Curving Dynamics in High Speed Trains
PUBLIC VERSION

Daniele Bigoni

Kongens Lyngby 2011
IMM-MSC-2011-59

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

Summary

This work presents a model for a generic railway vehicle running on straight or curved track with different profiles. The model employs the Newton-Euler formulation for dynamical systems. The wheel-rail interaction is modeled using the Hertz's static contact theory, corrected with the Kalker's theory for dynamical wheel-rail penetration. Tangential forces on the wheel-rail contact point are computed using the Kalker's linear theory with the appropriate corrections provided by the Shen, Hedrick and Elkins non-linear theory. Several type of elements of the suspension system have been modeled. The model is implemented in the program DYTSI. This is a framework for designing and testing railway vehicle models. DYTSI includes four numerical ODE algorithms that are used along the work: the Bulirsch-Stoer method, the Backward Differentiation Formula and two ESDIRK methods in the versions by Nielsen-Thomsen and by Jørgensen-Kristensen-Thomsen.

The hunting phenomenon has been studied on the Cooperrider model on straight and curved track. Results of previous works have been confirmed using DYTSI. The importance of precession forces on the dynamics, due to the high speed spinning of the wheel sets, has been highlighted. Additional results have been obtained for a complete wagon model and the symmetry assumption of the model running on straight track was rejected as different behaviors for the leading and trailing parts were obtained. On curved tracks the passage from the subcritical Hopf bifurcation to the super critical Hopf bifurcation was confirmed, for certain radii, also for the complete wagon model.

The dynamics of an AGV model, provided by ALSTOM, have been studied on curves with big radii and high cant deficiency. The results obtained, for the

model running on smooth tracks, have been confirmed.

Finally, the performances of the four ODE solvers have been compared on the highly stiff train dynamics problem. The ESDIRK methods have shown better stability for relaxed tolerances, but they require a big computational effort for finding accurate solutions. The BDF and the Bulisrch-Stoer methods turned out to be computationally more efficient, but they encountered stability problems on some of the test cases.

Keywords Train dynamics; Non-linear dynamics; Critical Speed; Cooperrider; ALSTOM AGV; ODE; ESDIRK; Bulirsch-Stoer; BDF method; Bifurcation Analysis; DYTSI.

Resumé

I dette arbejde præsenteres en model for general simulering af en jernbane togvogn der kører på lige eller kurvede jernbanelegemer med mulighed for valg af skinneprofiler. Modellen er baseret på en klassisk Newton-Euler formulering for dynamiske systemer. Hjul-skinne kontakt-kraft problemet er modeleret med Hertz's statiske kontakt teori ved brug af Kalker's teori for dynamisk hjul-skinne gennemtrængning. Tangentielle kræfter for hjul-skinne kontakt punkter beregnes med Kalker's lineære teori med passende korrektioner der opnås ved hjælp af Shen, Hedrick and Elkins ikke-linære teori. Flere forskellige elementer af systemet for vognophænget er modelleret. Modellen er implementeret I programmet DYTSI, der er et software værktøj til at designe og teste jernbane togvogne ved model simulering. DYTSI inkluderer fire numeriske ODE algoritmer som alle er blevet anvendt: Bulirsch-Stoer metoden, "Backward Differentiation Formula" metoder og to forskellige ESDIRK metoder i versioner udarbejdet af Nielsen-Thomsen og Jørgensen-Kristensen-Thomsen.

Periodiske rystelser I togvogn ("hunting" fænomen) er studeret med en Cooperrider model på lige og kurvede jernbanelegemer. Overensstemmelse med resultater opnået i tidligere arbejde er blevet genskabt ved hjælp af DYTSI. Vigtigheden af præcessions krafters påvirkning af dynamikken, der følger af høj hastigheder på hjulsættets spin, er blevet fremhævet. Nye resultater er opnået for en komplet model for en togvogn og symmetri antagelsen for modellen på et lige banelegeme er forkastet da forskellig dynamisk opførsel for the forreste og bagerste dele blev opnået i tests. På kurvede jernbanelegemer er passagen fra en subkritisk til en superkritisk Hopf bifurkation blevet bekræftet for vise radius, og ligeledes for den komplette model af en jernbanevogn.

Dynamikken af en AGV model, leveret af ALSTOM, er blevet studeret i kurver

med stor radius og høj hældning af skinneunderlag. De opnåede resultater for modellen på jævne jernbanelegemer er blevet bekræftet.

Endeligt, en undersøgelse af performance for fire forskellige ODE løsere er blevet sammenlignet på et meget stift jernbane dynamik problem. Denne undersøgelse viser at ESDIRK metoderne har bedre stabilitet for reduceret tolerance, men kræver større beregningsarbejde for at opnå nøjagtige løsninger. BDF og Bulirsh-Stoer metoderne viste sig at være mest beregningseffektive, men resulterede i stabilitetsproblemer i enkelte tests.

Preface

This thesis is a requirement for obtaining the Master of Science degree at the Technical University of Denmark (DTU). The project is the result of a collaboration between DTU and ALSTOM Transport, France. The work has been mainly carried out at the Informatics and Mathematical Modelling department of DTU, under the supervision of associate professor Allan Peter Engsig-Karup and emeritus associate professor Hans True. The project was enriched by a period spent at the manufacturing facility of ALSTOM Transport, Le Creusot (France). This project was started on the 1^{st} of February 2011 and completed on the 26^{th} of August 2011.

The thesis has been delivered to the examiners on the 10^{th} of August 2011. It contained the chapters 1-7 and the appendices A-G. Amendments have been added in chapter 8, as a results of the observations pointed out during the thesis defense and some additional work done aiming to the publication of the article "*On the Numerical and Computational Aspects of Non-Smoothnesses that occur in Railway Vehicle Dynamics*".

Lyngby, August 2011

Daniele Bigoni

Acknowledgements

I would like to thanks my supervisors Allan P. Engsig-Karup and Hans True, that have provided great support for this work and were able to share with me their passion for the covered topics. A thanks goes to Sergio Perez-Anton, Fabienne Bondon and all the "dynamics team" at ALSTOM Transport (Le Creusot, France), that have provided fundamental data for this work, technical support and a good experience in the industry.
I would like to thanks my parents, Aldino and Roberta, for being good examples and for supporting a quite nomadic son. Thanks to my sister Valentina for updating me with news from my place and for introducing me to math when I was a child.
I thank all the national and international friends that I met along these almost three years abroad, for the support, physical and moral, and the funny hours spent sipping beers. Thanks to the everlasting friends that sometimes popped up wondering about me. Thanks to Tommaso and Filippo for being source of infinite foolishness and friendship.
Finally, thanks to Stefania for bringing me back to the world with feelings everyday, and for being as she is.

I would also thanks my kitchen furniture for helping me feeling at home and my bike for carrying me around regardless of the weather conditions.

Vorrei ringraziare i miei supervisors Allan P. Engsig-Karup e Hans True, che mi hanno fornito un grande supporto durante questo lavoro e sono stati capaci di condividere con me la loro passione per gli argomenti trattati. Un ringraziamento va a Sergio Perez-Anton, Fabienne Bondon e a tutto il "dynamics team" presso ALSTOM Transport (Le Creusot, France), che hanno fornito dati fondamentali per la realizzazione di questo lavoro, supporto tecnico e una buona esperienza nell'industria.
Ringrazio i miei genitori, Aldino e Roberta, per essere stati buoni esempi e per aver supportato (e sopportato) un figlio alquanto nomadico. Ringrazio mia sorella Valentina per avermi aggiornato sulle news casalinghe e avermi introdotto alla matematica quando ero bambino.
Ringrazio tutti gli amici, nazionali e internazionali, che ho incontrato durante questi quasi tre anni all'estero, per il supporto, fisico e morale, e le deliziose ore spese sorseggiando birra. Grazie ai miei amici di vecchia data che ogni tanto sono sbucati curiosi di saper le mie condizioni. Grazie a Tommaso e Filippo per essere sorgenti di inifinita idiozia e amicizia.
Per finire, grazie a Stefania per riportarmi nel mondo sensibile ogni giorno e per essere com'è.

Vorrei anche ringraziare le stoviglie della mia cucina per avermi aiutato a sentrimi a casa e la mia bicicletta per avermi trasportato in giro sotto tutte le condizioni atmosferiche.

Contents

CHAPTER 1

Introduction

Railways were first used at the beginning of the 19^{th} century and they have been the most popular means of transportation on long distances (continental and intercontinental) until the end of the second world war and the take off of passenger aviation. Nonetheless, train transportation is still very common on medium distance transportation for goods as well as for passengers. Several are the factors that drive the transportation market: safety, comfort, speed (and punctuality), price and lately also the environmental impact. All these factors are interconnected and an increase of the cruising speed can result in a worsen safety and comfort as well as in higher prices and environmental impact. Thus, research need to focus in balancing these values and provide the optimal combination of these factors.

In spite of the apparent simplicity of railway vehicles, a lot of problems are still under investigation after almost two hundreds years of research and experience. For long time the speed at which a train could run was limited by the stability of the sideway motions rather than by the power of the engine. Lateral stability problems were first discussed by Stephenson in the 1821[Ste21] when he observed that lateral motion could develop on vehicles, even on straight track. In railway terminology this phenomenon is called *hunting motion* because the dynamics of the train get captured by the lateral motion when it reaches a certain *critical speed* and it's hard to gain stability again, even decelerating. This problem affects the rail comfort and the safety as a big lateral motion could

lead to the derailment of the train. At the time, static analysis was the only mathematical framework available in order to study this phenomenon. Even if it proved to be useful, static analysis has shown several shortcomings in the complete explanation of the hunting motion and even the derailment due to flange climbing.
The development of linear and non-linear dynamic analysis shed light on some of the problems on railway vehicles dynamics. An analytical explanation for the hunting motion has been found for simple models, but the same approach is not applicable to complex models like complete vehicles combined with flexible and non-smooth rails. The advent of numerical simulation has been crucial for the study of complex dynamical systems like railway vehicles. These new frameworks provided better explanations for the hunting problem. They also turned useful in investigating the cases of derailment due to flange climbing, in studying the wear of the wheel and rail profiles, in estimating the acceleration and the level of ride comfort inside the vehicles. Nowadays, in the industry, numerical simulations are heavily used during the development of new vehicle models, in both the design and testing phases, and they enabled the creation of high-speed ($< 300\frac{\mathrm{km}}{\mathrm{h}}$) and very-high-speed ($> 300\frac{\mathrm{km}}{\mathrm{h}}$) trains.

New range of speeds opens the way to new challenges. Stability is an issue that has to be addressed on straight tracks as on curves, and the behavior of vehicles can be quite different in such situations. Furthermore, the industry is quite concerned with the excessive wear due to the higher forces caused by the increased speed: the costs of maintenance of the infrastructure (rails in particular) and of the vehicle components can result unacceptable, pushing this means of transport out of market.

This work will focus on the study of the dynamics of a wagon model composed by two Cooperrider bogies and a realistic vehicle model provided by ALSTOM Transport. The modeling framework, presented in chapter 2, is based on previous works carried out at IMM-DTU and extended in a flexible modeling framework able to easily assemble new wagon designs. The model involves well known theories for the wheel-rail interaction that allow the usage of static contact data, computed using the RSGEO routine. Chapter 3 is dedicated to the presentation and the investigation of the properties of the numerical methods used for addressing the vehicle dynamics problem. Chapter 4 introduces the program that assembles the design framework and the solvers in a unique package called DYTSI-DYnamics Train SImulation. In chapter 5 a brief introduction to non-linear dynamics on railway vehicles will be provided. Chapter 6 presents the results obtained for the models studied and some observation on the behaviors of the numerical methods on such problems. Finally chapter 7 wraps up the work done and presents some possible future works.

Chapter 2

Vehicle Modeling

The model considered in this work is a four-axle bogie wagon, like the one shown in figure 2.1. A car body is connected through the secondary suspensions to two bogie frames, that in turn are connected to two wheel sets through the primary suspensions. Suspensions can include several type of components that can have linear or non-linear behaviors. The figure 2.2 shows the nomenclature that will be used during the work. The vehicle will have no traction engine, thus an external force will tow it at constant or quasi constant speed on an horizontal track that can be either straight or curved. The track will have a cant toward the center of the curve in order to improve the ride comfort and the safety.

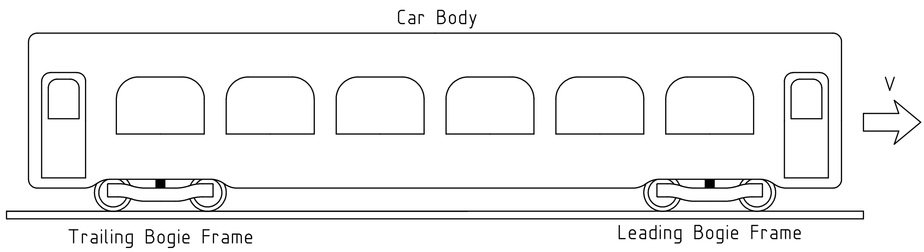
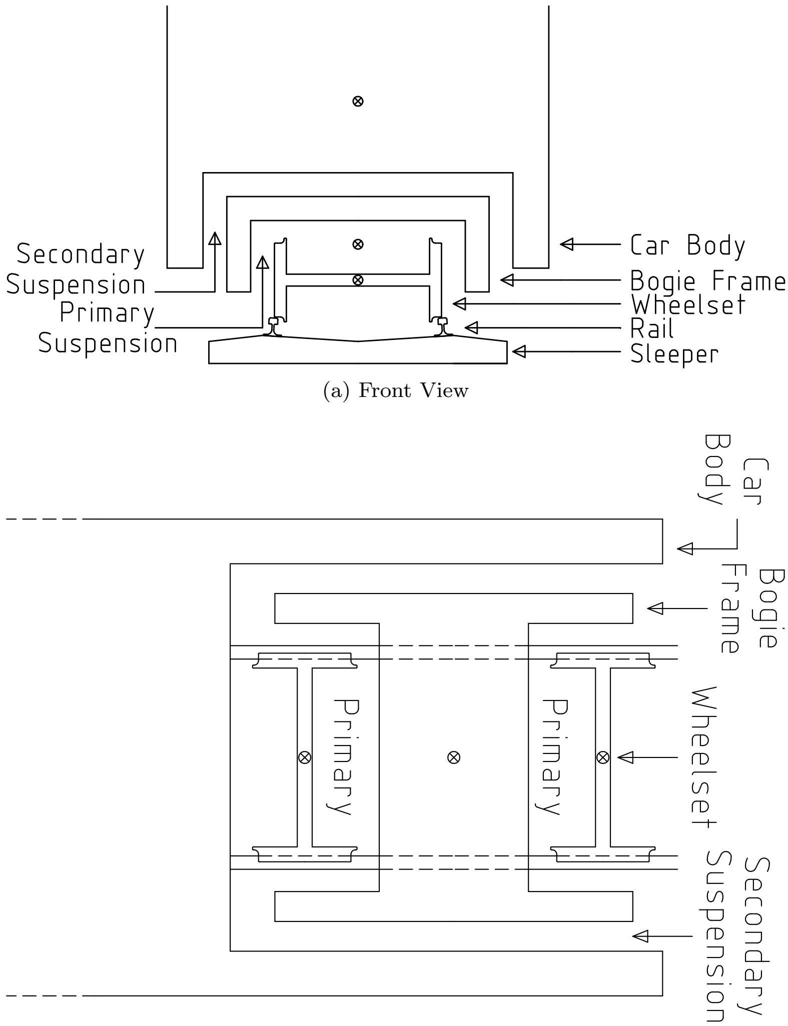


Figure 2.1: Lateral view of a four-axle bogie wagon.



(a) Front View

(b) Top View

Figure 2.2: Nomenclature for the four-axle bogie wagon.

In the field of mechanics, a dynamical system is written in terms of equations of motion. The dynamic behavior of a machinery can be described using two different approaches: the equilibrium of energy or the equilibrium of forces. The first one is called the Lagrangian formulation of dynamical systems, the second one is the Newton-Euler formulation. The approach adopted in this work is the

Newton-Euler, where the equations of motion are described by:

$$\sum_{i=1}^{n} {}^{I}\vec{F}_i = m\vec{a} \qquad \text{(Newton's Law)} \tag{2.1}$$

$$\sum_{i=1}^{m} {}^{B}\vec{M}_i = \frac{d}{dt}\left({}^{B}[J]\ {}^{B}\vec{\omega}\right) + {}^{B}\vec{\omega} \times \left({}^{B}[J]\ {}^{B}\vec{\omega}\right) \qquad \text{(Euler's Law)} \tag{2.2}$$

where ${}^{I}\vec{F}_i$ and ${}^{B}M_i$ are, respectively, the forces and torques acting on the center of mass, m and $[J]$ are the mass and the tensor moment of inertia respectively, $\vec{a}$ and $\dot{\vec{\omega}}$ are the linear acceleration and the angular acceleration of the bodies. The superscript I in ${}^{I}\vec{F}_i$ stands for Inertial and indicates that the forces are written in the inertial reference frame. This left superscript notation will be used during all the following work with the meaning that the corresponding vector is written in a particular reference frame. In the Euler's Law, the superscript B stands for the body reference frame. It's important to point out that the forces have to be written in the inertial reference frame in order for the Newton's law to be valid. On the contrary, the torques can be written in any of the reference systems, but in order to keep the tensor moment of inertia constant, they will be written in the body reference frame attached to its center of mass.

2.1 Reference Frames and Degrees of Freedom

The first step for modeling a multibody system is to identify some reference frames where the bodies can be described. As it was introduced, Newton's law holds only if applied in an inertial reference system (I). For phenomena like the running of a train, the earth reference system can be considered inertial (Coriolis' effects due to the earth rotation can be neglected because they have little influence on train dynamics). However this gives precision problems considering that the train is moving with respect to this system at the rate of tens of meters per second and the dynamics that we want to observe act in the the range of millimeters. Therefore it is necessary to introduce a track following reference system positioned at the center of the track at the height of the top of the track and moving with the speed of the train. All the other reference systems will be written with respect to this track following one. It is stressed that these reference systems are not inertial, so fictitious forces have to be considered, however their use will simplify the writing and the readability of the equation of motion. The reference frames defined will be listed below and can be seen in figure 2.3.

- I: The inertial reference system is placed anywhere on the earth and has $\{\mathbf{X}, \mathbf{Y}, \mathbf{Z}\}^T$ as base vectors.

- F: The track following reference frame will be attached to the center of the track and will run at the speed of the train. The base vectors will be denoted by $\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}^T$. Depending on the track geometries this frame will rotate around its axis. Since only horizontal curves with cant are considered, only rotations around the x and z axis will be considered. These rotations will determine fictitious forces that have to be considered whenever this frame is used.

- W, B, C: These reference frames refer to the bodies to which they are attached, respectively the wheel sets, the bogie frames and the car body. They are positioned in the center of mass of the bodies such that the tensor of inertia will be constant in time. Even if the center of mass is a fundamental quantity to know when the Newton-Euler equations are to be applied, it is not a suitable reference system when the geometrical position of points in the component has to be determined. The center of mass is difficult to determine and usually it doesn't correspond to the center of geometry of the component. In this modeling part the center of mass will be assumed to be in the center of geometry, but the implementation of the framework will allow these two points to be in different positions.

- $R(\xi, \eta, \zeta)$, $L(\xi, \eta, \zeta)$: These reference frames are attached respectively to the right and to the left contact point on the rails. The reference plane (ξ, η) is tangent to the contact ellipse.

All the reference frames (a part the inertial) will be free to move in the inertial reference frame. The possible rotations of the track following reference frame are shown in figure 2.4. All the rotations will be denoted by the angles ϕ for roll, χ for pitch and ψ for yaw. The rotations of the track following system will be marked by an additional subscript t, in order to distinguish the rotations that are due to the nominal track geometries and the rotations that are due to the dynamics of the system.
Several transformation matrices have to be considered in order to transform vectors from one reference frame to another. It was already stated that the main reference frame with respect to which all the displacements and rotations will be computed is the track following reference frame. So the important matrices are the one that transform all the reference frames to the track following one. Additional matrices can be obtained by composition of these. The main transformation matrices are listed in Appendix B.

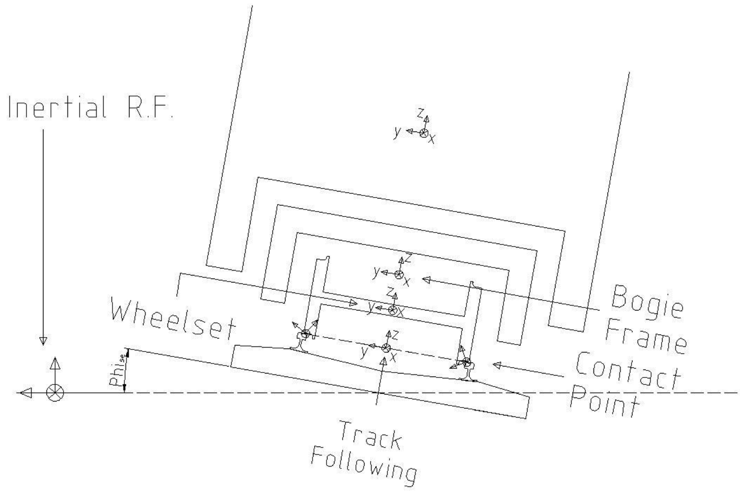


Figure 2.3: Reference frames considered for modeling the system. The track following reference frame is positioned at the center of track plane and moves with the train speed. Each body reference frame is attached to the respective center of mass. The contact point reference frames are determined by the orientation of the contact patch. The inertial reference frame is placed somewhere in the space.

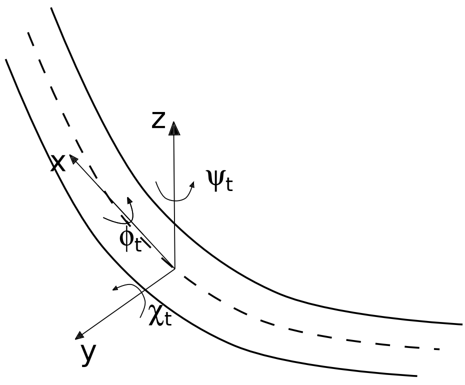


Figure 2.4: Positive direction of the rotation around the axis. The rotations comply with the right-hand grip rule.

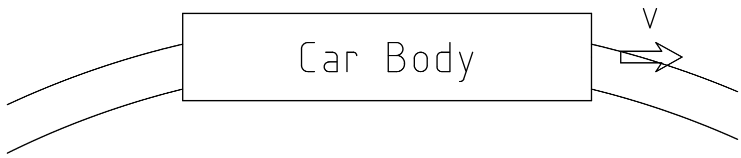


Figure 2.5: Right hand curve with positive radius. The track is canted toward the center of the curve in order to accommodate better the centrifugal forces.

2.1.1 The Track Following Reference System

The Track Following reference system is chosen as the main reference system for simplicity. However, its movement with respect to the inertial reference system will introduce some forces on the bodies that are moving with it. In order to study the equations of motion of the track following reference system with respect to the Inertial Frame the model proposed in [ABS07, Ch. 6] will be adopted. The different orientation of the reference systems considered in this work and the reference systems considered in [ABS07, Ch. 6] will require some additional discussion.
The track following reference system is moving on the center of the track with

$$ {}^{I}\mathbf{v}^{F} = \left\{ \begin{array}{c} v \\ 0 \\ 0 \end{array} \right\} \quad \text{and} \quad {}^{F}\tfrac{d}{dt}\left({}^{I}\mathbf{v}^{F}\right) = \left\{ \begin{array}{c} 0 \\ 0 \\ 0 \end{array} \right\} \tag{2.3} $$

where ${}^{F}\frac{d}{dt}$ is the differentiation with respect to the track following reference frame. The rail is covering an horizontal curve ($\chi_t = 0$), so the track following reference frame will rotate around its z axis. We define *positive curve* the curve that causes $d\psi_t > 0$. However, since the framework for finding the creep forces considers the radius R and the cant angle ϕ_t positive for a right hand curve, $d\psi_t < 0$ will be considered. For a right hand curve, the cant of the rail has to be toward the center of the curve, so $\phi_t > 0$. This is the main difference from the model used in [ABS07, Ch. 6] and causes the need of some changes in the equations. These changes will be highlighted along the work. Figure 2.5 shows an example of a right hand curve.
Since the formulation adopted is Newton-Euler, the computation of the derivative up to the second order of the position vector of the track following reference system will be required. This is done in the following relations:

$$ {}^{I}\mathbf{v}^{F} = {}^{I}\left(\tfrac{d}{dt}\right)\left({}^{I}\mathbf{p}^{F}\right) \tag{2.4} $$

$$ {}^{I}\mathbf{a}^{F} = {}^{F}\left(\tfrac{d}{dt}\right)\left({}^{I}\mathbf{v}^{F}\right) + {}^{I}\omega^{F} \times {}^{I}\mathbf{v}^{F} \tag{2.5} $$

$$ {}^{I}\dot{\omega}^{F} = {}^{F}\left(\tfrac{d}{dt}\right)\left({}^{I}\omega^{F}\right) \tag{2.6} $$

where the position ${}^{I}\mathbf{p}^{F}$, the speed ${}^{I}\mathbf{v}^{F}$ and the acceleration ${}^{I}\mathbf{a}^{F}$ of the track following reference system on the track are known whereas the angular velocities ${}^{I}\omega^{F}$ has to be derived and related to the nominal track geometry. In order to represent the rotations, three rotating steps around the principal axis will be taken:

$$\begin{Bmatrix} \mathbf{e}_1^A \\ \mathbf{e}_2^A \\ \mathbf{e}_3^A \end{Bmatrix} = \begin{bmatrix} \cos\psi_t & \sin\psi_t & 0 \\ -\sin\psi_t & \cos\psi_t & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} \mathbf{X} \\ \mathbf{Y} \\ \mathbf{Z} \end{Bmatrix} \tag{2.7}$$

$$\begin{Bmatrix} \mathbf{e}_1^B \\ \mathbf{e}_2^B \\ \mathbf{e}_3^B \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} \mathbf{e}_1^A \\ \mathbf{e}_2^A \\ \mathbf{e}_3^A \end{Bmatrix} \tag{2.8}$$

$$\begin{Bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{z} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_t & \sin\phi_t \\ 0 & -\sin\phi_t & \cos\phi_t \end{bmatrix} \begin{Bmatrix} \mathbf{e}_1^B \\ \mathbf{e}_2^B \\ \mathbf{e}_3^B \end{Bmatrix} \tag{2.9}$$

Superposition of the three angular velocities gives the angular velocity vector:

$${}^{I}\omega^{F} = \dot{\psi}_t \mathbf{e}_3^A + \dot{\phi}_t \mathbf{x} \tag{2.10}$$

This vector can be rewritten in only the base vectors of the track following system, obtaining:

$${}^{I}\omega^{F} = \begin{Bmatrix} \dot{\phi}_t \\ \dot{\psi}_t \sin\phi_t \\ \dot{\psi}_t \cos\phi_t \end{Bmatrix} \tag{2.11}$$

The angular velocities will now be related to the nominal track geometry and the speed of the vehicle:

$$\dot{\phi}_t = \frac{d\phi_t}{dt} = \frac{d\phi_t}{ds}\frac{ds}{dt} = \phi_t' v \tag{2.12}$$

$$\dot{\psi}_t = \frac{d\psi_t}{dt} = \frac{d\psi_t}{ds}\frac{ds}{dt} = -\frac{v}{R} \tag{2.13}$$

where the relation

$$d\psi_t = -\frac{ds}{R} \tag{2.14}$$

was used. In (2.14) ds is the differential of the length of the rail covered during the curve and the radius R is positive for right handed curves, causing the additional minus with respect to the formulation in [ABS07, Ch. 6].

Now inserting (2.12) and (2.13) into (2.11) and considering constant cant ($\phi_t' = 0$) gives:

$$ {}^{I}\omega^{F} = \begin{Bmatrix} 0 \\ -\frac{v}{R}\sin\phi_t \\ -\frac{v}{R}\cos\phi_t \end{Bmatrix} \tag{2.15} $$

Inserting (2.15) and (2.3) in (2.4), (2.5) and (2.6) the motion vectors of the track following reference system are obtained.

$$ {}^{I}\mathbf{v}^{F} = \begin{Bmatrix} v \\ 0 \\ 0 \end{Bmatrix} \tag{2.16} $$

$$ {}^{I}\mathbf{a}^{F} = {}^{F}\left(\tfrac{d}{dt}\right)\begin{Bmatrix} v \\ 0 \\ 0 \end{Bmatrix} + \begin{Bmatrix} 0 \\ -\frac{v}{R}\sin\phi_t \\ -\frac{v}{R}\cos\phi_t \end{Bmatrix} \times \begin{Bmatrix} v \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} 0 \\ -\frac{v^2}{R}\cos\phi_t \\ \frac{v^2}{R}\sin\phi_t \end{Bmatrix} \tag{2.17} $$

$$ {}^{I}\omega^{F} = \begin{Bmatrix} 0 \\ -\frac{v}{R}\sin\phi_t \\ -\frac{v}{R}\cos\phi_t \end{Bmatrix} \tag{2.18} $$

$$ {}^{I}\dot{\omega}^{F} = {}^{F}\left(\tfrac{d}{dt}\right)\begin{Bmatrix} 0 \\ -\frac{v}{R}\sin\phi_t \\ -\frac{v}{R}\cos\phi_t \end{Bmatrix} = \begin{Bmatrix} 0 \\ -\frac{a}{R}\sin\phi_t \\ -\frac{a}{R}\cos\phi_t \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 0 \end{Bmatrix} \tag{2.19} $$

2.1.2 The Body Following Reference System

The wagon of a train is composed by multiple bodies that move with respect to the track following reference system. These motion will be described by the little displacement of the center of mass of each body with respect to the track following reference system. The position of the centers of mass will be expressed as the sum of a nominal position plus displacements. Figure 2.6 shows the position of the center of mass with respect to the track following reference frame. The position vector is given by a nominal part and a displacement part. The nominal position represent the situation of the car resting on the center of the track before any force is applied to it. The position vector can be written as

$$ {}^{F}\mathbf{p}^{C^*} = \begin{Bmatrix} l + x \\ b + y \\ h + z \end{Bmatrix} \tag{2.20} $$

where l, b, h are the nominal coordinates of the center of mass and x, y, z are the displacements. This vector is written in the non inertial reference frame F, so time derivatives of it will give also components due to the movement of the

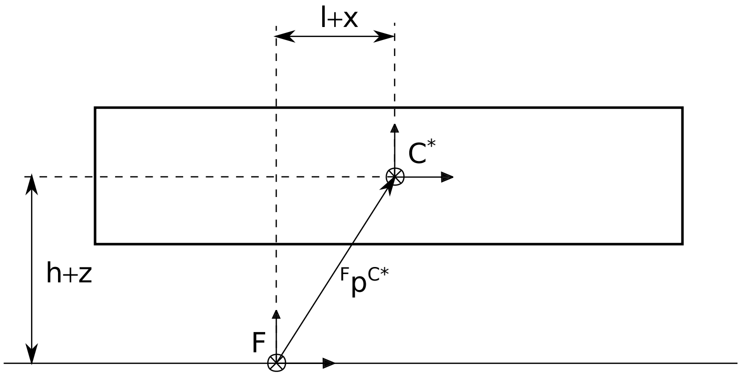


Figure 2.6: Location of the center of mass with respect to the track following reference frame.

track following reference frame. The acceleration vector is found in [ABS07, Ch. 6] and plugged in 2.1, giving as result the Newton part of the equations of motion:

$$
\begin{aligned}
& m\left[\ddot{x}+\omega_2(2\dot{z}+\omega_1 y-\omega_2 x)-\omega_3(2\dot{y}+\omega_3 x-\omega_1 z)+\dot{\omega}_2 z+\dot{\omega}_3 y\right] \\
& \quad = {}^F F_x - m[a+\omega_2(\omega_1 b-\omega_2 l)-\omega_3(\omega_3 l-\omega_1 h)+\dot{\omega}_2 z-\dot{\omega}_3 b] && (2.21) \\
& m\left[\ddot{y}+\omega_3(2\dot{x}+\omega_2 z-\omega_3 y)-\omega_1(2\dot{z}+\omega_1 y-\omega_2 x)+\dot{\omega}_3 x-\dot{\omega}_1 z\right] \\
& \quad = {}^F F_y - m\left[\omega_3 v+\omega_3(\omega_2 h-\omega_3 b)-\omega_1(\omega_1 b-\omega_2 l)+\dot{\omega}_3 l-\dot{\omega}_1 h\right] && (2.22) \\
& m\left[\ddot{z}+\omega_1(2\dot{y}+\omega_3 x-\omega_1 z)-\omega_2(2\dot{x}+\omega_2 z-\omega_3 y)+\dot{\omega}_1 y-\dot{\omega}_2 x\right] \\
& \quad = {}^F F_z - m\left[-\omega_2 v+\omega_1(\omega_3 l-\omega_1 h)-\omega_2(\omega_2 h-\omega_3 b)+\dot{\omega}_1 b-\dot{\omega}_2 l\right] && (2.23)
\end{aligned}
$$

It's worth to notice that now the remaining forces can be written in the track following reference frame. The insertion of the equations (2.16) - (2.19) into the equations of motion (2.21) - (2.23) gives:

$$
\ddot{x} = \frac{{}^F F_x}{m} + (l+x)\frac{v^2}{R^2} + 2\dot{z}\frac{v}{R}\sin\phi_t - 2\dot{y}\frac{v}{R}\cos\phi_t \tag{2.24}
$$

$$
\ddot{y} = \frac{{}^F F_y}{m} + \frac{v^2}{R}\cos\phi_t + 2\dot{x}\frac{v}{R}\cos\phi_t - (h+z)\frac{v^2}{R^2}\sin\phi_t\cos\phi_t + (b+y)\frac{v^2}{R^2}\cos^2\phi_t \tag{2.25}
$$

$$
\ddot{z} = \frac{{}^F F_z}{m} - \frac{v^2}{R}\sin\phi_t - 2\dot{x}\frac{v}{R}\sin\phi_t + (h+z)\frac{v^2}{R^2}\sin^2\phi_t - (b+y)\frac{v^2}{R^2}\sin\phi_t\cos\phi_t \tag{2.26}
$$

In addition to the linear displacements, the body is free to rotate, so the description provided by the moment equation in terms of nominal angles and rotational angles is used. The nominal angles in the three directions of rotation are zero, instead the rotational angles are given by the unknown ϕ, χ and ψ. The angular velocity and angular accelerations can be approximated by:

$$ {}^{I}\omega^{C} \approx [\omega_1 \mathbf{x} + \omega_2 \mathbf{y} + \omega_3 \mathbf{z}] + \left[\dot{\phi}\mathbf{x} + \dot{\chi}\mathbf{y} + \dot{\psi}\mathbf{z}\right] \tag{2.27} $$

$$ \begin{aligned} {}^{I}\dot{\omega}^{C} \approx & [\dot{\omega}_1 \mathbf{x} + \dot{\omega}_2 \mathbf{y} + \dot{\omega}_3 \mathbf{z}] + \left[\ddot{\phi}\mathbf{x} + \ddot{\chi}\mathbf{y} + \ddot{\psi}\mathbf{z}\right] + \\ & + [\omega_1 \mathbf{x} + \omega_2 \mathbf{y} + \omega_3 \mathbf{z}] \times \left[\dot{\phi}\mathbf{x} + \dot{\chi}\mathbf{y} + \dot{\psi}\mathbf{z}\right] \end{aligned} \tag{2.28} $$

Assuming the products of inertia to be zero (${}^{B}[J]$ diagonal), the substitution of (2.27) and (2.28) in (2.2) gives the moment equations of the body:

$$ J_\phi \left(\dot{\omega}_1 + \ddot{\phi} + \omega_2\dot{\psi} - \omega_3\dot{\chi}\right) - (J_\chi - J_\psi)(\omega_2 + \dot{\chi})\left(\omega_3 + \dot{\psi}\right) \approx M_\phi \tag{2.29} $$

$$ J_\chi \left(\dot{\omega}_2 + \ddot{\chi} + \omega_3\dot{\phi} - \omega_1\dot{\psi}\right) - (J_\psi - J_\phi)\left(\omega_3 + \dot{\psi}\right)\left(\omega_1 + \dot{\phi}\right) \approx M_\chi \tag{2.30} $$

$$ J_\psi \left(\dot{\omega}_3 + \ddot{\psi} + \omega_1\dot{\chi} - \omega_2\dot{\phi}\right) - (J_\phi - J_\chi)\left(\omega_1 + \dot{\phi}\right)(\omega_2 + \dot{\chi}) \approx M_\psi \tag{2.31} $$

Substituting the linear speed, linear acceleration, angular speed and angular acceleration of the track following reference system from (2.16) - (2.19) into (2.29) - (2.31), the Euler's part of the equations of motion of the body following reference frame is obtained:

$$ \begin{aligned} M_\phi \approx & J_\phi \left(\ddot{\phi} - \dot{\psi}\frac{v}{R}\sin\phi_t + \dot{\chi}\frac{v}{R}\cos\phi_t\right) - \\ & - (J_\chi - J_\psi)\left(\frac{v^2}{R^2}\sin\phi_t\cos\phi_t - \dot{\psi}\frac{v}{R}\sin\phi_t - \dot{\chi}\frac{v}{R}\cos\phi_t + \dot{\chi}\dot{\psi}\right) \end{aligned} \tag{2.32} $$

$$ M_\chi \approx J_\chi \left(\ddot{\chi} - \dot{\phi}\frac{v}{R}\cos\phi_t\right) - (J_\psi - J_\phi)\left(-\dot{\phi}\frac{v}{R}\cos\phi_t + \dot{\psi}\dot{\phi}\right) \tag{2.33} $$

$$ M_\psi \approx J_\psi \left(\ddot{\psi} - \dot{\phi}\frac{v}{R}\sin\phi_t\right) - (J_\phi - J_\chi)\left(-\dot{\phi}\frac{v}{R}\sin\phi_t + \dot{\phi}\dot{\chi}\right) \tag{2.34} $$

The forces and torques due to the geometry of the track and the speed of the train will be collected in the vectors ${}^{I}\vec{F}_c$ and ${}^{B}\vec{M}_c$ where c stands for centrifugal due to the big contribution in the lateral direction.

$$ {}^{F}\vec{F}_c = \left\{ \begin{array}{c} m\left[(l+x)\frac{v^2}{R^2} + 2\dot{z}\frac{v}{R}\sin\phi_t - 2\dot{y}\frac{v}{R}\cos\phi_t\right] \\ m\left[\frac{v^2}{R}\cos\phi_t + 2\dot{x}\frac{v}{R}\cos\phi_t - (h+z)\frac{v^2}{R^2}\sin\phi_t\cos\phi_t + (b+y)\frac{v^2}{R^2}\cos^2\phi_t\right] \\ m\left[-\frac{v^2}{R}\sin\phi_t - 2\dot{x}\frac{v}{R}\sin\phi_t + (h+z)\frac{v^2}{R^2}\sin^2\phi_t - (b+y)\frac{v^2}{R^2}\sin\phi_t\cos\phi_t\right] \end{array} \right\} \tag{2.35} $$

$$
{}^{B}\vec{M}_c = \left\{ \begin{array}{c} \left\{ J_\phi \left(\dot{\psi} \frac{v}{R} \sin\phi_t - \dot{\chi} \frac{v}{R} \cos\phi_t \right) + \right. \\ \left. (J_\chi - J_\psi) \left(\frac{v^2}{R^2} \sin\phi_t \cos\phi_t - \dot{\psi} \frac{v}{R} \sin\phi_t - \dot{\chi} \frac{v}{R} \cos\phi_t + \dot{\chi}\dot{\psi} \right) \right\} \\ J_\chi \dot{\phi} \frac{v}{R} \cos\phi_t + (J_\psi - J_\phi) \left(-\dot{\phi} \frac{v}{R} \cos\phi_t + \dot{\psi}\dot{\phi} \right) \\ J_\psi \dot{\phi} \frac{v}{R} \sin\phi_t + (J_\phi - J_\chi) \left(-\dot{\phi} \frac{v}{R} \sin\phi_t + \dot{\phi}\dot{\chi} \right) \end{array} \right\} \tag{2.36}
$$

Another force that is acting on each body of the model and is related to the track geometry, is the gravitational force. The following relation define this force depending on the track geometry:

$$
{}^{F}\vec{F}_g = \left\{ \begin{array}{c} 0 \\ -mg\sin\phi_t \\ -mg\cos\phi_t \end{array} \right\} \tag{2.37}
$$

$$
{}^{B}\vec{M}_g = \left\{ \begin{array}{c} 0 \\ 0 \\ 0 \end{array} \right\} \tag{2.38}
$$

Finally the equations of motion will be written as a system of second order differential equations with three forcing terms, namely the gravity, the centrifugal forces and the forces due to the suspensions and contact forces.

$$
m\ddot{\vec{x}} = {}^{F}\vec{F} + {}^{F}\vec{F}_g + {}^{F}\vec{F}_c \tag{2.39}
$$

$$
[J]\dot{\vec{\omega}} = {}^{B}\vec{M} + {}^{B}\vec{M}_g + {}^{B}\vec{M}_c \tag{2.40}
$$

2.1.3 Model reduction

It often happens that simplification of a big non-linear system has to be made in order to reach a fairly precise solution in a reasonable amount of time. In many cases these simplifications are due to assumptions made during the modeling phase. The same approach is taken here where some assumptions will be discussed and the model will be reduced accordingly.

The first assumption is that longitudinal displacements are negligible. However it's important to know what the model is missing using this assumption. When a train is running on a rigid track, the speed at which the wheel rotate is not the same speed at which the train moves. This is due to the creep between wheel and rail, that causes tangential forces along the contact plane. This phenomenon is better explained in Section 2.2, but it's important to understand here that these contact forces, in particular during a curve, are oddly distributed in the x direction, causing big displacement in the x direction for the whole vehicle. Intuitively, a trailed train running at constant speed will end up covering

more space when it is on a curve due to the slip between rail and track. This difference can generate big terms in (2.24) as long as no deceleration is considered in the system, in order to even out the exceeding speed. A work around to this situation is to consider $\ddot{x} = \dot{x} = x = 0$. However this simplification could influence the results: the uneven distribution of the contact forces during a curve cause a longitudinal displacement of each component, starting from the wheel set, that in turns can change the resulting forces due to the suspensions, in particular when non linear elements are considered. Thus, the reliability of the results obtained using this assumption has to be tested and validated with real test data.
In nonlinear dynamics analysis, it's common practice to linearize the systems whenever small angles and small displacements from the steady state are present. In this work the angle ϕ_t will be in the range $[-0.126; 0.126]$ that correspond to a maximum cant of 180mm of the outer rail in a standard gauge track of 1435mm. Thus all the sinusoidal functions could be simplified using a second order Taylor expansion around zero. However, the values of $\sin(\phi_t)$ and $\cos(\phi_t)$ can be precomputed, stored and used when necessary, thus the linearization of the cant angle will not be applied.
A last simplification can be done observing that some of the terms in (2.35) and (2.36) are negligible. In particular $\frac{v^2}{R^2}$ has an order of 10^{-3} for very fast trains ($100m/s$) on rather tight curves ($1.5km$). The term $\frac{v}{R}$ has an order of $10^{-1} - 10^{-2}$ and is often multiplied to small other terms. The difference of inertia are often small, in particular the difference between the yaw inertia and the pitch inertia, $J_\chi - J_\psi$, is often close to zero. Angular speeds are all usually small, of the order between 10^{-3} and 10^{-6}, a part from the pitch component $\dot{\chi}$ of the wheel sets that is composed by the nominal speed $\frac{V}{r_0}$ and the spin perturbation β due to the oddly distributed forces on the wheels.
In the following the fictitious forces due to the nominal track geometries will be simplified due to the assumptions done for the car body, the bogie frames and the wheel sets. The subscript c will stand for centrifugal because of the biggest force component in the y direction. The car body and the bogie frames will have the simplified form:

$$\left\{ \begin{array}{c} {}^F\vec{F}_c \\ {}^B\vec{M}_c \end{array} \right\} = \left\{ \begin{array}{c} 0 \\ m\left[\frac{v^2}{R}\cos(\phi_t)\right] \\ m\left[-\frac{v^2}{R}\sin(\phi_t)\right] \\ 0 \\ 0 \\ 0 \end{array} \right\} \tag{2.41}$$

whereas the wheel sets will be affected also by the nominal rolling speed, thus:

$$\left\{ \begin{array}{c} {}^{F}\vec{F}_c \\ {}^{B}\vec{M}_c \end{array} \right\} = \left\{ \begin{array}{c} 0 \\ m\left[\frac{v^2}{R}\cos(\phi_t)\right] \\ m\left[-\frac{v^2}{R}\sin(\phi_t)\right] \\ J_\phi\left(-\dot{\chi}\frac{v}{R}\cos(\phi_t)\right) + (J_\chi - J_\psi)\left(-\dot{\chi}\frac{v}{R}\cos(\phi_t) + \dot{\chi}\dot{\psi}\right) \\ 0 \\ (J_\phi - J_\chi)\left(\dot{\phi}\dot{\chi}\right) \end{array} \right\} \tag{2.42}$$

The consideration of the rolling speed in these equation will give rise to precession forces that can influence the dynamics. Models with and without precession forces have been tested in order to asses the qualitative different behaviors.

2.2 Wheel-rail interaction

The key part of a model for train dynamics is hidden in the interactions between wheels and rails. In order to keep the wheels on the track, avoiding derailment, the design and the material used for these two components is very important. Modern train wheels have a conical design that is used in order to center automatically the wheel set thanks to the gravitational force. The forces acting here define the guidance of the wheel set. However, considering these forces in the model is not sufficient for explaining the hunting phenomenon, a sideway vibration that occurs after certain speeds. If no friction is considered, the rotation of the wheel corresponds to the same translation of the wheel set along the track. If friction is considered, this condition doesn't hold anymore and the rotational speed starts being smaller than the translational. This effect is called creepage and can explain the hunting phenomenon.
The relative lateral, vertical and yaw movement of the wheel set with respect to the track center-line will cause the movement of the contact points that in normal situations are one per wheel. However, in some cases multiple contact points can be present, making the problem harder to solve due to the forces being shared among all the points. Some geometrical properties and factors can be precomputed in a static setting and updated with the adjustments due to the dynamics. The RSGEO routine [KM10] is used in order to get the static parameters into a table that will be accessed during the execution of the program.

Figure 2.7: Wheel guidance due to gravitational load of the wheel set on the rails. The contact angle determines the amount of normal force that will act in the lateral direction, bringing the wheel set back to the central position.

2.2.1 Guidance forces

Several wheel set profiles have been proposed over the years in order to get a good guidance and low wear, but after almost two hundreds years of trains running, wheels with positive conicity have outperformed all their competitors. The conicity is the ratio

$$\lambda = \frac{\Delta r}{\Delta y}$$

where Δr is the radius change and Δy is the relative lateral displacement. The figure 2.7 shows an example of a conical wheel set on a canted rail. The cant of the rail is the inclination of the rails of a track toward its centerline, in order to increase the restoring guidance forces. This cant is distinguished from the track cant that is a rotation of the whole track plane, used for negotiating curves. The contact angle is the angle formed by the contact patch plane and the wheel set horizontal plane. If this angle is zero, the contact will be flat and the normal forces will be vertically loaded on the rail. The conical profile of the wheels and the profile of the rails are such that the contact planes for the left and right wheels will cause the normal forces to have a lateral component pointing toward the track center-line. Already in the 1880's, Klingel and Boedecker derived equations for the sinusoidal motion of a non-suspended conical wheel set[Kli82].

In a static condition the Normal force is easily computed knowing the mass of the bodies. Since the two bodies are considered elastic, in a dynamic condition, the penetration of the wheel set in the rail will change over time and the forces acting on the contact patch will change accordingly. The Hertz's contact theory[Her95] (dated 1881-82) can be used in order to find the shape of the contact patch.

This contact patch will be considered to be elliptic. Even if this condition is not always true, it represents a good approximation. The ellipse will be described by its major axis a and minor axis b. Using the work by Kalker [Kal90, Kal91], the normal force is proportional to the actual wheel-rail penetration:

$$N \propto q^{\frac{3}{2}} \tag{2.43}$$

and this penetration can be computed using the actual geometry of the bodies. The resulting normal force is given by

$$N = N_0 \left(1 + \frac{\Delta q}{q_0}\right)^{\frac{3}{2}} \tag{2.44}$$

where N_0 and q_0 are the static normal force and the static penetration, while Δq is the additional penetration due to the dynamical movement of the wheel set.
These additional penetrations for the left and right contact points are given by

$$\Delta q_l \approx -(a_{Rl} - y - a_l - \phi r_l)\sin(\delta_l + \phi) + (-z - \phi a_l)\cos(\delta_l + \phi) \tag{2.45}$$

$$\Delta q_r \approx (-a_{Rr} - y + a_r - \phi r_r)\sin(\delta_r + \phi) + (-z + \phi a_r)\cos(\delta_r - \phi) \tag{2.46}$$

where $a_{Rl/r}$ is the lateral distance of the contact point on the rail, $a_{l/r}$ is the lateral distance of the contact point on the wheel, $r_{l/r}$ is the actual rolling radius. The derivation of the additional penetration is the same used by Petersen-Hoffmann [HP02]. The knowledge of the magnitude of the normal vector and the inclination of the contact plane allows the calculation of the normal forces acting on the contact point:

$${}^F\vec{F}_{N_l} = \left\{ \begin{array}{c} 0 \\ -N_l \sin(\delta_l + \phi) \\ N_l \cos(\delta_l + \phi) \end{array} \right\} \tag{2.47}$$

$${}^F\vec{F}_{N_r} = \left\{ \begin{array}{c} 0 \\ N_r \sin(\delta_r - \phi) \\ N_r \cos(\delta_r - \phi) \end{array} \right\} \tag{2.48}$$

2.2.2 Creep forces

The forces due to the slip between the wheels and the rails are really important for the dynamic stability of the vehicle [ABS07, Ch. 8.4]. If the two bodies were considered stiff bodies, the Coulomb's friction law would be sufficient for explaining the the friction forces. However the bodies cannot be considered stiff, but have to be considered elastic, thus the mutual penetration of the bodies combined with the rolling, produces creep forces. The best description of these

forces is given by Kalker in [Kal90], but an exact computation of the forces, provided by the routine CONTACT[KV], is computationally very expensive. So the forces will be approximated by the Shen, Hedrick and Elkins (SHE) non-linear theory[SHE84].
In order to use any of the theories available, the relative motion between the bodies has to be found. The relative speed normalized by the speed of the vehicle is called creepage and can be divided in three components in the contact point reference system: the longitudinal, the lateral and the spin component. Here the creepages are listed, a complete derivation can be found in the work by Petersen-Hoffmann [HP02]. In the following $r_{l/r}$ is the rolling radius on the left and the right wheels, $a_{l/r}$ is the lateral distance of the left and right contact points from the center of mass of the wheel set, $\delta_{l/r}$ is the contact angle on the left and right wheels, f_b is the longitudinal distance of the two wheel sets in a bogie frame, w_b is the width of a wheel set, $\Omega = \frac{v}{r_0} + \beta$ is the angular velocity of the wheel set, r_0 is the basic rolling radius. The creepages are denoted by $\xi_\bullet$, where the first position in the subscript is reserved for referring to the leading (f) or trailing (r) wheel sets, the second is for the wheel side (left or right) and the last is for the direction of the creepage (longitudinal/lateral/spinning).

$$
\begin{aligned}
\xi_{flx} &= \left[V + \left(\psi + \frac{f_b}{R}\right)\dot{y} + r_l\left(\left(\psi + \frac{f_b}{R}\right)\dot{\phi} - \Omega\right) - a_l\dot{\psi} + \frac{aV}{R}\cos\phi_t\right]/V \\
\xi_{frx} &= \left[V + \left(\psi + \frac{f_b}{R}\right)\dot{y} + r_r\left(\left(\psi + \frac{f_b}{R}\right)\dot{\phi} - \Omega\right) + a_r\dot{\psi} - \frac{aV}{R}\cos\phi_t\right]/V \\
\xi_{fly} &= \left[\left(-V\left(\psi + \frac{f_b}{R}\right)\left(1 + \frac{w_b}{R}\cos\phi_t\right) + \phi\dot{z} + \dot{y} + \dot{\phi}r_l\right)\cos\delta_l + \right. \\
&\quad \left. + \left(-\phi\dot{y} + \dot{z} + a_l\dot{\phi}\right)\sin\delta_l\right]/V \\
\xi_{fry} &= \left[\left(-V\left(\psi + \frac{f_b}{R}\right)\left(1 - \frac{w_b}{R}\cos\phi_t\right) + \phi\dot{z} + \dot{y} + \dot{\phi}r_r\right)\cos\delta_r - \right. \\
&\quad \left. - \left(-\phi\dot{y} + \dot{z} + a_r\dot{\phi}\right)\sin\delta_r\right]/V
\end{aligned}
$$

$$
\begin{aligned}
\xi_{rlx} &= \left[V + \left(\psi - \frac{f_b}{R}\right)\dot{y} + r_l\left(\left(\psi - \frac{f_b}{R}\right)\dot{\phi} - \Omega\right) - a_l\dot{\psi} + \frac{aV}{R}\cos\phi_t\right]/V \\
\xi_{rrx} &= \left[V + \left(\psi - \frac{f_b}{R}\right)\dot{y} + r_r\left(\left(\psi - \frac{f_b}{R}\right)\dot{\phi} - \Omega\right) + a_r\dot{\psi} - \frac{aV}{R}\cos\phi_t\right]/V \\
\xi_{rly} &= \left[\left(-V\left(\psi - \frac{f_b}{R}\right)\left(1 + \frac{w_b}{R}\cos\phi_t\right) + \phi\dot{z} + \dot{y} + \dot{\phi}r_l\right)\cos\delta_l + \right. \\
&\quad \left. + \left(-\phi\dot{y} + \dot{z} + a_l\dot{\phi}\right)\sin\delta_l\right]/V
\end{aligned}
$$

$$\xi_{rry} = \left[\left(-V\left(\psi - \frac{f_b}{R}\right)\left(1 - \frac{w_b}{R}\cos\phi_t\right) + \phi\dot{z} + \dot{y} + \dot{\phi} r_r \right)\cos\delta_r - \right.$$
$$\left. - \left(-\phi\dot{y} + \dot{z} + a_r\dot{\phi}\right)\sin\delta_r \right] / V$$

$$\xi_{fls} = \left[-\left(\Omega - \left(\psi + \frac{f_b}{R}\right)\dot{\phi}\right)\sin\delta_l + \dot{\psi}\cos\delta_l \right] / V$$
$$\xi_{frs} = \left[\left(\Omega - \left(\psi + \frac{f_b}{R}\right)\dot{\phi}\right)\sin\delta_r + \dot{\psi}\cos\delta_r \right] / V$$
$$\xi_{rls} = \left[-\left(\Omega - \left(\psi - \frac{f_b}{R}\right)\dot{\phi}\right)\sin\delta_l + \dot{\psi}\cos\delta_l \right] / V$$
$$\xi_{rrs} = \left[\left(\Omega - \left(\psi - \frac{f_b}{R}\right)\dot{\phi}\right)\sin\delta_r + \dot{\psi}\cos\delta_r \right] / V$$

Using Kalker's linear theory the forces along the contact plane are

$$\tilde{\mathbf{F}}_\tau = \left\{ \begin{array}{c} -abGC_{11}\xi_x \\ -abG\left(C_{22}\xi_y + \sqrt{ab}C_{23}\xi_s\right) \\ 0 \end{array} \right\} \tag{2.49}$$

where C_{11}, C_{22} and C_{23} are the Kalker's coefficients, a and b are the major and minor axis of the contact ellipse, G[1] is the shear modulus. The τ notation indicates that these forces are written in the contact reference system. The creep forces depend on the size of the contact ellipse, which depends on the position of the contact point and the dynamics that change the normal forces. Thus the creep forces are adjusted using the Shen, Hedrick and Elkins (SHE) non-linear theory:

$$|\hat{\mathbf{F}}_\tau| = \begin{cases} \mu N \left(\left[\frac{|\tilde{\mathbf{F}}_\tau|}{\mu N}\right] - \frac{1}{3}\left[\frac{|\tilde{\mathbf{F}}_\tau|}{\mu N}\right]^2 + \frac{1}{27}\left[\frac{|\tilde{\mathbf{F}}_\tau|}{\mu N}\right]^3 \right) & \text{if } \frac{|\tilde{\mathbf{F}}_\tau|}{\mu N} < 3 \\ \mu N & \text{if } \frac{|\tilde{\mathbf{F}}_\tau|}{\mu N} \geq 3 \end{cases} \tag{2.50}$$

$$\epsilon = \frac{|\hat{\mathbf{F}}_\tau|}{|\tilde{\mathbf{F}}_\tau|} \tag{2.51}$$

$$\mathbf{F}_\tau = \left\{ \begin{array}{c} \epsilon\tilde{F}_x \\ \epsilon\tilde{F}_y \\ 0 \end{array} \right\} \tag{2.52}$$

[1] $G = \frac{E}{2\cdot(1+\upsilon)}\,\frac{N}{m^2}$ where E is the Young's modulus and υ is the Poisson's ratio.

where μ is the friction coefficient[2]. Now the creep forces can be rewritten in the wheel set reference frame using the rotation matrices (see Appendix B).

$$ {}^F\vec{F}_{C_l} = {}^F\mathbf{T}^W\, {}^W\mathbf{T}^{C_l}\mathbf{F}_\tau \tag{2.53} $$

$$ {}^F\vec{F}_{C_r} = {}^F\mathbf{T}^W\, {}^W\mathbf{T}^{C_r}\mathbf{F}_\tau \tag{2.54} $$

2.2.3 Multiple contact points

It's often the case that particular combinations of wheel and rail profiles result in the appearance of multiple contact points for different displacements and yaw angles of the wheel set. It is also usual that worn wheels and rails increase the number of contact points. Thus a framework for addressing them is necessary in order to compute realistic values of contact forces. One approach is to approximate the two patches with a unique patch[PS91]. Another approach is to consider every single contact point separately when it appears. It is obvious that the load will be split among the two contact points unevenly. However, the static load on each of them can be computed as well as the patch geometry and location. In the following the subscript $i \in [1, \dots, n_l]$ will be used to address the forces and torques of the ith contact point on the left wheel of a wheel set. The subscript $j \in [1, \dots, n_r]$ will be used for the right wheel. Kalker's theories and Shen, Hedrick and Elkins (SHE) non-linear theory apply for each contact point, thus the forces on a contact point are given by

$$ {}^F\vec{F}_{L_i} = {}^F\vec{F}_{C_l,i} + {}^F\vec{F}_{N_l,i} \qquad \text{for } i \in [1, \dots, n_l] \tag{2.55} $$

$$ {}^F\vec{F}_{R_j} = {}^F\vec{F}_{C_r,j} + {}^F\vec{F}_{N_r,j} \qquad \text{for } j \in [1, \dots, n_r] \tag{2.56} $$

Now the torques on the wheel set can be computed for each contact point depending on the position of the patch with respect to the center of mass of the wheel set. The lateral distance will be denoted by $a_{l,i}$ and $a_{r,i}$ for the left and the right wheels respectively. The rolling radius on each contact point will be denoted by $r_{l,i}$ and $r_{r,i}$. Furthermore, the actuall roll ϕ and the yaw ψ of the wheel set is needed. For left contact points:

$$ {}^B M_{L_i} = \left\{ \begin{array}{c} a_{l,i}\left(\left\{{}^F\vec{F}_{L_i}\right\}_z - \left\{{}^F\vec{F}_{L_i}\right\}_y \phi\right) \\ -r_{l,i}\left(\left\{{}^F\vec{F}_{L_i}\right\}_x + \left\{{}^F\vec{F}_{L_i}\right\}_y \psi\right) \\ -a_{l,i}\left(\left\{{}^F\vec{F}_{L_i}\right\}_x + \left\{{}^F\vec{F}_{L_i}\right\}_y \psi\right) \end{array} \right\} \tag{2.57} $$

[2] μ has been chosen to be 0.15 for this work. This value was chosen in order to obtain results that are comparable with previous results obtained with the same model. In the industry, friction coefficients between $[0.30, 0.50]$ are adopted for safety reasons.

For right contact points:

$$
{}^{B}M_{R_j} = \left\{ \begin{array}{c} -a_{r,j}\left(\left\{{}^{F}\vec{F}_{R_j}\right\}_z - \left\{{}^{F}\vec{F}_{R_j}\right\}_y \phi\right) \\ -r_{r,j}\left(\left\{{}^{F}\vec{F}_{R_j}\right\}_x + \left\{{}^{F}\vec{F}_{R_j}\right\}_y \psi\right) \\ a_{r,j}\left(\left\{{}^{F}\vec{F}_{R_j}\right\}_x + \left\{{}^{F}\vec{F}_{R_j}\right\}_y \psi\right) \end{array} \right\} \tag{2.58}
$$

2.3 Suspension modeling

The primary and secondary suspensions have the fundamental role of guiding the wheel sets on the track and damping the vibrations introduced by the track irregularities and the contact forces. So, the selection of proper suspensions is important for both ride comfort, safety and wear, of both track and vehicle components. A suspension system is formed by a number of elements: link dampers, shear springs, bumpstops, air springs, rubber bushings etc. The reaction force of these elements is usually connected to the relative displacement and speed of the two attack points, the relative rotation and angular velocity of the two bodies. Thus, a first step for computing the forces due to the suspension is to find the relative displacement, speed and angles of the attack points of each element of the suspension system. When these quantities are found and the constitutive law for the suspension element is known, the reaction forces for each element can be computed.

2.3.1 Suspension geometry

In this section the displacement, speed, rotation and angular velocity of the bodies will be used in order to find the relative movement of the attack points belonging to a component of the suspension system. In the following, a generic element that belongs to a suspension system connecting two bodies through two attack points on their surface, will be called **Link**. The position of the two attack points will be provided with respect to the reference frame centered in the center of mass of the body they belong to. The notation ${}^{R_1}\vec{r}_l^{R_2} = \left\{x_l^R, y_l^R, z_l^R\right\}^T$ indicates the attack point of the link in the component C with reference frame R_2 written in the reference frame R_1. In order to compare the positions of the two attack points, they have to be written in the track following reference system F, obtaining the vectors ${}^{F}\vec{r}_l^{R1}$ and ${}^{F}\vec{r}_l^{R2}$, where $R1$ and $R2$ are the reference frames attached to the two bodies. The nominal position in the track following system of the center of mass of a body will be denoted by ${}^{F}\vec{r}_0^{R} = \{l, b, h\}^T$. Figure 2.8

shows the vector notation used in the case of a system of two components. As it is illustrated, the vectors ${}^{R1}\vec{r}_l^{R1}$ and ${}^{R1}\vec{r}_l^{R2}$, that express the position of the attack points w.r.t. the center of mass of the body, are constant. This is true under the assumption of rigid bodies. An additional displacement vector could be added in order to take in account flexible bodies.

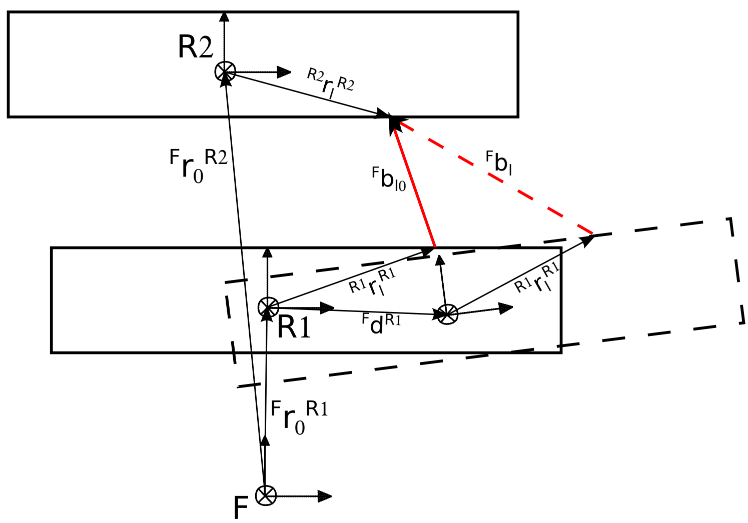


Figure 2.8: Vector notation for the center of mass, the attack points and the body displacements. The link length is modified due to a translation and a rotation of the component C1. In order to keep the drawing simple, the component C2 has not been moved, however the notation would be similar to the component C1. The vectors r_0 are the vectors between the center of the system (track centerline) and the center of mass of the components in their rest positions. Vector d represent a displacement of the center of mass of a component. Vectors r_l are the vectors from the center of mass of a component and the attack point of link l on the surface of that component. b_{l_0} is the vector between the two attack point of a suspension element (link) when the two connected components are in a rest position. Vector b_l represents the deformed length of the link when the connected components are displaced. The left superscript indicates the reference frame in which the vector is written and the right super script indicates on which component the vector ends.

In order to obtain the actual position of a point on a component, the displacement of its center of mass and the rotation angles of the body are needed. For dampers also the speed of the point is needed, so the speed of the center of mass with respect to the track following reference system and the angular velocity of the body are needed as well. Thus, displacement vectors are defined for the wheel sets (W), the bogie frames (B) and the car body (C):

$$ {}^F\vec{d}^W = \left\{ \begin{array}{c} x^W \\ y^W \\ z^W \end{array} \right\} \quad {}^F\vec{d}^B = \left\{ \begin{array}{c} x^B \\ y^B \\ z^B \end{array} \right\} \quad {}^F\vec{d}^C = \left\{ \begin{array}{c} x^C \\ y^C \\ z^C \end{array} \right\} \tag{2.59} $$

$$ {}^F\dot{\vec{d}}^W = \left\{ \begin{array}{c} \dot{x}^W \\ \dot{y}^W \\ \dot{z}^W \end{array} \right\} \quad {}^F\dot{\vec{d}}^B = \left\{ \begin{array}{c} \dot{x}^B \\ \dot{y}^B \\ \dot{z}^B \end{array} \right\} \quad {}^F\dot{\vec{d}}^C = \left\{ \begin{array}{c} \dot{x}^C \\ \dot{y}^C \\ \dot{z}^C \end{array} \right\} \tag{2.60} $$

The transformation matrices for passing from a body reference system to the track following reference system are listed in Appendix B.
Let's now consider the link l, in figure 2.8, between the components C1 and C2 with reference frames $R1$ and $R2$. The positions of the connecting points with respect to the Track Following reference frame are given by

$$ {}^F\vec{r}_l^{R1} = {}^F\vec{r}_0^{R1} + {}^F\vec{d}^{R1} + {}^F\mathbf{T}^{R1} \cdot {}^{R1}\vec{r}_l^{R1} \tag{2.61} $$

$$ {}^F\vec{r}_l^{R2} = {}^F\vec{r}_0^{R2} + {}^F\vec{d}^{R2} + {}^F\mathbf{T}^{R2} \cdot {}^{R2}\vec{r}_l^{R2} \tag{2.62} $$

In order to find the change in position of the attack points, derivatives of (2.61) and (2.62) have to be taken. Since the bodies of the components are considered rigid, the vector ${}^R\vec{r}_l^R$ doesn't change with time for all R and l, making a term of the derivative to drop out.

$$ {}^F\dot{\vec{r}}_l^{R1} = {}^F\dot{\vec{d}}^{R1} + {}^F\dot{\mathbf{T}}^{R1} \cdot {}^{R1}\vec{r}_l^{R1} \tag{2.63} $$

$$ {}_F\dot{\vec{r}}_l^{R2} = {}^F\dot{\vec{d}}^{R2} + {}^F\dot{\mathbf{T}}^{R2} \cdot {}^{R2}\vec{r}_l^{R2} \tag{2.64} $$

All the springs and dampers will be characterized by a length, a speed and a torsion at rest. The speed and torsion is usually zero at rest. Instead, the length has to be computed using the geometrical position of the attack points when the link is unloaded. Here the length will not be given as an Euclidean distance, because for some spring systems, the deformations in the three directions will be considered independent. Instead the distance vector between the two attack points at rest will be considered. The position vectors ${}^F\vec{r}_{l_0}^{R1}$ and ${}^F\vec{r}_{l_0}^{R2}$ of the attack points, w.r.t. the track following reference frame, can be found using (2.61) and (2.62) where the displacement vectors are ${}^F\vec{d}^{R1} = {}^F\vec{d}^{R2} = \{0\}$ and the rotation matrices are ${}^FT^{R1} = {}^FT^{R2} = [\mathbf{I}]$. The distance vector (R1-R2) at rest will be given by

$$ {}^F\vec{b}_{l_0} = {}^F\vec{r}_{l_0}^{R2} - {}^F\vec{r}_{l_0}^{R1} \tag{2.65} $$

Now let's assume that the forces acting on the components C1 has to be computed (see fig. 2.8). The displaced distance vector will be given by

$$ {}^{F}\vec{b}_l = {}^{F}\vec{r}_l^{R2} - {}^{F}\vec{r}_l^{R1} \tag{2.66} $$

Using the same approach also the relative speed of the two attack point can be found:

$$ {}^{F}\vec{v}_l = {}^{F}\dot{\vec{r}}_l^{R2} - {}^{F}\dot{\vec{r}}_l^{R1} \tag{2.67} $$

Also the angle and the angular velocities could be considered in some elements - for example in air springs that can buckle. These quantities are obtained as:

$$ {}^{F}\vec{\theta}_l = {}^{F}\vec{\theta}^{R2} - {}^{F}\vec{\theta}^{R1} = \left\{ \begin{array}{c} \phi^{R2} \\ \chi^{R2} \\ \psi^{R2} \end{array} \right\} - \left\{ \begin{array}{c} \phi^{R1} \\ \chi^{R1} \\ \psi^{R1} \end{array} \right\} \tag{2.68} $$

$$ {}^{F}\dot{\vec{\theta}}_l = {}^{F}\dot{\vec{\theta}}^{R2} - {}^{F}\dot{\vec{\theta}}^{R1} = \left\{ \begin{array}{c} \dot{\phi}^{R2} \\ \dot{\chi}^{R2} \\ \dot{\psi}^{R2} \end{array} \right\} - \left\{ \begin{array}{c} \dot{\phi}^{R1} \\ \dot{\chi}^{R1} \\ \dot{\psi}^{R1} \end{array} \right\} \tag{2.69} $$

The reaction force and moments due to the buckling of the link will be given by a function f, representing the constitutive law of the element:

$$ \left\{ \begin{array}{c} {}^{F}\vec{F}_l \\ {}^{F}\vec{T}_l \end{array} \right\} = f\left({}^{F}\vec{b}_{l_0},\ {}^{F}\vec{b}_l,\ {}^{F}\vec{v}_l,\ {}^{F}\vec{\theta}_l,\ {}^{F}\dot{\vec{\theta}}_l \right) \tag{2.70} $$

The torque applied to the center of mass will not depend only on the buckling of the links, but also on the directions of the forces that are not passing through the center of mass. This torque component is obtained using the geometrical position of the attack points on the body. The forces have to be written in the reference frame of the body they are applied to using the proper transformation matrix:

$$ {}^{R1}\vec{F}_l^{R1} = {}^{R1}\mathbf{T}^{F}\,{}^{F}\vec{F}_l^{R1} \tag{2.71} $$

This component of the torque is thus given by the cross product

$$ {}^{R1}\vec{\tau}_l = {}^{R1}\vec{r}_l^{R1} \times {}^{R1}\vec{F}_l^{R1} \tag{2.72} $$

So the total forces and torques are finally given by

$$ \left\{ \begin{array}{c} {}^{F}\vec{F}_l \\ {}^{F}\vec{M}_l \end{array} \right\} = \left\{ \begin{array}{c} {}^{F}\vec{F}_l \\ {}^{F}\vec{T}_l \end{array} \right\} + \left\{ \begin{array}{c} 0 \\ {}^{R1}\vec{\tau}_l \end{array} \right\} \tag{2.73} $$

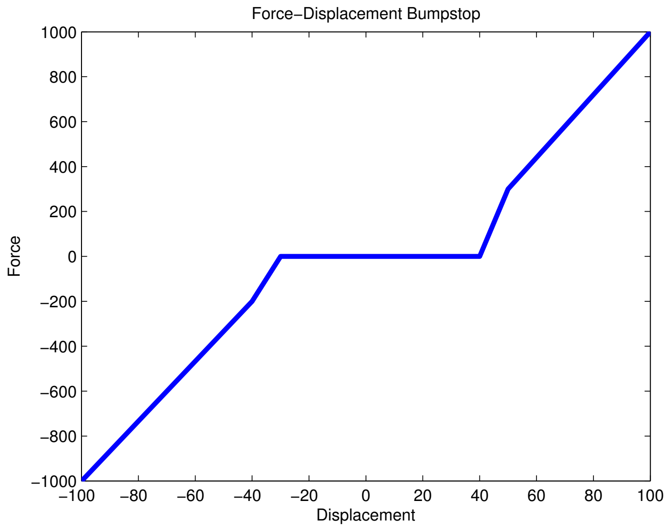


Figure 2.9: Example of Force-Displacement function for a Bumpstop. For displacements in the clearance region [-35,40], the force is zero. A sharp increase of the stiffness happens as soon as the bumpstop is hit.

2.3.2 Suspension components

Now that the motion of a pair of attack points has been defined, the force due to a component of the suspension system can be found depending on its force law. In reality, most of the suspension components don't behave only as a spring or a damper, but as a combination of them. In order to simplify the model, in the following the springs and the dampers will never be considered in series. Having springs and dampers in parallel will not constitute a problem because the resulting force will simply be the sum of the forces. Springs and dampers are governed by the displacement-force and speed-force functions respectively. In many situations the components can be considered to operate on their linear part, however, in real applications the non-linear part of the functions has to be taken into account. The presence of non-linear stiffness and damping functions forbids the usage of the superposition principle, adding some computational work during the calculation of the forces. Figure 2.9 shows a non-linear function example. This function belongs to a bumpstop. The slope of the curve represents the stiffness. It's clear that there is a fast increase in stiffness as soon as the clearance of the bumpstop is covered and the two bodies hit each other. The bumpstop will be discussed more in details in later sections, but it is important to note that such a sudden change in stiffness can force the numerical solver to take very small steps in order to follow the rapidly changing dynamics in this situation. Bumpstops are not the only elements with non-linear behaviors: springs and dampers can have non-linear responses to displacement and speed. In order to reduce the computational effort, the change in stiffness/damping ratio can be smoothed using interpolation functions such as polynomial interpolation, cubic splines or Akima[Aki70] interpolation. Another approach to non-linear elements is to implement an event detection framework, in order to block the integration just right at the time when the sudden change in stiffness/damping happens. This approach has been already applied to non-smooth dynamics related to railway vehicles[Hof06].

Vector springs and dampers

The reaction force due to a vector spring or damper acts in the direction of the link.
In spring elements, the deformation is given by the difference of the euclidean distance between the two attack points at rest and at the deformed state. Given $k : \mathbb{R} \to \mathbb{R}$, the stiffness function associated to the spring, the force magnitude is given by

$$\| {}^F\vec{F}_k \| = k \left(\| {}^F\vec{b_l} \| - \| {}^F\vec{b_{l_0}} \| \right) \tag{2.74}$$

This is then projected on the direction of the link vector

$${}^F\vec{F}_k = \| {}^F\vec{F}_k \| \cdot \frac{{}^F\vec{b_l}}{\| {}^F\vec{b_l} \|} \tag{2.75}$$

For dampers, the relative speed of each of the attack points is used in order to compute the reaction force. Given $d : \mathbb{R} \to \mathbb{R}$, the damping function associated to the damper, the force magnitude is given by

$$\| {}^F\vec{F}_d \| = d \left(\| {}^F\vec{v_l} \| \right) \tag{2.76}$$

This force is applied in the direction of the link, thus

$${}^F\vec{F}_d = \| {}^F\vec{F}_d \| \cdot \frac{{}^F\vec{b_l}}{\| {}^F\vec{b_l} \|} \tag{2.77}$$

Longitudinal, lateral, vertical springs and dampers

These kinds of springs and dampers work only in one direction. They can be combined together in order to compose more complex elements or in order to build simplified models. The principle is the same as that of the vector springs and dampers, but now the Euclidean distance is not used anymore. Given $k : \mathbb{R} \rightarrow \mathbb{R}$, the stiffness function associated to the spring, the force vector is given by

$$ {}^F\vec{F}_k = k\left(\left\{ {}^F\vec{b_l} \right\}_i - \left\{ {}^F\vec{b_{l_0}} \right\}_i\right) \cdot \mathbf{e}_i \tag{2.78} $$

where $i \in [x, y, z]$ is the index corresponding to one of the directions, the notation $\{\vec{a}\}_i$ means that the i component of the vector $\vec{a}$ is taken and $\mathbf{e}_i$ is the corresponding base vector.
Likewise, dampers act only in one of the directions of the relative speed of the attack points. Given $d : \mathbb{R} \rightarrow \mathbb{R}$, the stiffness function associated to the spring, the force vector is given by

$$ {}^F\vec{F}_d = d\left(\left\{ {}^F\vec{v_l} \right\}_i\right) \cdot \mathbf{e}_i \tag{2.79} $$

Flexicoil springs

Flexicoil springs are coil springs that act in the direction of their axis as well as in the plane with the axis as the normal. In train design, coil springs are usually placed in the vertical direction, however they contribute in counteracting also horizontal displacements. In the vertical direction the shear spring can be considered as a vertical spring, so given $k_z : \mathbb{R} \rightarrow \mathbb{R}$ the stiffness function, the force vector is given by

$$ {}^F\vec{F}_k^z = k_z\left(\left\{ {}^F\vec{b_l} \right\}_z - \left\{ {}^F\vec{b_{l_0}} \right\}_z\right) \cdot \mathbf{e}_z \tag{2.80} $$

In the horizontal plane, the spring can be considered a vector spring in two dimensions. Thus the force in the horizontal plane will be projected on the deformation vector. Since the shear spring is supposed to be placed in vertical position, $\left\{ {}^F\vec{b_{l_0}} \right\}_x = \left\{ {}^F\vec{b_{l_0}} \right\}_y = 0$. So, given $k_{xy} : \mathbb{R} \rightarrow \mathbb{R}$, the stiffness function in the horizontal plane, and

$$ {}^F\vec{b}_l^{xy} = \left\{ \begin{array}{c} \left\{ {}^F\vec{b_l} \right\}_x \\ \left\{ {}^F\vec{b_l} \right\}_y \\ 0 \end{array} \right\} \tag{2.81} $$

the deformation vector on the horizontal plane, the magnitude of the force is given by

$$ \| {}^F\vec{F}_k^{xy} \| = k_{xy}\left(\left\| {}^F\vec{b}_l^{xy} \right\|\right) \tag{2.82} $$

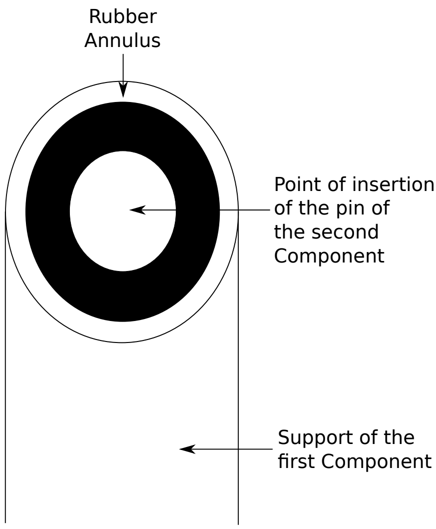


Figure 2.10: Conical rubber bush and mounting points.

This force is then projected in the direction of the deformation vector, obtaining

$$^{F}\vec{F}_k^{xy} = \|\vec{F}_k^{xy}\| \cdot \frac{^{F}\vec{b}_l^{xy}}{\left\| ^{F}\vec{b}_l^{xy} \right\|} \tag{2.83}$$

Finally the reaction force due to the deformation of a vertical shear spring is

$$^{F}\vec{F}_k = \vec{F}_k^{z} + \vec{F}_k^{xy} \tag{2.84}$$

Rubber Bushings

A rubber bushing (see fig. 2.10) is usually a conical element mounted in one of the components that have to be connected. The other component is provided with a pin that is inserted into the conical rubber bush. The movement of each components is counteracted by the damping and stiffness properties of the rubber. Also here the link element connecting two attack points can be used in order to determine the relative movement of the pin in the three directions. The same attack point is used for the two components, so that the length of the link at rest is zero. If the pin and the center of the rubber bush don't coincide then a reaction force will be produced by the rubber in order to bring the pin back to the center. The rubber used here has damping and stiffness properties independent of each other. So the rubber bushing can be modeled by three springs and three dampers in the three directions (longitudinal, lateral and vertical). Given $k_x, k_y, k_z : \mathbb{R} \to \mathbb{R}$ and $d_x, d_y, d_z : \mathbb{R} \to \mathbb{R}$, the reaction forces, due to the relative displacement ${}^F\vec{b}_l$ and speed ${}^F\vec{v}_l$ between the pin and the center of the rubber bushing, is given by

$$ {}^F\vec{F}_k = \left\{ \begin{array}{c} k_x\left(\left\{{}^F\vec{b}_l\right\}_x\right) \\ k_y\left(\left\{{}^F\vec{b}_l\right\}_y\right) \\ k_z\left(\left\{{}^F\vec{b}_l\right\}_z\right) \end{array} \right\} \tag{2.85} $$

$$ {}^F\vec{F}_d = \left\{ \begin{array}{c} d_x\left(\left\{{}^F\vec{v}_l\right\}_x\right) \\ d_y\left(\left\{{}^F\vec{v}_l\right\}_y\right) \\ d_z\left(\left\{{}^F\vec{v}_l\right\}_z\right) \end{array} \right\} \tag{2.86} $$

Bumpstops

In order to avoid big displacements some rigid bodies are added to the model. When these rigid bodies hit each other, they behave like very stiff springs that counteract the big displacement. Thus the best way of modeling bumpstops is to consider a piecewise linear spring that connects two attach points on the components. The Bumpstops are usually placed in orthogonal directions, so a longitudinal, lateral or vertical spring can be used. Figure 2.9 shows the typical force displacement function of an asymmetric bumpstop: a sudden stiffness increase happens as soon as all the clearance is covered and the two part of the bumpstop hit each other.

2.4 Equations of motion

In the previous sections all the forces that characterize the dynamics of the railway vehicle have been introduced. A lot of components and suspension systems are involved, so they need a unique way to be identified. Two kinds of right superscripts will be used in the following: j is dedicated to the bodies identification and i is dedicated to the suspension identification.

$$ j \in \{C, B_l, B_t, W_{ll}, W_{lt}, W_{tl}, W_{tt}\} $$

$$ i \in \{SS_l, SS_t, PS_{ll}, PS_{lt}, PS_{tl}, PS_{tt}\} $$

The symbols C, B and W stand for Car body, Bogie frame and Wheel set. For instance, the notation W_{lt} reads: the trailing wheel set attached to the leading bogie frame. The symbols SS and PS stand for Secondary Suspension and Primary Suspension respectively. The notation PS_{tl} reads: the leading Primary Suspension attached to the trailing bogie frame. Furthermore each force and torque will be denoted by a left superscript meaning that it is written either in the track following reference frame F or in the body following reference frame B.

The forces introduced in the previous section will be summarized here, before using them in order to construct the the equations of motion for each body of the model.

- ${}^F\vec{F}_g^j$ and ${}^B\vec{M}_g^j$: the gravitational forces apply to each of the bodies in the model.
- ${}^F\vec{F}_c^j$ and ${}^B\vec{M}_c^j$: the centrifugal forces due to the geometry of the track and the speed of the vehicle apply to all the bodies in the model.
- ${}^F\vec{F}_s^i$ and ${}^B\vec{M}_s^i$: the suspension forces include the forces due to springs and dampers in the primary and secondary suspension.
- ${}^F\vec{F}_{N_l,i}^j$, ${}^F\vec{F}_{N_r,i}^j$: the Normal forces on the wheel-rail contact point i.
- ${}^F\vec{F}_{C_l,i}^j$, ${}^F\vec{F}_{C_r,i}^j$: the Creep forces on the wheel-rail contact point i.
- ${}^F\vec{F}_{L_k}^j = {}^F\vec{F}_{N_l,k}^j + {}^F\vec{F}_{C_l,k}^j$, ${}^B\vec{M}_{L_k}^j$: contact forces and torques of the kth contact point on the left wheel.
- ${}^F\vec{F}_{R_k}^j = {}^F\vec{F}_{N_r,k}^j + {}^F\vec{F}_{C_r,k}^j$, ${}^B\vec{M}_{R_k}^j$: contact forces and torques of the ith contact point on the right wheel.

The equations of motion are built up using the Newton-Euler formulation, thus for each body there would be six second order differential equations. However the assumption of negligible longitudinal displacement was made in section 2.1.3, thus each body can be described by five second order differential equations, that can be rewritten as ten first order differential equation to be solved. The wheel sets, for which the pitch angle is of no significance, will be described by four second order differential equations and one first order ODE representing the spin perturbation.

2.4.1 Car Body

The car body is connected by the Secondary Suspensions to two bogie frames. In addition its motion is affected by the gravitational and the centrifugal forces.

Thus,

$$m\ddot{\vec{x}} = {}^F\vec{F}_g^C + {}^F\vec{F}_c^C + {}^F\vec{F}_s^{SS_l} + {}^F\vec{F}_s^{SS_t} \tag{2.87}$$

$$[J]\dot{\vec{\omega}} = {}^B\vec{M}_g^C + {}^B\vec{M}_c^C + {}^B\vec{M}_s^{SS_l} + {}^B\vec{M}_s^{SS_t} \tag{2.88}$$

2.4.2 Bogie Frame

Each of the bogie frame (leading and trailing) are connected through the Secondary Suspensions to the car body and by the Primary Suspensions to two wheel set. In addition the motion of the bogie frames is affected by the gravitational and centrifugal forces. For the leading bogie frame the equations of motion are

$$m\ddot{\vec{x}} = {}^F\vec{F}_g^{B_l} + {}^F\vec{F}_c^{B_l} + {}^F\vec{F}_s^{SS_l} + {}^F\vec{F}_s^{PS_{ll}} + {}^F\vec{F}_s^{PS_{lt}} \tag{2.89}$$

$$[J]\dot{\vec{\omega}} = {}^B\vec{M}_g^{B_l} + {}^B\vec{M}_c^{B_l} + {}^B\vec{M}_s^{SS_l} + {}^B\vec{M}_s^{PS_{ll}} + {}^B\vec{M}_s^{PS_{lt}} \tag{2.90}$$

Likewise, the equations of motion of the trailing bogie frame are

$$m\ddot{\vec{x}} = {}^F\vec{F}_g^{B_t} + {}^F\vec{F}_c^{B_t} + {}^F\vec{F}_s^{SS_t} + {}^F\vec{F}_s^{PS_{tl}} + {}^F\vec{F}_s^{PS_{tt}} \tag{2.91}$$

$$[J]\dot{\vec{\omega}} = {}^B\vec{M}_g^{B_t} + {}^B\vec{M}_c^{B_t} + {}^B\vec{M}_s^{SS_t} + {}^B\vec{M}_s^{PS_{tl}} + {}^B\vec{M}_s^{PS_{tt}} \tag{2.92}$$

2.4.3 Wheel set

The wheel set components are characterized by a rotational motion around the y axis. The total angle covered by this rotation during a simulation corresponds (a part from the missing distance due to the creepage) to the distance covered by the vehicle. Since this is a value we are not interested in and it is not needed for any other computation, the equation of motion relative to χ is not considered. Instead the equation of motion relative to $\dot{\chi}$ is kept because it expresses the angular velocity of the wheel set, that can change due to the different rolling radius for the left and the right wheel set. This quantity is composed by the nominal speed $\frac{V}{r_0}$ and the difference of speed due to the uneven distribution of the forces on each wheel, denoted by β. For the leading wheel set in the leading bogie frame, the equations of motion are given by

$$m\ddot{\vec{x}} = {}^F\vec{F}_g^{W_{ll}} + {}^F\vec{F}_c^{W_{ll}} + \sum_{i\in[1\ldots n_l]} {}^F\vec{F}_{L_i}^{W_{ll}} + \sum_{i\in[1\ldots n_r]} {}^F\vec{F}_{R_i}^{W_{ll}} + {}^F\vec{F}_s^{PS_{ll}} \tag{2.93}$$

$$\begin{aligned} J_\phi\ddot{\phi} = & \sum_{i\in[1\ldots n_l]} \left\{ {}^B M_{L_i}^{W_{ll}} \right\}_\phi + \sum_{j\in[1\ldots n_r]} \left\{ {}^B M_{R_j}^{W_{ll}} \right\}_\phi + \\ & \left\{ {}^B\vec{M}_g^{W_{ll}} \right\}_\phi + \left\{ {}^B\vec{M}_c^{W_{ll}} \right\}_\phi + \left\{ {}^B\vec{M}_s^{PS_{ll}} \right\}_\phi \end{aligned} \tag{2.94}$$

$$J_\chi \dot{\beta} = \sum_{i \in [1 \ldots n_l]} \left\{ {}^{B}M_{L_i}^{W_{ll}} \right\}_\chi + \sum_{j \in [1 \ldots n_r]} \left\{ {}^{B}M_{R_j}^{W_{ll}} \right\}_\chi \tag{2.95}$$

$$J_\psi \ddot{\psi} = \sum_{i \in [1 \ldots n_l]} \left\{ {}^{B}M_{L_i}^{W_{ll}} \right\}_\psi + \sum_{j \in [1 \ldots n_r]} \left\{ {}^{B}M_{R_j}^{W_{ll}} \right\}_\psi + \left\{ {}^{B}\vec{M}_g^{W_{ll}} \right\}_\psi + \left\{ {}^{B}\vec{M}_c^{W_{ll}} \right\}_\psi + \left\{ {}^{B}\vec{M}_s^{PS_{ll}} \right\}_\psi \tag{2.96}$$

where n_r and n_l are the numeber of contact points on the right and the left wheel at a certain time.
The equations of motion of the other wheel sets are written in the same way, with the exception that they use their own parameters and forces.

Chapter 3

Numerical Methods

Big systems of non-linear differential equations are usually cumbersome if not impossible to be solved analytically. The system introduced in the preceding chapter belongs to these kind of problems, with the additional complication of having discontinuous functions for the wheel-rail interaction provided by the RSGEO routine. Even for simpler cases that model just the wheel-rail interaction, the analysis of the dynamics is not straight forward. Thus, numerical methods can be used for finding approximate solutions to the system, depending on certain parameters and starting conditions. In chapter 2 the system was written as a set of 66 first order differential equations in the form:

$$\begin{aligned} \dot{y}(t) &= f\left(t, y, v, R, \phi_t\right) \\ y(0) &= y_0 \end{aligned} \tag{3.1}$$

where the parameters v, R and ϕ_t are the speed of the vehicle, the radius and the cant of the track respectively. A lot of standard solvers for solving ordinary differential equations stated in the form (3.1) exist and only requires that we can provide the right hand side function $f\left(t, y, v, R, \phi_t\right)$ as input. Some knowledge about the problem suggests that this is characterized by high stiffness. There is no specific definition of what a stiff problem is, but a good measure of stiffness can be given by the ratio between the maximum and the minimum eigenvalues of the Jacobian matrix, called in the following *Stiffness Ratio*. Each egienvalue expresses which is the variational speed of different modes in the system, thus

the stiffness ratio shows how big is the gap between the fastest mode and the slowest mode. A big stiffness ratio will require the explicit method to take a smaller step in order to work in its stability region and follow the fast mode. This can cause the failure to converge, in case of fixed step size, or the adoption of unreasonably small steps, in case of controlled step size. Problems in train dynamics can have a stiffness ratio that range around 10^{10}. Thus the usage of A-stable methods with step size control is suggested, in order to get convergence and acceptable error size [Lev07]
Three numerical solvers were selected for their good properties on stiff problems and were used during this work: the Bulirsch-Stoer method, the Backward Differentiation Formula and the Explicit Singly Diagonal Implicit Runge-Kutta (ESDIRK) solver.
The Bulirsch-Stoer(BS) method [BS66] is a second order accurate extrapolation method based on the semi-implicit mid-point rule. The implementation used in this work is taken from the GNU Scientific Library[GSL], that applies the algorithm proposed by Bader-Deuflhard [BD83].
The Backward Differentiation Formula(BDF) is a linear multi-step method suitable for stiff problems. The version used employs a variable order predictor-corrector BDF formula due to Byrne-Hindmarsh[BH75]. The implementation is taken from the GNU Scientific Library[GSL] that implements the algorithm similarly to the SUNDIALS package[HBG05, BB89].
The Explicit Singly Diagonal Implicit Runge-Kutta[KTJr08] method belongs to the Runge-Kutta family and can use error approximation in order to adapt the step size. Several orders of accurancy can be reached depending on the Butcher tableau used[But64]. The first Runge-Kutta method used is the Nielsen-Thomsen's ESDIRK34 (NT1)[NT93, Os98] of order 3 for the advancing method and order 4 for the embedded method. The second method used is the Jørgensen-Kristensen-Thomsen's ESDIR34 (JKT)[KTJr08] of order 3 for the advancing method and order 4 for the embedded method.

3.1 Bulirsch-Stoer method

The Bulirsch-Stoer method is an implicit method based on the mid-point rule. The mid-point rule is an explicit method and it can fail in solving problems with high stiffness. If constant step size is used the method could fail in converging, whereas if adaptive step size is used the controller could shrink the step so much to make the computational effort unacceptable.
In order to solve this weakness of explicit methods, implicit methods can be used. An alternative is to transform the problem as Lawson[Law67] proposed, with the aim of lowering down the stiffness ratio. The transformation

$$c(t) = e^{-tA}y(t) \tag{3.2}$$

where $A \in \mathbb{R}^{n \times n}$, is used for rewriting the initial value problem in (3.1) as

$$\begin{aligned} \dot{c}(t) &= g(t, c) = e^{-tA}\left[f\left(t, c(t)e^{tA}\right) - Ac(t)e^{tA}\right] \\ c(0) &= y_0 \end{aligned} \tag{3.3}$$

Using the fact that A commute with respect to e^{tA}, the Jacobian of this problem will have the form:

$$\left(\frac{\partial g}{\partial c}\right) = e^{-tA}\left(\frac{\partial f}{\partial y} - A\right)e^{tA} \tag{3.4}$$

Hence, the eigenvalues of the modified system (3.3) will be the eigenvalues of $\left(\frac{\partial f}{\partial y} - A\right)$. It will be sufficient to define A such that the all the eigenvalues will be equal in size and solve the problem back. The most intuitive candidate for A is the Jacobian itself taken at the beginning of each step. The matrix exponential e^{hA}, where h is the step size, is approximated by the *Padé approximation*:

$$e^{hA} \rightarrow E(hA) \approx \mathbf{I} + hA \tag{3.5}$$

Each step of the algorithm is split into l basic iteration steps. Given that $k = 0$ represents the start of a basic iteration step, and the following definitions

$$A := f_y(y_0) \tag{3.6}$$
$$\bar{f}(y) := f(y) - Ay \tag{3.7}$$

the Semi-implicit Mid-Point Rule is given by the following steps:

$$\eta_0 := y_0 \tag{3.8}$$
$$\eta_1 := (\mathbf{I} - hA)^{-1}\left[y_0 + h\bar{f}(y_0)\right] \tag{3.9}$$
$$\eta_{k+1} := (\mathbf{I} - hA)^{-1}\left[(\mathbf{I} + hA)\eta_{k-1} + 2h\bar{f}(\eta_k)\right] \quad \text{for } k = 1, \ldots, l \tag{3.10}$$
$$\bar{S}_{2m} := \frac{1}{2}\left(\eta_{2m+1} + \eta_{2m-1}\right) \quad \text{for } l = 2m \tag{3.11}$$

where the starting steps are given by the semi-implicit Euler step, and the final step, computed at the end of the basic iteration steps, is a smoothing step that have a key role in the stability of the method [Sha83]. The final value for a step of size h is retrieved by extrapolation of different series of basic iterations with different refinements according to the step size sequence:

$$l \in \{2, 6, 10, 14, 22, 34, 50, 70\} \tag{3.12}$$

This step size sequence is also used for estimating the order of the method and its error. The computation of multiple refinements is speed up by the fact that the Jacobian needs to be computed only once and that the factorization of the matrix $(\mathbf{I} - hA)$ can be reused.

The importance of the final smoothing step can be shown using the A-stability definition by Dahlquist[Dah63]. The test equation

$$\dot{y} = \lambda y \quad y(0) = y_0 \tag{3.13}$$

where $\lambda \in \mathbb{C}$ and $\Re(\lambda) \leq 0$, can be used for the analysis of the method. Defining $z := f_y h = \lambda h$, the discretization (3.8) - (3.11) leads to

$$\eta_{2m-1} = \frac{1}{1-z}\left(\frac{1+z}{1-z}\right)^{m-1} y_0 \tag{3.14}$$

$$\eta_{2m} = \left(\frac{1+z}{1-z}\right)^{m} y_0 \tag{3.15}$$

$$\bar{S}_{2m} = \frac{1}{(1-z)^2}\left(\frac{1+z}{1-z}\right)^{m-1} y_0 \tag{3.16}$$

The A-stability is given by the fact that for $\Re(z) \leq 0$

$$|\eta_l| \leq |y_0| \tag{3.17}$$

$$|\bar{S}_{2m}| \leq |y_0| \tag{3.18}$$

Furthermore, for $\Re(z) \to -\infty$,

$$\eta_{2m} \to (-1)^m y_0 \tag{3.19}$$

$$\eta_{2m-1} \to 0 \tag{3.20}$$

$$\bar{S}_{2m} \to 0 \tag{3.21}$$

proving that the method is strongly absolutely stable (L-stable) for l odd. The importance of the smoothing step on even l is now obvious because it provides absolute stability. These properties can be checked observing the region of absolute stability shown in figure 3.1. The even steps are clearly not L-stable because the magnitude of the transfer function gets close to one as $\Re(z) \to -\infty$. The smoothing function provides this additional property to the method.

3.1.1 Step Size Controller

In order to get an estimation of the error and the order of the method, the step is refined according to the step size sequence in (3.12) and some heuristics are used[BD83]. This error is then used in order to adapt the step size depending on the absolute and relative error requested for the problem. The GNU Scientific Library[GSL] provides an asymptotic step size controller with the possibility of tuning some parameters. In the implementation of this work, the desired error is computed as

$$D_i = \varepsilon_{rel} |y_i| + \varepsilon_{abs} \tag{3.22}$$

and the maximum of the ratios between the estimated error and the desired error is given by

$$r_{max} = \max_{i \in [1,n]} \left(\frac{E_i}{D_i} \right) \tag{3.23}$$

where i varies in the n degrees of freedom of the system. If the maximum ratio exceed of more than 10% the desired error, then the step is reduced

$$r = \frac{0.9}{(r_{max})^{\frac{1}{q}}} \tag{3.24}$$

$$h_{k+1} = \begin{cases} 0.2 \cdot h_k & \text{if } r < 0.2 \\ r \cdot h_k & \text{if } r \geq 0.2 \end{cases} \tag{3.25}$$

If the estimated error is more than 50% lower respect to the desired error, then the step can be increased

$$r = \frac{0.9}{(r_{max})^{\frac{1}{q+1}}} \tag{3.26}$$

$$h_{k+1} = \begin{cases} 1.0 \cdot h_k & \text{if } r < 1.0 \\ r \cdot h_k & \text{if } 1.0 \leq r \leq 5.0 \\ 5.0 \cdot h_k & \text{if } r > 5.0 \end{cases} \tag{3.27}$$

3.2 Backward Differentiation Formula

A very effective class of methods for stiff problems are the backward differentiation formula (BDF) methods[Lev07, Sec. 8.4]. Given the problem in (3.1), the estimation after $n + r$ steps is given by

$$\alpha_0 Y^n + \alpha_1 Y^{n+1} + \cdots + \alpha_r Y^{n+r} = k\beta_r f\left(Y^{n+r}\right) \tag{3.28}$$

Any r-step method is also rth order accurate. The first order accurate BDF method is the Euler method and methods for all orders can be found, however only the methods up to the 6-step BDF are zero-stable. Hence only method with $1 \leq r \leq 6$ are relevant in practice. BDF methods have the good property of working very well when the eigenvalues of the problem are very negative. However, for some problems, some eigenvalues can lay very close to the imaginary axis and in this case the BDF methods have to be used carefully. The 1st (backward Euler) and 2nd order methods are A-stable, meaning that all the left half-plane is in the stability region of the method. Methods with orders from 3 to 6 are only A(α)-stable, because some parts of the region of instability crosses the imaginary axis and cover the left half of the complex plane. Figure

3.2 shows the stability region for the methods of order from 2 to 5. The stability region of the BDF of order one is the unit circle centered on 1. A method is said to be A(α)-stable if the cone $\pi - \alpha \leq \arg(z) \leq \pi + \alpha$ belongs to the region of stability, so the methods from order 3 to 5 are A(α)-stable with angles 88°, 73°, 51° respectively.

These methods can be combined in what is called a "polyalgorithm" with adaptive order (for a detailed description of the method see [BH75]). The methods from order 1 to order 5 are employed and a predictor-corrector strategy is used for finding an accurate solution and for estimating the error. The predictor step is performed using a Taylor series method on the basis of the historical data stored in the Nordsieck array:

$$z_n = \left[y_n, h\dot{y}_n, \frac{h^2}{2!}\ddot{y}_n, \ldots, \frac{h^q}{q!} y_n^{(q)} \right] \tag{3.29}$$

This prediction is then corrected solving the implicit system of equations given by the BDF rule. The solution of the implicit system is found by means of a Modified Newton Iteration method. The Nordsieck matrix is then updated with the new values found and it is modified accordingly to the new step size estimation and the new order of the method. It is stressed that the usage of predictor-corrector methods change the stability region of the corrector method. In particular the stability region becomes bounded.
The decision of when to change the order q of the method is taken after $q + 1$ steps. The Local Truncation Error (LTE) is estimated for both $q - 1$ and $q + 1$. These errors will determine the next step size for the three different orders:

$$\begin{aligned} h_{k+1} &= \eta_{q-1} h_k \\ h_{k+1} &= \eta_q h_k \\ h_{k+1} &= \eta_{q+1} h_k \end{aligned} \tag{3.30}$$

where $\eta_\bullet$ is the step size gain related to the estimated LTE. The order is chosen in order to maximize the next step size. Since the change in order costs computational time, spent in during the update of the Nordsieck array, it's necessary to avoid the change of order when $\max_{i\in\{-1,0,1\}}(\eta_{q+i}) < 1.5$. The method uses also some heuristics for minimizing the number of Jacobian updates. For the details on the implementation and the heuristics used see [HBG05, BB89].

The step size controller used for this method is the asymptotic one presented in section 3.1.1.

