FUNDAÇÃO GETULIO VARGAS
ESCOLA de PÓS-GRADUAÇÃO em ECONOMIA

João Pedro Catão

Innovation race with continuous effort choice: a deep-learning approach

Rio de Janeiro
2024

João Pedro Catão

Innovation race with continuous effort choice: a deep-learning approach

Dissertação para obtenção do grau de mestre apresentada à Escola de Pós-Graduação em Economia

Área de concentração: Teoria Econômica

Orientador: Felipe Saraiva Iachan

Rio de Janeiro
2024

Dados Internacionais de Catalogação na Publicação (CIP)
Ficha catalográfica elaborada pelo Sistema de Bibliotecas/FGV

Silva, João Pedro Catão Queiroz da
Innovation race with continuous effort choice: a deep-learning approach / João Pedro Catão Queiroz da Silva. – 2024.

19 f.

Dissertação (mestrado) – Escola Brasileira de Economia e Finanças.
Orientador: Felipe Saraiva Iachan.
Inclui bibliografia.

1. Aprendizado do computador. 2. Inteligência artificial. 3. Gestão estratégica. 4. Redes neurais (Computação). I. Iachan, Felipe Saraiva. III. Fundação Getulio Vargas. Escola Brasileira de Economia e Finanças. IV. Título.

CDD - 343.09998

Elaborada por Marcelle Costal de Castro dos Santos – CRB-7-RJ-007517/O

FUNDAÇÃO GETULIO VARGAS
MESTRADO EM ECONOMIA
EPGE ESCOLA BRASILEIRA DE ECONOMIA E FINANÇAS - FGV EPGE

JOÃO PEDRO CATÃO QUEIROZ DA SILVA

"INNOVATION RACE WITH CONTINUOUS EFFORT CHOICE: A DEEP-LEARNING APPROACH''.

DISSERTAÇÃO APRESENTADO/A AO CURSO DE MESTRADO EM ECONOMIA PARA OBTENÇÃO DO GRAU DE MESTRE (A) EM ECONOMIA.

ESTE É UM TRABALHO ORIGINAL ONDE FOI VERIFICADA A NÃO EXISTÊNCIA DE PLÁGIO E DE UTILIZAÇÃO DE INTELIGÊNCIA ARTIFICIAL, NÃO EXPLICITADA, NO CORPO DO TRABALHO.

DATA DA DEFESA: 23/04/2024

ASSINATURA DOS MEMBROS DA BANCA EXAMINADORA

PRESIDENTE DA COMISSÃO EXAMINADORA: PROFº/ª FELIPE SARAIVA IACHAN

<ASSINADO ELETRONICAMENTE>
PROFº/ª FELIPE SARAIVA IACHAN
ORIENTADOR

<ASSINADO ELETRONICAMENTE>
PROFº/ª RAFAEL COUTINHO COSTA LIMA
MEMBRO

<ASSINADO ELETRONICAMENTE>
PROFº/ª LEANDRO GORNO
MEMBRO

RIO DE JANEIRO, 23 DE ABRIL DE 2024.

<ASSINADO ELETRONICAMENTE>
PROFº LUCAS JÓVER MAESTRI
COORDENADOR

<ASSINADO ELETRONICAMENTE>
PROFº ANTONIO DE ARAUJO FREITAS JUNIOR
PRÓ-REITOR DE ENSINO, PESQUISA E PÓS-GRADUAÇÃO

D4Sign 2e570690-fa9b-4991-96b6-1febc44c434e - Para confirmar as assinaturas acesse https://secure.d4sign.com.br/verificar
Documento assinado eletronicamente, conforme MP 2.200-2/01, Art. 10º, §2.

Resumo

Nós estudamos uma corrida de inovação, usando tempo contínuo. Cada firma observa seu próprio estado e os dos oponentes e decide continuamente o seu nível de esforço. Como as recompensas são baseadas em rankings relativos, nós observamos um grande nível de esforço quando as firmas estão perto da linha de chegada ao mesmo tempo. Por outro lado, nós observamos um efeito desencorajador quando um competidor lidera por uma grande margem. Além disso, para ampliarmos a análise para um cenário com mais firmas, nós introduzimos uma solução com redes neurais informadas pela física (PINNs) devido os seus benefícios em relação aos métodos tradicionais. Por fim, nós comparamos a solução para um jogo com duas firmas com a solução usando métodos finitos.

Palavras-chave: Corrida de inovação, tempo contínuo, esforço ótimo, redes neurais informadas pela física.

Abstract

We examine an innovation race using a continuous-time approach. Each firm observes its own and its opponents' states and, continuously decides the optimal level of effort. Since rewards are based on relative rank, we observe a high level of effort in states where players are jointly close to the goal. On the other hand, we observe a *discouragement effect* when a competitor holds a significant lead. Furthermore, to enhance the analysis's scalability to more firms, we employ a physics-informed neural networks (PINNs) approach, recognizing its potential benefits over conventional techniques. Finally, we compare the outcomes of our two-firm game with a benchmark derived from the finite element method (FEM).

Keywords: Innovation race, continuous-time, optimal effort, physics-informed neural networks.

List of Figures

1 Boundary and collocation points 5
2 Neural Network Architecture 8
3 Evolution of Player 1's loss 10
4 Value Function of Player 1 11
5 Value Function of Player 2 11
6 Absolute difference between Finite Element Method (FEM) and Physics-Informed Neural Network (PINN) results 12
7 Policy Function of Player 1 13
8 Policy Function of Player 2 14
9 Simulated at $t = 100$ 15
10 Simulated at $t = 500$ 15
11 Simulated at $t = 1000$ 16
12 Expected Discounted Effort 17

List of Tables

1 Parameter Values 11
2 Main statistics of the absolute difference between the results of FEM and PINN for Player 1's value function 13
3 PINN's and FEM's comparison 14

Contents

1 Introduction **1**

2 Model **2**

3 Methodology **3**

3.1 Physics-Informed Neural Network (PINN) 4

3.2 SoftAdapt 6

3.3 Architecture 7

3.4 Training Loop 8

4 Results **10**

4.1 Value Functions 10

4.2 Policy Functions 12

4.3 Robustness checks 13

4.3.1 Probability Density Function (PDF): 14

4.3.2 Expected Discounted Effort 16

5 Conclusion **17**

1 Introduction

A race is understood as a competition in which participants strive to reach a predetermined finish line to secure victory. In an innovation race, success is contingent upon being the first firm (or institution) to develop a product of a specific quality. Furthermore, in such a scenario, earnings depend to a great extent, on the *relative* rank of contestants rather than the *absolute* distance between them. Consequently, the margin of victory does not significantly influence the resulting payoffs, fostering heightened effort levels when participants are, at the same time, close to the ultimate goal.

In this paper, we aim to analyze how a principal could optimally design such a race and how the number of participants influences its dynamics. In our model, the state (product's quality) follows a Brownian motion with a drift that depends on players' effort, and the flow cost of effort functions is strictly convex. In contrast to (Gorno & Iachan, 2020), where states are private information and firms must decide the optimal time to launch their product, here all states are public information, and firms continuously choose the level of effort to achieve the established quality. Additionally, the rank-order structure of the prize plays a crucial role. We observe the same *discouragement effect* as shown in (Cao, 2014). Firms exert high effort only when they are close to the goal at the same time. If a firm observes that its rival is almost completing the objective, it reduces its level of effort. The rival, due to less competition, can safely reduce his effort as well.

In our game, the number of state variables equals the number of firms, and traditional methods like finite elements (FEM) become computationally unfeasible, in terms of both computing time and memory, as the number of state variables grows. Therefore, we have chosen to utilize a Physics-Informed Neural Network (PINN) approach to overcome this "curse of dimensionality". Despite being a promising method, implementing a PINN approach posed some challenges in our work due to the rank-order structure of the prize. To satisfy the system's physics laws, we used a continuous activation function which was incapable of capturing the discontinuity presented in the prize structure. This finding is consistent with (Sharma et al., 2023), who studied a 2D heat conduction problem with discontinuous solutions at corners. Nevertheless, this issue will not pose a significant problem to the race's dynamics, as elaborated in the following paragraph. Our contribution is twofold: first, we present economic insights derived from a two-firm environment, and also, we apply deep-learning techniques to solve the game.

