Universidad de La Habana
Facultad de Matemática y Computación

Solución del modelo Perona-Malik usando Redes Neuronales

Autor:
Leandro Rodríguez Llosa

Tutores:

Dr. José Alejandro Mesejo Chiong
MSc. Fernando Rodríguez Flores

Trabajo de Diploma
presentado en opción al título de
Licenciados en Ciencia de la Computación

13 de enero de 2024

A mi madre,
por ser mi mayor motivación.

A mi padre,
por entregarme lo mejor de él.

A mi abuela Tati,
por el amor desmesurado que me entrega siempre.

A mis otros abuelos,
que los llevaré en mi corazón el resto de mi vida.

A mi hermano,
por todo lo que nos falta por crecer y construir.

A mi familia del futuro,
la que se construye en la vida.

Agradecimientos

A Ramonita, por ser mi eterna maestra. Hoy cuento 17 años desde aquel día en primer grado de la primaria que se me ocurrió prenderle fuego al cesto de la basura.

A Tere, por la confianza y el apoyo que siempre ha depositado en mí.

A mi Cupidita, por ser una persona incondicional, donde la palabra incondicional cabe ser mencionada.

A Clau, por estar ahí; a veces más cerca, otras más lejos, pero está.

A Carlito, por convertirse en mi amigo.

A Felix, por ser mi paño de lágrimas, la baranda de la que me sujeto antes de caer al piso, el brazo que me ayuda a levantar las veces que caigo.

A Marquito, por ser mi compañero de madrugadas en estos últimos 4 años, y más que eso, mi amigo.

Al profe Piad, por su carisma, las oportunidades que me ha dado, y todo el conocimiento que me enseñado.

Al profe Fernan, por darle color a un mundo en blanco y negro, y enseñarnos a ver de ese modo.

Al profe Mesejo, por acompañarme en esta ~~locura~~ aventura que tuvo como nombre "Solución del modelo Perona-Malik usando Redes Neuronales".

A Alain, por ser mi principal mentor durante la carrera, y depositarme la confianza que un día no tenía y necesitaba recuperar.

A Erne, por el cariño y el apoyo que siempre me regaló.

Al Morgan, por trasmitirme tanta pasión por la carrera.

Opinión del tutor

Los modelos de la Inteligencia Artificial han irrumpido en la ciencia moderna de forma determinante. En el campo de la solución numérica de ecuaciones diferenciales, tanto ordinarias como en derivadas parciales, se publican diariamente numerosos artículos. Para ilustrar esta afirmación tenemos el hecho de que una búsqueda en Google Scholar con el término "neural network pde solution" limitada a trabajos de este año 2024 arroja ya 649 resultados. Precisamente en este campo se inscribe el trabajo de diploma del estudiante Leandro Rodríguez Llosa, en la solución del modelo de difusión anisotrópica de Perona y Malik mediante redes neuronales informadas por la física (PINNs por sus siglas en inglés).

Para acometer este trabajo Leandro se tuvo que adentrar en el tema de las ecuaciones en derivadas parciales parabólicas el cual no aparece ni remotamente en el currículo de la Carrera de Ciencia de la Computación de nuestra Facultad. Después de lograr entender aspectos esenciales de las ecuaciones en derivadas parciales Leandro estudió a fondo las principales publicaciones sobre las redes neuronales informadas por la física. Ambos temas son desafiantes y Leandro logró dominarlos con la debida rigurosidad e independencia investigativa que se espera de un egresado de nuestra Facultad.

En el desarrollo del trabajo Leandro identificó problemas fundamentales que reducen la aplicabilidad de las PINNs al modelo de difusión anisotrópica y analizó una nueva vía de solución. Esta radica en la aplicación de la propiedad de aproximación de las redes neuronales de cualquier operador no lineal tal como se implementa en DeepONet. Lamentablemente debido a limitaciones de recursos de tiempo y poder de cómputo este último proceder no se pudo explorar en la extensión debida, aunque se obtuvieron resultados preliminares prometedores.

En resumen Leandro Rodríguez Llosa abordó este Trabajo de Diploma con independencia y creatividad investigativa a pesar de las dificultades inherentes del tema. Demostró la capacidad de aplicar los conocimientos adquiridos en los años de estudio de forma apropiada para la creación de otros nuevos. Es mi convicción que esto lo hace merecedor del título de Licenciado en Ciencia de la Computación que concede nuestra Facultad con la máxima calificación.

Dr. José Alejandro Mesejo Chiong,

La Habana, 3 de enero de 2024

Resumen

Esta tesis explora la aplicación de Redes Neuronales Informadas por la Física y Redes Profundas de Operadores en la solución de la ecuación diferencial de Perona-Malik y de Difusión del Calor. El trabajo se enmarca en la intersección de ecuaciones diferenciales parciales y aprendizaje automático, específicamente en el aprendizaje profundo, para abordar desafíos matemáticos complejos.

El estudio se centra en integrar principios físicos en el aprendizaje de las redes neuronales, aplicándolos al modelo de Perona-Malik que propone una difusión adaptativa para la detección y realce de bordes en imágenes. A través de experimentos, se evalúan las capacidades y limitaciones de las Redes Neuronales Informadas por la Física y Redes Profundas de Operadores en la modelización de esta ecuación y de la ecuación de Difusión del Calor en una dimensión. Se identifican desafíos clave relacionados con la complejidad de las condiciones iniciales y la precisión del modelo.

Los resultados evidencian que, aunque las Redes Neuronales Informadas por la Física muestran potencial, enfrentan limitaciones para adaptarse a condiciones iniciales complejas. En contraste, las Redes Profundas de Operadores ofrece mayor flexibilidad y generalización, aunque con desafíos en la estimación de condiciones iniciales también. Este trabajo aporta una visión crítica sobre la aplicación de técnicas de aprendizaje profundo en ecuaciones diferenciales, destaca la importancia de un muestreo representativo y un equilibrio en la ponderación de errores para mejorar la precisión y eficacia en la solución de ecuaciones diferenciales.

Palabras clave: Redes Neuronales Informadas por la Física, Redes Profundas de Operadores, Ecuación de Perona-Malik, Ecuación del Difusión del Calor, Aprendizaje Profundo

Abstract

This thesis explores the application of Physics Informed Neural Networks and Deep Operator Networks in the solution of the Perona-Malik differential equation, an essential model in image processing. The work is framed at the intersection of partial differential equations and machine learning, specifically deep learning, to address complex mathematical challenges.

The study focuses on integrating physical principles in the learning of neural networks, applying them to the Perona-Malik model that proposes adaptive diffusion for the detection and enhancement of edges in images. Through experiments, the capabilities and limitations of Physics Informed Neural Networks and Deep Operator Networks in modeling this equation and the heat diffusion equation in one dimension are evaluated. Key challenges related to the complexity of initial conditions and model accuracy are identified.

The results show that, although Physics Informed Neural Networks show potential, they face limitations in adapting to complex initial conditions. In contrast, Deep Operator Networks offers greater flexibility and generalization, although with challenges in estimating initial conditions as well. This work provides a critical view on the application of Deep Learning techniques in Partial Differential Equations, highlighting the importance of representative sampling and a balance in the weighting of errors to improve the precision and effectiveness in solving Partial Differential Equations.

Key words: Physics Informed Neural Networks, Deep Operator Networks, Perona-Malik, Heat Equation, Deep Learning

Índice general

Introducción

En esta investigación, se aborda un terreno donde las ecuaciones diferenciales parciales (PDEs) y las Redes Neuronales (NNs) se entrecruzan para ofrecer soluciones a desafíos matemáticos complejos. Las PDEs son herramientas matemáticas fundamentales para modelar una amplia gama de problemas en diferentes campos entre los que se destacan: la física, la biología y la ingeniería. La complejidad inherente de las PDEs y la dificultad para hallar soluciones analíticas precisas en muchos casos conlleva a buscar alternativas.

Tradicionalmente, la solución a estos problemas complejos se ha realizado mediante métodos numéricos, que, a pesar de ser efectivos, suelen incurrir en un alto costo computacional y en la necesidad de discretizar el dominio del problema. En este contexto Raissi et al. [1, 2], demuestran que las NNs emergen como una solución prometedora, pues ofrecen la posibilidad de aproximar la solución de las PDEs de manera más eficiente.

Durante la última media década, a raíz de la propuesta de estos autores, el campo de las matemáticas computacionales ha experimentado un progreso significativo en la utilización de NNs para resolver PDEs, un reto notable en esta área de estudio. Uno de los desarrollos más destacados es la creación y evolución de las Redes Neuronales Informadas por la Física (PINNs) [3].

Estas redes, integran principios físicos esenciales en el proceso de aprendizaje, lo que resulta un método más eficaz y preciso para abordar las PDEs. El impacto y la relevancia de las PINNs se ha consolidado con rapidez, reflejado en la creciente atención y adaptación de los trabajos pioneros de Raissi et al. [1, 2, 3]. Esta metodología, que fusiona de manera innovadora las PDEs con el aprendizaje basado en datos, ha demostrado ser útil en la resolución de PDEs no lineales complejas como las ecuaciones de Schrödinger, Burgers y Allen-Cahn.

En paralelo, se han desarrollado otros enfoques: el Método de Ritz Profundo (DRM) [4] y el Método Galerkin Profundo (DGM) [5]. Estos métodos han enriquecido el repertorio de herramientas disponibles para abordar las PDEs con NNs y ofrecen soluciones únicas a diversos tipos de ecuaciones. Por ejemplo, el DRM introduce una perspectiva novedosa al definir la pérdida en términos de la energía de la solución del problema.

En este contexto surgen las Redes Profundas de Operadores (DeepONets) [6], que a diferencia de las PINNs no integran las leyes físicas en el aprendizaje. Las DeepONets se diseñan para aprender operadores funcionales de un conjunto diversificado de datos, abren así un campo de posibilidades que incluye problemas con datos heterogéneos y complejos. Más adelante surgen las Redes Profundas de Operadores Informados por la Física (PIDeepONet) en [7], que combinan la capacidad de las PINNs de realizar un aprendizaje no supervisado, con las DeepONets, una arquitectura que permite generalizar la solución de una PDE para un conjunto de condiciones iniciales. La capacidad para representar los operadores lineales y no lineales las coloca en una posición única dentro de las herramientas de aprendizaje automático aplicadas a las matemáticas computacionales, complementa y expande el horizonte de las metodologías existentes.

A pesar de estos avances, el campo no ha estado exento de desafíos. Krishnapriyan et al. [8]

arrojaron luz sobre posibles fallos en las PINNs al tratar con fenómenos físicos de alta complejidad. Ellos identificaron que ciertos tipos de regularización en las PINNs podrían introducir complicaciones sutiles en el proceso de optimización. Este hallazgo resalta la necesidad de continuar el desarrollo y refinado de estos modelos para encontrar problemas aún más complejos y detallados.

La fusión del aprendizaje automático con la computación científica redefine las PDEs desde un punto de vista computacional. Este cambio se ve reflejado en la investigación de Cuomo et al. [9], que proporciona una visión profunda de los avances y direcciones futuras en el aprendizaje automático científico, y pone especial énfasis en las PINNs. El análisis no solo destaca el estado actual del campo, sino que anticipa trayectorias emocionantes de investigación y desarrollo futuros.

La evolución de las técnicas para resolver PDEs mediante NNs en los últimos cinco años ha sido notable. Se ha logrado una integración armoniosa de las matemáticas computacionales clásicas con las técnicas modernas de aprendizaje automático, y se ha creado una sinergia entre dos campos que antes parecían dispares. Esta convergencia promete abrir nuevas posibilidades y aplicaciones en áreas críticas en la física, la ingeniería y las finanzas, donde las PDEs son fundamentales.

Las PDEs de alta dimensión son vitales en la modelización de fenómenos, pero la complejidad y no linealidad plantean desafíos significativos. Las PINNs son una solución revolucionaria en el campo de la computación científica, que brindan un enfoque novedoso para afrontar las PDEs sin la necesidad de mallas. Esto representa un gran avance en términos de eficiencia y versatilidad. Asimismo, en el extenso estudio de Cuomo et al. [9] también se resaltan los múltiples beneficios que aportan las PINNs; la capacidad de calcular soluciones bajo demanda después de un proceso de entrenamiento y la flexibilidad de manejar problemas directos e inversos dentro del mismo marco de optimización. Estas características subrayan la importancia de las NNs en la aproximación de soluciones a PDEs.

No obstante, las PINNs no están exentas de desafíos. Uno de los más destacados es el tiempo requerido para el entrenamiento de la red, considerado mayor en comparación con los métodos numéricos. Otro significativo es la capacidad de estas redes para aprender y modelar fenómenos físicos complejos como aquellos con comportamientos multiescala, caóticos o turbulentos.

A pesar de estos obstáculos, el uso de NNs en la búsqueda de soluciones a PDEs es un área de investigación activa y llena de potencial. Esta línea de estudio no solo promete transformar la manera en que se trabajan problemas complejos en ciencia e ingeniería, sino también abre la puerta a innovaciones y aplicaciones que podrían redefinir los límites de lo que es posible en estos campos. Tras considerar las ventajas y los retos inherentes a las PINNs; el tiempo de entrenamiento y la habilidad para modelar fenómenos complejos, surge la línea de investigación de evaluar la eficacia de las NNs para la aproximación de soluciones del modelo de Perona-Malik.

El modelo de Perona-Malik fue desarrollado en 1990 por Pietro Perona y Jitendra Malik [10], fundamental en el procesamiento de imágenes por crear la técnica de la difusión anisotrópica. Ofrece un escenario único para investigar la aplicabilidad y eficiencia de las NNs en las ecuaciones diferenciales. La premisa central del modelo de Perona-Malik es que la difusión debe ser adaptativa, es decir, más intensa en áreas de la imagen con baja variación de intensidad o contraste, y menos intensa en las regiones donde existen bordes o cambios bruscos de intensidad. Para lograr esta adaptabilidad, el modelo introduce un coeficiente de difusión local, que ajusta la velocidad de difusión en cada punto de la imagen en función de las características.

La aplicación del modelo de Perona-Malik mejora la detección de bordes en imágenes. La ecuación diferencial está diseñada para modelar cómo cambia la intensidad de cada pixel a lo largo del tiempo, considerando el entorno inmediato. A medida que el tiempo avanza, las áreas de la imagen con variaciones menores de intensidad se suavizan de manera gradual, mientras que las regiones con cambios drásticos de intensidad se mantienen o incluso se acentúan. Este proceso conduce a un realce de los bordes, hace que se destaquen con claridad. Así, el modelo no solo mejora la calidad

visual de las imágenes al reducir el ruido, sino que facilita la identificación y el análisis de estructuras y bordes importantes, lo cual es esencial en numerosas aplicaciones, desde la medicina hasta la visión artificial.

El presente trabajo se ha centrado en la comprensión de las NNs para la solución de ecuaciones diferenciales parciales, con especial atención en dos metodologías avanzadas: las PINNs y las DeepONet [1, 2, 6]. Además, se ha dedicado tiempo a entender y aplicar el método numérico de diferencias finitas, esencial para la solución numérica de PDE. Este conocimiento ha sido concluyente para comparar los resultados obtenidos mediante técnicas basadas en NNs con aquellos derivados de métodos numéricos más tradicionales.

Esta comparación ha permitido evaluar no solo la eficacia de las NNs en este campo, sino también la precisión y viabilidad en diversos contextos de aplicación. El impacto de la investigación se asocia a una perspectiva científica, los resultados podrían esclarecer cómo las PINNs se comportan en relación a condiciones iniciales complejas, como es el caso de las imágenes.

En el marco de la presente investigación, se identifica como **problema científico**:

> ¿Cómo hallar la solución de ecuaciones diferenciales en derivadas parciales, en especial del modelo de difusión anisotrópica de Perona-Malik, mediante NNs ?

El **objetivo general** es aplicar las técnicas de Aprendizaje Profundo (DL) para la aproximación de soluciones numéricas de ecuaciones diferenciales en derivadas parciales con énfasis en el modelo de Perona-Malik.

Objetivos específicos

- Sistematizar el marco teórico que sustenta las técnicas de Aprendizaje Profundo (DL) para la aproximación de la solución de ecuaciones diferenciales.
- Elaborar modelos preentrenados, flexibles y eficientes, para adaptarse a variaciones en los parámetros de la ecuación diferencial de Perona-Malik, y la ecuación de Difusión del Calor.
- Valorar los resultados de la aplicación de los técnicas utilizadas: las PINNs y las DeepONet.

Preguntas científicas:

1. ¿Cuál es el marco teórico que sustenta las técnicas de Aprendizaje Profundo (DL) para la aproximación de la solución de ecuaciones diferenciales?
2. ¿Cómo se estructuran los modelos preentrenados, flexibles y eficientes, para adaptarse a variaciones en los parámetros de la ecuación diferencial de Perona-Malik y la ecuación de Difusión del Calor?
3. ¿Cómo valorar los resultados de la aplicación de las técnicas utilizadas: las PINNs y las DeepONet?

La tesis se estructura en: introducción; capítulo I donde se sistematizan de las técnicas de Aprendizaje Profundo para la aproximación a la solución de ecuaciones diferenciales; el capítulo II dirigido a solución de la ecuación diferencial de Perona-Malik y la ecuación de Difusión del Calor; el capítulo III donde se valoran los resultados de la aplicación de las técnicas utilizadas: las PINNs y las DeepONet; además de las conclusiones, recomendaciones y bibliografías.

Capítulo 1

Marco Teórico

Este capítulo se dedica a una exploración exhaustiva en el campo de las ecuaciones diferenciales (EDs) y las metodologías variadas que se han desarrollado para su resolución. Dada la importancia en campos científicos y de ingeniería, las EDs se presentan en formas y complejidades diversas, desde modelos matemáticos simplificados hasta ecuaciones que describen fenómenos físicos de gran intricación. El capítulo está estructurado en cuatro epígrafes clave: el primero proporciona un contexto del estado del arte, aborda los métodos analíticos y numéricos tradicionales. El segundo epígrafe se enfoca en profundizar la teoría de una técnica emergente en la investigación: las PINNs, destaca la integración de principios físicos con técnicas de inteligencia artificial. El tercer epígrafe se centra en otra técnica avanzada, las DeepONet, que representan un enfoque revolucionario en el aprendizaje de operadores no lineales a partir de datos. El cuarto epígrafe aborda los softwares existentes para trabajar con las PINNs, resalta las herramientas y plataformas clave que facilitan y potencian la investigación en este ámbito.

1.1. Estado del Arte

Este epígrafe inicial se sumerge en la amplia y diversificada área de las estrategias para resolver ecuaciones diferenciales, abarca desde los tradicionales métodos analíticos y numéricos hasta las contemporáneas técnicas de Aprendizaje de Máquinas (ML). Comienza con una revisión de los métodos analíticos y numéricos, pilares fundamentales en la solución de ecuaciones diferenciales que han servido de base para muchos de los desarrollos posteriores en este campo. La atención se centra en la resolución de problemas inversos, una área de creciente interés que a menudo emplea metaheurísticas avanzadas para abordar desafíos únicos asociados con estas ecuaciones. Este análisis lleva a explorar los métodos convencionales de ML, que han empezado a jugar un papel significativo en este campo. Se pone especial énfasis en el ML orientado por la física, un área emergente que representa el foco principal de la investigación actual, y que promete integrar de manera sinérgica el rigor de los principios físicos con las avanzadas capacidades de las técnicas de aprendizaje automático.

