

Propuesta de trabajo de maestría

Machine learning methods to solve PDEs

Estudiante: Carlos Daniel Contreras Quiroz

Asesor: Mauricio Junca

Septiembre 2022

1. Descripción

Las ecuaciones en derivadas parciales aparecen comúnmente como herramientas útiles para la modelación de sistemas en múltiples disciplinas. Se encuentran frecuentemente aplicaciones en ciencias naturales como la física y biología, en diseño en ingeniería, y también en áreas como la economía y finanzas. Sin embargo, las propiedades matemáticas de las ecuaciones que aparecen son tan diversas como las áreas en que se aplican, y aunque se pueden clasificar según algunas de sus características, no podría existir una teoría completa que describa nuestro conocimiento sobre estas.

Por otro lado, las soluciones analíticas a estos modelos generalmente no están a nuestro alcance, por lo que es necesario recurrir a métodos numéricos para obtener aproximaciones. Para esto, usualmente se recurre a métodos clásicos como diferencias finitas, elementos finitos, volúmenes finitos o métodos espectrales, para los cuales existe una amplia teoría que soporta y justifica rigurosamente su funcionamiento.

No obstante, la aplicación de estos métodos a problemas particulares a veces se restringe por propiedades específicas de la ecuación que se resuelve. Por ejemplo, los métodos mencionados sufren de la maldición de la dimensionalidad ("*the curse of dimensionality*"), esto es, su complejidad computacional escala exponencialmente en la dimensión del problema, por lo que su uso se restringe a problemas de dimensión baja ($n = 1, 2, 3, 4$). Lo anterior dificulta su uso en aplicaciones como valoración en matemática financiera, donde la dimensión del problema está determinada por el número de activos considerados. También su eficiencia computacional se reduce considerablemente conforme se aumenta la complejidad de los dominios en que se resuelven, o por las no-linealidades que aparecen, como es el caso de la ecuación de Navier-Stokes modelando flujos turbulentos.

Otra área en donde estos inconvenientes aparecen es en el análisis de datos y aprendizaje de máquinas. Por ejemplo, la complejidad de algunos modelos de regresión no lineal crece exponencialmente con el tamaño de los datos subyacentes. Para este tipo de problemas se han desarrollado herramientas poderosas que permiten modelar problemas en altas dimensiones y con posibles no linealidades. Entre estas, las redes neuronales han demostrado ser de gran utilidad como modelo para representar funciones con estas complejidades.

En consecuencia, intentando replicar el éxito obtenido con estas herramientas en aprendizaje de máquinas, recientemente han surgido nuevas perspectivas para aproximar soluciones de ecuaciones en derivadas parciales usando estas mismas herramientas. Entre estas se encuentran las PINNs (Physics Informed Neural Networks), FNO (Fourier Neural Operators), y DGM (Deep Galerkin Method). La evidencia práctica muestra que estos métodos pueden proporcionar soluciones en casos donde los clásicos no, a pesar de usualmente no competir con su eficiencia en las situaciones donde los últimos sí aplican. Además, se ha venido desarrollando un marco teórico riguroso que permite justificar su aplicación en situaciones específicas.

Como objetivo razonable de este trabajo se propone estudiar teóricamente e implementar computacionalmente dos de estos métodos, para establecer sus ventajas y alcances.

El primero de estos se propone en [1], y se basa en explotar la conexión que existe entre las ecuaciones parabólicas (posiblemente no lineales) y las ecuaciones diferenciales estocásticas hacia atrás (BSDE Backward Stochastic Differential Equations). En este caso se considera ecuaciones parabólicas semilineales de la forma

$$\begin{aligned} \frac{\partial u}{\partial t}(t, x) + \frac{1}{2} \text{Tr} \left(\sigma \sigma^T(t, x) (\text{Hess}_x u)(t, x) \right) + \nabla u(t, x) \cdot \mu(t, x) \\ + f(t, x, u(t, x), \sigma^T(t, x) \nabla u(t, x)) = 0, \end{aligned} \quad (1)$$

con condición final $u(T, x) = g(x)$, donde t representa el tiempo, x la variable espacial d -dimensional, μ una función vectorial conocida, σ una matriz $d \times d$ y f una función no lineal. Similar al caso lineal, donde la solución a la ecuación diferencial parcial parabólica se le puede asignar una representación en términos del valor esperado de un proceso asociado a una ecuación diferencial estocástica a través de la formula de Feynman-Kac, se puede asignar una representación de la solución de la ecuación 1 en términos de la solución de una BSDE de la forma

$$\begin{aligned} u(t, X_t) - u(0, X_0) \\ = - \int_0^t f(s, X_s, u(s, X_s), \sigma^T(s, X_s) \nabla u(s, X_s)) ds \\ + \int_0^t [\nabla u(s, X_s)]^T \sigma(s, X_s) dW_s, \end{aligned} \quad (2)$$

donde X_t es la solución de la SDE

$$X_t = \xi + \int_0^t \mu(s, X_s) ds + \int_0^t \sigma(s, X_s) dW_s. \quad (3)$$

En cada paso de tiempo, la ecuación 3 se discretiza mediante el método Euler-Maruyama para obtener muestras de caminos $\{X_n\}_{0 \leq n \leq N}$, y para usar la representación de la solución 2 se aproxima en cada paso la función $x \rightarrow \sigma^T(t, x) \nabla u(t, x)$ con una red neuronal que se entrena minimizando la función de costo

$$l(\theta) = \mathbb{E} \left[\left| g(X_{t_N}) - \hat{u}(\{X_{t_n}\}_{0 \leq n \leq N}, \{W_{t_n}\}_{0 \leq n \leq N}) \right|^2 \right], \quad (4)$$

relacionada con el cumplimiento de las condiciones iniciales de la ecuación. Al final, el conjunto de redes neuronales optimizadas proporciona la aproximación de la solución requerida.

