Uso de XPINNs para Modelar el Potencial Eléctrico de Macromoléculas en un Medio Polarizable a Partir de la Ecuación de Poisson-Boltzmann

Martín Achondo Mercado

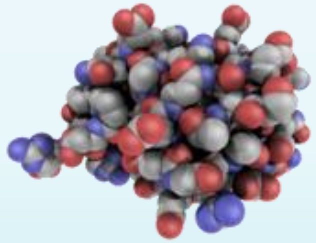
IPM-426

Prof. Guía: Dr. Christopher Cooper

28 de septiembre de 2023

Introducción y Objetivos

Se busca resolver la ecuación de Poisson-Boltzmann utilizando la tecnología de *Extended Physics Informed Neural Networks*.



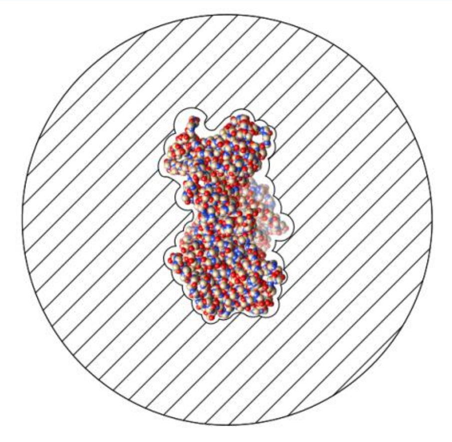
- Programar una ANN que resuelva la PBE.
- Evaluar error y convergencia respecto a hiperparámetros.
- Verificar la validez del método para macromoléculas reales.
- Evaluar el uso del método al incorporar datos experimentales en la función de pérdida.

Motivación uso de PINNs en PBE

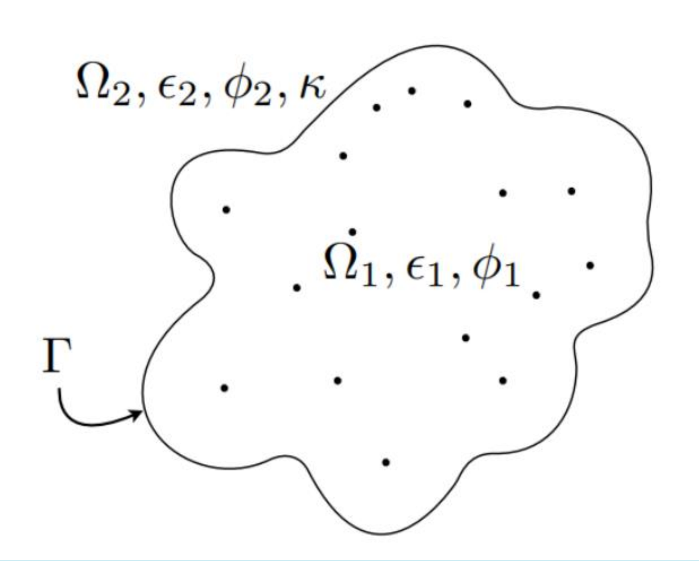
- Método nuevo.
- Fácil de implementar.
- Entrega solución relativamente rápido.
- Se pueden incluir datos experimentales (*effective near-surface potential*).
- Aplicable a problemas inversos, parámetros de PDE desconocidos.
- Facilidad para adaptar solución obtenida para otros escenarios (cargas On-OFF)
- Gran impacto:
  - Ámbito PINNs.
  - Electroestática molecular.

Ecuación de Poisson-Boltzmann

- Aplicada a macromoléculas en un medio polarizable.
- Se obtiene a partir del método de solvente implícito.



$$\begin{cases} \nabla^2 \phi_1 = -\dfrac{1}{\epsilon_1} \displaystyle\sum_k q_k \delta(\mathbf{x} - \mathbf{x}_k) & \mathbf{x} \in \Omega_1 \\ \nabla^2 \phi_2 = \dfrac{2c^\infty q_e}{\epsilon_2} \sinh\left(\dfrac{\phi_2 q_e}{k_b T}\right) & \mathbf{x} \in \Omega_2 \end{cases}$$

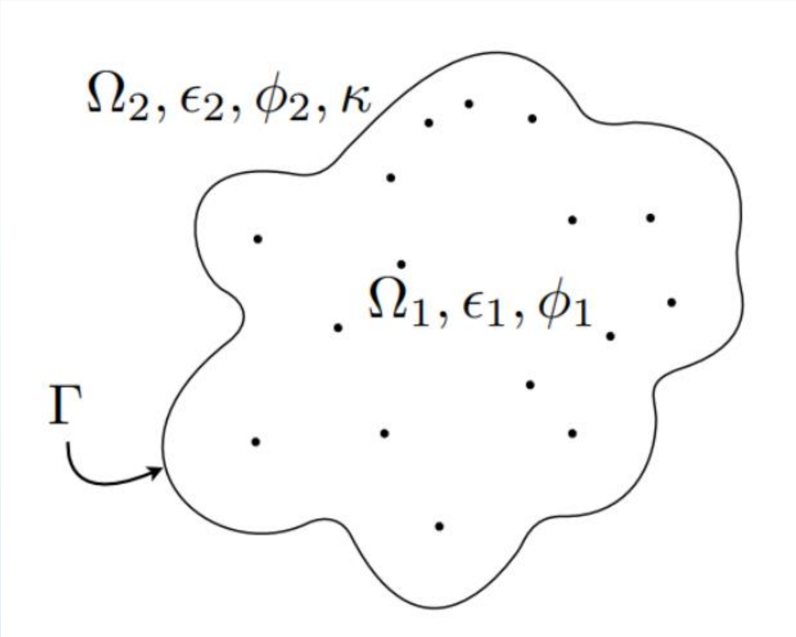


Linealizada: $$\nabla^2 \phi_2 = \kappa^2 \phi_2 \qquad \kappa^2 = \frac{2c^\infty q_e^2}{\epsilon_2 k_b T}$$

Ecuación de Poisson-Boltzmann

- Condiciones en la interfaz:

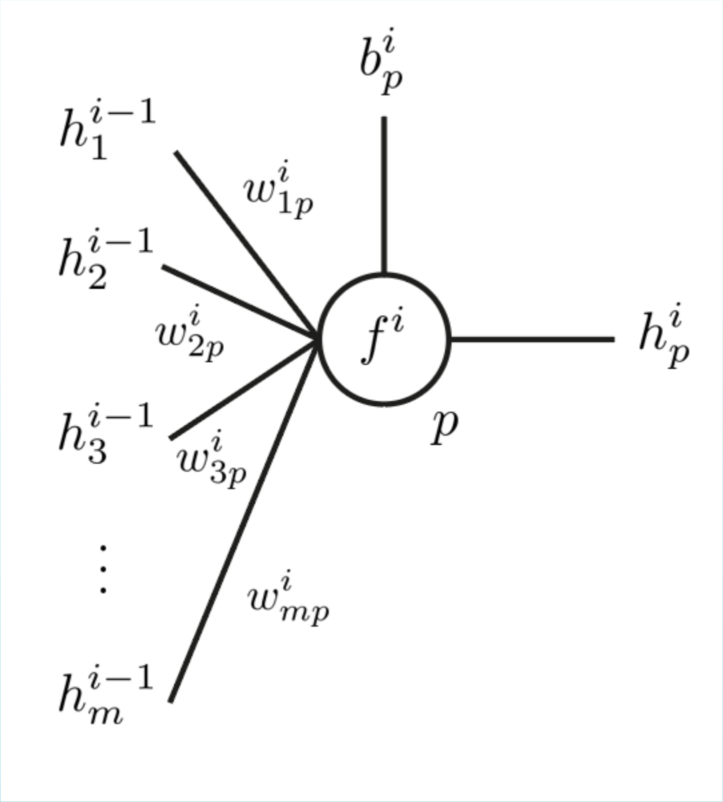
$$\begin{cases} \phi_1 = \phi_2 & \mathbf{x} \in \Gamma \\ \epsilon_1 \dfrac{\partial \phi_1}{\partial n} = \epsilon_2 \dfrac{\partial \phi_2}{\partial n} & \mathbf{x} \in \Gamma \end{cases}$$



Redes Neuronales Artificiales (ANN)

- Operaciones en cada perceptrón (neurona).
- Considerar perceptrón p de la capa i:

Tiene pesos en las conexiones w^i_{mp}, bias b^i_p, y función de activación f^i



$$h^i_p = f^i\left(\sum_{j=1}^{m} h^{i-1}_j w^i_{jp} + b^i_p\right)$$

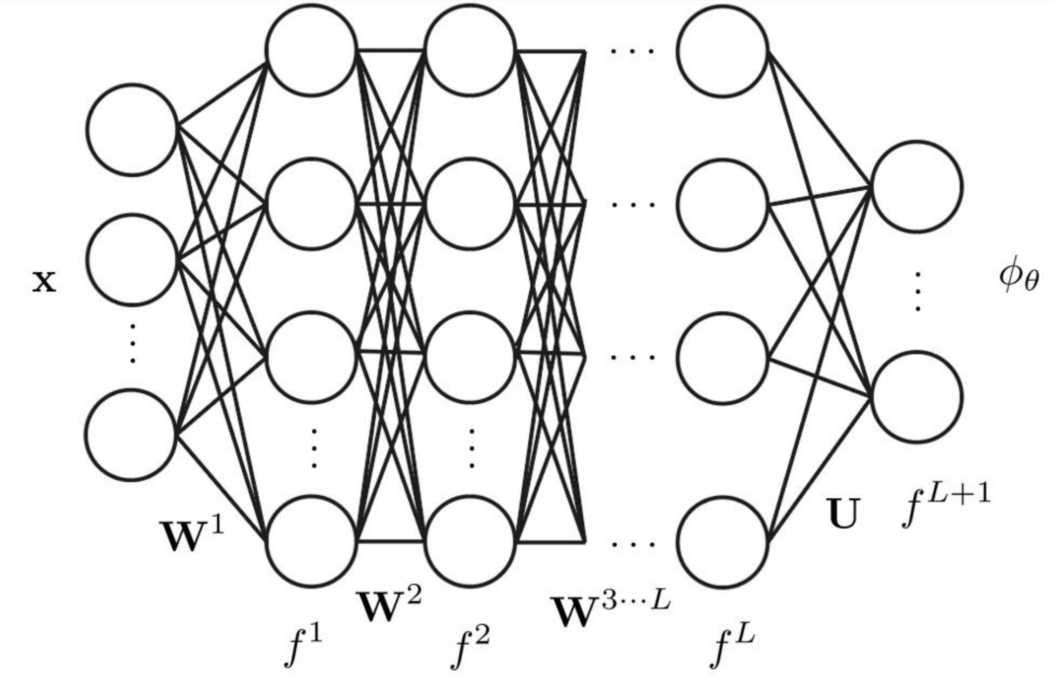
Redes Neuronales Artificiales (ANN)

- Juntar varios perceptrones interconectados genera una ANN.

$$\phi_\theta = \mathcal{N}(\mathbf{x}; \theta)$$

$$\theta = \{\boldsymbol{W}^1, \boldsymbol{b}^1, \ldots, \boldsymbol{W}^L, \boldsymbol{b}^L, \boldsymbol{U}, \boldsymbol{c}\}$$

- Existen distintas arquitecturas:
  - Fully Connected Neural Network
  - Residual Neural Network
  - Convolutional Neural Network



$$\begin{cases} \boldsymbol{h}^1 = f^1(\mathbf{x}\boldsymbol{W}^1 + \boldsymbol{b}^1) & i = 1 \text{ capa entrada} \\ \boldsymbol{h}^i = f^i(\boldsymbol{h}^{i-1}\boldsymbol{W}^i + \boldsymbol{b}^i) & 2 \leq i \leq L \text{ capas ocultas} \\ \phi_\theta = f^{L+1}(\boldsymbol{h}^L\boldsymbol{U} + \boldsymbol{c}) & i = L+1 \text{ salida} \end{cases}$$

Redes Neuronales Artificiales (ANN)

- ¿Cómo se ajustan los parámetros de la ANN?
- Algoritmos de optimización basados en descenso de gradiente.

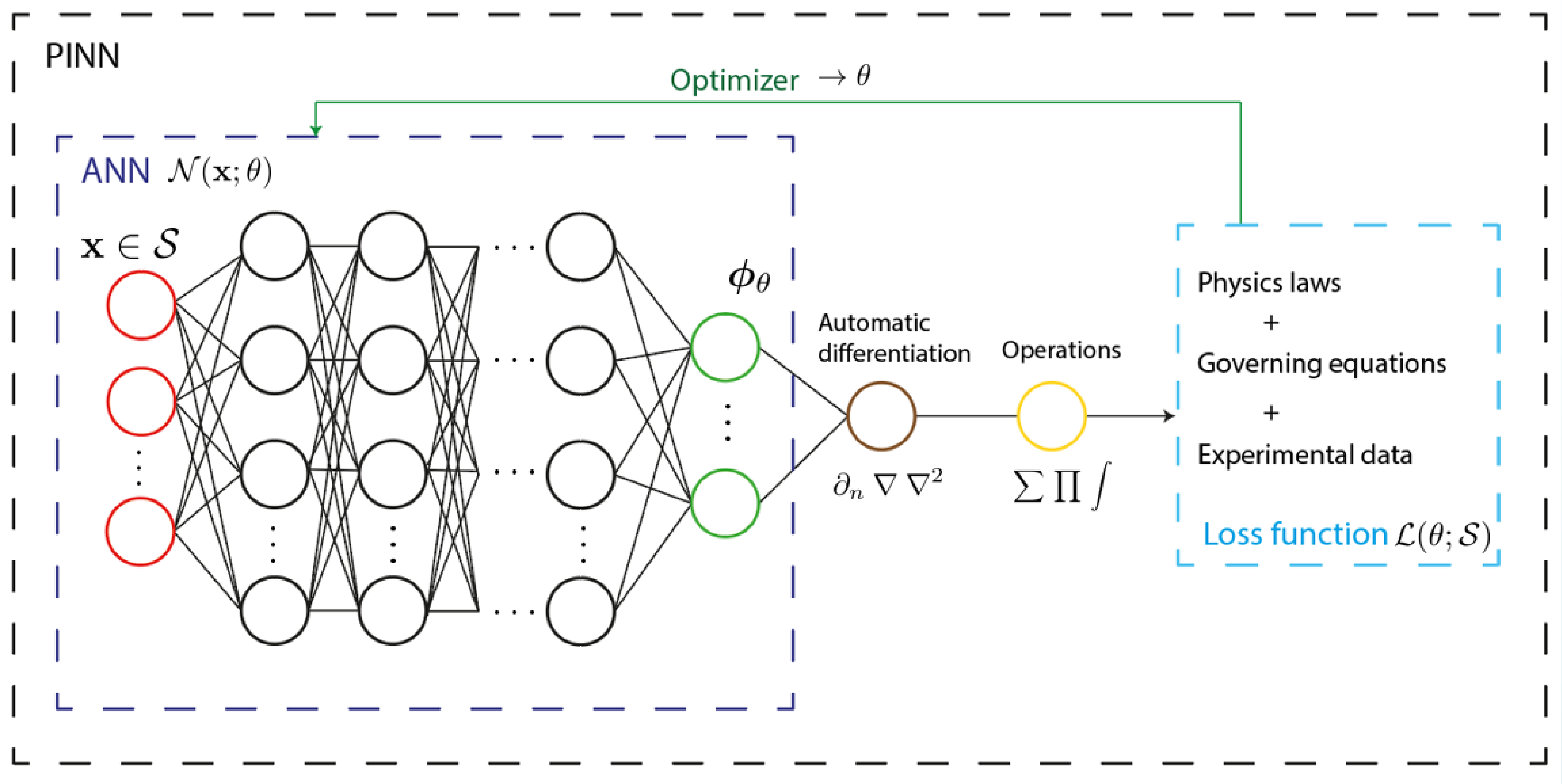
$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}(\theta; \mathcal{S})$$

- Función de pérdida $\mathcal{L}$ se evalúa en conjunto $\mathcal{S}$.
- Algoritmos: ADAM, L-BFGS + Métodos estocásticos.

$$\theta_{k+1} = \theta_k - \lambda \nabla_\theta \mathcal{L}(\theta_k; \mathcal{S}_j) \qquad \mathcal{S}_j \subset \mathcal{S}$$

Physics Informed Neural Networks (PINNs)

- Se le agregan las leyes y ecuaciones físicas a la función de pérdida.



Tipos de PINNs

Se diferencian en la forma en que se resuelve la PDE:

- Deep Collocation Method (DCM): Utilizan los residuales en los puntos del dominio para formar la función de pérdida.
- Deep Variational Method (DVM): Utiliza la formulación variacional para formar la función de pérdida.
- Deep Boundary Integral Method (DBIM): Utiliza la formulación integral en la frontera para formar la función pérdida.

