Facultad de Ciencias Bioquímicas y Farmacéuticas
Universidad Nacional de Rosario

“Estructura de complejos dsRBD:pri-miRNA a través de restricciones obtenidas por etiquetas paramagnéticas y fluorescentes”

Tesis para optar el título de Doctor en Ciencias Biológicas

Lic. Florencia Carla Mascali
Director: Dr. Rodolfo M. Rasia

2019

“Estructura de complejos dsRBD:pri-miRNA a través de restricciones obtenidas por etiquetas paramagnéticas y fluorescentes”

Florencia C. Mascali
Licenciada en Biotecnología
Universidad Nacional de Rosario

Esta Tesis es presentada como parte de los requisitos para optar al grado académico de Doctor en Ciencias Biológicas de la Universidad Nacional de Rosario y no ha sido presentada previamente para la obtención de otro título en esta u otra Universidad. La misma contiene los resultados obtenidos en investigaciones llevadas a cabo en el Instituto de Biología Molecular y Celular de Rosario (IBR-CONICET-UNR), durante el período comprendido entre el 1ro de abril de 2014 y el 31 de enero de 2019, bajo la dirección del Dr. Rodolfo M. Rasia.

AGRADECIMIENTOS

Al CONICET, a la ANPCyT y al programa ECOS por el apoyo económico.
Al IBR por el espacio y por la preocupación constante hacia el crecimiento profesional de todos.
A Fito, por confiar en mí para este trabajo de tesis, y por enseñarme y ayudarme en este camino de formación. Gracias por darme la confianza para hacer preguntas y por permitirme siempre exponer lo que pensaba y formar mi opinión con criterio.
A Leandro por dirigirme en su laboratorio y colaborar con mi proyecto.
A los Fitomejoradores por su buena onda y compañía. A Nah por estar predispuesto a resolver cualquier problema que surgía, por las correcciones en mis trabajos, por enseñarme a mirar desde otros puntos de vista y por hacerme reír tanto. A Robi por ser mi compañera todos estos años, forjando una amistad que trascendió siempre lo laboral.
A todos los que pasaron por el lab 5, me llevo hermosos recuerdos vividos con ustedes y amistades que perdurarán. A la Carli y la Cucky por las charlas dentro del lab y las cervezas afuera. A Pauli H., Solci y Anita, por demostrarme día a día que las distancias no importan.
A las Nerds, por seguir atravesando etapas juntas.
A mis padres, por brindarme su apoyo y amor incondicional. A mi hermana mayor Noe que me enseña a nunca dejar de buscar la felicidad.
Al “Rejunte”, por estar siempre.

A Leo, por el aguante, por la compañía y por el amor.

aquí y ahora

Parte de los trabajos realizados en esta Tesis han sido presentados en la siguiente publicación:

Mascali, F.C.; Ching, H.Y.; Rasia, R.M.; Un, S.; Tabares, L.C. (2016) Using Genetically Encodable Self-Assembling Gd(III) Spin Labels To Make In-Cell Nanometric Distance Measurements. *Angew Chem Int Ed Engl.* 55, 11041-11043.

ÍNDICE

ABREVIATURAS Y ANGLICISMOS

ADN	Ácido desoxirribonucleico
ARN	Ácido ribonucleico
ARNi	ARN de interferencia
ARNdh	ARN doble hebra
cluster	Grupo
dsRBD	double strand RNA Binding Domain - Dominio de unión a ARNdh
EPR	Electron Paramagnetic Resonance - Resonancia Paramagnética Electrónica
His-tag	Etiqueta de histidinas
HSQC	Heteronuclear Single Quantum Coherence- Experimento de coherencia cuántica simple heteronuclear
HYL1	Hyponastic Leaves 1
LB	Luria-Bertani
LBT	Lanthanide Binding Tag - Etiqueta de unión a lantánidos
linker	Secuencia de enlace
M9	medio mínimo M9
mARN	ARN mensajero
miARN	microARN
min	minuto/s
nt	nucleótido/s
ON	toda la noche
parse	análisis de la estructura de un archivo
pb	par/es de base/s
PCR	Polymerase Chain Reaction - Reacción en cadena de la polimerasa
PELDOR	Pulsed Electron-Electron Double Resonance
PRE	Paramagnetic Relaxation Enhancement - Incremento paramagnético de la relajación
Pol II	ARN polimerasa II
RMN	Resonancia Magnética Nuclear
s	segundo/s

RESUMEN

Los miARNs son moléculas de ARN pequeñas que están involucradas en la regulación de procesos fundamentales como el desarrollo, resistencia a estrés y respuestas a hormonas. Su biogénesis comienza con la transcripción de fragmentos más largos que son procesados en forma precisa por la maquinaria de procesamiento. Estos precursores son sumamente heterogéneos en plantas y el modo de reconocimiento de la maquinaria de procesamiento aún no ha sido resuelto. Por ello resulta interesante estudiar, desde una perspectiva estructural, la participación de las proteínas que forman parte del complejo. En particular, este trabajo de Tesis se centra en la proteína de procesamiento de miARN HYL1 de *Arabidopsis thaliana*.

HYL1 posee dos dominios de unión a ARN doble hebra (dsRBD) unidos por un *linker*. A partir de una búsqueda bioinformática se analizó la conservación de tamaño y de secuencia del *linker* en proteínas que contienen dobles dsRBDs en distintas especies. Se pudo concluir que el largo del *linker* está muy conservado en especies plantas, en contraste con lo que se observa en especies del reino animal.

Para comprender el rol de HYL1 dentro del complejo se planteó estudiar la distribución espacial que podrían presentar sus dominios de unión a ARN cuando están libres y/o unidos a ARN. En forma experimental se llevaron a cabo estudios que dan medidas de distancia en escala de nanómetros, en el rango de distancias que separan los dominios. Estas medidas se utilizaron luego en la selección de estructuras probables dentro de un conjunto de estructuras posibles generadas *in silico*. Las medidas fueron obtenidas a partir de técnicas de resonancia magnética sobre muestras que contienen la etiqueta LBT (Lanthanide Binding Tag). Se midió Pulsed Electron DOuble Resonance (PELDOR) entre las sondas para estimar la distancia entre dominios, tanto en forma libre como unida a ARN. La calidad de los datos obtenidos no permitió modelar con precisión la posible distribución de distancias. Por otro lado, se realizaron experimentos de Relajación Paramagnética Electrónica (PRE) midiendo la relajación producida en cada núcleo, la cual está relacionada con la distancia entre dichos núcleos y la sonda. Para la generación de estructuras posibles se trabajó con los programas *Modeller* y *Rosetta*, utilizando como molde estructuras cristalográficas disponibles en bases de datos. Con las medidas experimentales se seleccionó una subpoblación de estructuras que ilustra la dinámica entre los dominios cuando están libres utilizando código generado con el lenguaje *Python*. Las estructuras seleccionadas mostraron una notable cercanía entre sí, en donde los dos dominios se acercan por una de sus caras. Los experimentos sugieren que los dos dominios dsRBDs de

HYL1 se mueven libremente, y que la longitud y posición del *linker* producen una asimetría en la libertad conformacional que estaría dada por restricciones geométricas simples.

1. MARCO TEÓRICO

1.1. Biogénesis de miARNs de plantas

1.1.1. Introducción y función de los miARNs en plantas

El silenciamiento por ARN, también conocido como ARN de interferencia (ARNi), es un mecanismo molecular fundamental para regular la expresión génica que está conservado en la mayoría de los eucariotas. El mismo es desencadenado por una molécula de ARN pequeña, que puede ser un ARN pequeño de interferencia (siARN) o un microARN (miARN) dependiendo de su origen y de los procesos en los que participa [1].

Los miARNs son una clase de ARNs pequeños, endógenos y no codificantes, de 20 - 24 nt de largo, que regulan la expresión génica en forma postranscripcional a través de complementariedad de secuencias [2]. Desde su descubrimiento, se han identificado miles de miARNs en diversos animales y plantas, al principio mediante clonado de sus secuencias y, posteriormente, a través de análisis computacionales de genomas completos. Al presente hay anotados 38589 miARNs (miRBase, http://www.mirbase.org/ octubre de 2018). La importancia de los mismos radica en que participan en casi todos los procesos biológicos [3]. En plantas son esenciales para numerosos procesos, como el crecimiento y desarrollo de hojas, flores, tallos y semillas. También están involucrados en la respuesta a estrés por sequía, temperatura, salinidad, escasez de nutrientes, bacterias y virus, entre otros.

1.1.2. Biogénesis de miARNs en plantas

La generación de los dúplex miARN/miARN* en plantas ocurre dentro del núcleo en compartimentos especializados y sin membrana llamados Dicing-bodies (cuerpos-D). Existen entre uno y cuatro de estos cuerpos por célula. La biogénesis canónica (Figura 1) comienza con la transcripción de los genes de miARNs (MIR) por la ARN Polimerasa II (Pol II). Diferentes activadores transcripcionales y varios motivos de secuencia en los promotores de los MIR participan en el reclutamiento de la Pol II a los promotores [4].

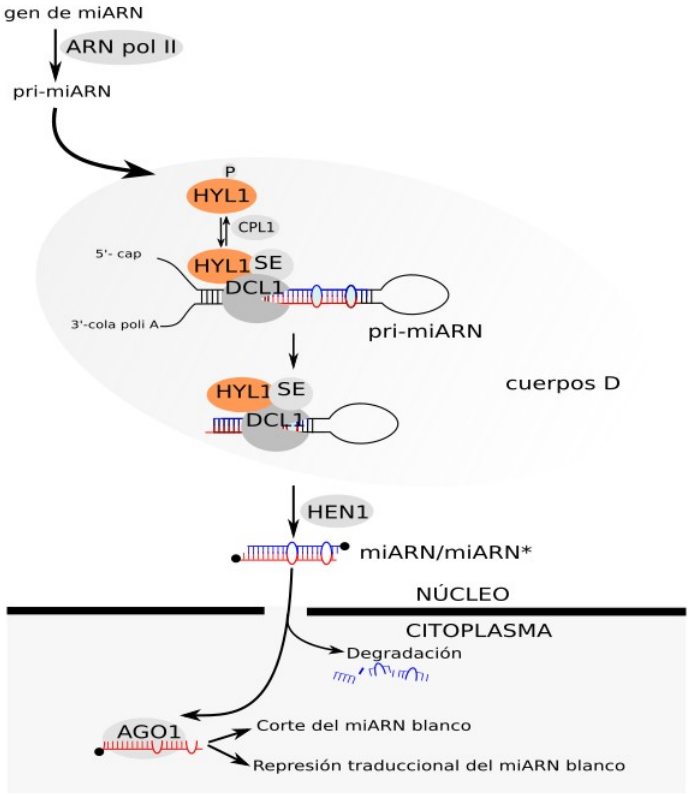


Figura 1: Procesamiento de miARNs en plantas. En color naranja se muestra la proteína HYL1, objeto de estudio de este trabajo de Tesis. En color rojo se muestra la hebra de miARN guía

Los transcriptos primarios sintetizados por la polimerasa, llamados pri-miARNs, poseen forma de hebilla con un tallo doble hebra imperfecto y un bucle terminal. Sus extremos sufren modificaciones postranscripcionales, como la incorporación de caperuzas en sus extremos 5' y la poliadenilación de sus extremos 3' [5].También son blanco de procesos de *splicing* alternativo, particularmente en la región 3' adyacente a la estructura de la hebilla [1,6].

La estructura en forma de hebilla que presentan los pri-miARNs es reconocida en plantas por los miembros de la familia de enzimas Dicer-like (DCL). La cantidad de miembros de la familia DCL varía entre diferentes especies de plantas. Por ejemplo, la familia DCL en Arabidopsis y uva tiene cuatro miembros, en arroz y sorgo hay cinco miembros, y solamente tres en el musgo *Physcomitrella patens* [7]. Dentro de la familia DCL de Arabidopsis, DCL1 es la enzima principal del procesamiento de precursores de miARNs. En cambio, DCL2, DCL3 y DCL4 producen varios tipos de ARNs pequeños de interferencia (siARNs), incluyendo siARNs endógenos, así como virales o transgénicos. DCL1, junto con otras proteínas como HYPONASTIC LEAVES1 (HYL1) y SERRATE (SE) forman el complejo de procesamiento de miARNs. Este complejo realiza dos cortes sucesivos sobre el

pri-miARN de un modo principalmente *base-to-loop*. Existen, sin embargo, formas alternativas de procesamiento que incluyen digestiones *loop-to-base* [8]. El primer corte se produce a 15-17 nt de la base y da lugar al pre-miARN correspondiente, mientras que el segundo libera el miARN/miARN* maduro [9]. Las proteínas DCL parecen funcionar como reglas moleculares que miden y cortan los pares de pequeños ARNs en longitudes específicas. La estructura molecular de las proteínas DCL, en particular la distancia entre los dominios protéicos RNAsa III y PAZ, serían los determinantes de esa especificidad para el largo de los productos [4]. Es así que los distintos tipos de DCL generan productos de diferente largo (21 nt para DCL1, 22 nt para DCL2, 24 para DCL3 y 21 para DCL4).

Los dúplex miARN/miARN* son modificados por metilación en su extremo 3' por la proteína HEN1 y exportados al citoplasma. Esta modificación es crucial para proteger al miARN de la acción de exonucleasas [10]. Una vez en el citoplasma, el par miARN/miARN* se separa y la hebra guía es cargada en el complejo de silenciamiento inducido por ARN (RISC) a través de la unión con la proteína Argonauta (AGO). Una vez ensamblado el complejo, los mARN blancos se unen a través de complementariedad de secuencias con la hebra de miARN maduro para dirigir la degradación del mARN o la inhibición traduccional [4].

1.1.3. Dominios dsRBDs

En el procesamiento de miARNs participan cuatro **dominios de unión a ARN doble hebra (dsRBDs)**, que tienen la particularidad de reconocer variantes de segmentos de ARN doble hebra [11]. Estos dominios cumplen importantes funciones tanto para el reclutamiento del sustrato como para el posicionamiento correcto de los sitios activos. Además de su capacidad de unir ARN, median interacciones proteína-proteína. Están formados por 65-70 residuos aminoacídicos, y se encuentran presentes en eucariotas, procariotas y en proteínas virales. Los dominios dsRBDs adoptan un plegamiento común α-β-β-β-α, donde las dos hélices α se apoyan sobre una misma cara de la lámina β. La interacción que se produce entre estos dominios y el ARN doble hebra ha sido caracterizada mediante la resolución de estructuras de complejos dsRBD-dsARN [12–15]. En las estructuras se pueden identificar tres regiones de unión: la primera corresponde a la hélice 1 que hace contacto con el surco menor del ARN (región 1), la segunda corresponde al bucle 2 e interacciona con el surco menor de la misma cara de la hélice que la región 1 (región 2) y la tercera al bucle 4 y parte de la hélice 2, que hace contacto con el surco mayor (región 3). Las regiones 1 y 2 interactúan con OH 2' de las ribosas y la región 3 con el esqueleto de fosfatos (Figura 2) [16]. Esta forma de unión explica la especificidad por ARN doble hebra (ARNdh), pero no alcanza para explicar el modo de reconocimiento de sustrato en la reacción de corte, considerando fundamentalmente que los precursores no comparten secuencia.

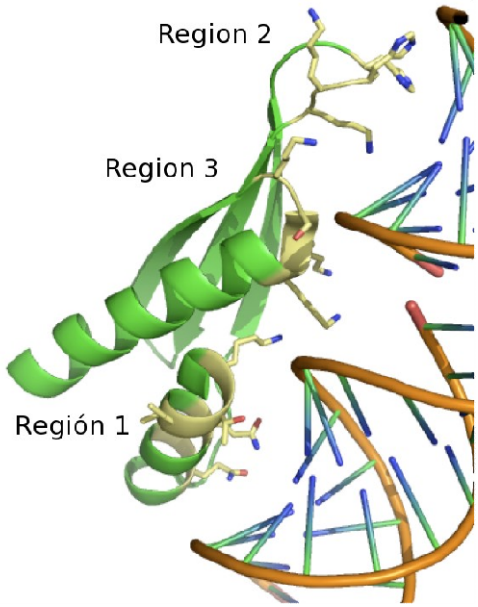


Figura 2: Estructura de un dominio dsRBD. Las regiones que interactúan con el ARNdh se muestran en color amarillo. (PDB 1di2)

1.1.4. Proteínas DRB

Las proteínas que contienen únicamente dominios dsRBDs, también llamadas DRBs, participan en distintos procesos, desde localización y transporte de mARN hasta maduración y degradación de ARNs, respuestas virales y traducción de señales. Entre las proteínas relacionadas a mecanismos de ARNi existen distintas proteínas cofactores auxiliares que asisten a RNAsas. En vertebrados, por ejemplo, DROSHA es asistida por DGCR8 para generar pre-miARNs en el núcleo, mientras que TRBP y PACT (o sus ortólogos Loqs y R2D2 en moscas, y RDE4 en gusanos) interactúan con Dicer en el procesamiento de pre-miARNs o ANRdh largos en fragmentos de 21 nt. De forma similar, en plantas HYL1 interacciona con DCL1, mientras que otras proteínas DRBs interaccionan con otras proteínas tipo Dicer, formando los complejos que procesan distintos ARNs pequeños [17].

La arquitectura de las proteínas auxiliares está bastante conservada, con dos o tres dsRBDs organizados en tándem. En todos los casos, el primero o los dos primeros dsRBDs unen ARNdh con una alta afinidad, mientras que el último dsRBD es diferente e interacciona con otras proteínas, o participa en su dimerización. En ausencia de ARN, los dominios individuales pueden moverse libremente, interactuar, adoptar conformaciones pre-formadas que se asemejan a la forma unida o formar estructuras cerradas que inhiban la asociación con ARN. Luego de la unión a ARN, los dominios pueden adoptar distintas conformaciones

[18]. En muchos casos, los dominios individuales dentro de la misma proteína poseen diferente especificidad de secuencia, lo que sugiere que podrían conducir a que una misma proteína se una a múltiples ARNs (unión de tipo *trans*), mientras que en otros casos múltiples dsRBDs interaccionarían con ARNs en forma no específica para incrementar la afinidad de unión (unión tipo *cis*).

Mientras que la arquitectura global de las proteínas DRBs está conservada, la longitud y estructura de los *linkers* que separan los dominios dsRBDs son variables. El ejemplo más extremo de variabilidad de estructura es DGCR8, donde los 43 residuos del *linker* adoptan estructuras helicoidales que se empaquetan junto con una hélice C-terminal adicional en el segundo dominio generando una estructura compacta [19]. En cuanto a la longitud de los *linkers*, la misma podrían ser un determinante crítico de la afinidad y el modo de unión. Los *linkers* cortos podrían favorecer la unión de tipo *cis*, mientras que los más largos podrían conducir a uniones de tipo *trans* [20]. También se ha postulado que las afinidades de unión de dos dsRBDs son aditivas en el caso de *linkers* extremadamente largos o multiplicativas en el caso de *linkers* cortos (o en *linkers* largos pero con dominios que interactúan) [18]. Las interacciones dinámicas entre las interfases proteína-ARN y proteína-proteína, permitidas por un *linker* flexible, contribuyen al mecanismo de formación del complejo proteína-ARN. En la Figura 3 se muestran dos mecanismos de reconocimiento de proteínas multidominio-ARN.

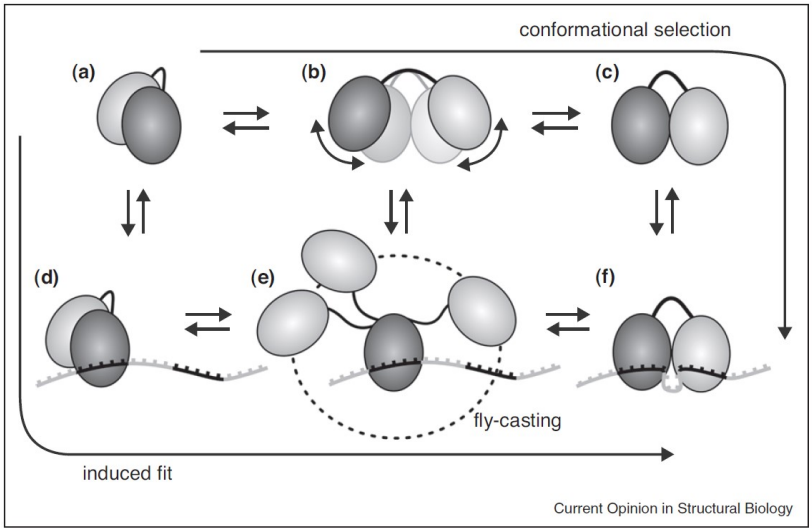


Figura 3: Mecanismos de reconocimiento de proteínas multidominio-ARN. El camino que se muestra por la parte superior corresponde a un proceso de selección conformacional. El camino que se muestra por la izquierda y abajo corresponde a un proceso de ajuste inducido. En a) se muestra una conformación cerrada estabilizada por interacciones débiles, mientras que en b) se muestra un ensamble de conformaciones posibles para la forma apo. La dinámica puede permitir la exploración de una pequeña proporción de conformaciones que se asemejan a la forma unida. En el mecanismo de selección conformacional esta población puede

unirse al ligando. En d) se muestra como la unión de uno de los dominios al ARN de una forma cerrada o autoinhibida puede promover un rearreglo de los dominios. En e) se esquematiza el proceso de "Fly-casting", que puede conducir a la búsqueda de motivos específicos. Figura extraída de Mackereth and Sattler, 2012[18].

Tanto TRBP como PACT tienen arquitecturas que consisten en tres dsRBDs separados por regiones *linkers* desestructuradas. En estas proteínas, el primer y segundo dominio unen ARN [21] mientras que el tercero media la dimerización e interacción con otras proteínas [22–25]. Se demostró para ciertas proteínas que los dsRBDs se deslizan a lo largo del ARNdh [26] y la dinámica de este movimiento deslizante para proteínas con dos dsRBDs se correlaciona con el largo de *linker* que conecta los dominios. TRBP, que posee un *linker* de 61 residuos entre sus dos dominios dsRBDs N-terminales, muestra un deslizamiento mayor que para PACT o Staufen, donde los dominios dsRBDs están separados por 25 residuos [27]. Un trabajo reciente de Sattler y colaboradores demostró la influencia de la longitud del *linker* en la afinidad, modo de unión y función en Loqs, el ortólogo de TRBP en *D. melanogaster* [28]. Los autores mostraron que el acortamiento del *linker* conduce a una menor afinidad y a un deslizamiento más pronunciado de los dsRBDs sobre el ARNdh. También demuestran que un *linker* corto dificulta a la proteína acomodar los dos dominios dsRBDs sobre la molécula rígida de ARNdh y sugieren que es necesaria una longitud mínima del *linker* para que haya una interacción funcional entre los dominios dsRBDs. Para el caso de RDE4, los dsRBDs tienen elementos estructurales adicionales dentro de su *linker* relativamente largo (63 residuos) pero no interaccionan entre sí [29]. También se ha demostrado que la región del *linker* es esencial para su interacción con DICER [30] y en el silenciamiento génico *in vivo* en *C. elegans* [31]. DRB4, la proteína auxiliar de DCL4, comparte la misma arquitectura que HYL1. Sin embargo, el *linker* entre los dsRBDs es mucho más corto, de solamente 9 residuos. Un trabajo reciente mostró que mientras los dos dominios de DRB4 son unidades estructurales independientes en solución, el *linker* corto restringe su orientación relativa, y esta orientación es importante para la unión a ARNdh [32]. Además, mostraron que ese *linker* corto dificulta el reconocimiento de ARNdh por parte del dominio que le sigue. Por todo esto, el *linker* parece jugar un papel importante en la selección de sustrato en DRB4.

Recientemente se ha resuelto la estructura del complejo Dicer-TRBP humano por microscopía crioelectrónica (cryo-EM) [25]. La estructura muestra la ubicación del tercer dominio de TRBP, y sugiere la posible ubicación de los dos dominios dsRBDs N-terminales. La distancia entre los dominios es grande, explicando la necesidad de un *linker* largo entre los dominios. Por lo tanto, es poco probable que los mecanismos de unión de HYL1 a DCL1 o DRB4 a DCL4 procedan de un modo similar, considerando la baja flexibilidad que le confieren sus *linker* más cortos.

1.1.5. HYL1

Existen cinco miembros de la familia de proteínas de unión a ARN doble hebra (DRB1-5) en *A. thaliana* y al menos ocho en arroz [33]. De los distintos miembros de la familia DRB, HYL1/DRB1 es el factor más importante de la biogénesis de miARNs [34]. Si bien DCL1 por sí misma es capaz de procesar los pri-miARNs para dar productos de ~ 21 nt, el corte no es específico. HYL1 también afecta al proceso de *splicing* de algunos pri-miARNs y a la selección de hebras guía de los dúplex de miARN/miARN* [35]. Las plantas con mutaciones en *HYL1* presentan un fenotipo pleiotrópico con disminución en el crecimiento radicular, hojas hiponásticas (curvatura hacia arriba), un tamaño de hojas reducidas, retraso en el crecimiento y floración, mayor cantidad de brotes laterales, menor fertilidad y alteración en la respuesta a hormonas (Figura 4) [36]. Las plantas *hyl1* presentan acumulación de precursores de miARN y niveles bajos de miARN, lo que indica que la ausencia de una proteína HYL1 funcional afecta el sistema a nivel del procesamiento. Sin embargo, la falta de HYL1 puede ser compensada por una forma más activa de DCL1 [37,38]. Diferentes mecanismos de regulación afectan la actividad, estabilidad y localización nuclear de HYL1. Por ejemplo, un mecanismo que conduce a una mejora de la actividad de HYL1 es la desfosforilación por parte de las proteínas CPL1 y CPL2 [39].

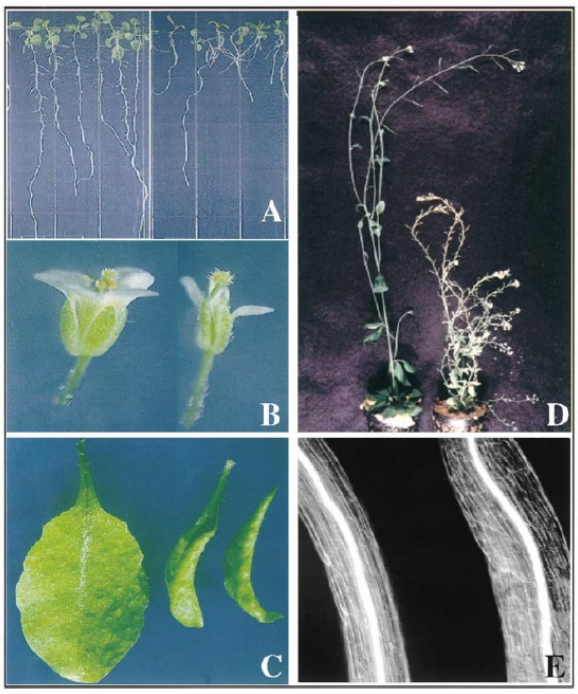


Figura 4: Fenotipos pleiotrópicos de la mutante hyl1 de Arabidopsis. Para cada foto, las plantas wt se muestran a la izquierda y las mutantes hyl1 a la derecha. Figura extraída de Lu y Federoff, 2000 [36]

La proteína HYL1 posee en el extremo N-terminal dos dominios dsRBDs independientes (residuos 1-170), unidos por un *linker* de 17 residuos que se considera flexible. Luego se encuentra una señal de localización nuclear (NLS) y una región desestructurada en el extremo C-terminal que contiene una secuencia de seis repeticiones de 28 aminoácidos (residuos 171-419) (Figura 5). La función reportada para el primer dominio dsRBD es la de unión al sustrato, que además es esencial para la localización de HYL1 en los cuerpos-D. Por su parte, al segundo dsRBD se le atribuyen funciones de reconocimiento proteico y de dimerización, interaccionando con DCL1 y regulando positivamente la eficiencia y precisión del procesamiento de los miARN. Su importancia fue demostrada tanto *in vitro* como *in vivo* [40,41]. Ambos dominios de HYL1 son suficientes para la función *in vivo* de la proteína [42].

La afinidad de los dominios por el ARN se determinó mediante la técnica de retardo en geles (EMSA, *Electrophoretic mobility shift assay*) con el precursor completo de miR172. Se pudo observar que el primer dominio dsRBD (D1 de acá en adelante) posee mayor afinidad por el ARNdh que el segundo dominio dsRBD (D2 de acá en adelante). El doble dominio mostró una afinidad similar a la de D1, indicando que el primer dominio es el principal responsable de la interacción con el sustrato pri-miARN. Los espectros de RMN (Resonancia Magnética Nuclear) de cada domino purificado independientemente, y de la proteína completa, muestran que estos dominios son estructuralmente independientes. Por titulación de cada dominio con cantidades crecientes de sustrato seguida por RMN fue posible mapear la superficie de interacción de la proteína con el ARNdh, mostrando que D1 interacciona *in vitro* con el ARN sustrato a través de las regiones canónicas descriptas para otras estructuras de complejos ARN:dsRBD. D2, en cambio, no mostró cambios significativos luego de la adición de ARN. Las resonancias de los núcleos de D2 en las regiones N-terminales de las hélices 1 y 2 fueron sustancialmente ensanchadas, sugiriendo una unión débil al ARN en un régimen de intercambio intermedio. Por otra parte, el bucle β1-β2, el cual es uno de los principales determinantes de la unión a ARN, no se vio afectado. Las perturbaciones inducidas por la unión de ARN en cada dominio dsRBD fueron las mismas que en la construcción con ambos dominios, indicando que los dos dominios son funcionalmente independientes [43].

Si bien ha sido posible describir a los dominios en forma aislada, se desconoce la dinámica entre ambos dominios cuando están libres o unidos al precursor. La flexibilidad entre los dominios, determinada por el *linker,* puede ser relevante para el funcionamiento de la proteína dentro del complejo.

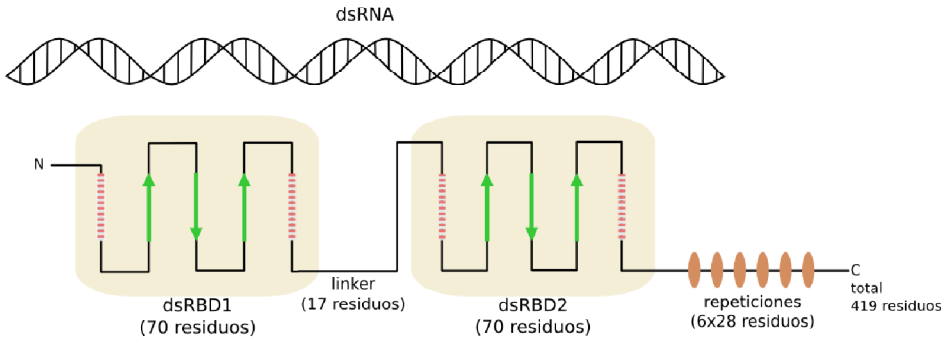


Figura 5: Esquema de dominios y regiones de HYL1. El gráfico esquematiza la unión de los dominios dsRBD al ARNdh a través de las tres regiones de interacción: N-terminal de la hélice 1, bucle beta 1 - beta 2 y N-terminal de la hélice 2.

1.2. Proteínas, estructura y movimientos

1.2.1. Dominios y proteínas multidominio

Los dominios proteicos son los módulos funcionales y estructurales básicos de las proteínas. La mayoría de los dominios son unidades que se pliegan en forma autónoma, y cada uno está frecuentemente asociado una función distinta. El gran número de combinaciones de dominios que se observa en proteomas sugiere que la mezcla al azar de los mismos es la mayor fuente de innovación evolutiva para nuevas funciones proteicas, junto con la duplicación de dominios y los eventos de recombinación [44]. Las proteínas multidominio comprenden un 80% de los proteomas eucariotas y dos tercios de las proteínas procariotas [45]. Estas poseen ventajas comparadas con las proteínas con un solo dominio, debido a que incrementan la concentración local efectiva de los sustratos (o productos) a lo largo de una ruta metabólica o de señalización [44].

1.2.2. *Linkers*

Los diferentes módulos en proteínas multidominio están conectados por cadenas de aminoácidos cortas o largas, que están usualmente caracterizadas por un cierto nivel de desorden. Nos referimos a ellas como *linkers*. Al igual que los dominios discretos y estructurados, los *linkers* que los unen también deben considerarse como unidades funcionales y no simplemente como conectores que mantienen a los dominios juntos [45]. Entre sus funciones podemos mencionar que (i) contribuyen a una modulación cooperativa de las interacciones interdominio y proteína-proteína, (ii) establecen comunicación distante entre diferentes módulos funcionales de la proteína, (iii) dirigen movimientos

correlacionados de los dominios actuando como elementos bisagra y (iv) actúan como espaciadores que mantienen a los dominios en distancias extremo-extremo. Los *linkers* contribuyen, al igual que la combinación al azar, la duplicación y la recombinación de dominios, a la variabilidad dentro del proteoma [44].

Análisis de secuencia indican que los *linkers* varían en longitud y que generalmente consisten en residuos flexibles [46]. La longitud del *linker* es importante ya que se ha visto que su modificación afecta, entre otras cosas, la estabilidad proteica, la velocidad de plegado y la interacción dominio-dominio. Trabajos previos demostraron también la relación entre la flexibilidad del *linker* y su función [46–48].

1.2.3. Movimientos conformacionales

Las proteínas son intrínsecamente dinámicas. Pueden experimentar cambios conformacionales de distinta magnitud, desde fluctuaciones en enlaces locales hasta transiciones de plegado/desplegado, en escalas de tiempo que van desde los femtosegundos hasta los días. Muchos de estos rearreglos están íntimamente relacionados con la función biológica de la proteína y con su capacidad de interaccionar con otras moléculas biológicas. Las dinámicas interdominio juegan un papel esencial en un gran número de procesos de reconocimiento molecular y de señalización. Por ello, es importante estudiar estos movimientos y su complejidad, experimentalmente, por técnicas computacionales, o por combinación de ambas [44].

1.3. Técnicas paramagnéticas

1.3.1. Uso de sondas paramagnéticas

Los movimientos que realizan las proteínas están íntimamente relacionados con su función biológica. Por ello, para el estudio de las mismas es importante el análisis tanto de su estructura como de las interacciones y su dinámica en condiciones nativas. Existen distintas técnicas para estudiar los estados conformacionales que exploran las macromoléculas.

Los dominios dsRBDs de HYL1 miden *ca.* 40 Å de largo. A su vez, su *linker* puede abarcar hasta 20Å en su forma totalmente extendida. Son pocos los métodos biofísicos que permiten realizar medidas estructurales con resolución y precisión en la escala de nanómetros, necesarias para el estudio de conformaciones de HYL1. Entre ellos, resulta útil el uso de especies paramagnéticas. Existen distintas técnicas que utilizan sondas paramagnéticas para obtener información estructural en escala de nanómetros, tanto de RMN, como de EPR (Resonancia Paramagnética Electrónica).

Una especie paramagnética es un átomo, molécula o ión que posee electrones desapareados. La introducción de un grupo paramagnético en una proteína puede efectuarse por sustitución del ión metálico en metaloproteínas (Figura 6A) o por incorporación de etiquetas paramagnéticas (Figura 6B y C). Existen dos tipos de etiquetas paramagnéticas: radicales nitróxido estables >N-O· y quelantes de metales (como EDTA, péptidos de unión a metales) que unen iones metálicos paramagnéticos con alta afinidad [49]. Hay distintas alternativas para unir covalentemente especies paramagnéticas a proteínas:

- por modificación covalente de cisteínas expuestas al solvente, las cuales pueden ser nativas o introducidas por mutagénesis sitio dirigida. Algunos ejemplos son los radicales nitróxido estables (>N-O·) y quelantes de metales paramagnéticos (derivados de EDTA). Entre las desventajas de esta metodología se encuentran los problemas derivados de reacciones de intercambio disulfuro. Además, la presencia de múltiples residuos de cisteína en la proteína dificulta la incorporación selectiva del grupo paramagnético, especialmente si esos residuos son funcionalmente importantes y no pueden mutarse.
- introducir genéticamente secuencias cortas de aminoácidos que unen iones de metales paramagnéticos, los cuales quedan encapsulados dentro de una esfera de ligando bien definida con alta afinidad por el metal. Algunos ejemplos son la etiqueta de histidinas (His-tag) y las etiquetas de unión a lantánidos (LBT). Estas etiquetas tienen como ventajas que pueden ser incorporadas por técnicas estándar de biología molecular a las proteínas de interés y que son autoensamblables.

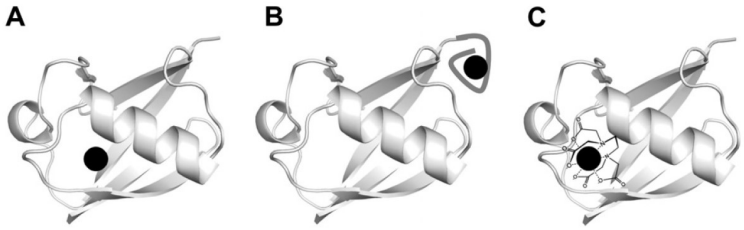


Figura 6: Métodos para introducir un centro paramagnético en una proteína. A) reemplazo de un ión metálico intrínseco en metaloproteínas. B) unión de un péptido que une metales. C) unión de una pequeña molécula a un residuo de cisteína que quela metales o que posee un radical. Figura extraída de Koehler y Meiler, 2011 [50].

Obtener una descripción de los estados conformacionales que explora una proteína es importante para entender su mecanismo de acción. En presencia de conformaciones heterogéneas, el resultado experimental refleja variabilidad estructural y provee de resultados promedio ya que la interconversión entre diferentes conformaciones suele ser rápida con respecto a los tiempos del experimento. El desafío está en recuperar, a partir de esos datos experimentales, información acerca del ensamble. Describir al ensamble a partir

de datos experimentales es, de hecho, un problema “mal-planteado”, ya que admite un número infinito de soluciones [51]. La naturaleza altamente dinámica de las proteínas multidominio necesita del desarrollo de enfoques analíticos para la interpretación de la información experimental disponible en términos de ensambles representativos. Aunque la descripción del espacio conformacional disponible representa, de por sí, una tarea demandante, un desafío igual de importante es la estimación de la población en los diferentes estados y sus tasas de interconversión. El análisis cuantitativo del espacio conformacional explorado por dos dominios es un problema desafiante [52].

1.3.2. Etiqueta de unión a lantánidos (LBT)

Un ejemplo de sonda paramagnética es la etiqueta de unión a lantánidos **Lanthanide Binding Tag (LBT)** [53–55]. Esta etiqueta consiste en una secuencia peptídica corta (de alrededor de 15 aminoácidos). Fue diseñada a partir de bucles naturales de unión a calcio, optimizados para quelar iones lantánidos de forma fuerte y selectiva. La misma puede ser incorporada por técnicas estándar de biología molecular a las proteínas de interés con un impacto mínimo en la estructura y función de las mismas. Puede ser incorporada tanto a los extremos de la proteína, como a cadenas laterales de cisteína o a regiones de bucles [56,57]. Es una etiqueta auto-ensamblable que utiliza únicamente componentes disponibles en forma natural en la célula, lo cual permite llevar a cabo estudios *in cell*. En la Figura 7 se muestra un ejemplo de secuencia LBT y su estructura resuelta por cristalografía de rayos X.

Tyr-Ile-Asp-Thr-Asn-Asn-Asp-Gly-Trp-Ile-Glu-Gly-Asp-Glu-Leu

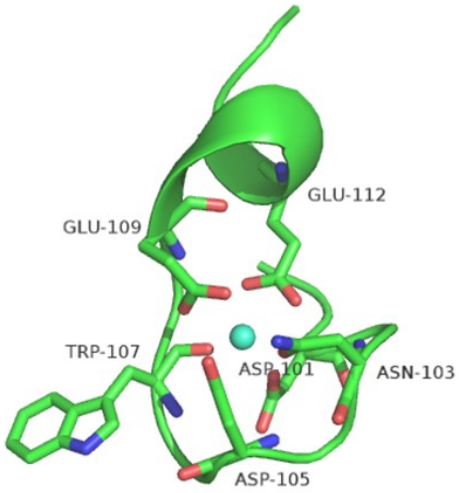


Figura 7: LBT. Arriba: ejemplo de secuencia LBT, abajo estructura de un LBT en donde se indican los residuos que interaccionan con el metal.

La incorporación de iones lantánidos paramagnéticos a proteínas es una herramienta valiosa en el estudio estructural debido a su baja reactividad química, su naturaleza bi-ortogonal y sus fuertes propiedades magnéticas [57]. Además de ser paramagnéticos, algunos metales lantánidos presentan otras características que resultan útiles. En particular, iones como Tb(III) y Eu(III) son luminiscentes luego de la sensibilización por fluoróforos orgánicos [56,58]. La transferencia de energía que ocurre entre la cadena lateral indol del triptófano excitado de la etiqueta y el Tb(III) unido a ella es posible solamente si ambos están suficientemente cerca (Figura 8). El proceso de emisión a partir del lantánido excitado no resulta de una transición singlete-singlete (Figura 9), por lo que no es considerado fluorescencia y se denomina luminiscencia. Experimentalmente consiste en excitar la muestra a la longitud de onda de excitación del triptófano (280 nm) y registrar la intensidad de emisión de la muestra a la longitud de onda de emisión del Tb(III) (544 nm) [59].

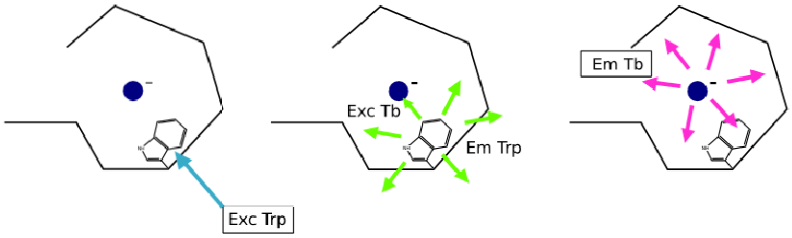


Figura 8: Transferencia de energía Trp-Tb(III)

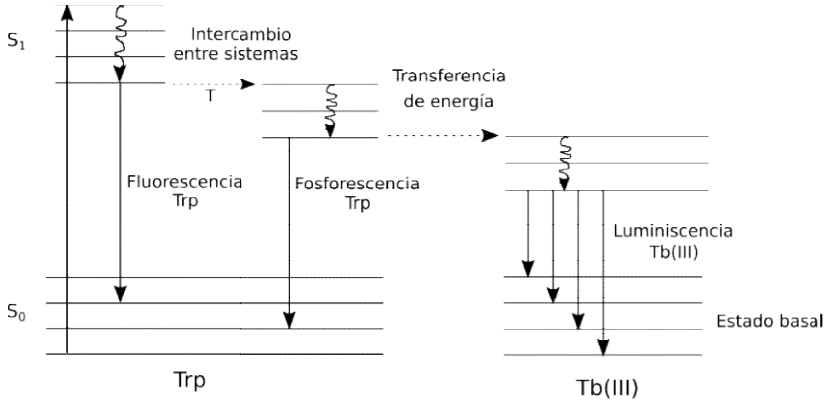


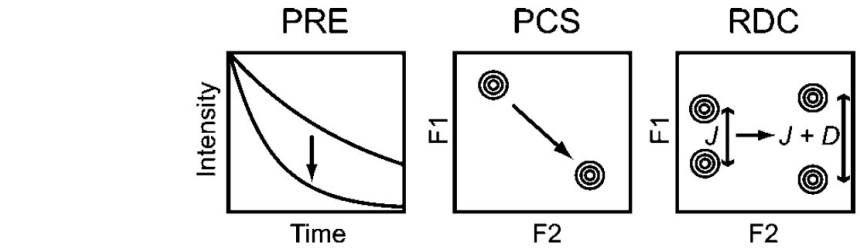
Figura 9: Transiciones energéticas entre Tyr y Tb(III). Figura adaptada de Di Gennaro, 2013 [60].

1.3.3. Medición de distancias por RMN

En RMN, los centros paramagnéticos interaccionan con los espines nucleares de la proteína. Esto resulta en efectos dependientes de la distancia y/o orientación, que pueden

ser utilizados como restricciones estructurales. Los tres fenómenos inducidos por el campo magnético local más utilizados son (Figura 10) [50]:

- Desplazamientos de pseudocontacto (Pseudocontact shifts o PCS): perturbación de las frecuencias de resonancia de los núcleos con dependencia de distancia de r^{-3} y de orientación.
- Acoplamientos dipolares residuales (Residual Dipolar Couplings o RDC): producto del alineamiento parcial de moléculas paramagnéticas. La fuerza del acoplamiento está relacionada con la distancia entre los dos núcleos y su promedio de orientaciones con respecto al campo magnético estático, lo que permite medir los ángulos entre los distintos vectores internucleares en la molécula.
- Incremento paramagnético de la relajación (**Paramagnetic Relaxation Enhancement o PRE**): aumento en la velocidad de relajación de núcleos cercanos al ión paramagnético con una dependencia de distancia de r^{-6}. Surge de interacciones dipolares magnéticas entre el núcleo y los electrones desapareados del centro paramagnético [49]. En el espectro de RMN se observa cómo dichas señales se ensanchan, disminuyendo su intensidad hasta incluso desaparecer. El efecto puede llegar a distancias de hasta 35 Å, dependiendo del radical libre que se utilice [61]. Todas las especies paramagnéticas exhiben este efecto



Unpaired electrons:			
Isotropic	Yes	No	No
Anisotropic	Yes	Yes	Yes

Figura 10: Efectos paramagnéticos observables por RMN. Figura extraída de G.Marius Clore, 2009 [49].

La presencia y magnitud de cada uno de los efectos paramagnéticos dependen del ión metálico con el que se trabaje, de su tensor de susceptibilidad magnética (χ), y de la velocidad de relajación del electrón. La susceptibilidad magnética es una propiedad inherente de las sustancias que indica qué tanto la sustancia se magnetiza en un campo magnético, o cuánto interacciona con el campo magnético. La anisotropía de susceptibilidad magnética surge cuando la magnetización depende de la orientación. Dado que el tensor-χ

electrónico y el tensor-g electrónico están muy relacionados, en general si el tensor-g electrónico es anisotrópico (depende de la orientación), el tensor-χ electrónico también es anisotrópico. El efecto PRE se puede detectar en cualquier sistema paramagnético, mientras que PCS y RDC solo son observados en sistemas con factores-χ electrónicos anisotrópicos. Algunos ejemplos se muestran en la Tabla 1.

Lantánido	PRE	PCS	RDC
Tb(III) y Dy(III)	Si	Si	Si
Gd(III)	Si	No	No
Lu(III)	No	No	No

Tabla 1: Efectos paramagnéticos de algunos iones lantánidos.

Todos los lantánidos son paramagnéticos excepto el primer y el último miembro de la serie, La(III) y Lu(III), los cuales son metales diamagnéticos. Por otra parte, Gd(III) posee el espín y el tiempo de correlación de espín electrónico más grandes, lo cual resulta en un ensanchamiento de las señales importante. Este efecto puede ser medido hasta 20 Å de distancia. Su anisotropía de susceptibilidad magnética es despreciable por lo que no conduce a un alineamiento de la proteína y por lo tanto no produce efectos PCS y RDC apreciables.

1.3.3.1. PRE

El efecto PRE se puede utilizar para extraer restricciones de distancia a partir de cocientes de intensidades de las señales. PRE define esferas de distancia alrededor del centro paramagnético. El radio de esas esferas depende de varios parámetros, como el número de electrones desapareados, el tiempo de correlación del espín electrónico (τ_e), el tiempo de correlación rotacional y la fuerza del campo magnético. PRE está determinado por el tamaño del tensor de susceptibilidad magnética y es menos pronunciado para ^{15}N y ^{13}C (en comparación con ^{1}H) por sus menores relaciones giromagnéticas. El PRE se puede estimar a partir del cociente de las intensidades o los anchos de las señales del espectro paramagnético vs el diamagnético [50]. Las ecuaciones que definen al efecto PRE se describen a continuación [51]:

$$R_{2M}^{\mathrm{PRE}} = \frac{k'_{\mathrm{Solomon}} + k'_{\mathrm{Curie}}}{r^6}$$

$$k'_{\text{Solomon}} = \left(\frac{\mu_0}{4\pi}\right)^2 \frac{\gamma_I^2 g_e^2 \mu_B^2 S(S+1)}{15}\left[4\tau_c + \frac{\tau_c}{1+(\omega_I-\omega_S)^2\tau_c^2} + \frac{3\tau_c}{1+\omega_I^2\tau_c^2} + \frac{6\tau_c}{1+(\omega_I+\omega_S)^2\tau_c^2} + \frac{6\tau_c}{1+\omega_S^2\tau_c^2}\right]$$

$$k'_{\text{Curie}} = \frac{1}{5}\left(\frac{\mu_0}{4\pi}\right)^2 \frac{\omega_I^2 g_e^4 \mu_B^4 S^2(S+1)^2}{(3k_BT)^2}\left[4\tau_{\text{Curie}} + \frac{3\tau_{\text{Curie}}}{1+\omega_I^2\tau_{\text{Curie}}^2}\right]$$

donde $\omega_I = \gamma_I B_0$ es la frecuencia nuclear de Larmor, ω_S es la frecuencia electrónica de Larmor, g_e es el factor de Lande del electrón g, μ_B es el magnetón de Bohr, μ_0 es la permeabilidad magnética en el vacío, S es el número cuántico de espín electrónico , y los tiempos de correlación relacionados con los mecanismos responsables de la relajación son:

$$\tau_c^{-1} = \tau_e^{-1} + \tau_r^{-1} + \tau_M^{-1} \quad \text{y} \quad \tau_{\text{Curie}}^{-1} = \tau_r^{-1} + \tau_M^{-1}$$

donde τ_e^{-1}es el tiempo de relajación del electrón, τ_r^{-1}es el tiempo de correlación rotacional y τ_M^{-1} es el tiempo de intercambio.

