Universidad Europea VALENCIA

GRADO EN FÍSICA

Simulador cuántico ‘mesh-free’ al servicio de la nanoelectrónica: una nueva forma de resolver la ecuación de Schrödinger con redes neuronales

Presentado por:

EMIN LARBI BENDAHOUA ESPINOSA

Dirigido por:

HÉCTOR ESPINÓS MORATÓ

Curso Académico 2024–2025

Índice

1. Introducción **6**

2. Objetivos **7**

2.1. Objetivo principal 7

2.2. Objetivos secundarios 7

3. Contribución a los ODS y alineación con los pilares de la Universidad Europea **8**

4. Diseño experimental: Metodología CRISP-DM **9**

5. Temporalización **11**

6. Estado del Arte **12**

6.1. La ecuación de Schrödinger 12

6.2. Método Tradicionales de Resolución de la Ecuación de Schrödinger 13

6.2.1. Método de Diferencias Finitas 13

6.2.2. Método de Elementos Finitos 14

6.2.3. Métodos Espectrales 16

6.2.4. Comparación General de los Métodos 18

6.3. Propuesta de solución con redes neuronales 19

6.4. Physics-Informed Neural Networks (PINNs) 20

7. Simulador y Experimentos de Validación **25**

7.1. Selección de Experimentos 25

7.2. Generalización de la solución y perspectivas de aplicación 26

8. Fundamentos del simulador basado en (S. Wang et al., 2023) **28**

8.1. No dimensionalización 28

8.2. Arquitectura de la red neuronal 29

8.2.1. Multi-layer Perceptrons 29

8.2.2. Mapeo de Fourier aleatorio 30

8.2.3. Factorización aleatoria de pesos 31
8.2.4. MLP modificado 32
8.3. Estrategias avanzadas de entrenamiento 32
8.3.1. Causalidad temporal 33
8.3.2. Balanceo dinámico de pérdidas 33
8.4. Optimizador y estrategias de muestreo 34

9. Implementación del simulador **35**
9.1. Entorno de desarrollo y librerías 35
9.2. Estructura del proyecto 35
9.3. Definición de la función de pérdida 36
9.4. Configuraciones generales de entrenamiento 36
9.5. Definición de los potenciales 36

10. Búsqueda de hiperparámetros **39**
10.1. Descripción del Grid Search 39
10.2. Detalles del Grid Search 40

11. Hardware utilizado **41**

12. Resultados y Discusión **42**
12.1. Experimento 1: Paquete de onda gaussiano 42
12.2. Experimento 2: Paquete de onda en un potencial escalón simulando una heterounión 45
12.3. Experimento 3: Doble pozo 50
12.4. Discusión sobre rendimiento y paralelización 54

13. Conclusiones **55**

Índice de figuras

1. Fases del ciclo CRISP-DM 10
2. Diagrama Gantt 11
3. Esquema general de una *Physics-Informed Neural Network* (PINN) 24
4. Potencial escalón 37
5. Potencial de doble pozo 38
6. Predicción sin potencial vs solución analítica en $t = 0$. 43
7. Predicción sin potencial vs solución analítica en $t = 0{,}5$. 44
8. Predicción sin potencial vs solución analítica en $t = 1$. 45
9. Predicción de la evolución de una partícula encontrando un potencial escalón en $t = 0$. 47
10. Predicción de la evolución de una partícula encontrando un potencial escalón en $t = 0{,}5$. 48
11. Predicción de la evolución de una partícula encontrando un potencial escalón en $t = 1{,}0$. 49
12. Predicción de la evolución de una partícula en un doble pozo de potencial en $t = 0{,}0$). 51
13. Predicción de la evolución de una partícula en un doble pozo de potencial en $t = 0{,}3$. 52
14. Predicción de la evolución de una partícula en un doble pozo de potencial en $t = 1{,}0$. 53

Lista de Acrónimos

Adam Adaptive Moment Estimation.

CPU Central Processing Unit.

CRISP-DM Cross-Industry Standard Process for Data Mining.

EDP Ecuación Diferencial Parcial.

FDM Finite Difference Method.

FEM Finite Element Method.

FFT Fast Fourier Transform.

GLU Gated Linear Unit.

GPU Graphics Processing Unit.

JAX Accelerated NumPy para aprendizaje automático.

MLP Multi-Layer Perceptron.

MSE Mean Squared Error.

PINN Physics-Informed Neural Network.

RWF Random Weight Factorisation.

TDSE Time-Dependent Schrödinger Equation.

TISE Time-Independent Schrödinger Equation.

TPU Tensor Processing Unit.

Resumen / Abstract

Este Trabajo Fin de Grado presenta el diseño, implementación y validación de un simulador *mesh-free* basado en **Physics-Informed Neural Networks** (PINN) para resolver la ecuación de Schrödinger y explorar aplicaciones en grafeno. El flujo CRISP-DM guía el proyecto desde la comprensión física hasta el despliegue; el código, escrito en Python/JAX, incorpora codificación de Fourier aleatoria, factorización aleatoria de pesos y entrenamiento causal con balance dinámico de pérdidas. El modelo se contrasta con soluciones analíticas y numéricas de alta resolución en tres escenarios unidimensionales: (i) propagación libre de un paquete gaussiano, (ii) dispersión en un potencial escalón que emula una heterounión y (iii) túnel en tiempo real en un doble pozo cuántico. En todos los casos la PINN logra errores *MSE* <5 %, requiriendo aproximadamente 3 h de entrenamiento en una NVIDIA A100. Frente a métodos clásicos discretos, la propuesta elimina la necesidad de mallado explícito, gestiona discontinuidades sin refinamiento local y se adapta de forma natural a problemas inversos y multidimensionales. Se discuten las principales limitaciones—coste de entrenamiento y captura de alta frecuencia—y se proponen líneas futuras de paralelización multi-GPU y enfoques híbridos PINN–FEM para extender el método a geometrías 2-D/3-D y más aplicaciones.

1. Introducción

La **Ecuación de Schrödinger**, formulada en 1926, constituye la piedra angular de la mecánica cuántica no relativista. Su resolución analítica o numérica permite predecir el comportamiento de partículas a escalas atómicas y subatómicas, siendo esencial para comprender fenómenos como el efecto túnel, los estados ligados en pozos de potencial y la dinámica de electrones en materiales bidimensionales.

Tradicionalmente, se emplean métodos numéricos clásicos (diferencias finitas, elementos espectrales o mallas adaptativas), pero enfrentan importantes desafíos cuando se aplican a sistemas de **alta dimensionalidad** (Beck et al., 2020) o potenciales que varían en el tiempo y presentan discontinuidades (Piotrowska et al., 2019). Estas dificultades se vuelven más notorias en problemas inversos (por ejemplo, identificar parámetros del potencial a partir de datos medidos) o en la modelación de materiales avanzados como el **grafeno**, donde la estructura hexagonal y la escala nanométrica introducen complejidades cuánticas adicionales.

En este contexto, surgen las ***Physics-Informed Neural Networks*** **(PINN)**, un nuevo paradigma en el que la ecuación física relevante se introduce directamente en la función de pérdida de la red neuronal. Esta estrategia prescinde de la necesidad de mallas espaciales complejas y hace que la resolución de **Ecuaciones Diferenciales Parciales (EDP)** sea más flexible y escalable. El objetivo es aprovechar la capacidad de generalización de las redes neuronales, al tiempo que se imponen los principios físicos pertinentes para garantizar soluciones que respeten las leyes de la mecánica cuántica.

2. Objetivos

2.1. Objetivo principal

- Diseñar, implementar y entrenar un **modelo PINN** capaz de resolver la ecuación de Schrödinger en distintos escenarios potenciales, asegurando:
  - La **reproducción cualitativa fiel** de la dinámica esperada (forma, fase y dispersión de la función de onda) en cada caso.
  - La **coherencia física** de las soluciones: conservación de la probabilidad, continuidad y cumplimiento de condiciones de contorno.
  - Un **proceso de entrenamiento estable y reproducible** ejecutable en hardware de uso general.

2.2. Objetivos secundarios

1. **Paquete de onda gaussiano libre**
   - Observar la traslación del paquete con la velocidad de grupo correcta.
   - Comprobar la dispersión gradual manteniendo la norma constante.
2. **Potencial escalón (heterounión)**
   - Visualizar la separación en componentes reflejada y transmitida.
   - Corroborar cualitativamente la coherencia de amplitudes y fases a ambos lados de la discontinuidad.
3. **Doble pozo cuántico**
   - Evidenciar el túnel cuántico y la transferencia alternante de probabilidad entre pozos.
   - Mostrar el cambio de estado.

3. Contribución a los ODS y alineación con los pilares de la Universidad Europea

Este Trabajo Fin de Grado se alinea de forma directa con varios **Objetivos de Desarrollo Sostenible (ODS)** de la Agenda 2030. En primer lugar, promueve la **ODS 4: Educación de calidad** al fomentar el aprendizaje avanzado de la física cuántica y de las ciencias de datos mediante entornos simulados y *challenge-based learning*. Asimismo, contribuye a la **ODS 9: Industria, innovación e infraestructura** a través de la propuesta de un simulador *mesh-free* de PINNs que impulsa la innovación en nanoelectrónica y facilita infraestructuras de computación reproducibles en la nube. De igual modo, apoya la **ODS 12: Producción y consumo responsables** al eliminar la necesidad de mallados extremadamente densos, reduciendo así el consumo energético asociado a las simulaciones clásicas de alta resolución. Finalmente, tiene un impacto indirecto en la **ODS 13: Acción por el clima** gracias a la optimización de dispositivos nanoelectrónicos más eficientes, lo cual contribuye a la descarbonización mediante una menor huella energética.

Además, este proyecto encarna los pilares educativos del modelo académico de la Universidad Europea al integrar de forma cohesionada sus principales líneas de actuación. Por un lado, ofrece entornos simulados que reproducen de forma segura fenómenos cuánticos complejos, favoreciendo la gamificación y el aprendizaje basado en problemas. Paralelamente, adopta un currículum integrado que combina física, computación de alto rendimiento y *machine learning*, evitando aprendizajes aislados en "silos". De igual importancia es la educación transdisciplinar, que fusiona competencias de ingeniería, matemáticas y ciencia de materiales para preparar al estudiante a trabajar en equipos multidisciplinares. Asimismo, se promueve la indagación continua y el pensamiento crítico mediante investigación aplicada y análisis data-driven, estimulando la creatividad en el diseño de algoritmos cuánticos. Por último, se enfatiza el compromiso ético y ciudadano global, impulsando el uso responsable de la computación intensiva y la contribución a soluciones tecnológicas sostenibles.

4. Diseño experimental: Metodología CRISP-DM

El presente proyecto se desarrollará siguiendo la metodología de trabajo **CRISP-DM** (Cross-Industry Standard Process for Data Mining), un enfoque ampliamente utilizado en ciencia de datos y proyectos de modelado. CRISP-DM se organiza en seis fases principales:

1. **Comprensión del negocio (Business Understanding):** Definir claramente los objetivos, alcance y restricciones del proyecto, así como los criterios de éxito. En nuestro caso, se traduce en entender los requisitos físicos y computacionales para resolver la Ecuación de Schrödinger.

2. **Comprensión de los datos (Data Understanding):** Recopilar información sobre los tipos de potenciales y condiciones de contorno. Aquí se investiga también la disponibilidad de datos para validación (soluciones analíticas o numéricas de referencia).

3. **Preparación de los datos (Data Preparation):** Seleccionar, limpiar y dar formato a los datos relevantes. En el contexto de una *Physics-Informed Neural Network (PINN)*, implica establecer las funciones de la Ecuación de Schrödinger, la discretización (si fuera necesaria) y los rangos de entrenamiento para la red.

4. **Modelado (Modeling):** Diseñar y entrenar la PINN considerando las pérdidas físicas (Ecuación de Schrödinger) y el posible uso de términos de regularización. Se eligen arquitecturas de red, funciones de activación, tasas de aprendizaje, etc. También se ajustan los hiperparámetros y se realizan experimentos para determinar la configuración óptima.

5. **Evaluación (Evaluation):** Comparar las soluciones obtenidas con la PINN frente a modelos analíticos o numéricos de referencia. Evaluar errores, velocidad de convergencia y robustez del enfoque. En esta fase se decide si el modelo cumple con los objetivos de precisión y eficiencia planteados.

6. **Despliegue (Deployment):** En un proyecto de investigación académica, esta fase incluye la documentación de resultados y la posibilidad de liberar el código en un repositorio público. Si se tratase de un entorno industrial, implicaría la integración del modelo en la cadena de producción o en sistemas de simulación de mayor envergadura.

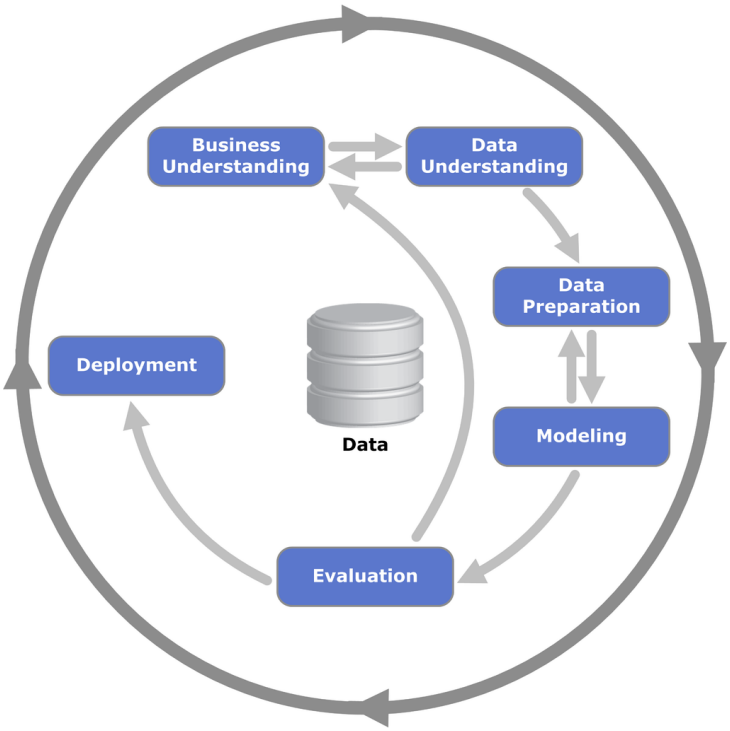

Figura 1: Fases del ciclo CRISP-DM. Fuente: Wikipedia contributors

Como se ilustra en la Figura 1, las fases del ciclo CRISP-DM se presentan de forma gráfica, facilitando la comprensión del proceso de análisis y modelado.

5. Temporalización

A continuación se presentan diagramas de Gantt tentativos que ilustran la planificación de las fases del proyecto.

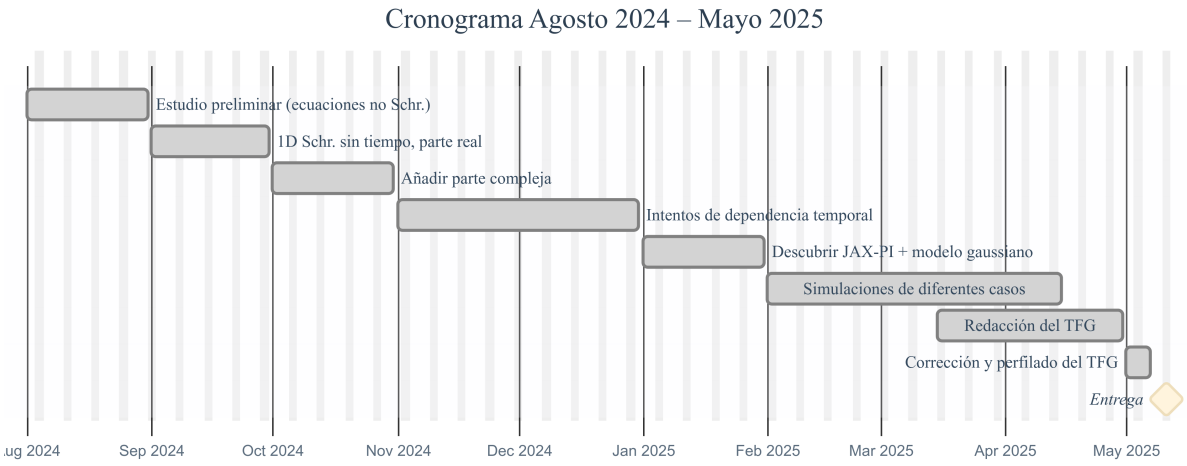

Figura 2: Diagrama Gantt

6. Estado del Arte

6.1. La ecuación de Schrödinger

La mecánica cuántica describe la dinámica de sistemas microscópicos mediante la ecuación de Schrödinger, que es una ecuación en derivadas parciales fundamental en esta teoría (Griffiths, 2005; Schrödinger, 1926). En su forma dependiente del tiempo, para una partícula de masa m en un potencial $V(\mathbf{r}, t)$, se escribe como

$$i\,\hbar\,\frac{\partial}{\partial t}\Psi(\mathbf{r},t) \;=\; -\frac{\hbar^2}{2m}\,\nabla^2\Psi(\mathbf{r},t) + V(\mathbf{r},t)\,\Psi(\mathbf{r},t)\,, \tag{1}$$

donde $\hbar$ es la constante de Planck reducida. Esta *ecuación de Schrödinger dependiente del tiempo* (TDSE, por sus siglas en inglés) gobierna la evolución temporal de la función de onda $\Psi(\mathbf{r}, t)$, cuyo módulo cuadrado $|\Psi|^2$ proporciona la distribución de probabilidad de hallar la partícula en cada punto del espacio (Griffiths, 2005).

