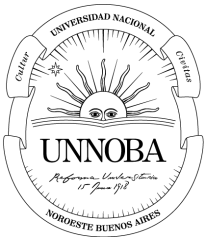


Desarrollo de una herramienta bioinformática para el estudio de coevolución en familias de proteínas

Eloy Adonis Colell

Universidad Nacional del Noroeste de la Provincia de Buenos Aires
Escuela de Agrarias, Naturales y Ambientales
Pergamino, Argentina
2017

Desarrollo de una herramienta bioinformática para el estudio de coevolución en familias de proteínas

Eloy Adonis Colell

Tesis presentada como requisito parcial para optar al título de:

Master en Bioinformática y Biología de Sistemas

Directora:
Doctora Cristina Marino-Buslje

Co-Director:
Doctor Franco Lucio Simonetti

Grupo de Investigación:
Unidad de Bioinformática. Fundación Instituto Leloir

Universidad Nacional del Noroeste de la Provincia de Buenos Aires
Escuela de Agrarias, Naturales y Ambientales
Pergamino, Argentina
2017

Agradecimientos

A mi directora, Cristina Marino-Buslje, por haberme elegido para llevar adelante este proyecto. A mi co-director, Franco Lucio Simonetti, por guiarme activamente en el desarrollo de esta tesis. A integrantes del Instituto Leloir, Diego Javier Zea, y Javier Iserte que colaboraron para reducir la cantidad de viajes hacia Capital Federal. A Lucas Capalbo Lavezzo, por ayudarme con revisiones y correcciones.

A mis padres Eva y Osvaldo, y a mis suegros Ada y Ernesto, por su apoyo, por su esfuerzo.

A Eugenia, mi compañera de aventuras. A mi hijo Lorenzo, el motivo de mi vida.

Resumen

Los alineamientos múltiples de secuencias (MSA) son agrupamientos de proteínas relacionadas evolutivamente entre sí que nos permiten inferir información a cerca de la familia proteica. Un tipo de información está dada por la conservación de aminoácidos en ciertas posiciones, mientras que la otra habla sobre la interrelación o coevolución entre dos o más posiciones. La coevolución de aminoácidos está íntimamente ligada a la función biológica de la proteína. La coevolución entre sitios se infiere utilizando métodos estadísticos que estiman la covariación entre posiciones a partir de un alineamiento múltiple de secuencias (MSA). En esta tesis, se desarrolló una herramienta web completa llamada MISTIC 2, de acceso libre y gratuito que permite a usuarios no expertos explorar diversos aspectos de una proteína y/o de una familia proteica. Permite integrar información evolutiva, secuencial y estructural. El principal resultado es el cálculo de covariación entre residuos de una proteína, con diferentes software actuales como: los derivados de la teoría de la información, información mutua (MIp) y nuevos métodos basados en información directa (Direct Coupling Analysis) como son el mfDCA, plmDCA and gaussianDCA. Un aspecto fundamental es que permite analizar los resultados de una manera gráfica, amigable e interactiva. Los resultados se presentan en una interfaz que permite explorar la red de covariación e integrarla con la estructura 3D de la proteína. Al mismo tiempo la arquitectura se encuentra adaptada para incorporar nuevos algoritmos que sean desarrollados en un futuro cercano.

El servidor provee una Application Programming Interface (API) y una interfaz de usuario gráfica. La interfaz programable retorna los resultados crudos, mientras que la interfaz gráfica presenta varias perspectivas de visualización y un pequeño post-procesamiento de los resultados que permite explorar de forma iterativa los resultados obtenidos. MISTIC 2 provee una visión integrada del MSA en términos de (i) puntajes de covariación entre pares de residuos (MI, mfDCA, plmDCA y gaussianDCA), (ii) conservación de secuencias y (iii) y la covariación acumulada en un área en la estructura tridimensional de una proteína (por residuos próximos en la estructura).

Además, una característica principal, es que permite la comparación de todos los métodos al mismo tiempo y que se pueden integrar múltiples resultados de covariación en un score combinado. Se proveen múltiples formas para seleccionar un subconjunto de nodos (o residuos) de una red que se visualizarán automáticamente en la estructura 3D.

Por último, a modo de ejemplo, se realiza un estudio sobre la familia de dominios *7 transmembrane receptor (rhodopsin family)* que forman parte de los receptores asociados a la proteína G.

Contenido

Indice de Figuras

Indice de Tablas

Abreviaturas

API	Application Programming Interface
AUC	Area Under the Curve
CCMpred	Correlated Mutations predicted quickly and accurately
gaussianDCA	Gaussian Direct Coupling Analysis
GPCR	G Protein Coupled Receptor
JSON	JavaScript Object Notation
mfDCA	Mean Field Direct Coupling Analysis
MI	Mutual Information
MISTIC	Mutual Information Service To Infer-Coevolution
MSA	Multiple Sequence Alignment
plmDCA	Pseudo-likelihood maximization Direct Coupling Analysis
ROC	Receiver Operating Characteristic

1 Introducción

El concepto de Coevolución fue definido por primera vez por Janzen (Janzen 1980), como aquel proceso por los cuales dos o más organismos ejercen presión de selección mutua y sincrónica, en tiempo geológico, descartando su uso como sinónimo de mutualismo, interacción o simbiosis.

Luego, en términos macroscópicos, una de las definiciones más ampliamente aceptadas fue dada por Thompson (Thompson 1994) como cambios evolutivos que ocurren de manera recíproca entre especies interactuantes. Esto se refiere a que el cambio en una población modifica la presión de selección sobre una segunda población y viceversa, siendo un ejemplo muy estudiado a nivel de organismos la coevolución patógeno-huésped.

De manera semejante, la evolución de los aminoácidos dentro de una proteína no es independiente. Existen interacciones como los puentes salinos entre residuos cargados, puentes de hidrógeno entre residuos aceptores y donadores de electrones, restricciones de tamaño que reflejan la interacción de cadenas laterales grandes y pequeñas, interacciones electrostáticas, efecto hidrofóbico, fuerzas de Van der Waals y otros fenómenos (Atchley et al. 2000). Estas interacciones dejan una firma evolutiva en la secuencia que puede verse en la covariación entre aminoácidos de una proteína que es lo que se denomina coevolución sitio específica.

Aunque existen diferentes propuestas acerca del mecanismo exacto por el cual las posiciones coevolucionan, Yanofsky et al. (Yanofsky et al. 1964) y Fitch y Markowitz (Fitch & Markowitz 1970) estuvieron de acuerdo en que la coadaptación de aminoácidos en las secuencias era importante para mantener la estructura y función apropiada de la proteína. Las posiciones que coevolucionan se reflejan como posiciones que covarían en familias de proteínas (Pazos & Valencia 2008).

La función y la conformación espacial de las proteínas imponen restricciones en las variaciones de residuos que pueden ser aceptados para que una proteína siga siendo funcional. Dado esto, es esperable que los sitios funcionales o los que se

encuentran próximos en la estructura tridimensional sean los que impongan restricciones en los cambios unos a otros, es decir que cambian de una manera concertada o coevolucionan (Aguilar et al. 2012, Altschul et al. 1997, Atchley et al. 2000, Gloor et al. 2005, Halabi et al. 2009) (ver Figura 1).

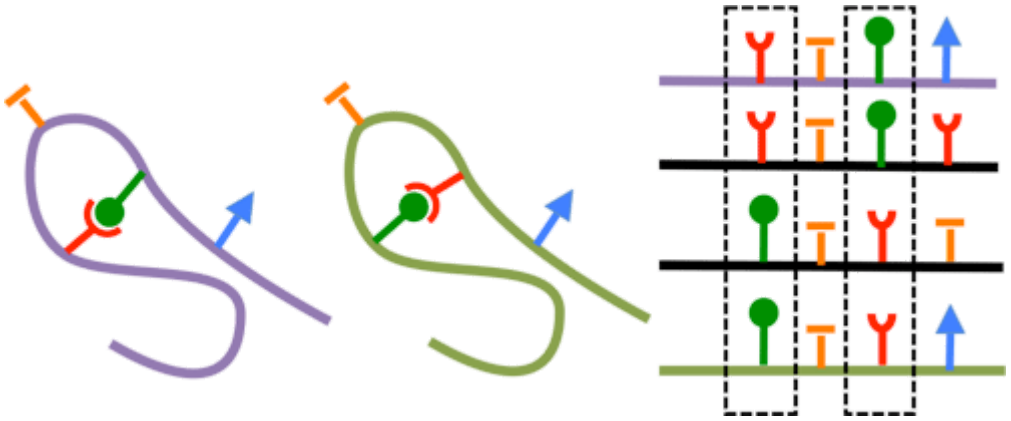


Figura 1: Representación esquemática de la coevolución sitio específica. A la izquierda se representan aminoácidos con formas complementarias (verde y rojo), que han intercambiado sus posiciones en la cadena polipeptídica. A la derecha se representa el alineamiento múltiple de secuencias correspondiente donde se puede observar la covariación de residuos en las posiciones marcadas en línea punteada. Imagen extraída de http://gremlin.bakerlab.org/gremlin_faq.php.

Los aminoácidos funcionalmente relacionados se encuentran vinculados evolutivamente ya que una mutación en alguno de ellos puede causar un efecto dramático en la función y en la presión de selección sobre otras posiciones de aminoácidos. En estos casos para que una mutación sea fijada, son necesarias mutaciones compensatorias en otro/s sitio/s para mantener o restaurar la función.

1.1 Coevolución y Covariación

Es importante distinguir entre la mera covariación de residuos, que puede observarse en un Alineamiento Múltiple de Secuencias (MSA, del inglés Multiple Sequence Alignment) y la coevolución. Mientras que la primera se refiere al cambio simultáneo observado de aminoácidos sin importar las causas de estos cambios, la

coevolución implica que el cambio sea recíproco y debido a un proceso evolutivo común (Codoñer & Fares 2008).

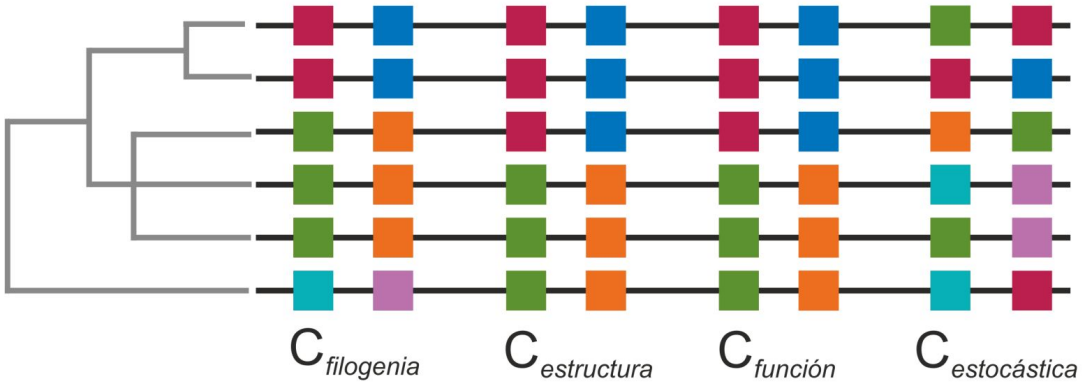


Figura 2: Componentes de la señal de coevolución.

Atchley et al. (Atchley et al. 2000) sugirieron que la covariación observada entre las posiciones i y j en una proteína está compuesta de una señal proveniente de restricciones estructurales y funcionales, junto con ruido de fondo agregado por una filogenia compartida y eventos estocásticos. De esta forma, las señales estructurales y funcionales están superpuestas sobre el ruido filogenético y otros procesos aleatorios. Así la covariación entre una posición i y j en un MSA puede ser descompuesta en diferentes términos como se muestra en la Ecuación 1 y se esquematiza en la Figura 2.

$$C_{ij} = C_{filogenetica} + C_{estructura} + C_{funcion} + C_{estocastica} \quad (1)$$

La covariación filogenética ($C_{filogenetica}$) señala la dependencia evolutiva entre las secuencias y esta dependencia es extensible a nivel de residuos proteicos. Los componentes $C_{estructura}$ y $C_{funcion}$ dan cuenta de la covariación debida a un proceso evolutivo común para mantener estructura y función (Atchley et al. 2000) y son difíciles de distinguir ya que no son mutuamente excluyentes, son grupos de aminoácidos que coevolucionan debido a una combinación de diferentes dependencias. La covariación estocástica ($C_{estocastica}$) puede deberse a variación independiente de los sitios debido a su dinámica de mutación que se asemeja a la señal de covariación o coevolución. Las dificultades para quitar este componente se deben a las limitaciones para producir un modelo de análisis para dar cuenta de la

covarianza estocástica, por lo que muchos métodos se basan en simulaciones de MSAs. Estos MSAs simulan los parámetros evolutivos de alineamientos biológicos y se pueden usar para producir una distribución de las probabilidades de detectar coevolución bajo un cierto modelo de sustitución de aminoácidos (Dunn et al. 2008). La identificación de la coevolución estocástica está muy condicionada por las propiedades estadísticas del MSA. Los MSAs de baja calidad y de pocas secuencias son más propensos a producir sitios falsos de coevolución (Fares & Travers 2006) es decir, es difícil separar el ruido de la señal.

Esta percepción sobre la composición de la covariación produjo un cambio en cómo las posiciones que covarían son identificadas debido a que, por primera vez, se definieron las limitaciones intrínsecas a la detección de covariación.

Es aceptado que estimaciones precisas de covariación sólo pueden ser generadas por un gran número de secuencias no redundantes en el MSA (Buslje et al. 2009, Lockless & Ranganathan 1999, Martin et al. 2005, Tillier & Lui 2003), aunque no hay un acuerdo común sobre cuál es el límite inferior. También está claro que es crucial reducir o remover la señal de covariación causada por la relación filogenética de las secuencias analizadas (Buslje et al. 2009, Dunn et al. 2008, Little & Chen 2009). Estos últimos métodos están todos basados en la idea de que el producto del promedio de covariación entre las posiciones i y j con todas las otras posiciones es un estimativo preciso de la señal filogenética de fondo (Dunn et al. 2008).

1.2 Información Mutua

El concepto de Información Mutua (MI, del inglés Mutual Information) proveniente del área de la Teoría de la Información, puede ser utilizado para estimar el grado de relación o coevolución entre dos posiciones de una familia de proteínas (Gloor et al. 2005, Martin et al. 2005, Tillier & Lui 2003). En términos biológicos, para el análisis de secuencias, la MI entre dos posiciones (dos columnas de un MSA) refleja la disminución de la incertidumbre en una posición a partir del conocimiento de otra posición. Intuitivamente, la MI mide la información compartida por dos columnas de un MSA. Dadas dos posiciones (o columnas) i y j cualesquiera en un MSA, la MI entre ellas está dada por la relación que se muestra en la Ecuación 2.

$$MI_{ij} = \sum_{(a,b)}^{(AA \times AA)} P(a_i, b_j) \cdot log\left(\frac{P(a_i, b_j)}{P(a_i) \cdot P(b_j)}\right) \quad (2)$$

Donde $P(a_i, b_j)$ es la frecuencia de aparición del aminoácido a en la posición i y el aminoácido b en la posición j en la misma secuencia; $P(a_i)$ es la frecuencia del aminoácido a en la posición i, $P(b_j)$ es la frecuencia de aparición del aminoácido b en la posición j en el alineamiento. Realizándose la sumatoria para cada par de aminoácidos (a, b) del conjunto de pares $(AA \times AA)$ posibles para las posiciones i y j (siendo AA el conjunto de los 20 aminoácidos estándar).