1.1.1. Métodos analíticos

Los métodos analíticos han sido pilares fundamentales en el estudio de las ecuaciones diferenciales en diversas disciplinas como la física y la ingeniería. Entre ellos, el Método de Separación de Variables es clave para transformar PDEs en ecuaciones diferenciales ordinarias (ODEs) más simples, detallado en la obra de Nagle, Saff y Snider [11].

La Transformada de Laplace, otra técnica esencial para ODEs, facilita la transformación de ecuaciones del dominio del tiempo a ecuaciones algebraicas en el dominio de Laplace, descrita por Goldberg y Potter [12]. Paralelamente, la Transformada de Fourier es crucial en PDEs, permite convertir funciones en conjuntos de senos y cosenos, facilitando su resolución. Esta técnica ha sido innovada por S. Nourazar y A. Nazari-Golshan, quienes la combinaron con el método de perturbación homotópica para resolver la ecuación de reacción-difusión de Cauchy no lineal [13].

El Método de Series de Potencias se utiliza para resolver ODEs mediante la expansión en series de potencias. Un desarrollo notable en este enfoque se encuentra en el trabajo de Muhammad Imran Liaqat et al., donde se propone el método de series de potencias residuales de Aboodh para ecuaciones diferenciales parciales fraccionarias en el tiempo [14]. Por otro lado, el Método de Funciones de Green, útil en PDEs lineales no homogéneas, ha sido explorado por Cole et al. en el contexto de la conducción de calor [15].

En lo que respecta a los métodos semi-analíticos, el Método de Perturbación utiliza un pequeño parámetro para simplificar ODEs y PDEs. La eficacia de este enfoque se destaca en el análisis de Muhammad Imran Liaqat y E. Okyere de las ecuaciones de precios de opciones de Black-Scholes fraccionarias en el tiempo [16]. El Método de Homotopía, que aproxima soluciones de ecuaciones difíciles mediante deformación continua, ha demostrado utilidad en sistemas complejos como los reactores nucleares, observado en el trabajo de Mohammad Shqair et al. [17].

Estos métodos representan herramientas fundamentales en el estudio de ecuaciones diferenciales y continúan su evolución para ofrecer soluciones más eficientes en diversos problemas complejos. Tras explorar los métodos analíticos, es importante considerar cómo los métodos numéricos complementan y expanden estas técnicas, especialmente en contextos donde las soluciones analíticas no son accesibles.

1.1.2. Métodos numéricos

Los métodos numéricos desempeñan un papel crucial en la resolución de ODEs y PDEs, en especial cuando las soluciones analíticas son inaccesibles. El Método de Runge-Kutta, en particular la clase de métodos de dos pasos, ha demostrado eficiencia en evaluaciones de funciones, según lo descrito por Zdzisław Jackiewicz y S. Tracogna [18]. Por otro lado, el Método de Euler, significativo en el campo, ha sido examinado en relación con la aplicación de grupos de Lie [19]. Asimismo, el Método de Heun, un método numérico implementado en MATLAB [20].

Para ecuaciones de mayor orden, los Métodos Adams-Bashforth y Adams-Moulton se han aplicado a ODEs fraccionarias, como se muestra en el estudio de Hacı Mehmet Baskonus y Hasan Bulut [21]. Estos métodos representan un avance en la resolución de ecuaciones diferenciales fraccionarias.

En el ámbito de las PDEs, el Método de Diferencias Finitas es un enfoque clásico bien documentado en el trabajo de George E. Forsythe, Wolfgang Wasow y W. Nachbar [22]. El Método de Elementos Finitos, aplicado a PDEs con entradas aleatorias, se explora en [23] por Max Gunzburger, Clayton G. Webster y Guannan Zhang. Además, el Método de Volúmenes Finitos, comparado con otros métodos en [24] por Joaquim Peiró y Spencer J. Sherwin, se ha establecido como una herramienta versátil para PDEs.

El Método de Líneas, utilizado en la integración de PDEs, se detalla por William E. Schiesser [25], y el Método de Descomposición Espectral, muestra su aplicación en programación en C [26].

Estos métodos numéricos, en constante desarrollo y refinamiento, son esenciales en la investigación y aplicación de ecuaciones diferenciales en diversas disciplinas científicas y de ingeniería, proporciona soluciones a problemas complejos. Además de los métodos analíticos y numéricos, los problemas inversos presentan desafíos únicos en la resolución de EDs, requieren enfoques especializados que se abordarán a continuación.

1.1.3. Problemas inversos

En el ámbito de los problemas inversos relacionados con ecuaciones diferenciales, se identifican dos enfoques principales que se distinguen por su adaptabilidad y precisión, en dependencia del grado de conocimiento previo sobre la estructura de la ecuación diferencial. Estos enfoques varían según si la estructura de la ecuación diferencial es conocida o desconocida, y ambos dependen de la disponibilidad de datos del proceso modelado, ya sean observaciones o simulaciones.

Cuando la estructura de una ecuación diferencial paramétrica es conocida, el objetivo se centra en la optimización de parámetros para ajustar la ecuación de manera precisa. Un ejemplo sobresaliente de esta aplicación es el uso de la Optimización por Enjambre de Partículas (PSO) para calibrar los parámetros en un sistema de ODEs paramétricas [27]. Este método ha probado ser eficaz en proporcionar modelos precisos para fenómenos complejos, como la propagación de enfermedades en redes complejas.

En contraste, cuando la estructura de la ecuación diferencial es desconocida, se adopta un enfoque de aproximación de la ecuación utilizando matemáticas simbólicas. Este método se emplea un algoritmo genético simbólico para descubrir ecuaciones diferenciales parciales a partir de los datos [28]. Este enfoque innovador prescinde del conocimiento previo de la estructura de la ecuación y, en cambio, utiliza un algoritmo genético para optimizar la representación y estructura de la ecuación basándose en datos disponibles.

Ambos enfoques resaltan la relevancia de los datos en el proceso de ajuste de parámetros de ecuaciones conocidas o en el descubrimiento de nuevas ecuaciones cuando la estructura es desconocida. Estos métodos numéricos reflejan la flexibilidad y capacidad de adaptación necesarias para abordar una amplia gama de problemas complejos en distintas áreas, desde la epidemiología hasta la física y más allá. Con el avance de la tecnología y la creciente complejidad de los problemas científicos, las técnicas de ML han emergido como herramientas poderosas en la resolución de EDs.

1.1.4. Aprendizaje de Máquinas

Los métodos basados en DL, han demostrado ser una fusión innovadora en la resolución de ecuaciones diferenciales. Los primeros enfoques [29], emplearon algoritmos neuronales para resolver numéricamente ecuaciones diferenciales mediante ecuaciones de diferencias finitas. Este método pionero abrió camino para la utilización de NNs en ecuaciones diferenciales, al mostrar el potencial para simular procesos numéricos complejos.

Lagaris et al. [30] introdujeron un método innovador que utiliza redes neuronales artificiales (ANNs) para resolver problemas de valores iniciales y de frontera, aplicable a una amplia gama de ecuaciones. Esta solución se estructuró en dos partes: una sección que cumplía automáticamente con las condiciones iniciales y de frontera, y otra sección que incorporaba una red neuronal feedforward con parámetros ajustables. Este enfoque mostró ventajas significativas sobre el método de elementos finitos en términos de precisión y eficiencia computacional.

El estudio de Kumar et al. [31] realizó una revisión exhaustiva de diversas técnicas que emplean Perceptrones Multicapa (MLP) y Funciones de Base Radial (RBF) en la solución de ecuaciones diferenciales, destaca la capacidad para manejar ecuaciones complejas de manera eficiente. Asimismo, se resaltó que las técnicas basadas en RBFNN tienen una mayor precisión y mejores propiedades de generalización en comparación con las redes MLP y los métodos tradicionales.

El trabajo de Nick Winovicha et al. [32] presentó el marco ConvPDE-UQ para la construcción de solucionadores numéricos utilizando Redes Neuronales Convolucionales (CNNs). Este enfoque evita los costosos pasos de inferencia bayesiana y permite escalar el entrenamiento a grandes conjuntos de datos. PDE-Net, introducido por Zichao Long et al. en [33], es una red neuronal profunda que aprende operadores diferenciales mediante kernels de convolución, y ofrece la flexibilidad de aprender

tanto operadores diferenciales como respuestas no lineales desconocidas.

Además, el desarrollo de DeepONet por Lu Lu et al. [6, 34], representa un avance significativo, centrándose en aprender operadores no lineales a partir de datos. DeepONet utiliza dos subredes para codificar la función de entrada y las ubicaciones de salida, lo que permite una aproximación eficiente de operadores funcionales.

El estudio de Leake et al. [35] explora la aplicación de Máquinas de Soporte Vectorial (SVM), en especial las Máquinas de Soporte Vectorial de Mínimos Cuadrados (LS-SVMs), en conjunto con la Teoría de Conexiones Funcionales (TFC) para resolver ecuaciones diferenciales. Este enfoque muestra el gran potencial de la integración de algoritmos de aprendizaje automático en el marco de TFC para aplicaciones futuras.

Estos avances en la aplicación de técnicas de ML para resolver ecuaciones diferenciales abren nuevas posibilidades en el análisis y solución de problemas complejos en diversas áreas, aprovecha la capacidad para transformar funciones, reducir el tiempo y los recursos computacionales necesarios. Dentro del DL, las PINNs representan un enfoque revolucionario, integra la inteligencia artificial con principios físicos.

1.1.5. Aprendizaje de Máquinas Orientado por la Física

Después de explorar una gama de técnicas clásicas para resolver ecuaciones diferenciales mediante el uso de ML, ahora se profundizará en una nueva familia de soluciones: aquellas basadas en la idea subyacente de las PINNs. Esta aproximación innovadora gana terreno rápidamente en el campo del Aprendizaje de Máquinas Científico (SciML), debido a la capacidad única para incorporar principios físicos conocidos en la arquitectura de aprendizaje profundo. Las PINNs, y sus variaciones, representan un salto significativo en la resolución de problemas complejos de PDEs, ya que integran el conocimiento físico en el proceso de aprendizaje de la red.

A continuación, se explorarán varios estudios y desarrollos recientes que se han inspirado en esta técnica, ampliados y enriquecidos en el campo de la resolución de ecuaciones diferenciales con herramientas de ML. Cada uno de estos enfoques ofrece una perspectiva única y soluciones innovadoras, marca el camino hacia un futuro donde la integración de la física y el aprendizaje automático se vuelva cada vez más sinérgica.

Este concepto, que combina principios físicos con NNs para resolver ecuaciones diferenciales, tiene sus raíces en investigaciones anteriores que integraron el conocimiento previo estructurado en el aprendizaje automático. Un hito fundamental en este desarrollo fue el trabajo de Owhadi [36], que destacó la importancia de utilizar conocimientos previos estructurados en la solución de problemas numéricos.

Raissi et al. [37, 38], emplearon la regresión de procesos gaussianos para construir representaciones de operadores lineales, ofrecen soluciones precisas e incertidumbres para una variedad de problemas físicos. Estos estudios fueron pioneros en combinar el aprendizaje automático probabilístico con ecuaciones diferenciales, permiten el uso de datos ruidosos y de multifidelidad para superar las limitaciones de la discretización numérica y problemas de consistencia en la integración temporal.

Se reformula la homogeneización numérica como un problema de inferencia bayesiana, donde el conocimiento previo estructurado sobre la solución juega un papel crucial en situaciones con datos limitados o ruidosos [36]. Este enfoque mejora la interpretabilidad y previene el sobreajuste en los resultados, es de gran utilidad en diversos campos científicos y de ingeniería.

Por otro lado, [38] extiende este marco para abordar sistemas más complejos, como ecuaciones con coeficientes variables y sistemas de ecuaciones integro-diferenciales lineales. Este enfoque absorbe la variabilidad en los coeficientes dentro de los kernels y utiliza la regresión de proceso gaussiano multivariante para capturar correlaciones cruzadas.

Las PINNs, introducidas y desarrolladas en 2017 [1, 2], representan un avance significativo. Estas redes no solo consideran la ecuación diferencial subyacente, sino que también integran las ecuaciones gobernantes y las condiciones iniciales/fronterizas en el proceso de entrenamiento. Raissi et al., en la versión consolidada de 2019 [3], proporcionan una visión integral del enfoque PINNs, demuestra la utilidad en la inferencia de soluciones a PDEs y en el descubrimiento de PDEs guiado por datos.

Este trabajo integral de Raissi et al. ilustra cómo las PINNs pueden abordar tanto problemas directos e inversos en PDEs no lineales: las ecuaciones de Schrödinger, Burgers y Allen–Cahn; manejan con eficacia los parámetros del modelo aprendidos a partir de datos observables. En conjunto, estos avances representan una evolución notable en el uso del aprendizaje automático para resolver ecuaciones diferenciales, aprovechan la integración del conocimiento físico estructurado y las técnicas de aprendizaje profundo.

El Deep Ritz Method (DRM), vanguardia de las técnicas de aprendizaje automático aplicadas a ecuaciones diferenciales, introducido por Weinan et al. [4], representa una innovación destacada. Este método reformula la resolución de PDEs en términos de la minimización de funcionales de energía, utiliza NNs para la aproximación de soluciones. La esencia del DRM radica en el enfoque variacional, donde la función de pérdida se basa en principios físicos, integra de manera efectiva la física en el núcleo del aprendizaje de la red.

A diferencia de las técnicas tradicionales, el DRM se especializa en problemas variacionales, destacando su adaptabilidad y capacidad para manejar escenarios de alta complejidad y dimensiones elevadas. La aplicación no se limita a problemas variacionales estándar, sino que también muestra potencial para extenderse a otros contextos relacionados con la construcción de funciones. Este enfoque representa un avance significativo en la unión de la física y el aprendizaje automático, similar a las PINNs, y marca un desarrollo notable en la solución de ecuaciones diferenciales mediante técnicas avanzadas de DL.

El Deep Galerkin Method (DGM), tendencia innovadora en el campo del aprendizaje automático para resolver ecuaciones diferenciales, propuesto por Justin Sirignano [5]. El DGM, inspirado en el método de Galerkin, se basa en la idea de multiplicar el residuo de la ecuación diferencial por una función de prueba. Lo que distingue al DGM es el uso de DNNs para representar la solución y la función de prueba, lo que resulta en una aproximación precisa de la solución.

El enfoque de DGM se caracteriza por emplear DNNs en lugar de las combinaciones lineales de funciones base que son típicas en los métodos de Galerkin convencionales. La principal fortaleza de DGM radica en su naturaleza sin malla, lo cual es crucial para abordar con eficiencia problemas de alta dimensión. Los métodos tradicionales a menudo se vuelven computacionalmente inviables en escenarios de alta dimensión debido al aumento exponencial en el número de puntos de malla y la necesidad de disminuir el tamaño de los pasos de tiempo.

DGM ha demostrado la capacidad para resolver con precisión PDEs de alta dimensión, incluye la PDE de Hamilton-Jacobi-Bellman y la ecuación de Burgers, incluso en configuraciones de hasta 200 dimensiones. Este algoritmo no solo maneja diversas condiciones de contorno y físicas, sino que también proporciona un marco teórico que valida la capacidad de aproximación de las NNs para una clase específica de PDEs parabólicas cuasilineales.

En la misma línea de innovación que las PINNs, el DRM y el DGM, otro enfoque notable en el uso del aprendizaje automático para resolver ecuaciones diferenciales es el de las Redes Neuronales con Restricciones Físicas (PCNNs). Este enfoque, que refuerza las restricciones físicas del problema, como las leyes de conservación u otras propiedades conocidas durante el proceso de entrenamiento, asegura que la solución de la red neuronal satisfaga la física subyacente.

Zhu et al. [39], aborda el modelado sustituto y la cuantificación de incertidumbre en sistemas de PDEs desde una perspectiva de aprendizaje profundo no supervisado. Esta investigación se distingue por integrar las ecuaciones gobernantes de los modelos físicos directamente en las funciones de pérdi-

da/probabilidad, lo cual permite entrenar modelos profundos sin la necesidad de datos etiquetados, al usar únicamente datos de entrada. Se emplea una red neuronal convolucional de codificador-decodificador y un modelo generativo condicional basado en flujos para resolver PDEs, construir modelos sustitutos y cuantificar incertidumbres.

La metodología propuesta por Zhu et al. se enfoca en minimizar la divergencia Kullback-Leibler inversa entre la densidad predictiva del modelo y una densidad condicional de referencia, definida por la distribución Boltzmann-Gibbs. Esta técnica permite la generalización de los modelos a entradas no vistas y posibilita una cuantificación efectiva de la incertidumbre predictiva. Este enfoque representa un avance significativo en el aprendizaje automático aplicado al modelado sustituto y la cuantificación de incertidumbre, en la solución de PDEs.

Los principios físicos en el aprendizaje automático para la resolución de ecuaciones diferenciales, cuenta con un enfoque aún más avanzado: las Redes Neuronales Variacionales Informadas por la Física (VPINNs). Este método, introducido por Ehsan Kharazmi et al. [40], representa una evolución de las PINNs, pero con un giro distintivo hacia el enfoque variacional.

Las VPINNs se construyen sobre la base metodológica de Petrov-Galerkin, seleccionando el espacio de las DNNs para las funciones de prueba y el espacio de los polinomios de Legendre para las funciones de ensayo. Esta configuración se utiliza para formular el residuo variacional de las PDEs, integrándolo en la función de pérdida de la red neuronal. Una innovación clave de las VPINNs es la integración por partes dentro de la formulación variacional, lo que reduce el orden de los operadores diferenciales representados por NNs, disminuye así el costo de entrenamiento y aumenta la precisión en comparación con las PINNs tradicionales.

En la evolución de las técnicas de aprendizaje automático aplicadas a las ecuaciones diferenciales, se destaca el ecosistema de software de SciML, descrito en el trabajo de Rackauckas et al. [41]. Este enfoque representa una integración pionera de leyes físicas y modelos científicos con técnicas de aprendizaje automático, centrada en las ecuaciones diferenciales universales (UDEs). Las UDEs proporcionan un marco matemático clave para conectar diversas aplicaciones dentro de SciML, abarca desde el descubrimiento automático de mecanismos biológicos hasta la solución de ecuaciones de Hamilton-Jacobi-Bellman en alta dimensión.

SciML se destaca por su capacidad para manejar estocasticidad, retrasos y restricciones implícitas, optimiza el entrenamiento, aprovecha el paralelismo distribuído y los aceleradores de GPU. Un componente esencial de este ecosistema es NeuralPDE.jl, que facilita el uso de PINNs para resolver PDEs a través de una interfaz simbólica y automatización del código de entrenamiento. La eficiencia y precisión del proceso se mejoran mediante métodos adaptativos para reponderar las funciones de pérdida y compatibilidad con GPU para acelerar el entrenamiento.

En la búsqueda constante de métodos más eficaces para resolver ecuaciones diferenciales se utiliza ML, una técnica destacada es el enfoque Kriging, también conocido regresión por procesos gaussianos. Un ejemplo sobresaliente de esta técnica se presenta por Zhu et al. [42], que combina la flexibilidad del Kriging con el conocimiento derivado de las PDEs.

El Kriging, conocido por su flexibilidad y expresiones de predicción en forma cerrada, enfrenta el desafío de la escasez de datos de medición, un problema común en los sistemas de ingeniería debido a limitaciones de medición y altos costos de detección. Aquí es donde el modelo PDE Informed Kriging (PIK) brilla, pues incorpora el conocimiento físico disponible en forma de PDEs. Este modelo introduce información de PDEs a través de un conjunto de puntos y realiza predicciones posteriores similares al método de kriging estándar.

