GERGELY GYEBRÓSZKI

Efficient numerical time-domain simulation of rail vehicles with coupled simulator systems

BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS

DEPARTMENT OF APPLIED MECHANICS

Gergely Gyebrószki

EFFICIENT NUMERICAL TIME-DOMAIN SIMULATION OF RAIL VEHICLES WITH COUPLED SIMULATOR SYSTEMS

Supervisors:

Dr. Zoltán Dombóvári

assistant teacher
DEPARTMENT OF APPLIED MECHANICS

Dr. Krisztián Kovács

developer engineer
KNORR-BREMSE RAIL VEHICLE SYSTEMS LTD.

Budapest, 2012.

Ide kell befűzni az eredeti, lepecsételt feladatkiírást

Feladatkiírás hátoldala

DECLARATION

I the undersigned, hereby declare that the Final Project submitted for assessment and defence exclusively contains the results of my own work assisted by my supervisors. Further to it, it is also stated that all other results taken from the technical literature or other sources are clearly identified and referred to according to copyright.

_____

Gergely Gyebrószki

TABLE OF CONTENTS

ABSTRACT

KNORR-BREMSE RAIL SYSTEMS HUNGARY uses various simulator systems for testing and developing brake systems for railway vehicles. There are purely virtual and Hardware-in-the-Loop (HiL) simulator systems. One common characteristic of these simulators is that during the simulation of the braking process, the inputs for the brake system (mainly the velocity and angular velocity of wheels) are calculated. To calculate these, horizontal and vertical oscillations of the vehicle and the dynamics of wheels have to be simulated in time domain. The main outputs of the simulation are the velocity and angular velocity of the wheels and the normal forces on the axles. Several on-train controllers are using these to calculate the amount of their intervention. (For example wheel-slip-protection valve changes the brake pressure, according to the actual state of the vehicle.)

Horizontal and vertical vehicle oscillations can be simulated adequately using a linear in-plane vehicle model. Linear mechanical models can be described clearly using mass-, stiffness- and damping matrices, and the differential equation system can be solved using the so called fundamental matrices. The theory of fundamental matrix solution is described in CHAPTER 2, and then the theory is applied to our mechanical model in CHAPTER 3.

The possibility of modelling nonlinear springs and dampers is described in CHAPTER 4. The literature of multirate partitioned methods is overviewed in CHAPTER 5.

The nonlinear dynamics of the wheel (including the modelling of friction at the rail-wheel contact) and the connection between the linear and non-linear systems are described in CHAPTER 6.

Finally a framework for the partitioned simulator is developed and described in CHAPTER 7 and then validated in CHAPTER 8.

The results of the simulation were analysed in CHAPTER 9.

TARTALMI ÖSSZEFOGLALÓ

A KNORR-BREMSE VASÚTI JÁRMŰRENDSZEREK KFT elektronikai fejlesztőcsoportjában többféle szimulátor rendszer használatos vasúti fékrendszerek teszteléséhez és fejlesztéséhez. Léteznek tisztán szoftveres és valós fékrendszeri elemeket is tartalmazó (Hardware-in-the-Loop, HiL) szimulátor rendszerek. Az összes szimulátor közös jellemzője, hogy a szimulált fékezési folyamat közben a fékrendszer bemeneti jellemzőit – elsősorban a keréksebesség időbeli alakulását – számítással kell meghatározni. Ehhez szükséges a jármű függőleges és hosszlengéseinek valamint a tengelyek forgási dinamikájának időtartományon végzett szimulációja. A szimuláció legfontosabb kimenetei a tengelyek haladó és forgó mozgásának sebessége valamint a keréktalpakon létrejövő dinamikus nyomóerők. Számos, a járművön található vezérlő ezeket a jellemzőket használja a beavatkozás mértékének meghatározásához. (Például a csúszásgátló szelep az aktuális sebességállapot alapján változtatja meg a féknyomást.)

A jármű függőleges és hosszlengéseinek szimulációja kielégítő pontossággal megoldható lineáris járműdinamikai modellekre alapozva, melyek egységes módon leírhatók tömeg-, merevségi- és csillapítási mátrixokkal, illetve a differenciál-egyenlet rendszer megoldható az ún. átviteli mátrix segítségével. Az átviteli mátrix-szal végzett megoldás elmélete a 2. FEJEZETben, az elmélet alkalmazása a mechanikai modellre pedig a 3. FEJEZETben található.

A nemlineáris rugó és csillapítóelemek modellezésének lehetőségét a 4. FEJEZETben ismertetem. A részekre bontott, különböző időskálájú szimulációs módszerek áttekintése pedig az 5. FEJEZETben található.

A nemlineáris kerékdinamikát (beleértve a kerék-sín kapcsolaton adódó súrlódást), valamint a lineáris és nemlineáris rendszerek kapcsolatát a 6. FEJEZETben ismertetem.

Végül a részekre bontott szimulációhoz szükséges keretrendszert mutatom be a 7. FEJEZETben, melyet a 8. FEJEZETben validálok.

A szimuláció eredményeit a 9. FEJEZETben értékelem.

1 INTRODUCTION

In this chapter a brief overview of the in-plane, linear model of the vehicle is presented, and the requirements of the simulation are described.

1.1 IN-PLANE LINEAR VEHICLE MODEL

The 3 dimensional train model is reduced to an in-plane model, by ignoring its extension along the direction perpendicular to the plane defined by the direction of movement and the *up* (y) axis. This *side view* model is adequate, because we are interested in the longitudinal (horizontal) and vertical oscillations of the vehicle.

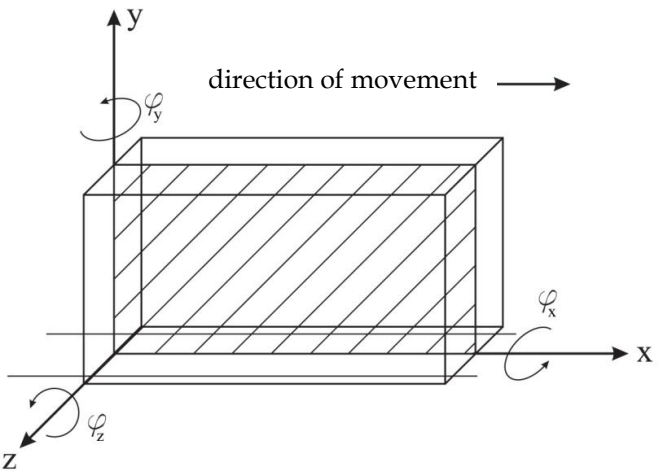


Fig. 1. The coordinate system of the train and the plane of modelling

In [1] an algorithm was developed to calculate the mass-, stiffness and damping matrices for a linear mechanical model assembled from the following elements:

- *body element*, with parameters: coordinate of center of mass (x, y), mass (m), inertia with respect to the z axis (θ), and degrees of freedom.
- *force element* between two nodes, with two parameters: stiffness (s) and damping (k)
- *node element* attached to a parent body, with parameters: location with respect to the center of mass of its parent body (x, y).

There are two other elements, which are used to handle external forces:

- *external force element*, which is used to introduce external force between two nodes. (The force acts on both parent bodies of the defined nodes – force and reaction)
- *displacement measuring element*, which is used to measure displacement between two nodes.

Any dynamic model consisting of the above elements can be written in a text editor and can be stored in a text file.

Using algorithms elaborated in [1], the program chooses the general coordinates for the linear system (as the displacements of / rotations around the center of mass of the bodies), and calculates the corresponding mass, stiffness and damping matrices. The equation of motion forms a system of differential equations, which can be written with the previously mentioned matrix coefficients:

$$\mathbf{M}\,\ddot{\mathbf{q}}(t) + \mathbf{K}\,\dot{\mathbf{q}}(t) + \mathbf{S}\,\mathbf{q}(t) = \mathbf{0}, \tag{1.1.1}$$

where **M** is the mass matrix, **K** is the damping matrix, **S** is the stiffness matrix and **q** is the vector of general coordinates. In case of external forces, the right hand side of the equation changes:

$$\mathbf{M}\,\ddot{\mathbf{q}}(t) + \mathbf{K}\,\dot{\mathbf{q}}(t) + \mathbf{S}\,\mathbf{q}(t) = \mathbf{Q}(t), \tag{1.1.2}$$

where **Q**(*t*) is the vector of general forces.

The vector of general forces can be calculated using the so called external force mapping matrix, which distributes the external forces (acting on the bodies defined by the external force elements), onto the general coordinates. If the total degree of freedom (DoF) of the system is *n*, and the number of external forces is *m*, then the vector of general forces:

$$\mathbf{Q}(t) = \mathbf{F}\,\mathbf{f}(t), \tag{1.1.3}$$

where **F** (*n* x *m*) is the external force mapping matrix (constant), and **f**(*t*) is the (*m* x 1) vector of external forces.

The displacement measuring elements can be used to generate displacement outputs for any given state of the system. If the total DoF of the system is *n*, and the number of displacement measuring elements is *l*, then:

$$\mathbf{d}(t) = \mathbf{P}\,\mathbf{q}(t), \tag{1.1.4}$$

where **P** (*l* x *n*) is the displacement mapping matrix (constant), and **d** is the (*l* x 1) vector of displacements corresponding to the defined displacement sensors.

The external force mapping and displacement mapping matrices are calculated by the program using the geometry of the model.

The in-plane, linear vehicle model is treated as the following constant matrix-coefficient system of differential equations:

$$\mathbf{M}\,\ddot{\mathbf{q}}(t) + \mathbf{K}\,\dot{\mathbf{q}}(t) + \mathbf{S}\,\mathbf{q}(t) = \mathbf{F}\,\mathbf{f}(t), \tag{1.1.5}$$

where $\mathbf{f}(t)$ vector of external forces are the input of simulation.

1.2 SIMULATION TASKS

The previously described linear vehicle model is needed to simulate the oscillations of the train. However we also have to simulate the nonlinear dynamics of wheels, including the rail-wheel contact.

Latter is done by simply solving the equation of motion of wheels for the rotation. (The x-directional movement of the wheels can be included in the linear model).

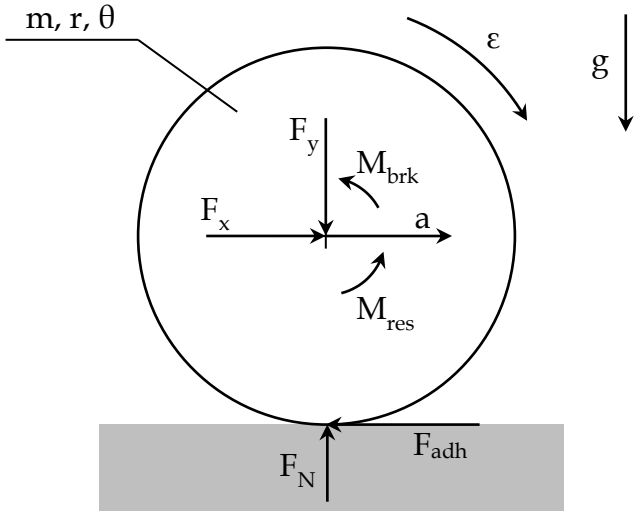


Fig. 2. Free body diagram of the wheel

The equation of motion is

$$\theta\,\dot{\omega} = \mathrm{r}\,\mathrm{F}_{adh} - \mathrm{M}_{res} - \mathrm{M}_{brk}, \tag{1.2.1}$$

where θ and r is the inertia and radius of the wheel, M_{res} is the internal resistance (friction) at the bearing of the axle, F_{adh} is the adhesion force at the wheel-rail contact and M_{brk} is the brake moment.

The brake moment can be expressed as:

$$\mathrm{M}_{brk} = \mathrm{r}_{brk}\,\mathrm{F}_{brk}, \tag{1.2.2}$$

and the adhesion force is:

$$\mathrm{F}_{adh} = \mathrm{F}_{\mathrm{N}}\,\mu(v, \omega), \tag{1.2.3}$$

where $\mu(v, \omega)$ function is the main source of nonlinearity.

Therefore the simulation of the whole vehicle can be done for a given time step as follows:

- Simulate the linear part of the vehicle (bogies, cars)
- Simulate the nonlinear part of the vehicle (wheel dynamics)
- Calculate contact forces between the wheels and other part of the vehicle.

The division of the vehicle and the proper coupling of the simulators are described in CHAPTER 6.

2 THEORY OF FUNDAMENTAL MATRIX SOLUTION

For any system of linear differential equations (with constant coefficients), there exists an operator called (special) fundamental matrix. This operator practically contains solution of the differential equation system, and therefore suitable to generate the solution of the system for a given initial condition and for a given time.

The goal of this chapter is to show the theory of fundamental matrices, in a similar way, than in [3].

2.1 FUNDAMENTAL MATRIX AND SPECIAL FUNDAMENTAL MATRIX

Consider the following (homogeneous) system of differential equations with constant coefficients (in matrix form)

$$\dot{\mathbf{y}} = \mathbf{A}\,\mathbf{y}, \tag{2.1.1}$$

where $\mathbf{A}$ is an (n x n) constant coefficient matrix.

Let us assume that the fundamental set of solutions for the system of differential equations is:

$$\mathbf{y}_1(t), \mathbf{y}_2(t), \mathbf{y}_3(t), \dots, \mathbf{y}_n(t). \tag{2.1.2}$$

The matrix:

$$\mathbf{\Psi}(t) = \begin{bmatrix} \mathrm{y}_{11}(t) & \cdots & \mathrm{y}_{\mathrm{n},1}(t) \\ \vdots & \ddots & \vdots \\ \mathrm{y}_{1,\mathrm{n}}(t) & \cdots & \mathrm{y}_{\mathrm{n},\mathrm{n}}(t) \end{bmatrix} \tag{2.1.3}$$

is called the fundamental matrix of the system [3], which is not singular because the columns of $\mathbf{\Psi}$ (solutions of the system) are linearly independent vectors.

The solution of (2.1.1) for a given initial value problem (IVP) is

$$\mathbf{y}(t) = \mathrm{c}_1\,\mathbf{y}_1(t) + \mathrm{c}_2\,\mathbf{y}_2(t) + \mathrm{c}_3\,\mathbf{y}_3(t) + \dots + \mathrm{c}_\mathrm{n}\,\mathbf{y}_\mathrm{n}(t), \tag{2.1.4}$$

which can be expressed using the fundamental matrix $\mathbf{\Psi}$:

$$\mathbf{y}(t) = \mathbf{\Psi}(t)\,\mathbf{c} = \begin{bmatrix} \mathrm{y}_{11}(t) & \cdots & \mathrm{y}_{\mathrm{n},1}(t) \\ \vdots & \ddots & \vdots \\ \mathrm{y}_{1,\mathrm{n}}(t) & \cdots & \mathrm{y}_{\mathrm{n},\mathrm{n}}(t) \end{bmatrix} \cdot \begin{bmatrix} \mathrm{c}_1 \\ \vdots \\ \mathrm{c}_n \end{bmatrix}. \tag{2.1.5}$$

Using the initial condition below

$$\mathbf{y}(t_0) = \mathbf{y}_0, \tag{2.1.6}$$

equation (2.1.5) at t_0 time is

$$\boldsymbol{\Psi}(t_0)\,\mathbf{c} = \mathbf{y}_0, \tag{2.1.7}$$

from which the vector of constants can be expressed as:

$$\mathbf{c} = \boldsymbol{\Psi}^{-1}(t_0)\,\mathbf{y}_0. \tag{2.1.8}$$

Substituting (2.1.8) back to (2.1.5) yields:

$$\mathbf{y}(t) = \underbrace{\boldsymbol{\Psi}(t)\,\boldsymbol{\Psi}^{-1}(t_0)}_{:=\boldsymbol{\Phi}}\,\mathbf{y}_0, \tag{2.1.9}$$

where

$$\boldsymbol{\Phi}(t) = \boldsymbol{\Psi}(t)\,\boldsymbol{\Psi}^{-1}(t_0) \tag{2.1.10}$$

is the special fundamental matrix of the system [3] representing the transformation between the initial condition and the solution for any given time, namely:

$$\mathbf{y}(t) = \boldsymbol{\Phi}(t)\,\mathbf{y}_0. \tag{2.1.11}$$

This operator is suitable to generate the solution of any constant coefficient homogeneous system of differential equations (instead of using numerical methods), however to construct the special fundamental matrix we actually have to know the solution of the system.

Once the system of differential equations is diagonalized (or in other words: de-coupled), the solution of the individual differential equations are known, therefore $\boldsymbol{\Phi}$ can be constructed analytically (see CHAPTER 2.3)

Another possibility is to calculate the special fundamental matrix as a matrix exponential (see [3] for proof)

$$\boldsymbol{\Phi}(t) = \mathrm{e}^{\mathbf{A}t}, \tag{2.1.12}$$

where the exponential can be calculated in several ways, for example with the following convergent series:

$$\mathrm{e}^{\mathbf{A}t} = \mathbf{I} + \sum_{n=1}^{\infty} \frac{\mathbf{A}^n t^n}{n!}. \tag{2.1.13}$$

However constructing $\boldsymbol{\Phi}$ as a matrix exponential introduces numerical errors.

2.2 TREATMENT OF INHOMOGENEITIES

Consider the following inhomogeneous system of differential equations, with constant coefficients (in matrix form)

$$\dot{\mathbf{y}} = \mathbf{A}\,\mathbf{y} + \mathbf{b}. \tag{2.2.1}$$

The general solution of (2.2.1) is

$$\mathbf{y}(t) = \boldsymbol{\Psi}(t)\,\mathbf{c} + \int^{t} \boldsymbol{\Psi}(t)\,\boldsymbol{\Psi}^{-1}(\tau)\,\mathbf{b}(\tau)\,\mathrm{d}\tau, \tag{2.2.2}$$

where the 2nd term represents the effect of inhomogeneities. See [3] and [4].

The solution for a specific initial value can be written as:

$$\mathbf{y}(t) = \boldsymbol{\Phi}(t)\,\mathbf{y}_0 + \int_{t_0}^{t} \boldsymbol{\Psi}(t)\,\boldsymbol{\Psi}^{-1}(\tau)\,\mathbf{b}(\tau)\,\mathrm{d}\tau. \tag{2.2.3}$$

Assuming, that we know the "kind" of inhomogeneity (**b**), the integral can be calculated analytically.

Let us assume that the inhomogeneity is linear with respect of time:

$$\mathbf{b}(\mathrm{t}) = \mathbf{b}_0 + \mathbf{b}_1\,\mathrm{t}. \tag{2.2.4}$$

In this case the (2.2.3) can be written as

$$\mathbf{y}(t) = \boldsymbol{\Phi}(t)\,\mathbf{y}_0 + \int_{t_0}^{t} \boldsymbol{\Psi}(t)\,\boldsymbol{\Psi}^{-1}(\tau)\,\mathbf{b}_0\,\mathrm{d}\tau + \int_{t_0}^{t} \boldsymbol{\Psi}(t)\,\boldsymbol{\Psi}^{-1}(\tau)\,\mathbf{b}_1\,\tau\,\mathrm{d}\tau, \tag{2.2.5}$$

after rearranging:

$$\mathbf{y}(t) = \boldsymbol{\Phi}(t)\,\mathbf{y}_0 + \boldsymbol{\Psi}(t)\,\mathbf{b}_0 \int_{t_0}^{t} \boldsymbol{\Psi}^{-1}(\tau)\,\mathrm{d}\tau + \boldsymbol{\Psi}(t)\,\mathbf{b}_1 \int_{t_0}^{t} \boldsymbol{\Psi}^{-1}(\tau)\,\tau\ \mathrm{d}\tau, \tag{2.2.6}$$

and introducing the following notations:

$$\boldsymbol{\Theta}_0(t) = \int_{t_0}^{t} \boldsymbol{\Psi}^{-1}(\tau)\,\mathrm{d}\tau, \tag{2.2.7}$$

$$\boldsymbol{\Theta}_1(t) = \int_{t_0}^{t} \boldsymbol{\Psi}^{-1}(\tau)\,\tau\ \mathrm{d}\tau,$$

the solution can be written as:

$$\mathbf{y}(t) = \boldsymbol{\Phi}(t)\,\mathbf{y}_0 + \boldsymbol{\Psi}(t)\,\boldsymbol{\Theta}_0(t)\,\mathbf{b}_0 + \boldsymbol{\Psi}(t)\,\boldsymbol{\Theta}_1(t)\,\mathbf{b}_1. \tag{2.2.8}$$

Again, these integrals can be calculated analytically only, if we know the solution of the (homogeneous) system, and can construct the fundamental matrix $\boldsymbol{\Psi}$.

The integral in the solution can be calculated for any inhomogeneity (linear, cubic, harmonic, etc.), and the final solution can always be constructed using the combination of analytically calculated integrals. (See [3] and [4].)

2.3 FUNDAMENTAL MATRIX OF DIAGONAL SYSTEMS

As we have seen in CHAPTER 2.1, to construct the (special) fundamental matrix we have to know the solution of the system of differential equations.

Consider the following (homogeneous) system of differential equations with constant coefficients (in matrix form)

$$\dot{\mathbf{y}} = \mathbf{D}\,\mathbf{y}, \tag{2.3.1}$$

where **D** is a diagonal matrix:

$$\mathbf{D} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}. \tag{2.3.2}$$

In this case the special fundamental matrix can be expressed analytically:

$$\mathbf{\Phi}^{\mathrm{D}}(t) = \mathrm{e}^{\mathbf{D}t} = \begin{bmatrix} \mathrm{e}^{\lambda_1 t} & 0 & \dots & 0 \\ 0 & \mathrm{e}^{\lambda_2 t} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathrm{e}^{\lambda_n t} \end{bmatrix}. \tag{2.3.3}$$

NOTE: Index D denotes that, the matrix belongs to the diagonal system.

Using the definition of the special fundamental matrix:

$$\mathbf{\Phi}^{\mathrm{D}}(t) = \mathbf{\Psi}^{\mathrm{D}}(t)\,\mathbf{\Psi}^{\mathrm{D}^{-1}}(t_0), \tag{2.3.4}$$

it can be seen that for $t_0 = 0$, the fundamental matrix is equal to the special fundamental matrix.

$$\mathbf{\Phi}^{\mathrm{D}}(t) = \mathbf{\Psi}^{\mathrm{D}}(t), \qquad t_0 = 0 \tag{2.3.5}$$

Because the fundamental matrices can be calculated analytically for diagonal systems, it is suitable to transform the non-diagonal system (2.2.1) into diagonal form. After calculating the fundamental matrices for the diagonal system, they can be transformed back to get the fundamental matrices of the original (non-diagonal) system.

The transformation of the system of differential equations can be done with its eigensystem, or when the coefficient matrix is singular, its Jordan decomposition. The application of diagonalization is described in CHAPTER 3.

2.4 SUMMARY

Opposite to the general approach in industry, where the simulation of constant coefficient differential equation systems is usually done by numerical methods (EULER's or RUNGE-KUTTA methods), it is suitable to simulate (or rather solve) the system using the analytically constructed fundamental matrices. In [2] the advantages of using the fundamental matrix-based simulation were shown.

3 APPLICATION OF FUNDAMENTAL MATRIX SOLUTION

In the previous chapter, the theory of fundamental matrices was briefly overviewed. In this chapter the theory is applied to our specific system of differential equations (1.1.5) and the calculation of the corresponding fundamental matrices is shown.

3.1 CAUCHY TRANSCRIPTION

In order to apply the theory described in CHAPTER 2, we need to transform the 2nd order system of differential equation to 1st order. The original 2nd order system:

$$\mathbf{M}\,\ddot{\mathbf{q}} + \mathbf{K}\,\dot{\mathbf{q}} + \mathbf{S}\,\mathbf{q} = \mathbf{F}\,\mathbf{f}, \tag{3.1.1}$$

from which, $\ddot{\mathbf{q}}$ can be expressed:

$$\ddot{\mathbf{q}} = -\mathbf{M}^{-1}\mathbf{K}\,\dot{\mathbf{q}} - \mathbf{M}^{-1}\mathbf{S}\,\mathbf{q} + \mathbf{M}^{-1}\mathbf{F}\,\mathbf{f}. \tag{3.1.2}$$

Introducing the following new state vector:

$$\mathbf{y}(t) = \begin{bmatrix}\dot{\boldsymbol{q}}(t)\\ \boldsymbol{q}(t)\end{bmatrix}. \tag{3.1.3}$$

Equation (3.1.2) can be written as

$$\dot{\mathbf{y}} = \begin{bmatrix}-\mathbf{M}^{-1}\mathbf{K} & -\mathbf{M}^{-1}\mathbf{S}\\ \mathbf{I} & \mathbf{0}\end{bmatrix}\cdot\mathbf{y} + \begin{bmatrix}\mathbf{M}^{-1}\mathbf{F}\\ \mathbf{0}\end{bmatrix}\cdot\mathbf{f}, \tag{3.1.4}$$

where $\mathbf{I}$ is an identity matrix and $\mathbf{0}$ is a matrix of zeros.