Un valor de MI igual a cero indica independencia evolutiva entre los sitios i y j. Dos posiciones invariables (conservadas) representan un caso extremo de co-aparición, y la MI entre ellas es cero. Es decir, se asume independencia evolutiva al no tener evidencia de covariación entre ellas. Para los valores positivos de MI, su magnitud depende de la fuerza de covariación entre ambos sitios (Blahut 1987). Como consecuencia el poder de MI para predecir la coevolución real depende del nivel de conservación en el MSA (Fodor & Aldrich 2004, Martin et al. 2005).

La MI permite inferir sitios que varían de manera correlacionada en secuencias homólogas. Aunque en principio el cálculo de MI es simple, su interpretación biológica y evolutiva ha sido discutida y diferentes enfoques han testeado su utilidad (Cover & Thomas 2005, Del Sol et al. 2007, Dunn et al. 2008, Halabi et al. 2009). La ventaja del uso de MI para cuantificar la coevolución radica en la aplicabilidad del método sin requerir de conocimientos sobre la relación entre residuos en el MSA y su dinámica evolutiva. Los valores de MI se ven afectados por la historia filogenética común entre las secuencias ($C_{filogenetica}$), a menos que esa dependencia sea corregida explícitamente en el modelo de detección de coevolución. Así, un gran desafío para la detección de la coevolución fue la distinción de correlaciones filogenéticas del resto de las correlaciones. Muchos estudios han intentado corregir los valores de MI quitando el efecto de la dependencia filogenética (Buslje et al. 2009, Dunn et al. 2008, Dutheil 2012, Gouveia-Oliveira & Pedersen 2007, Tillier & Lui 2003).

La dependencia entre las posiciones de un MSA se debe a que las secuencias proteicas no son independientes, sino que contienen una señal inherente dada por su historia coevolutiva en común. Esto ha sido probado por Gouveia-Oliveira y Pedersen (Gouveia-Oliveira & Pedersen 2007) donde muestran cómo secuencias relacionadas pueden resultar en una señal de covariación que se parece a la coevolución. En la Figura 3 se esquematiza el fenómeno mostrando variaciones independientes que dan una señal que imita a la señal de coevolución.

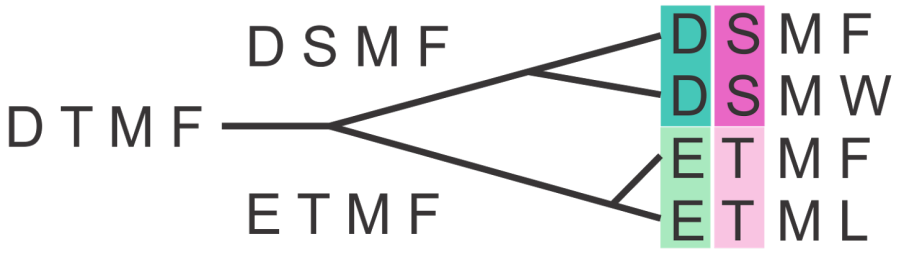


Figura 3: Señal filogenética que imita coevolución. Basada en la Figura 1 de (Gouveia-Oliveira & Pedersen 2007)

Dos mutaciones independientes pueden resultar en grupos de secuencias que muestran una señal que puede confundirse con coevolución. Las posiciones 1 y 2 del alineamiento presentan alta MI aunque no coevolucionan, ya que las mutaciones se produjeron de manera independiente en secuencias ancestrales y se fijaron, imitando coevoluciones en grupos de secuencias que observamos actualmente.

Se han propuesto diferentes abordajes para disminuir la señal impuesta por la filogenia a fin de mejorar la identificación de posiciones que coevolucionan. Una manera simple de tener en cuenta los ancestros compartidos es medir el grado de correlación que puede esperarse solamente por filogenia y estocasticidad. Esto se realiza mediante un procedimiento de randomización o *bootstrap*, el cual elimina cualquier correlación y permite obtener una estimación empírica de la distribución nula de las correlaciones (Buslje et al. 2009, Dunn et al. 2008).

Un obstáculo adicional para la detección de coevolución es la falta de un set de datos de referencia construido a partir de datos reales para evaluar los métodos, ya

que nunca se accede a la historia de coevolución real en los datos biológicos. Como aproximación se asume que los pares de residuos que están en contacto (por ejemplo menos de 6.05Å entre átomos pesados) coevolucionan (Buslje et al. 2009, Del Sol et al. 2007, Gouveia-Oliveira & Pedersen 2007). Es evidente que es una aproximación a la realidad, ya que existen muchos sitios que están en contacto y no coevolucionan, y a la vez muchos sitios que coevolucionan no necesariamente están en contacto directo. Sin embargo para la mayoría de los pares de aminoácidos que coevolucionan se puede asumir que están en contacto, considerando que pueden influenciarse mutuamente de manera directa.

La sensibilidad de la mayoría de los métodos desarrollados para detectar coevolución utilizando MI depende de ciertas características del MSA como ser: (a) su calidad; (b) el número de secuencias; y (c) el nivel de divergencia.

1.3 Correcciones al algoritmo de Información Mutua

Dos ejemplos de correcciones para afrontar el problema de la señal filogenética y la entropía inherente a cada columna de un MSA son, el *Row Column Weighting* (RCW) (Gouveia-Oliveira & Pedersen 2007) y el *Average Product Correction* (APC) (Dunn et al. 2008). El método de RCW divide el valor de MI entre dos posiciones por el promedio de MI que tienen esas posiciones con cualquier otra, como indica la Ecuación 3.

$$RCW = \frac{MI_{ij}}{(\overline{MI_{i\cdot}} \cdot \overline{MI_{i\cdot y}})/2} \tag{3}$$

Donde $\overline{MI_{i\cdot}}$. es el valor promedio de MI entre la posición i y el resto de las posiciones (.). De manera análoga se define $\overline{MI_{j\cdot}}$.

El método Average Product Correction sustrae el término definido como APC para generar un puntaje de MI corregida como se muestra en las Ecuaciones 4 y 5.

$$MI_{APC} = MI_{(i,j)} - APC_{(i,j)} \tag{4}$$

$$APC_{(i,j)} = \frac{\overline{MI_{i.}} \cdot \overline{MI_{j.}}}{\overline{MI_{..}}} \quad (5)$$

Donde $\overline{MI_{..}}$ es el promedio del valor de MI sobre todos los pares de posiciones del MSA. En el trabajo de Buslje et al. (Buslje et al. 2009) se ha comparado el desempeño predictivo de ambas correcciones, RCW y APC en dos sets de datos, uno compuesto por secuencias biológicas y otro construido con datos artificiales por Gouveia-Oliveira et al. (Gouveia-Oliveira & Pedersen 2007). Se demostró que el método con mejor performance incluye la corrección APC. Además se han propuesto dos nuevas correcciones que mejoran aún más el desempeño del método: el pesado de secuencias y la corrección por bajo número de observaciones (pseudo-cuentas). A la vez es común introducir una permutación de secuencias para el cálculo de MI que permite cambiar la escala del puntaje de MI y hacerlo comparable entre diferentes familias de proteínas. El algoritmo de MI utilizado en el presente trabajo proviene del desarrollo en (Buslje et al. 2009). Estas correcciones se explican a continuación.

1.3.1 Pesado de secuencias

Al trabajar con datos biológicos, frecuentemente los MSAs presentan un sesgo no natural y redundancia de secuencias, por ejemplo, debido a la secuenciación en múltiples cepas bacterianas o la selección de especies a secuenciar (por ser, por ejemplo, especies de mayor interés). Por lo tanto es esperable que un algoritmo de clusterización que reduzca la redundancia de secuencias mejore la performance predictiva del cálculo de MI. Se empleó el algoritmo Hobohm-1 (Hobohm et al. 1992) para definir grupos de secuencias. Los grupos de secuencias generados dependen del valor de corte de identidad de secuencia seleccionado (de 0 a 100%). En Buslje et al. (Buslje et al. 2009) y Shackelford et al. (Shackelford & Karplus 2007) se encontró un valor de corte óptimo de 62% de identidad para el clustering. Luego se le asignó un peso a cada secuencia, con un valor igual a la inversa del número de secuencias del grupo al que pertenece.

1.3.2 Corrección por pseudo-cuentas

Para MSAs con bajo número de secuencias, el cálculo de la ocurrencia de aminoácidos en una posición dada es estimado a partir de un bajo número de observaciones, y su contribución al cálculo de MI puede ser ruidosa. Por otro lado, una combinación (un par) de aminoácidos que no se observa en un MSA puede deberse meramente a que el muestreo no fue suficiente.

Por ello se introduce una corrección para el bajo número de observaciones. La probabilidad de los aminoácidos, $P(a_i, b_j)$, se calcula a partir de $N(a, b)$, el número de veces que el par (a, b) es observado en las posiciones i y j en el MSA más una constante λ. La frecuencia del aminoácido a en la posición i y del aminoácido b en la posición j se calcula como se muestra en la Ecuación 6 y que luego son incorporados a la Ecuación 2.

$$P(a_i, b_j) = \frac{\lambda + N(a_i, b_j)}{N} \tag{6}$$

Donde

$$N = \sum_{a,b} (\lambda + N(a_i, b_j)) \tag{7}$$

$$P(a_i) = \sum_{b} P(a_i, b_j) \tag{8}$$

$$P(b_j) = \sum_{a} P(a_i, b_j) \tag{9}$$

Se le da un valor inicial de $N(a_i, b_j) = \lambda$ para todo par de aminoácidos. Es decir, todo par posible de aminoácidos será observado λ veces. Solamente para MSAs con un bajo número de secuencias, donde una gran fracción de pares de aminoácidos no son observados, el parámetro λ influencia el cálculo de probabilidades. Contrariamente, para MSAs con un gran número de secuencias, la mayoría de los pares de aminoácidos serán observados al menos una vez, y la influencia de λ será menor. Se evaluó el desempeño del método a diferentes

valores de λ en el rango [0−2]. Se utiliza un λ de 0.05, encontrándose resultados similares en el rango 0.025−0.075. En síntesis, se inicializa una matriz con un valor de 0.05 para cada par de aminoácidos posible y sobre este valor se agregan las frecuencias observadas. Así, cada par de aminoácidos posible tiene la posibilidad de ser observado al menos con frecuencia λ. El valor óptimo se encontró de manera consistente siendo independiente del tamaño del set de datos usado, el modelo evolutivo y la tasa de evolución (Buslje et al. 2009).

1.3.3 Transformación a Z-score

Cada valor de MI entre un par dado de posiciones es comparado con la distribución de valores de MI obtenidos a partir de un grupo de MSAs aleatorizados. El Z-score es calculado como el número de desviaciones estándar que el valor de MI observado se aparta de la media obtenida con los MSAs aleatorios. Para cada MSA se realizaron 100 permutaciones, manteniendo los gaps fijos en sus posiciones originales. Se ensayaron dos métodos de permutación, uno basado en columnas (aleatorización vertical) y otro basado en secuencia (aleatorización horizontal). El primero desafía la hipótesis de que las secuencias son homólogas y están correctamente alineadas, pero que las columnas no están correlacionadas. Mientras que el segundo método desafía la hipótesis de que las secuencias no son homólogas. El mejor resultado fue obtenido con el segundo método de permutación (Buslje et al. 2009), por lo que el Z-score no testea de manera adecuada la hipótesis nula (que las columnas no sean correlacionadas) siendo adecuado interpretarlo sólo como un puntaje predictivo más, que permite la comparación entre familias de proteínas. De aquí en adelante, se referirá al puntaje Z-score de MI corregido como simplemente puntaje de MI.

1.4 Puntajes derivados de MI: cMI y pMI

El cálculo de MI genera un puntaje para cada par de columnas del MSA. Se derivaron dos puntajes por posición, uno que caracteriza la cantidad de información mutua que la posición o residuo acumula, basado solo en información secuencial, y otro que caracteriza la información que un residuo acumula en su proximidad física.

Estos puntajes se definieron en Teppa et al. (Teppa et al. 2012). Primero se calcula un puntaje de MI acumulada (cMI por Cumulative MI) para cada residuo como la suma de los valores de MI mayores a un valor de umbral en el que el residuo en particular participa, como se muestra en la Ecuación 10.

$$cMI_i = \sum_{i,MI(i,j)>u_p} MI(i,j) \tag{10}$$

Este puntaje de cMI captura cuánta información acumula cada residuo, pudiendo ser ésta debida a una o unas pocas covariaciones fuertes o que el aminoácido sea central en una red de información mutua, es decir, que covaríe con muchos otros (con valores mayor al umbral determinado u_p) (ver Figura 4).

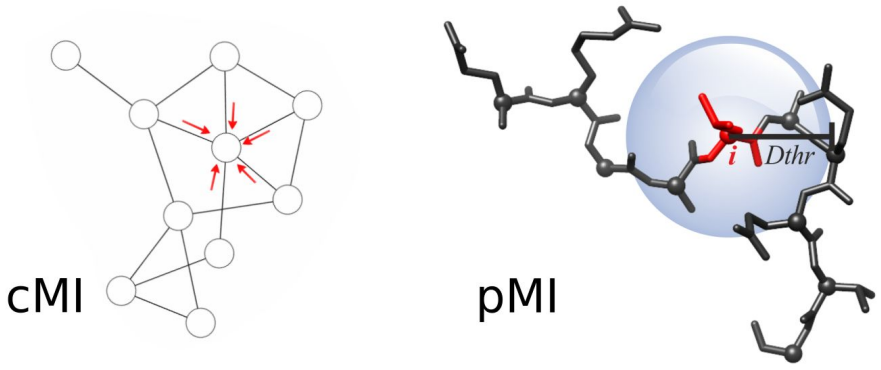


Figura 4: Ilustración de cMI y pMI. Esquema del cálculo de MI acumulado (cMI) (izquierda). Cada nodo acumula el puntaje de MI que tiene con sus primeros vecinos. Puntaje de MI en la proximidad (pMI) (derecha). Dthr: Distance threshold, valor de corte de distancia para el cálculo de pMI.

Luego, a partir del puntaje de cMI, se calculó la MI en la proximidad estructural (espacial) para cada residuo como el promedio de la cMI de todos los residuos que se encuentran en una esfera de distancia determinada. Finalmente se normalizaron los valores de proximidad para cada MSA para que los puntajes pertenezcan al rango [0-1], al que se llama puntaje por proximidad pMI, como se muestra en la Ecuación 11 y se esquematiza en la Figura 4.

$$pMI_i = \frac{1}{N} \cdot \sum_{j, d_{ij} < Dthr} cMI_j \tag{11}$$

Donde la suma se realiza sobre todos los residuos j en una proteína dada y una estructura de PDB dada, dentro de una distancia $d_{ij} < Dthr$. La distancia d_{ij} se calcula como la distancia mínima entre cualquier par de átomos, que no sean hidrógenos, entre los residuos i y j, cMI_j es la información mútua acumulada del residuo j, y $Dthr$ es la distancia umbral utilizada.

1.5 Métodos de Direct Information

Una de las hipótesis que asume el método de Mutual Information es que las posiciones i y j son independientes del resto de las posiciones. En el campo de la predicción de contactos por métodos de covariación, sin embargo, algunas posiciones pueden estar correlacionadas transitivamente, es decir, si un residuo en la posición i interacciona con uno en la posición k (una "interacción directa") y otro de la posición j con uno de la posición k ("interacción directa"), entonces los residuos en las posiciones i y j pueden dar covariación por los métodos de MI a pesar de no interaccionar directamente en la estructura ("interacción indirecta") (ver Figura 5).