En este artículo se presenta la aplicación de este método para resolver un modelo de Black-Scholes, una ecuación de HJB para un problema de control y una ecuación de Allen-Cahn, todas

en dimensiones altas. A parte de estudiar la relación entre ecuaciones parabólicas no lineales y BSDEs y su uso en este modelo, se pretende implementar uno de los ejemplos dados y estudiar su efectividad.

El segundo método que se quiere estudiar se propone en , y se basa en el algoritmo de Q-learning de reinforcement learning. En este caso se estudian problemas de la forma

$$\begin{aligned}\mathcal{L}u &= 0 \text{ en } \Omega \\ u &= f \text{ en } \partial\Omega,\end{aligned}\tag{5}$$

donde \mathcal{L} es un operador no lineal de segundo orden con ciertas condiciones de regularidad, y Ω es un dominio. Aquí se quiere usar una red neuronal de una capa con N neuronas de la forma

$$S^N(x; \theta) = \frac{1}{N^\beta} \sum_{i=1}^N c^i \sigma(w^i \cdot x + b^i),\tag{6}$$

donde w^i, b^i, c^i son los parámetros de la red y β es un factor de escalamiento. Con esta se propone una aproximación de la función

$$Q^N(x; \theta) := S^N(x; \theta)\eta(x) + (1 - \eta(x))f(x).\tag{7}$$

que se entrenará en cada paso intentando minimizar la función de costo

$$\int_{\Omega} [\mathcal{L}Q^N(x)]^2 d\mu(x),\tag{8}$$

que se puede estimar por métodos de Monte-Carlo. Al final Q^N proporcionará una aproximación de la solución que cumple las condiciones de frontera impuestas.

2. Objetivos

- Estudiar la relación entre BSDE y ecuaciones parabólicas no lineales
- Explotar esta relación para resolver una ecuación diferencial no parabólica usando métodos de Deep Learning
- Estudiar el algoritmo Q-PDE, basado en Q-learning, para resolver ecuaciones diferenciales elípticas.
- Implementar computacionalmente estos métodos.

3. Cronograma tentativo

- Septiembre-Octubre 2022: Aprender a implementar redes neuronales como métodos de aproximación para funciones Q
- Noviembre-Diciembre 2022: Estudiar la relación entre BSDE y ecuaciones diferenciales parabólicas no lineales.
- Enero 2023: Implementar método basado en BSDE.
- Febrero-Marzo 2023: Estudiar e implementar el algoritmo Q-PDE
- Abril 2023: Terminar el documento

Referencias

- [1] S. N. Cohen, D. Jiang, and J. Sirignano, “Neural Q-learning for solving elliptic PDEs,” p. 57, 2022.
- [2] J. Han, A. Jentzen, and W. E, “Solving high-dimensional partial differential equations using deep learning,” *Proceedings of the National Academy of Sciences*, vol. 115, pp. 8505–8510, Aug. 2018.
- [3] J. Blechschmidt and O. G. Ernst, “Three ways to solve partial differential equations with neural networks — A review,” *GAMM-Mitteilungen*, vol. 44, June 2021.
- [4] C. F. Higham and D. J. Higham, “Deep Learning: An Introduction for Applied Mathematicians,” *SIAM Review*, vol. 61, pp. 860–891, Jan. 2019.
- [5] S. Cuomo, V. S. di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli, “Scientific Machine Learning through Physics-Informed Neural Networks: Where we are and What’s next,” June 2022. arXiv:2201.05624 [physics].
- [6] J. B. Pedro, J. Maroñas, and R. Paredes, “Solving Partial Differential Equations with Neural Networks,” Dec. 2019. arXiv:1912.04737 [physics].
- [7] C. Beck, M. Hutzenthaler, A. Jentzen, and B. Kuckuck, “An overview on deep learning-based approximation methods for partial differential equations,” Mar. 2021. arXiv:2012.12348 [cs, math].
- [8] S. Mishra, “A machine learning framework for data driven acceleration of computations of differential equations,” July 2018. arXiv:1807.09519 [cs, math].
- [9] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, “Fourier Neural Operator for Parametric Partial Differential Equations,” May 2021. arXiv:2010.08895 [cs, math].
- [10] L. Lu, P. Jin, and G. E. Karniadakis, “DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators,” *Nature Machine Intelligence*, vol. 3, pp. 218–229, Mar. 2021. arXiv:1910.03193 [cs, stat].
- [11] J. Sirignano and K. Spiliopoulos, “DGM: A deep learning algorithm for solving partial differential equations,” *Journal of Computational Physics*, vol. 375, pp. 1339–1364, Dec. 2018.
- [12] C. Beck, M. Hutzenthaler, A. Jentzen, and B. Kuckuck, “An overview on deep learning-based approximation methods for partial differential equations,” Mar. 2021. arXiv:2012.12348 [cs, math].