

Resolución de EDP vía deep learning

Proyecto de investigación INF577

Mario Carlos Mallea Ruz

mario.mallea@sansano.usm.cl

Profesor: Carlos Valle
Universidad Técnica Federico Santa María

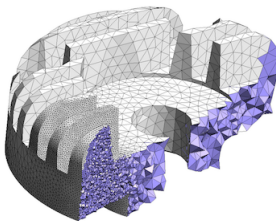
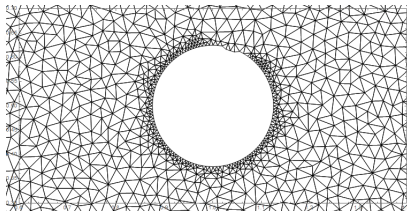
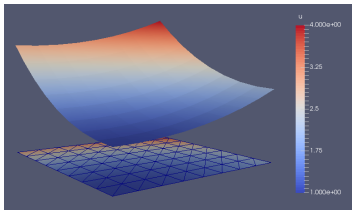
29 Diciembre, 2021



- 1 Introducción
- 2 Modelamiento
- 3 Análisis del método
- 4 Ejemplos
- 5 Conclusión

- Se presentará un estudio del esquema de resolución de EDP con redes neuronales bautizado como Deep Galerkin Method (DGM) por [Sirignano y Spiliopoulos 2018](#). Experimentando con el método en 4 EDP de distintas naturaleza y complejidades.
- DGM es libre de malla y converge a la solución de una clase particular de EDP de manera eficiente.

Introducción: FEM



Por qué un enfoque Deep

- La resolución de EDP altamente dimensionales es ineficiente con métodos basados en discretizar
- La red, puede encontrar de manera eficiente caminos entre puntos en alta dimensionalidad. [Kawaguchi, Huang y Kaelbling 2019](#)
- Resultados de densidad, convergencia y superación de la maldición de la dimensionalidad, sobre espacios de Sobolev. [Gühring, Raslan y Kutyniok 2020](#)

Taxonomía: Resolución de EDP con técnicas de machine learning. Willard y col. 2020

Trabajo pionero en la resolución de ecuaciones diferenciales con redes neuronales artificiales. Lagaris, Likas y Fotiadis 1998

En este trabajo muestran empíricamente que es factible aproximar/ representar con un kernel approach la función en variable compleja de Green que se utiliza para describir las impurezas magnéticas incrustadas en metales en el llamado Anderson impurity model. Arsenault y col. 2014

Proponen un esquema de resolución que combina los métodos clásicos del tipo Galerkin con un entrenamiento backpropagation con restricciones para obtener aproximaciones a ciertas integrales involucradas en el método clásico. Los autores destacan su aplicación a dominios irregulares y muestran una significativa mejora versus los métodos de elementos finitos más eficientes tanto en tiempo de computación como en precisión. Rudd y Ferrari 2014

En el contexto del desafío planteado por el problema de muchos cuerpos en la física cuántica que se origina en la dificultad de describir las correlaciones no triviales codificadas en la complejidad exponencial de la función de onda de muchos cuerpos, demuestran que usando una representación de los estados cuánticos basado en una RBM (restricted boltzmann machine) de la ecuación de onda puede reducir su complejidad a una forma computacionalmente manejable para la descripción de equilibrio y propiedades dinámicas de prototypical interacting spins models en una y dos dimensiones. [Carleo y Troyer 2017](#)

Estudian la committor function el objeto central de la teoría de caminos (estudio de transición entre estados gobernados por procesos estocásticos) la cual satisface la ecuación de Fokker-Planck en alta dimensionalidad, en particular proponen una aproximación con una red ff, formulada desde la forma variacional correspondiente, como un método de Galerkin no lineal, y teniendo en consideración las singularidades del dominio que puede presentar en el diseño de la arquitectura. [Khoo, Lu y Ying 2018](#)

El problema a resolver

Sea u una función desconocida definida en tiempo y espacio sobre la región $[0, T] \times \Omega$ con $\Omega \subset \mathbb{R}^d$ y asumimos que u satisface la EDP:

$$\begin{cases} (\partial_t + \mathcal{L}) u(t, \mathbf{x}) = 0, & (t, \mathbf{x}) \in [0, T] \times \Omega \\ u(0, \mathbf{x}) = u_0(\mathbf{x}), & \mathbf{x} \in \Omega \\ u(t, \mathbf{x}) = g(t, \mathbf{x}), & (t, \mathbf{x}) \in [0, T] \times \partial\Omega \end{cases}$$

El objetivo es aproximar u con una función $f(t, \mathbf{x}; \boldsymbol{\theta})$ con los parámetros **aprendidos** desde una red neuronal.