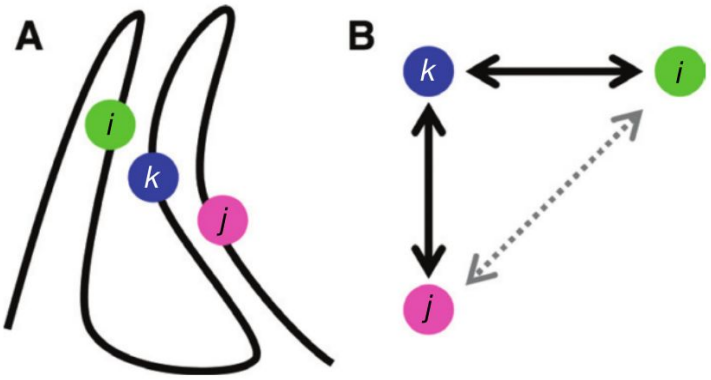


Figura 5: Efecto de covariación indirecta. A) Las posiciones i y k covarían y están en contacto en la estructura, al igual que los pares j y k. B) Debido al efecto de transitividad, los pares i y j pueden tener covariación y no estar en contacto. Imagen tomada de (Tetchner et al. 2014).

Esta correlación adicional aparece meramente como resultado de las dos interacciones reales que comparten con el residuo k. Se puede pensar que estos efectos indirectos pueden ser eliminados simplemente filtrando pares que covarían débilmente, pero esto es insuficiente, ya que la covariación "indirecta" puede llegar a mostrar covariación más fuerte que la covariación "directa" (Giraud et al. 1999).

El método de Direct Coupling Analysis (DCA) fue el primero en demostrar con éxito la implementación de un método global basado en el trabajo de Lapedes et al. (Lapedes et al. 1999) que sentó las bases teóricas 10 años antes (Weigt et al. 2009). DCA está basado en la observación de que las secuencias de proteínas en un MSA pueden ser representadas como un modelo de Potts de 21 estados (20 estados para aminoácidos y 1 para los gaps del alineamiento), el cual es capaz de separar couplings directos e indirectos. Cada modelo de Potts representa la probabilidad de observar una secuencia de aminoácidos:

$$P(A) = \frac{1}{Z} exp(\sum_{i=1}^{N} h_i(A_i) + \sum_{1 \leqslant i \leqslant j \leqslant N} J_{ij}(A_i, A_j)) \tag{12}$$

Donde $A = (A_1, A_2, ..., A_N)$ es una secuencia de aminoácidos de largo N, Z es una constante normalizadora (conocida como la función de partición), $h_i(A_i)$ es un término de cuerpo simple, y $J_{ij}(A_i, A_j)$ es un término de cuerpo doble. Los términos de cuerpo doble en el contexto de la secuencia de proteínas, pueden ser interpretados como las interacciones residuo-residuo o *couplings*, mientras que los términos de cuerpo simple dan cuenta de la conservación individual de los residuos. Calcular los *couplings* a partir de los modelos de Potts requiere un método de inferencia y una aproximación de la constante normalizadora Z, ya que esta función de normalización Z no es computable (Morcos et al. 2011).

La idea detrás de DCA se hizo posible a partir del uso de la aproximación de *mean-field* (Morcos et al. 2011). En mfDCA y EVfold (Kaján et al. 2014, Marks et al. 2011), se utiliza una aproximación de *mean-field* para estimar la función de partición Z de la Ecuación 12. A partir de esta ecuación se estiman los valores J_{ij} que indican la "información directa" (DI, del inglés Direct Information) entre pares de

residuos. El propósito de introducir DI es estimar la fuerza de “interacción” entre los pares de residuos, ayudar a rankear todos los posibles pares en términos de su probabilidad de ser observados, y eliminar las correlaciones transitivas.

El desarrollo más reciente en este área ha sido el uso de una aproximación de *pseudo-likelihood maximization (plm)* para la función de partición, y ha demostrado ser mejor para inferir los modelos de Potts que la aproximación de *mean-field* (Ekeberg et al. 2013, Kamisetty et al. 2013). El algoritmo más rápido y mejor optimizado para calcular DCA con pseudo-likelihood maximization es CCMpred, que implementa el enfoque de plmDCA y GREMLIN (Ekeberg et al. 2013, Kamisetty et al. 2013) en C y en CUDA, lo cual permite que sea altamente paralelizable utilizando placas gráficas (GPU).

1.6 Puntajes derivados: cS y pS

Teniendo en cuenta los puntajes derivados (cMI y pMI), construidos a partir del cálculo de MI, se generalizan dichos puntajes para otros algoritmos de covariación basados en Direct Information como mfDCA, gaussianDCA y CCMpred (ver Ecuación 13 y 14).

$$cS_i = \sum_{i,S(i,j)>u_p} S(i,j) \tag{13}$$

$$pS_i = \frac{1}{N} \cdot \sum_{j,d_{ij}<Dthr} cS_j \tag{14}$$

Al igual que en Mutual Information, sea $S(i,j)$ el valor del Score del algoritmo (MI, mfDCA, gaussianDCA o CCMpred), se establece que cS_i es el Score que captura cuánta información acumula el residuo i. Y luego, se puede establecer pS_i como el Score acumulado por proximidad espacial al residuo i.

Dada la diversidad de puntajes posibles, es necesario establecer una función de evaluación de la capacidad de predicción, para poder comparar los resultados obtenidos.

1.7 Evaluación de la capacidad de predicción

La curva ROC (del inglés Receiver Operating Characteristic) ilustra el desempeño de un clasificador binario como su discriminación variando los valores de cortes. La curva ROC se genera graficando la tasa de verdaderos positivos (TPR: True Positive Rate) contra la tasa de falsos positivos (FPR: false positive rate) variando el puntaje del predictor (ver Ecuación 15 y 16).

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{15}$$

$$FPR = 1 - Especificidad = \frac{FP}{N} = \frac{FP}{FP + TN} \tag{16}$$

Las curvas que se encuentran por encima de la diagonal representan métodos con cierto poder de discriminación, teniendo mayor poder cuando se acerca a la esquina superior izquierda del gráfico. Usualmente se calcula el área bajo la curva ROC (AUC) como un cuantificador global del desempeño del predictor, un clasificador aleatorio da un valor de AUC de 0.5, mientras que un clasificador perfecto corresponde a un AUC de 1.0, como se muestra en la Figura 6.

Donde TP, FP, TN y FN corresponden a verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos respectivamente, de acuerdo a un valor de corte determinado. El área bajo la curva ROC es utilizada para medir la habilidad global de un predictor para distinguir los positivos de los negativos. En la Figura 6 se grafican dos curvas ROC distinguiendo la diferencia entre AUC y AUC_{01}.

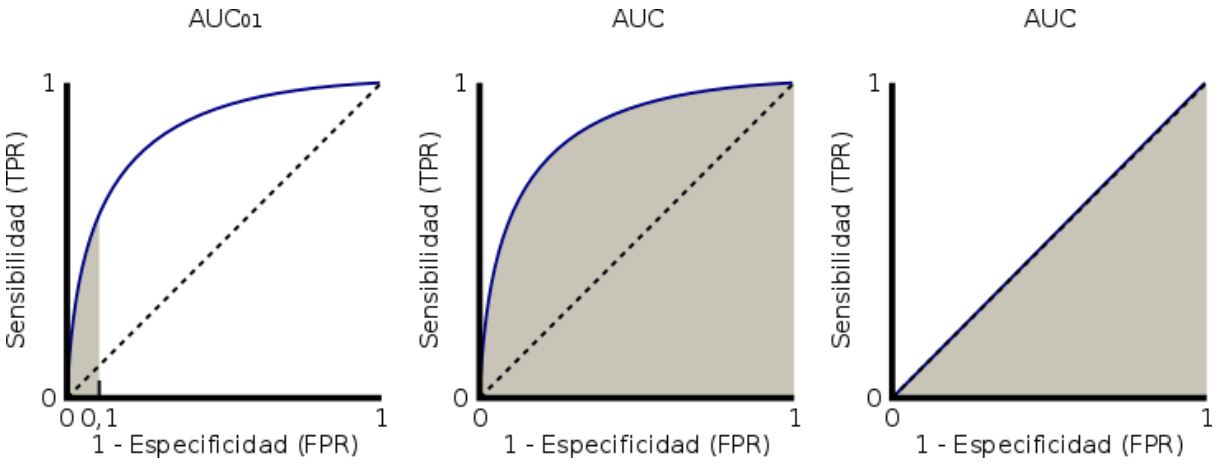


Figura 6: Curvas ROC. Las líneas punteadas representan la curva ROC para un predictor aleatorio. Si la curva toma el valor Y=1 para todos los valores de X, representa un método predictivo perfecto (AUC = 1), donde los valores de sensibilidad y especificidad son máximos. La curva azul representa un ejemplo de un método de clasificación real. El área gris debajo de la curva representa el AUC. En la imagen de la izquierda AUC_{01}: es el área resultante de la integración para una tasa de falsos positivos de 0.1 (10% FPR). A las de la derecha se representa el AUC completo.

1.8 Evolución de MISTIC

En el grupo de trabajo del Instituto Leloir, cuentan con un servidor para el cálculo de covariación denominado MISTIC 1 (Simonetti et al. 2013) y se encuentra disponible en http://mistic.leloir.org.ar. Aunque permite el cálculo de covariación con un solo método (MI), es una herramienta muy utilizada por la comunidad científica, desde su creación en el 2013 hasta la actualidad se han registrado cerca de 18.500 trabajos ("jobs"), enviados desde numerosos y diversos lugares del mundo (ver Figura 7) para el análisis de distintos problemas biológicos (Dellarole et al. 2015, McMurrough et al. 2014), siendo los mayores usuarios Estados Unidos, India y Argentina. En la actualidad MISTIC se encuentra obsoleto debido a factores científicos y técnicos.

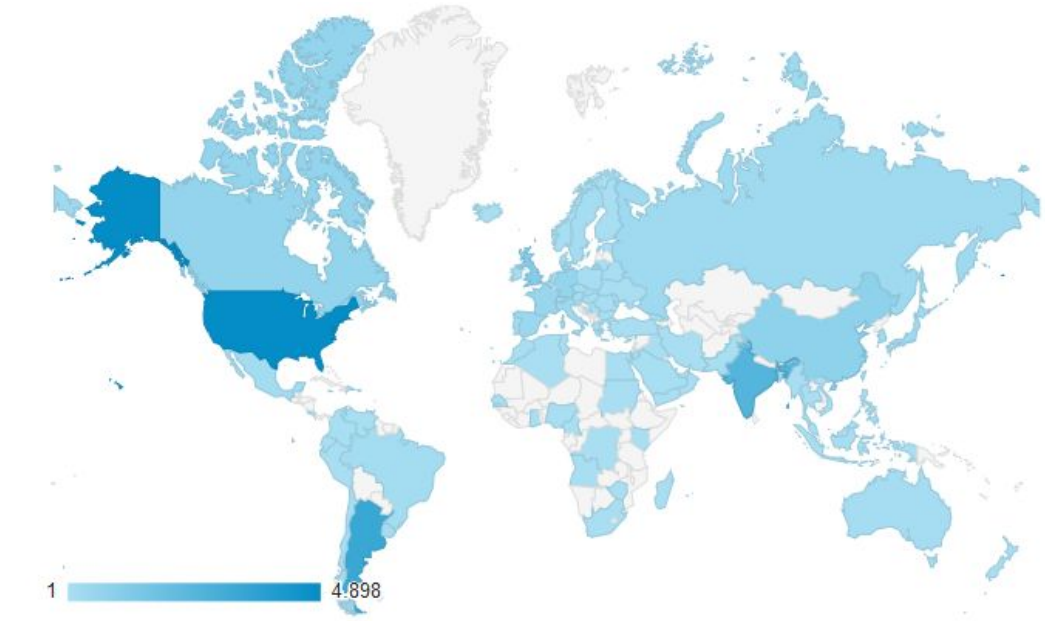
La necesidad y utilidad proporcionadas por MISTIC 1 nos impulsó a construir un nuevo servidor para responder a las nuevas necesidades académicas y a abordar los problemas técnicos observados desde su creación hace 4 años, donde las tecnologías han cambiado y avanzado a grandes pasos.

Desde el punto de vista académico:

- Se han desarrollado nuevos métodos para medir covariación. La nueva versión incorpora 3 nuevos métodos de cálculo: mfDCA; gaussianDCA y Gremlin (también conocido como CCMpred).
- Es necesaria la posibilidad de comparación de los 4 métodos y de obtener un *score* combinado de ellos.
- Es necesario mejorar la interacción con los resultados. La visualización no es trivial, en muchos casos permite simplificar el análisis y facilita el enfoque en la generación de hipótesis.

Desde el punto de vista técnico:

- Los plugins utilizados en MISTIC I requerían versiones de Java o la instalación de Adobe Flash, dos herramientas que han sido deprecadas en favor de librerías basadas en Javascript.
- Existen arquitecturas de un servidor que simplifican el mantenimiento y la modificabilidad del servicio. Usuarios avanzados se beneficiarán de la creación de una API para acceder programáticamente al cálculo de covariación sin necesidad de hacer uso de la interfaz gráfica.
- Existen nuevas estructuras de diseño de una aplicación web que optimizan la utilización del servidor, facilitan su mantenimiento y mejoran la experiencia del usuario.



		18,633 % of Total: 100.00% (18,633)	18,633 % of Total: 100.00% (18,633)
1.	United States	4,911	26.36%
2.	Argentina	3,450	18.52%
3.	India	2,497	13.40%
4.	United Kingdom	1,035	5.55%
5.	China	798	4.28%
6.	Canada	635	3.41%
7.	Germany	588	3.16%
8.	France	478	2.57%
9.	Spain	421	2.26%
10.	Russia	306	1.64%

Figura 7: Lugares desde donde se ha accedido a MISTIC 1. El mapa mundial muestra una visión general de las visitas realizadas por los distintos países. La tabla muestra el ranking de los 10 paises con mas consultas.

Por todo lo expresado, el desarrollo y puesta en marcha de la actualización MISTIC 2 es de suma importancia y de amplio impacto en la comunidad científica que utiliza el servicio. No solo se han solucionado problemas técnicos sino que también se pone al alcance de usuarios no expertos la capacidad de calcular covariación utilizando los últimos avances en el área, y más aún, la posibilidad de compararlos

entre sí, una tarea no trivial. La interfaz gráfica interactiva para visualizar los resultados juega un rol de suma importancia para su análisis y es esencial en la elaboración de hipótesis biológicas. MISTIC 2 juega un papel fundamental en la adopción de este tipo de cálculo como una herramienta básica para comprender la función y estructura de las proteínas, y que sin lugar a duda lo pondrá a la cabeza de los servidores de su tipo.

2 Materiales y Métodos

Con el foco puesto en herramientas estables y vigentes, se han seleccionado un subconjunto de ellas para construir pequeños componentes que darán la funcionalidad al servidor para calcular coevolución. Para el procesamiento y almacenamiento de los datos en el servidor se han utilizado tecnologías como MySQL, PostgreSQL, SQLAlchemy, Flask, RabbitMQ, Celery, entre otras; mientras que para la interacción con el usuario y la visualización de resultados en el cliente web se ha utilizado KnockoutJS, DataCollectionJs, PVmol y CytoscapeJs. A continuación, se realiza una breve descripción de cada una de ellas.

MySQL

MySQL es un sistema de base de datos reconocido por su versatilidad y simplicidad para desarrollar prototipos. Cumple con ACID (del inglés "*Atomicity, Consistency, Isolation and Durability*")[1], tiene soporte completo para claves foráneas y operaciones de *join* (Widenius & Axmark 2002).

