Mathematical Modelling and Optimization of a Vehicle Crash Test based on a Single-Mass

Andreas Klausen*, Sondre Sanden Tørdal*, Hamid Reza Karimi*, Kjell Gunnar Robbersmyr*, Mladen Jecmenica**, Ole Melteig**

*University of Agder,
Department of Engineering,
Faculty of Engineering and Science,
Jon Lilletuns vei 9, N-4898 Grimstad, Norway

**Telemark University College,
Department of Technology,
Institute for Process technology,
Kjølnes ring 56, N-3914 Porsgrunn, Norway

Abstract: In this paper mathematical modelling of a vehicle crash test based on a single mass is studied. The models under consideration consists of a single mass, a spring and/or a damper and are constructed according to the measured vehicle speed before the collision and measured vehicle accelerations in three directions at the centre of gravity. The objective of this paper is to compare different methods on establishing a simple model of a car crash and compare the results against real life crash data.

Key-Words: Kelvin-model, single-mass, optimization

1 Introduction

When car manufacturers design and build cars, they need to consider the safety of the occupant in the car. In order to test how a car will react to a crash, one could do a real life crash test. To do such a test requires good measurements, a locale, trained employees and a car. Instead of crashing a designed car many times to look for results, it's much more cost effective to crash one car and make a model that fits the crash. In order to make a model of the vehicle, there are generally two categories; Lumped Parameter Modeling (LPM) and Finitie Element Model (FEM) using CAD software to model the car. The first category LPM, uses up to several masses connected to each other with springs and dampers in between. The parameters of the springs and dampers are determined in order to match a vehicle crash. Jonsén et al. [1] used a lumped parameter model based on results from a crash test to identify the parameters to describe the crash. Another way to identify the parameters is to optimize a model versus the crash data using computational power. Kim et al. [2] used a lumped parameter model together with crash data to identify parameters using an optimization routine.

Pawlus et al. [3] presented a model with one mass, a spring and a damper to model the crash. This model is purely analytic, and the parameters can be calculated without using advanced programs or computational power. Pawlus et al. [4] presented a model with one mass and a spring. This model was used to explain the plastic deformation of the car in the crash. In order to do that, two models were used; one model for the crash at a time before maximum dynamic crush, and a different model for the crash after the time of maximum crush. The plasticity of the spring made sure the car wouldn't have a large rebound displacement, but rather oscillate around the maximum displacement. It was also shown that a Maxwell model could also prove to be a good model for the crash as it is suitable for material responses that exhibit relaxation and creep [5]. Finite element analysis, or FEA, is also widely used in crashworthiness research to make a model of a vehicle crash. Its upside compared to LPM are that FEM analysis gives much more detailed results about the deformation of the vehicle during the crash. However, making a model of the car and simulate the crash is a time consuming and computational heavy task. Zaouk et al. [6] made a finite element model of a pick-up truck and compared the model with multiple impact data set. They concluded that the simulation were consistent with the dataset, but some of the problems could be resolved by modeling more components in the cabin interior. A second reason to use FEM models is to look at the damage of the human body in a crash. Voo et al. [7] made a FE model of a human head to look for damage in the head during various load conditions.

Three models will be tested against each other to prove their accuracy compared to the crash test data. The aim of this paper is to investigate what kind of models shows promising results in a time interval and which models that shows good potential for further work.

2 Crash test data

The crash test data that are used in the models are collected from a calibration test done using a Ford Fiesta in a pole [8]. The car was equipped with an accelerometer in its center of gravity that measured the acceleration signals in x, y and z direction. The models in this paper uses the acceleration in the x-direction for computation. This acceleration is used to find the displacement and velocity of the car during the crash by using a forward-euler time integration algorithm. The parameters and the initial values for the crash are:

- m = 873 kg
- s(t=0) = 0cm
- v(t=0) = 35 $\frac{km}{h}$

The time $t = 0$ shows the time of impact, s is the displacement of the car and v is the velocity of the car. The results can be seen in Fig. 2.

