# Using Artificial Neural Networks To Solve PDEs in regions with non-trivial boundaries.

Santiago Morales
Tutor: Pedro González Rodríguez

Universidad Carlos III de Madrid

September 30, 2022



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Content


Introduction
- Motivation
- Theoretical Framework
  - Artificial Neural Networks (ANN)
  - PINNs method using an ansatz

Implementation
- Tools Used
- Algorithms
- Description of the Methodology

Results and Analysis
- Advection 1D
- Diffusion 1D
- Advection 2D
- Diffusion 2D

Conclusions and Contributions

# Introduction: Motivation

- In real world applications, it is very common to find PDEs that can not be solved analytically, and thus they must be solved numerically.
- Typically, numerical methods are based in a discretization of the domain, that is, the domain is covered with a grid, and the method approximates the solution over the grid's nodes.



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Introduction: Technique used

- Reduce the dimension of the solution space by parametrizing the solution.
  - Solution will be defined by a certain amount of parameters.
- The solution will have the form of an Artificial Neural Networks (ANN).
  - Solution is defined by the parameters of the ANN $(b^*, \vec{w}^*)$

This technique where a Neural Network is used to solve problems involving PDEs is called Physics-Informed Neural Networks (PINNs)



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Theoretical Framework: Artificial Neural Networks (ANN)

- Were originally designed to replicate the function of our brain and how the brain learns from experience.
- Became very popular because it performed better than the models of the time for supervised tasks.



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Theoretical Framework: Artificial Neural Networks (ANN)



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

How they work?

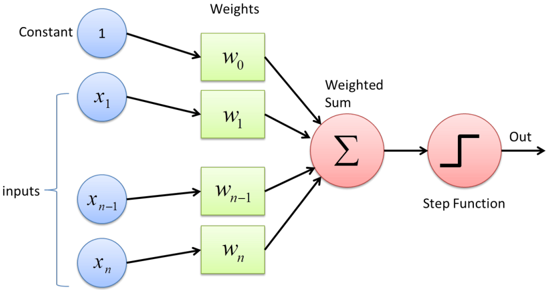



Figure: Graphical description of a perceptron. Image taken from: *What the hell is a perceptron.* [7]

# Theoretical Framework: PINNs method using an ansatz

Form of the problems that we are going to solve:

$$Lu = f \quad x \in \Omega \tag{1}$$

$$Bu = g \quad x \in \Gamma \subset \partial\Omega \tag{2}$$

where $L$ is a differential operator, $B$ the boundary operator, $f$ is a forcing function, $g$ the boundary data, $\Omega \subset \mathbb{R}^n$ the domain of interest and $\Gamma$ is the part of its boundary $(\partial\Omega)$ where the boundary conditions are imposed.


Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Theoretical Framework: PINNs method using an ansatz

Rewrite the problem as an optimization problem:
Find the value of $u$ that minimize:

$$|Lu - f| + |Bu - g|$$

To simplify the problem, we used the following ansatz [1]:

$$\hat{u} = G(x) + D(x)y^L(x; \vec{w}, b) \tag{3}$$

where $D(x)$ is a smooth extension of the distance function $d(x) = \min_{x_b \in \Gamma} ||x - x_b||$ and G(x) is a smooth extension for the boundary function



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Theoretical Framework: PINNs method using an ansatz

Calculation of $G(x), D(x)$ and $y^L(x; w, b)$:

- To calculate, $G(x)$ and $D(x)$ we used a small ANNs that use $g(x)$ and $d(x)$[1] as the labels for the supervised learning task.

[1] $d(x) = \min_{x_b \in \Gamma} ||x - x_b||$

# Theoretical Framework: PINNs method using an ansatz

Calculation of $G(x), D(x)$ and $y^L(x; w, b)$:

- To calculate $y^L(x; w, b)$ we used a larger ANN that was trained to minimize $||\vec{c}||_2$, such that:

$$c_i = |L\hat{u}(\vec{x}_i) - f(\vec{x}_i)|$$

- The boundary data is included in the ansatz function.




Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Implementation: Tools Used

- Python (version 3.7.13) [8]
- TensorFlow (version 2.8.2) [6]: Package for building, training and deploying deep learning models in Python.
- Keras (version 2.8.0) [2]: TensorFlow subpackage for simplifying the managing of standard deep learning models.
- Numpy (version 1.21.6) [4]: Package for calculations using multidimensional arrays (tensors) in Python.
- Matplotlib (version 3.2.2) [5]: Package for create visualizations
- Shapely (version 1.8.4) [3]: Package for manipulation and analysis of geometric objects in the Cartesian plane.

This running in a Linux-based virtual machine provided by Google for free.



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Implementation: Algorithms

We will use the Automatic Differentiation algorithm (AutoDiff) to calculate the derivatives inside the training loop.
One of the more important cases of use of the AutoDiff algorithm is Backpropagation. Backpropagation, is a well establish state-of-the-art algorithm to train deep learning models



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Implementation: Description of the Methodology

1. Define the domain in which the problem is going to be solved.
2. Select the N collocation points that will be used to solve the PINN.
3. Train both ANNs for $D$ and $G$.
4. Train the ANN for $y^L(x; w, b)$



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Description of the Methodology: Smooth Distance Function

Python function used to define distance (and boundary) model:

```
def build_model_distance(NUM_PERCEPTRONS):
  model_distance=tf.keras.Sequential([
    tf.keras.layers.Dense(NUM_PERCEPTRONS, activation='sigmoid',
                          input_shape=[None, 1], dtype='float64'
                          ),
    tf.keras.layers.Dense(NUM_PERCEPTRONS,
                          activation='sigmoid'
                          ),
    tf.keras.layers.Dense(1,
                          dtype='float64'
                          )
  ])

  optimizer = Adam(learning_rate=0.01, beta_1=0.9, beta_2=0.99)

  model_distance.compile(loss=custom_loss,
              optimizer=optimizer,
              metrics=['mae', 'mse'])
  return model_distance
```

1D: NUM_PERCEPTRONS = 10
2D: NUM_PERCEPTRONS = 20

# Description of the Methodology: Advection Problem

Rewriting the advection problem using the ansatz:

$$L\hat{u} = \frac{d\hat{u}}{dx} = f(x)$$

$\downarrow$ Using the ansatz

$$L\hat{u} = \frac{dG(x)}{dx} + D(x)\frac{dy^L}{dx} + y^L\frac{dD(x)}{dx} = f(x)$$

Optimization problem becomes to reduce $C$ such that:

$$C = \sum_{x \in \text{collocation points}} (L\hat{u} - f(x))^2$$



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Description of the Methodology: Advection Problem 2D

For the 2D advection problem, we can write it as:

$$L\hat{u} = a\frac{\partial \hat{u}}{\partial x_1} + b\frac{\partial \hat{u}}{\partial x_2} = f$$

$$\downarrow \text{ Using the ansatz}$$

$$\begin{aligned} L\hat{u} = a\left(\frac{\partial G}{\partial x_1} + D\frac{\partial y^L}{\partial x_1} + y^L\frac{\partial D}{\partial x_1}\right) \\ + b\left(\frac{\partial G}{\partial x_2} + D\frac{\partial y^L}{\partial x_2} + y^L\frac{\partial D}{\partial x_2}\right) = f \end{aligned}$$




Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Description of the Methodology: Advection Problem

Using the `GradientTape` method from TensorFlow, it is possible to calculate the derivative of $D(x)$ as follows:

```
x_variable = tf.Variable(x_train, dtype='float64')
with tf.GradientTape() as g:
    g.watch(x_variable)
    Dx = D(x_variable)
dD_dx = g.gradient(Dx, x_variable)
D_pred = D(x_variable)
```

And, replacing $D$ by $G$ we calculated the derivative of $G$.



