Resolución de Ecuaciones Diferenciales mediante redes PINN.

Redes PINNs y QPINNs en el Aprendizaje de Funciones

Antonio López Montes.

Dto. Análisis Matemático y Matemática Aplicada. UCM

Resumen:

El potencial de las redes neuronales se refuerza cuando se integran con conocimientos explícitos del dominio físico.

Las *Physics-Informed Neural Networks* (PINNs), propuestas por Raissi, Perdikaris y Karniadakis en 2019, permiten resolver ecuaciones diferenciales, tanto ordinarias como en derivadas parciales, al incorporar las leyes físicas del sistema directamente en la función de pérdida.

En esta charla, exploraremos cómo las redes PINNs permiten representar funciones con estructura física en general y soluciones de ecuaciones diferenciales en particular, más allá del ajuste estadístico convencional.

Además, abordaremos extensiones recientes del modelo, como su extensión al ámbito de la computación cuántica. Veremos el uso de funciones de activación cuánticas y arquitecturas híbridas con circuitos cuánticos, que amplían la eficiencia y estabilidad de este tipo de redes.

UNIVERSIDAD COMPLUTENSE MADRID

Facultad de Informática

ANUNCIO DE CONFERENCIA

POSGRADO

Resolución de Ecuaciones Diferenciales mediante redes PINN.

Antonio López Montes
Facultad de Ciencias Matemáticas UCM

Facultad de Informática
Sala de Grados - jueves 26 de junio de 2025 - 16:30
Entrada libre hasta completar el aforo

Resumen:

El potencial de las redes neuronales se amplía cuando se integran con conocimientos explícitos del dominio físico.Las Physics-Informed Neural Networks (PINNs), propuestas por Raissi, Perdikaris y Karniadakis en 2017, permiten resolver ecuaciones diferenciales, tanto ordinarias como en derivadas parciales, al incorporar las leyes físicas del sistema directamente en la función de pérdida. En este enfoque, exploramos además la posibilidad de utilizar funciones de activación cuánticas, que amplían el espacio de representaciones no lineales mediante principios de la computación cuántica, lo que podría enriquecer la capacidad de aproximación de las PINNs. Así, las redes no solo ajustan datos observacionales, sino que también garantizan consistencia con los modelos físico-matemáticos subyacentes, ofreciendo soluciones tanto teóricas como numéricas a problemas complejos.

Sobre Antonio López Montes:

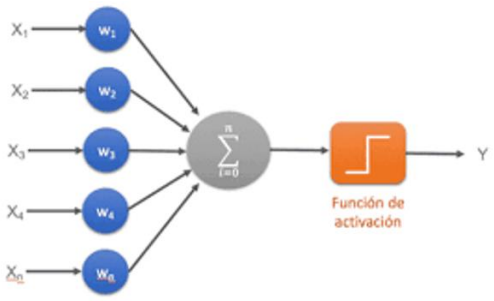
Licenciado en Ciencias Físicas. Especialidad en Física Teórica. Doctor en Matemáticas. Tema de la Tesis: Teoría de Máquinas. Profesor del departamento de Análisis Matemático y Matemática Aplicada UCM Autor del Libro: Computación Cuántica. Marzo 2024.

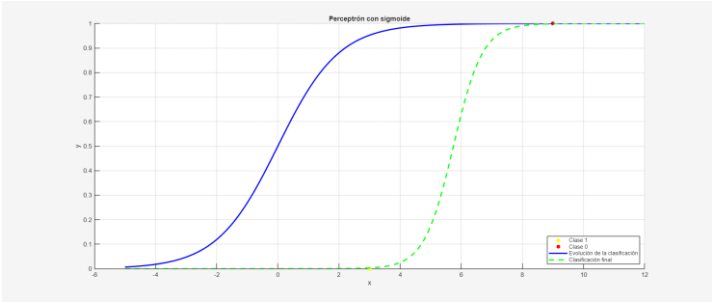
ÍNDICE

1. De redes neuronales a redes informadas por física

1.1 Primeros intentos de aprendizaje de funciones con redes clásicas

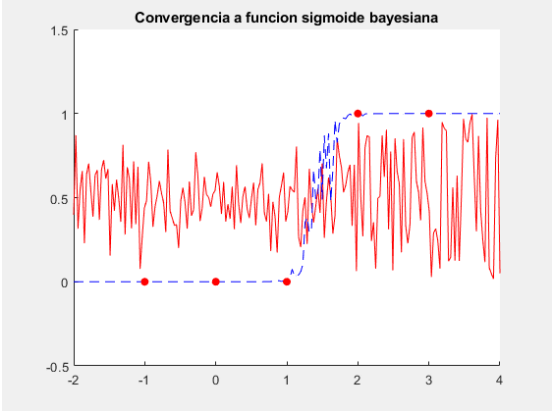
1950 – Podemos entender una red como una función que se ajusta al comportamiento deseado en el proceso de entrenamiento





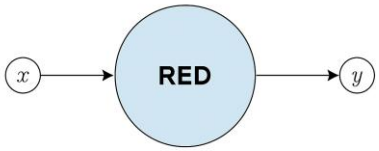
Las redes son capaces de aprender/imitar comportamientos sofisticados utilizando la composición repetida de estructuras, las funciones de activación.

Las redes pueden entrenarse con otros criterios, por ejemplo, podemos entrenar redes bayesianas



ELEMENTOS CLAVES EN LAS REDES NEURONALES	
Desde el punto de vista del diseño: Funciones de activación Existencias de capas profundas Funciones de pérdida $\mathcal{L}$	**Desde el punto de vista matemático:** No linealidad intrínseca Uso de la composición de funciones No hay buenas métricas en los procesos de convergencia

¿qué arquitectura de red neuronal se va a utilizar para aprender funciones?

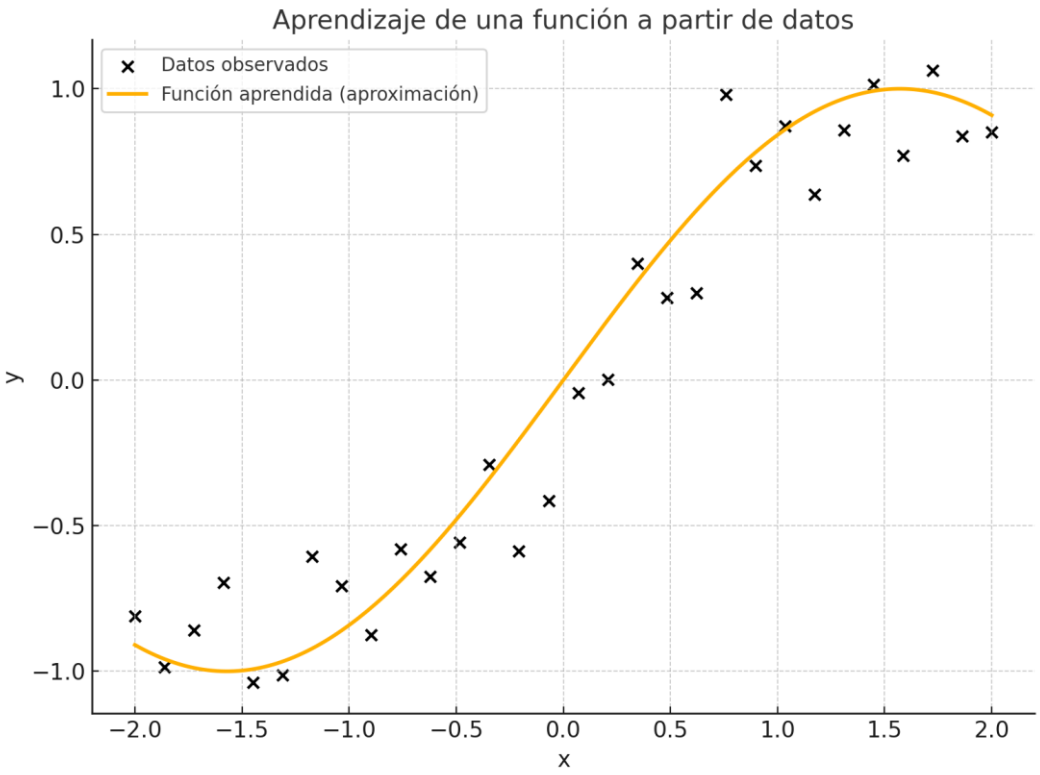


1.2 ¿Qué significa aprender una función?

En el contexto de la IA, aprender una función significa

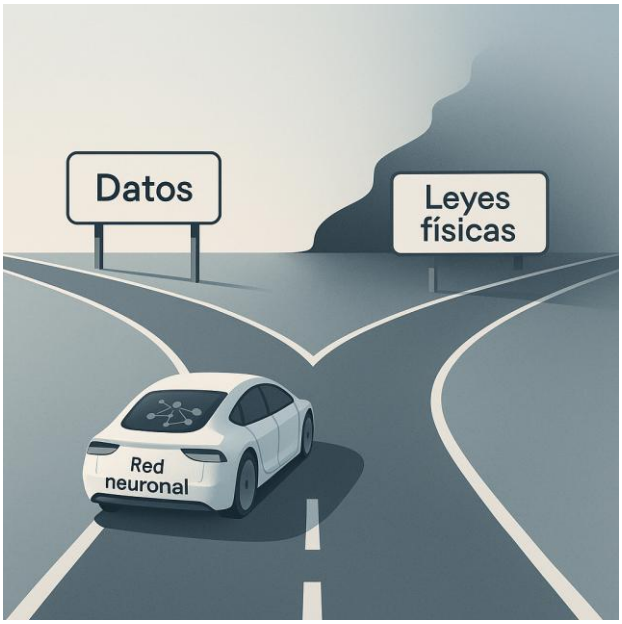
encontrar una representación computacional que modele la relación entre entradas (x) y salidas (y) de acuerdo con "cierta información" suministrada a través de unos datos

Utilizando procesos matemáticos (regresiones) o redes neuronales podemos aprender funciones a partir de nubes de puntos:



1.3 Limitaciones de los modelos tradicionales.

"Ni los modelos matemáticos de regresión ni las redes neuronales guiadas por datos pueden incorporar información adicional al ajustar nubes de puntos."