The race exhibits discontinuous solutions at the edges and corners, posing challenges for neural networks with continuous activation functions to learn effectively. Nonetheless, the areas near the discontinuity are associated with high efforts, so it's plausible that these regions are infrequently visited or traversed rapidly by the players. To gain deeper insights into the accuracy of our findings, we conduct two robustness checks. First, we use the policy functions derived from both PINN's and FEM's results to simulate 10,000 races and create a notion of a probability density function (PDF) in our domain. We then compare these PDFs using a *sup-norm* criterion. Secondly, we calculate the expected discounted effort (EDE) using both methods and compare the results. We show that for a time when most games have finished and players must have

crossed the finish line, the areas near discontinuities do not significantly impact the outcomes. We find that the PDFs' *sup-norm* is 0.16 percentage point. Also, we note that the starting point for a principal aiming to maximize the expected discounted effort is similar in both methods.

Our work contributes to the rapidly growing literature that applies machine-learning methods to economics. (Duarte et al., 2024) demonstrates that the notable asset-pricing effects observed in the cases of two trees (Cochrane et al., 2008) diminish with the inclusion of additional trees, which suggests that the number of firms in our game might play an important role. (Fernández-Villaverde et al., 2023) demonstrates how neural networks can successfully approximate the non-linear laws of motion of the economy's states. Additionally, (Gopalakrishna, 2021) shows how one can solve a heterogeneous agent portfolio choice problem by approximating the Hamilton-Jacobi-Bellman equation using neural networks. In this paper, we employ a physics-informed neural network (PINN) approach to solve a continuous-time innovation race.

The rest of the paper is organized as follows. Section 2 describes the race and presents the best response for each player. Section 3 discusses in detail the physics-informed neural network method used to solve the game numerically. Section 4 shows our results and sets some robustness checks, and Section 5 concludes.

2 Model

Firms compete to develop a product with a predetermined quality, aiming to be the first to achieve this goal. The winner receives a prize equal to P, while the losers receive nothing. Time is continuous, and the horizon is infinite. Players discount future payoffs at a rate of $r > 0$. Each player observes the evolution of all positions $X_i(t)$, where $X_i \equiv \{X_i(t)\}$ is a stochastic process. The evolution of X_i is described by the following stochastic differential equation:

$$dX_i(t) = a_i\, dt + \sigma_i\, dB_i(t) + \eta\, dZ(t) \tag{1}$$

Here, B_i for $i = 1{,}2$ are standard independent Brownian motions, and Z is common for all competitors. Moreover, B_i and Z are also independent for each player. The drift a_i represents the level of effort continuously chosen by the players, restricted to be positive. The cost to exert effort is given by the function:

$$c_i(a_i) = \frac{k_i a_i^2}{2}$$

with $k_i > 0$.

It is important to note that actual increments, $dX_i(t)$, can be negative, representing a downgrade in technology, for instance. In our case, $\sigma_i^2 + \eta^2$, with both $\sigma_i > 0$ and $\eta > 0$, represents their volatility.

The value function for player i, $V_i(x_i, x_j)$, satisfies the following stationary Hamilton-Jacobi-Bellman (HJB) equation:

$$rV_i(x_i, x_j) = \max_{a_i} \left\{ -\frac{k_i a_i^2}{2} + a_i \frac{\partial V_i}{\partial x_i} + \frac{1}{2}(\sigma_i^2 + \eta^2)\frac{\partial^2 V_i}{\partial x_i^2} + a_j \frac{\partial V_i}{\partial x_j} + \frac{1}{2}(\sigma_j^2 + \eta^2)\frac{\partial^2 V_i}{\partial x_j^2} + \eta^2 \frac{\partial^2 V_i}{\partial x_i \partial x_j} \right\} \tag{2}$$

with the following Dirichlet boundary conditions:

$$V_i(x_i = W, x_j \leq W) = P, \tag{3}$$

$$V_i(x_i < W, x_j = W) = 0, \tag{4}$$

$$V_i(x_i = L, x_j > L) = 0, \tag{5}$$

$$V_i(x_i \geq L, x_j = L) = P. \tag{6}$$

In equation (2) above, the left-hand side term represents the return of the project. On the right-hand side, the first term represents the project's cash flow (dividend) and carries a negative sign because it represents the cost incurred to exert effort. Additionally, the terms within the maximization, associated with first-order derivatives, depict effects arising from the drift in the stochastic process $X_i(t)$, which is continuously chosen by the players. Finally, the terms related to the second-order derivatives illustrate the effects of volatility.

The optimal policy is derived from the following first-order condition:

$$-k_i a_i^* + \frac{\partial V_i}{\partial x_i} = 0 \tag{7}$$

This condition leads competitors to continuously choose an effort level denoted as $a_i^* = \frac{1}{k}\frac{\partial V_i}{\partial x_i}$ for every $x_i \in [L, W]^2$. Now, the question arises: for which states $x \in [L, W]^2$ do we observe an escalation in competition? In simpler terms, when do players optimally choose a high level of effort?

To explore these questions, we employ numerical methods to approximate the players' value functions. Departing from conventional techniques such as finite element methods (FEM), we opt for a physics-informed neural network methodology. In Section IV a comparison with a FEM benchmark is shown. In sequence, we present a discussion about the methodology employed in this article.

3 Methodology

The general two-firm game described in the previous section will be further analyzed in this section. Moreover, we suppose the state space $(x_1, x_2) \in [-1, 1]^2$ and $P = 1$. This new race implies the following system of stationary PDE:

$$
\begin{aligned}
rV_1(x_1,x_2) = \max_{a_1}\Big\{ &-\frac{k_1 a_1^2}{2} + a_1\frac{\partial V_1}{\partial x_1} + \frac{1}{2}(\sigma_1^2+\eta^2)\frac{\partial^2 V_1}{\partial x_1^2} \\
&+ a_2\frac{\partial V_1}{\partial x_2} + \frac{1}{2}(\sigma_2^2+\eta^2)\frac{\partial^2 V_1}{\partial x_2^2} + \eta^2\frac{\partial^2 V_1}{\partial x_1 \partial x_2}\Big\} \\
rV_2(x_1,x_2) = \max_{a_2}\Big\{ &-\frac{k_2 a_2^2}{2} + a_2\frac{\partial V_2}{\partial x_2} + \frac{1}{2}(\sigma_2^2+\eta^2)\frac{\partial^2 V_2}{\partial x_2^2} \\
&+ a_1\frac{\partial V_2}{\partial x_1} + \frac{1}{2}(\sigma_1^2+\eta^2)\frac{\partial^2 V_2}{\partial x_1^2} + \eta^2\frac{\partial^2 V_2}{\partial x_1 \partial x_2}\Big\}
\end{aligned} \tag{8}
$$

subject to the Dirichlet boundary conditions:

$$
\begin{aligned}
&V_1(x_1 = 1, x_2 \leq 1) = V_1(x_1 \geq -1, x_2 = -1) \\
&= V_2(x_1 \leq 1, x_2 = 1) = V_2(x_1 = -1, x_2 \geq -1) = 1 \\
&V_1(x_1 < 1, x_2 = 1) = V_1(x_1 = -1, x_2 > -1) \\
&= V_2(x_1 = 1, x_2 < 1) = V_2(x_1 > -1, x_2 = -1) = 0
\end{aligned} \tag{9}
$$

