Comparativa de la eficiencia de los algoritmos para resolver ecuaciones diferenciales parciales tradicionales y los métodos con redes neuronales artificiales

Jose E. Ramos Martinez[1] Jorge Velázquez Castro [2]

[1]Benemerita Universidad Autonoma de Puebla [2]Facultad de Ciencias Físico-Matemáticas

RESUMEN

En este trabajo se presentan diferentes enfoques para resolver ecuaciones diferenciales parciales (EDPs), enfocándonos específicamente en la ecuación de difusividad de calor, dada su relevancia en el campo de la física. Utilizamos el proyecto FEniCS, que permite traducir rápidamente modelos científicos en código eficiente. FEniCS maneja eficientemente geometrías desde 1D hasta 3D (intervalos, cuadrados, cubos), pero para este estudio se extendió su aplicación a dimensiones más altas, utilizando productos tensoriales dentro del método de elementos finitos para resolver la ecuación de calor hasta en 6D y una dimensión temporal. Así mismo, se implementaron algoritmos basados en redes neuronales artificiales para abordar la ecuación de calor en altas dimensiones. La comparación entre estos dos enfoques se centra en la eficiencia computacional, evaluando tiempos de cómputo y precisión.

Introducción

Las ecuaciones diferenciales parciales (EDPs) son fundamentales en la modelación matemática de fenómenos físicos, químicos y biológicos. Tradicionalmente, métodos numéricos como las diferencias finitas, elementos finitos y volúmenes finitos han sido las herramientas preferidas para resolver estas ecuaciones. Sin embargo, estos métodos pueden ser computacionalmente costosos y difíciles de aplicar en problemas de alta dimensionalidad o con geometrías complejas. En los últimos años, los avances en inteligencia artificial han introducido métodos basados en redes neuronales artificiales (NNs) como una alternativa prometedora para resolver EDPs. Estos enfoques no solo aprovechan el poder del aprendizaje profundo para capturar patrones complejos y no lineales, sino que también ofrecen potenciales ventajas en términos de flexibilidad y eficiencia computacional.

FEniCs (FEM)

FEniCS es una reconocida plataforma de software de código abierto diseñada para resolver ecuaciones diferenciales parciales (EDP) mediante el método de elementos finitos (FEM). Esta herramienta facilita a los usuarios la rápida conversión de modelos científicos en código eficiente basado en elementos finitos.**FEniCs** Basandonos en el libro tutorial **Hans2017**, se puede realizar un modelado para la ecuación de calor como se muetra en la ecuacion.

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial q_i^2}, \quad in \quad \Omega \times (0,T] = \Omega_T, \quad u = u_D \quad on \quad \partial\Omega\times$$

(0,T]

$$, \quad u = u_0 \quad in \quad t = 0 \tag{1}$$

Sin embargo, llevando la ecuación a su forma variacionl nos queda la ecuación de calor en su form debil.**Hans2017**

$$a_0(u,\nu) = \int_{\Omega_T} u\nu dx \quad , \qquad L_0(\nu) = \int_{\Omega_T} u_0 \nu dx \tag{2}$$

y su función de prueba

$$u_0 = 1 + x^2 + \alpha y + \beta t \tag{3}$$

la cual ya es una ecuación que podemos ingresar a la computadora. Sin embargo existe una limitacion para FEniCs y es que solo puede resolver hasta un máximo de 3 dimensiones espaciales y una temporal. Por lo anterior, usamos el producto cartesiano para poder extender FEniCs a altas dimensiones, resolviendo la ecuación de calor.**McLoveland2022**

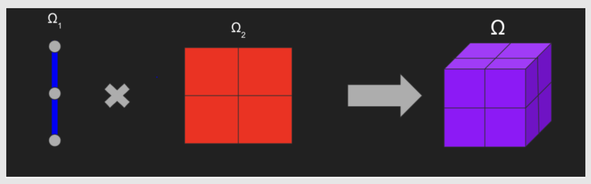


Figure 1. Ejemplo de un dominio global Ω que es el producto cartesiano de dos subdominios de dimensión menor Ω_1, Ω_2.