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Description of the Methodology: Advection Problem

Using the former results, and including it into the training cycle to calculate the derivative of $y^L$ we trained the PINN as follows:

```
opt = Adam(learning_rate=0.01, beta_1=0.9, beta_2=0.99)
ff = tf.constant(f(x_variable))
n_train_steps = 10000

for step in range(n_train_steps):
    # we need to convert x to a variable if we want the tape to be
    # able to compute the gradient according to x
    with tf.GradientTape() as model_tape:
        with tf.GradientTape() as y_tape:
            y_tape.watch(x_variable)
            y_pred = model(x_variable)
        dy_dx = y_tape.gradient(y_pred, x_variable)
        y_pred_old = y_pred.numpy()
        lu = dD_dx.numpy()*y_pred + D_pred.numpy()*dy_dx
        loss = tf.reduce_sum(tf.math.squared_difference(lu, ff))
    grad = model_tape.gradient(loss, model.trainable_variables)
    opt.apply_gradients(zip(grad, model.trainable_variables))
```



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Description of the Methodology: Diffusion Problem

Rewriting the diffusion problem using the ansatz:

$$L\hat{u} = \frac{d^2\hat{u}}{dx^2} = f(x)$$

$$\downarrow \text{ Using the ansatz}$$

$$L\hat{u} = \frac{d^2D(x)}{dx^2}y^L + 2\frac{dD(x)}{dx}\frac{dy}{dx} + D(x)\frac{d^2y}{dx^2} = f(x)$$

The only difference with the advection problem were the second order derivatives terms.



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Description of the Methodology: Diffusion Problem 2D

Therefore, the diffusion problem in 2D could be rewritten as:

$$Lu = \frac{\partial^2 u}{\partial x_1^2} + \frac{\partial^2 u}{\partial x_2^2} = f$$

$$\downarrow \text{ Using the ansatz}$$

$$\begin{aligned} L\hat{u} = {} & \frac{\partial^2 D}{\partial x_1^2} y^L + 2 \frac{\partial D}{\partial x_1} \frac{\partial y^L}{\partial x_1} + D \frac{\partial^2 y^L}{\partial x_1^2} \\ & + \frac{\partial^2 D}{\partial x_2^2} y^L + 2 \frac{\partial D}{\partial x_2} \frac{\partial y^L}{\partial x_2} + D \frac{\partial^2 y^L}{\partial x_2^2} = f \end{aligned}$$



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Advection 1D:

Problem:

$$\frac{du}{dx} = 2\pi(\cos(2\pi x)\cos(4\pi x) - 2\sin(2\pi x)\sin(4\pi x))$$
$$g_0 = 1$$

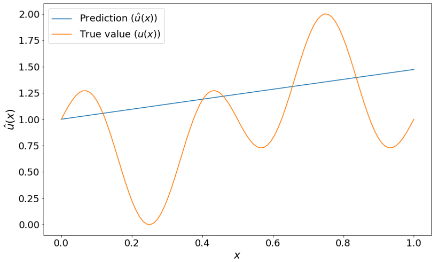
The real solution is given by:

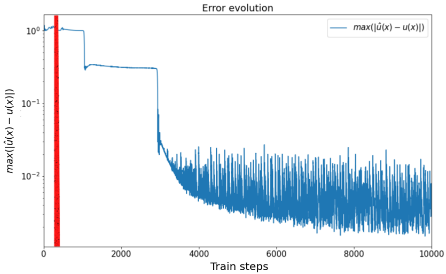
$$u(x) = sin(2\pi x)cos(4\pi x) + 1$$



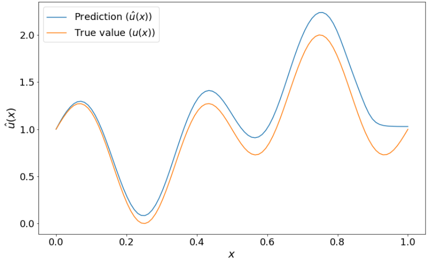
Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

