**Evaluación de Desempeño: Diferencias Finitas vs Redes Informadas por Física sobre la Ecuación de Divergencia**

**Fabricio Lopretto (a1616)**

Quinto Bimestre 2024

Resumen

A completar…

Palabras clave: diferencias finitas, PINN, a completar…

1. **OBJETIVOS**

Comparar el desempeño entre los métodos de diferencias finitas y de redes neuronales informadas por física para la búsqueda de la solución de las ecuaciones diferenciales propuestas, utilizar la solución exacta de heurística cuando corresponda.

1. Hallar la solución por ambos métodos para tres tamaños diferentes de grilla, proponiendo diferentes arquitecturas para el método con redes neuronales informadas por física.
2. Utilizar muestreos aleatorios de los puntos de colocación con la misma cantidad de muestras que en el caso de las grillas uniformes cuando corresponda.

**2. INTRODUCCIÓN**

La resolución de ecuaciones en derivadas parciales (EDPs) es fundamental en el modelado de fenómenos físicos y procesos complejos en diversas áreas de la ciencia e ingeniería. Tradicionalmente, métodos numéricos como las diferencias finitas (FDM) han sido ampliamente utilizados para aproximar soluciones a este tipo de ecuaciones, ofreciendo una precisión que depende del tamaño de la grilla y del orden de la aproximación [2]. Sin embargo, en los últimos años ha surgido una nueva metodología que aprovecha el poder de las redes neuronales: las Redes Neuronales Informadas por Física (RNIP). Las PINNs permiten abordar EDPs mediante la incorporación de las ecuaciones en el entrenamiento de la red, logrando así soluciones que respetan los principios físicos del problema [3].

En este trabajo, se propone un análisis comparativo entre dos enfoques de diferenciación: la diferenciación automática (autodiff) y el método de diferencias finitas, aplicados a la resolución de problemas de estado estacionario. La diferenciación automática facilita el cálculo de derivadas parciales de manera precisa y eficiente mediante la construcción de grafos computacionales [1], siendo una herramienta fundamental para entrenar PINNs. Por otro lado, el método de diferencias finitas utiliza aproximaciones incrementales para estimar derivadas, lo que permite su implementación en una variedad de problemas discretizados.

Para esta comparación, se emplean dos problemas representativos: uno relacionado con una función sinusoidal y otro con un modelo de conducción de calor con un término fuente no lineal. Cada problema se resuelve tanto con una PINN como con el método de diferencias finitas sobre grillas de distintos tamaños y configuraciones de red neuronal, y los resultados se contrastan con soluciones exactas. A través de este enfoque, se busca evaluar la precisión y las características computacionales de cada método, identificando ventajas y limitaciones en función de la configuración y la naturaleza del problema.

Este estudio tiene como objetivo proporcionar una visión integral sobre la efectividad de las PINNs combinadas con autodiff frente al método clásico de diferencias finitas, formando parte de la base para futuras investigaciones en la resolución eficiente de EDPs mediante herramientas de inteligencia artificial.

**3. METODOLOGÍA3.1 Método mediante diferencias finitas**

El método se basa en el polinomio de Taylor, que su expresión para una función suave de una variable afirma que:

A partir de la misma, es posible expandir hasta segundo orden alrededor de un punto evaluándose en y

(1)

(2)

Luego, para encontrar una aproximación de , se restan las ecuaciones anteriores de la siguiente manera:

Quedando:

**(3)**

De manera similar, para obtener una aproximación de , se suman las ecuaciones (1) y (2):

Resultando:

(4)

En este trabajo se proponen dos formas de la ecuación de Poisson:

(5)

Con las siguientes condiciones de borde:

y (6)

Por otro lado, se utiliza:

Con las siguientes condiciones de borde:

y

En cualquiera de los dos casos, la ecuación de Poisson se puede generalizar de la siguiente manera:

Donde:

es la función desconocida que queremos calcular.

es la función de la divergencia propuestas por (5) y (6). El operador Laplaciano bidimensional, que involucra las segundas derivadas parciales de *u* respecto a *x* e *y*.

Es posible utilizar la ecuación (4) para aproximar las expresiones de y

Usando una grilla uniforme con espaciado *dx* en la dirección *x*, la segunda derivada de *u* respecto a *x* en el punto *(i,j)* (es decir, en el punto con coordenadas *xi*​ e *yj*​) se aproxima por:

(7)

De manera similar, para la segunda derivada de *u* respecto a *y*, usando un espaciado *dy* en la dirección *y*, se tiene:

(8)

Entonces, la ecuación de Poisson se puede escribir la suma de las discretizaciones de las segundas derivadas en *x* e *y* dadas por la suma de las ecuaciones (7) y (8):

Resolviendo por diferencias finitas se tiene que *dx→x* y *dy→y*. Además, al usar una grilla de puntos equiespaciada se asume que *x=y.* A partir de esto, se tiene que:

Despejando *u(x,y)*, resulta:

(9)

Reemplazando según corresponda el caso (5) o (6).

Utilizando la ecuación (9), se realizaron tres experimentos en los que se buscó la solución de *u(x,y)* mediante este método en tres grillas equiespaciadas: 5x5, 10x10 y 20x20. Para ello, se inicializó la solución uuu en cero en cada punto de la grilla y luego se aplicó la ecuación (9) en cada uno de ellos mediante un bucle. En cada iteración, se recorrieron todos los puntos de la grilla para calcular *u* y verificar un criterio de convergencia basado en un umbral determinado por la diferencia entre el valor calculado y el valor de la iteración anterior. Si esta diferencia cae por debajo del umbral, el bucle se finaliza y se considera que se ha alcanzado la solución. En caso contrario, si no se logra convergencia en la solución, el bucle finaliza tras un máximo de 10.000 iteraciones.

**3.2 Método mediante redes neuronales informadas por física**

En este trabajo, se emplea el enfoque de redes neuronales informadas por física (PINN) para resolver la ecuación diferencial parcial de Poisson en el dominio . Este método aprovecha el poder de las redes neuronales para aproximar soluciones a ecuaciones diferenciales, incorporando las leyes físicas que gobiernan el problema directamente en el proceso de entrenamiento de la red.

La ecuación de Poisson, , donde *u* es la solución buscada y *f* es una función dada que actúa como fuente, es un problema clásico en matemáticas aplicadas, especialmente en campos como la física y la ingeniería. El objetivo es encontrar la función *u(x)*, que satisface esta ecuación en el dominio *Ω*, bajo condiciones iniciales y de frontera adecuadas.

En lugar de discretizar el dominio y resolver la ecuación de Poisson utilizando métodos numéricos tradicionales, como el método de diferencias finitas, se emplea una red neuronal profunda entrenada para aproximar la solución *u(x)* de manera directa. En este enfoque, la red neuronal no solo ajusta sus parámetros a partir de los datos etiquetados, sino que también es informada por las ecuaciones físicas que rigen el problema.

La red neuronal está configurada para minimizar una función de pérdida que combina dos términos principales en este trabajo:

1. El término de la ecuación de Poisson: La red aprende a aproximar la solución *u(x)* tal que cumpla con la ecuación en todo el dominio. Para ello, la red estima *u(x)* en puntos de malla distribuidos en *Ω*, y la derivada segunda de *u(x)* se calcula utilizando la diferenciación automática, luego se calcula el gradiente de la función de pérdida con respecto a los parámetros de la red neuronal.
2. El término de las condiciones de frontera: Las condiciones de frontera se imponen explícitamente en el entrenamiento de la red. Estas condiciones son cruciales para asegurar que la solución  tiene solución única y satisface las restricciones físicas en el límite del dominio. Para un dominio con frontera *∂Ω*, estas condiciones pueden ser Dirichlet (valores conocidos de *u* en la frontera) o Neumann (valores conocidos de la derivada normal de *u* en la frontera).

En este enfoque, se pueden incluir datos etiquetados en forma de puntos de entrenamiento donde la solución *u* es conocida (por ejemplo, en ciertos puntos del dominio o en la frontera). Estos datos etiquetados se incorporan en la función de pérdida para forzar a la red neuronal a aproximar correctamente los valores en esos puntos. Sin embargo, el mayor poder de las PINNs proviene de su capacidad para resolver las EDPs (como la ecuación de Poisson) a partir de la información inherente a la ecuación misma y las condiciones de frontera, sin necesidad de datos adicionales.

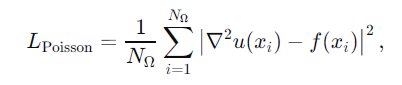
Las condiciones iniciales y de frontera se implementan de la siguiente manera:

* Condiciones de frontera de Dirichlet: Si la condición en la frontera es de Dirichlet en *∂Ω*, la red neuronal penaliza las discrepancias entre su estimación de *u(x)* en la frontera y los valores *g(x)* dados.
* Condiciones de frontera de Neumann: Si la condición en la frontera es de Neumann, es decir, en *∂Ω*, se penaliza la discrepancia entre el valor estimado de la derivada normal de *u(x)* y el valor *g(x)* en la frontera.