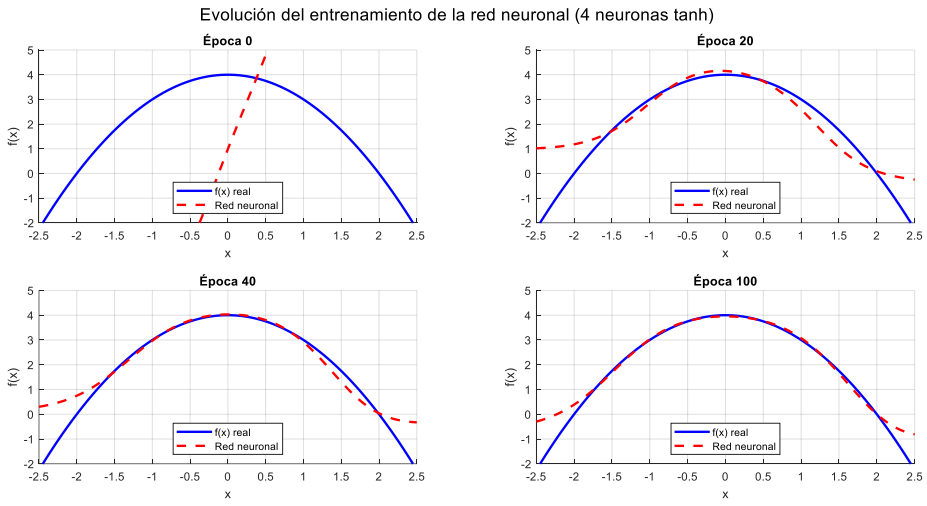
1.4 Redes guiadas por datos (Data driven)

Dado un conjunto de datos dado (x_i, y_i) se busca una función $y = f_\theta(x)$ que capte la dependencia entre las variables contenida en la nube de puntos.

La red se entrena para minimizar la función de pérdida

$$\min_\theta \ \mathcal{L}(\theta) = \min_\theta \sum_i c_i \| f_\theta(x_i) - y_i \|^2$$

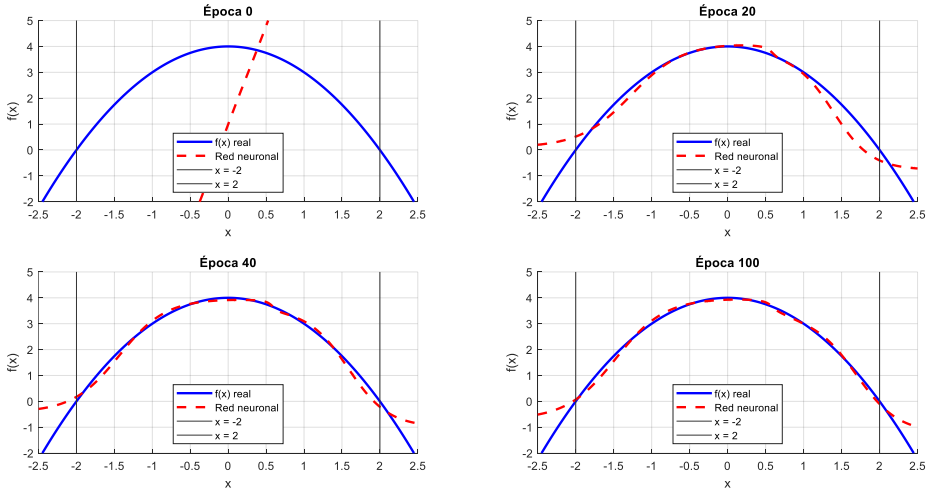
Ejemplo 1. Red que aprende una función a partir de su evaluación en un conjunto de puntos de un intervalo. En este caso $f(x) = 4 - x^2$ en el intervalo $[-2,2]$ con una red de cuatro neuronas.



El aprendizaje no se produce de la misma manera en el interior y en la frontera

Se observa que en general el aprendizaje es más lento en los extremos del intervalo. Para mejorar la convergencia pueden introducirse coeficientes en los puntos extremos del intervalo. En este caso se toma un coeficiente de 40 en los puntos del extremo.

Pesar con un factor 40 la frontera introduce cierta inestabilidad en el interior.

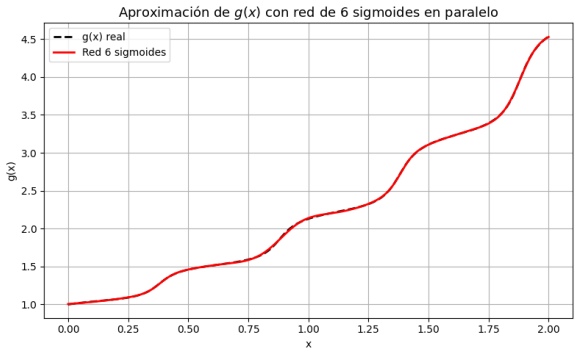


Multiplicar por un factor 100 en este caso introduce tanta inestabilidad en la red que imposibilita el aprendizaje.

Ejemplo 2. Red guiada por datos que aprende funciones más sofisticadas.

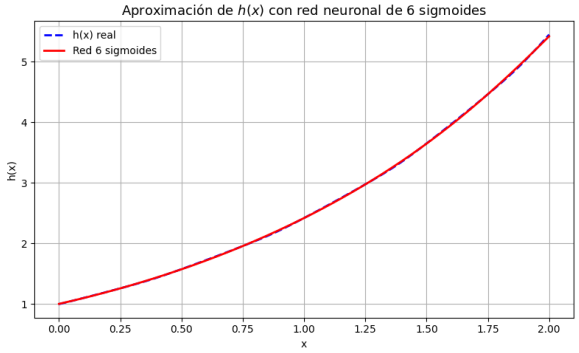
$$g(x) = e^{\int_0^x \frac{1}{2+1.5\sin(4\pi s)} ds}, \quad x \in [0,2].$$

La nube de puntos se genera por aproximación numérica de las integrales.



Podemos incluso aproximar funciones definidas mediante integrales anidadas como, por ejemplo

$$h(x) = \int_0^x e^{g(t)} dt + 1 = \int_0^x e^{\int_0^t \frac{1}{2+1.5\sin(4\pi s)} ds} dt + 1$$



Redes pequeñas tienen la capacidad de aprender funciones complejas

PRIMERA ESTRATEGIA PARA APRENDER SOLUCIONES DE ECUACIONES DIFERENCIALES.

Redes neuronales Data Driven y Ecuaciones diferenciales: Una estrategia posible para aprender soluciones de ecuaciones diferenciales consiste en dos etapas:

1. Aplicar un método numérico clásico (como diferencias finitas, elementos finitos, etc.) para generar una nube de puntos que aproximan la solución.
2. Entrenar una red neuronal puramente *data-driven* que, a partir de estos datos, aprenda una función (que será la solución) que interpole la solución y ofrezca una expresión continua y diferenciable.

Este enfoque convierte el problema diferencial en un problema de regresión, trasladando toda la información física al conjunto de datos, no al modelo.

Sin embargo, este enfoque presenta una limitación clave:

la red no "sabe" nada de la ecuación diferencial original.

Su conocimiento del problema se reduce a los puntos que ha visto, lo que puede dificultar la generalización o la precisión en regiones poco muestreadas.

2. 2019 – Redes PINN: la IA une datos y física

2.1 ¿Qué son las PINNs y por qué funcionan?

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

Una PINN (Physics-Informed Neural Network) es una red neuronal a la que se entrena tanto para ajustarse a datos como también para satisfacer una ecuación diferencial que describe un sistema físico y para aprovechar cualquier información “extra” de la que se disponga.

¿Cómo entrena?

Utilizando una función de pérdida que incluye varios términos

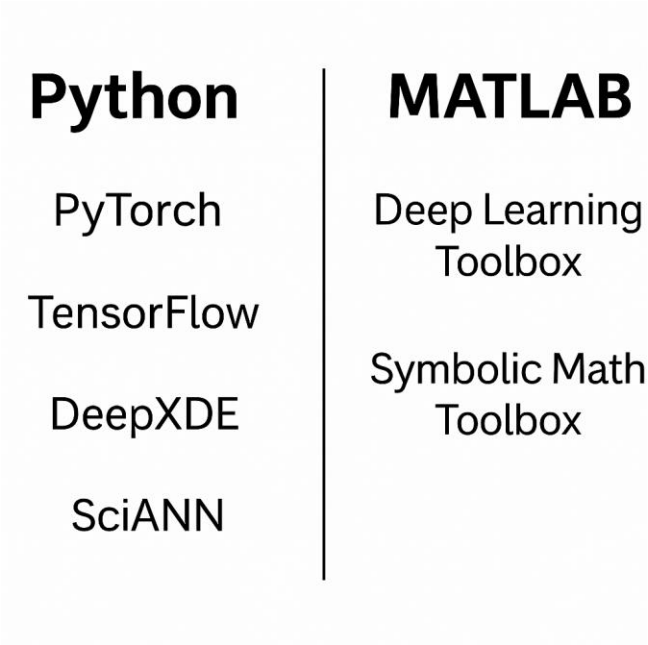
$$\mathcal{L} = \mathcal{L}_{datos} + \mathcal{L}_{Física} + \mathcal{L}_{condiciones}$$

- $\mathcal{L}_{datos}$ es el error sobre datos conocidos (si los hay)
- $\mathcal{L}_{Física}$ es el residuo de la ecuación diferencial (se usa derivación automática)
- $\mathcal{L}_{condiciones}$ es el error en las condiciones iniciales o de contorno.

Por qué funcionan las PINN.

- Las PINNs buscan aprender la solución de la ecuación diferencial utilizando toda la información disponible.
- No se utiliza una discretización en el sentido de los métodos numéricos, sino que se usa diferenciación automática.

Librerías que hemos utilizado



2.2 Diferenciación automática.

Linnainmaa, S. (1970).

The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors.

Report, Univ. of Helsinki.

La diferenciación automática es un método de derivación introducido en 1970 y que se utiliza en el ámbito del Machine Learning.