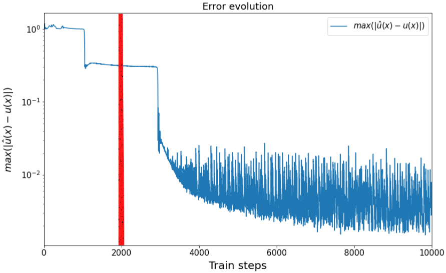
# Advection 1D: train steps = 0



# Advection 1D: train steps = 2000

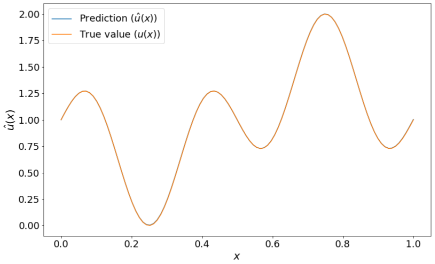


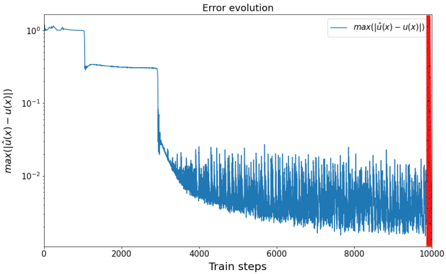






Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Advection 1D: train steps $= 10000$



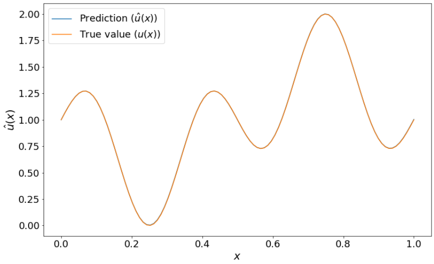



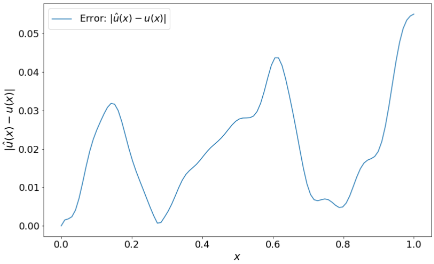




Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Advection 1D: Result






Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Diffusion 1D:

Problem:

$$\frac{d^2u}{dx^2} = -\frac{1}{4}\pi^2\left(17cos(2\pi x)sin\left(\frac{\pi x}{2}\right) + 8cos\left(\frac{\pi x}{2}\right)sin(2\pi x)\right)$$

$$g_0 = 1$$

$$g_1 = 2$$

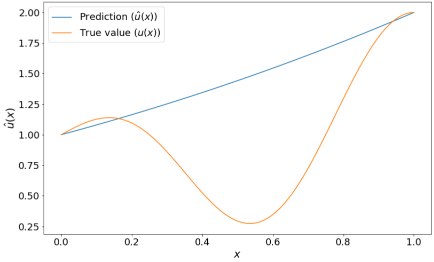
The real solution is given by:

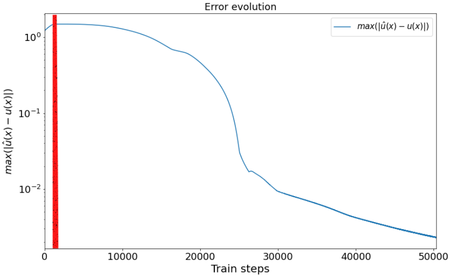
$$u(x) = sin\left(\frac{\pi x}{2}\right)cos(2\pi x) + 1$$