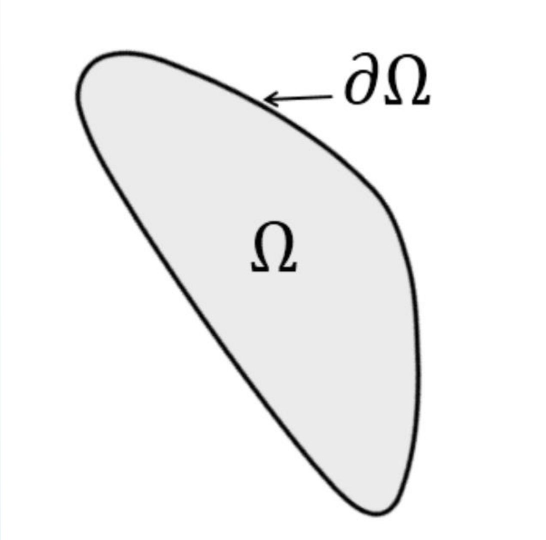
Tipos de PINNs

Se diferencian en la forma en que se resuelve la PDE:

- Deep Collocation Method (DCM): Utilizan los residuales en los puntos del dominio para formar la función de pérdida.
- Deep Variational Method (DVM): Utiliza la formulación variacional para formar la función de pérdida.
- Deep Boundary Integral Method (DBIM): Utiliza la formulación integral en la frontera para formar la función pérdida.

Ejemplo DCM

$$\begin{cases} \mathcal{D}\phi = f(\mathbf{x}) & \mathbf{x} \in \Omega \\ \mathcal{B}\phi = g(\mathbf{x}) & \mathbf{x} \in \partial\Omega \end{cases}$$



$$\boldsymbol{\phi} \approx \boldsymbol{\phi}_\theta = \mathcal{N}(\mathbf{x};\theta)$$

$$\mathcal{L}(\theta;\mathcal{S}) = w_{pde}\mathcal{L}_{pde}(\theta;\mathcal{S}_{pde}) + w_{bc}\mathcal{L}_{bc}(\theta;\mathcal{S}_{bc}) + w_{data}\mathcal{L}_{data}(\theta;\mathcal{S}_{data})$$

Ejemplo DCM

$$\mathcal{L}(\theta;\mathcal{S}) = w_{pde}\mathcal{L}_{pde}(\theta;\mathcal{S}_{pde}) + w_{bc}\mathcal{L}_{bc}(\theta;\mathcal{S}_{bc}) + w_{data}\mathcal{L}_{data}(\theta;\mathcal{S}_{data})$$

$$\mathcal{L}_{pde}(\theta;\mathcal{S}_{pde}) = \frac{1}{N_{pde}} \sum_{x_i \in \mathcal{S}_{pde}} \Big[\mathcal{D}\phi_\theta(x_i) - f(x_i)\Big]^2$$

$$\mathcal{L}_{bc}(\theta;\mathcal{S}_{bc}) = \frac{1}{N_{bc}} \sum_{x_i \in \mathcal{S}_{bc}} \Big[\mathcal{B}\phi_\theta(x_i) - g(x_i)\Big]^2$$

$$\mathcal{L}_{data}(\theta;\mathcal{S}_{data}) = \frac{1}{N_{data}} \sum_{x_i \in \mathcal{S}_{data}} \Big[\mathcal{A}\phi_\theta(x_i) - \mathcal{A}\phi_{data}(x_i)\Big]^2$$

Múltiples Dominios

- Se utiliza una ANN para cada dominio (XPINNs):

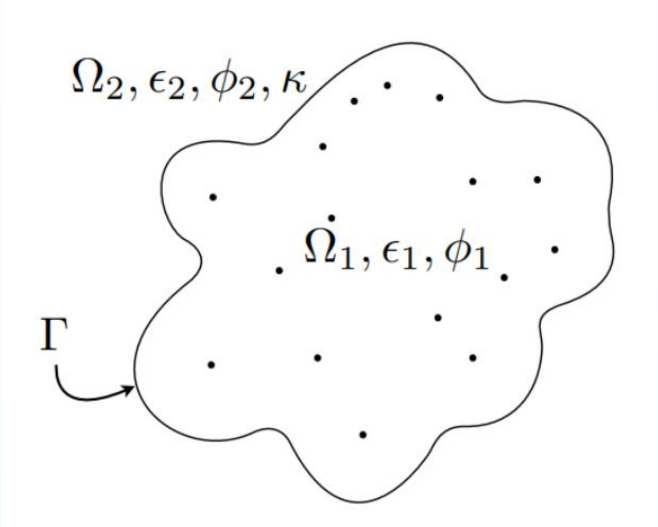
$$\phi \approx \phi_\theta = \begin{cases} \mathcal{N}_1(\mathbf{x};\theta^1) & \mathbf{x} \in \Omega_1 \\ \mathcal{N}_2(\mathbf{x};\theta^2) & \mathbf{x} \in \Omega_2 \end{cases}$$

- Para NN *j*:

$$\mathcal{L}^j = w_{bc}\mathcal{L}^j_{bc} + w_{pde}\mathcal{L}^j_{pde} + w_{data}\mathcal{L}^j_{data} + w_I\mathcal{L}^j_I$$

$$\mathcal{L}^j_I(\theta;\mathcal{S}_I) = \frac{1}{N_I}\sum_{x_i \in \mathcal{S}_I}\left[\mathcal{C}_1\phi^j_\theta(x_i) - \overline{\mathcal{C}\phi}_\theta(x_i)\right]^2$$

DCM Aplicado a PBE



$$\phi \approx \phi_\theta = \begin{cases} \mathcal{N}_1(\mathbf{x};\theta^1) & \mathbf{x} \in \Omega_1 \\ \mathcal{N}_2(\mathbf{x};\theta^2) & \mathbf{x} \in \Omega_2 \end{cases}$$

- Residuales:

$$\mathcal{L}^1_{pde}(\mathcal{S}_{pde}) = \frac{1}{N_{pde}} \sum_{x_i \in \mathcal{S}_{pde}} \left[\nabla^2 \phi^1_\theta(x_i) + \frac{1}{\epsilon_1} \sum_k q_k \delta(x_i - x_k)\right]^2$$

$$\mathcal{L}^2_{pde}(\mathcal{S}_{pde}) = \frac{1}{N_{pde}} \sum_{x_i \in \mathcal{S}_{pde}} \left[\nabla^2 \phi^2_\theta(x_i) - \kappa^2 \phi^2_\theta(x_i)\right]^2$$

DCM Aplicado a PBE

- Condición de borde:

$$\mathcal{L}_{bc}^{2}(\mathcal{S}_{bc}) = \frac{1}{N_{bc}} \sum_{x_i \in \mathcal{S}_{bc}} \left[\phi_\theta^2(x_i) - \frac{1}{4\pi\epsilon_2} \sum_k \frac{q_k \mathrm{e}^{-\kappa|x_i - x_k|}}{|x_i - x_k|} \right]^2$$

- Interfaz (red *j*):

$$\mathcal{L}_I^j(\mathcal{S}_I) = \frac{1}{N_I} \sum_{x_i \in \mathcal{S}_I} \left[\phi_\theta^j(x_i) - \overline{\phi}_\theta(x_i) \right]^2 + \frac{1}{N_I} \sum_{x_i \in \mathcal{S}_I} \left[\epsilon_j \partial_n \phi_\theta^j(x_i) - \overline{\epsilon \partial_n \phi}_\theta(x_i) \right]^2$$

Datos experimentales

- Solución conocida (software PyGBe) (red *j*):

$$\mathcal{L}^j_{data}(\mathcal{S}_{data}) = \frac{1}{N_{data}} \sum_{x_i \in \mathcal{S}_{data}} \left[\phi^j_\theta(x_i) - \phi^*_\theta(x_i)\right]^2$$

- Datos experimentales (effective near-surface potential)