La evolución de PIK es el marco Active PIK (APIK), que mejora aún más el rendimiento de aprendizaje mediante el diseño activo de puntos de PDE. Estos puntos seleccionados no solo exploran todo el espacio de entrada sino que también se enfocan en ubicaciones donde la información de PDEs es crítica para reducir la incertidumbre predictiva. La selección de estos puntos es un paso crucial

para optimizar el rendimiento del aprendizaje.

El estudio utiliza un algoritmo de expectación-maximización para la estimación de parámetros, lo que permite actualizaciones eficientes y una selección precisa de puntos de PDE. Este enfoque mejora la eficiencia del modelo al optimizar la estimación de parámetros y la selección de puntos informativos.

La efectividad del método APIK se demuestra a través de ejemplos sintéticos y estudios de caso reales, resalta cómo APIK supera en rendimiento a los métodos de kriging y PIK estándar, en escenarios donde los datos de medición son limitados. Este estudio presenta un enfoque novedoso y eficaz para la modelización y predicción en sistemas de ingeniería, destaca la integración efectiva del conocimiento físico en forma de PDEs y la implementación de un método activo y eficiente para la selección y utilización de puntos de PDEs.

En la intersección del aprendizaje automático científico y la modelización inversa, surge un enfoque novedoso conocido como "Adversarial Inverse Modeling" (AIM), delineado en el trabajo de Xu et al. [43]. Este método representa un cambio significativo en el tratamiento de problemas inversos en modelos estocásticos, destaca por la aplicación única de NNs para estimar distribuciones desconocidas. AIM se distingue por el uso de una red neuronal discriminativa para calcular discrepancias estadísticas entre procesos aleatorios observados y simulados, es efectivo en modelos estocásticos complejos.

La metodología de AIM se inspira en las Redes Generativas Adversarias (GANs), combina técnicas avanzadas de DL, la diferenciación automática y redes neuronales generativas. Esta integración permite abordar problemas inversos con mayor eficiencia y precisión que los métodos convencionales. AIM redefine el modelado inverso tradicional, formula el problema como un proceso de entrenamiento adversario, donde una red generativa se entrena para producir datos que imiten fielmente los observados. Esto conduce a una estimación más acertada de distribuciones desconocidas y mejora la eficiencia computacional.

A pesar de los desafíos en implementación, estabilidad, convergencia y la selección de arquitecturas de red, se propone el uso del optimizador L-BFGS para mejorar la estabilidad del modelo. AIM se postula como una herramienta valiosa en áreas de la ingeniería y las finanzas, permite el aprendizaje de distribuciones complejas en sistemas de ecuaciones diferenciales parciales y la estimación de parámetros en modelos estocásticos.

En la evolución de las metodologías para resolver ecuaciones diferenciales (EDs) mediante ML, el desarrollo de las Redes Profundas de Operadores Orientadas por la Física (PIDeepONets) destaca un hito significativo, que representa la convergencia de dos enfoques inicialmente independientes: las PINNs y las DeepONets. Las PINNs, a través de un aprendizaje no supervisado utiliza un conjunto de puntos de colocación, y logran aproximar la solución de la ecuación diferencial. En contraste, las DeepONets, mediante un conjunto de datos de entrada-salida, entrenan una red neuronal para aprender un operador no lineal implícito en la ecuación diferencial [7].

Los PIDeepONets, al integrar los principios físicos con la arquitectura de DeepONet, permiten un aprendizaje no supervisado, elimina la necesidad de conjuntos de datos emparejados de entrada-salida. Esta innovación ofrece una predicción rápida y precisa de soluciones para diversas PDEs paramétricas, supera a los solucionadores convencionales de PDEs en velocidad por hasta tres órdenes de magnitud. Además, esta técnica mejora la exactitud predictiva y el rendimiento de generalización, incluso en escenarios de extrapolación, al tiempo que demuestra una notable eficiencia en el uso de datos, al reducir la cantidad de ejemplos requeridos para el entrenamiento del modelo.

Fu et al. [44] presenta una variante novedosa de la red de Funciones de Base Radial (RBFs) denominada Redes Neuronales de Funciones de Núcleo Informadas por la Física (PIKFNNs), diseñada para abordar diversas PDEs. Las PIKFNNs introducen una innovación clave, el uso de Funciones de Núcleo Informadas por la Física (PIKFs) como funciones de activación, en lugar de las tradicionales

RBFs. Estas PIKFs se derivan de las ecuaciones gobernantes de las PDEs consideradas, ofrece una aproximación más precisa a las soluciones de las PDEs. Una diferencia distintiva con las conocidas PINNs es que las PIKFNNs integran la información física en las funciones de activación en lugar de en la función de pérdida.

Esta integración directa en las funciones de activación representa un cambio significativo en la forma en que se abordan las PDEs mediante el aprendizaje automático. Al utilizar las PIKFs, que satisfacen las ecuaciones gobernantes de PDEs homogéneas, no homogéneas y transitorias, las PIKFNNs solo requieren datos de contorno o iniciales para entrenar la red. Esta característica simplifica el proceso de entrenamiento y mejora la eficiencia computacional.

El estudio de Fu et al. valida la viabilidad y precisión de las PIKFNNs en varios ejemplos de referencia, abarca desde problemas de propagación de ondas de alta frecuencia hasta problemas en dominios infinitos y problemas inversos. Estos ejemplos demuestran la capacidad de las PIKFNNs para manejar una variedad de situaciones complejas y desafiantes.

Dentro del panorama del ML aplicado a la resolución de ecuaciones diferenciales, surge un enfoque novedoso que integra la arquitectura de Transformer en las PINNs [45]. Esta innovación aborda las deficiencias de los PINNs convencionales y ofrece soluciones mejoradas para una resolución más precisa y eficiente de PDEs. Presenta como aportes principales la incorporación de la estructura de Transformer para capturar dependencias temporales a largo plazo, crucial en PDEs donde las condiciones en un momento pueden influir en estados futuros o pasados. Este enfoque, denominado PINNsFormer, utiliza un Generador de Secuencias Pseudo para convertir entradas vectorizadas en secuencias temporales pseudo, y modela estas dependencias de manera más efectiva. Además, introduce una innovadora función de activación Wavelet, diseñada para mejorar la capacidad del modelo para aproximar soluciones complejas de PDEs.

Las innovaciones y características destacadas de PINNsFormer incluyen una mejora notable en escenarios donde los PINNs tradicionales suelen fallar, en casos con características de alta frecuencia o multiescala. A diferencia de los PINNs convencionales, PINNsFormer muestra una menor sensibilidad a los hiperparámetros, facilitando así su ajuste y optimización. A pesar de su arquitectura más compleja, mantiene una simplicidad computacional práctica, evita operaciones complicadas y aprovecha capas lineales con funciones de activación no lineales.

Este desarrollo se suma a una serie de enfoques innovadores en el campo del aprendizaje automático para resolver ecuaciones diferenciales. Las PINNs y sus variaciones, como las PCNNs, VPINNs, y ahora PINNsFormer, transforman el panorama de SciML. Cada uno de estos enfoques ofrece soluciones únicas y creativas, allana el camino hacia un futuro donde la integración de la física y el aprendizaje automático se vuelva cada vez más sinérgica y eficaz.

1.2. Redes Neuronales Informadas por la Física

En el ámbito de la resolución de ecuaciones diferenciales (EDs) mediante el aprendizaje automático, las PINNs integran la inteligencia artificial con principios físicos [1, 2, 3, 9]. Esta sección se dedica a explorar en detalle la teoría y la implementación de las PINNs, destaca la capacidad para resolver problemas complejos de ecuaciones diferenciales de una manera innovadora y eficiente.

Las PINNs representan una sinergia entre la física y las NNs, donde la red neuronal no es solo una caja negra, sino que está informada por las leyes físicas subyacentes del problema. En esencia, una PINNs es una función aproximadora, mapeando de $R^{n+1} \to R$, diseñada para estimar la solución de una ecuación diferencial. La clave del éxito de las PINNs reside en la habilidad para transformar un problema de aproximación en un problema de optimización, minimiza una función objetivo que refleja las características fundamentales de la solución de la ecuación diferencial.

Para comprender mejor las PINNs, considere una PDE genérica:

$$F(x_1, x_2, ..., x_n, t, U_{NN}, \frac{\partial(U_{NN})}{\partial x_1}, \frac{\partial(U_{NN})}{\partial x_2}, ..., \frac{\partial(U_{NN})}{\partial t}, \frac{\partial^2(U_{NN})}{\partial x_1^2}, \frac{\partial^2(U_{NN})}{\partial x_1 \partial x_2}, ..., \frac{\partial^2(U_{NN})}{\partial t^2}, ..., U_{NN}^{(n)}) = 0$$

donde U_{NN} es la red neuronal que aproxima la solución de la PDE. Las PINNs se entrenan para minimizar el error entre la solución predicha por la red y las condiciones físicas conocidas del problema. Este entrenamiento implica minimizar una función de pérdida compuesta que incluye términos para las condiciones iniciales, de frontera y residuales.

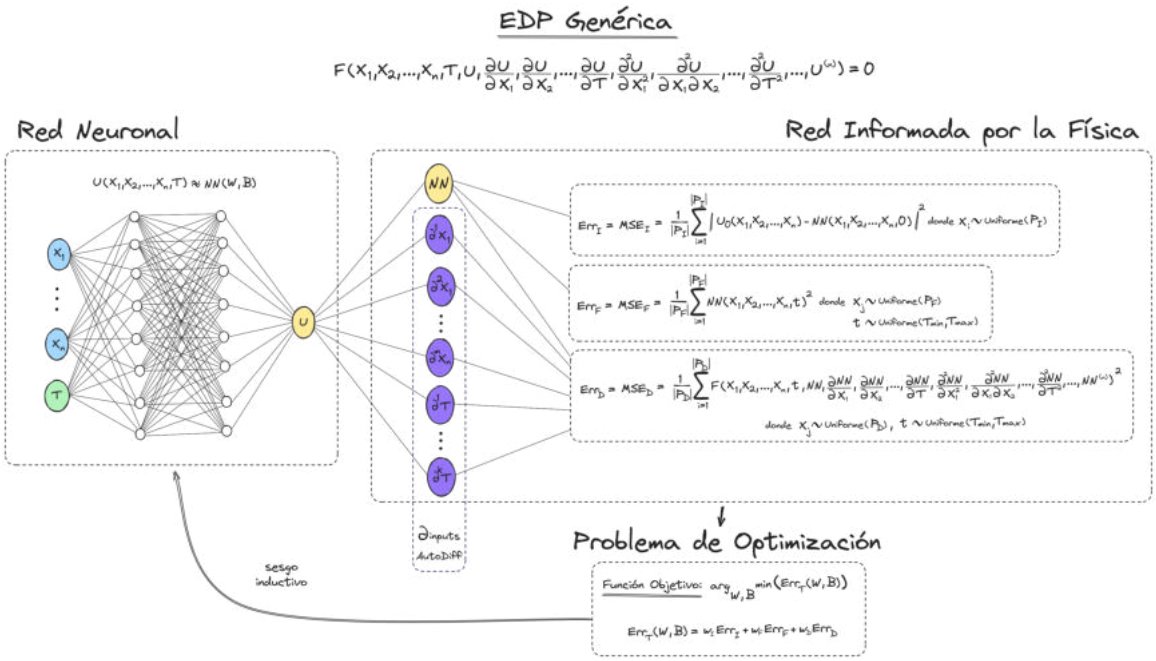


Figura 1.1: Arquitectura general de una PINN para resolver una PDE

El término de error de las condiciones iniciales, Err_I, se define como el error cuadrático medio (MSE) entre la solución de la red y la condición inicial en un conjunto de puntos con $T = 0$:

$$Err_I = MSE_I = \frac{1}{|P_I|} \sum_{i=1}^{|P_I|} \|u_0(x_1, x_2, ..., x_n) - U_{NN}(x_1, x_2, ..., x_n, 0)\|^2$$

El error de las condiciones de frontera, Err_F, evalúa la discrepancia en los puntos de frontera, asume en este caso una función Dirichlet con todos sus valores iguales a 0:

$$Err_F = MSE_F = \frac{1}{|P_F|} \sum_{i=1}^{|P_F|} U_{NN}(x_1, x_2, ..., x_n, 0)^2$$

La parte más crucial del enfoque de las PINNs es que la aproximación de la solución debe satisfacer la EDP en sí misma, lo que implica minimizar el error cuadrático medio de la evaluación de la

función F en un conjunto de puntos internos del dominio:

$$Err_D = MSE_D = \frac{1}{|P_D|}\sum_{i=1}^{|P_D|} F\left(x_1,\ldots,x_n,t,U_{NN},\frac{\partial U_{NN}}{\partial x_1},\ldots,\frac{\partial U_{NN}}{\partial t},\right.$$
$$\left.\frac{\partial^2 U_{NN}}{\partial x_1^2},\frac{\partial^2 U_{NN}}{\partial x_1 \partial x_2},\ldots,\frac{\partial^2 U_{NN}}{\partial t^2},\ldots,U_{NN}^{(n)}\right)^2$$

Un aspecto destacado de las PINNs es el uso de la diferenciación automática para calcular las derivadas parciales necesarias en la función F. Esto facilita la implementación y aumenta la eficiencia computacional.

A modo de conclusión, las PINNs representan un avance significativo en la resolución de ecuaciones diferenciales mediante el aprendizaje automático. Al integrar las leyes físicas en el proceso de entrenamiento de la red neuronal, las PINNs ofrecen una aproximación más precisa y físicamente coherente a la solución de problemas complejos de EDs, abre nuevas posibilidades en campos diversos: la ingeniería, la física y más allá. Además de las PINNs, las DeepONet ofrecen una perspectiva disruptora en la resolución de EDs mediante técnicas de ML.

1.3. Redes Neuronales de Operadores (DeepONet)

En el ámbito de la resolución de ecuaciones diferenciales, las DeepONet ofrecen un enfoque centrado en aprender operadores no lineales a partir de los datos [6, 34, 7, 46, 47]. Esto representa un avance significativo en la comprensión y manejo de ecuaciones diferenciales complejas. Las DeepONet, con su estructura única y capacidad para capturar relaciones no lineales entre funciones, se han posicionado como una herramienta poderosa en la resolución de EDs.

Fundamentos y Teoría

El desarrollo de las DeepONet se basa en la extensión del Teorema de Aproximación Universal de Operadores. Este teorema establece que una red neuronal con una sola capa oculta puede aproximar cualquier operador continuo no lineal de manera precisa. DeepONet adopta esta teoría y la extiende a DNNs, lo que permite la aproximación de una gama más amplia de operadores, incluye aquellos que son altamente no lineales y complejos.

Arquitectura de DeepONet

La arquitectura de DeepONet se compone de dos subredes principales: la red de ramas (branch net) y la red de tronco (trunk net). La red de ramas se encarga de codificar el espacio de funciones de entrada discretas, transformándolas en una representación compacta. Por otro lado, la red de tronco procesa esta representación para generar la salida deseada. La combinación de estas dos redes permite a DeepONet aprender una variedad de operadores explícitos e implícitos.

Minimización del Error de Generalización

DeepONet utiliza estrategias de una arquitectura no apilada y la adición de sesgos para minimizar el error de generalización. Este enfoque difiere de los métodos tradicionales en aprendizaje automático, enfocándose en el aprendizaje de operadores continuos o sistemas complejos a partir de

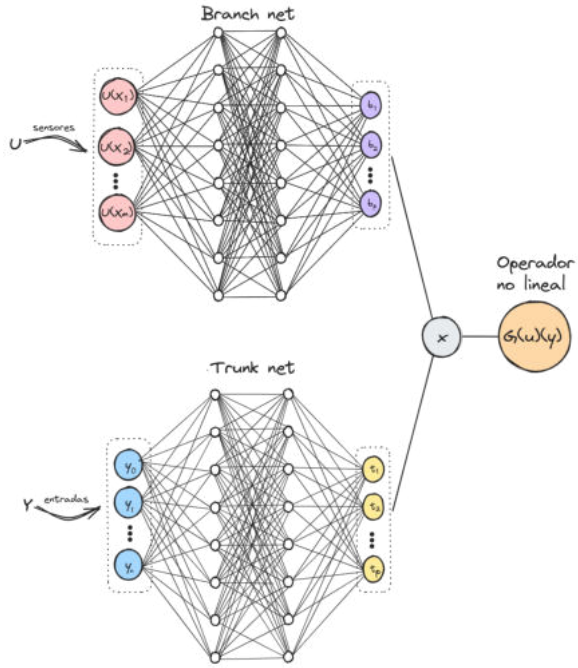


Figura 1.2: Arquitectura genérica de una DeepONet

datos dispersos. La generalización de DeepONet se logra a través de técnicas cuidadosas de entrenamiento y regularización, lo que le permite predecir con precisión la acción de operadores en funciones no vistas.

Representación del Espacio de Entrada y Aplicaciones

La selección y representación del espacio de entrada en DeepONet es crucial para su efectividad. Diferentes espacios, como los espacios de polinomios ortogonales o campos aleatorios gaussianos, influyen en el error de generalización de DeepONet. Este enfoque se ha aplicado con éxito en casos de uso, el aprendizaje de operadores explícitos e implícitos, y en la simulación de sistemas multiescala y multifísicos.

Comparación con Métodos Numéricos Tradicionales y Desafíos

DeepONet ofrece ventajas sobre los solucionadores numéricos convencionales, en términos de generalización y representación implícita. Sin embargo, enfrenta desafíos en la generación de datos de entrenamiento y en la comprensión teórica de los errores de generalización y la complejidad del operador. Estos desafíos requieren una investigación adicional para mejorar la precisión de DeepONet en comparación con los métodos numéricos tradicionales.

Aplicaciones en Tiempo Real y Costos Computacionales

A pesar de un mayor costo computacional durante el entrenamiento, DeepONet es eficiente en la fase de predicción, lo que lo hace adecuado para aplicaciones en tiempo real. La capacidad

para aprender y aproximar operadores complejos lo convierte en una herramienta prometedora para aplicaciones diversas y en tiempo real.

Casos de Uso y Evaluaciones de Desempeño

DeepONet ha demostrado versatilidad en una variedad de aplicaciones, desde el aprendizaje de operadores explícitos e implícitos hasta la simulación de sistemas multiescala y multifísicos. Estos casos de uso revelan las limitaciones de las DeepONet, destacan la necesidad de una mayor investigación en la comprensión teórica y el desarrollo de nuevas arquitecturas de red.

Potencial para Aprendizaje Multiescala y Multifísico

DeepONet puede extenderse para abordar problemas multiescala y multifísicos mediante la incorporación de representaciones adecuadas del espacio de entrada y el diseño de arquitecturas de red neuronal adecuadas. Sin embargo, enfrenta desafíos en términos de disponibilidad de datos de alta calidad, diseño de representaciones y eficiencia computacional.

Futuras Direcciones de Investigación

Las áreas de investigación futuras para DeepONet incluyen entender el tamaño óptimo de la red para la aproximación de operadores, explorar diferentes arquitecturas de red y desarrollar técnicas para entrenar DeepONet con datos heterogéneos. La incorporación de conocimientos previos y el avance en la comprensión teórica también son áreas clave para el desarrollo futuro de DeepONet.