La función de pérdida final se combina en un solo término que incluye tanto la penalización por el error en la ecuación de Poisson como el error en las condiciones de frontera.

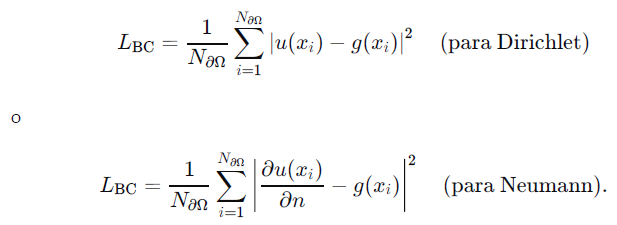
La función de pérdida *L* se compone de dos términos principales en este trabajo:

1. Pérdida de la ecuación de Poisson: Se calcula como el error entre el valor estimado de y la fuente *f(x)* en puntos de entrenamiento dentro del dominio *Ω*, es decir,



donde *NΩ*​ es el número de puntos de malla dentro del dominio *Ω*, y *xi*​ son los puntos en los que se evalúa la ecuación. La segunda derivada de *u(x)* se obtiene mediante diferenciación automática utilizando las herramientas de retropropagación de gradientes.

1. Pérdida de las condiciones de frontera: Se calcula como el error entre los valores de *u(x)* en la frontera y las condiciones de frontera especificadas *g(x)* (en el caso de Dirichlet) o el error en la derivada normal de *u(x)* (en el caso de Neumann), es decir,



Aquí, *N∂Ω* es el número de puntos en la frontera, y *xi*​ son los puntos en los que se aplican las condiciones de frontera.

La función de pérdida total se obtiene combinando ambos términos:

El proceso de entrenamiento de la red neuronal sigue el algoritmo estándar de retropropagación para optimizar la función de pérdida.

1. Cálculo de gradientes: Para cada iteración de entrenamiento, la red neuronal calcula el valor de la salida *u(x)* para un conjunto de puntos de entrenamiento en el dominio *Ω* y en la frontera *∂Ω*. Luego, se calcula el error entre la salida de la red y los valores deseados según la ecuación de Poisson y las condiciones de frontera.
2. Propagación hacia atrás: Usando el algoritmo de retropropagación, los gradientes de la función de pérdida respecto a los pesos de la red se calculan. Esto se logra mediante la aplicación de la regla de la cadena, que permite calcular cómo cada peso de la red contribuye al error final. Los gradientes se calculan tanto para el término relacionado con la ecuación de Poisson como para las condiciones de frontera.
3. Actualización de pesos: Los gradientes calculados se utilizan para actualizar los pesos de la red utilizando un algoritmo de optimización, como el optimizador Adam. El proceso de retropropagación continúa hasta que la función de pérdida se minimiza de manera efectiva, lo que resulta en una aproximación de la solución *u(x)* que satisface la ecuación de Poisson y las condiciones de frontera.

**3.3 Método mediante solución exacta**

En el caso de la búsqueda de la solución de la ecuación (1), se puede describir la solución exacta de la misma mediante la solución analítica de *u* dada por *u(x,y) = −[1/(2.)].sin(𝜋.𝑥).sin(𝜋.𝑦)*. Para corroborar que es solución de la ecuación (5), operamos derivando.

En primer lugar, se trabaja con las derivadas primera y segunda de *u* con respecto a *x*:

De manera similar, se trabaja con las derivadas primera y segunda de *u* con respecto a *y*:

Realizando la suma, se tiene que:

Que es igual a la expresión dada por la ecuación (5) de .

Al igual que métodos previos, se calculó el valor de *u(x,y)* con la solución exacta para cada punto de grilla, utilizando las resoluciones espaciales de 5x5, 10x10 y 20x20.

**4. RESULTADOS**

4.1 Soluciones de la ecuación sinusoidal

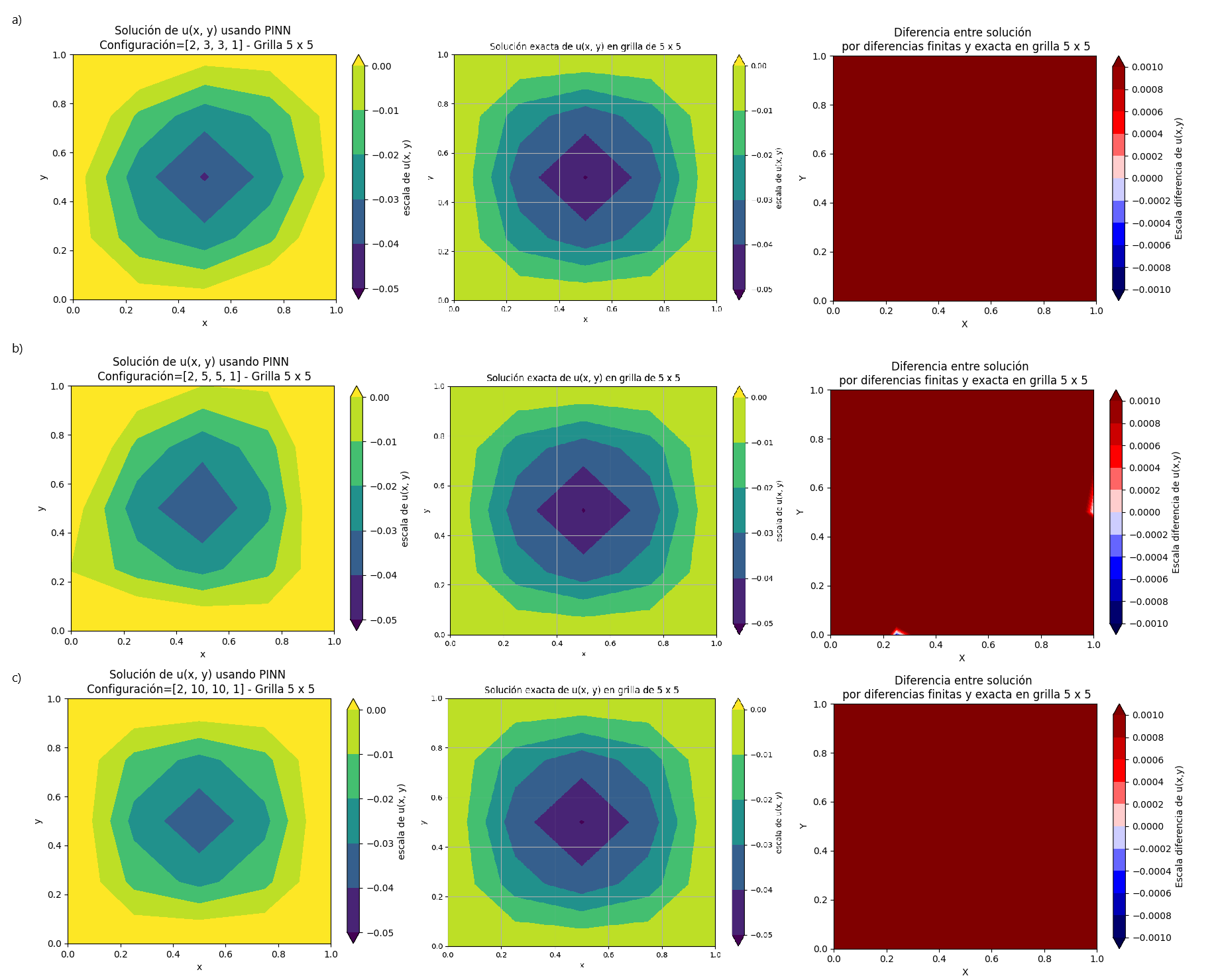
4.1.1 Utilizando una grilla de 5x5

Luego de realizar los cálculos por los métodos propuestos indicados en la sección 3, se obtuvieron los siguientes campos escalares de *u*.

***Fig.1****: Campos de de solución de u en grilla de 5x5 para: a) utilizando diferencias finitas, b) utilizando la solución exacta propuesta, c) diferencia entre soluciones.*

Se observó que el método de diferencias finitas arrojó buenos resultados, con la misma forma del campo escalar y orden de magnitud dado por la solución exacta. Además, en todo el dominio los valores que toma *u* son negativos con un mínimo absoluto en el centro de la región. No obstante, a medida que se desplaza al centro del dominio, se identificó valores negativos de diferencia de soluciones. Esto mostró que hacia esa zona, la solución dada por diferencias finitas tiende a sobrestimar, en intensidad, el valor de *u*.