1.3.3.2. $^1H^{15}N$-HSQC

Los efectos paramagnéticos sobre los espines nucleares pueden ser estudiados en distintos tipos de experimentos. Entre ellos, el más simple y popular es el uso de experimentos $^1H^{15}N$-HSQC (Heteronuclear Single Quantum Coherence). El experimento $^1H^{15}N$-HSQC es la base de gran parte de los estudios de RMN multidimensional en proteínas. En este experimento se correlaciona el desplazamiento químico de un núcleo de ^1H con el de un núcleo de ^{15}N al que se encuentra directamente unido. Así, en el espectro $^1H^{15}N$-HSQC de una proteína se espera observar un número total de señales igual al número de residuos de la proteína, sin contar los residuos de prolina, que no poseen ^1H amídico y, por ende, no generan señal en este espectro, y el residuo amino terminal, que intercambia los ^1H muy rápido con el solvente. A las señales correspondientes a los grupos amida del esqueleto de la proteína se suman las señales correspondientes a los grupos amida de las cadenas laterales de glutamina y asparagina y las señales de los pares NH de los anillos indólicos de los residuos de triptófano. La posición de cada señal en el espectro depende fuertemente del microentorno químico del par $^1H^{15}N$ amida correspondiente. Esta dependencia convierte al espectro $^1H^{15}N$-HSQC en la huella dactilar de una proteína bajo las condiciones de estudio. Por este motivo, el $^1H^{15}N$-HSQC es el experimento ideal, en la gran mayoría de los casos, para estudiar la interacción de una proteína con ligandos, así como también el efecto del pH, la temperatura o la fuerza iónica, entre muchas otras aplicaciones.

1.3.3.3. Asignación de señales

Teniendo en cuenta la información que aporta el espectro $^{1}H^{15}N$-HSQC, el potencial de dicho experimento para los estudios biofísicos de proteínas depende de que se disponga de la asignación de sus señales. Es decir, para poder interpretar los cambios observados en los espectros adquiridos en estados diferentes de la proteína, es necesario determinar a qué residuo corresponde cada señal. Para llevar a cabo la asignación de las señales de un espectro $^{1}H^{15}N$-HSQC, la estrategia más utilizada consiste en la adquisición de experimentos de triple resonancia. En estos experimentos, cada par ^{1}H-^{15}N es correlacionado con el valor de desplazamiento químico de distintos núcleos de ^{13}C del mismo residuo y/o de su residuo anterior en la secuencia proteica, según el caso. El éxito de esta estrategia se basa en que los desplazamientos químicos de ^{13}C de los carbonos C_α y C_β de un dado aminoácido son dependientes tanto de la identidad química del mismo como de su entorno, posibilitando la identificación del tipo de residuo.

1.3.3.4. T2

El retorno al equilibrio en resonancia magnética se llama relajación. La velocidad a la cual la magnetización en el plano xy vuelve a cero se llama R2 y se caracteriza por una constante de tiempo llamada T2. La medida de T2 se lleva a cabo mediante una secuencia simple de pulsos. Primero se rota la magnetización al plano xy con un pulso de 90° en x, y luego de distintos tiempos (T'') se miden los espectros correspondientes, cuya intensidad decae según la constante de tiempo T2. Para suprimir otros factores que puedan afectar la relajación, como por ejemplo heterogeneidad en el campo magnético, se insertan pulsos de 180° durante el tiempo T''. Las intensidades resultantes se grafican vs T'', y la curva obtenida se ajusta a la ecuación [62]:

$$M_{XY}(T'') = M_{Z0}\, e^{-T''/T2}$$

1.3.4 Medición de distancias por EPR

Otra técnica paramagnética que permite el estudio estructural de proteínas es **PELDOR (Pulsed Electron DOuble Resonance)** (también conocido como DEER por Double Electron-Electron Resonance). PELDOR es una técnica de EPR (Resonancia Paramagnética Electrónica) que permite determinar con precisión y en la escala de nanómetros (de 1.5 a 8 nm) la distancia entre dos dipolos paramagnéticos. No requiere cristalización y no está limitada al tamaño de la proteína. Esta herramienta ha sido aplicada en los últimos años sobre diferentes sistemas para el estudio de la estructura de proteínas y

ácidos nucléicos, y en la determinación de interacciones entre macromoléculas [63,64]. Una gran ventaja de la misma es que permite trabajar en una gran variedad de condiciones, incluso dentro de las células. La condición fundamental para aplicar la técnica es la presencia de dos o más sondas paramagnéticas dentro del sistema en estudio.

Como se muestra en la Figura 11, el experimento se compone de 4 pulsos básicos. Tres de ellos se realizan a la frecuencia de microonda de observación ($\nu_{observe}$) creando un eco de espín (negro). Otro de los pulsos se realiza a una segunda frecuencia de microondas (ν_{pump}) que se coloca entre el segundo y el tercer pulso a un tiempo variable *t*. Conforme se va barriendo este tiempo *t* entre los pulsos de observación, la intensidad del eco se va modulando (rojo), debido a las interacciones dipolares entre los espines que resuenan a las distintas frecuencias. La frecuencia de modulación de ese eco está relacionada con la distancia entre los espines. Una vez obtenidos las curvas se procede a eliminar la línea de base y se procesa para convertir la frecuencia de la modulación en una distribución de distancia.

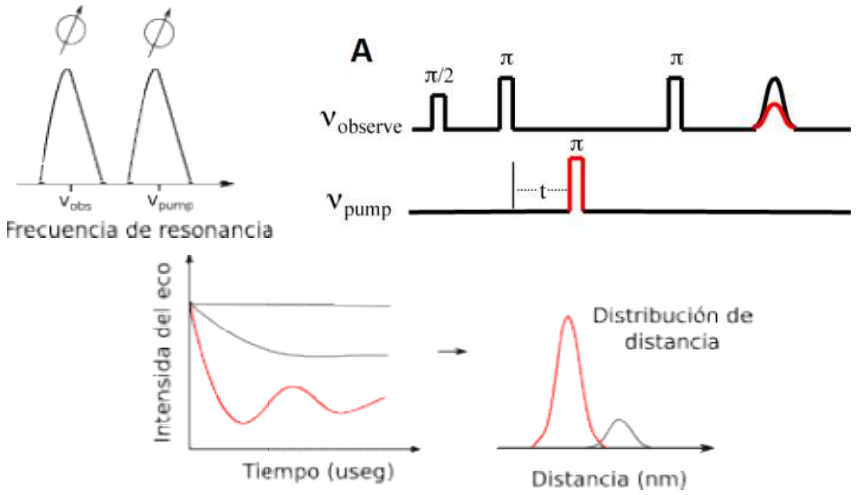


Figura 11: Experimento PELDOR. En la parte superior izquierda se esquematizan dos frecuencias distintas para los pulsos. En la parte superior derecha se indica la secuencia de pulsos. En la parte inferior se muestra un ejemplo de distintas curvas con sus distribuciones de distancia correspondientes.

1.3.4.1. Estudios *in cell*

Existen pocas técnicas capaces de estudiar la estructura de proteínas y sus interacciones en el interior celular. Este estudio es importante porque el ambiente celular es complejo y puede diferir significativamente con respecto a las condiciones *in vitro*. Una de las técnicas que permite estudiar estructuras dentro de las células es PELDOR [65–70]. Un ejemplo del uso de esta técnica es el estudio conformacional de α-sinucleína introducida

dentro de células de mamífero por electroporación [70]. Las sondas de espín basadas en radicales nitróxido tienen una estabilidad limitada en el ambiente citosólico reductor [71]. Por ello, para el estudio *in cell* las sondas de espín basadas en metales son más apropiadas. Una forma de incluir el metal ha sido incorporarlo encapsulado en una esfera de ligando bien definida que posee una alta afinidad por el metal [68,69,72–76]. Cabe resaltar que el uso de etiquetas autoensamblables para estudios por PELDOR *in cell* no había sido desarrollado al comienzo de este trabajo de Tesis.

2. OBJETIVOS

2.1. Objetivo general

Entender la importancia del *linker* entre dominios dsRBDs de la proteína de procesamiento de miARN HYL1 de *Arabidopsis thaliana*. Analizar su conservación en tamaño y secuencia por métodos bioinformáticos. Estudiar su participación en la modulación de la estructura y dinámica de los dominios a través de la combinación de restricciones dispersas y modelado de cuerpos rígidos.

HYL1 reconoce precursores de miARN y sus dominios tienen capacidad de unirse a su sustrato. Sin embargo, se desconoce la forma de unión y la ubicación relativa de los dominios sobre el ARNdh. La diferencia de afinidades entre los dominios puede dar lugar a distintas configuraciones. Por otra parte, se ha reportado que HYL1 dimeriza a través de su dominio C-terminal. Los dominios dsRBD de HYL1 están unidos por un *linker* de 17 residuos que les confieren libertad conformacional limitada. La libertad conformacional se va a manifestar en la forma libre y puede influir tanto en la unión de HYL1 a pre-miARNs como en la dimerización. Por las dimensiones de los dominios y del *linker* las técnicas más adecuadas para su estudio, como se explica en la introducción, son PRE y PELDOR. En la Figura 12 se ejemplifican distintas situaciones posibles y observables con las técnicas mencionadas. Desde la figura 12A hasta la 12E se esquematizan situaciones observables por PRE, mientras que de la 12F hasta la 12J las correspondientes a PELDOR. Las figuras 12A y 12F representan dominios que se mueven libremente, mientras que las figuras 12B y 12G corresponderían a dimerización. Por otro lado, en 12C y 12H uno de los dominios se une a ARN mientras que el otro permanecería libre. En el resto de las figuras ambos dominios se unen a ARN y su movimiento es limitado.

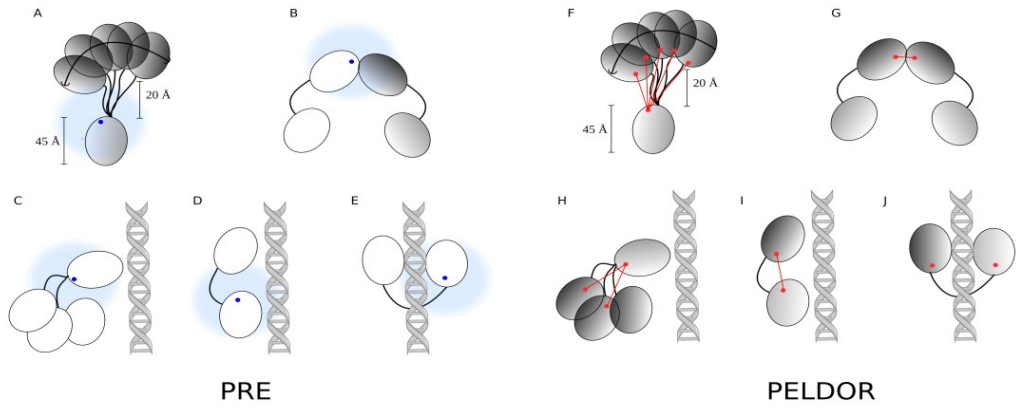


Figura 12: Situaciones posibles y observables con los experimentos planteados

2.2. Objetivos específicos

Para alcanzar el objetivo general se plantean los siguientes objetivos específicos:

- Análisis bioinformático de *linkers* que conectan dominios dsRBDs.
- Generación de construcciones que contienen etiquetas paramagnéticas que puedan ser utilizadas en los experimentos posteriores.
- Ejecución de los experimentos que dan restricciones de distancia
- Construcción *in silico* de conformaciones posibles y selección de una población de estructuras que explique los datos experimentales.

3. MATERIALES Y MÉTODOS

3.1. Cepas, genes sintéticos, vectores plasmídicos y medios de cultivo

3.1.1. Cepas bacterianas

- *Escherichia coli* DH5α [77]: Esta cepa se usó para guardar los plásmidos obtenidos en un cepario. Su genotipo es: *fhuA2 Δ (argF-lacZ) U169 phoA glnV44 Φ80 Δ(lacZ)M15 gyrA96 recA1 relA1 endA1 thi-1 hsdR17*.
- *Escherichia coli* BL21 (DE3): genotipo F^- ompT gal dcm lon $hsdS_B$ (r_B^- m_B^-) λ(DE3 [lacI lacUV5-T7 gene 1 ind1 sam7 nin5]). El profago ʎ (DE3) incluye el gen que codifica para la ARN polimerasa del fago T7 bajo control del promotor lacUV5, y el gen lacI, que codifica para el represor de la transcripción LacI. Esta cepa se utilizó para la expresión de todas las proteínas recombinantes.
- *Escherichia coli* BL21-Codon Plus(DE3)-RIL: cepa derivada de la anterior, que se utiliza para la expresión de proteasa TEV. Contiene un plásmido con resistencia a cloranfenicol que posee copias extra para genes de tARNs que suelen ser limitantes para la traducción de proteínas heterólogas en *E. coli*.

3.1.2. Genes sintéticos

Los genes sintéticos fueron adquiridos a través de la empresa GeneScript (http://www.genscript.com/). La empresa realizó la síntesis de los fragmentos de ADN correspondiente, incluyendo sitios de restricción solicitados en cada caso y su clonado en el vector pUC-57. Las preparaciones plasmídicas fueron entregadas en forma liofilizada. Previo a la transformación de bacterias competentes, las muestras fueron resuspendidas en 100 µl de agua miliQ estéril. Las secuencias solicitadas con sus sitios de restricción se muestran en el anexo (sección 7.1).

3.1.3. Vectores plasmídicos

- Plásmido pET-TEV [78]: Se utilizó como vector de expresión de proteínas recombinantes. Es un plásmido derivado del pET28-a, al cual se le reemplazó el sitio de corte de trombina por el sitio de corte de la proteasa viral TEV. Presenta un sitio de iniciación de la transcripción para la T7 ARN Polimerasa, resistencia a kanamicina como marcador de selección y una secuencia codificante para una etiqueta de histidina.
- Plásmido pUC-57: Vector en el cual fueron suministrados los genes sintéticos. Contiene un gen de resistencia al antibiótico ampicilina como marcador de selección.

3.1.4. Medios de cultivo

Para las purificaciones de proteínas se utilizaron dos medios de cultivo diferentes en función del tipo de experimento a realizar:

- Clonados, expresión de proteína sin marca isotópica y pre-cultivos de expresión: Medio de cultivo rico Luria-Bertani (en adelante, LB): peptona de caseína 1% p/v, extracto de levadura 0,5% p/v, NaCl 0,5% p/v. En su versión sólida, este medio incluye agar-agar de 1,5% p/v.
- Expresión de proteína marcada: Medio mínimo M9 (en adelante, M9): KH_2PO_4 25 mM, Na_2HPO_4 50 mM, NaCl 10 mM, $MgCl_2$ 1 mM, $CaCl_2$ 0,2 mM, Glucosa 0,4%, Solución de sales 1 ml/l (HCl concentrado 51,3 ml/l, $MgCl_2$ 10,75 g/l, $CaCO_3$ 2 g/l, $FeSO_4.7H_2O$ 4,5 g/l, $MnSO_4.4H_2O$ 1,12 g/l, $CoSO_4.7H_2O$ 280 mg/l, $H_3BO_3.H_2O$ 60 mg/l, $ZnSO_4.7H_2O$ 1,44 g/l) y 1 g/l NH_4Cl. Para la obtención de muestras con marcación isotópica se utilizó 1 g/l de $^{15}NH_4Cl$ o 1 g/l de $^{15}NH_4Cl$ y 2 g/l de [U- ^{13}C] glucosa (Cambridge Isotope Laboratories)

Para la preparación de ambos medios de cultivo el agua utilizada fue calidad miliQ.

3.2. Transformación bacteriana

3.2.1. Producción de células competentes

Para la generación de bacterias competentes de las cepas de *E. coli*, las células fueron crecidas en 500 ml de LB hasta alcanzar una DO_{600} de ~0,5. Luego, las células fueron enfriadas en hielo y cosechadas por centrifugación a 2600 g durante 10 min a 4 ^oC. El sedimento celular fue resuspendido en 100 ml de una solución fría de $MgSO_4$ 100 mM estéril y se incubó por 30 min en hielo. Posteriormente, las células fueron recuperadas por centrifugación a 2600 g, a 4^oC durante 10 min y el sedimento obtenido se resuspendió en 10 ml de una solución fría y estéril de $CaCl_2$ 100 mM, glicerol 15%. Por último, se fraccionó el volumen total en muestras de 300 µl que fueron congeladas y conservadas a -80°C hasta el momento de la transformación.

3.2.2. Transformación de células competentes

Se tomaron alícuotas de 100 µl de bacterias competentes, y se les adicionó 1 µl de plásmido ó 5 µl de mezcla de ligación según corresponda, en el caso de los controles negativos se agregaron 1 ó 5 µl de agua como reemplazo del plásmido. Esta mezcla se incubó en hielo durante media hora, luego 90 s a 42 ^oC y finalmente 2 min en hielo. A continuación, se adicionaron 900 µl de medio LB y las células se incubaron durante una hora a 37^oC. Transcurrida la incubación, el cultivo celular se centrifugó 2 min a 2600 g a temperatura ambiente, el sedimento de células se resuspendió en 100 µl de medio LB y

finalmente se sembró en placas de medio LB-agar preparadas con el antibiótico correspondiente para su selección (100 µg/ml de ampicilina para pUC-57 y 50 µg/ml de kanamicina para pET-TEV, 35 µg/ml de cloranfenicol para *E. coli* BL21-Codon Plus (DE3)-RIL). Las placas fueron incubadas toda la noche a 37°C.

3.2.3. Chequeo de transformantes

Para verificar la presencia del vector con el inserto en las colonias seleccionadas al azar se siguieron distintos protocolos:

- PCR de colonias: este procedimiento fue utilizado para hacer chequeos rápidos de un gran número de colonias. Requiere contar con oligonucleótidos que amplifiquen en forma diferencial según se encuentre o no el inserto en el vector (puede ser presencia/ausencia de banda, o diferencias de tamaño). Cada colonia a evaluar fue primero replicada e incubada en una placa fresca con el antibiótico correspondiente para su preservación. Una porción de cada colonia fue resuspendida en la mezcla de PCR, y se llevó a cabo la reacción con la enzima GO Taq polimerasa (Promega) siguiendo el protocolo sugerido por el fabricante. Las muestras amplificadas fueron resueltas por electroforesis de agarosa para el chequeo de los fragmentos. Los tamaños de los fragmentos de ADN se estimaron utilizando como marcador de ADN el marcador de peso molecular 100 bp Ladder (Promega).
- Análisis de restricción: este protocolo fue realizado sobre los plásmidos purificados de colonias transformantes. Requiere que los vectores a evaluar cuenten con sitios de restricción cuya digestión libere fragmentos cualitativamente distintos según se encuentre o no el inserto. Los cortes fueron realizados utilizando las enzimas correspondientes de acuerdo al protocolo descripto por el proveedor de cada una de ellas. Los fragmentos digeridos fueron analizaron mediante electroforesis en geles de agarosa. Los tamaños de los fragmentos de ADN se estimaron utilizando como marcador ADN el marcador de peso molecular 100 bp Ladder (Promega).
- Secuenciación de ADN: todas las construcciones de este trabajo fueron chequeadas por secuenciación. La secuenciación de los insertos se realizó en el servicio de secuenciación de la Universidad de Maine (USA). Una vez obtenida la secuencia, se verificó la identidad de la misma utilizando el programa de acceso libre *Ape* (A Plasmid Editor) (http://biologylabs.utah.edu/jorgensen/wayned/ape/).

3.3. Subclonado de fragmentos de ADN

3.3.1. Minipreparaciones de ADN plasmídico

Los plásmidos fueron purificados utilizando "kit" de purificación de ADN Wizard® SV Gel (Promega), siguiendo las indicaciones del fabricante.

3.3.2. Estimación de la concentración de ADN

La concentración y calidad de las muestras de ácidos nucleicos fueron estimadas a partir de medidas de absorbancia a 230 nm, 260 nm y 280 nm en equipos Nanovue. La concentración se estimó asumiendo que 1 unidad de absorbancia a 260 nm de ADN corresponde a una concentración de 40 µg/ml. La calidad se evaluó por los valores de los cocientes Abs_{260}/Abs_{230} para impurezas generales y Abs_{260}/Abs_{280} para proteínas.

3.3.3. Amplificación de fragmentos de ADN mediante reacción en cadena de la polimerasa (PCR)

Los oligonucleótidos sintéticos fueron adquiridos de la empresa GenBiotech y resuspendidos en agua calidad miliQ para su utilización.

Las reacciones de amplificación se llevaron a cabo con soluciones y enzimas de la marca Promega, tanto con la enzima Pfu ADN polimerasa o Pfx ADN polimerasa según su disponibilidad en el laboratorio. Las mezclas de reacción se prepararon siguiendo los protocolos recomendados por sus fabricantes.

3.3.4. Digestión de ADN con endonucleasas de restricción

Las digestiones de ADN con endonucleasas de restricción se realizaron utilizando enzimas y tampones comerciales. Las condiciones de reacción fueron las especificadas por el fabricante.

3.3.5. Electroforesis de ADN en geles de agarosa

La separación de moléculas de ADN por electroforesis en geles de agarosa se realizó mediante el sistema tipo submarino en cubas de acrílico. Se utilizaron concentraciones de agarosa de 1 a 2% (p/v) según el tamaño de las moléculas de ADN a resolver. Los geles se prepararon con solución amortiguadora TAE 1X (Tris-HCl 40 mM, Ácido acético glacial 20 mM y EDTA 1 mM) y se agregó el colorante de ADN SYBR® Safe (Invitrogen) o Bromuro de Etidio (Promega). Las muestras de ADN, previo a la siembra, se mezclaron con 1/6 volúmenes de solución de siembra (glicerol 30%, Xilene cyanol 0,25% y azul de bromo fenol 0,25%). Las corridas electroforéticas se llevaron a cabo en solución amortiguadora TAE 1X, empleando voltaje constante. Se utilizaron marcadores de tamaño molecular comerciales adecuados a la longitud de los fragmentos a visualizar.

3.3.6. Purificación de fragmentos de ADN a partir de geles de agarosa

Luego de la separación electroforética, el fragmento de ADN deseado se identificó mediante la visualización del gel con un transiluminador UV y se escindió la banda

correspondiente con bisturí. La purificación de los fragmentos de ADN del gel se llevó a cabo utilizando el¨kit¨de purificación AxyPrep DNA Gel Extraction Kit (Axygen).

3.3.7. Defosforilación de vectores

En los casos en que se había digerido el plásmido con una sola enzima de restricción, se realizó un paso de defosforilación del plásmido digerido previo a la ligación. Se agregó a cada digestión 1X de tampón FastAP y 1U de enzima FastAP thermosensitive. La reacción se llevó a cabo durante 1h a 37 °C.

3.3.8. Ligación de fragmentos de ADN

La ligación de fragmentos de ADN generados por corte con enzimas de restricción en los vectores plasmídicos digeridos correspondientes se llevó a cabo con la enzima T4 ADN ligasa (Promega). Las condiciones de la reacción fueron las especificadas por el fabricante. En todos los casos el volumen de reacción de ligación fue de 10 µl y se utilizaron 5 µl de la misma para la transformación.

3.4. Diseño de construcciones

3.4.1. Secuencias base para el diseño de construcciones

- 3Hx: secuencia *3H5L_2_mini* reportada por Huang y colaboradores en el año 2014 [79]. Esta secuencia de 80 residuos se pliega como un monómero que forma tres hélices alfa extremadamente estables y sin torsión que se extienden en torno a un eje central. Secuencia: TEEEIKKLEEEAKKLLEKLKKNGVTTTIIEEVKKKMEELLKKLKNS TKTKEAAEKMLKKMKELFKKAKLE.
- H6: motivo de seis residuos de Histidina con capacidad de quelar metales. Utilizado comúnmente para la purificación de proteínas con columnas de Níquel. Tiene capacidad de unir Mg(II). Secuencia: HHHHHH.
- LBT: secuencia *dSE3* optimizada para la unión a Gd(III) [55]. Secuencia: YIDTNNDGWIEGDEL.
- HYL1: Secuencia N-terminal de los dominios de HYL1 de *Arabidopsis thaliana*. Los dominios fueron definidos a partir de la herramienta PROSITE (http://prosite.expasy.org/). Secuencia:MTSTDVSSGVSN**C**YVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKSLFQSTVILD GVRYNSLPGFFNRKAAEQSAAEVALRELAKSSELSQ**C**VSQPVHETGL**C**KNLLQEYAQKMN YAIPLYQ**C**QKVETLGRVTQFT**C**TVEIGGIKYTGAATRTKKDAEISAGRTALLAIQS.

3.4.2. Diseño de construcciones con 3Hx

Las construcciones con 3Hx fueron utilizadas para la puesta a punto del sistema PELDOR. Las construcciones 3Hx, H63HxH6, L-3Hx-L y H6-L-3Hx-L fueron producidas mediante una colaboración con E. Bruch quien las generó por digestión y ligación a partir de genes sintéticos sobre el vector pET-TEV. Las construcciones L-3Hx, H6-L-3Hx, 3Hx-L y H6-3Hx-L fueron generadas a partir de las construcciones anteriores por digestión y ligación utilizando el sitio SacI presente dentro de la secuencia 3Hx.

3.4.3. Diseño de construcciones con HYL1

El gen sintético con la secuencia L-D1-L-D2-L (ver secuencia en anexo, sección 7.1) fue diseñado para la construcción de plásmidos que expresan el extremo N-terminal de HYL1 con etiquetas LBT en diferentes posiciones. Todas las construcciones diseñadas así como el procedimiento para generarlas se esquematizan en la Figura 13.

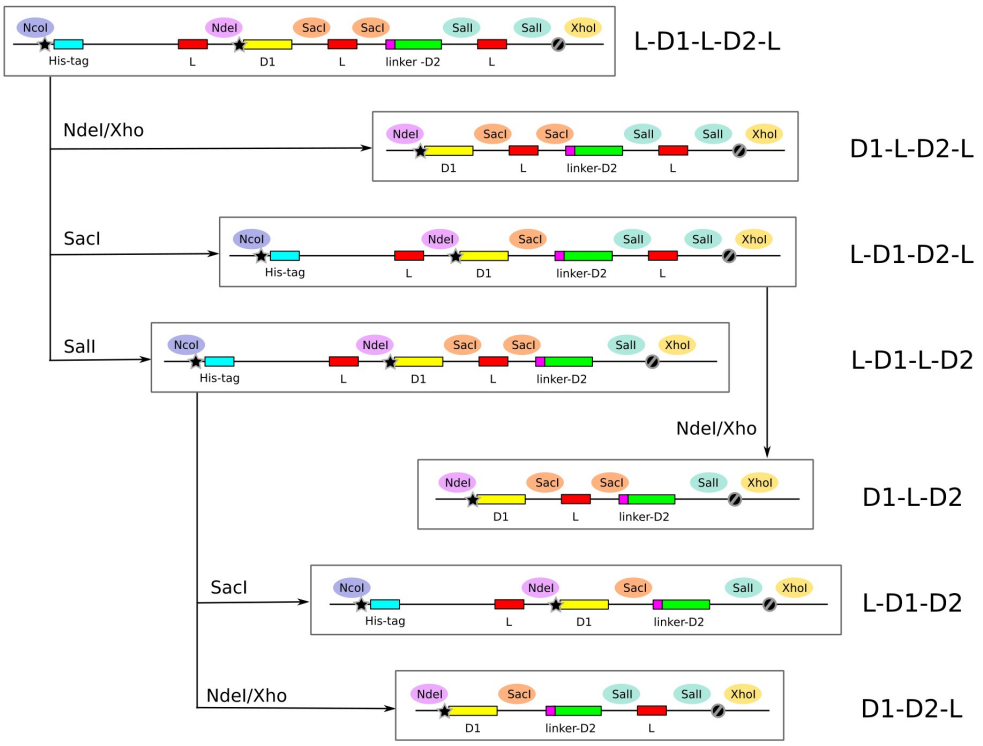


Figura 13: Generación de construcciones a partir del gen sintético

Los primeros 14 residuos de D1 no fueron tenidos en cuenta para reducir la flexibilidad en el extremo N-terminal. El primer paso fue clonar el gen sintético en el vector pET-TEV. Las construcciones con dos LBTs L-D1-L-D2 y L-D1-D2-L fueron preparadas por digestión y ligación sobre el mismo plásmido usando los sitios SalI y SacI respectivamente.

Por otro lado, el doble LBT restante fue preparado usando los sitios NdeI/XhoI. El fragmento obtenido fue ligado en los mismos sitios de restricción de un vector pET-TEV vacío para producir el gen D1-L-D2-L. Las construcciones D1-L-D2 y D1-D2-L fueron preparadas a partir de los plásmidos que contienen genes para L-D1-L-D2 y L-D1-D2-L respectivamente, usando el protocolo descrito previamente. Por último, el gen L-D1-D2 fue preparado a partir del plásmido L-D1-L-D2 por digestión y ligación sobre el mismo plásmido usando el sitio de restricción SacI. Todas las construcciones fueron confirmadas por secuenciación de ADN.

3.5. Expresión y purificación de proteínas

3.5.1. Prueba de expresión

Para las pruebas de expresión, se hizo una dilución 1/100 de un cultivo saturado de células transformantes para el plásmido que expresa la proteína de interés en 10 ml de medio LB con el antibiótico kanamicina. Dicho cultivo se incubó con agitación durante 90 min a 37 °C. Luego se retiró 1 ml para control y al resto se lo indujo con IPTG 250 µM. La inducción se dejó transcurrir a distintos tiempos y temperaturas según las condiciones a ensayar, y al finalizar se tomó una nueva muestra de 1 ml del cultivo. Las muestras obtenidas antes y después de la inducción fueron centrifugadas 5 min a 5000 rpm, y los pellets fueron resuspendidos en solución tampón Tris pH 8.0 10 mM EDTA 1 mM. Las muestras fueron sonicadas al 20% de amplitud con 20 pulsos de 1 s seguidos de 9 s de descanso, posteriormente se centrifugó a máxima velocidad durante 10 min a 4° C y el sobrenadante fue corrido en un gel de electroforesis de poliacrilamida. El resultado obtenido da información acerca de la expresión de la proteína en las distintas condiciones ensayadas. Las condiciones de inducción probadas fueron: 4 horas a 37 °C, ON a 37 °C, 4 horas a 25 °C y ON a 25 °C.

3.5.2. Purificación de la proteasa TEV

Para la digestión de las etiquetas de histidinas utilizadas para la purificación de proteínas se utilizó proteasa TEV producida en el laboratorio [78]. Los cultivos de proteasa TEV fueron realizados con células de *E. coli* BL21(DE3)-RIL transformadas el día anterior. Una colonia de la transformación se usó para inocular 10 ml de LB con antibióticos ampicilina y cloranfenicol. El cultivo se dejó crecer ON a 37 °C con agitación. Luego se centrifugó 5 min a 4000 rpm, y el pellet fue resuspendido en 2 ml de LB con los mismos antibióticos. Con este nuevo cultivo se inoculó un nuevo cultivo de mayor volumen con los mismos antibióticos, que al alcanzar una OD_{600}~ 0.6 fue inducido con IPTG 250 mM. La inducción se llevó a cabo durante 4 horas a 25 °C.

Para la purificación de la proteasa TEV se utilizaron las siguientes soluciones:

- O: Tris 50 mM (pH 8), NaCl 500 mM, Glicerol 10 %, Imidazol 20 mM y β-mercaptoetanol 1 mM
- P: Tris 50 mM (pH 8), NaCl 1,5 M, Glicerol 10 %, Imidazol 100 mM y β-mercaptoetanol 1 mM
- Q: Tris 50 mM (pH 8), NaCl 200 mM, Glicerol 10 %, EDTA 2 mM y β-mercaptoetanol 1 mM.

Las células inducidas fueron cosechadas mediante centrifugación a 6000 g por 10 min. El sedimento celular se resuspendió en 25 ml de la solución O y a continuación se sonicaron las células con seis pulsos de 30 s al 40% de potencia. El sobrenadante fue sembrado en una resina de afinidad de Ni (II) y se realizó un lavado con la solución P. La elución se llevó a cabo con un gradiente de Imidazol de 100 a 500 mM. A continuación, se juntaron las fracciones que contenían proteína, se agregó DTT y EDTA 1mM y se dializó contra la solución Q durante toda la noche a 4 °C. Finalmente, se adicionó Glicerol 10 % y DTT 10 mM, se fraccionó y se almacenó en un freezer -20^{o}C.

La evaluación de la actividad de la proteasa se realizó utilizando proteína fusionada a una etiqueta de histidina con el sitio de corte de TEV. Se realizaron cortes de distinta relación proteasa:proteína de fusión, desde 1:300 hasta 1:12. La digestión fue realizada a 4°C durante toda la noche y la evaluación de la actividad se observó a partir de geles de poliacrilamida.

3.5.3. Expresión y purificación de proteínas recombinantes

A partir de una placa fresca de una transformación de células de *E. coli* BL21 (DE3), se inoculó una colonia aislada en 2 ml de medio LB y se incubó durante 6 horas a 37 °C con agitación constante a 220 rpm. 500 µl del pre-cultivo anterior se inocularon en 50 ml de medio M9 (para proteínas con marca isotópica) o LB (para proteínas sin marca) y se incubó por 12 horas a 37°C con agitación constante. Finalmente, los 50 ml de este cultivo se fueron incorporados en 800 ml de medio M9 o LB, según corresponda. Las células fueron incubadas a 37 °C hasta alcanzar una OD_{600} de ~ 0,6. En este punto, se indujo la expresión de las proteínas recombinantes con IPTG 0,25 µM y se dejó crecer el cultivo durante 16 horas a 20°C. Las células inducidas fueron cosechadas mediante centrifugación a 4000 rpm por 10 min. Durante toda la purificación se trató de mantener las muestras en frío.

3.5.3.1. Purificación de proteínas en condiciones nativas

Para los péptidos 3Hx se realizó una purificación en condiciones nativas, para lo cual se utilizaron las siguientes soluciones:

- A: Tris-HCl 50 mM (pH 8), NaCl 500 mM, imidazol 5 mM y β- mercaptoetanol 1 mM

- B: Tris-HCl 50 mM (pH 8), NaCl 500 mM, imidazol 25 mM y β- mercaptoetanol 1 mM
- C: Tris-HCl 50 mM (pH 8), NaCl 500 mM, imidazol 350 mM y β- mercaptoetanol 1 mM
- D: Tris-HCl 50 mM (pH 8), NaCl 500 mM y β-mercaptoetanol 1 mM
- E: PO_4^{-3} 100 mM pH 7, NaCl 50 mM, β-mercaptoetanol 5 mM

El sedimento celular fue resuspendido en 20 ml de la solución A y las células fueron lisadas por sonicación. Se aplicaron 6 pulsos al 40% de potencia de 30 s con un intervalo de 1 min entre cada pulso. Con el objetivo de recuperar las proteínas en solución, el extracto de lisis se centrifugó a 15000 rpm 1 hora. El sobrenadante obtenido se sembró en una columna con 2 ml de resina Sepharose Ni-NTA superflow (QUIAGEN), previamente equilibrada con 5 volúmenes de resina de solución A. Los lavados de la resina se realizaron con 20 ml de solución B para eliminar las proteínas unidas inespecíficamente. Por último, la proteína de interés se eluyó con solución C en fracciones de 1 ml. Las fracciones que contenían proteína fueron dializadas durante dos horas contra 100 volúmenes de solución D para eliminar el imidazol. Previamente se agregó a la muestra proteasa TEV con cola de Histidina para que la digestión ocurra simultáneamente con la diálisis ON. En todos los pasos de diálisis se utilizaron membranas de diálisis de corte 10 kDa. Tanto la proteasa como otras proteínas e impurezas fueron eliminadas mediante una columna de afinidad de Ni(II) equilibrada con solución A. Luego de sembrar la muestra se lavó la resina con 20 ml de solución A para recuperar la mayor parte de la proteína sin etiqueta. La proteína eluída fue concentrada empleando concentradores de membrana de corte por peso molecular de tamaño (10 KDa, Amicon) centrifugando a 6000 g a 4°C. Se realizó en todos los casos un paso extra de purificación sembrando la muestra en una columna de exclusión molecular Superdex G75 (GE) equilibrada en solución E. Las fracciones que contienen proteína fueron colectadas y concentradas.

3.5.3.2. Purificación de proteínas en condiciones desnaturalizantes

Para las construcciones con HYL1, se realizó una purificación en condiciones desnaturalizantes utilizando las siguientes soluciones:

- I: PO_4^{-3} 100mM, Tris-HCl 10 mM, urea 8 M, β- mercaptoetanol 1 mM, pH 8,0
- II: PO_4^{-3} 100mM, Tris-HCl 10 mM, urea 8 M, β- mercaptoetanol 1 mM, pH 6,3
- III: PO_4^{-3} 100mM, Tris-HCl 1 0mM, β- mercaptoetanol 1 mM, urea 8 M, pH 4,5
- IV: PO_4^{-3} 100 mM (pH=7,0), NaCl 50 mM, β- mercaptoetanol 1 mM, Arg 50 mM, Glu 50 mM
- V: Tris 50 mM (pH 8,0), NaCl 500 mM y β-mercaptoetanol 1 mM
- VI: Tris 50 mM (pH 8,0), NaCl 500 mM, Imidazol 5 mM y β-mercaptoetanol 1 mM

- VII: 20 mM HEPES, 500 mM NaCl, 1 mM B-mercaptoetanol, pH 7.0.

El sedimento celular fue resuspendido en 20 ml de la solución I y sonicado para producir la lisis de las células. dicha sonicación fue realizada aplicando 6 pulsos de 30 s al 40% de potencia con un intervalo de 1 min entre cada pulso. Con el objetivo de recuperar las proteínas en solución, el extracto de lisis se centrifugó a 15000 rpm 1 hora. El sobrenadante obtenido se sembró en una columna que contenía 2 ml de resina de Sepharose Ni-NTA superflow (QUIAGEN), previamente equilibrada con 5 volúmenes de resina de solución I. Los lavados de la resina se realizaron con 20 ml de solución II para eliminar las proteínas unidas inespecíficamente. Por último, la proteína de interés fue eluída con solución III en fracciones de 1 ml. Las fracciones que contenían proteína fueron replegadas por diálisis contra 100 volúmenes de solución de replegado IV por 2 horas. En todos los pasos de diálisis se utilizaron membranas de diálisis de corte 10 KDa. A continuación, se procedió a dializar contra 100 volúmenes de solución V y en forma simultánea se realizó la digestión O.N. con proteasa TEV con etiqueta de Histidina producida en el laboratorio. La proteasa, así como otras proteínas e impurezas, fueron eliminadas mediante una columna de afinidad de Ni(II) equilibrada con solución VI. Para recuperar la mayor parte de la proteína sin etiqueta, luego de sembrar la muestra se lavó la resina con 20 ml de solución VI. La proteína eluída fue concentrada empleando concentradores de membrana de corte por peso molecular de tamaño (10 KDa, Amicon) centrifugando a 6000 g a 4°C. Finalmente, se realizó en todos los casos un paso extra de purificación sembrando la muestra en una columna de exclusión molecular Superdex G75 (GE) equilibrada en solución VII. Las fracciones que contienen proteína fueron colectadas y concentradas.

3.5.4. Liofilización

Las muestras de proteína purificada fueron liofilizadas para su transporte. Para esto, se realizó una diálisis a 4 °C ON contra una solución de Acetato de amonio 20 mM pH 6,5 y a continuación se colocaron en un liofilizador hasta la completa eliminación del agua.

3.5.5. Estimación de la concentración de proteínas

La concentración de proteínas fue estimada con espectros de absorción entre 340 y 240 nm en un espectrofotómetro Jasco v-530 con cubetas de paso óptico 1 cm. Los espectros se procesaron en el programa *Jasco Spectra Manager*. En cada caso, se utilizó un coeficiente de absorción a 280 nm obtenido a partir del programa *Protparam* (http://www.expasy.org/tools/protparam.html), utilizando la secuencia primaria de cada

proteína. Para el péptido $H_6$3HxH_6, que no absorbe a 280 nm, la concentración fue estimada a partir de su absorción a 220 nm asumiendo un coeficiente de extinción molar de 150000 $M^{-1}cm^{-1}$ [80].

3.5.6. Electroforesis de proteínas en geles de poliacrilamida

La electroforesis de proteínas en geles de poliacrilamida se realizó en condiciones desnaturalizantes y reductoras. Se utilizó un sistema discontinuo que consiste en un gel de concentración (poliacrilamida 4,5% en Tris-HCl 0,126 M, pH 6,8, SDS 0,26% p/v) seguido por un gel de separación (poliacrilamida 15% en Tris-HCl 0,36 M, pH 8,8, SDS 0,26% p/v). A las muestras se les agregó la solución de siembra 5x (glicerol 5%, SDS 2%, β-mercaptoetanol 0,1%, azul de bromofenol 0,1 mg/ml) y posteriormente fueron calentadas durante 5 min a 100 °C. Para la corrida se utilizó una solución amortiguadora de Tris 0,3% p/v, glicina 1,44% p/v y SDS 0,1% p/v y se aplicó una corriente constante de 180 V. En todos los casos se empleó el sistema Miniprotean 4 (Bio-Rad Laboratories). Luego de la corrida, las proteínas fueron fijadas tratando el gel durante 5 min en una solución decolorante [etanol: ácido acético: agua (30:10:60)] con agitación y luego teñidas con una solución de Azul Brillante de Coomásie R250 1 % p/v en etanol: ácido acético: agua (50:10:40). La decoloración de los geles se consiguió manteniendo el gel en solución decolorante con agitación constante.

3.5.7. Pruebas de replegado y estabilidad

Para optimizar las condiciones de replegado y estabilidad, la proteína fue disuelta en 8M Urea y concentrada hasta una concentración aproximada de 20 mg/ml. Luego, la muestra fue diluida 20 veces en las distintas soluciones tampones a ensayar. Se midieron los espectros de absorción UV correspondientes en el rango de 240 nm a 400 nm a tiempo cero, y luego de transcurridas 2 horas y 18 horas. La estabilidad fue inferida a partir de medidas de concentración a 280 nm y de dispersión a altas longitudes de onda. Las condiciones más estables son las que presentan menor turbidez y mayor concentración de proteína luego de una semana a temperatura ambiente. Las condiciones ensayadas fueron:

- PO_4^{-3} 100 mM, NaCl 50 mM, pH 7,0
- PO_4^{-3} 100 mM, NaCl 500 mM, pH 7,0
- PO_4^{-3} 100 mM, NaCl 50 mM, R mM, E mM, pH 7,0
- PO_4^{-3} 100 mM, NaCl 50 mM, pH 6,0
- PO_4^{-3} 100 mM, NaCl 500 mM, pH 6,0
- PO_4^{-3} 100 mM, NaCl 50 mM, R mM, E mM, pH 6,0
- AcO^- 100mM, pH 4,5

- AcO^- 100mM, pH 4,5
- AcO^- 100mM, pH 4,5
- HEPES 20 mM, NaCl 50 mM, pH 7,0
- HEPES 20 mM, NaCl 500 mM, pH 7,0
- HEPES 20 mM, R mM, E mM, pH 7,0
- HEPES 20 mM, NaCl 50 mM, pH 6,5
- HEPES 20 mM, NaCl 500 mM, pH 6,5
- HEPES 20 mM, R mM, E mM, pH 6,5
- HEPES 20mM, NaCl 50 mM, pH 6,0
- HEPES 20mM, NaCl 500 mM, pH 6,0
- HEPES 20mM, R mM, E mM, pH 6,0

3.6. Técnicas biofísicas

3.6.1. Preparación de proteínas con metales y precursores

Los derivados de Lu(III), Gd(III), Tb(III) y Dy(III) fueron preparados mezclando las proteínas que contienen LBT con los equivalentes correspondientes de $Cl_3Lu.6H_2O$, $Cl_3Gd.6H_2O$, $Cl_3Dy.6H_2O$ y $N_3O_9Tb.5H_2O$. Para los experimentos de proteína en complejo con precursor de miARN se trabajó con muestras del precursor miR172a que habían sido sintetizadas por el Dr. Rasia. Las muestras fueron preparadas en relación 1:1. La secuencia del precursor es: gcUGCUGUGGCAUCAUuucgAUGAUGCUGCAUCGGC.

3.6.2. PELDOR

Las medidas de PELDOR fueron realizadas en el Institute for Integrative Biology of the Cell (I2BC) de CEA Gif-sur-Yvette en Francia en colaboración con el Dr. Leandro Tabares. Para todos los experimentos se utilizó agua MilliQ™.

Las medidas de EPR a 94 GHz fueron realizadas en un espectrómetro Bruker Elexsys II 680 equipado con un criostato de flujo Oxford Instruments CF935. Los espectros de barrido de campo fueron adquiridos a 4.5 K, mientras que las medidas de PELDOR fueron realizadas a 10 K trabajando con un ciclo estándar de cuatro pulsos utilizando la secuencia “dead-time-free DEER sequence” [81]. La reconstrucción de las distribuciones de distancias fue realizada con el método de regularización de Tikhonov.

3.6.2.1 PELDOR *in cell*

Los plásmidos que contienen las distintas construcciones fueron transformados en células *E. coli* BL21(DE3). Las células transformantes fueron crecidas en medio LB toda la

noche a 37 °C. El cultivo saturado fue diluido en medio fresco 100 veces y luego se dejó crecer a 37°C hasta alcanzar una OD_{600}~ 0,6. Se redujo la temperatura a 25 °C y la sobreexpresión fue inducida con 1 mM IPTG. Luego de 30 min de inducción, se agregó $GdCl_3$ hasta alcanzar la concentración final correspondiente. Muestras de 2ml de cultivo fueron colectadas a diferentes tiempos de inducción, centrifugadas por 2 min a 5000 g y lavadas tres veces por resuspensión en 100 mM HEPES pH 8.0, 10% glicerol y 150 mM NaCl y centrifugación por 2 min a 5000 g a 4°C. El pellet resultante fue resuspendido en el mismo buffer y cargado en tubos de EPR de cuarzo estándar (Bruker, tubos para muestras de EPR a 95 GHz con fondo cerrado). Las células fueron compactadas colocando los tubos de EPR en tubos plásticos de 15 ml y centrifugando 2 min a 1000 g. Dado que el pellet resultante (2ul) rellena en forma óptima la cavidad de microonda, no hubo necesidad de extraer el sobrenadante. Para la conservación de la muestra a 200K, el tubo de EPR fue colocado en un criovial y congelado en nitrógeno líquido. Las muestras fueron pre-congeladas en nitrógeno líquido antes de ser introducidas en el equipo (enfriado a 10K) para asegurar que estuviesen congeladas en todo momento.

3.6.2.1.1. Pruebas de sobrevida en presencia de metal

Células de *E. coli* BL21(DE3) transformadas con el plásmido de expresión de L-3Hx-L fueron crecidas en medio LB ON a 37 °C. El cultivo saturado fue diluido 100 veces en medio fresco y se lo dividió en 6 cultivos independientes. Estos nuevos cultivos fueron crecidos a 37°C hasta un OD_{600} ~ 0.6, momento en que la temperatura fue reducida a 25 °C y la sobreexpresión inducida con 1mM IPTG. Luego de 30 min, 3 de estos cultivos fueron suplementados con $GdCl_3$ hasta una concentración final de 500 uM. Se colectaron muestras de todos los cultivos a las 4h de inducción, y se las plaqueó en placas con medio LB-1% agar. El número de unidades formadoras de colonias (UFC) fue cuantificado luego de incubar 24 horas a 37 °C.