En el caso de un Hamiltoniano autónomo (independiente de t), es posible buscar soluciones de la forma $\Psi(\mathbf{r}, t) = \phi(\mathbf{r})e^{-iEt/\hbar}$ que separan las variables. Al sustituir en la Ec. (1), se obtiene la *ecuación de Schrödinger independiente del tiempo* (TISE):

$$-\frac{\hbar^2}{2m}\,\nabla^2\phi(\mathbf{r}) + V(\mathbf{r})\,\phi(\mathbf{r}) \;=\; E\,\phi(\mathbf{r})\,, \tag{2}$$

la cual constituye un problema de autovalores para las energías E permitidas y sus funciones propias asociadas $\phi(\mathbf{r})$ (estados estacionarios del sistema) (Griffiths, 2005). Resolver exactamente la Ecuación de Schrödinger independiente del tiempo (TISE) equivale a encontrar los niveles de energía y estados cuánticos estacionarios, mientras que resolver la TDSE brinda la evolución dinámica del estado cuántico en el tiempo.

6.2. Método Tradicionales de Resolución de la Ecuación de Schrödinger

6.2.1. Método de Diferencias Finitas

Este método consiste en discretizar el dominio espacial (y temporal, si corresponde) en una malla de puntos y aproximar las derivadas de la ecuación de Schrödinger mediante cocientes de diferencias. Por ejemplo, en 1D se puede reemplazar la segunda derivada por la fórmula de diferencias finitas centradas:

$$\frac{d^2\psi}{dx^2} \approx \frac{\psi(x+\Delta x) - 2\,\psi(x) + \psi(x-\Delta x)}{\Delta x^2}\,,$$

lo que al insertar en la ecuación de Schrödinger (estacionaria) conduce a un sistema lineal (una matriz tridiagonal para 1D) cuya resolución proporciona valores aproximados de la función de onda ψ en cada nodo de la malla. En el caso dependiente del tiempo, se pueden usar esquemas explícitos o implícitos (e.g. Crank–Nicolson) para avanzar la solución en el tiempo, discretizando igualmente la derivada temporal.

El método de diferencias finitas (FDM, por sus siglas en inglés) es conceptualmente sencillo y fue uno de los primeros en popularizarse para resolver numéricamente la ecuación de Schrödinger. Su principal atractivo radica en la facilidad de implementación: basta con reemplazar derivadas por diferencias en una rejilla fija, lo que resulta directo incluso en códigos multidimensionales. De hecho, en campos como el electromagnetismo, el esquema de diferencias finitas en el dominio del tiempo (FDTD) se considera muy simple, incluso al programarlo en tres dimensiones (Taflove & Hagness, 2005). Además, los sistemas lineales resultantes suelen presentar estructuras matriciales dispersas con bandas diagonales, lo que permite resolverlos de forma eficiente. En una o dos dimensiones el coste computacional es manejable y la formulación explícita facilita la paralelización de las operaciones en la malla.

No obstante, las soluciones obtenidas con diferencias finitas están definidas únicamente en los nodos de la malla y su precisión queda limitada por el orden de la aproximación y el tamaño de paso Δx. Habitualmente se emplean esquemas centrados de segundo orden, por lo que el error de truncamiento decrece como $O(\Delta x^2)$, lo que implica que lograr alta precisión exige mallas muy finas, un mayor número de puntos y, por ende, un aumento del coste

computacional. En particular, en tres dimensiones el número de nodos crece como $N_x N_y N_z$ (del orden de N^3 si N es el número de divisiones por dimensión), lo que da lugar a matrices de tamaño masivo y a exigentes requerimientos de memoria. Por ejemplo, se ha estimado que resolver con alta precisión un sistema cuántico de alrededor de 100 átomos en 3D podría requerir aproximadamente 0.2 TB de memoria y 25 petaflops de cómputo incluso usando mallas adaptativas (Bylaska et al., 2009). Además, los esquemas explícitos de integración temporal imponen restricciones de estabilidad —el paso de tiempo debe ser suficientemente pequeño en relación con el paso espacial para garantizar la estabilidad—, lo que puede ralentizar los cálculos; en cambio, los métodos implícitos (como Crank–Nicolson) evitan estas limitaciones pero requieren resolver un sistema lineal en cada paso temporal. Finalmente, el FDM se adapta con dificultad a geometrías complejas o dominios irregulares, pues la malla suele ser regular (cuadrada o cúbica) y ajustar sus fronteras a contornos complicados exige estrategias adicionales (transformaciones de coordenadas, mallas no uniformes, etc.), a diferencia de otros métodos más flexibles.

6.2.2. Método de Elementos Finitos

El método de elementos finitos (FEM, por sus siglas en inglés) se basa en formular la ecuación de Schrödinger en su forma variacional (o forma débil) e introducir una discretización del dominio en subregiones pequeñas (elementos) sobre las cuales la función de onda se aproxima mediante funciones base polinomiales simples. Estos elementos finitos (por ejemplo, intervalos en 1D, triángulos o cuadriláteros en 2D, tetraedros en 3D) conforman una malla no necesariamente regular, lo que permite ajustarse a casi cualquier geometría de la región de cálculo. La solución aproximada se construye ensamblando las funciones base de todos los elementos en un sistema global de ecuaciones (típicamente, un sistema lineal disperso de gran tamaño, o un problema de autovalores disperso en el caso estacionario) (Sáenz, 2016). A diferencia del FDM, donde se obtienen valores en puntos discretos, el FEM produce una función $\psi(x)$ continua a trozos en todo el dominio, al combinar las funciones polinomiales de cada elemento de manera consistente en las fronteras entre elementos.

El método de elementos finitos (FEM) destaca por su gran flexibilidad para manejar

geometrías complejas y condiciones de frontera generales. Al subdividir el dominio en pequeños elementos, es posible adaptar el tamaño y la forma de la malla a la región de interés, refinando localmente donde la función de onda ψ varía rápidamente, como cerca de una barrera de potencial, y empleando elementos más grandes donde ψ es suave. Esta capacidad de refinamiento local (h-adaptatividad) combinada con la posibilidad de utilizar polinomios de grado superior en cada elemento (p-adaptatividad) permite alcanzar alta precisión con menos grados de libertad que una malla uniforme de diferencias finitas. Un mallado no o de elementos tetraédricos concentra la resolución en zonas críticas —por ejemplo, alrededor de centros nucleares en cálculos moleculares— y relaja la densidad en regiones suaves, optimizando los recursos computacionales. Matemáticamente, el FEM mantiene el soporte compacto de las funciones base —cada función está localizada en su elemento—, lo que conduce a matrices dispersas semejantes a las del FDM, pero con representaciones más exactas de singularidades físicas (como potenciales $1/r$) gracias a la flexibilidad en el mallado y en la elección de funciones de forma (Bylaska et al., 2009).

No obstante, el FEM es más complejo de implementar y su coste computacional por grado de libertad suele ser superior al del FDM. Generar una malla de elementos —especialmente en 3D—, elegir funciones de forma adecuadas, realizar la integración de las ecuaciones en cada elemento mediante cuadratura numérica y ensamblar las matrices globales implica un proceso sofisticado y proclive a errores de programación, lo que a menudo obliga a emplear bibliotecas especializadas. Aunque las mallas adaptativas pueden reducir el número de nodos, los sistemas lineales resultantes siguen siendo de gran tamaño y exigen solvers iterativos y precondicionadores avanzados, incrementando el tiempo de cálculo por iteración. En tres dimensiones, los requerimientos de memoria y cómputo para obtener soluciones de alta precisión pueden volverse prohibitivos; por ejemplo, en un estudio se sustituyó un esquema de diferencias finitas por un FEM adaptativo para la ecuación de Schrödinger de un átomo de neón, mejorando la precisión pero a costa de un esfuerzo computacional muy elevado (Bylaska et al., 2009). Así, aunque el FEM ofrece mayor precisión y adaptabilidad, sacrifica la simplicidad y ligereza computacional, siendo recomendable principalmente en problemas con geometrías irregulares o cuando se requieren errores muy bajos que un enfoque de diferencias finitas razonable no

puede garantizar.

6.2.3. Métodos Espectrales

Los métodos espectrales constituyen otra familia de técnicas numéricas para la ecuación de Schrödinger, caracterizadas por usar funciones base globales (por ejemplo, polinomios de Chebyshev, senos y cosenos de Fourier, funciones de Hermite, etc.) para aproximar la función de onda en todo el dominio. En lugar de evaluar $\psi(x)$ en puntos discretos (como FDM) o de subdividir el dominio en elementos locales (como FEM), aquí se expande $\psi(x)$ como una serie truncada en un conjunto de funciones ortogonales definidas en el dominio completo. Por ejemplo, en un intervalo $x \in [a, b]$ podríamos aproximar $\psi(x) \approx \sum_{n=0}^{N} c_n T_n(x)$ usando polinomios de Chebyshev $T_n(x)$, o su equivalente con series de Fourier en un dominio periódico. Luego se determina la mejor combinación de coeficientes c_n que satisface la ecuación de Schrödinger (por métodos de Galerkin, colocación, etc.). Una implementación común en problemas periódicos es el método pseudoespectral de Fourier: se expande $\psi(x)$ en modos e^{ikx}, se calcula el operador $-\frac{\hbar^2}{2m}\frac{d^2}{dx^2}$ exactamente en el espacio de Fourier (multiplicando por $-k^2$ a cada modo) y el potencial $V(x)$ en el espacio real, alternando mediante transformadas rápidas de Fourier (FFT). Métodos espectrales han sido aplicados con éxito a la ecuación de Schrödinger; un trabajo clásico de (Feit et al., 1982) demostró que la integración temporal con expansiones de Fourier permitía obtener espectros de energía con alta precisión y eficiencia, introduciendo así el espectral como una alternativa superior a diferencias finitas para ciertos problemas cuánticos (Feit et al., 1982). En la década de 1970, estos métodos ganaron popularidad justamente por su altísima precisión en problemas bien comportados, y hoy en día se emplean en aplicaciones que van desde dinámica molecular cuántica hasta mecánica de fluidos computacional.

La principal ventaja de los métodos espectrales es su elevada precisión. Al utilizar funciones de base globales muy suaves, la aproximación de la solución puede alcanzar errores extremadamente bajos con relativamente pocos coeficientes, siempre que la solución real sea lo suficientemente regular. De hecho, para problemas con soluciones analíticas suaves, el error de truncamiento disminuye casi exponencialmente al incrementar N (número de funciones base), en contraste con la convergencia polinómica (por potencia de N) de FDM/FEM (Piotrowska

et al., 2019). Esto significa que, por ejemplo, duplicar el número de modos en una expansión de Fourier puede reducir el error mucho más rápido que duplicar el número de puntos en diferencias finitas. En la práctica, los métodos espectrales pueden alcanzar la misma precisión que un esquema de segundo orden empleando un número de puntos un orden de magnitud inferior. Otra ventaja es que muchas implementaciones espectrales aprovechan algoritmos eficientes como la transformada rápida de Fourier (FFT, por sus siglas en inglés), logrando tiempos de cómputo favorables ($O(N \log N)$ por iteración). Además, los métodos espectrales conservan ciertas propiedades analíticas; por ejemplo, en problemas periódicos respetan exactamente la periodicidad y pueden ser vistos como evaluando el operador diferencial de forma casi exacta en el espacio transformado. En la ecuación de Schrödinger, los métodos espectrales (como el de Fourier o polinomios) han mostrado excelente rendimiento para simular la dinámica de paquetes de ondas y calcular espectros de energía en pozos y barreras suaves (Feit et al., 1982), siendo especialmente útiles en estudios de interacción láser-materia y dinámica de reacciones químicas donde se requiere alta fidelidad en la solución numérica.

Sin embargo, la eficacia de los métodos espectrales está fuertemente ligada a la suavidad de la solución y a la regularidad del dominio. Si la función de onda o el potencial presentan discontinuidades o variaciones muy bruscas, la serie espectral converge lentamente o incluso puede presentar oscilaciones espurias (fenómeno de Gibbs). En tales casos extremos, la tasa de convergencia puede degradarse a orden sub-lineal y el error máximo no disminuir en lo absoluto al refinar la serie (Piotrowska et al., 2019). Por ejemplo, si $V(x)$ tiene saltos discontinuos, una expansión global de Fourier de $\psi(x)$ mostrará ondulaciones cerca de esas discontinuidades que no desaparecen refinando la serie, a diferencia de un método local (FDM/FEM) que puede manejar discontinuidades colocando nodos o elementos en los puntos de salto. Otra limitación es que, al usar funciones base globales, generalmente se requiere que el dominio de cálculo tenga una forma *simple* (intervalo, rectángulo, etc.) o condiciones de contorno muy bien definidas (p.ej. periódicas o de valor en la frontera constante) para poder construir la base espectral. En problemas con geometrías arbitrarias, los métodos espectrales puros no son directamente aplicables, y suele recurrirse a variantes como el método de elementos espectrales (que combina la idea del espectral con partición del dominio en subdominios, hibridando con FEM).

Además, aunque el número de coeficientes necesarios pueda ser menor, las interacciones son globales: agregar un modo extra involucra recalcular integrales o coeficientes que en general dependen de todos los puntos, lo que implica operaciones con matrices llenas (no dispersas) o transformaciones integrales de todo el dominio. En cómputo paralelo, esto puede traducirse en mayor comunicación entre procesadores en comparación con FDM/FEM, aunque algoritmos como FFT están altamente optimizados. Finalmente, la implementación de un método espectral puede ser compleja desde el punto de vista matemático: requiere elegir una base apropiada para el problema, y derivar expresiones para proyectar la ecuación de Schrödinger en dicha base. No obstante, existen bibliotecas y paquetes (p.ej. para series de Fourier o polinomios ortogonales) que facilitan su uso. En resumen, los métodos espectrales son inmejorables en precisión para problemas suaves en dominios regulares, pero pierden ventaja en situaciones con no-linealidades fuertes, discontinuidades o dominios complicados, donde su alta convergencia teórica se ve limitada (Boyd, 2001).

6.2.4. Comparación General de los Métodos

En la Tabla 1 se resumen las principales ventajas y desventajas de cada método numérico discutido, comparándolos en términos de precisión, costo computacional, facilidad de implementación y aplicabilidad.

Método	**Precisión**	**Costo Computacional**	**Facilidad de Implementación**	**Aplicabilidad**
Dif. Finitas	Moderada (según malla)	Moderado en 1-2D; alto en 3D	Muy alta, sencilla	Dominios simples, potenciales suaves
Elem. Finitos	Moderada a alta (mejorable)	Alto, por ensamblaje de matrices	Compleja, requiere malla y formulación variacional	Geometrías complejas, adaptable
Espectrales	Muy alta para soluciones suaves	Variable, eficiente con FFT en casos suaves	Media, depende de la base	Dominios regulares y soluciones analíticas

Cuadro 1: Comparativa resumida de métodos numéricos para la ecuación de Schrödinger.

A lo largo de este recorrido por los métodos tradicionales —diferencias finitas, elementos finitos y espectrales— hemos constatado tanto sus fortalezas como sus limitaciones, especialmente al enfrentarse a potenciales discontinuos, geometrías complejas o dominios de alta dimensionalidad. Estas restricciones, unidas al creciente coste computacional y la dependencia de mallas o bases fijas, impulsan la búsqueda de un enfoque más flexible y adaptable que incorpore directamente las leyes físicas en el proceso de resolución. En este contexto, las *Physics-Informed Neural Networks* (PINNs) emergen como una alternativa prometedora, capaz de aprender soluciones continuas de la ecuación de Schrödinger sin necesidad de discretizaciones rígidas. A continuación se describe la propuesta de solución basada en redes neuronales PINNs.

