Asociación Argentina
de Mecánica Computacional

Mecánica Computacional Vol XLII, pp. 797-806
A. Caggiano, G. Etse, P. Folino, M. Goldschmit, M. Pucheta, M. Storti (Eds.)
M. Cruchaga, L. Garelli, E. López, G. Ríos Rodríguez, M. Storti (Issue eds.)
Buenos Aires, November 11-14, 2025

REDES NEURONALES GUIADAS PARA APLICACIONES MODELADAS POR LA ECUACIÓN DE ONDA

NEURAL NETWORKS GUIDED FOR WAVE EQUATION-MODELED APPLICATIONS

Matías Di Liscia[a,*], **Matias A. Vera**[a,b], **Leonardo J. Rey Vega**[a,b] **y Martín G. González**[a,b]

[a]*Universidad de Buenos Aires, Facultad de Ingeniería, Paseo Colón 850, C1063ACV, Buenos Aires, Argentina*

[b]*Consejo Nacional de Investigaciones Científicas y Técnicas, (CONICET), Godoy Cruz 2290, C1425FQB, Buenos Aires, Argentina,*

[*]*mdiliscia@fi.uba.ar*

Palabras clave: PINNs, aprendizaje profundo, ecuación de onda.

Resumen. En los últimos años, la inteligencia artificial y, en particular, las redes neuronales profundas han experimentado avances significativos, permitiendo abordar problemas que anteriormente resultaban computacionalmente inviables. Entre las aplicaciones más prometedoras se encuentran las redes neuronales guiadas por la física (PINNs), que son un tipo de aproximadores de funciones universales que pueden incorporar el conocimiento de cualquier ley física descripta mediante ecuaciones diferenciales que rija un conjunto de datos determinado en el proceso de aprendizaje. En este trabajo se propone evaluar el desempeño de las PINNS de base finita (FBPINNs), para resolver la ecuación de onda unidimensional bajo distintas condiciones de contorno. Los resultados obtenidos son comparados con simulaciones numéricas generadas mediante el software k-Wave. El presente trabajo constituye un primer paso y la perspectiva a futuro es sentar las bases para una futura extensión a dos y tres dimensiones y su aplicación al problema directo de tomografía optoacústica y ultrasónica. Se pueden encontrar las simulaciones utilizadas y ejemplos extra en https://github.com/mdl99-github/FBPINNs-for-Wave-Equation.

Keywords: PINNs, deep learning, wave equation.

Abstract. During the past few years, artificial intelligence and deep neural networks in particular have seen significant breakthroughs, allowing to solve problems that were previously computationally unfeasible. Among the most promising new applications are Physics-Informed Neural Networks (PINNs), which are universal function aproximators that can incorporate knowledge from any phyisical law described by differential ecuations governing a given set of data into the learning process. This paper proposes to evaluate the performance of Finite Basis PINNs (FBPINNs), to solve the unidimensional wave equation under different boundary conditions. Results are compared to numerical simulations generated with the k-Wave software. This work is a first step and the future outlook is to lay the groundwork for a future extension to two and three dimensions and its application to the optoacoustic and ultrasonic tomography direct problem. Simulations and additional examples can be found at https://github.com/mdl99-github/FBPINNs-for-Wave-Equation.

ISSN: 2591-3522 DOI: 10.70567/mc.v42.ocsid8265

1. INTRODUCCIÓN

Si bien en los últimos años han habido grandes avances computacionales que favorecieron el desarrollo de modelos de inteligencia artificial para diversos problemas, la necesidad de grandes cantidades de datos (etiquetados) para entrenar modelos para la resolución de problemas físicos es un inconveniente (Raissi et al., 2019). Las redes neuronales guiadas por la física (Physics-informed neural networks, PINNs) surgen como respuesta a esto. Gracias a la capacidad de poder realizar autodiferenciación (Bayadin et al., 2018) de forma eficiente es posible plantear un modelo que no necesite muestras etiquetadas para el entrenamiento y tenga embebido dentro de él las relaciones físicas del problema. Incluso ya se han planteado PINNs capaces de resolver en particular la ecuación diferencial de onda de manera aproximada y con buenos resultados (Moseley et al., 2020).