PostgreSQL

PostgreSQL es un sistema de base de datos objeto-relacional ampliamente establecido en la comunidad (Yu & Chen 1995). Con más de 20 años de desarrollo activo, es posible ejecutarlo en una amplia variedad de sistemas operativos (como Linux, UNIX, y Windows). También cumple con ACID, tiene soporte completo para claves foráneas, *joins*, entre otros. Soporta los tipos de datos especificados por SQL:2008[2]. También permite almacenar grandes objetos en formato binario.

Base de Datos Pfam

Las proteínas están comprendidas por una o más regiones funcionales, comúnmente llamadas dominios. La presencia de diferentes dominios en diversas combinaciones en diferentes proteínas da lugar al repertorio diverso que se encuentra en la naturaleza. La identificación de los dominios presentes en una

[1] Ver más información en https://en.wikipedia.org/wiki/ACID.
[2] Ver mas información en https://en.wikipedia.org/wiki/SQL:2008.

proteína proporciona información sobre la función y la historia evolutiva de esa proteína.

La base de datos de Pfam es una gran colección de familias de dominios de proteínas (Finn et al. 2016). Cada familia está representada por un Alineamiento Múltiple de Secuencias (en inglés, MSA) y un Modelo Oculto de Markov (en inglés, HMM). La información se encuentra distribuida en 58 tablas interrelacionadas importables en una base de datos MySQL. Esta base de datos contiene los MSA para cada familia, junto con estructuras 3D que se conocen de todas las proteínas de esa familia provenientes de la base de datos PDB.

SQLAlchemy

SQLAlchemy es un paquete desarrollado en Python para manipular y gestionar bases de datos SQL (Myers & Copeland 2015), siendo su principal característica su capacidad de mapear objetos de Python en tablas de PostgreSQL o MySQL, convirtiendo esta relación en una relación persistente.

Provee un conjunto completo de herramientas con patrones de persistencia ampliamente utilizados, diseñado para acceder eficientemente a la información con mínimo consumo de recursos, adaptado para poder ser utilizado dentro del lenguaje Python de manera simple.

Servidor Web

Un servidor web es uno o varios programas informáticos que procesan información del lado del servidor a partir de las solicitudes realizadas por el cliente. La respuesta recibida por el cliente puede ser renderizada en un navegador web o procesada por otro programa o servidor diferente. Para la transmisión de todos estos datos suele utilizarse el protocolo HTTP.

El protocolo HTTP establece varios métodos de petición que un cliente puede utilizar. Los más comunes de utilizar son el HEAD, GET, POST, PUT y DELETE[3].

[3] Ver RFC 2616.

El método GET, solicita el contenido de un recurso. El método HEAD pide una respuesta idéntica a la que corresponde a una petición GET, pero en la respuesta no se devuelve el cuerpo del mensaje. Esto es útil para poder recuperar los metadatos, sin tener que transportar todo el contenido.

El método POST, envía los datos al servidor para que sean procesados por el recurso identificado. Los datos se incluirán en el cuerpo de la solicitud. Esto puede resultar en la creación de una nueva entrada en la base de datos del servidor. Por otra parte, método PUT, permite realizar una actualización de un recurso ya existente. Y por último, el método DELETE, que borra el recurso especificado.

Flask

Flask es un framework[4] minimalista escrito en Python que permite crear aplicaciones rápidamente para ser ejecutadas en servidores web. Está basado en la especificación WSGI[5] de Werkzeug y el motor de templates Jinja2.

Flask-Restful

Se torna muy común el caso en que el servidor web procesa y transmite una gran cantidad de contenido que no varía con distintas solicitudes del cliente. Por ello surgieron las conexiones asíncronas a los servidores que solo recuperan la información que cambia a partir de la solicitud del cliente, optimizando así el uso del servidor, y reutilizando el contenido estático previamente recibido en el cliente.

Los servidores manejan este tipo de conexiones asíncronas del cliente, a través de múltiples API (en inglés, “Application Programming Interface”).

Flask-RESTful es una extensión de Flask que agrega soporte para construir rápidamente una API REST (en inglés, “Representational State Transfer”) que establece que el servidor no mantiene datos entre las distintas solicitudes de un cliente. Flask-RESTful es una abstracción liviana que funciona compatible con SQLAlchemy y fomenta las mejores prácticas con una configuración mínima.

[4] Ver el significado de framework en https://en.wikipedia.org/wiki/Framework.
[5] Ver el significado de WSGI en https://en.wikipedia.org/wiki/Web_Server_Gateway_Interface

Flask-Restplus

Flask-RESTPlus es otra extensión de Flask que agrega soporte para construir APIs REST rápidamente. Al igual que Flask-RESTful fomenta las mejores prácticas con una configuración mínima. A diferencia de Flask-RESTful este provee una colección coherente de decoradores de Python y herramientas para describir una API que luego serán utilizados para generar la documentación de la interfaz.

RabbitMQ

RabbitMQ es el comunicador de mensajes entre procesos más ampliamente implementado. Además, de ser liviano y fácil de implementar en las instalaciones y en la nube, es compatible con múltiples protocolos de mensajería. RabbitMQ se puede implementar en configuraciones distribuidas y federadas para cumplir con los requisitos de alta disponibilidad y alta escala.

Celery

Celery es una cola de tareas asíncrona/cola de trabajos basado en el envío de mensajes distribuidos que funciona sobre RabbitMQ. Se centra en el funcionamiento en tiempo real, pero también admite la ejecución periódica.

Las unidades de ejecución, denominadas tareas, se ejecutan simultáneamente en uno o más servidores de trabajo. Las tareas se pueden ejecutar asíncronamente (mientras se ejecutan otras cosas) o de manera sincrónica (bloqueando al proceso que realizó la solicitud).

KnockoutJs

KnockoutJs es una biblioteca de JavaScript que facilita crear interfaces de usuario con un modelo de datos subyacente. Debido a su diseño es muy fácil refrescar pequeñas porciones de la interfaz gráfica de forma dinámica a partir de un pequeño cambio en el modelo de datos.

Utilizar KnockoutJs termina produciendo interfaces gráficas que son fáciles de mantener en el tiempo.

DataCollectionJs

DataCollectionJs permite manipular una colección de datos recibidos desde una API RESTful con facilidad.

Inspirado por los modernos administradores relacionales de objetos, DataCollection.js es una librería de JavaScript para almacenar, filtrar, transformar, y acceder a grandes colecciones de datos.

Pvmol

PV es un visor de JavaScript para visualizar estructuras de proteínas directamente en los navegadores. Es rápido y fácil de integrar en un sitio web y no requiere la instalación de ningún complemento. Es el visor de proteínas por defecto disponible en SWISS-MODEL[6].

Cytoscapejs

Cytoscape.js es una biblioteca para el análisis y visualización de redes (o grafos) escrita en JavaScript (Franz et al. 2016). Esta permite una visualización y manipulación dinámica e interactiva, a través de eventos en las interfaces de escritorio y gestos en las interfaces móviles.

[6] Ver http://swissmodel.expasy.org/

3 Desarrollo del servidor

A partir de la experiencia proporcionada por MISTIC 1, se detectaron distintas oportunidades para mejorar la arquitectura, y optimizar algunos procesos.

Como se menciona en la Introducción, desde el lanzamiento de MISTIC 1 a la fecha se han desarrollado nuevos métodos para estimar coevolución; los métodos de *Direct Information.* De esta manera, es lógico pensar que el primer paso en el desarrollo del nuevo servidor MISTIC es agregar la capacidad de calcular coevolución con estos nuevos algoritmos. MISTIC 2, no solo permite configurar y ejecutar diferentes algoritmos de correlación (MI, CCMpred, mfDCA, y gaussianDCA) para poder compararlos, sino que también se realizó una separación entre el procesamiento de los algoritmos, la visualización, y la obtención de los datos desde la base de datos de Pfam.

Separar el procesamiento a partir de una API, permite establecer un medio de comunicación no solo con nuestra interfaz gráfica, sino que cualquier otro servidor o interfaz gráfica cuenta con la posibilidad de acceder a los cálculos y herramientas desarrollados. Con el mismo criterio, mantener un servicio local con la base de datos Pfam nos permite acceder de forma más eficiente a la información sobre dominios, alineamientos y estructuras. Además, esta separación permitirá actualizar cada una de las partes de forma independiente una de la otra, reduciendo las dimensiones de futuras actualizaciones (ver Figura 8).

A continuación se describe la arquitectura que brinda acceso a la base de datos de Pfam, luego se proseguirá con la arquitectura del procesamiento de los datos y el cálculo de covariación y por último se explicará la arquitectura de visualización de resultados de la interfaz gráfica.

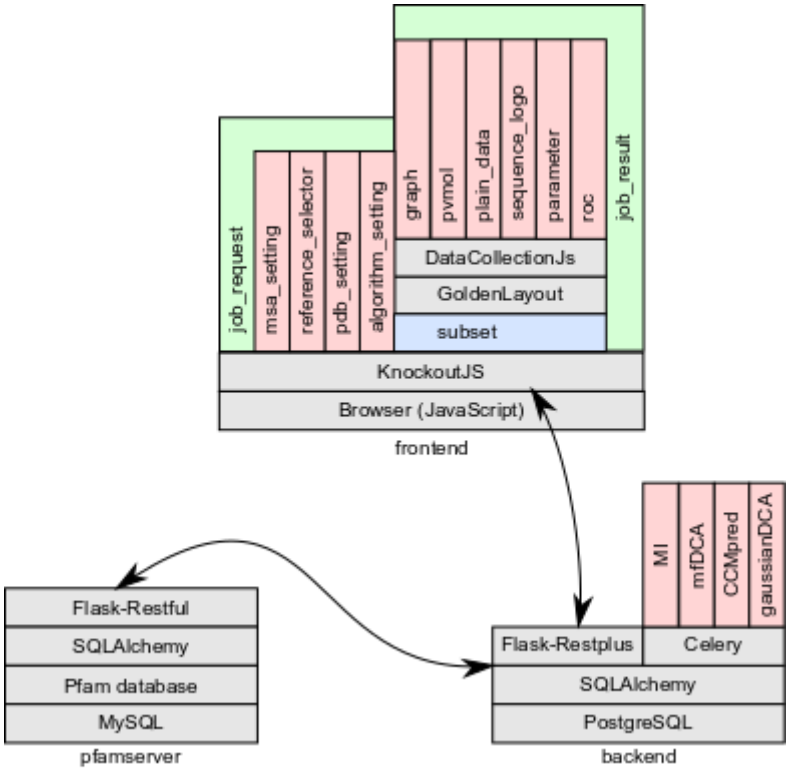


Figura 8: Arquitectura general. El frontend se conecta al backend, para simplificar la carga de los datos y representar los resultados, mientras que el backend accede al pfamserver para obtener información acerca de un dominio y los PDBs conocidos.

3.1 Arquitectura de acceso a los datos Pfam

Al analizar el proceso requerido por un usuario para cargar los datos necesarios para calcular coevolución, se han realizado pequeñas optimizaciones para simplificar aún más esta interacción.

El servidor MISTIC requiere como dato mínimo un MSA. Éste puede ser suministrado por el usuario como un archivo o puede utilizarse la base de datos de Pfam para sugerir un MSA a partir de una secuencia o identificador de interés del usuario. Anteriormente, el proceso para obtener una sugerencia de Pfam utilizando MISTIC consistía en ingresar a una página secundaria y colocar una secuencia o identificador.

En esta nueva versión existe un solo campo en donde cargar el MSA, dicho campo acepta una gran conjunto de formatos de archivos, pero también es capaz de reconocer cuando en lugar de un archivo se utiliza un identificador de Uniprot o Pfam.

En este último caso, el servicio de MISTIC 2 reconocerá el formato de dicho identificador, e intentará brindar de forma inmediata más información relacionada con dicha familia de proteína como una descripción y la cantidad de secuencias que contiene.

Esta información adicional es obtenida del servicio *pfamserver*; un encapsulamiento de la base de datos de Pfam que expone en forma de API un conjunto de consultas pre-diseñadas a las cuales el servidor MISTIC 2 tiene acceso.

```
(a)

GET /api/PfamA?q={"filters":[{"name":"pfamA_acc","op":"eq","val":"PF01030"}],"single":true}

GET /api/Uniprot?q={"filters":[{"name":"uniprot_id","op":"eq","val":"EGFR_HUMAN"}],"single":true}

(b)

GET /api/query/pfam_uniprot/EGFR_HUMAN

GET /api/query/sequencedescription_pfam/PF01030?with_pdb=true

GET /api/query/pdb_sequencedescription/EGFR_HUMAN,57,168
```

Figura 9: API de pfamserver. a) Consultas a tablas particulares, una a la tabla PfamA y la otra a Uniprot. b) Consultas que combinan múltiples tablas: la primera obtiene una lista de pfams a partir de un uniprot, la segunda la lista de identificadores de secuencias a partir de un pfam (se puede establecer si sólo se desean las que tienen PDB asociados), y por último la lista de identificadores de PDB para una secuencia de referencia.

En concreto, el servicio pfamserver permite realizar consultas directamente sobre cada una de las tablas de la base de datos Pfam (ver Figura 9-a). Además posee una pequeña interfaz en donde se listan algunas consultas específicas que

combinan varias tablas de la misma (ver Figura 9-b). Los resultados se retornan en formato JSON[7].

3.2 Arquitectura de procesamiento de datos

Existe cierta dificultad técnica al momento de ejecutar las distintas alternativas para calcular la covariación, y en donde cada una de ellas requiere la instalación de diferentes herramientas de software, acepta archivos de entrada con formatos particulares, reconoce un conjunto de parámetros para adaptar su configuración a un problema en particular, y por último retorna los resultados en un formato de archivo diferente.

MISTIC 2 agrupa el procesamiento de los datos y configuración del software necesario en un servicio web que seleccionar los parámetros de entrada a través de múltiples interacciones con el servicio y de una forma estándar para todos los algoritmos, evitándose al usuario tener que conocer los formatos de entrada y salida requeridos por cada uno de ellos.

Puntualmente, al proveer un archivo MSA el servicio se encarga de transformarlo en el formato requerido por cada uno de los algoritmos y luego se procesan los resultados producidos para adaptarlos al formato JSON.

La adaptación al formato JSON consiste en recuperar el resultado de un algoritmo, y para cada par de posiciones de la proteína (i, j) crear una lista con el formato $[i, j, puntaje]$. Por ejemplo, en $[10, 14, 5.1231]$, 10 y 14 son posiciones de la proteína y 5.1231 es el puntaje de covariación para dicho par de posiciones.

Esta estandarización de las entradas y salidas, también permitió extender el servicio con otros algoritmos con cálculos auxiliares, como la Conservación y la Distancia entre Residuos, para ampliar la información obtenida del MSA y de la estructura y proveer una visión más integrativa de la información contenida en los datos del usuario.

[7] Formato ampliamente utilizado en las tecnologías web.

Internamente, el servicio web y el procesamiento de los datos se implementó utilizando el lenguaje Python. Específicamente se utilizó la librería Flask-Restplus para implementar los diferentes puntos de entrada de la API (ver Apéndice A). Luego, se almacena la información en una base de datos postgresql mediante la librería SQLAlchemy (ver backend en Figura 7).

```
{'name': 'mfDCA', 'large_name': 'Mean Field Direct Coupling Analysis',
 'algorithm_type': 'correlation',
 'parameters': [
    {'name': 'Use sequence clustering',
     'short': 'clustering', 'default': True},
    {'name': 'Identity for sequence clustering',
     'short': 'identity', 'default': 0.62, 'depend': 'clustering',
     'min': 0.0001, 'max': 0.9999},
    {'name': 'Pseudocount',
     'short': 'pseudocount', 'default': 0.5,
     'min': 0},
    {'name': 'Max fraction of gaps per column allowed in calculations',
     'short': 'max_gap', 'default': 0.9, 'min': 0, 'max': 1}]}
```

Figura 10: Descripción de los parámetros del algoritmo mfDCA.

