Implementing a GPU-based numerical algorithm for modelling dynamics of a high-speed train

Syto E.S.¹ and Bratus A.S.¹ and Yurchenko D.²

¹Moscow institute of Information and Transport, Moscow, Russia.
²Institute of Mechanical, Process and Energy Engineering, Heriot-Watt University, Edinburgh, UK

Abstract

The paper discusses the initiative of implementing a GPU based numerical algorithm for studying various phenomena associated with dynamics of a high-speed railway transport. The proposed numerical algorithm for calculating a critical speed of the bogie is based on the first Lyapunov number. Numerical algorithm is validated by analytical results, derived for a simple model. A dynamic model of a carriage connected to a new dual-wheelset flexible bogie is studied for linear and dry friction damping. Numerical results obtained by CPU, MPU and GPU approaches are compared and appropriateness of these methods is discussed.

Keywords: Limit cycle, Bifurcation, Vibrations, GPU computing.

1 Introduction

The high-speed railway transport (HSRT) can provide a reasonable alternative to air travel. In fact the HSRT not only economically viable at distances of 500-800km, but also environmentally friendly, producing less noise and carbon emission, which aligns well with the latest EU guidelines. Train motion presents a complex dynamical process of the natural interaction between a train, track and catenary. To stay lucrative and competitive railways all over the world move towards increasing train speed and axle weight. Apart from the 300km/h Channel Tunnel Railway Line the maximum speed of trains in the UK is generally 200km/h. In France the TGV now operates at 320km/h, but a land speed record of 574km/h was reported on the Paris to Strasbourg line. It is safe to say that with the speed or/and axle load increase the interaction between train and track intensifies, larger forces transmitted onto the track from the train and back.
To predict behaviour of a HS train it is essential to properly model the train components, the track properties, the aerodynamics and catenary and the interaction of all these components at various speeds. Adequate modelling, concatenating various train modelling components, assists in catching a variety of effects and phenomena, such as hunting, derailment, voids, flats, wear, and others [1]. This becomes an extremely time demanding and challenging problem with a massive number of degrees-of-freedom (DOF). Indeed, a number of DOFs of a train-bogie-wheelset model has been increased from 2 to 31 in last 50 years [2-9]. Whereas in studying train dynamics the focus is usually on safety (hunting instability, derailment) and ride comfort, the track property investigations are focused on predicting the track degradation under the train load (gauge, cant, etc.), the waves propagation within the soil and their influence onto the nearby buildings and structures, as well as influence of the track properties onto the train motion [10-17]. These models may take from a finite (lumped mass models) to an infinite number of DOF (continuous models), which apparently, when discretize, may have tens and hundreds of thousands of DOFs. Finally, the track and train interaction happens due to the contact phenomenon between wheels and rails.

In general the resulted (creep) forces of this interaction can be found by solving a contact problem between two continuous media. Since these forces have to be identified on each and every time step it becomes inappropriate for studying the train dynamics due to enormous computational efforts required for solving the contact interaction. Besides, this problem should be solved for a given number of wheels, which is usually 2 or 4 per carriage. Some significant developments in understanding the contact phenomenon have been reported [18-20] and a simplified theory for estimating the creep forces has been developed. Despite these significant contributions and later improvements [21,22] calculating the wheel-rail interaction still remains an extremely expensive computational procedure.

There have been efforts to build a whole model of a moving and interacting with environment train, but apparently contemporary computational capabilities simply cannot handle the problem of such complexity, even when some simplifications in the contact problem and track properties are accepted. On the other hand, Graphics Processing Unit (GPU) based computing allows handling large size problems due to the fact that it hosts thousands of cores, whereas modern workstations may have tens of cores. However, one can benefit from GPU-based computing only when the computing can be done in parallel so that the communication between cores is minimal, as well as communication between GPU and CPU, which involves reading/writing data. GPU-based computing has not been as popular as it should have
been since it requires a different coding methodology due to its stream processing feature and knowledge of special programming languages such as CUDA or OpenAcc is essential. GPU computing has become very popular in the area of fluid dynamics, where various methods are used to implement GPU computing, such as DEM, SPH and others [23-25]. Some commercial software packages introduce GPU-based computing for accelerating numerical calculations, for instance Universal Mechanism. Recently, it has been reported that a GPU-based computational approach may speed up calculations up to 30 times compared to CPU-based approach [26].

It should be mentioned that besides the above mentioned needs, often a system of Ordinary Differential Equations (ODEs) describing a complete carriage model has to be studied numerous times either to get some statistical pattern under a random excitation or to complete the optimization procedure. This paper describes the first step towards a GPU-based numerical approach for studying the train vibrations and train-track interaction. However, instead of studying a model of a train consisting of several carriages with a number of the wheel-rail interactions, the paper implements the GPU, MPU and CPU algorithms for a single carriage with two dual-wheelset bogies and multiple values of the system’s parameters. This leads to tens and hundreds of thousands of equations that are solved simultaneously, and the advantages of different numerical approaches are discussed.

