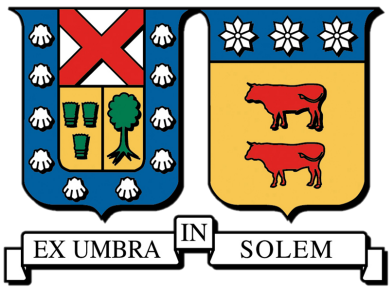
2019

IMPLEMENTACIÓN Y COMPARACIÓN DE FORMULACIONES INTEGRALES DE FRONTERA PARA EL CÁLCULO DE ENERGÍAS DE SOLVATACIÓN EN MACROMOLÉCULAS DISUELTAS EN UN MEDIO SALINO

FERNÁNDEZ MARTÍNEZ, PAULO ABEL

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INGENIERÍA MECÁNICA
VALPARAÍSO, CHILE



IMPLEMENTACIÓN Y COMPARACIÓN DE FORMULACIONES INTEGRALES DE FRONTERA PARA EL CÁLCULO DE ENERGÍAS DE SOLVATACIÓN EN MACROMOLÉCULAS DISUELTAS EN UN MEDIO SALINO

PAULO ABEL FERNÁNDEZ MARTÍNEZ

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL MECÁNICO MENCIÓN ENERGÍA

Profesor Guía: PhD Christopher Cooper Villagrán
Profesor Correferente: PhD Alejandro Pacheco Sanjuán

ENERO - 2019

Resumen:

En el contexto de la simulación de macromoléculas implicadas en procesos biológicos, la energía de solvatación es información importante para analizar el comportamiento, afinidad e interacciones entre moléculas. Para obtenerla es posible utilizar modelos simplificados basados en electrostática de medio continuo, mediante la implementación de la ecuación de Poisson-Boltzmann. El cálculo se puede realizar utilizando distintas formulaciones a partir de las mismas ecuaciones, ya sea resolviendo las ecuaciones diferenciales, o en forma de ecuaciones integrales de frontera.

Este trabajo expone los resultados obtenidos mediante el método de elementos de borde (BEM) utilizando las distintas formulaciones disponibles, implementado en la librería `bempp`. La implementación de las formulaciones en el mismo programa permite realizar una comparación directa, logrando determinar diferencias entre los tiempos de resolución, costos de memoria, y órdenes de convergencia para cada caso.

Palabras Clave: Energía de solvatación, Potencial electrostático, Método de elementos de borde, Formulación integral, bempp, Macromolécula

Abstract:

In the simulation of macromolecules in biological processes context, the solvation energy brings important information about the behavior, affinity and interactions between molecules. To obtain it, it is possible to use simplified models based in continum medium electrostatics, through the Poisson Boltzmann equation implementation. The computations can be done using different formulations from the same equations, solving differential equations, or solving the equations in its integral form.

This work shows the resutls obtained with border element method (BEM) using the differents formulations available, implemented on the `bempp` library. The implementation of diffrent formulations in the same program makes possible a direct comparation, determining diffrences between resolution times, memory costs and convergence orders in each case.

Keywords: Solvation Energy, Electrostatic Potential, Boundary elements method, Integral formulation, bempp, Macromolecule

Glosario

- **Energía de Solvatación:** Diferencia de energía provicada por sumergir una molécula en un medio acuoso salino
- **Solvente Continuo:** Modelo simplificado para tratar como un continuo una solución de agua e iones.
- **Potencial Electrostático:** Potencial en el espacio provocado por la presencia de un campo eléctrico.
- **Potencial de Reacción:** Potencial eléctrico ejercido por el solvente sobre una molécula sumergida.
- **Superficie de Exclusión:** Distancia mínima a la cual tienen acceso las moléculas de agua e iones en torno a la molécula.
- **Análisis Numérico:** Conjunto de algoritmos y métodos que permiten aproximar modelos matemáticos complejos.
- **Boundary Element Method:** Método numérico que permite aproximar ecuaciones diferenciales por medio de una formulación integral sobre las superficies que delimitan el dominio.
- **Python:** Lenguaje de programación de alto nivel de uso general
- **bempp:** Librería con interfaz programable en Python que permite resolver problemas con el método de elementos de borde.

Índice General

1 Introducción

En el estudio de interacciones moleculares, existe un amplio trabajo teórico y experimental, donde han tenido cada vez mayor aceptación los resultados obtenidos a través de simulaciones computacionales. Esta disciplina permite estudiar fenómenos con la ayuda de modelos matemáticos que evitan recrear gran parte de dichos fenómenos de manera experimental. En este contexto, el buen uso de las herramientas computacionales disponibles se vuelve fundamental para el avance eficiente de una investigación.

Los modelos utilizados en dinámica molecular explican y predicen de buena manera los fenómenos estudiados, sin embargo, la cantidad y complejidad de cálculos que es necesario realizar muchas veces vuelve inviable la aplicación directa de dichos modelos. Esto se debe principalmente al gran número de partículas que componen un sistema relativamente sencillo, volviendo poco escalable la simulación directa para sistemas complejos.

En la actualidad no existe un modelo alternativo para la simulación de sistemas dinámicos, como en el caso del plegamiento de proteínas, donde es necesario considerar cada una de las partículas involucradas. Sin embargo, para el estudio de casos estáticos es posible realizar ciertas aproximaciones, donde la molécula se encuentra en una configuración definida, o bien se sabe la posición inicial y final del sistema, obviando de esta forma la dinámica y el mecanismo involucrado y enfocando el estudio en el aumento o disminución de energía total del sistema en el proceso.

La energía de disolución en un sentido físico, es la diferencia de potencial que existe entre un conjunto de cargas en el vacío, y el potencial del mismo conjunto de cargas inmerso en un medio, en este caso una solución salina. Dicha energía da una idea de la probabilidad de ocurrencia de un fenómeno a nivel molecular, ya que la naturaleza siempre tiende a moverse en los estados de mínima energía. Este tipo de resultados permite estimar estados más probables entre un grupo de configuraciones posibles, ya sea para una molécula aislada o un conjunto más grande, es por ello que se plantea la búsqueda de algoritmos y formulaciones que permitan realizar este tipo de cálculos, que pueden llegar a ser importantes para el desarrollo de simulaciones de sistemas complejos que hasta ahora no se pueden modelar de manera directa.

En este campo existen diversas alternativas para realizar aproximaciones que permitan calcular estados energéticos. Una de ellas es considerar el entorno de la molécula y su entorno como medios continuos, en lo que se conoce como método de solvente implícito. Con este método se puede discretizar el espacio mediante difer-

encias o volúmenes finitos, lo que presenta una primera problemática al momento de definir el solvente, y una segunda al considerar una malla que adopte de manera correcta a la posición irregular de átomos en una molécula.

Este trabajo en particular pretende utilizar un método que evita discretizar todo el espacio al interior y exterior de la molécula, considerando en primer lugar la molécula en estudio como un conjunto de cargas puntuales, y en segundo lugar el entorno salino de la molécula como un continuo, definiendo para ello una *superficie de exclusión*, es decir, una frontera que separa el medio exterior del conjunto de átomos que componen la molécula.

El método numérico empleado en este trabajo es el método de elementos de borde, una discretización de la formulación integral del modelo, con ello se puede calcular el potencial electrostático solo sobre la superficie de exclusión. Esto evita modelar dominio completo, haciendo posible obtener el potencial electrostático en cualquier punto del dominio una vez resuelto el sistema y encontrada la solución del potencial electrostático sobre la frontera de la molécula.

Otra ventaja que presenta el método de elementos de borde, es que permite la simulación en un medio infinito en torno a la molécula, ya que la formulación integral incorpora implícitamente, como una integral impropia, una condición de potencial cero al alejarse de la frontera de la molécula.

Existe una gran variedad de programas disponibles que permiten calcular la energía de solvatación (entre otros parámetros) de una molécula disuelta. En la mayoría de los casos el modelo empleado no siempre se encuentra disponible de manera explícito, y por lo general no existe la posibilidad de modificar dicho modelo e implementar nuevas metodologías de resolución. En este contexto el uso de una herramienta que permita realizar modificaciones de manera sencilla al programa sirve para comparar diferencias entre las distintas formulaciones y métodos.

2 Formulación teórica

2.1 Modelo estudiado

El modelo propuesto se basa en una superficie de exclusión, que separa la región de la molécula y la región del solvente. El volumen que se encuentra en el interior de la superficie de exclusión, es una región donde los átomos que componen la molécula se consideran como cargas puntuales en un medio vacío. A esta región interna no pueden acceder moléculas de agua o iones de sal.

Esta región interior puede ser modelada mediante una ecuación de Laplace (o Poisson) como se muestra en la ecuación (1), considerando un potencial electrostático ϕ, un conjunto de cargas q_k, con una constante dieléctrica ϵ_1.

$$\nabla^2 \phi_1 (r) = -\frac{1}{\epsilon_1} \sum_k q_k \delta (r - r_k) \tag{1}$$

Se define entonces una superficie de exclusión a través de una molécula sonda, una molécula que no puede acceder al conjunto de átomos de la molécula, como se muestra en la Figura 1

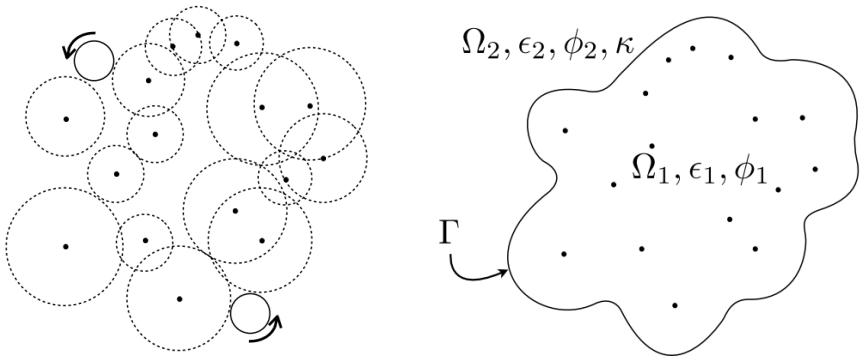


Figura 1: Esquema de sondeo de la superficie de exclusión con una molécula de agua (Extraído de Cooper 2015 [1])

Por otro lado, para evitar calcular cada una de las interacciones en una gran cantidad de moléculas de agua e iones, el exterior de la superficie de exclusión se considera como un medio dieléctrico continuo. Este medio tiene una distribución de partículas, por lo que la densidad de carga queda representada como una suma de cargas puntuales, desarrollada en la ecuación (2)

$$\nabla^2 \phi = -\frac{\rho}{\epsilon} = -\frac{1}{\epsilon} \sum_i q_i c_i(r) = -\frac{1}{\epsilon} \sum_i q_i c_o \exp\left(\frac{-\phi q_i}{k_B T}\right) \tag{2}$$

donde K_B es la constante de Boltzmann y T es la temperatura absoluta del medio. Para simplificar la ecuación, se hace una aproximación de primer orden mediante una expansión de Taylor (siempre que $\phi q << k_B T$) como se muestra en (3).

$$\exp\left(\frac{-\phi q(r)}{k_B T}\right) \approx 1 + \frac{-\phi q_i}{k_B T} \tag{3}$$

Se obtiene de esta manera la ecuación Poisson-Boltzmann Linealizada (4), donde $\phi\,(r)$ es el potencial electrostático en la coordenada r, y κ es la inversa de la longitud de Debye (distancia de influencia electrostática) propia del campo exterior.

$$\nabla^2 \phi_2 - \kappa^2 \phi_2 = 0 \tag{4}$$

Con estas ecuaciones queda todo el dominio completamente definido, es decir, se podría teóricamente obtener el potencial en cualquier punto del dominio, tanto dentro como fuera de la superficie de exclusión. El sistema de ecuaciones, como se ha mencionado, se puede armar discretizando el espacio o, como lo veremos más adelante, discretizando la frontera entre las dos regiones.

Las condiciones de interfaz que existen en la superficie de exclusión consideran en primer lugar, la continuidad del potencial electrostático sobre la frontera, es decir, el potencial debe ser el mismo tanto en el exterior como el interior inmediato de la superficie (5). La segunda es una condición que considera el cambio de medio, para esto se define la derivada normal del potencial sobre la superficie, ponderada con la constante dieléctrica de cada región (6).

$$\phi_1 = \phi_2 \tag{5}$$

$$\epsilon_1 \frac{\partial \phi_1}{\partial \mathbf{n}} = \epsilon_2 \frac{\partial \phi_2}{\partial \mathbf{n}} \tag{6}$$

La energía potencial electrostática de un de un cuerpo cargado puede ser calculada mediante la ecuación (7), para cualquier densidad de carga $\rho(r)$, y cualquier distribución de potencial $\phi(r)$, en este caso se puede obtener la energía de solvatación calculando la energía electrostática de la molécula, es decir, integrando sobre el dominio dentro del la superficie de exclusión (Ω_1 en la Figura 1).

$$E = \frac{1}{2} \int_{\Omega} \phi\,(r)\, \rho\,(r)\ \ \mathrm{d}\Omega \tag{7}$$

Cabe considerar que el potencial utilizado en esta definición es el potencial de

reacción, este solo considera la diferencia entre el potencial de las partículas en el vacío y el potencial de las partículas en el solvente. No considera el potencial de Coulomb, ya que al ser cargas puntuales dicho potencial sería infinito.

$$\phi_{reac} = \phi_{total} - \phi_{Coulomb} \tag{8}$$

Ya que en este caso la carga se encuentra concentrada en puntos en el espacio, la distribución de carga queda definida por cargas puntuales, es decir, la suma de las cargas individuales por deltas de Dirac que indican cada posición. Al manipular la ecuación resultante, se obtiene la ecuación (9) que permite calcular la energía de solvatación de un conjunto de cargas puntuales.

$$E = \frac{1}{2}\int_{\Omega} \phi\left(r\right) \sum_{i} q_i \delta\left(r_i\right) \ \mathrm{d}\Omega = \frac{1}{2}\sum_{k} q_i \int_{\Omega} \phi\left(r\right) \delta\left(r_i\right) \ \mathrm{d}\Omega$$

$$E = \frac{1}{2}\sum_{i} q_i \phi\left(r_i\right) \tag{9}$$

3 Formulación integral del problema

Ya que el problema en este caso se compone de dos regiones separadas por una única superficie, es conveniente implementar el método de elementos de borde, el cual nos permite reducir el problema que originalmente está definido sobre todo el dominio, a una única superficie que separa el dominio en las regiones mencionadas anteriormente. Una vez obtenido el potencial sobre dicha superficie, es posible calcular el potencial en cualquier punto del dominio como se desarrolla en esta sección. Como se ya se ha indicado, esto nos permite, por un lado, evitar la discretización completa del dominio, y por otro, definir un dominio teóricamente infinito fuera de la superficie de exclusión.