3.3 ESDIRK

The Runge-Kutta methods are a family of iterative one-step multi-stage methods. A lot of different solver with different stability properties are available. As it was already discussed, the problem that is going to be solved in this work shows a high stiffness ratio, thus only the implicit class of Runge-Kutta solvers is considered here. Each Runge-Kutta method can be described by a Butcher's tableau[But64] where all the parameters for each stage are listed. We will focus mainly on ESDIRK34 methods[KTJr08, NT93], with 4 stages and order 3, that have a tableau of the form

$$
\begin{array}{c|cccc}
0 & 0 & & & \\
c_2 & a_{21} & \gamma & & \\
c_3 & a_{31} & a_{32} & \gamma & \\
1 & b_1 & b_2 & b_3 & \gamma \\
\hline
 & b_1 & b_2 & b_3 & \gamma \\
 & \hat{b}_1 & \hat{b}_2 & \hat{b}_3 & \hat{b}_4 \\
\hline
 & d_1 & d_2 & d_3 & d_4
\end{array}
$$

This group of methods have the good property that the first stage is explicit, the stages 2,3,4 are singly diagonally implicit and the last stage is equal to the next first stage, practically saving computational time. The parameters in the tableau can be defined such that A-stability and L-stability properties hold. The implementation of ESDIRK34 can be done independently from its parameters, thus several versions of the solver can easily be applied. The first method considered is by Nielsen-Thomsen[NT93] (NT1), and is characterized by the following parameters:

$$
\begin{array}{c|cccc}
0 & 0 & & & \\
5/6 & 5/12 & 5/12 & & \\
10/21 & 95/588 & -5/49 & 5/12 & \\
1 & 59/600 & -31/75 & 539/600 & 5/12 \\
\hline
 & 59/600 & -31/75 & 539/600 & 5/12 \\
\hline
 & -37/600 & -37/75 & 1813/6600 & 37/132
\end{array}
$$

This method is A-stable but not L-stable.
The second method by Jørgensen-Kristensen-Thomsen[KTJr08] (JKT) is characterized by the following Butcher's tableau:

0	0			
0.8717330430	0.4358665215	0.4358665215		
0.4682387448	0.1407377747	−0.108365551	0.4358665215	
1	0.1023994006	−0.376878452	0.8386125301	0.4358665215
	0.1023994006	−0.376878452	0.8386125301	0.4358665215
	−0.054625497	−0.494208893	0.2219344997	0.3268998911

This method is both A-stable and L-Stable.
Even if an observation of the absolute stability region doesn't provide proves of A and L-stability, it gives good indications of the characteristics of the method[Lev07, Ch. 7]. The figure 3.3 suggests that the whole left half plane belongs to the stability region of the methods confirming that NT1 and JKT are A-stable. Denoting $R(z)$ the transfer function for the Runge-Kutta methods, the following holds

$$|R(z)| < 1 \text{ for } \Re(z) < 0$$

meaning that the methods are A-stable.

3.3.1 Step Size Controller

The SDIRK package[Os98] offers several choices for the step size controller. In this work the used one is the Proportional-Integral (PI) control[SÖ2]. The main difference with the asymptotical error control is that PI accounts also for error trends: if the error is increasing then the step size will be decreased faster than the asymptotic controller would do. Likewise, if the error is decreasing then the step size will be increased faster. The PI controller can be written as

$$h_{n+1} = \left(\frac{\varepsilon}{\hat{r}_{n+1}}\right)^{k_I} \left(\frac{\hat{r}_n}{\hat{r}_{n+1}}\right)^{k_P} h_n \tag{3.31}$$

where $\hat{r}_i = \|\hat{e}_i\|_2$ is the estimated magnitude of the error at the step i, ε is the tolerance error, k_I is the integral gain and k_P is the proportional gain. The two gains of the PI controller have to be set, in order to get proper adaptive step size. The knowledge of the underlying process in the problem to be solved can be an help in parameterizing the PI controller, however this analysis is not available for the present work, so standard choices for k_I and k_P are done. It is usually a conservative strategy to use pure integral control whenever a step is strongly rejected, meaning that there is the need of a big decrease of the stepsize independently from the trend. In particular:

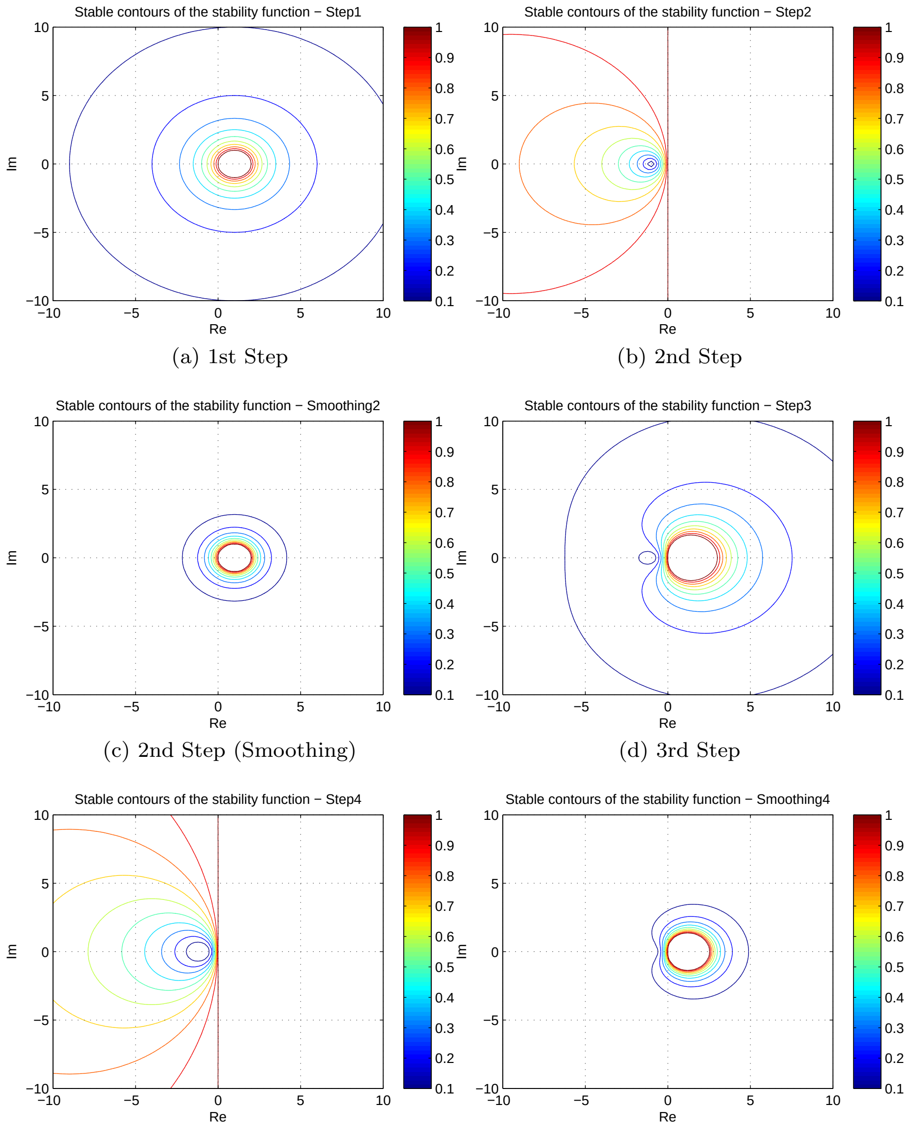
$$h_{n+1} = \left(\frac{\varepsilon}{\hat{r}_{n+1}}\right)^{0.2} h_n \qquad \text{if } \hat{r}_{n+1} > 1.2 \cdot \varepsilon \tag{3.32}$$

$$h_{n+1} = \begin{cases} \alpha \cdot h_n & \text{if } \alpha < 2.0 \\ 2.0 \cdot h_n & \text{if } \alpha \geq 2.0 \end{cases} \qquad \text{if } \hat{r}_{n+1} \leq 1.2 \cdot \varepsilon \tag{3.33}$$

where

$$\alpha = \left(\frac{\varepsilon}{\hat{r}_{n+1}}\right)^{\frac{0.3}{q}} \left(\frac{\hat{r}_n}{\hat{r}_{n+1}}\right)^{\frac{0.4}{q}} \tag{3.34}$$

and q is the order of the embedded method.

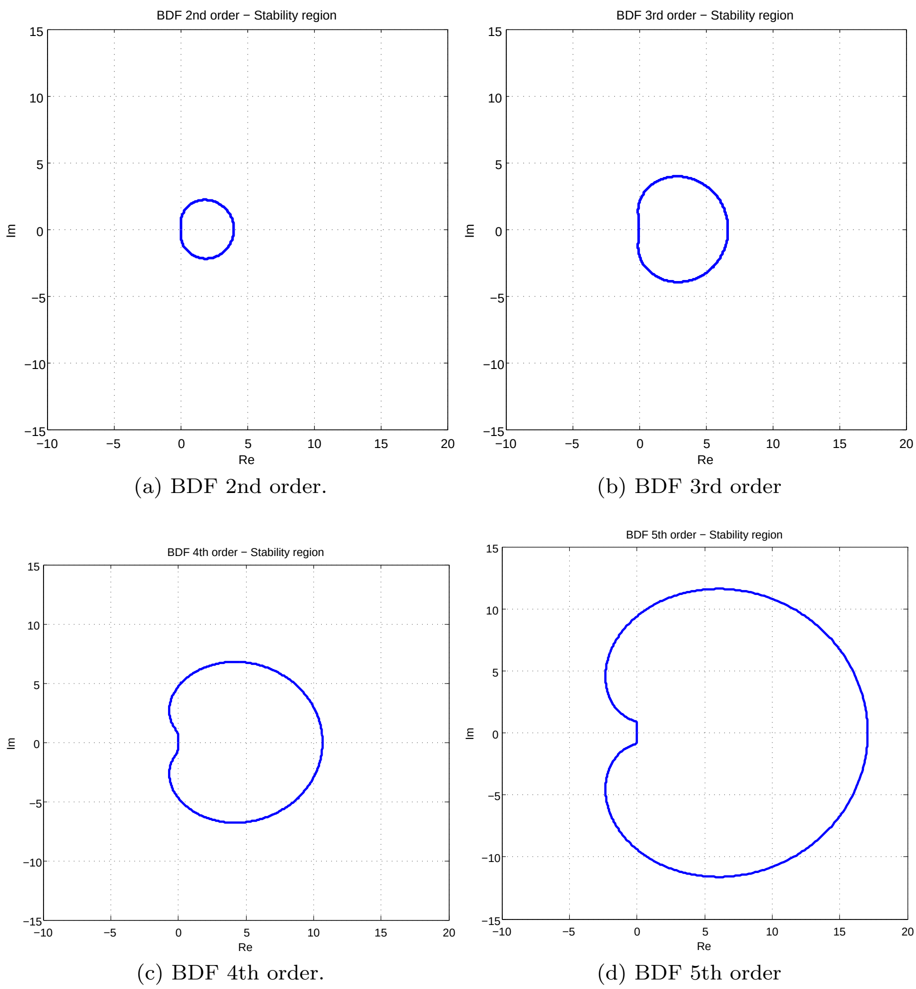


(a) 1st Step (b) 2nd Step

(c) 2nd Step (Smoothing) (d) 3rd Step

(e) 4th Step (f) 4th Step (Smoothing)

Figure 3.1: Regions of absolute stability for the Bulirsch-Stoer method. The region of absolute stability changes with the number of internal iterations.



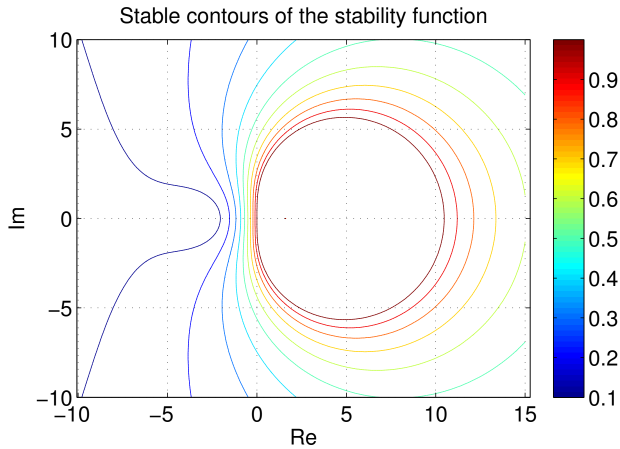
(a) BDF 2nd order.

(b) BDF 3rd order

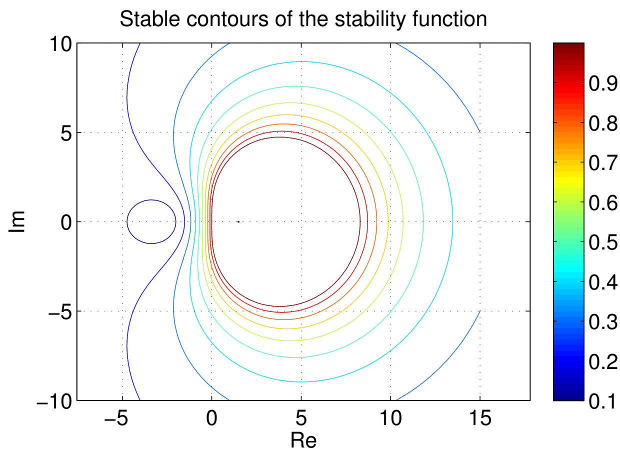
(c) BDF 4th order.

(d) BDF 5th order

Figure 3.2: Regions of absolute stability of the backward differentiation formula methods from order 2 to order 5. The methods are stable outside of the contour in the figures.



(a) Nielsen-Thomsen



(b) Jørgensen-Kristensen-Thomsen

Figure 3.3: Regions of absolute stability of the ESDIRK methods.

3.4 Convergence and Stability tests

DYTSI uses external packages for its solvers and they need to be tested for basic problems in order to check that the results correspond to the theoretical properties of the methods.
The test problem

$$\dot{u}(t) = \lambda u(t) \qquad\qquad u(t_0) = \mu \tag{3.35}$$

is used for testing the convergence of the selected methods. The solvers need to be disconnected from the step size controller, in order to be able to take steps of fixed sizes. The parameters used for the test problem are $\lambda = -1$ and $\mu = 1$ in the time interval $[0, 1]$. The error is computed for the last iteration and related to the step size. Figure 3.4 shows the convergence rate of the four methods

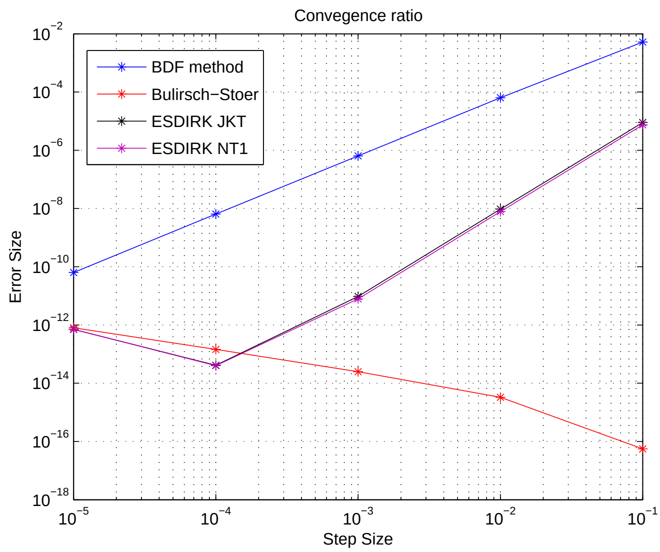


Figure 3.4: Convergence rate of the methods belonging to the GSL package and of the methods belonging to the ESDIRK package.

available in DYTSI. The BDF method shows a 2nd order behavior even if it has an adaptive-order algorithm and the test problem is smooth enough to use higher order methods. This is probably due to the step size gain that is not sufficient for choosing an higher order. The Bulirsch-Stoer method uses internal iterations for each step and the final accuracy is better than for the other methods. The two methods of the ESDIRK family are third order accurate and behave with similar performances on the test problem. The error shrinks as the step size is reduced, thus the four methods are convergent.

Now a simple harmonic oscillator without damping is considered in the form

$$
\begin{aligned}
m\ddot{x} &= -kx \\
x(t_0) &= x_0 \\
\dot{x}(t_0) &= x_1
\end{aligned}
\tag{3.36}
$$

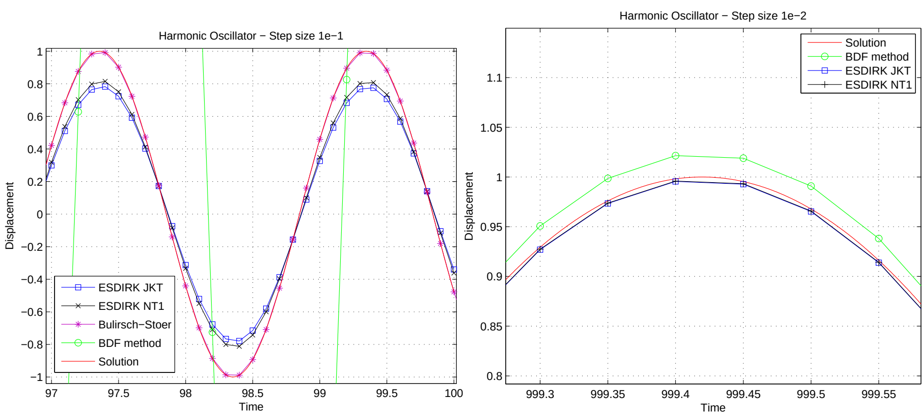
where k is the stiffness constant and m is the mass in a spring-mass system. The solution to such a problem is the undamped harmonic

$$
x(t) = A\cos(2\pi f t + \phi) \qquad\qquad f = \frac{1}{2\pi}\sqrt{\frac{k}{m}} \tag{3.37}
$$

For initial conditions $x_0 = 1$ and $x_1 = 0$, the solution becomes

$$
x(t) = \cos\left(t\sqrt{\frac{k}{m}}\right) \tag{3.38}
$$

The mode of the solution does not die out with time as the eigenvalues of the Jacobian of the system are positioned exactly on the imaginary axis.
The four solvers are tested with fixed step size on the problem. Figures in 3.5



(a) Step size 10^{-1}

(b) Step size 10^{-1}

Figure 3.5: Stability and damping effects on the harmonic oscillator.

show that the BDF method is unstable on this problem. The combination of the BDF method in a predictor-corrector scheme causes the region of absolute stability to be bounded, thus the method can suffer the same problems that explicit methods have. The step size reduction shown in figure 3.5b indicates that the amount of numerical damping for the ESDIRK methods decreases with the step size, as the eigenvalues of the system are moved toward the origin in the stability region (see fig.3.3). The Bulirsch-Stoer method is finding a good solution already for the step size 10^{-1} but is failing with smaller step sizes. The position of the eigenvalues of the system could be the cause of the problem: observing the region of absolute stability of the BS method and its internal iterations (see fig.3.1), in the even steps the imaginary axis is right on the boundary of the stability region. The eigenvalues of the problem lay exactly there and small numerical errors can move them on the unstable side. The unstable effect is much higher when the eigenvalues are moved toward the origin by taking smaller steps. The aim of the smoothing step on this method is to provide L-stability on the even steps, but it does not address the problem of eigenvalues positioned on the imaginary axis.

Next, the focus will move to the stability property of the methods on stiff problems. The computation of the train dynamics involves a system of 66 ODEs with stiffness that is around $10^3 - 10^4$ for the Cooperrider model used in section 6.1. Implicit methods are known to provide stability for stiff systems thanks to their stability region that span on all the left half plane. This allows the method to take big steps at the only cost of the accuracy. In order to improve accuracy some methods use predictor-corrector schemes that bound the stability region and this has to be taken in account when the step size is chosen. With the purpose of testing the stability of the methods on a simple stiff problem, the following second order differential equation is constructed:

$$\begin{aligned} &\ddot{y} + (\mu + 1)\dot{y} + \mu y = 0 \\ &y(0) = 1 \\ &\dot{y}(0) = -7 \end{aligned} \tag{3.39}$$

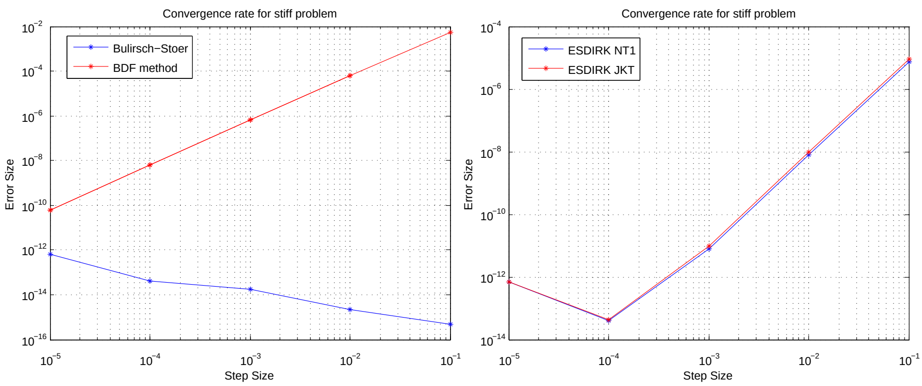
where $\mu \gg 1$. Using variable substitution the following system of ODEs is obtained

$$\begin{cases} \dot{y}_1 = y_2 \\ \dot{y}_2 = -(\mu + 1)y_2 - \mu y_1 \end{cases} \tag{3.40}$$

The eigenvalues of the Jacobian of this system are $\lambda_1 = -\mu$ and $\lambda_2 = -1$, that gives a stiffness ratio of μ. The solution of (3.39) is:

$$y(t) = \frac{6}{\mu - 1}\exp(-\mu t) + \frac{\mu - 7}{\mu - 1}\exp(-t) \tag{3.41}$$

The figures in 3.6 show the convergence rate of the methods when applied to the problem (3.39) with $\mu = 10^4$ in the time interval $[0, 1]$. All the methods are



(a) GSL Package (b) SDIRK Package

Figure 3.6: Convergence rate of the methods when applied to the stiff problem (3.39).

stable in spite of the rapidly decaying mode represented by the first part of the solution (3.41).

Some of the properties and weaknesses of the methods have been shown in this section and will help in the interpretation of the results in the following sections. In particular the results of sections 6.1 and 6.2 have been found using ESDIRK NT1. Some of the gross results, like the critical speeds, have been confirmed using all the four methods. Finally, methods are compared in terms of accuracy and computational efficiency in section 6.3.2.

CHAPTER 4

Implementation

In chapter 2 the mathematical formulation of the multi body dynamical system representing a four-axles wagon was introduced. The natural choice of programming paradigm for implementing a multi body system is the Object-Oriented Programming (OOP). This because each component of a multi body system can be easily represented by an object describing its characteristics and its behaviors. The OOP language chosen for the implementation is C++, due to it's portability and good performances. The implementation was done on a GNU/Linux Operative System and on a SunOS Solaris Sparc Sun4v (the multi-thread support is not available for the latter OS). Some functionalities of the numerical GNU Scientific Library[GSL] were used: the Bulirsch-Stoer solver, the Backward Differentiation Formula solver, the CBLAS library for matrix-vector computation, some root finding functions for the application of static loads and the interpolation functions for smoothing the RSGEO values. The SDIRK package[Os98] was integrated for the NT1 solver and expanded with the JKT solver (see sec. 3.3). The Xerces-C++ XML Parser[Pro07] was used for the input files. The RSGEO[KM10] routine was used for computing the static contact parameters needed for the calculation of the wheel-rail interaction forces.
In this chapter the structure of the DYnamics Train SImulation (DYTSI) program will be explained. The notation of Unified Modeling Language (UML) will be used in order to show the relation between classes and the execution of the program will be explained using flow charts.
A little guide to the usage of DYTSI for modeling new wagon models is provided

in Appendix D.

4.1 DYTSI: DYnamics Train SImulation

The implementation of DYTSI started out with the identification of some categories of classes. These categories are represented in the code by several namespaces that distinguish between auxiliary classes and classes that are at the core of the program. Figure 4.1 shows a simplified overview of the class diagram. The namespaces `DYTSI_Input` and `DYTSI_Output` collects all the classes that are in charge of reading the input, printing on the terminal and writing the output to external files. The `DYTSI_Modelling` namespace contains all the tools for building and representing a wagon: each component is represented by a class here and they can be connected each other. This part of DYTSI will be covered in section 4.1.1. The `DYTSI_Models` namespace collects all the available models in the package. This is the place where to put new models to be tested. The available models are described in Appendix C, whereas the basic steps for building new models are shown in Appendix D. The namespace `DYTSI_Solver` contains the classes representing the ODE solvers. In practice the GNU Scientific Library and the SDIRK package are integrated in a unique interface.

4.1.1 Vehicle Model

Any multi body system can be described by a set of static characteristics, such as the size, the weight, the inertia, and a set of differential equations that will describe its dynamical response. It was already stressed that each component of the vehicle can be represented by an object that will describe its static characteristics. The dynamics will be described by functions that belong to the object. Figure 4.2 shows a simplified model of the class diagram of the `DYTSI_Modelling` namespace. The connections between the classes are shown, but the attributes and the methods of each class have been removed due to space constraints. Different colors are used in order to distinguish between:

- Light blue. Abstract classes: these classes can only be extended and all the subclasses need to implement the virtual methods of the abstract class. These classes are used in order to represent abstract objects that share certain characteristics.
- Gray. Hidden Classes: classes used internally by the package and that the user doesn't need to use in order to build models.

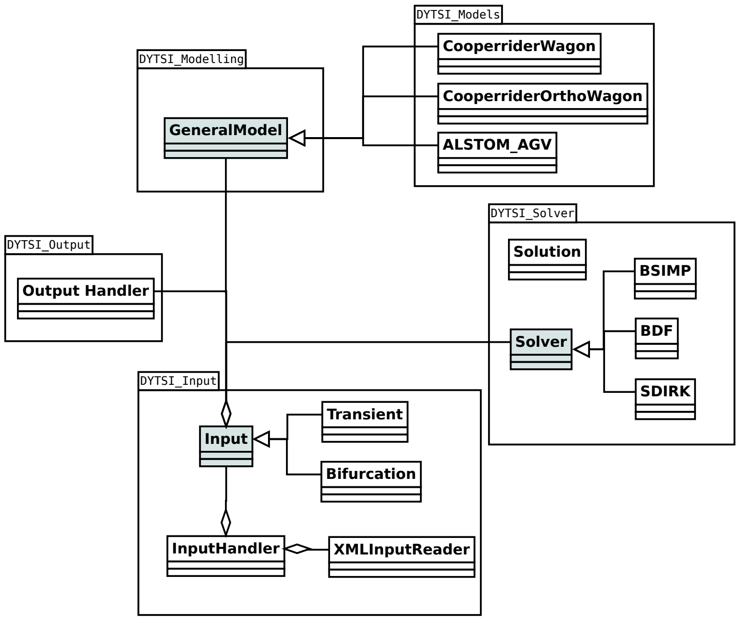


Figure 4.1: DYTSI namespaces. This simplified view shows how the code is split among different namespaces.

- White. The bricks of the model. These classes have to be used by the user in order to build models.
- Green. The `GeneralModel` class is the handler that keeps all the bricks together and provide the functions necessary for the computation of the dynamics, namely the right hand side function $f(t, y, v, R, \phi_t)$ and the Jacobian $f_y(t, y, v, R, \phi_t)$ of the system.

A four-axles wagon can be represented by a tree of component, where the wagon is the root component, the two bogie frames are two leaves of the root and the wheel sets are children of the bogie frames. Two main blocks of elements can be identified: the bodies of the model that are generalization of the class `Component` and the suspension elements that are generalization of the class `Link`. It's important now to explain how and by who the right hand side function and the Jacobian are computed. The idea is that each component should compute the values relative to itself. Figure 4.3 shows how the computational work is divided among the objects of the model. During the set up of the model each component reserves its set of degrees of freedom, and the necessary memory in order to contain them. For example, the Car Body component will reserve the memory in the red circle in figure 4.3. It will be in charge of computing

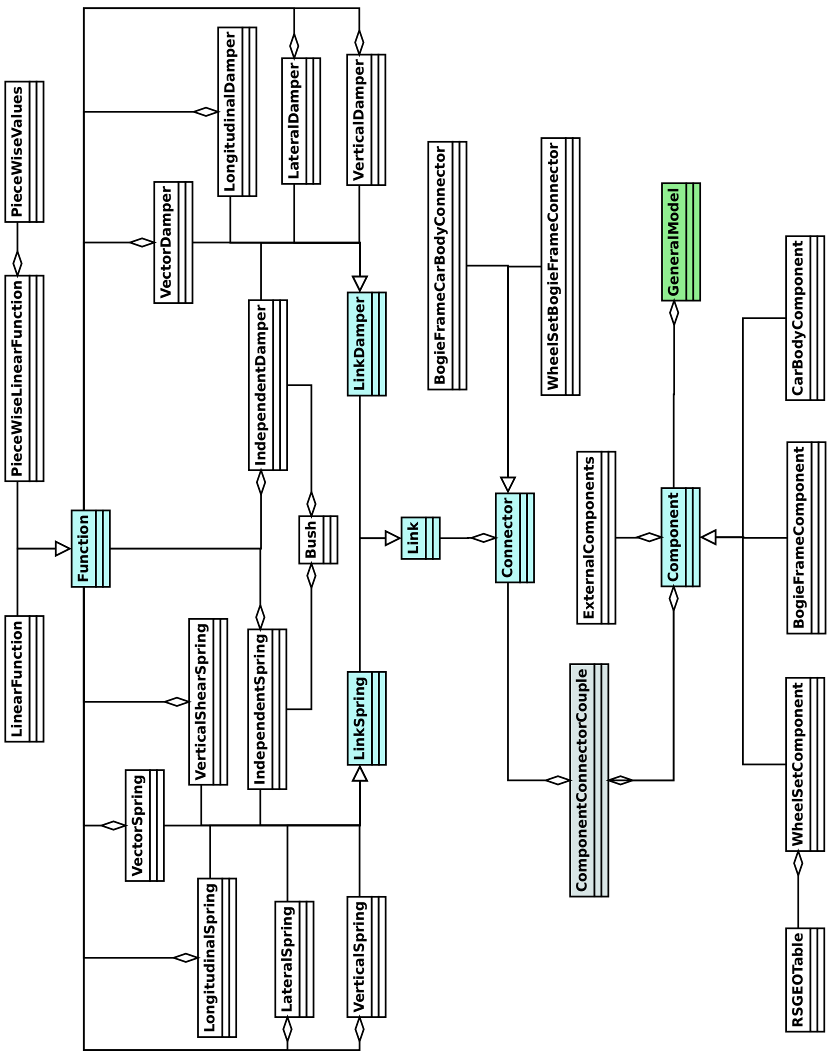


Figure 4.2: The `DYTSI_Modelling` namespace

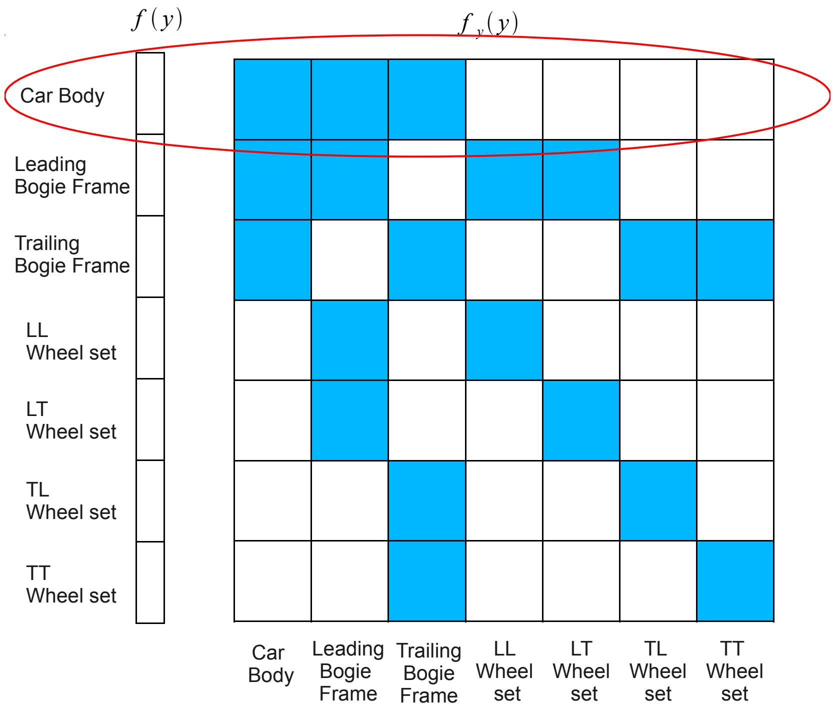


Figure 4.3: Assignment of the computational load to the different objects

only the part of degrees of freedom that it has reserved and that involve the car body. Thus, all the classes that generalize the `Component` class will have to declare a function `fun` and a function `jac` for the computation of the right hand side and the Jacobian. Furthermore each of the `Component` classes will write in different memory locations, allowing the parallel computation of these functions. The computation of the Jacobian is done numerically using difference approximation: each degree of freedom is perturbed in both the direction and

$$\{f_y(t,y)\}_{:,i} = \frac{f\left(t, \{y_0, \dots, y_i + h, \dots, y_n\}^T\right) - f\left(t, \{y_0, \dots, y_i - h, \dots, y_n\}^T\right)}{2h} \tag{4.1}$$

where the notation $\{\}_{:,i}$ means the ith column of the Jacobian matrix and h is little with respect to y. Equation (4.1) means that the ith column of the Jacobian matrix is computed applying a perturbation on the ith degree of freedom. If this degree of freedom, for example, belongs to the leading bogie frame, then the suspension forces on the car body and the two connected wheel sets will be changed. Instead, the suspension forces on the trailing bogie frame and the other two wheel sets will not change. Thus the Jacobian does not have to be computed completely, but only the highlighted blocks in figure 4.3 are relevant, since the rest of the blocks will be zero.

Let's now get a little deeper in the description of the classes and relate them to the entities introduced in chapter 2. A first step in the modeling of an Object Oriented Program is to abstract as much as possible from the real components:

- **Component**: all the generalizations of this class will have some common attributes like the mass, the inertia, and the dimensions. Some slot of degrees of freedom are assigned to each component: in particular each object knows the starting position of its degrees of freedom in the y array and the number of degrees of freedom related with a certain component (10 for the Car Body and the Bogie Frames, 9 for the Wheel sets). Furthermore each component is provided of a name and a unique ID that will allow to track possible errors or bugs. Each generalization of the class `Component` will be forced to implement a `fun` function and a `jac` function. Each component is also considered to be a leaf of a binary tree, so it has space for references to the a parent and two children `ComponentConnectorCouple` (this class is just composed by a reference to a `Component` and a reference to a `Connector`).

- **Connector**: the generalizations of this class will represent the primary and secondary suspensions, thus a collection of `Link`s, that can be springs and/or dampers. When asked, the connector is able to compute the suspension forces due to the displacement of the two connected `Components`. This is done asking to each `Link` in the `Connector` to compute its force due to the displacement of its attack points.

- **Link**: the `Link` class is exactly the implementation of the Link introduced in section 2.3.1. A link is simply a connection between two components. The connection is represented by two attack points on the surface of these components. In order to speed up the computation the deformation and its speed are not computed in the `Link` class, but separately in the `LinkSpring` and `LinkDamper` classes. The first computes only the deformation of the link, the second computes both the deformation (in order to derive the direction on which the link is working) and the speed. These classes compute the arguments of the function in equation (2.70), whereas their generalizations are in charge of computing the forces due to these deformations.

- **Function**: this class represents a one-dimensional function. All the generalizations of `Function` are forced to implement a `func` method that represents the force-displacement or force-speed relation for a certain element.

> The `PieceWiseLinearFunction` class is used for representing the behavior of non linear elements. It implements the following interpolation functions: linear interpolation, cubic spline and Akima interpolation[Aki70].

The vehicle model is formed by three types of components. The Car Body and the Bogie Frames are represented by the `CarBodyComponent` class and the `BogieFrameComponent` class respectively. They are similar in the behavior because they are simply connected to some suspensions, thus the forces to be computed will be the gravitational forces, the centrifugal forces and the suspension forces (see sec. 2.4.1 and 2.4.2). The latter are obtained asking each connected `Connector` to compute them. The Wheel Sets are represented by the `WheelSetComponent` class. This differs from the other two components by the fact that it has to compute the contact forces additionally. The contact forces are computed using one or more RSGEO tables representing one or more contact points (see sec. 4.2 for an explanation of how an RSGEO Table is built up). Thus the forces of each contact point are computed and summed to the gravitational forces, the centrifugal forces and the suspension forces (see sec. 2.4.3).
The suspension elements are implemented as generalizations of `LinkSpring` or `LinkDamper`. Their implementation is exactly as described in section 2.3.2.
Finally the class `GeneralModel` holds a reference to the `CarBodyComponent` and provides handles to the functions `fun` and `jac` that will be used by the solvers.

4.1.2 Solvers

Several solvers are provided both by the GNU Scientific Library and the SDIRK package. In order to bind these libraries to DYTSI, specific classes for each of the solvers have been implemented. Figure 4.4 shows a simplified view of the namespace `DYTSI_Solver`, where a generic `Solver` class defines how to solve an Initial Value Problem (IVP). The different solvers chosen will provide different `evolve` functions that depend on the stepping method chosen. The method to be used for a certain simulation can be defined in the input file. When a solver is asked to integrate an IVP on a time span $[t_0, t_f]$, it will iteratively call the `evolve` function until the final time is reached. The computed data will be stored, but in order to avoid huge dimension of the file, the output will be sampled at constant time. The `Solution` class is in charge of storing the states of the system. This structure is also in charge to store the Jacobians and all the forces per each sampling time.

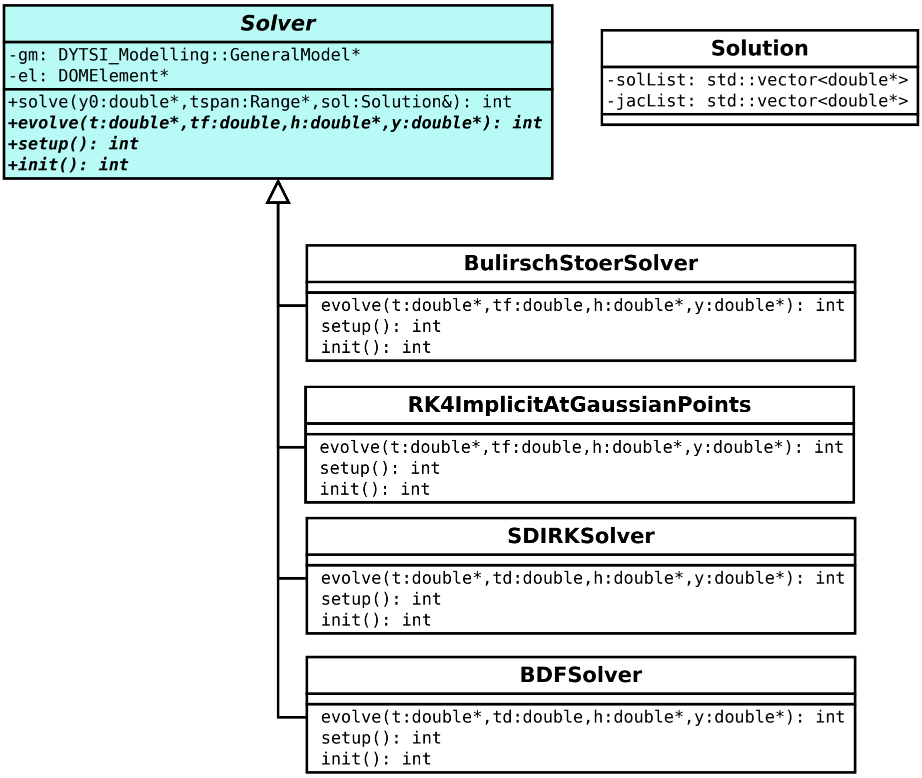


Figure 4.4: The `DYTSI_Solver` namespace

4.1.3 Program flow

A good way of getting an overview of how a program works, is to look to its execution flow during time, from the starting of the application to its natural end. The work flow of DYTSI is composed by two main parts. The first part is dedicated to the reading of the input and the creation of the data structure necessary for the elaboration of the output. The second part is in charge of executing the computational work and providing the output.

Input reading

The input provided to DYTSI is contained in an XML file that can describe several simulations to be done sequentially. The format of the input and its content are described in the tutorial of the program in appendix D. For each test found in

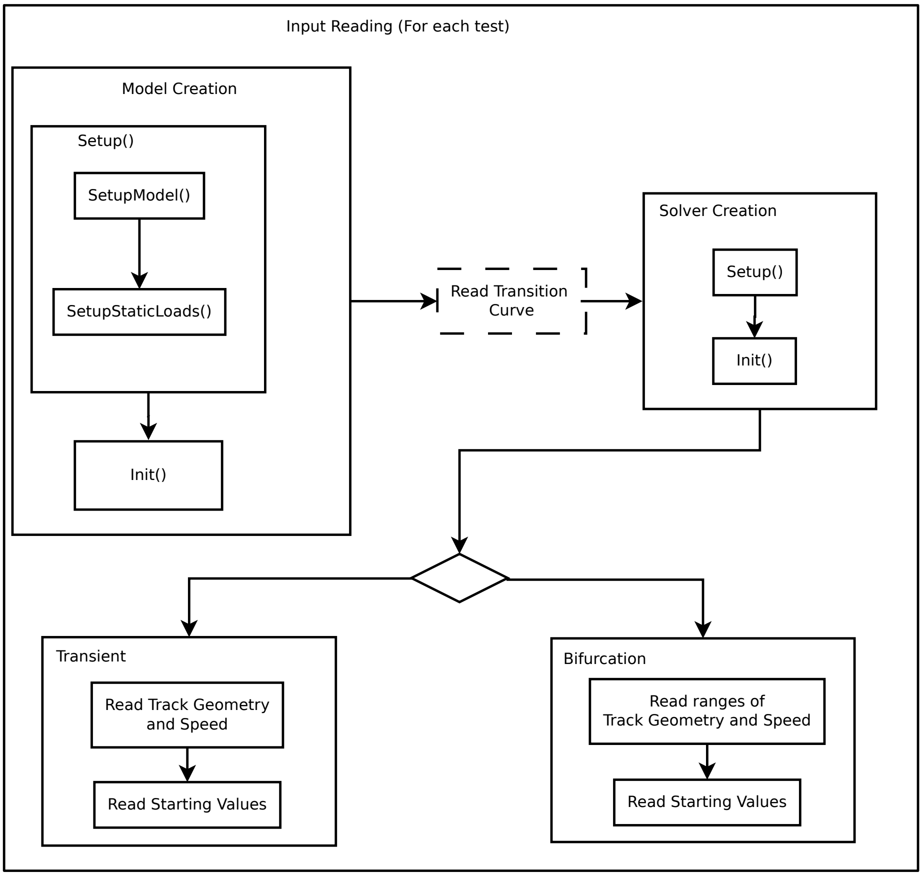


Figure 4.5: Flow char of the DYTSI program during the input phase.

the input file, DYTSI creates an instance of `DYTSI_Input::Input` that contains information about the model used, the type of simulation, the initial conditions, the solver used and the format of the output. Figure 4.5 shows the flow of the program for each test. The first information that it retrieve is about the model used: each particular model belonging to the namespace `DYTSI_Models` is able to read the information necessary for the definition of the multi body system, such as dimension and weight of the components, characteristics of the suspensions etc. The model is then assembled using the tools provided by the namespace `DYTSI_Modelling`. In this phase the RSGEO Table file can be read from the path provided in the input file. Since multiple contact points can be accounted and for each contact point there could be an RSGEO table, the user can decide whether to provide them in different files or in a properly formatted unique file. The interpolation functions, provided by the GNU Scientific Library, are set up

and initialized here in order to make the RSGEO values smoother. All this is done in the `setupModel` function and after this the program should be holding a pointer to an instance of `DYTSI_Modelling::GeneralModel` containing the wagon model assembled.
The setup of the model is not finished yet. Usually, vehicle models are designed under static loads, namely the gravity is already acting on the bodies so the vertical suspensions are already deformed before the simulation starts. In reality, the vertical stiff components are mounted before the assembly of the wagon. In order to maintain the wanted geometry of the suspension elements, the remaining components are mounted after the car body has been laid on the bogie frames and the bogie frames have been laid on the wheel sets. Thus the length at rest of all the vertical stiff components have to be corrected accordingly. Under the assumption of symmetry in the vertical components, the approach used for solving this problem statically is to share the load equally on each component and use equation (2.78) for vertical springs:

$$^{F}\vec{F_k} = k\left(\left\{{}^{F}\vec{b_l}\right\}_z - \left\{{}^{F}\vec{b_{l_0}}\right\}_z\right)\cdot \mathbf{e}_z \tag{4.2}$$

In this case the deformation length is known but the length at rest is not. Thus an iterative solver, such as the Brent-Dekker method[Bre73] (chosen among the methods provided by the GSL library[GSL]), can be used for solving:

$$\left\{{}^{F}\vec{b_{l_0}}\right\}_z = \left\{{}^{F}\vec{b_l}\right\}_z - k^{-1}\left(\left\{{}^{F}\vec{F_k}\right\}_z\right) \tag{4.3}$$

where $\left\{{}^{F}\vec{F_k}\right\}_z$ represents the part of vertical load due to the gravity charged on one stiff element (negative in the reference system used), $\left\{{}^{F}\vec{b_l}\right\}_z$ is the deformed length of the statically loaded element and the obtained $\left\{{}^{F}\vec{b_{l_0}}\right\}_z$ is the corrected length of the unloaded element. These adjustments are applied during the `setupStaticLoads` function.
The next step is to initialize the model by the function `init`. Different models can extend this function depending on whether they need some additional action to be taken, but, in general, this function is needed for resetting counters in all the components of the model. These counters have been inserted in order to track errors and bugs in the code. They provide information on the number of right hand side evaluations or Jacobian evaluations.
When the model has been built and initialized, DYTSI start constructing the optional *transition curve*, used in order to not have a sudden movement toward the outer rail when entering the curve. The presence of a transition curve will ask DYTSI to perform a transient analysis, over a certain period of time, with varying radius and cant. Since the study of the effect of transition curves on the train dynamics is out of the scope of this work, the linear transition curve model is adopted. Let's assume that the transition curve has to be covered in

a time Δt, obtaining the final radius of $\hat{R}$ and cant of $\hat{\phi}_{se}$. The cant and the radius at a certain time t will be given by:

$$\phi_{se}(t) = \begin{cases} 0 & \text{for } t \leq 0 \\ \frac{t}{\Delta t}\hat{\phi}_{se} & \text{for } 0 < t \leq \Delta t \\ \hat{\phi}_{se} & \text{for } t > \Delta t \end{cases} \tag{4.4}$$

$$R(t) = \begin{cases} \infty & \text{for } t \leq 0 \\ \frac{\Delta t}{t}\hat{R} & \text{for } 0 < t \leq \Delta t \\ \hat{R} & \text{for } t > \Delta t \end{cases} \tag{4.5}$$

After the transition curve, the track geometry and the speed will be constant for the length of the simulation. In the case the test being processed is a transient analysis, three single values of radius, cant and speed are needed. Instead, the bifurcation analysis is performed increasing (or decreasing) some of the parameters by a certain amount that will be called the parameters' refinement, thus ranges of values for the speed, the radius and the cant have to be defined. DYTSI proceeds now to initialize the solver selected, depending on the input parameters such as the accuracy of the method and the initial step length. Also the Solver is prepared in two steps: the `setup`, where the solver is created, and the `init`, where counters and internal states are reset.
Finally the initial values for each degree of freedom are read. For the bifurcation analysis two option for the initial values are available: the values can be used at the beginning of each transient analysis for the different parameters or can be used exclusively at the beginning of the first transient analysis. In the latter case, the next transient analysis will use the last states of the previous analysis as initial values.

Simulation

Two kinds of simulation are implemented in DYTSI. The first one is the *transient analysis* on a particular IVP problem. The transient analysis can be provided of a transition curve. The analysis of the dynamics in the transition curve is done using a transient analysis as well, where radius and cant are modified at each iteration. Figure 4.6 shows the flow chart of such analysis. The stepping function depends on the solver chosen, but all of them will require the computation of the right hand side function and the Jacobian matrix. These functions can be called thousands of times per transient analysis, and represent one of the critical parts of the computational load. As it was discussed in section 4.1.1, each body of the vehicle model is represented by a separate object and these objects are able to compute their right hand side function and Jacobian independently. This allows the usage of multi-threading in order to try to speed up

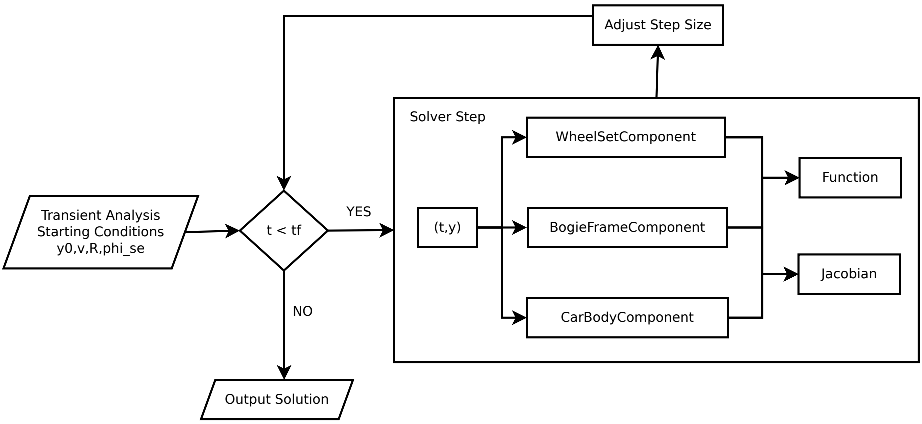


Figure 4.6: Flow char of a transient analysis.

the computational effort.
The second simulation implemented in DYTSI is the *bifurcation analysis*. The bifurcation diagram can be obtained observing the steady state of different transient analysis where one parameter is modified. The bifurcation analysis can be provided of a initial transition curve, such that the approach to the curve is gradual. Figure 4.7 shows the flow chart of a bifurcation analysis. Let's say

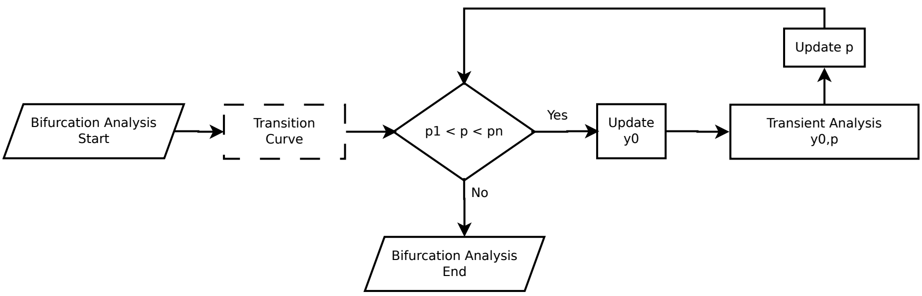


Figure 4.7: Flow char of a bifurcation analysis.

that the dynamical behaviors of the wagon in a curve with respect to a parameter $p_i \in [p_1, p_n]$ are to be analyzed. The varying parameter can be chosen exclusively between the speed, the radius and the cant. The simulation is started covering an optional transition curve. Then several transient analysis are computed sequentially with increasing or decreasing p_i. The change in the variational parameter is considered to be small, such that the last state of the transient analysis for p_{i-1} can be used as initial value for the transient analysis for p_i.
Two sampling times are defined in order to collect computational data. The

states of the dynamical system, as well as the forces acting on each body, are stored together in a big ordered set of arrays depending on the main sampling time. The Jacobian of the system is stored depending on the Jacobian sampling time. The output generated by DYTSI is explained in Appendix D.

4.2 RSGEO

Wheel-rail contact forces play an important role in the dynamics of a train, thus the correct computation of the geometry of the contact patch is mandatory in order to have reliable results. The theoretical approach for computing the contact forces in a dynamic setting was given in section 2.2. Some of the parameters necessary for the application of these theories can be computed in a statical setting, using the RSGEO routine[KM10]. The software lacks of a proper documentation about the usage and the interpretation of the output, so an introduction to RSGEO and to the output transformation necessary for retrieving the correct parameters will be given in Appendix E.
The RSGEO table will contain precomputed parameters with respect to the wheel set displacement. These parameters are listed in table 4.1 as they appear in the RSGEO table. In order to retrieve these parameters, the wheel and rail profiles have to be defined. For example, a GV40 wheel profile is considered and shown in Figure 4.8a. This is combined with a UIC60 rail profile with cant 1/20 toward the center of the track. A graphical output of the shape and the position of the patch for this combination depending on the displacement of the wheel set is shown in figure 4.8b. It's often the case that the contact patch does not move smoothly to the flange when the wheel set is moved outward. Several parts of the wheel profile could even never contact the rail profile. The passage from one contact point to another doesn't happen suddenly, but for a short range of displacements two contact points can be present, where one of them is unloading and the other is loading. This fact can be handled in several ways: either approximating the two contact patches with a unique patch[PS91], or considering both. DYTSI is able to handle one or more contact points, thus both the approaches are allowed.

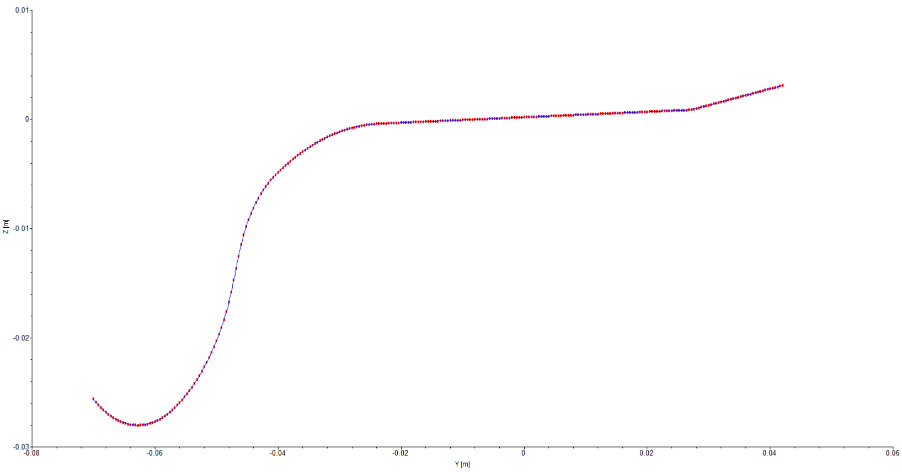
Figure 4.9 shows the values of the RSGEO table against the displacement of the wheel set for the wheel-rail combination S1002-UIC60 with rail cant of 1/40, used by Christiansen[Chr01]. In this setting up to three contact points appear, but they are approximated by a unique contact patch. The plots show all the parameters as they are listed in table 4.1. Figure 4.10 shows the wheel-rail combination GV40-UIC60 with rail cant of 1/20. The two contact points are not approximated by a unique contact patch, instead they are considered separately and the plots show how the contact patch on the tread of the wheel disappears while the contact patch on the flange appears, transferring the load from one to

Column	enum	Description
0	RSG_lat	Lateral displacement of the center of mass of the wheel set from the center of the track(m)
1	RSG_N	Normal force in the contact point coordinate system (N)
2	RSG_angle	Angle δ_l between the axle and the contact plane (rad)
3	RSG_a	Biggest semi-axis of the contact patch (m)
4	RSG_b	Smallest semi-axis of the contact patch (m)
5	RSG_Kwy	Lateral distance to the contact point from the center of mass of the wheelset (m)
6	RSG_Kwz	Actual rolling radius (positive) (m)
7	RSG_C11	Kalker's creepage coefficient C_{11}
8	RSG_C22	Kalker's creepage coefficient C_{22}
9	RSG_C23	Kalker's creepage coefficient C_{23}
10	RSG_Krz	Vertical position of the contact point on the rail measured from the top of the rail (m)
11	RSG_qN	Static penetration depht (m)

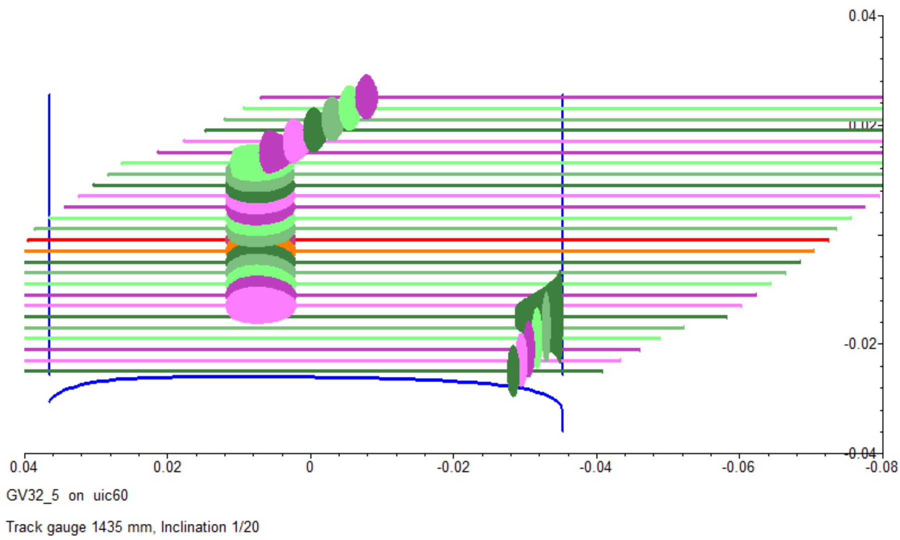
Table 4.1: Columns in the RSGEO table file. Here are listed both the number of column and the correspondent enumerator used in the implementation.

the other gradually.
In order to represent multiple contact points, several RSGEO tables are needed, one for each active contact point. The RSGEO routine automatically provides several tables when a new contact point appears, but these tables are usually mixed because the routine doesn't track each contact point. For this reason some additional manipulation is needed in order to prepare a correct and sufficiently smooth table (see Appendix E for more details). DYTSI will use these lookup tables during the computation of the contact forces. In order to detect whether a contact point is active or not, the size of the normal load is checked: if the load is nearly zero then the contact point doesn't exist, so the table is skipped. In order to prevent numerical complications, the threshold `MIN_RSGEO_NORMAL` is set to 1.0 in `WheelSetComponent.cpp`, so that contact points with normal loads smaller than 1N won't be considered. Each lookup on the RSGEO tables returns a value given by interpolation. Several interpolation techniques can be used, however some of them can undershoot when the values in the table are in the proximity of zero. This is the case when multiple contact points are considered, but it's not the case when only one contact point is considered. The solutions to this problem are two: either the interpolated values are forced to be positive or a different interpolation function is adopted. Usually cubic splines work well for contact tables that represent a single contact point, but they easily undershoot when applied to contact tables with multiple contact points. A

more robust method for these situations is the Akima[Aki70] interpolation, that employs a stronger locality such that the computed interpolation function between two points depends only on the coordinates of the two points and the slope at the points. In the case the RSGEO table is very refined, a simple linear interpolation could be sufficient.



(a) Profile of a GV40 (1/40 h28/e32.5/15) wheel.



(b) Vertical view of the contact patch movement with respect to the wheel set displacement. The horizontal lines represent the width of the wheel and its position when different displacements are applied. Patches are shown for each of such displacements.

Figure 4.8: Example of wheel profile and contact patch position on the rail. It's clearly visible that for this combination there is a big jump to the flange contact. A more detailed analysis of this wheel-rail combination would show that one of the contact points disappears while a new one appears.

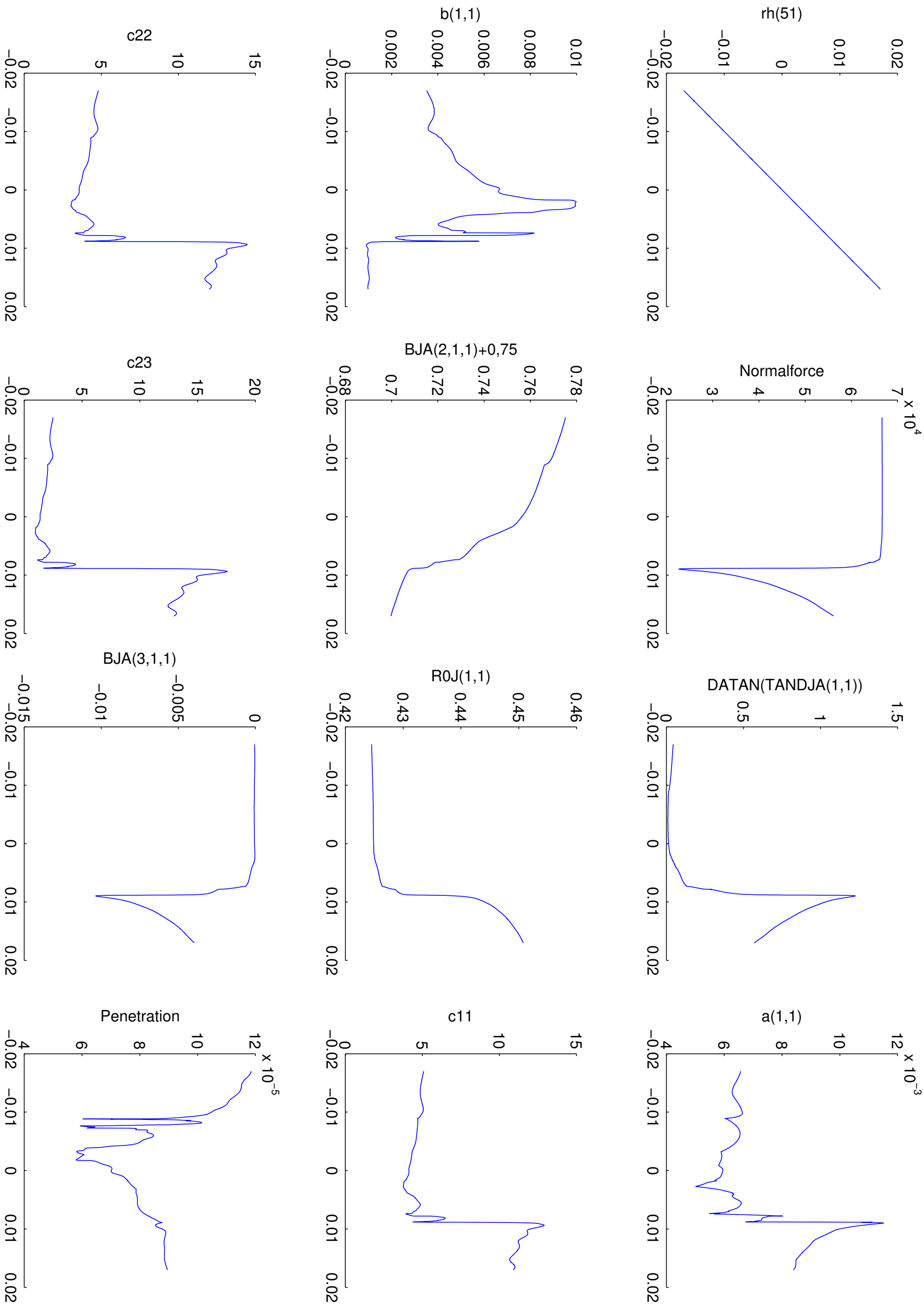


Figure 4.9: RSGEO values versus the displacement of the wheel set for the combination of wheel profile S1002 and rail UIC60 with cant 1/40 toward the center of the track. The multiple contact patches have been approximated by one unique contact patch.

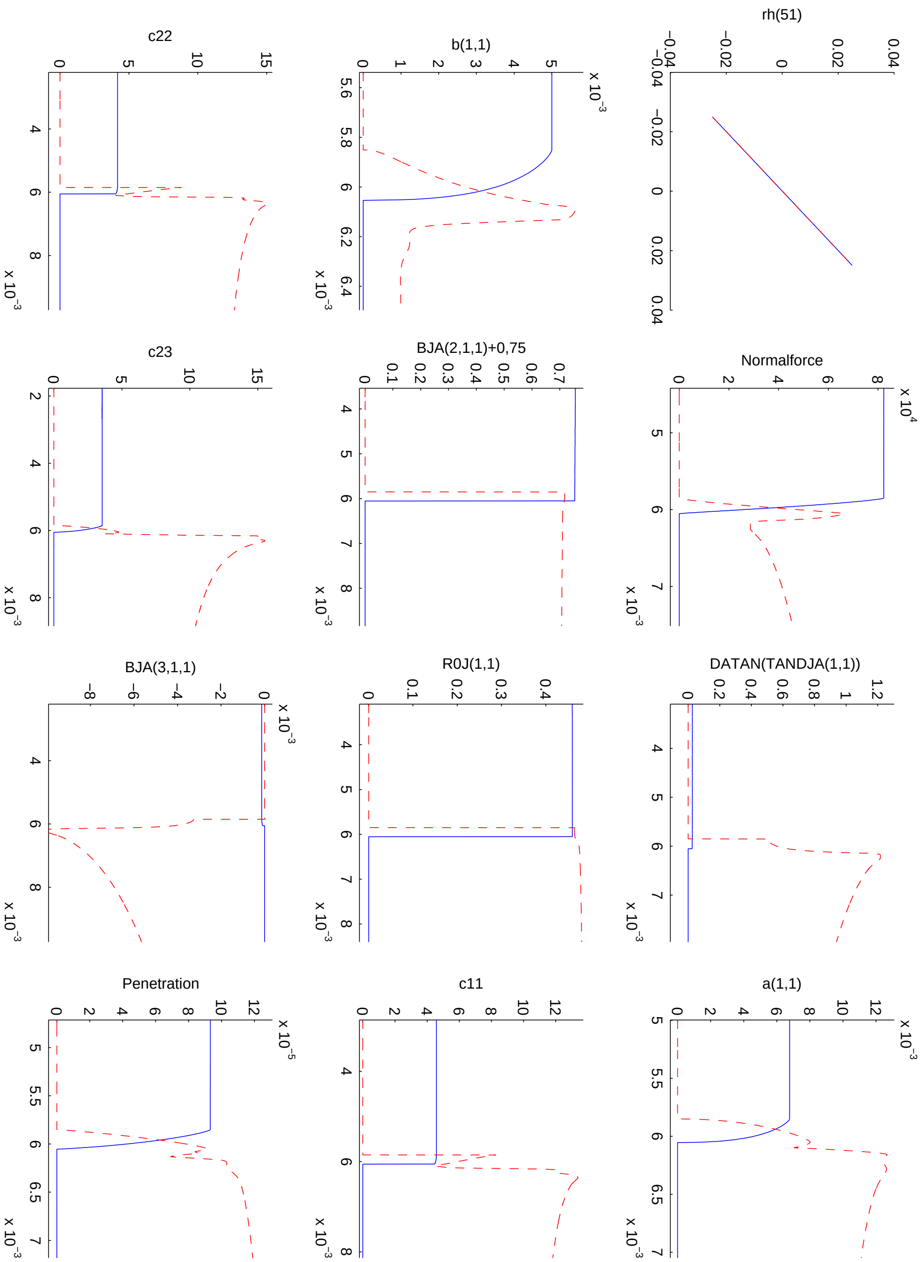


Figure 4.10: RSGEO values versus the displacement of the wheel set for the combination of wheel profile GV40 and rail UIC60 with cant 1/20 toward the center of the track. The shifting of the load between the two contact points (the continuous one on the wheel tread and the dashed one on the flange) is highlighted.

CHAPTER 5

Non-Linear Dynamics

For a long time static analysis was used in order to address problems like derailment and ride comfort. Even if this approach looks conservative, it has shown shortcomings in many situations where the dynamics of the system contributed as well. The use of dynamic analysis allows to explain problems like the hunting motion, but also to better explain derailments.
The hunting motion is a sideway oscillation of the wagon that appears above a certain speed. This phenomenon was first observed by Stephenson during the beginning of the 19th century[Ste21], but dynamic theory was not developed enough to understand the factors causing the hunting motion, thus a critical speed was set for each design and the maximum speed of the train was limited by this speed rather than by the power of the engine. Even if the causes were unknown, the consequences of the hunting motion were already clear: bad ride comfort, worse safety of the transported goods, higher wear of all the mechanical components, higher risk of derailment. The improvements in the field of non-linear dynamics and the application of these theories to railway vehicle dynamics[Tru99] has provided better explanations to the hunting phenomenon. The factors causing the hunting motion are several, such as the wheel-rail geometries, the coefficient of friction and the vehicle design. A vehicle running on a straight track usually finds its stable position in the middle of the track. This is true when the vehicle is running at low speeds and the guidance forces of the wheel sets dominate the motion and are gradually damped down. Wrong design of the suspensions and the bodies could allow the guidance motion to hit

a resonance frequency of the system, causing a sideway motion with frequency of 1-2Hz. The increase of the speed can cause the creep forces to start dominating the motion, determining the hunting phenomenon. On a straight track this should appear as a sideway motion around the centerline of the track with frequencies higher than 2Hz. The relation between the creep forces and the hunting phenomenon has been studied also analytically for simplified models[ABS07, Ch.8]. Its evident that a proper suspension system, that damps down the hunting frequencies, can improve the stability, increasing the critical speed.

In non-linear dynamics terms, the dynamical system undergoes a Hopf bifurcation with one or more stable attractors. The critical velocity comes in two different forms: the linear critical velocity and the non-linear critical velocity. The first is higher and the exceeding of such speed causes the starting of the hunting motion. The second is lower and does not necessarily cause the starting of the hunting motion. A particular perturbation of the system is necessary to start the hunting oscillation when the train is running between the non-linear and the linear critical speed. For security reasons, the top speed for a certain design is set depending on the non-linear critical speed rather than on the linear one. A bifurcation comes always with a change in the position of the eigenvalues of the Jacobian matrix of the system in the centered position for straight tracks or in its stable not hunting position in curves. The eigenvalues of the Jacobian determines whether the central solution is stable or unstable[NB95, Ch. 2]. If all the eigenvalues lay in the left-half of the complex plane ($\Re(\lambda_i) < 0$ for all the eigenvalues), the central solution is stable and the directions of the corresponding eigenvectors are contracting, thus the railway vehicle returns to its stable solution along the track centerline. If one of the eigenvalues lays in the right-half of the complex plane ($\Re(\lambda_i) > 0$ for one the eigenvalue), the central solution is unstable and depending on the characteristics of these eigenvalues several things can happen. In train dynamics the hunting phenomenon is associated to the crossing of the imaginary axis by a pair of conjugate eigenvalues. Figure 5.1 shows the subcritical Hopf bifurcation that is typical in railway dynamics. The amplitude of the oscillation is plotted against the speed: the attractor with high amplitude represents the hunting, thus it will be called the hunting attractor in the following. The linear and non-linear critical speeds are distinct and the hunting attractor can be entered well below the linear critical speed, as a consequence of a perturbation of the system due, for example, to track irregularities. Since the system taken in account has multiple degrees of freedom, the kind of perturbation necessary to enter the hunting attractor can be a combination of perturbations on the degrees of freedom, and not just a single sideway perturbation. This make the identification of the non-linear critical speed an active field of study. Particular configurations of the system can also cause the presence of multiple hunting attractors at different speeds, making the identification of the critical speeds even more complex.

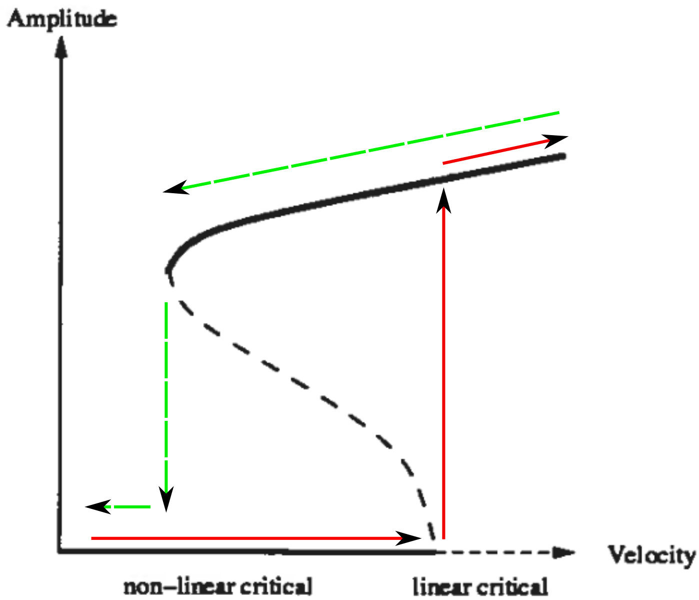


Figure 5.1: Subcritical Hopf bifurcation. The amplitude of the sideway oscillation is shown versus the speed. The stable attractor with big amplitude can be reached by the system due to a perturbation, even at speeds well below the linear critical velocity. The arrows represent the adiabatic increase of the speed and the adiabatic decrease of the speed used during the simulations in order to detect the linear and the non-linear critical speeds.

Some research on the hunting problem on curved tracks[HP02] have shown that both the critical speeds decrease in comparison with the speeds on straight track. It was also found that the linear and non-linear critical speed can match, transforming the subcritical Hopf bifurcation to the supercritical Hopf bifurcation shown in figure 5.2.

The analytical identification of the critical speeds of a complex system like a train is cumbersome. Thus, the only available tools in order to find them are numerical simulations. A simple simulation is represented by the wagon model running for a certain period at constant speed on a track that can be straight or curved. A lot of simulations like this can be performed with increasing speed obtaining the growing bifurcation branch shown in figure 5.1: when the linear critical speed is reached, the system will automatically jump to one of the hunting attractors. When the state of a dynamical system is in the domain of attraction of an attractor, it will stay there unless a particular perturbation is

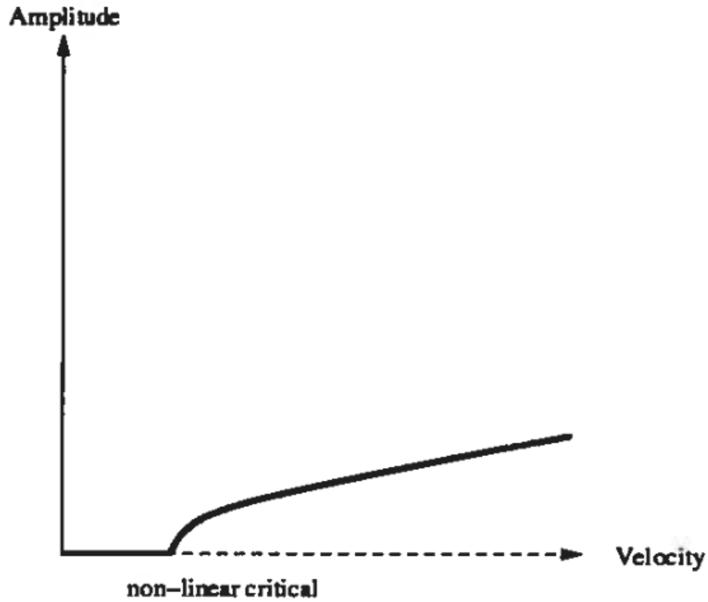


Figure 5.2: Supercritical Hopf bifurcation. The amplitude of the sideway oscillation is shown versus the speed. The linear and the non-linear critical speeds match in a unique bifurcation point.

applied. In order to find the non-linear critical speed, the simulations will be started with a state of the system inside the domain of attraction of the hunting attractor. The speed can then be decreased adiabatically (see fig. 5.1), until the system automatically stabilizes, determining the value of the non-linear critical speed.

The hunting stability problem is not the only one addressed using dynamical systems analysis. Other factors, like the acceleration on the car body or the instantaneous forces on the track, can be observed in order to assess the ride comfort and the risk of derailment.
Ride comfort is a difficult topic to be treated, because it can be affected by several factors (vibrations, noise etc) that sometimes are subjective. Anyway the most important factors that determines ride comfort are the accelerations inside the car body: the smaller they are the better is the ride experience.
Over more than a century of research, some relations between the wheel-rail contact forces and the risk of derailment have been found. These relations are still questioned because most of them lays on the concept of static analysis and therefore are thought to be the most conservative. However there are ev-

idences that sometimes these concepts fall short in explaining the derailment phenomenon, either because they don't account for dynamic effects or because they underestimate other parameters. The most widely used estimator for the risk of derailment is the Nadal's ratio[Nad08] between the lateral force Y and the vertical force Q for each wheel. Another factor that can be relevant for the risk of derailment is the angle of attack of the wheel sets. Usually the wheel sets have a tendency of over steering when covering a curve.
Some other dynamic values that can be interesting to observe are the clearances of the bumpstops and the change in behaviors of the non-linear elements. These sudden changes, in addition to make the solution computationally harder, can strongly affect the movement and the behavior of the whole system.

CHAPTER 6

Results

Newly implemented programs need always some verification and validation tests in order to check that the code does exactly what is wanted by the developer and that the results agree with theories and real life experiments. The verification step is related to the implementations and so a line-by-line checking of the program needs to be performed, seeking for errors and unwanted behaviors. The validation step is carried out by plausibility tests: results are compared to the physical expectation. The checking of consistency between the lateral forces on the rail and the lateral acceleration in curve due to the mass of the train is one of these tests. The distribution of the load on the internal and external wheels along a curve can be checked as well as the correct displacement of the car body due to centrifugal accelerations. All these plausibility tests have been performed looking at gross results. Further validation steps have been taken by comparing the results with previous works or, when possible, with real data. This steps are part of this section as well as new results obtained for models that were never investigated.

The DYnamic Train SImulation program has been used in order to test different wagon models running on straight track or over a curve. Transient analysis and bifurcation analysis have been employed to study the stability characteristics and the ride comfort of these models.

The amount of data generated for each simulation is quite big and, in order to plot everything clearly, a particular notation has been used for the degrees of freedom. Each component is referred with its own name: `CarBody`, `LeadingBogieFrame`, `TrailingBogieFrame`, `LLWheelSet`, `LTWheelSet`, `TLWheelSet`, `TTWheelSet` (where the wheel sets are listed from the first to the last of the wagon). Each degree of freedom of each component is addressed as `<Component>[ID]::<DOF>` where `DOF` can be one of the degrees of freedom of the particular component (`Y`, `YDOT` etc) and the ID is the unique identifier assigned to the component. Additionally to the degrees of freedom that describe the motion of the system, all the forces can be plotted as well. The following notations are used for the different kinds of forces:

- `<Component>[ID]::GRAV-<DOF>`: Gravitational forces (usually constant)
- `<Component>[ID]::CENT-<DOF>`: Centrifugal forces
- `<Component>[ID]::CONN-<DOF>`: Suspension forces (the sum of all the forces due to the suspensions on one component)
- `<WheelSetComponent>[ID]::C-<Side>-<DOF>`: Creep forces acting on the wheel on the side `<Side>` .
- `<WheelSetComponent>[ID]::N-<Side>-<DOF>`: Normal forces acting on the wheel on the side `<Side>` .

All these information can be plotted against five different parameters:

- `Time`: the time of the simulation
- `ExternalComponent::V`: The speed of the vehicle
- `ExternalComponent::R`: The radius of the curve
- `ExternalComponent::Phi_se`: The cant of the track in radians
- `ANC`: The uncompensated acceleration due to the centrifugal forces in curve:

$$ANC = \frac{v^2}{R} - \frac{c}{1500} \cdot g \tag{6.1}$$

  where c is the installed cant in millimeters and g is the gravitational acceleration.

Usually plots against time use continuous or dashed lines for representing the DOF or the force. The plots that show the dynamics and forces with respect to the external parameters (speed, radius, cant and ANC) display the variation of the DOF or the force by dots (e.g. fig.6.1a) or by two lines that mark the amplitude of variation (e.g. fig.6.7a).

6.1 The Cooperrider model

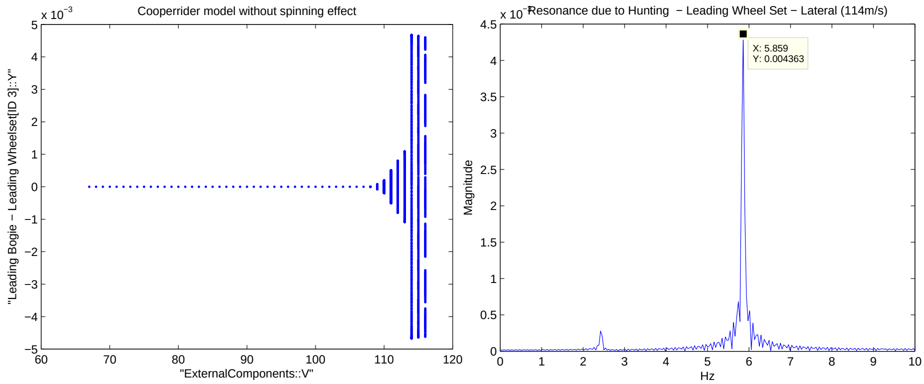
The Cooperrider model is a very common bogie frame model employed for studying the dynamics of railway vehicles. The dynamics of this model have already been studied in [Chr01], where straight track was assumed and the hunting phenomenon was related to the track perturbations, in [HP02], where the dynamics of smooth curved tracks was studied, and in [Big10], where different rail cants were considered. All the works focused on the dynamics of the bogie frame, neglecting the influence of the car body. This was done assuming the car body to be a fixed mass, with the roll as the only degree of freedom. In this work the model has been extended to the complete model of four axles train wagon. The results will be compared with the previous works, in order to assess the influence of the car body on the dynamics of the train. The model employed is presented in appendix C.1. One major difference between the model employed and the model used in [Chr01, HP02] is that the torsional spring and damper in the secondary suspensions are substituted by yaw springs and dampers, usually employed in the industry. It's stressed that, so far, this model has only been used for simulation and research purposes and it has never been manufactured.

6.1.1 Comparison with previous results

Since the Cooperrider model has been already widely used for research purposes, the model proposed in this work can be tested against the previous results obtained in [Chr01, HP02, Big10]. All the previous works considered a half-wagon where all the degrees of freedom of the car body were fixed but the roll. Thus, the dynamics were governed mostly by the bogie frame, the wheel sets and their suspensions. Furthermore the models used in those works didn't consider the precession effects due to the fast spin of the wheel sets.

The appropriate modification to the model implemented in DYTSI are done: the right hand side function for the car body is set to zero, so that the motion of the car body will be neglected. In order to avoid the coupling of the motion of the leading and trailing bogie frames, also the degrees of freedom relative to the roll motion of the car body are set to zero. This will give different results from the previous works, but still comparable. Furthermore the effects due to the spinning of the wheel set (precession forces) are neglected (see eq. (2.42)). The comparison is done on the results obtained on straight track, where preceding works found the linear critical speed at 117.3m/s and the non-linear critical speed at 53.97m/s.

Figures F.1-F.5 show the dynamics of the bogie frame and the two wheel sets,

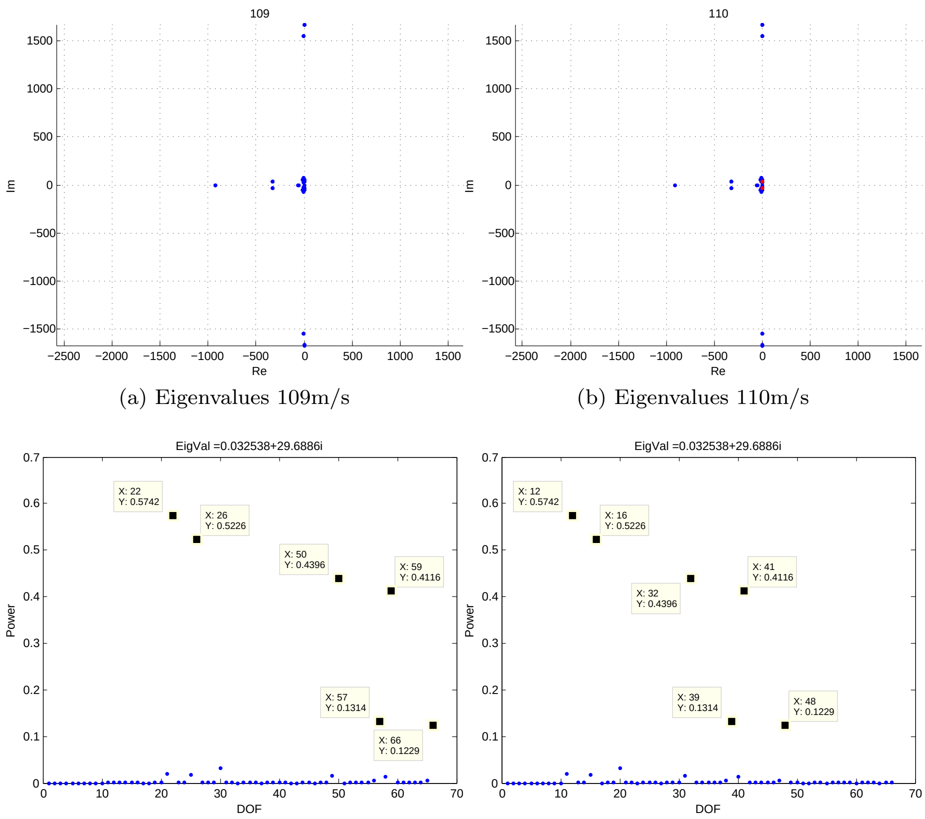


(a) Amplitude of Lateral Displacement (b) Displacement spectrum at 114m/s

Figure 6.1: Amplitude of the lateral displacement of the leading wheel set in the steady state against the running speed of the vehicle. The linear critical speed has been found to be 114m/s. The spectrum of the lateral displacement of the leading wheel set shows that the dominating frequency is 5.859Hz, more or less the same found in preceding works.

and the contact forces acting on the wheel sets against the increasing speed. The abrupt change in dynamics can be also checked in figure 6.1a, where linear critical speed was found at 114m/s. Figure 6.1b shows the frequency spectrum about the lateral motion of the leading wheel set when the model is hunting. The dominant frequency is around 6Hz, confirming the results obtained in [Chr01, HP02]. A further investigation of the dynamics can be done by mean of the eigenvalues on the stable central position. The central solution is expected to pass from stable to unstable around the linear critical speed. The figures 6.2a and 6.2b show that two couple of conjugate eigenvalues cross the imaginary axis at the speed of 110m/s, that is then identified as the linear critical speed. Figures 6.2c and 6.2c show that the modes that determine the loss of stability by the system are guided by the lateral displacement and the yaw of the wheel sets. In this case each couple of eigenvalues belongs to one of the bogie frames, whereas [Chr01, HP02] found only one couple of eigenvalue because only one bogie frame was considered.

The last state of the simulation with the model running at 114m/s is used for starting the adiabatic decrease of the speed, seeking for the non-linear critical speed. Figures F.6-F.10 show the dynamics of the bogie frame and the two wheel sets, and the contact forces acting on the wheel sets. The system recover its stability when the speed is decreased to 56.5m/s and figure 6.3a show that the leading wheel set gets back to its stable centered position. Figure 6.3b



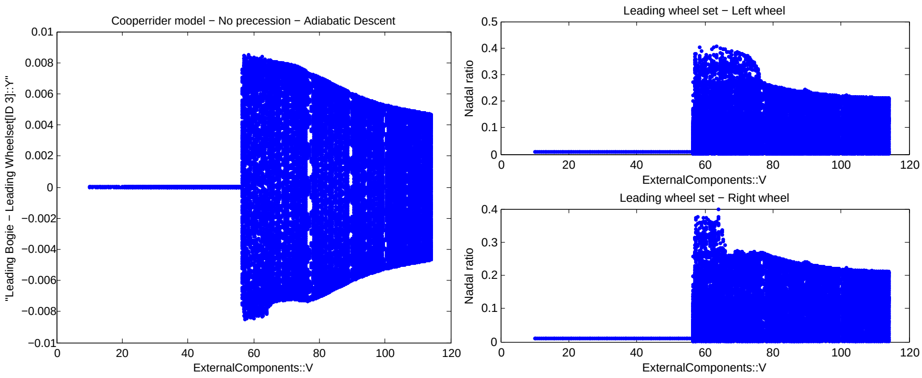
(a) Eigenvalues 109m/s (b) Eigenvalues 110m/s

(c) First Eigenmode 110m/s (d) Second Eigenmode 110m/s

Figure 6.2: Eigenvalue migration due to hunting. When the model runs at 110m/s two couple of eigenvalues cross the imaginary axis. The eigenmodes of these eivenvalues show that the main affected degrees of freedom are the lateral displacement and the yaw of the wheel sets, as well as the lateral displacement and the roll of the bogie frames.

shows that the Nadal's ratios increase drammatically when the train is running in its hunting limit cycle. Furthermore, the amplitude of the displacement of the wheel sets is close to 0.009 where the flange contact starts (see fig.4.9).

Finally the linear and non-linear speeds can be plotted in a unique bifurcation diagram where only the maximum amplitude of the steady states are plotted. Figure 6.4 shows the bifurcation diagram for the lateral displacement of the leading wheel set. The results from the adiabatic increase and the adiabatic decrease of speed have been concatenated, obtaining this complete figure. There



(a) Amplitude of Lateral Displacement (b) Nadal's ratio for leading wheel set

Figure 6.3: Amplitude of the lateral displacement of the leading wheel set in the steady state against the decreasing speed of the vehicle. The non-linear critical speed has been found at 56.5m/s. The Nadal's ratio increases dramatically when the train is running in the hunting limit cycle.

is a significant displacement between the linear critical speed and the non-linear critical speed, and a perturbation could make the system jump to one or the other attractor. It is stressed that the separatrix between the hunting limit cycle and the zero solution limit cycle is not only two dimensional, but multi-dimensional, thus multiple combinations of perturbations can cause the system to jump to the hunting limit cycle. There are also evidence that in some cases a single big perturbation of the system is not sufficient either. For example, let's perform a transient analysis of the system running at 70m/s. From figure 6.4 it looks like a perturbation of the leading wheel set of $8 \cdot 10^{-3}$m would be sufficient for entering the hunting limit cycle. Figures F.11-F.13 show the dynamics of the bogie frame and the two wheel sets, whereas figure 6.5 focus on the lateral dynamics of the leading wheel set, the one perturbed. The model regain the central stable solution in spite of the strong perturbation applied.

Tests on straight track have been repeated using an improved model which consider precession forces due to the high speed spinning of the wheel sets (see eq. (2.42)). No noticeable differences in behaviors have been observed in the two cases. It is stressed that, since the tests are performed on straight track (approximated by $R = 10^{99}$), the most of the terms in (2.42) cancel out and don't affect the dynamics

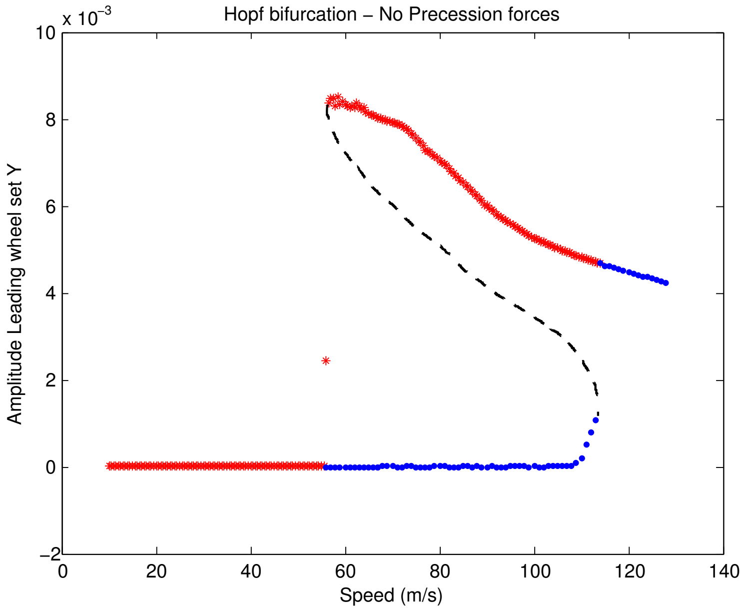


Figure 6.4: Bifurcation diagram for the lateral displacement of the leading wheel set. The dot values were found using the adiabatic increase of the speed, the asterisk values were found using the adiabatic descent from the hunting limit cycle. The dashed line has been added manually for indicating that there exists a separatrix between the hunting limit cycle and the zero solution limit cycle.

6.1.2 Dynamics on Straight Tracks

In the previous section the dynamics of a model where only the bogie frames and the wheel sets were free to move have been analyzed. Here the whole model will be considered first, in order to obtain the linear and non-linear critical speeds due to hunting. Both the models with and without precession forces have been tested. Since the results obtained using the two models don't show any relevant difference, only results using precession forces will be shown in the following.

The model is first tested on a straight track with a speed of $40\frac{m}{s}$(144km/h). The model is initially perturbed with a roll of 10^{-4}rad on all the wheel sets. Figures F.14-F.20 show the dynamics, the suspension forces and the contact forces for three components: the car body, the leading bogie frame and the leading wheel set attached to the leading bogie frame. All the components recover the track centerline position after the initial perturbation. For example figure 6.6a shows the lateral displacement of the trailing bogie against the time. The eigenvalues of the Jacobian matrix can be used as a confirmation of the fact that the system

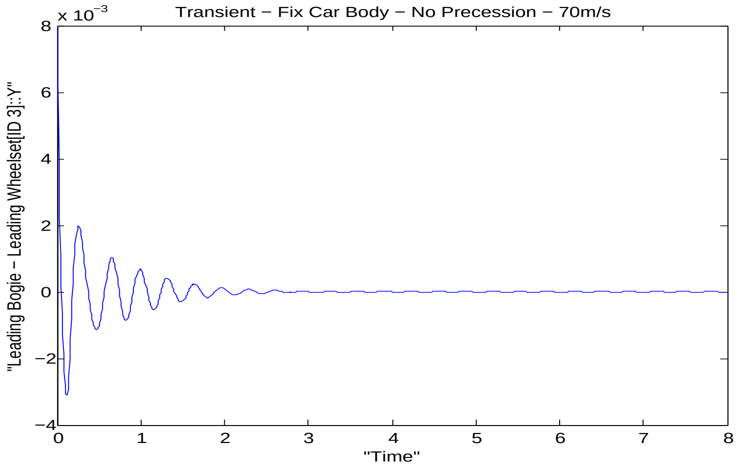
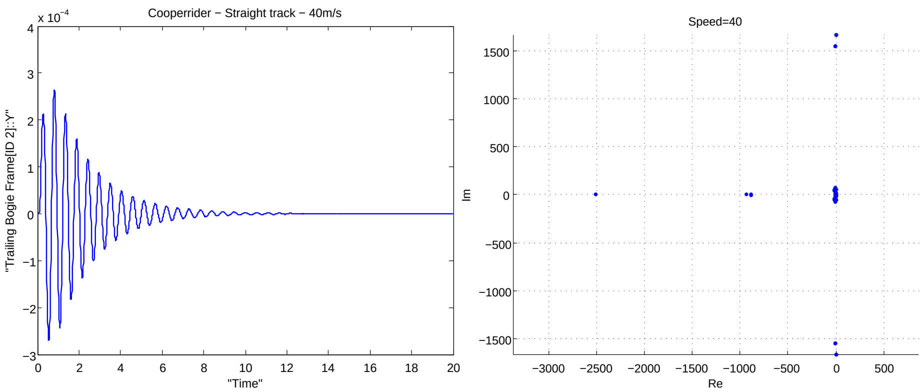


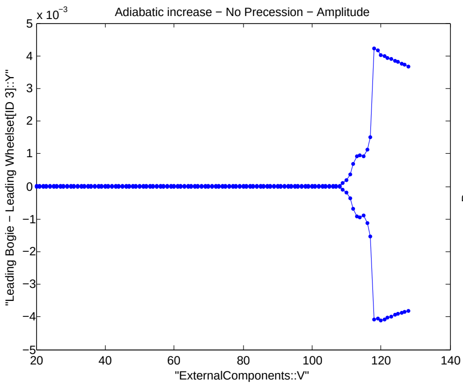
Figure 6.5: Lateral displacement of the leading wheel set after a strong perturbation has been applied. The model is running at 70m/s and is supposed to enter the hunting limit cycle but it does not.



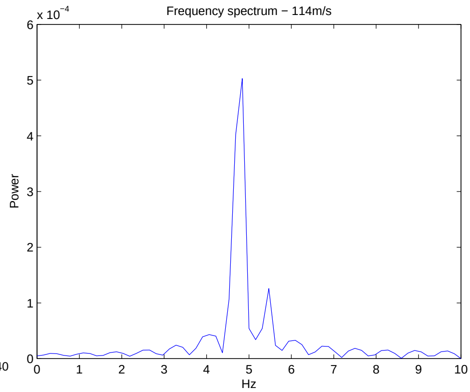
(a) Lateral Displacement (b) Eigenvalues

Figure 6.6: The lateral displacement of the trailing bogie frame of a Cooperrider vehicle running at 40m/s on straight track shows that the vehicle recover its stable centered position after being perturbed. The position of the eigenvalues confirms that the solution is stable, because all the eigenvalues have negative real part.

is stable when particular parameter of speed cant and radius are used. Figure 6.6b shows that all the eigenvalues are positioned in the left half of the complex plane, thus all the modes are contracting.



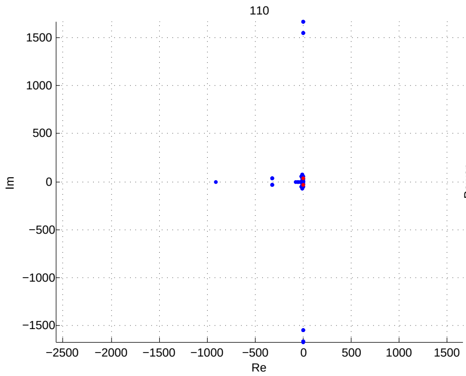
(a) Lateral displacement amplitude



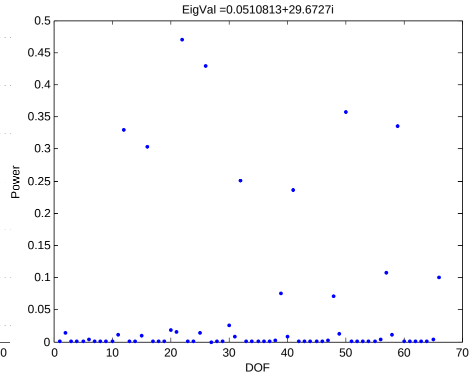
(b) Frequency Spectrum at 114m/s

Figure 6.7: Complete Cooperrider model. The amplitude of the lateral vibration of the leading wheel set attached to the leading bogie frame with respect to the running speed of the train shows that the hunting limit cycle is approached around 110m/s. The spectrum of the hunting motion for the leading wheel set shows that the driving frequency is 4.84Hz.

Now the model is launched with increasing speed on straight track and the amplitude of the variation of the degrees of freedom is taken in order to identify the linear critical speed. Figures F.21-F.27 show the dynamics of all the components of the model as the speed grows. Figure 6.7a show just the amplitude of the lateral vibration of the leading wheel set as the speed increase. Around 110m/s the system lose stability and is attracted by the hunting limit cycle. Figure 6.7b displays the spectrum of the lateral vibration of the leading wheel set: one of the effects of considering a complete model instead of a model with fixed car body is, then, to lower down the vibrational frequency from 5.86Hz to 4.84Hz. In order to find the exact speed at which the system lose stability, the eigenvalues of the Jacobian are investigated. Figure 6.8a shows that a couple of conjugate eigenvalues cross the imaginary axis when the speed of 110m/s has been reached. This result can be compared with the one obtained for the model with fixed car body, where two couple of conjugate eigenvalues crossed the imaginary axis. This because the model with fixed car body was then composed by two independent bogie frames and the observation of their modes (see fig.6.2) shows that the model was not coupled. Instead, when the complete model is considered, all the degrees of freedom are coupled and the eigenmode relative to the crossing eigenvalues is unique and involves degrees of freedom belonging to all the components. It is still remarkable that the driving degrees of freedom for the wheel sets are the lateral displacement and the yaw, whereas the lateral displacement and the roll are more involved in the bogie



(a) Eigenvalues 110m/s



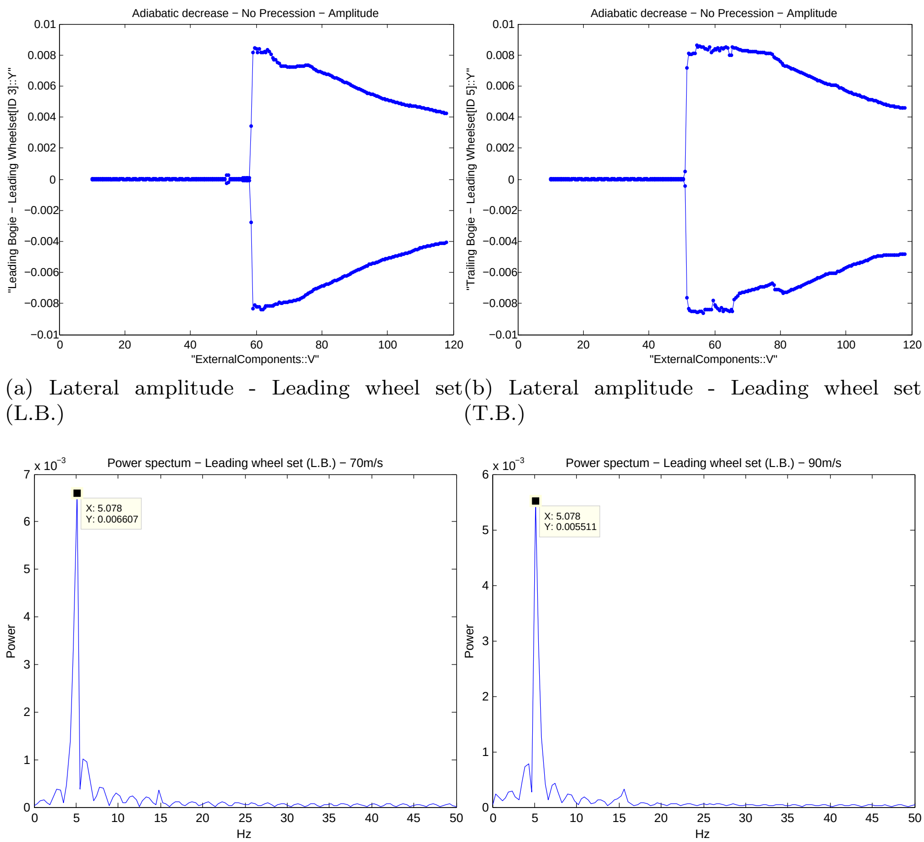
(b) First Eigenmode 110m/s

Figure 6.8: The eigenvalues for the complete model show that one couple of conjugate eigenvalues crosses the imaginary axis at the speed of 110m/s determining the Hopf bifurcation. The observation of the eigenmode relative to these eigenvalues show that the lateral displacement and the yaw of the wheel sets are relevant in the loss of stability.

frames. The observation of the eigenmode shows also that the trailing bogie frame and the wheel sets attached to it give a bigger contribution to this mode.

The knowledge of the linear critical speed and the behavior of the hunting limit cycle above this speed has been used in order to find the non-linear critical speed. The speed has been decreased adiabatically from speed 118m/s and the system has gained the central stable solution back at the speed of 51m/s. The figures F.28-F.34 show the effects of the adiabatic decrease of speed on the system. Figures 6.9a and 6.9b show the amplitude of the lateral displacement for the leading wheel sets attached to the leading and trailing bogie frame respectively. The two components behave differently as the speed is decreased, showing that the trailing bogie frame recovers the central stable solution at a lower speed respect to the leading bogie frame. The results obtained by the observation of the migration of the eigenvalues of the Jacobian agree with the dynamics and all the eigenvalues get back on the left of the imaginary axis at the speed of 51m/s.
The frequency response of the hunting phenomenon on the lateral direction has been investigated for different speeds and figures 6.9c and 6.9d show that the driving frequency is always around 5Hz for this design.

Using the results obtained with the adiabatic increase and the ones obtained with the adiabatic decrease of speed, the bifurcation diagram of the lateral displacement of the wheel sets in figure is constructed and shown in figure 6.10.



(a) Lateral amplitude - Leading wheel set (L.B.)