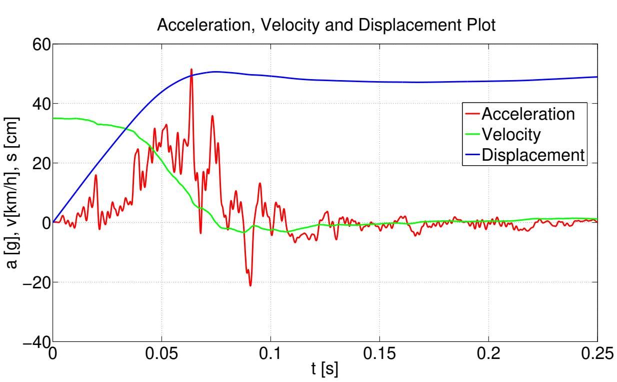

Figure 2: Crash test data

The red line shows the acceleration in x-direction in g (-9.81$\frac{m}{s^2}$), the blue line shows displacement in cm, and the green line shows velocity in km/h.

3 Mathematical Models

3.1 Spring-mass model

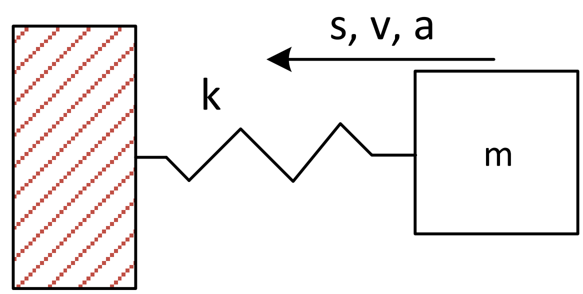

Figure 3: Spring-mass model

The model shown in Fig. 3 is based on the Kelvin model in [3] and [5]. In order to find the constant spring coefficient, some terms need to be defined and found in the crash test data:

t_m - Time of maximum dynamic crush, v(t_m) = 0.

C - Vehicle dynamic crush at time t_m, $s(t_m)$

t_c - The time at the geometric center of the crash pulse from time zero to t_m.

The time of dynamic crush is found in Fig. 2 where the velocity is 0. The centroid time can be found by either integrating the acceleration to find the area center, or use the following formula:

$$t_c = \frac{C}{v_0} = \frac{s(t_m)}{v(0)} \tag{1}$$

The normalized centroid time and angular position at dynamic crush is defined as [3]:

$$\tau_c = t_c\omega_e = \left(\frac{\alpha_m}{v_0}\right)\omega_e = e^{-\zeta\tau_m} \tag{2}$$

$$\tau_m = t_m\omega_e = \frac{1}{\sqrt{1-\zeta^2}}\arctan\frac{\sqrt{1-\zeta^2}}{\zeta} \tag{3}$$

where ω_e is the natural frequency of the system, α_m is the maximum dynamic crush and ζ is the damping factor. By combining Eqs. (2) and (3), the following relation is found:

$$\frac{\tau_c}{\tau_m} = \frac{t_c}{t_m} = \frac{\sqrt{1-\zeta^2}}{\arctan\frac{\sqrt{1-\zeta^2}}{\zeta}}e^{\left[\frac{-\zeta}{\sqrt{1-\zeta^2}}\arctan\frac{\sqrt{1-\zeta^2}}{\zeta}\right]} \tag{4}$$

By solving Eq (4) with respect to ζ, the damping of the system is determined. The eigenfrequency of the system (f_e) is found by using the time of maximum crush:

$$\tau_m = 2\pi f_e t_m \quad (5)$$

$$f_e = \frac{1}{2\pi t_m\sqrt{1-\zeta^2}}\arctan\frac{\sqrt{1-\zeta^2}}{\zeta} \quad (6)$$