Si bien prometedoras, las PINNs clásicas tienen sus limitaciones. Una de ellas es que dada la definición de su función de pérdida, el gradiente calculado por *back-propagation* tiene cierta rigidez que lleva a que la convergencia sea lenta y/o que los gradientes de los términos de la pérdida estén desbalanceados (Wang et al., 2021). Otro gran problema es la dificultad que este tipo de red tiene para lidiar con dominios grandes (Moseley et al., 2023). Dado que cuánto más grande sea el dominio (tanto espacial como temporal) más compleja suele ser la solución, las PINNs necesitan más parámetros y por ende más tiempo de convergencia para llegar a una solución aceptable. Esto se ve acentuado por el efecto de sesgo espectral que tienen los modelos de aprendizaje profundo, la tendencia a aprender mejor soluciones de menor frecuencia (Rahaman et al., 2019). Surgen entonces las redes neuronales guiadas por la física de base finita (Finite basis physics-informed neural networks, FBPINNs) (Moseley et al., 2023) que introducen posibles soluciones a estas limitaciones.

La principal novedad de esta metodología es dividir al dominio en subdominios superpuestos, en donde en cada uno habrá una red neuronal distinta. En principio, la forma de los subdominios y el tipo de red en cada uno puede ser cualquiera, aunque la elección de la función de activación de cada capa puede tener influencia en el proceso de optimización de la red y la calidad de los resultados.

En este trabajo se utiliza el modelo de FBPINNs, con dominios hiperrectangulares y preceptrones multicapa, para resolver el problema directo correspondiente a la ecuación diferencial de onda de manera eficiente. Se presentan propiedades matemáticas que al utilizarse alivianan la carga computacional y se comparan dos enfoques distintos.

2. DESARROLLO FÍSICO-MATEMÁTICO

2.1. Definición del problema

Sea $u(\mathbf{x}, t)$ la onda en la posición $\mathbf{x}$ en el instante t, el problema puede ser modelado a partir de

$$\nabla^2 u(\mathbf{x}, t) - \frac{1}{c^2} \frac{\partial^2 u(\mathbf{x}, t)}{\partial t^2} = 0 \tag{1}$$

donde

$$u(\mathbf{x}, 0) = f(\mathbf{x}), \qquad \left. \frac{\partial u(\mathbf{x}, t)}{\partial t} \right|_{t=0} = 0 \tag{2}$$

donde $f(\mathbf{x})$ es la condición inicial. Se supone un medio homogéneo, con velocidad de propagación c constante y ecuación diferencial homogénea. Por ende, el problema directo consiste en

hallar el valor de la onda $u(\mathbf{x}, t)$ para todo el espacio y en todo tiempo a partir de la condición y velocidad inicial.

En este trabajo nos centraremos en el problema de una dimiensión espacial $x \in \mathbb{R}$ y, por lo tanto, la condiciónón inicial $f(x)$ es una función escalar. Para este conjunto de ecuaciones se puede hallar una solución exacta al problema directo (Miersemann, 2014):

$$u(x,t) = \frac{1}{2}f(x-ct) + \frac{1}{2}f(x+ct) \tag{3}$$

A pesar de que en este trabajo consideraremos esta situación sencilla, todo lo presentado en este trabajo se puede generalizar a situaciones con condiciones de contorno de tipo Dirichlet o Von Neumann modificando la correspondiente función costo usado para entrenar a la red.

2.2. Propiedades

Con el fin de optimizar recursos computacionales se hará uso de ciertas propiedades matemáticas que surgen de la definición del problema.

- **Linealidad:** la linealidad de la ecuación de onda (1) permite entrenar un único modelo para generar una onda con condiciónón inicial arbitraria. Es decir, si $u_i(x,t)$ es solución para todo $i = 1, 2, ..., N$ bajo la condición inicial $f_i(x)$ entonces $v(x,t) = \sum_{i=1}^{N} a_i u_i(x,t)$ es solución para la condición incial $g(x) = \sum_{i=1}^{N} a_i f_i(x)$ con cada $a_i \in \mathbb{R}$.
- **Invariancia en el espacio:** es de utilidad también el hecho de que si $u(x,t)$ es solución generada por la condición inicial $f(x)$ entonces la condición inicial $g(x) = f(x-\mu)$ genera la solución $v(x,t) = u(x-\mu,t)$ para $\mu \in \mathbb{R}$. Cabe destacar que este resultado es válido únicamente para un dominio espacial infinito en donde no hay efectos de borde con medio homogéneo.
- **Escalamiento temporal:** si $u(x,t)$ es solución de la ecuación (1) y es generada por la condición inicial $f(x)$ con velocidad de propagación c_u entonces $v(x,t) = u(x, \frac{c_v}{c_u}t)$ también es solución generada por $f(x)$ para velocidad de propagación c_v. Si bien se supone al medio homogéneo con velocidad de propagación constante para todo x en el dominio de trabajo, tener una forma de relacionar soluciones con distinto c ahorra entrenamiento de modelos nuevos.