Integración de Conocimientos Previos y Nuevas Arquitecturas

La integración de conocimientos previos y la utilización de arquitecturas de redes alternativas, como CNNs o mecanismos de atención, pueden mejorar el desempeño, la precisión y la capacidad de generalización de las DeepONet. Esto representa un avance significativo en el aprendizaje automático aplicado a la resolución de ecuaciones diferenciales, fusionan técnicas avanzadas de DL con métodos analíticos. La capacidad para aprender operadores no lineales a partir de datos abre nuevas posibilidades en diversos campos de la ciencia y la ingeniería, marca un camino hacia el futuro donde la integración de la física y el aprendizaje automático se vuelva cada vez más sinérgica.

Además de las técnicas de aprendizaje automático aplicadas directamente a las ecuaciones diferenciales, los paquetes de software especializados han jugado un papel crucial en facilitar y agilizar el entrenamiento de modelos como las PINNs.

1.4. Softwares

En 2019, se lanzaron varios paquetes de software para facilitar el entrenamiento de las PINNs, se destacan por el uso de NNs y la diferenciación automática. Estos paquetes difieren en cómo manejan las condiciones de frontera, con una evaluación compleja en la función de pérdida, lo que justifica su necesidad en el diseño de PINNs.

DeepXDE [48], una biblioteca de Python, integra PINNs e incorpora PDEs en la función de pérdida de la red neuronal, permite abordar problemas directos e inversos. Se distingue por el método de refinamiento adaptativo basado en residuos y soporte para geometrías complejas, útil en educación e investigación.

Tabla 1.1: Principales bibliotecas de software diseñadas específicamente para el aprendizaje automático basado en PINNs

Nombre del Software	Backend	Uso
DeepXDE[48]	TensorFlow, Pytorch, JAX, PaddlePaddle	Solver
PyDEns[49]	TensorFlow	Solver
NVIDIA Moludus[50]	PyTorch	Solver
NeuroDiffEq[51]	PyTorch	Solver
SciANN[52]	TensorFlow	Wrapper
NeuralPDE[53]	Julia	Solver
ADCME[43]	Julia, TensorFlow	Wrapper
Nangs[54]	PyTorch	Solver
TensorDiffEq[55]	TensorFlow 2.x	Solver
IDRLnet[56]	PyTorch, Sympy	Solver
Elvet[57]	TensorFlow	Solver

NeuroDiffEq [51], construída sobre PyTorch, se especializa en resolver PDEs tradicionales mediante restricciones explícitas en la red neuronal, clasificándose como una PCNNs. A pesar de su eficacia, enfrenta limitaciones para ciertos límites no admitidos.

SimNet [58], un marco de simulación multifísica, soluciona múltiples configuraciones de manera simultánea, acelera las simulaciones y permite inferencias en tiempo real. Integra módulos de geometría sólida constructiva y de generación de nubes de puntos, que aumenta la flexibilidad y rendimiento.

Modulus [50], anteriormente NVIDIA SimNet, se destaca por la compatibilidad con redes de filtros multiplicativos y un método de agregación de gradientes, que maneja con eficacia los problemas multifísicos.

SciANN [52], basada en Tensorflow y Keras, simplifica la construcción de NNs para cálculos científicos, se destaca en la solución y el descubrimiento de PDEs.

PyDEns [49], un módulo de Python integrado con BatchFlow, permite a los usuarios controlar el proceso de entrenamiento y es eficaz en la predicción en contextos de propiedades geológicas inciertas y modelado inverso.

NeuralPDE.jl [53], parte de SciML, implementa PINNs en Julia con una interfaz simbólica para resolver PDEs, generar gemelos digitales y modelos neuronales sustitutos.

ADCME [59] combina simulaciones físicas y DNNs en un gráfico computacional unificado para resolver problemas inversos complejos, y supera los métodos tradicionales de discretización.

Nangs [54] utiliza NNs como funciones de aproximación para PDEs, ofrece soluciones continuas y diferenciables en todo el dominio.

TensorDiffEq [55], construída sobre TensorFlow 2.x, facilita la definición y resolución de problemas usando PINNs, con soporte escalable multi-GPU y compatibilidad con la API de Keras.

IDRLnet [56], sobre PyTorch, maneja problemas inversos ruidosos y ecuaciones integrales diferenciales, integra objetos geométricos y fuentes de datos en un marco de trabajo estructurado.

Elvet [57], una biblioteca Python, resuelve PDEs y problemas variacionales utilizando NNs para representar la solución, y ofrece flexibilidad en la definición de la arquitectura de la red.

Estos desarrollos en software reflejan la creciente sinergia entre la computación, la matemática y la física, explora nuevos horizontes en la resolución de ecuaciones diferenciales mediante técnicas de ML.

Capítulo 2

Propuesta

Este capítulo se dedica al desarrollo y análisis de metodologías avanzadas para resolver EDs, utiliza técnicas de aprendizaje profundo como las PINNs y las PIDeepONet. La discusión se articula en tres epígrafes principales, cada uno aborda un aspecto único de la resolución de ecuaciones diferenciales.

El primer epígrafe se enfoca en la Ecuación de Perona-Malik, una PDE fundamental en el procesamiento avanzado de imágenes y notoria por su naturaleza matemáticamente mal planteada. A pesar de los desafíos, esta ecuación es clave para el realce de bordes en imágenes. Se propone una metodología innovadora que utiliza las PINNs para abordar la resolución de esta ecuación, destaca las potencialidades y limitaciones de esta técnica en el contexto del procesamiento de imágenes. El segundo epígrafe analiza la reducción de la complejidad del problema de la Ecuación de Perona-Malik. Se examinan los posibles impedimentos en la solución de ecuaciones diferenciales similares, centrándose en la simplificación del problema para un análisis más detallado de las dificultades inherentes a este tipo de ecuaciones. El tercer epígrafe aborda la resolución de la ecuación del calor en una dimensión (1D) y utiliza la técnica de PIDeepONet. Representa un esfuerzo por aplicar técnicas de DL a una PDE más tradicional y bien entendida, proporciona así una evaluación del potencial de estas técnicas en la solución de problemas de difusión.

Cada uno de estos epígrafes contribuye a una comprensión más profunda de la aplicación de técnicas avanzadas de aprendizaje automático en la resolución de ecuaciones diferenciales, resaltan las capacidades y los desafíos de estos métodos emergentes. A través de este análisis, se espera no solo resolver ecuaciones específicas de manera eficiente, sino también ampliar el conocimiento sobre las capacidades y limitaciones de las PINNs y PIDeepONet. Establecen una base sólida para futuras investigaciones y aplicaciones prácticas en el procesamiento de imágenes y el SciML.

2.1. Ecuación diferencial de Perona-Malik

La Ecuación de Perona-Malik, situada en el epicentro de este estudio, representa un modelo matemático fundamental en el procesamiento avanzado de imágenes, reconocido por la eficacia que tiene en la detección y realce de bordes. A pesar de la naturaleza matemáticamente mal planteada, ha demostrado ser útil en aplicaciones prácticas, resaltan la capacidad para mejorar la identificación de bordes en imágenes.

Esta ecuación, definida por su característica difusión anisotrópica, ajusta la conductividad en función de la magnitud del gradiente de la imagen. Tal adaptabilidad permite un realce diferenciado de los bordes, dependen de las variaciones locales de intensidad, lo que resulta esencial para preservar

la calidad de la imagen mientras se enfatizan sus características más importantes.

La base matemática, se caracteriza por un coeficiente de difusión local que se adapta en función de las variaciones de intensidad en la imagen. Este coeficiente es más activo en áreas de baja variabilidad y se reduce en presencia de bordes marcados, permite así una difusión diferenciada que preserva los detalles críticos de la imagen.

Matemáticamente, la ecuación propuesta por Perona y Malik se formula como:

$$\frac{\partial I}{\partial t} = \operatorname{div}(c(x,y,t) \cdot \nabla I(x,y,t))$$

donde $I(x,y,t)$ representa la intensidad de la imagen en el punto (x,y) a lo largo del tiempo t, y $\nabla I(x,y,t)$ es el gradiente de la imagen, reflejan la tasa de cambio en la intensidad. El coeficiente $c(x,y,t)$, definido como una función del gradiente de la imagen, es:

$$c(x,y,t) = g(\|\nabla I(x,y,t)\|)$$

La función $g(\|\nabla I\|)$ es vital, ya que regula la sensibilidad del modelo a los cambios de intensidad, y debe satisfacer ciertas condiciones: aproximarse a 1 cuando la magnitud del gradiente es mínima (áreas homogéneas) y a 0 con gradientes pronunciados (bordes). Ejemplos de funciones de difusión comunes incluyen:

$$g_k(i) = \frac{1}{1 + \frac{i^2}{k^2}}$$

$$g_k(i) = e^{-\frac{i^2}{k^2}}$$

Estas funciones, con el parámetro ajustable k, controlan la sensibilidad del modelo a los bordes, siendo k un umbral que determina la prominencia requerida del gradiente para considerar un borde.

Diferenciándose de técnicas como el suavizado Gaussiano, la ecuación de Perona-Malik conserva fronteras claras entre regiones y mejora la localización de bordes en escalas gruesas mediante un coeficiente de difusión espacialmente variable. Este enfoque facilita la preservación y realce de bordes en la segmentación de imágenes multiescala, y reduce así la necesidad de pasos adicionales en el procesamiento. La estructura paralela promete eficiencia en el procesamiento de imágenes, y capacidad para manejar el ruido en la segmentación multiescala, utiliza estimaciones locales de contraste y ruido, representa un avance significativo en la visión por computadora y el procesamiento de imágenes.

Modelación de Problema con PINNs

La Ecuación de Perona-Malik, fundamental en el procesamiento de imágenes, se define matemáticamente como:

$$\frac{\partial I}{\partial t} = \nabla \cdot (c_k(\|\nabla I\|) \nabla I) \tag{2.1}$$

En esta ecuación, $I(x,y,t)$ la solución de la ecuación diferencial representa la intensidad de un pixel en la ubicación (x,y) y en un instante t, donde $I(x,y,0) = I_0$ con I_0 la condición inicial dada por una imagen de entrada. Esta ecuación modela cómo la intensidad de los píxeles de una imagen evoluciona a lo largo del tiempo, a través del mecanismo de difusión anisotrópica.

Para resolver esta ecuación diferencial, se propone una aproximación que utiliza una red neuronal, denotada por $U_{NN}(x,y,t) \approx I(x,y,t)$. Esta red neuronal actúa como aproximador universal de funciones, y procesando un tensor tridimensional (x,y,t) de entrada y genera un tensor unidimensional que estima la intensidad de un pixel en el tiempo t.

Inclusión del Parámetro de Sensibilidad k

Dado que la ecuación es paramétrica en función del parámetro k, que controla la sensibilidad de la función de difusión local, se extiende la entrada de la red neuronal para incluir este parámetro. Ahora, la red neuronal procesa un vector de cuatro dimensiones (x, y, k, t), permite modelar una familia de soluciones para diferentes valores de k.

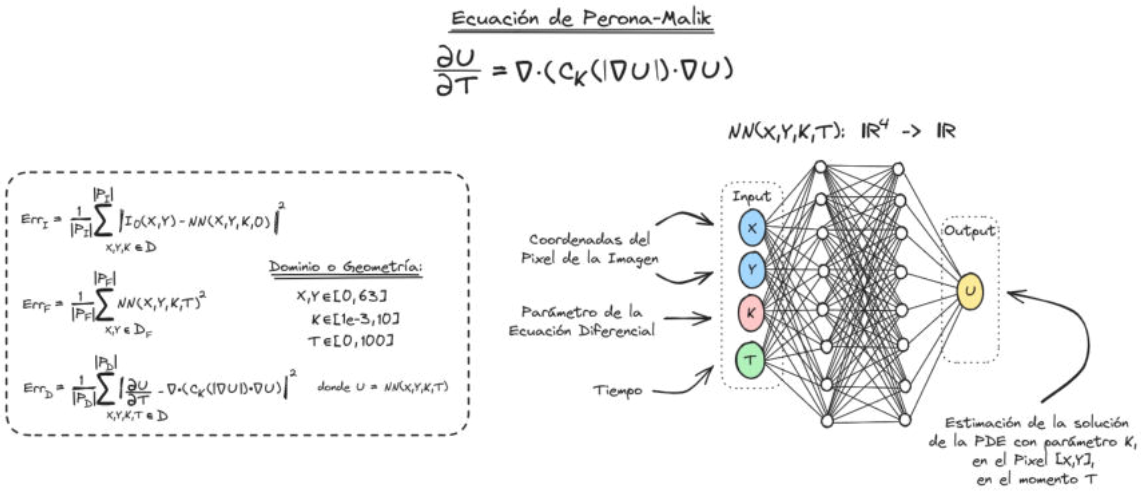


Figura 2.1: Red Neuronal para Perona-Malik

Definición del Dominio de Entrada

El dominio sobre el cual se entrena la red neuronal se define como sigue:

- Las coordenadas espaciales x, y se encuentran en el intervalo $[0, 63]$, adecuado para imágenes de 64×64 píxeles.
- El parámetro k varía en el rango $[1e{-}3, 10]$, cubriendo un espectro amplio de sensibilidad a los bordes.
- El tiempo t se considera en el intervalo $[0, 100]$, con una ventana temporal amplia para analizar la evolución de la imagen.

Construcción de la Función de Pérdida

La función de pérdida utilizada para entrenar la red neuronal comprende tres componentes esenciales:

1. **Error de las Condiciones Iniciales**:

$$Err_I = \frac{1}{|P_I|} \sum_{x,y,k \in D}^{|P_I|} \|I_0(x,y) - U_{NN}(x,y,k,0)\|^2 \tag{2.2}$$

Este término mide la discrepancia entre la imagen inicial y las predicciones de la red en $t = 0$.

2. **Error de las Condiciones de Frontera**:

$$Err_F = \frac{1}{|P_F|} \sum_{x,y \in D_F}^{|P_F|} U_{NN}(x,y,k,t)^2 \tag{2.3}$$

   Este término evalúa el cumplimiento de las condiciones de frontera, esencial en el mantenimiento de la integridad estructural de la imagen.

3. **Error Residual de la Ecuación Diferencial**:

$$Err_D = \frac{1}{|P_D|} \sum_{x,y \in D_F}^{|P_D|} \| \frac{\partial U_{NN}}{\partial t} - \nabla \cdot (c_k(\|\nabla U_{NN}\|)\nabla U_{NN}) \|^2 \tag{2.4}$$

   Este componente implica la incorporación de la información física en el entrenamiento de la red, asegurando que la solución aproximada cumpla con la dinámica de la ecuación diferencial.

Estrategia de Optimización y Conclusiones

La optimización de la red neuronal se lleva a cabo utilizando algoritmos como el descenso por gradiente estocástico o Adam, y se basa en la minimización de la función de pérdida definida. Este proceso se realiza sobre un dominio especificado, y emplea técnicas de diferenciación automática para calcular las derivadas necesarias de la red con respecto a las entradas. Este enfoque permite no solo la resolución precisa de la ecuación diferencial de Perona-Malik sino también ofrece una oportunidad para explorar aplicaciones más avanzadas en procesamiento de imágenes y visión por computadora, con las propiedades únicas de las PINNs.

Limitaciones

En el desarrollo de la implementación de las PINNs para abordar la ecuación diferencial de Perona-Malik, se han identificado ciertas limitaciones claves que plantean desafíos significativos. Estos obstáculos incluyen la dependencia de las condiciones iniciales específicas y el alto costo computacional asociado con el entrenamiento de la red neuronal.

1. **Dependencia de las Condiciones Iniciales**: una limitación crítica al emplear PINNs en la resolución de la ecuación de Perona-Malik radica en su dependencia explícita de una condición inicial determinada, que en este contexto es una imagen específica. Esta dependencia surge de la naturaleza de la función de pérdida utilizada para entrenar la red, la cual incluye un componente de error que se centra en las condiciones iniciales. Como resultado, para aplicar el proceso de difusión anisotrópica a nuevas imágenes usando el modelo ya entrenado, se requiere reentrenar la red neuronal con la nueva imagen de condición inicial. Esta restricción impide que las PINNs puedan generalizar la solución aprendida a diferentes condiciones iniciales, lo cual limita su utilidad en aplicaciones más amplias.

2. **Alto Costo Computacional**: otro desafío importante es el elevado costo computacional del entrenamiento de la red neuronal. Cada proceso de entrenamiento de una red de este tipo demanda un uso considerable de recursos computacionales. En ciertas instancias, este costo puede incluso superar al de los métodos numéricos tradicionales. Aunque una vez que el modelo está entrenado, es posible ajustar el parámetro k y la estimación del tiempo t sin necesidad de un nuevo entrenamiento, el costo inicial es un factor limitante significativo.

Estos impedimentos subrayan la necesidad de investigar enfoques alternativos o de realizar modificaciones en la estructura de las PINNs para superar tales limitaciones. La búsqueda de técnicas que permitan una generalización efectiva a diferentes condiciones iniciales y que reduzcan el costo computacional del entrenamiento será crucial para ampliar la aplicabilidad de las PINNs en el procesamiento avanzado de imágenes y en la resolución de ecuaciones diferenciales complejas como la de Perona-Malik.

Propuesta de solución

La propuesta para desarrollar un modelo de ML que aprenda y aplique el proceso de difusión anisotrópica de la ecuación de Perona-Malik se estructura en dos fases principales. Esta metodología busca superar las limitaciones observadas en las PINNs y los métodos numéricos tradicionales, enfocándose en la adaptabilidad y eficiencia del modelo.

Fase 1: Evaluación y Aprovechamiento de PINNs

1. **Ventajas de PINNs**: a diferencia de los métodos numéricos tradicionales, que son iterativos y dependientes del tamaño de paso y del tiempo estimado t, las PINNs poseen la capacidad de aprender y aproximar soluciones de ecuaciones diferenciales en un tiempo constante. Esto representa una ventaja significativa en términos de eficiencia y precisión, además que las PINNs operan sobre un dominio continuo en lugar de uno discreto.

2. **Incorporación del Parámetro k**: la ecuación de Perona-Malik es paramétrica del parámetro k que regula la sensibilidad de la difusión local. En este contexto, se propone que las PINNs aprendan una familia de soluciones para distintos valores de k, lo que implica incluir este parámetro como una entrada adicional en la red.

3. **Evaluación de PINNs en Casos Específicos**: se plantea investigar la eficacia de las PINNs en la aproximación de soluciones para imágenes específicas. Este análisis permitirá determinar la precisión de las PINNs en resolver la ecuación de Perona-Malik para condiciones iniciales concretas.