3.6.3. Espectroscopia de Resonancia Magnética Nuclear

Los espectros fueron adquiridos a 298 K en un espectrómetro Bruker Avance III 700 MHz equipado con una sonda TXI de triple resonancia (^{1}H, ^{13}C y ^{15}N). Las secuencias de pulsos utilizadas pertenecen a la librería estándar de Bruker. Los experimentos fueron diseñados, ejecutados y sus resultados visualizados con el programa *TopSpin* 2.1 (Bruker). Todos los espectros fueron procesados con nmrPipe [82] y analizados con el programa *CCPNMR Analysis* [83]. Las muestras fueron preparadas agregando 10% v/v de agua deuterada (Cambridge Isotope Laboratories), que se utiliza como referencia interna de frecuencia para la acumulación de los espectros consecutivos.

La asignación de resonancias para ambos dominios dsRBDs de HYL1 había sido obtenida previamente en el grupo de trabajo. Para trasladar dicha asignación a las señales de las variantes con LBT se adquirieron espectros de triple resonancia HNCA y HNCO sobre la muestra L-D1-D2 que permitieron confirmar la identidad de las señales.

La medida de velocidad de relajación transversal (R2) de los ^{15}N amídicos de D1-D2 fue realizada utilizando la secuencia estándar de Bruker. Las intensidades de las señales de espectros adquiridos con distintos tiempos de relajación se ajustaron a la ecuación: $I_t=I_0e^{(-t/T2)}+I_\infty$, siendo I_t la intensidad en altura de la señal y t el tiempo de relajación utilizado en el espectro.

Los cocientes de intensidades de PRE (I/I_0) fueron medidos a partir de la intensidad de las señales del espectro de la proteína en presencia de metal paramagnético Gd(III) vs la intensidad en presencia de metal diamagnético Lu(III). En la Figura 14 se muestra esquemáticamente la diferencia entre los distintos espectros adquiridos. Como se muestra en la Figura 14D a menor distancia entre el núcleo y el metal paramagnético menor es el cociente I/I_0 para dicho residuo.

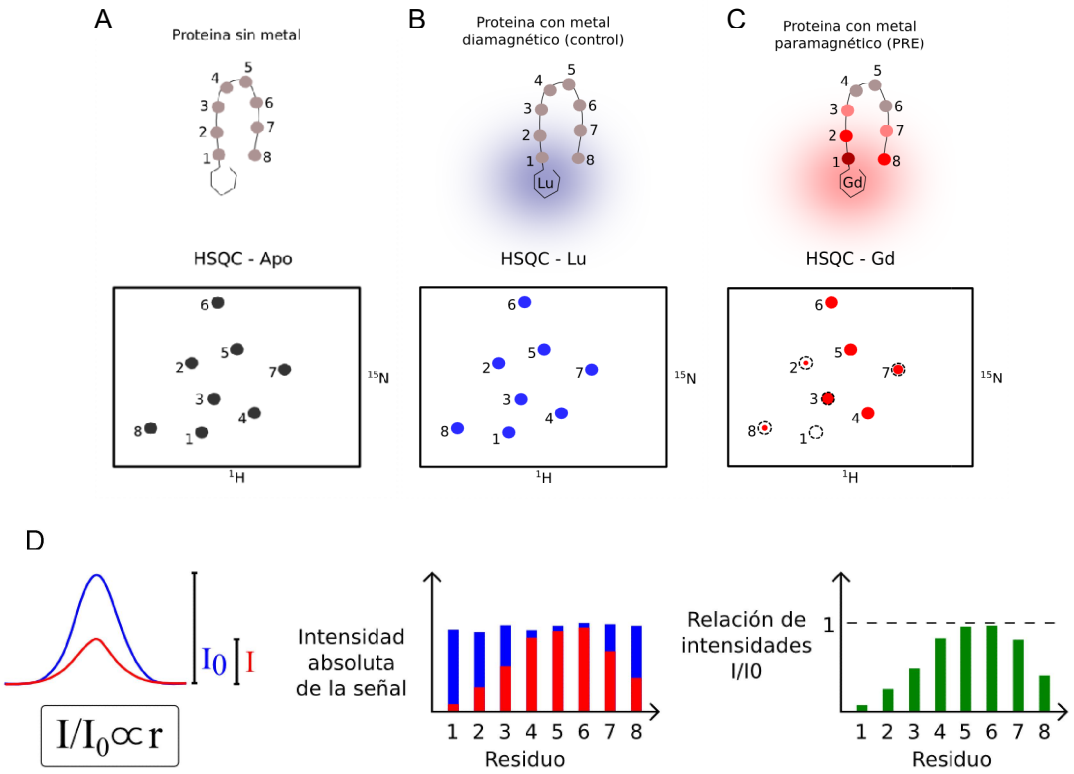


Figura 14: Experimentos PRE. Espectro HN-HSQC para una muestra A) sin metal, B) con metal diamagnético y C) con metal paramagnético. En D) se muestra el cociente I/I_0 calculado, considerando I_0 como la intensidad de las señales en el espectro con metal diamagnético, y a I como la intensidad de las señales en el espectro con metal paramagnético.

3.6.4. Espectroscopia de emisión de luminiscencia

Para estimar la afinidad entre la etiqueta LBT y el metal se midieron espectros de luminiscencia en una titulación de la etiqueta con el metal [84]. Las medidas fueron realizadas sobre muestras de cada proteína a una concentración aproximada de 10 uM en buffer HEPES 20 mM pH 7.0, NaCl 500 mM y β-mercaptoetanol 1 mM. Los espectros de emisión de luminiscencia se adquirieron en un espectrofluorómetro Varian Cary Eclipse. Las muestras fueron tituladas con concentraciones crecientes de Tb(III). La longitud de onda de excitación fue de 280 nm y los espectros de emisión fueron adquiridos en el rango de 300-600 nm. Se integró la banda de emisión de luminiscencia a 544 nm a cada concentración de metal. Los datos obtenidos fueron utilizados para estimar la constante de disociación que caracteriza al sistema (K_D) ajustándolos a la fórmula:

$$IF = IF_0 + Amp \times \left(\frac{([P]+[Tb]+K_D) - \sqrt{([P]+[Tb]+K_D)^2 - 4\times[P]\times[Tb]}}{2\times[P]}\right)$$

donde *[P]* es la concentración de proteína, *[Tb]* es la concentración del ión terbio, K_D es la constante de disociación, *IF* es la integral de la banda de emisión de luminiscencia a 544 nm e IF_0 es la integral de la banda de emisión de luminiscencia a 544 nm en ausencia de metal. Los valores para K_D, IF_0 y *Amp* fueron optimizados en el ajuste de los datos.

3.7. Estudios bioinformáticos

Para los estudios bioinformáticos se trabajó con en lenguaje *Python* 2.7 (Python Software Foundation (http://www.python.org). Los códigos fueron escritos y ejecutados dentro del entorno de Jupyter notebook (https://jupyter.org/). Parte de los códigos utilizados se muestra en el anexo (sección 7.3).

Para la visualización de estructuras en tres dimensiones y para los alineamientos estructurales se utilizó el programa *PyMOL* (The PyMOL Molecular Graphics System, Version 2.0 Schrödinger, LLC.).

3.7.1. Estudio de *linkers*

La información relativa a las proteínas que contienen dominios dsRBDs fue obtenida de la base de datos Uniprot (https://www.uniprot.org). A partir de la misma se generaron tres archivos para cada proteína. A continuación, se muestra el contenido de cada tipo de archivo utilizando como ejemplo la proteína HYL1:

- `uniprot_drbm_50188.fasta` → Nombre de la proteína en formato Uniprot y descripción / secuencia aminoacídica completa

```
>sp|O04492|DRB1_ARATH Double-stranded RNA-binding protein 1
OS=Arabidopsis thaliana OX=3702 GN=DRB1 PE=1 SV=1v
```

MTSTDVSSGVSNCYVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKSLFQSTVILDGVRYNSLPGFFNRK
AAEQSAAEVALRELAKSSELSQCVSQPVHETGLCKNLLQEYAQKMNYAIPLYQCQKVETLGRVTQFTC
TVEIGGIKYTGAATRTKKDAEISAGRTALLAIQSDTKNNLANYNTQLTVLPCEKKTIQAAIPLKETVK
TLKARKAQFKKKAQKGKRTVAKNPEDIIIPPQPTDHCQNDQSEKIETTPNLEPSSCMNGLKEAAFGSV
ETEKIETTPNLEPPSCMNGLKEAAFGSVETEKIETTPNLEPPSCMNGLKEAAFGSVETEKIETTPNLE
PSSCMNGLKEAAFGSVETEKIETTPNLEPPSCMNGLKEAAFGSVETEKIETTPNLESSSCMSGLKEAA
FGSVETEASHA

- uniprot_drbm_50188.tab → Nombre de la proteína en formato Uniprot / nombres y posiciones de los dominio

O04492 DRB1_ARATH DOMAIN 15 84 DRBM 1. {ECO:0000255|PROSITE-ProRule:PRU00266}.; DOMAIN 101 170 DRBM 2. {ECO:0000255|PROSITE-ProRule:PRU00266}

- uniprot_drbm_50188_taxo.tab → Nombre de la proteína en formato Unipot / descripción taxonómica

O04492 DRB1_ARATH cellular organisms, Eukaryota, Viridiplantae, Streptophyta, Streptophytina, Embryophyta, Tracheophyta, Euphyllophyta, Spermatophyta, Magnoliophyta, Mesangiospermae, eudicotyledons, Gunneridae, Pentapetalae, rosids, malvids, Brassicales, Brassicaceae, Camelineae, Arabidopsis, Arabidopsis thaliana (Mouse-ear cress)

Cada uno de los archivos fue sometido a un proceso de *parse* y su contenido asignado a distintas variables de la forma más conveniente en cada caso. El alineamiento de secuencias fue realizado con el método MUSCLE dentro del programa *MEGA* versión 10.0.4 [85]. La construcción del árbol filogenético fue realizada con el método estadístico de Máxima Verosimilitud.

3.7.2. Construcción *in silico* de conformaciones posibles y selección de estructuras

Para el modelado de estructuras de HYL1 con LBTs se utilizó el programa *Modeller* v 9.19 [86].

Para la construcción del ensamble se trabajó con la herramienta Py*Rosetta* [87] (http://www.pyrosettaorg/). La misma es una interfase para el programa de modelado molecular *Rosetta* basada en el lenguaje de programación *Python*.

El proceso de generación de *clusters* a partir de cada matriz de distancia fue realizado con la librería de *Python* scipy.cluster.hierarchy.linkage con el modo “ward”.

4. RESULTADOS

4.1. ANÁLISIS BIOINFORMÁTICO DE *LINKERS* QUE CONECTAN DOMINIOS dsRBDs

Los dominios de tipo dsRBD se encuentran frecuentemente en tándem en las proteínas. La presencia de múltiples dsRBDs en una sola proteína tiene varias funciones. En algunos casos, esta disposición le permite a la proteína aumentar la afinidad de unión al ARNdh por cooperatividad entre ellos. En otros casos, la presencia de múltiples dsRBDs ha permitido una divergencia funcional de los mismos que pasaron a actuar como módulos de reconocimiento proteína-proteína, inter e intraproteína. La proteína HYL1 de *Arabidopsis thaliana* tiene dos dominios dsRBD. Estos dominios, al igual que en todas las proteínas con dominios dsRBD en tándem, están unidos por un *linker*. Para estudiar la función que podría tener esta disposición de dominios, se decidió hacer un estudio basado en el *linker* que los conecta. Para ello se planteó realizar un estudio bioinformático del largo y la secuencia de *linkers* análogos presentes en proteínas de otros organismos. En el caso de encontrar indicios de conservación, se podría suponer que el *linker* es importante para la función de la proteína.

La base de datos Uniprot [88] fue elegida para realizar la búsqueda inicial dado que es líder mundial en el almacenamiento de información sobre proteínas. Entre sus características se destaca que es de acceso libre, está curada (en particular los registros de UniProtKB/Swiss-Prot están curados manualmente), y que cuenta con una interfaz gráfica de fácil manejo. Además, brinda la posibilidad de exportar los resultados obtenidos, lo cual resulta muy útil cuando se manejan grandes volúmenes de datos. Dentro del buscador se ingresó la palabra “DRBM” para obtener una primera aproximación de todas las proteínas disponibles en la base de datos que contenían este tipo de dominio. La búsqueda realizada en noviembre de 2017 arrojó 50.188 resultados de proteínas que se relacionaban con el término de búsqueda introducido. Los nombres asignados en Uniprot, sus secuencias de aminoácidos completas, la ubicación y nombre de cada uno de sus dominios y su descripción taxonómica fueron descargados para cada una de las proteínas obtenidas.

Debido a la necesidad de realizar un análisis específico, para el cual no se contaba con las herramientas adecuadas, se continuó el trabajo mediante la generación de código escrito en *Python*. Las proteínas fueron separadas en grupos en función de características taxonómicas generales. Los grupos definidos y la cantidad de proteínas encontradas para cada uno se describen en la Figura 15.

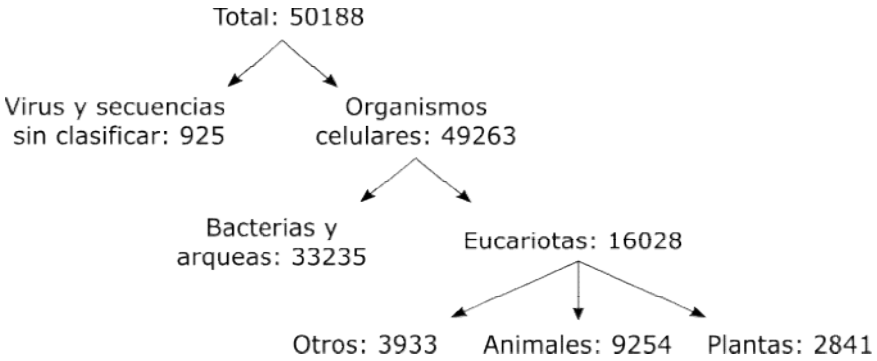


Figura 15: Análisis del total de las proteínas que contienen dominios dsRBD. La primera clasificación fue realizada en base a si eran organismos celulares o no, la siguiente según el tipo celular, y la última en función del reino.

Los dominios recuperados de la base de datos se clasifican a su vez en dos tipos de dominios, denominados “DRBM” y “5S DRBM”. Este último está relacionado con proteínas de ribosomas, por lo que se decidió continuar el resto del análisis considerando únicamente los dominios DRBM. Una vez identificadas las proteínas que corresponden a especies de plantas y de animales, se procedió a separarlas en función de la cantidad de dominios dsRBDs presentes en cada una de ellas. Los resultados se muestran en la Tabla 2.

Cantidad de dsRBD	Proteínas de especies de plantas	Proteínas de especies animales
1	1058	3391
2	1152 (2)	2137(2)
3	101	831
4	3	386
5	0	92
6	0	1

Tabla 2: División de proteínas de plantas y animales según la cantidad de dominios dsRBDs. Entre paréntesis se muestra la cantidad de proteínas que contienen algún otro dominio intermedio entre los dominios dsRBDs.

En este primer análisis observamos la prevalencia de proteínas con más de tres dsRBDs en animales, mientras que la mayoría de proteínas con dsRBDs en tándem de plantas tiene dos dsRBDs.

4.1.1. Análisis de longitud del *linker*

El siguiente paso fue la medición de las longitudes de todos los *linkers* que conectan dominios dsRBDs. Los resultados se muestran en las Figura 16 y Figura 17. El análisis visual de los resultados mostró la preferencia de *linkers* cortos en especies de plantas para proteínas con dos y tres dominios dsRBDs. En particular entre las proteínas con dos dsRBDs el 31% de los casos (359 proteínas) presentó *linkers* con 17 residuos de largo (Figura 16). Este valor contrasta con los resultados en especies de animales (Figura 17), en donde se observa una distribución homogénea de largo de *linkers* en todo el rango de longitudes observada.

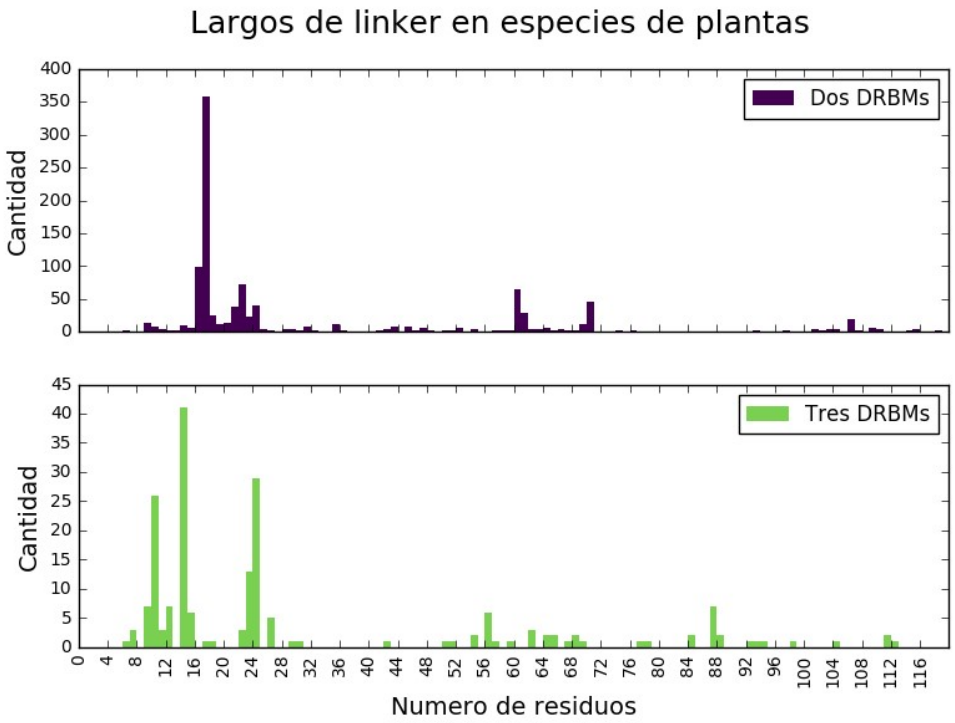


Figura 16: Cantidad de linkers para cada número de residuos en proteínas de especies de plantas.

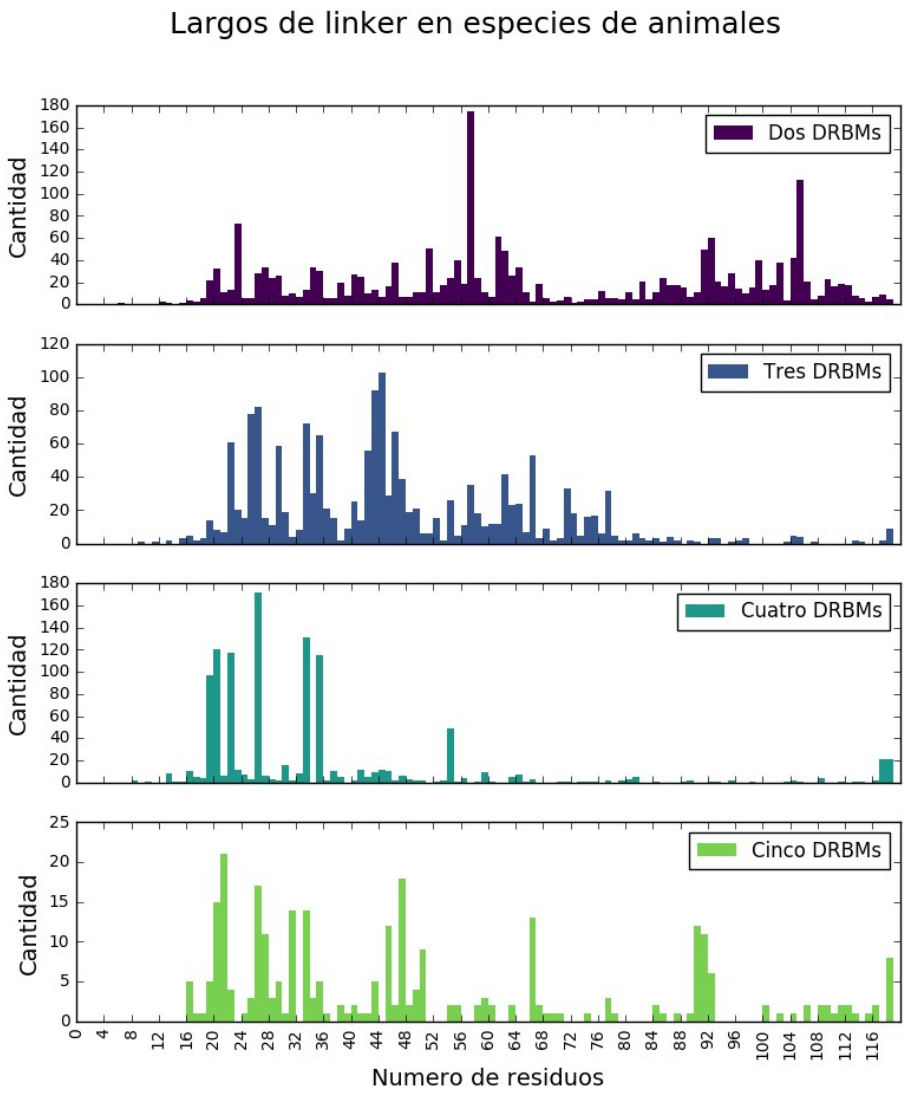


Figura 17: Cantidad de linkers para cada número de residuos en proteínas de especies de animales.

La conservación registrada fue analizada en mayor profundidad. Se seleccionaron aquellas proteínas de especies de plantas que no presentaban otro tipo de dominio en su secuencia, y la cantidad fue reducida a 659. Finalmente se analizó la presencia de duplicaciones de secuencia que podrían corresponder a variantes de la misma proteína. Para ello se buscaron aquellas proteínas con secuencias idénticas desde el comienzo del primer dominio dsRBD hasta el final del segundo dominio dsRBD. Se pudieron eliminar así 101 secuencias repetidas, quedando como resultado 540 proteínas con un 55% (299 proteínas) de *linkers* con largo de 17 residuos. Todos estos resultados se resumen en la Figura 18.

Figura 18: Cantidad de proteínas para cada largo de linker en proteínas de especies de plantas con dos dsRBDs, sin otro tipo de dominio y con secuencias únicas.

Para controlar que la conservación observada esté distribuida entre distintas especies del reino de plantas y no sesgada a un grupo de especies relacionadas se llevó a cabo un análisis taxonómico basado en la clasificación de *A. thaliana* que brinda Uniprot. Dicha clasificación fue estudiada desde la división de reino plantas (*Viridiplantae*) hasta la de orden (*Brassicales*), y se muestra a continuación:

Viridiplantae (reino de plantas), *Streptophyta* (plantas terrestres y algas carofitas), *Streptophytina*, *Embryophyta* (plantas terrestres), *Tracheophyta* (división, plantas vasculares), *Euphyllophyta*, *Spermatophyta* (con semillas), *Magnoliophyta* (clase, con flor), *Mesangiospermae*, *eudicotyledons* (dicotiledóneas), *Gunneridae*, *Pentapetalae*, *rosids*, *malvids*, *Brassicales* (orden)

Para cada división de la taxonomía anterior se cuantificó la cantidad de especies que se desprendían de la clasificación de *A. thaliana*, es decir, que tenían una clasificación diferente para esa división pero que compartían toda la clasificación previa. Entre las proteínas seleccionadas se analizó la presencia o ausencia de *linkers* con 16 o 17 residuos. Los resultados se muestran en la Figura 19.

Se puede ver que los largos de *linker* de 16 y 17 residuos están distribuidos a lo largo de las distintas clasificaciones, sugiriendo que su conservación no está restringida solo a especies relacionadas con *A. thaliana* sino que evolucionó temprano en el desarrollo de las proteínas con dos dominios dsRBDs de plantas.

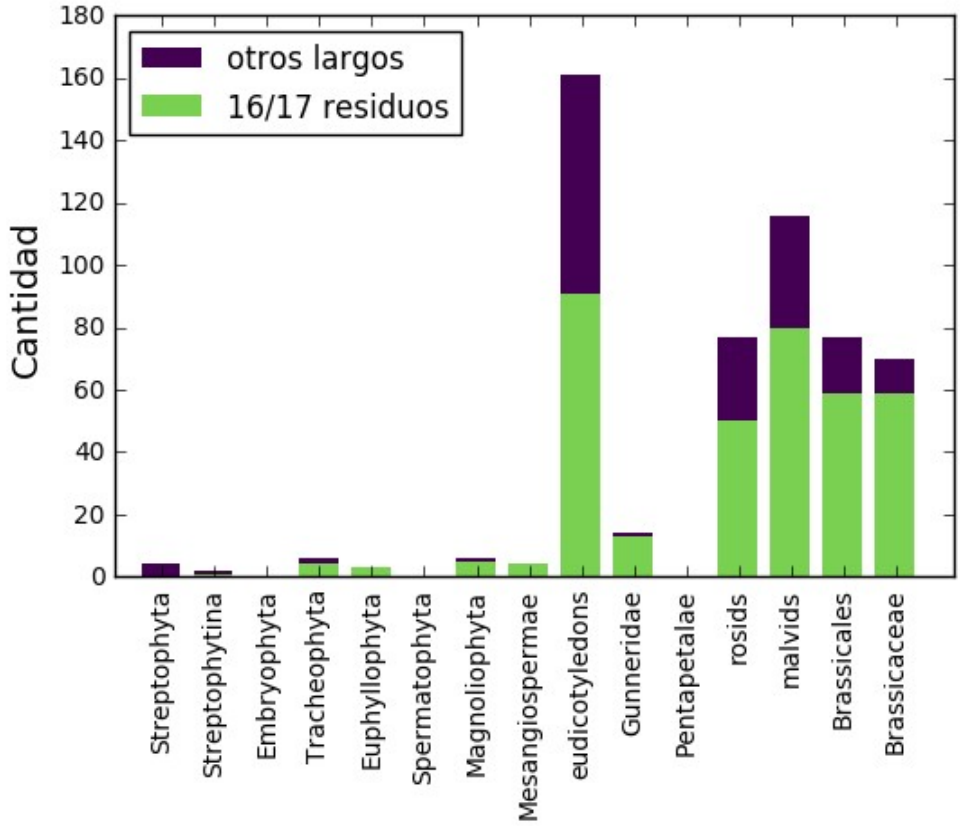


Figura 19: Análisis taxonómico del linker. Para cada clasificación se muestra el número de especies que difieren de A. thaliana.

4.1.2. Análisis de secuencias de *linker*

Considerando la conservación hallada para la longitud del *linker*, se propuso continuar con un análisis a nivel de secuencia. El estudio fue realizado sobre el conjunto de proteínas de plantas con dos dsRBDs y *linker* de 17 residuos de largo, sin otro tipo de dominio y con secuencias únicas. La conservación de la secuencia fue representada mediante la construcción de un logo (Figura 20). En ese tipo de representación, el tamaño relativo de las letras indica su frecuencia para esa posición, por lo que pone de manifiesto el grado de conservación del residuo en esa posición. Se pudo observar que la secuencia C-terminal “(L/V)(D/H)ETG(V/L/I)” está altamente conservada. Esta región se ubica adyacente al segundo dominio dsRBD (D2), lo que sugiere que en realidad forma parte del mismo pero no es identificado por diferir del conjunto de los dsRBDs. A partir de este análisis podemos proponer que esta secuencia es una firma de los segundos dsRBD de proteínas de plantas con dos dsRBD en tándem.

En cuanto al resto de la secuencia, existe un considerable nivel de conservación. Sin embargo, la secuencia correspondiente al *linker* de HYL1 no es la secuencia más conservada.

Figura 20: Logo de secuencias para linkers de 17 residuos en proteínas de especies de plantas. En amarillo se muestra la secuencia de HYL1 de A. thaliana.

4.1.3. Análisis de secuencia de según el tipo DRB

Las únicas proteínas de especies de plantas obtenidas desde la base de datos de Uniprot que permiten una identificación directa a partir de su nombre pertenecen a la familia DRB de las especies modelos *Arabidopsis thaliana* y *Oryza Sativa*. Esto lleva a suponer que, probablemente, las proteínas restantes forman parte de la misma familia, pero en especies menos estudiadas, y por lo tanto no han sido clasificadas como DRBX. Los distintos tipos de DRB cumplen distintas funciones dentro de la célula y esto podría reflejarse en las secuencias de sus *linkers*. Por todo esto se decidió continuar con un análisis a nivel de secuencia considerando el tipo de DRB. Recordamos en este punto que HYL1 se denomina alternativamente DRB1. Dado que la longitud de *linker* de 16 residuos también presentó un considerable nivel de conservación, las proteínas con dicha longitud de *linker* (70 proteínas) fueron incorporadas en el estudio.

Para el análisis de secuencia en base al tipo DRB son necesarios dos pasos. Primero, hay que dividir en grupos a partir de un alineamiento de secuencia para identificar miembros de distintas familias. Para ello, las secuencias de las 369 proteínas con *linkers* de 16 y 17 residuos fueron alineadas, y el alineamiento obtenido fue utilizado para la construcción de un árbol, con el cual fue posible separar a la mayoría de las secuencias en 4 grupos principales (con 36, 98, 101 y 127 proteínas, total 362). El segundo paso es determinar el tipo de DRB contenido en cada grupo. Considerando que, como fue mencionado previamente, para la mayoría de las proteínas de este estudio no se contaba con información relativa al tipo DRB (únicamente estaban identificadas 10 proteínas), se recurrió a la información disponible en un trabajo similar realizado por Clavel y colaboradores [17]. En ese trabajo se publicó la secuencia completa, el organismo al que pertenece y el tipo DRB para 133 proteínas. Se procedió a utilizar esta información para

poder identificar proteínas dentro de los grupos formados a partir del alineamiento. Luego de comparar ambas bases de datos, fue posible identificar tipos DRBs para 45 proteínas, resultando un total de 55 proteínas con tipo DRB identificados. Se pudo observar una correlación entre los 4 grupos generados en el alineamiento y la clasificación de DRBs.

Cuando se trabaja con bases de datos pueden presentarse duplicaciones originadas por variantes de la misma proteína que fueron identificadas como diferentes o por fallas en los procesos de curado. Por ello, se consideró importante eliminar posibles repeticiones entre las secuencias de trabajo. Se buscaron aquellas proteínas que pertenecían al mismo organismo y que presentaban una distancia dentro del árbol menor a 0.1. Para cada par encontrado se conservó solo una de ellas. Con este procedimiento fueron eliminadas 12 proteínas. Los grupos quedaron conformados con las siguientes cantidades: 88 proteínas para DRB1, 100 proteínas para DRB2, 36 proteínas para DRB6, y 126 proteínas para DRB3 y DRB5 juntos. Los logos de las secuencias de los *linkers* para cada grupo según el tipo DRB se muestran en la Figura 21. Se puede observar que el grupo DRB1 es el que tiene menor conservación de secuencia. También se encontró variabilidad en la distribución de longitudes de *linkers*. Mientras que casi todas las del tipo DRB6 tienen *linkers* de 16 residuos, todas las DRB2, 3 y 5 los tienen de 17 residuos de largo. Las DRB1, por su parte, presentan aproximadamente un tercio de secuencias con largos de 16, siendo el resto secuencias de 17 residuos.

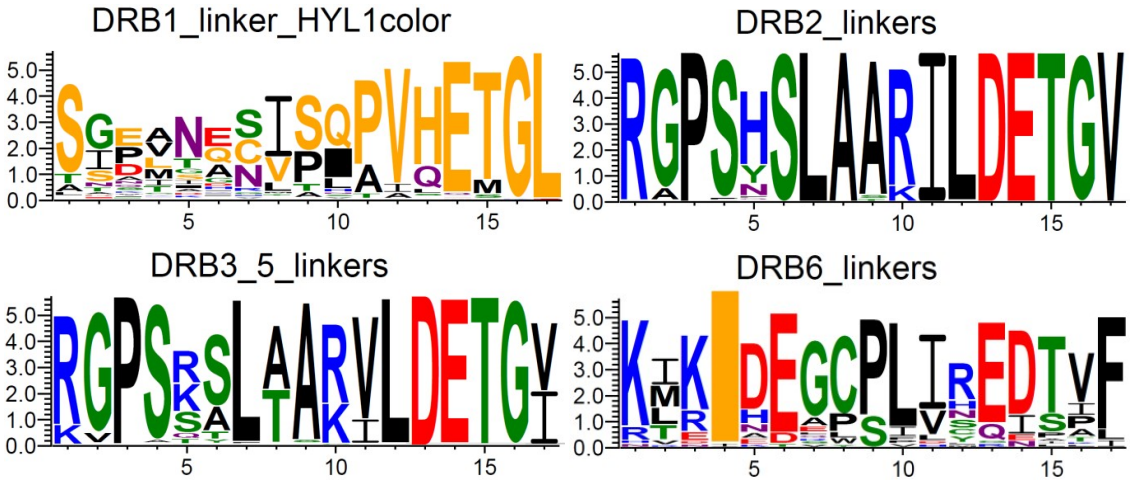


Figura 21: Logo de secuencias para linkers de 16 y 17 residuos en proteínas de especies de plantas para cada tipo de DRB. Los gaps de los alineamientos se muestran en color negro en el gráfico de DRB1 (posición 10), y en amarillo en el de DRB6 (posición 4).

Posteriormente se analizó el promedio de la identidad de secuencia de manera independiente para cada dominio dsRBD y para el *linker* dentro de cada grupo DRB. Los

resultados se muestran en la Figura 22. Se puede observar que el caso de DRB1 es particular, ya que la conservación de secuencia en sus dominios fue mucho mayor que la hallada para su *linker*.

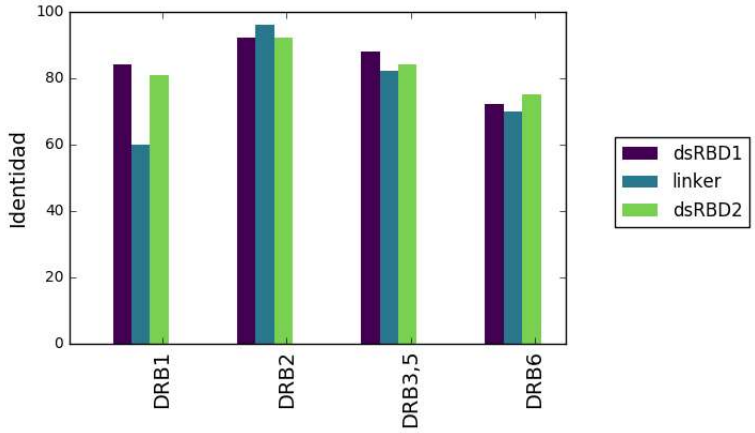


Figura 22: Identidad de secuencia para cada tipo de DRB y cada región de la proteína.

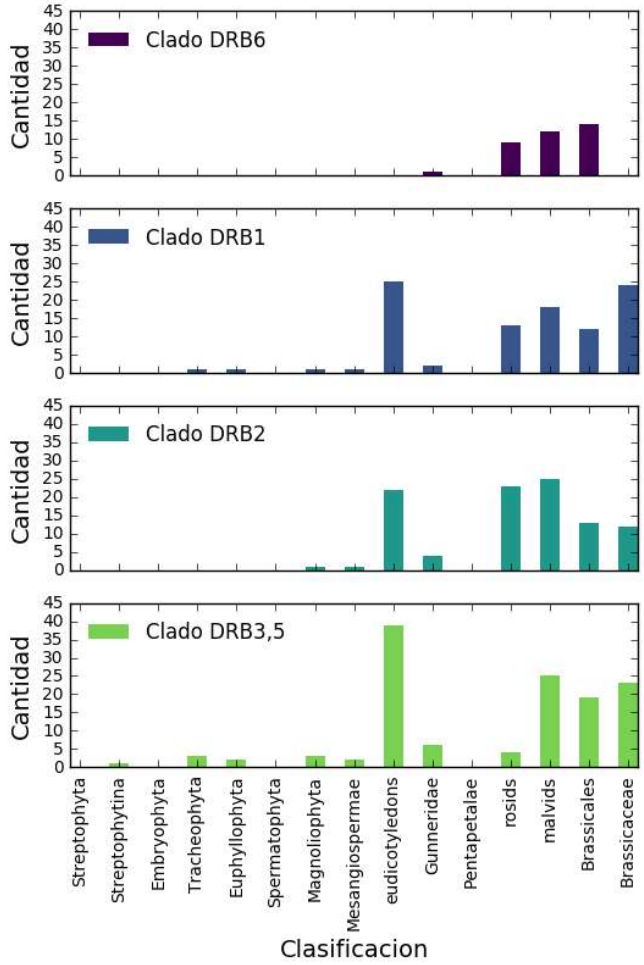


Figura 23: Análisis taxonómico del linker por grupos. Para cada clasificación se muestra el número de especies que difieren de A. thaliana.

Para confirmar que los *clusters* no estén sesgados a un grupo de especies relacionadas se hizo un análisis taxonómico análogo al realizado durante el estudio de la longitud del *linker*. Nuevamente, se encontró una variada distribución taxonómica (Figura 23).

4.1.4. Conclusiones del análisis bioinformático de *linkers* entre dsRBDs

Los dominios dsRBD dispuestos en tándem poseen secuencias *linker* que los conectan. Para comprender mejor el rol de HYL1 dentro del complejo de microprocesamiento se hizo un análisis de *linkers* presentes en proteínas análogas de distintos organismos. El análisis del largo de *linker* para proteínas con dos dominios dsRBD mostró una elevada conservación de largos de 16/17 residuos en proteínas de especies de plantas. Esta propiedad no se repite para proteínas de especies animales, por lo que podrían ser clave para explicar diferencias funcionales entre proteínas de los distintos reinos. Las proteínas de especies de plantas con *linkers* de longitud conservada fueron divididas en tipos de DRB en base al alineamiento de secuencia de sus *linkers*. El análisis de identidad de secuencia mostró que existe una desigualdad en la identidad de secuencia entre dominios y *linkers* solamente para las del tipo DRB1. Considerando que las proteínas DRB1 participan en un proceso esencial, la biogénesis de miARNs, y que su ausencia muestra el fenotipo más importante entre las distintas DRBs, podemos sugerir que el *linker* de proteínas DRB1 podría ser importante para el funcionamiento de este tipo de proteínas.

4.2. GENERACIÓN DE CONSTRUCCIONES Y PRODUCCIÓN DE PROTEÍNA

El estudio bioinformático de diferentes proteínas de especies de plantas con dominios dsRBD demostró que existe una conservación en la longitud del *linker*. A su vez, la menor conservación de secuencia para los *linkers* de proteínas DRB1 podría ser importante para determinar el tipo de complejo de procesamiento. Considerando estos resultados, se decidió estudiar cómo puede influir el *linker* de HYL1 en la exploración del espacio conformacional de la proteína. Un análisis de las dimensiones relativas de los dominios y del *linker* (Figura 24) sugiere que la longitud de 17 residuos podría limitar el rango de conformaciones posibles adquiridas por los dominios. Para evaluar esto, se planteó realizar un análisis funcional a través del estudio del muestreo conformacional entre los dos dominios.

Para estimar el espacio conformacional explorado por los dos dominios dsRBD es necesario realizar medidas experimentales de distancias. Las estructuras resueltas muestran que los dominios dsRBD tienen forma de elipsoide prolato, con un eje mayor de aproximadamente 4 nm. Por su parte, la longitud del *linker* extendido, estimada por simulación computacional, es de *ca.* 2 nm (Figura 24). Resulta evidente que las técnicas a utilizar para medir la distancia entre dominios deben ser capaces de dar resultados en la escala de nanómetros. Entre las técnicas biofísicas disponibles para medir en esa escala, resulta apropiado el uso de aquellas que utilizan especies paramagnéticas.

HYL1 no posee especies paramagnéticas en forma natural, por lo cual fue necesario diseñar variantes para la incorporación de las mismas. Dado que la región N-terminal de HYL1, que contiene los dos dominios dsRBD, es suficiente para complementar los defectos en el fenotipo de plantas *hyl1*, las construcciones planteadas en este trabajo abarcan secuencias hasta el final del segundo dominio dsRBD (residuo 170). De aquí en adelante este trabajo se refiere a la región de HYL1 que abarca los residuos 1-170 de la proteína completa como D1-D2. Todas las proteínas fueron expresadas en cultivos crecidos en medios LB o M9 suplementado con isótopos, según el caso. La purificación se realizó con columnas de cromatografía de afinidad de Ni(II)NTA-agarosa y de exclusión molecular, con un paso intermedio de corte de la cola de histidina con la proteasa TEV.

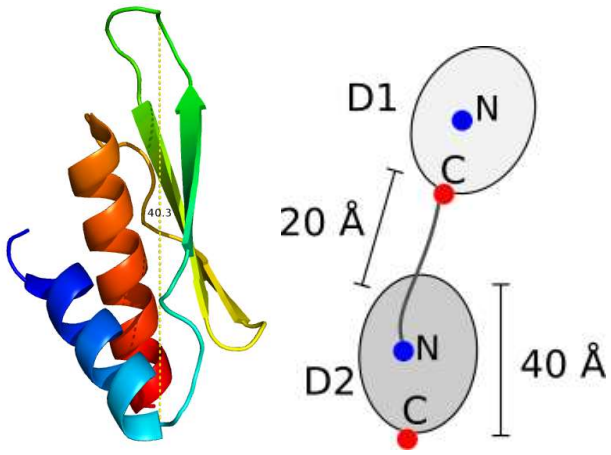


Figura 24: Distancias estimadas. A la izquierda estimación del largo del dominio dsRBD sobre D1 (PDB 3adg). A la derecha esquema de D1-D2 con las dimensiones estimadas.

4.2.1. Construcciones de mutantes para la unión de radicales a cisteínas

Para la incorporación de especies paramagnéticas sobre D1-D2 se propuso inicialmente la unión covalente de sondas con radicales nitróxido a residuos de cisteína. El uso de este tipo de sondas requiere que las únicas cisteínas disponibles para reaccionar estén posicionadas en lugares precisos que determinarán las distancias a medir. El primer paso fue, por lo tanto, el reemplazo de las cinco cisteínas nativas de D1-D2 (Figura 25, espectro en la Figura 26A) por serinas (D1-D2_noCys). La estructura de las mutantes fue evaluada por espectros $^{1}H^{15}N$-HSQC. Estos espectros muestran una señal separada por cada residuo de la proteína. La posición de las señales está determinada por la conformación e interacciones de cada residuo y es sensible a variaciones muy sutiles. Por esta razón los espectros $^{1}H^{15}N$-HSQC son una huella dactilar de la estructura de la proteína de fácil interpretación.

El espectro de la proteína purificada mostró que solamente D1 mantuvo su estructura, mientras que D2 aparece desplegado (Figura 26B). En la búsqueda de una construcción con mayor estabilidad estructural, se decidió reemplazar los residuos mutados en ese dominio por valina o alanina, para conservar el carácter parcialmente hidrofóbico del grupo -SH. Entre las variantes generadas, la mutante S105V (D1-D2_noCys_S105V) recupera el plegamiento en el segundo dominio (Figura 26B). El siguiente paso fue la incorporación de un residuo de cisteína en cada dominio. Las posiciones seleccionadas fueron elegidas en base al análisis de la estructura de cada dominio, considerando la ubicación de las cadenas laterales de las posiciones candidatas y evitando las regiones conocidas de unión a ARN (Figura 27) [16]. Las construcciones generadas fueron D1-D2_noCys_S105V_K31C para el primer dominio y D1-D2_noCys_S105V_T155C para el

segundo. Los espectros $^{1}H^{15}N$-HSQC mostraron señales correspondientes a ambos dominios (Figura 26C). Sin embargo, la proteína que había incorporado una nueva mutación en el segundo dominio presentó una alta inestabilidad que condujo a importantes pérdidas por precipitación durante la adquisición de los espectros. La baja estabilidad de estas proteínas sugiere que las cisteínas de D2 podrían ser importantes para el correcto plegamiento de la estructura y llevó a abandonar esta estrategia. Esto planteó la necesidad de buscar alternativas para la incorporación de sondas paramagnéticas en D1-D2.

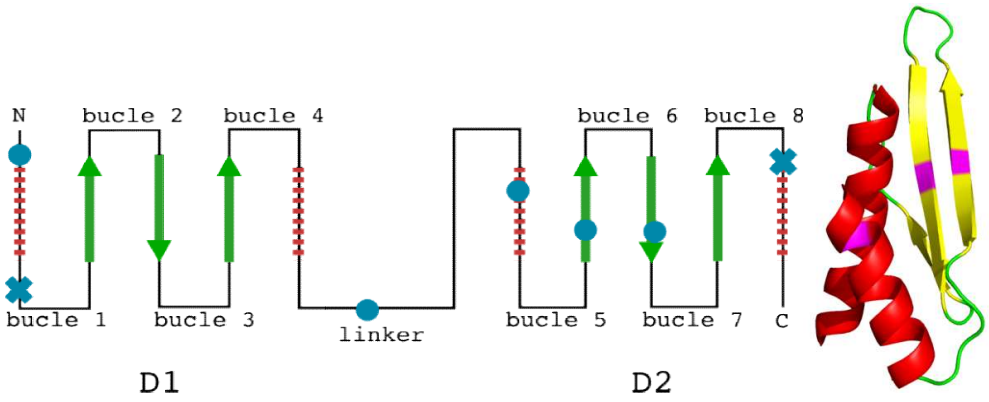
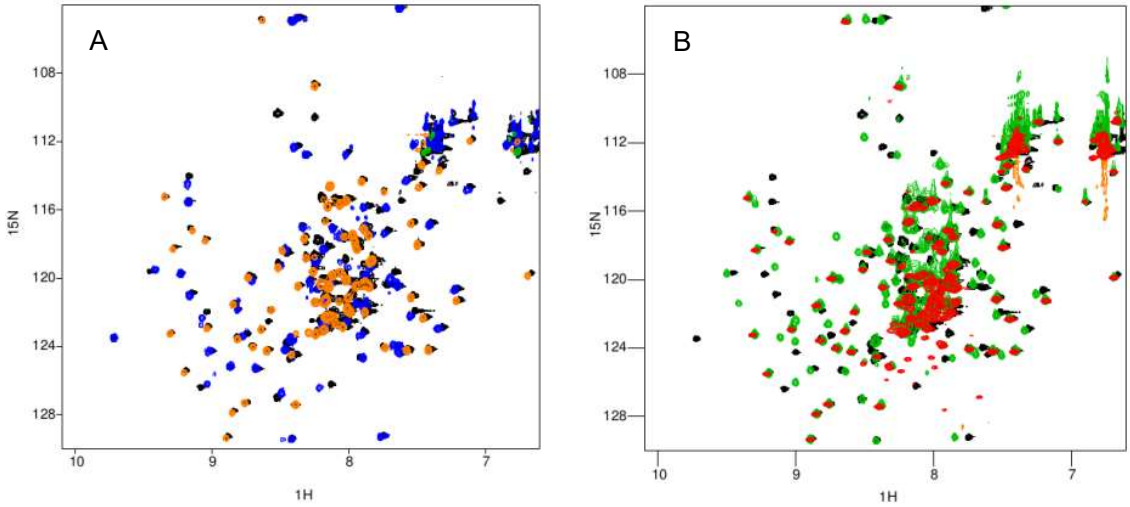


Figura 25: Cisteínas en HYL1. Izquierda: Esquema de las posiciones de las cisteínas nativas y mutantes sobre las construcciones de D1-D2. Las regiones en rojo corresponden a hélices alfa y en verde a láminas beta. Las posiciones de las cisteínas nativas se muestran como círculos azules y las de las incorporadas como cruces azules. Los bucles siguen la nomenclatura usada durante este trabajo. Los dominios dsRBD se unen al ARN a través de las regiones (N-terminal y bucles) ubicadas en la parte superior del esquema. Derecha: estructura de D2 con hélices alfa en rojo y láminas beta en amarillo, las cisteínas nativas se muestran en violeta.



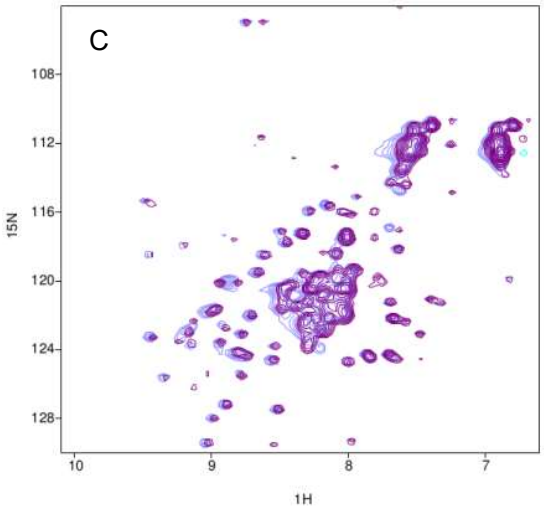


Figura 26: Espectros 1H15N-HSQC para las distintas construcciones. A) Los dominios son estructuralmente independientes (D1 se muestra en color naranja, D2 en color azul y D1-D2 en negro). B) El reemplazo de las cisteínas nativas desestabilizó D2, dicho dominio recuperó su plegamiento en la construcción S105V (D1-D2 en negro, D1-D2_noCys en rojo y D1-D2_noCys_S105V en color verde). C) Construcciones con una cisteína mutante (D1-D2_noCys_S105V_K31C en color violeta y D1-D2_noCys_S105V_T155C en color celeste).