2 Validation of the GPU algorithm

First a GPU architecture should be validated and for that a simple train model will be used. There are several phenomena related to a dynamics of a train, and one of them is hunting, which is directly related to Andronov-Hopf (AH) bifurcation [27]. Although the bifurcation diagram can be constructed numerically and the critical speed can be estimated [28-30], properly it should be done by calculating the first Lyapunov coefficient, according to the theorem, presented in Appendix A. The theorem basically implies that any general dynamic system with a given bifurcation parameter behaves itself in the vicinity of the equilibrium as a corresponding equivalent system with linear and cubic terms, derived from the original system. The stability loss of the system is associated with the signum of the first Lyapunov coefficient that defines the signum of the cubic term [27]. The algorithm for calculating the Lyapunov coefficient is provided in Appendix A.

The architecture of a software system has been formulated, bearing in mind that the programme should be flexible and work with a variety of models. A plugin structure of the code, when each model is connected as a dynamic library, has been
implemented. The code comprises a number of solvers which separately solves a set of ODEs, performs various calculations and plot the results.

To use CUDA language the CUDA SDK package has been used. This package allows writing a code in the simplified C version instead of CUDA. A parallel algorithm library THRUST has been used, which automatically identifies the optimal configuration for computing a specific problem. Runge-Kutta integration methods has been implemented for solving a set of ODEs with the Rosenbroke method for stiff systems [31]. The optimization package has implemented the Nelder-Mead method and the proposed GPU-based approach allows calculating the objective function in parallel in all points of simplex, significantly reducing the amount of computational time.

Figure 1 demonstrates the implementation of a parallel algorithm for calculating a train dynamics for various set of input parameters $\psi_i$ and track properties depending on the train speed $\eta(v_i)$, where “MM” stands for “Mathematical Model”. Ride comfort is calculated then as a function of a train speed for different set of parameters and can be compared later against that obtained for another set.

Figure 1: GPU architecture for a parallel implementation of a train evolution
The proposed GPU-based approach is validated on the problem from [4], where a single wheelset bogie has been considered.

\[ m\ddot{y} + 2 \frac{f_{11}}{V} \left[ (1 + r_0 \frac{\dot{\lambda}}{d}) \dot{y} - V \psi \right] + \frac{2f_{12}}{V} \dot{\psi} + W_A \frac{\dot{\lambda}}{d} y = F_{sy} - F_T \]  

(1)

\[ l_{wx} \ddot{\psi} + l_{wy} \frac{V}{r_0 d} \dot{\lambda} \dot{y} + \frac{2df_{33} \dot{\lambda}}{r_0} y - \frac{2f_{12}}{V} \left[ \left(1 + r_0 \frac{\dot{\lambda}}{d}\right) \dot{y} - V \psi \right] + 2d^2 f_{33} \frac{\dot{\psi}}{V} - dW_A \frac{\dot{\lambda} \psi}{\psi} + 2f_{22} \frac{\dot{\psi}}{V} = M_{sz} - 2bF_d \]  

(2)

where:

\[ F_{sy} = -2K_y y - 2C_y \dot{y}, M_{sz} = -2K_\psi b^2 \psi \]  

(3)

and nonlinear damping has the form:

\[ F_d = \begin{cases} C_1 V_{\psi} + C_2 V_{\psi}^2 + C_3 V_{\psi}^3 + C_4 V_{\psi}^4, & V_{\psi} > 0 \\ C_1 V_{\psi} - C_2 V_{\psi}^2 + C_3 V_{\psi}^3 - C_4 V_{\psi}^4, & V_{\psi} < 0 \end{cases} \]  

(4)

gde \( V_{\psi} = b \dot{\psi} \). However, this function does not have the second derivative at zero, which prevents one from studying its influence on the Lyapunov coefficient. Therefore, this function is approximated by an odd power polynomial:

\[ F_d = K_1 V_{\psi} + K_2 V_{\psi}^3 + K_3 V_{\psi}^5 \]  

(5)

Having examined the state-space portraits one can conclude that these systems behave very similar. Also, the above model takes into account the contact of the wheel with the rail:

\[ F_T = \begin{cases} K_\tau (y - \delta), & y > \delta \\ 0, & -\delta \leq y \leq \delta \\ -K_\tau (y - \delta), & y < -\delta \end{cases} \]  

(6)

The values of all the system’s parameters can be found in Appendix B.

