FEDERAL UNIVERSITY OF SANTA CATARINA
JOINVILLE TECHNOLOGICAL CENTER
AUTOMOTIVE ENGINEERING UNDERGRADUATE PROGRAM

PEDRO HENRIQUE STELLA COSTA

# GENETIC AND PARTICLE SWARM ALGORITHM APPLICATION TO TIRE MODEL FITTING

Joinville
2022

PEDRO HENRIQUE STELLA COSTA

# GENETIC AND PARTICLE SWARM ALGORITHM APPLICATION TO TIRE MODEL FITTING

Graduation Thesis submitted to the Automotive Engineering Undergraduate Program of Federal University of Santa Catarina to obtain the title of Bachelor in Automotive Engineering.
Advisor: Prof. Andrea Piga Carboni, Dr.
Co-Advisor: Claude Rouelle, MSc.

Joinville
2022

PEDRO HENRIQUE STELLA COSTA

**GENETIC AND PARTICLE SWARM ALGORITHM APPLICATION TO TIRE MODEL FITTING**

The present work at Graduation level was judged adequate to obtain of the title of Bachelor in Automotive Engineering and was approved in its final form by the Automotive Engineering Undergraduate Program of Federal University of Santa Catarina.

Examination committee:

Prof. Andrea Piga Carboni, Dr.
Committee President

Prof. Thiago Antonio Fiorentin, Dr.
Federal University of Santa Catarina

Girish Radhakrishnan, Master.
OptimumG LLC

Joinville, 2022.

## ACKNOWLEDGEMENTS

This work is dedicated to my family, Aline, José and Enzo, as well as my grandparents and uncles Jean and Wagner, for providing and taking care of me during the graduation so I could focus only on my course and projects. Not many people has a supportive family like I do, so I'm really grateful for them. Also, I would like to thank my professor Andrea and good friend Ariel for always giving support, ideas and for inviting me into the VOD research group, that presented me the world of optimization. I also would like to thank the people from my sub-sector of formula SAE team, Bruno Serpa, Paulo, Gustavo, André, Renan (DK), Caio and all the other friends I made on the team that made those 2 years a really great experience. Lastly, I would like to thank Claude Rouelle for the multiple opportunities to learn and grow not only as a professional but as a person as well.

## ABSTRACT

The tire is the primary source of forces and torques that provide control and stability to the vehicle. Thus, the performance of a vehicle is mainly influenced by the characteristics of its tires. It is vital that the automotive engineer has a mathematical/ computational tool that provides precision and consistency to model a tire. The Magic Formula is the main model currently in use in the automotive industry. The fitting process of the Magic Formula model is a complex task and can be treated as a optimization problem. For this reason a particle swarm and genetic algorithm are implemented. A benchmark and comparison is made between these two algorithms, for standard test functions and fitting of a Magic Formula 6.1 model.



**Keywords**: Tire 1. PSO 2. Genetic Algorithm 3.

## RESUMO

É vital que o engenheiro automotivo tenha uma ferramenta matemática/computacional que forneça precisão e consistência para modelar um pneu. A Fórmula Mágica é o principal modelo atualmente em uso na indústria automotiva. O processo de ajuste do modelo Magic Formula é uma tarefa complexa e pode ser tratada como um problema de otimização. Por esta razão, um enxame de partículas e um algoritmo genético são implementados. Um benchmark e uma comparação são feitos entre esses dois algoritmos, para funções de teste padrão e ajuste de um modelo Magic Formula 6.1.



**Palavras-chave**: Pneu 1. PSO 2. Algoritimo Genético 3.

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF SYMBOLS

| | |
|---|---|
| $SR$ | Slip Ratio |
| $\Omega$ | Angular velocity of the drive wheel |
| $\Omega_0$ | Angular velocity of the free-rolling wheel |
| $D$ | peak value |
| $C$ | shape factor |
| $B$ | stiffness factor |
| $E$ | curvature factor |
| $S_V$ | vertical shift |
| $S_H$ | horizontal shift |
| $F_x$ | Longitudinal force |
| $F_y$ | Lateral force |
| $M_z$ | Aligning torque |
| $\alpha$ | Slip angle |
| $\kappa$ | longitudinal wheel slip |
| $V_{i+1}$ | Particle's next iteration speed |
| $W$ | Particle's inertia |
| $V_i$ | Particle's current speed |
| $f_c$ | cognitive factor |
| $p_{best}$ | Particle's personal best position |
| $X_i$ | Particle's current position |
| $f_s$ | social factor |
| $g_{best}$ | Swarm's global best position |
| $X_{i+1}$ | Particle's next iteration position |
| $m_{best}$ | Swarm's global average best position |

**CONTENTS**

# 1 INTRODUCTION

According to W.F. Milliken and D.L. Milliken (1995) the tire is the primary source of forces and torques that provide control and stability to the vehicle. Also as stated by Jazar (2013) the tire is the main component interacting with the road; thus, the performance of a vehicle is mainly influenced by the characteristics of its tires, they affect handling, traction, ride comfort, and fuel consumption. For this reason, the tire is one of the most important components of an automobile. Taking this into account, it is vital that the automotive engineer has a mathematical/ computational model that provides precision and consistency in the calculations made during the project, such as simulations of vehicle dynamics. That said, other applications include the virtual prototyping of a tire, which helps to reduce the costs for tire manufacturers by reducing the multiple iterations of physical prototypes, and even, the use for a realistic tire representation on video games.

Currently, there are multiple tire models, either steady-state or transient systems, where there are two main ways of addressing the problem. The theoretical form, with a physical model, or the empirical form that comes directly from experimental data. Pacejka (2006) proposed a semi-empirical tire model that can accurately calculate lateral force and self-aligning torque. This study has been improved in new versions over the years and has become known as the “Magic Formula”, the last one being the Magic Formula version 6.2. The model implies in a treatment of the experimental raw data of each tire by a semi-empirical curve fitting method that provide the value of forces and moments generated, given as input a normal force and parameters such as pressure, inclination and slip angle. The relative simplicity of the equations, grants the possibility of calculating the outputs in real time and also not having to rely on data interpolation, resulting in an advantage over the use of experimental data by itself for vehicle dynamics simulations.

However according to W.F. Milliken and D.L. Milliken (1995), the parameters (constants) of the fit are difficult to obtain and require cross plotting to obtain effects of more than one variable. So, the definition of the Magic Formula parameters is a task that must be performed with the aid of computational tools, as they are arbitrary factors that vary for each tire test case. In this way, given the non-linearity of the problem and the potential local minima, we can approach the curve fitting as an optimization problem where several already establish algorithms can be used.

The evolutionary algorithms are search techniques inspired by the biological evolution of the species (DRÉO et al., 2006). This type of operation allows the application of multiple methods ensuring versatility in solving problems. An implementation of such techniques is the genetic algorithm (GA), that is a metaheuristic based on the process of natural selection, which takes a population of individuals and separates it

taking into account their fitness to an objective function. The best performing individuals reproduce or mutate into off-springs generating a new population, repeating the cycle until a desired result is found.

Another type of metaheuristic is the Particle Swarm Optimization (PSO) algorithm is a population-based evolutionary computation technique. It is motivated by the behavior of organisms such as fishing schooling and bird flock. In a PSO system, each particle corresponding to individual of the organism is a candidate solution to the problem at hand. Particles of the population fly around in a multi-dimensional search space, to find out an optimal or sub-optimal solution by competition as well as by cooperation among them. (SUN et al., 2004)

Due to the high amount of tires used in motorsport, and the high amount of test needed to model those tires, a fast and reliable tool is essential for the tire engineer. Therefore, this work proposes the implementation of both algorithms, to allow the user to o obtain the MF coefficients with efficiency and consistency, thus, the tire model based on the input of raw test data. Then, a comparison of both techniques is analyzed, for performance in convergence rate, precision and robustness. First, a theoretical foundation is given to the reader as an introductory view into the multiple fronts of knowledge this work addresses, later, the methods used to achieve the goals presented are explained, and a detailed view of the algorithms implementation is given. Then the results obtained when those algorithms are used for test and tire fitting functions are demonstrated and commented.

## 1.1 OBJECTIVES

The objectives of this work are explained in the followings sections.

### 1.1.1 Main objective

Implement and compare a particle swarm and genetic algorithm for tire fitting purpose.

### 1.1.2 Specific objectives

- Write and implement a genetic algorithm to fit an MF6.1.2 tire model,
- Write and implement a particle swarm optimization algorithm to fit an MF6.1.2 tire model,
- Compare and validate both optimization algorithms in terms of speed and result accuracy,
- Determine the most apt algorithm for tire model fitting purposes.

# 2 THEORETICAL FOUNDATION

This chapter presents a foundation on the topics covered in this work, granting a better understanding by explaining the main concepts and why or how they were used to achieve the results obtained.

First, the tires fundamentals in a automotive context are explained to give a broad vision on the complexity of the subject and the motivation to study said object. Later, tire modeling is explained comparing the existing types of approaches. Lastly the optimization problem is discussed and a focus is given on the particle swarm and genetic algorithm method.

## 2.1 TIRE FUNDAMENTALS

Tires are the primary source of the forces and torques which provide grip, balance and also, the control and stability (or "handling") of the vehicle (OPTIMUMG, 2018). Since tires are the only component of a vehicle which is in contact with the ground, all the forces required to move it, being braking, accelerating or cornering, are supplied from the tire/ road interaction. Furthermore, according to W.F. Milliken and D.L. Milliken (1995), the tires also supply the forces used for controlling and stabilizing the vehicle and for resisting external disturbances from road and wind.

A tire is an advanced engineering product made of rubber and a series of synthetic materials bound together. Fiber, textile, and steel cords are some of the components that go into the tire's inner liner, body plies, bead bundle, belts, sidewalls, and tread (JAZAR, 2013). In Figure 1 it is possible to see the different tire interior components and their arrangement.

The different components can be explained as stated by Jazar (2013):

- Bead bundle: is a high strength steel cable to give the tire the strength to seat on the wheel rim and to transfer the tire forces to the rim.
- Inner layers: they are made from different kind of fabrics called plies. usually made from polyester the material purpose is to keep the assembly together.
- Inner liner: is the rubber that forms the inside of a tubeless tire, it maintain the air pressure.
- Belts: are one or more rubber-coated layers of steel, polyester, nylon, Kevlar or other materials running circumferentially around the tire under the tread. They are designed to reinforce body plies to hold the tread flat on the road and make the best contact with the road.

- Carcass: are the main part in supporting the tension forces generated by tire air pressure. The carcass is made of rubber-coated steel or other high strength cords tied to bead bundles.
- Sidewall: provides lateral stability for the tire, protects the body plies, and helps to keep the air from escaping from the tire.
- Tread: is the portion that comes in contact with the road. Their design vary widely on the purpose of the tire and are made from different kinds of natural and synthetic rubbers.

Although the great importance of the tire in the dynamic behavior of the vehicle, it is also too complex due to its construction and chemical compound dependency, and sometimes is better understood when isolated and explained separately. Tire forces and moments can be represented as vectors and magnitudes on the tire/ road contact interface, this representations are separated according to an established coordinate system. An overview of the SAE system can be seen in the Figure 2.

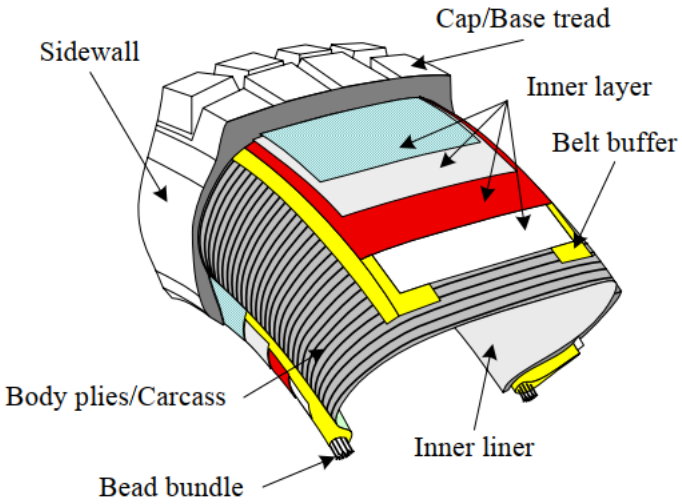
### 2.1.1 Tire forces and moments

Tire forces or grip are generated through the friction obtained by the contact between tire and road. This friction can be generated in three different ways according to (SEGERS, 2019):

- Indentation: Due to the slippage of the rubber over the rough road surface, the strike against this rough spots generates an asymmetrical deformation due to hysteresis effects. This in the other hand creates a force with a horizontal component which resists slippage. (SEGERS, 2019)
- Molecular adhesion: Adhesion results from molecular interactions on the ground/ rubber interface. After the molecular bound is created between the two surfaces, this molecular chain is stretched resulting in a force opposing slippage.
- Wear: According to Segers (2019), at higher deformation forces and higher slippage speeds local stress in the rubber can exceed the material's tensile strength, especially near the point of a sharp irregularity. When the rubber is deformed past the point of elastic recovery, tearing is caused. Tearing absorbs energy, resulting in an additional friction force opposing slippage.

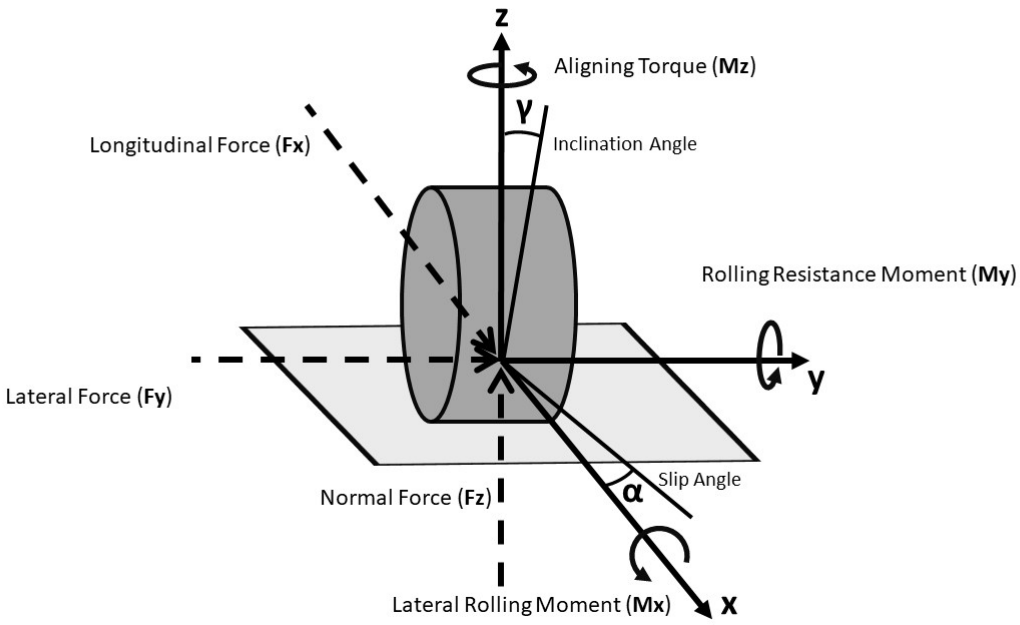
This forces can be separated and nominated according to its effect on the vehicle dynamic behavior and are going to be explained in the following subsections.

Figure 1 – Diagram of tire components




Source – Jazar (2013).

Figure 2 – Tire forces and coordinates diagram (SAE)



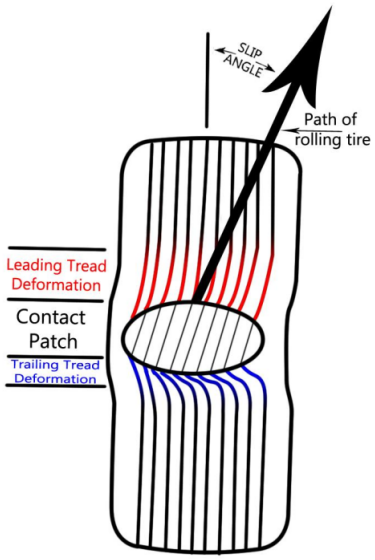

Source – Author (2021).

#### 2.1.1.1 Lateral force (Fy)

According to W.F. Milliken and D.L. Milliken (1995) a lateral tire force originates at the center of the tire contact with the road, lies in the horizontal road plane and is perpendicular to the direction in which the wheel is headed. This force is responsible to the vehicle steering and represents the tire capability to resist lateral acceleration.

It can be generated by or generate a slip angle. It is also influenced by the tire inclination angle and the normal force (Fz). According to Megaride (2021), the slip angle is defined by the angle between the forward speed and the equatorial plane of the tire. It is achieved by the contact patch deformation. A visualization of the slip angle can be seen on Figure 3

Figure 3 – Slip angle visualization diagram




Source – Megaride (2021).

The slope of the curve of the lateral force by slip angle is defined as cornering stiffness, $C_\alpha$, and it is related to the tire capability to deform laterally. The higher the cornering stiffness, less slip angle is required to produce the same amount of lateral force. The behavior of the tire in the scope of lateral force by slip angle curve, is presented with a linear region, reaching a peak of lateral force that is surrounded by a non linear region. Figure 4 shows an example for a racing tire, where is possible to see an elastic or linear behavior for lower slip angles, and the non linear region coming to a peak at higher values of slip angle.