Introducing the following notations:

$$\mathbf{A} = \begin{bmatrix}-\mathbf{M}^{-1}\mathbf{K} & -\mathbf{M}^{-1}\mathbf{S}\\ \mathbf{I} & \mathbf{0}\end{bmatrix}, \tag{3.1.5}$$

$$\mathbf{B} = \begin{bmatrix}\mathbf{M}^{-1}\mathbf{F}\\ \mathbf{0}\end{bmatrix},$$

$$\mathbf{b} = \mathbf{B}\,\mathbf{f},$$

the original system of equation can be written compactly as:

$$\dot{\mathbf{y}} = \mathbf{A}\,\mathbf{y} + \mathbf{b}. \tag{3.1.6}$$

Equation (3.1.6) is a system of first order differential equations, on which the theory of fundamental matrix generation can be applied.

3.2 DIAGONALIZATION WITH EIGENSYSTEM

Consider the coefficient matrix of equation (3.1.6). Matrix **A** (n x n) represents a real mechanical system, and therefore its eigenvalues can be:

- complex conjugates (eigenvalue pairs) – in case of damped oscillations
- real non-zero eigenvalue pairs – in case of overdamping
- zero eigenvalue pairs – in case of rigid body like translations

Assume that **A** is non-singular (so the mechanical system does not have any direction which is free for rigid body like translation). In this case its matrix of eigenvalues:

$$\mathbf{D} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}, \tag{3.2.1}$$

and its matrix of eigenvectors:

$$\mathbf{T} = [\boldsymbol{\xi}_1 \quad \cdots \quad \boldsymbol{\xi}_n] = \begin{bmatrix} \xi_{11} & \cdots & \xi_{n,1} \\ \vdots & \ddots & \vdots \\ \xi_{1,n} & \cdots & \xi_{n,n} \end{bmatrix}, \tag{3.2.2}$$

are forming the eigen-decomposition (eigensystem) of matrix **A**, where $\lambda_1, \dots, \lambda_n$ are the eigenvalues and $\boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_n$ are the eigenvectors of **A**.

For the above matrices, the following equality is true:

$$\mathbf{A}\,\mathbf{T} = \mathbf{T}\,\mathbf{D}, \tag{3.2.3}$$

which can be rearranged to express the transformation between **A** and **D**:

$$\mathbf{T}^{-1}\,\mathbf{A}\,\mathbf{T} = \mathbf{D}. \tag{3.2.4}$$

Substitute the following into (3.1.6):

$$\mathbf{y} = \mathbf{T}\,\mathbf{u}, \tag{3.2.5}$$

$$\dot{\mathbf{y}} = \mathbf{T}\,\dot{\mathbf{u}}.$$

After substitution, equation

$$\mathbf{T}\,\dot{\mathbf{u}} = \mathbf{A}\,\mathbf{T}\,\mathbf{u} + \mathbf{b}, \tag{3.2.6}$$

multiplied by the inverse of **T** from the left yields:

$$\dot{\mathbf{u}} = \mathbf{D}\,\mathbf{u} + \mathbf{T}^{-1}\,\mathbf{b}. \tag{3.2.7}$$

Introduce the following notation for the inhomogeneity:

$$\mathbf{v} = \mathbf{T}^{-1}\,\mathbf{b} = \mathbf{T}^{-1}\,\mathbf{B}\,\mathbf{f}, \tag{3.2.8}$$

so the diagonalized system of differential equations can be written as:

$$\dot{\mathbf{u}} = \mathbf{D}\,\mathbf{u} + \mathbf{v}. \tag{3.2.9}$$

3.3 DIAGONALIZATION WITH JORDAN DECOMPOSITION

In the case of modelling vehicles, the assumption stated in CHAPTER 3.2 is not true, and the mechanical system has one or more direction in which it can move freely.

This fact results in a singular coefficient matrix **A**, which cannot be transformed to diagonal form using its eigensystem (A is not diagonalizable).

In this case the Jordan-decomposition of A can be used to generate a block diagonal matrix J and an invertible similarity matrix P such that:

$$\mathbf{P}^{-1}\,\mathbf{A}\,\mathbf{P} = \mathbf{J}. \tag{3.3.1}$$

For every translational direction, matrix **A** will have a pair of zero eigenvalues, thus if our system has one free direction (like in the case of most cars and trains), the eigenvalue matrix of **A** will have one 2 by 2 zero block.

$$\mathbf{D} = \begin{bmatrix} \lambda_1 & \dots & 0 & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \dots & \lambda_{n-2} & 0 & 0 \\ 0 & \dots & 0 & 0 & 0 \\ 0 & \dots & 0 & 0 & 0 \end{bmatrix}, \tag{3.3.2}$$

It can be seen that the only difference between the **J** (Jordan normal form of **A**) and **D** is that the Jordan normal form contains a one in the block of zero eigenvalues. The Jordan normal form of **A** is:

$$\mathbf{J} = \begin{bmatrix} \lambda_1 & \dots & 0 & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \dots & \lambda_{n-2} & 0 & 0 \\ 0 & \dots & 0 & 0 & 1 \\ 0 & \dots & 0 & 0 & 0 \end{bmatrix}, \tag{3.3.3}$$

Considering the difference between **J** and **D**, and equation (3.3.1), it is clear that the **P** is also very similar to **T**, the only difference is the last column.

$$\mathbf{P} = [\boldsymbol{\xi}_1 \quad \boldsymbol{\xi}_2 \quad \cdots \quad \boldsymbol{\xi}_{n-2} \quad \boldsymbol{\xi}_{\lambda=0} \quad \boldsymbol{\xi}'_{\lambda=0}], \tag{3.3.4}$$

where $\boldsymbol{\xi}'_{\lambda=0}$ is the first generalized eigenvector corresponding to the zero eigenvalue which can be computed using the ordinary eigenvector corresponding to the zero eigenvalue:

$$\mathbf{A}\,\boldsymbol{\xi}'_{\lambda=0} = \boldsymbol{\xi}_{\lambda=0}. \tag{3.3.5}$$

Using a similar substitution, as in the previous chapter:

$$\mathbf{y} = \mathbf{P}\,\mathbf{u}, \tag{3.3.6}$$

$$\dot{\mathbf{y}} = \mathbf{P}\,\dot{\mathbf{u}}.$$

The original equation (3.1.6) can be written as

$$\dot{\mathbf{u}} = \mathbf{J}\,\mathbf{u} + \mathbf{v}, \tag{3.3.7}$$

where

$$\mathbf{v} = \mathbf{P}^{-1}\,\mathbf{b} = \mathbf{P}^{-1}\,\mathbf{B}\,\mathbf{f}. \tag{3.3.8}$$

3.4 PURE REAL EIGEN-DECOMPOSITION

It is important to highlight, that the implemented software uses a linear algebra library (TEMPLATE NUMERICAL TOOLKIT, see [5]), which offers pure real eigen-decomposition. This means that the resulting eigenvalue and eigenvector matrices (**D** and **T**) will be real matrices even if the matrix has complex eigenvalues.

The resulting eigenvalue matrix of this decomposition is block diagonal, where the complex conjugate eigenvalue pairs are forming 2 by 2 blocks:

$$\widehat{\mathbf{D}} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & Re(\lambda_m) & Im(\lambda_m) & \dots & 0 \\ 0 & 0 & \dots & -Im(\lambda_m) & Re(\lambda_m) & \dots & 0 \\ \vdots & \vdots & & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & \lambda_n \end{bmatrix}. \tag{3.4.1}$$

NOTE: Hat (^) denotes the pure real decomposition.

The corresponding eigenvector matrix is generated to satisfy the following equality:

$$\mathbf{A}\,\widehat{\mathbf{T}} = \widehat{\mathbf{T}}\,\widehat{\mathbf{D}}, \tag{3.4.2}$$

3.5 FUNDAMENTAL MATRICES

The equations of the homogeneous diagonalized system can be solved; therefore the corresponding special fundamental matrix can be calculated analytically.

The usage of the pure-real eigen-decomposition (or the Jordan decomposition) results in a de-coupled system consisting of:

- First order differential equations:

$$\dot{u}_l = \lambda_l u_l, \tag{3.5.1}$$

- Systems of two first order differential equations:

$$\begin{bmatrix} \dot{u}_m \\ \dot{u}_{m+1} \end{bmatrix} = \begin{bmatrix} Re(\lambda_m) & Im(\lambda_m) \\ -Im(\lambda_m) & Re(\lambda_m) \end{bmatrix} \begin{bmatrix} u_m \\ u_{m+1} \end{bmatrix}, \tag{3.5.2}$$

- System of first order differential equations, at the Jordan block of zero eigenvalues:

$$\begin{bmatrix} \dot{u}_n \\ \dot{u}_{n+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_n \\ u_{n+1} \end{bmatrix}. \tag{3.5.3}$$

The general solution of these equations is known. The solution of (3.5.1) is:

$$\mathrm{u}_l(\mathrm{t}) = c_l\, \mathrm{e}^{\lambda_l\, t}. \tag{3.5.4}$$

Introduce the following notation for a 1 by 1 matrix block:

$$\mathrm{J}_l = \mathrm{e}^{\lambda_l\, t}. \tag{3.5.5}$$

The solution of (3.5.2) is:

$$\mathrm{u}_m(t) = c_m\, \mathrm{e}^{a\, t} \cos(b\, t) + c_{m+1}\, \mathrm{e}^{a\, t} \sin(b\, t), \tag{3.5.6}$$
$$\mathrm{u}_{m+1}(t) = -c_m\, \mathrm{e}^{a\, t} \sin(b\, t) + c_{m+1}\, \mathrm{e}^{a\, t} \cos(b\, t),$$

where

$$a = Re(\lambda_m), \tag{3.5.7}$$
$$b = Im(\lambda_m).$$

Again introduce the following notation for a 2 by 2 matrix block:

$$\mathbf{J}_m = \begin{bmatrix} \mathrm{e}^{a\, t} \cos(b\, t) & \mathrm{e}^{a\, t} \sin(b\, t) \\ -\mathrm{e}^{a\, t} \sin(b\, t) & \mathrm{e}^{a\, t} \cos(b\, t) \end{bmatrix}. \tag{3.5.8}$$

The solution of (3.5.3) is:

$$\mathrm{u}_n(t) = c_n + c_{n+1}\, \mathrm{t}, \tag{3.5.9}$$
$$\mathrm{u}_{n+1}(t) = c_{n+1}.$$

Lastly introduce the following notation for another 2 by 2 matrix block:

$$\mathbf{J}_n = \begin{bmatrix} 1 & \mathrm{t} \\ 0 & 1 \end{bmatrix}. \tag{3.5.10}$$

Now the solution of the de-coupled system can be written in a compact form using the previously defined matrix blocks:

$$\begin{bmatrix} \mathrm{u}_1(t) \\ \mathrm{u}_2(t) \\ \vdots \\ \mathrm{u}_m(t) \\ \mathrm{u}_{m+1}(t) \\ \vdots \\ \mathrm{u}_n(t) \\ \mathrm{u}_{n+1}(t) \end{bmatrix} = \begin{bmatrix} \mathrm{J}_1 & 0 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ 0 & \mathrm{J}_2 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{J}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{J_n} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \\ c_{m+m} \\ \vdots \\ c_n \\ c_{n+1} \end{bmatrix}, \tag{3.5.11}$$

where the block-diagonal supermatrix is the fundamental matrix of the diagonalized system.

$$\tilde{\boldsymbol{\Psi}}(t) = \begin{bmatrix} \mathrm{J}_1 & 0 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ 0 & \mathrm{J}_2 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{J}_m & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{J_n} \end{bmatrix} \tag{3.5.12}$$

NOTE: Tilde (~) denotes that the marked matrix belongs to the diagonalized system.

The special fundamental matrix of the diagonalized system (according to CHAPTER 2) is:

$$\tilde{\boldsymbol{\Phi}}(t) = \tilde{\boldsymbol{\Psi}}(t)\, \tilde{\boldsymbol{\Psi}}^{-1}(t_0). \tag{3.5.13}$$

Because of the block-diagonal nature of $\tilde{\boldsymbol{\Psi}}(t)$ its inverse is a matrix containing the inverse of the individual blocks:

$$\tilde{\boldsymbol{\Psi}}^{-1}(t) = \begin{bmatrix} \mathrm{J}_1^{-1} & 0 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ 0 & \mathrm{J}_l^{-1} & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{J}_k^{-1} & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{J_n^{-1}} \end{bmatrix}, \tag{3.5.14}$$

where:

$$\mathrm{J}_l^{-1}(t) = \mathrm{e}^{-\lambda_n t}, \tag{3.5.15}$$

$$\mathbf{J}_m^{-1}(t) = \begin{bmatrix} \mathrm{e}^{-a\,t}\cos(b\,t) & -\mathrm{e}^{-a\,t}\sin(b\,t) \\ \mathrm{e}^{-a\,t}\sin(b\,t) & \mathrm{e}^{-a\,t}\cos(b\,t) \end{bmatrix},$$

$$\mathbf{J}_\mathrm{n}^{-1} = \begin{bmatrix} 1 & -\mathrm{t} \\ 0 & 1 \end{bmatrix}.$$

It can be seen, that for $t_0 = 0$, $\tilde{\boldsymbol{\Psi}}^{-1}(t_0) = \mathbf{I}$, thus the special fundamental matrix of the system

$$\tilde{\boldsymbol{\Phi}}(t) = \tilde{\boldsymbol{\Psi}}(t)\, \tilde{\boldsymbol{\Psi}}^{-1}(0) = \tilde{\boldsymbol{\Psi}}(t), \tag{3.5.16}$$

is equal to the fundamental matrix of the system.

3.6 TREATMENT OF INHOMOGENEITY

Recall the diagonalized system of differential equations (3.2.9)

$$\dot{\mathbf{u}} = \mathbf{D}\,\mathbf{u} + \mathbf{v}. \tag{3.6.1}$$

According to CHAPTER 2.2 the solution of the inhomogeneous system is:

$$\mathbf{u}(t) = \tilde{\boldsymbol{\Phi}}(t)\,\mathbf{u}_0 + \int^t \tilde{\boldsymbol{\Psi}}(t)\,\tilde{\boldsymbol{\Psi}}^{-1}(\tau)\,\mathbf{v}(\tau)\,\mathrm{d}\tau. \tag{3.6.2}$$

Let us examine the second term of the solution.

The fundamental matrix can be pulled from the integration:

$$\widetilde{\mathbf{\Psi}}(t) \int^{t} \widetilde{\mathbf{\Psi}}^{-1}(\tau)\, \mathbf{v}(\tau)\, \mathrm{d}\tau. \tag{3.6.3}$$

The integral can be calculated if we know the function $\mathbf{v}(\tau)$.

Consider the case, when the inhomogeneity is linear with respect of time.

$$\mathbf{v}(\mathrm{t}) = \mathbf{v}_0 + \mathbf{v}_1\, \mathrm{t}. \tag{3.6.4}$$

In this case, the integral splits into two terms:

$$\int^{t} \widetilde{\mathbf{\Psi}}^{-1}(\tau)\, \mathbf{v}(\tau)\, \mathrm{d}\tau = \mathbf{v}_0 \int^{t} \widetilde{\mathbf{\Psi}}^{-1}(\tau)\, \mathrm{d}\tau + \mathbf{v}_1 \int^{t} \widetilde{\mathbf{\Psi}}^{-1}(\tau)\, \tau\, \mathrm{d}\tau. \tag{3.6.5}$$

Examine the first term, and introduce the following notation:

$$\widetilde{\mathbf{\Theta}}_0(t) = \int^{t} \widetilde{\mathbf{\Psi}}^{-1}(\tau)\, \mathrm{d}\tau. \tag{3.6.6}$$

Since $\widetilde{\mathbf{\Psi}}^{-1}$ is block diagonal, it can be integrated by its blocks,

$$\widetilde{\mathbf{\Theta}}_0(t) = \begin{bmatrix} \mathrm{K}_{0,1} & 0 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ 0 & \mathrm{K}_{0,l} & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{K}_{0,m} & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{K}_{0,n} \end{bmatrix}, \tag{3.6.7}$$

where:

$$\mathrm{K}_{0,l}(t) = \int^{t} \mathrm{J}_l^{-1}(\tau)\, \mathrm{d}\tau = -\frac{1}{\lambda_n} \mathrm{e}^{-\lambda_n t}, \tag{3.6.8}$$

$$\mathbf{K}_{0,m}(t) = \int^{t} \mathbf{J}_m^{-1}(\tau)\, \mathrm{d}\tau =$$

$$= \frac{\mathrm{e}^{-at}}{a^2 + b^2} \begin{bmatrix} -a\cos(b\,t) + b\sin(b\,t) & a\sin(b\,t) + b\cos(b\,t) \\ -a\sin(b\,t) - b\cos(b\,t) & -a\cos(b\,t) + b\sin(b\,t) \end{bmatrix},$$

$$\mathbf{K}_{0,n}(t) = \int^{t} \mathbf{J}_{\mathrm{n}}^{-1}(\tau)\, \mathrm{d}\tau = \begin{bmatrix} \mathrm{t} & -\frac{\mathrm{t}^2}{2} \\ 0 & \mathrm{t} \end{bmatrix}.$$

Introduce the following notation for the second term of the solution:

$$\widetilde{\mathbf{\Theta}}_1(t) = \int^{t} \widetilde{\mathbf{\Psi}}^{-1}(\tau)\, \tau\, \mathrm{d}\tau. \tag{3.6.9}$$

The integral blocks of the second term can be calculated:

$$\widetilde{\mathbf{\Theta}}_1(t) = \begin{bmatrix} \mathrm{K}_{1,1} & 0 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ 0 & \mathrm{K}_{1,l} & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{K}_{1,m} & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{K}_{1,n} \end{bmatrix}, \tag{3.6.10}$$

similarly as in the previous case:

$$\mathrm{K}_{1,l}(t) = \int^t \mathrm{J}_l^{-1}(\tau)\, \tau \, \mathrm{d}\tau = -\frac{1}{\lambda_n^2} \mathrm{e}^{-\lambda_n t}(1 + \lambda_n\, t), \tag{3.6.11}$$

$$\mathbf{K}_{1,m}(t) = \int^t \mathbf{J}_m^{-1}(\tau)\, \tau \, \mathrm{d}\tau =$$

$$= \frac{\mathrm{e}^{-at}}{(a^2+b^2)^2} \begin{bmatrix} \mathrm{k}_{m1} \sin(b\, t) - \mathrm{k}_{m2} \cos(b\, t) & \mathrm{k}_{m2} \sin(b\, t) + \mathrm{k}_{m1} \cos(b\, t) \\ -\mathrm{k}_{m2} \sin(b\, t) - \mathrm{k}_{m1} \cos(b\, t) & \mathrm{k}_{m1} \sin(b\, t) - \mathrm{k}_{m2} \cos(b\, t) \end{bmatrix},$$

$$\mathrm{k}_{m1} = b(t(a^2+b^2) + 2a),$$

$$\mathrm{k}_{m2} = (a\, t(a^2+b^2) + a^2 - b^2),$$

$$\mathbf{K}_{1,n}(t) = \int^t \mathbf{J}_{\mathrm{n}}^{-1}(\tau)\, \tau \, \mathrm{d}\tau = \begin{bmatrix} \frac{\mathrm{t}^2}{2} & -\frac{\mathrm{t}^3}{3} \\ 0 & \frac{\mathrm{t}^2}{2} \end{bmatrix}.$$

Now the solution of (3.6.1) can be written as:

$$\mathbf{u}(t) = \widetilde{\mathbf{\Phi}}(t)\, \mathbf{u}_0 + \widetilde{\mathbf{\Psi}}(t)\big(\, \widetilde{\mathbf{\Theta}}_0(t)\, \mathbf{v}_0 + \widetilde{\mathbf{\Theta}}_1(t)\, \mathbf{v}_1\big). \tag{3.6.12}$$

Introducing the following notations:

$$\widetilde{\mathbf{\Omega}}_0(t) = \widetilde{\mathbf{\Psi}}(t)\, \widetilde{\mathbf{\Theta}}_0(t), \tag{3.6.13}$$

$$\widetilde{\mathbf{\Omega}}_1(t) = \widetilde{\mathbf{\Psi}}(t)\, \widetilde{\mathbf{\Theta}}_1(t),$$

which can be called as solution operators for the inhomogeneity, the solution can be written more compactly as:

$$\mathbf{u}(t) = \widetilde{\mathbf{\Phi}}(t)\, \mathbf{u}_0 + \widetilde{\mathbf{\Omega}}_0(t)\, \mathbf{v}_0 + \widetilde{\mathbf{\Omega}}_1(t)\, \mathbf{v}_1. \tag{3.6.14}$$

3.7 FORMULATING A NUMERICAL METHOD

The last step of converting the theory into an applicable solution method is to calculate the fundamental matrices for a definite time step (or a set of time steps).

A numerical method can be formulated if we consider a solution between $t_0 = 0$ and $t_1 = \Delta t$, within the inhomogeneity is defined by the previously described linear function.

$$\mathbf{u}_1 = \widetilde{\mathbf{\Phi}}(\Delta \mathrm{t})\, \mathbf{u}_0 + \widetilde{\mathbf{\Omega}}_0(\Delta \mathrm{t})\, \mathbf{v}_{0,0} + \widetilde{\mathbf{\Omega}}_1(\Delta \mathrm{t})\, \mathbf{v}_{1,0}, \tag{3.7.1}$$

which can be generalized as:

$$\mathbf{u}_{n+1} = \widetilde{\mathbf{\Phi}}(\Delta t)\,\mathbf{u}_n + \widetilde{\mathbf{\Omega}}_0(\Delta t)\,\mathbf{v}_{0,n} + \widetilde{\mathbf{\Omega}}_1(\Delta t)\,\mathbf{v}_{1,n}, \tag{3.7.2}$$

where $\mathbf{v}_{0,n}$ and $\mathbf{v}_{1,n}$ are parameters for the (piecewise linear) inhomogeneity in the nth time slice.

The fundamental and special fundamental matrices can be simply calculated for the fixed time step by substituting the definite value of Δt:

$$\widetilde{\mathbf{\Psi}}(\Delta t) = \widetilde{\mathbf{\Phi}}(\Delta t) = \begin{bmatrix} J_1(\Delta t) & 0 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ 0 & J_2(\Delta t) & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{J}_m(\Delta t) & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{J_n}(\Delta t) \end{bmatrix}. \tag{3.7.3}$$

The solution operators can be calculated for the definite time step, by calculating the definite integral:

$$\widetilde{\mathbf{\Omega}}_0(\Delta t) = \widetilde{\mathbf{\Psi}}(\Delta t)\,\widetilde{\mathbf{\Theta}}_0(\Delta t) = \widetilde{\mathbf{\Psi}}(\Delta t)\int_0^{\Delta t} \widetilde{\mathbf{\Psi}}^{-1}(\tau)\,\mathrm{d}\tau, \tag{3.7.4}$$

$$\widetilde{\mathbf{\Omega}}_1(\Delta t) = \widetilde{\mathbf{\Psi}}(\Delta t)\,\widetilde{\mathbf{\Theta}}_1(\Delta t) = \widetilde{\mathbf{\Psi}}(\Delta t)\int_0^{\Delta t} \widetilde{\mathbf{\Psi}}^{-1}(\tau)\,\tau\,\mathrm{d}\tau.$$

This can be done, using the previously introduced block notation

$$\widetilde{\mathbf{\Theta}}_0(\Delta t) = \tag{3.7.5}$$

$$= \begin{bmatrix} \ddots & 0 & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ 0 & K_{0,l}(\Delta t) - K_{0,l}(0) & \cdots & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{K}_{0,m}(\Delta t) - \mathbf{K}_{0,m}(0) & \cdots & \mathbf{0} \\ \vdots & \vdots & & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \cdots & \mathbf{K}_{0,n}(\Delta t) - \mathbf{K}_{0,n}(0) \end{bmatrix},$$

and similarly for $\widetilde{\mathbf{\Theta}}_1(\Delta t)$.

This way a numerical method (3.7.2) can be constructed for a set of time steps, which is based on the analytically calculated solution of the diagonalized system.

3.8 FUNDAMENTAL MATRIX FOR THE 1ST ORDER SYSTEM

To obtain the fundamental matrices of the 1st order system (3.1.6), we can transform the fundamental matrices of the diagonalized system back to our original coordinate system. (NOTE: in this chapter **T** will denote the transformation matrix used for diagonalization – either the matrix of eigenvectors or the similarity matrix of the Jordan normal form)

The inverse of coordinate transformation (3.2.5) is

$$\mathbf{u} = \mathbf{T}^{-1}\,\mathbf{y}, \tag{3.8.1}$$

which can be substituted into equation (3.6.14):

$$\mathbf{T}^{-1}\,\mathbf{y}(t) = \widetilde{\mathbf{\Phi}}(t)\,\mathbf{T}^{-1}\,\mathbf{y}_0 + \widetilde{\mathbf{\Omega}}_0(t)\,\mathbf{v}_0 + \widetilde{\mathbf{\Omega}}_1(t)\,\mathbf{v}_1. \tag{3.8.2}$$

After multiplying the equation with $\mathbf{T}$ from left,

$$\mathbf{y}(t) = \mathbf{T}\,\widetilde{\mathbf{\Phi}}(t)\,\mathbf{T}^{-1}\,\mathbf{y}_0 + \mathbf{T}\,\widetilde{\mathbf{\Omega}}_0(t)\,\mathbf{v}_0 + \mathbf{T}\,\widetilde{\mathbf{\Omega}}_1(t)\,\mathbf{v}_1. \tag{3.8.3}$$

the special fundamental matrix of the 1st order system can be recognized:

$$\mathbf{\Phi}(t) = \mathbf{T}\,\widetilde{\mathbf{\Phi}}(t)\,\mathbf{T}^{-1}. \tag{3.8.4}$$

NOTE: Tilde (~) denotes that the marked matrix belongs to the diagonalized system.