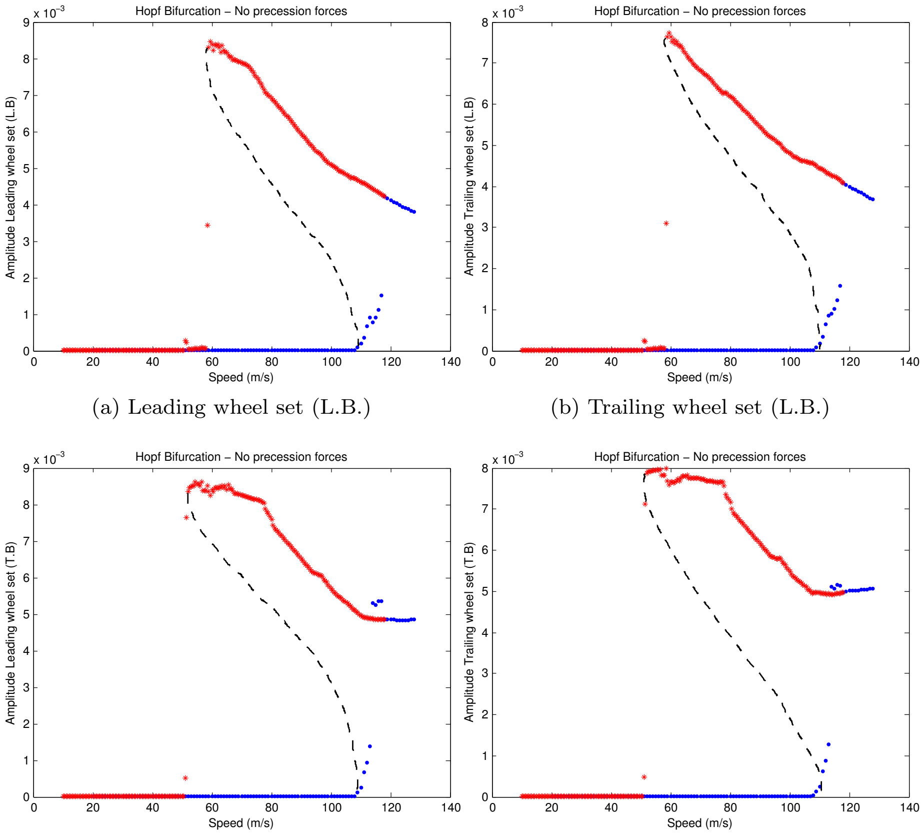
(b) Lateral amplitude - Leading wheel set (T.B.)

(c) Frequency Spectrum at 70m/s

(d) Frequency Spectrum at 90m/s

Figure 6.9: Complete Cooperrider model. The speed is adiabatically decreased starting form above the linear critical speed. The amplitude of the lateral displacement for the leading wheel sets of the leading and trailing bogie frames are shown. The frequency spectrum for different speeds has been found to be identical with the main frequency around 5Hz.

These results show that the linear critical speed for the complete model agrees with the linear critical speed found for the half model used in section 6.1.1. Instead, the non-linear critical speed is passed from 56.5m/s for the half model to 51m/s for the complete model. The components of the complete model present a diverse behaviors: it was already displayed in figure 6.9 that the wheel sets attached to the trailing bogie frame recovered the central stable position at a lower speed than the one for the wheel sets attached to the leading bogie frame. The test performed there was started from the speed of 118m/s where all the



(a) Leading wheel set (L.B.)

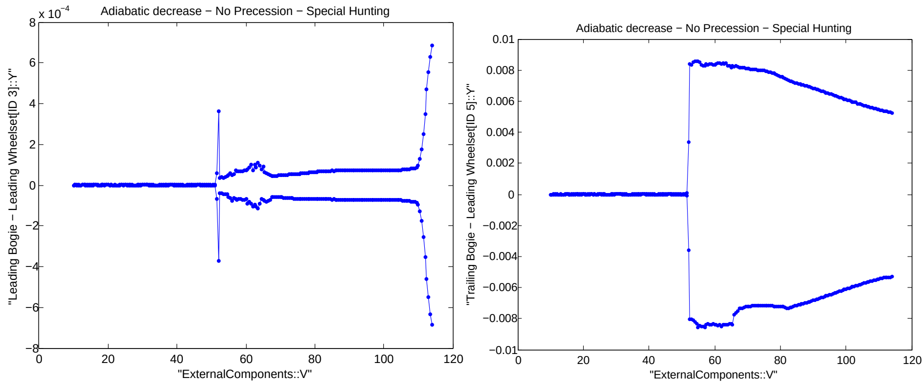
(b) Trailing wheel set (L.B.)

(c) Leading wheel set (T.B.)

(d) Trailing wheel set (T.B.)

Figure 6.10: Bifurcation diagram for the lateral displacement of the wheel sets. The dot values were found using the adiabatic increase of the speed, the asterisk values were found using the adiabatic descent from the hunting limit cycle. The dashed line has been added manually for indicating that there exists a separatrix between the hunting limit cycle and the zero solution limit cycle.

wheel sets where hunting. Another test was performed from the speed of 114m/s down to 10m/s. Figure 6.11 shows that the wheel sets in the leading bogie frame stop hunting shortly after reducing the speed and they are just affected by the vibration of the wheel sets in the trailing bogie frame, that are hunting. This shows that there exists the possibility of only a couple of wheel sets to be hunting and the other not. It also suggests that the system is not symmetric along the lateral axis in spite of the symmetry of the design, because of the direction of the speed. The investigation of the migration of the eigenvalues has already



(a) Leading wheel set (L.B.) (b) Leading wheel set (T.B.)

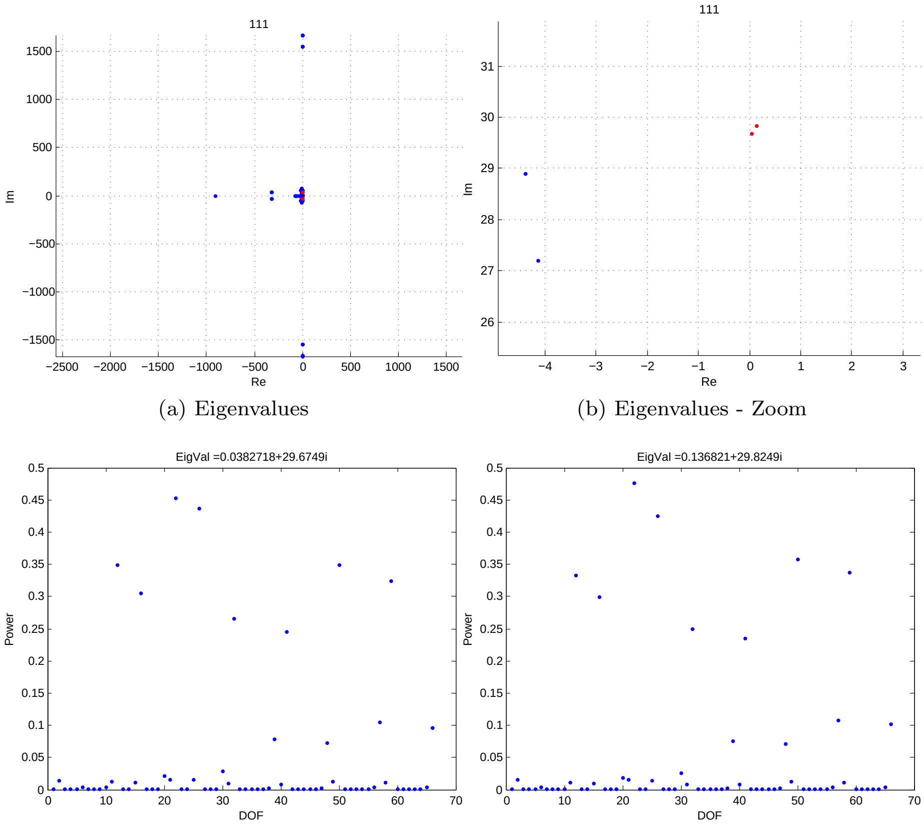
Figure 6.11: Adiabatic decrease of the speed starting from the hunting limit cycle at 114m/s. The wheel sets attached to the leading bogie frame behave completely differently from the wheel sets attached to the trailing bogie frame.

shown that a couple of conjugate complex eigenvalues cross the imaginary axis determining an Hopf bifurcation at the speed of 110m/s(see fig.6.8). Further analysis has proved that another bifurcation happens right after at the speed of 111m/s. The extra couple of conjugate complex eigenvalues is very close to the first couple. The position of the eigenvalues at the speed of 111m/s is shown in figure 6.12.

6.1.3 Dynamics on Curved Tracks

The dynamics of the Cooperrider model are now investigated when the train is running on a curved track. The complete model without precession forces is considered first. Precession forces will be discussed in the cases where results are significantly different. All the simulations on curved tracks are started with an initial transient curve, linear in the radius and the cant. The transient is used for approaching the curve smoothly and not to have sudden dangerous jumps to flange contact. The linear transient curve is not the best curve shape for approaching a curve, however the speeds at which the train will enter the curve are supposed to be low, because the main focus of this section will be on curves with constant radius and cant.

Figure 6.13 shows the lateral displacements of the wheel sets attached to the leading bogie frame as the radius gets smaller and the cant gets bigger. The simulation considers a train running at 20m/s approaching a curve of 600m of radius and 110mm of cant. The transient curve lasts for the first 20s and the



(a) Eigenvalues

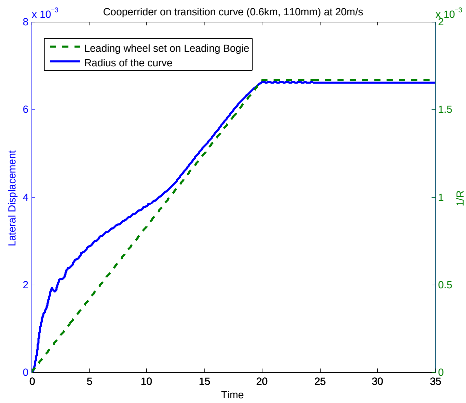
(b) Eigenvalues - Zoom

(c) First eigenmode

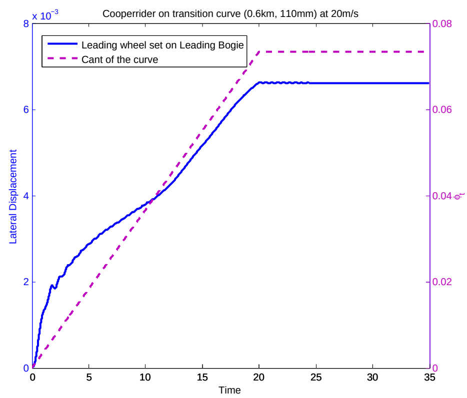
(d) Second eigenmode

Figure 6.12: Eigenvalues for the complete model running on straight track at the speed of 111m/s. Two couples of complex conjugate eigenvalues cross the imaginary axis. The two eigenmodes of these eigenvalues are shown as well.

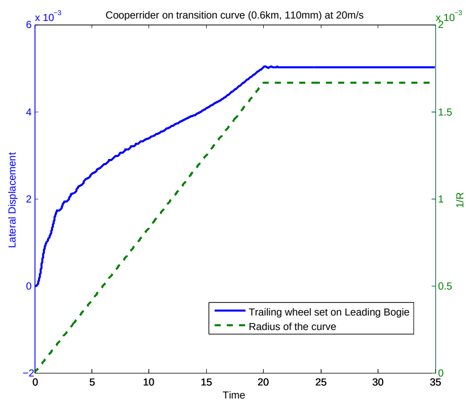
simulation is then continued for 20s more with constant curve and radius. The approach to the curve is smooth and there is no relevant overshooting with respect to the stable position on the constant curve. The wheel-rail profiles are such that the stable position is found far away from the flange contact. Furthermore, the leading and the trailing wheel sets find different equilibrium positions, where the leading wheel set is more displaced than the trailing one. The wheel sets attached to the trailing bogie frame show the same behaviors. Figure 6.14 shows the attack angles of the wheel sets as the train cover the transient curve. The angles displayed are given in radians and, accordingly to the reference system used and the right-hand grip rule (see sec.2.1, fig.2.4), a negative angle denotes a rotation toward the center of the right hand curve.



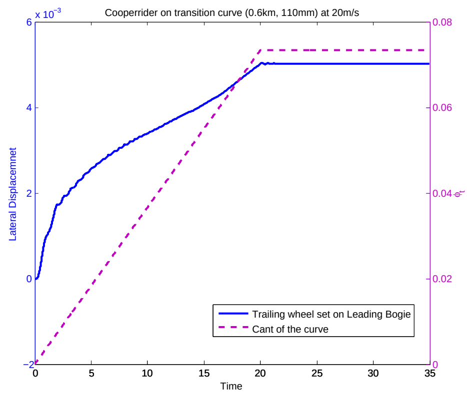
(a) Leading wheel set (L.B.) - VS radius



(b) Leading wheel set (L.B.) - VS cant



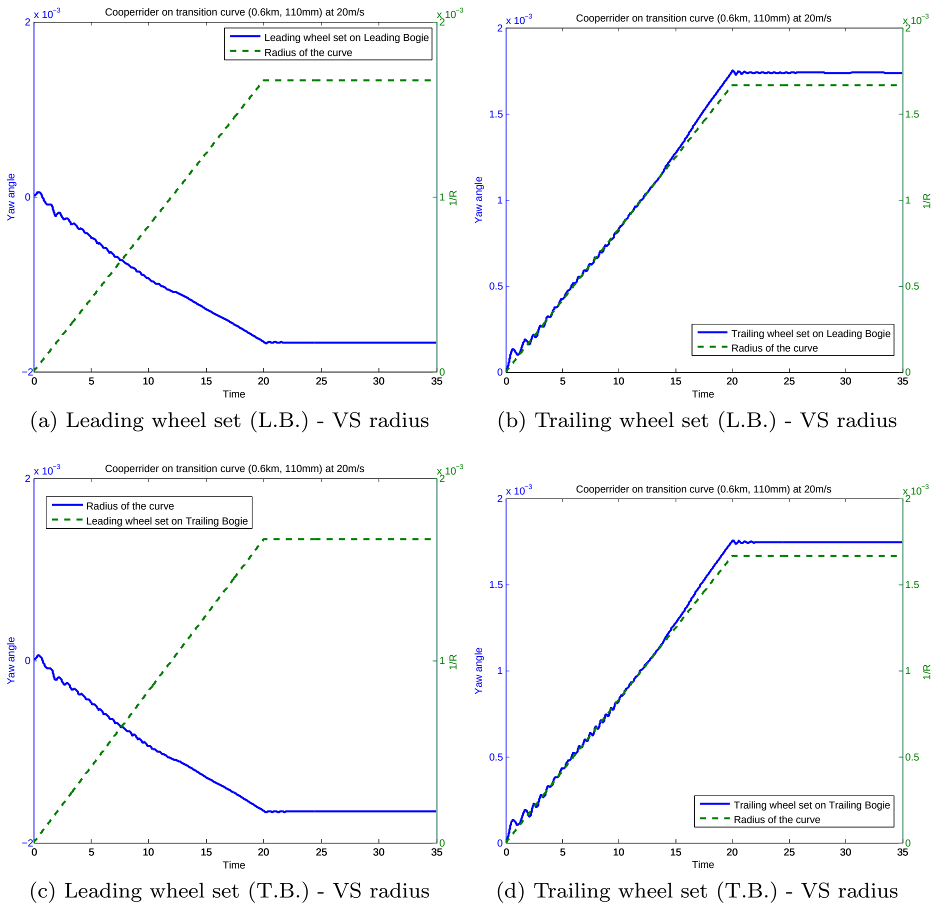
(c) Trailing wheel set (L.B.) - VS radius



(d) Trailing wheel set (L.B.) - VS cant

Figure 6.13: Lateral displacement of the leading and trailing wheel sets attached to the leading bogie frame (L.B.) for the train running at 20m/s. The inverse of the radius and the cant change linearly to 0.6km and 110mm respectively, as the wheel sets find their stable position away from the center of the track. The leading and the trailing wheel sets of the trailing bogie frame find similar positions to the one displayed here.

The leading wheel sets of each bogie frame turn toward the center of the curve whereas the trailing wheel sets turn in the opposite direction with the same angle. The angle of attack is important for the study of derailment. In the rest of the work the angle of attack can be observed by looking at the degree of freedom ψ of each component. Also the bogie frames and the car body assume a position that is not parallel to the tangent of the curved track: figure 6.15 show that for this slow speed these components turn toward the outer part of



(a) Leading wheel set (L.B.) - VS radius

(b) Trailing wheel set (L.B.) - VS radius

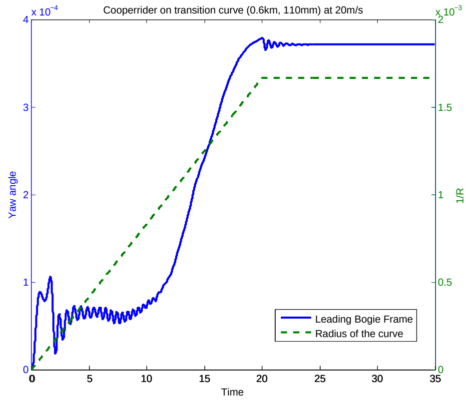
(c) Leading wheel set (T.B.) - VS radius

(d) Trailing wheel set (T.B.) - VS radius

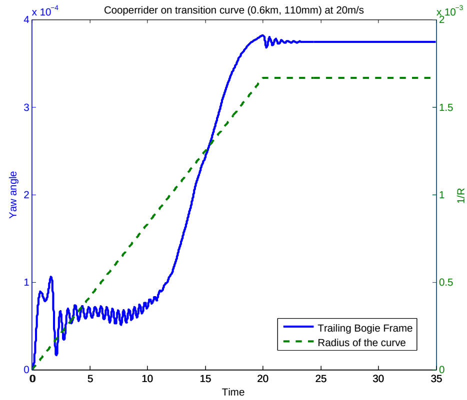
Figure 6.14: Yaw angle (radiant) of each wheel set for the train running at 20m/s. The inverse of the radius and the cant change linearly to 0.6km and 110mm respectively, as the leading wheel sets turns toward the center of the curve whereas the trailing wheel set turn in the opposite direction.

the curve.

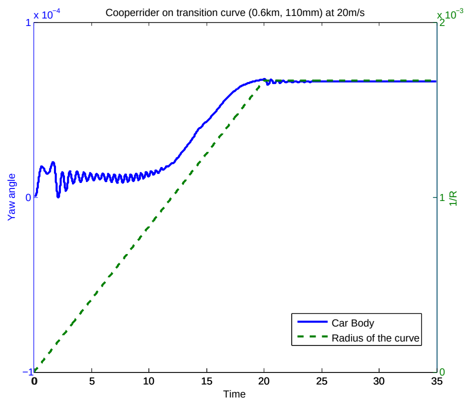
The model is now launched on different curves with different radius and with cant 110mm. The speed is adiabatically increased in order to check the effects of the uncompensated acceleration on the the dynamics of the train. Figures 6.16a-6.16b show the lateral dynamics of the leading and the trailing wheel sets as the speed is increased adiabatically in a curve with radius of 0.6km and cant of 110mm. The leading wheel set enters a resonance around 37m/s, when



(a) Leading bogie frame - VS radius



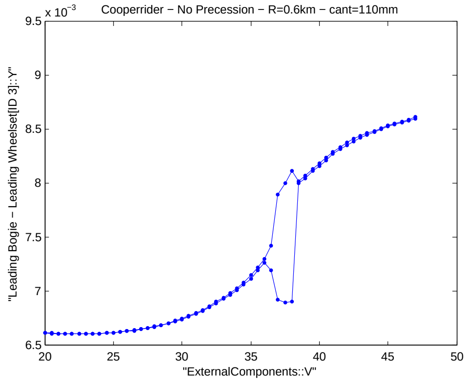
(b) Trailing bogie frame - VS radius



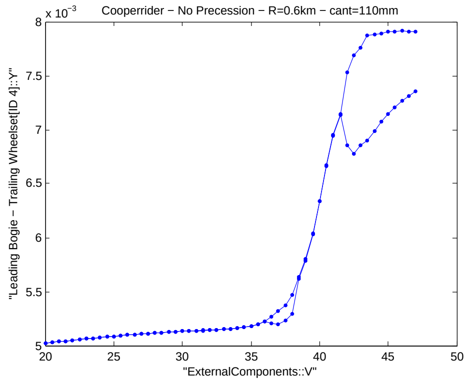
(c) Car body - VS radius

Figure 6.15: Yaw angle (radiant) of the bogie frames and the car body of the train running at 20m/s. The inverse of the radius and the cant change linearly to 0.6km and 110mm respectively, as the bogie frames and the car body turn outward for this low speed.

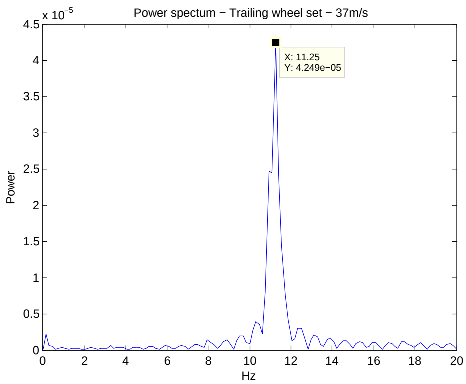
it migrates migrate toward the flange contact. The leading wheel set returns stable when the flange is hit. The frequency characteristic of this resonance is 11.25Hz as shown in figure 6.16c. The same phenomenon happens with the trailing wheel set, but at higher speeds. The resonance frequency in this case is 24.61Hz as shown in figure 6.16d.
Totally different behavior is obtained if a wider curve is considered. For instance, let's consider the dynamics of the train in a curve with radius 1.6km and cant 110mm. Figures 6.17a-6.17b show the dynamics of the wheel sets attached to



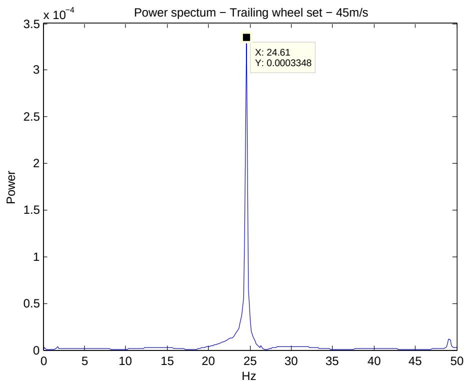
(a) Lateral dynamics - Leading wheel set (L.B.)



(b) Lateral dynamics - Trailing wheel set (L.B.)



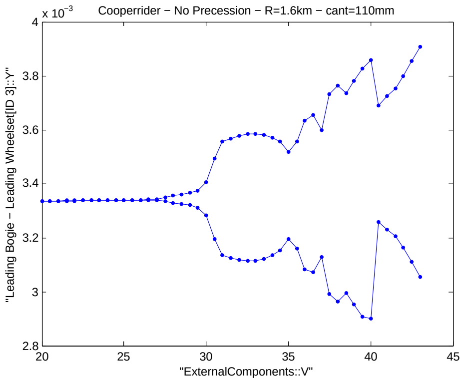
(c) Spectrum 37m/s - Leading wheel set (L.B.)



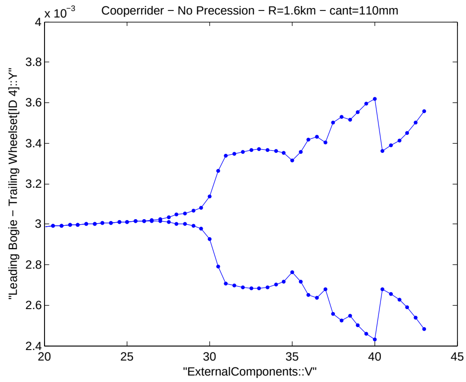
(d) Spectrum 45m/s - Trailing wheel set (L.B.)

Figure 6.16: The lateral dynamics of the wheel sets attached to the leading bogie frame have different behaviors as the speed is increased in a curve with radius of 0.6km and cant of 110mm. The driving frequencies of the resonances found for each wheel set is different as well.

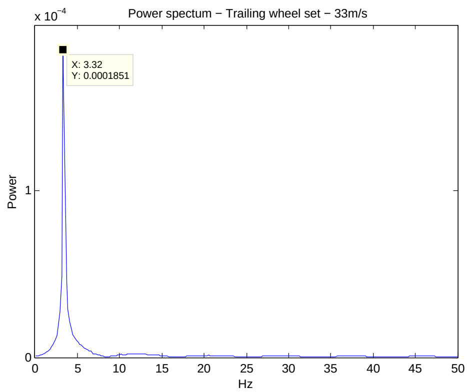
the leading bogie frame. The first observation is that the radius and the cant of the curve are such that both of them stay away from the flange contact. The stability of the system is broken relatively early, around 27m/s, with similar effects on the leading and the trailing wheel sets. This phenomenon is not very surprising: flange contact can be erroneously thought as an extreme event used only for pushing back the train when lateral forces are too big. However trains that run on flange contact along a curve show better stability than trains that don't. One reason is related to the contact forces: when there is flange contact the the guidance forces have a great influence on the system. Instead, when the cant is high enough to avoid flange contact, the guidance forces are



(a) Lateral dynamics - Leading wheel set (L.B.)



(b) Lateral dynamics - Trailing wheel set (L.B.)

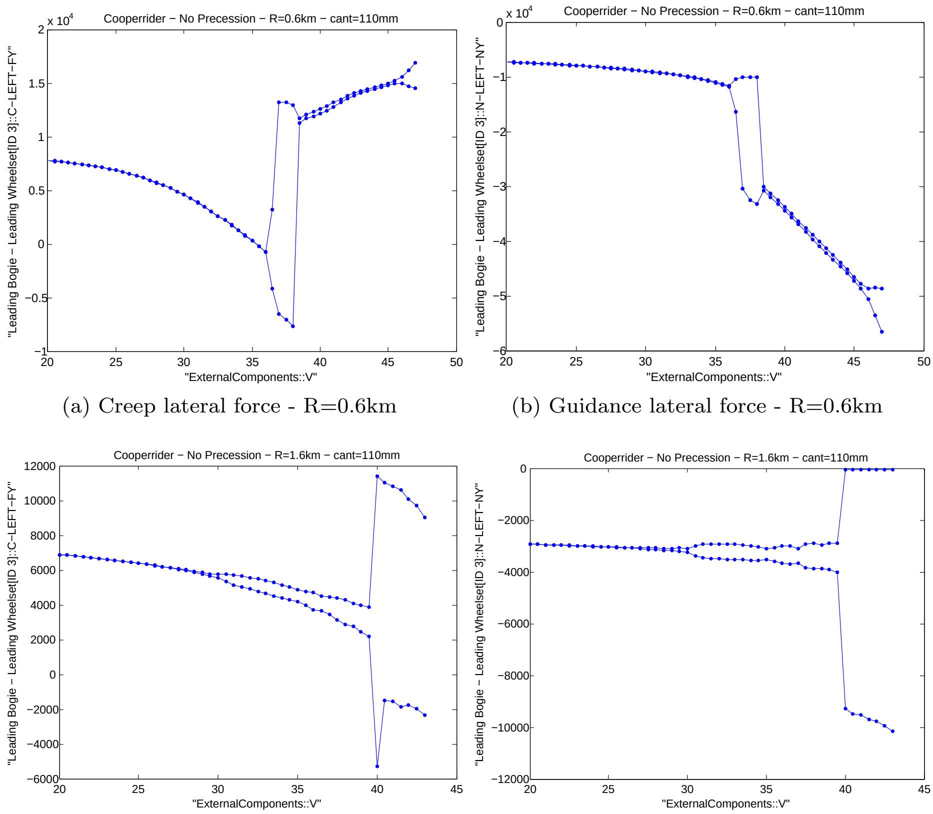


(c) Spectrum 37m/s - Leading wheel set (L.B.)

Figure 6.17: The lateral dynamics of the wheel sets attached to the leading bogie frame have different behaviors as the speed is increased in a curve with radius of 1.6km and cant 110mm. The driving frequency of the hunting motion is 3.32Hz.

smaller and the creep forces drive the system. Figures 6.18a and 6.18b show that for tight radius where the flange is hit already at relatively low speeds, the creep forces and the guidance forces start being of the same magnitude but the guidance forces get almost three times bigger than the creep forces as the speed is increased. On the contrary, figures 6.18c and 6.18d show that the creep forces are more than double of the guidance forces for all the speeds bigger than 27.0m/s where the critical speed was detected.

The model has been tested on several curves with different radius and with cant



(a) Creep lateral force - R=0.6km

(b) Guidance lateral force - R=0.6km

(c) Creep lateral force - R=1.6km

(d) Guidance lateral force - R=1.6km

Figure 6.18: Lateral component of the contact forces on the left wheel (external) of the leading wheel set attached to the leading bogie frame.

110mm (approx. 4^o). The radius considered are in the range between 0.6km and 2.0km. The dynamics of some of these settings are shown in figures F.35-F.62. These tests were used in order to identify the linear and non-linear critical speed when it was possible.

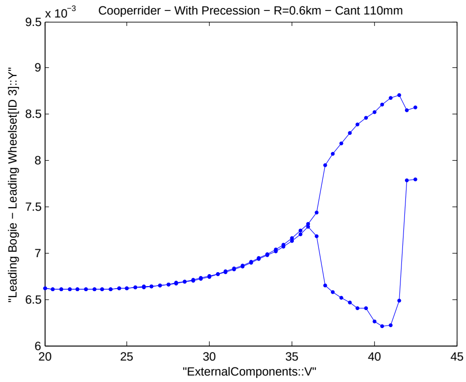
For tight curves with radius between 0.6km and 1.2km, the wheel sets move sharply toward the outer rail and the flange hits the rail, providing stability to the system. As shown in figure 6.16 the wheel sets get slightly disturbed during this transition toward the outer rail. From the ride comfort perspective, the car body doesn't get disturbed by these motions as can be checked, for example, in figure F.35. However, in tight curves the lateral forces applied on the external rail are big and this compromises safety. The Nadal's ratio for the outer wheels in figure F.63 show that the risk of derailment increase significantly with the

speed.
For wider curves with radius from 1.4km to 2.0km, the wheel sets keep a more central position because of the smaller lateral acceleration due to the centrifugal force. The cant of the track, that distributes some of the lateral acceleration perpendicularly on the track, causes the wheel-rail mutual penetration to be bigger, that in turns increase the size of the creep forces, moving the linear critical speed at lower speeds than for the straight track setting. In particular the following values have been found for the linear and non-linear critical speeds.

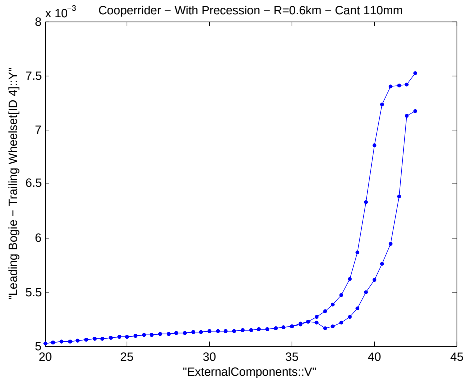
Radius	Lin. Crit. Speed	Non Lin. Crit. Speed
1.4km	25.0m/s	25.0m/s
1.6km	27.0m/s	27.0m/s
1.8km	20.0m/s	20.0m/s
2.0km	28.0m/s	26.5m/s

The subcritical Hopf bifurcation, characteristic of the hunting phenomenon on straight track, becomes a supercritical Hopf bifurcation for all the curves but the one with radius 2.0km where a very little fold is present in the bifurcation diagram. The effect of the hunting was already shown in figure 6.17. In these cases, the relatively low frequency vibration caused by the hunting affects also the stability of the car body (e.g. fig.F.49 and F.56). On the other hand the Nadal's ratio for the outer wheels is smaller than for tight curves (see fig.F.64), suggesting a lower risk of derailment in wide curves.

The model that considers the precession forces has been tested as well with the same combinations of radii and cant. In a curved track the symmetry axes of the wheel sets change direction due to the track geometry and thus all the terms in (2.42) contribute to the precession forces. The dynamics of some of the radius-cant combinations are displayed in figures F.65-F.92.
For wide curves the results are similar to the one obtained for the model without precession forces: the linear critical and non-linear critical speeds don't change for the radius and cant considered. The critical speeds were already low for the model without precession forces and it's possible that these forces don't affect the model at these low speeds. However, as the speed is increased, the effect of the precession forces adds up to the hunting phenomenon, resulting in vibrations with bigger amplitude.
For tight curves the precession forces are amplified by the rapidly changing orientation of the symmetry axis of the wheel sets. Figure 6.19 shows the dynamics of the leading and the trailing wheel sets attached to the leading bogie frame. The temporary perturbation due to the motion of the leading wheel set toward the outer rail, seen in figure 6.18a, is now more consistent and doesn't decay with the increasing speed.



(a) Lateral dynamics - Leading wheel set (L.B.)



(b) Lateral dynamics - Trailing wheel set (L.B.)

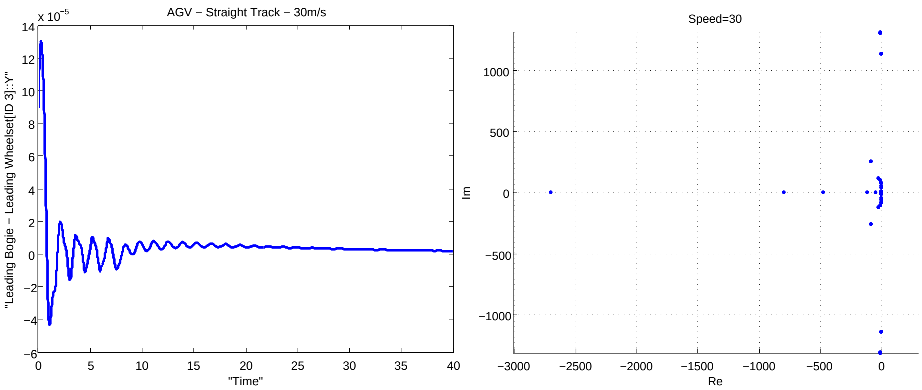
Figure 6.19: The lateral dynamics of the wheel sets attached to the leading bogie frame have different behaviors as the speed is increased in a curve with radius of 0.6km and cant 110mm. Precession forces cause a bigger perturbation as the wheel sets approach the flange contact.

6.2 ALSTOM very high speed power car[1]

The ALSTOM very high speed power car (VHSPC) is a model designed for running at speeds V$>$ 300km/h. The characteristics of this model are described in appendix C.2.

In order to check the validity of the constructed system, the model is run on a straight track with an initial perturbation. The relatively low speed chosen for the test is 30m/s (108km/h). The leading wheel set is displaced of 10^{-4}m and the simulation is run for 40s. Figures F.93-F.99 show the dynamics, and the main forces acting on the car body, the leading bogie frame and the leading wheel set attached to the leading bogie frame. The system converges to the stable solution at the track centerline. Figure 6.20a shows the lateral displacement of the leading wheel set of the leading bogie frame. The initial perturbation is visible at the beginning but it's correctly damped down by the suspension system. The stability of such solution is proved by the observation of figure 6.20b, that shows the eigenvalues of the Jacobian matrix that are all positioned in the left half of the complex plane.

[1]The design parameters of this model have been kindly provided by ALSTOM Transport.



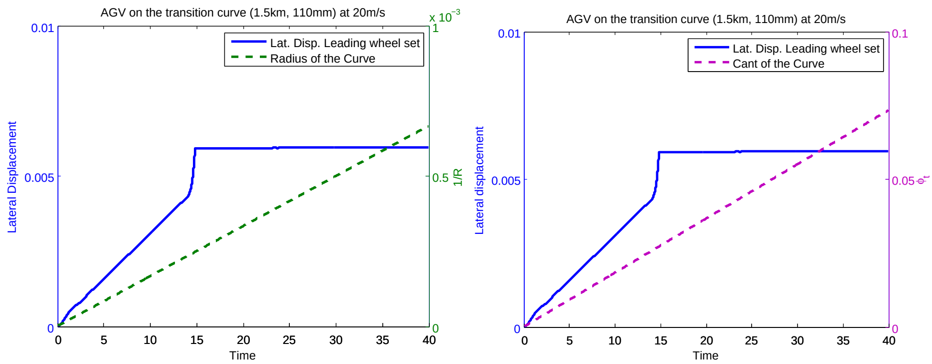
(a) Lateral Displacement (b) Eigenvalues

Figure 6.20: The lateral displacement of the leading wheel set of the leading bogie frame of a AGV vehicle running at 30m/s on straight track shows that the vehicle recover its stable centered position after being perturbed. The position of the eigenvalues confirms that this is a stable solution, because all the eigenvalues have negative real part.

6.2.1 Dynamics on Curved Tracks

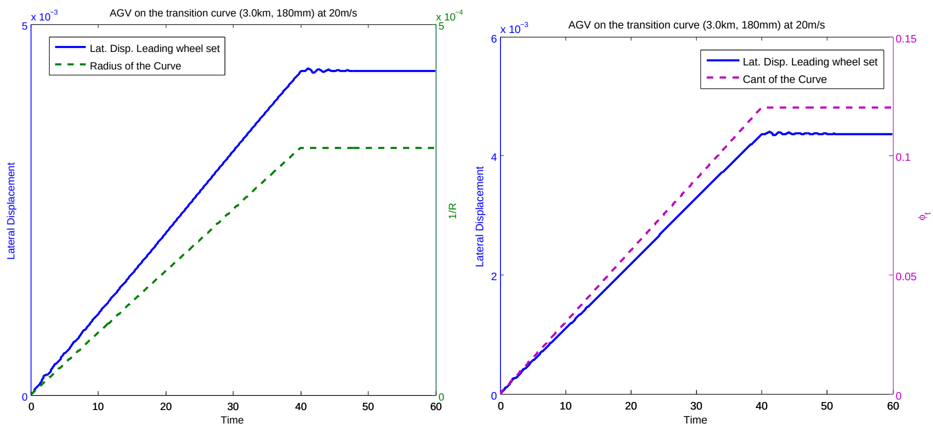
The analysis of the dynamics of this model in curves with high cant deficiency is necessary in order to establish its safety and ride comfort. High cant deficiency derives from the fact that the maximum cants mounted in railways is between 150mm and 180mm. Usual curves are canted between 110mm and 150mm. In special cases, for very high speed trains, bigger cants can be used, but this causes bigger wear of the rails and higher risk of derailment for freight wagons that travel at lower speeds.

The model is now prepared for running on several curves with different radii. In order to have a smooth approach to the curve, a transition curve is added. Even if it is known not to be the best shape approaching a curve, the linear transition curve (instead of more complex parabolic transition curves) was chosen because the speeds at which this will be covered are considered to be low and the focus of this work is on the dynamics on curves with fixed radius and cant. The weakness of this kind of transition curves can be seen in figure 6.21, where the displacement of the leading wheel set attached to the leading bogie frame is plotted against the inverse of the radius and the cant of the track. The wheel sets move outward and the flange contacts the rail suddenly. This is not kind of behavior is not desirable, thus linear transition curves are usually not employed in reality. Figure 6.22 shows that the position of flange contact is not reached if



(a) Displacement vs. Radius (b) Displacement vs. Cant

Figure 6.21: The lateral displacement of the leading wheel set is plotted against the inverse of the radius and the cant. The final radius is of 1.5km and the final cant is of 110mm. The sudden flange contact can be spotted also for such a low speed (20m/s).



(a) Displacement vs. Radius (b) Displacement vs. Cant

Figure 6.22: The lateral displacement of the leading wheel set is plotted against the inverse of the radius and the cant. The final radius is of 3.0km and the final cant is of 180mm. In this case final radius and cant are such that the flange contact doesn't happen (speed 20m/s).

the radius of the curve and the cant are sufficiently big, because the centrifugal forces will be weaker and better compensated by the track inclination. Even if flange contact is an abrupt event, this work will show that it provides good stability in curve.

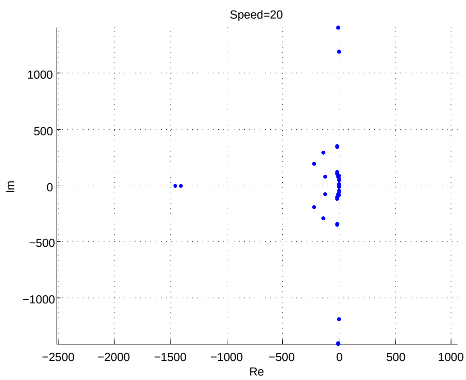
The relation between the radius and the cant of a curve, the speed of the vehicle and the dynamics in these condition is now investigated. The dynamics and the forces that are generated by these motions will be computed for a range of radii from 1.5km to 3.0km, with a refinement of 100m. For each of these curves, the cants between 110mm and 180mm will be employed with a refinement of 10mm. The simulation will be started with a transition curve that will be covered at the low speed of 20m/s. The speed will be then gradually increased of 1m/s every 40s, in order to give the time to the system to stabilize. The execution of the program will terminate when some of the displacements of one of the wheel sets will go out of range or the computation of the forces due to non-linear elements goes out of range.

The figures F.100-F.127 show the dynamics of all the components of the AGV model with respect to the speed. The only curves shown have a radius of 1.5km, 2.0km, 2.5km and 3.0km, with cant of 150mm. Similar figures have been obtained for the rest of the radii and cants tested. As expected, high cant and big radius help the train reaching higher speeds. Figures F.128-F.134 show that for curves with high cant (180mm) and very big radius (3.0km) the train doesn't travel on the external flange when running at low speeds. This fact gets more relevant when heavy freight wagons are running on such curves and they hit the internal rail causing very high wear. The model has shown good stability characteristics up to the speeds where the computation ends. The observation of the migration of the eigenvalues as the speed is increased confirms this fact. The figure 6.23 shows multiple plots of the position of the eigenvalues in the complex plane for the VHSPC model running in a curve with 1.5km of radius and 150mm of cant: all the eigenvalues have negative real part, thus the model has not stability issues for this type of curves.

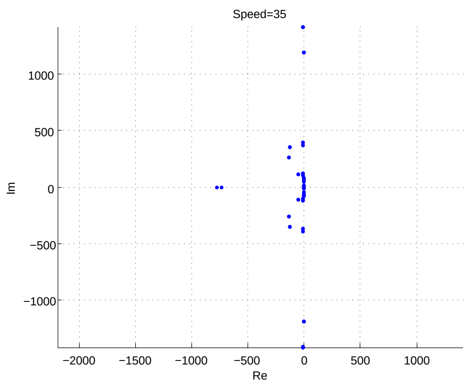
Using the results obtained, further analysis of the safety of the model, the wear factor and the ride comfort can be done. These analysis will be done with respect to the uncompensated acceleration(ANC). The Nadal's ratio is usually employed in order to assess the safety of a particular static condition. The figures F.135-F.142 show the evolution of the Nadal's ratio at increasing uncompensated acceleration. According to [UIC05, CEN05], the safety limit for twisted tracks is

$$\left(\frac{Y}{Q}\right)_{20Hz,2m,mean,99.95\%} \leq 1.2 \qquad (6.2)$$

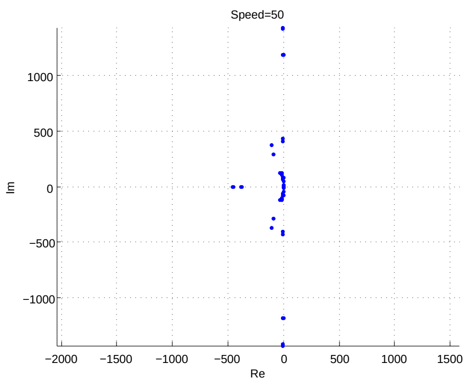
where the the 99.95% of the standard deviation around the mean over 2m of track with a disturbance with frequency 20Hz is considered. Our tests are performed without any disturbance on the track, thus the values of Nadal's ratio obtained are much lower than the limit. Furthermore the figures F.135-F.142 correctly show that the Nadal's ratio for the internal wheels (right wheels in this case) decreases with the increase of the uncompensated acceleration. The



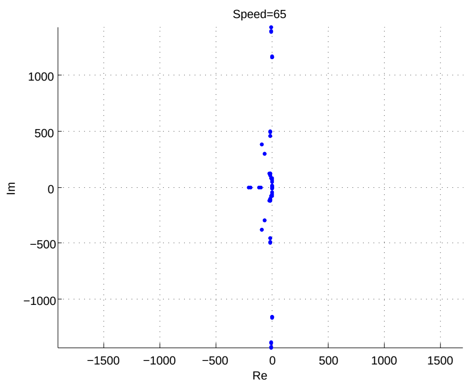
(a) Speed = 20m/s



(b) Speed = 35m/s



(c) Speed = 50m/s



(d) Speed = 65m/s

Figure 6.23: Migration of the eigenvalues for the VHSPC model running on a curve of 1.5km with 150mm of cant: the real part of the eigenvalues is always negative (an eigenvalue crossing the imaginary axis should get colored in red). The maximum speed reached for this simulation is 66m/s, confirming that the model has no stability issues.

opposite is true for the external wheels.

A good indication of the wear caused by a train running is given by the friction coefficient (μ) and the wear coefficient(k)[ABS07, Ch. 10]. These coefficients in turn depend on the materials used, the contact forces, the creepage, the weather conditions and the lubrication. In the simulation done through this work the friction coefficient chosen was $\mu = 0.15$ that corresponds to a condition of dry friction. On straight track the wear is mainly affecting the central tread of the wheel set and the rails, causing the increase of the flange height. The main forces causing this wear are the normal forces. On curves, the wear is typically located on the gauge corner of the outer rail and the top surface of the inner rail. The same happens for the wheel sets. The wear is connected, in this case, to the lateral forces acting on the contact points. This kind of wear can increase a lot the maintenance costs of both rails and wheel set, thus the lateral forces need to be reduced as much as possible. The ratio

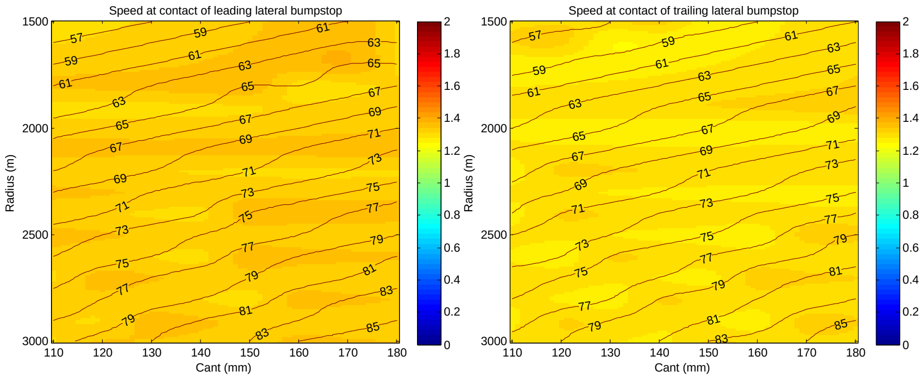
$$\frac{Y_{ext}}{Y_{int}} \tag{6.3}$$

gives an indication about the odd distribution of the lateral loads and figures F.143-F.146 show its change against the increase of the uncompensated accelerations. The size of the lateral loads on the external wheel get more and more important with respect to the internal wheel, meaning that the flange of the external wheel is pushing harder and harder against the gauge corner of the outer rail. Another illustration of the lateral forces is given by their sum for each wheel set. The lateral forces for the right and left contact points are directed always in opposite directions, thus there could be a situation in which these force compensate each other. Figures F.147-F.150 show the sum of these lateral forces against the increase of uncompensated acceleration. The value of unbalance between the inner and the outer lateral forces can reach values of the order of 10^4N.

A factor that increases the probability of flange climbing is the angle of attack of the wheel sets. Wheel sets composed by wheel profiles with positive conicity are known to over steer toward the center of the curve. This is confirmed by figures F.151-F.158, where also the angle of attack of the bogie frames and the car body are shown.

The ride comfort is mainly affected by the accelerations felt in the car body. In order to compute them, the equations of motion (2.87) end (2.88) were used plugging in the force values that are available along with the output. The lateral accelerations were computed in two locations: at the center of the car body and on the car body but over the position of one of the bogie frames, where the yaw angular acceleration adds to the lateral acceleration. The figures F.159-F.162 show the change in amplitude of the accelerations with respect to the increase of uncompensated acceleration.

The observation of all the results obtained highlighted that some perturbation of the dynamics was happening around 1.5ANC. For example there is a sharp increase in the accelerations felt on the car body (fig. F.159-F.162), but also the attack angles change drastically (fig. F.151-F.158). Also the contact forces get slightly affected by this phenomenon. The reason was sought in the suspension design. In particular the secondary suspensions are provided of a safety lateral bumpstop that avoid large oscillations. When the clearance of this bumpstop



(a) Leading lateral bumpstop (b) Trailing lateral bumpstop

Figure 6.24: The speeds at which the clearances of the lateral bumpstops are covered are shown as level sets against the radius and the cant of the track. The uncompensated accelerations corresponding to these speeds are shown by the use of colors: the bumpstop starts being active when the uncompensated acceleration is around 1.4ANC. Interpolation between the radius-cants grid was used here for estimating the values of the speeds.

is covered, the element starts behaving as a very stiff spring. In order to investigate the relation between the bumpstop and the perturbation in the results, the clearance of the lateral bumpstop in the secondary suspension are plotted against the uncompensated acceleration in figures F.163-F.166. The clearance is positive when there is still empty space between the two bodies. In general, the uncompensated acceleration necessary for the two bodies to hit each other is different between the leading and the trailing bogie frame. This is due to the fact that the car body will have an over steering attack angle, thus the trailing bumpstop will be hit first. The same values have been computed for all the combination of radius and cants and the speed at which the two parts of the bumpstop hit each other, has been detected. Figure 6.24 groups all these results together: the bumpstops are used approximatively when the uncompensated acceleration is bigger than 1.4ANC. The relation between radius, cant and uncompensated acceleration can be used for finding the maximum speed in order to avoid the perturbations to the dynamics due to the use of the bumpstops.

6.3 Numerics and Performances

Computational time and precision of the results are two driving factors for the evaluation of methods and implementations. The models analyzed along this work are composed by 66 first order differential equations. Non-linearities of the system can be due to suspension elements and to the contact forces. Simulations can last from few seconds, for simple cases where the transient in the solution is short and all the modes decay in the steady state, to hours, when the system is hunting or is passing through a long transient.
This section will cover some analysis done on DYTSI, in order to find the critical parts of the implementation, to study the different performance of the methods on different problems and to highlight the problems caused by the presence of non-linear elements in the system.

6.3.1 Profiling

After the correct execution of DYTSI has been verified and the results have proven its validity, the program can be profiled, in order to find weak parts of the code, where improvements in the implementation can help in saving computational time. When talking about computational time it's important to distinguish between algorithmic efficiency and numerical efficiency. The algorithmic efficiency is measured in function evaluations, Jacobian evaluations and number of iterations. The numerical efficiency depends on both the choice of the solver and the implementation, and is measured in terms of CPU time. Both the solver and the right hand function implementation can bear down on the final numerical efficiency. The right hand function for the train dynamics model is quite complex in some of its parts and can represent a bottleneck for the performances. The use of profiling software is then useful for finding where to make the needed changes.

The first profiling has been performed using the GNU profiler[2]. Figures F.167 and F.168 show the call diagram of DYTSI when a complete train model running through a transient curve and a constant curve for a total of 40s is considered and a transient analysis is performed using one of the ESDIRK solvers. The diagram shows only the part of the program where the integration is performed, that is where the most of the computational time is spent (around 98%). Each block of the diagram represents a function call and contains three additional information: the percentage of computational time due to the function and all its children, the computational time due only to the function, the number of times the function

[2] The GNU profiler is part of the GNU Compiler Collection. `http://gcc.gnu.org/`

has been called. The function that more influence the performances of the program are the ones that "personally" use a lot of computational time. Due to space constraints, the graph in figures F.167-F.168 was realized by filtering the functions with overall cost of less than 1.5%. The remaining part of the graph highlights that a big part of the computational work is done during the LU decomposition (23%), and the solution of the system (40%): thus around 65% of the computational load is used by the ESDIRK solver. The rest of the time is spent mostly during the computation of the right hand side function (4%) and the Jacobian(25%). A further observation of figure F.168 shows that all the children function calls for the right hand side and Jacobian computation have used little computational time, except the `WheelSetComponent::find_contact_forces` (3%) and the `Connector::getForcesAndMoments` (20%). So a lot of time was used for the computation of the suspension geometries and the forces.

In order to get a deeper view of the program execution and work load distribution a second profiling software, named Callgrind[3], was used. This profiler provides also an overview of the calls of the external libraries used by the program as well as the machine calls. A view focused on the functions that compute the forces and the moments of the suspensions is shown in figure F.169: a big amount of time is spent for the allocation and the deallocation of vector structures. In the version tested memory was dynamically allocated and this is highly expensive. In most of the cases there was no need for this kind of allocation and a simple resetting of a statically allocated vector was sufficient. A new version has been implemented that uses static allocation and the performances improved drastically. This affected the possibility of using multi-threading as the the same suspension elements could be asked to compute the forces by two different threads. Anyway multi-threading was discharged along the development of the project because the overhead work due to the loading and unloading of multiple threads by the scheduler was slowing down the computation instead of speeding it up.

6.3.2 Performances

The quality of a program for scientific computing is driven by the reliability of the results obtained and the computational effort required. The reliability of the results is equally influenced by the quality of the mathematical model, the accuracy and the stability of the ODE solver. The computational effort is determined by the model complexity, the quality of the ODE solver and its step size controller. The mathematical model is rooted in the physics laws

[3]Callgrind is part of the framework Valgrind for dynamic analysis. `http://valgrind.org/`

that govern the mechanical motion and groups a number of theories that are considered to be a good trade-off between complexity and accuracy. Thus, the performances of the program are mainly driven by the good choice of the ODE solver and its characteristics.
DYTSI employs three kinds of solvers: the Bulirsch-Stoer method, the backward differentiation formula with adaptive order selection and the ESDIRK method, in the Nielsen-Thomsen and Jørgensen-Kristensen-Thomsen versions. These methods have already been presented and tested on simple problems in chapter 3. Here the methods will be compared on the train dynamics problem.

The performances of the methods will be evaluated using two factors: the accuracy and the computational effort.
The accuracy is difficult to be checked for complex models, in particular if there is no possibility of having some real test data to compare with. The only way of checking accuracy is to use gross results like the values of critical speeds and the reliability of the dynamics. All the results shown up to now were obtained using the Nielsen-Thomsen ESDIRK solver[NT93] and are considered to be valid thanks to comparison with preceding works. Thus, this method is used as a meter of comparison with the other methods in terms of accuracy.
The computational effort can be evaluated using the computational time spent, the number of computations of the right hand side function and the Jacobian. Hereafter the CPU time is the real computational time, that considers also the possibility of being delayed by the operative system and other programs. Another useful tool for the evaluation of the computational effort is given by the evolution of the step size: the two class of methods uses different step size controllers and the results will be drastically different in some cases.

The model used for the following tests is the Cooperrider complete model. This model has already been analyzed in section 6.1, with results comparable to preceding works. Furthermore, the model suffers of a lot of the dynamical problems that can be found in railway vehicles dynamics, so it's perfect for a comparison in different situations. Several test cases have been designed in order to assess the behavior of the solvers in different situations.

1. Transient analysis on **straight track (not hunting)** at 40m/s.
   The simulation is started with a displacement of the leading wheel set of the leading bogie frame of $1 \cdot 10^{-4}$m. The solution is computed for 10s.

2. Transient analysis on **straight track (hunting)** at 60m/s.
   The simulation is started on the hunting limit cycle computed in section 6.1.2. The solution is computed for 10s.

3. Transient analysis on **transition curve (not hunting)** at 20m/s.
   The simulation is started from the track centerline position and a linear

transition curve is covered in 20s. The final radius is 600m and the final cant is 110mm.

4. Transient analysis on **curve (not hunting)** at 20m/s.
   The simulation is started from the displaced position where the transient curve has left the train on a curve with radius 600m and cant 110m. The solution is computed for 10s.

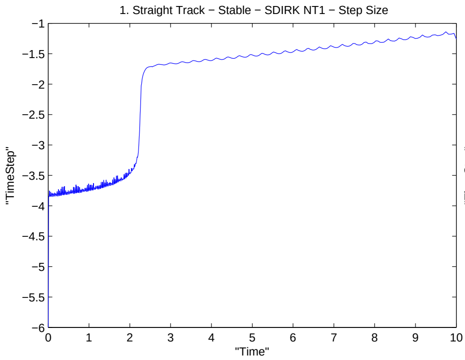
5. Transient analysis on **curve (hunting)** at 35m/s.
   The simulation is started on the hunting limit cycle computed in section 6.1.3 on a curve with radius 1.6km and cant 110mm. The solution is computed for 10s.

The four solvers of DYTSI have been tested on each of the test cases. Different step size controllers are used for the three classes of methods: the Bulirsch-Stoer method and the BDF method uses the asymptotic step size controller presented in section 3.1.1 whereas the ESDIRK methods use the PI controller discussed in section 3.3.1. The asymptotic step size controller uses the maximum ratio between the desired error and the estimated error. The desired error is obtained by a combination of the absolute tolerance and relative tolerance, that are set to $\varepsilon_{abs} = 10^{-8}$ and $\varepsilon_{rel} = 10^{-6}$. The PI controller uses, instead, the magnitude of the estimated error, that is compared only to the absolute tolerance $\varepsilon_{abs} = 10^{-8}$. Since the first controller bounds the error of the single degree of freedom with the biggest error, it turns out to be more conservative than the PI controller, where the magnitude of the error is used. The solvers are started with an initial step size of 10^{-6}, that is considered small even for problems with very fast transients.

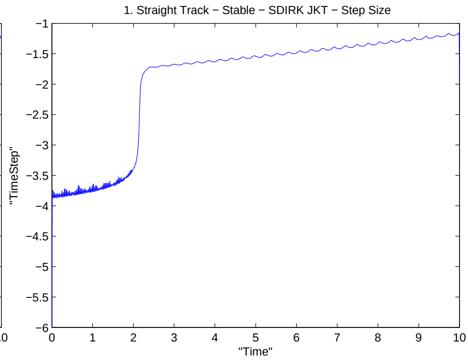
The performances of the methods are listed in table 6.1.
The first important observation is that the Bulirsch-Stoer method fails in two of the test cases: the not hunting straight track and the transition curve. The first of these tests is started with exact zero value for all the degrees of freedom except for the lateral displacement of the leading wheel set. The second test is started with exact zero value for all the degrees of freedom. Observation of the results shows that the step size is reduced dramatically and the solver is not able to proceed any further. This happens only when an exact zero starting condition is used, if the starting values are set to some small value (10^{-12}) the solver proceed and a solution comparable to the other methods is found. The step size controller is excluded from any responsibility for this behavior as it is used by the BDF method as well and it doesn't suffer the same problem. The suspects are on the error estimator that overestimates the error for some degree of freedom, as the step size controller only consider the maximum ratio of the estimated error and the desired error (see sec.3.1.1).

Test	Solver	CPU Time(s)	# Fun. ev.	# Jac. ev.	# Accept.	# Reject.	# Newt. Div.	Max Err.
1	ESDIRK NT1	201.71	146272	16168	11093	5074	0	1.20935e-07
	ESDIRK JKT	196.05	145396	16073	11104	4968	0	1.13138e-07
	BS IMP	x	x	x	x	x	x	x
	BDF	50.06	223984	1230	51407	10963	-	-
2	ESDIRK NT1	4653.63	4081015	339914	304172	35741	0	3.07523e-07
	ESDIRK JKT	4457.79	4122554	343293	310511	32781	0	2.96363e-07
	BS IMP	365.68	1534363	10999	7433	3565	-	-
	BDF	310.61	1374955	7758	323741	70900	-	-
3	ESDIRK NT1	1238.79	906675	100516	99916	599	0	1.27957e-08
	ESDIRK JKT	1163.77	853158	94500	93117	1381	0	1.60843e-08
	BS IMP	x	x	x	x	x	x	x
	BDF	106.48	474698	3100	157953	50	-	-
4	ESDIRK NT1	5.70	4664	446	424	22	0	1.83246e-08
	ESDIRK JKT	5.98	4884	469	439	29	0	1.86324e-08
	BS IMP	44.69	184365	1319	1298	21	-	-
	BDF	2.74	7463	44	2075	124	-	-
5	ESDIRK NT1	436.48	404210	33687	26146	7540	0	2.91751e-08
	ESDIRK JKT	456.02	419066	34925	27170	7754	0	2.88307e-08
	BS IMP	80.23	332452	2390	2175	214	-	-
	BDF	46.79	201272	1171	48144	11457	-	-

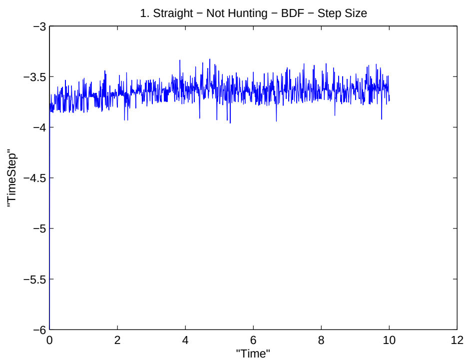
Table 6.1: Performances of the solvers on five test cases. The methods used are: the Nielsen-Thomsen ESDIRK method[NT93] (SDIRK NT1), the Jørgensen-Kristensen-Thomsen ESDIRK method[KTJr08] (SDIRK JKT), the Bulirsch-Stoer method[BS66] (BS IMP), the Backward Differentiation Formula[BH75] (BDF). The performance measures considered are: the CPU time (numerical efficiency), the number of function evaluation and the number of Jacobian evaluation (algorithmic efficiency), the number of accepted steps and the number of rejected steps (step size controller efficiency), the number of Newton divergences, the maximum estimated error. The symbol “x” indicates that the method failed, the symbol “-” indicates that the particular value is not provided by the method.



(a) SDIRK NT1



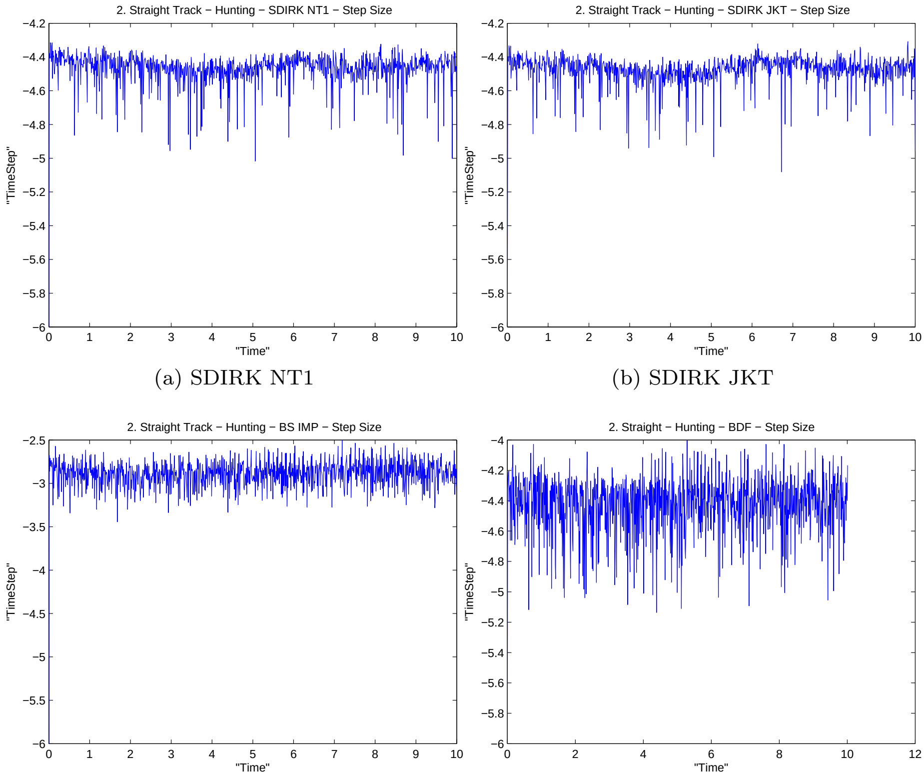
(b) SDIRK JKT



(c) BDF method

Figure 6.25: Step size (in logarithmic scale) of the solvers for the test case number 1: transient analysis on straight track with speed 40m/s (not hunting). The plots of the solution are shown in figures F.170- F.172.

The first test case on straight track is solved by the two ESDIRK methods and the BDF method. The behavior of the step size for the two methods is quite different: figure 6.25 shows that the two ESDIRK methods are able to take longer steps after the main modes of the transient die out. Nevertheless, from table 6.1 the CPU time suggests that the BDF iteration is quite faster than the ESDIRK iteration and this is confirmed by the significantly higher number of Jacobian evaluation for the ESDIRK method. Figures F.170- F.172 show the dynamics of the trailing wheel set attached to the leading bogie frame obtained by the three different solvers. This component has been chosen for the comparison between the methods because it contains both slow and fast modes due to the presence of the suspension with relatively low stiffness and the wheel-rail contact with very high stiffness. Furthermore, since the model

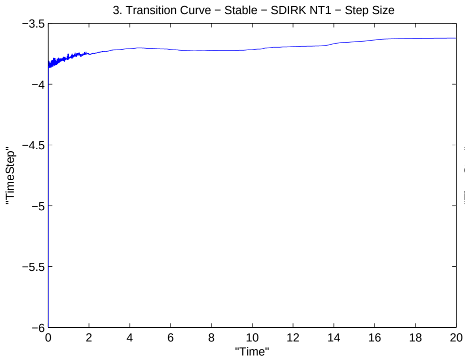


(a) SDIRK NT1 (b) SDIRK JKT

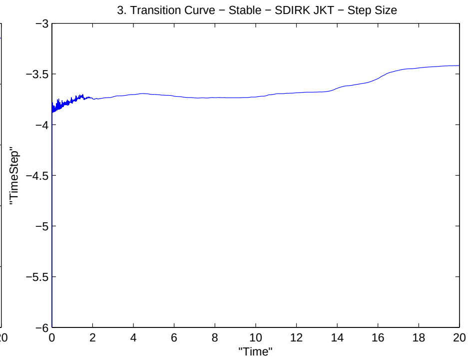
(c) Bulirsch-Stoer (d) BDF method

Figure 6.26: Step size (in logarithmic scale) of the solvers for the test case number 2: transient analysis on straight track with speed 60m/s (hunting). The plots of the solution are shown in figures F.174- F.177.

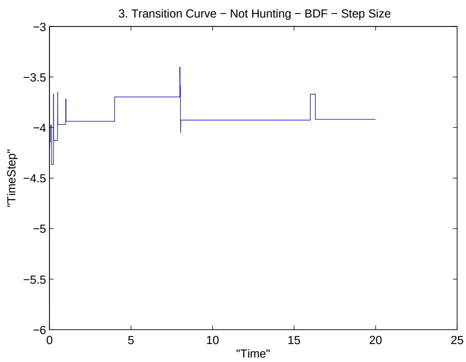
is not longitudinally symmetric due to the speed, the trailing components were found to be more sensitive to oscillatory motion and the hunting phenomenon (see sec.6.1.2). The observation of the obtained dynamics shows that the slow modes die out in the same way for the three methods. Instead, the fast vertical mode, due to the wheel-rail interaction, behave differently: for both the ESDIRK methods the mode dies out quite rapidly as it is for the other modes, for the BDF method the vertical motion is present for longer time, even if this do not affect the other degrees of freedom. The numerical damping inserted by the ESDIRK methods has already been discussed on the simple harmonic oscillator. In order to get results comparable to the ones obtained by the BDF method, the tolerance has been decreased to $\varepsilon_{abs} = 10^{-12}$. Figure F.173 shows that the vertical dynamics are now found by the ESDIRK method as well. The use of



(a) SDIRK NT1



(b) SDIRK JKT



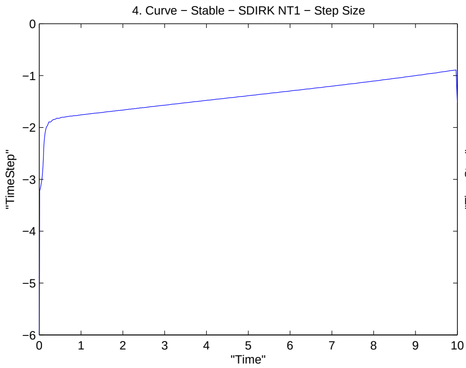
(c) BDF method

Figure 6.27: Step size (in logarithmic scale) of the solvers for the test case number 3: transient analysis on transition curve with speed 20m/s (not hunting). The plots of the solution are shown in figures F.178- F.180.

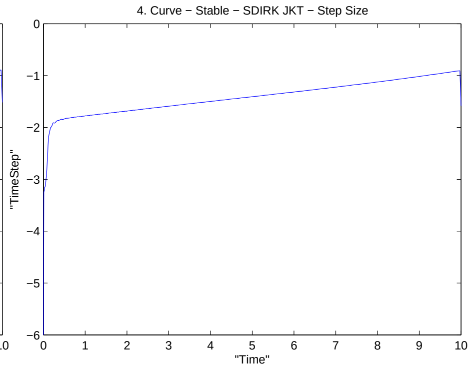
lower tolerance causes the method to take bigger steps and add damping to the high frequency modes.

The second test case regards the hunting motion on straight track. The transient is computed correctly by all the methods as the results show in figures F.174-F.177. From the performance point of view the Bulirsch-Stoer method and the BDF method outperform the others by an order of ten in CPU time. The first one uses bigger step sizes, the second one gain a lot by the low number of updates of the Jacobian matrix. This is confirmed by the figures in 6.26.

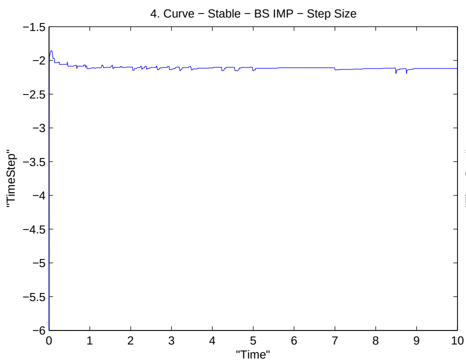
The third test is about the transition curve covered before entering a curve with fixed radius. This is correctly carried out by the ESDIRK methods and the



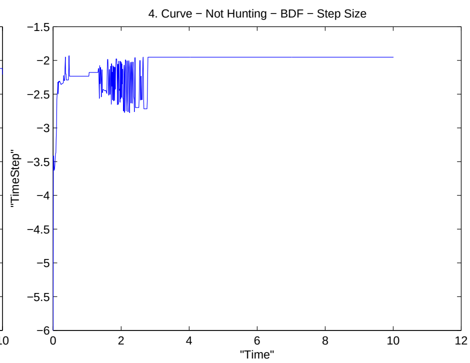
(a) SDIRK NT1



(b) SDIRK JKT



(c) Bulirsch-Stoer

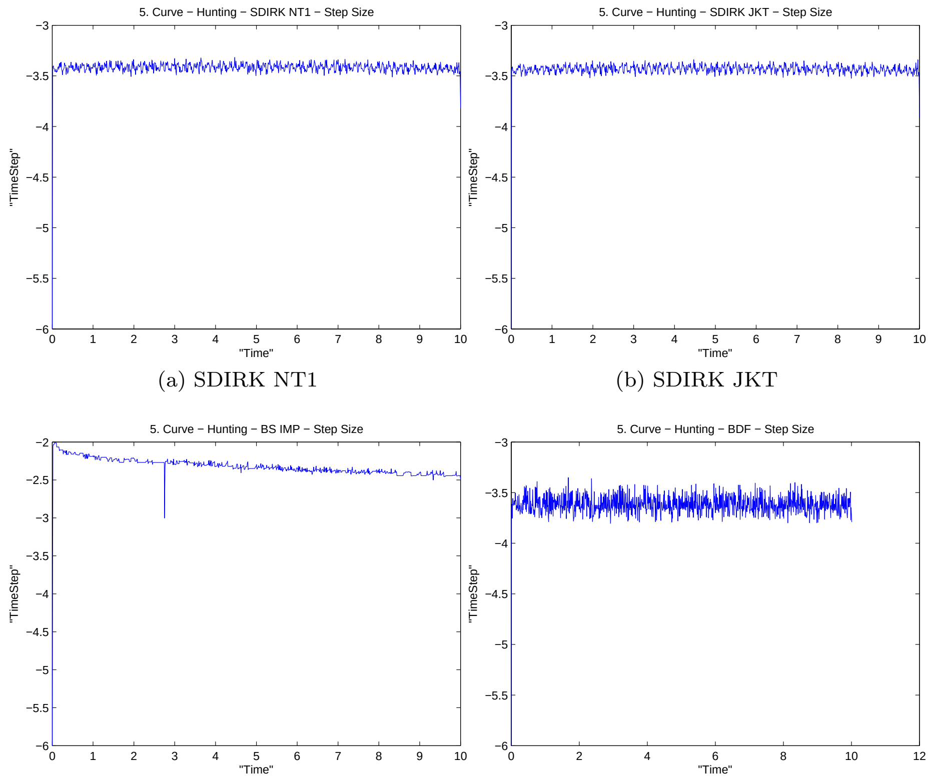


(d) BDF method

Figure 6.28: Step size (in logarithmic scale) of the solvers for the test case number 4: transient analysis on curve with speed 20m/s (not hunting). The plots of the solution are shown in figures F.181- F.184.

BDF method, with a significantly better performance of the latter one. The figures F.178- F.180 show the resulting dynamics and the figure 6.27 shows the step size behavior of the three methods.

The dynamics on curve with constant radius are analyzed in the fourth test case. The curve has a radius of 600m and a cant of 110mm, the train is running at the low speed of 20m/s, so there is no flange contact as discussed in section 6.1.3. Figures F.181- F.184 show the resulting dynamics computed by the four methods. The Bulirsch-Stoer method is able to catch the high frequency vertical oscillations due to the wheel-rail contact, whereas the two ESDIRK methods and the BDF method don't. Anyway, this does not affect the other degrees of freedom that are properly computed. The figure 6.28 shows the step sizes of



(a) SDIRK NT1

(b) SDIRK JKT

(c) Bulirsch-Stoer

(d) BDF method

Figure 6.29: Step size (in logarithmic scale) of the solvers for the test case number 5: transient analysis on curve with speed 35m/s (hunting). The plots of the solution are shown in figures F.185- F.188.

the four methods. The CPU time for the ESDIRK methods is much lower than for the other methods that are probably delayed by the computation of the fast modes.

The fifth test case regards the hunting motion in curve. The curve selected has a radius of 1.6km and a cant of 110mm and the train is running at the speed of 35m/s. The hunting motion is shown in figures F.185-F.188. Significantly differences appear between the results obtained by the ESDIRK method, the Bulirsch-Stoer and the BDF methods. The vertical component of the displacement of the wheel set is not stable when computed by the BS method. Instead this effect does not appear in the results obtained by the ESDIRK methods and the BDF method. If the tolerance is lowered down, the non stable behavior appears for the other three methods as well. This can be due to finite precision error accumulation or to some discontinuity in the track as it was found in other works. This behavior has not been investigated any further for now.

The dynamics obtained for the test cases show that results strictly depend on the method and the tolerances used. The best results are the one that can be confirmed by several methods. The most conservative tolerances could be used for all the methods and the result should converge, however this would require an unacceptable time. It's always a convenient strategy to find the tolerances such that accuracy (and in some cases stability) is preserved while the computational time is acceptable. The results found on test case 1 (straight track with no hunting) by ESDIRK NT1 with absolute tolerance 10^{-12} are used as meter of comparison for the different methods: the degree of damping shown in the vertical component in figure F.173 is considered and the tolerances, for finding the same behavior, are found for all the methods. Table 6.2 shows these tolerances with some of the performances characteristics (The BS method was included in these tests by working around its problem in case of zero initial conditions, setting the starting values to 10^{-12}). The BDF and BS methods outperform the ESDIRK method in this case.

In general a reduction of the tolerance, limited by the finite precision of the system, provides more accurate results at the expense of computational time. However, sometimes this level of accuracy is not necessary as the main dynamics observed are not affected by a more accurate solution. When this is the case, higher tolerances can be used. The four methods used here have different performances with tight or relaxed tolerances. The BDF and the BS method have shown worse stability when lower accuracy was required, but they are relatively fast in finding accurate solutions. The ESDIRK methods show much worse performances in finding accurate solutions, but they are safer as they had no stability issue when the tolerances were relaxed.

Gross results, like the critical speeds, are not strongly affected by the accuracy as the same results were found using different tolerances. Furthermore, the same gross results were reproduced using the four different methods with the appropriate tolerances.

6.3.3 Non-smooth dynamical systems

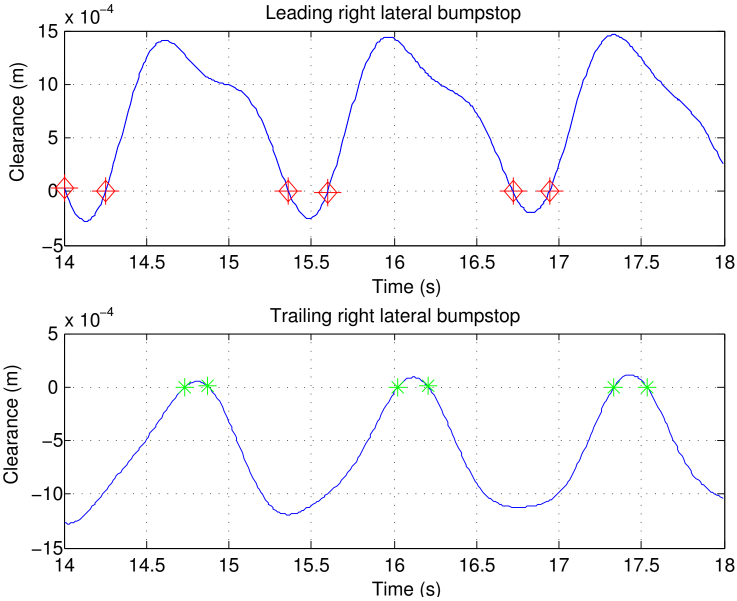
Several forms of non-linearities can be present in a dynamical system. Sometimes non-linearities are given by terms of high order, other times by functions that are not continuous or non smooth. The second case is quite usual in mechanical systems where only discrete data generated by physical observations can

Solver	ε_{abs}	ε_{rel}	CPU Time(s)	# Fun. ev.	# Jac. ev.	# Accept.	# Reject.
BDF	10^{-12}	10^{-6}	174.28	829155	4509	192348	32543
	10^{-11}	10^{-6}	154.65	746677	3928	174414	23952
	10^{-10}	10^{-5}	103.25	488636	2654	111667	20413
BS IMP	10^{-12}	10^{-5}	75.44	322231	2315	2241	73
	10^{-11}	10^{-5}	68.85	294147	2106	2100	5
	10^{-10}	10^{-5}	65.68	279853	2003	1989	13
ESDIRK NT1	10^{-10}		2796.80	2045084	218778	177573	41204
ESDIRK JKT	10^{-10}		2910.18	2117465	227469	185696	41772

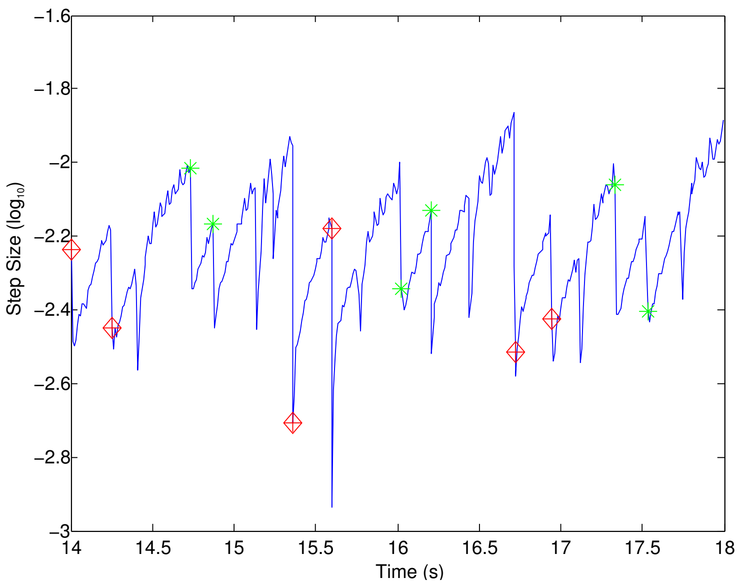
Table 6.2: Minimum tolerances for obtaining the vertical motion of the wheel sets with the same amount of numerical damping. The dynamics for such tolerances are more or less equal to the ones shown in figure F.173.

be used. In train dynamics these non-linearities can arise from the wheel-rail contact, that is only computed exactly on a discretized grid, or in the suspension components, that are designed for having different behaviors as a response to different stresses. Such non-linearities decrease the performance of the numerical solver dramatically.
DYTSI provides only the possibility of using different interpolations for these elements. However, new approaches to the problem are available for non-smooth problems, where the solver is interrupted before crossing a non-smooth point, called event, and the integration is done just up to the exact time the event happens. This permits the solver to take the necessary step up to the non-smooth point without surpassing it and then avoiding the usage of the step size controller that would reduce the step too much. This approach has been already applied on train dynamics by Hoffmann in [Hof06].
Here, the reasons for such an approach are justified using a realistic wagon model where lateral bumpstops are employed in order to avoid big lateral displacements. The model employs other non-linear components, but they are already quite smooth, in comparison to the bumpstops, and they are approximated by interpolation functions. Instead, the force-displacement relation for the bumpstops is defined by a piece-wise linear function where the stiffness undergoes a very big jump when the two parts hit each other. The model is run in a right hand curve, so that the centrifugal acceleration pushes the whole system toward the outer rail and the right lateral bumpstop start working. The exact speed at which the two parts of the bumpstops hit each others repeatedly is used (at higher speed there would only be contact as the centrifugal force would be too high). The bumpstops attached to the leading and the trailing bogie frames don't work simultaneously as the car body is running skewed in curve (it's over steering), so the trailing bumpstop works more in the highly stiff part of its force-displacement relation respect to the leading one. This is shown by the clearance graphs in figure 6.30a. In the figure each event is marked and the same mark is shown in the step size graph in figure 6.30b. The step size controller drastically shrinks the step all the time the two parts of the bumpstop hit each other. Thus, special integration rules for such situations are highly welcome in order to speed up the solver. Not only bumpstops could benefit of the usage of an event-driven system, but all the non-linear components in the system.

DYTSI doesn't implement the event driven framework. It is left as future work.



(a) Clearances



(b) Step Size

Figure 6.30: Relation between the clearance of the bumpstops in the model and the step size evolution. The step size is drastically reduced all the times the two parts of a bumpstop hit each other.

Chapter 7

Conclusions

A modeling framework for generic four-axles two-bogies wagons has been developed along this work, using basic physics laws for the suspension modeling and well known theories for the computation of contact forces. Straight and canted curved tracks have been considered.
The framework has been implemented in the program DYTSI. It has been verified by plausibility tests aiming at the comparison between numerical results and physical results. It has been validated observing gross results and comparing them with previous works and with real data.

The analysis performed on the Cooperrider model (see sec.6.1) have confirmed previous results found by Petersen-Hoffmann in [HP02]. The analysis was extended to a complete wagon model: the value of the linear critical speed was found at $110\,\frac{\mathrm{m}}{\mathrm{s}}$, $7\,\frac{\mathrm{m}}{\mathrm{s}}$ less than for the half model used in preceding works. The non linear critical speed for the complete model was found at $51\,\frac{\mathrm{m}}{\mathrm{s}}$ whereas it was $56.5\,\frac{\mathrm{m}}{\mathrm{s}}$ for the half-model. Also the characteristic vibration frequency is lowered down from 5.8Hz to 4.8Hz. An important observation is that the complete model on straight track is not symmetric along the lateral axis (even if the design is) because of the direction of the speed. This causes the leading and the trailing part of the wagon to have different dynamical behaviors: the trailing bogie frame seems to be more sensitive to the hunting phenomenon. This was shown by being able to enter the hunting limit cycle with only the trailing bogie frame. This was also confirmed by the presence of two couples

of conjugate complex eigenvalues that cross the imaginary axis at two different speeds ($110\frac{\mathrm{m}}{\mathrm{s}}$ and $111\frac{\mathrm{m}}{\mathrm{s}}$).
Previous works have never considered the effects of precession forces due to the fast spinning of the wheel sets. On straight track precession forces don't affect the dynamics, instead in curved tracks, the results can be qualitatively different. On curved tracks the stabilizing effect due to flange contact was found as well as a decrease of the critical speeds on curves with small cant deficiency, where flange contact does not happen until higher speeds are reached. The appearance of super-critical Hopf bifurcations for certain curves is confirmed also for the complete wagon model.

Part of this work has been carried out in collaboration with ALSOMT Transport, that provided a realistic very-high-speed model. The analysis was focused on the dynamics in curved tracks with high cant deficiency and with radii that ranged between 1.5km to 3.0km (see sec.6.2). The tests were performed on smooth tracks and they provided results that agree with such a condition. Right now, the main concerns in industry are about the maintenance costs due to the wear of rails and wheels, that increase dramatically with the speed. The analysis of the lateral forces on the wheel-rail contact point gives a good indication of the increase of wear due to the increasing speed. Finally a relation between the instability found around 1.4ANC and the end of the clearance in the lateral bumpstops in the secondary suspension was found.

The project focused also on the analysis of the performance of four different methods on the highly stiff train dynamics problem. It was found that the two methods from the ESDIRK family have very good stability properties and can be used safely for getting reliable results even when tolerances are relaxed. However, they didn't show good performances when small tolerances were used. Better performances were achieved by both the BDF method and the Bulirsch-Stoer method, at the expenses of more instability for relaxed tolerances.

The work leaves multiple directions for further developments and investigations. The first thing to bear in mind is that the real world is not smooth, not to mention mechanical components. The approximation of the track with a smooth set of rails is not sufficient most of the times. Even if phenomena, like the hunting motion, are still under investigation for simple smooth tracks, there are evidences of the contribution by perturbed tracks in entering the hunting limit cycle. There are several kinds of irregularities that can affect the dynamics of a railway vehicles and they need to be included in the modeling framework in order to get good approximations of the reality. Analysis of the dynamics on perturbed tracks is also important for a better estimation of the contact forces, that help in the estimation of wheel-rail wear and in the study of derailment.
Tracks are not the only non-smooth components in a railway vehicle model. Most of the elements of the suspensions present non-linear behaviors and some

of them have even non-smooth behaviors. Approximations of these elements using assumptions of linearity or interpolation functions, can affect the results. On the other hand, the presence of non-smooth functions in the system causes worse performances of the solvers that are required to take smaller steps in the proximity of a non smooth point (see sec.6.3.3). The implementation of an event driven framework can speed up the computations as shown in preceding works[Hof06].
From the point of view of the results, some of them need confirmation and further analysis. For example the effect of precession forces should be confirmed. Furthermore, in section 6.1.1 an example has shown that the necessary perturbation in order to enter the hunting limit cycle needs to be composed by several perturbations of the degrees of freedom. The investigation of the kind of perturbation necessary for the train to start hunting would shed more light on the hunting phenomenon. The result about non symmetric dynamics on a symmetric wagon along the lateral axis needs to be confirmed by other works as well.

Chapter 8

Amendments

This section is the fruit of the some observations that were pointed out during the defense of the thesis and during the preparation of an article with title "*On the Numerical and Computational Aspects of Non-Smoothnesses that occur in Railway Vehicle Dynamics*".

8.1 Explicit vs. Implicit methods

In section 6.3.2 the performances of several methods have been compared in order to shed some light on which solver would be more suitable for train dynamics problems. The choice of the methods were restricted to implicit solvers that are considered to be better than the explicit ones in solving stiff problems. In this section the profitability of choosing an implicit method will be challenged.

In order to obtain a fair comparison between explicit and implicit solvers, two methods with the same order of accuracy and that utilize the same solver need to be selected. The two methods are selected among the Runge-Kutta family, as they are relatively easy to implement (the SDIRK package can be extended adding very few modifications). The implicit solver selected is the ESDIRK34 NT1 already presented in section 3.3 and used along all this work.
The explicit method used is the Runge-Kutta-Fehlberg (ERKF34) with order

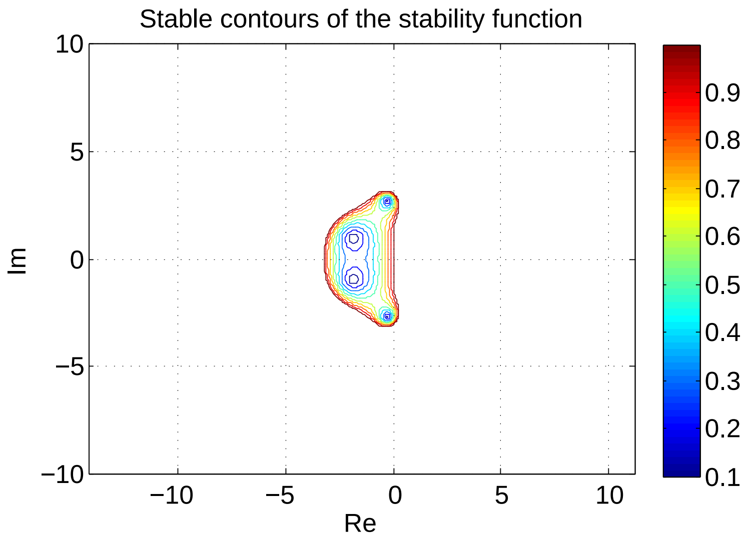


Figure 8.1: Region of absolute stability of the ERKF34 method.

3 for the advancing method and order 4 for the embedded method[HNrW91]. The method is defined by the Butcher tableau:

$$\begin{array}{c|ccccc}
0 & 0 & & & & \\
\frac{2}{7} & \frac{2}{7} & 0 & & & \\
\frac{7}{15} & \frac{77}{900} & \frac{343}{900} & 0 & & \\
\frac{35}{38} & \frac{805}{1444} & -\frac{77175}{54872} & \frac{97125}{54872} & 0 & \\
1 & \frac{79}{490} & 0 & \frac{2175}{3626} & \frac{2166}{9065} & 0 \\
\hline
 & \frac{229}{1470} & 0 & \frac{1125}{1813} & \frac{13718}{81585} & \frac{1}{18} \\
\hline
 & -\frac{888}{163170} & 0 & \frac{3375}{163170} & -\frac{11552}{163170} & \frac{9065}{163170}
\end{array}$$

Since this method is explicit, the region of absolute stability of ERKF34 is bounded and thus it is not A-stable. This is shown also in figure 8.1.
The step size controller adopted for both the explicit and the implicit methods is the PI controller presented in section 3.3.1.

The methods have been tested on the Cooperrider wagon model running on straight track with different speeds. In the first test case the model is running at 40m/s (Not Hunting). An initial perturbation on the leading wheel set attached to the leading bogie frame is applied in order to observe the different behaviors on the solution of the transient. A tolerance of 10^{-5} has been used. Both the

ε_a	Solver	CPU T.(s)	# Fun.	# Jac.	# Acc.	# Rej.
10^{-5}	ERKF34	15.11s	93361	0	23336	4
10^{-5}	ESDIRK34 NT1	1.73s	1302	128	114	14

Table 8.1: Performances of the ESIDRK34 NT1 method and the ERKF34 method on the test with the Cooperrider wagon running on straight track at V=40m/s (Not Hunting). The table shows the absolute tolerance used for the tests, the wall clock time, the number of function evaluations, the number of Jacobian evaluations, the number of accepted steps and the number of rejected steps.

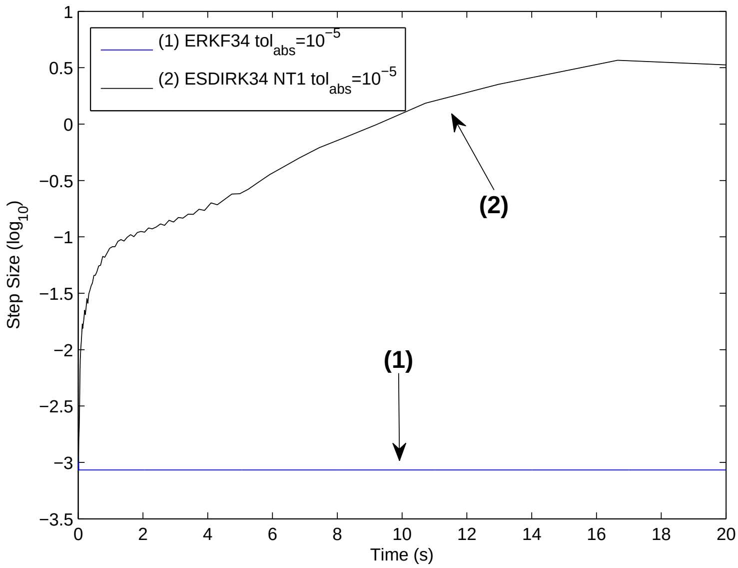


Figure 8.2: Step size evolution of the ESDIRK34 NT1 and the ERKF34 for the transient analysis of the Cooperrider model running on straight track at 40m/s (not hunting).

methods were able to find correctly the solution. The implicit method strongly outperformed the explicit one (see table 8.1). Figure 8.2 shows that the implicit solver NT1 is able to take longer steps as soon as the transient dies out, whereas the step size that the explicit solver is able to take is clearly bounded.

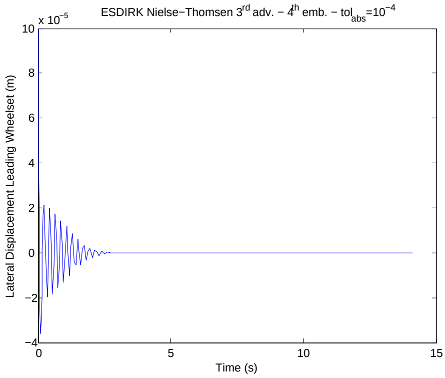
The same two solvers are then compared on the test case with the Cooperrider wagon running on straight track at V=120m/s (Hunting). The tolerances $\varepsilon_a = 10^{-4}$ and $\varepsilon_a = 10^{-5}$ have been used for the tests. The performances of the two methods are listed in table 8.2. As shown in figure 8.3a the implicit method

ε_a	Solver	CPU T.(s)	# Fun.	# Jac.	# Acc.	# Rej.
10^{-4}	ERKF34	15.34s	74505		12617	6009
	ESDIRK34 NT1	1.47s	1181	112	90	20
10^{-5}	ERKF34	21.25s	130389		22989	9613
	ESDIRK34 NT1	467.12s	428957	34911	25525	9385

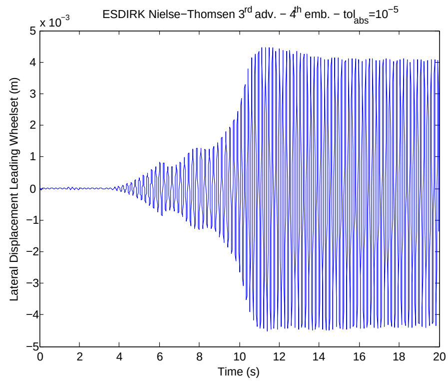
Table 8.2: Performances of the RKF34 and ESDIRK34 NT1 for solving a transient analysis of 20s of a Cooperrider model hunting. ESDIRK34 NT1 with tolerance 10^{-4} fails in detecting the hunting phenomenon.

fails in detecting the hunting motion when high tolerances are used. The NT1 method is able to detect the correct dynamics only if the tolerance is reduced enough, as it is shown in figure 8.3b. This doesn't happen to the explicit method that is able to find the correct solution with both the tolerances (see figure 8.3c). From the performance point of view, the explicit solver clearly outperforms the implicit one with a factor of 20. The reason can be sought in the step size history shown in figure 8.3d. Both the methods are forced to take steps with the same length, due to accuracy issues. However the computational cost per step for the implicit solver is much higher than for the explicit solver, because the Jacobian matrix needs to be approximated.

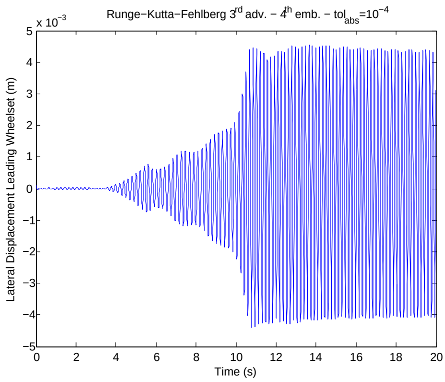
The reason why both the methods are forced to take the same step lengths can be found in the stiffness of the problem. Stiff problems are characterized by the presence of modes with very different speeds. In the train dynamics problem the fastest modes are the ones that involve the vertical displacement of the wheel sets. This movement is influenced by the very high stiffness of the material used for the wheels and the rails. As it was pointed out along this work, the wheel-rail interaction is very important for the hunting phenomenon and thus a correct approximation of the vertical motion of the wheel set is mandatory in order to obtain correct dynamics.
Implicit solvers are known for being able to take steps with the length bounded only by accuracy constraints. They do this at the expense of damping fast modes and this gives very inaccurate results when the dynamics depend on them, as it is the case in train dynamics. Explicit solvers don't introduce additional damping, enabling the observation of the correct dynamics with the desired accuracy. Furthermore, since the fast modes need to be considered, the implicit solvers and the explicit solvers will be forced to take the same step size, resulting in better performances by the explicit ones.
Concluding, the choice of the method and the tolerance needs to be done considering the dynamics that are investigated. Implicit methods are suggested only for problems with known dynamics where all the fast modes are contracting in the steady state (not hunting case). In all the other cases a preliminary investigation using high order explicit methods is recommended.



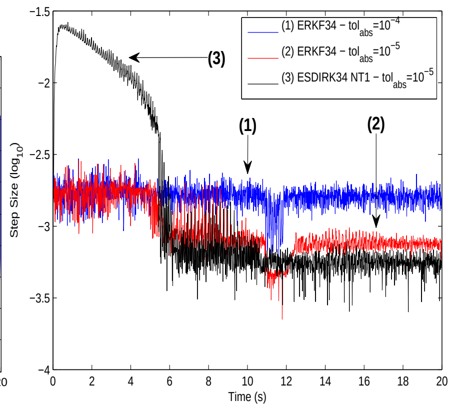
(a) ESDIRK34 NT1 ($\epsilon_a = 10^{-4}$)



(b) ESDIRK34 NT1 ($\epsilon_a = 10^{-5}$)



(c) ERKF34 ($\epsilon_a = 10^{-4}$)



(d) Step size histories

Figure 8.3: Computed results for lateral displacement of leading wheelset of Cooperrider's bogie model for different user-defined absolute tolerance levels. Using ESDIRK34 NT1 it is found that a) the transient behavior is fully damped for $\epsilon_a = 10^{-4}$ and b) periodic oscillations (hunting) is captured for $\epsilon_a = 10^{-5}$. With ERKF34 it is found that c) periodic oscillations (hunting) is captured already at $\epsilon_a = 10^{-4}$.

8.2 Friction coefficient

The friction coefficient is the ratio between the force of friction between two bodies and the force pressing them together. It was introduced in section 2.2.2 and used for the calculation of the creep forces. The friction coefficient used for computing the results so far was $\mu = 0.15$. This value is considered to be too low

μ	Linear	Non-Linear
0.03	114m/s	111m/s
0.06	111m/s	73.5m/s
0.09	110m/s	62.5m/s
0.12	110m/s	59.5m/s
0.15	110m/s	51m/s
0.18	109m/s	Derail 79m/s
0.21	109m/s	Derail 92m/s
0.24	109m/s	Derail 96.5m/s
0.27	109m/s	Derail 101.5m/s
0.30	109m/s	Derail 102.5m/s

Table 8.3: Linear and Non-linear critical speeds for different choices of the friction coefficient. For some coefficients the adiabatic decrease of speed leaded to the computational break of DYTSI due to a too big dispalcement of one of the wheel sets.

for safety reasons, thus, usually, the values between $\mu = [0.30, 0.50]$ are adopted. This section wants to show how the dynamics depend on this parameter. Table 8.3 shows that the linear critical speed doesn't vary considerably with respect to the friction coefficient. However, figure 8.4 shows that the change in the dynamics is relevant. The non-linear critical speed is found at different velocities depending on the friction coefficient. As the coefficient approaches zero, the subcritical Hopf bifurcation is transformed in a supercritical Hopf bifurcation with increased critical speed (see table 8.3). As the friction coefficient increases, the dynamics due to the contact forces are amplified and the non-linear critical speed decreases. For the Cooperrider wagon model this results also in the failure to find the non-linear critical speed for friction coefficients that are too high and cause the climb of the flange.
Even if not tested, similar effects are expected to appear for curved tracks.

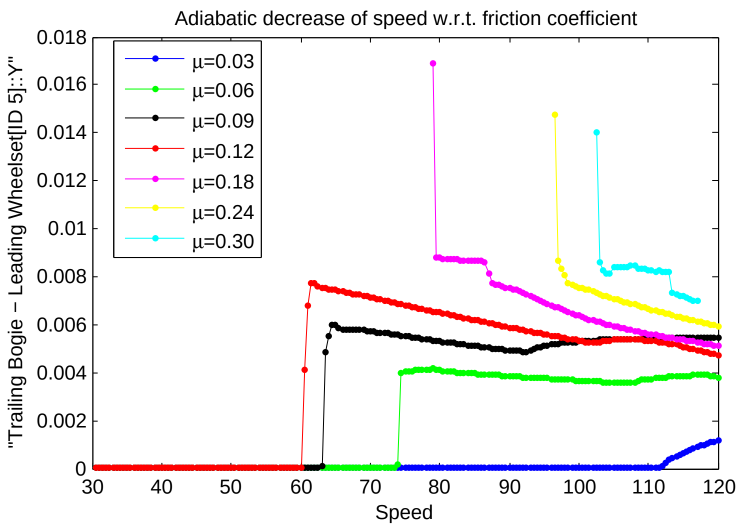


Figure 8.4: Adiabatic decrease of the speed starting from the hunting limit cycle for different friction coefficients.

Appendix A

Notation

The notation used for building the multi body model in chapter 2 is presented here. The symbols are divided per sections because some of them have been reused and could refer to different things. However, the symbols used for addressing the reference frames, the forces and the moments are valid for all the sections, because they are all reused in 2.4 to build the equations of motion.