Para ilustrar la lógica del método se considera la ecuación de Laplace con las siguientes condiciones de borde

$$\nabla^2 \phi = 0 \tag{10}$$

$$\phi = \phi_0$$
$$\frac{\partial \phi}{\partial \mathbf{n}} = \frac{\partial \phi_0}{\partial \mathbf{n}}$$

se integra sobre todo el dominio Ω con una función peso w, en lo que se denomina una formulación débil de la ecuación original.

$$\int_\Omega \nabla^2 \phi \left(r'\right) w \left(r'\right) \ \mathrm{d}\Omega = 0 \tag{11}$$

se utiliza como función peso una función de Green para la ecuación de Laplace, solución fundamental de la ecuación original

$$\int_\Omega \nabla^2 \phi \left(r'\right) \mathrm{G} \left(r, r'\right) \ \mathrm{d}\Omega = 0 \tag{12}$$

la ecuación puede ser reescrita de la siguiente manera

$$\int_\Omega \left[\nabla \cdot \left(\nabla \phi \left(r'\right) \mathrm{G} \left(r, r'\right) - \phi \left(r'\right) \nabla \mathrm{G} \left(r, r'\right)\right) + \phi \left(r'\right) \nabla^2 \mathrm{G} \left(r, r'\right)\right] \ \mathrm{d}\Omega = 0 \tag{13}$$

al aplicar el teorema de la divergencia al primer término, con Γ como superficie del dominio Ω

$$\int_{\Gamma} \mathrm{G}\left(r, r'\right) \nabla\phi\left(r'\right) \cdot \mathbf{n} \ \mathrm{d}r' - \int_{\Gamma} \phi\left(r'\right) \nabla \mathrm{G}\left(r, r'\right) \cdot \mathbf{n} \ \mathrm{d}r' - \int_{\Omega} \phi\left(r'\right) \delta\left(r\right) \ \mathrm{d}\Omega = 0 \quad (14)$$

el último término puede ser reducido mediante la definición de la función Delta Dirac, y sabiendo que la componente normal del gradiente no es más que la derivada normal del campo, obtenemos la ecuación (15)

$$\phi\left(r\right) = \int_{\Gamma} \mathrm{G}\left(r, r'\right) \frac{\partial\phi}{\partial\mathbf{n}}\left(r'\right) \ \mathrm{d}r' - \int_{\Gamma} \phi\left(r'\right) \frac{\partial \mathrm{G}}{\partial\mathbf{n}}\left(r, r'\right) \ \mathrm{d}r' \quad (15)$$

con esta ecuación y conociendo el valor del potencial sobre la superficie (Γ).

Para dominios con ecuaciones distintas, el proceso es idéntico, ya que para obtener el potencial en cualquier punto deseado, basta con utilizar la función de Green correspondiente a la ecuación que gobierna el dominio.

En el caso de la ecuación (15) existen solo dos operadores integrales debido a que solo se estudia una superficie. Para agregar más superficies, basta con agregar más operadores que integren sobre el resto de las superficies presentes en el modelo, la ecuación que permite obtener el potencial con dos superficies (Γ_1 y Γ_2) quedaría como se muestra en la ecuación (16)

$$\begin{aligned} \phi\left(r\right) = & \int_{\Gamma 1} \mathrm{G}\left(r, r'\right) \frac{\partial\phi}{\partial\mathbf{n}}\left(r'\right) \ \mathrm{d}r' - \int_{\Gamma 1} \phi\left(r'\right) \frac{\partial \mathrm{G}}{\partial\mathbf{n}}\left(r, r'\right) \ \mathrm{d}r' \\ & + \int_{\Gamma 2} \mathrm{G}\left(r, r'\right) \frac{\partial\phi}{\partial\mathbf{n}}\left(r'\right) \ \mathrm{d}r' - \int_{\Gamma 2} \phi\left(r'\right) \frac{\partial \mathrm{G}}{\partial\mathbf{n}}\left(r, r'\right) \ \mathrm{d}r' \end{aligned} \quad (16)$$

3.1 Formulación del método de elementos de borde

Para poder obtener el potencial electrostático y resolver el problema propuesto, es necesario definir el sistema de ecuaciones. Este sistema se forma al evaluar las dos ecuaciones que describen cada región, sobre el area que comparten, es decir, la superficie de exclusión.

A continuación se presenta la ecuación (17) que es la ecuación (1) en su forma integral, la cual describe el interior de la superficie de exclusión y como queda evidenciado en el lado derecho de la ecuación, donde se encuentran las cargas puntuales. En segundo lugar la ecuación (18) es la forma integral de la ecuación (4) que representa el solvente continuo en el exterior mediante de un campo de Poisson-Boltzmann linealizado.

La ecuación de Poisson-Boltzmann linealizada también se le llama ecuación Poisson apantallada o equación de Helmholtz modificada.

$$\phi_1(r) + \int_\Gamma \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial \mathbf{n}}(r, r') \ \mathrm{d}r' - \int_\Gamma \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial \mathbf{n}}(r') \ \mathrm{d}r' = \frac{1}{4\pi\epsilon_1} \sum_k \frac{q_k}{|r - r_k|} \tag{17}$$

$$\phi_2(r) - \int_\Gamma \phi_2(r') \frac{\partial \mathrm{G}_{PB}}{\partial \mathbf{n}}(r, r') \ \mathrm{d}r' + \int_\Gamma \mathrm{G}_{PB}(r, r') \frac{\partial \phi_2}{\partial \mathbf{n}}(r') \ \mathrm{d}r' = 0 \tag{18}$$

Cabe destacar el signo que recibe cada una de las integrales, son opuestos en cada ecuación, esto debido a la orientación del vector normal al momento de integrar sobre el interior o exterior de la frontera.

Como se ha mencionado, G_L y G_{PB} a las funciones de Green para la ecuación de Laplace y Poisson-Boltzmann respectivamente, y se definen como se muestra en las ecuaciones (19) y (20).

$$G_L(r, r') = \frac{1}{4\pi|r - r'|} \tag{19}$$

$$G_{PB}(r, r') = \frac{e^{-\kappa|r-r'|}}{4\pi|r - r'|} \tag{20}$$

El sistema de ecuaciones obtenido hasta ahora se puede resolver de manera directa, ya que estas ecuaciones representan el potencial en cualquier punto del dominio, incluyendo puntos que no comparten. Para definir el sistema de ecuaciones de manera correcta, es necesario llevar las ecuaciones al conjunto de puntos que comparten ambas regiones, es decir, es necesario evaluar las ecuaciones en los puntos que se encuentran sobre la superficie de excluisión.

Al estar llevando los puntos de evaluación justo sobre la frontera, se considera en cada ecuación la mitad del potencial de cada lado de la frontera como se muestra en la Figura 2.

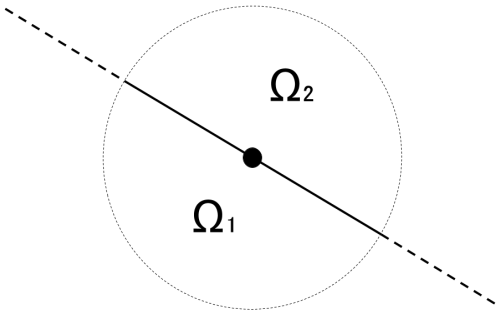


Figura 2: Esquema de punto de evaluación sobre elemento de borde plano

Las condiciones de contorno sobre la superficie son las mencionadas anteriormente en las ecuaciones (5) y (6), la condición de contorno de la derivada se modifica levemente para despejar el potencial externo y dejarlo en función del potencial interno,

$$\phi_1 = \phi_2$$
$$\frac{\epsilon_1}{\epsilon_2}\frac{\partial \phi_1}{\partial \mathbf{n}} = \frac{\partial \phi_2}{\partial \mathbf{n}}$$

Para evaluar las ecuaciones en el borde es necesario calcular el límite cuando la coordenada r' se acerca a la superficie de exclusión. A partir de la segunda identidad de Green,

$$\int_\Omega \left(\nabla^2 \phi G_L - \phi \nabla^2 G_L\right) \mathrm{d}\Omega = \int_\Gamma G_L \frac{\partial}{\partial \mathbf{n}}\phi - \phi \frac{\partial}{\partial \mathbf{n}} G_L \mathrm{d}\Gamma \tag{21}$$

se aplica el límite en ambos términos de lado derecho desde el interior de la superficie. El operador (19) y su derivada normal son los que se utilizan en este caso. Se simboliza como ε el módulo del término $|r - r'|$, el cual también es colineal al vector normal $\mathbf{n}$. Ya que solo se integra sobre el dominio interno, solo se considera un hemisferio del potencial total.

Primer término del lado derecho de (21)

$$\lim_{\varepsilon \to 0} \frac{1}{4\pi} \int_{\Gamma_{hem}} \frac{\partial}{\partial \mathbf{n}} \phi(r') \frac{1}{|r - r'|} \mathrm{d}\Gamma' = \lim_{\varepsilon \to 0} \frac{1}{4\pi\varepsilon} \int_{\Gamma_{hem}} \frac{\partial}{\partial \mathbf{n}} \phi(r') \mathrm{d}\Gamma' \tag{22}$$

$$\lim_{\varepsilon \to 0} \frac{1}{4\pi\varepsilon} \frac{\partial}{\partial \mathbf{n}} \phi(r) \int_{\Gamma_{hem}} \mathrm{d}\Gamma' = \lim_{\varepsilon \to 0} \frac{1}{4\pi\varepsilon} \frac{\partial}{\partial \mathbf{n}} \phi(r) 2\pi\varepsilon^2 = 0$$

Segundo término del lado derecho de (21)

$$\lim_{\varepsilon \to 0} -\frac{1}{4\pi} \int_{\Gamma_{hem}} \phi(r') \frac{(r - r') \cdot \mathbf{n}}{|r - r'|^3} \mathrm{d}\Gamma' = \lim_{\varepsilon \to 0} -\frac{\varepsilon}{4\pi\varepsilon^3} \int_{\Gamma_{hem}} \phi(r') \mathrm{d}\Gamma' \tag{23}$$

$$\lim_{\varepsilon \to 0} -\frac{1}{4\pi\varepsilon^2} \phi(r) \int_{\Gamma_{hem}} \mathrm{d}\Gamma' = \lim_{\varepsilon \to 0} -\frac{1}{4\pi\varepsilon^2}\, \phi(r)\, 2\pi\varepsilon^2 = -\frac{\phi(r)}{2}$$

Finalmente, evaluando solo la mitad del potencial en el primer término de las ecuaciones (17) y (18), reemplazando las condiciones de contorno para que el sistema completo para que quede en función del potencial interno, el sistema de ecuaciones queda de la siguiente forma

$$\frac{\phi_1}{2}(r) + \int_\Gamma \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \ \mathrm{d}r' - \int_\Gamma \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial n}(r') \ \mathrm{d}r' = \frac{1}{4\pi\epsilon_1} \sum_k \frac{q_k}{|r - r_k|} \tag{24}$$

$$\frac{\phi_1}{2}(r) - \int_\Gamma \phi_1(r') \frac{\partial \mathrm{G}_{PB}}{\partial n}(r, r') \ \mathrm{d}r' + \epsilon \int_\Gamma \mathrm{G}_{PB}(r, r') \frac{\partial \phi_1}{\partial n}(r') \ \mathrm{d}r' = 0 \tag{25}$$

donde ϵ es el cociente entre el la constante dieléctrica interna y externa obtenido al aplicar las condiciones de frontera del sistema. Al resolver el sistema de ecuaciones se obtiene el potencial sobre la superficie de exclusión y con ello se utilizan las ecuaciones (17) y (9) para obtener el potencial al interior de la molécula y la energía de solvatación respectivamente.

Queda entonces definir el sistema de manera matricial y obtener una metodología de resolución que se pueda implementar.

3.2 Operadores Integrales

Para facilitar la lectura de las ecuaciones se introduce una notación para los operadores integrales utilizados en este trabajo. Cada operador lleva necesita de una variable, en nuestro caso el potencial electrostático $\phi(x)$, una superficie de integración, y una función de Green asociada.

La variable x es la coordenada donde se desea conocer el potencial, mientras que la variable y es la coordenada sobre la superficie de integración. El *subíndice* n indica la función de Green que contiene el operador, y el *superíndice* a la superficie sobre la cual está integrando.

De la variedad de operadores disponibles en la librería de `bempp`, solo se utilizan los siguientes cuatro :

Single Layer

$$V_n^a\left[\phi(x)\right] = \int_{\Gamma a} G_n(x,y)\phi(y)\mathrm{d}y$$

Double Layer

$$K_n^a\left[\phi(x)\right] = \int_{\Gamma a} \frac{\partial G_n(x,y)}{\partial n(y)}\phi(y)\mathrm{d}y$$

Adjoint Double Layer

$$K_n'^a\left[\phi(x)\right] = \int_{\Gamma a} \frac{\partial G_n(x,y)}{\partial n(x)}\phi(y)\mathrm{d}y$$

Hypersingular Layer

$$D_n^a\left[\phi(x)\right] = -\frac{\partial}{\partial n(x)}\int_{\Gamma a} \frac{\partial G_n(x,y)}{\partial n(y)}\phi(y)\mathrm{d}y$$

3.3 Formulación Directa

El modelo que en adelante se trata como Formulación directa, es aquel que no considera ninguna modificación a la formulación integral original de las ecuaciones, un desarrollo más elaborado de dicha formulación se puede encontrar en la publicación de Yoon & Lenhoff [8]. Ya solo considera una única superficie, que es la superficie de exclusión del agua, por esta razón, esta vez se omite el *superíndice a* en los operadores presentados previamente.