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

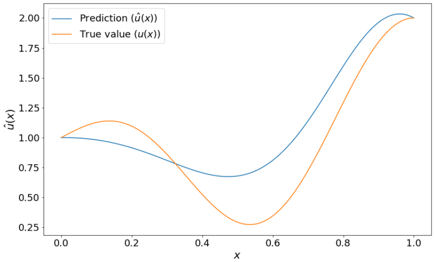
# Diffusion 1D: train steps $= 0$

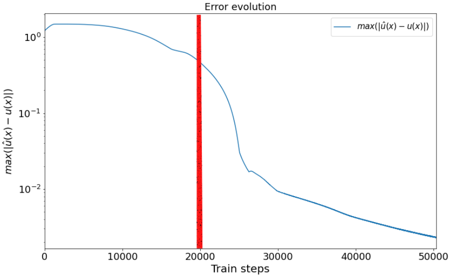



PDE solving using PINNs
Santiago Morales
Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Diffusion 1D: train steps = 20000



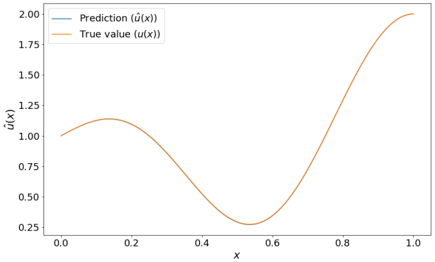



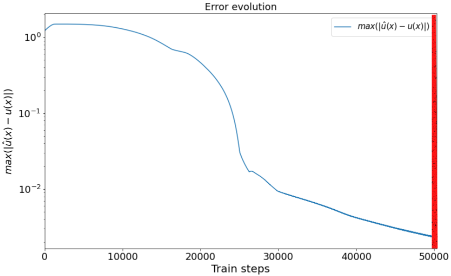




Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Diffusion 1D: train steps $= 50000$



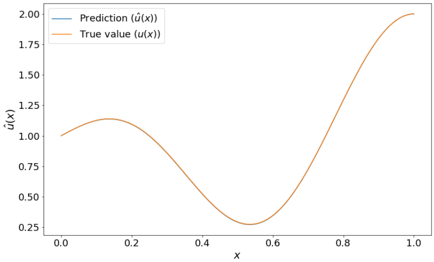



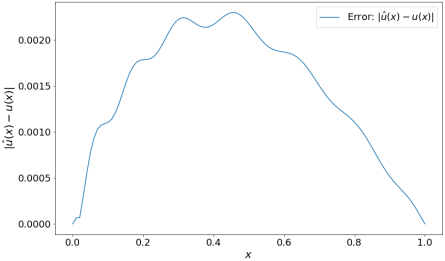




Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Diffusion 1D: Result






Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Advection 2D:

Problem:

$$\frac{\partial u}{\partial x} + \frac{1}{2}\frac{\partial u}{\partial y} = \frac{\pi}{2}\left(-\sin(\pi x)\sin(\pi y) + \frac{1}{2}\cos(\pi x)\cos(\pi y)\right)$$

The real solution is given by:

$$u(x,y) = \frac{1}{2}\cos(\pi x)\sin(\pi y)$$

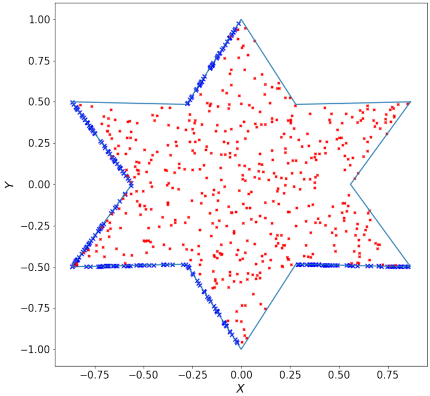


Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Results and Analysis: Advection 2D

The points used to solve the Advection problem in 2D must follow:

$$\Gamma = \{x \in \partial\Omega : (a, b) \cdot n(x) < 0\}.$$





PDE solving using PINNs

Santiago Morales

Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz

Implementation
Tools Used
Algorithms
Description of the Methodology

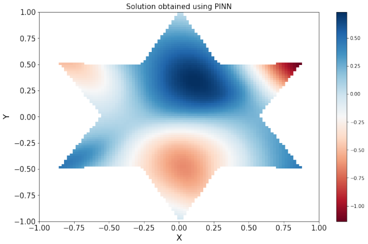
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D