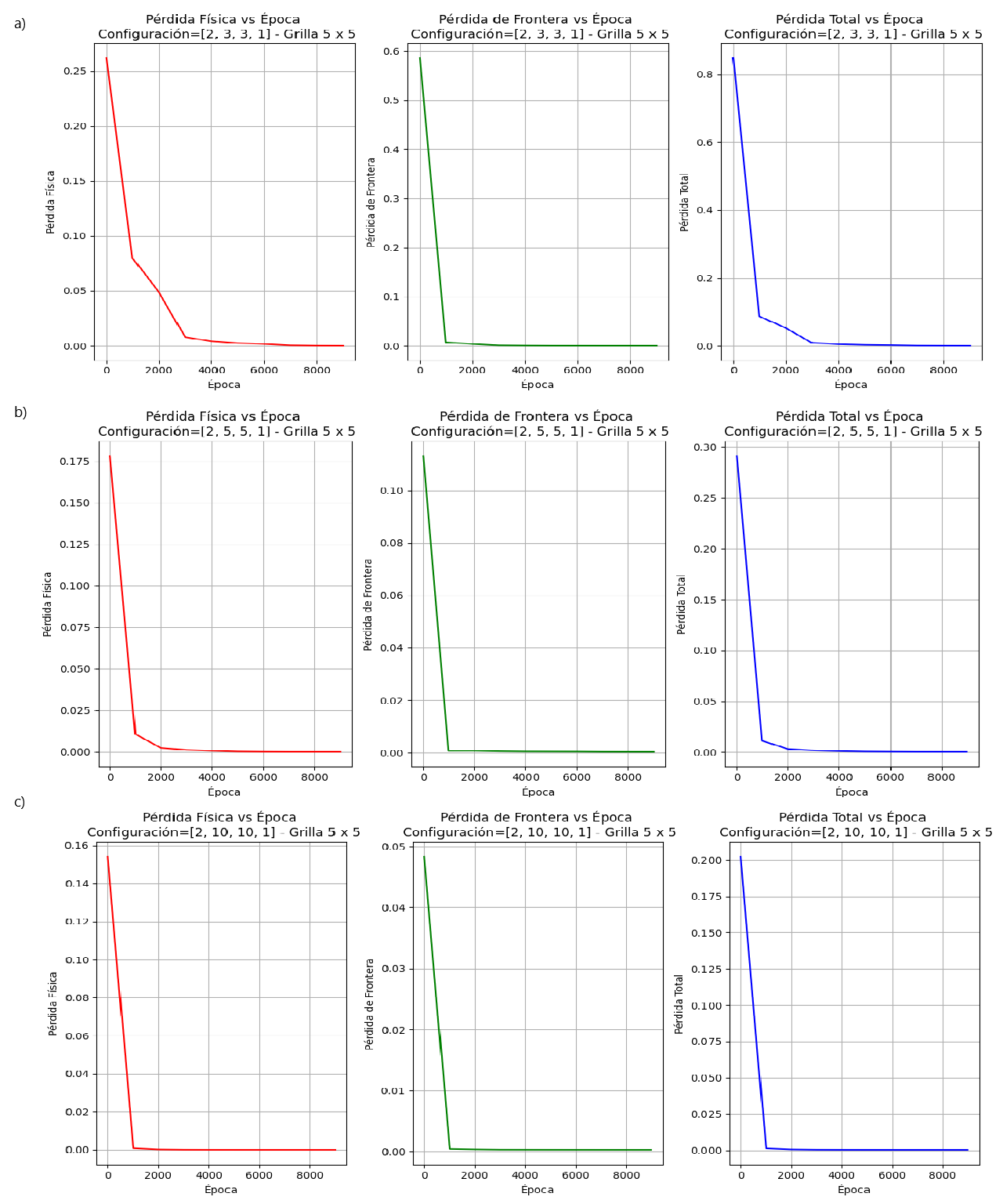
Por otro lado, se buscó la solución utilizando PINN. Para esto se utilizó tres diferentes configuraciones de las capas ocultas, trabajando con 3, 5 y 10 neuronas por cada capa oculta en cada caso. En la figura 2 se muestran los resultados de la red con estas diferentes configuraciones, y sus diferencias con la solución exacta.



***Fig.2****: Campos de solución de u(x,y) por PINN, de solución exacta y de diferencia entre estos en grilla de 5x5 para configuración PINN de: a) [2, 3, 3, 1], b) [2, 5, 5, 1] y c) [2, 10, 10, 1].*

Se observó que la solución dada por PINN tiene la misma estructura que la solución exacta, con el mínimo en el centro y valores que tienden a cero en el contorno del dominio como indica la condición de borde. No obstante, a partir de la diferencia entre la solución por PINN y la solución exacta, se identificó valores positivos en todo el dominio de forma homogénea. Esto puso en evidencia que la solución por PINN tiende a subestimar (en intensidad) los valores de u. En cuanto a las diferencias entre el uso de diferentes configuraciones de la red, se vio que las diferencias entre la solución por PINN y exacta, no se modifican al variar la cantidad de neuronas de las capas ocultas.

Por otro lado, se generaron las curvas de pérdida para cada componente y configuración de la red.

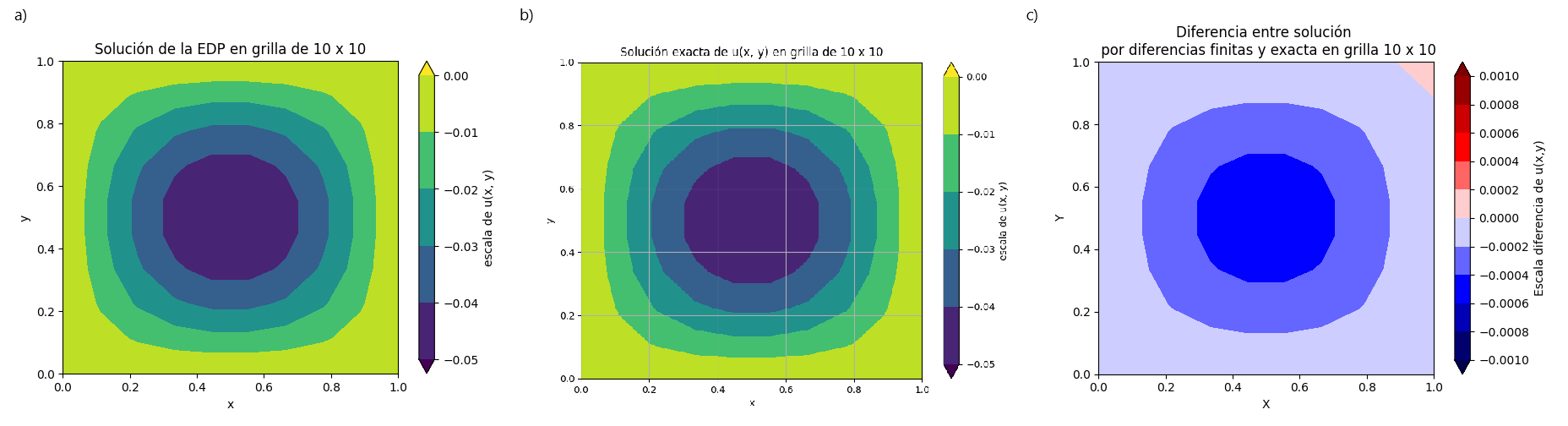


***Fig 3.****: Curvas de pérdida total, ecuación de gobierno y de condiciones de borde en grilla de 5x5 para diferentes con****figuraciones de*** *PINN: a) [2, 3, 3, 1], b) [2, 5, 5, 1] y a) [2, 10, 10, 1]*

En todos los casos se observó que entre 1000 y 3000 épocas la pérdida converge a el mínimo. En el caso de 3 neuronas por capa oculta, se observó que el mínimo por ecuación de gobierno demanda más épocas que para las condiciones de borde. Mientras que esta demora en la convergencia disminuye al aumentar la cantidad de neuronas en capas ocultas.