Reference frames, forces and torques

I	Inertial reference frame	5
F	Track following reference frame (used as superscript)	6
C	Reference frame attached to the Car body	6
B	Reference frame attached to the Bogie frame	6
W	Reference frame attached to the Wheel set	6
R	Reference frame of a contact point on the right wheel	6

L	Reference frame of a contact point on the left wheel	6
X	Longitudinal coordinate of the inertial reference frame	5
Y	Lateral coordinate of the inertial reference frame	5
Z	Vertical coordinate of the inertial reference frame	5
x	Longitudinal coordinate of the body attached reference frames	6
y	Lateral coordinate of the body attached reference frames	6
z	Vertical coordinate of the body attached reference frames	6
ξ	Longitudinal coordinate of the contact patch reference frame	6
η	Lateral coordinate of the contact patch reference frame	6
ζ	Normal coordinate of the contact patch reference frame	6
ϕ	Roll angle (rotation along the longitudinal axis)	6
χ	Pitch angle (rotation along the lateral axis)	6
ψ	Yaw angle (rotation along the vertical axis)	6
$\vec{F}$	Force vector	5
$\vec{M}$	Torques vector	5

Reference frames and degrees of freedom (sec. 2.1)

m	Mass of a body	5
[J]	Tensor of inertia	5
$\vec{a}$	Acceleration vector	5
$\vec{\omega}$	Angular acceleration	5
$_t$	Subscript notation for track	6

$\mathbf{v}$	Speed of the track following reference frame	8
${}^{I}\mathbf{v}^{F}$	Speed vector of the track following reference frame from the Inertial r.f.	8
${}^{\bullet}\frac{d}{dt}$	Derivative with respect to the reference system $\bullet$	8
$\mathbf{p}$	Position vector	9
R	Radius of the curve	8
ϕ_t	Cant of the curve	8
C^*	Center of mass of a generic body	10
l	Longitudinal nominal position of the C.M. of a body with respect to the track following reference frame	10
b	Lateral nominal position of the C.M. of a body with respect to the track following reference frame	10
h	Vertical nominal position of the C.M. of a body with respect to the track following reference frame	10
g	Gravitational acceleration	13
$_c$	Used referring to centrifugal effects (subscript)	12
$_g$	Used referring to gravitational effects (subscript)	13

Wheel-rail interaction (sec. 2.2)

Q	Vertical force acting on the wheel sets	16
N	Normal force to the contact patch plane	16
q	Wheel rail penetration	17
Δq	Additional penetration	17
q_0	Static penetration	17
$N_{l/r}$	Normal forces on the left/right wheel	17
$_{N_{l/r}}$	Normal forces on the left/right wheel	17

$r_{l/r}$	Rolling radius on the left/right wheel	18
$a_{l/r}$	Distance of the contact point from the center of the wheel set	18
$\delta_{l/r}$	Contact angle for the left/right wheel	18
f_b	Bogie frame's wheel base (longitudinal base)	18
w_b	Width of the wheel set	18
Ω	Angular velocity of the wheel set	18
$\xi_\bullet$	Creep on the front/rear, left/right wheel	18
a	Major axis of the contact patch	19
b	Minor axis of the contact patch	19
$C_\bullet$	Kalker's coefficients	19
G	Shear modulus	19
μ	Friction coefficient	20

Suspensions modeling (sec. 2.3)

${}^{R_1}\vec{r}_l^{R_2}$	Attack point vector of link l on the component with r.f. R_2 written in the r.f. R_1	22
${}^{F}\vec{r}_0^{R}$	Position vector of the center of mass of the body with reference frame R with respect to the track following reference frame	22
${}^{F}\vec{b}_l$	Actual length vector of the link l with respect to the track following reference frame	22
${}^{F}\vec{b}_{l_0}$	Length vector of the link l at rest with respect to the track following reference frame	22
${}^{F}\vec{d}^{R}$	Displacement of the body with reference frame R with respect to the track following reference frame.	23
${}^{F}\dot{\vec{d}}^{R}$	Speed of the body with reference frame R with respect to the track following reference frame.	23
${}^{R1}\vec{\tau}_l$	Torque acting on the attack point of the link l on the body with reference frame $R1$ w.r.t the reference frame $R1$.	24
${}^{F}\vec{F}_k$	Vector force due to a stiff element written in the track following r.f.	26

Equations of motion (sec. 2.4)

Appendix B

Transformation matrices

The rotation matrices are used for passing from a reference frame to another and vice versa. A good characteristic of rotation matrices is that they are orthonormal, thus their inverse is equal to their transpose. Therefore, this section will present only one of the transformation matrices for two reference frames, because the second one can easily be found by transposition. The notation used will be ${}^{R1}\mathbf{T}^{R2}$, with the meaning that the application of this matrix will convert a vector written in the $R2$ reference frame to the $R1$ reference frame. Namely

$$ {}^{R1}\vec{x} = {}^{R1}\mathbf{T}^{R2}\,{}^{R2}\vec{x} $$

As it was introduced during the modeling of the system, a lot of reference frames have to be considered in order to build up the model. The rotation from the inertial reference frame to the track following reference frame is completely described in section 2.1.1.
The second relevant coordinate transformations are the ones from the bodies' reference frames to the track following reference frame. Linearization of sinusoidal functions for small angles gives

$$ {}^{F}\mathbf{T}^{W} \approx \begin{bmatrix} 1 & -\psi^{W} & 0 \\ \psi^{W} & 1 & -\phi^{W} \\ 0 & \phi^{W} & 1 \end{bmatrix} \tag{B.1} $$

$$ {}^{F}\mathbf{T}^{B} \approx \begin{bmatrix} 1 & -\psi^{B} & \chi^{B} \\ \psi^{B} & 1 & -\phi^{B} \\ -\chi^{B} & \phi^{B} & 1 \end{bmatrix} \tag{B.2} $$

$$ {}^{F}\mathbf{T}^{C} \approx \begin{bmatrix} 1 & -\psi^{C} & \chi^{C} \\ \psi^{C} & 1 & -\phi^{C} \\ -\chi^{C} & \phi^{C} & 1 \end{bmatrix} \tag{B.3} $$

where ϕ, χ and ψ are the roll, the pitch and the yaw respectively. The positivity of the angles is given by the right-hand grip rule and can be checked in fig. 2.4. In section 2.3.1, the first derivative of the rotation matrices was used in order to compute the relative speed of the attack points on two connected components.

$$ {}^{F}\dot{\mathbf{T}}^{W} \approx \begin{bmatrix} 0 & -\dot{\psi}^{W} & 0 \\ \dot{\psi}^{W} & 0 & -\dot{\phi}^{W} \\ 0 & \dot{\phi}^{W} & 0 \end{bmatrix} \tag{B.4} $$

$$ {}^{F}\dot{\mathbf{T}}^{B} \approx \begin{bmatrix} 0 & -\dot{\psi}^{B} & \dot{\chi}^{B} \\ \dot{\psi}^{B} & 0 & -\dot{\phi}^{B} \\ -\dot{\chi}^{B} & \dot{\phi}^{B} & 0 \end{bmatrix} \tag{B.5} $$

$$ {}^{F}\dot{\mathbf{T}}^{C} \approx \begin{bmatrix} 0 & -\dot{\psi}^{C} & \dot{\chi}^{C} \\ \dot{\psi}^{C} & 0 & -\dot{\phi}^{C} \\ -\dot{\chi}^{C} & \dot{\phi}^{C} & 0 \end{bmatrix} \tag{B.6} $$

The last two coordinate system that have to be addressed are ones attached to the contact points. These coordinate systems have the vertical plane in common with the wheelset coordinate system, but they are rotated along the x axis, due to the conicity of the wheels and the shape of the rail. The angle formed by the contact plane and the wheelset axle is called δ_l and δ_r for the left and right wheel respectively. The rotation matrices from the contact points reference system to the wheelset reference system are:

$$ {}^{W}\mathbf{T}^{C_l} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\delta_l & -\sin\delta_l \\ 0 & \sin\delta_l & \cos\delta_l \end{bmatrix} \tag{B.7} $$

$$ {}^{W}\mathbf{T}^{C_r} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\delta_r & \sin\delta_r \\ 0 & -\sin\delta_r & \cos\delta_r \end{bmatrix} \tag{B.8} $$

Here the linearization of the sinusoidal functions cannot be performed because the contact angle can be very big, in particular when flange contact occurs. The application of the rotation matrices from the wheelset reference frame to the track following reference frame will give:

$$ {}^{F}\mathbf{T}^{C_l} = {}^{F}\mathbf{T}^{W} \cdot {}^{W}\mathbf{T}^{C_l} \approx \begin{bmatrix} 1 & -\psi\cos\delta_l & \psi\sin\delta_l \\ \psi & \cos(\delta_l+\phi) & -\sin(\delta_l+\phi) \\ 0 & \sin(\delta_l+\phi) & \cos(\delta_l+\phi) \end{bmatrix} \tag{B.9} $$

$$ {}^{F}\mathbf{T}^{C_r} = {}^{F}\mathbf{T}^{W} \cdot {}^{W}\mathbf{T}^{C_r} \approx \begin{bmatrix} 1 & -\psi\cos\delta_r & -\psi\sin\delta_r \\ \psi & \cos(\delta_r-\phi) & \sin(\delta_r-\phi) \\ 0 & -\sin(\delta_r-\phi) & \cos(\delta_r-\phi) \end{bmatrix} \tag{B.10} $$

Appendix C

Vehicle models

The aim of the project is to investigate the dynamics of four-axles two-bogies wagon that is running either on straight track or in a curve. The flexibility of DYTSI allows to define new vehicles and test them, thus several models have been used in order to compare their performances as well. In the following, the characteristics of each model will be shown and the important parameters will be listed.

C.1 Cooperrider model

The Cooperrider bogie is a well known model that has been extensively used in order to study the dynamics of railway vehicles. This bogie exhibits most of the dynamical phenomena that affect more complex bogies used in the industry today and in the meanwhile is composed by a simple design that make of it a good testing model. The vehicle is composed of a car body, two Cooperrider bogies and four wheel sets. In several previous works [Chr01, HP02], a simplified version of the Cooperrider model has been used, where only half part of a wagon was considered. In this work the whole wagon has been considered, thus additional dimensions are needed and some modification to the model has to be done. In [Chr01, HP02], the only motion considered for the car body was the rolling motion. Since a complete wagon is modeled here, yaw and pitch

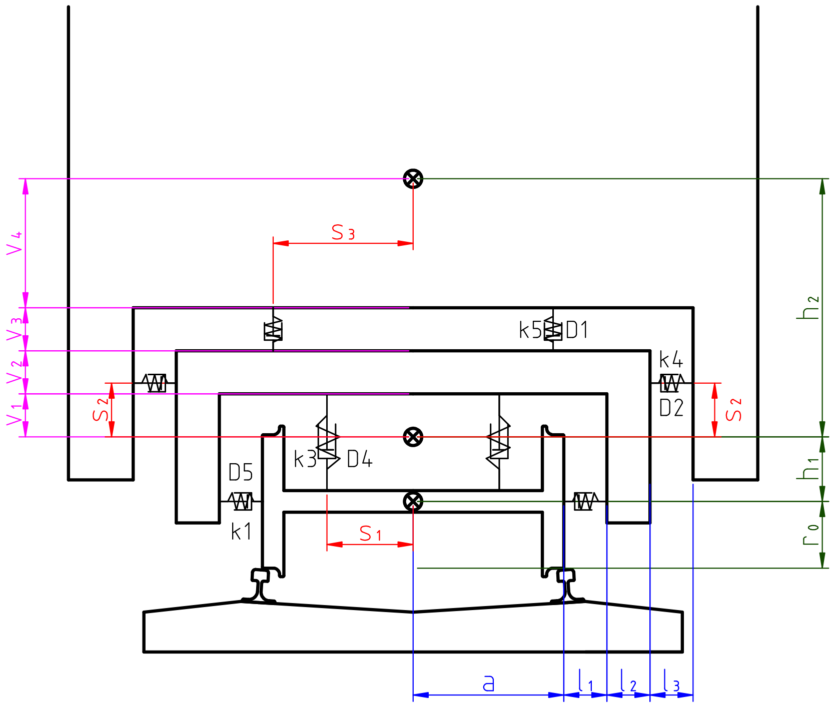


Figure C.1: Front view of the Cooperrider model.

rotations of the car body have to be considered as well. The torsional springs and dampers used in [Chr01, HP02] in order to counteract the yaw movement of the bogie frame, add now a torque to the car body as well. This torque is not centered in the center of mass of the car body, making it difficult to compute the net torque and the displacement caused by the torsional springs and dampers. In the industry, this kind of elements are not usually mounted, instead yaw springs and dampers are attached on the side of the bogie frame. The effect is not exactly the same of having a torsional element in the center of the bogie frame, because some lateral and sometimes vertical forces can be added.
Figures C.1 and C.2 show the front and top view of the Cooperrider model. The design is symmetric with respect to the vertical axis passing through the center of mass of the car body. All the stiff elements are considered to be unidirectional, thus the `LateralSpring`, `LongitudinalSpring` and `VerticalSpring` classes are used for modeling all the springs. The dampers are modeled as `VectorDampers`. The damping elements in the primary suspensions are added but set to zero, so that the basic model has the same characteristics of the one used in [Chr01, HP02], but further investigation can be done adding damping elements, a common practice in the industry.

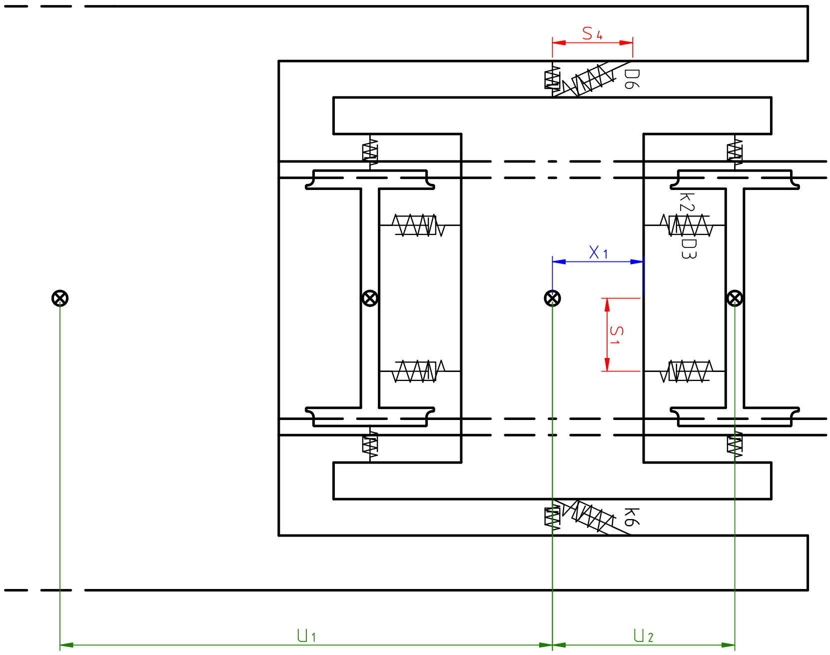


Figure C.2: Top view of the Cooperrider model.

Parm.	Value	Unit	Parm.	Value	Unit
r_0	0.425	$[m]$	a	0.75	$[m]$
h_1	0.0762	$[m]$	h_2	1.5584	$[m]$
l_1	0.30	$[m]$	l_2	0.30	$[m]$
l_3	0.30	$[m]$	x_1	0.349	$[m]$
v_1	0.6488	$[m]$	v_2	0.30	$[m]$
v_3	0.30	$[m]$	v_4	0.3096	$[m]$
s_1	0.62	$[m]$	s_2	0.6584	$[m]$
s_3	0.68	$[m]$	s_4	0.759	$[m]$
u_1	7.5	$[m]$	u_2	1.074	$[m]$

Table C.1: Dimensions of the Cooperrider model.

Tables C.1, C.2 and C.3 list the parameters used for the Cooperrider model. The dimensions are used for identifying all the attack points for each element of the system. The dimensions needed by DYTSI for identifying such points depends of the model implementation and not on DYTSI itself.

Parm.	Value	Unit	Parm.	Value	Unit
m_c	44388.0	$[kg]$	I_{cx}	$2.80 \cdot 10^5$	$[kgm^2]$
I_{cy}	$5.0 \cdot 10^5$	$[kgm^2]$	I_{cz}	$5.0 \cdot 10^5$	$[kgm^2]$
m_f	2918.0	$[kg]$	I_{fx}	6780.0	$[kgm^2]$
I_{fy}	6780.0	$[kgm^2]$	I_{fz}	6780.0	$[kgm^2]$
m_w	1022.0	$[kg]$	I_{wx}	678.0	$[kgm^2]$
I_{wy}	80.0	$[kgm^2]$	I_{wz}	678.0	$[kgm^2]$

Table C.2: Mass and Inertia values for the Car Body, the Bogie Frames and the Wheel Sets. The subscripts c, f and w are used for the three bodies and the directions of the inertia x, y and z are written in the body following reference system used in section 2.1.

Parm.	Value	Unit	Parm.	Value	Unit
k_1	1823.0	$[kN/m]$	k_2	3646.0	$[kN/m]$
k_3	3646.0	$[kN/m]$	k_4	182.3	$[kN/m]$
k_5	333.3	$[kN/m]$	k_6	903.35	$[kN/m]$
D_1	20.0	$[kNs/m]$	D_2	29.2	$[kNs/m]$
D_3	0.0	$[kNs/m]$	D_4	0.0	$[kNs/m]$
D_5	0.0	$[kNs/m]$	D_6	166.669	$[kNs/m]$

Table C.3: Stiffness and damping parameters of the Cooperrider model.

This model is provided of a wheel set with profile S1002 (DSB97-1 is used in [Chr01, HP02]) and is laid on a track with standard gauge, rail inclination of 1/40 and rail profile UIC60. The RSGEO table with the static contact parameters has been computed using the RSGEO routine version 3.01 (Build 101 - 07-11-2010), whereas [Chr01, HP02] used the version 6.10 (Build 20-07-99). The multiple contact patches are always approximated by one unique patch[PS91] and the values are then interpolated by a cubic spline. The resulting values are shown in figure 4.9.
The model is described by the class `CooperriderOrthogonalModel`, whereas the class `CooperriderModel` represents a model with all `VectorSpring`s and `VectorDamper`s. Both of them have been tested and the orthogonal model has shown to be as valid as the vector model for showing the dynamical behaviors that are researched in this work. The parameters of the models can be modified using the XML input file shown in listing C.1.

Listing C.1: Parameters for the Cooperrider model in the XML input file

```
<Model>
   <CooperriderOrthogonalModel>
      <RSGEO_Path Interp="cspline">RSGEO/1o40/RSGEO.dat</
```

```
          RSGEO_Path>
      <r0>0.425</r0>
      <h1>0.0762</h1>
      <h2>1.5584</h2>
      <a>0.75</a>
      <l1>0.30</l1>
      <l2>0.30</l2>
      <l3>0.30</l3>
      <v1>0.6488</v1>
      <v2>0.30</v2>
      <v3>0.30</v3>
      <v4>0.3096</v4>
      <u1>7.5</u1>
      <u2>1.074</u2>
      <x1>0.349</x1>
      <s1>0.62</s1>
      <s2>0.6584</s2>
      <s3>0.68</s3>
      <s4>0.759</s4>
      <k1>1823000.0</k1>
      <k2>3646000.0</k2>
      <k3>3646000.0</k3>
      <k4>182300.0</k4>
      <k5>333300.0</k5>
      <k6>1000000.0</k6>
      <D1>20000.0</D1>
      <D2>29200.0</D2>
      <D3>0.0</D3>
      <D4>0.0</D4>
      <D5>0.0</D5>
      <D6>185000.0</D6>
   </CooperriderOrthogonalModel>
</Model>
```

C.2 Very high speed power car (ALSTOM)[1]

The section has been removed due to confidentiality agreements.

[1] The design and parameters contained in this section are property of ALSTOM Transport. They were kindly provided for the realization of this work. The exploitation and reproduction of the contained material is strictly prohibited.

Appendix D

DYTSI usage

DYnamics Train SImulation is a solver of railway vehicle dynamical systems. The physical laws that govern the motion of the multibody system are hidden in the implementation of DYTSI and an interface is provided in order to define new specific models to be tested. This section represents an end user guide to the usage of DYTSI.

D.1 System requirements

DYTSI is written in C++ and was developed on a GNU/Linux OS and a Sun Solaris OS. The program can be compiled using GNU Compiler Collection GCC (tested version 4.4.x). The program requires the following packages to be installed:

- `pkg-config`: generates automatically the flags for other packages on which DYTSI depends (tested version 0.25)
- `Xerces-c`[1]: provides all the tool for the manipulation of XML files (tested version 3.1.1).

[1] http://xerces.apache.org/xerces-c/

- `GSL`[2]: the GNU Scientific library (tested version 1.15, required version > 1.15).

In order to check that the exact or higher version of the tools are correctly installed on the machine, use the commands:

```
$ pkg-config --modversion xerces-c
$ pkg-config --modversion gsl
```

D.2 Compiling

DYTSI is written in C++ and was developed on a GNU/Linux OS and a Sun Solaris OS. The program can be compiled using GNU Compiler Collection GCC. Since there is an high number of files to be compiled, the best way for doing this, is to use the `qmake` command. Along with the distribution there is a file `DYTSI.pro` that contains the list of files necessary for the compilation, the definition of the output path and some auxiliary settings. Using the following commands the program should be compiled without problems.

```
$ qmake DYTSI.pro
$ make
```

D.3 Model description

The aim of having a clear separation between the core of the dynamical system solver and the particular models that are tested, is that of facilitate the creation and modification of new models. Each model can be described by a C++ class that assemble the elements and a set of parameters that are passed using the XML input file.

D.3.1 XML input file

The XML input file defines the characteristics of a list of simulations. Each simulation will describe three parts:

[2] http://www.gnu.org/software/gsl/

- the parameters of the model employed
- the characteristics of the simulation
- the settings of the solver

The format of the XML input file is better explained by an example (more examples can be found along with the software). The listing D.1 shows an input file that defines a transient analysis simulation of a Cooperrider model on straight track over a period of 60 seconds. Each block of lines will be referenced and explained.

3 : The main tag for each XML input file is the `input`, that identify the document as a DYTSI XML input file. The `OutputFolder` attribute has to be defined with the paths of an existing/not-existing directory where to store the output of the simulations performed.

7 - 83 : each simulation in the input file is listed in the order of execution. In this case just a Transient analysis will be performed. The possible tags here are `Transient` or `Bifurcation`. The argument `TestName` will be the name assigned to the test and also the name of the subdirectory where the particular test will be stored.

8 - 14 : the information about the solver for the system of differential equations. The attribute `Name` will allow to chose amongst the implemented solvers. The possible options for this attribute are: `bsimp` (Bulirsch-Stoer, see sec. 3.1), `bdf` (Backward Differentiation Formula, see sec. 3.2), `sdirkNT1` or `sdirkJKT` (ESDIRK methods, see sec. 3.3). Inside the `Solver`'s block, the precision parameters for the step size controller have to be defined (the relative precision is not considered in the ESDIRK methods), the initial step length, the frequency with which the solution is stored (in seconds between each storing time) and the frequency with which the Jacobian is stored.

15 - 18 : some parameters that define the output format. Actually the values shown are the only available now, but extensions to DYTSI could require the usage of such parameters.

19 - 29 : a listing of parameters that belong to the analyzed model. The names of these parameters depend on the the particular implementation of the class file for the model (see sec. D.3.2). The RSGEO table can be included amongst these parameters and an interpolation function can be specified in the attribute `Interp`. If no interpolation function is specified, the linear interpolation will be assumed. The possible values for this field are `linear`, `cspline` or `akima` (see sec. 4.2). Some elements can have a piece wise

definition, like the example bumpstop in lines 24-27, and an interpolation function can be defined by the `Interp` attribute.

30 - 33 : the time span of the simulation.

34 - 36 : the running speed of the model, the radius of the track (1e99 stands for straight track) and the applied cant. In a transient analysis these values are constant, but in a bifurcation analysis ranges of speeds, radius and cants have to be defined (see some example distributed with the software).

37 - 82 : the starting values for the simulation. They are listed by the components they belong to and with the name of the degree of freedom as they are presented in section 2.4 (`BETA` is used for the pitch motion of the wheel sets, whereas `CHIDOT` is used for the pitch of the rest of the components). The components are declared in a nested order starting from the car body.

Listing D.1: Example of an XML input file.

```
1  <?xml version="1.0" encoding="ISO-8859-1"?>
2
3  <input OutputFolder="./Output/2011-07-01-CooperriderOrth-Transient"
4  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5  xsi:noNamespaceSchemaLocation="DYTSI_Input.xsd">
6
7     <Transient TestName="06-Transient">
8        <Solver Name="sdirkNT1">
9           <AbsPrecision>1e-8</AbsPrecision>
10          <RelPrecision>1e-8</RelPrecision>
11          <InitStepLength>1e-6</InitStepLength>
12          <StoreFrequency>0.01</StoreFrequency>
13          <JacStoreFrequency>0.5</JacStoreFrequency>
14       </Solver>
15       <Output>
16          <Format>TAB</Format>
17          <Plotting>GNUPlot</Plotting>
18       </Output>
19       <Model>
20          <CooperriderOrthogonalModel>
21             <RSGEO_Path Interp="cspline">RSGEO/1o40/RSGEO.dat</RSGEO_Path>
22             <r0>0.425</r0>
23             ...
24             <BUMPSTOP_PWLVals Interp="akima">
25                   <X>-0.100 -0.0400 -0.0200 0.0 0.0200 0.0400 0.1000</X>
26                   <Y>-1000.0e3 -300.0e3 0.0 0.0 0.0 300.0e3 1000.0e3</Y>
```

```
      </BUMPSTOP_PWLVals>
    </CooperriderOrthogonalModel>
  </Model>
  <TSpan>
    <StartValue>0.0</StartValue>
    <EndValue>60.0</EndValue>
  </TSpan>
  <Speed>30.0</Speed>
  <Radius>1e99</Radius>
  <Phi_se>0.0</Phi_se>
  <StartingValues>
    <CarBody>
      <CarBodyValues>
      <Y>0.00001</Y>
      <YDOT>0.0</YDOT>
      <Z>0.0</Z>
      ...
      </CarBodyValues>
      <LeadingBogieFrame>
        <LeadingBogieFrameValues>
          <Y>0.0</Y>
          <YDOT>0.0</YDOT>
          ...
        </LeadingBogieFrameValues>
        <LeadingWheelSet>
          <LLWheelSetValues>
            <Y>0.0</Y>
            <YDOT>0.0</YDOT>
            ...
          </LLWheelSetValues>
        </LeadingWheelSet>
        <TrailingWheelSet>
          <LTWheelSetValues>
            <Y>0.0</Y>
            <YDOT>0.0</YDOT>
            ...
          </LTWheelSetValues>
        </TrailingWheelSet>
      </LeadingBogieFrame>
      <TrailingBogieFrame>
        <TrailingBogieFrameValues>
          <Y>0.0</Y>
          ...
        </TrailingBogieFrameValues>
        <LeadingWheelSet>
          <TLWheelSetValues>
          ...
```

```
                  </TLWheelSetValues>
               </LeadingWheelSet>
               <TrailingWheelSet>
                  <TTWheelSetValues>
                  </TTWheelSetValues>
               </TrailingWheelSet>
            </TrailingBogieFrame>
         </CarBody>
      </StartingValues>
   </Transient>

</input>
```

D.3.2 Model Class

Each model has to be defined by a class in the `DYTSI_Models` namespace. The tools provided by the namespace `DYTSI_Modelling` can be used for assembling a train model. The Cooperrider model will be used as an example in order to explain how to define new models and to link them to DYTSI.
New models will be defined by adding to the `DYTSI.pro` file the source (`.cpp`) and the header (`.h`) files and re-compiling the whole program. The listings D.2 and D.3 show the header and the source files for the Cooperrider orthogonal model. The structure of this files is quite standard, so that it is easy to write new ones. They will be explained line by line in order to make clear what needs to be included in the definition of a model.
The header file must contain:

3 : include the `DYTSI_Modelling` framework.

4 - 7 : include additional tools.

10 : the class is a extension of the `GeneralModel` class.

15 : this class returns the order with which the components will be listed in the output.

17 - 21 : the functions that returns the parameters needed by the model.

26 - 29 : the functions for modifying the parameters needed for the construction of the model.

31 : function for loading from the XML definition the parameters needed for the construction of the model.

33 - 36 : the parameters needed for the construction of the model.

39 : main function that assembles the model properly.

40 : defines how the static load is distributed amongst the stiff vertical elements of the model.

41 : function that loads the starting values for the simulation from the XML input file.

Listing D.2: Header file for the Cooperrider orthogonal model (CooperriderOrthogonalModel.h)

```
#ifndef COOPERRIDERORTHOGONALMODEL_H
#define  COOPERRIDERORTHOGONALMODEL_H
#include "../GeneralModel/DYTSI_Modelling.h"
#include <string>

#include <xercesc/dom/DOM.hpp>
#include <xercesc/dom/DOMElement.hpp>

namespace DYTSI_Models {
    class CooperriderOrthogonalModel : public DYTSI_Modelling::GeneralModel {
    public:
        CooperriderOrthogonalModel();
        virtual ~CooperriderOrthogonalModel();

        std::vector<DYTSI_Modelling::Component*> getComponentList();

        double getD1() const;
        ...
        double getS4() const;

        XERCES_CPP_NAMESPACE::DOMNode* getRSGEO_path() const;

    private:
        typedef DYTSI_Modelling::GeneralModel super;

        void setRSGEO_path(XERCES_CPP_NAMESPACE::DOMNode* RSGEO_Path);
        void setD1(double D1);
        ...
        void setS4(double val);

        bool loadProperties(XERCES_CPP_NAMESPACE::DOMElement * el);
```

```

        XERCES_CPP_NAMESPACE::DOMNode* pathRSGEO;
        double k1;
        ...
        double s4;

    protected:
        bool setupModel(XERCES_CPP_NAMESPACE::DOMElement* el);
        bool setupStaticLoads();
        bool loadStartingValues(XERCES_CPP_NAMESPACE::DOMElement* el, std::vector<double>& startingValues);

    };
}

#endif   /* COOPERRIDERORTHOGONALMODEL_H */
```

The source code of the Cooperrider model is listed in D.3, where some intuitive parts have been omitted in order to shorten it down (the complete version of the file is distributed along with the software). The main parts of the class will be described here.

21 - 53 : returns the list of the components in the order that will be used for the output.

55 - 73 : loads the attributes that are listed in the XML file. The tools provided by the `XMLaux` class are used for reading from the XML file.

75 - 110 : laods the starting values provided in the XML input file. The tools provided by the `XMLaux` class are used for reading the XML file. Here, the order and the structure in which the starting values are written in the XML file is defined. Each component reads its own starting values from the XML node that is referred to it.

112 - 439 : the model is assembled. The properties are loaded and the components are attached using the suspension elements required.

441 - 467 : auxiliary functions for getting and setting the parameters read from the XML input file.

Listing D.3: Source code for the Cooperrider orthogonal model (`CooperriderOrthogonalModel.cpp`)

```
#include "CooperriderOrthogonalModel.h"

#include "../Application/Input/XMLaux.h"
#include "../Application/Input/RSGEOTableReader.h"
#include "../Application/PrintingHandler.h"

#include <gsl/gsl_vector.h>

#include <sstream>

namespace DYTSI_Models {
    const double g = 9.81;

    CooperriderOrthogonalModel::CooperriderOrthogonalModel()
    : DYTSI_Modelling::GeneralModel(){
    }

    CooperriderOrthogonalModel::~CooperriderOrthogonalModel
        () {
    }

    std::vector<DYTSI_Modelling::Component*>
        CooperriderOrthogonalModel::getComponentList(){
        std::vector<DYTSI_Modelling::Component*> out;

        // Add the Root Component (Car Body)
        DYTSI_Modelling::Component* carBodyComponent = this
            ->getRootComponent();
        out.push_back(carBodyComponent);

        // Add the Leading Bogie Frame
        DYTSI_Modelling::Component*
            leadingBogieFrameComponent = carBodyComponent->
            getLowerLeading()->getComponent();
        out.push_back(leadingBogieFrameComponent);

        // Add the Trailing Bogie Frame
        DYTSI_Modelling::Component*
            trailingBogieFrameComponent = carBodyComponent->
            getLowerTrailing()->getComponent();
        out.push_back(trailingBogieFrameComponent);

        // Add the Leading Wheelset in the Leading Bogie
            Frame
        DYTSI_Modelling::Component* llWheelSetComponent =
            leadingBogieFrameComponent->getLowerLeading()->
            getComponent();
```

```
        out.push_back(llWheelSetComponent);

        // Add the Trailing Wheelset in the Leading Bogie Frame
        DYTSI_Modelling::Component* ltWheelSetComponent = leadingBogieFrameComponent->getLowerTrailing()->getComponent();
        out.push_back(ltWheelSetComponent);

        // Add the Leading Wheelset in the Trailing Bogie Frame
        DYTSI_Modelling::Component* tlWheelSetComponent = trailingBogieFrameComponent->getLowerLeading()->getComponent();
        out.push_back(tlWheelSetComponent);

        // Add the Trailing Wheelset in the Trailing Bogie Frame
        DYTSI_Modelling::Component* ttWheelSetComponent = trailingBogieFrameComponent->getLowerTrailing()->getComponent();
        out.push_back(ttWheelSetComponent);

        return out;
    }

    bool CooperriderOrthogonalModel::loadProperties(XERCES_CPP_NAMESPACE::DOMElement* el){
        bool errCode = false;

        // Set RSGEO path
        errCode = XMLaux::loadRSGEOProperty(el,"RSGEO_Path",this->pathRSGEO);
        if (errCode) return errCode;

        // Set r0
        errCode = XMLaux::loadDoubleProperty(el,"r0",this->r0);
        if (errCode) return errCode;

        ...

        // Set D6
        errCode = XMLaux::loadDoubleProperty(el,"D6",this->D6);
        if (errCode) return errCode;

```

```
        return errCode;
    }

    bool CooperriderOrthogonalModel::loadStartingValues(XERCES_CPP_NAMESPACE::DOMElement* el, std::vector<double>& startingValues){
        bool error = false;

        if (this->getSettedUp() == false){
            error = true;
        } else {
            // Set the y0 vector to the correct size
            startingValues.resize(this->get_N_DOF());

            // Set Starting values for the CarBody root component
            DYTSI_Modelling::CarBodyComponent* carBody = (DYTSI_Modelling::CarBodyComponent*)this->getRootComponent();
            XERCES_CPP_NAMESPACE::DOMElement* carBodySV;
            error = XMLaux::loadElementProperty(el, "CarBody", carBodySV);
            if (error) return error;
            XERCES_CPP_NAMESPACE::DOMElement* carBodyValues;
            error = XMLaux::loadElementProperty(carBodySV, "CarBodyValues", carBodyValues);
            if (error) return error;
            carBody->loadStartingValues(carBodyValues, startingValues);

            // Set Starting values for the Leading Bogieframe
            DYTSI_Modelling::BogieFrameComponent* leadingBogieFrame =
                    (DYTSI_Modelling::BogieFrameComponent*)carBody->getLowerLeading()->getComponent();
            XERCES_CPP_NAMESPACE::DOMElement* leadingBogieFrameSV;
            error = XMLaux::loadElementProperty(carBodySV, "LeadingBogieFrame", leadingBogieFrameSV);
            if (error) return error;
            XERCES_CPP_NAMESPACE::DOMElement* leadingBogieFrameValues;
            error = XMLaux::loadElementProperty(leadingBogieFrameSV, "LeadingBogieFrameValues", leadingBogieFrameValues);
```

```
            if (error) return error;
            leadingBogieFrame->loadStartingValues(leadingBogieFrameValues, startingValues);

            ...

        }

        return error;
    }

    bool CooperriderOrthogonalModel::setupModel(XERCES_CPP_NAMESPACE::DOMElement* el ){
        bool error = this->loadProperties(el);
        if ( error ) return error;

        DYTSI_Modelling::CarBodyComponent *carBody;
        DYTSI_Modelling::BogieFrameComponent *leadingBogieFrame;
        DYTSI_Modelling::BogieFrameComponent *trailingBogieFrame;

        // Setup the Car Body as root Component
        {
            double mass = 44388.0;
            double Ix = 2.80e5;
            double Iy = 5.0e5;
            double Iz = 5.0e5;
            gsl_vector* centerOfGeometry = gsl_vector_calloc(3);
            gsl_vector_set(centerOfGeometry, 2, r0+h1+h2);
            gsl_vector* centerOfMass = gsl_vector_calloc(3);
            carBody = new DYTSI_Modelling::CarBodyComponent(centerOfGeometry, centerOfMass,
                    mass,Ix,Iy,Iz,this->getExternalComponents(),
                    "Car Body", this);
            this->setRootComponent( carBody );
        };

        // Setup the Bogieframes, the connection and add them to the model
        {
            double mass = 2918.9;
            double Ix = 6780.0;
            double Iy = 6780.0;
            double Iz = 6780.0;
```

```
            // Leading Bogie Frame
            gsl_vector* centerOfGeometryLeading =
                gsl_vector_calloc(3);
            gsl_vector_set(centerOfGeometryLeading, 0, u1);
            gsl_vector_set(centerOfGeometryLeading, 2, r0+h1
                );
            gsl_vector* centerOfMassLeading =
                gsl_vector_calloc(3);
            leadingBogieFrame = new DYTSI_Modelling::
                BogieFrameComponent(
                    DYTSI_Modelling::LEADING,
                        centerOfGeometryLeading,
                        centerOfMassLeading,
                    mass, Ix, Iy, Iz, u2,
                    this->getExternalComponents(),
                    "Leading Bogie Frame", this );

        // Trailing Bogie Frame
        ...

            ///////////////////////////////
            // Secondary Suspensions Leading bogie frame
            DYTSI_Modelling::BogieFrameCarBodyConnector *
                connectorLeading =
                    new DYTSI_Modelling::
                        BogieFrameCarBodyConnector(
                    "Secondary Suspensions Leading bogie
                        frame", carBody,
                    leadingBogieFrame, DYTSI_Modelling::UP,
                        DYTSI_Modelling::LL);

            // Left lateral spring and damper
            {
                gsl_vector* connectionPointUp =
                    gsl_vector_alloc(3);
                gsl_vector_set(connectionPointUp, 0, u1);
                gsl_vector_set(connectionPointUp, 1, a+l1+l2
                    +l3);
                gsl_vector_set(connectionPointUp, 2, -(h2-s2
                    ));
                gsl_vector* connectionPointDown =
                    gsl_vector_alloc(3);
                gsl_vector_set(connectionPointDown, 0, 0);
                gsl_vector_set(connectionPointDown, 1, a+l1+
                    l2);
                gsl_vector_set(connectionPointDown, 2, s2);
```

```
        DYTSI_Modelling::LinearFunction* stiffnessFunction = new DYTSI_Modelling::LinearFunction(k4);
        DYTSI_Modelling::LinearFunction* dampingFunction = new DYTSI_Modelling::LinearFunction(D2);
        DYTSI_Modelling::LateralSpring *spring = new DYTSI_Modelling::LateralSpring(
                "Left lateral spring", connectionPointUp, connectionPointDown, stiffnessFunction);
        connectorLeading->addLink(spring);
        DYTSI_Modelling::LateralDamper *damper = new DYTSI_Modelling::LateralDamper(
                "Left lateral damper", connectionPointUp, connectionPointDown, dampingFunction);
        connectorLeading->addLink(damper);
    };

    // Right lateral spring and damper
    ...

    // Left Vertical Spring and damper
    {
        gsl_vector* connectionPointUp = gsl_vector_alloc(3);
        gsl_vector_set(connectionPointUp, 0, u1);
        gsl_vector_set(connectionPointUp, 1, s3);
        gsl_vector_set(connectionPointUp, 2, -v4);
        gsl_vector* connectionPointDown = gsl_vector_alloc(3);
        gsl_vector_set(connectionPointDown, 0, 0);
        gsl_vector_set(connectionPointDown, 1, s3);
        gsl_vector_set(connectionPointDown, 2, v1+v2);
        DYTSI_Modelling::LinearFunction* stiffnessFunction = new DYTSI_Modelling::LinearFunction(k5);
        DYTSI_Modelling::LinearFunction* dampingFunction = new DYTSI_Modelling::LinearFunction(D1);
        DYTSI_Modelling::VerticalSpring *spring = new DYTSI_Modelling::VerticalSpring(
```

```
                    "Left Vertical Spring", connectionPointUp, connectionPointDown, stiffnessFunction);
            connectorLeading->addLink(spring);
            this->staticLoadedSecondarySuspensionSpringList.push_back(spring);
            DYTSI_Modelling::VerticalDamper *damper = new DYTSI_Modelling::VerticalDamper(
                    "Left Vertical Spring", connectionPointUp, connectionPointDown, dampingFunction);
            connectorLeading->addLink(damper);
        };

        // Right Vertical Spring and damper
        ...

        // Left yaw spring and damper
        {
            gsl_vector* connectionPointUp = gsl_vector_alloc(3);
            gsl_vector_set(connectionPointUp, 0, u1 + s4);
            gsl_vector_set(connectionPointUp, 1, a+l1+l2+l3);
            gsl_vector_set(connectionPointUp, 2, -h2);
            gsl_vector* connectionPointDown = gsl_vector_alloc(3);
            gsl_vector_set(connectionPointDown, 0, 0.0);
            gsl_vector_set(connectionPointDown, 1, a+l1+l2);
            gsl_vector_set(connectionPointDown, 2, 0.0);
            DYTSI_Modelling::LinearFunction* dampingFunction = new DYTSI_Modelling::LinearFunction(D6);
            DYTSI_Modelling::VectorDamper * damper = new DYTSI_Modelling::VectorDamper(
                    "Left Yaw Damper", connectionPointUp, connectionPointDown, dampingFunction);
            connectorLeading->addLink(damper);
            DYTSI_Modelling::LinearFunction* stiffnessFunction = new DYTSI_Modelling::LinearFunction(k6);
```

```
            DYTSI_Modelling::LongitudinalSpring* spring = new DYTSI_Modelling::LongitudinalSpring(
                "Left Yaw Spring", connectionPointUp, connectionPointDown, stiffnessFunction);
            connectorLeading->addLink(spring);
        };

        // Right yaw damper
        ...

        // End Connector Front Bogie Frame
        //////////////////////////////

        //////////////////////////////
        // Secondary Suspension Trailing bogie frame
        DYTSI_Modelling::BogieFrameCarBodyConnector * connectorTrailing =
            new DYTSI_Modelling::BogieFrameCarBodyConnector(
            "Secondary Suspension Trailing bogie frame", carBody,
            trailingBogieFrame, DYTSI_Modelling::UP, DYTSI_Modelling::LT);

        // Left lateral spring and damper
        ...

        // Right lateral spring and damper
        ...

        // Left Vertical Spring and damper
        ...

        // Right Vertical Spring and damper
        ...

        // Left yaw spring and damper
        ...

        // Right yaw damper
        ...

        // End Connector Trailing Bogie Frame
        //////////////////////////////
```

```
        }; // End Bogie frames and upper connectors

        // Set up the wheelsets and connect them to the bogie frames
        {
            double mass = 1022.0;
            double Ix = 678.0;
            double Iy = 80.0;
            double Iz = 678.0;
            double a = 0.75;
            double mu = 0.15;
            double G = 2.1e11/(2*(1-0.27));
            std::vector<DYTSI_Modelling::RSGEOTable*> rsgeo_table;
            DYTSI_Input::RSGEOTableReader::read(rsgeo_table, this->getRSGEO_path(), '#');

            DYTSI_Modelling::WheelSetComponent *llwheelset;
            DYTSI_Modelling::WheelSetComponent *ltwheelset;
            DYTSI_Modelling::WheelSetComponent *tlwheelset;
            DYTSI_Modelling::WheelSetComponent *ttwheelset;
            DYTSI_Modelling::WheelSetBogieFrameConnector *llconnector;
            DYTSI_Modelling::WheelSetBogieFrameConnector *ltconnector;
            DYTSI_Modelling::WheelSetBogieFrameConnector *tlconnector;
            DYTSI_Modelling::WheelSetBogieFrameConnector *ttconnector;

            // Leading Bogie Frame - Leading wheelset
            {
                gsl_vector* centerOfGeometry = gsl_vector_calloc(3);
                gsl_vector_set(centerOfGeometry, 0, u1 + u2);
                gsl_vector_set(centerOfGeometry, 2, r0);
                gsl_vector* centerOfMass = gsl_vector_calloc(3);
                llwheelset = new DYTSI_Modelling::WheelSetComponent(
                        DYTSI_Modelling::LEADING, centerOfGeometry, centerOfMass,
                        mass,Ix,Iy,Iz,a,mu,G,
                        this->getExternalComponents(),
                        rsgeo_table,
```

```
                "Leading Bogie - Leading Wheelset", this);
        };

        // Leading Bogie Frame - Trailing wheelset
        ...

        // Trailing Bogie Frame - Leading wheelset
        ...

        // Trailing Bogie Frame - Trailing wheelset
        ...

        ///////////////////////////////
        // Primary Suspension: Leading Bogie Frame - Leading Wheelset
        llconnector = new DYTSI_Modelling::WheelSetBogieFrameConnector(
                "Primary Suspension: Leading Bogie Frame - Leading Wheelset",
                leadingBogieFrame, llwheelset,
                DYTSI_Modelling::UP, DYTSI_Modelling::LL);

        // Left lateral spring
        {
            gsl_vector* connectionPointUp = gsl_vector_alloc(3);
            gsl_vector_set(connectionPointUp, 0, u2);
            gsl_vector_set(connectionPointUp, 1, a+l1);
            gsl_vector_set(connectionPointUp, 2, -h1);
            gsl_vector* connectionPointDown = gsl_vector_calloc(3);
            gsl_vector_set(connectionPointDown, 1, a);
            DYTSI_Modelling::LinearFunction* stiffnessFunction = new DYTSI_Modelling::LinearFunction(k1);
            DYTSI_Modelling::LateralSpring* spring = new DYTSI_Modelling::LateralSpring(
                "Left lateral spring",
                    connectionPointUp,
                    connectionPointDown,
                    stiffnessFunction);
            llconnector->addLink(spring);
            DYTSI_Modelling::LinearFunction* dampingFunction = new DYTSI_Modelling::LinearFunction(D5);
```

```
                DYTSI_Modelling::VectorDamper* damper = new
                    DYTSI_Modelling::VectorDamper(
                         "Left lateral damper",
                             connectionPointUp,
                             connectionPointDown,
                             dampingFunction);
                llconnector->addLink(damper);
            };

            // Right lateral spring
            ...

            // Left vertical spring
            {
                gsl_vector* connectionPointUp =
                    gsl_vector_alloc(3);
                gsl_vector_set(connectionPointUp, 0, u2);
                gsl_vector_set(connectionPointUp, 1, s1);
                gsl_vector_set(connectionPointUp, 2, v1);
                gsl_vector* connectionPointDown =
                    gsl_vector_calloc(3);
                gsl_vector_set(connectionPointDown, 1, s1);
                DYTSI_Modelling::LinearFunction*
                    stiffnessFunction = new DYTSI_Modelling::
                    LinearFunction(k3);
                DYTSI_Modelling::VerticalSpring* spring =
                    new DYTSI_Modelling::VerticalSpring(
                         "Left vertical spring",
                             connectionPointUp,
                             connectionPointDown,
                             stiffnessFunction);
                llconnector->addLink(spring);
                this->
                    staticLoadedPrimarySuspensionSpringList.
                    push_back(spring);

                DYTSI_Modelling::LinearFunction*
                    dampingFunction = new DYTSI_Modelling::
                    LinearFunction(D4);
                DYTSI_Modelling::VectorDamper* damper = new
                    DYTSI_Modelling::VectorDamper(
                         "Left vertical damper",
                             connectionPointUp,
                             connectionPointDown,
                             dampingFunction);
                llconnector->addLink(damper);
            };
```

```
        // Right vertical spring
        ...

        // Left longitudinal spring
        {
            gsl_vector* connectionPointUp = gsl_vector_alloc(3);
            gsl_vector_set(connectionPointUp, 0, x1);
            gsl_vector_set(connectionPointUp, 1, s1);
            gsl_vector_set(connectionPointUp, 2, -h1);
            gsl_vector* connectionPointDown = gsl_vector_calloc(3);
            gsl_vector_set(connectionPointDown, 1, s1);
            DYTSI_Modelling::LinearFunction* stiffnessFunction = new DYTSI_Modelling::LinearFunction(k2);
            DYTSI_Modelling::LongitudinalSpring* spring = new DYTSI_Modelling::LongitudinalSpring(
                    "Left longitudinal spring", connectionPointUp, connectionPointDown, stiffnessFunction);
            llconnector->addLink(spring);
            DYTSI_Modelling::LinearFunction* dampingFunction = new DYTSI_Modelling::LinearFunction(D3);
            DYTSI_Modelling::VectorDamper* damper = new DYTSI_Modelling::VectorDamper(
                    "Left longitudinal damper", connectionPointUp, connectionPointDown, dampingFunction);
            llconnector->addLink(damper);
        };

        // Right longitudinal spring
        ...

        // End Setup connector: Leading Bogie Frame - Leading Wheelset
        ////////////////////////////////

        ////////////////////////////////
        // Primary Suspension: Leading Bogie Frame - Trailing Wheelset
```

```
            ltconnector = new DYTSI_Modelling::WheelSetBogieFrameConnector(
                    "Primary Suspension: Leading Bogie Frame - Trailing Wheelset",
                    leadingBogieFrame, ltwheelset,
                    DYTSI_Modelling::UP, DYTSI_Modelling::LT);

            // Left lateral spring
            ...

            // Right lateral spring
            ...

            // Left vertical spring
            ...

            // Right vertical spring
            ...

            // Left longitudinal spring
            ...

            // Right longitudinal spring
            ...

            // End Setup connector: Leading Bogie Frame - Trailing Wheelset
            ////////////////////////////////

            ////////////////////////////////
            // Primary Suspension: Trailing Bogie Frame - Leading Wheelset
            tlconnector = new DYTSI_Modelling::WheelSetBogieFrameConnector(
                    "Primary Suspension: Trailing Bogie Frame - Leading Wheelset",
                    trailingBogieFrame, tlwheelset,
                    DYTSI_Modelling::UP, DYTSI_Modelling::LL);

        ...

            // End Setup connector: Trailing Bogie Frame - Leading Wheelset
            ////////////////////////////////

```

```
            ////////////////////////////////
            // Primary Suspension: Trailing Bogie Frame - Trailing Wheelset
            ttconnector = new DYTSI_Modelling::WheelSetBogieFrameConnector(
                    "Primary Suspension: Trailing Bogie Frame - Trailing Wheelset",
                    trailingBogieFrame, ttwheelset,
                    DYTSI_Modelling::UP, DYTSI_Modelling::LT);

            ...

            // End Setup connector: Leading Bogie Frame - Leading Wheelset
            ////////////////////////////////
        };

        return error;
    }

    void CooperriderOrthogonalModel::setD1(double D1) {
        this->D1 = D1;
    }

    ...

    void CooperriderOrthogonalModel::setS4(double val){
        this->s4 = val;
    }

    double CooperriderOrthogonalModel::getL1() const{
        return this->l1;
    }

    ...

    double CooperriderOrthogonalModel::getS4() const{
        return this->l1;
    }

    XERCES_CPP_NAMESPACE::DOMNode* CooperriderOrthogonalModel::getRSGEO_path() const {
        return this->pathRSGEO;
    }

```

```
    void CooperriderOrthogonalModel::setRSGEO_path(
        XERCES_CPP_NAMESPACE::DOMNode* RSGEO_Path){
        this->pathRSGEO = RSGEO_Path;
    }
}
```

D.4 Running and Output

Once the program is compiled and linked to the additional models that has been built, the program can be run using the command

```
$ ./DYTSI [-T/-G] [-nt/-pt] <InputFile>.xml
```

where the first option tells DYTSI if the program is started in text mode (`-T`) or in graphic mode (`-G`) (only the text mode is implemented right now). The second argument tells the program whether to use no multithreading (`-nt`) or to use `pthread` multithreading (`-pt`) (not recommended). Last, the path to the XML input file has to be passed to the program. During the exectution, the terminal should show something like the listing shown in D.4.

Listing D.4: Example of the execution of DYTSI

```
$ ./DYTSI -T -nt Input/inputModifiedDimensionsStraight.xml
Starting the program
Thread mode: OFF
================================================
Starting the elaboration of the Input
0) File: Input/inputModifiedDimensionsStraight.xml
   Creating output folder: ./Output/2011-07-01-
       CooperriderOrth-Transient   [DONE]
   Test Name: 06-Transient
   Creating output folder: ./Output/2011-07-01-
       CooperriderOrth-Transient/06-Transient   [DONE]
      Cooperrider Orthogonal Model.
         Reading RSGEO file RSGEO/1o40/RSGEO.dat
         Interpolating RSGEO   [DONE]
      Solver Construction: "SDIRK"   [DONE]
      Solver Setup: "SDIRK"   [DONE]
      Output Handler Construction   [DONE]
      Loading Starting Values.   [DONE]
Finished the elaboration of the Input
================================================
```

```
================================================
Starting the Simulations

   Creating output folder: ./Output/2011-07-01-
       CooperriderOrth-Transient/06-Transient/Jacobians/   [
       DONE]

#####################################
    Transient Analysis: 06-Transient
|============================================
|           Solver call information
|============================================
|  Time Span:           [0,60]
|  Initial Step Length: 1e-06
|============================================
|           System Properties
|============================================
|  Speed:               30
|  Radius:                 1e+99
|  Phi_se:              0
|============================================


0%   10   20   30   40   50   60   70   80   90   100%
|----|----|----|----|----|----|----|----|----|----|
***********************
$
```

The simulation can finish successfully or some kinds of exit conditions can be met, making the simulation terminate and the next simulation to start. One exit condition could be that the lateral displacement of one wheel set exceeded the range of displacements contained in the RSGEO table (what could correspond to a derailment in real life). This exit condition will be signaled by the following output:

```
Leading Bogie - Trailing Wheelset[ID 4]: Derailment detected
   !
Current displacement: 0.017
error: input domain error
Current iteration number: 40383
Counters:
Function evaluations:   608423
Jacobian evaluations:   54929
Car Body[ID 0]: RHS evaluations         3904162
Leading Bogie Frame[ID 1]: RHS evaluations      4783026
```

```
Leading Bogie - Leading Wheelset[ID 3]: RHS evaluations
           2695725
Leading Bogie - Trailing Wheelset[ID 4]: RHS evaluations
          2695724
Trailing Bogie Frame[ID 2]: RHS evaluations     4783026
Trailing Bogie - Leading Wheelset[ID 5]: RHS evaluations
          2695724
Trailing Bogie - Trailing Wheelset[ID 6]: RHS evaluations
         2695724
Elapsed time for the solution: 1501.52 s
```

Each element has an unique ID and the number of function evaluations and Jacobian evaluations are shown in order to make easier the debug when necessary. The second exit condition that could be met is that the state of the system cause a non-linear element to work out of its described range. In this case the output will be similar but some information about the failing element will be provided. Elements of the suspensions have a unique ID as well. The termination of the program due to these conditions will not affect the data already generated and stored in the output folder.
When all the input has been elaborated and the simulations have been performed, the output folder should contain several files with the results. The output files can take up to hundreds of Megabytes even if the storing frequency is kept low. The list of files that should be present inside the output folder are:

```
$ ls -Rlh
.:
total 280M
-rw------- 1 bigo bigo 3,6K 2011-07-04 08:53 input.xml
drwx------ 1 bigo bigo  16K 2011-06-27 07:51 Jacobians
-rw------- 1 bigo bigo 280M 2011-06-27 08:07 output.dat

./Jacobians:
total 1,4M
-rw------- 1 bigo bigo 14K 2011-06-27 04:40 jac-01.dat
-rw------- 1 bigo bigo 14K 2011-06-27 04:41 jac-02.dat
-rw------- 1 bigo bigo 14K 2011-06-27 04:42 jac-03.dat
-rw------- 1 bigo bigo 14K 2011-06-27 04:43 jac-04.dat
-rw------- 1 bigo bigo 14K 2011-06-27 04:44 jac-05.dat
-rw------- 1 bigo bigo 14K 2011-06-27 04:45 jac-06.dat
...
```

The `input.xml` file contains the part of the XML input file that describe the actual test, so that this can be repeated if necessary. The file `output.dat` is the biggest file where the output is stored with the frequency defined in the input file. Each entry is stored in a new row of the file. The file contains also an header that explain the content of each column and can be used for plotting purposes, however an explanation of how the output is organized is necessary. As it was stated before, the order in which the components appear listed in the output file is defined by the `getComponentList` function in the model description class. However, for each component the output is stored always in the same order. The table D.1 shows the description of each column of the output file. The order in which the degrees of freedom are listed is:

$$x, \dot{x}, y, \dot{y}, z, \dot{z}, \phi, \dot{\phi}, \chi, \dot{\chi}, \psi, \dot{\psi}$$

Along with the output file DYTSI generate also the folder `Jacobians` where the Jacobian's matrices are stored with the frequency defined in the input file. The files named `jac-*.dat` contain the matrix and the numeration can be used to associate them to the time of the simulation. In particular, if the storing frequency of the Jacobian has been set to the time of the transient analysis, the Jacobian will be stored at the last stage of the transient, allowing the analysis of the steady state solution (if there is) of the system.

Cols.	Description
1	**Index of transient analysis**. Each transient analysis, starting from the one for the transient curve if present, is distinguished by this unique ID. In the bifurcation analysis where multiple transient analysis are performed sequentially, several IDs will be used.
2	**Entry time**. Time of the simulation at which the state of the system is sampled and stored.
3	**Step size**. Actual step size.
4	**Speed**. Actual speed of the vehicle.
5	**Radius**. Actual radius of the curve.
6	**Cant**. Actual cant of the curve.
7-16	**Car body DOF**. State of the degrees of freedom of the car body.
17-22	**Car body suspension forces**. Suspension forces acting on the car body.
23-29	**Car body gravitational forces**. Force components due to the gravitational force.
30-35	**Car body centrifugal force**. Forces due to the track geometry.
36-45	**Leading bogie frame DOF**.
46-51	**Leading bogie frame suspension forces**.
52-57	**Leading bogie frame gravitational forces**.
58-63	**Leading bogie frame centrifugal forces**.
63-72	**Trailing bogie frame DOF**.
73-84	...
85-90	**Trailing bogie frame centrifugal forces**.
91-99	**DOF of the leading wheel set (Leading BF)**.
100-109	**Contact forces of the leading wheel set(Leading BF)**. Actual guidance forces and creep forces.
110-115	**Suspension forces of the leading wheel set(Leading BF)**.
116-121	**Gravitational forces of the leading wheel set(Leading BF)**.
122-127	**Centrifugal forces of the leading wheel set(Leading BF)**.
128-136	**DOF of the trailing wheel set (Leading BF)**.
137-165	...
166-202	**Leading wheel set (Trailing BF)**.
203-238	**Trailing wheel set (Trailing BF)**.

Table D.1: Description of the columns in the output file.

Appendix E

RSGEO usage and data manipulation

The characteristics of the wheel-rail contact are very important for the dynamics of railway vehicles, so the correct generation of the values that describe this interaction is of primary importance. The model used in this work requires some static parameters that were introduced in table 4.1. These parameters can be precomputed and provided to DYTSI in the form of one or more RSGEO lookup tables. The program used in order to obtain these tables is RSGEO[KM10] version 3.01 (Build 101 - 07-11-2010) (A modified version, kindly provided by Walter Kik, was used in order to generate a more refined version of the RSGEO table). This program has a useful tutorial that explain how to use the program, however it's lacking the description of the output and what it represents. This section will explain which are the steps to be taken in order to generate a proper RSGEO table to be provided to DYTSI.

Step 1: Wheel profile generation

The rail profiles are quite standardized and the most common choice is the UIC60, where 60 stands for 60 kilograms per meter. The main difference between

tracks is given by the cant of the rail rather than by their profile. Thus the UIC60 profile is included along with the RSGEO routine.
A variety of wheel profiles are common in the industry and the choice of a correct combination with the rail profile can provide improved stability. The wheel profiles are usually defined by piecewise curves with respect to the displacement from the central tread of the profile. Another format for defining the wheel profiles is by measurements at various displaced points from the central tread. RSGEO uses a particular input file that defines the profiles using B-splines.
Passing from the piecewise definition to the measurement points to the cubic splines is rather easy. Mathematical software, like MatLab or Octave, can be used for the generation of the measurement points with the desired precision. The measurement points will be listed as subsequent tuples from the inside of the wheel toward the outside. This data can be read by the RSPROF program (version 2.72 - Build 072 01-09-2010), usually provided along with the RSGEO program, and a B-spline can be created. It's stressed that the data have to be scaled such that the points express dimensions in meters and the whole set of points have to be centred on the central tread position. The spline can be saved and will be stored in a file with extension `.prf`.
The program DYTSI expects the contact table for the right wheel to be provided (the left wheel data are assumed by symmetry), thus only the right profile needs to be generated.

Step 2: RSGEO tables generation

The profile generated using RSPROF can be used as input for RSGEO. The refinement requested by DYTSI in order to give proper results is rather high, so the usage of the batch mode of the RSGEO is recommended. It can be started using the file menu. Two files are requested by the program: the configuration file (extension `.CFG`) and the control file (extension `.PRS`). The listing E.1 shows an example of the configuration file used for the generation of the contact table of the combination UIC60 rails canted 1/20 with a particular GV40 wheel profile (lines 130-148). In this particular setting, a high refinement was wanted ($2 \cdot 10^{-6}$), but RSGEO has memory limits that doesn't allow to generate too big tables, so the tables were generated piece by piece starting from the displacement -0.025 to the displacement 0.025. This file refers to the last block of data in the range $[0.015; 0.025]$ (line 105).
The second file needed by RSGEO is the control file, where the profiles are defined. The listing E.2 shows the listing for the same example. A complete description of the values that have to be listed in this file is provided in the header of the file.
Once these two files are ready, the program can be run and the output files

will be placed in files called `<prefix>_*R.dat` or `<prefix>_*L.dat`, where the prefix can be chosen before starting the program and the asterisk is a number between 1 and 4. Only the tables about the right wheel are needed, thus the ones ending by `R`. These files contains all the information that are necessary to build up the RSGEO table in the format accepted by DYTSI. However some more manipulation has to be done in order to obtain good smooth tables.

Listing E.1: Example of the batch configuration file.

```
* =====================================
*  RSGEO parameter data
* =====================================

* =====================================
*  Material data
* =====================================
*
* Young's modul, poisson ratio wheel
  21.e10  .27
*
* Young's modul, poisson ratio rail
  21.e10  .27

* =====================================
*  Configuration data
* =====================================

***********>  BLOCK 1  <==============
*   wheelset data
*
*   yaw angle
    0.0

***********>  BLOCK 2  <==============
*
*   spacing of wheel back
    1.376


***********>  BLOCK 3  <==============
*
*   difference between wheel back and wheel base
*   and origin of the profile systems
*   1. left
*   2. right
*   3. origin of the profile systems at wheel back (1) or wheel base (0)
    0.062    0.062     1
```

```


***********>  BLOCK 4  <==============
*
*   rolling radii and wheel flange depths (optional)
*   1. R left  2. R right  3. WFD left   4. WFD right
*    0.445      0.445      0.028         0.028
    0.46      0.46      -1.0          -1.0

***********>  BLOCK 5  <==============
*   track data
*
*   curve radii
*   1. number of different radii
*      = 0, only straight track
*   2. different radii
*      0. => straight track
     1 0. 120.


***********>  BLOCK 6  <==============

*   1: t   -   value for gauge
*      f   -   value for the spacing of the radii
*   2. gauge or spacing
*   3. vertical  distance for the gauge measurement
*   .t.   1.435 .009
    .t.   1.435 .014
*   .f.   1.5     0.0

***********>  BLOCK 7  <==============

*   rail inclination
    2*.05

***********>  BLOCK 8  <==============

*   simulation on a roller rig
    .F.  no
*   .T.  yes, than the following values
*   1. radii of the rolls, left, right
*   2*.5
*   1. profile inclination againts the rolls, left, right
*      the rail inclination is then the inclination of the roll
*      axis. the profile inclination is the inclination of the
```

```
*      profil system against the roll axis.
*   2*.0
*   1. yaw angle left, right
*   2*0.

***********>  BLOCK 9  <==============

*   displacments
*   1. type
*       type  1:
*              2. initial lateral displacement
*              3. max. lateral displacement
*              4. increment
*       type  2:
*              2. initial lateral displacement
*              3. increment
*              4. number of steps
*       type  3:
*              2. increment
*              3. lift of wheel
*
    1    0.015   0.025  0.000002
*   2    0.   .0005  30
*   3    .0002   0.005

***********>  BLOCK 10  <==============

*   algorithm:
*   method
*      0 - lateral displacment
*      1 - quasilinearization
*      2 - quasilinearization UIC519 Klingel
*      3 - quasilinearization UIC519 Linear regression
    0

***********>  BLOCK 11  <==============

*    1. .true. normalforce iteration (not used)
*    2. relative residue of normal force iteration (not used)
*    3. no. of forces left side (not used)
*    4. values of forces left side
*    5. no. of forces right side (not used)
*    6. values of forces right side
  .t.  .005   1  82376.0  1  82376.0


```

```
***********>  BLOCK 12  <=============
*
*   profiles:
*
*   rail profile on the left side
* 1     2     3     4  5  6      7  8      9   10   11   12   13   14   15
*[ 0     0  1.0  1.0  2  3  0.025  1  0.001  0.0  0.0  0.05  0.05 0.05 0.05
*[ 0     0  1.0  1.0  2  3  0.025  1  0.001  0.0  0.0  50.0  50.0 50.0 50.0
*[ 0     0  1.0  1.0  2  3  0.025  1  0.001  0.0  0.0
  profil\batch\uic60.dat
*   rail profile on the right side
*[ 0     0  1.0  1.0  1  2  0.025  1  0.001  0.0  0.0
  profil\batch\uic60_.dat
* wheel profile on the left side
*[ '#'   0  1.0 -1.0  1  2  0.0    1  0.002  0.0  0.0
  profil\GV32_5.prf
* wheel profile on the right side
*[ '#'   0  1.0 -1.0  1  2  0.0    1  0.002  0.0  0.0
  profil\GV32_5.prf


***********>  BLOCK 13  <=============
*
*   spacing and level of of wheelbacks measure point A1 (down)
*
*    1. A1l spacing left
*    2. h1l level left
*    3. A1r spacing right
*    4. h1r level right
  1.440  0.0    1.440  0.0


***********>  BLOCK 14  <=============
*
*   spacing and level of of wheelbacks measure point A2 (middle)
*
*    1. A2l spacing left
*    2. h2l level left
*    3. A2r spacing right
*    4. h2r level right
  1.440  0.2500 1.440  0.2500

```

```

***********>  BLOCK 15  <=============
*
*   spacing and level of of wheelbacks measure point A3 (top
    )
*
*    1. A3l spacing left
*    2. h3l level left
*    3. A3r spacing right
*    4. h3r level right
  -1.0  -1.0  -1.0  -1.0


***********>  BLOCK 16  <=============
*
*   key words in the file header of measured profile data
*
*    1. Flag and key word for Gauge
*    2. Flag and key word for Wheels Flange Depth
*    3. Flag and key word for Wheels Back To Back
*    4. Flag and key word for Diameter Taperline
*       Flag = 0 - ignore the keyword
*              1 - use the keyword
    1   'Gauge='
    0   'SH='
    0   ''
    1   'DiameterTaperline='

* ====================================
```

Listing E.2: Example of the batch control file.

```
***********************************
*  Batch file for RSGEO
***********************************

...

[7  1.360  0.07  0.07  0           Spacing of wheel back,
    diff.to wheel base, profiles origin
* type 0, fixed sequence of the profiles
  0
* rail profile on the left side
  profil\uic60.prf
* rail profile on the right side
  =
* wheel profile on the left side
  profil\GV32_5.prf
```

```
* wheel profile on the right side
  =
@  new wheel, new rail
```

Step 3: Table manipulation

The four files generated using RSGEO contain raw data, from which the RSGEO tables can be obtained. The parameters needed are contained in the tables:

```
rsgTab_001_03R.dat
rsgTab_001_04R.dat
```

Several steps have to be taken in order to obtain the format that can be used by DYTSI

1. It's very common for multiple contact points to be present in a certain range of displacements, so each of the files can contain more than one table. Thus the files have to be split for each contact point. For example, if the contact point are two, then they will be separated in the raw tables by the header

   ```
   !  2 contact point
   ```

   The raw tables can then be split in the files

   ```
   Tab3-CP1.dat
   Tab4-CP1.dat
   Tab3-CP2.dat
   Tab4-CP2.dat
   ```

   that each describe one single point.

2. The content of the contact point raw tables have to be related to the parameters wanted by DYTSI (see table 4.1). In most of the cases simple geometrical transformations are needed. In some cases, such as for the Kalker's coefficients, their relation with the size of the contact ellipse has to be used. The MatLab files used for these transformations are listed in G.1, for the RSGEO version 6.10 (Build 20-07-99), and in G.2, for the RSGEO version 3.01 (Build 101 - 07-11-2010). The usage of such scripts will return a table for each contact point.

3. Understanding the way RSGEO works is very important in order to make order out of the data generated. RSGEO start the computation from one side of the discretization interval and takes step toward the end of the interval. For each displacement computes the parameters and then moves to the next displacement. When a second contact point pops up, RSGEO will clone the table generated so far and will start filling the two tables in parallel with different values. This means that the multiple tables in the raw files have equal values when there is a unique contact point and have different values when multiple contact points are present. Thus, when the two tables are split, values could be repeated and the usage of such table will be give very wrong results.
   DYTSI needs the tables to represent the exactly one point each. Whenever the load on a point fades away to disappear, the table for that contact point should consistently show that the patch is disappearing, like it was shown in figure 4.10. This can be obtained performing this kind of separation of the values by applying the MatLab script listed in G.3.

4. The manipulation of the RSGEO table is not over yet. Even if two tables have been constructed when different values in the raw tables were present, the contact points are granted to be separated. This means that there could be some entrances in one table that doesn't belong to the correct contact point. This can happen because RSGEO doesn't provide a point tracking facility during the generation of the raw data. Basically the program stores the values of two contact points in two different tables, without checking whether the values refers to one contact point or another. This makes the raw tables a little messy, but with some manual work they can be tidied up.
   The usage of plotting tools, like the RSGEO plotting script listed in G.4, can be used for plotting multiple tables (for each contact point) over each other. Usually sudden discontinuities can be observed whenever the contact points are wrongly switched among tables.

5. When the RSGEO tables for each contact point are obtained, they can be written in separate files, using the script `writeRSGEO` listed in G.5.

6. The files need to be processed by a script that attach them an header and concatenate the multiple tables. Each table in the final file need to be preceded by the header:

```
#
"rh(51)" "Normalforce"  "DATAN(TANDJA(1,1))" "a(1,1)" "b(1,1)" "BJA(2,1,1)+0,75" "ROJ(1,1)"  "c11" "c22" "c23" "BJA(3,1,1)"  "Penetration"  "BJA(2,1,1)+0.75+rh(51)"
```

and followed by an empty row.
This step can be done by adding and pasting the proper parts together or using the bash script listed in G.6.

The file is now ready to be used by DYTSI.

Appendix F

Figures

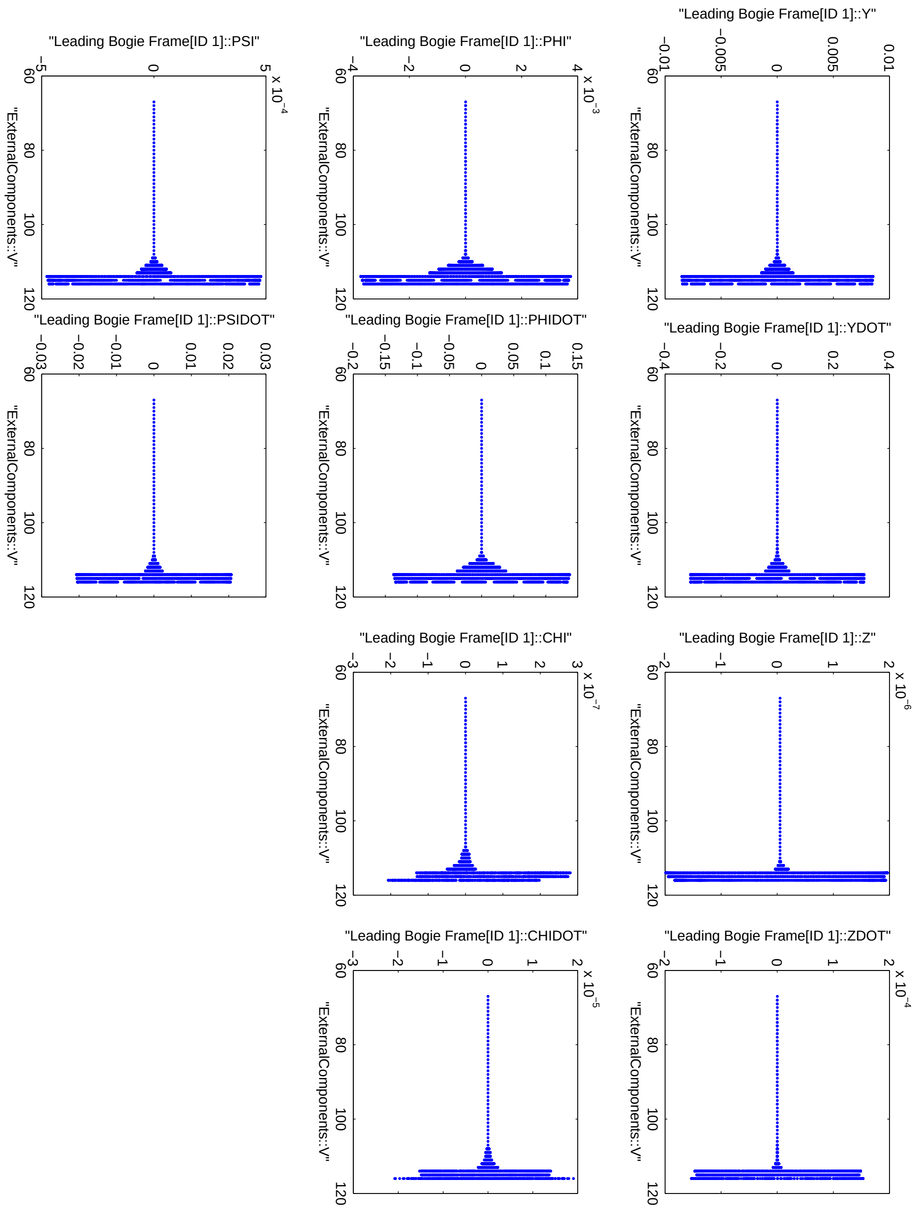


Figure F.1: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the bogie frame at increasing speeds.

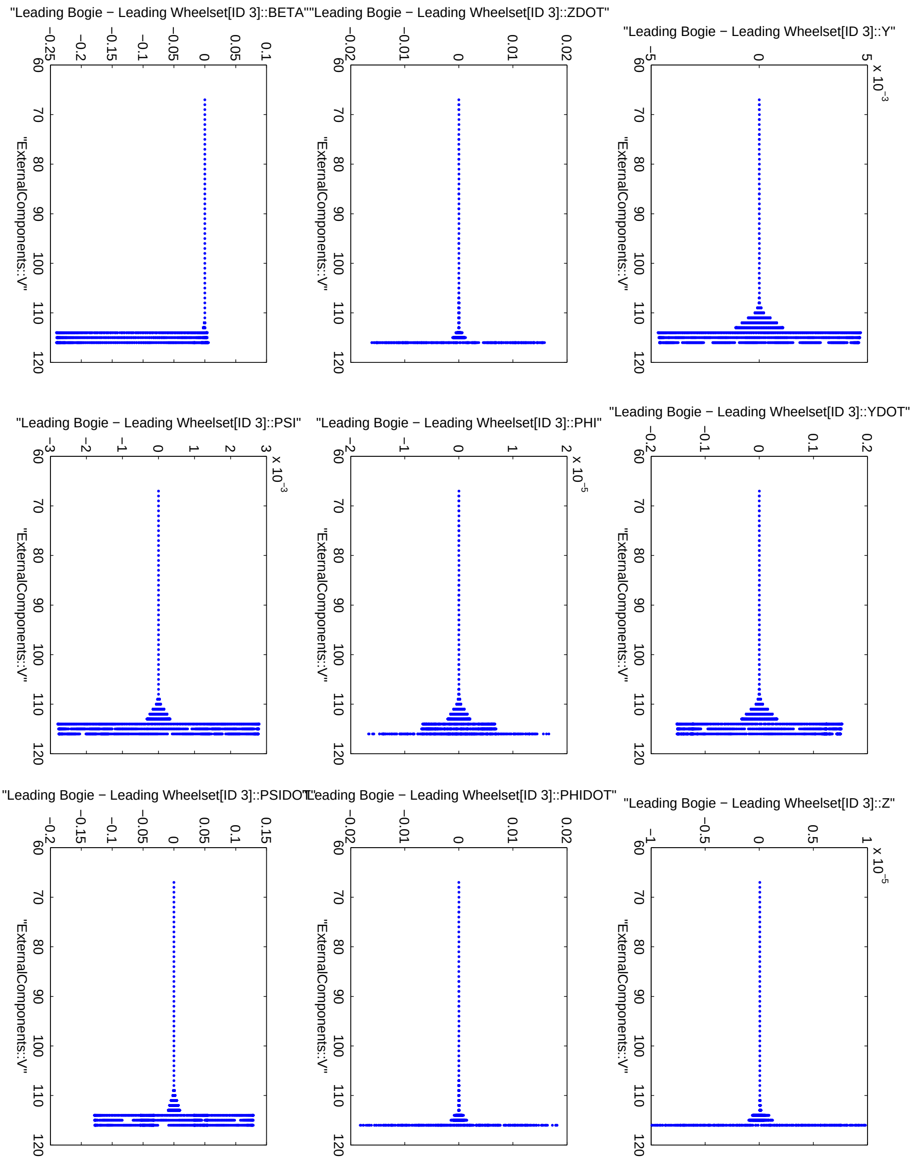


Figure F.2: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the leading wheel set at increasing speeds.

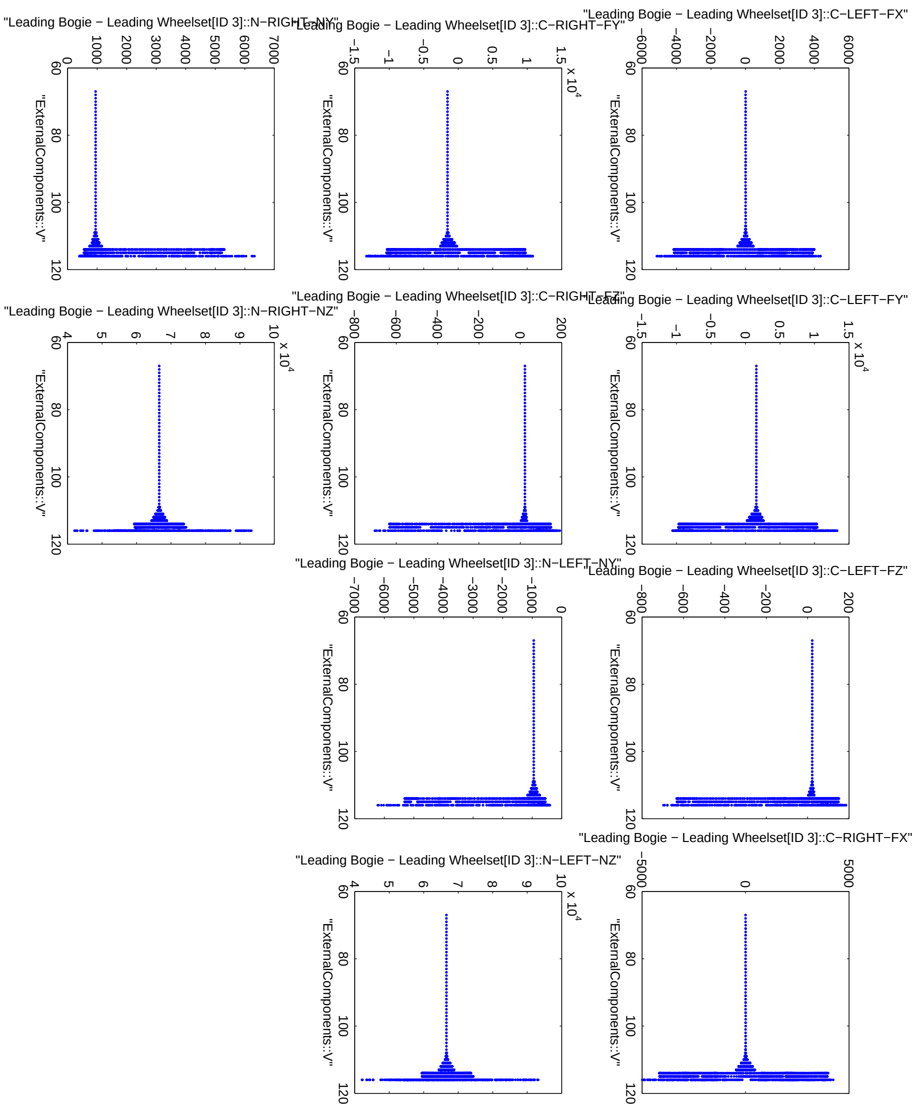


Figure F.3: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Creep and normal forces acting on the leading wheel set at increasing speeds.

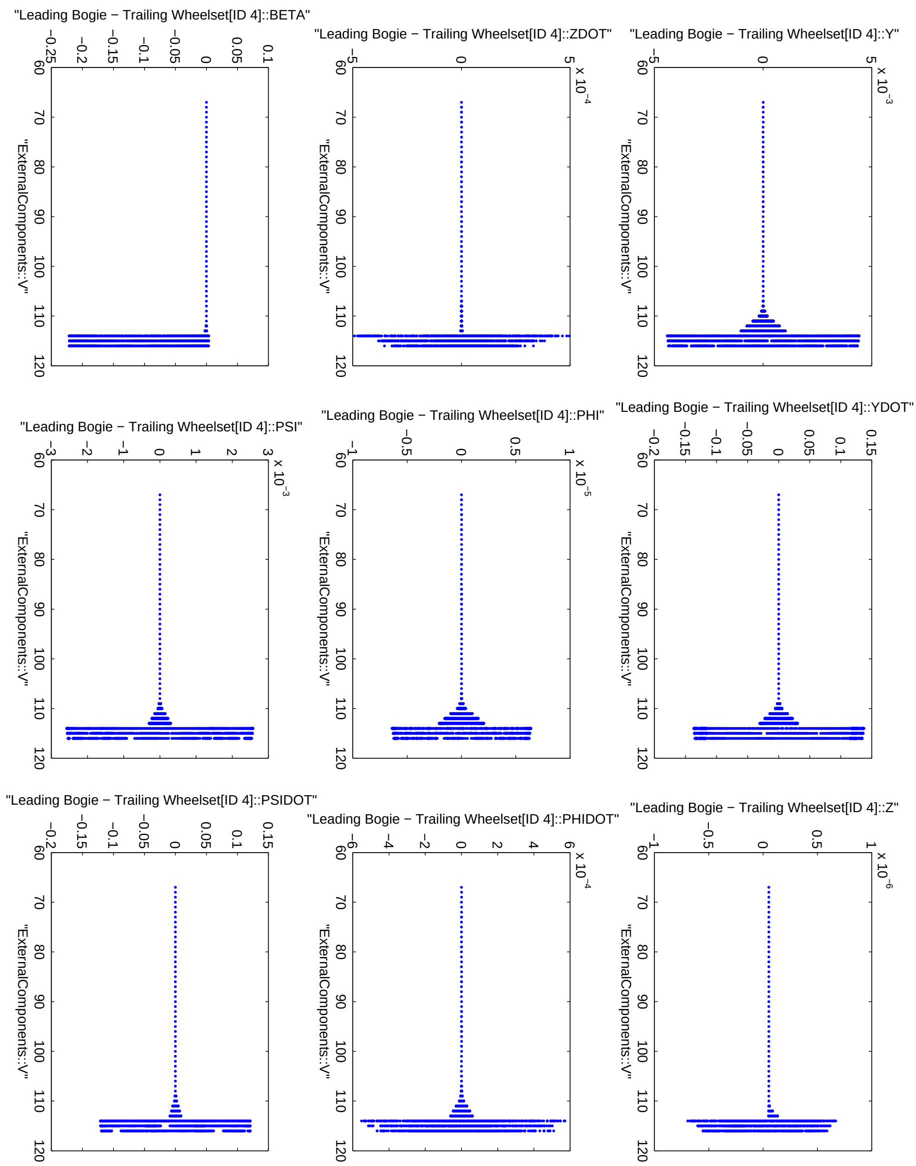


Figure F.4: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the trailing wheel set at increasing speeds.

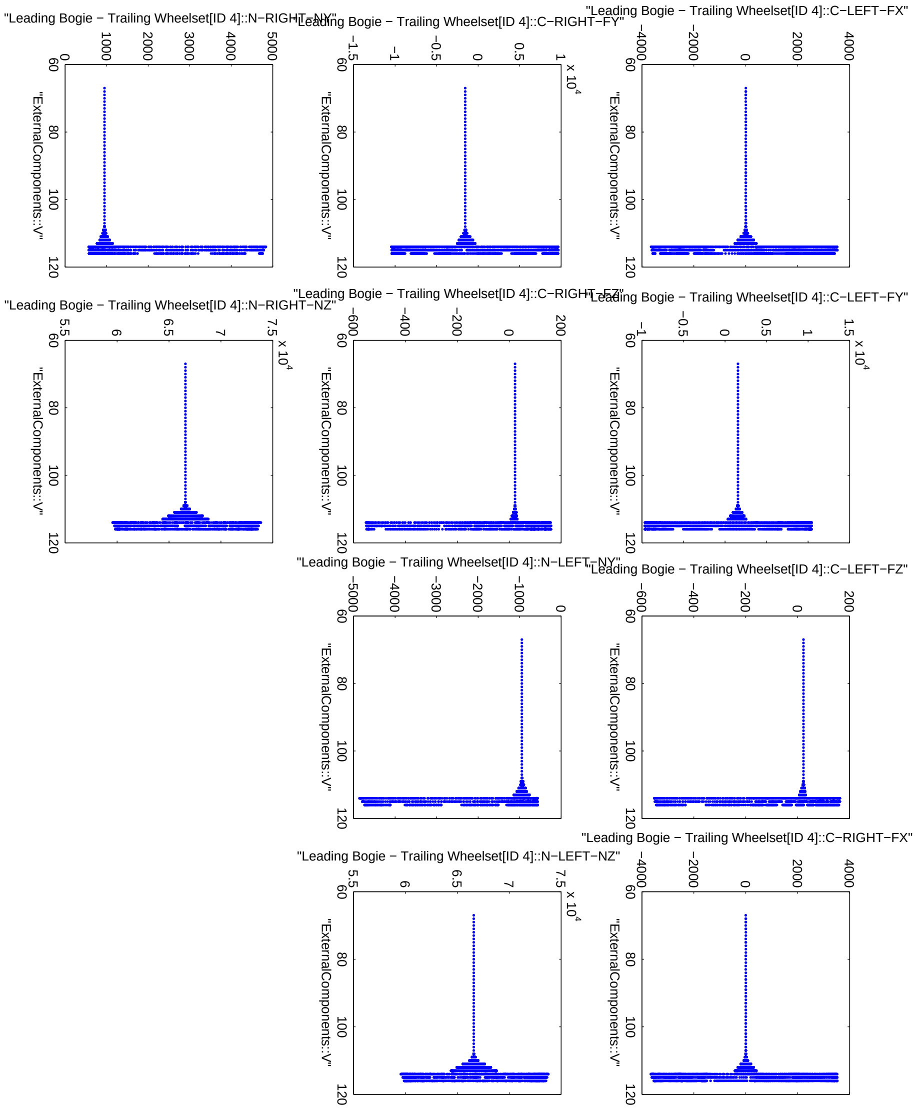


Figure F.5: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Creep and normal forces acting on the trailing wheel set at increasing speeds.

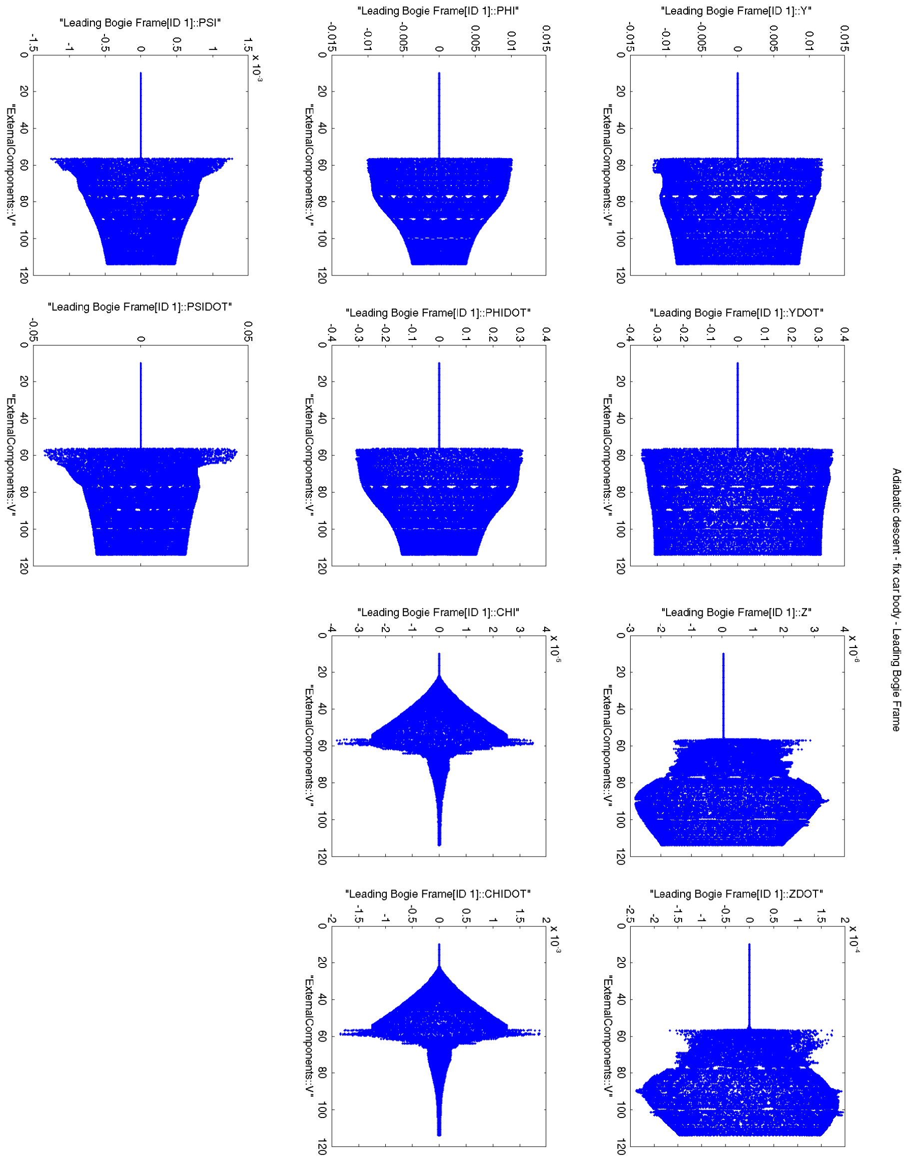


Figure F.6: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the bogie frame at decreasing speeds starting in the hunting limit cycle.

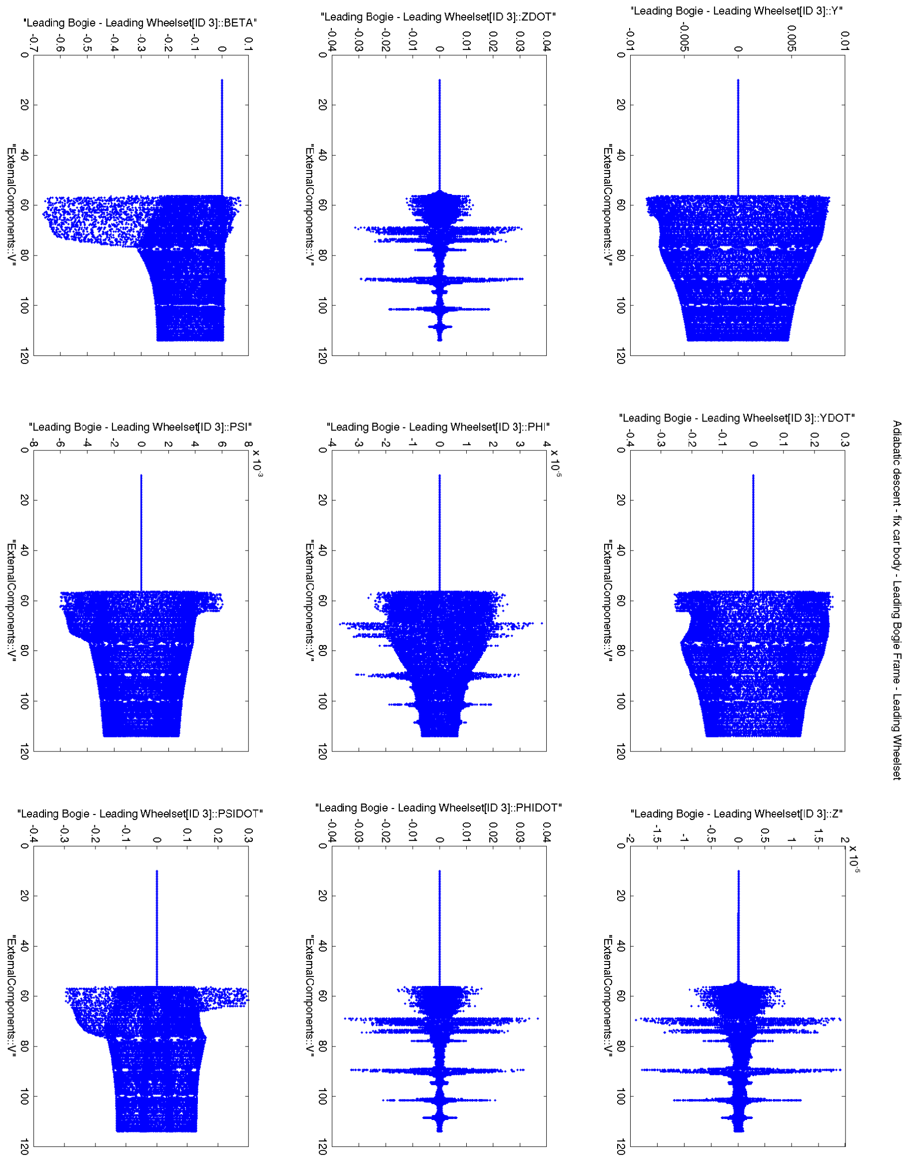


Figure F.7: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the leading wheel set at decreasing speeds starting in the hunting limit cycle.

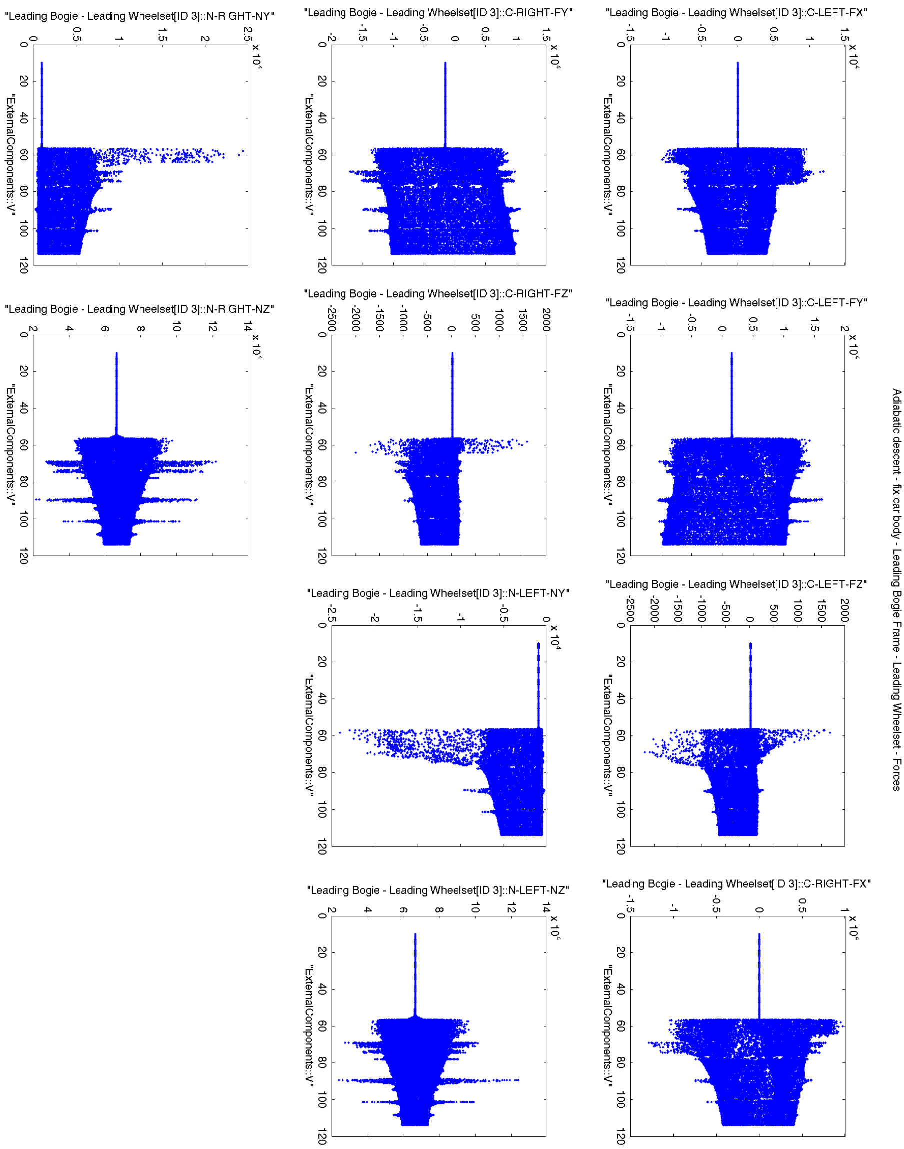


Figure F.8: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Creep and normal forces acting on the leading wheel set at decreasing speeds starting in the hunting limit cycle.

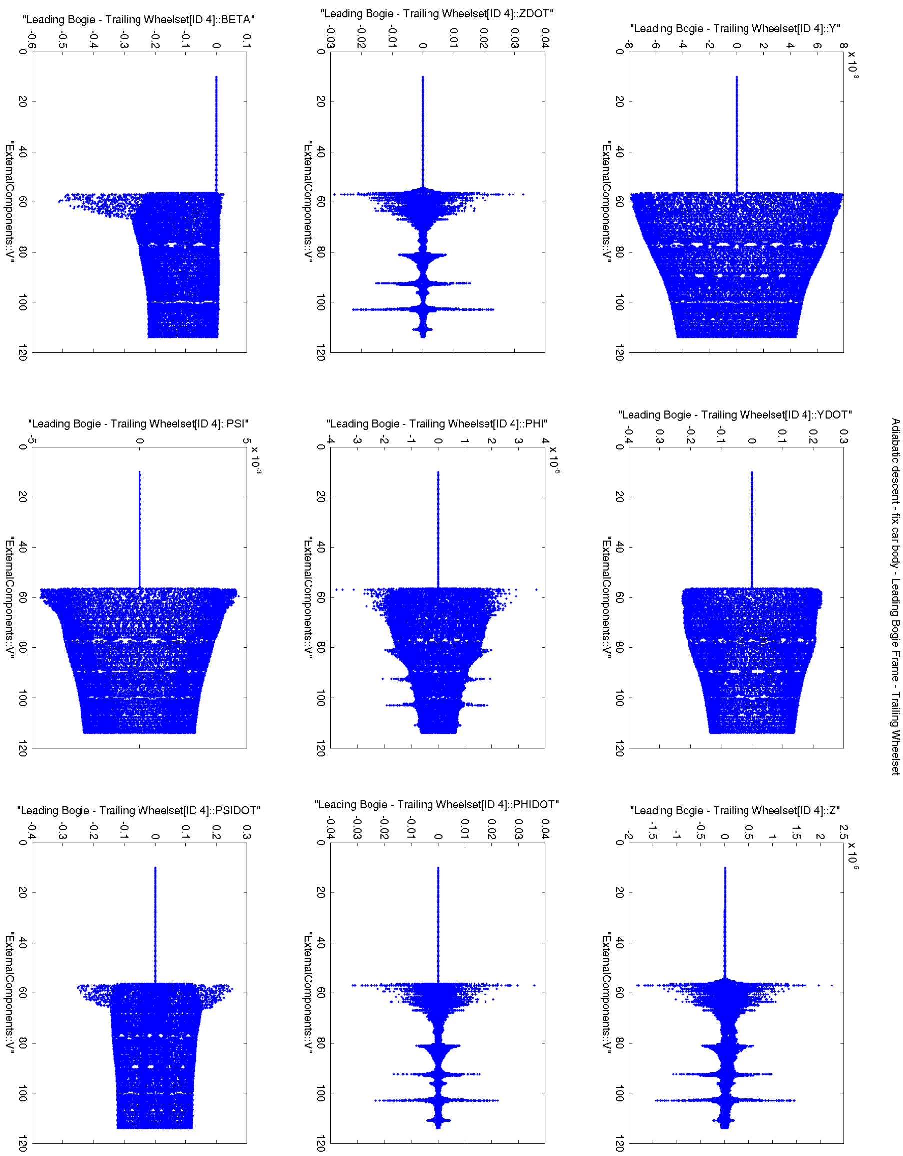


Figure F.9: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the trailing wheel set at decreasing speeds starting in the hunting limit cycle.

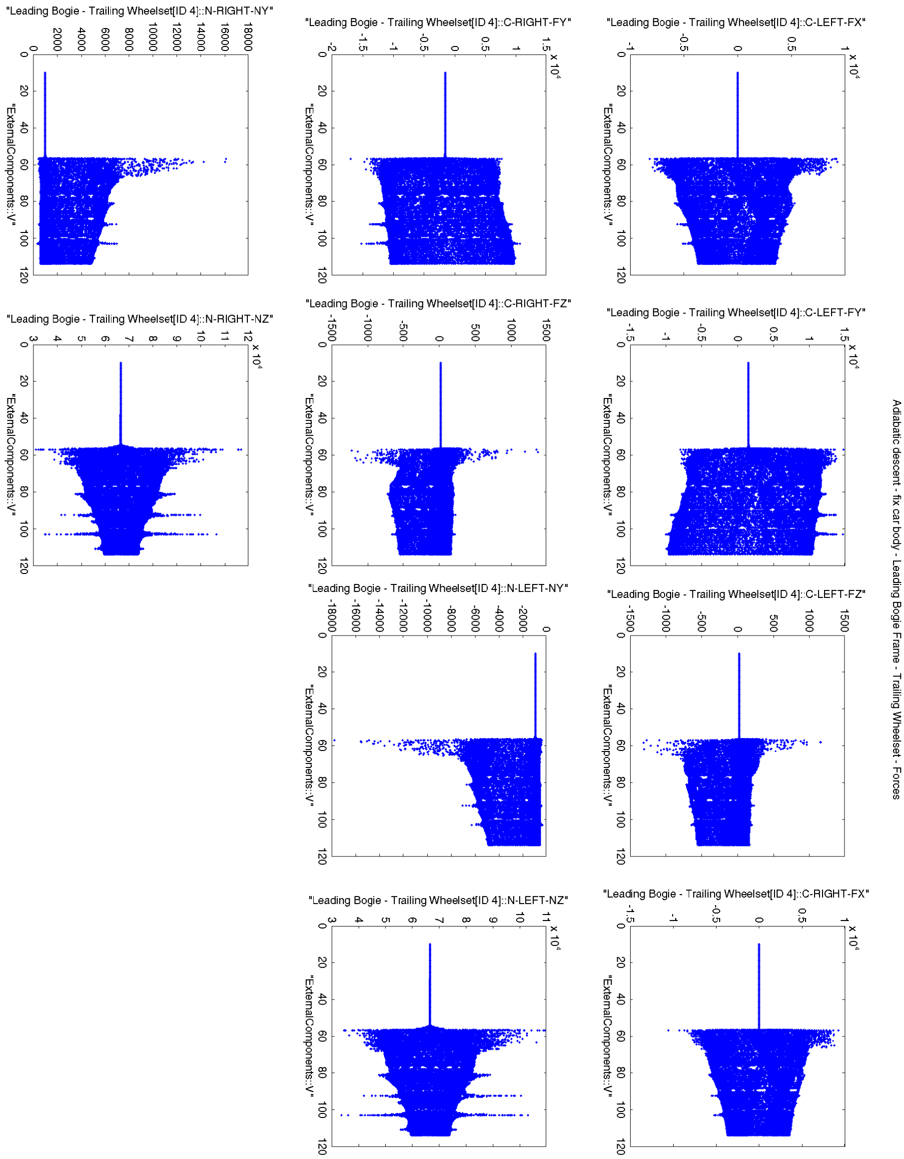


Figure F.10: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Creep and normal forces acting on the trailing wheel set at decreasing speeds starting in the hunting limit cycle.