El sistema de ecuaciones integrales tal como se encuentran planteadas en (24) y (25), escritas de forma vectorial y con la notación de operadores introducida previamente queda de la siguiente forma.

$$\begin{bmatrix} \frac{I}{2} + K_L & -V_L \\ \frac{I}{2} - K_{PB} & +\epsilon V_{PB} \end{bmatrix} \cdot \begin{Bmatrix} \phi \\ \frac{\partial \phi}{\partial n} \end{Bmatrix} = \begin{Bmatrix} \sum q_k G_L\left(r, r_k\right) \\ 0 \end{Bmatrix} \tag{26}$$

3.4 Formulación de Juffer

La formulación elaborada por A. Juffer [9] consiste en modificar el sistema buscando mejorar su condicionamiento, de esta manera se asegura la convergencia del sistema de ecuaciones mediante métodos iterativos de resolución, y al mismo tiempo el *solver* utilizado requiere de menos iteraciones para converger al mismo resultado que la formulación directa.

3.4.1 Formulación integral de Juffer

Esto se logra al sumar las ecuaciones 24 y 25, obteniendo la siguiente ecuación

$$\frac{1}{2}\left(1+\epsilon\right)\phi\left(r\right) = V_L\left(\frac{\partial \phi}{\partial n}\right) - K_L\left(\phi\right) - V_{PB}\left(\frac{\partial \phi}{\partial n}\right) + \epsilon K_{PB}\left(\phi\right) + \sum_k q_k G_L\left(r, r_k\right)$$

ya que el sistema requiere de dos ecuaciones para poder ser resuelto, se utiliza como segunda ecuación, la derivada de la ecuación anterior.

$$\frac{1}{2}\left(1+\frac{1}{\epsilon}\right)\frac{\partial \phi}{\partial n}\left(r\right) = K_L'\left(\frac{\partial \phi}{\partial n}\right) + D_L\left(\phi\right) - \frac{1}{\epsilon}K_{PB}'\left(\frac{\partial \phi}{\partial n}\right) - D_{PB}\left(\phi\right) + \frac{\partial}{\partial n}\sum_k q_k G_L\left(r, r_k\right)$$

Para efectos de lectura y facilidad a la hora de implementar la formulación en el código, se introducen lo siguientes operadores que permiten resumir las dos ecuaciones obtenidas,

$$
\begin{aligned}
L_1 &= \epsilon K_{PB} - K_L \\
L_2 &= V_L - V_{PB} \\
L_3 &= D_{PB} - D_L \\
L_4 &= K'_L - \frac{1}{\epsilon} K'_{PB}
\end{aligned}
\tag{27}
$$

quedando el sistema de ecuaciones en su forma matricial con notación vectorial de la siguiente manera.

$$
\begin{bmatrix} \frac{I}{2}(1+\epsilon)I - L_1 & -L_2 \\ -L_3 & \frac{I}{2}(1+1/\epsilon)I - L_4 \end{bmatrix} \cdot \begin{Bmatrix} \phi \\ \frac{\partial \phi}{\partial n} \end{Bmatrix} = \begin{Bmatrix} \sum q_k V_k \\ \frac{\partial}{\partial n} \sum q_k V_k \end{Bmatrix} \tag{28}
$$

3.5 Stern Layer

El modelo estudiado se basa en la idea de asumir el solvente en torno a la molécula como un medio dieléctrico continuo, el cual representaría los iones disueltos en el medio como una distribución de cargas fuera de la superficie de exclusión. Esto resulta una buena aproximación para sectores lejanos a la molécula, sin embargo, al representar la concentración de carga como una variable del campo pueden presentarse situaciones anómalas en la región más próxima a la superficie de exclusión.

Ya que existen zonas cargadas sobre la superficie de exclusión, es de esperar que la concentración de carga aumente en torno a estas zonas, lo que en el caso del solvente continuo puede provocar una concentración de carga superior al máximo teórico, ya que en la realidad existe un número limitado de iones que pueden juntarse sobre la superficie cargada debido a la geometría que poseen, la distribución de partículas de agua e iones se esquematiza en la Figura 3.

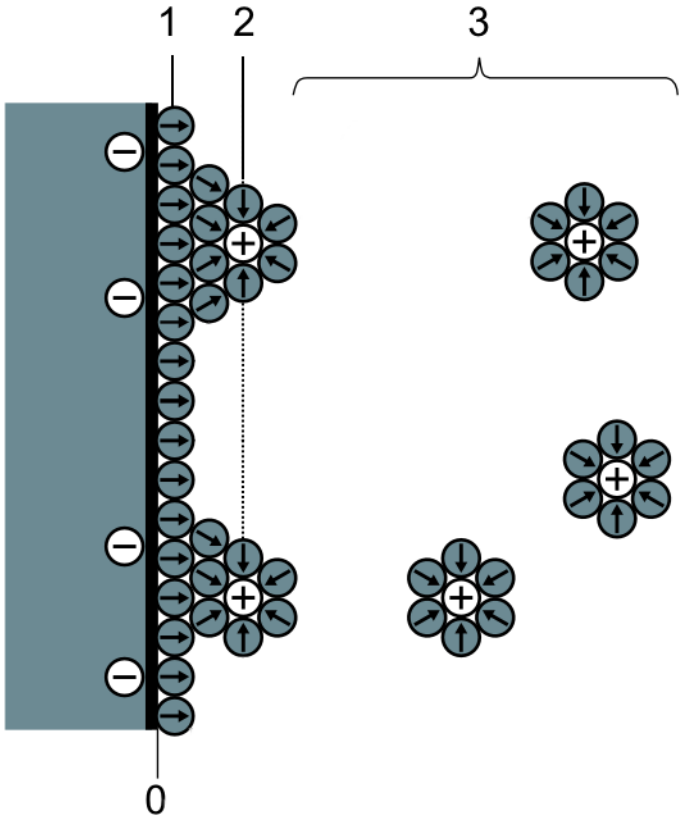


Figura 3: Distribución de partículas en torno a una superficie cargada,
(Imagen por Tosaka, Electric double-layer (BMD model) NT.PNG, Wikipedia)

En el esquema el 0 marca la posición de la superficie de exclusión de la molécula, el 1 es la primera capa de moléculas de agua (también llamado plano de Helmholtz), el 2 indica la distancia mínima a la que pueden aproximarse iones solvatados, y 3 la zona difusa, la cual se encuentra alejada de la superficie de la molécula.

Para evitar esta alta concentración provocada por el modelo de solvente continuo, se utiliza una segunda capa llamada *Stern Layer*. Este modelo fue propuesto propuesto por el físico alemán Otto Stern, y considera una superficie de exclusión para los iones por sobre la superficie de exclusión del agua.

3.5.1 Formulación teórica

En este caso se agrega una zona intermedia a la región que contiene los átomos de la molécula, y la región externa con el solvente, esta región intermedia al no poseer cargas, se puede modelar con una ecuación de Laplace homogénea.

Tanto la región interna que contiene la molécula, como la externa del solvente siguen siendo exactamente iguales que en la formulación original. Dentro de la región intermedia al solo existir moléculas de agua se considera una permitividad eléctrica constante igual a la del solvente, sin embargo al no tener presencia de iones, no existe

la constante κ. La capa Stern se modela con una ecuación de Laplace homogénea y el sistema de ecuaciones queda expresado como se muestra en las ecuaciones (29), (30) y (31).

$$\nabla^2\phi_1(r) = -\frac{1}{\epsilon_1}\sum_k q_k\delta(r - r_k) \tag{29}$$

$$\nabla^2\phi_2(r) = 0 \tag{30}$$

$$\nabla^2\phi_3 - \kappa\phi_3 = 0 \tag{31}$$

Las condiciones de borde que se aplican en este caso son similares a la formulación directa de superficie única. En la primera interfaz existe una continuidad de potencial electrostático, y para su derivada normal existe una condición de cambio de medio.

$$\begin{aligned}\phi_1 &= \phi_2\\ \epsilon_1\frac{\partial\phi}{\partial\mathbf{n}_1} &= \epsilon_2\frac{\partial\phi}{\partial\mathbf{n}_2}\end{aligned}$$

Para la interfaz exterior existe continuidad para el potencial, y en el caso de la derivada normal, ya que la permitividad es la misma en ambos medios, se anulan dejando las condiciones de la siguiente manera

$$\begin{aligned}\phi_2 &= \phi_3\\ \frac{\partial\phi}{\partial\mathbf{n}_2} &= \frac{\partial\phi}{\partial\mathbf{n}_3}\end{aligned}$$

3.5.2 Formulación integral con *Stern Layer*

Como se hace en el caso de la formulación directa con una única superficie, es necesario expresar las ecuaciones de forma integral. Las ecuaciones (29), (30) y (31) quedan de la forma mostrada en (32), (33) y (34) respectivamente. Estas ecuaciones sirven para obtener el potencial en cualquier punto dentro de cada región.

$$\phi_1(r) + \int_{\Gamma 1}\phi_1(r')\frac{\partial \mathrm{G}_L}{\partial n}(r, r')\ \mathrm{d}r' - \int_{\Gamma 1}\mathrm{G}_L(r, r')\frac{\partial\phi_1}{\partial n}(r')\ \mathrm{d}r' = \frac{1}{4\pi\epsilon_1}\sum_k\frac{q_k}{|r - r_k|} \tag{32}$$

$$\phi_2(r) - \int_{\Gamma 1} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \ \mathrm{d}r' + \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \ \mathrm{d}r' + \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \ \mathrm{d}r' - \int_{\Gamma 2} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \ \mathrm{d}r' = 0 \tag{33}$$

$$\phi_3(r) - \int_{\Gamma 2} \phi_3(r') \frac{\partial \mathrm{G}_{PB}}{\partial n}(r, r') \ \mathrm{d}r' + \int_{\Gamma 2} \mathrm{G}_{PB}(r, r') \frac{\partial \phi_3}{\partial n}(r') \ \mathrm{d}r' = 0 \tag{34}$$

Luego se deben evaluar las ecuaciones sobre cada una de las superficies para crear el sistema de ecuaciones. A diferencia del caso de una superficie, en la ecuación (33) existen operadores sobre cada superficie, por lo que quedan términos cruzados en el sistema de ecuaciones final.

Para la resolución al momento de utilizar las condiciones de contorno, se utiliza el campo interior $\phi_1(r)$ sobre la superficie interior y el campo intermedio $\phi_2(r)$ sobre la superficie externa.

3.5.3 Ecuaciones en el Borde interno

Se dejan ambas ecuaciones en función del potencial interno sobre la superficie interna, considerando que la permitividad no se mantiene en el primer borde ($\epsilon_1 \neq \epsilon_2$).

$$\phi_1 = \phi_2$$
$$\frac{\epsilon_1}{\epsilon_2} \frac{\partial \phi_1}{\partial \mathbf{n}} = \frac{\partial \phi_2}{\partial \mathbf{n}}$$

$$\frac{\phi_1}{2}(r) + \int_{\Gamma 1} \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \ \mathrm{d}r' - \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial n}(r') \ \mathrm{d}r' = \frac{1}{4\pi\epsilon_1} \sum_k \frac{q_k}{|r - r_k|} \tag{35}$$

$$\frac{\phi_1}{2}(r) - \int_{\Gamma 1} \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \ \mathrm{d}r' + \frac{\epsilon_1}{\epsilon_2} \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial n}(r') \ \mathrm{d}r' + \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \ \mathrm{d}r' - \int_{\Gamma 2} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \ \mathrm{d}r' = 0 \tag{36}$$

3.5.4 Ecuaciones en el Borde Externo

Sobre la segunda superficie se mantiene la permitividad del medio ($\epsilon_2 = \epsilon_3$), y se dejan todos los operadores en función del potencial electrostático de la capa Stern.

$$\phi_2 = \phi_3$$
$$\frac{\partial \phi_2}{\partial \mathbf{n}} = \frac{\partial \phi_3}{\partial \mathbf{n}}$$

$$\frac{\phi_2}{2}(r) - \int_{\Gamma 1} \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' + \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\epsilon_1}{\epsilon_2} \frac{\partial \phi_1}{\partial n}(r') \, \mathrm{d}r' + \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' - \int_{\Gamma 2} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \, \mathrm{d}r' = 0 \tag{37}$$

$$\frac{\phi_2}{2}(r) - \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_{PB}}{\partial n}(r, r') \, \mathrm{d}r' + \int_{\Gamma 2} \mathrm{G}_{PB}(r, r') \frac{\partial \phi_2}{\partial n}(r') \, \mathrm{d}r' = 0 \tag{38}$$

3.5.5 Sistema de ecuaciones con Stern-Layer

Al escribir las ecuaciones con la notación de operadores de forma matricial se obtiene lo mostrado en (39). Una análisis más extenso de esta formulación se encuentra en la publicación de Altman, Bardhan, White y Tidor [10].

$$\begin{bmatrix} \frac{I}{2} + K_L^1 & -V_1^1 & & \\ \frac{I}{2} - K_L^1 & \frac{\epsilon_1}{\epsilon_2} V_L^1 & K_L^2 & -V_L^2 \\ -K_L^1 & \frac{\epsilon_1}{\epsilon_2} V_L^1 & \frac{I}{2} + K_L^2 & -V_L^2 \\ & & \frac{I}{2} - K_{PB}^2 & V_{PB}^2 \end{bmatrix} \cdot \begin{Bmatrix} \phi_1 \\ \frac{\partial \phi}{\partial n_1} \\ \phi_2 \\ \frac{\partial \phi}{\partial n_2} \end{Bmatrix} = \begin{Bmatrix} \sum q_k G_L^k \\ 0 \\ 0 \\ 0 \end{Bmatrix} \tag{39}$$

4 Metodología

Para llevar a cabo el trabajo, es necesario obtener la información de las distintas moléculas analizadas, su composición, y la posición en el espacio de los átomos que la componen. Esto se obtiene a partir de un conjunto de fuentes y métodos reconocidos en el estudio de la dinámica molecular.

