PINN | Informe

Programación científica

Tomas Escobar Rivera
tescobarr@unal.edu.co

Juan Jose Garcia Arias
jgarciaari@unal.edu.co

Manuel Santiago Martinez Hurtado
mmartinezhu@unal.edu.co

Andrea Sánchez Jimenez
asanchezji@unal.edu.co

Santiago Uribe Echavarria
sauribee@unal.edu.co

Universidad Nacional de Colombia

Facultad de Ciencias

Sede Medellín

07 de marzo de 2025

Tabla de contenidos

1 Introducción 1

2 Marco teórico 1

2.1 Redes neuronales artificiales 1

2.2 Redes neuronales informadas por la física (PINN) 1

2.3 Métodos implementados en PINN 1

2.4 Error relativo 2

3 Planteamiento del problema 2

4 Objetivos 2

4.1 Objetivo general 2

4.2 Objetivos específicos 3

5 Metodología 3

5.1 Herramientas y tecnologías 3

5.2 Experimentación y optimización 3

5.3 Evaluación del error 4

6 Implementación de los planteamientos 4

6.1 Planteamiento 1 4

6.2 Planteamiento 2 4

7 Conclusiones y resultados 5

7.1 Resultados de la experimentación 5

7.2 Análisis de resultados 6

7.3 Conclusiones 7

1 Introducción

Las ecuaciones diferenciales son fundamentales en la modelación de sistemas físicos, pero su resolución mediante métodos numéricos tradicionales puede ser costosa y compleja. Las Redes Neuronales Informadas por la Física han emergido como una alternativa innovadora, integrando el conocimiento físico directamente en el proceso de aprendizaje de la red.

Este proyecto compara dos planteamientos distintos para el diseño y entrenamiento de PINNs, con el objetivo de determinar cuál ofrece mejor rendimiento en términos de precisión y eficiencia computacional.

2 Marco teórico

2.1 Redes neuronales artificiales

Las redes neuronales artificiales son modelos computacionales inspirados en la estructura y funcionamiento del cerebro humano. Están compuestas por unidades llamadas neuronas artificiales, organizadas en capas que procesan la información de manera jerárquica. Su entrenamiento se basa en la propagación de información y el ajuste de pesos mediante algoritmos de optimización.

2.2 Redes neuronales informadas por la física (PINN)

Las PINN son una variante de redes neuronales diseñadas para resolver ecuaciones diferenciales parciales. A diferencia de las redes tradicionales, integran información física en su función de pérdida para garantizar que las soluciones obtenidas cumplan con las ecuaciones diferenciales y sus condiciones de frontera.

El entrenamiento de una PINN implica definir una ecuación diferencial como parte de la función de pérdida, evaluar la red en múltiples puntos del dominio y optimizar los parámetros para minimizar el error en la predicción.

2.3 Métodos implementados en PINN

En la implementación de las PINN, existen dos enfoques comunes para integrar ecuaciones diferenciales y condiciones de frontera en el proceso de entrenamiento:

1. **Incorporación directa de la ecuación diferencial**: Se integra la ecuación diferencial y las condiciones de frontera en la función de pérdida. La red neuronal se entrena minimizando esta función.

$$\mathcal{L}(u_{NN}) = \frac{1}{N}\sum_{i=1}^{N}\left[\lambda_0\left(-\Delta u_{NN}(x_i) + \alpha u_{NN}(x_i) - f(x_i)\right)^2 + \lambda_1\left(u_{NN}(0) - g(0)\right)^2 + \lambda_2\left(u_{NN}(\pi) - g(\pi)\right)^2\right]$$

   En donde λ_0, λ_1 y λ_2 son parámetros de ajuste.