Haciendo una serie de cálculos pasos mencionados en [**McLoveland2022**] podemos convertir la ecuación (1) a su forma debil, multiplicando por una funcion de prueba ν y integrando sobre todo Ω_T nos da la forma debil.

$$\int_{\Omega_T} -u\frac{\partial \nu}{\partial t} \quad + \quad \nabla u \cdot \nabla \nu dq_s dt \quad + \quad \int_{\partial\Omega_T} u\nu \cdot \mathbf{n} dq_i dt = \int_{\Omega_T} f\nu \quad dq_s dt \tag{4}$$

Déspues hacemos una aproximacion por producto base a (4) dandonos la forma (6). Se muestra la ecuación para dos subdominios espaciales y uno temporal $\Omega_1, \Omega_2, \Omega_3$

$$u = \sum_{i=j=k=0}^{N,M,L} u_{i,j,k}\Phi_i\Psi_j\gamma_k \quad , \qquad \nu = \sum_{l=p=q=0}^{N,M,L} \nu_{l,p,q}\alpha_l\beta_p\Theta_q \tag{5}$$

$$\int_{\Omega_1} \nabla_1(\Phi_i)\cdot\nabla_1(\alpha_l)dq_1 \int_{\Omega_2} \Psi_j\beta_p dq_2 \int_{\Omega_3} \gamma_k\Theta_q dt \quad + \quad \int_{\Omega_2} \nabla_2(\Psi_j)\cdot\nabla_2(\beta_p)dq_2 \int_{\Omega_1} \Phi_i\alpha_l dq_1 \int_{\Omega_3} \gamma_k\Theta_q dt$$
$$+ \int_{\Omega_3} \nabla_3(\gamma_k)\cdot\nabla_3(\Theta_q)dt \int_{\Omega_1} \Phi_i\alpha_l dq_1 \int_{\Omega_2} \Psi_j\beta_p dq_2 \quad - \quad \int_{\Omega_3} \gamma_k\frac{\partial\Theta_q}{\partial t}dt \quad + \quad \int_{\partial\Omega_3} \gamma_k\Theta_q\cdot\mathbf{n}dSdt(u)$$
$$= \int_{\Omega_1} \Phi_i\alpha_l dq_1 \int_{\Omega_2} \Psi_j\beta_p dq_2 \int_{\Omega_3} \gamma_k\Theta_q dt\, f_{i,j,k} \tag{6}$$

Dónde reescribiendo la ecuación como matrices de Kronecker,e.g., stiffness matrices computadas con FEniCs [**McLoveland2022**],una vez haciendo esto ya se puede resolver computacional-mente.

Método con redes neuronales (ANN)

Las PINNs son una clase de algoritmos de aprendizaje profundo diseñados para resolver ecuaciones diferenciales parciales (PDE)integrando las leyes físicas subyacentes directamente en el proceso de entrenamiento de la red

.

[**raissi2019physics**]

- Red Neuronal: Utiliza un modelo Sequential con varias capas densas (Dense) y funciones de activación tanh y linear.
- Cálculo de derivadas mediante GradientTape: se utiliza para calcular las derivadas de la salida de la red neuronal con respecto a las entradas
- Ecuación Diferencial: Modelamos una PDE del tipo difusión o calor.

$$u_t = u_{xx} + u_{yy} \tag{7}$$

- Optimización: La red se entrena minimizando la función de pérdida usando el optimizador Adam, ajustando los pesos de la red neuronal para que la solución que produce satisfaga la PDE.