4.1.2 Utilizando una grilla de 10x10

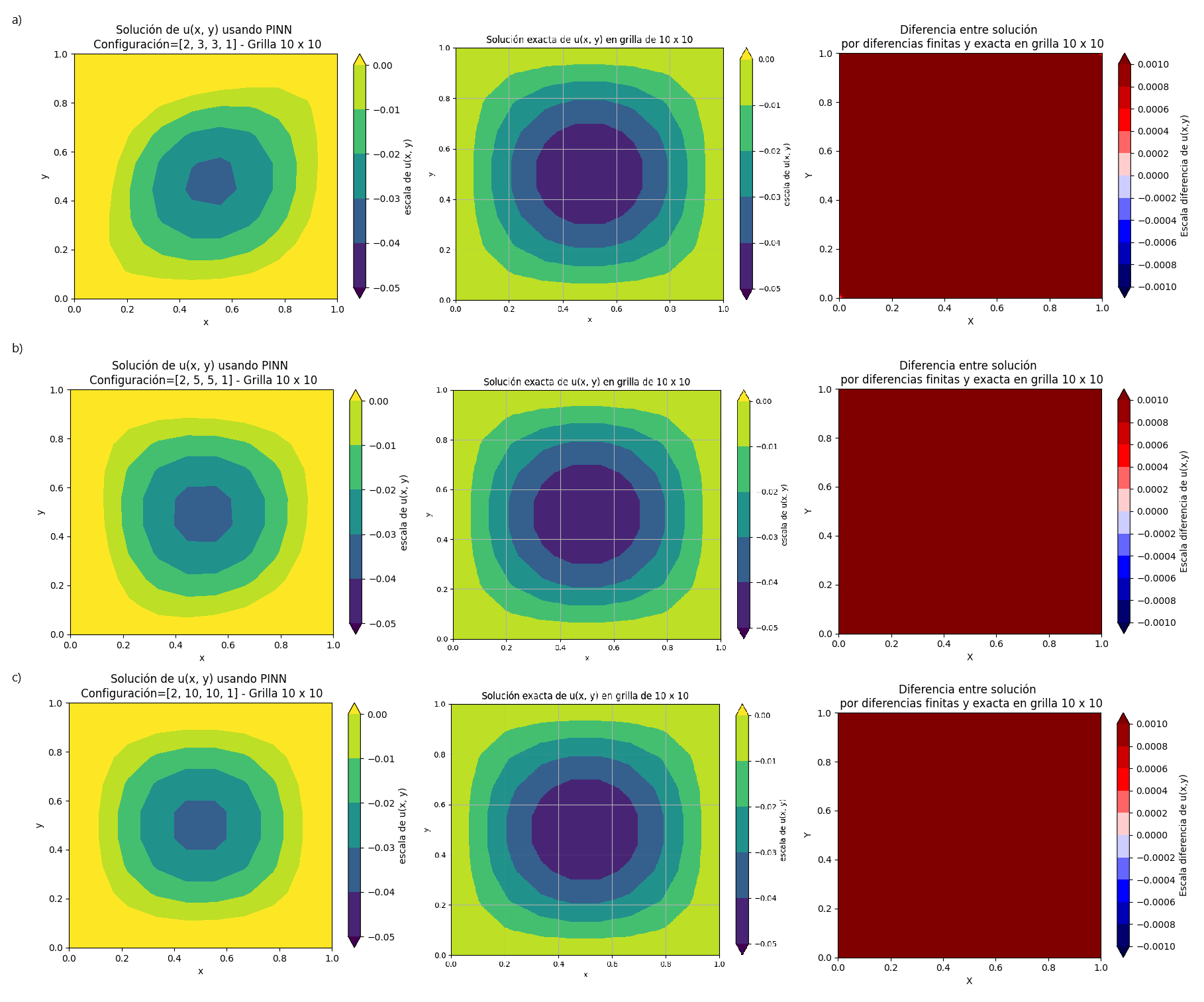
De manera similar, luego de realizar los cálculos de la solución por diferencias finitas, de la solución exacta y su diferencia, se generaron los campos respectivos.



***Fig. 4****: Campos de de solución de u en grilla de 10x10 para: a) utilizando diferencias finitas, b) utilizando la solución exacta propuesta, c) diferencia entre soluciones.*

Al igual que en el caso de 5x5, los campos mostraron la misma morfología. No obstante, la forma del mínimo absoluto se ajustó mejor a la forma suave sinusoidal tanto en los campos de la solución como en el de la diferencia. En cuanto a las diferencias, se vio que la intensidad de las mismas disminuyó. Esto puso en evidencia que al aumentar la resolución espacial el método por diferencias finitas se ajusta mejor a la solución exacta.

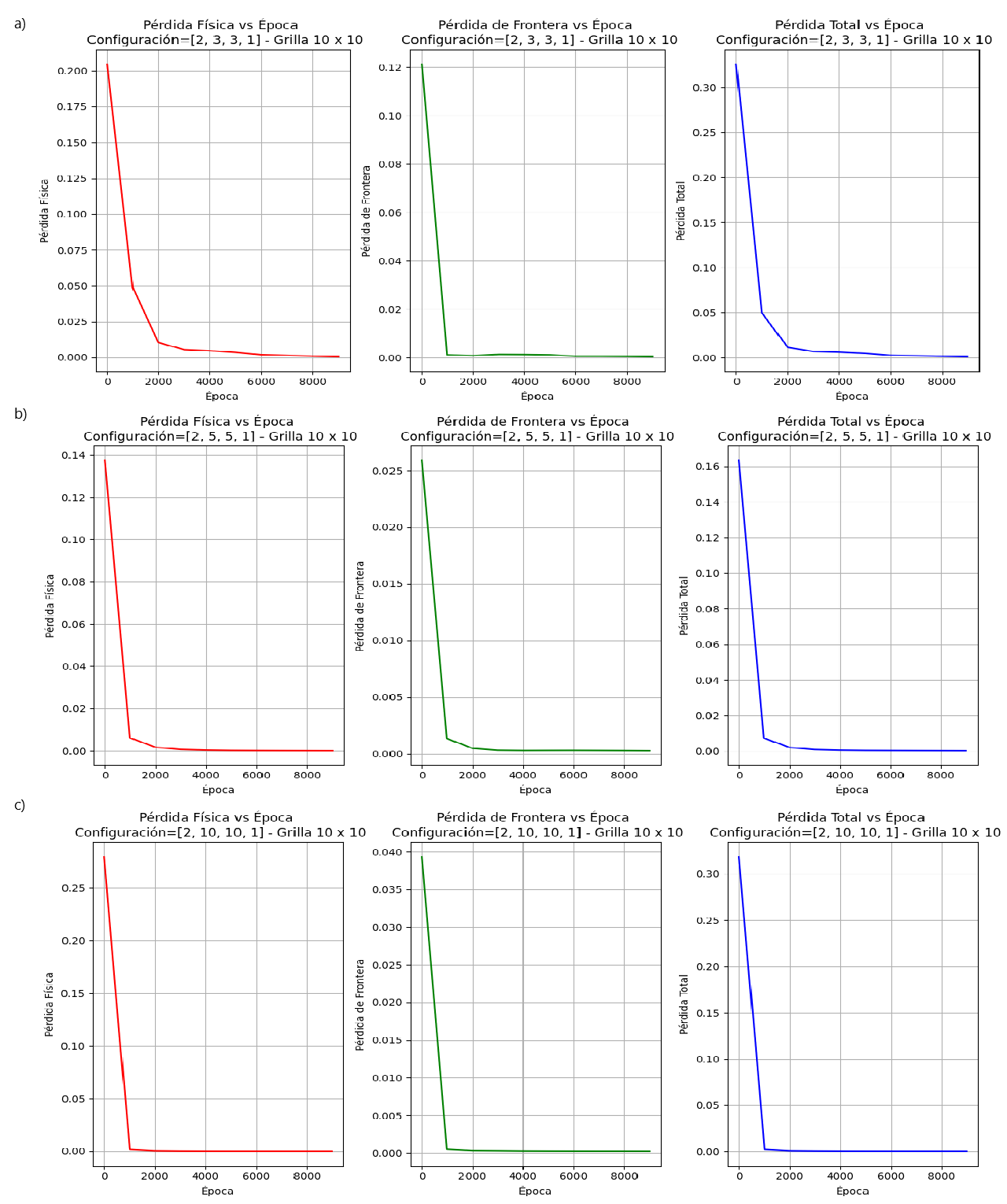
Luego, al trabajar con la red PINN en las configuraciones de capas ocultas propuestas, se llegó a los resultados de la figura 5.



***Fig. 5****: Campos de solución de u(x,y) por PINN, de solución exacta y de diferencia entre estos en grilla de 10x10 para configuración PINN de: a) [2, 3, 3, 1], b) [2, 5, 5, 1] y c) [2, 10, 10, 1]*

Al igual que el caso anterior, al usar una grilla de 10x10, la solución por PINN presentó la misma estructura que la solución exacta. También se mantuvieron homogéneas las diferencias entre las soluciones con las diferentes configuraciones, y sin grandes variaciones entre estas.

Nuevamente, se generaron las curvas de pérdida vs las épocas para cada componente de pérdida y para cada configuración de capas.