Figure 4 – Slip angle visualization diagram

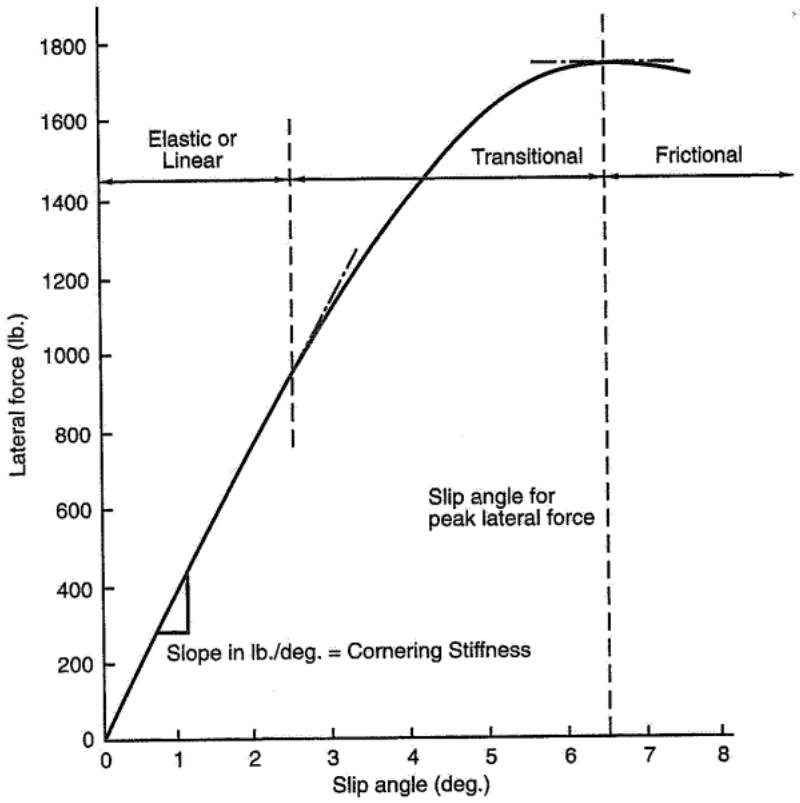



***Figure 2.7** Lateral force vs. slip angle for a racing tire.*

Source – W.F. Milliken and D.L. Milliken (1995)

#### 2.1.1.2 Longitudinal force (Fx)

In order to accelerate or brake a vehicle, longitudinal forces must be developed between the tires and the ground, in the tire footprints (MILLIKEN, W.; MILLIKEN, D., 1995). The longitudinal forces also depend on the normal force (Fz) and inclination angle but in this case a slip ratio is responsible to the generation of this component.

SAE (2008) defines slip ratio, *SR*, as the difference between the angular velocity of the driven or braked wheel, $\Omega$, and the angular velocity of the free-rolling wheel, $\Omega_0$, representing as a fraction being as Equation (1)

$$SR = \frac{\Omega - \Omega_0}{\Omega_0} \tag{1}$$

Being, *SR* = 0 free rolling, *SR* = -1 for locked breaking and *SR* = 1 for a spinning wheel. As the slip ratio increases (numerically) from zero, the forces rise rapidly to a maximum which usually occurs in the range of 0.10 to 0.15 slip ratio, after which the forces fall off. Up to the peak the forces depend heavily on the elastic properties of the

tread and carcass. After the peak the forces depend on a variety of factors such as tread composition, road texture, surface moisture, speed, tire temperature, etc. (MILLIKEN, W.; MILLIKEN, D., 1995).

#### 2.1.1.3 Aligning Torque (Mz)

SAE (2008) describes the aligning torque as a tire's tendency to steer about a vertical axis through the center of the print (the origin of the tire axis system). W.F. Milliken and D.L. Milliken (1995) states that between low and medium slip angles the tires tents to align its heading with its path. In other words, tires tends to point the way they are going.

The aligning torque is created from the deformation of the tire print. The elastic distortion increases from front to back and this gives an uneven distribution of lateral force in the length of the print, thus giving rise to the aligning torque. The Aligning torque (Mz) is an important moment generated on the tire since it is one of the main factors that gives the driver a feedback on the steering of the car.

### 2.1.2 Other tire effects

The previous sections were focused on tire forces and moments, but there are other effects that affect tire dynamic behavior and they are important to the understanding of this work.

#### 2.1.2.1 Tire pressure

The pressure that the pneumatic tire is inflated have a significant effect on tire behavior. Higher tire pressure causes higher tire stiffness and thus a higher cornering stiffness. So, as stated before, less slip angle is needed for the same amount of lateral force generated. This effect also translates to the aligning torque, since lower cornering stiffness represents a higher steering effort and an increase in the aligning torque due to the contact patch size. Lowering the tire pressure increases the size of the contact patch resulting in more friction, according to W.F. Milliken and D.L. Milliken (1995) this results in a higher peak of lateral and longitudinal force. But a balance is required since it also increases the tire wear and drag. According to Segers (2019) other parameters may be influenced by tire pressure, such as:

- Dynamic ride-height witch influence in the aerodynamic behavior of the vehicle;
- Tire vertical spring rate, affecting the non suspended mass natural frequency;
- And the tire temperature build up over time.

#### 2.1.2.2 Tire temperature

Tire temperature affects both the force-producing capability of the tire and also the life of the tire (MILLIKEN, W.; MILLIKEN, D., 1995). Tire temperature is measured in different locations. It can be referred to the tread temperature, inside the carcass and the inside air temperature, where all of them presents different effects in tire behavior. Tire temperature is an important effect of the dynamic behaviour and it can affect the vehicle in many ways, but is not approached in this work due to its complexity to model.

## 2.2 TIRE MODELING

In the last section, the importance the tire has over the vehicle dynamic behaviour was introduced. Tire modeling is present in the automotive industry for years and according to Megaride (2021) the main reasons are:

- Decreases the number of iterations of design and manufacture by generating a better understanding of the behaviour even before the manufacture process.
- Decreases the time and money invested in the design, manufacture and vehicle dynamics modeling processes by the same reasons mentioned above.
- Possibilities of generating "what if" scenarios, meaning a possibility of extrapolating the physics to give a better grasp on the behaviour of the tire.
- Increases computational efficiency, by implementing a model it allows the analysis of vehicle dynamics without the need of an interpolation table that requires tire testing for each tire analysed.

### 2.2.1 Model types

Several types of mathematical models of the tire have been developed during the last half century. Each type has a specific purpose. Different levels of accuracy and complexity may be introduced in the various applications. This often involves entirely different ways of approach (PACEJKA, 2006). Megaride (2021) describes different types of tire modeling, that are presented in the following subsections.

#### 2.2.1.1 Empirical or curve fit model

A empirical or curve fit model is based on a "grey-box" approach on fitting the tire test data to a mathematical model that may have some basing on a physics understanding, but it is mainly based on mathematical methods. This type of model is widely used in the industry for its great accuracy and the lower need for computational power. But it has the disadvantage of being limited by the test range, and the model parameters not having a direct meaning.

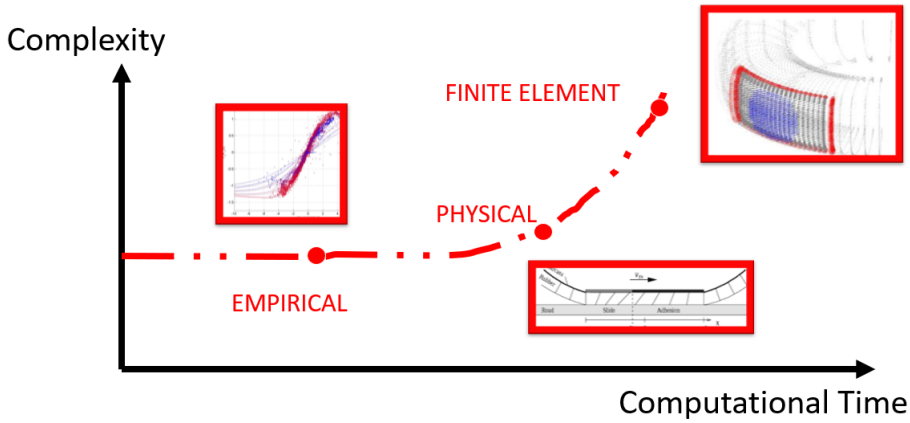
#### 2.2.1.2 Physical model

The physical tire model is more complex than the the empirical model since it uses only physical methods to model the tire. It is still relatively fast to obtain results, and presents great accuracy for regions that the empirical model may not model with precision, but it is limited to the assumptions made on the physical approach. Also some parameters needed to model may be difficult to measure or obtain, such as precise composite composition factors, tire belt stiffness, etc.

#### 2.2.1.3 Finite element model

The finite element tire model presents the higher complexity, and has a realist representation of the whole tire by modeling material properties, compounds, assembly, etc. Since it has the higher computational time and higher accuracy, this type of model is used for tire design. In the following Figure 5, we can see the relationship of model type and computational cost and complexity.

Figure 5 – Model type complexity and complexity relationship




Source – Author (2021)

### 2.2.2 Magic formula

A semi-empirical tire model widely used to calculate steady-state tire force and moment characteristics for use in vehicle dynamics studies is based on the so-called Magic Formula. The development of the model was started in the mid-eighties, since then, TU-Delft and Volvo developed several versions. The combined slip scenario was modeled from a physical point of view in the MF. In 1993 Michelin introduced a purely empirical method using Magic Formula based functions to describe the tyre horizontal force generation at combined slip (PACEJKA, 2006). There are several iterations of

the Magic Formula (MF) tire model, the ones used for this work are based on the MF 5.2 and the MF 6.1 that is capable of modeling tire pressure effects. It is considered a semi-empirical tire model since it has some physical foundation, but the main objective is to fit the test data in a series of equations developed to better represent the tire behavior. The general form of the formula by Pacejka (2006) that holds for given values of vertical load and camber angle reads:

$$y = Dsin[Carctan(Bx - E(Bx - arctanBx))] \tag{2}$$

with

$$Y(X) = y(x) + S_V \tag{3}$$

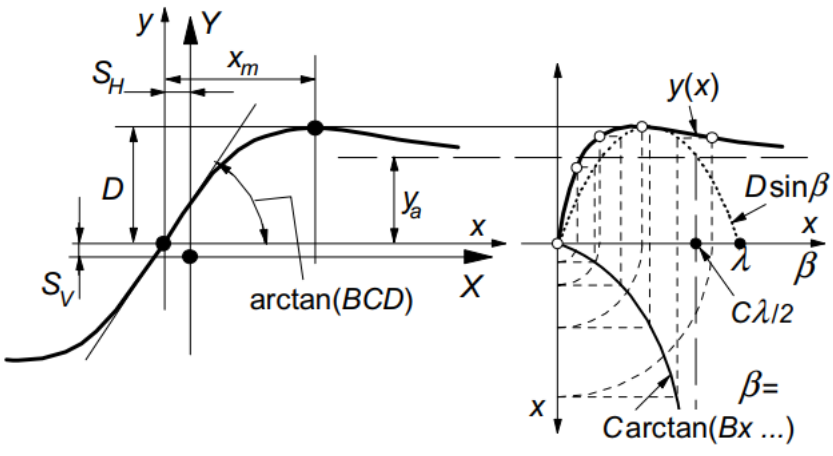
$$x = X + S_H \tag{4}$$

being:

- Y: output variable $F_x$, $F_y$ or possibly $M_z$
- X: input variable tan$\alpha$ or $\kappa$ where $\alpha$ is slip angle and $\kappa$ is slip ratio.

The Magic Formula y(x) typically produces a curve that passes through the origin x = y = 0, reaches a maximum and subsequently tends to a horizontal asymptote. For given values of the coefficients B, C, D and E the curve shows an anti-symmetric shape with respect to the origin. To allow the curve to have an offset with respect to the origin, two shifts $S_H$ and $S_V$ have been introduced (PACEJKA, 2006). Figure 6 shows the curve produced by the original formula, Equation (2).

Figure 6 – MF original curve and its parameter's meaning indicated




Source – Pacejka (2006)

The formula is capable of reproducing characteristics that matches the curves measured from test data of $F_y$, $F_x$ and $M_z$ as functions of their respected slip quantities ( slip angle $\alpha$ and longitudinal slip $\kappa$).The shift factors presented before models the offset effect given by the inclination angle and normal load. A sine function is assigned for each force curve to better capture its peculiarities and to better fit the data, so each force has multiple parameters to be fit. The complete MF 6.1 equations can be found in the book "Tyre and vehicle dynamics" by Pacejka (2012).

## 2.3 CURVE FITTING

Curve fitting is the process of constructing a curve, or mathematical function, that has the best fit to a series of data points. Curve fitting can involve either interpolation, where an exact fit to the data is required, or smoothing, in which a "smooth" function is constructed that approximately fits the data. The Magic Formula takes the smoothing approach by modeling the tire forces that approximate the model behavior to the behavior obtained from the test data. Fitted curves can be used as an aid for data visualization, to infer values of a function where no data are available, and to summarize the relationships among two or more variables. Extrapolation is the use of a fitted curve beyond the range of the observed data, and is subject to a degree of uncertainty since it may reflect the method used to construct the curve as much as it reflects the observed data. This is also one of the uses of the Magic Formula, where the tire behavior can be characterized outside of the testing range. The process of fitting the Magic Formula curves to the tire test data can be treated as an optimization problem where the goal is to minimize the error between the model and the equivalent data points.

## 2.4 OPTIMIZATION

Everyday, engineers and the decision makers are confronted with problems of growing complexity, which emerge in diverse technical sectors, such as in-operations research, design of mechanical systems, image processing, and electronics. The problem to be solved can be often expressed as an optimization problem. Here one can define an (or several) objective function, or cost function, that is sought to be minimized or maximized for all the parameters concerned. The optimization problem is often supplemented by the information of constraints. All the parameters of the adopted solutions must satisfy these constraints, or otherwise these solutions are not realizable (DRÉO et al., 2006). As stated before the Magic Formula model requires the fitting of the equations parameters to better represent the test data. This fitting can be achieved by a number of ways, but since the magic formula model requires a lot of parameters without direct meaning to be fit, the problem was treated as an optimization problem where the goal is to find the coefficients which generates the model that have the least error from

the measured data. Thus, the genetic and particle swarm algorithms can be used to tackle this problem.

### 2.4.1 Genetic algorithm

Dréo et al. (2006) states that the evolutionary algorithms (EA) are the optimization techniques inspired by the biological evolution of the species. One of this techniques is the genetic algorithm (GA). The principle of a genetic algorithm can be described as follows. A set of N points in a search space, chosen a priory at random, constitutes the initial population; each individual x of the population has a certain fitness value, which measures its degree of adaptation to the objective aimed. In the case of the minimization of an objective function z, the fitness of x will be higher, if z(x) is smaller. An EA consists in evolving gradually, in successive generations, the composition of the population, by maintaining its size constant. During generations, the objective is to overall improve the fitness of the individuals; such a result is obtained by simulating the two principal mechanisms which govern the evolution of the living beings (DRÉO et al., 2006). According to the theory of Darwin (1859) those mechanisms are:

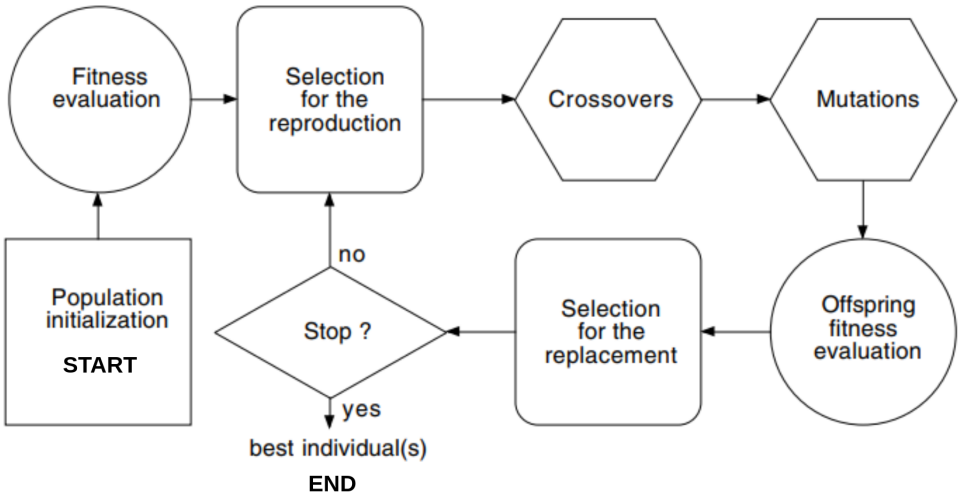
- Selection, which supports the reproduction and the survival of the fittest individuals,
- Reproduction, which allows mixing, the recombination and the variations of the hereditary features of the parents, to form offspring with new potentialities.

In Figure 7 we can visualize the genetic algorithm block diagram to get a better grasp of its workings. Dréo et al. (2006) also states that, since the GA handle a population of solution instances, the genetic algorithm are particularly indicated to propose a set of multiple solutions, when a objective function comprises several global optima. This can be applied to the MF fitting problem, since multiples solutions of different parameters are possible.

The GA uses several mechanisms to control and maintain the individual's population, they are known as operators. During each generation, a succession of operators is applied to the individuals of a population to generate the new population for the next generation.

When one or more individuals are used by an operator, they are labeled as parents, that will generate new individuals for the next generation which on its turn are labeled as offspring. Thus, when two operators are applied successively, the offspring generated by one can become parents for the other (DRÉO et al., 2006). There are different implementations for each operator that affects greatly the algorithm behavior. The two main operators types are the selection and variation operators. They will be explained briefly in the sections to follow.´

Figure 7 – Principle of a genetic algorithm




Source – Dréo et al. (2006)

#### 2.4.1.1 Selection operators

According to Dréo et al. (2006) on every generation, the individuals reproduce, survive or disappear from the population under action of two selection operators:

- The selection for reproduction, witch determines if and how many times an individual will reproduce in a generation;
- The selection for replacement, that determines which individuals will disappear from the population in each generation so that, the population size remains constant throughout the evolution process. It's important to note that the individual can be selected by both operators at the same time.

In accordance with Darwin (1859), the better or more fit an individual is, the more often it is selected to reproduce or survive. To make a selection possible, a fitness value, which depends on the objective function (function being optimized), must be attached to each individual. In a curve fitting case, the objective function can be the error value between the curve and the data points, so each time that an individual is evaluated, the error between the curve that it generates is evaluated with the data points. This, however, implies that, in each generation, all the individuals are evaluated against a determined goal or objective, which can be computational intensive.

#### 2.4.1.2 Variation operators

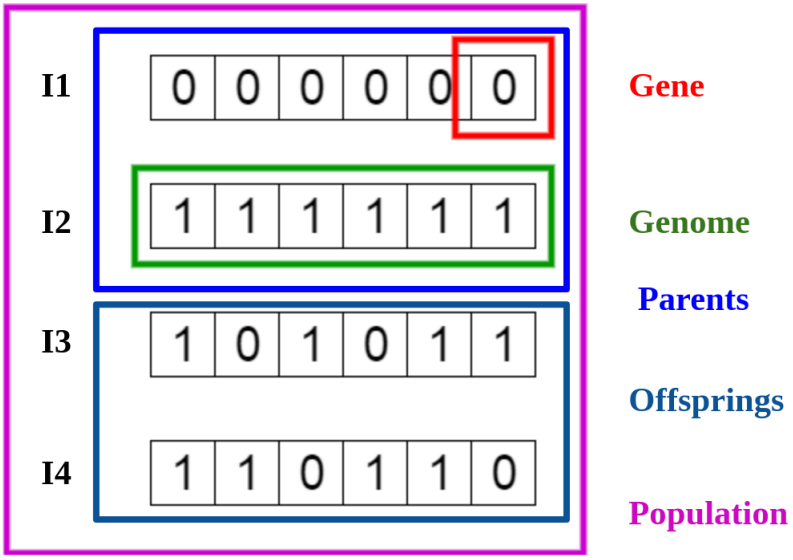
Dréo et al. (2006) says that in order for the algorithm to find better solutions than those present in the current population, it is required that they are transformed by the

application of variation operators. A large variety of them can be imagined. They are classified into two categories:

- The mutation operators, which modify an individual to form another;
- The crossover operators, which generate one or more offspring from combinations of two or more parents. The designations of these operators are based on the real life concept of the sexual reproduction of the living beings, but without the constrain of a real life limitation to two parents for each offspring.

In Figure 8 a simple example is demonstrated. Each individual is represented by the number container, that represents a genome, each genome is composed by multiple genes. The parents can be seen, for simplicity sake, in the evenly distributed containers, and it's offspring on the following containers. Figure 9 shows an example for the evaluation process. The genome of an individual is taken and its information is used to generate a curve, then this curve is compared with the data points and the error value is used as the individual fitness value.

Figure 8 – Simple demonstration of the different GA organization layers.

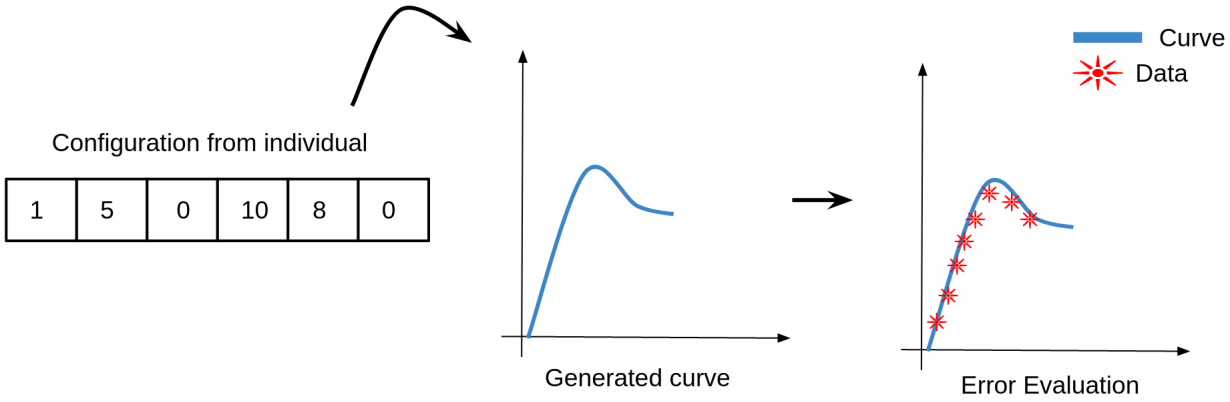



Source – **Author (2022)**

### 2.4.2 Particle Swarm Optimization

The particle swarm optimization (PSO) evolved from an analogy drawn with the collective behavior of the animal displacements (in fact,the metaphor was largely derived from socio-psychology). Indeed, for certain groups of animals, e.g. the fish schools, the dynamic behavior in relatively complex displacements can be observed, where the

Figure 9 – Example of GA evaluation step for a curve fitting application.

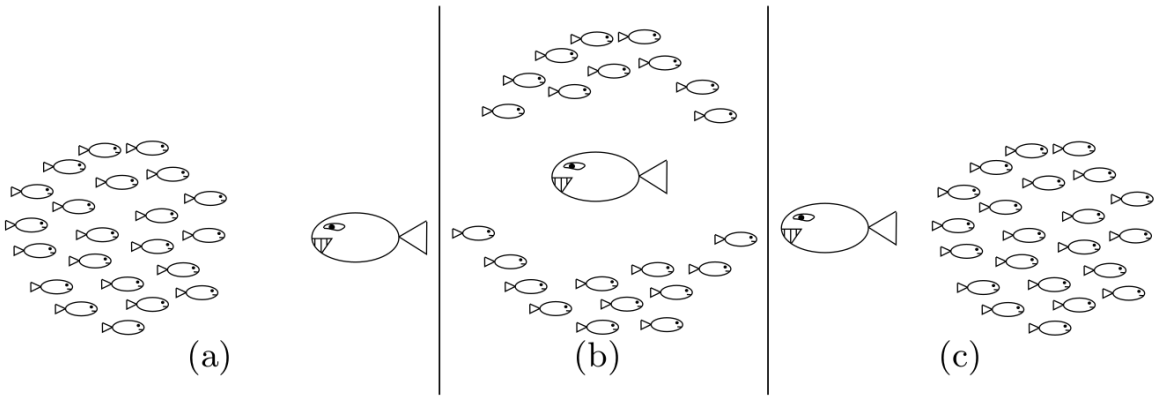



Source – **Author (2022)**

individuals themselves have access only to limited information, like the position and the speed of their closer neighbors. (DRÉO et al., 2006).

An example can be made in a fish school, which is able to avoid a predator by initially dividing in two groups then reforming the original school while maintaining cohesion seen in figure Figure 10.

Figure 10 – Fish school example




Source – Dréo et al. (2006)

According to Dréo et al. (2006) these collective behaviors completely conform to the theory of self-organization. In this theory, each individual uses the local information regarding the movement of his closer neighbors, which are reachable by him, to decide on his own movement.

Simple rules like “remain close to the other individuals”, “go in the same direction”, “at same speed” etc. are good enough to maintain the cohesion among the entire group, and to allow complex and adaptive collective behaviors.

This behavior can be achieved by a group of "particles" that have a position in the solution space that is described by it’s configuration. Each particle has a velocity that will update the particle’s configuration (position) in the next iteration, and is linked with two main factors that represent both the particle’s individualism and it’s social dependency. The speed and position equation that can be used to represent the swarm behavior is expressed in equation Equation (5) and Equation (6).

$$V_{i+1} = WV_i + f_c\rho_1(p_{best} - X_i) + f_s\rho_2(g_{best} - X_i) \tag{5}$$

$$X_{i+1} = X_i + V_{i+1} \tag{6}$$

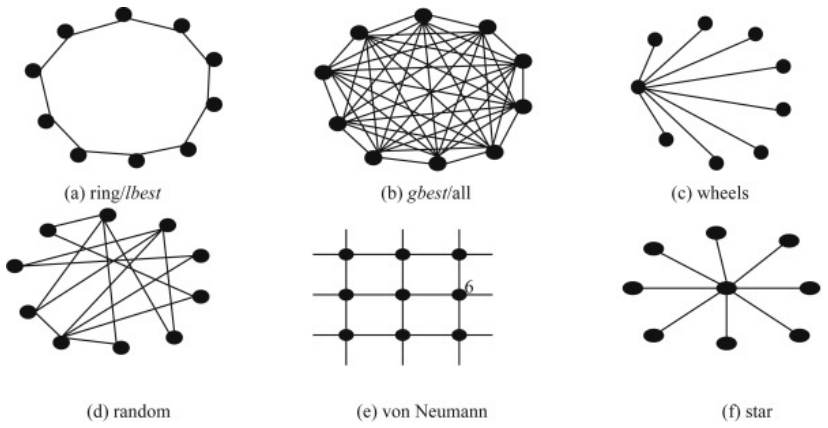
Being $V_{i+1}$ the particle speed in generation i + 1. The social factor ($f_s$) represents the particle’s tendency of following other particles. The cognitive factor ($f_c$) is the particle’s independence in searching the solution based on it’s own findings, and the inertia ($W$) is the particle’s tendency to reduce it’s speed for each iteration during the search process. $\rho_1$ and $\rho_2$ are randomly generated numbers, $p_{best}$ and $g_{best}$ are the particle’s personal best position and the connected particles’ best position respectively.

The connected particles refers to the particles that are in direct relation with the reference particle. This relation can be established in a variety of ways and the rules that determines which particle are connected to which is called topology. The most simple one being that every particle is connected with each other, where $g_{best}$ is the best overall swarm’s position. Another example of topology is the ring topology, where each particle is connected only with the two closest particles, in this case $g_{best}$ is the best position achieved by only the two closest particles. It’s possible to get an overall view of the many types of topology in Figure 11.

#### 2.4.2.1 Quantum particle swarm optimization

A variation of the standard particle swarm algorithm is the quantum particle swarm, proposed by Sun et al. (2004) it replaces the classical physics movement equations to a quantum behavior of the particles, where a wave function represents the particle’s position variation. The main purpose is the change in behavior of the particles resulting in an uncertain or probability search algorithm, which has a higher chance of finding solutions that are far away from the local search that the standard particle swarm usually stays. As Grotti et al. (2020) mentions, the equations Equation (7) and Equation (8) determine the state of the particles based on the Monte Carlo method, since in order to obtain the position of the particle, the state of the particle needs to be

Figure 11 – Some topology possibilities




Source – Lynn et al. (2018)

collapsed from the quantum state (which gives a probability of position) to the classical state (that gives a deterministic location).

$$\begin{aligned} X_{i+1} &= P_i - \beta(m_{best} - X_i)ln(\frac{1}{u_i}) \; for \rho >= 0.5; \\ X_{i+1} &= P_i + \beta(m_{best} - X_i)ln(\frac{1}{u_i}) \; for \rho < 0.5 \end{aligned} \tag{7}$$

$$\beta(t) = [(\beta_1 - \beta_0)(T - t)/T] + \beta_0 \tag{8}$$

Where $\beta_0$and $\beta_1$ are the shape parameters of the wave function, and $\beta(t)$ dependent on the current iteration $t$ and the max number of iterations $T$.

### 2.4.3 Related work

Alagappan et al. (2015) compares multiple algorithms for the fitting of a MF 6.0 tire model, including a particle swarm algorithm and a variation of the genetic algorithm. The research resulted in relatively close error values for the pure lateral force optimization, Figure 12 presents the pure lateral coefficients results. However all of the used algorithms are already established optimization libraries for Matlab, and no direct implementation to the problem were done by the authors. Zhuo et al. (2015) uses different implementations of the particle swarm algorithm, such as the adaptative control weight, the dynamically changing weight and the KPSO. Good fit was achieved by the author, however, the tire model used in the research was a simplified magic formula model with much less coefficients.

Figure 12 – Comparison for pure lateral force for the researched algorithms.

| Coefficients | $p_{Cy1}$ | $p_{Dy1}$ | $p_{Dy2}$ | $p_{Ey1}$ | $p_{Ey2}$ | $p_{Ey3}$ | $p_{Ky1}$ | $p_{Ky2}$ | $p_{Ky4}$ | $p_{Hy1}$ | $p_{Hy2}$ | $p_{Vy1}$ | $p_{Vy2}$ | SSE | MSE | RSQ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TRR(UB) | 1.535 | − 1.093 | 0.395 | 0.172 | − 5.185 | 0.045 | − 11.985 | 259.367 | 370.925 | − 0.002 | 0.001 | − 0.001 | − 0.002 | 497,090 | 0.022 | 1.000 |
| TRR(B) | 1.535 | − 1.093 | 0.396 | 0.171 | − 5.191 | 0.045 | − 11.984 | 5.000 | 7.230 | − 0.002 | 0.001 | − 0.001 | − 0.002 | 498,877 | 0.022 | 1.000 |
| NMS(UB) | 1.707 | − 1.085 | 0.240 | 0.455 | − 0.046 | − 0.068 | − 15.832 | 3.060 | 2.668 | − 0.003 | 0.001 | − 0.004 | 0.000 | 714,281 | 0.026 | 0.999 |
| PTS(UB) | 1.568 | − 1.096 | 0.418 | 0.272 | − 4.847 | 0.191 | − 12.089 | 13.367 | 18.587 | − 0.001 | − 0.006 | 0.010 | − 0.080 | 544,573 | 0.023 | 0.999 |
| PTS(B) | 1.535 | − 1.093 | 0.396 | 0.171 | − 5.190 | 0.045 | − 11.984 | 5.000 | 7.230 | − 0.002 | 0.001 | − 0.001 | − 0.002 | 498,877 | 0.022 | 1.000 |
| DE(UB) | − 1.535 | − 1.093 | 0.395 | 0.172 | − 5.186 | 0.045 | − 11.984 | − 11,099,787 | − 15,874,868 | − 0.002 | 0.001 | − 0.001 | − 0.002 | 497,089 | 0.022 | 1.000 |
| DE(B) | 1.535 | − 1.093 | 0.396 | 0.171 | − 5.190 | 0.045 | − 11.984 | 5.000 | 7.231 | − 0.002 | 0.001 | − 0.001 | − 0.002 | 498,877 | 0.022 | 1.000 |
| PSO(B) | 1.535 | − 1.093 | 0.396 | 0.171 | − 5.190 | 0.045 | − 11.984 | 5.000 | 7.231 | − 0.002 | 0.001 | − 0.001 | − 0.002 | 498,877 | 0.022 | 1.000 |
| CS(UB) | 1.598 | 1.091 | − 0.377 | 0.279 | − 4.256 | 0.025 | − 12.189 | − 0.779 | 5.046 | − 0.002 | 0.002 | − 0.002 | 0.010 | 431,402 | 0.020 | 1.000 |
| CS(B) | 1.535 | − 1.093 | 0.396 | 0.171 | − 5.190 | 0.045 | − 11.984 | 5.000 | 7.231 | − 0.002 | 0.001 | − 0.001 | − 0.002 | 498,877 | 0.022 | 1.000 |

Source – Alagappan et al. (2015)

# 3 METHODOLOGY

This chapter is intended to explain the methods used to obtain the results presented in this thesis. To this work, a fully working software was written, that includes all the necessary steps to fit a MF tire model. However, to keep in focus with the thesis objectives, only the relevant software implementations are explained.

## 3.1 THE PROCESS OF MODELING A TIRE

A set of steps is required in order to model a tire using the Pacejka MF model. First, tire test data is required. This data can be obtained by two main ways: Field testing using a sensory equipped test car or a laboratory test with tire testing machines. These two types of obtaining tire data are explained in the following sections

### 3.1.1 Outdoor testing

Field or outdoor testing, is referred to as a test that obtains the tire data from a instrumented tire/ vehicle on a test performed typically at proving grounds. There are two possibilities for a field test: movable testing rig or vehicle testing.

#### 3.1.1.1 Movable testing rig

A vehicle is used to tow an instrumented trailer that in its turn, has the tire being tested in contact with the road mounted in a testing rig. This trailer is used to simulate the tire testing machines of lab use. The main difference is that the surface of contact is more representative than of the material used in lab testing. Some disadvantages include limitation to lower test speeds, the uncontrolled environment and the low degree of repeat-ability. Figure 13 shows an example of such testing machinery.

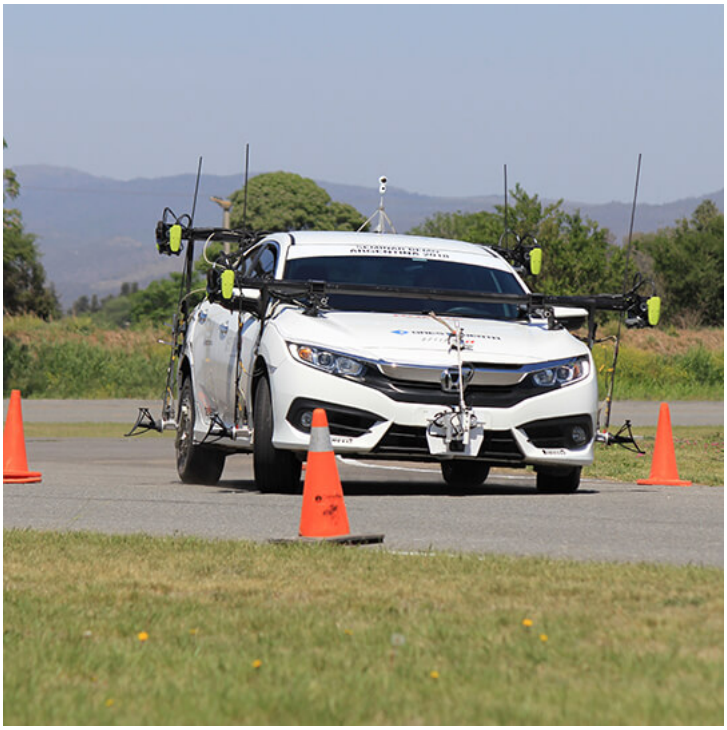
#### 3.1.1.2 Vehicle testing

The other way of outdoor testing is by using an instrumented car and testing the tires as they are mounted in this vehicle. This kind of test produces accurate measurements for the designed test range, due to the tire being in the environment that its meant to be used and mounted to a real vehicle. However, it is limited to what the driver can achieve, since the maneuvers required to test all the tire capability are difficult to perform. The forces in the tire are typically measured with wheel transducers, which is an expensive sensor adding to the monetary cost of the test, one of the major disadvantages of this type of test. Figure 14 shows an example of a test car using wheel force transducers.

Figure 13 – Tire test trailer example.


Source – Tass International (2021).

Figure 14 – Optimum G test car.



Source – Optimum G (2021)

### 3.1.2 Indoor testing

The other option to obtain tire test data is with indoor testing, where laboratory test facilities are used to generate more repeatable data by performing the test in environmentally controlled conditions. The machine used to obtain the data typically consists of one or more rotating steel drums, covered by a steel belt with an abrasive surface to simulate road surface. The tire is mounted in an articulated head which regulates the load, slip angle, speed, pressure and other conditions while measuring the forces and moments generated. Figure 15 illustrates a Calspan testing machine.

Figure 15 – Calspan tire testing machine.

Source – Calspan (2021)

The advantage of this setup is that the tyre can be tested across a range of load cases while under very consistent test conditions. This is due the rigs being able to accurately apply to requested load case, while the indoor nature of the testing means that the local temperature and moisture can be controlled (SMITH; BLUNDELL, 2017). According to Megaride (2021) indoor tire testing is usually less expensive than outdoor vehicle testing, since there are companies specialized in this kind of testing that offers the testing service such as Calspan. Also, there is much more preparation and time consuming tasks in outdoor vehicle testing that add to the overall cost. The difference in the surface used in the steel belt and the asphalt in a road, can be corrected in the MF model using its scaling factors. For this reason indoor testing is suitable to obtain tire test data to be used in the MF model.

### 3.1.3 FSAE Tire test consortium

The data to test the application, and validate the tire models implementation of this work were obtained via the FSAE tire test consortium.

The FSAE Tire Test Consortium (FSAE TTC) is a volunteer-managed organization of member schools who pool their financial resources to obtain high quality tire force and moment data targeted for Formula SAE and Formula Student competitions. The FSAE TTC's role is to gather funds from registering member schools, organize and conduct tire force and moment tests and make the data available to individuals at member schools (FSAE TTC, 2005).

### 3.1.4 Data treatment

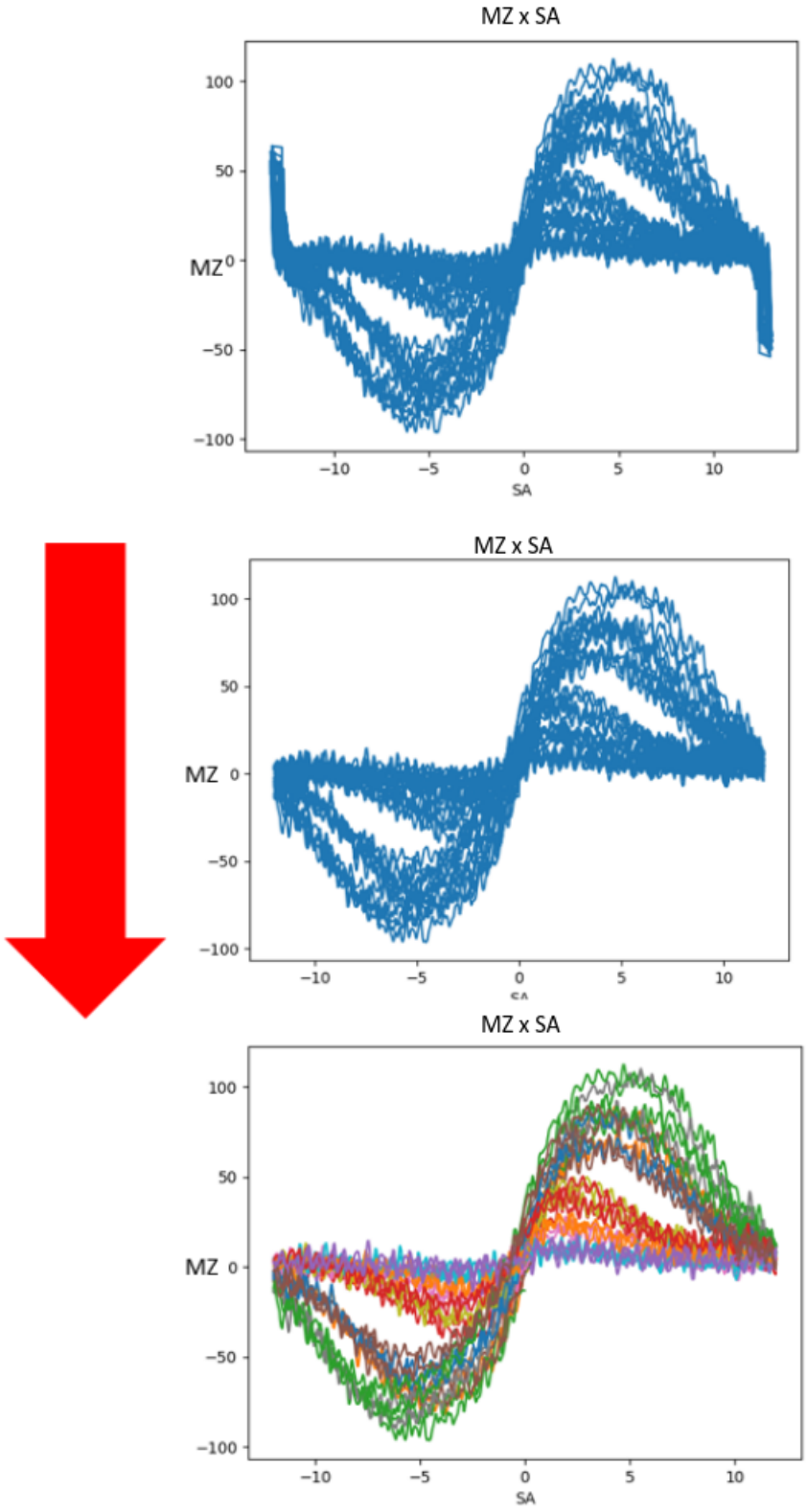
After obtaining the test data, it is required to treat it, to filter any bad data and to separate the tests cases. Tire data from indoor testing machines are usually separated in a free-rolling test and a combined test. Free-rolling tests means that there is no acceleration applied to the tire as it is free-rolling in the test surface so that the SR is equal to 0. For the combined test different fixed values of slip angles are set and the machine varies the SR. Figure 16 presents the treatment done to a free-rolling test, the different colors imply the different test cases. The graphs are plotting the aligning torque by slip angle measured by the machine. The first graph on the figure shows the raw data from the machine, for the second, the slip angle range was filtered to capture only the tire working range. Further treatment may be applied to help the fitting process, such as down-sampling and collapsing.

#### 3.1.4.1 Binning

The process of separating the tire data into test cases is known in more general terms as data binning. It consists in the process of separating the data in different bins (test cases) where all the collected data which fall into a specific small interval is labeled or some times replaced by a value representative of that interval, being a way of quantization. This process is specially important for on-track tests that are much less controlled than indoor tests, being more susceptible to effects of minor observation errors.

For this work, a very simple method of binning was used due to the in-door nature of the data. Given a sensibility value to the tests input variables, (normal force, inclination and slip angle, etc.) the algorithm detects variations in those data channels, separating the original data set into different smaller versions containing all the data in the range where the variation is lower than the sensibility.

Figure 16 – Example of tire data treatment.



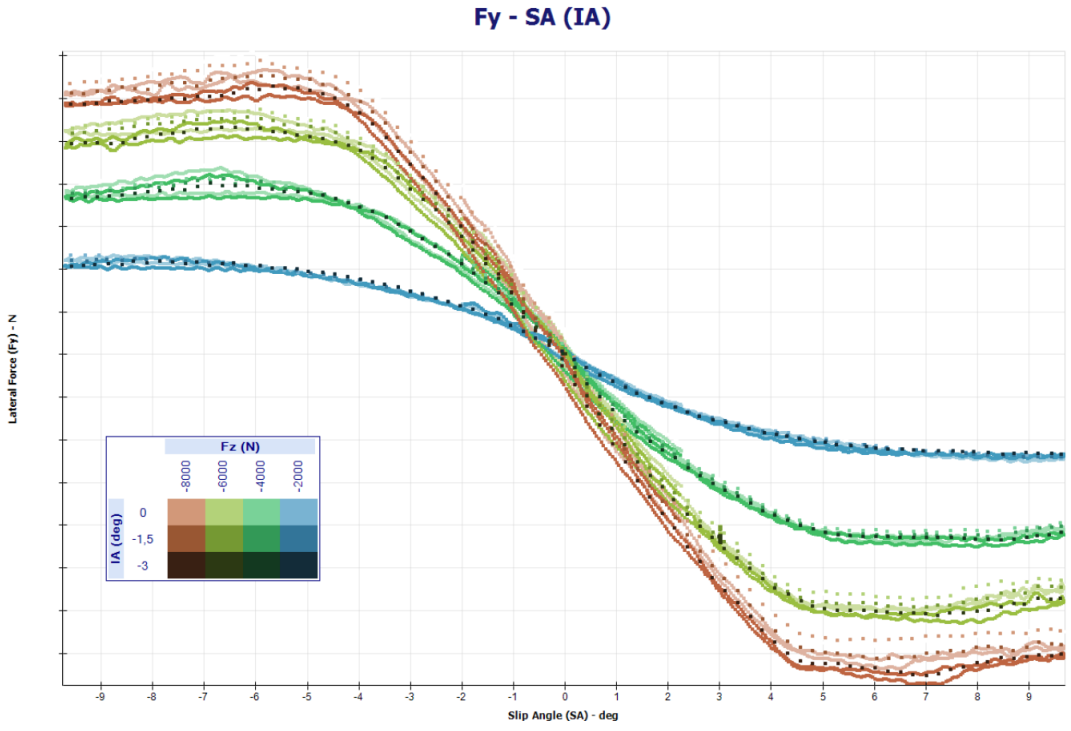

Source – Author (2022).

#### 3.1.4.2 Collapsing

Collapsing is the process of using one or several grouping variables to average data points in the data set, reducing the overall sample number. This is done to help the optimization algorithm by reducing the number points it needs to compare in each iteration. Also, in the tire fitting case, it also helps by reducing and averaging the tire hysteresis effect which MF models does not usually capture.

There is also multiple ways to collapse a data set, most known methods include K-means clustering and it's variations, however a different and simpler method was used in this work, since it can better filter the hysteresis effects granting better error results for tire fitting. It consists in averaging the data channels according to the normal load average for each sample number. Figure 17 shows the difference in the original and collapsed data.

Figure 17 – Difference of raw (lines) and collapsed (points) test data. (values hidden due confidentiality). IA = inclination angle




Source – OptimumTire.

## 3.2 SOFTWARE DEVELOPMENT

In recent years, software engineering has become an effective engineering discipline. Due to the constantly increasing complexity of its tasks and the diversity of

its application domains, a portfolio of software engineering techniques has been constructed, offering a customized range of suitable methods and concepts for the application domain, criticality, and complexity of each system to be developed. Techniques for management of projects, configuration, variant and quality, as well as software product lines, development processes, specification techniques, analysis and design patterns, and best practices for specific tasks are only some elements of this portfolio (RUMPE, 2016).

### 3.2.1 Code structure

The object-oriented programming (OOP) is a technique or a paradigm created to approach the programmer problem in a more effective and communicative way than the traditional procedural approach. In OOP, computer programs are designed by making them out of objects that interact with one another (KINDLER; KRIVÝ, 2005).

For this work OOP was used to structure the proposed application, since it allows a more divided approach that suited for the genetic and PSO algorithms. The OOP also allows the use of software architecture tool such as the Unified Modeling Language (UML) diagrams. according to Rumpe (2016) the UML is a communication and mapping tool based on visualization that allows the software development team share the same concepts and transfer knowledge between themselves more efficiently.

UML class diagrams describes the structure or rather the architecture of a system and are thus the basis for almost all other description techniques. The class concept is used universally in modeling and programming; it therefore offers a backbone that enables us to trace requirements and errors through the different activities of a project (RUMPE, 2016).

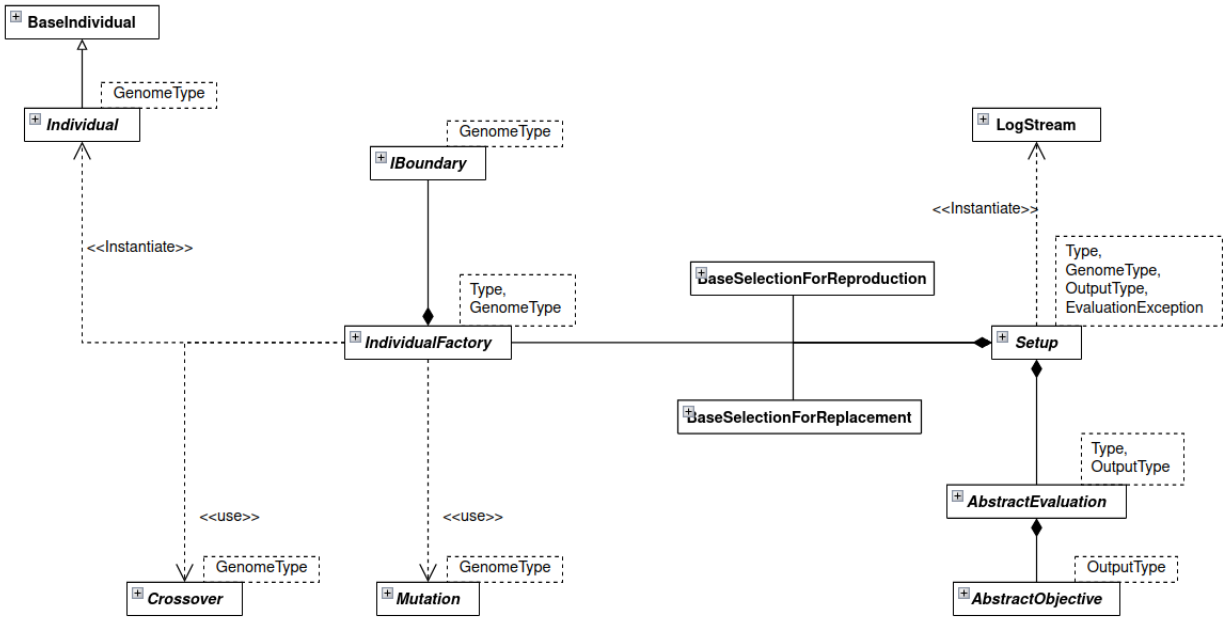
The entire framework for this work was coded in C++ by the Author in collaboration with OptimumG, including the GA and PSO algorithms, the data management and tire modeling classes. The following sections are intended to explain the software implementation of the algorithms used.

### 3.2.2 Genetic algorithm implementation

The genetic algorithm development was a collaboration between the author and Avi (2021), since the GA capability needed to be extended to any optimization problem. For this reason the OptimumGenetics library was created. It consists in a template library coded in the C++ language that can be applied with little to no modification to any optimization problem applicable to a genetic algorithm. In the Figure 18 we can visualize the UML class diagram of the library. Each template class requires a defined type to be operational according to the problem being optimized.

The genome type refers to the variable that affects the fitting of the individual, for this application each MF coefficient is a gene (which is the division of a genome) that

Figure 18 – OptimumGenetics library class diagram.




Source – Author (2022)

has the type of a double numeric variable referring to its value. This value is contained in a standard library vector that is then modified in the evolution process of the genetic algorithm to generate new model configurations.

The individual for the optimization is the tire model. This way each individual has it's set of genes (MF parameters) that are modified in the evolution process. The output and data type is an standard library vector of doubles which refers to the forces arrays. Each force and moment is divided and has it's own array object for a given input of normal force, slip angle, slip ratio inclination angle and pressure, depending on the force and model implementation.

The exception type is required for the algorithm to handle the possible exceptions that occur in the evolution process. The defined "NaNFitnessException" was implemented to capture possible miss evaluations obtained from the forces error calculation. For the application proposed in this work the template type parameters for the genetic algorithm are defined according with Table 1.

The main class of the library is the Setup, this is where the user sets the algorithm behavior. It uses the evaluation class and the selection operators to evaluate and select the individuals from the population that is generated with the IndividualFactory class, which in it's turn, uses the variation operators (crossover and mutation) classes to generate new tire models. The Logstream class is used to store and display statistical data from the optimization.

Table 1 – Template parameters for the kinematics optimization.

| **Template name** | **Assigned class** |
|---|---|
| GenomeType | std::vector<double> |
| Type | TireModel |
| OutputType | std::vector<double> |
| EvaluationException | NaNFitnessException |

Source – Author (2021)

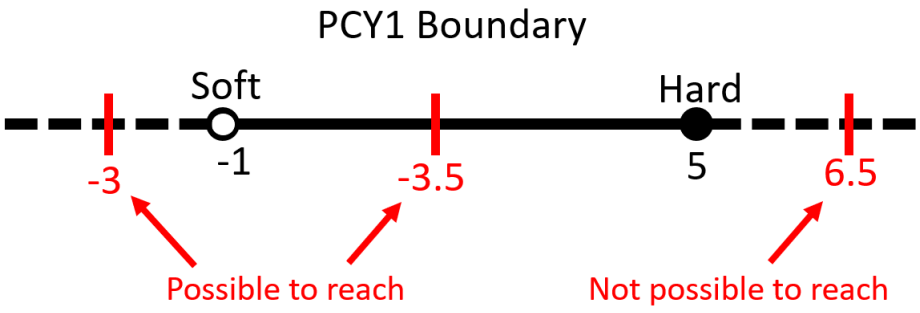
The template nature of the library allows it to be adaptive to any problem type, but it also requires some classes to be re-implemented specifically for each problem. Those are explained in the following sections.

#### 3.2.2.1 Boundaries class

The boundaries are specific to each problem so a class is needed to be implemented for the fitting process. A linear boundary class was created, this class is intended to limit the MF coefficient to a user defined linear boundary. This boundary can be defined to be hard or soft in each lower or upper side, this allow a better control of the search space and allows the algorithm to search for solutions that can help the population to get out of a local minima.

Figure 19 illustrates the behavior of a linear boundary like the one defined for the current work. It is important to remember that each MF coefficient has its own defined boundary.

Figure 19 – Linear boundary illustration.




Source – author (2021)

#### 3.2.2.2 Operators

The cross over and mutation classes are defined according to the genome type of the algorithm. For this reason each method of cross over or mutation need to be implemented in it's own class. The library allows multiple implementations to be used, however, for simplicity sake, only the best performing implementation will be explained.

For this work the Blx-$\alpha$ was used to allow a better coverage of the design space allowing a search for out of boundary values according to a defined user parameter of $\alpha$. This grants the search of MF parameters combinations that usually would be discarded.

The mutation implementation follows a Gaussian principle that generates mutations according to a Gaussian distribution, the standard variation $\sigma$ allows the strength of this mutation to be user defined.

The selection for reproduction used is known as the truncation selection. It selects the n best individuals in the population, being n a parameter chosen by the user.It has a fast conversion rate but it can also trap the population in a local minima.

The selection for replacement used is similar to the truncation selection, but for replacement. The steady state selection for replacement selects the n worst individuals to be replaced, with n usually being a small number.This strategy is useful when the representation of the solution is distributed on several individuals, that is, when the design variables are really disperse through the population, but the population has a low overall fitness standard deviation.(AVI, 2021)

#### 3.2.2.3 Evaluation class

The evaluation class also depends on the problem being optimized, since it is responsible to evaluate the individuals with the fitness function. For this reason the evaluation class was implemented taking into account the error analysis between the tire test data and tire model data as a fitness function. Multiple error calculations were tested to analyze the best convergence of the GA and the one that is used for this work is defined in the Equation (9).

$$Error = \frac{\sqrt{\sum (Model - Data)^2}}{\sum |Data|} \tag{9}$$

The error is calculated by taking a force or moment output vector from the model using the test data input parameters (Normal force, inclination angle, slip angle, pressure, etc.) and comparing it to the correspondent force or moment vector from the test data. Weight functions can be used to give more value to specific inputs values but for simplicity's sake were not used in this work.

Each evaluation contain a one or more objectives, so it's possible to fit multiple forces and moments at the same time, however, due to the MF coefficients nature of being dependent on one another, it is best to fit one or two forces at a time.

#### 3.2.2.4 Extinction mechanism

A mechanism to help the algorithm to get unstuck from local minima was created by the author and Avi (2021). Based on nature's great extinctions in history, where during this events, most species were removed from earth's environment leaving only a few very fit individuals, which mutated and reproduced in the following years, granting the possibility for new species to flourish over time. The concept is the same for the Author's genetic algorithm. When a elongated stagnation is reached and the best fitness value is repeated during multiple generations (possibly a local minima), an extinction event is triggered and all but the best individual is removed. Then the population is refilled with mutated individuals generating a whole new set of solutions that can lead to an ever better error number.

### 3.2.3 Particle swarm implementation

The particle swarm development was also a collaboration between the author and OptimumG and it's also able to be extended to any optimization problem. A simpler approach was taken than the GA in the sense that fewer template classes were created, and the configuration for each particle is set in a defined type. This requires a serialization of the problem being optimized in the form of a vector of doubles. For the tire fitting purpose, this step was easy to follow since it is the same approach used in the GA, being the vector of doubles, the vector of MF coefficients. The algorithm's simpler nature also allowed an easy overall implementation with much less classes and functions to be re-implemented for each specific problem. Figure 20 shows the UML class diagram of the library. For the application proposed in this work the template type parameters are defined according with the Table 2.

Table 2 – Template parameters for the kinematics optimization.

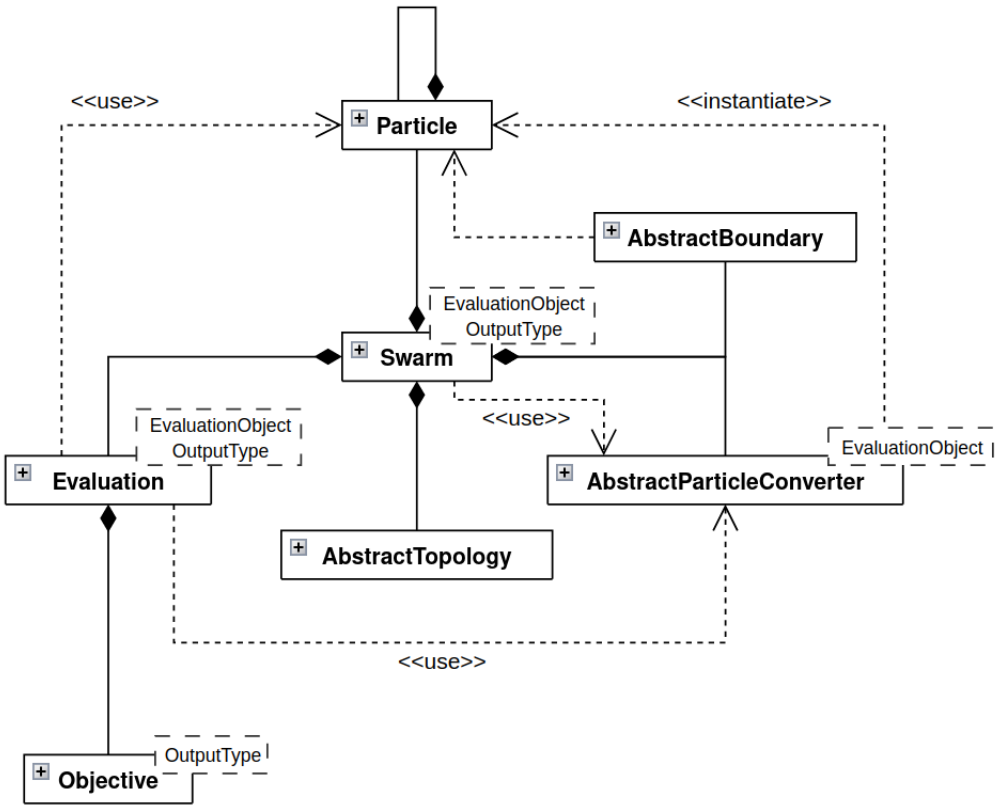
| **Template name** | **Assigned class** |
|---|---|
| EvaluationObject | TireModel |
| OutputType | std::vector<double> |

Source – Author (2022)

The evaluation and objective class have the exact same functionality as the classes with same name in the GA, only with some differences in method signatures to accommodate the rest of the library. The AbstractParticleConverter class has a similar functionality of the IndividualFactory class from the GA, where it's responsible of converting particle to evaluation object (In this case tire models) and vice versa.

The swarm is the main class from this library, where the user control's the optimization process and it's possible configurations. The swarm is responsible of initializing the particles at the initial iteration, and updating it's position and velocity in the subse-

Figure 20 – OptimumSwarm library class diagram.




Source – Author (2022)

quent ones. It's also responsible to call the evaluation class to evaluate each particle. During the initialization process it's also responsible to call the topology class to tell the particles which particle it's connected to.

The particle class is the building block of the swarm, it is responsible of searching the best solution, storing the tire model configuration in a vector of doubles. The movement behavior are described by the equations mentioned in chapter 2 and can be easily changed into the standard or quantum behavior. The boundaries are used when initializing a particle to grant that it is initialized in bounds, although they do not necessarily stay on set boundaries during the optimization process.

The classes that need re-implementation for a specific problem are the particle converter, boundary, evaluation and objective. Since they are dependent on the configuration of the evaluation object that change for each problem.

#### 3.2.3.1 Evaluation and objective classes

As stated before the evaluation and objective classes work the same way it does for the GA. The error is also calculated the same way (Equation (9)), for comparison

reasons.

#### 3.2.3.2 Boundaries

The boundaries were also implemented in the same way, the main difference is that it is only used in the initialization of the particles, and differently than the GA, that initialize individuals each generation, the PSO only initializes the particles on the first iteration and the population is kept the same throughout the process only changing the configuration. A different approach can be implemented where, when searching for a new solution, if the particle set up a coefficient out of bounds, it's velocity is changed to make the particle "bounce" off back to the bounds. However for this work some times searching outside the bounds is desirable.

#### 3.2.3.3 Particle converter

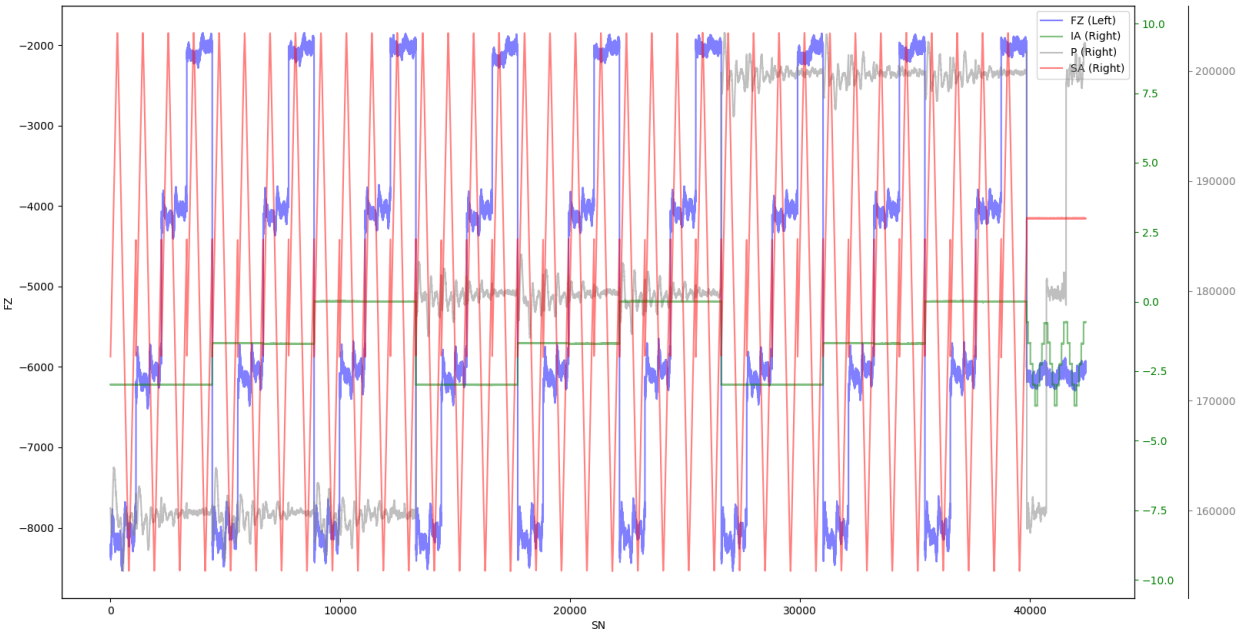
The particle converter class works similarly to the individual factory class from the GA. It's responsible for converting the particle to the evaluation object (tire model) to be evaluated in each iteration. Converting the evaluation object to a particle is also possible but not as much used. For the particle swarm implementation, the particle configuration that stores the information to generate a new evaluation object, was chosen as a fixed type, instead a variable like the GA's genome type. This allows a simpler implementation for different problems and overall algorithm implementation. However it also limits the algorithm reach to only problems that can be serialized into a vector.

### 3.2.4 Data used for the comparison

The data used in the algorithm comparison is real race car tire data provided by OptimumG, for this reason, all the values and tire specifics characteristics are modified and do not represent the real values since it is classified. However, are still useful to present the methodologies and results obtained. It also "stress" tests the software to a real life use case, validating it's usage for a commercial software.

The data was taken as it's raw state obtained from the test facility. It was taken from a in-door test similar to the one explained previously in this chapter. Due to the data raw state, a treatment was also needed to allow a fitting process to take place. The following subsection will explain the treatment applied. The same final data was used for both algorithms to a fair comparison. Figure 21 shows an overall representation of the data used, where FZ is normal force in Newtons, SN is sample number, IA and SA are inclination angle and slip angle respectively in degrees, and P is pressure in Pascal.

Figure 21 – Free rolling data (input channels by sample number) used for comparison.




Source – Author (2022)

#### 3.2.4.1 Data treatment applied

The data comes organized in a tabular text file, the free rolling test file is separated from the combined one, so, in this section the explanation follows the treatment applied to the free rolling test, the same treatment can be applied to any other tests with little modifications.

Figure 21 shows the input channels that the machine uses in this specific test to acquire the desired force and moments channels. It's possible to see the slip angle sweeps in red, the inclination angle steps in green, normal force in blue and pressure in gray. The test behavior changes abruptly at the end, indicating a different purpose for that specific setup. Most of the time, temperature controlling sweeps occur before a step of pressure is changed, these are only for tire temperature control and should not be included in the fitting process, since it can skew the data, being not representative of a tire behavior that MF can capture.

After cropping the unwanted or problematic data, the binning process is performed. In this work the data binning resulted in 39 bins being 36 with approximately 1110 samples each and 3 bins with 900 samples each. After the binning is done, the collapsing process can begin. Multiples values were tested for the collapsing method implemented, the value of 80 samples per bin resulted in a good compromise between performance and preserving data. The final treatment numbers can be found in Table 3.

Table 3 – Data treatment numbers.

| Parameter | Value |
|---|---|
| Total Number of Bins | 39 |
| Initial Number of Samples | 42438 |
| Final Number of Samples | 3120 |
| Percent Decrease | 92.6 |

Source – Author (2021)

### 3.2.5 Optimization test functions

Test functions, also known as artificial landscapes by Bäck et al. (1996), are useful to evaluate characteristics of optimization algorithms, such as:

- Convergence rate
- Precision
- Robustness
- General performance

These test functions are aimed for giving an idea about the different situations that optimization algorithms have to face when encountering these kinds of problems. For the purpose of this work, two types of single-objective landscapes were used.

#### 3.2.5.1 Rastrigin function

The Rastrigin function is a non-convex function used as a performance test problem for optimization algorithms. It is a typical example of non-linear multi-modal function, first proposed in 1974 by Rastrigin as a 2-dimensional function. The generalized version was popularized by Bäck et al. (1996). Finding the minimum of this function is a fairly difficult problem due to its large search space and its large number of local minima.

On an n-dimensional domain it is defined by:

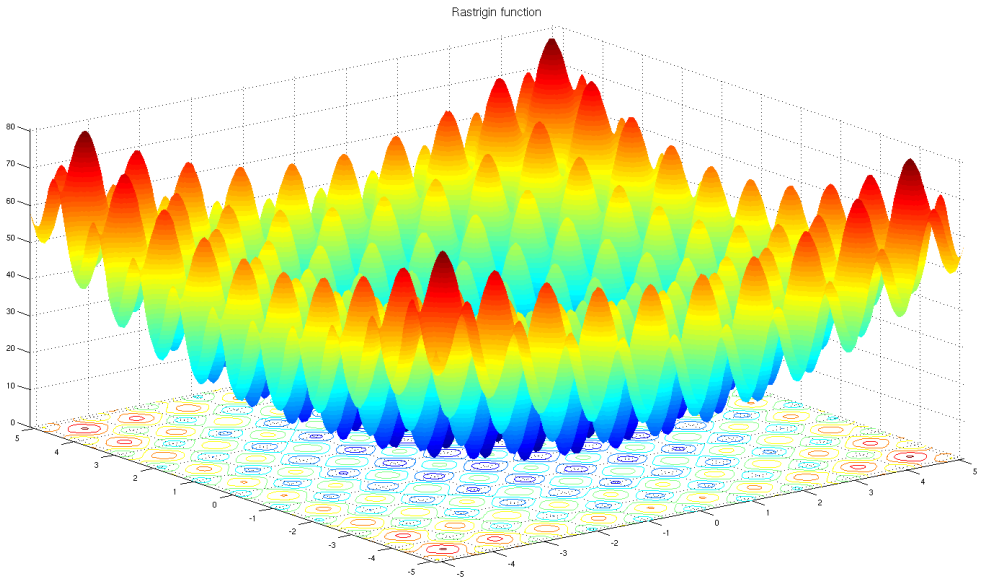
$$f(x) = A_n + \sum_{i=1}^{n} [x_i - A\cos(2\pi x_i)] \tag{10}$$

A 3d view of the function can be seen in Figure 22.

#### 3.2.5.2 Rosenbrock function

The Rosenbrock function also known as Rosenbrock's valley or Rosenbrock's banana function, is a non-convex function that was introduced by Howard H. Rosenbrock in 1960, and is used as a performance test problem for optimization algorithms. The global minimum is inside a long, narrow, parabolic shaped flat valley. Finding the

Figure 22 – 3D view of the Rastrigin function.



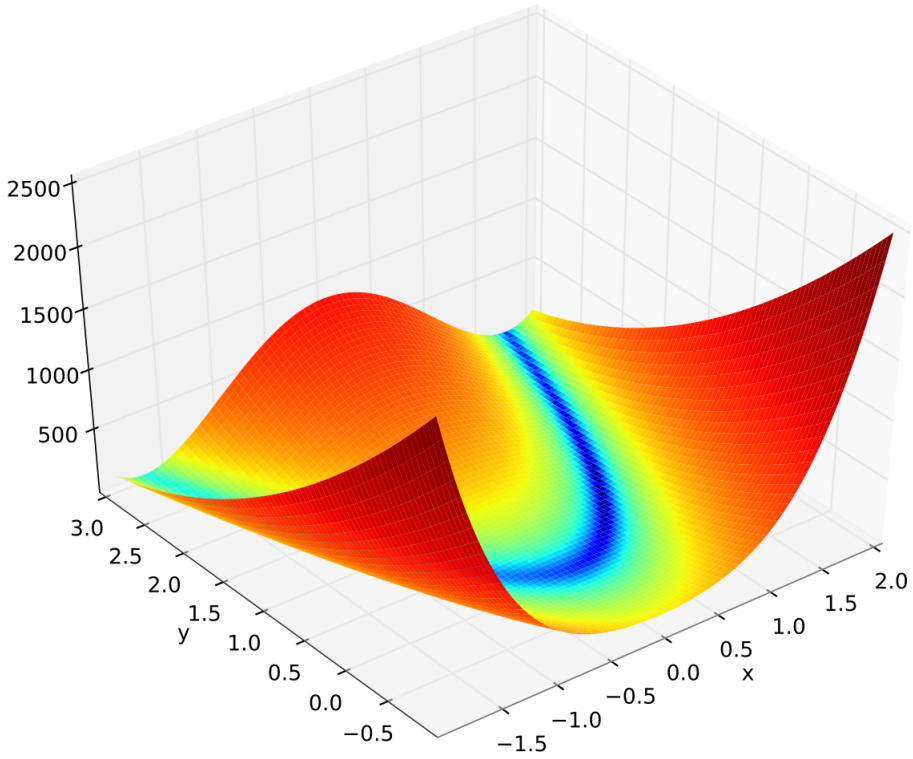

Source – (WIKIPEDIA, 2022)

valley is an easy task, however, the convergence to the global minimum a very difficult problem due to extreme slow gradient leading to the minimum. It is a contrast compared to the Rastrigin function since there is only one valley and it's a very different optimization problem. The Rosenbrock function is defined by:

$$f(x, y) = (a - x) + b(y - x) \tag{11}$$

Usually $a = 1$ and $b = 100$ and the local minima is $(1, 1)$. The 3d view of the function can be seen in Figure 23

Figure 23 – 3D view of the Rosenbrock function.