4.2.2. Construcciones con etiquetas de unión a metales

Debido a que la mutación de residuos nativos para la eliminación o incorporación de cisteínas dio lugar a proteínas inestables, se decidió probar con otro tipo de sondas paramagnéticas. Se propuso entonces el uso de etiquetas para la unión de metales paramagnéticos, particularmente de LBTs para la unión de lantánidos [55]. Para disminuir la probabilidad de generar constructos inestables por la interrupción de estructuras secundarias, se decidió la incorporación de las secuencias LBTs en bucles dentro de los dominios [56]. Los bucles fueron seleccionados evitando la cara del dominio que interacciona con el ARNdh (Figura 27) [12]. Las posiciones seleccionadas fueron denominadas bucle 3, bucle 5 y bucle 7 (ver nomenclatura en la Figura 25).

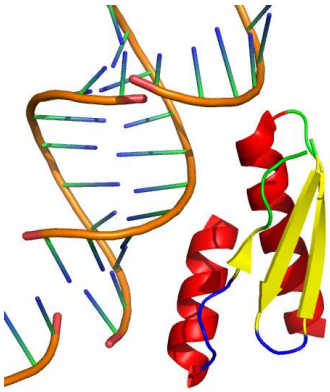


Figura 27: Posiciones de los bucles del dominio dsRBD con respecto al ARNdh en un complejo formado entre ambos. En azul se muestran los bucles que no interaccionan con el ARNdh (bucles 1 y 3 para D1 y bucles 5 y 7 para D2) y en verde los que sí interaccionan con el ARNdh (bucles 2 y 4 para D1 y bucles 6 y 8 para D2. (PDB 1DI2).

Fueron clonadas con éxito tres variantes en donde la secuencia LBT reemplaza algunos aminoácidos del bucle 3, 5 o 7. Sin embargo, las proteínas con LBT en los bucles 5 y 7 fueron inestables y no fue posible purificarlas para ser estudiadas por RMN. Por otro lado, la construcción con un LBT en el bucle 3 pudo ser purificada, pero su espectro $^{1}H^{15}N$-HSQC indica que se encuentra en conformaciones heterogéneas y en intercambio. Es interesante verificar que si bien la inserción se encuentra en D1, el espectro demuestra que también se desestabiliza D2 (Figura 28). Esto pone de manifiesto que, a pesar de ser aparentemente independientes, el *linker* flexible induce una interdependencia estructural de los dos dominios.

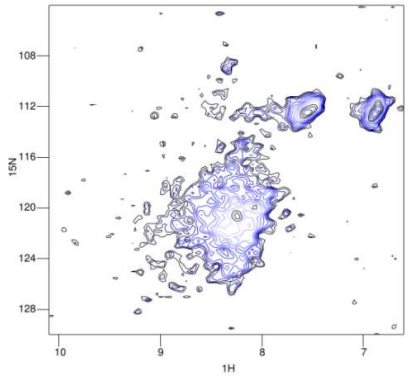


Figura 28: Espectro 1H15N-HSQC de D1-D2 con un LBT en bucle 3.

La dificultad para obtener construcciones estables puede deberse a que el LBT sea demasiado grande como para ser incorporado en un bucle sin afectar al resto de la estructura. Se decidió planificar nuevas construcciones en las que el LBT interfiera lo menos

posible con el plegado de cada dominio. Los sitios seleccionados para la unión de la etiqueta en las siguientes construcciones fueron el extremo N-terminal de D1, el extremo N-terminal del *linker* y el extremo C-terminal de D2. Estas posiciones no habían sido consideradas en un principio para evitar que la flexibilidad natural de dichas regiones afectase a la distribución de distancias que se quería analizar. Para confirmar la factibilidad de utilizar estos sitios y además poner a punto la técnica de PELDOR se decidió estudiar el uso de LBTs en los extremos de una estructura modelo simple. Los experimentos fueron realizados sobre una estructura de tres hélices alfa denominado "3-helix-bundle" (3Hx) diseñada por Huang y colaboradores [79]. Se generaron construcciones de 3Hx que contienen en sus extremos LBTs (L) para la unión de Gd(III) para estudios tanto *in vitro* como *in cell*, quedando conformadas así las construcciones 3Hx, L-3Hx, 3Hx-L, L-3Hx-L y H6-L-3Hx-L. Todas las variantes fueron expresadas y purificadas según el caso para llevar a cabo los experimentos que se detallan en la sección siguiente.

Una vez confirmada la utilidad de los LBT ubicados en los extremos de una estructura a medir se continuó en el diseño de las variantes de D1-D2. Se excluyeron los primeros 14 aminoácidos de D1, ya que los mismos no forman parte de la estructura secundaria y podían aumentar la flexibilidad en ese extremo. Todas las combinaciones de dobles LBT, así como las versiones con un solo LBT, fueron generadas a partir de un gen sintético adquirido para ese fin, tal y como se detalla en materiales y métodos (sección 3.4.3). En la Figura 29 se esquematizan las distintas construcciones, y la información relativa a cada una se muestra en el anexo (sección 7.2). Con la construcción L-D1-D2 se llevó a cabo una prueba de estabilidad en distintos buffers para seleccionar la mejor condición de trabajo (ver en materiales y métodos, sección 3.5.7.) La condición seleccionada fue HEPES 20 mM, NaCl 500 mM, pH 6.5, la cual fue utilizada tanto para la purificación final de las proteínas como para los experimentos de PRE.

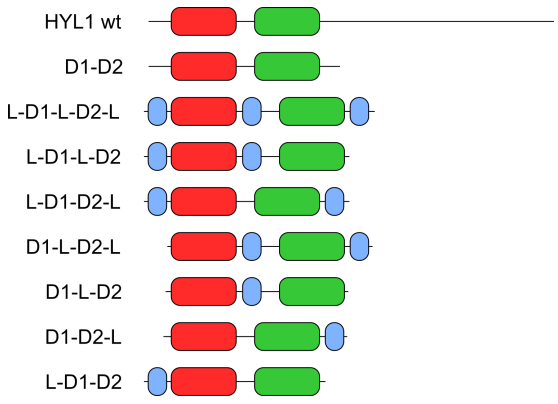


Figura 29: construcciones derivadas de HYL1 con etiquetas LBT. D1 se muestra en rojo, D2 en verde, y el LBT en azul.

Para confirmar la viabilidad del uso de LBTs en las concentraciones de trabajo se llevaron a cabo medidas de afinidad de unión de la etiqueta a lantánidos. Para ello se cuantificó la transferencia de excitación del Triptófano (Trp) presente en el LBT al lantánido Terbio (Tb(III)) por espectroscopía de luminiscencia. Dicha transferencia de excitación requiere que ambos fluoróforos se encuentran suficientemente cerca. Si el LBT no tiene cargado al metal no se verá emisión de luminiscencia a la longitud de onda de emisión. En cambio, si el metal se encuentra unido a la etiqueta habrá una transferencia de excitación por un mecanismo no radiante desde el Trp excitado al metal, y este último emitirá luminiscencia. Se cuantificó la intensidad de emisión de luminiscencia de Tb(III) durante una titulación de la proteína con el metal, y los datos obtenidos fueron ajustados a una ecuación de unión de ligando a un sitio único (sección 3.6.4). Las constantes de disociación (Kd) obtenidas del ajuste se muestran en la Tabla 3. Los valores obtenidos se encuentran en el rango de 1-9 uM, y son suficientes para el trabajo tanto en RMN como en EPR. La menor afinidad registrada con respecto a reportes previos para la misma secuencia de LBT (secuencia dSE3 Kd 3.6 nm [55]) podría deberse a que las construcciones utilizadas en esos trabajos correspondían a construcciones de LBT aislados, mientras que en el presente trabajo la estructura proteica podría estar dificultando el acceso del metal a la etiqueta.

Construcción	Kd Tb(III)
L-D1-D2-L	8.7 uM
L-D1-L-D2	5.55 uM
D1-L-D2-L	6.62 uM
D1-D2-L	0.77 uM
L-D1-D2	3.55 uM

Tabla 3: Constantes de afinidad medidas por luminiscencia de Tb(III). D1 corresponde al primer dominio dsRBD, D2 al segundo dominio dsRBD y L a la secuencia LBT.

4.2.3. Conclusiones de la generación de construcciones y producción de proteína

La medida de distancias entre dominios, en el rango de nm, requiere el uso de sondas paramagnéticas. Se exploraron distintas estrategias para su incorporación a proteínas. La mutación de residuos de cisteína para el uso de radicales nitróxido, así como la incorporación de secuencias LBT en los bucles de los distintos dsRBDs dio lugar a proteínas inestables. La única alternativa viable resultó ser la incorporación de los LBTs en los extremos de las estructuras. Se generaron construcciones sobre el sistema modelo 3Hx

con etiquetas LBT para la puesta a punto del sistema tanto *in vitro* como *in cell*. Por otra parte, se produjeron con éxito construcciones de D1-D2 que contienen LBTs para su uso en los estudios posteriores.

4.3 ESTUDIOS POR PELDOR

4.3.1. Puesta a punto con sistema 3Hx

4.3.1.1. PELDOR *in vitro*

La puesta a punto de la técnica se llevó a cabo con construcciones que derivan de la estructura modelo 3Hx con etiquetas de LBTs unidas al metal paramagnético Gd(III) en los extremos. A partir del modelado computacional realizado por el Dr. Tabares para la construcción L-3Hx-L se estimó la distancia entre metales en 4.3 nm con un ancho a mitad de la altura de 1.0 nm (Figura 30).

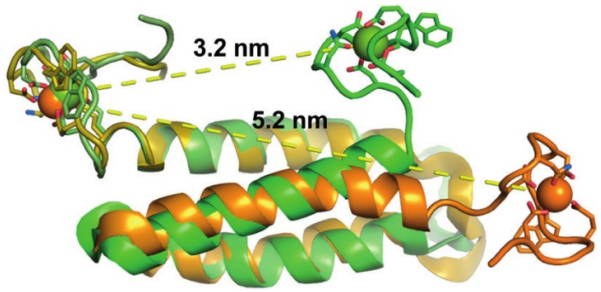


Figura 30: Estructura de L-3Hx-L en complejo con los iones metálicos obtenidas por dinámica molecular. Se muestran las disposiciones con la mayor y la menor distancia entre metales.

El espectro de EPR de barrido de campo sobre una muestra purificada de L-3Hx-L con dos equivalentes de metal se muestra en la Figura 31. El Gd(III) tiene un espín electrónico S=7/2, y sus isótopos más abundantes (^{156}Gd, ^{158}Gd y ^{160}Gd, con un total de 67%) tienen un espín nuclear I=0, por lo que su espectro está compuesto de siete transiciones. Sin embargo, en el rango estudiado solo se puede apreciar un único pico correspondiente a la transición central -½ → ½. Los “hombros” a los costados de la señal principal son indicios de la presencia de Gd(III) unido a la etiqueta. El experimento de PELDOR sobre la misma muestra (Figura 33) dio una distribución a 3.8 nm con un ancho a mitad de la altura de 1.2 nm.

4.3.1.2. PELDOR *in cell*

El mismo sistema fue utilizado para realizar medidas *in cell* [89]. En forma resumida, cultivos de *E. coli* transformados con los distintos plásmidos de expresión fueron crecidos hasta un OD_{600} de aproximadamente 0.6. En este punto los cultivos fueron inducidos e incubados a menor temperatura. El agregado de metal al medio fue realizado media hora después de haber incorporado el inductor. Una vez finalizado el proceso de inducción, las células fueron recolectadas y lavadas para eliminar el metal que no ingresó al interior

celular. Luego, la muestra fue compactada dentro del tubo de EPR y congelada, quedando preparada para adquirir los espectros. Los espectros de onda continua para cultivos a distintas concentraciones de metal en el medio, y a distintos tiempos de inducción total, se muestran la Figura 31. En todos ellos se puede apreciar las seis líneas correspondientes al Mn(II) presente de forma natural en el medio intracelular. Todas las construcciones con LBT mostraron en sus señales de Gd(III) los hombros característicos de metal unido a la etiqueta. La construcción de 3Hx sin LBT, en cambio, presentó una señal sin hombros que se corresponde con el metal libre. En base a la intensidad de las señales de Gd(III) obtenidas se pudo determinar que el equilibrio en la incorporación de metal al interior celular se consigue con una concentración 500 uM de Gd(II) en el medio de cultivo y con cuatro horas de inducción total.

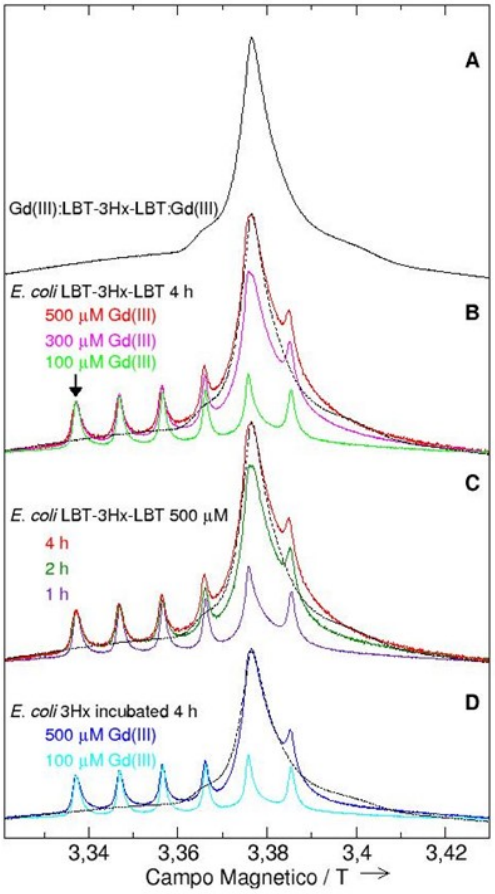


Figura 31: Espectros de EPR a 94 GHz y 4.5 K de L-3Hx-L A) purificada (se muestra como una línea discontinua negra en los otros espectros). B) en células de E. coli luego de 4 horas de inducción en medio suplementado con las concentraciones de Gd(III) indicadas. C) luego de distintos tiempos de inducción en medios suplementados con 500 uM de Gd(III). D) Espectros de 3Hx en células E. coli luego de 4 horas de inducción en medios suplementados con las concentraciones de Gd(III) que se indican. Todos los espectros fueron normalizados según las intensidades de las señales de Mn(II).

Los resultados de los experimentos de PELDOR *in cell* se muestran en la Figura 32, y las señales luego de la corrección de línea de base en la Figura 33. Las proteínas L-3Hx y 3Hx-L no presentaron modulación, indicando que la estructura no dimeriza. Para confirmar que la modulación observada corresponde a interacciones entre Gd(III) independientes de la presencia de Mn(II), se realizó el experimento control pulsando a la frecuencia de este último (flecha en Figura 31). La ausencia de modulación confirma que el resto de los experimentos responden a medidas que involucran únicamente Gd(III). La medida sobre la construcción con dos LBT en medio con 500 uM de Gd(III) dio una modulación de 0.3%, menor respecto al 0.6% observado *in vitro*. La menor modulación observada puede deberse a interacciones dipolares con otras especies, en particular Mn(II) nativo y Gd(III) libre. El análisis de Tikhonov dio una distancia de 3.8 nm con una distribución de 1.5 nm, lo cual coincide con el experimento *in vitro*.

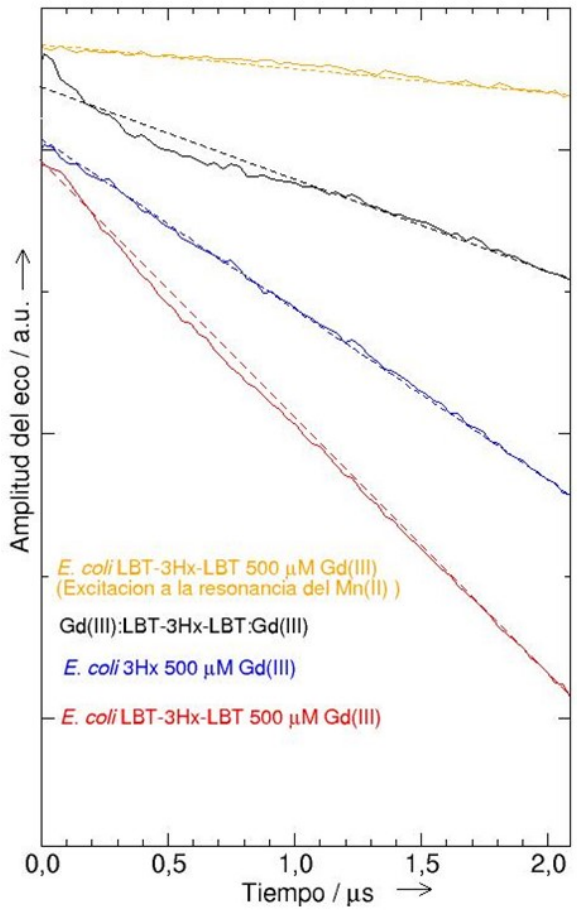


Figura 32: Señales de DEER para L-3Hx-L purificada (negro), en células de E. coli luego de 4h de inducción en medio suplementado con 500 uM de Gd(III) (rojo) y de células E coli luego de 4h de inducción de 3Hx en medio suplementado con 500 uM Gd(III) (azul). El control pulsando a la frecuencia de resonancia de Mn(II) se muestra en naranja. Las líneas discontinuas corresponden a las líneas de base calculadas correspondientes a interacciones intermoleculares.

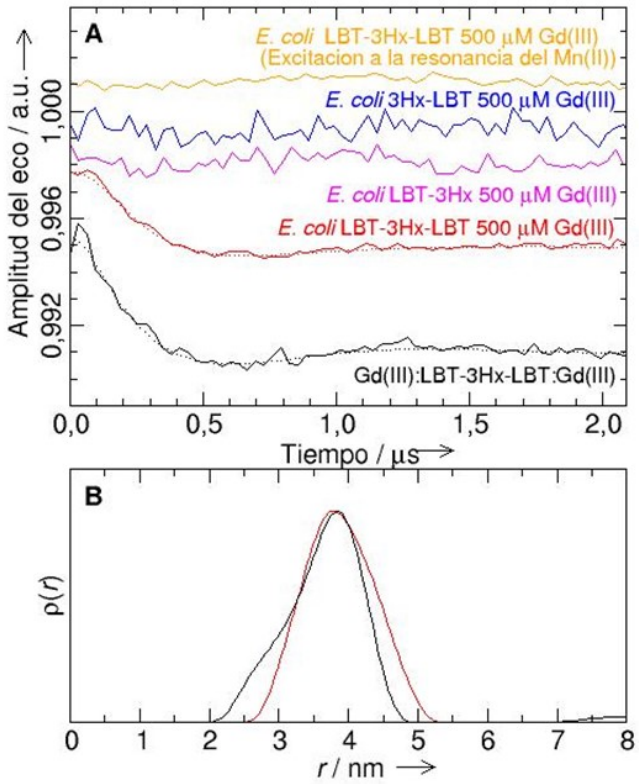


Figura 33: Medidas de DEER a 94 GHz en células de E. coli luego de 4 horas de inducción en medios suplementados con Gd(III) A) Medidas de DEER corregidas (líneas continuas) y ajustes (líneas de puntos). En cada curva se especifica la construcción y condiciones ensayadas. B) Distribuciones de distancias Gd(III)-Gd(III) obtenidas por análisis de Tikhonov.

4.3.1.2.1. Análisis de sobrevida y producción proteica en cepas de *E. coli* cultivadas con Gd(III)

La presencia de alta concentración de Gd(III) en el cultivo es potencialmente tóxica. Para asegurar que la presencia del metal no estuviese afectando a los cultivos, se llevaron a cabo ensayos de sobrevida y de producción proteica en las mismas condiciones del experimento.

El análisis de la tasa de sobrevida mostró que, en ausencia de Gd(III), los cultivos al final de la incubación presentan $(3,2 \pm 1,1)x10^9$ ufc/ml, mientras que en presencia del metal tienen $(3,6 \pm 0,5)x10^9$ ufc/ml. Estos resultados indican que concentraciones de Gd(III) en el medio de cultivo de hasta 500 uM no afectan la tasa de sobrevivencia de *E. coli*.

La producción proteica en presencia del metal en el medio fue estudiada comparando, por electroforesis en gel de poliacrilamida, las intensidades de las bandas de proteína obtenidas para distintos cultivos (Figura 34). En base a los resultados obtenidos se pudo concluir que las concentraciones de metal utilizadas no afectan la expresión proteica.

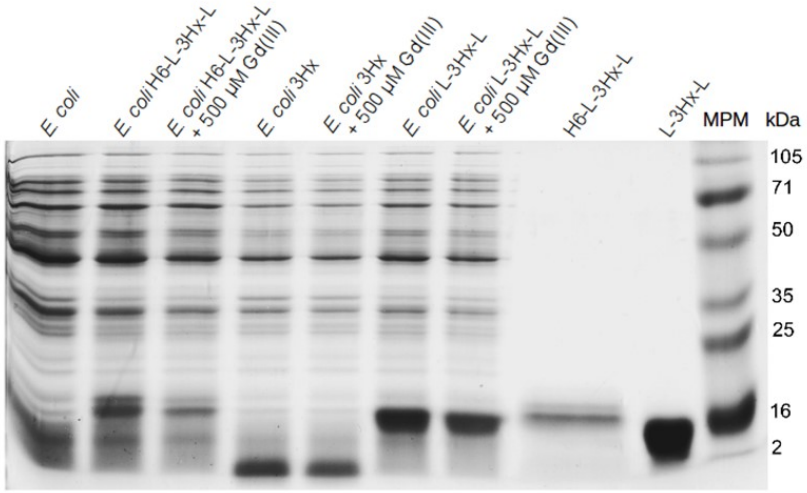


Figura 34: Expresión y purificación de las diferentes construcciones. Extractos solubles totales de células de E. coli luego de 4h de inducción con IPTG a 25 °C sin plásmido (línea 1) o que contienen el plásmido para la expresión de H6-L-3Hx-L (líneas 2 y 3, peso molecular esperado = 14,1 KDa), 3Hx (líneas 4 y 5, peso molecular esperado 8,4 KDa) y L-3Hx-L (líneas 6 y 7, peso molecular esperado 11,9 Kda) La expresión proteica fue realizada en ausencia (líneas 1,2,4 y 6) o presencia (líneas 3,5 y 7) de 500 uM Gd(III). En las líneas 8 y 9 fue sembrada H6-L-3Hx-L purificada antes y después de la digestión con proteasa TEV (peso molecular esperado para la forma digerida 12,1 KDa).

4.3.1.2.2. Concentraciones intracelulares de Gd(III) y L-3Hx-L

Para conocer mejor el sistema PELDOR y confirmar la utilidad del mismo como metodología para realizar medidas de distancia, resulta interesante conocer la concentración tanto de metal como de proteína en el interior celular.

La intensidad de una señal en un espectro de EPR se correlaciona en forma directa con la concentración de metal. Espectros de muestras de proteína purificada fueron utilizados para estimar la concentración de metal dentro de la muestra de *E. coli* incubado con 500 uM Gd(III) por 4 h. La concentración de Gd(III) dentro del tubo fue estimada en 800 uM. Con un procedimiento análogo se estimó la concentración de Mn(II) en la misma muestra en 30 uM. Considerando que dicha concentración se corresponde con una concentración intracelular establecida previamente en 120 uM, se puede considerar un factor de 4 entre la concentración nominal de metal en el tubo y la intracelular real. Usando ese factor de diferencia, la concentración intracelular de Gd(III) fue estimada en 3 mM. Para confirmar el valor obtenido, se resuspendió y sonicó en agua un pellet celular preparado a partir de un cultivo en las mismas condiciones. La concentración de metal en el lisado fue cuantificada por ICP-MS (Espectroscopía de Masas con Plasma Acoplado Inductivamente) en las instalaciones de ISIDSA (Facultad de Ciencias Químicas, Universidad Nacional de

Córdoba). A partir del número conocido de células viables por ml de cultivo y, asumiendo un volumen de 4 fl por célula [90], la concentración intracelular de Gd(III) fue estimada en 6 mM.

La concentración intracelular de proteína fue estimada a partir de la cuantificación de H6-L-3Hx-L purificada a partir de cultivo y de los niveles de expresión relativa de la misma con respecto al cultivo de L-3Hx-L. Este procedimiento indirecto fue necesario dado que, al carecer de la cola de histidina, no se contaba con un método simple de purificación para esas construcciones. A partir de un litro de cultivo celular que contiene $(9,8 \pm 0,5)x10^8$ cfu/ml se pudieron obtener aproximadamente 10 mg de H6-L-3Hx-L. Considerando un volumen celular de 4fl, la cantidad producida corresponde a una concentración intracelular de proteína de 0,2 mM. La cuantificación del gel realizada por el Dr. Tabares mostró que los niveles de expresión de L-3Hx-L fueron 5 a 10 veces mayores que para H6-L-3Hx-L. Por lo tanto, la concentración intracelular de L-3Hx-L en los cultivos inducidos por 4h en presencia de 500 uM de Gd(III) se estima en 1-2 mM.

En base a las medidas realizadas, se puede decir que con una concentración intracelular de Gd(III) estimada en 6 mM y de proteína estimada en 1-2 mM, las concentraciones se encuentran en el mismo orden de magnitud y son apropiadas para la aplicación de la técnica. Además, la relación entre ambas cantidades se aproxima a la relación óptima de 1:2 de proteína con respecto al metal.

4.3.2. Conclusión de la puesta a punto

Las medidas de PELDOR *in cell* sobre la construcción 3Hx utilizando etiquetas de LBT fueron exitosas. El trabajo con etiquetas de LBTs fue efectivo y, tanto las secuencias de pulsos como la incorporación de metal y el manejo de las muestras, fue optimizado. Fue posible, además, estimar las concentraciones necesarias de proteína y metal para llevar a cabo medidas *in cell*. La influencia del metal sobre la viabilidad y producción proteica fue evaluada en las condiciones de trabajo, confirmando la utilidad de la técnica. Este trabajo fue el primero en realizar medidas PELDOR con etiquetas autoensamblables codificadas genéticamente para medidas estructurales no disruptivas.

Sin embargo, la modulación obtenida fue menor que la que se suele obtener para proteínas purificadas. Muchos factores del sistema pueden estar disminuyendo la señal, incluyendo la presencia de Gd(III) libre y la posibilidad de que el metal se una a otras estructuras del interior celular. Por todo esto, se considera poco viable el estudio *in cell* en sistemas menos rígidos, en donde las distribuciones de distancias son más difíciles de ajustar.

4.3.3. Experimentos PELDOR sobre construcciones de D1-D2

Para investigar la dinámica de los dominios dsRBD de HYL1 se propuso estudiar las diferencias en la amplitud conformacional entre la proteína libre y unida a su sustrato. Para ello se llevaron a cabo estudios de PELDOR *in vitro* sobre las construcciones de D1D2 que contienen etiquetas de LBTs, tanto en ausencia como en presencia del precursor miR172a. Los LBTs fueron posicionados en los extremos de la secuencia de la proteína y al comienzo del *linker* que une los dominios. Se trabajó con tres construcciones (Figura 35) que corresponden a todas las combinaciones de dos LBTs en esas posiciones:

- L-D1-L-D2: presenta las etiquetas a ambos lados del primer dominio dsRBD, por lo que la distancia esperada se aproxima al largo del dominio y ronda los 4-4,5 nm.
- D1-L-D2-L: esta construcción mide el *linker* y el largo del segundo dominio, por lo que la distribución de distancia esperada es mayor que para la construcción anterior.
- L-D1-D2-L: contiene los LBTs en los extremos de la estructura. La distribución de distancia para esta última debería estar fuertemente influenciada por la flexibilidad del *linker*.

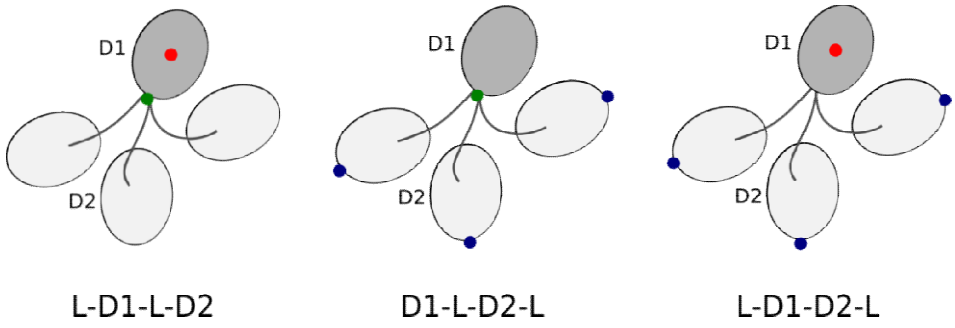


Figura 35: Construcciones para los estudios de PELDOR.

Los resultados obtenidos a partir de los estudios de PELDOR *in vitro* se muestran en la Figura 36. La modulación obtenida para L-D1-L-D2 libre permitió calcular una distribución de distancia centrada en aproximadamente 4 nm. Como era de esperar, la presencia del precursor de miARN en el medio no produjo cambios significativos. Sin embargo, para el resto de las construcciones no fue posible calcular distribuciones de distancias en forma robusta. Esto es debido a que sus curvas no presentaron modulaciones apreciables, por lo que pueden ser ajustadas a múltiples y diversas distribuciones de distancias dependiendo de la metodología utilizada para procesar los datos. La baja modulación puede reflejar que el sistema es muy dinámico, o que las distancias exceden lo medible por esta técnica.

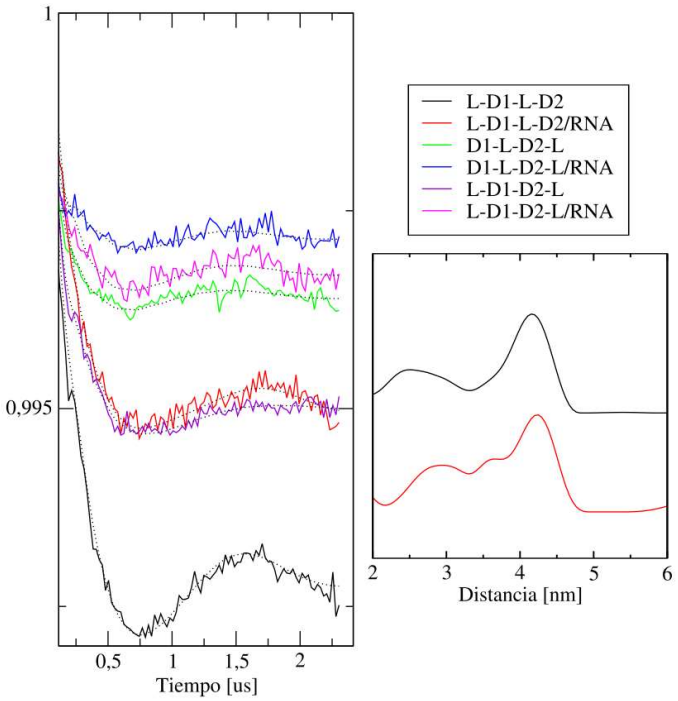


Figura 36: experimento PELDOR sobre las construcciones de D1-D2 con dos LBTs. A) curvas PELDOR obtenidas luego de corregir la línea de base. B) distribuciones de distancia correspondientes en escala de nm. El código de colores utilizado se muestra en el recuadro.

4.3.4. Conclusiones de los estudios PELDOR

Se llevaron a cabo experimentos de PELDOR para estudiar la distribución de distancia explorada por los dos dominios dsRBD de HYL1 tanto libre como en complejo con un precursor modelo. Se trabajó con tres construcciones con dos etiquetas de LBT cada una. Los resultados obtenidos solo permitieron estimar distancias para la construcción en la que las etiquetas flanquean el primer dominio. En ese caso, la distancia obtenida se corresponde con lo esperado, y no sufre modificaciones por la presencia del precursor. Para el resto de las construcciones la dificultad de cuantificar distancias puede deberse a un alto dinamismo del sistema o a la presencia de distancias largas. La calidad de los datos obtenidos no permite modelar con precisión la posible distribución de distancias y se requiere complementar el resultado aplicando otras técnicas.

4.4. ESTUDIOS DE RELAJACIÓN PARAMAGNÉTICA POR RMN

Para obtener una descripción más detallada del conjunto de conformaciones explorado por los dominios de HYL1 unidos por su *linker* se recurrió a la espectroscopía de RMN. Las señales de RMN son muy sensibles a la presencia de electrones desapareados, permitiendo evaluar la distancia entre el centro paramagnético y los núcleos detectados en base a la alteración de la velocidad de relajación de los núcleos. A su vez, los espectros de correlación en dos dimensiones $^{1}H^{15}N$-HSQC presentan una señal por cada grupo amida de la proteína, brindando un mapa directo del comportamiento del sistema con detalle a nivel de residuo. De esta forma los estudios de RMN permitieron obtener información estructural precisa para modelar luego las conformaciones exploradas por la proteína.

4.4.1. Construcciones utilizadas y espectros

Los estudios por RMN fueron realizados sobre las construcciones de D1-D2 con un LBT en los extremos N- o C- terminal. Los espectros de referencia se adquirieron sobre muestras con el LBT cargado con Lu(III), un ión lantánido diamagnético (configuración f^{14}, capa llena). Para evaluar la interacción de la proteína con la etiqueta LBT se cargó la proteína con los iones lantánidos paramagnéticos Dy(III) o Gd(III). El Gd(III) tiene una configuración electrónica f^{7}, simétrica, por lo que solamente se espera obtener un efecto de aumento de la relajación (PRE). Por su parte el Dy(III) de configuración f^{9} presenta una elevada anisotropía de susceptibilidad magnética y sus complejos producen fuertes efectos anisotrópicos sobre los núcleos cercanos. En particular, se espera obtener desplazamientos de pseudocontacto (PCS) fuertes y una tendencia al alineamiento del sistema en el campo magnético, dando lugar a acoplamientos residuales dipolares (RDC). Sin embargo, los espectros adquiridos sobre las muestras cargadas con Dy(III) fueron de baja calidad y no fue posible detectar efectos PCS o RDC apreciables, por lo que se decidió no continuar trabajando con dicho metal. La dificultad en obtener efectos definidos puede estar relacionada con la flexibilidad conformacional de la etiqueta respecto de la proteína, lo que resultaría en el promediado de los efectos anisotrópicos esperados.

Los espectros $^{1}H^{15}N$-HSQC adquiridos sobre las distintas construcciones cargadas con Lu(III) o Gd(III) se muestran en la Figura 37. Todas las medidas fueron realizadas en las condiciones óptimas de estabilidad determinadas previamente: solución HEPES 20 mM, NaCl 500 mM, DTT 10 mM, pH 7.0. Para L-D1-D2 fue posible asignar 96 de 176 resonancias esperadas de HN, mientras que para D1-D2-L fue posible asignar 117 resonancias de HN de 178.

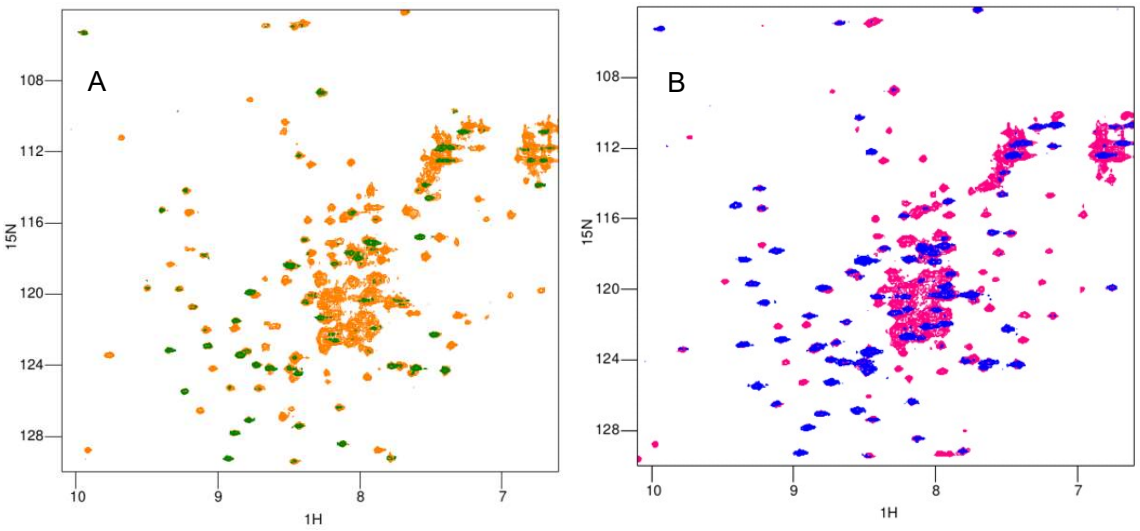


Figura 37: Espectros PRE 1H15N-HSQC sobres construcciones de D1-D2 con un LBT. A) En naranja L-D1-D2 con Lu(III) y en verde L-D1-D2 con Gd(III). B) En rosa D1-D2-L con Lu(III) y en azul D1-D2-L con Gd(III). La concentración de las muestras de L-D1-D2 fue de 126 uM y de D1-D2-L de 210 uM.

La evaluación sitio específica de la relajación paramagnética inducida por el metal se realizó midiendo la intensidad de las señales en espectros $^{1}H^{15}N$-HSQC. La relación de intensidad entre las señales en la condición paramagnética con respecto a la diamagnética es proporcional a la relación de velocidad de relajación transversal (T2) de las formas diamagnéticas y paramagnéticas. Ésta, a su vez, puede correlacionarse con la distancia entre el metal y cada núcleo dentro de la proteína. Los resultados obtenidos se muestran en la Figura 38. En cada caso se graficó la relación de intensidades de la muestra cargada con un equivalente de metal paramagnético Gd(III) (I) con respecto a la muestra con un equivalente de metal diamagnético Lu(III) (I_0) para cada residuo.

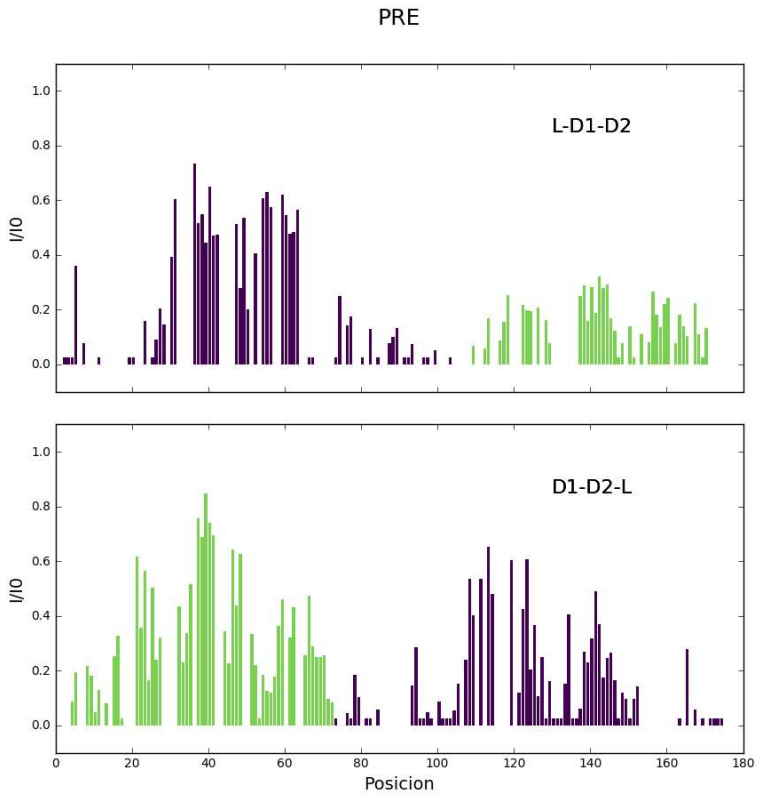


Figura 38: Resultados de PRE sobre L-D1-D2 (arriba) y D1-D2-L (abajo). En la gráfica se representan los valores de I/I_0 para cada residuo, considerando I_0 como la intensidad de las señales en ausencia de Gd(III). Los dominios más informativos se muestran en verde (D2 de L-D1-D2 y D1 de D1-D2-L).

La presencia de Gd(III) en el extremo C-terminal de la proteína dio un claro patrón PRE en D2, consistente con su estructura. Varias señales en D1 muestran también efecto PRE de menor magnitud, lo que indica la presencia de interacciones entre el extremo C-terminal y el N-terminal de corta duración. La incorporación de Gd(III) en el extremo N-terminal dio un resultado no esperado. Mientras que el patrón en D1 es consistente con la estructura, se observó un efecto PRE global mayor para el dominio distante D2. Es decir, la incorporación de Gd(III) en L-D1-D2 y en D1-D2-L resulta en un efecto inesperadamente asimétrico.

Las modificaciones realizadas sobre la proteína durante la puesta a punto del sistema mostraron que el segundo dominio es particularmente sensible y su estructura marginalmente estable. Para descartar algún problema de estabilidad estructural por parte del segundo dominio, el metal fue retirado del sitio LBT mediante el agregado de cinco equivalentes de quelante EDTA. Los espectros adquiridos, luego de quelar el metal, mostraron que las señales habían recuperado casi la totalidad de su intensidad inicial en forma homogénea. Se puede concluir, entonces, que los dominios mantuvieron su

estructura estable durante el experimento y que la disminución en la intensidad de las señales se corresponde con un efecto PRE y no resulta de artefactos de la muestra.

4.4.2. Experimentos de control

Una posible explicación para la asimetría de los PRE sobre los dominios es que exista una unión débil y directa de Gd(III) a D2. Para comprobar si existe unión inespecífica del metal sobre D1D2 una muestra de dicha construcción fue titulada con Gd(III) (Figura 39).

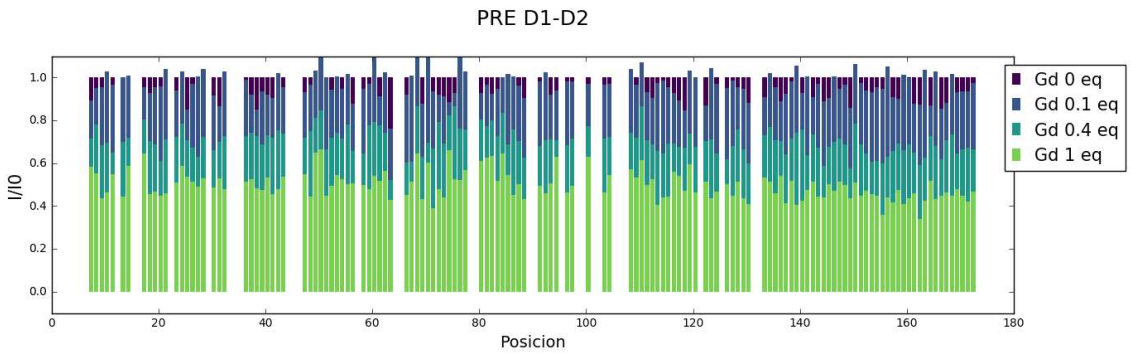


Figura 39: Control de PRE con D1D2. En la gráfica se representan los valores de I/I_0 para cada residuo, considerando I_0 como la intensidad de las señales en ausencia de Gd(III).

La presencia de cantidades crecientes de metal produjo un ensanchamiento de las señales en forma uniforme a lo largo de la secuencia de la proteína. La ausencia de efecto PRE localizado sugiere que *D1-D2* no une Gd(III) en forma inespecífica en ninguna parte de su estructura. Por lo tanto, el efecto PRE observado sobre las construcciones con etiqueta LBT proviene de la unión específica del metal sobre cada etiqueta dentro de la estructura completa. En presencia de 1 equivalente de Gd se observa un efecto PRE global de *ca.* 50%, mientras que en las muestras con LBT se observan señales que retienen hasta un 80% de su intensidad, confirmando la ausencia de Gd(III) libre en las muestras con LBT.

Una explicación para el resultado asimétrico observado es la posibilidad de que HYL1 dimerice a través de D2, tal y como se ha reportado en estudios previos[89]. Si la proteína forma dímeros como los esquematizados en la Figura 40, el dominio D2 de un monómero quedaría posicionado cerca del LBT del otro monómero del dímero imprimiendo efecto PRE sobre este último. Para evaluar esta posibilidad se realizaron experimentos sobre muestras de ^{15}N D1-D2 en mezclas 1:1 con ^{14}N L[Gd(III)]-D1-D2 o ^{14}N D1-D2-

L[Gd(III)]. A partir de dichas combinaciones se puede plantear la formación de distintos dímeros (Figura 40). Entre estos solamente los que contengan una forma marcada unida a otra sin marcar dan efecto PRE. Los resultados se muestran en la Figura 41.

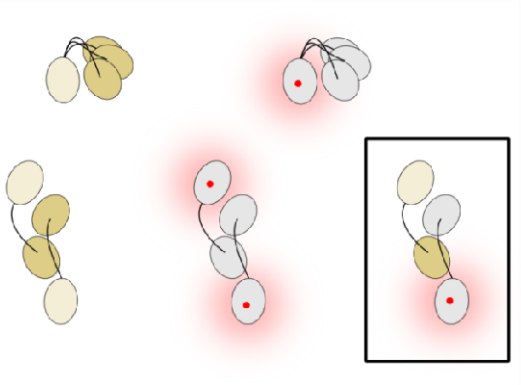


Figura 40: Situaciones posibles en una muestra de proteína 15N D1-D2 (marrón) con un equivalente de proteína 14N L-D1-D2 o 14N D1-D2-L cargado con Gd(III) (gris). El punto rojo representa el metal. El recuadro indica la única estructura con efecto PRE teórico apreciable.

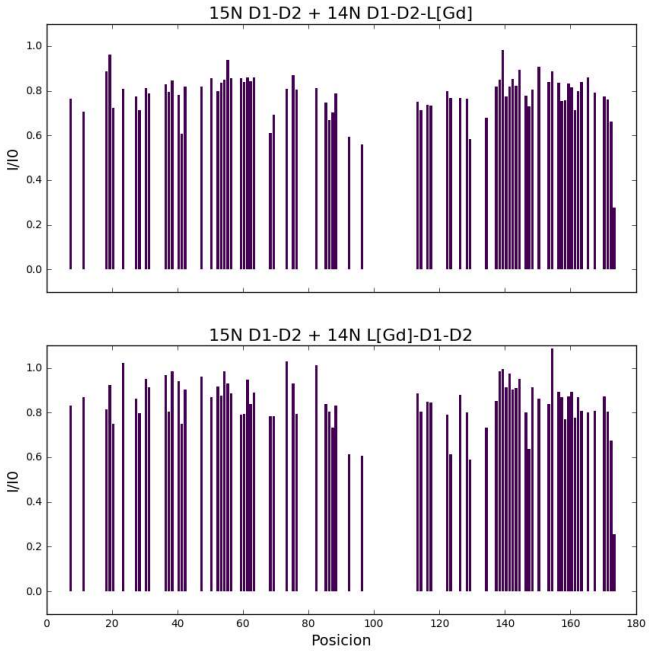


Figura 41: Control de dimerización. 15N D1-D2 con D1-D2-L sin marca (arriba) y 15N D1-D2 con L-D1-D2 sin marca (abajo). En la gráfica se representan los valores de I/I0 para cada residuo, considerando I0 como la intensidad de las señales en ausencia de Gd(III).

Los valores de I/I0 obtenidos fueron nuevamente uniformes a lo largo de la secuencia de ambas proteínas, por lo que no se aprecia efecto PRE selectivo. Esto indica

que, en las condiciones del experimento, no se produce dimerización. Alternativamente, si se produjera la misma puede no manifestarse debido a la localización de los LBTs con respecto a la interfaz de dimerización. Por lo tanto, los perfiles asimétricos de PRE observados en L-D1-D2 y D1-D2-L son debidos a una asimetría en el movimiento entre dominios.

4.4.3. Conclusiones de Estudios de relajación paramagnética

Para estudiar el rol que cumple el *linker* dentro de la estructura de HYL1 se llevaron a cabo experimentos de PRE. Se trabajó sobre dos construcciones de D1D2, que contienen en cada extremo una etiqueta LBT cargada con metal paramagnético. El metal afecta la intensidad de las señales de RMN de los núcleos en función de la distancia. El análisis visual de los resultados obtenidos sobre la muestra con la etiqueta en el extremo C-terminal se condice con lo esperado. Sin embargo, para la construcción que contiene una etiqueta en el extremo N-terminal, el dominio D2 se mostró más afectado que lo esperado. Este resultado asimétrico indicaría que los dos dominios dsRBDs de HYL1 tienen restricciones en términos de orientación relativa y distancia. Para hacer un análisis más exhaustivo de los resultados, se plantea utilizarlos para seleccionar *in silico* conformaciones probables dentro de un ensamble de conformaciones posibles.