If ζ is 0 or $\frac{t_c}{t_m} > \frac{\pi}{2}$, Huang [5] explains a method for undamped systems. The system is reduced to a mass-spring model, and a new time of maximum crush is calculated:

$$t_m = \frac{\pi t_c}{2} \quad (7)$$

The spring and damper constants (k and c respectively) can be found by using ζ and f_e in two second-order system equations respectively as follows:

$$k = 4\pi^2 f_e^2 m \quad (8)$$

$$c = 4\pi f_e \zeta m \quad (9)$$

Results

The crash test data shown in Fig. 2 is used in order to produce the results. The data extracted from the plot is:

$$t_m = 0.0749\text{s}$$

$$t_c = 0.0520\text{s}$$

$$\text{C} = 0.5063\text{m}$$

$$t_c/t_m = 1.5730$$

The relation between t_m and t_c gives a number larger than $\pi/2$ and therefore the system is reduced to a mass-spring system as shown in Fig. 3. The new time of dynamic crush is calculated:

$$t_m = \frac{t_c\pi}{2} = 0.818s \quad (10)$$

Since the damping ζ is zero, the eigenfrequency is calculated as:

$$f_e = \frac{1}{t_m 2\pi}\arctan(\infty) \quad (11)$$

$$f_e = \frac{1}{t_m 2\pi}\cdot\frac{\pi}{2} = 3.0544Hz \quad (12)$$

The spring coefficient can finally be determined by using Eq. (8):

$$k = 4\pi^2 f_e^2 m = 321.53\frac{kN}{m} \quad (13)$$

The results of a time integration using the spring constant determined in Eq. (13) and a comparison with the crash data can be seen in Fig. 4.

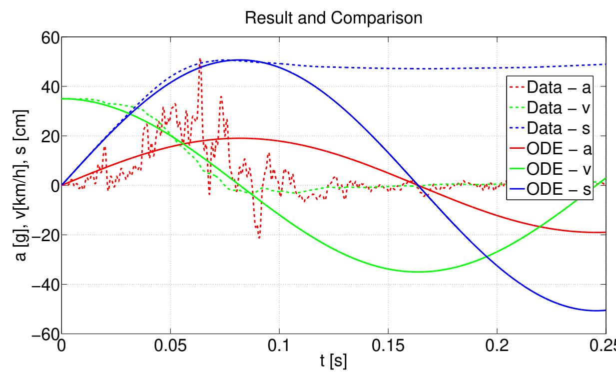

Figure 4: Results and comparison between Kelvin model and crash data

The crash data from Fig. 2 are shown as stapled lines in Fig. 4, while the results from the time integration is shown as continuous lines.

3.2 Elasto-plastic spring

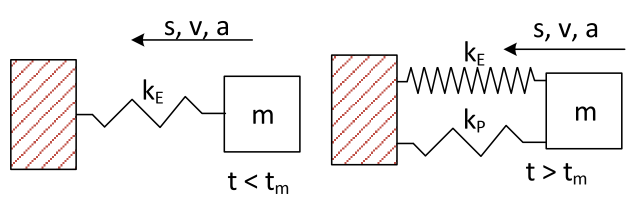

Figure 5: Elasto-plastic model

The model shown in Fig. 5 is based on an elasto-plastic model in [4]. k_E stands for elastic spring constant and k_P is used for the plastic deformation in the spring. In order to find the two spring constants that fits the crash data, some terms need to be defined and located in the crash data:

d_c - Maximum dynamic crush at $t = t_m$

v' - Maximum rebound velocity

Based on energy consideration, the following equations can be set up:

$$\Delta E = \frac{1}{2}k_E d_c^2 = \frac{1}{2}mv_o^2 \tag{14}$$
$$\Delta E' = \frac{1}{2}k_P d_e^2 = \frac{1}{2}mv'^2 \tag{15}$$

Where ΔE is the energy before the crash and $\Delta E'$ is the energy after the crash. Although the deflection energy should include the elastic spring coefficient k_E, it has been neglected because the plastic spring constant is much larger than the elastic spring constant. The maximum force on the car makes a relationship between the elastic and plastic spring coefficients:

$$F = k_E d_c = k_P d_e \tag{16}$$
$$\frac{d_e}{d_e} = \frac{k_L}{k_U} \tag{17}$$