2.3. Reconstrucción

La linealidad nos permite descomponer cualquier condición inicial a partir de una familia de funciones base adecuada que permita reconstruir una familia funciones a la cual pertenecerán las condiciones iniciales en las que podamos estar interesados. Es decir:

$$f(x) \approx \sum_{i=1}^{N} a_i b_i(x) \tag{4}$$

con a_i coeficientes a determinar y $\{b_1, b_2, ..., b_N\}$ las funciones base, para $x \in (l, L)$ con $l, L \in \mathbb{R}$ los extremos del dominio en donde las condiciones iniciales estarán definidas. La ventaja de usar esta aproximación va a ser que para resolver el problema directo solamente hay que entrenar una red neuronal con un único tipo de funciones para cualquier función que se quiera como condiciónón inicial (dentro de la familia mencionada arriba).

En este trabajo nos centraremos en las siguientes funciones base:

- **Pulsos triangulares:** a partir de la función $\text{tri}(x) = \text{máx}(1 - |x|, 0)$, se define una interpolación lineal o de orden uno. En este caso si se tienen N muestras equiespaciadas de la función en el intervalo (l, L) y se define la constante $A = \frac{L-l}{N}$ que representa la mitad del ancho del pulso entonces:

$$f(x) \approx \sum_{i=1}^{N} f((i-1)A - l) \cdot \text{tri}\left(\frac{x_i - (i-1)\cdot A - l}{A}\right) \tag{5}$$

  Sin embargo, se podrían utilizar pulsos de ancho arbitrario, distinto a A, que aunque no resultaría exactamente en una interpolación lineal, su calidad de aproximación es aceptable:

$$f(x) \approx \sum_{i=1}^{N} a_i \cdot \text{tri}\left(\frac{x - \mu_i}{A^*}\right) \tag{6}$$

  y donde los coeficientes son solución del problema de optimización:

$$\underset{a_j,\mu_j}{\arg\min} \sum_{i=1}^{N} \left| f(x_i) - \sum_{j=1}^{M} a_j \cdot \text{tri}\left(\frac{x_i - \mu_j}{A^*}\right)\right|^2 \tag{7}$$

  con $i = 1, ..., N$, $j = 1, ..., M$ y $A \neq A^*$.

- **Pulsos gaussianos:** dada su diferenciabilidad en todo su dominio y su alta capacidad de aproximación (Calcaterra y Boldt, 2008) una función puede reconstruirse como:

$$f(x) \approx \sum_{i=1}^{N} a_i e^{-\frac{(x-\mu_i)^2}{2\sigma^2}} \tag{8}$$

  En donde las medias μ_i están prefijadas y equiespaciadas entre sí y donde $2\sigma^2 = 1$, para todo $f \in L^2(\mathbb{R})$. De todas formas, esto puede extenderse para el caso donde σ^2 y las medias son arbitrarios realizando la minimización de manera numérica:

$$\underset{a_j,\mu_j}{\arg\min} \sum_{i=1}^{N} \left| f(x_i) - \sum_{j=1}^{M} a_j e^{-\frac{(x_i-\mu_j)^2}{2\sigma^2}}\right|^2 \tag{9}$$

3. PHYSICS-INFORMED NEURAL NETWORKS

Sea la ecuación diferencial parcial en su forma residual:

$$\mathcal{F}\left(u, x_1, x_2, ..., x_d, \left\{\frac{\partial^k u}{\partial x_i^k}\right\}_{k=1,i=1}^{M,d}\right) = 0, \tag{10}$$

en donde $u(x_1, x_2, ..., x_d)$ es la solución a la ecuación deseada, $(x_1, x_2, ..., x_d) \in \Omega$ las variables asociadas al dominio de la función y $\left\{\frac{\partial^k u}{\partial x_i^k}\right\}_{k=1,i=1}^{M,d}$ las derivadas parciales de u asociadas a cada variable de entrada hasta orden M. Las PINNs definen una arquitectura de red neuronal (habitualmente perceptrones multicapa) cuya pérdida sea de la forma:

$$\mathcal{L} = \lambda_{PDE}\mathcal{L}_{PDE} + \lambda_{data}\mathcal{L}_{data} \tag{11}$$

$$\mathcal{L}_{PDE} = \frac{1}{N}\sum_{i=1}^{N} |\mathcal{F}(u_\theta(\mathbf{x_i}))|^2 \tag{12}$$

$$\mathcal{L}_{data} = \frac{1}{M}[\sum_{\mathbf{x_j}\in\partial\Omega} |u_\theta(\mathbf{x_j}) - u(\mathbf{x_j})|^2 + \sum_{\mathbf{x_k}\in\partial\Omega} |u_{x_1,\theta}(\mathbf{x_k}) - u_{x_1}(\mathbf{x_k})|^2 + ...] \tag{13}$$