The system has been solved numerically for \( V=48 \text{m/s} \) and \( V=62 \text{m/s} \) and a stable focus has been replaced by a stable limit cycle, which demonstrates the supercritical AH bifurcation.

It has been found that the bifurcation happens when \( V=53.9 \text{m/s} \). Numerical results have shown that the first Lyapunov coefficient is \( l_1(0) = -0.032128 \). For the considered model, which is a relatively simple one, it is possible to calculate analytically this number to validate the code. With this purpose let’s rewrite (2) and (3) in the form:
\[
\begin{align*}
\begin{cases}
    \dot{x}_2 &= \left[-W_A \frac{\ddot{d}}{d} x_1 - 2 \frac{f_{11}}{V} \left([1 + r_0 \frac{\ddot{d}}{d}] x_2 - V \varphi_1\right) - \frac{2f_{12}}{V} \varphi_2 + F_{sy} - F_T\right]/m_\omega \\
    \dot{\varphi}_1 &= \varphi_2 \\
    \dot{\varphi}_2 &= \frac{\frac{-2a^2 f_{33} \frac{\ddot{d}}{d} x_1 - l_{wyz} d x_2 + \frac{2f_{12}}{V} \left([1 + r_0 \frac{\ddot{d}}{d}] x_2 - V \varphi_1\right)}{l_{wz}} + \frac{dW_A \frac{\ddot{d}}{d} x_1 - l_{wyz} d x_2 + \frac{2f_{12}}{V} \left([1 + r_0 \frac{\ddot{d}}{d}] x_2 - V \varphi_1\right)}{l_{wz}}}{l_{wz}}
\end{cases}
\end{align*}
\]

Since the nonlinearity is present here in the 4th line, then B and C functions (see Appendix A) will have nonzero components in the 4th line. Besides, it is obvious that the second derivative of \(F_d\) is always equal to zero. Thus, one arrives to the following expressions:

\[
B(x, y) = \begin{pmatrix}
0 \\
0 \\
0
\end{pmatrix} \quad C(x, y, z) = \begin{pmatrix}
0 \\
0 \\
0 \\
-\frac{12K_2 b^4}{l_{wz}} x_4 y_4 z_4
\end{pmatrix}
\]

Consequently \(l_1(0) = \frac{1}{2\omega_0} Re(\langle p, C(q, q, \bar{q}) \rangle)\). One can calculate the eigenvalues \((i\omega_0)\) and eigenvectors \((q_i, p_i)\) at the critical speed and substitute them into (8) arriving to \(l_1(0) = -0.032024\). This result is very close to the value obtained numerically and its negative signum indicates a stable limit cycle. It should be mentioned that the critical speed is rather high, despite the fact that the linear creep model has been used, which is suitable for a relatively small speed values. Nevertheless, the presented model was simple enough to demonstrate the methodology for investigating the stability of a limit cycle and validating the GPU-based approach.

### 3 A new bogie testing and GPU implementation

The code is used to investigate a carriage with two bogies consisting of a dual wheelset each and depicted in Figure 2 and Figure 3 [32]. This system has 27 DOFs, parameters of which are shown in Appendix B. These figures show the linear model with linear springs and viscous damping. A standard model of the creep forces have been adapted in this study, however to model the side wheel-rail contact, occurring due to a lateral displacement of the bogies, a cubic nonlinearity has been introduced:

\[
F(x) = \begin{cases}
0, & x < X_{cr} \\
x a + bx^3, & x < X_{cr}
\end{cases}
\]
Figure 2: Side view of the two dual-wheelset model

Figure 3: Top view of the two dual-wheelset model
This is more realistic since it corresponds to the soft-impact model of the interaction between the wheel and rail [33]. Moreover, the comparison of results between the models with viscous ($\alpha \dot{x}$) and dry friction $R_{\text{sign}}[\dot{z}]$ is performed. Dry friction has been substituted instead of viscous friction between the carriage and bogies in lateral and longitudinal directions, with the nominal value provided in Table B4 of Appendix B. In this case $\dot{z}$ represents a relative velocity between two connecting bodies with no “stick” mode. Values of $R$ have been taken based on the equivalence of the mean energy dissipation over a period $R_o$ and divided by $R=R_o/3$. Some numerical values used for simulations are presented in Appendix B. It should be mentioned that the dry friction with value of $R_o$ dissipates energy much stronger than that of viscous damping.

