los que se mantienen en ambas estructuras (ver figura 4.12) apoyan la idea de que se produce una reorganización en el reparto de tensiones de la molécula (y por lo tanto en sus modos normales de vibración) que puede producir un cambio general en la dinámica de la proteína sin necesidad de modificar su estructura.

Todas las variaciones producidas en las dinámicas internas por el efecto de la mutación repercuten en todos los procesos físicos que impliquen cierta movilidad como pueden ser los procesos de intercambio con el disolvente, solvatación, movimientos globales de la proteína e indirectamente en el comportamiento termodinámico.

Análisis de densidad espectral reducida

Una forma típica de contrastar el nuevo formalismo propuesto y el programa desarrollado es a través de su comparación con una aproximación diferente para el análisis de la dinámica molecular, específicamente el formalismo de densidad espectral reducida [90]. Mediante el análisis de densidad espectral reducida podemos distinguir entre los residuos con movimientos internos lentos y rápidos mediante la comparación de los valores de $J(0)$ de cada residuo con el promedio y su desviación estándar. En la figura 4.18 se puede comparar los valores del tiempo de correlación efectivo de cada residuo con la diferencia entre valores de $J(0)$.

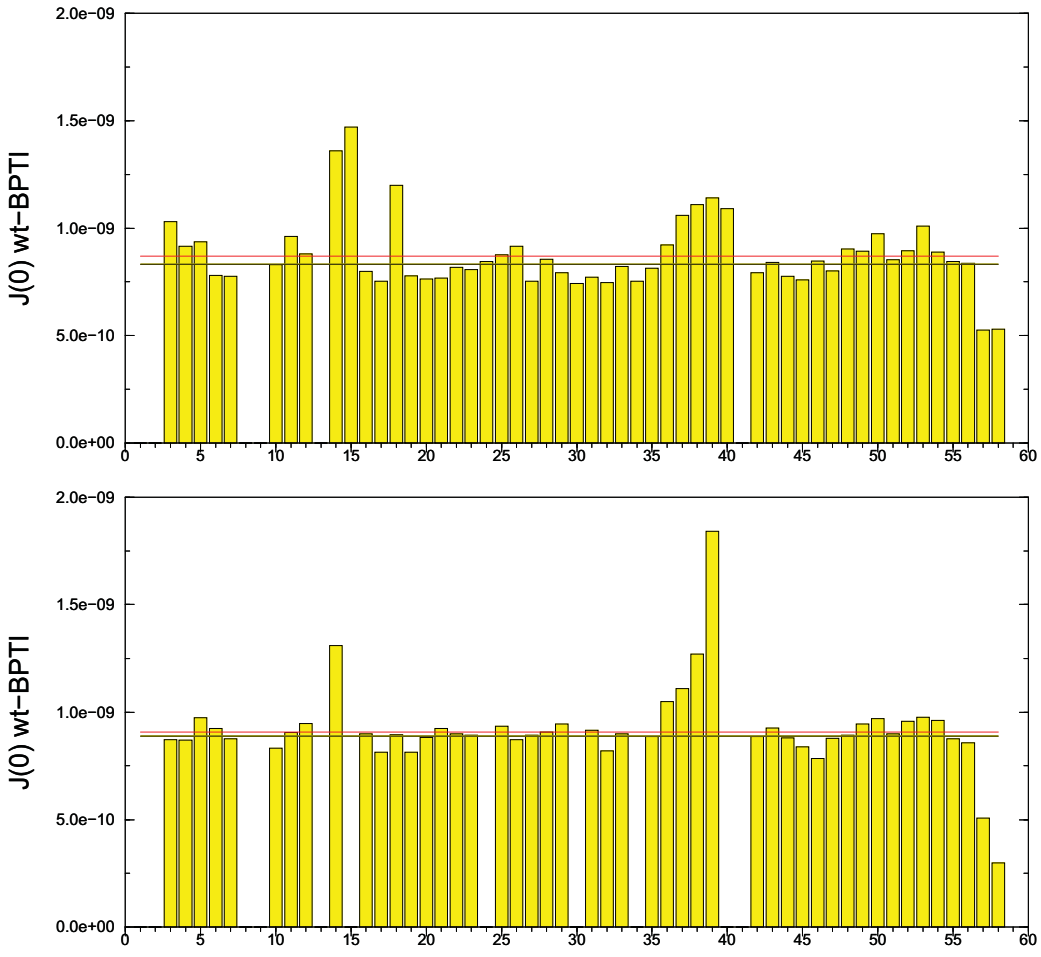


Figura 4.18: Representación de J(0) frente a la secuencia con su desviación estandar para el wt-BPTI y el [C30V,C51A]-BPTI.

Se puede observar una correlación bastante buena entre los residuos de

τ_e elevado con residuos lentos en el análisis de densidad espectral reducida en el wt-BPTI. Esta correlación se puede apreciar en la región en torno a la LEU6, en la región en torno a la ILE19 y sobre todo en la región de los residuos de 35 a 38. En la gráfica de las mismas características obtenida para el mutante [C30V,C51A]-BPTI es mucho más difícil apreciar dicha correlación debido a la mayor rapidez de los movimientos internos en esta proteína. Sin embargo, se puede observar que en ambos casos casi todos los residuos presentan mayor rapidez en los movimientos internos siendo la única excepción los residuos en torno a la CYS38 en el análisis de densidad espectral reducida y los residuos GLY57 y ALA58 en los resultados de SEARCHEL. En SEARCHEL, sin embargo, que los τ_e tabulados sean tan elevados no indican un movimiento interno lento. Al ser ambos residuos correspondientes a modelo 5, sólo se tabulan los τ_e correspondientes al movimiento lento ya que el τ_e del movimiento rápido es despreciable frente al primero (ver tabla 4.1). Sin embargo, la escala de tiempos utilizada en el análisis de densidad espectral reducida para indicar que residuos son lentos y que residuos son rápidos parece diferir de la escala absoluta utilizada en el formalismo de Lipari-Szabo. Por otra parte, los tiempos de correlación global calculados mediante el análisis de densidad espectral reducida son ligeramente más largos que los obtenidos utilizando el formalismo libre de modelo de Lipari-Szabo, tanto mediante el método tradicional de minimización como en el desarrollado en este trabajo.

4.4 Conclusiones

Se ha diseñado y programado un nuevo método de cálculo de los parámetros dinámicos de Lipari-Szabo utilizando tanto el formalismo libre de modelo como su ampliación con intercambio químico y varias escalas de tiempo.

Los resultados obtenidos utilizando el programa SEARCHEL tanto con datos simulados como con datos bibliográficos indican que tanto el algoritmo utilizado para el desarrollo del método como su traducción en un programa de cálculo eficiente proporcionan no sólo soluciones correctas sino que aportan nueva información desde el punto de vista dinámico a la interpretación de parámetros de relajación. La coincidencia con los análisis realizados sobre el mismo formalismo con otros métodos dan consistencia a la interpretación del formalismo aplicada en el método de búsqueda.

Por otra parte, la comparación de los resultados con análisis realizados utilizando otras aproximaciones teóricas como pueda ser el análisis de

densidad espectral reducida nos indica que tanto el método como el programa están preparados para ser utilizados exhaustivamente en el análisis e interpretación de medidas experimentales de relajación.

La contribución realizada en la interpretación de la forma y tamaño de la incertidumbre en los parámetros dinámicos de Lipari-Szabo permite orientar el diseño de nuevos experimentos con el objetivo de reducir en la medida de lo posible la imprecisión de los resultados obtenidos.

La automatización del proceso de búsqueda utilizando unos parámetros optimizados para varios ejemplos tanto simulados como reales de datos de relajación elimina la subjetividad que puede tener un método en que la elección del modelo y del valor de τ_m depende del usuario. Sin embargo, el análisis exhaustivo de las soluciones para cada uno de los modelos es posible y el usuario puede deshechar la solución propuesta por el programa y hacer su propio análisis.

La dinámica del wt-BPTI se ve afectada por las mutaciones C30V y C51A en toda su secuencia como consecuencia de la reorganización de las tensiones intramoleculares y de los modos normales de vibración. Se puede verificar dicha reorganización comprobando que las mayores variaciones se producen alrededor de las zonas de puente disulfuro.

Los cambios en la dinámica intramolecular del wt-BPTI se traducen en variaciones en las dinámicas individuales de varios residuos y en una variación en el tiempo de correlación global de la proteína τ_m. Se puede apreciar una variación en el intercambio químico en general en toda la molécula y en particular un aumento del R_{ex} localizado en los alrededores del puente disulfuro desaparecido.

El aumento del valor del tiempo de correlación global de la proteína con la mutación puede deberse a que la mayor velocidad de los movimientos internos de los residuos produzca una aumento en el equilibrio de intercambio de moléculas de solvatación haciendo que aumente el tamaño aparente de la proteína (tamaño real más primera capa de solvatación). El principal efecto de este aumento en el tamaño es una disminución en la velocidad de movimiento global de la molécula y, por lo tanto, un aumento del tiempo de correlación global, τ_m.

Los valores de tiempo de correlación efectivo para la mayoría de residuos es más corto en el caso del [C30V,C51V]-BPTI que en el wt-BPTI lo cual parece indicar una mayor velocidad de movimiento en el primero. La aparición de dinámicas de dos escalas en algunos residuos del wt-BPTI

indican mayor complejidad tanto en el comportamiento dinámico de los mismos como en los procesos de relajación implicados.

Capítulo 5

Refinamiento de estructuras de RMN mediante restricciones de ángulos diedros. Aportación al programa HYPER.

5.1 Introducción.

En el cálculo de estructuras tridimensionales de biopolímeros, la precisión de las restricciones estructurales a nivel atómico resulta fundamental para conseguir estructuras de calidad. Aunque las restricciones de distancia son muy importantes para determinar el plegamiento global de las proteínas y las conformaciones locales de los residuos, no son tan limitantes como puedan ser las restricciones de ángulos diedros. Una restricción de ángulo diedro delimita de modo mucho más efectivo el espacio conformacional accesible que un grupo de varias restricciones de distancia de precisión similar. De este modo la convergencia de las estructuras finales se ve mejorada. Además, en los cálculos basados en geometría de distancias, donde los únicos grados de libertad son los ángulos diedros, los cálculos se aceleran en gran medida. Entre los objetivos de esta tesis el perfeccionamiento de la metodología para la determinación estructural de biomoléculas es uno de los más importantes. Por consiguiente, se ha desarrollado un método para convertir la información contenida en restricciones de distancia y constantes de acoplamiento en restricciones de ángulos diedros efectivas tanto para la cadena principal como para las cadenas laterales.

5.1.1 Restricciones estructurales

Los cálculos de estructuras de proteínas mediante datos de RMN se ve mejorado significativamente en calidad y velocidad mediante la adición de asignaciones estereoespecíficas de los centros proquirales de la proteína y mediante la adición de restricciones de ángulo diedro derivadas de las medidas de conectividades NOE y de las constantes de acoplamiento escalar ${}^3J$ [112]. Las restricciones de distancia extraidas de las conectividades del espectro NOESY permiten deducir en qué intervalo se encuentran ciertos ángulos diedros característicos de una estructura de proteína del mismo modo en que permiten identificar ciertos patrones de estructura secundaria (ver capítulo 3). Las restricciones de ángulo diedro, además de limitar el espacio conformacional accesible que es necesario explorar en la búsqueda de la estructura que satisfaga todas las restricciones estructurales, mejora la eficacia de los métodos de generación de estructuras incrementando los niveles de convergencia [113]. Por ejemplo, se ve incrementado el número de conformaciones iniciales que resultan en estructuras finales safisfaciendo todas las restricciones experimentales. La utilidad de esas restricciones de ángulos diedros se ha hecho evidente en el gran esfuerzo que se está realizando en los ultimos años para desarrollar métodos mejorados para determinar constantes de acoplamiento escalar homo y heteronuclear en proteínas [114, 115]. Sin embargo, el desarrollo de estas metodologías debe ser apoyado de modo paralelo por la mejora de los métodos de interpretación de dichas constantes de acoplamiento escalar y de los datos NOE en términos de restricciones estructurales efectivas.

Diferentes aproximaciones se han realizado sobre este aspecto mediante la combinación de la información de distancias para extraer restricciones de ángulos diedros. Las conectividades NOE intraresiduo se utilizan para evaluar los ángulos diedros relacionados con las mismas (ϕ, ψ y χ_1) de modo indirecto. Es decir, se ensayan conjuntos de valores de ángulos diedros hasta encontrar uno que sea compatible con las restricciones de distancia impuestas. Al reducir el alcance de las variables a un nivel local, la velocidad de evaluación de los diedros se ve incrementada respecto a un cálculo global de estructuras. Sin embargo, el uso habitual de expresiones analíticas para relacionar las distancias y los ángulos diedros produce cierta rigidez en los programas desarollados con tal fin, al tiempo que introduce, en cierto modo, un error sistemático difícil de evaluar.

El cálculo de distancias interatómicas mediante algebra matricial en un sistema en que los únicos grados de libertad son los ángulos diedros ha sido

descrito profusamente en numerosos tratados sobre biopolímeros [116, 117]. El problema se reduce a una transformación de coordenadas de un sistema de referencia local a un sistema global.

Suponiendo que cada enlace atómico es un vector al que llamaremos enlace-vector, las componentes de cada enlace-vector se pueden expresar de un modo sencillo tomando como sistema de referencia un triedro cartesiano con el eje x coincidente con la dirección del vector. De este modo, el enlace-vector será, en dicho sistema de referencia $(l, 0, 0)$. En esta situación, cada enlace-vector tendrá esta expresión dentro de su sistema de referencia local. La distancia entre dos átomos cualesquiera del sistema se puede evaluar como el módulo del vector que une ambos átomos. Este vector se puede calcular como suma de los enlace-vectores que van de un átomo al otro. Sin embargo, para poder realizar esta suma es necesario que todos los enlace-vectores estén definidos en el mismo sistema de referencia. Mediante transformaciones de coordenadas relativamente simples se pueden convertir todos los enlace-vectores implicados a un único sistema de referencia.

Para convertir un enlace-vector de su sistema de referencia al sistema de referencia del enlace-vector inmediatamente anterior basta un par de rotaciones según el ángulo de enlace y el ángulo diedro implicados por ambos enlace-vectores. Para determinar exactamente las matrices de transformación asociadas a las operaciones descritas es necesario conocer el sistema de referencia local de cada enlace-vector. Este sistema se define tomando el eje x coincidente con la dirección del enlace-vector. El eje y se hace coincidente con el plano definido por el eje x y el enlace-vector inmediatamente anterior, de modo que la proyección del eje y sobre dicho enlace-vector sea positiva. El eje z queda pues definido por los ejes x y y en un sistema de referencia ortonormal (ver figura 5.1).

Tomando como referencia del ángulo diedro η la posición trans ($\eta = 0$) la matriz de transformación global tiene la siguiente expresión:

$$\mathbf{X} = \begin{pmatrix} cos(\pi - \theta) & sen(\pi - \theta) & 0 \\ sen(\pi - \theta)cos(\eta) & -cos(\pi - \theta)cos(\eta) & sen(\eta) \\ sen(\pi - \theta)sen(\eta) & -cos(\pi - \theta)sen(\eta) & -cos(\eta) \end{pmatrix} \qquad (5.1)$$

donde η es el ángulo diedro que implica a los dos enlace-vectores y θ es el ángulo de enlace entre ambos enlace-vectores. Según el criterio utilizado como referencia en los ángulos la expresión puede variar ligeramente de

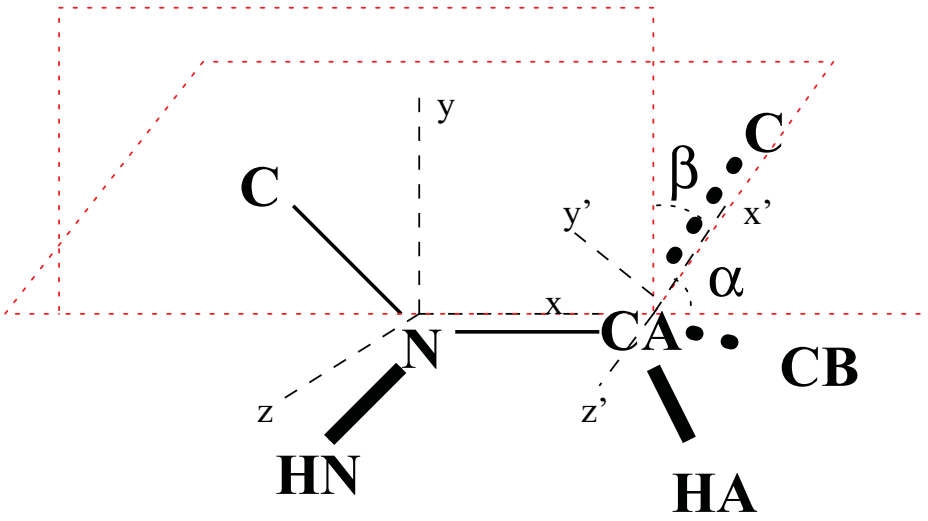


Figura 5.1: Sistema de coordenadas general utilizado en la determinación de las distancias en función de ϕ, ψ y χ_1.

unos sistemas a otros.

Mediante la aplicación de la matriz sucesivamente se puede transformar cualquier enlace-vector de una molécula al sistema de referencia de otro enlace-vector cualquiera en la molécula. Así, cualquier distancia interatómica podría ser calculada como función de los ángulos de enlace y ángulos diedros intermedios.

La aplicación de estas operaciones a una proteína es inmediata y permite evaluar las distancias interprotónicas derivadas de las conectividades NOE como función de los diedros principales de la proteína ayudando a la determinación de restricciones de ángulos diedros.

5.1.2 Asignación estereoespecífica

La asignación estereoespecífica de los centros proquirales de una proteína consiste en la diferenciación entre los protones β_2 y β_3 de dichos centros proquirales. Esta diferenciación evita tener que usar pseudoátomos en lugar de los propios protones metilénicos. El uso de los pseudoátomos en los cálculos de estructuras debilita las restricciones experimentales de distancia debido a la adición de un radio adicional a la distancia para compensar la incertidumbre inherente al pseudoátomo. La diferenciación de los protones metilénicos conlleva asociada la determinación del ángulo diedro χ_1

que los relaciona con la cadena principal.

Se han descrito diferentes estrategias y programas de ordenador para la determinación de las asignaciones estereoespecíficas de los protones metilénicos y los ángulos diedros asociados a la conformación de las cadenas laterales respecto a la cadena principal a partir de datos experimentales de RMN [118, 119, 120]. También se han descrito aproximaciones en que dicha asignación estereoespecífica se realiza mediante datos estadísticos extraídos de bases de datos de estructuras de proteínas [121, 122]. La mayoría de estas aproximaciones se basan en el análisis combinado de las distancias locales y las constantes de acoplamiento escalar ${}^3J$. Una idea muy utilizada es el uso de búsquedas en rejilla en el espacio de ángulos diedros con respecto a las restricciones internucleares para el análisis de los datos de NOE de protón. Esta aproximación es inherente a cualquier estudio inicial de las relaciones entre los ángulos diedros y las distancias intraresiduo y secuenciales derivadas experimentalmente que constituyen la base de los métodos convencionales de asignación de resonancias descritas en el capítulo 3.

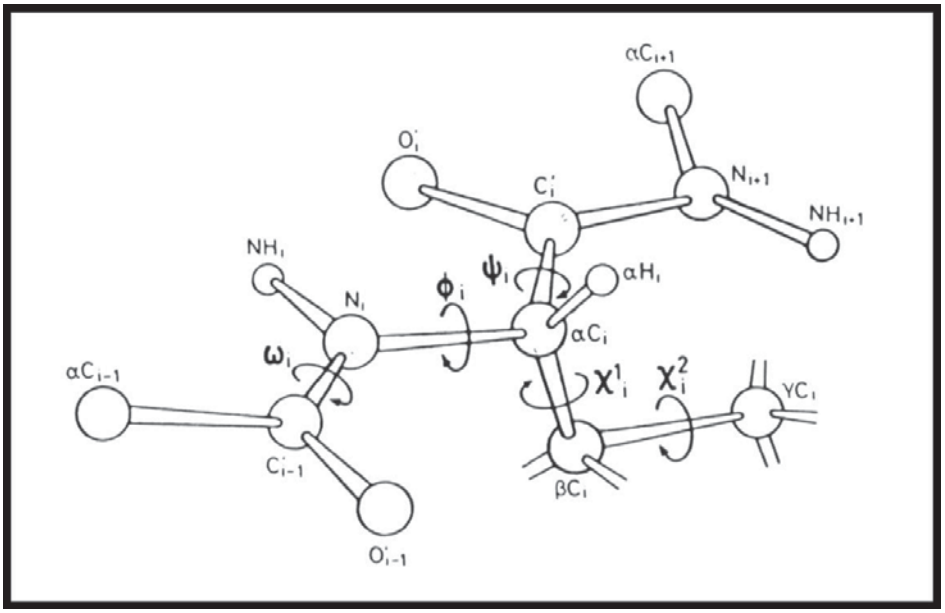


Figura 5.2: Representación tridimensional de un residuo con los ángulos carácteristicos.

Un método cualitativo de gran interés para la asignación estereoespecífica de los protones metilénicos β es el análisis y comparación de las constantes de acoplamiento ${}^3J_{\alpha\beta}$ y ${}^3J_{N\beta}$ que son función de χ_1 exclusivamente, junto a la comparación de los NOEs intraresiduo y secuenciales $H_{Ni} - H_{\beta i}$, $H_{\alpha i} - H_{\beta i}$ y $H_{\beta i} - H_{Ni+1}$, que son función de ϕ, ψ y χ_1. El NOE

intraresiduo $H_{\alpha i} - H_{\beta i}$ es función exclusivamente de χ_1 por lo que es el más indicado para restringir los valores de χ_1.

5.2 Métodos.

5.2.1 Cálculo de distancias

Considerando el segmento de cadena polipeptídica

$$NH - C^{\alpha}H - C^{\beta}H_2 - C' - NH_{i+1}$$

mostrado en la figura 5.3, se puede observar que las posiciones de los cinco átomos de hidrógeno (H_N, H_α, $H_{\beta 2}$, $H_{\beta 3}$ y $H_{N\,i+1}$) están relacionadas por 10 ($5*4/2$) distancias interprotónicas. Una de estas distancias está fijada por la geometría covalente (la distancia entre los protones geminales $H_{\beta 2}$ y $H_{\beta 3}$) y no es relevante para el análisis a realizar. Las otras nueve distancias están determinadas por los valores de los cuatro diedros que intervienen en el esquema, ϕ, ψ, ω y χ_1. Asumiendo que el ángulo diedro ω está fijado a valores de 0^o (confomación cis) o de 180^o (conformación trans), los otros tres ángulos diedros son los únicos grados de libertad a considerar y están determinados por las nueve distancias experimentales.

Estas distancias son las que se pueden observar en la figura 5.3 y se encuentran tabuladas en la tabla 5.1 junto a la dependencia de los ángulos diedros correspondientes. Aunque se han descrito varias relaciones trigonométricas entre algunas de estas distancias y los correspondientes ángulos diedros para geometrías covalentes específicas, prácticamente en su totalidad son empíricas. Con el fin de disponer de un conjunto analítico de dichas relaciones se ha implementado un método originalmente desarrollado en los tratados de mecánica estadística de polímeros [117] para calcular el subconjunto de distancias requeridas. Se ha tomado esta aproximación en primer lugar, por la mayor adaptabilidad y precisión del método a diferentes geometrías covalentes, y en segundo lugar por la posible extensión del método a un espacio de más dimensiones. Este último aspecto hace al método fácilmente ampliable a analizar distancias entre protones separados más de un residuo, es decir, más de tres o cuatro ángulos diedros principales. Además, el hecho de que el cálculo de distancias se haga de modo algebráico permite incluir diferentes valores para las distancias de enlace y ángulos de enlace de modo paramétrico.

La distancia entre el protón H^a y el protón H^b se calcula como el módulo

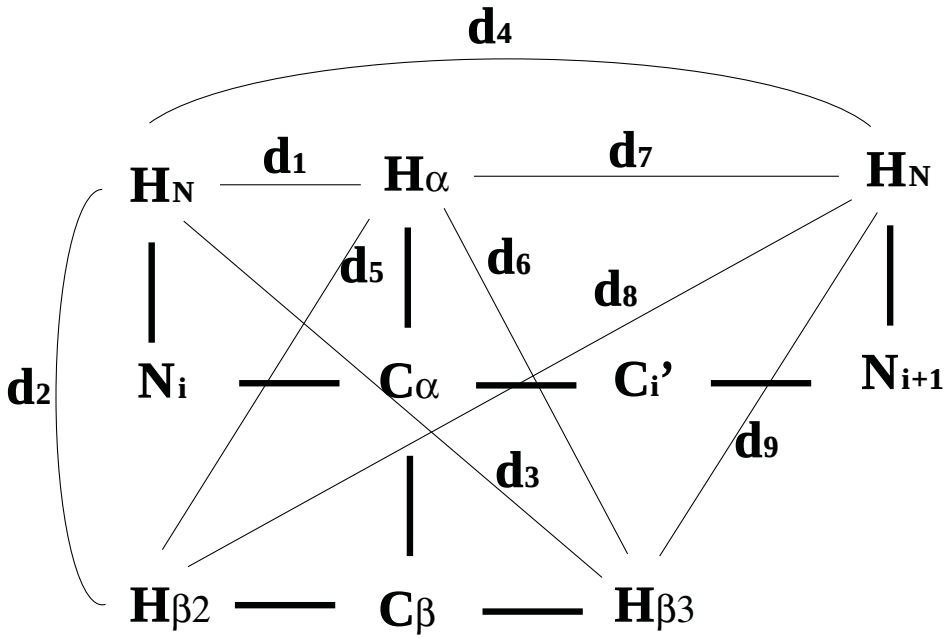


Figura 5.3: Visualización de las distancias utilizadas por el programa HYPER para la búsqueda de ángulos diedros.

Distancia	Diedros	Clasificación
d1	ϕ	fuerte
d2	ϕ,χ_1	débil
d3	ϕ,χ_1	débil
d4	ϕ,χ_1	débil
d5	χ_1	fuerte
d6	χ_1	fuerte
d7	ψ	fuerte
d8	ψ,χ_1	débil
d9	ψ,χ_1	débil

Tabla 5.1: Dependencia de las distancias calculadas respecto a los ángulos diedros implicados y clasificación en poder de restricción del espacio accesible.

del vector que conecta ambos protones. Este vector se construye mediante adición de todos los enlace-vectores definidos por los enlaces covalentes en la trayectoria de un protón al otro. El uso de este método requiere que todos los enlace-vectores estén expresados en el mismo sistema de coordenadas de referencia (el sistema de coordenadas de referencia global). El sistema de coordenadas de referencia es elegido de modo que todas las transformaciones requeridas sean tan simples e intuitivas como sea posible. En el criterio utilizado en este estudio, el sistema de coordenadas de referencia global (mostrado en la figura 5.1) es aquel en que el eje x, **i**, está alineado con el enlace-vector $N - C^{\alpha}$ de modo que

$$v_{N-CA} = (l_{N-CA}, 0, 0) \tag{5.2}$$

donde l_{N-CA} es la longitud del enlace $N - C^{\alpha}$. El eje y, **j**, es escogido de modo que se encuentre en el plano definido por los enlace-vectores v_{N-CA} y v_{HN-N}, perpendicular al enlace-vector v_{N-CA} y de modo que su proyección sobre el enlace-vector v_{HN-N} sea positiva. Finalmente, el eje z, **k**, se construye ortonormalmente a **i** y **j** (**k** = **i**x**j**). Una vez este sistema de coordenadas de referencia general ha sido definido se necesita únicamente transformar todos los enlace-vectores en la trayectoria de H^a a H^b a este sistema de coordenadas y sumarlos para definir el vector $v_{H^a-H^b}$ entre los átomos H^a y H^b.

Inicialmente, cada enlace-vector se expresa en su propio sistema de coordenadas (por ejemplo, x', y' y z' para el enlace-vector $v_{CA-C'}$ en la figura 5.1). Los sistemas de coordenadas de cada enlace-vector se transforman en el sistema de coordenadas de referencia del enlace-vector precedente mediante matrices de rotación estándar. Estas transformaciones generalmente sólo requieren dos rotaciones consecutivas, la primera de las cuales sucede sobre el eje z y la segunda sobre el eje x (rotaciones α y β en la figura 5.1) De los dos ángulos implicados en la transformación, uno es el suplementario del ángulo de enlace que liga ambos enlace-vectores (α en la figura 5.1) y el otro es el correspondiente ángulo diedro que puede ser fijo en el caso de depender de ω o rotable en el caso de depender de ϕ, ψ y χ_1 (β en la figura 5.1). Este ángulo diedro está relacionado, según los enlace-vectores específicos y las distancias interprotónicas implicadas, con los valores de ϕ, ψ, ω o χ_1 tal como han sido definidos por la convención estándar de la IUPAC [123].

Este proceso se aplica de modo recursivo hasta que el enlace-vector deseado es expresado en el sistema de coordenadas de referencia gene-

ral. Usando esta aproximación básica, los algoritmos programados para el programa HYPER, calculan rápidamente la dependencia de las distancias mostradas en la figura 5.3 de los ángulos diedros ϕ, ψ y χ_1 (y ω en caso de que sea necesario) para los valores definidos de distancias de enlace y ángulos de enlace proporcionados por el usuario. Los valores por defecto de longitudes de enlace y ángulos de enlace son los definidos en el campo de fuerzas AMBER [124, 125].

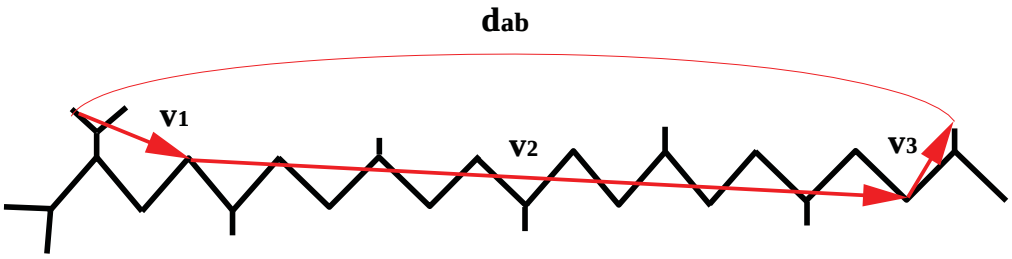


Figura 5.4: Esquema del método de cálculo implementado para distancias entre átomos separando N residuos. La distancia d_{ab} es el módulo del vector suma de v_1, v_2 y v_3.

Aunque en estos momentos el programa accesible al usuario HYPER [175] no incluye la extensión del cálculo de distancias a más de 2 residuos, el método de cálculo ya ha sido implementado y comprobado. Para extender el método de cálculo de distancias a un número de residuos genérico sólo es necesario establecer un sistema de coordenadas de referencia global para toda la molécula y las transformaciones necesarias para convertir todos los enlace-vectores a dicho sistema. El sistema de coordenadas de referencia general del primer residuo se escoge como sistema de coordenadas de referencia global para toda la molécula. Las transformaciones correspondientes se pueden realizar del mismo modo en que se realizan a nivel local, aunque se ha escogido un método más rápido para evitar realizar un número elevado de productos de matrices en cada cálculo de distancias. En una primera etapa de cálculo se transforman todos los enlace-vectores de la cadena principal al sistema de referencia global y se definen unos residuo-vectores en el sistema de referencia global que unen los átomos de N de residuos consecutivos. De este modo, para calcular la distancia entre dos átomos separados N residuos, únicamente hay que sumar los vectores del átomo a hasta el N del residuo siguiente, los residuo-vectores entre los dos residuos y el vector del N del residuo del átomo b hasta el átomo b (ver

figura 5.4).

Mediante búsqueda en rejilla jerarquizada, el programa HYPER explora el espacio de ángulos diedros calculando las distancias interprotónicas mediante el algebra matricial y las constantes de acoplamiento escalar mediante las ecuaciones de Karplus, y comparando los valores obtenidos con los experimentales de modo que el resultado final sea un intervalo de valores de ángulos diedros compatible con los datos experimentales. En la búsqueda se distinguen entre restricciones de carácter fuerte y débiles según su carácter restrictivo y se aplican en primer lugar las de carácter fuerte para reducir el espacio a explorar en las siguientes etapas. Las restricciones de carácter débil sirven para refinar y estrechar el intervalo de diedros obtenido en las primeras etapas.

5.2.2 Asignación estereoespecífica

La asignación estereoespecífica de los protones metilénicos β se puede realizar mediante el análisis y comparación de las constantes de acoplamientos ${}^3J_{\alpha\beta}$, unido a la comparación del NOE intraresiduo $H_\alpha - H_\beta$. En la tabla 5.2 se pueden observar los valores típicos de distancias $H_\alpha - H_\beta$ y la relación entre los valores de ${}^3J_{\alpha\beta}$ para las conformaciones más favorables de cadena lateral. Mediante la comparación de estos valores con los determinados mediante análisis de los datos experimentales de RMN se puede pues determinar el valor de χ_1 así como la asignación estereoespecífica de los protones metilénicos.

Sin embargo, únicamente con los valores de constantes de acoplamiento escalar y de conectividades NOE entre los protones α y β no es posible determinar unívocamente la asignación estereoespecífica de los protones metilénicos. En algunos casos es necesario introducir como información adicional, las constantes de acoplamiento ${}^3J_{N\beta}$ o los NOEs intraresiduo $H_{Ni} - H_{\beta i}$ que son función de ϕ, ψ y χ_1 para resolver posibles ambigüedades en la asignación estereoespecífica. La principal dificultad de considerar estas constantes de acoplamiento y conectividades NOE es que dependen también de los valores de ángulos diedros de la cadena principal. En la tabla 5.3 se pueden observar los valores de ${}^3J_{N\beta}$ y de distancias $H_N - H_\beta$ para diferentes valores de ϕ y ψ característicos de elementos habituales de estructura secundaria y para los valores de χ_1 más favorables.

De este modo, mediante el análisis cualitativo de las constantes de acoplamiento escalar y las distancias interprotónicas que impliquen a los pro-

Distancias (Å)			
	$\chi_1 = -60$	$\chi_1 = 180$	$\chi_1 = 60$
d($H_\alpha - H_{\beta 2}$)	3.06	2.45	2.48
d($H_\alpha - H_{\beta 3}$)	2.24	2.89	2.82
Constantes de Acoplamiento (Hz)			
$^3J_{\alpha\beta 2}$	12.40	3.25	3.25
$^3J_{\alpha\beta 2}$	3.25	12.40	3.25
$^3J_{\alpha\beta 2}$ / $^3J_{\alpha\beta 2}$	3.82	0.26	1.00
$^3J_{\alpha\beta 2}$ / $^3J_{\alpha\beta 2}$	0.26	3.82	1.00
$^3J_{N15\beta 2}$	-0.40	-0.40	-5.50
$^3J_{N15\beta 3}$	-5.50	-0.40	-0.40

Tabla 5.2: Distancias y constantes de acoplamiento $^3J_{\alpha\beta}$ para las conformaciones de cadena lateral más favorecidas. Las constantes de acoplamiento han sido calculadas según las ecuaciones de Karplus correspondientes.

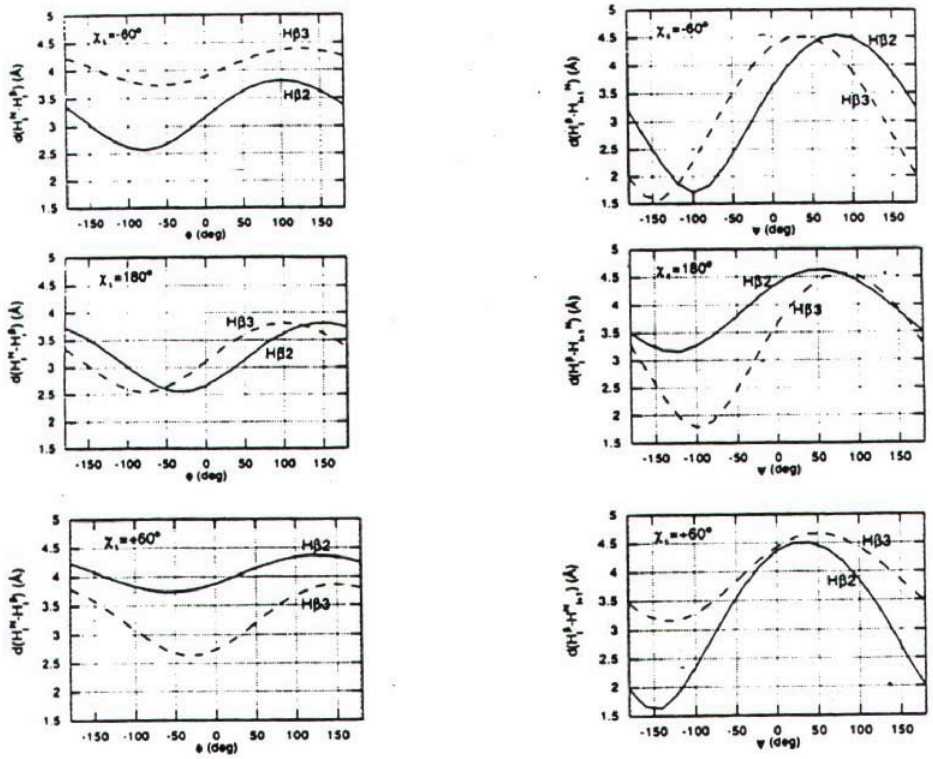


Figura 5.5: Gráficas de las funciones d($H_{Ni} - H_{\beta i}$)=f(ϕ,χ_1) y d($H_{\beta i} - H_{Ni+1}$)=f(ψ,χ_1) para una estructura modelo de polipéptido en función de los tres rotámeros rígidos, $\chi_1 = -60^o$, χ_1=180^o y χ_1=60^o.

	ϕ,ψ	$\chi_1 = -60$	$\chi_1 = 180$	$\chi_1 = 60$
	α-hélice			
d($H_{Ni} - H_{\beta 2i}$)	-40,-20	2.63	2.34	3.53
d($H_{Ni} - H_{\beta 3i}$)	-40,-20	3.51	2.85	2.10
d($H_{Ni+1} - H_{\beta 2i}$)	-40,-20	3.12	4.18	4.11
d($H_{Ni+1} - H_{\beta 3i}$)	-40,-20	4.11	3.12	4.18
d($H_{Ni} - H_{\beta 2i}$)	-64,-41	2.48	2.46	3.53
d($H_{Ni} - H_{\beta 3i}$)	-64,-41	3.50	2.84	2.07
d($H_{Ni+1} - H_{\beta 2i}$)	-64,-41	2.61	3.91	3.73
d($H_{Ni+1} - H_{\beta 3i}$)	-64,-41	3.73	2.61	3.91
d($H_{Ni} - H_{\beta 2i}$)	-80,-60	2.45	2.59	3.56
d($H_{Ni} - H_{\beta 3i}$)	-80,-60	3.53	2.87	2.17
d($H_{Ni+1} - H_{\beta 2i}$)	-80,-60	2.18	3.67	3.32
d($H_{Ni+1} - H_{\beta 3i}$)	-80,-60	3.32	2.18	3.67
d($H_{Ni} - H_{\beta 2i}$)	-140,-100	2.81	3.24	3.86
d($H_{Ni} - H_{\beta 3i}$)	-140,-100	3.84	3.12	2.96
d($H_{Ni+1} - H_{\beta 2i}$)	-140,-100	1.76	3.30	2.36
d($H_{Ni+1} - H_{\beta 3i}$)	-140,-100	2.36	1.76	3.30
	hoja β			
d($H_{Ni} - H_{\beta 2i}$)	-40,60	2.63	2.34	3.53
d($H_{Ni} - H_{\beta 3i}$)	-40,60	3.51	2.85	2.10
d($H_{Ni+1} - H_{\beta 2i}$)	-40,60	4.47	4.73	4.47
d($H_{Ni+1} - H_{\beta 3i}$)	-40,60	4.47	4.47	4.73
d($H_{Ni} - H_{\beta 2i}$)	-120,120	2.62	3.03	3.75
d($H_{Ni} - H_{\beta 3i}$)	-120,120	3.72	3.02	2.68
d($H_{Ni+1} - H_{\beta 2i}$)	-120,120	4.37	4.41	3.58
d($H_{Ni+1} - H_{\beta 3i}$)	-120,120	3.58	4.37	4.41
d($H_{Ni} - H_{\beta 2i}$)	-180,180	3.24	3.55	4.07
d($H_{Ni} - H_{\beta 3i}$)	-180,180	4.07	3.30	3.50
d($H_{Ni+1} - H_{\beta 2i}$)	-180,180	3.32	3.32	2.18
d($H_{Ni+1} - H_{\beta 3i}$)	-180,180	2.18	3.67	3.67

Tabla 5.3: Distancias en elementos de estructura secundaria para las conformaciones de cadena lateral más favorecidas.

tones β se puede realizar la asignación estereoespecífica de los protones metilénicos y la determinación del ángulo diedro χ_1. Mediante el programa HYPER se pueden obtener estas asignaciones utilizando las expresiones de Karplus para el cálculo de las constantes de acoplamiento escalar correspondientes y el algebra matricial ya desarrollada para la determinación de distancias.

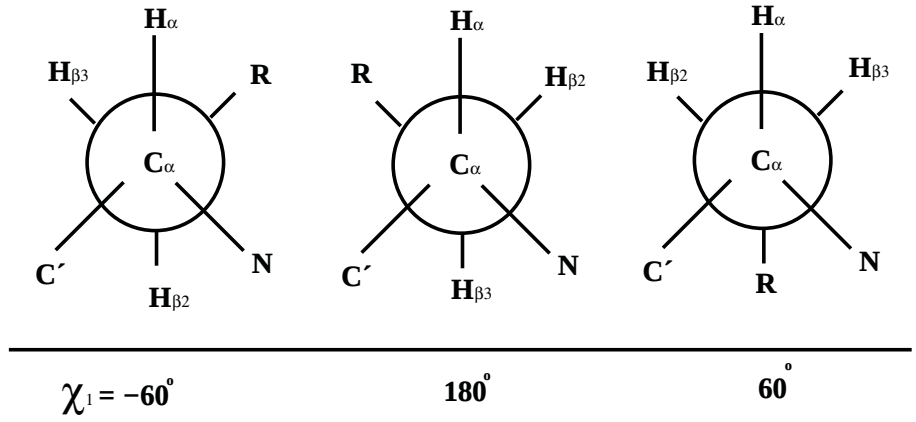


Figura 5.6: Proyección de Newman mostrando las posibles conformaciones favorables e indicando las relaciones cualitativas entre los NOEs y las constantes de acoplamiento escalar.

El programa HYPER supone *a priori* que la asignación estereoespecífica no ha sido realizada. Como punto de partida se supone que el protón metilénico con resonancia a campo magnético más alto corresponde a $H_{\beta 2}$ y el de resonancia a campo magnético más bajo a $H_{\beta 3}$ y se realiza la búsqueda en rejilla de los ángulos diedros correspondientes. A continuación, se intercambian las posiciones de los protones metilénicos y se realiza de nuevo la búsqueda de ángulos diedros. La posición de protones β que sea consistente con alguna solución de ángulos diedros se considera la asignación estereoespecífica correcta. En caso de que ambas posiciones den soluciones consistentes se considera que la información experimental no es suficiente para determinar unívocamente la asignación estereoespecífica y se considera como solución la unión de soluciones para cada uno de ellos. En caso de que ninguna posición dé solución consistente, los datos experimentales se consideran inconsistentes.

5.3 Resultados y discusión

5.3.1 Aplicación de la asignación estereoespecífica a dos proteínas: m-EGF y Dominio Z de la proteína A.

Como ya hemos visto, la aplicación de la información contenida en las constantes de acoplamiento vecinal ${}^3J_{\alpha\beta}$ y las distancias intraresiduo $H_\alpha - H_\beta$ y $H_N - H_\beta$ permite realizar la asignación estereoespecífica de los residuos metilénicos β no degenerados y la determinación del ángulo de rotación χ_1, paso necesario para el proceso final de perfeccionamiento de la estructura de una proteína mediante RMN. No obstante, la determinación experimental de las constantes de acoplamiento vecinal ${}^3J_{\alpha\beta}$ en muestras no enriquecidas resulta complicada, tanto por la limitada resolución, en algunos casos el valor de la constante de acoplamiento es cercano al error experimental, como por los solapamientos posibles en los espectros homonucleares. Se han desarrollado multitud de estrategias, incluyendo sofisticadas secuencias de pulsos [115]. En este apartado se mostrará una estrategia, en gran parte desarrollada por nuestro grupo [64, 126], así como por otros laboratorios.

La utilización de espectros TOCSY a diversos τ_{mezcla} no sólo ha mostrado su utilidad para la completa identificación de sistemas de espín en proteínas [64], sino que la adquisición conjunta de TOCSY con un intervalo de τ_{mezcla} entre 15 y 90 ms puede permitir en conjunción con datos de NOE una vía sencilla para la asignación estereoespecífica y la determinación del ángulo χ_1. Esta estrategía se ha verificado en una pequeña proteína, m-EGF (factor de crecicimiento epidérmico) de la que se disponía de datos abundantes y precisos de ${}^3J_{\alpha\beta}$ y NOE intra y secuenciales.

Como paso previo, se ha comprobado experimentalmente una relación directa entre las constantes de acoplamiento vecinales ${}^3J_{\alpha\beta}$, directamente correlacionadas con la asignación estereoespecífica y el valor de χ_1, y la intensidad de los picos cruzados $H_\alpha - H_\beta$ de los espectros bidimensionales de correlación total (TOCSY) a tiempos de mezcla, τ_{mezcla}, pequeños. En la figura 5.7 se pueden observar los picos cruzados TOCSY a tiempo de mezcla (τ_{mezcla}) de 13 ms para 6 aminoácidos, N32 y S28; C14 y Y29; N16 y L26, de izquierda a derecha y de arriba a abajo, para la proteína m-EGF de la que se dispone de datos de ${}^3J_{\alpha\beta}$ obtenidos mediante espectros E.COSY. Cómo se puede observar en dicha figura, las intensidades de los picos correspondientes a ambos protones β para N32 y S28 son bastante similares, ${}^3J_{\alpha\beta d}$(N32) $\sim^3 J_{alpha\beta u}$(N32) $\sim 7Hz$, y ${}^3J_{\alpha\beta d}$(S28) $\sim^3 J_{alpha\beta u}$(S28)

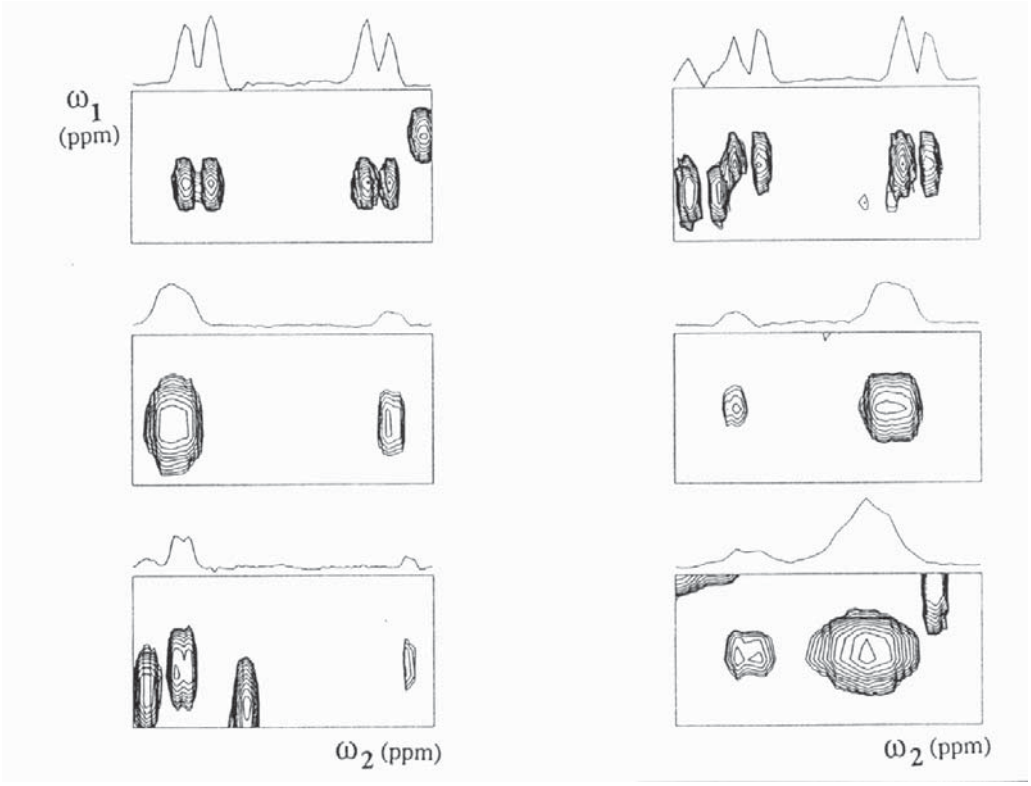


Figura 5.7: Diagrama de contornos de diversos aminoácidos de un espectro ^1H TOCSY a 500 MHz de una muestra de m-EGF 2.2 mM en 100 % D_2O, a pH 3.1 y 28^oC. a) ASN32 y SER28, CYS14 y TYR29 y ASN16 y LEU26 de izquierda a derecha y de arriba a abajo.

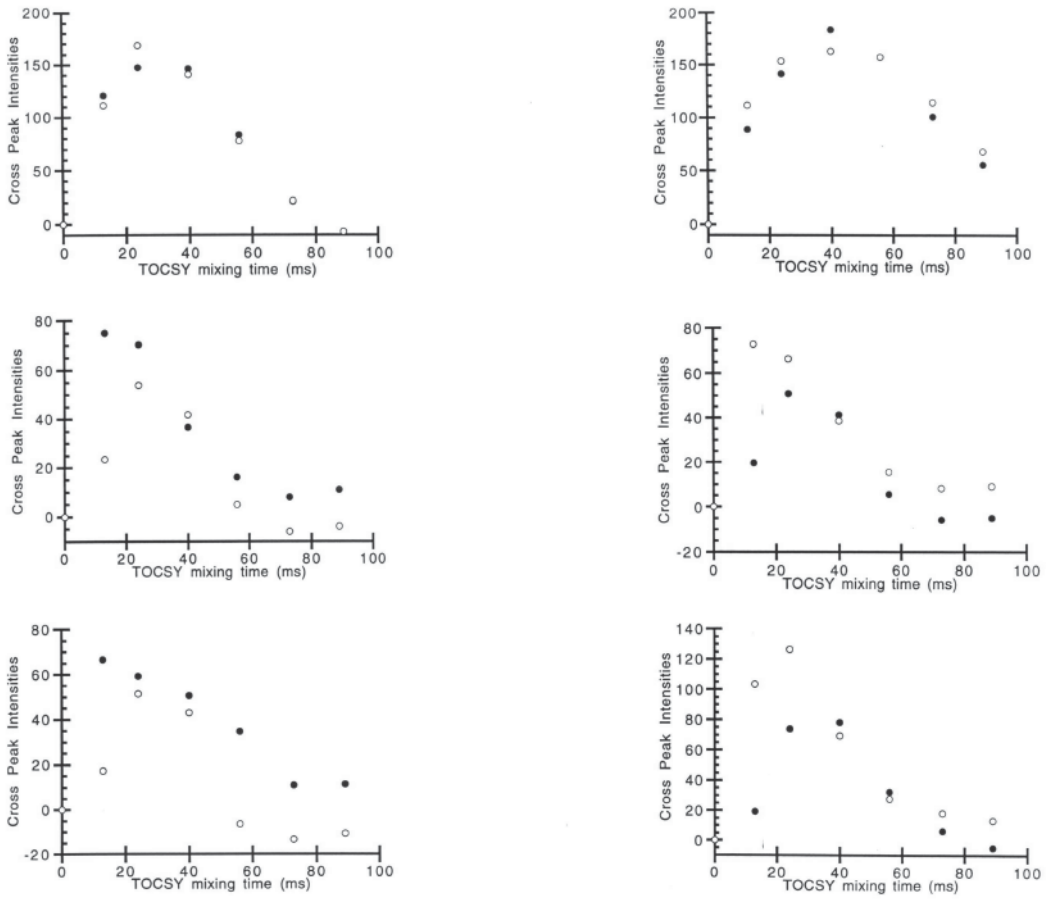


Figura 5.8: Variación de los picos cruzados TOCSY α-β de distintos aminoácidos de un espectro ^1H TOCSY a 500 MHz de una muestra de m-EGF 2.2 mM en 100 % D_2O, a pH 3.1 y 28^oC. a) ASN32 y SER28, CYS14 y TYR29 y ASN16 y LEU26 de izquierda a derecha y de arriba a abajo. $\circ H_u$, $\bullet H_d$.

$\sim 4Hz$, donde d y u significan resonancia a campo bajo y alto respectivamente. En el caso de C14 y N16, ${}^3J_{\alpha\beta d} >^3 J_{\alpha\beta u}$ mientras que para Y29 y L26 ${}^3J_{\alpha\beta d} <^3 J_{\alpha\beta u}$.

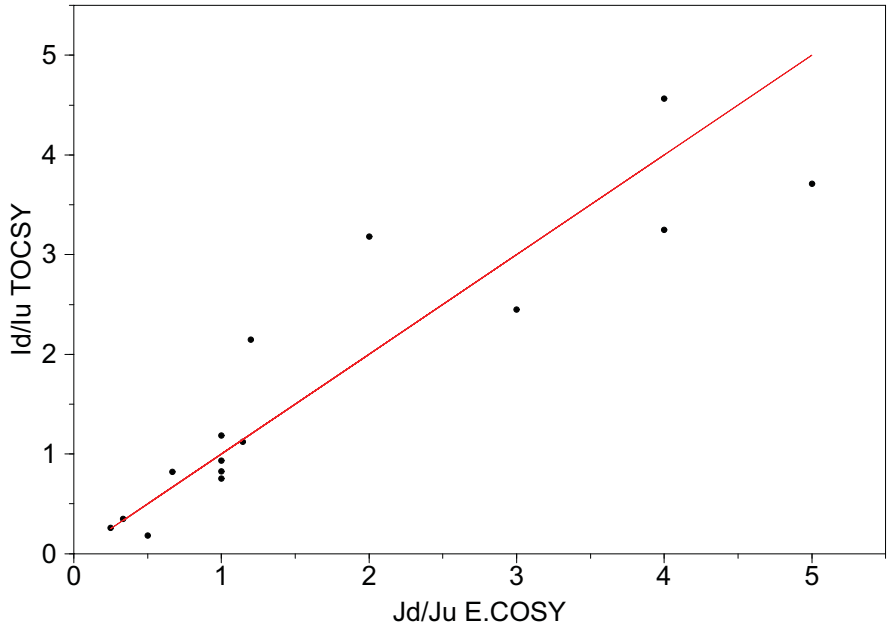


Figura 5.9: Variación de la razón de intensidades de picos cruzados (I_d/I_u) de un TOCSY con $\tau_{mezcla} = 13ms$ a 500 MHz vs constantes de acoplamiento (J_d/J_u) de E.COSY.

Complementariamente, se llevó a cabo un análisis empírico de la transferencia de magnetización a lo largo del sistema de espín en función de la potencia del bloqueo de espín del experimento TOCSY, τ_{mezcla}. En la figura 5.10 se puede observar la variación de la intensidad de picos cruzados de un TOCSY a distintos τ_{mezcla} para el m-EGF. El análisis de las gráficas obtenidas nos permite extraer dos conclusiones:

- sólo a tiempos de mezcla cortos, $\tau_{mezcla} < 15ms$, se mantiene la relación lineal entre $I_{\alpha\beta d}/I_{\alpha\beta u}$ y ${}^3J_{\alpha\beta d}/{}^3J_{\alpha\beta u}$
- en el caso de que ${}^3J_{\alpha\beta d} \sim^3 J_{\alpha\beta u}$ se presentan dos posibilidades. Las constantes pueden ser superiores a 5 Hz o inferiores a 5 Hz, ${}^3J_{\alpha\beta d} > 5Hz$ o ${}^3J_{\alpha\beta d} < 5Hz$, relacionadas con una situación dinámica de la cadena lateral, o con un rotámero único ($\chi_1 = 60^o$), respectivamente.

La relación lineal entre $I_{\alpha\beta d}/I_{\alpha\beta u}$ y ${}^3J_{\alpha\beta d}/{}^3J_{\alpha\beta u}$ se comprueba en la figura 5.9, en la que se representa $I_{\alpha\beta d}/I_{\alpha\beta u}$(TOCSY) a tiempo de mezcla 13

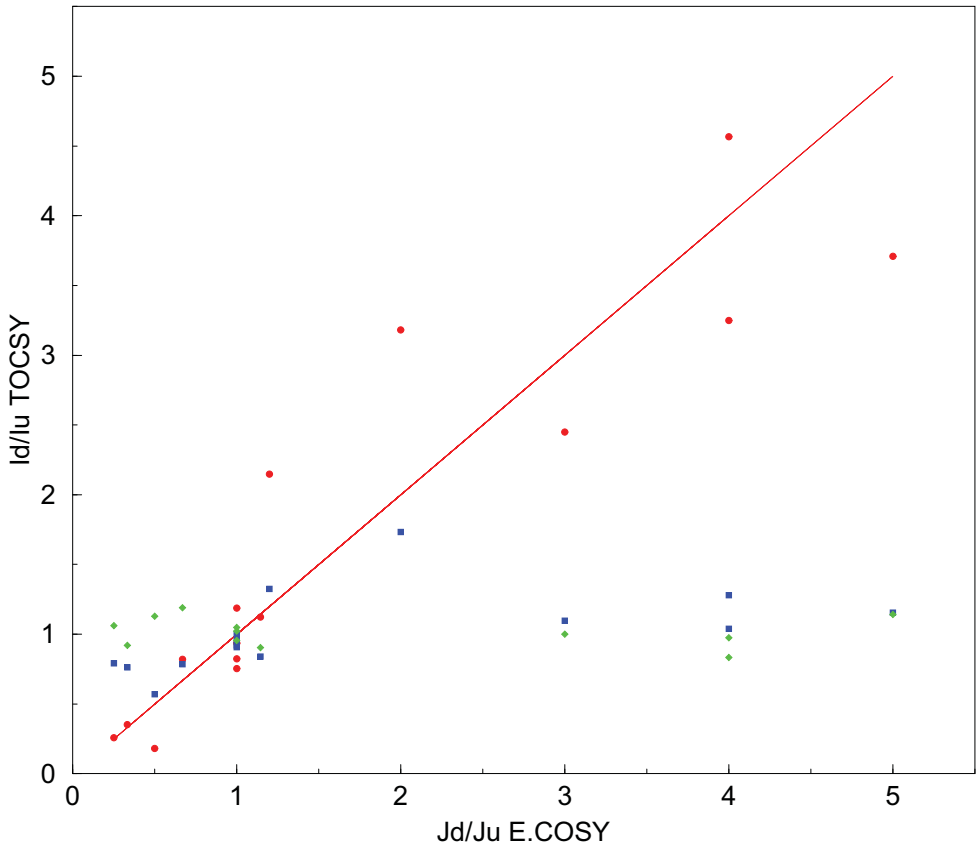


Figura 5.10: Variación de las intensidades de picos cruzados TOCSY con τ_{mezcla}.i $\tau_{mezcla} = 13ms$, (circulos rojos); $\tau_{mezcla} = 24ms$, (cuadrados azules); $\tau_{mezcla} = 40ms$, (rombos verdes).

ms vs ${}^3J_{\alpha\beta d}/{}^3J_{\alpha\beta u}$ (E.COSY) [128] para el m-EGF. Para tiempos de mezcla superiores, estas representaciones gráficas no corresponden a una relación lineal (como puede observarse en la figura 5.10), debido a que la transferencia de magnetización entre sistemas de espín directamente acoplados, únicamente se produce a τ_{mezcla} muy cortos ($\tau_{mezcla} \leq 15ms$). A τ_{mezcla} más largos la magnetización circula a lo largo de todo el sistema de espín en ambos sentidos, con una función de transferencia complicada y distinta para cada sistema de espín, eliminando de esta manera la linealidad de la relación entre la intensidad de la magnetización y el valor de las constantes de acoplamiento [127, 129]. Incluso para sistemas de espín tan sencillos como AMX (p.e. SER, CYS, ASP, ASN, HIS, PHE, TYR y TRP en proteínas) este comportamiento desaparece para τ_{mezcla} relativamente cortos, como por ejemplo $\tau_{mezcla} \geq$ 24 ms para m-EGF, como puede observarse en la figura 5.10. Asi mismo, en la parte superior de dicha figura se puede comparar la diferente conducta de la evolución de la magnetización para dos casos distintos, N32 (${}^3J_{\alpha\beta} \sim$ 7 Hz, conformación promedio, estado dinámico) y S28 (${}^3J_{\alpha\beta} \sim$ 4 Hz, rotámero único, conformación *gauche-gauche*). Para N32 la magnetización disminuye hasta anularse, incluso llegando a valores negativos, un comportamiento parecido se observa para todos los aminoácidos con valores de ${}^3J_{\alpha\beta}$ similares (Y3, D27, N32, D46, W49 y W50). Sin embargo, en el caso del residuo S28, la intensidad de los picos cruzados evoluciona de tal modo, que a τ_{mezcla} superiores a 90 ms es del orden del valor a τ_{mezcla} cortos. Esta conducta diferente puede estar relacionada con la distinta dinámica de ambas cadenas. Es bien conocido, que los efectos de relajación, entre ellos el tiempo de correlación del movimiento de giro global de la molécula τ_m o particular de una parte de la misma, disminuyen la magnitud de funciones de transferencia de coherencia dependientes del tiempo [129]. Por consiguiente se puede diferenciar ambas situaciones, conformación rotacional promedio y única *gauche-gauche*, a través de la evolución de la función de transferencia de coherencia (intensidad de picos cruzados) con τ_{mezcla}. Esta estrategia no ha sido explorada hasta el momento y por consiguiente es la primera vez que se demuestra la utilidad del análisis de espectros TOCSY a τ_{mezcla} superiores a 70 ms como ayuda para la asignación estereoespecífica.

Una vez diferenciadas las dos situaciones generales, conformación única – *trans-gauche*, *gauche-trans* y *gauche-gauche* – rotámero promedio, a partir de la razón entre I_d/I_u, la asignación estereoespecífica y la determinación de χ_1 necesita ser completada mediante la verificación de las distintas distancias d($H_\alpha - H_{\beta 2}$) y d($H_\alpha - H_{\beta 3}$), o la relación entre ambas. Los va-

lores experimentales obtenidos son: ${}^3J_{\alpha\beta 2}$ y ${}^3J_{\alpha\beta 3}$, $I_{\alpha\beta}$ (TOCSY), $I_{C\alpha H\beta}$ (HSQCTOCSY), $I_{\alpha\beta}$ (NOESY). La asignación estereoespecífica de los protones β y la determinación de χ_1 no puede ser precisada con esta información. La limitación se encuentra centrada en la imprecisión, debido a la transferencia por acoplamiento escalar, en la evaluación de $I_{\alpha\beta}$ (NOESY). No obstante esta dificultad puede ser superada a través de las consideraciones geométricas conformacionales de distancias d($H_{Ni}H_{\beta 2}$), d($H_{Ni}H_{\beta 3}$), d($H_{\beta 2}N_{i+1}$) y d($H_{\beta 3}N_{i+1}$) como se ha explicado en la sección de métodos mediante comparación con las gráficas de la figura 5.5. Como resultado final en la tabla 5.4 se presentan los datos completos para el m-EGF. Así mientras que mediante la aplicación combinada de un espectro de correlación específicamente diseñado, E.COSY, para determinar los valores de ${}^3J_{\alpha\beta}$ [128] y NOE se pudo realizar la asignación estereoespecífica de 9 residuos, con la utilización de un sencillo experimento TOCSY con τ_{mezcla} igual a 15 ms, la asignación estereoespecífica y la determinación del valor de χ_1 se amplió en 7 residuos más (LYS6, TYR10, LYS15, CYS33, CYS42, GLU51 y ARG53). En otras palabras, la adquisición de un conjunto simultáneo de espectros TOCSY a diversos τ_{mezcla} (de 15 a 90 ms) [134] y su combinación con datos de NOE ha permitido la asignación estereoespecífica y consiguiente evaluación de χ_1, para 16 residuos para m-EGF.