Con $\lambda_{PDE}, \lambda_{data}$ constantes para ajustar el peso de cada término, $\mathbf{x_i} \in \Omega$ puntos muestreados en el dominio, $\partial\Omega$ su frontera y u_θ la aproximación a la solución de la red. El término $\mathcal{L}_{PDE}$ penaliza si la solución no se ajusta a la ecuación diferencial, por lo que busca que evolucione de acuerdo a la dinámica definida por la ecuación, y el término $\mathcal{L}_{data}$ apunta a que se cumplan las coindiciones de contorno. Como se ve, aprovechando la diferenciación automática solamente con muestrear puntos dentro del dominio y en sus fronteras se puede plantear un aprendizaje similar a uno no supervisado. Dado que si bien técnicamente las muestras en los bordes necesitan de valores de u (o sus derivadas) exactos, éstas ya están definidas desde el planteo de la ecuación diferencial a resolver, por lo que son conocidos.

Una alternativa a utilizar puntos de las fronteras del dominio, es resolver el mismo problema de manera totalmente no supervisada. Se define el *ansatz* de la solución:

$$u_\theta(\mathbf{x}) = A(\mathbf{x}) + B(\mathbf{x})u^*_\theta(\mathbf{x}) \tag{14}$$

donde A, B son funciones a determinar, $u^*_\theta(\mathbf{x})$ la salida de las últimas neuronas de la red y $u_\theta(\mathbf{x})$ la salida final sobre la cual se calculará la pérdida. Si se eligen A, B de manera que u_θ satisfaga las condiciones de contorno entonces se podría realizar el entrenamiento con el término correspondiente a la ecuación (12) únicamente. A este tipo de red se la denota *hard* PINN mientras que al primero, donde se muestrea en la frontera, se los denomina *vanilla* PINN (Baty, 2024).

3.1. Finite basis physics-informed neural networks

Como se mencionó anteriormente, la característica distintiva de este modelo es la división del dominio en subdominios (ver Fig. 1) en donde en cada uno trabaja una red neuronal distinta. Esto ayuda a solucionar el problema de sesgo espectral permitiendo que cada red "vea" una frecuencia de solución menor gracias a trabajar sobre una parte reducida del dominio. Teniendo esta división en subdominios, en donde a cada uno le corresponde una red PINN distinta, la salida completa de una red FBPINN es:

$$u_\theta(\mathbf{x}) = \sum_{s=1}^{S} W_s(\mathbf{x}).unnorm\left(u_{s,\theta}(norm_s(\mathbf{x}))\right) \tag{15}$$

donde S es la cantidad de subdominios, $u_{s,\theta}$ la salida de la red correspondiente al subdominio s-esimo, $norm_s$ la normalización de la entrada para cada subdominio que asegura que se mantenga entre $[-1, 1]$ y $unnorm$ la desnormalización general. La ventana W_s asegura que la solución de cada subdominio quede confinada dentro él y que la superposición sea suave. El proceso de aprendizaje y la minimización de la pérdida se hace sobre esta salida y no sobre la salida individual de cada subred. De esta forma, la red neuronal aprenderá a modificar cada término en (15) para que su suma en las fronteras de los subdominios esa la adecuada.

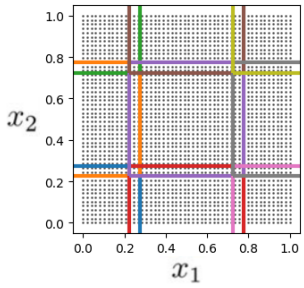

Figura 1: Descomposición de un dominio. Se tienen 3x3 subdominios, con superposiciones de 0.05 unidades tanto en la primera como en la segunda dimensión. $x_1, x_2 \in [0, 1]$.

Otra novedad que presentan las FBPINNs es la posibilidad de usar un planificador. Dada la dificultad que tienen las PINNs de aprender la solución en la frontera, el planificador permite entrenar primero las redes correspondientes a subdominios en los bordes y luego ir barriendo desde allí, como se ejemplifica en la Fig. 2. Esto fortalece la capacidad de la red de evitar que se llegue a un mínimo local cuya solución no corresponda a la condición de contorno fijada (Moseley et al., 2023).