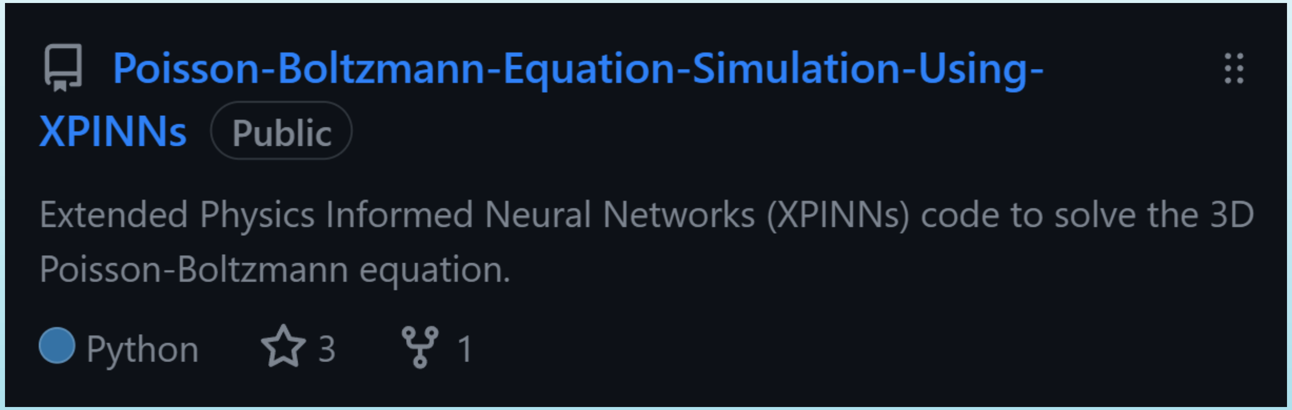
  Para cada átomo de hidrógeno *h*:

$$\phi_{ENS,\theta}(x_h) = \frac{-k_b T}{2q_e} \ln\left(\frac{\sum_{x_i \in \mathcal{S}_E} |x_i - x_h|^{-6} \mathrm{e}^{-\frac{q_e \phi_\theta(x_i)}{k_b T}}}{\sum_{x_i \in \mathcal{S}_E} |x_i - x_h|^{-6} \mathrm{e}^{\frac{q_e \phi_\theta(x_i)}{k_b T}}} \right)$$

$$\mathcal{L}_E(\mathcal{S}_H) = \frac{1}{N_H} \sum_{x_h \in \mathcal{S}_H} \left[\phi_{ENS,\theta}(x_h) - \phi_{ENS}(x_h)\right]^2$$

Avances

- Se tiene un código (elaboración propia) en Python y Tensorflow para resolver la ecuación de Poisson-Boltzmann usando XPINNs.
- Actualmente adaptado solo con DCM.
- Link Github: https://github.com/MartinAchondo/Poisson-Boltzmann-Equation-Simulation-Using-XPINNs

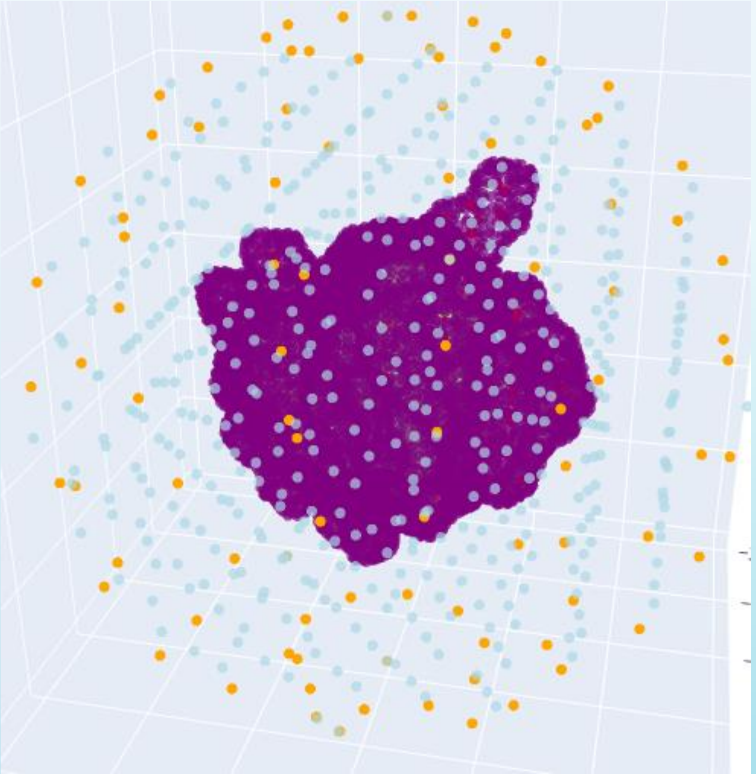
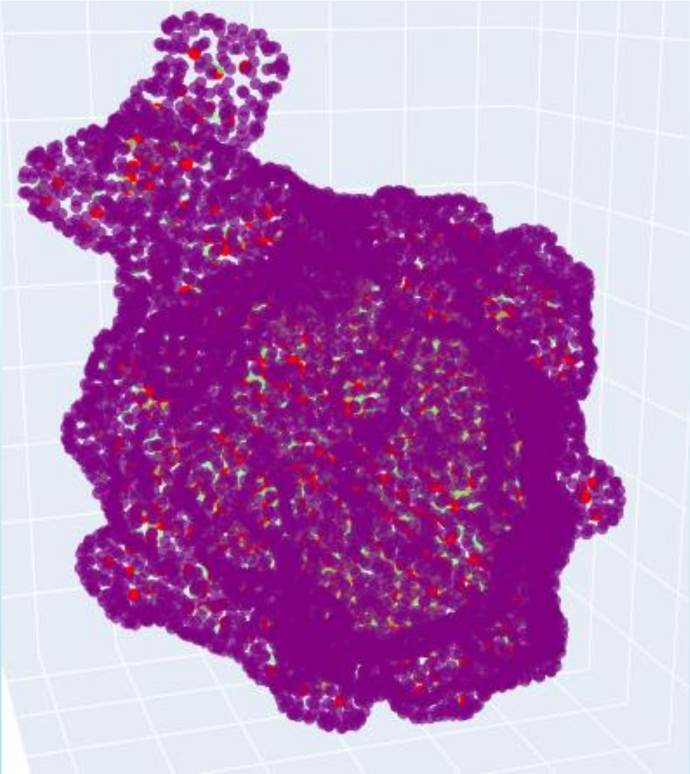
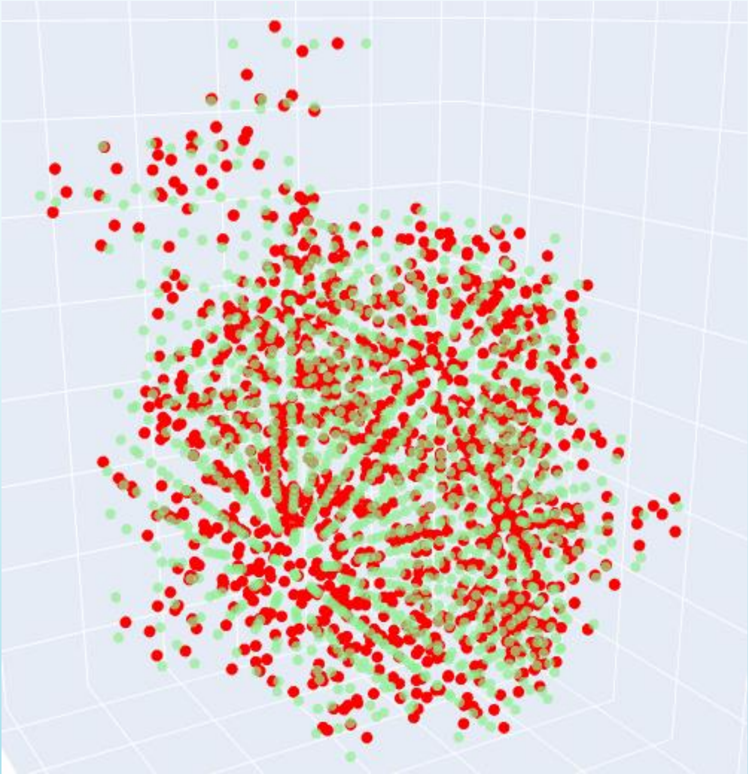


Código

- El código lee información estructural de una molécula.
- Entrega el potencial electroestático en todo el dominio.
- Hiperparámetros, arquitecturas y optimizador modificables.
- Tiene precondicionamiento y ponderación de las funciones de pérdidas.
- Tiene la posibilidad de añadir datos experimentales.