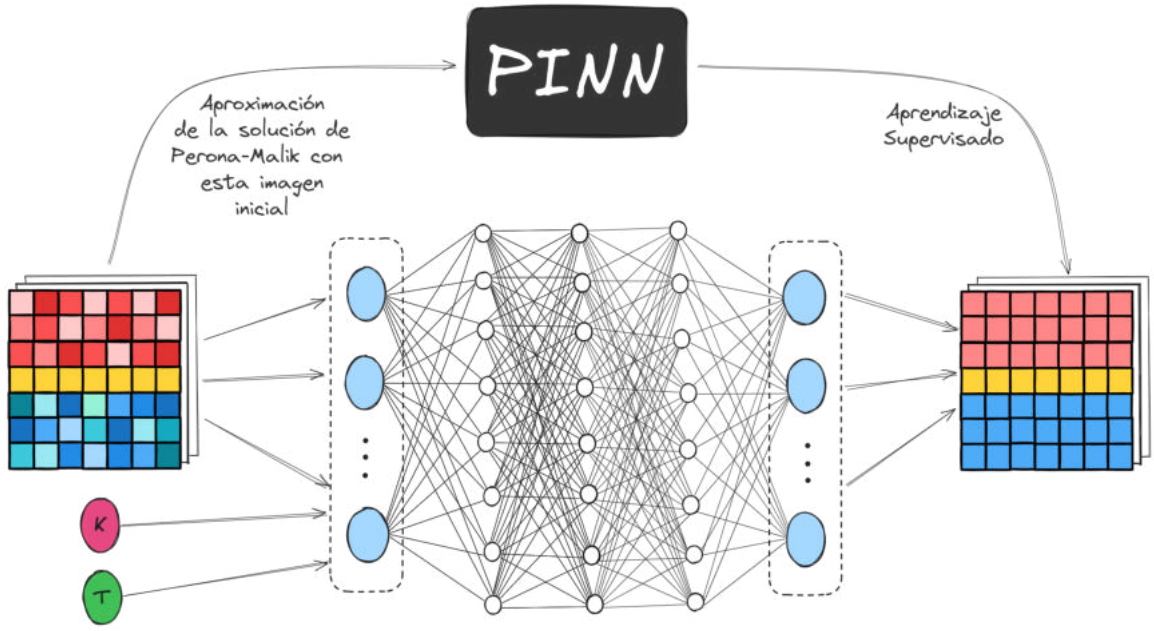
Fase 2: Desarrollo de un Modelo Basado en Aprendizaje Supervisado

1. **Construcción de un Dataset Sintético con PINNs**: se utiliza la capacidad de las PINNs para ajustarse a condiciones iniciales específicas, se generará un conjunto de datos sintético. Este conjunto de datos consistirá en imágenes procesadas por PINNs bajo diferentes condiciones iniciales, y crea una variedad de modelos preentrenados para diversos escenarios.

2. **Implementación de una Red Neuronal de Aprendizaje Supervisado**: a diferencia del enfoque no supervisado de las PINNs, se desarrollará una red neuronal basada en aprendizaje supervisado. Esta red recibirá de entrada una imagen junto con los parámetros k y t, y generará como salida la imagen transformada por el proceso de difusión anisotrópica correspondiente.

3. **Entrenamiento y Validación del Modelo**: el modelo se entrenará utilizando el conjunto de datos sintéticos creado por las PINNs. A través de este proceso, la red aprenderá a mapear imágenes con parámetros específicos a sus imágenes resultantes tras el proceso de difusión anisotrópica. Esto permitirá un aprendizaje más preciso y adaptable a una variedad de condiciones iniciales.

4. **Evaluación de la Efectividad del Modelo**: se examinará la capacidad de la red neuronal para aprender y generalizar el proceso de difusión anisotrópica en diferentes imágenes y parámetros, contrasta su rendimiento con los resultados obtenidos por las PINNs.



Esta propuesta busca abordar las limitaciones de las PINNs, en especial su incapacidad para generalizar a distintas condiciones iniciales, mediante un enfoque de aprendizaje supervisado. El modelo resultante se espera que sea capaz de aplicar el proceso de difusión anisotrópica de manera eficiente y generalizada, lo que ampliaría las aplicaciones potenciales en el campo del procesamiento avanzado de imágenes.

Resultados preliminares

Al abordar la ecuación de Perona-Malik utilizando las PINNs, se han enfrentado desafíos significativos en términos de precisión para la aproximación de condiciones iniciales y la adaptabilidad del modelo. Dada la complejidad de la función de difusión anisotrópica y la relevancia en el realce de bordes para el procesamiento de imágenes, el modelo requiere una implementación sofisticada que garantice eficiencia y precisión. Sin embargo, los resultados preliminares indican que las estrategias adoptadas, aunque prometedoras, no han logrado superar estas dificultades de manera satisfactoria; por este motivo se presenta a continuación las variaciones de la implementación base que se realizaron para combatir estos impedimentos.

1. **Ponderación de Errores de Condiciones Iniciales**: a pesar de incrementar la relevancia del error asociado a las condiciones iniciales en la función de pérdida, el modelo no logró capturar con la precisión deseada la imagen inicial, lo que indica una limitación en la adaptabilidad del modelo a variaciones específicas en las condiciones iniciales.
2. **Uso de Números Flotantes de Doble Precisión (64 bits)**: la implementación de mayor precisión en los cálculos, aunque útil teóricamente, no resultó en mejoras significativas en la exactitud del modelo, sugiere que los desafíos del modelo no se limitan a cuestiones de precisión numérica.

3. **Exploración de Diversas Arquitecturas de Red Neuronal**: el experimento con distintas configuraciones de red no proporcionó una mejora notable en la capacidad del modelo para capturar con precisión las condiciones iniciales, lo que subraya la complejidad inherente al problema.
4. **Implementación de Mecanismos de Regularización**: las técnicas de regularización aplicadas, como la norma L2 o Dropout, no contribuyeron de manera decisiva a evitar el sobreentrenamiento o mejorar la generalización del modelo.
5. **Selección de Funciones de Activación**: la experimentación con diferentes funciones de activación no generó una influencia positiva significativa en la habilidad de la red para modelar las dinámicas de la ecuación diferencial.
6. **Iniciativas de Inicialización de Parámetros**: aunque se implementaron técnicas de inicialización avanzadas, como He y Xavier [60, 61, 62], estas no resultaron en una mejora sustancial del proceso de aprendizaje.
7. **Experimentación con Diferentes Optimizadores**: la evaluación de distintos algoritmos de optimización tampoco llevó a una mejora en la efectividad del entrenamiento de la red.

Frente a estos desafíos, se propone una reducción de la complejidad del problema. Esta estrategia implica adoptar un enfoque similar al utilizado en la ecuación del calor en una dimensión, una difusión lineal en lugar de anisotrópica. Se espera que este enfoque simplificado facilite un análisis más detallado de las posibles causas de las limitaciones encontradas con las PINNs en la ecuación de Perona-Malik.

La implementación de estas estrategias tenía como objetivo principal superar las limitaciones identificadas en el uso de PINNs para la resolución de la Ecuación de Perona-Malik. Sin embargo, a pesar de los esfuerzos realizados, ninguna de las variaciones implementadas en la resolución de la ecuación logró los resultados esperados, lo que subraya la necesidad de replantear el enfoque o explorar métodos alternativos. El fracaso en superar estos desafíos no solo limita la aplicación eficiente de la difusión anisotrópica en el procesamiento de imágenes, sino que también plantea interrogantes sobre la generalización y la adaptabilidad de estas técnicas en el procesamiento de imágenes.

2.2. Reducción a la Ecuación de Difusión del Calor 1D

La Ecuación de Difusión del Calor en una dimensión (1D), central en este segmento de estudio, se erige como un modelo matemático esencial en el análisis de procesos de transferencia de calor. Se destaca en su simplicidad y aplicabilidad en una variedad de contextos físicos y de ingeniería, ofrece una comprensión fundamental del fenómeno de difusión térmica.

Esta ecuación, definida por su naturaleza lineal y predictiva, modela la distribución de la temperatura en un medio a lo largo del tiempo. La importancia radica en la capacidad para describir cómo se propaga el calor en diferentes materiales, lo cual es crucial para el diseño y análisis en campos como la termodinámica, la ingeniería de materiales y la física.

En su formulación matemática, la Ecuación de Difusión del Calor 1D se expresa como:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2}$$

donde $u(x,t)$ representa la temperatura en un punto espacial x y un tiempo t, y α es el coeficiente de difusión térmica, indicativo de la rapidez con que se difunde el calor en el medio.

La simplicidad de esta ecuación yace en el enfoque lineal y unidimensional, lo que facilita el análisis y solución. A diferencia de modelos más complejos, como la Ecuación de Perona-Malik en el procesamiento de imágenes, la ecuación de Difusión del Calor ofrece una aproximación directa y menos computacionalmente intensiva para entender la dinámica de la difusión.

En este estudio, se aborda la resolución de la Ecuación de Difusión del Calor 1D empleando las PINNs. El propósito es investigar la eficacia, precisión y adaptabilidad de estas técnicas en un escenario más simplificado, lo que puede proporcionar perspectivas valiosas para la aplicación en problemas más complejos. Este enfoque no solo contribuye a resolver la ecuación diferencial de manera eficiente, sino también amplía la comprensión sobre las capacidades y limitaciones de las PINNs. Sienta las bases para futuras aplicaciones en campos que requieren análisis de difusión térmica.

Modelación del problema

La ecuación del calor en una dimensión (1D) se establece como un modelo simplificado para investigar el desempeño de las PINNs. La ecuación del calor 1D se define matemáticamente como:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} \tag{2.5}$$

donde $u(x,t)$ representa la temperatura en una ubicación espacial x y un tiempo t, y α es un coeficiente de difusión térmica constante de un material.

Aplicación de PINNs para la Ecuación del Calor 1D

Para abordar esta ecuación diferencial parcial (EDP), se propone el uso de una red neuronal, denotada por $U_{NN}(x,t) \approx u(x,t)$. Esta red neuronal actuará como un aproximador universal de funciones, procesa un tensor bidimensional (x,t) de entrada y genera un valor escalar que estima la temperatura en el tiempo t.

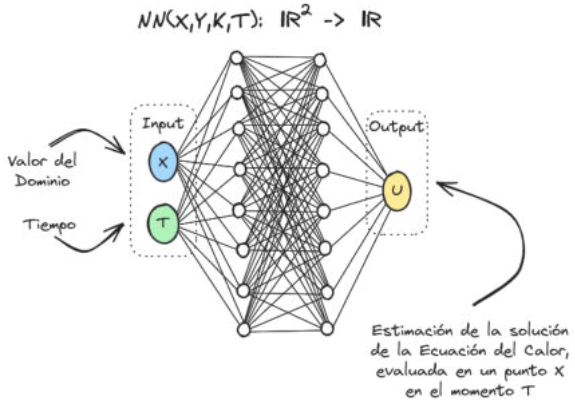


Figura 2.2: Representación esquemática de la aplicación de PINNs para la ecuación del calor 1D.

Definición del Dominio de Entrada

El dominio sobre el cual se entrena la red neuronal se define de la siguiente manera:

- La coordenada espacial x se encuentra en un intervalo adecuado, por ejemplo, $[0, L]$, donde L representa la longitud del dominio espacial.
- El tiempo t se considera en un intervalo, por ejemplo, $[0, T]$, donde T es el tiempo final del análisis.

Construcción de la Función de Pérdida

La función de pérdida para entrenar la red neuronal incluye tres componentes fundamentales:

1. **Error de las Condiciones Iniciales**:

$$Err_I = \frac{1}{|P_I|} \sum_{x \in D}^{|P_I|} \|u_0(x) - U_{NN}(x,0)\|^2 \tag{2.6}$$

   Este término mide la discrepancia entre la condición inicial de temperatura $u_0(x)$ y las predicciones de la red en $t = 0$.

2. **Error de las Condiciones de Frontera**:

$$Err_F = \frac{1}{|P_F|} \sum_{x \in D_F}^{|P_F|} U_{NN}(x,t)^2 \tag{2.7}$$

   Este término evalúa el cumplimiento de las condiciones de frontera, esencial para mantener la integridad de la solución a lo largo del dominio.

3. **Error Residual de la Ecuación Diferencial**:

$$Err_D = \frac{1}{|P_D|} \sum_{x \in D}^{|P_D|} \| \frac{\partial U_{NN}}{\partial t} - \alpha \frac{\partial^2 U_{NN}}{\partial x^2} \|^2 \tag{2.8}$$

   Este componente implica la incorporación de la información física en el entrenamiento de la red, y asegura que la solución aproximada obedezca la dinámica de la ecuación del calor.

Resultados preliminares

La aplicación de las PINNs a la Ecuación de Difusión del Calor en una dimensión ha arrojado resultados satisfactorios. A pesar de que la aproximación de la solución refleja con precisión el proceso físico subyacente, el ajuste a la condición inicial aún no alcanza el nivel de precisión deseado. Por tanto, se ha decidido continuar con la misma ecuación del calor, pero adoptando un enfoque diferente. El objetivo principal de este nuevo enfoque es examinar la eficacia de las PIDeepONet en la aproximación de las condiciones iniciales y la capacidad para aplicar el mismo modelo preentrenado a diferentes condiciones iniciales. El análisis comparativo de los resultados obtenidos con PINNs y PIDeepONet proporcionará una visión valiosa sobre las fortalezas y limitaciones de cada enfoque.

2.3. Resolución de la Ecuación del Calor 1D con PIDeepONet

Con la línea de investigación establecida en este epígrafe, se explorará una metodología alternativa para la resolución de la Ecuación del Calor 1D a través de la arquitectura innovadora de PIDeepONet. Este enfoque forma parte de un esfuerzo más amplio para evaluar el potencial de distintas arquitecturas de aprendizaje profundo en la resolución de PDEs, enfocándose en aquellas situaciones donde se requiere alta precisión y eficiencia computacional.

La PIDeepONet se propone como una solución a los retos encontrados en las PINNs, en términos de generalización a diversas condiciones iniciales y la gestión eficiente de los recursos computacionales. Conocida por su habilidad para aprender representaciones funcionales complejas, PIDeepONet promete un enfoque alternativo para abordar PDEs desde una nueva perspectiva, con expectativas de resultados precisos y adaptables.

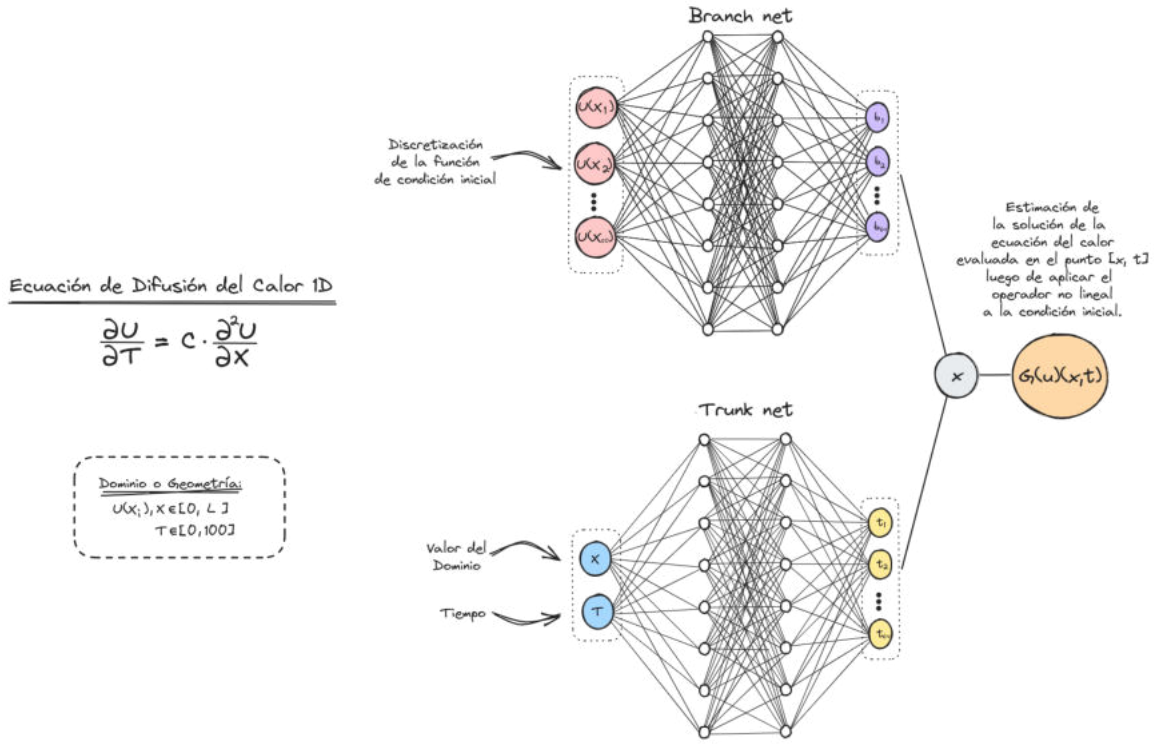


Figura 2.3: Representación esquemática de la arquitectura PIDeepONet aplicada a la ecuación del calor 1D.

En este epígrafe, se detalla la implementación de PIDeepONet para abordar la Ecuación de Difusión del Calor 1D. La descripción se centra en la arquitectura de la red neuronal (ver figura 2.3), dado que los aspectos relacionados con los puntos de colocación y la construcción de las funciones de error, orientadas por principios físicos, siguen una metodología similar a la expuesta en el epígrafe anterior sobre las PINNs.

El enfoque aquí proporciona una solución efectiva para la Ecuación del Calor 1D y, al mismo tiempo, aporta conocimientos aplicables a otros problemas complejos de PDEs mediante el uso de

técnicas avanzadas de aprendizaje automático. Este estudio representa un paso significativo en la comprensión y aplicación de arquitecturas de aprendizaje profundo como PIDeepONet en el ámbito del análisis numérico y la simulación de procesos físicos.

Capítulo 3

Resultados y experimentación

En este capítulo se presentan los resultados de una serie de experimentos centrados en la aplicación de PINNs y la arquitectura DeepONet para resolver la ecuación de Perona-Malik y la ecuación de Difusión del Calor en una dimensión. El primer experimento aborda la modelación de la ecuación de Perona-Malik mediante PINNs, analiza cómo el modelo replica las condiciones iniciales y su comportamiento a través del tiempo. El segundo experimento profundiza en la ecuación de Difusión del Calor 1D, para examinar en detalles la capacidad del modelo de ajustarse a variadas condiciones iniciales, resalta la importancia de un adecuado muestreo y de la selección de una arquitectura de red pertinente. El tercer experimento introduce la arquitectura DeepONet, combinándola con el enfoque no supervisado de las PINNs. Este experimento es esencial para evaluar la efectividad de DeepONet en la aproximación a condiciones iniciales diversas y la habilidad para aplicar el mismo modelo preentrenado a diferentes escenarios sin la necesidad de reentrenamiento.

En los experimentos realizados, se utilizó la librería DeepXDE que contribuyó a una comprensión más profunda de las capacidades y limitaciones de las PINNs y las DeepONet en la solución de ecuaciones diferenciales parciales. Los resultados obtenidos ofrecen perspectivas valiosas sobre la aplicabilidad de estas técnicas avanzadas en la modelización de procesos físicos, abre caminos para futuras investigaciones en el ámbito del aprendizaje profundo aplicado a problemas complejos de física e ingeniería.

3.1. Evaluación de PINNs en la Ecuación de Perona-Malik

La imagen, como una discretización de una función bidimensional, requiere de una aproximación continua para permitir un muestreo libre en el espacio definido. Se empleó el método de Funciones de Base Radial (RBF) de la librería Scipy en Python, del cual se obtiene una aproximación precisa mostrada en la figura 3.1. Esta función sirve como condición inicial en la PDE a resolver. Sin embargo, la predicción del modelo en el tiempo $T = 0$, para distintos valores de K, revela discrepancias notables, tal como se observa en la figura 3.2. Se evidencia la falta de unicidad en los valores resultantes para diferentes K y un alejamiento de las condiciones iniciales predichas por el modelo con respecto a las esperadas.

Al seleccionar $K = 2$ y analizar la evolución del modelo a lo largo del tiempo, se observan diferencias significativas entre los instantes $T = 0$ y $T = 1$, así como en momentos posteriores. Estas diferencias indican una inadecuada adaptación del modelo a la solución de la PDE y a las condiciones iniciales, ilustrado en la figura 3.3.