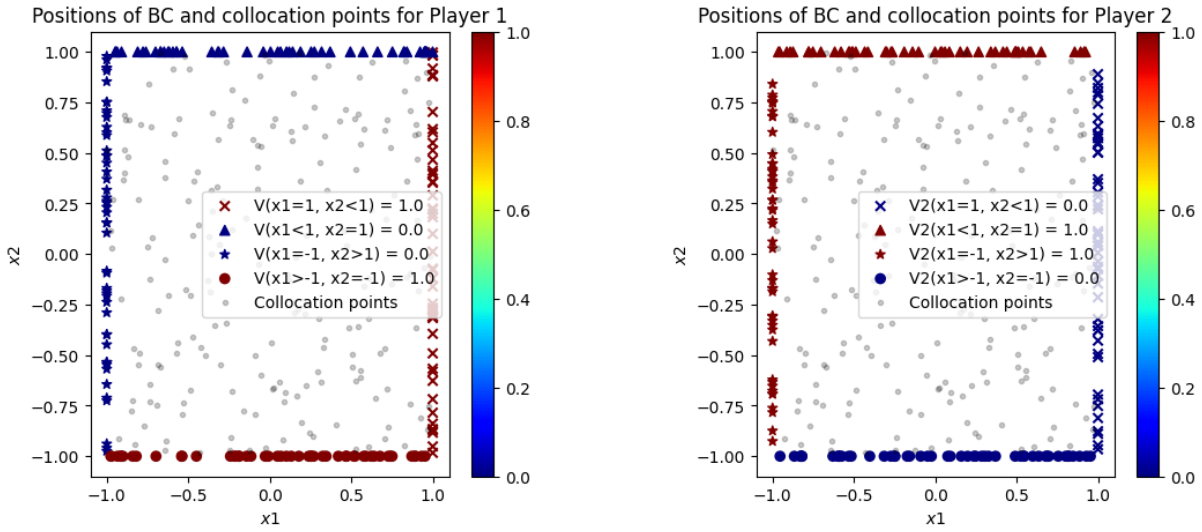
The main objective of this work is to solve the above system numerically. Duarte et al., 2024 shows that the noteworthy asset-pricing effects observed in the cases of two trees (Cochrane et al., 2008) diminish with the inclusion of additional trees. Consequently, our focus on a two-firm setting serves as an initial step, intending to extend the analysis to a multiple-firm scenario. The rationale behind employing a deep-learning approach, as opposed to traditional methods, stems from the challenge posed by the "curse of dimensionality". As discussed in Bellman, 1957 dynamic programming becomes more demanding in terms of computing time and memory as the number of state variables increases. Given that the number of state variables corresponds to the number of players involved, scaling up to include more firms results in significantly higher computational costs.

3.1 Physics-Informed Neural Network (PINN)

A neural network is a function $y = f(x;\theta)$ with parameters θ that maps an input x to output y (Prince, 2023). A physics-informed neural network (PINN) is a neural network trained to approximate the solution of a partial differential equation. In our case, we apply this technique to solve the stationary PDE system (8) subjected to the Dirichlet boundary conditions (9). To do so, we created two PINNs, $V_{NN_1}(x;\theta)$ and $V_{NN_2}(x;\theta)$, where $x \in [-1,1]^2$ and θ is a vector of parameters, to approximate the players' value functions. Instead of employing two additional PINNs for the policy functions, we chose to directly utilize the networks of the value functions, in accordance with the first-order condition stated in equation (7).

The method seeks to satisfy the HJB and all boundary conditions for both players. The Universal Approximation Theorem states that any continuous function on compact subsets of $\mathbb{R}^n$ can be uniformly approximated by enough hidden units (Cybenko, 1989; Hornik, 1991). However,

due to the rank-order structure of the prize in our game, the players' value functions $V_1(x_1, x_2)$ and $V_2(x_1, x_2)$ are discontinuous at the corners $[-1, -1]$ and $[1, 1]$. We can better visualize this fact in Figure 1. This will impose some challenges in our solution that will be discussed later in the work. The dynamics behind the method work as follows. At each iteration, we sample collocation points $x \in (-1, 1)^2$ and boundary points as we can see in Figure 1 and we use them to calculate the HJB loss and the boundary loss for each player.



(a) (b)

Figure 1: Boundary and collocation points for both players.

The Hamilton-Jacobi-Bellman (HJB) and boundary losses are defined in equations (10) and (11), respectively. In equation (10), we calculate the difference between the right and left-hand sides of equation (2). Given the presence of a maximum operator, we substitute a_i and a_j with the optimal policies provided by equation (7). Furthermore, N_C denotes the number of collocation points utilized for computing the difference between the two sides. Subsequently, we calculated the mean squared error of the collocation points. Similarly, in equation (12), we evaluate the difference between the right and left-hand sides of equations (3)-(6). Finally, N_{B_1}, N_{B_2}, N_{B_3}, and N_{B_4} represent the number of points employed at each boundary, and the boundary loss is computed as the mean squared error.

$$\mathcal{L}_{HJB}(\theta) = \frac{1}{N_C}\sum_{i=1}^{N_C}\left(\frac{1}{2k_i}\left(\frac{\partial V_{NN_i}}{\partial x_i}\right)^2 + \frac{1}{2}(\sigma_i^2+\eta^2)\frac{\partial^2 V_{NN_i}}{\partial x_i^2} + \frac{1}{k_2}\frac{\partial V_{NN_j}}{\partial x_j}\frac{\partial V_{NN_i}}{\partial x_j} + \frac{1}{2}(\sigma_j^2+\eta^2)\frac{\partial^2 V_{NN_i}}{\partial x_j^2} + \eta^2\frac{\partial^2 V_{NN_i}}{\partial x_i \partial x_j} - rV_{NN_i}\right)^2 \tag{10}$$

$$\begin{aligned}\mathcal{L}_{BC}(\theta) = &\frac{1}{N_{B_1}}\sum_{i=1}^{N_{B_1}}(V_{NN_i}(x_i = 1, x_j \leq 1) - 1)^2 \\ &+ \frac{1}{N_{B_2}}\sum_{i=1}^{N_{B_2}}(V_{NN_i}(x_i < 1, x_j = 1) - 0)^2 \\ &+ \frac{1}{N_{B_3}}\sum_{i=1}^{N_{B_3}}(V_{NN_i}(x_i \geq -1, x_j = -1) - 1)^2 \\ &+ \frac{1}{N_{B_4}}\sum_{i=1}^{N_{B_4}}(V_{NN_i}(x_i = -1, x_j > -1) - 0)^2\end{aligned} \tag{11}$$

The total loss is given by:

$$\mathcal{L}_{total}(\theta) = \mathcal{L}_{HJB}(\theta) + \mathcal{L}_{BC}(\theta) \tag{12}$$

Finally, the method moves the parameter θ of $V_{NN_i}(x;\theta)$ in the direction that minimizes $\mathcal{L}_{total}(\theta)$ the fastest.

3.2 SoftAdapt

In traditional approaches, equal weights are often assigned to the elements of a multi-target loss function, as depicted in equation (12). However, as addressed by (Heydari et al., 2019), this can yield sub-optimal results. The issue arises from variations in the magnitudes of individual components, causing one element of the loss to exert disproportionate influence over the optimization process. Consequently, the optimizer may prioritize reducing the component with the highest magnitude, neglecting the others.

For each player, we modify the structure of the total loss by adding weights to its components:

$$\mathcal{L}_{total}(\theta) = \alpha_{HJB}\mathcal{L}_{HJB}(\theta) + \alpha_{BC}\mathcal{L}_{BC}(\theta)$$

The SoftAdapt method, introduced by (Heydari et al., 2019), calculates each weight by the following mechanism:

$$\alpha_k^i = \frac{f_k^i e^{\beta s_k^i}}{\sum_{l=1}^n f_l^i e^{\beta s_l^i}}$$

where $f_k^i := f_k(x^i)$ represents the value of component k at the iteration i and $s_k^i = f_k^i - f_k^{i-1}$ represents the recent change of given component. Moreover, the β is a tunable *hyperparameter* that controls which component of the loss we favor the most.

In our work, after a few tests, we noticed that the boundary loss term had a higher magnitude than the HJB loss term. We opted to use the SoftAdapt method to balance them and avoid prioritization dynamically. [1]

3.3 Architecture

A neural network $y = f(x;\theta)$ is a composition of linear transformations and non-linear functions. Moreover, it can be understood as an algorithm to approximate functions. In our work, we use two physics-informed neural networks, each one representing a player's value function. They map a 2-dimensional input ($x \in [-1,1]^2$) to a 1-dimensional output as we can see in Figure 2.

To gain insight into why neural networks offer a promising approach for capturing complex patterns from data and for modeling value functions, it is essential to delve into their internal workings. In our case, the neural network receives a two-dimensional input, denoted as $X = [X_1, X_2]$, which is represented in the input layer by two neurons (see Figure 2). Subsequently, this input is passed to the hidden layers. Within these hidden layers, each neuron receives the input X, undergoes a linear transformation, and then applies a non-linear activation function. These operations, depicted below, illustrate the intricate processes occurring within the neural network.

$$
\begin{aligned}
z_1 &= w_1 X + b_1 & a_1 &= \sigma(z_1)\\
z_2 &= w_2 X + b_2 & a_2 &= \sigma(z_2)\\
z_3 &= w_3 X + b_3 & a_3 &= \sigma(z_3)\\
z_4 &= w_4 X + b_4 & a_4 &= \sigma(z_4)\\
&\vdots & &\vdots\\
z_{64} &= w_{64} X + b_{64} & a_{64} &= \sigma(z_{64})
\end{aligned}
$$

where z is the result of a linear transformation, w is a vector of weights and b is a bias term. Moreover, $\sigma(.)$ is a non-linear function applied to the result from the linear transformation. This is important to add non-linearity to capture complex patterns.