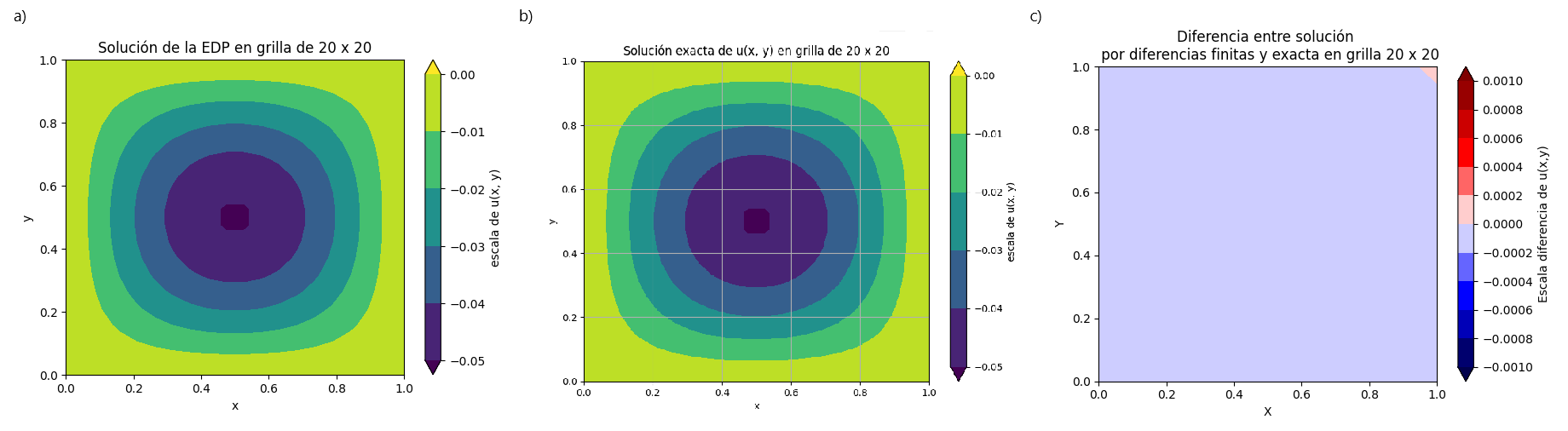


***Fig 6.****: Curvas de pérdida total, ecuación de gobierno y de condiciones de borde en grilla de 10x10 para diferentes con****figuraciones de*** *PINN: a) [2, 3, 3, 1], b) [2, 5, 5, 1] y a) [2, 10, 10, 1]*

Al igual del caso con grilla de 5x5, al aumentar la cantidad de neuronas en capas ocultas, disminuye la cantidad de épocas necesarias para alcanzar el mínimo de pérdida en la componente de ecuación de gobierno. En cuanto a la pérdida de condiciones de borde, no se apreciaron grandes diferencias al modificar la configuración.

4.1.3 Utilizando una grilla de 20x20

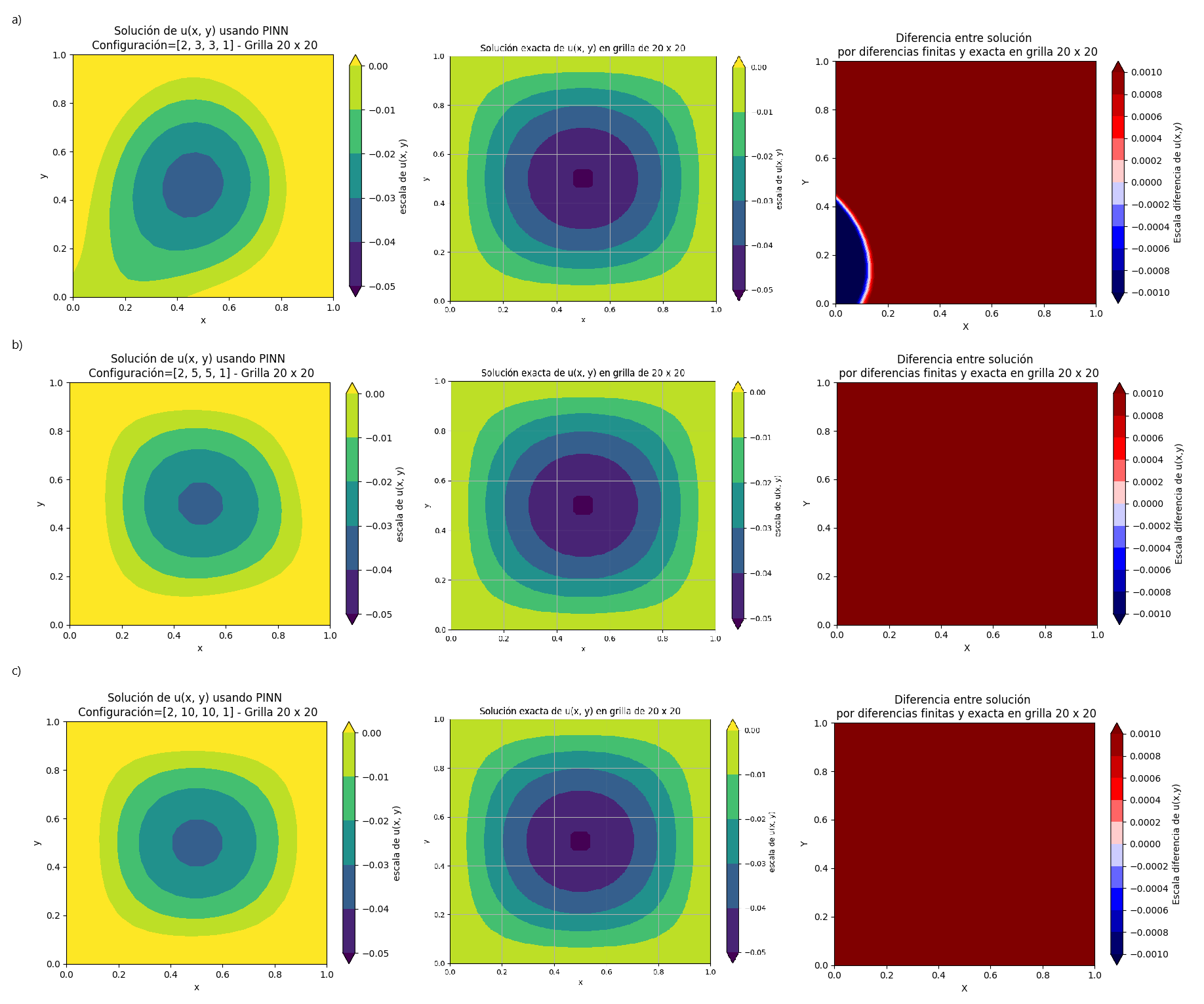
Se repitió el experimento de búsqueda de la solución mediante diferencias finitas y por solución exacta analítica de *u*, en la Figura 7 se muestran los resultados.



***Fig. 7****: Campos de de solución de u en grilla de 20x20 para: a) utilizando diferencias finitas, b) utilizando la solución exacta propuesta, c) diferencia entre soluciones.*

Al aumentar la resolución espacial (aumentar la cantidad de puntos de grilla del dominio) se logró representar mejor la solución *u(x,y)* al punto tal que se pudo representar de mejor forma la suavidad característica de la sinusoide. Nuevamente, al aumentar la resolución espacial de la grilla se logró disminuir la sobrestimación (en intensidad) de u por diferencias finitas que se vió en casos previos.

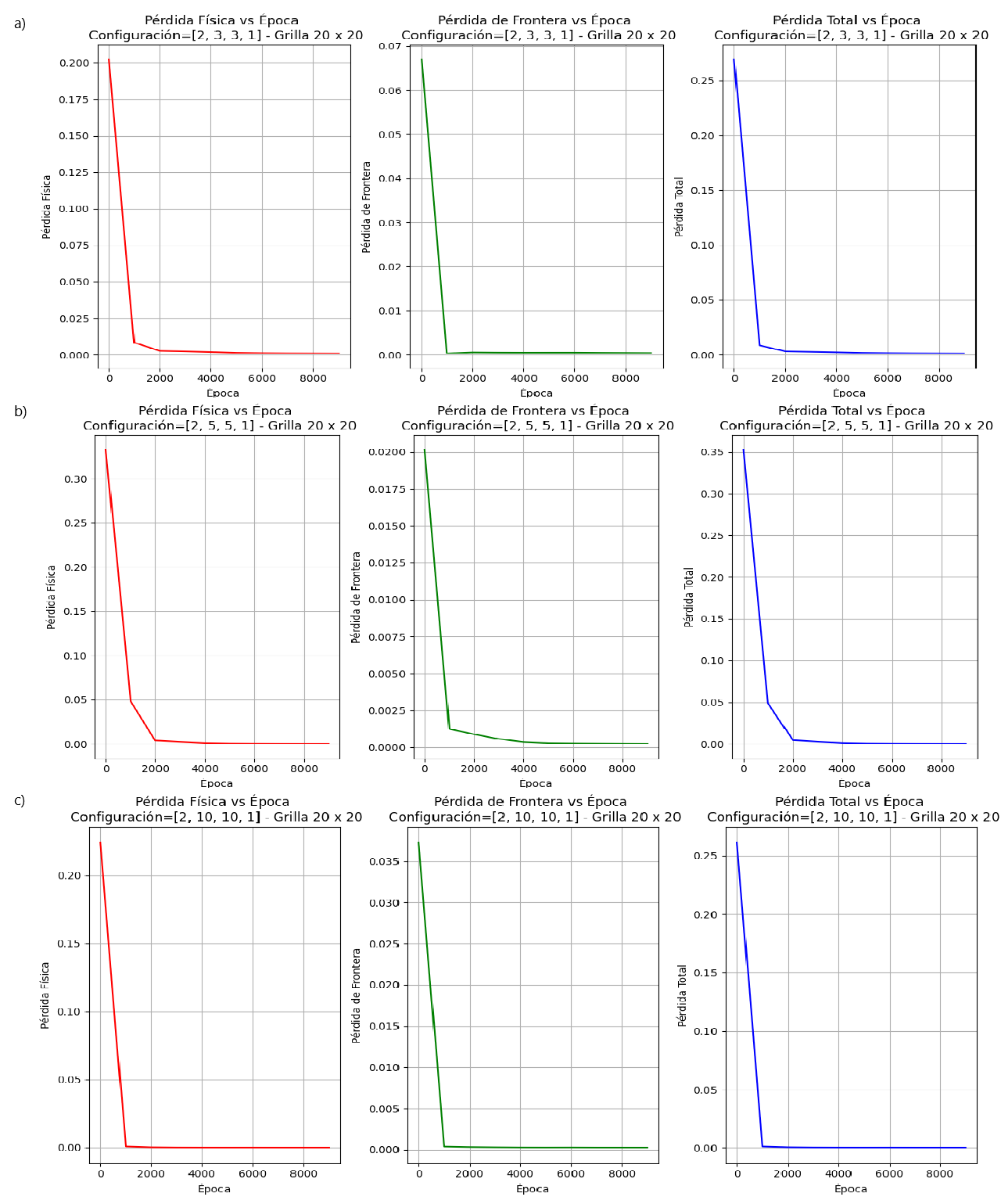
También se generaron los campos de solución con PINN en las tres configuraciones propuestas, y se las comparó con la solución exacta.