Este comportamiento sugiere una limitación en la densidad de puntos utilizados en la fase de

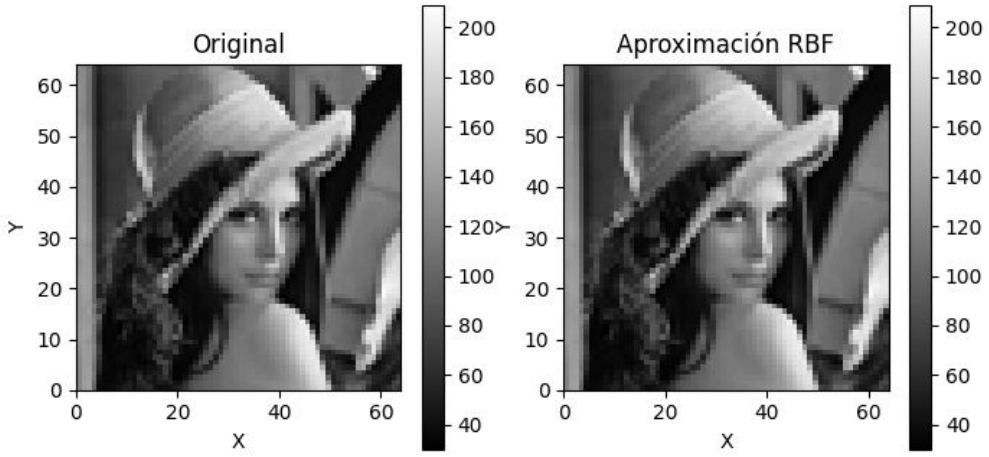


Figura 3.1: Foto original y aproximación RBF de la imagen inicial

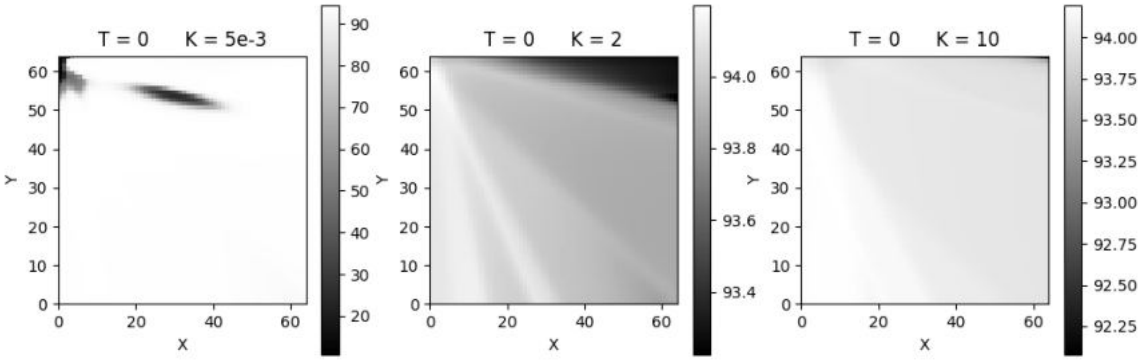


Figura 3.2: Predicción del modelo para $T = 0$ y diferentes valores de K

entrenamiento. Con aproximadamente 12,000 puntos muestreados para las condiciones iniciales y tiene en cuenta que para cada valor de K en el rango $[1e{-}3, 10]$ se deben ajustar 4096 puntos para cada pixel $[x, y]$, resulta la necesidad de un muestreo más amplio. Esta restricción afecta también a los puntos internos del dominio, donde por cada valor de T y K se deberían ajustar 4096 puntos, pero sólo se muestrean 16,000 en total durante el entrenamiento.

La implementación actual en la librería DeepXDE limita el muestreo a los puntos mencionados, cargándolos en la GPU al inicio del entrenamiento y utiliza el lote completo en cada época. Una solución potencial sería permitir un ajuste adecuado del `batch_size` y generar nuevos puntos en cada época para el ajuste de los parámetros de la red.

Al analizar la función de pérdida, mostrada en la figura 3.4, se observa que después de 5000 épocas, el conjunto de entrenamiento no mejora significativamente. Esto sugiere que la arquitectura de la red neuronal podría ser demasiado simple para el proceso que se intenta modelar. Para validar

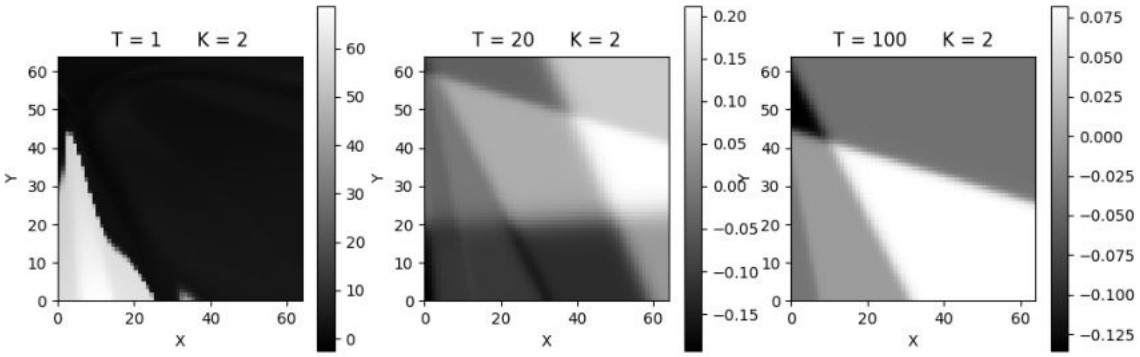


Figura 3.3: Predicción del modelo para $K = 2$ y diferentes valores de T

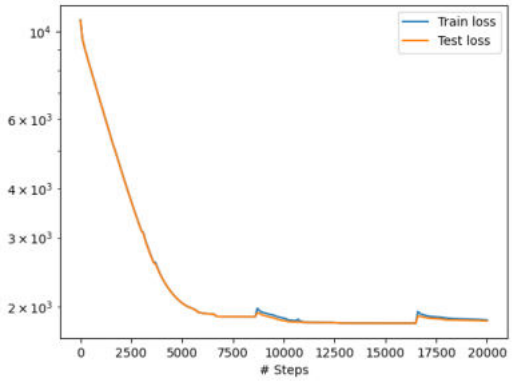


Figura 3.4: Función de pérdida

este planteamiento, se propone experimentar con una ecuación diferencial más sencilla, como la ecuación del calor 1D, que permita un muestreo adecuado y un ajuste efectivo de la red.

3.1.1. Detalles de implementación

Para el muestreo de datos, se seleccionaron 16,000 puntos internos del dominio, 1,000 puntos de frontera, y 12,288 puntos correspondientes a las condiciones iniciales. Durante cada época de entrenamiento, se eligen aleatoriamente 15,000 puntos para la validación.

La arquitectura de la red neuronal consta de 8 capas ocultas, y utiliza la función de activación tangente hiperbólica. Para el proceso de optimización utiliza el optimizador Adam con una tasa de aprendizaje de 10^{-3} y se ejecuta durante 20,000 iteraciones.

La implementación de la ecuación de Perona-Malik y la estructura de la red se detallan en el código proporcionado. Se define la función residual de la PDE de Perona-Malik, que es fundamental para calcular la función de pérdida durante el entrenamiento. Las derivadas parciales y las condiciones de difusión necesarias se obtienen mediante diferenciación automática proporcionada por DeepXDE.

3.2. Experimentación de la solución de la Ecuación de Difusión del Calor 1D

Este análisis se concentra en la capacidad del modelo para predecir la evolución de la temperatura a lo largo del tiempo, parte de una condición inicial definida.

Se estableció como condición inicial la función

$$f(x) = 0{,}5\sin\left(\frac{2\pi x}{5}\right) + 0{,}3\sin(\pi x) + 0{,}2\sin\left(\frac{8\pi x}{5}\right)$$

.

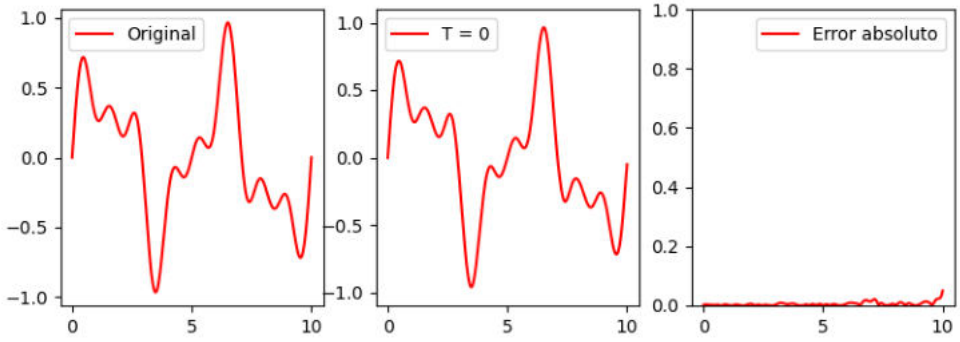


Figura 3.5: Función de condición inicial, predicción del modelo ($T = 0$) y función de error absoluto (de izquierda a derecha)

La representación gráfica de esta función (figura 3.5), comparada con la predicción del modelo en $T = 0$, muestra una notable precisión en la aproximación inicial del modelo. El análisis del error absoluto entre la función original y la predicción del modelo indica una estrecha correlación, confirma la eficacia del modelo en replicar la condición inicial.

Posteriormente, se examinó la habilidad del modelo para predecir la evolución temporal del fenómeno de difusión del calor. Las predicciones para distintos momentos ($T = 2, 10, 20, 100$) en la figura 3.6 revelan que el modelo aprende de manera efectiva el proceso físico descrito por la ecuación diferencial. Se observa cómo la temperatura se difumina progresivamente, y alcanza una temperatura unánime en todo el dominio al instante $T = 100$.

La gráfica 3.7 de la función de pérdida muestra una disminución global del error a lo largo de las iteraciones. Esto sugiere que extender el número de épocas en el entrenamiento podría mejorar aún más la precisión del modelo. Sin embargo, es importante destacar que la función de condición inicial utilizada en este experimento es relativamente simple, con variaciones limitadas de monotonía en pequeños intervalos del dominio.

Para futuros experimentos, se propone emplear funciones de condición inicial más complejas, como imágenes, para evaluar la capacidad de las PINNs en ajustar condiciones iniciales de mayor complejidad. Este enfoque permitirá una comprensión más profunda de la versatilidad y las limitaciones del modelo en la simulación de procesos físicos mediante ecuaciones diferenciales.

Complejización de la condición inicial

La condición inicial para la ecuación diferencial estuvo definida por una función notablemente compleja, expresada como:

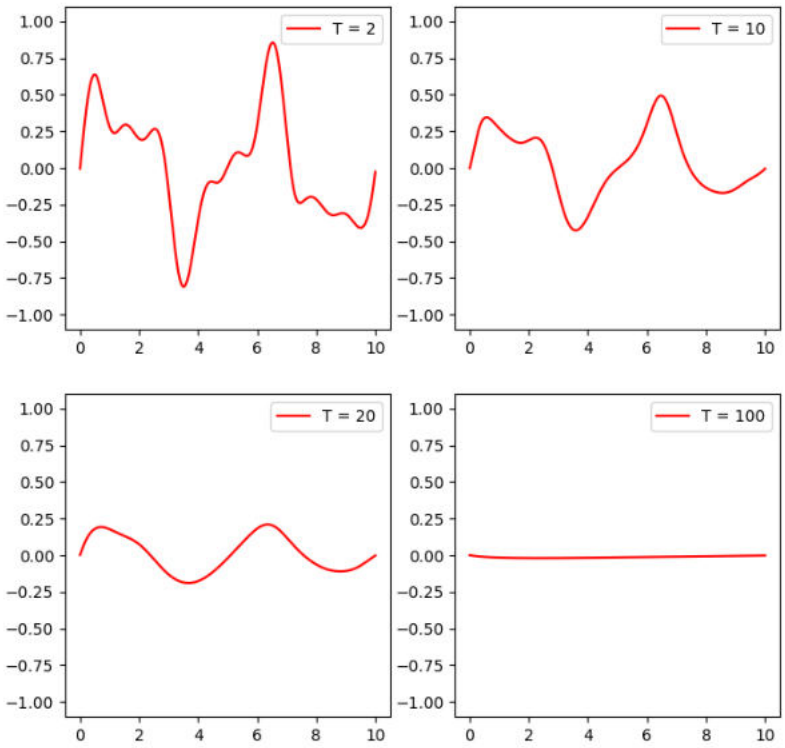


Figura 3.6: Predicción del modelo, para diferentes parámetros T

$$\begin{aligned} f(x) = 0{,}5\sin\left(\frac{2\pi x}{5}\right) + 0{,}3\sin(\pi x) + 0{,}2\sin\left(\frac{8\pi x}{5}\right) \\ + 0{,}1\sin(20\pi x) + 0{,}05\sin(40\pi x) \\ + 0{,}1\sin(60\pi x) + 0{,}2\sin(80\pi x) \\ + 0{,}3\sin(100\pi x). \end{aligned}$$

La complejidad de esta función, caracterizada por numerosos cambios de monotonía en intervalos reducidos del dominio, presentó un reto significativo para el modelo predictivo. Como se evidencia en las representaciones gráficas de la figura 3.8, la predicción del modelo no logró un ajuste preciso con la condición inicial, se revelan discrepancias notables en el patrón de error absoluto (figura 3.9).

Este fenómeno es análogo al desafío que representa aprender la función correspondiente a una imagen. Dicha función, al estar definida en dos dimensiones, implica una mayor cantidad de puntos a representar en comparación con funciones unidimensionales, aumenta así la complejidad en el aprendizaje del modelo.

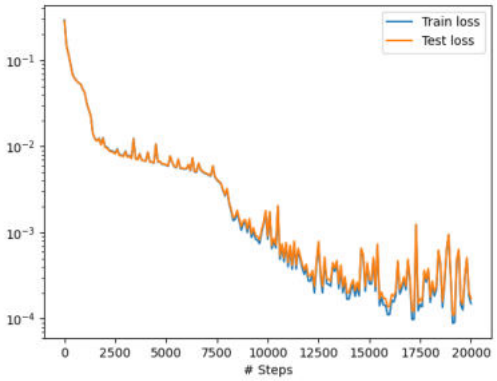


Figura 3.7: Función de pérdida

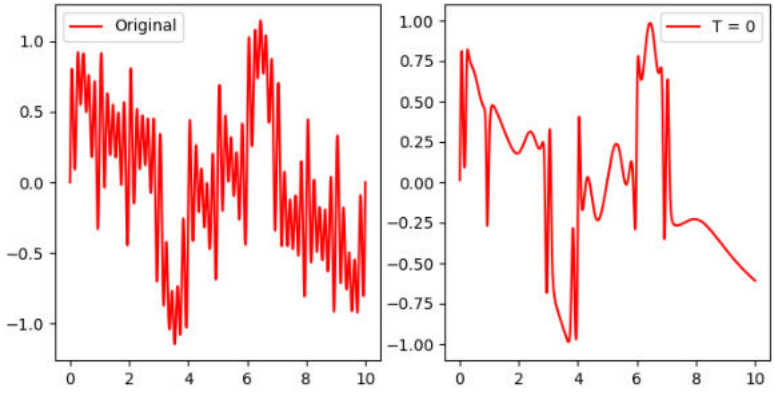


Figura 3.8: Condición inicial y predicción del modelo en $T = 0$ (de izquierda a derecha)

A pesar de las limitaciones en la precisión inicial, se observó en la figura 3.10 que el proceso de difusión lineal se ejecutó correctamente. La gráfica 3.11 de la función de pérdida mostró una tendencia decreciente, e indica la potencialidad para una reducción continua en el error de la función.

Incremento de la densidad de puntos

Se experimentó también con un incremento del doble de la densidad de puntos, tanto en la condición inicial como en los puntos frontera e internos. Los resultados obtenidos muestran una similitud con los anteriores (figura 3.12), esto evidencia un comportamiento similar tanto en el error absoluto para predecir la condición inicial (figura 3.13), como en los puntos internos del dominio, lo que hace cumplir correctamente el proceso de difusión del calor.

Este conjunto de experimentaciones y observaciones subraya la capacidad de las PINNs para abordar problemas complejos de difusión, a la vez que enfatiza la necesidad de continuar refinando las técnicas para mejorar la precisión en condiciones iniciales complejas.

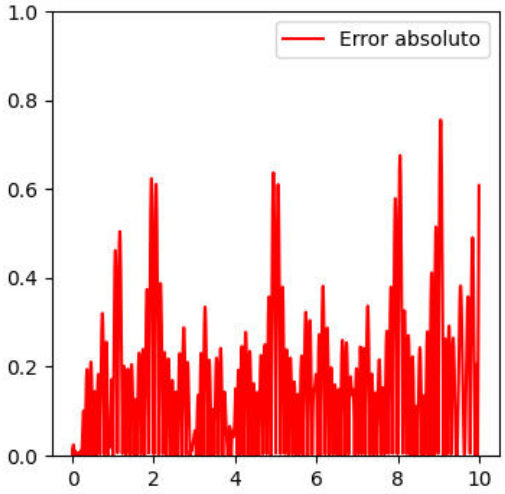


Figura 3.9: Función del error absoluto de la condición inicial respecto a la predicción del modelo en $T = 0$

Ajuste en la Ponderación de las Condiciones Iniciales

En esta variante del experimento, se incrementó significativamente el peso asignado al error de las condiciones iniciales, multiplicándolo por 1000, con el objetivo de forzar una mayor precisión en estas condiciones. Este ajuste se refleja en la figura 3.14, donde se muestra una mejora notable en la función del error absoluto, la cual presenta una suma de errores similar a la primera variante de este experimento, como se observa en la figura 3.15.

Sin embargo, al analizar las predicciones del modelo en diferentes instantes de tiempo, se percibe una coherencia reducida en el proceso de difusión del calor, en comparación con experimentos previos. Esta observación sugiere que la mayor ponderación otorgada a las condiciones iniciales pudo haber restado importancia al error de los puntos internos del dominio, lo que afectó negativamente la precisión de la solución en estos puntos. Esta idea se ve sustentada por las representaciones gráficas de la figura 3.16.

Además, la gráfica 3.17 de la función de pérdida corrobora esta hipótesis. A pesar de mostrar valores diferentes a los experimentos anteriores, se observa una variación significativa atribuible al factor de 1000 aplicado al error de las condiciones iniciales, manteniendo un comportamiento global similar.

En conclusión, el incremento en el peso del error de las condiciones iniciales no logró disminuir efectivamente dicho error; en cambio, provocó un desequilibrio con los otros componentes del error total, lo que resultó una peor aproximación de la solución global. Esta observación subraya la importancia de un balance adecuado entre los diferentes errores que constituyen la función de pérdida total en las PINNs.