Conclusions and Contributions

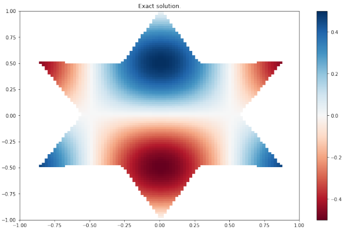
References

# Advection 2D:

## PINN Result:

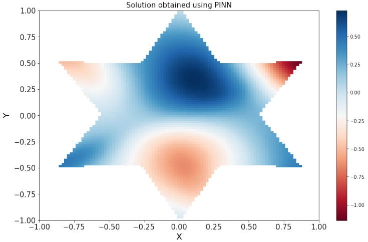



## Real Solution:

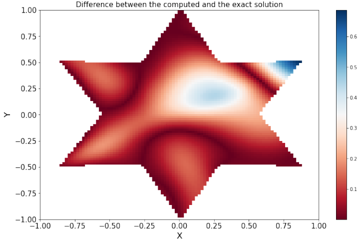


# Advection 2D: Error

PINN Result:




Solution Error:



# Diffusion 2D:

Problem:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \frac{\pi}{2}\left(-\sin(\pi x)\sin(\pi y) + \frac{1}{2}\cos(\pi x)\cos(\pi y)\right)$$

The real solution is given by:

$$u(x,y) = \exp\left(-(2x^2 + 4y^2)\right) + 1/2$$

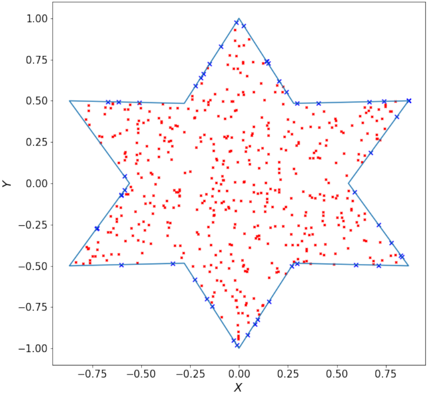


Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Diffusion 2D:

The points used to solve the Diffusion problem in 2D must follow:

$$\Gamma = \{x \in \partial\Omega\}.$$





PDE solving using PINNs

Santiago Morales

Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz

Implementation
Tools Used
Algorithms
Description of the Methodology

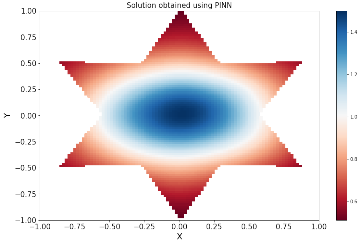
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D

Conclusions and Contributions

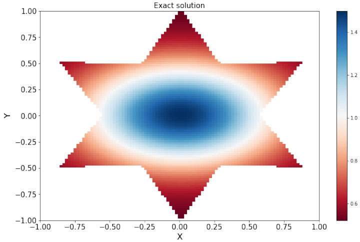
References

# Diffusion 2D: Result

## PINN Result:




## Real Solution:

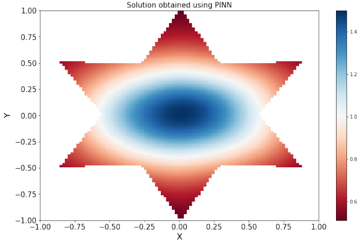




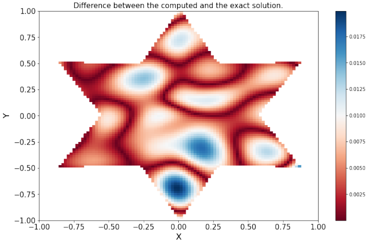
PDE solving using PINNs
Santiago Morales
Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Diffusion 2D: Error

## PINN Result:




## Solution Error:



# Conclusions and Contributions

Contribution:

- Programming PINNs to solve ODEs and PDEs using TensorFlow.