Since the inhomogeneity also contains the transformation matrix $\mathbf{T}$

$$\mathbf{v}(\mathrm{t}) = \mathbf{v}_0 + \mathbf{v}_1\,\mathrm{t}, \tag{3.8.5}$$

$$\mathbf{v} = \mathbf{T}^{-1}\,\mathbf{b} = \mathbf{T}^{-1}\,\mathbf{B}\,\mathbf{f},$$

the solution operators for the inhomogeneity, of the 1st order system can be written as:

$$\mathbf{\Omega}_0(t) = \mathbf{T}\,\widetilde{\mathbf{\Omega}}_0(t)\,\mathbf{T}^{-1}, \tag{3.8.6}$$

$$\mathbf{\Omega}_1(t) = \mathbf{T}\,\widetilde{\mathbf{\Omega}}_1(t)\,\mathbf{T}^{-1}.$$

Using the previously recognized matrices, the solution of the 1st order system (3.1.6) can be written compactly as:

$$\mathbf{y}(t) = \mathbf{\Phi}(t)\,\mathbf{y}_0 + \mathbf{\Omega}_0(t)\,\mathbf{b}_0 + \mathbf{\Omega}_1(t)\,\mathbf{b}_1, \tag{3.8.7}$$

where the inhomogeneity is already mapped to the state vector:

$$\mathbf{b}(\mathrm{t}) = \mathbf{b}_0 + \mathbf{b}_1\,\mathrm{t}, \tag{3.8.8}$$

$$\mathbf{b} = \mathbf{B}\,\mathbf{f}.$$

NOTE: See (3.1.5)

Solution (3.8.7) can be also generalized for a given time step, using the generalized operators of the diagonalized system. The resulting equation will be:

$$\mathbf{y}_{n+1} = \mathbf{\Phi}(\Delta\mathrm{t})\,\mathbf{y}_n + \mathbf{\Omega}_0(\Delta\mathrm{t})\,\mathbf{b}_{0,n} + \mathbf{\Omega}_1(\Delta\mathrm{t})\,\mathbf{b}_{1,n}. \tag{3.8.9}$$

3.9 SUMMARY

In this chapter, our system of differential equations was diagonalized then the calculation of the fundamental matrices and solution operators for the inhomogeneity was presented. These operators then transformed back to the 1st order system, finally a numerical method is formulated.

It is important to highlight that all the calculations (except the diagonalization) are purely analytical, therefore no numerical errors are introduced in the calculation.

The only operation which introduces numerical errors in the calculation of fundamental matrices is the decomposition which is based on an eigenvalue-eigenvector calculation.

4 MODELLING DYNAMIC SYSTEMS WITH NONLINEAR SPRINGS AND OR DAMPERS

This chapter contains a brief overview of the literature of modelling dynamic systems with nonlinear springs or dampers and summarizes the ways in which the current simulator can handle nonlinear elements.

4.1 BRIEF OVERVIEW OF THE LITERATURE

Real mechanical models often contain springs and dampers with nonlinear characteristics and sometimes modelling these components by linearizing them is not a good decision. Modelling nonlinear components can be usually done in two ways:

- by linearizing the system and creating a (piecewise) linear model (described in different approaches in [6],[7] and [8]) , or
- by keeping the nonlinear mechanical model. (as presented in [9] and [10])

In [6] two families of linearization methods for solving both autonomous and non-autonomous ordinary differential equations are introduced.

The first family of linearization techniques keeps the time (or independent variable) continuous and the right-hand side of the ordinary differential equations is approximated by the first two term of its Taylor series expansion (in a piecewise linear fashion). As a consequence, a system of linear ordinary differential equations is obtained. This system can be solved analytically in each time interval.

The second family, the family of θ-techniques are used for the time discretization, and the resulting nonlinear system of algebraic equations are linearized with respect to the previous time level, and a linear system of algebraic equations can be obtained. (This family of finite difference schemes are referred to as linearized θ-methods in [6], and are linearly implicit techniques)

In [8] it is shown, that the most basic approach is to linearize the system about an operating point and use standard linear estimation techniques (the first derivative of the nonlinear function evaluated at a specific operating point is used to develop a first order set of linear differential equations). [8] also shows a technique, which is using piecewise linear models, that cover the expected range of state variables, therefore the technique does not limit the behaviour of the system in case of larger displacements. The only restriction is that the nonlinearities must be able to be

approximated as piecewise linear functions. A simple example of the piecewise linear modelling of a pendulum is also presented.

The other main approach of modelling nonlinear components in a mechanical system is to elaborate a detailed nonlinear model, either by using finite element method like techniques as in [9] or by simply solving the system of nonlinear differential equations as presented in [10].

In [9] it is shown, that nonlinear springs are usually modelled based on detailed finite element models. While these models are accurate, the number of nodal degrees of freedom can be too large, making the model impractical for use in multibody vehicle system applications. Because these models do not allow reducing the number of degrees of freedom by using component mode synthesis techniques, [9] introduces a new modelling strategy; a nonlinear finite element floating frame of reference formulation, that allows the use of modal reduction techniques. (The leaves of the springs are discretized using the finite element method, and after assembling the mass and stiffness matrices of the finite elements, the nonlinear finite element formulation allows the use of component mode synthesis methods)

[10] describes the importance of air springs in rail-vehicle dynamics (as the key component, which guarantees good ride comfort for the passengers), and presents a detailed nonlinear thermo-dynamical air spring model. The model consists of a set of springs, dampers and a mass (see the Figure below).

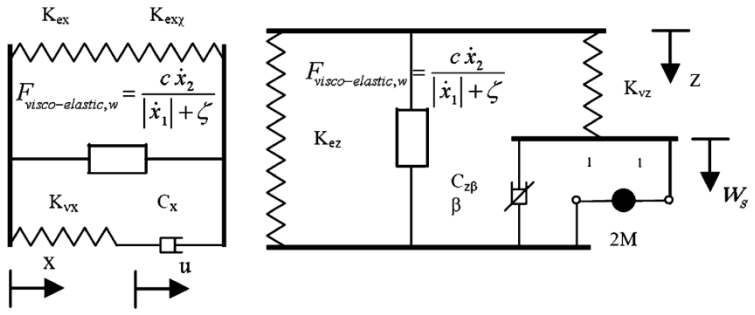


Fig. 3. Lateral and vertical model of an air spring [10]

The air spring model is then validated by comparing simulation results of 42 DoF train model with measured data.

4.2 HANDLING NONLINEAR SPRINGS AND DAMPERS

The currently implemented simulator system can handle nonlinear springs and dampers in two different ways.

- by using linear *force* elements and feeding back the difference of the spring / damper force using *external force* and *displacement measuring* elements.
- by creating a piecewise linear system (creating fundamental matrices and solution operators for different linear spring / damper characteristics).

Consider the following example of a progressive spring with the following spring force characteristic:

$$\mathrm{F}_{spring} = \mathrm{s}\,\mathrm{x}^2. \tag{4.2.1}$$

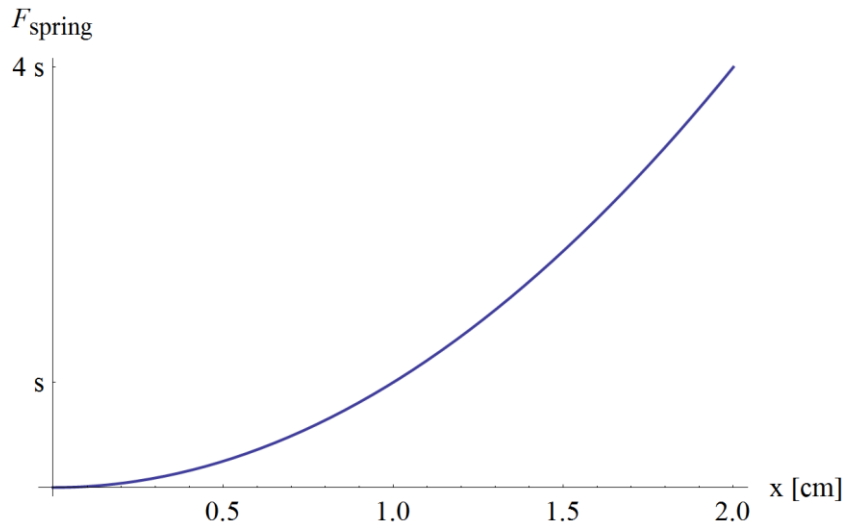


Fig. 4. Spring force characteristic of a progressive nonlinear spring

If we know the range of compression of the spring, the spring can be approximated with a linear characteristic.

$$\mathrm{F}_{lin} = a\,\mathrm{s}\,\mathrm{x}. \tag{4.2.2}$$

In this case the square of the difference of the parabola and a linear function is minimal if the stiffness of the linear spring is $\frac{3}{2}$s, so the value of parameter is $a = \frac{3}{2}$.

This kind of nonlinearity can be treated well by the simulator using a linear spring in the linear mechanical model and feeding back the difference between the force of the linear and nonlinear springs using *external force* and *displacement measuring* elements.

The vector of displacements can be obtained through the mapping matrix shown in CHAPTER 1.

$$\begin{bmatrix} \vdots \\ d_{spring} \\ \vdots \end{bmatrix} = \mathbf{d}(t) = \mathbf{P}\,\mathbf{q}(t), \tag{4.2.3}$$

then the force difference can be applied using the external force mapping matrix

$$\mathbf{Q}(t) = \mathbf{F}\,\mathbf{f}(t) = \mathbf{F}\begin{bmatrix} \vdots \\ f_{s,diff} \\ \vdots \end{bmatrix}, \tag{4.2.4}$$

where the external force is the force difference between the linear and nonlinear springs:

$$f_{s,diff} = \mathrm{s}\, d_{spring}^2 - a\,\mathrm{s}\, d_{spring}. \tag{4.2.5}$$

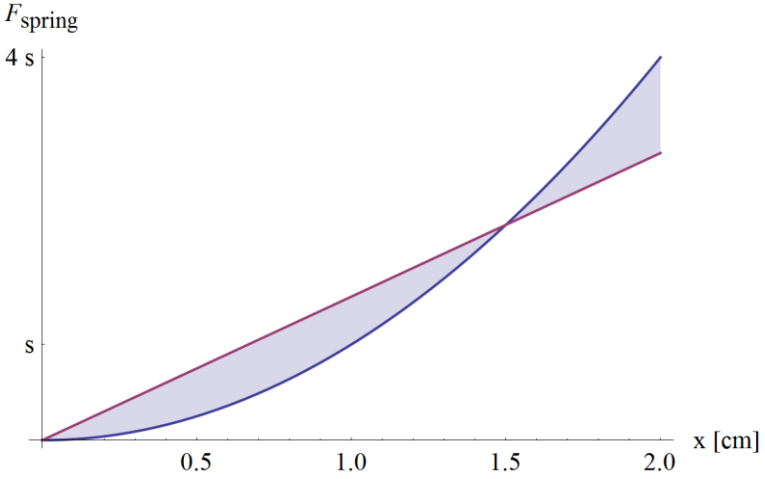


Fig. 5. Spring force difference

However the feedback of the actual difference of spring forces is only possible in the next time step, so this feedback is delayed.

Consider the following simple example of nonlinear connection between two cars in a rail vehicle:

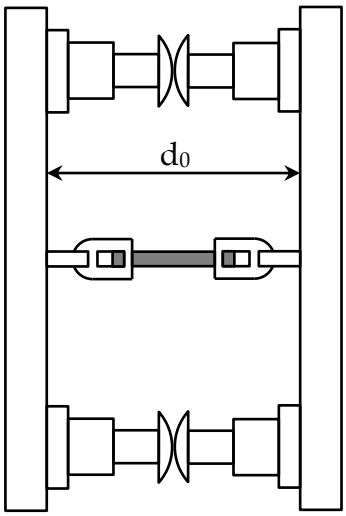


Fig. 6. "Buffers and chain" between two cars (top view)

In the simplest case the chain can be considered as a very stiff spring with stiffness s_{chain} and the spring force characteristic is piecewise linear

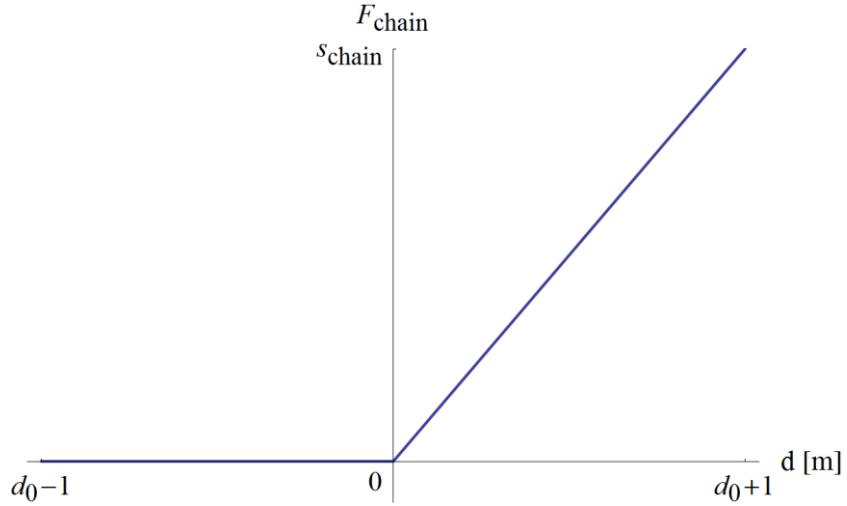


Fig. 7. Nonlinear chain characteristic

The buffers can be considered similarly as nonlinear dampers with damping ratio d_{buffer}, and damping characteristic:

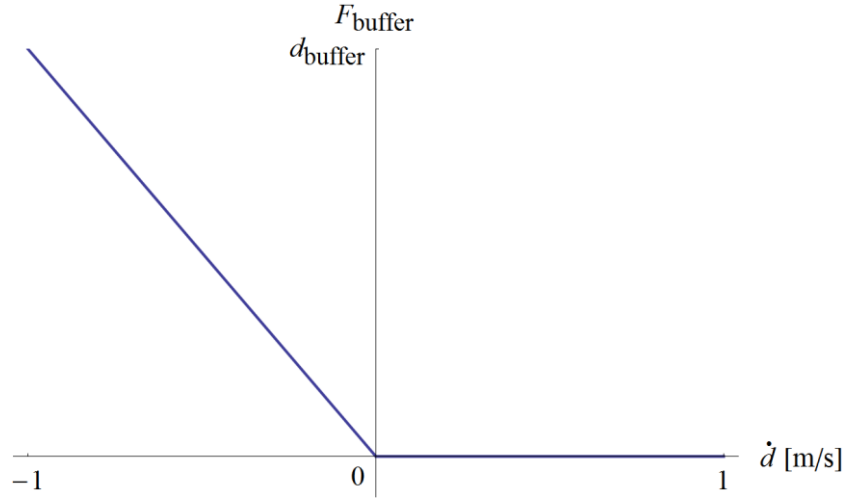


Fig. 8. Nonlinear buffer characteristic

NOTE: The above characteristic is only valid, when the buffers are in contact ($d < d_0$).

This kind of nonlinearity, where piecewise linear characteristics occur, can be handled easily by generating the stiffness, damping matrices and solution operators for all combinations of the system with linear characteristics.

In this simple case we have 4 different combinations (2 times 2), however with more nonlinear components, the number of combinations quickly grows, and therefore generating the system matrices and storing them requires more time and memory.

NOTE: Matrices are generated before the simulation, therefore the speed of the simulation is only affected by the time, which determining the actual configuration (i.e. which matrices to use) takes.

4.3 SUMMARY

Currently, the simulator system can handle nonlinear elements using the previously described strategies. Because of the main direction of this final project was the development and analysis of the partitioned simulator system, the strategies of handling nonlinear elements were not examined. The analysis of these strategies is planned in future works.

5 MULTIRATE PARTITIONED SIMULATION METHODS

In this chapter, some of the most referenced publications about multirate partitioned methods [11-15] are briefly summarized, an example of multi rate method is shown and extrapolation techniques are summarized.

Multirate methods are designed to solve systems of ordinary differential equations consisting of subsystems with different time scales. Multirate schemes exploit the different time scales by using different time steps for the subsystems.

Multi rate methods have three classes:

- Multi-step methods with fixed partitions
- Multi-stage methods with two partitions (active and latent)
- Self-adjusting multirate methods which determine the partitions automatically

Methods within these classes can utilize fixed or adaptively changing step sizes for the partitions, and should decide the order of computation of the subsystems.

Generally if step sizes are fixed, *fast first* strategy is used, but if step sizes are changing during simulation, *slow first* strategy is more advisable.

The most important difference between multi-step and multi-stage methods is that multi-stage methods avoid the coupling (and therefore the extrapolations and interpolations of state variables) between the active and latent partitions.

The stability analysis of these methods can be found in [13].

[14] presents an example of multirate extension of higher order numerical methods and [15] shows a multirate approach on solving problems related to electrical networks.

5.1 GENERALIZED MULTI-STAGE PARTITIONED RUNGE-KUTTA METHOD

In general, the whole system can be described by an initial value problem of a system of ordinary differential equations

$$\dot{\mathbf{y}} = \mathrm{f}(\mathrm{t}, \mathbf{y}), \qquad \mathbf{y}(\mathrm{t}_0) = \mathbf{y}_0 \tag{5.1.1}$$

By identifying different subsystems, the system can be partitioned into p subsystems.

$$\dot{y}_1 = f_1(t, y_1, \dots, y_p), \tag{5.1.2}$$
$$\vdots$$
$$\dot{y}_p = f_p(t, y_1, \dots, y_p),$$

with appropriate initial values.

Let us assume that the system is autonomous and that the subsystems can be grouped into only two parts: latent components and a smaller number of active components.

$$\dot{\mathbf{y}}_A = f_A(\mathbf{y}_A, \mathbf{y}_L), \qquad \mathbf{y}_A(t_0) = \mathbf{y}_{A0} \tag{5.1.3}$$
$$\dot{\mathbf{y}}_L = f_L(\mathbf{y}_A, \mathbf{y}_L), \qquad \mathbf{y}_L(t_0) = \mathbf{y}_{L0}$$

The components are integrated with different step sizes, the active components ($\mathbf{y}_A$) with the micro time step h and the latent components ($\mathbf{y}_L$) with the macro time step H. Macro and micro time steps are assumed to be constants

$$H = m\,h \tag{5.1.4}$$

where m is a positive integer.

The method assumes that all the stiffness is contained in the latent part and the active part is characterized by rapid changes. This assumption leads to a mixed implicit/explicit approach, i.e. to compute the latent part implicitly but the active part explicitly.

The simulation scheme starts with a compound macro and first micro step:

$$\mathbf{y}_L^H = \mathbf{y}_{L0} + (\boldsymbol{b}_L^{\mathrm{T}} \otimes \boldsymbol{I}_L)\boldsymbol{k}_L, \tag{5.1.5}$$
$$\mathbf{y}_{A,1}^h = \mathbf{y}_{A0} + (\boldsymbol{b}_A^{\mathrm{T}} \otimes \boldsymbol{I}_A)\boldsymbol{k}_A^0,$$
$$\boldsymbol{k}_L = H f_L(\overline{\boldsymbol{Y}}_A, \boldsymbol{Y}_L) + H(\widetilde{\boldsymbol{G}} \otimes \boldsymbol{J}_L)\boldsymbol{k}_L,$$
$$\boldsymbol{k}_A^0 = h f_A(\boldsymbol{Y}_A^0, \overline{\boldsymbol{Y}}_L^0),$$

with the Jacobian:

$$\boldsymbol{J}_L = \frac{\delta f_L}{\delta x}(\mathbf{y}_{A0}, \mathbf{y}_{L0}). \tag{5.1.6}$$

The intermediate stage values $\overline{\boldsymbol{Y}}_A, \overline{\boldsymbol{Y}}_L^0, \boldsymbol{Y}_A^0$ and $\boldsymbol{Y}_L$ are given in [11].

Then the successive micro time steps for l = 1, 2, … m.

$$\mathbf{y}_{A,l+1}^h = \mathbf{y}_{Al} + (\boldsymbol{b}_A^{\mathrm{T}} \otimes \boldsymbol{I}_A)\boldsymbol{k}_A^l, \tag{5.1.7}$$
$$\boldsymbol{k}_A^0 = h f_A(\boldsymbol{Y}_A^l, \overline{\boldsymbol{Y}}_L^l),$$

the intermediate stage values $\boldsymbol{Y}_A^l, \overline{\boldsymbol{Y}}_L^l$ are also given in [11]. $\boldsymbol{I}_{\mathrm{A}}$ and $\boldsymbol{I}_{\mathrm{L}}$ denotes identity matrices with corresponding dimensions, and $\mathbf{b}_{\mathrm{A}}$ and $\mathbf{b}_{\mathrm{L}}$ are vectors of constants.

To summarize this method, we have to highlight the following characteristics:

- Assume, that all the stiffness is contained by the latent components.
- Implicit calculation of macro and first micro step (coupled).
- Explicit calculation of the rest of micro steps (decoupled).

5.2 MULTIRATE EXTRAPOLATION METHODS

In general, multi-step multirate methods consist of subsystems, which are coupled therefore during the simulation of the different components, extrapolation and interpolation of different state variables are necessary.

[12] presents different extrapolation methods and strategies, and introduces a multirate method based on Richardson extrapolation. Its basic idea is to stop building up the extrapolation tableau for components that have been recognized to be already sufficiently accurate.

[12] categorizes extrapolation techniques into the following groups

- Classical extrapolation, where one constructs a table of EULER (or RUNGE-KUTTA) approximations for the needed state variables.
- Multirate extrapolation, where the construction of the approximation table is stopped at those components which are sufficiently accurate.

5.3 SUMMARY

The literature of multirate methods contains many advanced methods. In the case of this final project however, only a simplified approach is used. The partitions of the vehicle model are fixed; linear system of vehicle bodies (latent partition) and nonlinear subsystem of wheels and rail-wheel contact (active partition). In our system unavailable state variables are treated as constants (i.e. extrapolated as constant functions), and after finishing a macro time step, refining iterations are possible (where previously unavailable state variables becomes available).

The multirate approach used in our system is shown in CHAPTER 7, where the cooperation of the two subsystems is described.

6 DIVISION OF RAIL VEHICLE INTO SUBSYSTEMS

This chapter shows the division of the rail vehicle into subsystems. The general approach is to divide the vehicle into a linear, latent part (dynamics of vehicle body – cars, bogies, etc.), and a nonlinear, active part (dynamics of wheels).

First the complete model of the rail vehicle is presented then the subsystems and the interface between them are described. The subsystems in the simulator are defined so, that they can be replaced by more complex models if required.

6.1 MODEL OF RAIL VEHICLE

The complete in-plane model of the rail vehicle can be separated into the following components (from the top of the vehicle to the rail)

- Dynamics of vehicle bodies (cars, bogies, etc.)
- Contact between wheels and bogies (primary suspension)
- Dynamics of wheels
- Contact between wheels and rail

6.2 LINEAR PART – DYNAMICS OF VEHICLE BODIES

The linear part of the model is the multibody system consisting of 2D rigid bodies, linear springs and dampers, described in CHAPTER 1.