Incremento del Número de Épocas en el Entrenamiento

En una variante de este experimento, se incrementó significativamente el número de iteraciones, llevándolo a 100,000 épocas, a diferencia de las 20,000 épocas utilizadas en los experimentos ante-

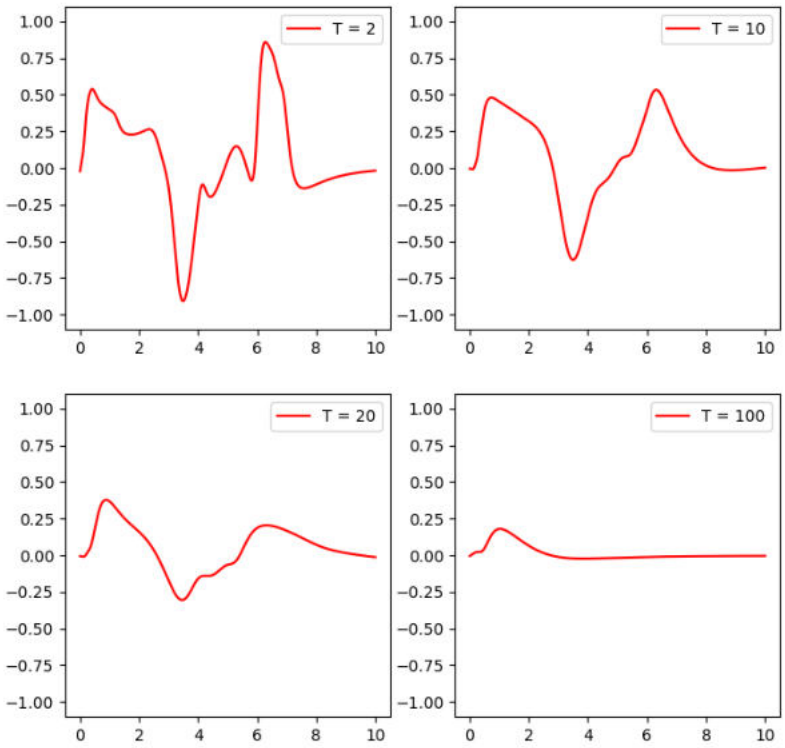


Figura 3.10: Predicción del modelo, en distintos instantes de tiempo T

riores. Este cambio se realizó con el fin de observar el impacto de un entrenamiento más prolongado en la precisión del modelo. Los resultados de este ajuste son evidentes en la figura 3.18, donde se muestra una mejora en la aproximación de la condición inicial, con un incremento de más del 20 % en la precisión, según la función de error absoluto representada en la figura 3.19.

Además, se observó que el proceso de difusión del calor se ejecutaba de manera más coherente, y se logra una aproximación más precisa de la ecuación diferencial, como se puede apreciar en la figura 3.20. La gráfica 3.21 de la función de pérdida muestra una tendencia decreciente a nivel global, aunque con un ritmo de decrecimiento cada vez menos pronunciado a medida que avanza el número de épocas.

Este experimento ampliado proporciona una visión valiosa sobre la complejidad que conlleva el proceso de optimización en las PINNs cuando se enfrentan a condiciones iniciales complejas. La mejora en la precisión a través de un número aumentado de épocas destaca la importancia de la duración del entrenamiento en la capacidad del modelo para ajustarse a condiciones iniciales complejas y procesos físicos dinámicos.

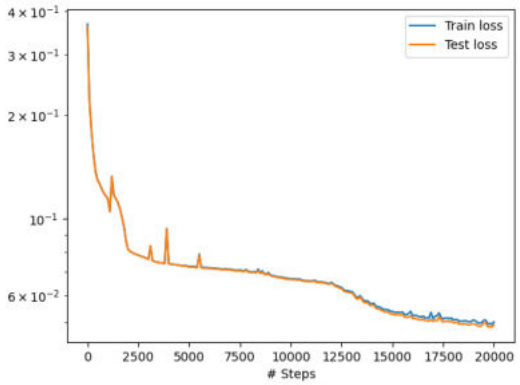


Figura 3.11: Función de pérdida

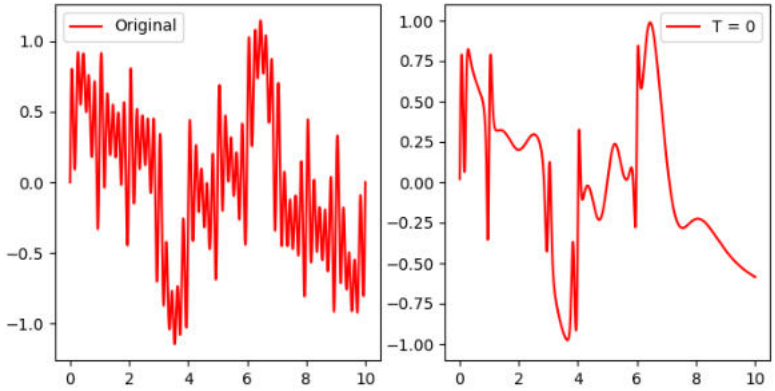


Figura 3.12: Predicción del modelo en $T = 0$, entrenado con una mayor densidad de puntos

3.2.1. Detalles de implementación

En cuanto al muestreo de datos, de manera general en los experimentos se seleccionaron 254,000 puntos internos del dominio, 20,000 puntos de frontera, y 100,000 puntos correspondientes a las condiciones iniciales; excepto en un caso que se duplicó la densidad de puntos en cada tipo. Durante cada época de entrenamiento, se eligen aleatoriamente 15,000 puntos como conjunto de validación.

La arquitectura de la red neuronal consta de 8 capas ocultas con 20 neuronas cada una, y utiliza la función de activación tangente hiperbólica. Para el proceso de optimización se utilizó el optimizador Adam con una tasa de aprendizaje de 10^{-3} y se ejecutó durante 20,000 iteraciones en la mayoría de los experimentos a excepción de uno que se corrió por 100,000 iteraciones.

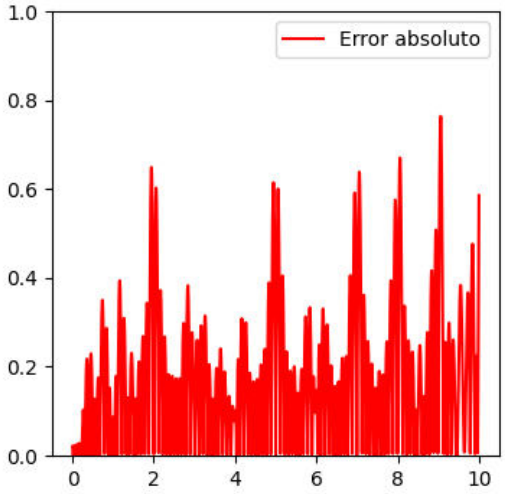


Figura 3.13: Error absoluto

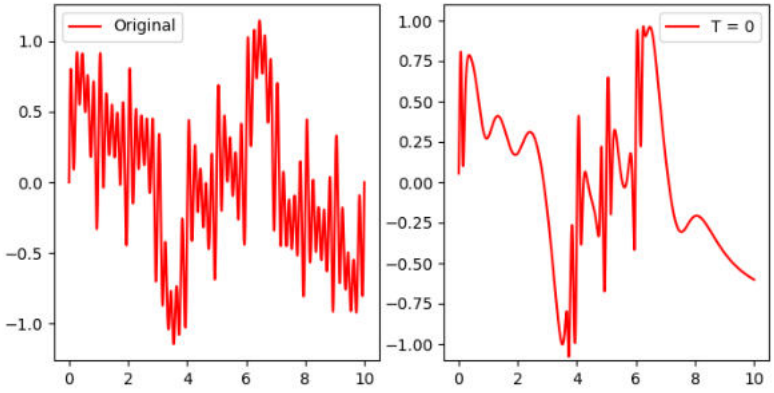


Figura 3.14: Condición inicial y predicción del modelo en $T = 0$

3.3. Integración de DeepONet en la Modelización de la Ecuación de Difusión del Calor

Este experimento incorpora la arquitectura DeepONet integrando el enfoque no supervisado de las PINNs. Los resultados obtenidos han sido satisfactorios, aunque la estimación de las condiciones iniciales es un desafío debido al limitado conjunto de datos utilizados para el entrenamiento de la red. En la figura 3.22, se muestra la diferencia entre la condición inicial y la predicción del modelo para $T = 0$, lo que evidencia la capacidad de la red para aproximarse a la condición inicial dada.

En cuanto al comportamiento del modelo frente al proceso de difusión del calor, se observa un

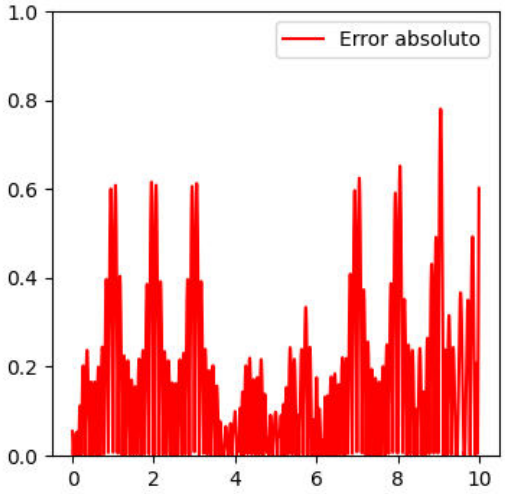


Figura 3.15: Función del error absoluto

desempeño adecuado. La figura 3.23 ilustra cómo, dada una condición inicial en la fila $Y = 0$, la intensidad del calor se difumina a lo largo del eje X con el aumento de Y, alcanzando un estado de equilibrio en $Y = 50$. Esta representación muestra la efectividad del modelo en simular la dinámica de difusión.

A pesar de que la predicción de la condición inicial no es tan precisa como en el enfoque anterior con una PINNs, la arquitectura DeepONet ofrece la ventaja de aproximar la solución de la ecuación diferencial para una variedad de condiciones iniciales. Esto representa un avance significativo con respecto al enfoque anterior, ya que ofrece una mayor flexibilidad y capacidad de generalización en la modelización de procesos dinámicos complejos.

En la gráfica 3.24 se muestra en la función de pérdida, se puede apreciar el sobreentrenamiento que experimenta la red al tener tan pocos datos, y entrenarlo durante tantas iteraciones. Las otras figuras que fueron mostradas en esta sección se obtuvieron con el modelo preentrenado con 10,000 épocas.

3.3.1. Detalles de implementación

Para la captura de la condición inicial, se elige un enfoque que involucra la utilización de 50 sensores equidistantes. Estos sensores captan la información necesaria para definir el estado inicial del sistema modelado por la ecuación de Difusión del Calor 1D. La dimensión del dominio de la función resultante, tras aplicar el operador no lineal, es 2, que incluye X como valor del dominio de la función, y T que representa el tiempo.

Respecto al muestreo de los puntos de colocación se utilizaron 7,500 puntos interiores del dominio, 3,000 puntos de condiciones iniciales, 1,000 puntos de frontera y 500 puntos del conjunto de validación. Se utilizaron además 1000 funciones diferentes para el ajuste con los puntos de colocación, generadas con el espacio de funciones de Campo Gausiano Aleatorio.

La arquitectura de la red neuronal empleada para abordar este problema se divide en dos subredes principales: la subred Branch y la subred Trunk. La subred Branch está diseñada para recibir un

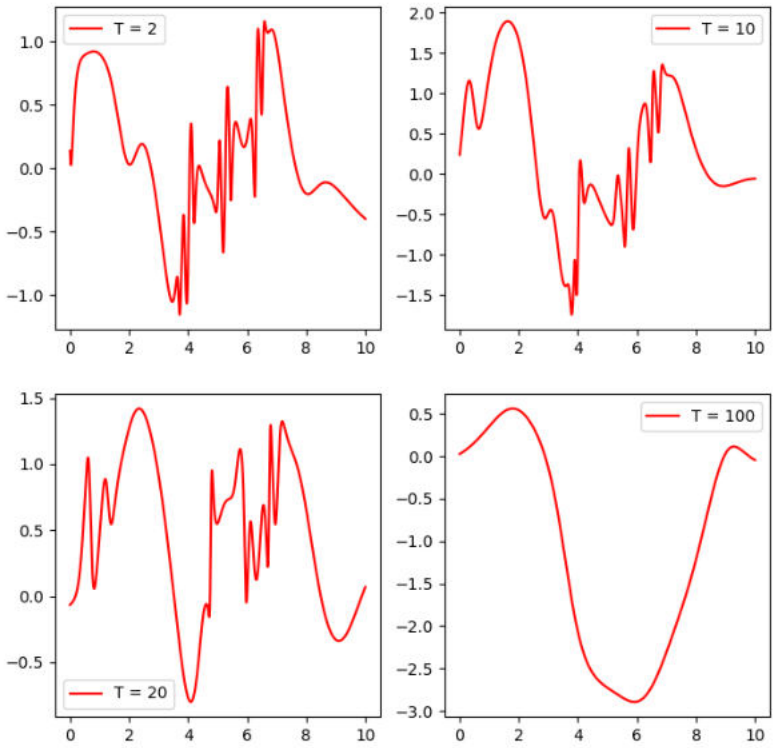


Figura 3.16: Inconsistencias en la predicción del proceso de difusión

tensor de tamaño 50 y consta de 6 capas ocultas, cada una con 64 neuronas. Por otro lado, la subred Trunk, que procesa un tensor de entrada de tamaño 2, también cuenta con 6 capas ocultas de 64 neuronas. Para ambas subredes, se selecciona la función de activación Tangente Hiperbólica por su efectividad en las PINNs.

Para el proceso de optimización se utiliza el optimizador Adam con un learning rate de 1×10^{-3} y se entrenó el modelo por 100,000 iteraciones.

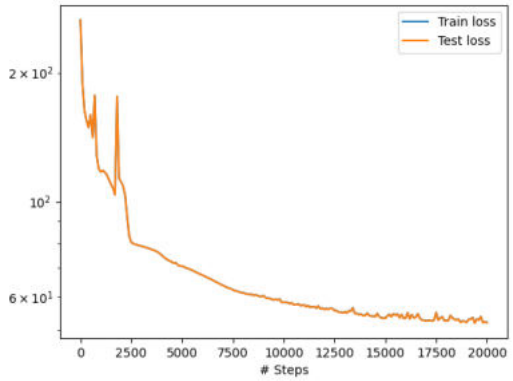


Figura 3.17: Función de pérdida ajustada

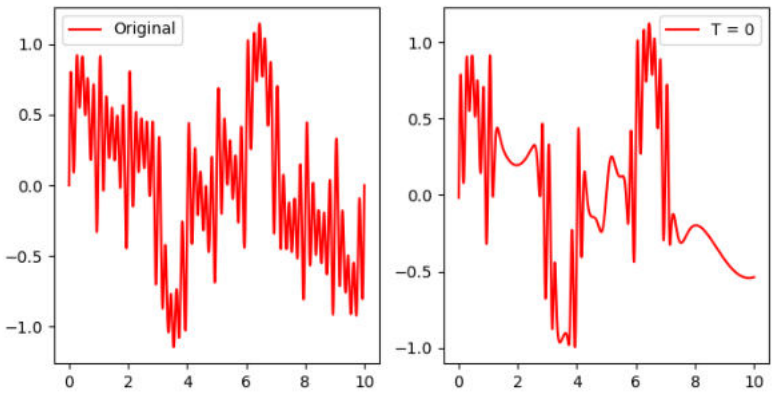


Figura 3.18: Mejora en la condición inicial con 100 mil épocas

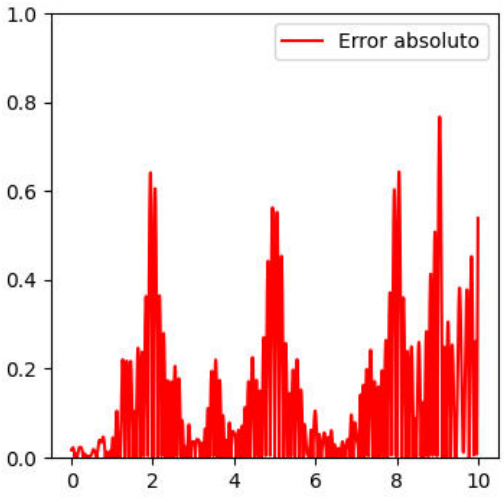


Figura 3.19: Función del error absoluto con entrenamiento extendido

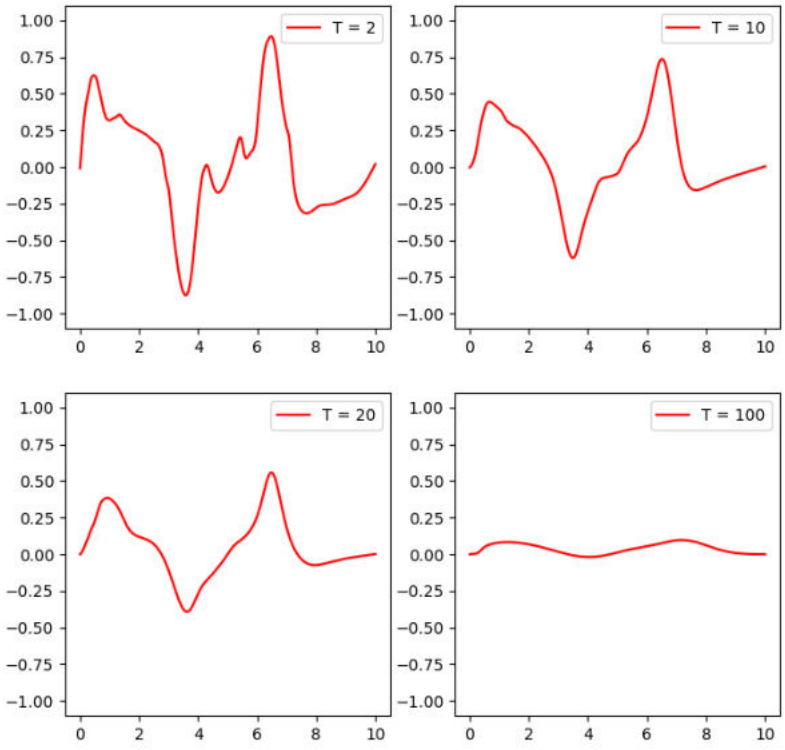


Figura 3.20: Aproximación del proceso de difusión del calor con mayor número de épocas

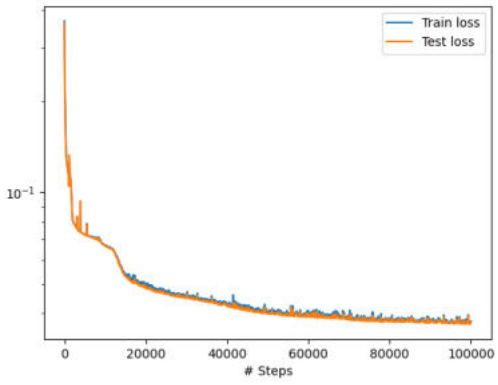


Figura 3.21: Función de pérdida durante 100 mil épocas

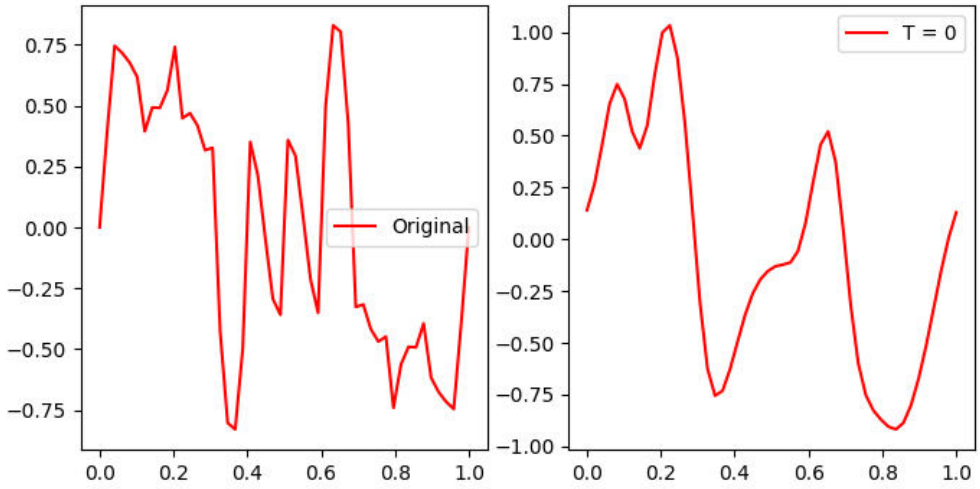


Figura 3.22: Comparación entre la condición inicial y la predicción del modelo en $T = 0$