Un caso particular para la determinación del ángulo diedro χ_1 la constituye la treonina. Para este residuo la asignación estereoespecífica es obvia, la distinción entre la resonancia del protón β y del grupo metilo en el Cγ_2 es sencilla. No obstante, paralelamente no es tan simple la determinación del valor χ_1. Aunque la orientación de las cadenas laterales puede quedar en gran parte fijada por NOEs intra e interresiduo, la determinación del valor de χ_1 puede ayudar en gran medida a mejorar la calidad de la estructura de la proteína, especialmente en las zonas cercanas o en contacto inmediato con las treoninas. La metodología desarrollada, mediante el uso de los espectros TOCSY a varios τ_{mezcla} (de 15 a 90 ms), y aplicada a aquellos aminoácidos con dos protones metileno se puede extender a la treonina. La evolución de la magnetización desde el protón α al protón β en la treonina en función del τ_{mezcla} es bastante diferente dependiendo del valor de la constante de acoplamiento escalar ${}^3J_{\alpha\beta}$ [129]. Así, por ejemplo, las funciones teóricas de transferencia de coherencia indican que cuando ${}^3J_{\alpha\beta}$ es 3.5 Hz, la transferencia de magnetización desde el protón α hacia H_β y $\gamma_2 CH_3$ sólo es efectiva para valores de τ_{mezcla} inferiores a 50 ms. Por el contrario, cuando ${}^3J_{\alpha\beta}$ es del orden de 12 Hz, la transferencia de magnetización desde el protón α al protón β alcanza un máximo a τ_{mezcla} inferiores

Residuo	H_β	E.COSY $^3J_{H\alpha/H\beta}$	I.TOCSY H_α/H_β	I.NOE $H_{Ni}/H_{\beta i}$	I.NOE $H_{\beta i}/H_{Ni+1}$	I.NOE $H_{\alpha i}/H_{\beta i}$	ϕ	ψ	χ_1	Asig. Estereo.
Y3	3.08	6	52	400	–	13	-80 ± 20	150 ± 30	–	–
	2.73	9	63	600	–	4			–	–
C6	2.98	–	68	500	–	8.5	-160 ± 90	100 ± 80	-60 ± 30	β_2
	2.81	–	7	300	–	8.2				β_3
Y10	3.28	–	7.1	400	300	18.5	-100 ± 30	20 ± 50	-60 ± 30	β_3
	2.67	–	26.5	600	250	12.1				β_2
C14	2.58	12	74.4	600	250	7.4	-90 ± 30	120 ± 30	-60 ± 30	β_2
	2.26	3	22.9	300	400	10.6				β_3
L15	1.61	–	94.7	1200	1000	13.0	-120 ± 50	130 ± 50	-60 ± 30	β_2
	1.25	–	11.5	300	1500	18.5				β_3
N16	1.98	10	37.1	100	80	9	60 ± 100	-5 ± 30	-60 ± 30	β_2
	1.37	2	10	50	200	18				β_3
C20	3.36	4	12.4	1200	300	14	-90 ± 20	130 ± 50	180 ± 30	β_2
	3.13	12	35.3	1200	200	12				β_3
H22	3.19	12	49.7	500	200	6.5	-110 ± 40	130 ± 50	-60 ± 30	β_2
	3.13	4	20.3	sol.	400	13.5				β_3
L26	1.55	5	10.3	500	sol.	18	-140 ± 70	-5 ± 30	180 ± 30	β_2
	1.44	10	56.5	500		11				β_3
D27	3.17	8	62.0	350	400	18	66 ± 110	30 ± 70	–	–
	2.69	7	55.3	400	600	16.5				–
S28	3.65	4	25	250	1000	10	-90 ± 20	140 ± 40	60 ± 30	β_2
	3.52	4	30.3	400	600	8.5				β_3
Y29	2.48	3	10.3	150	1000	13	-100 ± 30	140 ± 40	-60 ± 30	β_3
	2.28	12	40	500	400	7				β_2
C31	2.77	12	34.7	500	sol.	10	-140 ± 70	130 ± 50	-60 ± 30	β_2
	2.59	3	7.6	200	400	16				β_3
N32	2.98	7	39.7	500	600	14	-90 ± 30	80 ± 110	–	–
2.76	7	33.5	sol.	400	22				–	
C33	3.28	–	4.1	200	600	–	-80 ± 10	-135 ± 80	-60 ± 30	β_3
	2.64	–	17.6	600	300	–				β_2
C42	3.60	–	108	200	–	sol.	80 ± 130	45 ± 80	180 ± 30	β_3
	3.15	–	54	250	–	sol.				β_2
Q43	2.18	–	–	350	250	20	-90 ± 20	-30 ± 8	–	–
	1.77	–	–	400	250	18				–
R45	1.63	–	90.3	–	sol.	–	-90 ± 20	120 ± 60	–	–
	1.56	–	25.6	–	sol.	–				–
D46	2.75	7	37.1	400	800	9	-90 ± 20	-100 ± 40	–	–
	2.48	7	39.7	800	1000	14				–
R48	1.43	10	36.2	–	–	10	-40 ± 10	20 ± 50	–	–
	1.40	5	14	–	–	8.5				–
W49	3.22	7	32.6	1000	sol.	17	-40 ± 5	-5 ± 30	–	–
	3.12	7	30	1000	200				–	
W50	3.05	6	30.6	sol.	300	16.5	-100 ± 40	-10 ± 30	–	–
	2.77	5	14.4	sol.	250	12				–
E51	1.97	–	23.9	250	300	9.5	-100 ± 40	15 ± 50	-60 ± 30	β_3
	1.80	–	55.0	400	300	14				β_2
R53	1.81	–	–	250	–	14			-60 ± 30	β_3
	1.65	–	–	400	–	31.2				β_2

Tabla 5.4: Asignación estereoespecífica completa del EGF realizada mediante la metodología propuesta.

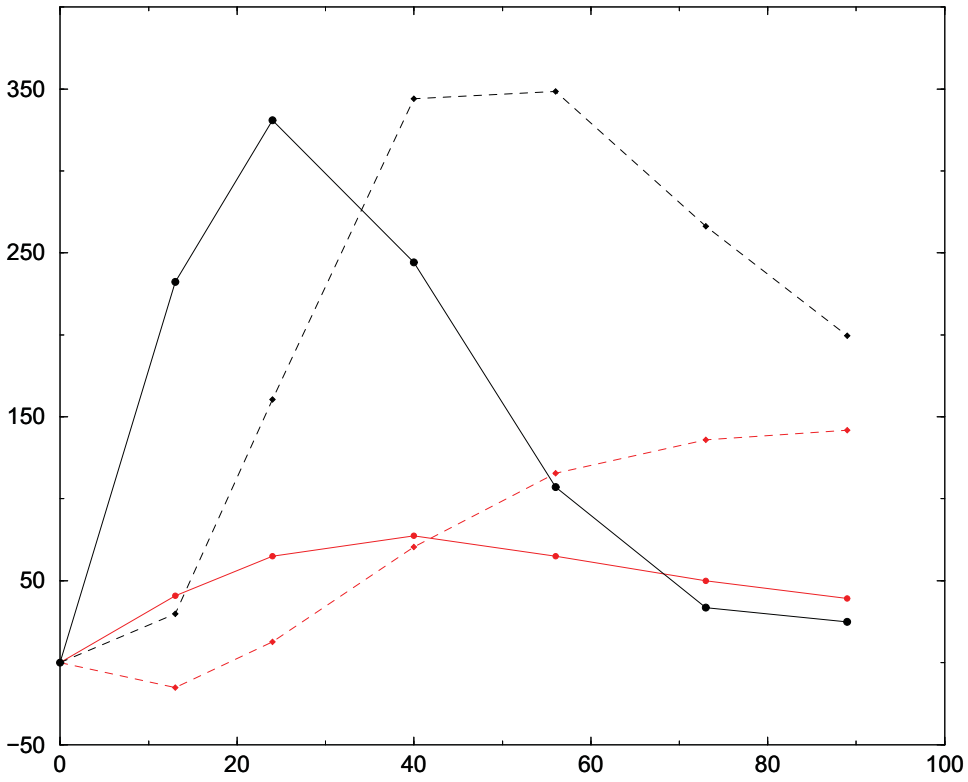


Figura 5.11: Representación de las intensidades tocsy de los picos $\alpha - \beta$ (trazo contínuo) y $\alpha - \gamma$ (trazo discontínuo) para THR30 (negro) y THR44 (rojo) del EGF frente al tiempo de mezcla τ_{mezcla}.

a 50 ms, y de el protón α al $\gamma_2 CH_3$ cuando el τ_{mezcla} es superior a 50 ms. Este comportamiento teórico queda totalmente reflejado en los datos experimentales correspondientes a las dos treoninas en el m-EGF, THR30 y THR44, tal y como queda patente en la figura 5.11.

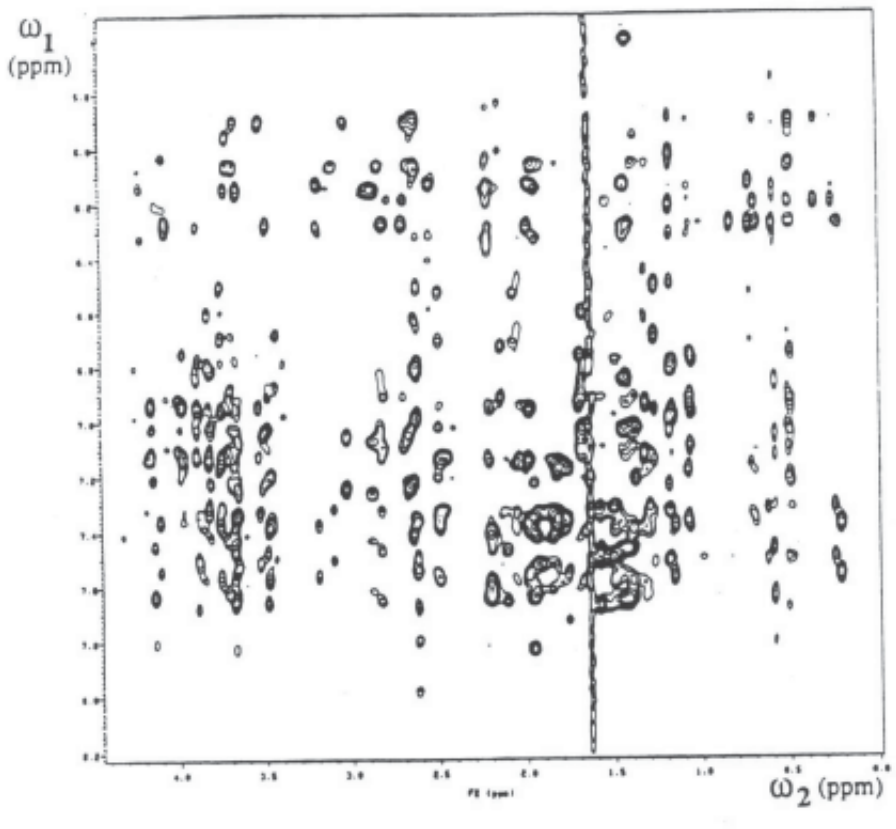


Figura 5.12: Región ω_1(5.50, 8.22 ppm) y ω_2(0.00,4.50 ppm) del espectro NOESY del dominio Z 2mM sin desacoplamiento de los espínes ^{15}N para observar la separación.

De la figura 5.11 se puede deducir que $^3J_{\alpha\beta} \geq$ 10 Hz para el residuo THR44 y $^3J_{\alpha\beta} \leq$ 5 Hz para el resiudo THR30. Según estos valores, el valor de χ_1 sería de -60^o para THR44, mientras que para THR30 χ_1 podría adoptar los valores 60^o o -180^o. Esta situación es similar a la que se puede observar para sistemas con dos protones metilénicos, en la que los valores de $^3J_{\alpha\beta}$ pueden diferenciar valores de χ_1 de 60^o respecto a valores de -60^o y 180^o, pero la distinción entre estos últimos requiere de datos adicionales como son otras constantes de acoplamiento o datos de NOE. En el caso de la treonina, la diferenciación entre χ_1 = -180^o respecto al rotámero χ_1 = 60^o puede realizarse a través del NOE. Concretamente, la distancia

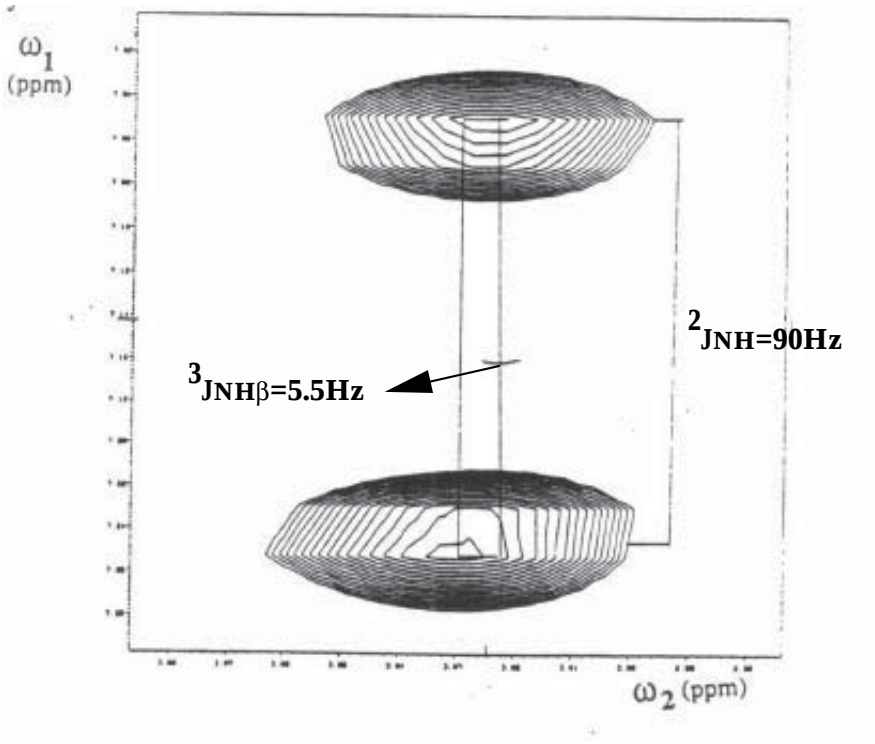


Figura 5.13: Ampliación un pico cruzado del espectro NOESY del dominio Z en las mismas condiciones que la figura 5.12 sin desacoplar los espínes ^{15}N.

entre el protón α y el grupo metilo de γ_2 es aproximadamente de 4 Å cuando χ_1 vale -180^o, mientras que en el caso de que χ_1 sea 60^o la diferencia entre las distancias $H_\alpha - H_\beta$ y $H_\alpha - H_{\gamma 2CH3}$ es de 0.6 Å apoximadamente. Por consiguiente, cuando χ_1 valga -180^o el NOE correspondiente a $H_\alpha - H_\beta$ será intenso (distancia inferior a 2.5 Å) y el NOE correspondiente a $H_\alpha - H_{\gamma 2CH3}$ débil. Esta es la situación observada para el residuo THR30, en el que $I_{NOE\alpha/\beta}$ es aproximadamente el doble que $I_{NOE\alpha/\gamma 2CH3}$ (medidos en D_2O a un τ_{mezcla} de 25 ms). Por ello, el valor de χ_1 más adecuado sería -180^o. Con respecto a la THR44 $I_{NOE\alpha/\alpha} \sim I_{NOE\alpha/\gamma 2CH3}$, con intensidades medias-débiles como corresponde a las distancias superiores a 3 Å esperadas para un valor de χ_1 igual a -60^o. Estos valores de χ_1 deducidos en función de los valores de acoplamiento escalar $^3J_{\alpha\beta}$ a partir de la evolución de la transferencia de magnetización con el τ_{mezcla} más los valores de intensidades NOE son totalmente concordantes con los valores promedio del conjunto de estructuras refinadas de m-EGF [135], tal que $\chi_1 = -170^o \pm 20^o$ y $\chi_1 = -70^o \pm 20^o$ para THR30 y THR44 respectivamente.

La aplicación de la metodología propuesta así como las conclusiones experimentales derivadas del análisis de los datos obtenidos para el m-EGF se comprobaron mediante su utilización con los datos experimentales obtenidos para el dominio Z de la proteína A. El dominio Z está formado por 72 aminoácidos, y estructuralmente está constituido por 3 hélices α, lo que la diferencia del mEGF que está compuesto en su mayoría por hojas β. De esta manera se contrastaba el formalismo para evaluar χ_1 y realizar la asignación estereoespecífica, y simulatáneamente se ha comprobado la correlación conformacional entre ϕ, ψ y χ_1,para otro tipo de estructura regular secundaria, hélices α con ϕ y ψ centrados alrededor de -60^o y -40^o, respectivamente.

Como hecho diferencial, en esta proteína no se conocían los valores de $^3J_{\alpha\beta}$ mediante E.COSY. No obstante, y como se disponía de una muestra enriquecida en ^{15}N, se determinaron los valores de $^3J_{N\beta}$ y se pudó realizar el análisis sobre ellos expuesto en la tabla 5.3. Para ello se realizó un experimento bidimensional homonuclear NOE (NOESY) sin desacoplamiento de los espines ^{15}N, para que, de esta manera se obtuvieran picos cruzados dobles como se pueden observar en la figura 5.12, tal que en la dimensión de la frecuencia ω_1 la separación corresponde a $^2J_{NH\alpha}$ mientras que en la direccion ω_2 la separación corresponde a $^2J_{H\alpha}$.

En la figura 5.13 se presenta la expansión de un pico cruzado, en el que de una manera aproximada se incluyen los valores de ambas constantes

Conformación	ϕ, ψ, χ_1	RMS (s.d.)	RMS (c.d.)
α-hélice dextro	-57,-47,-60	0.110	0.006
hoja β paralela	-139,135,180	0.105	0.009
hoja β antiparalela	-119,113,60	0.112	0.011
hoja β tipo I (2)	-60,-30,-60	0.112	0.007
hoja β tipo I (3)	-90,0,180	0.111	0.004
hoja β tipo II (2)	-60,120,60	0.111	0.008
hoja β tipo II (3)	80,0,-60	0.110	0.005
hélice 3_{10} dextro	-49,-26,-60	0.096	0.006
α-hélice levo	57,47,180	0.110	0.009

Tabla 5.5: Tabla de resultados de desviación cuadrática media para los valores de distancias calculadas para una geometría ideal en diferentes elementos de estructura secundaria. s.d.: sin corrección de desfase, c.d.: con corrección de desfase.

de acoplamiento. La combinación de toda la información, ${}^3J_{\alpha\beta}$, ${}^3J_{N\beta}$, $I_{\alpha\beta}$, $I_{N\beta}$ y $I_{\beta iNi+1}$ ha permitido la asignación estereoespecífica, evaluación de χ_1, así como la verificación de ϕ y ψ necesarios para el refinamiento final de la estructura [133].

5.3.2 Datos simulados. Geometría ideal

Para comprobar la validez del algoritmo utilizado en el cálculo de distancias se construyeron varias cadenas polipeptídicas ideales cada una conteniendo tres residuos de serina mediante el paquete de Modelización Molecular InsightII [136]. En cada cadena los valores de ϕ y ψ de cada residuo fueron establecidos iguales a los valores correspondientes a cada uno de los elementos de estructura secundaria mostrados en la tabla 5.5. Los valores de los ángulos diedros χ_1 fueron puestos de acuerdo a las conformaciones más favorecidas de orientación de cadena lateral en cada una de las estructuras regulares. Las distancias d1 a d9 fueron medidas para el residuo central en cada una de las cadenas polipeptídicas utilizando la capacidad gráfica interactiva del programa InsightII.

Dad la flexibilidad de HYPER para incluir diferentes topologías, en los cálculos de las distancias se utilizaron como distancias de enlace y ángulos de enlace los establecidos para el residuo serina en las librerias de topología del programa InsightII. De este modo, se comprueba únicamente la calidad del método de cálculo y la exactitud de la programación de las operaciones matriciales. Los resultados obtenidos de distancias se compararon con

Conformación	ϕ	ψ	χ_1	ϕ c.	ψ c.	χ_1 c.
α-hélice dextro	-57	-47	-60	-58 / -56	-52 / -44	-60 / -60
hoja β paralela	-139	135	180	-140 / -138	128 / 140	180 / 180
hoja β antiparalela	-119	113	60	-122 / -116	106 / 120	60 / 60
hoja β tipo I (2)	-60	-30	-60	-60 / -60	-32 / -28	-60 / -60
hoja β tipo I (3)	-90	0	180	-90 / -90	0 / 0	180 / 180
hoja β tipo II (2)	-60	120	60	-60 / -60	110 / 128	60 / 60
hoja β tipo II (3)	80	0	-60	78 / 80	0 / 0	-60 / -60
triple hélice colágeno	-78	-149	60	-78 / -78	-154 / -146	60 / 60
hélice 3_{10} dextro	-49	-26	-60	-50 / -48	-26 / / -26	-60 / -60
α-hélice levo	57	47	180	56 / 58	36 / 54	180 / 180

Tabla 5.6: Tabla de resultados de la búsqueda de ángulos diedros realizada por HYPER para una geometría ideal en diferentes elementos de estructura secundaria. *c.* indica valores calculados.

los medidos interactivamente y se calculó la desviación cuadrática media de las mismas que se recogen en la tabla 5.5. Como se puede apreciar las desviaciones cuadráticas medias son muy bajas aunque no son nulas. Analizando en detalle los resultados para cada una de las distancias se comprobó que las distancias que implican a los protones β tenian un ligero error sistemático debido fundamentalmente a la diferencia en el desfase de los ángulos diedros que implican a dichos protones respecto al ángulo diedro χ_1. Aunque en principio cabe suponer una distribución simétrica sobre el ángulo χ_1 con desfase de 120^o y -120^o, en la práctica se observa que debidos al efecto estérico inducido por el oxígeno de carbonilo esta distribución es ligeramente asimétrica siendo los valores de desfase de 115^o y -125^o. Una vez corregido la asimetría del desfase, se obtuvieron valores de la desviación cuadrática media inferiores a 0.01 Å en todos los casos para la geometría ideal.

En la tabla 5.6 se pueden observar los resultados de la búsqueda de diedros que produjo el programa HYPER sobre las cadenas polipeptídicas ideales aplicando márgenes de error de 0.2 Å a las distancias medidas interactivamente y de 0.2 Hz a las constantes de acoplamiento escalar calculadas mediante las ecuaciones de Karplus. La autoconsistencia del método de búsqueda se ve confirmada por el hecho de que los valores de ángulos diedros originales se encuentran en todos los casos dentro de los intervalos propuestos por el programa. Además, se obtuvieron soluciones consistentes sólo en los casos de asignación estereoespecífica correcta com-

probándose también la calidad del método en este aspecto.

5.3.3 Datos simulados. Estructura de alta resolución de Rayos X.

Se realizó una segunda comprobación de consistencia del modo en que se manejan las restricciones en el programa HYPER mediante el cálculo de intervalos de ángulos diedros permitidos a partir de datos de distancias y constantes de acoplamiento escalar derivados de las coordenadas cartesianas de una estructura de proteína de alta resolución determinada mediante espectroscopía de Rayos X. Para ello se seleccionó una estructura cristalográfica determinada mediante difracción de Rayos X y difracción de neutrones con una resolución de 1.26 Å. La estructura empleada es la de la ribonucleasa A de páncreas bovino (RNasa A) con 124 aminoácidos conteniendo una combinación de regiones en α-hélices, hoja β y regiones sin estructurar [137]. Se calcularon las distancias d1 a d9 con el programa PDBSTAT [161] para los 124 residuos y se les aplicó una desviación de 0.5 Å a cada una para crear los límites superior e inferior de las restricciones simuladas correspondientes necesarias para la búsqueda de ángulos diedros.

Las constantes de acoplamiento escalar necesarias se calcularon a partir de los ángulos diedros correspondientes medidos con el programa PDBSTAT [161] en la estructura cristalográfica y con las ecuaciones de Karplus adecuadas aplicándose una desviación de 1.0 Hz para crear los correspondientes límites superior e inferior de las restricciones de constantes de acoplamiento escalar simuladas. Las constantes relativas se calcularon suponiendo una incertidumbre de 0.5 o de 2 unidades. Los residuos 93 y 114 resultaron ser prolinas en cis. Otra de las características de HYPER es que permite incluir residuo por residuo distintas topologías por lo que en los datos de entrada se seleccionó ω igual a 0.0^o.

Los resultados de la búsqueda de diedros para un segmento representativo de la proteína se pueden observar en la figura 5.14 en la que se puede apreciar la coincidencia de los intervalos propuestos por el programa HYPER con los valores reales de la estructura cristalográfica utilizada. En todos los casos el valor real del ángulo diedro se encuentra dentro del intervalo propuesto por el programa HYPER. Cuando se introdujeron como restricciones las distancias correspondientes a protones metilénicos, la asignación estereo específica correcta fué siempre el resultado. En todos los casos HYPER dió solución.

El pequeño tamaño de los intervalos propuestos por el programa HY-

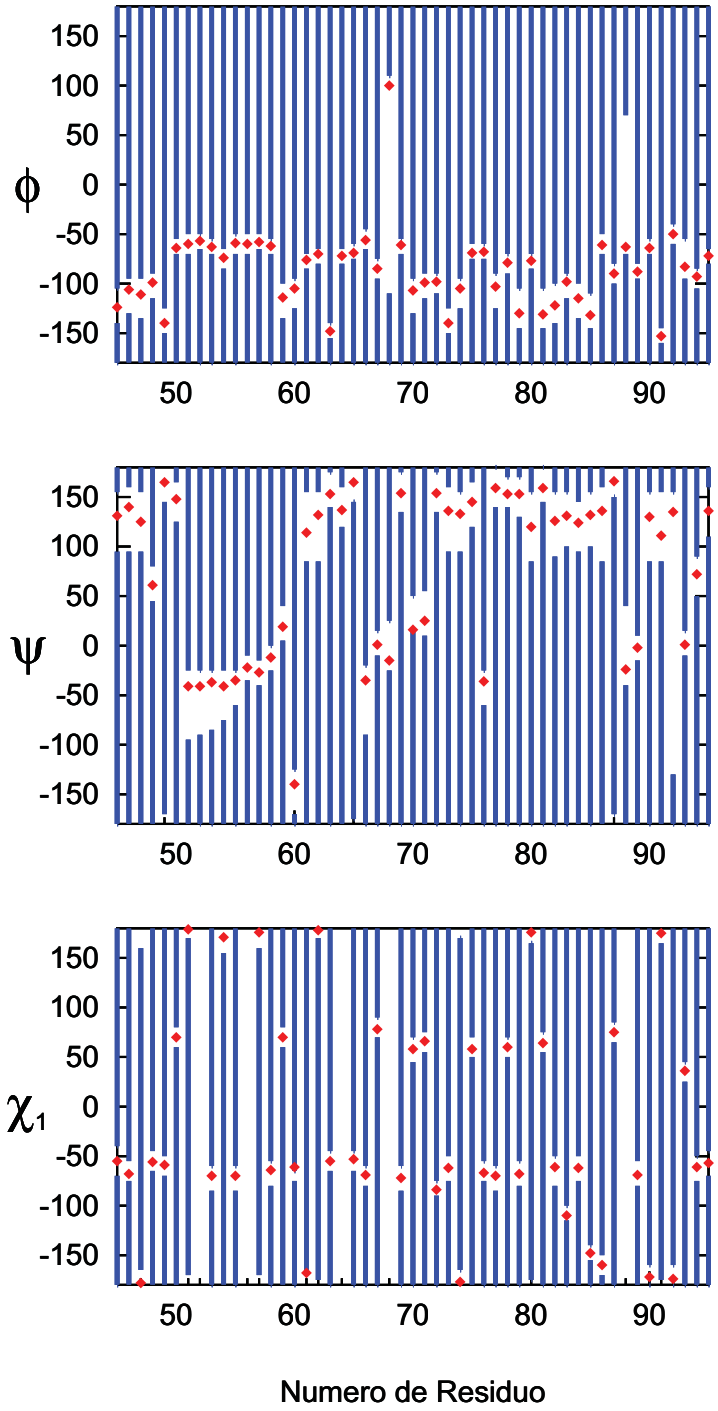


Figura 5.14: Representación esquemática de los resultados de la búsqueda de ángulos diedros realizada por HYPER para un intervalo representativo de residuos de la estructura de RNasa A utilizando el algoritmo de cálculo de distancias propuesto. Las líneas en negro indican el espacio de ángulo diedro prohibido según HYPER y los rombos indican el valor real en la estructura de RMN.

PER responde al elevado número de restriccones estructurales impuestas en la comprobación (22 restricciones por residuo). En todos los casos se ha considerado que todas las distancias interprotónicas de cada uno de los residuos ha sido determinada al tiempo que las constantes de acoplamiento escalar han sido medidas. Esto restringe en gran medida el espacio conformacional accesible aunque en la práctica es muy díficil encontrar en disolución un sistema en que la información accesible sea tan cuantiosa. En los resultados experimentales de RMN, los solapamientos y las ambigüedades suelen producir pequeños vacíos en la información estructural accesible aunque tanto las técnicas heteronucleares como las de RMN multidimensionales de más de dos dimensiones tienden día a día a paliar estos inconvenientes.

Sin embargo, en nuestro caso, como comprobación de la calidad del método de cálculo y del método de búsqueda, un número elevado de restricciones y el resultado coherente obtenido asegura al menos la autoconsistencia del procedimiento. Los resultados obtenidos demuestran de nuevo que el conjunto de valores de distancias y acoplamientos escalares calculados por el programa HYPER son completamente autoconsistentes a un nivel de resolución elevado para los 124 puntos específicos de ϕ, ψ y χ_1 correspondientes a la conformación cristalina de la RNasa A.

5.3.4 Datos experimentales de RMN. Dominio Z de la Proteína A

Una vez demostrada la consistencia interna del método de cálculo de distancias y del método de búsqueda utilizando el programa HYPER, se procedió a comprobar la eficacia del programa con datos experimentales reales obtenidos para el dominio Z de la proteína A de estafilococo [133]. Se dispusó de un razonable conjunto de 246 valores de restricciones de distancias de d1 a d9 y de relaciones entre constantes de acoplamiento escalar para 56 residuos del dominio Z de los 72 que contiene la proteína dando lugar a un promedio de 4.4 restricciones por residuo.

Las restricciones de distancia fueron determinadas mediante cálculos de matriz de relajación completa y se asumió una precisión de 0.5 Å. Los valores de las constantes de acoplamiento escalar ${}^3JH_N - H_\alpha$ se estimaron mediante el análisis de los espectros 2D-HSQC-J [138] y HMQC-J [139] y se asumió una precisión de 1 Hz. Las constantes de acoplamiento vecinal ${}^3J_{{}^{15}N-H\beta}$ se extrajeron de los espectros 2D homonucleares NOESY y los espectros 3D PFG [^{15}N] HSQC-NOESY utilizando muestra enriquecida en

^{15}N sin desacoplamiento en ^{15}N durante los dos periodos de evolución del protón [140]. Las valores relativos de las contantes de acoplamiento escalar $^{3}J_{\alpha-\beta}$ fueron estimadas a partir de la relación de intensidades de los picos cruzados $H_\alpha - H_\beta$ del espectro 2D-TOCSY [141] recogido con un tiempo de mezcla de 15 ms. Tanto los datos en sí como los detalles sobre los parámetros experimentales han sido tomados de la bibliografía.

Se realizaron los correspondientes cálculos con el programa HYPER para los 56 residuos del dominio Z de la proteína A con las restricciones indicadas cuando estas estuvieron disponibles utilizando un tamaño de rejilla de 5^o. El dominio Z no contiene glicinas. Los intervalos permitidos para ϕ, ψ y χ_1 propuestos por el programa HYPER están indicados en la figura 5.15. No se obtuvieron soluciones para todos los residuos debido al bajo número de restricciones pero en los residuos con un número de restricciones suficiente se obtuvieron intervalos de ángulos diedros suficientemente estrechos como puede observarse en la figura 5.15.

Se obtuvieron asignaciones estereoespecíficas para 5 de un total de 46 pares de protones β metilénicos no degenerados. Para un residuo, la HIS58, se encontraron soluciones inconsistentes para las dos posibilidades de asignación estreoespecífica, sugiriendo un promedio dinámico (rotámeros).

Una de las principales suposiciones en el procedimiento utilizado es que las restricciones a ser analizadas provienen de una conformación molecular simple estática. Sin embargo, incluso proteína globulares son de hecho un conjunto de conformaciones, a menudo interconvirtiéndose unas en otras en una escala de tiempos que es rápida comparada con las escalas de tiempo de desplazamiento químico y constantes de acoplamiento. Esta asumción de estaticidad suele ser una aproximación aceptable para las conformaciones de cadena principal y las cadenas laterales interiores, mientras que las cadenas laterales en la superficie de la proteína y las conformaciones de cadena principal en las regiones de los bucles generalmente adoptan multiples conformaciones que producen un resultado promedio en las medidas de RMN.

Un posible modo de detectar la presencia de estas conformaciones múltiples promedio es mediante el análisis de las restricciones obtenidas utilizando el programa HYPER. La obtención de inconsistencias en las restricciones de distancias locales puede ser indicativo de una interconversión entre diferentes conformaciones o de una falta de exactitud en las medidas experimentales. El usuario ha de decidir si la inconsistencia procede de una situación de intercambio o de posibles errores experimentales. Un

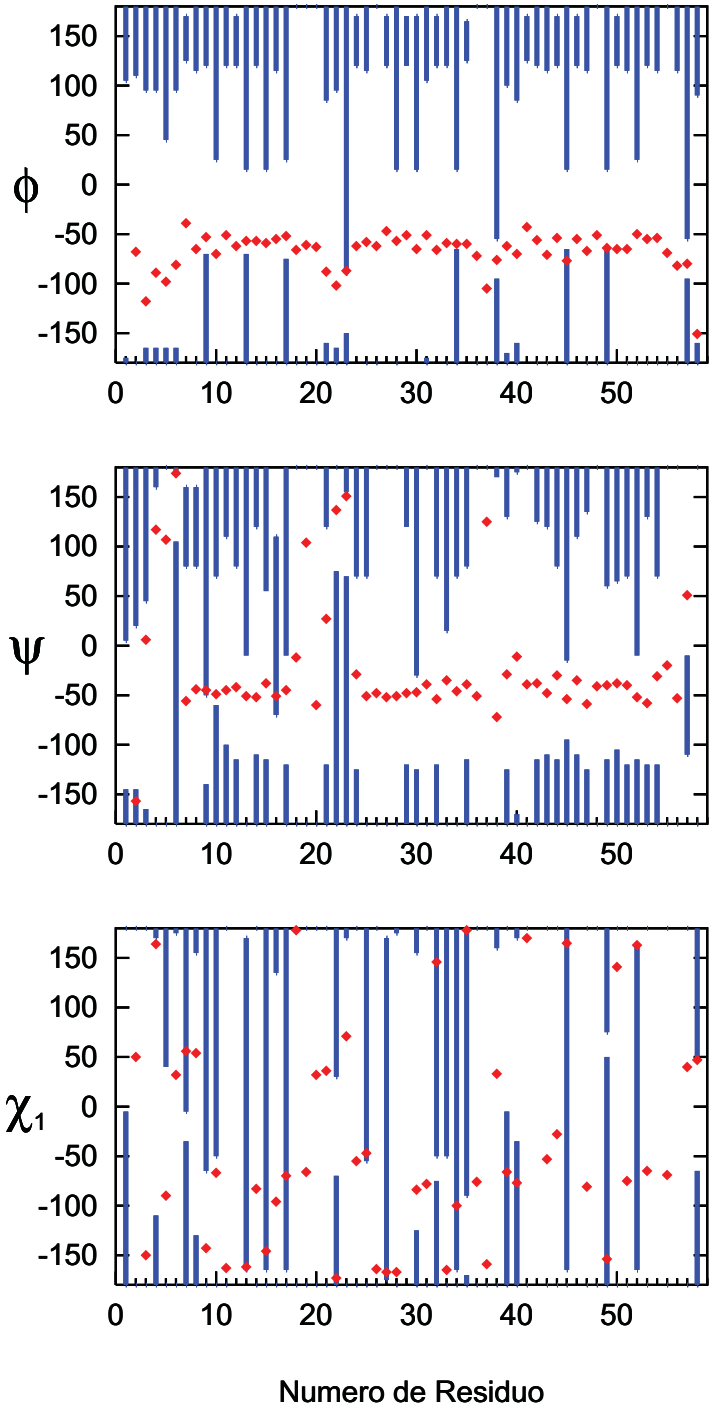


Figura 5.15: Representación esquemática de los resultados de la búsqueda de ángulos diedros realizada por HYPER para la estructura de del dominio Z de la Proteína A utilizando el algoritmo de cálculo de distancias propuesto. Las líneas en negro indican el espacio de ángulo diedro prohibidosegún HYPER y los rombos indican el valor real en la estructura cristalográfica.

modo posible de distinguirlos es realizar los cálculos sin las restricciones que producen inconsistencias y con ellas, la compatibilidad del resultado global con las inconsistencias locales.

5.3.5 Distancias a N residuos

Aunque el cálculo de distancias a más de dos residuos aún no ha sido incluido en el método de búsqueda en rejilla utilizado en el programa HYPER, el método de cálculo ya ha sido implementado y comprobado su consistencia. Para ello, se ha escogido la estructura del dominio Z de la proteína A determinada mediante RMN [133] y se ha procedido al cálculo de distancias entre protones a larga distancia en la cadena secuencial y su comparación con los valores medidos interactivamente. Se ha calculado la desviación cuadratica media de dichas distancias hasta un número de residuos de 10 para 20 distancias en cada nivel de separación. Los resultados de desviación cuadrática media se pueden observar en la figura 5.16.

Se puede observar que, aunque la desviación cuadrática media se mantiene dentro de unos niveles aceptables hasta una separación de 3 residuos, aumenta rápidamente al aumentar el número de residuos llegando a alcanzar valores muy superiores al error experimental propio de las restricciones introducidas. Esto hace posible la aplicación de este tipo de restricciones a larga distancia en la secuencia a elementos de estructura secundaria del tipo α-hélice. Sin embargo, la aplicación a hojas β es prácticamente imposible además de por la imprecisión de las medidas de distancia, por el gran número de grados de libertad implicados (cada residuo intermedio supone dos grados de libertad adicionales por lo menos).

No obstante, la falta de precisión en las medidas de distancia entre residuos bastante separados no es causa del método de cálculo sino de la falta de coincidencia entre los parámetros geométricos utilizados en el cálculo con los valores reales para los elementos topológicos característicos (distancias de enlace, ángulos de enlace y desfase en los diedros). Se puede comprobar esto observando la poca o nula desviación cuadrática media que se obtiene para cálculos de distancia hasta 10 residuos de separación utilizando la geometría ideal (ver figura 5.16). En la figura 5.17 podemos obsevar también la dispersión respecto a algunos parámetros geométricos estándar de distancia de enlace y ángulo de enlace de los valores medidos para la estructura del dominio Z de la proteína A [70].

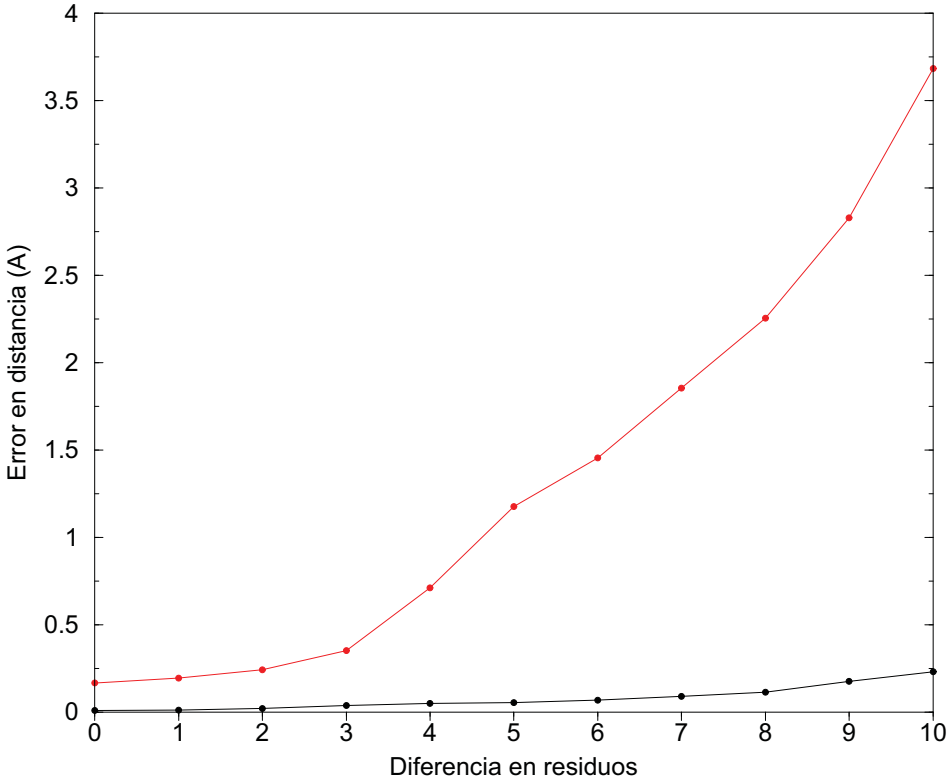


Figura 5.16: Curvas de desviación cuadrática media de las distancias calculadas mediante el algoritmo de cálculo propuesto respecto a las distancias reales frente al número de residuos de separación. Negro: geometría ideal, rojo: Dominio Z de la Proteína A por RMN.

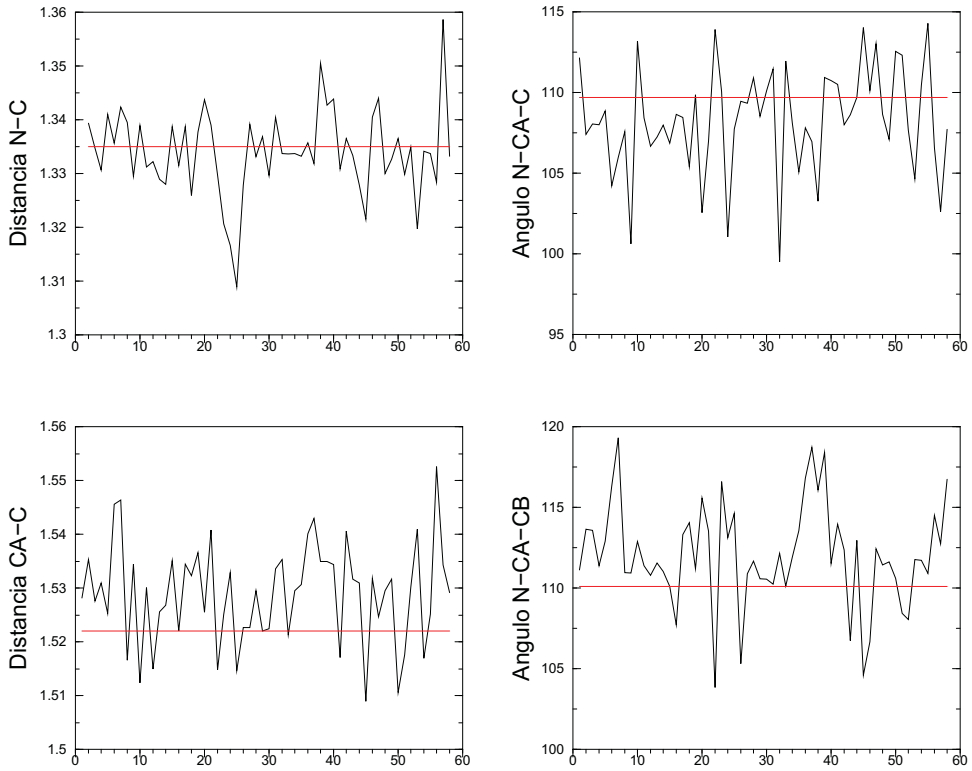


Figura 5.17: Desviación de los parámetros geométricos estándar para algunas longitudes de enlace y ángulos de enlace en la estructura por RMN del dominio Z de la proteína A.

5.4 Conclusiones

El método propuesto para calcular las distancias interprotónicas características de un residuo resulta, además de preciso, adaptable a diferentes situaciones y geometrías al tiempo que no resulta excesivamente costoso en tiempo de cálculo. La posibilidad de ampliar la búsqueda a un número de residuos más amplio hace también muy interesante la aproximación algebraica utilizada.

Por otra parte, la utilización de unos parámetros geométricos lo más ajustados posible a la geometría del sistema que se pretenda estudiar es muy importante para obtener unos resultados de elevada precisión y resulta crucial en el caso de intentar estudiar sistemas multiresiduo, vista la dependencia que muestran las medidas de distancias de los parámetros geométricos utilizados.

El método de asignación estereoespecífica propuesto nos ha permitido la asignación de un gran número de cadenas laterales cuando la información experimental ha sido suficiente. El análisis de las constantes de acoplamiento escalar y de los picos cruzados NOESY que implican a los protones β metilénicos no degenerados permite extraer información sobre el ángulo diedro χ_1 y sobre la asignación de los protones $H_{\beta 2}$ y $H_{\beta 3}$ de manera inequívoca.

Las inconsistencias señaladas en el resultado del programa HYPER puede aportar información sobre intercambios dinámicos entre múltiples conformaciones en las cadenas laterales. El análisis posterior de dichas inconsistencias debe permitir al usuario elucidar si proceden de un intercambio rápido entre diferentes conformaciones o si es necesario realizar más experimentos para resolver la inconsistencia.

El método de búsqueda en rejilla jerarquizada propuesto para el programa HYPER resulta ser efectivo cuando las restricciones estructurales de distancias y constantes de acoplameinto escalar son suficientes para restringir el espacio accesible a los diedros. El tiempo empleado en dichas determinaciones se ve reducido al utilizar algoritmos de búsqueda y de cálculo de distancias optimizados.

La ampliación de la búsqueda en rejilla a un número superior de residuos se ve condicionada por dos factores fundamentalmente. Por una parte la imprecisión asociada a la no coincidencia en los parámetros geométricos ideales utilizados en el cálculo y los parámetros geométricos reales y variables de la estructura en disolución puede producir grandes errores en los

cálculos de las distancias implicadas. Por otra parte, el número de grados de libertad a explorar se ve incrementada en gran medida al considerar un número superior de residuos y por lo tanto, el tiempo dedicado a la exploración aumentará exponencialmente haciendo impracticable la búsqueda a más de tres residuos por el momento (explosión combinatorial).

Capítulo 6

Estudio estructural de un hexanucleótido mediante RMN.

6.1 Introducción.

Durante años después del descubrimiento de la estructura en doble hélice del ADN por Watson y Crick [142], éste ha sido tratado como una molécula uniforme. Suponiendo que la información genética se encuentra encriptada como un vector lineal de codones, parece que la clave para la comprensión de la regulación de la expresión genética y los procesos de replicación se encuentra en la interacción de proteínas específicas con lugares determinados de la cadena de ADN que muestran una determinada secuencia de nucleótidos en una estructura uniforme. Sin embargo, en los últimos años, nuestro conocimiento sobre la estructura del ADN y sus interacciones con las proteínas ha aumentado considerablemente. De hecho, se ha descubierto que una gran cantidad de variaciones estructurales son posibles sobre la conocida estructura base propuesta por Watson y Crick dependiendo de la presencia de ciertos motivos en la secuencia. Dentro de la estructura estándar de dúplex B-ADN existen una gran cantidad de variaciones dependientes de la secuencia y del modo en que los pares de bases secuenciales se ven mutuamente [143]. Estas características, que son muy habituales en las estructuras de ADN y por lo tanto no se pueden considerar anómalas, adquieren especial importancia cuando se requiere de la cadena de ADN ciertas peculiaridades estructurales con fines de acoplamiento o reconoci-

miento frente a otras moléculas. Tal es el caso de la flexión de la cadena de ADN requerida para la complejación frente a una histona. En una escala espacial superior, el ADN puede adquirir conformaciones espaciales peculiares relativamente alejadas de la conocida doble hélice dextrógira como puedan ser, dobles hélices levógiras, triples hélices e incluso tetrahélices.

Los aspectos más importantes de las variaciones estructurales del ADN se encuentran en los mecanismos de reconocimiento molecular y de la manipulación por las proteínas. Incluso para las proteínas cuya única misión es enlazarse a una secuencia específica de ADN e impedir la transcripción, se requieren distorsiones de la estructura de ADN, como una flexión especial de la cadena a nivel local o un enrollamiento acompañando al enlace de la proteína [144]. Algunas proteínas muestran tendencia a la manipulación de la estructura de ADN para desarrollar sus funciones. Aunque una secuencia dada puede adoptar ciertas estructuras *in vitro*, nada nos asegura que lo mismo pueda ocurrir dentro de la célula. Por ello es de gran importancia la elucidación del papel biológico de cada variación estructural presente en el ADN. No hay ninguna duda de que, a pesar de que en muchas ocasiones se suponga que el ADN es una molécula rígida, posee una flexibilidad conformacional importante que puede y debe ser relacionada con su topología y con las interacciones con proteínas.

En este capítulo se ha desarrollado la determinación de la estructura de un pequeño oligonucleótido que presenta peculiaridades de gran interés cómo por ejemplo su peculiar estructura a altas concentraciones de Na^+ [145] o su elevada tendencia a formar complejos por intercalación [146]. Para ello se ha hecho uso de una de las técnicas más completas en la determinación de estructuras tridimensionales de biopolímeros ya explicada en capítulos anteriores como es la Resonancia Magnética Nuclear. Para obtener la estructura, además de las clásicas restricciones de distancias derivadas de los picos NOE se han introducido constantes de acoplamiento escalar, asociadas a los ángulos diedros, determinadas mediante simulación teórica de la estructura fina de los picos DQF-COSY de los protones del azúcar de cada nucleótido. La conjunción de ambas restricciones permitirá la obtención de una estructura en disolución de alta calidad para el oligonucleótido d(CCGCGG)$_2$ [179] y posibilitará la comparación de la misma con la estructura cristalina determinada mediante Rayos X.

6.1.1 Estructura del ADN

La importancia del ADN como material genético puede ser comprendida mediante un análisis detallado de la estructura del mismo. El aspecto crítico del ADN es la ordenación lineal que presente de cuatro elementos de repetición que son los nucleótidos. Con simplemente cuatro *bits* de información, el ADN codifica la información necesaria para el desarrollo de formas de vida complejas partiendo de una única célula. La estructura del ADN protege físicamente todos los átomos importantes de las bases de posibles modificaciones químicas producidas por el entorno. El patrón de puentes de hidrógeno de las base individuales se encuentra en la parte interior, en el centro de la doble hélice mientras que la cadena fosfatada se encuentra en la parte exterior. Desde este punto de vista general, el ADN presenta una superficie exterior uniforme y monótona que pueden reconocer muchas proteínas que se enlazarán al ADN de modo independiente de la secuencia. Estas proteínas pueden reconocer los aspectos generales de la estructura de ADN, como son los surcos menor y mayor. Para algunos procesos en que la secuencia juega un papel fundamental como la replicación o la transcripción, el ADN debe desenrrollarse para dar acceso a la información genética protegida en el centro de la doble hélice. Sin embargo, no todos los procesos dependientes de la secuencia conllevan el desenrrollamiento de la doble hélice. Una de las cuestiones pendientes de resolución relativas a este aspecto es cúales son las bases del reconocimiento ADN / proteína cuya acción es dependiente de una secuencia determinada si las bases implicadas no son accesibles desde el exterior.

En la actualidad, las técnicas de RMN en dos y tres dimensiones han permitido la elucidación de estructuras de varios fragmentos de ADN en disolución. Del análisis de las coordenadas cartesianas atómicas extraídas de la información de los experimentos de RMN se pueden observar una variedad significativa de estructuras helicoidales. De estas estructuras se deduce que la conformación helicoidal del ADN no es ni mucho menos uniforme y monótona, por lo que su parte exterior es “modo dependiente” de la secuencia. El ADN es una molécula dinámica que puede sufrir una amplia variedad de reordenaciones en su estructura secundaria.

Para entender la estructura de B-ADN (la más frecuente en disolución) y sus numerosas variaciones, es importante comprender los componentes individuales de la estructura del ADN. El ADN está compuesto de bases aromáticas (un anillo de purina o pirimidina), ribosas y grupos fosfato. Cada uno de estos elementos es importante en la estructura global del ADN

por motivos diferentes y cada uno es indispensable en el delicado equilibrio en que se encuentra el ADN.

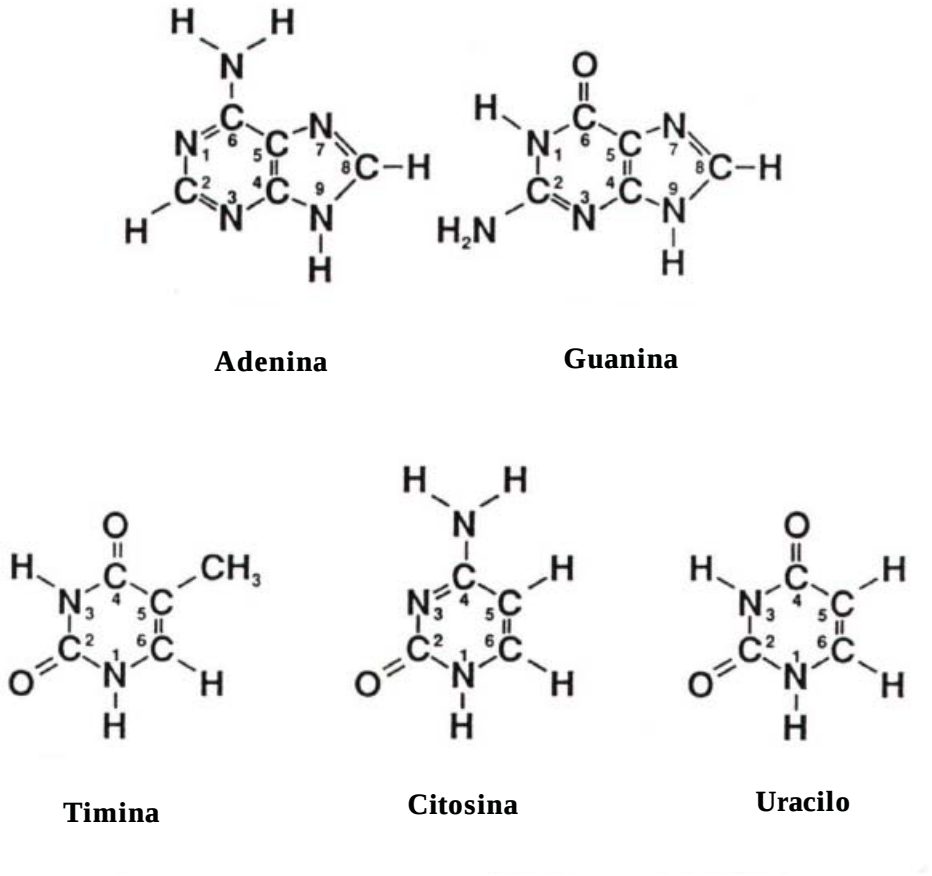


Figura 6.1: Estructuras de las bases púricas y pirimidínicas que forman parte del ADN.

Las bases aromáticas son los elementos que quedan en la parte interior de la molécula. Además de contener en sí la información genética gracias al modo en que se ordenan en la secuencia del ADN, son las principales responsables de la autocomplementariedad de la doble cadena de ADN. Hay dos tipos de bases aromáticas en los ácidos nucleicos que son las púricas y las pirimidínicas. En cada tipo hay dos bases y son complementarias a la par gracias a unos patrones de puentes de hidrógeno bien establecidos. Las bases púricas son la adenina (A) y la guanina (G) y las bases pirimidínicas son la timina (T) y la citosina (C) y son complementarias dos a dos (C-G y A-T) en la conformación estándar del ADN. Sus estructuras se pueden observar en la figura 6.1. La diferente situación de los grupos dadores y acep-

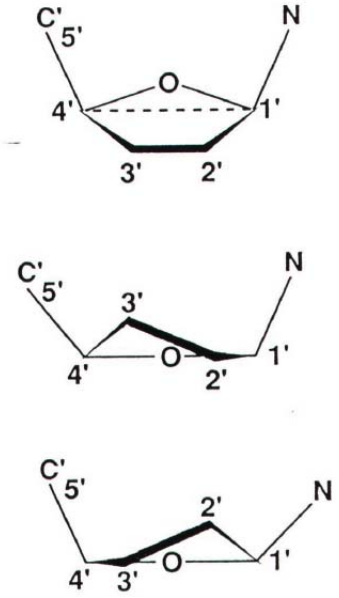


Figura 6.2: Representación esquemática de las conformaciones más habituales de los anillos de azúcar en ADN.

tores de puente de hidrógeno en cada una de las moléculas proporciona la base de la identidad secuencial requerida para transcribir la información genética. Los protones de los grupos amino actúan como dadores de puente de hidrógeno y los oxígenos de carbonilo y los nitrógenos aromáticos actúan como aceptores. La naturaleza aromática de los anillos proporciona la planaridad necesaria al conjunto para la organización de las bases dentro de la hélice en una zona de carácter claramente hidrófobico al tiempo que favorece las interacciones de apilamiento entre bases consecutivas aportando estabilidad adicional al conjunto [143].

Los azúcares que forman parte del ADN, constituyen una de las partes mas flexibles y dinámicas de la molécula (ver figura 6.2). El oxígeno suele situarse, en las conformaciones más favorables, fuera del plano del anillo de azúcar. Girando los carbonos C_2' y C_3' respecto a los otros átomos del azúcar resultan varias conformaciones del anillo de ribosa [147, 168, 148]. La conformación C_2' *endo* del anillo de ribosa se construye colocando por encima del plano formado por los otros carbonos el carbono C_2'. La conformacion C_3' endo se alcanza situando al carbono C_3' por encima del plano formado por los otros carbonos. El azúcar se une en su carbono C_5' a un grupo fosfato y en su carbono C_3' a otro grupo fosfato. Por el otro lado del azúcar se une a una base aromática nitrogenada en el carbono C_1' forman-

Dinucleótido	Ángulo de rotación
(AA)·(TT)	35.6 ± 0.1
(AC)·(GT)	34.4 ± 1.3
(AG)·(CT)	27.7 ± 1.5
(AT)·(AT)	31.5 ± 1.1
(CA)·(TG)	34.5 ± 0.9
(CC)·(GG)	33.7 ± 0.1
(CG)·(CG)	29.8 ± 1.1
(GA)·(TC)	36.9 ± 0.9
(GC)·(GC)	40.0 ± 1.2
(TA)·(TA)	36.0 ± 1.0

Tabla 6.1: Tabla de ángulos de rotación para las posibles combinaciones de dinucleótidos [149].

do un nucleósido. Los grupos fosfatos son los puentes de unión entre los azúcares y por lo tanto entre los nucleósidos y constituyen la cadena principal de la molécula al tiempo que aportan una polaridad adicional y hacen al ADN especialmente soluble. Una hebra de ADN empieza por un grupo hidróxilo en el carbono C_3' del primer nucleótido y termina en un fosfato unido al carbono C_5' del último nucleotido, siguiendo la hebra complementaria el sentido opuesto.

A nivel local, hay una gran variedad de parámetros que permiten la identificación y clasificación de las peculiaridades estructurales de cada estructura de ADN característica. Estos parámetros locales implican en la mayoría de los casos conformaciones del azúcar, distancias interatómicas características, ángulos diedros respecto a planos de referencia, etc. Estos parámetros, además de la lógica influencia en las estructuras particulares de los nucleótidos, tienen una importante repercusión en la morfología propia de la hélice. Por ello, los parámetros locales y los parámetros globales de la hélice de ADN están íntimamente relacionados. Datos estructurales como el número de residuos por vuelta, el incremento en longitud por residuo, la longitud de una vuelta, el ángulo del par de bases respecto al eje de la hélice (*tilt*) o el propio eje de la hélice, están relacionados directamente con la conformación de los azúcares constituyentes o las distancias entre los átomos de fósforo de un par de bases.

Lógicamente, la estructura local de un nucleótido en el seno de una hebra de ADN depende en gran medida de su entorno, es decir, de los nucleótidos que le rodean. Así, la conformación local de los nucleótidos es

Parámetro	A-ADN	B-ADN	Z-ADN
Sentido de la hélice	Dextro	Dextro	Levo
Residuos por vuelta	11	10.5	12
Incremento Axial (Å)	2.55	3.4	3.7
Helix Pitch (^{o})	28	34	45
Inclinación del par de bases (^{o})	20	-6	7
Rotación por residuo (^{o})	33	34.3	-30
Diámetro máximo de hélice (Å)	23	20	18
Configuración enlace glicosídico			
dA, dT, dC	anti	anti	anti
dG	anti	anti	syn
Conformación Azúcar			
dA, dT, dC	C_3' endo	C_2' endo	C_2' endo
dG	C_3' endo	C_2' endo	C_3' endo
Distancia entre fosfatos interhebra (Å)			
dA,dT,dC	5.9	7.0	7.0
dG	5.9	7.0	5.9

Tabla 6.2: Tabla de parámetros estructurales típicos de las conformaciones más representativas del ADN.

función, lógicamente, de la secuencia y por lo tanto la conformación global de la cadena de ADN también lo es. De hecho, algo tan simple como el ángulo de giro de un par de bases consecutivas, en el que sólo dos nucleotidos se ven implicados, varía entre las diferentes combinaciones de nucleótidos posibles como se puede apreciar en la tabla 6.1. Se pueden observar variaciones de hasta 10^o desde el valor mínimo al máximo de giro de par de bases (*twist*). La forma del nucleótido se verá afectada, lógicamente por el giro del par de bases. De hecho, un oligonucleótido como el que vamos a estudiar, de 6 elementos, contiene 1024 (4^5) posibilidades de pares de bases consecutivos. Así, considerando sólo los efectos estructurales de la combinación de dos pares de bases tenemos 1024 posibles resultados globales para la cadena. Es fácil comprender, que la influencia de los nucleótidos adyacentes al par de bases analizado no es despreciable por lo que las posibilidades aumentan de modo exponencial.

Aunque hemos visto que la variedad de conformaciones que puede adoptar el ADN es inmensa, se pueden distinguir varios tipos de conformaciones básicas. La forma B es la predominante en disolución y condiciones fisiológicas estándar. Una característica peculiar de la forma B de ADN es

la presencia de dos surcos distintos claramente diferenciados: uno mayor y uno menor. Estos dos surcos suponen dos superficies diferentes en las que las proteínas pueden complejarse. En cada uno de los surcos, los grupos químicos de las bases que son accesibles son diferentes aunque en ningún caso se puede acceder a los puentes de hidrógeno. Sin embargo, la superficie de los puentes de hidrógeno según el patrón de Hoogsteen es accesible en el surco mayor. La forma A ADN presenta unos surcos menos profundos y por lo tanto los grupos químicos de las bases son menos accesibles al disolvente y a las proteínas complejantes. Otra diferencia fundamental entre la forma A y la forma B de ADN es que mientras en la primera la conformación del azúcar es C_3' endo en la segunda es C_2' endo. La presencia de ADN en forma A en organismos vivos es bastante poco frecuente y siempre se da en muy pequeños fragmentos. Aunque hay otras muchas variantes de la estructura de ADN (C, D y T-ADN), la más peculiar de las restantes en doble hélice es la forma Z-ADN [150]. La forma Z-ADN es levógira por lo que es muy diferente al resto de formas dextrógiras. Se puede dar en regiones de alternancia entre purinas y pirimidinas bajo ciertas condiciones incluyendo alta concentración de sal, la presencia de ciertos cationes divalentes o en zonas de superenrrollamiento de ADN [151]. En la tabla 6.2 se pueden observar diferentes aspectos estructurales propios de cada una de las formas más representativas de ADN [148].

6.1.2 d(CCGCGG)$_2$

Aunque ya hemos visto que la conformación Z de ADN es la más peculiar de las tres más representativas, es posible que, bajo determinadas condiciones se dé dicha confomación alternada con alguna de las otras. Algunos oligonucleótidos tienen la capacidad de adoptar estructuras levógiras y dextrógiras dependiendo de las condiciones experimentales. Como se puede apreciar en la tabla 6.2, la guanina presenta ciertas características estructurales dentro de las estructuras en Z-ADN que hacen bastante interesante su influencia en dichas conformaciones. Por otra parte, en la tabla 6.1 se puede apreciar que el par dGC es el que presenta mayor rotación mientras que el dCG es de los que menos rotación muestran [153]. Estas peculiaridades asociadas al par CG hacen a los oligonucleótidos con gran proporción de dichos pares muy interesantes para el estudio estructural. Además, estudios de dicroísmo circular realizados sobre poli dG-dC [152] muestran que dichos oligonucleótidos existen en conformaciones dextrógiras bajo condiciones de baja salinidad (alrededor de 1 molar en Na^+) y en conforma-

ciones levógiras bajo condiciones de concentración salina elevada (4 M en Na^+).

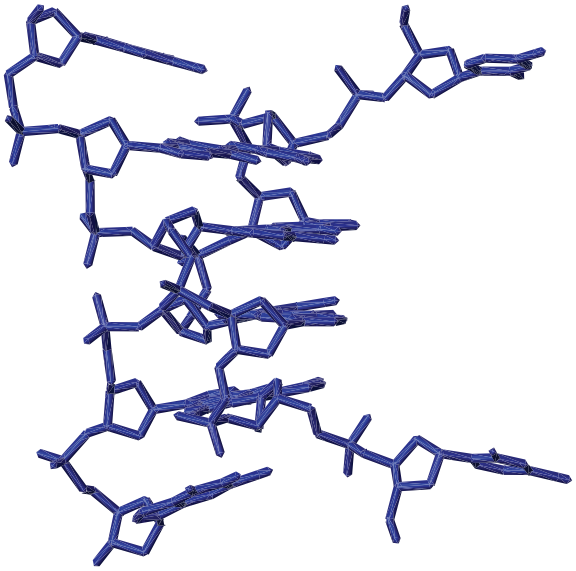


Figura 6.3: Representación de la estructura tridimensional del oligonucleótido $d_(CCGCGG)_2$ determinada por difracción de Rayos X [145] en concentración de Na^+ 4 M.