***Fig. 8****: Campos de solución de u(x,y) por PINN, de solución exacta y de diferencia entre estos en grilla de 20x20 para configuración PINN de: a) [2, 3, 3, 1], b) [2, 5, 5, 1] y c) [2, 10, 10, 1]*

Se observó un alargamiento en la configuración con 3 neuronas por capa oculta. Es posible que el modelo no haya logrado aprender las condiciones de borde con esta configuración, cuestión que corrigió al aumentar la cantidad de neuronas. Nuevamente, la subestimación en la intensidad de la solución se mantuvo homogénea para los casos con más conexiones.

Por último, las funciones de pérdida por componente y configuración mostraron los siguientes resultados.



***Fig. 9.****: Curvas de pérdida total, ecuación de gobierno y de condiciones de borde en grilla de 20x20 para diferentes configuraciones de PINN: a) [2, 3, 3, 1], b) [2, 5, 5, 1] y a) [2, 10, 10, 1].*

En este caso se apreció un aumento en la cantidad de épocas necesarias para alcanzar el mínimo de pérdida en la componente de ecuación de gobierno en la configuración de cinco neuronas por capa oculta.

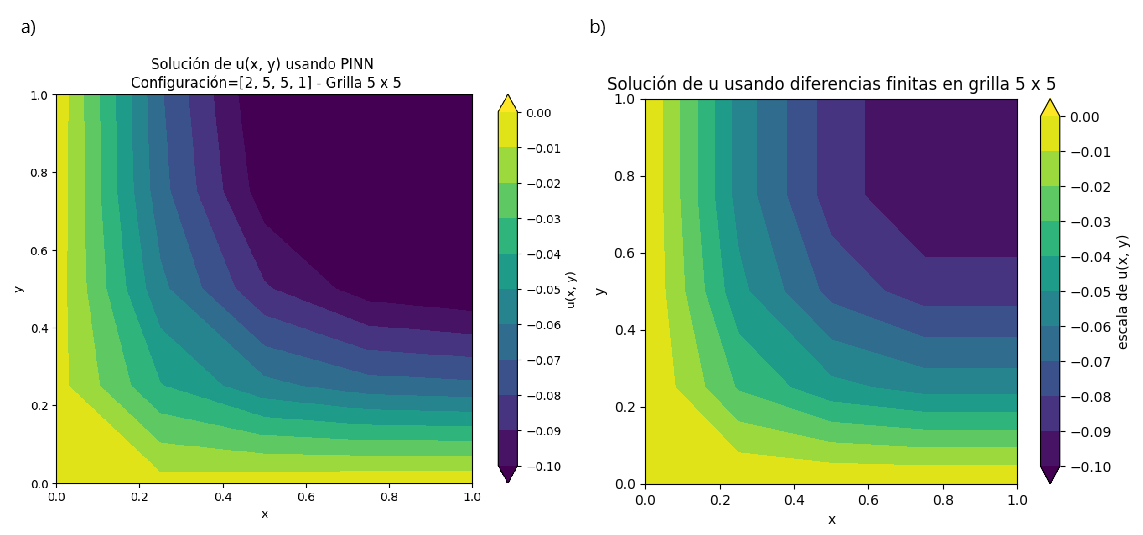
4.1.4 Utilizando arquiteturas alternativas.

A completar…

4.2 Soluciones de la ecuación de conducción de calor con término fuente no lineal

4.2.1 Utilizando una grilla de 5x5

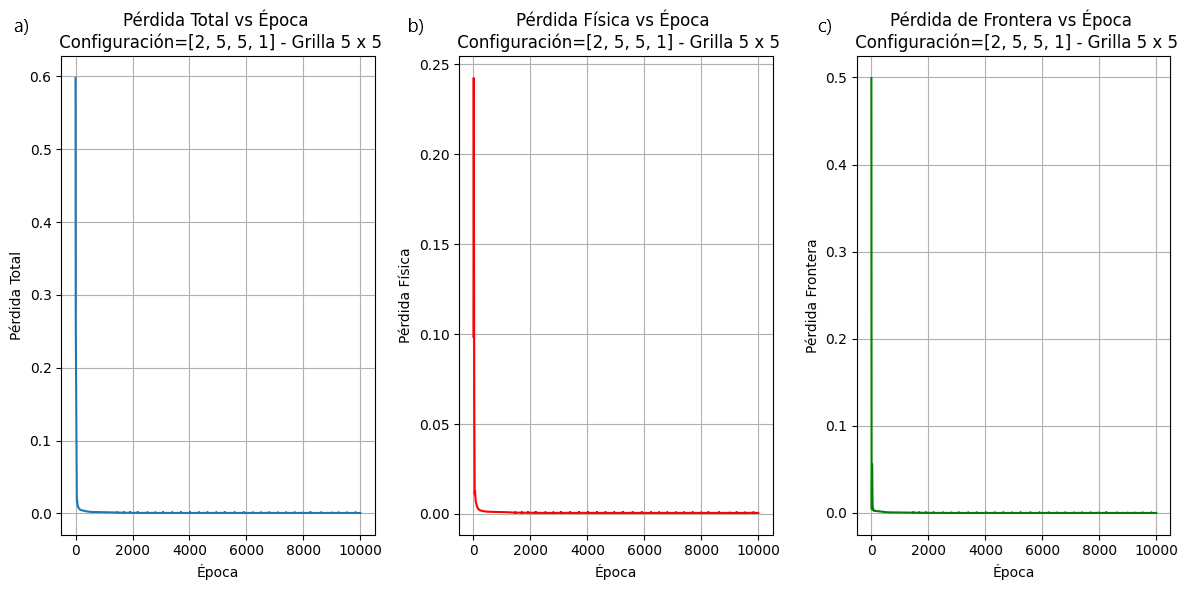
Utilizando la metodología definida en la sección 3, se procedió a generar la solución mediante PINN con la configuración con cinco neuronas por capa oculta y por diferencias finitas en grilla de 5 x5 en ambos casos.



***Fig. 10****: Campos de solución de u(x,y) en una grilla de 5x5 con: a) PINN, y b) diferencias finitas.*

Ante todo se pudo observar la misma estructura de la solución *u* con ambos métodos. Esto es, un mínimo absoluto en *x = y* = 1. En ambos casos se respetan las condiciones de borde. A partir de la configuración de las paletas, la magnitud de la solución en las gráficas resulta comparable. Con esto, se vio que en el caso de PINN, la extensión espacial del mínimo es mayor, resolviendo con una intensidad de gradiente mayor en el resto del dominio.

También se generaron las curvas de aprendizaje para cada componente.