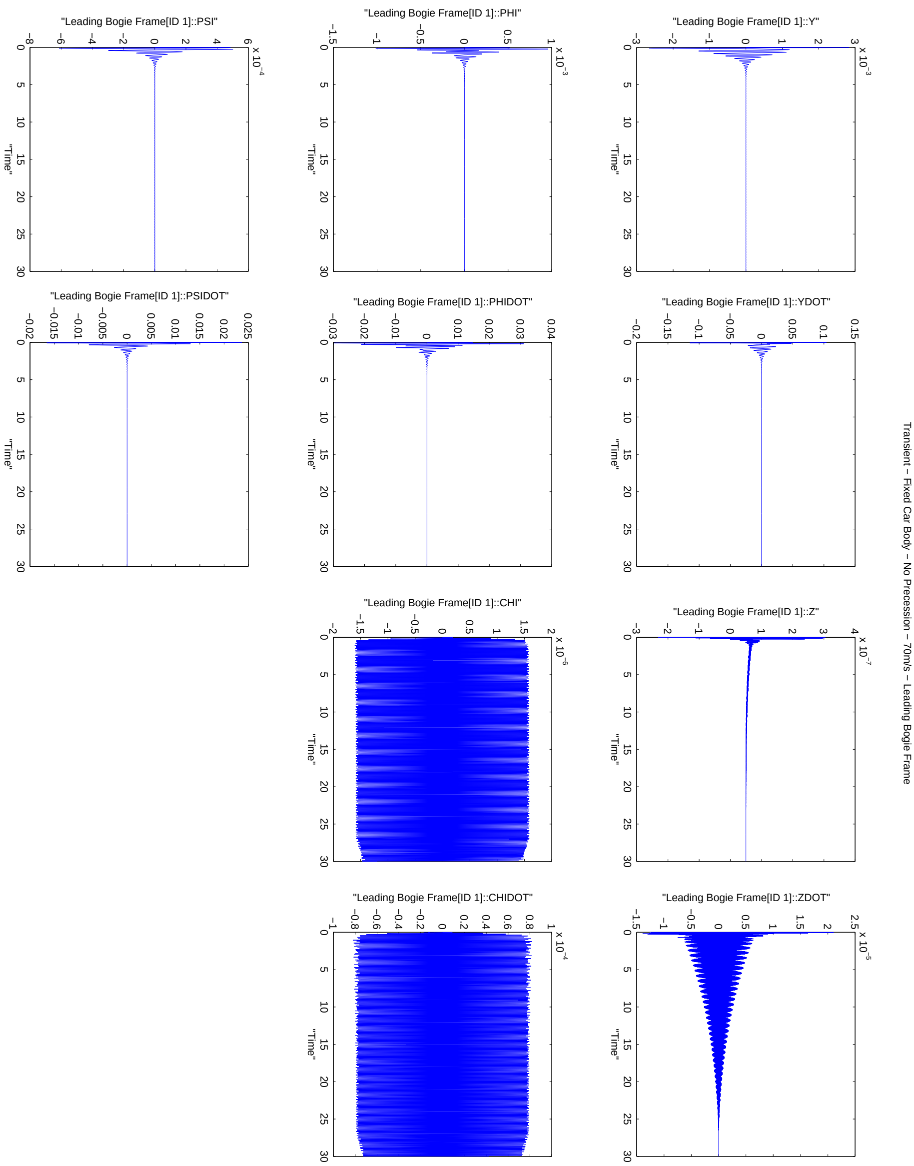


Figure F.11: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the bogie frame when the leading wheel set of the system is strongly perturbed in the effort of entering the hunting limit cycle at 70m/s. The system gets back to the stable solution in spite of this perturbation.

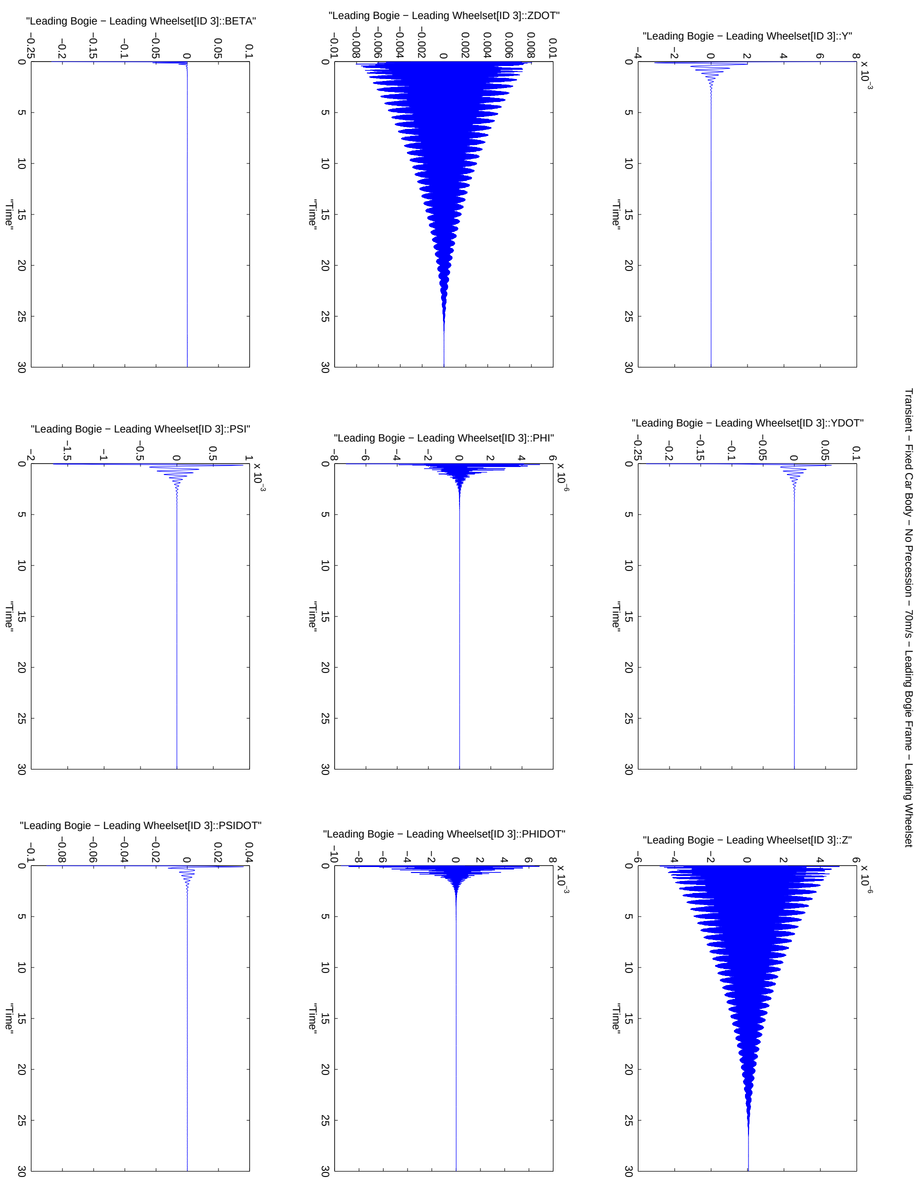


Figure F.12: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the leading wheel set when it is strongly perturbed in the effort of entering the hunting limit cycle at 70m/s. The system gets back to the stable solution in spite of this perturbation.

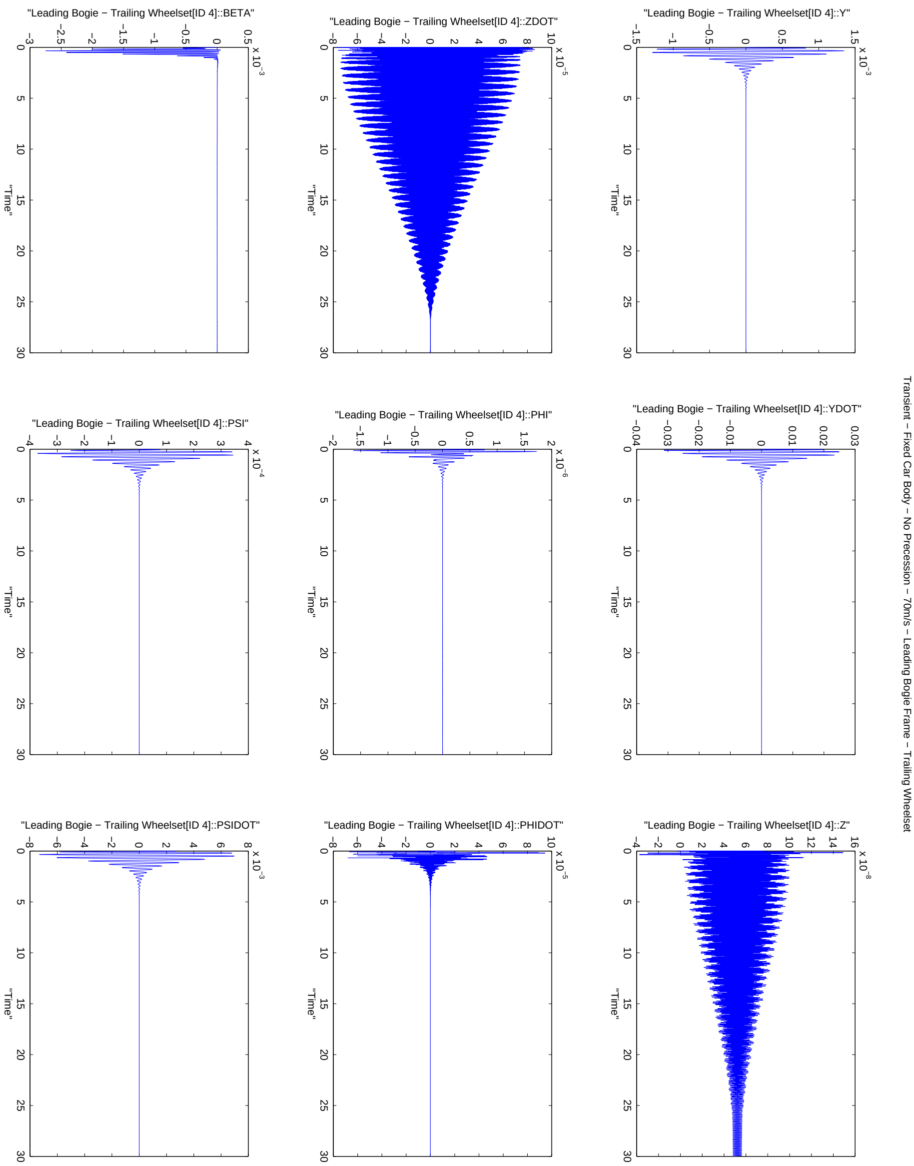


Figure F.13: Cooperrider model: modified version with fixed car body (model used in [Chr01, HP02, Big10] with neglected precession forces). Dynamics of the trailing wheel set when the leading wheel set of the system is strongly perturbed in the effort of entering the hunting limit cycle at 70m/s. The system gets back to the stable solution in spite of this perturbation.

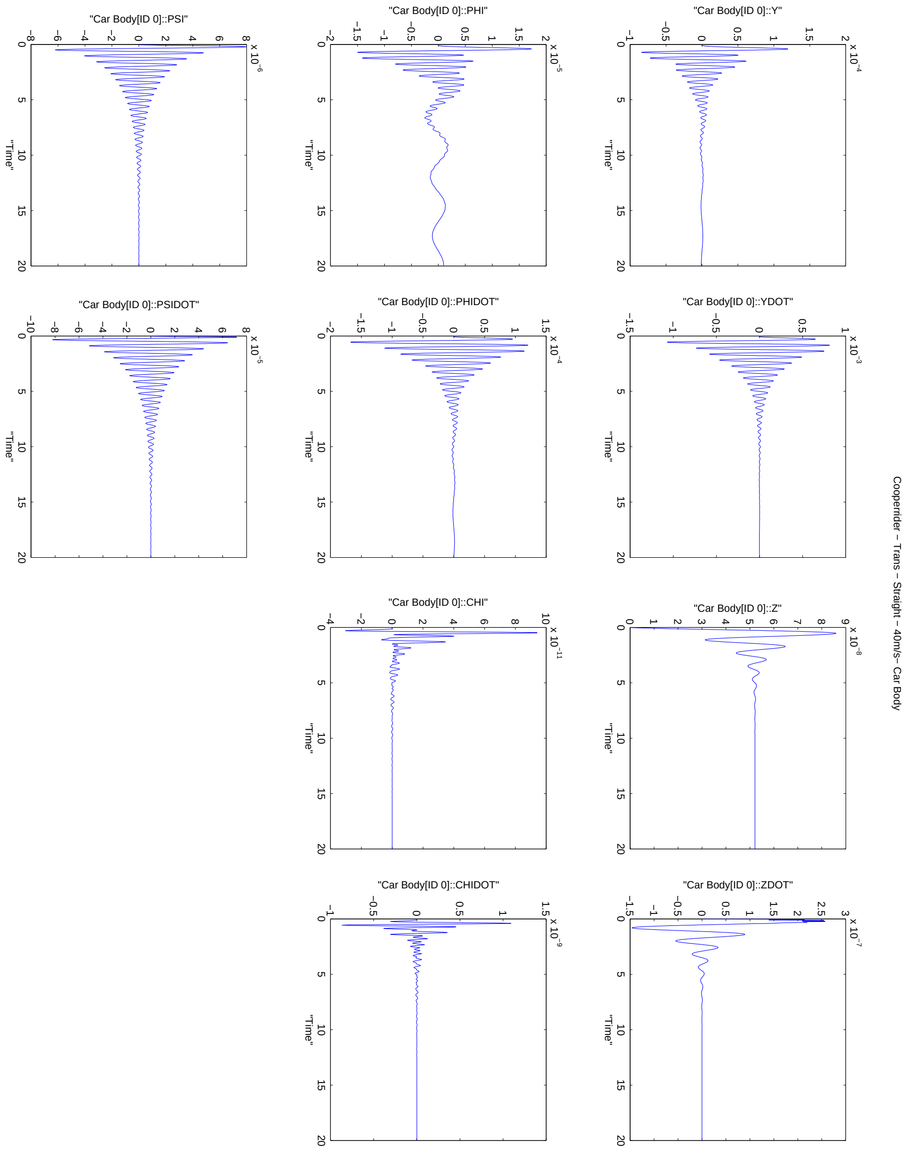


Figure F.14: Cooperrider model: Dynamics of the car body running at 40m/s on straight track. The model doesn't consider precession forces.

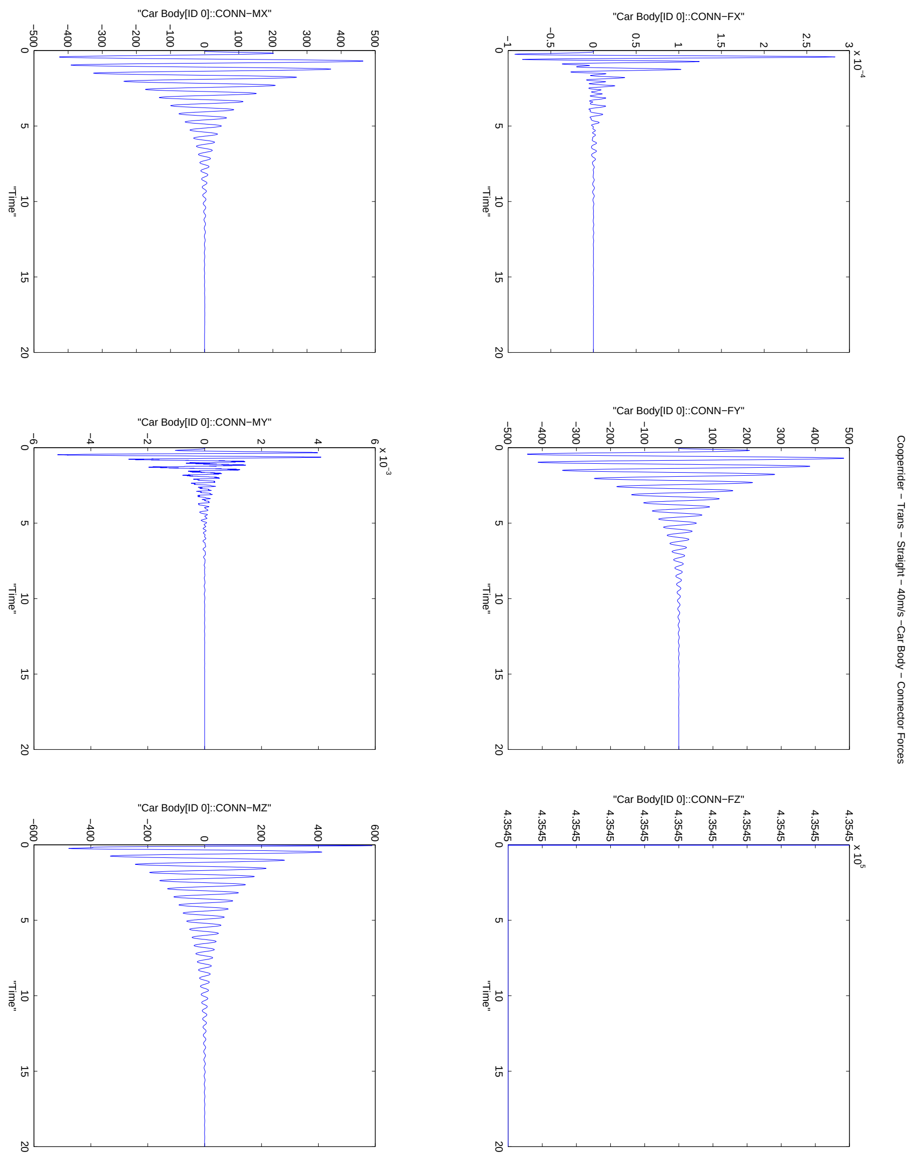


Figure F.15: Cooperrider model: Suspension forces acting on the car body running at 40m/s on straight track. The model doesn't consider precession forces.

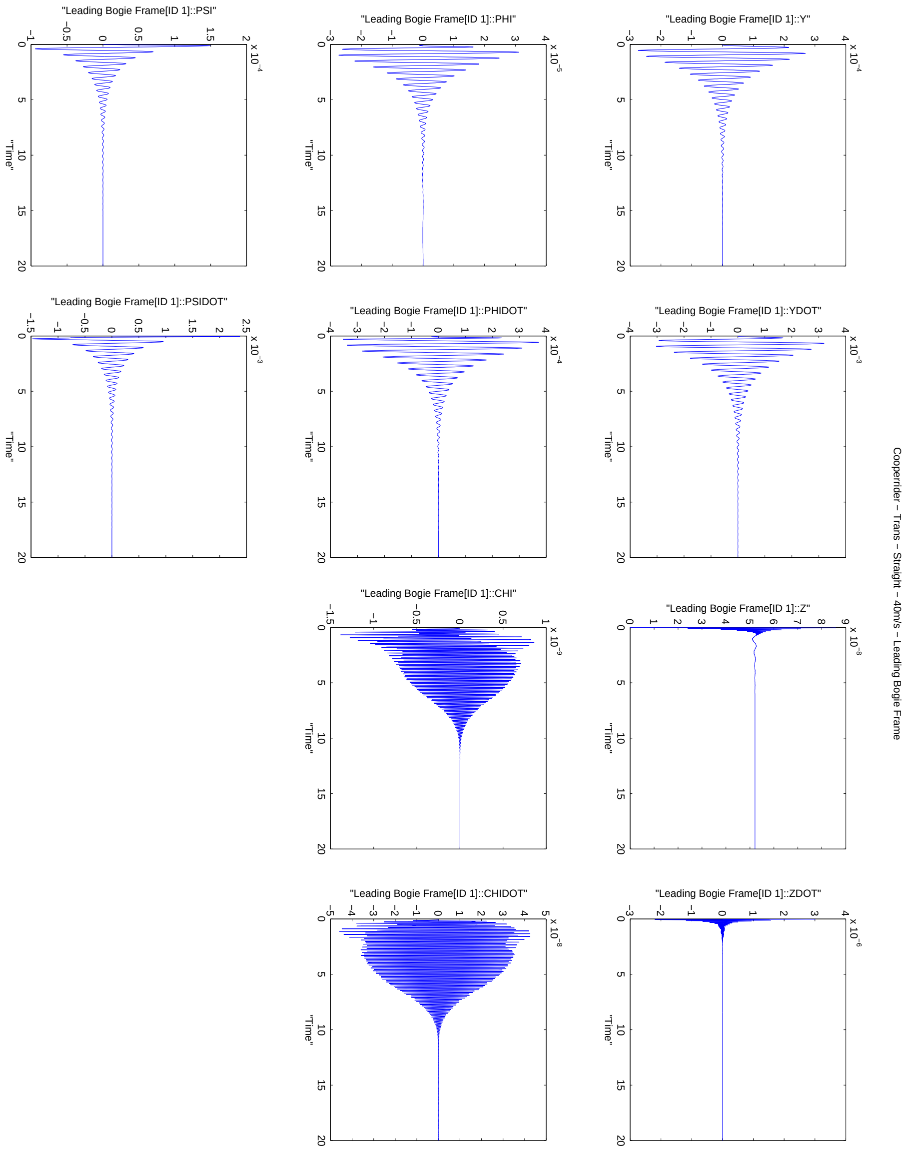


Figure F.16: Cooperrider model: Dynamics of the leading bogie frame running at 40m/s on straight track. The model doesn't consider precession forces.

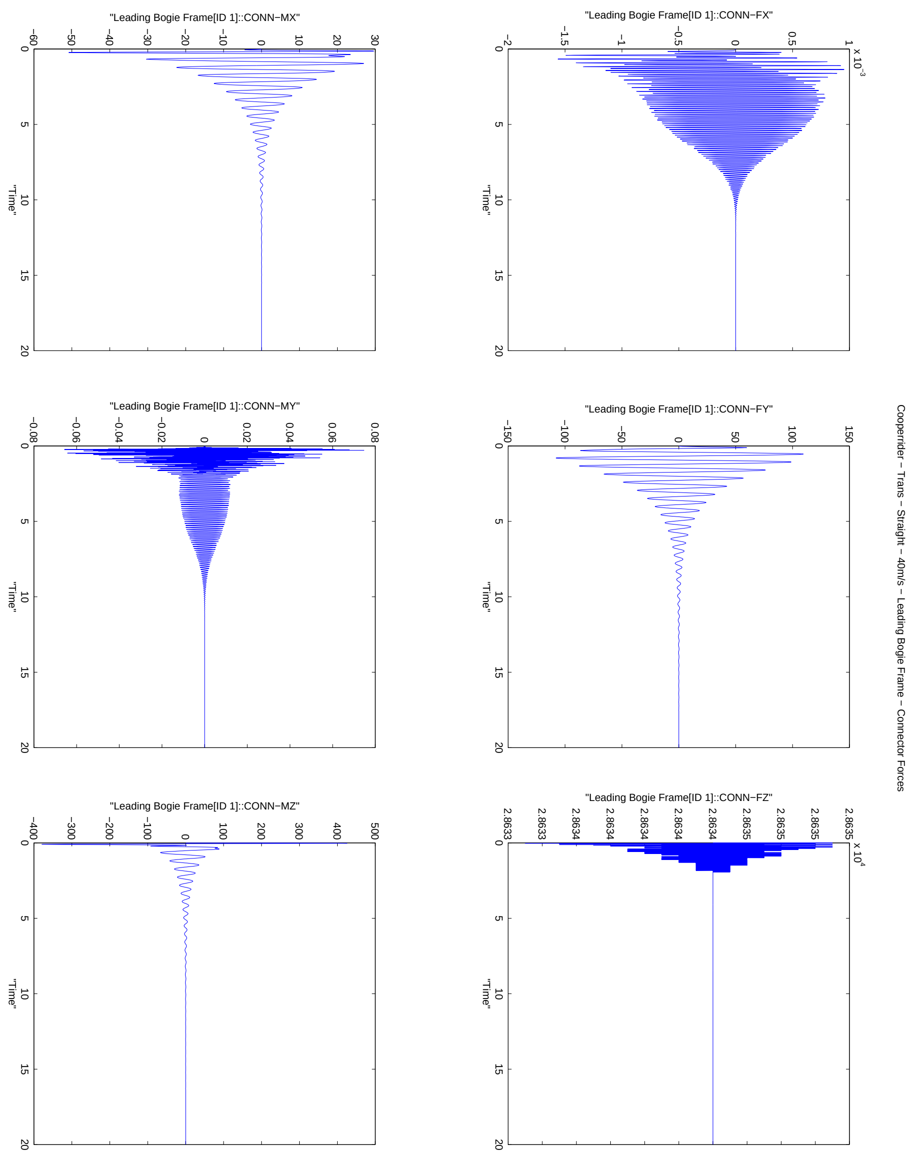


Figure F.17: Cooperrider model: Suspension forces acting on the leading bogie frame running at 40m/s on straight track. The model doesn't consider precession forces.

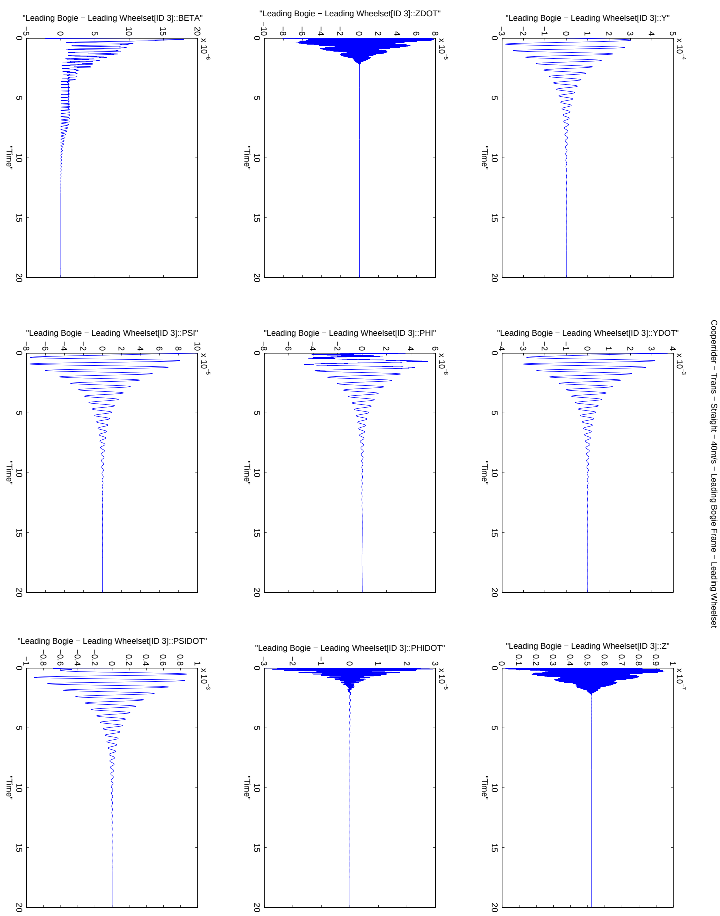


Figure F.18: Cooperrider model: Dynamics of the leading wheel set mounted on the leading bogie frame running at 40m/s on straight track. The model doesn't consider precession forces.

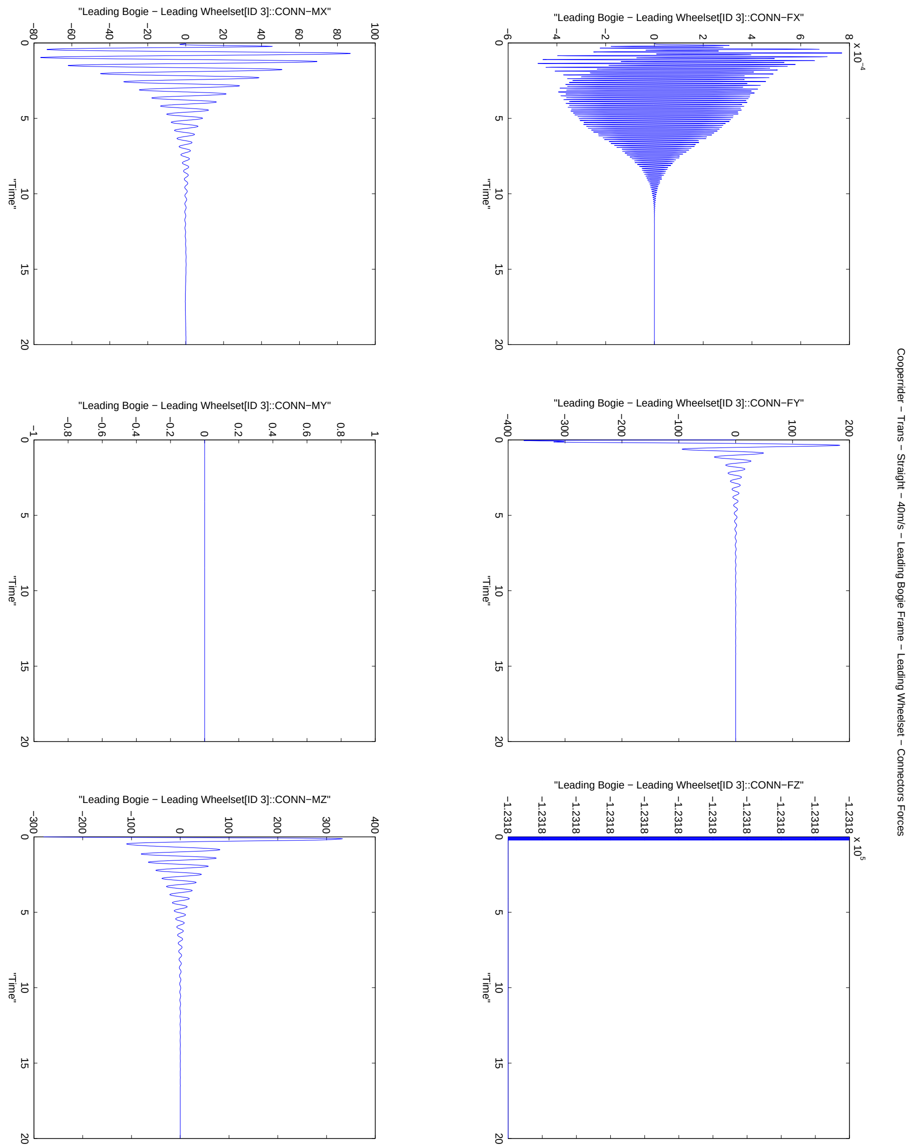


Figure F.19: Cooperrider model: Suspension forces acting on the leading wheel set mounted on the leading bogie frame running at 40m/s on straight track. The model doesn't consider precession forces.

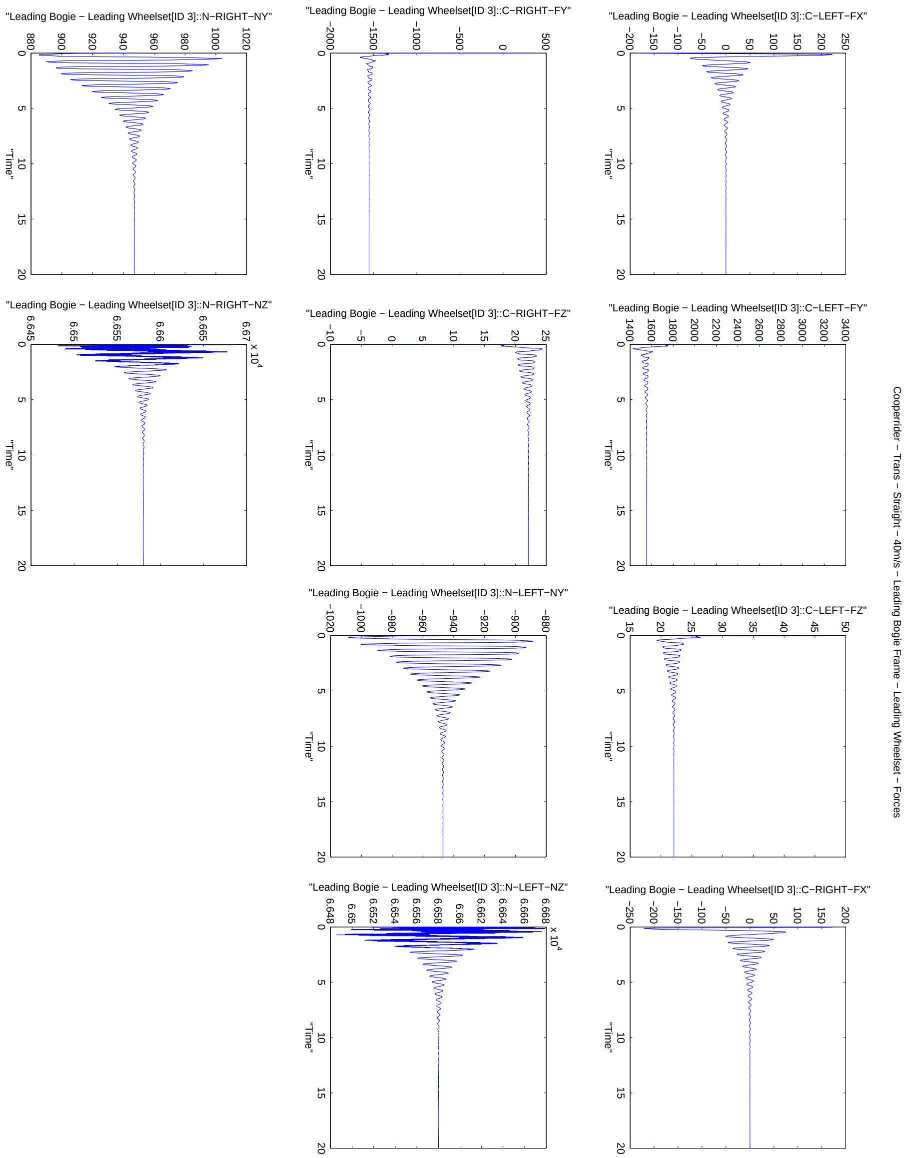


Figure F.20: Cooperrider model: Contact forces acting on the leading wheel set mounted on the leading bogie frame running at 40m/s on straight track. The model doesn't consider precession forces.

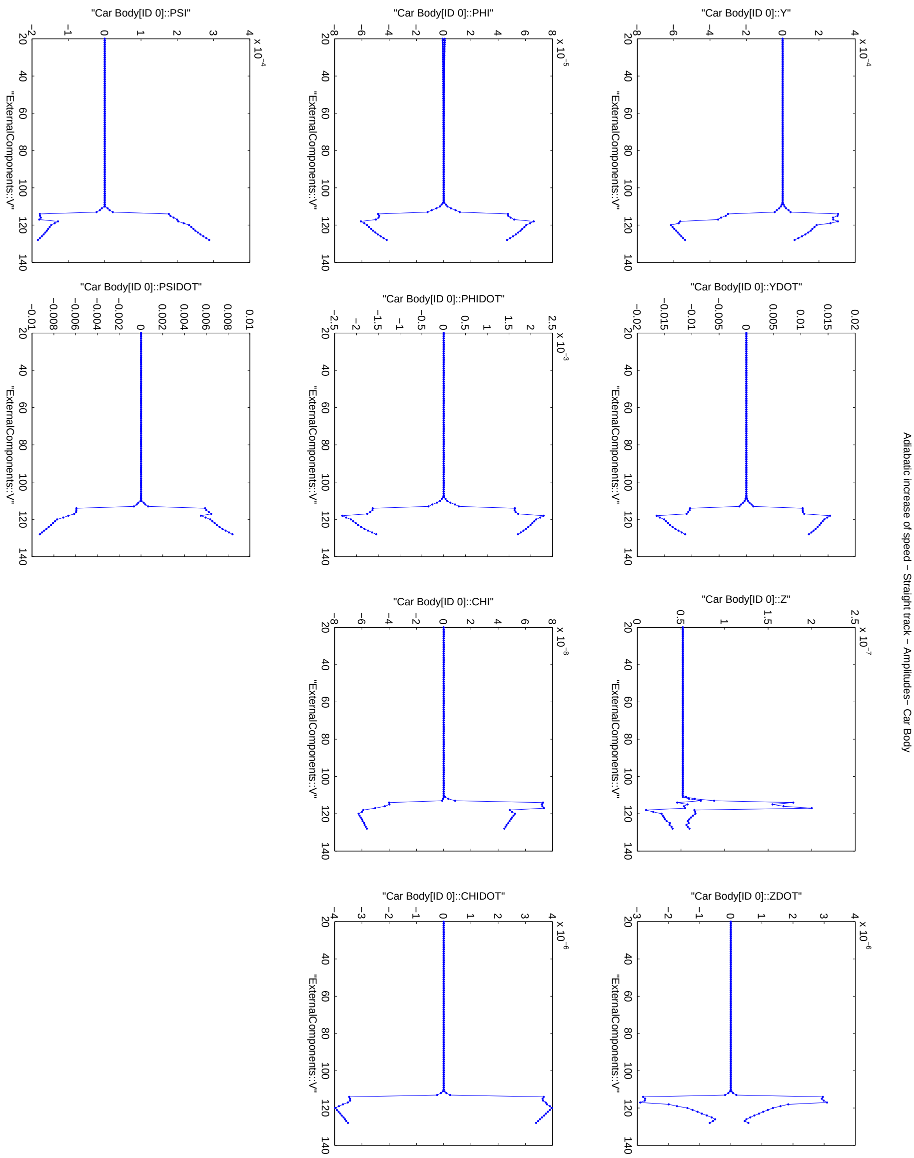


Figure F.21: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the car body with respect to the increasing speed. The model is running on straight track and does not consider precession forces. The hunting limit cycle is entered when the model reach the speed of $110\frac{m}{s}$.

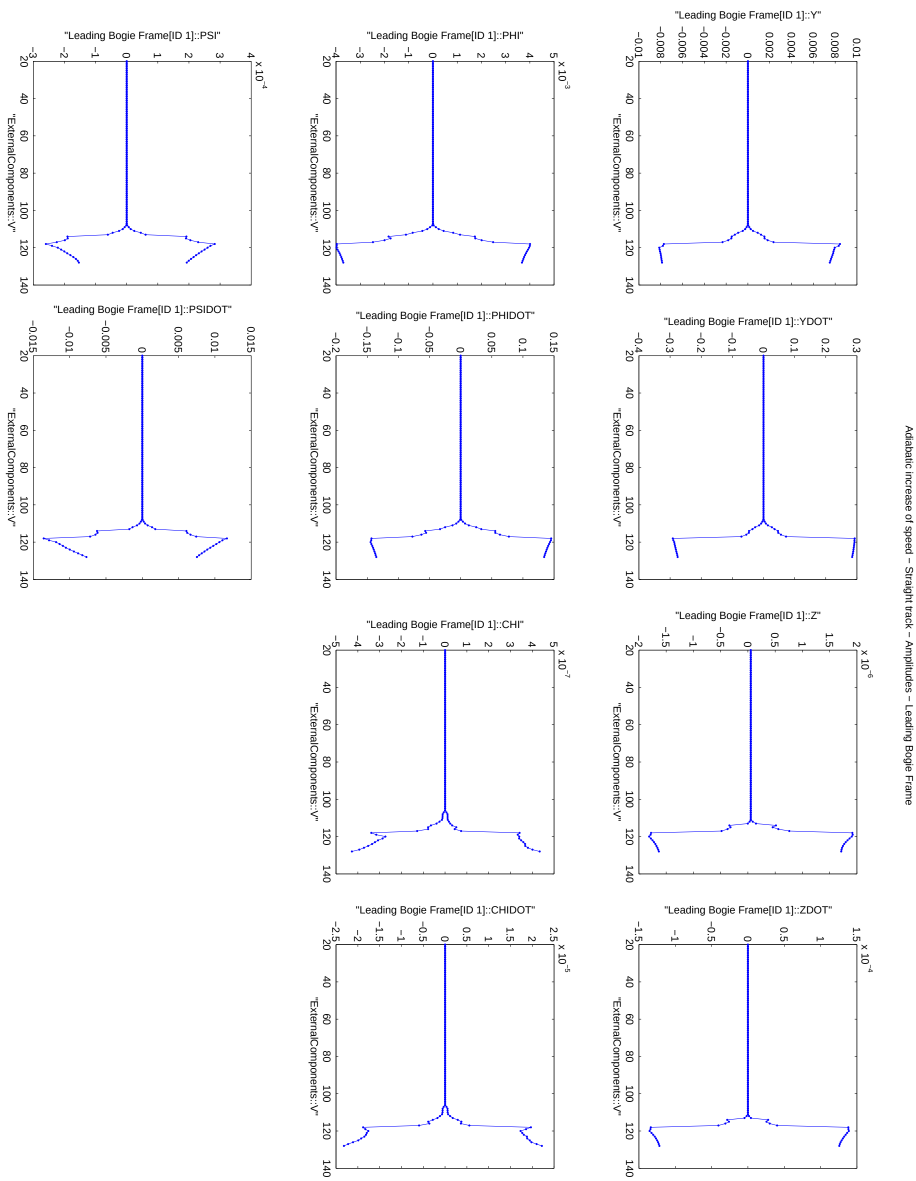


Figure F.22: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on straight track and does not consider precession forces. The hunting limit cycle is entered when the model reach the speed of $110\frac{m}{s}$.

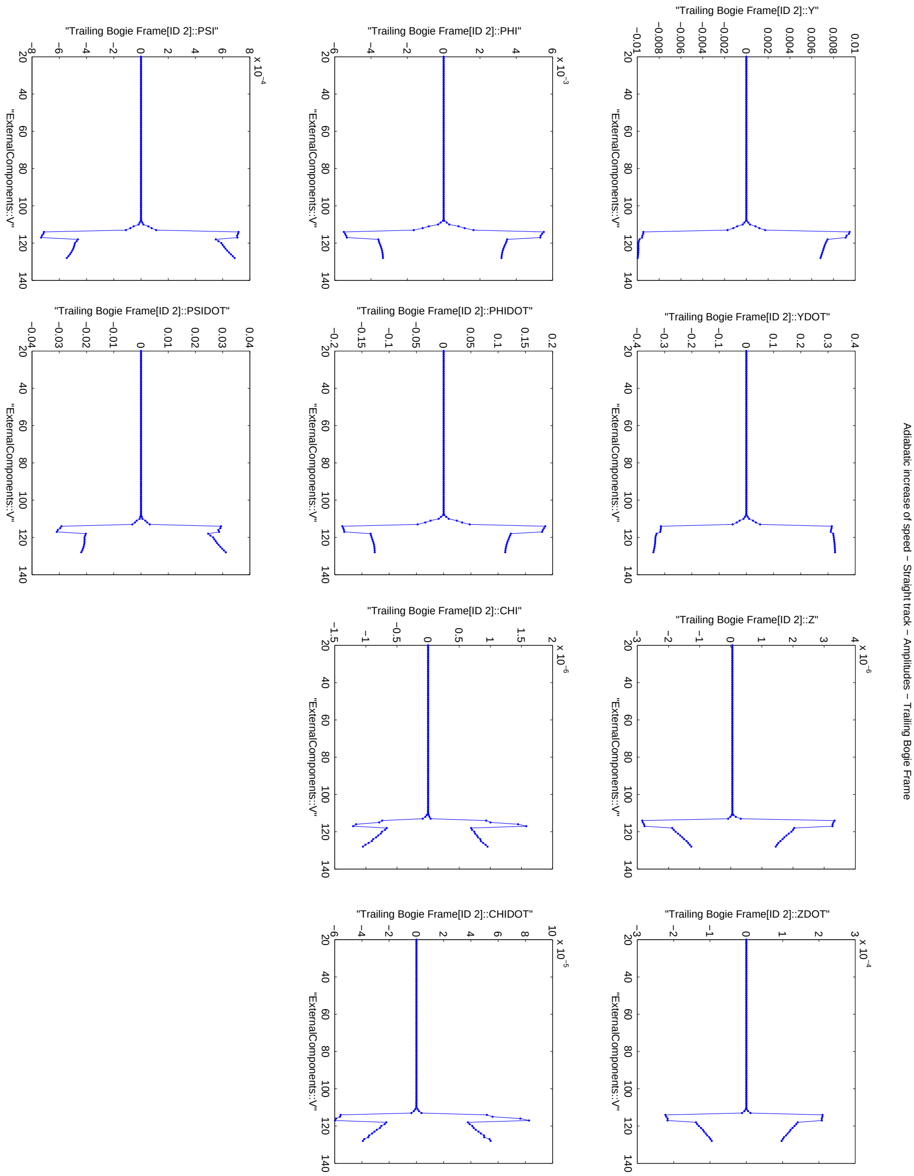


Figure F.23: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on straight track and does not consider precession forces. The hunting limit cycle is entered when the model reach the speed of $110\frac{m}{s}$.

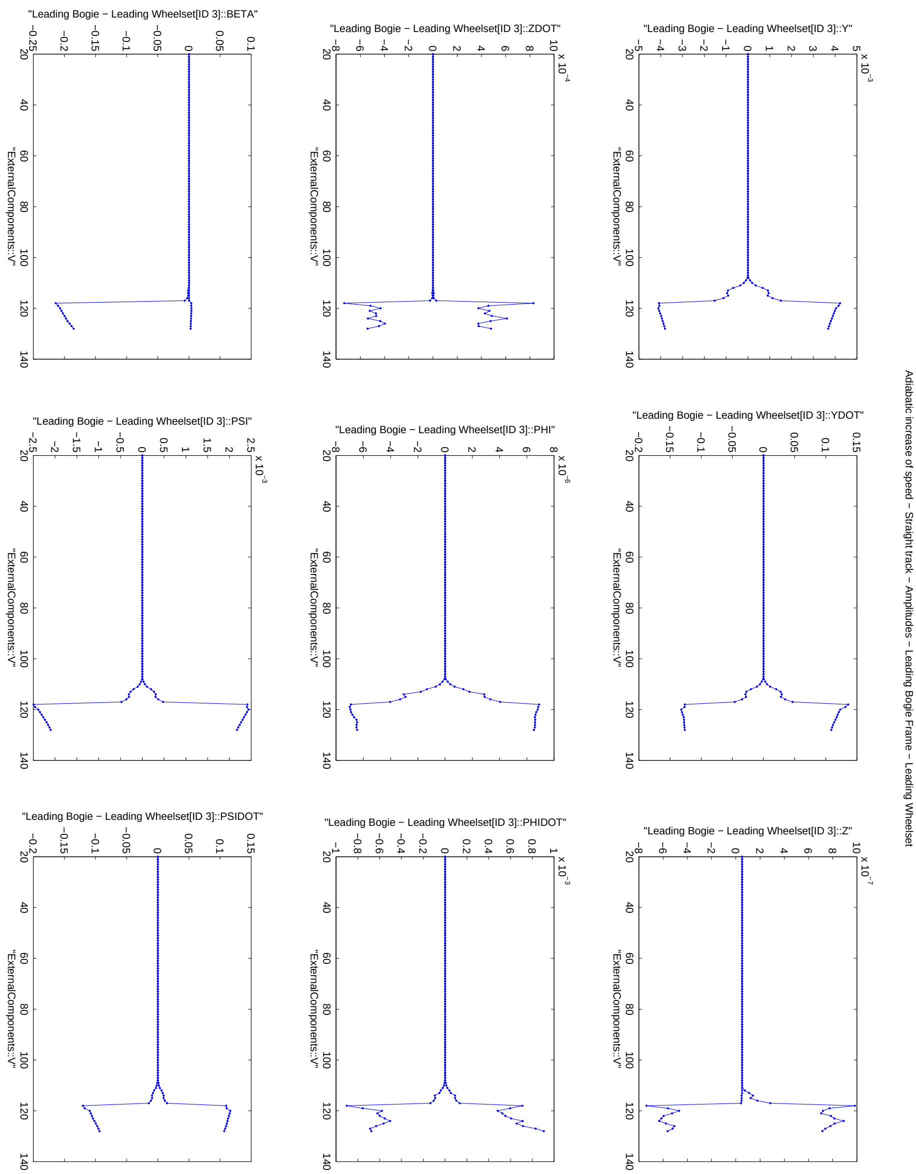


Figure F.24: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the leading wheel set attached to the leading bogie frame frame with respect to the increasing speed. The model is running on straight track and does not consider precession forces. The hunting limit cycle is entered when the model reach the speed of $110\frac{m}{s}$.

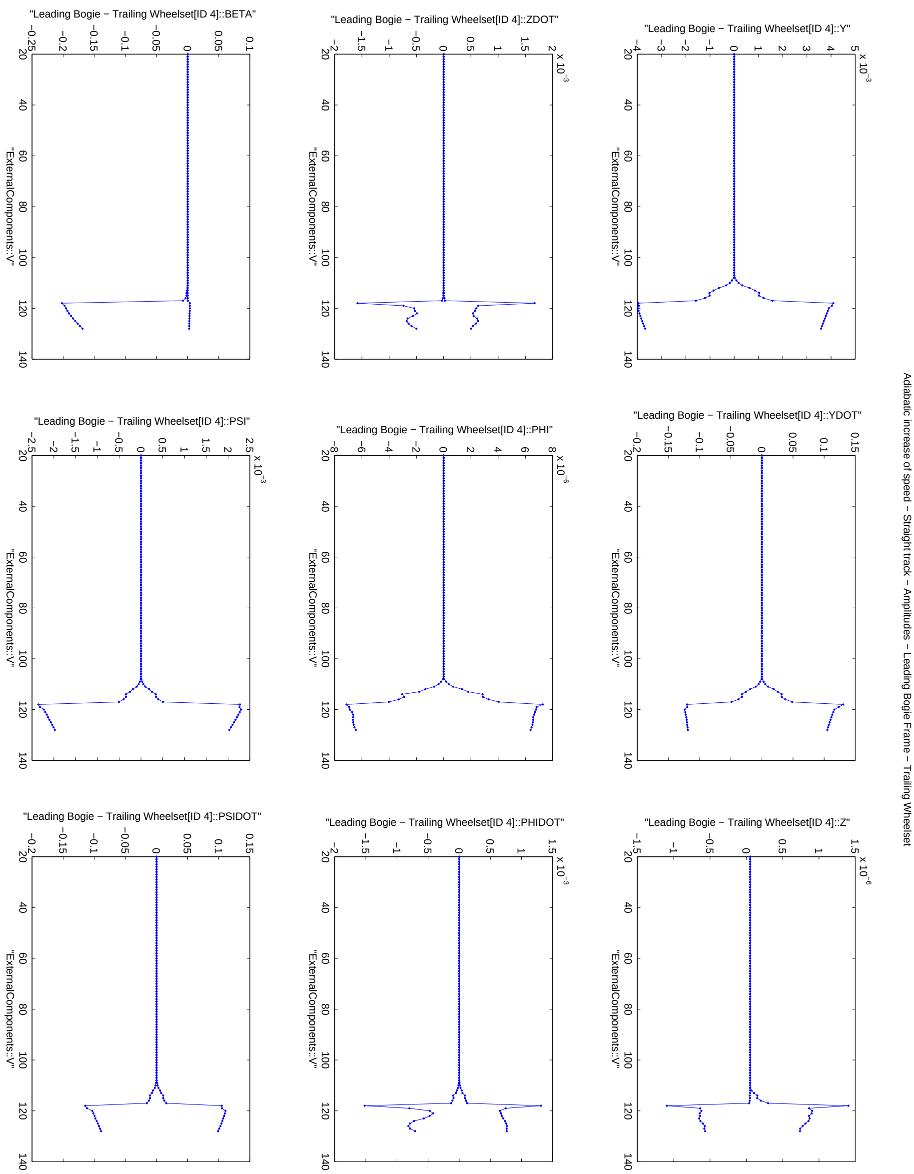


Figure F.25: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the trailing wheel set attached to the leading bogie frame frame with respect to the increasing speed. The model is running on straight track and does not consider precession forces. The hunting limit cycle is entered when the model reach the speed of $110\,\frac{m}{s}$.

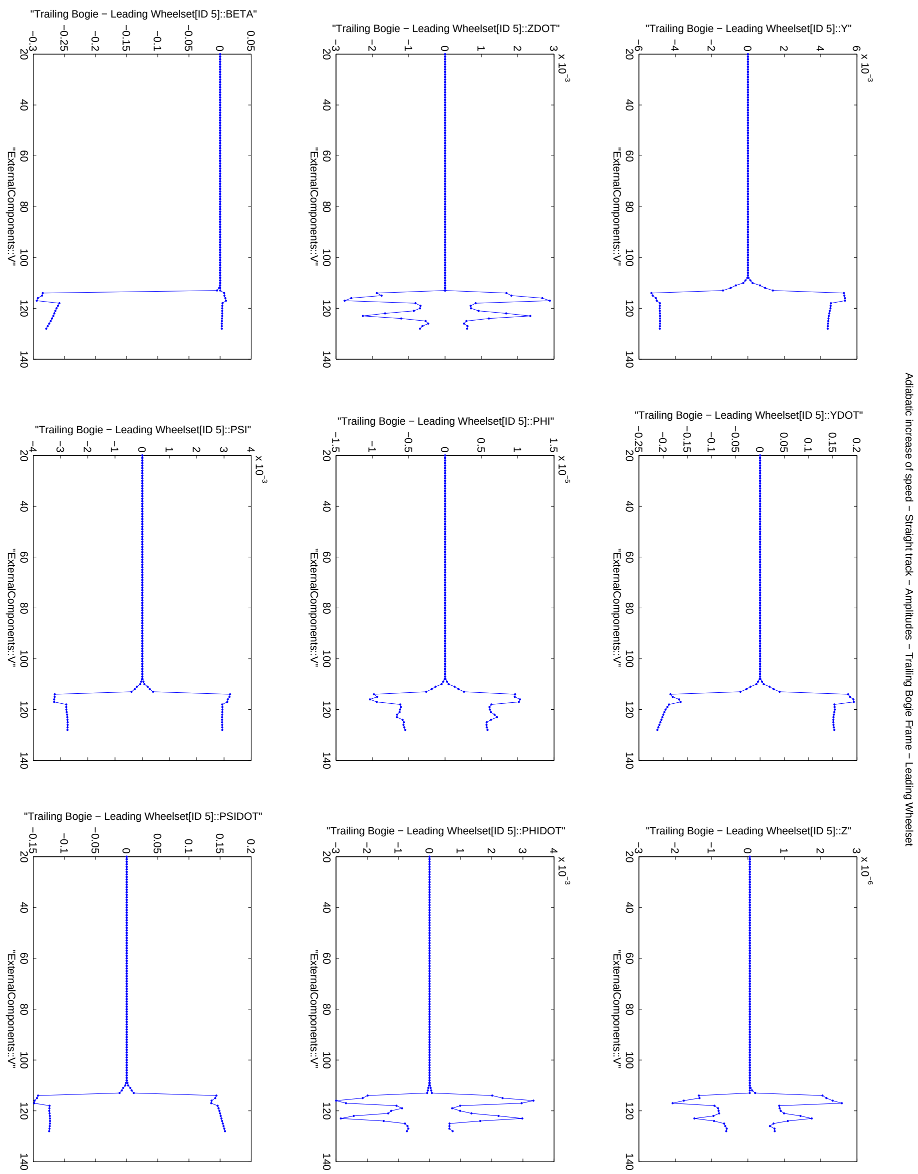


Figure F.26: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the leading wheel set attached to the trailing bogie frame frame with respect to the increasing speed. The model is running on straight track and does not consider precession forces. The hunting limit cycle is entered when the model reach the speed of $110\frac{m}{s}$.

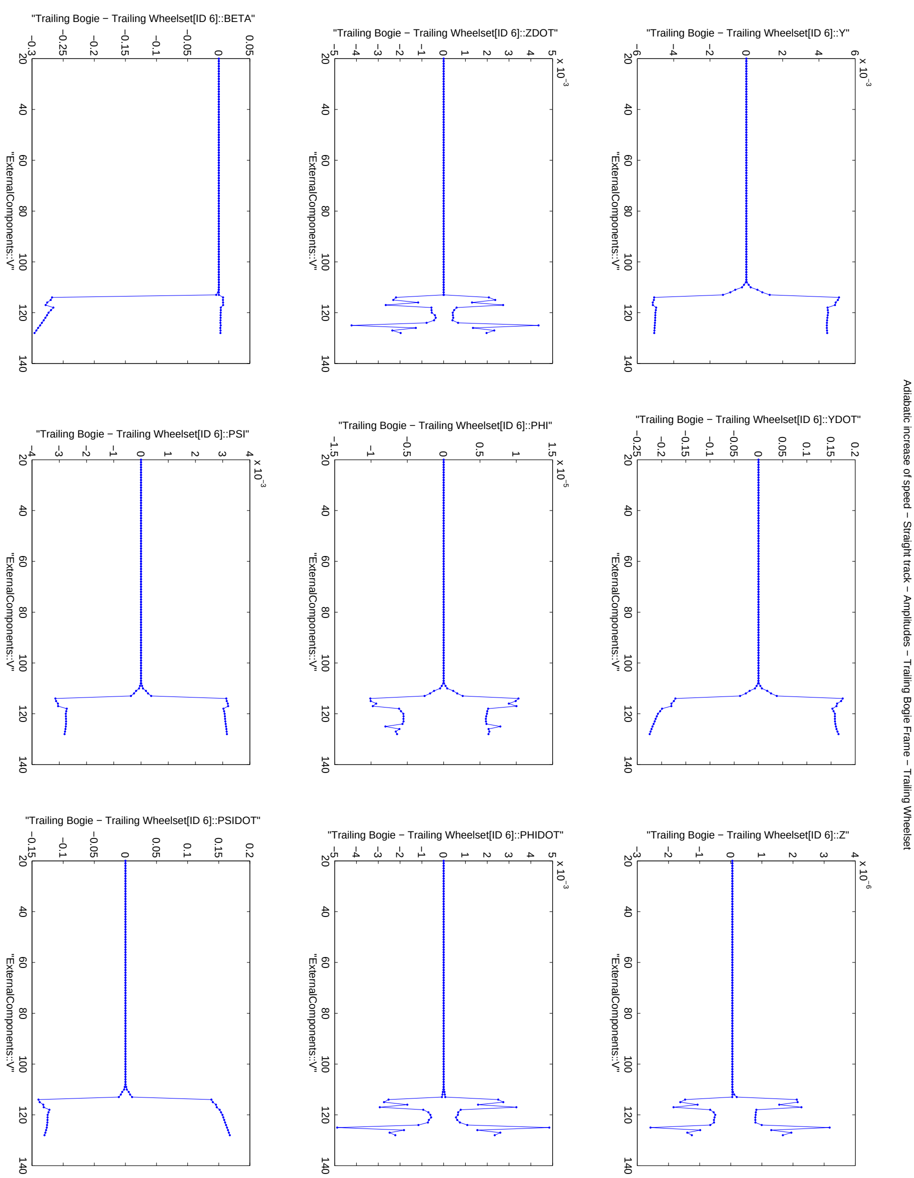


Figure F.27: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame frame with respect to the increasing speed. The model is running on straight track and does not consider precession forces. The hunting limit cycle is entered when the model reach the speed of $110\frac{m}{s}$.

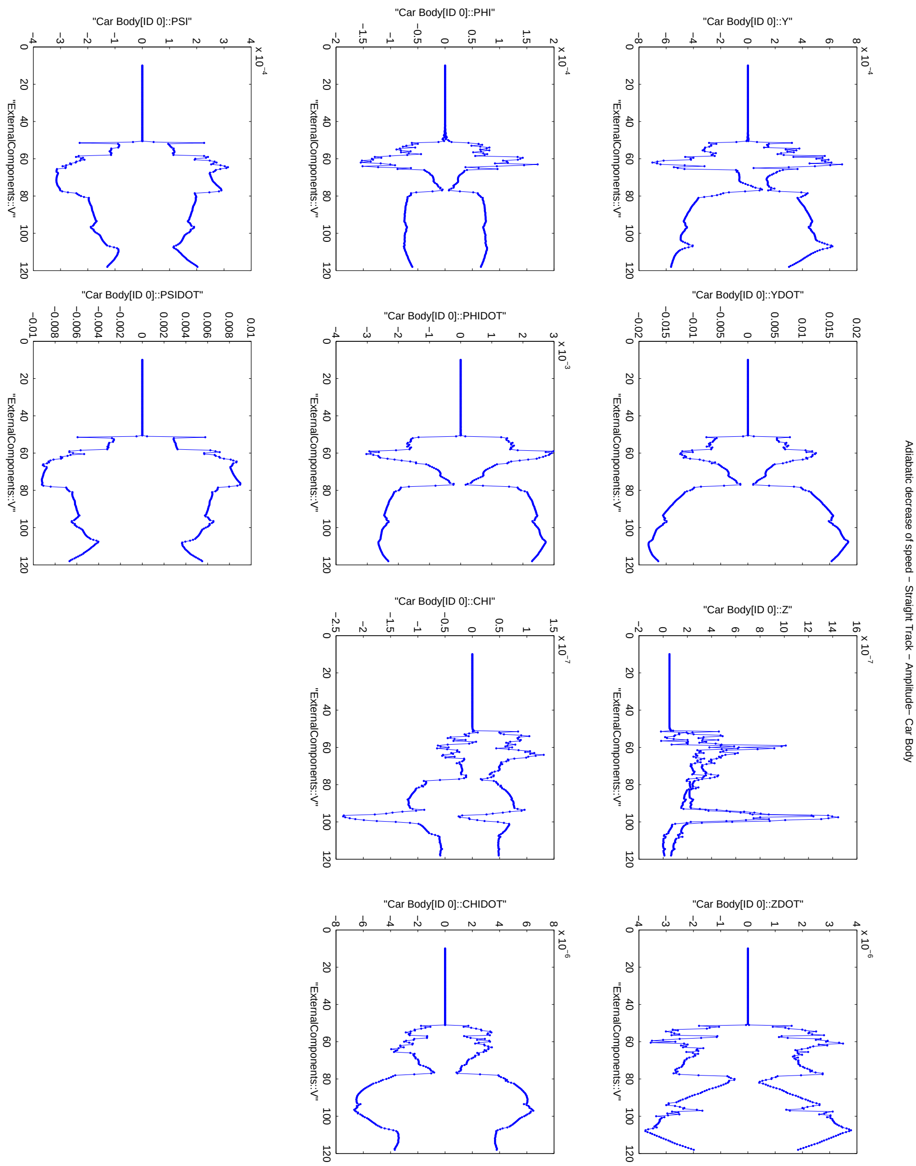


Figure F.28: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the car body with respect to the decreasing speed. The model is running on straight track and does not consider precession forces. The centered stable position is gained back when the speed is decreased to $52\frac{m}{s}$.

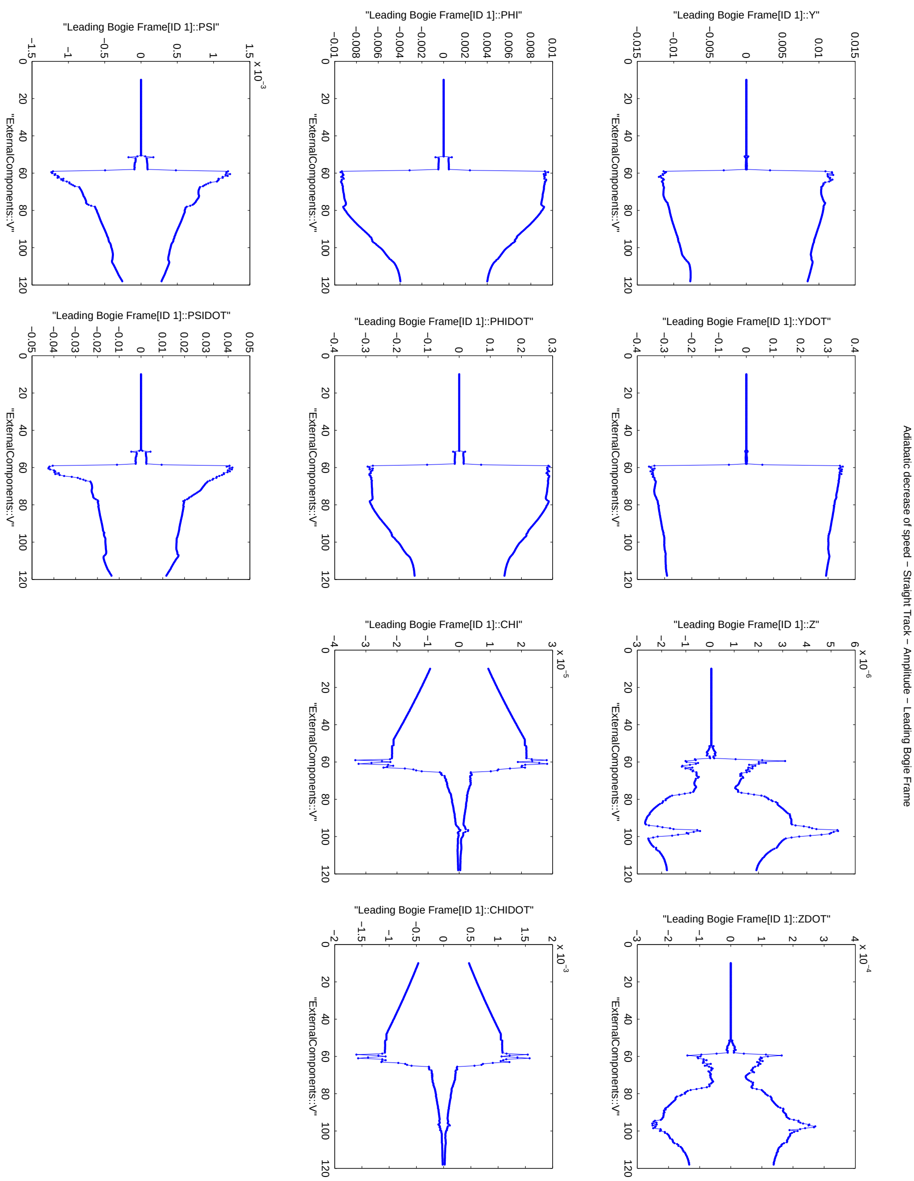


Figure F.29: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the leading bogie frame with respect to the decreasing speed. The model is running on straight track and does not consider precession forces. The centered stable position is gained back when the speed is decreased to $52\frac{m}{s}$.

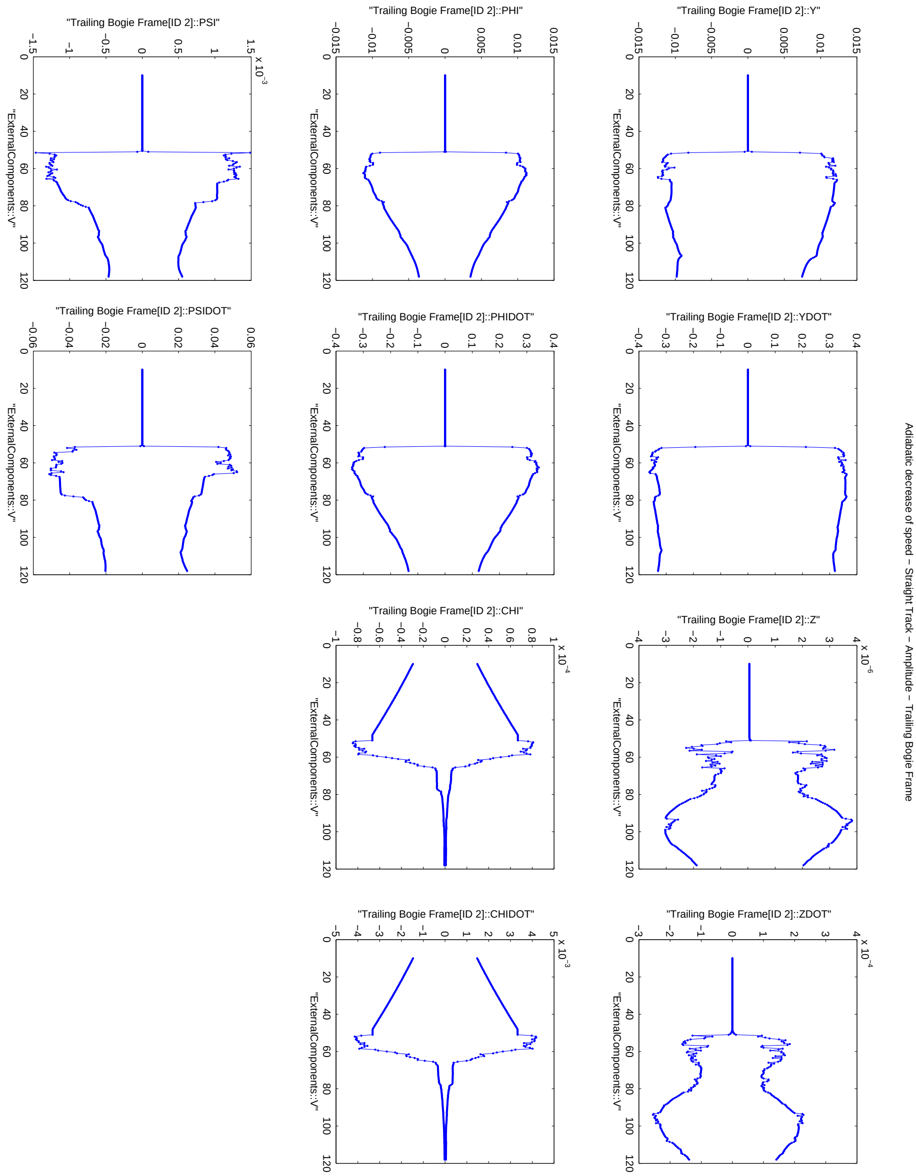


Figure F.30: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the trailing bogie frame with respect to the decreasing speed. The model is running on straight track and does not consider precession forces. The centered stable position is gained back when the speed is decreased to $52\frac{m}{s}$.

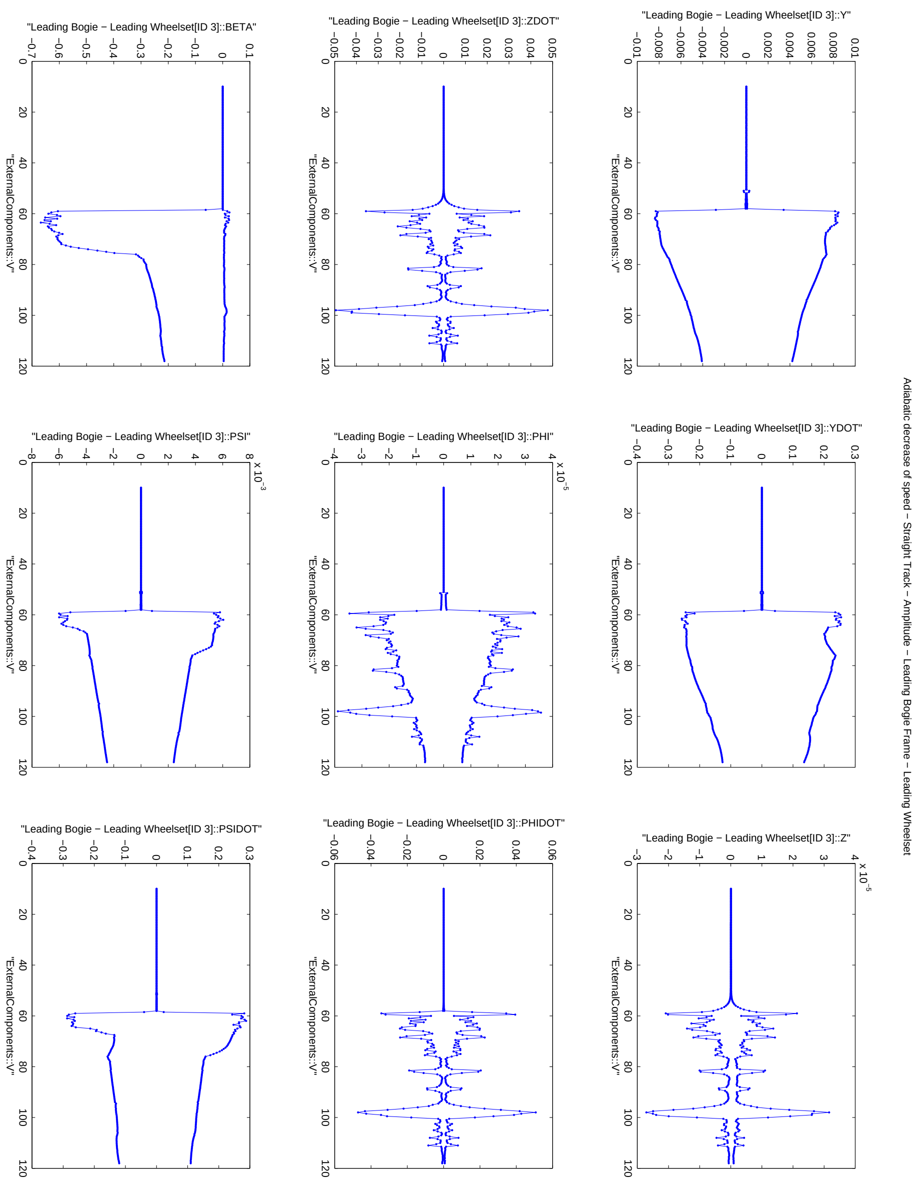


Figure F.31: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the leading wheel set attached to the leading bogie frame frame with respect to the decreasing speed. The model is running on straight track and does not consider precession forces. The centered stable position is gained back when the speed is decreased to $52\frac{m}{s}$.

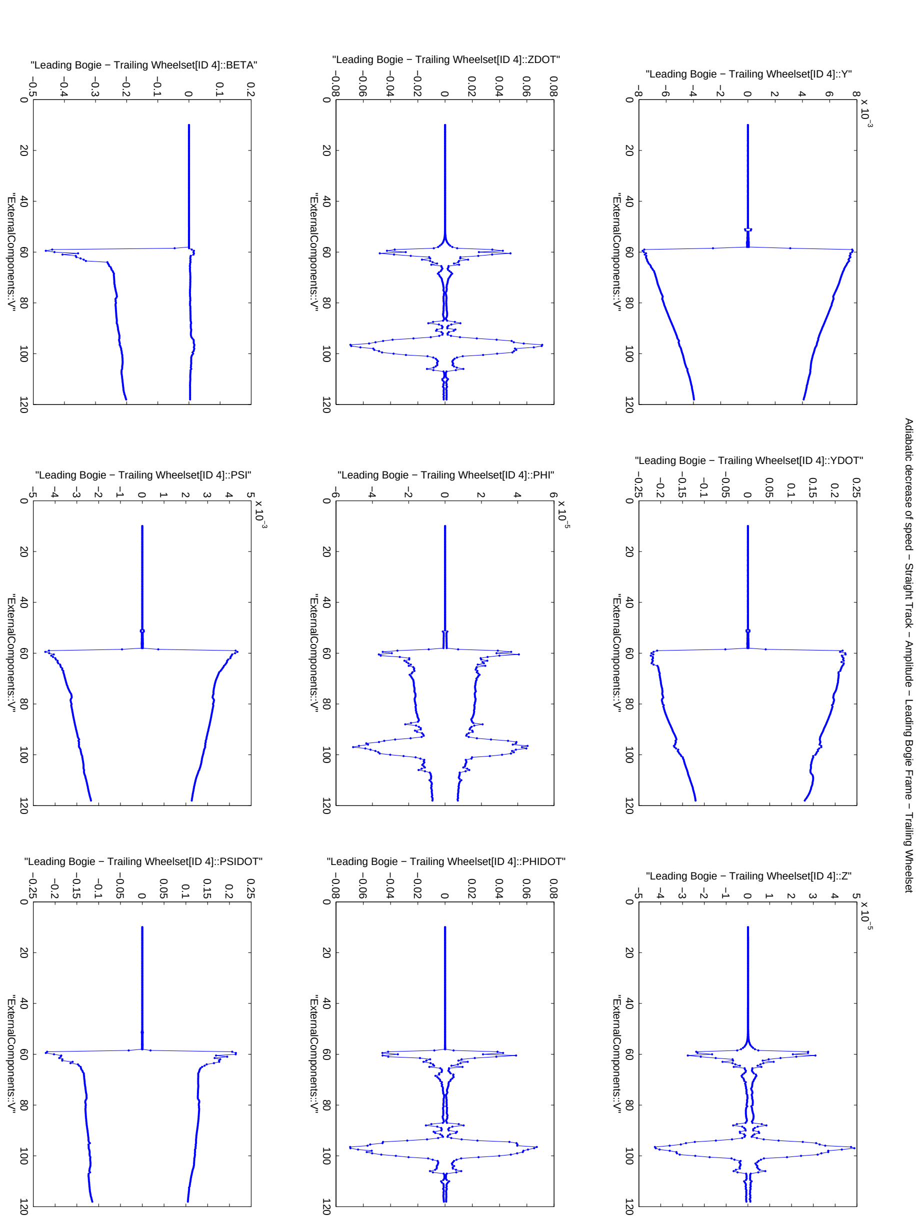


Figure F.32: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the trailing wheel set attached to the leading bogie frame frame with respect to the decreasing speed. The model is running on straight track and does not consider precession forces. The centered stable position is gained back when the speed is decreased to $52\frac{m}{s}$.

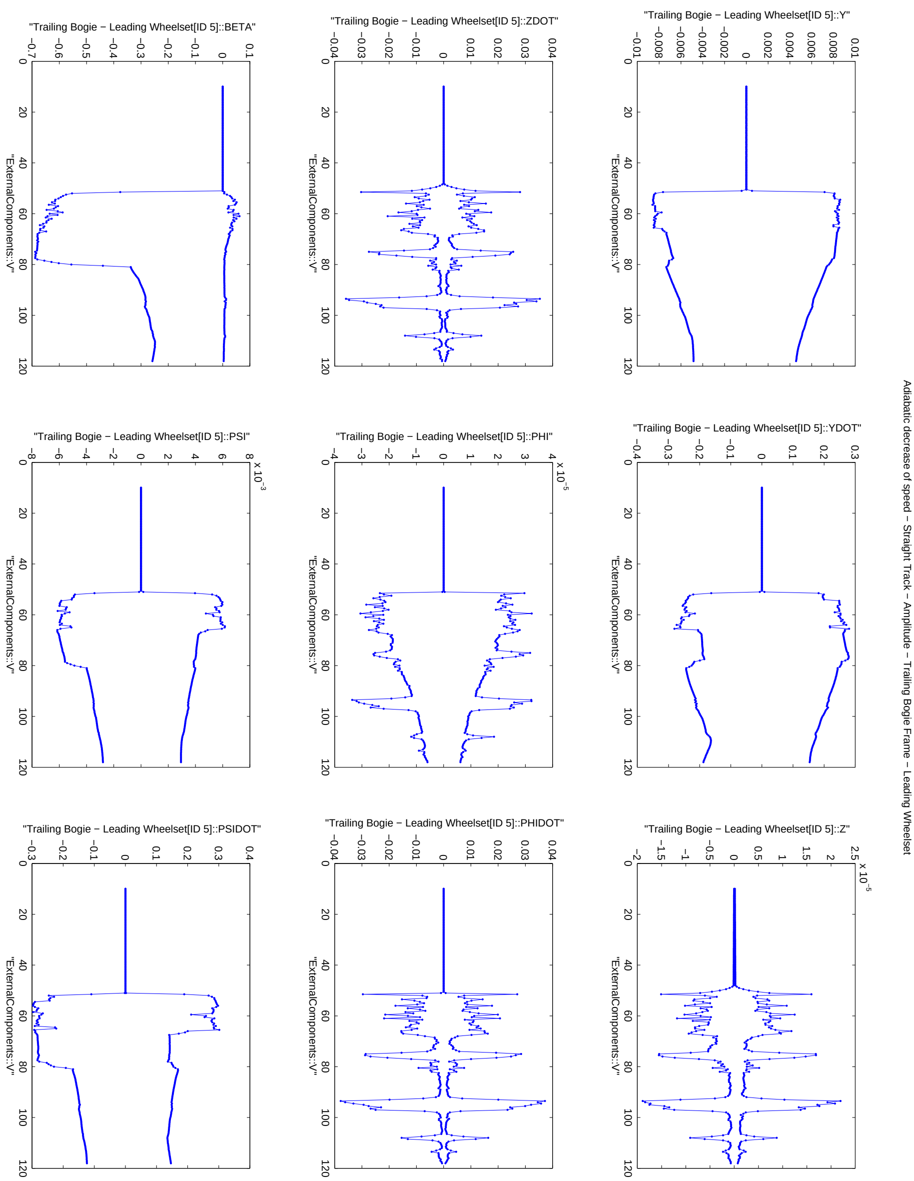


Figure F.33: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the leading wheel set attached to the trailing bogie frame frame with respect to the decreasing speed. The model is running on straight track and does not consider precession forces. The centered stable position is gained back when the speed is decreased to $52\frac{m}{s}$.

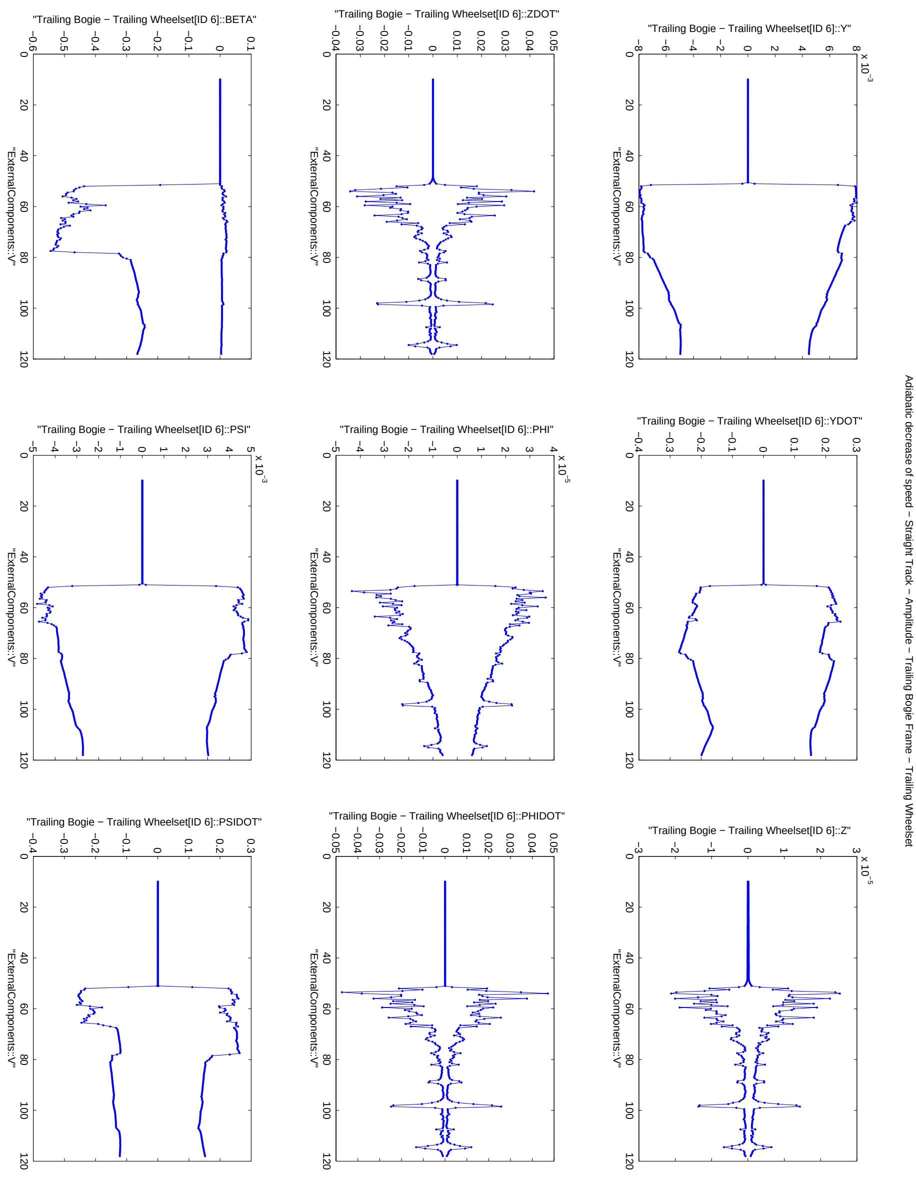


Figure F.34: Cooperrider model: variation amplitude from the central stable solution of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame frame with respect to the decreasing speed. The model is running on straight track and does not consider precession forces. The centered stable position is gained back when the speed is decreased to $52\frac{m}{s}$.

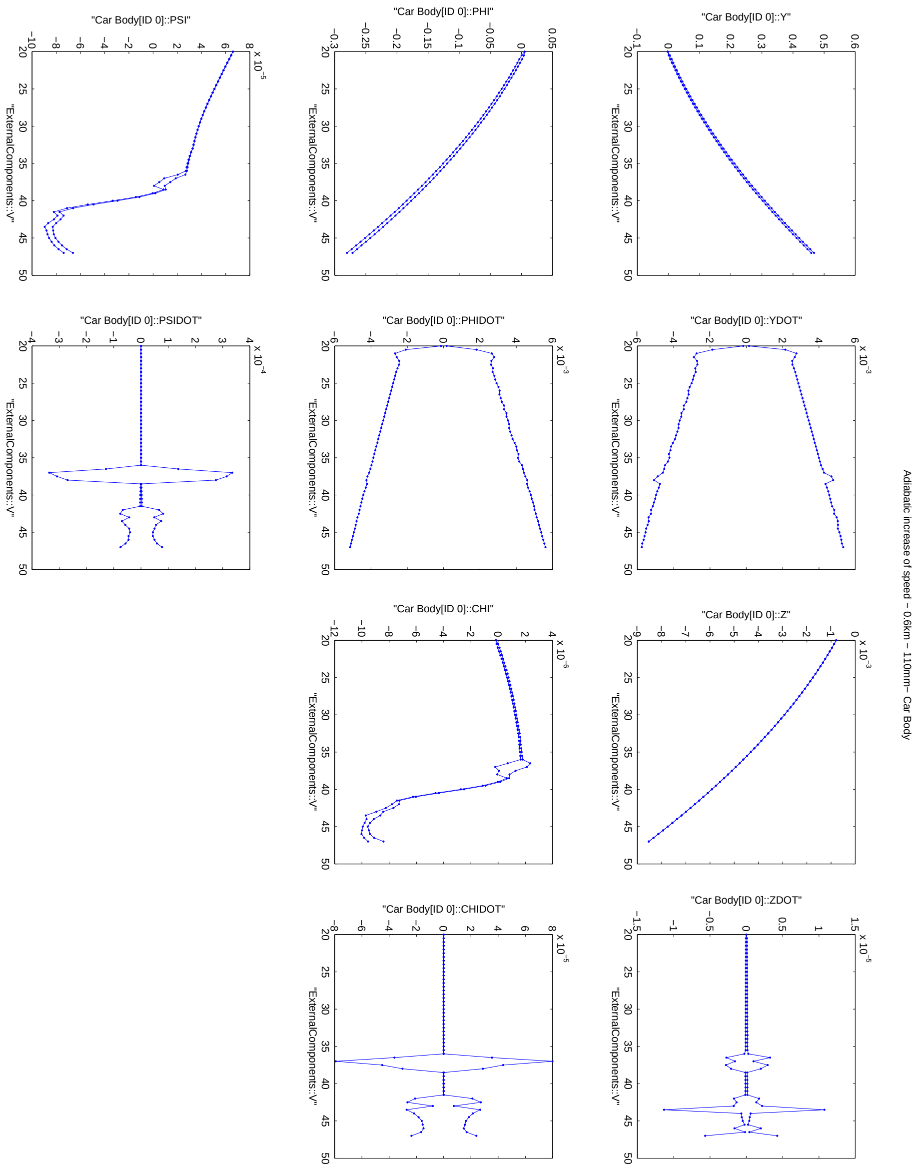


Figure F.35: Cooperrider model: variation amplitude of the degrees of freedom of the car body with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model does not consider precession forces.

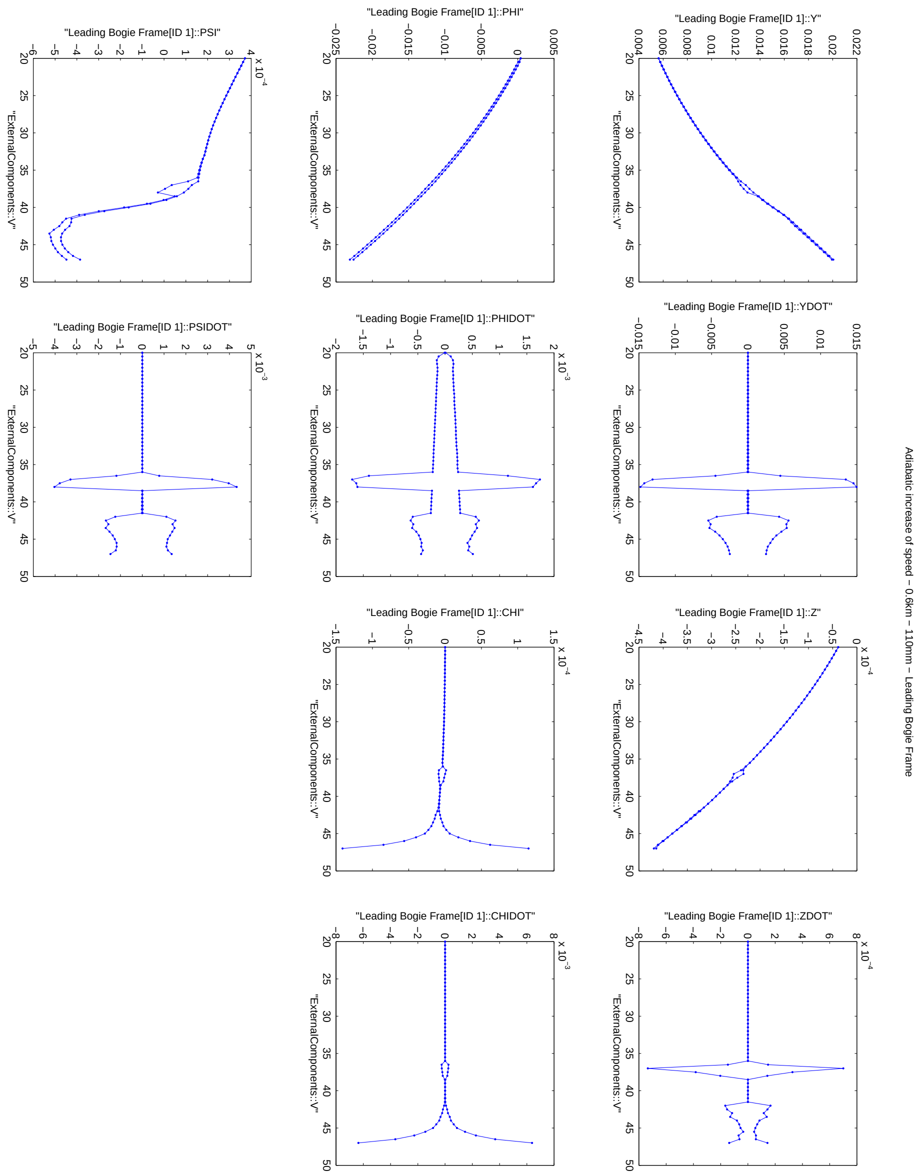


Figure F.36: Cooperrider model: variation amplitude of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model does not consider precession forces.

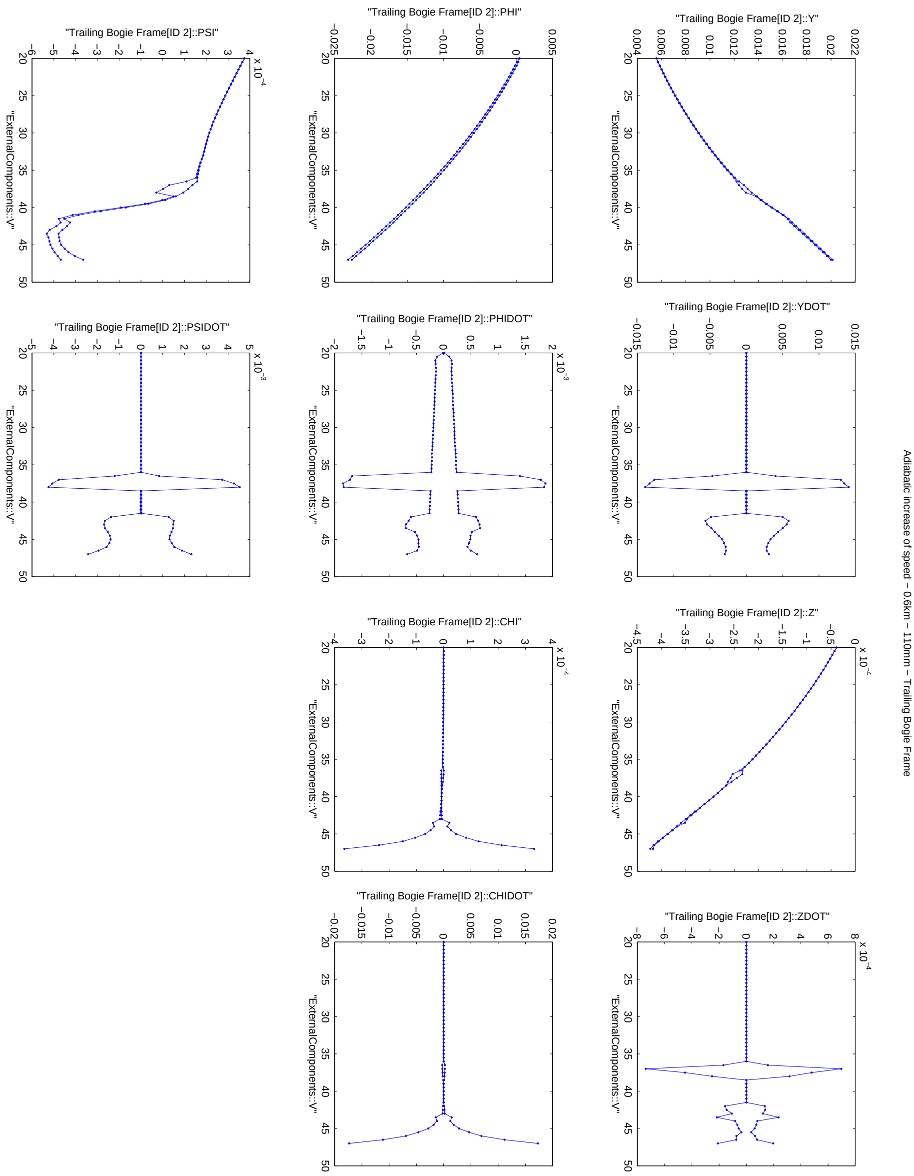


Figure F.37: Cooperrider model: variation amplitude of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model does not consider precession forces.

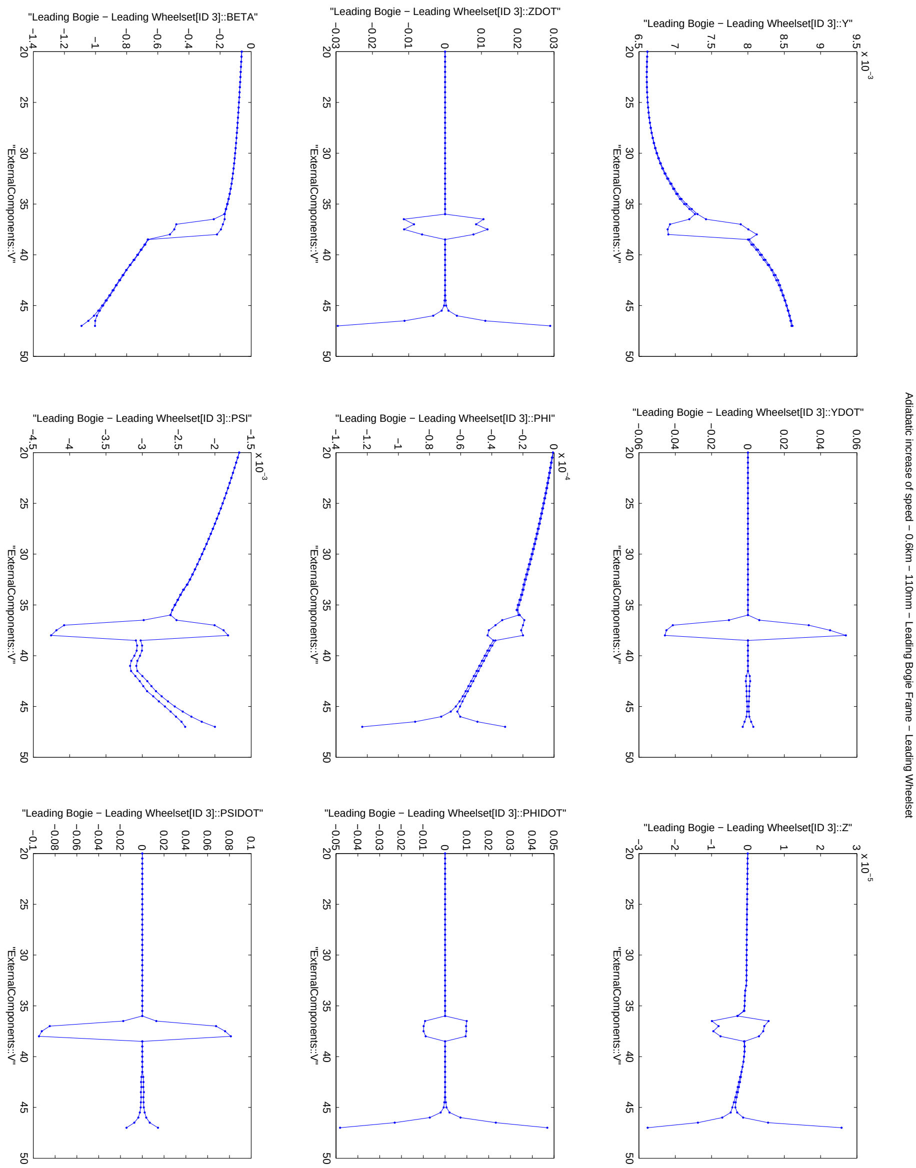


Figure F.38: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model does not consider precession forces.

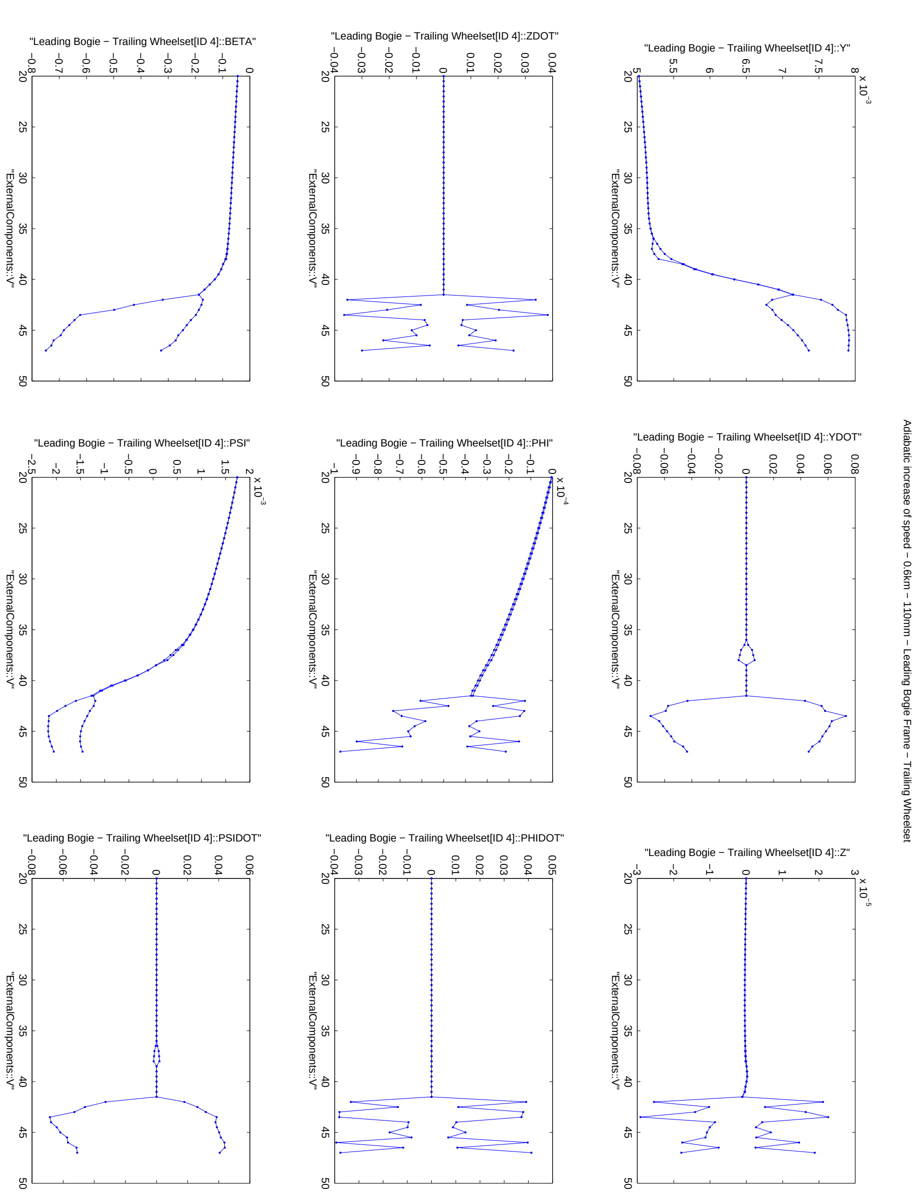


Figure F.39: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model does not consider precession forces.

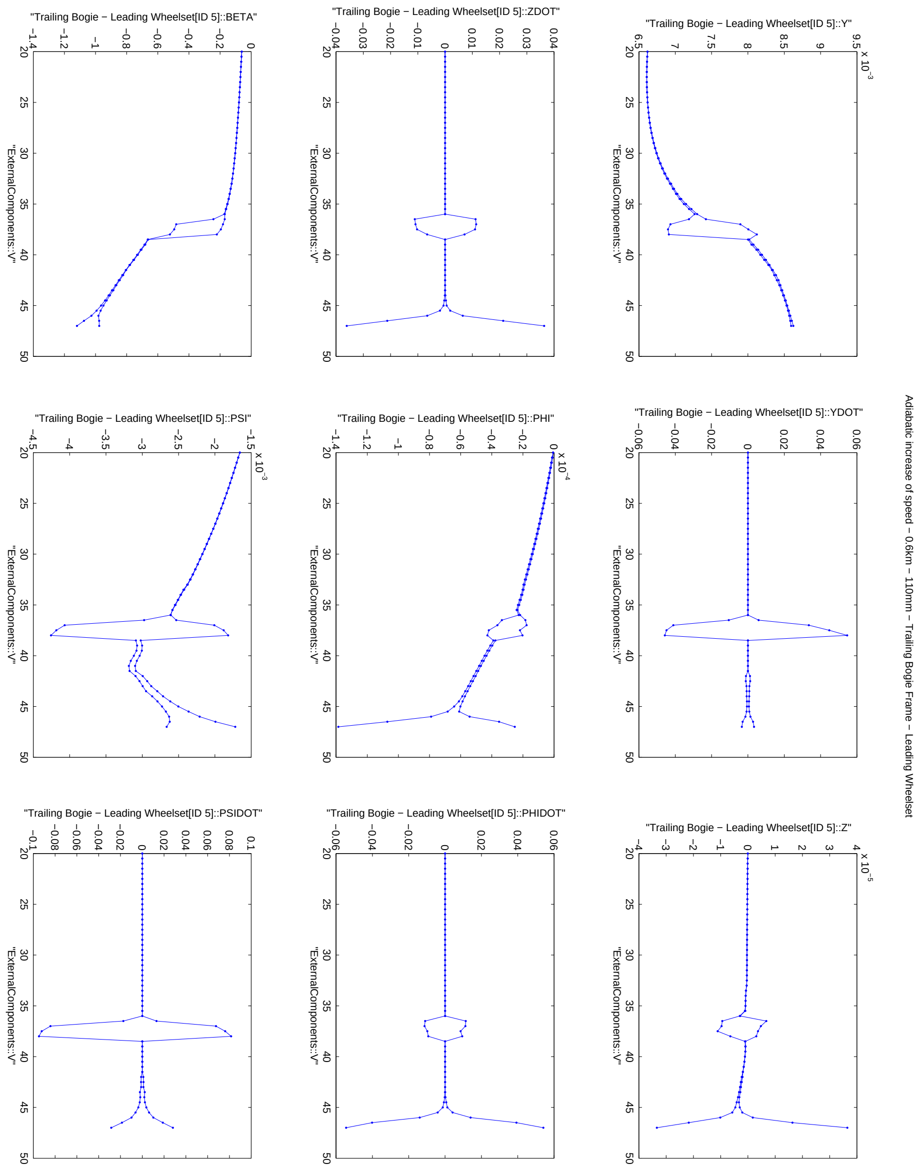


Figure F.40: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model does not consider precession forces.