Estudios más recientes han mostrado más peculiaridades en fragmentos de ADN con gran proporción de uniones dG-dC. En concreto, se ha obtenido la estructura cristalina del oligonucleótido d(CCGCGG)$_2$ bajo alta concentración salina (4 M en Na^+) y se ha determinado que su estructura presenta una conformación de Z-ADN en el tetrámero central CGCG con los pares de bases de los residuos terminales expuestos al exterior como se puede apreciar en la figura 6.3 [145]. Este fragmento de ADN es parte de la mutación crítica de un gen que produce formas hereditarias de retraso mental (síndrome de fragilidad en X). Se ha encontrado que la disposición de las bases hacia el exterior en este fragmento de ADN se estabiliza mediante la formación de tetrámeros asimétricos de ADN y la formación de puentes de hidrógeno entre las bases expuestas de dúplex de ADN adyacentes. Se obtienen así, según las condiciones de cristalización, tetrámeros centrales alternantes CGCG en conformación Z [154] mientras que la citosina inicial se orienta hacia el exterior formando un par de Watson-Crick con la guanina del hexámero adyacente, posibilitando un apilamiento de unidades tetrámericas. Esta estructura necesita una estabilización adicional mediante la coordinación a un ión de Na^+ por parte de la cadena fosfatada de los residuos centrales. La estructura total parece tener un papel importante

en la recombinación del ADN [155].

Para elucidar la posible formación de dichas estructuras irregulares en disolución se han realizado experimentos de RMN sobre el mismo oligonucleótido en diferentes concentraciones salinas. A partir de los mismos, se ha determinado la estructura para su comparación con la estructura cristalina y su posterior análisis.

6.1.3 RMN de ácidos nucleicos

La aplicación de las técnicas de RMN a los ácidos nucleicos tiene lugar un año después del histórico descubrimiento de Watson y Crick. Aunque la estructura de los genes es demasiado grande (pesos moleculares de 200000 a 800000 gr / mol) para los estudios de RMN en disolución, las principales secuencias de control en el ADN , tales como los operadores, consistentes en oligonucleótidos de 10 a 20 pares de bases son susceptibles de ser estudiadas mediante esta técnica. Aunque el análisis de los estudios de RMN sobre oligonucleótidos puedan parecer más sencillos *a priori* que los mismos estudios realizados sobre proteínas, el ADN presenta diferentes problemáticas a resolver en la resolución de estructuras como puedan ser la poca información de distancias de largo alcance o lo poco restrictiva que ésta puede llegar a ser. El ADN tiene sólo 4 bases aromáticas (5 incluyendo el uracilo) que, comparadas con los 20 aminoácidos de las proteínas hace que la asignación de las resonancias sea en principio menos complicada que en estas últimas. Sin embargo, no todas las distancias que se pueden calcular mediante los espectros NOE determinan de forma unívoca un tipo de conformación de ADN. De hecho, sólo algunas distancias interprotónicas son diferenciadoras entre los diferentes tipos de ADN y, a menudo, el error experimental hace poco fiable el análisis mediante distancias exclusivamente. Las distancias interprotónicas que se pueden calcular mediante los datos de RMN en ADN se restringen a pares de base adyacentes. En la figura 6.4 se pueden ver algunas gráficas de distancias típicas de ADN frente al ángulo de pseudorotación P.

El protocolo de asignación de resonancias en ácidos nucleicos propuesto por Wüthrich [37] es bastante similar al de asignación de proteínas. El primer paso en la asignación secuencial de la resonancias de un espectro de un oligonucleótido es la identificación de los sistemas de espín. Mediante la comparación con los desplazamientos químicos habituales para cada tipo de protón se puede realizar una parte de la identificación de sis-

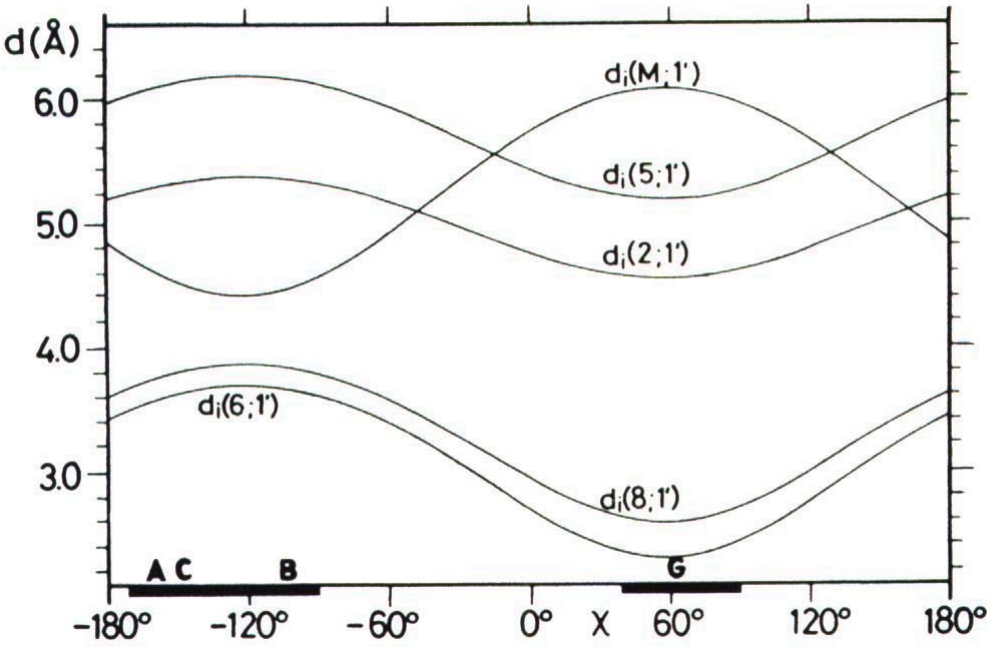


Figura 6.4: Variación de algunas distancias interprotónicas con el ángulo de pseudorotación P [37].

temas de espín. Hay dos sistemas de espín independientes dentro de cada nucleótido aislado. El primero comprende el azúcar y el segundo a la base aromática nitrogenada. Se puede localizar el mismo mediante el DQF-COSY y el TOCSY de modo similar a como se hace en proteínas siguiendo las conectividades de acoplamiento escalar entre H_1' y H_2'/H_2'', entre estos y H_3', entre este y H_4' y entre este y H_5'/H_5'' en cada unidad de azúcar. Las conectividades entre los protones H_5 y H_6 de las citosinas y entre los protones H_6 y CH_3 de las timinas en las bases pirimidínicas se puede observar también en los mismos espectros. Los protones de la base y el azúcar de un mismo nucleótido se pueden asociar mediante las conectividades NOE de acoplamiento dipolar entre los protones H_1'/H_2' del azúcar y H_8/H_6 de la base. La signación secuencial se pude realizar mediante las conectividades NOE entre los protones $H_1'/H_2'/H_2''$ de un nucleótido y los protones H_8/H_6 del siguiente y mediante las conectividades entre los protones H_8/H_6 de un nucleótido y los protones H_5/CH_3 del siguiente. De este modo, se cierra el ciclo de un residuo al siguiente y la asignación secuencial completa puede ser realizada. En principio, debido a que los experimentos pueden ser realizados en D_2O gracias a la ausencia de protones intercambiables en los

protones implicados en la signación secuencial, la asignación completa podría ser realizada mediante las conectividades NOE exclusivamente para pequeños oligonucleótidos. La asignación de los protones imino así como la asignación de los protones H_2 de la adenina se realizan secuencialmente mediante experimentos NOE en H_2O.

Los experimentos heteronucleares implicando al ^{31}P como puedan ser el COSY heteronuclear (HETCOR) con detección en ^{31}P, el HMQC o el TOCSY heteronuclear con detección en ^{31}P permiten la asignación de constantes de acoplamiento escalar J^{HP} y resultan especialmente importantes para oligonucleótidos grandes (más de 25 pares de bases) en los que se puede emplear otros isótopos para facilitar la asignación.

6.1.4 Simulación de espectros de RMN

Las constantes de acoplamiento escalar ^3J pueden servir para delimitar cuatro de los seis ángulos de torsión que determinan la conformación del esqueleto principal de un ácido nucleico, y pueden, simultáneamente, localizar la presencia de flexibilidad conformacional [156]. La conformación global del azúcar (δ) puede ser definido por tres constantes de acoplamiento homonuclear ^1H-^1H como son $^3J_{H1'-H2'}$, $^3J_{H2'-H3'}$ y $^3J_{H3'-H4'}$. Con los ángulos de rotación individuales que forman parte de la estructura del azúcar ocurre lo mismo. Por ejemplo, el ángulo de torsión γ en torno al enlace $C_{5'} - C_{4'}$ puede ser delimitado mediante las dos constantes de acoplamiento escalar homonucleares $^3J_{H4'-H5'}$ y $^3J_{H4'-H5''}$.

La determinación de constantes de acoplamiento escalar ^3J mediante las técnicas habituales utilizadas en proteínas (por ejemplo, E.COSY) se complica bastante por las deformaciones, respecto a las estructuras teóricas de los picos cruzados, originados por las intensas constantes de acoplamiento pasivas, p.e. $H_{1'} - H_{2''}$,$H_{1'} - H_{3'}$, $H_{2'} - H_{3'}$, $H_{2''} - H_{3'}$ y $H_{3'} - H_{4'}$ en el pico cruzado DQF-COSY $H_{1'} - H_{2'}$. Por ello, se suele utilizar de modo más frecuente la simulación de la estructura hiperfina de los picos DQF-COSY para determinar las constantes de acoplamiento escalar del sistema de espín completo del azúcar [1].

Hoy en día los espectros COSY se adquieren en modo sensitivo a la fase y con alta resolución digital casi en su totalidad. La estructura hiperfina de los picos cruzados COSY puede así ser resuelta y analizada para proporcionar información sobre la simetría del sistema de espín y el valor de las constantes de acoplamiento. Para los espectros de primer orden, la

estructura fina de los picos puede ser calculada mediante el formalismo de operadores [157]. En algunos experimentos, como el DQF-COSY, algunas reglas simples se pueden utilizar para desarrollar representaciones esquemáticas de los patrones de cada pico. Sin embargo, estas reglas simples pierden eficacia en presencia de fuertes acoplamientos y el cálculo de los picos requiere procesos más elaborados. Por otra parte, en los espectros experimentales, la apariencia de los picos depende de factores externos al sistema de espín como la anchura de línea, el tiempo de adquisición, los parámetros de procesado o la resolución digital [158].

El sistema de espín del anillo de desoxiribosa incluye siete protones entre los cuales hay dos pares de protones metilénicos fuertemente acoplados 2'CH_2 y 5'CH_2. Los protones $H_{3'}$, $H_{5'}$ y $H_{5''}$ están débilmente acoplados con los espínes ^{31}P de los grupos fosfodiéster [166]. En la práctica, los picos cruzados entre $H_{1'}$, $H_{2'}$, $H_{2''}$ y $H_{1'}$ [1] son los más interesantes para el estudio de las conformaciones de ADN. Las características del espectro que dependen del acoplamiento entre los protones $H_{4'}$ y $H_{5'}$ y $H_{5''}$ son de utilización limitada debido a los solapamientos mutuos que suelen presentarse. Así, los estudios de simulación de picos cruzados DQF-COSY se centran en los picos cruzados $H_{1'} - H_{2'}$, $H_{1'} - H_{2''}$, $H_{3'} - H_{2'}$ y $H_{3'} - H_{2''}$. Las simulaciones se basan en las siguientes consideraciones básicas. Se supone un fuerte acoplamiento entre los protones geminales $H_{2'}$ y $H_{2''}$, para los cuales se suele suponer una constante de acoplamiento alrededor de -14 Hz [167]. Este fuerte acoplamiento se manifiesta especialmente en el hecho de que, en la mayoría de picos cruzados del azúcar, los componentes de la estructura fina más próximos al elemento con el que el acoplamiento es fuerte son más intensos que el resto. Si el acoplamiento fuera débil, los picos cruzados deberían mostrar una simetría próxima a D_2. En el caso de todas las demás constantes de acoplamiento, el valor de J es muy inferior a las diferencias en desplazamiento químico entre los protones acoplados.

Como ya se ha comentado, el anillo de desoxiribosa existe principalmente en dos conformaciones preferidas, C2'endo-C3'exo (S), presente mayoritariamente en el B-ADN, y C3'exo-C2'endo (N) [169], conformación preferida en el A-ADN. A menudo, se suele trabajar sobre estas dos conformaciones suponiendo una u otra o bien una combinación de ambas asociada a una rápida interconversión entre los dos confórmeros. De este modo, la constante de acoplamiento escalar en una conformación intermedia con un porcentaje X_N en conformación N sería,

$$J_{ik}^{obs} = X_N J_{ik}^N + X_S J_{ik}^S, \qquad X_N + X_S = 1 \tag{6.1}$$

donde J_{ik}^N y J_{ik}^S son las constantes de acoplamiento entre los protones i y k en los confórmeros N y S respectivamente, y X_N y X_S las contribuciones de los confórmeros N y S respectivamente al intercambio dinámico del azúcar.

Un aspecto práctico de gran utilidad en la simulación es el hecho de que los picos cruzados $H_{1'} - H_{2'}$ y $H_{1'} - H_{2''}$ están en diferente región del espectro que los picos cruzados $H_{3'} - H_{2'}$ y $H_{3'} - H_{2''}$, con lo que las posibles similaridades entre las estructuras hiperfinas de los picos cruzados de $H_{1'}$ y $H_{3'}$ nunca producirán ambigüedades en la asignación de resonancias. De este modo, la estructura fina de los cuatro picos seleccionados para el estudio manifiesta de modo claro las diferentes conformaciones del anillo de desoxiribosa [1]. La comparación de las simulaciones teóricas con los datos experimentales permitirá la identificación de la conformación del azúcar.

6.2 Material y métodos

6.2.1 Preparación de la muestra

La muestra de d-(CCGCGG)$_2$ fué sintetizada en fase sólida y purificada mediante HPLC de fase reversa con una pureza superior al 95 %. La concentración de las disoluciones utilizadas en los experimentos de RMN fué determinada mediante espectroscopía de absorción Ultravioleta-Visible a 25^o utilizando el coeficiente de extinción para el ADN [170] $\epsilon_{260} = 6400 M^{-1} cm^{-1}$. La doble helipticidad del ADN fué confirmada por su hipercromicidad y por la típica transición de su desnaturalización térmica [171]. Las muestras para los experimentos de RMN se prepararon en disolución tamponada a pH 7 con tampón fosfato 50 mM, con NaCl 50 mM (para controlar la fuerza iónica) y un 0.1 % de azida sódica para eliminar las posibles bacterias nucleófagas que destruirían el ADN. Las disoluciones se realizaron en agua pesada al 98 %, se liofilizaron con N_2 líquido, para evitar la presencia de O_2, que disuelto crea una atmósfera paramagnética que puede interferir en los experimentos, repitiendo este proceso dos veces, y por último se disolvió el producto de liofilización en D_2O al 100 %.

Para comprobar la influencia de la fuerza iónica en la conformación del d-(CCGCGG)$_2$ se realizaron experimentos a fuerza iónica elevada mediante

Campo (MHz)	Experimento	[NaCl] Mezcla	Tiempo de
400	DQF-COSY	0.05,2	
	TOCSY	0.05	40,80,120
	TOCSY	2	40,80,120
	NOESY	0.05	50,100,200
	NOESY	2	50,100
600	DQF-COSY	2	
	NOESY	2	100,200

Tabla 6.3: Tabla de experimentos de RMN realizados sobre la muestra de $d(CCGCGG)_2$.

la adición de NaCl hasta hacer la disolución 4 M en NaCl. Sin embargo, la liofilización de dichas muestras presentó algunas dificultades por lo que la concentración final de NaCl en dichas muestras resultó de 2 M.

6.2.2 Experimentos realizados

Los espectros de RMN fueron obtenidos en un espectrómetro Unity de Varian a 400 MHz. Se acumularon espectros DQF-COSY, NOESY y TOCSY a diferentes tiempos de mezcla. En la tabla 6.3 se pueden observar todos los experimentos 2D que se realizaron. Los tiempos de mezcla de los experimentos NOESY fueron de 50, 100 y 200 ms, mientras que para los experimentos TOCSY fueron de 40, 80 y 120 ms. Se registraron 512 incrementos de t_1 con 2048 puntos recogidos a lo largo de t_2 en cada uno para todos los experimentos realizados en 2D. La anchura espectral abarcada fué de 3900 Hz en ω_1 y 7800 Hz en ω_2. La fase de las secuencias estándar que ofrece el espectrómetro empleado para los experimentos 2D a 400 MHz está ligeramente desplazada produciendo un hundimiento de las señales cerca de la señal del agua. No se ha tenido en cuenta debido a que la señal del agua en nuestro espectro no era demasiado intensa y porque los picos de más interés estaban lejos de dicho hundimiento.

El tratamiento de datos se realizó multiplicando todas las señales por una función campana sinusoidal desplazada y expandiendo por ajuste a 2048 puntos en ω_1 y 4096 en ω_2 antes de realizar la transformada de Fourier para todos los espectros salvo el NOESY de 50 ms que se expandió por ajuste a 1024 puntos en ω_1 y 2048 en ω_2. Este tratamiento dió una resolución digital de 3.9 Hz/punto en ω_1 y 1.9 Hz/punto en ω_2 (1.9 y 0.9 respectiva-

mente para el NOESY a 50 ms).

Se obtuvieron espectros adicionales en un espectrómetro Brucker a 600 MHz sobre las muestras con concentración de NaCl 2 M con objeto de conseguir mayor resolución en los picos solapados de las guaninas terminales. Se aplicó un protocolo de procesado similar al desarrollado sobre los experimentos a 400 MHz.

6.2.3 Simulación de picos DQF-COSY

Las simulaciones de los picos cruzados DQF-COSY se realizaron con los programa SPHINX y LINSHA [172]. El fichero de datos necesario para el programa SPHINX incluye los parámetros del sistema de espín y la secuencia de pulsos y ciclos de fases del experimento. SPHINX realiza un cálculo de matriz de densidad y produce una lista de transiciones, caracterizadas por las frecuencias de resonancia ω_1 y ω_2, las fases y las intensidades. Esto produce un espectro simplificado tal y como se puede obtener para sistemas de espín débilmente acoplados utilizando reglas simples para describir los términos en fase y en antifase debidos a los acoplamientos pasivos y activos respectivamente.

El programa LINSHA calcula la matriz S(ω_1,ω_2) que representa el espectro en el dominio de frecuencias. Los datos de entrada consisten en el espectro simplificado calculado por SPHINX y un conjunto de parámetros para el procesado bidimensional de la señal.

La anchura espectral se ha situado en 6 Hz en ω_1 y 4 Hz en ω_2. La transformación de los picos simulados se ha realizado mediante la aplicación de una función seno cuadrado con un desfase de 80^o. Para la presentación de los picos simulados se han realizado mapas de contornos a niveles en incrementos exponenciales con trazado en línea contínua para las señales positivas y en línea discontínua para las señales negativas.

Se ha comparado empíricamente los picos simulados y los picos experimentales de los espectros DQF-COSY y se han determinado los conjuntos de constantes de acoplamiento ^3J que mejor representaban las formas obtenidas en los espectros. Las constantes de acoplamiento escalar J han sido variadas en incrementos de 0.3 Hz en la búsqueda del valor óptimo. Se varió inicialmente las constantes $^3J_{1'2'}$ y $^3J_{2'3'}$ en la búsqueda de los valores por ser las más sensibles a la forma de los picos simulados y a continuación se modificaron progresivamente el resto hasta la máxima concordancia con los datos experimentales.

6.2.4 Restricciones estructurales

Se obtuvieron distancias interprotónicas de las resonancias de los experimentos NOESY a 50, 100 y 200 ms clasificándose en 4 grupos. La clasificación de las señales se hizó mediante un doble criterio de intensidad y tiempo de mezcla requerido para la observación del pico. Las resonancias observadas en cada espectro se clasificaron en fuertes, medias o débiles según la intensidad relativa a los picos cruzados de los protones H_5 y H_6 de las citosinas que por estar en un anillo aromático deben estar a una distancia fija de 2.4 Å. Las resonancias que aparecieron como débiles en el espectro a tiempo de mezcla de 200 ms y no aparecieron en el resto de espectros se clasificaron como D. Las resonancias que aparecieron en el espectro de tiempo de mezcla 100 ms con intensidades débiles y medias-fuertes se clasificaron C y B respectivamente siempre y cuando aparecieran en el de 200 ms y no aparecieran como resonancias fuertes a 50 ms de tiempo de mezcla. Las resonancias fuertes en el espectro de 50 ms de tiempo de mezcla se clasificaron como A. Los límites de distancia asociadas a cada tipo NOE fueron 1.8-2.5 Å, 2.4-3.5 Å, 3.2-4.2 Å y 4.0-5.0 Å para las distancias A, B, C y D respectivamente.

El margen concedido a las distancias responde a que para mantener los átomos en unas posiciones acordes con la observación de distancias en RMN no es necesario fijarlos en el espacio mediante una distancia única. El intervalo de distancias concedido corresponde a la idea de que en lugar de una distancia de equilibrio, existe un intervalo de distancias de equilibrio interprotónica en el que la función de potencial penaliza las restricciones. De este modo, permitimos a los átomos oscilar en torno a su posición de equilibrio, como corresponde a cualquier sistema a temperatura ambiente. Las restricciones de distancias obtenidas se clasifican tal y como se ve en la tabla 6.4.

Mediante la determinación de las constantes de acoplamiento y los ángulos de pseudorotación P de cada uno de los azúcares de la molécula se pueden evaluar los ángulos endocíclicos que forman parte de la desoxiribosa (ϕ_1 a ϕ_4) [1] mediante la expresión:

$$\phi_j = \phi_m cos(P + 4\pi\frac{(j-2)}{5}) \qquad j = 0 - 4 \tag{6.2}$$

donde ϕ_j corresponde a uno de los ángulos endocíclicos que definen la conformación de cada desoxiribosa mostrados en la tabla 6.5, ϕ_m es la amplitud de empaquetamiento de cada anillo del azúcar (ϕ_m vale 38^o para

	$d_i(1';4')$	$d_i(2'';4')$	$d_i(1';6,8)$	$d_i(6,8;2')$	$d_s(2';6,8)$	$d_s(3';6,8)$
C_1	B	C	C	B	D	-
C_2	A	C	C	B	C	C
G_3	A	B	C	C	D	D
C_4	A	B	-	-	C	D
G_5	B	C	C	D	D	-
G_6	A	C	-	-	-	-

Por la simetría de la molécula sólo hay 6 residuos diferenciados

Tabla 6.4: Clasificación de las distancias derivadas de las resonancias NOE según se explica en el texto para algunos de los picos más importantes de los espectros NOESY.

ϕ_1	$O_{4'} - C_{1'} - C_{2'} - C_{3'}$
ϕ_2	$C_{1'} - C_{2'} - C_{3'} - C_{4'}$
ϕ_3	$C_{2'} - C_{3'} - C_{4'} - O_{4'}$
ϕ_4	$C_{3'} - C_{4'} - O_{4'} - C_{1'}$

Tabla 6.5: Tabla de ángulos endocíclicos de la desoxiribosa.

anillos de cinco miembros de azúcares [166]) y P es el ángulo de fase de pseudorotación.

Especialmente en este fragmento de ADN se ha observado una elevada incertidumbre en las constantes de acoplamiento escalar debida principalmente a la complejidad de los sistemas de espín y a la gran flexibilidad mostrada en el análisis cualitativo de la estructura fina de los picos DQF-COSY. Por ello sólo algunos ángulos de rotación del azúcar han podido ser determinados de modo casi absoluto tal y como se puede apreciar en los resultados de simulación (ver tabla 6.10).

6.2.5 Cálculo de estructuras

Se calculó un conjunto de estructuras mediante la aplicación de simulación de dinámica molecular con restricciones a 300 K utilizando el programa DISCOVER de Molecular Simulations Inc. [173] y el campo de fuerzas AMBER [124]. La elección del programa y el campo de fuerzas se debió a que han sido utilizados ampliamente en numerosos estudios sobre ácidos nucleicos [90] e interacciones entre biopolímeros y fármacos. Además, este campo de fuerzas ha sido modificado en nuestro grupo para las interacciones de

complejos [146, 164]. Se han reescalado las interacciones del potencial AMBER p1_4 a 0.5 según la recomendación de los autores para el tratamiento de biopolímeros. Todos los cálculos han sido realizados en un medio simulando el disolvente de modo implícito mediante una constante dieléctrica dependiente de la distancia $\epsilon = 1 * r$. Se consideraron todas las interacciones no enlazantes de la molécula mediante la aplicación de un límite en distancia superior al tamaño de la molécula.

Se realizaron cálculos partiendo de las estructuras ideales promedio obtenidas por cristalografía de rayos X de las tres formas más habituales de ADN, A-ADN, B-ADN y Z-ADN. En primer lugar se aplicó una minimización de 100 pasos de "steepest descent" seguidos de 1000 pasos de gradientes conjugados con restricciones de diedros y distancias. A continuación, se simularon 100 ps de dinámica molecular restringida a 300 K despues de una etapa previa de calentamiento progresivo de 10 ps. Durante la dinámica molecular restringida se extrajeron estructuras intermedias cada 10 ps que fueron relajadas mediante minimización de energía con 100 pasos de "steepest descents" y 5000 de gradientes conjugados con restricciones. A las restricciones de diedros se les aplicó una constante de fuerza de 200 $Kcal/mol^o$ y a las de distancia 100 $Kcal/mol$ Å^{-2} con un potencial armónico de fondo plano.

Se realizaron cálculos con el conjunto completo de restricciones, ángulos diedros y distancias, y sólo con diedros. De este modo, mediante los cálculos con distancias y diedros se ha obtenido la estructura del d-$(CCGCGG)_2$ en disolución consistente con toda la información estructural recogida en los espectros y mediante los cálculos sólo con diedros se exploró la posibilidad de obtener estructuras anómalas con pequeños cambios conformacionales en las conformaciones de los anillos de desoxiribosa, que pudieran constituir pasos intermedios en la formación de los tetrámeros.

6.2.6 Análisis de las estructuras. NDBSTAT.

Para el análisis de los parámetros estructurales de las estructuras obtenidas se ha utilizado un programa desarrollado en nuestro grupo por V. Esteve llamado NDBSTAT [174]. El programa NDBSTAT permite conocer y evaluar las peculiaridades de una estructura de ADN, en simple, doble o triple hebra a partir de sus coordenadas cartesianas en el formato estándar del PDB (*Protein Data Bank*) y del NDB (*Nucleic acid Data Bank*). Además de medir los ángulos y distancias más reveladores en una estructura de ADN tanto

en las bases nitrogenadas como en los anillos de azúcar, el programa es capaz de localizar los ejes de las hélices a nivel local y global y, por lo tanto, de evaluar los parámetros geométricos relativos al sistema de coordenadas de la hélice.

El método de cálculo de la hélice global es bastante innovador ya que, en lugar de realizar un promedio de los ejes locales, como se suele hacer en la mayoría de programas de análisis (NEWHELIX92 [176], por ejemplo), la determinación del eje global es independiente de la de los ejes locales. Se busca el eje z que haga menor la desviación de la proyección de los puntos de la hélice respecto a la proyección de una hélice perfecta.

El cálculo de otros parámetros globales de la cadena de ADN también se ha realizado buscando la máxima precisión y adecuación a las características intrínsecas a los ácidos nucleicos. Tal es el caso de parámetros como la profundidad de surco y curvatura de la hélice.

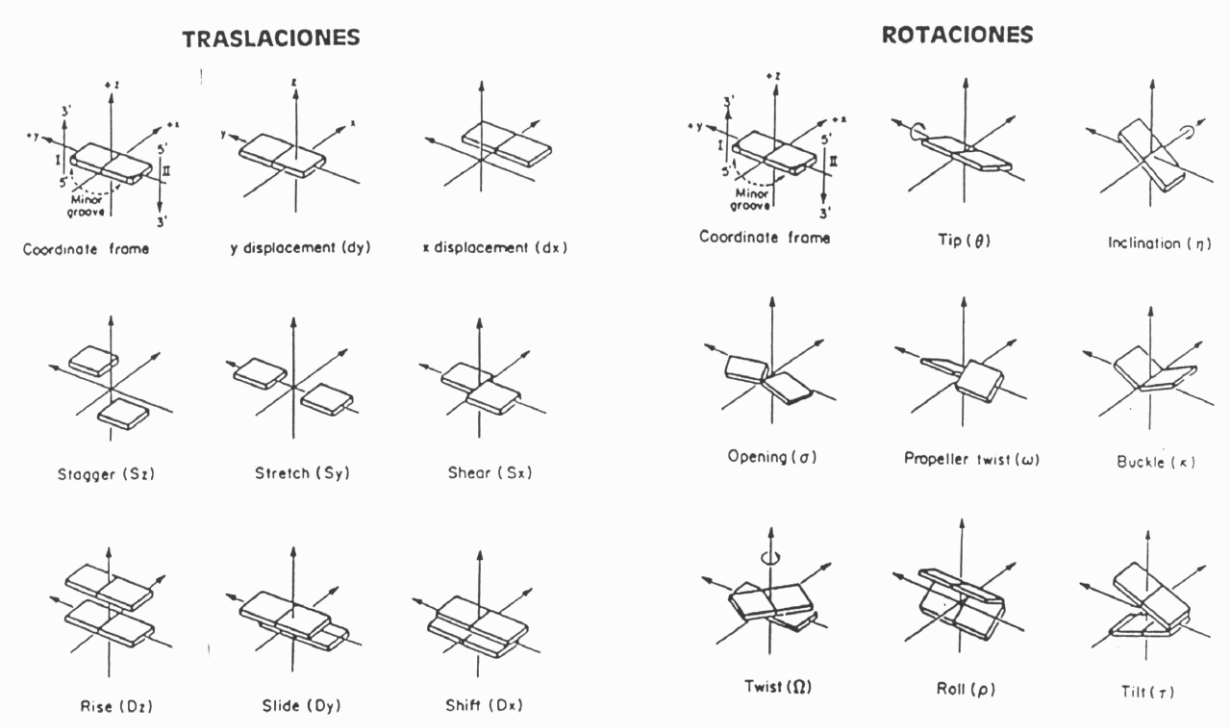


Figura 6.5: Algunos de los parámetros geométricos calculados por NDBSTAT tal como define la IUPAC.

El programa NDBSTAT cumple rigurosamente los criterios establecidos por la IUPAC en la medición de los parámetros establecidos por la misma. Sin embargo, también calcula algunos parámetros de gran interés que no

han sido regulados por la misma, así como un conjunto de distancias interatómicas de gran intereés en el análisis de la forma que toma cada cadena de ADN. La flexibilidad e interactividad del programa permite utilizarlo de un modo directo sobre los resultados de los cálculos. Además, la posibilidad de aplicación de un protocolo automático estándar permite la automatización de los análisis.

6.3 Resultados y discusión

6.3.1 Asignación secuencial

Los espectros DQF-COSY del oligonucleótido d(CCGCGG)$_2$ mostraron que la molécula mantenía una completa simetría. La aparición de seis únicos sistemas de espín diferenciados aportaron la evidencia de que este oligonucleótido autocomplementario mantiene una total simetría en su estructura ya que de no haber sido así, el número de sistemas de espín observado debiera haber sido superior a 6. Esta simetría se puede observar tanto en los espectros DQF-COSY (ver figura 6.7) como en los espectros 1D (ver figura 6.6). La aparición de tres únicos picos cruzados NOESY para los protones aromáticos H_5 y H_6 de las citosinas corrobora la simetría ya observada en los sistemas de espín de los azúcares (ver figura 6.8).

Se llevó a cabo la asignación de los sistemas de espín de los azúcares mediante la metodología propuesta por Wüthrich [37] y explicada en la sección 6.1.3. En la figura 6.7 se puede observar la asignación de dichos sistemas de espín. La asignación secuencial de los sistemas de espín se llevó a cabo mediante las conectividades NOE entre los protones aromáticos H_6 y H_8 de las citosinas y las guaninas respectivamente con los protones del azúcar del oligonucleótido anterior $H_{2'}$ y $H_{2''}$ como puede apreciarse en la figura 6.9.

La presencia de los picos cruzados DQF-COSY $H_{2'} - H_{3'}$ para los residuos terminales C1 y G6 en el espectro indica interconversión en los mismos entre las formas N y S del anillo de desoxiribosa. Por otra parte, la ausencia de picos secuenciales $H_{2'}G - H_5C$ indica que la forma en disolución no es Z-ADN, por lo que se puede suponer que la estructura en disolución difiere de la de rayos X.

En la tabla 6.6 se pueden observar los desplazamientos químicos de todos los protones asignados del oligonucleótido d(CCGCGG)$_2$ [179]. Debido a que sólo se adquirieron espectros en D_2O para las muestras a baja concen-

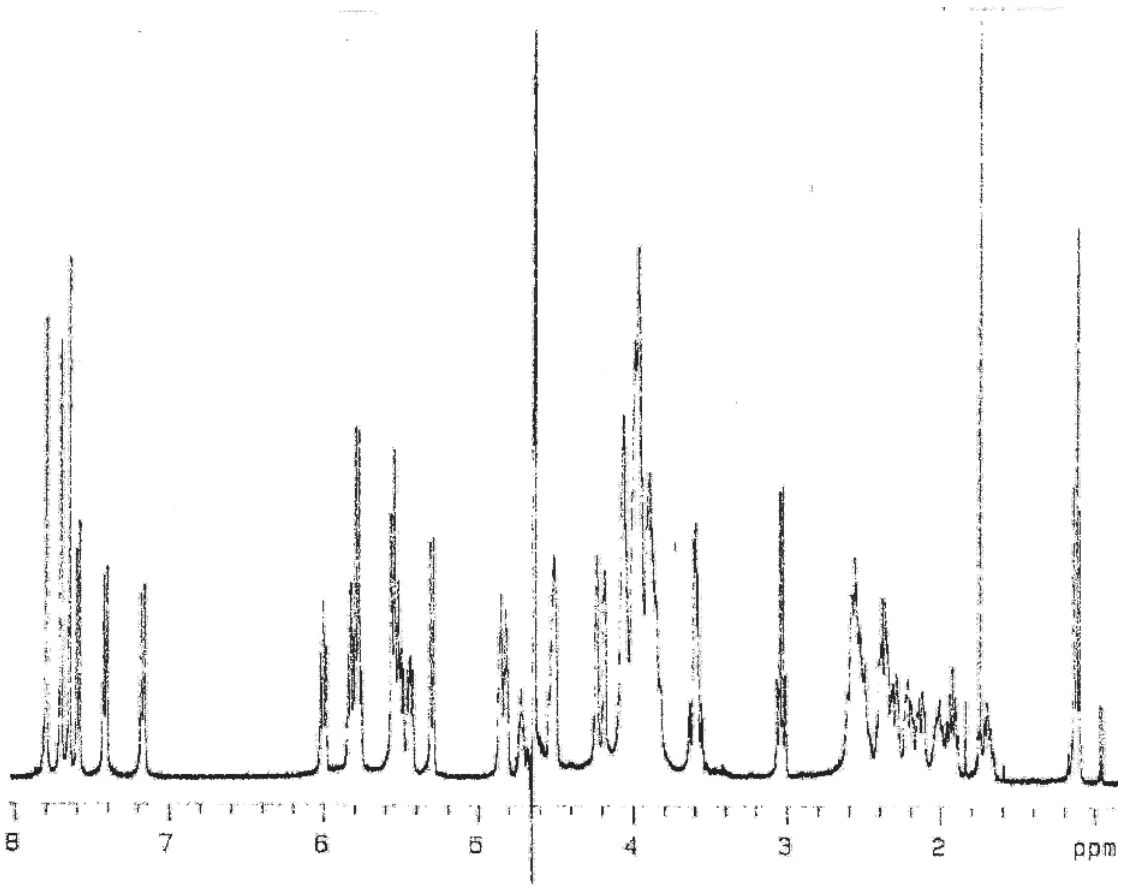


Figura 6.6: Espectro 1D PRESAT del d(CCGCGG)$_2$ en D_2O a 400 MHz, 2 mM en d(CCGCGG)$_2$, 50 mM NaCl, pH 7.0 y 298 K.

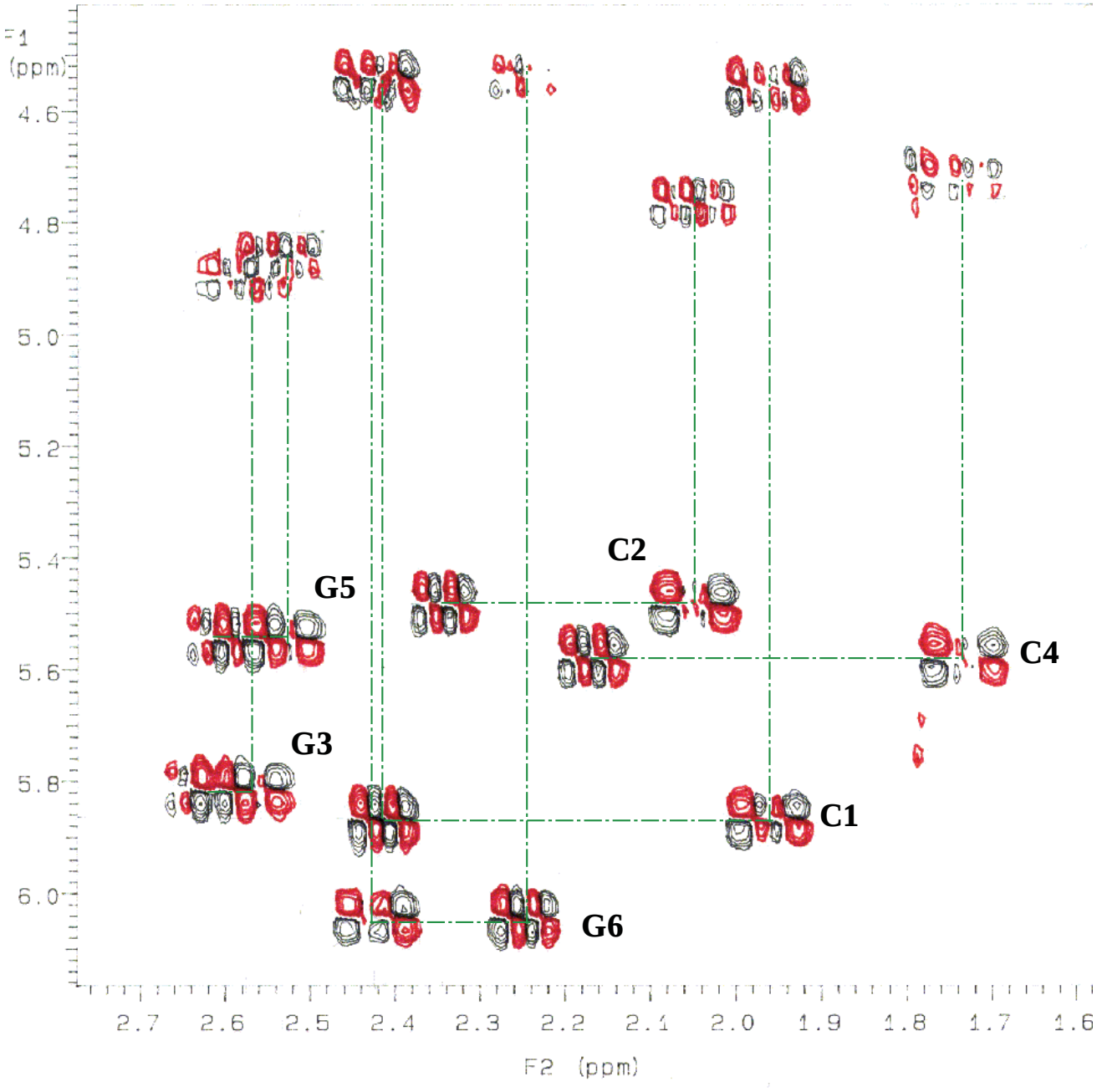


Figura 6.7: Región de protones del azúcar del espectro DQF-COSY del $d(CCGCGG)_2$, condiciones experimentales idénticas a figura 6.6. Se ha señalizado la identificación de los sistemas de espín de los azúcares mediante líneas.

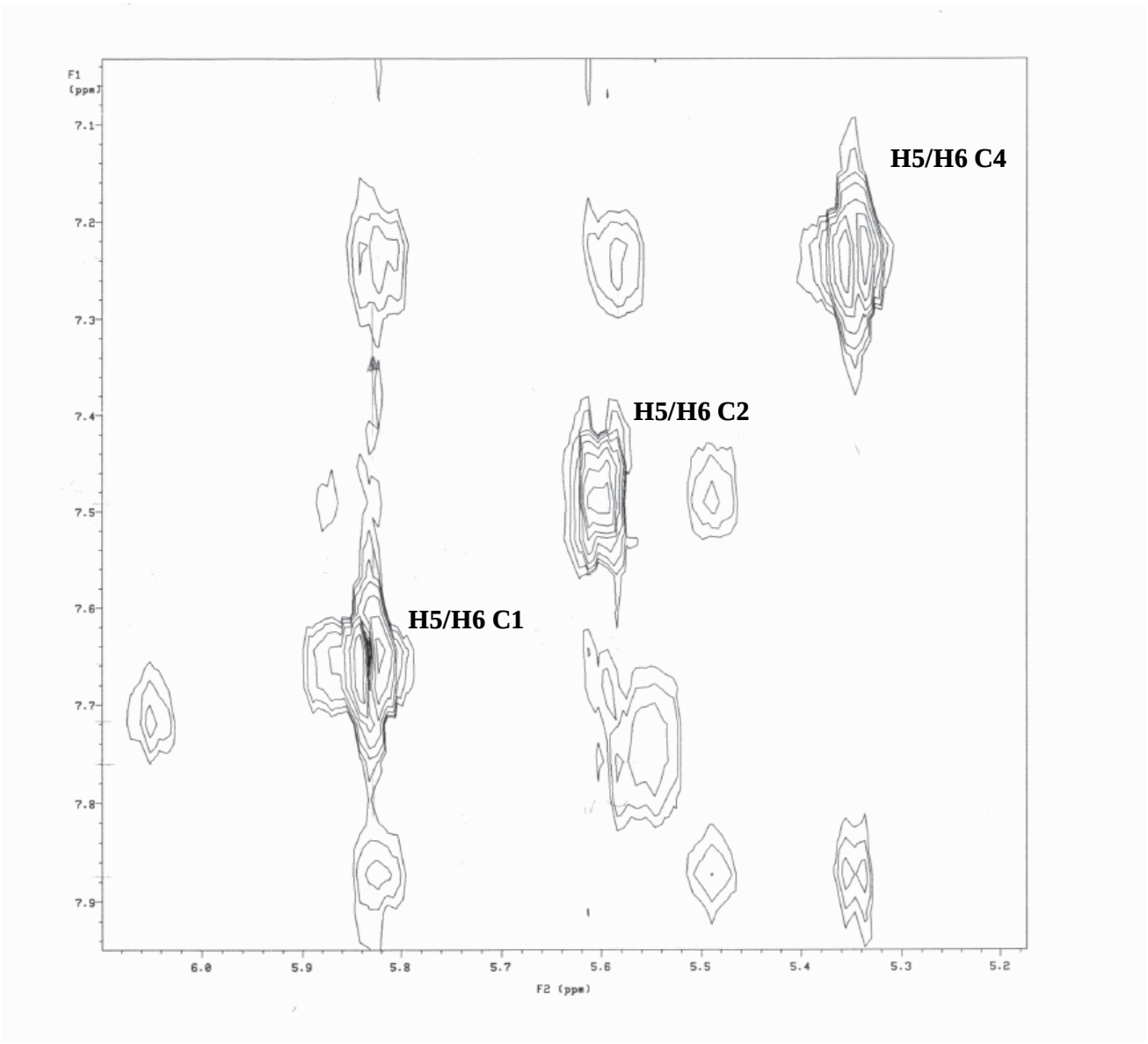


Figura 6.8: Región de protones aromáticos del espectro NOESY del d(CCGCGG)$_2$, condiciones experimentales idénticas a figura 6.6. Se puede observar la simetría del sistema en la aparición de tres únicos picos cruzados para las resonancias $H_5 - H_6$ de las citosinas.

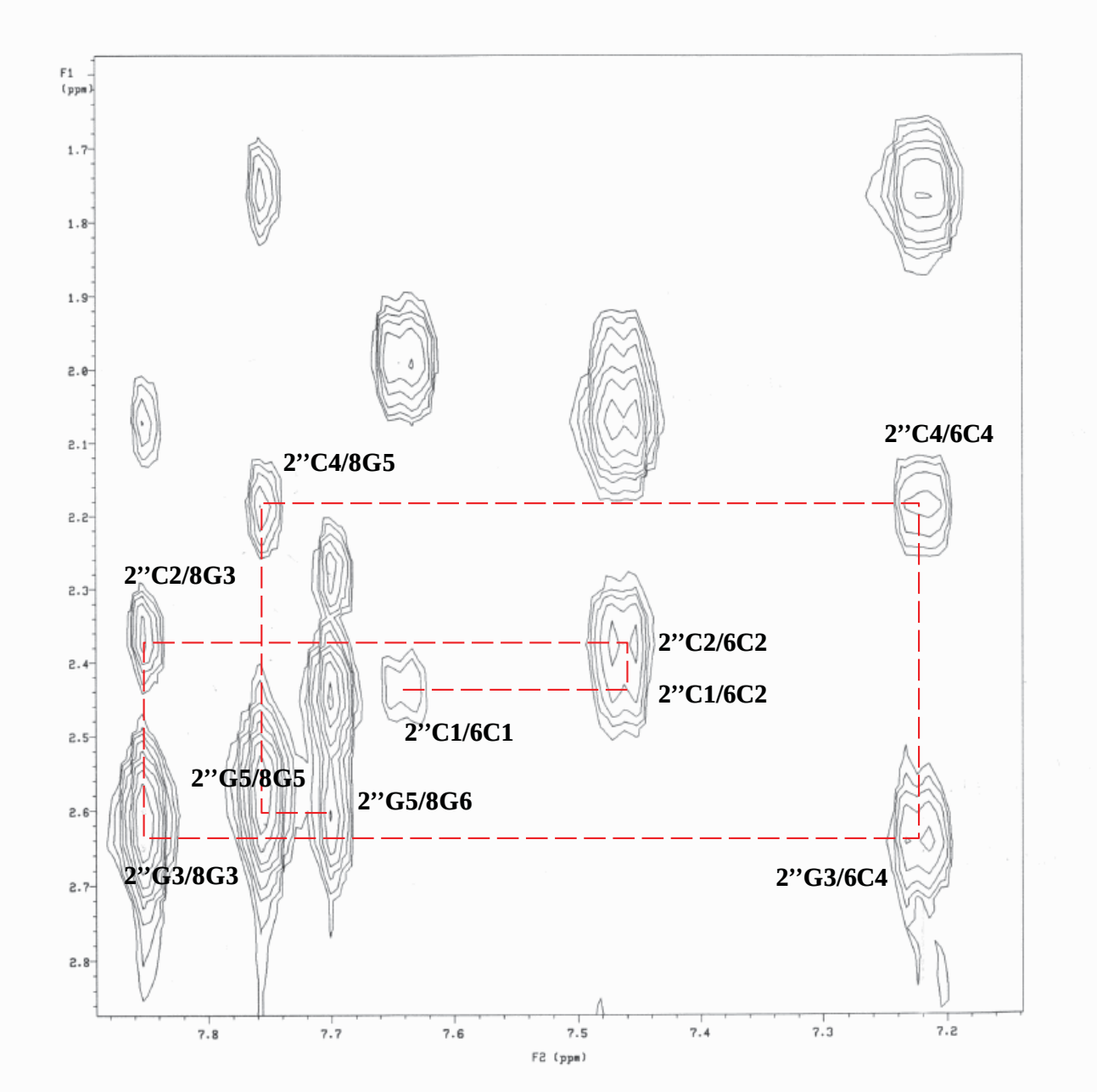


Figura 6.9: Asignación secuencial de los sistemas de espín del d(CCGCGG)$_2$ en el espectro NOESY, condiciones experimentales idénticas figura 6.6.

	$H_{1'}$	$H_{2'}$	$H_{2''}$	$H_{3'}$	$H_{4'}$	$H_{5'}$	$H_{5''}$	H_5	H_6	H_8
C_1	5.86	1.96	2.42	4.55	4.01	3.61	3.69	5.84	3.65	–
C_2	5.48	2.06	2.35	4.77	4.03	3.95	4.05	5.60	7.49	–
G_3	5.82	2.59	2.64	4.88	4.28	4.02	4.05	–	–	7.87
C_4	5.58	1.75	2.17	4.72	4.05	??	??	5.35	7.25	–
G_5	5.54	2.55	2.61	4.84	4.23	3.91	3.99	–	–	7.78
G_6	6.04	2.24	2.42	4.54	4.12	4.03	4.13	–	–	7.73

Tabla 6.6: Tabla de asignación de resonancias de los protones del d(CCGCGG)$_2$ a pH = 7, [NaCl] = 0.05 M y 25^o C [179].

	$H_{1'}$	$H_{2'}$	$H_{2''}$	$H_{3'}$	$H_{4'}$	$H_{5'}$	$H_{5''}$
C_1	5.98	2.07	2.50	4.67	4.15	3.78	4.05
C_2	5.56	2.17	2.42	4.88	4.16	3.79	4.06
G_3	5.90	2.66	2.72	5.03	4.40	4.05	4.12
C_4	5.67	1.86	2.26	4.84	4.18	4.13	??
G_5	5.68	2.71	2.65	4.99	4.35	4.00	4.09
G_6	6.14	2.36	2.54	4.62	4.23	4.15	4.25

Tabla 6.7: Tabla de asignación de resonancias de los protones de los azúcares del d(CCGCGG)$_2$ a pH = 7, [NaCl] = 4M y 30^o C [179] a 600 MHz.

tración salina, los protones imino no fueron asignados en dichos espectros.

Se comprueba una inversión de los desplazamientos químicos de los protones $H_{2'}$ y $H_{2''}$ para el nucleótido G6. Dicha inversión posiblemente se deba a una variación conformacional en dicho nucleótido como resultado de la mayor flexibilidad de los extremos de la molécula. De hecho, en la estructura de rayos X, éste es uno de los nucleótidos que orienta su base aromática hacia el exterior de la molécula. También se ha observado un fuerte solapamiento entre los picos cruzados correspondientes a los nucleótidos G3 y G5.

En contra de lo que cabría esperar, la apariencia de los espectros obtenidos a fuerzas iónicas elevadas no mostraron grandes variaciones respecto a los de fuerza iónica baja como se puede comprobar en la figura 6.10. Incluso la estructura hiperfina de la mayoría de los picos cruzados DQF-COSY se mantiene, observándose únicamente pequeñas diferencias en los nucleótidos terminales. Sin embargo, los desplazamientos químicos sí que varían bastante como se puede comprobar en la tabla 6.7. Así, algunos fuertes solapamientos presentes en los espectros a baja concentración salina se

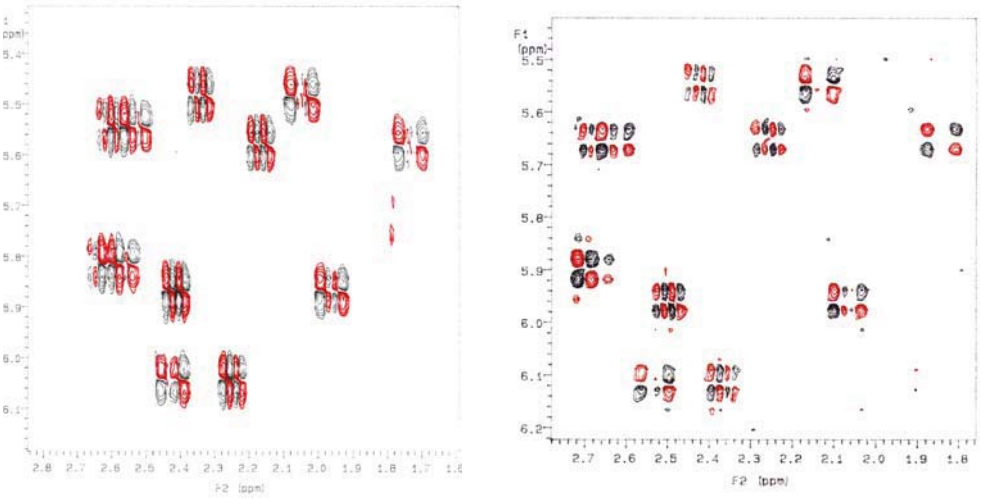


Figura 6.10: Comparación de la región de los azúcares en el espectro DQF-COSY para las muestras a 0.005 M en Na^+ 400 MHz (derecha) y 4 M en Na^+ (izquierda), 600 MHz, 2 mM en d(CCGCGG)$_2$, pH 7.0 y 298 K [179].

ven semiresueltos y las formas de los picos conjuntos varían ligeramente. Lamentablemente, los espectros a concentración de NaCl de 4 molar sólo se pudieron realizar a temperaturas diferentes a los 25^o utilizados en los espectros a baja concentración salina por lo que dichas diferencias en los nucleótidos terminales bien podrían deberse a la diferente contribución térmica.

6.3.2 Conformación de los azúcares

La simulación de picos DQF-COSY permitió la evaluación de los valores de constantes de acoplamiento escalar ^3J que mejor reproducían los picos cruzados experimentales. Sin embargo, el solapamiento presente entre algunos picos como $H_{1'} - H_{2'}$ del nucleótido C1 con $H_{1'} - H_{2''}$ de G6, o los picos $H_{1'} - H_{2'}$ de las guaninas G3 y G5 dificultaron en cierta medida la simulación de dichos picos. No obstante, se pudo conseguir un conjunto aceptable de valores de ^3J para los que la forma de los picos simulados y los picos experimentales es razonablemente similar como se puede comprobar en la figura 6.11. Los valores de ^3J obtenidos se pueden observar en la tabla 6.9

Para obtener los valores de ángulo de pseudorotación más adecuado

	50 ms		200 ms	
	1'H / 4'H	2"H / 4'H	1'H / 4'H	2"H / 4'H
C1	94	71	268	166
C2	155	67	374	134
G3	140	83	254	888 †
C4	186	99	433	148
G5	106	51	266	111
G6	183	59	225	104

Tabla 6.8: Intensidades NOESY de los picos cruzados H1'H4' y H2"H4' para d(CCGCGG)$_2$ a 0.005 M en NaCl, condiciones experimentales idénticas a figura 6.6. † Solapamiento con 2'H / 4'H.

	$J_{1'2'}$	$J_{1'2''}$	$J_{2'2''}$	$J_{2'3'}$	$J_{3'4'}$	$J_{3'P}$
C_1	7.4 a 8.0	5.8 a 6.4	-12.0 a -13.0	5.0 a 6.0	3.5 a 4.5	< 3.5
C_2	9.5 a 10.6	4.7 a 5.7	-12.0 a -13.0	4.5 a 5.5	< 3.0	< 2.0
G_3	10.0 a 11.5	4.7 a 5.7	-13.0 a -14.0	4.5 a 5.5	< 3.0	< 2.0
C_4	10.4 a 11.0	5.4 a 6.0	-13.0 a -14.0	8.0 a 8.7	< 3.0	< 3.0
G_5	9.0 a 10.5	5.2 a 6.4	-13.0 a -14.0	4.0 a 5.0	3.5 a 4.5	< 3.5
G_6	7.2 a 8.2	5.5 a 6.5	-13.0 a -14.0	5.2 a 7.0	< 4.5	< 3.5

Tabla 6.9: Tabla de constantes de acoplamiento que proporcionan resultados óptimos en la comparación de picos simulados y picos experimentales DQF-COSY.

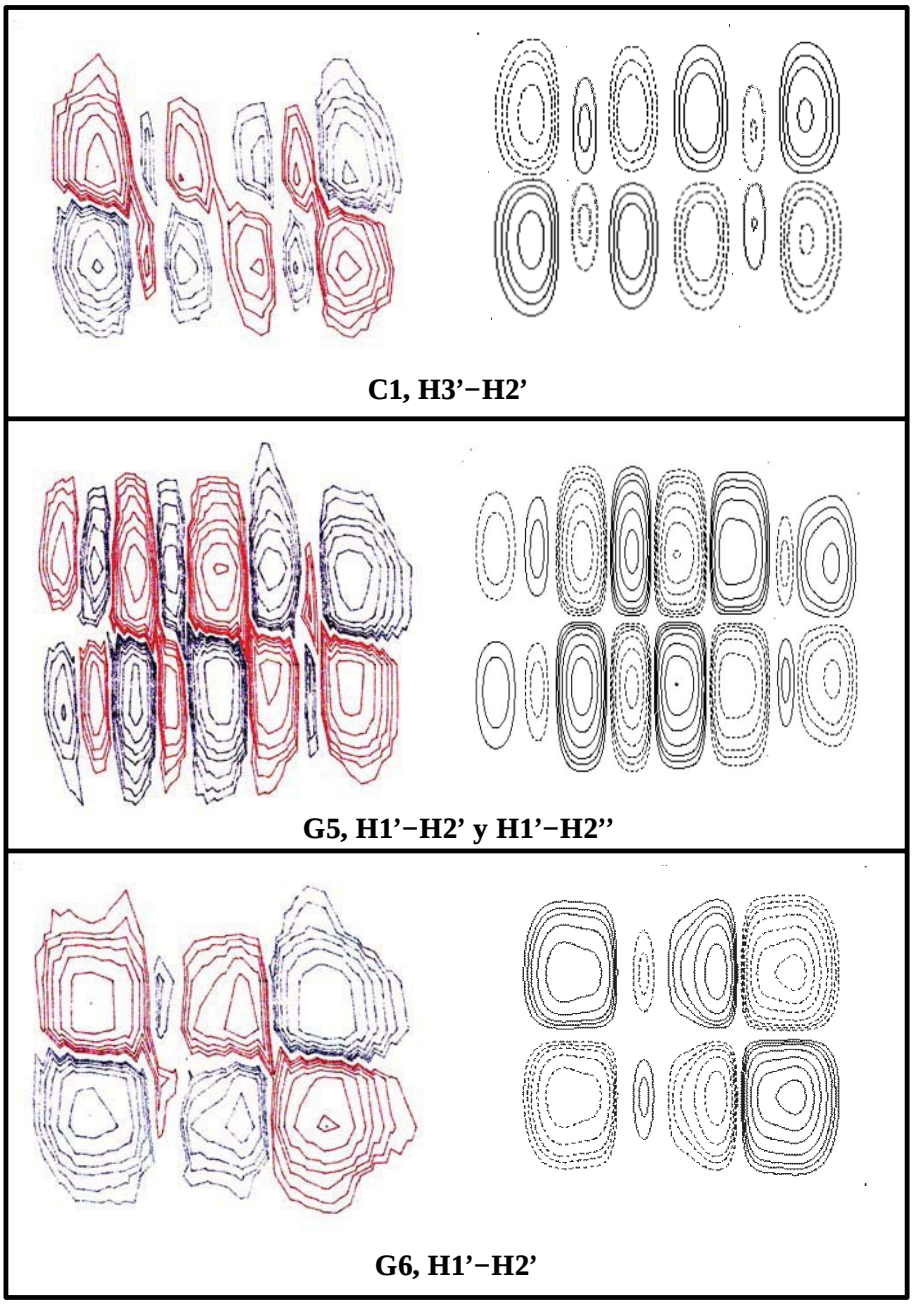


Figura 6.11: Comparación de algunos picos cruzados DQF-COSY entre simulados y experimentales [179].

	Situación	P	ϕ_1	ϕ_2
C_1	X_S = 0.7	110 ± 10	26 a 34	-22 a -6
C_2	X_S = 0.9	150 ± 10	35 a 45	-40 a -30
G_3	X_S = 0.9	150 ± 10	35 a 45	-40 a -30
C_4	Estática	115 ± 15	26 a 50	-26 a -6
G_5	X_S = 0.9	150 ± 10	35 a 45	-40 a -30
G_6	X_S = 0.7	110 ± 10	26 a 34	-22 a -6

Tabla 6.10: Tabla de ángulos de pseudorotación P y ángulos endocíclicos calculados mediante simulación teórica de los picos cruzados DQF-COSY para el d(CCGCGG)$_2$.

a cada conjunto de constantes de acoplamiento, se utilizaron las tablas de constantes de acoplamiento vs ángulo de pseudorotación calculadas teóricamente [1]. Se consideraron las dos posibles situaciones generales. Por una parte, una situación estática con una conformación única y por otra la interconversión entre dos conformaciones extremas S y N con ángulos de pseudorotación 171^o y 9^o respectivamente, descritas en la sección 6.1.4. Mediante esta comparación y la expresión 6.2 se obtuvieron los intervalos de ángulos de pseudorotación y ángulos endocíclicos de la tabla 6.10.

Por otra parte, la pequeña separación entre los desplazamientos químicos de los protones $2'$ y $2''$ en los nucleótidos G3 y G5 hace que las peculiaridades de la estructura hiperfina se vean solapadas entre ellas produciendo picos díficiles de simular. No obstante, se ha podido reproducir la mayoría de estos picos mediante la combinación de diferentes conjuntos de ^3J en los nucleótidos implicados.

En la tabla 6.10 se puede observar que la mayoría de nucleótidos presentan interconversión rápida entre las conformaciones S y N con porcentajes elevados de X_S. Los residuos G5 y C4 parecen ser los que presentan mayores diferencias respecto al resto, siendo el C4 el único que no muestra interconversión pero adoptando una conformación con un valor de P bastante lejano a los habituales para forma estándar B-ADN. Así, la mayoría de residuos adoptan conformaciones con valores de P próximos a la conformación $C_{2'}$-endo característica de la conformación B-ADN típica de oligonucleótidos en disolución. Sin embargo, los valores de P de los residuos extremos y, sobre todo, de C4 unido a la inversión de los protones $H_{2'}$ y $H_{2''}$ de G6 indican fuertes distorsiones de la estructura regular B-ADN en dichos nucleótidos.

Los 3 nucleótidos que presentan interconversión entre las conformacio-

nes S y N con valores de X_S próximos a 0.9 podrían simularse mediante una situación estática próxima a la conformación pura S (P = 170^o) con pequeñas desviaciones. Sin embargo, en los nucleótidos G3 y G5 aunque dicha opción da resultados similares en los picos cruzados $H_{1'} - H_{2'}$ y $H_{1'} - H_{2''}$, la simulación del pico $H_{3'} - H_{2'}$ se ve ligeramente mejorada utilizando los valores de J de interconversión con X_S 0.9. En el nucleótido C2, por otra parte, la simulación con P = 171^o no reproduce tan bien como X_S = 0.9 los picos experimentales por lo que las situación dinámica ha sido preferida sobre la estática.

6.3.3 Estructuras obtenidas

Se han obtenido 30 estructuras en total, clasificadas según su estructura de partida en A, B y Z, y según el nivel de restricciones utilizado en $dihe$ y $dihe + dist$. Cada conjunto de estructuras consta de 10 conformaciones obtenidas mediante extracción y minimización del proceso de dinámica molecular a 300 K cada 10 ps tal y como se ha explicado en la sección de métodos. En la tabla 6.11 se pueden observar los valores de energías totales y de restricciones promedio de cada conjunto de estructuras. Aunque las violaciones son bastante pequeñas, lo cual indica buen acuerdo entre la información experimental y las estructuras obtenidas, las energías obtenidas no son muy bajas comparadas con otras estructuras de ADN. Esto es debido a las fuertes distorsiones que presentan las estructuras. Estas distorsiones podrían verse estabilizadas en cierta medida mediante factores entrópicos como la solvatación de difícil evaluación en las actuales técnicas de modelización molecular. Sin embargo, teniendo en cuenta las altísimas energías obtenidas para la estructura de rayos X, es difícil evaluar todos los efectos que estabilizan este tipo de estructuras tanto en disolución como en estado cristalino.

En general, se observa que las estructuras calculadas con información estructural de diedros y distancias presentan grados de convergencia mayores como corresponde a un mayor número de restricciones. En concreto, la conformación de los nucleótidos extremos es muy variable a lo largo de las estructuras determinadas con información sólo de diedros. De todos los conjuntos de estructuras $B_{dihe+dist}$, parece ser el que presenta menos distorsión respecto a los parámetros ideales de alguna de las formas de ADN. El nivel de violaciones y de energía no es muy diferente entre todos los conjuntos, aunque $B_{dihe+dist}$ muestra los mejores resultados con pequeñas diferencias respecto al resto. Por otra parte, la estructura $B_{dihe+dist}$ es la

	Energía Total	Energía Rest. Diedros	Energía Rest. Distancias	Energía VdW
$A_{dihe+dist}$	-105	1.80	1.20	-82
$B_{dihe+dist}$	-118	1.82	1.10	-95
$Z_{dihe+dist}$	255	14.30	4.20	211
A_{dihe}	-62	1.1	–	-50
B_{dihe}	-123	1.3	–	-94
Z_{dihe}	-80	1.7	–	-61

Tabla 6.11: Tabla de energías promedio de cada conjunto de estructuras en kcal / mol.

