How Simulation can help researchers extend Game Theory population models

Florent Daburon, Lucien Lorenz

May 2023

1 Abstract

This study presents the development and adaptation of simultaneous game models, lessening the requirement for comprehensive command over Game Theory and its complex mathematics. The Hawk and Dove game is utilized as a foundational model. The theoretical framework predicts a persistent evolutionary equilibrium, which the initial simulation aims to replicate. Subsequent model modifications incorporate factors such as genetic mutations, population regulation, and unique player properties. Results indicate that the simulation not only reproduces expected outcomes but also maintains a distinct equilibrium with the applied model adaptations. These findings hold potential for future applications across various simultaneous games, provided there are certain restrictions on the payoff matrix.

Contents

2 Game Theory and expected results

2.1 Game Theory : Simultaneous Games

A game, as defined in Game Theory, has three parts :

- A list of players
- A strategy space of the players
- A payoff function for each player

For this project we work on two player and two strategies per player leading to four payoffs. To model the evolution of a population we will use a simultaneous symmetric game of complete information. In a simultaneous game, both players choose a strategy at the same time assessing the action of the other player and "predicting" its best response thanks to the payoff matrix (supposing complete information).

The **Nash Equilibrium**, a fundamental concept in Game Theory, refers to a state in a strategic interaction where no player can unilaterally improve their outcome by deviating from their current strategy, given the strategies of the other players. Essentially, it represents a stable state of a game where players, with knowledge of each other's strategies, have no incentive to change their own. This equilibrium serves as a predictor of the potential outcome of non-cooperative games assuming rational decision-making. Moreover a **mixed strategy** is a probabilistic combination of multiple pure strategies that a player might adopt. It provides an optimal strategy when no single pure strategy secures the best outcome, allowing a player to randomize their choices to keep opponents uncertain. In the famous prisoner's dilemma, both players choose to betray every time. However, such an equilibrium in Pure Strategy might not arise. When the payoffs are opposite (when one player wins, the other always loses), the equilibrium is then a **Mixed Strategy Equilibrium**. It states that instead of player1 and player2 always choosing the same strategy, they will on average play a strategy with probability p and q respectively, and the other with probability $1-p$, $1-q$ respectively.[1]

The goal of this paper is then is to assess if there is a set of strategies that dominates through simulation where theoretical calculation may be too complicated or even impossible to perform.

2.2 Initial specification : Hawk and Dove

The Hawk-Dove game,first introduced by John Maynard Smith and George Price[2], also known as the game of chicken, is a model in evolutionary game theory that simulates an encounter between two individuals contesting a shared resource. The strategies available are 'Hawk', representing aggression, and 'Dove', symbolizing peacefulness or submission. Hawks fight each other until one is injured (paying a large cost) while they intimidate Doves without a fight, gaining the resource. Doves, on the other hand, either share the resource or concede to Hawks, avoiding the cost of conflict. There are many specific payoff matrix of this game, which can be entered in the accompanying program. For the paper will however be using the general form :

	Hawk	**Dove**
Hawk	X,X	W,L
Dove	L,W	T,T

Solutions to this specification hold true if $W > T > L > X$ [5]. With this game, and our project, we're introducing the concept of **Evolutionary Game Theory**, signifying a strategy that resists replacement once it's prevalent in a population. This stability emerges from its inherent environmental advantages, granting it the ability to outperform any rival strategies, thus maintaining its dominant presence. Which is to add to the standard Nash Equilibrium specification specifying that any variation to any strategy could lead to a better payoff.

Nonetheless, in the context off biology, standard game theory assumptions need some adaptation. **Strategies** are traits given buy genetics and new traits given by mutation. Payoffs are a fitness function (probability

[1] See : Appendix for example and equilibrium derivation

to survive if below 1 and the number of offspring if above. Mixed strategies are observed at a population level where two types of individual coexist and Pure strategy when only one trait survive. Finally, rationality is provided by the sole fact that unoptimal strategy adopted by a player will prevent him to survive and thus propagate it to its descendent.