To generate data for multiple values of parameters the GPU strategy, implemented in this study, is shown in Figure 4. At this point the code calculates the entire train/track interaction problem for a number of input data sets $I_1, I_2, \ldots I_M$. The blocks surrounded by blue colour are executed in parallel, progressing forward in time using Runge-Kutta method with $dt=0.01s$ until the final time instant $T=24$ seconds. As a result of these calculations each set inside a blue block generates a corresponding set of results $R_1, R_2, \ldots R_M$, which is later analysed. The system has been excited by the rail surface roughness using the following formula for modelling auto and cross correlation of roughness of both rails in vertical and horizontal directions

$$R(x) = S^2 \sum_k a_k \exp(-\alpha_k^2 x^2) \cos(2\pi\beta_k x)$$  \hspace{3cm} (10)

This formula approximates the experimental data provided by Russian State Railway company, and the values of parameters in expression (10) are provided in Table C1.

Figure 4: GPU architecture for a parallel implementation of a train evolution

The implemented algorithm have produced a number of results, presented below. Figures 5 shows the results of numerical modelling for lateral and vertical
displacements and the amplitude of the pitching angle of the carriage vs time at 20m/s. It can be seen that the nominal value of the dry friction damp the vibrations no better or at the same level as the viscous one in all the presented degrees-of-freedom.

Figure 5: Lateral displacement (top), vertical displacement (middle), pitch angle of the carriage (bottom) at v=20m/s with viscous (red) and dry (blue) friction
Figure 6: Lateral displacement of the wheelset vs time at v=20m/s (top), v=60m/s (middle) and v=100m/s (bottom) with viscous (red) and dry (blue) friction.

Figure 6 presents results of the lateral displacement of the wheelset at 20m/s, 60m/s and 100m/s top to bottom correspondingly. In top figure one can see a limit cycle oscillations for dry friction case and the amplitude of the response is relatively small, preventing the wheelset hitting the rails. With increase of the speed to 60 m/s the wheelset experiences side impacts against the rails quite regularly (see the peaks and troughs of the responses) and not much difference is observed between two types of damping. Somewhat like beating phenomenon is observed in the case of viscous damping (red line) with the main period of about 5 seconds. In Figure 6 bottom one can see the lateral vibrations of wheelset at 100m/s and the response seems to be more irregular with multiple impacts against the rails.
Figure 7 left presents the carriage amplitudes as a function of train velocity for viscous and dry friction. It is seen that the dry friction provides worse vibration mitigation alternative at low train speeds and better alternative at speed of 80m/s and higher for the considered model. Figure 7 right demonstrates similar graphs for the lateral vibrations of the wheelset. Dry friction attenuates vibrations slightly worse than viscous damping, however this difference becomes less apparent at high speeds.

Figure 7: Amplitude of the carriage (left) and wheelset (right) lateral oscillations with viscous (red) and dry (blue) friction.

At this point it is important to compare three different approaches: regular CPU computing, parallel or multiprocessing CPU computing (GPU MPC) and GPU computing. All the computations were performed on Intel Core i5-5200U, NVIDIA GeForce 920M card and the results are presented in Figure 8 and Figure 9. As it had been expected the amount of time grows very rapidly with the increase of the number of equations. However, for low dimensional systems (Figure 8 left) both CPU and parallel CPU computing are much more efficient than the GPU approach. This observation can be explained by the following reasoning. The overall processing time comprises time required to upload/download data into/from GPU memory, communicate with CPU (I/O-time), and time required for performing actual calculations (P-time). Apparently in this case, when the number of equations is small and there is enough GPU memory to upload all the data into it, I/O-time dominates the overall time.
Figure 8: Time (ms) required to solve the system of N-equations using CPU (red), CPU-MPC (blue) and GPU (black) approaches.

Thus, when the number of equations is in the order of a couple of thousands there are no advantages in engaging GPU technique. However when the number of equations is close to several thousand CPU-MPC and GPU approaches become a faster alternative to CPU computing, although GPU still takes longer to perform the task. When the number of equations is in order of $10^5$ GPU and CPU-MPC demonstrate the equivalent performance, which can be seen in Figure 9 left.

Figure 9: Time (ms) required to solve the system of N-equations (X-axis) using CPU (red), CPU-MPC (blue) and GPU (black) approaches.

Further increase in the number of equations demonstrates the superiority of the GPU-based approach, which is shown in Figure 9 right. Apparently in this case P-time dominates I/O time and since GPU approach performs calculations faster overall it become superior in completing such highly dimensional tasks. One can see that the GPU-based approach takes almost 6 times less P-time than CPU one and halve CPU-MPC time.
It should be mentioned that in these tests the amount of memory required by the data was always below the available GPU memory. This trend, also observed in [26], will change when GPU memory is not sufficient to keep all the data and communication with CPU, which may currently be considered as a bottle neck, is inevitable.