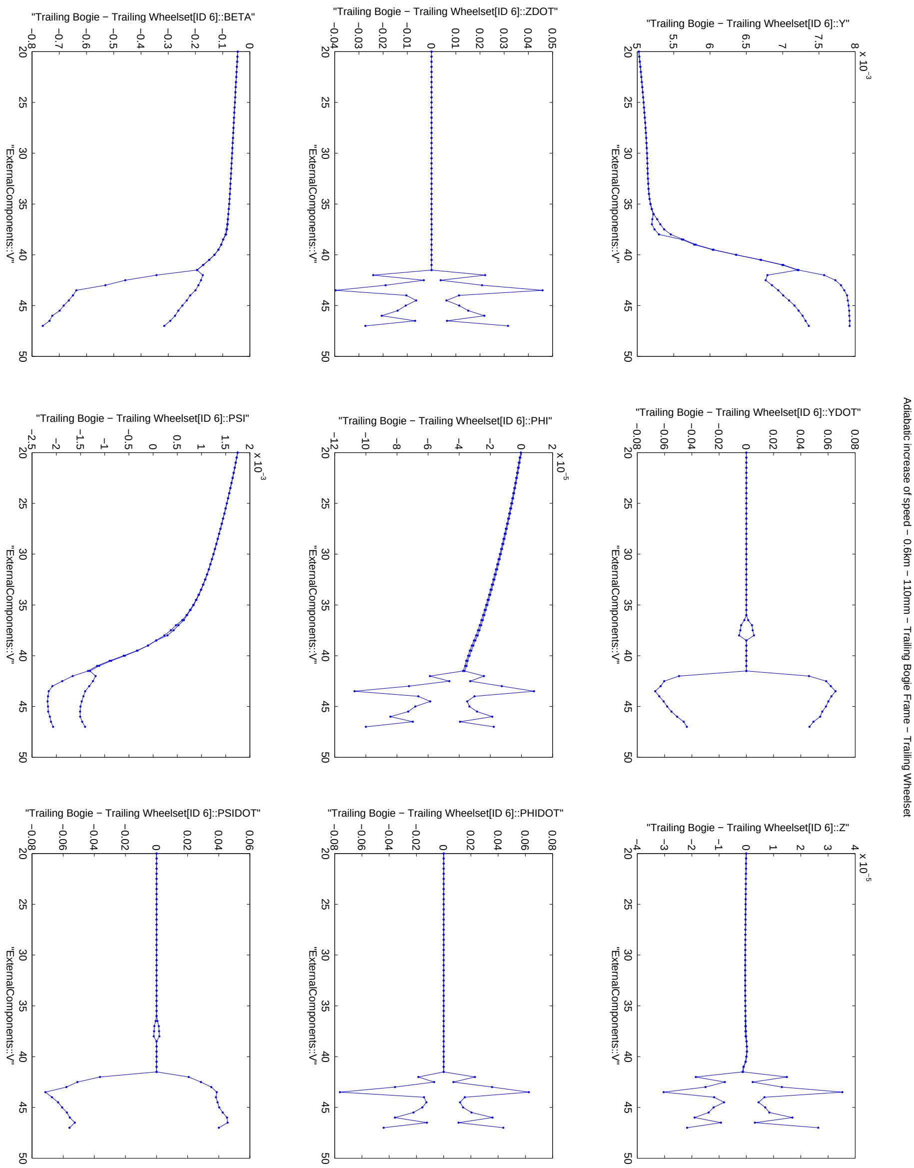


Figure F.41: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model does not consider precession forces.

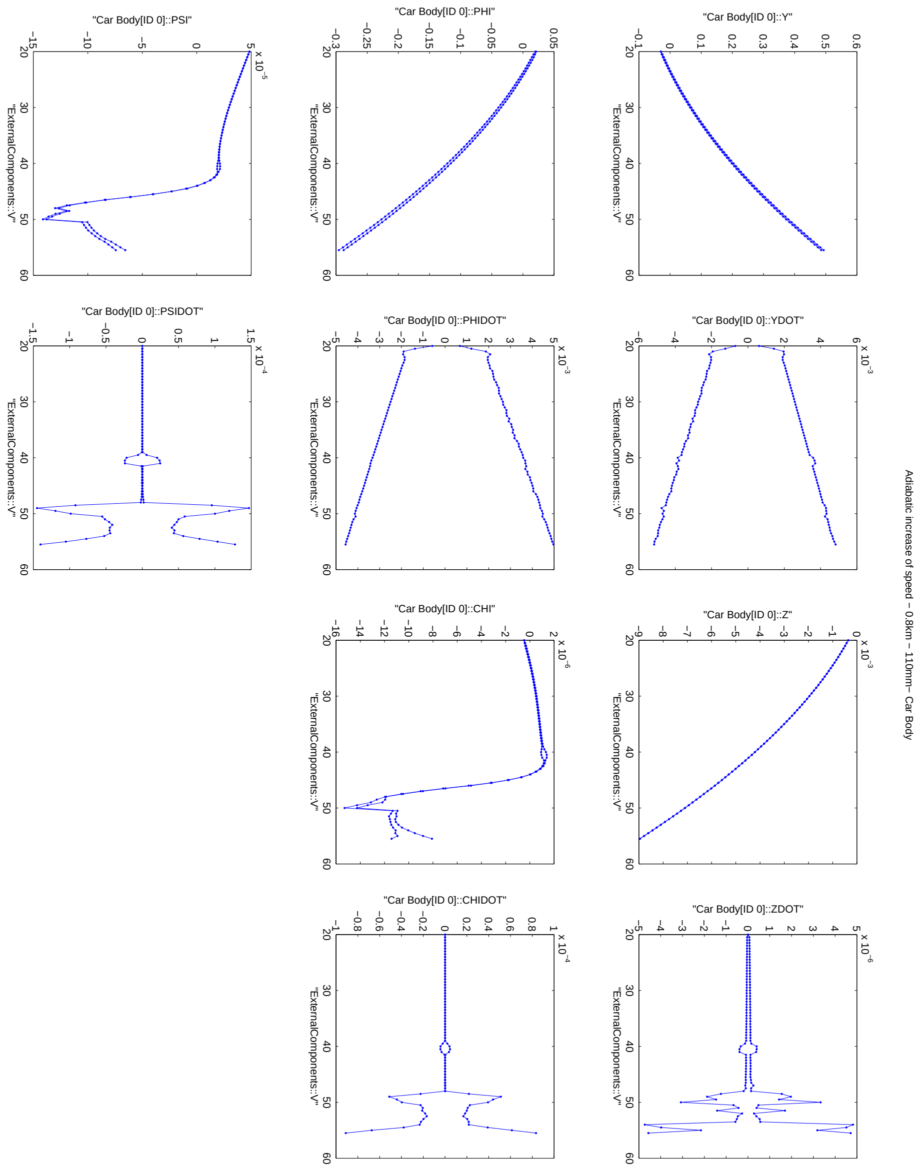


Figure F.42: Cooperrider model: variation amplitude of the degrees of freedom of the car body with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model does not consider precession forces.

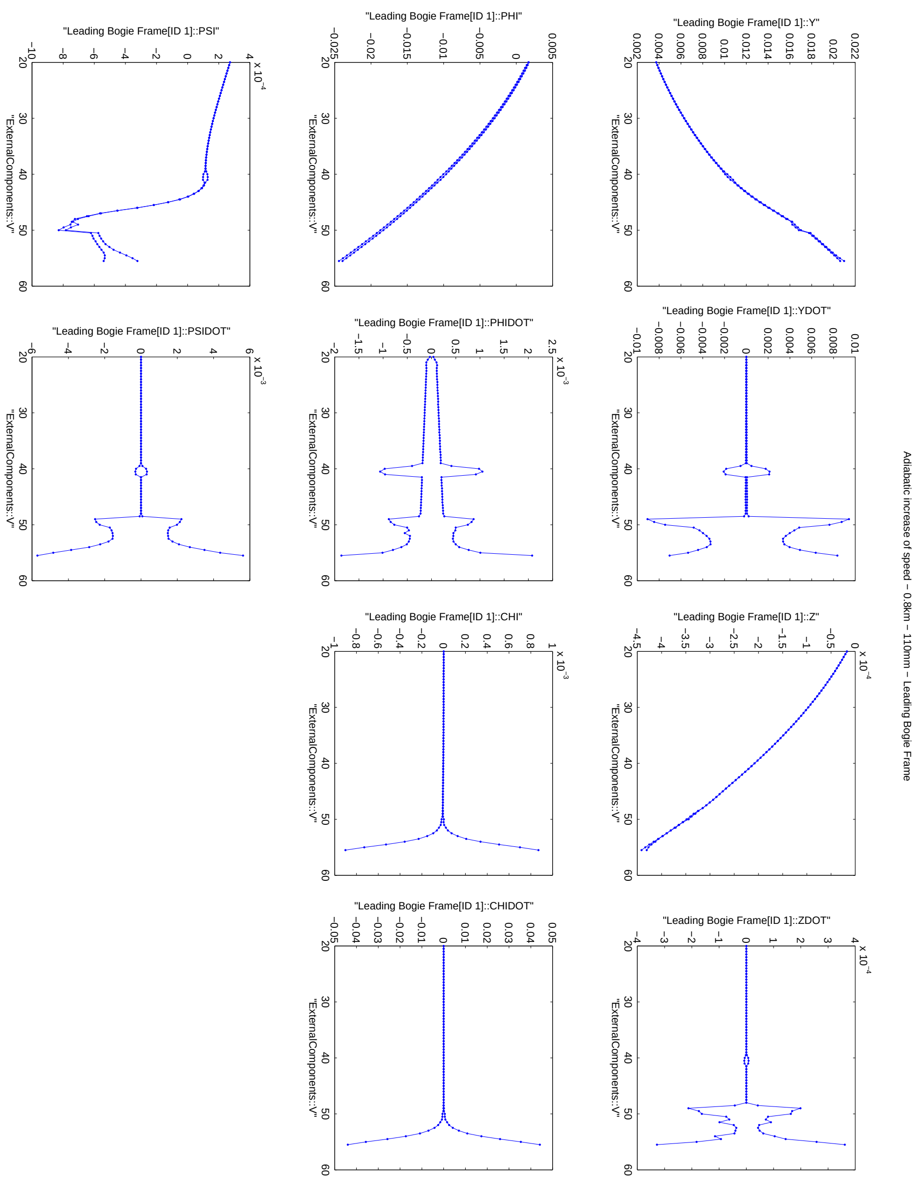


Figure F.43: Cooperrider model: variation amplitude of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model does not consider precession forces.

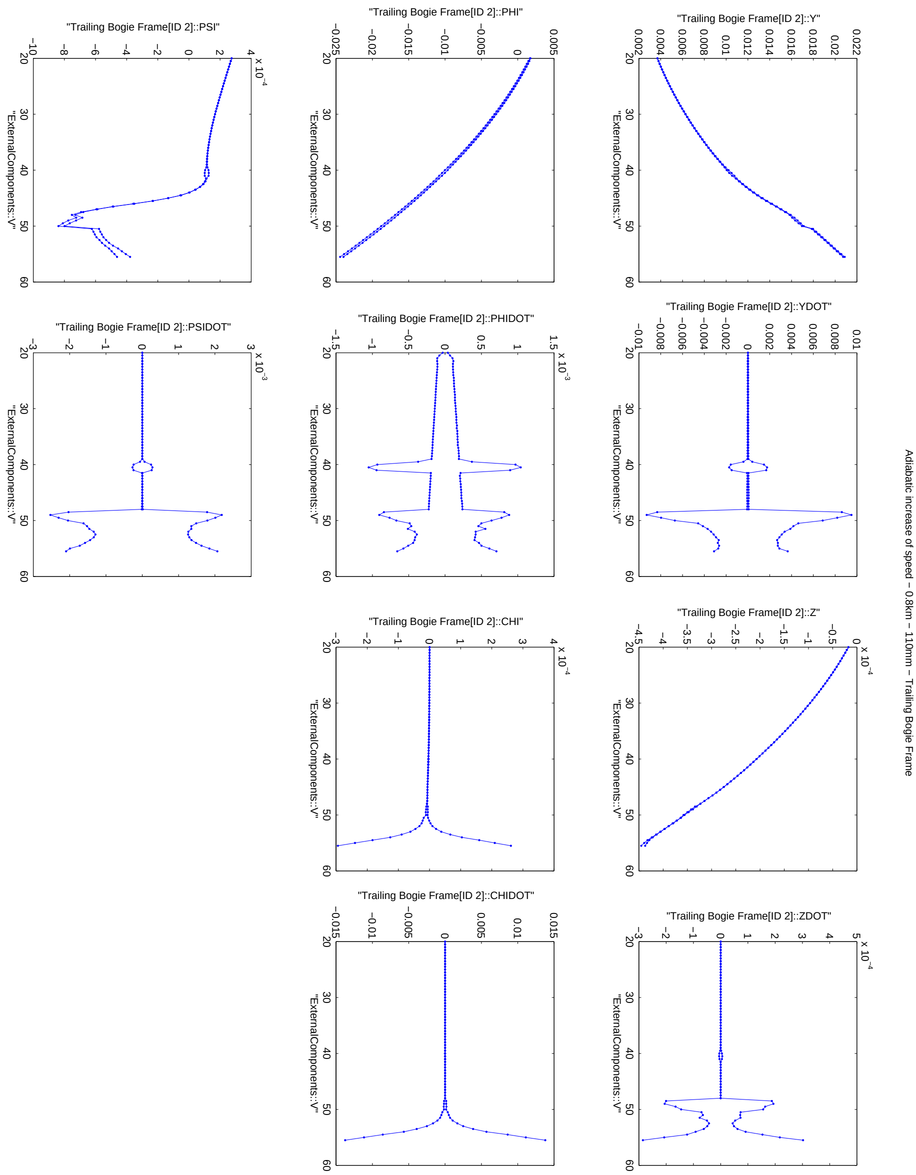


Figure F.44: Cooperrider model: variation amplitude of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model does not consider precession forces.

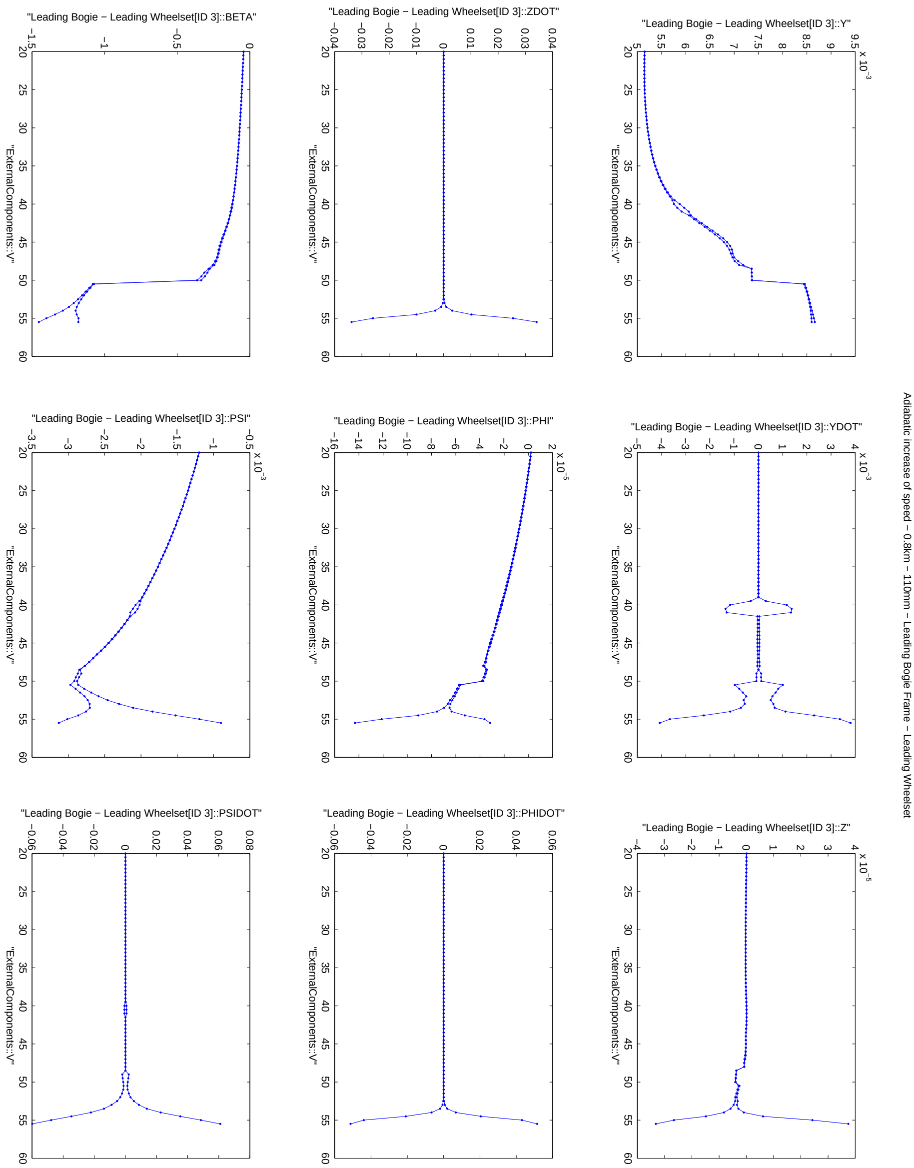


Figure F.45: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model does not consider precession forces.

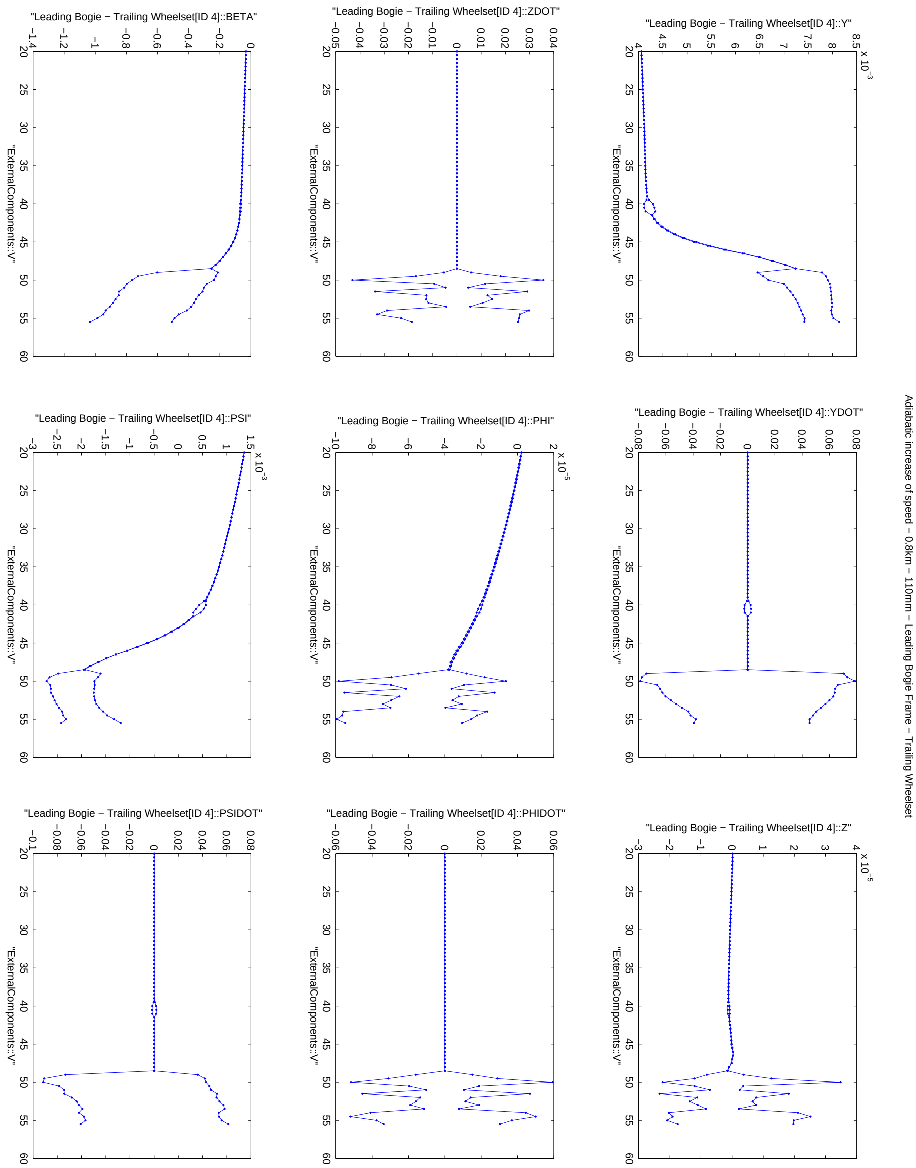


Figure F.46: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model does not consider precession forces.

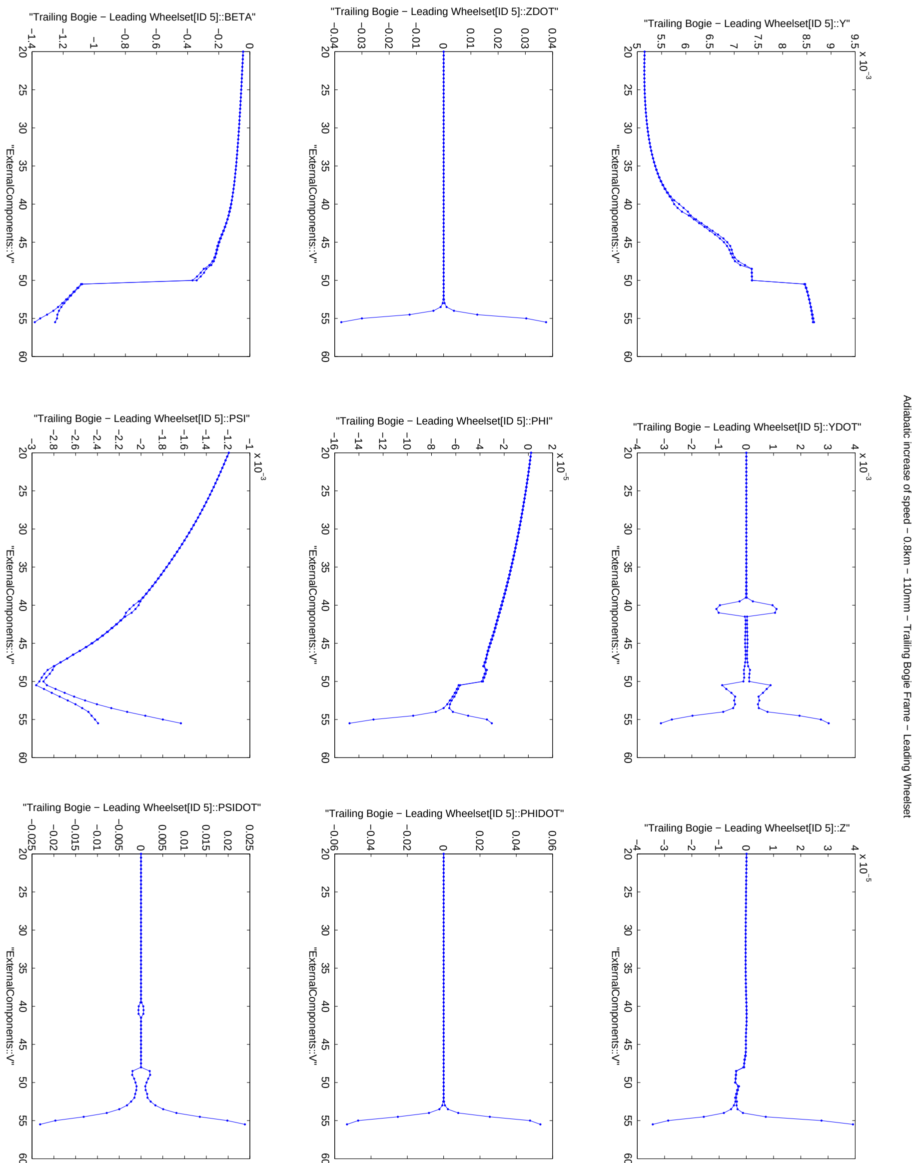


Figure F.47: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model does not consider precession forces.

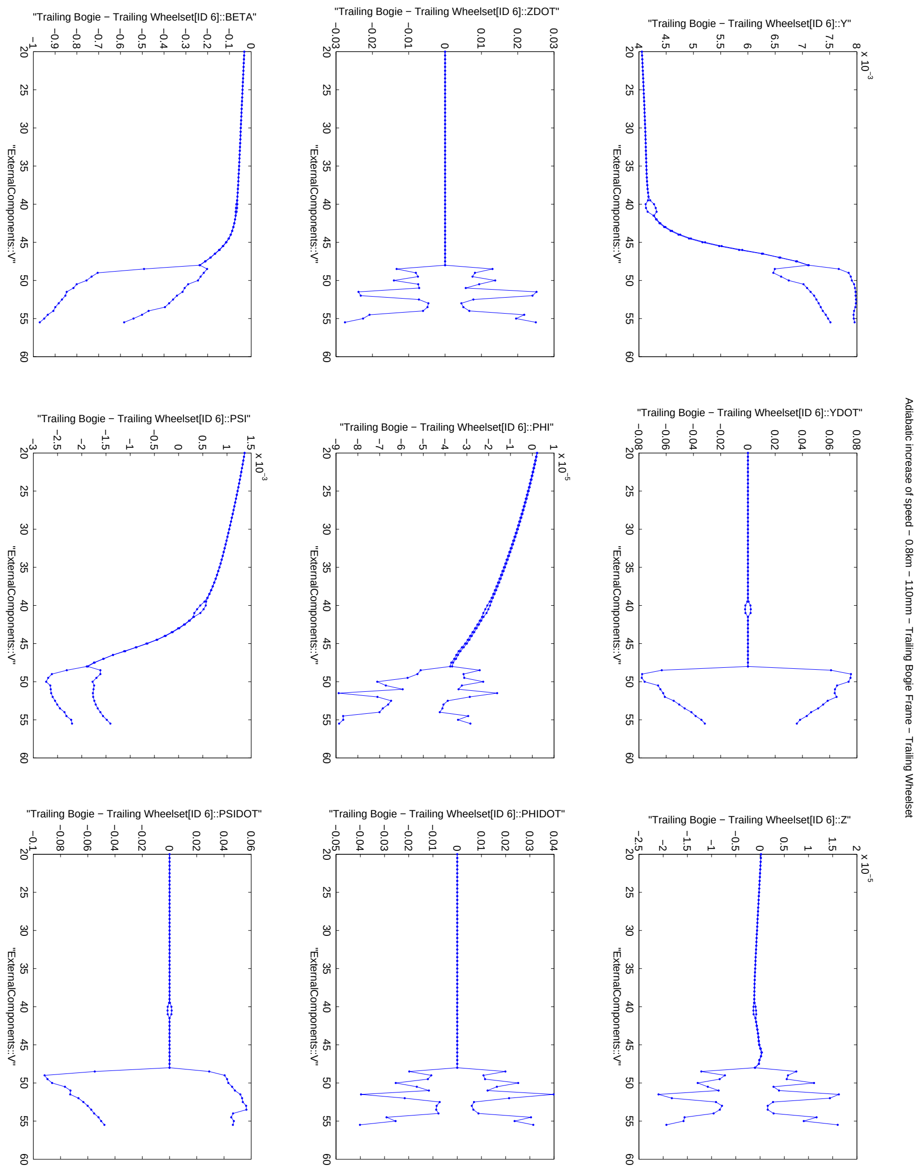


Figure F.48: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model does not consider precession forces.

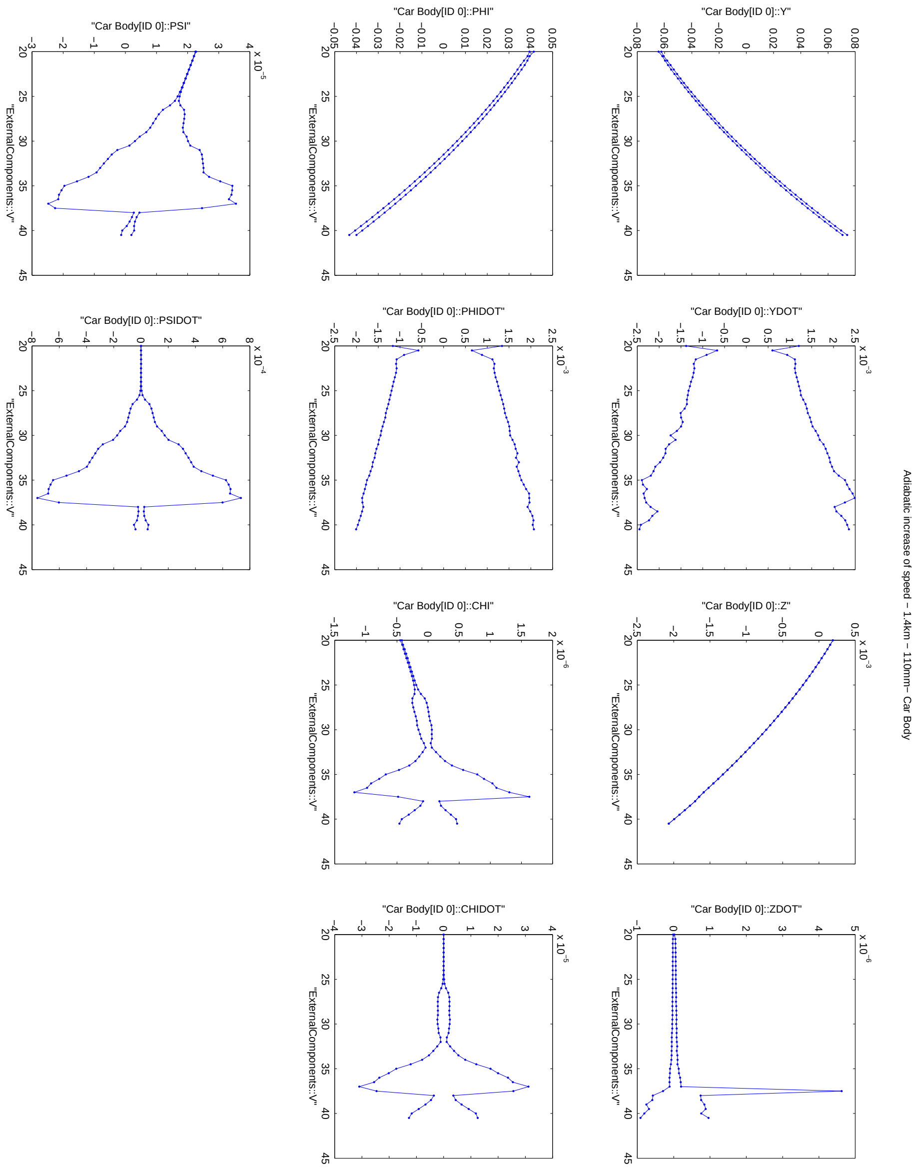


Figure F.49: Cooperrider model: variation amplitude of the degrees of freedom of the car body with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model does not consider precession forces.

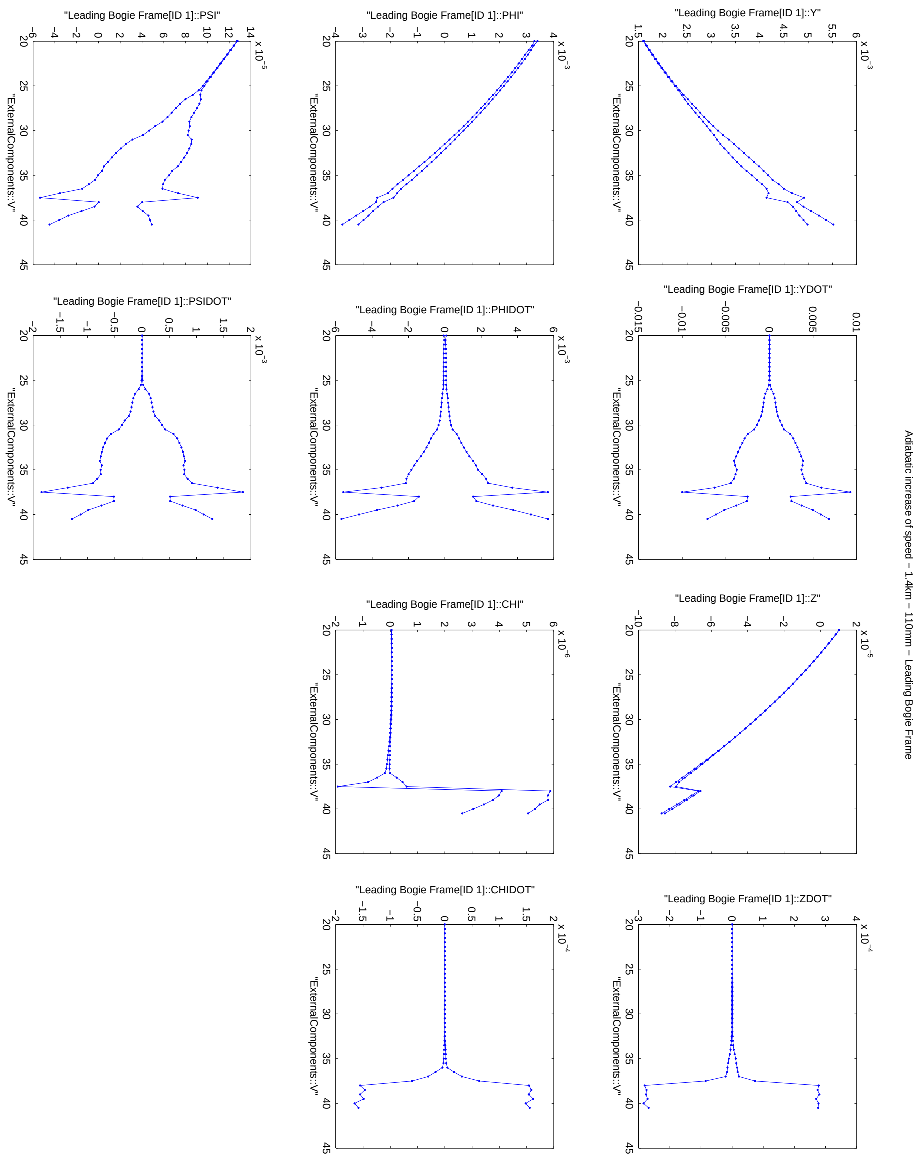


Figure F.50: Cooperrider model: variation amplitude of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model does not consider precession forces.

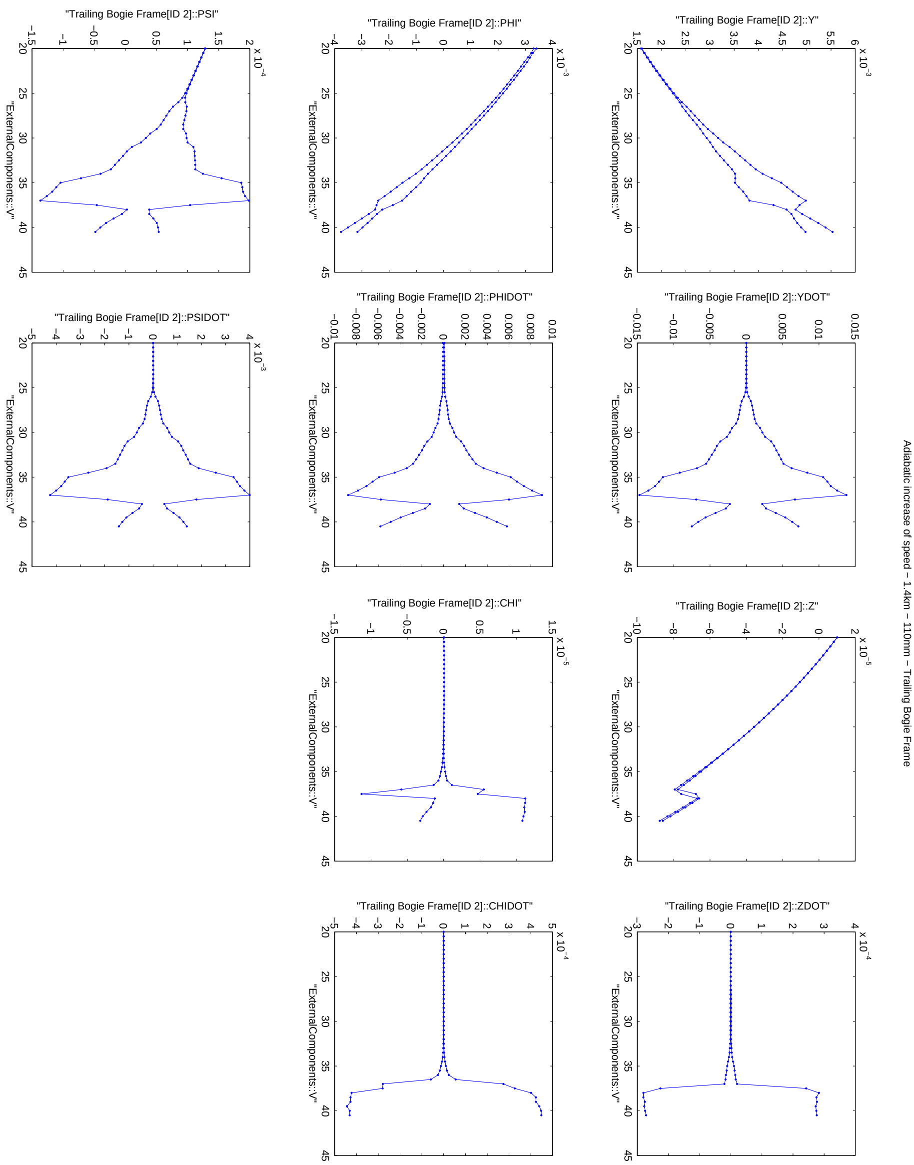


Figure F.51: Cooperrider model: variation amplitude of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model does not consider precession forces.

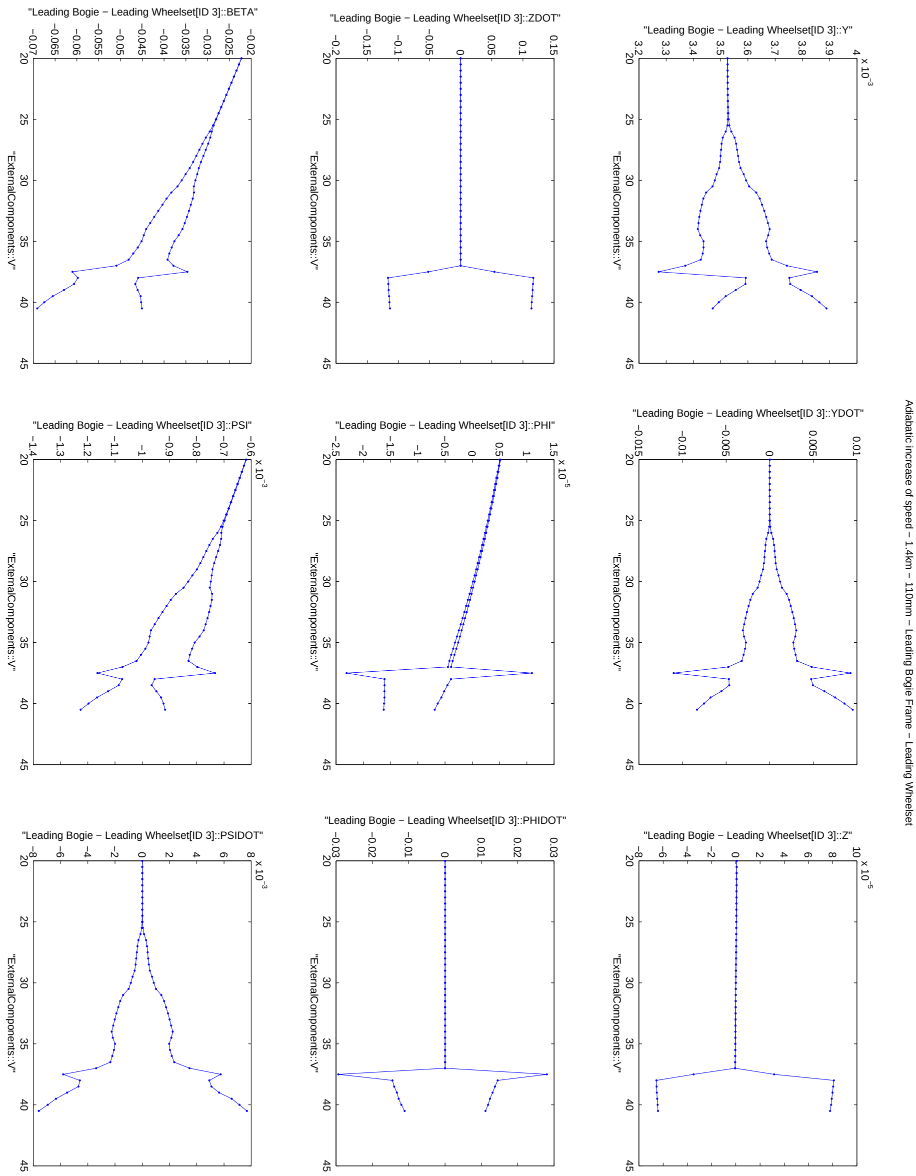


Figure F.52: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model does not consider precession forces.

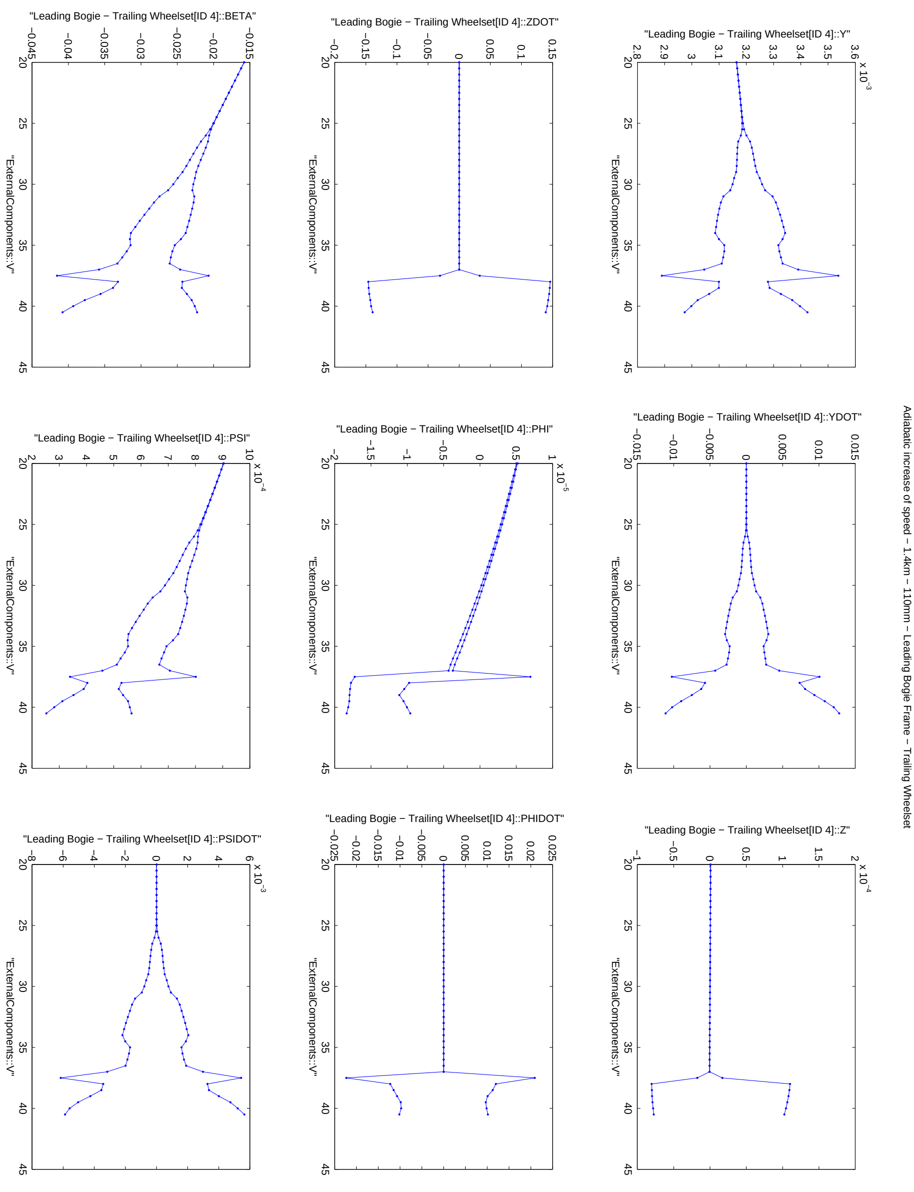


Figure F.53: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model does not consider precession forces.

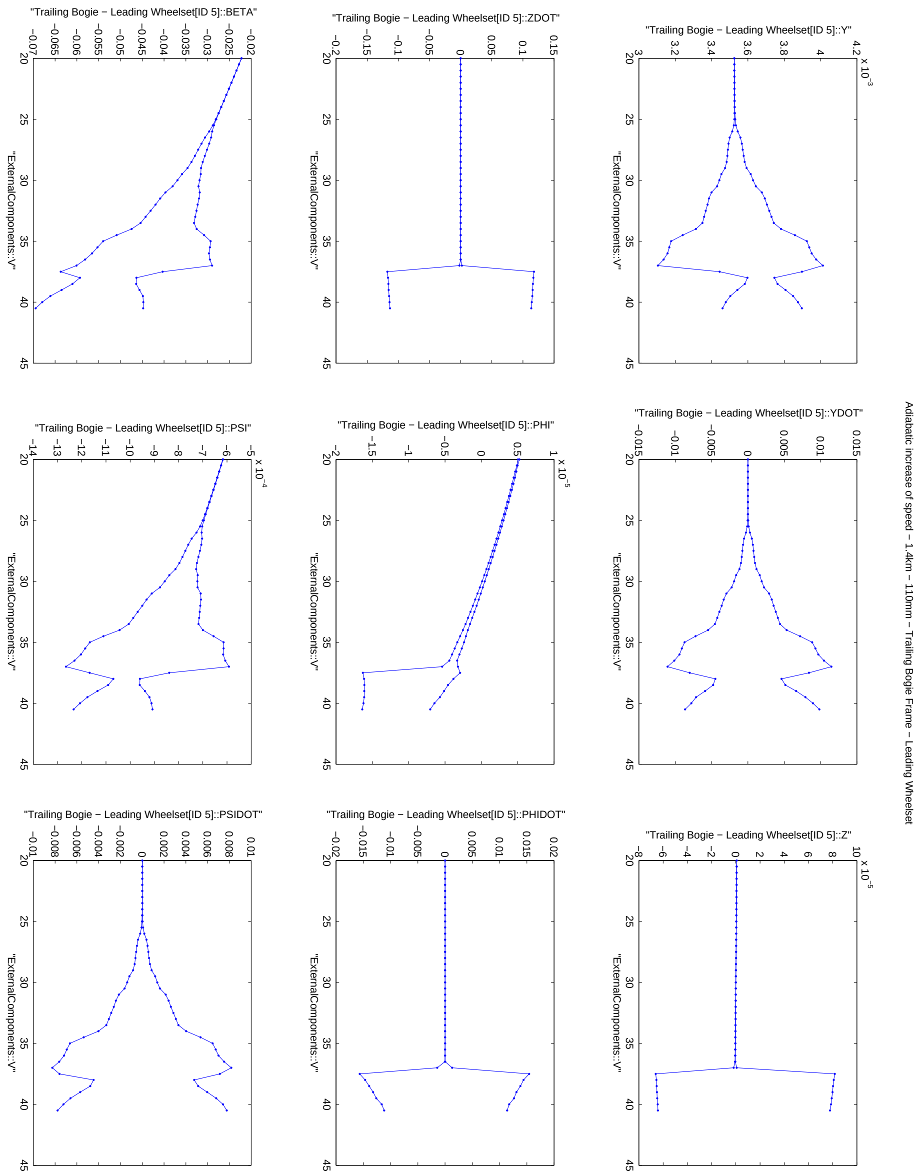


Figure F.54: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model does not consider precession forces.

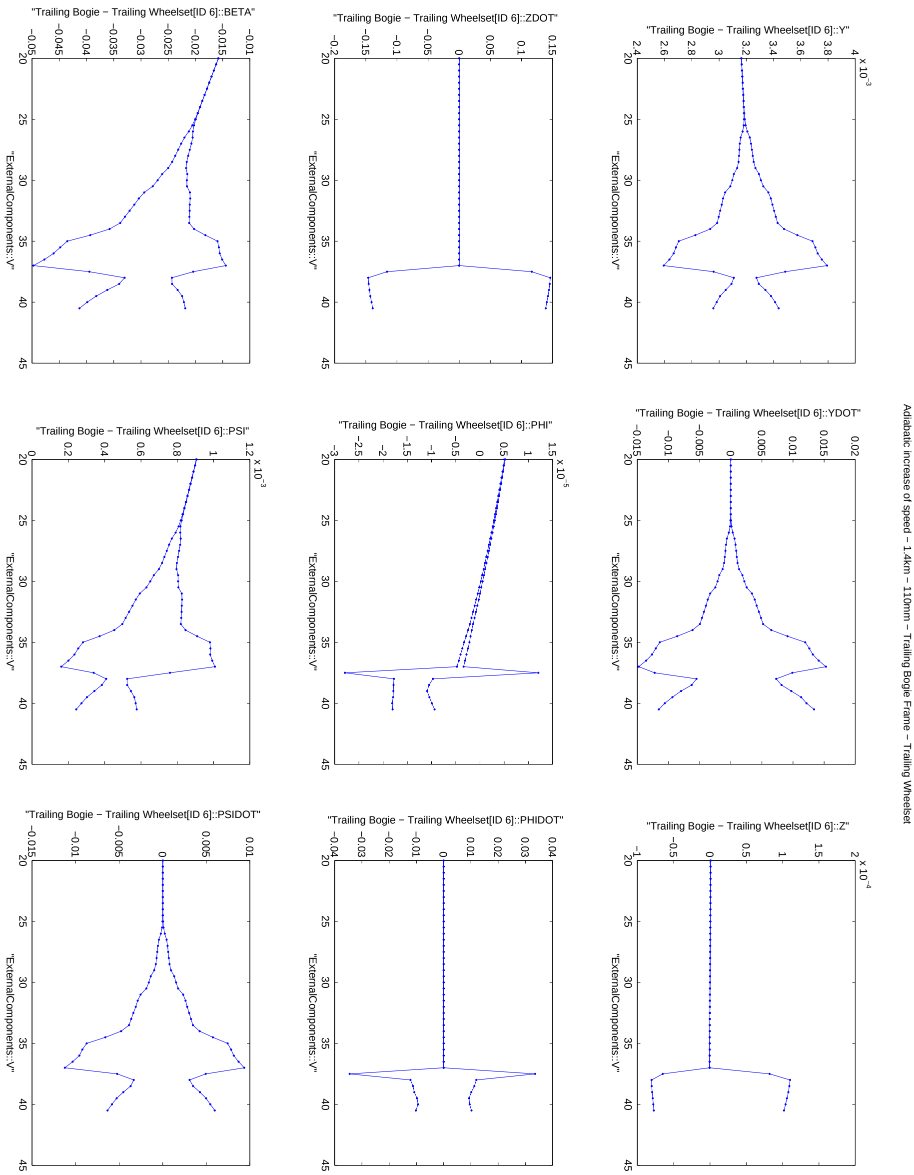


Figure F.55: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model does not consider precession forces.

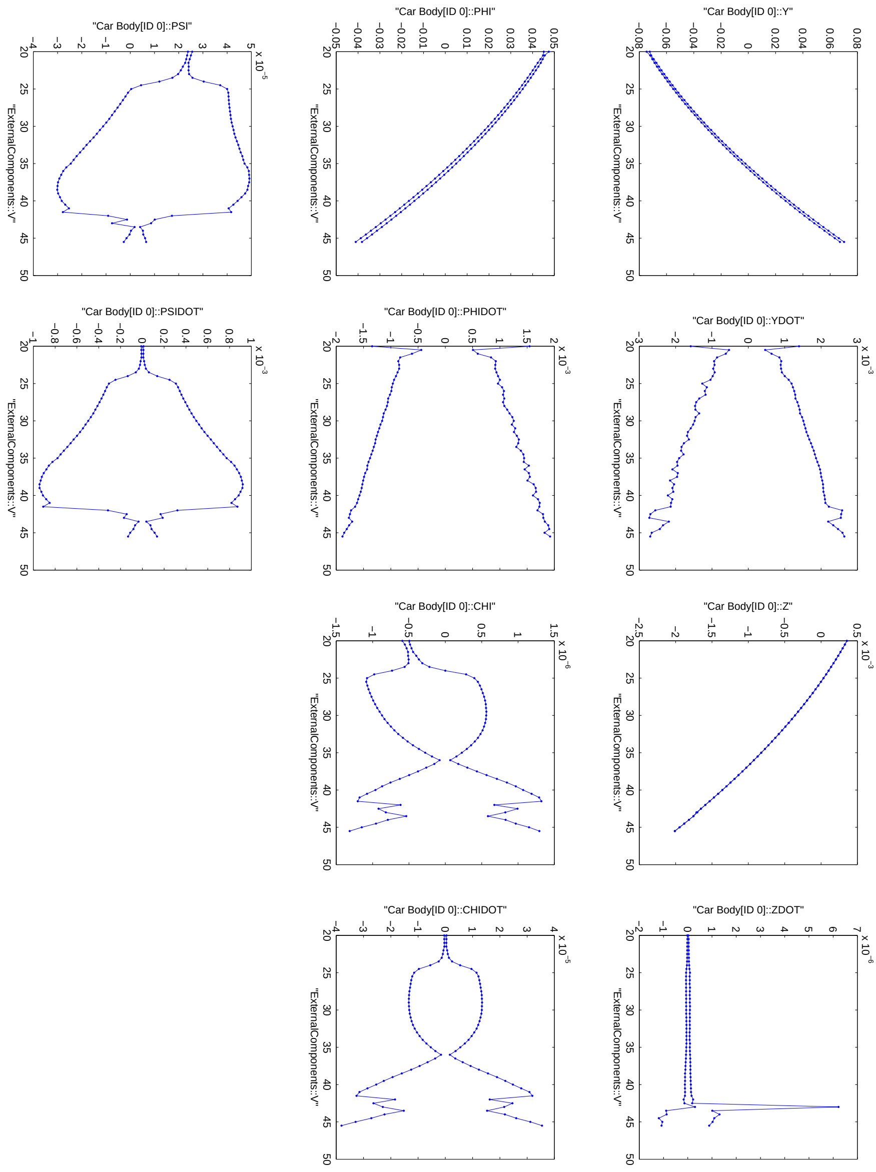


Figure F.56: Cooperrider model: variation amplitude of the degrees of freedom of the car body with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model does not consider precession forces.

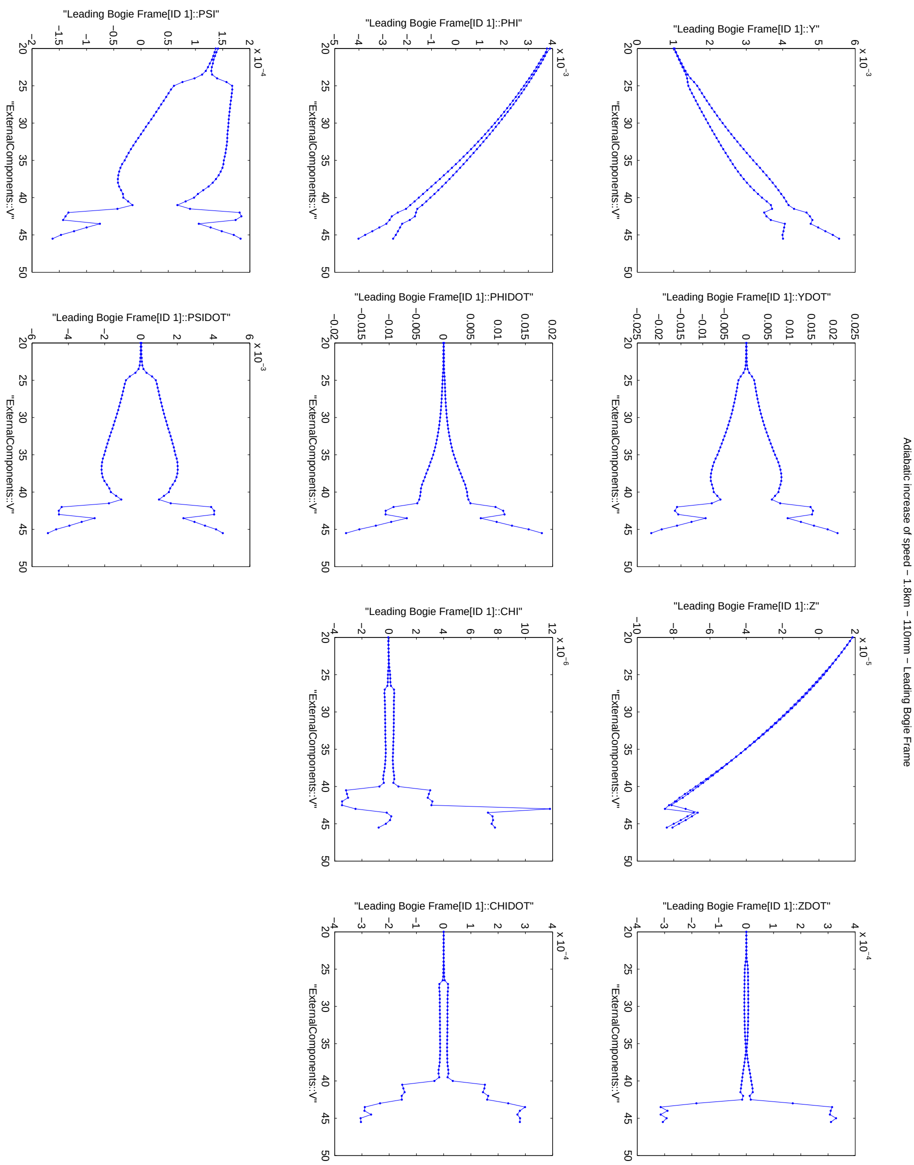


Figure F.57: Cooperrider model: variation amplitude of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model does not consider precession forces.

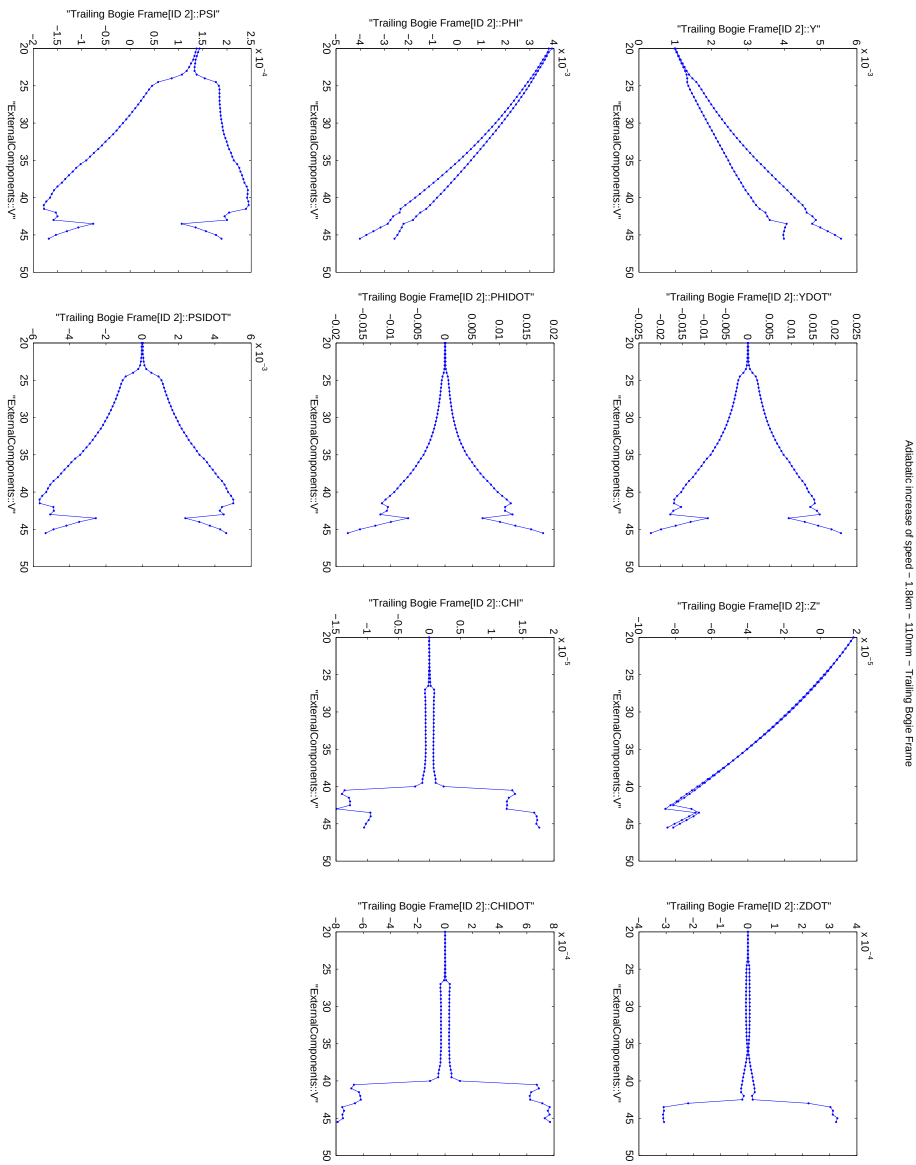


Figure F.58: Cooperrider model: variation amplitude of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model does not consider precession forces.

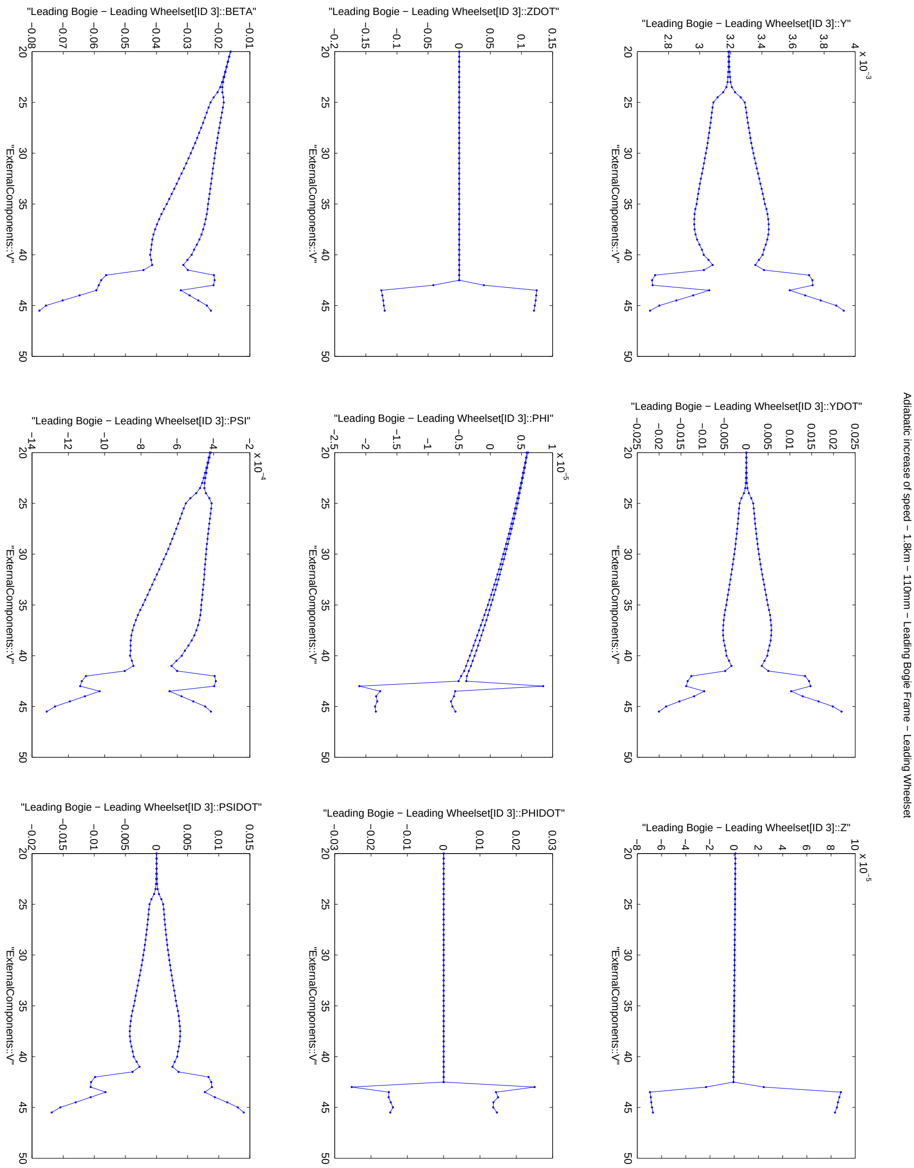


Figure F.59: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model does not consider precession forces.

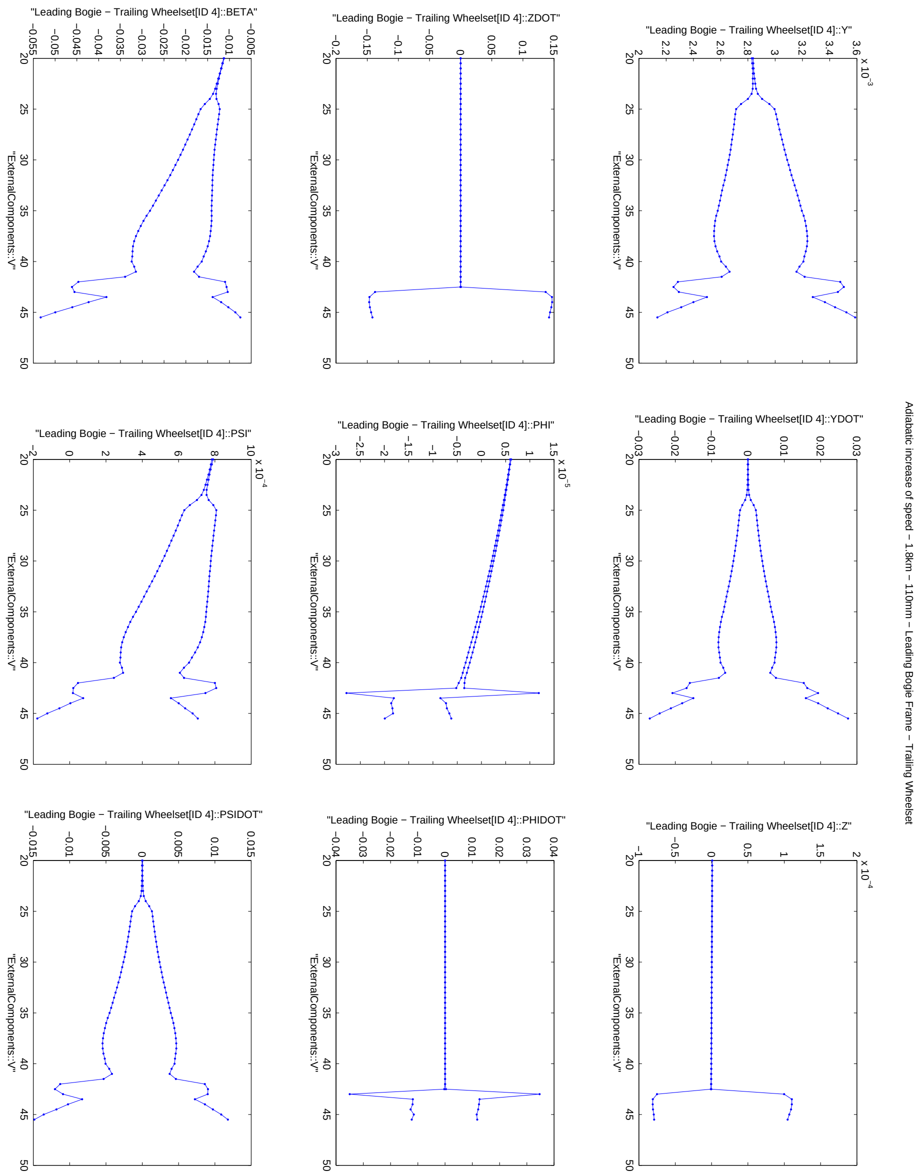


Figure F.60: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model does not consider precession forces.

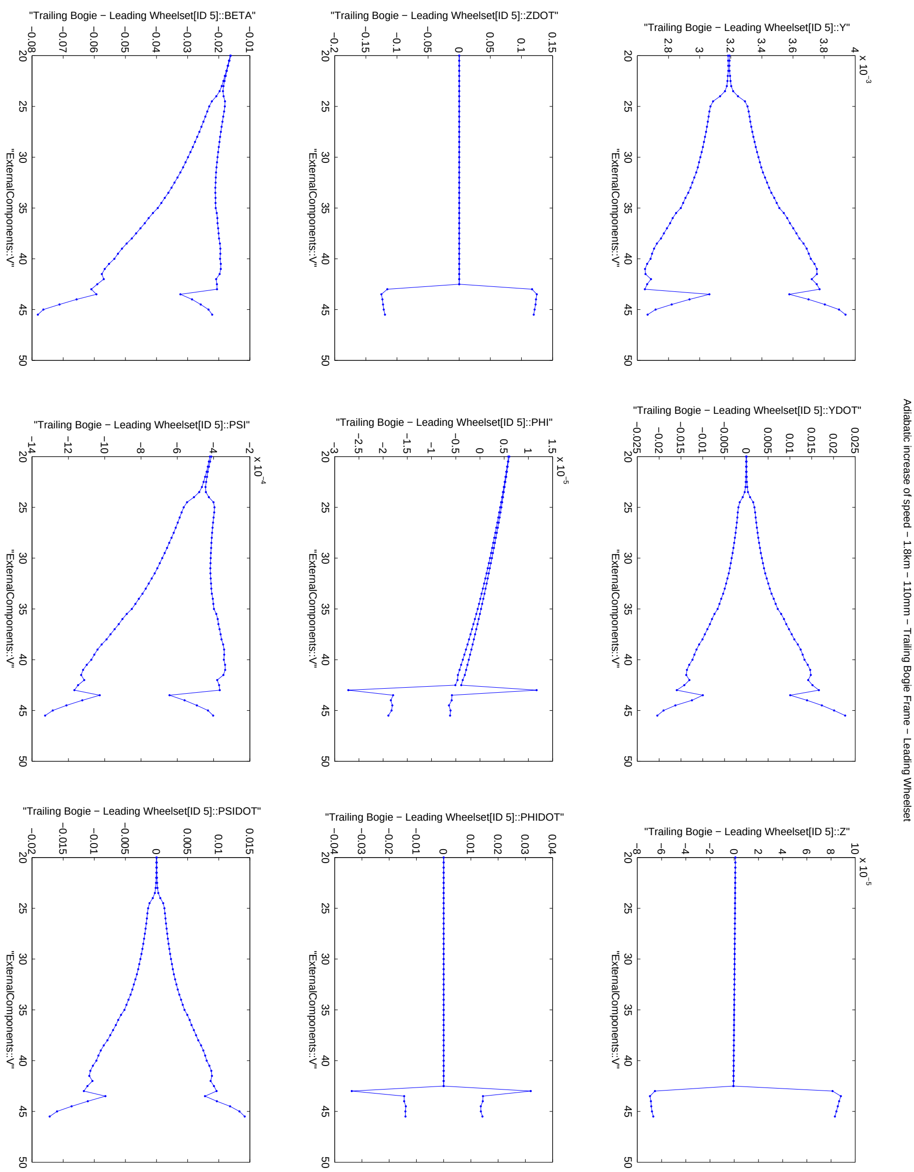


Figure F.61: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model does not consider precession forces.

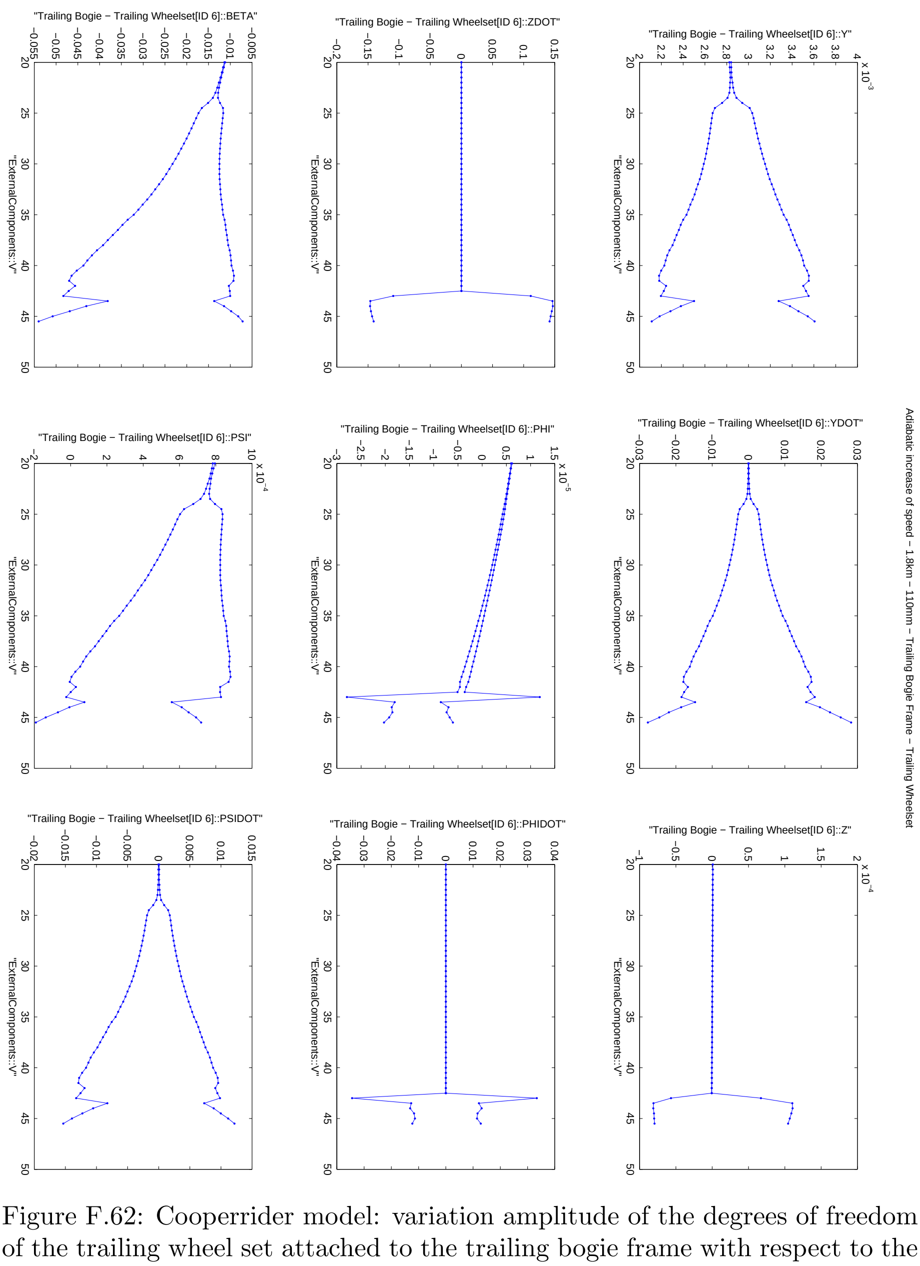


Figure F.62: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model does not consider precession forces.

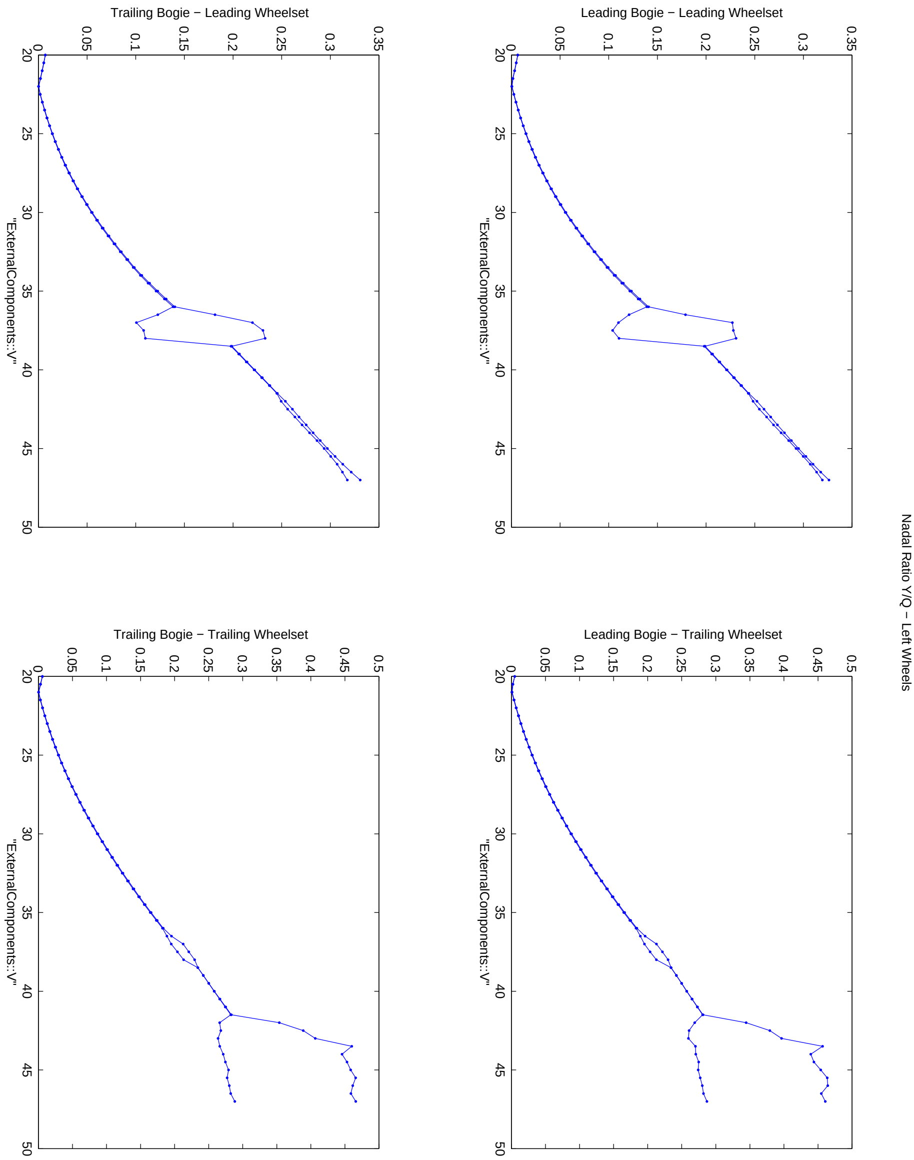


Figure F.63: Cooperrider model: Nadal's ratio on the external wheels as the speed is increased in a curved track with radius of 0.6km and cant of 110mm. The model does not consider precession forces.

Cooperrider: Curved track. With precession forces 0.6km

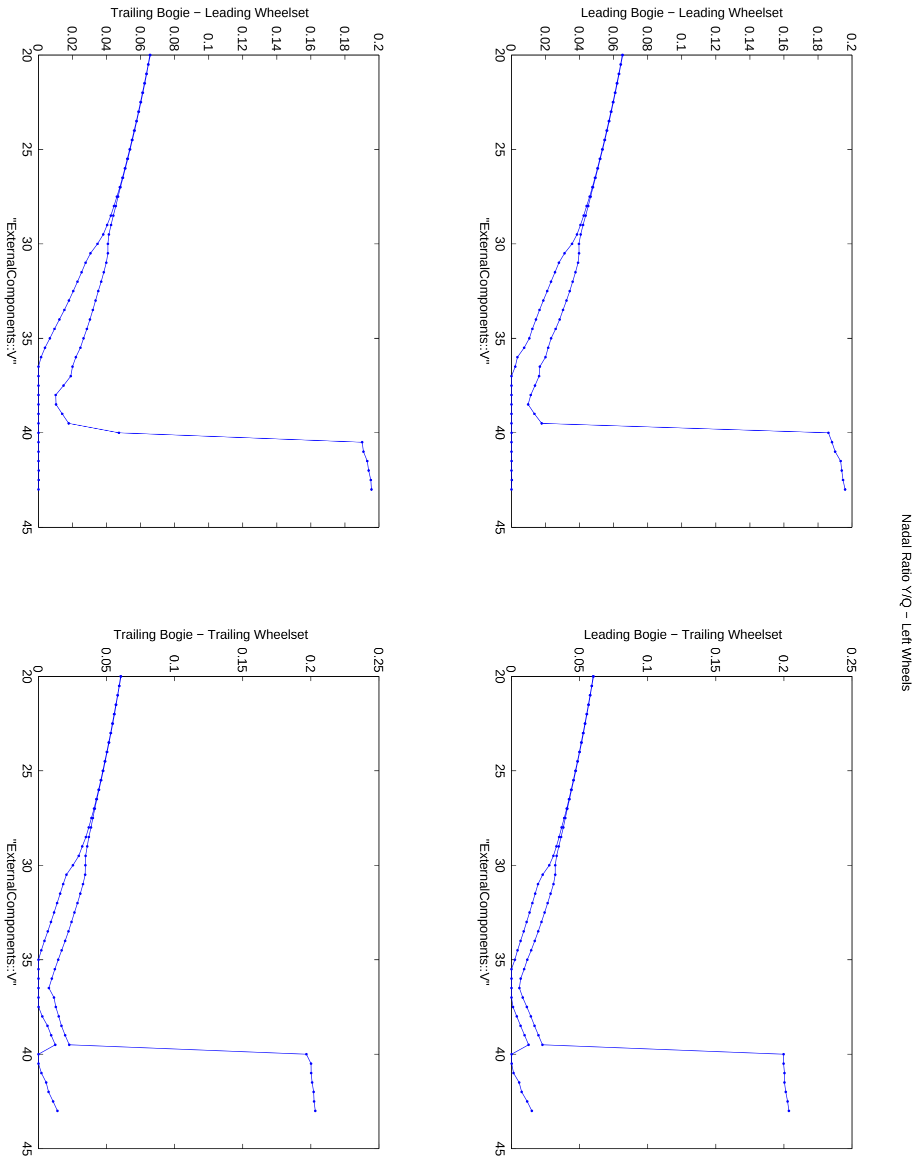


Figure F.64: Cooperrider model: Nadal's ratio on the external wheels as the speed is increased in a curved track with radius of 1.6km and cant of 110mm. The model does not consider precession forces.

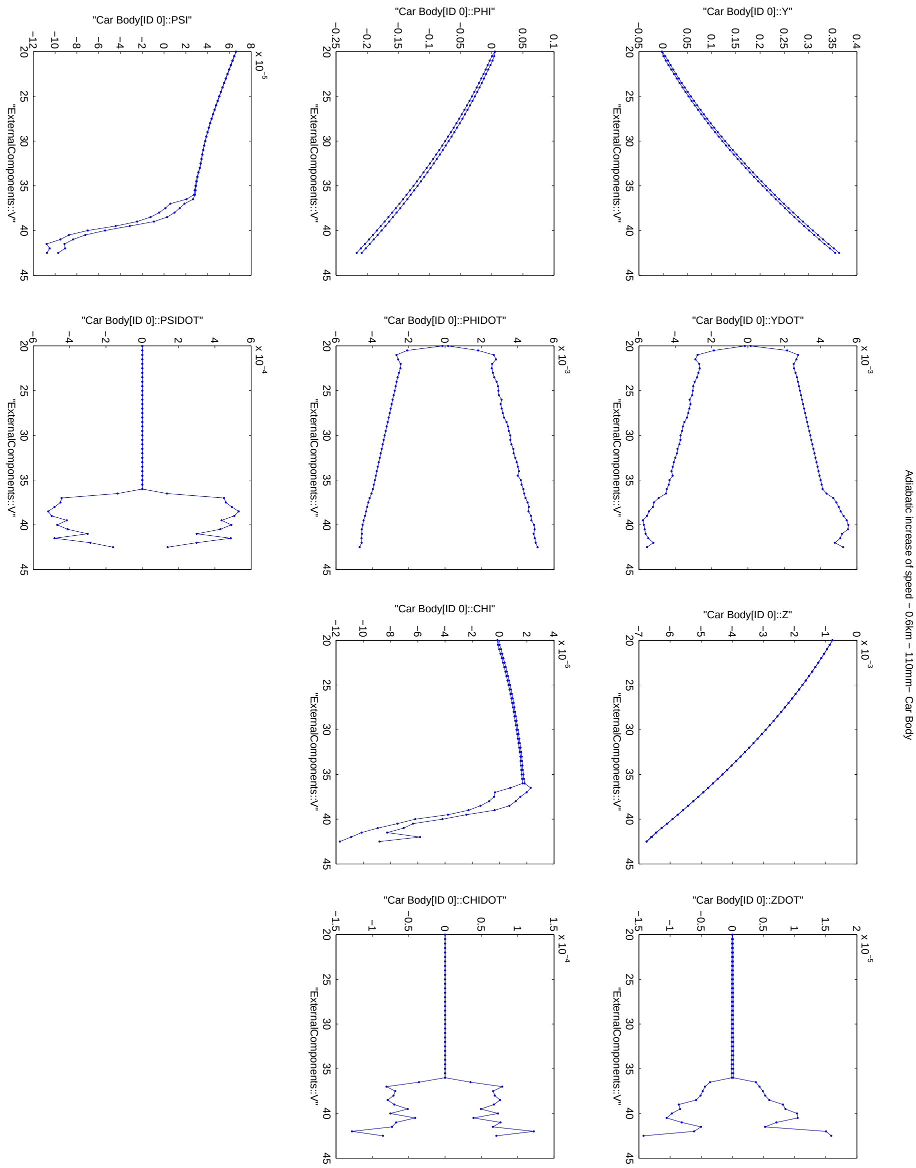


Figure F.65: Cooperrider model: variation amplitude of the degrees of freedom of the car body with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model considers precession forces.

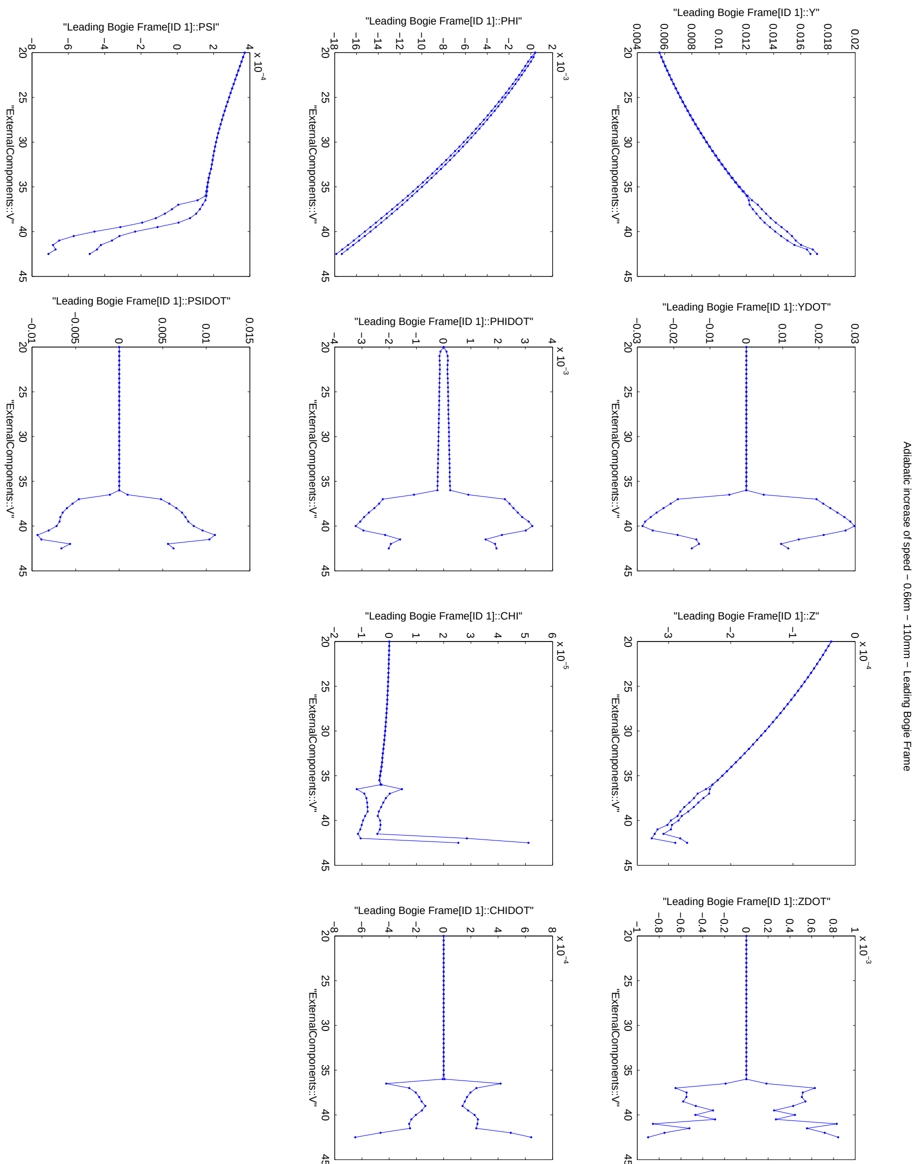


Figure F.66: Cooperrider model: variation amplitude of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model considers precession forces.

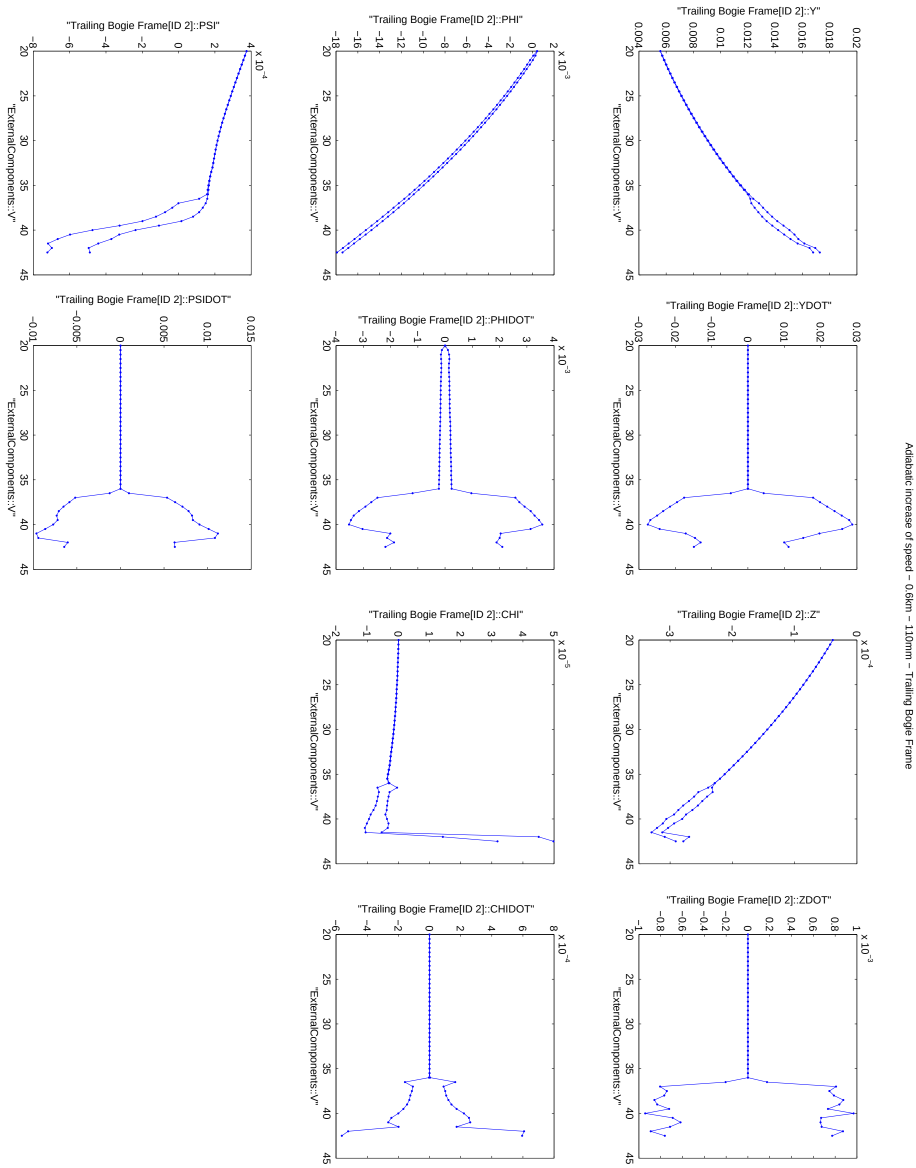


Figure F.67: Cooperrider model: variation amplitude of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model considers precession forces.

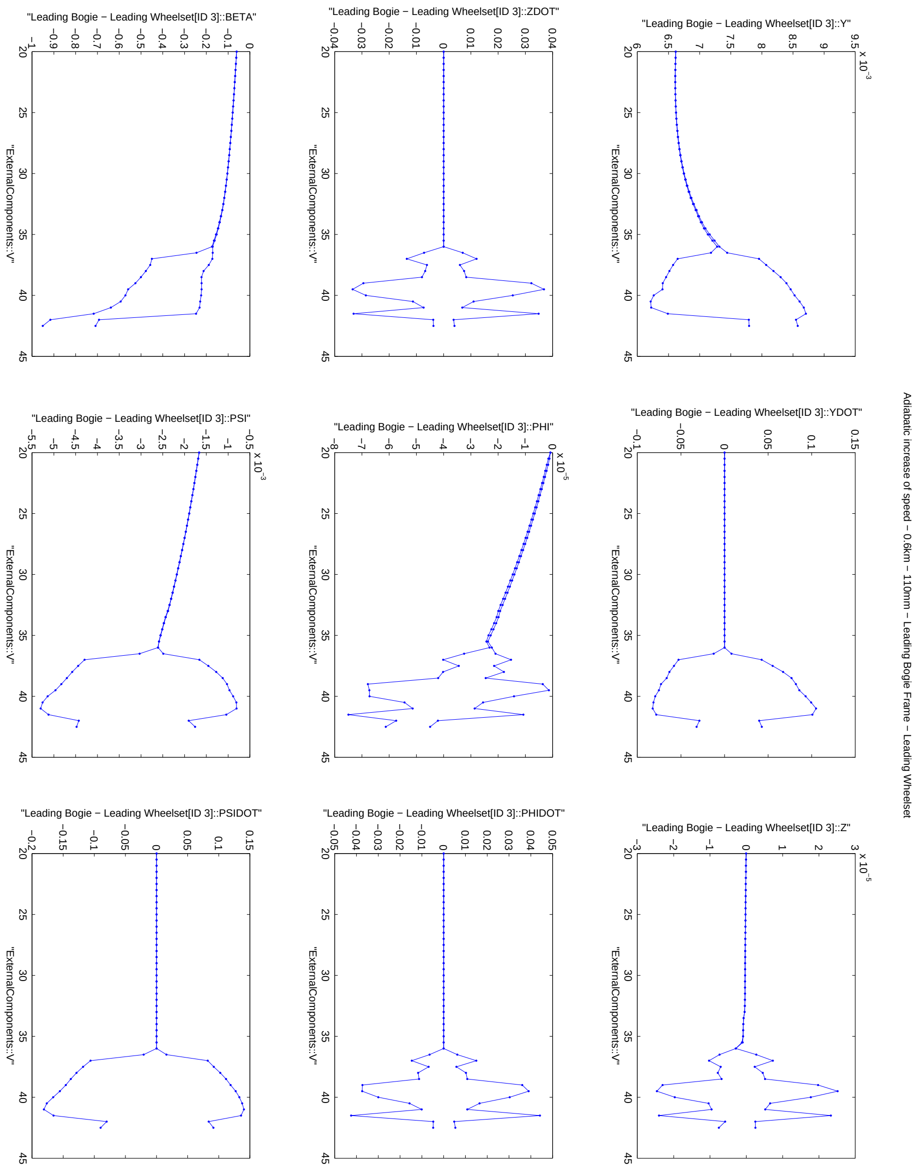


Figure F.68: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model considers precession forces.

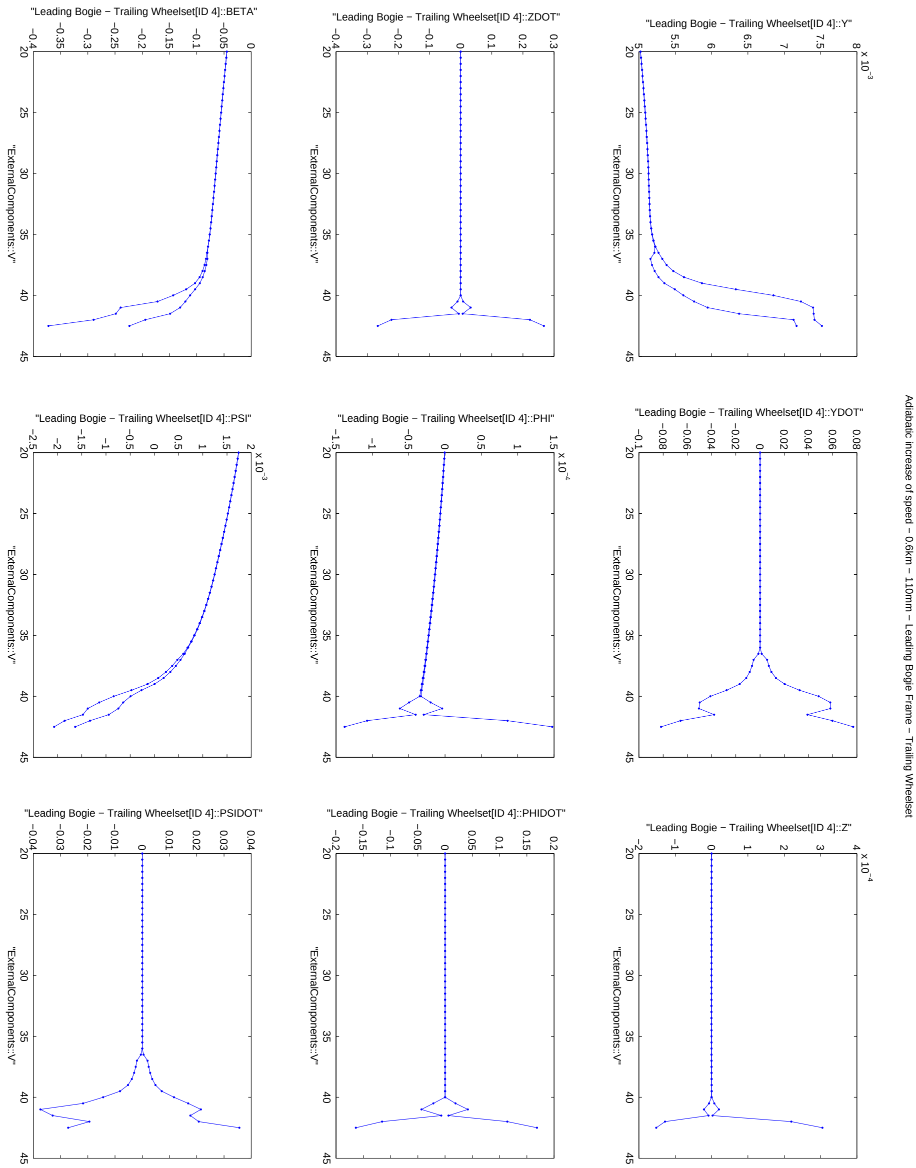


Figure F.69: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model considers precession forces.

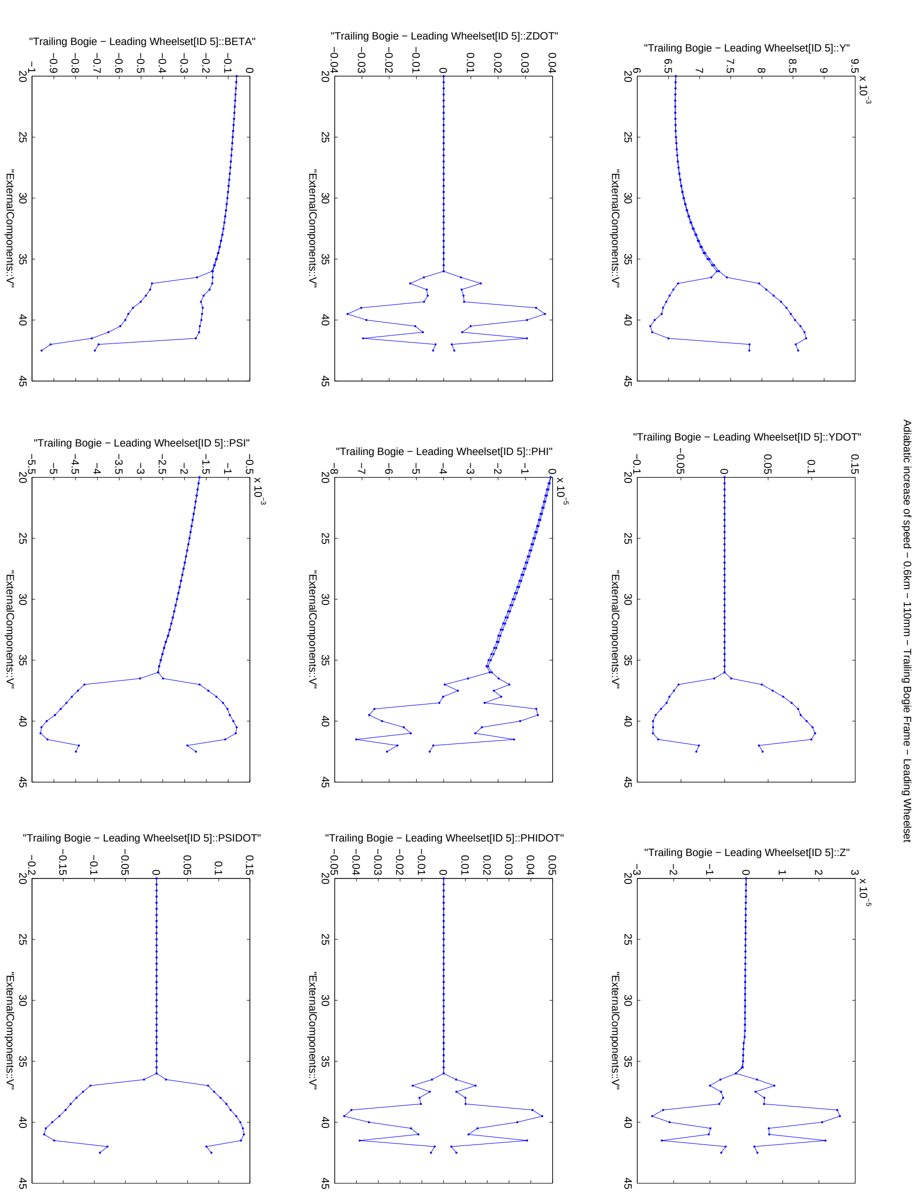


Figure F.70: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model considers precession forces.