	$B_{dihe+dist}$	Z_{dihe}	$RayosX$
Diámetro (Å)	10.63	11.59	13.51
Giro de la hélice (^{o})	22.49	28.07	46.25
Par de bases por vuelta	10.94	9.64	14.12
Elevación por par de bases (Å)	2.06	2.91	3.27
Curvatura de la hélice (^{o})	86.5	134.1	0.6
Curvatura de la hélice (Å)	11.3	7.3	1700.0

Se han tomado los C_1' para los cálculos

Tabla 6.12: Tabla comparativa de parámetros generales de las estructuras de menor energía $B_{dihe+dist}$, A_{dihe} y la estructura de rayos X calculados con NDBSTAT.

	$B_{dihe+dist}$	Z_{dihe}	$RayosX$
$C_1 - G12$	0.470	0.083	1.317
$C_2 - G11$	0.449	0.185	0.109
$G_3 - C10$	0.338	0.156	0.207
$C_4 - G_9$	0.345	0.201	0.123
$G_5 - C_8$	0.463	0.391	0.050
$G_6 - C_7$	0.644	0.948	1.404

Tabla 6.13: Desviación de la planaridad en Å de las estructuras $B_{dihe+dist}$, Z_{dihe} y Rayos X calculada con NDBSTAT.

que conserva mejor el centro de simetría observado experimentalmente en los espectros de RMN. Se puede observar que todas las estructuras presentan una curvatura de hélice muy elevada respecto a las conformaciones estándar de A-ADN, B-ADN y Z-ADN. En particular, esta gran curvatura choca con la nula curvatura obtenida en la estructura de rayos X de este mismo nucleótido (ver tabla 6.12). Sin embargo, esta nula curvatura en la estructura de rayos X puede ser relacionada con los efectos de apilamiento propios de la formación de los tetrámeros. Por otra parte, la presencia del ión Na^+ entre los dos duplex que forman el tetrámero, produce un acercamiento de las bases centrales de cadenas diferentes que endereza en cierta manera las hélices de ADN individuales para favorecer la interacción entre ambas. El diámetro de la hélice es aproximadamente del mismo orden de otros calculados tomando los mismos átomos de referencia para estructuras estándar [169].

Es posible comprobar también tanto en la tabla 6.13 como en la figura 6.12 que los pares de bases se encuentran en posiciones ligeramente distorsionadas respecto a la posición ideal para la formación de puentes de hidrógeno de tipo Watson-Crick. En concreto, la desviación de la planaridad de las bases y el valor del ángulo ω son algo superiores a las observadas en las estructuras ideales de rayos X en la estructura en tetrámero. En las estructuras Z_{dihe}, sin embargo, esta desviación es menor en el segmento central CGCG mientras que en los residuos de los extremos la distorsión es muy elevada. Como se puede observar en las figuras 6.12 y 6.13, el nucleótido C1 de estas estructuras muestra tal nivel de movilidad a lo largo de la dinámica que se reorienta hacia el exterior de la hélice, en conforma-

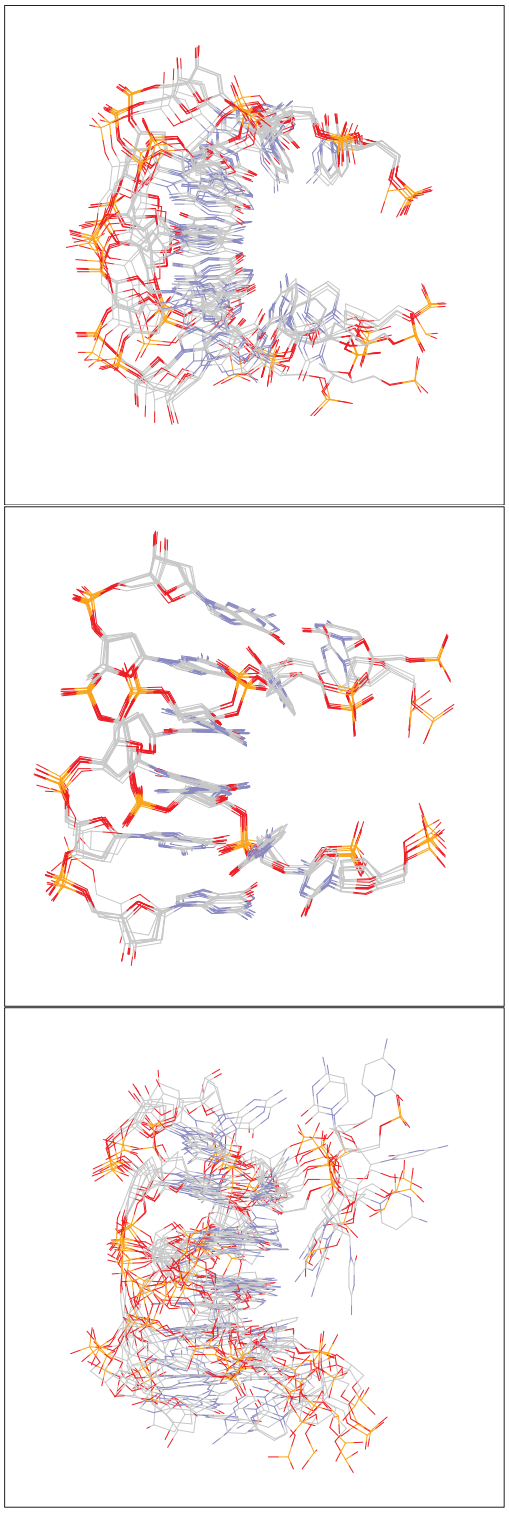


Figura 6.12: Representación tridimensional de las estructuras calculadas $A_{dihe+dist}$ (arriba), $B_{dihe+dist}$ (centro) y Z_{dihe} (abajo).

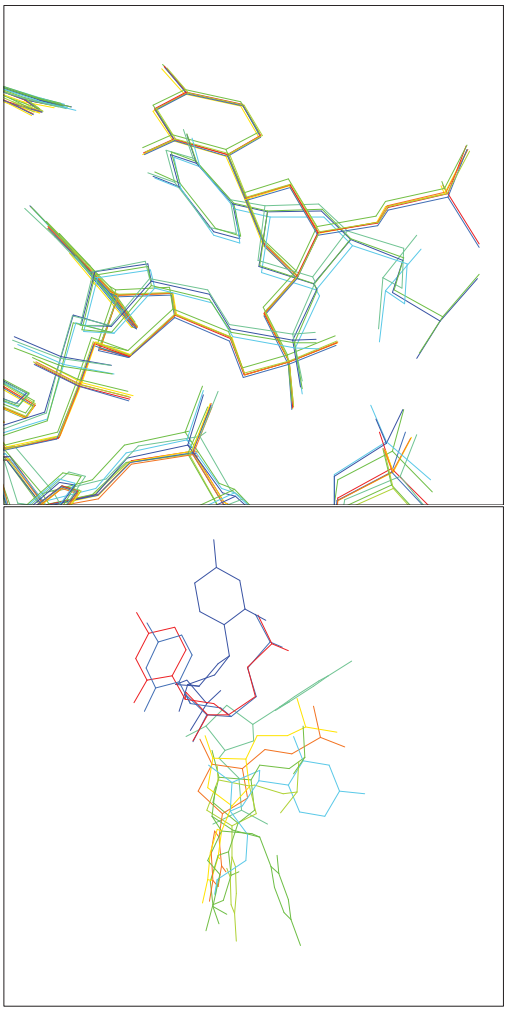


Figura 6.13: Ampliación del par CG terminal de las estructuras calculadas $B_{dihe+dist}$ (arriba) y Z_{dihe} (abajo).

ciones que recuerdan a la estructura obtenida por rayos X.

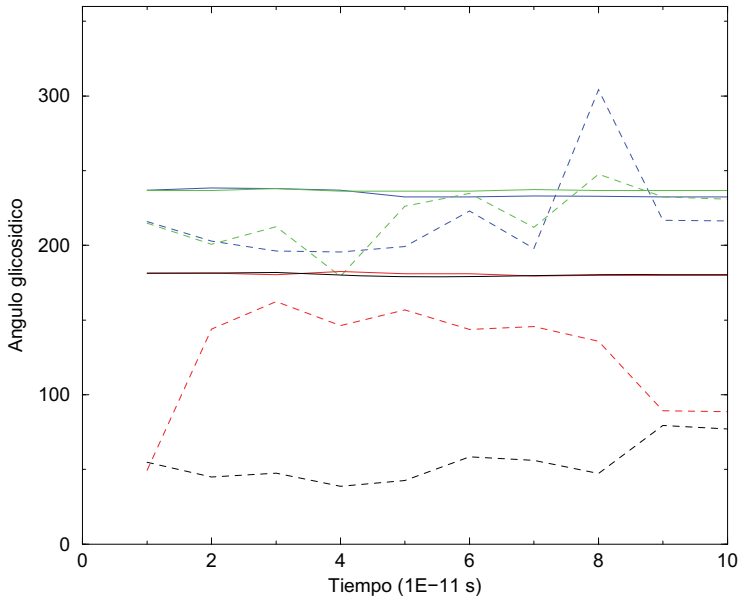


Figura 6.14: Variación del ángulo de glicosídico χ con el tiempo de dinámica para los nucleótidos C1 (azul),G3 (verde),C7 (rojo) y G9(negro) para las estructuras $B_{dihe+dist}$ (línea contínua), $A_{dihe+dist}$ (línea discontínua) y Z_{dihe} (línea de puntos).

En la figura 6.14 se puede observar la variación del ángulo glicosídico χ [177] para diversos nucleótidos de los extremos y centrales con el tiempo de dinámica en las estructuras $B_{dihe+dist}$ y Z_{dihe}. Se puede comprobar que en la estructura en disolución $B_{dihe+dist}$ los valores de χ tanto en los extremos como en los nucleótidos centrales varían muy poco con el tiempo mientras que en la estructura distorsionada Z_{dihe} la variación en general es mayor siendo en los extremos muy superior indicando una gran flexibilidad en la posición de las bases nitrogenadas correspondientes[178]. Un efecto similar se puede observar en la figura 6.15 en la que el ángulo de pseudorotaión P ha sido representado en las mismas condiciones. En este caso, se puede observar que la conformación del azúcar varía menos que el ángulo glicosídico debido a las restricciones de diedros impuestas. Sin embargo, también se puede apreciar un aumento de la flexibilidad en los extremos incluso en el residuo G12 de la estructura $B_{dihe+dist}$ que sufre un cambio de conformación en los 80 ps de dinámica.

La posible reorientación de las bases nitrogenadas de los nucleótidos terminales así como su mayor flexibilidad se pueden apreciar también en la figura 6.16 que representa la variación de una de las distancias más re-

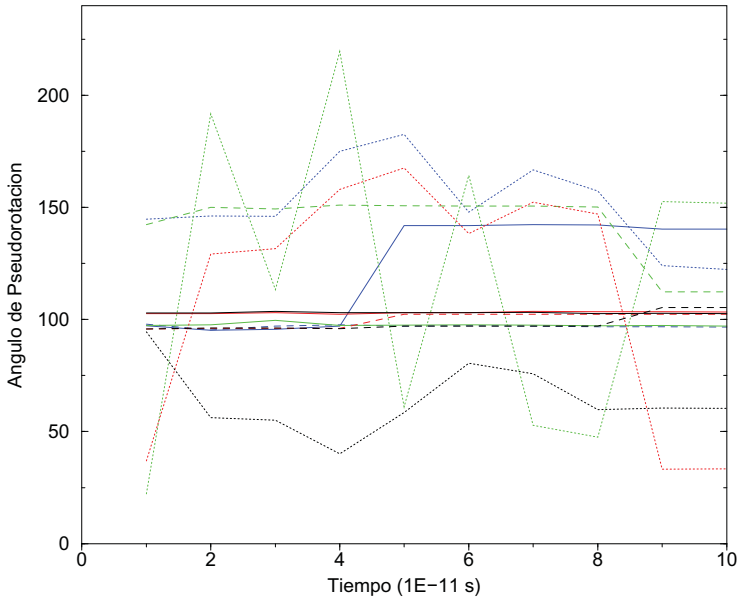


Figura 6.15: Variación del ángulo de pseudorotación P con el tiempo de dinámica para los nucleótidos C1 (azul),G3 (verde),C7 (rojo) y G9(negro) para las estructuras $B_{dihe+dist}$ (línea contínua), $A_{dihe+dist}$ (línea discontínua) y Z_{dihe} (línea de puntos).

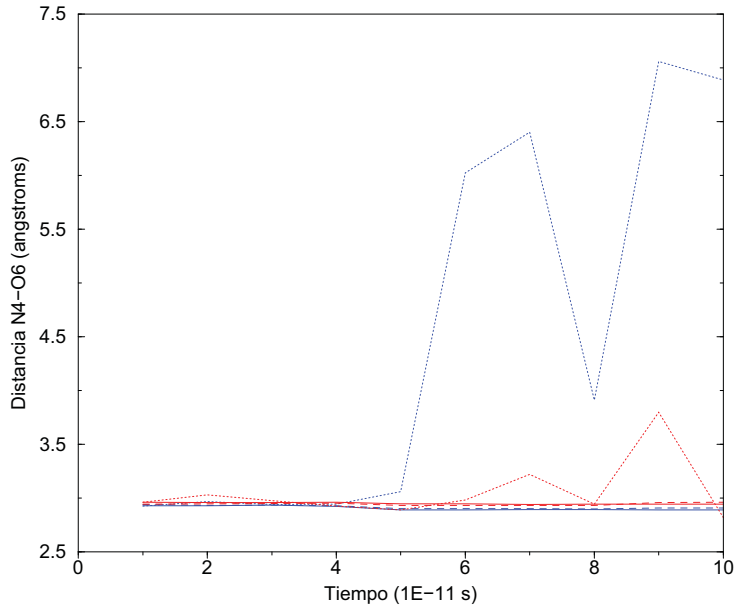


Figura 6.16: Variación de la distancia $N_4 - O_6$ de las bases nitrogenadas con el tiempo de dinámica para los nucleótidos C1 (azul) y G3 (rojo) para las estructuras $B_{dihe+dist}$ (línea contínua), $A_{dihe+dist}$ (línea discontínua) y Z_{dihe} (línea de puntos).

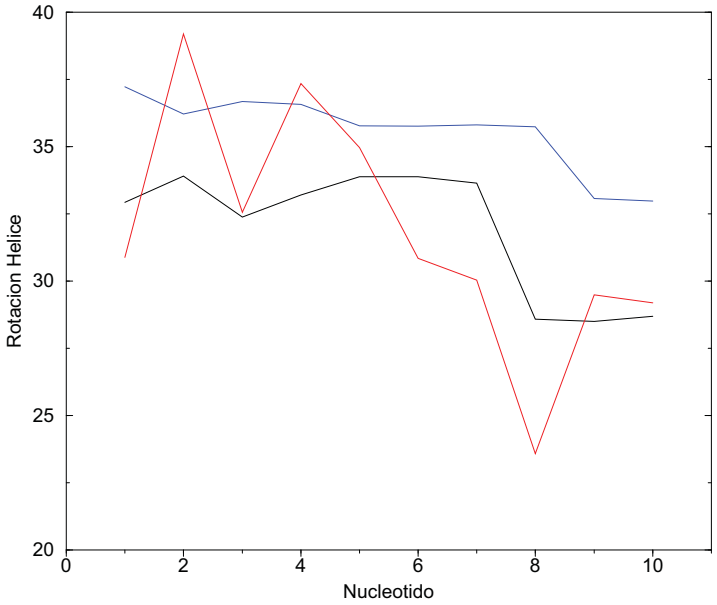


Figura 6.17: Variación del ángulo de rotación de la hélice ($twist$) con el tiempo de dinámica para las estructuras $B_{dihe+dist}$ (negro), $A_{dihe+dist}$ (azul) y Z_{dihe} (rojo).

presentativas entre las bases nitrogenadas con el tiempo de dinámica. Se puede comprobar que, si bien en las estructuras $A_{dihe+dist}$ y $B_{dihe+dist}$ la variación es apenas imperceptible en ninguno de los residuos representados, en el caso de la estructura Z_{dihe} se verifica un aumento bastante elevado en el par de bases G6-C7 apoyando las observaciones realizadas a nivel de ángulo glicosídico y ángulo de pseudorotación sobre la posible reorientación del anillo aromático en esta estructura. Por la evolución de la energía total en las dinámicas realizadas con el tiempo, no parece que exista una barrera de energía potencial muy elevada para dicha reorientación ya que no se observan grandes variaciones energéticas con el tiempo de dinámica [177]. La poca variación con el tiempo del giro de la hélice (ver figura 6.17) también nos indica que la conformación global de la cadena no experimenta grandes cambios. Por otra parte, el hecho de que las estructuras obtenidas no presenten un nivel de violaciones muy elevado sobre las restricciones de ángulos diedros impuestas apoya la suposición de que la reorientación de los anillos aromáticos de los nucleótidos extremos se puede alcanzar mediante pequeñas modificaciones conformacionales a nivel local no muy energéticas partiendo de la estructura en disolución.

Las estructuras obtenidas en disolución muestran una conformación cercana a la forma B-ADN. La conformación de los azúcares es próxima a a

Par de bases	θ (Tip)	η (*Inclinación*)	ω ($Propeller\ Twist$)	κ ($Buckle$)	δx (Å)	δy (Å)
$C_1 - G_{12}$	1.675	30.454	-24.133	145.394	-3.147	0.900
$C_2 - G_{11}$	3.212	30.190	-32.567	147.012	-3.570	0.098
$G_3 - C_{10}$	5.867	34.317	-40.684	-179.345	-2.980	-0.120
$C_4 - G_9$	-6.916	33.998	-40.687	-180.000	-2.993	0.153
$G_5 - C_8$	-2.331	29.872	-30.153	-145.532	-3.565	-0.066
$G_6 - C_7$	-2.845	31.893	-28.086	-134.144	-3.329	-1.162
Promedio	-0.223	31.787	-32.718	-177.770	-3.264	-0.033
Desviación Estándar	4.661	1.964	6.763	33.309	0.267	0.664

Tabla 6.14: Parámetros de los pares de base del d(CCGCGG)$_2$ respecto al eje de la hélice para la estructura $B_{dihe+dist}$ calculados con NDBSTAT.

Pares de bases	Dz ($Rise$)	Dy ($Slide$)	Dx ($Shift$)	Ω ($Twist$)	ρ ($Roll$)	τ ($Tilt$)
$1-2$	2.389	-2.471	-0.061	22.997	15.128	1.669
$2-3$	2.053	-1.827	1.141	25.794	18.767	5.935
$3-4$	2.279	-1.500	0.460	36.304	9.288	3.153
$4-5$	1.995	-2.203	0.075	26.050	21.347	3.453
$5-6$	1.956	-2.587	1.242	24.967	13.482	6.410
Promedio	2.135	-2.118	0.571	27.222	15.602	4.124
Desviación Estándar	0.190	0.452	0.598	5.216	4.680	1.995

Tabla 6.15: Parámetros de posición relativa de pares de base consecutivos del d(CCGCGG)$_2$ para la estructura $B_{dihe+dist}$ calculados con NDBSTAT.

Par de bases	θ (Tip)	η (*Inclinación*)	ω ($Propeller\ Twist$)	κ ($Buckle$)	δx (Å)	δy (Å)
$C_1 - G_{12}$	24.004	5.387	-2.463	-176.065	-1.357	0.422
$C_2 - G_{11}$	6.008	19.754	11.947	-170.591	-0.257	0.233
$G_3 - C_{10}$	7.515	21.230	8.632	178.699	-0.742	-0.242
$C_4 - G_9$	2.498	23.038	25.401	179.463	-0.371	0.654
$G_5 - C_8$	-18.895	17.327	-11.372	-153.866	-0.098	-1.958
$G_6 - C_7$	-11.083	40.869	89.588	29.697	0.230	1.494
Promedio	1.675	21.267	20.289	161.223	-0.433	0.101
Desviación Estándar	15.087	11.466	36.205	65.214	0.554	1.160

Tabla 6.16: Parámetros de los pares de base del d(CCGCGG)$_2$ respecto al eje de la hélice para la estructura Z_{dihe} calculados con NDBSTAT.

Pares de bases	Dz ($Rise$)	Dy ($Slide$)	Dx ($Shift$)	Ω ($Twist$)	ρ ($Roll$)	τ ($Tilt$)
$1-2$	2.963	-0.076	1.321	43.401	7.826	0.613
$2-3$	3.284	0.010	-0.484	33.194	-14.062	-1.353
$3-4$	2.613	0.969	0.934	42.629	-12.025	-2.274
$4-5$	5.297	-2.525	-0.211	20.414	14.022	5.156
$5-6$	-1.631	2.920	1.264	26.351	-14.958	-38.403
Promedio	2.505	0.260	0.565	33.198	-3.839	-7.252
Desviación Estándar	2.536	1.969	0.851	10.042	13.695	17.648

Tabla 6.17: Parámetros de posición relativa de pares de base consecutivos del d(CCGCGG)$_2$ para la estructura Z_{dihe} calculados con NDBSTAT.

conformación $C_2' - endo$ característica de las estructuras en B-ADN aunque presenta distorsiones importantes sobre todo en los extremos. Los puentes de hidrógeno entre bases parecen bastante tensionados debido a la orientación relativa entre los anillos aromáticos (ver figura 6.18). No obstante, cuando sólo se han utilizado las restricciones de diedros, los puentes de hidrógeno de los pares de bases extremos no han sido suficiente para retener el par de bases. De este modo, una distorsión adicional se produce como efecto de la búsqueda de dadores y aceptores de puente de hidrógeno por parte de la base desemparejada (ver figura 6.19). De hecho, en algunos de las estructuras obtenidas sólo con restricciones de diedros se han medido puentes de hidrógeno entre las bases C8 y G6 que no están enfrentadas en la estructura regular de ADN.

Si nos fijamos en la conformación global de la estructura propuesta como estructura del d(CCGCGG)$_2$ en disolución ($B_{dihe+dist}$) podemos observar que, a pesar de la gran curvatura que muestra la estructura en doble hebra calculada, los ejes locales de las bases se alínean correctamente indicando que no se produce una transición fuerte a lo largo de la secuencia (ver figura 6.22). La estructura es altamente simétrica y el eje de la hélice aparece situado en el centro de la circunferencia que forma la proyección de los átomos de fósforo [174]. Debido a la gran curvatura mostrada en un número tan pequeño de nucleótidos, las diferencias en profundidad del surco mayor y menor no son muy elevadas (6.2 Å y 5.3 Å, respectivamente). Esto hace posible que la complejación con compuestos intercalantes se produzca por el surco menor en lugar del mayor [146] tal y como se ha observado en estudios anteriores.

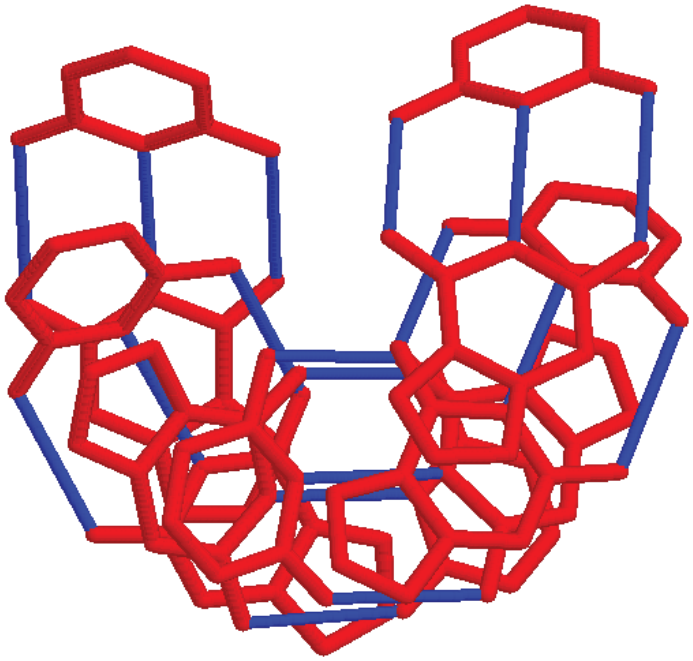


Figura 6.18: Representación tridimensional de la estructura $B_{dihe+dist}$ mostrando sólo las bases nitrogenadas y los puentes de hidrógeno carácteristicos desde una vista superior. Se puede apreciar la simetría del sistema.

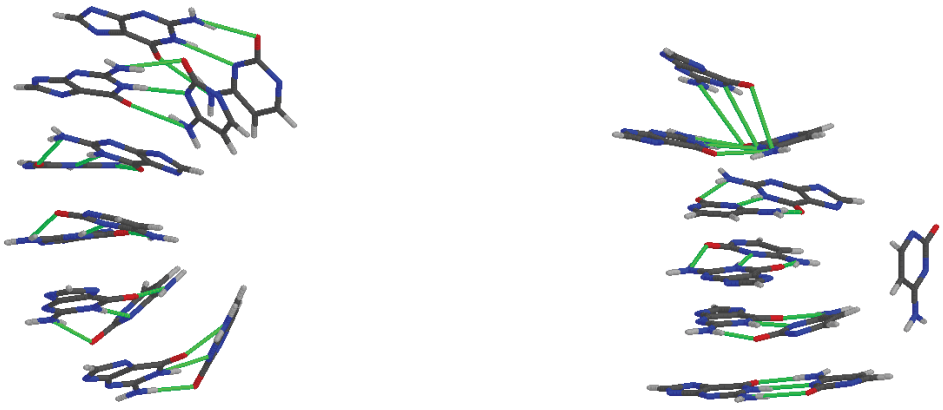


Figura 6.19: Representación tridimensional de la estructura $B_{dihe+dist}$ (derecha) y Z_{dihe} (izquierda) mostrando sólo las bases nitrogenadas y los puentes de hidrógeno carácteristicos desde una vista lateral.

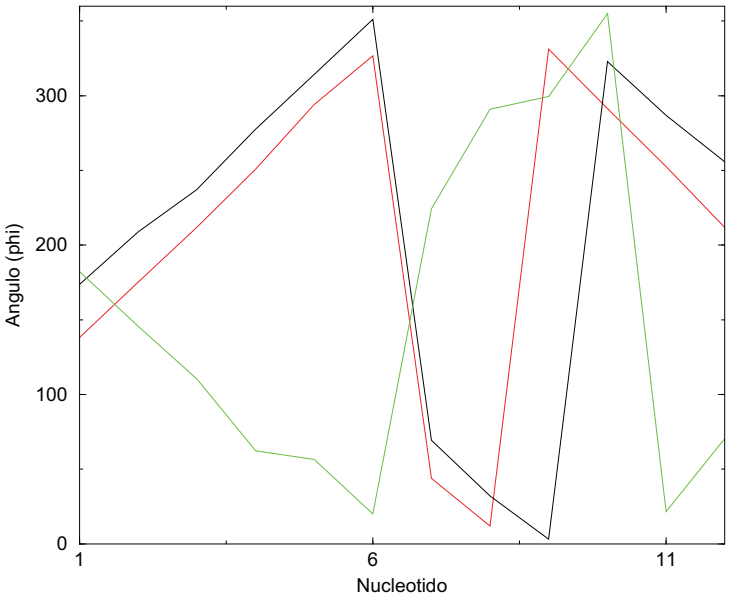


Figura 6.20: Variación de las coordenadas cilíndricas (ángulo ϕ) de los átomos de C_1' con respecto al sistema de coordenadas de la hélice global en la secuencia de nucleótidos para las estructuras de mínima energía $B_{dihe+dist}$ (negro), $A_{dihe+dist}$ (rojo) y Z_{dihe} (verde).

6.3.4 Comparación con estructura de Rayos X

Aunque a la vista de los parámetros estructurales (ver tablas 6.18 y 6.19), es obvio que la estructura en disolución del d(CCGCGG)$_2$ ($B_{dihe+dist}$) guarda pocas similitudes con la estructura cristalina determinada mediante rayos X del mismo [145], tomando como punto de partiza la forma estándar de Z-ADN y utilizando las restricciones experimentales sobre la conformación del azúcar se han encontrado estructuras con un bajo nivel de violaciones que muestran ciertas peculiaridades similares a las observadas en la estructura cristalina. Las violaciones sin embargo, se sitúan en cada caso en diedros diferentes. Esto puede ser debido a las interconversiones entre conformaciones extremas del anillo de desoxiribosa. Al observar una situación promedio e intentar que el sistema la cumpla, se producen violaciones en uno u otro sentido según la conformación calculada sea próxima a uno u otro extremo.

La estructura recogida en el NDB (*Nucleic acid Data Bank*) pertenece a una unidad y no al tetrámero completo. Para comparar los elevados valores de energía obtenidos en los cálculos, se calculó la energía de la estructura de rayos X. Para ello, se le adicionaron los átomos de hidrógeno

	$B_{dihe+dist}$		Z_{dihe}		$Rayos X$	
Nucleótido	Conf.	χ	Conf.	χ	Conf.	χ
C_1	O4*-endo	$anti$	C2*-endo	$anti$	C4*-exo	$anti$
C_7	O4*-endo	$anti$	C4*-endo	$anti$	C4*-exo	$anti$
C_2	C1*-exo	$anti$	C2*-endo	$anti$	C2*-endo	$anti$
C_8	C1*-exo	$anti$	C2*-endo	$anti$	C3*-exo	$anti$
G_3	O4*-endo	$anti$	C2*-endo	$anti$	C3*-endo	syn
G_9	O4*-endo	$anti$	C4*-exo	syn	C3*-endo	syn
C_4	O4*-endo	$anti$	C1*-exo	$anti$	C2*-endo	$anti$
C_{10}	O4*-endo	$anti$	C2*-endo	$anti$	C2*-endo	$anti$
G_5	C1*-exo	$anti$	C4*-exo	syn	C2*-endo	syn
G_{11}	C1*-exo	$anti$	C2*-endo	$anti$	C2*-endo	syn
G_6	O4*-endo	$anti$	O4*-endo	syn	C1*-exo	$anti$
G_{12}	O4*-endo	$anti$	C2*-endo	$anti$	C2*-exo	$anti$

Tabla 6.18: Conformaciones del azúcar y ángulo glicosídico para las estructuras $B_{dihe+dist}$, Z_{dihe} y rayos X del d(CCGCGG)$_2$.

y se procedió a relajar sus posiciones. La energía final obtenida fue de 320 kcal / mol, muy superior a la obtenida en los cálculos realizados. Sin embargo, hay que recordar la ausencia de tetrámero de modo que la inestabilización de las bases extremas es muy elevada. Intentos de disminuir esta energía mediante dinámica o minimizaciones sobre la estructura de rayos X del dúplex llevaron a distorsiones muy importantes en la planaridad de las bases y en la estructura en general, sin producirse una disminución de energía considerable.

Las distorsiones observadas en estas estructuras se deben, además de a la aplicación de restricciones promedio sobre las conformaciones de los azúcares, a que en disolución no hay factores estabilizadores para la formación de un posible tetrámero que sí se encuentran presentes en la estructura cristalina. La estructura obtenida es un promedio de un elevado número de posibilidades conformacionales y como tal, las conclusiones estructura-

	$B_{dihe+dist}$		Z_{dihe}		$Rayos X$	
Par de bases	d($O_2 - N_2$)	d($C_1' - C_1'$)	d($O_2 - N_2$)	d($C_1' - C_1'$)	d($O_2 - N_2$)	d($C_1' - C_1'$)
$C_1 - G_{12}$	3.002	10.830	2.934	10.870	17.049	19.140
$C_2 - G_{11}$	2.943	10.460	2.929	10.760	2.735	10.780
$G_3 - C_{10}$	2.917	10.700	2.908	10.750	2.823	10.640
$C_4 - G_9$	2.917	10.700	2.999	10.800	2.653	10.750
$G_5 - C_8$	2.946	10.480	3.018	10.900	2.846	10.850
$G_6 - C_7$	3.012	10.590	13.567	15.480	16.857	18.880

Tabla 6.19: Distancias interhebra más representativas en las estructuras $B_{dihe+dist}$, Z_{dihe} y rayos X para el d(CCGCGG)$_2$ calculadas con NDBSTAT.

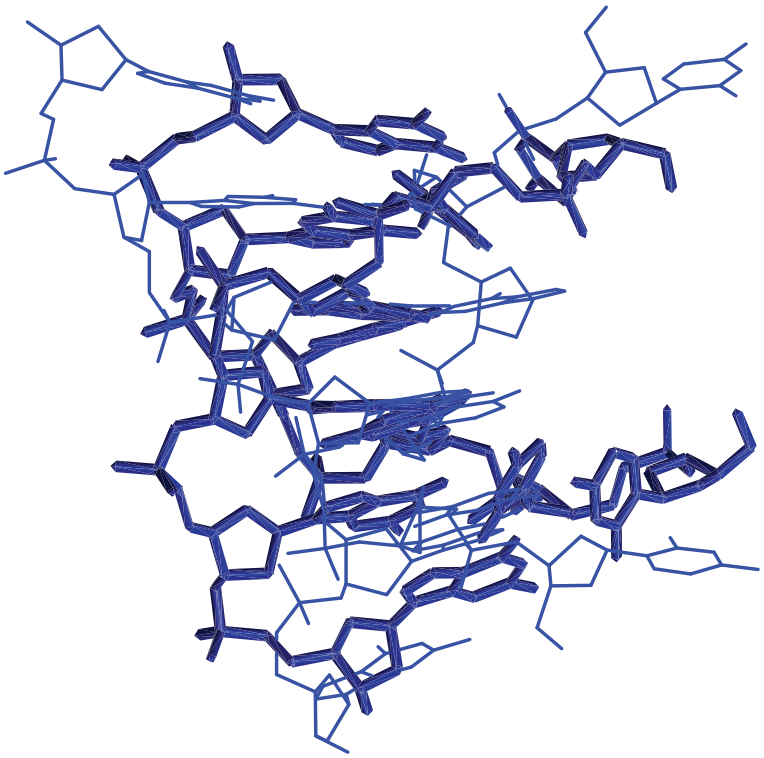


Figura 6.21: Superposición de la estructura promedio de RMN (trazo grueso) y la estructura del dúplex por rayos X (trazo fino) mostrando sólo los átomos pesados. Se ha minimizado la desviación entre C_1'.

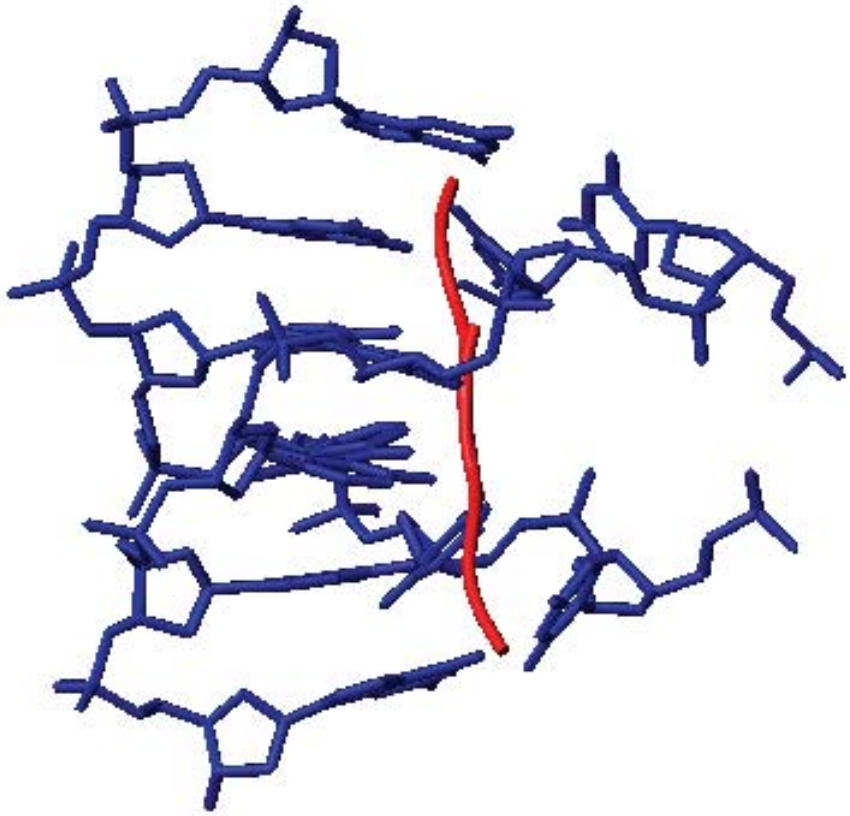


Figura 6.22: Representación de la estructura del d(CCGCGG)$_2$ con los ejes locales (en rojo) calculados por NDBSTAT.

les sobre ella han de tener en cuenta dicha consideración. Por otra parte, en el cristal, tanto las fuerzas de empaquetamiento como la coordinación a iones Na^+ estabilizan la formación de tetrámeros y por lo tanto, la reorientación compensada de los anillos aromáticos de los nucleótidos de los extremos. En disolución dicha estabilización no se produce ni siquiera a fuerzas iónicas elevadas, al menos tal y como se deduce de la información experimental obtenida.

6.4 Conclusiones

Se ha obtenido la estructura promedio en disolución del hexanucleótido d(CCGCGG)$_2$ mediante espectroscopía de RMN. Dicha estructura es notablemente diferente a la estructura de la misma molécula en monocristal afuerza iónica elevada. Mientras la estructura en disolución consiste en una forma B-ADN distorsionada con una gran curvatura, la estructura en monocristal consiste en un segmento central en forma Z-ADN con las bases de los nucleótidos terminales orientados hacia el exterior y formando puentes de hidrógeno con las bases de la molécula adyacente dando lugar a un tetrámero. La evidente simetría mostrada en los espectros de RMN por el sistema descarta dicha estructura en disolución.

La estructura en disolución a altas concentraciones de Na^+ del hexanucleótido no parece presentar grandes variaciones conformacionales respecto a la estructura a baja fuerza iónica. La simetría con centro de inversión del sistema se mantiene eliminando la posibilidad de formación de tetrámeros.

El análisis de la conformación de los azúcares mediante simulación de picos DQF-COSY muestra que los anillos de desoxiribosa de los nucleótidos terminales C1, C7, G6 y G12 muestran una movilidad elevada. Los desplazamientos químicos del residuo G6 muestran una inversión respecto a la situación habitual delatando conformaciones poco habituales en el mismo. Los valores de constantes de acoplamiento obtenidas indican un estructura con mayoría de residuos en situación de interconversión de conformaciones del azúcar. Tan solo el nucleótido C4 muestra una conformación estática.

Los cálculos de dinámica molecular confirman la movilidad observada en el análisis de las constantes de acoplamiento medidas. Cuando sólo se introducen las restricciones de ángulos diedros, dicha movilidad produce situaciones muy distorsionadas con cierto parecido a la estructura de monocristal del hexanucleótido. Esto parece indicar que la barrera de potencial entre las conformaciones en disolución y en monocristal no es muy elevada. Las interacciones de solvatación, demasiado complejas para simular con las técnicas de modelización disponibles, podrían estabilizar formas intermedias aunque.