CHAPTER 3 shows how this part can be solved using the theory of fundamental matrices. Recall the solution described in CHAPTER 3.8.

$$\mathbf{y}_{n+1} = \mathbf{\Phi}(\Delta t)\,\mathbf{y}_n + \mathbf{\Omega}_0(\Delta t)\,\mathbf{b}_{0,n} + \mathbf{\Omega}_1(\Delta t)\,\mathbf{b}_{1,n}. \tag{6.2.1}$$

The state vector $\mathbf{y}$ contains the velocities and displacements, angular velocities and rotation angles of vehicle bodies.

$$\mathbf{y}(t) = \begin{bmatrix} \dot{\boldsymbol{q}}(t) \\ \boldsymbol{q}(t) \end{bmatrix}. \tag{6.2.2}$$

The linear part of the vehicle model contains "everything above the wheels", so in case of the simple 17 DoF train model (used in CHAPTER 8), the linear part consists of

- a 3 DoF car body and
- two 3 DoF bogies (attached to the car body via the secondary suspension)
- four nodes (attached to the bogies through the primary suspension)

The interface of this subsystem is defined by the nodes at the position of axles, as shown in the following figure:

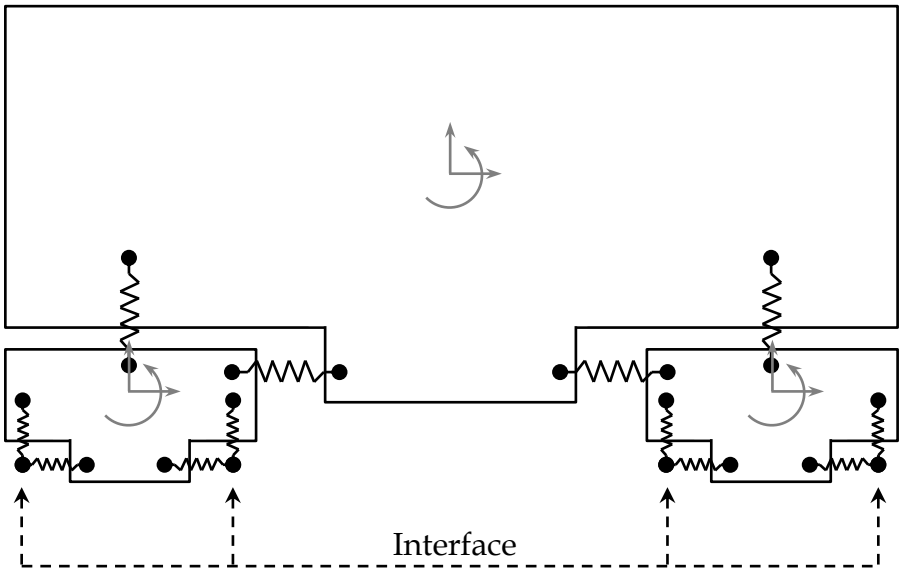


Fig. 9. Illustration of the linear part of the vehicle model and its interface

NOTE: The horizontal dynamics of wheels can be included in the linear part using rigid body elements with 1 DoF in the horizontal direction.

The linear part is considered as the latent part of the system and is solved using equation (6.2.1) at constant – macro – time step.

6.3 INTERFACE BETWEEN THE LINEAR AND NONLINEAR PARTS

The dynamics of the wheels and the dynamics of the vehicle bodies can be joined through external forces. In this case external force elements are needed to be defined along the interface and during simulation, the effect of other mechanical components of the vehicle can be introduced as a set of external forces.

6.4 NONLINEAR PART – DYNAMICS OF WHEELS

Recall the free body diagram of the wheel presented in CHAPTER 1, the interface of the subsystem is denoted by dashed arrows.

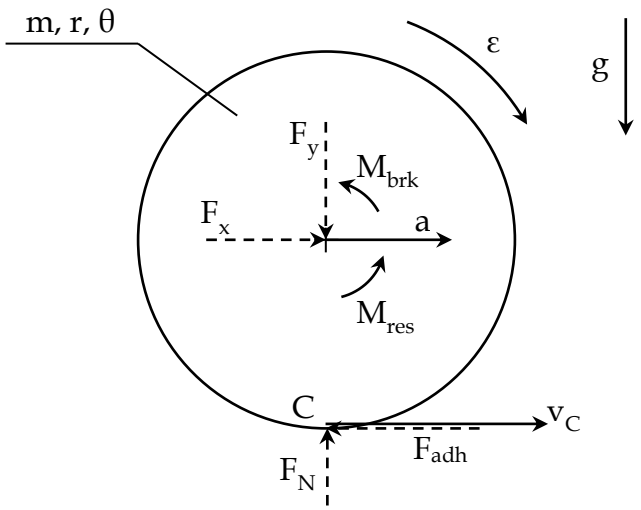


Fig. 10. Free body diagram of the wheel, and its interfaces

The equation of motion of the wheel (for the rotation) is:

$$\theta\,\dot{\omega} = \mathrm{r}\,\mathrm{F}_{adh} - \mathrm{M}_{res} - \mathrm{M}_{brk}, \tag{6.4.1}$$

where θ and r is the inertia and radius of the wheel, M_{res} is the internal resistance (friction) at the bearing of the axle, F_{adh} is the adhesion force at the wheel-rail contact and M_{brk} is the brake moment.

The absolute value of the brake moment can be expressed as:

$$|\mathrm{M}_{brk}| = \mathrm{r}_{brk}\,\mathrm{F}_{brk}, \tag{6.4.2}$$

and has the same sign as ω, therefore:

$$\mathrm{M}_{brk} = sgn(\omega)\,\mathrm{r}_{brk}\,\mathrm{F}_{brk}. \tag{6.4.3}$$

The adhesion force is:

$$|\mathrm{F}_{adh}| = \mathrm{F}_{\mathrm{N}}\,\mu(v,\omega), \tag{6.4.4}$$

where $\mu(v,\omega)$ function is the main nonlinearity, and is described in CHAPTER 6.5. The sign of the rail-wheel adhesion depends on the velocity of the rail-wheel contact point ($v_{\mathrm{C}} = v - r\omega$), therefore the adhesion force can be written as:

$$\mathrm{F}_{adh} = sgn(v_{\mathrm{C}})\,\mathrm{F}_{\mathrm{N}}\,\mu(v,\omega), \tag{6.4.5}$$

NOTE: $v_{\mathrm{C}} > 0$ means that the angular velocity is not enough, therefore the rail-wheel adhesion force increases ω.

The internal resistance depends on the angular velocity also:

$$\mathrm{M}_{res} = sgn(\omega)\, res(v, \omega), \tag{6.4.6}$$

where $res(v, \omega)$ is a function for the absolute value of internal resistances for a specific state of the wheel.

The state space of a wheel is divided into four regions, by the signs of adhesion force and braking moments ($\mathrm{M}_{brk}, \mathrm{M}_{res}$) i.e. by the sign of the angular velocity and axle velocity of the wheel.

In normal operation, the angular velocity is positive, and the velocity is also positive. In this case

- the region above the $v_C = 0$ line means that the wheel has lower angular velocity than which is needed for rolling, so the wheel is being slowed down and the train is braking.
- the region below the $v_C = 0$ line means, that the wheel has higher angular velocity than which is needed for rolling, so the wheel is driven, and the vehicle is accelerating.

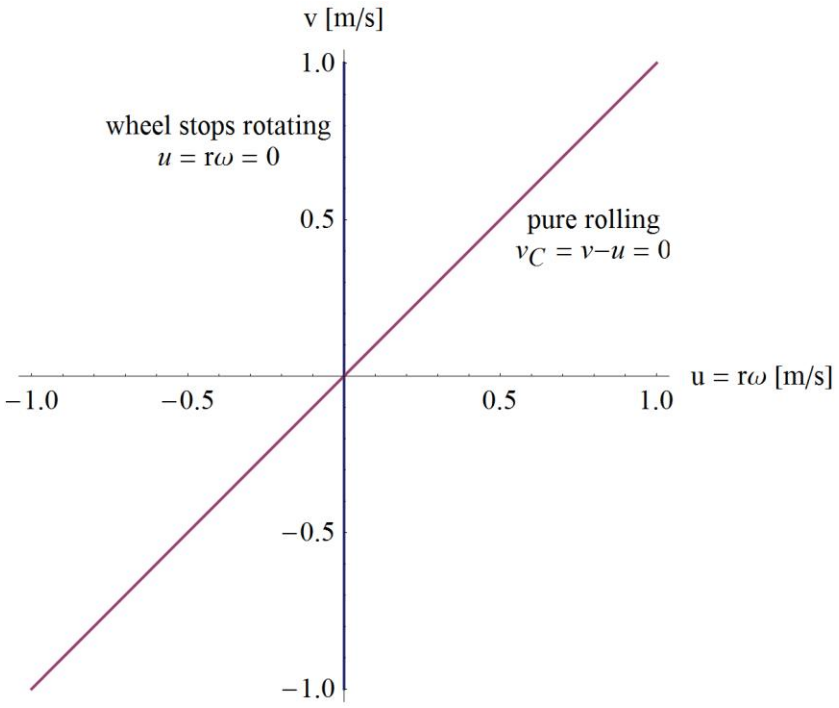


Fig. 11. Borders of the different regions in the state space of the wheel

The dynamics of the wheels is the active component of the system. The equations of motion of the wheels are solved by 4th order RUNGE-KUTTA method (for every wheel), at a – micro – time step, which is smaller than or equal to the macro time step.

The horizontal dynamics of the wheels can be included in the linear (latent) part of the system. If done so, the extrapolation is needed to be done on the following values

- horizontal velocity of the axle (v)
- brake moment (M_{brk})
- internal resistance (M_{res})
- rail-wheel adhesion force (F_{adh})

6.5 RAIL-WHEEL CONTACT

For the rail-wheel contact currently a very simple model is used. The adhesion capacity of the rail-wheel contact is given by the following exponential formula.

$$\mu(v,\omega) = \mu_0 \left((1-\mathrm{A})\, e^{\frac{-|v-r\omega|}{v^*}} + \mathrm{A} \right), \tag{6.5.1}$$

where μ_0, A and v^* are parameters and r is the radius of the wheel.

For the simulation the model is used with the following parameters

$$\begin{aligned} \mu_0 &= 0.1\ [-], \\ \mathrm{A} &= 0.2\ [-], \\ v^* &= 5\ \left[\frac{m}{s}\right]. \end{aligned} \tag{6.5.2}$$

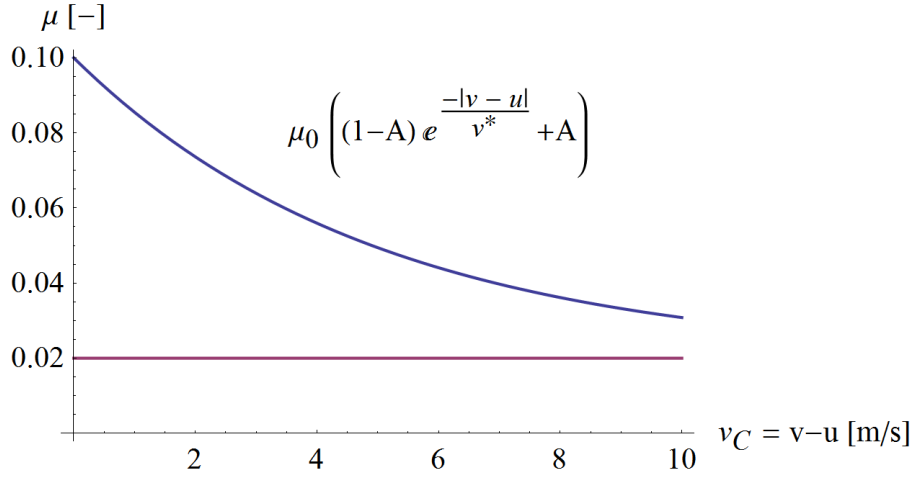


Fig. 12. Function of rail-wheel adhesion capacity and its asymptote

The sign of the adhesion capacity depends on the velocity of the contact point (v_C), as shown in (6.4.5):

$$\mathrm{F}_{adh} = sgn(v_\mathrm{C})\, \mathrm{F}_\mathrm{N}\, \mu(v,\omega), \tag{6.5.3}$$

The rail-wheel adhesion force behaves in a way, that around smaller (or zero) contact point velocities, the actual adhesion force is smaller than the capacity (meaning that, the system does not use the available capacity – $\mathrm{F}_\mathrm{N}\, \mu$)

The simplest way to model this behaviour is replacing the discontinuity of the sign function in (6.5.3) is by a linear region. For this, a continuous sign function ($\widehat{sgn}$) is introduced:

$$\widehat{sgn}(x) = \begin{cases} x\ \frac{1}{\mathrm{T_{sgn}}}, & |x| < \mathrm{T_{sgn}} \\ sgn(x), & |x| \geq \mathrm{T_{sgn}} \end{cases} \tag{6.5.4}$$

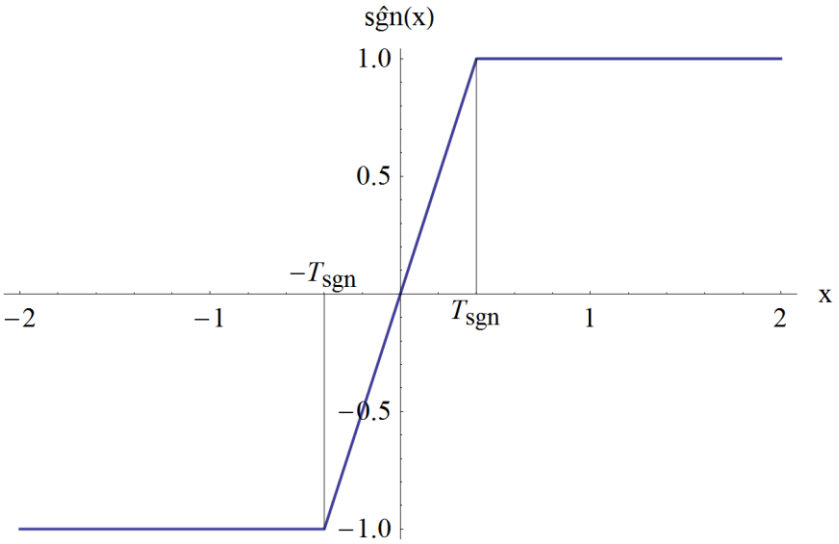


Fig. 13. Continuous sign function

With this model, the adhesion force tends to zero as the system approaches pure rolling. Thus the adhesion force will not push the system over the $v_C = 0$ line and the system will not oscillate about the state of rolling.

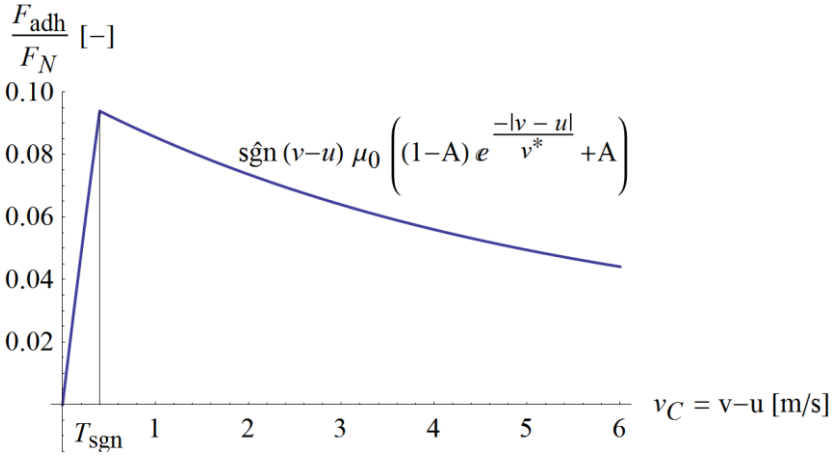


Fig. 14. The ratio of the adhesion and normal force

The value of the sign threshold ($\mathrm{T_{sgn}}$) has key importance. At small micro time steps, a smaller value can be used without having the above described oscillation. See CHAPTER 9.2 for further details.

However this friction model is quite simple, it is not included in the dynamics of the wheel (calculated in a different module, at macro time step), to preserve the modularity of the simulator and to make the model easily replaceable with more complex ones.

Application and analysis of more complex friction models is possible and planned for the future.

7 FRAMEWORK FOR INVESTIGATING THE COOPERATION OF THE SUBSYSTEMS

In this chapter the framework of the simulator is described. The simulator consists of the subsystems shown in CHAPTER 6. The relationship between the components of the framework is briefly described then the key parameters of the subsystems are identified, finally different simulation strategies are shown.

7.1 COMPONENTS OF THE FRAMEWORK

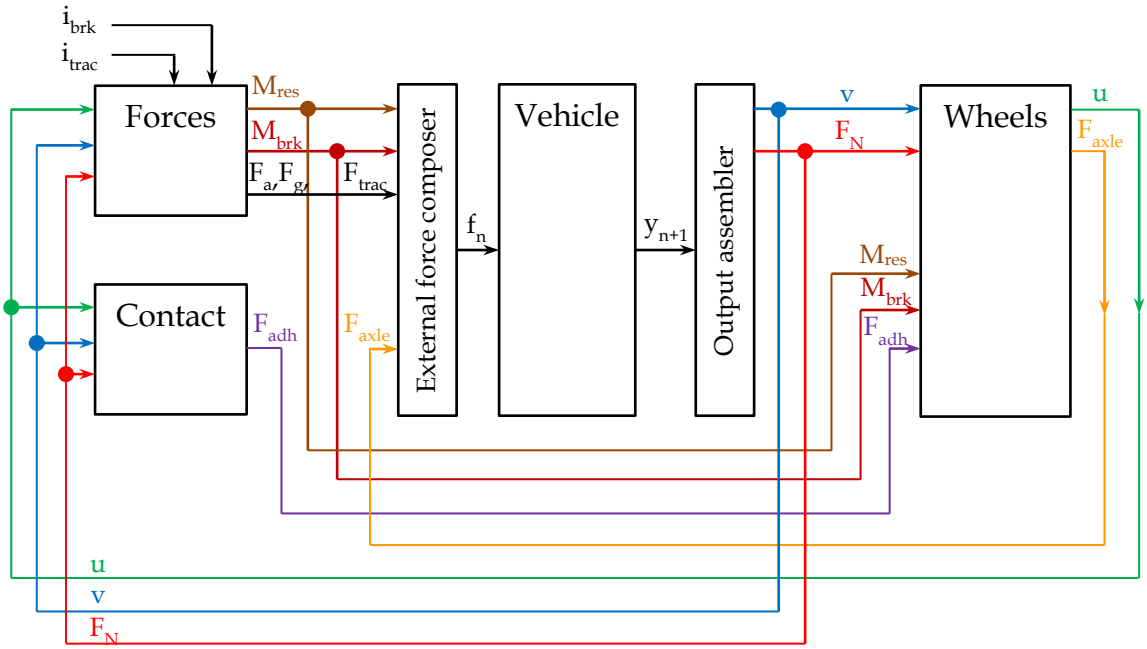


Fig. 15. Subsystems of the simulator

The simulator framework – see Fig. 15 – was developed and implemented in C++ programming language. The modules of the framework are well defined, replaceable classes taking the role of the modules shown in CHAPTER 6.

The notations used in the above figure are:

Notation	Description
y	State vector of the linear part, see (3.8.9)
f	Inhomogeneity of the linear part, see (3.8.8)
u	Vector of wheel circumferential velocities ($u = r\ \omega$)
v	Vector of horizontal wheel (axle) velocities
F_N	Vector of normal forces acting on axles
F_{axle}	Vector of horizontal forces acting on axles
F_{adh}	Vector of rail-wheel adhesion forces
F_a	Air resistance
F_g	Horizontal component of gravity
F_{trac}	Traction force (or locomotive force), pulling the vehicle
M_{res}	Vector of internal resistance moments on axles
M_{brk}	Vector of brake moments (acting on wheels and bogies)
i_{trac}	Input signal indicating traction
i_{brk}	Input signal indicating braking

The simulator contains 4 subsystems, which are briefly described in this chapter.

The FORCES MODULE calculates all forces, which are acting between the components of the vehicle. These forces and moments are

$$\mathbf{M_{res}} = f(\boldsymbol{u}, \boldsymbol{v}, \mathbf{F_N}), \tag{7.1.1}$$

$$\mathbf{M_{brk}} = f(\boldsymbol{u}, \mathrm{i_{brk}}),$$

$$\mathrm{F_g} = f\left(\int \boldsymbol{v}\,\mathrm{dt}\right),$$

$$\mathrm{F_a} = f(\boldsymbol{v}),$$

$$\mathrm{F_{trac}} = f(\mathrm{i_{trac}}).$$

The CONTACT MODULE calculates the adhesion capacity, using the model described in CHAPTER 6.5.

$$\mathbf{F_{adh}} = f(\boldsymbol{u}, \boldsymbol{v}, \mathbf{F_N}). \tag{7.1.2}$$

The VEHICLE MODULE calculates its own state vector and the normal forces and horizontal axle velocities using the mapping matrices defined in CHAPTER 1 and [1].

$$\mathbf{b}(\mathrm{t}) = \mathbf{b}_0 + \mathbf{b}_1\,\mathrm{t}, \tag{7.1.3}$$

$$\mathbf{b}_n = \mathbf{B}\,\mathbf{f}_n,$$

$$\mathbf{y}_{n+1} = \boldsymbol{\Phi}(\Delta\mathrm{t})\,\mathbf{y}_n + \boldsymbol{\Omega}_0(\Delta\mathrm{t})\,\mathbf{b}_{0,n} + \boldsymbol{\Omega}_1(\Delta\mathrm{t})\,\mathbf{b}_{1,n},$$

$$\mathbf{F_N} = f(\mathbf{y}, mapping\ matrices),$$

$$\boldsymbol{v} = select(\mathbf{y}).$$

The WHEELS MODULE calculates the angular velocities and positions of the wheels, and the horizontal force acting on the axles. The equation of motion is solved using 4th order RUNGE-KUTTA method at micro step size.

$$\boldsymbol{u} = \mathrm{RK4}\left(\theta\ \dot{\mathrm{u}} = \mathrm{F}_{adh} - (\mathrm{M}_{res} + \mathrm{M}_{brk})\frac{1}{r}\,,\ \delta\mathrm{t}\right), \tag{7.1.4}$$

$$\mathbf{F_{axle}} = f(\mathbf{F_{adh}}).$$

7.2 FLOW-CHART OF SIMULATION

The operation of the framework is shown in Fig. 16. The simulation begins with opening the model and configuration files, calculating mass-, stiffness- and damping matrices and mapping matrices of the linear components. Afterwards the fundamental matrices and solution operators of the inhomogeneity are prepared for the VEHICLE MODULE and the necessary parameters of wheels are given to the WHEELS MODULE. Finally the initial condition and a pre-defined traction / brake signal history is read and the simulation is started.

The process involves two different time steps, macro time step (Δt) for the linear part of the system, and micro time step (δt) for the dynamics of wheels, where

$$\delta\mathrm{t} = \frac{\Delta\mathrm{t}}{\mathrm{N_{sub}}}, \qquad \mathrm{N_{sub}} \in \mathbb{N}^+. \tag{7.2.1}$$

The simulation process starts with updating the forces at the beginning of the macro time step (FORCES and CONTACT MODULES), then the VEHICLE MODULE simulates the oscillations of vehicle bodies, while considering its inputs as constant within the macro time step. Finally the WHEELS MODULE calculates the motion of the wheel by stepping $\mathrm{N_{sub}}$ steps (using the micro time step), considering the axle velocities and normal forces (v and $\mathrm{F_N}$) linear and the other inputs as constant within the macro time step.

At this point the macro step can be finished or refining iteration(s) can be started using the state of the system at the end of the macro time step.

By making additional (N_{iter}) iterations, the second evaluation of the VEHICLE and WHEELS MODULES can be done by approximating all of the inputs as a linear function within the macro time step (since an estimate of the system state at the end of the macro time step is available).

The flowchart of the simulation can be seen in the following figure:

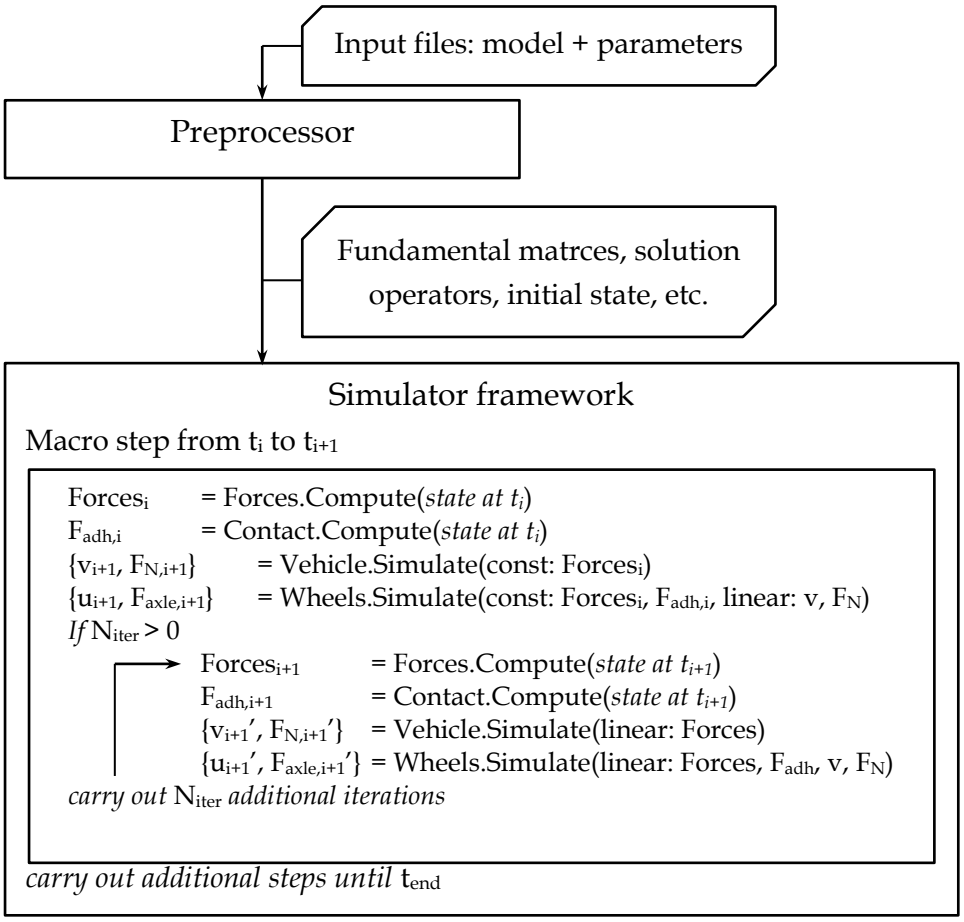


Fig. 16. Flow chart of the simulation indicating the iteration

NOTE: The micro steps in the WHEELS MODULE (Wheels.Simulate) are not shown in the figure.