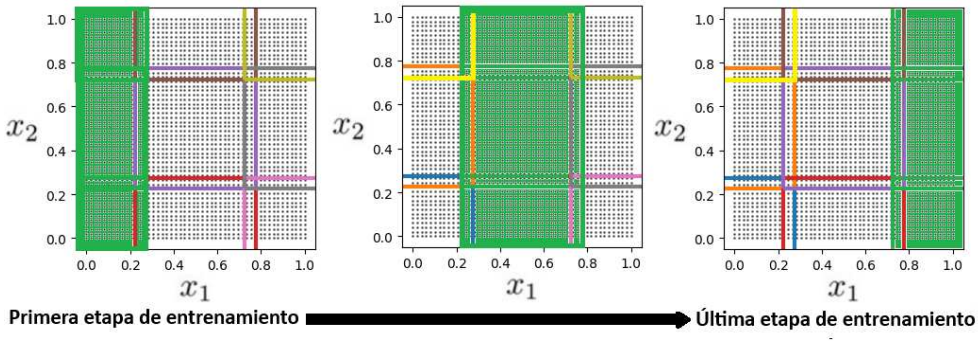

Figura 2: Evolución del entrenamiento con planificador activo. En verde se marcan los subdominios activos a medida que avanza el entrenamiento.

Finalmente, en este trabajo se utilizará la salida de las FBPINNs para aproximar las funciones base de acuerdo a la reconstrucción de la condición inicial (ver Sección 2.3) y las propiedades de la Sección 2.2 para obtener la solución completa deseada:

$$u(x,t) = \sum_{i=1}^{N} a_i u_{\theta,i}(x,t) \tag{16}$$

Con a_i los coeficientes de la reconstrucción y $u_{\theta,i}(x,t)$ las salidas de la FBPINN en (15) que representan las funciones base. Estas últimas podrán ser obtenidas directamente de la salida de la red, como en la Sección 4.1, o tomando la salida de la red y aplicándole la propiedad de invariancia en el tiempo, como en la Sección 4.2.

4. RESULTADOS

En esta sección se presentan los resultados correspondientes al entrenamiento de FBPINNs para pulsos gaussianos y triangulares, utilizando las propiedades descriptas en la Sección 2.2 para reconstruir funciones arbitrarias como condiciones iniciales. Para las simulaciones se usó velocidad de propagación $c = 1\frac{m}{s}$ y los siguientes intervalos para los dominios espacial y

temporal $x \in [-50, 50], t \in [0, 50]$. Con las redes ya entrenadas, se procede a reconstruir dos condiciones iniciales arbitrarias con las funciones base correspondientes: $p_0^1(x) = (\cos(2\frac{\pi}{40}x) + 2)\mathbf{1}\{-40 < x < 40\}$ y $p_0^2(x) = \sin(2\frac{\pi}{40}x)\mathbf{1}\{-40 < x < 40\}$. Se comparan los resultados obtenidos con los generados con el software de simulación k-Wave[1].

4.1. Pulsos gaussianos

En vez de utilizar la propiedad de invariancia en el tiempo para entrenar un único pulso y luego desplazarlo para encontrar las funciones base para la reconstrucción, se plantea una alternativa. En vez de usar como entrada de la red el dominio del problema físico (x, t) se utiliza (x, t, μ) en donde el último elemento representa la posición del pulso. De esta manera, la red logra aprender la solución para cualquier posición de pulso inicial dentro del dominio físico.

Se tienen 4 subdominios en la primera dimensión del dominio x, 4 en la segunda dimensión t y 4 en la tercera dimensión μ, es decir, 4x4x4 subdominios de 100, 50 y 98 unidades de ancho con centros equiespaciados y forma cúbica, se muestrearon 40x40x40 puntos durante el entrenamiento, sin utilizar planificador. Las redes eran perceptrones multicapa con activación tangente hiperbólica y 4 capas ocultas de 8 neuronas. Se utilizaron *hard* PINNs con *ansatz*:

$$u_\theta(\mathbf{x}) = \phi\left(5\left(2 - \frac{t}{t_1}\right)\right) f(x) + \tanh\left(\frac{t}{t_1}\right) u_\theta^*(\mathbf{x}) \tag{17}$$

Con $f(x) = e^{-\frac{(x-\mu)^2}{2\cdot 4^2}}$ la condiciónón inicial, $t_1 = \frac{\sigma}{c} = 4$ y $\phi(.)$ la función sigmoide. Notar que esta función cumple de manera aproximada las condiciones de contorno, pero la función sigmoide es muy cercana a 1 cuando $t = 0$ y muy cercana a 0 cuando $t \gg 2t_1$ (Moseley et al., 2023). Entrenando por 80000 épocas se obtuvo un error cuadrático medio de $1{,}337 \cdot 10^{-4}$ para el conjunto de testeo. Se puede ver una solución en particular en la Figura 3.