COR (Coefficient Of Restitution) is defined as the percentage of the initial energy that is restored after the crash. This ratio can be found by:

$$COR^2 = \frac{\Delta E'}{\Delta E} = \left(\frac{v'}{v}\right)^2 \tag{18}$$
$$COR^2 = \frac{\Delta E'}{\Delta E} = \frac{k_P d_e^2}{k_E d_c^2} \tag{19}$$

The elastic spring coefficient can be found by using Eq. (14) and the plastic spring coefficient is determined by substitution Eq. (17) into Eq. (19):

$$k_E = m\frac{v_0^2}{d_c^2} \tag{20}$$
$$k_P = \frac{k_E}{COR^2} \tag{21}$$

Results

Using the crash data shown in Fig. 2, these data were extracted:

$$d_c = 0.5063m \tag{22}$$
$$v' = 0.9183\frac{m}{s} \tag{23}$$

The COR is calculated from equation (18):

$$COR = \left(\frac{v'}{v}\right) = \left(\frac{0.9183}{10}\right) = 0.0945 \tag{24}$$

Since the COR is known, the next step is to determine the different spring coefficients by using Eqs. (20) and (21) in the following:

$$k_E = 873 \cdot \frac{10^2}{0.5063^2} = 321.9\frac{kN}{m} \tag{25}$$
$$k_P = \frac{321.9}{0.0945^2} = 59458\frac{kN}{m} \tag{26}$$

Results and comparison between the crash data and the simulated crash can be seen in Fig. 6.

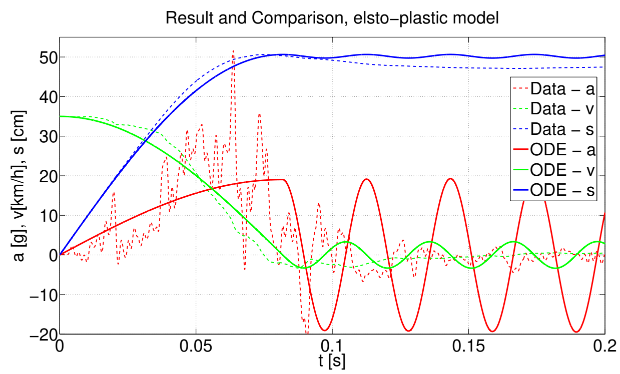

Figure 6: Results and comparison elasto-plastic model

The stapled lines are the crash data and the continuous lines are the simulated response.

3.3 Maxwell Spring-Damper model

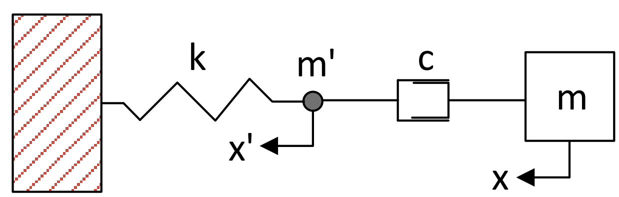

Figure 7: Maxwell Spring-Damper model

This model is based on the Maxwell Spring-Damper model shown by Huang in [5]. The mass is connected to the wall with a spring and a damper in series. This model is suitable for material responses that exhibit relaxation and creep, a time dependent phenomena. In vehicle impact modeling, it is suited for the localized impact where the vehicle effective stiffness is low [5]. The system can be described with the following differential equations:

$$m'\ddot{x}' = -x'k - (\dot{x}' - \dot{x}) \cdot c \tag{27}$$
$$m\ddot{x} = (\dot{x}' - \dot{x}) \cdot c \tag{28}$$