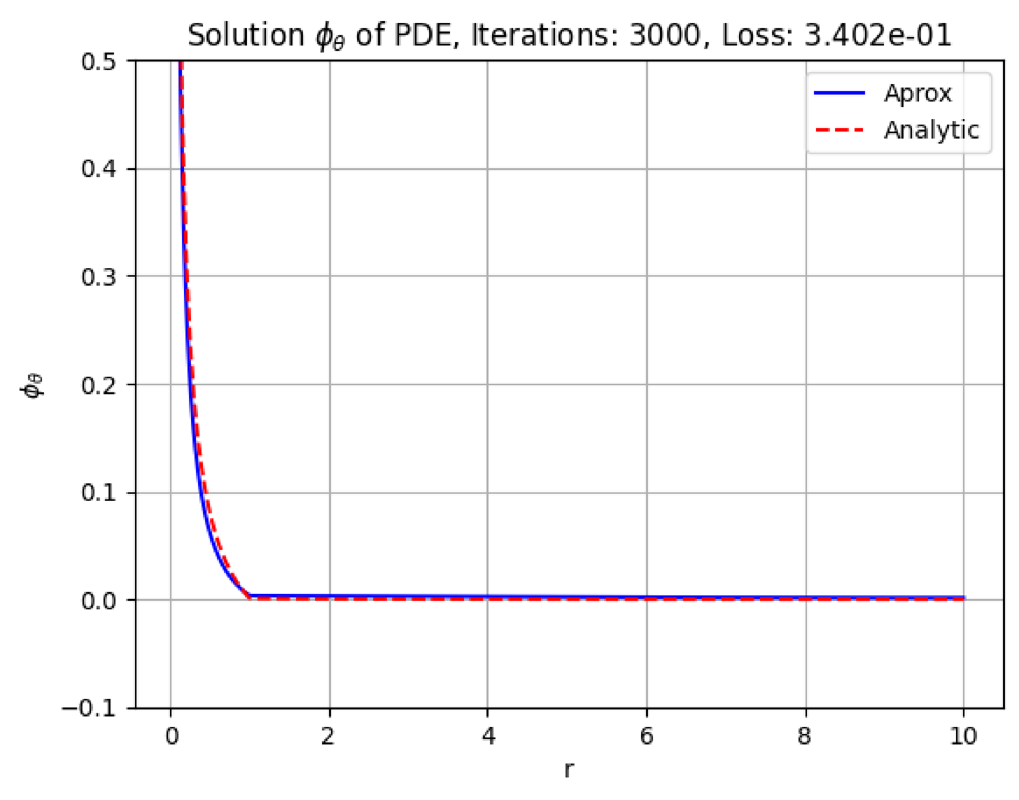
Código

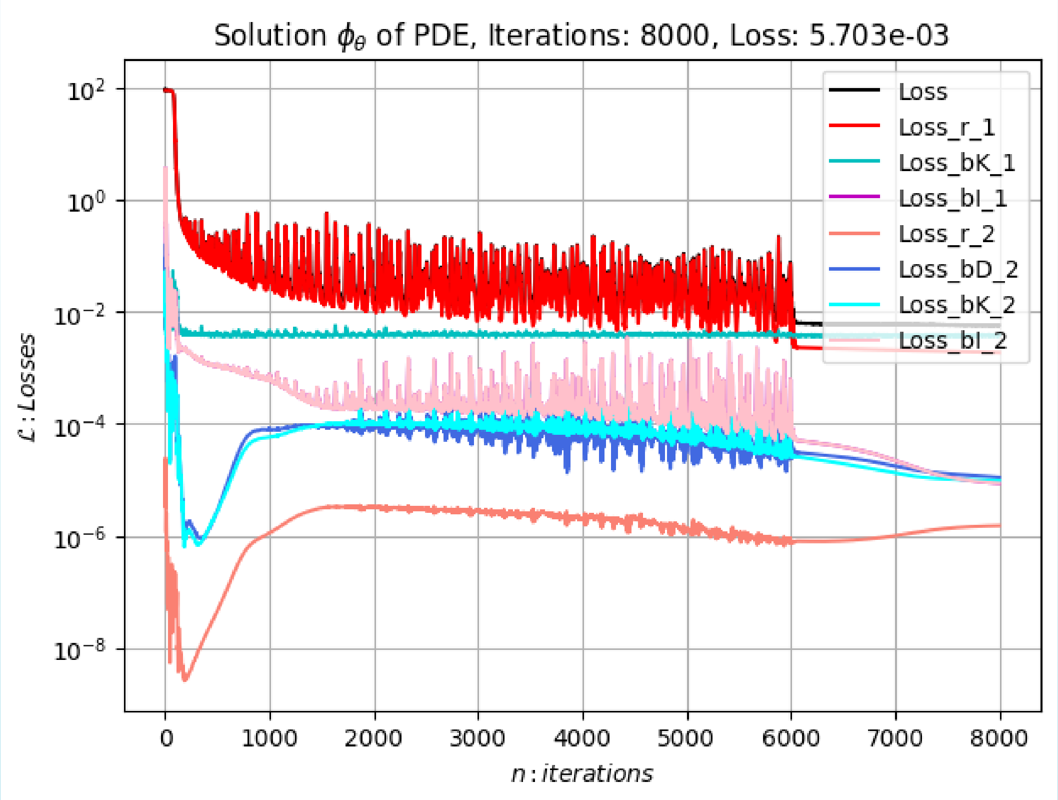
- Ejemplo de puntos de colocación: Ubiquitina

Resultados Preliminares

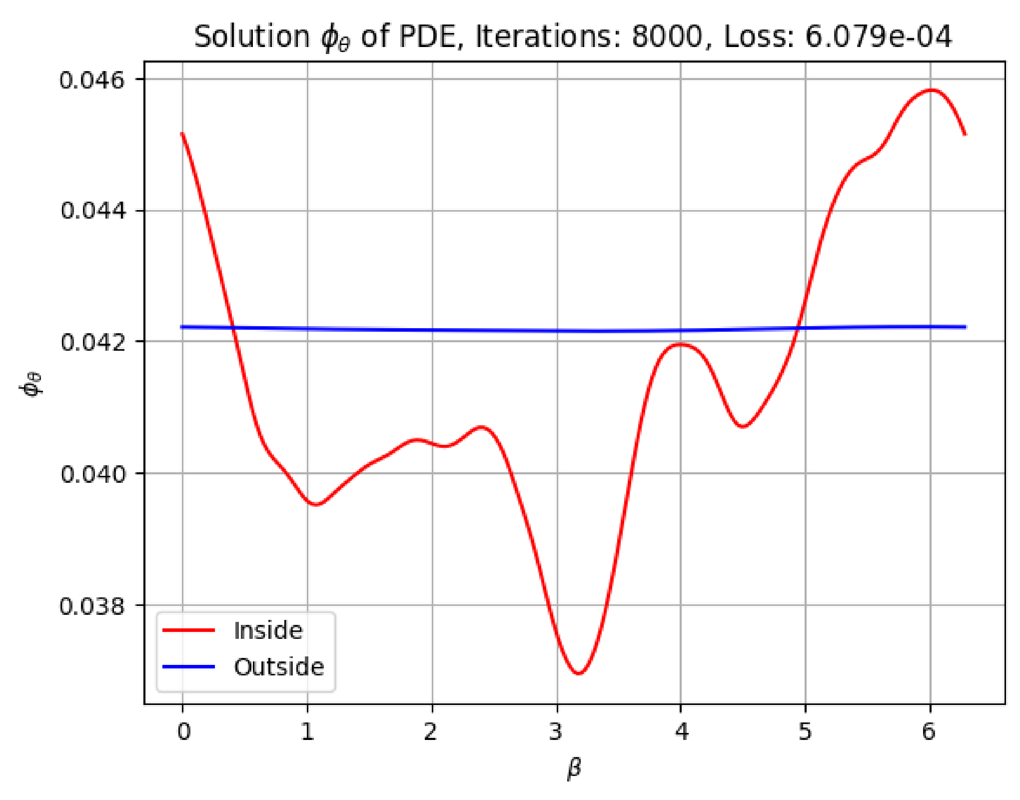
- Molécula esférica con carga puntual en el centro (Validación):