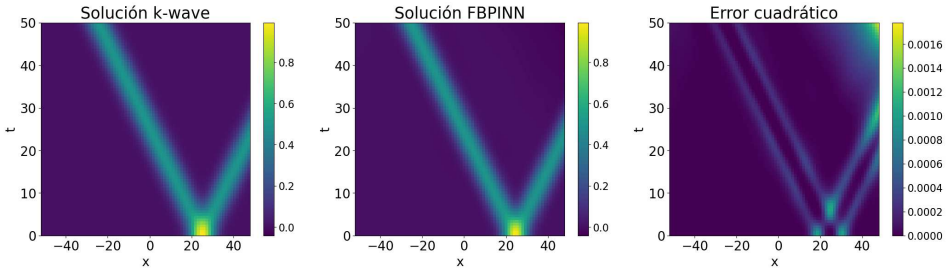

Figura 3: Comparación entre solución de FBPINNs y k-Wave para pulso inicial gaussiano centrado en $\mu = 25m$.

En cuanto a las ondas generadas por p_0^1 y p_0^2 (ver Figura 4) se obtienen errores del mismo orden para el seno y se aprecian los problemas por la discontinuidad del coseno.

4.2. Pulsos triangulares

Ahora se entrena un único pulso triangular de ancho 16 en la base y se utiliza la propiedad de invariancia para encontrar las funciones base. Se tienen 4x10 subdominios de 99, 16.67 unidades de ancho con centros equiespaciados y forma rectangular, se muestrearon 100x100 puntos durante el entrenamiento, y se usó planificador comenzando desde $t = 0$. Las redes eran perceptrones multicapa con activación tangente hiperbólica y 6 capas ocultas de 16 neuronas,

[1] http://www.k-wave.org/

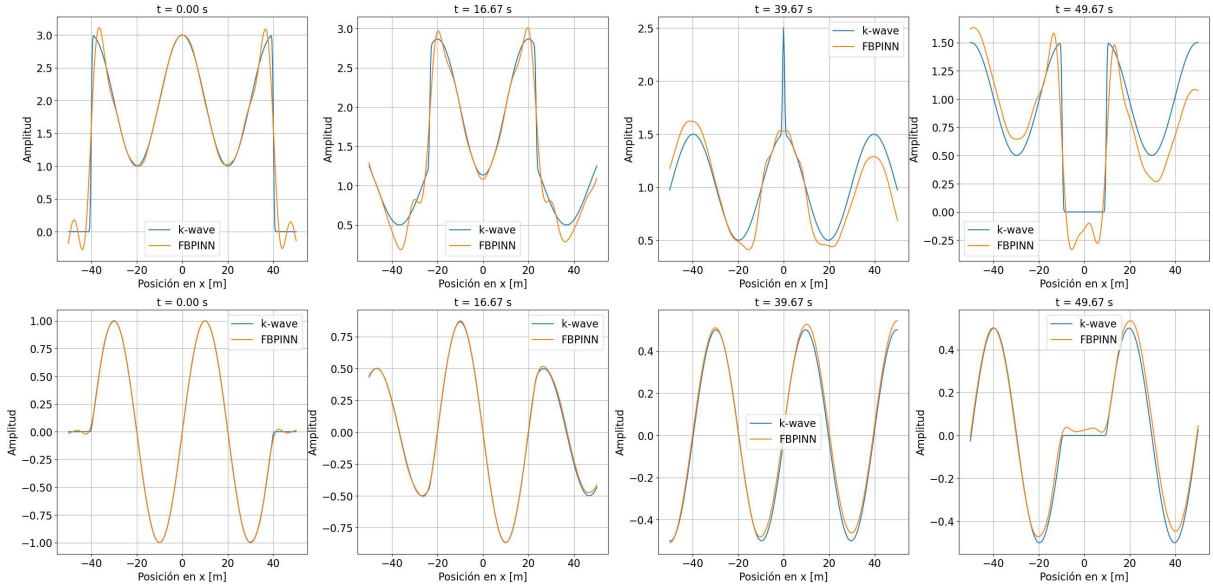

Figura 4: Soluciones para condiciones iniciales p_0^1 (arriba) y p_0^2 (abajo) a lo largo del tiempo comparadas con las obtenidas con el software k-Wave - Base de pulsos gaussianos.

salvo la primera con 32. Se utilizaron *vanilla* PINNs, y se entrenó para $x \in [-60, 60]$ para que se vea el pulso completo y se pueda desplazar, como se muestra en la Figura 5. Se entrenó por 100000 épocas obteniéndose al finalizar un error cuadrático medio de $3{,}157 \cdot 10^{-4}$.