4.1 Protein Data Bank

PBD (por sus siglas en inglés) es un banco de datos de proteínas y moléculas orgánicas en general, fundada en 1971, para tener libre disposición y de manera estandarizada la información dela estructura, características y contextos de proteínas de todo tipo [2]. Actualmente se utilizan diversos métodos para obtener la estructura de una molécula, destacando para esta aplicación la cristalografía de rayos X y la resonancia magnética nuclear.

Todos estos datos son recopilados y almacenados en esta plataforma, sin embargo, el formato que utiliza otorga más información sobre estructuras internas (principalmente aminoácidos y sus interacciones) que de los átomos puntuales de la molécula en estudio. Por lo que es necesario recurrir a métodos externos a esta plataforma para extraer la información relevante en el formato requerido.

4.2 Campos de Fuerza

Para emplear el método en estudio es necesario conocer la carga y el radio atómico de cada uno de los átomos que componen la molécula. Para ello es necesario analizar cada átomo en su contexto particular y emplear criterios que permitan determinar su estado de ionización e influencia sobre el entorno.

Los diversos métodos que existen para determinar dichos parámetros en cada átomo se llaman campos de fuerza (*forcefields* en inglés), y son una mezcla entre modelos teóricos y resultados experimentales de diversos estudios, existe una gran diversidad de campos utilizados, dependiendo del tipo de molécula y contexto en que se encuentra. Para efectos de este trabajo se utiliza un campo llamado *amber* [11], el cual es una familia de criterios utilizados para determinar los parámetros necesarios en cada átomo.

Tanto el cálculo de cargas y radios atómicos es realizado por un programa llamado `pdb2pqr` [5], el cual como su nombre lo indica, recibe un archivo obtenido en PDB y genera un archivo con extensión `.pqr`, un formato de posiciones, cargas y radios de cada átomo, calculados según el campo de fuerzas deseado.

4.3 Superficie de exclusión y malladores

Como se definió anteriormente la superficie de exclusión es la distancia mínima a la cual se puede aproximar una molécula de agua, o iones de sal a la molécula en estudio. Para obtener dicha superficie se utilizan programas especializados tales como `msms` [6] y `NanoShaper`[7].

Ambos programas utilizan un archivo con las coordenadas y los radios de cada uno de los átomos (este tipo de archivo tiene extensión (`.pqr`)) para realizar la operación que se muestra en la Figura 1. Haciendo '*rodar*' en torno a la molécula una *sonda* de un radio determinado, obteniendo la distancia mínima de interacción, y mediante este método entregan una malla de triángulos que describe la superficie de exclusión de la molécula.

Tanto `nanoShaper` como `msms` permiten el control de diversos parámetros relevantes. Uno de estos parámetros es el radio de exclusión, que en este caso es el radio aproximado de la molécula de agua, otro parámetro importante es la densidad de malla, que sirve para poder refinar y obtener resultados más detallados de la distribución de carga sobre la superficie.

Además señalan si existen cavidades internas en la molécula, y así tratarlas de manera apropiada en caso de encontrarse- Una opción es crear una región interna con un campo de solvente al igual que el exterior, otra opción es considerar moléculas de agua internas como cargas puntuales de hidrógeno y oxígeno. Cabe señalar que en este trabajo no se estudian moléculas que presenten este tipo de cavidades.

4.4 Python

Python es un lenguaje de programación interpretado, cuya principal característica es su sencilla sintaxis y lectura, lo que permite desarrollar programas simples con una escritura de fácil comprensión para gente no especializada en el área de ciencias de la computación. Este lenguaje posee actualmente una de las comunidades más grandes y activas debido a su amplia gama de aplicaciones, además de disponer de un gran número de librerías para todo tipo de disciplinas.

En particular en este trabajo se utilizan las librerías `numpy` para arreglos como matrices y vectores, `scipy` que es la librería de donde se extrae el sistema de resolución (*solver* de ahora en adelante), en este caso GMRES, y `matplotlib` librería que se usa para realizar gráficos y de esa manera para visualizar información, principalmente resultados.

4.5 bempp

La elección del lenguaje de programación `python` se hace principalmente por la existencia de la libreía `bempp` (o BEM++) [4], dedicada a la implementación del método de elementos de borde. En ella se encuentran incorporados todos los procedimientos necesarios para definir los sistemas de ecuaciones, realizar los cálculos y visualizar resultados. Esto incluye el tratamiento de mallas de la superficie de exclusión, definición de espacios para definir el potencial y su derivada sobre la superficie, vectores y matrices para operar y resolver.

La principal característica que la hace apropiada para este trabajo, es la definición y evaluación de los operadores integrales vistos anteriormente (3.2).

Cuenta con una interfaz para `python`, sin embargo, los algoritmos y procesos más costosos computacionalmente están escritos en el lenguaje de programación C++ (de ahí el nombre), con ello se obtiene una librería sencilla de implementar y comprender, sin perder eficiencia que implica utilizar un lenguaje de programación completamente interpretado.

En la documentación oficial de esta librería se puede encontrar un demo del programa realizado para este trabajo. [1]

4.6 5PTI

La molécula utilizada para realizar las evaluaciones recibe el nombre de 5PTI. Esta molécula es una enzima pancreática bovina inhibidora de tripsina, presente en muchos vertebrados sirve para el control de los niveles de tripsina, otra enzima relacionada con la digestión de proteínas [19]. Su configuración configuración es obtenida mediante refracción de rayos X, el cual según la fuente tiene una resolución de 1 Å[20].

[1] Calculating the solvation energy of a protein using Laplace and modified Helmholtz, https://bempp.com/documentation/

5 Funcionamiento del programa e implementación en Python y bempp

El trabajo central de esta investigación es realizar un programa que permita obtener la energía de solvatación de cualquier molécula, a partir de la información disponible en el Protein Data Bank. La figura 4 muestra un diagrama con el resumen de las operaciones principales del programa elaborado para este trabajo.

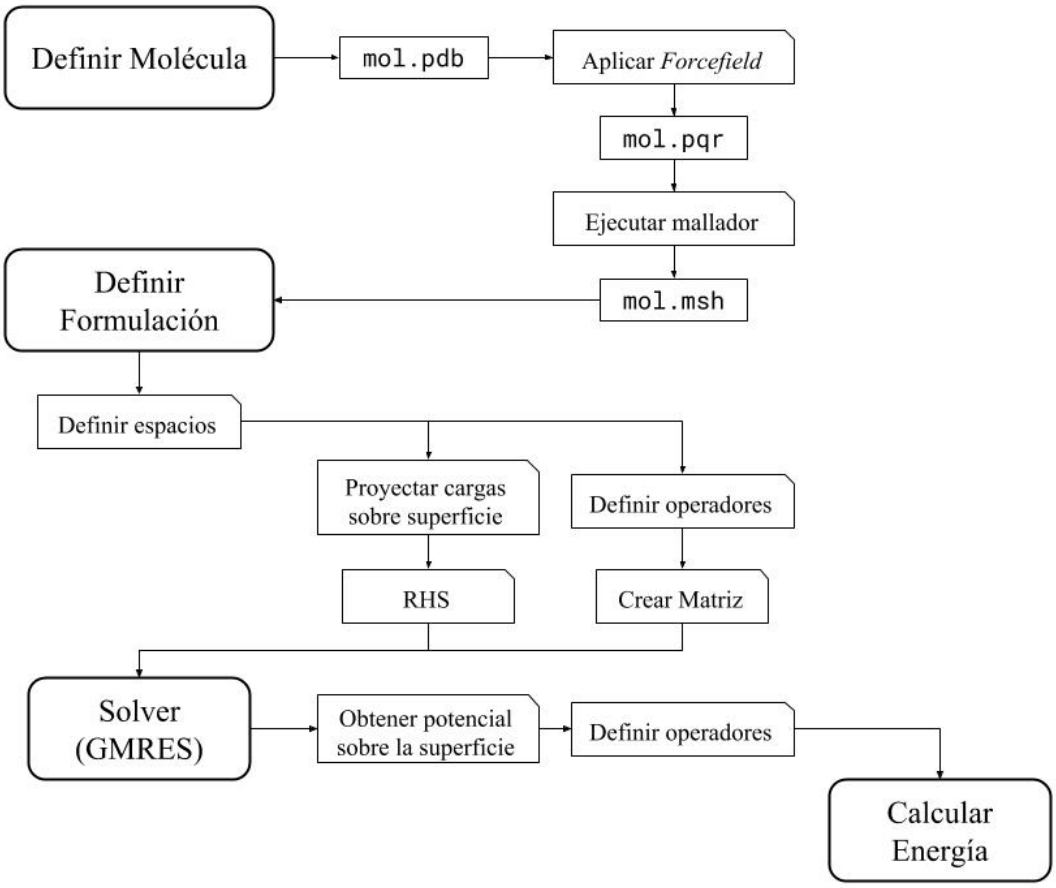


Figura 4: Esquema de funcionamiento del programa

Primero es necesario definir qué molécula se va a estudiar y buscar el archivo correspondiente en la base de datos del Protein Data Bank, luego se debe aplicar un *forcefield* que calcula los radios atómicos y cargas de cada uno de los átomos de la molécula, para finalmente obtener la geometría de la malla requerida para iniciar la simulación. A la malla obtenida se le hace un filtro para evitar la presencia de elementos demasiado pequeños, esto genera inestabilidad en el sistema (muchas veces provoca que el programa entregue errores de división por cero).

Todos estos procesos pueden ser automatizados mediante *scripts* simples escritos en `bash` (o directamente en la terminal), debido a que son programas externos no están hechos para ser llamados directamente desde `python`. Una vez terminado el proceso de elaboración de las mallas, el resto de los cálculos puede ser implementado directamente sin la necesidad de programas externos.

5.1 Importar malla y definir espacios

Para definir operadores sobre una superficie, es necesario definir la superficie, importando desde una archivo de malla (`mesh_file`), el cual contiene el arreglo de vértices y aristas que componen la superficie obtenida desde `msms` o `nanoShaper`. Para cada una de las superficies analizadas es necesario definir un operador para el potencial, y uno para su derivada normal, estos son llamados espacios de Dirichlet y Neumann respectivamente.

Todo el trabajo se realiza utilizando elementos de borde con valor constante (orden 0) y por lo tanto discontinuos (Discontinuous Polynomial).

```
import bempp.api

grid = bempp.api.import_grid(mesh_file)

dirichl_space = bempp.api.function_space(grid, 'DP', 0)
neumann_space = bempp.api.function_space(grid, 'DP', 0)

# grid.plot() # para imprimir visualizar la malla
```

En la Figura 5 se muestra la visualización de distintas mallas al ser importadas en `bempp`

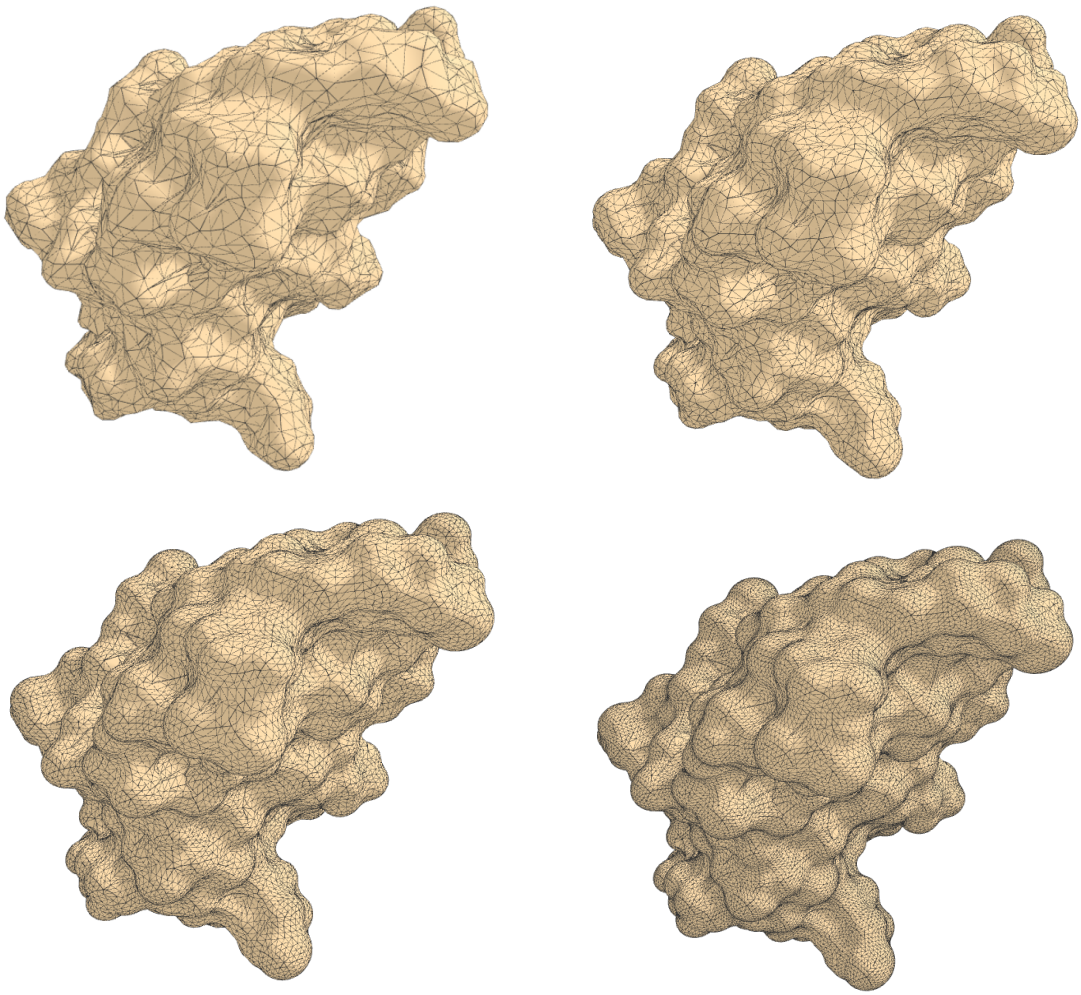


Figura 5: Visualización de refinación de malla para la molécula 5PTI, mallas de 12508, 24174, 47404 y 97204 elementos