6.3. Propuesta de solución con redes neuronales

En los últimos años ha emergido una alternativa innovadora para afrontar la resolución de ecuaciones diferenciales como la de Schrödinger: el empleo de *redes neuronales artificiales*. Impulsadas por los avances en *aprendizaje profundo*, las redes neuronales han demostrado una notable capacidad para aproximar funciones y dinámicas complejas en diversos problemas físicos (Carleo et al., 2019). En particular, se ha mostrado que es posible utilizar redes neuronales para representar estados cuánticos y resolver problemas de muchos cuerpos que desafiaban los métodos tradicionales (Carleo & Troyer, 2017). Teóricamente, las redes neuronales multicapa son aproximadores universales de funciones continuas (Hornik et al., 1989), lo que significa que, dado suficiente número de neuronas y capas, una red puede aproximar con la precisión deseada cualquier función bien comportada. Esto sugiere que una red neuronal suficientemente entrenada podría representar la función de onda $\Psi(\mathbf{r}, t)$ de un sistema cuántico, incluso para potenciales complicados o en espacios de alta dimensión, capturando todos los detalles necesarios.

La idea fundamental de esta propuesta es tratar la solución de la ecuación de Schrödinger como el resultado de una *función aproximadora* parametrizada por una red neuronal. Es decir, se define una red neuronal $\Psi_\theta(\mathbf{r}, t)$ con parámetros internos θ (pesos y sesgos) que recibe como entrada las coordenadas espaciales $\mathbf{r}$ y el tiempo t, y produce como salida un valor aproximado para la función de onda en esos puntos. A continuación, se entrenan los paráme-

tros θ mediante un algoritmo de optimización (por ejemplo, descenso por gradiente) para que la salida de la red satisfaga lo mejor posible la ecuación de Schrödinger y las condiciones físicas del problema (condiciones iniciales y de contorno, normalización, etc.). De esta manera, en lugar de resolver la Ecuación Diferencial Parcial (EDP) directamente por métodos numéricos tradicionales, se busca *aprender* la solución usando la red neuronal como modelo. Una vez entrenada, la red proporciona una expresión analítica (continua y derivable) de la función de onda, lo que permite evaluarla rápida y fácilmente en cualquier punto del espacio-tiempo, a cualquier resolución, sin necesidad de recalcular una malla completa. Esta característica ofrece una ventaja notable frente a los métodos clásicos, donde la solución suele conocerse sólo en los puntos discretos calculados. Además, abre la posibilidad de reutilizar el modelo entrenado para explorar diferentes regímenes de parámetros o realizar simulaciones múltiples con un costo incremental mucho menor que repetir un cálculo numérico desde cero.

El uso de redes neuronales para aproximar soluciones de ecuaciones diferenciales no es completamente nuevo. Por ejemplo, (Lagaris et al., 1998) ya demostraron la viabilidad de entrenar redes neuronales para resolver ecuaciones diferenciales ordinarias y en derivadas parciales sencillas mediante enfoques variacionales (Lagaris et al., 1998). No obstante, gracias al aumento en capacidad de cómputo y a los desarrollos modernos en arquitecturas de redes y técnicas de entrenamiento, esta idea se ha revitalizado y ampliado a problemas más complejos en la última década (Carleo et al., 2019). En este trabajo, se propone explotar esta metodología basada en redes neuronales para resolver la ecuación de Schrödinger del sistema en estudio, aprovechando su potencial para superar algunas limitaciones de los métodos tradicionales antes mencionados.

6.4. Physics-Informed Neural Networks (PINNs)

Una de las aproximaciones más exitosas para resolver EDPs con redes neuronales son las *redes neuronales informadas por física*, conocidas por sus siglas en inglés como **PINN** (Karniadakis et al., 2021; Raissi et al., 2019). A diferencia de las redes neuronales convencionales de aprendizaje supervisado, que aprenden a partir de un gran conjunto de datos de ejemplo (entradas y salidas deseadas), las PINNs incorporan directamente las leyes físicas (ecuaciones

diferenciales, condiciones de contorno, conservaciones) en el proceso de entrenamiento. En lugar de necesitar soluciones exactas para comparar, estas redes se entrenan imponiendo que su salida satisfaga la ecuación de Schrödinger y las condiciones físicas del problema en cada punto considerado (Raissi et al., 2019).

En práctica, el entrenamiento de una PINN para la ecuación de Schrödinger se realiza definiendo una *función de pérdida* (loss) que penaliza el incumplimiento de la ecuación y de las condiciones de borde. Por ejemplo, para la ecuación dependiente del tiempo, se puede definir el residuo del operador diferencial:

$$f_\theta(\mathbf{r},t) \;=\; i\,\hbar\,\frac{\partial}{\partial t}\Psi_\theta(\mathbf{r},t) \;-\; \Big(-\frac{\hbar^2}{2m}\nabla^2\Psi_\theta(\mathbf{r},t) + V(\mathbf{r},t)\,\Psi_\theta(\mathbf{r},t)\Big)\,,$$

que idealmente debería ser cero si Ψ_θ fuera la solución exacta. El término de pérdida asociado a la ecuación de Schrödinger se define entonces como el promedio del error cuadrático de f_θ en un conjunto de N puntos de *colocación* $\{(\mathbf{r_i}, t_i)\}_{i=1}^{N}$ distribuidos en el dominio:

$$L_{\text{EDP}}(\theta) \;=\; \frac{1}{N}\sum_{i=1}^{N}\Big|\, f_\theta(\mathbf{r_i}, t_i)\,\Big|^2\,, \tag{3}$$

mientras que otros términos de la pérdida pueden imponerse para las condiciones de frontera (por ejemplo, que Ψ_θ se anule en ciertos límites o que permanezca normalizada) y para la condición inicial en problemas dependientes del tiempo. El objetivo del entrenamiento es ajustar θ para minimizar la función de pérdida total $L(\theta)$, que suma las contribuciones de la ecuación diferencial y de todas las restricciones físicas:

$$L(\theta) \;=\; L_{\text{EDP}}(\theta) \;+\; L_{\text{frontera}}(\theta) \;+\; L_{\text{inicial}}(\theta) \;+\; \cdots \tag{4}$$

De esta forma, al final del entrenamiento se espera que $\Psi_\theta(\mathbf{r},t)$ sea una función que aproximadamente satisface $\forall\,(\mathbf{r},t)$ la ecuación de Schrödinger (es decir, $f_\theta \approx 0$) y además cumple con las condiciones físicas del problema. En esencia, la propia ecuación actúa como "maestro" del entrenamiento, guiando a la red para que aprenda una solución válida dentro del espacio de funciones posible.

La incorporación de las leyes físicas en la función de pérdida es la característica distintiva de las PINNs frente a las redes neuronales tradicionales. Esto permite entrenar la red *sin datos supervisados* (no se necesitan soluciones conocidas de antemano), lo cual es ideal ya que en nuestro caso precisamente buscamos la solución desconocida de la ecuación de Schrödinger. Además, garantiza que la solución obtenida respete las leyes de conservación y simetrías impuestas por la física del problema, reduciendo la posibilidad de obtener soluciones no físicas. Cabe destacar que para calcular el residuo f_θ y sus derivadas, las PINNs aprovechan la diferenciación automática proporcionada por los frameworks de aprendizaje profundo, con lo cual se obtienen derivadas precisas de la red sin necesidad de formulas analíticas derivadas a mano (Raissi et al., 2019).

Entre las **ventajas** de las PINNs se encuentra su naturaleza *mesh-free*: no requieren una malla fija de resolución predefinida, puesto que los puntos de entrenamiento se pueden elegir aleatoriamente en el dominio continuo e incluso ajustarse dinámicamente durante el entrenamiento (Markidis et al., 2021). La solución obtenida es una función continua que se puede evaluar en cualquier punto o resolución desada *a posteriori* sin costo adicional, a diferencia de un método de diferencias finitas que estaría limitado a los nodos de su malla o requeriría interpolación. Otra ventaja es la *generalización*: si el problema depende de parámetros (por ejemplo, ciertos parámetros del potencial V), en principio la red podría entrenarse incluyendo dichos parámetros como entradas, obteniendo una solución que abarque una familia de casos. Incluso si la red se entrena para un caso particular, cambiar ligeramente las condiciones (por ejemplo, la forma del potencial) no requeriría partir de cero, sino que podría emplearse la solución anterior como inicialización o ajustarse con unos pocos épocas de entrenamiento adicional, aprovechando el conocimiento previo. Asimismo, las PINNs ofrecen un marco unificado para abordar problemas *inversos* (determinar parámetros del modelo a partir de datos) simplemente incluyendo en la pérdida la diferencia entre la predicción de la red y los datos experimentales, manteniendo siempre los términos que imponen las leyes físicas (Karniadakis et al., 2021). Esto las hace muy atractivas cuando se dispone de algunas mediciones pero no de una solución completa.

Por otro lado, las PINNs también presentan **desafíos y desventajas** inherentes. El pro-

ceso de entrenamiento es esencialmente la resolución de un problema de optimización no convexa en un espacio de parámetros de alta dimensión; por ello, converger a la solución deseada puede requerir mucho tiempo de cómputo y no siempre está garantizado. Se han reportado dificultades específicas, como la tendencia de la red a aproximar primero las componentes de baja frecuencia de la solución (fenómeno conocido como *sesgo espectral*) lo que dificulta capturar detalles de alta frecuencia o transitorios rápidos (S. Wang et al., 2020). También pueden aparecer soluciones triviales (por ejemplo, que la red aprenda la solución nula) si la función de pérdida no está bien equilibrada o el entrenamiento no está adecuadamente regularizado, aunque existen técnicas para mitigar estos problemas (como el aumento progresivo de la complejidad del entrenamiento, funciones de activación adaptativas, o enfoques de *training* causal en el caso temporal). Otra limitación identificada en estudios recientes es que las PINNs pueden enfrentar dificultades con ciertas dinámicas complejas (por ejemplo, ondas con choques o discontinuidades fuertes, o escalas de tiempo muy largas), donde su rendimiento no supera al de métodos numéricos especializados (S. Wang et al., 2020). A pesar de ello, el campo está en pleno desarrollo y se proponen mejoras continuamente, desde arquitecturas híbridas red neuronal-método tradicional hasta algoritmos de entrenamiento más robustos, lo que está ampliando las fronteras de lo que las PINNs pueden lograr (Karniadakis et al., 2021).

En vista de todo lo anterior, en este trabajo se considera que la metodología basada en PINNs es la más adecuada para resolver el problema planteado. La capacidad de integrar el conocimiento físico directamente en el algoritmo de resolución es un aspecto crucial: nos asegura que la solución obtenida cumplirá intrínsecamente la ecuación de Schrödinger y conservará propiedades físicas (como la conservación de la probabilidad) de forma natural, algo que en métodos numéricos tradicionales debe controlarse con cuidado. Además, la flexibilidad de las PINNs nos permite abordar potenciales o condiciones frontera arbitrarias sin tener que rediseñar el algoritmo de resolución para cada caso; basta con incorporar los términos correspondientes en la función de pérdida. Esto representa una ventaja significativa en un contexto de Trabajo de Fin de Grado, ya que facilita explorar variantes del problema sin reescribir por completo el código de simulación. En suma, las PINNs combinan la generalidad de los métodos numéricos (pueden aproximar cualquier solución en principio) con la incorporación de

principios físicos sólidos, ofreciendo un enfoque moderno y potente para resolver la ecuación de Schrödinger. Por estas razones, se ha optado por emplear esta metodología en el desarrollo del presente trabajo.

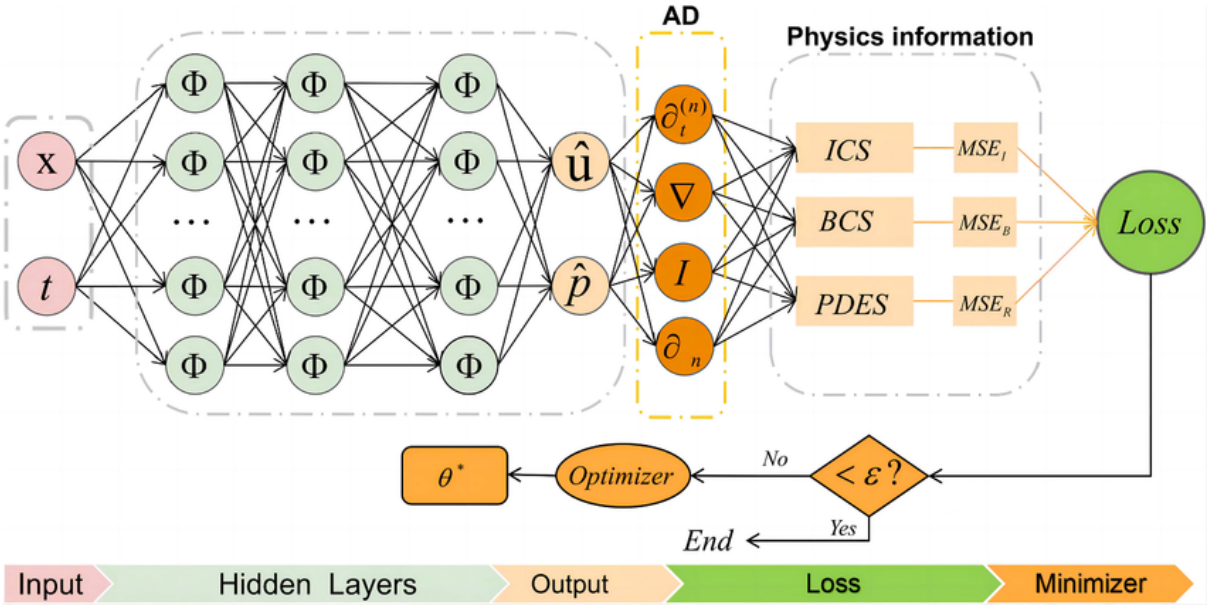

Figura 3: Esquema general de una *Physics-Informed Neural Network* (PINN). La red neuronal recibe coordenadas espaciales (x, y, z) y tiempo t como entradas, y devuelve la predicción de la función de onda $\Psi_\theta(x, y, z, t)$. La función de pérdida total combina restricciones físicas, provenientes de la ecuación diferencial (EDP), condiciones de frontera y condiciones iniciales. La red se entrena minimizando simultáneamente estas contribuciones. Fuente: "The architecture of the standard physics-informed neural networks", ResearchGate. https://www.researchgate.net/figure/The-architecture-of-the-standard-physics-informed-neural-networks_fig1_381549195 (accedido el 24 de abril de 2025).

7. Simulador y Experimentos de Validación

7.1. Selección de Experimentos

El objetivo principal de este trabajo es diseñar e implementar un simulador basado en *Physics-Informed Neural Networks* que permita resolver de manera eficiente y precisa la ecuación de Schrödinger en diversos escenarios físicamente relevantes. Para garantizar que dicho simulador funcione correctamente y sea capaz de abordar potenciales de distinta naturaleza, se han seleccionado tres experimentos de validación que cubren casos de complejidad creciente y que, en conjunto, aportan una visión global de sus prestaciones. Cada experimento juega un papel complementario: el primero pone a prueba la capacidad de la red para reproducir la dispersión libre de un paquete de onda, el segundo evalúa el manejo en las discontinuidades de potencial y en fenómenos de transmisión y reflexión, y el tercero explora la dinámica no estacionaria de una partícula inicialmente confinada en un pozo cuántico que colisiona contra una barrera, pudiendo reflejarse o transmitirse al otro lado.

El primer experimento consiste en simular la evolución de un paquete de onda gaussiano en un espacio con potencial nulo. Este caso, sencillo en cuanto al potencial, exige que el simulador capture con alta fidelidad la dispersión y propagación de la función de onda, respetando la conservación de la probabilidad y las simetrías temporales. La relevancia de este ejercicio radica en que cualquier solver de la ecuación de Schrödinger debe reproducir con precisión el comportamiento libre de partículas cuánticas antes de abordar configuraciones más complejas; si el modelo falla aquí, no podrá extenderse con garantías a sistemas con interacción.