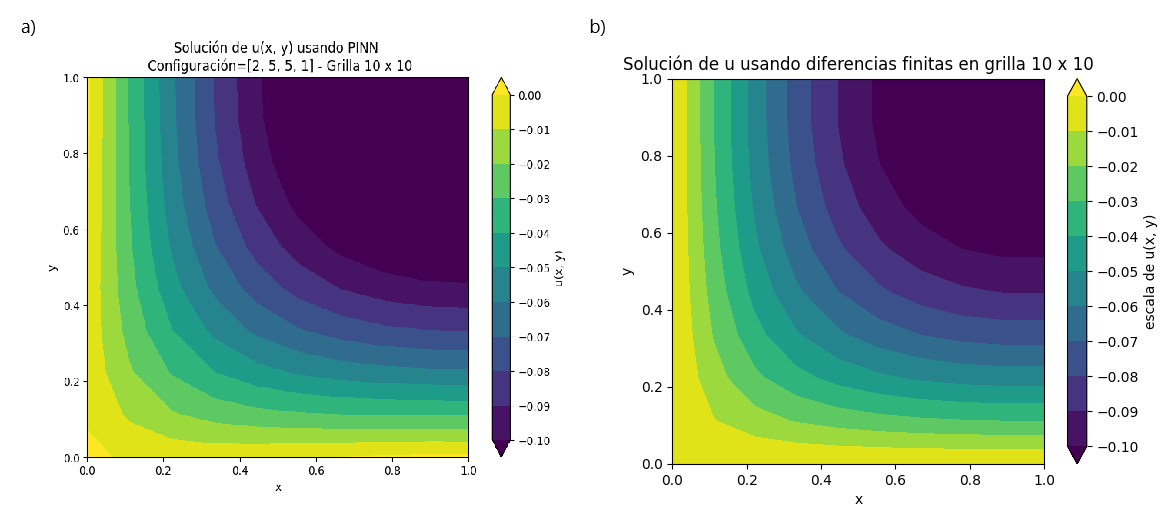


***Fig. 11.****: Curvas de pérdida total (a), ecuación de gobierno (b) y de condiciones de borde (c) en grilla de 5x5 para la configuración**de**PINN [2, 5, 5, 1].*

Se observó que en ambos casos, y por ende en la pérdida total, el mínimo se alcanza en las primeras épocas.

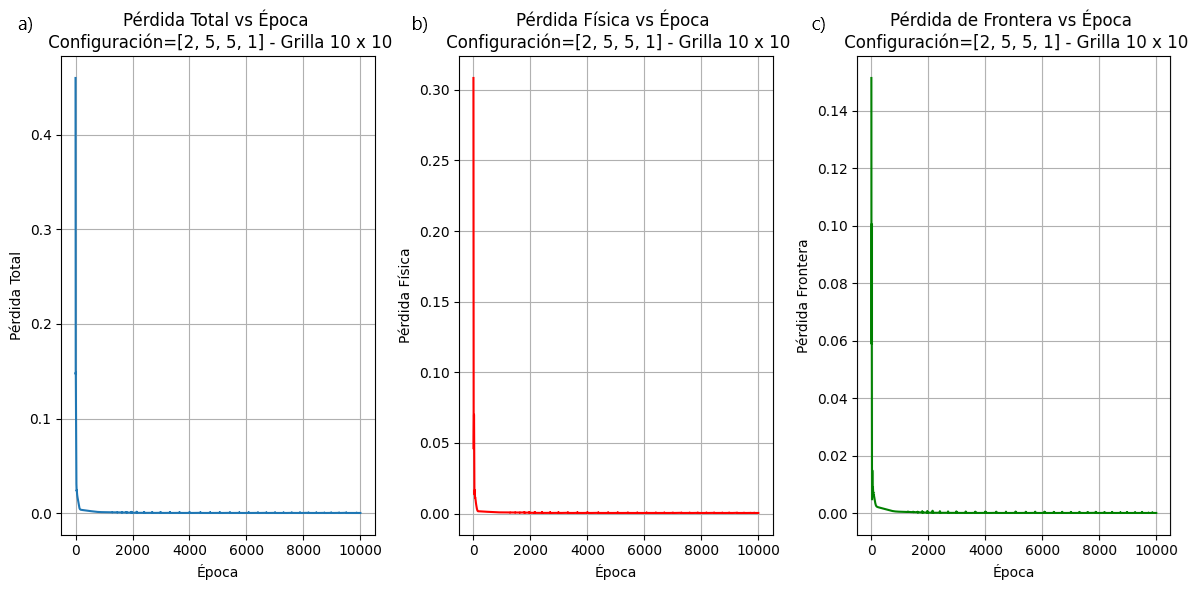
4.2.2 Utilizando una grilla de 10x10

Analizando los resultados de la solución para el caso con PINN y diferencias finitas, se observó nuevamente que la estructura se respeta en ambos casos.



***Fig. 12****: Campos de solución de u(x,y) en una grilla de 10x10 con: a) PINN, y b) diferencias finitas.*

Por otro lado, se pudo ver que la extensión del mínimo es similar en ambos casos (si bien continúa siendo mayor en el caso de PINN), al menos en comparación con lo visto en el caso de grilla de 5x5.

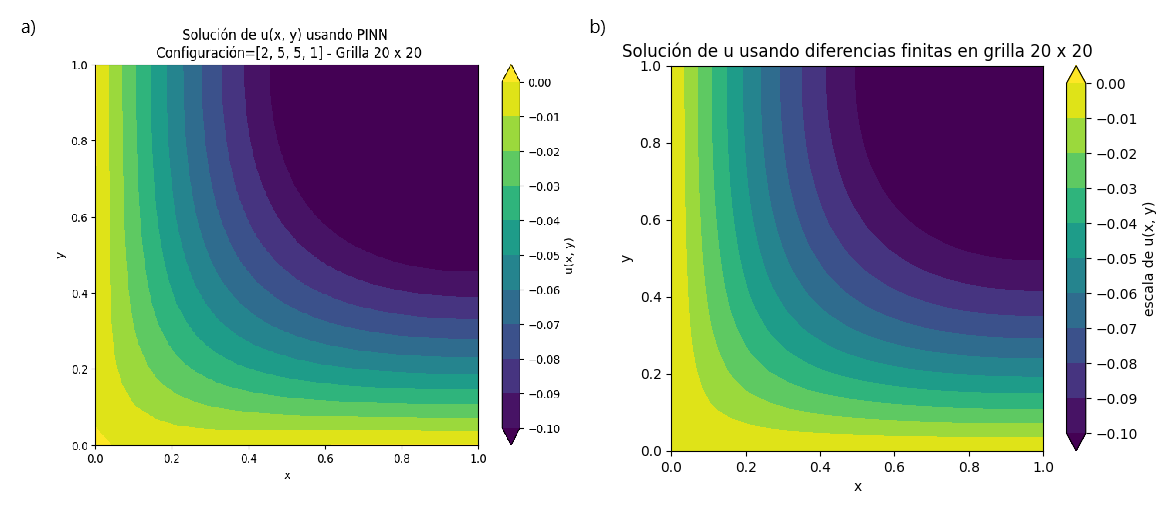


***Fig. 13.****: Curvas de pérdida total (a), ecuación de gobierno (b) y de condiciones de borde (c) en grilla de 10x10 para la configuración**de**PINN [2, 5, 5, 1].*

El comportamiento de las curvas de aprendizaje no mostró variaciones con respecto al caso de grilla de 5x5. Nuevamente se apreció que el mínimo se alcanzó en pocas épocas en ambas componentes (ecuación de gobierno y condiciones de borde).

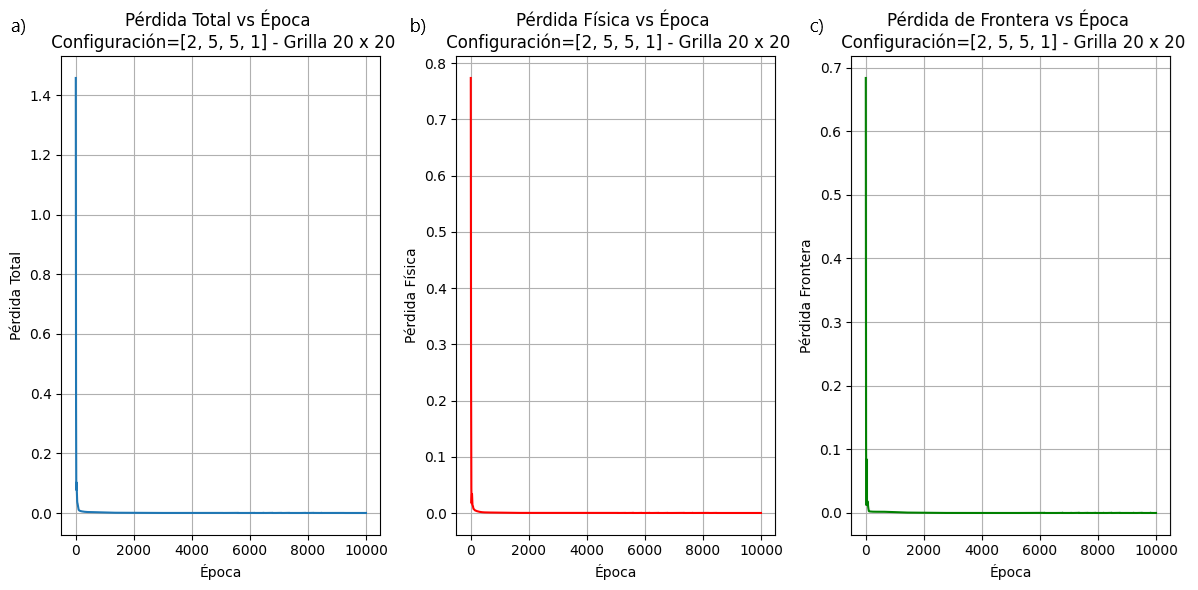
4.2.3 Utilizando una grilla de 20x20

Una vez más las soluciones por ambos métodos presentaron la misma estructura, si bien se conservó que el mínimo por PINN fuese un poco más extenso que el caso por diferencias finitas.