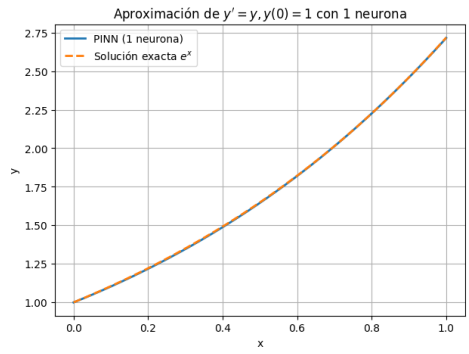
Diferenciación automática (una combinación entre análisis funcional y teoría de grafos)	
• Es exacta. • Proporciona una salida numérica de valores exactos de derivadas en puntos concretos. • Utiliza la regla de la cadena • Se lleva a cabo el proceso de derivación mediante grafos	• No usa fórmulas de derivación simbólica. • No usa aproximaciones numéricas

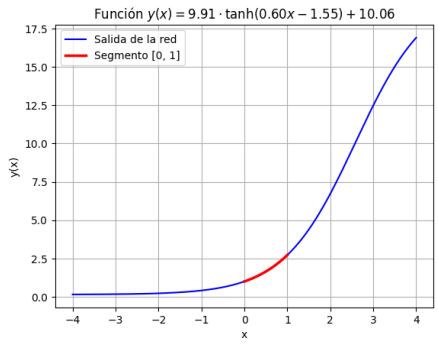
2.3 Aplicaciones.

Vamos a empezar resolviendo la ecuación diferencial

$$y' = y; \quad y(0) = 1$$

mediante una red PINN con una sola neurona con función de activación tangente hiperbólica.





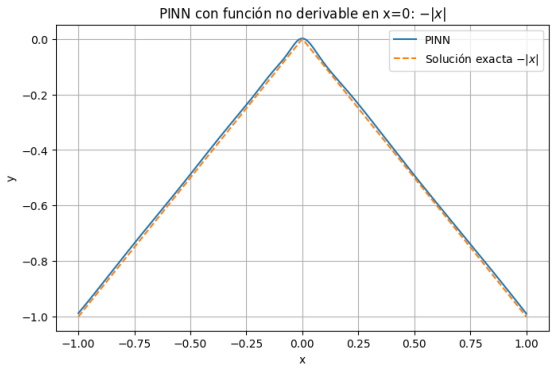
El trozo rojo es la solución de la ED y'=y

Podemos resolver también ecuaciones diferenciales con soluciones no derivables, por ejemplo:

$$y' = -signo(x); \quad y(0) = 0,$$

Donde $signo(x)$ es la llamada función signo, que vale 1 si el signo del número es positivo y menos 1 si el signo de x es negativo.

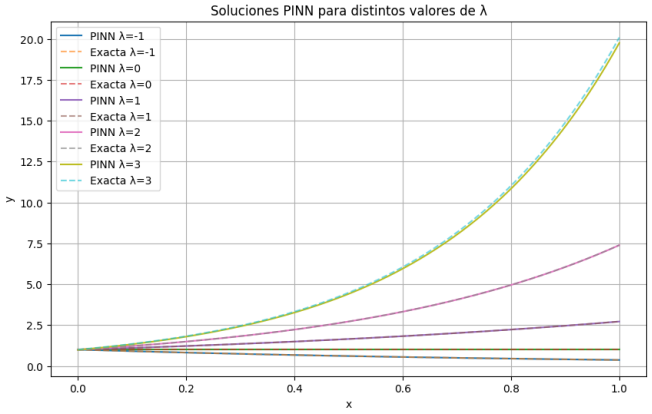
Con una arquitectura de 10 neuronas obtenemos



Una de las ventajas adicionales de las redes PINN es que pueden resolver ecuaciones diferenciales que involucran parámetros.

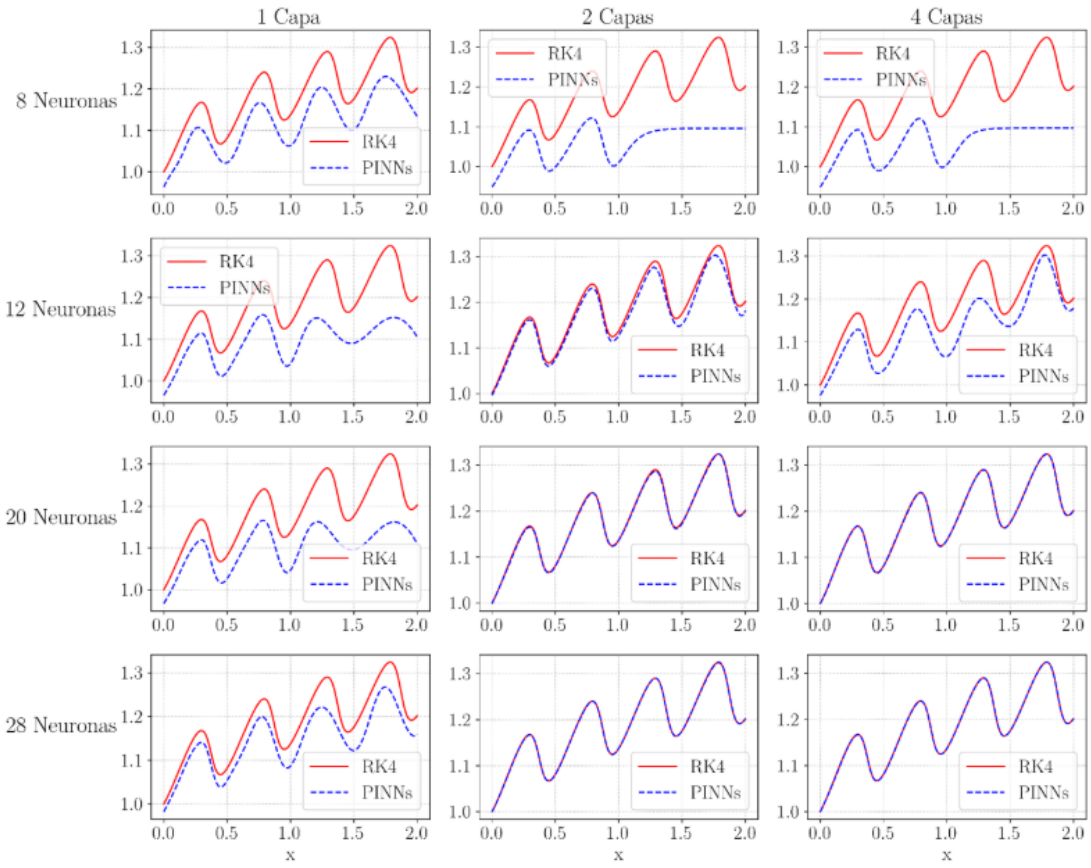
Vamos a resolver con un solo código y una red de 10 neuronas en paralelo la ecuación diferencial

$$y' = \lambda y; \quad y(0) = 1; \quad \lambda = -1,0,1,2,3$$



Podemos visualizar el comportamiento de las redes PINN en ecuaciones diferenciales más complejas:

$$y' = 1 - \frac{y}{2+\frac{3}{2}sin(4\pi x)}; \quad y(0) = 1; \quad x \in [0,2]$$



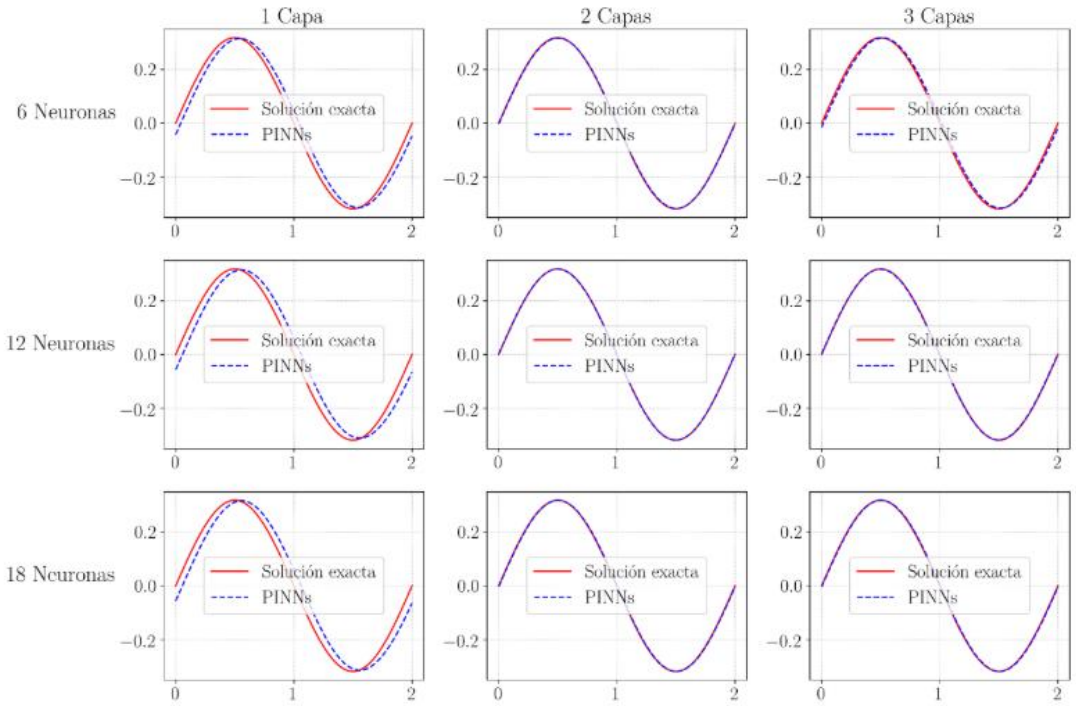
Fuente: TFG Salvador Dzimah

Datos: 5.000 épocas de entrenamiento. Sin pesos adicionales para el dato inicial. Tasa de aprendizaje 0.01

Oscilador armónico clásico.