The choice of the activation function is an important feature in neural network literature as it influences directly the results. The most used is the Rectified Linear Unit (ReLU), defined below:

$$
f(x) = \begin{cases} x, \text{if } x \geq 0 \\ 0, \text{if } x < 0 \end{cases}
$$

However, since we are working with a second-order stochastic PDE as in Equation (2), we need an activation function that has flexible derivatives. This is not the case with ReLU because its

[1] The code available on a GitHub repository:https://github.com/dr-aheydari/SoftAdapt

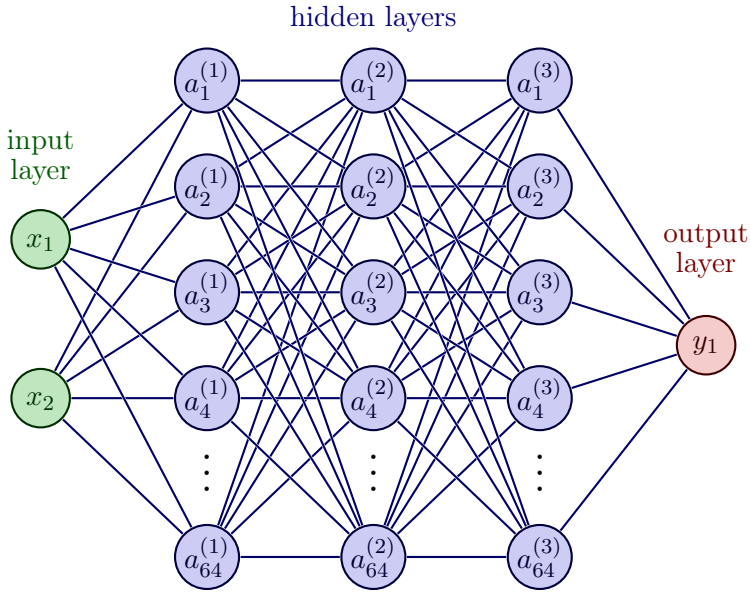


Figure 2: Neural Network Architecture: each column represents a layer and each node, a neuron. We opted for a fully connected NN composed of 3 hidden layers with 64 neurons each. The input layer is bi-dimensional because the state space $x \in [-1, 1]^2$ and the output layer is uni-dimensional.

first-order derivative is 0 for $x < 0$, 1 for $x > 0$, and not defined at $x = 0$. This means that the optimal effort given by $a_i^* = \frac{1}{k}\frac{\partial V_i}{\partial x_i}$ would be 0 for $x < 0$, $\frac{1}{k_i}$ for $x > 0$ and not defined at $x = 0$. Moreover, the second-order derivative is 0 everywhere, but $x = 0$ where it is not defined. Therefore, the HJB in equation (2) would become impossible to be satisfied, so we opted for the Sigmoid Linear Unit (SiLU), defined below:

$$f(x) = x\sigma(x)$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic sigmoid function.

Even though ReLU can help to capture the discontinuities at the corners, we had to opt for another activation function to satisfy the physics law described by our stochastic PDE. Consequently, we faced challenges in capturing the solution near discontinuities, consistent with findings from Sharma et al., 2023. Finally, our architecture is formed by 3 hidden layers with 64 neurons each and the SiLU activation function.

3.4 Training Loop

To begin the training process, we need some artificial data $X = [X_1, X_2]$ to feed into our Physics-Informed Neural Network (PINN). To achieve this, we developed a function responsible for sampling collocation points within $(-1, 1) \times (-1, 1)$, and boundary points as well. As illustrated in Figure 1, each point within the domain is sampled with equal probability. In our code, on each epoch, we used 5.000 collocation points and 1.000 points for each boundary. Moreover, we resample during the training loop in a way to perform a Monte Carlo simulation.

Subsequently, the network processes the data using linear transformations and non-linear activation functions to compute the output $y = f(x;\theta)$. Following this computation, the output is utilized to evaluate the loss ($\mathcal{L}(\theta)$) outlined in equation (12). In our study, we have adopted the Mean Squared Error (MSE) criterion for this loss. The method, then, computes the derivatives (gradients) of the loss with respect to the parameters θ and adjusts the parameters based on the gradients to decrease the loss (Prince, 2023). The goal of an optimization algorithm is to find $\hat{\theta}$ such that:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}(\theta) \tag{13}$$

In this work, we use an algorithm called *gradient descent*. Starting with initial parameters $\theta = [\theta_1, \theta_2, \ldots, \theta_n]^T$, it works as follows:

Step 1: It computes the derivatives of the loss with respect to the parameters:

$$\frac{\partial \mathcal{L}}{\partial \theta} = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial \theta_1} \\ \frac{\partial \mathcal{L}}{\partial \theta_2} \\ \vdots \\ \frac{\partial \mathcal{L}}{\partial \theta_n} \end{bmatrix}$$

Step 2: It updates the parameters according to the rule:

$$\theta \leftarrow \theta - \alpha \frac{\partial \mathcal{L}}{\partial \theta}$$

where α is a *hyperparameter* known as the learning rate and controls the magnitude of the change.

The learning rate choice is a very important issue for neural network training. For example, if we choose a too-small value, our training process can get stuck. On the other hand, if we choose a too-large value, we can learn a sub-optimal set of weights too fast or we can have an unstable training process.[2] To illustrate these features, we trained a fully connected neural network with 2 hidden layers with 32 neurons each, using three distinct fixed learning rate values. The training process becomes increasingly unstable with higher learning rate values, as shown in Figure 3.

[2] http://tinyurl.com/bdhvn3fa

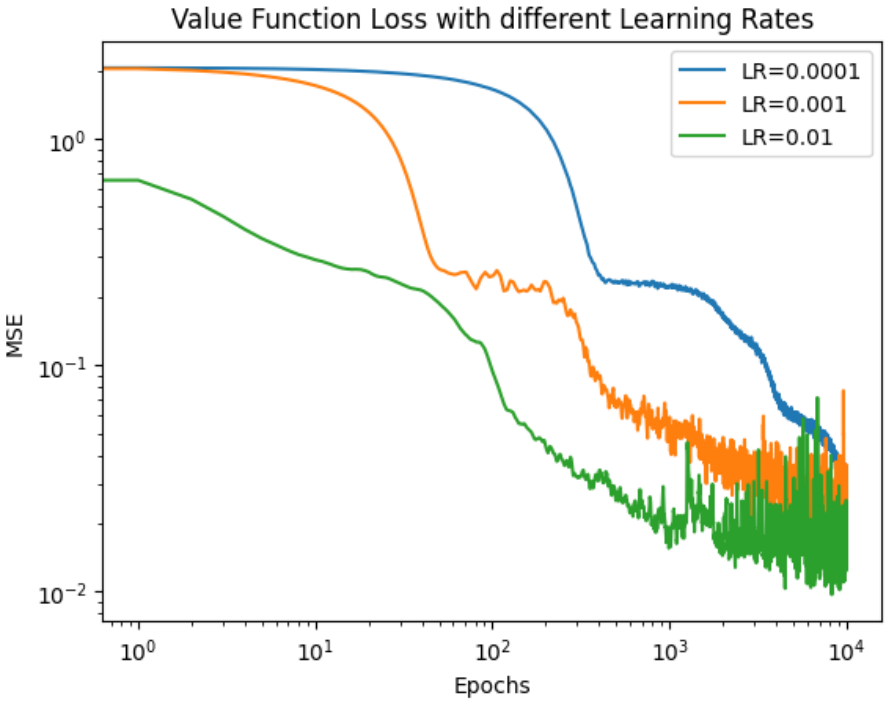


Figure 3: The evolution of Player 1's loss is depicted. We trained a fully connected neural network with 2 hidden layers with 32 neurons each. During the training loop, we experimented with three distinct fixed learning rate values to assess the impact of this hyperparameter on the loss. The training process becomes increasingly unstable with higher learning rate values.