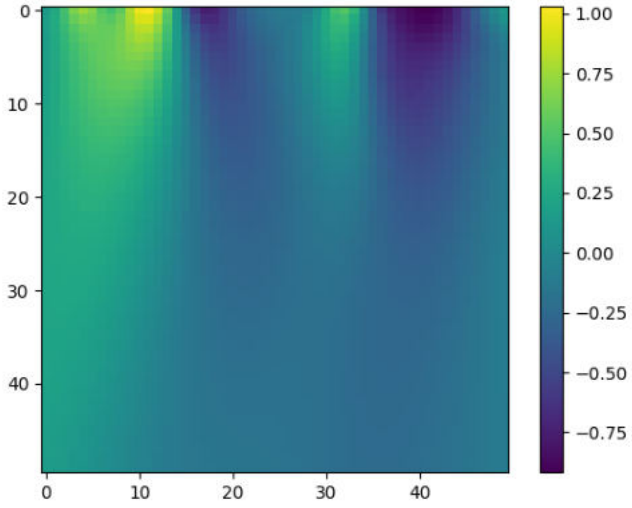


Figura 3.23: Comportamiento del modelo en el proceso de difusión del calor

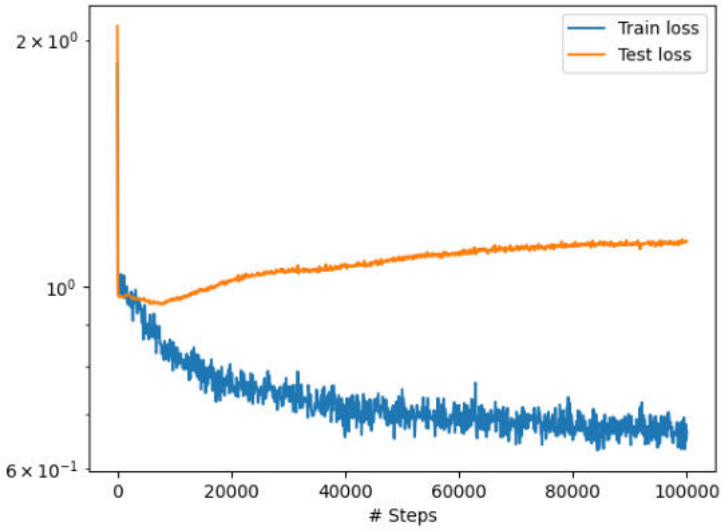


Figura 3.24: Función de pérdida

Conclusiones

En este trabajo se abordaron los desafíos inherentes a la aproximación de la solución de ecuaciones diferenciales complejas utilizando técnicas avanzadas de DL. Se exploraron dos enfoques principales: las PINNs y las DeepONet, con el objetivo de adaptar modelos preentrenados, flexibles y eficientes a variaciones en los parámetros de la ecuación diferencial de Perona-Malik.

El marco teórico que sustenta las técnicas de DL aplicadas se centró en la incorporación de principios físicos en el aprendizaje no supervisado de la solución de ecuaciones diferenciales mediante NNs. Este enfoque se basó en el uso de PINNs y se complementó con la arquitectura DeepONet, que destaca por prescindir de un dataset predefinido, desafío significativo en el campo del ML.

En cuanto a la estructuración de los modelos preentrenados, el primer enfoque, centrado en la ecuación de Perona-Malik, no se logró implementar completo debido a la incapacidad de las PINNs para obtener soluciones exitosas con una imagen como condición inicial. Esto impidió la construcción del dataset sintético necesario para entrenar una segunda red en el proceso de difusión anisotrópica. El segundo enfoque se dirigió hacia el aprendizaje de operadores no lineales mediante DeepONet. Este modelo, a diferencia del anterior, no dependía de forma explícita de una condición inicial específica y era capaz de aprender la operación aplicada a la condición inicial, estima un valor para una configuración de entrada dada. Aunque se aplicó a la ecuación del calor debido a limitaciones de recursos computacionales, este enfoque abre puertas para futuras investigaciones en la aplicación a la ecuación de Perona-Malik.

Los resultados obtenidos proporcionan una visión crítica sobre las limitaciones de las PINNs en su enfoque clásico para resolver la ecuación de Perona-Malik y subrayan la importancia de un muestreo representativo del dominio de la ecuación diferencial. Se observó que una ponderación excesiva de las condiciones iniciales puede, en algunos casos, deteriorar la solución de la ecuación y causar efectos adversos. Esta investigación resalta la necesidad de un equilibrio adecuado en la ponderación de errores y ofrece un análisis detallado de las fortalezas y debilidades de las PINNs y DeepONet en el contexto de la solución de ecuaciones diferenciales complejas.

Recomendaciones

A partir de los hallazgos y resultados de esta investigación, se proponen las siguientes recomendaciones para futuros trabajos en el campo del Aprendizaje Profundo aplicado a ecuaciones diferenciales complejas:

1. Se sugiere continuar la investigación en el ámbito de las DeepONet orientadas por la física, en específico para la resolución del modelo de Perona-Malik. Dado el potencial demostrado por DeepONet en este estudio, su aplicación en el modelo de Perona-Malik podría ofrecer nuevas perspectivas y avances significativos en el entendimiento y resolución de ecuaciones diferenciales complejas mediante técnicas de aprendizaje profundo.

2. Se recomienda extender la librería DeepXDE, o explorar alternativas como NeuralPDE.jl o Modulus, para permitir el entrenamiento con un conjunto más amplio de puntos de colocación sin la necesidad de almacenar todos ellos en tiempo de ejecución. Aunque esta modificación puede ralentizar el proceso de optimización, permitirá un análisis más robusto y exhaustivo del dominio definido por las ecuaciones diferenciales, para mejorar la precisión y generalización del modelo.

3. Se aconseja experimentar nuevas metodologías para modelar el proceso de difusión anisotrópica. Una posibilidad sería el desarrollo de un modelo basado en aprendizaje supervisado, que utilice un conjunto de datos construído mediante la aplicación de métodos numéricos como el Método de Diferencias Finitas. Este enfoque podría incluir el uso de técnicas más tradicionales de DL, como las Redes Neuronales Convolucionales (CNN), para aprender y replicar el proceso de difusión anisotrópica.

Estas recomendaciones buscan abordar las limitaciones identificadas en esta investigación, en lo que respecta a la capacidad de las PINNs y DeepONet para resolver con precisión ecuaciones diferenciales complejas. Al seguir estas sugerencias, futuros estudios podrían superar los desafíos actuales, avanzando así en el campo del aprendizaje profundo y su aplicación en problemas científicos y de ingeniería de alto impacto.

Bibliografía

[1] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations, November 2017. arXiv:1711.10561 [cs, math, stat]. (Citado en las páginas 1, 3, 8 y 11).

[2] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations, November 2017. arXiv:1711.10566 [cs, math, stat]. (Citado en las páginas 1, 3, 8 y 11).

[3] Raissi et al. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations - ScienceDirect, 2019. (Citado en las páginas 1, 8 y 11).

[4] Weinan E and Bing Yu. The Deep Ritz Method: A Deep Learning-Based Numerical Algorithm for Solving Variational Problems. *Communications in Mathematics and Statistics*, 6(1):1–12, March 2018. (Citado en las páginas 1 y 8).

[5] Justin Sirignano and Konstantinos Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, December 2018. (Citado en las páginas 1 y 8).

[6] Lu Lu, Pengzhan Jin, and George Em Karniadakis. Deeponet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, March 2021. arXiv:1910.03193 [cs, stat]. (Citado en las páginas 1, 3, 7 y 13).

[7] Hanwen Wang Sifan Wang and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepONets. *SCIENCE ADVANCES*, 2021. (Citado en las páginas 1, 10 y 13).

[8] Aditi S. Krishnapriyan, Amir Gholami, Shandian Zhe, Robert M. Kirby, and Michael W. Mahoney. Characterizing possible failure modes in physics-informed neural networks, 2021. (Citado en la página 1).

[9] Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maizar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92, 2022. (Citado en las páginas 2 y 11).

[10] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990. (Citado en la página 2).

[11] R. Kent Nagle, Edward B. Saff, and Arthur David Snider. *Fundamentals of Differential Equations and Boundary Value Problems, 6th Edition*. Pearson Education, 2014. (Citado en la página 4).

[12] J.L. Goldberg and M.C. Potter. *Differential Equations: A Systems Approach*. Coming of age series. Prentice-Hall, 1998. (Citado en la página 5).

[13] S. S. Nourazar and A. Nazari-Golshan. A new modification to homotopy perturbation method combined with fourier transform for solving nonlinear cauchy reaction diffusion equation. *Indian Journal of Physics*, 89(1):61–71, 2015. (Citado en la página 5).

[14] Muhammad Imran Liaqat, Sina Etemad, Shahram Rezapour, and Choonkil Park. A novel analytical aboodh residual power series method for solving linear and nonlinear time-fractional partial differential equations with variable coefficients. *AIMS Mathematics*, 7(9):16917–16948, 2022. (Citado en la página 5).

[15] K. Cole, J. Beck, A. Haji-Sheikh, and B. Litkouhi. *Heat Conduction Using Green's Functions*. CRC Press, 2 edition, 2010. (Citado en la página 5).

[16] Muhammad Imran Liaqat and Eric Okyere. Comparative Analysis of the Time-Fractional Black–Scholes Option Pricing Equations (BSOPE) by the Laplace Residual Power Series Method (LRPSM). *Journal of Mathematics*, 2023:6092283, February 2023. Publisher: Hindawi. (Citado en la página 5).

[17] Mohammad Shqair, Emad A. M. Farrag, and Mohammed Al-Smadi. Solving multi-group reflected spherical reactor system of equations using the homotopy perturbation method. *Mathematics*, 10(10), 2022. (Citado en la página 5).

[18] Zdzisław Jackiewicz and S. Tracogna. A general class of two-step runge–kutta methods for ordinary differential equations. Oct 1995. (Citado en la página 5).

[19] Peter J. Olver. Applications of lie groups to differential equations. Sep 1990. (Citado en la página 5).

[20] Won Young Yang, Wenwu Cao, Tae Sang Chung, and John Morris. Applied numerical methods using matlab. Nov 2005. (Citado en la página 5).

[21] Hacı Mehmet Baskonus and Hasan Bulut. On the numerical solutions of some fractional ordinary differential equations by fractional adams-bashforth-moulton method. Sep 2015. (Citado en la página 5).

[22] George E. Forsythe, Wolfgang Wasow, and W. Nachbar. Finite-difference methods for partial differential equations. Apr 1961. (Citado en la página 5).

[23] Max Gunzburger, Clayton G. Webster, and Guannan Zhang. Stochastic finite element methods for partial differential equations with random input data. May 2014. (Citado en la página 5).

[24] Joaquim Peiró and Spencer J. Sherwin. Finite difference, finite element and finite volume methods for partial differential equations. Jan 2005. (Citado en la página 5).

[25] William E. Schiesser. The numerical method of lines: Integration of partial differential equations. Jan 1993. (Citado en la página 5).

[26] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. Numerical recipes in c: the art of scientific computing. Jun 1993. (Citado en la página 5).

[27] Abel Antonio Cruz Suárez. Difusión de epidemias sobre redes empleando movimiento lagrangiano, Noviembre 2022. (Citado en la página 6).

[28] Yuntian Chen, Yingtao Luo, Qiang Liu, Hao Xu, and Dongxiao Zhang. Any equation is a forest: Symbolic genetic algorithm for discovering open-form partial differential equations (SGA-PDE). *Physical Review Research*, 4(2):023174, June 2022. arXiv:2106.11927 [physics]. (Citado en la página 6).

[29] Hyuk Lee and In Seok Kang. Neural algorithm for solving differential equations. *Journal of Computational Physics*, 91(1):110–131, 1990. (Citado en la página 6).

[30] I.E. Lagaris, A. Likas, and D.I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998. (Citado en la página 6).

[31] Manoj Kumar and Neha Yadav. Multilayer perceptrons and radial basis function neural network methods for the solution of differential equations: A survey. *Computers & Mathematics with Applications*, 62(10):3796–3811, November 2011. (Citado en la página 6).

[32] Nick Winovich, Karthik Ramani, and Guang Lin. ConvPDE-UQ: Convolutional neural networks with quantified uncertainty for heterogeneous elliptic partial differential equations on varied domains. *Journal of Computational Physics*, 394:263–279, October 2019. (Citado en la página 6).

[33] Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. PDE-Net: Learning PDEs from Data, January 2018. arXiv:1710.09668 [cs, math, stat]. (Citado en la página 6).

[34] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, March 2021. (Citado en las páginas 7 y 13).

[35] Carl Leake, Hunter Johnston, Lidia Smith, and Daniele Mortari. Analytically Embedding Differential Equation Constraints into Least Squares Support Vector Machines using the Theory of Functional Connections. *Machine Learning and Knowledge Extraction*, 1(4):1058–1083, October 2019. arXiv:1812.05571 [cs, stat]. (Citado en la página 7).

[36] Houman Owhadi. Bayesian numerical homogenization. *Multiscale Modeling & Simulation*, 13(3):812–828, 2015. (Citado en la página 7).

[37] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Inferring solutions of differential equations using noisy multi-fidelity data. *Journal of Computational Physics*, 335:736–746, April 2017. (Citado en la página 7).

[38] Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Machine learning of linear differential equations using Gaussian processes. *Journal of Computational Physics*, 348:683–693, 2017. (Citado en la página 7).

[39] Yinhao Zhu, Nicholas Zabaras, Phaedon-Stelios Koutsourelakis, and Paris Perdikaris. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *Journal of Computational Physics*, 394:56–81, October 2019. arXiv:1901.06314 [physics, stat]. (Citado en la página 8).

[40] E. Kharazmi, Z. Zhang, and G. E. Karniadakis. Variational Physics-Informed Neural Networks For Solving Partial Differential Equations, November 2019. arXiv:1912.00873 [physics, stat]. (Citado en la página 9).

[41] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2020. (Citado en la página 9).

[42] Jialei Chen, Zhehui Chen, Chuck Zhang, and C. F. Jeff Wu. Apik: Active physics-informed kriging model with partial differential equations, December 2020. arXiv:2012.11798 [cs, stat]. (Citado en la página 9).

[43] Kailai Xu and Eric Darve. Solving inverse problems in stochastic models using deep neural networks and adversarial training. *Computer Methods in Applied Mechanics and Engineering*, 384:113976, 2021. (Citado en las páginas 10 y 16).

[44] Zhuojia Fu, Wenzhi Xu, and Shuainan Liu. Physics-Informed Kernel Function Neural Networks for Solving Partial Differential Equations, June 2023. arXiv:2306.02606 [cs, math]. (Citado en la página 10).

[45] Leo Zhiyuan Zhao, Xueying Ding, and B Aditya Prakash. Pinnsformer: A transformer-based framework for physics-informed neural networks. *arXiv preprint arXiv:2307.11833*, 2023. (Citado en la página 11).

[46] Sifan Wang, Hanwen Wang, and Paris Perdikaris. Learning the solution operator of parametric partial differential equations with physics-informed DeepOnets, March 2021. arXiv:2103.10974 [cs, math, stat]. (Citado en la página 13).

[47] Somdatta Goswami, Aniruddha Bora, Yue Yu, and George Em Karniadakis. Physics-Informed Deep Neural Operator Networks, July 2022. arXiv:2207.05748 [cs, math]. (Citado en la página 13).

[48] Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. Deepxde: A deep learning library for solving differential equations. *SIAM Review*, 63(1):208–228, 2021. (Citado en las páginas 15 y 16).

[49] Alexander Koryagin, Roman Khudorozkov, and Sergey Tsimfer. Pydens: A python framework for solving differential equations with neural networks. *arXiv preprint arXiv:1909.11544*, 2019. (Citado en la página 16).

[50] NVIDIA Corporation (2021). Modulus user guide. 2021. (Citado en la página 16).

[51] Feiyu Chen, David Sondak, Pavlos Protopapas, Marios Mattheakis, Shuheng Liu, Devansh Agarwal, and Marco Di Giovanni. Neurodiffeq: A python package for solving differential equations with neural networks. *Journal of Open Source Software*, 5(46):1931, 2020. (Citado en la página 16).

[52] Ehsan Haghighat and Ruben Juanes. Sciann: A keras/tensorflow wrapper for scientific computations and physics-informed deep learning using artificial neural networks. *Computer Methods in Applied Mechanics and Engineering*, 373:113552, 2021. (Citado en la página 16).

[53] Kirill Zubov, Zoe McCarthy, Yingbo Ma, Francesco Calisto, Valerio Pagliarino, Simone Azeglio, Luca Bottero, Emmanuel Luján, Valentin Sulzer, Ashutosh Bharambe, et al. Neuralpde: Automating physics-informed neural networks (pinns) with error approximations. *arXiv preprint arXiv:2107.09443*, 2021. (Citado en la página 16).

[54] Juan B Pedro, Juan Maroñas, and Roberto Paredes. Solving partial differential equations with neural networks. *arXiv preprint arXiv:1912.04737*, 2019. (Citado en la página 16).

[55] Levi D McClenny, Mulugeta A Haile, and Ulisses M Braga-Neto. Tensordiffeq: Scalable multi-gpu forward and inverse solvers for physics informed neural networks. *arXiv preprint arXiv:2103.16034*, 2021. (Citado en la página 16).

[56] Wei Peng, Jun Zhang, Weien Zhou, Xiaoyu Zhao, Wen Yao, and Xiaoqian Chen. Idrlnet: A physics-informed neural network library. *arXiv preprint arXiv:2107.04320*, 2021. (Citado en la página 16).

[57] Jack Y Araz, Juan Carlos Criado, and Michael Spannowsky. Elvet–a neural network-based differential equation and variational problem solver. *arXiv preprint arXiv:2103.14575*, 2021. (Citado en la página 16).

[58] Oliver Hennigh, Susheela Narasimhan, Mohammad Amin Nabian, Akshay Subramaniam, Kaustubh Tangsali, Zhiwei Fang, Max Rietmann, Wonmin Byeon, and Sanjay Choudhry. Nvidia simnet™: An ai-accelerated multi-physics simulation framework. In *International Conference on Computational Science*, pages 447–461. Springer, 2021. (Citado en la página 16).

[59] Kailai Xu and Eric Darve. Adcme: Learning spatially-varying physical fields using deep neural networks, 2020. (Citado en la página 16).

[60] Siddharth Krishna Kumar. On weight initialization in deep neural networks, May 2017. arXiv:1704.08863 [cs]. (Citado en la página 23).

[61] Justin Sirignano and Konstantinos Spiliopoulos. Scaling Limit of Neural Networks with the Xavier Initialization and Convergence to a Global Minimum, April 2022. arXiv:1907.04108 [cs, math, stat]. (Citado en la página 23).

[62] Leonid Datta. A Survey on Activation Functions and their relation with Xavier and He Normal Initialization, March 2020. arXiv:2004.06632 [cs]. (Citado en la página 23).