***Fig. 14****: Campos de solución de u(x,y) en una grilla de 20x20 con: a) PINN, y b) diferencias finitas.*

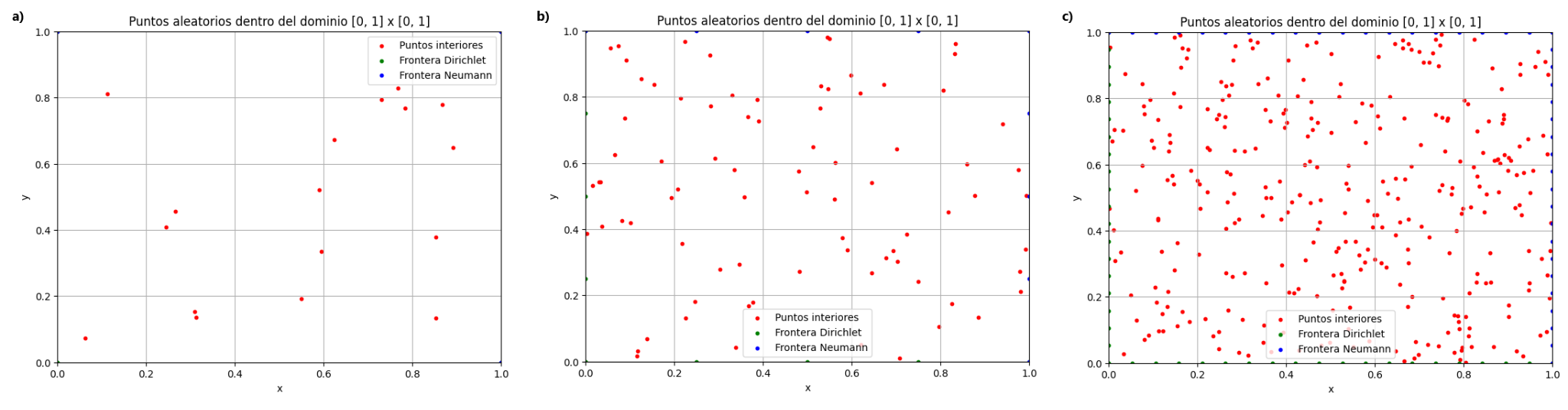
Sin mayores cambios, las curvas de pérdida mostraron el mismo comportamiento que en casos previos, alcanzando el mínimo en pocas épocas.



***Fig. 15.****: Curvas de pérdida total (a), ecuación de gobierno (b) y de condiciones de borde (c) en grilla de 20x20 para la configuración**de**PINN [2, 5, 5, 1].*

4.2.4 Utilizando una muestra aleatoria de puntos de colocación

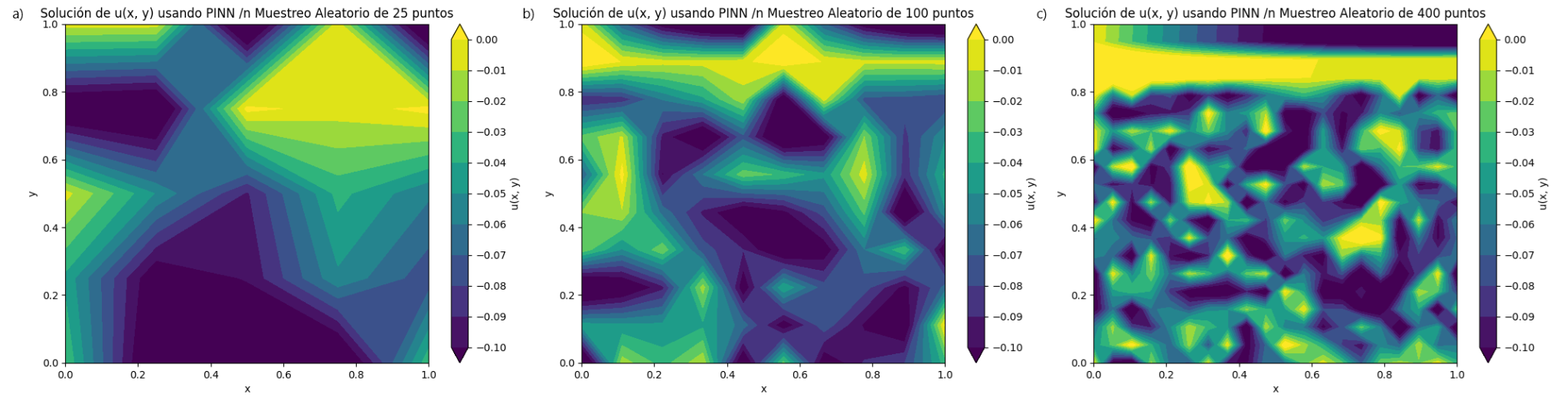
Para este experimento se utilizó nuevamente una configuración [2, 5, 5, 1] ubicando la misma cantidad de puntos de colocación que las grillas de 5x5, 10x10 y 20x20 pero de manera uniforme y aleatoria. Destinando un 20% de los datos a la frontera en los casos de 400 y puntos de colocación, y un 32% en el caso de 25 puntos. En la figura 16 se muestran la distribución de puntos en cada caso.



***Fi. 16****: Distribución de puntos de colocación para: a) 25 puntos, b) 100 puntos y c) 400 puntos. En rojo se marcan los puntos internos del dominio, mientras que en verde y azul los puntos de frontera.*

En el caso de 25 puntos aleatorios, se destinó 8 puntos (32%) a la frontera, con dos puntos por borde. Luego, en el caso de 100 y 400 puntos aleatorios, se destinó el 20% de los mismos a la frontera (20 y 80 respectivamente). En el primer caso (Figura 16a), fue evidente la gran cantidad de espacios sin representación, lo que en principio no pareció ser una buena resolución para hallar la solución. En el segundo caso (Figura 16b), si bien hubo una mejora en la resolución espacial, aún se pudieron identificar regiones con ausencia de puntos. Sin ir más lejos, el último caso (Figura 16c), fue la que mejor distribución de puntos presentó, tanto en el interior del dominio como en la frontera.

Definidos estos dominios, se calculó la solución mediante PINN con la configuración [2, 5, 5, 1], los resultados se aprecian en la Figura 17.



***Fig. 17****: Soluciones de campo u mediante PINN para los casos: a) 25 puntos, b) 100 puntos, y c) 400 puntos.*

Utilizando un esquema *vanilla* los resultados no son buenos para ninguna de las tres cantidades de puntos de colocación propuestos. En ningún caso se llega a identificar la morfología exponencial de la ecuación diferencial. Lo único que se pudo observar que responde a lo esperable, es en el borde superior en el caso de 400 puntos de colocación. Allí, se apreciaron valores mínimos de *u* que responden a la condición de frontera impuesta.

**5. CONCLUSIONES**

Relativo a la ecuación sinusoidal

El método de diferencias finitas proporciona buenos resultados, logrando una morfología similar a la de la solución exacta, con un mínimo en el centro del dominio. Sin embargo, se observó que la solución obtenida mediante diferencias finitas tiende a sobrestimar la intensidad en comparación con la solución exacta. Además, esta sobrestimación es más pronunciada hacia el centro del dominio. Por otro lado, dicha sobrestimación tiende a disminuir y a homogeneizarse espacialmente a medida que aumenta el tamaño de la grilla.

Las soluciones mediante PINN también presentaron una morfología similar a la de la solución exacta. Sin embargo, a diferencia de lo observado en el método de diferencias finitas, la solución obtenida con PINN tiende a subestimar la intensidad respecto a la solución exacta. Esta subestimación parece no variar con la resolución espacial, mostrando un campo de diferencias homogéneo.

Es importante recordar que los resultados obtenidos mediante PINN incluyen una componente aleatoria, dada el uso de optimizadores estocásticos en su arquitectura.

A completar…

Relativo a la ecuación exponencial

Independientemente del tamaño de grilla empleado, las soluciones obtenidas mediante diferencias finitas y PINN presentaron una morfología similar, con un mínimo absoluto en la esquina superior derecha del dominio (x = y = 1). En todos los casos, las condiciones de borde de Dirichlet y Neumann se ajustaron adecuadamente. Sin embargo, en las soluciones mediante PINN, la extensión del mínimo es mayor; esta diferencia, no obstante, tiende a disminuir al incrementar el tamaño de la grilla.

En cuanto al uso de puntos aleatorios: A completar…

**6. REFERENCIAS**

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[2] Aniruddha Bora and Weizhong Dai. *Gradient preserved method for solving heat conduction equation with variable coefficients in double layers*. *Applied Mathematics and Computation*, 386:125516, 2020.

[3] MWM Gamini Dissanayake and Nhan Phan-Thien. *Neural-network-based approximations for solving partial differential equations*. *Communications in Numerical Methods in Engineering*, 10(3):195–201, 1994.

A completar…