7.3 SIMULATORS BASED ON FRAMEWORK CONFIGURATION

The key parameters of the framework are listed below:

- The main (macro) time step: Δt
- The subdivision of the macro time step: N_{sub}
- The number of additional iterations within a macro step: N_{iter}
- The threshold of the friction model's sign function: T_{sgn}

By varying the number of iterations and subdivision four class of simulators can be defined:

- NI+NM: non-iterating and non-multirate
- I+NM: iterating, but non-multirate
- NI+M: multirate, but non-iterating
- I+M: iterating and multirate

Integers	$N_{iter} = 0$	$N_{iter} > 0$
$N_{sub} = 1$	NI+NM	I+NM
$N_{sub} > 1$	NI+M	I+M

The error and computational time requirement of these four simulator types are examined in the following chapters, at different macro time steps and at different sign threshold values.

Our goal is to find the parameters of the appropriate configuration, matching the requirements.

8 VALIDATION

In this chapter, the simulator system presented in CHAPTER 7 is validated with 4th order RUNGE-KUTTA method in case of a 17 DoF rail vehicle model.

8.1 17 DOF TRAIN MODEL

The train model used for validation can be seen in Fig. 17. The model consists of 7 rigid bodies and 12 linear springs (with linear spring / damping force characteristics).

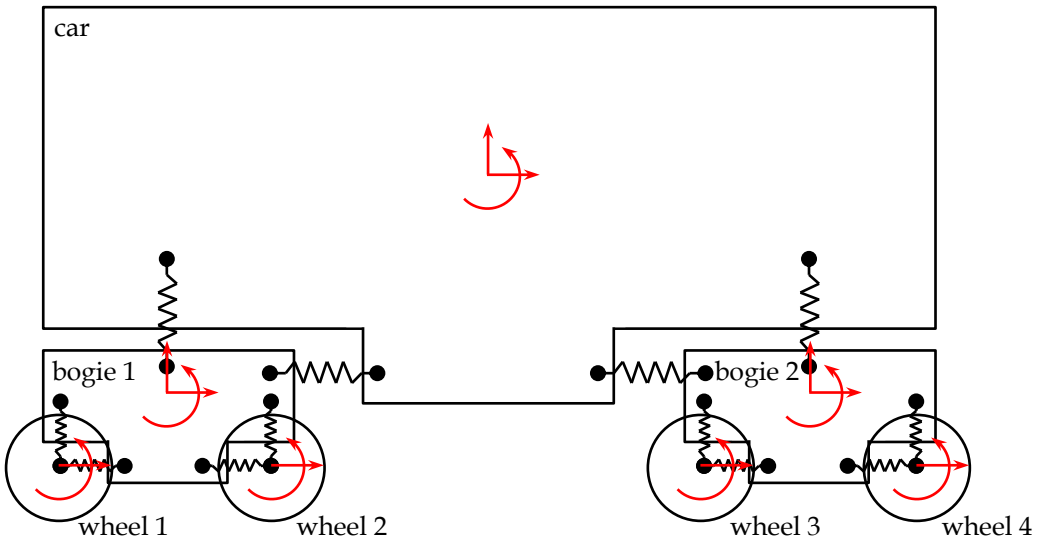


Fig. 17. Illustration of the 17 DoF train model.

The equations of motion of the car are:

$$a_{car,x} = \left(F_{loc} + F_{bog1,x} + F_{bog2,x}\right)/m_{car}, \tag{8.1.1}$$

$$a_{car,y} = \left(F_{bog1,y} + F_{bog2,y}\right)/m_{car},$$

$$\varepsilon_{car} = \left(\left(F_{bog1,x} + F_{bog2,x}\right) k_{c,y} + \left(F_{bog2,y} - F_{bog1,y}\right) k_{c,x} + F_{loc}\, k_{c,loc}\right)/\theta_{car},$$

where F_{loc} is the locomotive force, $F_{bog,x}$ and $F_{bog,y}$ (with the appropriate bogie indices) are the spring forces between the car and the bogies, $k_{c,x}$, $k_{c,y}$ and $k_{c,loc}$ are geometric parameters, m_{car} is the mass and θ_{car} is the inertia of the car body.

The equations of motion of a bogie:

$$a_{bog1,x} = \left(F_{whl1,x} + F_{whl2,x} - F_{bog1,x}\right)/m_{bog}, \tag{8.1.2}$$

$$a_{bog1,y} = \left(F_{whl1,y} + F_{whl2,y} - F_{bog1,y} + F_{brk1} - F_{brk2}\right)/m_{bog},$$

$$\varepsilon_{bog1} = \Big(\left(F_{whl1,x} + F_{whl2,x}\right) k_{b,y} + \left(F_{whl2,y} - F_{whl1,y}\right) k_{b,x} - \left(F_{brk2} + F_{brk1}\right) k_{b,brk} + F_{bog1,x}\, k_{b,car}\Big)/\theta_{bog},$$

where $F_{whl,x}$ and $F_{whl,y}$ (with the appropriate wheel indices) are the spring forces between the indicated wheel and the bogies, F_{brk} (with the appropriate wheel indices) is the brake force between wheels and bogies. $k_{b,x}$, $k_{b,y}$ $k_{b,brk}$ and $k_{b,car}$ are geometric parameters, m_{bog} is the mass and θ_{bog} is the inertia of the bogie.

The equations of motion of a wheel:

$$a_{whl1,x} = \left(-F_{adh1} - F_{whl1,x}\right)/m_{whl}, \tag{8.1.3}$$

$$\varepsilon_{whl1} = \left(F_{adh1} - F_{sum1}\right)/m_{whl,red},$$

where

$$F_{sum1} = F_{brk1}\, r_{brk}/r_{whl} + F_{res,1}, \tag{8.1.4}$$

$$F_{adh1} = sgn(v_{whl1} - \mathrm{r}\,\omega_{whl1})\, \mu(v_{whl1},\, \omega_{whl1})\, \left(F_{brk1} + F_{whl1,y} + F_{N0}\right),$$

F_{res} is the internal resistance at the axle (similarly as in CHAPTER 6), r_{brk}, r_{whl} are radius and hand of brake force on the wheel, m_{whl} and $m_{whl,red}$ are the mass and reduced mass ($\theta_{whl} = m_{whl,red}\, r_{whl}^2$) of the wheel.

8.2 TEST CASE

The simulator was validated with a simple test case. The rail vehicle was accelerated by a virtual locomotive force of 100 [kN] for about 1.5 seconds then decelerated by brake forces slightly shifted in time, in two stages. The first stage causes the wheels to reach zero angular velocity, during the time interval between the braking stages, the wheels will roll again, finally the second braking stage stops the vehicle without large wheel slips.

The pre-defined external force history can be seen on Fig. 18.

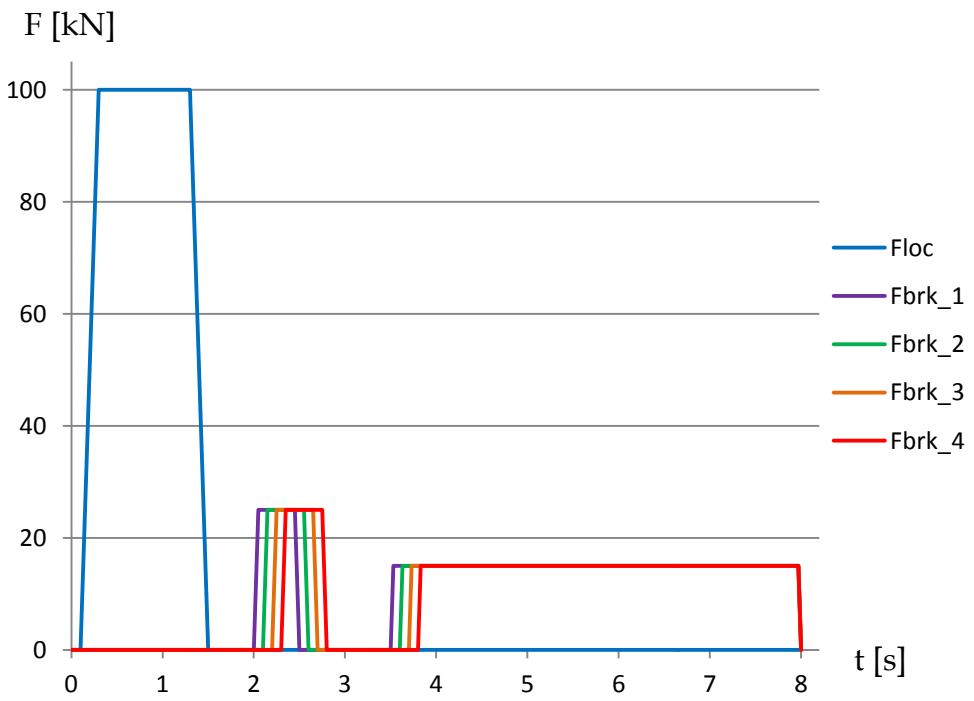


Fig. 18. Force history in the test case

8.3 RESULTS OF VALIDATION

The equations of the 17 DoF train model were solved by 4th order RUNGE-KUTTA method with pre-defined external forces described in CHAPTER 8.2.

The validator solver used a time step of 0.0005 seconds (0.5 ms). The absolute values of differences of state variables were integrated to determine the total error between the partitioned simulator and the validator.

The *sum* (with respect to 1 second of simulation) of position errors can be seen in the following figure. The error of position shows nearly quadratic dependence on the (macro) time step, the position error is approximated by the following function:

$$Err = a\,\Delta t^{b}. \tag{8.3.1}$$

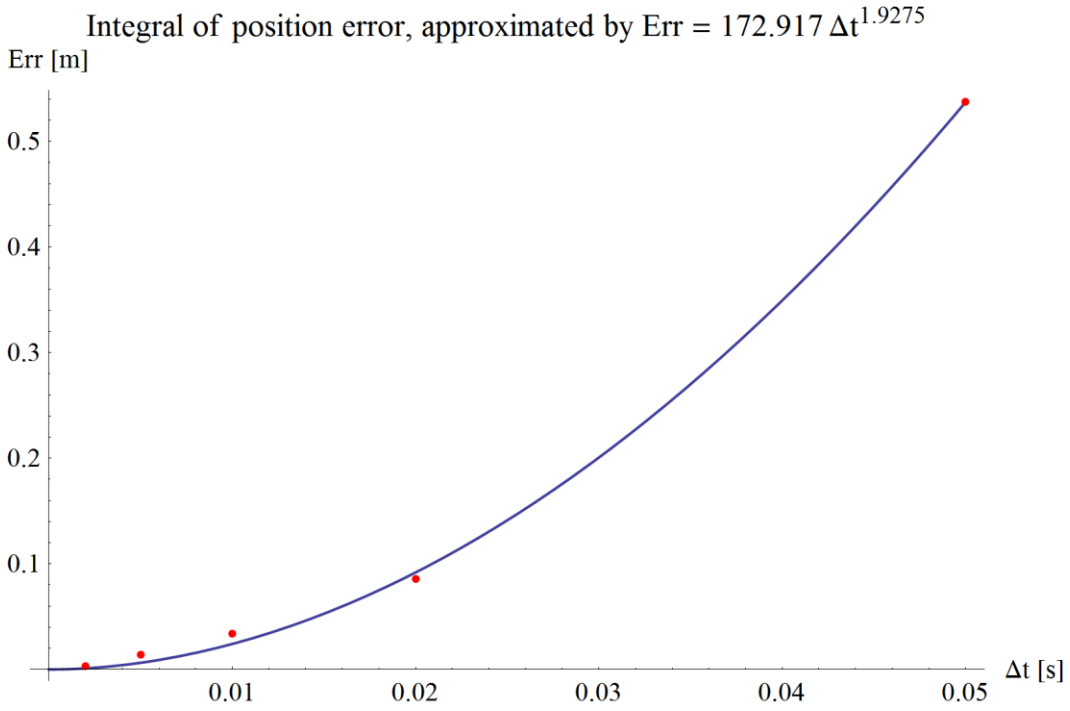
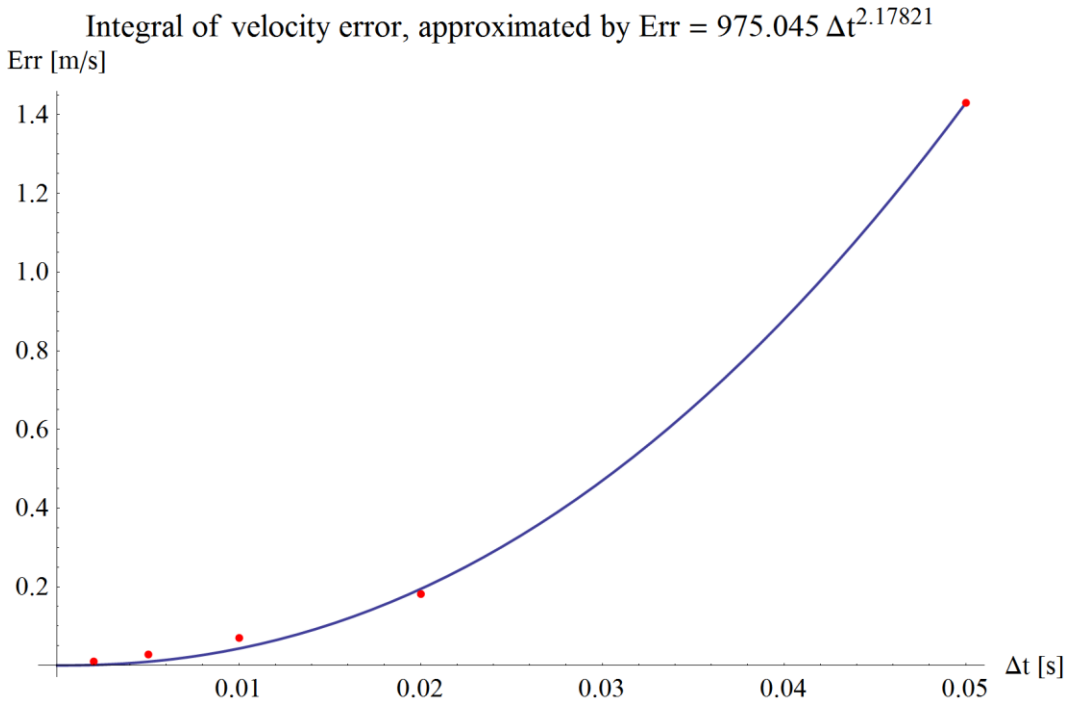


Fig. 19. Integral of position error and its approximation.

The error of velocity and its approximation:



As we can see, the errors are nearly quadratic with respect to the macro time step, and tending to zero as the time step decreases and therefore the partitioned simulator is considered to produce valid simulation results.

9 SIMULATION RESULTS

The optimum value of the sign function threshold in the rail-wheel contact is selected by examining a set of simulations. After that, a total of 180 simulations were carried out at five different macro time steps (with Δt = 0.05, 0.02, 0.01, 0.005 and 0.002 seconds), at six different iteration counts (N_{iter} = 0, 1, 2, 3, 4 and 5), with six different micro time steps (N_{sub} = 1, 2, 5, 10, 15 and 20) using the optimum T_{sgn}.

9.1 SELECTING THE OPTIMUM VALUE OF SIGN THRESHOLD

To study the effect of the threshold of the sign function in the rail-wheel contact model (see CHAPTER 6.5), 11 simulations were carried out with different sign threshold values. (T_{sgn} = 1, 0.5, 0.2, 0.1, 0.05, 0.02, 0.01, 0.005, 0.002, 0.001, 0.0005 [m/s])

This threshold value affects the system in two different ways.

First it changes the behaviour of the friction by introducing an elastic behaviour at the contact, meaning that it takes time for the friction force to build up. This causes the system to stop later at larger T_{sgn} values, or to oscillate around the pure rolling region at smaller T_{sgn} values (where the capacity of the adhesion is utilized fully.)

Secondly, micro oscillations around the pure rolling region can appear depending on the micro time step. At larger micro time steps, the adhesion force changes its sign repeatedly by pushing the system over the state of pure rolling, therefore changing its sign in the next micro step and causing the system to oscillate. At smaller micro time steps (or using larger T_{sgn}), the system can stay on one side of the pure rolling state, and no oscillations occur.

Let us examine the elastic behaviour of the friction model according to the value of T_{sgn}.

Consider the following plots with T_{sgn} = 1 and 0.5 [m/s]. It can be seen, that the train does not stop at the end of the simulation, thanks to the high threshold value.

VS	
time step	0.005
multirate	5
iterations	1
signum T	1

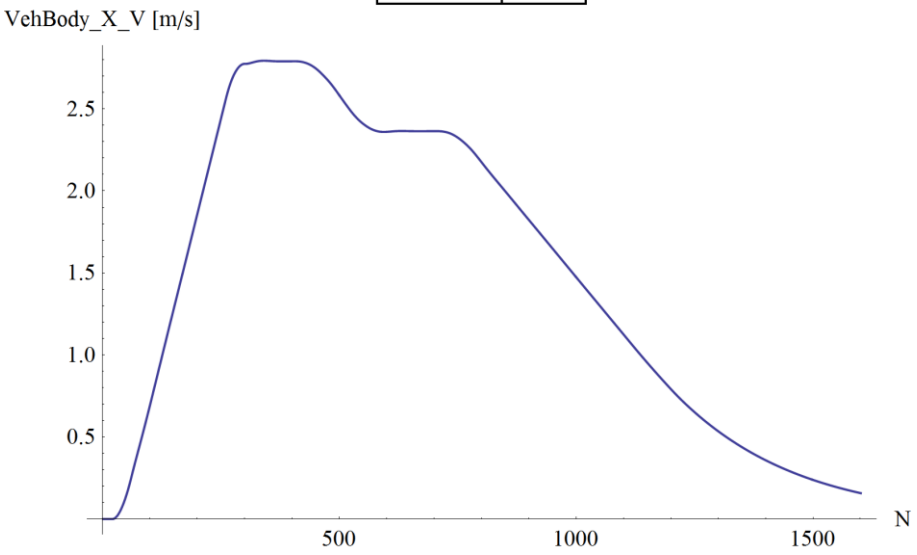


Fig. 20. Velocity plot of the vehicle with T_{sgn} = 1 [m/s]

VS	
time step	0.005
multirate	5
iterations	1
signum T	0.5

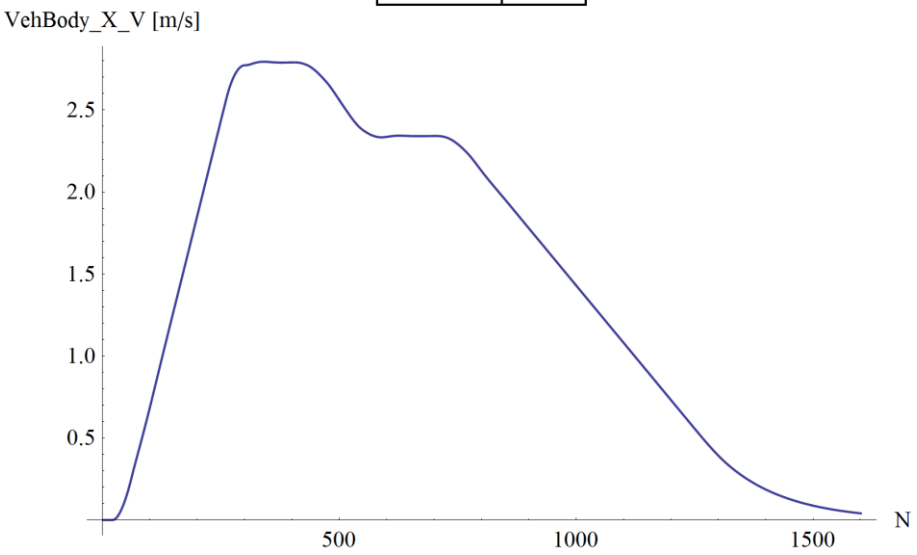


Fig. 21. Velocity plot of the vehicle with T_{sgn} = 0.5 [m/s]

It can be seen on the following velocity plots that the rail-wheel contact force builds up faster and the train actually stops at the end of the simulation, thanks to lower T_{sgn} values.

VS	
time step	0.005
multirate	5
iterations	1
signum T	0.2

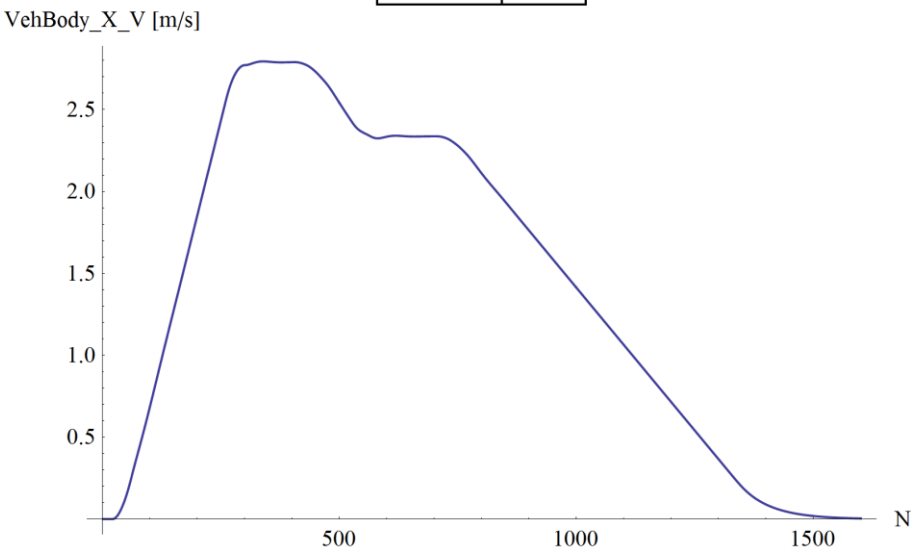


Fig. 22. Velocity plot of the vehicle with T_{sgn} = 0.2 [m/s]

VS	
time step	0.005
multirate	5
iterations	1
signum T	0.05

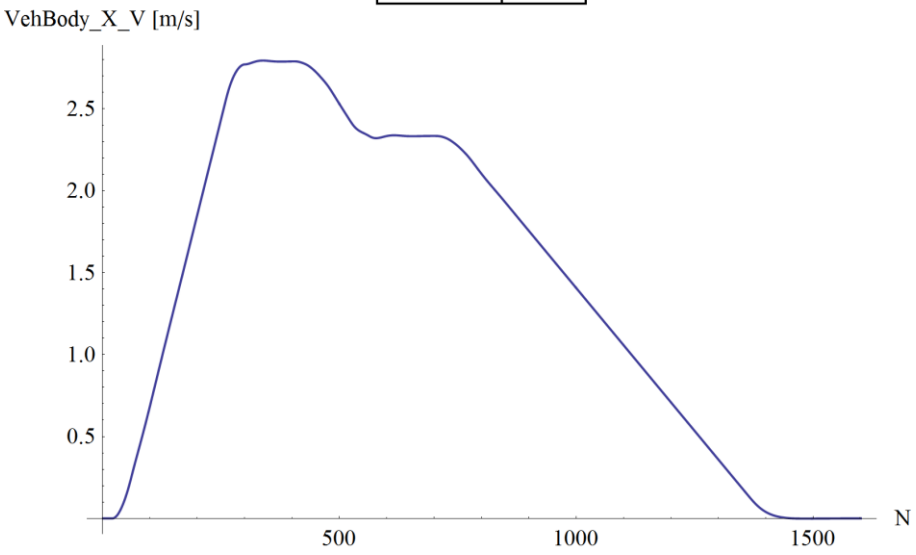


Fig. 23. Velocity plot of the vehicle with T_{sgn} = 0.05 [m/s]

By lowering the threshold value of the sign function, the adhesion capacity is utilized more and more suddenly, therefore the rail-wheel contact force pushes the system over the pure rolling state, and the velocity of the car starts to oscillate during stopping.