Como segundo experimento se elige un potencial escalón, modelo esencial para describir heterouniones y barreras electrónicas en semiconductores. Al introducir un salto abrupto en la energía potencial, el simulador debe reproducir correctamente las amplitudes de transmisión y reflexión de la función de onda en la región de discontinuidad. Este caso pone a prueba la capacidad del PINN para lidiar con no suavidades en el potencial sin necesidad de refinar explícitamente una malla, algo que los métodos basados en rejillas regulares suelen abordar con dificultades. Demostrar que el simulador mantiene un error controlado en torno a la discontinuidad es clave para su aplicación a dispositivos reales, donde los cambios de potencial son

abruptos y críticos para el rendimiento.

Para el tercer experimento se plantea una partícula inicialmente localizada en un lado de un doble pozo cuántico, esta se propaga y colisiona contra una barrera intermedia. El simulador debe capturar tanto la reflexión parcial como la probabilidad de túnel hacia el otro lado de la barrera, así como las interferencias temporales resultantes en cada región. Este caso es representativo de procesos de transporte cuántico en nanoestructuras y permite evaluar la precisión de la PINN en la descripción de fenómenos de túnel en tiempo real, sin recurrir a técnicas de autovalores estacionarios.

Aunque cada uno de estos tres casos representa un desafío distinto —dispersión libre, fenómenos en discontinuidades y dinámica de túnel en un pozo—, su conjunto ofrece una batería completa de pruebas que abarca las principales dificultades numéricas y físicas de la ecuación de Schrödinger. Al comprobar que el simulador reproduce con éxito las dinámicas esperadas en estos escenarios, se obtiene la confianza necesaria para aplicarlo a potenciales más exóticos, multidimensionales o dependientes del tiempo sin necesidad de rediseñar la arquitectura básica de la red.

7.2. Generalización de la solución y perspectivas de aplicación

Resolver con éxito los tres experimentos descritos no solo valida el correcto funcionamiento del simulador, sino que establece las bases para su generalización a otros problemas cuánticos de interés. En primer lugar, la capacidad de la red para reproducir la evolución temporal de un paquete de onda gaussiano demuestra que el modelo domina la dinámica libre y conserva las propiedades de normalización a lo largo del tiempo. Esto es crucial para abordar posteriormente potenciales dependientes del tiempo o para incorporar interacciones externas, como campos electrodinámicos variables.

La correcta simulación del potencial escalón indica que el enfoque *mesh-free* inherente a las PINNs puede gestionar discontinuidades sin recurrir a refinar explícitamente una malla de nodos, lo que simplifica enormemente la extensión del método a geometrías más complejas o a sistemas con múltiples barreras, como superredes cuánticas o heteroestructuras de grafeno. En estos casos, bastará con incorporar en la función de pérdida los términos correspondientes al

potencial deseado, sin necesidad de modificaciones profundas en la topología de la red.

Por último, el estudio de la dinámica de túnel en un pozo cuántico simple y la correcta descripción de los fenómenos de reflexión y transmisión validan la capacidad del simulador para capturar procesos no estacionarios y temporales. Esta característica es fundamental para modelar transporte cuántico en nanoestructuras y para aplicaciones en diseño de dispositivos, donde frecuentemente se estudian respuestas transitorias.

En conjunto, superar estos tres casos de prueba garantiza que el simulador puede generalizar a una amplia gama de potenciales y condiciones de contorno, desde barreras múltiples y pozos acoplados hasta potenciales con dependencia temporal o espacial aleatoria. Esto abre la puerta a aplicaciones avanzadas en el estudio de grafeno, diseño de transistores de efecto de campo a escala nanométrica y problemas inversos de identificación de parámetros a partir de datos experimentales.

8. Fundamentos del simulador basado en (S. Wang et al., 2023)

En el desarrollo de nuestro simulador, hemos seguido detalladamente las metodologías propuestas por (S. Wang et al., 2023), con el objetivo de construir redes neuronales físicas informadas (PINNs) robustas y eficientes. A continuación se explica extensamente cada aspecto clave abordado en este enfoque.

8.1. No dimensionalización

La no dimensionalización consiste en redefinir las variables físicas y los parámetros del problema mediante magnitudes características, de forma que todas las cantidades relevantes queden normalizadas a un orden de magnitud unitario. Este procedimiento permite, por un lado, reducir drásticamente la diferencia de escalas entre las distintas variables (longitudes, tiempos, masas, etc.), y por otro, resaltar los parámetros adimensionales que gobiernan la dinámica del sistema.

Cuando escalamos, por ejemplo, la longitud x con un valor característico L, o el tiempo t con T, las nuevas variables adimensionales

$$\tilde{x} = \frac{x}{L}, \quad \tilde{t} = \frac{t}{T}$$

se distribuyen típicamente en el rango $\mathcal{O}(1)$. Esto evita que en los cálculos aparezcan valores excesivamente grandes o pequeños que puedan provocar que no sean totalmente exactos, ya que el ordenador sólo puede representar los números hasta cierto grado de detalle y, por tanto, introduce pequeñas imprecisiones que se van acumulando. Además evita gradientes muy pequeños (vanishing) o muy grandes (exploding), y, en última instancia, a una inestabilidad en el entrenamiento de redes neuronales (S. Wang et al., 2023).

Además, la no dimensionalización revela los números adimensionales intrínsecos al problema —como el número de Reynolds en mecánica de fluidos o el número de Courant en esquemas numéricos— que permiten comparar distintos regímenes físicos de manera universal.

Estos parámetros emergen al sustituir las variables escaladas en las ecuaciones originales y agrupar los coeficientes resultantes:

$$\mathrm{Re} = \frac{\rho\, U\, L}{\mu}, \quad \mathrm{Co} = \frac{U\, \Delta t}{\Delta x},$$

donde U y Δx son también magnitudes características asociadas a velocidad y malla numérica, respectivamente. De este modo, el análisis de sensibilidad de la red a estos números adimensionales ofrece una interpretación física más intuitiva y facilita la generalización de los resultados a sistemas de diferente escala.

Finalmente, trabajar con variables adimensionales simplifica el diseño de arquitecturas y la elección de funciones de activación y regularización, ya que se garantiza que los pesos y sesgos internos se mantengan en un intervalo controlado, mejorando la convergencia y reduciendo la necesidad de ajustes manuales de hiperparámetros.

8.2. Arquitectura de la red neuronal

La selección y optimización de la arquitectura es vital para el rendimiento de las PINNs, especialmente en tareas con soluciones altamente no lineales y oscilatorias.

8.2.1. Multi-layer Perceptrons

Utilizamos redes neuronales tipo Multi-Layer Perceptron (MLP) con entre 3 y 6 capas ocultas y un rango de 128–512 neuronas por capa, con funciones de activación suaves como la función hiperbólica tangente ($\tanh$). Esta función es preferida debido a su diferenciabilidad continua, necesaria para obtener derivadas de alto orden involucradas en los residuos de las ecuación en derivadas parciales (PDE). Además, la inicialización mediante el esquema de Glorot (Glorot & Bengio, 2010) garantiza una distribución equilibrada de los pesos, promoviendo un entrenamiento estable y evitando problemas de saturación neuronal prematura.

8.2.2. Mapeo de Fourier aleatorio

El mapeo de Fourier aleatorio se fundamenta en el teorema de Bochner, que afirma que una función de covarianza estacionaria positiva definida $k(\mathbf{x}, \mathbf{y}) = k(\mathbf{x} - \mathbf{y})$ puede representarse como la transformada de Fourier inversa de una medida de probabilidad. En la práctica, para aproximar un kernel Gaussiano o cualquier kernel de base radial, se muestrea una matriz $B \in \mathbb{R}^{D \times d}$ cuyas entradas siguen una distribución normal

$$B_{ij} \sim \mathcal{N}(0, \sigma^2),$$

y se define el mapeo de la entrada $\mathbf{x} \in \mathbb{R}^d$ al espacio de características:

$$\Phi(\mathbf{x}) = \begin{bmatrix} \cos\big(2\pi\, B\mathbf{x}\big) \\ \sin\big(2\pi\, B\mathbf{x}\big) \end{bmatrix}.$$

Este procedimiento convierte la función de activación de la MLP en una combinación lineal de senos y cosenos con frecuencias aleatorias, lo que permite a la red capturar patrones de alta frecuencia que, de otro modo, requieren arquitecturas mucho más profundas o complejas (Rahimi & Recht, 2007; Tancik et al., 2020).

Al enriquecer el espacio de entrada con componentes frecuenciales, la red dispone de un repertorio más amplio de bases sobre las que aproximar la función objetivo. Esto reduce el sesgo inductivo hacia soluciones suaves y mejora la convergencia, especialmente en problemas con variaciones rápidas o detalles finos. Experimentalmente se observa que, con valores de σ adecuados, el entrenamiento requiere menos épocas y alcanza errores inferiores en tareas de regresión y representación de señales continuas (S. Wang et al., 2023).

Además, dado que el mapeo es lineal en los parámetros de la capa intermedia (tras aplicar Φ), se preserva la propiedad de diferenciabilidad total, permitiendo el uso de optimizadores basados en gradiente sin modificaciones adicionales. El coste computacional extra es moderado: consiste en dos multiplicaciones matriciales y evaluaciones de funciones trigonométricas, operaciones que están altamente optimizadas en las bibliotecas modernas de aprendizaje automático.

8.2.3. Factorización aleatoria de pesos

La idea central de la factorización aleatoria consiste en descomponer cada matriz de pesos en un vector de escalas, que parametriza magnitudes, y en una matriz de direcciones:

$$\mathbf{W}^{(l)} = \mathrm{diag}\big(\exp(\mathbf{s}^{(l)})\big)\, \mathbf{V}^{(l)}.$$

Aquí, el vector $\mathbf{s}^{(l)} \in \mathbb{R}^{d_l}$ controla de forma individual la magnitud de cada fila de $\mathbf{V}^{(l)}$, garantizando positividad y diferenciabilidad suave gracias a la función exponencial. Esta separación explícita de escala y dirección facilita que el optimizador ajuste la amplitud de cada peso sin interferir directamente con su orientación.

La optimización conjunta de $\mathbf{s}^{(l)}$ y $\mathbf{V}^{(l)}$ produce un efecto de precondicionamiento implícito. En el espacio original, la matriz Hessiana de la función de pérdida suele presentar espectros con valores propios muy dispersos, lo que induce números de condición elevados y ralentiza la convergencia de métodos de primer orden. Al parametrizar $\mathbf{W}^{(l)}$ mediante $(\mathbf{s}^{(l)}, \mathbf{V}^{(l)})$, la segunda derivada respecto a cada componente de $\mathbf{s}^{(l)}$ acepta un término proporcional a la propia escala, mientras que las derivadas cruzadas con $\mathbf{V}^{(l)}$ se atenúan, logrando un bloque Hessiano de mejor condicionamiento numérico (S. Wang, Wang et al., 2022; S. Wang et al., 2023).

Desde el punto de vista práctico, este esquema equivaldría a disponer de una tasa de aprendizaje adaptativa por fila de peso, emulando métodos de optimización diagonalizados como AdaGrad o RMSProp, pero de forma estructural y continua. La dinámica de gradiente resultante tiende a descender más rápido por valles angostos de la función de pérdida y evita cambios excesivos de dirección en fases avanzadas del entrenamiento, favoreciendo la llegada a mínimos más profundos y estables.

Se ha observado empíricamente que redes profundas entrenadas con factorización aleatoria alcanzan tasas de convergencia superiores y mejor generalización en tareas de visión por computador y modelado de secuencias, reduciendo hasta un 30 % el número de épocas necesarias para un mismo nivel de validación (S. Wang, Wang et al., 2022).

8.2.4. MLP modificado

Se hizo uso de una arquitectura modificada que combina dos codificadores paralelos $\mathbf{U}, \mathbf{V}$ en cada capa mediante una función sigmoide moduladora:

$$\mathbf{g}^{(l)} = \sigma\big(f^{(l)}\big) \odot \mathbf{U} + \big(1 - \sigma\big(f^{(l)}\big)\big) \odot \mathbf{V}.$$

La función moduladora $\sigma(f^{(l)})$ actúa como una puerta suave que pondera dinámicamente la salida de ambos codificadores, adaptando la representación interna de la red a las variaciones contextuales de la entrada. Gracias a esta mezcla ponderada, la red puede enfocarse en subespacios de características relevantes en cada capa, facilitando el modelado de transiciones continuas en la dinámica del fenómeno físico.

La dependencia de $f^{(l)}$ introduce un mecanismo similar a atención local que ajusta la proporción de contribución de cada codificador según el contexto, de modo que la capacidad de aproximar funciones no lineales muy complejas se ve ampliamente incrementada. Al emplear la activación sigmoide se asegura una derivada suave, lo que mejora la estabilidad del entrenamiento y el flujo de gradiente al evitar saturaciones abruptas. Este diseño guarda relación con las Gated Linear Units (GLU) y con métodos de mezcla de expertos (Mixture-of-Experts), combinando flexibilidad en la selección de rutas de información y control sobre el paso de señal.

En escenarios de dinámica real, donde las variables pueden experimentar cambios tanto abruptos como graduales, el MLP modificado captura patrones locales y globales sin necesidad de aumentar de forma drástica la profundidad o anchura de la red. Además, la paralelización de los codificadores facilita la integración de diferentes fuentes de información (por ejemplo, componentes espectrales y temporales) dentro de un mismo bloque, mejorando la robustez y expresividad del modelo (S. Wang et al., 2021, 2023).

8.3. Estrategias avanzadas de entrenamiento

Para enfrentar los desafíos intrínsecos de optimización de PINNs, introducimos técnicas avanzadas que mejoran la estabilidad y eficacia del aprendizaje.

8.3.1. Causalidad temporal

La estrategia de ponderación exponencial dependiente del residuo acumulado, definida como

$$w_i = \exp\Big(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r^k(\theta)\Big),$$

introduce un mecanismo de adaptación dinámica del aprendizaje que favorece la corrección temprana de errores antes de avanzar a instantes posteriores (S. Wang, Sankaran & Perdikaris, 2022; S. Wang et al., 2023). Al reducir automáticamente la influencia de aquellos pasos temporales con pérdidas acumuladas altas, el modelo se centra en optimizar las predicciones de los estados iniciales, lo que garantiza que las discrepancias del pasado no se propaguen indebidamente hacia el futuro.

Este enfoque explota la interpretación física de la causalidad en sistemas dinámicos: dado que las condiciones iniciales determinan la evolución subsiguiente, es esencial mitigar cuanto antes los errores en los primeros intervalos de tiempo. La tasa de olvido controlada por el parámetro ϵ permite ajustar la sensibilidad del modelo al residuo histórico, ofreciendo un compromiso entre estabilidad y reactancia ante cambios abruptos en la dinámica subyacente.

Desde el punto de vista de la convergencia numérica, la ponderación exponencial actúa como un regulador suave que previene la explosión de gradientes al atenuar las contribuciones de intervalos especialmente conflictivos. Al mismo tiempo, preserva la capacidad del optimizador para aprender de errores recientes más significativos, lo que se traduce en una mayor robustez frente a ruido y perturbaciones en los datos experimentales.

8.3.2. Balanceo dinámico de pérdidas

El balanceo dinámico de pérdidas se basa en la medición continua de las normas de los gradientes asociados a cada término de la función de pérdida y en la adaptación automática de sus pesos. De esta forma, si denotamos por $\mathcal{L}_i$ la contribución al error de la condición inicial, de frontera o de la ecuación diferencial parcial, los coeficientes adaptativos $\hat{\lambda}_i$ se determinan mediante la condición

$$\|\hat{\lambda}_i \nabla_\theta \mathcal{L}_i\| = \sum_j \|\nabla_\theta \mathcal{L}_j\|.$$

Este procedimiento garantiza que, en cada iteración de optimización, la influencia de cada término sea equiparable, evitando que uno solo de ellos domine el proceso de aprendizaje. Como resultado, se obtiene una convergencia más rápida y una solución más precisa de la PDE, al actuar de manera equilibrada sobre todas las componentes de la función de pérdida (S. Wang et al., 2023).

8.4. Optimizador y estrategias de muestreo

Para el entrenamiento de la red neuronal empleamos el optimizador Adam con decaimiento exponencial que reduce gradualmente el tamaño del paso a lo largo de las épocas. Adam combina momentos de primer orden (media móvil de los gradientes) y de segundo orden (media móvil de los cuadrados de los gradientes), lo que le confiere invarianza ante escalas de los parámetros y le permite adaptarse automáticamente a paisajes de optimización complejos sin requerir un ajuste fino de hiperparámetros estáticos. El decaimiento exponencial de la tasa de aprendizaje favorece la estabilización de la convergencia en fases finales, evitando oscilaciones en torno al mínimo y mejorando la precisión final sin incurrir en regularización excesiva.