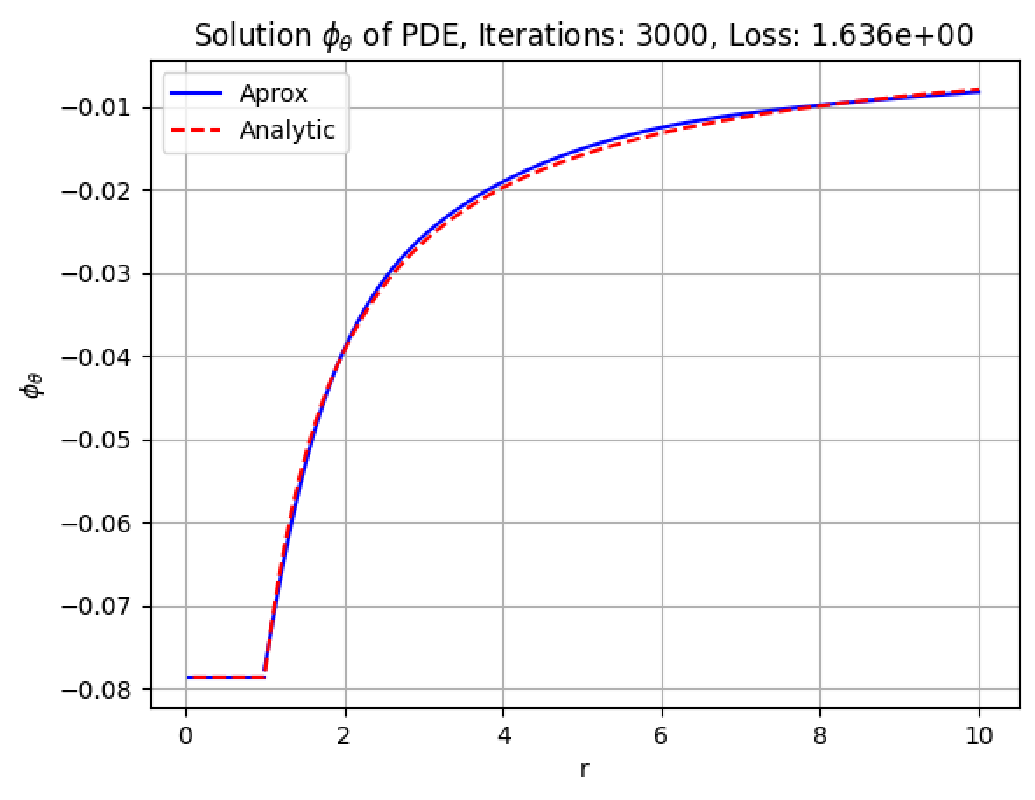




Resultados Preliminares

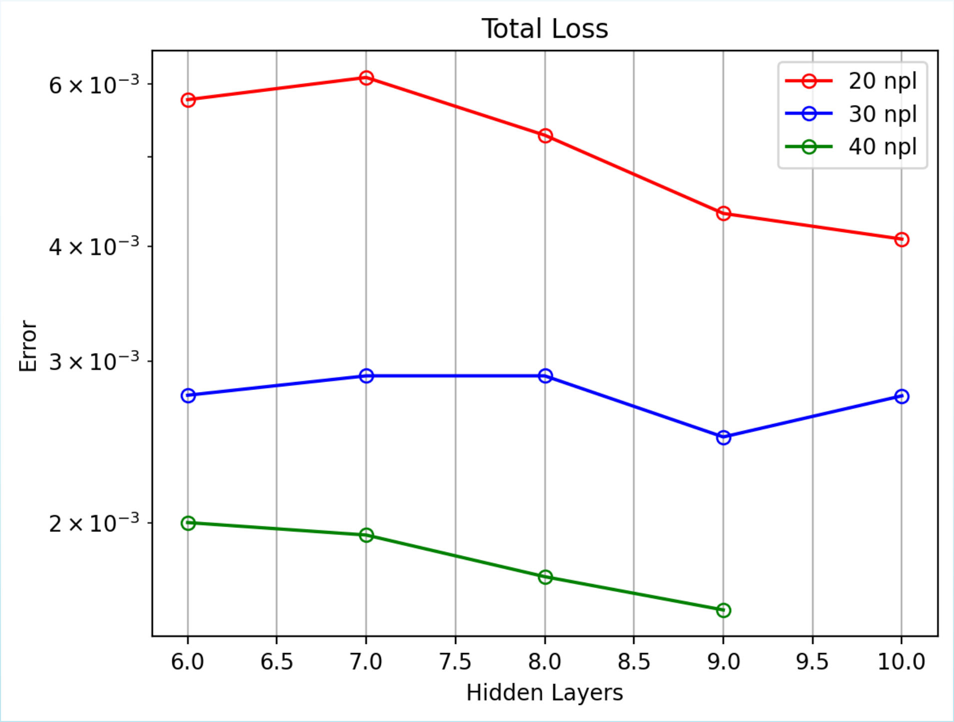
- Molécula esférica con carga puntual en el centro (Validación):

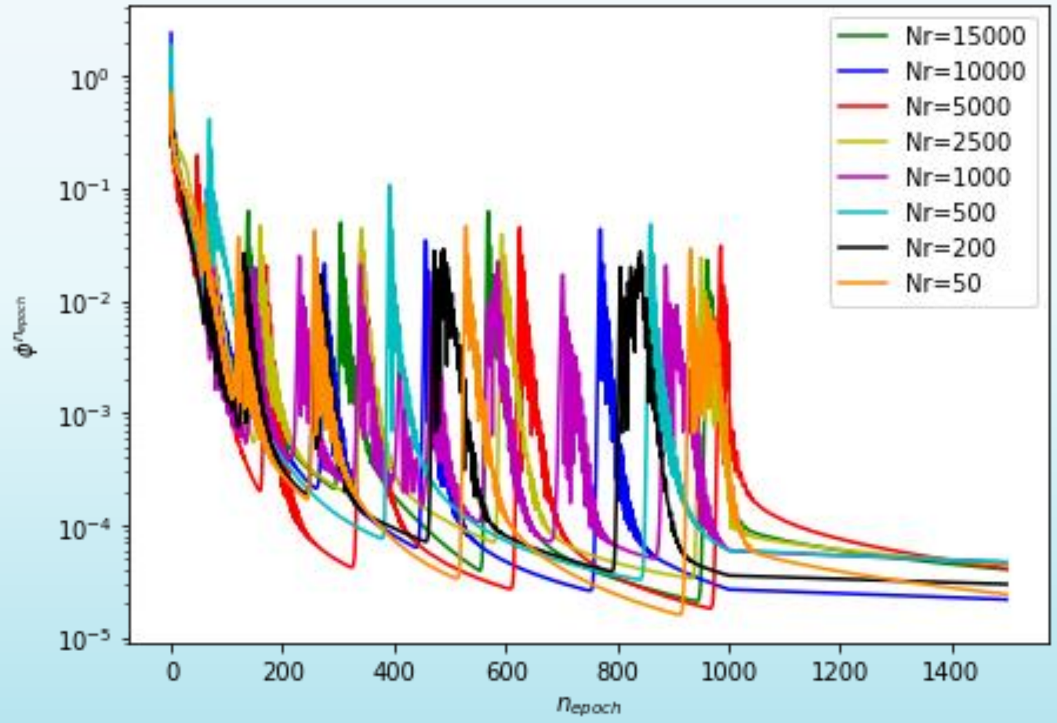




Resultados Preliminares

- Molécula esférica con carga puntual en el centro (Caso 2D) Ec. Poisson:

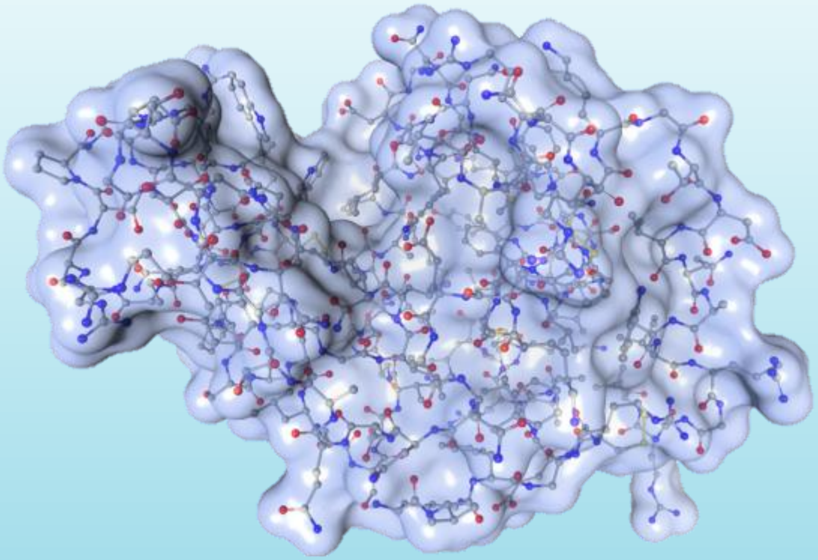




- Se ha notado que caso 3D es bastante más complejo que 2D.
- No son extrapolable.

