ANNALS OF THE UNIVERSITY OF ORADEA
Fascicle of Management and Technological Engineering
ISSUE #3, DECEMBER 2014, http://www.imtuoradea.ro/auo.fmte/

MATHEMATICAL MODEL AND SOFTWARE SIMULATION OF SUSPENSION'S SYSTEM FROM OPEL CARS

Lucian ROMAN[1], Adrian FLOREA[2], Ileana Ioana COFARU[3]
[1] Industrial Engineering and Management Department, 'Lucian Blaga' University of Sibiu, lucian.roman@autohaushuber.ro
[2] Computer Science and Electrical Engineering Department, 'Lucian Blaga' University of Sibiu, adrian.florea@ulbsibiu.ro
[3] Industrial Engineering and Management Department, 'Lucian Blaga' University of Sibiu, ioanalogistictop@gmail.com

***Abstract*—**For increasing the reliability of the suspension system we propose an optimal design solution based on evolutionary algorithms, that finds parameters of stability and comfort under different exploiting conditions, minimizing the discomfort during movement on uneven roads. We described the mathematical model followed by C# software implementation of the quarter-car suspension model with two-degrees-of-freedom customizing on OPEL cars. Our Genetic Algorithm proposes finding the optimal values for coefficients of stiffness and damping of sprung body, in order to minimize the maximum bouncing acceleration of the sprung mass and minimize the average suspension displacement. This work has the merit that successfully combines a number of different scientific fields such as Computer Science (Artificial Intelligence, Programming Languages, and Distributed Computing), Mathematics (Differential Equations), Transportation Engineering (Vehicle Design), Mechanical Engineering (Vibrations) to solve a real problem, even critical in Romania, characterized by a developing infrastructure and poor quality roads.

***Keywords*—**Differential Equations System, Quarter-Car Model, Software Application, Suspension System.

I. INTRODUCTION

IN our previous work [1] we studied the operational reliability at Opel cars in order to forecast their failure. The conclusion was that, for vehicles which operate in Romania, the suspension system components represent the vulnerable point of the whole vehicle with serious implications for security and passenger safety. For this reason, we are concerned to improve suspension's system by optimizing the stiffness and damping coefficients in order to minimize the maximum bouncing acceleration of the sprung mass and minimize the average suspension displacement during movement on random roads, especially with bumps and potholes. We analyzed the quarter-car model (QCM) with two-degrees-of-freedom (2DOF) [2], [3] in order to find optimal parameters of stability and to keep high standard of ride comfort under different exploiting conditions, minimizing the discomfort during movement.

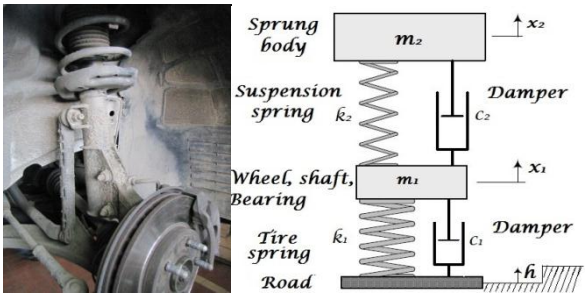
For solving these issues, in this paper we describe and software implement the mathematical model of suspension with the help of differential equations and introduce a non-Pareto multi-objective optimization technique (weighted-sum approach). The developed software application was implemented in Microsoft Visual Studio 2012 (C#), .NET Framework 4.5, using DotNumerics a numerical library for .NET [4]. The main advantage of our application is represented by the features of customization and extensibility; it is not restricted to certain functions and optimization methods implemented in simulation environments like Matlab or Simulink, many other options might be added.

The organization of the rest of this paper is as follows. In section II we review the basic components and functions of a vehicle suspension's system. Section III describes the mathematical model of the dynamic system in order to software implement. It provides programming details on solving the system of differential equations by "Runge-Kutta" method and, determines the maximum bouncing acceleration of the sprung mass and the average suspension displacement on the sample interval. Section IV illustrates some simulation results on different OPEL car models, whereas, section V suggests directions for future work and concludes the paper.