VS	
time step	0.005
multirate	5
iterations	1
signum T	0.01

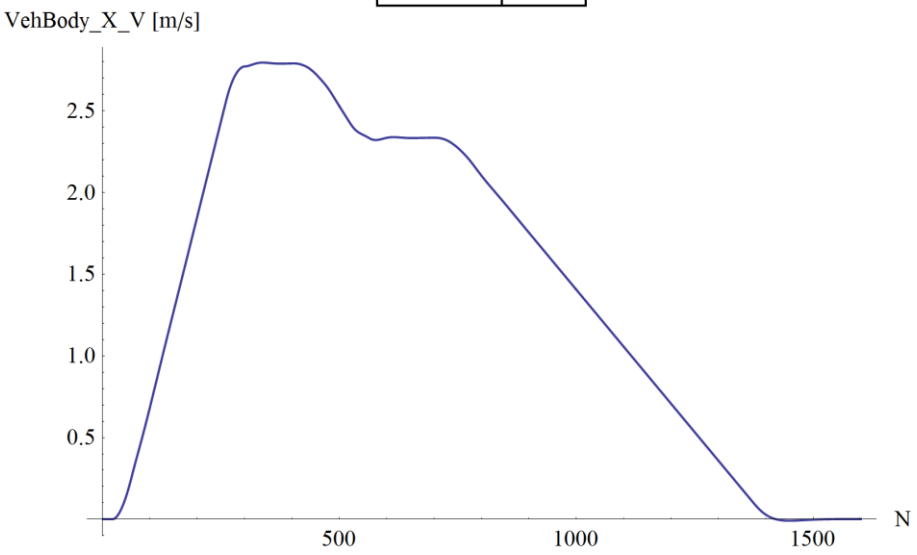


Fig. 24. Velocity plot of the vehicle with T_{sgn} = 0.01 [m/s]

VS	
time step	0.005
multirate	5
iterations	1
signum T	0.005

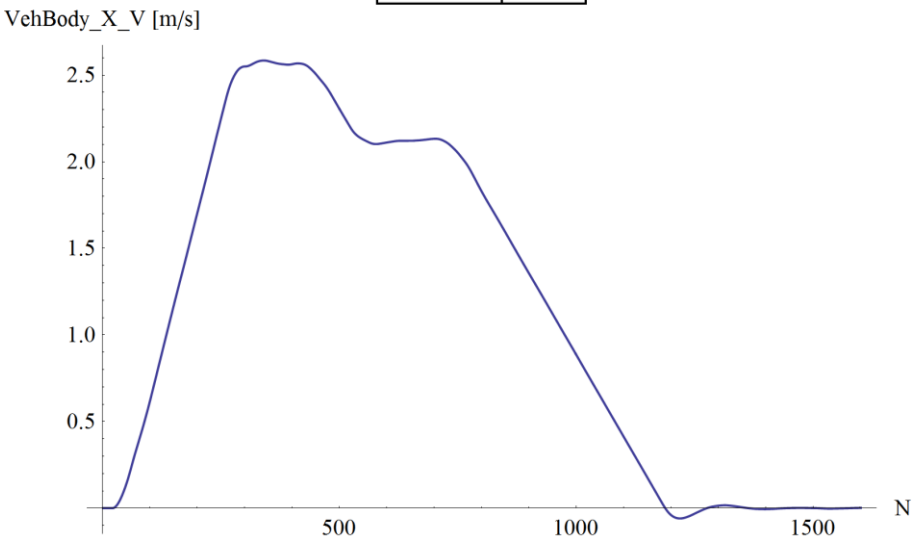


Fig. 25. Velocity plot of the vehicle with T_{sgn} = 0.005 [m/s]

By further examining the differences between various state variables, I've chosen 0.05 [m/s] as the optimum value of T_{sgn}. All further simulation results were calculated using this value.

9.2 INSTABILITY OF THE WHEEL DYNAMICS

As described before, micro oscillations around the pure rolling region can appear depending on the micro time step. At larger micro time steps, the adhesion force changes its sign repeatedly by pushing the system over the state of pure rolling, therefore changing its sign in the next micro step and causing the system to oscillate. At smaller micro time steps (or using larger T_{sgn}), the system can stay on one side of the pure rolling state, and no oscillations occur. Fig. 26 and Fig. 27 are illustrating these behaviours.

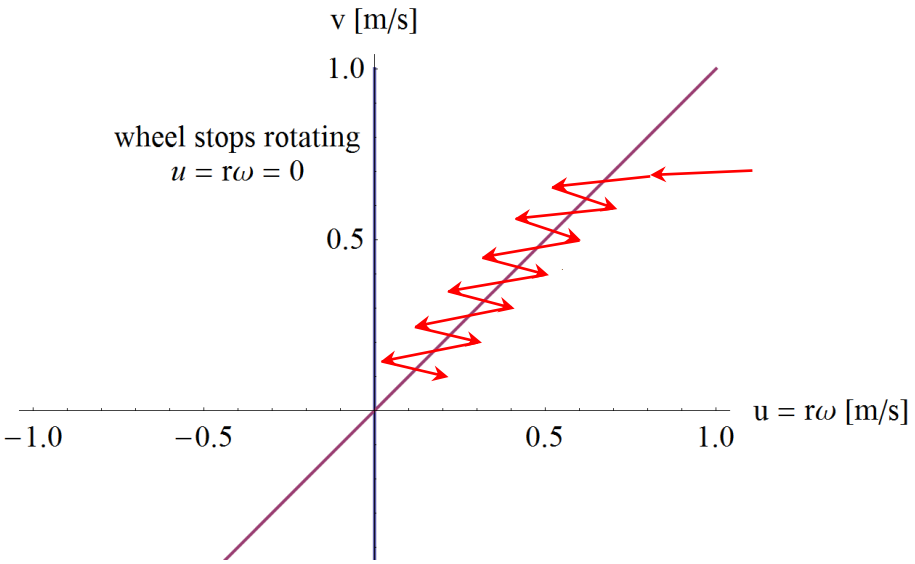


Fig. 26. Oscillation around the pure rolling

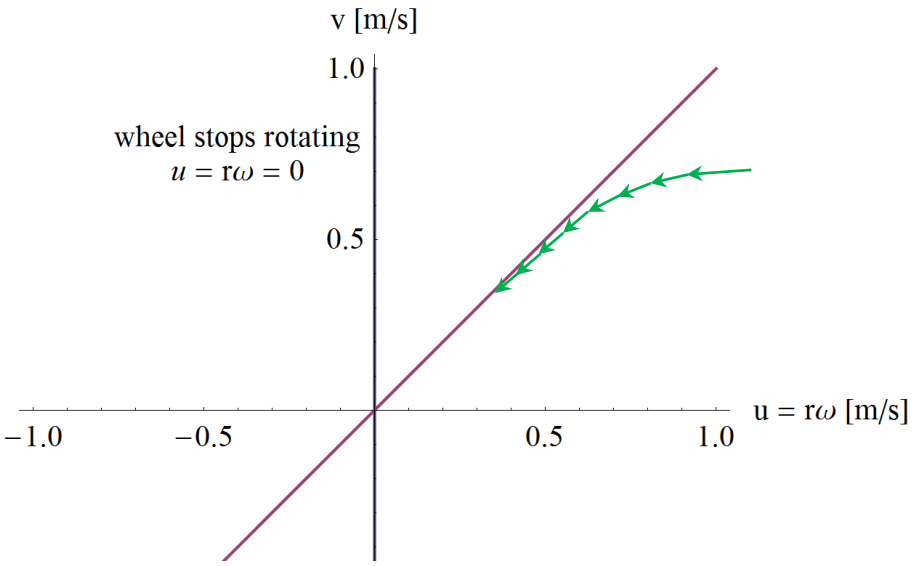


Fig. 27. No oscillations in case of smaller micro time steps

To find out, how this behaviour depends on the macro, micro time step and the number of iterations (Δt, N_{sub}, N_{iter}), the discrete Fourier transform of wheel velocities were analysed.

At relatively larger macro time step of $\Delta t = 0.02$ [s], with same micro time step ($N_{sub} = 1$) and without iterations ($N_{iter} = 0$), hard oscillations are clearly visible.

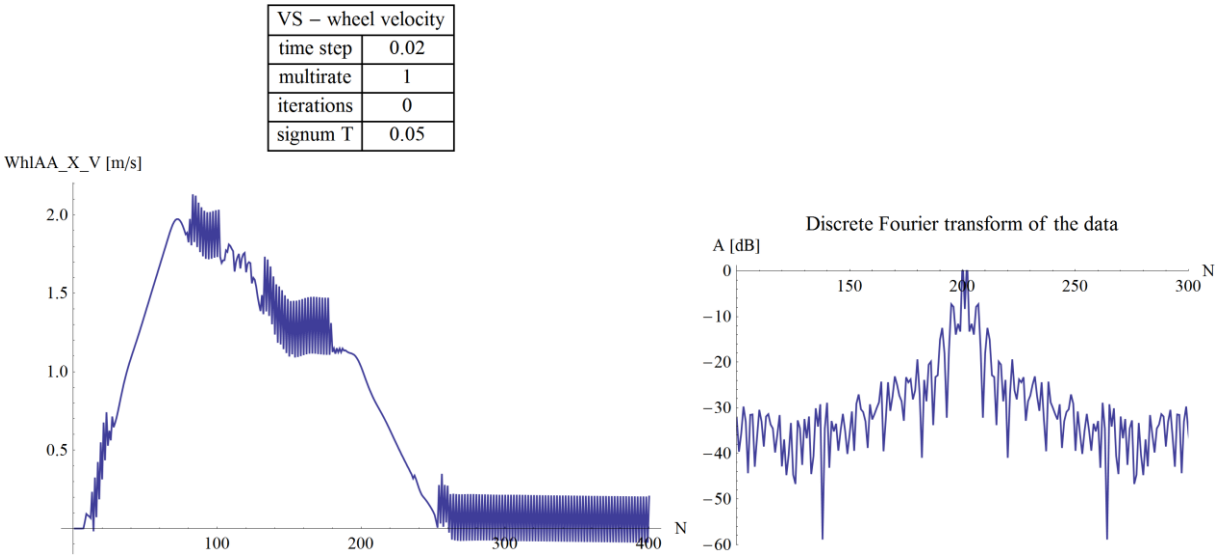


Fig. 28. Wheel velocity plot with $N_{iter} = 0$ and $N_{sub} = 1$, $\Delta t = 0.02$ [s]

By decreasing the micro time step, the oscillation begins to disappear.

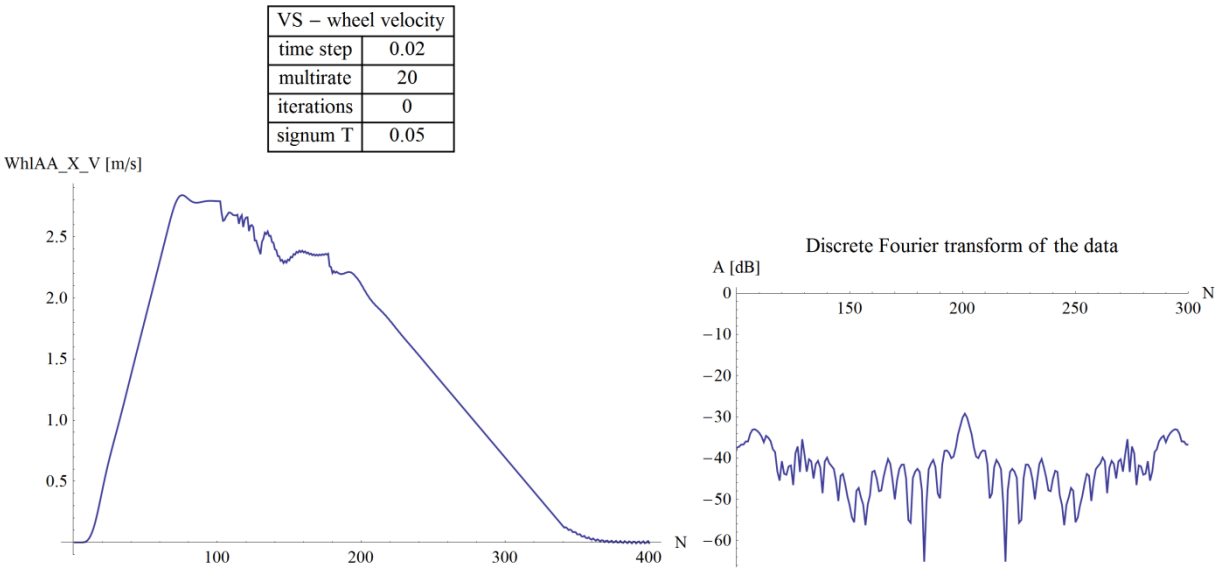


Fig. 29. Wheel velocity plot with $N_{iter} = 0$ and $N_{sub} = 20$, $\Delta t = 0.02$ [s]

By allowing one iteration for the simulator, the oscillation behaviour further reduces, and only minor oscillations remain at the end of the simulated time interval (during stopping).

VS – wheel velocity	
time step	0.02
multirate	20
iterations	1
signum T	0.05

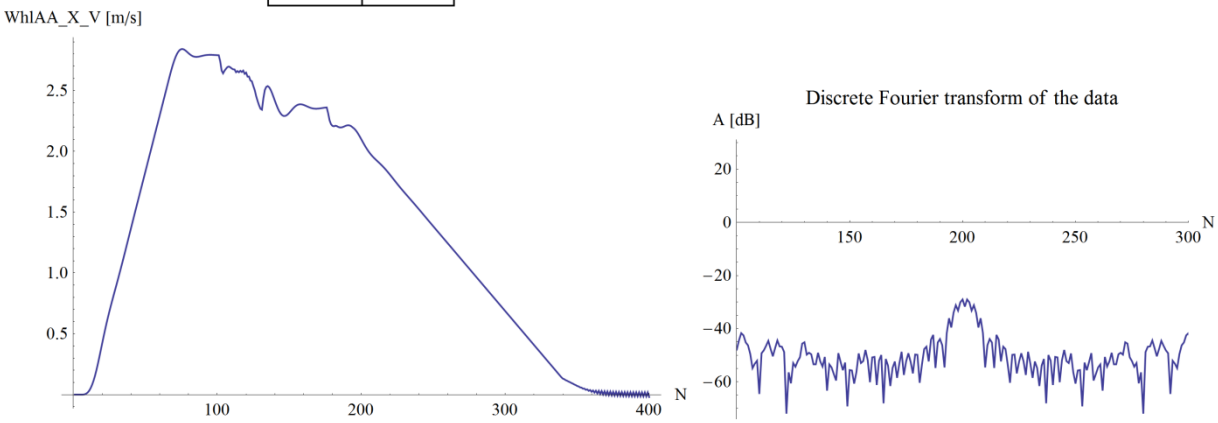


Fig. 30. Wheel velocity plot with N_{iter} = 1 and N_{sub} = 20, Δt = 0.02 [s]

By using two additional iterations, the oscillation fully disappears, even with reduced micro time step (N_{sub} = 5 instead of 20)

VS – wheel velocity	
time step	0.02
multirate	5
iterations	3
signum T	0.05

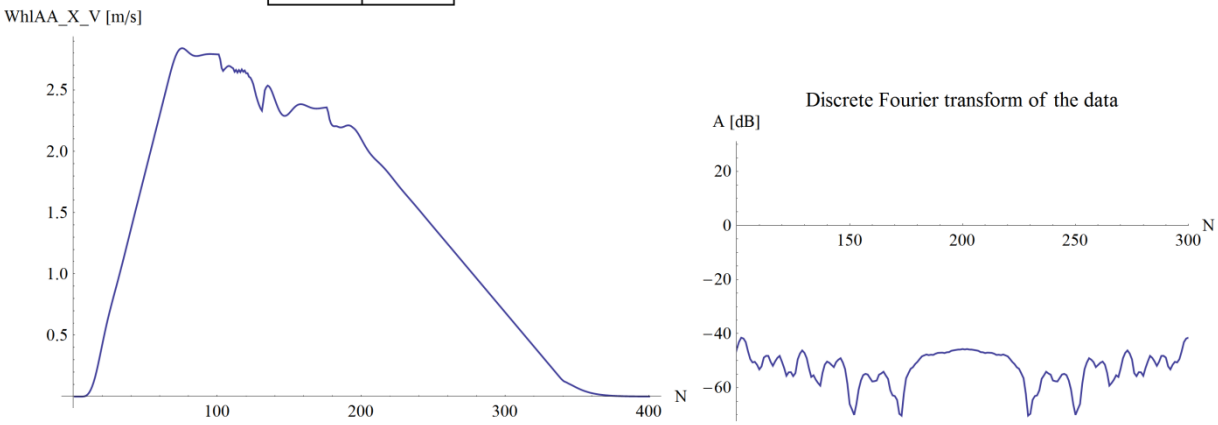


Fig. 31. Wheel velocity plot with N_{iter} = 3 and N_{sub} = 5, Δt = 0.02 [s]

Further examining the discrete Fourier transform of simulations, a stability map can be assembled. The oscillation behaviour was categorized into 5 classes according to the amplitude of the highest frequency (which can be seen in the middle of the symmetric discrete Fourier transform, at N = 200).

The stability diagrams can be seen on Fig. 32 and Fig. 33.

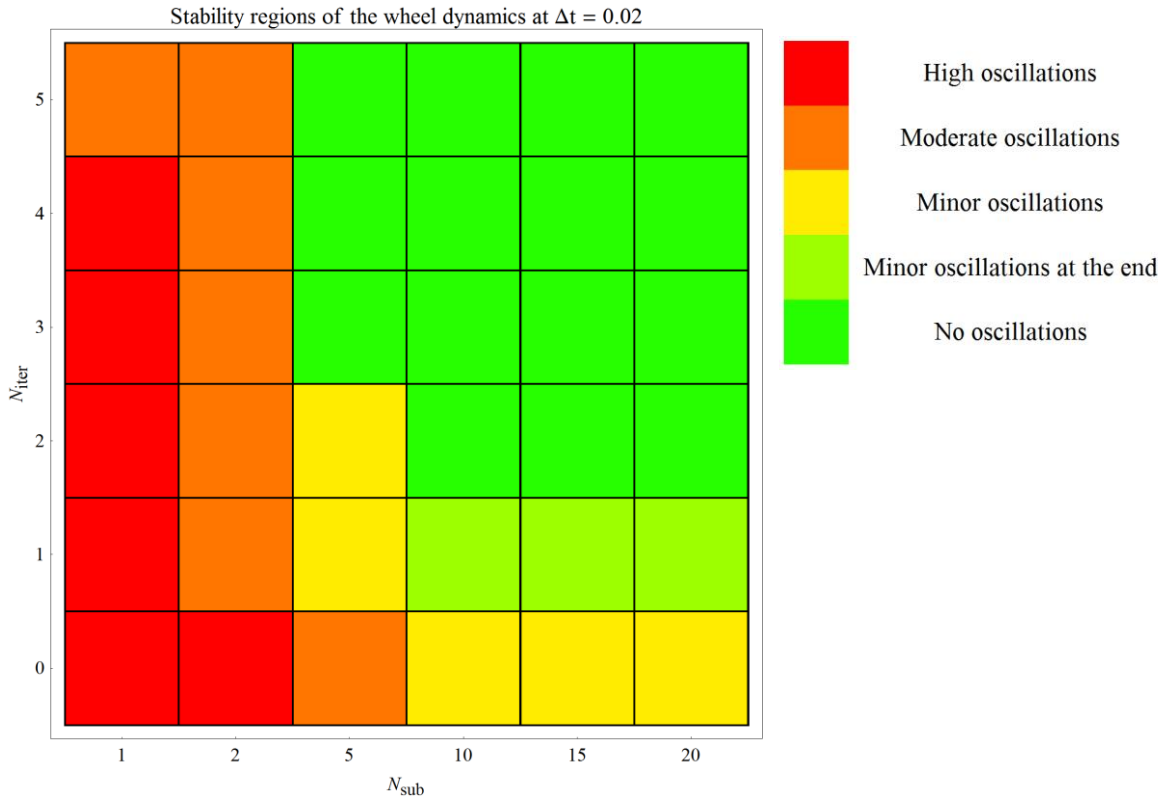


Fig. 32. Stability diagram of wheel dynamics at Δt = 0.02 [s]

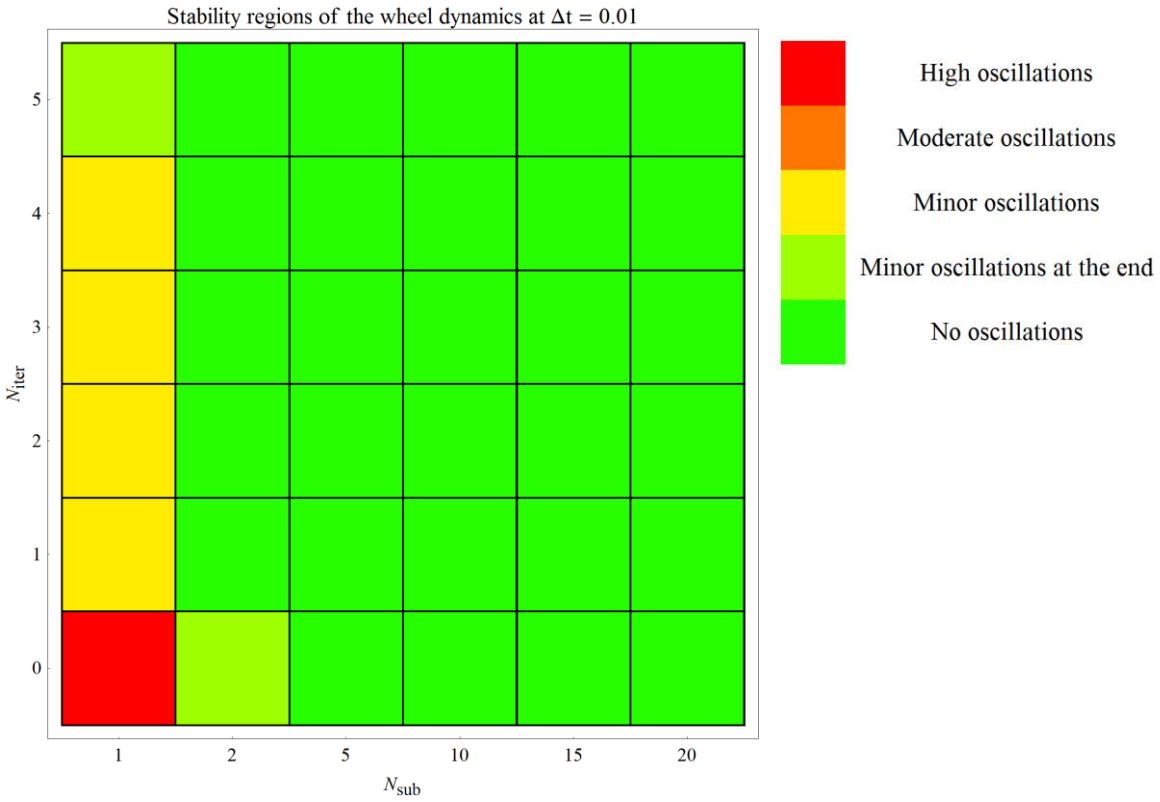


Fig. 33. Stability diagram of wheel dynamics at Δt = 0.01 [s]

It can be seen, that the oscillation of the wheel dynamics disappears as the micro time step becomes one magnitude smaller than the value of the threshold (T_{sgn}) in the continuous sign function of the rail-wheel contact model.

9.3 EFFECT OF MICRO TIME STEP AND ADDITIONAL ITERATIONS

By analysing the simulation results, different errors of velocities and positions are integrated, and displayed in contour plots. The reference value in these error calculations is the result of the best simulation configuration (Δt = 0.002 [s], N_{iter} = 5, N_{sub} = 20)

The integrated error is calculated by summing the absolute value of differences (from the reference simulation), then multiplying the sum by the time step and dividing by the length of the simulation interval.

$$Err_A = \frac{\Delta t}{T_{sim}} \sum_{i}^{N} \delta A_i. \tag{9.3.1}$$

This way, the integrated error of value A (Err_A) represents the total absolute error accumulated in 1 second of simulation.

The following contour plots show the integrated error of positions at different macro time steps.