Where x and x′ are the displacement by the masses m and m′ respectively, see Fig. 7. By differentiating Eqs. (27) and (28) with respect to time and setting $m' = 0$, one has:

$$0 = -\dot{x}'k - (\ddot{x}' - \ddot{x}) \cdot c \tag{29}$$

$$m\frac{d\ddot{x}}{dt} = (\ddot{x}' - \ddot{x}) \cdot c \tag{30}$$

$$\tag{31}$$

From Eqs. (29) and (30), we obtain:

$$\dot{x}' = -\frac{m}{k} \cdot \frac{d\ddot{x}}{dt} \tag{32}$$

By inserting Eq. (32) into (28), a differential equation without the displacement is formed:

$$m\ddot{x} = \left(-\frac{m}{k} \cdot \frac{d\ddot{x}}{dt} - \dot{x}\right) \cdot c \tag{33}$$

$$\text{Or} \quad \frac{d\ddot{x}}{dt} = -\frac{k}{c}\ddot{x} - \frac{k}{m}\dot{x} \tag{34}$$

Solving this solution analytically gives the following characteristic equation:

$$r^3 + \frac{k}{c}r^2 + \frac{k}{m}r = 0 \tag{35}$$

Eq. (35) can be solved in order to find the roots of the equation. However, the coefficients k and c are unknown, and therefore one can't tell whether the roots contain imaginary numbers. Two solutions are therefore proposed; one for real roots and one for imaginary roots. For the real roots, the roots are defined as:

$$r_1 = 0$$
$$r_2 = a + b$$
$$r_3 = a - b$$

$$\text{Where} \quad a = -\frac{k}{2c} \tag{36}$$

$$b = \sqrt{\left(\frac{k}{2c}\right)^2 - \frac{k}{m}} \tag{37}$$

The equation for b clearly state that $\left(\frac{k}{2c}\right)^2 > \frac{k}{m}$ in order for the roots to be real. If the roots are imaginary, the following roots are defined:

$$r_1 = 0$$
$$r_2 = a + ib$$
$$r_3 = a - ib$$

$$\text{Where} \quad a = -\frac{k}{2c} \tag{38}$$

$$b = \sqrt{\frac{k}{m} - \left(\frac{k}{2c}\right)^2} \tag{39}$$

The solution for the displacement of the mass can be written as two equations, each depends on whether the roots are real or imaginary:
Real roots:
$x(t) = C_1 + C_2e^{(a+b)t} + C_3e^{(a-b)t}$
Imaginary roots:
$x(t) = C_1 + e^{at}\left[C_2Cos(bt) + C_3Sin(bt)\right]$

Now that the shapes of the different solutions are known, one can fit the equation to the displacement of the crash data to determine the unknown variables C_1, C_2, C_3, a and b. Once these variables are found, the spring and damper coefficients can be found by using the equations for a and b.

Results

The crash data were time integrated to find the displacement of the real crash. This displacement were used in a Curve Fit Toolbox in Matlab in order to find the unknown coefficients C_1, C_2, C_3, a and b. The Curve fit for the real roots can be seen in Fig. 8 and the curve fit for the imaginary roots in Fig. 9.