II. THE SUSPENSION SYSTEM: COMPONENTS, FUNCTIONS

The vehicle's suspension system is a dynamic mechanism that connects the body (sprung mass) and wheels of the vehicle [5], [6]. The suspension is designed to transmit toward the running surface all forces acting on the car, and at the same time, it isolates forces arising from road, thus ensuring a high level of manageability and convenience. A running car produces vibration due to multiple environment disturbances such as: unevenness surfaces, aerodynamic forces, vibrations of car components, weather conditions. These vibrations are

harmful to whole human body (stomach, head, chest, heart, thorax, etc) having a negative impact on comfort experienced by passengers on the move. A suspension system must meet the following main functions: comfort, safety and handling. Comfort is a wellbeing state of the human body. In order to obtain it is required to respect the constraints arising from kinematic considerations given that the wheels follow an uneven road. Thus, is required minimizing the vertical acceleration and vertical relative displacement between sprung mass and wheel and keeping vibration levels of the suspension system below the threshold imposed by the natural frequency established by ISO 2631 standard [2]. The safety function aims to protect the passengers in different driving conditions, reacting to control forces produced in case of sudden braking, or when arise unexpected situations. The handling function of suspension system refers to the skill of keeping the contact with the road with minimal load variations and resisting roll of the chassis during movement. For describing the vehicle's dynamic model we chose a linear quarter-car model with two degrees of freedom (sprung body and wheel) that is stimulated by a roughness road profile [7] (Fig. 1).



(a) (b)

Fig. 1. The suspension system of OPEL Astra I model (a –mechanical view; b – QCM physical view).

The QCM is a basic model used to simulate the vehicle's suspension system performance. The QCM name is due to the fact that total vehicle mass is divided equally on the four wheels. In the simplest form, the model is composed from a solid sprung mass (m_2) representing 1/4 of the body of the vehicle and unsprung mass (m_1) which is the mass of one wheel of the vehicle. The main suspension consists in a spring of stiffness (elastic constant) k_2, and a shock absorber with damping coefficient C_2 for supporting the sprung mass. The unsprung mass is directly contacting with the road through a spring having its tire stiffness (k_1) and damping (C_1) [5]. The spring task is to support the vehicle's weight while the damper is designed to dissipate the vibrations and the forces that are transmitted from the running surface unevenness. The chosen values for the coefficients of elasticity and damping determine the behavior of the suspension for different characteristics of the vehicle and of the road. Both masses, of the sprung body and of the wheel, are considered to be rigid, and vertically move with x_2 (sprung mass vertical offset) and x_1 (unsprung mass vertical offset) displacements, respectively, considering a road profile h [1], [7].

III. Modeling and Software Implementation of The Dynamic System

A. The mathematical model

The mathematical model used for describing and studying of the suspension is provided by a system of two simple differential equations of order 2. Differential equations constitute a major field of study in mathematics with wide applicability in problems of engineering (mechanic, electrical circuits, automata theory, etc.). With their help it is studied the evolution of processes that are deterministic, differentials and dimensional finite. An ordinary differential equation (ODE) is a differential equation that describes the predetermined relationship between an unknown function, its arguments and its ordinary derivatives. The order of a differential equation is given by the number of the highest derivative. A linear first order differential equation (or of order 1) is a differential equation in which the unknown function is a function of a single independent variable.

One of the usual methods for approximating solutions of differential equations is the "Runge-Kutta" numerical method having different orders of accuracy, developed around 1900 by the German mathematicians, C. Runge and M.W. Kutta [8]. The p order solution obtained by "Runge-Kutta" method is equivalent to the Taylor series expansion up to d^p, where d is the difference between two points on the axis of the independent variable. Due to the high accuracy obtained it is preferred the "Runge-Kutta" method of order 4 (p=4).