4.5. CONSTRUCCIÓN *in silico* DE CONFORMACIONES POSIBLES Y SELECCIÓN DE ESTRUCTURAS

Los resultados de PRE sugieren que el muestreo de espacio conformacional de los dominios de HYL1 no es homogéneo. Se observó una asimetría interdominio que no tiene una explicación directa si consideramos al sistema como dos cuerpos rígidos unidos por un hilo flexible. Para la interpretación de los resultados de PRE se decidió generar, *in silico,* un ensamble formado por un gran número de estructuras posibles, para luego seleccionar aquellas que expliquen los resultados experimentales y permitan inferir la dinámica que existe entre ambos dominios.

El primer paso para la generación del ensamble es generar un modelo base. Considerando la asimetría en los valores PRE interdominio obtenidos, el modelo que se genere debe poder explicar simultáneamente los resultados sobre las muestras L-D1-D2 y D1-D2-L. Para poder trabajar con ambas construcciones en forma simultánea, las estructuras simuladas corresponden a la construcción L-D1-D2-L. El modelo de dicha construcción fue generado con el programa *Modeller* utilizando las estructuras cristalográficas disponibles. Una vez construido el modelo se procedió a simular con el programa *Rosetta* un ensamble formado por un gran número de conformaciones posibles. Luego, todas las medidas de distancias entre cada residuo y los metales fueron calculadas y traducidas a valores de cociente de velocidad de relajación. Las estructuras fueron separadas en grupos en base a sus perfiles PRE predichos. Los perfiles simulados de las estructuras contenidas dentro de cada grupo fueron comparados con los datos experimentales para seleccionar aquellos grupos con mayor correlación. Las estructuras seleccionadas fueron graficadas para interpretar los resultados.

4.5.1. Modelado de estructuras iniciales en *Modeller*

Se modelaron 100 posibles estructuras de *L-D1-D2-L* con sus etiquetas LBTs cargadas con metal con el programa *Modeller*, utilizando como moldes las estructuras PDB 3ADG (para D1), PDB 3ADJ (para D2) [91] y PDB 1TJB (LBT cargado con metal) [92].En base a los valores del parámetro DOPE para cada estructura, que dan indicio de la calidad de los modelos, se seleccionó aquella con menor valor de DOPE (-15774.713) para continuar con el trabajo.

4.5.2. Generación de ensamble con *Rosetta*

A partir del modelo seleccionado de L-D1-D2-L fueron simuladas 50.000 conformaciones con el programa *Rosetta*. Para la generación de las distintas conformaciones se aleatorizaron los valores de los ángulos Phi/Psi de los residuos que

corresponden al *linker* entre dominios (residuos 90-104) y aquellos de las regiones entre la proteína y el LBT (residuos 17, 18, 175 y 176), mientras que los dsRBDs y el LBT fueron considerados cuerpos rígidos. Las distintas regiones dentro de la secuencia se muestran en la Figura 42.

GYIDTNNDGWIEGDELHMVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKSLFQSTVILDGVRYNSLPGFFNRKAAEQSAAE
VALRELAKSSELSQCVSQPVHETGLCKNLLQEYAQKMNYAIPLYQCQKVETLGRVTQFTCTVEIGGIKYTGAATRTKKDA
EISAGRTALLAIQSVDYIDTNNDGWIEGDELVD*

Figura 42: Secuencia de L-D1-D2-L. Los LBTs, D1 y D2 (en rosa, amarillo y verde respectivamente) fueron consideradas como cuerpos rígidos. En azul se indican los residuos para los cuales se permitió el movimiento.

El protocolo para la aleatorización fue programado utilizando Py*Rosetta* como se detalla a continuación. Un residuo de los definidos como móviles es seleccionado al azar. Se seleccionan aleatoriamente para ese residuo valores de sus ángulos Phi y Psi a partir de una biblioteca de valores *random coil* provista por la herramienta Flexible Mecano . La energía de la nueva estructura es evaluada. Los movimientos son aceptados o rechazados dependiendo del cambio de energía que producen, utilizando el criterio de Metrópolis: se aceptan todos los movimientos que reducen la energía, mientras que aquellos que la incrementan son aceptados con una probabilidad $e^{(\Delta E/kT)}$. Esto genera un muestreo de distribución de Boltzmann permitiendo superar posibles barreras de energía en la búsqueda conformacional. Si luego de 100 intentos no se alcanza un valor aceptable se conservan los valores originales de Phi y Psi para ese residuo en esa estructura. Mediante este análisis se evita la generación de estructuras en donde existan choques entre distintas partes de la proteína. El protocolo se repite hasta aleatorizar los ángulos de todos los residuos seleccionados. Usando este procedimiento se generaron 50.000 estructuras para L-D1-D2-L. El total de estructuras se separó en 5 grupos de 10.000 estructuras para simplificar el procesamiento posterior, las cuales se indican a lo largo del trabajo con números romanos.

4.5.3. Análisis gráfico del ensamble

La evaluación visual de los ensambles generados no es práctica, y el número de variables asociado a cada conformación (18 pares de ángulos phi/psi) es muy alto. El objetivo del protocolo es producir un conjunto de estructuras que explore todo el espacio conformacional de los dsRBDs de HYL1 permitido por el *linker*. El *linker* entre los dominios limita esencialmente las posibles orientaciones relativas entre ellos. Para evaluar si el protocolo diseñado cubre en forma aceptable el espacio conformacional disponible se optó por determinar la orientación relativa entre los dos dominios. Sobre cada confórmero

generado se calculó el centro de masa de cada dominio (CDM1 para D1 y CDM2 para D2), el vector que conecta CDM1 con su correspondiente CDM2 (vCDM) y el tensor del momento de inercia de D1 (Figura 43). Este último tiene una orientación fija respecto de la estructura del dominio, de manera que sirve como marco de referencia para las orientaciones relativas. Los vectores de este tensor fueron definidos como Z, Y y X en orden decreciente de magnitud. Se calcularon las proyecciones de vCDM sobre este nuevo marco de referencia. Esto permitió definir la conformación relativa de los dominios en base a los dos ángulos que se muestran en la Figura 43. El primer ángulo fue denominado theta (θ) y corresponde a la proyección de vCDM sobre el autovector Z, mientras que un segundo ángulo denominado phi (ψ) corresponde al ángulo que se forma entre la proyección de vCDM con el autovector X en el plano formado por el autovector X y el autovector Y. Finalmente se graficaron los valores de los ángulos calculados para cada conformación en el ensamble.

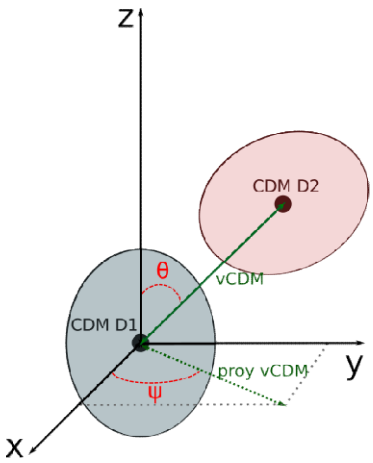


Figura 43: Vectores y ángulos para el análisis de la cobertura del espacio conformacional. D1 se encuentra esquematizado en gris y D2 en rosa. Los vectores se muestran como líneas rectas en verde, y los ángulos como líneas curvas y en rojo.

La Figura 44 muestra los valores obtenidos para los 5 grupos de estructuras. El análisis del histograma obtenido muestra que el protocolo utilizado genera una buena cobertura del espacio conformacional disponible para la proteína. Se pueden identificar zonas pobladas y zonas prohibidas. Mientras que las conformaciones exploran todo el rango de ángulos phi, los ángulos theta están limitados por la restricción impuesta por el *linker*, y solo admite valores mayores a 50 grados, e incluso más limitados cuando Phi es cercano a cero grados.

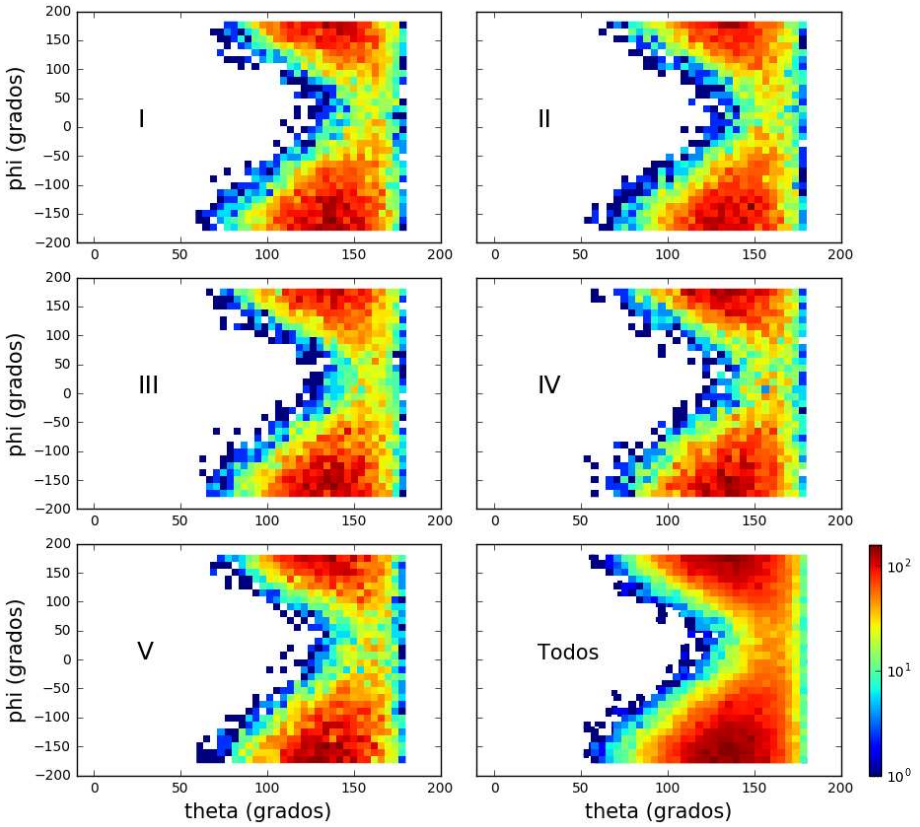


Figura 44: Representación gráfica del espacio conformacional cubierto por el ensamble generado de L-D1-D2-L. Cada gráfico corresponde a uno de los cinco grupos, y abajo a la derecha se muestra el histograma para todos juntos. A la derecha se encuentra el código de colores para la interpretación del histograma.

4.5.4. Predicción de patrones PRE para el ensamble

Una vez generadas las conformaciones físicamente posibles para los dominios dsRBD de HYL1 se calcularon los valores de PRE esperados de manera de compararlos con los datos experimentales. Se midió la distancia a ambos metales para cada grupo amida del esqueleto dentro de cada estructura. Los valores de velocidad de relajación paramagnética (r2p) para las distancias medidas fueron estimados usando las ecuaciones correspondientes [51]. Las velocidades de relajación dependen de varias constantes físicas y de parámetros del sistema. Entre estos últimos fue necesario fijar los siguientes parámetros:

- Los valores de $r_{2,0}$ para cada residuo fueron calculados a partir de medidas de relajación (Figura 45).
- tiempo de relajación electrónico (t_e): 2 ns, medido en el CEA Saclay a 298 K.
- tiempo de correlación rotacional (t_r): 5 ns, se tomó el tiempo de correlación rotacional promedio estimado por medidas de relajación sobre los dominios dsRBD aislados.

- tiempo de intercambio (t_m): 5 ns, estimado a partir de datos de dinámica molecular.

Los cocientes de intensidades I/I_0 predichos se calcularon con la fórmula $I/I_0 = r_{2,0}/(r_{2p}+r_{2,0})$. En la Figura 46 se muestra el efecto PRE teórico utilizando los parámetros descritos y un valor promedio de $r_{2,0}$ para distintos valores de distancia. Como se puede observar, el rango de efecto PRE coincide con el rango teórico.

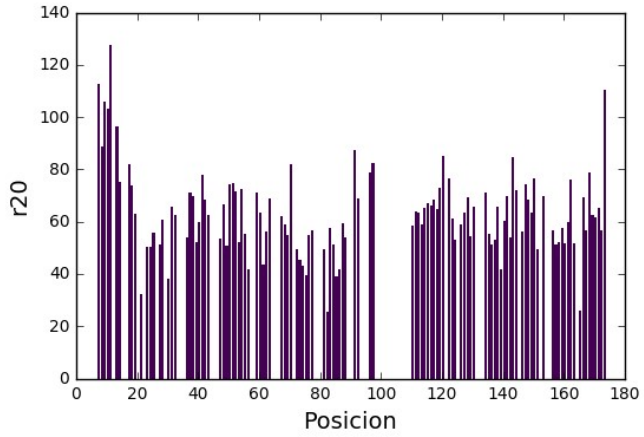


Figura 45: Valores r20 calculados a partir de experimentos de relajación.

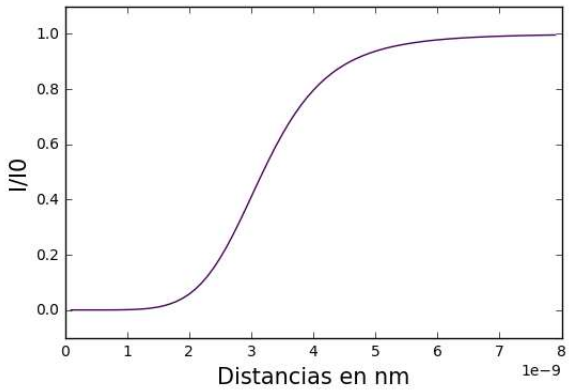


Figura 46: Valores de PRE teóricos calculados con $r_{2,0}$ promedio, t_e = 2 ns, t_r = 5 ns y t_m = 5 ns.

Como se explicó anteriormente, el ensamble conformacional que se seleccione debe responder simultáneamente a las restricciones obtenidas desde las dos etiquetas, la N- y la C-terminal. Para graficar los valores obtenidos considerando ambos metales en forma simultánea, a lo largo del resto de este trabajo las figuras muestran en la primer mitad los resultados correspondientes a la construcción L-D1-D2-L considerando el metal en el primer LBT, y en la segunda mitad aquellos correspondientes a la construcción L-D1-D2-L considerando al metal en el segundo LBT (esquema en la Figura 47). La numeración de los

gráficos sigue la de la construcción L-D1-D2-L, con los números sin modificar para los modelos con el metal en el LBT N-terminal y agregando "'" a los números de los modelos con el metal en el LBT C-terminal.

Figura 47: Esquema del procedimiento utilizado para graficar los resultados considerando ambos extremos de la proteína simultáneamente.

En la Figura 48 se muestran ejemplos de los valores obtenidos para 5 estructuras simuladas aleatorias.

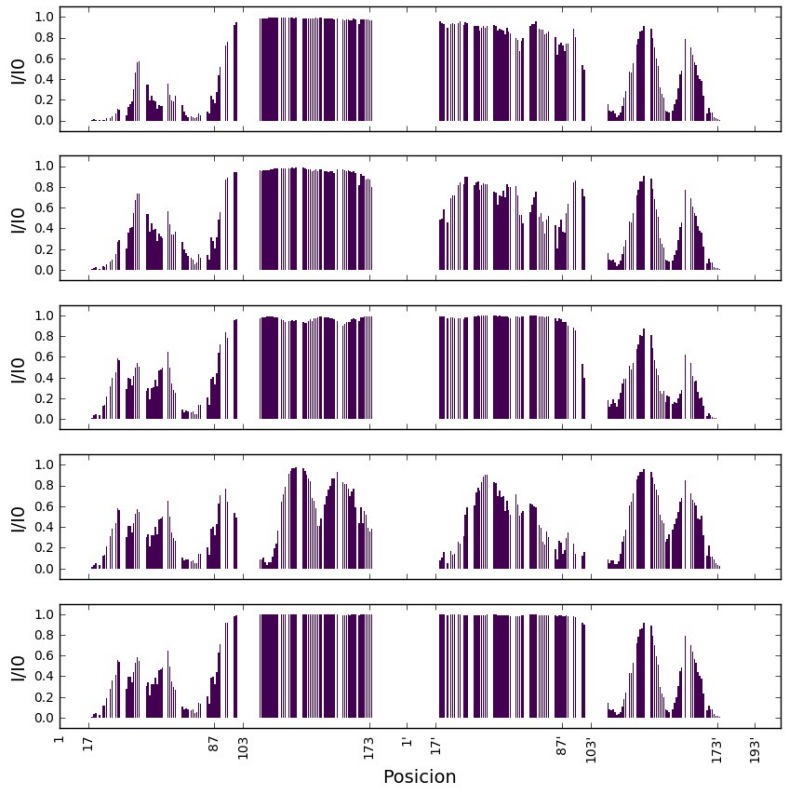


Figura 48: Ejemplos de perfiles I/I0 obtenidos para 5 estructuras simuladas aleatorias. La primera mitad corresponde a la presencia del metal en el LBT N-terminal, y la segunda parte al metal en el LBT C-terminal.

4.5.5. Primera reducción del ensamble

En el presente sistema dinámico, los tiempos de interconversión entre los distintos confórmeros son menores que los tiempos de vida de las magnetizaciones que manifiestan

los efectos PRE. Esto significa que, si existen múltiples conformaciones dentro de un sistema en movimiento, el resultado del PRE experimental no es un promedio directo de los patrones de las conformaciones que explora la proteína. Por esta razón, el cálculo de datos experimentales a partir de ensambles de estructuras no es trivial para este tipo de experimentos. Cuando una proteína experimenta múltiples conformaciones conocidas los resultados experimentales de PRE también pueden ser predichos en forma directa. Sin embargo, reconstruir un ensamble a partir de datos experimentales es un problema *ill-posed* (mal definido) [51]. El mayor número de grados de libertad del sistema con respecto al número de parámetros para fijarlo se traduce en que existen infinitas soluciones para el modelado. Existen diversas técnicas para resolver este problema [93–97]. El protocolo utilizado en este trabajo pretende reducir el número de estructuras del ensamble en forma gradual. En cada reducción son necesarios dos pasos. El primero consiste en agrupar estructuras en conjuntos según la similitud que presentan en sus valores de I/I_0 calculados. El segundo paso es el análisis de la similitud entre los perfiles de I/I_0 de las estructuras de cada grupo y los valores experimentales. Aquellas estructuras presentes en los grupos que presentan las mayores diferencias con respecto a los valores experimentales son eliminadas. Este procedimiento se repite tantas veces como se considere necesario. El grupo final de estructuras contendrá, por lo tanto, estructuras con perfiles de PRE similares tanto entre sí como con respecto al experimental.

Un primer análisis visual y global de los perfiles I/I_0 calculados muestra que existen dos regiones bien definidas. Los dominios directamente unidos a la etiqueta que se está considerando muestran una baja variabilidad en sus patrones de I/I_0 ya que responden únicamente a la estructura del dominio y a la posición de la etiqueta. En cambio, los dominios opuestos muestran perfiles muy variables. Estas dos regiones se pueden apreciar claramente en la Figura 49. Considerando que este trabajo pretende conocer la orientación relativa entre dominios, para el análisis de similitud de los perfiles I/I_0 se decidió analizar solamente los datos de las regiones que son informativas respecto a la relación estructural entre los dominios. Es decir, de los datos experimentales obtenidos sobre la construcción L-D1-D2 (modelados sobre la construcción L-D1-D2-L considerando el metal N-terminal) se utilizaron en la comparación únicamente los valores de I/I_0 obtenidos sobre D2. En cambio, de los datos experimentales obtenidos sobre la construcción D1-D2-L (modelados sobre la construcción L-D1-D2-L considerando el metal C-terminal) se utilizaron en la comparación los valores obtenidos sobre D1 (Figura 47).

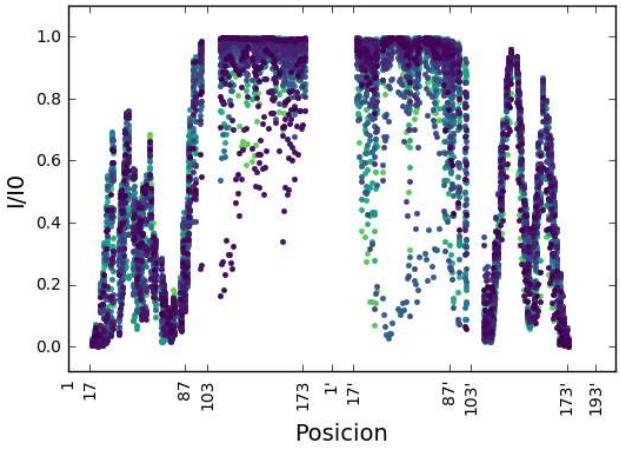


Figura 49: Ejemplos de 50 perfiles PRE predichos, en donde se observan regiones menos variables que corresponden a dominios unidos a la etiqueta (17-87 y 103'-173'), y regiones más variables que corresponde a los dominios restantes (103-173 y 17'-87').

Para el primer paso de agrupamiento, la diferencia entre dos modelos fue definida como la suma de las diferencias en los valores de I/I_0 para cada residuo de D1 (posiciones 18 a 88) considerando el Gd(IIII) C-terminal, y para D2 (posiciones 104 a 174) considerando el Gd(III) N-terminal. Por cada grupo de estructuras se generó una matriz con estas diferencias. Con la matriz obtenida se hizo un análisis de agrupamiento y se dividió cada grupo de datos en 3 grupos llamados 1, 2 y 3. El análisis gráfico de la distribución conformacional para estos grupos se muestra en la Figura 50.

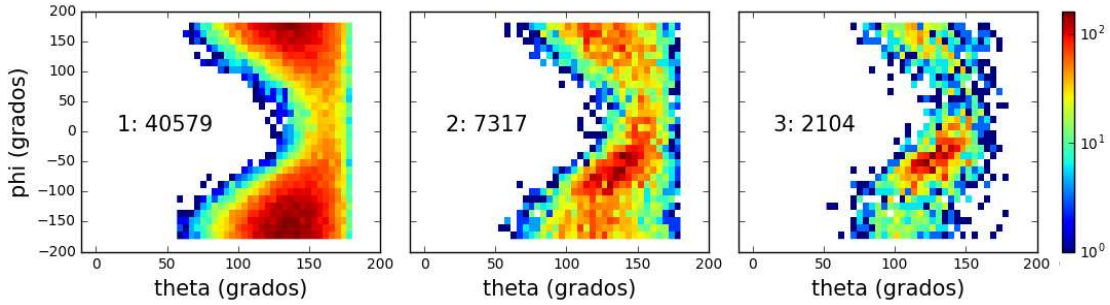


Figura 50: Distribución conformacional para el primer agrupamiento en los cinco grupos juntos. La leyenda dentro de cada gráfica indica el grupo y su número de estructuras. A la derecha se encuentra el código de colores para la interpretación del histograma.

El segundo paso para la reducción del número de estructuras del ensamble es determinar la similitud entre los grupos formados y los datos experimentales. En este caso, la diferencia entre cada grupo y los datos experimentales fue definida como la sumatoria de las diferencias al cuadrado entre los valores de I/I_0 para cada estructura simulada con

respecto a los valores experimentales. Los valores para los residuos de D2 (posiciones 104 a 174) de los modelos de L-D1-D2-L considerando el Gd(III) N-terminal fueron comparados con los de D2 de los valores experimentales de L-D1-D2. De forma análoga, los valores para los residuos de D1 (posiciones 18 a 88) de los modelos de L-D1-D2-L considerando el Gd(III) C-terminal fueron comparados con los de D1 de D1-D2-L experimental. Las diferencias calculadas representan la divergencia encontrada entre los valores experimentales de PRE trabajando con L-D-D2 y D1-D2-L, y los teóricos correspondientes a cada una de las estructuras simuladas dentro de cada grupo (Figura 51).

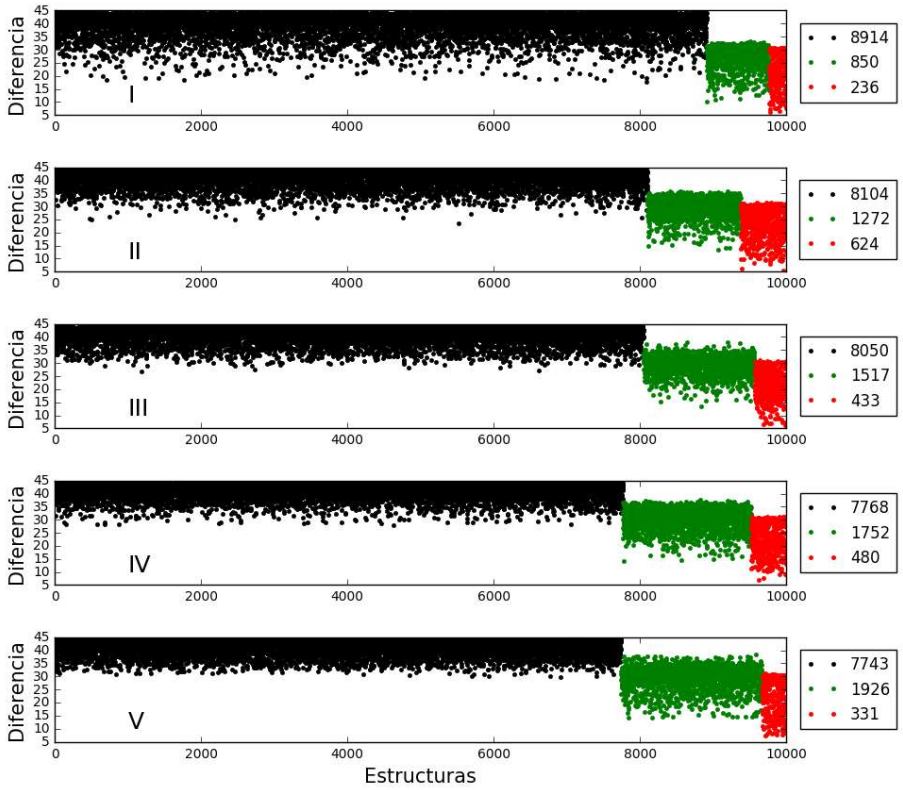


Figura 51: Divergencias de PRE entre cada estructura simulada y los resultados experimentales para cada grupo. El grupo 1 se muestra en negro, el 2 en verde y el 3 en rojo. La leyenda a la derecha de cada gráfica muestra el nº de estructuras de cada cluster.

El análisis visual permite observar que, en cada uno de los 5 grupos, aquel que presenta el mayor número de estructuras contiene a la vez los modelos que más difieren con respecto a los valores de PRE experimentales. Por lo tanto, las estructuras contenidas en dichos grupos fueron eliminadas, conservando aproximadamente el 20% de las estructuras restantes para el análisis posterior.

4.5.6. Segunda reducción del ensamble

La primera reducción del ensamble, con el protocolo descrito en el apartado anterior, redujo el número de estructuras considerablemente. Se realizó un paso posterior de refinado siguiendo el mismo protocolo. Los modelos correspondientes a los grupos remanentes luego del primer análisis (grupos 2 y 3) fueron reagrupados. El conjunto formado dio lugar a 10 nuevos grupos en base al análisis de similitud de sus perfiles I/I0 (grupos A a J). La distribución conformacional para los nuevos grupos se muestra en la Figura 52 y la correlación con los datos experimentales en la Figura 53.

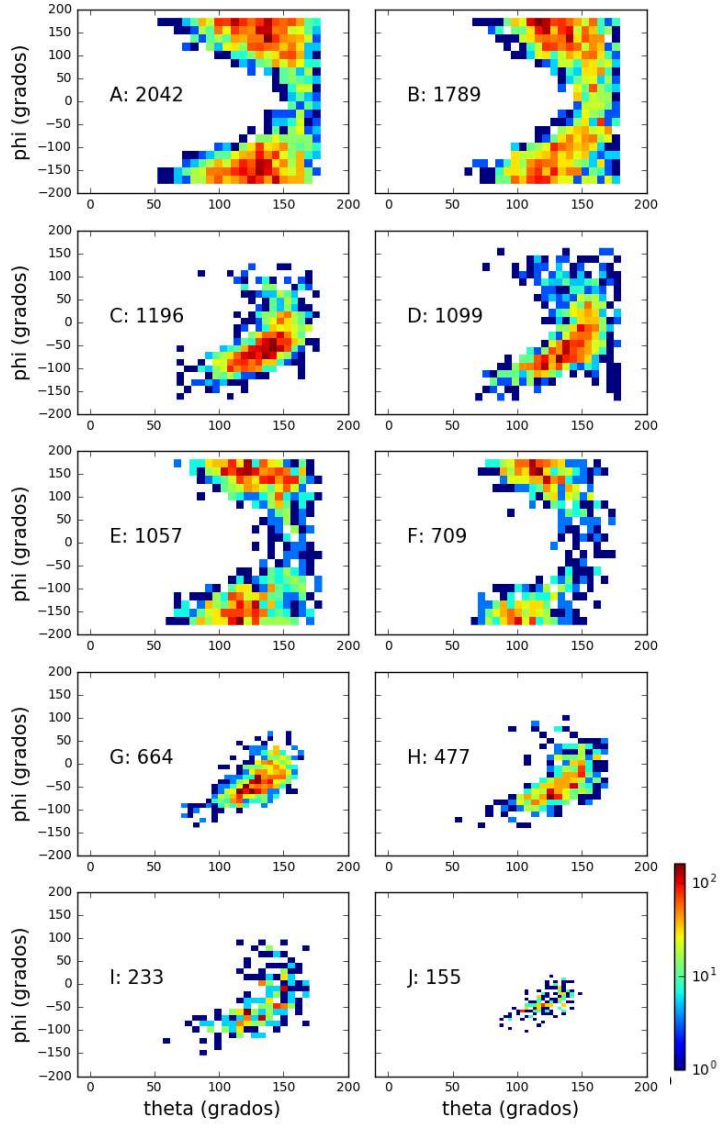


Figura 52: Representación gráfica en forma de histograma del espacio conformacional cubierto por cada grupo generado durante el segundo agrupamiento. La leyenda dentro de cada gráfica indica el grupo y su número de estructuras. A la derecha del último gráfico se encuentra el código de colores para la interpretación del histograma.

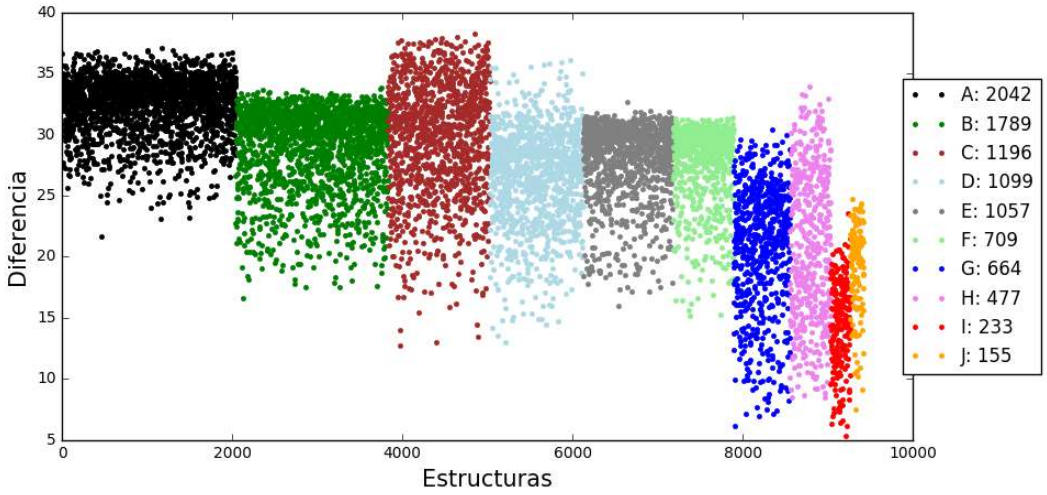


Figura 53: Distancias entre cada estructura simulada y los resultados experimentales para cada grupo del segundo agrupamiento. La leyenda a la derecha de cada gráfica muestra el nº de estructuras de cada cluster.

Se puedo observar que los *cluster* I y J, con 233 y 155 estructuras respectivamente, contienen aquellas que presentan los perfiles de PRE con las menores diferencias con respecto a los resultados experimentales. Los perfiles de PRE para cada *cluster* mostraron que el I ajustó muy bien para D2 cuando el metal está en el primer dominio, mientras que el *cluster* J ajustó bien para D1 cuando el metal estaba en el segundo dominio (Figura 54). El resto de los *clusters* no presentaron ajustes destacables.

Los valores obtenidos en las medidas de distancia entre los centros de masa y entre los metales para las estructuras de los *cluster* I y J se muestran en la Figura 55. Los resultados de distancias entre centros de masa mostraron que ambos dominios están cercanos entre sí, como era de esperar por los resultados experimentales de PRE. Es notable que los *cluster* I y J presentaron valores de distancias similares entre centros de masa, pero no entre metales. Una posible interpretación es que los dominios D2s estarían ubicados en la misma región del espacio, y que son los LBTs quienes determinan las diferencias entre los *clusters*. Las conformaciones con distancias más cortas entre metales (*cluster* I), son las que tienen al metal en el LBT C-terminal más cercano a D1, y eso se refleja en sus perfiles PRE (Figura 54).

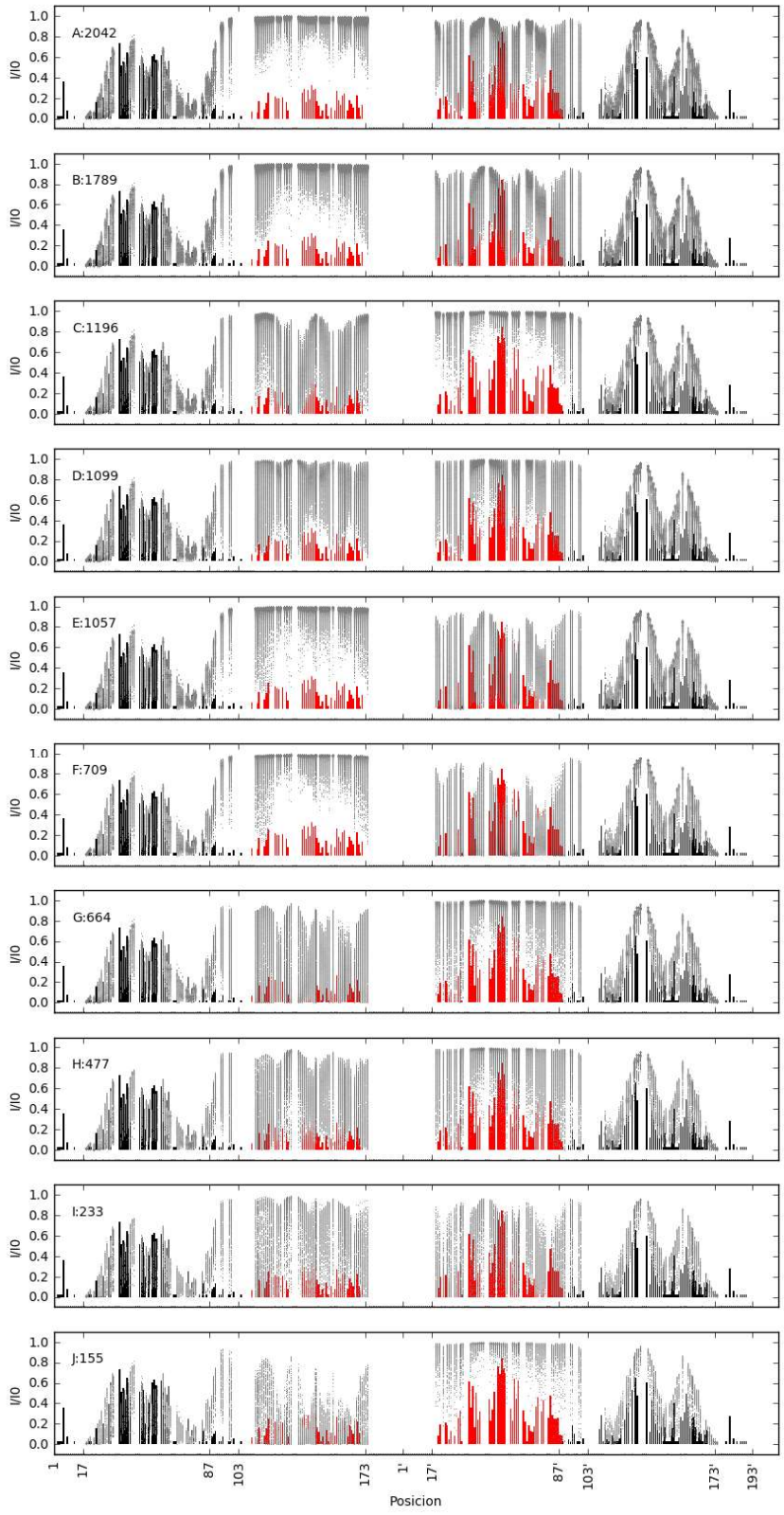


Figura 54: Perfiles PRE calculados para cada estructura dentro de cada cluster. En barras se muestran los resultados de los experimentos in vitro. En rojo se destacan las zonas que se tuvieron en cuenta para la formación de los clusters y la medición de las diferencias.

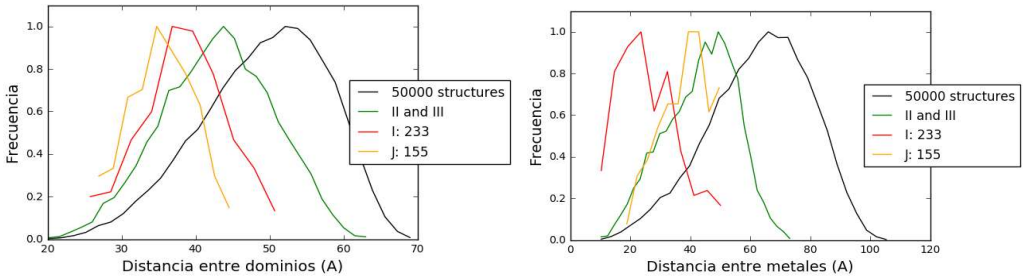


Figura 55: Distancia en Å entre los centros de masa (izquierda) y entre los metales (derecha) para las estructuras simuladas.

El análisis visual de las estructuras se hizo con el programa *Pymol* (Figura 56). Todas las estructuras fueron alineadas estructuralmente en D1. Como representación del centro de masa de D2 se eligió un residuo cercano al mismo (Tyr 149). Se analizaron tanto las posiciones de los centros de masa como de los metales. Se pudo observar que en todas las estructuras los dominios están cercanos entre sí. La mayoría de los *linkers* que los conectan presentan un giro que ubica a los D2s en una de las caras de D1, pero no en la otra. A la vez se puede ver que el *cluster* I cubre un espacio conformacional mayor que el *cluster* J. Estas dos características se condicen con lo observado en la Figura 52. No se puede concluir de este análisis que el *linker* no pueda plegarse hacia otros lados, sino que las estructuras que mejor representan el resultado experimental responden a ese tipo de conformaciones.

En cuanto a las distancias, se puede ver claramente como los centros de masa están ubicados en regiones similares, mientras que los metales tienen posiciones que difieren en forma sustancial. Mientras que los metales del LBT C-terminal del *cluster* I están entre los centros de masas y D1, los metales del LBT C-terminal del *cluster* J están apuntando en la dirección contraria. Esto explica los resultados de las medidas de distancia entre metales (Figura 55).

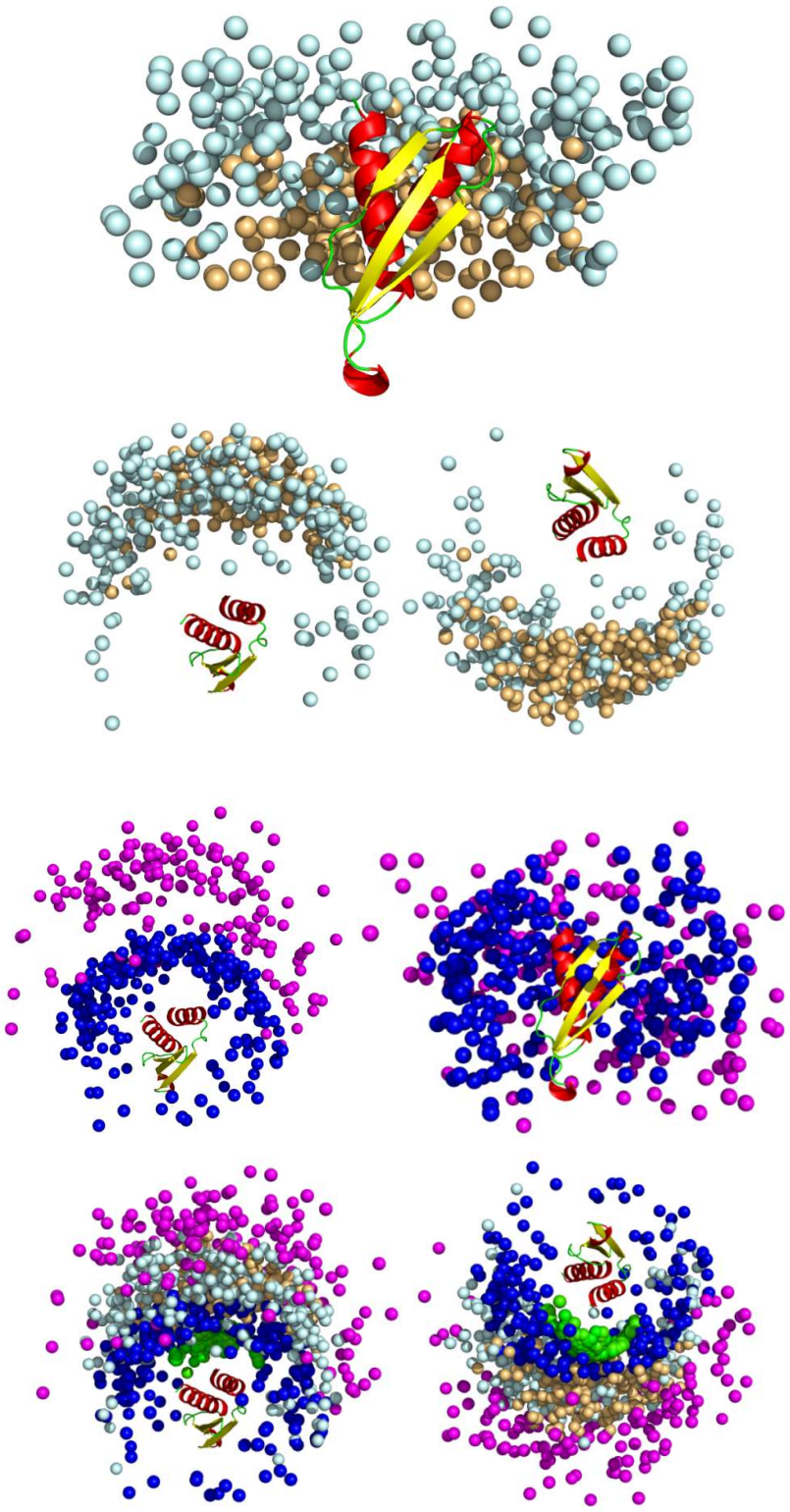


Figura 56: Representación tridimensional de los cluster I y J. Las estructuras fueron alineadas en D1 (en cartoon). Los centros de masa del segundo dominio se muestran como esferas celestes para el cluster I y como esferas naranjas para el cluster J. Los metales del LBT C-terminal se muestran como esferas azules para el cluster I y como esferas rosas para el cluster J. Los metales del LBT N-terminal se muestran como esferas verdes.

La restricción orientacional observada podría explicarse por la diferencia en las posiciones en las que se ubican los extremos en un dominio dsRBD. Dado que el extremo N-terminal no se posiciona en un extremo físico del dominio, sino que se ubica en el centro de la estructura, el perfil de conformaciones posibles desde el punto de vista de D1 no es simétrico con respecto a lo que se puede observar desde D2. Por ello, tanto la longitud como la posición del *linker* estarían limitando las conformaciones posibles. A su vez, la posición de los LBTs también es asimétrica. Esto podría estar conduciendo a que, pese a que existan conformaciones que cubren un amplio espectro de orientaciones, la ubicación de los LBTs condicionaría la población visible en los experimentos.

4.5.7. Conclusiones de la construcción *in silico* de conformaciones posibles y selección de estructuras

La interpretación de los resultados de PRE sobre las construcciones de D1-D2 que contienen etiquetas de LBT fue realizada mediante simulación y selección de estructuras. 50.000 conformaciones para L-D1-D2-L fueron simuladas. A partir de medidas de distancias entre residuos y metales para cada una de los modelos, se obtuvieron perfiles PRE teóricos. Las estructuras del ensamble inicial fueron separadas en grupos en base a los perfiles PRE calculados y, posteriormente, cada conjunto fue comparado con los resultados experimentales. Las estructuras de los grupos con perfiles PRE con las mayores diferencias con respecto a los experimentales fueron eliminadas. Este proceso de agrupamiento y selección fue repetido para refinar el ensamble. Dos *clusters,* de 233 y 155 estructuras respectivamente, mostraron las menores diferencias con respecto a los valores experimentales. Las estructuras seleccionadas mostraron dominios cercanos entre sí, pero dicha proximidad es alcanzada únicamente por una de las caras. La asimetría intrínseca de los extremos de los dominios podría explicar las restricciones orientacionales encontradas.

5. DISCUSIÓN

Los dominios en las proteínas multidominio están conectados por secuencias denominadas *linkers*. Estas regiones son variables, y dicha diversidad podría explicar la divergencia funcional que existe entre distintas proteínas. Numerosos estudios han tratado de entender la función de estas secuencias, y han demostrado que cumplen roles más allá de la mera unión covalente de estructuras [44–48,98]. Entre las proteínas multidominio conocidas, la familia DRB tiene una arquitectura global conservada, con dominios dsRBD en tándem[17] separados por *linkers* variables. La variabilidad de los *linkers* podría explicar la diferencia funcional entre distintos tipos de DRB. Este trabajo se focalizó en el estudio del rol que cumpliría el *linker* que conecta los dominios dsRBD en la proteína HYL1 de *Arabidopsis thaliana*.

Si bien existen trabajos previos que han recopilado información sobre proteínas de la familia DRB[17,99], realizan sus análisis focalizándose solamente en sus dominios, sin estudiar las características de sus *linkers*. Por lo tanto, como primer acercamiento en el análisis se decidió hacer un estudio bioinformático de los *linkers* que conectan dominios dsRBDs presentes en proteínas de distintos organismos. Para ello, se recopilaron las secuencias y taxonomías de proteínas de la base de datos Uniprot que contenían dominios dsRBD. Aquellas pertenecientes a especies de animales y de plantas fueron separadas, y se las analizó en función del número de dominios dsRBDs que contenían. El *linker* que conecta los dos dominios dsRBD de HYL1 tiene 17 residuos. El análisis de la longitud del *linker* para las proteínas de la base de datos mostró que existe un considerable nivel de conservación para longitudes de 16-17 residuos en proteínas de especies de plantas que contienen dos dominios dsRBDs. Debido a que no ocurre lo mismo para proteínas de especies animales, y a la notable conservación de la longitud en todo el reino *plantae*, se puede sugerir que la longitud del *linker* es importante para el funcionamiento de las proteínas de especies de plantas con dos dominios dsRBD. Luego se continuó con una evaluación de los *linkers* de proteínas de especies de plantas con dos dominios dsRBDs pero a nivel de secuencia. Las secuencias de las proteínas fueron alineadas y separadas en grupos en función de dicho alineamiento. Considerando que todas las proteínas que pudieron ser identificadas pertenecían a la familia DRB, se pudo clasificar a cada grupo formado a partir del alineamiento como tipos de DRBs. El estudio a nivel de secuencia de las proteínas de cada grupo mostró que el nivel de conservación de secuencia entre *linker* y dominios difiere solamente para proteínas de tipo DRB1, lo que podría ser clave en la diferenciación entre tipos DRBs. Por lo tanto, tanto el largo como la secuencia del *linker* de HYL1, y de las proteínas DRB1 en general, podrían ser importantes para su funcionamiento. Este es el

primer trabajo que analiza la variabilidad de longitud y secuencia de *linkers* entre dominios dsRBDs de proteínas de la familia DRB.

Conociendo la potencial importancia funcional del *linker* de HYL1, evidenciada por los estudios bioinformáticos, se decidió hacer un análisis *in vitro* de su función. La caracterización estructural de los dominios dsRBDs ha sido reportada previamente y el *linker* que los conecta es, en principio, flexible. Dado que es una estructura conectora que mantiene a los dominios unidos, su rol puede ser inferido a través de un estudio de la dinámica que existe entre los dominios dsRBDs que conecta. Para llevar a cabo este análisis se recurrió al uso de etiquetas paramagnéticas, que permiten realizar medidas en escalas de distancia apropiadas. Para la puesta a punto se diseñaron construcciones del sistema modelo 3Hx con la etiqueta LBT para la unión de metales paramagnéticos. Por otro lado, se generaron con éxito construcciones de D1-D2 que contienen la misma etiqueta.