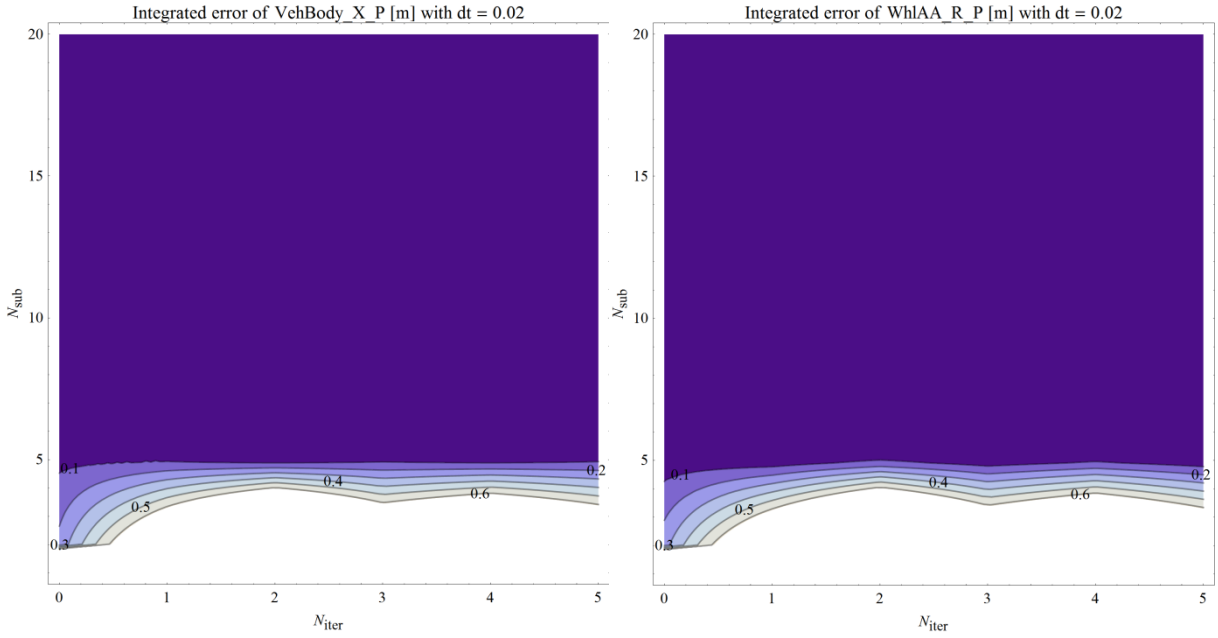


Fig. 34. Integrated error of car body and wheel position at Δt = 0.02 [s]

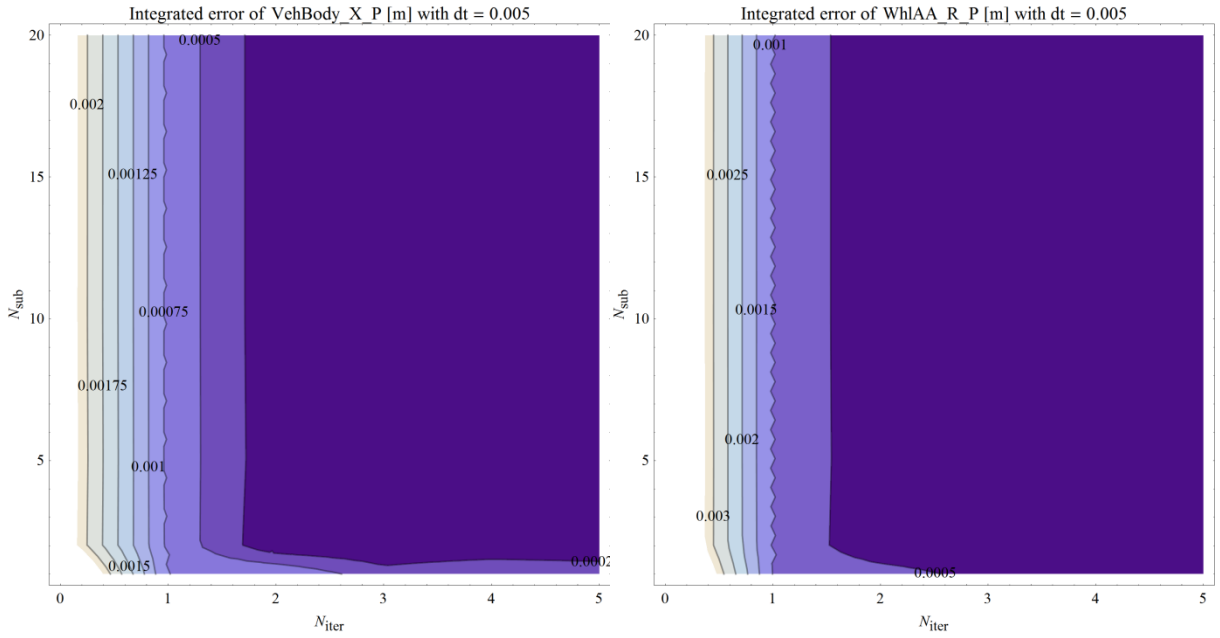


Fig. 35. Integrated error of car body and wheel position at Δt = 0.005 [s]

The following contour plots show the integrated error of velocities at different macro time steps.

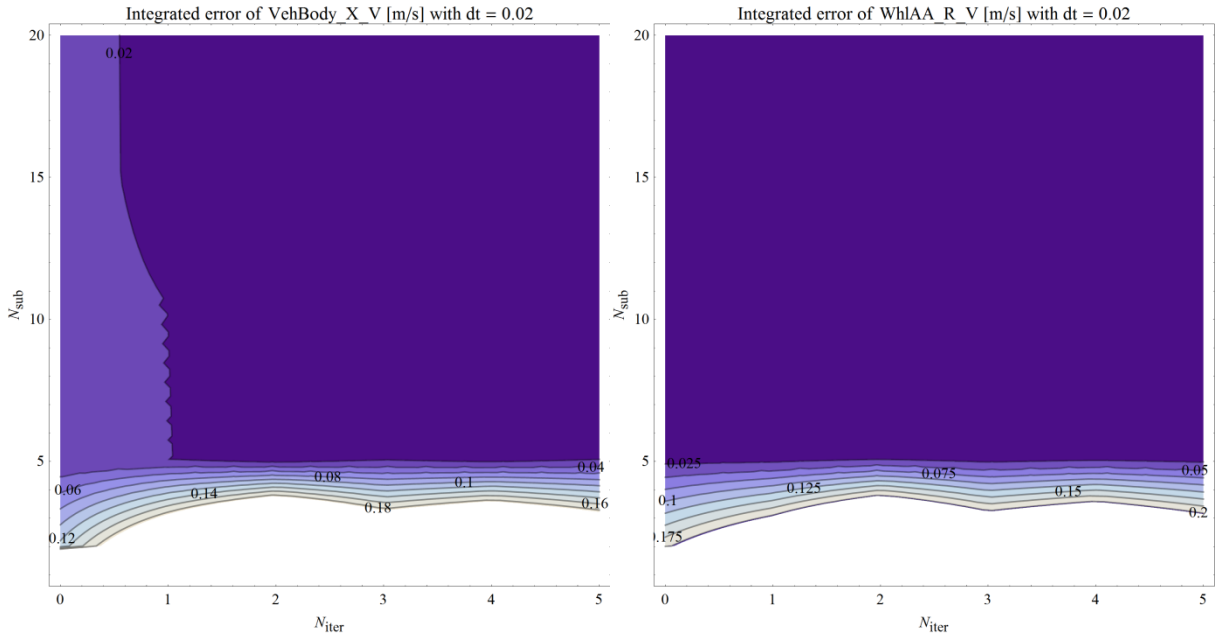


Fig. 36. Integrated error of car body and wheel velocity at Δt = 0.02 [s]

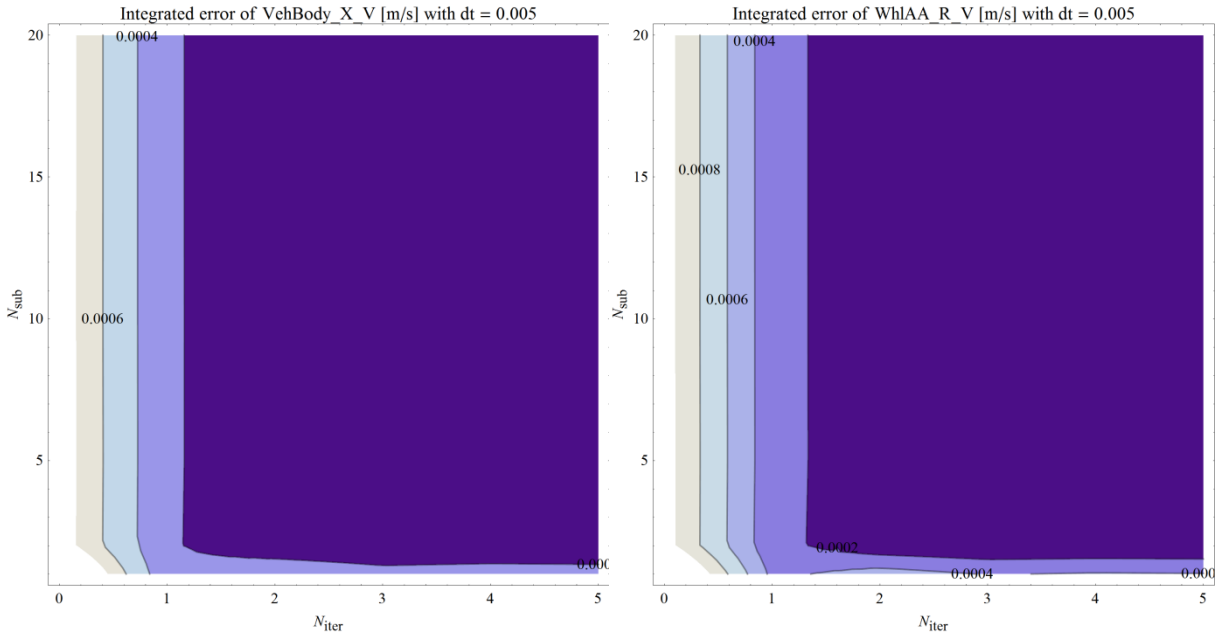


Fig. 37. Integrated error of car body and wheel velocity at Δt = 0.005 [s]

According to results, the error of different state variables mainly depends on the micro time step. At larger macro time steps (Δt = 0.02 [s]) the error gradually reduces by decreasing the micro time step (by increasing N_{sub}), thanks to the stabilization of the wheel dynamics. At smaller macro time steps (one magnitude below the threshold value T_{sgn}) the error does not depend on N_{sub}, but rather decreases by increasing the number additional iterations.

9.4 COMPUTATIONAL EFFORT

The following contour plots are illustrating the relative computational time required by the simulations. The relative computational time is the time required by the simulator to simulate 1 second of the system's behaviour (at the given macro time step, with the defined number of iterations and micro time step).

The limiting value of the relative computation time is 0.4, because currently in simulations performed at KNORR-BREMSE we have a maximum computation time of 2 ms available to calculate a 5 ms macro time step.

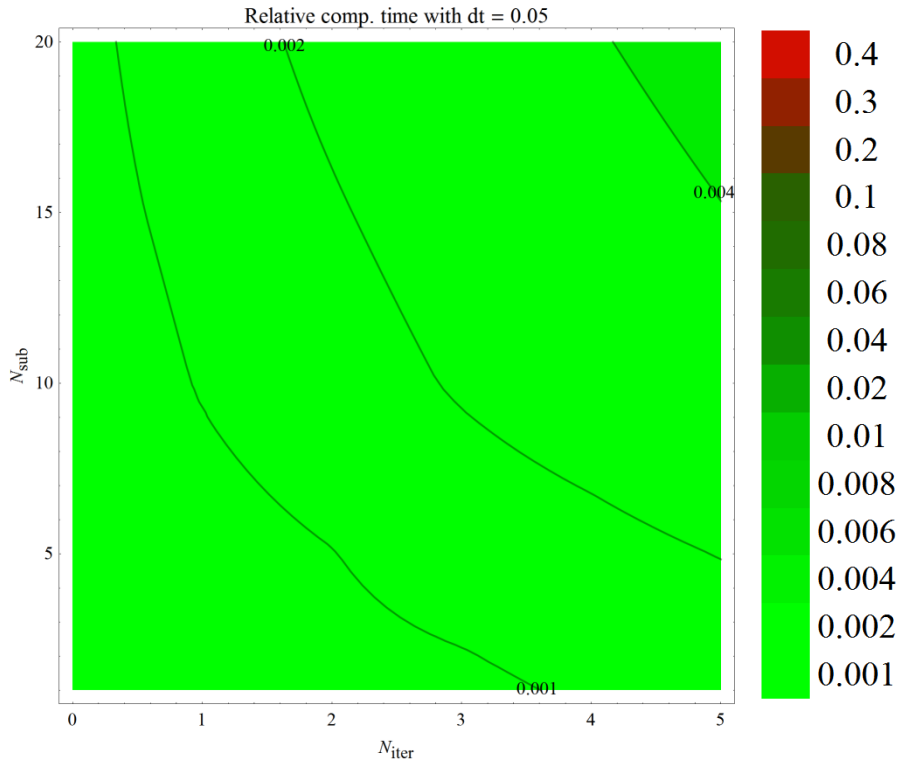


Fig. 38. Relative computational time requirement, Δt = 0.05 [s]

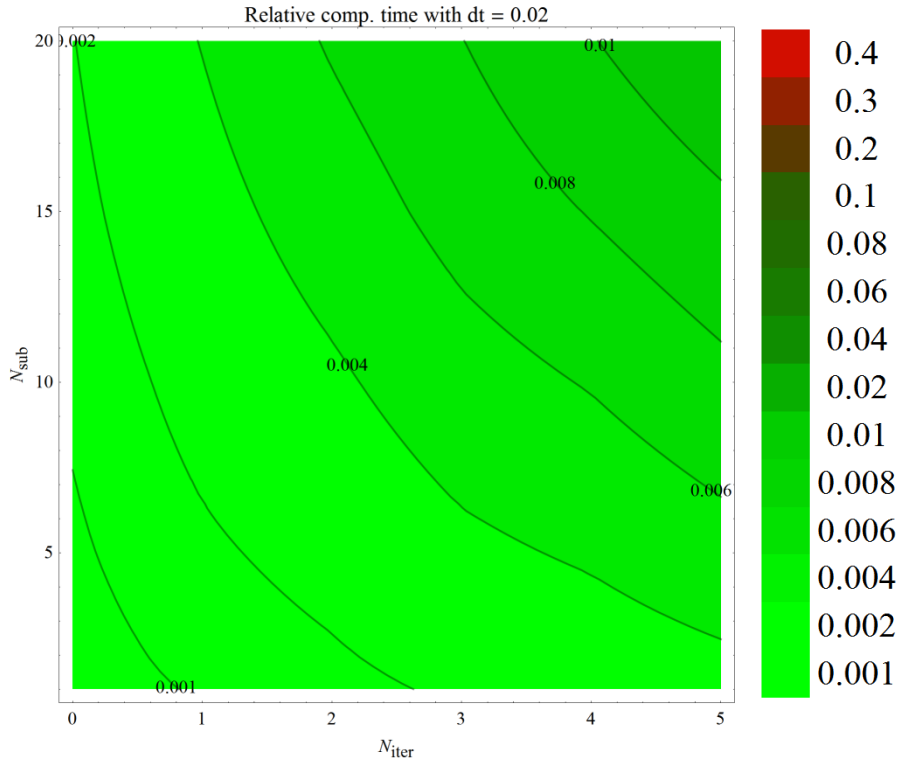


Fig. 39. Relative computational time requirement, Δt = 0.02 [s]

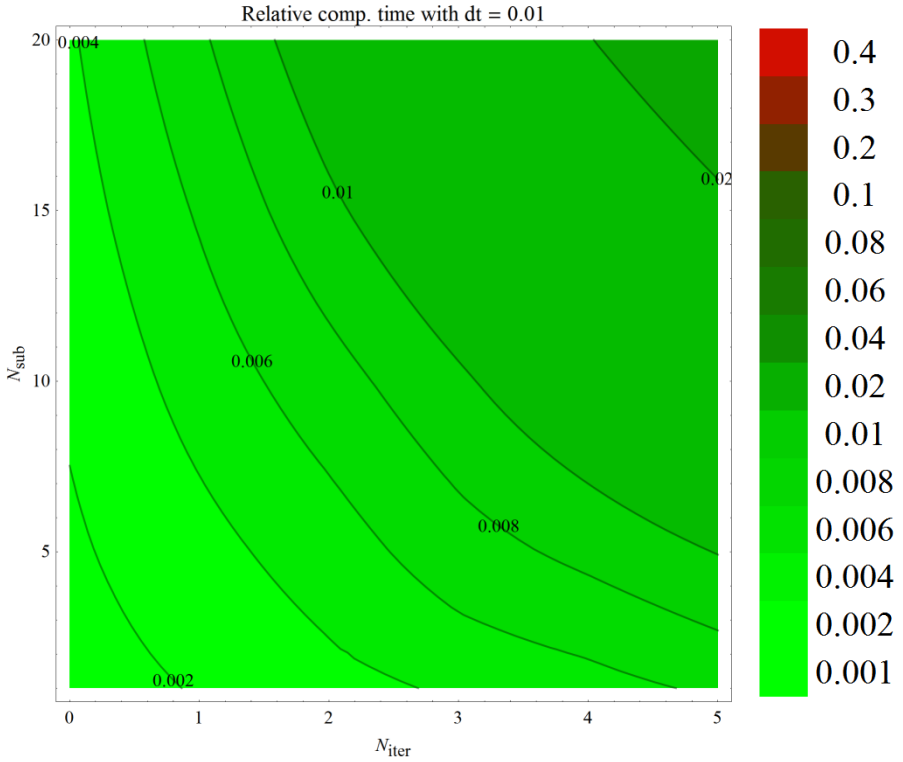


Fig. 40. Relative computational time requirement, Δt = 0.01 [s]

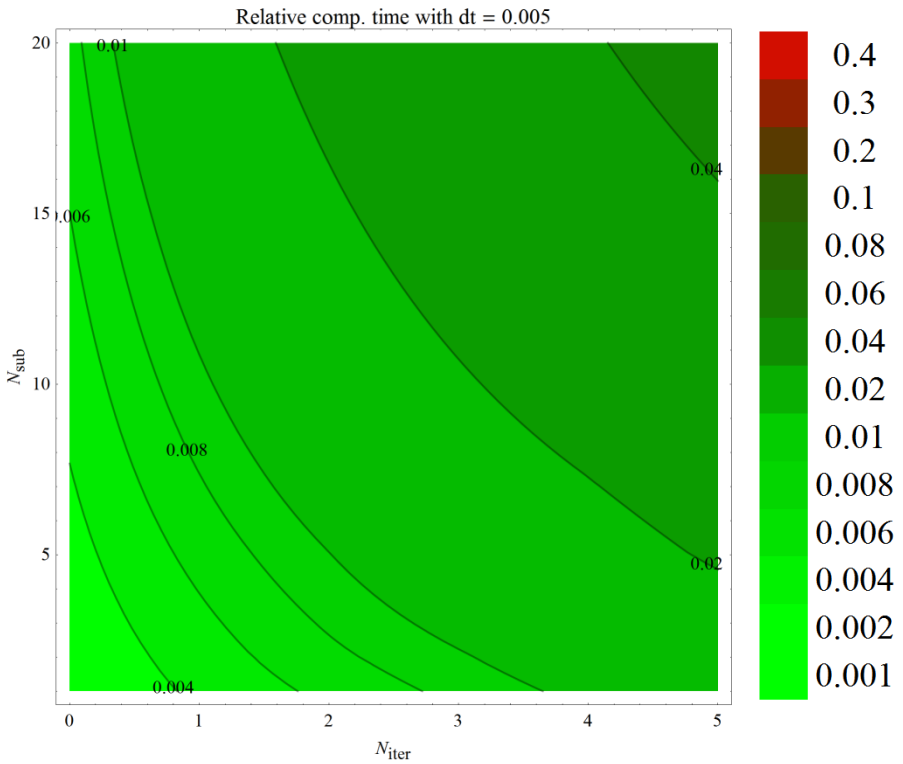


Fig. 41. Relative computational time requirement, Δt = 0.005 [s]

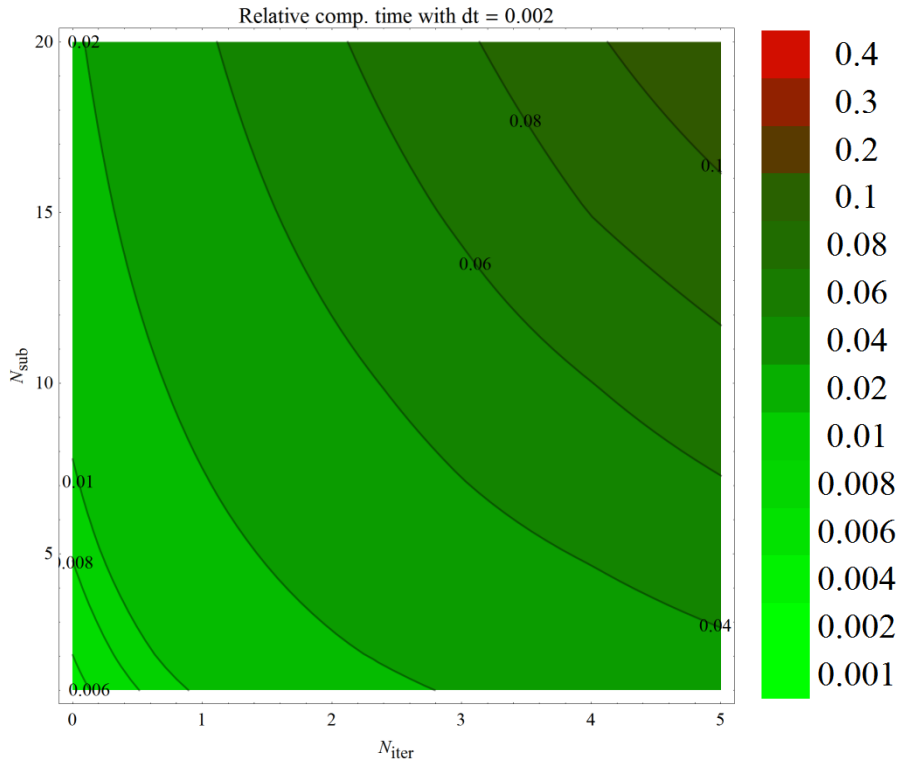


Fig. 42. Relative computational time requirement, Δt = 0.002 [s]

The contours can be approximated using the following function, which is a combination of linear ramp and hyperbolic paraboloid:

$$t_C = a\,\mathrm{N_{iter}} + b\,\mathrm{N_{sub}} + c\,\mathrm{N_{iter}}\,\mathrm{N_{sub}} + d. \qquad (9.4.1)$$

The diagrams of the approximations can be seen in the Appendix. The parameters of the relative computation time are decreasing as the macro time step increases. The step time dependence of the parameters can be approximated by an exponential function:

$$\mathrm{A}\,\Delta\mathrm{t}^{-\mathrm{B}}\,. \qquad (9.4.2)$$

The parameters of the exponential fitting on the parameters of the hyperbolic paraboloid can be found in the following table:

Parameter	A	B
a	1.03623×10^{-5}	0.97761
b	1.53271×10^{-6}	0.97764
c	1.48274×10^{-6}	1.00532
d	1.02277×10^{-5}	0.98842

It can be seen, that the parameters are nearly proportional to the reciprocal of the macro time step Δt ($B \approx -1$). Plots of these hyperbolas can be seen in the Appendix.

9.5 SUMMARY

The partitioned simulator system will be used in a Hardware-in-the-Loop system, where the main requirement is to generate wheel velocity output at a rate of 5 ms.

The initial idea was to use the partitioned simulator system with the same macro time step $\Delta t = 0.005$ [s]. However considering the previously presented results, an equivalent simulator configuration can be found for a given error tolerance.

Let us assume, that the allowed integrated error of velocities is 2 [cm/s]. In this case an equivalent simulator configuration can be chosen with

- macro time step: $\Delta t = 0.02$ [s]
- micro time step: $\Delta t = 0.002$ [s] with $N_{sub} = 10$,
- number of additional iterations: $N_{iter} = 2$,

and while producing wheel velocity output at more than two times higher rate (2 ms instead of 5 ms, allowing refined HiL coupling), this configuration offers 16% save in the computational capacity (3.65 ms instead of 4.34 ms – see equation (9.4.1) – is needed to simulate 1 second of reality).

10 CONCLUSION

In this final project I have elaborated a numerical method based on the theory of Fundamental matrices for the simulation of linear systems. This method does not introduce numerical errors during the simulation of the linear (part of the) system. All the necessary operators can be calculated analytically for real mechanical models, numerical errors are introduced only through the eigenvalue-eigenvector calculation.

Since the vehicle model is not fully linear, I have developed a simulator system consisting of two subsystems; the simulator for linear part of the vehicle and the simulator for the nonlinear wheel dynamics (including the rail-wheel contact).

The subsystems were described and the cooperation of the subsystems were shown, highlighting important configuration parameters of the framework.

The partitioned simulator system was validated, the error of state variables, stability of wheel dynamics and computation time requirement of the different simulator configurations were determined and an example on how to select equivalent simulator configurations is shown.

The most important conclusion of this work is the method of the division of a vehicle model and the fact that equivalent simulator configurations can be found, which offer either less computational time requirement or finer temporal resolution for the simulation result of the wheel dynamics for the Hardware-in-the-Loop system.

Based on this work further analysis of handling nonlinear components and application of more complex adhesion models are planned in the future.

11 ACKNOWLEDGEMENTS

I would like to thank to my supervisor at the DEPARTMENT OF APPLIED MECHANICS, DR. ZOLTÁN DOMBÓVÁRI for the valuable guidance and advices. He inspired me greatly to work on this project.