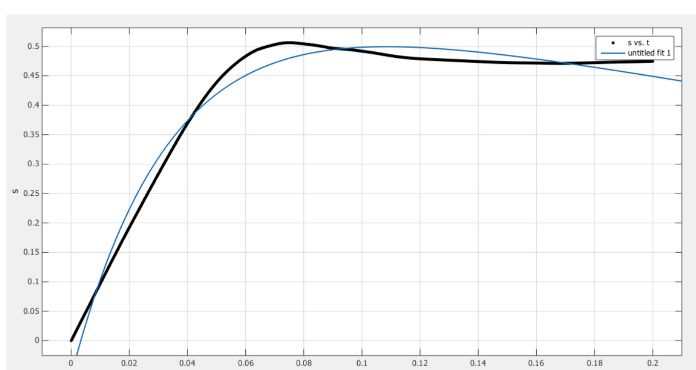

Figure 8: Curve fit with real roots

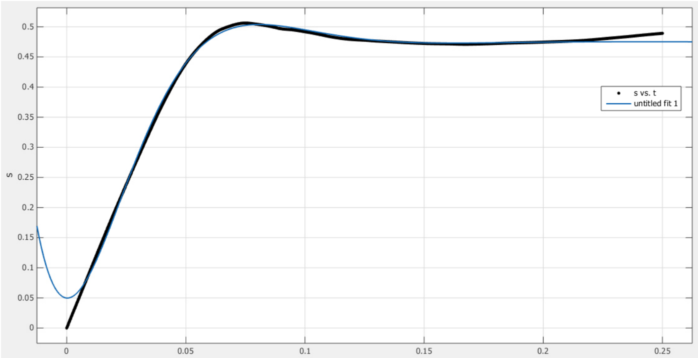

Figure 9: Curve fit with imaginary roots

Tab. 1 shows the results from the curve fitting and their corresponding damper and spring coefficients:

Table 1: Damper and spring constants for Maxwell Model.

	Real roots	Imaginary roots
a	-14.5	-33.42
b	-14.6	39.02
k	-2540 $\frac{N}{m}$	2304 $\frac{kN}{m}$
c	-87.6 $\frac{Ns}{m}$	34.47 $\frac{kNs}{m}$

Obviously, the values for k and c extracted from the real roots curve fit are wrong because they're negative. On the other hand, the damper and spring coefficient found using the curve for imaginary roots are valid. The result of the time integration using the spring and damper and a comparison to the crash data can be seen in Fig. 10.

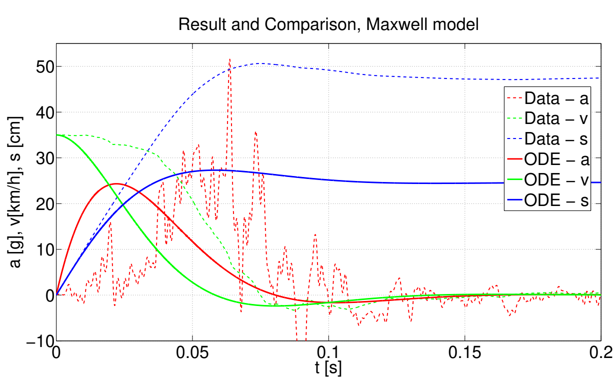

Figure 10: Results and comparison Maxwell Model

In Fig. 10, the crash data are shown as stapled lines, while the simulation is shown in continuous lines. The simulated displacement is not close to the real crash data, however, the shape of it matches the crash data quite well.

3.4 Non-linear spring and damper

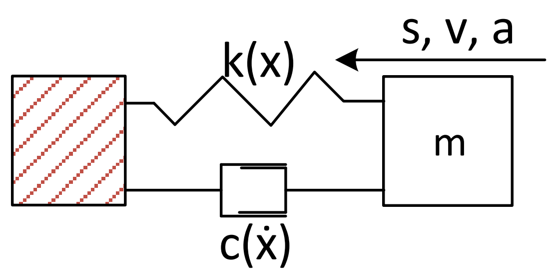

Figure 11: Non-linear spring and damper model

The model shown in Fig. 11 represents the car with a non-linear spring and damper that crashes into an obstacle. The purpose of using non-linear spring and damper coefficients is to make the simulated crash act non-linear based on the speed and position of the car. To justify the use of non-linear coefficients in the crash, Fig. 2 shows a velocity curve that matches a mass-spring system before the time of maximum crush. However, right after the crash, the system is damped critically to almost a tenth of the initial velocity. With a linear damper, the velocity of the simulated system would decrease rapidly because of the high initial velocity. But with a nonlinear damper, the shape of the damping coefficient can be controlled in order to dampen the car less during high velocities, and critically dampen it when the velocity is small.