La primera técnica utilizada para estudiar la dinámica entre dominios permitida por el *linker* fue PELDOR. Esta metodología permite medir la distribución de distancias entre dos metales paramagnéticos en una gran variedad de condiciones, incluso dentro de las células. Las construcciones de 3Hx que contienen etiquetas LBT fueron utilizadas para la puesta a punto. Se pudieron adquirir con éxito medidas tanto *in vitro* como *in cell* dentro de células de *E. coli*. Los estudios PELDOR *in cell* desarrollados hasta el momento implicaban la introducción de proteínas que habían sido marcadas *in vitro* a las células en estudio [65–70]. Este trabajo fue el primero en utilizar etiquetas autoensamblables que unen metales producidas biosintéticamente para realizar medidas no disruptivas *in cell*.

Se aplicó la misma técnica a las construcciones de D1-D2 con la etiqueta LBT, tanto en ausencia como en presencia de precursor. Sin embargo, los resultados fueron ambiguos, pudiéndose ajustar los datos experimentales a diferentes distribuciones de distancia. Esto se puede deber a que las distancias sean más largas que los límites detectables por la técnica o a que la distribución de distancias sea muy heterogénea. Por lo tanto en este sistema de estudio, PELDOR con LBT no permite determinar el espacio conformacional explorado por los dos dominios unidos por el *linker* flexible. Este resultado solamente permite concluir que los dos dominios de HYL1 no se encuentran cercanos entre sí y a una distancia fija.

La otra técnica estructural utilizada en este trabajo para medidas de distancia en proteínas es PRE. La misma permite identificar residuos cercanos a una sonda paramagnética. Los estudios fueron realizados sobre las construcciones de D1-D2 que contienen una etiqueta de LBT en alguno de sus extremos. El análisis cualitativo de los resultados indicó que los dos dominios no se acercan entre sí en forma simétrica. Por el contrario, el segundo dominio dsRBD de HYL1 se aproxima al extremo N-terminal de la proteína en forma más estrecha que lo que se acerca el primer dominio dsRBD al extremo

C-terminal. Esto sugiere que los dos dominios dsRBD de HYL1 tienen restricciones de movimiento debidos al *linker* que los conecta.

La interpretación de los resultados PRE interdominios es compleja, y existen diversas metodologías para su estudio[93–97]. El procedimiento general comprende la generación de ensambles, la predicción de sus patrones PRE y el cálculo de errores, para concluir con una interpretación de los resultados en el contexto biológico. En cuanto a la generación del ensamble los análisis comprenden distintas variantes, desde el estudio de la influencia del tamaño del ensamble[93], hasta la reducción del número de estructuras en base a la similitud en la orientación de los dominios en el espacio [97]. Entre las variantes en el cálculo de errores se puede mencionar un estudio en el que solamente se tuvieron en cuenta los valores de PRE entre dominios por ser los más informativos[97]. No existe un procedimiento unificado para la traducción de resultados de PRE en ensambles de conformaciones, y mucho menos se dispone de un software o algoritmo establecido para llevar a cabo dicho análisis. Por todo esto, se decidió desarrollar un nuevo enfoque para el análisis de los resultados basado en los métodos ya probados en bibliografía. Para obtener una explicación estructural de los resultados de PRE, se propuso generar un ensamble de un gran número de conformaciones posibles, dividirlas en grupos en base a la similitud de sus patrones predichos, y seleccionar aquellas que se corresponden con los resultados experimentales. Se obtuvo un modelo inicial para la estructura de D1-D2 que contiene etiquetas LBTs en sus extremos. A partir de dicha estructura se modelaron 50000 conformaciones posibles, aleatorizando la conformación del *linker* y de los residuos que conectan al LBT. Luego se las separó en grupos en función de la similitud de sus perfiles PRE teóricos calculados. Los grupos con los patrones que más se corresponden con los experimentales fueron conservados y el resto fueron apartados del análisis. Este procedimiento de agrupamiento y selección fue repetido nuevamente. El grupo final quedó conformado por 388 estructuras que presentan dominios D2 próximos a una de las caras de D1. La asimetría encontrada podría explicarse por la asimetría en la posición de los extremos de los dominios. La ubicación central del extremo N-terminal con respecto a la posición terminal del extremo C-terminal hacen que el movimiento permitido por el *linker* no sea simétrico.

Por lo expuesto en este trabajo tanto en el análisis bioinformático como *in vitro* se puede concluir que los dos dominios de HYL1 se mueven libremente. Sin embargo, la longitud y posición del *linker* produce una asimetría en la libertad conformacional que estaría dada por restricciones geométricas simples. La alta conservación en longitud observada entre las proteínas DRB de plantas indica que esta característica estructural impuesta por el *linker* podría ser importante para la función de estas proteínas.

6. REFERENCIAS BIBLIOGRÁFICAS

1. Fukudome, A. & Fukuhara, T. Plant dicer-like proteins: double-stranded RNA-cleaving enzymes for small RNA biogenesis. *J. Plant Res.* **130**, 33–44 (2017).
2. Yu, Y., Jia, T. & Chen, X. The 'how' and 'where' of plant microRNAs. *New Phytol.* **216**, 1002–1017 (2017).
3. Budak, H. & Akpinar, B. A. Plant miRNAs: biogenesis, organization and origins. *Funct. Integr. Genomics* **15**, 523–531 (2015).
4. Rogers, K. & Chen, X. Biogenesis, Turnover, and Mode of Action of Plant MicroRNAs. *Plant Cell* **25**, 2383–2399 (2013).
5. Xie, Z., Khanna, K. & Ruan, S. Expression of microRNAs and its regulation in plants. *Semin. Cell Dev. Biol.* **21**, 790–797 (2010).
6. Bologna, N. G. & Voinnet, O. The Diversity, Biogenesis, and Activities of Endogenous Silencing Small RNAs in *Arabidopsis*. *Annu. Rev. Plant Biol.* **65**, 473–503 (2014).
7. Liu, Q., Feng, Y. & Zhu, Z. Dicer-like (DCL) proteins in plants. *Funct. Integr. Genomics* **9**, 277–286 (2009).
8. Bologna, N. G., Mateos, J. L., Bresso, E. G. & Palatnik, J. F. A loop-to-base processing mechanism underlies the biogenesis of plant microRNAs miR319 and miR159. *Embo J* **28**, 3646–3656 (2009).
9. Axtell, M. J., Westholm, J. O. & Lai, E. C. Vive la différence: biogenesis and evolution of microRNAs in plants and animals. *Genome Biol.* **12**, 1–13 (2011).
10. Ren, G., Chen, X. & Yu, B. Small RNAs meet their targets: When methylation defends miRNAs from uridylation. *RNA Biol.* **11**, 1099–1104 (2014).
11. Masliah, G., Barraud, P. & Allain, F. H. T. RNA recognition by double-stranded RNA binding domains: A matter of shape and sequence. *Cell. Mol. Life Sci.* **70**, 1875–1895 (2013).
12. Ryter, J. M. & Schultz, S. C. Molecular basis of double-stranded RNA-protein interactions: Structure of a dsRNA-binding domain complexed with dsRNA. *EMBO J.* **17**, 7505–7513 (1998).
13. Ramos, A. *et al.* RNA recognition by a Staufen double□stranded RNA□binding domain | The EMBO Journal. *EMBO J.* **19**, 997–1009 (2000).
14. Blaszczyk, J. *et al.* Noncatalytic assembly of ribonuclease III with double-stranded RNA. *Structure* **12**, 457–466 (2004).
15. Stefl, R. *et al.* The Solution Structure of the ADAR2 dsRBM-RNA Complex Reveals a Sequence-Specific Readout of the Minor Groove. *Cell* **143**, 225–237 (2010).
16. Burdisso, P. *et al.* Structural determinants of Arabidopsis thaliana Hyponastic leaves 1 function in vivo. *PLoS One* **9**, e113243 (2014).

17. Clavel, M. *et al.* Evolutionary history of double-stranded RNA binding proteins in plants: identification of new cofactors involved in easiRNA biogenesis. *Plant Mol. Biol.* **91**, 131–147 (2016).
18. Mackereth, C. D. & Sattler, M. Dynamics in multi-domain protein recognition of RNA. *Curr. Opin. Struct. Biol.* **22**, 287–296 (2012).
19. Sohn, S. Y. *et al.* Crystal structure of human DGCR8 core. *Nat. Struct. Mol. Biol.* **14**, 847–853 (2007).
20. Shamoo, Y., Abdul-manan, N. & Williams, K. R. Multiple RNA binding domains (RBDs) just don't add up. *Nucleic Acids Res.* **23**, 725–728 (1995).
21. Benoit, M. P. M. H. *et al.* The RNA-binding region of human TRBP interacts with microRNA precursors through two independent domains. *Nucleic Acids Res.* **41**, 4241–4252 (2013).
22. Heyam, A., Lagos, D. & Plevin, M. Dissecting the roles of TRBP and PACT in double-stranded RNA recognition and processing of noncoding RNAs. *Wiley Interdiscip. Rev. RNA* **6**, 271–289 (2015).
23. Heyam, A. *et al.* Conserved asymmetry underpins homodimerization of Dicer-Associated double-stranded RNA-binding proteins. *Nucleic Acids Res.* **45**, 12577–12584 (2017).
24. Wilson, R. C. *et al.* Dicer-TRBP complex formation ensures accurate mammalian MicroRNA biogenesis. *Mol. Cell* **57**, 397–408 (2015).
25. Liu, Z. *et al.* Cryo-EM Structure of Human Dicer and Its Complexes with a Pre-miRNA Substrate. *Cell* **173**, 1191–1203.e12 (2018).
26. Koh, H. R., Kidwell, M. A., Ragunathan, K., Doudna, J. A. & Myong, S. ATP-independent diffusion of double-stranded RNA binding proteins. *Proc. Natl. Acad. Sci.* **110**, 151–156 (2013).
27. Wang, X., Vukovic, L., Koh, H. R., Schulten, K. & Myong, S. Dynamic profiling of double-stranded RNA binding proteins. *Nucleic Acids Res.* **43**, 7566–7576 (2015).
28. Tants, J. N. *et al.* Molecular basis for asymmetry sensing of siRNAs by the Drosophila Loqs-PD/Dcr-2 complex in RNA interference. *Nucleic Acids Res.* **45**, 12536–12550 (2017).
29. Chiliveri, S. C. & Deshmukh, M. V. Structure of RDE-4 dsRBDs and mutational studies provide insights into dsRNA recognition in the *Caenorhabditis elegans* RNAi pathway. *Biochem. J.* **458**, 119–130 (2014).
30. Parker, G. S., Maity, T. S. & Bass, B. L. dsRNA Binding Properties of RDE-4 and TRBP Reflect Their Distinct Roles in RNAi. *J. Mol. Biol.* **384**, 967–979 (2008).
31. Blanchard, D. *et al.* On the nature of in vivo requirements for rde-4 in RNAi and developmental pathways in C. elegans. *RNA Biol.* **8**, 458–467 (2011).

32. Chiliveri, S. C., Aute, R., Rai, U. & Deshmukh, M. V. DRB4 dsRBD1 drives dsRNA recognition in Arabidopsis thaliana tasi/siRNA pathway. *Nucleic Acids Res.* **45**, 8551–8563 (2017).
33. Achkar, N. P., Cambiagno, D. A. & Manavella, P. A. miRNA Biogenesis: A Dynamic Pathway. *Trends Plant Sci.* **21**, 1034–1044 (2016).
34. Hiraguri, A. *et al.* Specific interactions between Dicer-like proteins and HYL1/DRB-family dsRNA-binding proteins in Arabidopsis thaliana. *Plant Mol. Biol.* **57**, 173–188 (2005).
35. Moro, B. *et al.* Efficiency and precision of microRNA biogenesis modes in plants. *Nucleic Acids Res.* **46**, 10709–10723 (2018).
36. Lu, C. & Fedoroff, N. A Mutation in the Arabidopsis HYL1 Gene Encoding a dsRNA Binding Protein Affects Responses to Abscisic Acid, Auxin, and Cytokinin. *Plant Cell* **12**, 2351–2365 (2000).
37. Liu, C., Axtell, M. J. & Fedoroff, N. V. The Helicase and RNaseIIIa Domains of Arabidopsis Dicer-Like1 Modulate Catalytic Parameters during MicroRNA Biogenesis. *Plant Physiol.* **159**, 748–758 (2012).
38. Tagami, Y., Motose, H. & Watanabe, Y. A dominant mutation in DCL1 suppresses the hyl1 mutant phenotype by promoting the processing of miRNA. *Rna* **15**, 450–458 (2009).
39. Manavella, P. A. *et al.* Fast-forward genetics identifies plant CPL phosphatases as regulators of miRNA processing factor HYL1. *Cell* **151**, 859–870 (2012).
40. Dong, Z., Han, M.-H. & Fedoroff, N. The RNA-binding proteins HYL1 and SE promote accurate in vitro processing of pri-miRNA by DCL1. *Proc. Natl. Acad. Sci.* **105**, 9970–9975 (2008).
41. Kurihara, Y., Takashi, Y. & Watanabe, Y. The interaction between DCL1 and HYL1 is important for efficient and precise processing of pri-miRNA in plant microRNA biogenesis. *RNA* **12**, 206–212 (2006).
42. Wu, F. *et al.* The N-terminal double-stranded RNA binding domains of Arabidopsis HYPONASTIC LEAVES1 are sufficient for pre-microRNA processing. *Plant Cell* **19**, 914–925 (2007).
43. Rasia, R. M. *et al.* Structure and RNA interactions of the plant MicroRNA processing-associated protein HYL1. *Biochemistry* **49**, 8237–9 (2010).
44. Papaleo, E. *et al.* The Role of Protein Loops and Linkers in Conformational Dynamics and Allostery. *Chem. Rev.* **116**, 6391–6423 (2016).
45. Ma, B., Tsai, C. J., Haliloğlu, T. & Nussinov, R. Dynamic allostery: Linkers are not merely flexible. *Structure* **19**, 907–917 (2011).
46. George, R. A. & Heringa, J. An analysis of protein domain linkers: their classification

and role in protein folding. *Protein Eng. Des. Sel.* **15**, 871–879 (2002).

47. Gokhale, R. S. & Khosla, C. Role of linkers in communication between protein modules. *Curr. Opin. Chem. Biol.* **4**, 22–27 (2000).
48. Wriggers, W., Chakravarty, S. & Jennings, P. A. Control of protein functional dynamics by peptide linkers. *Biopolym. - Pept. Sci. Sect.* **80**, 736–746 (2005).
49. Marius Clore, G. & Iwahara, J. Theory, Practice and Applications of Paramagnetic Relaxation Enhancement for the Characterization of Transient Low- Population States of Biological Macromolecules and Their Complexes. *NIH Public Access* **109**, 4108–4139 (2009).
50. Koehler, J. & Meiler, J. Expanding the utility of NMR restraints with paramagnetic compounds: background and practical aspects. *Prog. Nucl. Magn. Reson. Spectrosc.* **59**, 360–89 (2011).
51. Fragai, M., Luchinat, C., Parigi, G. & Ravera, E. Conformational freedom of metalloproteins revealed by paramagnetism-assisted NMR. *Coord. Chem. Rev.* **257**, 2652–2667 (2013).
52. Delaforge, E. *et al.* Investigating the Role of Large-Scale Domain Dynamics in Protein-Protein Interactions. *Front. Mol. Biosci.* **3**, 1–8 (2016).
53. Barthelmes, K. & Allen, K. N. Encoded loop-lanthanide-binding tags for long-range distance measurements in proteins by NMR and EPR spectroscopy. *J. Biomol. NMR* **63**, 275–282 (2015).
54. Wöhnert, J., Franz, K. J., Nitz, M., Imperiali, B. & Schwalbe, H. Protein Alignment by a Coexpressed Lanthanide-Binding Tag for the Measurement of Residual Dipolar Couplings. *J. Am. Chem. Soc.* **125**, 13338–13339 (2003).
55. Daughtry, K. D., Martin, L. J., Sarraju, A., Imperiali, B. & Allen, K. N. Tailoring Encodable Lanthanide-Binding Tags as MRI Contrast Agents. *ChemBioChem* **13**, 2567–2574 (2012).
56. Barthelmes, K. *et al.* Engineering encodable lanthanide-binding tags into loop regions of proteins. *J. Am. Chem. Soc.* **133**, 808–819 (2011).
57. Barb, A. W. & Subedi, G. P. An encodable lanthanide binding tag with reduced size and flexibility for measuring residual dipolar couplings and pseudocontact shifts in large proteins. *J. Biomol. NMR* **64**, 75–85 (2016).
58. Allen, K. N. & Imperiali, B. Lanthanide-tagged proteins--an illuminating partnership. *Curr. Opin. Chem. Biol.* **14**, 247–54 (2010).
59. Sculimbrene, B. R. & Imperiali, B. Lanthanide-Binding Tags as Luminescent Probes for Studying Protein Interactions. *J Am Chem Soc* **128**, 7346–7352 (2006).
60. Di Gennaro, A. K., Gurevich, L., Skovsen, E., Overgaard, M. T. & Fojan, P. Study of the tryptophan-terbium FRET pair coupled to silver nanoprisms for biosensing

applications. *Phys. Chem. Chem. Phys.* **15**, 8838–8844 (2013).

61. Clore, G. M. Interplay between conformational selection and induced fit in multidomain protein-ligand binding probed by paramagnetic relaxation enhancement. *Biophys. Chem.* **186**, 3–12 (2014).
62. Pascal, S. M. *An HSQC-based Approach*. (IM Publications LLP, 2008).
63. Schiemann, O. & Prisner, T. F. Long-range distance determinations in biomacromolecules by EPR spectroscopy. *Q. Rev. Biophys.* **40**, 1–53 (2007).
64. Ching, H. Y. V. *et al.* The Use of Mn(II) Bound to His-tags as Genetically Encodable Spin-Label for Nanometric Distance Determination in Proteins. *J. Phys. Chem. Lett.* **7**, 1072–1076 (2016).
65. Azarkh, M. *et al.* Long-range distance determination in a DNA model system inside Xenopus laevis Oocytes by In-Cell Spin-Label EPR. *ChemBioChem* **12**, 1992–1995 (2011).
66. Igarashi, R. *et al.* Distance determination in proteins inside xenopus laevis oocytes by double electron-electron resonance experiments. *J. Am. Chem. Soc.* **132**, 8228–8229 (2010).
67. Krstić, I. *et al.* Long-range distance measurements on nucleic acids in cells by pulsed EPR spectroscopy. *Angew. Chemie - Int. Ed.* **50**, 5070–5074 (2011).
68. Qi, M., Groß, A., Jeschke, G., Godt, A. & Drescher, M. Gd(III)-PyMTA label is suitable for in-cell EPR. *J. Am. Chem. Soc.* **136**, 15366–15378 (2014).
69. Martorana, A. *et al.* Probing protein conformation in cells by EPR distance measurements using Gd3+spin labeling. *J. Am. Chem. Soc.* **136**, 13458–13465 (2014).
70. Theillet, F.-X. *et al.* Structural disorder of monomeric α-synuclein persists in mammalian cells. *Nature* **530**, 45–50 (2016).
71. Jagtap, A. P. *et al.* Sterically shielded spin labels for in-cell EPR spectroscopy: Analysis of stability in reducing environment. *Free Radic. Res.* **49**, 78–85 (2015).
72. Matalon, E. *et al.* Gadolinium(III) spin labels for high-sensitivity distance measurements in transmembrane helices. *Angew. Chemie - Int. Ed.* **52**, 11831–11834 (2013).
73. Abdelkader, E. H. *et al.* Protein conformation by EPR spectroscopy using gadolinium tags clicked to genetically encoded p-azido-l-phenylalanine. *Chem. Commun.* **51**, 15898–15901 (2015).
74. Vincent Ching, H. Y. *et al.* Nanometric distance measurements between Mn(II)DOTA centers. *Phys. Chem. Chem. Phys.* **17**, 23368–23377 (2015).
75. Martorana, A. *et al.* Mn(II) tags for DEER distance measurements in proteins via C-S attachment. *Dalt. Trans.* **44**, 20812–20816 (2015).

76. Cunningham, T. F. *et al.* Cysteine-specific Cu2+chelating tags used as paramagnetic probes in double electron electron resonance. *J. Phys. Chem. B* **119**, 2839–2843 (2015).
77. Hanahan, D. & Harbor, C. S. Studies on Transformation of Escherichia coli with Plasmids. *J. Mol. Biol.* **166**, 557–580 (1983).
78. Kapust, R. B. *et al.* Tobacco etch virus protease: mechanism of autolysis and rational design of stable mutants with wild-type catalytic proficienc. *Protein Eng. Des. Sel.* **14**, 993–1000 (2002).
79. Huang, P.-S. *et al.* High thermodynamic stability of parametrically designed helical bundles. *Science* **346**, 481–5 (2014).
80. Kuipers, B. & Gruppen, H. Prediction of Molar Extinction Coefficients of Proteins and Peptides Using UV absorption of the constituent amino acids at 214 nm to enable quantitative reverse phase high-performance liquid chromatography-mass spectrometry analysis. *J. Agric. food …* **55**, 5445–5451 (2007).
81. Martin, R. E. *et al.* Determination of End-to-End Distances in a Series of TEMPO Diradicals of up to 2.8 nm Length with a New Four-Pulse Double Electron Electron Resonance Experiment. *Angew. Chemie Int. Ed.* **37**, 2833–2837 (1998).
82. Delaglio, F. *et al.* NMRPipe: A multidimensional spectral processing system based on UNIX pipes. *J. Biomol. NMR* **6**, 277–293 (1995).
83. Vranken, W. F. *et al.* The CCPN data model for NMR spectroscopy: Development of a software pipeline. *Proteins Struct. Funct. Genet.* **59**, 687–696 (2005).
84. Barb, A. W., Ho, T. G., Flanagan-Steet, H. & Prestegard, J. H. Lanthanide binding and IgG affinity construct: Potential applications in solution NMR, MRI, and luminescence microscopy. *Protein Sci.* **21**, 1456–1466 (2012).
85. Kumar, S., Stecher, G., Li, M., Knyaz, C. & Tamura, K. MEGA X: Molecular evolutionary genetics analysis across computing platforms. *Mol. Biol. Evol.* **35**, 1547–1549 (2018).
86. Sali, A. & Tom, B. Comparative protein modeling by satisfaction of spatial restraints. *Molecular Medicine Today* **1**, 270–277 (1995).
87. Chaudhury, S., Lyskov, S. & Gray, J. J. PyRosetta: a script-based interface for implementing molecular modeling algorithms using Rosetta. *Bioinformatics* **26**, 689–691 (2010).
88. Apweiler, R. *et al.* UniProt: The universal protein knowledgebase. *Nucleic Acids Res.* **45**, D158–D169 (2017).
89. Mascali, F. C., Ching, H. Y. V., Rasia, R. M., Un, S. & Tabares, L. C. Using Genetically Encodable Self-Assembling Gd III Spin Labels to Make In-cell Nanometric Distance Measurements. *Angew. Chemie Int. Ed.* **55**, 11041–11043 (2016).

90. Volkmer, B. & Heinemann, M. Condition-dependent cell volume and concentration of Escherichia coli to facilitate data conversion for systems biology modeling. *PLoS One* **6**, e23126 (2011).
91. Yang, S. W. *et al.* Structure of Arabidopsis HYPONASTIC LEAVES1 and its molecular implications for miRNA processing. *Structure* **18**, 594–605 (2010).
92. Nitz, M. *et al.* Structural origin of the high affinity of a chemically evolved lanthanide-binding peptide. *Angew. Chemie - Int. Ed.* **43**, 3682–3685 (2004).
93. Tang, C., Iwahara, J. & Clore, G. M. Visualization of transient encounter complexes in protein-protein association. *Nature* **444**, 383–386 (2006).
94. Iwahara, J., Schwieters, C. D. & Clore, G. M. Ensemble Approach for NMR Structure Refinement Against 1 H Paramagnetic Relaxation Enhancement Data Arising from a Flexible Paramagnetic Group Attached to a Macromolecule. *J. Am. Chem. Soc* **126**, 5879–5896 (2004).
95. MacKereth, C. D. *et al.* Multi-domain conformational selection underlies pre-mRNA splicing regulation by U2AF. *Nature* **475**, 408–413 (2011).
96. Simon, B., Madl, T., Mackereth, C. D., Nilges, M. & Sattler, M. An efficient protocol for NMR-spectroscopy-based structure determination of protein complexes in solution. *Angew. Chemie - Int. Ed.* **49**, 1967–1970 (2010).
97. Zhu, G. *et al.* Investigating energy-based pool structure selection in the structure ensemble modeling with experimental distance constraints: The example from a multidomain protein Pub1. *Proteins Struct. Funct. Genet.* **86**, 501–514 (2018).
98. Bhaskara, R. M., de Brevern, A. G. & Srinivasan, N. Understanding the role of domain-domain linkers in the spatial orientation of domains in multi-domain proteins. *J. Biomol. Struct. Dyn.* **31**, 1467–80 (2013).
99. Dias, R., Manny, A., Kolaczkowski, O. & Kolaczkowski, B. Convergence of Domain Architecture , Structure , and Ligand Affinity in Animal and Plant RNA-Binding Proteins. **34**, 1429–1444 (2017).

7. ANEXO

7.1. Genes sintéticos

L-D1-L-D2-L

CCATGGGCAGCAGCCATCATCATCATCATCACAGCAGCGGCGAAAACCTGTATTTTCAGGGCTATATTGATA
CCAACAACGATGGCTGGATTGAAGGCGATGAACTGCATATGGTTTTCAAAAGTCGGTTGCAGGAGTATGCTC
AGAAGTACAAGCTCCCAACGCCTGTTTATGAGATCGTTAAAGAAGGCCCTTCACACAAATCTTTATTTCAATC
GACTGTGATACTGGATGGTGTCAGATATAATTCTTTGCCTGGATTCTTCAATCGTAAGGCTGCAGAGCAATCA
GCTGCCGAGGTTGCTCTCCGGGAATTAGCAAAATCCAGTGAGCTCTATATTGATACCAACAACGATGGCTGG
ATTGAAGGCGATGAACTGGAGCTCAGCCAATGTGTTTCACAACCTGTTCACGAAACGGGATTATGCAAGAAC
CTACTTCAAGAATACGCTCAAAAGATGAATTACGCGATTCCATTGTATCAGTGCCAGAAGGTCGAAACTCTTG
GGAGAGTTACACAATTCACATGTACTGTAGAGATTGGAGGCATAAAGTACACAGGAGCTGCAACAAGAACTA
AAAAAGATGCTGAGATTAGCGCTGGGAGAACTGCTCTTTTAGCGATCCAGTCAGTCGACTATATTGATACCA
ACAACGATGGCTGGATTGAAGGCGATGAACTGGTCGACTGACTCGAG

D1-D2_noCys

(CCATGG)NcoI

GCAGCAGCCATCATCATCATCATCACAGCAGCGGCGAAAACCTGTATTTTCAGGGCCATATGACCTCTACGG
ATGTGTCTAGTGGTGTCTCAAATAGCTACGTGTTCAAAAGCCGCCTGCAAGAATACGCTCAGAAATACAAAC
TGCCGACCCCGGTGTATGAAATTGTTAAAGAAGGCCCGTCGCATAAAAGCCTGTTTCAGTCTACGGTTATCC
TGGATGGCGTCCGTTACAACAGTCTGCCGGGCTTTTTCAATCGCAAAGCGGCCGAACAAAGCGCGGCGGAA
GTGGCACTGCGTGAACTGGCTAAAAGCTCTGAACTGAGTCAGTCCGTCTCACAACCGGTGCACGAAACCGG
CCTGTCAAAAAACCTGCTGCAAGAATATGCGCAAAAAATGAATTATGCCATTCCGCTGTACCAGTCGCAAAA
AGTGGAAACCCTGGGTCGCGTTACGCAGTTCACCAGCACGGTTGAAATTGGCGGTATCAAATACACCGGCG
CGGCCACCCGTACGAAAAAAGACGCGGAAATTTCCGCCGGTCGTACGGCACTGCTGGCTATCCAGTCATAA
GGATCC(BamHI)

7.2. Construcciones

En la tabla siguiente se muestra información de las construcciones utilizadas en este trabajo. En las secuencias que se muestran en la segunda columna se indican: sitios de corte en negrita, 3Hx en naranja, LBT en rojo, His-tag en azul, D1 en verde, *linker* en rosa y D2 en violeta.

Datos de la construcción	Secuencia ADN / proteína
Nombre: 3Hx N° residuos: 72 Peso molecular: 8436.2 Generación del clonado: colaboración E. Bruch Producción de la proteína: *in cell* Aplicación: puesta a punto sistema PELDOR	**CCATGG**GCACCGAAGAAGAAATTAAAAAACTGGAAGAAGAAGC GAAAAAACTGCTGGAAAAACTGAAAAAAAACGGCGTGACCACC ACCATTATTGAAGAAGTGAAGAAGAAGATGGAA**GAGCTC**CTGA AAAAACTGAAAAACAGCACCAAAACCAAAGAAGCGGCGGAAAA AATGCTGAAAAAAATGAAAGAACTGTTTAAAAAAGCGAAACTGG AATAA**CTCGAG** MGTEEEIKKLEEEAKKLLEKLKKNGVTTTIIEEVKKKMEELLKKLKN STKTKEAAEKMLK KMKELFKKAKLE
Nombre: L-3Hx N° residuos: 87 Peso molecular: 10171.98 Generación del clonado: corte y digestión sitio SacI entre 3Hx L-3Hx-L y 3Hx Producción de la proteína: *in cell* Aplicación: puesta a punto sistema PELDOR	**CCATGG**GCTATATTGATACCAACAACGATGGCTGGATTGAAGG CGATGAACTGACCGAAGAAGAAATTAAAAAACTGGAAGAAGAA GCGAAAAAACTGCTGGAAAAACTGAAAAAAAACGGCGTGACCA CCACCATTATTGAAGAAGTGAAGAAGAAGATGGAAGAGCTCCT GAAAAAACTGAAAAACAGCACCAAAACCAAAGAAGCGGCGGAA AAAATGCTGAAAAAAATGAAAGAACTGTTTAAAAAAGCGAAACT GGAATAA**CTCGAG** MGYIDTNNDGWIEGDELTEEEIKKLEEEAKKLLEKLKKNGVTTTIIE EVKKKMEELLKKLKNSTKTKEAAEKMLKKMKELFKKAKLE
Nombre: 3Hx-L N° residuos: 87 Peso molecular: 10171.98	**CCATGG**GCACCGAAGAAGAAATTAAAAAACTGGAAGAAGAAGC GAAAAAACTGCTGGAAAAACTGAAAAAAAACGGCGTGACCACC ACCATTATTGAAGAAGTGAAGAAGAAGATGGAAGAGCTCCTGA

Generación del clonado: corte y digestión sitio SacI entre 3Hx y L-3Hx-L Producción de la proteína: *in cell* Aplicación: puesta a punto sistema PELDOR	AAAAACTGAAAAACAGCACCAAAACCAAAGAAGCGGCGGAAAA AATGCTGAAAAAAATGAAAGAACTGTTTAAAAAAGCGAAACTGG AATACATCGACACGAATAATGACGGTTGGATCGAGGGTGACGA GCTGTAA**CTCGAG** MGTEEEIKKLEEEAKKLLEKLKKNGVTTTIIEEVKKKMEELLKKLKN STKTKEAAEKMLKKMKELFKKAKLEYIDTNNDGWIEGDEL
Nombre: L-3Hx-L N° residuos: 102 Peso molecular: 11907.7 Generación del clonado: colaboración E. Bruch Producción de la proteína: *in cell* Aplicación: puesta a punto sistema PELDOR	**CCATGG**GCTATATTGATACCAACAACGATGGCTGGATTGAAGG CGATGAACTGACCGAAGAAGAAATTAAAAAACTGGAAGAAGAA GCGAAAAAACTGCTGGAAAAACTGAAAAAAAACGGCGTGACCA CCACCATTATTGAAGAAGTGAAGAAGAAGATGGAA**GAGCTC**CT GAAAAAACTGAAAAACAGCACCAAAACCAAAGAAGCGGCGGAA AAAATGCTGAAAAAAATGAAAGAACTGTTTAAAAAAGCGAAACT GGAATACATCGACACGAATAATGACGGTTGGATCGAGGGTGAC GAGCTGTAA**CTCGAG** MGYIDTNNDGWIEGDELTEEEIKKLEEEAKKLLEKLKKNGVTTTIIE EVKKKMEELLKKLKNSTKTKEAAEKMLKKMKELFKKAKLEYIDTNN DGWIEGDEL
Nombre: H6-L-3Hx-L N° residuos: 123 Peso molecular: 14313.27 Generación del clonado: colaboración E. Bruch Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV. Aplicación: puesta a punto sistema PELDOR	**CATATG**GGCAGCAGCCATCATCATCATCATCACAGCAGCGGCG AAAACCTGTATTTTCAGGGCCATATGGGCTATATTGATACCAAC AACGATGGCTGGATTGAAGGCGATGAACTGACCGAAGAAGAAA TTAAAAAACTGGAAGAAGAAGCGAAAAAACTGCTGGAAAAACT GAAAAAAAACGGCGTGACCACCACCATTATTGAAGAAGTGAAG AAGAAGATGGAA**GAGCTC**CTGAAAAAACTGAAAAACAGCACCA AAACCAAAGAAGCGGCGGAAAAAATGCTGAAAAAAATGAAAGA ACTGTTTAAAAAAGCGAAACTGGAATACATCGACACGAATAATG ACGGTTGGATCGAGGGTGACGAGCTGTAA**CTCGAG** GHMGYIDTNNDGWIEGDELTEEEIKKLEEEAKKLLEKLKKNGVTTT IIEEVKKKMEELLKKLKNSTKTKEAAEKMLKKMKELFKKAKLEYIDT NNDGWIEGDEL
Nombre: D1-D2 N° residuos: 173 Peso molecular: 19067.7 Generación del clonado: generado por el Dr. Nicolás Bologna (grupo del Dr. Javier Palatnik) Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV. Aplicación: estudios de PRE y controles de PELDOR	ATGGGCAGCAGCCATCATCATCATCATCACAGCAGCGGCGAAA ACCTGTATTTTCAGGGCCATATGATGACCTCCACTGATGTTTCC TCTGGTGTTTCCAATTGCTATGTTTTCAAAAGTCGGTTGCAGGA GTATGCTCAGAAGTACAAGCTCCCAACGCCTGTTTATGAGATC GTTAAAGAAGGCCCTTCACACAAATCTTTATTTCAATCGACTGT GATACTGGATGGTGTCAGATATAATTCTTTGCCTGGATTCTTCA ATCGTAAGGCTGCAGAGCAATCAGCTGCCGAGGTTGCTCTCCG GGAATTAGCAAAATCCAGTGAGCTAAGCCAATGTGTTTCACAAC CTGTTCACGAAACGGGATTATGCAAGAACCTACTTCAAGAATAC GCTCAAAAGATGAATTACGCGATTCCATTGTATCAGTGCCAGAA GGTCGAAACTCTTGGGAGAGTTACACAATTCACATGTACTGTA GAGATTGGAGGCATAAAGTACACAGGAGCTGCAACAAGAACTA AAAAAGATGCTGAGATTAGCGCTGGGAGAACTGCTCTTTTAGC GATCCAGTCATGA GHMMTSTDVSSGVSNCYVFKSRLQEYAQKYKLPTPVYEIVKEGP SHKSLFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSS ELSQCVSQPVHETGLCKNLLQEYAQKMNYAIPLYQCQKVETLGRV TQFTCTVEIGGIKYTGAATRTKKDAEISAGRTALLAIQS
Nombre: D1-D2_NoCys N° residuos: 170 Peso molecular: 18662.1 Generación del clonado: a partir de gen sintético Proteína inestable	ATGACCTCTACGGATGTGTCTAGTGGTGTCTCAAATAGCTACGT GTTCAAAAGCCGCCTGCAAGAATACGCTCAGAAATACAAACTG CCGACCCCGGTGTATGAAATTGTTAAAGAAGGCCCGTCGCATA AAAGCCTGTTTCAGTCTACGGTTATCCTGGATGGCGTCCGTTA CAACAGTCTGCCGGGCTTTTTCAATCGCAAAGCGGCCGAACAA AGCGCGGCGGAAGTGGCACTGCGTGAACTGGCTAAAAGCTCT GAACTGAGTCAGTCCGTCTCACAACCGGTGCACGAAACCGGC CTGTCAAAAAACCTGCTGCAAGAATATGCGCAAAAAATGAATTA TGCCATTCCGCTGTACCAGTCGCAAAAAGTGGAAACCCTGGGT CGCGTTACGCAGTTCACCAGCACGGTTGAAATTGGCGGTATCA AATACACCGGCGCGGCCACCCGTACGAAAAAAGACGCGGAAA

	TTTCCGCCGGTCGTACGGCACTGCTGGCTATCCAGTCATAA MTSTDVSSGVSNSYVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKS LFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSSELSQS VSQPVHETGLSKNLLQEYAQKMNYAIPLYQSQKVETLGRVTQFTS TVEIGGIKYTGAATRTKKDAEISAGRTALLAIQS
Nombre: D1-D2_NoCys_S105V N° residuos: 170 Peso molecular: 18764.1 Generación del clonado: a partir de gen sintético Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV.	ATGACCTCTACGGATGTGTCTAGTGGTGTCTCAAATAGCTACGT GTTCAAAAGCCGCCTGCAAGAATACGCTCAGAAATACAAACTG CCGACCCCGGTGTATGAAATTGTTAAAGAAGGCCCGTCGCATA AAAGCCTGTTTCAGTCTACGGTTATCCTGGATGGCGTCCGTTA CAACAGTCTGCCGGGCTTTTTCAATCGCAAAGCGGCCGAACAA AGCGCGGCGGAAGTGGCACTGCGTGAACTGGCTAAAAGCTCT GAACTGAGTCAGTCCGTCTCACAACCGGTGCACGAAACCGGC CTGGTAAAAAACCTGCTGCAAGAATATGCGCAAAAAATGAATTA TGCCATTCCGCTGTACCAGTCGCAAAAAGTGGAAACCCTGGGT CGCGTTACGCAGTTCACCAGCACGGTTGAAATTGGCGGTATCA AATACACCGGCGCGGCCACCCGTACGAAAAAAGACGCGGAAA TTTCCGCCGGTCGTACGGCACTGCTGGCTATCCAGTCATAAGG A MTSTDVSSGVSNSYVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKS LFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSSELSQS VSQPVHETGLVKNLLQEYAQKMNYAIPLYQSQKVETLGRVTQFTS TVEIGGIKYTGAATRTKKDAEISAGRTALLAIQS
Nombre: D1-D2_NoCys_S105V_K31C N° residuos: 170 Peso molecular: 18649.12 Generación del clonado: a partir de gen sintético Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV.	ATGACCTCTACGGATGTGTCTAGTGGTGTCTCAAATAGCTACGT GTTCAAAAGCCGCCTGCAAGAATACGCTCAGAAATACTGCCTG CCGACCCCGGTGTATGAAATTGTTAAAGAAGGCCCGTCGCATA AAAGCCTGTTTCAGTCTACGGTTATCCTGGATGGCGTCCGTTA CAACAGTCTGCCGGGCTTTTTCAATCGCAAAGCGGCCGAACAA AGCGCGGCGGAAGTGGCACTGCGTGAACTGGCTAAAAGCTCT GAACTGAGTCAGTCCGTCTCACAACCGGTGCACGAAACCGGC CTGGTAAAAAACCTGCTGCAAGAATATGCGCAAAAAATGAATTA TGCCATTCCGCTGTACCAGTCGCAAAAAGTGGAAACCCTGGGT CGCGTTACGCAGTTCACCAGCACGGTTGAAATTGGCGGTATCA AATACACCGGCGCGGCCACCCGTACGAAAAAAGACGCGGAAA TTTCCGCCGGTCGTACGGCACTGCTGGCGATCCAGTCATGA MTSTDVSSGVSNSYVFKSRLQEYAQKYCLPTPVYEIVKEGPSHKS LFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSSELSQS VSQPVHETGLVKNLLQEYAQKMNYAIPLYQSQKVETLGRVTQFTS TVEIGGIKYTGAATRTKKDAEISAGRTALLAIQS
Nombre: D1-D2_NoCys_S105V_T155C N° residuos: 170 Peso molecular: 18676.1 Generación del clonado: a partir de gen sintético Proteína inestable	ATGACCTCTACGGATGTGTCTAGTGGTGTCTCAAATAGCTACGT GTTCAAAAGCCGCCTGCAAGAATACGCTCAGAAATACAAACTG CCGACCCCGGTGTATGAAATTGTTAAAGAAGGCCCGTCGCATA AAAGCCTGTTTCAGTCTACGGTTATCCTGGATGGCGTCCGTTA CAACAGTCTGCCGGGCTTTTTCAATCGCAAAGCGGCCGAACAA AGCGCGGCGGAAGTGGCACTGCGTGAACTGGCTAAAAGCTCT GAACTGAGTCAGTCCGTCTCACAACCGGTGCACGAAACCGGC CTGGTAAAAAACCTGCTGCAAGAATATGCGCAAAAAATGAATTA TGCCATTCCGCTGTACCAGTCGCAAAAAGTGGAAACCCTGGGT CGCGTTACGCAGTTCACCAGCACGGTTGAAATTGGCGGTATCA AATACACCGGCGCGGCCACCCGTTGCAAAAAAGACGCGGAAA TTTCCGCCGGTCGTACGGCACTGCTGGCTATCCAGTCATAAGG A MTSTDVSSGVSNSYVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKS LFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSSELSQS VSQPVHETGLVKNLLQEYAQKMNYAIPLYQSQKVETLGRVTQFTS TVEIGGIKYTGAATRCKKDAEISAGRTALLAIQS
Nombre: L-D1-D2-L N° residuos: 193	**CCATGG**GCAGCAGCCATCATCATCATCATCACAGCAGCGGCG AAAACCTGTATTTTCAGGGCTATATTGATACCAACAACGATGGC

Peso molecular: 21535.26 Generación del clonado: corte y digestión a partir del gen sintético Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV. Aplicación: estudios por PELDOR y PRE Kd: 8.7 uM	TGGATTGAAGGCGATGAACTG**CATATG**GTTTTCAAAAGTCGGT TGCAGGAGTATGCTCAGAAGTACAAGCTCCCAACGCCTGTTTA TGAGATCGTTAAAGAAGGCCCTTCACACAAATCTTTATTTCAAT CGACTGTGATACTGGATGGTGTCAGATATAATTCTTTGCCTGGA TTCTTCAATCGTAAGGCTGCAGAGCAATCAGCTGCCGAGGTTG CTCTCCGGGAATTAGCAAAATCCAGT**GAGCTC**AGCCAATGTGT TTCACAACCTGTTCACGAAACGGGATTATGCAAGAACCTACTTC AAGAATACGCTCAAAAGATGAATTACGCGATTCCATTGTATCAG TGCCAGAAGGTCGAAACTCTTGGGAGAGTTACACAATTCACAT GTACTGTAGAGATTGGAGGCATAAAGTACACAGGAGCTGCAAC AAGAACTAAAAAAGATGCTGAGATTAGCGCTGGGAGAACTGCT CTTTTAGCGATCCAGTCA**GTCGAC**TATATTGATACCAACAACGA TGGCTGGATTGAAGGCGATGAACTG**GTCGAC**TGA GYIDTNNDGWIEGDELHMVFKSRLQEYAQKYKLPTPVYEIVKEGP SHKSLFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSS ELSQCVSQPVHETGLCKNLLQEYAQKMNYAIPLYQCQKVETLGRV TQFTCTVEIGGIKYTGAATRTKKDAEISAGRTALLAIQSVDYIDTNN DGWIEGDELVD
Nombre: L-D1-L-D2 N° residuos: 193 Peso molecular: 21563.31 Generación del clonado: corte y digestión a partir del gen sintético Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV. Aplicación: estudios por PELDOR y PRE Kd: 5.55 uM	**CCATGG**GCAGCAGCCATCATCATCATCATCACAGCAGCGGCG AAAACCTGTATTTTCAGGGCTATATTGATACCAACAACGATGGC TGGATTGAAGGCGATGAACTG**CATATG**GTTTTCAAAAGTCGGT TGCAGGAGTATGCTCAGAAGTACAAGCTCCCAACGCCTGTTTA TGAGATCGTTAAAGAAGGCCCTTCACACAAATCTTTATTTCAAT CGACTGTGATACTGGATGGTGTCAGATATAATTCTTTGCCTGGA TTCTTCAATCGTAAGGCTGCAGAGCAATCAGCTGCCGAGGTTG CTCTCCGGGAATTAGCAAAATCCAGT**GAGCTC**TATATTGATACC AACAACGATGGCTGGATTGAAGGCGATGAACTG**GAGCTC**AGC CAATGTGTTTCACAACCTGTTCACGAAACGGGATTATGCAAGAA CCTACTTCAAGAATACGCTCAAAAGATGAATTACGCGATTCCAT TGTATCAGTGCCAGAAGGTCGAAACTCTTGGGAGAGTTACACA ATTCACATGTACTGTAGAGATTGGAGGCATAAAGTACACAGGA GCTGCAACAAGAACTAAAAAAGATGCTGAGATTAGCGCTGGGA GAACTGCTCTTTTAGCGATCCAGTCA**GTCGAC**TGA GYIDTNNDGWIEGDELHMVFKSRLQEYAQKYKLPTPVYEIVKEGP SHKSLFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSS ELYIDTNNDGWIEGDELELSQCVSQPVHETGLCKNLLQEYAQKMN YAIPLYQCQKVETLGRVTQFTCTVEIGGIKYTGAATRTKKDAEISAG RTALLAIQSVD
Nombre: D1-L-D2-L N° residuos: 195 Peso molecular: 21777.53 Generación del clonado: corte y digestión a partir del gen sintético Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV. Aplicación: estudios por PELDOR y PRE Kd: 6.62 uM	**CCATGG**GCAGCAGCCATCATCATCATCATCACAGCAGCGGCG AAAACCTGTATTTTCAGGGC**CATATG**GTTTTCAAAAGTCGGTTG CAGGAGTATGCTCAGAAGTACAAGCTCCCAACGCCTGTTTATG AGATCGTTAAAGAAGGCCCTTCACACAAATCTTTATTTCAATCG ACTGTGATACTGGATGGTGTCAGATATAATTCTTTGCCTGGATT CTTCAATCGTAAGGCTGCAGAGCAATCAGCTGCCGAGGTTGCT CTCCGGGAATTAGCAAAATCCAGT**GAGCTC**TATATTGATACCAA CAACGATGGCTGGATTGAAGGCGATGAACTG**GAGCTC**AGCCA ATGTGTTTCACAACCTGTTCACGAAACGGGATTATGCAAGAACC TACTTCAAGAATACGCTCAAAAGATGAATTACGCGATTCCATTG TATCAGTGCCAGAAGGTCGAAACTCTTGGGAGAGTTACACAAT TCACATGTACTGTAGAGATTGGAGGCATAAAGTACACAGGAGC TGCAACAAGAACTAAAAAAGATGCTGAGATTAGCGCTGGGAGA ACTGCTCTTTTAGCGATCCAGTCA**GTCGAC**TATATTGATACCAA CAACGATGGCTGGATTGAAGGCGATGAACTG**GTCGAC**TGA GHMVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKSLFQSTVILDGV RYNSLPGFFNRKAAEQSAAEVALRELAKSSELYIDTNNDGWIEGD ELELSQCVSQPVHETGLCKNLLQEYAQKMNYAIPLYQCQKVETLG RVTQFTCTVEIGGIKYTGAATRTKKDAEISAGRTALLAIQSVDYIDT NNDGWIEGDELVD
Nombre: D1-L-D2 N° residuos: 178	**CCATGG**GCAGCAGCCATCATCATCATCATCACAGCAGCGGCG AAAACCTGTATTTTCAGGGC**CATATG**GTTTTCAAAAGTCGGTTG