I would also like to thank to my supervisor at KNORR-BREMSE RAIL VEHICLE SYSTEMS DR. KRISZTIÁN KOVÁCS for his everyday support and the opportunity to work at the development of the rail vehicle simulator.

This final project would not have been possible without their help and motivation.

12 APPENDIX

12.1 DIGITAL VERSION OF THIS DOCUMENT

The PDF version of this document with bookmarks and clickable cross references can be downloaded by navigating to the following URL with an internet browser:

http://goo.gl/vojZp

or alternatively navigating to the same page by reading the following QR code:

12.2 TRIAL VERSION OF THE SIMULATOR

A trial version of the simulator system can be downloaded by navigating to the following URL with an internet browser:

http://goo.gl/u7YqZ

or alternatively navigating to the same page by reading the following QR code:

To run the simulator, after downloading and extracting the compressed archive, the application "VehSys+OpenGL.exe" should be executed. The simulator requires a computer (or a virtual machine) with WINDOWS operating system.

After starting the executable, a console window should indicate the preparation of the simulation:

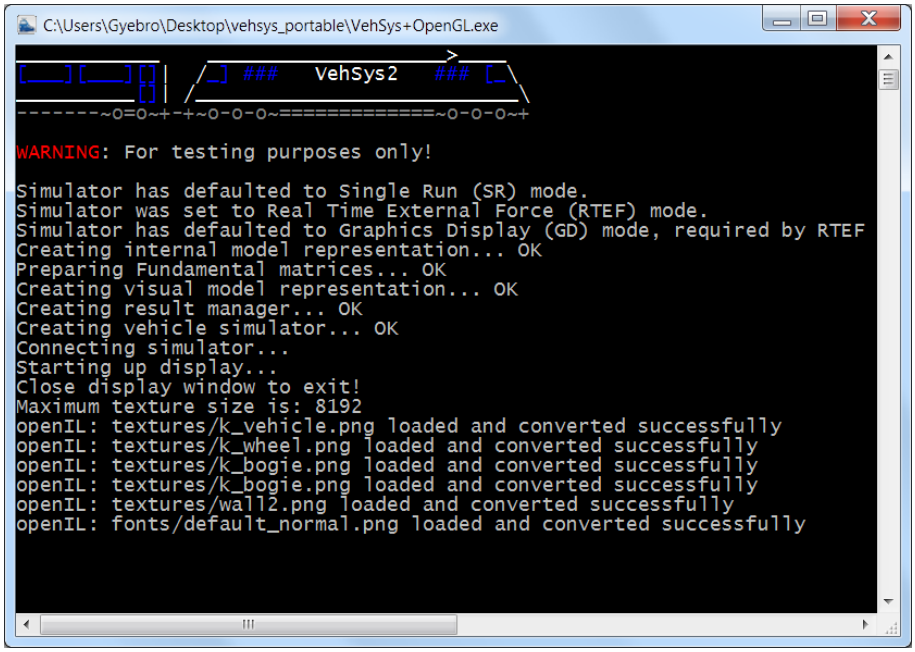


Fig. 43. Console window of the simulator

When the preparation is finished, another window should pop up displaying the 17 DoF vehicle.

Fig. 44. Display window of the simulator

The simulation runs in real time, the traction and brake forces can be controlled using the slider in the bottom right corner. The deformation scale can be controlled using the bottom slider in the bottom left corner of the window. (The time step slider is disabled.)

Display of various elements such as bodies, nodes, background, etc. can be switched in the right-click menu.

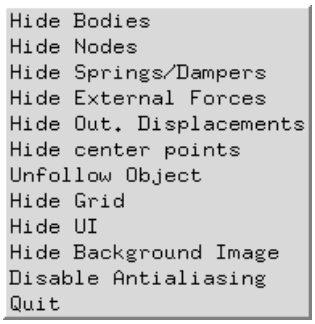


Fig. 45. Right-click menu of the display window

The displayed content can be zoomed with the mouse scroll and panned with the left mouse button. The simulation can be stopped or reset using the buttons in the top left corner.

The opacity of the textured components (bodies) can be increased or decreased by the F1 and F2 keys. After making bodies partially visible (or hiding them in the right-click menu), spring, force and external force elements become visible in the model.

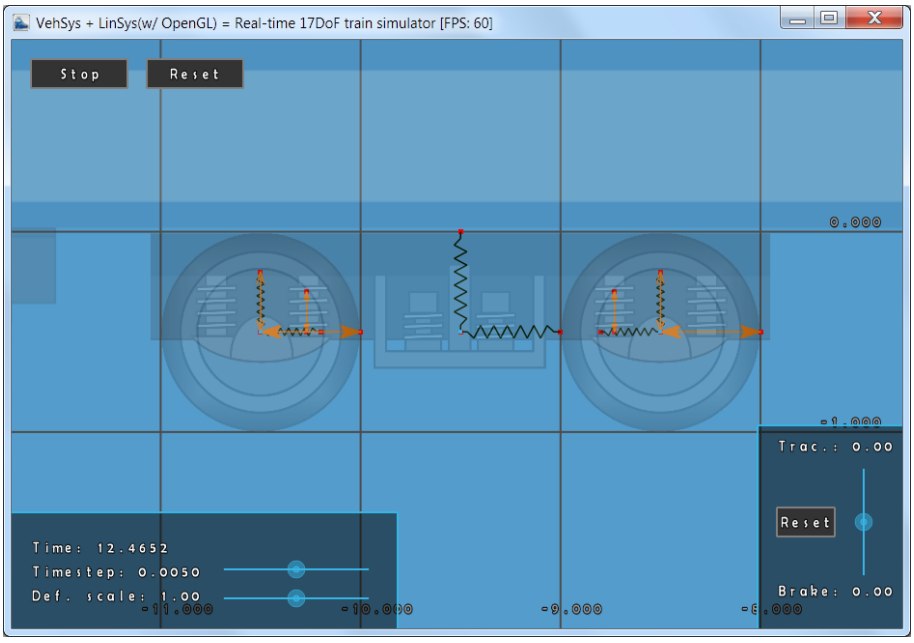


Fig. 46. Spring, force and external force elements in the model

12.3 APPROXIMATION OF THE COMPUTATION TIME REQUIREMENT

The following diagrams show the measured computational time requirement (in blue) and its approximation: $t_C = a\ \mathrm{N_{iter}} + b\ \mathrm{N_{sub}} + c\ \mathrm{N_{iter}}\ \mathrm{N_{sub}} + d$. (in red).

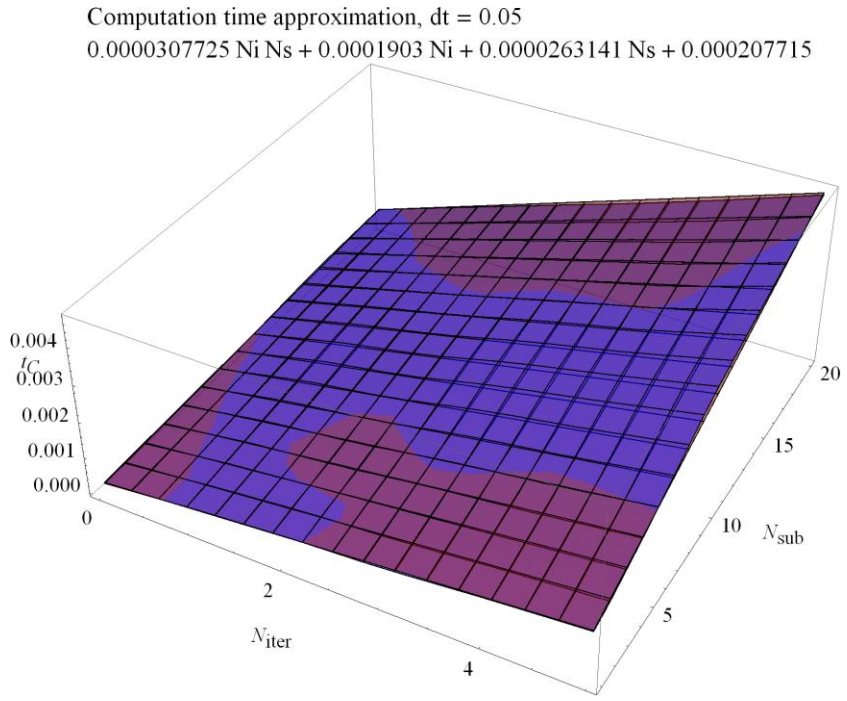


Fig. 47. Approximation of computation time requirement, Δt = 0.05 [s]

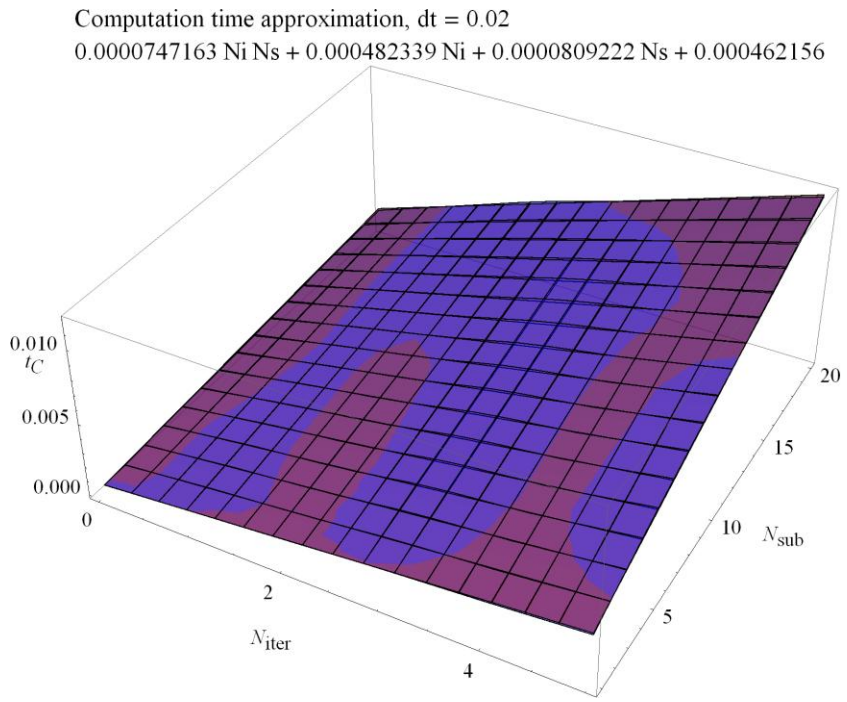


Fig. 48. Approximation of computation time requirement, Δt = 0.02 [s]

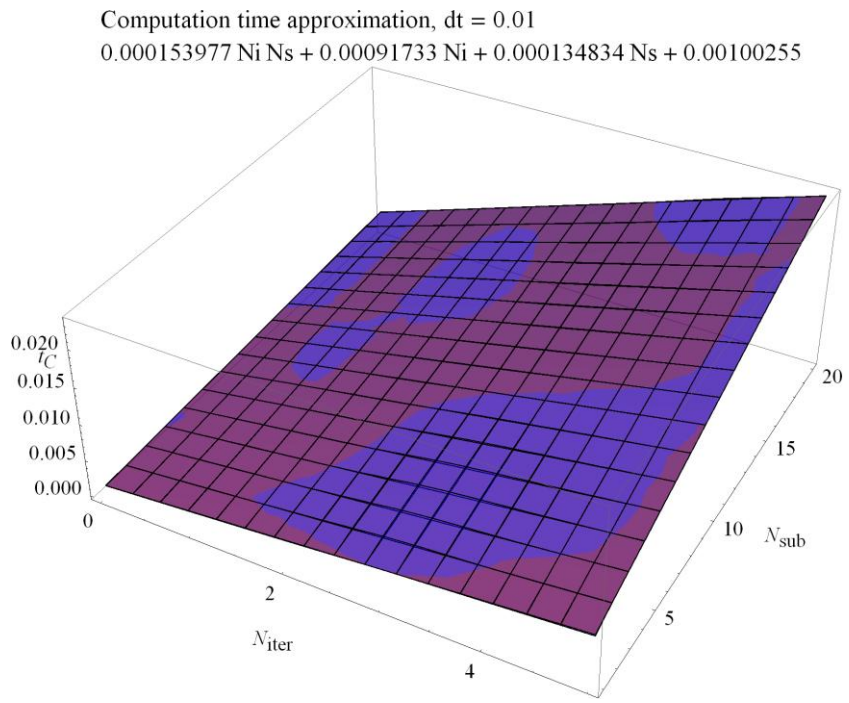


Fig. 49. Approximation of computation time requirement, Δt = 0.01 [s]

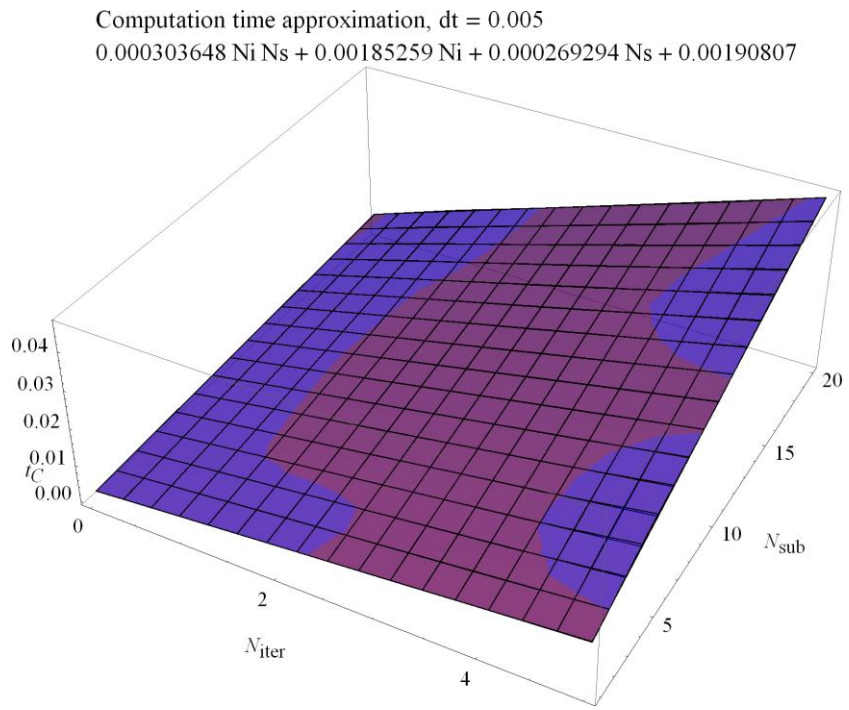


Fig. 50. Approximation of computation time requirement, Δt = 0.005 [s]

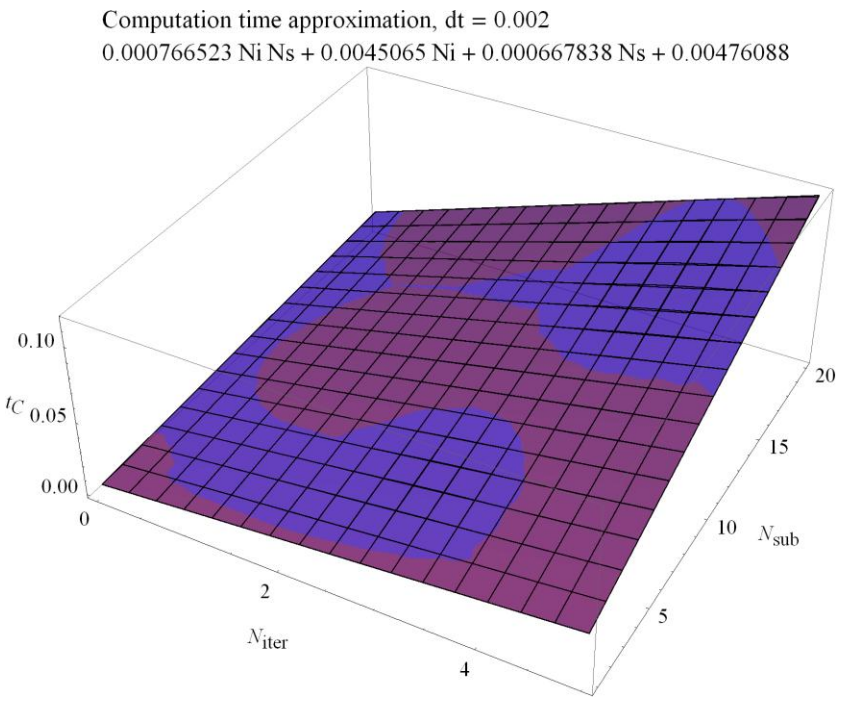


Fig. 51. Approximation of computation time requirement, Δt = 0.002 [s]

12.4 TIME STEP DEPENDENCE OF PARAMETERS

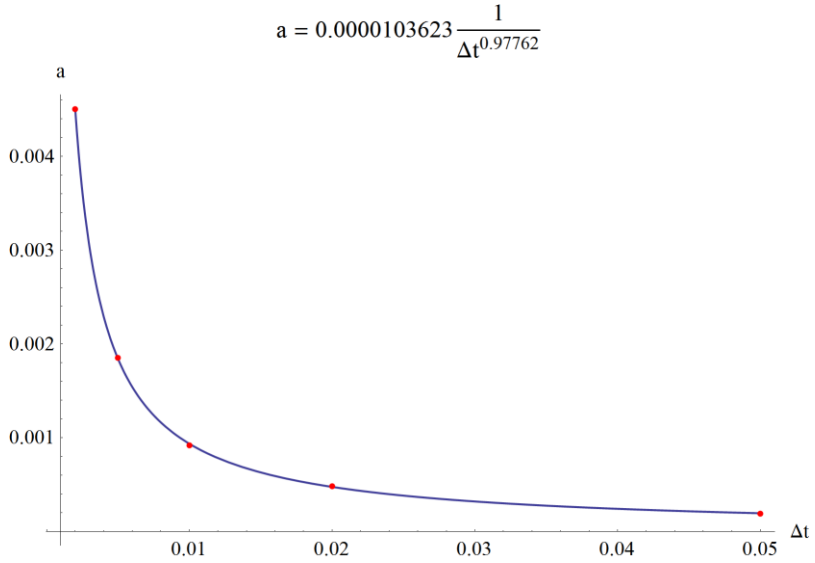


Fig. 52. Time step dependence of parameter *a* in (9.4.1)

$$b = 1.53271 \times 10^{-6} \frac{1}{\Delta t^{0.977639}}$$

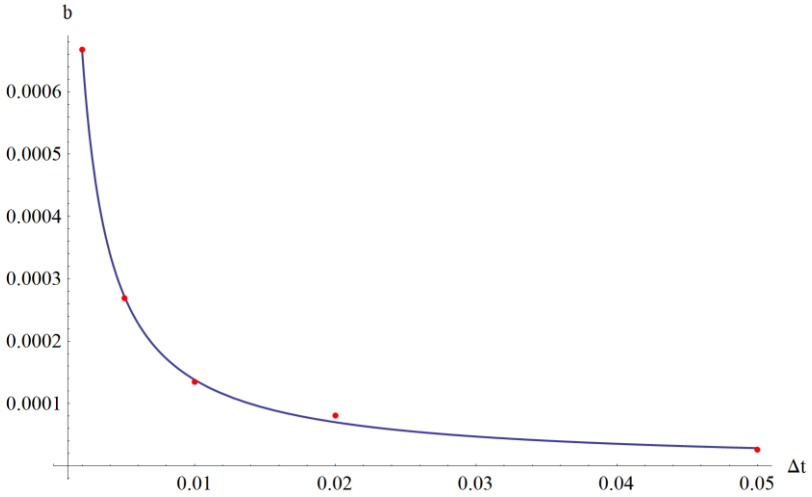


Fig. 53. Time step dependence of parameter b in (9.4.1)

$$c = 1.48274 \times 10^{-6} \frac{1}{\Delta t^{1.00532}}$$

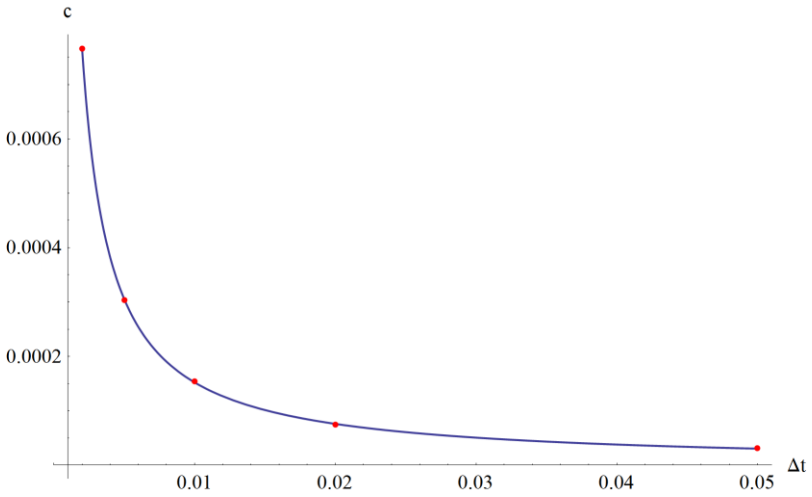


Fig. 54. Time step dependence of parameter c in (9.4.1)

$$d = 0.0000102277 \frac{1}{\Delta t^{0.988424}}$$

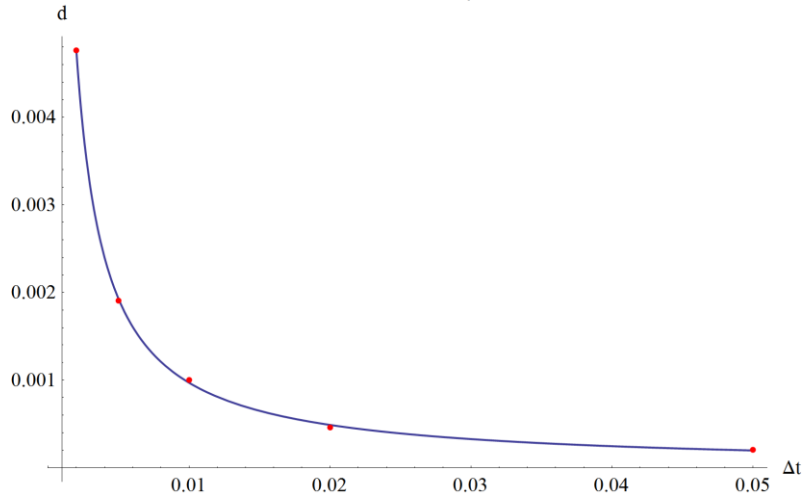


Fig. 55. Time step dependence of parameter d in (9.4.1)

REFERENCES

[1] GERGELY GYEBRÓSZKI – *Linearizált járműdinamikai modellek rendszermátrixainak automatizált előállítása*
2010 – BSc Thesis – Department of Applied Mechanics

[2] GERGELY GYEBRÓSZKI – *Járműdinamikai rendszerek hatékony valós idejű szimulációja*
2011 – Students Scientific Conference – Department of Applied Mechanics

[3] WILLIAM E. BOYCE, RICHARD C. DIPRIMA – *Elementary Differential Equations and Boundary Value Problems with Constant Coefficients*
2000 – John Wiley & Sons Inc. US, ISBN 0-471-31999-6

[4] XU-YAN CHEN – *Nonhomogeneous Linear Systems of Differential Equations with Constant Coefficients*
Georgia Institure of Technology

[5] ROLDAN POZO – *Template Numerical Toolkit Documentation*
2004 - National Institute of Standards and Technology

[6] J.I. RAMOS – *Linearized methods for ordinary differential equations*
1998 – University of Malaga – Applied Mathematics and Computation

[7] MARTIN ENQUIST – *Linear Models of Nonlinear Systems*
2005 – University of Linköping – Sweden

[8] G.K. LOWE, M.A. ZOHDY – *Modeling nonlinear systems using multiple piecewise linear equations*
2010 – Oakland University – USA – Nonlinear Analysis: Modelling and Control

[9] H. SUGIYAMA, A.A. SHABANA, M.A. OMAR – *Development of nonlinear elastic leaf spring model for multibody vehicle systems*
2005 – University of Illinois – Chicago – Computer methods in applied mechanics and engineering

[10] H. SAYYAADI, N. SHOKOUHI – *A new model in rail-vehicles dynamics considering nonlinear susbension components behaviour*
2009 – Sharif University of Technology, Tehran, Iran – International Journal of Mechanical Sciences

[11] M. GÜNTHER, A. KVÆRNØ – *Multirate partitioned Runge-Kutta methods*
2001 – BIT Vol. 41

[12] CH. ENGSTLER, CH. LUBICH – *Multirate extrapolation methods for differential equations with different time scales*
1995 – Computing – Springer-Verlag

[13] A. KVÆRNØ – *Stability of multirate Runge-Kutta schemes*
Norwegian University of Science and Technology
[14] CH. ENGSTLER, CH. LUBICH – *MUR8: a multirate extension of the eighth-order Dormand-Prince method*
1997 – Applied Numerical Mathematics
[15] A. BARTEL, M. GÜNTHER – *A multirate W-method for electrical networks in state-space formulation*
2002 – Journal of Computation and Applied Mathematics

LIST OF FIGURES