Por último, Celery permitió realizar el procesamiento de forma asíncrona a la interacción con el usuario. Se realiza un preprocesamiento de cada solicitud para garantizar que los diferentes formatos de entrada sean adaptados, se corren todos los algoritmos y se notifica al usuario vía e-mail cuando todos los algoritmos concluyeron su ejecución. Al mismo tiempo, Celery permitirá en un futuro trasladar y/o distribuir su cálculo de covariación a otros servidores de cálculo independiente, a los cuales MISTIC 2 podría conectarse y delegar trabajos, evitando así consumir los recursos del servidor web.

3.2.1 Extensión a nuevos métodos de cálculo

Recientemente se han desarrollado nuevos métodos para calcular covariación (Ekeberg et al. 2013, Kaján et al. 2014, Morcos et al. 2011, Seemayer et al. 2014, Weigt et al. 2009) y que han sido ampliamente aceptados en la comunidad en lugar de los métodos clásicos como MI. Numerosas variantes del mismo método han surgido, cada una con correcciones agregadas que brindan ciertas ventajas en cuanto a la calidad de los resultados, dependiendo el caso de estudio. Por ello, en el

diseño de MISTIC 2 se desarrolló una arquitectura de trabajo que permita la incorporación de nuevos algoritmos de forma sencilla y sistemática.

Dar soporte a un nuevo algoritmo para inferir coevolución consiste en registrar la lista de parámetros que acepta dicho algoritmo, incluyendo los rangos de valores posibles para cada parámetro (ver Figura 10), y luego implementar una tarea de Celery que ejecute el algoritmo en cuestión y recupere los resultados de la ejecución para que puedan ser utilizados por MISTIC 2 (ver Figura 11).

```
1. def mfdca(parámetros_del_usuario):
2.     parámetros_del_algoritmo = traducir(parámetros_del_usuario)
3.     resultados_del_algoritmo = ejecutar(parámetros_del_algoritmo)
4.     resultados_de_mistic = adaptar(resultados_del_algoritmo)
5.     return resultados_de_mistic
```

Figura 11: Pseudocódigo para ejecutar mfDCA y recuperar los resultados. [2] Traduce los valores proporcionados por el usuario. [3] Ejecuta el algoritmo mfDCA. [4] Adapta los resultados obtenidos al formato JSON.

3.3 Arquitectura de visualización de resultados

MISTIC 2 mantiene la misma funcionalidad provista en MISTIC 1, pero ha simplificado la forma en que el usuario interactuaba con algunos procesos, volviéndose más dinámico e intuitivo.

La arquitectura de visualización se diseñó e implementó utilizando el framework de javascript KnockoutJs que permite diseñar interfaces gráficas dentro de un navegador web. Este permitió desarrollar la aplicación como un conjunto de componentes reutilizables, que fueron construyendo componentes más complejos, hasta llegar a la aplicación web completa (ver *frontend* en la Figura 7).

El componente *job_request* se encarga de ayudar al usuario a completar los parámetros de entrada. Este contiene 3 subcomponentes, uno que se enfoca en el MSA (y la secuencia de referencia), otro en el PDB y otro en los parámetros de los algoritmos que se desean correr (ver Figura 12).

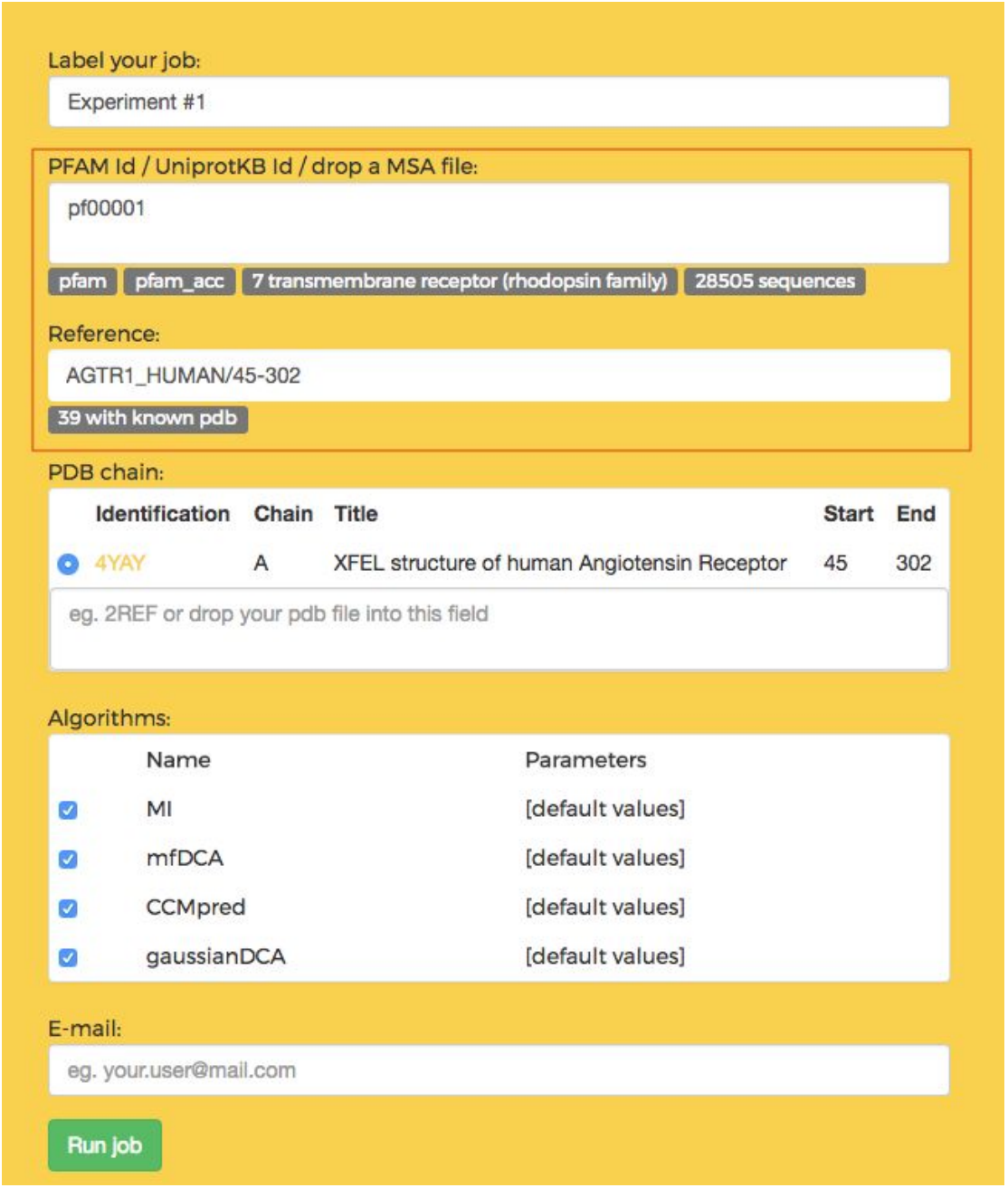


Figura 12: Captura de pantalla del componente job_request. Compuesto por los subcomponentes msa_setting (rectángulo rojo), pdb_setting (con título PDB Chain) y algorithm_setting (con título Algorithms).

Por otra parte, el componente *job_result* es el encargado de recuperar los resultados de los algoritmos, y descargalos dentro del navegador web. Cuando el usuario selecciona un subconjunto de los resultados procesados en el servidor, estos serán pasados como parámetros del componente *subset*.

El componente *subset* cumple la función de agrupar a uno o más algoritmos para ser visualizados de manera conjunta; toma los datos que se le pasaron como parámetros y construye una pequeña base de datos utilizando la librería datacollectionJs.

Figura 13: Componente subset. Panel izquierdo: muestra la red de covariación coloreado por conservación de los nodos. Se muestra la información de covariación entre las posiciones 240 y 242 para mfDCA. Panel derecho superior: visualización de estructura 3D (pdb: 4YAY). Panel derecho inferior: varias pestañas muestran distintas herramientas disponibles. En la pestaña actual se muestra la secuencia de referencia seleccionada. La línea gris sobre la secuencia muestra la porción alineada con el PDB. El residuo marcado en amarillo muestra la selección de esa posición.

Esta base de datos, simplifica en gran medida la forma de realizar consultas a los datos, reduciendo el trabajo requerido para post-procesarlos y filtrarlos teniendo en cuenta lo seleccionado en el componente *parameters*. Éste último componente

permite al usuario establecer para diferentes variables un rango de valores, como por ejemplo, mostrar solamente valores de covariación mayores a un cierto valor.

Además de *parameters*, el componente *subset*, contiene a los componentes *graph, pvmol, logos, roc*, que utilizan la base de datos para mostrar los resultados de una manera interactiva y desde diferentes perspectivas (ver Figura 13).

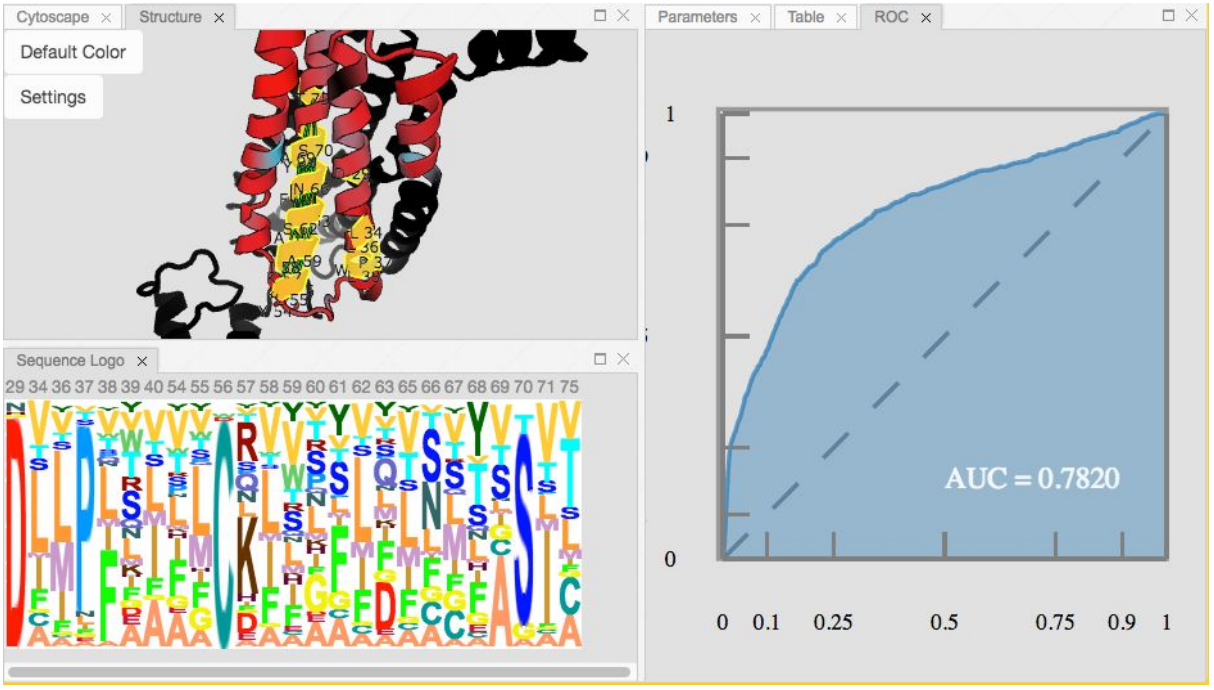


Figura 14: Sequence Logo de nodos seleccionados y Curva ROC con valor de AUC. Panel izquierdo inferior: Sequence Logo de nodos seleccionados mostrando frecuencias relativas. Panel izquierdo superior: las mismas posiciones mostradas en el Logo se muestran seleccionadas en amarillo en la estructura. Panel derecho: Curva ROC y valor de AUC para predicción de contactos en el score de covariación elegido.

En el caso del componente *graph* (ver Figura 13, Cytoscape), utiliza la librería *cytoscapeJs* para mostrar las distintas posiciones de la secuencia de referencia como nodos y la covariación entre las posiciones como enlaces entre ellos. Los nodos, pueden ser coloreados en base a los distintos resultados, como

Conservación, Cumulative Score, Proximity Score, Estructura secundaria[8], y por Clusters obtenidos a partir de un Markov clustering (Enright et al. 2002).

Además, se ha creado un mecanismo global de selección que permite mantener sincronizada la selección de aminoácidos y covariaciones, entre todos los subcomponentes dentro de *subset* (ver selección en amarillo en Figura 14, Structure).

En la solapa Sequence Logo (ver Figura 14), se puede visualizar el componente *logos* con las posiciones seleccionadas; mientras que en la solapa ROC, se visualiza la Curva ROC y el área bajo la curva ROC(AUC) en términos de predicción de contactos en la estructura 3D.

3.3.1 Extensión a nuevos componentes de visualización

Considerando que cada visualización sintetiza perspectivas o formas de ver los datos, y previendo que nuevas visualizaciones podrían permitir resaltar distintas características de los mismos, se diseñó MISTIC 2 de una manera que permite incorporar nuevos componentes de forma sencilla.

Dar soporte a una nueva visualización, consiste en definir un nuevo componente mediante KnockoutJs, y modificar el componente *subset* para establecer la solapa en que se visualizará el nuevo componente.

3.4 Combinación entre algoritmos

Dado que cada algoritmo produce diferentes resultados, se plantea la combinación de los puntajes de múltiples algoritmos en uno solo. Esto podría ser ventajoso para el usuario en el caso de que múltiples algoritmos concuerden en los resultados y refuercen las conclusiones obtenidas. En el caso de predicción de contactos, si diferentes algoritmos encuentran diferentes pares de posiciones con alta covariación, la mejora se manifestará en un aumento del AUC en la curva ROC.

[8] Obtenida a partir de las anotaciones del archivo MSA.

Esta estrategia depende íntimamente de la manera en que los resultados serán combinados. La estrategia de combinación implementada consiste en normalizar los resultados de cada uno de los algoritmos a valores dentro del intervalo cerrado 0-1, para luego promediar los valores normalizados de los diferentes algoritmos para cada par (i, j). Si bien la actual fórmula para combinar scores es simple, se deja la posibilidad de agregar fácilmente otras formas de combinar scores. Proyectando trabajo a futuro, existe la posibilidad de permitir al usuario implementar su propia fórmula para combinar scores.

Luego de calculado el score conjunto, automáticamente se recalculan los puntajes derivados como cScore y pScore.

4 Significado biológico de los resultados provistos

4.1 Conservación y Logos

La conservación de posiciones de un MSA es la información más básica y fundamental de una familia de proteínas. Los residuos conservados generalmente corresponden a posiciones que desempeñan papeles biológicos o estructurales importantes. Un score de conservación, nos muestra (sin importar cuál es el aminoácido) posiciones claves en nuestra familia de proteínas. Existen varias métricas para medir la conservación en secuencias de proteínas. Una de ellas es la divergencia de Kullback-Leibler (KL), otra opción ampliamente utilizada es la entropía de Shannon (S) y, más tradicionalmente la frecuencia de aminoácidos.

Los logos de secuencias son una representación gráfica del contenido de información almacenado en un MSA y proporcionan una representación compacta y altamente intuitiva de la composición de aminoácidos específica en secuencias biológicas (Thomsen & Nielsen 2012).