5.2 Proyectar cargas sobre la superficie

Para definir el lado derecho del sistema obtenido en la ecuación (26), es necesario definir la función que se aplica sobre la superficie de exclusión. Esta función es la que contiene la sumatoria de cargas proyectadas hasta cada elemento de la superficie, y queda escrita de la siguiente manera.

```
# Functions to proyect the carges potential to the boundary
def green_func(x, n, domain_index, result):
   result[:] = np.sum(q/np.linalg.norm( x - x_q, axis=1 ))
       /(4.*np.pi*ep_in)
```

Luego de esto, se aplica dicha función sobre el espacio requerido. Al finalizar

se concatenan los coeficientes (valores numéricos reales) de dicha operación con una columna de ceros igual al número de elementos de la derivada del potencial, respetando de esta manera el numero de variables del sistema.

```
charged_grid_fun = bempp.api.GridFunction(dirichl_space, fun=green_func)
rhs = np.concatenate([charged_grid_fun.coefficients,
    np.zeros(neumann_space.global_dof_count)])
```

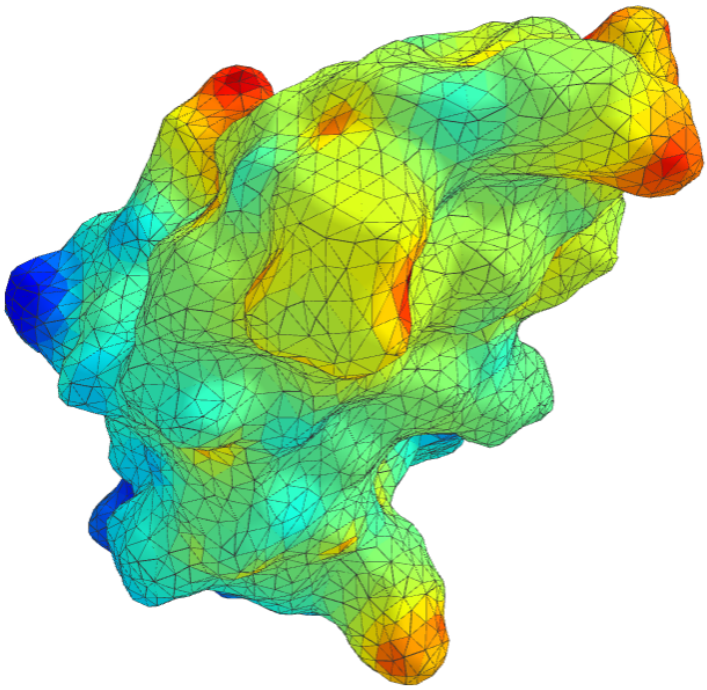


Figura 6: Visualización del potencial electrostático de las cargas internas proyectadas sobre la superficie de exclusión

5.3 Definir matriz del sistema

Para obtener la matriz del sistema, es necesario definir en primer lugar los operadores sobre los espacios correspondientes. Para el caso de una única superficie sólo se requieren 4 operadores como indica la ecuación (26), para luego definir la matriz por bloques.

La librería permite hacer operaciones aritméticas simples sobre los operadores, y de esta forma definir cada uno de los bloques de la matriz de manera sencilla (notar que los términos `ep_in` y `ep_ex` son las constantes dieléctricas dentro y fuera de la superficie respectivamente).

```
identity = sparse.identity(dirichl_space, dirichl_space, dirichl_space)
slp_in = laplace.single_layer(neumann_space, dirichl_space,
    dirichl_space)
dlp_in = laplace.double_layer(dirichl_space, dirichl_space,
    dirichl_space)
slp_ex = modified_helmholtz.single_layer(neumann_space, dirichl_space,
    dirichl_space, kappa)
dlp_ex = modified_helmholtz.double_layer(dirichl_space, dirichl_space,
    dirichl_space, kappa)

blocked = bempp.api.BlockedOperator(2, 2)
blocked[0, 0] = 0.5*identity + dlp_in
blocked[0, 1] = -slp_in
blocked[1, 0] = 0.5*identity - dlp_ex
blocked[1, 1] = (ep_in/ep_ex)*slp_ex
A = blocked.strong_form()
```

Al definir la matriz comienza el proceso de ensamblado, donde se definen internamente los operadores que luego son ingresados al *solver* para obtener la solución final del sistema.

5.4 Resolución del sistema

En este caso se utiliza el *solver* GMRES [13] (Generalized Minimal RESidual method) de la librería `scipy`. Es necesario mencionar que el *solver* trabaja realizando iteraciones entre operadores definidos por la librería hasta llegar a la solución, es decir, en ningún momento obtiene la matriz numérica del sistema.

Se usa una tolerancia de 10^{-5}, un número máximo de iteraciones de 500.

```
from scipy.sparse.linalg import gmres
x, info = gmres(A, rhs, tol=1e-5, maxiter=500)
```

En la Figura 7 se muestra el potencial electrostático ϕ al solucionar el sistema.

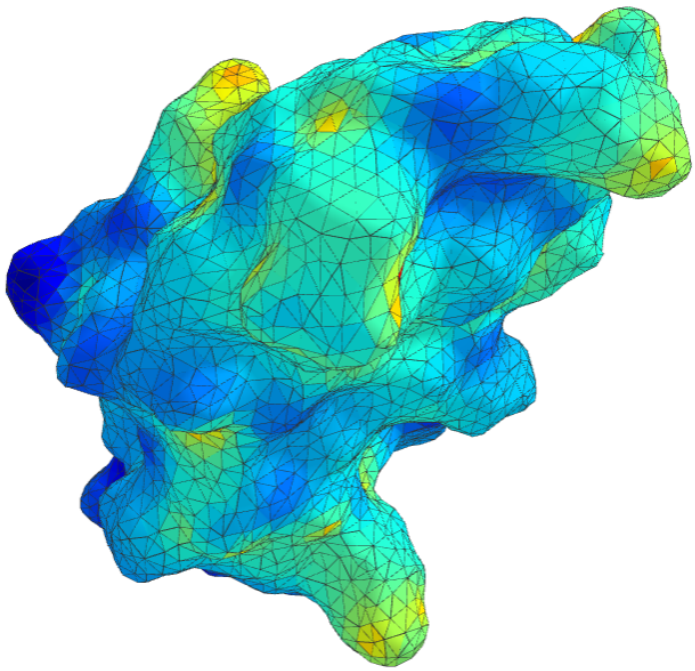


Figura 7: Visualización del potencial sobre la superficie al encontrar la solución

5.5 Cálculo de la Energía de Solvatación

La solución obtenida en el *solver*, es un vector que incluye los valores del potencial y su derivada normal de manera concatenada. Para obtener la energía de solvatación, es necesario separarlos y aplicarlos sobre los operadores como lo indica la ecuación (15).

```
solution_dirichl = bempp.api.GridFunction(dirichl_space,
      coefficients=x[:dirichl_space.global_dof_count])
solution_neumann = bempp.api.GridFunction(neumann_space,
      coefficients=x[dirichl_space.global_dof_count:])

from bempp.api.operators.potential.laplace \
 import single_layer, double_layer

slp_q = single_layer(neumann_space, x_q.transpose())
dlp_q = double_layer(dirichl_space, x_q.transpose())
```

Los operadores definidos sobre la región interior de la superficie, se utilizan para calcular el potencial en los puntos donde se encuentran las cargas. Una vez obtenidos estos valores, se puede calcular la energía de solvatación utilizando la ecuación (9).

```
phi_q = slp_q*solution_neumann - dlp_q*solution_dirichl
const = 2*np.pi*332.064
total_energy = const*np.sum(q*phi_q).real
```

La constante incluida es un factor para transformar unidades a [kcal/mol]. El programa se puede encontrar en su repositorio para su libre uso y modificación.[2] El programa no tiene incorporada la librería `bempp`, ni los programas para mallar la superficie. Estos deben ser adquiridos desde sus fuentes originales.

5.6 Implementación de formulación de Juffer

Para el caso de la formulación de Juffer, ya que solo requiere de una superficie se utiliza la misma malla, sin embargo, cambia la definición de la matriz, ya que al derivar una o ambas ecuaciones los operadores cambian. Las ecuaciones (27) muestra cómo se obtienen dichos operadores, y la matriz obtenida en (28) se define en `bempp` de la siguiente manera.

```
from bempp.api.operators.boundary import sparse, laplace,
    modified_helmholtz

phi_id = sparse.identity(dirichl_space, dirichl_space, dirichl_space)
dph_id = sparse.identity(neumann_space, neumann_space, neumann_space)
ep = ep_ex/ep_in

dF = laplace.double_layer(dirichl_space, dirichl_space, dirichl_space)
dP = modified_helmholtz.double_layer(dirichl_space, dirichl_space,
    dirichl_space, kappa)
L1 = ep*dP - dF

F = laplace.single_layer(neumann_space, dirichl_space, dirichl_space)
P = modified_helmholtz.single_layer(neumann_space, dirichl_space,
    dirichl_space, kappa)
L2 = F - P

ddF = laplace.hypersingular(dirichl_space, neumann_space, neumann_space)
ddP = modified_helmholtz.hypersingular(dirichl_space, neumann_space,
    neumann_space, kappa)
L3 = ddP - ddF # Cambio de signo por definicion de bempp
```

[2] https://gitlab.com/PauloFernandez/molecular-formulations

```
dF0 = laplace.adjoint_double_layer(neumann_space, neumann_space,
    neumann_space)
dP0 = modified_helmholtz.adjoint_double_layer(neumann_space,
    neumann_space, neumann_space, kappa)
L4 = dF0 - (1./ep)*dP0

blocked = bempp.api.BlockedOperator(2, 2)
blocked[0, 0] = 0.5*(1. + ep)*phi_id - L1
blocked[0, 1] = -L2
blocked[1, 0] = -L3
blocked[1, 1] = 0.5*(1. + 1./ep)*dph_id - L4
A = blocked.strong_form()
```

Al igual que en la formulación directa, se define el lado derecho como la función de los espacios definidos sobre la malla, pero en este caso ambas ecuaciones poseen funciones distintas.

```
def d_green_func(x, n, domain_index, result):
        const = -1./(4.*np.pi*ep_in)
        result[:] = const*np.sum(q*np.dot( x - x_q, n
            )/(np.linalg.norm( x - x_q, axis=1 )**3))

def green_func(x, n, domain_index, result):
        result[:] = np.sum(q/np.linalg.norm( x - x_q, axis=1
            ))/(4.*np.pi*ep_in)

rhs_1 = bempp.api.GridFunction(dirichl_space, fun=green_func)
rhs_2 = bempp.api.GridFunction(dirichl_space, fun=d_green_func)
rhs = np.concatenate([rhs_1.coefficients, rhs_2.coefficients])
```

El resto del procedimiento sigue siendo el mismo que el explicado originalmente, tanto el *solver* como la obtención de la energía final son exactamente iguales.

5.7 Implementación de Stern Layer

A diferencia de las otras dos formulaciones, esta requiere de dos superficies, por lo que es necesario importar mallas, definir espacios y definir operadores:

```
grid_in = bempp.api.import_grid(mesh_file_in)
grid_ex = bempp.api.import_grid(mesh_file_ex)

dirichl_space_in = bempp.api.function_space(grid_in, "DP", 0)
neumann_space_in = bempp.api.function_space(grid_in, "DP", 0)
dirichl_space_ex = bempp.api.function_space(grid_ex, "DP", 0)
neumann_space_ex = bempp.api.function_space(grid_ex, "DP", 0)
```

Para implementar el sistema de ecuaciones (39) se definen los operadores sobre cada superficie.

```
from bempp.api.operators.boundary import sparse, laplace,
    modified_helmholtz
# Operator for internal surface identity
idn_in = sparse.identity(dirichl_space_in, dirichl_space_in,
    dirichl_space_in)

# Internal Boundary
slp_in = laplace.single_layer(neumann_space_in, dirichl_space_in,
    dirichl_space_in)
dlp_in = laplace.double_layer(dirichl_space_in, dirichl_space_in,
    dirichl_space_in)

# External Boundary
# adj_1T1 = laplace.single_layer(neumann_space_in, dirichl_space_in,
    dirichl_space_in)
# adj_1T1 = laplace.double_layer(dirichl_space_in, dirichl_space_in,
    dirichl_space_in)

slp_2T1 = laplace.single_layer(neumann_space_ex, dirichl_space_in,
    dirichl_space_in)
dlp_2T1 = laplace.double_layer(dirichl_space_ex, dirichl_space_in,
    dirichl_space_in)

# Operator for external surface identity
```

```
idn_ex = sparse.identity(dirichl_space_ex, dirichl_space_ex,
    dirichl_space_ex)

# Internal Boudary
slp_1T2 = laplace.single_layer(neumann_space_in, dirichl_space_ex,
    dirichl_space_ex)
# dlp_1T2 = laplace.double_layer(dirichl_space_in, dirichl_space_ex,
    dirichl_space_ex)

slp_2T2 = laplace.single_layer(neumann_space_ex, dirichl_space_ex,
    dirichl_space_ex)
dlp_2T2 = laplace.double_layer(dirichl_space_ex, dirichl_space_ex,
    dirichl_space_ex)

# External Boundary
slp_ex = modified_helmholtz.single_layer(neumann_space_ex,
    dirichl_space_ex, dirichl_space_ex, kappa)
dlp_ex = modified_helmholtz.double_layer(dirichl_space_ex,
    dirichl_space_ex, dirichl_space_ex, kappa)
```

Finalmente se definen el lado derecho (`rhs`) y la matriz del sistema `A`. Al igual que la formulación directa y Juffer, tanto el *solver* como la obtención de la energía se calcula de la misma forma, es decir, con el potencial electrostático de la superficie interior.

```
rhs = np.concatenate([charged_grid_fun.coefficients,
                      np.zeros(neumann_space_in.global_dof_count),
                      np.zeros(dirichl_space_ex.global_dof_count),
                      np.zeros(neumann_space_ex.global_dof_count)])

ep = (ep_in/ep_ex)

# Matrix Assemble
blocked = bempp.api.BlockedOperator(4, 4)
blocked[0, 0] = .5*idn_in + dlp_in
blocked[0, 1] = -slp_in
# blocked[0, 2] = 0
# blocked[0, 3] = 0
```

```
# Original formulation
blocked[1, 0] = .5*idn_in - dlp_in # dlp_in = dlp_1T1
blocked[1, 1] = ep*slp_in        # slp_in = slp_1T1
blocked[1, 2] = dlp_2T1
blocked[1, 3] = -slp_2T1

blocked[2, 0] = -dlp_1T2 # - for ASC
blocked[2, 1] = ep*slp_1T2
blocked[2, 2] = .5*idn_ex + dlp_2T2
blocked[2, 3] = -slp_2T2

# blocked[3, 0] = 0
# blocked[3, 1] = 0
blocked[3, 2] = .5*idn_ex - dlp_ex
blocked[3, 3] = slp_ex

A = blocked.strong_form()
```

6 Programas disponibles

Para tener una referencia de los resultados obtenidos en las diferentes formulaciones, se realiza una comparación con dos programas disponibles: `TABI` y `PyGBe`.