Returning to the model of the suspensions system, a mathematical analysis theorem says that a system of high order differential equations can be transformed into a first order differential equations system by introducing new unknown functions [8], [9]. Generally, a system of k differential equations having k unknown functions, of order o_1, o_2, ..., o_k, turns into an first order system with $o_1+o_2+...+o_k$ differential equations. Whereas in many scientific papers the model is presented only by the matrix equation of the kinematics system without providing of solving details, in the following we chose to present a method for solving the differential system of equations of the second order. We proceeded like this because, although there are some Matlab implementations or a standalone library for developers, in Visual Studio 2012 C#, that solves systems of differential equations by "Runge-Kutta" method, these are only first order equations. Thus, it is required transforming the differential equations system of order 2, into a first order system by introducing two new functions, and writing it as matrix equation: $z'(t)=A \cdot z(t)$, where $z'(t)$ is the derivative functions array and $z(t)$ is the unknown functions array, and A is the square matrix of whose elements should passed as inputs to our implemented software applications.

The kinetic of the 2DOF QCM are governed by the following matrix equation:

$$\mathbf{M}\cdot\ddot{\mathbf{x}}+\mathbf{C}\cdot\dot{\mathbf{x}}+\mathbf{K}\cdot\mathbf{x}=\mathbf{f(t)} \quad (1)$$

where $x(t) = (x_1 \quad x_2)^T$, is the response vector and M, C and K respectively, are mass matrix, damping matrix and stiffness matrix. The vector f(t) represents the forcing terms, arising from the road unevenness. Besides the aforementioned matrixes the system has as input the road profile h and as outputs the sprung mass vertical offset – x_2 and unsprung mass vertical offset – x_1.

$$M=\begin{pmatrix} m_1 & 0 \\ 0 & m_2 \end{pmatrix}, C=\begin{pmatrix} c_1+c_2 & -c_2 \\ -c_2 & c_2 \end{pmatrix} \text{ and } K=\begin{pmatrix} k_1+k_2 & -k_2 \\ -k_2 & k_2 \end{pmatrix}.$$

Thus, (1) become:

$$\begin{pmatrix} m_1 & 0 \\ 0 & m_2 \end{pmatrix}\begin{pmatrix} \ddot{x}_1 \\ \ddot{x}_2 \end{pmatrix}+\begin{pmatrix} c_1+c_2 & -c_2 \\ -c_2 & c_2 \end{pmatrix}\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix}+\begin{pmatrix} k_1+k_2 & -k_2 \\ -k_2 & k_2 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}=\begin{pmatrix} k_1\cdot h \\ 0 \end{pmatrix}$$

Let's consider $\alpha = k_1 \cdot h$ and $\beta = 0$ and we propose to solve the system of differential equations with constant coefficients, in the general case, inhomogeneous.

$$\begin{cases} \mathbf{m_1x_1''+(c_1+c_2)x_1'-c_2x_2'+(k_1+k_2)x_1-k_2x_2=\alpha} \\ \mathbf{m_2x_2''-c_2x_1'+c_2x_2'-k_2x_1+k_2x_2=\beta} \end{cases} \quad (2)$$

The first step in solving consists in change depending, by insertion of two new functions y_1 and y_2, in order to transform the equations system into a homogeneous one.

Let $x_1 = y_1 + P, x_2 = y_2 + Q$ (P, Q constants). We substitute x_1 and x_2 in (2) and determine P and Q so that the system became homogeneous. Thus,

$$P=\frac{\alpha+\beta}{k_1} \text{ and } Q=\frac{\alpha k_2+\beta(k_1+k_2)}{k_1k_2}$$

So, after change depending, using the determined value of constants P and Q, respectively, and transformation:

$$x_1=y_1+\frac{\alpha+\beta}{k_1}, x_2=y_2+\frac{\alpha k_2+\beta(k_1+k_2)}{k_1k_2},$$ the system defined in (2) becomes:

$$\begin{cases} \mathbf{y_1''+\frac{(c_1+c_2)}{m_1}y_1'-\frac{c_2}{m_1}y_2'+\frac{(k_1+k_2)}{m_1}y_1-\frac{k_2}{m_1}y_2=0} \\ \mathbf{y_2''-\frac{c_2}{m_2}y_1'+\frac{c_2}{m_2}y_2'-\frac{k_2}{m_2}y_1+\frac{k_2}{m_2}y_2=0} \end{cases} \quad (3)$$