Resultados

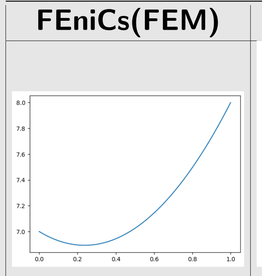
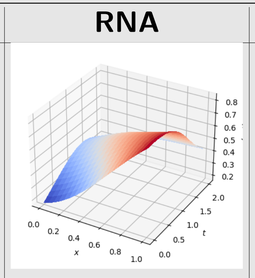
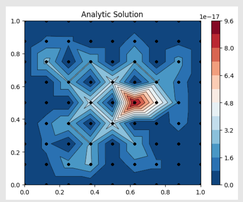
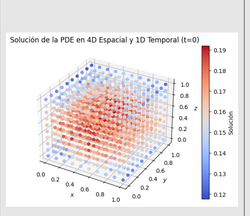
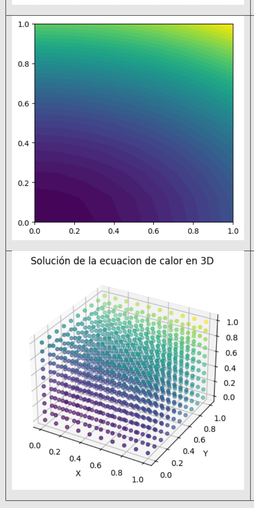
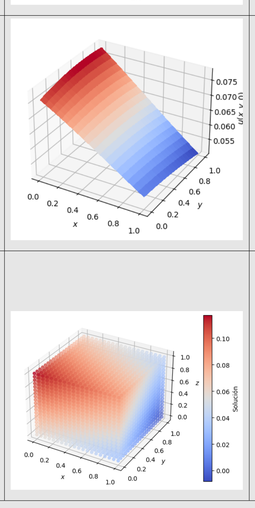
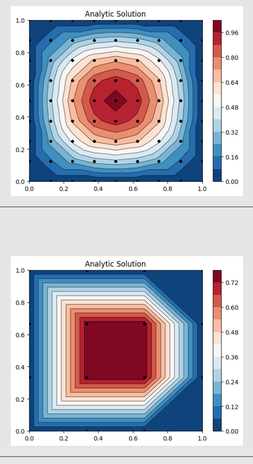
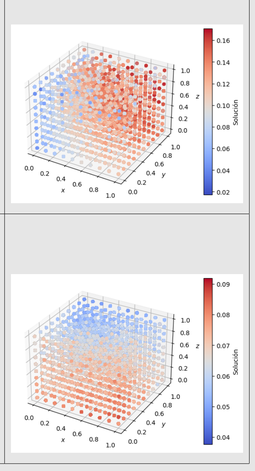
FEniCs(FEM)	RNA	Altas dimensiones(FEniCs)	RNA
			
			

Table 1. Se muestran los resultados obetenidos para ecuación de calor tanto en FEniCs como en RNA.

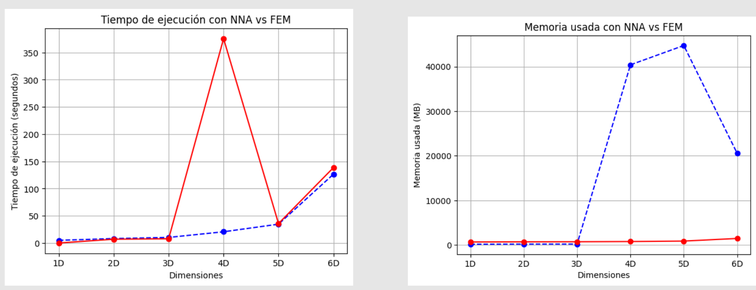


Figure 2. Evaluación de tiempos de computó y de memoria usada en la resolución de EDP.

Conclusiones

Finalmente, hemos observado que el uso de redes neuronales para resolver ecuaciones diferenciales parciales, específicamente la ecuación de calor, resulta más eficiente tanto en términos de tiempo de cálculo como en la reducción del uso de memoria. Esta eficiencia representa un avance significativo para la comunidad científica, ya que permite la resolución de ecuaciones complejas incluso con equipos de menor capacidad computacional.

Referencias

- FEniCs Project. (30/08/2024). FEniCs Project. Obtenido de FEniCs Project: https://fenicsproject.org
- Hans Petter Langtangen*, A. L. (2017). Solving PDEs in Python– The FEniCS Tutorial Volume I. Springer.
- Mark Loveland, E. V. (2022). Extending FEniCS to Work in Higher Dimensions Using Tensor Product Finite Elements. ELSEVIER, Obtenido de https://www.sciencedirect.com/science/article/abs/pii/S1877750322001922
- Raissi, M. a. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 686–707.