**MASTER DEGREE PROGRAM IN DATA SCIENCE AND ADVANCED ANALYTICS**





Eva Casal, 20201511

Matilde Parreira, 20201587

Sofia Tátá, 20230546



# 1. Introduction

This report presents a comprehensive approach to solving the Portuguese Football League Scheduling Problem using Genetic Algorithm techniques. Sports league scheduling is a complicated and important topic that affects teams, spectators, broadcasters, and league operations as a whole. A well-planned schedule respects several logistical and legal requirements while guaranteeing fair competition, maximising attendance, and improving the viewing experience.

The primary goal of this project is to evaluate the effectiveness of various selection, crossover, and mutation methods within the GA framework in optimising the league schedule. By exploring different configurations, we aim to achieve an optimal solution that satisfies both hard and soft constraints inherent in the scheduling problem.

Our project consists of seven files, each addressing different facets of the problem:

1. **Data.py**: This file contains the data necessary for the problem, including a randomly generated schedule with defined dates and times. This random schedule is grounded on the idea that stadium and broadcast availability information is predetermined and not a component of our optimization challenge. Our task is to allocate the teams to the available dates. The teams selected for schedule optimization include Sporting CP, Benfica, Porto, Braga, Vitória SC, Moreirense, Arouca, Famalicão, Casa Pia, and Farense. These teams are the subject of this study because they are among the top 10 teams in Portugal's first division.
2. **charles.py**: This file defines the structures for both individuals and populations and serves as the foundation of our implementation. It features the fitness function, essential for assessing the caliber of our timetable, and streamlines the evolutionary processes of crossover, mutation, and selection.
3. **selection.py**: It includes several selection methods to choose the best candidates for reproduction.
4. **xo.py**: This file explores several crossover techniques that facilitate the exchange of genetic material between individuals, thereby generating diverse offsprings.
5. **mutation.py**: Covers several mutation approaches that add variety and avoid premature convergence. They modify the genetic composition of individuals, ensuring a broad search of the solution space.
6. **test.py**: This file is used to test different configurations of selection, crossover, and mutation methods separately. It initialises populations, manages the evolutionary cycle over multiple generations, and records the fitness progression.
7. **ABF\_test\_results.py**: This file was created to test every conceivable configuration of the various approaches using a function. It includes code to plot the results and export an Excel file with all the information required to select the suitable algorithmic setups.

To obtain access to our project code and further details, please visit our GitHub repository:

<https://github.com/sofs-22/Women-s-League->

Throughout the project's execution, our team's cooperative effort guaranteed equitable contributions.

# 

# 2. Methodology

## 2.1. Representation

In this project, we chose an indirect representation: instead of directly displaying the schedule, we represented the teams and then combined them with the actual schedule in a predetermined order. This method provides for more flexible adjustment of teams and matchups, facilitating the use of genetic algorithm techniques.

The **Individual** class is crucial to this portrayal. Each individual in the population represents a possible solution to the scheduling issue. An individual's representation is made up of pairs of teams that are scheduled to play each other, with the first team in each pair being designated as the home team.

## 2.2. Selection

Selection is a crucial step in the evolutionary process, determining which individuals are chosen to pass their genes to the next generation. We implemented several selection techniques, including:

* **Fitness Proportionate Selection (FPS)**: Also known as Roulette Wheel Selection, individuals are selected based on their fitness proportionate probability. Ensuring that individuals with higher fitness scores have a higher chance of being selected.
* **Rank Selection**: Individuals are ranked based on their fitness, and selection is done based on these ranks. This method helps in maintaining diversity in the population by giving a chance to lower-ranked individuals.
* **Tournament Selection**: A subset of individuals is chosen randomly, and the best individual from this subset is selected. This method provides a balance between selection pressure and maintaining diversity.

These methods ensure that individuals with higher fitness have a higher chance of being selected, promoting the propagation of favourable genes.

## 2.3. Crossover

Crossover is the process through which genetic material is exchanged between two parent individuals to generate offspring. It is essential for introducing genetic diversity into the population. We explored several crossover techniques, such as:

* **Single Point Crossover**: A single crossover point is chosen, and the games in the positions before and after this point are exchanged between the parents to produce offspring.
* **Two Point Crossover**: Two crossover points are chosen, and the games between these points are exchanged between the parents to generate offspring.
* **Uniform Crossover**: For each position in the offspring, the game from either parent is selected with equal probability, resulting in a diverse combination of games.
* **Cycle Crossover**: Ensures that each parent contributes equally to the offspring by creating cycles of games that are swapped between the parents, maintaining the relative order of games within cycles.
* **Modified Order Crossover**: Preserves the order of games as much as possible by copying a subset of games from one parent and then filling the remaining positions with games from the other parent, while maintaining their relative order.
* **Subtour Crossover**: Similar to order crossover, but swaps subsequences of games between parents to produce offspring, preserving the order of games within the subsequences.

These techniques facilitate the combination of different genetic traits from the parents, potentially leading to better solutions in the offspring.

## 2.4. Mutation

Mutation introduces variability into the population by randomly altering the genetic material of individuals, we incorporated several mutation techniques, including:

* **Swap Mutation**: Two games are randomly selected, and their positions in the schedule are swapped.
* **Insertion Mutation**: A game is randomly selected and inserted into a different position within the schedule.
* **Inversion Mutation**: A subset of games is selected, and their order within the schedule is reversed.
* **Scramble Mutation**: A subset of games is selected and their positions are shuffled randomly within the schedule.
* **Displacement Mutation**: A subset of games is selected and moved to a different position within the schedule.

These mutation methods ensure a broad search of the solution space by introducing new genetic combinations and maintaining diversity within the population.

## 2.5. Fitness Evaluation

Each individual from the population was assigned a fitness value, representing its ability to solve the problem. We started by designing a simpler fitness function and kept adding constraints in order to increase the complexity of our problem gradually. Our function calculates the fitness considering our soft and hard constraints, giving significantly more weight to the **hard constraints**, the individual can only have a non-negative fitness if it follows all the hard constraints:

* **Check if all teams play together twice (once at home and once away),** deducting 10 points from the fitness if our solution doesn’t follow this because it is mandatory;
* **Check if no team plays more than two consecutive away games,** following the same reasoning as before, minus 10 points due to this constraint being absolutely needed;
* **Check if no team plays more than once in one weekend,** also removing 10 points from the fitness if this condition isn’t met.

**Soft constraints:**

* **Classics should be on prime time**, prime time meaning the game is on a Saturday, this would be a plus in our Portuguese Football League Schedule so we are adding 1 point to the fitness if this is implemented on our solution;
* **Check if the ratio of Saturday games is balanced with Sunday's,** to ensure fairness to all our teams we are calculating the absolute difference between the number of saturday and sunday games for each team and summing these differences, then the points added to the fitness are decided by subtracting that number to 5, so in our best chance (when the number of saturdays and sundays is exactly the same) we add 5 points to our fitness;
* **Ensure no two classic matches occur on the same weekend,** to ensure that classic matches are properly spaced out, we reward the schedule if there is at least a weekend between classic matchups. Specifically, if there is a weekend break between classic matches (more than 8 days apart), we add 2 points to the fitness score. If the next classic match is on the next weekend but not on the same weekend (7 or 8 days apart), we add 1 point to the fitness.

## 2.6. Evolution Process

Moving on to our evolve function, we followed the given Charles Genetic Algorithm library from classes applying elitism, and besides returning the best fitness we also created a list to keep the fitness from every generation to be returned, with the intent of being later used to create some visualisations.

# 3. Tests and Evaluation

To assure error-free performance, we first tested our evolutionary algorithms using a variety of selection, mutation, and crossover procedures in our test file. Following that, in our ABF\_Test\_results file, we conducted a systematic investigation of 270 algorithm combinations (including selection, crossover, mutation, and elitism). To evaluate our models, we employed an evolutionary approach with 200 generations and a population size of 50, running each algorithm 30 times for each combination. This rigorous repetition contributed to the achievement of consistent average and standard deviation performance indicators. The fitness data was aggregated and averaged, then exported to an Excel file hosted on our GitHub repository. Additionally, our team visualised the average fitness trends across generations, categorised by selection, crossover, or mutation strategies. Figures 1, 2, and 3 illustrate these trends. Notably, tournament selection appeared as the best option among selection methods. However, further research is needed to determine the most successful crossover and mutation procedures.

Table 1 summarises our findings, highlighting our top-performing models. Remarkably, we discovered two models with an average fitness score of 9 and a standard deviation of 0, indicating consistent performance. The first model includes elitism, tournament selection, modified order crossover, and swap mutation; the second model excludes elitism but uses tournament selection, cycle crossover, and swap mutation. When choosing the best model, we focused on computational efficiency, concentrating on discrepancies in crossover approaches and the inclusion of elitism. Elitism, while requiring memory to store top individuals, allows for faster convergence by retaining the best individuals throughout the optimization process. Modified order crossover, with its efficient gene subset copying technique, contrasts with cycle crossover, which is more computationally costly and requires sophisticated gene switching processes.

As a result, we determined that our best model combines **elitism, tournament selection, modified order crossover, and swap mutation**. Finally, we used this model in our test.py script to create the final schedule in Figure 4, which is accessible for viewing in the HTML file on our GitHub repository.

# 4. Conclusion

To sum up, this report presents a comprehensive analysis of the Portuguese Football League scheduling problem through the application of genetic algorithm approaches. The group tried to optimise scheduling taking into account competitive and logistical constraints by carefully testing and analysing different configurations. Our results highlight, with the right combination of selection mutation and crossover methods, how well GA’s handle difficult scheduling problems.

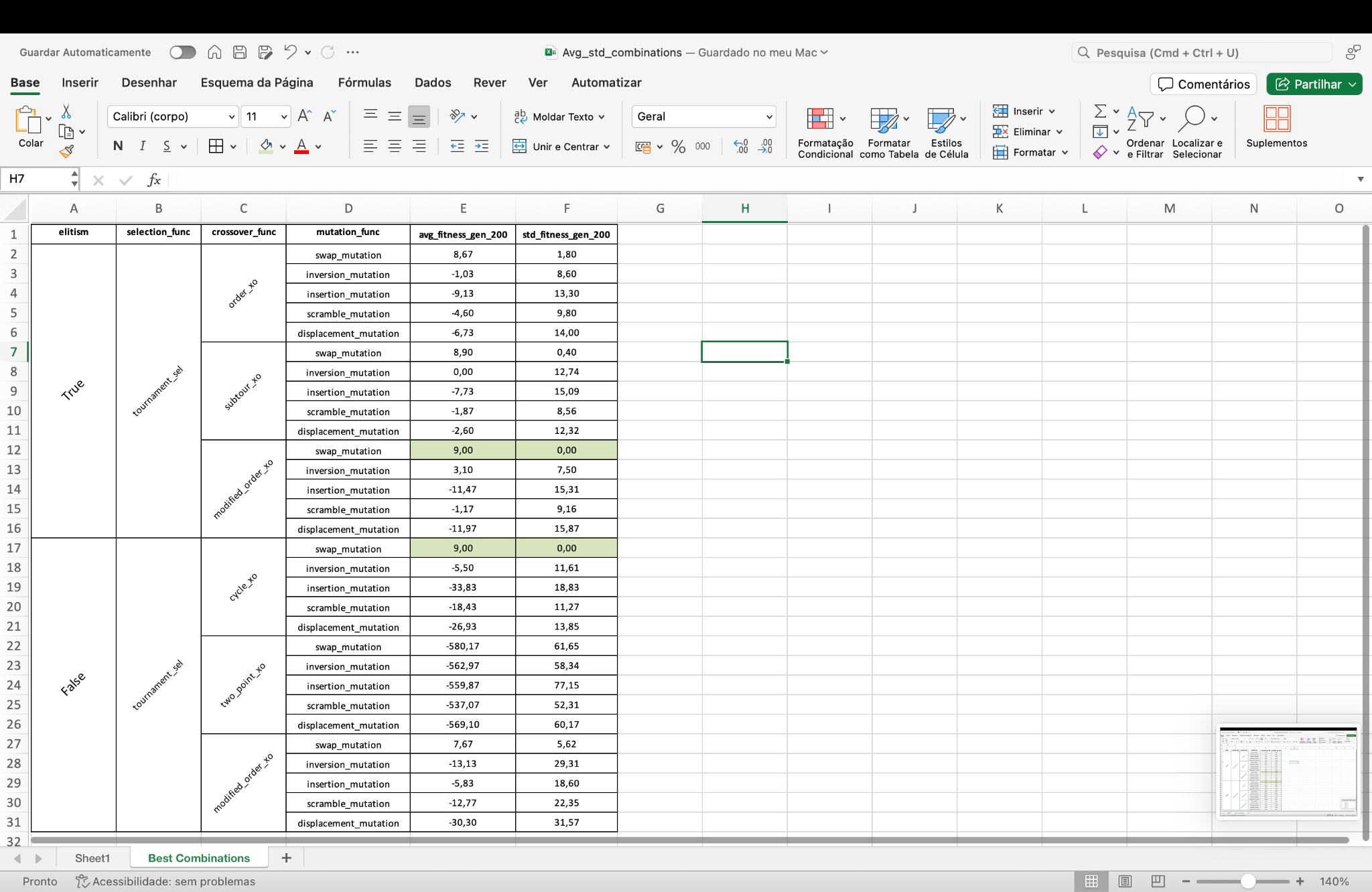
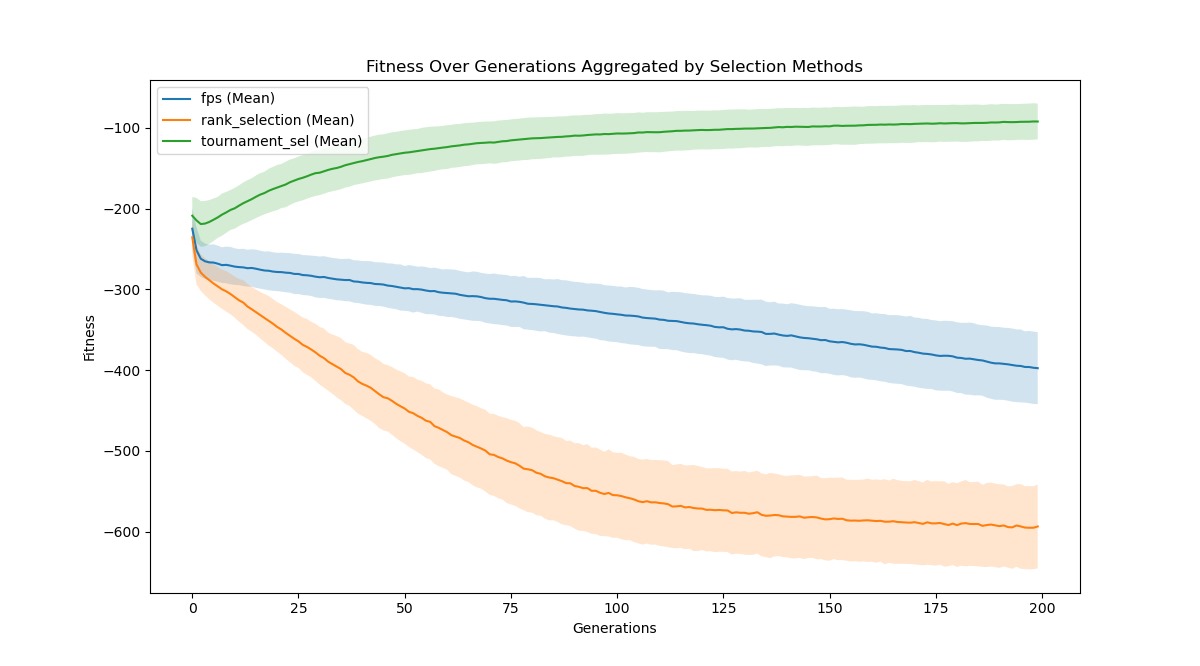
# 5. References

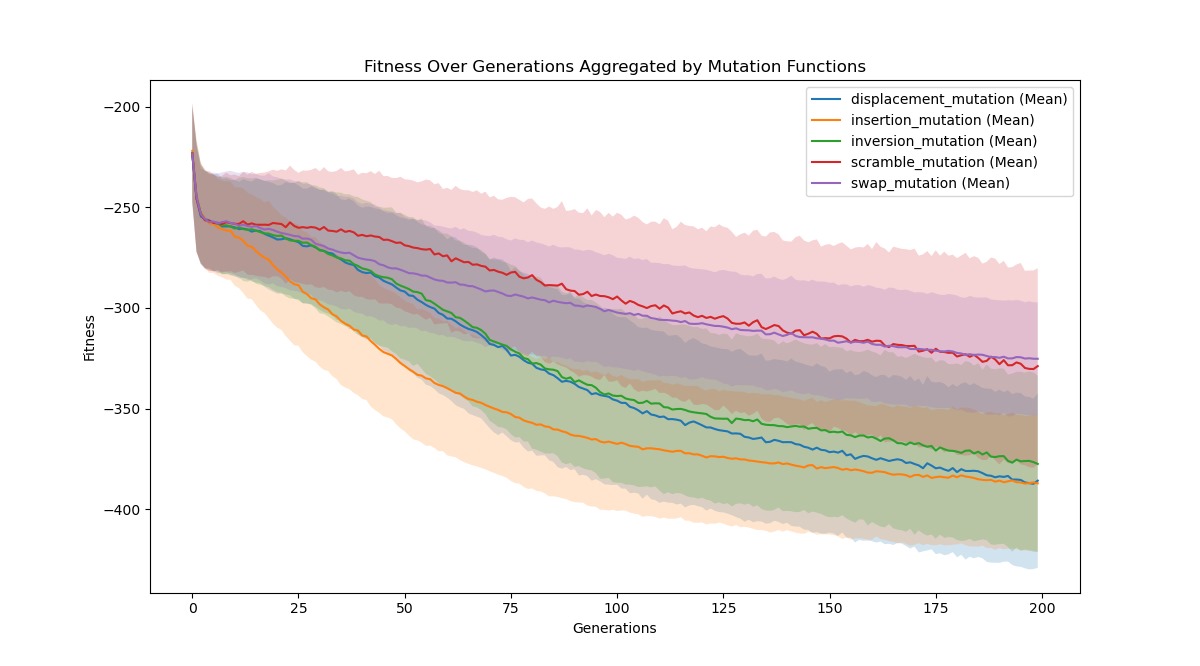
[Vanneschi-Silva-Book.pdf (unl.pt)](https://elearning.novaims.unl.pt/pluginfile.php/219999/mod_resource/content/1/Vanneschi-Silva-Book.pdf)

[Microsoft Word - WCE2012-ICAEM\_116 Camera Ready Fullpaper-Razamin Ramli (iaeng.org)](https://www.iaeng.org/publication/WCE2012/WCE2012_pp95-99.pdf)

[thesis.dvi (wisc.edu)](https://pages.cs.wisc.edu/~goldberg/publications/AndrewBGoldbergThesis.pdf)

# 6. Annex



*Table 1 - small segment of excel export*crossover










*Figure 4 - Final Schedule*