We introduce the functions z_0, z_1, z_2, z_3 according with the following notations:

$$\begin{cases} z_0=y_1 \\ z_1=y_1' \\ z_2=y_2 \\ z_3=y_2' \end{cases}, \text{ where it can be notice that : } \begin{cases} z_0'=z_1 \\ z_2'=z_3 \end{cases}$$

By the substitutions introduced in (3) the order of differential equations is reduced from 2 to 1.

$$\begin{cases} z_0'=z_1 \\ z_1'=-\frac{(k_1+k_2)}{m_1}z_0-\frac{(c_1+c_2)}{m_1}z_1+\frac{k_2}{m_1}z_2+\frac{c_2}{m_1}z_3 \\ z_2'=z_3 \\ z_3'=\frac{k_2}{m_2}z_0+\frac{c_2}{m_2}z_1-\frac{k_2}{m_2}z_2-\frac{c_2}{m_2}z_3 \end{cases}$$

$$\begin{pmatrix} \dot{z}_0 \\ \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \end{pmatrix}=\begin{pmatrix} 0 & 1 & 0 & 0 \\ -\frac{k_1+k_2}{m_1} & -\frac{c_1+c_2}{m_1} & \frac{k_2}{m_1} & \frac{c_2}{m_1} \\ 0 & 0 & 0 & 1 \\ \frac{k_2}{m_2} & \frac{c_2}{m_2} & -\frac{k_2}{m_2} & -\frac{c_2}{m_2} \end{pmatrix}\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} \quad (4)$$

Thus, we brought the system to the desired form:

$\dot{z} = A \cdot z$, where $A \in M_4(\Re)$

As per a mathematical analysis theorem regarding to systems of homogeneous linear differential equations with constant coefficients, the solution of (4) is the form:

$$\mathbf{z(t)=e^{tA}C} \quad (5)$$

where $C = z(0) = \begin{pmatrix} z_0(0) \\ z_1(0) \\ z_2(0) \\ z_3(0) \end{pmatrix} = \begin{pmatrix} y_1(0) \\ y_1'(0) \\ z_2(0) \\ y_2'(0) \end{pmatrix}$ and,

e^{tA} is the matrix exponential of A,

$$\mathbf{e^{tA}=\sum_{m\geq 0}\frac{t^m}{m!}A^m} \quad (6)$$

$\Rightarrow$ If e^{tA} is computed then it may find z_0, $z_2 \Rightarrow x_1$, x_2.

Since the matrix exponential is a Taylor series expansion, and because from computer science viewpoint any algorithm must be finite, we set a Calculation Error. Thus, if a new term added to the existing series has a norm lower than the Calculation Error, then the algorithm stops. Let's consider the general term of the series:

$$S_m=\sum_{k=0}^{m}\frac{t^k}{k!}A^k \Rightarrow$$

$$\mathbf{S_{m+1}=\sum_{k=0}^{m}\frac{t^k}{k!}A^k+\frac{t^{m+1}}{(m+1)!}A^{m+1}=S_m+\frac{t^{m+1}}{(m+1)!}A^{m+1}} \quad (7)$$

Recall the property that the norm ($\|\cdot\|$) of product of two matrices is less than or equal to the product of matrix norms. Thus,

$$\mathbf{\|A\cdot B\|\leq\|A\|\cdot\|B\|} \quad (8)$$

From (7) results:

$$\mathbf{\|S_{m+1}-S_m\|=\frac{t^{m+1}}{(m+1)!}\|A^{m+1}\|\leq\frac{t^{m+1}}{(m+1)!}\|A\|^{m+1}} \quad (9)$$

Taking $A=(a_{ij}), i, j=\overline{1,4}$, then the norm of matrix A may be one of the following:

i. $$\|A\| = \sum_{i,j=\overline{1,4}} a_{ij}^2 \quad (10)$$

ii. $$\|A\| = \max_{i=1,4} \sum_{j=1}^{4} |a_{ij}| \quad (11)$$

iii. $$\|A\| = \max_{j=1,4} \sum_{i=1}^{4} |a_{ij}| \quad (12)$$

In the study of linear systems [10], the obtained errors in the calculation of exponential matrix are translated, by means of reverse propagation in the disturbance in the input data, in our case the matrix A. Thus, in order to limit the sensitivity of matrix e^{tA} depending with the variation of values from A is necessary to impose the limit $t \cdot \|A\| \le 1$.