6.1 PyGBe

`PyGBe` (Python, GPUs and Boundary elements for biomolecular electrostatics) [14] es un programa desarrollado en python que incorpora el método de elemetos de borde. Cuenta con un algoritmo de treecode para realizar la integración sobre cada elemento de la superficie de manera más eficiente.

6.2 TABI

`TABI` (Treecode-Accelerated Boundary Integral) [15] es un programa que, como su nombre lo indica, incorpora el método de elementos de borde en conjunto con un algoritmo de treecode al igual que `PyBGe`. La diferencia es que utiliza la formulación de Juffer en lugar de la formulación directa.

6.3 Otros

Otros programas que caben mencionar pero no fueron utilizados para comparar, son `APBS` (Adaptive Poisson-Boltzmann Solver) [18] y `MIBPB` (Matched Interface and Boundary Based Poisson-Boltzmann Solver) [16]. Ambos son proyectos de código abierto y permiten el estudio de diversos parámetros asociados al estudio de la simulación de moléculas en medio continuo.

7 Resultados

7.1 Convergencia de malla

El propósito de refinar la malla es acercarse lo más posible a una superficie suave, y con ello analizar la tendencia del resultado final, que en este caso es la energía de solvatación obtenida. Al hacer este estudio de convergencia se obtiene la tendencia del valor de la energía de solvatación, y para determinar dicha tendencia se utiliza como herramienta la extrapolación de Richarson.

La extrapolación de Richardson como se muestra en las ecuaciones (40) requiere información de los resultados y de las mallas con que se obtienen dichos resultados. La idea es obtener el valor de f_{ex} que es el resultado al que tiende el sistema si presenta convergencia. Para ello se tiene que f_i son los valores obtenidos en cada caso, h_i el número de elementos de la malla utilizados en cada caso, lo cual sirve para calcular r que nos indica cuánto se refina la malla en cada nueva medición. Un parámetro importante es el p, que es el orden de convergencia del sistema y nos indica qué tan rápido converge a la solución teórica f_{ex}

$$\begin{aligned} f_{ex} &\approx f_1 + \frac{f_1 - f_2}{r^p - 1} \\ p &= \frac{\log \frac{f_3 - f_2}{f_2 - f_1}}{\log r} \\ r &= \frac{h_3}{h_2} \end{aligned} \tag{40}$$

Para el primer análisis se utilizan se comparan la formulación directa, Juffer, Stern, y el programa `PyGBe`. Se utilizan mallas creadas en `msms` con densidades de 2, 4, 8, 16, y 32 vértices por $Å^2$, esto para obtener un ratio constante para el número de elementos de la malla (parámetro r de la extrapolación de Richardson).

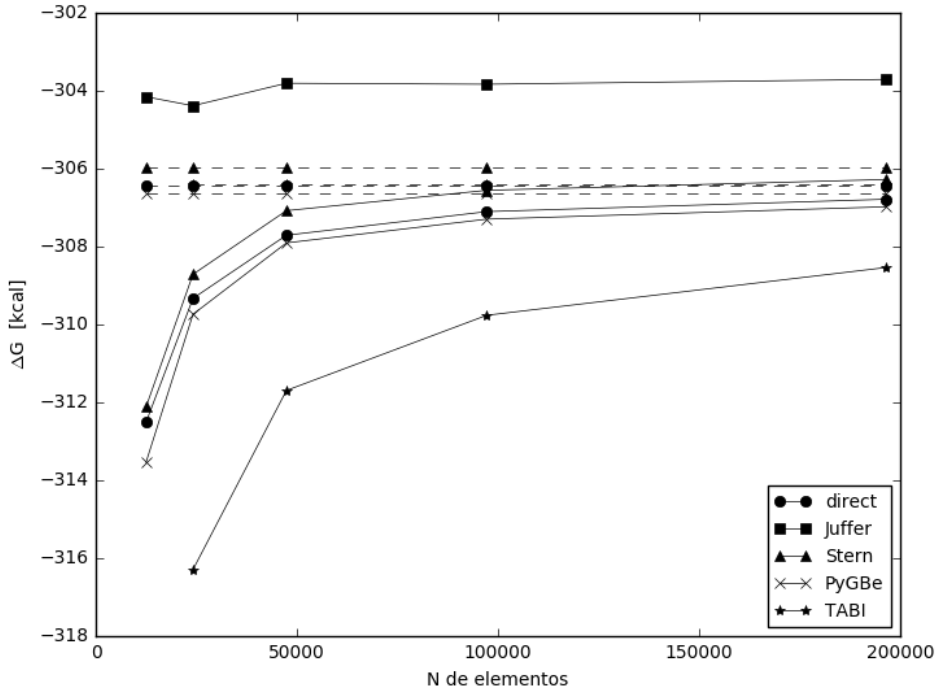


Figura 8: Convergencia de la solución al refinar la malla

Como se observa en la Figura 8, tanto la formulación directa como la formulación con Stern Layer se comportan de manera similar que `PyGBe`. Sin embargo la formulación de Juffer presenta valores levemente más altos.

A continuación se presentan los valores de energía a los que tienden cada uno de las formulaciones y programas comparados, junto con su orden de convergencia.

Tabla 1: Valores de energía de solvatación y orden de convergencia

Formulación	**Energía [kcal/mol]**	**p**
Directa	-306.43	0.9320
Juffer	–	–
Stern	-305.96	0.9104
PyGBe	-306.63	0.9427
TABI	-307.26	0.9676

Hay que destacar que con la formulación de Juffer se obtienen valores que oscilan, y no presentan una tendencia clara de convergencia hacia un retultado. Esto impide realizar la extrapolación de Richardson e identificar cual es el valor al que tiende el sistema al refinar la malla.

7.2 Influencia del radio *Stern* en la energía de solvatación

El modelo estudiado en el caso de una segunda superficie de exclusión, la cual es una aproximación para evitar las concentraciones excesivas de carga sobre la superficie de la molécula provocada por el modelo de solvente continuo. Sin embargo el radio Stern, que es la distancia a la que se encuentran ambas superficies de exclusión, tienen influencia en la energía final calculada. Tanto `PyGBe` como `bempp` consideran la doble superficie y resuelven el sistema (39).

La Figura 9 ilustra la variación de la energía, considerando radios de exclusión para los iones entre 0.3 y 4 Å.

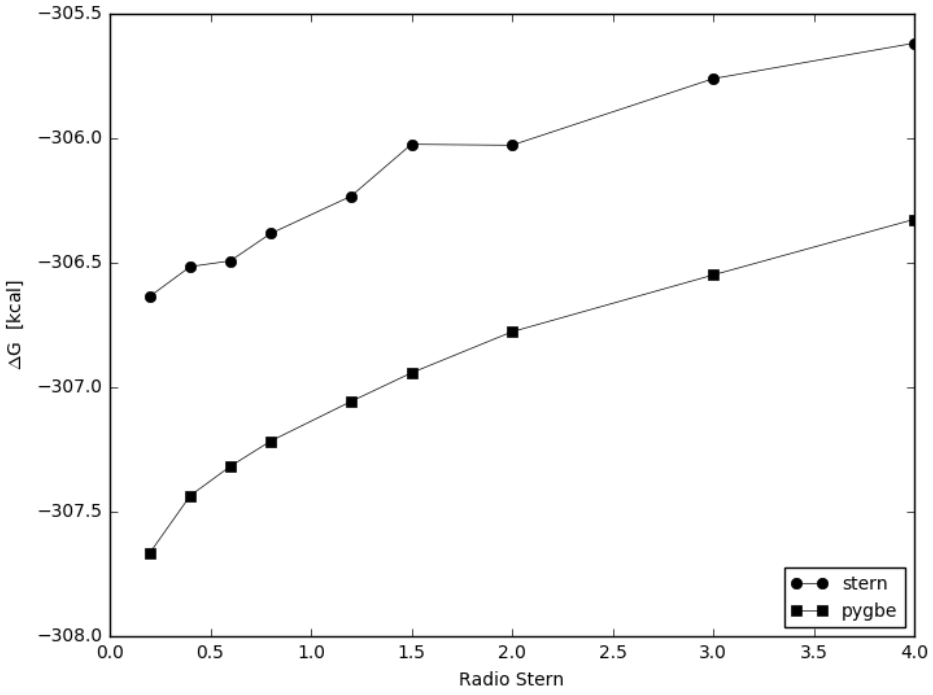


Figura 9: Energía en función del radio *stern* en Å

7.3 Tiempo de cómputo

Ya que la complejidad algorítmica es difícil de obtener de manera exacta para este tipo de problemas, un indicador valioso de los recursos utilizados para realizar los cálculos es el tiempo de cómputo. Debido a la naturaleza del método utilizado por la librería, el tiempo que tarda en realizarse el proceso completo se encuentra compuesto principalmente por dos partes, el tiempo de definición y ensamblaje de los operadores y la matriz del sistema, y el tiempo de resolución del sistema.

7.3.1 Tiempo Ensamblaje

Ya que la librería utiliza una matriz de operadores para calcular el resultado, es necesario construirla antes de comenzar la resolución del sistema. Es durante este proceso que el programa crea todos los operadores utilizados sobre las distintas superficies involucradas, separando por bloques los operadores definidos sobre distintas incógnitas.

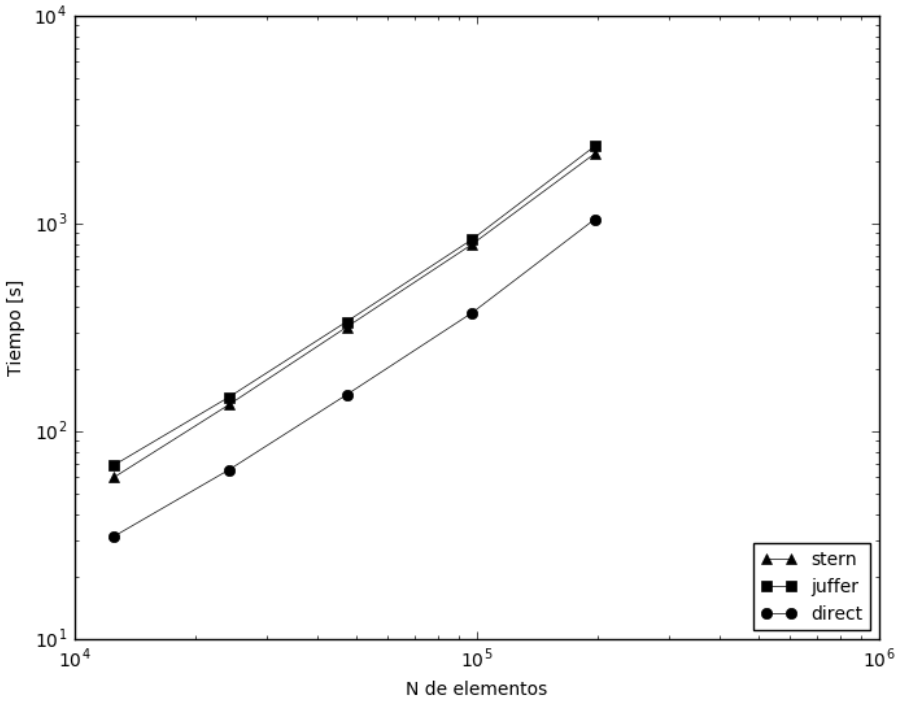


Figura 10: Tiempo de ensamblaje de la matriz de operadores

Cabe destacar que `PyGBe` y `TABI` no ensamblan operadores debido a que no utilizan matrices jerárquicas. En su lugar utilizan el algoritmo de treecode, el cual es menos costoso al momento de definir el sistema, pero cada iteración es relativamente

más costosa en términos de uso de memoria y tiempo de cómputo. Debido a esto se opta por no comparar en este item.

7.3.2 Tiempo de Resolución

Para obtener el resultado del sistema con cada uno de las formulaciones se utiliza el método iterativo GMRES. Debido al condicionamiento que poseen las distintas formulaciones es posible observar diferencias en los tiempos de cálculo, sin cambiar considerablemente el orden de complejidad.