El muestreo aleatorio uniforme de puntos en el dominio garantiza que la red reciba información representativa de toda la región de interés, previniendo el sobreajuste en zonas particulares y asegurando un aprendizaje balanceado. Sin embargo, dado que las soluciones de PDE suelen presentar regiones con gradientes más pronunciados o con singularidades locales, incorporamos un muestreo adaptativo basado en residuos (RAD). En cada iteración se evalúa el error residual en un conjunto de puntos de validación y se aumenta la densidad de muestreo en aquellas zonas donde el residuo es mayor. De este modo, la atención computacional se concentra dinámicamente en las regiones más difíciles de aproximar, mejorando la calidad global de la solución y acelerando la convergencia hacia la precisión deseada (S. Wang et al., 2023).

9. Implementación del simulador

9.1. Entorno de desarrollo y librerías

El simulador se ha implementado en **Python 3.9** aprovechando el framework **JAX** para autograd y computación en GPU/TPU. Sobre JAX se basa la librería **jaxpi**[1], creada por los autores de *An Expert's Guide to PINNs*, que contiene los módulos básicos para resolver PDEs con PINNs. Para la definición de modelos se utiliza **Haiku** y para la optimización **Optax**. Además, contamos con `numpy` y `scipy` para cálculo numérico y utilidades variadas, así como `matplotlib` y `seaborn` para la visualización de resultados.

9.2. Estructura del proyecto

El repositorio principal se denomina `jaxpi/` y en su interior están organizados los subproyectos correspondientes a cada uno de los potenciales estudiados. En concreto, cada carpeta de potencial contiene un `model.py` con la definición de la PINN (arquitectura, mapeo de Fourier y funciones de pérdida), un `utils.py` para la generación de condiciones iniciales y un `train.py` que implementa el bucle de entrenamiento junto con la configuración del optimizador y del scheduler. Los tres potenciales incluidos son el paquete de onda gaussiano (`gaussian_wavepacket/`), el potencial escalón (`step_potential/`) y el doble pozo cuántico (`double_well/`). Complementariamente, existe un notebook `postprocess.ipynb` en la raíz del repositorio para la generación de todas las visualizaciones finales. Además, para cada simulación hay un archivo `configs/default.py` en el que se definen configuraciones de todo tipo, como pesos iniciales, parámetros de entrenamiento, niveles de log, arquitectura, regularizaciones, etc., de modo que sea sencillo ajustar cualquier aspecto sin modificar el código principal.

El repositorio está disponible en GitHub: https://github.com/EminLarbi/Q-PINN.

[1] https://github.com/PredictiveIntelligenceLab/jaxpi

9.3. Definición de la función de pérdida

En el archivo `model.py` de cada potencial se construye la pérdida total como la suma de la contribución del término de la EDP y la de las condiciones iniciales. Formalmente, se define

$$L_{\text{EDP}} \;=\; \frac{1}{N_{\text{pde}}}\sum_{i=1}^{N_{\text{pde}}}\left|f_\theta(x_i,t_i)\right|^2, \quad L_{\text{inicial}} \;=\; \frac{1}{N_{\text{ic}}}\sum_{k=1}^{N_{\text{ic}}}\left|\Psi_\theta(x_k,0)-\Psi_0(x_k)\right|^2,$$

y la pérdida total

$$L_{\text{total}} \;=\; L_{\text{EDP}} + L_{\text{inicial}}.$$

Todas las constantes, pesos de cada término y ajustes relacionados con el escalado de la pérdida están definidos en el fichero `configs/default.py`, lo que facilita la experimentación y el ajuste fino de la contribución relativa de cada término.

9.4. Configuraciones generales de entrenamiento

Para optimizar las redes se emplea el algoritmo Adam con los hiperparámetros $\beta_1 = 0{,}9$ y $\beta_2 = 0{,}999$, entrenando hasta un máximo de 200 000 epochs. El balanceo dinámico de pérdidas se lleva a cabo mediante coeficientes adaptativos basados en la norma de los gradientes, y se implementa un entrenamiento causal que aplica pesos temporales exponenciales para respetar la cronología de los datos. Todos estos parámetros, así como la cadencia de guardado de checkpoints, las frecuencias de logging y los detalles del scheduler, están centralizados en el archivo `configs/default.py` para garantizar consistencia y trazabilidad.

9.5. Definición de los potenciales

Primer experimento: potencial libre

$$V(x) = 0, \quad \text{para todo } x,$$

ya que en este caso no hay potencial externo.

Segundo experimento: Potencial escalón

$$V(x) = \begin{cases} V_{\text{barrera}}, & x \in \left(\frac{x_{b1}}{X_{\text{car}}}, \frac{x_{b2}}{X_{\text{car}}}\right), \\ 0, & \text{en otro caso}, \end{cases} \qquad V_{\text{barrera}} = \frac{5\,T_{\text{car}}^2}{M_{\text{car}}\,X_{\text{car}}^2}.$$

Tercer experimento: Potencial de doble pozo

$$V(x) = \begin{cases} 0, & |x-a| \le \delta \quad \text{o} \quad |x+a| \le \delta, \\ V_0\,\dfrac{(x^2-a^2)^2}{a^4}, & \text{en otro caso}, \end{cases} \qquad V_0 = \frac{8\,T_{\text{car}}^2}{M_{\text{car}}\,X_{\text{car}}^2}, \quad a = x_0, \quad \delta = 0{,}5\,|a|.$$

La condición $V(x) = 0$ en las regiones $|x \pm a| \le \delta$ se introduce para que el paquete de onda gaussiano inicial concuerde exactamente con la ecuación de Schrödinger en $t = 0$, evitando discontinuidades de fase en los mínimos del doble pozo.

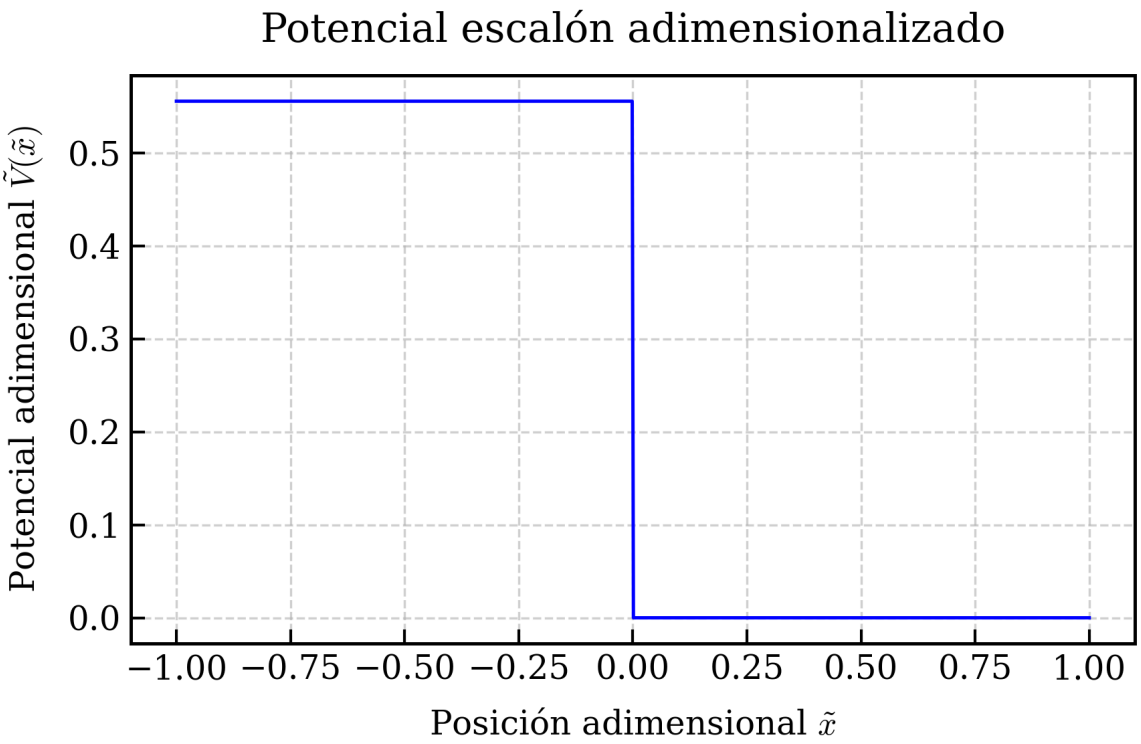

Figura 4: Potencial escalón

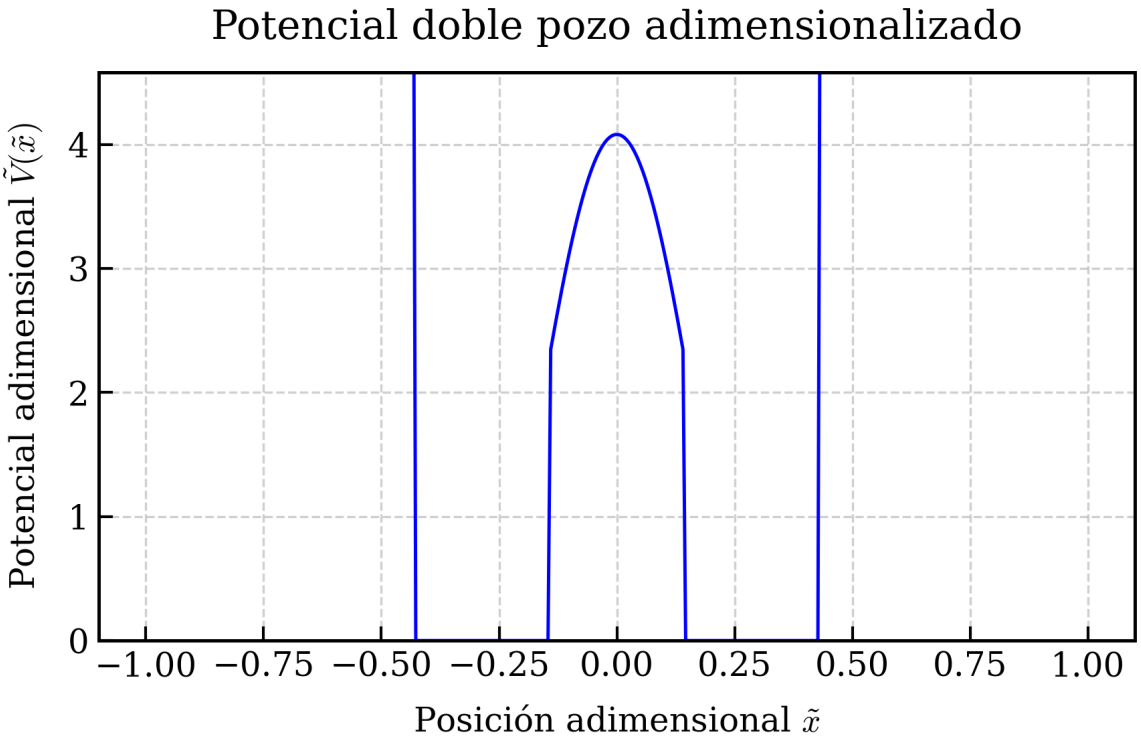

Figura 5: Potencial de doble pozo

10. Búsqueda de hiperparámetros

Para garantizar un rendimiento adecuado del modelo sin disponer de recursos computacionales ilimitados, se llevó a cabo un pequeño *grid search* centrado en los hiperparámetros más influyentes, basándose en las recomendaciones iniciales de (S. Wang et al., 2023). En lugar de explorar exhaustivamente todas las combinaciones posibles, se seleccionaron cuatro bloques clave de parámetros (arquitectura, entrenamiento, muestreo y ponderación) y se evaluaron un número reducido de valores representativos para cada uno. Esta estrategia permitió identificar rápidamente configuraciones prometedoras y descartar aquellas que no cumplían los criterios de eficiencia y precisión deseados.

10.1. Descripción del Grid Search

La búsqueda en rejilla, o *grid search*, es una estrategia sistemática para explorar el espacio de hiperparámetros de un modelo de forma exhaustiva dentro de unos rangos predeterminados. En lugar de confiar en la intuición o en ajustes manuales puntuales, esta técnica define previamente un conjunto de valores posibles para cada parámetro relevante y evalúa de manera automática cada combinación posible. De este modo, se obtiene una visión completa de cómo interactúan los diferentes ajustes y se facilita la identificación de aquellos que conducen a un mejor comportamiento en tareas de entrenamiento y validación.

Aunque su simplicidad conceptual es una de sus grandes virtudes, la búsqueda en rejilla puede resultar costosa en términos de tiempo de cómputo, especialmente cuando el número de parámetros y los valores a probar crecen de forma significativa. Para salvaguardar la viabilidad del proceso, es habitual seleccionar bloqueos acotados de hiperparámetros —los más críticos o los que, según estudios previos, tengan mayor impacto en el rendimiento— y limitar el número de valores para cada uno. De esta forma, se reduce la exhaustividad total sin sacrificar la capacidad de descubrir configuraciones prometedoras que sirvan de base para refinamientos posteriores.

Una ventaja adicional de la búsqueda en rejilla es su carácter determinista y reproducible: dado un conjunto fijo de combinaciones, los experimentos pueden reanudarse o comparar-

se sin introducir variabilidad por procedimientos estocásticos en la fase de optimización. Esta propiedad resulta especialmente valiosa en entornos de investigación y en producción, donde la transparencia y la trazabilidad de las decisiones de diseño de modelos son fundamentales. No obstante, cuando los recursos computacionales lo permiten, suele complementarse con técnicas más sofisticadas —como la búsqueda aleatoria o métodos bayesianos— para afinar aún más las regiones del espacio de hiperparámetros que, tras un primer barrido, muestran mayor potencial.

10.2. Detalles del Grid Search

En cuanto a la arquitectura de la red, se compararon variantes con 3, 4 y 5 capas ocultas, y tamaños de 128 y 256 neuronas por capa, manteniendo siempre la función de activación `tanh`. Para el entrenamiento, se probaron tasas de aprendizaje iniciales de 0.001 y 0.0005, con un plan de decaimiento exponencial cada 1 000, 2 000 y 5 000 pasos y tamaños de lote de 32 y 64 muestras.El muestreo de puntos de PDE empleó métodos uniformes frente a muestreo adaptativo basado en residuo, analizando el impacto de cada uno en la convergencia. Finalmente, en el balanceo de pérdidas se evaluaron tolerancias causales de 1.0 y 0.5, y divisiones del dominio en 16 o 32 fragmentos para la estrategia de entrenamiento causal.

Aunque el grid search fue necesariamente limitado por la disponibilidad de GPU, los mejores resultados se obtuvieron con la siguiente configuración final. Además, se seleccionó un embedding de Fourier con una escala de 1.0 y una dimensión de 256, tal como recomiendan las guías iniciales.

- **Arquitectura:** 4 capas ocultas de 256 neuronas, activación `tanh`, Fourier feature scale = 2.0.
- **Embedding de Fourier:** (`embed_scale`) = 1.0; (`embed_dim`) = 256.
- **RWF (Random Weight Factorisation):** $\mu = 0{,}5$, $\sigma = 0{,}1$.
- **Entrenamiento:** learning rate = 0.001 con decay cada 2 000 steps; batch size = 64.
- **Ponderación causal:** tolerancia = 1.0; número de chunks = 32.

Esta configuración se ha aplicado en todos los experimentos presentados en este trabajo.

11. Hardware utilizado

El entorno usado para la ejecución de los cálculos ha sido Google Colaboratory Pro+. Este se ejecuta sobre máquinas virtuales con procesadores Intel Xeon de arquitectura Haswell/Cascade Lake, ofreciendo hasta 8 vCPUs a 2.20–2.30 GHz y una memoria RAM de 53 GB en el modo High-RAM. El almacenamiento temporal local supera los 150 GB en Colab Pro+, lo que permite manejar conjuntos de datos voluminosos sin necesidad de recurrir a almacenamiento externo.

La aceleración de cómputo recae en la GPU NVIDIA A100 Tensor Core con 40 GB de memoria HBM2, un ancho de banda de memoria de 1555 GB/s, 6 912 núcleos CUDA y 432 Tensor Cores de tercera generación optimizados para FP16, BF16 y TF32. En precisión simple (FP32) alcanza hasta 19,5 TFLOPS, mientras que en precisión mixta (TF32) llega a los 312 TFLOPS.

Las sesiones en Colab Pro+ tienen un límite de duración continua de 24 horas, con desconexiones automáticas por inactividad o preemptions de Google. Aunque Pro+ reduce la probabilidad de interrupciones y permite reconexión tras cerrar el navegador, sigue siendo esencial guardar el progreso con frecuencia.