4 Conclusions

The paper proposes a GPU-based numerical algorithm for studying vibrations of high-speed train and calculating the critical speed of a bogie based on the first Lyapunov coefficient. It has been shown that the proposed GPU-based approach is much more efficient than CPU and multicore CPU (MPC) when a large set of equations (over $10^5$) has to be solved. The algorithm has been validated on a model, adapted from [4], and demonstrated on a new model with two dual-wheelset flexible bogie. The cubic type nonlinearity has been employed to simulate side impact of the wheel against the rail. Effect of two different types of damping (viscous and dry friction) has been studied and indicated that the behaviour of the system with a properly selected dry friction coefficient is almost similar to one with viscous damping in terms of displacements, although the pattern of motion is different. Moreover, the viscous damping found to be more appropriate at low values of the train speed, whereas the dry friction proved to be better alternative at high values of the train speed.

Acknowledgement

This research work has been supported by the Ministry of Education and Science of the Russian Federation from 24.11.2014 №14.607.21.0091.

References


Appendix A. Algorithm for calculating the first Lyapunov number

**Theorem.** (Normal form of the AH bifurcation)

Let
\[ \dot{x} = f(x, \alpha) \]  
be an autonomous dynamical system, dependent on a parameter \( \alpha \). Assume that
1) for all small values of \( \alpha \) the system has the equilibrium position \( x=0 \)
2) Jacobi matrix of the system \( A(\alpha) = f_x(x_0(\alpha), \alpha) \) has a pair of complex conjugate eigenvalues:
\[ \lambda_{1,2} = \mu(\alpha) \pm i \omega(\alpha) \]  
where \( \mu(0) = 0, \omega(0) = \omega(0) > 0 \) and \( n_s - \) number of eigenvalues with a negative real part, \( n_u - \) a positive real part, with the following conditions satisfied:
3) \( l_1(0) \neq 0 \), \( \) where \( l_1(0) - \) the first Lyapunov coefficient
4) \( \mu'(0) \neq 0 \)

Then the system is locally topologically equivalent to the following system:
\[ \begin{align*}
    \dot{y}_1 &= \beta y_1 - y_2 + \text{sign}(l_1(0))y_1(y_1^2 + y_2^2), \\
    \dot{y}_2 &= y_1 + \beta y_2 + \text{sign}(l_1(0))y_2(y_1^2 + y_2^2), \\
    \end{align*} \]  
\[ \begin{align*}
    \dot{y}^s &= -y^s, \dot{y}^u = y^u,
\end{align*} \]  
where \( y = (y_1, y_2)^T, y^s \in \mathbb{R}^{n_s}, y^u \in \mathbb{R}^{n_u}. \)

Thus, the stability loss is defined by the sign of the first Lyapunov coefficient.

Let’s use the Taylor series for the right hand side of eq. (a1):
\[ f(x, 0) = A_0 x + \frac{1}{2} B(x, x) + \frac{1}{6} C(x, x, x) + O(\|x^4\|), \]
where \( B, C - \) polynomials:
\[ B_j(x, y) = \sum_{k,l=1}^{n} \frac{\partial^2 f_j(\xi, 0)}{\partial \xi_k \partial \xi_l} |_{\xi=0} x_k x_l \]  
\[ C_j(x, y, z) = \sum_{k,l,m=1}^{n} \frac{\partial^3 f_j(\xi, 0)}{\partial \xi_k \partial \xi_l \partial \xi_m} |_{\xi=0} x_k x_l x_m \]

where \( j = 1,2.., n \). Let \( q - \) eigenvector of the Jacobi matrix, \( p - \) conjugate eigenvector:
\[ A_0 q = i \omega_0 q, \]
\[ A_0^T p = -i \omega_0 p. \]  
Assume that \( \langle p, q \rangle = 1 \), which can always be achieved through normalization, the first Lyapunov number can be found as:
Analytical calculations of the Lyapunov coefficient in large dynamical systems with nonlinearities is a very cumbersome work, therefore it is reasonable to apply a numerical algorithm to get its value \cite{2}. The algorithm is based on the fact that for coinciding vectors of multilinear functions \( B, C \) can be obtained as a directional derivatives:

\[
B(v, v) = \left. \frac{d^2}{d\tau^2} f(x^0 + \tau v, \alpha^0) \right|_{\tau=0},
\]

\[
C(v, v, v) = \left. \frac{d^3}{d\tau^3} f(x^0 + \tau v, \alpha^0) \right|_{\tau=0}.
\]

The finite difference approximation then is:

\[
B(v, v) = \frac{1}{h^2} \left( f(x^0 + hv, \alpha^0) - f(x^0 - hv, \alpha^0) \right)
\]

\[
C(v, v, v) = \frac{1}{8h^3} \left( f(x^0 + 3hv, \alpha^0) - 3f(x^0 + hv, \alpha^0) + 3f(x^0 - hv, \alpha^0) - f(x^0 - 3hv, \alpha^0) \right)
\]

where \( h<<1 \). The expressions for \( B \) and \( C \) with not coinciding vectors can be obtained from the properties of multilinear functions. Then the following steps take place:

1. Calculate the Jacobi matrix \( A = f(x^0, \alpha^0) \) for system (a1) at the equilibrium position with a critical value of the parameter.
2. Calculate real and imaginary components of the eigenvector, corresponding to eigenvalue \( i\omega_0 \) and its conjugate problem:

\[
\begin{align*}
AQ_R + \omega_0 q_I &= 0 \\
AQ_I - \omega_0 q_R &= 0 \\
A^T p_R - \omega_0 p_I &= 0 \\
A^T p_I + \omega_0 p_R &= 0
\end{align*}
\]

Vectors can be normalized in the following way:

\[
\langle q_R, q_R \rangle + \langle q_I, q_I \rangle = 1, \quad \langle q_R, q_I \rangle = 0,
\]

\[
\langle p_R, q_R \rangle + \langle p_I, q_I \rangle = 1, \quad \langle p_R, q_I \rangle - \langle p_I, q_R \rangle = 0.
\]

Orthogonality of real and imaginary parts of vector \( q \) can be achieved by introducing a new vector \( \tilde{q} = (k_1 + ik_2)q \), where coefficients \( k_1, k_2 \) can be found as:

\[
(k_1^2 - k_2^2)\langle q_I, q_R \rangle + k_1 k_2 (\langle q_R, q_R \rangle - \langle q_I, q_I \rangle) = 0.
\]
3. Calculate

\[ a = \left. \frac{d^2}{d\tau^2} f(x^0 + \tau q_R, \alpha^0) \right|_{\tau=0}, \]

\[ b = \left. \frac{d^3}{d\tau^3} f(x^0 + \tau q_l, \alpha^0) \right|_{\tau=0} \]

\[ c = \left. \frac{1}{4} \frac{d^2}{d\tau^2} \left[ f(x^0 + \tau (q_R + q_l), \alpha^0) - f(x^0 + \tau (q_R - q_l), \alpha^0) \right] \right|_{\tau=0} \]

4. Solve a system of algebraic equations and find vectors \( r, (s_R, s_l) \):

\[ Ar = a + b; \begin{cases} -A s_R - 2\omega_0 s_l = a + b, \\ 2\omega_0 s_R - A s_l = 2c \end{cases} \]

5. Calculate

\[ \sigma_1 = \left. \frac{1}{4} \frac{d^2}{d\tau^2} \langle p_R, f(x^0 + \tau (q_R + r), \alpha^0) - f(x^0 + \tau (q_R - r), \alpha^0) \rangle \right|_{\tau=0} \]

\[ \sigma_2 = \left. \frac{1}{4} \frac{d^2}{d\tau^2} \langle p_l, f(x^0 + \tau (q_l + r), \alpha^0) - f(x^0 + \tau (q_l - r), \alpha^0) \rangle \right|_{\tau=0} \]

\[ \Sigma_0 = \sigma_1 + \sigma_2. \]

6. Calculate

\[ \delta_1 = \left. \frac{1}{4} \frac{d^2}{d\tau^2} \langle p_R, f(x^0 + \tau (q_R + s_R), \alpha^0) - f(x^0 + \tau (q_R - s_R), \alpha^0) \rangle \right|_{\tau=0} \]

\[ \delta_2 = \left. \frac{1}{4} \frac{d^2}{d\tau^2} \langle p_R, f(x^0 + \tau (q_l + s_l), \alpha^0) - f(x^0 + \tau (q_l - s_l), \alpha^0) \rangle \right|_{\tau=0} \]

\[ \delta_3 = \left. \frac{1}{4} \frac{d^2}{d\tau^2} \langle p_l, f(x^0 + \tau (q_R + s_l), \alpha^0) - f(x^0 + \tau (q_R - s_l), \alpha^0) \rangle \right|_{\tau=0} \]

\[ \delta_4 = \left. \frac{1}{4} \frac{d^2}{d\tau^2} \langle p_l, f(x^0 + \tau (q_l + s_R), \alpha^0) - f(x^0 + \tau (q_l - s_R), \alpha^0) \rangle \right|_{\tau=0} \]

\[ \Delta_0 = \delta_1 + \delta_2 + \delta_3 - \delta_4. \]