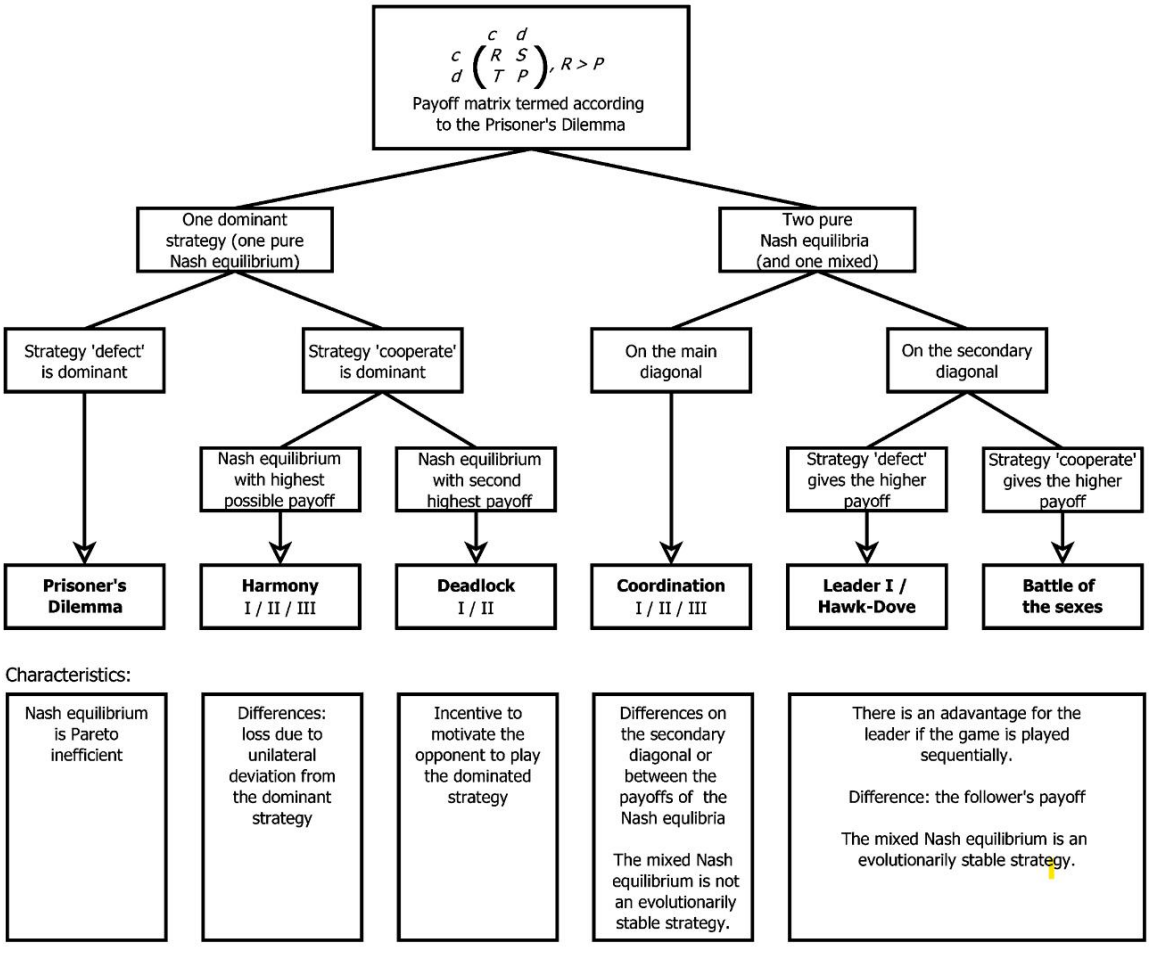
We can also briefly introduce the concepts of **Carrying Capacity** and **Density-dependent growth**, as we will use and observe them in our simulations.
The carrying capacity is the population size the environment can sustain given all constraints on reproduction and survival.

Density-dependent growth shows that in a population, there is competition for material resources and space. The fitness therefore is reduced, the higher the population. We will model this through "resource nodes" (see : Model and Extensions).

In our simulation, using the Hawk and Dove model, we should expect to see a stable ESS, as well as the concept non binding maximum population depending on the payoff matrix.