Peso molecular: 19827.53 Generación del clonado: corte y digestión a partir del gen sintético Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV. Aplicación: estudios por PELDOR y PRE	CAGGAGTATGCTCAGAAGTACAAGCTCCCAACGCCTGTTTATG AGATCGTTAAAGAAGGCCCTTCACACAAATCTTTATTTCAATCG ACTGTGATACTGGATGGTGTCAGATATAATTCTTTGCCTGGATT CTTCAATCGTAAGGCTGCAGAGCAATCAGCTGCCGAGGTTGCT CTCCGGGAATTAGCAAAATCCAGT**GAGCTC**TATATTGATACCAA CAACGATGGCTGGATTGAAGGCGATGAACTG**GAGCTC**AGCCA ATGTGTTTCACAACCTGTTCACGAAACGGGATTATGCAAGAACC TACTTCAAGAATACGCTCAAAAGATGAATTACGCGATTCCATTG TATCAGTGCCAGAAGGTCGAAACTCTTGGGAGAGTTACACAAT TCACATGTACTGTAGAGATTGGAGGCATAAAGTACACAGGAGC TGCAACAAGAACTAAAAAAGATGCTGAGATTAGCGCTGGGAGA ACTGCTCTTTTAGCGATCCAGTCA**GTCGAC**TGA GHMVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKSLFQSTVILDGV RYNSLPGFFNRKAAEQSAAEVALRELAKSSELYIDTNNDGWIEGD ELELSQCVSQPVHETGLCKNLLQEYAQKMNYAIPLYQCQKVETLG RVTQFTCTVEIGGIKYTGAATRTKKDAEISAGRTALLAIQSVD
Nombre: D1-D2-L N° residuos: 178 Peso molecular: 19799.48 Generación del clonado: corte y digestión a partir del gen sintético Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV. Aplicación: estudios por PELDOR y PRE Kd: 0.77 uM	**CCATGG**GCAGCAGCCATCATCATCATCATCACAGCAGCGGCG AAAACCTGTATTTTCAGGGC**CATATG**GTTTTCAAAAGTCGGTTG CAGGAGTATGCTCAGAAGTACAAGCTCCCAACGCCTGTTTATG AGATCGTTAAAGAAGGCCCTTCACACAAATCTTTATTTCAATCG ACTGTGATACTGGATGGTGTCAGATATAATTCTTTGCCTGGATT CTTCAATCGTAAGGCTGCAGAGCAATCAGCTGCCGAGGTTGCT CTCCGGGAATTAGCAAAATCCAGT**GAGCTC**AGCCAATGTGTTT CACAACCTGTTCACGAAACGGGATTATGCAAGAACCTACTTCAA GAATACGCTCAAAAGATGAATTACGCGATTCCATTGTATCAGTG CCAGAAGGTCGAAACTCTTGGGAGAGTTACACAATTCACATGT ACTGTAGAGATTGGAGGCATAAAGTACACAGGAGCTGCAACAA GAACTAAAAAAGATGCTGAGATTAGCGCTGGGAGAACTGCTCT TTTAGCGATCCAGTCA**GTCGAC**TATATTGATACCAACAACGATG GCTGGATTGAAGGCGATGAACTG**GTCGAC**TGA GHMVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKSLFQSTVILDGV RYNSLPGFFNRKAAEQSAAEVALRELAKSSELSQCVSQPVHETGL CKNLLQEYAQKMNYAIPLYQCQKVETLGRVTQFTCTVEIGGIKYTG AATRTKKDAEISAGRTALLAIQSVDYIDTNNDGWIEGDELVD
Nombre: L-D1-D2 N° residuos: 176 Peso molecular: 19585.26 Generación del clonado: corte y digestión a partir del gen sintético Producción de la proteína: expresión, purificación por columna de Níquel y corte con TEV. Aplicación: estudios por PELDOR y PRE Kd: 3.55 uM	**CCATGG**GCAGCAGCCATCATCATCATCATCACAGCAGCGGCG AAAACCTGTATTTTCAGGGCTATATTGATACCAACAACGATGGC TGGATTGAAGGCGATGAACTG**CATATG**GTTTTCAAAAGTCGGT TGCAGGAGTATGCTCAGAAGTACAAGCTCCCAACGCCTGTTTA TGAGATCGTTAAAGAAGGCCCTTCACACAAATCTTTATTTCAAT CGACTGTGATACTGGATGGTGTCAGATATAATTCTTTGCCTGGA TTCTTCAATCGTAAGGCTGCAGAGCAATCAGCTGCCGAGGTTG CTCTCCGGGAATTAGCAAAATCCAGTGAGCTCAGCCAATGTGT TTCACAACCTGTTCACGAAACGGGATTATGCAAGAACCTACTTC AAGAATACGCTCAAAAGATGAATTACGCGATTCCATTGTATCAG TGCCAGAAGGTCGAAACTCTTGGGAGAGTTACACAATTCACAT GTACTGTAGAGATTGGAGGCATAAAGTACACAGGAGCTGCAAC AAGAACTAAAAAAGATGCTGAGATTAGCGCTGGGAGAACTGCT CTTTTAGCGATCCAGTCAGTCGACTGA GYIDTNNDGWIEGDELHMVFKSRLQEYAQKYKLPTPVYEIVKEGP SHKSLFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSS ELSQCVSQPVHETGLCKNLLQEYAQKMNYAIPLYQCQKVETLGRV TQFTCTVEIGGIKYTGAATRTKKDAEISAGRTALLAIQSVD

7.3. Códigos

ANÁLISIS DE LINKERS

```
In [1]:
import re
import numpy as np
import matplotlib.pyplot as plt
from Bio.SeqIO import *
from Bio import Phylo
import StringIO
import weblogolib as w
from collections import defaultdict
```

Análisis para separar las 50188 proteínas en base a su taxonomía (determinar cuáles de todas son plantas o animales)

```
In [2]:
#Leyendo el archivo con la descripción de taxonomía
contenido_taxo = ""
file_taxo = open("uniprot-drbm-50188-taxo.tab", "r")#Archivo que fue descargado desde Uniprot
contenido_taxo = file_taxo.read()
file_taxo.close()

#El contenido del archivo se guarda dentro del diccionario "nombre_contenidoArchivoTaxo", con los
nombres de Uniprot y su taxonomía
nombre_contenidoArchivoTaxo = {}
for i in (contenido_taxo.split("\n")[1:-1]): #Se elimina el primer item (titulos de las columnas) y el
ultimo (vacio)
    nombre_contenidoArchivoTaxo[i.split("\t")[1]] = i.split("\t")[2]

#Lista con todos los nombres de las proteínas
lista_nombres = list(nombre_contenidoArchivoTaxo.keys())
```

```
In [3]:
###Visualización de la taxonomia de HYL1###
print "Taxonomí de HYL11 (DRB1_ARATH): ", nombre_contenidoArchivoTaxo["DRB1_ARATH"]
```

```
Taxonomía de HYL11 (DRB1_ARATH):  cellular organisms, Eukaryota, Viridiplantae, Streptophyta,
Streptophytina, Embryophyta, Tracheophyta, Euphyllophyta, Spermatophyta, Magnoliophyta,
Mesangiospermae, eudicotyledons, Gunneridae, Pentapetalae, rosids, malvids, Brassicales, Brassicaceae,
Camelineae, Arabidopsis, Arabidopsis thaliana (Mouse-ear cress)
```

```
In [4]:
#Separando proteínas que no son organismos celulares
noSonCellular=[x for x in lista_nombres
            if nombre_contenidoArchivoTaxo[x].split(",")[0] !=
            nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[0]
            ]
print "No son cellular organisms", len(noSonCellular)

sonCellular=[x for x in lista_nombres
            if nombre_contenidoArchivoTaxo[x].split(",")[0] ==
            nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[0]
            ]
print "Son cellular organisms", len(sonCellular)
```

```
No son cellular organisms 925
Son cellular organisms 49263
```

```
In [5]:
#Separando proteínas de especies que no son organismos eucariotas
noSonEukaryota=[x for x in sonCellular
            if nombre_contenidoArchivoTaxo[x].split(",")[1] !=
            nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[1]
            ]
print "No son Eukaryota", len(noSonEukaryota)

sonEukaryota=[x for x in sonCellular
            if nombre_contenidoArchivoTaxo[x].split(",")[1] ==
            nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[1]
            ]
print "Son Eukaryota", len(sonEukaryota)
No son Eukaryota 33235
```

```
Son Eukaryota 16028

In [6]:
#Separando proteínas de especies de plantas y animales
noSonViridiplantae=[x for x in sonEukaryota
            if nombre_contenidoArchivoTaxo[x].split(",")[2] !=
            nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[2]
            ]
print "No son plantas", len(noSonViridiplantae)

sonMetazoa=[x for x in noSonViridiplantae
            if nombre_contenidoArchivoTaxo[x].split(",")[3] ==
            " Metazoa"
            ]
print "Son animales (Metazoa)", len(sonMetazoa)

sonViridiplantae=[x for x in sonEukaryota
            if nombre_contenidoArchivoTaxo[x].split(",")[2] ==
            nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[2]
            ]
print "Son plantas (Viridiplantae)", len(sonViridiplantae)
```

```
No son plantas 13187
Son animales (Metazoa) 9254
Son plantas (Viridiplantae) 2841
```

Separar según cantidad de dominios DRBM en proteínas de especies de plantas y de animales.

```
In [7]:
#Leyendo el archivo con la descripción de dominios
contenido_dominios = ""
file_dominios = open("uniprot-drbm-50188.tab", "r") #Archivo que fue descargado desde Uniprot
contenido_dominios = file_dominios.read()
file_dominios.close()

# El contenido del archivo se guarda dentro del diccionario "nombre_contenidoArchivoDominios", con los
nombres de Uniprot y la descripción de sus dominios
nombre_contenidoArchivoDominios = {}
for i in (contenido_dominios.split("\n")[1:-1]): #Se elimina el primer item (titulos de las columnas)
y el ultimo (vacio)
    nombre_contenidoArchivoDominios[i.split("\t")[1]] = i.split("\t")[2]
```

```
In [8]:
###Calculo de distancias entre dsRBDs (DRBM)###

nombresreinos = ["plantas", "animales"] #primero analiza todas las proteínas de especies de plantas y
luego de especies de animales
for n,reino in enumerate([sonViridiplantae, sonMetazoa]):
    pos = 0
    cantidadDRBM = 0
    nombre_distanciaCero = {}
    nombre_distanciaUno = {}
    nombre_distanciaDos = {}
    nombre_distanciaTres = {}
    nombre_distanciaCuatro = {}
    nombre_distanciaCinco = {}
    nombre_distanciaSeis = {}
    nombre_dominioIntermedio = []
    distanciaDos = []
    distanciaTres = []
    distanciaCuatro = []
    distanciaCinco = []
    distanciaSeis = []
    distanciaTodas = []

    for key in reino:
        value = nombre_contenidoArchivoDominios[key]#Lee la información de dominios para esa proteína
        coincidencias = ""
        coincidencias = re.findall("DOMAIN.[0-9]+\s[0-9]+.DRBM", value, flags=0) #Busca la
coincidencia de dominios DRBM.
        #Si hay dominios se analiza la posiciones de los mismos y se registra la distancia entre ellos
if len(coincidencias) == 0:
            nombre_distanciaCero[key] = np.nan
```

```
        if len(coincidencias) == 1:
            cantidadDRBM +=1
            nombre_distanciaUno[key] = 0

        if len(coincidencias) == 2:
            cantidadDRBM +=2
            final_uno = 0
            ppio_dos = 0
            dif_dos_uno = 0
            final_uno = coincidencias[0].split(" ")[2]
            ppio_dos = coincidencias[1].split(" ")[1]
            #Para asegurarse que no hay un dominio de por medio se usan índices y se busca el string
"DOMAIN"
            index_final_uno = np.nan
            index_ppio_dos = np.nan
            index_final_uno =  value.index(coincidencias[0]) +11
            index_ppio_dos =  value.index(coincidencias[1])
            if value.find("DOMAIN", index_final_uno, (index_ppio_dos-7)) == -1: #-1 es que no está
                dif_dos_uno = int(ppio_dos) - int(final_uno)
                nombre_distanciaDos[key] = int(dif_dos_uno)
                distanciaDos.append(dif_dos_uno)
                distanciaTodas.append(dif_dos_uno)
            else:
                nombre_dominioIntermedio.append(key)

        if len(coincidencias) == 3:
            cantidadDRBM +=3
            nombre_distanciaTres[key]=[]
            final_uno = 0
            ppio_dos = 0
            final_dos = 0
            ppio_tres = 0
            dif_dos_uno = 0
            dis_tres_dos = 0
            final_uno = coincidencias[0].split(" ")[2]
            ppio_dos = coincidencias[1].split(" ")[1]
            index_final_uno = np.nan
            index_ppio_dos = np.nan
            index_final_uno =  value.index(coincidencias[0]) +11
            index_ppio_dos =  value.index(coincidencias[1])
            if value.find("DOMAIN", index_final_uno, (index_ppio_dos-7)) == -1:
                dif_dos_uno = int(ppio_dos) - int(final_uno)
                nombre_distanciaTres[key].append(int(dif_dos_uno))
                distanciaTres.append(dif_dos_uno)
                distanciaTodas.append(dif_dos_uno)
            final_dos = coincidencias[1].split(" ")[2]
            ppio_tres = coincidencias[2].split(" ")[1]
            index_final_dos = np.nan
            index_ppio_tres = np.nan
            index_final_dos =  value.index(coincidencias[1]) +11
            index_ppio_tres =  value.index(coincidencias[2])
            if value.find("DOMAIN", index_final_dos, index_ppio_tres) == -1:
                dif_tres_dos = int(ppio_tres) - int(final_dos)
                nombre_distanciaTres[key].append(int(dif_tres_dos))
                distanciaTres.append(dif_tres_dos)
                distanciaTodas.append(dif_tres_dos)

        if len(coincidencias) == 4:
            cantidadDRBM +=4
            nombre_distanciaCuatro[key]=[]
            final_uno = 0
            ppio_dos = 0
            final_dos = 0
            ppio_tres = 0
            final_tres = 0
            ppio_cuatro = 0
            dif_dos_uno = 0
            dif_tres_dos = 0
            dif_cuatro_tres = 0
            final_uno = coincidencias[0].split(" ")[2]
            ppio_dos = coincidencias[1].split(" ")[1]
            index_final_uno = np.nan
            index_ppio_dos = np.nan
```

```
            index_final_uno =  value.index(coincidencias[0]) +11
            index_ppio_dos =  value.index(coincidencias[1])
            if value.find("DOMAIN", index_final_uno, (index_ppio_dos-7)) == -1:
                dif_dos_uno = int(ppio_dos) - int(final_uno)
                nombre_distanciaCuatro[key].append(int(dif_dos_uno))
                distanciaCuatro.append(dif_dos_uno)
                distanciaTodas.append(dif_dos_uno)
            final_dos = coincidencias[1].split(" ")[2]
            ppio_tres = coincidencias[2].split(" ")[1]
            index_final_dos = np.nan
            index_ppio_tres = np.nan
            index_final_dos =  value.index(coincidencias[1]) +11
            index_ppio_tres =  value.index(coincidencias[2])
            if value.find("DOMAIN", index_final_dos, (index_ppio_tres-7)) == -1:
                dif_tres_dos = int(ppio_tres) - int(final_dos)
                nombre_distanciaCuatro[key].append(int(dif_tres_dos))
                distanciaCuatro.append(dif_tres_dos)
                distanciaTodas.append(dif_tres_dos)
            final_tres = coincidencias[2].split(" ")[2]
            ppio_cuatro = coincidencias[3].split(" ")[1]
            index_final_tres = np.nan
            index_ppio_cuatro = np.nan
            index_final_tres =  value.index(coincidencias[2]) +11
            index_ppio_cuatro =  value.index(coincidencias[3])
            if value.find("DOMAIN", index_final_tres, (index_ppio_cuatro-7)) == -1:
                dif_cuatro_tres = int(ppio_cuatro) - int(final_tres)
                nombre_distanciaCuatro[key].append(int(dif_cuatro_tres))
                distanciaCuatro.append(dif_cuatro_tres)
                distanciaTodas.append(dif_cuatro_tres)

        if len(coincidencias) == 5:
            cantidadDRBM +=5
            nombre_distanciaCinco[key]=[]
            final_uno = 0
            ppio_dos = 0
            final_dos = 0
            ppio_tres = 0
            final_tres = 0
            ppio_cuatro = 0
            final_cuatro = 0
            ppio_cinco = 0
            dif_dos_uno = 0
            dif_tres_dos = 0
            dif_cuatro_tres = 0
            dif_cinco_cuatro = 0
            final_uno = coincidencias[0].split(" ")[2]
            ppio_dos = coincidencias[1].split(" ")[1]
            index_final_uno = np.nan
            index_ppio_dos = np.nan
            index_final_uno =  value.index(coincidencias[0]) +11
            index_ppio_dos =  value.index(coincidencias[1])
            if value.find("DOMAIN", index_final_uno, (index_ppio_dos-7)) == -1:
                dif_dos_uno = int(ppio_dos) - int(final_uno)
                nombre_distanciaCinco[key].append(int(dif_dos_uno))
                distanciaCinco.append(dif_dos_uno)
                distanciaTodas.append(dif_dos_uno)
            final_dos = coincidencias[1].split(" ")[2]
            ppio_tres = coincidencias[2].split(" ")[1]
            index_final_dos = np.nan
            index_ppio_tres = np.nan
            index_final_dos =  value.index(coincidencias[1]) +11
            index_ppio_tres =  value.index(coincidencias[2])
            if value.find("DOMAIN", index_final_dos, (index_ppio_tres-7)) == -1:
                dif_tres_dos = int(ppio_tres) - int(final_dos)
                nombre_distanciaCinco[key].append(int(dif_tres_dos))
                distanciaCinco.append(dif_tres_dos)
                distanciaTodas.append(dif_tres_dos)
            final_tres = coincidencias[2].split(" ")[2]
            ppio_cuatro = coincidencias[3].split(" ")[1]
            index_final_tres = np.nan
            index_ppio_cuatro = np.nan
            index_final_tres =  value.index(coincidencias[2]) +11
            index_ppio_cuatro =  value.index(coincidencias[3])
            if value.find("DOMAIN", index_final_tres, (index_ppio_cuatro-7)) == -1:
```

```
                dif_cuatro_tres = int(ppio_cuatro) - int(final_tres)
                nombre_distanciaCinco[key].append(int(dif_cuatro_tres))
                distanciaCinco.append(dif_cuatro_tres)
                distanciaTodas.append(dif_cuatro_tres)
            final_cuatro = coincidencias[3].split(" ")[2]
            ppio_cinco = coincidencias[4].split(" ")[1]
            index_final_cuatro = np.nan
            index_ppio_cinco = np.nan
            index_final_cuatro =  value.index(coincidencias[3]) +11
            index_ppio_cinco =  value.index(coincidencias[4])
            if value.find("DOMAIN", index_final_cuatro, (index_ppio_cinco-7)) == -1:
                dif_cinco_cuatro = int(ppio_cinco) - int(final_cuatro)
                nombre_distanciaCinco[key].append(int(dif_cinco_cuatro))
                distanciaCinco.append(dif_cinco_cuatro)
                distanciaTodas.append(dif_cinco_cuatro)

        if len(coincidencias) == 6:
            cantidadDRBM +=6
            nombre_distanciaSeis[key]=[]
            final_uno = 0
            ppio_dos = 0
            final_dos = 0
            ppio_tres = 0
            final_tres = 0
            ppio_cuatro = 0
            final_cuatro = 0
            ppio_cinco = 0
            final_cinco = 0
            ppio_seis = 0
            dif_dos_uno = 0
            dif_tres_dos = 0
            dif_cuatro_tres = 0
            dif_cinco_cuatro = 0
            dif_seis_cinco = 0
            ppio_dos = coincidencias[1].split(" ")[1]
            final_uno = coincidencias[0].split(" ")[2]
            index_final_uno = np.nan
            index_ppio_dos = np.nan
            index_final_uno =  value.index(coincidencias[0]) +11
            index_ppio_dos =  value.index(coincidencias[1])
            if value.find("DOMAIN", index_final_uno, (index_ppio_dos-7)) == -1:
                dif_dos_uno = int(ppio_dos) - int(final_uno)
                nombre_distanciaSeis[key].append(int(dif_dos_uno))
                distanciaSeis.append(dif_dos_uno)
                distanciaTodas.append(dif_dos_uno)
            final_dos = coincidencias[1].split(" ")[2]
            ppio_tres = coincidencias[2].split(" ")[1]
            index_final_dos = np.nan
            index_ppio_tres = np.nan
            index_final_dos =  value.index(coincidencias[1]) +11
            index_ppio_tres =  value.index(coincidencias[2])
            if value.find("DOMAIN", index_final_dos, (index_ppio_tres-7)) == -1:
                dif_tres_dos = int(ppio_tres) - int(final_dos)
                nombre_distanciaSeis[key].append(int(dif_tres_dos))
                distanciaSeis.append(dif_tres_dos)
                distanciaTodas.append(dif_tres_dos)
            final_tres = coincidencias[2].split(" ")[2]
            ppio_cuatro = coincidencias[3].split(" ")[1]
            index_final_tres = np.nan
            index_ppio_cuatro = np.nan
            index_final_tres =  value.index(coincidencias[2]) +11
            index_ppio_cuatro =  value.index(coincidencias[3])
            if value.find("DOMAIN", index_final_tres, (index_ppio_cuatro-7)) == -1:
                dif_cuatro_tres = int(ppio_cuatro) - int(final_tres)
                nombre_distanciaSeis[key].append(int(dif_cuatro_tres))
                distanciaSeis.append(dif_cuatro_tres)
                distanciaTodas.append(dif_cuatro_tres)
            final_cuatro = coincidencias[3].split(" ")[2]
            ppio_cinco = coincidencias[4].split(" ")[1]
            index_final_cuatro = np.nan
            index_ppio_cinco = np.nan
            index_final_cuatro =  value.index(coincidencias[3]) +11
            index_ppio_cinco =  value.index(coincidencias[4])
            if value.find("DOMAIN", index_final_cuatro, (index_ppio_cinco-7)) == -1:
```

```
                dif_cinco_cuatro = int(ppio_cinco) - int(final_cuatro)
                nombre_distanciaSeis[key].append(int(dif_cinco_cuatro))
                distanciaSeis.append(dif_cinco_cuatro)
                distanciaTodas.append(dif_cinco_cuatro)
            final_cinco = coincidencias[4].split(" ")[2]
            ppio_seis = coincidencias[5].split(" ")[1]
            index_final_cinco = np.nan
            index_ppio_seis = np.nan
            index_final_cinco =  value.index(coincidencias[4]) +11
            index_ppio_seis =  value.index(coincidencias[5])
            if value.find("DOMAIN", index_final_cinco, (index_ppio_seis-7)) == -1:
                dif_seis_cinco = int(ppio_seis) - int(final_cinco)
                nombre_distanciaSeis[key].append(int(dif_seis_cinco))
                distanciaSeis.append(dif_seis_cinco)
                distanciaTodas.append(dif_seis_cinco)

    nombre_distanciaTodas = {} #Todas las distancias se guardan en un diccionario
    nrosletras = ["Dos", "Tres", "Cuatro", "Cinco", "Seis"]
    for nro in nrosletras:
        for key,value in globals()["nombre_distancia%s"%nro].iteritems():
            nombre_distanciaTodas[key] = value

    #Contando las cantidades que quedaron
    numeros = ["Cero", "Uno", "Dos", "Tres", "Cuatro", "Cinco", "Seis", "Todas"]
    print nombresreinos[n]
    for nro in numeros:
        print "Cantidad de DRBM", nro, "; cantidad de proteinas", len(globals()["nombre_distancia%s"
%nro])

    #Identificando las variables según si son plantas o animales
    nombres = ["Dos","Tres","Cuatro","Cinco","Seis","Todas"]
    for nom in nombres:
        globals()["%s_distancia%s"%(nombresreinos[n],nom)] = []
        for i in globals()["distancia%s"%nom]:
            globals()["%s_distancia%s"%(nombresreinos[n],nom)].append(i)
        globals()["%s_nombre_distancia%s"%(nombresreinos[n],nom)] = {}
        for k,v in globals()["nombre_distancia%s"%nom].iteritems():
            globals()["%s_nombre_distancia%s"%(nombresreinos[n],nom)][k]=v
```

```
plantas
Cantidad de DRBM Cero ; cantidad de proteinas 525
Cantidad de DRBM Uno ; cantidad de proteinas 1058
Cantidad de DRBM Dos ; cantidad de proteinas 1152
Cantidad de DRBM Tres ; cantidad de proteinas 101
Cantidad de DRBM Cuatro ; cantidad de proteinas 3
Cantidad de DRBM Cinco ; cantidad de proteinas 0
Cantidad de DRBM Seis ; cantidad de proteinas 0
Cantidad de DRBM Todas ; cantidad de proteinas 1256
animales
Cantidad de DRBM Cero ; cantidad de proteinas 2414
Cantidad de DRBM Uno ; cantidad de proteinas 3391
Cantidad de DRBM Dos ; cantidad de proteinas 2137
Cantidad de DRBM Tres ; cantidad de proteinas 831
Cantidad de DRBM Cuatro ; cantidad de proteinas 386
Cantidad de DRBM Cinco ; cantidad de proteinas 92
Cantidad de DRBM Seis ; cantidad de proteinas 1
Cantidad de DRBM Todas ; cantidad de proteinas 3447
```

Se observa conservación para proteínas de plantas con dos DRBMs, se continúa trabajando con ese grupo (1152 proteínas). Curado: eliminar proteínas que presenten otros dominios, eliminar proteínas iguales o muy similares.

```
In [9]:
#Seleccionando las que solamente tienen dominios DRBM
plantas = [x for x in plantas_nombre_distanciaDos.keys()]#Proteínas de plantas con dos DRBMs
plantas_soloDRBM = []
for nom in plantas:
    if nombre_contenidoArchivoDominios[nom].count("DOMAIN") == 2: #Analisis de la cantidad total de
dominios, si es mayor a dos es porque hay otros dominios además de los dos DRBMs encontrados
anteriormente
        plantas_soloDRBM.append(nom)
print "De las 1152 proteínas de especies de plantas con dos DRBM, no tienen otro tipo de DRBM",
len(plantas_soloDRBM)
```

```
De las 1152 proteínas de especies de plantas con dos DRBM, no tienen otro tipo de DRBM 659
```

```
In [10]:
#Leyendo el archivo con las secuencias completas
nombre_secuencia = {}
for rec in parse('uniprot-drbm-50188.fasta', "fasta"): #Archivo que fue descargado desde Uniprot
    nombre_secuencia[rec.name.split("|")[2].split()[0]] = str(rec.seq)
```

```
In [11]:
#Extrayendo secuencias de linkers y dominios para proteínas de especies de plantas con 2 DRBM
nombre_linker = {}
nombre_dsRBD1 = {}
nombre_dsRBD2 = {}
for nro in range(200): #Se analizan linkers de hasta 200 residuos de largo
    #La distancia del linker queda grabada en los nombres de las variables
    globals()["nombre_linker_%s" %nro] = {}
    globals()["nombre_dsRBD1_%s"%nro] = {}
    globals()["nombre_dsRBD2_%s"%nro] = {}
    for nombre,dist in plantas_nombre_distanciaDos.iteritems():
        if nombre in plantas_soloDRBM: #Continúa desde el curado anterior
            if dist == nro:
                coincidencias = re.findall("DOMAIN.[0-9]+\s[0-9]+.DRBM",
nombre_contenidoArchivoDominios[nombre], flags=0)
                ppio_uno = np.nan
                final_uno = np.nan
                ppio_dos = np.nan
                final_dos = np.nan
                ppio_uno = coincidencias[0].split(" ")[1]
                final_uno = coincidencias[0].split(" ")[2]
                ppio_dos = coincidencias[1].split(" ")[1]
                final_dos = coincidencias[1].split(" ")[2]
                linker = ""
                dsRBD1 = ""
                dsRBD2 = ""
                dsRBD1 = nombre_secuencia[nombre][int(ppio_uno):int(final_uno)] #Secuencia dsRBD1
                linker = nombre_secuencia[nombre][int(final_uno):int(ppio_dos)] #Secuencia linker
                dsRBD2 = nombre_secuencia[nombre][int(ppio_dos):int(final_dos)] #Secuencia dsRBD2
                globals()["nombre_dsRBD1_%s"%nro][nombre]= dsRBD1
                nombre_dsRBD1[nombre] = dsRBD1
                globals()["nombre_linker_%s" %nro][nombre] = linker
                nombre_linker[nombre] = linker
                globals()["nombre_dsRBD2_%s"%nro][nombre]= dsRBD2
                nombre_dsRBD2[nombre] = dsRBD2
print "Se extrajeron secuencias de linkers para", len(nombre_linker)
print "Se extrajeron secuencias del primer dsRBD para", len(nombre_dsRBD1)
print "Se extrajeron secuencias del segundo dsRBD para", len(nombre_dsRBD2)
```

```
Se extrajeron secuencias de linkers para 641
Se extrajeron secuencias del primer dsRBD para 641
Se extrajeron secuencias del segundo dsRBD para 641
```

```
In [12]:
#Selección de secuencias únicas. Compara el organismo al que pertenece (que los infiere a partir de
parte de su nombre), y su secuencia. En caso que haya coincidencias se queda con solo una.
regiones = ["dsRBD1", "linker", "dsRBD2"]
for region in regiones:
    globals()["nombre_%sUnico"%region] = {}
    for n in range(200):
        globals()["orga%s_nombre"%region] = {}
        globals()["orga%s_nombre"%region] =[["%s-%s" %(x[0].split("_")[1],x[1]),x[0]] for x in
globals()["nombre_%s_%s"%(region,n)].iteritems()]
        globals()["orga%sUnicos"%region]=set([x[0] for x in globals()["orga%s_nombre"%region]])
        globals()["orga%sUnicos_nombre"%region] = {}
        globals()["orga%sUnicos_nombre"%region] = dict([[x,[]] for x in
globals()["orga%sUnicos"%region]])
        for x in globals()["orga%s_nombre"%region]:
            globals()["organ%sx"%region]=x[0]
            nombrex=x[1]
            globals()["orga%sUnicos_nombre"%region][globals()["organ%sx"%region]].append(nombrex)

    #Solo correr una vez!

        for globals()["organito%s"%region] in globals()["orga%sUnicos_nombre"%region].keys():
            if len(globals()["orga%sUnicos_nombre"%region][globals()["organito%s"%region]])>1:
                largoNombre=[]
```

```
                largoNombre=sorted([[len(x),x] for x in
globals()["orga%sUnicos_nombre"%region][globals()["organito%s"%region]]]) #lista con largo,nombre
globals()["orga%sUnicos_nombre"%region][globals()["organito%s"%region]]=largoNombre[0][1] #se
sobreescribre con el más corto
            else:
globals()["orga%sUnicos_nombre"%region][globals()["organito%s"%region]]=globals()["orga%sUnicos_nombre
"%region][globals()["organito%s"%region]][0]

        globals()["nombre_%s%sUnico"%(region,n)] = {}
        for k,v in globals()["orga%sUnicos_nombre"%region].iteritems():
            globals()["nombre_%s%sUnico"%(region,n)][v]=k.split("-")[1]
            globals()["nombre_%sUnico"%region][v]=k.split("-")[1]

    print "La cantidad de %s únicos es"%region, len(globals()["nombre_%sUnico"%region])
```

```
La cantidad de dsRBD1 únicos es 520
La cantidad de linker únicos es 479
La cantidad de dsRBD2 únicos es 524
```

```
In [13]:
#Lista de distancias con solo DRBM
plantas_distanciaDos_soloDRBM = []
for k,v in nombre_linker.iteritems():
    plantas_distanciaDos_soloDRBM.append(len(v))

#Lista de distancia de linker único
for region in regiones:
    globals()["plantas_distanciaDos_%sUnico"%region] = []
    for k,v in globals()["nombre_%sUnico"%region].iteritems():
        globals()["plantas_distanciaDos_%sUnico"%region].append(len(v))

#Extrayendo los que no son únicos
soloDRBM = [x for x in nombre_linker.keys()]
no_linkerUnico = [x for x in nombre_linker.keys() if x not in nombre_linkerUnico.keys()]
no_dsRBD1Unico = [x for x in nombre_linker.keys() if x not in nombre_dsRBD1Unico.keys()]
no_dsRBD2Unico = [x for x in nombre_linker.keys() if x not in nombre_dsRBD2Unico.keys()]
```

```
In [14]:
#Eliminar las que tienen dominios y linker repetidos
no_todos = [x for x in soloDRBM if x in no_linkerUnico and x in no_dsRBD1Unico and x in
no_dsRBD2Unico]
unicas = [x for x in nombre_linker.keys() if x not in no_todos]
print "Las proteínas con dos DRBMs, sin otros dominios, y cuyas secuencias de dominios y linkers son
únicas son", len(unicas)
plantas_distanciaDos_UnicosTodo = [] #Lista de distancias
for x in unicas:
    plantas_distanciaDos_UnicosTodo.append(len(nombre_linker[x]))
```

```
Las proteínas con dos DRBMs, sin otros dominios, y cuyas secuencias de dominios y linkers son únicas
son 540
```

```
In [15]:
#Guardar en archivos
seqsUnicas_file = open("seqsUnicas.fasta", "w") #Secuencias completas de la base curada
for x in unicas:
    seqsUnicas_file.write(">")
    seqsUnicas_file.write(x)
    seqsUnicas_file.write("\n")
    seqsUnicas_file.write(nombre_secuencia[x])
    seqsUnicas_file.write("\n")
seqsUnicas_file.close()

linkerUnicos17_file_dist = open("linkerUnicos17.txt", "w")#Secuencias de linkers de 17 residuos de la
base curada en formato txt para logos
for x in unicas:
    if x in nombre_linker_17.keys():
        linkerUnicos17_file_dist.write(nombre_linker_17[x])
        linkerUnicos17_file_dist.write("\n")
linkerUnicos17_file_dist.close()

linkerUnicos17_file = open("linkerUnicos17.fasta", "w")#Secuencia de linkers de 17 residuos de la base
curada en formato fasta
for x in unicas:
    if len(nombre_linker[x]) == 17:
        linkerUnicos17_file.write(">")
        linkerUnicos17_file.write(x)
```

```
        linkerUnicos17_file.write("\n")
        linkerUnicos17_file.write(nombre_linker_17[x])
        linkerUnicos17_file.write("\n")
linkerUnicos17_file.close()

nombre_linkerUnico = {x:nombre_linker[x] for x in unicas}#Secuencias de linkers de la base curada
linkerUnicos_file = open("linkerUnicos.fasta", "w")
for k,v in nombre_linkerUnico.iteritems():
    linkerUnicos_file.write(">")
    linkerUnicos_file.write(k)
    linkerUnicos_file.write("\n")
    linkerUnicos_file.write(v)
    linkerUnicos_file.write("\n")
linkerUnicos_file.close()
```

540 proteínas de plantas con dos DRBMs, sin otros dominios y sin repeticiones, análisis para determinar si la selección estuvo sesgada a un grupo de especies relacionadas

```
In [16]:
#Separando en grupos taxonómicos (se buscan el nivel de coincidencia con la taxonomía de HYL1)
no3 = [x for x in unicas if nombre_contenidoArchivoTaxo[x].split(",")[3] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[3]]
no4 = [x for x in unicas if x not in no3 and nombre_contenidoArchivoTaxo[x].split(",")[4] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[4]]
no5 = [x for x in unicas if x not in no3 and x not in no4 and
nombre_contenidoArchivoTaxo[x].split(",")[5] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[5]]
no6 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and
nombre_contenidoArchivoTaxo[x].split(",")[6] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[6]]
no7 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and
nombre_contenidoArchivoTaxo[x].split(",")[7] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[7]]
no8 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and nombre_contenidoArchivoTaxo[x].split(",")[8] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[8]]
no9 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and x not in no8 and nombre_contenidoArchivoTaxo[x].split(",")[9] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[9]]
no10 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and x not in no8 and x not in no9 and nombre_contenidoArchivoTaxo[x].split(",")[10] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[10]]
no11 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and x not in no8 and x not in no9 and x not in no10 and
nombre_contenidoArchivoTaxo[x].split(",")[11] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[11]]
no12 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and x not in no8 and x not in no9 and x not in no10 and x not in no11 and
nombre_contenidoArchivoTaxo[x].split(",")[12] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[12]]
no13 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and x not in no8 and x not in no9 and x not in no10 and x not in no11 and x not in no12 and
nombre_contenidoArchivoTaxo[x].split(",")[13] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[13]]
no14 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and x not in no8 and x not in no9 and x not in no10 and x not in no11 and x not in no12 and x
not in no13 and nombre_contenidoArchivoTaxo[x].split(",")[14] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[14]]
no15 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and x not in no8 and x not in no9 and x not in no10 and x not in no11 and x not in no12 and x
not in no13 and x not in no14 and nombre_contenidoArchivoTaxo[x].split(",")[15] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[15]]
no16 = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6 and x not
in no7 and x not in no8 and x not in no9 and x not in no10 and x not in no11 and x not in no12 and x
not in no13 and x not in no14 and x not in no15 and nombre_contenidoArchivoTaxo[x].split(",")[16] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[16]]
brassicaceae = [x for x in unicas if x not in no3 and x not in no4 and x not in no5 and x not in no6
and x not in no7 and x not in no8 and x not in no9 and x not in no10 and x not in no11 and x not in
no12 and x not in no13 and x not in no14 and x not in no15 and x not in no16]
```

```
In [17]:
cantidadesTaxo = []
```

```
cantidades1617Taxo = []

for n in range(3,17):
    cantidadesTaxo.append(len(globals()["no%s"%n])) #contar el total para cada división en la
taxonomía
    cantidades1617Taxo.append([len(nombre_linker[x]) for x in globals()["no%s"%n]].count(16) +
[len(nombre_linker[x]) for x in globals()["no%s"%n]].count(17))#contar cuantos tienen 16 o 17 residuos

cantidadesTaxo.append(len(brassicaceae))#Agregar el total de brassicaceae
cantidades1617Taxo.append([len(nombre_linker[x]) for x in brassicaceae].count(16) +
[len(nombre_linker[x]) for x in brassicaceae].count(17))#Agregar el total de 16 y 17 en brassicaceae
```

```
In [18]:
print cantidadesTaxo
print cantidades1617Taxo
```

```
[4, 2, 0, 6, 3, 0, 6, 4, 161, 14, 0, 77, 116, 77, 70]
[0, 1, 0, 4, 3, 0, 5, 4, 91, 13, 0, 50, 80, 59, 59]
```

```
In [19]:
cantidadesTaxo = [4, 2, 6, 3, 6, 4, 161, 14, 77, 116, 77, 70]#elimino ceros
cantidades1617Taxo = [0, 1, 4, 3, 5, 4, 91, 13, 50, 80, 59, 59]
```

```
In [20]:
nombresTaxo = ["Chlorophyta", "Klebsormidiophyceae", "Bryophyta", "Lycopodiopsida",
"Acrogymnospermae", "basal Magnoliophyta", "Liliopsida", "early-diverging eudicotyledons", "asterids",
"fabids", "Sapindales", "Brassicales"] #Taxonomía de cada grupo que se diferenció de HYL1
```

Análisis a nivel de secuencias

```
In [21]:
##Logos - código extraído de Internet
class RefSeqColor(w.ColorRule):
    def __init__(self, ref_seq, color, description=None):
        self.ref_seq = ref_seq
        self.color = w.Color.from_string(color)
        self.description = description
    def symbol_color(self, seq_index, symbol, rank):
        if symbol == self.ref_seq[seq_index]:
            return self.color
baserules = [
            w.SymbolColor("GSTYC", "green", "polar"),
            w.SymbolColor("NQ", "purple", "neutral"),
            w.SymbolColor("KRH", "blue", "basic"),
            w.SymbolColor("DE", "red", "acidic"),
            w.SymbolColor("PAWFLIMV", "black", "hydrophobic")
        ]
protein_alphabet = w.Alphabet('ACDEFGHIKLMNOPQRSTUVWYBJZX*-adefghiklmnopqrstuvwybjzx', [])

def plotseqlogo(refseq, mseqs, name):
    fasta = "> \n" + "\n> \n".join(mseqs)
    seqs = w.read_seq_data(StringIO.StringIO(fasta), alphabet=protein_alphabet)

    colorscheme = w.ColorScheme([RefSeqColor(refseq, "orange", "refseq")] + baserules,
                                alphabet = protein_alphabet)
    data = w.LogoData.from_seqs(seqs)
    options = w.LogoOptions()
    options.logo_title = name
    options.show_fineprint = False
    options.yaxis_label = ""
    options.color_scheme = colorscheme
    mformat = w.LogoFormat(data, options)
    fname = "%s.pdf" % name
    with open(fname, "wb") as f:
        f.write(w.pdf_formatter(data, mformat))
```

```
In [22]:
#Logo para linkeres de 17 unicos de brassicaceae
if __name__ == "__main__":
    fin = open("linkerUnicos17.txt")
    testdata = fin.readlines()
    #el primero es una secuencia opcional para marcarla en amarillo
    #el ultimo es el nombre del archivo
    plotseqlogo("SSELSQCVSQPVHETGL", testdata, "logoLinkerUnicos")
```

La secuencia de HYL1 no es la más conservada.
Identificar los organismos a los que pertenece cada proteína y agregar linkers de 16 residuos.

```
In [23]:
#para convertir los nombres abreviados de los organismos se extrae el nombre completo para cada
abreviatura de Uniprot
titulos=[]
for rec in parse("uniprot-dsRBD_51428.fasta", "fasta"):
    titulos.append(rec.description)#descripciones de Uniprot en donde se incluye el nombre del
organismo

nombresUniprot_Organismos = {}
for i in titulos:
    if len(i.split("|"))==3:
        nombreUni = i.split("|")[2].split()[0].split("_")[1]
        Orga = i.split("|")[2].split("OS=")[1].split(" ")[0]+ (i.split("|")[2].split("OS=")[1].split("
")[1][0]).upper() + i.split("|")[2].split("OS=")[1].split(" ")[1][1:]
        nombresUniprot_Organismos[nombreUni]=Orga
print "Nombres de organismos", len(nombresUniprot_Organismos)
```

```
Nombres de organismos 3887
```

```
In [24]:
#Cargando nombres de organismos con linker únicos en una lista, y listas de linkers para cada uno
DRBs = []
for rec in parse('linkerUnicos.fasta', "fasta"):
    DRBs.append(rec.name)
    globals()["%s"%rec.name]={}
    globals()["%s"%rec.name]["seqlinker"]=rec.seq

#Cargando secuencias completas de proteínas con linker unicos
for rec in parse('seqsUnicas.fasta', "fasta"):
    globals()["%s"%rec.name]["seq"]=rec.seq
```

```
In [25]:
#Nueva lista agregando longitudes de 16, Agrego nombres de organismos (curado a mano)
DRBs1617 = []
for i in DRBs:
    if 16<=len(globals()["%s"%i]["seqlinker"])<=17:
        DRBs1617.append(i)
        if i.split("_")[1] in nombresUniprot_Organismos.keys():
            globals()["%s"%i]["Orga"]=nombresUniprot_Organismos[i.split("_")[1]]
        else:
            if i.split("_")[1] == "9MAGN":
                globals()["%s"%i]["Orga"]="MacleayaCordata"
            if i.split("_")[1] == "PUNGR":
                globals()["%s"%i]["Orga"]="PunicaGranatum"
```

Separar en tipos DRBS. Para ello hacer un alineamiento, dividir en clusters, y comparar después con los nombres asignados en una publicación (comparar las secuencias y los organismos a los que pertenecen, en caso de coincidencias se renombran con los nombres del paper que especifican el tipo DRB)

```
In [26]:
#Archivo para alinear secuencias completas de proteínas con linkers de 16 y 17 residuos
file_seq1617_file = open("nombreUniprot_seq1617.fasta", "w")
for i in DRBs1617:
    file_seq1617_file.write(">")
    file_seq1617_file.write(i)
    file_seq1617_file.write("\n")
    file_seq1617_file.write(str(globals()["%s"%i]["seq"]))
    file_seq1617_file.write("\n")
file_seq1617_file.close()
```

En este punto se hizo un alineamiento y se generó un árbol con el programa MEGA

```
In [27]:
#Parsear el arbol obtenido
tree_ML = Phylo.read("nombreUniprot_seq1617_align_ML.txt", "newick")#Archivo generado en MEGA
```

```
In [28]:
```

```
tree2_ML = tree_ML.as_phyloxml()
tree2_ML.root.color = "gray"
tree2_ML.clade[0].color = "blue"
tree2_ML.clade[2,0].color = "orange"
tree2_ML.clade[2,1,1,1,1,0].color = "green"
tree2_ML.clade[2,1,1,1,1,1].color = "maroon"
# tree2_ML.clade[1].color = "pink"
Phylo.draw(tree2_ML) #Imprime un gráfico para ver la distribución de los clados
```

```
In [29]:
clados_ML = [[0],[2,0],[2,1,1,1,1,0],[2,1,1,1,1,1]] #Me quedo con estos cuatro clados
```

```
In [30]:
#Extraer clados
cladosML_nombreUniprotOrga = {}
total = 383 #total de los clados de interes
for n in range(4):
    total = total -len(tree_ML.clade[clados_ML[n]].get_terminals())
    items_ML = []
    for i in tree_ML.clade[clados_ML[n]].get_terminals():
        items_ML.append(i.name)
    cladosML_nombreUniprotOrga[n]=items_ML
    print "Clado", n , len(cladosML_nombreUniprotOrga[n])
print "Resto", total #El resto contiene estructuras muy divergentes que quedaron fuero de los clados
principales
```

```
Clado 0 36
Clado 1 98
Clado 2 101
Clado 3 127
Resto 21
```

```
In [31]:
#Organismos en las seqs del paper
nombresOrganismos = {"Pp": "PhyscomitrellaPatens", "Sm": "SelaginellaMoellendorffii",
                     "Sb": "SorghumBicolor", "Zm": "ZeaMays", "Si": "SetariaItalic",
                     "Os": "OryzaSativa", "Ac": "AquilegiaCoerulea", "Mg": "MimulusGuttatus",
                     "Eg": "EucalyptusGrandis", "Cs": "CitrusSinensis", "Tc": "TheobromaCacao",
                     "Th": "ThellungiellaHalophile", "Br": "BrassicaRapa",
                     "At": "ArabidopsisThaliana", "Cus": "CucumisSativus",
                     "Pv": "PhaseolusVulgaris", "Pt": "PopulusTrichocarpa",
                     "Rc": "RicinusCommunis", "Bd": "BrachypodiumDistachyon",
                     "Prp" : "PrunusPersica"}
```

```
In [32]:
#Parseo secuencias del paper
nombrePaperOrga_seq = {}
for rec in parse('seqsPaper-.fasta', "fasta"):
    nombreNuevo = "D" + rec.name.split("D")[1] + "_" + nombresOrganismos[rec.name.split("D")[0]]
    nombrePaperOrga_seq[nombreNuevo] = str(rec.seq)
print len(nombrePaperOrga_seq)
```

```
133
```

```
In [33]:
#Comparar. Misma secuencia y mismo organismo, agrago como namePaper
for i in DRBs1617:
    equal = [x for x in nombrePaperOrga_seq.keys() if nombrePaperOrga_seq[x]==
globals()["%s"%i]["seq"]] #misma secuencia
    if equal:
        if equal[0].split("_")[1] == globals()["%s"%i]["Orga"]:#mismo organismo
            globals()["%s"%i]["namePaper"] = equal[0].split("_")[0]
        else:
            #imprimo las que son para chequear
            print equal[0], i, globals()["%s"%i]["Orga"] #Lo que se imprime se usa para el curado
```

```
[…](Lista para chequear)
```

```
In [34]:
#Curado manual, agrego
V4KDT6_EUTSA["namePaper"] = "DRB1"
A0A022QGJ3_ERYGU["namePaper"] = "DRB1"
A0A022RHZ0_ERYGU["namePaper"] = "DRB2.1"
V4LSP7_EUTSA["namePaper"] = "DRB3-5.1"
V4KZA2_EUTSA["namePaper"] = "DRB3-5.2"
A0A022RMN4_ERYGU["namePaper"] = "DRB6"
```

```
V4NWS1_EUTSA["namePaper"] = "DRB2"
K3Z571_SETIT["namePaper"] = "DRB3-5"
A0A022QSY4_ERYGU["namePaper"] = "DRB3-5"
```

```
In [35]:
#Asignar clados
for c in cladosML_nombreUniprotOrga:
    for prot in cladosML_nombreUniprotOrga[c]:
        globals()["%s"%prot]["cladeNro"]=c
```

```
In [36]:
#Contando DRBs en cada clado
count0 = 0
count1 = 0
count2 = 0
count3 = 0
for i in DRBs1617:
    if "namePaper" in globals()["%s"%i].keys() and "cladeNro" in globals()["%s"%i].keys():
        if globals()["%s"%i]["cladeNro"] == 0:
            count0 += 1
        if globals()["%s"%i]["cladeNro"] == 1:
            count1 += 1
        if globals()["%s"%i]["cladeNro"] == 2:
            count2 += 1
        if globals()["%s"%i]["cladeNro"] == 3:
            count3 += 1

#Contando cantidades totales
for i in range(4):
    print "Cantidades en clado", i, globals()["count%s"%i] ,len(cladosML_nombreUniprotOrga[i])
```

```
Cantidades en clado 0 3 36
Cantidades en clado 1 15 98
Cantidades en clado 2 10 101
Cantidades en clado 3 26 127
```