Apéndice A

Código fuente del programa SEARCHEL

Se expone a continuación el codigo fuente del programa SEARCHEL como apéndice a la tesis. Se puede ver el nombre del fichero correspondiente al principio de cada línea:

```
constantes.h: #define GH 2.67519e8
constantes.h: #define GN -2.711e7
constantes.h: #define DNH 2.279055
constantes.h: #define DELTA -0.160
constantes.h: #ifndef PI
constantes.h: #define PI   3.14159265358979323846
constantes.h: #endif
constantes.h:
constantes.h: #define REXUNC 0.5
constantes.h: #define MAXLINE 1000
constantes.h: #define MAXRES 100
constantes.h: #define MAXCICLE 20
constantes.h: #define TMGRID 0.05
constantes.h: #define REXMAX 30.0
constantes.h: #define TOL 0.5*0.5
constantes.h: #define NEGTOL -0.500
constantes.h: #define POSTOL 0.0
constantes.h: #define GRIDSIZE_TE 0.01
constantes.h: #define GRIDSIZE_S2 0.01
constantes.h: #define GRIDSIZE_REX 0.1
```

```
functions.h: struct para model2(float te, float s2, float tm, float wh);
functions.h: struct para model3(float rex, float s2, float tm, float wh);
functions.h: struct para model4(float te, float s2, float rex, float tm, float wh);
functions.h: struct para model5(float te, float ss2, float sf2, float tm, float wh);
functions.h: struct para model6(float te, float ss2, float sf2, float rex, float tm, float wh);
```

```
global.h: struct para {
global.h:       float R1;
global.h:       float R2;
global.h:       float HNOE;
```

```
global.h: };
global.h:
global.h: struct sol {
global.h:       float max;
global.h:       float min;
global.h:       float mean;
global.h: };
global.h:
global.h: struct solution {
global.h:       struct sol sols2;
global.h:       struct sol solsf2;
global.h:       struct sol solss2;
global.h:       struct sol solte;
global.h:       struct sol solrex;
global.h:       float percent;
global.h:       int test;
global.h: };
global.h:
global.h: struct residue {
global.h:       int model;
global.h:       struct para respara;
global.h: };
global.h:
global.h:
global.h: struct solution eval_model(int typemodel, struct para paraexp, struct para uncert, float tm , float w);
global.h: int check_model(int typemodel, struct para paraexp, struct para uncert, float tm , float w);
global.h: struct solution eval_modelf(int typemodel, struct para paraexp, struct para uncert, float tm , FILE *fpout, fl
oat w);
global.h: float eval_rex(struct para paraexp, struct para uncert);
global.h: int test_para(struct para paraexp, struct para uncert, struct para parateor);
global.h: int getline(FILE *fpinp,char s[],int lim);
global.h: float tmmed(float tminit, struct para residue[], int test, int m2[], int nres);
global.h: float ftm(float tm, float wh, float wx);
global.h: float jwmf(float w,float tm);
global.h: float tmfr2r1(float tm, float r2r1, float min, float max,float wh, float wx);
global.h: struct sol marg(float r2r1,float x1,float  x2, float wh, float wx);
global.h:
global.h:
global.h: float jsim(float s2, float w, float tm);
global.h: float jext(float sf2, float ss2, float w, float tm, float te);
global.h: float j(float s2, float w, float tm, float te);
global.h:
global.h:
global.h: struct para model1(float s2, float tm, float wh);
global.h: struct para model2(float te, float s2, float tm, float wh);
global.h: struct para model3(float rex, float s2, float tm, float wh);
global.h: struct para model4(float te, float s2, float rex, float tm, float wh);
global.h: struct para model5(float te, float ss2, float sf2, float tm, float wh);
global.h: struct para model6(float te, float ss2, float sf2, float rex, float tm, float wh);
global.h:
global.h: struct sol calctm(FILE *fpinp, float gridtm, float limit, float w);
global.h: void calcall(float gtmmin, float gtmmax, FILE *fpinp, float w, float limit, char fileout[]);
global.h: void calctree(float gtmmin, float gtmmax, FILE *fpinp, float w, float limit, char fileout[]);
global.h: void calcone(float gtmmin, float gtmmax, FILE *fpinp, float w, float limit, char fileout[]);
global.h: void simul(float tm, float w, FILE *fpinp, char output[]);
global.h: void init(char bandera[]);
global.h:
global.h:          float negtol;
global.h:          float postol;
global.h:          float rexmaxg;
global.h:          float fgridsize_s2;
global.h:          float fgridsize_ss2;
global.h:          float fgridsize_sf2;
global.h:          float fgridsize_te;
global.h:          float fgridsize_rex;
global.h:
global.h: #define absmio(X)  ((X) > 0 ? (X) : (-X))
```

```
search.h: #define  REXUNC  0.5
search.h: struct para {
search.h:       float R1;
search.h:       float R2;
search.h:       float HNOE;
```

```
search.h: };
search.h:
search.h: struct residue {
search.h:       int model;
search.h:       struct para respara;
search.h: };
```

```
calcall.c: #include <math.h>
calcall.c: #include <stdio.h>
calcall.c: #include <string.h>
calcall.c: /*
calcall.c: #include "search.h"
calcall.c: #include "functions.h"
calcall.c: #include "string.h"
calcall.c:
calcall.c: */
calcall.c:
calcall.c: #include "global.h"
calcall.c: #include "constantes.h"
calcall.c:
calcall.c: void calcall(float gtmmin, float gtmmax, FILE *fpinp, float w, float limit, char fileout[MAXLINE])
calcall.c: {
calcall.c:
calcall.c:         extern float negtol;
calcall.c:         extern float postol;
calcall.c:         extern float rexmaxg;
calcall.c:         extern float fgridsize_s2;
calcall.c:         extern float fgridsize_ss2;
calcall.c:         extern float fgridsize_sf2;
calcall.c:         extern float fgridsize_te;
calcall.c:         extern float fgridsize_rex;
calcall.c:
calcall.c:
calcall.c: struct para residue[MAXRES];
calcall.c: struct para parateor;
calcall.c: struct para residue_unc[MAXRES];
calcall.c: struct solution solucion, res_sol[MAXRES][6];
calcall.c: float tm;
calcall.c: FILE *fpout1;
calcall.c: FILE *nosolution;
calcall.c: char filename2[MAXLINE];
calcall.c: char filenosol[MAXLINE];
calcall.c: char line[MAXLINE];
calcall.c: char resname[MAXRES][10];
calcall.c: int mod, first;
calcall.c: float percentage,tmtotal;
calcall.c: float gridtm = TMGRID;
calcall.c: int nres;
calcall.c: int i=1;
calcall.c: int solucionado;
calcall.c:
calcall.c:
calcall.c:         sprintf(filenosol,"nosolution_%4.2f",limit);
calcall.c: nosolution=fopen(filenosol,"w");
calcall.c:
calcall.c:
calcall.c:         while(getline(fpinp,line,sizeof(line)) > 0)
calcall.c: {
calcall.c: sscanf(line, "%s %f %f %f %f %f %f \n", &resname[i], &residue[i].R1, &residue_unc[i].R1, &residue[i].R2, &re
sidue_unc[i].R2, &residue[i].HNOE, &residue_unc[i].HNOE);
calcall.c:
calcall.c:               if(residue_unc[i].R1/residue[i].R1 < limit)
calcall.c:                   residue_unc[i].R1=absmio(residue[i].R1*limit);
calcall.c:               if(residue_unc[i].R2/residue[i].R2 < limit)
calcall.c:                   residue_unc[i].R2=absmio(residue[i].R2*limit);
calcall.c:               if(residue_unc[i].HNOE/residue[i].HNOE < limit)
calcall.c:                   residue_unc[i].HNOE=absmio(residue[i].HNOE*limit);
calcall.c:       i++;
calcall.c:
calcall.c:        }
calcall.c:         nres=i-1;
calcall.c:
calcall.c: for(i=1;i<=nres;i++)
```

```
calcall.c:   {
calcall.c:    solucionado=0;
calcall.c:    fprintf(stdout,"SCH> Residuo %d  %s\n",i, resname[i]);
calcall.c:    for(mod=1;mod<=5;mod++)
calcall.c:     {
calcall.c:      fprintf(stdout,"SCH> Modelo %d\n",mod);
calcall.c:      percentage=0.0;
calcall.c:      tmtotal=0.0;
calcall.c:      first=1;
calcall.c:      sprintf(filename2,"%s_%s_mod%d",fileout,resname[i],mod);
calcall.c:      fpout1 = fopen(filename2, "w");
calcall.c:      for(tm=gtmmin;tm <=gtmmax;tm=tm+gridtm)
calcall.c:       {
calcall.c:       fprintf(stdout,"SCH> Tm %6.3f\n",tm);
calcall.c:       solucion=eval_modelf(mod,residue[i],residue_unc[i],tm,fpout1,w);
calcall.c:
calcall.c:       if((first == 1)&&(solucion.percent != 0.0))
calcall.c:        {
calcall.c:   first=0;
calcall.c:   res_sol[i][mod]=solucion; }
calcall.c:                else
calcall.c:   {
calcall.c:   if(solucion.percent != 0.0)
calcall.c:   {
calcall.c:    if( solucion.sols2.max > res_sol[i][mod].sols2.max )
calcall.c:       res_sol[i][mod].sols2.max=solucion.sols2.max;
calcall.c:    if( solucion.sols2.min < res_sol[i][mod].sols2.min )
calcall.c:       res_sol[i][mod].sols2.min=solucion.sols2.min;
calcall.c:    if( solucion.solss2.max > res_sol[i][mod].solss2.max )
calcall.c:       res_sol[i][mod].solss2.max=solucion.solss2.max;
calcall.c:    if( solucion.solss2.min < res_sol[i][mod].solss2.min )
calcall.c:       res_sol[i][mod].solss2.min=solucion.solss2.min;
calcall.c:    if( solucion.solsf2.max > res_sol[i][mod].solsf2.max )
calcall.c:       res_sol[i][mod].solsf2.max=solucion.solsf2.max;
calcall.c:    if( solucion.solsf2.min < res_sol[i][mod].solsf2.min )
calcall.c:       res_sol[i][mod].solsf2.min=solucion.solsf2.min;
calcall.c:    if( solucion.solte.max > res_sol[i][mod].solte.max )
calcall.c:       res_sol[i][mod].solte.max=solucion.solte.max;
calcall.c:    if( solucion.solte.min < res_sol[i][mod].solte.min )
calcall.c:       res_sol[i][mod].solte.min=solucion.solte.min;
calcall.c:    if( solucion.solrex.max > res_sol[i][mod].solrex.max )
calcall.c:       res_sol[i][mod].solrex.max=solucion.solrex.max;
calcall.c:    if( solucion.solrex.min < res_sol[i][mod].solrex.min )
calcall.c:       res_sol[i][mod].solrex.min=solucion.solrex.min;
calcall.c:
calcall.c:       }
calcall.c:                   }
calcall.c:     percentage=percentage+solucion.percent*tm/TMGRID;
calcall.c:     tmtotal=tmtotal+tm;
calcall.c:     }
calcall.c:     fclose(fpout1);
calcall.c: /*            fprintf(stdout,"SCH> Aqui llego V\n"); */
calcall.c:
calcall.c:             if(( res_sol[i][mod].solte.max > 0.0 )&&(res_sol[i][mod].solte.min < 0.0))
calcall.c:                 res_sol[i][mod].solte.min=0.0;
calcall.c:
calcall.c:
calcall.c:
calcall.c:     res_sol[i][mod].percent=percentage/(tmtotal*TMGRID);
calcall.c:             res_sol[i][mod].sols2.mean=(res_sol[i][mod].sols2.max+res_sol[i][mod].sols2.min)/2;
calcall.c:             res_sol[i][mod].solss2.mean=(res_sol[i][mod].solss2.max+res_sol[i][mod].solss2.min)/2;
calcall.c:             res_sol[i][mod].solsf2.mean=(res_sol[i][mod].solsf2.max+res_sol[i][mod].solsf2.min)/2;
calcall.c:             res_sol[i][mod].solte.mean=(res_sol[i][mod].solte.max+res_sol[i][mod].solte.min)/2;
calcall.c:             res_sol[i][mod].solrex.mean=(res_sol[i][mod].solrex.max+res_sol[i][mod].solrex.min)/2;
calcall.c:             res_sol[i][mod].test=0;
calcall.c:
calcall.c:                for(tm=gtmmin;tm <=gtmmax;tm=tm+TMGRID)
calcall.c:                {
calcall.c:                switch(mod)
calcall.c:                {
calcall.c:                   case 1:
calcall.c:                          parateor=model1(res_sol[i][mod].sols2.mean, tm, w);
calcall.c:                          break;
calcall.c:                   case 2:
calcall.c:                          parateor=model2(res_sol[i][mod].solte.mean, res_sol[i][mod].sols2.mean, tm, w);
calcall.c:                          break;
```

```
calcall.c:                    case 3:
calcall.c:                           parateor=model3(res_sol[i][mod].solrex.mean, res_sol[i][mod].sols2.mean, tm, w);
calcall.c:                           break;
calcall.c:                    case 4:
calcall.c:                           parateor=model4(res_sol[i][mod].solte.mean, res_sol[i][mod].sols2.mean, res_sol[i][
mod].solrex.mean, tm, w);
calcall.c:                           break;
calcall.c:                    case 5:
calcall.c:                           parateor=model5(res_sol[i][mod].solte.mean, res_sol[i][mod].solss2.mean, res_sol[i]
[mod].solsf2.mean, tm, w);
calcall.c:                           break;
calcall.c:                    case 6:
calcall.c:                           parateor=model6(res_sol[i][mod].solte.mean, res_sol[i][mod].solss2.mean, res_sol[i]
[mod].solsf2.mean,res_sol[i][mod].solrex.mean,  tm,
calcall.c: w);
calcall.c:                           break;
calcall.c:                           }
calcall.c:                   if(test_para(residue[i], residue_unc[i], parateor)==1)
calcall.c:                   {
calcall.c:                    res_sol[i][mod].test=1;
calcall.c:                    break;
calcall.c:                    }
calcall.c:                  }
calcall.c:
calcall.c:            fprintf(stdout,"SCH> Residuo %d Modelo %d\n",i,mod);
calcall.c:            fprintf(stdout,"SCH> ============================\n");
calcall.c:            fprintf(stdout,"SCH> \n");
calcall.c:
calcall.c:    if(percentage != 0.0)
calcall.c:    {
calcall.c:    solucionado=1;
calcall.c:            switch(mod)
calcall.c:            {
calcall.c:               case 1:
calcall.c:    fprintf(stdout,"SCHSOL | %s > s2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].sols2.mean,res_sol[i][m
od].sols2.min,res_sol[i][mod].sols2.max);
calcall.c:    fprintf(stdout,"SCHSOL> \n");
calcall.c:               case 2:
calcall.c:    fprintf(stdout,"SCHSOL | %s > s2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].sols2.mean,res_sol[i][m
od].sols2.min,res_sol[i][mod].sols2.max);
calcall.c:    fprintf(stdout,"SCHSOL | %s > te %6.3f %6.3f %6.3f ",resname[i],res_sol[i][mod].solte.mean,res_sol[i][mod
].solte.min,res_sol[i][mod].solte.max);
calcall.c:            if(res_sol[i][mod].solte.max < 0.0)
calcall.c:               fprintf(stdout,"  *******\n");
calcall.c:            else
calcall.c:               fprintf(stdout,"\n");
calcall.c:
calcall.c:    fprintf(stdout,"SCHSOL> \n");
calcall.c:            break;
calcall.c:       case 3:
calcall.c:    fprintf(stdout,"SCHSOL | %s > s2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].sols2.mean,res_sol[i][m
od].sols2.min,res_sol[i][mod].sols2.max);
calcall.c:    fprintf(stdout,"SCHSOL | %s > rex %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solrex.mean,res_sol[i]
[mod].solrex.min,res_sol[i][mod].solrex.max);
calcall.c:    fprintf(stdout,"SCHSOL> \n");
calcall.c:    break;
calcall.c:    case 4:
calcall.c:    fprintf(stdout,"SCHSOL | %s > s2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].sols2.mean,res_sol[i][m
od].sols2.min,res_sol[i][mod].sols2.max);
calcall.c:    fprintf(stdout,"SCHSOL | %s > te %6.3f %6.3f %6.3f ",resname[i],res_sol[i][mod].solte.mean,res_sol[i][mod
].solte.min,res_sol[i][mod].solte.max);
calcall.c:            if(res_sol[i][mod].solte.max < 0.0)
calcall.c:               fprintf(stdout,"  *******\n");
calcall.c:            else
calcall.c:               fprintf(stdout,"\n");
calcall.c:
calcall.c:    fprintf(stdout,"SCHSOL | %s > rex %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solrex.mean,res_sol[i]
[mod].solrex.min,res_sol[i][mod].solrex.max);
calcall.c:    fprintf(stdout,"SCHSOL> \n");
calcall.c:    break;
calcall.c:
calcall.c:    case 5:
calcall.c:    fprintf(stdout,"SCHSOL | %s > ss2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solss2.mean,res_sol[i]
[mod].solss2.min,res_sol[i][mod].solss2.max);
calcall.c:    fprintf(stdout,"SCHSOL | %s > sf2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solsf2.mean,res_sol[i]
[mod].solsf2.min,res_sol[i][mod].solsf2.max);
```

```
calcall.c:    fprintf(stdout,"SCHSOL | %s > te %6.3f %6.3f %6.3f ",resname[i],res_sol[i][mod].solte.mean,res_sol[i][mod
].solte.min,res_sol[i][mod].solte.max);
calcall.c:            if(res_sol[i][mod].solte.max < 0.0)
calcall.c:               fprintf(stdout,"  *******\n");
calcall.c:            else
calcall.c:               fprintf(stdout,"\n");
calcall.c:
calcall.c:    fprintf(stdout,"SCHSOL> \n");
calcall.c:    break;
calcall.c:
calcall.c:    case 6:
calcall.c:    fprintf(stdout,"SCHSOL | %s > ss2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solss2.mean,res_sol[i]
[mod].solss2.min,res_sol[i][mod].solss2.max);
calcall.c:    fprintf(stdout,"SCHSOL | %s > sf2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solsf2.mean,res_sol[i]
[mod].solsf2.min,res_sol[i][mod].solsf2.max);
calcall.c:    fprintf(stdout,"SCHSOL | %s > te %6.3f %6.3f %6.3f ",resname[i],res_sol[i][mod].solte.mean,res_sol[i][mod
].solte.min,res_sol[i][mod].solte.max);
calcall.c:            if(res_sol[i][mod].solte.max < 0.0)
calcall.c:               fprintf(stdout,"  *******\n");
calcall.c:            else
calcall.c:               fprintf(stdout,"\n");
calcall.c:
calcall.c:    fprintf(stdout,"SCHSOL | %s > rex %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solrex.mean,res_sol[i]
[mod].solrex.min,res_sol[i][mod].solrex.max);
calcall.c:    fprintf(stdout,"SCHSOL> \n");
calcall.c:    break;
calcall.c: }
calcall.c:
calcall.c: /*    fprintf(stdout,"SCHSOL | %s > Percent %6.3f \n",resname[i],res_sol[i][mod].percent);
calcall.c: */
calcall.c:    fprintf(stdout,"SCHSOL | %s > Percent %6.3f \n",resname[i],percentage/tmtotal);
calcall.c:
calcall.c:    fprintf(stdout,"SCHSOL | %s > Test %d \n",resname[i],res_sol[i][mod].test);
calcall.c:    fprintf(stdout,"SCHSOL> \n");
calcall.c:     }
calcall.c:     else
calcall.c:   fprintf(stdout,"SCHSOL | %s > No solution...  \n",resname[i]);
calcall.c:     }
calcall.c:    if( solucionado == 0)
calcall.c: {  fprintf(stdout,"SCHSOL | %s > No solution for any model  \n",resname[i]);
calcall.c:            fprintf(nosolution,"%s %6.3f %6.3f %6.3f %6.3f %6.3f %6.3f %6.3f \n",resname[i], residue[i].R1, residue_
unc[i].R1, residue[i].R2, residue_unc[i].R2, residue[i].HNOE, residue_unc[i].HNOE);
calcall.c:    fflush(nosolution);
calcall.c:    fflush(stdout);
calcall.c:    fflush(stderr);
calcall.c:    }
calcall.c:   }
calcall.c:         fclose(nosolution);
calcall.c: return;
calcall.c: }
```

```
calcone.c: #include <math.h>
calcone.c: #include <stdio.h>
calcone.c: #include <string.h>
calcone.c: /*
calcone.c: #include "search.h"
calcone.c: #include "functions.h"
calcone.c: #include "string.h"
calcone.c:
calcone.c: */
calcone.c:
calcone.c: #include "global.h"
calcone.c: #include "constantes.h"
calcone.c:
calcone.c: void calcone(float gtmmin, float gtmmax, FILE *fpinp, float w, float limit, char fileout[MAXLINE])
calcone.c: {
calcone.c:
calcone.c:
calcone.c:         extern float negtol;
calcone.c:         extern float postol;
calcone.c:         extern float rexmaxg;
calcone.c:         extern float fgridsize_s2;
calcone.c:         extern float fgridsize_ss2;
```

```
calcone.c:         extern float fgridsize_sf2;
calcone.c:         extern float fgridsize_te;
calcone.c:         extern float fgridsize_rex;
calcone.c:
calcone.c:
calcone.c: struct para residue[MAXRES];
calcone.c: struct para parateor;
calcone.c: struct para residue_unc[MAXRES];
calcone.c: struct solution solucion, res_sol[MAXRES][6];
calcone.c: float tm;
calcone.c: FILE *fpout1;
calcone.c: char filename2[MAXLINE];
calcone.c: char line[MAXLINE];
calcone.c: char resname[MAXRES][10];
calcone.c: int mode[MAXRES];
calcone.c: int mod, first;
calcone.c: float percentage,tmtotal;
calcone.c: float gridtm = TMGRID;
calcone.c: int nres;
calcone.c: int i=1;
calcone.c:
calcone.c:
calcone.c:         while(getline(fpinp,line,sizeof(line)) > 0)
calcone.c: {
calcone.c: sscanf(line, "%s %d %f %f %f %f %f %f \n",
calcone.c:       &resname[i], &mode[i],
calcone.c:               &residue[i].R1, &residue_unc[i].R1,
calcone.c:       &residue[i].R2, &residue_unc[i].R2,
calcone.c:       &residue[i].HNOE, &residue_unc[i].HNOE);
calcone.c:
calcone.c:               if(residue_unc[i].R1/residue[i].R1 < limit)
calcone.c:                   residue_unc[i].R1=absmio(residue[i].R1*limit);
calcone.c:               if(residue_unc[i].R2/residue[i].R2 < limit)
calcone.c:                   residue_unc[i].R2=absmio(residue[i].R2*limit);
calcone.c:               if(residue_unc[i].HNOE/residue[i].HNOE < limit)
calcone.c:                   residue_unc[i].HNOE=absmio(residue[i].HNOE*limit);
calcone.c:       i++;
calcone.c:
calcone.c:        }
calcone.c:         nres=i-1;
calcone.c:
calcone.c: for(i=1;i<=nres;i++)
calcone.c:   {
calcone.c:    fprintf(stdout,"SEARCHER> Residuo %d  %s\n",i, resname[i]);
calcone.c:      mod=mode[i];
calcone.c:      fprintf(stdout,"SEARCHER> Modelo %d\n",mod);
calcone.c:      percentage=0.0;
calcone.c:      tmtotal=0.0;
calcone.c:      first=1;
calcone.c:      sprintf(filename2,"%s_%s_mod%d",fileout,resname[i],mod);
calcone.c:      fpout1 = fopen(filename2, "w");
calcone.c:      for(tm=gtmmin;tm <=gtmmax;tm=tm+gridtm)
calcone.c:       {
calcone.c:       fprintf(stdout,"SEARCHER> Tm %f\n",tm);
calcone.c:       solucion=eval_modelf(mod,residue[i],residue_unc[i],tm,fpout1,w);
calcone.c:
calcone.c:       if((first == 1)&&(solucion.percent != 0.0))
calcone.c:        {
calcone.c:   first=0;
calcone.c:   res_sol[i][mod]=solucion; }
calcone.c:               else
calcone.c:   {
calcone.c:   if(solucion.percent != 0.0)
calcone.c:   {
calcone.c:    if( solucion.sols2.max > res_sol[i][mod].sols2.max )
calcone.c:       res_sol[i][mod].sols2.max=solucion.sols2.max;
calcone.c:    if( solucion.sols2.min < res_sol[i][mod].sols2.min )
calcone.c:       res_sol[i][mod].sols2.min=solucion.sols2.min;
calcone.c:    if( solucion.solss2.max > res_sol[i][mod].solss2.max )
calcone.c:       res_sol[i][mod].solss2.max=solucion.solss2.max;
calcone.c:    if( solucion.solss2.min < res_sol[i][mod].solss2.min )
calcone.c:       res_sol[i][mod].solss2.min=solucion.solss2.min;
calcone.c:    if( solucion.solsf2.max > res_sol[i][mod].solsf2.max )
calcone.c:       res_sol[i][mod].solsf2.max=solucion.solsf2.max;
calcone.c:    if( solucion.solsf2.min < res_sol[i][mod].solsf2.min )
calcone.c:       res_sol[i][mod].solsf2.min=solucion.solsf2.min;
```

```
calcone.c:    if( solucion.solte.max > res_sol[i][mod].solte.max )
calcone.c:       res_sol[i][mod].solte.max=solucion.solte.max;
calcone.c:    if( solucion.solte.min < res_sol[i][mod].solte.min )
calcone.c:       res_sol[i][mod].solte.min=solucion.solte.min;
calcone.c:    if( solucion.solrex.max > res_sol[i][mod].solrex.max )
calcone.c:       res_sol[i][mod].solrex.max=solucion.solrex.max;
calcone.c:    if( solucion.solrex.min < res_sol[i][mod].solrex.min )
calcone.c:       res_sol[i][mod].solrex.min=solucion.solrex.min;
calcone.c:       }
calcone.c:                   }
calcone.c:      percentage=percentage+solucion.percent*tm/TMGRID;
calcone.c:      tmtotal=tmtotal+tm;
calcone.c:      }
calcone.c:      fclose(fpout1);
calcone.c: /*            fprintf(stdout,"SEARCHER> Aqui llego V\n"); */
calcone.c:
calcone.c:
calcone.c:
calcone.c:      if(( res_sol[i][mod].solte.max > 0.0 )&&(res_sol[i][mod].solte.min < 0.0))
calcone.c:       res_sol[i][mod].solte.min=0.0;
calcone.c:
calcone.c:
calcone.c:
calcone.c:      res_sol[i][mod].percent=percentage/(tmtotal*TMGRID);
calcone.c:              res_sol[i][mod].sols2.mean=(res_sol[i][mod].sols2.max+res_sol[i][mod].sols2.min)/2;
calcone.c:              res_sol[i][mod].solss2.mean=(res_sol[i][mod].solss2.max+res_sol[i][mod].solss2.min)/2;
calcone.c:              res_sol[i][mod].solsf2.mean=(res_sol[i][mod].solsf2.max+res_sol[i][mod].solsf2.min)/2;
calcone.c:              res_sol[i][mod].solte.mean=(res_sol[i][mod].solte.max+res_sol[i][mod].solte.min)/2;
calcone.c:              res_sol[i][mod].solrex.mean=(res_sol[i][mod].solrex.max+res_sol[i][mod].solrex.min)/2;
calcone.c:              res_sol[i][mod].test=0;
calcone.c:
calcone.c:                 for(tm=gtmmin;tm <=gtmmax;tm=tm+TMGRID)
calcone.c:                 {
calcone.c:                 switch(mod)
calcone.c:                 {
calcone.c:                    case 2:
calcone.c:                           parateor=model2(res_sol[i][mod].solte.mean, res_sol[i][mod].sols2.mean, tm, w);
calcone.c:                           break;
calcone.c:                    case 3:
calcone.c:                           parateor=model3(res_sol[i][mod].solrex.mean, res_sol[i][mod].sols2.mean, tm, w);
calcone.c:                           break;
calcone.c:                    case 4:
calcone.c:                           parateor=model4(res_sol[i][mod].solte.mean, res_sol[i][mod].sols2.mean, res_sol[i][
mod].solrex.mean, tm, w);
calcone.c:                           break;
calcone.c:                    case 5:
calcone.c:                           parateor=model5(res_sol[i][mod].solte.mean, res_sol[i][mod].solss2.mean, res_sol[i]
[mod].solsf2.mean, tm, w);
calcone.c:                           break;
calcone.c:                    case 6:
calcone.c:                           parateor=model6(res_sol[i][mod].solte.mean, res_sol[i][mod].solss2.mean, res_sol[i]
[mod].solsf2.mean,res_sol[i][mod].solrex.mean,  tm,
calcone.c: w);
calcone.c:                           break;
calcone.c:                           }
calcone.c:                  if(test_para(residue[i], residue_unc[i], parateor)==1)
calcone.c:                  {
calcone.c:                   res_sol[i][mod].test=1;
calcone.c:                   break;
calcone.c:                   }
calcone.c:                  }
calcone.c:            fprintf(stdout,"SCH> Residuo %d Modelo %d\n",i,mod);
calcone.c:            fprintf(stdout,"SCH> ============================\n");
calcone.c:            fprintf(stdout,"SCH> \n");
calcone.c:            switch(mod)
calcone.c:            {
calcone.c:               case 2:
calcone.c:            fprintf(stdout,"SCH> s2 %6.3f %6.3f %6.3f \n",res_sol[i][mod].sols2.mean,res_sol[i][mod].sols2.min
,res_sol[i][mod].sols2.max);
calcone.c:            fprintf(stdout,"SCH> te %6.3f %6.3f %6.3f ",res_sol[i][mod].solte.mean,res_sol[i][mod].solte.min,r
es_sol[i][mod].solte.max);
calcone.c:            if(res_sol[i][mod].solte.max < 0.0)
calcone.c:               fprintf(stdout,"  *******\n");
calcone.c:            else
calcone.c:               fprintf(stdout,"\n");
calcone.c:
```

```
calcone.c:            fprintf(stdout,"SCH> \n");
calcone.c:            break;
calcone.c:               case 3:
calcone.c:            fprintf(stdout,"SCH> s2 %6.3f %6.3f %6.3f \n",res_sol[i][mod].sols2.mean,res_sol[i][mod].sols2.min
,res_sol[i][mod].sols2.max);
calcone.c:            fprintf(stdout,"SCH> rex %6.3f %6.3f %6.3f \n",res_sol[i][mod].solrex.mean,res_sol[i][mod].solrex.
min,res_sol[i][mod].solrex.max);
calcone.c:            fprintf(stdout,"SCH> \n");
calcone.c:            break;
calcone.c:            case 4:
calcone.c:            fprintf(stdout,"SCH> s2 %6.3f %6.3f %6.3f \n",res_sol[i][mod].sols2.mean,res_sol[i][mod].sols2.min
,res_sol[i][mod].sols2.max);
calcone.c:            fprintf(stdout,"SCH> te %6.3f %6.3f %6.3f ",res_sol[i][mod].solte.mean,res_sol[i][mod].solte.min,r
es_sol[i][mod].solte.max);
calcone.c:            if(res_sol[i][mod].solte.max < 0.0)
calcone.c:               fprintf(stdout,"  *******\n");
calcone.c:            else
calcone.c:               fprintf(stdout,"\n");
calcone.c:
calcone.c:            fprintf(stdout,"SCH> rex %6.3f %6.3f %6.3f \n",res_sol[i][mod].solrex.mean,res_sol[i][mod].solrex.
min,res_sol[i][mod].solrex.max);
calcone.c:            fprintf(stdout,"SCH> \n");
calcone.c:            break;
calcone.c:
calcone.c:            case 5:
calcone.c:            fprintf(stdout,"SCH> ss2 %6.3f %6.3f %6.3f \n",res_sol[i][mod].solss2.mean,res_sol[i][mod].solss2.
min,res_sol[i][mod].solss2.max);
calcone.c:            fprintf(stdout,"SCH> sf2 %6.3f %6.3f %6.3f \n",res_sol[i][mod].solsf2.mean,res_sol[i][mod].solsf2.
min,res_sol[i][mod].solsf2.max);
calcone.c:            fprintf(stdout,"SCH> te %6.3f %6.3f %6.3f ",res_sol[i][mod].solte.mean,res_sol[i][mod].solte.min,r
es_sol[i][mod].solte.max);
calcone.c:            if(res_sol[i][mod].solte.max < 0.0)
calcone.c:               fprintf(stdout,"  *******\n");
calcone.c:            else
calcone.c:               fprintf(stdout,"\n");
calcone.c:
calcone.c:            fprintf(stdout,"SCH> \n");
calcone.c:            break;
calcone.c:            case 6:
calcone.c:            fprintf(stdout,"SCH> ss2 %6.3f %6.3f %6.3f \n",res_sol[i][mod].solss2.mean,res_sol[i][mod].solss2.
min,res_sol[i][mod].solss2.max);
calcone.c:            fprintf(stdout,"SCH> sf2 %6.3f %6.3f %6.3f \n",res_sol[i][mod].solsf2.mean,res_sol[i][mod].solsf2.
min,res_sol[i][mod].solsf2.max);
calcone.c:            fprintf(stdout,"SCH> te %6.3f %6.3f %6.3f ",res_sol[i][mod].solte.mean,res_sol[i][mod].solte.min,r
es_sol[i][mod].solte.max);
calcone.c:            if(res_sol[i][mod].solte.max < 0.0)
calcone.c:               fprintf(stdout,"  *******\n");
calcone.c:            else
calcone.c:               fprintf(stdout,"\n");
calcone.c:
calcone.c:            fprintf(stdout,"SCH> rex %6.3f %6.3f %6.3f \n",res_sol[i][mod].solrex.mean,res_sol[i][mod].solrex.
min,res_sol[i][mod].solrex.max);
calcone.c:            fprintf(stdout,"SCH> \n");
calcone.c:            break;
calcone.c: }
calcone.c:
calcone.c:
calcone.c:
calcone.c:
calcone.c:
calcone.c:    fprintf(stdout,"SEARCHER> Percent %f \n",percentage/tmtotal);
calcone.c:
calcone.c:    fprintf(stdout,"SEARCHER> Test %d \n",res_sol[i][mod].test);
calcone.c:    fprintf(stdout,"SEARCHER> \n");
calcone.c:    }
calcone.c:
calcone.c:
calcone.c: return;
calcone.c: }
```

```
calctm.c: #include <math.h>
calctm.c: #include <stdio.h>
calctm.c: #include <string.h>
```

```
calctm.c: #include "global.h"
calctm.c: #include "constantes.h"
calctm.c:
calctm.c: struct sol calctm(FILE *fpinp, float gridtm, float limit, float w)
calctm.c: {
calctm.c:
calctm.c:         extern float negtol;
calctm.c:         extern float postol;
calctm.c:         extern float rexmax;
calctm.c:         extern float fgridsize_s2;
calctm.c:         extern float fgridsize_ss2;
calctm.c:         extern float fgridsize_sf2;
calctm.c:         extern float fgridsize_te;
calctm.c:         extern float fgridsize_rex;
calctm.c:
calctm.c: struct para residue[MAXRES];
calctm.c: struct para residue_unc[MAXRES];
calctm.c: struct sol bounds;
calctm.c: struct sol tmsol;
calctm.c: float tm;
calctm.c: char line[MAXLINE];
calctm.c: char resname[MAXRES][10];
calctm.c: int m2[MAXRES], oldm2[MAXRES];
calctm.c: int found=0;
calctm.c: int difm2=0;
calctm.c: int ini=0;
calctm.c:         int imin=0;
calctm.c: int conteo, conteo2;
calctm.c: float hist[MAXCICLE];
calctm.c:         float uncsq, uncsqmin;
calctm.c: float r2r1med;
calctm.c:         float r2r1sum=0.0;
calctm.c: float tmmax, tmmin, tminit;
calctm.c:         float gtmmin, gtmmax;
calctm.c:         float gtmminlast, gtmmaxlast;
calctm.c: int nres;
calctm.c: int i=1;
calctm.c:
calctm.c: int *m2ptr;
calctm.c:
calctm.c: gtmmin=0.0;
calctm.c: gtmmax=100.0;
calctm.c:
calctm.c:         while(getline(fpinp,line,sizeof(line)) > 0)
calctm.c: {
calctm.c: sscanf(line, "%s %f %f %f %f %f %f \n",
calctm.c:       &resname[i],
calctm.c:               &residue[i].R1, &residue_unc[i].R1,
calctm.c:       &residue[i].R2, &residue_unc[i].R2,
calctm.c:       &residue[i].HNOE, &residue_unc[i].HNOE);
calctm.c:
calctm.c:               if(residue_unc[i].R1/residue[i].R1 < limit)
calctm.c:                   residue_unc[i].R1=residue[i].R1*limit;
calctm.c:               if(residue_unc[i].R2/residue[i].R2 < limit)
calctm.c:                   residue_unc[i].R2=residue[i].R2*limit;
calctm.c:               if(residue_unc[i].HNOE/residue[i].HNOE < limit)
calctm.c:                   residue_unc[i].HNOE=residue[i].HNOE*limit;
calctm.c:       i++;
calctm.c:
calctm.c:        }
calctm.c:         nres=i-1;
calctm.c:         fprintf(stdout, "CALCTM> Nres:  %d\n",nres);
calctm.c:
calctm.c:         r2r1sum=0;
calctm.c: for(i=1; i <= nres; i++)
calctm.c:     r2r1sum= r2r1sum + residue[i].R2/residue[i].R1;
calctm.c:
calctm.c:         r2r1med=r2r1sum/nres;
calctm.c:         m2ptr=&m2[0];
calctm.c: r2r1med=tmmed(r2r1med, residue,0,m2ptr,nres);
calctm.c:
calctm.c:
calctm.c:
calctm.c:
calctm.c: bounds=marg(r2r1med,r2r1med/1.8,r2r1med*1.8,w,GN/GH*w);
calctm.c: tminit=tmfr2r1(r2r1med*1.80*1e-9,r2r1med,bounds.min,bounds.max,w,GN/GH*w);
```

```
calctm.c:         fprintf(stdout,"CALCTM> Tm init\t:%f\n",tminit);
calctm.c:        /* printf("Tm init\t:%f\n",tminit); */
calctm.c:         fprintf(stdout,"CALCTM> W\t:%f\n",w);
calctm.c:
calctm.c:
calctm.c:         uncsqmin=100.0;
calctm.c:         for(i=1;i <= nres;i++)
calctm.c: {
calctm.c: if(check_model(2, residue[i], residue_unc[i], tminit, w)==1)
calctm.c:           {
calctm.c:      m2[i]=1;
calctm.c:              uncsq=residue_unc[i].R1*residue_unc[i].R1+
calctm.c:                    residue_unc[i].R2*residue_unc[i].R2+
calctm.c:                    residue_unc[i].HNOE*residue_unc[i].HNOE;
calctm.c:              if(uncsq < uncsqmin)
calctm.c:               {
calctm.c:                 uncsqmin=uncsq;
calctm.c:                 imin=i;
calctm.c:                }
calctm.c:             }
calctm.c:         else
calctm.c:      m2[i]=0;
calctm.c:      }
calctm.c:
calctm.c:         conteo=0;
calctm.c:         conteo2=0;
calctm.c:
calctm.c: while((difm2 > 0)||(ini==0))
calctm.c: {
calctm.c:
calctm.c: difm2=0;
calctm.c: ini = 1;
calctm.c:
calctm.c:         fprintf(stdout,"CALCTM> \n");
calctm.c:         fprintf(stdout,"CALCTM> Residues that fit model 2 for this tm \n");
calctm.c:         for(i=1; i <= nres; i++)
calctm.c:          {
calctm.c:     oldm2[i]=m2[i];
calctm.c:     if(m2[i]==1)
calctm.c:               fprintf(stdout,"%s ",resname[i]);
calctm.c:         }
calctm.c:         fprintf(stdout,"\nCALCTM> \n");
calctm.c:
calctm.c:    m2ptr=&m2[0];
calctm.c:         r2r1med=tmmed(tminit/1.80,residue,1, m2ptr,nres);
calctm.c: bounds=marg(r2r1med,r2r1med/1.8,r2r1med*1.8,w,GN/GH*w);
calctm.c: tminit=tmfr2r1(r2r1med*1.80*1e-9,r2r1med,bounds.min,bounds.max,w,GN/GH*w);
calctm.c:         fprintf(stdout,"CALCTM> New Tm init (calculated with model 2 residues)\t:%f\n",tminit);
calctm.c:
calctm.c: gtmmin=0.0;
calctm.c: gtmmax=100.0;
calctm.c:
calctm.c: tmmin=tminit-gridtm;
calctm.c: tmmax=tminit+gridtm;
calctm.c:
calctm.c:         while((check_model(2, residue[imin], residue_unc[imin], tmmin , w)==1) && (tmmin > gtmmin))
calctm.c:            tmmin=tmmin-gridtm;
calctm.c:         gtmmin = tmmin;
calctm.c:
calctm.c:         while((check_model(2, residue[imin], residue_unc[imin], tmmax , w)==1) && (tmmax < gtmmax))
calctm.c:     tmmax=tmmax+gridtm;
calctm.c:         gtmmax = tmmax;
calctm.c:
calctm.c:         fprintf(stdout,"CALCTM> \n");
calctm.c: fprintf(stdout,"CALCTM> Tm interval for least uncertainty model 2 residue ( %s )  :\t%f %f \n",resname[imin],
gtmmin,gtmmax);
calctm.c:
calctm.c:         for(i=1;i <= nres;i++)
calctm.c: {
calctm.c:
calctm.c: tmmin=tminit-gridtm;
calctm.c: tmmax=tminit+gridtm;
calctm.c:
calctm.c: if(m2[i]==1)
calctm.c: {
calctm.c:         while((check_model(2, residue[i], residue_unc[i], tmmin , w)==1) && (tmmin > gtmmin))
```

```
calctm.c:            tmmin=tmmin-gridtm;
calctm.c:
calctm.c:         gtmmin = tmmin;
calctm.c:
calctm.c:         while((check_model(2, residue[i], residue_unc[i], tmmax , w)==1) && (tmmax < gtmmax))
calctm.c:     tmmax=tmmax+gridtm;
calctm.c:
calctm.c:         gtmmax = tmmax;
calctm.c:      }
calctm.c: }
calctm.c:
calctm.c:         fprintf(stdout,"CALCTM> \n");
calctm.c: fprintf(stdout,"CALCTM> Tm interval compatible with all model 2 residues:\t%f %f \n",gtmmin,gtmmax);
calctm.c:
calctm.c:         uncsqmin=100.0;
calctm.c:         difm2=0;
calctm.c:
calctm.c: for(i=1;i <= nres;i++)
calctm.c: {
calctm.c: for(tm=gtmmin;tm <= gtmmax;tm=tm+gridtm)
calctm.c: {
calctm.c: if(check_model(2, residue[i], residue_unc[i], tm , w)==1)
calctm.c:          {
calctm.c:    found=1;
calctm.c:              uncsq=residue_unc[i].R1*residue_unc[i].R1+
calctm.c:                    residue_unc[i].R2*residue_unc[i].R2+
calctm.c:                    residue_unc[i].HNOE*residue_unc[i].HNOE;
calctm.c:              if(uncsq < uncsqmin)
calctm.c:               {
calctm.c:                 uncsqmin=uncsq;
calctm.c:                 imin=i;
calctm.c:                }
calctm.c:
calctm.c:            break;
calctm.c:           }
calctm.c:         else
calctm.c:    found=0;
calctm.c:    }
calctm.c:    m2[i]=found;
calctm.c:    difm2=difm2+(m2[i]-oldm2[i])*(m2[i]-oldm2[i]);
calctm.c:    }
calctm.c:    conteo++;
calctm.c:    hist[conteo]=gtmmin*gtmmax;
calctm.c:    for(conteo2=1;conteo2 < conteo; conteo2++)
calctm.c:      if( hist[conteo]==hist[conteo2] || conteo >= MAXCICLE )
calctm.c:        {
calctm.c:         fprintf(stdout,"CALCTM> A periodic jump has been detected ( %d )\n", conteo);
calctm.c:        /* fprintf(stdout,"CALCTM> A periodic jump has been detected  %d \n", conteo); */
calctm.c:         gtmmin=gtmminlast;
calctm.c:         gtmmax=gtmmaxlast;
calctm.c:         difm2=0;
calctm.c:         }
calctm.c:            gtmminlast=gtmmin;
calctm.c:            gtmmaxlast=gtmmax;
calctm.c:         }
calctm.c:
calctm.c:
calctm.c:         tmsol.min=gtmmin;
calctm.c:         tmsol.max=gtmmax;
calctm.c:         tmsol.mean=(gtmmax+gtmmin)/2;
calctm.c:
calctm.c: return tmsol;
calctm.c: }
```

```
calctree.c: #include <math.h>
calctree.c: #include <stdio.h>
calctree.c: #include <string.h>
calctree.c: /*
calctree.c: #include "search.h"
calctree.c: #include "functions.h"
calctree.c: #include "string.h"
calctree.c:
calctree.c: */
```

```
calctree.c:
calctree.c: #include "global.h"
calctree.c: #include "constantes.h"
calctree.c:
calctree.c: void calctree(float gtmmin, float gtmmax, FILE *fpinp, float w, float limit, char fileout[MAXLINE])
calctree.c: {
calctree.c:         extern float negtol;
calctree.c:         extern float postol;
calctree.c:         extern float rexmaxg;
calctree.c:         extern float fgridsize_s2;
calctree.c:         extern float fgridsize_ss2;
calctree.c:         extern float fgridsize_sf2;
calctree.c:         extern float fgridsize_te;
calctree.c:         extern float fgridsize_rex;
calctree.c:
calctree.c:
calctree.c: struct para residue[MAXRES];
calctree.c: struct para parateor;
calctree.c: struct para residue_unc[MAXRES];
calctree.c: struct solution solucion, res_sol[MAXRES][6];
calctree.c: float tm;
calctree.c: FILE *fpout1;
calctree.c: FILE *nosolution;
calctree.c:
calctree.c: char filename2[MAXLINE];
calctree.c: char line[MAXLINE];
calctree.c: char resname[MAXRES][10];
calctree.c: char filenosol[MAXLINE];
calctree.c: int mod, first;
calctree.c: float percentage,tmtotal;
calctree.c: float gridtm = TMGRID;
calctree.c: int nres;
calctree.c: int i=1;
calctree.c:
calctree.c:         sprintf(filenosol,"nosolution_%4.2f",limit);
calctree.c:         nosolution=fopen(filenosol,"w");
calctree.c:         while(getline(fpinp,line,sizeof(line)) > 0)
calctree.c: {
calctree.c: sscanf(line, "%s %f %f %f %f %f %f \n", &resname[i], &residue[i].R1, &residue_unc[i].R1, &residue[i].R2, &r
esidue_unc[i].R2, &residue[i].HNOE, &residue_unc[i].HNOE);
calctree.c:
calctree.c:               if(residue_unc[i].R1/residue[i].R1 < limit)
calctree.c:                   residue_unc[i].R1=absmio(residue[i].R1*limit);
calctree.c:               if(residue_unc[i].R2/residue[i].R2 < limit)
calctree.c:                   residue_unc[i].R2=absmio(residue[i].R2*limit);
calctree.c:               if(residue_unc[i].HNOE/residue[i].HNOE < limit)
calctree.c:                   residue_unc[i].HNOE=absmio(residue[i].HNOE*limit);
calctree.c:       i++;
calctree.c:
calctree.c:        }
calctree.c:         nres=i-1;
calctree.c:
calctree.c: for(i=1;i<=nres;i++)
calctree.c:   {
calctree.c:    fprintf(stdout,"SCH | %s > Residuo %d  %s\n",resname[i],i, resname[i]);
calctree.c:    for(mod=1;mod<=5;mod++)
calctree.c:     {
calctree.c:      fprintf(stdout,"SCH | %s > Modelo %d\n",resname[i],mod);
calctree.c:      percentage=0.0;
calctree.c:      tmtotal=0.0;
calctree.c:      first=1;
calctree.c:      sprintf(filename2,"%s_%s_mod%d",fileout,resname[i],mod);
calctree.c:      fpout1 = fopen(filename2, "w");
calctree.c:      for(tm=gtmmin;tm <=gtmmax;tm=tm+gridtm)
calctree.c:       {
calctree.c:       fprintf(stdout,"SCH | %s > Tm %6.3f\n",resname[i],tm);
calctree.c:       solucion=eval_modelf(mod,residue[i],residue_unc[i],tm,fpout1,w);
calctree.c:
calctree.c:       if((first == 1)&&(solucion.percent != 0.0))
calctree.c:        {
calctree.c:            first=0;
calctree.c:   res_sol[i][mod]=solucion; }
calctree.c:               else
calctree.c:   {
calctree.c:   if(solucion.percent != 0.0)
calctree.c:   {
```

```
calctree.c:    if( solucion.sols2.max > res_sol[i][mod].sols2.max )
calctree.c:       res_sol[i][mod].sols2.max=solucion.sols2.max;
calctree.c:    if( solucion.sols2.min < res_sol[i][mod].sols2.min )
calctree.c:       res_sol[i][mod].sols2.min=solucion.sols2.min;
calctree.c:    if( solucion.solss2.max > res_sol[i][mod].solss2.max )
calctree.c:       res_sol[i][mod].solss2.max=solucion.solss2.max;
calctree.c:    if( solucion.solss2.min < res_sol[i][mod].solss2.min )
calctree.c:       res_sol[i][mod].solss2.min=solucion.solss2.min;
calctree.c:    if( solucion.solsf2.max > res_sol[i][mod].solsf2.max )
calctree.c:       res_sol[i][mod].solsf2.max=solucion.solsf2.max;
calctree.c:    if( solucion.solsf2.min < res_sol[i][mod].solsf2.min )
calctree.c:       res_sol[i][mod].solsf2.min=solucion.solsf2.min;
calctree.c:    if( solucion.solte.max > res_sol[i][mod].solte.max )
calctree.c:       res_sol[i][mod].solte.max=solucion.solte.max;
calctree.c:    if( solucion.solte.min < res_sol[i][mod].solte.min )
calctree.c:       res_sol[i][mod].solte.min=solucion.solte.min;
calctree.c:    if( solucion.solrex.max > res_sol[i][mod].solrex.max )
calctree.c:       res_sol[i][mod].solrex.max=solucion.solrex.max;
calctree.c:    if( solucion.solrex.min < res_sol[i][mod].solrex.min )
calctree.c:       res_sol[i][mod].solrex.min=solucion.solrex.min;
calctree.c:       }
calctree.c:                   }
calctree.c:      percentage=percentage+solucion.percent*tm/TMGRID;
calctree.c:      tmtotal=tmtotal+tm;
calctree.c:      }
calctree.c:      fclose(fpout1);
calctree.c: /*             fprintf(stdout,"SCH | %s > Aqui llego V\n"); */
calctree.c:
calctree.c:
calctree.c:
calctree.c:      if(( res_sol[i][mod].solte.max > 0.0 )&&(res_sol[i][mod].solte.min < 0.0))
calctree.c:  res_sol[i][mod].solte.min=0.0;
calctree.c:
calctree.c:      res_sol[i][mod].percent=percentage/(tmtotal*TMGRID);
calctree.c:               res_sol[i][mod].sols2.mean=(res_sol[i][mod].sols2.max+res_sol[i][mod].sols2.min)/2;
calctree.c:               res_sol[i][mod].solss2.mean=(res_sol[i][mod].solss2.max+res_sol[i][mod].solss2.min)/2;
calctree.c:               res_sol[i][mod].solsf2.mean=(res_sol[i][mod].solsf2.max+res_sol[i][mod].solsf2.min)/2;
calctree.c:               res_sol[i][mod].solte.mean=(res_sol[i][mod].solte.max+res_sol[i][mod].solte.min)/2;
calctree.c:               res_sol[i][mod].solrex.mean=(res_sol[i][mod].solrex.max+res_sol[i][mod].solrex.min)/2;
calctree.c:               res_sol[i][mod].test=0;
calctree.c:
calctree.c:                  for(tm=gtmmin;tm <=gtmmax;tm=tm+TMGRID)
calctree.c:                  {
calctree.c:                  switch(mod)
calctree.c:                  {
calctree.c:                     case 1:
calctree.c:                            parateor=model1(res_sol[i][mod].sols2.mean, tm, w);
calctree.c:                            break;
calctree.c:                     case 2:
calctree.c:                            parateor=model2(res_sol[i][mod].solte.mean, res_sol[i][mod].sols2.mean, tm, w);
calctree.c:                            break;
calctree.c:                     case 3:
calctree.c:                            parateor=model3(res_sol[i][mod].solrex.mean, res_sol[i][mod].sols2.mean, tm, w);
calctree.c:                            break;
calctree.c:                     case 4:
calctree.c:                            parateor=model4(res_sol[i][mod].solte.mean, res_sol[i][mod].sols2.mean, res_sol[i]
[mod].solrex.mean, tm, w);
calctree.c:                            break;
calctree.c:                     case 5:
calctree.c:                            parateor=model5(res_sol[i][mod].solte.mean, res_sol[i][mod].solss2.mean, res_sol[i
][mod].solsf2.mean, tm, w);
calctree.c:                            break;
calctree.c:                     case 6:
calctree.c:                            parateor=model6(res_sol[i][mod].solte.mean, res_sol[i][mod].solss2.mean, res_sol[i
][mod].solsf2.mean,res_sol[i][mod].solrex.mean,  tm,
calctree.c: w);
calctree.c:                            break;
calctree.c:                            }
calctree.c:                   if(test_para(residue[i], residue_unc[i], parateor)==1)
calctree.c:                   {
calctree.c:                    res_sol[i][mod].test=1;
calctree.c:                    break;
calctree.c:                    }
calctree.c:                   }
calctree.c:    fprintf(stdout,"SCHSOL> Residuo %d Modelo %d\n",i,mod);
calctree.c:    fprintf(stdout,"SCHSOL> ============================\n");
```

```
calctree.c:    fprintf(stdout,"SCHSOL> \n");
calctree.c:    if(percentage != 0.0)
calctree.c:            switch(mod)
calctree.c:            {
calctree.c:               case 1:
calctree.c:    fprintf(stdout,"SCHSOL | %s > s2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].sols2.mean,res_sol[i][
mod].sols2.min,res_sol[i][mod].sols2.max);
calctree.c:    fprintf(stdout,"SCHSOL> \n");
calctree.c:            break;
calctree.c:               case 2:
calctree.c:    fprintf(stdout,"SCHSOL | %s > s2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].sols2.mean,res_sol[i][
mod].sols2.min,res_sol[i][mod].sols2.max);
calctree.c:    fprintf(stdout,"SCHSOL | %s > te %6.3f %6.3f %6.3f ",resname[i],res_sol[i][mod].solte.mean,res_sol[i][mo
d].solte.min,res_sol[i][mod].solte.max);
calctree.c:    if(res_sol[i][mod].solte.max < 0.0)
calctree.c:       fprintf(stdout,"  *******\n");
calctree.c:            else
calctree.c:       fprintf(stdout,"\n");
calctree.c:    fprintf(stdout,"SCHSOL> \n");
calctree.c:            break;
calctree.c:       case 3:
calctree.c:    fprintf(stdout,"SCHSOL | %s > s2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].sols2.mean,res_sol[i][
mod].sols2.min,res_sol[i][mod].sols2.max);
calctree.c:    fprintf(stdout,"SCHSOL | %s > rex %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solrex.mean,res_sol[i
][mod].solrex.min,res_sol[i][mod].solrex.max);
calctree.c:    fprintf(stdout,"SCHSOL> \n");
calctree.c:    break;
calctree.c:    case 4:
calctree.c:    fprintf(stdout,"SCHSOL | %s > s2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].sols2.mean,res_sol[i][
mod].sols2.min,res_sol[i][mod].sols2.max);
calctree.c:    fprintf(stdout,"SCHSOL | %s > te %6.3f %6.3f %6.3f ",resname[i],res_sol[i][mod].solte.mean,res_sol[i][mo
d].solte.min,res_sol[i][mod].solte.max);
calctree.c:    if(res_sol[i][mod].solte.max < 0.0)
calctree.c:       fprintf(stdout,"  *******\n");
calctree.c:            else
calctree.c:       fprintf(stdout,"\n");
calctree.c:    fprintf(stdout,"SCHSOL | %s > rex %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solrex.mean,res_sol[i
][mod].solrex.min,res_sol[i][mod].solrex.max);
calctree.c:    fprintf(stdout,"SCHSOL> \n");
calctree.c:    break;
calctree.c:
calctree.c:    case 5:
calctree.c:    fprintf(stdout,"SCHSOL | %s > ss2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solss2.mean,res_sol[i
][mod].solss2.min,res_sol[i][mod].solss2.max);
calctree.c:    fprintf(stdout,"SCHSOL | %s > sf2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solsf2.mean,res_sol[i
][mod].solsf2.min,res_sol[i][mod].solsf2.max);
calctree.c:    fprintf(stdout,"SCHSOL | %s > te %6.3f %6.3f %6.3f ",resname[i],res_sol[i][mod].solte.mean,res_sol[i][mo
d].solte.min,res_sol[i][mod].solte.max);
calctree.c:    if(res_sol[i][mod].solte.max < 0.0)
calctree.c:       fprintf(stdout,"  *******\n");
calctree.c:            else
calctree.c:       fprintf(stdout,"\n");
calctree.c:    fprintf(stdout,"SCHSOL> \n");
calctree.c:    break;
calctree.c:
calctree.c:    case 6:
calctree.c:    fprintf(stdout,"SCHSOL | %s > ss2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solss2.mean,res_sol[i
][mod].solss2.min,res_sol[i][mod].solss2.max);
calctree.c:    fprintf(stdout,"SCHSOL | %s > sf2 %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solsf2.mean,res_sol[i
][mod].solsf2.min,res_sol[i][mod].solsf2.max);
calctree.c:    fprintf(stdout,"SCHSOL | %s > te %6.3f %6.3f %6.3f ",resname[i],res_sol[i][mod].solte.mean,res_sol[i][mo
d].solte.min,res_sol[i][mod].solte.max);
calctree.c:    if(res_sol[i][mod].solte.max < 0.0)
calctree.c:       fprintf(stdout,"  *******\n");
calctree.c:            else
calctree.c:       fprintf(stdout,"\n");
calctree.c:    fprintf(stdout,"SCHSOL | %s > rex %6.3f %6.3f %6.3f \n",resname[i],res_sol[i][mod].solrex.mean,res_sol[i
][mod].solrex.min,res_sol[i][mod].solrex.max);
calctree.c:    fprintf(stdout,"SCHSOL> \n");
calctree.c:    break;
calctree.c: }
calctree.c:
calctree.c: /*    fprintf(stdout,"SCHSOL | %s > Percent %6.3f \n",resname[i],res_sol[i][mod].percent);
calctree.c: */
calctree.c:    fprintf(stdout,"SCHSOL | %s > Percent %6.3f \n",resname[i],percentage/tmtotal);
calctree.c:
```

```
calctree.c:    fprintf(stdout,"SCHSOL | %s > Test %d \n",resname[i],res_sol[i][mod].test);
calctree.c:    fprintf(stdout,"SCHSOL> \n");
calctree.c:    if(percentage != 0.0)
calctree.c:       mod=10;
calctree.c:     }
calctree.c:         if((mod != 10)&&(percentage == 0.0))
calctree.c: {  fprintf(stdout,"SCH | %s > No solution...  \n",resname[i]);
calctree.c:            fprintf(nosolution,"%s %6.3f %6.3f %6.3f %6.3f %6.3f %6.3f \n",resname[i], residue[i].R1, residue
_unc[i].R1, residue[i].R2, residue_unc[i].R2, residue[i].HNOE, residue_unc[i].HNOE);
calctree.c:    fflush(nosolution);
calctree.c:    fflush(stdout);
calctree.c:    fflush(stderr);
calctree.c:    }
calctree.c:
calctree.c:    }
calctree.c:         fclose(nosolution);
calctree.c: return;
calctree.c: }
```

```
checkmodel.c: #include <math.h>
checkmodel.c: #include <stdio.h>
checkmodel.c: /* #include "search.h"
checkmodel.c: #include "functions.h" */
checkmodel.c: #include "global.h"
checkmodel.c: #include "constantes.h"
checkmodel.c: #define GRIDSIZE 0.01
checkmodel.c:
checkmodel.c:
checkmodel.c:
checkmodel.c: int check_model(int typemodel, struct para paraexp, struct para uncert, float tm , float w)
checkmodel.c: {
checkmodel.c:      struct para parateor;
checkmodel.c:
checkmodel.c:      float gridsize_s2, gridsize_te, gridsize_sf2, gridsize_ss2, gridsize_rex;
checkmodel.c:      float fgridsize_s2, fgridsize_te, fgridsize_sf2, fgridsize_ss2, fgridsize_rex;
checkmodel.c:      float ngridsize_s2, ngridsize_te, ngridsize_sf2, ngridsize_ss2, ngridsize_rex;
checkmodel.c:      float s2, s2min, s2max;
checkmodel.c:      float ss2, ss2min, ss2max;
checkmodel.c:      float sf2, sf2min, sf2max;
checkmodel.c:      float te, temin, temax;
checkmodel.c:      float rex, rexmin, rexmax;
checkmodel.c:      float ns2min, ns2max;
checkmodel.c:      float nss2min, nss2max;
checkmodel.c:      float nsf2min, nsf2max;
checkmodel.c:      float ntemin, ntemax;
checkmodel.c:      float nrexmin, nrexmax;
checkmodel.c:
checkmodel.c:      int evaluation, found;
checkmodel.c:      int red;
checkmodel.c:
checkmodel.c:      evaluation=0;
checkmodel.c:
checkmodel.c:      fgridsize_s2=GRIDSIZE;
checkmodel.c:      fgridsize_te=GRIDSIZE/10;
checkmodel.c:      fgridsize_sf2=GRIDSIZE;
checkmodel.c:      fgridsize_ss2=GRIDSIZE;
checkmodel.c:      fgridsize_rex=GRIDSIZE*10;
checkmodel.c:
checkmodel.c:      found=0;
checkmodel.c:              s2min=0.0;
checkmodel.c:      s2max=1.0;
checkmodel.c:              sf2min=0.0;
checkmodel.c:      sf2max=1.0;
checkmodel.c:              ss2min=0.0;
checkmodel.c:      ss2max=1.0;
checkmodel.c:      temin=0.0;
checkmodel.c:      temax=tm;
checkmodel.c:      rexmin=0.0;
checkmodel.c:      rexmax=REXMAX;
checkmodel.c:              ns2min=0.0;
checkmodel.c:      ns2max=1.0;
checkmodel.c:              nsf2min=0.0;
checkmodel.c:      nsf2max=1.0;
```

```
checkmodel.c:              nss2min=0.0;
checkmodel.c:      nss2max=1.0;
checkmodel.c:      ntemin=0.0;
checkmodel.c:      ntemax=tm;
checkmodel.c:      nrexmin=0.0;
checkmodel.c:      nrexmax=REXMAX;
checkmodel.c:
checkmodel.c:
checkmodel.c:      switch (typemodel)
checkmodel.c:        {
checkmodel.c:          case 2:
checkmodel.c:              s2min=0.0;
checkmodel.c:      s2max=1.0;
checkmodel.c:      temin=0.0;
checkmodel.c:      temax=tm;
checkmodel.c:
checkmodel.c:              ns2min=0.0;
checkmodel.c:      ns2max=1.0;
checkmodel.c:      ntemin=0.0;
checkmodel.c:      ntemax=tm;
checkmodel.c:      red=0;
checkmodel.c:
checkmodel.c:      gridsize_s2 = 0.05;
checkmodel.c:              gridsize_te = tm /20.0 ;
checkmodel.c:      while ((gridsize_s2 != fgridsize_s2) && (gridsize_te != fgridsize_te))
checkmodel.c: {
checkmodel.c:
checkmodel.c: /* fprintf(stdout,".");  */
checkmodel.c:    red++;
checkmodel.c:
checkmodel.c: /*
checkmodel.c:                    ns2min=(s2min+s2max)/2.0;
checkmodel.c:            ns2max=(s2min+s2max)/2.0;
checkmodel.c:            ntemin=(temin+temax)/2.0;
checkmodel.c:            ntemax=(temin+temax)/2.0;
checkmodel.c: */
checkmodel.c:                    found = 0;
checkmodel.c:    for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
checkmodel.c:     {
checkmodel.c:      for(te = temin;te <= temax; te = te + gridsize_te)
checkmodel.c:        {
checkmodel.c:   parateor=model2(te, s2, tm, w);
checkmodel.c:
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:     found=1;
checkmodel.c:     return 1;
checkmodel.c:    }
checkmodel.c:                          if(found==1)
checkmodel.c:     break;
checkmodel.c:     }
checkmodel.c:  }
checkmodel.c: /*
checkmodel.c:                  if(ns2min!=(s2min+s2max)/2.0)
checkmodel.c:     s2min=ns2min-gridsize_s2;
checkmodel.c:                  if(ns2max!=(s2min+s2max)/2.0)
checkmodel.c:     s2max=ns2max+gridsize_s2;
checkmodel.c:                  if(ntemin!=(temin+temax)/2.0)
checkmodel.c:     temin=ntemin-gridsize_te;
checkmodel.c:                  if(ntemax!=(temin+temax)/2.0)
checkmodel.c:     temax=ntemax+gridsize_te;
checkmodel.c: */
checkmodel.c: /*
checkmodel.c:     s2min=ns2min-gridsize_s2;
checkmodel.c:     s2max=ns2max+gridsize_s2;
checkmodel.c:     temin=ntemin-gridsize_te;
checkmodel.c:     temax=ntemax+gridsize_te;
checkmodel.c:     */
checkmodel.c:
checkmodel.c:                  ngridsize_s2=gridsize_s2/10.0;
checkmodel.c:  ngridsize_te=gridsize_te/10.0;
checkmodel.c:
checkmodel.c:
checkmodel.c:  if((ngridsize_s2/gridsize_s2 > 0.5) ||
checkmodel.c:                     (fgridsize_s2/ngridsize_s2 < 0.5) || (ngridsize_s2 < fgridsize_s2 ))
checkmodel.c:  {
```

```
checkmodel.c:     gridsize_s2 = fgridsize_s2;
checkmodel.c:                  }
checkmodel.c:  else
checkmodel.c:  {
checkmodel.c:     gridsize_s2 = ngridsize_s2;
checkmodel.c:  }
checkmodel.c:
checkmodel.c:  if((ngridsize_te/gridsize_te > 0.5) ||
checkmodel.c:                      (fgridsize_te/ngridsize_te < 0.5) || (ngridsize_te < fgridsize_te ))
checkmodel.c:  {
checkmodel.c:     gridsize_te = fgridsize_te;
checkmodel.c:                  }
checkmodel.c:  else
checkmodel.c:  {
checkmodel.c:     gridsize_te = ngridsize_te;
checkmodel.c:  }
checkmodel.c:
checkmodel.c:
checkmodel.c:         }
checkmodel.c: /*  fprintf(stdout,"%d ", red);  */
checkmodel.c:                evaluation=found;
checkmodel.c:
checkmodel.c: if(evaluation==0)
checkmodel.c: {
checkmodel.c:  for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
checkmodel.c:  {
checkmodel.c:    for(te = temin;te <= temax; te = te + gridsize_te)
checkmodel.c:        {
checkmodel.c:
checkmodel.c:   parateor=model2(te, s2, tm, w);
checkmodel.c:
checkmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:
checkmodel.c:     evaluation=1;
checkmodel.c:     return 1;
checkmodel.c:                           }
checkmodel.c:                          }
checkmodel.c: if(evaluation==1)
checkmodel.c: return 1;
checkmodel.c: }
checkmodel.c: }
checkmodel.c:
checkmodel.c: break;
checkmodel.c:
checkmodel.c:          case 3:
checkmodel.c:              s2min=0.0;
checkmodel.c:      s2max=1.0;
checkmodel.c:      rexmin=0.0;
checkmodel.c:      rexmax=REXMAX;
checkmodel.c:      gridsize_s2 = 0.1;
checkmodel.c:              gridsize_rex = (rexmax-rexmin) /10.0 ;
checkmodel.c:      while ((gridsize_s2 != fgridsize_s2) && (gridsize_rex != fgridsize_rex))
checkmodel.c: {
checkmodel.c:    for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
checkmodel.c:      for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
checkmodel.c:        {
checkmodel.c:   parateor=model3(rex, s2, tm, w);
checkmodel.c:
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:   if((s2 < ns2min)&&(s2 > s2min))
checkmodel.c:        ns2min=s2;
checkmodel.c:   else if((s2 > ns2max)&&(s2 < s2max))
checkmodel.c:        ns2max=s2;
checkmodel.c:   if((rex < nrexmin)&&(rex > rexmin))
checkmodel.c:        nrexmin=rex;
checkmodel.c:   else if((rex > nrexmax)&&(rex < rexmin))
checkmodel.c:        nrexmax=rex;
checkmodel.c:    }
checkmodel.c: /*
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:   if(s2 < ns2min)
checkmodel.c:        ns2min=s2;
checkmodel.c:   else if(s2 > ns2max)
```

```
checkmodel.c:        ns2max=s2;
checkmodel.c:   if(rex < nrexmin)
checkmodel.c:        nrexmin=rex;
checkmodel.c:   else if(rex > nrexmax)
checkmodel.c:        nrexmax=rex;
checkmodel.c:    }
checkmodel.c: */
checkmodel.c: }
checkmodel.c:
checkmodel.c: /*
checkmodel.c:                  ngridsize_s2=(ns2max-ns2min)/10.0;
checkmodel.c:  ngridsize_rex=(nrexmax-nrexmin)/10.0;
checkmodel.c:
checkmodel.c:  */
checkmodel.c:
checkmodel.c:                  ngridsize_s2=gridsize_s2/10.0;
checkmodel.c:  ngridsize_rex=gridsize_rex/10.0;
checkmodel.c:
checkmodel.c:  if(((ngridsize_s2/gridsize_s2 > 0.5)&&(ngridsize_rex/gridsize_rex > 0.5)) ||
checkmodel.c:     ((fgridsize_s2/ngridsize_s2 > 0.5)&&(fgridsize_rex/ngridsize_rex > 0.5)))
checkmodel.c:  {
checkmodel.c:     gridsize_s2 = fgridsize_s2;
checkmodel.c:     gridsize_rex = fgridsize_rex;
checkmodel.c:                  }
checkmodel.c:  else
checkmodel.c:  {
checkmodel.c:     gridsize_s2 = ngridsize_s2;
checkmodel.c:     gridsize_rex = ngridsize_rex;
checkmodel.c:  }
checkmodel.c:
checkmodel.c:
checkmodel.c:
checkmodel.c: }
checkmodel.c:
checkmodel.c:  for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
checkmodel.c:    for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
checkmodel.c:        {
checkmodel.c:   parateor=model3(rex, s2, tm, w);
checkmodel.c:
checkmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:     evaluation=1;
checkmodel.c:                           }
checkmodel.c:                         }
checkmodel.c:
checkmodel.c: break;
checkmodel.c:
checkmodel.c:         case 4:
checkmodel.c:              s2min=0.0;
checkmodel.c:      s2max=1.0;
checkmodel.c:      temin=0.0;
checkmodel.c:      temax=tm;
checkmodel.c:      rexmin=0.0;
checkmodel.c:      rexmax=REXMAX;
checkmodel.c:              gridsize_rex = (rexmax-rexmin) /10.0 ;
checkmodel.c:      gridsize_s2 = 0.1;
checkmodel.c:              gridsize_te = tm /10.0 ;
checkmodel.c:      while ((gridsize_s2 != fgridsize_s2) && (gridsize_te != fgridsize_te) &&
checkmodel.c:     (gridsize_rex != fgridsize_rex))
checkmodel.c: {
checkmodel.c:    for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
checkmodel.c:      for(te = temin;te <= temax; te = te + gridsize_te)
checkmodel.c:        for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
checkmodel.c:        {
checkmodel.c:   parateor=model4(te, s2, rex, tm, w);
checkmodel.c:
checkmodel.c:
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:   if((s2 < ns2min)&&(s2 > s2min))
checkmodel.c:        ns2min=s2;
checkmodel.c:   else if((s2 > ns2max)&&(s2 < s2max))
checkmodel.c:        ns2max=s2;
checkmodel.c:   if((te < ntemin)&&(te > temin))
checkmodel.c:        ntemin=te;
checkmodel.c:   else if((te > ntemax)&&(te < temin))
```

```
checkmodel.c:        ntemax=te;
checkmodel.c:   if((rex < nrexmin)&&(rex > rexmin))
checkmodel.c:        nrexmin=rex;
checkmodel.c:   else if((rex > nrexmax)&&(rex < rexmin))
checkmodel.c:        nrexmax=rex;
checkmodel.c:    }
checkmodel.c: /*
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:   if(s2 < ns2min)
checkmodel.c:        ns2min=s2;
checkmodel.c:   else if(s2 > ns2max)
checkmodel.c:        ns2max=s2;
checkmodel.c:   if(te < ntemin)
checkmodel.c:        ntemin=te;
checkmodel.c:   else if(te > ntemax)
checkmodel.c:        ntemax=te;
checkmodel.c:   if(rex < nrexmin)
checkmodel.c:        nrexmin=rex;
checkmodel.c:   else if(rex > nrexmax)
checkmodel.c:        nrexmax=rex;
checkmodel.c:    }
checkmodel.c: */
checkmodel.c: }
checkmodel.c:
checkmodel.c: /*
checkmodel.c:                  ngridsize_s2=(ns2max-ns2min)/10.0;
checkmodel.c:  ngridsize_te=(ntemax-ntemin)/10.0;
checkmodel.c:  ngridsize_rex=(nrexmax-nrexmin)/10.0;
checkmodel.c:  */
checkmodel.c:                  ngridsize_s2=gridsize_s2/10.0;
checkmodel.c:  ngridsize_te=gridsize_te/10.0;
checkmodel.c:  ngridsize_rex=gridsize_rex/10.0;
checkmodel.c:
checkmodel.c:  if(((ngridsize_s2/gridsize_s2 > 0.5)&&(ngridsize_te/gridsize_te > 0.5) &&
checkmodel.c:      (ngridsize_rex/gridsize_rex > 0.5)) ||
checkmodel.c:     ((fgridsize_s2/ngridsize_s2 > 0.5)&&(fgridsize_te/ngridsize_te > 0.5) &&
checkmodel.c:      (fgridsize_rex/ngridsize_rex > 0.5) ))
checkmodel.c:  {
checkmodel.c:     gridsize_s2 = fgridsize_s2;
checkmodel.c:     gridsize_te = fgridsize_te;
checkmodel.c:     gridsize_rex = fgridsize_rex;
checkmodel.c:                  }
checkmodel.c:  else
checkmodel.c:  {
checkmodel.c:     gridsize_s2 = ngridsize_s2;
checkmodel.c:     gridsize_te = ngridsize_te;
checkmodel.c:     gridsize_rex = ngridsize_rex;
checkmodel.c:  }
checkmodel.c:
checkmodel.c:
checkmodel.c: }
checkmodel.c:  for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
checkmodel.c:    for(te = temin;te <= temax; te = te + gridsize_te)
checkmodel.c:        for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
checkmodel.c:        {
checkmodel.c:   parateor=model4(te, s2, rex, tm, w);
checkmodel.c:
checkmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:     evaluation=1;
checkmodel.c:                             }
checkmodel.c:                          }
checkmodel.c:
checkmodel.c: break;
checkmodel.c:
checkmodel.c:         case 5:
checkmodel.c:              sf2min=0.0;
checkmodel.c:      sf2max=1.0;
checkmodel.c:              ss2min=0.0;
checkmodel.c:      ss2max=1.0;
checkmodel.c:      temin=0.0;
checkmodel.c:      temax=tm;
checkmodel.c:      gridsize_sf2 = 0.1;
checkmodel.c:      gridsize_ss2 = 0.1;
checkmodel.c:              gridsize_te = tm /10.0 ;
```

```
checkmodel.c:      while ((gridsize_ss2 != fgridsize_ss2) && (gridsize_te != fgridsize_te)
checkmodel.c:                   && (gridsize_sf2 != fgridsize_sf2) )
checkmodel.c: {
checkmodel.c:    for(ss2 = ss2min;ss2 <= ss2max; ss2 = ss2 + gridsize_ss2)
checkmodel.c:     for(sf2 = sf2min;sf2 <= sf2max; sf2 = sf2 + gridsize_sf2)
checkmodel.c:      for(te = temin;te <= temax; te = te + gridsize_te)
checkmodel.c:        {
checkmodel.c:   parateor=model5(te, ss2, sf2, tm, w);
checkmodel.c:
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:   if((sf2 < nsf2min)&&(sf2 > sf2min))
checkmodel.c:        nsf2min=sf2;
checkmodel.c:   else if((sf2 > nsf2max)&&(sf2 < sf2max))
checkmodel.c:        nsf2max=sf2;
checkmodel.c:   if((ss2 < nss2min)&&(ss2 > ss2min))
checkmodel.c:        nss2min=ss2;
checkmodel.c:   else if((ss2 > nss2max)&&(ss2 < ss2max))
checkmodel.c:        nss2max=ss2;
checkmodel.c:   if((te < ntemin)&&(te > temin))
checkmodel.c:        ntemin=te;
checkmodel.c:   else if((te > ntemax)&&(te < temin))
checkmodel.c:        ntemax=te;
checkmodel.c:    }
checkmodel.c: /*
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:   if(ss2 < nss2min)
checkmodel.c:        nss2min=ss2;
checkmodel.c:   else if(ss2 > nss2max)
checkmodel.c:        nss2max=s2;
checkmodel.c:   if(sf2 < nsf2min)
checkmodel.c:        nsf2min=sf2;
checkmodel.c:   else if(ss2 > nss2max)
checkmodel.c:        nss2max=s2;
checkmodel.c:   if(te < ntemin)
checkmodel.c:        ntemin=te;
checkmodel.c:   else if(te > ntemax)
checkmodel.c:        ntemax=te;
checkmodel.c:    }
checkmodel.c: */
checkmodel.c: }
checkmodel.c: /*
checkmodel.c:                   ngridsize_sf2=(nsf2max-nsf2min)/10.0;
checkmodel.c:                   ngridsize_ss2=(nss2max-nss2min)/10.0;
checkmodel.c:  ngridsize_te=(ntemax-ntemin)/10.0;
checkmodel.c:  */
checkmodel.c:                   ngridsize_sf2=gridsize_sf2/10.0;
checkmodel.c:                   ngridsize_ss2=gridsize_ss2/10.0;
checkmodel.c:  ngridsize_te=gridsize_te/10.0;
checkmodel.c:
checkmodel.c:  if(((ngridsize_sf2/gridsize_sf2 > 0.5)&&(ngridsize_te/gridsize_te > 0.5)
checkmodel.c:  && (ngridsize_ss2/gridsize_ss2 > 0.5)) ||
checkmodel.c:     ((fgridsize_sf2/ngridsize_sf2 > 0.5)&&(fgridsize_te/ngridsize_te > 0.5)
checkmodel.c:  && (fgridsize_ss2/gridsize_ss2 > 0.5)))
checkmodel.c:  {
checkmodel.c:     gridsize_ss2 = fgridsize_ss2;
checkmodel.c:     gridsize_sf2 = fgridsize_sf2;
checkmodel.c:     gridsize_te = fgridsize_te;
checkmodel.c:                   }
checkmodel.c:  else
checkmodel.c:  {
checkmodel.c:     gridsize_sf2 = ngridsize_sf2;
checkmodel.c:     gridsize_ss2 = ngridsize_ss2;
checkmodel.c:     gridsize_te = ngridsize_te;
checkmodel.c:  }
checkmodel.c:
checkmodel.c:
checkmodel.c:
checkmodel.c:  for(ss2 = ss2min;ss2 <= ss2max; ss2 = ss2 + gridsize_ss2)
checkmodel.c:   for(sf2 = sf2min;sf2 <= sf2max; sf2 = sf2 + gridsize_sf2)
checkmodel.c:    for(te = temin;te <= temax; te = te + gridsize_te)
checkmodel.c:        {
checkmodel.c:   parateor=model5(te, ss2, sf2, tm, w);
checkmodel.c:
checkmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
```

```
checkmodel.c:   {
checkmodel.c:     evaluation=1;
checkmodel.c:                             }
checkmodel.c:                           }
checkmodel.c: }
checkmodel.c: break;
checkmodel.c:
checkmodel.c:          case 6:
checkmodel.c:
checkmodel.c:
checkmodel.c:               sf2min=0.0;
checkmodel.c:      sf2max=1.0;
checkmodel.c:               ss2min=0.0;
checkmodel.c:      ss2max=1.0;
checkmodel.c:      temin=0.0;
checkmodel.c:      temax=tm;
checkmodel.c:      rexmin=0.0;
checkmodel.c:      rexmax=REXMAX;
checkmodel.c:      gridsize_sf2 = 0.1;
checkmodel.c:      gridsize_ss2 = 0.1;
checkmodel.c:               gridsize_te = tm /10.0 ;
checkmodel.c:      gridsize_rex = (rexmax-rexmin) /10.0 ;
checkmodel.c:      while ((gridsize_ss2 != fgridsize_ss2) && (gridsize_te != fgridsize_te)
checkmodel.c:                   && (gridsize_sf2 != fgridsize_sf2) && (gridsize_rex != fgridsize_rex) )
checkmodel.c: {
checkmodel.c:    for(ss2 = ss2min;ss2 <= ss2max; ss2 = ss2 + gridsize_ss2)
checkmodel.c:     for(sf2 = sf2min;sf2 <= sf2max; sf2 = sf2 + gridsize_sf2)
checkmodel.c:      for(te = temin;te <= temax; te = te + gridsize_te)
checkmodel.c:        for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
checkmodel.c:        {
checkmodel.c:   parateor=model6(te, ss2, sf2,rex,  tm, w);
checkmodel.c:
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:   if((sf2 < nsf2min)&&(sf2 > sf2min))
checkmodel.c:        nsf2min=sf2;
checkmodel.c:   else if((sf2 > nsf2max)&&(sf2 < sf2max))
checkmodel.c:        nsf2max=sf2;
checkmodel.c:   if((ss2 < nss2min)&&(ss2 > ss2min))
checkmodel.c:        nss2min=ss2;
checkmodel.c:   else if((ss2 > nss2max)&&(ss2 < ss2max))
checkmodel.c:        nss2max=ss2;
checkmodel.c:   if((te < ntemin)&&(te > temin))
checkmodel.c:        ntemin=te;
checkmodel.c:   else if((te > ntemax)&&(te < temin))
checkmodel.c:        ntemax=te;
checkmodel.c:   if((rex < nrexmin)&&(rex > rexmin))
checkmodel.c:        nrexmin=rex;
checkmodel.c:   else if((rex > nrexmax)&&(rex < rexmin))
checkmodel.c:        nrexmax=rex;
checkmodel.c:    }
checkmodel.c: /*
checkmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:   if(ss2 < nss2min)
checkmodel.c:        nss2min=ss2;
checkmodel.c:   else if(ss2 > nss2max)
checkmodel.c:        nss2max=s2;
checkmodel.c:   if(sf2 < nsf2min)
checkmodel.c:        nsf2min=sf2;
checkmodel.c:   else if(ss2 > nss2max)
checkmodel.c:        nss2max=s2;
checkmodel.c:   if(te < ntemin)
checkmodel.c:        ntemin=te;
checkmodel.c:   else if(te > ntemax)
checkmodel.c:        ntemax=te;
checkmodel.c:   if(rex < nrexmin)
checkmodel.c:        nrexmin=rex;
checkmodel.c:   else if(rex > nrexmax)
checkmodel.c:        nrexmax=rex;
checkmodel.c:    }
checkmodel.c: */
checkmodel.c: }
checkmodel.c: /*
checkmodel.c:                   ngridsize_sf2=gridsize_sf2/10.0;
checkmodel.c:                   ngridsize_ss2=gridsize_ss2/10.0;
```

```
checkmodel.c:  ngridsize_te=gridsize_te/10.0;
checkmodel.c:  ngridsize_rex=gridsize_rex/10.0;
checkmodel.c:  */
checkmodel.c:  if(((ngridsize_sf2/gridsize_sf2 > 0.5)&&(ngridsize_te/gridsize_te > 0.5)
checkmodel.c:  && (ngridsize_ss2/gridsize_ss2 > 0.5) && (ngridsize_rex/gridsize_rex > 0.5)) ||
checkmodel.c:     ((fgridsize_sf2/ngridsize_sf2 > 0.5)&&(fgridsize_te/ngridsize_te > 0.5)
checkmodel.c:  && (fgridsize_ss2/gridsize_ss2 > 0.5) && (fgridsize_rex/ngridsize_rex > 0.5)))
checkmodel.c:  {
checkmodel.c:     gridsize_ss2 = fgridsize_ss2;
checkmodel.c:     gridsize_sf2 = fgridsize_sf2;
checkmodel.c:     gridsize_te = fgridsize_te;
checkmodel.c:     gridsize_rex = fgridsize_rex;
checkmodel.c:                  }
checkmodel.c:  else
checkmodel.c:  {
checkmodel.c:     gridsize_sf2 = ngridsize_sf2;
checkmodel.c:     gridsize_ss2 = ngridsize_ss2;
checkmodel.c:     gridsize_te = ngridsize_te;
checkmodel.c:     gridsize_rex = ngridsize_rex;
checkmodel.c:  }
checkmodel.c:
checkmodel.c:
checkmodel.c:  if(found == 1)
checkmodel.c:  {
checkmodel.c:  sf2max=nsf2max;
checkmodel.c:  sf2min=nsf2min;
checkmodel.c:  ss2max=nss2max;
checkmodel.c:  ss2min=nss2min;
checkmodel.c:  temax=ntemax;
checkmodel.c:  temin=ntemin;
checkmodel.c:  rexmax=nrexmax;
checkmodel.c:  rexmin=nrexmin;
checkmodel.c:  }
checkmodel.c:
checkmodel.c:
checkmodel.c:  for(ss2 = ss2min;ss2 <= ss2max; ss2 = ss2 + gridsize_ss2)
checkmodel.c:   for(sf2 = sf2min;sf2 <= sf2max; sf2 = sf2 + gridsize_sf2)
checkmodel.c:    for(te = temin;te <= temax; te = te + gridsize_te)
checkmodel.c:      for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
checkmodel.c:        {
checkmodel.c:   parateor=model6(te, ss2, sf2, rex, tm, w);
checkmodel.c:
checkmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
checkmodel.c:   {
checkmodel.c:     evaluation=1;
checkmodel.c:                           }
checkmodel.c:                         }
checkmodel.c: }
checkmodel.c: break;
checkmodel.c: }
checkmodel.c:
checkmodel.c:                 return evaluation;
checkmodel.c: }
```

```
evalmodel.c: #include <math.h>
evalmodel.c: #include <stdio.h>
evalmodel.c: #include "global.h"
evalmodel.c: #include "constantes.h"
evalmodel.c: #define GRIDSIZE 0.01
evalmodel.c:
evalmodel.c:
evalmodel.c:
evalmodel.c: struct solution eval_model(int typemodel, struct para paraexp, struct para uncert, float tm, float w)
evalmodel.c: {
evalmodel.c:      struct para parateor;
evalmodel.c:      struct solution solucion;
evalmodel.c:
evalmodel.c:      float gridsize_s2, gridsize_te, gridsize_sf2, gridsize_ss2, gridsize_rex;
evalmodel.c:      float fgridsize_s2, fgridsize_te, fgridsize_sf2, fgridsize_ss2, fgridsize_rex;
evalmodel.c:      float ngridsize_s2, ngridsize_te, ngridsize_sf2, ngridsize_ss2, ngridsize_rex;
evalmodel.c:      float s2, s2min, s2max;
evalmodel.c:      float ss2, ss2min, ss2max;
evalmodel.c:      float sf2, sf2min, sf2max;
```

```
evalmodel.c:      float te, temin, temax;
evalmodel.c:      float rex, rexmin, rexmax;
evalmodel.c:      float ns2min, ns2max;
evalmodel.c:      float nss2min, nss2max;
evalmodel.c:      float nsf2min, nsf2max;
evalmodel.c:      float ntemin, ntemax;
evalmodel.c:      float nrexmin, nrexmax;
evalmodel.c:
evalmodel.c:      int evaluation, found;
evalmodel.c:      int acertados, totales;
evalmodel.c:
evalmodel.c:      evaluation=0;
evalmodel.c:      acertados=0;
evalmodel.c:      totales=0;
evalmodel.c:      found=0;
evalmodel.c:
evalmodel.c:      fgridsize_s2=GRIDSIZE;
evalmodel.c:      fgridsize_te=GRIDSIZE;
evalmodel.c:      fgridsize_sf2=GRIDSIZE;
evalmodel.c:      fgridsize_ss2=GRIDSIZE;
evalmodel.c:      fgridsize_rex=GRIDSIZE*10;
evalmodel.c:
evalmodel.c:              s2min=0.0;
evalmodel.c:      s2max=1.0;
evalmodel.c:              sf2min=0.0;
evalmodel.c:      sf2max=1.0;
evalmodel.c:              ss2min=0.0;
evalmodel.c:      ss2max=1.0;
evalmodel.c:      temin=0.0;
evalmodel.c:      temax=tm;
evalmodel.c:      rexmin=0.0;
evalmodel.c:      rexmax=REXMAX;
evalmodel.c:              ns2min=0.0;
evalmodel.c:      ns2max=1.0;
evalmodel.c:              nsf2min=0.0;
evalmodel.c:      nsf2max=1.0;
evalmodel.c:              nss2min=0.0;
evalmodel.c:      nss2max=1.0;
evalmodel.c:      ntemin=0.0;
evalmodel.c:      ntemax=tm;
evalmodel.c:      nrexmin=0.0;
evalmodel.c:      nrexmax=REXMAX;
evalmodel.c:
evalmodel.c:
evalmodel.c:      switch (typemodel)
evalmodel.c:        {
evalmodel.c:          case 2:
evalmodel.c:
evalmodel.c:              s2min=0.0;
evalmodel.c:      s2max=1.0;
evalmodel.c:      temin=0.0;
evalmodel.c:      temax=tm;
evalmodel.c:
evalmodel.c:              ns2min=0.0;
evalmodel.c:      ns2max=1.0;
evalmodel.c:      ntemin=0.0;
evalmodel.c:      ntemax=tm;
evalmodel.c:
evalmodel.c: /*      gridsize_s2 = 0.05;
evalmodel.c:              gridsize_te = tm /20.0 ; */
evalmodel.c:      gridsize_s2 = (s2max - s2min)/10.0;
evalmodel.c:              gridsize_te = (temax - temin)/10.0;
evalmodel.c:
evalmodel.c:      while ((gridsize_s2 != fgridsize_s2) && (gridsize_te != fgridsize_te))
evalmodel.c: {
evalmodel.c:
evalmodel.c:                      found = 0;
evalmodel.c:    for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
evalmodel.c:      for(te = temin;te <= temax; te = te + gridsize_te)
evalmodel.c:        {
evalmodel.c:   parateor=model2(te, s2, tm, w);
evalmodel.c:
evalmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:
evalmodel.c:   if((s2 < ns2min)||(found==0))
```

```
evalmodel.c:        ns2min=s2;
evalmodel.c:    else if((s2 > ns2max)||(found==0))
evalmodel.c:        ns2max=s2;
evalmodel.c:    if((te < ntemin)||(found==0))
evalmodel.c:        ntemin=te;
evalmodel.c:    else if((te > ntemax)||(found==0))
evalmodel.c:        ntemax=te;
evalmodel.c:                            found = 1;
evalmodel.c:     }
evalmodel.c:
evalmodel.c: }
evalmodel.c:
evalmodel.c:     if(found == 1)
evalmodel.c:     {
evalmodel.c:     if(s2min!=ns2min)
evalmodel.c:     s2min=ns2min-gridsize_s2;
evalmodel.c:     if(s2max!=ns2max)
evalmodel.c:     s2max=ns2max+gridsize_s2;
evalmodel.c:     if(temin!=ntemin)
evalmodel.c:     temin=ntemin-gridsize_te;
evalmodel.c:     if(temax!=ntemax)
evalmodel.c:     temax=ntemax+gridsize_te;
evalmodel.c:     }
evalmodel.c:
evalmodel.c:                 ngridsize_s2=(s2max-s2min)/10.0;
evalmodel.c:  ngridsize_te=(temax-temin)/10.0;
evalmodel.c:
evalmodel.c:
evalmodel.c:  if((ngridsize_s2/gridsize_s2 > 0.5) ||
evalmodel.c:                     (fgridsize_s2/ngridsize_s2 > 0.2) || (ngridsize_s2 < fgridsize_s2 ))
evalmodel.c:  {
evalmodel.c:     gridsize_s2 = fgridsize_s2;
evalmodel.c:                 }
evalmodel.c:  else
evalmodel.c:  {
evalmodel.c:     gridsize_s2 = ngridsize_s2;
evalmodel.c:  }
evalmodel.c:
evalmodel.c:  if((ngridsize_te/gridsize_te > 0.5) ||
evalmodel.c:                     (fgridsize_te/ngridsize_te > 0.2) || (ngridsize_te < fgridsize_te ))
evalmodel.c:  {
evalmodel.c:     gridsize_te = fgridsize_te;
evalmodel.c:                 }
evalmodel.c:  else
evalmodel.c:  {
evalmodel.c:     gridsize_te = ngridsize_te;
evalmodel.c:  }
evalmodel.c:
evalmodel.c:
evalmodel.c:         }
evalmodel.c:
evalmodel.c:
evalmodel.c:
evalmodel.c:  found=0;
evalmodel.c:  for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
evalmodel.c:    for(te = temin;te <= temax; te = te + gridsize_te)
evalmodel.c:        {
evalmodel.c:
evalmodel.c:   parateor=model2(te, s2, tm, w);
evalmodel.c:
evalmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:   if((s2 < ns2min)||(found==0))
evalmodel.c:        ns2min=s2;
evalmodel.c:   else if((s2 > ns2max)||(found==0))
evalmodel.c:        ns2max=s2;
evalmodel.c:   if((te < ntemin)||(found==0))
evalmodel.c:        ntemin=te;
evalmodel.c:   else if((te > ntemax)||(found==0))
evalmodel.c:        ntemax=te;
evalmodel.c:                            found = 1;
evalmodel.c:     acertados++;
evalmodel.c:     evaluation=1;
evalmodel.c:                            }
evalmodel.c:                          }
evalmodel.c: totales=tm/(fgridsize_te*fgridsize_s2);
```

```
evalmodel.c:
evalmodel.c:              s2min=ns2min;
evalmodel.c:      s2max=ns2max;
evalmodel.c:              sf2min=999.999;
evalmodel.c:      sf2max=999.999;
evalmodel.c:              ss2min=999.999;
evalmodel.c:      ss2max=999.999;
evalmodel.c:      temin=ntemin;
evalmodel.c:      temax=ntemax;
evalmodel.c:      rexmin=999.999;
evalmodel.c:      rexmax=999.999;
evalmodel.c:
evalmodel.c:
evalmodel.c:
evalmodel.c: break;
evalmodel.c:
evalmodel.c:          case 3:
evalmodel.c:              s2min=0.0;
evalmodel.c:      s2max=1.0;
evalmodel.c:      rexmin=0.0;
evalmodel.c:      rexmax=REXMAX;
evalmodel.c:      gridsize_s2 = 0.1;
evalmodel.c:              gridsize_rex = (rexmax-rexmin) /10.0 ;
evalmodel.c:
evalmodel.c:      while ((gridsize_s2 != fgridsize_s2) && (gridsize_rex != fgridsize_rex))
evalmodel.c: {
evalmodel.c:    found=0;
evalmodel.c:    for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
evalmodel.c:      for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
evalmodel.c:        {
evalmodel.c:   parateor=model3(rex, s2, tm, w);
evalmodel.c:
evalmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:   if((s2 < ns2min)||(found==0))
evalmodel.c:        ns2min=s2;
evalmodel.c:   else if((s2 > ns2max)||(found==0))
evalmodel.c:        ns2max=s2;
evalmodel.c:   if((rex < nrexmin)||(found==0))
evalmodel.c:        nrexmin=rex;
evalmodel.c:   else if((rex > nrexmax)||(found==0))
evalmodel.c:        nrexmax=rex;
evalmodel.c:                             found=1;
evalmodel.c:    }
evalmodel.c: }
evalmodel.c: if(found==1)
evalmodel.c: {
evalmodel.c: if(ns2min!=s2min)
evalmodel.c:                     s2min=ns2min-gridsize_s2;
evalmodel.c: if(ns2max!=s2max)
evalmodel.c:                     s2max=ns2max+gridsize_s2;
evalmodel.c: if(nrexmin!=rexmin)
evalmodel.c:                     rexmin=nrexmin-gridsize_rex;
evalmodel.c: if(nrexmin!=rexmin)
evalmodel.c:                     rexmax=nrexmax+gridsize_rex;
evalmodel.c:     }
evalmodel.c:
evalmodel.c:
evalmodel.c:                 ngridsize_s2=(ns2max-ns2min)/10.0;
evalmodel.c:  ngridsize_rex=(nrexmax-nrexmin)/10.0;
evalmodel.c:
evalmodel.c:  if((ngridsize_s2/gridsize_s2 > 0.5) ||
evalmodel.c:     (fgridsize_s2/ngridsize_s2 > 0.2) || (ngridsize_s2 < fgridsize_s2 ))
evalmodel.c:  {
evalmodel.c:     gridsize_s2 = fgridsize_s2;
evalmodel.c:                 }
evalmodel.c:  else
evalmodel.c:  {
evalmodel.c:     gridsize_s2 = ngridsize_s2;
evalmodel.c:  }
evalmodel.c:
evalmodel.c:                 if((ngridsize_rex/gridsize_rex > 0.5) ||
evalmodel.c:                     (fgridsize_rex/ngridsize_rex > 0.2) || (ngridsize_rex < fgridsize_rex ))
evalmodel.c:                 {
evalmodel.c:                    gridsize_rex = fgridsize_rex;
evalmodel.c:                 }
```

```
evalmodel.c:                 else
evalmodel.c:                 {
evalmodel.c:                    gridsize_rex = ngridsize_rex;
evalmodel.c:                 }
evalmodel.c:
evalmodel.c:
evalmodel.c:                }
evalmodel.c:
evalmodel.c:
evalmodel.c: found=0;
evalmodel.c:  for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
evalmodel.c:    for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
evalmodel.c:        {
evalmodel.c:   parateor=model3(rex, s2, tm, w);
evalmodel.c:
evalmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:   if((s2 < ns2min)||(found==0))
evalmodel.c:        ns2min=s2;
evalmodel.c:   else if((s2 > ns2max)||(found==0))
evalmodel.c:        ns2max=s2;
evalmodel.c:   if((rex < nrexmin)||(found==0))
evalmodel.c:        nrexmin=rex;
evalmodel.c:   else if((rex > nrexmax)||(found==0))
evalmodel.c:        nrexmax=rex;
evalmodel.c:                          found=1;
evalmodel.c:    acertados++;
evalmodel.c:    evaluation=1;
evalmodel.c:                          }
evalmodel.c:                        }
evalmodel.c: totales=REXMAX/(fgridsize_rex*fgridsize_s2);
evalmodel.c:             s2min=ns2min;
evalmodel.c:     s2max=ns2max;
evalmodel.c:             sf2min=999.999;
evalmodel.c:     sf2max=999.999;
evalmodel.c:             ss2min=999.999;
evalmodel.c:     ss2max=999.999;
evalmodel.c:     temin=999.999;
evalmodel.c:     temax=999.999;
evalmodel.c:     rexmin=nrexmin;
evalmodel.c:     rexmax=nrexmax;
evalmodel.c:
evalmodel.c: break;
evalmodel.c:
evalmodel.c:         case 4:
evalmodel.c:             s2min=0.0;
evalmodel.c:     s2max=1.0;
evalmodel.c:     temin=0.0;
evalmodel.c:     temax=tm;
evalmodel.c:     rexmin=0.0;
evalmodel.c:     rexmax=REXMAX;
evalmodel.c:             gridsize_rex = (rexmax-rexmin) /10.0 ;
evalmodel.c:     gridsize_s2 = 0.1;
evalmodel.c:             gridsize_te = tm /10.0 ;
evalmodel.c:
evalmodel.c:     while ((gridsize_s2 != fgridsize_s2) && (gridsize_te != fgridsize_te) &&
evalmodel.c:    (gridsize_rex != fgridsize_rex))
evalmodel.c: {
evalmodel.c:
evalmodel.c:    found=0;
evalmodel.c:    for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
evalmodel.c:      for(te = temin;te <= temax; te = te + gridsize_te)
evalmodel.c:        for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
evalmodel.c:        {
evalmodel.c:   parateor=model4(te, s2, rex, tm, w);
evalmodel.c:
evalmodel.c:
evalmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:   if((s2 < ns2min)||(found==0))
evalmodel.c:        ns2min=s2;
evalmodel.c:   else if((s2 > ns2max)||(found==0))
evalmodel.c:        ns2max=s2;
evalmodel.c:   if((te < ntemin)||(found==0))
evalmodel.c:        ntemin=te;
evalmodel.c:   else if((te > ntemax)||(found==0))
```

```
evalmodel.c:        ntemax=te;
evalmodel.c:   if((rex < nrexmin)||(found==0))
evalmodel.c:        nrexmin=rex;
evalmodel.c:   else if((rex > nrexmax)||(found==0))
evalmodel.c:        nrexmax=rex;
evalmodel.c:                            found=1;
evalmodel.c:    }
evalmodel.c: }
evalmodel.c:
evalmodel.c: if( found == 1)
evalmodel.c:    {
evalmodel.c:     if(ns2min!=s2min)
evalmodel.c:                     s2min=ns2min-gridsize_s2;
evalmodel.c:     if(ns2max!=s2max)
evalmodel.c:                     s2max=ns2max+gridsize_s2;
evalmodel.c:     if(ntemin!=temin)
evalmodel.c:                     temin=ntemin-gridsize_te;
evalmodel.c:     if(ntemax!=temax)
evalmodel.c:                     temax=ntemax+gridsize_te;
evalmodel.c:     if(nrexmin!=rexmin)
evalmodel.c:                     rexmin=nrexmin-gridsize_rex;
evalmodel.c:     if(nrexmax!=rexmax)
evalmodel.c:                     rexmax=nrexmax+gridsize_rex;
evalmodel.c:     }
evalmodel.c:
evalmodel.c:
evalmodel.c:                 ngridsize_s2=(ns2max-ns2min)/10.0;
evalmodel.c:  ngridsize_te=(ntemax-ntemin)/10.0;
evalmodel.c:  ngridsize_rex=(nrexmax-nrexmin)/10.0;
evalmodel.c:
evalmodel.c:
evalmodel.c:                 if((ngridsize_s2/gridsize_s2 > 0.5) ||
evalmodel.c:                    (fgridsize_s2/ngridsize_s2 > 0.2) || (ngridsize_s2 < fgridsize_s2 ))
evalmodel.c:                 {
evalmodel.c:                    gridsize_s2 = fgridsize_s2;
evalmodel.c:                 }
evalmodel.c:                 else
evalmodel.c:                 {
evalmodel.c:                    gridsize_s2 = ngridsize_s2;
evalmodel.c:                 }
evalmodel.c:
evalmodel.c:                 if((ngridsize_te/gridsize_te > 0.5) ||
evalmodel.c:                     (fgridsize_te/ngridsize_te > 0.2) || (ngridsize_te < fgridsize_te ))
evalmodel.c:                 {
evalmodel.c:                    gridsize_te = fgridsize_te;
evalmodel.c:                 }
evalmodel.c:                 else
evalmodel.c:                 {
evalmodel.c:                    gridsize_te = ngridsize_te;
evalmodel.c:                 }
evalmodel.c:
evalmodel.c:                 if((ngridsize_rex/gridsize_rex > 0.5) ||
evalmodel.c:                     (fgridsize_rex/ngridsize_rex > 0.2) || (ngridsize_rex < fgridsize_rex ))
evalmodel.c:                 {
evalmodel.c:                    gridsize_rex = fgridsize_rex;
evalmodel.c:                 }
evalmodel.c:                 else
evalmodel.c:                 {
evalmodel.c:                    gridsize_rex = ngridsize_rex;
evalmodel.c:                 }
evalmodel.c:
evalmodel.c:
evalmodel.c: }
evalmodel.c:  found=0;
evalmodel.c:  for(s2 = s2min;s2 <= s2max; s2 = s2 + gridsize_s2)
evalmodel.c:    for(te = temin;te <= temax; te = te + gridsize_te)
evalmodel.c:        for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
evalmodel.c:        {
evalmodel.c:   parateor=model4(te, s2, rex, tm, w);
evalmodel.c:
evalmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:   if((s2 < ns2min)||(found==0))
evalmodel.c:        ns2min=s2;
evalmodel.c:   else if((s2 > ns2max)||(found==0))
evalmodel.c:        ns2max=s2;
```

```
evalmodel.c:   if((te < ntemin)||(found==0))
evalmodel.c:        ntemin=te;
evalmodel.c:   else if((te > ntemax)||(found==0))
evalmodel.c:        ntemax=te;
evalmodel.c:   if((rex < nrexmin)||(found==0))
evalmodel.c:        nrexmin=rex;
evalmodel.c:   else if((rex > nrexmax)||(found==0))
evalmodel.c:        nrexmax=rex;
evalmodel.c:                           found=1;
evalmodel.c:                             acertados++;
evalmodel.c:    evaluation=1;
evalmodel.c:                           }
evalmodel.c:                         }
evalmodel.c:                totales=REXMAX*tm/(fgridsize_te*fgridsize_rex*fgridsize_s2);
evalmodel.c:             s2min=ns2min;
evalmodel.c:     s2max=ns2max;
evalmodel.c:             sf2min=999.999;
evalmodel.c:     sf2max=999.999;
evalmodel.c:             ss2min=999.999;
evalmodel.c:     ss2max=999.999;
evalmodel.c:     temin=ntemin;
evalmodel.c:     temax=ntemin;
evalmodel.c:     rexmin=nrexmin;
evalmodel.c:     rexmax=nrexmax;
evalmodel.c:
evalmodel.c: break;
evalmodel.c:
evalmodel.c:         case 5:
evalmodel.c:             sf2min=0.0;
evalmodel.c:     sf2max=1.0;
evalmodel.c:             ss2min=0.0;
evalmodel.c:     ss2max=1.0;
evalmodel.c:     temin=0.0;
evalmodel.c:     temax=tm;
evalmodel.c:             nsf2min=0.0;
evalmodel.c:     nsf2max=1.0;
evalmodel.c:             nss2min=0.0;
evalmodel.c:     nss2max=1.0;
evalmodel.c:     ntemin=0.0;
evalmodel.c:     ntemax=tm;
evalmodel.c:
evalmodel.c:     gridsize_sf2 = 0.1;
evalmodel.c:     gridsize_ss2 = 0.1;
evalmodel.c:             gridsize_te = tm /10.0 ;
evalmodel.c:     while ((gridsize_ss2 != fgridsize_ss2) && (gridsize_te != fgridsize_te)
evalmodel.c:                 && (gridsize_sf2 != fgridsize_sf2) )
evalmodel.c: {
evalmodel.c:                  found=0;
evalmodel.c:    for(ss2 = ss2min;ss2 <= ss2max; ss2 = ss2 + gridsize_ss2)
evalmodel.c:     for(sf2 = sf2min;sf2 <= sf2max; sf2 = sf2 + gridsize_sf2)
evalmodel.c:      for(te = temin;te <= temax; te = te + gridsize_te)
evalmodel.c:        {
evalmodel.c:   parateor=model5(te, ss2, sf2, tm, w);
evalmodel.c:
evalmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:
evalmodel.c:                           if((ss2 < nss2min)||(found==0))
evalmodel.c:                                nss2min=ss2;
evalmodel.c:                           else if((ss2 > nss2max)||(found==0))
evalmodel.c:                                nss2max=ss2;
evalmodel.c:                           if((sf2 < nsf2min)||(found==0))
evalmodel.c:                                nsf2min=sf2;
evalmodel.c:                           else if((sf2 > nsf2max)||(found==0))
evalmodel.c:                                nsf2max=sf2;
evalmodel.c:                           if((te < ntemin)||(found==0))
evalmodel.c:                                ntemin=te;
evalmodel.c:                           else if((te > ntemax)||(found==0))
evalmodel.c:                                ntemax=te;
evalmodel.c:                           found = 1;
evalmodel.c:
evalmodel.c:    }
evalmodel.c: }
evalmodel.c:
evalmodel.c: if(found == 1)
evalmodel.c: {
```

```
evalmodel.c:   if(nss2min!=ss2min)
evalmodel.c:                     ss2min=nss2min-gridsize_ss2;
evalmodel.c:   if(nss2max!=ss2max)
evalmodel.c:                     ss2max=nss2max+gridsize_ss2;
evalmodel.c:   if(nsf2min!=sf2min)
evalmodel.c:                     sf2min=nsf2min-gridsize_sf2;
evalmodel.c:   if(nsf2max!=sf2max)
evalmodel.c:                     sf2max=nsf2max+gridsize_sf2;
evalmodel.c:   if(ntemin!=temin)
evalmodel.c:                     temin=ntemin-gridsize_te;
evalmodel.c:   if(ntemax!=temax)
evalmodel.c:                     temax=ntemax+gridsize_te;
evalmodel.c:     }
evalmodel.c:
evalmodel.c:
evalmodel.c:                 ngridsize_sf2=(nsf2max-nsf2min)/10.0;
evalmodel.c:                 ngridsize_ss2=(nss2max-nss2min)/10.0;
evalmodel.c:  ngridsize_te=(ntemax-ntemin)/10.0;
evalmodel.c:
evalmodel.c:
evalmodel.c:                 if((ngridsize_ss2/gridsize_ss2 > 0.5) ||
evalmodel.c:                     (fgridsize_ss2/ngridsize_ss2 > 0.2) || (ngridsize_ss2 < fgridsize_ss2 ))
evalmodel.c:                 {
evalmodel.c:                    gridsize_ss2 = fgridsize_ss2;
evalmodel.c:                 }
evalmodel.c:                 else
evalmodel.c:                 {
evalmodel.c:                    gridsize_ss2 = ngridsize_ss2;
evalmodel.c:                 }
evalmodel.c:
evalmodel.c:                 if((ngridsize_sf2/gridsize_sf2 > 0.5) ||
evalmodel.c:                     (fgridsize_sf2/ngridsize_sf2 < 0.2) || (ngridsize_sf2 < fgridsize_sf2 ))
evalmodel.c:                 {
evalmodel.c:                    gridsize_sf2 = fgridsize_sf2;
evalmodel.c:                 }
evalmodel.c:                 else
evalmodel.c:                 {
evalmodel.c:                    gridsize_sf2 = ngridsize_sf2;
evalmodel.c:                 }
evalmodel.c:
evalmodel.c:                 if((ngridsize_te/gridsize_te > 0.5) ||
evalmodel.c:                     (fgridsize_te/ngridsize_te > 0.2) || (ngridsize_te < fgridsize_te ))
evalmodel.c:                 {
evalmodel.c:                    gridsize_te = fgridsize_te;
evalmodel.c:                 }
evalmodel.c:                 else
evalmodel.c:                 {
evalmodel.c:                    gridsize_te = ngridsize_te;
evalmodel.c:                 }
evalmodel.c:
evalmodel.c:                }
evalmodel.c:
evalmodel.c:
evalmodel.c:  for(ss2 = ss2min;ss2 <= ss2max; ss2 = ss2 + gridsize_ss2)
evalmodel.c:   for(sf2 = sf2min;sf2 <= sf2max; sf2 = sf2 + gridsize_sf2)
evalmodel.c:    for(te = temin;te <= temax; te = te + gridsize_te)
evalmodel.c:        {
evalmodel.c:   parateor=model5(te, ss2, sf2, tm, w);
evalmodel.c:
evalmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:                           if((ss2 < nss2min)||(found==0))
evalmodel.c:                                nss2min=ss2;
evalmodel.c:                           else if((ss2 > nss2max)||(found==0))
evalmodel.c:                                nss2max=ss2;
evalmodel.c:                           if((sf2 < nsf2min)||(found==0))
evalmodel.c:                                nsf2min=sf2;
evalmodel.c:                           else if((sf2 > nsf2max)||(found==0))
evalmodel.c:                                nsf2max=sf2;
evalmodel.c:                           if((te < ntemin)||(found==0))
evalmodel.c:                                ntemin=te;
evalmodel.c:                           else if((te > ntemax)||(found==0))
evalmodel.c:                                ntemax=te;
evalmodel.c:                           found = 1;
evalmodel.c:                             acertados++;
evalmodel.c:     evaluation=1;
```

```
evalmodel.c:                          }
evalmodel.c:                        }
evalmodel.c:                       totales=tm/(fgridsize_te*fgridsize_ss2*fgridsize_sf2);
evalmodel.c:              s2min=999.999;
evalmodel.c:      s2max=999.999;
evalmodel.c:              sf2min=nsf2min;
evalmodel.c:      sf2max=nsf2max;
evalmodel.c:              ss2min=nss2min;
evalmodel.c:      ss2max=nss2min;
evalmodel.c:      temin=ntemin;
evalmodel.c:      temax=ntemin;
evalmodel.c:      rexmin=999.999;
evalmodel.c:      rexmax=999.999;
evalmodel.c: break;
evalmodel.c:
evalmodel.c:          case 6:
evalmodel.c:
evalmodel.c:
evalmodel.c:              sf2min=0.0;
evalmodel.c:      sf2max=1.0;
evalmodel.c:              ss2min=0.0;
evalmodel.c:      ss2max=1.0;
evalmodel.c:      temin=0.0;
evalmodel.c:      temax=tm;
evalmodel.c:      rexmin=0.0;
evalmodel.c:      rexmax=REXMAX;
evalmodel.c:              nsf2min=0.0;
evalmodel.c:      nsf2max=1.0;
evalmodel.c:              nss2min=0.0;
evalmodel.c:      nss2max=1.0;
evalmodel.c:      ntemin=0.0;
evalmodel.c:      ntemax=tm;
evalmodel.c:      nrexmin=0.0;
evalmodel.c:      nrexmax=REXMAX;
evalmodel.c:
evalmodel.c:      gridsize_sf2 = 0.1;
evalmodel.c:      gridsize_ss2 = 0.1;
evalmodel.c:              gridsize_te = tm /10.0 ;
evalmodel.c:      gridsize_rex = (rexmax-rexmin) /10.0 ;
evalmodel.c:
evalmodel.c:      while ((gridsize_ss2 != fgridsize_ss2) && (gridsize_te != fgridsize_te)
evalmodel.c:                  && (gridsize_sf2 != fgridsize_sf2) && (gridsize_rex != fgridsize_rex) )
evalmodel.c: {
evalmodel.c:    for(ss2 = ss2min;ss2 <= ss2max; ss2 = ss2 + gridsize_ss2)
evalmodel.c:     for(sf2 = sf2min;sf2 <= sf2max; sf2 = sf2 + gridsize_sf2)
evalmodel.c:      for(te = temin;te <= temax; te = te + gridsize_te)
evalmodel.c:        for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
evalmodel.c:        {
evalmodel.c:   parateor=model6(te, ss2, sf2,rex,  tm, w);
evalmodel.c:
evalmodel.c:   if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:
evalmodel.c:                            if((ss2 < nss2min)||(found==0))
evalmodel.c:                                 nss2min=ss2;
evalmodel.c:                            else if((ss2 > nss2max)||(found==0))
evalmodel.c:                                 nss2max=ss2;
evalmodel.c:                            if((sf2 < nsf2min)||(found==0))
evalmodel.c:                                 nsf2min=sf2;
evalmodel.c:                            else if((sf2 > nsf2max)||(found==0))
evalmodel.c:                                 nsf2max=sf2;
evalmodel.c:                            if((te < ntemin)||(found==0))
evalmodel.c:                                 ntemin=te;
evalmodel.c:                            else if((te > ntemax)||(found==0))
evalmodel.c:                                 ntemax=te;
evalmodel.c:                            if((rex < nrexmin)||(found==0))
evalmodel.c:                                 nrexmin=rex;
evalmodel.c:                            else if((rex > nrexmax)||(found==0))
evalmodel.c:                                 nrexmax=rex;
evalmodel.c:
evalmodel.c:                            found = 1;
evalmodel.c:
evalmodel.c:
evalmodel.c:    }
evalmodel.c: }
evalmodel.c: if(found == 1)
```

```
evalmodel.c: {
evalmodel.c: if(nss2min!=ss2min)
evalmodel.c:                       ss2min=nss2min-gridsize_ss2;
evalmodel.c: if(nss2max!=ss2max)
evalmodel.c:                       ss2max=nss2max+gridsize_ss2;
evalmodel.c: if(nsf2min!=sf2min)
evalmodel.c:                       sf2min=nsf2min-gridsize_sf2;
evalmodel.c: if(nsf2max!=sf2max)
evalmodel.c:                       sf2max=nsf2max+gridsize_sf2;
evalmodel.c: if(ntemin!=temin)
evalmodel.c:                       temin=ntemin-gridsize_te;
evalmodel.c: if(ntemax!=temax)
evalmodel.c:                       temax=ntemax+gridsize_te;
evalmodel.c: if(nrexmin!=rexmin)
evalmodel.c:                       rexmin=nrexmin-gridsize_rex;
evalmodel.c: if(nrexmax!=rexmax)
evalmodel.c:                       rexmax=nrexmax+gridsize_rex;
evalmodel.c: }
evalmodel.c:
evalmodel.c:
evalmodel.c:                   ngridsize_sf2=(nsf2max-nsf2min)/10.0;
evalmodel.c:                   ngridsize_ss2=(nss2max-nss2min)/10.0;
evalmodel.c:  ngridsize_te=(ntemax-ntemin)/10.0;
evalmodel.c:  ngridsize_rex=(nrexmax-nrexmin)/10.0;
evalmodel.c:
evalmodel.c:                   if((ngridsize_ss2/gridsize_ss2 > 0.5) ||
evalmodel.c:                       (fgridsize_ss2/ngridsize_ss2 > 0.2) || (ngridsize_ss2 < fgridsize_ss2 ))
evalmodel.c:                   {
evalmodel.c:                      gridsize_ss2 = fgridsize_ss2;
evalmodel.c:                   }
evalmodel.c:                   else
evalmodel.c:                   {
evalmodel.c:                      gridsize_ss2 = ngridsize_ss2;
evalmodel.c:                   }
evalmodel.c:
evalmodel.c:                   if((ngridsize_sf2/gridsize_sf2 > 0.5) ||
evalmodel.c:                       (fgridsize_sf2/ngridsize_sf2 > 0.2) || (ngridsize_sf2 < fgridsize_sf2 ))
evalmodel.c:                   {
evalmodel.c:                      gridsize_sf2 = fgridsize_sf2;
evalmodel.c:                   }
evalmodel.c:                   else
evalmodel.c:                   {
evalmodel.c:                      gridsize_sf2 = ngridsize_sf2;
evalmodel.c:                   }
evalmodel.c:
evalmodel.c:                   if((ngridsize_te/gridsize_te > 0.5) ||
evalmodel.c:                       (fgridsize_te/ngridsize_te > 0.2) || (ngridsize_te < fgridsize_te ))
evalmodel.c:                   {
evalmodel.c:                      gridsize_te = fgridsize_te;
evalmodel.c:                   }
evalmodel.c:                   else
evalmodel.c:                   {
evalmodel.c:                      gridsize_te = ngridsize_te;
evalmodel.c:                   }
evalmodel.c:
evalmodel.c:                   if((ngridsize_rex/gridsize_rex > 0.5) ||
evalmodel.c:                       (fgridsize_rex/ngridsize_rex > 0.2) || (ngridsize_rex < fgridsize_rex ))
evalmodel.c:                   {
evalmodel.c:                      gridsize_rex = fgridsize_rex;
evalmodel.c:                   }
evalmodel.c:                   else
evalmodel.c:                   {
evalmodel.c:                      gridsize_rex = ngridsize_rex;
evalmodel.c:                   }
evalmodel.c:
evalmodel.c:                  }
evalmodel.c: found=0;
evalmodel.c:  for(ss2 = ss2min;ss2 <= ss2max; ss2 = ss2 + gridsize_ss2)
evalmodel.c:   for(sf2 = sf2min;sf2 <= sf2max; sf2 = sf2 + gridsize_sf2)
evalmodel.c:    for(te = temin;te <= temax; te = te + gridsize_te)
evalmodel.c:      for(rex = rexmin;rex <= rexmax; rex = rex + gridsize_rex)
evalmodel.c:        {
evalmodel.c:   parateor=model6(te, ss2, sf2, rex, tm, w);
evalmodel.c:
evalmodel.c:  if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
```

```
evalmodel.c:                            if((ss2 < nss2min)||(found==0))
evalmodel.c:                                 nss2min=ss2;
evalmodel.c:                            else if((ss2 > nss2max)||(found==0))
evalmodel.c:                                 nss2max=ss2;
evalmodel.c:                            if((sf2 < nsf2min)||(found==0))
evalmodel.c:                                 nsf2min=sf2;
evalmodel.c:                            else if((sf2 > nsf2max)||(found==0))
evalmodel.c:                                 nsf2max=sf2;
evalmodel.c:                            if((te < ntemin)||(found==0))
evalmodel.c:                                 ntemin=te;
evalmodel.c:                            else if((te > ntemax)||(found==0))
evalmodel.c:                                 ntemax=te;
evalmodel.c:                            if((rex < nrexmin)||(found==0))
evalmodel.c:                                 nrexmin=rex;
evalmodel.c:                            else if((rex > nrexmax)||(found==0))
evalmodel.c:                                 nrexmax=rex;
evalmodel.c:
evalmodel.c:                            found = 1;
evalmodel.c:     acertados++;
evalmodel.c:     evaluation=1;
evalmodel.c:                            }
evalmodel.c:                           }
evalmodel.c: totales=tm*REXMAX/(fgridsize_ss2*fgridsize_sf2*fgridsize_te*fgridsize_rex);
evalmodel.c:              s2min=999.999;
evalmodel.c:      s2max=999.999;
evalmodel.c:              sf2min=nsf2min;
evalmodel.c:      sf2max=nsf2max;
evalmodel.c:              ss2min=nss2min;
evalmodel.c:      ss2max=nss2min;
evalmodel.c:      temin=ntemin;
evalmodel.c:      temax=ntemin;
evalmodel.c:      rexmin=nss2min;
evalmodel.c:      rexmax=nss2min;
evalmodel.c:
evalmodel.c: break;
evalmodel.c:
evalmodel.c:
evalmodel.c: }
evalmodel.c:
evalmodel.c:
evalmodel.c:
evalmodel.c: solucion.sols2.max=s2max;
evalmodel.c: solucion.solsf2.max=sf2max;
evalmodel.c: solucion.solss2.max=ss2max;
evalmodel.c: solucion.solrex.max=rexmax;
evalmodel.c: solucion.solte.max=temax;
evalmodel.c: solucion.sols2.min=s2min;
evalmodel.c: solucion.solss2.min=ss2min;
evalmodel.c: solucion.solsf2.min=sf2min;
evalmodel.c: solucion.solrex.min=rexmin;
evalmodel.c: solucion.solte.min=temin;
evalmodel.c: solucion.sols2.mean=(s2max+s2min)/2;
evalmodel.c: solucion.solss2.mean=(ss2max+ss2min)/2;
evalmodel.c: solucion.solsf2.mean=(sf2max+sf2min)/2;
evalmodel.c: solucion.solte.mean=(temax+temin)/2;
evalmodel.c: solucion.solrex.mean=(rexmax+rexmin)/2;
evalmodel.c: solucion.percent=acertados/totales*100;
evalmodel.c: solucion.test=0.0;
evalmodel.c: /*
evalmodel.c: for(tm=tmmin;tm <=tmmax;tm+TMGRID)
evalmodel.c: {
evalmodel.c: switch(typemodel)
evalmodel.c: {
evalmodel.c:    case 2:
evalmodel.c:                            parateor=model2(solucion.solte.mean, solucion.sols2.mean, tm, w);
evalmodel.c:   break;
evalmodel.c:    case 3:
evalmodel.c:                            parateor=model3(solucion.solrex.mean, solucion.sols2.mean, tm, w);
evalmodel.c:   break;
evalmodel.c:    case 4:
evalmodel.c:                            parateor=model4(solucion.solte.mean, solucion.sols2.mean, solucion.solrex.mean, t
m, w);
evalmodel.c:   break;
evalmodel.c:    case 5:
evalmodel.c:                            parateor=model5(solucion.solte.mean, solucion.solss2.mean, solucion.solsf2.mean,
tm, w);
```

```
evalmodel.c:   break;
evalmodel.c:    case 6:
evalmodel.c:                              parateor=model6(solucion.solte.mean, solucion.solss2.mean, solucion.solsf2.mean,s
olucion.solrex.mean,  tm, w);
evalmodel.c:   break;
evalmodel.c:   }
evalmodel.c:                     if(test_para(paraexp, uncert, parateor)==1)
evalmodel.c:   {
evalmodel.c:    solucion.test=1);
evalmodel.c:    break;
evalmodel.c:    }
evalmodel.c:   }
evalmodel.c: */
evalmodel.c:
evalmodel.c:
evalmodel.c:
evalmodel.c:                  return solucion;
evalmodel.c: }
```

```
functions.c: #include <math.h>
functions.c: #include <stdio.h>
functions.c: /* #include "search.h"
functions.c: #include "functions.h"
functions.c: */
functions.c:
functions.c: #include "global.h"
functions.c: #include "constantes.h"
functions.c: #define GRIDSIZE 0.01
functions.c:
functions.c: #define minim(A,B) ((A) > (B) ? (B) : (A))
functions.c: #define sign(A,B) ((B) > 0 ? (A) : (-A))
functions.c:
functions.c:
functions.c:       float jwmf(float w,float tm)
functions.c: {
functions.c:       float result;
functions.c:
functions.c:       result=(2.0/5.0)*tm/(w*w*tm*tm+1.0);
functions.c:
functions.c:       return result;
functions.c: }
functions.c:
functions.c:       float ftm(float tm, float wh, float wx)
functions.c:       {
functions.c:       float denom,numer;
functions.c:
functions.c:
functions.c:       if(tm == 0.0)
functions.c: numer=1.0;
functions.c:       else
functions.c:       {
functions.c:         denom=jwmf(wh-wx,tm) + 3*jwmf(wx,tm) + 6*jwmf(wh+wx,tm);
functions.c:         numer=4*jwmf(0,tm) + 6*jwmf(wh,tm);
functions.c:         numer=(numer/denom+1)/2;
functions.c: }
functions.c:       return numer;
functions.c:       }
functions.c:
functions.c: int test_para(struct para paraexp, struct para uncert, struct para parateor)
functions.c: {
functions.c: int evaluation;
functions.c:
functions.c:         evaluation=0;
functions.c:
functions.c:
functions.c: if (( parateor.R1 <= (paraexp.R1 + uncert.R1)) &&
functions.c:     ( parateor.R1 >= (paraexp.R1 - uncert.R1)) &&
functions.c:     ( parateor.R2 <= (paraexp.R2 + uncert.R2)) &&
functions.c:     ( parateor.R2 >= (paraexp.R2 - uncert.R2)) &&
functions.c:     ( parateor.HNOE <= (paraexp.HNOE + uncert.HNOE)) &&
functions.c:     ( parateor.HNOE >= (paraexp.HNOE - uncert.HNOE)))
functions.c:     evaluation = 1;
functions.c:          else
```

```
functions.c:     evaluation = 0;
functions.c:
functions.c: return evaluation;
functions.c: }
functions.c:
functions.c: int getline(FILE *fpinp,char s[],int lim)
functions.c: {
functions.c:         int c,i;
functions.c:
functions.c:         i=0;
functions.c:         while (--lim > 0 && (c=getc(fpinp)) != EOF && c != '\n')
functions.c:            s[i++]=c;
functions.c:         if (c == '\n')
functions.c:            s[i++] = c;
functions.c:         s[i] = '\0';
functions.c:         return i;
functions.c: }
functions.c:
functions.c: float tmmed(float tminit, struct para residue[MAXRES], int test, int *m2, int nres)
functions.c: {
functions.c:       int i, tot;
functions.c:       float r2r1, r2r1med, r2r1medant, r2r1sum;
functions.c:
functions.c:       r2r1med=tminit;
functions.c:       r2r1medant=100;
functions.c: /*      printf("Test\n ");
functions.c:       printf("Test %d\n",test);
functions.c: */
functions.c: tot=0;
functions.c: r2r1sum=0.0;
functions.c:
functions.c: if(test == 0)
functions.c: {
functions.c:       while((r2r1med-r2r1medant)*(r2r1med-r2r1medant) > 0.2)
functions.c:        {
functions.c:         for(i=1; i <= nres; i++)
functions.c:         {
functions.c: /*    printf("%f %f \n",  residue[i].R2,residue[i].R1);
functions.c:    printf("%d \n",  test);
functions.c:    */
functions.c:            r2r1= residue[i].R2/residue[i].R1;
functions.c:            if((r2r1-r2r1med)*(r2r1-r2r1med) < TOL)
functions.c:              {
functions.c:             r2r1sum= r2r1sum + r2r1;
functions.c: /*     printf("%f \n", r2r1sum);  */
functions.c:             tot++;
functions.c:              }
functions.c:    }
functions.c:          r2r1medant=r2r1med;
functions.c:          r2r1med=r2r1sum/tot;
functions.c:
functions.c: /*         printf("%f %f \n",  r2r1medant, r2r1med); */
functions.c:        }
functions.c: }
functions.c: else
functions.c: {
functions.c:       while((r2r1med-r2r1medant)*(r2r1med-r2r1medant) > 0.1)
functions.c:        {
functions.c:         for(i=1; i <= nres; i++)
functions.c:         {
functions.c: /*    printf("%f %f \n",  residue[i].R2,residue[i].R1);
functions.c:    printf("%d \n",  test);
functions.c:    */
functions.c:    m2++;
functions.c:            r2r1= residue[i].R2/residue[i].R1;
functions.c:   /*       printf("%d \n",  *m2);  */
functions.c:            if(((r2r1-r2r1med)*(r2r1-r2r1med) < TOL)&&(*m2==1))
functions.c:              {
functions.c:             r2r1sum= r2r1sum + r2r1;
functions.c: /*     printf("%f \n", r2r1sum);  */
functions.c:             tot++;
functions.c:              }
functions.c:     }
functions.c:          r2r1medant=r2r1med;
functions.c:          r2r1med=r2r1sum/tot;
functions.c:
```

```
functions.c:  /*        printf("%f %f \n",  r2r1medant, r2r1med); */
functions.c:        }
functions.c: }
functions.c: return r2r1med;
functions.c: }
functions.c:
functions.c: float tmfr2r1(float tm, float r2r1, float min, float max,float wh, float wx)
functions.c: {
functions.c:     float r1, r2, r3, r4;
functions.c:
functions.c:     int iter;
functions.c:     float a, b, c, d, e, p, q, r, s, fa, fb, fc, xm;
functions.c:     float tol1;
functions.c:     float tol=1e-12;
functions.c:
functions.c:     wh=2*PI*1e9*wh;
functions.c:     wx=2*PI*1e9*wx;
functions.c:     a = min ;
functions.c:     b = max;
functions.c:     fa = ftm(a,wh,wx) - r2r1;
functions.c:     fb = ftm(b,wh,wx) - r2r1;
functions.c:     fc = fb;
functions.c:     for (iter = 1; iter <= 100; ++iter) {
functions.c: if (fb * fc > 0.0) {
functions.c:     c = a;
functions.c:     fc = fa;
functions.c:     d = b - a;
functions.c:     e = d;
functions.c: }
functions.c: if (fabs(fc) < fabs(fb)) {
functions.c:     a = b;
functions.c:     b = c;
functions.c:     c = a;
functions.c:     fa = fb;
functions.c:     fb = fc;
functions.c:     fc = fa;
functions.c: }
functions.c: tol1 = fabs(b) * 5e-8 + tol * 0.5;
functions.c: xm = (c - b) * 0.5;
functions.c: if (fabs(xm) <= tol1 || fb == 0.0) {
functions.c:     tm=b*1e9;
functions.c:     return tm;
functions.c: }
functions.c: /* fprintf(stdout,"una %f \n", b); */
functions.c: if (fabs(e) >= tol1 && fabs(fa) > fabs(fb)) {
functions.c:     s = fb / fa;
functions.c:     if (a == c) {
functions.c: p = xm * 2.0 * s;
functions.c: q = 1.0 - s;
functions.c:     } else {
functions.c: q = fa / fc;
functions.c: r = fb / fc;
functions.c: p = s * (xm * 2.0 * q * (q - r) - (b - a) * (r - 1.0));
functions.c: q = (q - 1.0) * (r - 1.0) * (s - 1.0);
functions.c:     }
functions.c:     if (p > 0.0) {
functions.c: q = -q;
functions.c:     }
functions.c:     p = fabs(p);
functions.c: /* Computing MIN */
functions.c:     r3 = xm * 3.0 * q - (r1 = tol1 * q, fabs(r1)), r4 =
functions.c:     (r2 = e * q, fabs(r2));
functions.c:     if (p * 2.0 < minim(r3,r4)) {
functions.c: e = d;
functions.c: d = p / q;
functions.c:     } else {
functions.c: d = xm;
functions.c: e = d;
functions.c:     }
functions.c: } else {
functions.c:     d = xm;
functions.c:     e = d;
functions.c: }
functions.c: a = b;
functions.c: fa = fb;
functions.c: if (fabs(d) > tol1) {
```

```
functions.c:     b += d;
functions.c: } else {
functions.c:     b += sign(tol1, xm);
functions.c: }
functions.c: fb = ftm(b,wh,wx) - r2r1;
functions.c:     }
functions.c:     fprintf(stdout,"no no no\n");
functions.c:     tm=b*1e9;
functions.c:     return tm;
functions.c: }
functions.c:
functions.c:
functions.c:
functions.c: struct sol marg(float r2r1,float x1,float  x2, float wh, float wx)
functions.c: {
functions.c:     int j;
functions.c:     struct sol result;
functions.c:     float  f1, f2;
functions.c:
functions.c:     wh=2*PI*1e9*wh;
functions.c:     wx=2*PI*1e9*wx;
functions.c:     result.min=x1*1e-9;
functions.c:     result.max=x2*1e-9;
functions.c:     result.mean = (result.min + result.min)/2.0;
functions.c:     f1 = ftm(result.min, wh, wx) - r2r1;
functions.c:     f2 = ftm(result.max, wh, wx) - r2r1;
functions.c:     for (j = 1; j <= 50; ++j) {
functions.c: if (f1 * f2 < 0.0) {
functions.c:     return result;
functions.c: }
functions.c: if (fabs(f1) < fabs(f2)) {
functions.c:     result.min += (result.min - result.max) * 1.6;
functions.c:     if (result.min < 0.0) {
functions.c: result.min = 0.0;
functions.c:     }
functions.c:     f1 = ftm(result.min, wh, wx) - r2r1;
functions.c: } else {
functions.c:     result.max += (result.max - result.min) * 1.6;
functions.c:     if (result.max < 0.0) {
functions.c: result.max = 0.0;
functions.c:     }
functions.c:     f2 = ftm(result.max, wh, wx) - r2r1;
functions.c: }
functions.c:     }
functions.c:     return result;
functions.c: }
```

```
help.c: #include <math.h>
help.c: #include <stdio.h>
help.c: #include <string.h>
help.c: /*
help.c: #include "search.h"
help.c: #include "functions.h"
help.c: #include "string.h"
help.c:
help.c: */
help.c:
help.c: #include "global.h"
help.c: #include "constantes.h"
help.c:
help.c: void init(char bandera[])
help.c: {
help.c: FILE *fpinp;
help.c: extern float limit;
help.c: extern float toldown;
help.c: extern float tolup;
help.c: extern float fgridsize_s2;
help.c: extern float fgridsize_ss2;
help.c: extern float fgridsize_sf2;
help.c: extern float fgridsize_te;
help.c: extern float fgridsize_rex;
help.c:
help.c: if(strcmp(bandera,"default")==0)
```

```
help.c:     {
help.c: limit=LIMIT;
help.c: toldown=TOLDOWN;
help.c: tolup=TOLUP;
help.c: fgridsize_s2=GRIDSIZE_S2;
help.c: fgridsize_te=GRIDSIZE_TE;
help.c: fgridsize_rex=GRIDSIZE_REX;
help.c: fgridsize_ss2=GRIDSIZE_SS2;
help.c: fgridsize_sf2=GRIDSIZE_SF2;
help.c: return;
help.c: }
help.c: else
help.c:     {
help.c:         fpinp(bandera,"w");
help.c: return;
help.c: }
```

```
init.c: #include <math.h>
init.c: #include <stdio.h>
init.c: #include <string.h>
init.c: /*
init.c: #include "search.h"
init.c: #include "functions.h"
init.c: #include "string.h"
init.c:
init.c: */
init.c:
init.c: #include "global.h"
init.c: #include "constantes.h"
init.c:
init.c: void init(char bandera[MAXLINE])
init.c: {
init.c:
init.c:
init.c: FILE *fpinp;
init.c: extern float negtol;
init.c: extern float postol;
init.c: extern float rexmaxg;
init.c: extern float fgridsize_s2;
init.c: extern float fgridsize_ss2;
init.c: extern float fgridsize_sf2;
init.c: extern float fgridsize_te;
init.c: extern float fgridsize_rex;
init.c: char line[MAXLINE];
init.c:
init.c: if(strcmp(bandera,"defecto")==0)
init.c:     {
init.c: negtol=NEGTOL;
init.c: postol=POSTOL;
init.c: fgridsize_s2=GRIDSIZE_S2;
init.c: fgridsize_te=GRIDSIZE_TE;
init.c: fgridsize_rex=GRIDSIZE_REX;
init.c: fgridsize_ss2=GRIDSIZE_S2;
init.c: fgridsize_sf2=GRIDSIZE_S2;
init.c: rexmaxg=REXMAX;
init.c: }
init.c: else
init.c:     {
init.c:         fpinp=fopen(bandera,"r");
init.c:         getline(fpinp, line, sizeof(line));
init.c:         sscanf(line, "%f %f ",&negtol, &postol);
init.c:         getline(fpinp, line, sizeof(line));
init.c:         sscanf(line, "%f ", &rexmaxg);
init.c:         getline(fpinp, line, sizeof(line));
init.c:         sscanf(line, "%f %f %f", &fgridsize_s2, &fgridsize_ss2, &fgridsize_sf2);
init.c:         getline(fpinp, line, sizeof(line));
init.c:         sscanf(line, "%f ", &fgridsize_te);
init.c:         getline(fpinp, line, sizeof(line));
init.c:         sscanf(line, "%f ", &fgridsize_rex);
init.c:         fclose(fpinp);
init.c: }
init.c:
init.c:         fprintf(stdout, "INIT> Tolerances: %f %f \n",negtol, postol);
```

```
init.c:         fprintf(stdout, "INIT> Rexmax: %f \n",rexmaxg);
init.c:         fprintf(stdout, "INIT> Gridsizes(s2,ss2,sf2): %f %f %f\n",fgridsize_s2,fgridsize_ss2,fgridsize_sf2);
init.c:         fprintf(stdout, "INIT> Gridsizes(te): %f\n",fgridsize_te);
init.c:         fprintf(stdout, "INIT> Gridsizes(rex): %f \n",fgridsize_rex);
init.c:
init.c:       return;
init.c:       }
```

```
main.c: #include <math.h>
main.c: #include <stdio.h>
main.c: #include <string.h>
main.c: #include <sys/times.h>
main.c: #include <limits.h>
main.c: /*
main.c: #include "search.h"
main.c: #include "functions.h"
main.c: #include "string.h"
main.c:
main.c: */
main.c:
main.c: #include "constantes.h"
main.c: #include "global.h"
main.c:
main.c: main()
main.c: {
main.c:
main.c: struct sol tmsolution;
main.c: float tm;
main.c: FILE *fpinp; /* *fpout; */
main.c: float w;
main.c: float limit;
main.c: char fileinp[MAXLINE],fileout[MAXLINE];
main.c: char line[MAXLINE];
main.c: char defecto[MAXLINE];
main.c: char command[MAXLINE],arginit[MAXLINE];
main.c: int calculado;
main.c: float tmmax, tmmin;
main.c:         float gtmmin, gtmmax;
main.c:
main.c:         struct  tms temp ;
main.c:         float tiempo_user, tiempo_sys;
main.c:
main.c: /* Default initialization of gridsizes, rexmax, negative and positive tolerance */
main.c:
main.c:         strcpy(defecto,"defecto");
main.c:         init(defecto);
main.c:         calculado=0;
main.c: fprintf(stdout,"################################################################");
main.c: fprintf(stdout,"\n");
main.c: fprintf(stdout,"\n");
main.c: fprintf(stdout,"\n");
main.c: fprintf(stdout,"            Wellcome to Searcher              \n");
main.c: fprintf(stdout,"\n");
main.c: fprintf(stdout,"\n");
main.c: fprintf(stdout,"\n");
main.c: fprintf(stdout,"################################################################");
main.c: fprintf(stdout,"\n");
main.c:
main.c:         while(getline(stdin,line,sizeof(line)) > 0)
main.c:         {
main.c:
main.c: /*  Get command */
main.c:
main.c: fprintf(stdout,"SCH> %s \n",line);
main.c: sscanf(line, "%s",&command);
main.c:
main.c: /* Get args and execute rutine */
main.c:
main.c:         if(strcmp(command,"init")==0)
main.c:         {
main.c: /* Initialization by the user */
main.c:
main.c: getline(stdin, line, sizeof(line));
```

```
main.c: sscanf(line, "%s",&arginit);
main.c: init(arginit);
main.c: }
main.c:         else if(strcmp(command,"calctm")==0)
main.c: {
main.c:
main.c: /* Calculation  of tm. You need a experimental
main.c: data file , the frequency and the limit of uncertainty */
main.c:
main.c: /* gtmmin=0.0;
main.c: gtmmax=100.0; */
main.c: getline(stdin, line, sizeof(line));
main.c: sscanf(line, "%s %f %f",&fileinp,&w, &limit);
main.c: fpinp = fopen(fileinp, "r");
main.c: tmsolution=calctm(fpinp, TMGRID, limit, w);
main.c: gtmmin=tmsolution.min;
main.c: gtmmax=tmsolution.max;
main.c: calculado=1;
main.c: fclose(fpinp);
main.c: }
main.c: else if(strcmp(command,"calcall")==0)
main.c:         {
main.c:         getline(stdin, line, sizeof(line));
main.c:         sscanf(line, "%s %f %f %f %f %s",&fileinp,&w, &limit, &tmmin, &tmmax, &fileout);
main.c:         fpinp = fopen(fileinp, "r");
main.c:
main.c: /* If limits for tm is -1 and -1, get limits from calcualted
main.c:   values... */
main.c:
main.c:         if(((tmmin!=-1)&&(tmmax!=-1))||(calculado==0))
main.c:           {
main.c:            fprintf(stdout,"SCH> Calculating models with Tm: %6.3f %6.3f \n",tmmin, tmmax);
main.c:            gtmmin=tmmin;
main.c:            gtmmax=tmmax;
main.c:            }
main.c:         calcall(gtmmin,gtmmax,fpinp,w,limit,fileout);
main.c: fclose(fpinp);
main.c:         }
main.c: else if(strcmp(command,"calctree")==0)
main.c:         {
main.c:         getline(stdin, line, sizeof(line));
main.c:         sscanf(line, "%s %f %f %f %f %s",&fileinp,&w, &limit, &tmmin, &tmmax, &fileout);
main.c:         fpinp = fopen(fileinp, "r");
main.c:         if(((tmmin!=-1)&&(tmmax!=-1))||(calculado==0))
main.c:           {
main.c:            fprintf(stdout,"SCH> Calculating models with Tm: %6.3f %6.3f \n",tmmin, tmmax);
main.c:            gtmmin=tmmin;
main.c:            gtmmax=tmmax;
main.c:            }
main.c:         calctree(gtmmin,gtmmax,fpinp,w,limit,fileout);
main.c: fclose(fpinp);
main.c:         }
main.c: else if(strcmp(command,"calcone")==0)
main.c:         {
main.c:         getline(stdin, line, sizeof(line));
main.c:         sscanf(line, "%s %f %f %f %f %s",&fileinp,&w, &limit, &tmmin, &tmmax, &fileout);
main.c:         fpinp = fopen(fileinp, "r");
main.c:         if(((tmmin!=-1)&&(tmmax!=-1))||(calculado==0))
main.c:           {
main.c:            fprintf(stdout,"SCH> Calculating models with Tm: %6.3f %6.3f \n",tmmin, tmmax);
main.c:            gtmmin=tmmin;
main.c:            gtmmax=tmmax;
main.c:            }
main.c:         calcone(gtmmin,gtmmax,fpinp,w,limit,fileout);
main.c: fclose(fpinp);
main.c:         }
main.c: else if(strcmp(command,"simul")==0)
main.c:         {
main.c:         getline(stdin, line, sizeof(line));
main.c:         sscanf(line, "%s %f %f %s",&fileinp,&w, &tm, &fileout);
main.c:         fpinp = fopen(fileinp, "r");
main.c:         simul(tm,w,fpinp,fileout);
main.c: fclose(fpinp);
main.c:         }
main.c: else if(strcmp(command,"end")==0)
main.c: {
```

```
main.c:         times(&temp);
main.c:         tiempo_user = (temp.tms_utime+temp.tms_cutime) / (CLK_TCK*60.0) ;
main.c:         tiempo_sys = (temp.tms_stime+temp.tms_cstime) / (CLK_TCK*60.0) ;
main.c:
main.c: fprintf(stdout,"Tiempos (en minutos)\n");
main.c: fprintf(stdout,"User: %10.3f Sys: %10.3f\n",tiempo_user, tiempo_sys);
main.c: fprintf(stdout,"\n");
main.c: fprintf(stdout,"################################################################");
main.c: fprintf(stdout,"\n");
main.c:   fprintf(stdout,"          The End!!!!!\n");
main.c: fprintf(stdout,"\n");
main.c: fprintf(stdout,"################################################################");
main.c: fprintf(stdout,"\n");
main.c:           return;
main.c:   }
main.c:         }
main.c:
main.c: return;
main.c: }
```

```
models.c: #include <math.h>
models.c: #include <stdio.h>
models.c: #include "constantes.h"
models.c: #include "search.h"
models.c:
models.c: float jsim(float s2, float w, float tm)
models.c: {
models.c:
models.c:      float jresult, wp;
models.c:
models.c:      wp = w*2.0*PI ;
models.c:
models.c:      jresult = 2.0/5.0 *(s2*tm/(1.0+wp*wp*tm*tm));
models.c:
models.c:      return jresult;
models.c: }
models.c:
models.c: float jext(float sf2, float ss2, float w, float tm, float te)
models.c: {
models.c:
models.c:      float s2, tsp, jresult, wp;
models.c:
models.c:      s2 = sf2*ss2;
models.c:      tsp = te*tm/(te+tm);
models.c:      wp = w*2.0*PI ;
models.c:
models.c:      jresult = 2.0/5.0 *(s2*tm/(1.0+wp*wp*tm*tm)+sf2*(1.0-ss2)*tsp/(1.0+wp*wp*tsp*tsp));
models.c:
models.c:      return jresult;
models.c: }
models.c:
models.c: float j(float s2, float w, float tm, float te)
models.c: {
models.c:
models.c:      float tsp, jresult, wp;
models.c:
models.c:      tsp = te*tm/(te+tm);
models.c:      wp = w*2.0*PI ;
models.c:
models.c:      jresult = 2.0/5.0 *(s2*tm/(1.0+wp*wp*tm*tm)+(1.0-s2)*tsp/(1.0+wp*wp*tsp*tsp));
models.c:
models.c:      return jresult;
models.c: }
models.c:
models.c: struct para model1(float s2, float tm, float wh)
models.c: {
models.c:      struct para result;
models.c:      float wn,wnp;
models.c:
models.c:      wn = GN/GH * wh;
models.c:      wnp = wn*2.0*PI ;
models.c:
models.c:      result.R1 = (DNH*DNH/4.0)*(jsim(s2,wh-wn,tm)+3.0*jsim(s2,wn,tm)+6.0*jsim(s2,wh+wn,tm))+
```

```
models.c:                 (wnp*wnp/3.0)*1000*DELTA*DELTA*jsim(s2,wn,tm);
models.c:
models.c:      result.R2 = (DNH*DNH/8.0)*(4.0*jsim(s2,0.0,tm)+jsim(s2,wh-wn,tm)+
models.c:                 3.0*jsim(s2,wn,tm)+6.0*jsim(s2,wh+wn,tm)+6.0*jsim(s2,wh,tm))+
models.c:                 (wnp*wnp/18.0)*1000*DELTA*DELTA*(4.0*jsim(s2,0.0,tm)+3.0*jsim(s2,wn,tm));
models.c:
models.c:      result.HNOE = 1.0 + (DNH*DNH*(GH/GN)/(4.0*result.R1))*(6.0*jsim(s2,wh+wn,tm)-
models.c:                 jsim(s2,wh-wn,tm));
models.c:
models.c:
models.c:      return result;
models.c: }
models.c:
models.c: struct para model2(float te, float s2, float tm, float wh)
models.c: {
models.c:      struct para result;
models.c:      float wn,wnp;
models.c:
models.c:      wn = GN/GH * wh;
models.c:      wnp = wn*2.0*PI ;
models.c:
models.c:      result.R1 = (DNH*DNH/4.0)*(j(s2,wh-wn,tm,te)+3.0*j(s2,wn,tm,te)+6.0*j(s2,wh+wn,tm,te))+
models.c:                 (wnp*wnp/3.0)*1000*DELTA*DELTA*j(s2,wn,tm,te);
models.c:
models.c:      result.R2 = (DNH*DNH/8.0)*(4.0*j(s2,0.0,tm,te)+j(s2,wh-wn,tm,te)+
models.c:                 3.0*j(s2,wn,tm,te)+6.0*j(s2,wh+wn,tm,te)+6.0*j(s2,wh,tm,te))+
models.c:                 (wnp*wnp/18.0)*1000*DELTA*DELTA*(4.0*j(s2,0.0,tm,te)+3.0*j(s2,wn,tm,te));
models.c:
models.c:      result.HNOE = 1.0 + (DNH*DNH*(GH/GN)/(4.0*result.R1))*(6.0*j(s2,wh+wn,tm,te)-
models.c:                 j(s2,wh-wn,tm,te));
models.c:
models.c:
models.c:      return result;
models.c: }
models.c:
models.c: struct para model3(float rex, float s2, float tm, float wh)
models.c: {
models.c:      struct para result;
models.c:      float wn,wnp;
models.c:
models.c:      wn = GN/GH * wh;
models.c:      wnp = wn*2.0*PI ;
models.c:
models.c:      result.R1 = (DNH*DNH/4.0)*(jsim(s2,wh-wn,tm)+3.0*jsim(s2,wn,tm)+6.0*jsim(s2,wh+wn,tm))+
models.c:                 (wnp*wnp/3.0)*1000*DELTA*DELTA*jsim(s2,wn,tm);
models.c:
models.c:      result.R2 = (DNH*DNH/8.0)*(4.0*jsim(s2,0.0,tm)+jsim(s2,wh-wn,tm)+
models.c:                 3.0*jsim(s2,wn,tm)+6.0*jsim(s2,wh+wn,tm)+6.0*jsim(s2,wh,tm))+
models.c:                 (wnp*wnp/18.0)*1000*DELTA*DELTA*(4.0*jsim(s2,0.0,tm)+3.0*jsim(s2,wn,tm))+rex;
models.c:
models.c:      result.HNOE = 1.0 + (DNH*DNH*(GH/GN)/(4.0*result.R1))*(6.0*jsim(s2,wh+wn,tm)-
models.c:                 jsim(s2,wh-wn,tm));
models.c:
models.c:
models.c:      return result;
models.c: }
models.c:
models.c:
models.c:
models.c:
models.c: struct para model4(float te, float s2, float rex, float tm, float wh)
models.c: {
models.c:      struct para result;
models.c:      float wn,wnp;
models.c:
models.c:      wn = GN/GH * wh;
models.c:      wnp = wn*2.0*PI ;
models.c:
models.c:      result.R1 = (DNH*DNH/4.0)*(j(s2,wh-wn,tm,te)+3.0*j(s2,wn,tm,te)+6.0*j(s2,wh+wn,tm,te))+
models.c:                 (wnp*wnp/3.0)*1000*DELTA*DELTA*j(s2,wn,tm,te);
models.c:
models.c:      result.R2 = (DNH*DNH/8.0)*(4.0*j(s2,0.0,tm,te)+j(s2,wh-wn,tm,te)+
models.c:                 3.0*j(s2,wn,tm,te)+6.0*j(s2,wh+wn,tm,te)+6.0*j(s2,wh,tm,te))+
models.c:                 (wnp*wnp/18.0)*1000*DELTA*DELTA*(4.0*j(s2,0.0,tm,te)+3.0*j(s2,wn,tm,te))+rex;
models.c:
models.c:      result.HNOE = 1.0 + (DNH*DNH*(GH/GN)/(4.0*result.R1))*(6.0*j(s2,wh+wn,tm,te)-
```

```
models.c:                 j(s2,wh-wn,tm,te));
models.c:
models.c:
models.c:      return result;
models.c: }
models.c:
models.c:
models.c: struct para model5(float te, float ss2, float sf2, float tm, float wh)
models.c: {
models.c:      struct para result;
models.c:      float wn,wnp;
models.c:
models.c:      wn = GN/GH * wh;
models.c:      wnp = wn*2.0*PI ;
models.c:
models.c:      result.R1 = (DNH*DNH/4.0)*(jext(sf2,ss2,wh-wn,tm,te)+3.0*jext(sf2,ss2,wn,tm,te)+6.0*jext(sf2,ss2,wh+wn,tm
,te))+
models.c:                  (wnp*wnp/3.0)*1000*DELTA*DELTA*jext(sf2,ss2,wn,tm,te);
models.c:
models.c:      result.R2 = (DNH*DNH/8.0)*(4.0*jext(sf2,ss2,0.0,tm,te)+jext(sf2,ss2,wh-wn,tm,te)+
models.c:                  3.0*jext(sf2,ss2,wn,tm,te)+6.0*jext(sf2,ss2,wh+wn,tm,te)+6.0*jext(sf2,ss2,wh,tm,te))+
models.c:                  (wnp*wnp/18.0)*1000*DELTA*DELTA*(4.0*jext(sf2,ss2,0.0,tm,te)+3.0*jext(sf2,ss2,wn,tm,te));
models.c:
models.c:      result.HNOE = 1.0 + (DNH*DNH*(GH/GN)/(4.0*result.R1))*(6.0*jext(sf2,ss2,wh+wn,tm,te)-
models.c:                  jext(sf2,ss2,wh-wn,tm,te));
models.c:
models.c:
models.c:      return result;
models.c: }
models.c:
models.c:
models.c:
models.c: struct para model6(float te, float ss2, float sf2, float rex, float tm, float wh)
models.c: {
models.c:      struct para result;
models.c:      float wn,wnp;
models.c:
models.c:      wn = GN/GH * wh;
models.c:      wnp = wn*2.0*PI ;
models.c:
models.c:      result.R1 = (DNH*DNH/4.0)*(jext(sf2,ss2,wh-wn,tm,te)+3.0*jext(sf2,ss2,wn,tm,te)+6.0*jext(sf2,ss2,wh+wn,tm
,te))+
models.c:                  (wnp*wnp/3.0)*1000*DELTA*DELTA*jext(sf2,ss2,wn,tm,te);
models.c:
models.c:      result.R2 = (DNH*DNH/8.0)*(4.0*jext(sf2,ss2,0.0,tm,te)+jext(sf2,ss2,wh-wn,tm,te)+
models.c:                  3.0*jext(sf2,ss2,wn,tm,te)+6.0*jext(sf2,ss2,wh+wn,tm,te)+6.0*jext(sf2,ss2,wh,tm,te))+
models.c:                  (wnp*wnp/18.0)*1000*DELTA*DELTA*(4.0*jext(sf2,ss2,0.0,tm,te)+3.0*jext(sf2,ss2,wn,tm,te))+rex;
models.c:
models.c:      result.HNOE = 1.0 + (DNH*DNH*(GH/GN)/(4.0*result.R1))*(6.0*jext(sf2,ss2,wh+wn,tm,te)-
models.c:                  jext(sf2,ss2,wh-wn,tm,te));
models.c:
models.c:
models.c:      return result;
models.c: }
```

```
simul.c: #include <math.h>
simul.c: #include <stdio.h>
simul.c: #include <string.h>
simul.c: /*
simul.c: #include "search.h"
simul.c: #include "functions.h"
simul.c: #include "string.h"
simul.c:
simul.c: */
simul.c:
simul.c: #include "global.h"
simul.c: #include "constantes.h"
simul.c:
simul.c: void simul(float tm, float w, FILE *fpinp, char output[MAXLINE] )
simul.c: {
simul.c:
simul.c: FILE *fpout;
simul.c: char line[MAXLINE];
```

```
simul.c: char resname[MAXRES][10];
simul.c: int mod;
simul.c: struct para parateor;
simul.c:
simul.c:         float s2,te,ss2,sf2,rex;
simul.c:
simul.c:         fpout = fopen(output, "w");
simul.c:
simul.c:         while(getline(fpinp,line,sizeof(line)) > 0)
simul.c: {
simul.c: sscanf(line, "%s %d %f %f %f %f %f \n", &resname, &mod, &s2, &te, &ss2, &sf2, &rex);
simul.c:
simul.c:    fprintf(stdout,"SIMUL> Residuo %s\n",resname);
simul.c:
simul.c:            switch(mod)
simul.c:                 {
simul.c:                    case 2:
simul.c:
simul.c:                           parateor=model2(te,s2,tm,w);
simul.c:                           break;
simul.c:
simul.c:                    case 3:
simul.c:                           parateor=model3(rex,s2,tm,w);
simul.c:                           break;
simul.c:                    case 4:
simul.c:                           parateor=model4(te, s2,rex, tm, w);
simul.c:                           break;
simul.c:                    case 5:
simul.c:                           parateor=model5(te, ss2, sf2, tm, w);
simul.c:                           break;
simul.c:                    case 6:
simul.c:                           parateor=model6(te,ss2, sf2,rex,tm,w);
simul.c:                           break;
simul.c:                           }
simul.c:    fprintf(stdout,"SIMUL> %f %f %f \n",parateor.R1, parateor.R2, parateor.HNOE);
simul.c:    fprintf(fpout,"%s %6.3f %6.3f %6.3f %6.3f %6.3f %6.3f \n",resname,parateor.R1,absmio(parateor.R1*0.01), par
ateor.R2,absmio(parateor.R2*0.01), parateor.HNOE,absmio(parateor.HNOE*0.01));
simul.c:    fprintf(stdout,"SIMUL> \n");
simul.c:    }
simul.c:
simul.c:
simul.c:         fclose(fpout);
simul.c: return;
simul.c: }
```