Furthermore, by imposing the condition

$$\frac{t^{m+1}}{(m+1)!}\|A\|^{m+1} < \text{Calculation Error} \quad (13)$$

we determine the value of m, resulting the sum of S_m series which will be replace first in (6) and then in (5).

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} = S_m C = \sum_{k=0}^{m} \frac{t^m}{k!} A^k \begin{pmatrix} z_0(0) \\ z_1(0) \\ z_2(0) \\ z_3(0) \end{pmatrix} \quad (14)$$

Finally, we determine z_0, z_2 and obtain x_1, x_2.

B. Software implementation of QCM in C#

Naturally, after description of the mathematical model, we develop a software application that implements it in order to compute the objectives: acceleration of the sprung mass and the suspension displacement during movement on random roads. Fig. 2 presents the software application's flowchart.

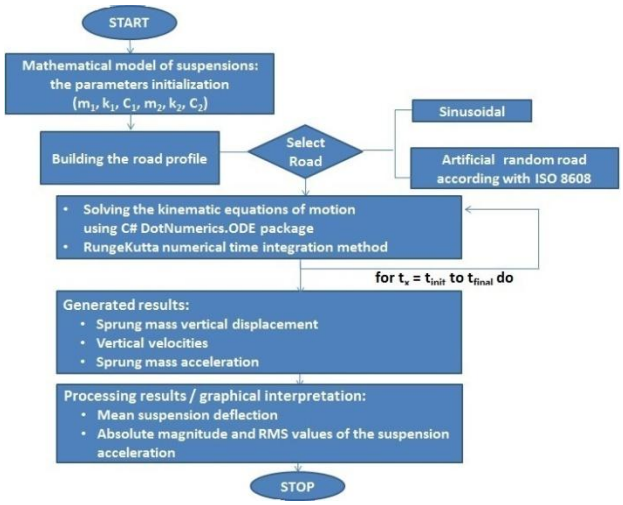


Fig. 2. Flowchart for solving the kinematic equations of the motion for a certain road profile

The first stage in our software application consists in setting the initial values for suspensions' system parameters. We also establish the output variables (vertical displacement and acceleration). Then we have to build the road profile [1] and select the type of simulated road: sinusoidal or random with unevenness, set the initial conditions, the number of iterations (time), the road length, the sampling interval, car velocity. The next stage aims solving the kinematic equations of motion using C# DotNumerics.ODE package by "Runge-Kutta" numerical time integration method. The program ends after executing of the predetermined number of iterations (after completing the entire road and calculation of the targets in each point of the interval). The output results, depending on time, are displacements of the sprung mass, the velocities and the accelerations. These will be further processed computing the mean suspension deflection, the absolute magnitude and root mean square value (RMS) of the suspension acceleration. Finally are graphical represented and qualitatively interpreted.

For the implementation simplicity we chose to extend DotNumerics.ODE [4] which is an integrated library in the Visual Studio 2012 development environment [11]. It implements classes with attributes and methods, in order to solve systems of linear differential equations by the "Runge-Kutta" numerical method, with the fourth order accuracy. Also, DotNumerics.ODE library contains a rich set of algorithms, mathematical subroutines and other computational tools dedicated for programmers, mathematicians, engineers or anyone interested in developing applications oriented to mathematics using C# programming language. In Fig. 3 we present software details regarding the C# implementation of the "SuspensionOptimization" class using properties and methods. We used the following variables and functions to solve the equations system expressed in (4).