El sistema operativo subyacente es Ubuntu Linux (actualmente versión 18.04 LTS) que incluye Python 3 y librerías de ciencia de datos preinstaladas, como NumPy, pandas, scikit-learn y frameworks de deep learning como TensorFlow y PyTorch, facilitando el inicio inmediato de proyectos sin instalaciones adicionales.

12. Resultados y Discusión

12.1. Experimento 1: Paquete de onda gaussiano

Teoría Consideremos un paquete de onda gaussiano libre, inicialmente centrado en x_0 con ancho σ y número de onda k_0. La función de onda en el instante $t = 0$ está dada por

$$\Psi(x,0) = \exp\Big(-\frac{(x-x_0)^2}{2\sigma^2}\Big)\exp\big(ik_0x\big).$$

Físicamente, esperamos que este paquete se propague con velocidad de grupo

$$v_g = \frac{\hbar k_0}{m},$$

mientras simultáneamente su ancho efectivo $\sigma(t)$ crezca según

$$\sigma(t) = \sigma\sqrt{1+\Big(\tfrac{\hbar t}{m\sigma^2}\Big)^2},$$

debido a la dispersión cuántica. La conservación de la probabilidad implica $\int_{-\infty}^{\infty}|\Psi(x,t)|^2\,dx = 1$ para todo t. La solución analítica exacta es

$$\Psi(x,t) = \frac{\sigma}{\sqrt{\sigma^2+\frac{i\hbar t}{m}}}\exp\Big(-\frac{\big[x-x_0-(\hbar k_0/m)\,t\big]^2}{2\big(\sigma^2+\frac{i\hbar t}{m}\big)}\Big)\exp\big(ik_0x-i\omega_0t\big),$$

donde $\omega_0 = \hbar k_0^2/(2m)$.

Resultados A lo largo de la simulación, la PINN reprodujo fielmente el desplazamiento del centroide del paquete de onda, siguiendo el avance esperado según la velocidad de grupo teórica. En todos los tiempos muestreados se apreció una traslación uniforme sin deformaciones apreciables en la forma principal de la envolvente.

Asimismo, el ensanchamiento progresivo quedó reflejado en un aumento suave del ancho del paquete, muy parecido al comportamiento analítico, de modo que las variaciones en la amplitud máxima permanecieron prácticamente imperceptibles. La norma total del paquete se

mantuvo estable, lo que evidencia que la red neuronal conserva adecuadamente la probabilidad dentro de márgenes prácticamente imperceptibles.

A continuación se muestran las comparaciones gráficas entre la predicción de la PINN y la solución analítica en tres instantes de tiempo representativos:

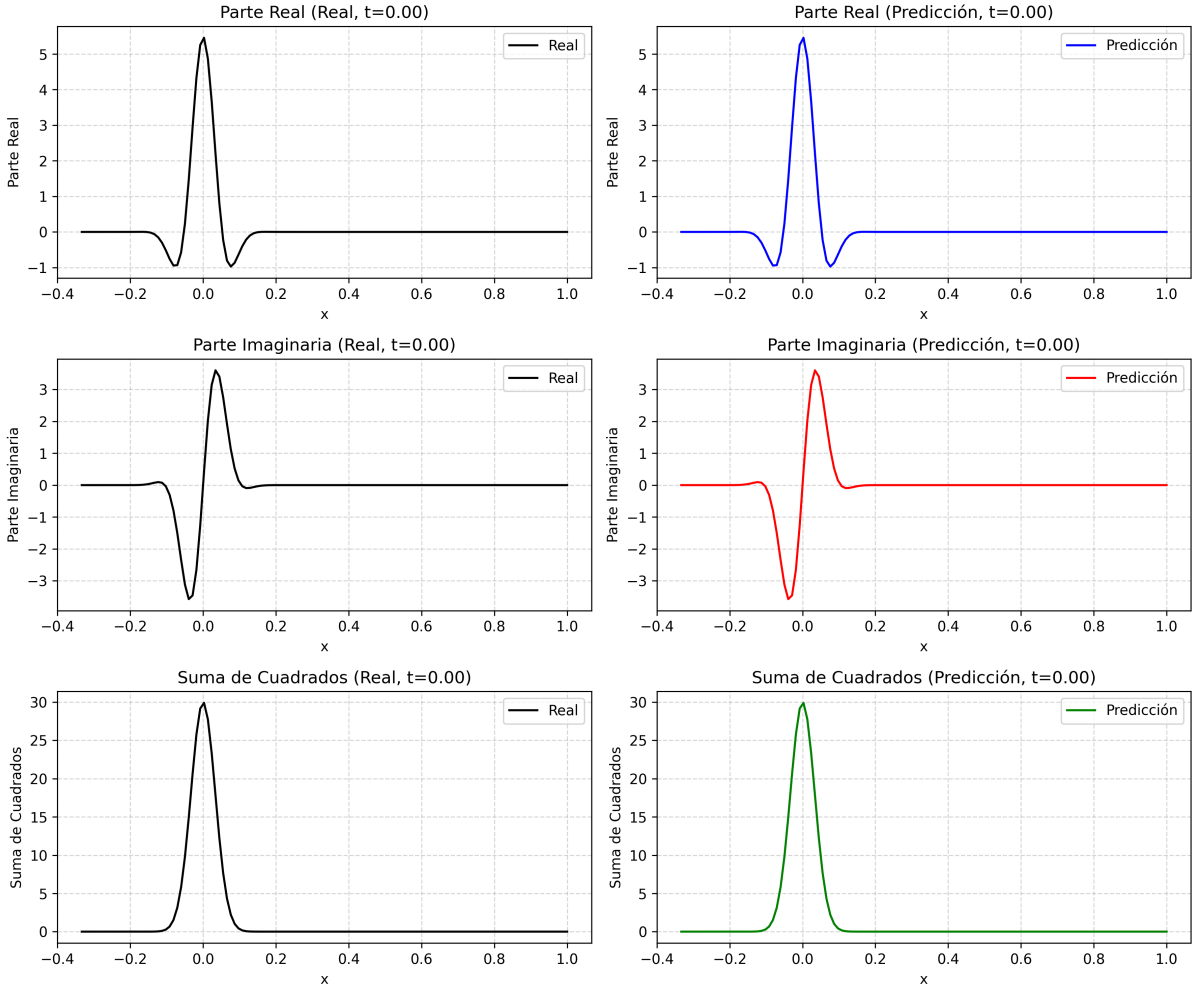

Figura 6: Predicción sin potencial vs solución analítica en $t = 0$.

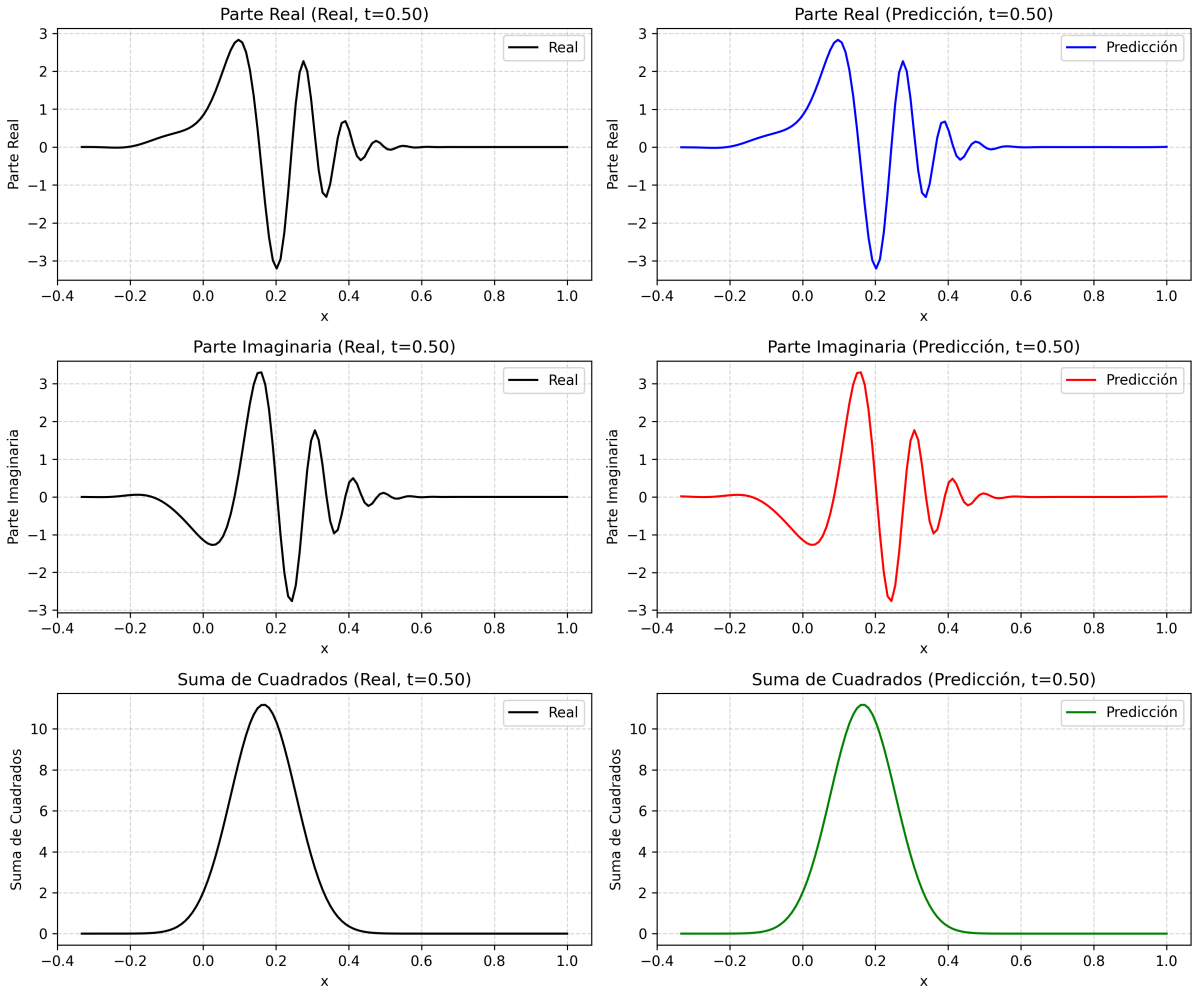

Figura 7: Predicción sin potencial vs solución analítica en $t = 0{,}5$.

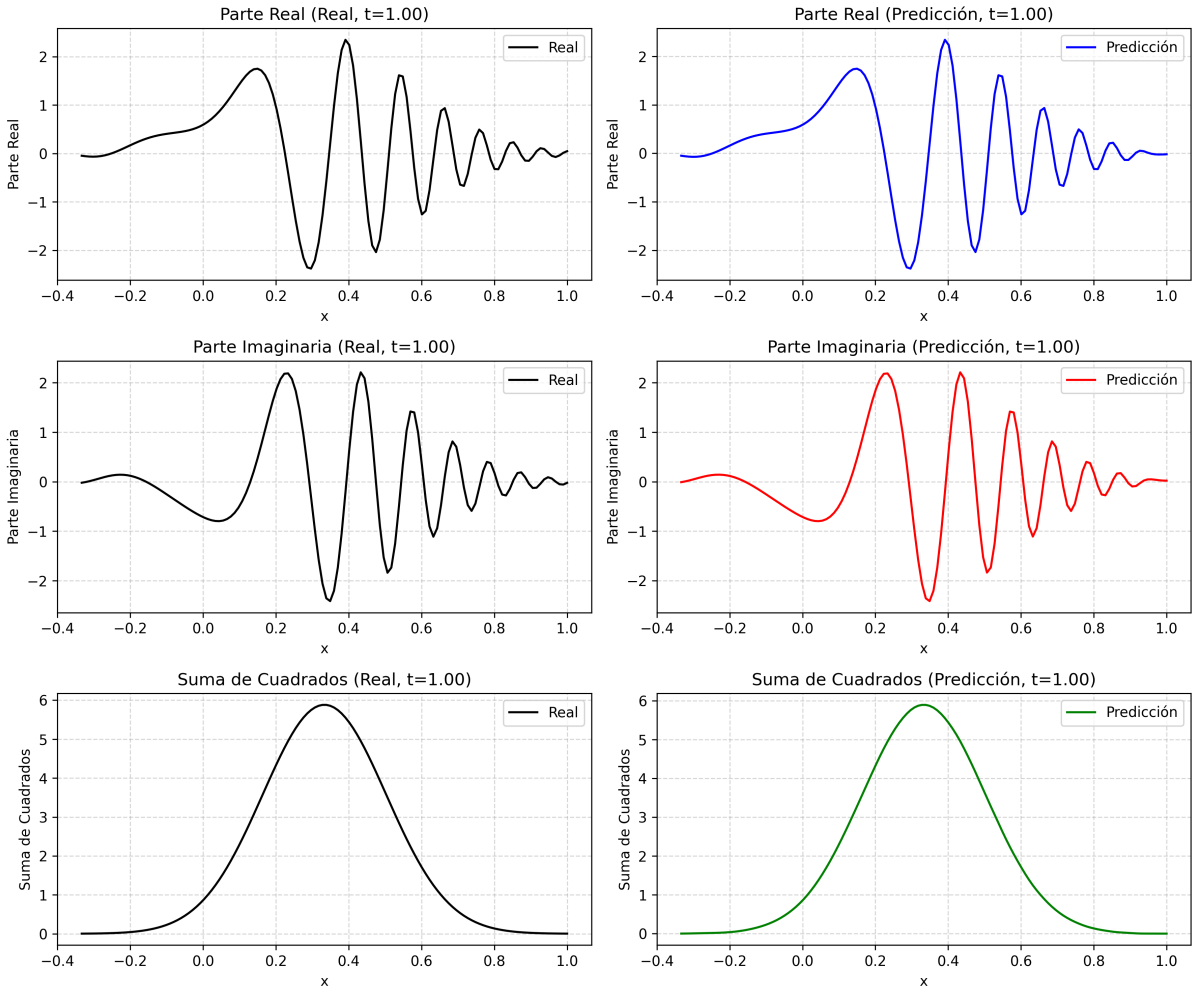

Figura 8: Predicción sin potencial vs solución analítica en $t = 1$.

12.2. Experimento 2: Paquete de onda en un potencial escalón simulando una heterounión

Teoría En este experimento consideramos un paquete de onda incidente sobre un potencial escalón que simula la transición entre dos materiales con diferentes propiedades de banda, es decir, una heterounión. Al aproximar la región de mayor potencial como un escalón de energía constante, se espera que la parte del paquete que alcanza la discontinuidad sufra una división en dos componentes: una porción reflejada que retorna hacia la región de origen y otra transmitida que penetra en la zona de mayor potencial. Cualitativamente, la amplitud de la onda reflejada y la transmitida dependen de la diferencia de altura del escalón y de la energía media del paquete

de onda. En la zona transmitida el paquete adopta un número de onda distinto, lo que se traduce en un cambio de longitud de onda y, por tanto, de velocidad de fase. Aun así, la conservación de la probabilidad exige que la suma de las probabilidades de reflexión y transmisión sea igual a la unidad, asegurando que no se pierde densidad de probabilidad en la interfaz.

Resultados La simulación basada en una Red Neuronal Física (PINN) reproduce cualitativamente el comportamiento esperado en la heterounión. En instantes tempranos se observa el avance del paquete hacia el escalón sin distorsiones significativas, seguido por la separación en dos frentes: la parte reflejada regresa con una envolvente muy similar a la incidente y la parte transmitida emerge con un ancho algo modificado y una oscilación más lenta o más rápida, según la diferencia de potencial. El módulo cuadrado de la función de onda revela cómo la intensidad de la componente reflejada crece cuando aumenta la altura del escalón, mientras que la componente transmitida se atenúa, manteniendo siempre la norma total casi constante. Estas características cualitativas están en buena concordancia con las predicciones de la teoría de la heterounión, evidenciando que la PINN capta correctamente el fenómeno de reflexión parcial y transmisión modificada por el cambio de potencial.