Dado secuencias \mathbf{x} de sub muestras aleatorias de diferentes partes del dominio , se buscará que la aproximación en aquellos puntos satisfaga la EDP. El objetivo es encontrar θ tales que minimicen el valor esperado de la equivocación.

$$\begin{aligned} L(\theta) = & \underbrace{\|(\partial_t + \mathcal{L})f(t, \mathbf{x}; \theta)\|_{[0, T] \times \Omega, \nu_1}^2}_{\text{operador diferencial}} + \\ & \underbrace{\|f(t, \mathbf{x}; \theta) - g(t, \mathbf{x})\|_{[0, T] \times \partial\Omega, \nu_2}^2}_{\text{condición de borde}} + \\ & \underbrace{\|f(0, \mathbf{x}; \theta) - u_0(\mathbf{x})\|_{\Omega, \nu_3}^2}_{\text{condición inicial}} \end{aligned}$$

Algorithm 2 Deep Galerkin con minibatch adaptativo

1. Inicializar los parámetros θ_0 y otros hiper parámetros como la tasa de aprendizaje α_n .

2. **Para cada época:**

2.1 Generar muestras aleatorios del interior del dominio y fronteras espacio/temporales:

- Generar (t_n, x_n) de $[0, T] \times \Omega$ según ν_1
- Generar (τ_n, z_n) de $[0, T] \times \partial\Omega$ según ν_2
- Generar w_n de Ω , según ν_3

2.2. **Para k pasos de descenso:**

2.2.1 Calcular la loss para el mini-batch correspondiente $s_n = \{(t_n, x_n), (\tau_n, z_n), w_n\}$:

- Calcular $L_1(\theta_n; t_n, x_n) = ((\partial_t + L) f(\theta_n; t_n, x_n))^2$
- Calcular $L_2(\theta_n; \tau_n, z_n) = (f(\tau_n, z_n) - g(\tau_n, z_n))^2$
- Calcular $L_3(\theta_n; w_n) = (f(0, w_n) - u_0(w_n))^2$
- Calcular $L(\theta_n; s_n) = L_1(\theta_n; t_n, x_n) + L_2(\theta_n; \tau_n, z_n) + L_3(\theta_n; w_n)$

2.2.2. Tomando un paso de descenso en los puntos aleatorios s_n con paso tipo Adam:

$$\theta_{n+1} = \theta_n - \alpha_n \nabla_{\theta} L(\theta_n; s_n)$$

2.3 **Repetir:** Adaptando el minibatch hacia zonas donde la loss no disminuye lo suficiente. Hasta que $\|\theta_{n+1} - \theta_n\|$ converja.

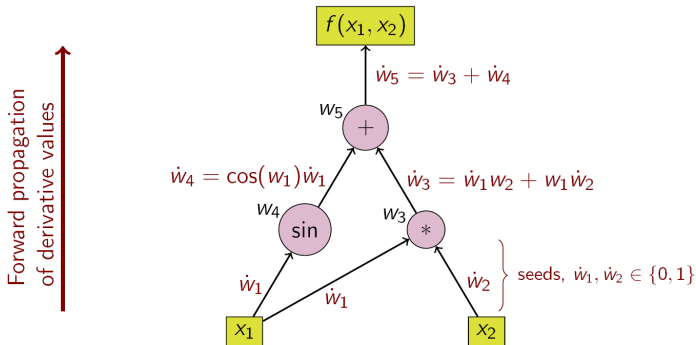
Output: aproximación f (red) definido por θ

El gradiente involucrado en el paso es calculado con el método de Backpropagation.

La adaptación evita muestrear constantemente la frontera.