Apéndice B

Ejemplos de ficheros de entrada y salida para SEARCHEL

Se expone a continuación un ejemplo de fichero de entrada, fichero de comandos y fichero de salida reducido del programa SEARCHEL.

B.1 Fichero de comandos: bpti_m.com

```
calctm
bpti_m.inp 0.5 0.00
calctree
bpti_m.inp 0.5 0.01 -1 -1 bpti_m
init
generoso
calctree
nosolution_0.01 0.5 0.02 -1 -1 bpti_m_nosolution
end
```

B.2 Fichero de datos: bpti_m.inp

```
ASP3 2.71 0.06 4.22 0.29 0.55 0.02
PHE4 2.79 0.19 4.25 0.30 0.56 0.07
CYS5 3.04 0.19 4.71 0.05 0.60 0.01
LEU6 2.08 0.01 4.04 0.60 0.67 0.18
GLU7 2.74 0.04 4.25 0.22 0.54 0.03
TYR10 2.59 0.13 4.03 0.14 0.54 0.05
THR11 2.77 0.12 4.36 0.05 0.55 0.02
GLY12 2.79 0.07 4.50 0.22 0.55 0.01
CYS14 2.85 0.15 5.70 0.31 0.58 0.01
ALA16 2.57 0.07 4.23 0.12 0.57 0.01
ARG17 2.49 0.05 3.92 0.27 0.51 0.01
ILE18 2.75 0.06 4.31 0.37 0.55 0.03
ILE19 2.82 0.01 4.08 0.42 0.61 0.12
ARG20 2.71 0.07 4.25 0.24 0.58 0.02
TYR21 2.74 0.06 4.41 0.25 0.53 0.03
PHE22 2.70 0.24 4.30 0.26 0.53 0.03
TYR23 2.80 0.03 4.33 0.34 0.58 0.01
ALA25 2.69 0.13 4.40 0.22 0.60 0.03
LYS26 2.65 0.09 4.17 0.19 0.65 0.13
ALA27 2.51 0.16 4.18 0.18 0.57 0.01
GLY28 2.37 0.15 4.15 0.26 0.58 0.01
LEU29 2.12 0.29 4.14 0.13 0.57 0.19
```

```
GLN31 2.71 0.05 4.36 0.25 0.56 0.01
THR32 2.56 0.04 3.97 0.44 0.55 0.04
PHE33 2.89 0.02 4.40 0.21 0.57 0.03
TYR35 2.71 0.10 4.27 0.29 0.53 0.07
GLY36 2.80 0.08 4.82 0.32 0.57 0.01
GLY37 3.03 0.18 5.14 0.05 0.59 0.05
CYS38 2.87 0.07 5.58 0.11 0.53 0.04
ARG39 2.74 0.05 7.35 0.04 0.53 0.04
ARG42 2.46 0.09 4.14 0.15 0.52 0.04
ASN43 2.71 0.02 4.39 0.32 0.56 0.01
ASN44 2.68 0.08 4.23 0.27 0.56 0.03
PHE45 2.73 0.06 4.13 0.19 0.52 0.01
LYS46 4.18 2.97 4.70 0.08 0.61 0.05
SER47 2.73 0.17 4.25 0.20 0.56 0.01
ALA48 2.93 0.12 4.40 0.19 0.55 0.01
GLU49 2.61 0.08 4.39 0.20 0.59 0.02
ASP50 2.83 0.09 4.58 0.10 0.62 0.02
CYS51 2.90 0.18 4.41 0.12 0.55 0.02
MET52 2.82 0.24 4.55 0.29 0.56 0.02
ARG53 2.85 0.12 4.62 0.16 0.61 0.01
THR54 2.70 0.29 4.50 0.10 0.55 0.02
CYS55 2.65 0.02 4.20 0.31 0.56 0.04
GLY56 2.75 0.04 4.20 0.36 0.52 0.01
GLY57 2.11 0.11 2.78 0.17 0.21 0.04
ALA58 1.38 0.15 1.76 0.23 -0.29 0.04
```

B.3 Fichero de Salida Reducido: bpti_m.abs

```
SCHSOL> Residuo 1 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ASP3 > Percent  0.000
SCHSOL | ASP3 > Test 0
SCHSOL>
SCHSOL> Residuo 1 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ASP3 > s2  0.865  0.840  0.890
SCHSOL | ASP3 > te  0.035  0.010  0.060
SCHSOL>
SCHSOL | ASP3 > Percent  0.897
SCHSOL | ASP3 > Test 1
SCHSOL>
SCHSOL> Residuo 2 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | PHE4 > s2  0.885  0.840  0.930
SCHSOL>
SCHSOL | PHE4 > Percent 52.808
SCHSOL | PHE4 > Test 1
SCHSOL>
SCHSOL> Residuo 3 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS5 > s2  0.940  0.930  0.950
SCHSOL>
SCHSOL | CYS5 > Percent  6.423
SCHSOL | CYS5 > Test 1
SCHSOL>
SCHSOL> Residuo 4 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | LEU6 > Percent  0.000
SCHSOL | LEU6 > Test 0
SCHSOL>
SCHSOL> Residuo 4 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | LEU6 > s2  0.705  0.700  0.710
SCHSOL | LEU6 > te -0.060 -0.070 -0.050   *******
SCHSOL>
SCHSOL | LEU6 > Percent  0.032
SCHSOL | LEU6 > Test 1
SCHSOL>
```

```
SCHSOL> Residuo 5 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLU7 > Percent  0.000
SCHSOL | GLU7 > Test 0
SCHSOL>
SCHSOL> Residuo 5 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | GLU7 > s2  0.870  0.850  0.890
SCHSOL | GLU7 > te  0.045  0.010  0.080
SCHSOL>
SCHSOL | GLU7 > Percent  0.977
SCHSOL | GLU7 > Test 1
SCHSOL>
SCHSOL> Residuo 6 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | TYR10 > s2  0.830  0.800  0.860
SCHSOL>
SCHSOL | TYR10 > Percent 17.552
SCHSOL | TYR10 > Test 1
SCHSOL>
SCHSOL> Residuo 7 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | THR11 > Percent  0.000
SCHSOL | THR11 > Test 0
SCHSOL>
SCHSOL> Residuo 7 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | THR11 > s2  0.875  0.850  0.900
SCHSOL | THR11 > te  0.035  0.010  0.060
SCHSOL>
SCHSOL | THR11 > Percent  0.512
SCHSOL | THR11 > Test 1
SCHSOL>
SCHSOL> Residuo 8 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY12 > Percent  0.000
SCHSOL | GLY12 > Test 0
SCHSOL>
SCHSOL> Residuo 8 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY12 > s2  0.895  0.870  0.920
SCHSOL | GLY12 > te  0.045  0.020  0.070
SCHSOL>
SCHSOL | GLY12 > Percent  0.656
SCHSOL | GLY12 > Test 1
SCHSOL>
SCHSOL> Residuo 9 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS14 > Percent  0.000
SCHSOL | CYS14 > Test 0
SCHSOL>
SCHSOL> Residuo 9 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS14 > Percent  0.000
SCHSOL | CYS14 > Test 0
SCHSOL>
SCHSOL> Residuo 9 Modelo 3
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS14 > s2  0.920  0.880  0.960
SCHSOL | CYS14 > rex  1.200  0.700  1.700
SCHSOL>
SCHSOL | CYS14 > Percent  1.814
SCHSOL | CYS14 > Test 1
SCHSOL>
SCHSOL> Residuo 10 Modelo 1
SCHSOL> ============================
```

```
SCHSOL>
SCHSOL | ALA16 > s2  0.850  0.850  0.850
SCHSOL>
SCHSOL | ALA16 > Percent  1.593
SCHSOL | ALA16 > Test 1
SCHSOL>
SCHSOL> Residuo 11 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG17 > Percent  0.000
SCHSOL | ARG17 > Test 0
SCHSOL>
SCHSOL> Residuo 11 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG17 > s2  0.785  0.770  0.800
SCHSOL | ARG17 > te  0.040  0.030  0.050
SCHSOL>
SCHSOL | ARG17 > Percent  0.224
SCHSOL | ARG17 > Test 1
SCHSOL>
SCHSOL> Residuo 12 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ILE18 > s2  0.885  0.870  0.900
SCHSOL>
SCHSOL | ILE18 > Percent  6.373
SCHSOL | ILE18 > Test 1
SCHSOL>
SCHSOL> Residuo 13 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ILE19 > s2  0.910  0.900  0.920
SCHSOL>
SCHSOL | ILE19 > Percent  6.388
SCHSOL | ILE19 > Test 1
SCHSOL>
SCHSOL> Residuo 14 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG20 > s2  0.870  0.850  0.890
SCHSOL>
SCHSOL | ARG20 > Percent 28.823
SCHSOL | ARG20 > Test 1
SCHSOL>
SCHSOL> Residuo 15 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | TYR21 > Percent  0.000
SCHSOL | TYR21 > Test 0
SCHSOL>
SCHSOL> Residuo 15 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | TYR21 > s2  0.875  0.850  0.900
SCHSOL | TYR21 > te  0.060  0.020  0.100
SCHSOL>
SCHSOL | TYR21 > Percent  1.537
SCHSOL | TYR21 > Test 1
SCHSOL>
SCHSOL> Residuo 16 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | PHE22 > Percent  0.000
SCHSOL | PHE22 > Test 0
SCHSOL>
SCHSOL> Residuo 16 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | PHE22 > s2  0.865  0.800  0.930
SCHSOL | PHE22 > te  0.085  0.010  0.160
SCHSOL>
SCHSOL | PHE22 > Percent  4.224
SCHSOL | PHE22 > Test 1
SCHSOL>
SCHSOL> Residuo 17 Modelo 1
```

```
SCHSOL> ============================
SCHSOL>
SCHSOL | TYR23 > s2  0.905  0.900  0.910
SCHSOL>
SCHSOL | TYR23 > Percent  6.383
SCHSOL | TYR23 > Test 1
SCHSOL>
SCHSOL> Residuo 18 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA25 > s2  0.870  0.830  0.910
SCHSOL>
SCHSOL | ALA25 > Percent 38.442
SCHSOL | ALA25 > Test 1
SCHSOL>
SCHSOL> Residuo 19 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | LYS26 > s2  0.855  0.830  0.880
SCHSOL>
SCHSOL | LYS26 > Percent 33.599
SCHSOL | LYS26 > Test 1
SCHSOL>
SCHSOL> Residuo 20 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA27 > s2  0.845  0.830  0.860
SCHSOL>
SCHSOL | ALA27 > Percent  6.373
SCHSOL | ALA27 > Test 1
SCHSOL>
SCHSOL> Residuo 21 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY28 > s2  0.805  0.800  0.810
SCHSOL>
SCHSOL | GLY28 > Percent  4.790
SCHSOL | GLY28 > Test 1
SCHSOL>
SCHSOL> Residuo 22 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | LEU29 > Percent  0.000
SCHSOL | LEU29 > Test 0
SCHSOL>
SCHSOL> Residuo 22 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | LEU29 > Percent  0.000
SCHSOL | LEU29 > Test 0
SCHSOL>
SCHSOL> Residuo 22 Modelo 3
SCHSOL> ============================
SCHSOL>
SCHSOL | LEU29 > s2  0.680  0.590  0.770
SCHSOL | LEU29 > rex  0.750  0.200  1.300
SCHSOL>
SCHSOL | LEU29 > Percent  3.118
SCHSOL | LEU29 > Test 1
SCHSOL>
SCHSOL> Residuo 23 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLN31 > Percent  0.000
SCHSOL | GLN31 > Test 0
SCHSOL>
SCHSOL> Residuo 23 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | GLN31 > s2  0.870  0.850  0.890
SCHSOL | GLN31 > te  0.025  0.010  0.040
SCHSOL>
SCHSOL | GLN31 > Percent  0.336
SCHSOL | GLN31 > Test 1
SCHSOL>
SCHSOL> Residuo 24 Modelo 1
```

```
SCHSOL> ============================
SCHSOL>
SCHSOL | THR32 > s2  0.830  0.820  0.840
SCHSOL>
SCHSOL | THR32 > Percent  7.977
SCHSOL | THR32 > Test 1
SCHSOL>
SCHSOL> Residuo 25 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | PHE33 > s2  0.935  0.930  0.940
SCHSOL>
SCHSOL | PHE33 > Percent  4.785
SCHSOL | PHE33 > Test 1
SCHSOL>
SCHSOL> Residuo 26 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | TYR35 > s2  0.870  0.840  0.900
SCHSOL>
SCHSOL | TYR35 > Percent 41.629
SCHSOL | TYR35 > Test 1
SCHSOL>
SCHSOL> Residuo 27 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY36 > s2  0.930  0.930  0.930
SCHSOL>
SCHSOL | GLY36 > Percent  1.593
SCHSOL | GLY36 > Test 1
SCHSOL>
SCHSOL> Residuo 28 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY37 > Percent  0.000
SCHSOL | GLY37 > Test 0
SCHSOL>
SCHSOL> Residuo 28 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY37 > Percent  0.000
SCHSOL | GLY37 > Test 0
SCHSOL>
SCHSOL> Residuo 28 Modelo 3
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY37 > s2  0.960  0.920  1.000
SCHSOL | GLY37 > rex  0.350  0.100  0.600
SCHSOL>
SCHSOL | GLY37 > Percent  0.618
SCHSOL | GLY37 > Test 1
SCHSOL>
SCHSOL> Residuo 29 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS38 > Percent  0.000
SCHSOL | CYS38 > Test 0
SCHSOL>
SCHSOL> Residuo 29 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS38 > Percent  0.000
SCHSOL | CYS38 > Test 0
SCHSOL>
SCHSOL> Residuo 29 Modelo 3
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS38 > Percent  0.000
SCHSOL | CYS38 > Test 0
SCHSOL>
SCHSOL> Residuo 29 Modelo 4
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS38 > s2  0.910  0.880  0.940
SCHSOL | CYS38 > te  0.120  0.010  0.230
SCHSOL | CYS38 > rex  1.050  0.800  1.300
```

```
SCHSOL>
SCHSOL | CYS38 > Percent  0.032
SCHSOL | CYS38 > Test 1
SCHSOL>
SCHSOL> Residuo 30 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG39 > Percent  0.000
SCHSOL | ARG39 > Test 0
SCHSOL>
SCHSOL> Residuo 30 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG39 > Percent  0.000
SCHSOL | ARG39 > Test 0
SCHSOL>
SCHSOL> Residuo 30 Modelo 3
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG39 > Percent  0.000
SCHSOL | ARG39 > Test 0
SCHSOL>
SCHSOL> Residuo 30 Modelo 4
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG39 > s2  0.875  0.850  0.900
SCHSOL | ARG39 > te  0.060  0.010  0.110
SCHSOL | ARG39 > rex  3.000  2.800  3.200
SCHSOL>
SCHSOL | ARG39 > Percent  0.008
SCHSOL | ARG39 > Test 1
SCHSOL>
SCHSOL> Residuo 31 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG42 > Percent  0.000
SCHSOL | ARG42 > Test 0
SCHSOL>
SCHSOL> Residuo 31 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG42 > s2  0.795  0.780  0.810
SCHSOL | ARG42 > te  0.040  0.020  0.060
SCHSOL>
SCHSOL | ARG42 > Percent  0.289
SCHSOL | ARG42 > Test 1
SCHSOL>
SCHSOL> Residuo 32 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ASN43 > Percent  0.000
SCHSOL | ASN43 > Test 0
SCHSOL>
SCHSOL> Residuo 32 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ASN43 > s2  0.870  0.860  0.880
SCHSOL | ASN43 > te  0.025  0.010  0.040
SCHSOL>
SCHSOL | ASN43 > Percent  0.208
SCHSOL | ASN43 > Test 1
SCHSOL>
SCHSOL> Residuo 33 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ASN44 > s2  0.865  0.840  0.890
SCHSOL>
SCHSOL | ASN44 > Percent 15.959
SCHSOL | ASN44 > Test 1
SCHSOL>
SCHSOL> Residuo 34 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | PHE45 > Percent  0.000
SCHSOL | PHE45 > Test 0
SCHSOL>
```

```
SCHSOL> Residuo 34 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | PHE45 > s2  0.865  0.850  0.880
SCHSOL | PHE45 > te  0.050  0.040  0.060
SCHSOL>
SCHSOL | PHE45 > Percent  0.208
SCHSOL | PHE45 > Test 1
SCHSOL>
SCHSOL> Residuo 35 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | LYS46 > s2  0.950  0.920  0.980
SCHSOL>
SCHSOL | LYS46 > Percent 20.812
SCHSOL | LYS46 > Test 1
SCHSOL>
SCHSOL> Residuo 36 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | SER47 > Percent  0.000
SCHSOL | SER47 > Test 0
SCHSOL>
SCHSOL> Residuo 36 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | SER47 > s2  0.865  0.820  0.910
SCHSOL | SER47 > te  0.025  0.010  0.040
SCHSOL>
SCHSOL | SER47 > Percent  0.816
SCHSOL | SER47 > Test 1
SCHSOL>
SCHSOL> Residuo 37 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA48 > Percent  0.000
SCHSOL | ALA48 > Test 0
SCHSOL>
SCHSOL> Residuo 37 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA48 > s2  0.920  0.900  0.940
SCHSOL | ALA48 > te  0.045  0.020  0.070
SCHSOL>
SCHSOL | ALA48 > Percent  0.543
SCHSOL | ALA48 > Test 1
SCHSOL>
SCHSOL> Residuo 38 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLU49 > s2  0.850  0.840  0.860
SCHSOL>
SCHSOL | GLU49 > Percent  9.629
SCHSOL | GLU49 > Test 1
SCHSOL>
SCHSOL> Residuo 39 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ASP50 > Percent  0.000
SCHSOL | ASP50 > Test 0
SCHSOL>
SCHSOL> Residuo 39 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ASP50 > s2  0.930  0.900  0.960
SCHSOL | ASP50 > te -0.255 -0.500 -0.010   *******
SCHSOL>
SCHSOL | ASP50 > Percent  2.318
SCHSOL | ASP50 > Test 0
SCHSOL>
SCHSOL> Residuo 40 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS51 > Percent  0.000
SCHSOL | CYS51 > Test 0
SCHSOL>
```

```
SCHSOL> Residuo 40 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS51 > s2  0.895  0.860  0.930
SCHSOL | CYS51 > te  0.040  0.010  0.070
SCHSOL>
SCHSOL | CYS51 > Percent  1.327
SCHSOL | CYS51 > Test 1
SCHSOL>
SCHSOL> Residuo 41 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | MET52 > s2  0.930  0.880  0.980
SCHSOL>
SCHSOL | MET52 > Percent 17.527
SCHSOL | MET52 > Test 1
SCHSOL>
SCHSOL> Residuo 42 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG53 > Percent  0.000
SCHSOL | ARG53 > Test 0
SCHSOL>
SCHSOL> Residuo 42 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ARG53 > s2  0.930  0.890  0.970
SCHSOL | ARG53 > te -0.255 -0.500 -0.010   *******
SCHSOL>
SCHSOL | ARG53 > Percent  1.934
SCHSOL | ARG53 > Test 0
SCHSOL>
SCHSOL> Residuo 43 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | THR54 > Percent  0.000
SCHSOL | THR54 > Test 0
SCHSOL>
SCHSOL> Residuo 43 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | THR54 > s2  0.905  0.870  0.940
SCHSOL | THR54 > te  0.055  0.010  0.100
SCHSOL>
SCHSOL | THR54 > Percent  1.584
SCHSOL | THR54 > Test 1
SCHSOL>
SCHSOL> Residuo 44 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | CYS55 > s2  0.855  0.850  0.860
SCHSOL>
SCHSOL | CYS55 > Percent 11.198
SCHSOL | CYS55 > Test 1
SCHSOL>
SCHSOL> Residuo 45 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY56 > Percent  0.000
SCHSOL | GLY56 > Test 0
SCHSOL>
SCHSOL> Residuo 45 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY56 > s2  0.875  0.860  0.890
SCHSOL | GLY56 > te  0.065  0.050  0.080
SCHSOL>
SCHSOL | GLY56 > Percent  0.336
SCHSOL | GLY56 > Test 1
SCHSOL>
SCHSOL> Residuo 46 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY57 > Percent  0.000
SCHSOL | GLY57 > Test 0
SCHSOL>
```

```
SCHSOL> Residuo 46 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY57 > Percent  0.000
SCHSOL | GLY57 > Test 0
SCHSOL>
SCHSOL> Residuo 46 Modelo 3
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY57 > Percent  0.000
SCHSOL | GLY57 > Test 0
SCHSOL>
SCHSOL> Residuo 46 Modelo 4
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY57 > Percent  0.000
SCHSOL | GLY57 > Test 0
SCHSOL>
SCHSOL> Residuo 46 Modelo 5
SCHSOL> ============================
SCHSOL>
SCHSOL | GLY57 > ss2  0.445  0.200  0.690
SCHSOL | GLY57 > sf2  0.795  0.740  0.850
SCHSOL | GLY57 > te  1.790  0.890  2.690
SCHSOL>
SCHSOL | GLY57 > Percent  3.306
SCHSOL | GLY57 > Test 1
SCHSOL>
SCHSOL> Residuo 47 Modelo 1
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA58 > Percent  0.000
SCHSOL | ALA58 > Test 0
SCHSOL>
SCHSOL> Residuo 47 Modelo 2
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA58 > Percent  0.000
SCHSOL | ALA58 > Test 0
SCHSOL>
SCHSOL> Residuo 47 Modelo 3
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA58 > Percent  0.000
SCHSOL | ALA58 > Test 0
SCHSOL>
SCHSOL> Residuo 47 Modelo 4
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA58 > Percent  0.000
SCHSOL | ALA58 > Test 0
SCHSOL>
SCHSOL> Residuo 47 Modelo 5
SCHSOL> ============================
SCHSOL>
SCHSOL | ALA58 > ss2  0.265  0.000  0.530
SCHSOL | ALA58 > sf2  0.750  0.540  0.960
SCHSOL | ALA58 > te  0.990  0.090  1.890
SCHSOL>
SCHSOL | ALA58 > Percent  0.385
SCHSOL | ALA58 > Test 0
SCHSOL>
```

Apéndice C

Código fuente del cálculo de distancias en HYPER

Se expone a continuación el codigo fuente del programa CALCDIST parte del programa HYPER como apéndice a la tesis. Se puede ver el nombre del fichero correspondiente al principio de cada línea:

```
def.h:#define PI_DIV_180 PI/180.0
def.h:#define MAXLINE 1000
def.h:#define MAXRES  20
def.h:/*
def.h:   Avoid Linux warnings about having PI already defined
def.h:   Added by RTT.
def.h:*/
def.h:#ifndef PI
def.h:#define PI   3.14159265358979323846
def.h:#endif
```

```
dist.h:#define MAXLINE 1000
dist.h:#define MAXRES 100
dist.h:#define PI_DIV_180 PI/180.0
dist.h:/*
dist.h:   Avoid Linux warnings about having PI already defined
dist.h:   Added by RTT.
dist.h:*/
dist.h:#ifndef PI
dist.h:#define PI   3.14159265358979323846
dist.h:#endif
dist.h:void    crea_transf(float *ptr_mat, float ang_a, float ang_b);
dist.h:void    transf_vec(float *ptr_vec, float *ptr_mat);
dist.h:float   modulo(float *ptr_vec);
dist.h:float   calcdist(int res1,char atom1[],int res2,char atom2[]);
dist.h:void    unidad(float *ptr_mat);
dist.h:void    iguala(float *ptr_mat1,float *ptr_mat2);
dist.h:int     getline(FILE *fpinp,char s[],int lim);
dist.h:void    sumvect(float *ptr_vect1,float *ptr_vect2,float *ptr_vects);
dist.h:void    mult_mat(float *ptr_mat1, float *ptr_mat2,float *ptr_matr);
dist.h:void    mult_vect(float *ptr_vect1, float *ptr_vect2,float *ptr_vect);
dist.h:void    calcvect(FILE *fpinp);
dist.h:void    unitario(float *ptr_vect1, float *ptr_vect2);
```

```
vect.h:float   vect[MAXRES][MAXRES][14][4];
vect.h:float   vectres[MAXRES][MAXRES][4];
```

```
calcdist.c:#include <math.h>
calcdist.c:#include <string.h>
calcdist.c:#include <stdio.h>
calcdist.c:#include "dist.h"
calcdist.c:float calcdist(int res1,char atom1[5],int res2,char atom2[5])
calcdist.c:{
calcdist.c:    extern float   vect[MAXRES][MAXRES][14][4];
calcdist.c:    extern float   vectres[MAXRES][MAXRES][4];
calcdist.c:    float   vectdif[4];
calcdist.c:    float   vectORIG[4],vectDEST[4],vectdist[4];
calcdist.c:    float *ptr_vect;
calcdist.c:    float distancia;
calcdist.c:    char attmp[5];
calcdist.c:    int c, test, restmp;
calcdist.c:/* Cambio de orden si residuo 1 mayor que 2 */
calcdist.c:        test=0;
calcdist.c:        if( res2 < res1 )
calcdist.c:        { restmp = res1;
calcdist.c:          res1 = res2;
calcdist.c:          res2 = restmp;
calcdist.c:          strcpy(attmp,atom1);
calcdist.c:          strcpy(atom1,atom2);
calcdist.c:          strcpy(atom2,attmp);
calcdist.c:          test =1;
calcdist.c:         }
calcdist.c:        if( res1 != res2 )
calcdist.c:        {
calcdist.c: vectdif[1] = vectres[res1][res2-1][1] - vectres[res1][res1][1];
calcdist.c: vectdif[2] = vectres[res1][res2-1][2] - vectres[res1][res1][2];
calcdist.c: vectdif[3] = vectres[res1][res2-1][3] - vectres[res1][res1][3];
calcdist.c:        }
calcdist.c:        else
calcdist.c:        {
calcdist.c: vectdif[1] = - vectres[res1][res1][1];
calcdist.c: vectdif[2] = - vectres[res1][res1][2];
calcdist.c: vectdif[3] = - vectres[res1][res1][3];
calcdist.c:        }
calcdist.c: c = '0';
calcdist.c: if ((strcmp(atom1,"HA")) == 0) c = '1';
calcdist.c: if ((strcmp(atom1,"HA1")) == 0) c = '2';
calcdist.c: if ((strcmp(atom1,"HA2")) == 0) c = '3';
calcdist.c: if ((strcmp(atom1,"HB1")) == 0) c = '4';
calcdist.c: if ((strcmp(atom1,"HB2")) == 0) c = '5';
calcdist.c: if ((strcmp(atom1,"HB3")) == 0) c = '6';
calcdist.c: if ((strcmp(atom1,"HN")) == 0) c = '7';
calcdist.c:     vectORIG[1]= 0.0;
calcdist.c:     vectORIG[2]= 0.0;
calcdist.c:     vectORIG[3]= 0.0;
calcdist.c: switch (c)
calcdist.c: {
calcdist.c: case '1':
calcdist.c:      vectORIG[1]= -vect[res1][res1][3][1] + vect[res1][res1][4][1] + vect[res1][res1][6][1];
calcdist.c:      vectORIG[2]= -vect[res1][res1][3][2] + vect[res1][res1][4][2] + vect[res1][res1][6][2];
calcdist.c:      vectORIG[3]= -vect[res1][res1][3][3] + vect[res1][res1][4][3] + vect[res1][res1][6][3];
calcdist.c:            break;
calcdist.c: case '2':
calcdist.c:      vectORIG[1]= -vect[res1][res1][3][1] + vect[res1][res1][4][1] + vect[res1][res1][6][1];
calcdist.c:      vectORIG[2]= -vect[res1][res1][3][2] + vect[res1][res1][4][2] + vect[res1][res1][6][2];
calcdist.c:      vectORIG[3]= -vect[res1][res1][3][3] + vect[res1][res1][4][3] + vect[res1][res1][6][3];
calcdist.c:            break;
calcdist.c: case '3':
calcdist.c:      vectORIG[1]= -vect[res1][res1][9][1] + vect[res1][res1][4][1] + vect[res1][res1][6][1];
calcdist.c:      vectORIG[2]= -vect[res1][res1][9][2] + vect[res1][res1][4][2] + vect[res1][res1][6][2];
calcdist.c:      vectORIG[3]= -vect[res1][res1][9][3] + vect[res1][res1][4][3] + vect[res1][res1][6][3];
calcdist.c:            break;
calcdist.c: case '4':
calcdist.c:      vectORIG[1]= -vect[res1][res1][7][1] - vect[res1][res1][5][1] + vect[res1][res1][4][1] + vect[res1][res
1][6][1];
calcdist.c:      vectORIG[2]= -vect[res1][res1][7][2] - vect[res1][res1][5][2] + vect[res1][res1][4][2] + vect[res1][res
1][6][2];
```

```
calcdist.c:      vectORIG[3]= -vect[res1][res1][7][3] - vect[res1][res1][5][3] + vect[res1][res1][4][3] + vect[res1][res
1][6][3];
calcdist.c:            break;
calcdist.c: case '5':
calcdist.c:      vectORIG[1]= -vect[res1][res1][8][1] - vect[res1][res1][5][1] + vect[res1][res1][4][1] + vect[res1][res
1][6][1];
calcdist.c:      vectORIG[2]= -vect[res1][res1][8][2] - vect[res1][res1][5][2] + vect[res1][res1][4][2] + vect[res1][res
1][6][2];
calcdist.c:      vectORIG[3]= -vect[res1][res1][8][3] - vect[res1][res1][5][3] + vect[res1][res1][4][3] + vect[res1][res
1][6][3];
calcdist.c:            break;
calcdist.c: case '6':
calcdist.c:      vectORIG[1]= -vect[res1][res1][10][1] - vect[res1][res1][5][1] + vect[res1][res1][4][1] + vect[res1][re
s1][6][1];
calcdist.c:      vectORIG[2]= -vect[res1][res1][10][2] - vect[res1][res1][5][2] + vect[res1][res1][4][2] + vect[res1][re
s1][6][2];
calcdist.c:      vectORIG[3]= -vect[res1][res1][10][3] - vect[res1][res1][5][3] + vect[res1][res1][4][3] + vect[res1][re
s1][6][3];
calcdist.c:            break;
calcdist.c: case '7':
calcdist.c:      vectORIG[1]= vect[res1][res1][1][1] + vect[res1][res1][2][1] + vect[res1][res1][4][1] + vect[res1][res1
][6][1];
calcdist.c:      vectORIG[2]= vect[res1][res1][1][2] + vect[res1][res1][2][2] + vect[res1][res1][4][2] + vect[res1][res1
][6][2];
calcdist.c:      vectORIG[3]= vect[res1][res1][1][3] + vect[res1][res1][2][3] + vect[res1][res1][4][3] + vect[res1][res1
][6][3];
calcdist.c:            break;
calcdist.c: case '0':
calcdist.c:      printf("Unknown atom?");
calcdist.c:            return 0;
calcdist.c:      break;
calcdist.c: }
calcdist.c: c = '0';
calcdist.c: if ((strcmp(atom2,"HA")) == 0) c = '1';
calcdist.c: if ((strcmp(atom2,"HA1")) == 0) c = '2';
calcdist.c: if ((strcmp(atom2,"HA2")) == 0) c = '3';
calcdist.c: if ((strcmp(atom2,"HB1")) == 0) c = '4';
calcdist.c: if ((strcmp(atom2,"HB2")) == 0) c = '5';
calcdist.c: if ((strcmp(atom2,"HB3")) == 0) c = '6';
calcdist.c: if ((strcmp(atom2,"HN")) == 0) c = '7';
calcdist.c:     vectDEST[1]= 0.0;
calcdist.c:     vectDEST[2]= 0.0;
calcdist.c:     vectDEST[3]= 0.0;
calcdist.c: switch (c)
calcdist.c: {
calcdist.c: case '1':
calcdist.c:      vectDEST[1]= vect[res1][res2][3][1] + vect[res1][res2][2][1];
calcdist.c:      vectDEST[2]= vect[res1][res2][3][2] + vect[res1][res2][2][2];
calcdist.c:      vectDEST[3]= vect[res1][res2][3][3] + vect[res1][res2][2][3];
calcdist.c:            break;
calcdist.c: case '2':
calcdist.c:      vectDEST[1]= vect[res1][res2][3][1] + vect[res1][res2][2][1];
calcdist.c:      vectDEST[2]= vect[res1][res2][3][2] + vect[res1][res2][2][2];
calcdist.c:      vectDEST[3]= vect[res1][res2][3][3] + vect[res1][res2][2][3];
calcdist.c:            break;
calcdist.c: case '3':
calcdist.c:      vectDEST[1]= vect[res1][res2][9][1] + vect[res1][res2][2][1];
calcdist.c:      vectDEST[2]= vect[res1][res2][9][2] + vect[res1][res2][2][2];
calcdist.c:      vectDEST[3]= vect[res1][res2][9][3] + vect[res1][res2][2][3];
calcdist.c:            break;
calcdist.c: case '4':
calcdist.c:      vectDEST[1]= vect[res1][res2][7][1] + vect[res1][res2][2][1] + vect[res1][res2][5][1];
calcdist.c:      vectDEST[2]= vect[res1][res2][7][2] + vect[res1][res2][2][2] + vect[res1][res2][5][2];
calcdist.c:      vectDEST[3]= vect[res1][res2][7][3] + vect[res1][res2][2][3] + vect[res1][res2][5][3];
calcdist.c:            break;
calcdist.c: case '5':
calcdist.c:      vectDEST[1]= vect[res1][res2][8][1] + vect[res1][res2][2][1] + vect[res1][res2][5][1];
calcdist.c:      vectDEST[2]= vect[res1][res2][8][2] + vect[res1][res2][2][2] + vect[res1][res2][5][2];
calcdist.c:      vectDEST[3]= vect[res1][res2][8][3] + vect[res1][res2][2][3] + vect[res1][res2][5][3];
calcdist.c:            break;
calcdist.c: case '6':
calcdist.c:      vectDEST[1]= vect[res1][res2][10][1] + vect[res1][res2][2][1] + vect[res1][res2][5][1];
calcdist.c:      vectDEST[2]= vect[res1][res2][10][2] + vect[res1][res2][2][2] + vect[res1][res2][5][2];
calcdist.c:      vectDEST[3]= vect[res1][res2][10][3] + vect[res1][res2][2][3] + vect[res1][res2][5][3];
calcdist.c:            break;
calcdist.c: case '7':
```

```
calcdist.c:      vectDEST[1]= -vect[res1][res2][1][1];
calcdist.c:      vectDEST[2]= -vect[res1][res2][1][2];
calcdist.c:      vectDEST[3]= -vect[res1][res2][1][3];
calcdist.c:            break;
calcdist.c: case '0':
calcdist.c:      printf("Unknown atom?");
calcdist.c:            return 0;
calcdist.c:      break;
calcdist.c: }
calcdist.c: vectdist[1] = vectORIG[1] + vectdif[1] + vectDEST[1];
calcdist.c: vectdist[2] = vectORIG[2] + vectdif[2] + vectDEST[2];
calcdist.c: vectdist[3] = vectORIG[3] + vectdif[3] + vectDEST[3];
calcdist.c: ptr_vect = &vectdist[1];
calcdist.c: distancia= modulo(ptr_vect);
calcdist.c:       if( test == 1 )
calcdist.c:       { restmp = res1;
calcdist.c:         res1 = res2;
calcdist.c:         res2 = restmp;
calcdist.c:         strcpy(attmp,atom1);
calcdist.c:         strcpy(atom1,atom2);
calcdist.c:         strcpy(atom2,attmp);
calcdist.c:        }
calcdist.c:   return distancia;
calcdist.c:}
```

```
calcvect.c:#include <math.h>
calcvect.c:#include <string.h>
calcvect.c:#include <stdio.h>
calcvect.c:#include "dist.h"
calcvect.c:void calcvect(FILE *fpinp)
calcvect.c: {
calcvect.c:    char    line[MAXLINE];
calcvect.c:    float   transf[11][4][4];
calcvect.c:    extern float   vect[MAXRES][MAXRES][14][4];
calcvect.c:    extern float   vectres[MAXRES][MAXRES][4];
calcvect.c:    float   vectsys[MAXRES][4][4];
calcvect.c:    float   *ptr_transf,*ptr_vect;
calcvect.c:    float   *ptr_vect1,*ptr_vect2;
calcvect.c:    float   phi[MAXRES], psi[MAXRES], chi[MAXRES];
calcvect.c:    float   vcttmp1, vcttmp2, vcttmp3;
calcvect.c:
calcvect.c:    int rescont, residx;
calcvect.c:    int nres,i;
calcvect.c:    float   dn_ct,
calcvect.c:            dn_c,
calcvect.c:            dn_h,
calcvect.c:            dct_ct,
calcvect.c:            dct_c,
calcvect.c:            dct_hc;
calcvect.c:    float   angh_n_ct,
calcvect.c:            angn_ct_hc,
calcvect.c:            angn_ct_ct,
calcvect.c:            angn_ct_c,
calcvect.c:            angct_ct_hc,
calcvect.c:            angct_c_n,
calcvect.c:            angc_n_h,
calcvect.c:            angc_n_ct;
calcvect.c:
calcvect.c:    float   desf_phi1,
calcvect.c:            desf_phi2,
calcvect.c:            desf_chi1,
calcvect.c:            desf_chi2;
calcvect.c:    dn_ct  = 1.459  ;
calcvect.c:    dn_c   = 1.335  ;
calcvect.c:    dn_h   = 1.010  ;
calcvect.c:    dct_ct = 1.536  ;
calcvect.c:    dct_c  = 1.522  ;
calcvect.c:    dct_hc = 1.110  ;
calcvect.c:    angh_n_ct   = 118.4;
calcvect.c:    angn_ct_hc  = 109.5;
calcvect.c:    angn_ct_ct  = 109.7;
calcvect.c:    angn_ct_c   = 110.1;
calcvect.c:    angct_ct_hc = 109.5;
```

```
calcvect.c:    angct_c_n   = 116.6;
calcvect.c:    angc_n_h    = 119.8;
calcvect.c:    angc_n_ct   = 121.8;
calcvect.c:
calcvect.c:    angh_n_ct   = PI - angh_n_ct   * PI_DIV_180 ;
calcvect.c:    angn_ct_hc  = PI - angn_ct_hc  * PI_DIV_180 ;
calcvect.c:    angn_ct_ct  = PI - angn_ct_ct  * PI_DIV_180 ;
calcvect.c:    angn_ct_c   = PI - angn_ct_c   * PI_DIV_180 ;
calcvect.c:    angct_ct_hc = PI - angct_ct_hc * PI_DIV_180 ;
calcvect.c:    angct_c_n   = PI - angct_c_n   * PI_DIV_180 ;
calcvect.c:    angc_n_h    = PI - angc_n_h    * PI_DIV_180 ;
calcvect.c:    angc_n_ct   = PI - angc_n_ct   * PI_DIV_180 ;
calcvect.c:  /*
calcvect.c:    desf_phi1 = -115.0 * PI_DIV_180 ;
calcvect.c:    desf_phi2 =  125.0 * PI_DIV_180 ;
calcvect.c:    desf_chi1 = -120.0 * PI_DIV_180 ;
calcvect.c:    desf_chi2 =  120.0 * PI_DIV_180 ;
calcvect.c:*/
calcvect.c:    desf_phi1 = -115 * PI_DIV_180 ;
calcvect.c:    desf_phi2 = 125 * PI_DIV_180 ;
calcvect.c:    desf_chi1 = -120.0 * PI_DIV_180 ;
calcvect.c:    desf_chi2 =  120.0 * PI_DIV_180 ;
calcvect.c:    rescont = 0;
calcvect.c:
calcvect.c:    getline(fpinp,line, sizeof(line));
calcvect.c:    sscanf(line,"%d",&nres);
calcvect.c:    printf("Numero de residuos: %d\n",nres);
calcvect.c:    printf("Leyendo phi psi chi...\n");
calcvect.c:while(++rescont <= nres)
calcvect.c:{
calcvect.c:    getline(fpinp,line,sizeof(line));
calcvect.c:    sscanf(line,"%f %f %f",&phi[rescont],&psi[rescont],&chi[rescont]);
calcvect.c:    printf("%f %f %f \n",phi[rescont],psi[rescont],chi[rescont]);
calcvect.c:    phi[rescont] = phi[rescont] * PI_DIV_180;
calcvect.c:    psi[rescont] = psi[rescont] * PI_DIV_180;
calcvect.c:    chi[rescont] = chi[rescont] * PI_DIV_180;
calcvect.c:}
calcvect.c:    rescont=0;
calcvect.c:
calcvect.c:    while(++rescont <= nres)
calcvect.c:{
calcvect.c:    vectres[rescont][rescont-1][1] = 0.0;
calcvect.c:    vectres[rescont][rescont-1][2] = 0.0;
calcvect.c:    vectres[rescont][rescont-1][3] = 0.0;
calcvect.c:    vectsys[rescont-1][1][1] = 1.0;
calcvect.c:    vectsys[rescont-1][1][2] = 0.0;
calcvect.c:    vectsys[rescont-1][1][3] = 0.0;
calcvect.c:    vectsys[rescont-1][2][1] = 0.0;
calcvect.c:    vectsys[rescont-1][2][2] = 1.0;
calcvect.c:    vectsys[rescont-1][2][3] = 0.0;
calcvect.c:    vectsys[rescont-1][3][1] = 0.0;
calcvect.c:    vectsys[rescont-1][3][2] = 0.0;
calcvect.c:    vectsys[rescont-1][3][3] = 1.0;
calcvect.c:    residx=rescont;
calcvect.c:    while(residx <= nres)
calcvect.c:{
calcvect.c:    ptr_transf=&transf[1][1][1];
calcvect.c:    crea_transf(ptr_transf, angn_ct_hc  , PI + desf_phi1 - phi[residx] );
calcvect.c:    ptr_transf=&transf[2][1][1];
calcvect.c:    crea_transf(ptr_transf, angn_ct_c   , PI - phi[residx]);
calcvect.c:    ptr_transf=&transf[3][1][1];
calcvect.c:    crea_transf(ptr_transf, angn_ct_ct  , PI + desf_phi2 - phi[residx] );
calcvect.c:    ptr_transf=&transf[4][1][1];
calcvect.c:    crea_transf(ptr_transf, angct_c_n   , PI - psi[residx]);
calcvect.c:    ptr_transf=&transf[5][1][1];
calcvect.c:    crea_transf(ptr_transf, angct_ct_hc , PI + desf_chi1 - chi[residx] );
calcvect.c:    ptr_transf=&transf[6][1][1];
calcvect.c:    crea_transf(ptr_transf, angct_ct_hc , PI + desf_chi2 - chi[residx] );
calcvect.c:    ptr_transf=&transf[7][1][1];
calcvect.c:    crea_transf(ptr_transf, angc_n_ct , 0.0 );
calcvect.c:    ptr_transf=&transf[8][1][1];
calcvect.c:    crea_transf(ptr_transf, angct_ct_hc , PI - chi[residx] );
calcvect.c:    ptr_transf=&transf[9][1][1];
calcvect.c:    crea_transf(ptr_transf, angc_n_h , PI );
calcvect.c:/*
calcvect.c:  Definicion y calculo de los vectores que representan cada enlace del
```

```
calcvect.c:  sistema. Las transformaciones necesarias se hacen mediante la funcion
calcvect.c:  transf
calcvect.c:  Definition and calculation of the vectors representing each bond in the
calcvect.c:  system. The transformations are carried out by function transf
calcvect.c:*/
calcvect.c:/* Creation of vector 1 */
calcvect.c:    vect[rescont][residx][1][1]=  dn_h * cos( angh_n_ct );
calcvect.c:    vect[rescont][residx][1][2]= -dn_h * sin( angh_n_ct );
calcvect.c:    vect[rescont][residx][1][3]= 0.000;
calcvect.c:/* Creation of vector 2 */
calcvect.c:
calcvect.c:    vect[rescont][residx][2][1]= dn_ct;
calcvect.c:    vect[rescont][residx][2][2]= 0.000;
calcvect.c:    vect[rescont][residx][2][3]= 0.000;
calcvect.c:/* Creation of vect[rescont][residx]or 3 */
calcvect.c:    vect[rescont][residx][3][1]= dct_hc;
calcvect.c:    vect[rescont][residx][3][2]= 0.000;
calcvect.c:    vect[rescont][residx][3][3]= 0.000;
calcvect.c:    ptr_transf=&transf[1][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][3][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:/* Creation of vect[rescont][residx]or 4 */
calcvect.c:    vect[rescont][residx][4][1]= dct_c;
calcvect.c:    vect[rescont][residx][4][2]= 0.000;
calcvect.c:    vect[rescont][residx][4][3]= 0.000;
calcvect.c:    ptr_transf=&transf[2][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][4][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:/* Creation of vect[rescont][residx]or 5 */
calcvect.c:    vect[rescont][residx][5][1]= dct_ct;
calcvect.c:    vect[rescont][residx][5][2]= 0.000;
calcvect.c:    vect[rescont][residx][5][3]= 0.000;
calcvect.c:    ptr_transf=&transf[3][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][5][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:/* Creation of vect[rescont][residx]or 6 */
calcvect.c:    vect[rescont][residx][6][1]= dn_c;
calcvect.c:    vect[rescont][residx][6][2]= 0.000;
calcvect.c:    vect[rescont][residx][6][3]= 0.000;
calcvect.c:    ptr_transf=&transf[4][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][6][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:    ptr_transf=&transf[2][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][6][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:
calcvect.c:/* Creation of vect[rescont][residx]or 7 */
calcvect.c:    vect[rescont][residx][7][1]= dct_hc;
calcvect.c:    vect[rescont][residx][7][2]= 0.000;
calcvect.c:    vect[rescont][residx][7][3]= 0.000;
calcvect.c:    ptr_transf=&transf[5][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][7][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:    ptr_transf=&transf[3][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][7][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:/* Creation of vect[rescont][residx]or 8 */
calcvect.c:    vect[rescont][residx][8][1]= dct_hc;
calcvect.c:    vect[rescont][residx][8][2]= 0.000;
calcvect.c:    vect[rescont][residx][8][3]= 0.000;
calcvect.c:    ptr_transf=&transf[6][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][8][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:    ptr_transf=&transf[3][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][8][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:/* Creation of vect[rescont][residx]or 9 */
calcvect.c:    vect[rescont][residx][9][1]= dct_hc;
calcvect.c:    vect[rescont][residx][9][2]= 0.000;
calcvect.c:    vect[rescont][residx][9][3]= 0.000;
calcvect.c:    ptr_transf=&transf[3][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][9][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:/* Creation of vect[rescont][residx]or 10 */
calcvect.c:    vect[rescont][residx][10][1]= dct_hc;
calcvect.c:    vect[rescont][residx][10][2]= 0.000;
```

```
calcvect.c:    vect[rescont][residx][10][3]= 0.000;
calcvect.c:    ptr_transf=&transf[8][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][10][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:    ptr_transf=&transf[3][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][10][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:
calcvect.c:/* Creation of vect[rescont][residx]or 11 */
calcvect.c:    vect[rescont][residx][11][1]= dn_h;
calcvect.c:    vect[rescont][residx][11][2]= 0.000;
calcvect.c:    vect[rescont][residx][11][3]= 0.000;
calcvect.c:    ptr_transf=&transf[9][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][11][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:    ptr_transf=&transf[4][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][11][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:    ptr_transf=&transf[2][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][11][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:/* Creation of vect[rescont][residx]or 12 */
calcvect.c:    vect[rescont][residx][12][1]= dn_ct;
calcvect.c:    vect[rescont][residx][12][2]= 0.000;
calcvect.c:    vect[rescont][residx][12][3]= 0.000;
calcvect.c:    ptr_transf=&transf[7][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][12][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:    ptr_transf=&transf[4][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][12][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:    ptr_transf=&transf[2][1][1];
calcvect.c:    ptr_vect=&vect[rescont][residx][12][1];
calcvect.c:    transf_vec(ptr_vect,ptr_transf);
calcvect.c:/* Transformation of vectors */
calcvect.c:   for(i=1;i<13;i++)
calcvect.c:  {
calcvect.c:    vcttmp1=vect[rescont][residx][i][1];
calcvect.c:    vcttmp2=vect[rescont][residx][i][2];
calcvect.c:    vcttmp3=vect[rescont][residx][i][3];
calcvect.c:    vect[rescont][residx][i][1]=vectsys[residx-1][1][1]*vcttmp1 +
calcvect.c:                         vectsys[residx-1][2][1]*vcttmp2 +
calcvect.c:                         vectsys[residx-1][3][1]*vcttmp3;
calcvect.c:    vect[rescont][residx][i][2]=vectsys[residx-1][1][2]*vcttmp1 +
calcvect.c:                         vectsys[residx-1][2][2]*vcttmp2 +
calcvect.c:                         vectsys[residx-1][3][2]*vcttmp3;
calcvect.c:    vect[rescont][residx][i][3]=vectsys[residx-1][1][3]*vcttmp1 +
calcvect.c:                         vectsys[residx-1][2][3]*vcttmp2 +
calcvect.c:                         vectsys[residx-1][3][3]*vcttmp3;
calcvect.c:  }
calcvect.c:/* Creation of vectsys */
calcvect.c:   ptr_vect=&vectsys[residx][1][1];
calcvect.c:   *ptr_vect++=vect[rescont][residx][12][1];
calcvect.c:   *ptr_vect++=vect[rescont][residx][12][2];
calcvect.c:   *ptr_vect=vect[rescont][residx][12][3];
calcvect.c:   ptr_vect1 = &vect[rescont][residx][12][1];
calcvect.c:   ptr_vect2 = &vect[rescont][residx][11][1];
calcvect.c:   ptr_vect = &vectsys[residx][3][1];
calcvect.c:   mult_vect(ptr_vect1,ptr_vect2,ptr_vect);
calcvect.c:   ptr_vect1 = &vectsys[residx][3][1];
calcvect.c:   ptr_vect2 = &vect[rescont][residx][12][1];
calcvect.c:   ptr_vect = &vectsys[residx][2][1];
calcvect.c:   mult_vect(ptr_vect1,ptr_vect2,ptr_vect);
calcvect.c:   for(i=1;i<4;i++)
calcvect.c:   {
calcvect.c:    ptr_vect1=&vectsys[residx][i][1];
calcvect.c:    ptr_vect2=&vectsys[residx][i][1];
calcvect.c:    unitario(ptr_vect1,ptr_vect2);
calcvect.c:   }
calcvect.c:    vectres[rescont][residx][1] = vectres[rescont][residx-1][1] + vect[rescont][residx][2][1] +
calcvect.c:                         vect[rescont][residx][4][1] + vect[rescont][residx][6][1];
calcvect.c:    vectres[rescont][residx][2] = vectres[rescont][residx-1][2] + vect[rescont][residx][2][2] +
calcvect.c:                         vect[rescont][residx][4][2] + vect[rescont][residx][6][2];
calcvect.c:    vectres[rescont][residx][3] = vectres[rescont][residx-1][3] + vect[rescont][residx][2][3] +
calcvect.c:                         vect[rescont][residx][4][3] + vect[rescont][residx][6][3];
calcvect.c:   residx++;
```

```
calcvect.c:  }
calcvect.c: }
calcvect.c:}
```

```
funciones.c:#include <math.h>
funciones.c:#include <stdio.h>
funciones.c:void crea_transf(float *ptr_mat, float ang_a, float ang_b)
funciones.c:{
funciones.c:/*
funciones.c: The matrix is created in function of the angles theta and epsilon, where
funciones.c: theta it is the angle formed by the bonds i and i+1, and epsilon the
funciones.c: angle among the plans formed by the bonds i-1, i and i, i+1 respectively.
funciones.c: cos(theta)           sin(theta)           0
funciones.c: cos(epsi)*sin(theta) -cos(epsi)*cos(theta)  sin(epsi)
funciones.c: sin(epsi)*sin(theta) -sin(epsi)*cos(theta)  cos(epsi)
funciones.c:*/
funciones.c:  *ptr_mat++ =  cos(ang_a);
funciones.c:  *ptr_mat++ =  sin(ang_a);
funciones.c:  *ptr_mat++ =  0.000;
funciones.c:  *ptr_mat++ =  cos(ang_b) * sin(ang_a);
funciones.c:  *ptr_mat++ = -cos(ang_b) * cos(ang_a);
funciones.c:  *ptr_mat++ =  sin(ang_b);
funciones.c:  *ptr_mat++ =  sin(ang_b) * sin(ang_a);
funciones.c:  *ptr_mat++ = -sin(ang_b) * cos(ang_a);
funciones.c:  *ptr_mat   = -cos(ang_b);
funciones.c:  return  ;
funciones.c:}
funciones.c:void transf_vec(float *ptr_vec, float *ptr_mat)
funciones.c:{
funciones.c:/*
funciones.c: The transformation consists in multiplying the
funciones.c: matrix transf for the vect[rescont]or in question.
funciones.c:*/
funciones.c:    float    orig_comp1,
funciones.c:             orig_comp2,
funciones.c:             orig_comp3;
funciones.c:    float    t11,t12,t13,
funciones.c:             t21,t22,t23,
funciones.c:             t31,t32,t33;
funciones.c:
funciones.c:    orig_comp1 = *ptr_vec++ ;
funciones.c:    orig_comp2 = *ptr_vec++ ;
funciones.c:    orig_comp3 = *ptr_vec ;
funciones.c:    ptr_vec = ptr_vec - 2 ;
funciones.c:    t11 = *ptr_mat++;
funciones.c:    t12 = *ptr_mat++;
funciones.c:    t13 = *ptr_mat++;
funciones.c:    t21 = *ptr_mat++;
funciones.c:    t22 = *ptr_mat++;
funciones.c:    t23 = *ptr_mat++;
funciones.c:    t31 = *ptr_mat++;
funciones.c:    t32 = *ptr_mat++;
funciones.c:    t33 = *ptr_mat;
funciones.c:    *ptr_vec++ = orig_comp1 * t11 + orig_comp2 * t12 + orig_comp3 * t13;
funciones.c:    *ptr_vec++ = orig_comp1 * t21 + orig_comp2 * t22 + orig_comp3 * t23;
funciones.c:    *ptr_vec   = orig_comp1 * t31 + orig_comp2 * t32 + orig_comp3 * t33;
funciones.c:
funciones.c:    return;
funciones.c:}
funciones.c:float modulo(float *ptr_vec)
funciones.c:{
funciones.c:/*
funciones.c: Calculate the module of a vector.
funciones.c:*/
funciones.c:   float    result,
funciones.c:             result_quad;
funciones.c:   float    a,b,c;
funciones.c:
funciones.c:   a = *ptr_vec++;
funciones.c:   b = *ptr_vec++;
funciones.c:   c = *ptr_vec ;
funciones.c:   result_quad = a*a + b*b + c*c;
funciones.c:   result = sqrt(result_quad);
```

```
funciones.c:
funciones.c:   return result;
funciones.c:}
funciones.c:void unidad(float *ptr_mat)
funciones.c:{
funciones.c:/*
funciones.c: Hace una matriz igual a la unidad
funciones.c:*/
funciones.c: *ptr_mat++ = 1;
funciones.c: *ptr_mat++ = 0;
funciones.c: *ptr_mat++ = 0;
funciones.c: *ptr_mat++ = 0;
funciones.c: *ptr_mat++ = 1;
funciones.c: *ptr_mat++ = 0;
funciones.c: *ptr_mat++ = 0;
funciones.c: *ptr_mat++ = 0;
funciones.c: *ptr_mat = 1;
funciones.c: return;
funciones.c:}
funciones.c:void iguala(float *ptr_mat1,float *ptr_mat2)
funciones.c:{
funciones.c:/*
funciones.c: Hace una matriz 2 igual a la matriz 1
funciones.c:*/
funciones.c: *ptr_mat2++ = *ptr_mat1++;
funciones.c: *ptr_mat2++ = *ptr_mat1++;
funciones.c: *ptr_mat2++ = *ptr_mat1++;
funciones.c: *ptr_mat2++ = *ptr_mat1++;
funciones.c: *ptr_mat2++ = *ptr_mat1++;
funciones.c: *ptr_mat2++ = *ptr_mat1++;
funciones.c: *ptr_mat2++ = *ptr_mat1++;
funciones.c: *ptr_mat2++ = *ptr_mat1++;
funciones.c: *ptr_mat2 = *ptr_mat1;
funciones.c: return;
funciones.c:}
funciones.c:void mult_mat(float *ptr_mat1, float *ptr_mat2,float *ptr_matr)
funciones.c:{
funciones.c:/*
funciones.c: Producto de dos matrices de tres por tres.
funciones.c:*/
funciones.c:    float    a11,a12,a13,
funciones.c:             a21,a22,a23,
funciones.c:             a31,a32,a33;
funciones.c:
funciones.c:    float    b11,b12,b13,
funciones.c:             b21,b22,b23,
funciones.c:             b31,b32,b33;
funciones.c:
funciones.c:    a11 = *ptr_mat1++;
funciones.c:    a12 = *ptr_mat1++;
funciones.c:    a13 = *ptr_mat1++;
funciones.c:    a21 = *ptr_mat1++;
funciones.c:    a22 = *ptr_mat1++;
funciones.c:    a23 = *ptr_mat1++;
funciones.c:    a31 = *ptr_mat1++;
funciones.c:    a32 = *ptr_mat1++;
funciones.c:    a33 = *ptr_mat1;
funciones.c:    b11 = *ptr_mat2++;
funciones.c:    b12 = *ptr_mat2++;
funciones.c:    b13 = *ptr_mat2++;
funciones.c:    b21 = *ptr_mat2++;
funciones.c:    b22 = *ptr_mat2++;
funciones.c:    b23 = *ptr_mat2++;
funciones.c:    b31 = *ptr_mat2++;
funciones.c:    b32 = *ptr_mat2++;
funciones.c:    b33 = *ptr_mat2;
funciones.c:    *ptr_matr++ = a11 * b11 + a12 * b21 + a13 *b31;
funciones.c:    *ptr_matr++ = a11 * b12 + a12 * b22 + a13 *b32;
funciones.c:    *ptr_matr++ = a11 * b13 + a12 * b23 + a13 *b33;
funciones.c:    *ptr_matr++ = a21 * b11 + a22 * b21 + a23 *b31;
funciones.c:    *ptr_matr++ = a21 * b12 + a22 * b22 + a23 *b32;
funciones.c:    *ptr_matr++ = a21 * b13 + a22 * b23 + a23 *b33;
funciones.c:    *ptr_matr++ = a31 * b11 + a32 * b21 + a33 *b31;
funciones.c:    *ptr_matr++ = a31 * b12 + a32 * b22 + a33 *b32;
funciones.c:    *ptr_matr   = a31 * b13 + a32 * b23 + a33 *b33;
funciones.c:
```

```
funciones.c:    return;
funciones.c:}
funciones.c:int getline(FILE *fpinp,char s[],int lim)
funciones.c:{
funciones.c: int c,i;
funciones.c: i=0;
funciones.c: while (--lim > 0 && (c=getc(fpinp)) != EOF && c != '\n')
funciones.c:            s[i++]=c;
funciones.c: if (c == '\n')
funciones.c:    s[i++] = c;
funciones.c: s[i] = '\0';
funciones.c: return i;
funciones.c:}
funciones.c:void sumvect(float *ptr_vect1,float *ptr_vect2,float *ptr_vects)
funciones.c:{
funciones.c:       *ptr_vects++ = *ptr_vect1++ + *ptr_vect2++;
funciones.c:       *ptr_vects++ = *ptr_vect1++ + *ptr_vect2++;
funciones.c:       *ptr_vects = *ptr_vect1 + *ptr_vect2;
funciones.c: return;
funciones.c:}
funciones.c:void mult_vect(float *ptr_vect1, float *ptr_vect2,float *ptr_vect)
funciones.c:{
funciones.c:/*
funciones.c: Producto de dos vectores de tres elementos.
funciones.c:*/
funciones.c:    float    a1,a2,a3;
funciones.c:
funciones.c:    float    b1,b2,b3;
funciones.c:
funciones.c:    a1 = *ptr_vect1++;
funciones.c:    a2 = *ptr_vect1++;
funciones.c:    a3 = *ptr_vect1;
funciones.c:    b1 = *ptr_vect2++;
funciones.c:    b2 = *ptr_vect2++;
funciones.c:    b3 = *ptr_vect2;
funciones.c:    *ptr_vect++ = a2 * b3 - a3 * b2;
funciones.c:    *ptr_vect++ = a3 * b1 - a1 * b3;
funciones.c:    *ptr_vect   = a1 * b2 - a2 * b1;
funciones.c:
funciones.c:    return;
funciones.c:}
funciones.c:void unitario(float *ptr_vect1, float *ptr_vect2)
funciones.c:{
funciones.c:
funciones.c:     float a,b,c;
funciones.c:     float modulo;
funciones.c:     a=*ptr_vect1++;
funciones.c:     b=*ptr_vect1++;
funciones.c:     c=*ptr_vect1;
funciones.c:
funciones.c:     modulo=sqrt(a*a+b*b+c*c);
funciones.c:     *ptr_vect2++=a/modulo;
funciones.c:     *ptr_vect2++=b/modulo;
funciones.c:     *ptr_vect2=c/modulo;
funciones.c:     return;
funciones.c:}
```

```
main.c:#include <math.h>
main.c:#include <string.h>
main.c:#include <stdio.h>
main.c:#include "dist.h"
main.c:#include "vect.h"
main.c:main()
main.c: {
main.c:    FILE *distinp;
main.c:    FILE *vectinp;
main.c:    char namevec[MAXLINE],namedis[MAXLINE];
main.c:    char line[MAXLINE];
main.c:    int res1,res2;
main.c:    char atom1[5],atom2[5];
main.c:    float distancia;
main.c:    printf("Fichero de diedros: ");
main.c:    getline(stdin,line, sizeof(line));
```

```
main.c:    sscanf(line,"%s",&namevec);
main.c:    vectinp = fopen(namevec,"r");
main.c:    printf("Fichero de distancias: ");
main.c:    getline(stdin,line, sizeof(line));
main.c:    sscanf(line,"%s",&namedis);
main.c:
main.c:    distinp = fopen(namedis,"r");
main.c:    calcvect(vectinp);
main.c:     while(getline(distinp,line,sizeof(line)) > 0)
main.c:        {
main.c:     sscanf(line,"%d %s %d %s",&res1,&atom1,&res2,&atom2);
main.c:     distancia=calcdist(res1,atom1,res2,atom2);
main.c:     printf("%-4d%-6s%-4d%-6s%-7.3f \n",res1,atom1,res2,atom2,distancia);
main.c:        }
main.c:    return;
main.c:}
```

Apéndice D

Conjunto de macros GRAPHIREX

A continuación se exponen el conjunto de programas de UNIX (*shell scripts*) que preparan los ficheros de entrada para el programa gnuplot y superponen todos los mapas de contornos. También se incluyen un ejemplo de fichero de comandos para las macros y un fichero de entrada estándar. El nombre de la macro se incluye al principio de cada línea.