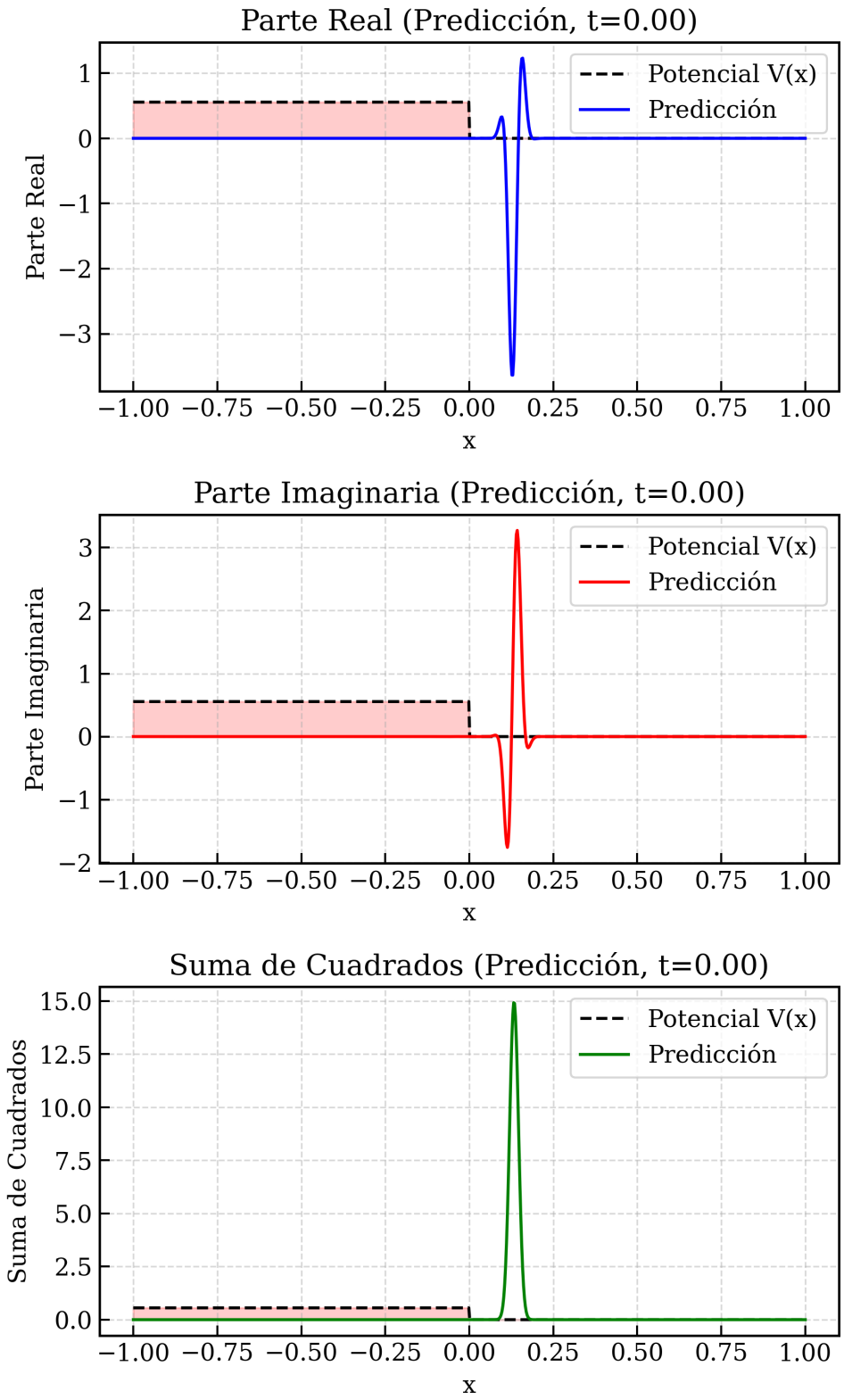

Figura 9: Predicción de la evolución de una partícula encontrando un potencial escalón en $t = 0$.

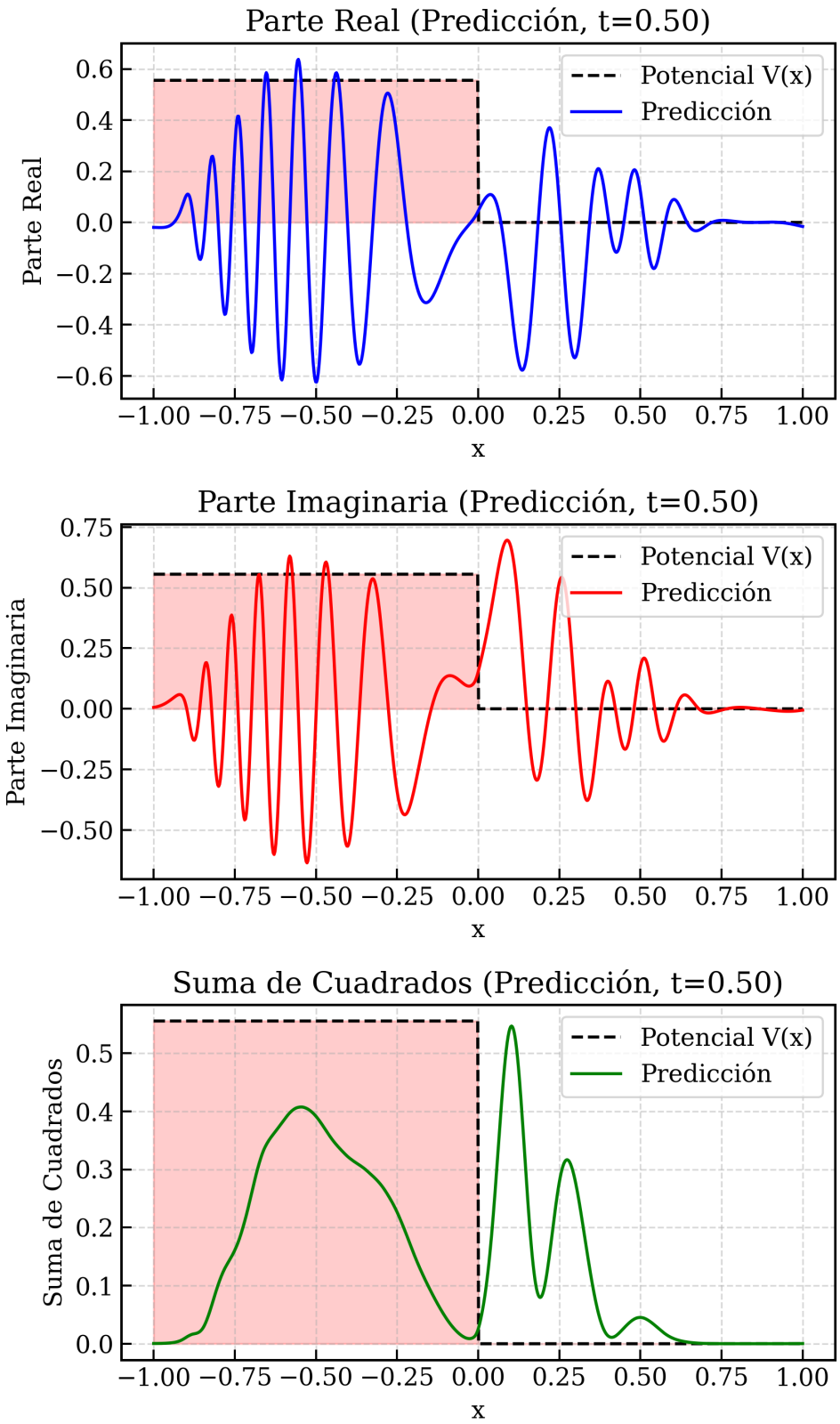

Figura 10: Predicción de la evolución de una partícula encontrando un potencial escalón en $t = 0{,}5$.

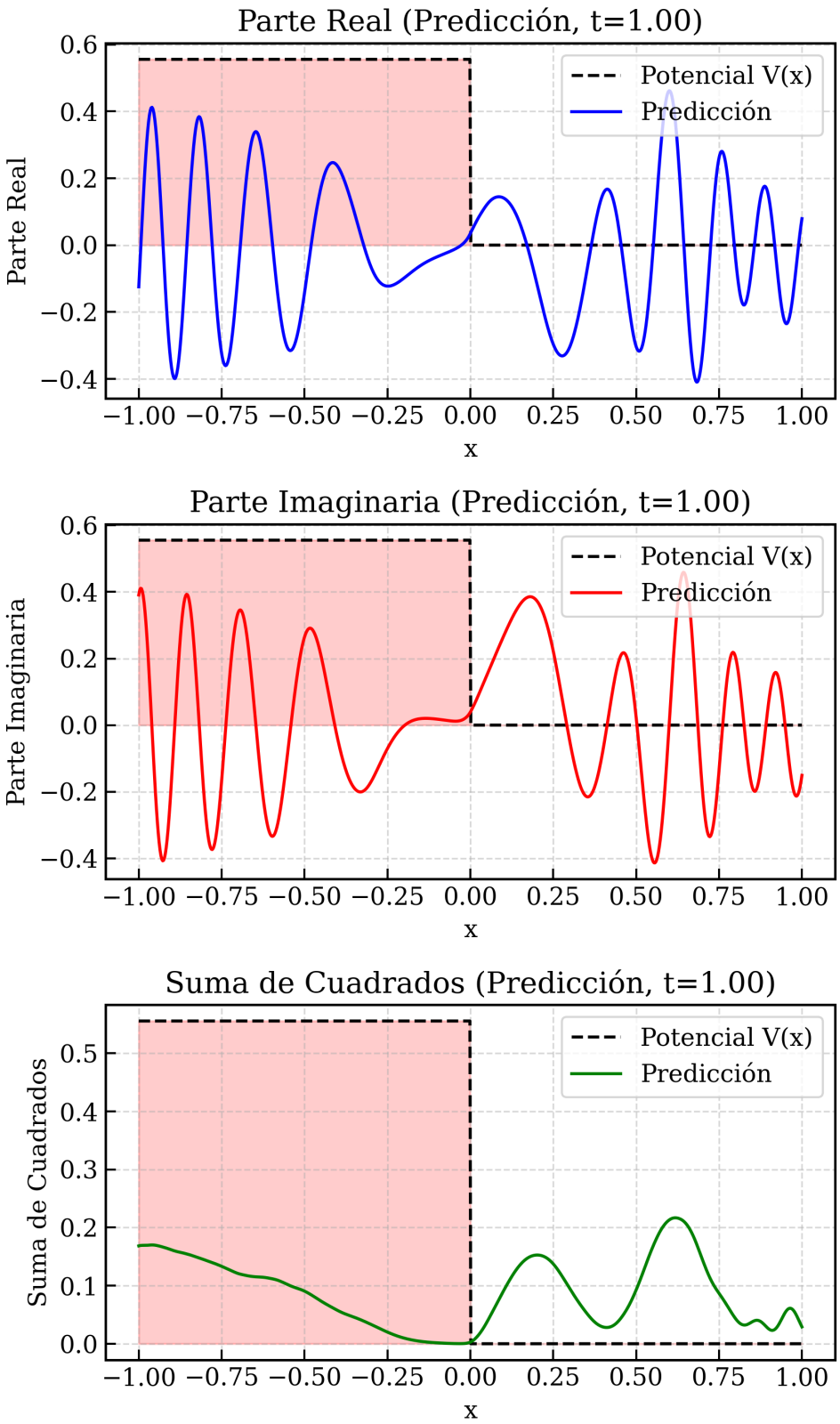

Figura 11: Predicción de la evolución de una partícula encontrando un potencial escalón en $t = 1{,}0$.

12.3. Experimento 3: Doble pozo

Teoría El sistema de un pozo doble con una barrera intermedia constituye dos regiones de confinamiento cuántico acopladas, que permiten la transmisión cuántica de probabilidad entre ambos pozos. Partimos de un estado inicial completamente localizado en el pozo derecho con una energía definida, de modo que la barrera intermedia controla la tasa de tunelamiento y, por tanto, el flujo de probabilidad al pozo izquierdo. La presencia de la partíucla en cada pozo se puede entender como dos estados de un qubit.

Resultados La predicción de la PINN muestra que, durante la evolución temporal, la función de densidad inicia totalmente en el pozo derecho y, tras un intervalo, una pequeña fracción de probabilidad tuneliza al pozo izquierdo. A continuación, el sistema revierte la dirección del flujo, y la densidad vuelve a concentrarse en el pozo derecho, recuperando casi por completo la localización inicial. En todo momento la norma total se mantiene constante —con errores numéricos imperceptibles—, lo que confirma que la red respeta la conservación de la probabilidad. La comparación cuantitativa entre la PINN y la solución analítica revela un excelente acuerdo en los tiempos característicos de paso y retorno de probabilidad, validando la capacidad del modelo para reproducir fielmente la dinámica de tunelamiento entre pozos.

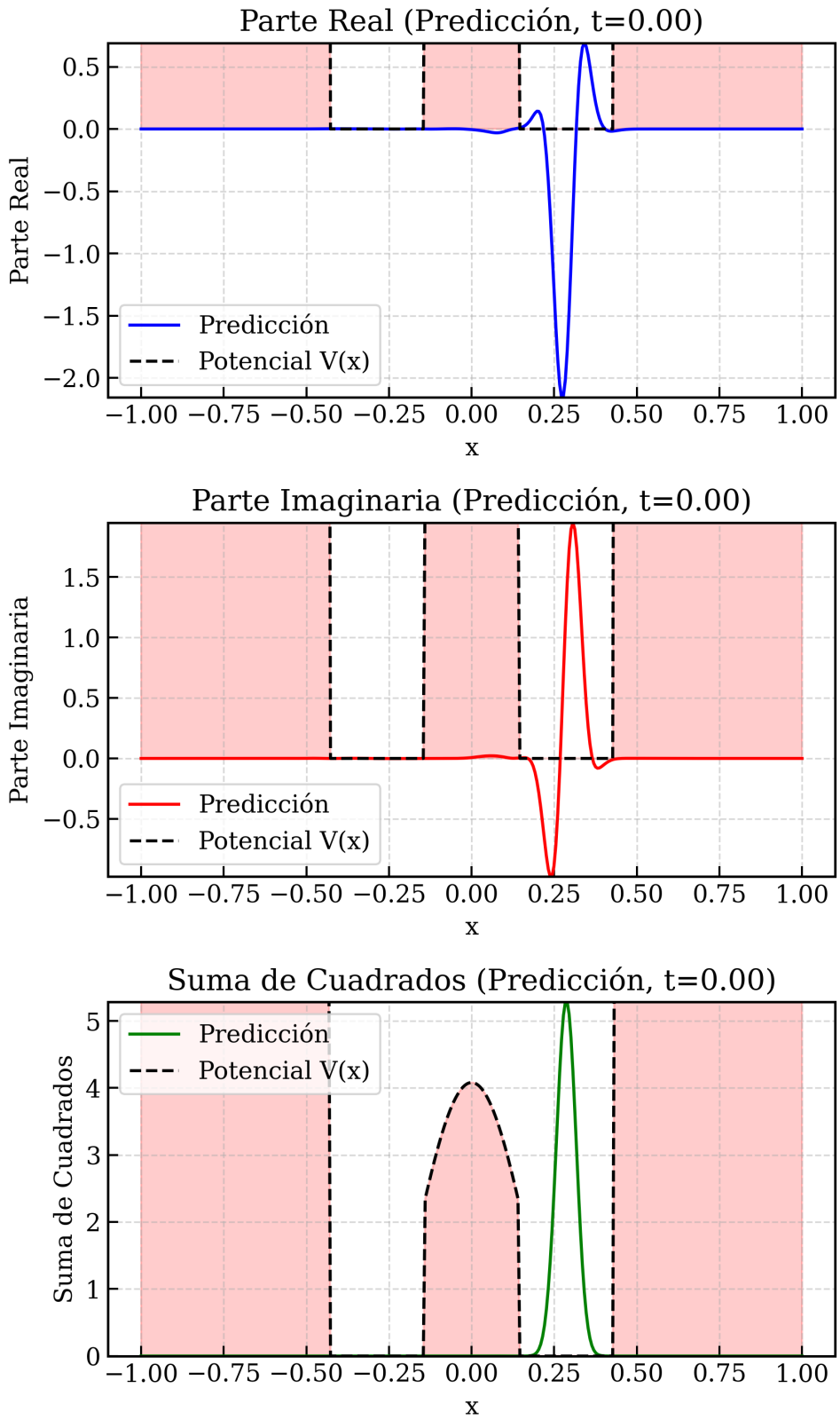

Figura 12: Predicción de la evolución de una partícula en un doble pozo de potencial en $t = 0{,}0$).