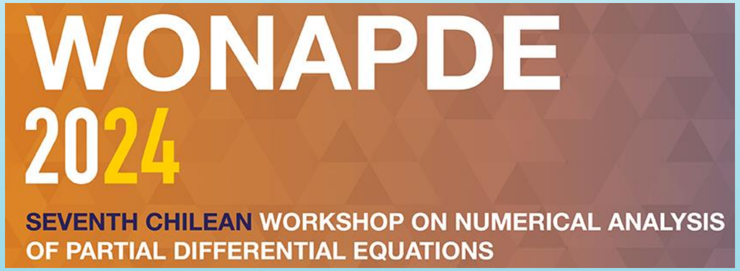
Próximos Desafíos

- Programar método DBIM (código base ya está hecho).
- Correr códigos (DCM y DBIM) para varias moléculas !!



Conclusiones

- Hay avance y se han obtenido los resultados esperados (casos simples).
- Falta trabajo por hacer.
- Se espera poder tener resultados con moléculas reales para enero (inscrito en conferencia WONAPDE).
- Se espera para la misma fecha tener resultados publicables.



Referencias

1) Baker, N. A. (2004). Poisson--Boltzmann methods for biomolecular electrostatics. Methods in enzymology, 383, 94--118. Elsevier.
2) Fogolari, F., Brigo, A., & Molinari, H. (2002). The Poisson--Boltzmann equation for biomolecular electrostatics: a tool for structural biology. Journal of Molecular Recognition, 15(6), 377--392. Wiley Online Library.
3) Yoon, B. J., & Lenhoff, A. M. (1990). A boundary element method for molecular electrostatics with electrolyte effects. Journal of Computational Chemistry, 11(9), 1080--1086. Wiley Online Library.
4) Lee, A., Geng, W., & Zhao, S. (2021). Regularization methods for the Poisson-Boltzmann equation: comparison and accuracy recovery. Journal of Computational Physics, 426, 109958. Elsevier.
5) Roux, B., & Simonson, T. (1999). Implicit solvent models. Biophysical chemistry, 78(1-2), 1--20. Elsevier.
6) Kellogg, O. D. (1953). Foundations of potential theory. Courier Corporation.
7) Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. Nature Reviews Physics, 3(6), 422--440. Nature Publishing Group UK London.
8) Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational physics, 378, 686--707. Elsevier.
9) Kharazmi, E., Zhang, Z., & Karniadakis, G. E. (2019). Variational physics-informed neural networks for solving partial differential equations. arXiv preprint arXiv:1912.00873.
10) Cai, S., Mao, Z., Wang, Z., Yin, M., & Karniadakis, G. E. (2021). Physics-informed neural networks (PINNs) for fluid mechanics: A review. Acta Mechanica Sinica, 37(12), 1727--1738. Springer.

Referencias

11) Wang, S., Sankaran, S., Wang, H., & Perdikaris, P. (2023). An Expert's Guide to Training Physics-informed Neural Networks. arXiv preprint arXiv:2308.08468.
12) Sun, Y., Sun, Q., & Qin, K. (2021). Physics-Based Deep Learning for Flow Problems. Energies, 14(22), 7760. MDPI.
13) Krishnapriyan, A., Gholami, A., Zhe, S., Kirby, R., & Mahoney, M. W. (2021). Characterizing possible failure modes in physics-informed neural networks. Advances in Neural Information Processing Systems, 34, 26548--26560.
14) Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., & Piccialli, F. (2022). Scientific machine learning through physics--informed neural networks: Where we are and what’s next. Journal of Scientific Computing, 92(3), 88. Springer.
15) Shin, Y., Darbon, J., & Karniadakis, G. E. (2020). On the convergence of physics informed neural networks for linear second-order elliptic and parabolic type PDEs. arXiv preprint arXiv:2004.01806.
16) Wu, S., Zhu, A., Tang, Y., & Lu, B. (2022). On convergence of neural network methods for solving elliptic interface problems. arXiv preprint arXiv:2203.03407.
17) Teng, Y., Zhang, X., Wang, Z., & Ju, L. (2022). Learning green’s functions of linear reaction-diffusion equations with application to fast numerical solver. In Mathematical and Scientific Machine Learning (pp. 1--16). PMLR.
18) Li, S., & Feng, X. (2022). Dynamic Weight Strategy of Physics-Informed Neural Networks for the 2D Navier--Stokes Equations. Entropy, 24(9), 1254. MDPI.
19) Mills, E., & Pozdnyakov, A. (2022). Stochastic Scaling in Loss Functions for Physics-Informed Neural Networks. arXiv preprint arXiv:2208.03776.
20) Hu, Z., Jagtap, A. D., Karniadakis, G. E., & Kawaguchi, K. (2021). When do extended physics-informed neural networks (XPINNs) improve generalization? arXiv preprint arXiv:2109.09444.

Referencias

21) Jagtap, A. D., & Karniadakis, G. E. (2021). Extended Physics-informed Neural Networks (XPINNs): A Generalized Space-Time Domain Decomposition based Deep Learning Framework for Nonlinear Partial Differential Equations. In AAAI spring symposium: MLPS (Vol. 10).

22) Sun, J., Liu, Y., Wang, Y., Yao, Z., & Zheng, X. (2023). BINN: A deep learning approach for computational mechanics problems based on boundary integral equations. Computer Methods in Applied Mechanics and Engineering, 410, 116012. Elsevier.

23) Lin, G., Hu, P., Chen, F., Chen, X., Chen, J., Wang, J., & Shi, Z. (2021). BINet: learning to solve partial differential equations with boundary integral networks. arXiv preprint arXiv:2110.00352.

24) LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436--444. Nature Publishing Group UK London.

25) Hornik, K., Stinchcombe, M., & White, H. (1990). Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. Neural networks, 3(5), 551--560. Elsevier.

26) He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition, 770--778.

27) Jagtap, A. D., Kharazmi, E., & Karniadakis, G. E. (2020). Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. Computer Methods in Applied Mechanics and Engineering, 365, 113028. Elsevier.

28) Yu, B., Pletka, C. C., Pettitt, B. M., & Iwahara, J. (2021). De novo determination of near-surface electrostatic potentials by NMR. Proceedings of the National Academy of Sciences, 118(25), e2104020118. National Acad Sciences.

29) Developers, TensorFlow. TensorFlow. Zenodo, 2022.

30) Martín Achondo. Poisson-Boltzmann-Equation-Simulation-Using-XPINNs. 2023. [En línea]. Disponible en: https://github.com/MartinAchondo/Poisson-Boltzmann-Equation-Simulation-Using-XPINNs.

Uso de XPINNs para Modelar el Potencial Eléctrico de Macromoléculas en un Medio Polarizable a Partir de la Ecuación de Poisson-Boltzmann

Martín Achondo Mercado

IPM-426

Prof. Guía: Dr. Christopher Cooper

28 de septiembre de 2023

UNIVERSIDAD TECNICA FEDERICO SANTA MARIA

DEPARTAMENTO DE INGENIERÍA MECÁNICA

Otros posibles términos: Función de pérdida

- Datos experimentales (energía libre de solvatación):

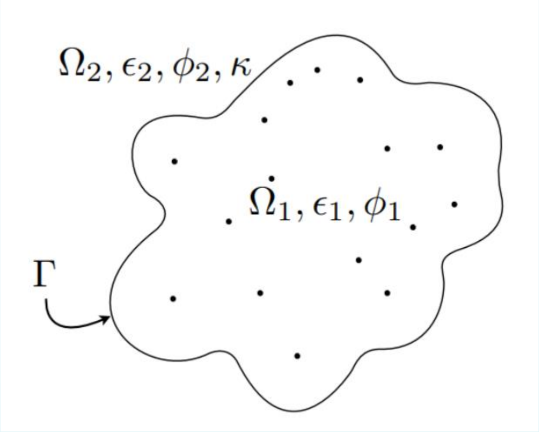
$$\mathcal{L}_{solv} = \left| \frac{1}{2} \sum_k q_k (\phi_\theta - \phi_{coulomb})(x_k) - G_{solv,exp} \right|^2$$

- Leyes Físicas (Ley de Gauss)

$$\mathcal{L}_{Gauss} = \left| \oint_\Gamma \epsilon_1 \partial_n \phi_\theta(x') \, \mathrm{d}S(x') - \sum_k q_k \right|^2$$