Cada macro funciona con un tipo de modelo de Lipari-Szabo. graphirex2,sh trabaja con el modelo 2, graphirex3 con el modelo 3, etc. Las macros auxiliares de cálculo de R_{ex}, por ejemplo, son invocadas automáticamente por la macro principal cuando se necesita. El fichero formula.inp contiene todas las fórmulas matemáticas necesarias para la generación de los mapas de contornos.

```
eval_rex.sh:#!/bin/sh
eval_rex.sh:curro=`pwd`
eval_rex.sh:echo -n "R1  :"
eval_rex.sh:read r1
eval_rex.sh:echo -n "R1 uncertain :"
eval_rex.sh:read r1incr
eval_rex.sh:echo -n "R2  :"
eval_rex.sh:read r2
eval_rex.sh:echo -n "R2 uncertain :"
eval_rex.sh:read r2incr
eval_rex.sh:echo -n "Noe :"
eval_rex.sh:read noe
eval_rex.sh:echo -n "Noe uncertain :"
eval_rex.sh:read noeincr
eval_rex.sh:echo -n "Tm min :"
eval_rex.sh:read tm
eval_rex.sh:tm=`echo "$tm" |bc -l`
eval_rex.sh:echo -n "Tm max :"
eval_rex.sh:read tmfinal
eval_rex.sh:tmfinal=`echo "$tmfinal " |bc -l`
eval_rex.sh:echo -n "Tm increment :"
eval_rex.sh:read tmincr
eval_rex.sh:tmincr=`echo "$tmincr " |bc -l`
```

```
eval_rex.sh:echo -n "WH :"
eval_rex.sh:read wh
eval_rex.sh:echo -n "Te min"
eval_rex.sh:read temin
eval_rex.sh:echo -n "Te max :"
eval_rex.sh:read temax
eval_rex.sh:echo -n "Ficheros PS :"
eval_rex.sh:read filename
eval_rex.sh:tesfinal=`echo "$tmfinal * 100000 " |bc -l`
eval_rex.sh:tes=`echo "$tm * 100000 " |bc -l`
eval_rex.sh:tesinc=`echo "$tmincr * 100000 " |bc -l`
eval_rex.sh:r1max=`echo "$r1 + $r1incr " |bc -l`
eval_rex.sh:r1min=`echo "$r1 - $r1incr " |bc -l`
eval_rex.sh:r2max=`echo "$r2 + $r2incr " |bc -l`
eval_rex.sh:r2min=`echo "$r2 - $r2incr " |bc -l`
eval_rex.sh:noemax=`echo "$noe + $noeincr " |bc -l`
eval_rex.sh:noemin=`echo "$noe - $noeincr " |bc -l`
eval_rex.sh:tetic=`echo " ( $temax - $temin ) / 10 " | bc -l`
eval_rex.sh:if  [  -d "${filename}_PS" ]
eval_rex.sh:then
eval_rex.sh:echo "${filename}_PS exists"
eval_rex.sh:echo "Maybe files will be overwriten"
eval_rex.sh:else
eval_rex.sh:mkdir ${filename}_PS
eval_rex.sh:fi
eval_rex.sh:cd ${filename}_PS
eval_rex.sh:echo "#Input para contornos" > gnutmp.inp
eval_rex.sh:while [ $tes -le $tesfinal ]
eval_rex.sh:do
eval_rex.sh:echo "gh = 2.67519e8" >> gnutmp.inp
eval_rex.sh:echo "gn = -2.711e7" >> gnutmp.inp
eval_rex.sh:echo "wh = $wh *2.0*3.14159" >> gnutmp.inp
eval_rex.sh:echo "wn = wh * gn/gh  " >> gnutmp.inp
eval_rex.sh:echo "tm = $tm" >> gnutmp.inp
eval_rex.sh:echo "u = 0.519409479 * gh/gn  " >> gnutmp.inp
eval_rex.sh:echo "tau(y) = (y*tm)/(y+tm)" >> gnutmp.inp
eval_rex.sh:echo "jac(w,x,y) = (x*tm/(1.0+(w*tm)**2)+(1.0-x)*tau(y)/(1.0+(w*tau(y))**2))" >> gnutmp.inp
eval_rex.sh:echo "r1(x,y) = (0.519409479)*(jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y))+1.0/3.0*0.4*((-0.16)**2)*w
n**2*1000.0*jac(wn,x,y) " >> gnutmp.inp
eval_rex.sh:echo "r2(x,y) = (0.2597047039)*(4.0*jac(0.0,x,y)+jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y)+6.0*jac(w
h,x,y))+1.0/18.0*0.4*((-0.16)**2)*wn**2*1000.0*(4.0*jac(0.0,x,y)+3.0*jac(wn,x,y))" >> gnutmp.inp
eval_rex.sh:echo "noe(x,y)= 1 + u*(6*jac(wh+wn,x,y)-jac(wh-wn,x,y))/r1(x,y)" >> gnutmp.inp
eval_rex.sh:echo "set term postscript " >> gnutmp.inp
eval_rex.sh:echo "set output '${filename}_${tm}_rex.ps' " >> gnutmp.inp
eval_rex.sh:echo "set title 'R1 $r1min $r1max R2 $r2min $r2max HNOE $noemin $noemax Tm ${tm} Wh $wh ' " >> gnutmp.inp
eval_rex.sh:echo "set label '${filename}' at -0.5,${temax}  " >> gnutmp.inp
eval_rex.sh:echo "set xlabel 'S2' " >> gnutmp.inp
eval_rex.sh:echo "set ylabel 'Te' " >> gnutmp.inp
eval_rex.sh:echo "set view 180, 90, 1, 1" >> gnutmp.inp
eval_rex.sh:echo "set samples 100" >> gnutmp.inp
eval_rex.sh:echo "set isosamples 100" >> gnutmp.inp
eval_rex.sh:echo "set nosurface" >> gnutmp.inp
eval_rex.sh:echo "set contour base" >> gnutmp.inp
eval_rex.sh:echo "set cntrparam bspline" >> gnutmp.inp
eval_rex.sh:echo "set xrange [0:1]" >> gnutmp.inp
eval_rex.sh:echo "set yrange [${temin}:${temax}]" >> gnutmp.inp
eval_rex.sh:echo "set xtics 0,0.2,1" >> gnutmp.inp
eval_rex.sh:echo "set ytics ${temin},${tetic},${temax}" >> gnutmp.inp
eval_rex.sh:echo "set contour" >> gnutmp.inp
eval_rex.sh:echo "set key 1.0,12" >> gnutmp.inp
eval_rex.sh:echo "set cntrparam levels disc $r1max,$r1min" >> gnutmp.inp
eval_rex.sh:echo "splot r1(x,y)" >> gnutmp.inp
eval_rex.sh:echo "set term postscript color solid " >> gnutmp.inp
eval_rex.sh:echo "set key 0.8,${temax}" >> gnutmp.inp
eval_rex.sh:echo "set cntrparam levels inc 0.1,0.1,10.0" >> gnutmp.inp
eval_rex.sh:echo "splot r2(x,y)" >> gnutmp.inp
eval_rex.sh:echo "set term postscript default " >> gnutmp.inp
eval_rex.sh:echo "set key 0.6,12" >> gnutmp.inp
eval_rex.sh:echo "set cntrparam levels disc $noemax,$noemin" >> gnutmp.inp
eval_rex.sh:echo "splot noe(x,y)" >> gnutmp.inp
eval_rex.sh:tm=`echo "$tm + $tmincr" |bc -l`
eval_rex.sh:tes=`echo "$tes + $tesinc" |bc -l`
eval_rex.sh:done
eval_rex.sh:gnuplot gnutmp.inp
eval_rex.sh:# rm gnutmp.inp
eval_rex.sh:for i in ${filename}_*.ps
```

```
eval_rex.sh:do
eval_rex.sh:grep -v age $i  > tmp
eval_rex.sh:echo "showpage" >> tmp
eval_rex.sh:mv tmp $i
eval_rex.sh:done
eval_rex.sh:cd $curro
```

```
graphirex2.sh:#!/bin/sh
graphirex2.sh:curro=`pwd`
graphirex2.sh:echo -n "R1  :"
graphirex2.sh:read r1
graphirex2.sh:echo -n "R1 uncertain :"
graphirex2.sh:read r1incr
graphirex2.sh:echo -n "R2  :"
graphirex2.sh:read r2
graphirex2.sh:echo -n "R2 uncertain :"
graphirex2.sh:read r2incr
graphirex2.sh:echo -n "Noe :"
graphirex2.sh:read noe
graphirex2.sh:echo -n "Noe uncertain :"
graphirex2.sh:read noeincr
graphirex2.sh:echo -n "Tm min :"
graphirex2.sh:read tm
graphirex2.sh:tm=`echo "$tm" |bc -l`
graphirex2.sh:echo -n "Tm max :"
graphirex2.sh:read tmfinal
graphirex2.sh:tmfinal=`echo "$tmfinal " |bc -l`
graphirex2.sh:echo -n "Tm increment :"
graphirex2.sh:read tmincr
graphirex2.sh:tmincr=`echo "$tmincr " |bc -l`
graphirex2.sh:echo -n "WH :"
graphirex2.sh:read wh
graphirex2.sh:echo -n "Te min"
graphirex2.sh:read temin
graphirex2.sh:echo -n "Te max :"
graphirex2.sh:read temax
graphirex2.sh:echo -n "Output PS Directory :"
graphirex2.sh:read filename
graphirex2.sh:tesfinal=`echo "$tmfinal * 100000 " |bc -l`
graphirex2.sh:tes=`echo "$tm * 100000 " |bc -l`
graphirex2.sh:tesinc=`echo "$tmincr * 100000 " |bc -l`
graphirex2.sh:r1max=`echo "$r1 + $r1incr " |bc -l`
graphirex2.sh:r1min=`echo "$r1 - $r1incr " |bc -l`
graphirex2.sh:r2max=`echo "$r2 + $r2incr " |bc -l`
graphirex2.sh:r2min=`echo "$r2 - $r2incr " |bc -l`
graphirex2.sh:noemax=`echo "$noe + $noeincr " |bc -l`
graphirex2.sh:noemin=`echo "$noe - $noeincr " |bc -l`
graphirex2.sh:tetic=`echo " ( $temax - $temin ) / 10 " | bc -l`
graphirex2.sh:if  [  -d "${filename}_PS" ]
graphirex2.sh:then
graphirex2.sh:echo "${filename}_PS exists"
graphirex2.sh:echo "Maybe files will be overwriten"
graphirex2.sh:# echo -n "Do you want overwrite it ([y]/n)?"
graphirex2.sh:# read answer
graphirex2.sh:else
graphirex2.sh:mkdir ${filename}_PS
graphirex2.sh:fi
graphirex2.sh:# if [ "$answer" = "n" ]
graphirex2.sh:# then
graphirex2.sh:# exit 1
graphirex2.sh:# fi
graphirex2.sh:cd ${filename}_PS
graphirex2.sh:echo "#Input para contornos" > gnutmp.inp
graphirex2.sh:while [ $tes -le $tesfinal ]
graphirex2.sh:do
graphirex2.sh:echo "gh = 2.67519e8" >> gnutmp.inp
graphirex2.sh:echo "gn = -2.711e7" >> gnutmp.inp
graphirex2.sh:echo "wh = $wh *2.0*3.14159" >> gnutmp.inp
graphirex2.sh:echo "wn = wh * gn/gh  " >> gnutmp.inp
graphirex2.sh:echo "tm = $tm" >> gnutmp.inp
graphirex2.sh:echo "u = 0.519409479 * gh/gn  " >> gnutmp.inp
graphirex2.sh:echo "tau(y) = (y*tm)/(y+tm)" >> gnutmp.inp
graphirex2.sh:echo "jac(w,x,y) = (x*tm/(1.0+(w*tm)**2)+(1.0-x)*tau(y)/(1.0+(w*tau(y))**2))" >> gnutmp.inp
```

```
graphirex2.sh:echo "r1(x,y) = (0.519409479)*(jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y))+1.0/3.0*0.4*((-0.16)**2)
*wn**2*1000.0*jac(wn,x,y) " >> gnutmp.inp
graphirex2.sh:echo "r2(x,y) = (0.2597047039)*(4.0*jac(0.0,x,y)+jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y)+6.0*jac
(wh,x,y))+1.0/18.0*0.4*((-0.16)**2)*wn**2*1000.0*(4.0*jac(0.0,x,y)+3.0*jac(wn,x,y))" >> gnutmp.inp
graphirex2.sh:echo "noe(x,y)= 1 + u*(6*jac(wh+wn,x,y)-jac(wh-wn,x,y))/r1(x,y)" >> gnutmp.inp
graphirex2.sh:echo "set term postscript color solid " >> gnutmp.inp
graphirex2.sh:echo "set output '${filename}_${tm}.ps' " >> gnutmp.inp
graphirex2.sh:echo "set title 'R1 $r1min $r1max R2 $r2min $r2max HNOE $noemin $noemax Tm ${tm} Wh $wh' " >> gnutmp.inp
graphirex2.sh:echo "set label '${filename}' at -0.5,${temax}  " >> gnutmp.inp
graphirex2.sh:echo "set xlabel 'S2' " >> gnutmp.inp
graphirex2.sh:echo "set ylabel 'Te' " >> gnutmp.inp
graphirex2.sh:echo "set view 180, 90, 1, 1" >> gnutmp.inp
graphirex2.sh:echo "set samples 100" >> gnutmp.inp
graphirex2.sh:echo "set isosamples 100" >> gnutmp.inp
graphirex2.sh:echo "set nosurface" >> gnutmp.inp
graphirex2.sh:echo "set contour base" >> gnutmp.inp
graphirex2.sh:echo "set cntrparam bspline" >> gnutmp.inp
graphirex2.sh:echo "set xrange [0:1]" >> gnutmp.inp
graphirex2.sh:echo "set yrange [${temin}:${temax}]" >> gnutmp.inp
graphirex2.sh:echo "set xtics 0,0.2,1" >> gnutmp.inp
graphirex2.sh:echo "set ytics ${temin},${tetic},${temax}" >> gnutmp.inp
graphirex2.sh:echo "set contour" >> gnutmp.inp
graphirex2.sh:echo "set key 1.0,12" >> gnutmp.inp
graphirex2.sh:echo "set cntrparam levels disc $r1max,$r1min" >> gnutmp.inp
graphirex2.sh:echo "splot r1(x,y)" >> gnutmp.inp
graphirex2.sh:echo "set key 0.8,12" >> gnutmp.inp
graphirex2.sh:echo "set cntrparam levels disc $r2max,$r2min" >> gnutmp.inp
graphirex2.sh:echo "splot r2(x,y)" >> gnutmp.inp
graphirex2.sh:echo "set key 0.6,12" >> gnutmp.inp
graphirex2.sh:echo "set cntrparam levels disc $noemax,$noemin" >> gnutmp.inp
graphirex2.sh:echo "splot noe(x,y)" >> gnutmp.inp
graphirex2.sh:tm=`echo "$tm + $tmincr" |bc -l`
graphirex2.sh:tes=`echo "$tes + $tesinc" |bc -l`
graphirex2.sh:done
graphirex2.sh:gnuplot gnutmp.inp
graphirex2.sh:# rm gnutmp.inp
graphirex2.sh:for i in ${filename}_*.ps
graphirex2.sh:do
graphirex2.sh:grep -v age $i  > tmp
graphirex2.sh:echo "showpage" >> tmp
graphirex2.sh:mv tmp $i
graphirex2.sh:done
graphirex2.sh:cd $curro
```

```
graphirex3.sh:#!/bin/sh
graphirex3.sh:curro=`pwd`
graphirex3.sh:echo -n "R1  :"
graphirex3.sh:read r1
graphirex3.sh:echo -n "R1 uncertain :"
graphirex3.sh:read r1incr
graphirex3.sh:echo -n "R2  :"
graphirex3.sh:read r2
graphirex3.sh:echo -n "R2 uncertain :"
graphirex3.sh:read r2incr
graphirex3.sh:echo -n "Noe :"
graphirex3.sh:read noe
graphirex3.sh:echo -n "Noe uncertain :"
graphirex3.sh:read noeincr
graphirex3.sh:echo -n "Tm min :"
graphirex3.sh:read tm
graphirex3.sh:tm=`echo "$tm" |bc -l`
graphirex3.sh:echo -n "Tm max :"
graphirex3.sh:read tmfinal
graphirex3.sh:tmfinal=`echo "$tmfinal " |bc -l`
graphirex3.sh:echo -n "Tm increment :"
graphirex3.sh:read tmincr
graphirex3.sh:tmincr=`echo "$tmincr " |bc -l`
graphirex3.sh:echo -n "WH :"
graphirex3.sh:read wh
graphirex3.sh:echo -n "Rex min"
graphirex3.sh:read temin
graphirex3.sh:echo -n "Rex max :"
graphirex3.sh:read temax
```

```
graphirex3.sh:echo -n "Ficheros PS :"
graphirex3.sh:read filename
graphirex3.sh:tesfinal=`echo "$tmfinal * 100000 " |bc -l`
graphirex3.sh:tes=`echo "$tm * 100000 " |bc -l`
graphirex3.sh:tesinc=`echo "$tmincr * 100000 " |bc -l`
graphirex3.sh:r1max=`echo "$r1 + $r1incr " |bc -l`
graphirex3.sh:r1min=`echo "$r1 - $r1incr " |bc -l`
graphirex3.sh:r2max=`echo "$r2 + $r2incr " |bc -l`
graphirex3.sh:r2min=`echo "$r2 - $r2incr " |bc -l`
graphirex3.sh:noemax=`echo "$noe + $noeincr " |bc -l`
graphirex3.sh:noemin=`echo "$noe - $noeincr " |bc -l`
graphirex3.sh:tetic=`echo " ( $temax - $temin ) / 10 " | bc -l`
graphirex3.sh:if [  -d "${filename}_PS" ]
graphirex3.sh:then
graphirex3.sh:echo "${filename}_PS exists"
graphirex3.sh:echo "Maybe files will be overwriten"
graphirex3.sh:else
graphirex3.sh:mkdir ${filename}_PS
graphirex3.sh:fi
graphirex3.sh:cd ${filename}_PS
graphirex3.sh:echo "#Input para contornos" > gnutmp.inp
graphirex3.sh:while [ $tes -le $tesfinal ]
graphirex3.sh:do
graphirex3.sh:echo "gh = 2.67519e8" >> gnutmp.inp
graphirex3.sh:echo "gn = -2.711e7" >> gnutmp.inp
graphirex3.sh:echo "wh = $wh *2.0*3.14159" >> gnutmp.inp
graphirex3.sh:echo "wn = wh * gn/gh  " >> gnutmp.inp
graphirex3.sh:echo "tm = $tm" >> gnutmp.inp
graphirex3.sh:echo "u = 0.519409479 * gh/gn  " >> gnutmp.inp
graphirex3.sh:echo "jac(w,x) = (x*tm/(1.0+(w*tm)**2)" >> gnutmp.inp
graphirex3.sh:echo "r1(x,y) = (0.519409479)*(jac(wh-wn,x)+3.0*jac(wn,x)+6.0*jac(wh+wn,x))+1.0/3.0*0.4*((-0.16)**2)*wn**2
*1000.0*jac(wn,x) " >> gnutmp.inp
graphirex3.sh:echo "r2(x,y) = (0.2597047039)*(4.0*jac(0.0,x)+jac(wh-wn,x)+3.0*jac(wn,x)+6.0*jac(wh+wn,x)+6.0*jac(wh,x))+
1.0/18.0*0.4*((-0.16)**2)*wn**2*1000.0*(4.0*jac(0.0,x)+3.0*jac(wn,x))+y" >> gnutmp.inp
graphirex3.sh:echo "noe(x,y)= 1 + u*(6*jac(wh+wn,x)-jac(wh-wn,x))/r1(x)" >> gnutmp.inp
graphirex3.sh:echo "set term postscript color solid " >> gnutmp.inp
graphirex3.sh:echo "set output '${filename}_${tm}.ps' " >> gnutmp.inp
graphirex3.sh:echo "set title 'R1 $r1min $r1max R2 $r2min $r2max HNOE $noemin $noemax Tm ${tm} Wh $wh' " >> gnutmp.inp
graphirex3.sh:echo "set label '${filename}' at -0.5,${temax}  " >> gnutmp.inp
graphirex3.sh:echo "set xlabel 'S2' " >> gnutmp.inp
graphirex3.sh:echo "set ylabel 'Te' " >> gnutmp.inp
graphirex3.sh:echo "set view 180, 90, 1, 1" >> gnutmp.inp
graphirex3.sh:echo "set samples 100" >> gnutmp.inp
graphirex3.sh:echo "set isosamples 100" >> gnutmp.inp
graphirex3.sh:echo "set nosurface" >> gnutmp.inp
graphirex3.sh:echo "set contour base" >> gnutmp.inp
graphirex3.sh:echo "set cntrparam bspline" >> gnutmp.inp
graphirex3.sh:echo "set xrange [0:1]" >> gnutmp.inp
graphirex3.sh:echo "set yrange [${temin}:${temax}]" >> gnutmp.inp
graphirex3.sh:echo "set xtics 0,0.2,1" >> gnutmp.inp
graphirex3.sh:echo "set ytics ${temin},${tetic},${temax}" >> gnutmp.inp
graphirex3.sh:echo "set contour" >> gnutmp.inp
graphirex3.sh:echo "set key 1.0,12" >> gnutmp.inp
graphirex3.sh:echo "set cntrparam levels disc $r1max,$r1min" >> gnutmp.inp
graphirex3.sh:echo "splot r1(x,y)" >> gnutmp.inp
graphirex3.sh:echo "set key 0.8,12" >> gnutmp.inp
graphirex3.sh:echo "set cntrparam levels disc $r2max,$r2min" >> gnutmp.inp
graphirex3.sh:echo "splot r2(x,y)" >> gnutmp.inp
graphirex3.sh:echo "set key 0.6,12" >> gnutmp.inp
graphirex3.sh:echo "set cntrparam levels disc $noemax,$noemin" >> gnutmp.inp
graphirex3.sh:echo "splot noe(x,y)" >> gnutmp.inp
graphirex3.sh:tm=`echo "$tm + $tmincr" |bc -l`
graphirex3.sh:tes=`echo "$tes + $tesinc" |bc -l`
graphirex3.sh:done
graphirex3.sh:gnuplot gnutmp.inp
graphirex3.sh:# rm gnutmp.inp
graphirex3.sh:for i in ${filename}_*.ps
graphirex3.sh:do
graphirex3.sh:grep -v age $i  > tmp
graphirex3.sh:echo "showpage" >> tmp
graphirex3.sh:mv tmp $i
graphirex3.sh:done
graphirex3.sh:cd $curro
```

```
graphirex4.sh:#!/bin/sh
graphirex4.sh:curro=`pwd`
graphirex4.sh:if [ $# -ne 4 ]
graphirex4.sh:then
graphirex4.sh:echo "Usage: $0 rexinitial rexfinal rexincrement fileinput"
graphirex4.sh:exit 1
graphirex4.sh:fi
graphirex4.sh:rex=$1
graphirex4.sh:rexfinal=$2
graphirex4.sh:rexinc=$3
graphirex4.sh:inputfile=$4
graphirex4.sh:tesfinal=`echo "$rexfinal * 100000 " |bc -l`
graphirex4.sh:tes=`echo "$rex * 100000 " |bc -l`
graphirex4.sh:tesinc=`echo "$rexinc * 100000 " |bc -l`
graphirex4.sh:while [ $tes -le $tesfinal ]
graphirex4.sh:do
graphirex4.sh:rex=`echo "$rex + $rexinc" |bc -l`
graphirex4.sh:tes=`echo "$tes + $tesinc" |bc -l`
graphirex4.sh:graphirex4base.sh $rex < $inputfile
graphirex4.sh:done
graphirex4.sh:cd $curro
```

```
graphirex4base.sh:#!/bin/sh
graphirex4base.sh:curro=`pwd`
graphirex4base.sh:rex=$1
graphirex4base.sh:echo -n "R1  :"
graphirex4base.sh:read r1
graphirex4base.sh:echo -n "R1 uncertain :"
graphirex4base.sh:read r1incr
graphirex4base.sh:echo -n "R2  :"
graphirex4base.sh:read r2
graphirex4base.sh:echo -n "R2 uncertain :"
graphirex4base.sh:read r2incr
graphirex4base.sh:echo -n "Noe :"
graphirex4base.sh:read noe
graphirex4base.sh:echo -n "Noe uncertain :"
graphirex4base.sh:read noeincr
graphirex4base.sh:echo -n "Tm min :"
graphirex4base.sh:read tm
graphirex4base.sh:tm=`echo "$tm" |bc -l`
graphirex4base.sh:echo -n "Tm max :"
graphirex4base.sh:read tmfinal
graphirex4base.sh:tmfinal=`echo "$tmfinal " |bc -l`
graphirex4base.sh:echo -n "Tm increment :"
graphirex4base.sh:read tmincr
graphirex4base.sh:tmincr=`echo "$tmincr " |bc -l`
graphirex4base.sh:echo -n "WH :"
graphirex4base.sh:read wh
graphirex4base.sh:echo -n "Te min"
graphirex4base.sh:read temin
graphirex4base.sh:echo -n "Te max :"
graphirex4base.sh:read temax
graphirex4base.sh:echo -n "Ficheros PS :"
graphirex4base.sh:read filename
graphirex4base.sh:tesfinal=`echo "$tmfinal * 100000 " |bc -l`
graphirex4base.sh:tes=`echo "$tm * 100000 " |bc -l`
graphirex4base.sh:tesinc=`echo "$tmincr * 100000 " |bc -l`
graphirex4base.sh:r1max=`echo "$r1 + $r1incr " |bc -l`
graphirex4base.sh:r1min=`echo "$r1 - $r1incr " |bc -l`
graphirex4base.sh:r2max=`echo "$r2 + $r2incr " |bc -l`
graphirex4base.sh:r2min=`echo "$r2 - $r2incr " |bc -l`
graphirex4base.sh:noemax=`echo "$noe + $noeincr " |bc -l`
graphirex4base.sh:noemin=`echo "$noe - $noeincr " |bc -l`
graphirex4base.sh:tetic=`echo " ( $temax - $temin ) / 10 " | bc -l`
graphirex4base.sh:if  [  -d "${filename}_PS" ]
graphirex4base.sh:then
graphirex4base.sh:echo "${filename}_PS exists"
graphirex4base.sh:echo "Maybe files will be overwriten"
graphirex4base.sh:else
graphirex4base.sh:mkdir ${filename}_PS
graphirex4base.sh:fi
graphirex4base.sh:cd ${filename}_PS
graphirex4base.sh:echo "#Input para contornos" > gnutmp.inp
```

```
graphirex4base.sh:while [ $tes -le $tesfinal ]
graphirex4base.sh:do
graphirex4base.sh:echo "gh = 2.67519e8" >> gnutmp.inp
graphirex4base.sh:echo "gn = -2.711e7" >> gnutmp.inp
graphirex4base.sh:echo "wh = $wh *2.0*3.14159" >> gnutmp.inp
graphirex4base.sh:echo "wn = wh * gn/gh  " >> gnutmp.inp
graphirex4base.sh:echo "tm = $tm" >> gnutmp.inp
graphirex4base.sh:echo "u = 0.519409479 * gh/gn  " >> gnutmp.inp
graphirex4base.sh:echo "tau(y) = (y*tm)/(y+tm)" >> gnutmp.inp
graphirex4base.sh:echo "jac(w,x,y) = (x*tm/(1.0+(w*tm)**2)+(1.0-x)*tau(y)/(1.0+(w*tau(y))**2))" >> gnutmp.inp
graphirex4base.sh:echo "r1(x,y) = (0.519409479)*(jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y))+1.0/3.0*0.4*((-0.16)
**2)*wn**2*1000.0*jac(wn,x,y) " >> gnutmp.inp
graphirex4base.sh:echo "r2(x,y) = (0.2597047039)*(4.0*jac(0.0,x,y)+jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y)+6.0
*jac(wh,x,y))+1.0/18.0*0.4*((-0.16)**2)*wn**2*1000.0*(4.0*jac(0.0,x,y)+3.0*jac(wn,x,y))+${rex}" >> gnutmp.inp
graphirex4base.sh:echo "noe(x,y)= 1 + u*(6*jac(wh+wn,x,y)-jac(wh-wn,x,y))/r1(x,y)" >> gnutmp.inp
graphirex4base.sh:echo "set term postscript color solid " >> gnutmp.inp
graphirex4base.sh:echo "set output '${filename}_${tm}_${rex}.ps' " >> gnutmp.inp
graphirex4base.sh:echo "set title 'R1 $r1min $r1max R2 $r2min $r2max HNOE $noemin $noemax Tm ${tm} Wh $wh Rex ${rex}' "
>> gnutmp.inp
graphirex4base.sh:echo "set label '${filename}' at -0.5,${temax}  " >> gnutmp.inp
graphirex4base.sh:echo "set xlabel 'S2' " >> gnutmp.inp
graphirex4base.sh:echo "set ylabel 'Te' " >> gnutmp.inp
graphirex4base.sh:echo "set view 180, 90, 1, 1" >> gnutmp.inp
graphirex4base.sh:echo "set samples 100" >> gnutmp.inp
graphirex4base.sh:echo "set isosamples 100" >> gnutmp.inp
graphirex4base.sh:echo "set nosurface" >> gnutmp.inp
graphirex4base.sh:echo "set contour base" >> gnutmp.inp
graphirex4base.sh:echo "set cntrparam bspline" >> gnutmp.inp
graphirex4base.sh:echo "set xrange [0:1]" >> gnutmp.inp
graphirex4base.sh:echo "set yrange [${temin}:${temax}]" >> gnutmp.inp
graphirex4base.sh:echo "set xtics 0,0.2,1" >> gnutmp.inp
graphirex4base.sh:echo "set ytics ${temin},${tetic},${temax}" >> gnutmp.inp
graphirex4base.sh:echo "set contour" >> gnutmp.inp
graphirex4base.sh:echo "set key 1.0,12" >> gnutmp.inp
graphirex4base.sh:echo "set cntrparam levels disc $r1max,$r1min" >> gnutmp.inp
graphirex4base.sh:echo "splot r1(x,y)" >> gnutmp.inp
graphirex4base.sh:echo "set key 0.8,12" >> gnutmp.inp
graphirex4base.sh:echo "set cntrparam levels disc $r2max,$r2min" >> gnutmp.inp
graphirex4base.sh:echo "splot r2(x,y)" >> gnutmp.inp
graphirex4base.sh:echo "set key 0.6,12" >> gnutmp.inp
graphirex4base.sh:echo "set cntrparam levels disc $noemax,$noemin" >> gnutmp.inp
graphirex4base.sh:echo "splot noe(x,y)" >> gnutmp.inp
graphirex4base.sh:tm=`echo "$tm + $tmincr" |bc -l`
graphirex4base.sh:tes=`echo "$tes + $tesinc" |bc -l`
graphirex4base.sh:done
graphirex4base.sh:gnuplot gnutmp.inp
graphirex4base.sh:# rm gnutmp.inp
graphirex4base.sh:for i in ${filename}_*.ps
graphirex4base.sh:do
graphirex4base.sh:grep -v age $i  > tmp
graphirex4base.sh:echo "showpage" >> tmp
graphirex4base.sh:mv tmp $i
graphirex4base.sh:done
graphirex4base.sh:cd $curro
```

```
zoom.sh:#!/bin/sh
zoom.sh:curro=`pwd`
zoom.sh:echo -n "R1  :"
zoom.sh:read r1
zoom.sh:echo -n "R1 uncertain :"
zoom.sh:read r1incr
zoom.sh:echo -n "R2  :"
zoom.sh:read r2
zoom.sh:echo -n "R2 uncertain :"
zoom.sh:read r2incr
zoom.sh:echo -n "Noe :"
zoom.sh:read noe
zoom.sh:echo -n "Noe uncertain :"
zoom.sh:read noeincr
zoom.sh:echo -n "Tm min :"
zoom.sh:read tm
zoom.sh:tm=`echo "$tm" |bc -l`
zoom.sh:echo -n "Tm max :"
```

```
zoom.sh:read tmfinal
zoom.sh:tmfinal=`echo "$tmfinal " |bc -l`
zoom.sh:echo -n "Tm increment :"
zoom.sh:read tmincr
zoom.sh:tmincr=`echo "$tmincr " |bc -l`
zoom.sh:echo -n "WH :"
zoom.sh:read wh
zoom.sh:echo -n "Te min"
zoom.sh:read temin
zoom.sh:echo -n "Te max :"
zoom.sh:read temax
zoom.sh:echo -n "Output PS Directory :"
zoom.sh:read filename
zoom.sh:tesfinal=`echo "$tmfinal * 100000 " |bc -l`
zoom.sh:tes=`echo "$tm * 100000 " |bc -l`
zoom.sh:tesinc=`echo "$tmincr * 100000 " |bc -l`
zoom.sh:r1max=`echo "$r1 + $r1incr " |bc -l`
zoom.sh:r1min=`echo "$r1 - $r1incr " |bc -l`
zoom.sh:r2max=`echo "$r2 + $r2incr " |bc -l`
zoom.sh:r2min=`echo "$r2 - $r2incr " |bc -l`
zoom.sh:noemax=`echo "$noe + $noeincr " |bc -l`
zoom.sh:noemin=`echo "$noe - $noeincr " |bc -l`
zoom.sh:tetic=`echo " ( $temax - $temin ) / 10 " | bc -l`
zoom.sh:if  [  -d "${filename}_PS" ]
zoom.sh:then
zoom.sh:echo "${filename}_PS exists"
zoom.sh:echo "Maybe files will be overwriten"
zoom.sh:# echo -n "Do you want overwrite it ([y]/n)?"
zoom.sh:# read answer
zoom.sh:else
zoom.sh:mkdir ${filename}_PS
zoom.sh:fi
zoom.sh:# if [ "$answer" = "n" ]
zoom.sh:# then
zoom.sh:# exit 1
zoom.sh:# fi
zoom.sh:cd ${filename}_PS
zoom.sh:echo "#Input para contornos" > gnutmp.inp
zoom.sh:while [ $tes -le $tesfinal ]
zoom.sh:do
zoom.sh:echo "gh = 2.67519e8" >> gnutmp.inp
zoom.sh:echo "gn = -2.711e7" >> gnutmp.inp
zoom.sh:echo "wh = $wh *2.0*3.14159" >> gnutmp.inp
zoom.sh:echo "wn = wh * gn/gh  " >> gnutmp.inp
zoom.sh:echo "tm = $tm" >> gnutmp.inp
zoom.sh:echo "u = 0.519409479 * gh/gn  " >> gnutmp.inp
zoom.sh:echo "tau(y) = (y*tm)/(y+tm)" >> gnutmp.inp
zoom.sh:echo "jac(w,x,y) = (x*tm/(1.0+(w*tm)**2)+(1.0-x)*tau(y)/(1.0+(w*tau(y))**2))" >> gnutmp.inp
zoom.sh:echo "r1(x,y) = (0.519409479)*(jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y))+1.0/3.0*0.4*((-0.16)**2)*wn**2
*1000.0*jac(wn,x,y) " >> gnutmp.inp
zoom.sh:echo "r2(x,y) = (0.2597047039)*(4.0*jac(0.0,x,y)+jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y)+6.0*jac(wh,x,
y))+1.0/18.0*0.4*((-0.16)**2)*wn**2*1000.0*(4.0*jac(0.0,x,y)+3.0*jac(wn,x,y))" >> gnutmp.inp
zoom.sh:echo "noe(x,y)= 1 + u*(6*jac(wh+wn,x,y)-jac(wh-wn,x,y))/r1(x,y)" >> gnutmp.inp
zoom.sh:echo "set term postscript color solid " >> gnutmp.inp
zoom.sh:echo "set output '${filename}_${tm}.ps' " >> gnutmp.inp
zoom.sh:echo "set title 'R1 $r1min $r1max R2 $r2min $r2max HNOE $noemin $noemax Tm ${tm} Wh $wh' " >> gnutmp.inp
zoom.sh:echo "set label '${filename}' at -0.5,${temax}  " >> gnutmp.inp
zoom.sh:echo "set xlabel 'S2' " >> gnutmp.inp
zoom.sh:echo "set ylabel 'Te' " >> gnutmp.inp
zoom.sh:echo "set view 180, 90, 1, 1" >> gnutmp.inp
zoom.sh:echo "set samples 100" >> gnutmp.inp
zoom.sh:echo "set isosamples 100" >> gnutmp.inp
zoom.sh:echo "set nosurface" >> gnutmp.inp
zoom.sh:echo "set contour base" >> gnutmp.inp
zoom.sh:echo "set cntrparam bspline" >> gnutmp.inp
zoom.sh:echo "set xrange [0.6:1]" >> gnutmp.inp
zoom.sh:echo "set yrange [${temin}:${temax}]" >> gnutmp.inp
zoom.sh:echo "set xtics 0,0.2,1" >> gnutmp.inp
zoom.sh:echo "set ytics ${temin},${tetic},${temax}" >> gnutmp.inp
zoom.sh:echo "set contour" >> gnutmp.inp
zoom.sh:echo "set key 1.0,12" >> gnutmp.inp
zoom.sh:echo "set cntrparam levels disc $r1max,$r1min" >> gnutmp.inp
zoom.sh:echo "splot r1(x,y)" >> gnutmp.inp
zoom.sh:echo "set key 0.8,12" >> gnutmp.inp
zoom.sh:echo "set cntrparam levels disc $r2max,$r2min" >> gnutmp.inp
zoom.sh:echo "splot r2(x,y)" >> gnutmp.inp
```

```
zoom.sh:echo "set key 0.6,12" >> gnutmp.inp
zoom.sh:echo "set cntrparam levels disc $noemax,$noemin" >> gnutmp.inp
zoom.sh:echo "splot noe(x,y)" >> gnutmp.inp
zoom.sh:tm=`echo "$tm + $tmincr" |bc -l`
zoom.sh:tes=`echo "$tes + $tesinc" |bc -l`
zoom.sh:done
zoom.sh:gnuplot gnutmp.inp
zoom.sh:# rm gnutmp.inp
zoom.sh:for i in ${filename}_*.ps
zoom.sh:do
zoom.sh:grep -v age $i  > tmp
zoom.sh:echo "showpage" >> tmp
zoom.sh:mv tmp $i
zoom.sh:done
zoom.sh:cd $curro
```

```
formula.inp:#Input para contornos
formula.inp:gh = 2.67519e8
formula.inp:gn = -2.711e7
formula.inp:wn = wh * gn/gh
formula.inp:u = 0.519409479 * gh/gn
formula.inp:tau(y) = (y*tm)/(y+tm)
formula.inp:jac(w,x,y) = (x*tm/(1.0+(w*tm)**2)+(1.0-x)*tau(y)/(1.0+(w*tau(y))**2))
formula.inp:r1(x,y) = (0.519409479)*(jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y))+1.0/3.0*0.4*((-0.16)**2)*wn**2*1
000.0*jac(wn,x,y)
formula.inp:r2(x,y) = (0.2597047039)*(4.0*jac(0.0,x,y)+jac(wh-wn,x,y)+3.0*jac(wn,x,y)+6.0*jac(wh+wn,x,y)+6.0*jac(wh,x,y)
)+1.0/18.0*0.4*((-0.16)**2)*wn**2*1000.0*(4.0*jac(0.0,x,y)+3.0*jac(wn,x,y))+
formula.inp:noe(x,y)= 1 + u*(6*jac(wh+wn,x,y)-jac(wh-wn,x,y))/r1(x,y)
```

```
sample.input:2.68
sample.input:0.06
sample.input:5.19
sample.input:0.62
sample.input:0.60
sample.input:0.02
sample.input:3.66
sample.input:3.86
sample.input:0.1
sample.input:0.5
sample.input:0.0
sample.input:1.5
sample.input:RES46
```

Bibliografía

[1] Celda, B., Widmer, H., Leupin, W., Chazin, W.J., Denny, W.A., y Wüthrich, K. (1989) *Biochemistry* **28**, 1462–1471.

[2] Lesk, A.M. y Chothia, C.H. (1986) *Philos. Trans. R. Soc. London Ser.B.* **317**, 345–356.

[3] Chothia, C.H., Lesk, A.M., Levitt, M., Amit, A.G., Mariuzza, R.A., Phillips, S.E.V. y Poljak, R.J. (1986) *Science* **233**, 755–758.

[4] Needleman, M. y Wunch, M.A. (1982) *Biochemistry* **21**, 747–759.

[5] Woodak, S. y Rooman, M. (1993) *Curr. Opin. Struct. Biol.* **3**, 247–259.

[6] Gross, E.L. (1996) D.R. Ort y C.F. Yocum (Eds.) *Oxygenic Photosynthesis: The Light Reactions* , Kluwer, Dordrecht, 413–429.

[7] Navarro, J.A., Hervás, M. y De la Rosa, M.A. (1997) *J.Biol.Inorg.Chem.* **2** 11-15.

[8] Johnson, M., Srinivasan, N., Sowdhamini, R. y Blundell, T. (1994) *CRC Crit. Rev. Biochem. Mol. Biol.* **29**, 1–68.

[9] Abola, E. Protein Data Bank. (1987) Brookhaven (PDB). Data Comission of the International Union of Cristallography.

[10] Colman, P.M., Freeman, H.C., Guss, J.M., Murata, M., Norris, V.A., Ramshaw, J.A.M., y Venkatappa, M.P. (1978) *Nature (London)* **272**, 319–324.

[11] Badsberg, U., Jorgensen, A.M.M., Gesmar, H., Led, J.J., Hammerstad, J.M., Jespersen, L.L. y Ulstrup, J. (1996) *Biochemistry* **35**, 7021–7029.

[12] Bottin, H. y Mathis, P. (1985) *Biochemistry* **24**, 6453–6460.

[13] Hervás, M., Navarro, J.A., Díaz, A., Bottin, H. y De la Rosa, M.A. (1995) *Biochemistry* **34**, 11321–11326.

[14] Sali, A. y Blundell, T.L. (1993) *J.Mol.Biol.* **234**, 779–815.

[15] Blundell, T.L., Sibanda, B.L. y Pearl, L. (1983) *Nature* **304**, 273–275.

[16] Kabsch,W. y Sander,C. (1983) *Biopolymers* **22**, 2577-2637.

[17] Frishman,D. y Argos,P. (1995) *Proteins: structure, function and genetics* **23**, 566-579.

[18] Schiffer, C.A., Caldweml, J.W., Kollman, P.A. Y Stroud, R.M. (1990) *Proteins* **257**, 961–964.

[19] Overington, J., Donnelly, D., Johnson, M.S., Sâli, A. y Blundell, T.L. (1992) *Protein Sci.* **1**, 216–226.

[20] Sâli, A., Overington, J.P., Johnson, M.S. y Blundell, T.L. (1990) *TIBS* **15**, 235–240.

[21] Sahasrabudhe, P.V., Tejero, R., Kitao, S., Furuichi, Y. y Montelione, G.T. (1998) *Proteins: Structure, Function, Genetics* En prensa.

[22] Bruccoleri, R.E., Haber. E. y Novotny, J. (1988) *Nature* **335**, 564–568.

[23] Bruccoleri, R.E., Haber. E. y Novotny, J. (1988) *Nature* **336**, 1266.

[24] Brooks, B.R., Bruccoleri, R.E., Olafson, B.D., States, D.J., Swaminathan, S. y Karplus, M. (1983) *J.Comput.Chem.* **4**, 187–217.

[25] Bruccoleri, R.E. y Karplus, M. (1987) *Bioplymers* **26**, 137–168.

[26] Li, H., Tejero, R., Monleón, D., Bassolino-Klimas, D., Abate-Shen, C., Bruccoleri, R.E. y Montelione, G.T. (1997) *Prot.Sci* **PONER**, a-b.

[27] Bruccoleri, R.E. (1993) *Mol.Simul.* **10** 151–174.

[28] Esteve, V., Pérez-Paya, E., Tejero, R. y Celda, B. (1998) *EMBO*, enviado.

[29] Redinbo, M.R., Yeates, T.O., Merchant, S. (1994) *J.Bioenerg.Biomembr.* **26** 49–54.

[30] Bagby, S. Driscoll, P.C., Harvey, T.S. y Hill, A.O. (1994) *Biochemistry* **33** 6611–6622.

[31] Koradi, R., Billeter, M., y Wüthrich, K. (1996) *J. Mol. Graphics* **14**, 51–55.

[32] Aue, W.P., Bartholdi, E. y Ernst, R.R. (1975) *J. Chem. Phys.* **64**, 2229-22241.

[33] Jeener, J. (1971) *Ampère Int. Summr School*, Basko Polje, Yugoslavia.

[34] Bax, A. y Freeman, R. (1981) *J.Magn.Reson.* **44**, 542–561.

[35] Shaka, A.J. y Freeman, R. (1983) *J.Magn.Reson.* **51**, 169–173.

[36] Müller, N., Ernst, R.R. y Wüthrich, K. (1986) *J.Am.Chem.Soc.* **108**, 6482–6492.

[37] Wüthrich, K. (1986) J. Wiley e hijos (Eds.) *NMR of Proteins and Nucleic Acids*

[38] Montelione, G.T., Lyons, B.A., Emerson, S.D., Tashiro, M. J. (1992) *J.Amer.Chem.Soc.* **114**, 10974–10975.

[39] Güntert, P., Braun, W. y Wüthrich, K. (1991) *J.Mol.Biol.* **217** 517–530.

[40] Braun, W. y Gö, N. (1985) *J.Mol.Biol.* **186**, 611–626.

[41] Güntert, P., Berndt, K.D.. y Wüthrich, K. (1993) *J.Biomol.NMR* **3**, 601–606.

[42] Polshakov, V.I., Frenkiel, T.A., Birdsall, B., Soteriu, A. y Feeney, J. (1995) *J.Magn.Reson. Ser. B* **108**, 31–43.

[43] Jacoby, S.L.S., Kowalik, J.S. y Pizzo, J.T. (1972) *Iterative Methods for Nonlinear Optimization Problems* . Englewood Cliffs (Eds.) Prentice Hall.

[44] Fletcher, R. (1972) *Conjugate Direction Methods* en Numerical Methods fo Unconstrained Optimization, Lond Academic (Eds.), 73–87.

[45] Fletcher, R. y Reeves, C.M. (1964) J.Wiley (Eds.) *Practical Methods of Optimization*

[46] Havel, T.F. y Snow, M.E. (1991) *Mol.Sim.* **10** 175–210.

[47] QUANTA 4.0, Mol.Sim. Inc. 16 New England Executive Park, Burlington, MA, 01803-5297, USA.

[48] Tejero, R., Bassolino-Klimas, D., Bruccoleri, R.E. y Montelione, G.T. (1996) *Protein Sci.* **5**, 572–592.

[49] Bassolino-Klimas, D., Tejero, R., Krsystek, S.R., Metzler, W.J., Bruccoleri, R.E. y Montelione, G.T. (1996) *Protein Sci.* **5**, 593–603.

[50] Hervás, M., Navarro, F., Navarro, J.A., Chávez, S., Díaz, A., Florencio, F.J. y De la Rosa, M.A. (1993) *FEBS* **319**, 257–258.

[51] Marion, D., y Wüthrich, K. (1983) *Biochem. Biophys. Res. Commun.* **113**, 967–968.

[52] Piotto, M., Sandek, V. y Sklenár, V. (1992) *J.Biomol.NMR* **2**, 661–669.

[53] Wüthrich, K., Billeter, M. y Braun, W. (1983) *J.Mol.Biol.* **169**, 949–961.

[54] Montelione, G.T., Winkler, M.E., Rauenbühler, P. y Wagner, G. (1989) *J.Magn.Reson* **82**, 198–208.

[55] Guss, J.M., Harrowell, P.R., Murata, M., Norris, V.A. y Freeman, H.C. (1986) *J.Mol.Bio.* **192**, 361–387.

[56] Moore, J.M., Lepre, C.A., Gippert, G.P., Chazin, W.J., Case, D.A. y Wright, P.E. (1991) *J.Mol.Biol* **221**, 533–555.

[57] Delsuc M.A. y Levy,G.C. (1988) *J.Magn.Reson.* **76**, 06–315

[58] Press, W.H., Flannery, B.P., Teukolsky, S.A. y Vetterling, W.T. (1986) *Numerical Recipes*, Cambridge University Press, Cambridge.

[59] Adobe Systems Inc. (1985). PostScript Language Reference Manual. Addison-Wesley, Reading, MA.

[60] Güntert, P. Qian, Y.Q., Otting, G., Müller, M., Gehring, W. y Wüthrich, K. (1991) *J.Mol.Biol.* **217**, 531–540.

[61] Güntert, P. y Wüthrich, K. (1991) *J.Biomol.NMR* **1**, 447–456.

[62] Wishart, D.S., Sykes, B.D y Richards, F.M. (1992) *Biochemistry* **31**, 1647–1654.

[63] Croasmun y Carlson R.M.K. (1994) *Two-Dimensional NMR Spectroscopy*, VCH, New York.

[64] Celda, B. y Montelione, G.T. (1993) *J.Magn.Reson.* **B101**, 189–196.

[65] Constantine, K.L., Friedrichs, M.S. y Mueller, L. (1994) *J.Magn.Reson.* **B104**, 62–70.

[66] Cavanagh, J., Chazin, W.J. y Rance, M. (1990) *J.Magn.Reson.* **87**, 110–118.

[67] Engh, R.A. y Huber, R. (1991) *Acta Cryst.* **A47**, 392–400.

[68] Laskowski, R.A., MacArthur, M.W., Moss, D.S. y Thornton, J.M. (1993) *J.Appl.Cryst* **26**, 283–291.

[69] Kabsch, W. y Sander, C. (1983) *Biopolymers* **22**, 2577–2637.

[70] Morris, A.L., MacArthur, M.W., Hutchinson, E.G. y Thornton, J.M. (1992) *Protein* **12**, 345–364.

[71] Nishikawa, K. y Ooi, T. (1986) *J.Biochem.* **100**, 1043–1047.

[72] Moore, J.M., Case, D.A., Chazin, W.J., Gippert, G.P., Havel, T.F., Powls, R. y Wright, P.E. (1988) *Science* **240**, 314–320.

[73] Collyer, C.A., Guss, J.M., Sigimura, Y., Yoshikazi, F. y Freeman, H.C. (1990) *J.Mol.Biol* **211**, 617–632.

[74] Brooks, C.L.III., Karplus, M. y Pettitt, B.M. (1988) *Proteins: A Theoretical Perspective of Dynamics, Structura and Thermodynamics* , Wiley, New York.

[75] Barnes, J.E. (1989) *Nature* **338**, 123–126.

[76] Karplus, M. y Petsko, G.A. (1990) *Nature* **347**, 631–639.

[77] Mumenthaler, C. y Braun, W. (1995) *J.Mol.Modelling* **1** 1–10.

[78] Case, D.A. y Karplus, M. (1979) *J.Mol.Biol.* **132**, 343–368.

[79] Harvey, S. (1989) *Proteins* **5**, 78–92.

[80] Celda, B., Biamonti, C., Arnau, M.J., Tejero, R. y Montelione, G.T. (1995) *J.Biomol.NMR.* **5**, 161–172.

[81] Borgias, B.A. y James, T.L. (1990) *J.Magn.Reson.* **87**, 475–487.

[82] Kline, T.P., Brown, F.K., Brown, S.C., Jeffs, P.W., Kopple, K.D. y Mueller, L. (1990) *Biochemistry* **29**, 7805–7813.

[83] Montelione, G.T., Wüthrich, K., Nice, E.C., Burgess, A.W. y Scheraga, H.A. (1987) *Proc.Natl.Acad.Sci. USA* **84**, 5226–5230.

[84] Bull, T.E., Norne, J.E., Reimarson, P. y Lindman, B. (1978) *J.Am.Chem.Soc.* **100**, 4643–4650.

[85] Lipari, G. y Szabo, A. (1980) *Biophys.J.* **30**, 489–496.

[86] Wittebort, R.J. y Szabo, A. (1978) *J.Chem.Phys.* **69**, 1722–1740.

[87] Abragam, A. (1961) *The Principles of Nuclear Magnetism*, Clarendon Press (Eds.) Oxford.

[88] Richarz, R., Nagayama, K. y Wüthrich, K. (1980) *Biochemistry* **19**, 5189–5201.

[89] Lipari, G. y Szabo, A. (1982) *J.Am.Chem.Soc* **104**, 4546–4570.

[90] Arnau, M.J. (1998) *Tesis Doctoral*, Universitat de València, Dpto. Química Física.

[91] Clore, G.M., Szabo, A., Bax, A., Kay, L.E., Driscoll, P.C. y Gronenborn, A.M. (1990) *J.Am.Chem.Soc.* (1990) **112**, 4989–4991.

[92] Palmer, A.G., Skelton, N.J., Chazin, W.J., Wright, P.E. y Rance, M. (1992) *Mol.Phys.* **75**, 699–711.

[93] Otting, G, Lipinsh, E. y Wüthrich, K. (1993) *Biochemistry* **32**, 3571–3582.

[94] Parkin, S., Rupp, B. y Hope, H. (1993) *Biochemistry* **32**, 1221–1229.

[95] Smith, P.E., Van Schaik, R.C., Szyperski, T., Wüthrich, K. y Van Gunsteren, W.F. (1995) *J.Mol.Biol.* **22** 356–365.

[96] Jansson, M., Li, Y.C., Jenderberg, L., Anderson, S., Montelione, G.T. y Nilsson, B. (1995) *J.Biomol.NMR* **5**, 330–338.

[97] Jansson, M., Li, Y.C., Jenderberg, L., Edlund P., Anderson S., Montelione, G.T. y Nilsson, B. (1994) *J.Biomol.NMR* **4**, 220–230.

[98] Altman, J.D., Henner, D., Nilsson, B. Anderson, S. y Kuntz, I.D. (1991) *Protein Engineering* **4**, 593–600.

[99] Shaka, A.J., Barker, P.B. y Freeman, R. (1985) *J.Magn.Reson.* **64**, 547–560.

[100] Kay, L.E., Nicholson, L.K., Delaglio, F. Bax, A. y Torchia, D.A. (1992) *J.Magn.Reson.* **97**, 359–375.

[101] Kay, L.E., Torchia, D.A. y Bax, A. (1989) *Biochemistry* **28**, 8972–8984.

[102] Li, Y.C. y Montelione, G.T. (1993) *J.Magn.Reson.* **101**, 315–327.

[103] Palmer, A.G., Rance, M. y Wright, P.E. (1991) *J.Am.Chem.Soc.* **113**, 4371–4389.

[104] Mandel, A.M., Akke, M. y Palmer, A.G. (1995) *J.Mol.Biol.* **246**, 144–163.

[105] Jin, D., Figueirido, F., Montelione, G.T. y Levy, R.M. (1997) *J.Am.Chem.Soc.* **119**, 6923–6924.

[106] Wagner, G. (1993) *Curr.Opin.Struct.Biol.* **3**, 748–754.

[107] Lefèvre, J., Dayie, K.T., Peng, J.W. y Wagner, G. (1996) *Biochemistry* **35**, 2674–2686.

[108] Castro, M. J. M., Biamonti, C., Celda, B., Anderson, S. y Montelione, G. T. (1997) *J.Mol.Biol.* **269**, 592–610.

[109] Li, Y.C., y Montelione, G.T. (1995) *Biochemistry* **34**, 2408–2423

[110] Campbell, F. y Bork, A. (1993) *Biochemistry* **32**, 2401–2423

[111] Li, Y.C. (1995) *Tesis Doctoral*. Rutgers University. CABM. New Jersey, USA.

[112] Güntert, P., Braun, W., Billeter, M. y Wüthrich, K. (1989) *J.Am.Chem.Soc.* **111**, 3997–4004.

[113] Güntert, P., Mummenthaler, C. y Wütrich, K (1997) *J.Mol.Biol.* **273**, 283–298.

[114] Bax, A. Vuister, G.W., Grzesiek, S. , Delaglio, F. Wang, A.C., Tshudin, R. y Zhu, G. (1994) *Nulcear Magnetic Resonance*, James, T.L. y Oppenheimer, N.L. (Eds.) Ac.Press, San Diego, CA.

[115] Biamonti, C., Rios, C.B., Lyons, B.A. y Montelione, G.T. (1994) *Adv. in Biophys.Chem.* **4**, 51–120.

[116] Horta, A. (1992) *Macromoléculas.*

[117] Flory, P.J. (1969) *Statistical Mechanics of Chain Molecules*, John Wiley (Eds.), New York.

[118] Leach, S.J., Némethy, G. y Scheraga, H.A. (1977) *Biochem.Biophys.Res.Commun.*, **75**, 207–215.

[119] Clore, G.M., Bax, A. y Gronenborn, A.M. (1991) *J.Biomol.NMR* **1**, 13–22.

[120] Polshakov, V.I., Frenkiel, T.A., Birdsall, B., Soteriou, A. y Feeney, J. (1995) *J.Magn.Reson.Ser.B* **108**, 31–43.

[121] Gibrat, J.F., Robson, B. y Garnier, J. (1991) *Biochemistry* **30**, 1578–1586.

[122] Kuszewsky, J., Qin, J., Gronenborg, A. y Clore, G.M. (1995) *J.Magn.Reson.Ser.B* **106**, 92–96.

[123] IUPAC-IUB. (1970) *J.Mol.Biol.* **52**, 1–17.

[124] Kollman, P.A., Weiner, P.K. y Dearing, A. (1981) *Biopolymers* **20**, 2583–2604.

[125] Weiner, S.J., Kollman, P.A., Nguyen, D.T. y Case, D.A. (1986) *J.Comp.Chem.* **7**, 230–252.

[126] Donaire, A., Jiménez, H.R., Moratal, J.M., De la Rosa, M.A., Hervás, M., Navarro, J.A., Monleón, D., Tejero, R. y Celda, B. (1998) *Inorg.Chem.Acta* **275-276**, 73–89.

[127] Montelione, G.T., Emerson, S.D. y Lyons, B.A. (1992) *Biopolymers* **32**, 327–334

[128] Montelione, G.T., Wüthrich, K., Burgens, A.W., Nice, E.C., Wagner, G., Gibson, K.D. y Scheraga, H.A. (1992) *Biochemistry* **31**, 236–

[129] Cavanagh, J, Chazin, W.J. y Rance, M. (1990) *J.Magn.Reson.* **87**, 110–123.

[130] Kessler,H., Gehrke, M. y Griesinger, C. (1988) *Angew.Chem.* **27**, 490–536.

[131] Jeener, J. Meier, B.H., Bachmannnm, P. y Ernst, R.R. (1979) *J.Chem.Phys.* **71**, 4546–4558.

[132] Hurd, R.E. (1990) *J.Magn.Reson.* **87**, 422–428.

[133] Tashiro, M., Tejero, R., Zimmerman, D.E., Celda, B., Nilsson, B. y Montelione, G.T. (1997) *J.Mol.Biol.* **272**, 573–590

[134] Celda, B. y Montelione, G.T. *J.Magn.Reson.* **B101** 189–193

[135] Moy, F.J., Scheraga, H.A., Lui, J.-F., Wu, R. y Montelione, G.T. *Proc.Natl. Acad.Sci. USA.* **86** 9836–9840

[136] InsightII, Biosym / MSI, 9685 Scranton Road, San Diego, CA 92121-3752, USA.

[137] Wlodawer, A., Svensson, L.A., Sjolin, L. y Gilliland, G. (1988) *Biochemistry* **27**, 2705–2717.

[138] Neri, D., Otting, G. y Wüthrich, K. (1990) *J.Am.Chem.Soc.* **112**, 3663–3665.

[139] Kuboniwa, H., Grzesiek, S., Delaglio, F. y Bax, A. (1994) *J.Biomol.NMR* **4**, 871–878.

[140] Montelione, G.T., Winkler, M.E., Rauenbuehler, P. Y Wagner, G. (1989) *J.Magn.Reson.* **82**, 198–204.

[141] Constantine, K.L. y Friedrichs, M.S. (1994) *J.Magn.Reson.Ser.B* **104**, 62–68.

[142] Watson, J.D. y Crick, F.H. (1953) *Nature* **171**, 737–738.

[143] Dickerson, R.E. (1983) *J.Mol.Biol.* **166**, 419–441.

[144] Drlica, K. (1990) *Mol.Microbiol.* **6**, 425–433.

[145] Malinina, L., Urpí, L., Salas, X., Tam H-D. y Subirana, J.A. (1994) *J.Mol.Biol.* **243**, 484–493.

[146] Monleón, D. (1994) *Tesis de Licenciatura*. Universitat de València. Dpto. Química Física.

[147] Martínez, M.C. (1997) *Tesis de Licenciatura*. Universitat de València. Dpto. Química Física.

[148] Sinden, R.R. (1994) *DNA Structura and Function*, Ac.Press (Eds.)

[149] Kabcsh, W., Sander, C. y Trifonov, E.N. (1982) *Nucleic Acid Res.* **10**, 1097–1104.

[150] Gessner, R.V., Frederick, C.A., Quigley, G.J., Rich, A. y Wang, A.H. (1989) *J.Biol.Chem.* **264**, 7921–7935.

[151] Jupe, E.R., Sinden, R.R. y Cratwright, I.L. (1993) *EMBO J.* **12**, 1067–1075.

[152] Sheth, A., Ravikumar, M., Hosur R.V. y Girjesh, G. (1987) *Biopolymers* **26**, 1301–1313.

[153] Dickerson, R.E. (1992) *Methods Enzymol.* **211**, 67–111.

[154] Dai, Z., Dauchez, M., Thomas, G. y Peticolas, W.L. (1992) *J.Mol.Struct.Dynam.* **9**, 1155–1183.

[155] Panayotatos, N. y Fontaine, A. (1987) *J.Biol.Chem.* **262**, 11364–11368.

[156] Hasnoot, C.A.G., Leeuw, F.A.A.M. y Altona, C. (1980) *Tetrahedron* **36**, 2783–2792.

[157] Chazin, W.J., Wüthrich, K., Rance, M. Hyberts, S. Denny, W.A. y Leupin, W. (1986) *J.Mol.Biol.* **190**, 439–453.

[158] Widmer, H. y Wüthrich, K. (1987) *J.Magn.Reson.* **74**, 316–336.

[159] Morris, A.L., M.W., Hutchinson, E.G. y Thornton, J.M. (1992) *Protein* **12**, 365–382.

[160] Kelley, C. y Williams, T. (1993) GNUPLOT, Versión 3.5 para Unix. Free Software Foundation, Inc.

[161] Tejero, R. (1994) PDBSTAT, Version 3.4 para Unix.

[162] GRASP. Modulo de MOLMOL version 2.5. Koradi, R., Billeter, M., y Wüthrich, K. (1996).

[163] Sayle, R. (1995) RASMOL Version 2.6 *Biomolecular Structure* Glaxo Res. Dev., Greenford, Middlesex, UK.

[164] Navarro, E. (1997) *Tesis de Licenciatura*. Universitat de València. Dpto. Química Física.

[165] Tejero, R., Monleón, D., Medina, P. y Celda, B. (1995) *Protein Sci.*, **4**, suppl., 1, 57–63.

[166] Altona, C. y Sundaralingam, M. (1973) *J.Am.Chem.Soc.* **95**, 2333–2344.

[167] Altona, C. y Sundaralingam, M. (1972) *J.Am.Chem.Soc.* **94**, 8205–8212.

[168] Doménech, A., García-España, E., Ramírez, J.A., Celda, B. Martínez, M.C., Monleón, D., Tejero, R., Bencini A. y Bianchi, A. (1998) *J. Am.Chem.Soc.Perkin Trans* , en prensa.

[169] Saenger, W. (1984) *Principles of Nucleic Acid Structure*, Springer, New York.

[170] Wakelin, L.P.G. y Wanning, M.J. (1976) *Biochem.J.* **157**, 721–730.

[171] McGhee, J.P. y Von Hippel, P.H. (1974) *J.Mol.Biol.* **86**, 469–481.

[172] Widmer, H. y Wütrich, K. (1986) *J.Magn.Reson.* **70**, 270–295.

[173] Discover, Biosym / MSI, 9685 Scranton Road, San Diego, CA 92121-3752, USA.

[174] Esteve, V. (1996) *Tesis de Licenciatura*. Universitat de València. Dpto. Química Física.

[175] Tejero, R., Monleón, D., Celda, B. y Montelione, G.T. (1998) En prensa.

[176] Dickerson, R.E. (1993) Manual de Instrucciones NEWHEL93.

[177] Haschemeyer, A.E.V. y Rich, A. (1967) *J.Mol.Biol.* **27**, 369–384.

[178] Rinkel, L.J. y Altona, C. (1987) *J.Biomol.Struct.Dynam.* **4**, 621–649.

[179] Monleón, D., Martínez, M.C., Esteve, V. y Celda, B. (1997) *Spectroscopy of Biological Moleculas: Modern Trends* Kluwer Ac.Publ., P.Carmona, R.Navarro y A.Hernanz Eds. 237–238.

Índice de Figuras

Índice de Tablas