Conclusions:

- The only parameter that is important for scaling is the complexity of the ANN used to find $y^L$, the extra code used for solving the 2D problems is for defining and generating the geometry and the collocation points.
- The results obtained for the 1D problem and for the diffusion problem in 2D are good approximations to the real solution. The result presented for Advection in 2D, on the other hand, is a poor approximation to the solution.



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# References


[1] Jens Berg and Kaj Nyström. "A unified deep artificial neural network approach to partial differential equations in complex geometries". In: *Neurocomputing* 317 (2018), pp. 28–41. DOI: 10.1016/j.neucom.2018.06.056.

[2] Francois Chollet et al. *Keras*. 2015. URL: https://github.com/fchollet/keras.

[3] Sean Gillies et al. *Shapely: manipulation and analysis of geometric objects*. 2007–. URL: https://github.com/shapely/shapely.

[4] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[5] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: 10.5281/zenodo.4030140.

[6] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: https://www.tensorflow.org/.

[7] SAGAR SHARMA. *What the Hell is Perceptron?* 2022. URL: https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53.

[8] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN: 1441412697.





Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

Thank you!



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Algorithms: AutoDiff algorithm

## Forward accumulation:

Forward Primal Trace

--------------------------

$$y_0 = x_1 = 1$$
$$y_1 = x_2 = 2$$
$$y_2 = \exp\{y_0\} = \exp\{1\}$$
$$y_3 = \cos y_1 = \cos 2$$
$$y_4 = y_0 y_1 = 1 \times 2$$
$$y_5 = y_2 + y_3 = 2.718 - 0.416$$
$$y_6 = y_5 + y_4 = 2.302 + 2$$

--------------------------

$$F(x_1, x_2) = y_6 = 4.302$$

Forward Derivative Trace

--------------------------

$$\dot{y}_0 = \partial x_1 / \partial x_1 = 1$$
$$\dot{y}_1 = \partial x_2 / \partial x_1 = 0$$
$$\dot{y}_2 = \dot{y}_0 \exp\{y_0\} = 1 \times \exp\{1\}$$
$$= 2.718$$
$$\dot{y}_3 = -\dot{y}_1 \sin y_1 = 0 \times \sin 0 = 0$$
$$\dot{y}_4 = y_0 \dot{y}_1 + \dot{y}_0 y_1 = 1 \times 0 + 1 \times 2$$
$$= 2$$
$$\dot{y}_5 = \dot{y}_2 + \dot{y}_3 = 2.718 + 0$$
$$\dot{y}_6 = \dot{y}_5 + \dot{y}_4 = 2.718 + 2$$

--------------------------

$$\partial F(x_1, x_2)/\partial x_1 = \dot{y}_6 = 4.718$$



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

# Algorithms: AutoDiff algorithm

## Reverse accumulation:

Forward Primal Trace

---

$$y_0 = x_1 = 1$$
$$y_1 = x_2 = 2$$
$$y_2 = \exp\{y_0\} = \exp\{1\}$$
$$y_3 = \cos y_1 = \cos 2$$
$$y_4 = y_0 y_1 = 1 \times 2$$
$$y_5 = y_2 + y_3 = 2.718 - 0.416$$
$$y_6 = y_5 + y_4 = 2.302 + 2$$