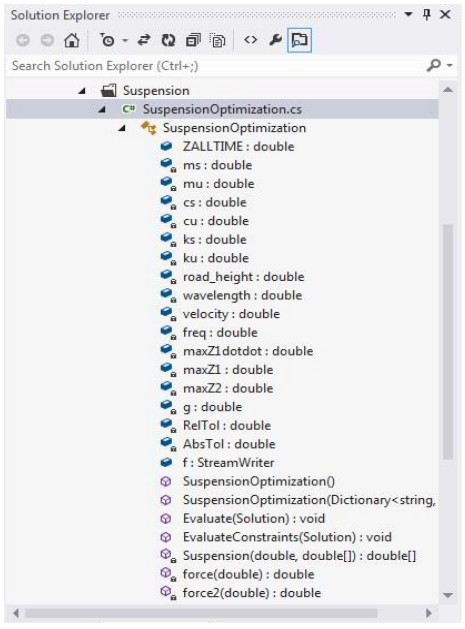


Fig. 3. C# software implementation: the properties and methods of the suspension system class.

TABLE I
MAPPING MODEL PARAMETERS TO CLASS PROPERTIES

Mathematical model parameter	"*SuspensionOptimization*" class property
m_1	mu
m_2	ms

k_1	ku
k_2	ks
c_1	cu
c_2	cs
h	double force(double t) An instance of the force() method at a certain moment during movement.
z'_0, z'_1, z'_2, z'_3	double[] ydot (see Fig. 4)
z_0, z_1, z_2, z_3	double[] y (see Fig. 4)

In Table I, there are exhibited the correspondences between suspensions' system parameters and the properties of "SuspensionOptimization" class.

Summarizing, Fig. 4 illustrates the Suspension() method which computes the derivative vector from mathematical model z'_0, z'_1, z'_2, z'_3, starting from (4) for each sample value from road.

```
private double[] Suspension(double t, double[] y) {
    double[] ydot = new double[4];
    ydot[0] = y[1];
    ydot[1] = (1 / ms) * (-ks * (y[0] - y[2]) - cs * (y[1] - y[3]));
    ydot[2] = y[3];
    ydot[3] = (1 / mu) * (ks * (y[0] - y[2]) + cs * (y[1] - y[3]) - ku * (y[2] - force(t)) - cu * y[3]);
    return ydot;
}
private double force(double t){
    return road_height * Math.Sin(freq * t);
}
```

Fig. 4. The source code of Suspension() method.

```
using DotNumerics.ODE;
//...
public override void Evaluate(Solution solution) {
//.....
    OdeFunction fun = new OdeFunction(Suspension);
    OdeExplicitRungeKutta45 RK45 = new OdeExplicitRungeKutta45(fun, 4);
    RK45.RelTol = this.RelTol;
    RK45.AbsTol = this.AbsTol;
    RK45.SetInitialValues(t0, y0); // Establish the initial values of functions in y0 array
    //Setting the sampling interval in t array
    y = RK45.Solve(y0, t);
    /* Computing the maximum bouncing acceleration maxZ1dotdot and the maximum
       displacement maxZ1 of sprung body on the sampling interval */
    maxZ1dotdot = -1;
    maxZ1 = -1;
    for (int i = 0; i < y.GetLength(0); i++){
        double xs = y[i, 1];
        double xsdot = y[i, 2];
        double xu = y[i, 3];
        double xudot = y[i, 4];
        double z1dotdot = (1 / ms) * (-ks * (xs - xu) - cs * (xsdot - xudot));
        maxZ1dotdot = Math.Max(maxZ1dotdot, Math.Abs(z1dotdot));
        maxZ1 = Math.Max(maxZ1, Math.Abs(xs));
    }
//....
    solution.Objectives[0] = maxZ1;
    solution.Objectives[1] = maxZ1dotdot;
}
```

Fig. 5. Using DotNumerics.ODE library for solving differential equations system with "Runge-Kutta" method.

C. Briefly about the Genetic Algorithm used in Optimization procedure

Genetic Algorithms (GA) are stochastic optimization algorithms based on Darwinian principles of natural selection and genetic inheritance [3], [12]. Their evolutionary goal is to improve a population of potential solutions, random initialized, for a design problem, after a number of generations or until a convergence threshold was reached. In each generation is applied on population, which is unit of evolution, a selection mechanism and variation operators such as crossover and mutation in order to explore and exploit the entire search space for the optimal solutions. The selection mechanism is based on the each solutions quality, computed as fitness function, and tries to avoid premature convergence or permanent loss of diversity within population.