2.3 Extension to other simultaneous games

Traditional game theory's limitation is that in repeated games, players adjust their strategies based on previous payoffs before reaching an equilibrium, typically converging to a Nash Equilibrium or Mixed Strategy Equilibrium. However, our simulation, focused on population dynamics, models simultaneous games more effectively. It accounts for natural consequences like the elimination of individuals for sub-optimal choices, such as an excess of hawks. This capability expands the potential for theoretical modeling of a broad spectrum of simultaneous games presented in this graph [4] :



3 Model and Extensions

3.1 Main Specifications

We represent our players by a Player class, that has three attributes : its type (either Hawk or Dove), its ID, and its fitness, representing the individual's chance of survival and/or offspring generation.

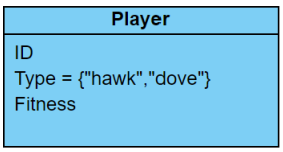


Figure 1: UML representation of the Player class

The simulation will use this player class in a vector to do all the operations needed. We have three main components of the simulation :

- fight() : Handles the pairing of each individual over food nodes
- selection() : Handles the survival and offspring generation
- purge() : Handles the eventual population limit

The fitness of an individual will depend on its pairing and the payoff matrix, which is defined in a standard way through a simultaneous game : If a Hawk ends up against a Hawk, we will use the payoff Payoff_{HH}. If

Type	**Hawk**	**Dove**
Hawk	Payoff_{HH}	Payoff_{HD}
Dove	Payoff_{DH}	Payoff_{DD}

Table 1: Payoff Matrix

a Dove ends up against a Hawk, it will have the fitness of Payoff_{DH}, and Payoff_{HD} if a hawk pairs against a dove.

For the purposes of this paper, we will be tracking different statistics :

- current generation
- total population
- proportion of hawks and dove
- average life expectancy, measured in generations

Generating the Initial Population

To get started with the most basic version of our model, the user needs to specify the initial population, the number of food nodes (see : Pairing), the proportion of doves, the maximal population, the default payoff, and the number of generations wanted.

We then create the deterministic initial population, and add different statistics to a dataframe[2] to store the data of the simulation.

[2]Included in the pandas Python package

Pairing

To determine how the population interacts with itself, we use the specification used by Primer's "Simulating the Evolution of Aggression" [3] :
The population has a specific number of food nodes available. Each individual chooses a node at random. If he is alone on that node, he gets the default fitness. Otherwise, a fight ensues according to the payoff matrix, and each individual gets their respective payoff.
Practically, this means that we fill our vector of individual with 0's until the node amount is reached.

```
nodes_list = [0]*(number_nodes - len(population))
temp_list = population + nodes_list
```

We then proceed to shuffle this list to get the pairing. This raises 5 possibilities :

- A node is by itself : nothing happens
- An individual is on a node by itself : The individual gets the default fitness
- A hawk ends up against a hawk
- A hawk ends up against a dove
- A dove ends up against a dove

The last three cases end up in a so-called "fight", each individual leaving with an amount of food represented by their fitness.

Survival

Once the fight() function is over, each individual has their fitness for this generation. We can then determine if they survive, and if they create offspring (in reality, clones of themselves). Regarding the fitness, we have three cases :

- fitness < 0 : the individual surely dies
- 0 <fitness< 1 : the individual might survive
- fitness ≥ 1 : the individual surely survives, and might create offspring

Fitness between 0 and 1 In this case, we use the random() function provided by the random package [3] : if the random number is larger than the fitness, the individual dies and isn't added to the next generation.

Fitness is larger or equal to 1 The first point of fitness means the individual survives for sure. The remaining integers create offsprings with certainty. If there is a decimal remainder, we use the `random()` function to determine if one more is created.

Purge

We check if the length of the population is larger than the parameter of the maximum population. By default, if it is the case, we remove the excess individuals at random. The user can specify if we instead purge the oldest, or the youngest.

Graphical Interface

We create a user friendly interface for entering the parameters and viewing the graph. This is usefull to be able to change and visualize quickly the selected parameters and is also a way to make it accessible to anyone. This could also be useful in a teaching environment for courses like "Genes, Population et Evolution" here at HEC Lausanne to help students to discover and test key principles. Please have a look a the appendix for a screenshot or consider visiting our GitHub : Hawk and Dove[4].