7. Calculate

\[ \gamma_1 = \left. \frac{d^3}{d\tau^3} \langle p_R, f(x^0 + \tau q_R, \alpha^0) \rangle \right|_{\tau=0} \]
\[
\gamma_2 = \left. \frac{d^3}{d\tau^3} (p_I, f(x^0 + \tau q_I, \alpha^0)) \right|_{\tau=0} \\
\gamma_3 = \left. \frac{d^3}{d\tau^3} (p_R + p_I, f(x^0 + \tau (q_R + q_I), \alpha^0)) \right|_{\tau=0} \\
\gamma_4 = \left. \frac{d^3}{d\tau^3} (p_R - p_I, f(x^0 + \tau (q_R - q_I), \alpha^0)) \right|_{\tau=0} \\
\Gamma_0 = \frac{2}{3} (\gamma_1 + \gamma_2) + \frac{1}{6} (\gamma_3 + \gamma_4)
\]

8. Then the Lyapunov number is:
\[
l_1(0) = \frac{1}{2\omega_0} (\Gamma_0 - 2\Sigma_0 + \Delta_0)
\]
Appendix B. Data used in the calculations

Table B1. Model parameters from [4]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mass of a wheelset</td>
<td>( m_w = 1800 , kg )</td>
</tr>
<tr>
<td>moment of inertia of the wheelset</td>
<td>( I_{wx} = 625.7 , kg \cdot \text{m}^2 )</td>
</tr>
<tr>
<td></td>
<td>( I_{wy} = 133.92 , kg \cdot \text{m}^2 )</td>
</tr>
<tr>
<td>Wheel radius</td>
<td>( r_0 = 0.533 , \text{м} )</td>
</tr>
<tr>
<td>Half distance between the rails</td>
<td>( d = 0.7176 , \text{м} )</td>
</tr>
<tr>
<td>Wheel conicity</td>
<td>( \lambda = 0.05 )</td>
</tr>
<tr>
<td>Axial load</td>
<td>( W_A = 18000 , N )</td>
</tr>
<tr>
<td>Stiffness coefficient (transverse)</td>
<td>( K_y = 8.67 \times 10^4 , N/\text{м} )</td>
</tr>
<tr>
<td>Damping coefficient (transverse)</td>
<td>( C_y = 2.1 \times 10^4 , N \cdot \text{с}/\text{м} )</td>
</tr>
<tr>
<td>Stiffness coefficient of hunting</td>
<td>( K_y = 8.67 \times 10^4 , N/\text{м} )</td>
</tr>
<tr>
<td>Half a distance between the wheels</td>
<td>( b = 1 , \text{м} )</td>
</tr>
<tr>
<td>Lateral creep</td>
<td>( f_{11} = 6.728 \times 10^6 , N )</td>
</tr>
<tr>
<td>Torsional creep</td>
<td>( f_{22} = 1000 , N \cdot \text{м}^2 )</td>
</tr>
<tr>
<td>Transverse-torsional creep</td>
<td>( f_{12} = 1.2 \times 10^3 , N \cdot \text{м} )</td>
</tr>
<tr>
<td>Transverse creep</td>
<td>( f_{33} = 6.728 \times 10^6 , N )</td>
</tr>
<tr>
<td>Rail stiffness</td>
<td>( K_r = 1.617 \times 10^7 , N/\text{м} )</td>
</tr>
<tr>
<td>Gap between a wheel and rail</td>
<td>( \delta = 0.923 , \text{см} )</td>
</tr>
<tr>
<td>Damping coefficients of a rotational motion</td>
<td>( C_1 = 1.923 \times 10^4 )</td>
</tr>
<tr>
<td></td>
<td>( C_2 = 5.14 \times 10^5 )</td>
</tr>
<tr>
<td></td>
<td>( C_3 = -3.1127 \times 10^6 )</td>
</tr>
<tr>
<td></td>
<td>( C_4 = 5.14 \times 10^6 )</td>
</tr>
</tbody>
</table>
Table B2. Parameters’ values used in the two wheelset bogie model

Physical parameters:

<table>
<thead>
<tr>
<th></th>
<th>( m ) (tonnes)</th>
<th>( J_z ) (t·m²)</th>
<th>( J_x ) (t·m²)</th>
<th>( J_y ) (t·m²)</th>
<th>Length</th>
<th>Width</th>
<th>Height</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carriage</td>
<td>50</td>
<td>1500</td>
<td>127</td>
<td>1400</td>
<td>( a_1=7.5 )</td>
<td>( b_1=2.54 )</td>
<td>( h_1=1.595 )</td>
</tr>
<tr>
<td>Bogie</td>
<td>3</td>
<td>1.16</td>
<td>1.27</td>
<td>–</td>
<td>( a_2=2.6 )</td>
<td>( b_2=2.54 )</td>
<td>( h_2=0.525 )</td>
</tr>
<tr>
<td>Wheelset</td>
<td>2.86</td>
<td>2.26</td>
<td>2.26</td>
<td>–</td>
<td>( b_3=2.54 )</td>
<td></td>
<td>( h_3=0.45 )</td>
</tr>
</tbody>
</table>