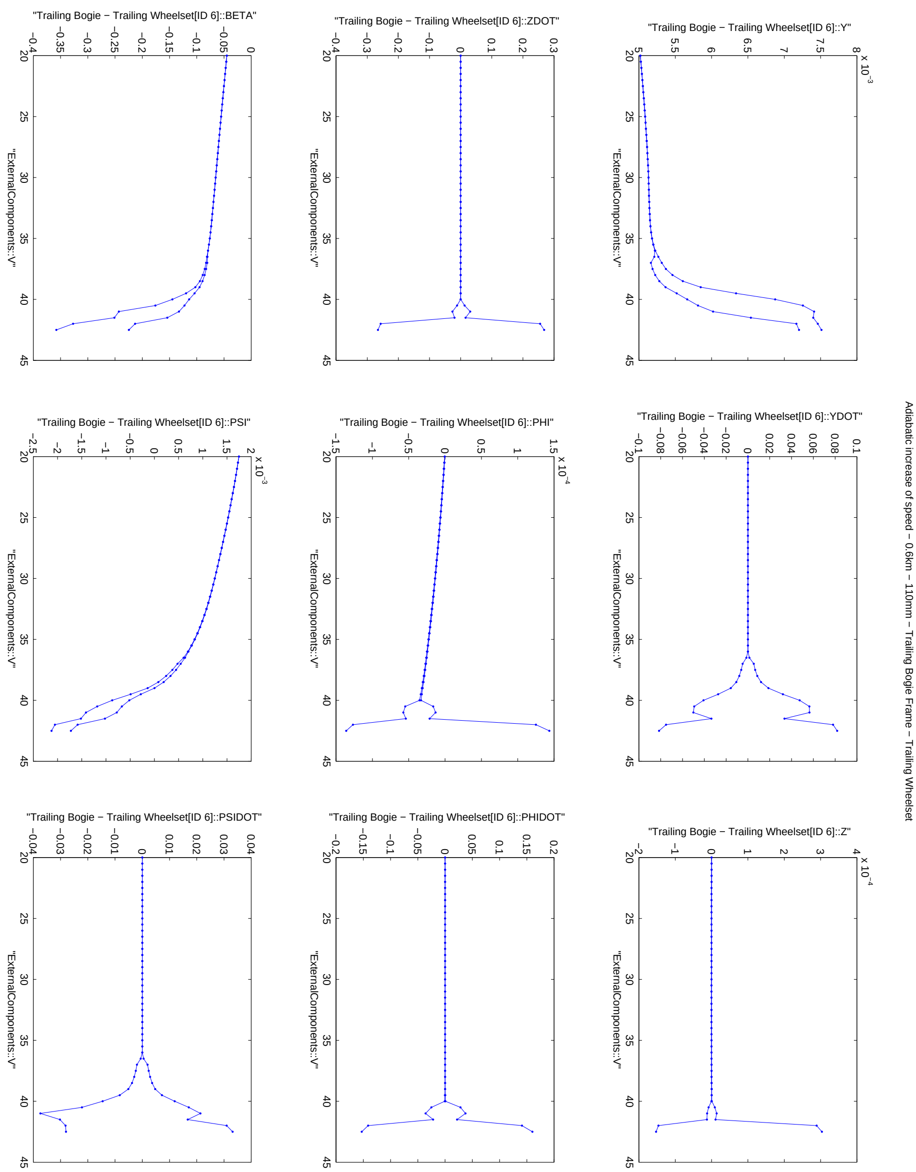


Figure F.71: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 600m and cant of 110mm. The model considers precession forces.

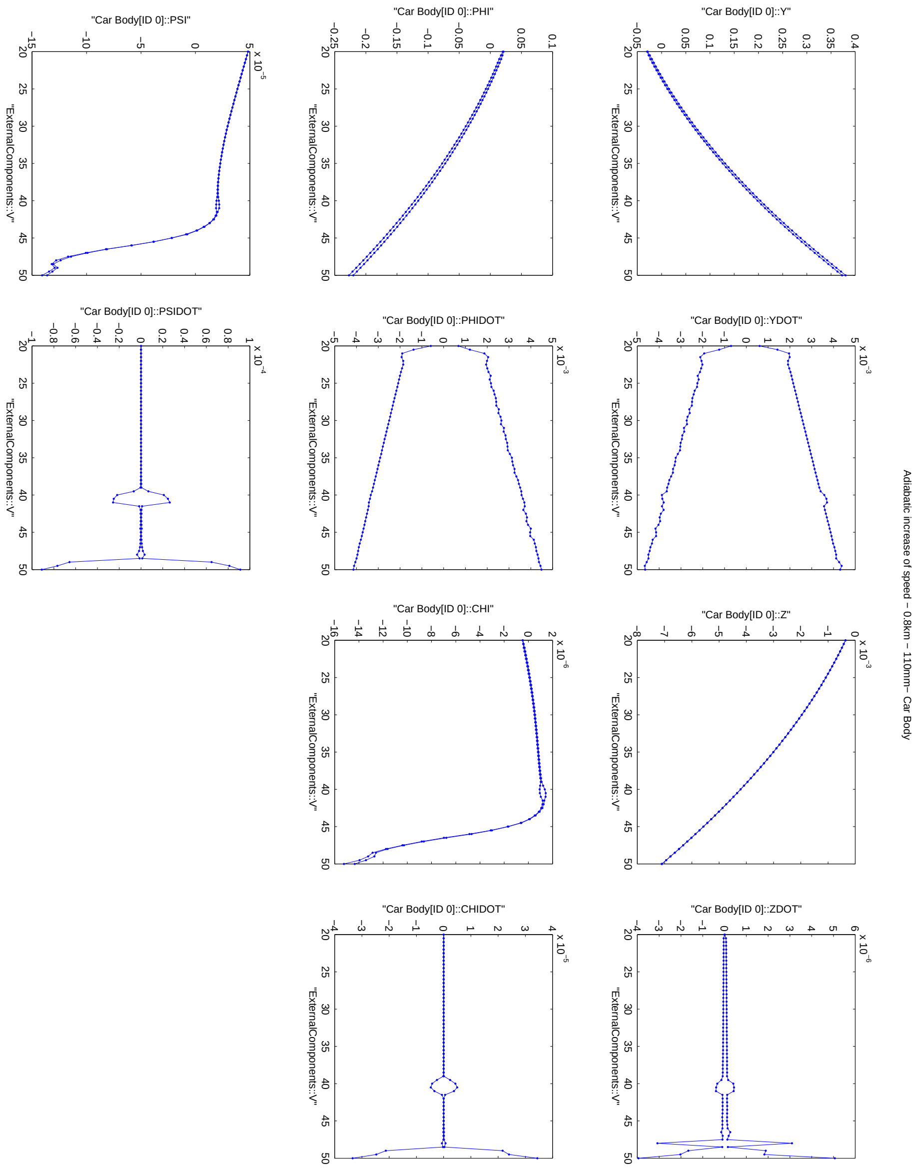


Figure F.72: Cooperrider model: variation amplitude of the degrees of freedom of the car body with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model considers precession forces.

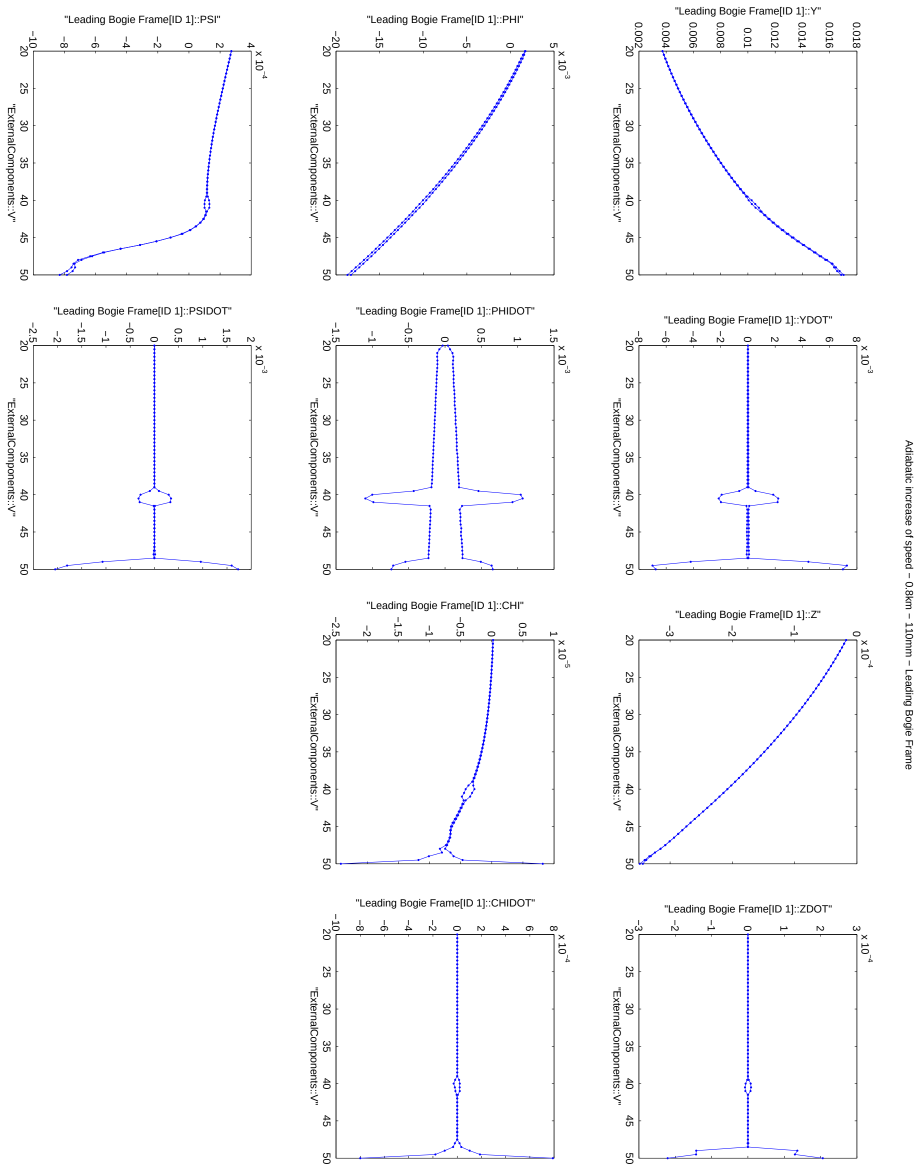


Figure F.73: Cooperrider model: variation amplitude of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model considers precession forces.

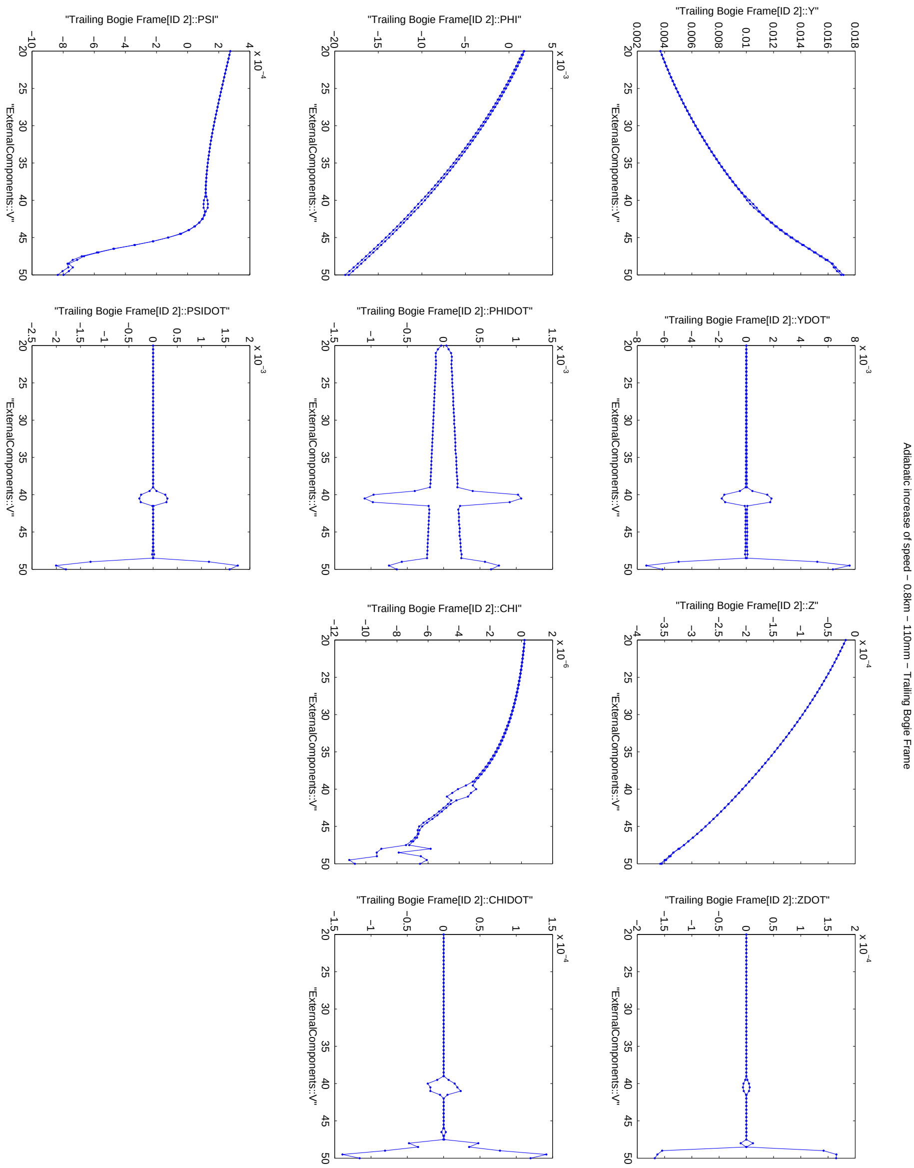


Figure F.74: Cooperrider model: variation amplitude of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model considers precession forces.

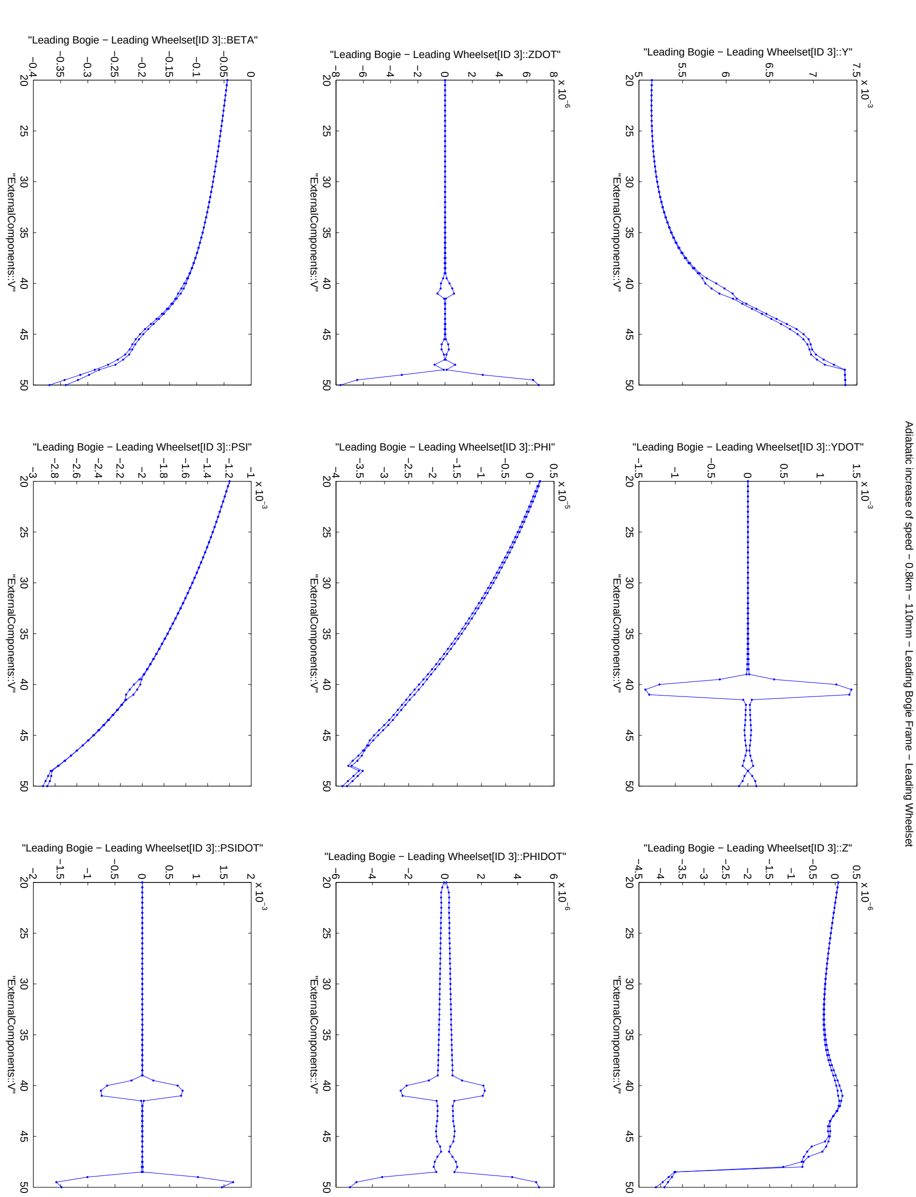


Figure F.75: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model considers precession forces.

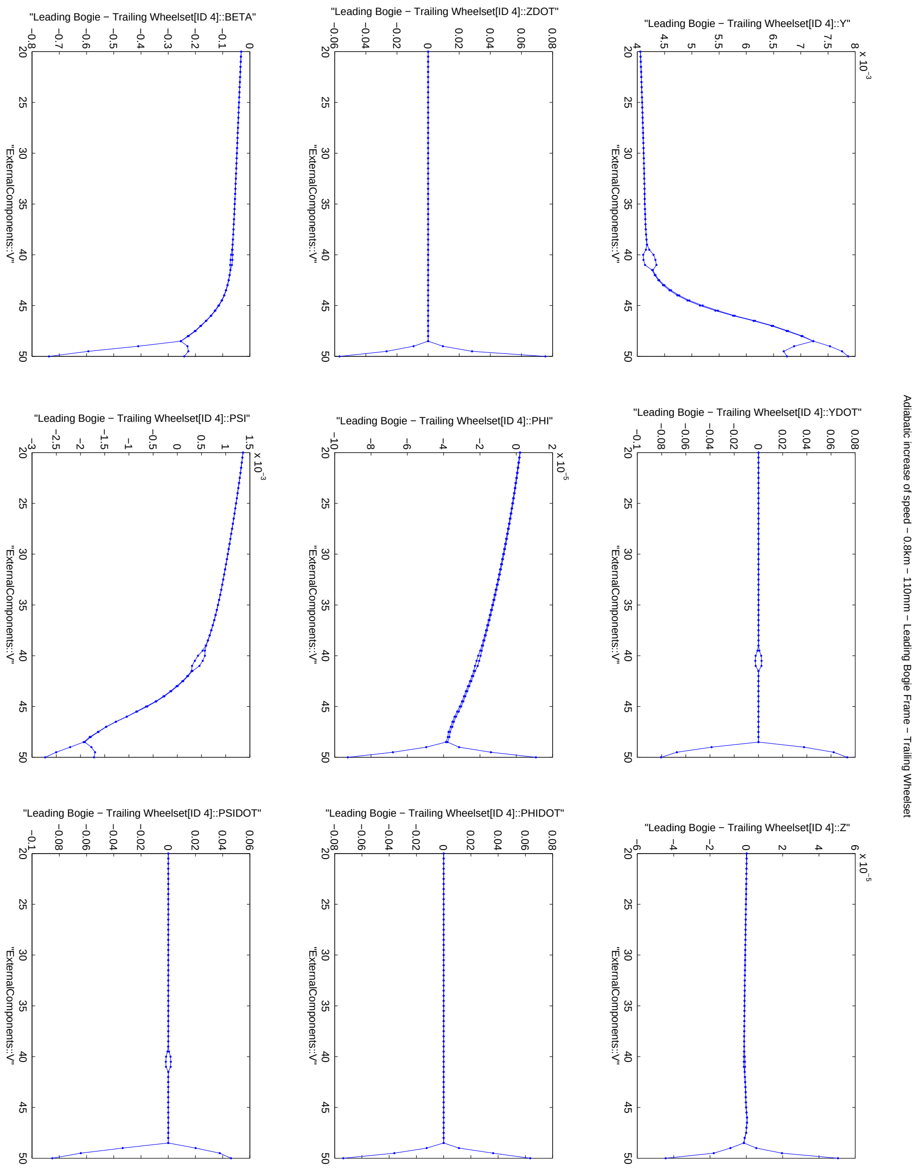


Figure F.76: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model considers precession forces.

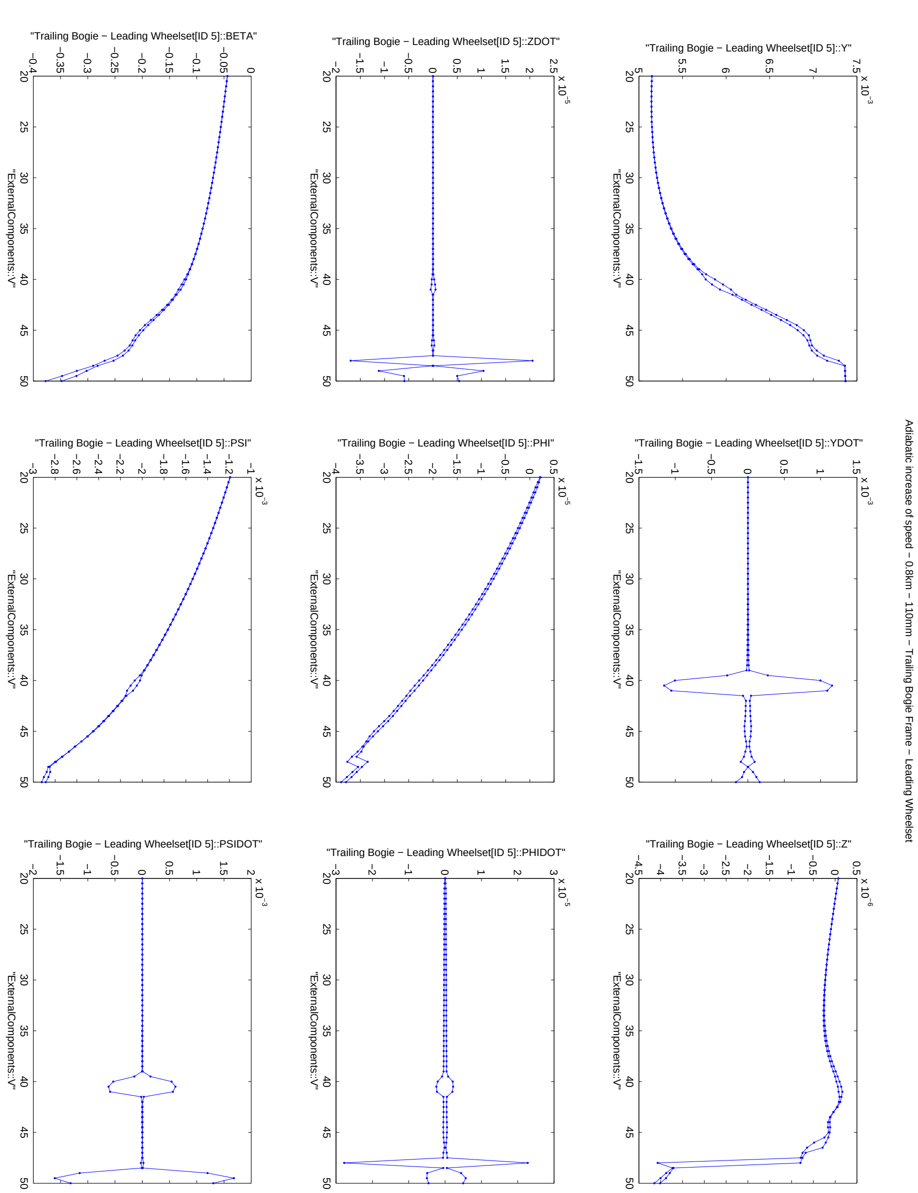


Figure F.77: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model considers precession forces.

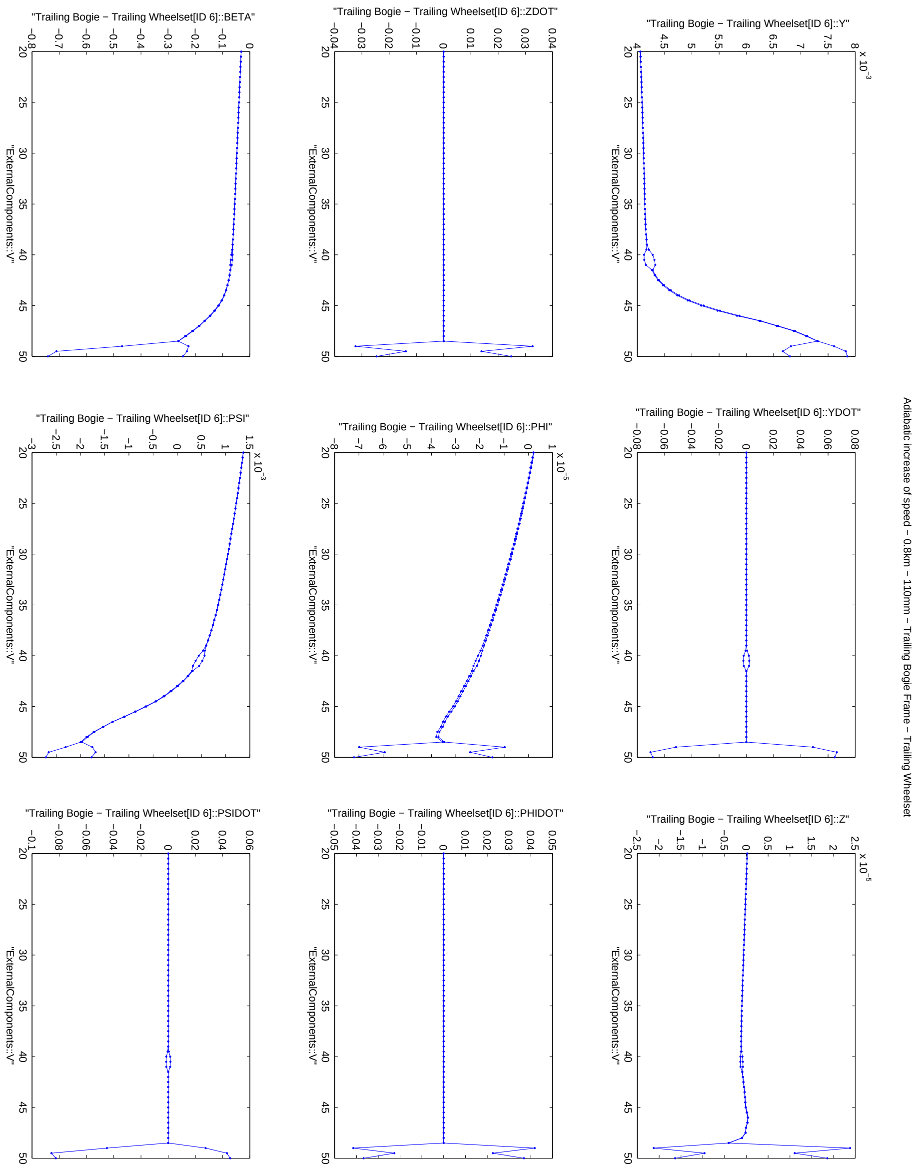


Figure F.78: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 800m and cant of 110mm. The model considers precession forces.

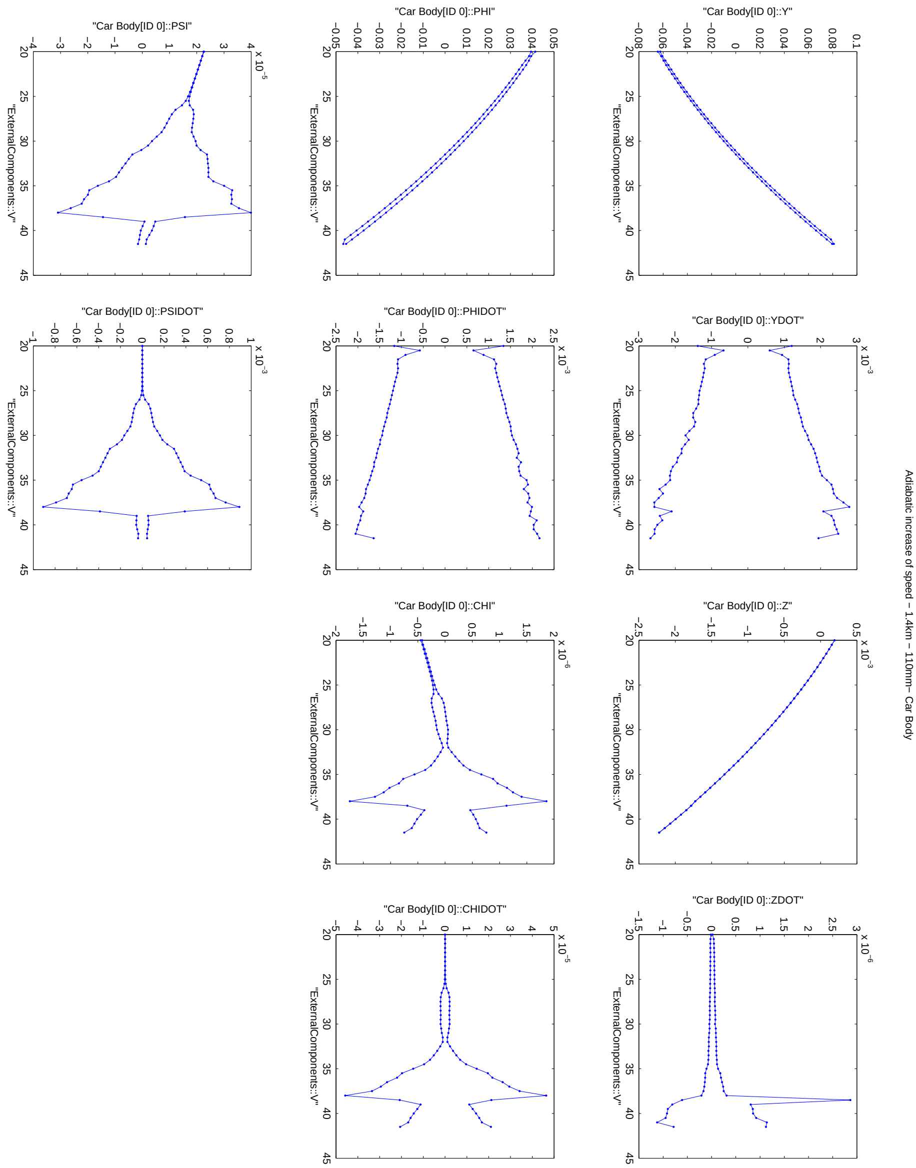


Figure F.79: Cooperrider model: variation amplitude of the degrees of freedom of the car body with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model considers precession forces.

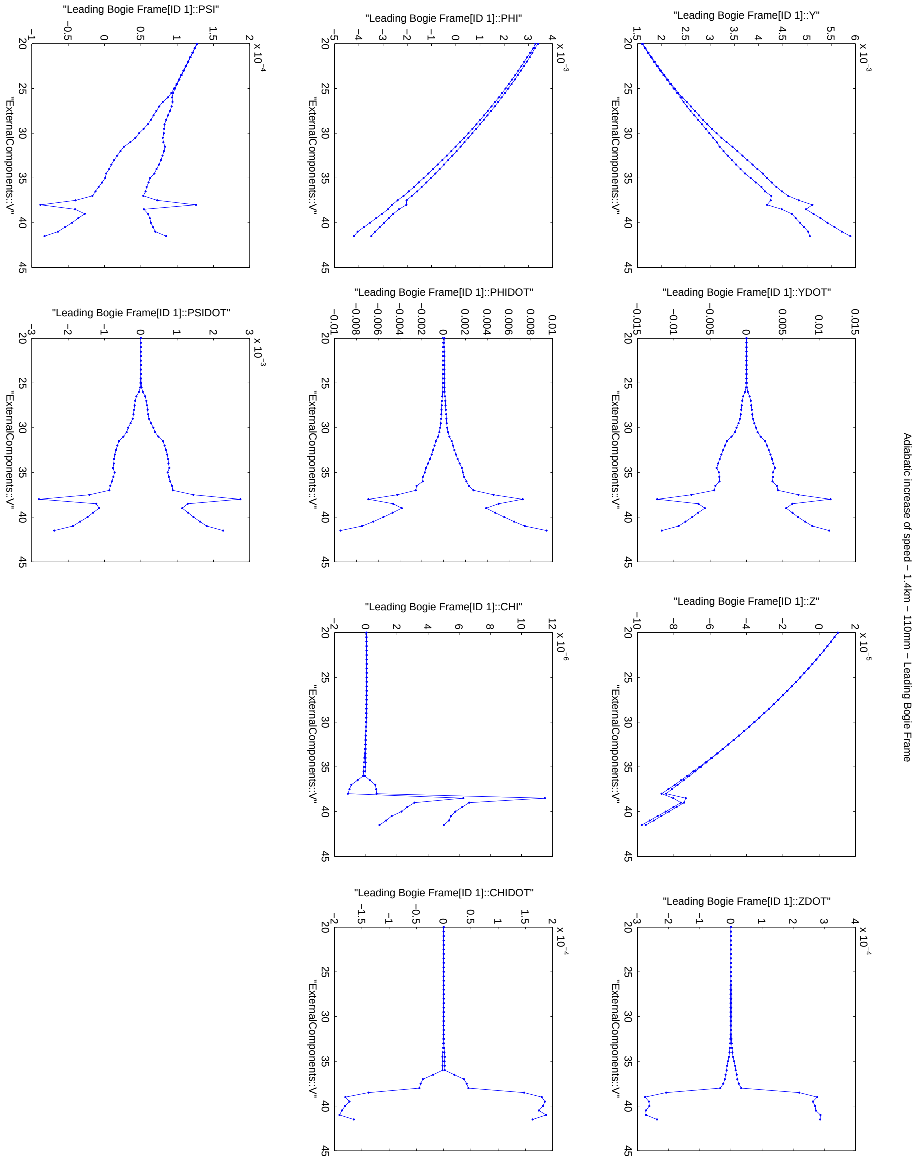


Figure F.80: Cooperrider model: variation amplitude of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model considers precession forces.

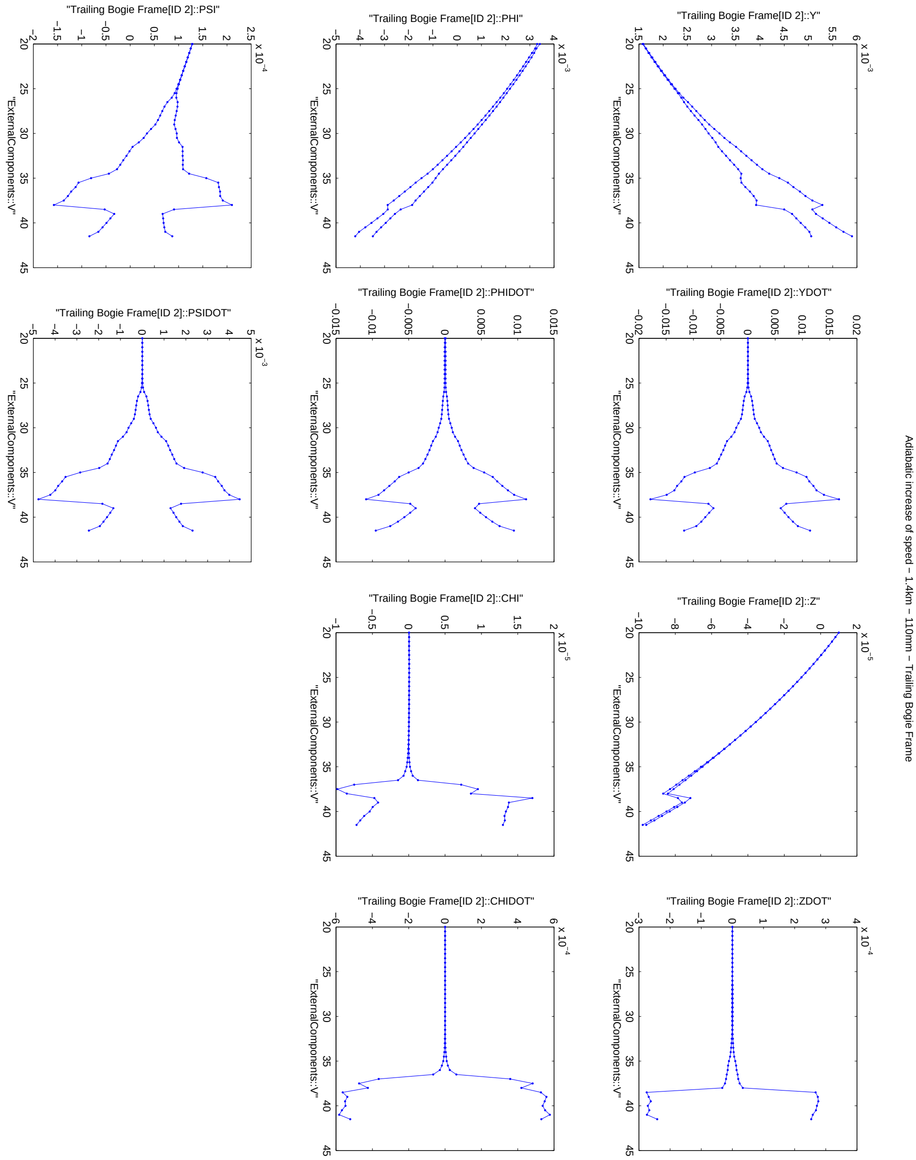


Figure F.81: Cooperrider model: variation amplitude of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model considers precession forces.

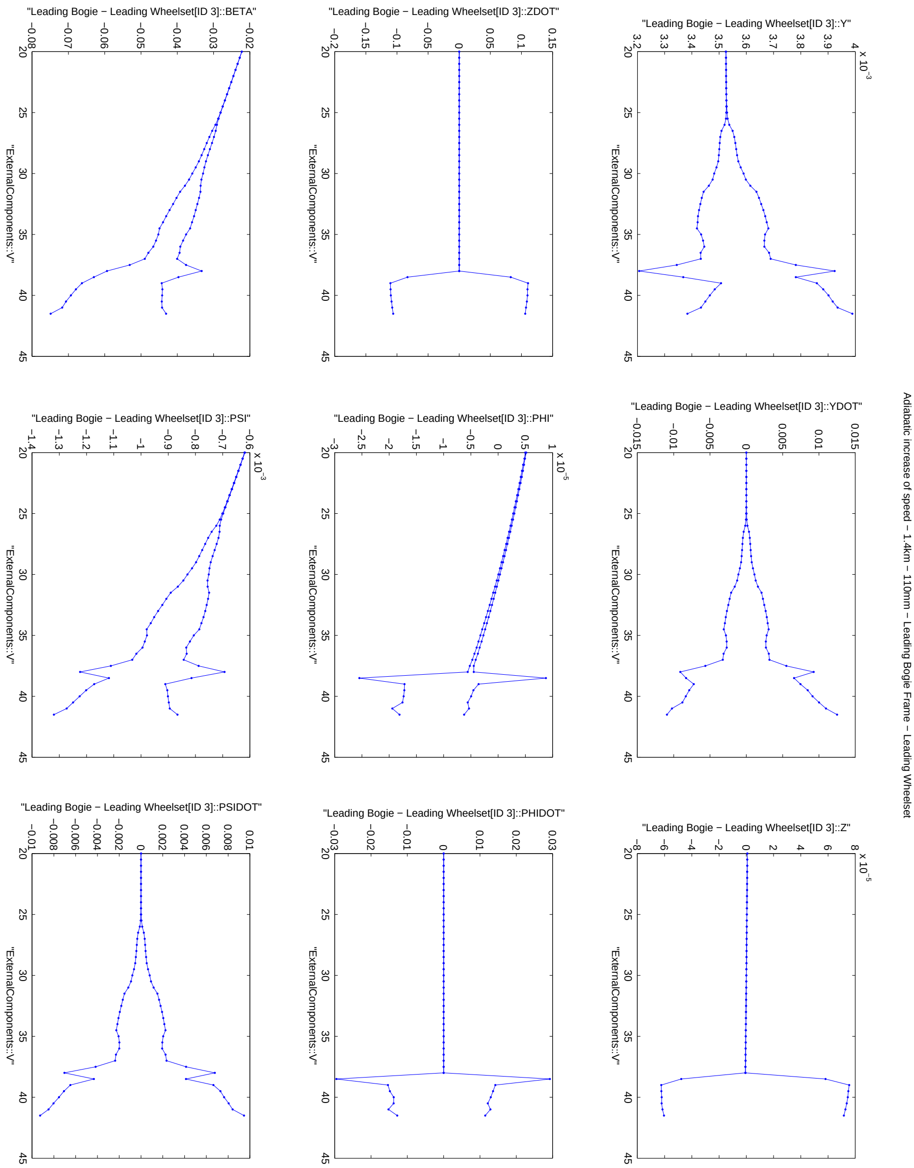


Figure F.82: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model considers precession forces.

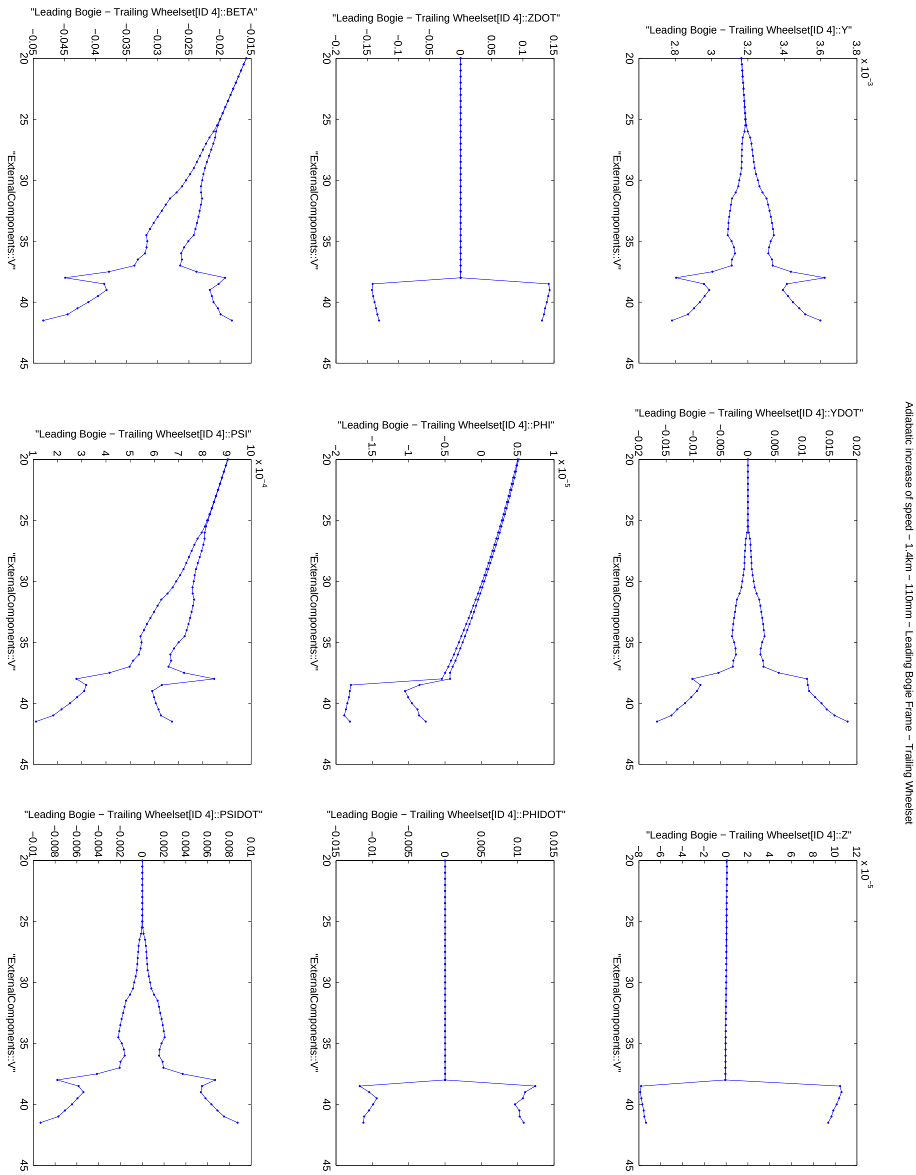


Figure F.83: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model considers precession forces.

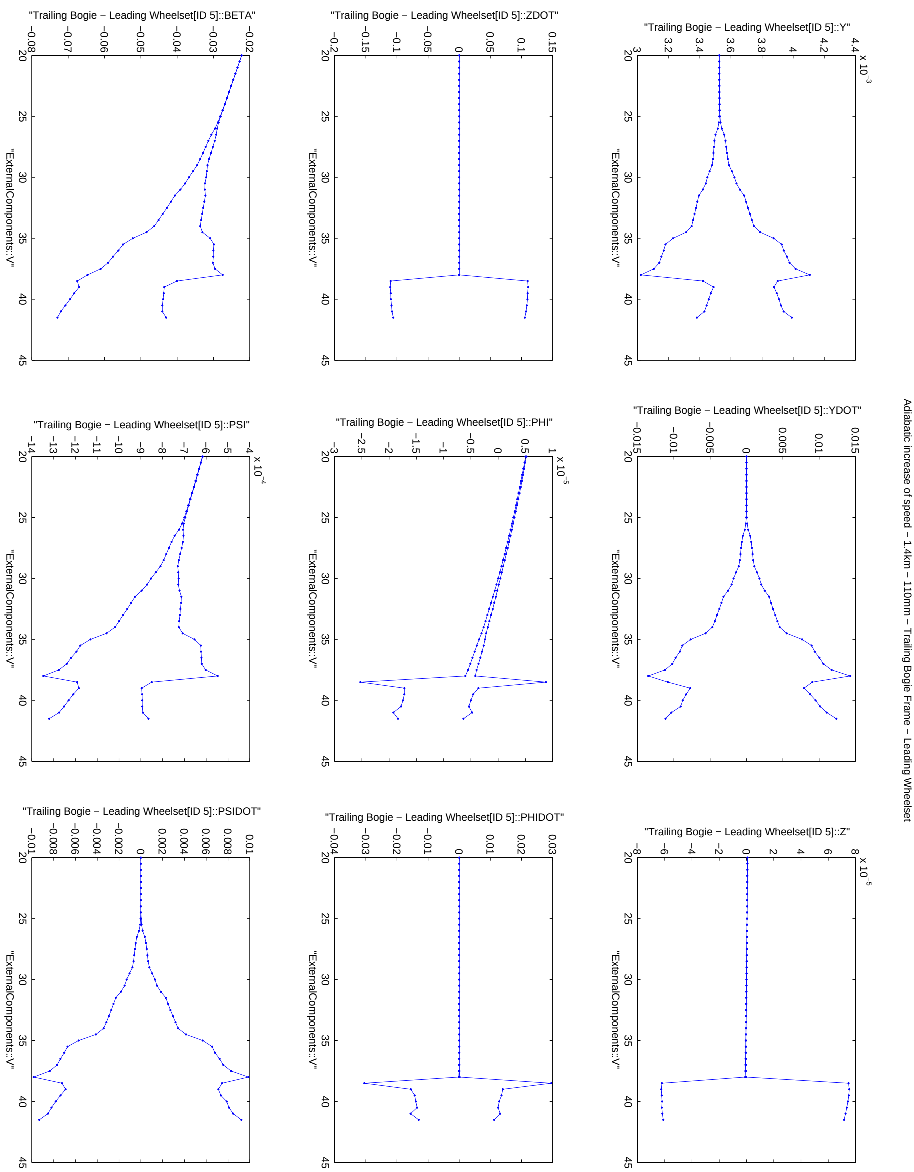


Figure F.84: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model considers precession forces.

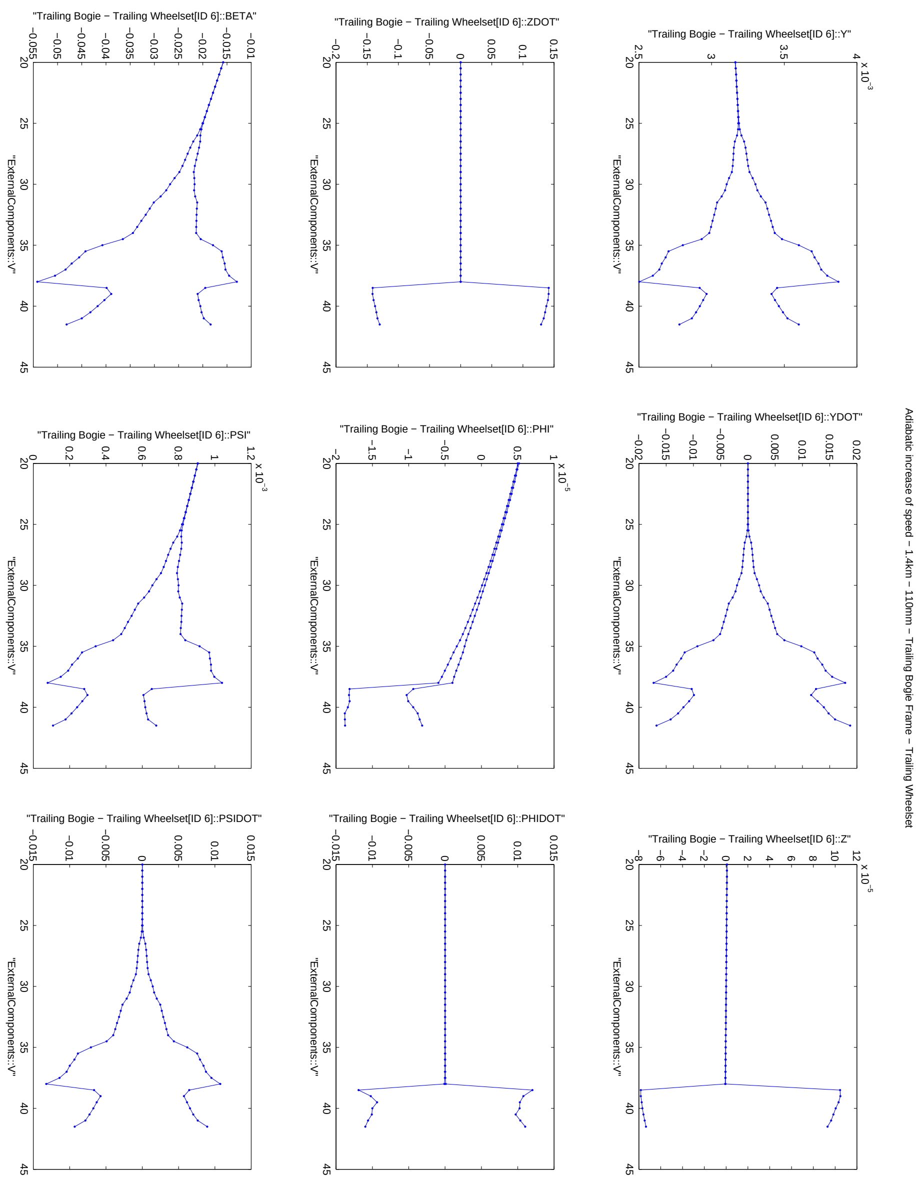


Figure F.85: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.4km and cant of 110mm. The model considers precession forces.

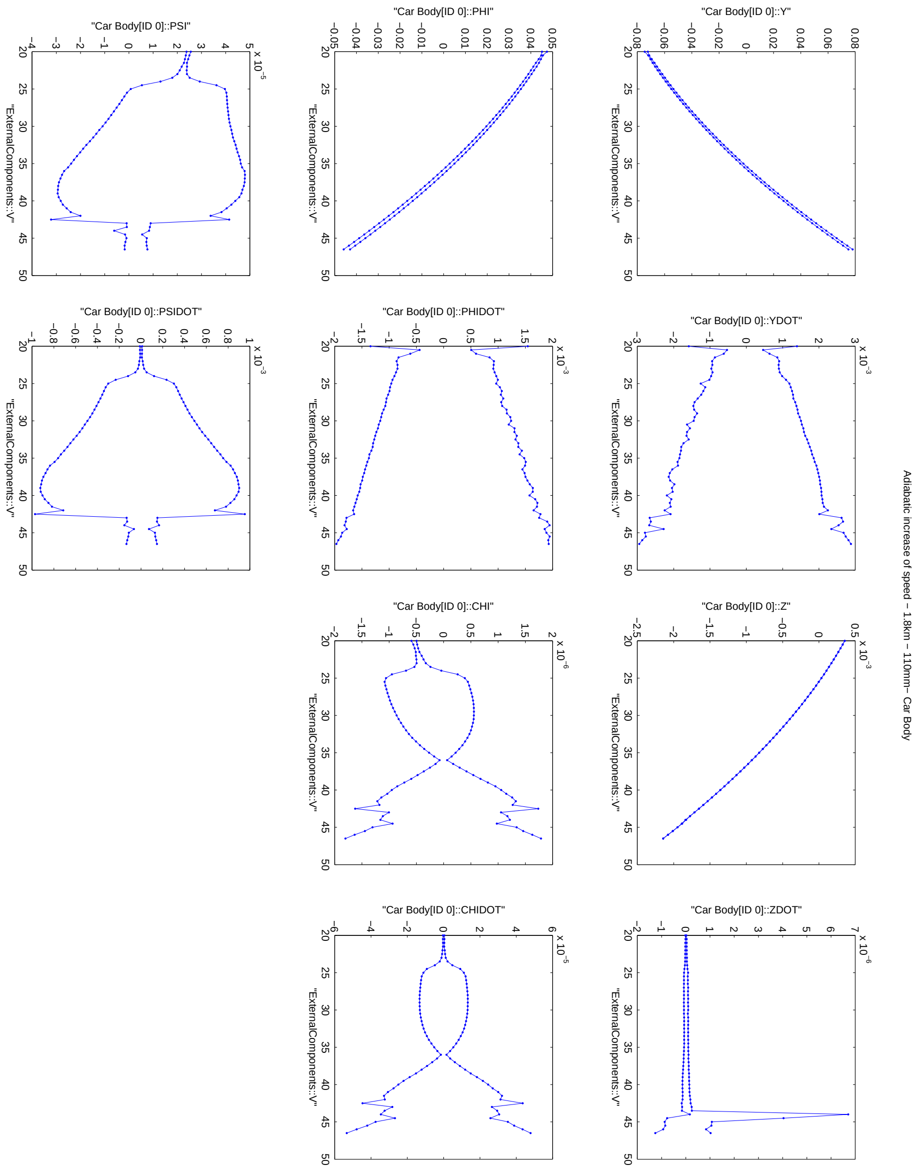


Figure F.86: Cooperrider model: variation amplitude of the degrees of freedom of the car body with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model considers precession forces.

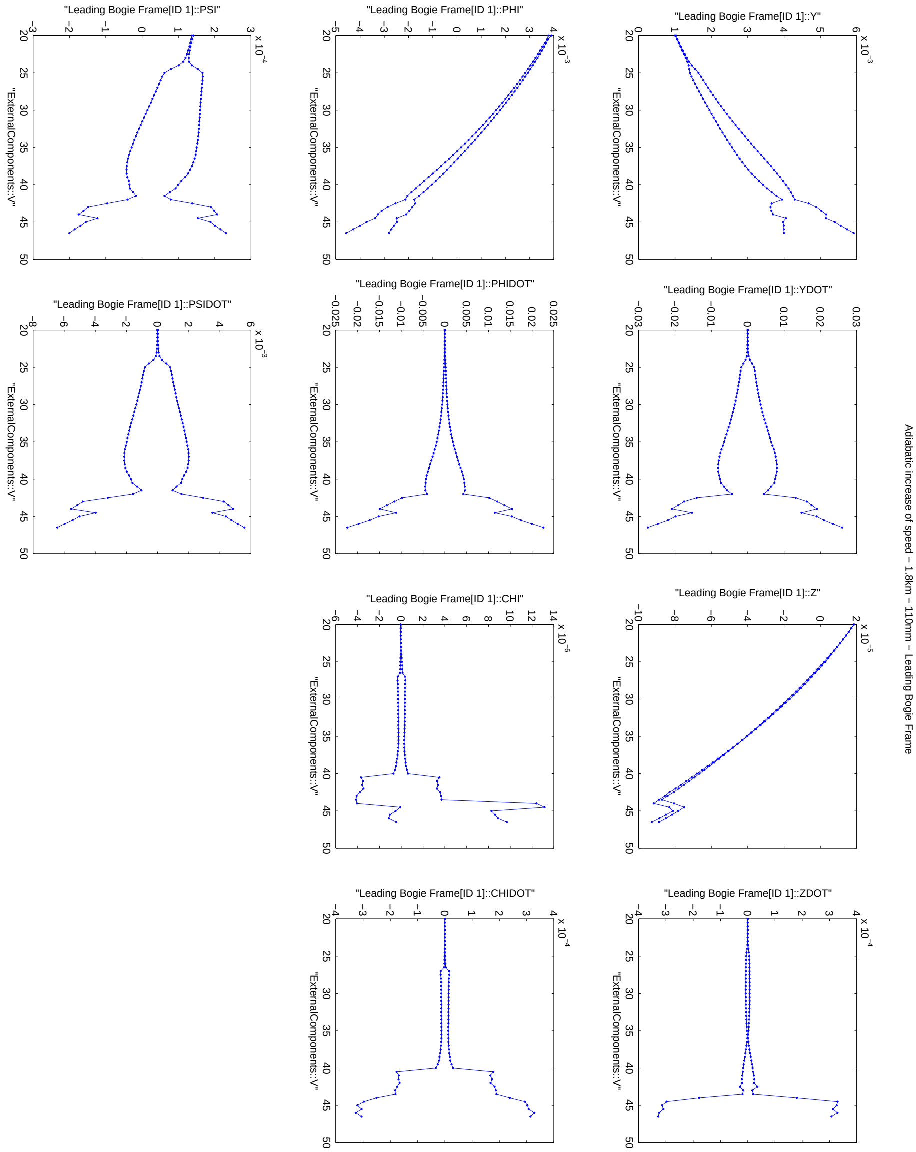


Figure F.87: Cooperrider model: variation amplitude of the degrees of freedom of the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model considers precession forces.

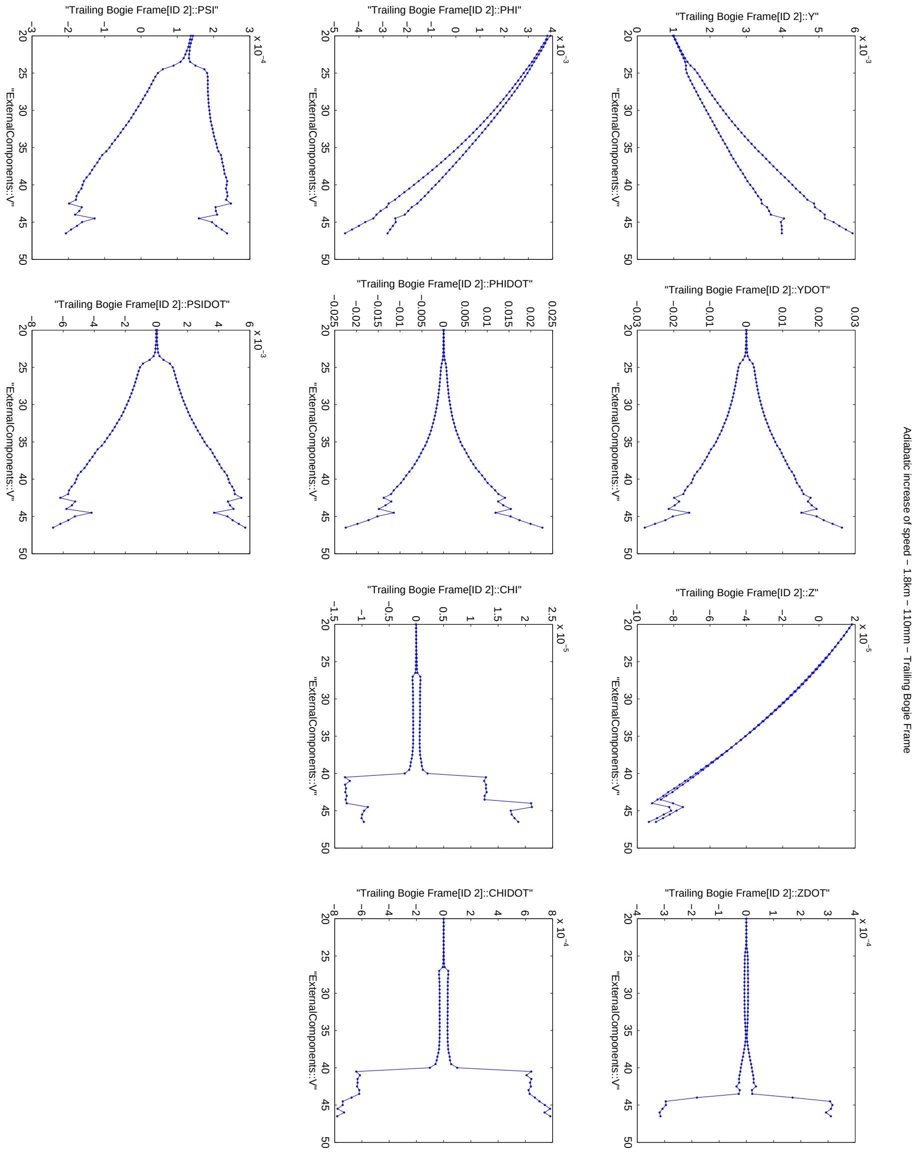


Figure F.88: Cooperrider model: variation amplitude of the degrees of freedom of the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model considers precession forces.

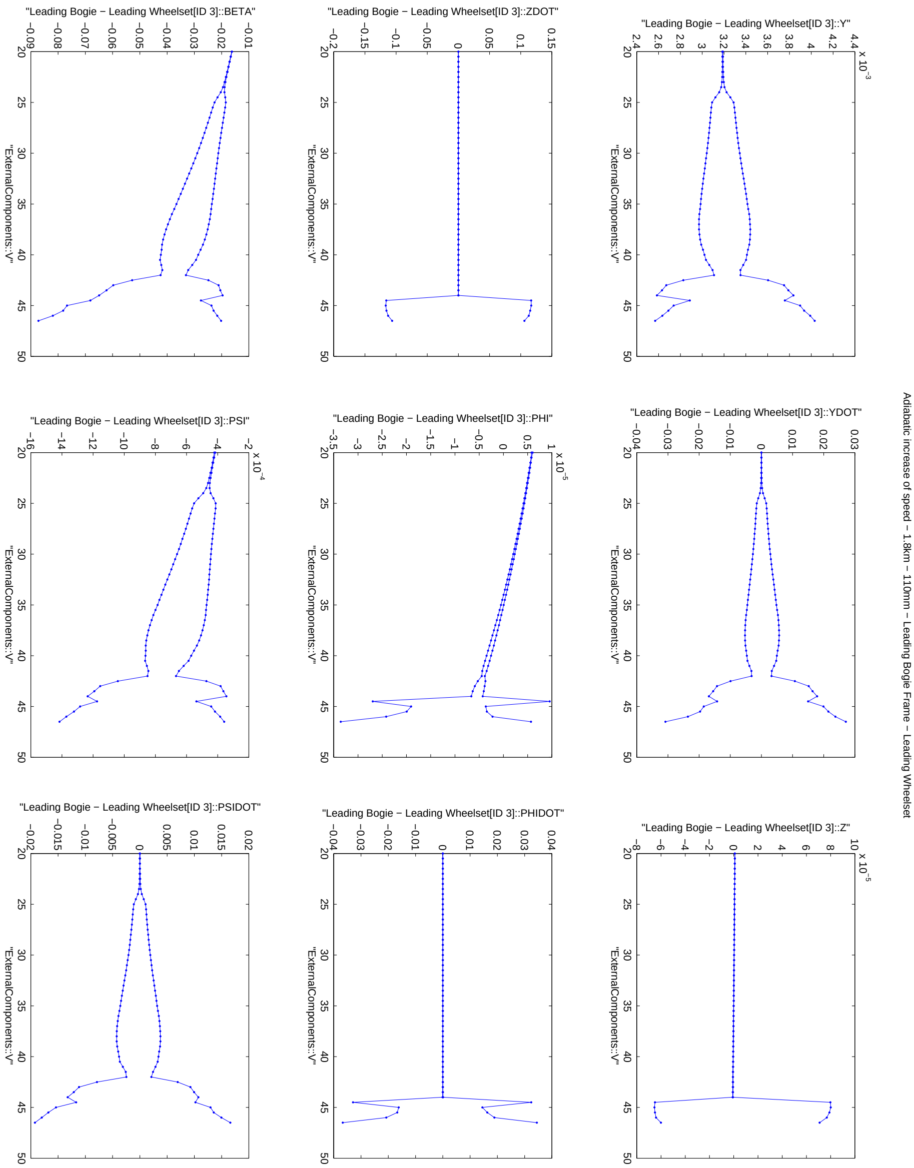


Figure F.89: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model considers precession forces.

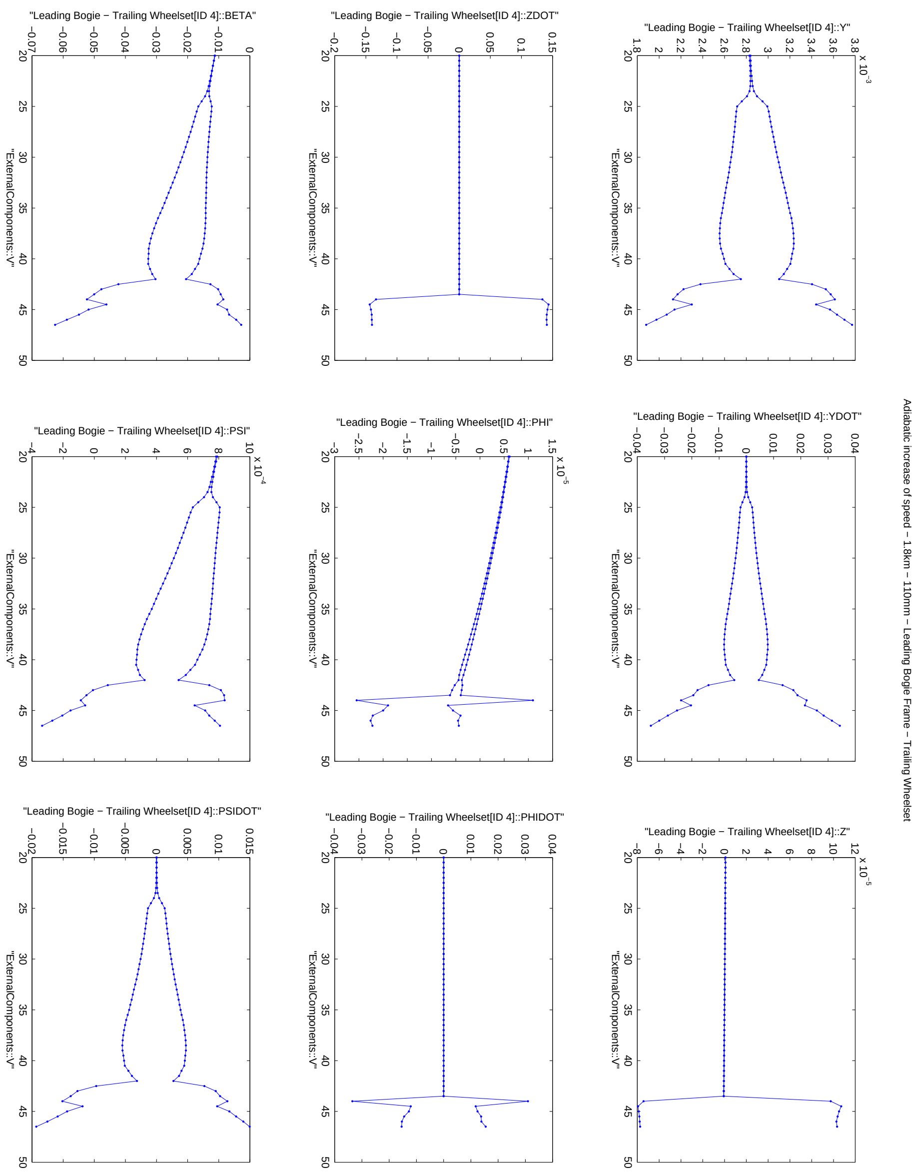


Figure F.90: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the leading bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model considers precession forces.

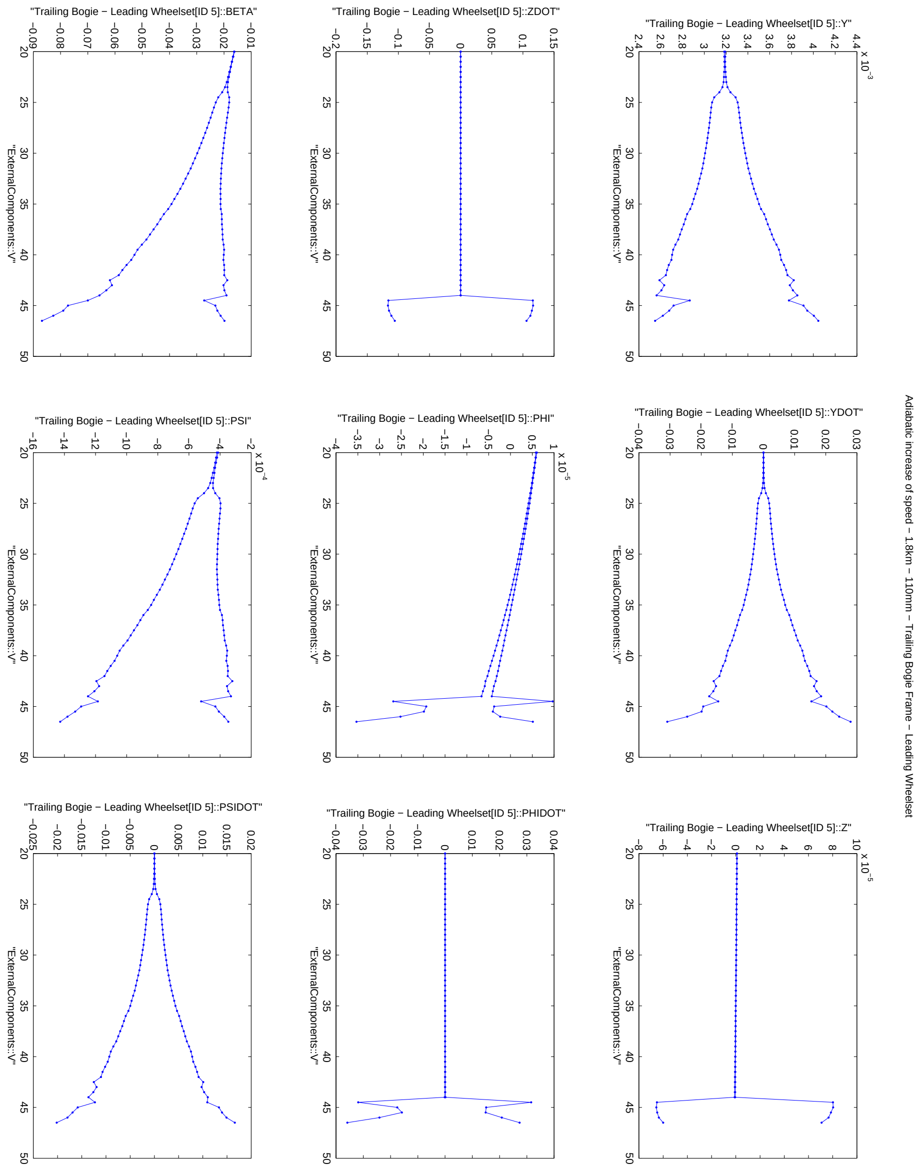


Figure F.91: Cooperrider model: variation amplitude of the degrees of freedom of the leading wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model considers precession forces.

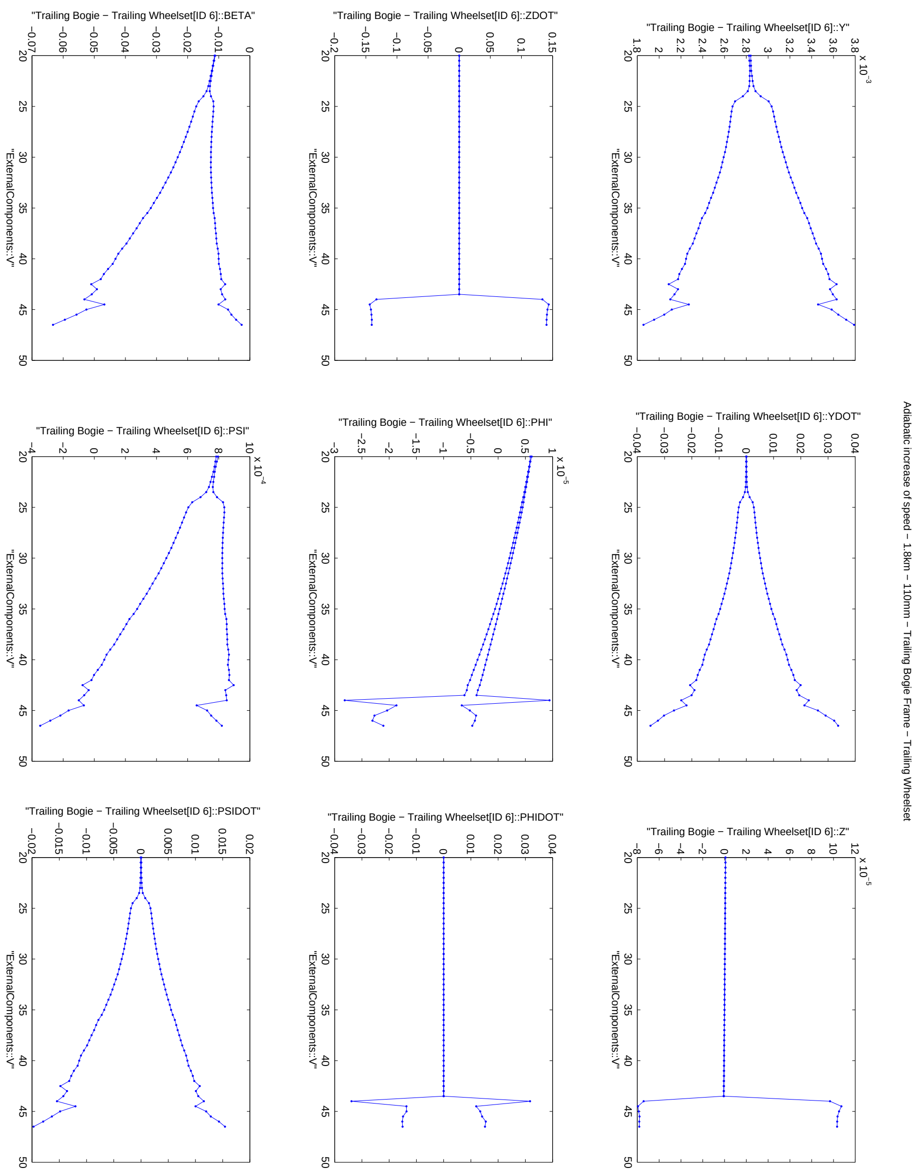


Figure F.92: Cooperrider model: variation amplitude of the degrees of freedom of the trailing wheel set attached to the trailing bogie frame with respect to the increasing speed. The model is running on curved track with radius of 1.8km and cant of 110mm. The model considers precession forces.

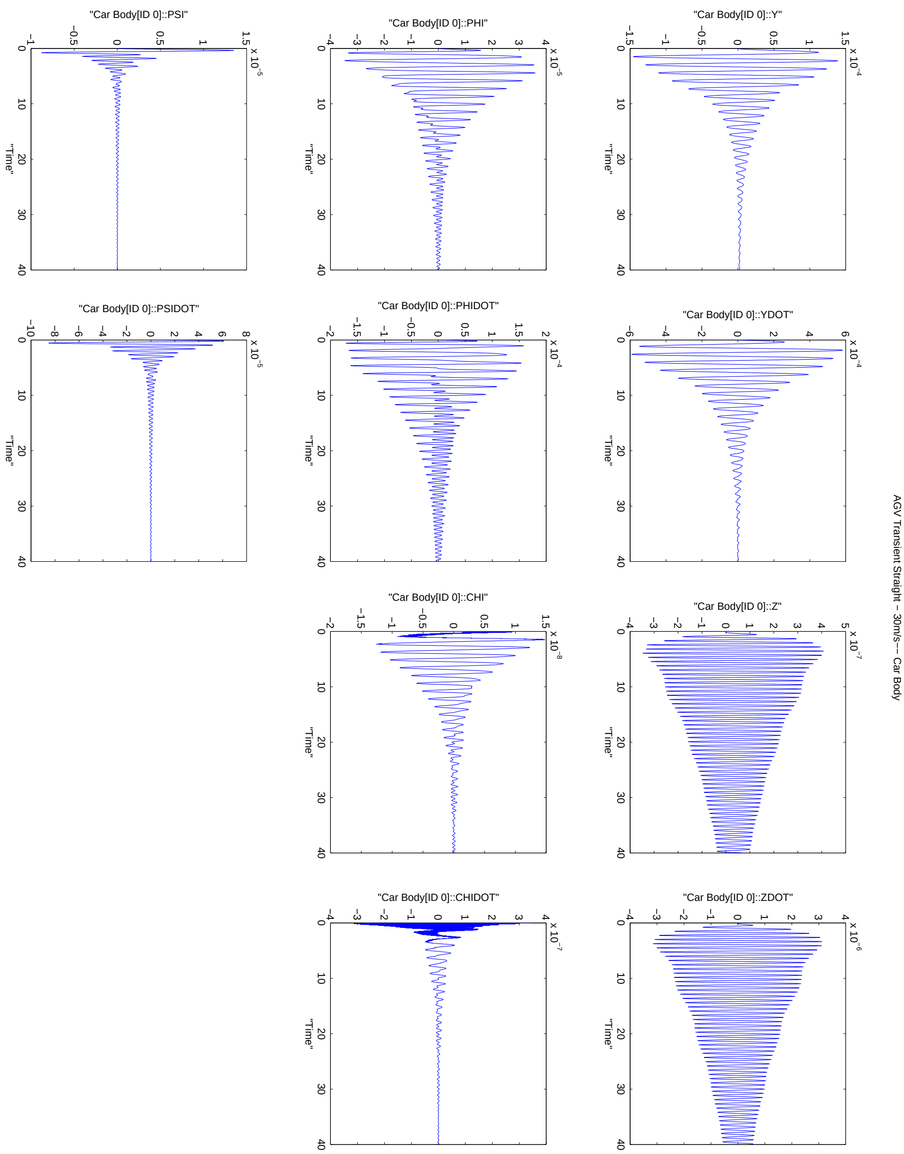


Figure F.93: AGV model: Dynamics of the car body running at 30m/s on straight track.

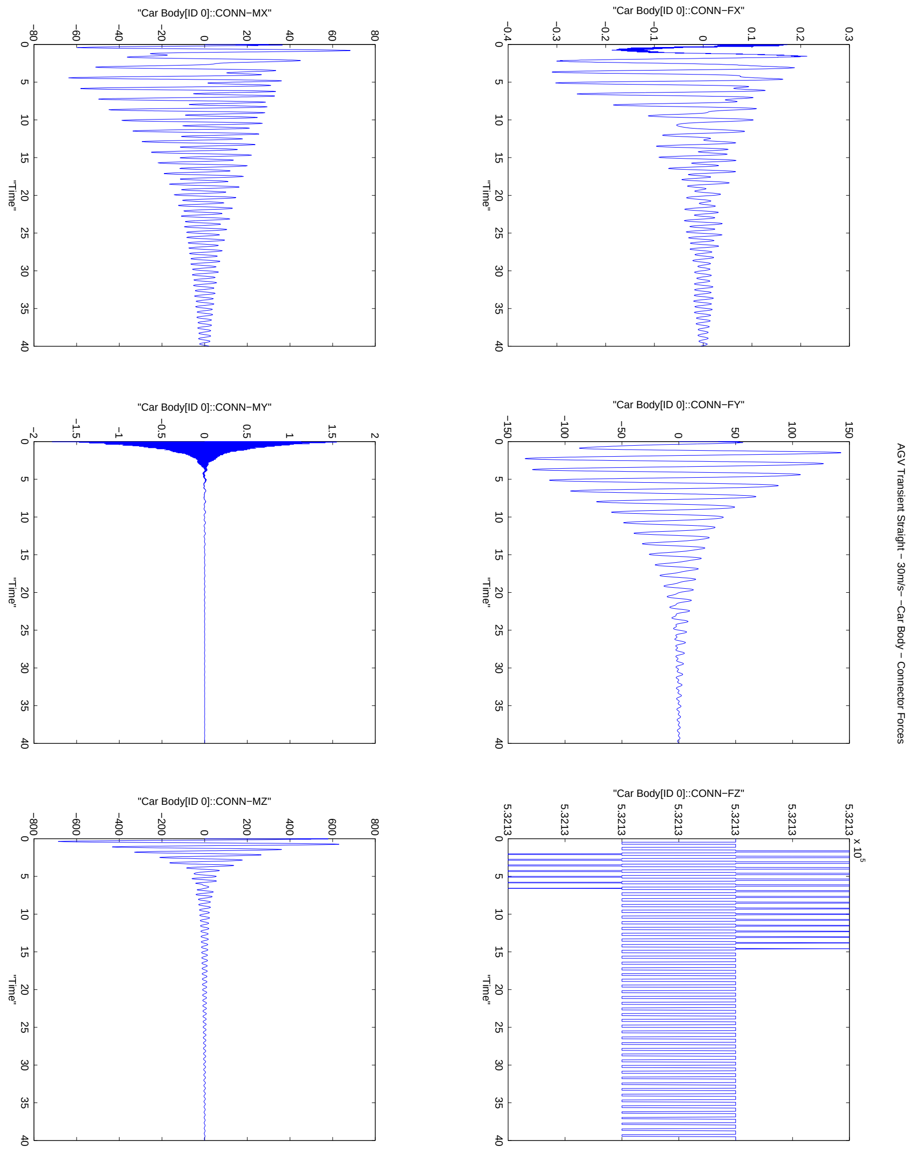


Figure F.94: AGV model: Suspension forces acting on the car body running at 30m/s on straight track.

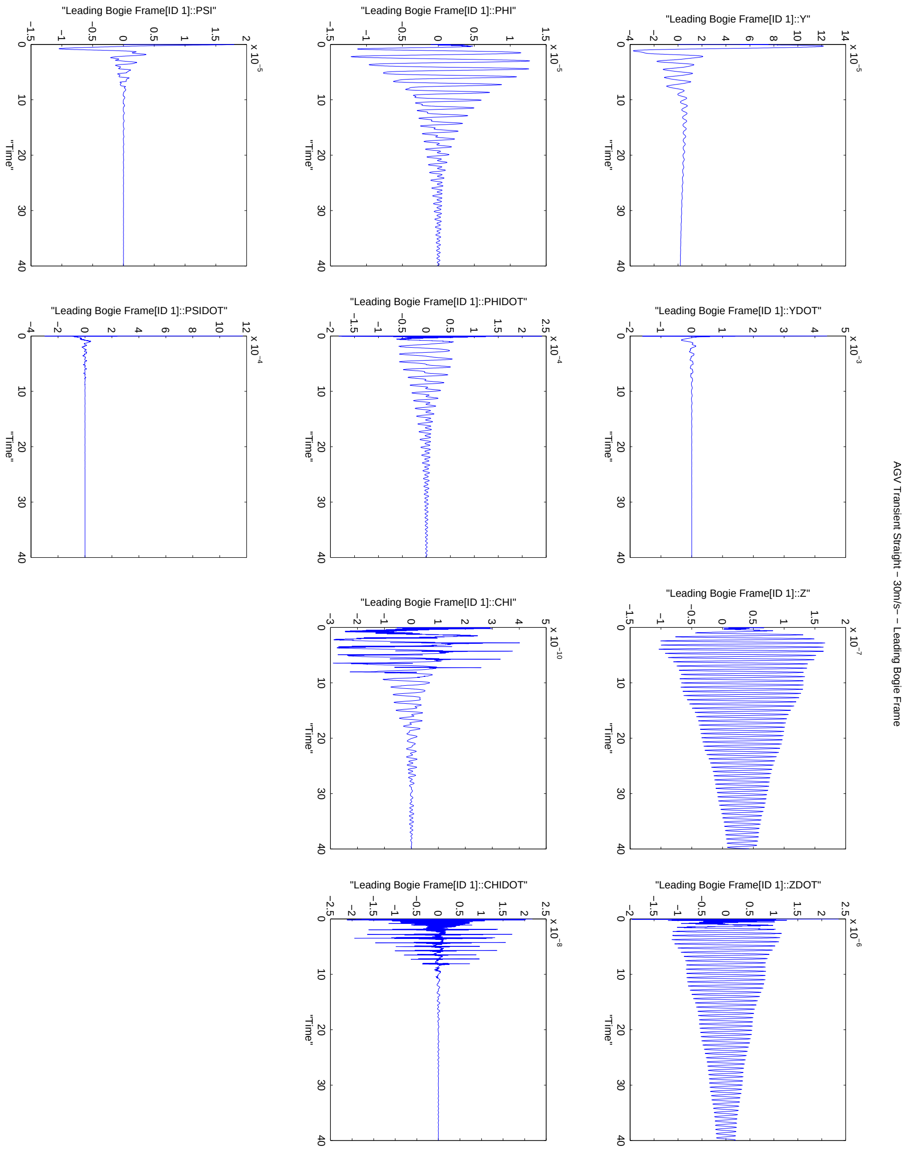


Figure F.95: AGV model: Dynamics of the leading bogie frame running at 30m/s on straight track.

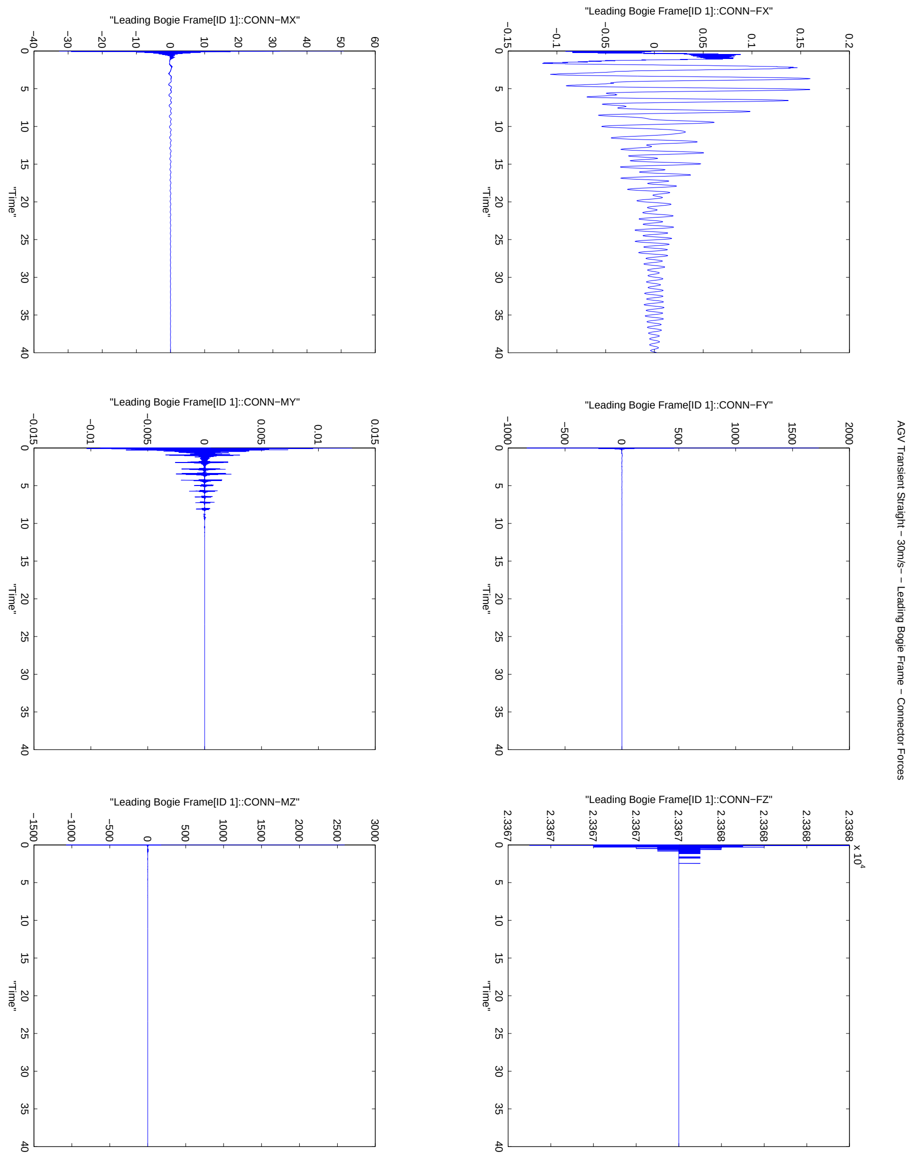


Figure F.96: AGV model: Suspension forces acting on the leading bogie frame running at 30m/s on straight track.

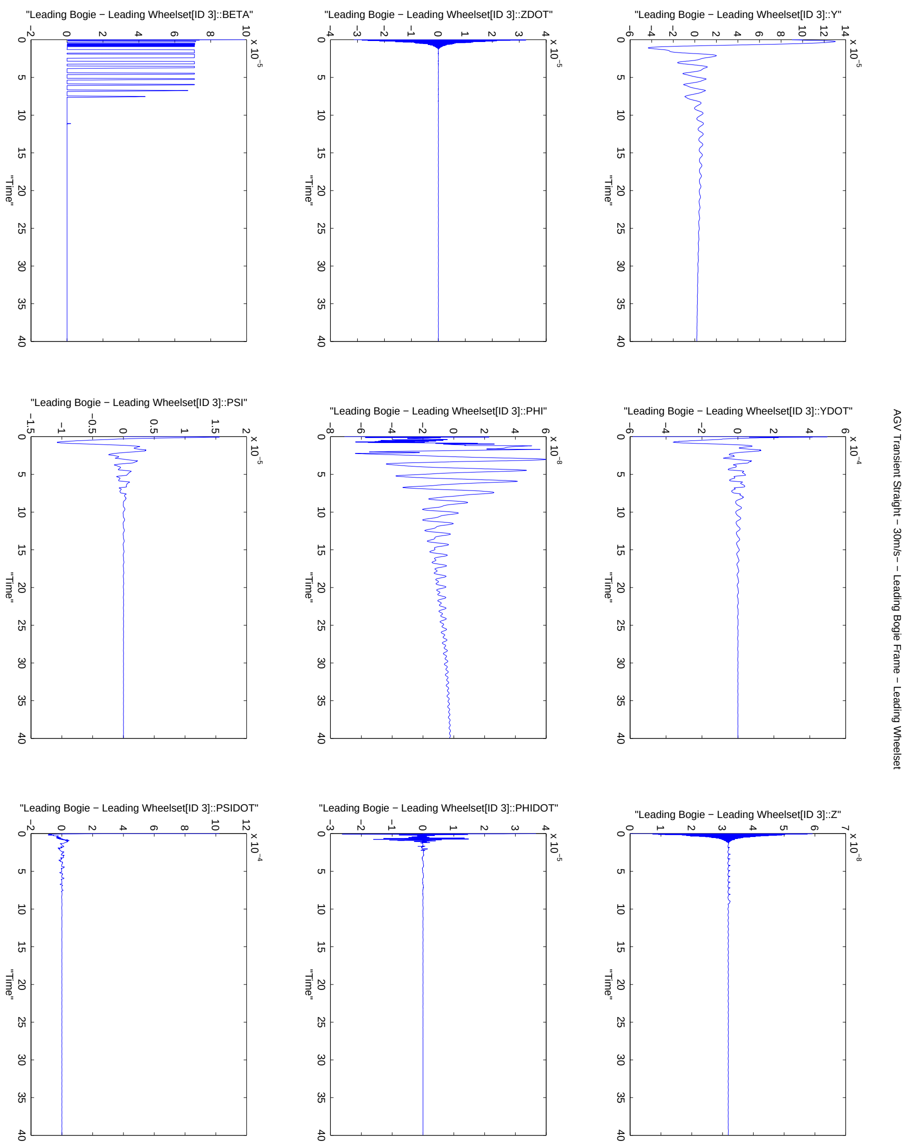


Figure F.97: AGV model: Dynamics of the leading wheel set mounted on the leading bogie frame running at 30m/s on straight track.

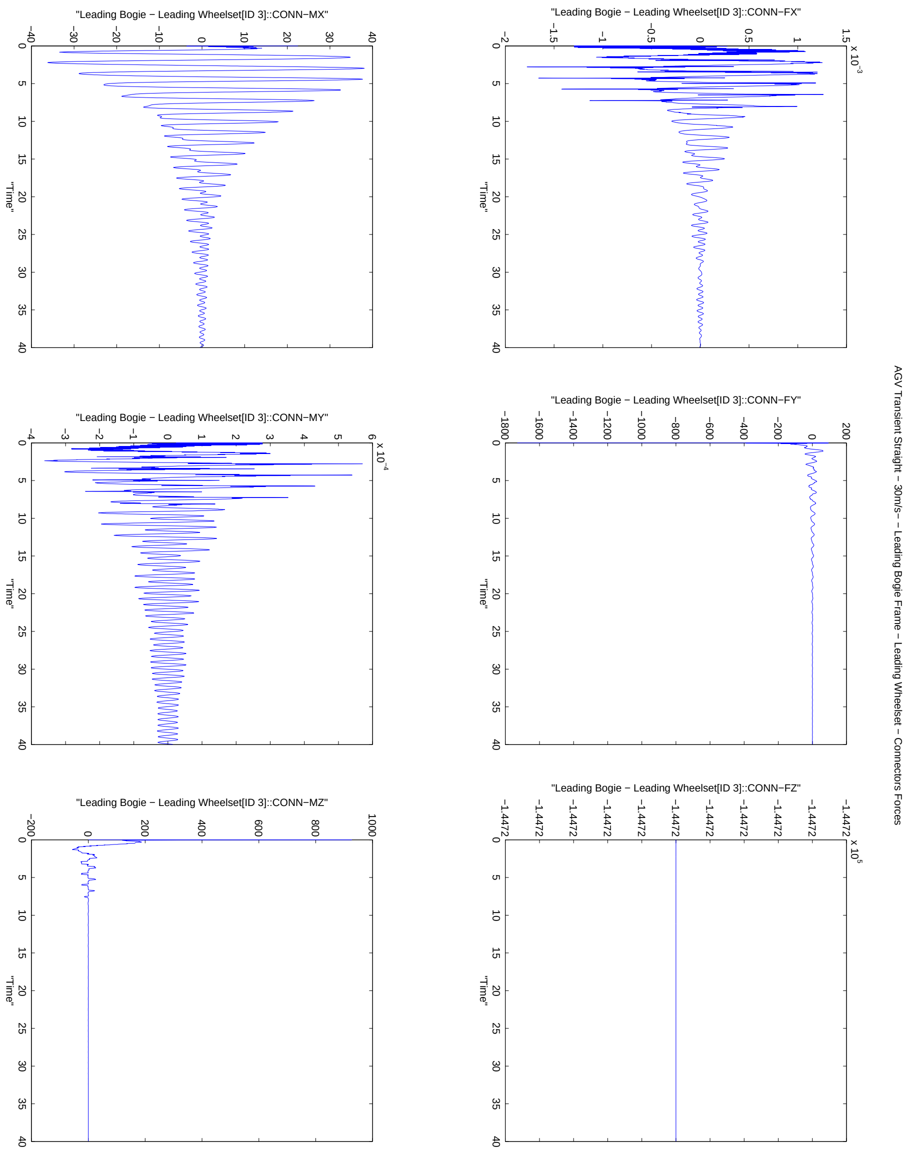


Figure F.98: AGV model: Suspension forces acting on the leading wheel set mounted on the leading bogie frame running at 30m/s on straight track.

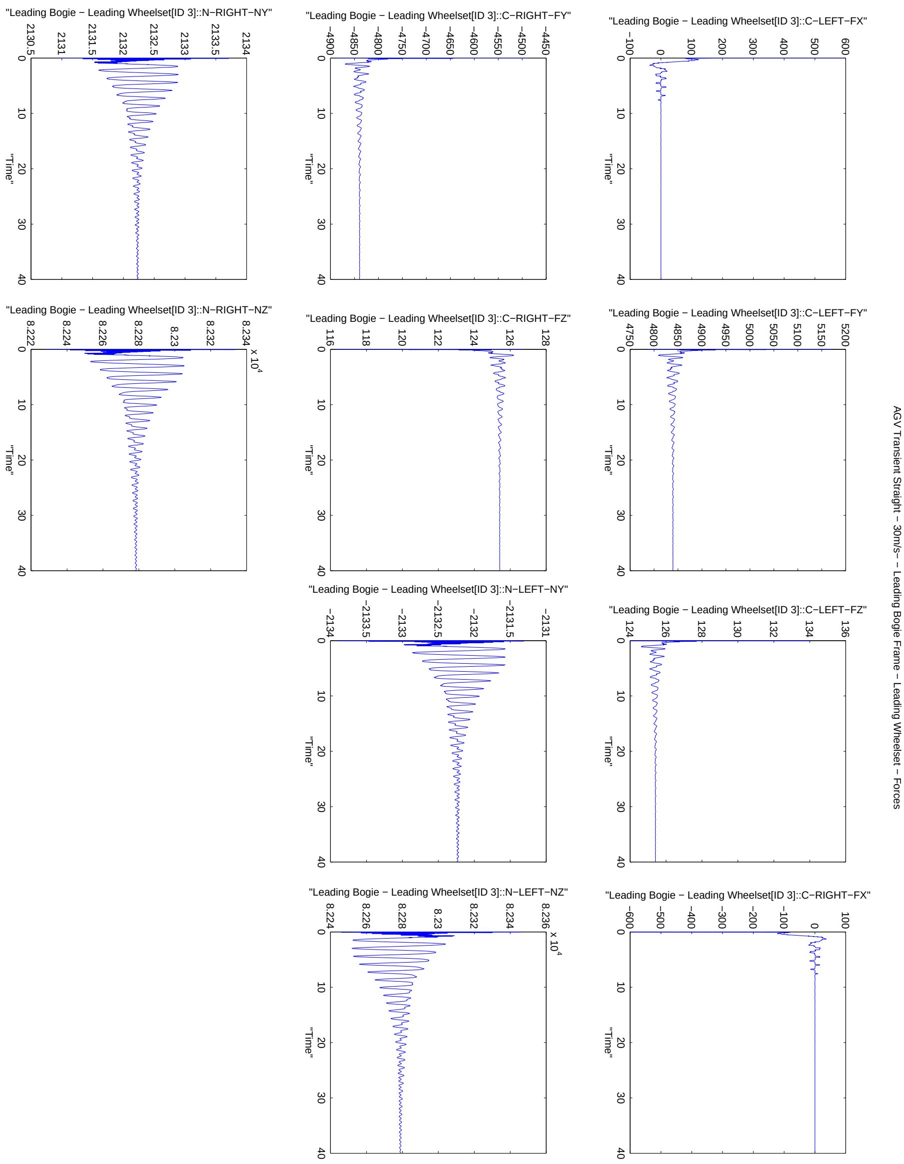


Figure F.99: AGV model: Contact forces acting on the leading wheel set mounted on the leading bogie frame running at 30m/s on straight track.

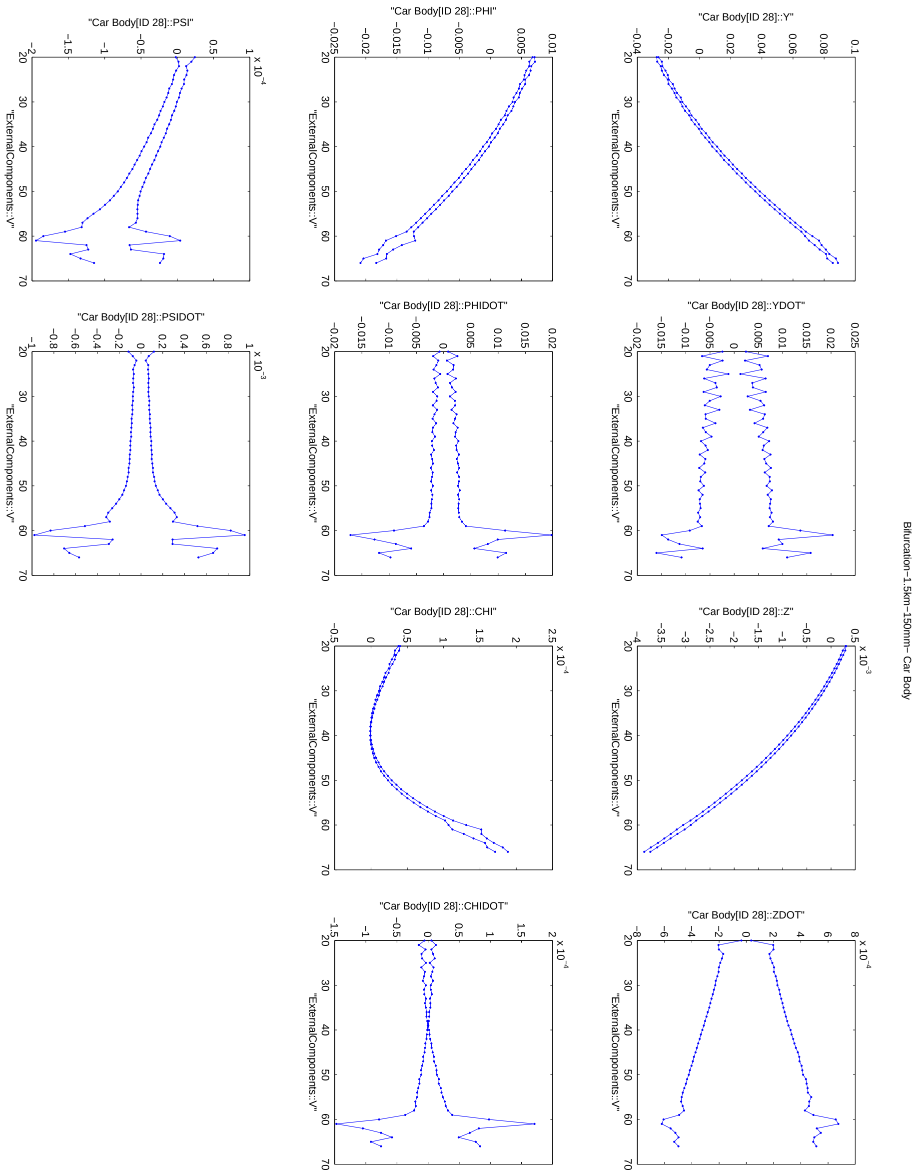


Figure F.100: AGV model: dynamics of the car body at increasing speed on a curve of 1.5km with 150mm of cant.