[3] https://docs.python.org/3/library/random.html
[4] https://github.com/Sakagraisse/Hawk$_{a}nd_{D}ove$

Git Hub

We decided to use github as a platform to co-develop the code in an easier way and improve our productivity. Moreover we make available both the source code and the standalone app for Windows and Linux. We set standard to provide anyone the possibility to access and improve our initial release by metting requirement for the declaration of libraries, compilation files for Linux and Windows and finally a comprehensive "read me".

3.2 Extensions

Population Mutation We include a characterisation which allows the offspring to mutate to the other type. The user can specify the probability to mutate from dove to hawk, and vice-versa.

Simulating time to arrive at node We can model the fact that each type has different times to arrive at a node, that can decrease or increase its fitness. This can take into account a fast individual that can "eat" before the opponent arrives. We assume that such a model follows a Normal Distribution [5], with parameters μ and σ, the expectation and standard deviation. When generating the fitness, we generate a random number that will get substracted :

```
animal.fitness -= normal(loc = mean, scale = sd)
```

Because the normal distribution supports positive and negative numbers, this can accurately describe individuals being faster / slower.

4 Results

If you want to reproduce the results, use a seed of 1

Standard Model

There are many specifications to choose. We will, for a baseline to test if our model is correct, the specification given by Primer's specification[3]. The payoff matrix is as follows :

Type	**Hawk**	**Dove**
Hawk	0	1.5
Dove	0.5	1

The Mixed Strategy Equilibrium suggests a probability of playing "Hawk" equal to the one of playing "Dove", or :

$$p^*_{Hawk} = 0.5, p^*_{Dove} = 0.5$$

The other parameters specify :

$$\begin{aligned} \pi_{\text{default}} &= 2 \\ population_{\text{initial}} &= 500 \\ population_{\text{max}} &= 1000 \\ nodes &= 1000 \\ p^{ini}_{\text{dove}} &= 20\% \\ generations &= 500 \end{aligned}$$

Graphically, we we see two important results : Convergence to the expected equilibrium is correct, and we have an example of carrying capacity. Even though the population can rise to 1000 (and there are sufficient food nodes to support that population level), it converges around 800 :

[5] See : Appendix for distribution behaviour

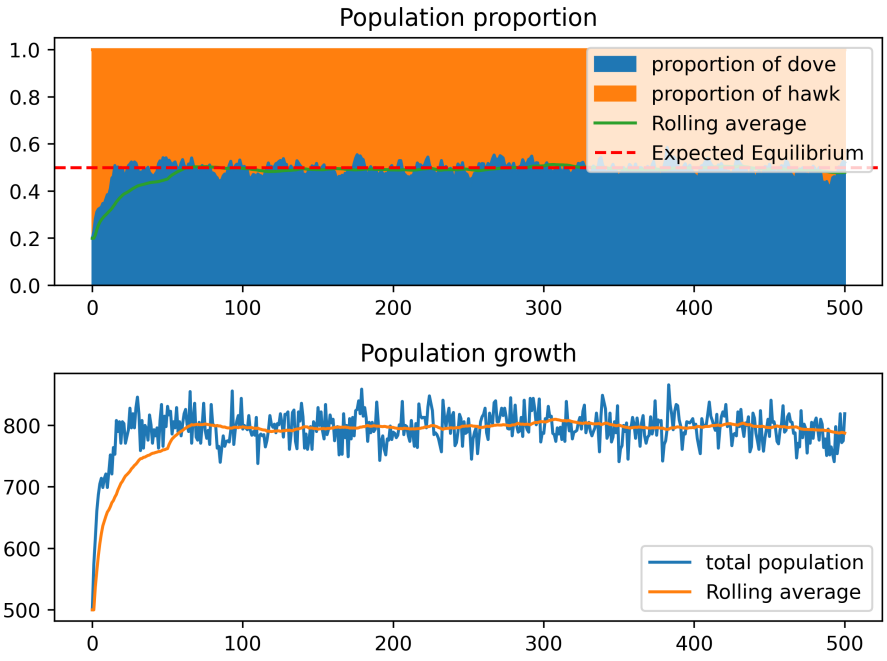


Figure 2: Results vs. Theory

Added mutation

With the same specifications, we allow each offspring to mutate to the other type with probability $p_{\text{dove-to-hawk}} = p_{\text{hawk-to-dove}} = 10\%$. We now see that the equilibrium supports a slightly higher proportion of doves, and the carrying capacity slightly increases.