4 Results

As previously mentioned, Duarte et al., 2024 utilizes a neural network approach to address the asset pricing problem outlined in the two-tree economy model by Cochrane et al., 2008. In their study, they validate their findings by comparing them with an analytical solution. However, the stochastic partial differential equation (PDE) system (8) under the Dirichlet boundary conditions (9) that we are tackling lacks an analytical solution. Consequently, we employ the Finite Element Method (FEM) as a numerical approximation, which serves as a benchmark. Despite being an approximation, FEM is widely utilized in the literature and is well-suited for addressing a broad spectrum of PDEs.[3] Therefore, we are confident in comparing the results obtained from our Physics-Informed Neural Networks (PINNs) with those derived from FEM. Table 1 contains the parameters that we established for the numerical solution and Figures 4 and 5 contain the graphs comparing the results.

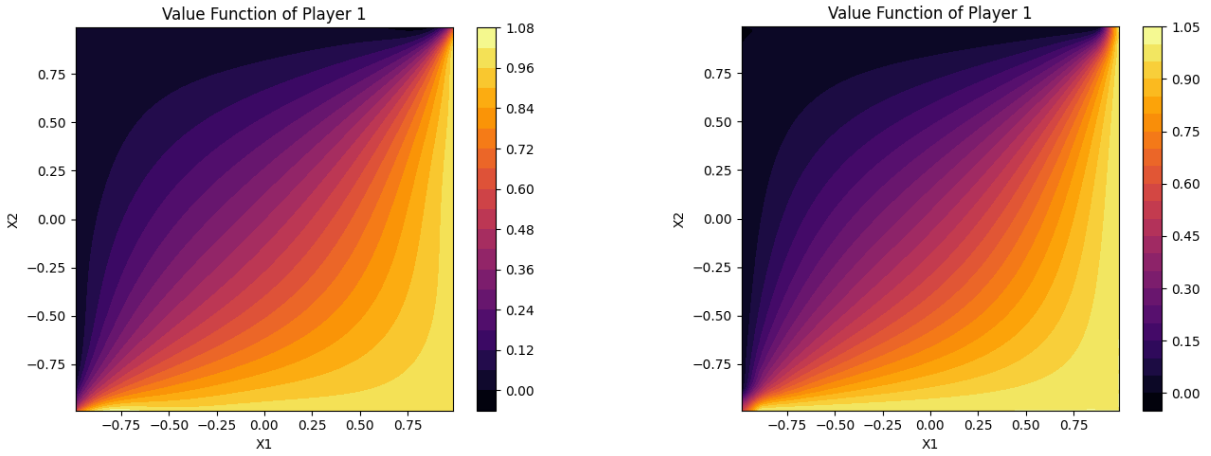
4.1 Value Functions

Although a physics-informed neural network holds promise for solving PDE systems, it did not perform as expected in our case. As discussed in Section 3, we chose the SiLU activation function due to its flexible derivatives. However, our PINN with SiLU failed to accurately capture

[3]For further details, refer to https://reference.wolfram.com/language/FEMDocumentation/tutorial/SolvingPDEwithFEM.html

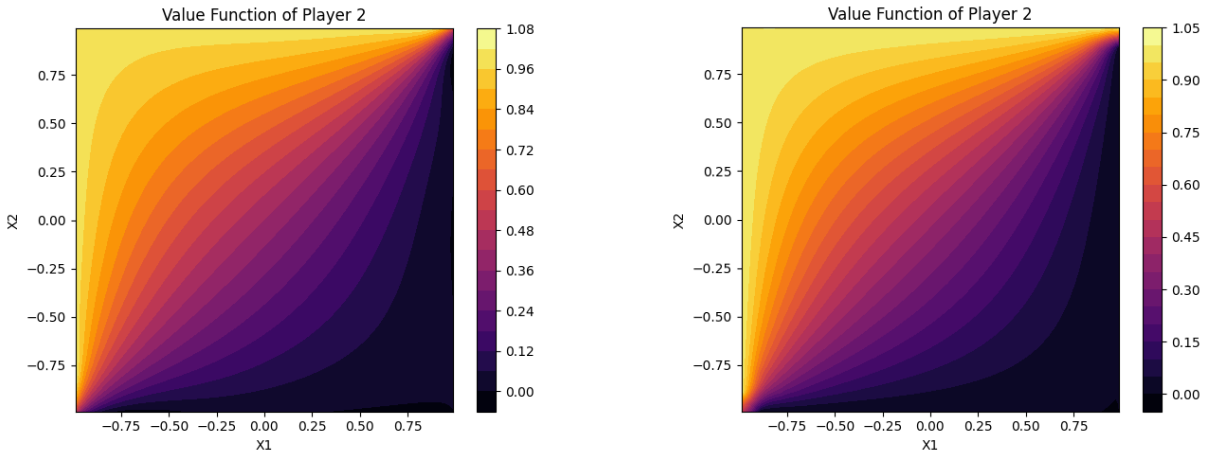
Parameter	Values
r	0.1
σ_1	1.0
σ_2	1.0
η	0.25
k_1	1.0
k_2	1.0

Table 1: Parameter values for the numerical solution. Initially, we considered a symmetric problem. In future work, we plan to investigate how introducing an asymmetric race influences the game's dynamics.



(a) PINN (b) FEM

Figure 4: Figures 4a and 4b represent the solutions using PINN and FEM. Notably, the regions where $V_1(x_1 = 1, x_2 \leq 1) = V_1(x_1 \geq -1, x_2 = -1) = 1$ are represented by the hottest colors, while $V_1(x_1 < 1, x_2 = 1) = V_1(x_1 = -1, x_2 > -1) = 0$ is indicated by black. Near the corners at $[-1, -1]$ and [1,1], where discontinuities are observed, the function exhibits a blend of colors.

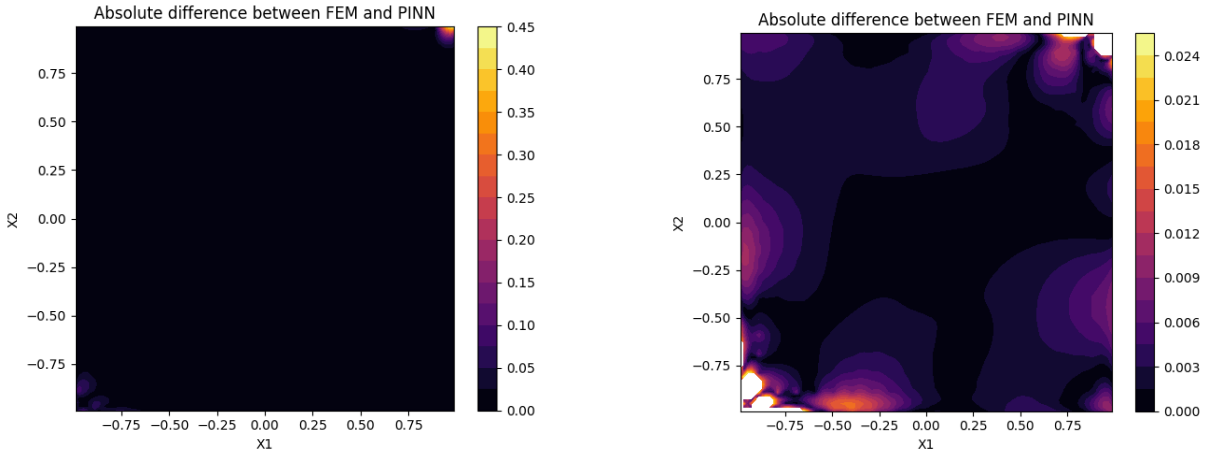


(a) PINN (b) FEM

Figure 5: Figures 5a and 5b represent the solutions using PINN and FEM. Notably, the regions where $V_2(x_1 \leq 1, x_2 = 1) = V_2(x_1 = -1, x_2 \geq -1) = 1$ are represented by the hottest colors, while $V_2(x_1 = 1, x_2 < 1) = V_1(x_1 > -1, x_2 = -1) = 0$ is indicated by black. Near the corners at $[-1, -1]$ and [1,1], where discontinuities are observed, the function exhibits a blend of colors.

the discontinuities at the corners $[-1,-1]$ and $[1,1]$. Consequently, we were unable to minimize the loss magnitude sufficiently, and our PINN results did not closely match those obtained from FEM. Table 2 presents the main statistics regarding the absolute difference between the results of FEM and PINN. Furthermore, Figure 6 shows a heatmap of the absolute difference on the domain $[-1,1]^2$ and we can see that the discontinuities at the corners were the main problem, consistent with Sharma et al., 2023.