$$y'' + \pi^2 y = 0; \quad y(0) = 0; \quad y'(1) = 1; \quad x \in [0,2].$$

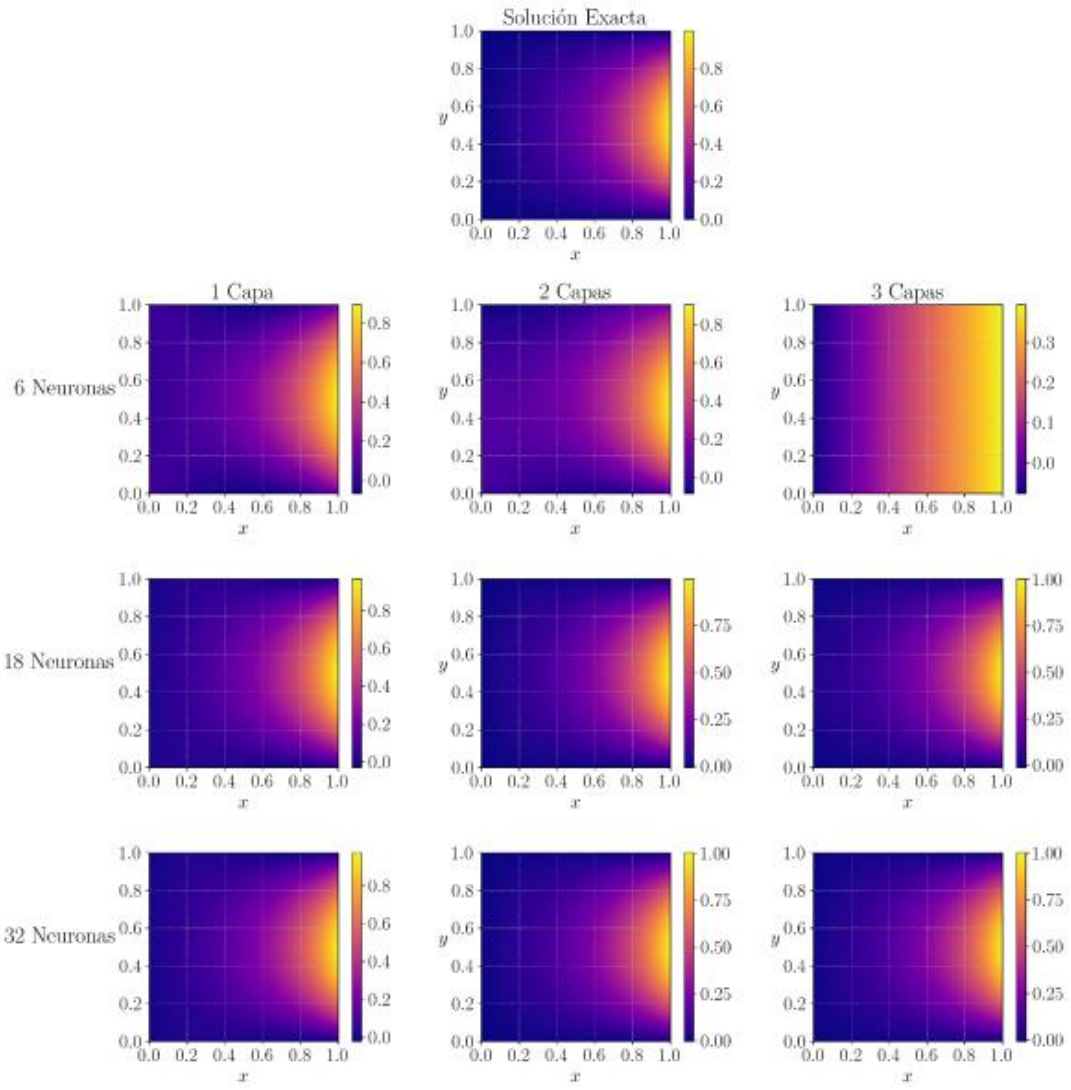
Mismos datos de entrenamiento que en el ejemplo anterior.



Fuente: TFG Salvador Dzimah

Ecuación de Laplace

$$u_{xx} + u_{yy} = 0; \quad u(x,0) = 0; \quad u(x,1) = \sin(\pi x); \quad u(0,y) = u(1,y) = 0$$



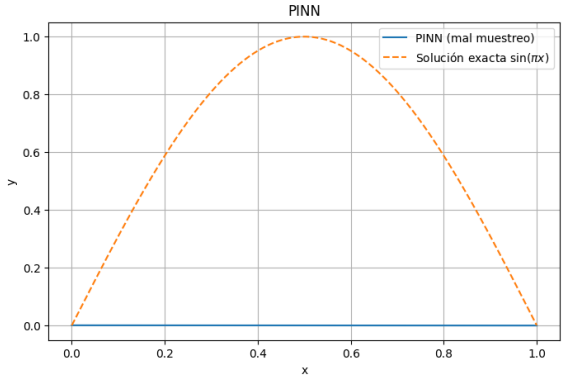
Fuente: TFG Salvador Dzimah

2.4 Problemas de las redes PINN.

En situaciones de falta de unicidad las redes PINN pueden no aproximar la solución deseada. Por ejemplo, la ecuación diferencial

$$y'' + \pi^2 y = 0; \quad y(0) = 0; \quad y(1) = 0;$$

Tiene dos soluciones $y(x) = sin(\pi x)$ y también $y(x) = 0$.



3. Mejoras de las redes PINN: modelos híbridos y computación cuántica

Primera idea:

Una de las mejoras que se pueden introducir para mejorar las soluciones proporcionadas por las redes PINN es combinar estas redes con métodos numéricos clásicos

Idea clave: Las redes pueden complementar los métodos numéricos clásico, no sustituirlos

Segunda idea:

Cambiar el entorno de trabajo. La computación cuántica

3.1 Introducción conceptual a la computación cuántica (cúbits, puertas, circuitos)

La computación cuántica se ha desarrollado desde los años 80 del siglo pasado para

aprovechar propiedades específicas del mundo subatómico.

La primera idea importante en relación con la computación cuántica es que

en un computador cuántico no hay una unidad central
que reciba instrucciones secuenciales
y las ejecute sobre datos almacenados en memoria.

Computación clásica vs Computación cuántica

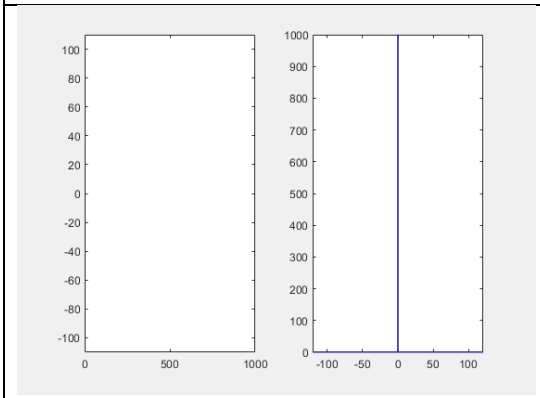
En computación cuántica cada vector, cada pieza de información, debe entenderse como un estado de un sistema cuántico real que existe en el interior del ordenador cuántico y cada matriz unitaria debe entenderse como un proceso o experimento real que se desarrolla o se lleva a cabo en el interior del propio ordenador

Literalmente, los computadores cuánticos
son laboratorios cuánticos ultrarrápidos.

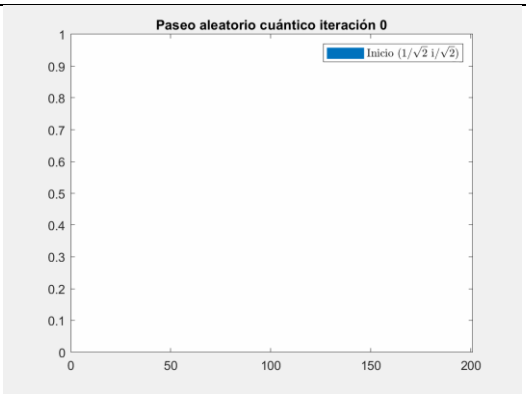
El procesado de la información
se realiza a través de la realidad física
y no a través de reglas de lógica formal.

En computación cuántica no se piensa, se actúa.

Un problema de difusión según reglas clásicas vs según reglas cuánticas



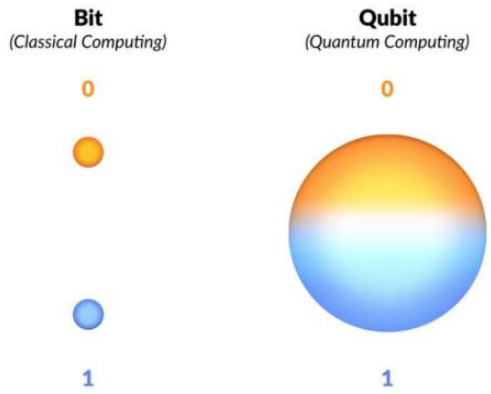
Paseo aleatorio clásico. Los fenómenos de difusión gaussiana tienden a concentrar las partículas en el centro



Paseo aleatorio cuántico.
La acumulación de ondas en el centro crea fenómenos de interferencia que empujan a las partículas hacia los extremos

Cúbits, puertas y circuitos cuánticos.

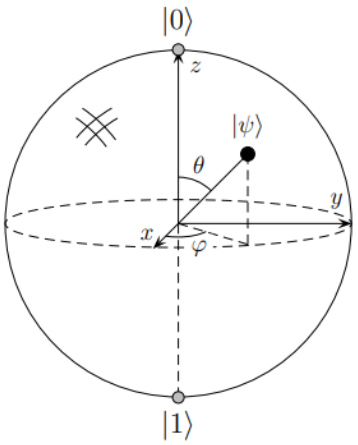
En computación cuántica, la unidad básica de información se llama **cúbit** (*quantum bit*). A diferencia del bit clásico, que solo puede estar en uno de dos estados posibles (0 o 1), el cúbit puede encontrarse en una **superposición** de ambos estados al mismo tiempo.



El bit representa una elección (0 frente a 1).
El cúbit no representa una elección,
sino una amplitud de posibilidad.

Entre las tecnologías detrás de los ordenadores cuánticos opciones podemos citar:

- **Cúbits superconductores**, utilizados por IBM y Google.
- **Cúbits de trampas de iones**, empleados por IonQ y Honeywell.
- **Cúbits fotónicos**, con aplicaciones clave en comunicación cuántica.
- **Cúbits basados en el spin de electrones en puntos cuánticos**, explorados por Intel y otras compañías.
- **Cúbits basados en defectos en diamante**, como los centros NV, aplicados en sensores cuánticos.