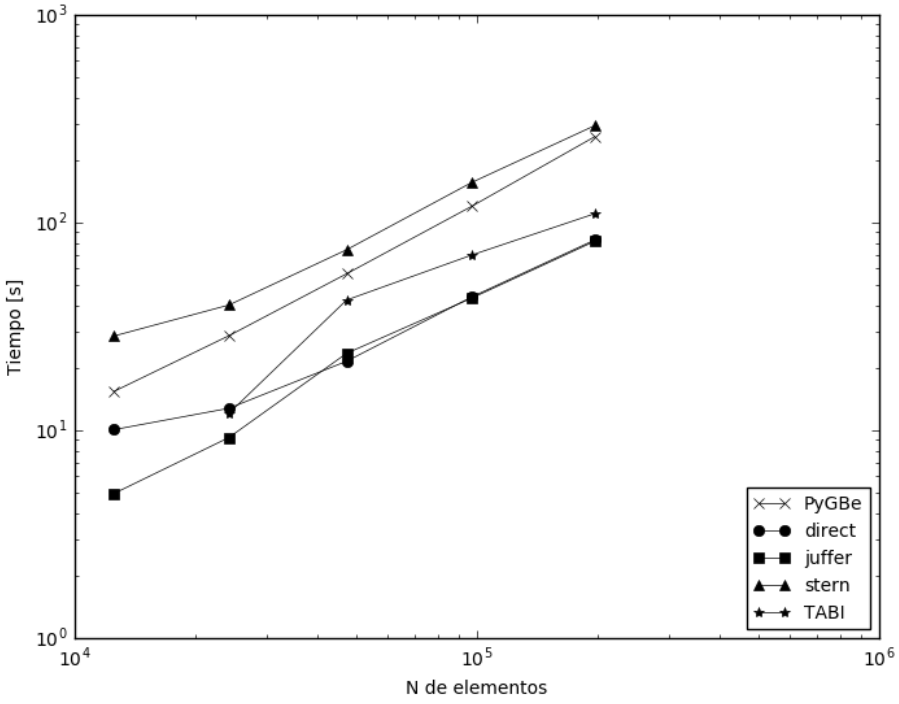


Figura 11: Tiempo de convergencia del sistema

El tiempo que se tarda en resolver el sistema, no necesariamente dice relación con el condicionamiento de la matriz, ni con la estabilidad del sistema. El tiempo que tarda cada iteración cambia entre un método y otro, dependiendo principalmente del número de operadores incorporados en el sistema.

El número de iteraciones se vuelve entonces un indicador importante para comparar entre las distintas formulaciones y `PyBGe`. Para este caso se utilizó una tolerancia de error de 10^{-5} en el solver.

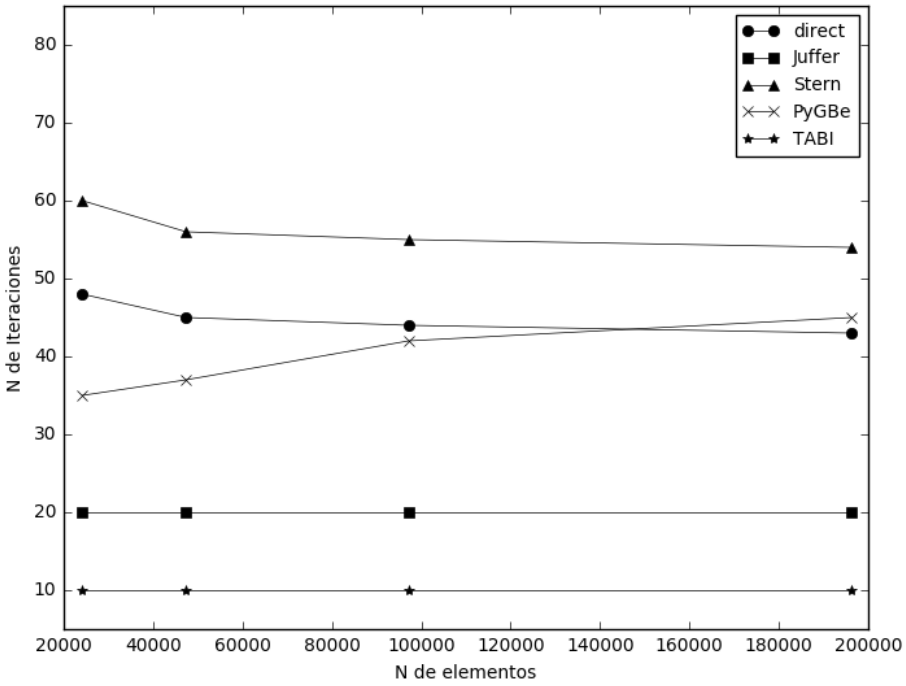


Figura 12: Número de iteraciones hasta convergencia

Se puede notar una tendencia a disminuir la cantidad de iteraciones necesarias para converger a la solución al refinar la malla en el caso de las formulaciones implementadas en `bempp`, mientras que en `PyGBe` aumentan.

7.3.3 Tiempo total de cómputo

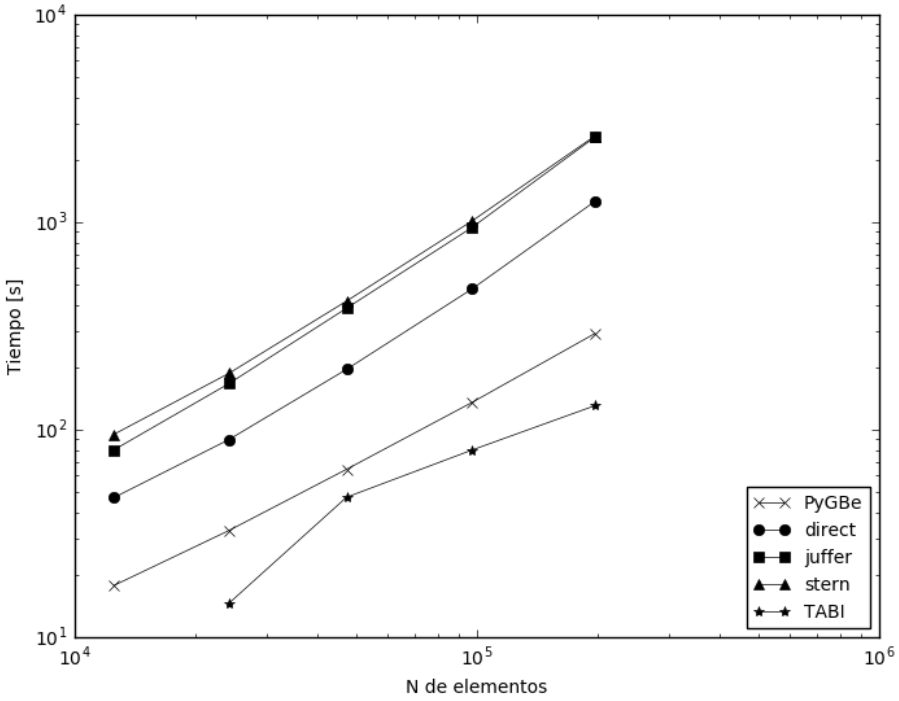


Figura 13: Tiempo total de las distintas formulaciones

Es evidente que en cuánto a la comparación entre los tiempos totales obtenidos en las distintas formulaciones, el tiempo de ensamblaje es la mayor limitante al momento de escalar el método, ya que aumenta más rápido que el tiempo de resolución. Esto se puede observar en la clara ventaja de `PyGBe` sobre el resto de las formulaciones en la Figura 13.

8 Analisis de resultados

8.1 Energía de solvatación y convergencia de malla

Los resultados obtenidos en todos los métodos utilizados se encuentran en torno a las 307 kcal, y todos tienen un orden de convergencia cercanos a 1, lo que es esperable ya que en todos los métodos comparados se utilizan elementos de borde lineales.

El único método que presenta una convergencia de malla distinta es la formulación de Juffer, que presenta una oscilación en torno al resultado

8.2 Tiempos de cómputo

Los resultados obtenidos muestran una clara desventaja por parte de la implementación de la librería en comparación con `PyGBe` y `TABI`, esto principalmente debido a que estos dos programas son especializados en este tipo de cálculos y por lo tanto es de esperar que se encuentren optimizados, mientras que la librería `bempp` es una herramienta de uso general que permite implementar de manera sencilla formulaciones alternativas y probar distintas operaciones sobre las ecuaciones, asumiendo el costo de tener un producto final menos optimizado.

El método de matrices jerárquicas utilizado por `bempp` tiene una desventaja clara al momento de ensamblar el sistema de ecuaciones, el tiempo que tarda en crear los operadores es la principal limitante cuando se pretende escalar este tipo de problemas, ya que tiene un crecimiento acelerado al aumentar el número de elementos de la malla. Esto se evidencia al observar que se vuelve más costoso implementar formulaciones con un número mayor de operadores, como es el caso de la formulación de Juffer, donde si bien es una formulación con un condicionamiento mejor que el resto y por lo tanto converge más rápido a la solución, se vuelve altamente costosa la definición de operadores en cuanto a tiempo y memoria requerida. El caso de implementar la capa Stern se observa un costo aun mayor, debido al gran número de operadores que es necesario generar.

Obviando el problema de definición y ensamblaje de operadores, el tiempo de resolución es bastante similar para los programas como `PyGBe` y `TABI` en comparación con los de `bempp`, no existen mayores diferencias, debido a que los sistemas a resolver son similares.

8.3 Juffer

La formulación de Juffer tiene dos características importantes. La primera tiene relación con la convergencia del sistema, ya que es un sistema bien condicionado, el sistema siempre converge en un número fijo de iteraciones, independientemente de la malla utilizada.

Por otro lado el número de operadores requeridos es mayor que la formulación original, lo que provoca inevitablemente un costo mayor en memoria y en tiempo de ensamblaje de la matriz. Esto, como se ha mencionado, presenta la mayor limitante al momento de comparar con otros métodos y formulaciones. A modo de conclusión se estima que una buena implementación de los operadores necesarios para implementar esta formulación, podría otorgar ventajas en aquellos casos donde el resto de los métodos no presenten una buena convergencia del sistema.

Además se puede observar una oscilación en los resultados obtenidos al refinar la malla. si bien los valores oscilan en torno a valores similares, esto no permite la proyección de la solución al aplicar la extrapolación de Richardson, haciendo más difícil la comparación con el resto de las formulaciones.

8.4 Capa Stern

La presencia de una capa Stern no presenta mayores diferencias en el valor de la energía de solvatación, y los análisis de convergencia de malla no muestran una ventaja por parte de este método. Los cálculos no se pueden realizar con tanta rapidez como con la formulación directa de una superficie, debido al alto consumo de memoria que demanda el sistema de ecuaciones.

Además se observa un cambio en el valor de la energía de solvatación al variar la distancia entre la capa Stern y la superficie de exclusión original, sin un límite aparente al aumentar la distancia, y sin converger al valor original al disminuir la distancia entre las mallas. Esto se debe principalmente al error inducido al tener una distancia entre mallas menor al tamaño de los elementos de la malla.

8.5 Implementación sobre bempp

Para la implementación de este tipo de modelos, `bempp` es una alternativa sencilla y rápida que permite la manipulación de operadores sin necesidad de definir cada operación manualmente. Sin embargo, no es la alternativa óptima para aplicaciones complejas que requieran ser escaladas.

En caso que un modelo que requiera un gran número de operadores, superficies,

o un gran número de elementos de malla, los tiempos y la memoria necesaria para ensamblar esa cantidad de operadores se vuelven altamente costosos, superando con creces los requerimientos necesarios para resolver el sistema de ecuaciones. Finalmente, los programas especializados no pierden la ventaja si el interés es solo obtener el resultado final.

9 Conclusiones

Los resultados de la energía de solvatación obtenida mediante las distintas formulaciones muestran una clara convergencia del modelo estudiado. Si bien algunas formulaciones presentan ciertas ventajas sobre otras, los valores obtenidos no varían de manera considerable. La principal causa de variación de resultados de la energía de solvatación en todos los casos está dada por la resolución de la malla utilizada. Mientras que la variación entre formulaciones determinan principalmente el orden de convergencia al refinar la malla, y el número de iteraciones necesarias para obtener el resultado del sistema de ecuaciones.

Entre las formulaciones estudiadas es posible ver una ventaja clara en cuanto a tiempos y uso de memoria de la formulación directa. Mientras que la formulación de Juffer converge en menos iteraciones, siendo la definición del sistema de ecuaciones su principal limitante debido a su costo de memoria y tiempo por la cantidad de operadores utilizados.

El alto costo en tiempo y memoria demuestra la principal desventaja de herramientas de propósito múltiple como `bempp`, que permiten realizar una gran cantidad de pruebas y manipulación rápida de las ecuaciones. Sin embargo, no permiten optimizar a voluntad todos los procesos. Aplicar precondicionadores al sistema de ecuaciones o manipular el método de integración utilizado debe ser implementado para toda la librería antes de poder ser utilizado en un programa específico, lo cual vuelve restrictiva la personalización de las operaciones.

Los programas especializados, como PyGBe y TABI, no requieren crear los operadores, y su método de *treecode* para acelerar el proceso de integración es más eficiente en cuanto a memoria y tiempo de ensamblaje. Por lo que si el objetivo es netamente calcular la energía de solvatación, son la mejor alternativa debido tanto a su facilidad de uso, como rapidez de cálculo.

References

[1] Christopher Cooper Villagrán, PhD
Biomolecular electrostatics with continuum models: a boundary integral implementation and applications to biosensors,
Boston University, 2015

[2] RCSB Protein Data Bank,
`https://www.rcsb.org/`

[3] Python programming language,
`https://www.python.org/`

[4] Bempp Services,
`https://bempp.com/`

[5] APBS and PDB2PQR: electrostatic and solvation properties for complex molecules,
`http://www.poissonboltzmann.org/`

[6] Molecular surfaces computation, Michel Sanner, 1996
`http://mgl.scripps.edu/people/sanner/html/msms_home.html`

[7] S. Decherchi, W. Rocchia
A general and Robust Ray-Casting-Based Algorithm for Triangulating Surfaces at the Nanoscale.