Throughout the training loop, the total loss magnitude, as described in equation (12), consistently remained around 10^{-2}, as illustrated in Figure 3. Despite employing a fully connected neural network with three hidden layers, each containing 64 neurons, our final results showed minimal alteration. Specifically, the magnitude of the total loss remained unchanged. Although the neural network appeared to have converged as evidenced by the cessation of loss reduction, its relatively high magnitude indicates that it likely settled in a local minimum rather than a global one. Attempts to mitigate this issue, such as altering the activation function, increasing the number of epochs or collocation points, and adjusting the learning rate, proved ineffective in improving results. Additionally, efforts to sample more points near the corners at $[-1,-1]$ and [1,1] were made; however, due to the continuous activation function, our Physics-Informed Neural Networks (PINNs) failed to capture the discontinuities.



(a) (b)

Figure 6: Absolute difference between Finite Element Method (FEM) and Physics-Informed Neural Network (PINN) results. Figure 6b excludes values above the 99^{th} percentile to enhance visualization. Figure 6a illustrates that the corners [-1, -1] and [1, 1], where discontinuities occur, exhibit the largest differences. Due to the necessity of using a continuous activation function, our PINN approach was unable to effectively handle these discontinuities.

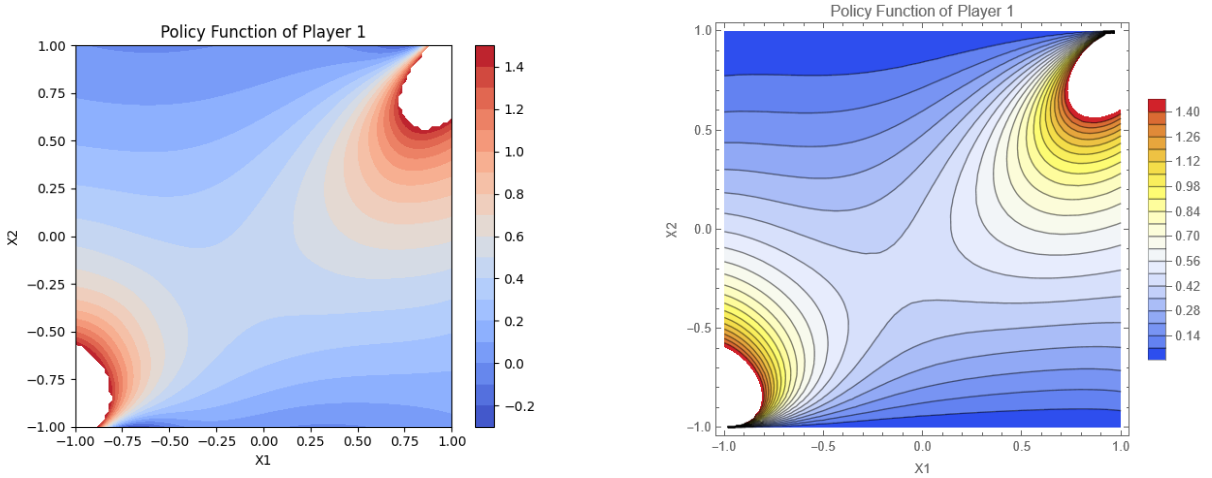
4.2 Policy Functions

Having obtained the players' value functions, we derive the optimal policy directly using Equation (8). The results for the players' policies are illustrated in Figures 7 and 8. The effort choices are asymmetric, with higher efforts in regions near the discontinuities where players face more competition. Additionally, efforts are slightly displaced towards regions where the player leads the race. Furthermore, we observe that both players reduce their effort when a competitor holds

Absolute Difference between PINNs and FEM	
Statistic	Result
Mean	3.019×10^{-3}
Median	1.887×10^{-3}
Standard Deviation	9.201×10^{-3}
(0,0)	1.016×10^{-4}

Table 2: Main statistics of the absolute difference between the results of FEM and PINN for Player 1's value function. To compute the difference, we divided the domain $[-1, 1]^2$ into a grid with 40.000 points. Additionally, we computed the absolute error at (0,0) because it is equidistant from the corners $[-1, -1]$ and $[1, 1]$, where the discontinuities occur. Similar results were obtained for Player 2.

a significant lead, a phenomenon known as *discouragement effect*.

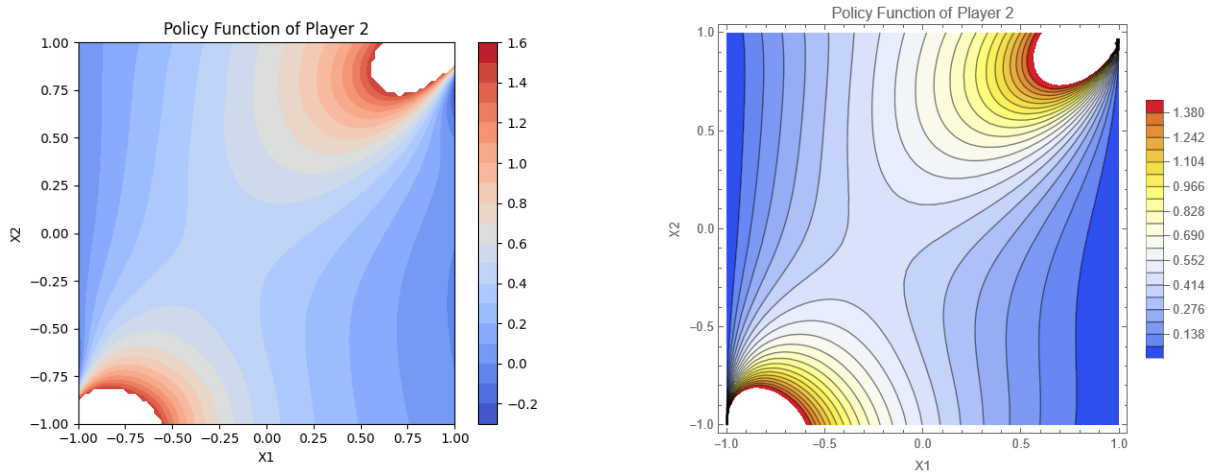


(a) PINN (b) FEM

Figure 7: Figure 7a and 7b were plotted using Python and Wolfram Mathematica, respectively. We can see a heightened effort level when players are jointly close to the finish line. Also, when a competitor holds a significant lead, both players reduce their level of effort (*discouragement effect*).

4.3 Robustness checks

While it's true that we didn't achieve the expected low magnitudes for the loss function, it's important to carefully consider the implications of this outcome. The errors are particularly pronounced in regions near the discontinuities as shown in Figure 6, where high levels of effort are associated. It's plausible that these regions are infrequently visited or traversed rapidly by the players. To gain deeper insights into the accuracy of our findings, we have conducted two robustness checks. First, we simulate the game using both FEM's and PINN's policy functions results to create a notion of a probability density function (PDF) and we compare them using the sup-norm criterion. In other words, we evaluate $D = \sup_x |f_{FEM}(x) - f_{PINN}(x)|$. Secondly, we compute the expected discounted effort (EDE), using both a PINN and a FEM approach. Then, we check whether the starting point for a principal aiming to maximize the EDE is similar



(a) PINN (b) FEM

Figure 8: Figure 8a and 8b were plotted using Python and Wolfram Mathematica, respectively. We can see a heightened effort level when players are jointly close to the finish line. Also, when a competitor holds a significant lead, both players reduce their level of effort (*discouragement effect*).

in both methods.