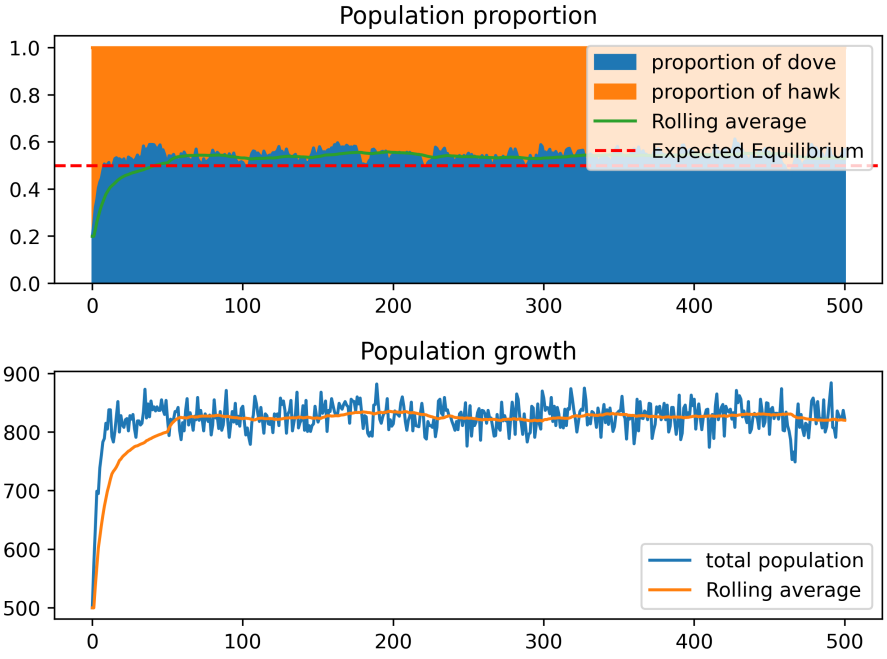


Figure 3: Base model with mutation

Added food search

Finally, we assume that on top of the mutation, Hawks and dove can be faster / slower to get to the food node.

$$\mu_{\text{Hawk}} = -0.3, \quad \sigma_{\text{Hawk}} = 0.3, \quad \mu_{\text{Dove}} = 0, \quad \sigma_{\text{Dove}} = 0.3,$$

This gives us :

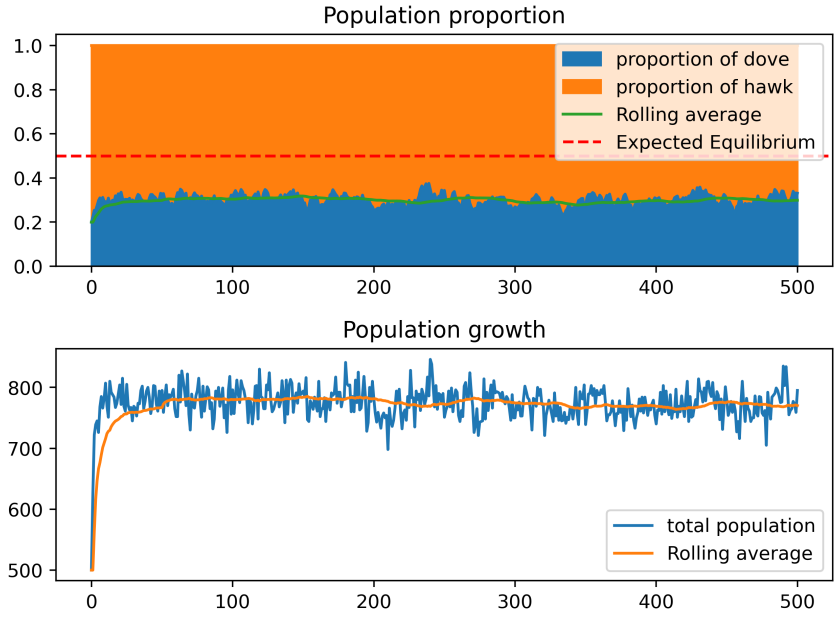


Figure 4: Base Model with Mutation and Search

On average, hawks are faster. This is shown by an increase in the proportion of hawk, yet we still have a convergence. Carrying capacity is slightly lower than the baseline, but this is expected as the number of predators increased.

It is to say that the power of our implementation is the capacity to change parameters at wish and thus it is not possible to cover every case in this paper. The goal here is to prove that it is coherent with theory on the base case and coherent with the logic and the biology theory in its extension. We succeeded in this goal and hopefully some would find it usefull for it specific use case.

5 Conclusion

On one hand, we have developed a framework that allows for modeling the standard game theory results for the Hawk and Dove game within a versatile package, providing flexibility in selecting desired parameters. Additionally, we have successfully incorporated parameters for known biological situations and generated results that align with both theoretical expectations and real-life observations.
On the other hand, we have developed an efficient package featuring a user-friendly interface that emphasizes simplicity and accessibility. By sharing it on GitHub, our intention is to contribute to the community and foster collaboration. Although the package may have some imperfections, we have made efforts to provide all the necessary resources for its utilization, encouraging further development and enhancements.

6 Appendix

Mixed Strategy Equilibrium

Let's assume we have two football players : one shooter, and one keeper. The shooter can either shoot left or right, and the keeper can dive in the same directions. If the keeper dives in the same direction as the shooter shoots, he intercepts the ball and gets an utility of 1, otherwise the shooter marks a goal and the keeper gets an utility of -1. The shooter has mirrored payoffs. In a normal form, we can represent the game as : In this

		Keeper	
		Left	**Right**
Shooter	**Left**	-1,1	1,-1
	Right	1,-1	-1,1

game, because the two players have mirrored goals, there is no equilibrium in pure strategies. However, we can determine how often the shooter shoots left, and the keeper dives left.
Let's assume the keeper thinks that the shooter chooses left with probability p, and right with probability $1-p$. Similarly, let's suppose the shooter thinks the keeper dives left with probability q and right with probability $1-q$. Now, looking at expected payoffs :

$$\begin{aligned}\mathbb{E}\left(\pi_{\text{shooter}}(\text{left})\right) &= -1q + 1(1-q)\\ \mathbb{E}\left(\pi_{\text{shooter}}(\text{right})\right) &= 1q + (-1)(1-q)\end{aligned}$$

Which we can do for the keeper as well. The Equilibrium in Mixed Strategies states that each player has to be indifferent between playing either strategies. In other words :

$$\begin{aligned}\mathbb{E}\left(\pi_{\text{shooter}}(\text{left})\right) &= \mathbb{E}\left(\pi_{\text{shooter}}(\text{right})\right)\\ \mathbb{E}\left(\pi_{\text{keeper}}(\text{left})\right) &= \mathbb{E}\left(\pi_{\text{keeper}}(\text{right})\right)\end{aligned}$$

Solving this system gives us a Mixed Strategy Equilibrium :

$$\begin{aligned}p^* &= 0.5\\ q^* &= 0.5\end{aligned}$$

This can be interpreted as : On average, the shooter shoots left half of the time, and the keeper dives left half of the time.

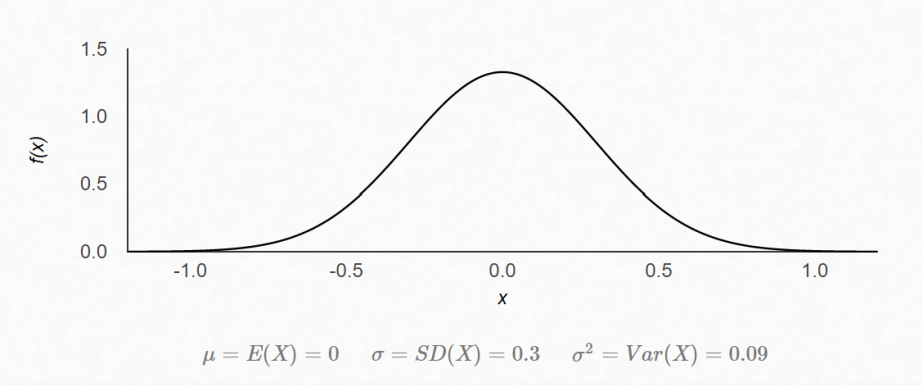
Normal Distribution

the Normal Distribution is a symmetric, and often used distribution function. The probability density function, with parameters μ and λ is defined as :