Representación matemática de un cúbit	
	$\left\|\psi > = cos\left(\frac{\theta}{2}\right)\right\| 0 > + e^{i\varphi} sin\left(\frac{\theta}{2}\right) \|1 > = \left(cos\left(\frac{\theta}{2}\right), e^{i\varphi} sin\left(\frac{\theta}{2}\right)\right),$ $\theta \in [0, \pi], \ \ \varphi \in [0, 2\pi],$ $\|0 >= \|\psi >_{\theta=0} = (1,0),$ (polo norte de la esfera de Bloch) $\|1 >= \|\psi >_{\theta=\pi} = (0,1),$ (polo sur de la esfera de Bloch)

Conviene subrayar que **la esfera de Bloch no es un objeto físico real**: no hay ninguna esfera girando dentro de un computador cuántico. Se trata de una representación matemática idealizada, que permite visualizar de forma intuitiva la información contenida en un cúbit, así como su evolución bajo diferentes operaciones cuánticas.

La esfera de Bloch actúa como un lenguaje visual unificado

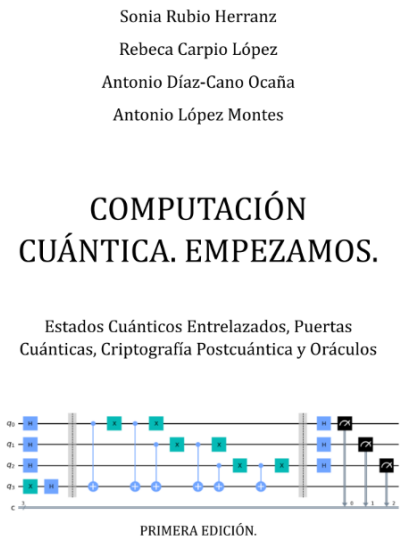
Hay muchos procesos elementales que se pueden llevar a cabo dentro de un computador cuántico. Cada uno de estos procesos viene representado por una matriz unitaria que se llama PUERTA CUÁNTICA. Estas puertas actúan como **transformaciones unitarias** sobre los cúbits y constituyen los bloques fundamentales con los que se construyen los circuitos cuánticos.

Un ejemplo de puesta cuántica es la puerta de rotación $R_y(\theta)$ que representa la rotación del estado de un cúbit en torno al eje Y de la esfera de Bloch.

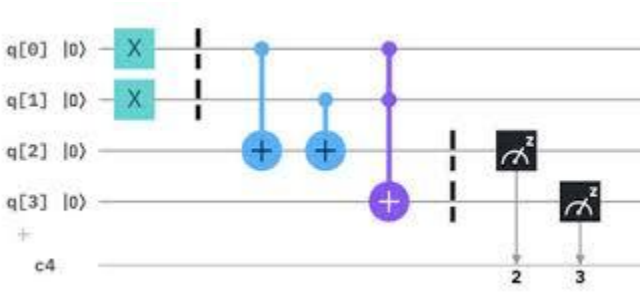
Su matriz asociada es

$$R_y(\theta) = \begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix}$$

Un libro sobre computación cuántica con un ejemplo de circuito cuántico en la portada



Un circuito cuántico es una sucesión de cúbits y puertas de tal manera que al final se hace una medida de uno o varios o todos los cúbits del circuito



Un circuito cuántico se construye a partir de cúbits (vectores) y de puertas (matrices) que actúan sobre esos vectores. Por tanto, se trata de un entorno de trabajo lineal. ¿Dónde está la no linealidad? En el proceso de medida.

Comparación: Redes de IA clásica e IA cuántica

Aspecto	Red neuronal clásica	Red neuronal cuántica
Evolución interna	No lineal (por funciones de activación)	Lineal (evolución unitaria)
Fuente de no linealidad	Función de activación	Proceso de medida (colapso probabilístico)
Acceso a la información	Directo (salida numérica tras activación)	Indirecto (probabilidad tras medición)
Naturaleza de la salida	Determinista	Basada en distribuciones de probabilidad

3.2 Representación de funciones en computación cuántica

En computación cuántica las funciones se representan **a través del estado de un sistema físico**. Y más concretamente, a través de:

- **Ángulos de rotación** de cúbits (codificación angular)
- **Amplitudes de probabilidad** (codificación en amplitud)
- **Fases relativas** (codificación en fase)
- **Estructuras de superposición o interferencia**

Vamos a ver un ejemplo sencillo. Imaginemos que partimos de un cúbit en un estado $|0>$ al que hacemos pasar por una puerta $R_y(x)$. El cúbit evoluciona hacia un estado

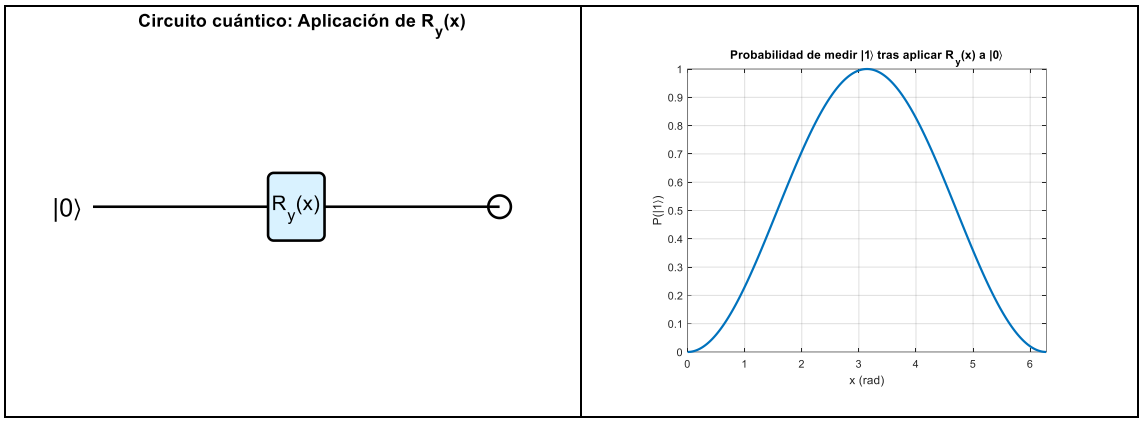
$$|\psi>=\begin{pmatrix}\cos\left(\frac{x}{2}\right) & -sin\left(\frac{x}{2}\right)\\ sin\left(\frac{x}{2}\right) & \cos\left(\frac{x}{2}\right)\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}=\left(cos\left(\frac{x}{2}\right), sin\left(\frac{x}{2}\right)\right)$$

Al medir las probabilidades de los estados $|0>$ y $|1>$ vamos a obtener respectivamente

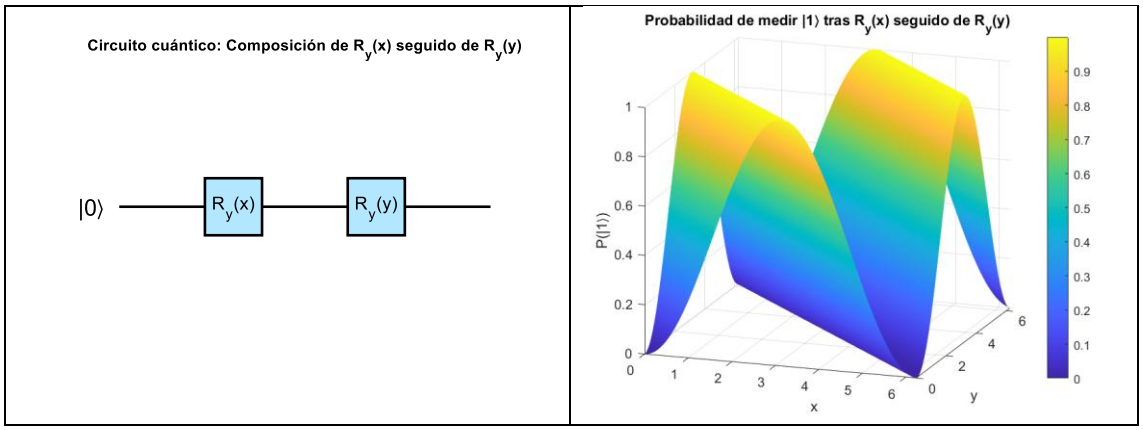
$$f(x)=P_{|0>}=\left(cos\left(\frac{x}{2}\right)\right)^2$$

$$g(x)=P_{|1>}=\left(sin\left(\frac{x}{2}\right)\right)^2$$

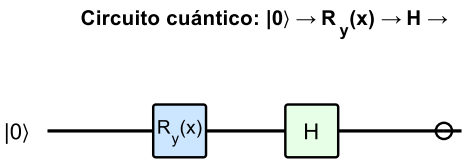
Estas funciones son ‘funciones puras’ que provienen de las medidas de un experimento cuántico, no son aproximaciones decimales mediante series de potencias o algún mecanismo similar.