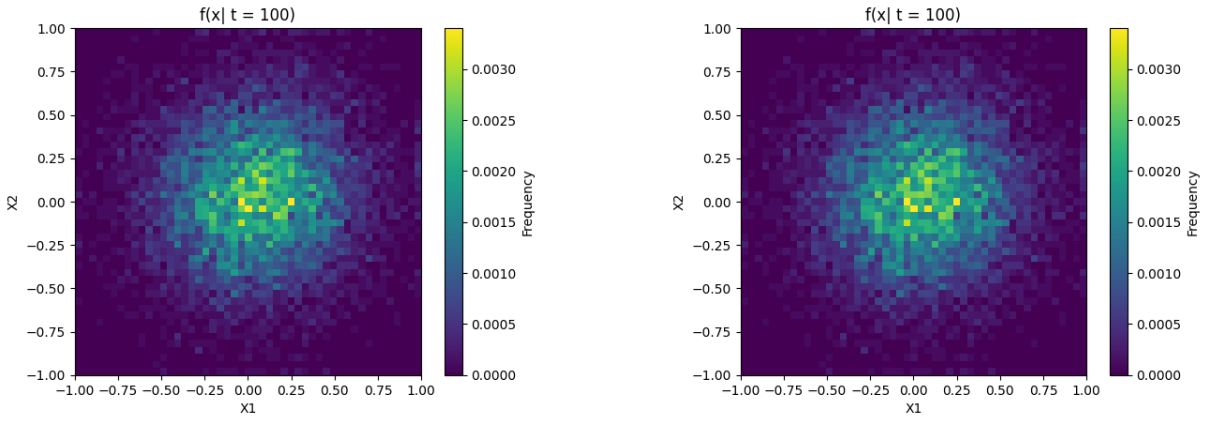
4.3.1 Probability Density Function (PDF):

To conduct the simulations, we create three matrices of shocks (one common and two individual shocks as in Equation (1)). We use them for both PINN and FEM simulations to isolate the shocks' effects and only account for differences in the methods' results. Also, we suppose $dt = 0.001$. The objective is to create a notion of PDF over the game's domain $[-1, 1] \times [-1, 1]$ and compare both functions. Moreover, we will be able to calculate not only the distribution given a specific time t after the game starts but also the game's condition (finished or not). In other words, we will access both $f(x|t = T)$ and $f(x|t = T, \mathcal{I} = i)$ where $\mathcal{I}$ is an indicator function that assumes one if the game finished and zero, otherwise. This allows us to understand the dynamics of games that finished early or late, accessing the path incurred. Figure 9, 10 and 11 display $f(x|t = 100)$, $f(x|t = 500)$ and $f(x|t = 1000)$, respectively, for both methods. Additionally, Table 3 shows the sup-norm between them.

Sup-norm $f(x\|t = T)$	
T	Result
100	2.999×10^{-4}
500	1.200×10^{-3}
1000	1.600×10^{-3}

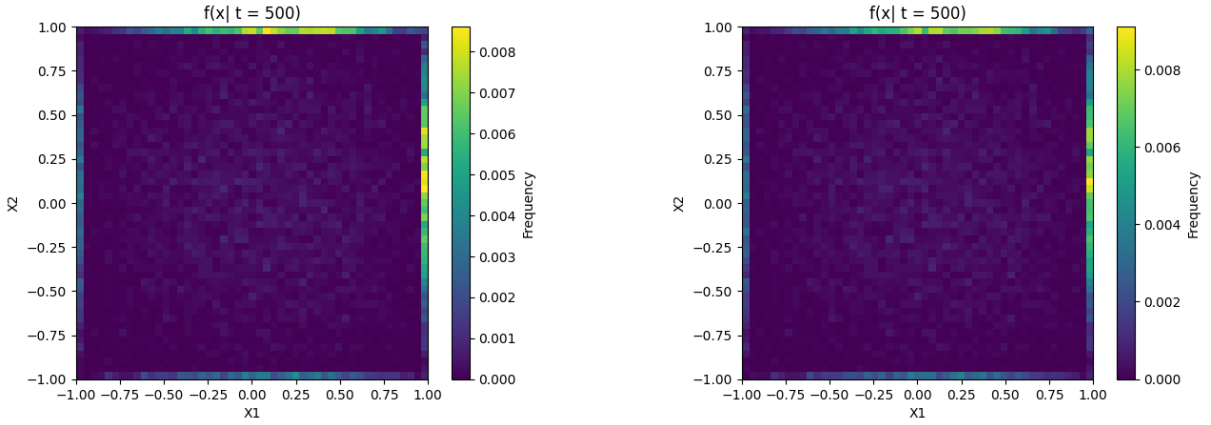
Table 3: We compared PINN's and FEM's results using a sup-norm criterion. As we can see, the maximum differences between the probability density functions are 0.03, 0.12, and 0.16 percentual points when T is equal to 100, 500, and 1000, respectively.

While the corners $[-1, -1]$ and [1,1] exhibited high errors in our Physics-Informed Neural Net-



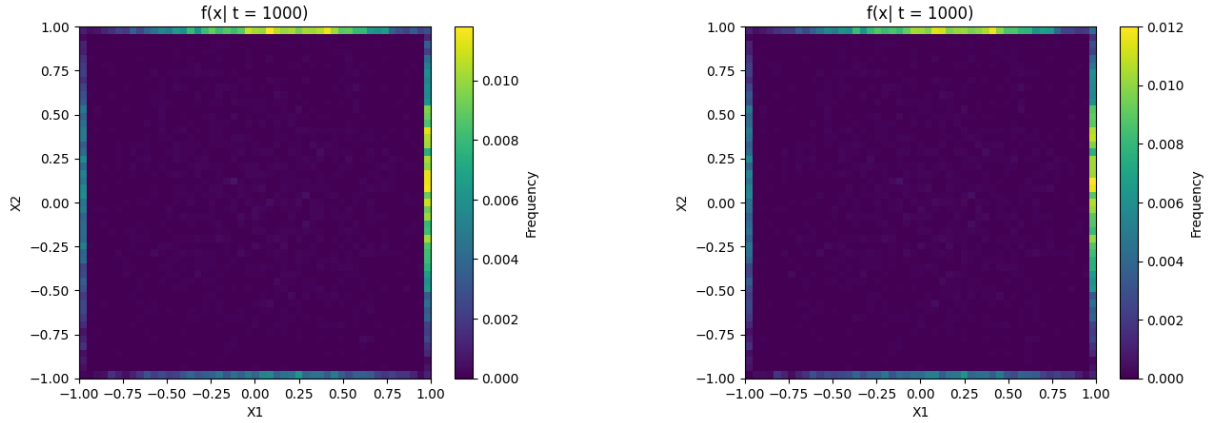
(a) PINN (b) FEM

Figure 9: We simulated 10.000 races starting at (0,0). Then, we discretized the domain $[-1, 1]^2$ and computed the probability density function (PDF) at $t = 100$, denoted as $f(x|t = 100)$. The majority of games are still in an early stage, with only a small number of games finished. Notably, regions near discontinuities are not frequently visited.



(a) PINN (b) FEM

Figure 10: We simulated 10.000 races starting at (0,0). Then, we discretized the domain $[-1, 1]^2$ and computed the probability density function (PDF) at $t = 500$, denoted as $f(x|t = 500)$. The majority of games are finished, but a considerable number of games remain in progress. Notably, regions near discontinuities are not frequently visited.



(a) PINN

(b) FEM

Figure 11: We simulated 10.000 races starting at (0,0). Then, we discretized the domain $[-1,1]^2$ and computed the probability density function (PDF) at $t = 1000$, denoted as $f(x|t = 1000)$. Most games have concluded, with only a few remaining in progress. Notably, regions near discontinuities are infrequently visited.

work (PINN) approach compared with the Finite Element Method (FEM), Figures (9)-(11) reveal that these regions are infrequently visited. Furthermore, the probability density functions (PDFs) generated by the results of both methods do not show considerable differences, as demonstrated by the *sup-norm* criterion in Table 3. Even at a time like $T = 1000$, when most games have finished and players must have crossed the finish line, the areas near the discontinuities do not significantly impact the outcomes. The results in Table 3 indicate that for $T = 1000$, the *sup-norm* is 0.16 percentage point. Ultimately, while these areas may have affected training, they do not substantially influence the dynamics of the race when compared to races derived from the Finite Element Method (FEM).