Formulación Integral

- Utiliza las identidades de Green (forma lineal PBE):



$$\phi_1(\mathbf{x}^s) = \frac{1}{2\pi\epsilon_1} \sum_k \frac{q_k}{|\mathbf{x}^s - \mathbf{x}_k|} + \frac{1}{2\pi} \oint_\Gamma \frac{\partial \phi_1}{\partial n} \frac{1}{|\mathbf{x}^s - \mathbf{x}'|} \,\mathrm{d}S(\mathbf{x}')$$

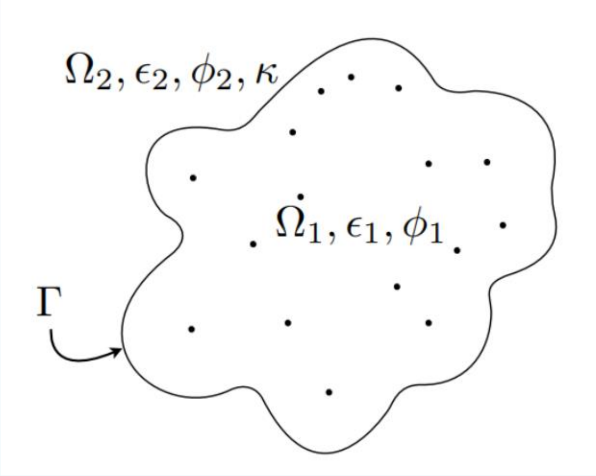
$$-\frac{1}{2\pi} \oint_\Gamma \phi_1 \frac{\partial}{\partial n} \left(\frac{1}{|\mathbf{x}^s - \mathbf{x}'|} \right) \mathrm{d}S(\mathbf{x}')$$

$$\phi_2(\mathbf{x}^s) = -\frac{1}{2\pi} \oint_\Gamma \frac{\partial \phi_2}{\partial n} \frac{\exp(-\kappa|\mathbf{x}^s - \mathbf{x}'|)}{|\mathbf{x}^s - \mathbf{x}'|} \,\mathrm{d}S(\mathbf{x}')$$

$$+\frac{1}{2\pi} \oint_\Gamma \phi_2 \frac{\partial}{\partial n} \left(\frac{\exp(-\kappa|\mathbf{x}^s - \mathbf{x}'|)}{|\mathbf{x}^s - \mathbf{x}'|} \right) \mathrm{d}S(\mathbf{x}')$$

DBIM Aplicado a PBE

$$\phi_\theta^1 = \mathcal{N}(\mathbf{x};\theta) \quad\quad \mathbf{x} \in \Gamma$$



- La ANN es entrenada con la formulación integral en la interfaz.

- La función de pérdida se descompone en 2 términos.

$$\mathcal{L}(\theta;\mathcal{S}) = w_1\mathcal{L}_1(\theta;\mathcal{S}_b) + w_2\mathcal{L}_2(\theta;\mathcal{S}_b)$$

- Se necesita un método de cuadratura (librería Bempp).

DBIM Aplicado a PBE

$$\mathcal{L}_1(\theta;\mathcal{S}_b) = \frac{1}{N_b}\sum_{x_i\in\mathcal{S}_b}\left[\phi_\theta^1(x_i) - \frac{1}{2\pi\epsilon_1}\sum_k \frac{q_k}{|x_i - x_k|} - \frac{1}{2\pi}\oint_\Gamma \frac{\partial\phi_\theta^1}{\partial n}(x')\frac{1}{|x_i - x'|}\,\mathrm{d}S(x')\right.$$

$$\left. + \frac{1}{2\pi}\oint_\Gamma \phi_\theta^1(x')\frac{\partial}{\partial n}\left(\frac{1}{|x_i - x'|}\right)\mathrm{d}S(x')\right]^2$$

$$\mathcal{L}_2(\theta;\mathcal{S}_b) = \frac{1}{N_b}\sum_{x_i\in\mathcal{S}_b}\left[\phi_\theta^1(x_i) + \frac{1}{2\pi}\frac{\epsilon_1}{\epsilon_2}\oint_\Gamma \frac{\partial\phi_\theta^1}{\partial n}(x')\frac{\exp(-\kappa|x_i - x'|)}{|x_i - x'|}\,\mathrm{d}S(x')\right.$$

$$\left. - \frac{1}{2\pi}\oint_\Gamma \phi_\theta^1\frac{\partial}{\partial n}(x')\left(\frac{\exp(-\kappa|x_i - x'|)}{|x_i - x'|}\right)\mathrm{d}S(x')\right]^2$$

Algoritmos

- DCM y DBIM:

Algoritmo 1 Resolver PBE con DCM

Input: Loss functions $\mathcal{L}$, Points $\mathcal{S}$, Hyperparameters
  Initialize $\theta_0^1, \mathcal{N}_1, \theta_0^2, \mathcal{N}_2$
  for $k = 1$ to $k = n$ **do**
    Calculate $\phi_\theta^1, \phi_\theta^2$ with $\mathcal{N}_1, \theta_k^1, \mathcal{N}_2, \theta_k^2$
    Calculate $\partial_n \phi_\theta^1, \partial_n \phi_\theta^2$ with AD
    Calculate $\mathcal{D}_1 \phi_\theta^1, \mathcal{D}_2 \phi_\theta^2$ with AD
    Calculate $\mathcal{L}^1$
    Calculate $\mathcal{L}^2$
    Update θ_{k+1}^1 with ADAM$(\theta_k^1, \nabla \mathcal{L}^1)$
    Update θ_{k+1}^2 with ADAM$(\theta_k^2, \nabla \mathcal{L}^2)$
  end for
Output: Parameters θ^1, θ^2

Algoritmo 2 Resolver PBE con DBIM

Input: Loss function $\mathcal{L}$, Points $\mathcal{S}$, Hyperparameters
  Initialize $\theta_0, \mathcal{N}$
  for $k = 1$ to $k = n$ **do**
    Calculate $\phi_\theta^1 = \mathcal{N}(\mathbf{x}; \theta_k)$
    Calculate $\partial_n \phi_\theta^1$ with AD
    Calculate boundary integrals with cuadrature
    Calculate $\mathcal{L}$
    Update θ_{k+1} with ADAM$(\theta_k, \nabla \mathcal{L})$
  end for
Output: Parameters θ

Preacondicionamiento

- Se preacondiciona conjunto de parámetros θ:

$$\mathcal{L}_{pre}(\theta; \mathcal{S}) = \frac{1}{N} \sum_{x_i \in S} \left[\phi_\theta(x_i) - \phi_*(x_i)\right]^2$$

Algoritmo 5 Resolver PDE con DCM preacondicionado

Input: Loss functions $\mathcal{L}$ and $\mathcal{L}_{pre}$, Points $\mathcal{S}$, Hyperparameters

Initialize $\theta_0, \mathcal{N}$
for $k = 1$ to $k = n_{\text{pre}}$ **do**
  Calculate $\phi_\theta = \mathcal{N}(\mathbf{x}; \theta_k)$
  Calculate $\mathcal{L}_{\text{pre}}$
  Update θ_{k+1} with ADAM$(\theta_k, \nabla\mathcal{L})$
end for
for $k = n_{\text{pre}} + 1$ to $k = n$ **do**
  Calculate $\phi_\theta = \mathcal{N}(\mathbf{x}; \theta_k)$
  Calculate $\mathcal{L}$ with PINN
  Update θ_{k+1} with ADAM$(\theta_k, \nabla\mathcal{L})$
end for

Output: Parameters θ

Ajuste de pesos para funciones de pérdida

- Se busca que todos los términos ponderen de igual forma para modificar los parámetros θ.

$$\hat{w}_k = \frac{\sum_j \|\nabla_\theta \mathcal{L}_j\|}{\|\nabla_\theta \mathcal{L}_k\|}$$

$$w_{k,\text{new}} = \alpha w_{k,\text{old}} + (1-\alpha)\hat{w}_k$$

- No se realiza todas las iteraciones, cada 1000 o 2000 funciona bien.

Uso de XPINNs para Modelar el Potencial Eléctrico de Macromoléculas en un Medio Polarizable a Partir de la Ecuación de Poisson-Boltzmann

UNIVERSIDAD TECNICA FEDERICO SANTA MARIA
MEC USM
DEPARTAMENTO DE INGENIERÍA MECÁNICA

Martín Achondo Mercado

IPM-426

Prof. Guía: Dr. Christopher Cooper

28 de septiembre de 2023