Posiciones donde los residuos están más conservados, significa que uno o unos pocos aminoácidos están permitidos en esa posición y probablemente tenga importancia biológica. Por ejemplo, motivos de unión a ligandos, unión entre proteínas, sitios activos, posiciones fundamentales para mantener el plegado, etc.

Posiciones donde no hay una restricción en los residuos permitidos, pueden dar cuenta de regiones flexibles en una proteína, entre otros. Estas posiciones son, en principio, las más factibles de ser cambiadas sin poner en riesgo la función y/o estructura de la proteína. Conocer esto podría ser útil para un diseño racional de experimentos como: mutaciones, añadir sitios funcionales a una proteína, crear proteínas de fusión, etc.

Los logos nos muestran visualmente que las distintas posiciones de un MSA están sujetas a distintas presiones selectivas, siendo unas más restringidas que otras.

El logo de la Figura 15 nos muestra que hay una serie de aminoácidos fundamentales alrededor de los residuos 35-45 (FxxxWCxxCxxxxP) estos probablemente sean el centro activo de la proteína y fundamentales para su función.

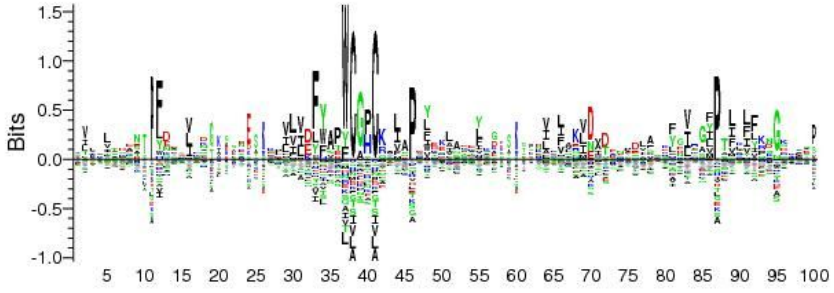


Figura 15: Logo de secuencias con el algoritmo kullback-Leibler de la familia de proteínas tiorredoxina. Las posiciones más altas muestran los aminoácidos más conservados. Los residuos que se ven por debajo de la línea 0, son aminoácidos que se encuentran en una frecuencia inferior a la frecuencia básica en la naturaleza, por consiguiente, seguramente no permitidos en esa posición.

4.2 Mapeo en la estructura

Observar en la estructura 3D relaciones entre aminoácidos o posiciones conservadas de un documento bi dimensional como es el MSA es complejo para usuarios no expertos. Saber si dos aminoácidos están cerca en la estructura tridimensional es tarea imposible si no se realiza un mapeo a la estructura. Proveer este tipo de mapeo, permite al usuario no experto, como ejemplo biólogos, sacar hipótesis de relevancia biológica que no hubiera sido posible aún contando con el MSA y la estructura por separado (ver Figura 16).

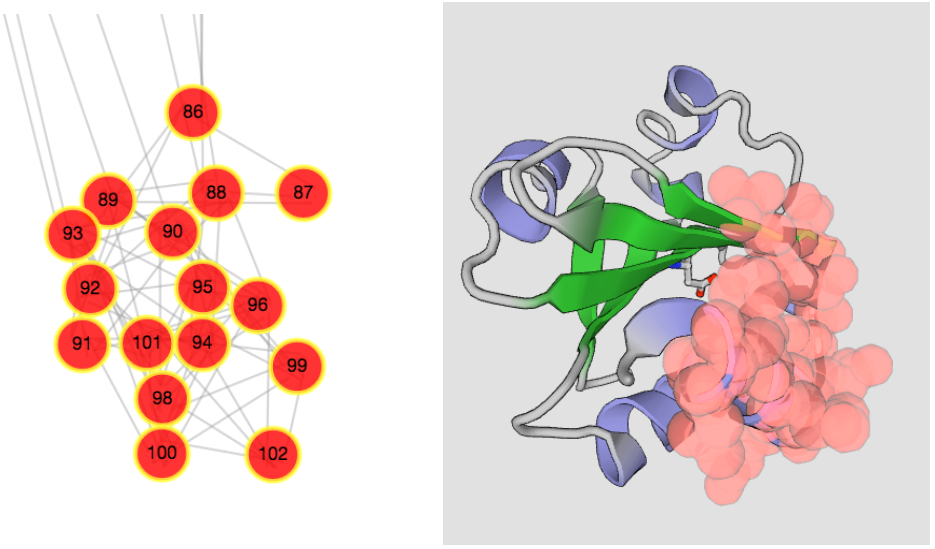


Figura 16: Red de residuos que coevolucionan mostrando el mismo par de posiciones en la red y en la estructura 3D. En la izquierda, ubicación de las posiciones correlacionadas en la red de coevolución. En la derecha, la estructura 3D y en rojo las posiciones de la red.

Con MISTIC 2 es posible mapear en la estructura los residuos más conservados y los que más covarian y ver cómo se relacionan entre sí estos dos grupos de aminoácidos. Se ha visto en Buslje CM et al. (Buslje et al. 2009) que alrededor de residuos conservados, en la estructura 3D de proteínas, existe una esfera de residuos con alta información. De esta observación se hipotetiza que los aminoácidos más conservados son indispensables para realizar una función específica (como crear o romper un enlace), pero las posiciones cercanas no están libres de restricciones, sino que permiten unos pocos residuos y si alguno cambia, probablemente otro en su proximidad tridimensional deba cambiar. Estas restricciones son necesarias por ejemplo, para mantener un entorno y que la afinidad de la molécula que se va a transformar siga siendo similar.

4.3 Redes de covariación

Como se mencionó a lo largo de esta tesis, las redes de coevolución nos muestran posiciones relacionadas entre sí, probablemente necesarias para, en coordinación, realizar una función biológica.

MISTIC2 provee esta red y nos permite interactuar con la misma. Por ejemplo, se puede ver cuáles son los 10 residuos más interrelacionados y cuales son relaciones de un aminoácido dado (primeros vecinos).

Además se da la opción de agrupar nodos usando la topología de la red, a través de un procedimiento de análisis de redes llamado clusters de Markov. Este procedimiento reconoce módulos de nodos que están más interrelacionados entre sí que con otros nodos. El agrupamiento de un conjunto de nodos podría indicar distintas funciones biológicas de cada uno de los grupos de aminoácidos (Aguilar et al. 2012). Un análisis de este tipo ha permitido a Najeeb Halabi et al. (Halabi et al. 2009) estudiar la familia de proteinas Serin proteasas S1A. El análisis indica una descomposición de la proteína en tres grupos cuasi independientes de aminoácidos correlacionados que se denominará como "sectores de proteínas".

Cada sector está conectado físicamente en la estructura terciaria, tiene un papel funcional distinto y constituye un modo independiente de divergencia de secuencia en la familia de proteínas. Un sector comprende un anillo de residuos dentro del núcleo de los dos barriles β, otro comprende el bolsillo S1 y su entorno, y el último comprende el mecanismo catalítico de la proteasa ubicada en la interfaz de los dos barriles β. Estos sectores se relacionan con la especificidad catalítica, el tipo de organismo o el mecanismo químico. Los resultados de ese trabajo muestran que los sectores funcionalmente relevantes se encuentran en varias familias de proteínas y proporcionan una base no trivial para el diseño del experimento.

Una característica fundamental de MISTIC 2, es que permite reproducir este tipo de análisis de manera simple, donde se mapean estos clusters obtenidos directamente en la estructura 3D para ver donde se ubican sin recurrir a software adicional.

Cada ítem de la información mostrada en este capítulo puede ser de relevancia biológica y puede ser utilizado de manera independiente, pero el mayor provecho que otorga esta herramienta es la capacidad de integrar toda la información: secuencial estructural y evolutiva en una representación sencilla, visual e interactiva.

5 Caso de estudio utilizando GPCRs

Para utilizar el servicio desarrollado, se procede a explorar a la familia de receptores acoplados a proteínas G (GPCR, del inglés “G-Protein coupled receptors”). Son receptores ubicados en la membrana plasmática y participan en una gran cantidad de eventos de transducción de señales desde el exterior celular hacia el interior.

Existen distintos miembros de la GPCR en procesos muy diversos, tales como participar de la señalización del sistema inmune, o neurogénesis, entre otros. Cerca del 4% de los genes humanos codifican GPCRs. La función alterada de las GPCRs está asociada a diversas enfermedades, varias de la cuales son muy prevalentes en humanos.

El modelo de acción canónico de la GPCR implica la unión del ligando al sitio de reconocimiento extracelular, el cual induce cambios conformacionales en la disposición de las hélices transmembrana que repercuten en los bucles intracelulares de interacción con proteínas G.

En el estado activado, mediado por la interacción con el ligando, los bucles se acoplan al trímero de proteínas G. Esta acción promueve la eliminación de una molécula de GDP de la subunidad $G\alpha$ del trímero, su disociación y la carga de una molécula de GTP. $G\alpha$ en este estado interactúa con el subsiguiente actor en la cascada de señalización. El ciclo de señalización se cierra por la auto-hidrólisis del GTP a GDP de $G\alpha$ y la liberación del ligando de su sitio de reconocimiento. La activación de GPCR se ha implicado en 5 vías que conducen a la proliferación celular.

La estructura tridimensional de una GPCR consiste en siete hélices que atraviesan la membrana plasmática de forma casi paralela, del lado extracelular se encuentra el sitio de unión al ligando y del lado citoplasmático hay bucles que pueden interaccionar con un heterotrímero de proteínas G. En particular se analizó una familia de proteínas receptores, familia “Rodopsina”, Pfam PF00001. Para el análisis se utilizó la secuencia de referencia *AGTR1_HUMAN/45-302* y los resultados se

alinearon en 2 estructuras de PDB (*4YAY* y *4ZUD*). Los resultados obtenidos usando cada una de las estructuras se los llama experimentos de aquí en más. La secuencia de referencia, contiene a las posiciones proteicas de las alfa hélices que se encuentran en la membrana. (ver Figura 17).

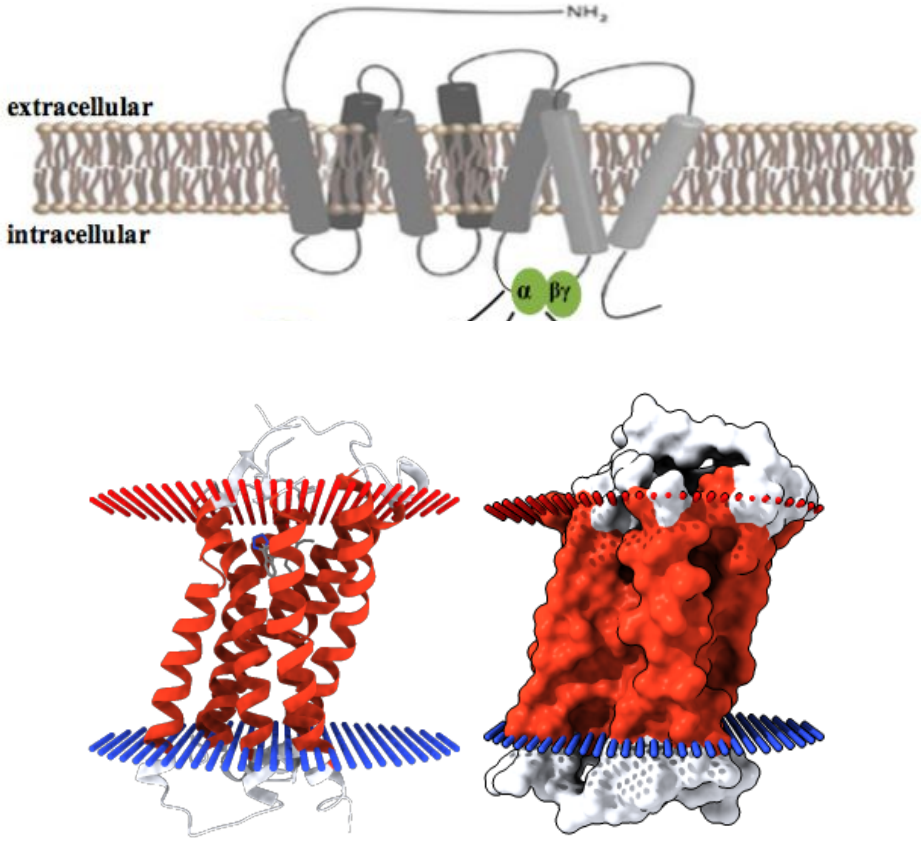


Figura 17: Panel superior: Representación esquemática de las 7 hélices transmembrana de los receptores GPCR. Panel inferior: representación de cinta de la estructura 3D del receptor AGTR1 (pdb: 4YAY) mostrando su inserción en la membrana, representación de superficie de la misma zona. Coloreado en rojo, el dominio incluído en la familia de Pfam PF00001 (las 7 hélices).

De cada experimento, se toman métricas de los 4 algoritmos disponibles (CCMpred, MI, gaussianDCA, mfDCA) y las combinación de a pares posibles de los mismos (MI+mfDCA, MI+CCMpred, MI+gaussianDCA, mfDCA+CCMpred, etc).

5.1 Comparación de rendimiento entre estructuras

De cada algoritmo o par de algoritmos se verifica el AUC de la curva ROC para predicción de contactos. Además, se selecciona el top 5% de las correlaciones y se registra la cantidad encontrada, la cantidad de clusters de Markov, y la cantidad de nodos a distancia de contactos en los respectivos PDBs (ver Tablas 1 y 2).

Algotitmo	AUC	Clusters	Correlaciones	Contactos	Porcentaje
CCMpred	**0.8361**	27	2002	192	0.0959
MI	0.5769	3	1459	113	0.0775
gDCA	0.611	55	1503	224	0.1490
mfDCA	**0.7823**	18	1502	163	0.1085
MI+CCMpred	0.656	6	2009	162	0.0806
MI+gDCA	0.4928	12	1653	152	0.0920
MI+mfDCA	**0.7511**	14	1664	177	0.1064
mfDCA+CCMpred	**0.7626**	19	2013	209	0.1038
mfDCA+gDCA	**0.7072**	18	1670	197	0.1180
CCMpred+gDCA	0.4329	40	2010	326	0.1622

Tabla 1: Resultados obtenidos para el PDB 4YAY.

En términos generales, ambas estructuras de PDB obtuvieron resultados parecidos. Al realizar una comparación del mismo algoritmo en las 2 estructuras, se notan diferencias sutiles tanto en el AUC como en la cantidad de correlaciones con distancia de contacto. En general, todos los algoritmos pudieron explicar sutilmente mejor la estructura de 4YAY que la estructura de 4ZUD. Esto nos demuestra que en general, cualquier estructura de referencia nos aporta información similar y representativa de toda la familia en estudio.

En general, el desempeño de un método (en este caso para predecir contactos) se considera aceptable si tiene un AUC superior a 0.7. Como se vió en las tablas CCMpred, mfDCA, mfDCA+CCMpred y MI+mfDCA cumplen con esta condición.

Para evaluar el provecho de la información que otorga el servidor en términos de significado biológico, se analizan distintas variables en el caso de estudio.

Algoritmo	AUC	Clusters	Correlaciones	Contactos	Porcentaje
CCMpred	**0.8324**	27	2002	193	0.0964
MI	0.5726	3	1459	92	0.0631
gDCA	0.6101	54	1503	212	0.1411
mfDCA	**0.7761**	18	1502	155	0.1032
MI+CCMpred	0.6536	6	2009	135	0.0672
MI+gDCA	0.4905	12	1653	129	0.0780
MI+mfDCA	**0.7474**	15	1664	152	0.0913
mfDCA+CCMpred	**0.7616**	19	2013	199	0.0989
mfDCA+gDCA	0.697	18	1670	186	0.1114
CCMpred+gDCA	0.4308	41	2010	318	0.1582

Tabla 2: Resultados obtenidos para el PDB 4ZUD.