---

$$F(x_1, x_2) = y_6 = 4.302$$

Reverse Derivative Trace

---

$$\bar{y_6} = \bar{F}(x_1, x_2) = \partial F / \partial F = 1$$

---

$$\bar{y_5} = \bar{y_6} \partial y_6 / \partial y_5 = 1 \times 1 = 1$$
$$\bar{y_4} = \bar{y_6} \partial y_6 / \partial y_4 = 1 \times 1 = 1$$
$$\bar{y_3} = \bar{y_5} \partial y_5 / \partial y_3 = 1 \times 1 = 1$$
$$\bar{y_2} = \bar{y_5} \partial y_5 / \partial y_2 = 1 \times 1 = 1$$
$$\bar{y_1} = \bar{y_4} \partial y_4 / \partial y_1 = 1 \times y_0 = 1$$
$$\bar{y_0} = \bar{y_4} \partial y_4 / \partial y_0 = 1 \times y_0 = 2$$
$$\bar{y_1} = \bar{y_1} + \bar{y_3} \partial y_3 / \partial y_1$$
$$= 1 - 1 \times \sin y_1 = 0.091$$
$$\bar{y_0} = \bar{y_0} + \bar{y_2} \partial y_2 / \partial y_0$$
$$= 2 + 1 \times \exp\{y_0\} = 4.718$$

---

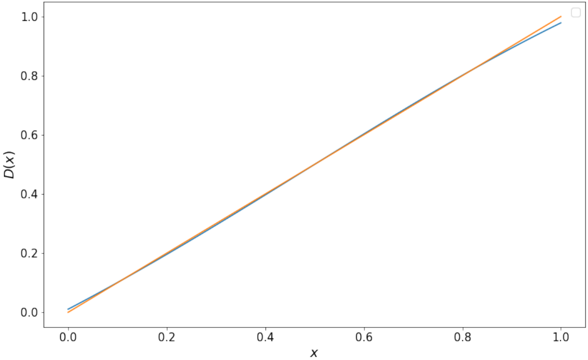
$$\partial F / \partial x_1 = \bar{x_1} = \bar{y_0} = 4.718$$
$$\partial F / \partial x_2 = \bar{x_2} = \bar{y_1} = 0.091$$



Introduction
Motivation
Theoretical Framework
Artificial Neural Networks (ANN)
PINNs method using an ansatz
Implementation
Tools Used
Algorithms
Description of the Methodology
Results and Analysis
Advection 1D
Diffusion 1D
Advection 2D
Diffusion 2D
Conclusions and Contributions
References

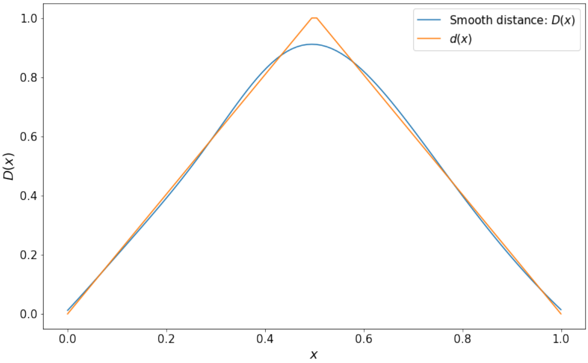
# Distance function Advection 1D

$D(x)$:



# Distance function Diffusion 1D

$D(x)$:







Introduction

Motivation

Theoretical Framework

Artificial Neural Networks (ANN)

PINNs method using an ansatz

Implementation

Tools Used

Algorithms

Description of the Methodology

Results and Analysis

Advection 1D

Diffusion 1D

Advection 2D

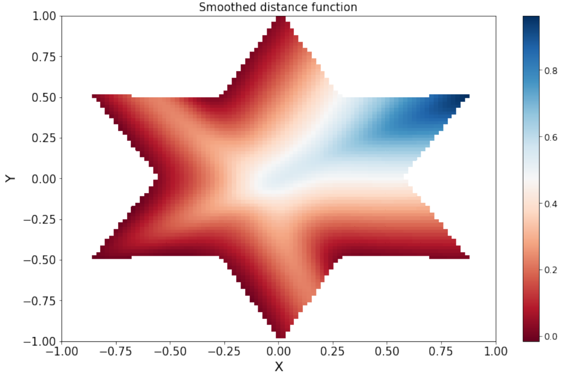
Diffusion 2D

Conclusions and Contributions

References

# Distance function Advection 2D

$D(x)$:



# Distance function Diffusion 2D

$D(x)$: