Development of an Optimization Framework for Suspension Parameters in Automotive Vehicles Using Genetic Algorithms

Iúri Silva[1], Fernando C. Batista[1,2], Olivier R. Gouveia[1,2,3]

[1] ESTG-School of Technology and Management-Polytechnic University of Leiria, Portugal
[2] CDRSP-Center for Rapid and Sustainable Product Development- Polytechnic University of Leiria, Portugal
[3] IDMEC-Institute of Mechanical Engineering-Instituto Superior Técnico, University of Lisbon, Portugal

Abstract. This study presents an optimization framework for automotive suspension systems using Genetic Algorithms (GA) to improve ride comfort and handling performance. Traditional suspension tuning relies on trial-and-error methods, which are time-consuming and suboptimal. The proposed method addresses these limitations by optimizing key suspension parameters—stiffness and damping—using GA, which mimics natural selection processes. The framework is applied to a quarter-car model representing the vertical dynamics of a vehicle. GA optimizes the parameters based on performance metrics such as minimizing body displacement and acceleration. Simulations using sinusoidal and step road profiles demonstrated significant improvements, with reductions of 8,1% and 22,8% in displacement and 5,6% and 66,2% in acceleration for sinusoidal and step disturbances, respectively. These results highlight GA's potential to enhance suspension tuning, outperforming conventional methods. Future work will expand this approach to multi-objective optimization and more complex vehicle models.

Keywords: Optimization, Vehicle Suspension, Genetic Algorithm

1 Introduction

The vehicle suspension system has a direct influence on performance, stability and comfort, for both vehicle and passengers. Usually consisting of springs and dampers, its characteristics vary depending on the type of vehicle and road, requiring specific optimizations to ensure good performance and an optimal driving experience. The complexity of optimizing these suspensions arises from the interaction between multiple variables, being key two parameters, stiffness and damping. Traditionally, the optimization of these parameters has relied on expert knowledge and trial-and-error methods, which can be time-consuming, demand significant resources and cannot ensure a global optimal solution. However, new modeling and optimization technologies, such as genetic algorithms, simplify this process by identifying which are the best parameters for different kinds of scenarios, such as sports driving, comfortable driving or off-road conditions. We propose an optimization framework based on Genetic Algorithms (GA), a powerful search heuristic inspired by the principles of natural selection and genetics [1]. The framework aims to optimize the suspension parameters-stiffness and damping- for automotive vehicles, enhancing both ride comfort and handling performance. Albadr et al. [2] highlighted the importance of genetic algorithms and its efficiency on optimizing systems, which can be applied in optimizing vehicle suspensions systems.

2 Numerical Model

Although exists many mathematical models to represent vertical movement on automotive suspension, the quarter car model is the simplest model according to [3]. This model simplifies the whole problem by analyzing each wheel isolated. A representation of this model is presented in **Fig. 1**.

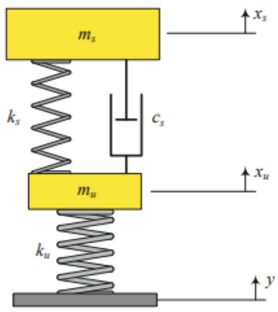

Fig. 1. Quarter Car model [3]

It is represented by two masses, a sprung mass, m_s and an unsprung mass, m_u. The movement of each mass is represented by, x_s, for the sprung mass and m_u for the unsprung mass, while y represents the vertical movement caused by the road profile. The tire stiffness is taken in account represented by, k_u, which connects the unsprung mass to the ground. The sprung mass represents one quarter of the total vehicle mass, while the unsprung mass represents one of the vehicle wheels. A spring with stiffness, k_s, and a damper with damping coefficient, c_s, supports the sprung mass.

The equations of motion that describe the behavior of this model, as referenced by Jazar and Marzbani [3], are equations (1) and (2):

$$m_s\ddot{x}_s = -k_s(x_s - x_u) - c_s(\dot{x}_s - \dot{x}_u) \tag{1}$$

$$m_u\ddot{x}_u = k_s(x_s - x_u) + c_s(\dot{x}_s - \dot{x}_u) - k_u(x_u - y) \tag{2}$$

With this numerical model it is possible to analyze the vertical movement of the suspension and optimize its parameters. Due to the simplicity of this model, it does not require high computational resources, which makes a good option for optimization.

3 Optimization

Optimization can be defined as the science that determines the best solution for a particular mathematical problem that usually represents a real physical system. The objective is to identify the ideal combination of variables that minimize, or in some cases maximize, a determined objective function *f(x)*[4]. According to Fletcher [1] the definition of the optimum criteria, development of algorithms, the study of the structure and computational experiments are part of problem optimization.

Simple optimization problems can be represented as [1]:

$$\textit{Minimize} \quad f(x)$$

$$\textit{Subject to} \quad \begin{cases} h_i(x) = 0, \ i = 1, 2, \ldots, m \\ g_j(x) \leq 0, \ j = 1, 2, \ldots, r \\ x \in \Omega \end{cases}$$

In this formulation x is an "n" dimensional vector with unknown values, $x = [x_1, x_2, \ldots, x_n]$ and $\{f, h_i \ e \ g_j\}$ are real-valued functions that depend on the values of x. The numerical set Ω represents all x possible values inside the imposed search bounds.

3.1 Gradient based methods

Some of the classical optimization methods are gradient based methods, where through an iterative process a search for the minimum value of function *f(x)* is made. These methods start from an initial point, either chosen arbitrarily or by the user, and generate a sequence of approximate solutions. The search direction is determined by the descent direction of the gradient in the objective function, aiming to find a point where *f(x)* value is minimum. A simple method is the gradient descent method, which follows the opposite direction of the gradient [5].The accuracy of this method depends on the step size between iterations, which must be adjusted to avoid premature convergence to local minima.

Other methods like Newton's Method use the Hessian matrix to approximate the objective function as quadratic, enabling more efficient descent directions, although, it may not guarantee fast convergence for non-quadratic functions [6]. Quasi Newton methods improve upon this by iteratively estimating the Hessian matrix, reducing computational cost in large-scale problems, but still requiring matrix storage [4]. Meanwhile, the Conjugate Gradient Method avoids second-derivative calculations, combining gradients in different directions, making it particularly effective for quadratic functions while using less computational storage [4].

3.2 Meta-Heuristics Methods

Meta-heuristic based methods explore large-scale solutions and escape local minima more efficiently than gradient based methods. These methods add an aleatory component in the search of the best solution that minimizes *f(x).* Methods such as evolutionary algorithms are widely used to optimize complex, real world problems. An example of it is the Genetic Algorithm, that simulates Darwin's process of natural evolution [7]. The algorithm begins with an initial population of solutions and, in each iteration, generates a new population based on genetic operators such as crossover and mutation, until it finds a global optimal solution.

Genetic Algorithms

According to Alhijawi and Awajan [8] a simple genetic algorithm starts with an initial population of individuals, each one representing a possible solution to the problem. These individuals have chromosomes composed of genes, any of the individuals are a possible solution of the problem. For example, in binary encoding, a chromosome could look like, x = [10011], where each bit represents a gene. The initial population is randomly generated, allowing the algorithm to explore different regions of the search space based on the imposed restrictions. The algorithm iterates through generations, selecting individuals based on their performance in a fitness function that evaluates how good each solution is. The individuals with best ranking on the fitness function, in each generation, are more likely to be selected to exchange information (genes), with other individuals through crossover and mutation process which will generate more individuals like the best ones of the last generation, obtaining better solutions in each generation. This fitness function is crucial, as it determines which individuals are more likely to reproduce and pass on their genetic information to the next generation. In **Fig. 2** is presented a flow chart representing the operation of a simple genetic algorithm.

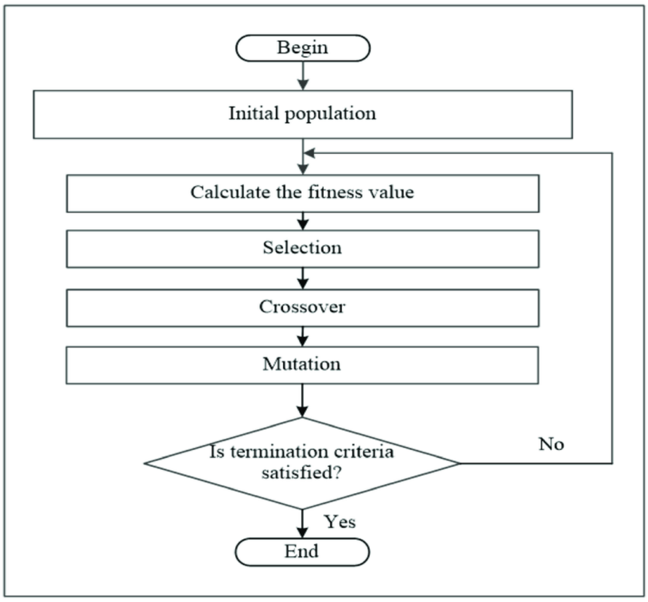

Fig. 2. Flow Chart of a simple Genetic Algorithm [2]

Genetic Operators

Selection: This operator prepares the individuals for reproduction, choosing them based on their fitness scores. Higher fitness individuals are more likely to be selected, but even weaker individuals can be chosen occasionally to maintain diversity. Various selection methods can be used, such as probabilistic selection, where fitter individuals have a greater chance, or stochastic selection, which ensures a wide search by selecting individuals from different areas of the search space.

Crossover: Once individuals are selected, crossover is used to create new individuals by combining genetic material from two parents. In a single-point crossover, a random point on the chromosome is chosen, and the genetic material is swapped between the two parents to generate two new individuals. A double-point crossover works similarly but swaps genes between two random points, increasing diversity.

Mutation: During mutation, random changes are introduced to some genes in the new individuals to maintain diversity in the population. It consists in swapping one or more genes randomly in a chromosome, which helps the algorithm avoid getting stuck in local optima by exploring new areas of the search space. The mutation rate is typically kept low to avoid turning the algorithm into a random search.

Replacement Techniques: After generating a new set of individuals, the algorithm must decide which individuals to retain for the next generation. One approach is to replace the entire population with the offspring, focusing on the assumption that new individuals will improve overall fitness. Alternatively, some of the best individuals from the previous generation can be retained, or the worst individuals in the new generation can be discarded in favor of those generated by crossover and mutation.

4 Case Study

The vehicle model was optimized according to different objective functions, one being to minimize acceleration RMS (Root Mean Square) value of the sprung mass and the other being minimize displacement RMS value of the sprung mass. Additionally, two types of excitations were assessed as road profiles: sinusoidal and step profiles.

4.1 Road Inputs

To analyze the vertical movement of the vehicle suspension, just like presented in [9], the first excitation used was a step road with a pulse of 0,05 m as depicted in **Fig. 3**. The vehicle test speed used was 50 km/h.

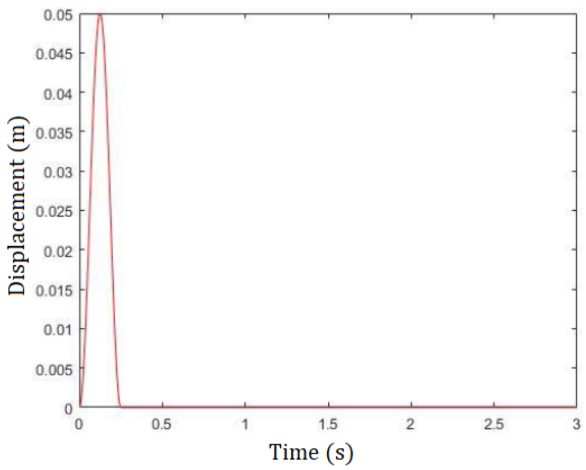

Fig. 3. Step-road profile from Simulink

The second excitation used was a sinusoidal function that depends on the vehicle velocity v_{car} , with an amplitude of 0,01 m and wavelength, λ, of 5 m. The sinusoidal road profile can then be modelled as per equation (3).

$$y = A * \sin\left(\frac{v_{car}/3{,}6}{\lambda} \cdot 2\pi \cdot t\right) \quad [m] \tag{3}$$

Time is represented by the variable t, in seconds, and A is the amplitude of the sinusoidal wave in meters.

4.2 Vehicle Dynamics Modelling

The properties of the quarter car model chosen are presented in **Table 1**, and refer to a real car, a Mercedes-Benz AMG SLC43, adapted to a quarter car model [9].

Table 1. Quarter car model properties

Propriedade	Simbologia	Valor
Sprung mass	m_s	395 kg
Unsprung mass	m_u	38 kg
Spring stiffness	k_s	29300 N/m
Damping coefficient	c_s	3000 Ns/m
Tire stiffness	k_t	290000 N/m

4.3 Optimization problem

The definition of the optimization problem with the objective of the optimization, minimize vertical acceleration and displacement, respectively, and the restrictions are presented as:

Objective Function #1

$$\min_{\mathrm{k}_s, \mathrm{c}_s} RMS\,(\ddot{x}) \quad s.t. \begin{cases} 10000\ \mathrm{N/m} \leq \mathrm{k}_s \leq 200000\ \mathrm{N/m} \\ 500\ \mathrm{Ns/m} \leq \mathrm{c}_s \leq 10000\ \mathrm{Ns/m} \end{cases}$$

Objective Function #2

$$\min_{\mathrm{k}_s, \mathrm{c}_s} RMS\,(x) \quad s.t. \begin{cases} 10000\ \mathrm{N/m} \leq \mathrm{k}_s \leq 200000\ \mathrm{N/m} \\ 500\ \mathrm{Ns/m} \leq \mathrm{c}_s \leq 10000\ \mathrm{Ns/m} \end{cases}$$

A script was created in MATLAB®, which utilized the software's own native libraries "Global optimization Toolbox", to solve the problem with genetic algorithm. The section of the script related to the implementation of the genetic algorithm is now presented and explained.

```
problem.solver   = 'ga()';
problem.fitnessfcn = @(x)V1_quarter_car_funcao([x(1) x(2)]);
problem.nvars = 2;
problem.lb=[10000,500];
problem.ub=[200000,10000];
problem.options = optimoptions('ga','PopulationSize',…
  popsz,'MaxGenerations',maxgen, 'FunctionTolerance',…
  tol, 'CrossoverFraction',xfrac, 'EliteCount',…
  mutfrac*popsz,'SelectionFcn',…
  ,{@selectiontournament,2},'PlotFcn',…
  {@gaplotbestf, @gaplotstopping});
```

The genetic algorithm method is selecting using the function 'ga( )'. In 'problem.fitnessfcn' the input is a Simulink model representative of the quarter car model, which in this case will be the fitness function that evaluate and rank the individuals of each generations and provide the best solution to the problem. The section 'problem.nvars' defines the number of variables in the problem, in this case are k_s and c_s. The 'problem.lb' and 'problem.ub' specify the lower and upper bounds of the variables within the search space, respectively. The first argument in each corresponds to the restrictions for k_s and the second argument for c_s. Finally 'problem.options' is used to define some options about the algorithm in use, which can be changed by the user.

The genetic algorithm parameters used in this study are presented in **Table 2**.

Table 2. Genetic Algorithm Parameters

Property	**Value**
Population size	80
Crossover	70 %
Mutation	10 %
Máx number of generations	100
Tolerance	0,0001 $[m\ or\ m/s^2]$

The choice of these parameters directly influences the effectiveness of the algorithm. Mitchell [10] outlines several studies using genetic algorithms, indicating that the optimal parameters include a population size between 50 and 100 individuals, a crossover rate between 60 and 90%, and a recommended mutation rate below 10%. A high crossover rate means that this percentage of the population from the previous generation exchanges information, generating new individuals similar to the existing ones. In regions where the algorithm has found good solutions, more points are formed. Those that perform better in the fitness function survive and are selected to form new points in the next generation, contributing to the algorithm's convergence. Maintaining a low mutation fraction allows the algorithm to escape local minima by randomly altering a portion of the information, which promotes exploration of new areas in the search space and preserves genetic diversity. This prevents the algorithm from becoming too random, which could otherwise difficult convergence.