¿Cómo se calculan esas derivadas? Diferenciación automática

Grafo de computación de tensorflow para $f(x_1, x_2) = x_1 x_2 + \sin(x_1)$



Ejemplo $f_{x_1} = x_2 + \cos(x_1) = \dot{w}_5$ para $\dot{w}_1 = 1; \dot{w}_2 = 0$

Arquitectura de DGM

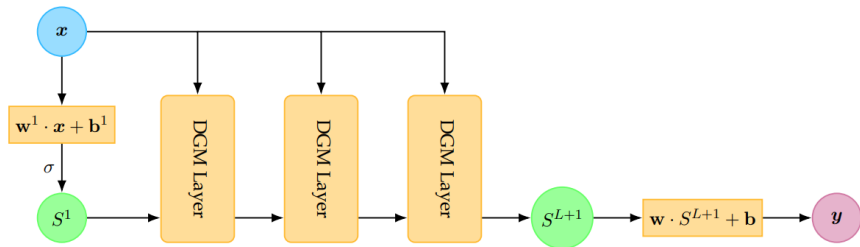


Figura: Arquitectura DGM

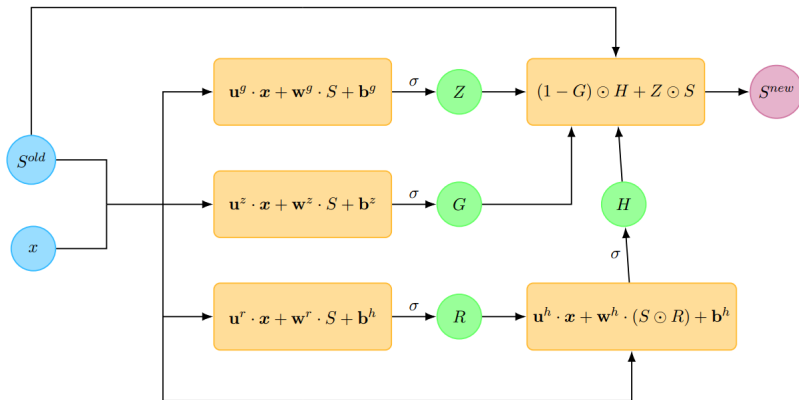


Figura: Capa DGM

Las operaciones involucradas:

$$\begin{aligned}S^1 &= \sigma(w^1 \cdot x + b^1) \\Z^\ell &= \sigma(u^{z,\ell} \cdot x + w^{z,\ell} \cdot S^\ell + b^{z,\ell}) & \ell = 1, \dots, L \\G^\ell &= \sigma(u^{g,\ell} \cdot x + w^{g,\ell} \cdot S^\ell + b^{g,\ell}) & \ell = 1, \dots, L \\R^\ell &= \sigma(u^{r,\ell} \cdot x + w^{r,\ell} \cdot S^\ell + b^{r,\ell}) & \ell = 1, \dots, L \\H^\ell &= \sigma(u^{h,\ell} \cdot x + w^{h,\ell} \cdot (S^\ell \odot R^\ell) + b^{h,\ell}) & \ell = 1, \dots, L \\S^{\ell+1} &= (1 - G^\ell) \odot H^\ell + Z^\ell \odot S^\ell & \ell = 1, \dots, L \\f(t, x; \theta) &= w \cdot S^{L+1} + b\end{aligned}$$

El espacio paramétrico es:

$$\theta = \left\{ w^1, b^1, \left(u^{z,\ell}, w^{z,\ell}, b^{z,\ell} \right)_{\ell=1}^L, \left(u^{g,\ell}, w^{g,\ell}, b^{g,\ell} \right)_{\ell=1}^L, \left(u^{r,\ell}, w^{r,\ell}, b^{r,\ell} \right)_{\ell=1}^L, \right. \\ \left. \left(u^{h,\ell}, w^{h,\ell}, b^{h,\ell} \right)_{\ell=1}^L, w, b \right\}$$

Como cada una de las sub capas contienen M neuronas de cálculo (osea, la operación se repite M veces) se tiene que la activación $\sigma : \mathbb{R}^M \rightarrow \mathbb{R}^M$ es una no linealidad ϕ element wise:

$$\sigma(z) = (\phi(z_1), \phi(z_2), \dots, \phi(z_M))$$

Por lo que los parámetros involucrados para un problema con d dimensiones espaciales, tienen tamaño:

$$\begin{aligned} w^1 &\in \mathbb{R}^{M \times (d+1)}, b^1 \in \mathbb{R}^M, \\ u^{z,\ell} &\in \mathbb{R}^{M \times (d+1)}, w^{z,\ell} \in \mathbb{R}^{M \times M}, b^{z,\ell} \in \mathbb{R}^M, \\ u^{g,\ell} &\in \mathbb{R}^{M \times (d+1)}, w^{g,\ell} \in \mathbb{R}^{M \times M}, b^{g,\ell} \in \mathbb{R}^M, \\ u^{r,\ell} &\in \mathbb{R}^{M \times (d+1)}, w^{r,\ell} \in \mathbb{R}^{M \times M}, b^{r,\ell} \in \mathbb{R}^M, \\ u^{h,\ell} &\in \mathbb{R}^{M \times (d+1)}, w^{h,\ell} \in \mathbb{R}^{M \times M}, b^{h,\ell} \in \mathbb{R}^M, \\ w &\in \mathbb{R}^{1 \times M}, b \in \mathbb{R} \end{aligned}$$