[8] Byung Jun Yoon and A.M. Lenhoff
A Boundary Element Method for Molecular Electrostatics with Electrolyte Effects
Department of Chemical Engineering, University of Delaware
Received 9 February 1990; accepted 7 May 1990

[9] Andre H. Juffer, Eugen F. F. Botta, Bert A. M. Van Keul
The Electric Potential of a Macromolecule in a Solvent:
A Fundamental Approach
Laboratory of Physical Chemistry and Department of Mathematics,
University of Groningen, Groningen, The Netherlands
October 25, 1990

[10] Michael D. Altman, Jaydeep P. Bardhan, Jacob K. White, Bruce Tidor
Accurate Solution of Multi-Region Continuum Biomolecule Electrostatic Problems Using the Linearized Poisson–Boltzmann Equation with Curved Boundary Elements
Massachusetts Institute of Technology
Received 19 May 2007; Revised 27 November 2007; Accepted 9 April 2008

[11] R. Salomon-Ferrer, D.A. Case, R.C. Walker
An overview of the Amber biomolecular simulation package.
WIREs Comput. Mol. Sci. 3, 198-210 (2013)

[12] BREBBIA, C.A.,
The Boundary Element Method for Engineers,
Pentech Press, London, 1978

[13] Youcef Saad & Martin H. Schultz,
GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems,
Society for Industrial and Applied Mathematics, 1986

[14] C. Cooper, J. Bardhan, L. Barba
A biomolecular electrostatics solver using Python, GPUs and boundary elements that can handle solvent-filled cavities and Stern layers.
Boston University 2014

[15] Weihua Geng, Robert Krasny,
A treecode-accelerated boundary integral Poisson–Boltzmann solver for electrostatics of solvated biomolecules, 2013

[16] Duan Chen, Zhan Chen, Changjun Chen, Weihua Geng and Guo-Wei Wei,
MIBPB: A software package for electrostatic analysis.
Journal of Computational Chemistry, 2011

[17] Baker NA, Sept D, Joseph S, Holst MJ, McCammon JA.
Electrostatics of nanosystems: application to microtubules and the ribosome.
Proceedings of the National Academy of Sciences, USA, 1998
http://www.pnas.org/content/98/18/10037

[18] Jurrus E, Engel D, Star K, Monson K, Brandi J, Felberg LE, Brookes DH, Wilson L, Chen J, Liles K, Chun M, Li P, Gohara DW, Dolinsky T, Konecny

R, Koes DR, Nielsen JE, Head-Gordon T, Geng W, Krasny R, Wei GW, Holst MJ, McCammon JA, Baker NA.
Improvements to the APBS biomolecular solvation software suite.
Protein Science, 27, 112-128, 2018
http://dx.doi.org/10.1002/pro.3280

[19] Wlodawer, A.,Deisenhofer, J.,Huber, R.
Comparison of Two Highly Refined Structures of Bovine Pancreatic Trypsin Inhibitor
Journal of Molecular Biology, 1987, 193: 145

[20] Walter, J.,Huber, R.
Pancreatic Trypsin Inhibitor. A New Crystal Form and its Analysis
Journal of Molecular Biology, 1983, 167: 911

Anexo 1

El siguiente anexo contiene un intento por reducir el sistema de ecuaciones con el fin de mejorar el desempeño del programa. Dentro del estudio se detecta que dentro de la formulación de Stern, existen operadores que aportan menos numéricamente al potencial, y por lo tanto a la energía de solvatación.

No existen resultados cercanos a las referencias, por lo que no se presenta dentro del trabajo principal. De todas formas se adjunta para dejar registro del tratamiento de los operadores.

Reducción del sistema

Debido al número de operadores que es necesario definir al agregar una capa extra en la formulación de Stern, en un proceso similar al seguido por la formulación de Juffer, se busca reducir el sistema de ecuaciones. Al reducir el número de operadores y ecuaciones a resolver, disminuye el costo computacional del problema.

Dada la formulación final con Stern Layer que se muestra en la ecuación (39), es fácil observar que al sumar las dos primeras ecuaciones, se elimina el término K_L^1, reduciendo de manera manera inmediata un operador del sistema. El sistema queda de la siguiente manera

Reducción de la formulación integral

Las ecuaciones integrales que gobiernan el modelo con Stern Layer son las que se muestran en (41), (42) y (43).

Ecuaciones Forma Integral

$$\phi_1(r) + \int_{\Gamma 1} \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' - \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial n}(r') \, \mathrm{d}r' = \frac{1}{4\pi\epsilon_1} \sum_k \frac{q_k}{|r - r_k|} \tag{41}$$

$$\begin{aligned} \phi_2(r) - \int_{\Gamma 1} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' + \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \, \mathrm{d}r' \\ + \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' - \int_{\Gamma 2} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \, \mathrm{d}r' = 0 \end{aligned} \tag{42}$$

$$\phi_3(r) - \int_{\Gamma 2} \phi_3(r') \frac{\partial \mathrm{G}_Y}{\partial n}(r, r') \, \mathrm{d}r' + \int_{\Gamma 2} \mathrm{G}_Y(r, r') \frac{\partial \phi_3}{\partial n}(r') \, \mathrm{d}r' = 0 \tag{43}$$

Al evaluar las dos primeras ecuaciones sobre la superficie interna y aplicando las condiciones de borde conocidas $\phi_1 = \phi_2$ y $\epsilon_1 \frac{\partial \phi_1}{\partial n} = \epsilon_2 \frac{\partial \phi_2}{\partial n}$, se obtienen las ecuaciones (44) y (9).

$$\frac{\phi_1}{2}(r) + \int_{\Gamma 1} \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' - \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial n}(r') \, \mathrm{d}r' = \frac{1}{4\pi\epsilon_1} \sum_k \frac{q_k}{|r - r_k|} \tag{44}$$

$$\begin{aligned} &\frac{\phi_1}{2}(r) - \int_{\Gamma 1} \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' + \frac{\epsilon_1}{\epsilon_2} \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial n}(r') \, \mathrm{d}r' \\ &\quad + \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' - \int_{\Gamma 2} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \, \mathrm{d}r' = 0 \end{aligned} \tag{45}$$

Al sumar estas dos ecuaciones se obtiene elimina uno de los operadores integrales sobre la superficie Γ_1, quedando la ecuación (46),

$$\begin{aligned} &\phi_1(r) + \left(\frac{\epsilon_1}{\epsilon_2} - 1\right) \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial n}(r') \, \mathrm{d}r' \\ &\quad + \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' - \int_{\Gamma 2} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \, \mathrm{d}r' \\ &\qquad = \frac{1}{4\pi\epsilon_1} \sum_k \frac{q_k}{|r - r_k|} \end{aligned} \tag{46}$$

luego se aplica la derivada normal $\frac{\partial}{\partial n}$ sobre la superficie interna, con esto se logra eliminar la variable ϕ_1 de esta ecuación.

$$\begin{aligned} &\frac{\partial \phi_1}{\partial n}(r) + \left(\frac{\epsilon_1}{\epsilon_2} - 1\right) \int_{\Gamma 1} \frac{\partial \mathrm{G}_L}{\partial n(r)}(r, r') \frac{\partial \phi_1}{\partial n}(r') \, \mathrm{d}r' \\ &\quad + \frac{\partial}{\partial n(r)} \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' - \int_{\Gamma 2} \frac{\partial \mathrm{G}_L}{\partial n(r)}(r, r') \frac{\partial \phi_2}{\partial n}(r') \, \mathrm{d}r' \\ &\qquad = \frac{1}{4\pi\epsilon_1} \frac{\partial}{\partial n} \sum_k \frac{q_k}{|r - r_k|} \end{aligned} \tag{47}$$

Evaluando la ecuación (42) sobre la superficie exterior se obtiene (48).

$$\begin{aligned} &\frac{\phi_2}{2}(r) - \int_{\Gamma 1} \phi_1(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' + \frac{\epsilon_1}{\epsilon_2} \int_{\Gamma 1} \mathrm{G}_L(r, r') \frac{\partial \phi_1}{\partial n}(r') \, \mathrm{d}r' \\ &\quad + \int_{\Gamma 2} \phi_2(r') \frac{\partial \mathrm{G}_L}{\partial n}(r, r') \, \mathrm{d}r' - \int_{\Gamma 2} \mathrm{G}_L(r, r') \frac{\partial \phi_2}{\partial n}(r') \, \mathrm{d}r' = 0 \end{aligned} \tag{48}$$

para simplificar el sistema se desprecia el operador integral $\int_{\Gamma 1} \phi_1 (r') \frac{\partial \mathrm{G}_L}{\partial n} (r, r') \ \mathrm{d}r'$, y al aplicar las condiciones de borde $\phi_2 = \phi_3$ y $\frac{\partial \phi_2}{\partial n} = \frac{\partial \phi_3}{\partial n}$ se obtiene la ecuación (49)

$$\frac{\phi_2}{2} (r) - \int_{\Gamma 2} \phi_2 (r') \frac{\partial \mathrm{G}_Y}{\partial n} (r, r') \ \mathrm{d}r' + \int_{\Gamma 2} \mathrm{G}_Y (r, r') \frac{\partial \phi_2}{\partial n} (r') \ \mathrm{d}r' = 0 \tag{49}$$

El sistema completo al final queda descrito solo por tres ecuaciones y tres variables, simplificando de esta manera la matriz.

$$\begin{bmatrix} I - \left(\frac{\epsilon_1}{\epsilon_2} - 1\right) K_L'^1 & -D_L^2 & -K_L'^2 \\ \frac{\epsilon_1}{\epsilon_2} V_L^1 & \frac{I}{2} + K_L^2 & -V_L^2 \\ & \frac{I}{2} - K_Y^2 & V_Y^2 \end{bmatrix} \cdot \begin{Bmatrix} \frac{\partial \phi}{\partial n}_1 \\ \phi_2 \\ \frac{\partial \phi}{\partial n}_2 \end{Bmatrix} = \begin{Bmatrix} \sum q_k G_L'^k \\ 0 \\ 0 \end{Bmatrix} \tag{50}$$

Influencia de Double Layer

Ya que en la formulación propuesta se busca eliminar un término para simplificar el sistema de ecuaciones, es necesario comparar los resultados de las formulaciones originales sin este término, y de esta manera, determinar si influye de manera significativa en el resultado final del sistema.

La Figura 14 muestra la comparación de la formulación directa con y sin el operador K_L, y el resultado de `PyGBe` sin alteraciones.

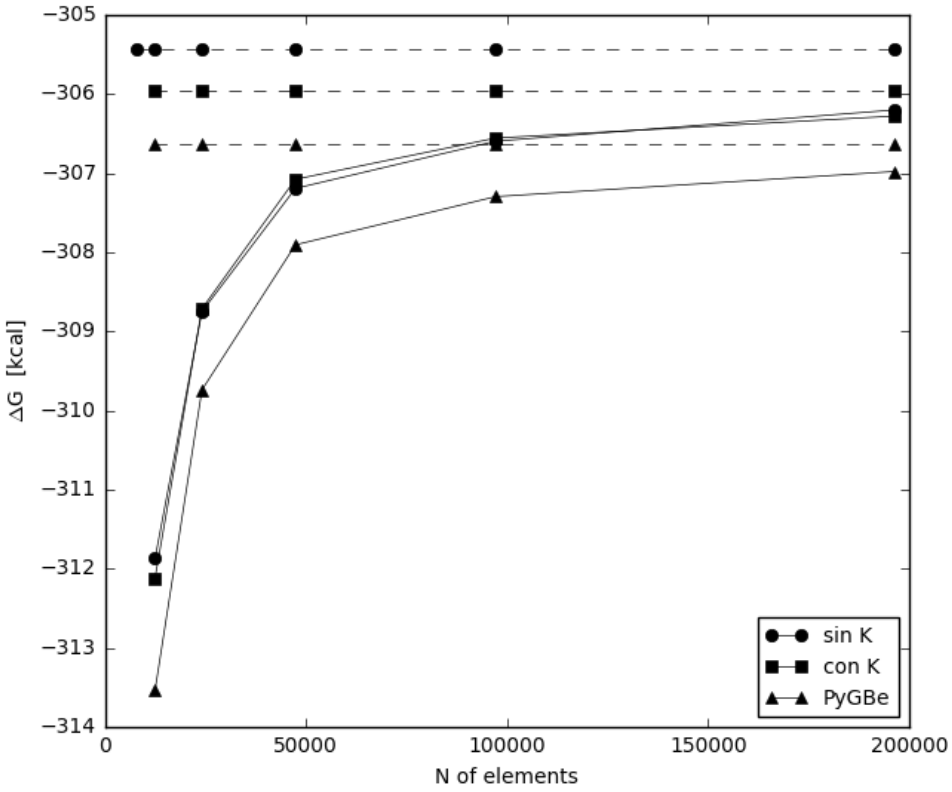


Figura 14: Convergencia de la solución al refinar la malla, comparando formulación directa con y sin el operador K[ϕ]

los resultados de energías obtenidas mediante la extrapolación de Richardson, junto con el orden de convergencia se adjuntan en la siguiente Tabla.

Tabla 2: Valores de energía y convergencia

Formulación	**Energía**	**p**
Directa con K	-305.96	0.91044
Directa sin K	-305.43	0.59845
PyGBe	-306.63	0.94272

Conclusiones

Si bien los resultados al eliminar el término $K_L^{\Gamma 1}$ en la formulación Stern directa no se alteran de manera significativa, el orden de convergencia disminuye bastante. Esto se puede deber a un mal condicionamiento del sistema, lo que conlleva a resultados erroneos.