Source – (WIKIPEDIA, 2022)

# 4 RESULTS AND DISCUSSION

Both algorithms were implemented and applied to tire model fitting. First, tests were made with the mentioned test function, to achieve the best configuration. Then the best results for these tests were compared for both algorithms. Later, an application on tire fitting was performed, and a new comparison was made.

### 4.0.1 Genetic algorithm results

The genetic algorithm configuration used for the test functions can be seen in Table 4.

Table 4 – GA Test Setup.

| **Parameter** | **Value** |
|---|---|
| Population size | 100 |
| Maximum number of generations | 200 |
| Selection for reproduction type | Tournament (selection: 20, size: 5) |
| Selection for replacement type | Truncation (selection: 50) |
| Alpha value (crossover) | 1.7 |
| Standard deviation value (mutation) | 1.0 |
| Mutation rate | 10% |

Source – Author (2021)

#### 4.0.1.1 Test function results

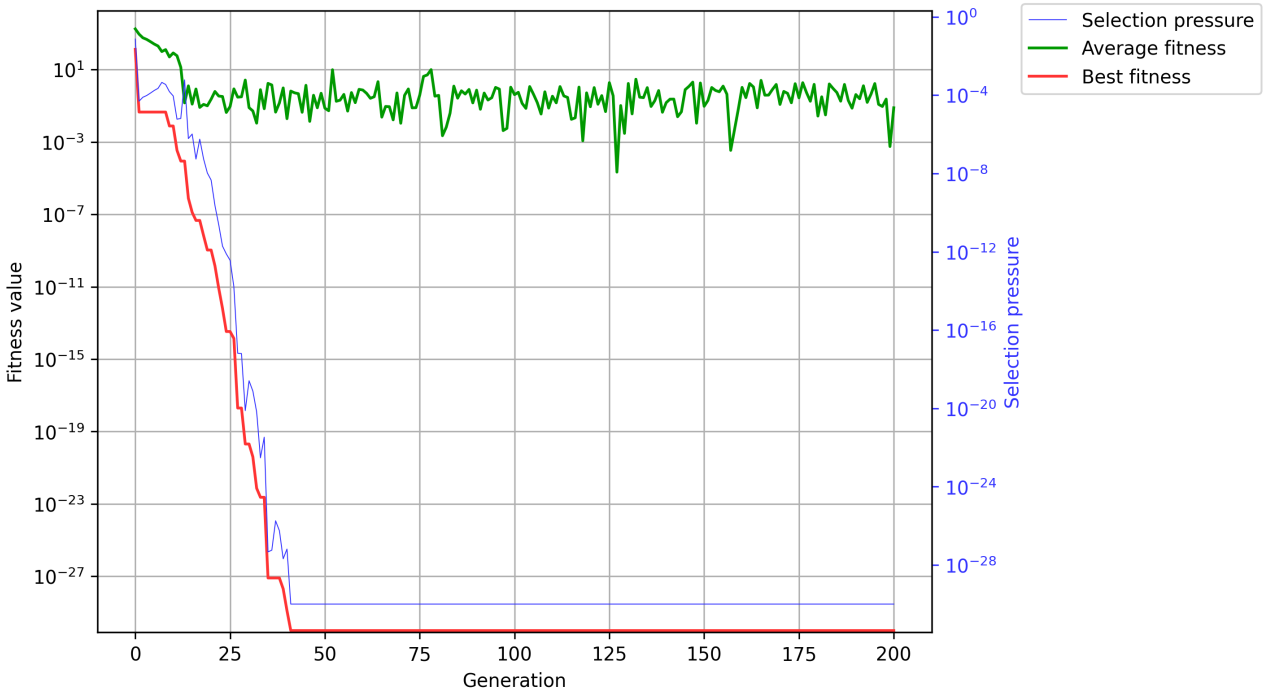
Due to the random nature of the algorithms, after the best configuration was achieved a series of 3 optimization for each test function were run to gather an average result. The convergence graph seen in Figure 24 and Figure 25 is a random result picked from the tests runs. The average fitness is the population average fitness on that specific generation, the best fitness, is the fitness value of the best individual on that generation and the selection pressure is Table 5 and Table 6 presents the average results for the Rastringin and Rosenbrock functions respectively.

Table 5 – GA Rastrigin test average results.

| **Parameter** | **Value [unit]** |
|---|---|
| Time Elapsed | 66 [ms] |
| Total Evaluation Count | 10150 |
| Time per Evaluation | 6.5 [us] |
| Final Fitness Value | 0 |
| Iteration Where Minima was Found | 64 |
| Final Average Population Fitness | 0.81 |

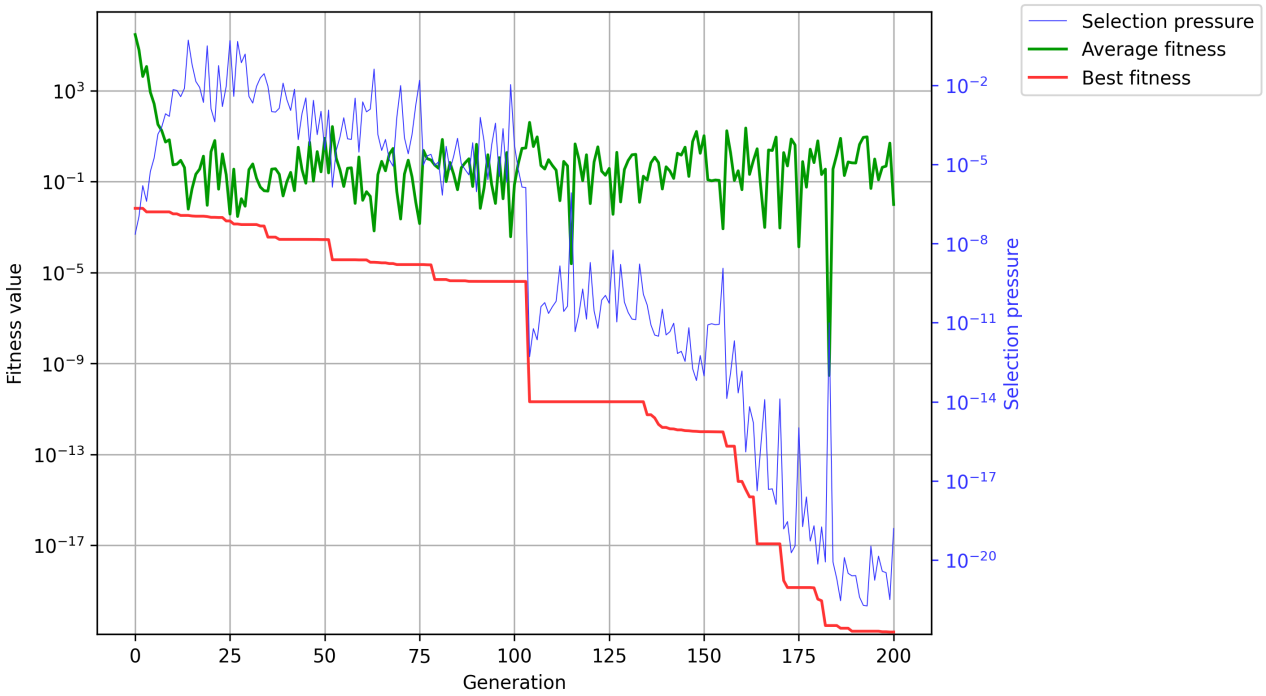
Source – Author (2022)

Figure 24 – Convergence plot of GA's Rastrigin test function.




Source – Author(2022)

Figure 25 – Convergence plot of GA's Rosenbrock test function.




Source – Author(2022)

Table 6 – GA Rosenbrock test average results.

| **Parameter** | **Value [unit]** |
|---|---|
| Time Elapsed | 71.6 [ms] |
| Total Evaluation Count | 10150 |
| Time per Evaluation | 6.96 [us] |
| Final Fitness Value | 3.35e-10 |
| Iteration Where Minima was Found | 194 |
| Final Average Population Fitness | 0.058 |

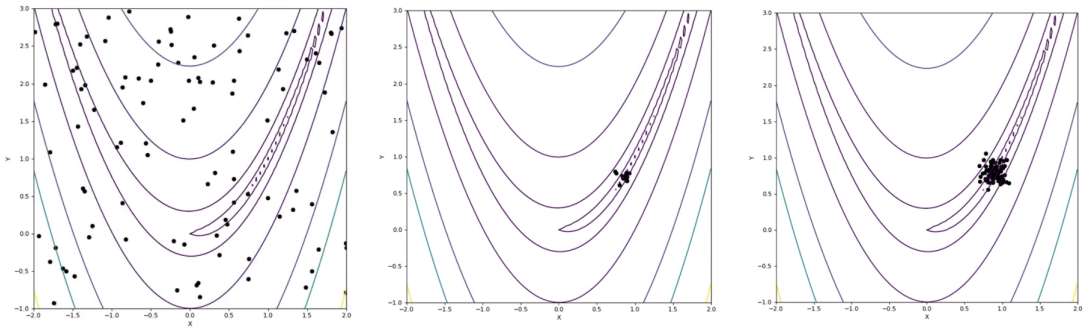
Source – Author (2022)

It's possible to note a huge difference in algorithm behavior in the two test functions. The Rastrigin function presents an early convergence rate and it finds the global minima pretty easily. The Rosenbrock function, however, never actually converges to a single value, and although it's numerically close to the global minima, the value found is farthest from zero than the result found in the Rosenbrock test.

It is possible to notice the effect of the extinction mechanism in the Rosenbrock test. In Figure 25 near generation 100 a big dip on the best fitness occurs after a long period of stagnation. This happens because an extinction is triggered and a best result is found in the mutated population. This, however, doesn't occurs in the Rastrigin test, the algorithm can find the minima in it's own without the aid from the mechanism.

Figure 26 shows a 2D projection of the Rosenbrock function and the individuals from a GA test optimization. It's possible to see that the algorithm rapidly converges near the global minima, however on iteration 51 we see a high fitness value, but after an extinction is triggered (seen by the suddenly appearance of a cloud of mutated individuals), the fitness value rapidly decreases.

Figure 26 – 2D projection of a Rosenbrock test optimization presenting the behavior of the extinction mechanism.



Best Fitness: 6e-3
Generation: 1

Best Fitness: 1e-4
Generation: 51

Best Fitness: 1e-13
Generation: 52

Source – Author(2022)

#### 4.0.1.2 Tire fitting results

The GA setup was a bit different for the tire fitting application. The best setup found is presented in Table 7. More individuals and population were needed to find a good fit value. The selection for reproduction was also different. The tournament selection present a more steady and consistent convergence rate, however the truncation selection presents better final values. the mutation and crossover values were also adjusted to better fit the other parameters. Another parameter not mentioned before is the thread number, for the test functions only one thread was used, since the process of queuing tasks were longer than evaluating the individual, however a sweet spot of 6 threads was found for the tire fitting application.

Table 7 – GA Tire Fitting Setup.

| **Parameter** | **Value** |
|---|---|
| Population size | 300 |
| Maximum number of generations | 500 |
| Selection for reproduction type | Truncation (selection: 75) |
| Selection for replacement type | Steady-state (selection: 75) |
| Alpha value (crossover) | 2.2 |
| Standard deviation value (mutation) | 1.2 |
| Mutation rate | 15% |

Source – Author (2022)

The force fit used for this work is the Fy pure, because it is the first step in fitting the MF tire model, leaving all the other forces and moments dependent on the Fy pure coefficients, thus a model with good Fy pure fitting has a greater possibility of resulting in a good overall tire model.

The genetic algorithm presented a great overall fitness. Table 8 shows an average results for 3 different fit procedures, Figure 27 presents the algorithm behavior in an example optimization and Figure 28 shows the force plot of the model on top of the collapsed data used for fitting.

The tire fitting plots shown here were done using the OptimumTire software, since it has a good and convenient plotting tool, however it's important to remember that OptimumTire was strictly used only to plot the results, all the calculations, fit and tire treatment process were done with the software discussed and implemented along this thesis.

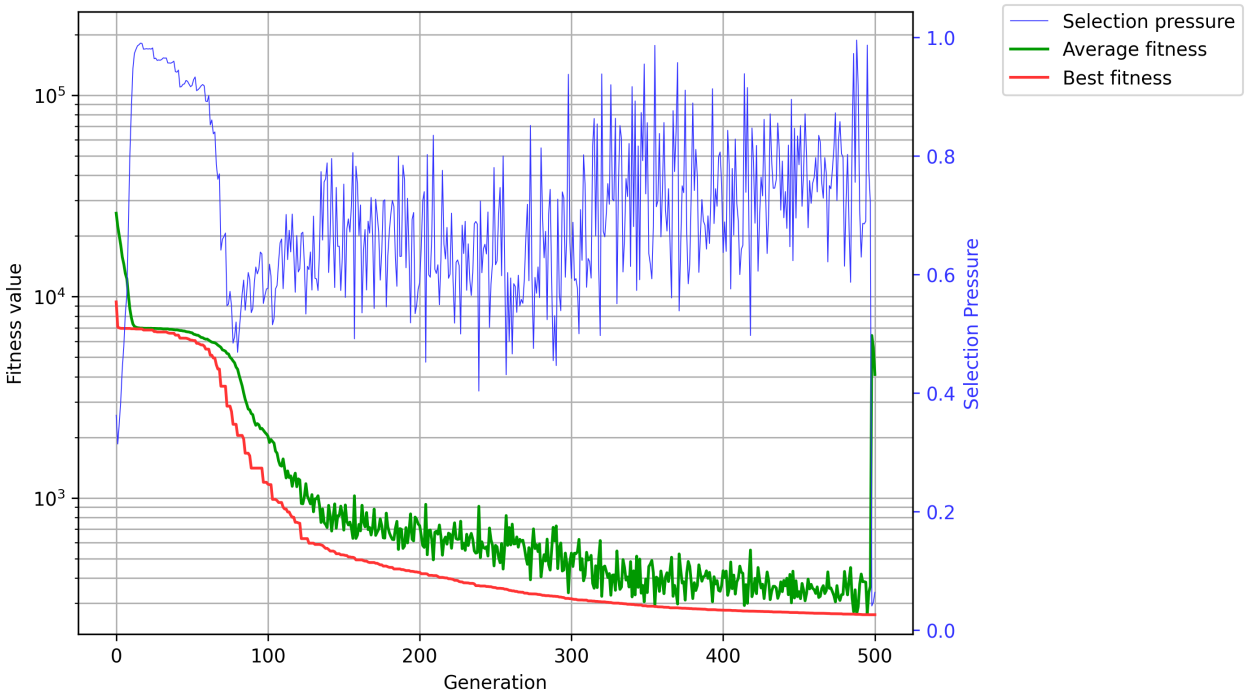
The final model obtained with the genetic algorithm resulted in a great fit considering the difficulty of data and model nature. In the convergence graph is possible to see that there was a rapid convergence in the first 100 generations and a much slow gradient in the other 400. It's also possible to see the adaptative mutation rate working on those generations, seen by the fluctuations in average fitness. This means that in the last 400 generations the algorithm is more dependent on the mutation than the

Table 8 – GA Tire Fitting average results.

| **Parameter** | **Value [unit]** |
|---|---|
| Time Elapsed | 24 [sec] |
| Total Evaluation Count | 38025 |
| Time per Evaluation | 649 [us] |
| Final Fitness Value | 234 |
| Iteration Where Minima was Found | - |
| Final Average Population Fitness | 4099 |

Source – Author (2022)

Figure 27 – GA convergence plot for tire fitting.



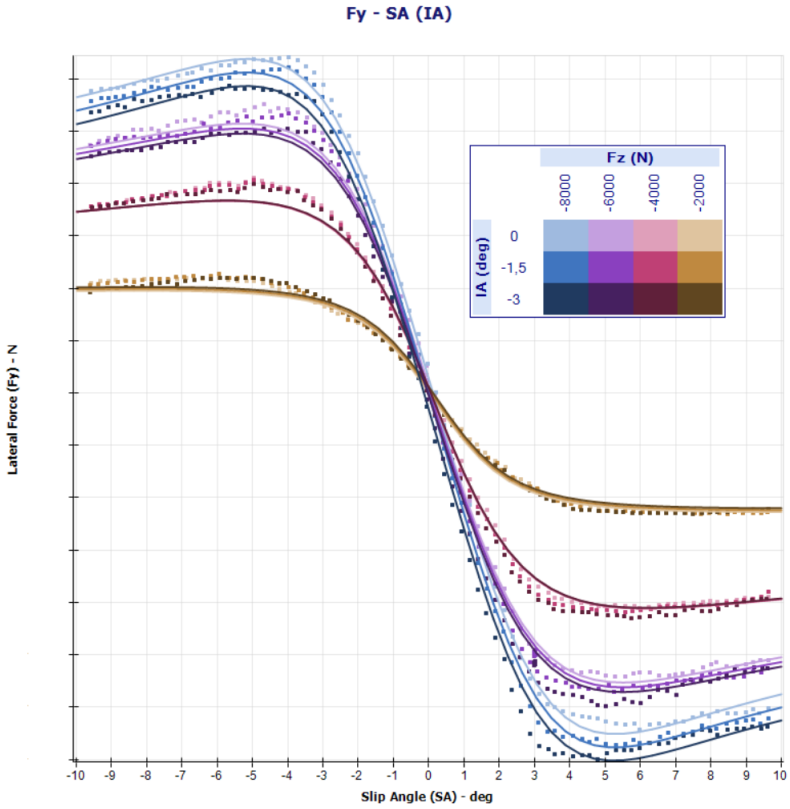

Source – Author(2022)

crossover to get better results. In Figure 29 it's possible to see what an optimization without mutation looks like, the best fitness value is much worse and no fluctuations on the average fitness occur leading to a premature convergence.

### 4.0.2 Particle swarm results

The particle swarm test approach was the same of the genetic algorithm, however much less fiddling with the test setup was needed, since there is much less parameters to tune and the results were great in the first couple of tries. Table 9 shows the tests setups for the standard PSO. The quantum PSO was tested separately but using the same configurations, the only difference is that, instead of using the parameters

Figure 28 – GA resulting tire model Fy x SA curve (values hidden due to confidentiality).