La aproximación vía DGM converge a la solución de una **EDP parabólica y quasi lineal** (entre otras condiciones) cuando el numero de neuronas de las capas ocultas tiende a infinito.

Se logra minimizar la Loss

Existe $f^n \in \mathcal{C}^n$ tal que $L(f) \rightarrow 0$ cuando $n \rightarrow \infty$

Se converge a la solución

$f^n \rightarrow u$ cuando $n \rightarrow \infty$ fuertemente en $L^\rho([0, T] \times \Omega)$, con $\rho < 2$

- Una característica relevante que no cuentan los métodos de resolución clásicos es que las soluciones obtenidas via redes neuronales son naturalmente **diferenciables** y tienen una forma **(semi) analítica** que puede ser transferida a cualquier cálculo posterior. Por ejemplo el tiempo de inferencia en una red neuronal es muy bajo, a diferencia de los métodos de Runge-Kutta que ofrecen soluciones discretas o soluciones con diferenciabilidad limitada como los elementos finitos.
- A diferencia de los métodos FEM, DGM puede superar la maldición de la dimensionalidad en el espacio paramétrico. [Gühring, Raslan y Kutyniok 2020](#)
- Para un problema de 200 dimensiones existen 200000 derivadas de segundo orden lo cual puede llegar a ser muy costoso para un método de diferenciación automática. Por lo que vale la pena proponer aproximaciones más rápidas de esas derivadas con montecarlo y diferencias finitas. [Sirignano y Spiliopoulos 2018](#)
- La poca disminución de la loss puede deberse a pobre aproximaciones del gradiente, por lo que tiene sentido reforzar el muestreo en aquellas zonas.

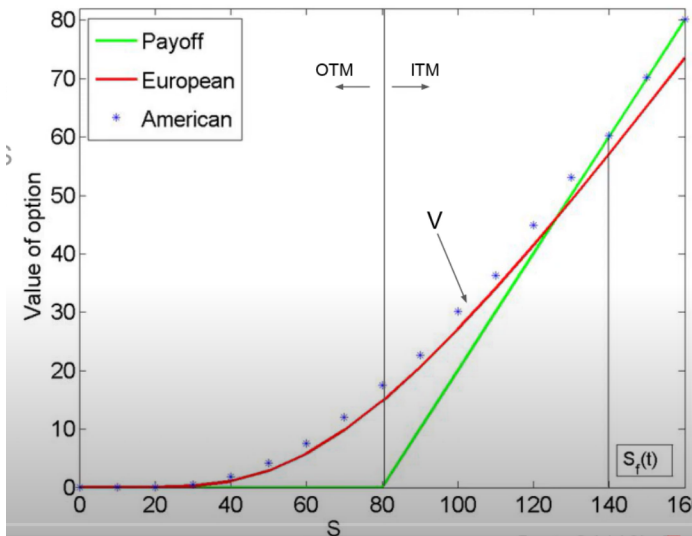
Aplicación en finanzas: European Call Options

Una **opción** es un instrumento financiero derivado de forma de un contrato que le da derecho al comprador, pero no la obligación de comprar o vender un activo subyacente (acciones, bonos, índices bursátiles etc) a un precio predeterminado (**strike** o precio de ejercicio) en una fecha concreta (vencimiento) **estilo europeo**.

El precio pagado por comprar la opción se llama **prima**.

El estilo europeo afirma que los instrumentos financieros son escritos sobre un fuente de incertidumbre modelado por un **movimiento Browniano**, con una liquidación que depende del nivel del subyacente en una fecha de vencimiento predeterminada. Lo que significa el movimiento Browniano es que el incremento de precios futuro (variable aleatoria desconocida) sería modelado por una distribución lognormal, y no dependería del pasado. Tiene una tendencia lineal, que no considera eventos anormales.