2. **Reformulación de la solución**: La solución se reformula para satisfacer automáticamente las condiciones de frontera, permitiendo que la red solo aprenda una corrección sobre una solución base que ya satisface las condiciones de frontera. Particularmente, la solución se define como:

$$u_{NN} = NN(x) \cdot x \cdot (x - \pi)$$

   Con función de pérdida definida dada por:

$$\mathcal{L}(u_{NN}) = \frac{1}{N}\sum_{i=1}^{N}\left[-\Delta u_{NN}(x_i) + \alpha u_{NN}(x_i) - f(x_i)\right]^2$$

2.4 Error relativo

El error relativo es una métrica que mide la diferencia entre una solución aproximada y la solución exacta en relación con la magnitud de la solución exacta. Se utiliza para evaluar la precisión de modelos numéricos y su desempeño en la aproximación de soluciones a problemas matemáticos.

Matemáticamente, el error relativo se define como:

$$\text{Error Relativo} = \frac{\|\text{Solución Exacta} - \text{Solución Aproximada}\|}{\|\text{Solución Exacta}\|}$$

Donde:

- $\|\text{Solución Exacta} - \text{Solución Aproximada}\|$ representa la diferencia entre ambas soluciones.
- $\|\text{Solución Exacta}\|$ es la norma de la solución exacta utilizada como referencia.

El error relativo es útil cuando se comparan soluciones en diferentes escalas, ya que permite evaluar la precisión de una solución aproximada sin verse afectado por el tamaño absoluto de la solución exacta.

3 Planteamiento del problema

Los métodos tradicionales para resolver ecuaciones diferenciales requieren discretización y altos recursos computacionales. Las PINN han surgido como una alternativa viable, integrando las ecuaciones diferenciales y su física subyacente en el proceso de entrenamiento de redes neuronales. Sin embargo, existen múltiples estrategias para implementar las PINN, y la elección del enfoque adecuado puede impactar significativamente la precisión y eficiencia computacional de los modelos.

En este estudio, se comparan dos metodologías para entrenar PINN, basadas en los siguientes enfoques:

1. **Incorporación directa de la ecuación diferencial en la función de pérdida**.
2. **Reformulación de la solución para satisfacer automáticamente las condiciones de frontera**.

La pregunta clave que este estudio busca responder es:

¿Cuál de estos dos enfoques proporciona una mejor aproximación en términos de precisión y eficiencia computacional al resolver el siguiente problema de valores de frontera?

$$\begin{cases} -\Delta u + \alpha u = f, & x \in (0, \pi) \\ u = g, & x \in \{0, \pi\} \end{cases}$$

donde $u(x)$ es la solución, Δ es el operador Laplaciano, α es un parámetro, $f(x)$ es una función fuente y $g(x)$ son las condiciones de frontera.

4 Objetivos

4.1 Objetivo general

Comparar dos planteamientos distintos para entrenar redes neuronales informadas por la física y evaluar su desempeño en la resolución del problema con valores de frontera propuesto con la finalidad de determinar cuál enfoque ofrece mejor precisión y eficiencia computacional.

4.2 Objetivos específicos

- Implementar dos arquitecturas diferentes de PINNs.
- Resolver un conjunto de ecuaciones diferenciales mediante cada planteamiento.
- Evaluar la precisión de los resultados mediante el error cuadrático medio (L^2 error).
- Analizar el costo computacional de cada enfoque.
- Determinar cuál estrategia es más efectiva en términos de precisión y eficiencia computacional.

5 Metodología

5.1 Herramientas y tecnologías

Para la implementación de los modelos se utilizaron las siguientes herramientas:

- **Lenguaje de programación**: Python
- **Bibliotecas**: TensorFlow y Keras para la construcción y entrenamiento de las redes neuronales, NumPy para operaciones matemáticas y manipulación de datos
- **Visualización**: Matplotlib para graficar los resultados y compararlos con la solución analítica

5.2 Experimentación y optimización

Para analizar el impacto de los hiperparámetros en el desempeño de las redes neuronales PINN, se realizaron 45 experimentos utilizando una estrategia de búsqueda aleatoria (*random search*).

Los siguientes hiperparámetros fueron seleccionados para su variación:

- **Número de neuronas por capa**: entre 4 y 10.
- **Número de capas ocultas**: entre 4 y 5.
- **Número de puntos de muestreo**: entre 500 y 3000.
- **Pesos de penalización** para la ecuación diferencial (λ_0) y las condiciones de contorno (λ_1, λ_2): entre 1 y 10.