In particular in our GA, we started with the solution representation. We chose two real numbers for the variables of suspension design – stiffness and damping coefficients (ks and cs). We initialized them with random values. These were passed into QCM, which is solved and is computed the dynamic response of the system (are found the values of sprung mass vertical displacement – objective 1, and sprung body acceleration – objective 2, meaning that our optimization problem is multi-objective one). These values are replaced back into the GA to compute the fitness of the suspension design. We had a population with 100 chromosomes [12] but the population size is parameterized. We used ranking selection method in order to choose the best individuals as parents for two new individuals. We applied arithmetic crossover and non-uniform mutation and obtained two new individuals. We applied elitist survival selection mechanism for keeping the best solutions from old population merged with the new obtained individuals. The objective function of the genetic algorithm is to minimize the sum between the maximum value of objective 2 and the average value of objective 1. This procedure is repeated until the stopping criterion is met.

IV. Simulation Results

After selecting the problem to solve (which embed in this version the road building module [1]) and setting the GA parameters (Fig. 6) the simulation begins. When finishes the next results are available: the sum between acceleration and vertical displacement, the optimal values for stiffness and damping coefficients and graphics with acceleration and displacement time varying (Fig. 7).

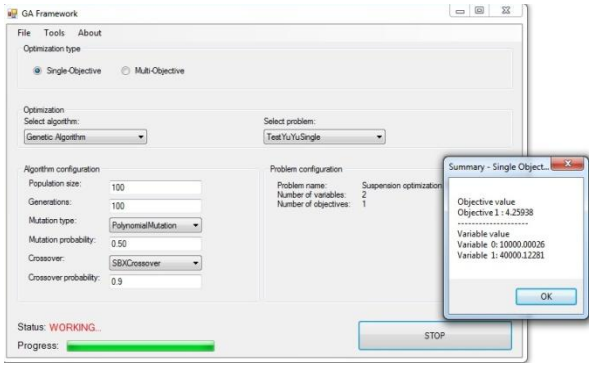


Fig. 6. The application GUI and synthetic results.

TABLE II
Comparison to Results on Different OPEL Car Models

Car features	OPEL car model			
	Corsa 10 generations	Corsa 100 generations	Astra I 10 generations	Astra I 100 generations
ku (N/m)	200000	200000	200000	200000
cu (N.s/m)	850	850	850	850

mu (kg)	14	14	19.5	19.5
ks (N/m)	40533.00	40002.00	43625.95	40000.12
cs (N.s/m)	10008.44	10000.00	10233.94	10000.00
ms (kg)	289	289	365	365
$\max(\ddot{z}_1)(\frac{m}{s^2})$	4.91813	4.91437	4.26569	4.22143
$mean(\ddot{z}_1)(\frac{m}{s^2})$	0.03680	0.03678	0.03806	0.03795

The Table II illustrates some simulation results obtained on two types of OPEL car models, run for different number of generations, keeping the mutation probability as 0.05 and crossover probability 0.90 and a population with 100 chromosomes.

Increasing the number of generations in the implemented Genetic Algorithm leads to higher design accuracy and reduces the maximum bouncing acceleration of the sprung mass. Also, due to the stochastic nature of genetic algorithm, the mutation parameter and the initial population could influence the obtained results.