5 Results and discussions

In this chapter are presented and discussed the results obtained in the optimization of the quarter car model for both excitations, first step-road and next sinusoidal function.

In **Table 3** are presented the results obtained for step-road optimization.

Table 3. Quarter car model optimization through step-road excitation

	OEM	$RMS\ (\ddot{x})$	$RMS\ (x)$
$RMS\ (\ddot{x})\ [m/s^2]$	1,1032	0,3728	-
$RMS\ (x)\ [m]$	0,00987	-	0,00762
$\mathbf{k}_s\ [N/m]$	29300	10000	10194
$\mathbf{c}_s\ [Ns/m]$	3000	516	2461
Improvement [%]	-	**66,21**	**22,79**

As we can observe in both optimizations-acceleration and displacement-the spring stiffness parameter is relatively close to the minimum limit set in the problem definition. This suggests that the drastic reduction of this parameter contributes significantly to minimizing both acceleration and displacement. The proximity of this parameter to the minimum constraint value indicates that a better solution could potentially be achieved if this value were further reduced. However, extremely low values might not be physically feasible for automotive suspensions, as components with such characteristics might not exist.

In **Fig. 4** it is presented the difference between the acceleration value throughout the simulation in the original (OEM) and optimized model for step road excitation.

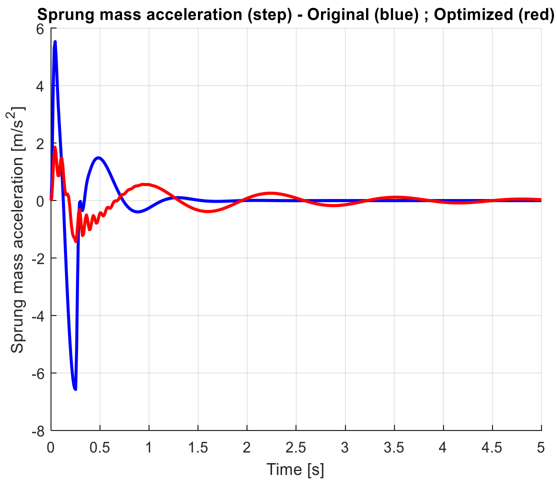

Fig. 4. Comparison of the acceleration OEM vs Optimized (step)

The difference between the highest peak of both models (original and optimized) is easily detected. The optimized model reaches a maximum vertical acceleration just below 2 m/s^2 against the almost 6 m/s^2 in the original model. Although the system with the original parameters reaches equilibrium sooner, around 1,5 seconds compared to the 4,5 seconds for the optimized system, the latter shows a better RMS response in reducing vertical acceleration. The optimized model demonstrates a 66,21% improvement over the original.

In **Fig. 5** it is presented the difference between the displacement value throughout the simulation in the original (OEM) and optimized model for the step road excitation.

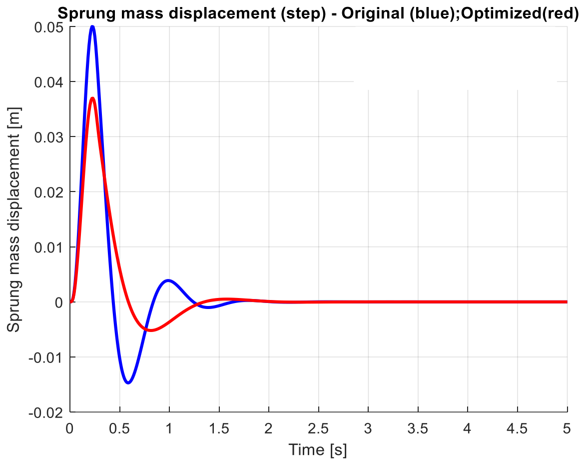

Fig. 5. Comparison of the displacement OEM vs Optimized (step)

Once again, the difference between the maximum values obtained in each system, original and optimized, is clear. In the original configuration, the maximum displacement of the sprung mass was around 0,05 m, while in the optimized model, the maximum displacement is slightly above 0,035 m, representing a reduction of approximately 30% in the maximum displacement value. It is also observed that, although both systems return to equilibrium at almost the same time, around 2 seconds after the excitation, the movement of the sprung mass in the optimized system is much smoother allowing the RMS response of the sprung mass displacement to be lower than in the original configuration. In **Table 4Table 3** are presented the results obtained for sinusoidal excitation optimization.