$$f(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$

$$\begin{aligned}\mathbb{E}(X) &= \mu\\ \text{Var}(X) &= \sigma^2\end{aligned}$$

In the case of the food search specification (see : Results), with $\mu = 0, \sigma = 0.3$, we get the following distribution[1] : We can then shift it and scale it as wished to get different results.



$$\mu = E(X) = 0 \quad \sigma = SD(X) = 0.3 \quad \sigma^2 = Var(X) = 0.09$$

GUI

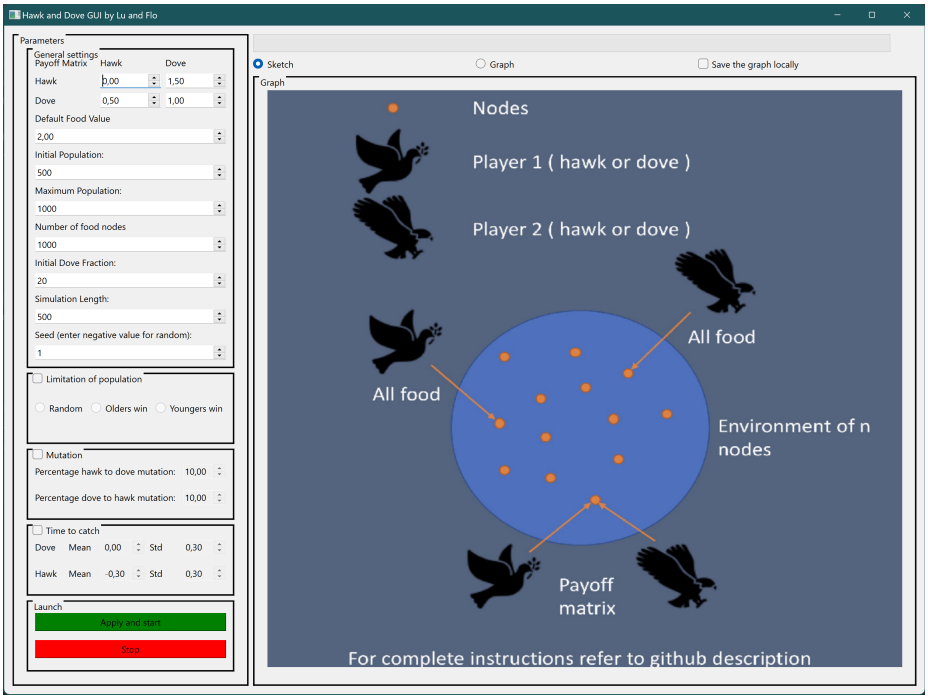


Figure 5: Screenshot of the interface

References

[1] Matt Bognar. *Normal Distribution Applet/Calculator*. Last accessed 26 May 2023. 2021. URL: https://homepage.divms.uiowa.edu/~mbognar/applets/normal.html.

[2] G. R. Maynard Smith J. ; Price. "The Logic of Animal Conflict". In: *Nature* 246.5427 (1973), pp. 15–18. DOI: 10.1038/246015a0.

[3] Primer. *Simulating the Evolution of Aggression*. Last accessed 26 May 2023. 2020. URL: https://www.youtube.com/watch?v=YNMkADpvO4w.

[4] Sabine Hummert; Katrin Bohl ; David Basanta ; Andreas Deutsch ; Sarah Werner ; Günter Theißen ; Anja Schroeter ; Stefan Schuster. "Evolutionary game theory: cells as players". In: *Molecular BioSystems* 10 (2014), pp. 3044–3065. DOI: 10.1039/C3MB70602H.

[5] John Smith. *Evolution and the theory of games*. Cambridge New York: Cambridge University Press, 1982. ISBN: 978-0-521-28884-2.