Distribución del precio de la opción con respecto al precio del activo



One dimensional Black-Scholes PDE

$$\begin{cases} \partial_t g(t, x) + rx \cdot \partial_x g(t, x) + \frac{1}{2} \sigma^2 x^2 \cdot \partial_{xx} g(t, x) = r \cdot g(t, x) \\ g(T, x) = G(x) = \max\{0, x - K\} \end{cases}$$

Solución:

$$g(t, x) = x\Phi(d_+) - Ke^{-r(T-t)}\Phi(d_-)$$

donde
$$d_{\pm} = \frac{\ln(x/K) + (r \pm \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}}$$

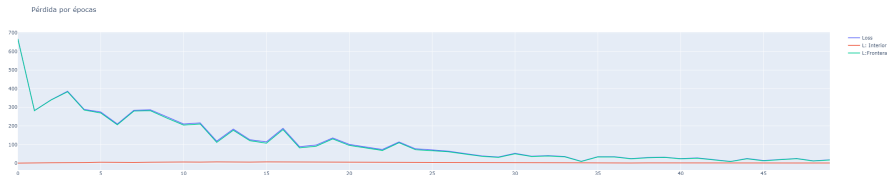
El problema a resolver considera: Una tasa de interés $r = 5\%$, volatilidad $\sigma = 25\%$, tiempo de vencimiento $T = 1$ y precio de ejercicio $K = 50$.

Loss: $L = L_1 + L_2$

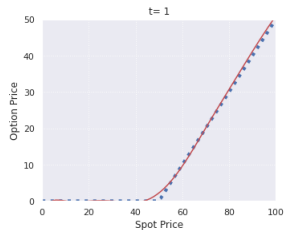
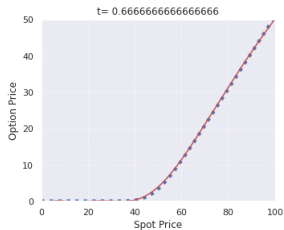
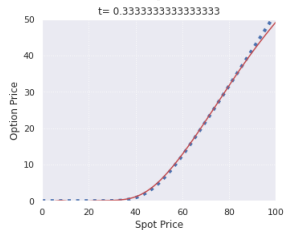
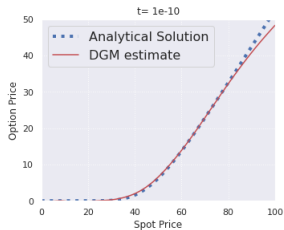
$$L_1(\theta) = \sum_{i=1}^{n_1} (NN_t(x, t; \theta) + \frac{1}{2} \sigma^2 NN(x, t; \theta)^2 NN_{xx}(x, t; \theta) + rx NN_x(x, t; \theta))^2$$

$$L_2(\theta) = \sum_{i=1}^{n_2} (\max\{0, x - K\} - NN(x, T))^2$$

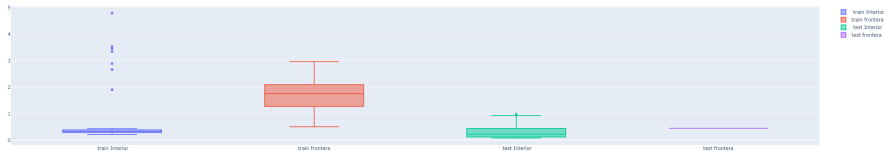
Con $(x, t) \sim (U[0, 2K], U[0, 1])$



Resultados



Distibución de los errores para muestras de train y test.



Mapa de calor del error en fase de test.



Mostrar la diferencia de construcción del minibatch.

RMSEt \ muestreo	Normal	Adaptativo
Interior	3.1	0.32
T final	0.71	0.45

Error en fase de interpolación o test.

La ecuación del calor es una importante EDP del tipo parabólica que describe la distribución del calor (o variaciones de la temperatura) en una región a lo largo del transcurso del tiempo.