Podemos aprovechar las propiedades de las matrices de giro para llevar a cabo una representación de funciones de dos variables:

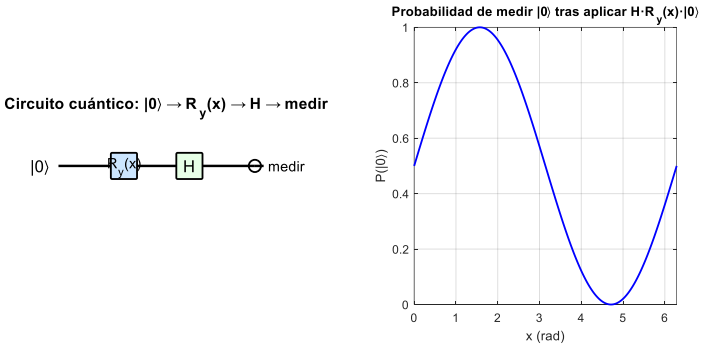


Otra posibilidad es utilizar el siguiente circuito

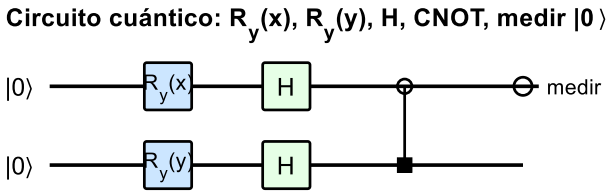


La medida de la probabilidad del estado $|0>$ nos va a dar la función

$$f(x) = \frac{1 + \sin(x)}{2}$$

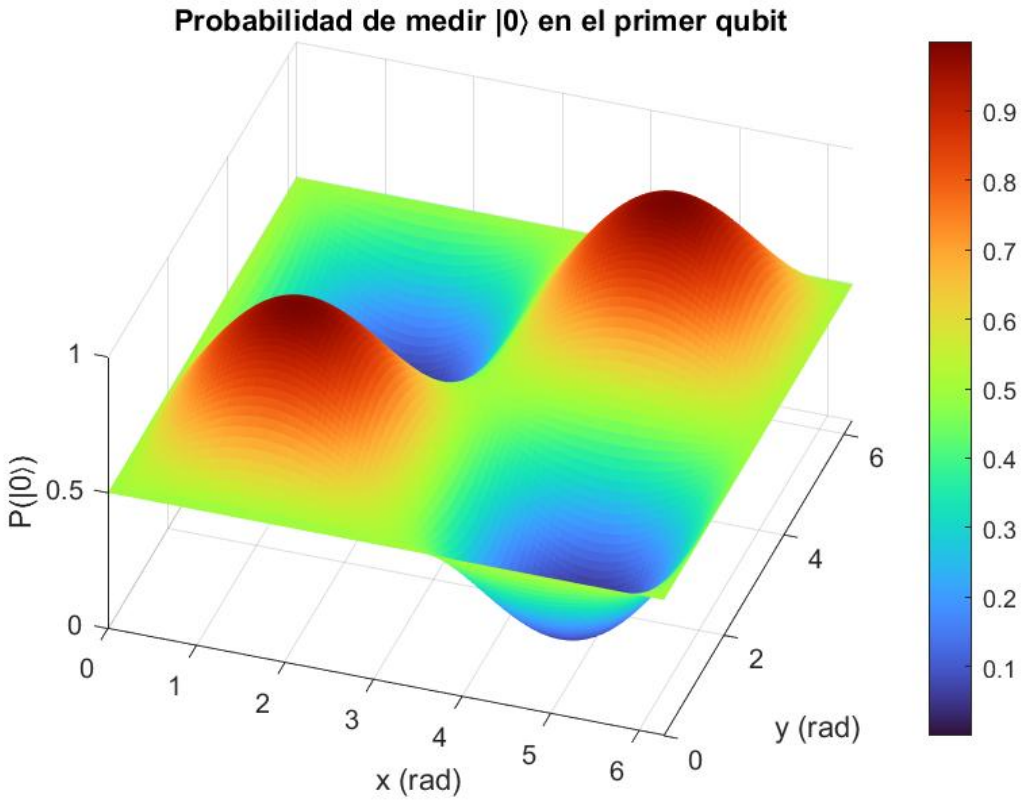


Otra opción es el siguiente circuito,



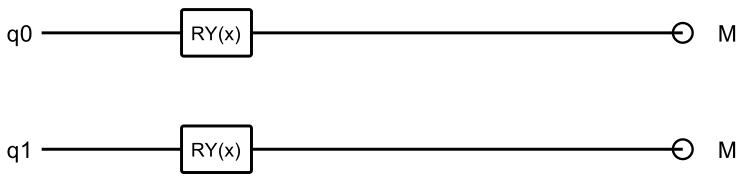
En este caso la puerta CNOT hace que el estado del **qubit 0 (control)** influya **no linealmente** sobre el estado final del **qubit 1 (target)**, introduciendo **interferencia cruzada** que da a la función

$$f(x, y) = \frac{1}{2}(1 + \cos(x - y)).$$



Hay otras arquitecturas que son interesantes cuando utilizamos varios cúbits. Por ejemplo, podemos construir un circuito con dos cúbits cada uno de ellos con una puerta Ry(x).

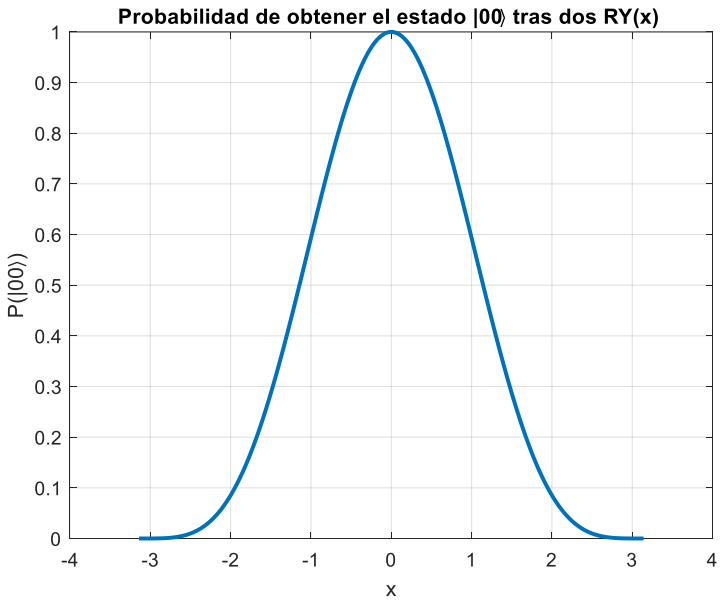
Circuito cuántico: RY(x) en dos qubits y medición P(00)



La medida de la probabilidad del estado $|00>$ genera la función

$$f(x) = \left(cos\left(\frac{x}{2}\right)\right)^4$$

que se representa en la siguiente figura.

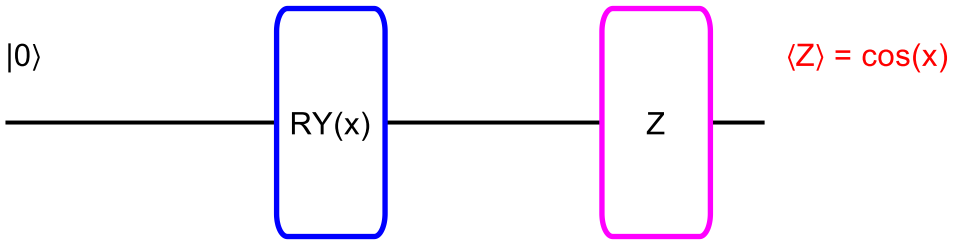


Asimismo, haciendo un circuito con n cúbits y midiendo la probabilidad del esta do $|00 \ldots 00>$ podemos construir un circuito cuántico cuya salida sea

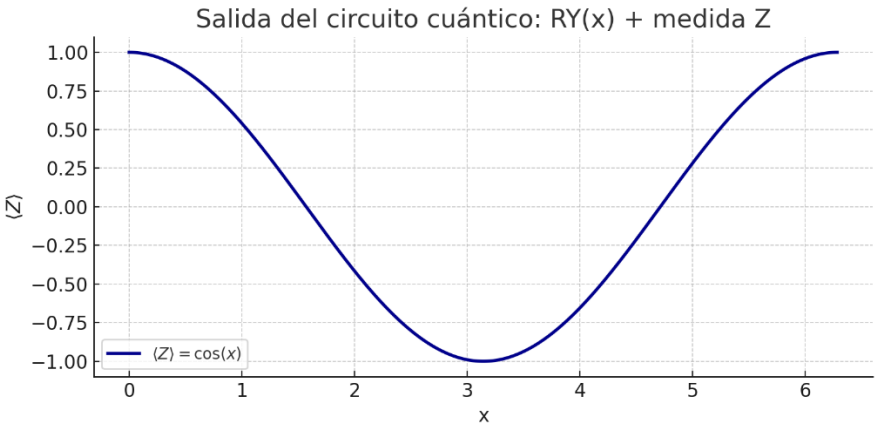
$$f(x) = \left(cos\left(\frac{x}{2}\right)\right)^{2n}$$

Por otro lado, podemos construir circuitos que tengan como salida funciones más simples. A continuación, tomamos un circuito con un cúbit y **medimos a salida la probabilidad del estado $|0>$ menos la probabilidad del estado $|1>$.**

Circuito: RY(x) seguido de medida Z → salida = cos(x)

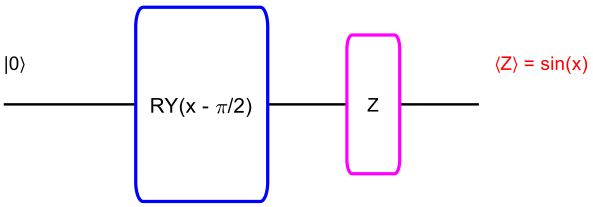


Obtendremos como resultado la función $f(x) = \cos(x)$.