4.3.2 Expected Discounted Effort

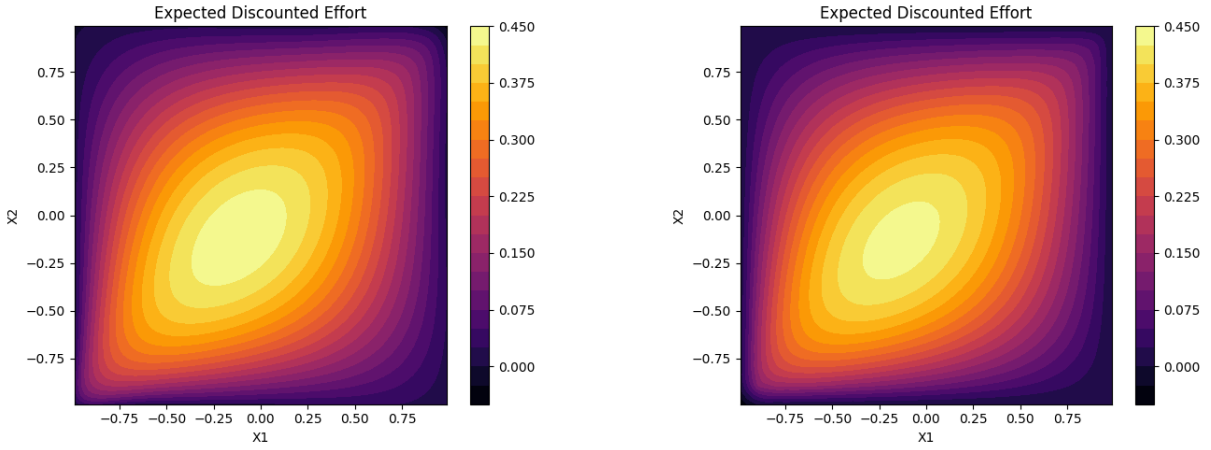
According to Duarte et al., 2024, we can incorporate parameters of interest such as the prize structure and product quality, along with state variables, as inputs to the network. This helps perform comparative statics and design an optimal race. However, we encountered challenges in our two-firm race due to discontinuities at the corners [-1, 1] and [1, 1], which required further investigation. Consequently, we will not analyze how a principal can use the prize structure or the product's quality to design an optimal race. Instead, assuming these parameters are fixed, our focus is on understanding how a principal can maximize his expected utility. Furthermore, suppose that his utility is a function of players' level of effort, the principal aims to maximize:

$$\begin{aligned} ru(x_1,x_2) = a_1(x_1,x_2) + a_2(x_1,x_2) + a_1(x_1,x_2)\frac{\partial u(x_1,x_2)}{\partial x_1} + \frac{1}{2}(\sigma_1^2+\eta^2)\frac{\partial^2 u(x_1,x_2)}{\partial x_1^2} \\ + a_2(x_1,x_2)\frac{\partial u(x_1,x_2)}{\partial x_2} + \frac{1}{2}(\sigma_2^2+\eta^2)\frac{\partial^2 u(x_1,x_2)}{\partial x_2^2} + \eta^2\frac{\partial^2 u(x_1,x_2)}{\partial x_1 \partial x_2} \end{aligned} \tag{14}$$

subject to the Dirichlet boundary conditions:

$$u(W, x_2) = u(L, x_2) = u(x_1, W) = u(x_1, L) = 0 \tag{15}$$

To conduct the robustness check, we solve the Hamilton-Jacobi-Bellman (HJB) equation mentioned earlier using the policy functions $a_1(x_1, x_2)$ and $a_2(x_1, x_2)$ derived from both Physics-Informed Neural Network (PINN) and Finite Element Method (FEM) results. Additionally, we develop a new neural network to solve for $u(x_1, x_2)$ using PINN's policy functions and compare it with FEM's solution using their respective policies. The objective is to assess whether these methods can produce similar results for the expected discounted effort. The results are depicted in Figure 12, showing that a principal aiming to maximize the expected discounted effort should initiate the race below the point (0,0) along the 45-degree line in both methods. Specifically, the optimal starting point from FEM is (-0.137, -0.137), whereas PINN's is (-0.131, -0.131), indicating that both methods yielded similar results.



(a) PINN (b) FEM

Figure 12: Figures 12a and 12b represent the solution using PINN and FEM. Notably, the maximum point occurs below the point (0,0) along the 45-degree line in both methods. Specifically, they are (-0.137, -0.137) and (-0.131, -0.131) for FEM and PINN, respectively. Moreover, both methods captured the Dirichlet boundary conditions $u(1, x_2) = u(-1, x_2) = u(x_1, 1) = u(x_1, -1) = 0$.

5 Conclusion

We investigated an innovation race with continuous effort choices and observed that the rank-order prize structure encourages players to increase their effort when they are jointly close to the finish line. Conversely, both players tend to reduce their effort when a competitor holds a substantial lead, a phenomenon known as *discouragement effect* in line with (Cao, 2014), illustrated in Figures 7 and 8. Although we employed a promising deep-learning approach, the Physics-Informed Neural Network (PINN) method implemented in our study did not achieve sufficiently low magnitudes for the loss function shown in Equation (12). Our results, compared with the Finite Element Method (FEM), did not closely align, mainly due to our neural network's use

of a continuous activation function, which failed to capture the discontinuities at the corners $[-1,-1]$ and [1,1], consistent with findings by (Sharma et al., 2023).

Consequently, we were unable to generalize our race model for a multiple-firm scenario without conducting additional robustness checks. We conducted simulations using results from both methods to compare the probability density function (PDF) $f(x|t=T)$ derived from them, represented in heatmaps for $T = 100$, 500, and 1000 in Figures 9, 10, and 11, respectively. Regions near the corners $[-1,-1]$ and [1,1], where our PINN approach exhibited higher errors due to discontinuities in the prize structure, were infrequently visited. Furthermore, the PDFs generated by both methods did not exhibit considerable differences, as indicated by the *sup-norm* criterion in Table 3. Therefore, despite errors near discontinuities in our physics-informed neural network, these did not significantly impact race simulations.

Moreover, we utilized the policy functions $a_1(x_1, x_2)$ and $a_2(x_1, x_2)$ derived from both Physics-Informed Neural Networks (PINN) and the Finite Element Method (FEM) results to compute the expected discounted effort (EDE). In this analysis, we developed a new neural network to calculate the EDE and then compared it with FEM's solution. Our findings indicate that a principal seeking to maximize the expected discounted effort should start the race below the point (0,0) along the 45-degree line in both methods. Notably, the maximum points from PINN and FEM were (-0.131, -0.131) and (-0.137, -0.137), respectively. Thus, we observe that both methods yielded similar results for the expected discounted effort.

For future work, we aim to extend our two-firm race model to a multiple-firm scenario. Despite encountering training errors, these did not substantially affect race simulations compared to the Finite Element Method. Additionally, exploring alternative PINN architectures such as Highway, convolutional, and skip connection networks may improve our results, which we plan to investigate in future studies. Finally, given that most errors stemmed from the rank-order prize structure, we intend to explore tournaments with continuous prize structures to assess the deep-learning approach's capabilities presented in this paper.

References

Bellman, R. (1957). *Dynamic programming* (1st ed.). Princeton University Press.

Cao, D. (2014). Racing under uncertainty: Boundary value problem approach. *Journal of Economic Theory*, *151*, 508–527.

Cochrane, J. H., Longstaff, F. A., & Santa-Clara, P. (2008). Two trees. *The Review of Financial Studies*, *21*(1), 347–385.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, *2*(4), 303–314.

Duarte, V., Duarte, D., & Silva, D. (2024). Machine learning for continuous-time finance.

Fernández-Villaverde, J., Marbet, J., Nuño, G., & Rachedi, O. (2023). *Inequality and the zero lower bound* (tech. rep.). National Bureau of Economic Research.

Gopalakrishna, G. (2021). Aliens and continuous time economies. *Swiss Finance Institute Research Paper*, (21-34).

Gorno, L., & Iachan, F. S. (2020). Competitive real options under private information. *Journal of Economic Theory*, *185*, 104945.

Heydari, A. A., Thompson, C. A., & Mehmood, A. (2019). Softadapt: Techniques for adaptive loss weighting of neural networks with multi-part loss functions. *arXiv preprint arXiv:1912.12355*.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks*, *4*(2), 251–257.

Prince, S. J. (2023). *Understanding deep learning*. MIT Press. http://udlbook.com

Sharma, P., Evans, L., Tindall, M., & Nithiarasu, P. (2023). Stiff-pdes and physics-informed neural networks. *Archives of Computational Methods in Engineering*, *30*(5), 2929–2958.