Heat equation 2D

$$\frac{\partial u}{\partial t} = \nabla^2 u + f$$

$$\Omega \times (0, T]$$

$$u = u_D$$

$$\partial\Omega \times (0, T]$$

$$u = u_0$$

$$t = 0$$

Para $f = \beta - 2 - 2\alpha$, $u_D = 1 + x^2 + \alpha y^2 + \beta t$, $u_0 = 1 + x^2 + \alpha y^2$

La ecuación tiene solución $u = 1 + x^2 + \alpha y^2 + \beta t$ para $T = 2$, $\alpha = 3$, $\beta = 1,2$ y $\Omega = [0, 1]^2$

Mostrar solución

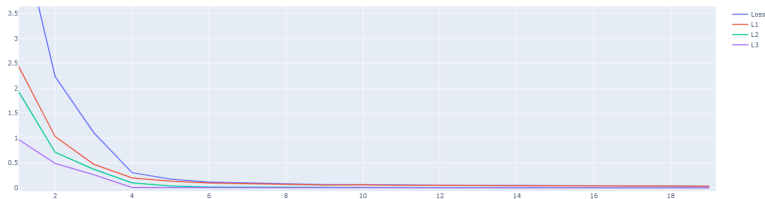
$$\text{Loss } L = L_1 + L_2 + L_3$$

$$L_1(\theta) = \sum_{i=1}^{n_1} (NN_t(X, t; \theta) - NN_{xx}(X, t; \theta) - NN_{yy}(X, t; \theta) - f)^2 \quad (X, t) \sim (U(\Omega), U[0, T])$$

$$L_2(\theta) = \sum_{i=1}^{n_2} (NN(X, t) - u_D(X, t))^2 \quad (X, t) \sim (U(\partial\Omega), U[0, T])$$

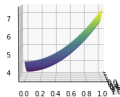
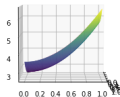
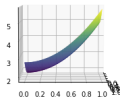
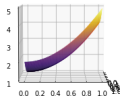
$$L_3(\theta) = \sum_{i=1}^{n_3} (NN(X, 0) - u_0(X))^2 \quad X \sim U(\Omega)$$

Pérdida por épocas



Resultado

Tiempos 0; 0,7; 1,4; 2



Error en fase de test o interpolación.

RMSEt \ muestreo	Normal	Adaptativo
Interior	$2.6 \cdot 10^{-3}$	$7.3 \cdot 10^{-4}$
Frontera	$2.6 \cdot 10^{-3}$	$1.1 \cdot 10^{-4}$
T inicial	$2.4 \cdot 10^{-3}$	$2.5 \cdot 10^{-4}$

La ecuación de onda es una EDP lineal de segundo orden para la descripción de ondas, como ocurren en la física clásica, como las ondas mecánicas (por ejemplo, ondas de agua, ondas de sonido y ondas sísmicas) u ondas de luz. Surge en campos como la acústica, electromagnética y dinámica de fluidos.

Wave equation 2D

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \quad \Omega \times [0, T]$$

$$u(x, y, t) = 0 \quad \forall (x, y) \in \partial\Omega$$

$$u(x, y, 0) = J_0 \left(\lambda_4 \sqrt{x^2 + y^2} \right) \quad \forall (x, y) \in \Omega$$

$$\frac{\partial u}{\partial t}(x, y, 0) = 0 \quad \forall (x, y) \in \Omega$$

Solución:

$$u(x, y, t) = J_0 \left(\lambda_4 \sqrt{x^2 + y^2} \right) \cos(c\lambda_4 t)$$

Donde:

Con $\Omega = \{(x, y) \mid x^2 + y^2 \leq 1\}$, $T = 3$ y $c = 2$.

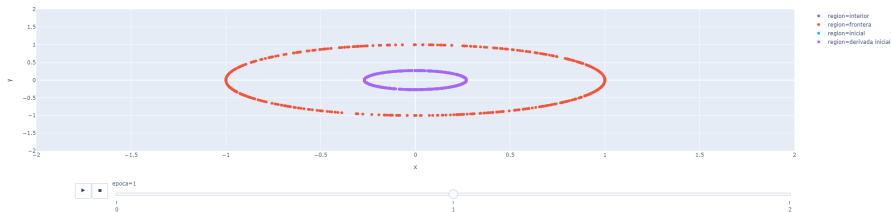
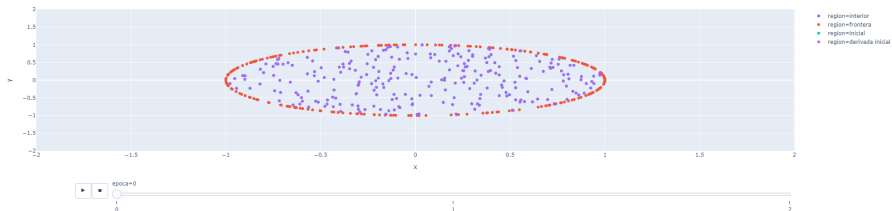
Donde $J_0(\cdot)$ representa la función de Bessel de primer tipo, definida como:

$$J_0(r) = \sum_{m=0}^{\infty} \frac{(-1)^m}{(m!)^2} \left(\frac{r}{2}\right)^{2m}$$

y λ_4 representa el cuarto cero de $J_0(\cdot)$

Mostrar solución

Muestreo



Loss: $L = L_1 + L_2 + L_3 + L_4$

$$L_1(\theta) = \sum_{i=1}^{n_1} (NN_{tt}(X, t; \theta) - c^2 (NN_{xx}(X, t; \theta) + NN_{yy}(X, t; \theta)))^2$$

$$(X, t) \sim (U(\Omega), U[0, T])$$

$$L_2(\theta) = \sum_{i=1}^{n_2} (NN(X, t))^2$$

$$(X, t) \sim (U(\partial\Omega), U[0, T])$$

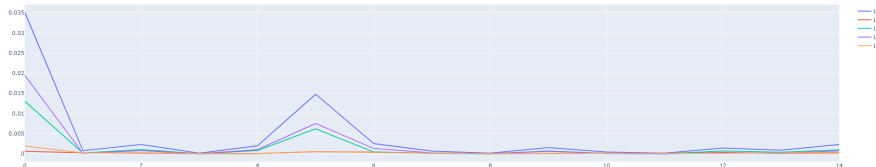
$$L_3(\theta) = \sum_{i=1}^{n_3} (NN(X, 0) - J_0(\lambda_4 \sqrt{x^2 + y^2}))^2$$

$$X = (x, y) \sim U(\partial\Omega)$$

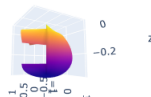
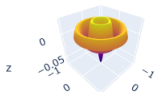
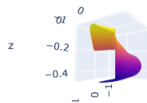
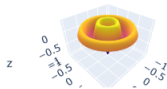
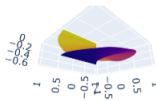
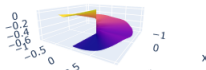
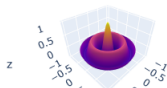
$$L_4(\theta) = \sum_{i=1}^{n_4} (NN_t(X, 0))^2$$

$$X = (x, y) \sim U(\partial\Omega)$$

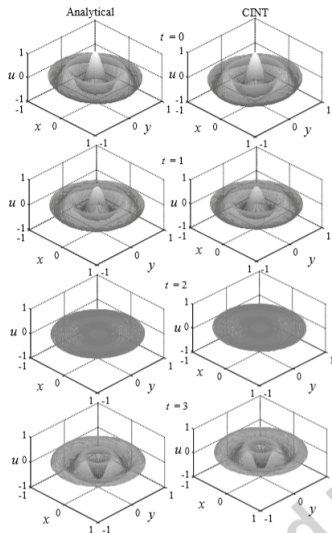
Pérdida por épocas



Solución analítica - Red Neuronal



Resultados de los autores



Simula la dinámica de un mercado financiero que contiene instrumentos financieros derivados. El modelo se ha convertido en el estándar de facto para estimar el precio de las opciones sobre acciones. La idea clave detrás del modelo es cubrir las opciones en una cartera de inversiones comprando y vendiendo el activo subyacente (como una acción) de la manera correcta y, como consecuencia, eliminar el riesgo.

Black-Scholes-Barenblatt 100D

$$dX_t = \sigma \text{diag}(X_t) dW_t \quad t \in [0, T]$$

$$X_0 = \xi$$

$$dY_t = r(Y_t - Z_t' X_t) dt + \sigma Z_t' \text{diag}(X_t) dW_t \quad t \in [0, T]$$

$$Y_T = g(X_T)$$

A partir de una ecuación diferencial estocástica forward-backward que tiene como principal característica que la condición terminal es aleatoria.

Donde W_t es un vector de evaluación de un movimiento Browniano, y $X_t, Y_t = u(t, X_t), Z_t = Du(t, X_t)$ procesos estocásticos asociados. Además, donde $T = 1, \sigma = 0,4, r = 0,05, \xi = (1, 0,5, 1, 0,5, \dots, 1, 0,5) \in \mathbb{R}^{100}$, y $g(x) = \|x\|^2$.

La cual bajo condiciones de regularidad adecuadas puede ser descrita como una función determinista del tiempo y estado del proceso, la llamada Black-ScholesBarenblatt equation:

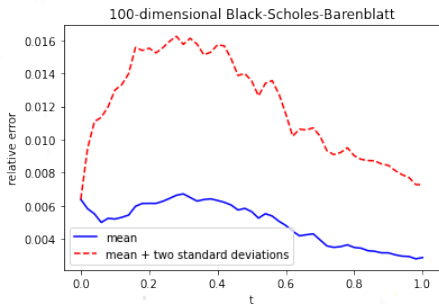
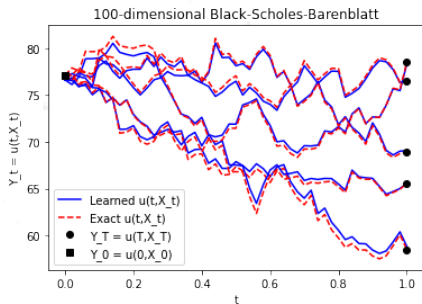
$$u_t = -\frac{1}{2} \text{Tr} (\sigma^2 \text{diag} (X_t^2) D^2 u) + r (u - (Du)^T x)$$

que admite como solución:

$$u(t, x) = \exp ((r + \sigma^2) (T - t)) g(x)$$





Resultados

Con una red neuronal de 5 capas y 256 neuronas, que se ajusta con una loss definida sobre una aproximación tipo Euler-Maruyama de los procesos estocásticos en la EDP estocástica.



En el presente trabajo se introdujo la resolución de ecuaciones diferenciales parciales a través de metodologías de deep learning, presentando una revisión del método y aplicando el mismo a diferentes contextos exponiendo ventajas y debilidades del mismo.

- DGM es un método altamente flexible y eficiente. Pero requiere de una serie de adaptaciones para aproximar la solución. Cómo modificar la base a una geometría adecuada al problema.
- Mejores aproximaciones pueden depender del dominio específico donde se formula la EDP. Incorporar información de la formulación variacional particular para cada ecuación diferencial.

-  Sirignano, Justin y Konstantinos Spiliopoulos (2018). “DGM: A deep learning algorithm for solving partial differential equations”. En: *Journal of Computational Physics* 375, págs. 1339-1364.
-  Kawaguchi, Kenji, Jiaoyang Huang y Leslie Pack Kaelbling (dic. de 2019). “Every Local Minimum Value Is the Global Minimum Value of Induced Model in Nonconvex Machine Learning”. En: *Neural Computation* 31.12, págs. 2293-2323.
-  Gühring, Ingo, Mones Raslan y Gitta Kutyniok (2020). “Expressivity of Deep Neural Networks”. En: *ArXiv* abs/2007.04759.
-  Willard, Jared y col. (2020). “Integrating Physics-Based Modeling with Machine Learning: A Survey”. En: *ArXiv* abs/2003.04919.



Lagaris, Isaac, Aristidis Likas y Dimitrios Fotiadis (sep. de 1998). "Artificial neural networks for solving ordinary and partial differential equations". En: *IEEE Transactions on Neural Networks* 9, págs. 987-1000. DOI: 10.1109/72.712178.



Arsenault, Louis-Francois y col. (ago. de 2014). "Machine learning for Many-Body Physics: the case of the Anderson impurity model". En: *Physical Review B* 90. DOI: 10.1103/PhysRevB.90.155136.



Rudd, Keith y Silvia Ferrari (dic. de 2014). "A Constrained Integration (CINT) Approach to Solving Partial Differential Equations using Artificial Neural Networks". En: *Neurocomputing* 19. DOI: 10.1016/j.neucom.2014.11.058.



Carleo, Giuseppe y Matthias Troyer (feb. de 2017). "Solving the Quantum Many-Body Problem with Artificial Neural Networks". En: *Science* 355, pág. 602. DOI: 10.1126/science.aag2302.



Khoo, Yuehaw, Jianfeng Lu y Lexing Ying (oct. de 2018). "Solving for high dimensional committor functions using artificial neural networks". En: *Research in the Mathematical Sciences* 6. DOI: 10.1007/s40687-018-0160-2.



Al-Aradi, A. y col. (2018). "Solving Nonlinear and High-Dimensional Partial Differential Equations via Deep Learning". En: *arXiv: Computational Finance*.



Raissi, Maziar (abr. de 2018). "Forward-Backward Stochastic Neural Networks: Deep Learning of High-dimensional Partial Differential Equations". En.

Gracias!

Preguntas?