Source – Author(2022)

inertia, max speed, social and cognitive factor, it uses only $\beta_1 = 0.3$ and $\beta_2 = 1.3$.

Table 9 – PSO Test Setup.

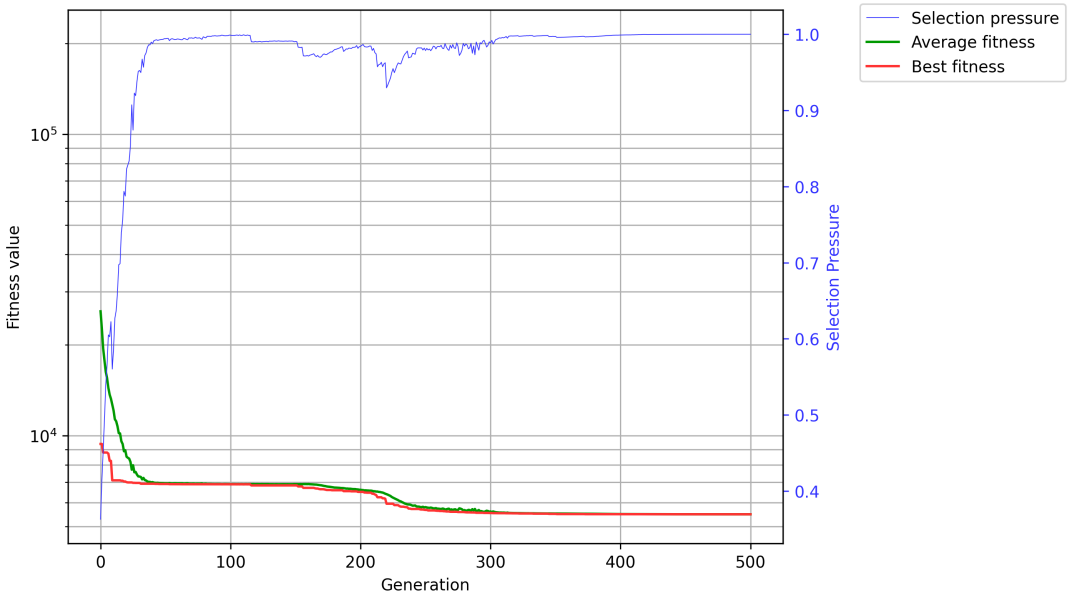
| **Parameter** | **Value** |
|---|---|
| Particle count | 100 |
| Maximum number of iterations | 200 |
| Max speed | 5 |
| Particle Inertia | 0.7 |
| Social Factor | 1.0 |
| Cognitive Factor | 1.4 |

Source – Author (2022)

#### 4.0.2.1 Test function results

The results presented here are separated into standard particle swarm and quantum behavior PSO. Table 10 and Table 11 present the average results of 3 runs for the standard variation on the Rastrigin and Rosenbrock functions respectively. The same goes to Figure 30 and Figure 31.

Figure 29 – GA with no mutation convergence plot.




Source – Author(2022)

Table 10 – PSO Rastrigin test average results.

| **Parameter** | **Value [unit]** |
|---|---|
| Time Elapsed | 219 [ms] |
| Total Evaluation Count | 20100 |
| Time per Evaluation | 10.96 [us] |
| Final Fitness Value | 0 |
| Iteration Where Minima was Found | 169 |
| Final Average Population Fitness | 3.19 |

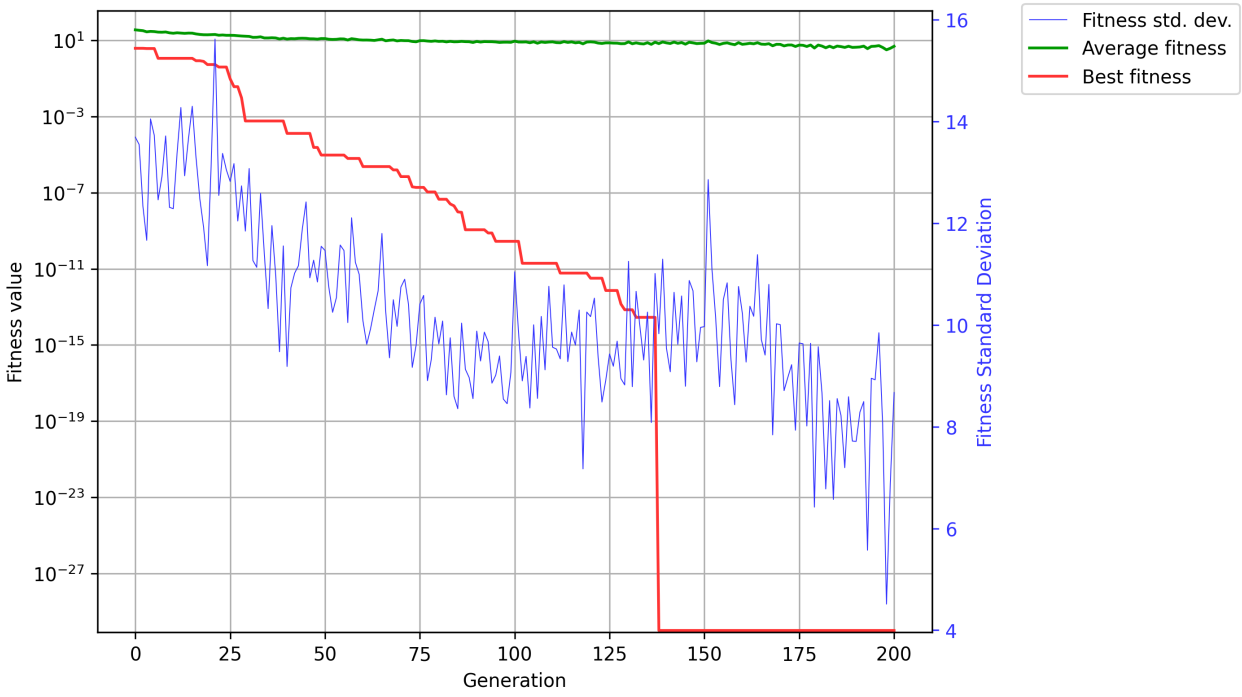
Source – Author (2022)

Table 11 – PSO Rosenbrock test average results.

| **Parameter** | **Value [unit]** |
|---|---|
| Time Elapsed | 222 [ms] |
| Total Evaluation Count | 20100 |
| Time per Evaluation | 11.16 [us] |
| Final Fitness Value | 8.9e-22 |
| Iteration Where Minima was Found | 196 |
| Final Average Population Fitness | 9.33e-08 |

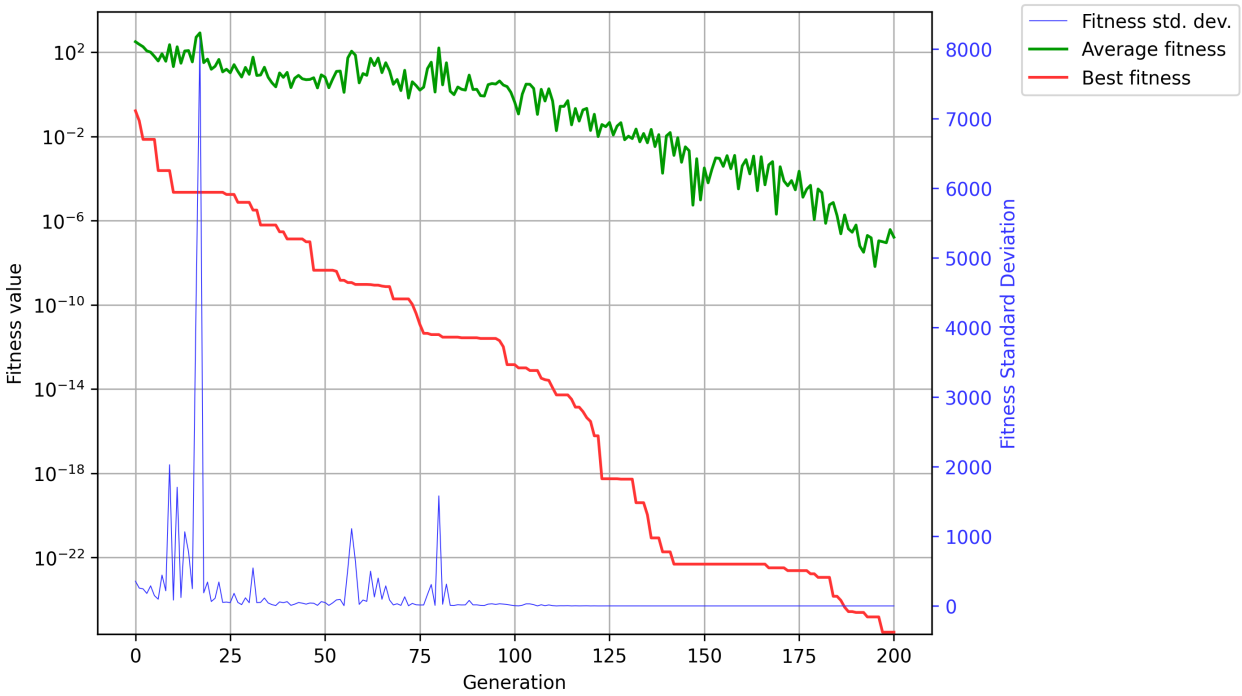
Source – Author (2022)

Figure 30 – Convergence plot of PSO's Rastrigin test function.




Source – Author(2022)

Figure 31 – Convergence plot of PSO's Rosenbrock test function.




Source – Author(2022)

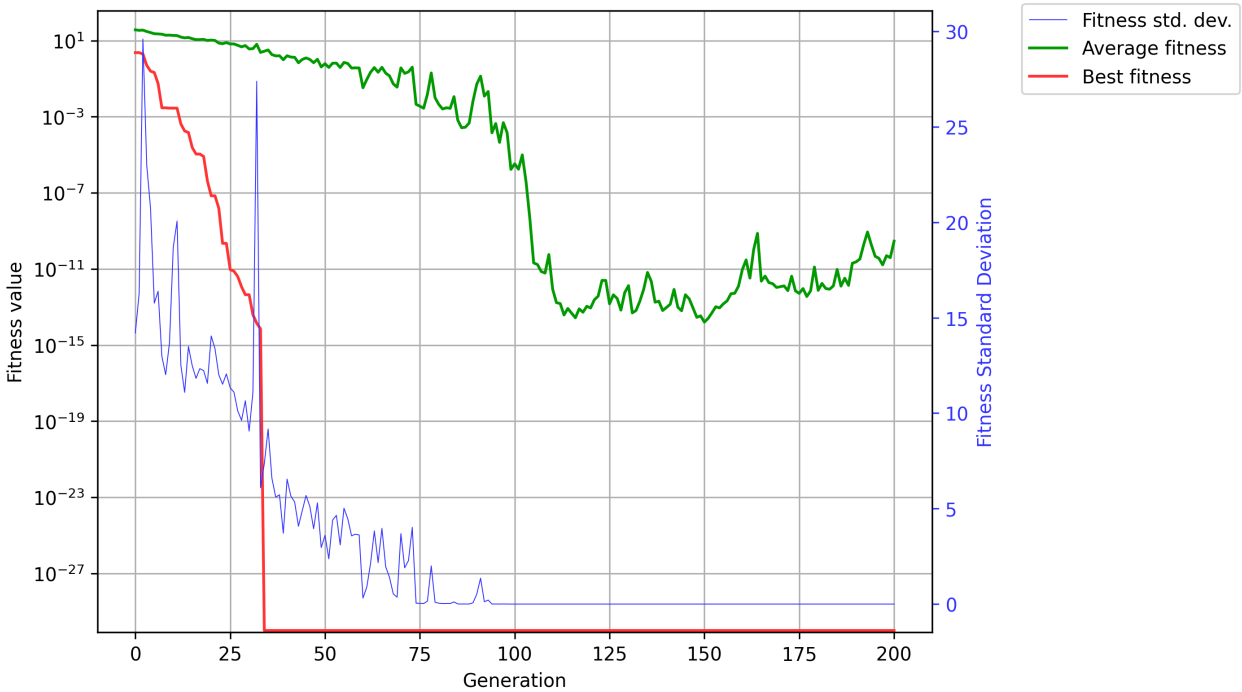
The standard PSO presented a different result in behavior for both test functions. In the Rastrigin function, the global minima is always found, and the Rosenbrock optimization has a convergence behavior instead to suddenly finding the global minima.The quantum PSO (QPSO) has a very different behavior like seen in Figure 32 and Figure 33. Table 12 and Table 13 presents the result of the Rastrigin and Rosenbrock respectively.

Table 12 – QPSO Rastrigin test average results.

| **Parameter** | **Value [unit]** |
|---|---|
| Time Elapsed | 240 [ms] |
| Total Evaluation Count | 20100 |
| Time per Evaluation | 11.8 [us] |
| Final Fitness Value | 0 |
| Iteration Where Minima was Found | 34 |
| Final Average Population Fitness | 1.10e-10 |

Source – Author (2022)

Figure 32 – Convergence plot of QPSO's Rastrigin test function.




Source – Author(2022)

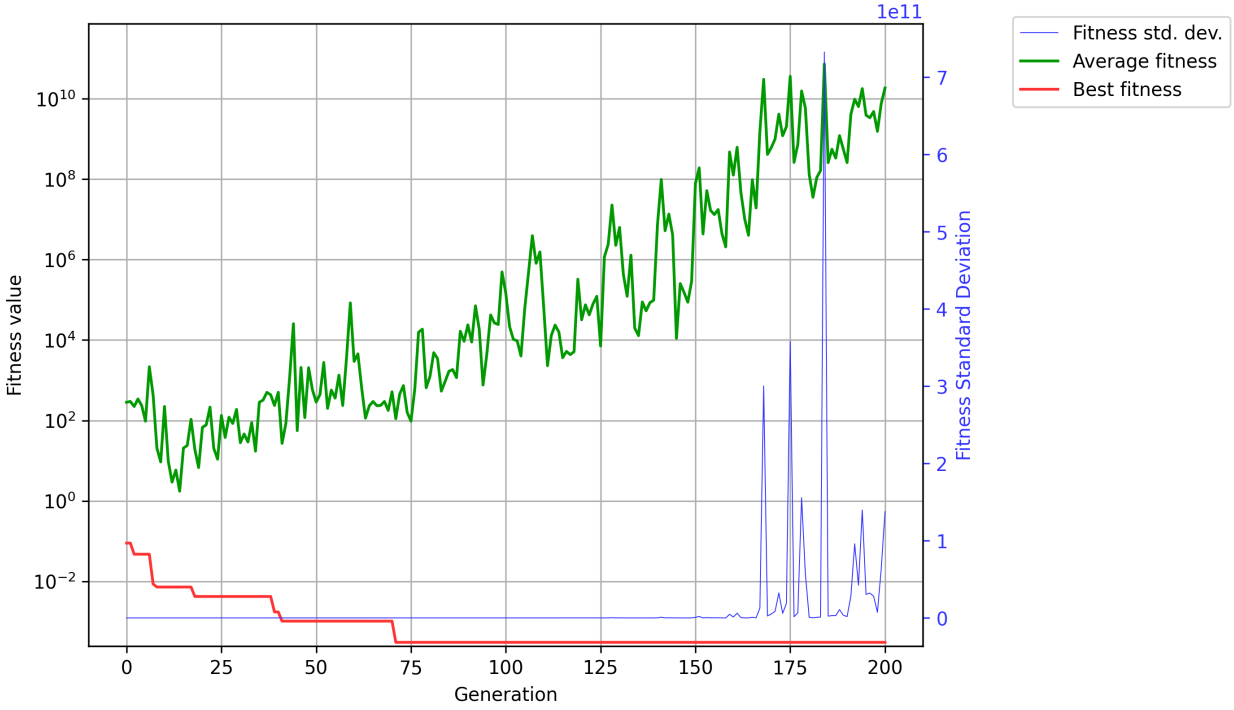
The quantum PSO presented a much better performance for the Rastrigin function, but was not able to find the minima in the Rosenbrock function. The quantum behavior allows the algorithm to easily get out of local minima but in contrast, it is much worse to search low gradient functions.

Table 13 – QPSO Rosenbrock test average results.

| **Parameter** | **Value [unit]** |
|---|---|
| Time Elapsed | 230 [ms] |
| Total Evaluation Count | 20100 |
| Time per Evaluation | 11.06 [us] |
| Final Fitness Value | 8.97e-4 |
| Iteration Where Minima was Found | 33 |
| Final Average Population Fitness | 9.87e+14 |

Source – Author (2022)

Figure 33 – Convergence plot of QPSO's Rosenbrock test function.