Curado basado en eliminar secuencias que son muy parecidas. En los casos en que alguna de ella tenga asignada un nombre basado en el paper se prioriza conservar ese nombre

```
In [37]:
#unificando segun parecido en arbol
list_repetidos = []
list_repetidos_todos = []
drbEs=""
drbNoEs = ""
for n1, x1 in enumerate(DRBs1617):
    for n2, x2 in enumerate(DRBs1617[n1+1:]):
        if tree_ML.distance(x1,x2)<0.1:
            if globals()["%s"%x1]["Orga"] == globals()["%s"%x2]["Orga"]:
                if "namePaper" in globals()["%s"%x1].keys():
                    drbEs = x1
                    drbNoEs = x2
                else:
                    drbEs = x2
                    drbNoEs =  x1
                if drbEs in list_repetidos:
                    globals()["%s_repetido"%drbEs].append(drbNoEs)
                    list_repetidos_todos.append(drbNoEs)
                else:
                    if drbNoEs in list_repetidos:
                        globals()["%s_repetido"%drbNoEs].append(drbEs)
                        list_repetidos_todos.append(drbEs)
                    else:
                        if drbEs in list_repetidos_todos or drbNoEs in list_repetidos_todos:
                            for j in list_repetidos:
                                if drbEs in globals()["%s_repetido"%j]:
                                    globals()["%s_repetido"%j].append(drbNoEs)
                                    list_repetidos_todos.append(drbNoEs)
                                else:
                                    if drbNoEs in globals()["%s_repetido"%j]:
                                        globals()["%s_repetido"%j].append(drbEs)
                                        list_repetidos_todos.append(drbEs)
                        else:
```

```
                            list_repetidos.append(drbEs)
                            globals()["%s_repetido"%drbEs]=[]
                            globals()["%s_repetido"%drbEs].append(drbNoEs)
                            list_repetidos_todos.append(drbEs)
```

In [38]:
```
#Curado a mano
for i in ['Q8GUP6_BRAOG', 'Q9FY36_BRAOC', 'Q8GUP9_BRAOB', 'Q8GUP7_BRAOT', 'Q8GUP6_BRAOG',
'Q9FY36_BRAOC', 'Q8GUP9_BRAOB', 'Q8GUP7_BRAOT', 'A0A0D3C8U3_BRAOL', 'Q8GUP8_BRAOL', 'Q8GUQ0_BRAOV',
'Q5IZK9_BRARC', 'Q8GUQ1_BRARR', 'A0A1U8PUF5_GOSHI', 'A0A0D3DNR2_BRAOL', 'I2DBG5_ORYSI']:
    globals()["%s"%i]["Repe"]="Yes"
```

In [39]:
```
for i in range(4):
    print "Cantidades curadas en clado", i, len([x for x in cladosML_nombreUniprotOrga[i] if "Repe"
not in globals()["%s"%x].keys()]), len([x for x in cladosML_nombreUniprotOrga[i] if "Repe" not in
globals()["%s"%x].keys() and "namePaper" in globals()["%s"%x].keys()])
```

```
Cantidades curadas en clado 0 36 3
Cantidades curadas en clado 1 88 15
Cantidades curadas en clado 2 100 10
Cantidades curadas en clado 3 126 26
```

In [40]:
```
#Agregando secuencias de dsRBD1 y dsRBB2
for i in DRBs1617:
    value = nombre_contenidoArchivoDominios[i]
    coincidencias = ""
    coincidencias = re.findall("DOMAIN.[0-9]+\s[0-9]+.DRBM", value, flags=0)

    if len(coincidencias) == 2:
        ppio_uno = 0
        final_uno = 0
        ppio_dos = 0
        final_dos = 0
        ppio_uno = coincidencias[0].split(" ")[1]
        final_uno = coincidencias[0].split(" ")[2]
        ppio_dos = coincidencias[1].split(" ")[1]
        final_dos = coincidencias[1].split(" ")[2]
        globals()["%s"%i]["dsRBD1"] = globals()["%s"%i]["seq"][int(ppio_uno)-1:int(final_uno)]
        globals()["%s"%i]["dsRBD2"] = globals()["%s"%i]["seq"][int(ppio_dos)-1:int(final_dos)]
        globals()["%s"%i]["dsRBD2"] = globals()["%s"%i]["seq"][int(ppio_dos)-1:int(final_dos)]
```

In [41]:
```
#Archivos para alinear con los clados formados
cladosDRB = ["DRB6", "DRB1", "DRB2", "DRB3_5"]
parametros = ["dsRBD1" , "seqlinker", "dsRBD2"]
for nro,region in enumerate(["dsRBD1" , "linkers", "dsRBD2"]):
    for n, clad in enumerate(cladosDRB):
        clados_file = open("Clado%s_%s.fasta"%(clad,region), "w")
        for k in [x for x in cladosML_nombreUniprotOrga[n] if "Repe" not in globals()["%s"%x].keys()]:
            clados_file.write(">")
            clados_file.write(k)
            clados_file.write("\n")
            clados_file.write(str(globals()["%s"%k][parametros[nro]]))
            clados_file.write("\n")
        clados_file.close()
```

In [42]:
```
#Hice alineamiento con MUSCLE de cada región por separada para los logos
```

In [43]:
```
#Parseo de los archivos alineados
for nro,region in enumerate(["dsRBD1" , "linkers", "dsRBD2"]):
    globals()["clados_%sAlign"%region] = {}
    for clad in cladosDRB:
        globals()["%ss"%region] = []
        for rec in parse('Clado%s_%s_align.fas'%(clad, region), "fasta"):
            globals()["%ss"%region].append(str(rec.seq))
        globals()["clados_%sAlign"%region][clad] = globals()["%ss"%region]
```

In [44]:
```
#Archivos txt para logos
for nro,region in enumerate(["dsRBD1" , "linkers", "dsRBD2"]):
```

```
    for clad in cladosDRB:
        globals()["clados_file_txt_%s"%region] = open("Clado%s_%s.txt"%(clad,region), "w")
        for seqregion in globals()["clados_%sAlign"%region][clad]:
            globals()["clados_file_txt_%s"%region].write(seqregion)
            globals()["clados_file_txt_%s"%region].write("\n")
        globals()["clados_file_txt_%s"%region].close()
```

```
In [45]:
#Logos por clados
for region in ["dsRBD1", "linkers", "dsRBD2"]:
    for clad in cladosDRB:
        if __name__ == "__main__":
            fin = open("Clado%s_%s.txt"%(clad, region))
            testdata = fin.readlines()
            seq = ("-"*len(testdata[0]))
            plotseqlogo(seq, testdata, clad+"_%s"%region)
```

Analisis taxonomico de los clados

```
In [46]:
#Separando en grupos taxonómicos
for nroClado in range(4):
    no3 = [x for x in cladosML_nombreUniprotOrga[nroClado] if
nombre_contenidoArchivoTaxo[x].split(",")[3] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[3]]
    no4 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and
nombre_contenidoArchivoTaxo[x].split(",")[4] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[4]]
    no5 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and
nombre_contenidoArchivoTaxo[x].split(",")[5] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[5]]
    no6 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and nombre_contenidoArchivoTaxo[x].split(",")[6] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[6]]
    no7 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and nombre_contenidoArchivoTaxo[x].split(",")[7] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[7]]
    no8 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and nombre_contenidoArchivoTaxo[x].split(",")[8] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[8]]
    no9 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and x not in no8 and
nombre_contenidoArchivoTaxo[x].split(",")[9] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[9]]
    no10 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and x not in no8 and x not in no9 and
nombre_contenidoArchivoTaxo[x].split(",")[10] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[10]]
    no11 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and x not in no8 and x not in no9 and x not in no10 and
nombre_contenidoArchivoTaxo[x].split(",")[11] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[11]]
    no12 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and x not in no8 and x not in no9 and x not in no10 and x not
in no11 and nombre_contenidoArchivoTaxo[x].split(",")[12] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[12]]
    no13 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and x not in no8 and x not in no9 and x not in no10 and x not
in no11 and x not in no12 and nombre_contenidoArchivoTaxo[x].split(",")[13] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[13]]
    no14 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and x not in no8 and x not in no9 and x not in no10 and x not
in no11 and x not in no12 and x not in no13 and nombre_contenidoArchivoTaxo[x].split(",")[14] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[14]]
    no15 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and x not in no8 and x not in no9 and x not in no10 and x not
in no11 and x not in no12 and x not in no13 and x not in no14 and
nombre_contenidoArchivoTaxo[x].split(",")[15] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[15]]
    no16 = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4 and x not
in no5 and x not in no6 and x not in no7 and x not in no8 and x not in no9 and x not in no10 and x not
in no11 and x not in no12 and x not in no13 and x not in no14 and x not in no15 and
nombre_contenidoArchivoTaxo[x].split(",")[16] !=
nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[16]]
```

```
    brassicaceae = [x for x in cladosML_nombreUniprotOrga[nroClado] if x not in no3 and x not in no4
and x not in no5 and x not in no6 and x not in no7 and x not in no8 and x not in no9 and x not in no10
and x not in no11 and x not in no12 and x not in no13 and x not in no14 and x not in no15 and x not in
no16]

    globals()["cantidadesTaxo%s"%nroClado] = []

    for x in range(3,17):
        globals()["cantidadesTaxo%s"%nroClado].append(len(globals()["no%s"%x]))
    globals()["cantidadesTaxo%s"%nroClado].append(len(brassicaceae))
```

```
In [47]:
nombresTaxo =  nombre_contenidoArchivoTaxo["DRB1_ARATH"].split(",")[3:18]
```

Identidad de secuencia por región y por clado

```
In [48]:
#Para cada región se analiza la identidad de la secuencia en cada clado
for k in ["dsRBD1", "linkers", "dsRBD2"]:#Para cada región
    print "identidad para", k
    for clado in globals()["clados_%sAlign"%k]: #Para cada clado
        identidadTotal = 0
        for pos in range(len(globals()["clados_%sAlign"%k][clado][1])): #Para cada posición
            valores = []
            for seq in globals()["clados_%sAlign"%k][clado]:
                valores.append(seq[pos]) #Residuos en esa posición
            d = defaultdict(int)
            for i in valores:
                d[i] += 1#Va contando cantidad de ocurrencias
            maxcons = max(d.iteritems(), key=lambda x: x[1]) #Busca la que está en mayor cantidad
            identidad = maxcons[1]*100/len(globals()["clados_%sAlign"%k][clado]) #Calcula la identidad
            identidadTotal = identidadTotal + identidad#Suma para todas las posiciones en esa región
        print clado,  identidadTotal/len(globals()["clados_%sAlign"%k][clado][1])
```

```
identidad para dsRBD1
DRB3_5 88
DRB6 72
DRB1 84
DRB2 92
identidad para linkers
DRB3_5 82
DRB6 70
DRB1 60
DRB2 96
identidad para dsRBD2
DRB3_5 84
DRB6 75
DRB1 81
DRB2 92
```

```
In [49]:
ind = np.arange(4)
dsRBD1 = [84, 92, 88 , 72]
linker = [60, 96, 82 , 70]
dsRBD2 = [81, 92, 84 , 75]
```

MODELADO ESTRUCTURAS INICIALES EN MODELLER

```
In [1]:
from modeller import *
from modeller.automodel import *
import sys
```

```
In [2]:
#Códigos para inicializar Modeller
env = environ()
env.io.hetatm = True #Para poder considerar el Gd
aln = alignment(env)
```

```
[…](Inicia Modeller)
```

Generación de archivo .ali

```
In [ ]:
#a = alignment(env,file="FCM21_seqsCompletas_ubicados.fasta", alignment_format="FASTA")
#a.write(file="D1D2_seqsCompletas_ubicados.ali",alignment_format='PIR')
#aln.append(file="FCM21_seqsCompletas_ubicados.ali",align_codes="FCM21_ubicados")
#mdl = model(env,file="3adg_ubicada_v2.pdb",model_segment=("FIRST:A","LAST:A"))
#aln.append_model(mdl,align_codes="3adg_ubicada_v2",atom_files="3adg_ubicada_v2.pdb")
#mdl = model(env,file="3adj_ubicada_v2.pdb",model_segment=("FIRST:A","LAST:A"))
#aln.append_model(mdl,align_codes="3adj_ubicada_v2",atom_files="3adj_ubicada_v2.pdb")
#mdl = model(env,file="LBT1_ubicado_v2.pdb",model_segment=("FIRST:A","LAST:A"))
#aln.append_model(mdl,align_codes="LBT1_ubicado_v2",atom_files="LBT1_ubicado_v2.pdb")
#mdl = model(env,file="LBT2_ubicado_v2.pdb",model_segment=("FIRST:A","LAST:A"))
#aln.append_model(mdl,align_codes="LBT2_ubicado_v2",atom_files="LBT2_ubicado_v2.pdb")
#aln
#aln.align2d()
#aln.write(file="FCM21-LBT-3adg-3adj_ubicados.ali", alignment_format="PIR")
#aln.write(file="FCM21-LBT-3adg-3adj_ubicados.pap", alignment_format="PAP")
```

Generación de modelos

```
In [3]:
#Seteo de parámetros y archivos molde
a =automodel(env, alnfile="D1D2-LBT-3adg-3adj-LBT_ubicados_FLO_Gd.ali", knowns =
          ("LBT1_ubicado_v2_LBT_Gd", "3adg_ubicada_v2","3adj_ubicada_v2","LBT2_ubicado_v2_LBT_Gd"),
          sequence="D1D2_ubicados_Gd", assess_methods=(assess.DOPE, assess.GA341))
```

[…] (Salida del programa)

```
In [4]:
a.starting_model = 1
a.ending_model = 100 #Cantidad de modelos a generar
a.make()
```

[…] (Salida de la generación de modelos)

```
In [5]:
#Para evaluar el de menor DOPE
ok_models = [x for x in a.outputs if x["failure"] is None]
print "La cantidad de modelos OK es", len(ok_models)
key = "DOPE score"
if sys.version_info[:2] == (2,3):
    print "Es versión 2.3"# Python 2.3 no tiene el argument 'key'
    ok_models.sort(lambda a,b: cmp(a[key], b[key])) #Ordena en función del DOPE
else:
    ok_models.sort(key=lambda a: a[key])
m = ok_models[0] #Mejor modelo
print("Top model: %s (DOPE score %.3f)" % (m["name"], m[key]))
```

```
La cantidad de modelos OK es 100
Top model: D1D2_ubicados_Gd.B99990006.pdb (DOPE score -15774.713)
```

GENERACIÓN DE ENSAMBLES CON ROSETTA

```
In [1]:
import pyrosetta as ro
import numpy as np
from rosetta.core.scoring import *
```

```
In [2]:
###ro.init()
```

```
In [3]:
#Inicializar Rosetta
ro.init("-in:auto_setup_metals")
```

[…] (Salida del programa)

```
In [4]:
scoreFun=ro.get_fa_scorefxn()
```

[…] (Salida del programa)

```
In [5]:
three2one={"ALA":"A","ARG":"R","ASN":"N","ASP":"D","CYS":"C","GLN":"Q","GLU":"E","GLY":"G","HIS":"H","
ILE":"I","LEU":"L","LYS":"K","MET":"M","PHE":"F","PRO":"P","SER":"S","THR":"T","VAL":"V","TRP":"W","TY
R":"Y"}
```

Defino valore para phi y psi

```
In [6]:
FMdbase=open("./phi_psi.txt",'r').readlines()
#La base de datos de FM tiene varios tipos de residuos
#que no son los aminoasidos estandar
residueTypes=set([x.split()[0] for x in FMdbase])

#Crea un diccionario con los valores de phi y psi de la base de datos de Flexible Meccano para cada
tipo de residuo
FMphi=dict()
FMpsi=dict()
FMphipsi=dict()
for r in residueTypes:
    if r in list(three2one.keys()):#toma solo los residuos estandar
        FMphi[three2one[r]]=[float(x.split()[1]) for x in FMdbase if len(x.split())==3 and x[:3]==r]
        FMpsi[three2one[r]]=[float(x.split()[2]) for x in FMdbase if len(x.split())==3 and x[:3]==r]
        FMphipsi[three2one[r]]=[[float(x.split()[1]),float(x.split()[2])] for x in FMdbase if
len(x.split())==3 and x[:3]==r]
```

Defino el molde y los residuos a rotar

```
In [7]:
#from pyrosetta.toolbox import cleanATOM
#cleanATOM("D1D2_ubicados_Gd.B99990006.pdb")#elimina el Gd
```

```
In [8]:
poseD1D2_Gd_original=ro.pose_from_pdb("D1D2_ubicados_Gd.B99990006_HETATM_CA.pdb")
```

```
[…] Salida del programa
```

```
In [9]:
seqD1D2_Gd=poseD1D2_Gd_original.sequence()#Secuencia
NumResiduesD1D2_Gd=poseD1D2_Gd_original.total_residue()#Cantidad de residuos
```

```
GYIDTNNDGWIEGDELHMVFKSRLQEYAQKYKLPTPVYEIVKEGPSHKSLFQSTVILDGVRYNSLPGFFNRKAAEQSAAEVALRELAKSSELSQCVSQPVHE
TGLCKNLLQEYAQKMNYAIPLYQCQKVETLGRVTQFTCTVEIGGIKYTGAATRTKKDAEISAGRTALLAIQSVDYIDTNNDGWIEGDELVDZZ
195
```

```
In [10]:
scoreFun(poseD1D2_Gd_original) #Energía del molde
```

```
[…] Salida del programa
1189.546684925321
```

```
In [11]:
residuosAModificar=list(range(90,104))+[17,18]+[175,176] #Defino residues a rotar
```

```
[90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 17, 18, 175, 176]
```

Generación del ensamble

```
In [12]:
numAgenerar=10000#Numero máximo de estructuras del ensamble
derechos=0
rechazados=0
metropolis=0
metroTemp=100
zarpadosDeVueltas=0
maxVueltas=100#Cantidad de veces que va a intentar si no consigue mejorar el modelo

poseD1D2_Gd=ro.Pose()
poseModificado_Gd =ro.Pose()

#Itera por la cantidad máxima de Ensambles (NumAgenerar)
for i in range(numAgenerar):
    residuosAModificar=list(range(90,104))+[17,18]+[175,176] #Se definen los nros de los residuos a
modificar
    poseD1D2_Gd.assign(poseD1D2_Gd_original) #cada vez q se comienza de nuevo se parte de un pose
igual al de la proteína inicial

    #Itera por los residuos a modificar, en última instancia se habrán modificado todos en cada nueva
estructura
    for r in range(len(residuosAModificar)):
        residuoModificado=residuosAModificar.pop(np.random.randint(len(residuosAModificar))) #elige un
residuo al azar
        resi=seqD1D2_Gd[residuoModificado-1] #en letras el aminoácido al que corresponde
```

```
        sco_ini=scoreFun(poseD1D2_Gd)#score inicial para las vueltas dentro de un residuo


        probaDeNuevo=True #inicio en True para las vueltas dentro de un residuo
        vueltas=0
        while probaDeNuevo and vueltas < maxVueltas:    #mientras no se consiga un score mejor sigue
dando vueltas
                                                        #dentro del mismo residuo
            choice=np.random.randint(len(FMpsi[resi]))  #toma un valor de la base de datos de FM al
azar y se lo
                                                        #asigna al residuo
            Phi=FMphipsi[resi][choice][0]
            Psi=FMphipsi[resi][choice][1]

            poseModificado_Gd.assign(poseD1D2_Gd)

            poseModificado_Gd.set_phi(residuoModificado,Phi)#modifica pose
            poseModificado_Gd.set_psi(residuoModificado,Psi)#modifica pose

            scoNew=scoreFun(poseModificado_Gd)#calcula el score y lo compara con el anterior

            if scoNew<= sco_ini:

                poseD1D2_Gd.assign(poseModificado_Gd)

                probaDeNuevo=False #ya se puede pasar al residuo que sigue
                sco_ini=scoNew #nuevo score inicial
                derechos+=1
            else:
                delta=scoNew-sco_ini #ver cuánta es la diferencia
                prob=np.exp(delta)
                azar=np.random.random()*metroTemp
                if azar>prob:

                    poseD1D2_Gd.assign(poseModificado_Gd)

                    probaDeNuevo=False#ya se puede pasar al residuo que sigue
                    sco_ini=scoNew#nuevo score inicial
                    metropolis+=1
                else:
                    rechazados+=1
                    vueltas+=1
            if vueltas==maxVueltas:
                zarpadosDeVueltas+=1    #casos en que luego de alcanzar el máximo de vueltas no se
consiguió
                                        #mejorar el score cambiando phi y psi para ese residuo

    print("\r Score %i = %5.7e"%(i,sco_ini))#antes decia scoNew

    poseD1D2_Gd.dump_pdb(("%05d_%i_D1D2_Gd.pdb"%(i,metroTemp)))

print "Derechos=%i, metropolis=%i, rechazados=%i, muchas vueltas=%i"
%(derechos,metropolis,rechazados,zarpadosDeVueltas)
```

[…] En la salida se muetran los score para cada estructura y la cantidad de derechos, metropolis, rechazados y muchas vueltas. Se repite cuatro veces más. Las salidas fueron:
Derechos=117586, metropolis=52032, rechazados=1718366, muchas vueltas=10382
Derechos=117005, metropolis=52614, rechazados=1725811, muchas vueltas=10381
Derechos=117455, metropolis=52101, rechazados=1729328, muchas vueltas=10444
Derechos=117148, metropolis=52383, rechazados=1721568, muchas vueltas=10469
Derechos=117221, metropolis=52351, rechazados=1726234, muchas vueltas=10428

DISTANCIAS

```
In [1]:
import numpy as np
import os, sys
from Bio.PDB import *
parser = PDBParser()
```

```
In [2]:
```

```
#Importo la lista de archivos pdb del ensamble
listaPDB=[x for x in os.listdir("/D1D2_Ensambles/") if "100_D1D2_Gd.pdb" in x]
print len(listaPDB)
```
```
50000
```

```
In [3]:
#Forma de parseo de las estructuras: Estructura[modelo]["cadena"][residuo]["atomo"]
largoSeq = len(parser.get_structure("00000", "/D1D2_Ensambles/00000_100_D1D2_Gd.pdb")[0][" "])
```

```
In [4]:
#Distancias al primer metal
diccDistanciasGd1 = {}
rangoLargoSeq =  range(1,largoSeq-1)
listaPDB.sort()
for s in listaPDB:
    sys.stdout.write("\r%s"%s)#Para ir viendo el progreso
    #Crear un objeto estructura del archivo PDB (nombre de la estructura, archivo)
    structure = parser.get_structure(s.split("_")[0], "/D1D2_Ensambles/"+s)
    distancias1 = [((structure[0][" "][i]["N"])-(structure[0][" "]['H_ CA', largoSeq, ' ']["CA"])) for
i in (rangoLargoSeq)]
    diccDistanciasGd1[s.split("_")[0]] = distancias1
np.save('distancias_D1D2_Gd1.npy', diccDistanciasGd1)
```
```
49999_100_D1D2_Gd.pdb
```

```
In [5]:
#Distancias al segundo metal
diccDistanciasGd2 = {}
rangoLargoSeq =  range(1,largoSeq-1)
listaPDB.sort()
for s in listaPDB:
    sys.stdout.write("\r%s"%s)#Para ir viendo el progreso
    #Crear un objeto estructura del archivo PDB (nombre de la estructura, archivo)
    structure = parser.get_structure(s.split("_")[0], "/D1D2_Ensambles/"+s)
    distancias2 = [((structure[0][" "][i]["N"])-(structure[0][" "]['H_ CA', largoSeq-1, ' ']["CA"]))
for i in (rangoLargoSeq)]
    diccDistanciasGd2[s.split("_")[0]] = distancias2
np.save('distancias_D1D2_Gd2.npy', diccDistanciasGd2)
```
```
49999_100_D1D2_Gd.pdb
```

ANÁLISIS GRÁFICO DEL ENSAMBLE

```
In [1]:
import sys
import numpy as np
from Bio import PDB
```

```
In [2]:
#Definición de centro de masa y vectores para D1
def angSinRef(file_name):

    parser=PDB.PDBParser()
    structure = parser.get_structure("Referencia", file_name)

    CAs=[atom for atom in structure.get_atoms() if atom.name=="CA"]
    CAsD1=CAs[19:87] #Átomos de D1
    coordsCAD1=np.array([x.coord for x in CAsD1])
    centroMasaCAD1=np.mean(coordsCAD1,axis=0)
    coordsCAzeroD1=coordsCAD1-centroMasaCAD1#Se lleva al centro de los ejes de coordenadas

    inertiaD1 = np.dot(coordsCAzeroD1.T, coordsCAzeroD1)
    e_valuesD1, e_vectorsD1 = np.linalg.eig(inertiaD1) #Cálculo de vectores
    orderD1 = np.argsort(e_valuesD1)[-3:] #Se ordenan
    eval3D1, eval2D1, eval1D1 = e_valuesD1[orderD1] #Se asignan en orden
    axis3D1, axis2D1, axis1D1 = e_vectorsD1[:, orderD1].T

    return axis3D1, axis2D1, axis1D1, CAsD1
```

```
In [3]:
#Cálculo de posiciones de los vectores para una de las estructuras que es tomada como referencia
axis3D1, axis2D1, axis1D1, RefCAsD1 = angSinRef("/D1D2_Ensambles/00014_100_D1D2_Gd.pdb")
```

```
In [4]:
```

```
parser=PDB.PDBParser()
largoSeq = len(parser.get_structure("00000", "/D1D2_Ensambles/00000_100_D1D2_Gd.pdb")[0][" "])
print largoSeq
```
195

```
In [5]:
#Función para el cálculo de angulos y distancias con estructuras alineadas
def angConRef(file_name):

    parser=PDB.PDBParser()
    structure = parser.get_structure(file_name.split("/")[-1].split("_")[0], file_name)

    CAs=[atom for atom in structure.get_atoms() if atom.name=="CA"]
    CAsD1=CAs[19:87] #Átomos de D1

    sup = PDB.Superimposer()
    sup.set_atoms(RefCAsD1, CAsD1) #Alinea D1 de la estructura con el D1 de la referencia
    sup.apply(CAs)

    coordsMov=np.array([x.coord for x in CAs])
    CAsD1_Mov=CAs[19:87]
    CAsD2_Mov=CAs[104:174]

    coordsCAD1=np.array([x.coord for x in CAsD1_Mov])
    coordsCAD2=np.array([x.coord for x in CAsD2_Mov])

    centroMasaCAD1=np.mean(coordsCAD1,axis=0)
    centroMasaCAD2=np.mean(coordsCAD2,axis=0)

    coordsCAzeroD1=coordsCAD1-centroMasaCAD1
    coordsCAzeroD2=coordsCAD2-centroMasaCAD2

    vectorEntreCOM = (centroMasaCAD1-centroMasaCAD2)
    vectorEntreCOM = vectorEntreCOM/abs(np.linalg.norm(vectorEntreCOM))

    ang1 = (np.arccos(np.dot(axis1D1,vectorEntreCOM)))*360/np.pi/2 #Defición del primer ángulo

    projEquis=np.dot(axis3D1, vectorEntreCOM)
    projYe=np.dot(axis2D1, vectorEntreCOM)

    ang2=(np.arctan(projYe/projEquis))*360/(2*np.pi) #Definición del Segundo ángulo
    #Correcciones según cuadrantes
    if projEquis<0:
        if projYe>0:
            ang2 = 180+ang2
        else:
            ang2 = -(180-ang2)

    distGdGd = structure[0][" "]['H_ CA', largoSeq, ' ']["CA"]-structure[0][" "]['H_ CA', largoSeq-1,
' ']["CA"] #Distancia entre metales

    return ang1, ang2, np.linalg.norm(centroMasaCAD1-centroMasaCAD2), distGdGd
```

```
In [6]:
#Cálculo de distancias y ángulos
DiccAngulos={}
for i in range(50000):
    iletras = str(i).zfill(5)
    angulos = angConRef("../2_Ensambles_Rosetta/D1D2_Ensambles/%s_100_D1D2_Gd.pdb"%iletras)
    DiccAngulos[iletras] = angulos
    sys.stdout.write("\r%i" % (i))
np.save("D1D2_AngulosDistCOMDistGd.npy", DiccAngulos)
```
49999

PREDICCIÓN DE PATRONES PRE PARA EL ENSAMBLE

```
In [1]:
import math as math
import numpy as np
import sys
```

Defino parámetros y funciones

In [2]:
```
u0 = 1.26e-6 #Magnetic permeability of a vacuum = 1.26x10-6 N/A2
u04pi2 = (u0/(4*math.pi))**2
yI = 42.58e6 #Gyromagnetic ratio H+ = 42.58 MHz/T = 42.58x106 /s*T
ge = -2.002 #Electron g factor = -2.002
uB = 9.27e-24 #Electron Bohr magneton = 9.27x10-24 J/T
S = 7/2 #Electron spin quantum number (=J) = 7/2
wI = 2*np.pi*700e6 #Nuclear Larmor frequency = 4398.23 x106 s-1
wS = 2*np.pi*460.720e9#Electron Larmor frequency = 2.895x1012 s-1
k = 1.38e-23 #kB Boltzman constant = 1.38x10-23 J/K
T = 298 #Absolute temperature = (298K)
```

In [3]:
```
def kSolomon(te,tr,tm):
    tc=1/(1/te +1/tr +1/tm)#
    kSolomon2 = ( (yI**2)*(ge**2)*(uB**2)*S*(S+1) ) / 15
    kSolomon32 = tc / ( 1 + (((wI-wS)**2) * (tc**2) ) )
    kSolomon33 = 3*tc / ( 1 + ( (wI**2)*(tc**2) ) )
    kSolomon34 = 6*tc / ( 1 + ( ( (wI + wS)**2 ) * (tc**2) ) )
    kSolomon35 =  6*tc / (1+ ((wS**2) * (tc**2)) )
    kSolomon = u04pi2 * kSolomon2 * ((4*tc) + kSolomon32 + kSolomon33 + kSolomon34 + kSolomon35)
    return kSolomon
```

In [4]:
```
def kCurie(tr,tm):
    tCurie=1/(1/tr + 1/tm)
    kCurie3 =((wI**2) * (ge**4) * (uB**4) * (S**2) * ((S+1)**2))/((3*k*T)**2)
    kCurie4 = ( 4*tCurie ) + ( ( 3*tCurie ) / ( 1 + ((wI**2) * (tCurie**2)) ) )
    kCurie = 0.2 * u04pi2 * kCurie3 * kCurie4
    return kCurie
```

In [5]:
```
def R2para(r,te,tr,tm):
    R2=(kSolomon(te,tr,tm) + kCurie(tr,tm))/r**6
    return R2
```

In [6]:
```
t2file = open("/T2.txt", "r")#Archivo con valores de R2 experimentales
T2_contenido = t2file.readlines()
t2file.close()
```

In [7]:
```
R20s = {}
for i in T2_contenido[1:]:
    R20s[int(i.split("\t")[0][0:-4])] = float(i.split("\t")[1][0:-1])
```

In [8]:
```
def II0r(r, posD1D2):
    if (17<posD1D2<174) and (posD1D2-1 in R20s.keys()):
        te = 2e-9
        tr = 5e-9
        tm = 5e-9
        posNB14 = posD1D2-1
        R20 = R20s[posNB14]/1000
        R2p=R2para(r,te,tr,tm)
        R2=R20+R2p
        return R20/R2
    else:
        if 210<posD1D2<367 and posD1D2-1-193 in R20s.keys():
            te = 2e-9
            tr = 5e-9
            tm = 5e-9
            posNB14 = posD1D2-1-193
            R20 = R20s[posNB14]/1000
            R2p=R2para(r,te,tr,tm)
            R2=R20+R2p
            return R20/R2
        else:
            return -0.1
```

In [9]:
```
diccDistanciasD1D2_Gd1 = np.load("distancias_D1D2_Gd1.npy").item()
diccDistanciasD1D2_Gd2 = np.load("distancias_D1D2_Gd2.npy").item()
```

Unificar medidas considerando ambos metales

```
In [10]:
diccDistanciasD1D2= {}
for i in range(0,50000):
    iceros = str(i).zfill(5)
    diccDistanciasD1D2[iceros] =(diccDistanciasD1D2_Gd1[iceros] + diccDistanciasD1D2_Gd2[iceros])
    sys.stdout.write("\r%s"%i)
```

```
49999
```

Cálculo de distancias

```
In [11]:
#Grabar diccionario con I/I0 D1D2
dicII016={}
for i in sorted(diccDistanciasD1D2.keys()):
    II0r_array_i = np.array([II0r(val*1e-10,n) for n,val in enumerate(diccDistanciasD1D2[i])])
    dicII016[i]=II0r_array_i
    sys.stdout.write("\r%s"%i)
np.save('D1D2_II0.npy', dicII016)
```

```
49999
```

PRIMERA REDUCCIÓN DEL ENSAMBLE

```
In [1]:
import sys
import numpy as np
from scipy.cluster.hierarchy import fcluster
from scipy.cluster.hierarchy import dendrogram, linkage
from collections import Counter
```

```
In [2]:
#Definición de distancia
def distancia(v1,v2):
    return np.sum((v1-v2)**2)
```

```
In [3]:
#Cargar los valores de II0
II0D1D2=np.load("D1D2_II0.npy").item()
todosnombres=sorted(II0D1D2.keys())
```

Para reducir el procesamiento divido en 5 grupos de 10000 estructuras

```
In [4]:
for i in range(5):
    globals()["rango_%s"%(i)] = [str(x).zfill(5) for x in range(10000*i, 10000*(i+1))]#nombres
    globals()["II0D1D2_%s"%(i)] = {x:II0D1D2[x] for x in globals()["rango_%s"%(i)]}#Perfiles II0
```

```
In [5]:
#Extraigo de cada perfil de PRE calculado las regiones variables de interés
for i in range(5):
    globals()["II0D1D2_D2D1_%s"%i] = {}
    for x in globals()["rango_%s"%(i)]:
        arrayD2D1 = []
        for n in range(104,174) + range((193+18),(193+88)):
            arrayD2D1.append(globals()["II0D1D2_%s"%i][x][n])
            globals()["II0D1D2_D2D1_%s"%i][x] = np.array(arrayD2D1)
```

Matriz con distancias entre perfiles

```
In [6]:
#Crear una matriz vacia
for i in range(5):
    globals()["matrixDistD1D2_D2D1_%s"%i] = np.zeros([10000,10000])
```

```
In [7]:
##Esto fue repetido para cada grupo

#Llenar la matriz con valores de distancias para D1D2
for i,n1 in enumerate(rango_0[:]):
    for j,n2 in enumerate(rango_0[i:]):
        d=distancia(II0D1D2_D2D1_0[n1],II0D1D2_D2D1_0[n2])
```

```
        matrixDistD1D2_D2D1_0[i,j+i]=d
        matrixDistD1D2_D2D1_0[j+i,i]=d
    if i%10 == 0:
        sys.stdout.write("\r%4i" % (i))
np.save("matrizDistII0D1D2_D2D1_0.npy", matrixDistD1D2_D2D1_0)
```
```
9990
```

Agrupamiento

```
In [8]:
##Esto fue repetido para cada grupo
clusteringD1D2_D2D1_0 = linkage(matrixDistD1D2_D2D1_0, "ward")
np.save("clusteringD1D2_D2D1_0.npy", clusteringD1D2_D2D1_0)
```

```
In [9]:
#Definiendo composicion de cluster segun limite de cantidad de cluster formados
k = 3 #número de clusters
for i in range(5):
    globals()["clustersD1D2_D2D1_3cluster_%s"%i] = fcluster(globals()["clusteringD1D2_D2D1_%s"%i], k,
criterion='maxclust')
    Counter(globals()["clustersD1D2_D2D1_3cluster_%s"%i])

    globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i] = {}
    for j in range(1,k+1):
        globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i][j] = [(str(n+(10000*i)).zfill(5)) for n,x in
enumerate(globals()["clustersD1D2_D2D1_3cluster_%s"%i]) if x == j]

    np.save("DiccClusters_D1D2_3cluster_%s.npy"%i, globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i])
```

```
In [10]:
#Ordenar los clusters según cantidad de estructuras que contienen
for i in range(5):
    largo1 = len(globals()["DiccClusters_D1D2_D2D1_3cluster_%s_sinordenar"%i][1])
    largo2 = len(globals()["DiccClusters_D1D2_D2D1_3cluster_%s_sinordenar"%i][2])
    largo3 = len(globals()["DiccClusters_D1D2_D2D1_3cluster_%s_sinordenar"%i][3])
    largos = [largo1, largo2, largo3]
    largosordenados = sorted(largos)

    indicesLargosordenados = []
    for n in [2,1,0]:
        indicesLargosordenados.append([x for x in
globals()["DiccClusters_D1D2_D2D1_3cluster_%s_sinordenar"%i].keys() if
                           len(globals()["DiccClusters_D1D2_D2D1_3cluster_%s_sinordenar"%i][x]) ==
largosordenados[n]])

    globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i] = {}


globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i][1]=globals()["DiccClusters_D1D2_D2D1_3cluster_%s_sin
ordenar"%i][indicesLargosordenados[0][0]]

globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i][2]=globals()["DiccClusters_D1D2_D2D1_3cluster_%s_sin
ordenar"%i][indicesLargosordenados[1][0]]

globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i][3]=globals()["DiccClusters_D1D2_D2D1_3cluster_%s_sin
ordenar"%i][indicesLargosordenados[2][0]]
```

```
In [11]:
for i in range(5):
    np.save("DiccClusters_D1D2_3clusterOrdenados_%s.npy"%i,
globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i])
```

Cargar valores experimentales y seleccionar regiones para comparar con los clusters

```
In [12]:
#Cargar los valores de II0 experimentales
expFCM20 = open("FCM20_IvsI0_CaColor_160816.txt").readlines()
expFCM21 = open("FCM21_IvsI0_CaColor_160920.txt").readlines()
```

```
In [13]:
resiFCM20_exp = [int(x.split()[0])+1 for x in expFCM20]
II0FCM20_exp = [float(x.split()[1]) for x in expFCM20]
```

```
resiFCM21_exp = [int(x.split()[0])+1 for x in expFCM21]
II0FCM21_exp = [float(x.split()[1]) for x in expFCM21]
```

Extrear de los PRE calculados los residuos para los cuales hubo medidas experimentales

```
In [14]:
II0FCM20_exp_D1 = []
for x in range(len(II0FCM20_exp)):
    if resiFCM20_exp[x] in range(3,73) :
        II0FCM20_exp_D1.append(II0FCM20_exp[x])

resiFCM20_exp_D1 = [x for x in resiFCM20_exp if x in range(3,73)]

print len(II0FCM20_exp_D1), len(resiFCM20_exp_D1)
```

```
49 49
```

```
In [15]:
II0FCM21_exp_D2 = []
for x in range(len(II0FCM21_exp)):
    if resiFCM21_exp[x] in range(104,174):
        II0FCM21_exp_D2.append(II0FCM21_exp[x])

resiFCM21_exp_D2 = [x for x in resiFCM21_exp if x in range(104,174)]

print len(II0FCM21_exp_D2), len(resiFCM21_exp_D2)
```

```
42 42
```

```
In [16]:
II0D1D2_D2D1_resiexp={}
for i in todosnombres:
    II0D1D2_D2D1_resiexp[i]=[]
    for n in resiFCM21_exp_D2:
        II0D1D2_D2D1_resiexp[str(i)].append(II0D1D2[str(i)][n-1])
    for n in resiFCM20_exp_D1:
        II0D1D2_D2D1_resiexp[str(i)].append(II0D1D2[str(i)][n-1+193+15])
```

```
In [17]:
#Por cluster
for r in range(5):
    for i in range(1,4):
        globals()["II0D1D2_D2D1_resiexp_%s_cluster%s"%(r, str(i).zfill(2))] = {}
        for j in globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%r][i]:
            globals()["II0D1D2_D2D1_resiexp_%s_cluster%s"%(r,str(i).zfill(2))][j] =
II0D1D2_D2D1_resiexp[j]
```

```
In [18]:
II0FCM21_exp_D2_FCM20_exp_D1 = II0FCM21_exp_D2 + II0FCM20_exp_D1
print len(II0FCM21_exp_D2_FCM20_exp_D1)
```

```
91
```

Divergencias entre PRE de cada estructura simulada y los resultados experimentales

```
In [19]:
for r in range(5):
    for i in range(1,4):
        globals()["diccDistD1D2_calc_D2D1_%s_cluster%s"%(r,str(i).zfill(2))] = {}
        for n in globals()["II0D1D2_D2D1_resiexp_%s_cluster%s"%(r,str(i).zfill(2))].keys():

globals()["diccDistD1D2_calc_D2D1_%s_cluster%s"%(r,str(i).zfill(2))][n]=distancia(np.array(II0FCM21_ex
p_D2_FCM20_exp_D1), np.array(II0D1D2_D2D1_resiexp[n]))
```

SEGUNDA REDUCCIÓN DEL ENSAMBLE

```
In [1]:
import sys
import numpy as np
from Bio import PDB
from scipy.cluster.hierarchy import fcluster
```

```
from scipy.cluster.hierarchy import dendrogram, linkage
from collections import Counter
```

```
In [2]:
#Definición de distancia
def distancia(v1,v2):
    return np.sum((v1-v2)**2)
```

```
In [3]:
#Cargar los valores de II0
II0D1D2=np.load("D1D2_II0.npy").item()
todosnombres=sorted(II0D1D2.keys())
```

Reagrupamiento de clusters anteriores

```
In [4]:
cluster2y3_desordenados = []
for i in range(5):
    for c in range(2,4):
        for stru in globals()["DiccClusters_D1D2_D2D1_3cluster_%s"%i][c]:
            cluster2y3_desordenados.append(stru)

cluster2y3 = sorted(cluster2y3_desordenados)
print len(cluster2y3)
np.save("cluster2y3.npy", cluster2y3)
```

```
9421
```

```
In [5]:
II0D1D2_2y3 = {}
for i in cluster2y3:
    II0D1D2_2y3[i] = II0D1D2[i]
print len(II0D1D2_2y3)
```

```
9421
```

```
In [6]:
#Extraigo de cada perfil de PRE calculado las regiones variables de interés
II0D1D2_2y3_D2D1 = {}
for x in II0D1D2_2y3.keys():
    arrayD2D1 = []
    for n in range(104,174) + range((193+18),(193+88)):
        arrayD2D1.append(II0D1D2[x][n])
        II0D1D2_2y3_D2D1[x] = np.array(arrayD2D1)
```

Matriz con distancias entre perfiles

```
In [7]:
#Crear una matriz vacia
matrixDistD1D2_2y3_D2D1 = np.zeros([9421,9421])
```

```
In [8]:
#Llenar la matriz con valores de distancias para D1D2 cluster 2 y 3
for i,n1 in enumerate(cluster2y3[:]):
    for j,n2 in enumerate(cluster2y3[i:]):
        d=distancia(II0D1D2_2y3_D2D1[n1],II0D1D2_2y3_D2D1[n2])
        matrixDistD1D2_2y3_D2D1[i,j+i]=d
        matrixDistD1D2_2y3_D2D1[j+i,i]=d
    if i%10 == 0:
        sys.stdout.write("\r%4i" % (i))
np.save("matrizDistII0D1D2_2y3_D2D1.npy", matrixDistD1D2_2y3_D2D1)
```

```
9420
```

Agrupamiento

```
In [9]:
clusteringD1D2_2y3_D2D1 = linkage(matrixDistD1D2_2y3_D2D1, "ward")
np.save("clusteringD1D2_2y3_D2D1.npy", clusteringD1D2_2y3_D2D1)
```

```
In [10]:
nombrescluster = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J"]
```

```
In [11]:
#Definiendo composicion de cluster segun limite de cantidad de cluster formados
```

```
k = 10

clustersD1D2_D2D1_10cluster = fcluster(clusteringD1D2_2y3_D2D1, k, criterion='maxclust')
#Counter(clustersD1D2_D2D1_10cluster)

DiccPosiEstruD1D2 = {}
for n,i in enumerate(cluster2y3):
    DiccPosiEstruD1D2[n]=i

DiccClusters_D1D2_D2D1_10cluster_sinordenar = {}
for j in range(1,k+1):
    DiccClusters_D1D2_D2D1_10cluster_sinordenar[j] = [DiccPosiEstruD1D2[n] for n,x in
enumerate(clustersD1D2_D2D1_10cluster) if x == j]

np.save("DiccClusters_D1D2_10cluster_sinordenar.npy", DiccClusters_D1D2_D2D1_10cluster_sinordenar)
```

```
In [12]:
DiccClusters_D1D2_D2D1_10cluster_sinordenar.keys()
```

```
 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
In [13]:
#Ordenar los clusters según cantidad de estructuras que contienen
for i in range(1,11):
    globals()["largo%s"%i] = len(DiccClusters_D1D2_D2D1_10cluster_sinordenar[i])
largos = [largo1, largo2, largo3, largo4, largo5, largo6, largo7, largo8, largo9, largo10] #Curado a
mano
largosordenados = sorted(largos)
print largosordenados

indicesLargosordenados = []
for n in [9,8,7,6,5,4,3,2,1,0]:
    indicesLargosordenados.append([x for x in DiccClusters_D1D2_D2D1_10cluster_sinordenar.keys() if
                        len(DiccClusters_D1D2_D2D1_10cluster_sinordenar[x]) == largosordenados[n]])
print indicesLargosordenados

DiccClusters_D1D2_D2D1_10cluster = {}
for i in range(1,11):

DiccClusters_D1D2_D2D1_10cluster[i]=DiccClusters_D1D2_D2D1_10cluster_sinordenar[indicesLargosordenados
[i-1][0]]

np.save("DiccClusters_D1D2_10clusterOrdenados.npy", DiccClusters_D1D2_D2D1_10cluster)
```

```
[155, 233, 477, 664, 709, 1057, 1099, 1196, 1789, 2042]
[[1], [2], [10], [8], [4], [3], [7], [9], [5], [6]]
```

Cargar valores experimentales y seleccionar regiones para comparar con los clusters

```
In [14]:
#Cargar los valores de II0 experimentales
expFCM20 = open("FCM20_IvsI0_CaColor_160816.txt").readlines()
expFCM21 = open("FCM21_IvsI0_CaColor_160920.txt").readlines()
```

```
In [30]:
resiFCM20_exp = [int(x.split()[0])+1 for x in expFCM20]
II0FCM20_exp = [float(x.split()[1]) for x in expFCM20]

resiFCM21_exp = [int(x.split()[0])+1 for x in expFCM21]
II0FCM21_exp = [float(x.split()[1]) for x in expFCM21]
```

Extrear de los PRE calculados los residuos para los cuales hubo medidas experimentales

```
In [15]:
II0FCM20_exp_D1 = []
for x in range(len(II0FCM20_exp)):
    if resiFCM20_exp[x] in range(3,73) :
        II0FCM20_exp_D1.append(II0FCM20_exp[x])

resiFCM20_exp_D1 = [x for x in resiFCM20_exp if x in range(3,73)]

print len(II0FCM20_exp_D1), len(resiFCM20_exp_D1)
```

```
49 49
```

```
In [16]:
II0FCM21_exp_D2 = []
for x in range(len(II0FCM21_exp)):
    if resiFCM21_exp[x] in range(104,174):
        II0FCM21_exp_D2.append(II0FCM21_exp[x])

resiFCM21_exp_D2 = [x for x in resiFCM21_exp if x in range(104,174)]

print len(II0FCM21_exp_D2), len(resiFCM21_exp_D2)
```

```
42 42
```

```
In [17]:
II0D1D2_D2D1_resiexp={}
for i in todosnombres:
    II0D1D2_D2D1_resiexp[i]=[]
    for n in resiFCM21_exp_D2:
        II0D1D2_D2D1_resiexp[str(i)].append(II0D1D2[str(i)][n-1])
    for n in resiFCM20_exp_D1:
        II0D1D2_D2D1_resiexp[str(i)].append(II0D1D2[str(i)][n-1+193+15])
```

```
In [18]:
#Por cluster
for i in range(1,11):
    globals()["II0D1D2_D2D1_resiexp_cluster%s"%(str(i).zfill(2))] = {}
    for j in DiccClusters_D1D2_D2D1_10cluster[i]:
        globals()["II0D1D2_D2D1_resiexp_cluster%s"%(str(i).zfill(2))][j] = II0D1D2_D2D1_resiexp[j]
```

```
In [19]:
II0FCM21_exp_D2_FCM20_exp_D1 = II0FCM21_exp_D2 + II0FCM20_exp_D1
print len(II0FCM21_exp_D2_FCM20_exp_D1)
```

```
91
```

Divergencias entre PRE de cada estructura simulada y los resultados experimentales

```
In [20]:
for i in range(1,11):
    globals()["diccDistD1D2_calc_D2D1_cluster%s"%(str(i).zfill(2))] = {}
    for n in globals()["II0D1D2_D2D1_resiexp_cluster%s"%(str(i).zfill(2))].keys():

globals()["diccDistD1D2_calc_D2D1_cluster%s"%(str(i).zfill(2))][n]=distancia(np.array(II0FCM21_exp_D2_
FCM20_exp_D1), np.array(II0D1D2_D2D1_resiexp[n]))
```