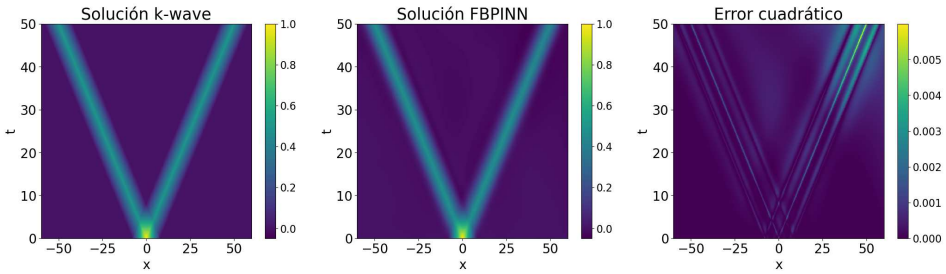

Figura 5: Comparación entre solución de FBPINNs y k-Wave para pulso inicial triangular centrado en $\mu = 0m$.

La solución con los pulsos iniciales senoidales se aprecian en la Figura 6. Se destaca el problema para tiempos altos que tiene la red, que se hacen evidentes en la onda generada por p_0^1. La desventaja de entrenar un único pulso es que los problemas que tenga esa aproximación los van a tener todas las funciones base, y se replicarán en las reconstrucciones.

4.3. Análisis de resultados

Se muestra para finalizar esta sección un cuadro comparativo (Tabla 1) con información relevante para cada modelo.

En primer lugar se puede apreciar la dificultad que tienen ambos modelos para aproximar la onda generada por el coseno. Esto no se debe a un problema de la red neuronal en sí sino a que la reconstrución de la condición inicial es mala gracias a la discontinuidad del coseno en -40m y 40m. Con el seno no sucede esto y se obtienen errores cuadráticos medios dos órdenes de magnitud menores como consecuencia.

Otro punto importante a notar es la amplia diferencia entre los tiempos de entrenamiento de ambos modelos. El entrenar para todas las medias posibles (modelo gaussiano) hace que la red

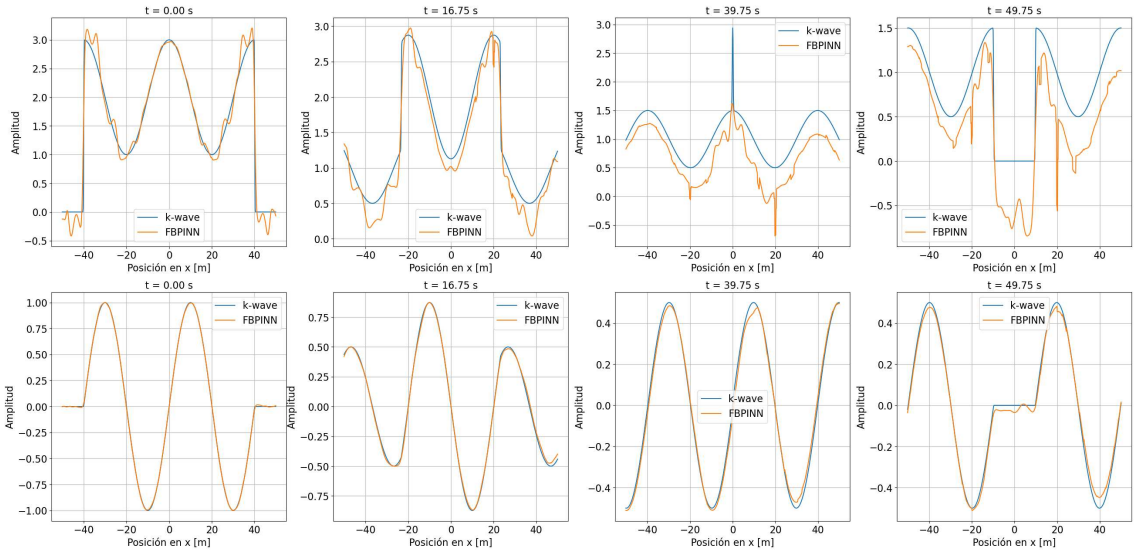

Figura 6: Soluciones para condiciones iniciales p_0^1 (arriba) y p_0^2 (abajo) a lo largo del tiempo comparadas con las obtenidas con el software k-Wave - Base de pulsos triangulares.

tenga una dimensión de entrada adicional complejizando el problema y resultando en tiempos altos. Para el modelo triangular el tiempo es significativamente menor ya que se entrena un único pulso que luego se desplaza. Sin embargo, notar que la cantidad de parámetros por subdominio es mayor para el triangular, y se debe a que aprender ondas con discontinuidades en la derivada primera es una tarea más desafiante para la red.