Table 4. Quarter car model optimization through sinusoidal excitation

	OEM	$RMS\ (\ddot{x})$	$RMS\ (x)$
$RMS\ (\ddot{x})\ [m/s^2]$	0,07294	0,06885	-
$RMS\ (x)\ [m]$	0,007808	-	0,007172
$\mathbf{k}_s\ [N/m]$	29300	10000	199994
$\mathbf{c}_s\ [Ns/m]$	3000	4836	8936
Improvement [%]	**-**	**5,61**	**8,14**

As we can observe in the acceleration optimization the spring stiffness value coincides once again with lower bound limit for this parameter, while the spring damping coefficient value rises slightly to achieve the best solution. This suggests that lowering the spring stiffness has more impact in minimizing acceleration for the sinusoidal excitation. In the displacement optimization the spring stiffness value is relatively close with the upper bound limit of this parameter as well as the damping coefficient. For minimizing the displacement of the sprung mass, both parameters have a high impact on solution. In **Fig. 6** it is presented the difference between the acceleration value throughout the simulation in the original (OEM) and optimized model for the sinusoidal excitation.

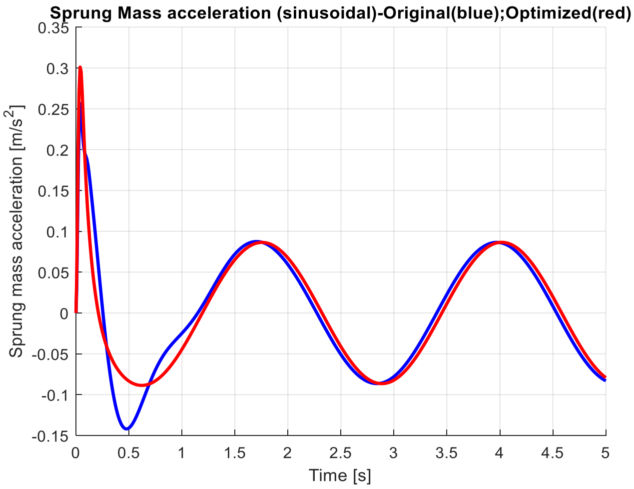

Fig. 6. Comparison of the acceleration OEM vs Optimized (sinusoidal)

This time, although the optimized model initially shows a higher peak acceleration of the sprung mass, around 0,3 m/s^2 compared to approximately 0,25 m/s^2 in the original configuration, the optimized model quickly adjusts to the sinusoidal excitation conditions, reducing the downward overshoot and providing a 5,61% reduction in the RMS response of the sprung mass acceleration to this excitation.

In **Fig. 7** it is presented the difference between the displacement value throughout the simulation in the original (OEM) and optimized model for the sinusoidal excitation.

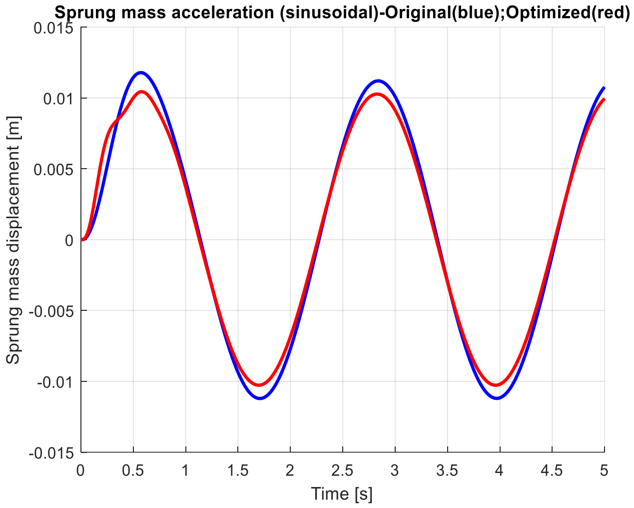

Fig. 7. Comparison of the displacement OEM vs Optimized (sinusoidal)