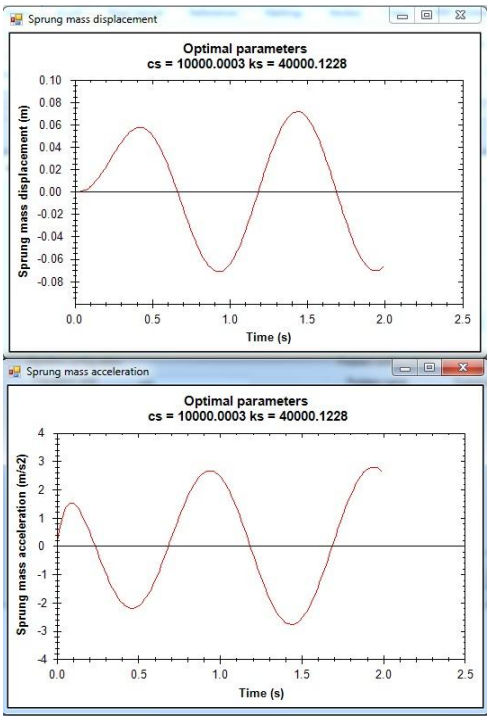


Fig. 7. Sprung body acceleration and vertical displacement on sinusoidal road followed by OPEL Astra I.

As it can be observed from Fig. 7, both the sprung mass acceleration, and the displacement take the wave form of the traveled road.

V. Conclusion and Further Work

In this paper we presented the mathematical model of suspensions system at OPEL cars. Helped by powerful software tools such as Microsoft Visual Studio 2012 C# and DotNumerics package we solved by numerical time integration with respect to time, the differential motion equations system, in order to optimize the 2DOF-QCM.

We implemented a genetic algorithm for finding the optimal values of coefficients of stiffness and damping of sprung body, in order to minimize the maximum bouncing acceleration of the sprung mass and minimize the average suspension displacement during movement on uneven roads. For simplicity, in this first stage we had a weighted sum approach for our multi-objective problem, converting it into the single-objective problem with equal weights for each objective, the new converted objective being necessary to be as small as possible.

For further work we are concerned to improve suspension's system by optimizing the stiffness and damping coefficients with the help of multi-objective optimization methods – Pareto (NSGA-II, SPEA2, etc), bio-inspired (SMPSO) or application of fuzzy logic technique to the control of automotive suspension system.

Acknowledgment

We thank to our student Sorin Gyorfi from Information Technology specialization for providing useful help in software implementation of algorithms that solve systems of differential equations by “Runge-Kutta” method.

References

[1] L. Roman, A. Florea, I.I. Cofaru, “*Software application for assessment the reliability of suspension system at OPEL cars and of road profiles*”, Annual Session of Scientific Papers - IMT Oradea 2014, submitted for publication.

[2] S. Badran, A. Salah, W. Abbas, O. Abouelatta, “Design of Optimal Linear Suspension for Quarter Car with Human Model using Genetic Algorithms”, *Research Bulletin of Jordan ACM*, Vol. 2, No. 8, April 2012, pp. 42-51.

[3] Yu, H., and Nan Yu., “*Application of Genetic Algorithms to Vehicle Suspension Design*”, Mechanical Engineering Department, The Pennsylvania State University, University Park, pp. 1-9, 2003.

[4] Santiago-Castillo J. A., *DotNumerics*, http://www.dotnumerics.com/NumericalLibraries/DifferentialEquations/Default.aspx, (accessed 02 April 2014).

[5] R.N. Jazar, *Vehicle Dynamics: Theory and Application*, Springer Science and Business Media, 2008.

[6] J.Y. Wong, *Theory of ground vehicles*, 4th ed., John Wiley & Sons, 2008.

[7] M. Agostinacchio, D. Ciampa, S. Olita, “The vibrations induced by surface irregularities in road pavements – a Matlab approach”, *European Transport Research Review*, Ed. Springer Berlin Heidelberg, December 2013.

[8] Rosculet M., *The mathematical analysis*, Didactical and Pedagogical Publishing Hall, Bucharest, 1979 (in Romanian).

[9] Stănășilă O., Brînzănescu V., *Special Mathematics. Theory, Examples, Applications*, All Publisher, Bucharest, 1994 (in Romanian).

[10] Jora B., Popeea C., Barbulea S., *Numerical methods in Automation. Linear Systems*, Enciclopedica Publisher, Bucharest, 1996 (in Romanian).

[11] Waldemar Dos Passos, *Numerical Methods, Algorithms and Tools in C#*, CRC Press, 2009.

[12] C.A. Coello Coello, D.A. van Veldhuizen, G.B. Lamont, *Evolutionary algorithms for solving multi-objective problems*, Springer, 2nd ed. 2007.