En principio se tienen en cuenta solo los valores obtenidos con MI, ya que es el único algoritmo con el que se ha evaluado su significado biológico de las variables conservación, cMI y pMI, aunque los algoritmos de información directa (mfDCA, Gremlin y CMPred) probablemente sean mejores para la predicción de contactos.

5.2 Puntajes de residuos derivados de MI

5.2.1 Conservación

Al analizar los residuos cuya conservación es superior a 2 que es una conservación elevada en la escala de 0 a 4.6 de la entropía de Shannon, se observa que sólo 16 residuos superan ese valor (ver Figura 18).

Dichos residuos se encuentran hacia el interior del anillo formado por las 7 hélices, en contacto entre sí y probablemente sean necesarios para mantener la estructura. Las interacciones entre las hélices es importante para que se mantengan en esa posición, garantizando la conformación en forma de anillo.

Una observación interesante es que no existen muchas relaciones de co-variación entre estos residuos. Esto es lo que se espera ya que los residuos conservados no varían, por lo tanto no sería posible encontrar co-variación con otros residuos.

Además en la Ecuación 2, se puede ver que si la conservación es perfecta, la MI es cero.

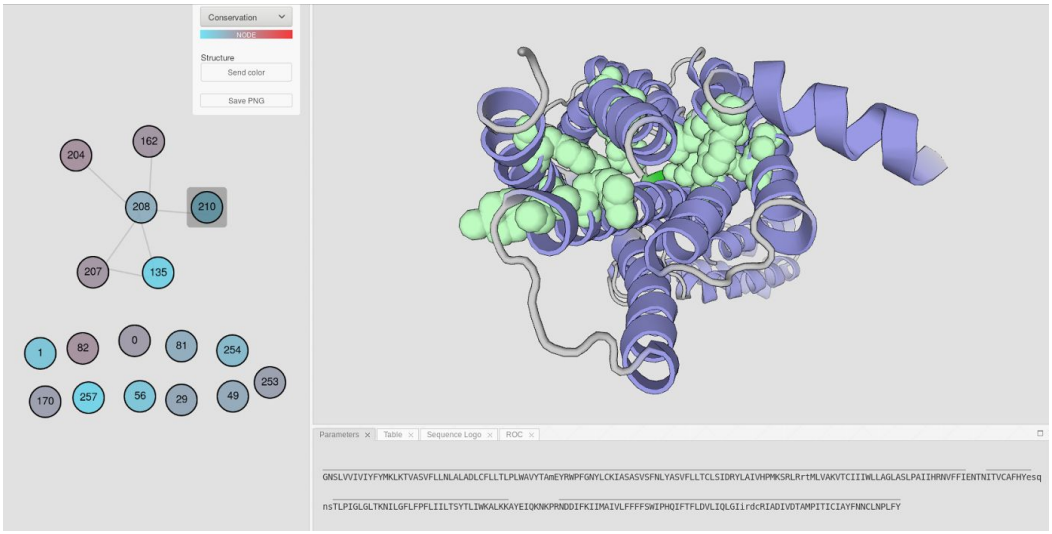


Figura 18: Residuos de PF00001 con conservación mayor a 2. Izquierda: residuos más conservados. Derecha: los mismos residuos mapeados en la estructura con representación de esferas, coloreados en verde (pdb: 4YAY).

5.2.2 Cummulative MI (cMI)

Si se representan los residuos con mayor cMI, por el contrario, se verifica que en la red se encuentran muy interrelacionados. Por su definición, estos residuos son los que más “acumulan” información. Obtienen el valor de cMI de la sumatoria del valor de coevolución con los nodos vecinos (ver Figura 19).

Mapeando ambos subset de residuos, conservados y de mayor cMI, observamos que son distinto set de aminoácidos (ver Figura 20) probablemente con funciones distintas dentro de la proteína.

Figura 19: *Residuos de PF00001 con mayor cMI. Izquierda: red de residuos con mayor cMI. Derecha: los mismos residuos mapeados en la estructura con representación de esferas, coloreados en rosa (pdb: 4YAY).*

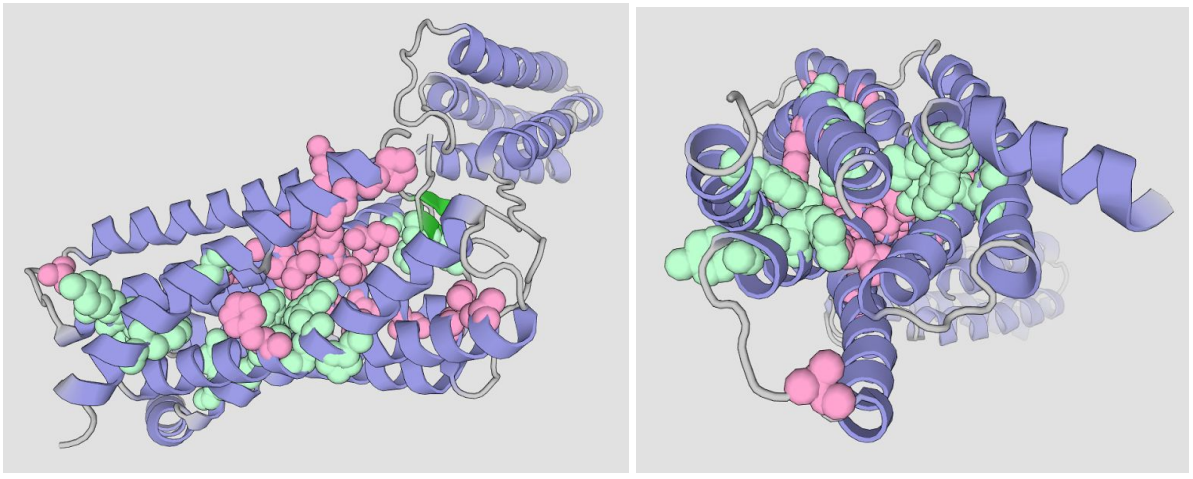


Figura 20: Residuos conservados y con alto cMI de la estructura 4YAY. Izquierda: vista lateral del dominio; Derecha, vista frontal. Se muestran con representación de esferas los residuos conservados (verde) y de mayor cMI (rosa) (pdb: 4YAY).

5.2.3 Proximity MI (pMI)

En Buslje CM et al. (Buslje et al. 2009) se encontró que residuos funcionalmente importantes, como son los residuos catalíticos de enzimas tenían elevada pMI, osea tenían residuos con alto contenido de información (cMI) próximos en la estructura (en una esfera < 7 Angstroms). Vale la pena recordar que la pMI es un score que se le da a cada residuo calculado como la suma de la cMI de todos los residuos en la esfera de 7A en la estructura. Los residuos con alta pMI suelen tener conservacion elevada y por lo tanto ellos no tienen alta cMI. Aunque no se ha demostrado, el mencionado trabajo sugiere que otros residuos funcionalmente importantes pueden tener el mismo comportamiento.

En la Figura 21 (panel superior) se muestran los 10 aminoácidos de mayor pMI mapeados a la estructura. Se ha encontrado en la base de datos COSMIC (Catalogue Of Somatic Mutations in Cancer) (Forbes et al. 2017) que dos de esos aminoácidos, el 34 y el 152, que en la numeración de Uniprot corresponden al 78 y al 196 respectivamente, adquieren mutaciones somáticas en cáncer. Las mutaciones somáticas son aquellas que se adquieren durante el desarrollo del tumor (ver Tabla 3).

Gene Name	Sample ID	AA Mutation	CDS Mutation	Primary Tissue	Histology	Histology Subtype 1	Pubmed Id	Somatic Status	Sample Source
AGTR1	1651640	p.L78V	c.232T>G	Large intestine	Carcinoma	Adenocarcinoma	-	Confirmed Somatic	Tumour Sample
AGTR1	2480876	p.G196C	c.586G>T	Lung	Carcinoma	Small cell carcinoma	26168399	Confirmed Somatic	Tumour Sample
AGTR1	1780059	p.G196C	c.586G>T	Lung	Carcinoma	Adenocarcinoma	-	Previously Reported	Tumour Sample

Tabla 3: Mutaciones somáticas de los residuos 78 y 196 (numeración de Uniprot). Extraido de la base de datos de mutaciones somáticas en cáncer COSMIC.

De este análisis, se puede hipotetizar que el puntaje de pMI podría estar indicando que estos residuos son funcionalmente importantes y una sustitución en estas

posiciones puede ser causante de disfunción de la proteína y estar implicado en el desarrollo del tumor. No hemos encontrado en literatura ni bases de datos si los demás residuos con elevada pMI tienen alguna importancia biológica.

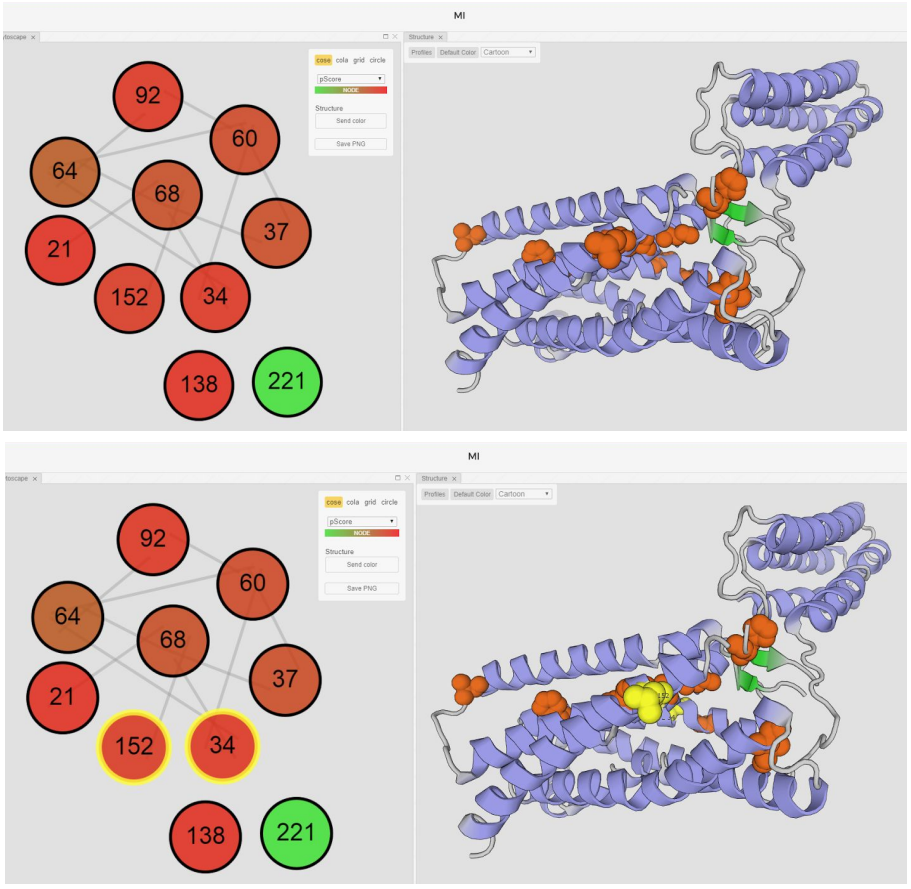


Figura 21: Residuos con mayor pMI con selección de mutaciones encontradas en COSMIC. Panel superior: mapeo en la estructura de los 10 residuos con mayor pMI. Coloreados según patrón de pMI de verde (valores más alto) a rojo (valores más bajos). Panel inferior: seleccionados los aminoácidos 34 y 152 (Uniprot 78 y 196), coloreados en la estructura en amarillo (pdb: 4YAY).

5.3 Cálculos derivados de CCMpred

A continuación se demuestra la posibilidad de utilizar el método CCMpred dado que es el algoritmo que tiene mejor desempeño predictivo para determinar contactos (ver Tablas 1 y 2).

La Figura 22 muestra el clustering the Markov. Si bien no se realiza ninguna demostración del rol biológico de los sectores de esta proteína, es muy interesante observar que la topología de la red de covariación permite un clustering (con el algoritmo de clustering de Markov) que genera sectores muy bien definidos en la estructura de la proteína.

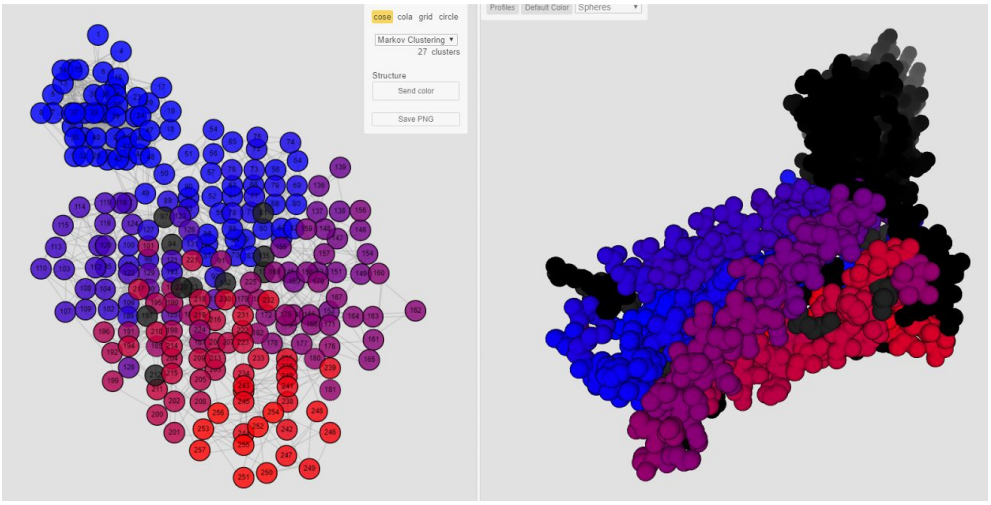


Figura 22: Clustering de Markov sobre los resultados de CCMpred. Izquierda, red de covariación obtenida con CCMpred coloreada por clusters. Derecha, mapeo de los clusters en la estructura. Representación de esferas (pdb:4YAY).

6 Conclusiones

La creación de 3 arquitecturas permitió reducir la complejidad total del sistema, y al establecer la inter-comunicación por solicitudes HTTP, cada arquitectura puede ser reemplazada o actualizada sin necesidad de realizar cambios en las otras.

En particular, la utilización de Celery, simplificó la complejidad del procesamiento en paralelo de los algoritmos. Mientras que la librería KnockoutJs permitió construir componentes de visualización que se encuentren interrelacionados, dando lugar a mejorar la experiencia del usuario en procesos como aplicación de filtros y selección de aminoácidos.

A futuro, en cada una de las 3 arquitecturas presentadas, se desprenden diferentes líneas de trabajo.

En la arquitectura de acceso a los datos de Pfam, se puede mejorar el proceso de actualización automática para garantizar la utilización de la última versión de la base de datos. Por otra parte, también existe potencial para agregar nuevas consultas que mejoren asistencia al usuario de MISTIC 2, ya sea proporcionando nuevas formas para seleccionar un archivo de MSA o una estructura de PDB.

En la arquitectura de procesamiento, aún falta agregar el soporte para el resto de los algoritmos existentes para estimar puntajes de coevolución, y continuar agregando los nuevos algoritmos que vayan surgiendo.

Además, es necesario agregar nuevos algoritmos complementarios (como actualmente lo son Conservación o la Distancia entre residuos de la estructura), que aunque no sean el eje de la arquitectura de procesamiento, brindan información complementaria en la arquitectura de visualización.