Source – Author(2022)

#### 4.0.2.2 Tire fitting results

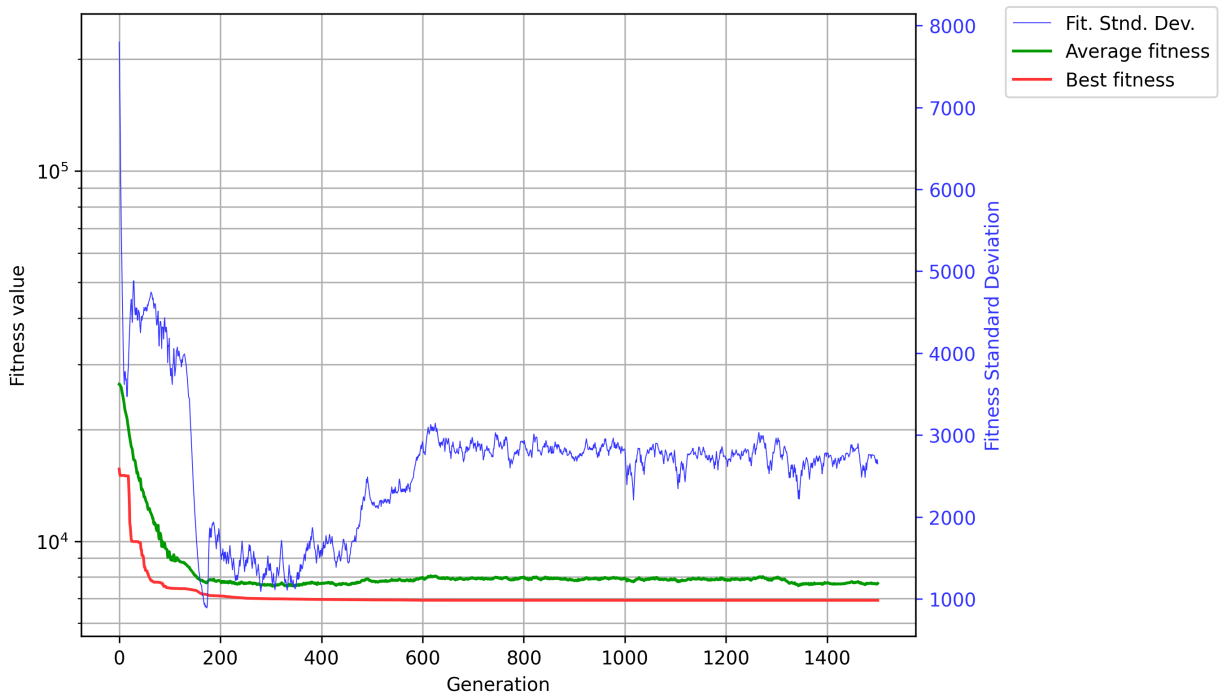
Just like the genetic algorithm, the particle swarm also needed different setup configurations for the tire fitting shown in Table 14. However as seen by the convergence plot (Figure 34 and Figure 35), neither the standard or the quantum PSO, were able to get a good curve fit, not even to plot the results comparing to the collapsed data.Table 15 presents the average optimization values for both standard and QPSO. The possible reasons for this poor performance is discussed in the next section.

Table 14 – PSO Tire Fitting Setup.

| **Parameter** | **Value** |
|---|---|
| Particle count | 30 |
| Maximum number of iterations | 1500 |
| Max speed | 0.5 |
| Particle Inertia | 0.9 |
| Social Factor | 1.0 |
| Cognitive Factor | 1.4 |

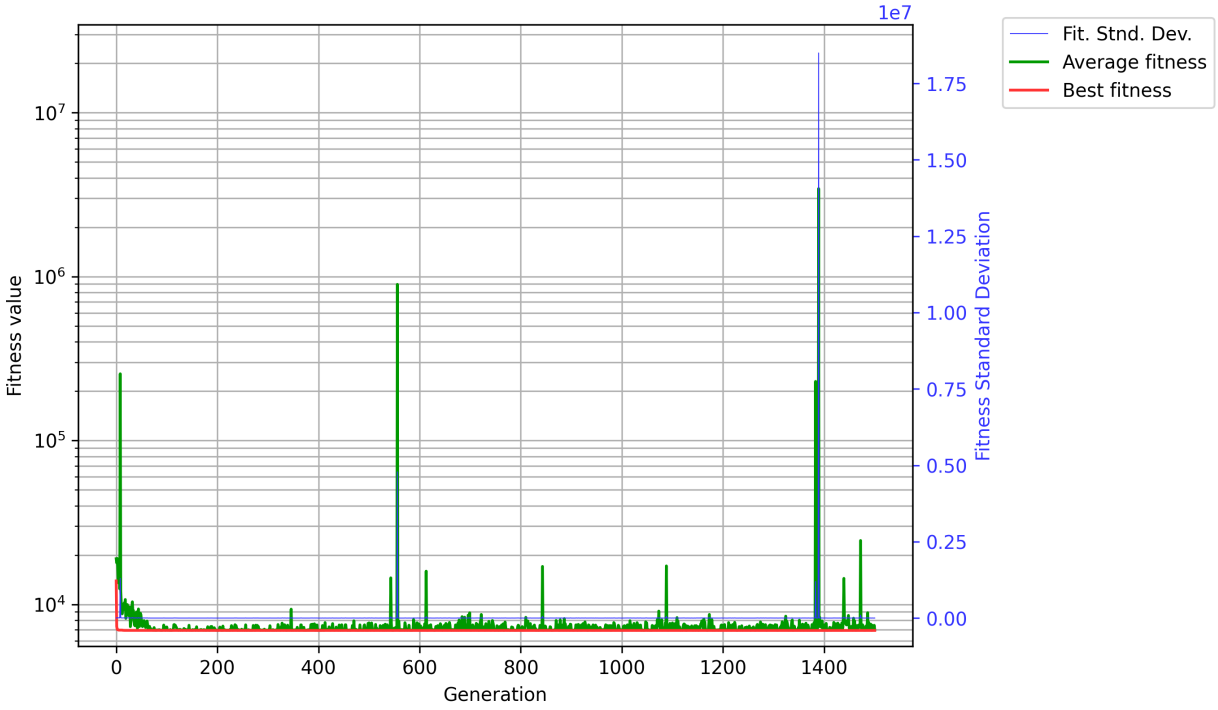
Source – Author (2022)

Figure 34 – Convergence plot of PSO tire fitting.




Source – Author(2022)

Figure 35 – Convergence plot of QPSO tire fitting.




Source – Author(2022)

Table 15 – Standard and QPSO tire fitting results.

| **Parameter** | **PSO Value [unit]** | **QPSO Value [unit]** |
|---|---|---|
| Time Elapsed | 23 [sec] | 22 [sec] |
| Total Evaluation Count | 90000 | 90000 |
| Time per Evaluation | 256 [us] | 244 [us] |
| Final Fitness Value | 6940 | 6880 |
| Iteration Where Minima was Found | 1000 | 200 |
| Final Average Population Fitness | 7200 | 54000 |

Source – Author (2022)

### 4.0.3 Comparison between algorithms

The following section is intended to comment on the comparison of both algorithms results and is divided by test function and tire fitting application.

#### 4.0.3.1 Test functions comparison

Compared to the genetic algorithm, the PSO has a better performance overall when finding the global minima of both functions. Although it takes more time to find the minima in the Rastrigin function, for the Rosenbrock, the values are much better than the GA. Even if the global minima is not found in the 200 test iterations, when allowing more time to search the algorithm is capable of finding it, unlike the GA that very rarely finds the global minima even with more generations, since it's more dependent of it's random nature.

However the GA is much faster than the PSO due to needing only half the evaluation count to get similar behavior and even has faster evaluation calculation. But at the same time, it's much more dependent of the multiple parameters values and implementations, so it is required a skilled optimization engineer to adapt the algorithm to different problems. Table 16 and Table 17 shows the compiled results for all test cases.

Table 16 – Test comparison for Rastrigin function.

| **Parameter** | **PSO Value [unit]** | **QPSO Value [unit]** | **GA Value [unit]** |
|---|---|---|---|
| Time Elapsed | 219 [ms] | 240 [ms] | 66[ms] |
| Total Evaluation Count | 20100 | 20100 | 10150 |
| Time per Evaluation | 10.96 [us] | 11.8[us] | 6.5 [us] |
| Final Fitness Value | 0 | 0 | 0 |
| Iteration Where Minima was Found | 169 | 34 | 64 |
| Final Average Population Fitness | 3.19 | 1.1e-10 | 0.81 |

Source – Author (2022)

Table 17 – Test comparison for Rosenbrock function.

| **Parameter** | **PSO Value [unit]** | **QPSO Value [unit]** | **GA Value [unit]** |
|---|---|---|---|
| Time Elapsed | 222 [ms] | 230 [ms] | 71.6 [ms] |
| Total Evaluation Count | 20100 | 20100 | 10150 |
| Time per Evaluation | 11.16 [us] | 11.06[us] | 6.96 [us] |
| Final Fitness Value | 8.9e-22 | 8.97e-4 | 3.35e-10 |
| Iteration Where Minima was Found | 196 | 33 | 194 |
| Final Average Population Fitness | 9.33e-8 | 9.87e+14 | 0.058 |

Source – Author (2022)

#### 4.0.3.2 Tire fitting comparison

For the tire model fitting application, there is no doubt that the genetic algorithm performed better than the PSO, with it not even being able to get a usable result. The PSO algorithms suffers for being heavily stuck on a local minima, there is no implemented configurations that allows it to escape. The probable main factor for this is the lack of a "turbulence" mechanism, like the mutation for the genetic algorithm. This is visible when looking to Figure 29, where the optimization process of the GA without mutation looks very similar to the PSO, with almost the same results.

The quantum PSO, however, has in it's nature a way of getting out local minima as seen in the test functions and in it's fluctuations of the particle's average fitness throughout the optimization process (see Figure 35). But the bad results may indicate that this mechanism is not suited for the tire fitting problem. So by comparing the tire fitting and test functions results for all the algorithms, a possible approximate "surface" format can be visualized. The GA and QPSO convergence plot and the overall algorithm behavior indicates that the surface has multiple local minima, like the Rastrigin function, but the bad results for the QPSO and the slow and highly mutation dependent convergence for the GA, also indicates that there is a low gradient slope for the global minima, resulting in a mix between the Rosenbrock and Rastrigin functions behavior. With this in mind, it's clear that an algorithm that presents good performance on both test functions and a good adaptability like the GA, would result in a great performance for the tire model fitting task.

A possible way to get better results for tire fitting with the PSO is by implementing a "turbulence" mechanism, which gives the particles a way to get out of a local minima. A potential approach would be by modifying the speed behavior of the particles. Liu and Abraham (2005) proposes a fuzzy adaptive turbulent particle swarm optimization. It is based on a minimal velocity threshold that is tuned adaptively with a fuzzy logic controller. The particle's velocity are kept under a minimal value that changes according to the need of a broader exploration of the solution space. Liu and Abraham (2005) also explains that the bigger the dimension of the optimization problem is, the poorest is the performance of the standard PSO, which comes into agreement with this works results, since, although the test functions were kept with 3 dimensions, the tire fitting problem dimension is related to how many coefficients are being optimized. For the Fy pure case there are 26 coefficients with gives the optimization problem 26 dimensions.

Table 18 – Comparison for tire model fitting.

| **Parameter** | **PSO Value [unit]** | **QPSO Value [unit]** | **GA Value [unit]** |
|---|---|---|---|
| Time Elapsed | 23 [sec] | 22 [sec] | 24 [sec] |
| Total Evaluation Count | 90000 | 90000 | 38025 |
| Time per Evaluation | 256 [us] | 244 [us] | 649 [us] |
| Final Fitness Value | 6940 | 6880 | 234 |
| Iteration Where Minima was Found | 1000 | 200 | - |
| Final Average Population Fitness | 7200 | 54000 | 4099 |

Source – Author (2022)

# 5 CONCLUSIONS

This work presented very efficient and flexible implementations for a genetic and particle swarm optimization algorithms. Both were compared with artificial landscapes and in the tire model fitting application, where a Magic Formula 6.1 pure lateral force model was fitted. For the test functions, the PSO presented a great performance, being easier to implement and resulting in good convergence values. However, due to the high complexity and high dimension number of the tire fitting problem, it was not able to result in a usable model. In contrast, the GA presented a good but slightly worse performance than the PSO on the artificial landscape tests, yet, exhibited great capability when fitting the Magic Formula model. A possible upgrade for the particle swarm algorithm was purposed, where a minimum velocity threshold is implemented, giving the algorithm capable tools for higher dimensional optimization problems.

The speed of both algorithms are similar and very fast overall, taking advantage of the optimizations granted by the C++ language. The particle swarm algorithm has less time per evaluation than the GA, however it also takes double the amount of evaluations to reach the same result. The complexity of the individuals has a big impact on the optimization time. For the test functions, an individual consists in a 3D point, but for the tire modeling application, each individual carries the configuration of a tire model. For the GA, this makes the process of crossover, mutation and selection much more complex and time demanding, highlighting the advantage in simplicity of the PSO algorithm.

The results presented here allow us to understand why the genetic algorithm is widely used in the tire modeling industry, given it's adaptability to multiple kinds of problems and great overall performance to complex optimization problems. A possible expansion for this work would include different implementations of the particle swarm optimization that has numerous variations to help it adapt to different kind of problems. Also, other types of algorithms could be implemented to be tested for the current application. A good contender would be the trust region algorithm or the ant colony optimization. Another focus could be given if a usable result with the modified particle swarm is achieved, where different tire model fitting, beyond the pure lateral force could be applied, presenting a different optimization problem like the combined self aligning torque, loaded radius or effective rolling radius optimizations.

# REFERENCES

ALAGAPPAN, A. Vijay; RAO, K.V. Narasimha; KUMAR, R. Krishna. A comparison of various algorithms to extract Magic Formula tyre model coefficients for vehicle dynamics simulations. **Vehicle System Dynamics**, Taylor  Francis, v. 53, n. 2, p. 154–178, 2015. DOI: 10.1080/00423114.2014.984727. eprint: https://doi.org/10.1080/00423114.2014.984727. Available from: https://doi.org/10.1080/00423114.2014.984727.

AVI, Ariel Gustavo. MUTLI-OBJECTIVE OPTIMIZATION OF SUSPENSION KINEMATICS. Federal University of Santa Catarina. [S.l.], 2021.

BÄCK, T.; BACK, M.D.S.R.F.T.; BACKTHOMAS. **Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms**. [S.l.]: Oxford University Press, 1996. ISBN 9780195099713. Available from: https://books.google.com.br/books?id=9MLnCwAAQBAJ.

CALSPAN. **Tire testing for motorsports and race cars**. 2021. Available from: https://www.calspan.com/services/transportation-testing-research-equipment/tire-performance-testing/motorsports-tire-testing/. Visited on: 19 Mar. 2021.

DARWIN, Charles. **On the Origin of Species by Means of Natural Selection**. London: Murray, 1859. or the Preservation of Favored Races in the Struggle for Life.

DRÉO, J.; CHATTERJEE, A.; PÉTROWSKI, A.; SIARRY, P.; TAILLARD, E. **Metaheuristics for Hard Optimization: Methods and Case Studies**. Berlin, Germany: Springer Berlin Heidelberg, 2006. ISBN 9783540309666.

FSAE TTC. **Formula SAE tire test consortium**. 2005. Available from: https://www.millikenresearch.com/fsaettc.html. Visited on: 25 Apr. 2021.

GROTTI, Ewerton; MIZUSHIMA, Douglas Makoto; BACKES, Artur Dieguez; FREITAS AWRUCH, Marcos Daniel de; GOMES, Herbert Martins. A novel multi-objective quantum particle swarm algorithm for suspension optimization. **Computational and Applied Mathematics**, v. 39, n. 2, 2020. DOI: 10.1007/s40314-020-1131-y.

JAZAR, R. N. **Vehicle Dynamics: Theory and Application**. Bundoora, Australia: Springer New York, 2013. (SpringerLink : Bücher). ISBN 9781461485445.

KINDLER, Eugene; KRIVÝ, Ivan. Object-oriented simulation of systems with sophisticated control. **International Journal of General Systems - INT J GEN SYSTEM**, v. 40, p. 313–343, Jan. 2005. DOI: 10.1080/03081079.2010.539975.

LIU, H.; ABRAHAM, Ajith. Fuzzy adaptive turbulent particle swarm optimization. In: p. 6. DOI: 10.1109/ICHIS.2005.49.

LYNN, Nandar; ALI, Mostafa Z.; SUGANTHAN, Ponnuthurai Nagaratnam. Population topologies for particle swarm optimization and differential evolution. **Swarm and Evolutionary Computation**, v. 39, p. 24–35, 2018. ISSN 2210-6502. DOI: https://doi.org/10.1016/j.swevo.2017.11.002. Available from: https://www.sciencedirect.com/science/article/pii/S2210650217308805.

MEGARIDE. **Megaride Tire Webinar**. [S.l.: s.n.], 2021.

MILLIKEN, W.F.; MILLIKEN, D.L. **Race Car Vehicle Dynamics**. [S.l.]: SAE International, 1995. (Premiere Series). ISBN 9781560915263.

OPTIMUM G. **Optimum G services**. 2021. Available from: https://optimumg.com/services/. Visited on: 19 Mar. 2021.

OPTIMUMG. **Advanced Vehicle Dynamics Seminar**. [S.l.: s.n.], 2018.

PACEJKA, H.B. **Tyre and Vehicle Dynamics**. Burlington, United Kingdom: Butterworth-Heinemann, 2006. (Automotive engineering). ISBN 9780750669184.

PACEJKA, H.B. **Tyre and Vehicle Dynamics**. Burlington, United Kingdom: Butterworth-Heinemann, 2012. (Automotive engineering). ISBN 9780750669184.

RUMPE, B. **Modeling with UML: Language, Concepts, Methods**. [S.l.]: Springer International Publishing, 2016. ISBN 9783319339337.

SAE, Vehicle Dynamics Standards Committee. **Vehicle Dynamics Terminology**. [S.l.], Jan. 2008. DOI: https://doi.org/10.4271/J670_200801. Available from: https://doi.org/10.4271/J670_200801.

SEGERS, Jorge. **Porsche Motor sport Performance Engineering Course**. [S.l.: s.n.], 2019.

SMITH, Gregory; BLUNDELL, Mike. A new efficient free-rolling tyre-testing procedure for the parameterisation of vehicle dynamics tyre models. **Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering**, v. 231, n. 10, p. 1435–1448, 2017. DOI: 10.1177/0954407016675216. eprint: https://doi.org/10.1177/0954407016675216. Available from: https://doi.org/10.1177/0954407016675216.

SUN, Jun; FENG, Bin; XU, Wenbo. Particle swarm optimization with particles having quantum behavior. In: PROCEEDINGS of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753). [S.l.: s.n.], 2004. 325–331 vol.1. DOI: 10.1109/CEC.2004.1330875.

TASS INTERNATIONAL. **Tyre Testing Dedicated on-road test laboratory**. 2021. Available from: https://www.tassinternational.com/tyre-testing. Visited on: 19 Mar. 2021.

WIKIPEDIA. **Wikipedia**. 2022. Available from: https://en.wikipedia.org/. Visited on: 16 Feb. 2022.

ZHUO, Guirong; WANG, Jin; ZHANG, Fengbo. Parameter Identification of Tire Model Based on Improved Particle Swarm Optimization Algorithm. In: SAE Technical Paper Series. [S.l.]: SAE International, Apr. 2015. DOI: 10.4271/2015-01-1586. Available from: https://doi.org/10.4271%2F2015-01-1586.