Table B3. Stiffnesses (kN/m)

<table>
<thead>
<tr>
<th>( k_i ) (kN/m)</th>
<th>( k_i ) (kN/m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( k_{1y} )</td>
<td>1000</td>
</tr>
<tr>
<td>( k_{2y} )</td>
<td>1600</td>
</tr>
<tr>
<td>( k_{3y} )</td>
<td>130</td>
</tr>
<tr>
<td>( k_{4y} )</td>
<td>3000</td>
</tr>
<tr>
<td>( k_{1z} )</td>
<td>150</td>
</tr>
<tr>
<td>( k_{2z} )</td>
<td>7200</td>
</tr>
</tbody>
</table>

Table B4. Dissipation

<table>
<thead>
<tr>
<th>( \beta_i ) (kN/ms)</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \beta_{2y} ) (kN/ms)</td>
<td>Longitudinal direction between the carriage and bogie</td>
<td>100</td>
</tr>
<tr>
<td>( \beta_{3z} ) (kN/ms)</td>
<td>Vertical direction between the carriage and bogie</td>
<td>20</td>
</tr>
<tr>
<td>( \beta_{3y} ) (kN/ms)</td>
<td>Lateral direction between the carriage and bogie</td>
<td>230</td>
</tr>
<tr>
<td>( R_{2y} ) (kN)</td>
<td>Longitudinal dry friction</td>
<td>3</td>
</tr>
<tr>
<td>( R_{3y} ) (kN)</td>
<td>Later dry friction coefficient between the carriage and bogie</td>
<td>5</td>
</tr>
</tbody>
</table>
Appendix C.

Table C1. Values of the surface roughness coefficients

<table>
<thead>
<tr>
<th>Components</th>
<th>LH</th>
<th>RH</th>
<th>LV</th>
<th>RV</th>
<th>LVLH</th>
<th>LVRH</th>
<th>RVRH</th>
<th>RVLH</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{\eta,mm^2}^2$</td>
<td>_</td>
<td>6.75</td>
<td>8.28</td>
<td>27.0</td>
<td>26.2</td>
<td>2.90</td>
<td>2.70</td>
<td>1.8</td>
</tr>
<tr>
<td>$a_k$</td>
<td>1</td>
<td>0.000</td>
<td>0.000</td>
<td>0.010</td>
<td>0.010</td>
<td>0.010</td>
<td>0.010</td>
<td>0.010</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.150</td>
<td>0.200</td>
<td>0.150</td>
<td>0.200</td>
<td>0.250</td>
<td>0.200</td>
<td>0.220</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.700</td>
<td>0.650</td>
<td>0.700</td>
<td>0.750</td>
<td>0.600</td>
<td>0.700</td>
<td>0.600</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.150</td>
<td>0.150</td>
<td>0.140</td>
<td>0.040</td>
<td>0.140</td>
<td>0.090</td>
<td>0.170</td>
</tr>
<tr>
<td>$\alpha_{k,m^{-1}}$</td>
<td>1</td>
<td>0.000</td>
<td>0.000</td>
<td>0.040</td>
<td>0.040</td>
<td>0.040</td>
<td>0.040</td>
<td>0.040</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.015</td>
<td>0.020</td>
<td>0.010</td>
<td>0.010</td>
<td>0.021</td>
<td>0.021</td>
<td>0.019</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.040</td>
<td>0.040</td>
<td>0.035</td>
<td>0.035</td>
<td>0.030</td>
<td>0.035</td>
<td>0.035</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.021</td>
<td>0.032</td>
<td>0.025</td>
<td>0.021</td>
<td>0.028</td>
<td>0.035</td>
<td>0.035</td>
</tr>
<tr>
<td>$\beta_{k,m^{-1}}$</td>
<td>1</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.125</td>
<td>0.125</td>
<td>0.125</td>
<td>0.125</td>
<td>0.125</td>
<td>0.125</td>
<td>0.125</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.250</td>
<td>0.250</td>
<td>0.250</td>
<td>0.250</td>
<td>0.250</td>
<td>0.250</td>
<td>0.250</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.500</td>
<td>0.500</td>
<td>0.500</td>
<td>0.500</td>
<td>0.500</td>
<td>0.500</td>
<td>0.500</td>
</tr>
<tr>
<td>$\chi_c, m$</td>
<td>_</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5.0</td>
<td>4.5</td>
<td>8.0</td>
</tr>
</tbody>
</table>

where the coefficients are related to roughness in L-left and R-right rails and in H-horizontal and V-vertical directions, $x_c$ – shift coefficient.