One way to determine the non-linear spring and damper coefficients is to solve the non-linear ODE analytically. That could be difficult and time consuming if the system is large. Instead of solving it analytically, we can use computational power to optimize a design that fits the crash data best. Matlab is used to create an optimization routine based on a predefined shape of the nonlinear spring and damper coefficients, an objective and side constraints. Fig. 12 shows the predefined shape of the nonlinear spring and damper.

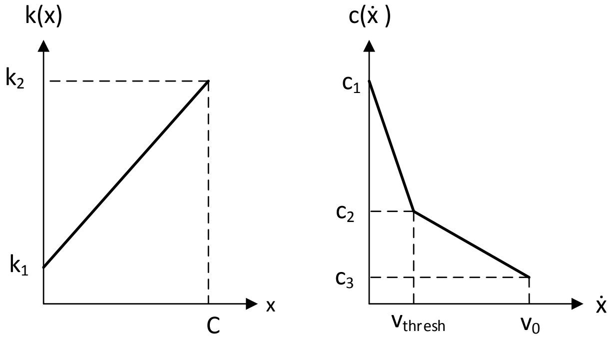

Figure 12: The predefined shape of the nonlinear spring and damper

The unknown variables k_1, k_2, c_1, c_2, c_3 and v_{thresh} are determined by the optimization routine to fit the crash data best. An optimization routine's intention is to minimize a given function by adjusting the unknown variables shown in Fig. 12. The function that is chosen to be minimized is the error between the displacement of the simulated crash and the crash data:

$$[\bar{\text{Error}}] = |\bar{x} - \bar{s}| \tag{40}$$

Where $\bar{x}$ is a vector containing displacement of the simulated data and $\bar{s}$ is a vector containing displacement of the crash data. Since the error is a vector, it cannot be minimized. To fix that, the norm of the error is minimized instead by multiplying the transposed of the error with the error:

$$|\bar{\text{Error}}|^2 = [\bar{\text{Error}}]^T \cdot [\bar{\text{Error}}] \tag{41}$$

This will yield a single scalar function to minimize. The last part of an optimization routine is to set side-constraints for the program to follow. Side constraints are split into two functions; functions to keep smaller or equal to zero, and functions to keep equal to zero. These side-contraints will help the program to keep the unknown variables within a threshold and to make the end result match satisfactory. The side constraints that are used are:

$$k_2 \geq k_1$$
$$c_1 \geq c_2$$
$$c_1 \geq c_3$$
$$c_2 \geq c_3$$
$$v_{thresh} \leq v_0$$
$$v_{thresh} \geq 0$$
$$x(t = t_m) = C$$
$$\dot{x}(t = t_m) = 0$$
$$k_1, k_2, c_1, c_2, c_3 \geq 0$$

Many of the side constraints are used to keep the predefined shape of the nonlinear spring and damper and to keep the damper and spring coefficients as positive values throughout the simulation. The side constraints for x and $\dot{x}$ are used to match the time and displacement at maximum dynamic crush in the crash data. When the objective function and the side constraints are determined, the program can start and the optimization routine will find the best value for the unknown variables that fits the crash test data and stays within the constraint.

Results

After setting initial values for the guess a couple of times, the program returned these values for the unknown parameters:

$$c_1 = 568.16[\frac{kNs}{m}]$$
$$c_2 = 39.96[\frac{kNs}{m}]$$
$$c_3 = 179.50[\frac{Ns}{m}]$$
$$v_{thresh} = 0.7948[\frac{m}{s}]$$
$$k_1 = 237.04[\frac{N}{m}]$$
$$k_2 = 253.44[\frac{kN}{m}]$$

By using these values within a time integration and comparing the results to the crash data, the following graph is made:

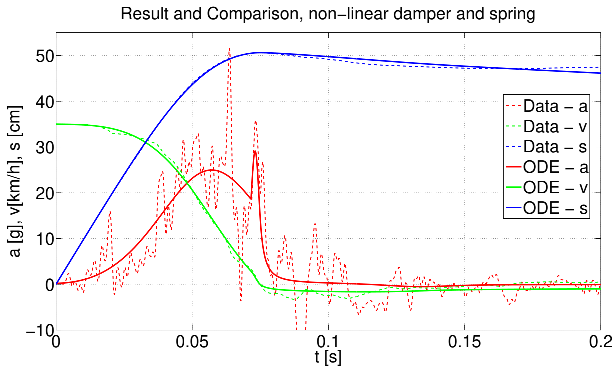

Figure 13: Results and comparison between crash data and simulated data

The stapled lines are the data from the crash test and the continuous lines are the results from the time simulation. As seen on Fig. 13, the velocity of the simulated model at the time of maximum dynamic crush is zero, which is satisfactory. The results as a whole matches the crash data quite good, but could also use some tweaking in order to fit perfectly.

4 Conclusion

The Kelvin model in Section 3.1 that consisted of a mass, a spring and a damper showed good results prior to the time of maximum crush. The reason for this is the model was reduced to a spring-mass system since the damping was calculated to be zero based on the crash test data. As there are no damping, the system will oscillate for infinite time with the maximum crush displacement as the amplitude. This infinite oscillation makes the time of interest in this model be between $t \in [0, t_m]$.

The elasto-platic model in Section 3.2 shows mixed but good results. The first part, the time before maximum crush, shows similarities to the Kelvin model. Both models are undamped and therefore the curves are similar. However, right after the time of maximum crush, the theory of a plastic damaged spring kicks in and makes the system oscillate around the maximum crush displacement. This gives the system a relatively stable displacement after maximum crush, but like the Kelvin model, the oscillation will never stop and therefore the time of interest should be somewhere around $t \in [0, 0.1s]$.

The Maxwell model in Section 3.3 showed great potential in its description that it would fit a physical car crash. The results however lacked good results. The damper kicked in too early and damped the response. This makes the time integrated displacement curve go halfway to the maximum crush. However when looking at the shape of the simulated displacement curve, one can see that it resembles the shape of the crash data quite well. The possible reasons for the bad results may come from inaccurate curve fitting or other faults. Either way, the Maxwell model shows potential for a crash test in its shape.

The Non-linear spring and damper model shown in Section 3.4 showed great results with small amounts of error when comparing the simulated data and the crash data. However, one problem with dealing with such optimization are large number of local minima that exists. By testing different initial conditions, one could find a global minima that gives the best result, but this is very time consuming if done manually. A more robust algorithm that searches a wider area of parameters is more suitable to this kind of optimization.

References

[1] P. Jonsén, E. Isaksson, K. G. Sundin, and M. Oldenburg. Identification of lumped parameter automotive crash models for bumper system development. *International Journal of Crashworthiness*, 14:6:533–541, 2009.

[2] C. H. Kim, A. R. Mijar, and J. S. Arora. Development of simplified models for design and optimization of automotive structures for crashworthiness. *Structural and Multidisciplinary Optimization*, 22-4:307–321, 2001.

[3] W. Pawlus, J. E. Nielsen, H. R. Karimi, and K. G. Robbersmyr. Mathematical modeling and analysis of a vehicle crash. *Proceedings of the 4th European Computing Conference*, pages 194–199.

[4] W. Pawlus, H. R. Karimi, and K. G. Robbersmyr. Mathematical modeling of a vehicle crash test based on elasto-plastic unloading scenarios of spring-mass models. 2010.

[5] M. Huang. *Vehicle Crash Mechanics*.

[6] A. K. Zaouk, N. E. Bedewi, C. Kan, and D. Marzougui. Validation of a non-linear finite element vehicle model using multiple imoact data. 1996.

[7] L. Voo, S. Kumaresan, F. A. Pintar, N. Yoganandan, and A. Sances. Finite-element models of the human head. *Medical and Biological Engineering and Computing*, 34-5:375–381, 1996.

[8] K. G. Robbersmyr. Calibration test of a standard ford fiesta 1.1l, model year 1987, according to ns-en 12767. (reference no. 04-02). 2004.