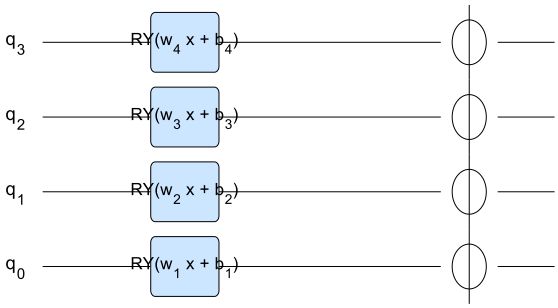
Si ahora tomamos el mismo circuito, pero introducimos una puerta $Ry\left(x-\frac{\pi}{2}\right)$ tendremos como salida la función seno

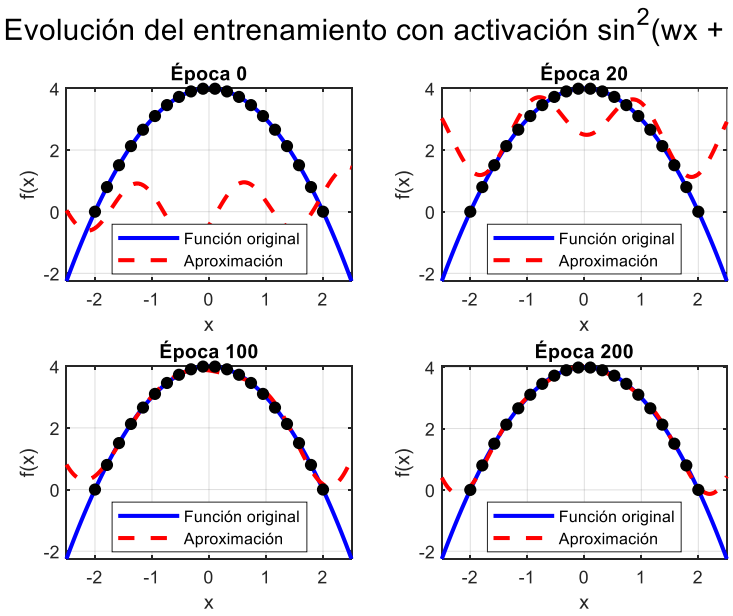
Circuito: RY(x - π/2) seguido de medida Z → salida = sin(x)



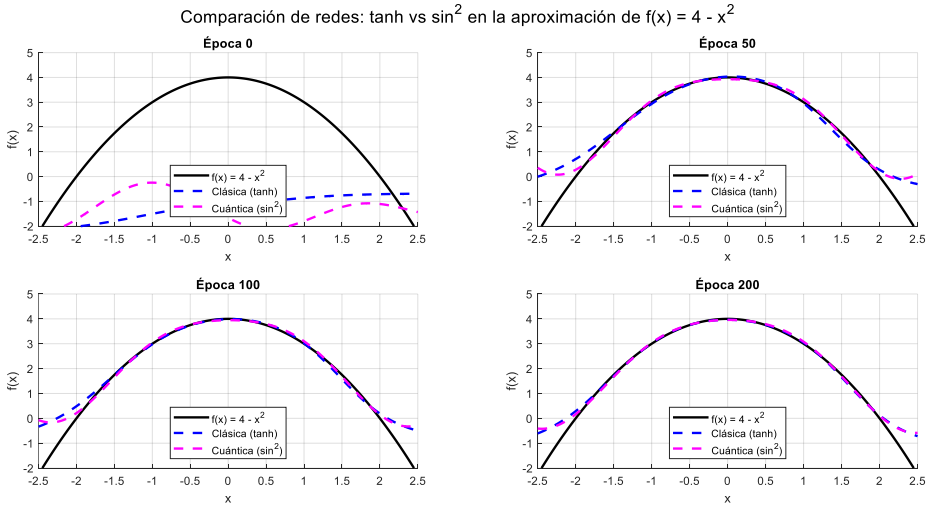
Vamos a ver la capacidad de una red de perceptrones cuánticos de aproximar la función parabólica $f(x) = 4 - x^2$

Red de 4 perceptrones cuánticos en paralelo





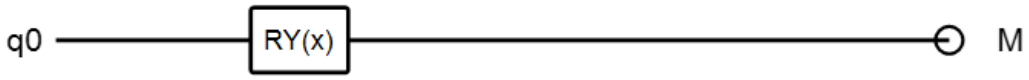
Tenemos la siguiente figura de comparación de las aproximaciones clásicas y cuánticas



Aviso. A veces las cosas no acaban de salir bien en computación cuántica.

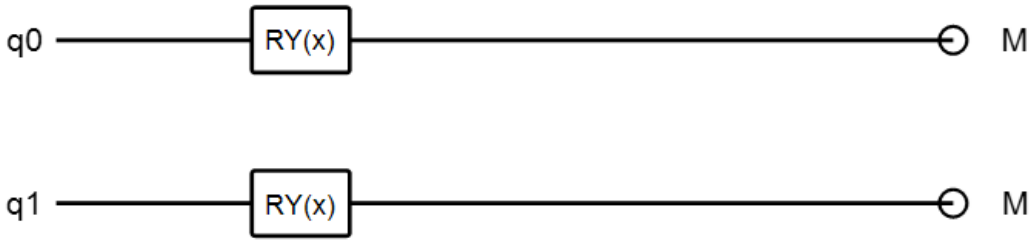
Segunda estrategia: Utilizar redes neuronales con funciones de activación tipo $\left(\boldsymbol{cos}\left(\frac{x}{2}\right)\right)^{\boldsymbol{n}}$

Un cúbit con una puerta Ry(x), es decir una arquitectura como esta



genera una probabilidad del estado $|0>$ cuyo valor es $\left(\boldsymbol{cos}\left(\frac{x}{2}\right)\right)^{\boldsymbol{2}}$.

Si utilizamos una arquitectura de la forma



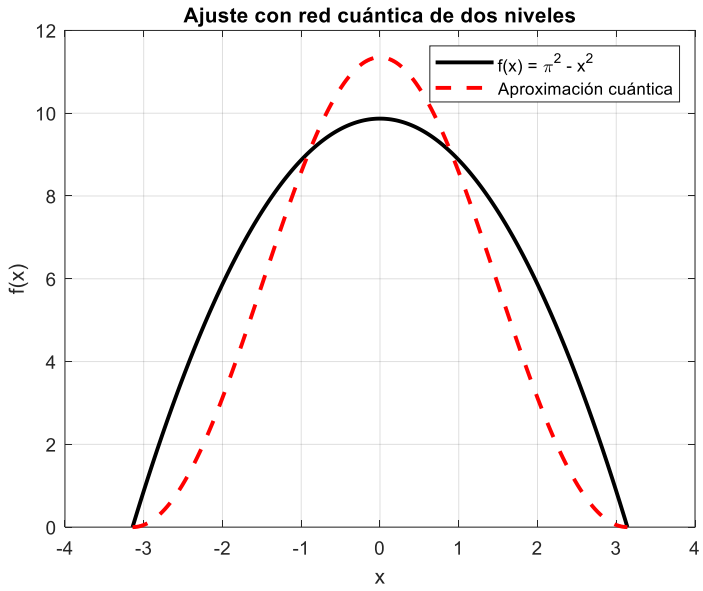
La probabilidad del estado $|00>$ vale $\left(\boldsymbol{cos}\left(\frac{x}{2}\right)\right)^{\boldsymbol{4}}$.

Y así sucesivamente, un circuito de n cúbits genera una probabilidad del estado $|00\ldots00>$ igual a $\left(cos\left(\frac{x}{2}\right)\right)^{2n}$.

En definitiva, vamos a tratar de aprender la parábola utilizando esta expresión:

$$\pi^2 - x^2 = a_1\left(cos\left(\frac{x}{2}\right)\right)^2 + a_2\left(cos\left(\frac{x}{2}\right)\right)^4, \quad x \in [-\pi, \pi]$$

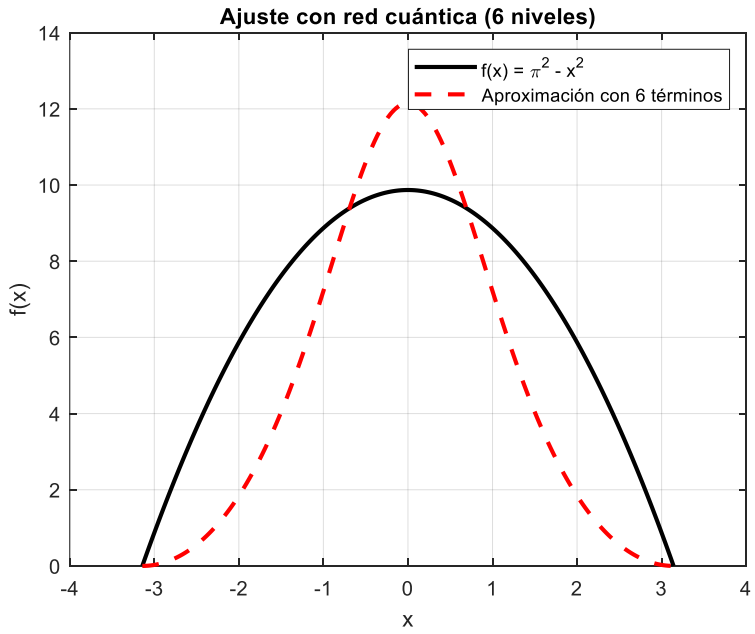
Tomando 20 puntos interiores en la función de pérdida vamos a tener la aproximación de la figura



Tomando seis términos en el desarrollo de cosenos, desde $\left(cos\left(\frac{x}{2}\right)\right)^2$ a

$\left(cos\left(\frac{x}{2}\right)\right)^{10}$

Obtenemos

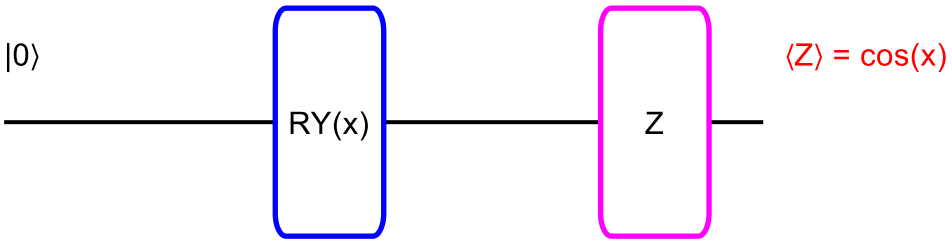


Se observa que el resultado no mejora. De hecho, empeora.

Tercera estrategia: Utilizar redes neuronales con funciones de activación tipo $cos(x)$, es decir, directamente desarrollos de Fourier.