	Modelo gaussiano	Modelo triangular
ECM p_0^1 inicial	6.866e-02	2.316e-02
ECM p_0^2 inicial	8.983e-05	4.019e-05
ECM p_0^1 completo	2.266e-02	9.386e-02
ECM p_0^2 completo	4.809e-04	4.196e-04
# Parámetros	47168	69160
#Parámetros por subdominio	737	1729
# Épocas	80000	100000
Tiempo de entrenamiento	511 mins	29 mins

Tabla 1: Comparación de modelos.

Si bien el costo de entrenamiento es importante, el proceso se tiene que realizar una única vez. El utilizar las FBPINNs para obtener las funciones base permite que se usen ellas para reconstruir cualquier condición inicial sin necesidad de reentrenar. Con un software de resolución numérica tradicional, se tiene que generar una simulación por cada solución. Además, una vez definida la grilla de puntos del dominio ésta queda fija y no se pueden obtener puntos en otros sitios. Con la metodología presentada eso tampoco es un problema ya que la red fue entrenada para recibir cualquier punto del dominio y devolver la solución aproximada, independientemente de los puntos que se hayan usado para entrenar.

5. CONCLUSIONES Y FUTUROS TRABAJOS

En este trabajo, se presentó un método para la resolución del problema directo de la ecuación diferencial de onda unidimensional. Analizando los resultados, se los considera prometedores y dignos de seguir explorando con el fin de obtener una extensión a dos y tres dimensiones del modelo.

En el futuro cercano, se pretende utilizar estos resultados como complemento en la generación de imágenes de tomografía optoacústica (TOA), que consiste en la resolución inversa de la ecuación diferencial de onda. En la TOA se ilumina el tejido de interés y se mide la onda acústica generada en respuesta a esto a un tiempo posterior en la ubicación de los sensores. Se busca entonces reconstruir sus condiciones iniciales que son las que definen la imagen del tejido (Oraevsky et al., 1996). Dado que un método posible para esto es mediante redes neuronales profundas, es deseable una manera de generar muestras etiquetadas para el entrenamiento. Finalmente, los resultados expuestos en este trabajo serán la base para diseñar un programa capaz de generar muestras sintéticas de TOA confiables, que podrán ser utilizadas para la resolución del problema inverso.

REFERENCIAS

Baty H. A hands-on introduction to physics-informed neural networks for solving partial differential equations with benchmark tests taken from astrophysics and plasma physics. *arXiv preprint arXiv:2403.00599*, 2024.

Bayadin A., Pearlmutter B., Radul A., y Siskind J. Automatic differentiation in machine learning: a survey. *Journal of Machine Learning Research*, 18:1–43, 2018. http://doi.org/https://doi.org/10.48550/arXiv.1502.05767.

Calcaterra C. y Boldt A. Approximating with Gaussians. 2008. http://doi.org/10.48550/arXiv.0805.3795.

Miersemann E. *Partial Differential Equations*. CreateSpace Independent Publishing Platform, 2014.

Moseley B., Markham A., y Nissen-Meyer T. Solving the wave equation with physics-informed deep learning. 2020. http://doi.org/10.48550/arXiv.2006.11894.

Moseley B., Markham A., y Nissen-Meyer T. Finite basis physics-informed neural networks (FBPINNs): a scalable domain decomposition approach for solving differential equations. *Advances in Computational Mathematics*, 49:62, 2023. http://doi.org/10.1007/s10444-023-10065-9.

Oraevsky A., Esenaliev R., Jacques S., y Tittel F. Laser opto-acoustic tomography for medical diagnostics: Principles. *Proceedings of SPIE - The International Society for Optical Engineering*, 1996. http://doi.org/10.1117/12.238786.

Rahaman N., Baratin A., Arpit D., Draxler F., Lin M., Hamprecht F., Bengio Y., y Courville A. On the spectral bias of neural networks. En *International conference on machine learning*, páginas 5301–5310. PMLR, 2019. http://doi.org/https://doi.org/10.48550/arXiv.1806.08734.

Raissi M., Perdikaris P., y Karniadakis G.E. 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, 2019. http://doi.org/https://doi.org/10.1016/j.jcp.2018.10.045.

Wang S., Teng Y., y Perdikaris P. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43:A3055–A3081, 2021. http://doi.org/10.1137/20M1318043.