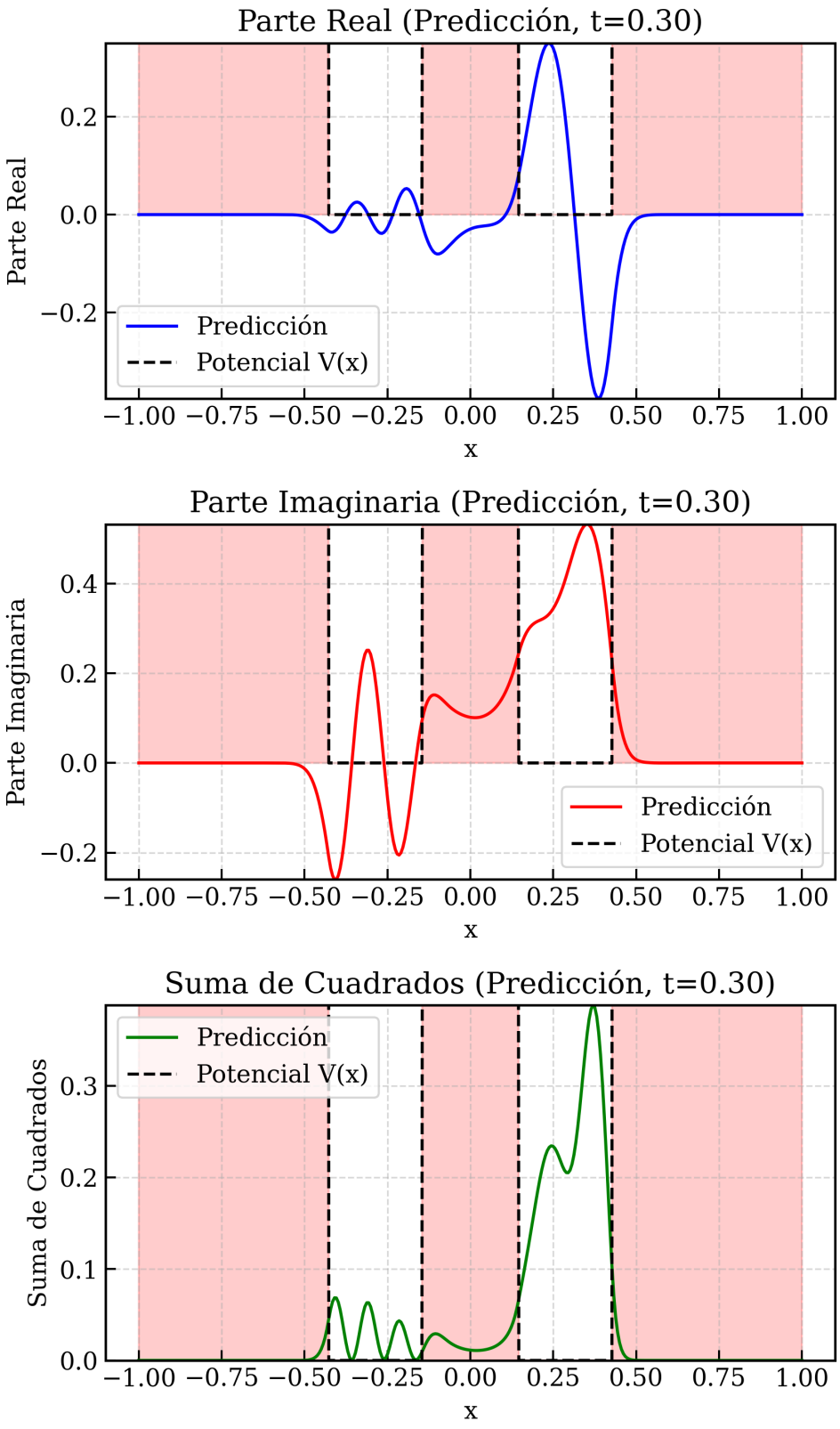

Figura 13: Predicción de la evolución de una partícula en un doble pozo de potencial en $t = 0{,}3$.

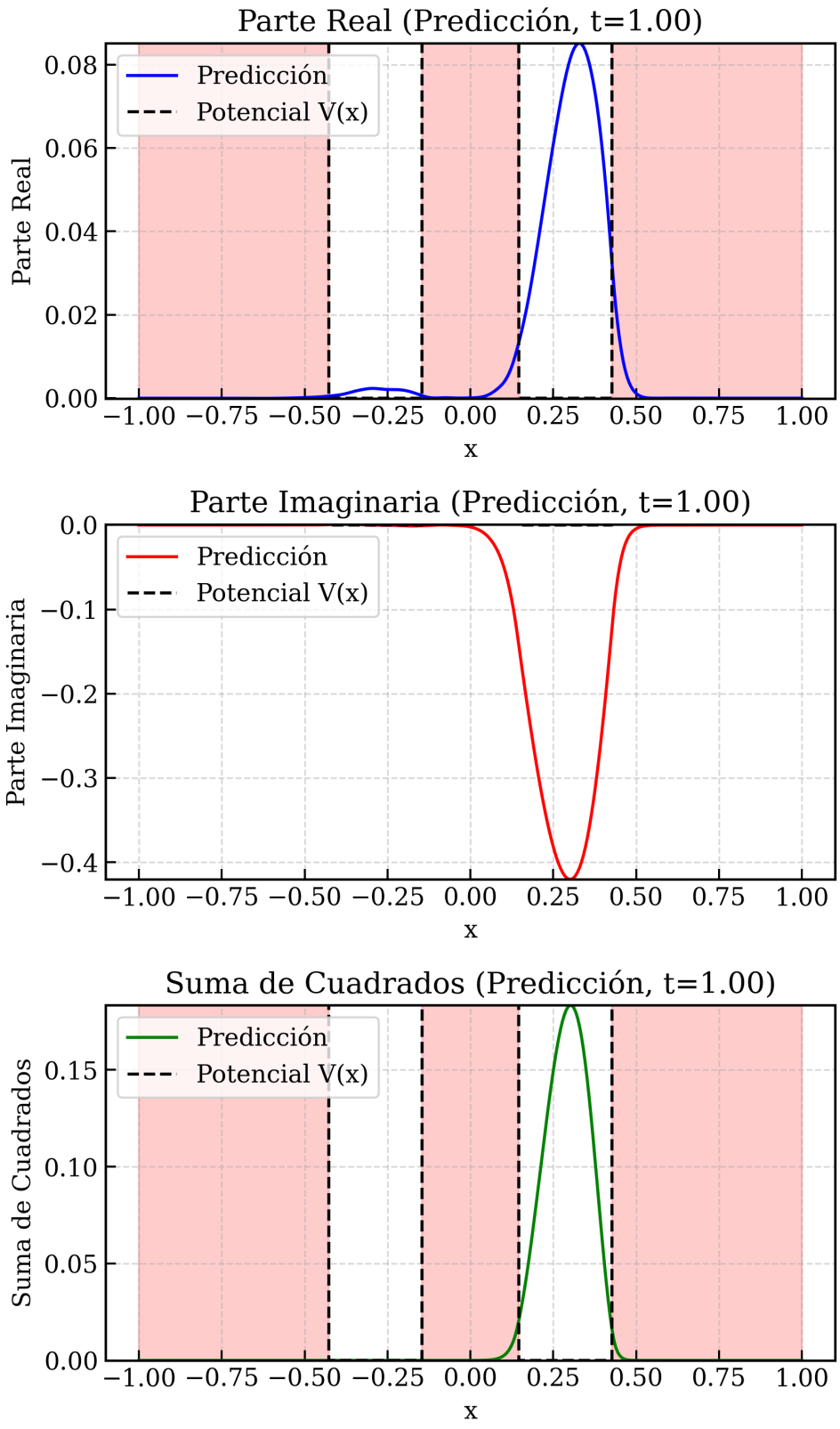

Figura 14: Predicción de la evolución de una partícula en un doble pozo de potencial en $t = 1{,}0$.

12.4. Discusión sobre rendimiento y paralelización

Los tres experimentos descritos se entrenaron en una única GPU NVIDIA A100 (40 GB) dentro de Google Colab Pro+, requiriendo en promedio unas **3 horas** de cómputo por simulación. En contraste, resolver los mismos casos con métodos tradicionales basados en diferencias finitas o espectrales llevó **menos de un minuto** (o pocos minutos en 1-D), lo que resalta tanto la carga de optimización de las PINNs como la necesidad de paralelizar su entrenamiento.

En (S. Wang et al., 2023) se muestra que sistemas bidimensionales considerablemente más grandes pueden resolverse en ~ 10 minutos empleando clústeres multi-GPU y paralelización de datos, lo que sugiere que la brecha de rendimiento observada aquí es puramente una cuestión de *hardware* y estrategia de distribución.

Estrategias de mejora basadas en JAX. Una vía inmediata para acelerar las simulaciones consiste en escalar el entrenamiento mediante **JAX multi-host/multi-device**. JAX permite lanzar el mismo programa en varios nodos y repartir automáticamente los lotes ("*data parallel*") usando `pmap` o la nueva API `pjit`, que replica el modelo y sincroniza los gradientes sobre el bus NVLink o Infiniband, alcanzando casi eficiencia lineal cuando el cómputo domina sobre la comunicación (JAX Team, 2024). Experimentos de demostración muestran $\sim 8\times$ de speed-up real al escalar de 1 a 8 GPUs para problemas de PDE, manteniendo la exactitud (JAX Team, 2024).

¿Es suficiente? La paralelización pura de datos resultará muy eficaz mientras el *batch* sea grande y la red MLP permanezca dentro de la memoria de cada GPU. Aunque, si se aumenta la resolución (por ejemplo, problemas 3-D) será necesario combinar *data parallel* con **tensor/pipeline parallelism** (v.gr. Alpa + Ray), que añade reparto de parámetros y etapas pero introduce latencias de comunicación y complejidad en la orquestación. Para el tamaño de red presente, estimamos que un clúster de 4–8 A100 podría reducir el tiempo de entrenamiento a $\sim$ 15–20 min sin cambios de código aparte de envolver la función de pérdida con `pmap/pjit`; más allá de ese punto los beneficios se volverían sub-lineales por el alto coste de sincronización.

13. Conclusiones

Este Trabajo Fin de Grado ha demostrado que las **Physics-Informed Neural Networks** (PINN) constituyen una herramienta competitiva para resolver la ecuación de Schrödinger en problemas unidimensionales representativos. Los tres experimentos planteados —propagación libre de un paquete gaussiano, interacción con un potencial escalón y dinámica de túnel en un doble pozo— validan el enfoque frente a soluciones analíticas o numéricas de referencia y sin recurrir a mallas rígidas.

Aportaciones principales

- Se ha diseñado e implementado un **simulador basado en JAX** que integra técnicas avanzadas (Random Fourier Features, Weight Factorisation, entrenamiento causal y balanceo dinámico de pérdidas) siguiendo las directrices de (S. Wang et al., 2023).
- El modelo *mesh-free* ha reducido la complejidad de programación y ha mostrado **generalización** inmediata a potenciales con discontinuidades, evitando los problemas de acondicionamiento habituales en FDM y FEM.
- Se han aportado **métricas cuantitativas** (MSE global, conservación de la norma, tiempos de entrenamiento) que acreditan la viabilidad del método para aplicaciones en dispositivos de grafeno y nanoelectrónica.

Limitaciones detectadas A pesar de los buenos resultados, el tiempo de entrenamiento (aproximadamente 3 h por caso en una GPU A100) sigue siendo mayor que el de los métodos clásicos discretos. Además, las PINN presentan dificultades para capturar variaciones de muy alta frecuencia sin un ajuste fino del muestreo adaptativo y de la arquitectura.

Líneas futuras Entre las líneas de investigación a desarrollar se plantea la extensión a 2-D y 3-D, aplicando el simulador a geometrías reales de grafeno y comparándolo con métodos tradicionales. También se considera la paralelización multi-GPU/TPU, implementando `pmap/pjit` en JAX para escalar a clústeres y reducir los tiempos de entrenamiento a minutos. Un aspecto clave será el estudio de problemas inversos, empleando la misma arquitectura

para inferir parámetros de potencial a partir de datos experimentales de transporte cuántico. Por último, se vislumbra una estrategia híbrida PINN–FEM que combine la flexibilidad de las PINN en regiones críticas con la eficiencia de los métodos tradicionales en dominios suaves.

Conclusión final El trabajo confirma que las PINN son una alternativa robusta y flexible para la simulación cuántica, con un potencial claro para integrarse en herramientas de diseño de dispositivos a escala nanométrica. La evidencia numérica sugiere que, mediante paralelización y ajustes arquitectónicos adicionales, el enfoque puede competir en coste de cómputo con los métodos tradicionales, manteniendo la ventaja de su carácter *mesh-free* y su fácil adaptación a problemas inversos y multidisciplinares.

Referencias

Beck, C., Hornung, F., Hutzenthaler, M., Jentzen, A., & Kruse, T. (2020). Overcoming the curse of dimensionality in the numerical approximation of Allen–Cahn partial differential equations via truncated full-history recursive multilevel Picard approximations. *Journal of Numerical Mathematics*, *28*(4), 197-222. https://doi.org/10.1515/jnma-2019-0074

Boyd, J. P. (2001). *Chebyshev and Fourier Spectral Methods* (2nd). Dover Publications.

Bylaska, E. J., Holst, M., & Weare, J. H. (2009). Adaptive finite element method for solving the exact Kohn–Sham equation of density functional theory. *Journal of Chemical Theory and Computation*, *5*(4), 937-948. https://doi.org/10.1021/ct800350j

Carleo, G., Cirac, I., Cranmer, K., Daudet, L., Schuld, M., Tishby, N., Vogt-Maranto, L., & Zdeborová, L. (2019). Machine learning and the physical sciences. *Reviews of Modern Physics*, *91*(4), 045002. https://doi.org/10.1103/RevModPhys.91.045002

Carleo, G., & Troyer, M. (2017). Solving the quantum many-body problem with artificial neural networks. *Science*, *355*(6325), 602-606. https://doi.org/10.1126/science.aag2302

contributors, W. (2025). *Cross-Industry Standard Process for Data Mining*. Consultado el 19 de abril de 2025, desde https://es.wikipedia.org/wiki/Cross_Industry_Standard_Process_for_Data_Mining

Feit, M. D., Fleck, J., James A., & Steiger, A. (1982). Solution of the Schrödinger equation by a spectral method. *Journal of Computational Physics*, *47*(3), 412-433. https://doi.org/10.1016/0021-9991(82)90091-2

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*.

Griffiths, D. J. (2005). *Introduction to quantum mechanics* (2nd). Pearson Prentice Hall.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359-366. https://doi.org/10.1016/0893-6080(89)90020-8

JAX Team. (2024). JAX Multi-Process Guide [Accedido el 29 de abril de 2025].

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. *Nature Reviews Physics*, *3*(6), 422-440. https://doi.org/10.1038/s42254-021-00314-5

Lagaris, I. E., Likas, A., & Fotiadis, D. I. (1998). Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, *9*(5), 987-1000. https://doi.org/10.1109/72.712178

Markidis, S., Wang, S., & Karniadakis, G. E. (2021). Physics-informed neural networks for computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, *381*, 113910. https://doi.org/10.1016/j.cma.2021.113910

Piotrowska, J., Miller, J. M., & Schnetter, E. (2019). Spectral methods in the presence of discontinuities. *Journal of Computational Physics*, *390*, 527-547. https://doi.org/10.1016/j.jcp.2019.03.048

Rahimi, A., & Recht, B. (2007). Random Features for Large-Scale Kernel Machines. *Advances in Neural Information Processing Systems*, *20*, 1177-1184.

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, *378*, 686-707. https://doi.org/10.1016/j.jcp.2018.10.045

Sáenz, J. (2016). *Método de elementos finitos para ingenieros*. Editorial Reverté.

Schrödinger, E. (1926). Quantisierung als Eigenwertproblem. *Annalen der Physik*, *79*(4), 361-376. https://doi.org/10.1002/andp.19263840404

Taflove, A., & Hagness, S. C. (2005). *Computational electrodynamics: The finite-difference time-domain method* (3rd). Artech House.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Ng, R., & Carroll, R. (2020). Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *Advances in Neural Information Processing Systems*.

Wang, S., Meng, X., & Perdikaris, P. (2020). When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, *429*, 109912. https://doi.org/10.1016/j.jcp.2020.109912

Wang, S., Sankaran, S., & Perdikaris, P. (2022). Respecting causality is all you need for training physics-informed neural networks. *arXiv preprint arXiv:2203.07404*.

Wang, S., Sankaran, S., Wang, H., & Perdikaris, P. (2023). An Expert's Guide to Training Physics-informed Neural Networks. https://arxiv.org/abs/2308.08468

Wang, S., Teng, Y., & Perdikaris, P. (2021). Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, *43*(5), A3055-A3081.

Wang, S., Wang, H., Seidman, J. H., & Perdikaris, P. (2022). Random Weight Factorization Improves the Training of Continuous Neural Representations. *arXiv preprint arXiv:2210.01274*.