Por el estudio hecho anteriormente sabemos que una red de la forma

Circuito: RY(x) seguido de medida Z → salida = cos(x)



Donde medimos como salida la probabilidad del estado $|0>$ menos la probabilidad del estado $|1>$ genera una salida de la forma

$$\left(\cos\left(\frac{x}{2}\right)\right)^2 - \left(\sin\left(\frac{x}{2}\right)\right)^2 = \cos(x)$$

Ahora bien, utilizando cúbits con puertas $RY(2x)$,… podemos construir una red neuronal cuya salida sea

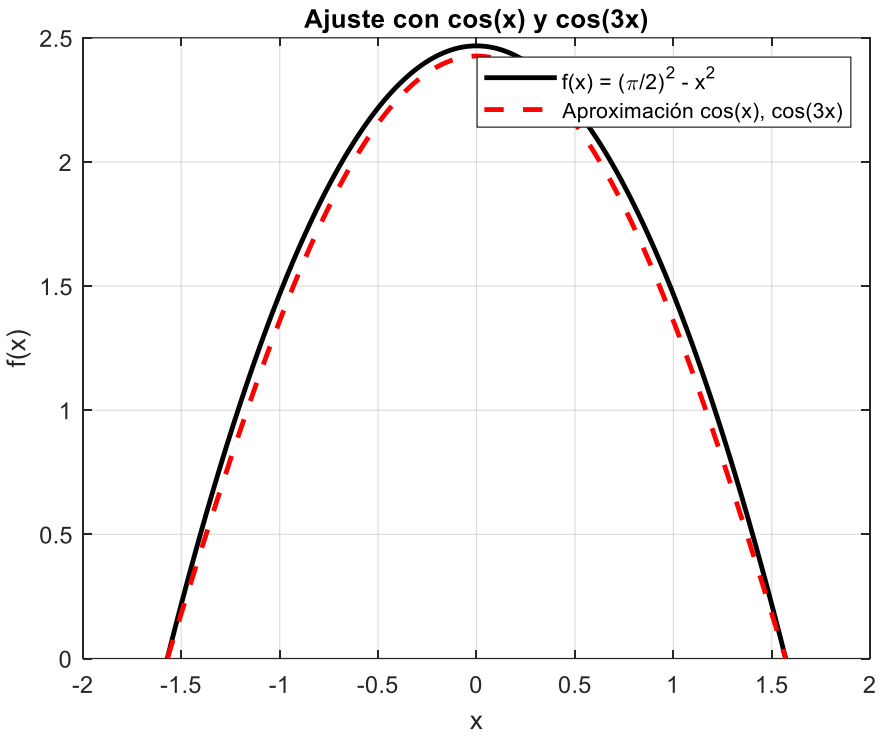
$$a_1 cos(x) + a_2 \cos(2x) + \cdots$$

Es decir, podemos generar desarrollos de Fourier.

Con este tipo de redes y este tipo de desarrollos vamos a tratar de aprender la función

$$\left(\frac{\pi}{2}\right)^2 - x^2 = a_1 cos(x) + a_2 cos(3x)\ , \quad x \in [\frac{\pi}{2}, \frac{\pi}{2}]$$

En este caso obtenemos



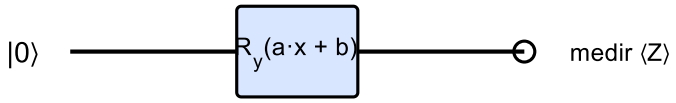
3.3 QPINNs

En computación cuántica resulta natural la aproximación de la solución de todo tipo de problemas relacionados con fenómenos oscilatorios. Por ejemplo, para aproximar el oscilador armónico

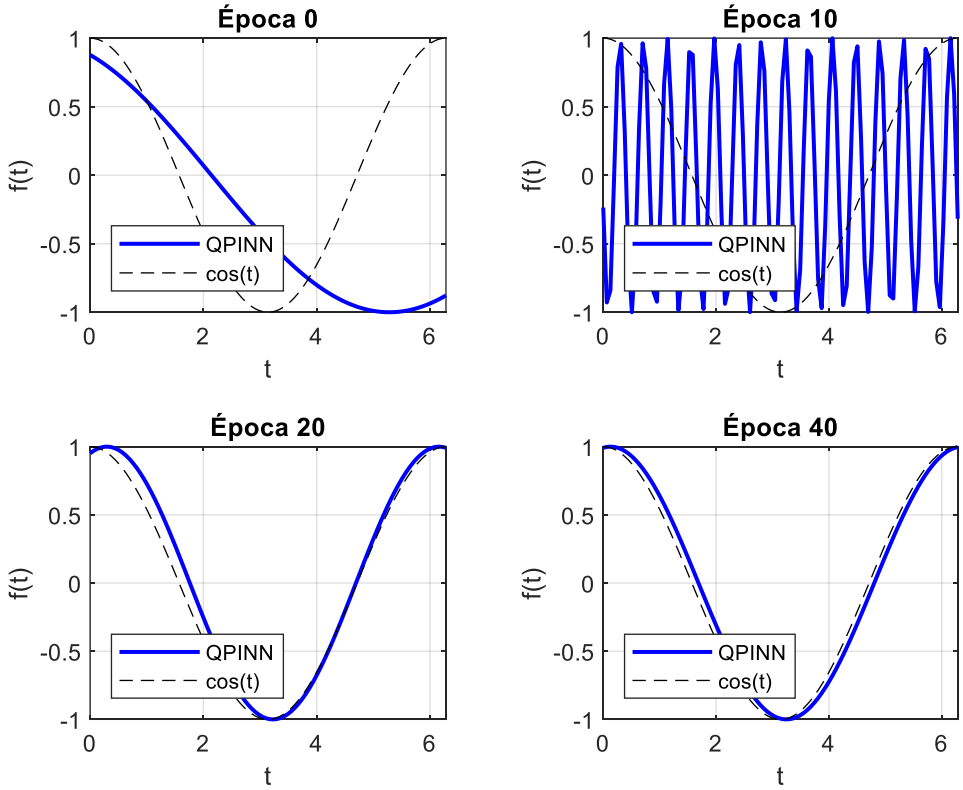
$$f'' + f = 0, \quad f(0) = 1, \quad f(2\pi) = 0, \quad x\epsilon[0,2\pi]$$

podemos utilizar la siguiente red cuántica

Circuito cuántico QPINN: $|0\rangle \rightarrow R_y(a \cdot x + b) \rightarrow \langle Z \rangle = P(0) - P(1)$



Obtenemos la siguiente secuencia de entrenamiento



Segundo ejemplo de red QPINN

Vamos a resolver la ecuación de Schrödinger unidimensional para el estado base del oscilador armónico cuántico

$$-\psi'' + x^2\psi = E\psi$$

$$\psi(-\infty) = \psi(\infty) = 0$$

La solución exacta es conocida:

$$\psi(x) = e^{-\frac{x^2}{2}}$$

Vamos a ver cómo queda la resolución de esta ecuación diferencial mediante un desarrollo de 4 cosenos del tipo

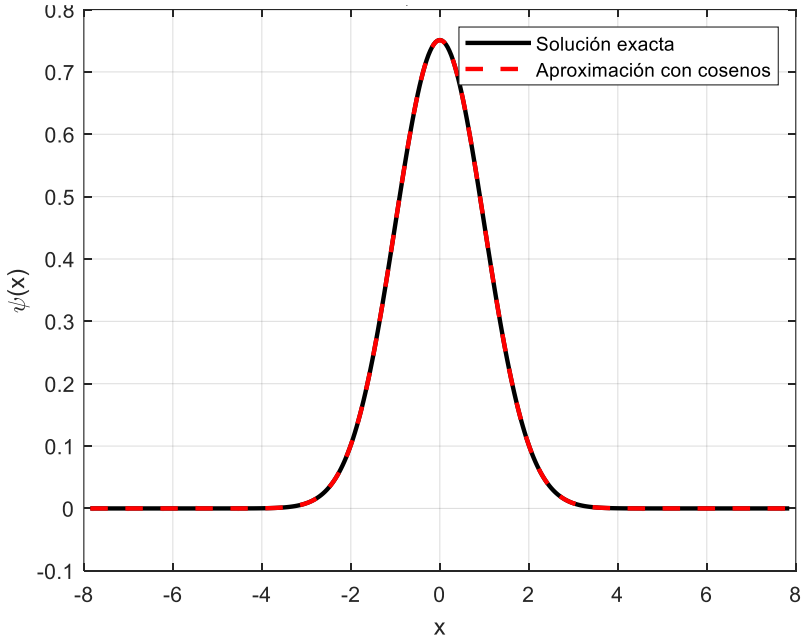
$$\phi_n(x) = cos\left(\frac{2n-1}{5}x\right); \quad n = 1,2,3,4; \quad x \in \left[-\frac{5\pi}{2}, \frac{5\pi}{2}\right]$$

La elección del intervalo no es arbtraria sino que garantiza que los cosenos se anulen en los extremos y también que sean ortogonales en este intervalo.

Buscamos una solución de la forma:

$$\psi(x) = a_1\phi_1(x) + \cdots + a_4\phi_4(x)$$

Obtenemos un sistema lineal que resolvemos y



Referencias.

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

- E. F. Caicedo Bravo and J. A. L. Sotelo, *Una aproximación práctica a las redes neuronales artificiales*, Santiago de Cali, Colombia: Programa Editorial Universidad del Valle, 2009. ISBN 978-958-670-767-1, ed. digital julio 2017, doi: 10.25100/peu.64.

  https://bibliotecadigital.univalle.edu.co/bitstream/handle/10893/10330/Una-aproximacion-practica-a-las-redes.pdf?sequence=6&isAllowed=y

- S. Rubio Herranz, R. Carpio López, A. Díaz-Cano Ocaña y A. López Montes, *Computación cuántica: ¡Empezamos!*, 1.ª ed. Madrid, España: Repro-Expres, S.L., 12 abr. 2024, 93 p. ISBN 978-84-127903-7-5.

- M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*, 10th Anniversary ed., Cambridge, U.K.: Cambridge Univ. Press, 2010.
  https://profmcruz.wordpress.com/wp-content/uploads/2017/08/quantum-computation-and-quantum-information-nielsen-chuang.pdf

- P. Wittek, *Quantum Machine Learning: What Quantum Computing Means to Data Mining*, Amsterdam, Netherlands: Academic Press, 2014.
  https://www.researchgate.net/publication/264825604_Quantum_Machine_Learning_What_Quantum_Computing_Means_to_Data_Mining/link/5ababcfba6fdcc71647085db/download?_tp=eyJjb250ZXh0Ijp7ImZpcnN0UGFnZSI6InB1YmxpY2F0aW9uIiwicGFnZSI6InB1YmxpY2F0aW9uIn19