In this case the difference between the peak displacements of the sprung mass is not so clear. However, the maximum displacement in the original configuration reaches 0,012 m, while in the optimized model, the maximum displacement is around 0,01 m. The optimized system provided an 8,14% improvement in the RMS response of the sprung mass displacement compared to the original configuration, contributing to a smoother ride.

6 Conclusions

A quarter car model representing a vehicle suspension system was successfully optimized with genetic algorithms. Significant improvements in the performance were presented in response to the proposed excitations. Something that could have been done is conducting laboratory tests on the suspension systems under study to corroborate the results obtained from the optimization. It was found that the effectiveness of the suspension parameter optimization process directly depends on the variable being optimized.

For the step road excitation, the spring stiffness had more direct impact to minimize both RMS responses on acceleration and displacement, where the recommended values coincide with the lower boundaries of the optimization problem. The damping coefficient was slightly lower for the displacement optimization. For the acceleration optimization the damping coefficient was lowered until the lower boundary value preestablished on algorithm parameters.

For the sinusoidal function excitation, although the spring stiffness had a bigger difference in the optimized models in relation to the original characteristics, the increase of damping coefficient allowed the optimization of the system. So, in this case both parameters had impact on the optimization, for both excitations, with more emphasis in spring stiffness variation. On acceleration optimization the spring stiffness value recommended is close to the minimum limit of this parameter established on the algorithm properties. But to optimize the model minimizing the displacement of the sprung mass, it is indicated a bigger spring stiffness value relatively close to the maximum limit of the parameter.

It is important to keep in mind that, although the parameter values in the optimized systems can coincide with the constraint values and better solutions could exist, some mechanical elements for suspension systems with that characteristics might not exist in real life.

Future work will focus on the development of more complex models and consider multi-objective optimization of the system. Analysis of other types of excitations, such as optimizing suspension parameters in the context of motorsport can be possible as well.

References

[1] Fletcher Roger, *Practical Methods of Optimization*, Second. John Wiley & Sons, 1987.

[2] M. A. Albadr, S. Tiun, M. Ayob, and F. Al-Dhief, "Genetic algorithm based on natural selection theory for optimization problems," *Symmetry (Basel)*, vol. 12, no. 11, pp. 1–31, Nov. 2020, doi: 10.3390/sym12111758.

[3] R. N. Jazar and H. Marzbani, *Vehicle Vibrations*, 1st ed. Cham: Springer International Publishing, 2024. doi: 10.1007/978-3-031-43486-0.

[4] E. Gouvêa, "MÉTODOS CONVERGENTES DE OTIMIZAÇÃO GLOBAL BASEADOS NO VETOR Q-GRADIENTE," Instituto nacional de pesquisas espaciais - INPE, São José dos campos, 2016.

[5] J. Nocedal and S. J. Wright, *Numerical Optimization*, Second. Springer, 2006.

[6] P. M. Pardalos and M. G. C. Resende, *Handbook of applied optimization*, 1st ed. Oxford University Press, 2002.

[7] C. Darwin, *The origin os species*. P. F. Collier & Son, 1909.

[8] B. Alhijawi and A. Awajan, "Genetic algorithms: theory, genetic operators, solutions, and applications," *Evol Intell*, Feb. 2023.

[9] S. Ebrahimi-Nejad, M. Kheybari, and S. V. N. Borujerd, "Multi-objective optimization of a sports car suspension system using simplified quarter-car models," *Mechanics and Industry*, vol. 21, no. 4, 2020, doi: 10.1051/meca/2020039.

[10] M. Mitchell, *An introduction to genetic algorithms*. Massachusetts Institute of Technology, 1996.