Además, se realizó un experimento con una configuración base que empleaba los siguientes valores predeterminados: Cada configuración se entrenó durante 400 épocas utilizando el optimizador Adam con una tasa de

Hiperparámetro	**Valor**
Número de neuronas	10
Número de capas	5
Puntos de muestreo	2000
λ_0	5
λ_1	7
λ_2	7

aprendizaje fija de 1×10^{-3}. Los resultados de los experimentos fueron evaluados en términos del error relativo y el costo computacional, medido como el porcentaje de uso de CPU durante la ejecución. Esta evaluación permite comparar la precisión y la eficiencia computacional de cada configuración de hiperparámetros. Finalmente Se seleccionó la arquitectura con mejor precisión y menor costo computacional.

5.3 Evaluación del error

Se utilizó el error H^1 como métrica principal para evaluar la precisión de los modelos, calculado como sigue:

$$\text{Error Relativo} = \frac{\frac{1}{N}\sum(u'_{NN}(x) - u'(x))^2}{\frac{1}{N}\sum(u'(x))^2}$$

en donde $u'_{NN}(x)$ es la derivada aproximada por la red neuronal, $u'(x)$ es la derivada exacta.

6 Implementación de los planteamientos

6.1 Planteamiento 1

Se utiliza la siguiente configuración inicial para la implementación de la red neuronal:

- Se establecen los parámetros clave:
  - Número de neuronas y capas en la red
  - Cantidad de puntos de muestreo (`nPts`)
  - Iteraciones de entrenamiento (`iterations`)
  - Factores de penalización (λ_0, λ_1, λ_2)

Luego, se construye la red neuronal de la siguiente manera:

- `makeModel1(neurons, nLayers, activation)`:
  - Crea la red neuronal `uModel1` con capas densas y activación `tanh`
- `Loss1`:
  - Define la función de pérdida, incorporando la ecuación diferencial y condiciones de frontera
  - Usa diferenciación automática (`tf.GradientTape`) para calcular derivadas

6.2 Planteamiento 2

Se implementa la reformulación de la solución para satisfacer automáticamente las condiciones de frontera. La red neuronal se entrena para aprender una corrección sobre una solución base que ya cumple con las condiciones de frontera, es decir:

$$u_{NN}(x) = NN(x) \cdot x \cdot (x - \pi)$$

La función de pérdida satisface lo siguiente:

- No incluye términos de penalización de frontera
- Se optimiza usando el optimizador Adam
- Se implementa `RelativeErrorCallback` para monitoreo

7 Conclusiones y resultados

7.1 Resultados de la experimentación

Se realizaron 45 experimentos para evaluar el impacto de los hiperparámetros en la precisión y eficiencia computacional de las redes neuronales PINN. Los gráficas obtenidas se muestran a continuación:

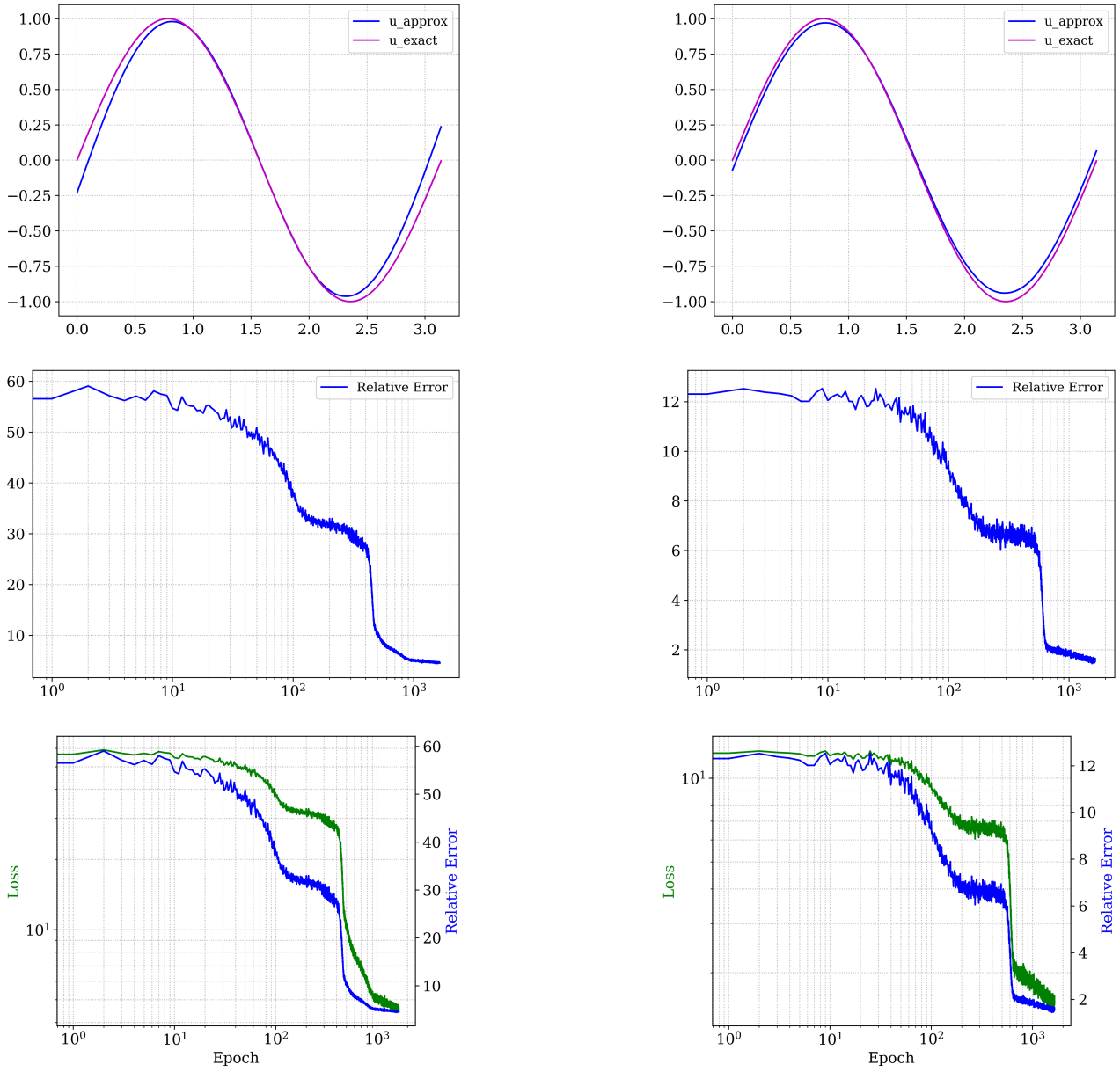

Figure 1: Resultados del planteamiento 1

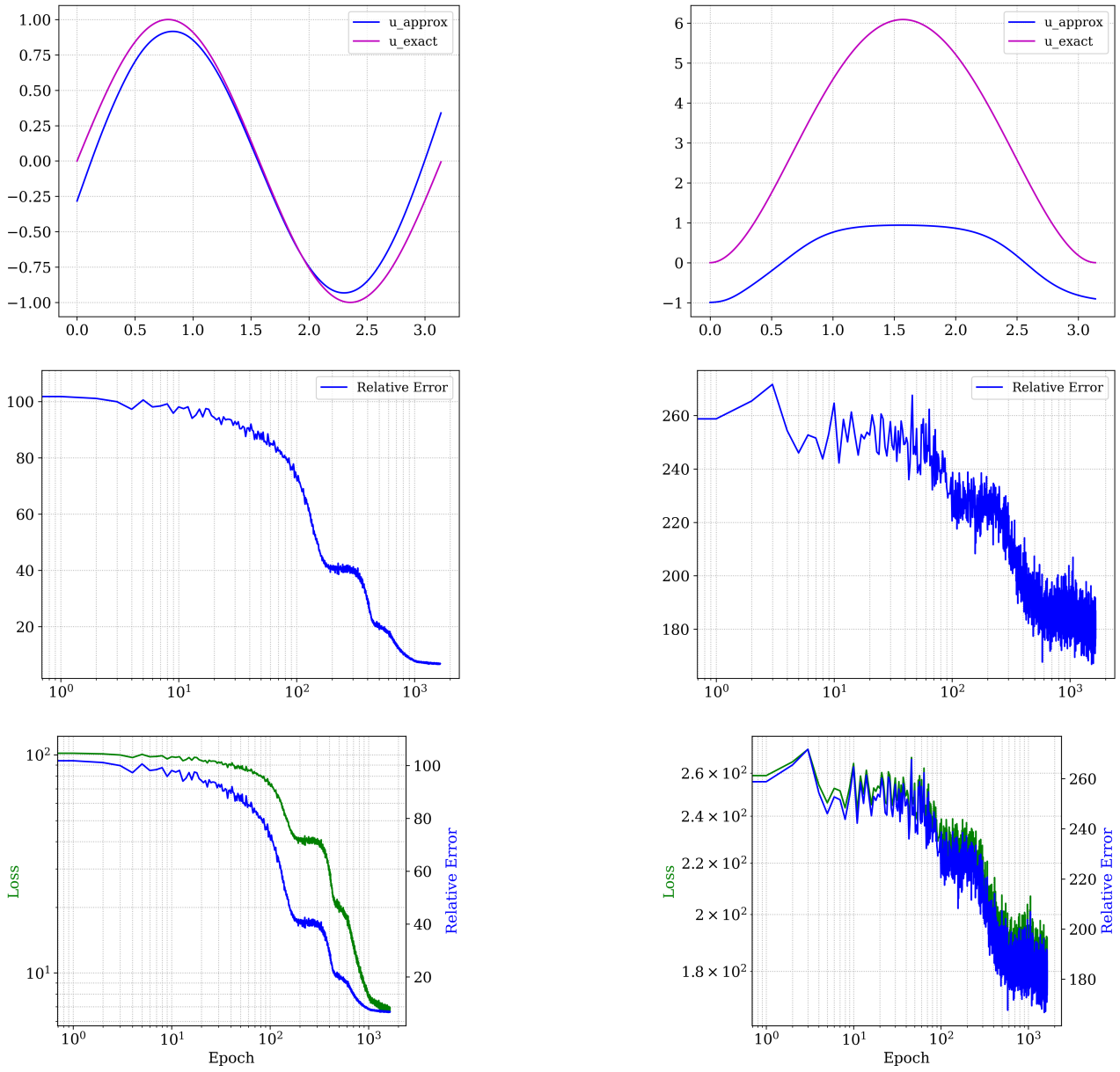

Figure 2: Resultados del planteamiento 2

7.2 Análisis de resultados

Los resultados obtenidos muestran que el planteamiento 1, basado en la asignación directa de la ecuación diferencial en la función de pérdida, presenta una mayor precisión en la solución de la ecuación diferencial en los dos casos analizados. Sin embargo, el planteamiento 2, que reformula la solución para satisfacer automáticamente las condiciones de frontera, ofrece una mayor eficiencia computacional y un menor costo de entrenamiento.

La reformulación de la solución, en el segundo planteamientos permite que la red neuronal aprenda una corrección sobre una solución base que ya satisface las condiciones de frontera, lo que simplifica el proceso de

entrenamiento y mejora la precisión de los resultados. Por otro lado, la incorporación directa de la ecuación diferencial en la función de pérdida puede resultar en una mayor complejidad y requerir un mayor número de iteraciones para alcanzar una solución precisa.

7.3 Conclusiones

En conclusión, la elección del enfoque para el diseño y entrenamiento de redes neuronales informadas por la física depende de los objetivos específicos del problema y las restricciones computacionales. Si se prioriza la precisión en la solución de ecuaciones diferenciales, el planteamiento 1 puede ser más adecuado. Por otro lado, si se busca una mayor eficiencia computacional y un menor costo de entrenamiento, el planteamiento 2 puede brindar resultados satisfactorios. En general, ambos enfoques ofrecen una alternativa innovadora y eficaz para la resolución de problemas de valores de frontera mediante redes neuronales.