Por otra parte, en la arquitectura de visualización, queda agregar nuevos componentes que den visibilidad a nuevas dimensiones de los datos (a partir de algoritmos de clusterización diferentes) o que permitan comprimir múltiples

dimensiones en un solo gráfico (como es el Circos). También queda por proporcionar diferentes estrategias para combinar dos o más algoritmos.

Por último, se ha hecho una análisis inicial de la familia de proteínas GPCR. El alineamiento múltiple de secuencias de proteínas homólogas, nos permite calcular señales evolutivas en toda la familia, como la conservación y scores de covariacion. Estos valores nos dan información biológica relevante como posiciones con posibilidades de reemplazo reducidas, que son las posiciones conservadas y otras con restricciones coordinada con otro/otros residuos, como son los residuos que co-varían.

Tomando una proteina de la familia como referencia (AGTR1_HUMAN) se ha podido mapear en la estructura diferentes grupos de residuos y de esta manera obtener información biológica y generar hipótesis que luego deberán ser comprobadas con soporte bibliografico o por técnicas experimentales.

Como ejemplo, se observó que los aminoácidos más conservados se orientan hacia el centro del anillo formado por las hélices lo que permite hipotetizar que serán importantes para mantener la estructura. Otra observación es que los aminoácidos con alto score de cMI son un grupo distinto, por lo que podemos hipotetizar que cada grupo de aminoácidos cumple una función específica y ello se traduce en restricciones evolutivas que podemos inferir por sus scores de conservación y co-variación. Otra observación es que el score de pMI nos podría estar resaltando posiciones de importancia funcional. Dos de los residuos con mayor score se encuentran implicados en cáncer de intestino y pulmón.

Además, se observó que los algoritmos tienen buen rendimiento para la predicción de contactos, siendo el mejor CCMpred y mejorando la predicción de otros con la combinación de métodos. Así mismo se observó que la topología de una de las redes de covariacion delimita sectores, que son regiones definidas en la estructura 3D de la proteína. Queda por comprobar si estos sectores cumplen distintas funciones.

Los análisis realizados en nuestro caso de ejemplo, son una muestra de las aplicaciones de esta herramienta, habiendo lugar para correr experimentos más complejos.

En resumen, MISTIC 2 es un metaserver que principalmente ofrece el cálculo de covariacion con diferentes métodos y la posibilidad de compararlos y combinarlos. Además ofrece una visualización interactiva de las redes de covariación que permite su exploración de una manera relativamente sencilla. Otra cualidad es el mapeo de toda la información en la estructura, mediante distintas coloraciones según el score y mediante la selección de distinto grupos de aminoácidos. MISTIC 2 es una suite de programas con una interface de visualización e interacción muy atractiva que acerca a los usuarios de nivel intermedio el análisis integrado de la secuencia, estructura, función y evolución de en familias de proteínas.

7 Apéndice A

A continuación se muestra un programa escrito en Python 3, que utilizando la API carga un trabajo en el servidor del Instituto Leloir, espera hasta que sea procesado, y por último descarga los resultados en archivos con formato JSON.

```
import requests
import json
import time

SERVER = 'http://mistic2.leloir.org.ar{path}'

def main():
   headers = {'Accept': 'application/json',
              'Content-Type': 'application/json',
              'Authorization': ''}

   # Job request general data
   jobs_url = SERVER.format(path='/api/v0/job_requests')
   job = {'description': 'Sample for automatic load through the API.',
          'email': 'eloy.colell+misticscript@email.com'}
   res = requests.post(jobs_url, data=json.dumps(job), headers=headers)
   job = res.json()
   job_url = '{base}/{public_key}'.format(base=jobs_url,
                                          public_key=job['public_key'])
   job_request_path = '/#/job_request/{public_key}'.format(public_key=job['public_key'])
   job_request_url = SERVER.format(path=job_request_path)
   print("The job request {job_request_url} was created.".format(job_request_url=job_request_url))

   # MSA Settings
   msa_url = '{base}/msa'.format(base=job_url)
   res = requests.get(msa_url, headers=headers)
   msa = res.json()
   # Update MSA storage
   msa_storage_url = '{base}/storage'.format(base=msa_url)
   msa_storage = {"format": "pfam_acc",
                  "input": "pf00131",
                  "origin": "pfam",
                  "storage": "pf00131"}
   res = requests.post(msa_storage_url, data=json.dumps(msa_storage), headers=headers)
   msa_storage = res.json()
   print("Found {description} with {number}.".format(description=msa_storage['input_description'],
                                                     number=msa_storage["num_full"]))
   # Get MSA references
   msa_references_url = '{base}/references'.format(base=msa_url)
   res = requests.get(msa_references_url, headers=headers)
   msa_references = res.json()
   print("Found {amount} with known PDB
structure.".format(amount=len(msa_references['references_with_pdb'])))
   # Select reference sequence
   msa_reference = {"selected_reference": msa_references['references_with_pdb'][0]}
   res = requests.put(msa_url, data=json.dumps(msa_reference), headers=headers)
   msa.update(res.json())
   print("Selected {reference} as the reference
sequence.".format(reference=msa['selected_reference']))

   # PDB Settings
   pdb_url = '{base}/pdb'.format(base=job_url)
   res = requests.get(pdb_url, headers=headers)
   pdb = res.json()
   pdb_selected = {"selected": pdb['available'][0]}
   res = requests.put(pdb_url, data=json.dumps(pdb_selected), headers=headers)
   pdb.update(res.json())
   selected = pdb['selected']
   print("Selected chain {chain} of {pdb_id} as the structure.".format(chain=selected['chain'],
```

```
                                                                    pdb_id=selected['pdb_id']))

    # Algorithm Settings
    algorithms_url = '{base}/algorithm'.format(base=job_url)
    res = requests.get(algorithms_url, headers=headers)
    algorithms = res.json()
    # Run only CCMpred and MI.
    for algorithm in algorithms:
        if algorithm['parameters'] != [] and algorithm['algorithm']['name'] not in ['MI', 'CCMpred']:
            algorithm['active'] = False
            algorithm_url = '{base}/{id}/result'.format(base=algorithms_url, id=algorithm['id'])
            res = requests.put(algorithm_url, data=json.dumps(algorithm), headers=headers)
            algorithm.update(res.json())

    # Run job
    job_run_url = '{base}/run'.format(base=job_url)
    res = requests.put(job_run_url, data=json.dumps({}), headers=headers)
    job.update(res.json())
    print("Job status is {status}.".format(status=job['status']))

    # Waiting for the results
    while job['status'] != 'finished':
        res = requests.get(job_url, headers=headers)
        job.update(res.json())
        time.sleep(10)
    print("Job status is {status}.".format(status=job['status']))

    # Obtain the results
    results = {}
    for algorithm in algorithms:
        result_url = '{base}/{id}/result'.format(base=algorithms_url, id=algorithm['id'])
        res = requests.get(result_url, headers=headers)
        results[algorithm['algorithm']['name']] = res.json()
    print("Obtained results: {keys}".format(keys=results.keys()))
    with open('{:}_results.json'.format(msa['input']), 'w') as f:
        json.dump(results, f)


if __name__ == "__main__":
  main()
```

8 Bibliografía

Aguilar D, Oliva B, Marino Buslje C. 2012. Mapping the mutual information network of enzymatic families in the protein structure to unveil functional features. *PLoS ONE*. 7(7):e41430

Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, et al. 1997. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*. 25(17):3389–3402

Atchley WR, Wollenberg KR, Fitch WM, Terhalle W, Dress AW. 2000. Correlations among amino acid sites in bHLH protein domains: an information theoretic analysis. *Mol Biol Evol*. 17(1):164–78

Blahut RE. 1987. *Principles and Practice of Information Theory*. Reading, Mass: Addison-Wesley

Buslje CM, Santos J, Delfino JM, Nielsen M. 2009. Correction for phylogeny, small number of observations and data redundancy improves the identification of coevolving amino acid pairs using mutual information. *Bioinformatics*. 25(9):1125–31

Codoñer FM, Fares MA. 2008. Why should we care about molecular coevolution? *Evol Bioinform Online*. 4:29–38

Cover TM, Thomas JA. 2005. *Elements of Information Theory*. Hoboken, NJ, USA: John Wiley & Sons, Inc.

Del Sol A, Araúzo-Bravo MJ, Amoros D, Nussinov R. 2007. Modular architecture of protein structures and allosteric communications: potential implications for signaling proteins and regulatory linkages. *Genome Biol*. 8(5):R92

Dellarole M, Caro JA, Roche J, Fossat M, Barthe P, et al. 2015. Evolutionarily conserved pattern of interactions in a protein revealed by local thermal expansion properties. *J Am Chem Soc*. 137(29):9354–62

Dunn SD, Wahl LM, Gloor GB. 2008. Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. *Bioinformatics*. 24(3):333–40

Dutheil JY. 2012. Detecting coevolving positions in a molecule: why and how to account for phylogeny. *Brief Bioinformatics*. 13(2):228–43

Ekeberg M, Lövkvist C, Lan Y, Weigt M, Aurell E. 2013. Improved contact prediction in proteins: using pseudolikelihoods to infer Potts models. *Phys Rev E Stat Nonlin Soft Matter Phys*. 87(1):012707

Enright AJ, Van Dongen S, Ouzounis CA. 2002. An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res*. 30(7):1575–84

Fares MA, Travers SAA. 2006. A novel method for detecting intramolecular coevolution: adding a further dimension to selective constraints analyses. *Genetics*. 173(1):9–23

Finn RD, Coggill P, Eberhardt RY, Eddy SR, Mistry J, et al. 2016. The Pfam protein families database: towards a more sustainable future. *Nucleic Acids Res*. 44(D1):D279-85

Fitch WM, Markowitz E. 1970. An improved method for determining codon variability in a gene and its application to the rate of fixation of mutations in evolution. *Biochem Genet*. 4(5):579–93

Fodor AA, Aldrich RW. 2004. Influence of conservation on calculations of amino acid covariance in multiple sequence alignments. *Proteins*. 56(2):211–21

Forbes SA, Beare D, Boutselakis H, Bamford S, Bindal N, et al. 2017. COSMIC: somatic cancer genetics at high-resolution. *Nucleic Acids Res*. 45(D1):D777–83

Franz M, Lopes CT, Huck G, Dong Y, Sumer O, Bader GD. 2016. Cytoscape.js: a graph theory library for visualisation and analysis. *Bioinformatics*. 32(2):309–11

Giraud BG, Heumann JM, Lapedes AS. 1999. Superadditive correlation. *Phys Rev E Stat Phys Plasmas Fluids Relat Interdiscip Topics*. 59(5 Pt A):4983–91

Gloor GB, Martin LC, Wahl LM, Dunn SD. 2005. Mutual information in protein multiple sequence alignments reveals two classes of coevolving positions. *Biochemistry*. 44(19):7156–65

Gouveia-Oliveira R, Pedersen AG. 2007. Finding coevolving amino acid residues using row and column weighting of mutual information and multi-dimensional amino acid representation. *Algorithms Mol Biol*. 2:12

Halabi N, Rivoire O, Leibler S, Ranganathan R. 2009. Protein sectors: evolutionary units of three-dimensional structure. *Cell*. 138(4):774–86

Hobohm U, Scharf M, Schneider R, Sander C. 1992. Selection of representative protein data sets. *Protein Sci*. 1(3):409–17

Janzen DH. 1980. When is it Coevolution? *Evolution*. 34(3):611

Kaján L, Hopf TA, Kalaš M, Marks DS, Rost B. 2014. FreeContact: fast and free software for protein contact prediction from residue co-evolution. *BMC Bioinformatics*. 15:85

Kamisetty H, Ovchinnikov S, Baker D. 2013. Assessing the utility of coevolution-based residue-residue contact predictions in a sequence- and structure-rich era. *Proc Natl Acad Sci U S A*. 110(39):15674–79

Lapedes AS, Giraud BG, Liu L, Stormo GD. 1999. Correlated Mutations in Models of

Protein Sequences: Phylogenetic and Structural Effects. *IMS Lecture Notes-Monograph Series*. 33:236–56

Little DY, Chen L. 2009. Identification of coevolving residues and coevolution potentials emphasizing structure, bond formation and catalytic coordination in protein evolution. *PLoS ONE*. 4(3):e4762

Lockless SW, Ranganathan R. 1999. Evolutionarily conserved pathways of energetic connectivity in protein families. *Science*. 286(5438):295–99

Marks DS, Colwell LJ, Sheridan R, Hopf TA, Pagnani A, et al. 2011. Protein 3D structure computed from evolutionary sequence variation. *PLoS ONE*. 6(12):e28766

Martin LC, Gloor GB, Dunn SD, Wahl LM. 2005. Using information theory to search for co-evolving residues in proteins. *Bioinformatics*. 21(22):4116–24

McMurrough TA, Dickson RJ, Thibert SMF, Gloor GB, Edgell DR. 2014. Control of catalytic efficiency by a coevolving network of catalytic and noncatalytic residues. *Proc Natl Acad Sci U S A*. 111(23):E2376-83

Morcos F, Pagnani A, Lunt B, Bertolino A, Marks DS, et al. 2011. Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *Proc Natl Acad Sci U S A*. 108(49):E1293-301

Myers J, Copeland R. 2015. *Essential SQLAlchemy*. Sebastopol, CA: O'Reilly Media. Second edition. ed.

Pazos F, Valencia A. 2008. Protein co-evolution, co-adaptation and interactions. *EMBO J*. 27(20):2648–55

Seemayer S, Gruber M, Söding J. 2014. CCMpred--fast and precise prediction of protein residue-residue contacts from correlated mutations. *Bioinformatics*.

30(21):3128–30

Shackelford G, Karplus K. 2007. Contact prediction using mutual information and neural nets. *Proteins*. 69 Suppl 8:159–64

Simonetti FL, Teppa E, Chernomoretz A, Nielsen M, Marino Buslje C. 2013. MISTIC: Mutual information server to infer coevolution. *Nucleic Acids Res*. 41(Web Server issue):W8-14

Teppa E, Wilkins AD, Nielsen M, Buslje CM. 2012. Disentangling evolutionary signals: conservation, specificity determining positions and coevolution. Implication for catalytic residue prediction. *BMC Bioinformatics*. 13:235

Tetchner S, Kosciolek T, Jones DT. 2014. Opportunities and limitations in applying coevolution-derived contacts to protein structure prediction. *Bio-Algorithms and Med-Systems*. 10(4):

Thompson JN. 1994. *The Coevolutionary Process*. University of Chicago Press

Thomsen MCF, Nielsen M. 2012. Seq2Logo: a method for construction and visualization of amino acid binding motifs and sequence profiles including sequence weighting, pseudo counts and two-sided representation of amino acid enrichment and depletion. *Nucleic Acids Res*. 40(Web Server issue):W281-7

Tillier ERM, Lui TWH. 2003. Using multiple interdependency to separate functional from phylogenetic correlations in protein alignments. *Bioinformatics*. 19(6):750–55

Weigt M, White RA, Szurmant H, Hoch JA, Hwa T. 2009. Identification of direct residue contacts in protein-protein interaction by message passing. *Proc Natl Acad Sci U S A*. 106(1):67–72

Widenius M, Axmark D. 2002. *Mysql Reference Manual*. Sebastopol, CA, USA:

O'Reilly & Associates, Inc. 1st ed.

Yanofsky C, Horn V, Thorpe D. 1964. PROTEIN STRUCTURE RELATIONSHIPS REVEALED BY MUTATIONAL ANALYSIS. *Science*. 146(3651):1593–94

Yu A, Chen J. 1995. *The POSTGRES95 User Manual*