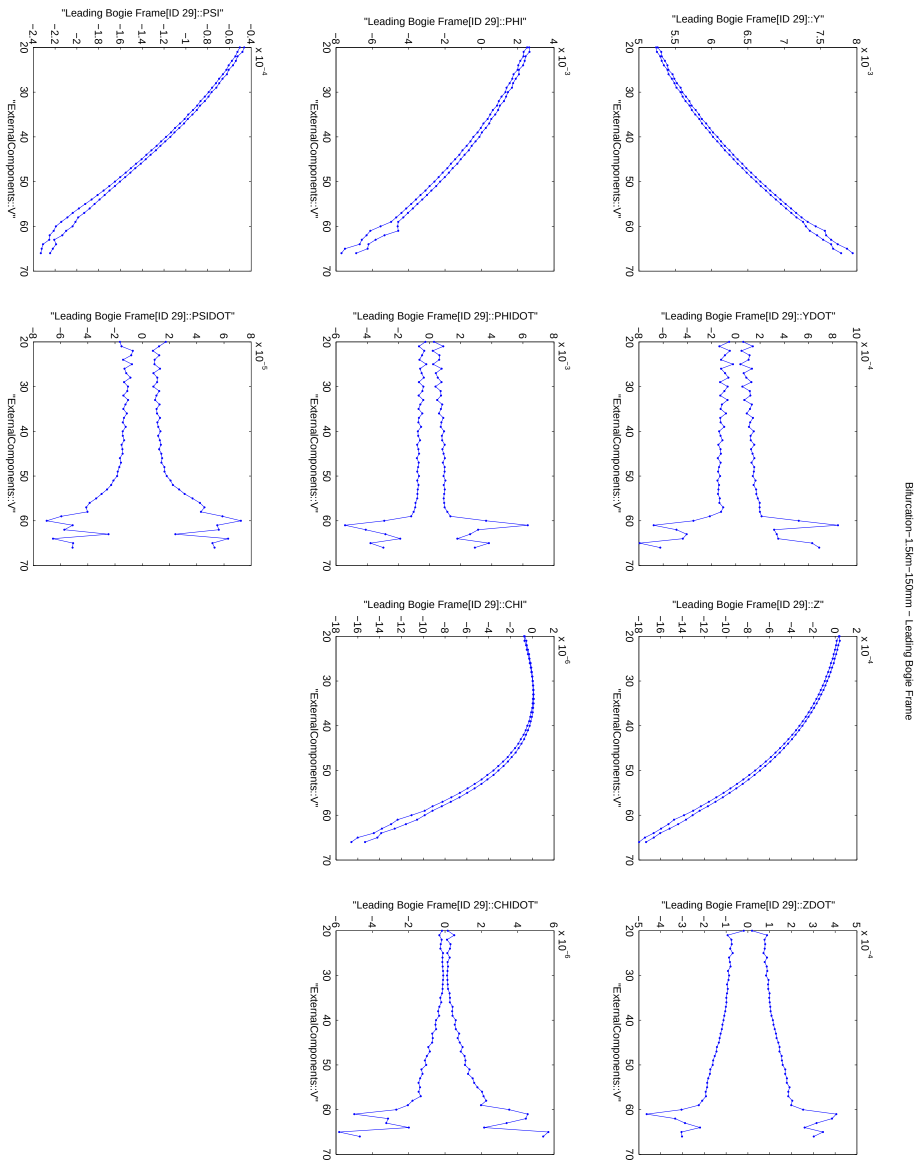


Figure F.101: AGV model: dynamics of the leading bogie frame at increasing speed on a curve of 1.5km with 150mm of cant.

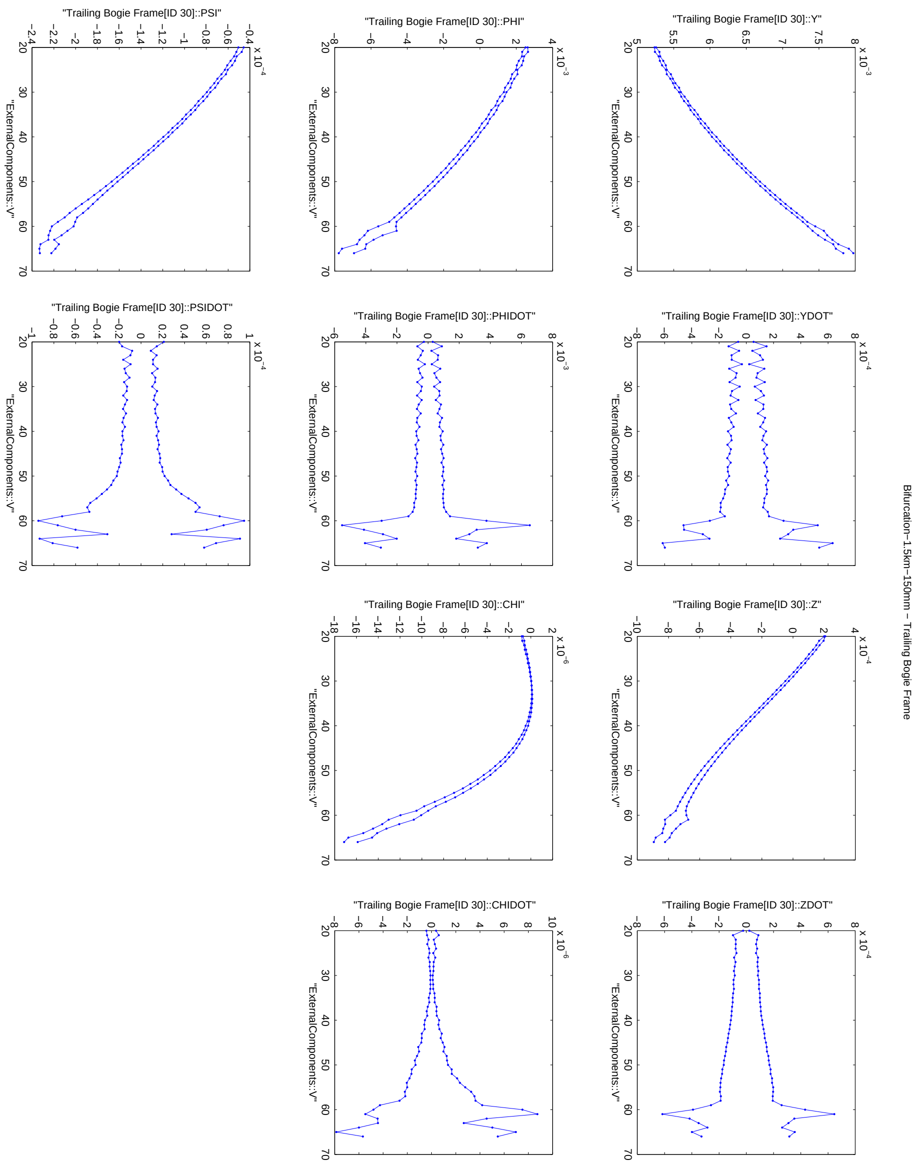


Figure F.102: AGV model: dynamics of the trailing bogie frame at increasing speed on a curve of 1.5km with 150mm of cant.

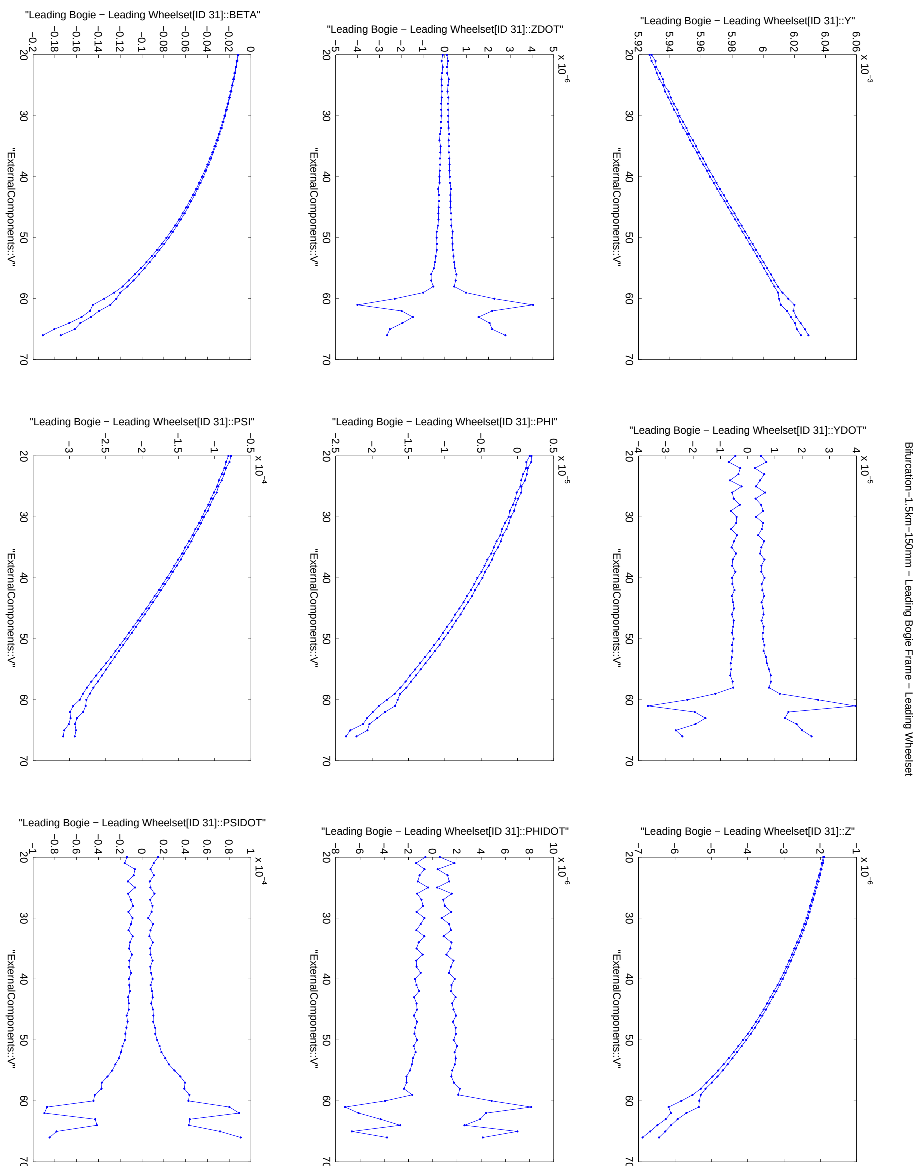


Figure F.103: AGV model: dynamics of the leading wheel set attached to the leading bogie frame at increasing speed on a curve of 1.5km with 150mm of cant.

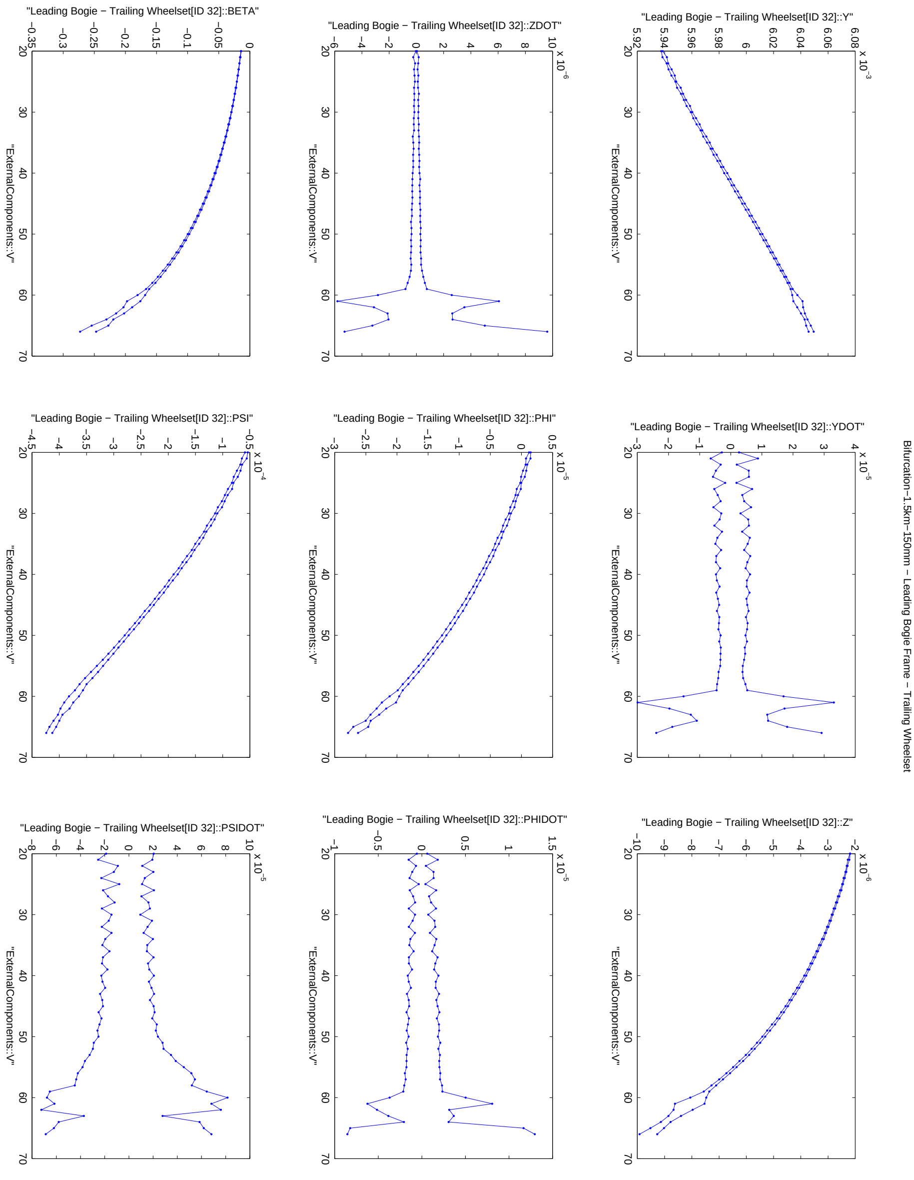


Figure F.104: AGV model: dynamics of the trailing wheel set attached to the leading bogie frame at increasing speed on a curve of 1.5km with 150mm of cant.

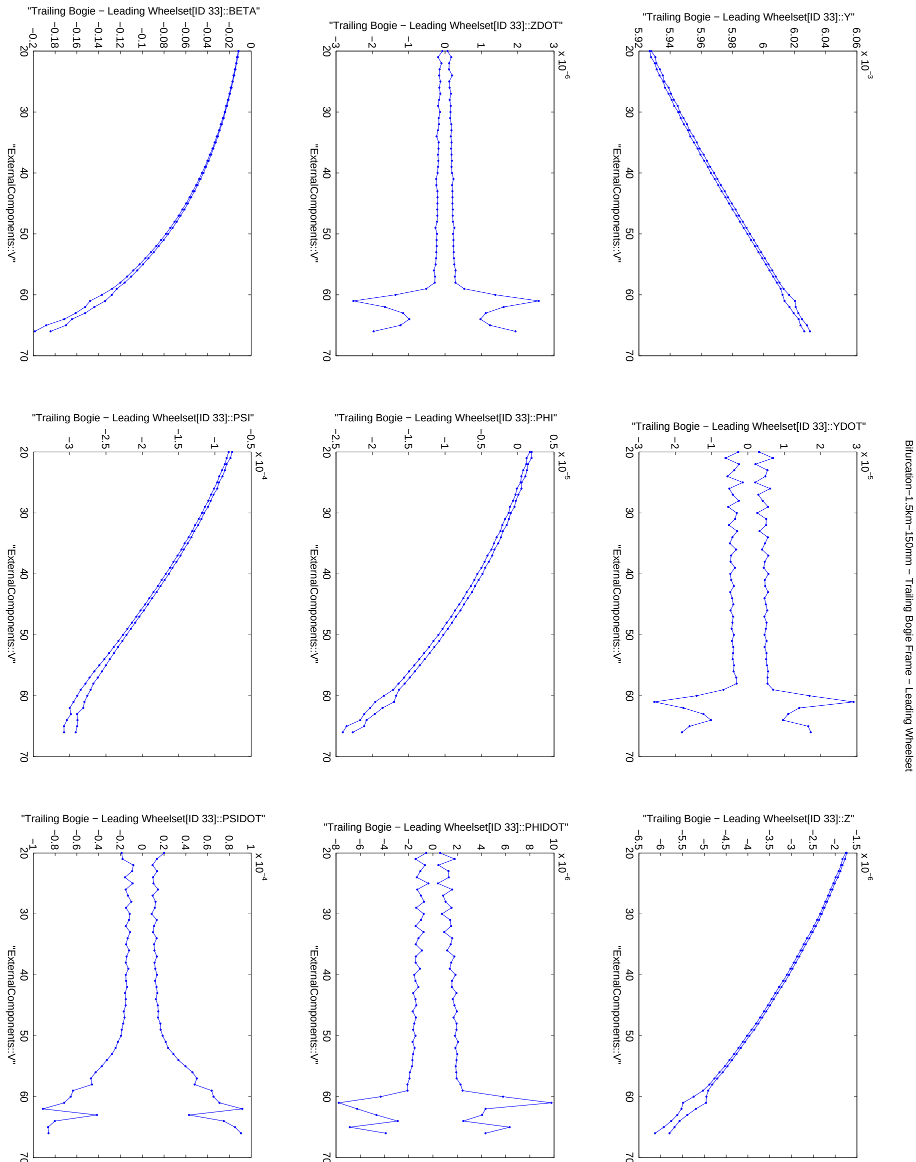


Figure F.105: AGV model: dynamics of the leading wheel set attached to the trailing bogie frame at increasing speed on a curve of 1.5km with 150mm of cant.

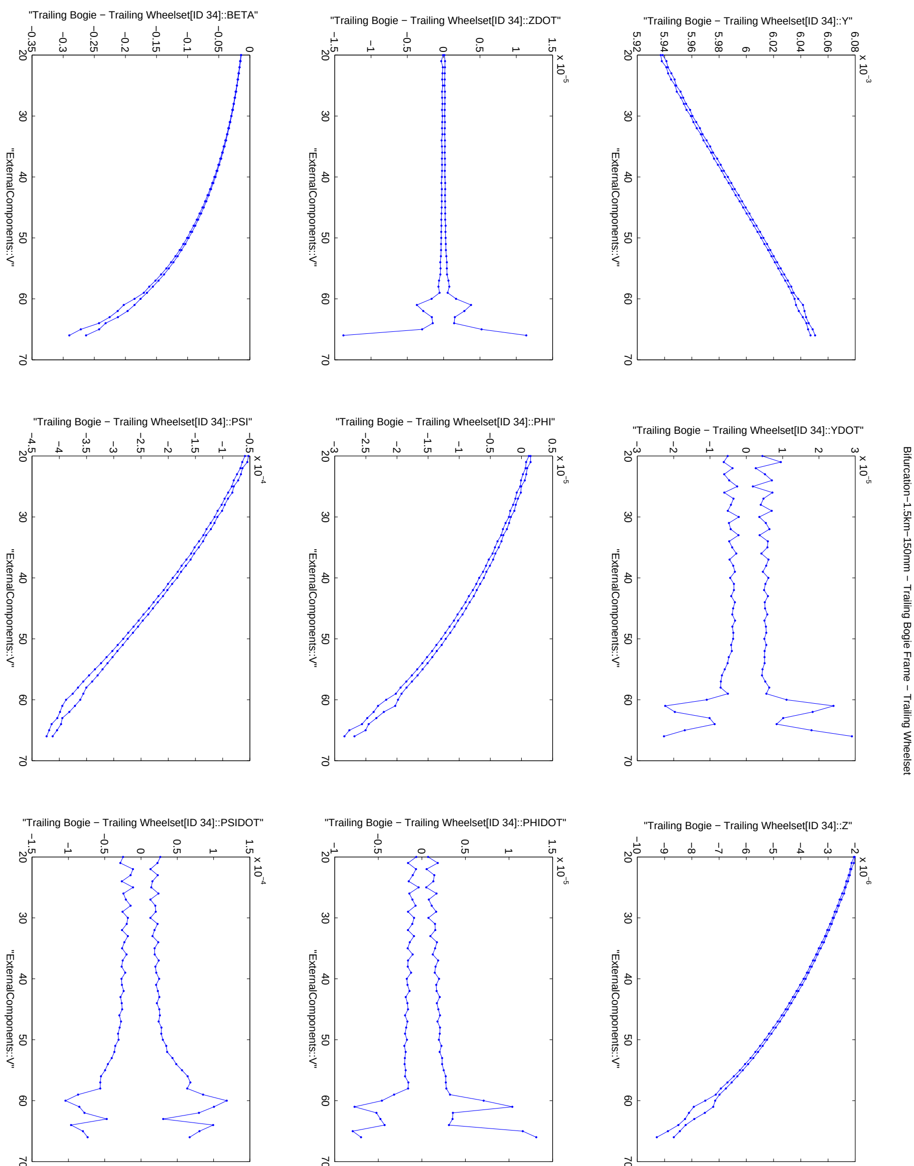


Figure F.106: AGV model: dynamics of the trailing wheel set attached to the trailing bogie frame at increasing speed on a curve of 1.5km with 150mm of cant.

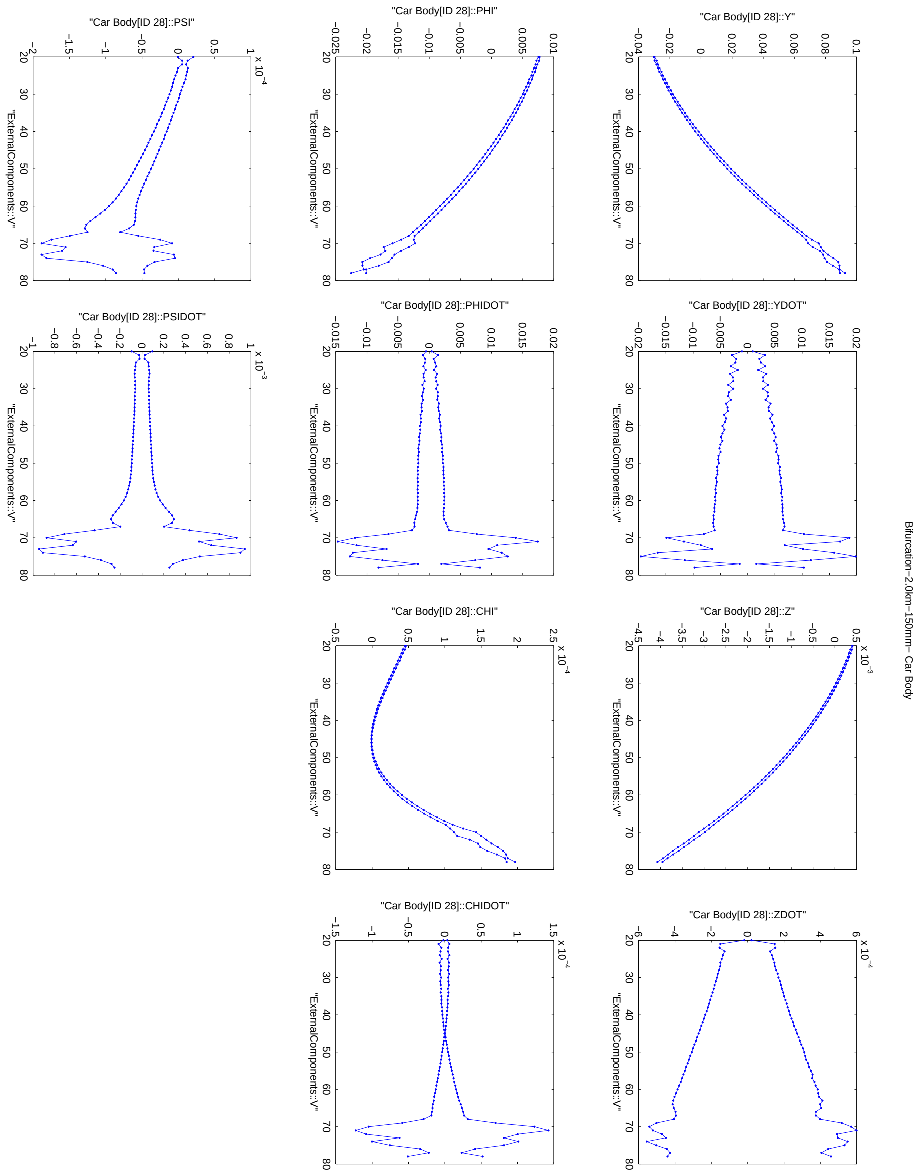


Figure F.107: AGV model: dynamics of the car body at increasing speed on a curve of 2.0km with 150mm of cant.

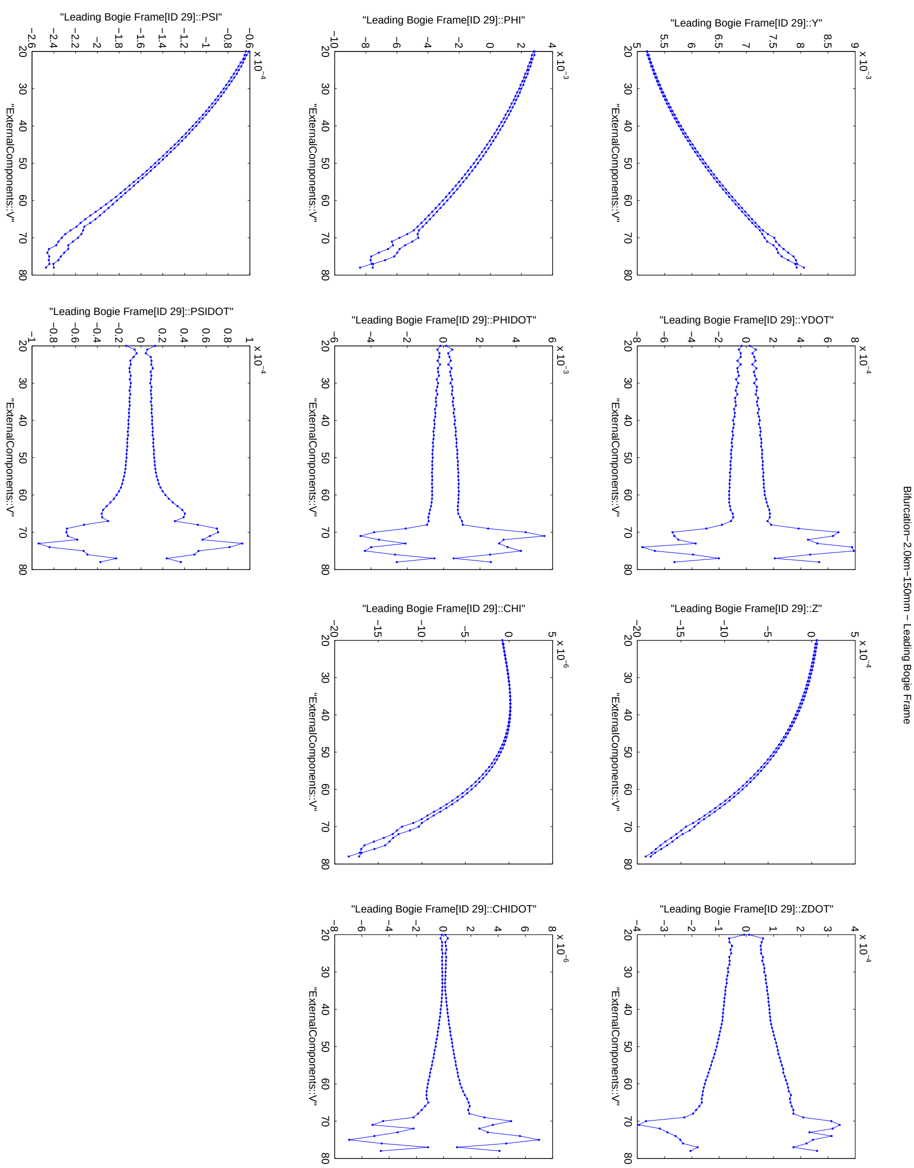


Figure F.108: AGV model: dynamics of the leading bogie frame at increasing speed on a curve of 2.0km with 150mm of cant.

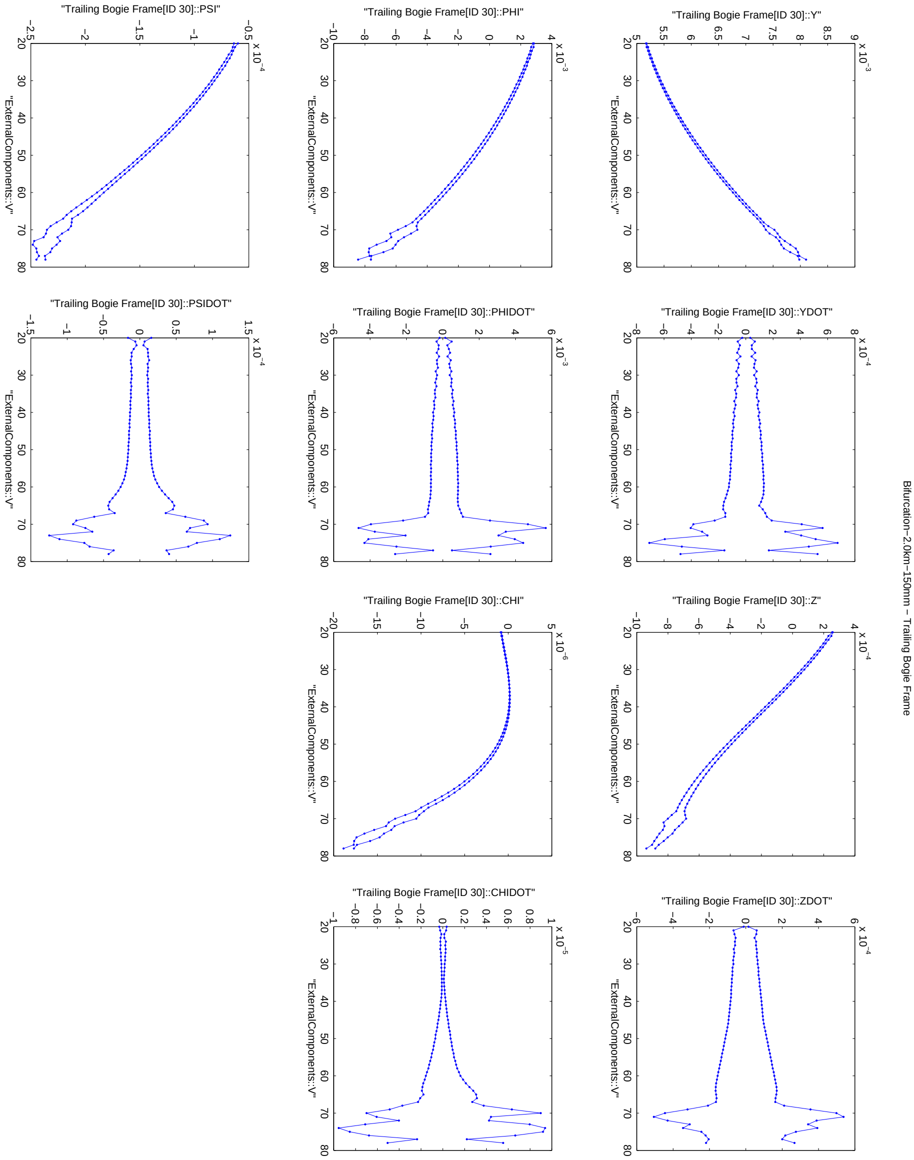


Figure F.109: AGV model: dynamics of the trailing bogie frame at increasing speed on a curve of 2.0km with 150mm of cant.

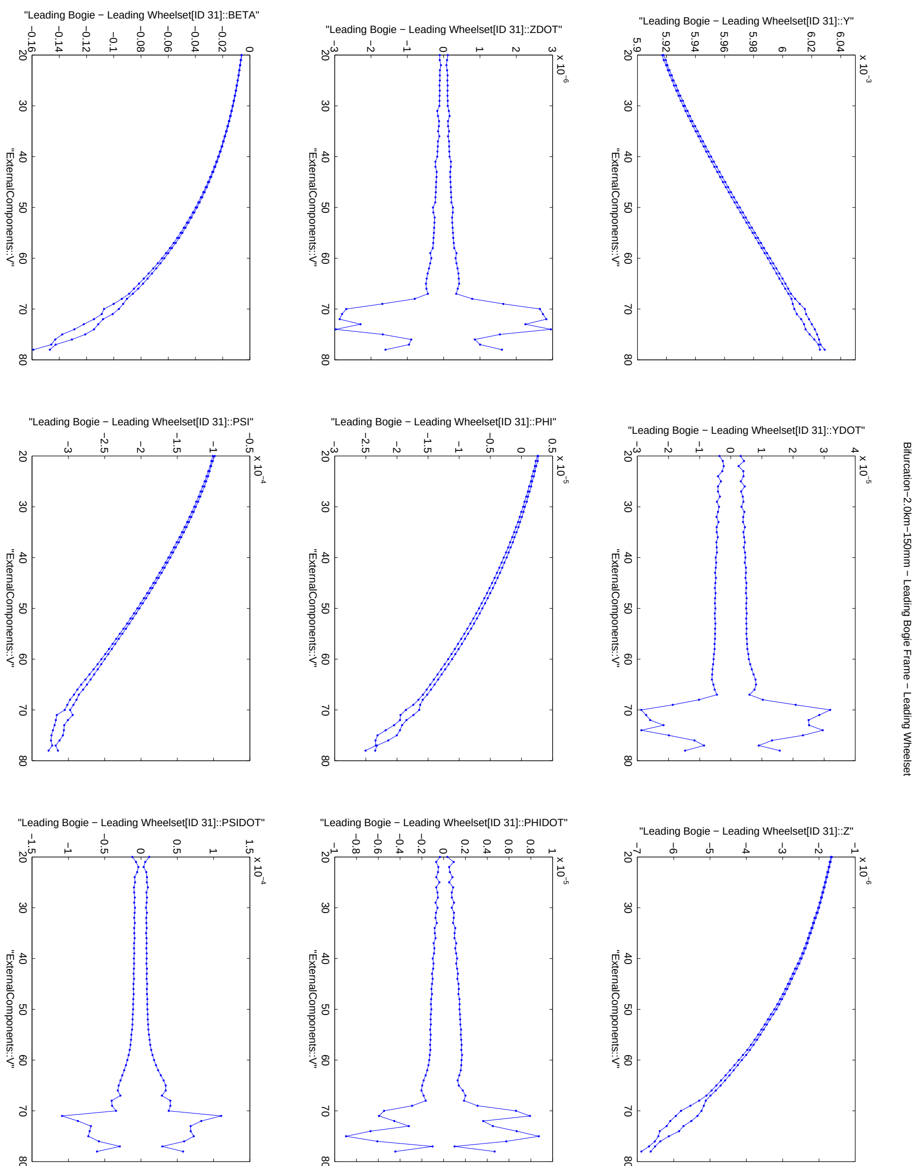


Figure F.110: AGV model: dynamics of the leading wheel set attached to the leading bogie frame at increasing speed on a curve of 2.0km with 150mm of cant.

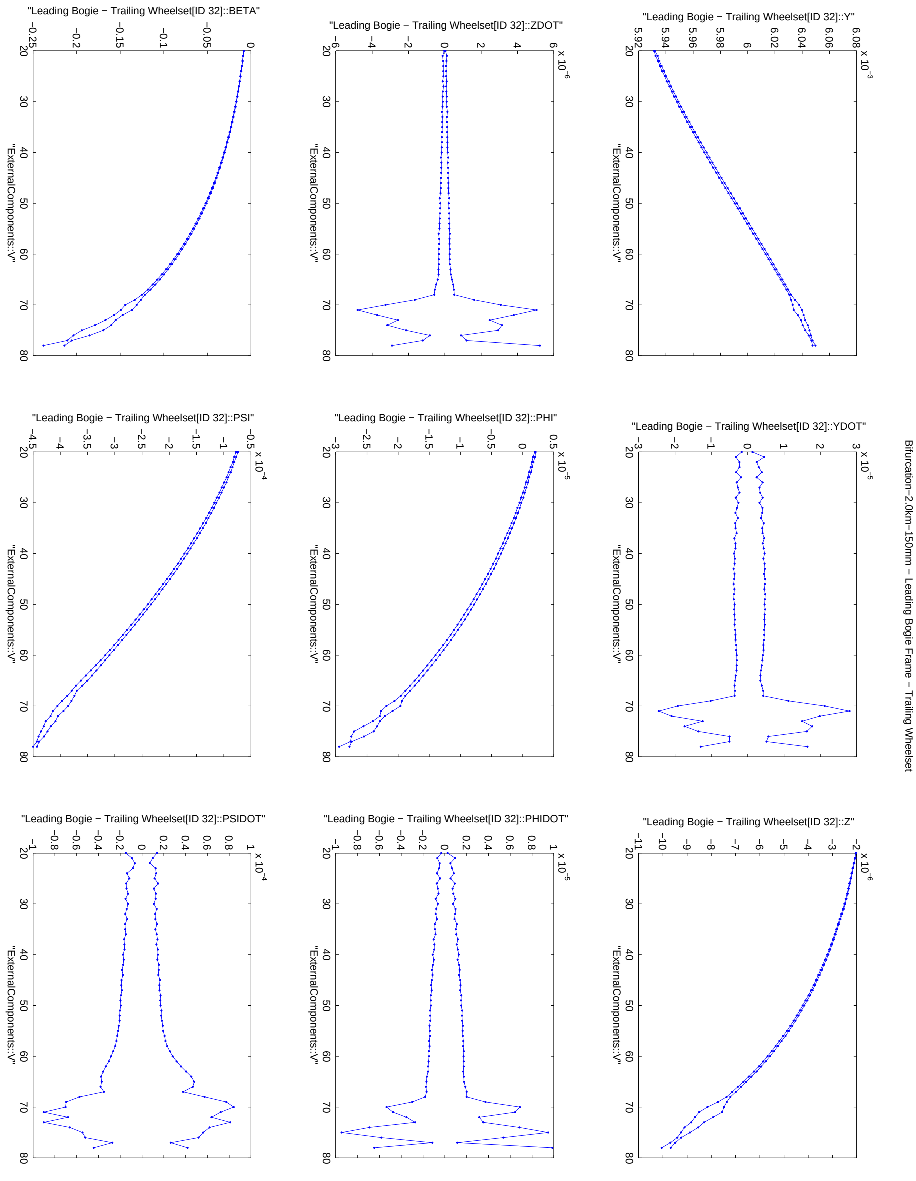


Figure F.111: AGV model: dynamics of the trailing wheel set attached to the leading bogie frame at increasing speed on a curve of 2.0km with 150mm of cant.

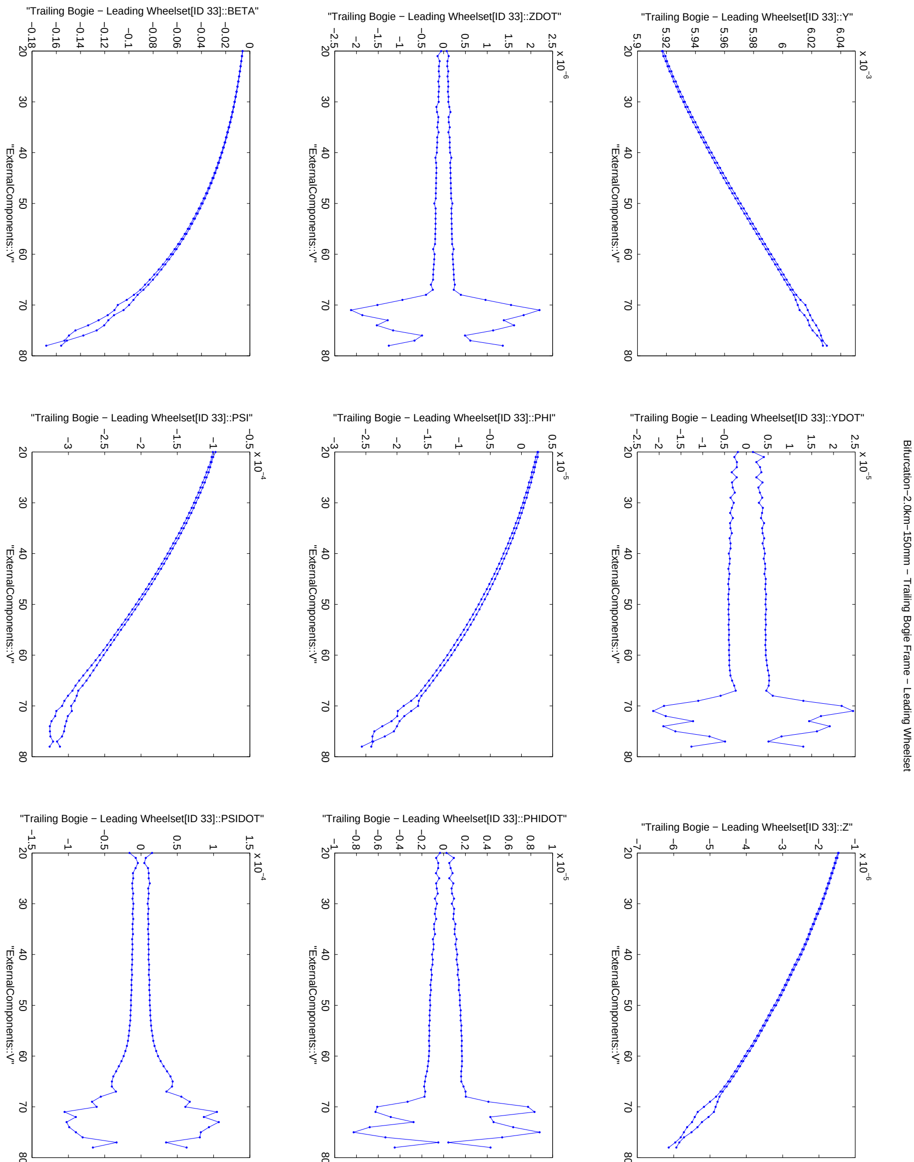


Figure F.112: AGV model: dynamics of the leading wheel set attached to the trailing bogie frame at increasing speed on a curve of 2.0km with 150mm of cant.

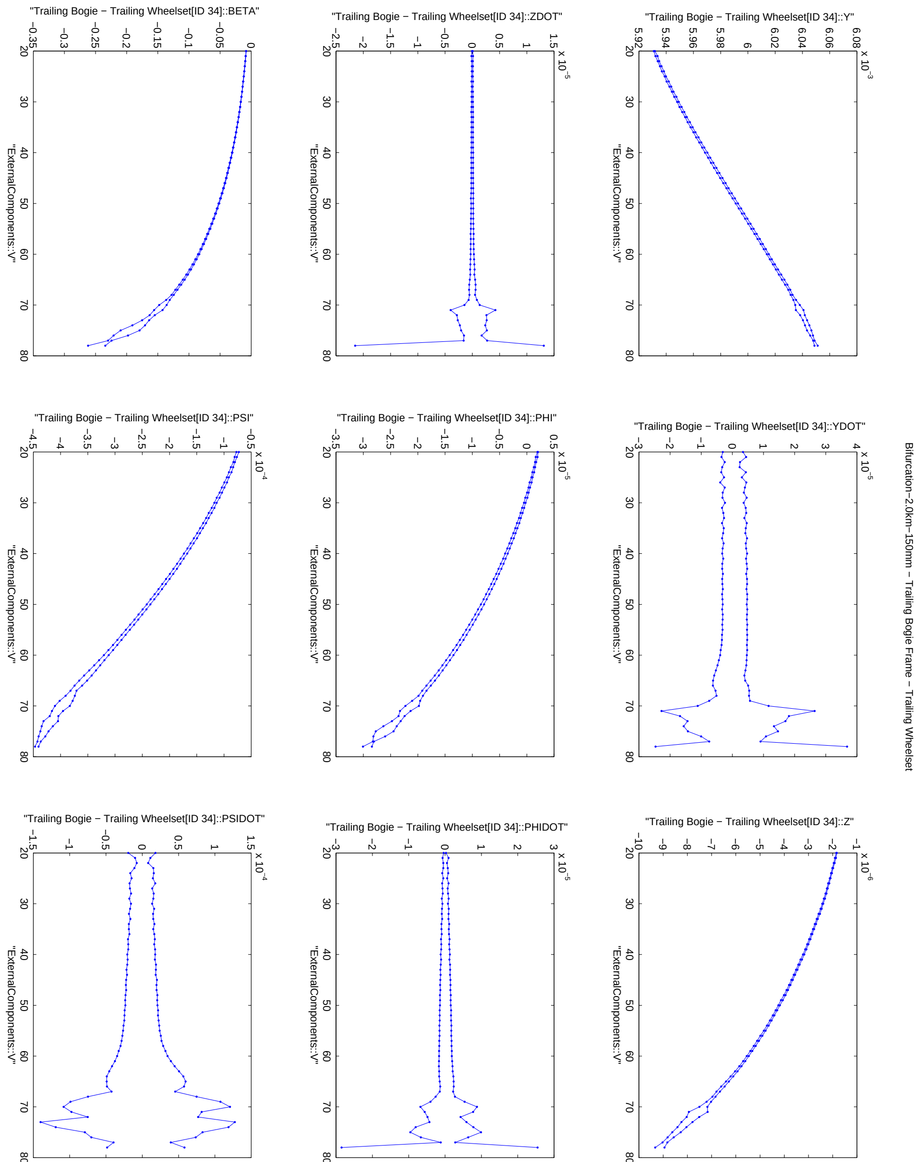


Figure F.113: AGV model: dynamics of the trailing wheel set attached to the trailing bogie frame at increasing speed on a curve of 2.0km with 150mm of cant.

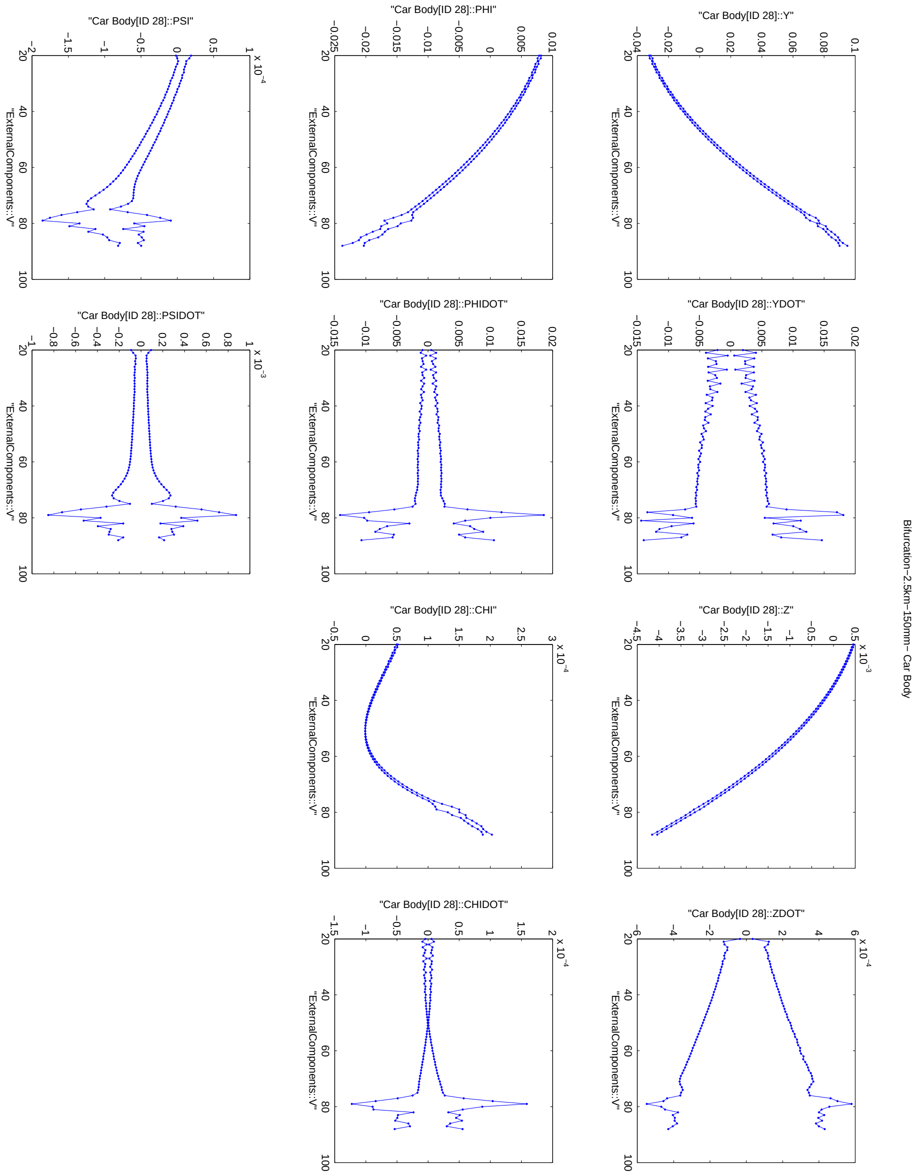


Figure F.114: AGV model: dynamics of the car body at increasing speed on a curve of 2.5km with 150mm of cant.

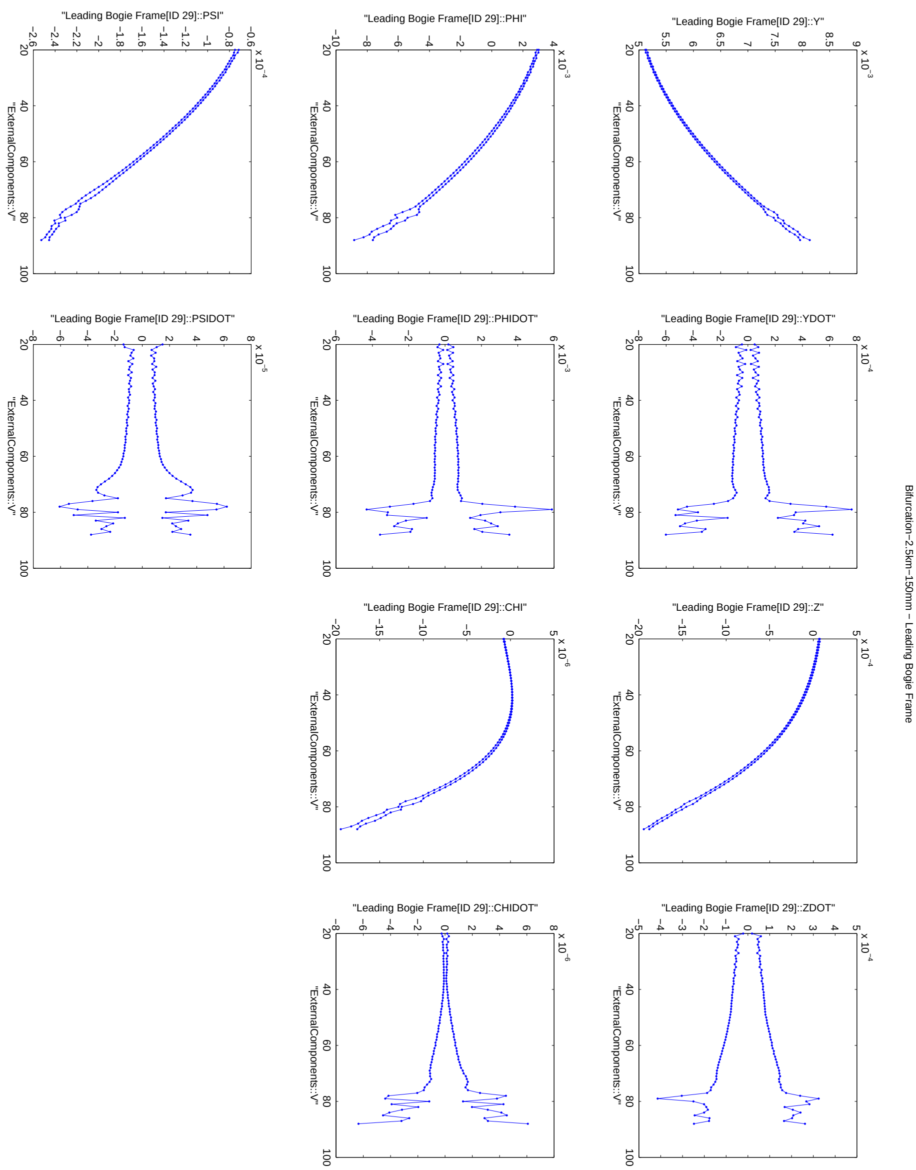


Figure F.115: AGV model: dynamics of the leading bogie frame at increasing speed on a curve of 2.5km with 150mm of cant.

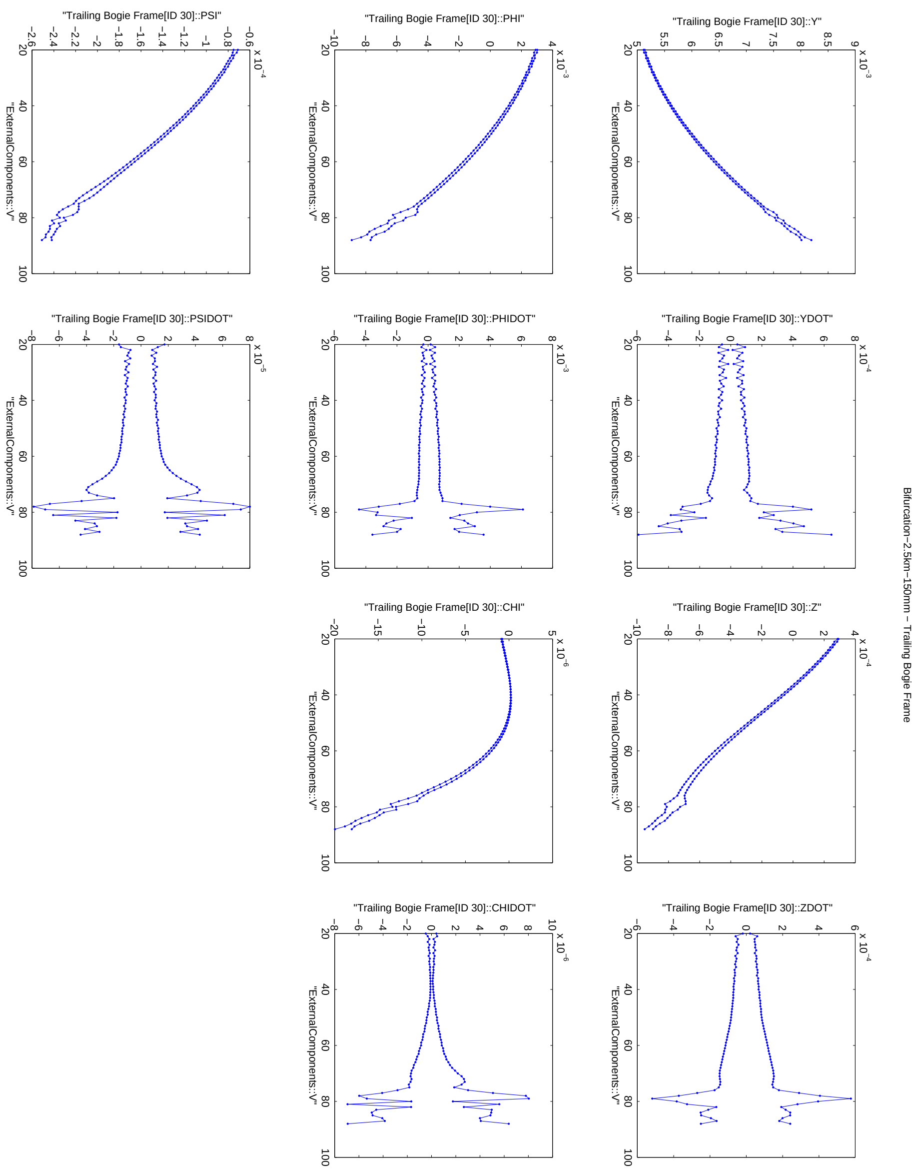


Figure F.116: AGV model: dynamics of the trailing bogie frame at increasing speed on a curve of 2.5km with 150mm of cant.

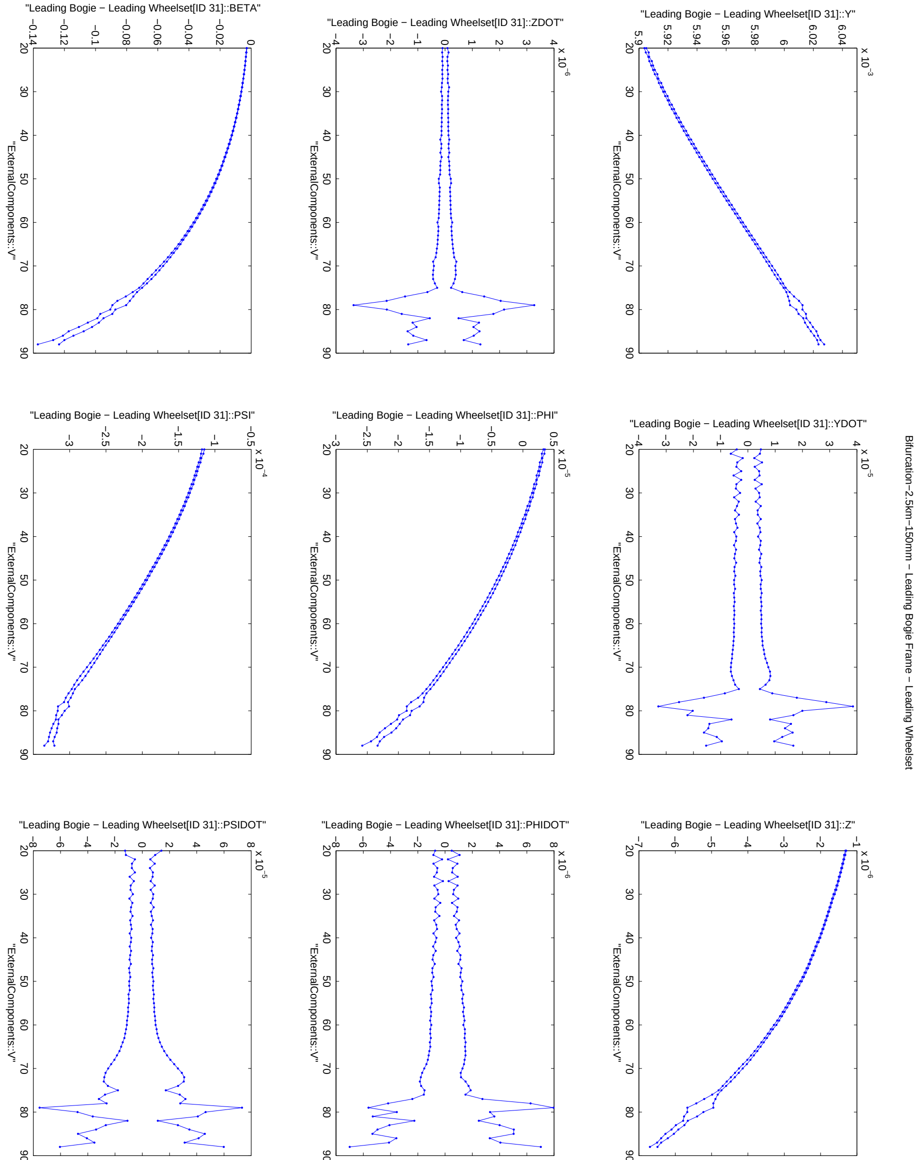


Figure F.117: AGV model: dynamics of the leading wheel set attached to the leading bogie frame at increasing speed on a curve of 2.5km with 150mm of cant.

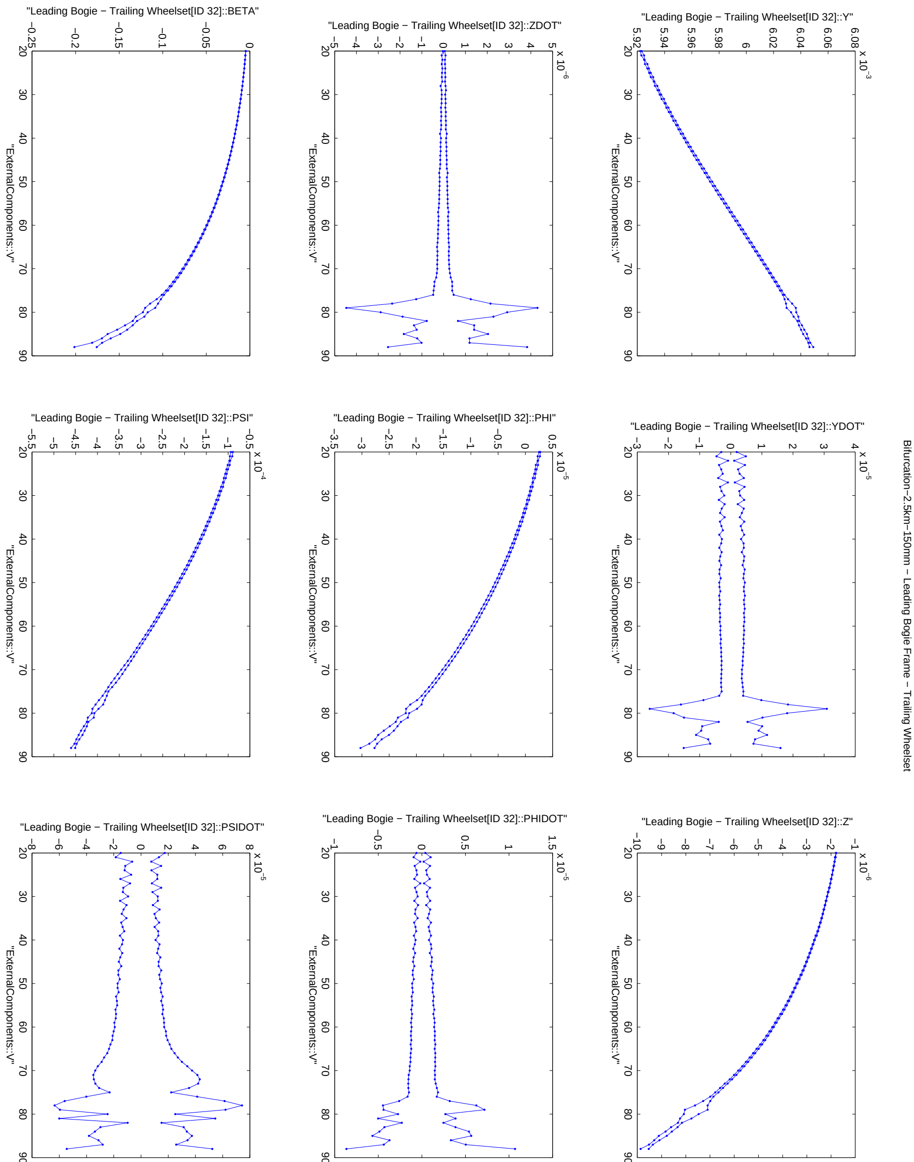


Figure F.118: AGV model: dynamics of the trailing wheel set attached to the leading bogie frame at increasing speed on a curve of 2.5km with 150mm of cant.

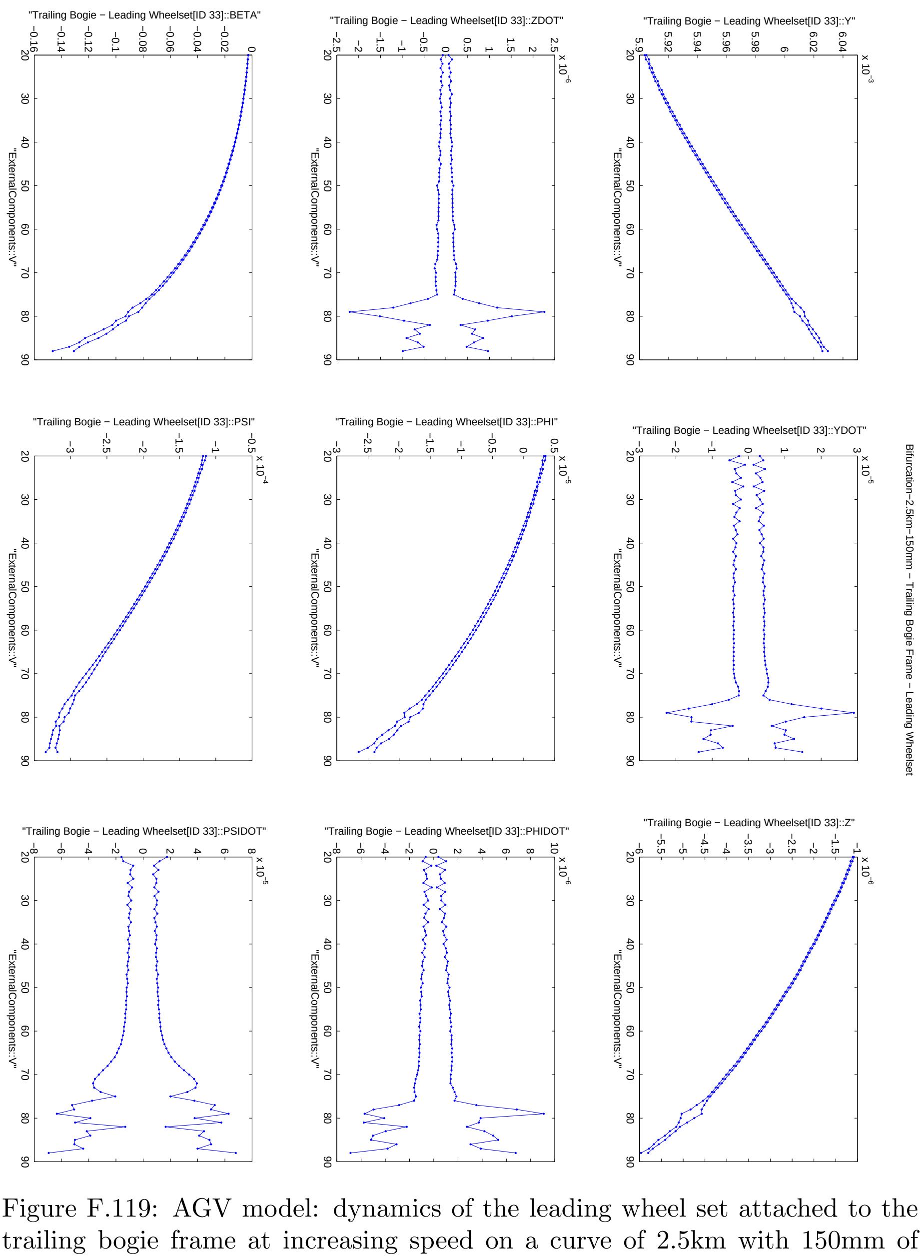


Figure F.119: AGV model: dynamics of the leading wheel set attached to the trailing bogie frame at increasing speed on a curve of 2.5km with 150mm of cant.

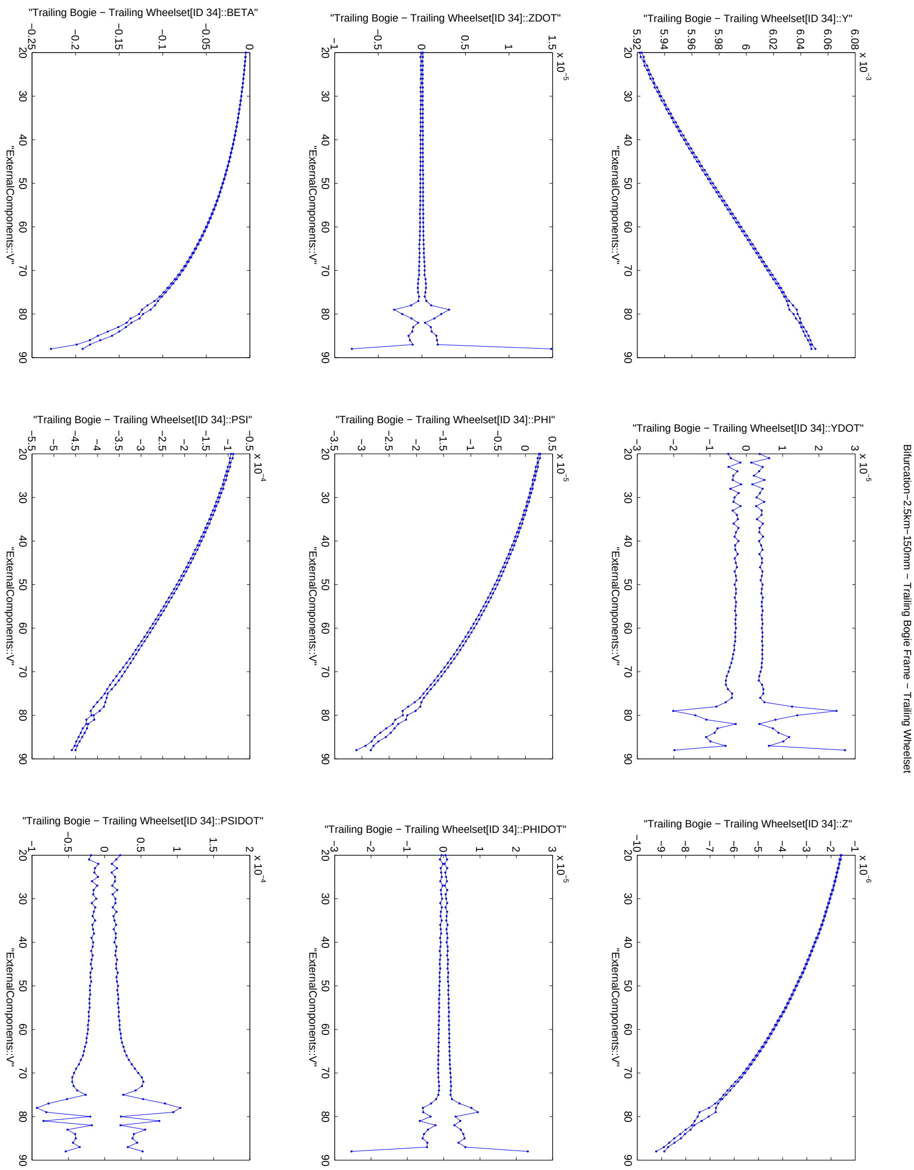


Figure F.120: AGV model: dynamics of the trailing wheel set attached to the trailing bogie frame at increasing speed on a curve of 2.5km with 150mm of cant.

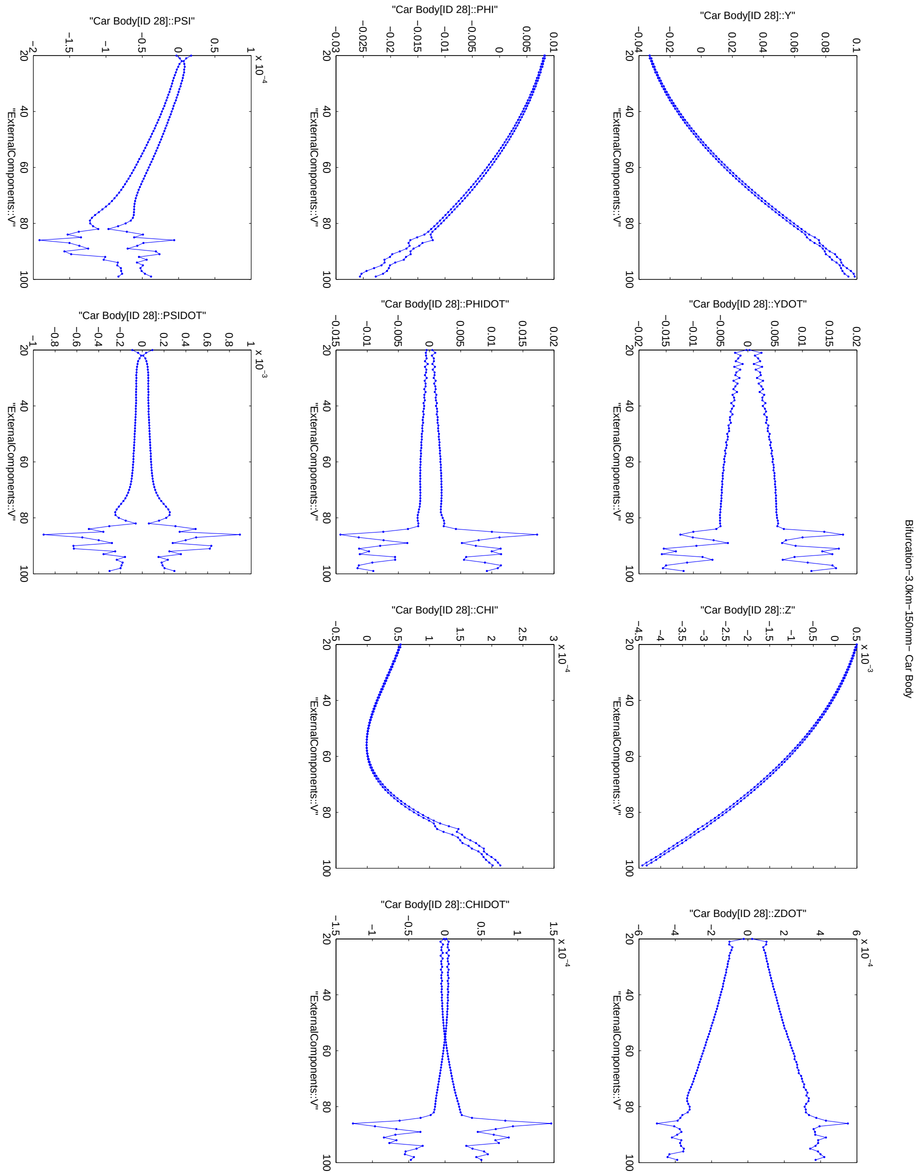


Figure F.121: AGV model: dynamics of the car body at increasing speed on a curve of 3.0km with 150mm of cant.

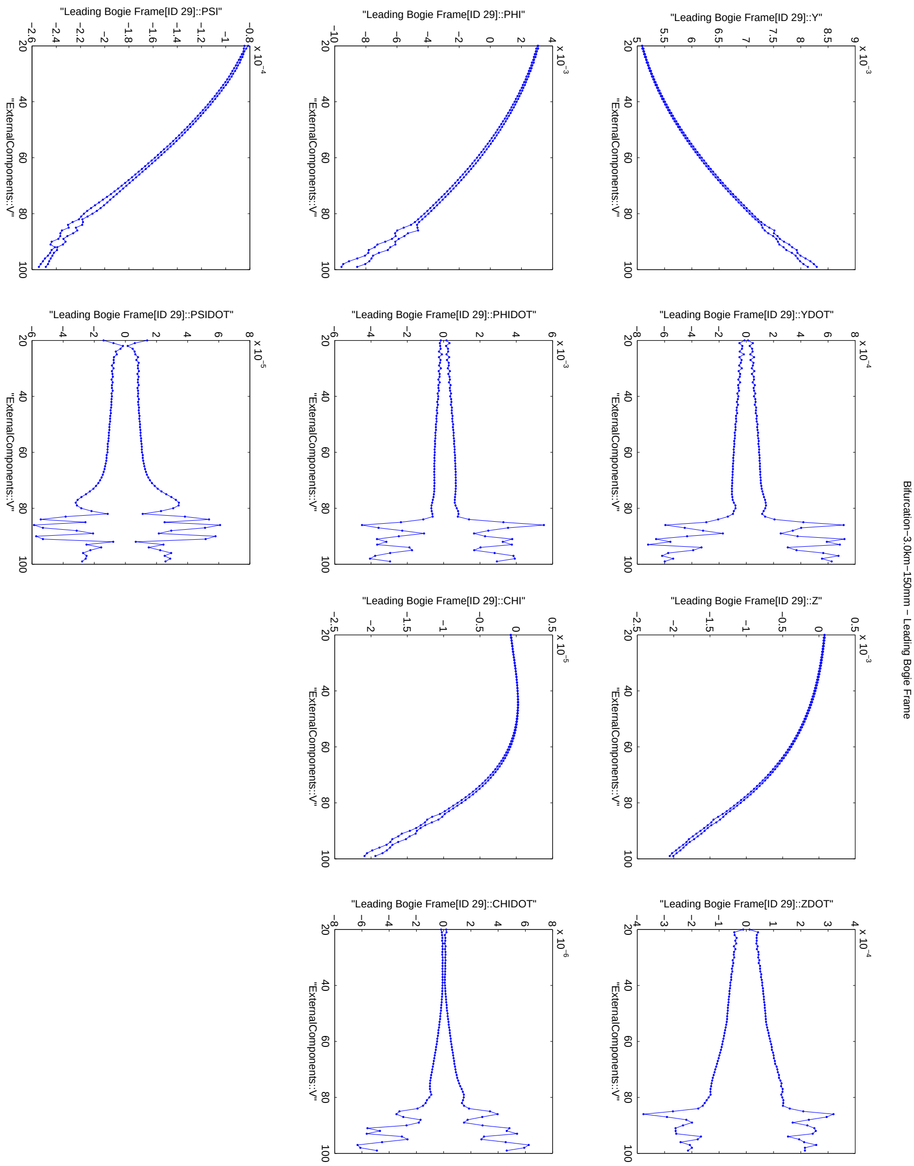


Figure F.122: AGV model: dynamics of the leading bogie frame at increasing speed on a curve of 3.0km with 150mm of cant.

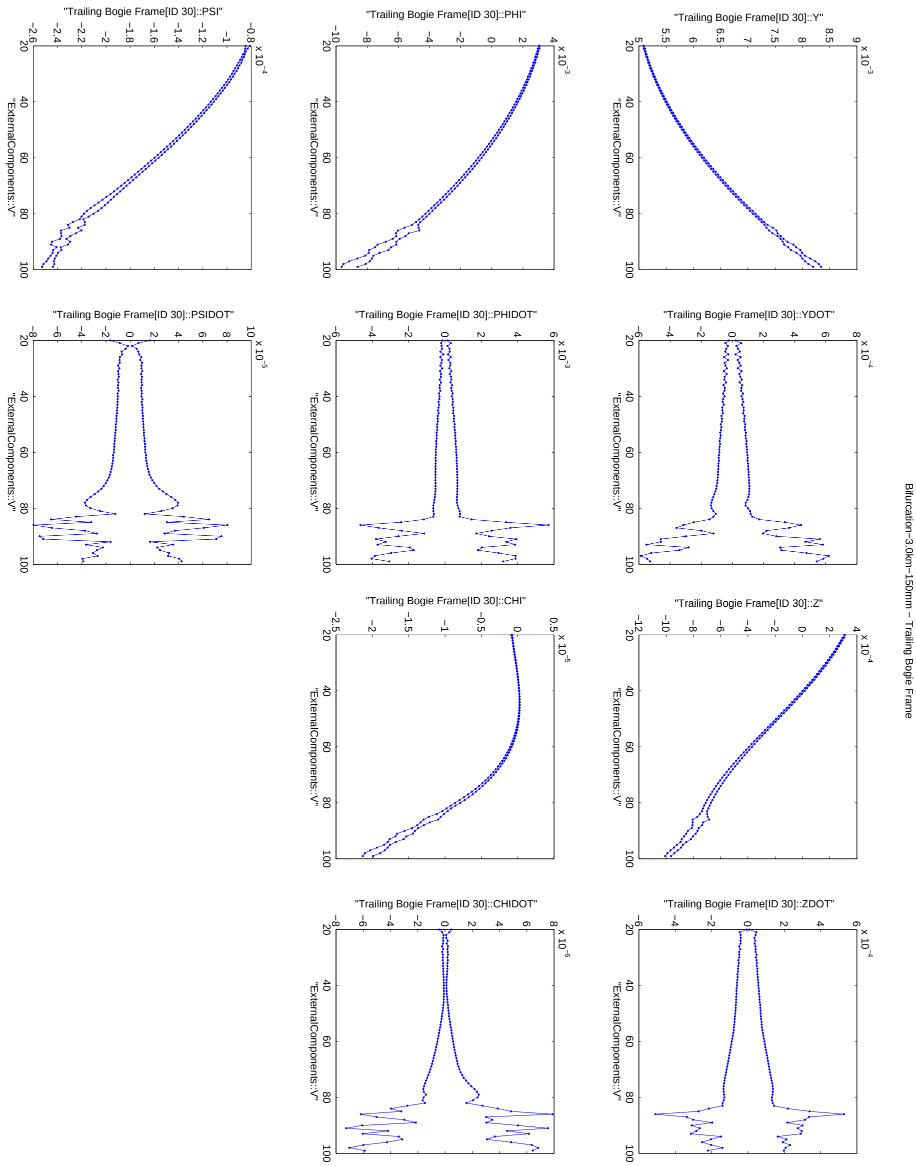


Figure F.123: AGV model: dynamics of the trailing bogie frame at increasing speed on a curve of 3.0km with 150mm of cant.

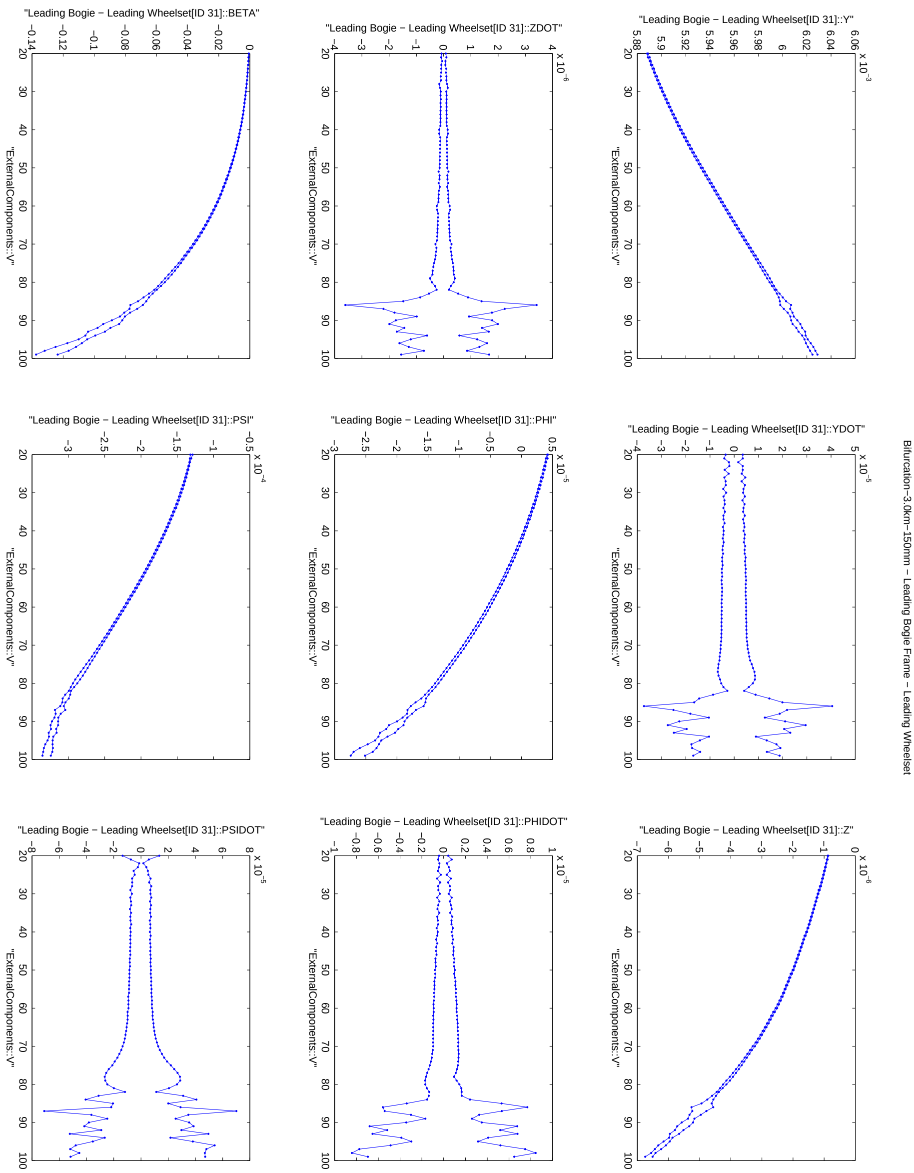


Figure F.124: AGV model: dynamics of the leading wheel set attached to the leading bogie frame at increasing speed on a curve of 3.0km with 150mm of cant.

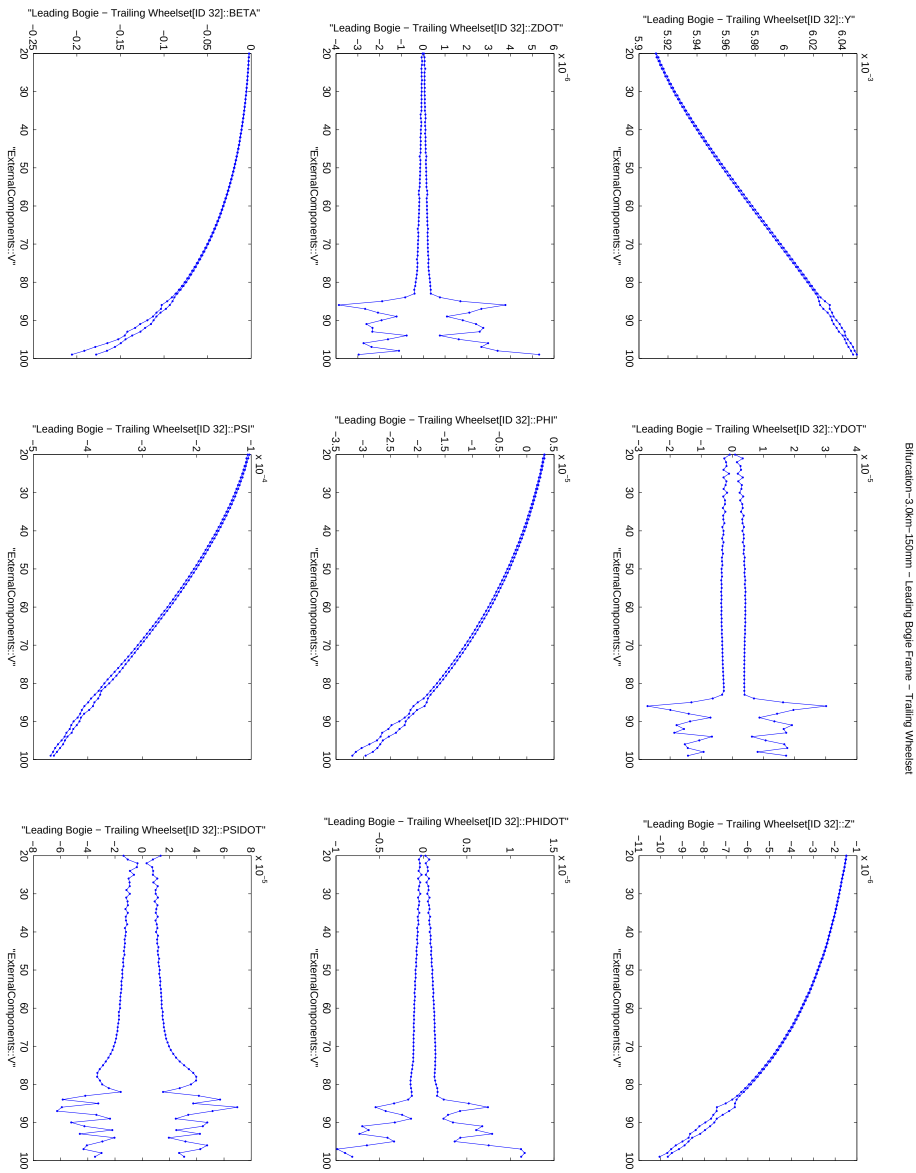


Figure F.125: AGV model: dynamics of the trailing wheel set attached to the leading bogie frame at increasing speed on a curve of 3.0km with 150mm of cant.

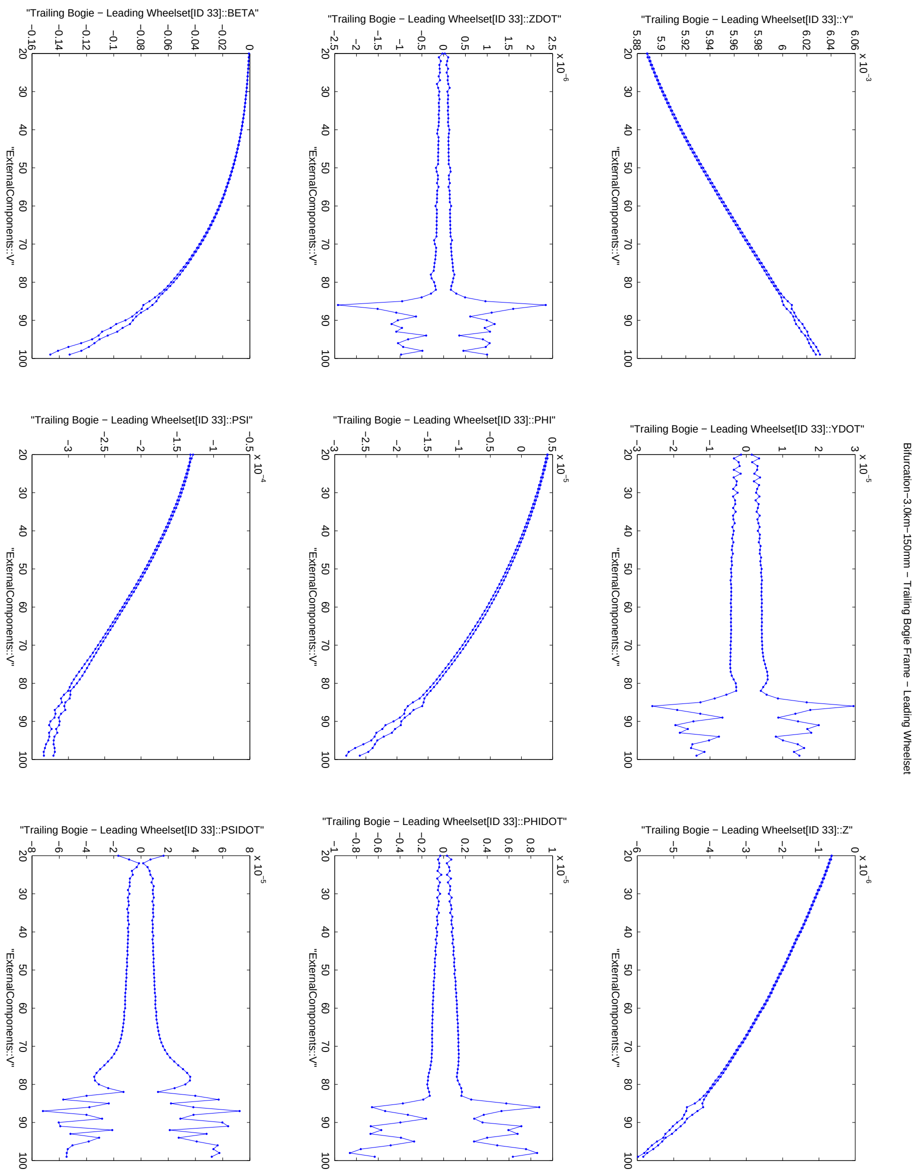


Figure F.126: AGV model: dynamics of the leading wheel set attached to the trailing bogie frame at increasing speed on a curve of 3.0km with 150mm of cant.

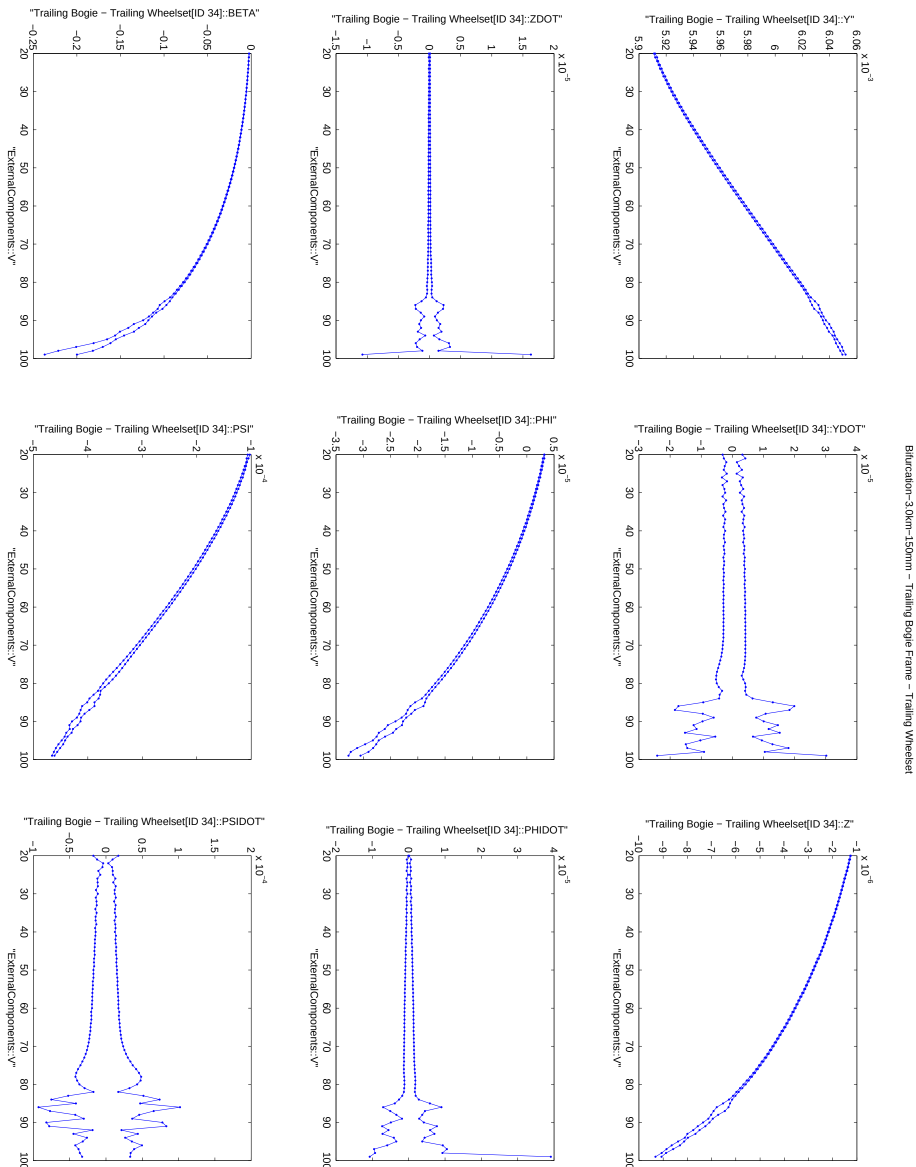


Figure F.127: AGV model: dynamics of the trailing wheel set attached to the trailing bogie frame at increasing speed on a curve of 3.0km with 150mm of cant.

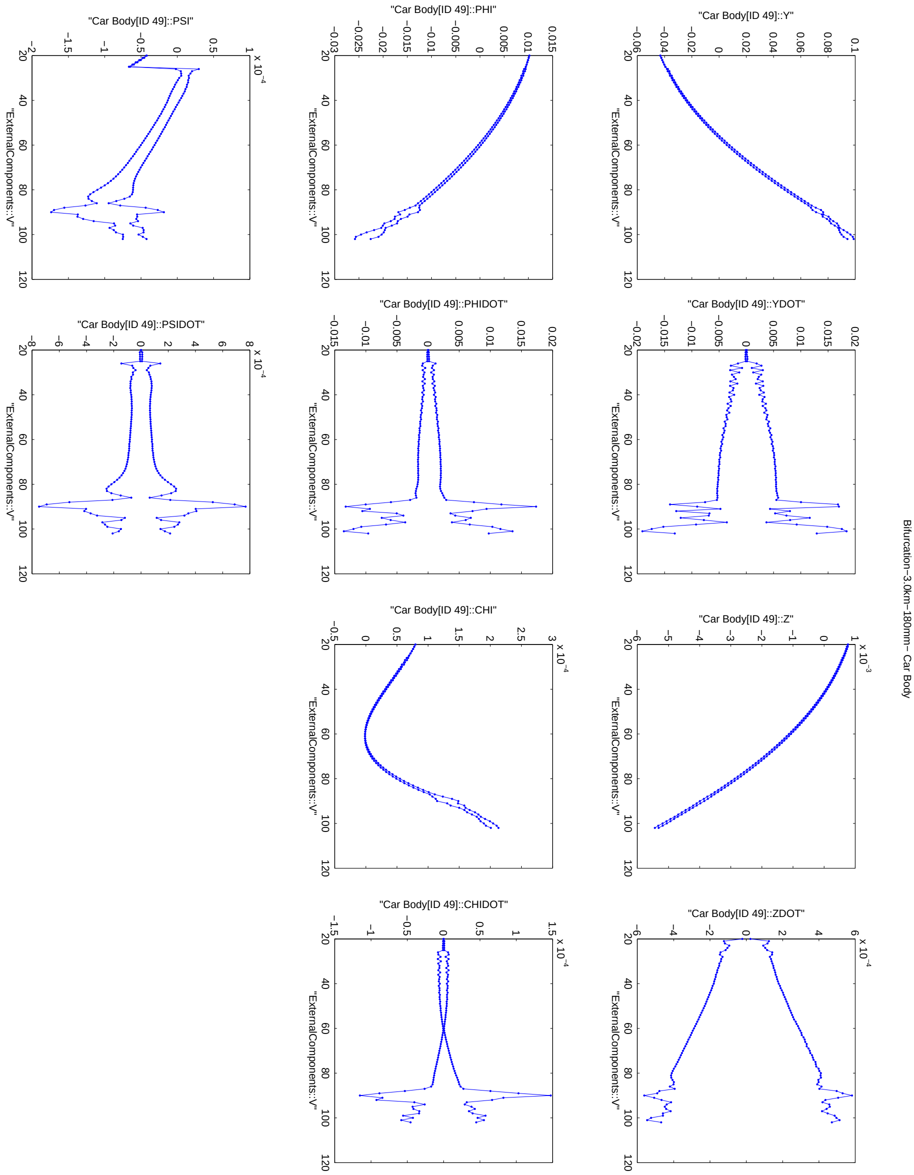


Figure F.128: AGV model: dynamics of the car body at increasing speed on a curve of 3.0km with 180mm of cant.

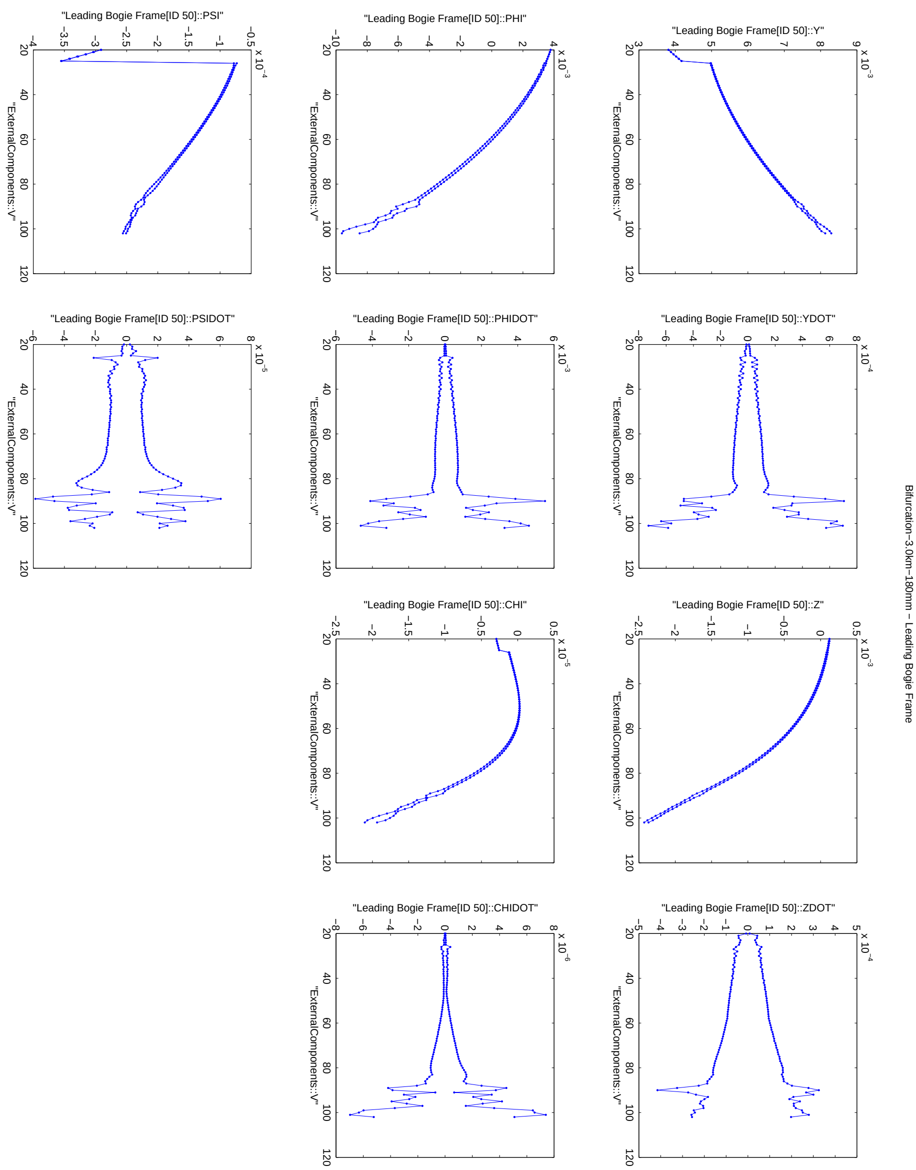


Figure F.129: AGV model: dynamics of the leading bogie frame at increasing speed on a curve of 3.0km with 180mm of cant.

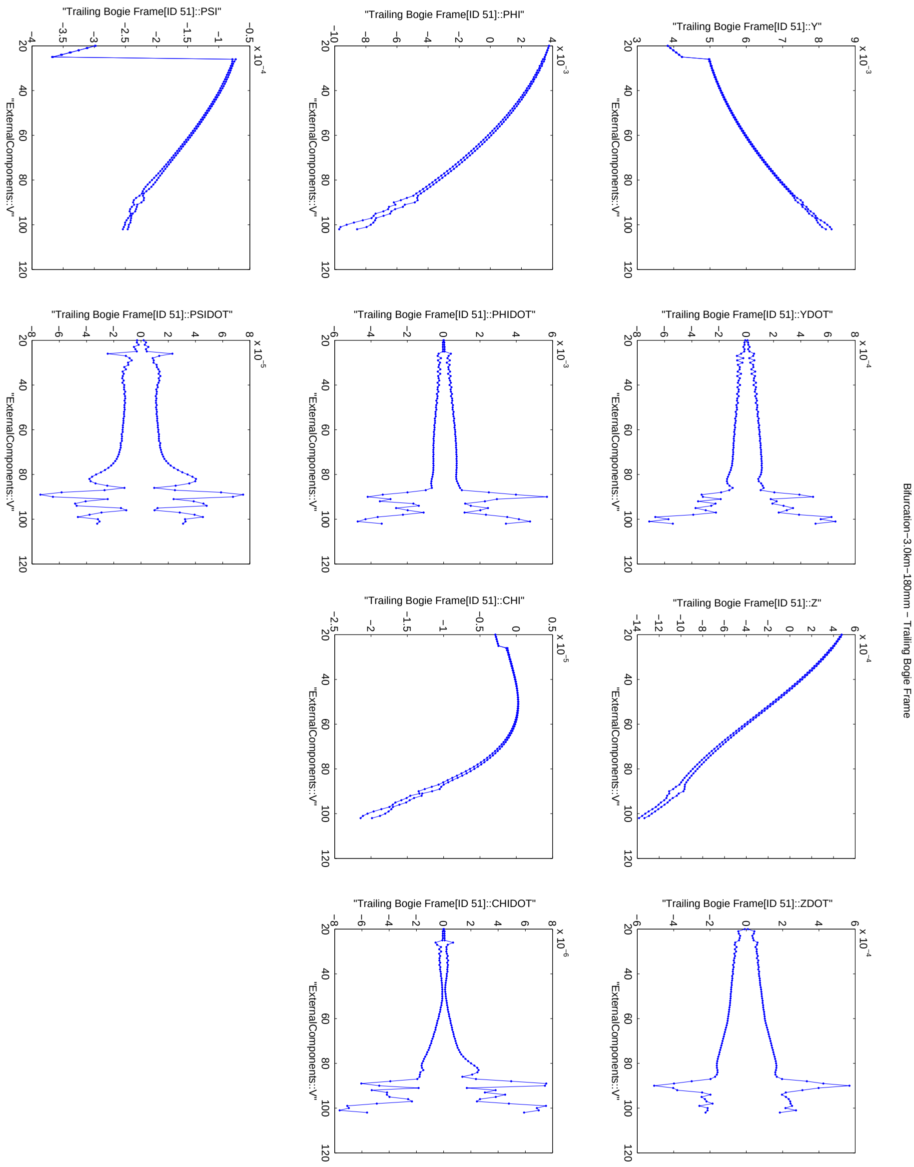


Figure F.130: AGV model: dynamics of the trailing bogie frame at increasing speed on a curve of 3.0km with 180mm of cant.

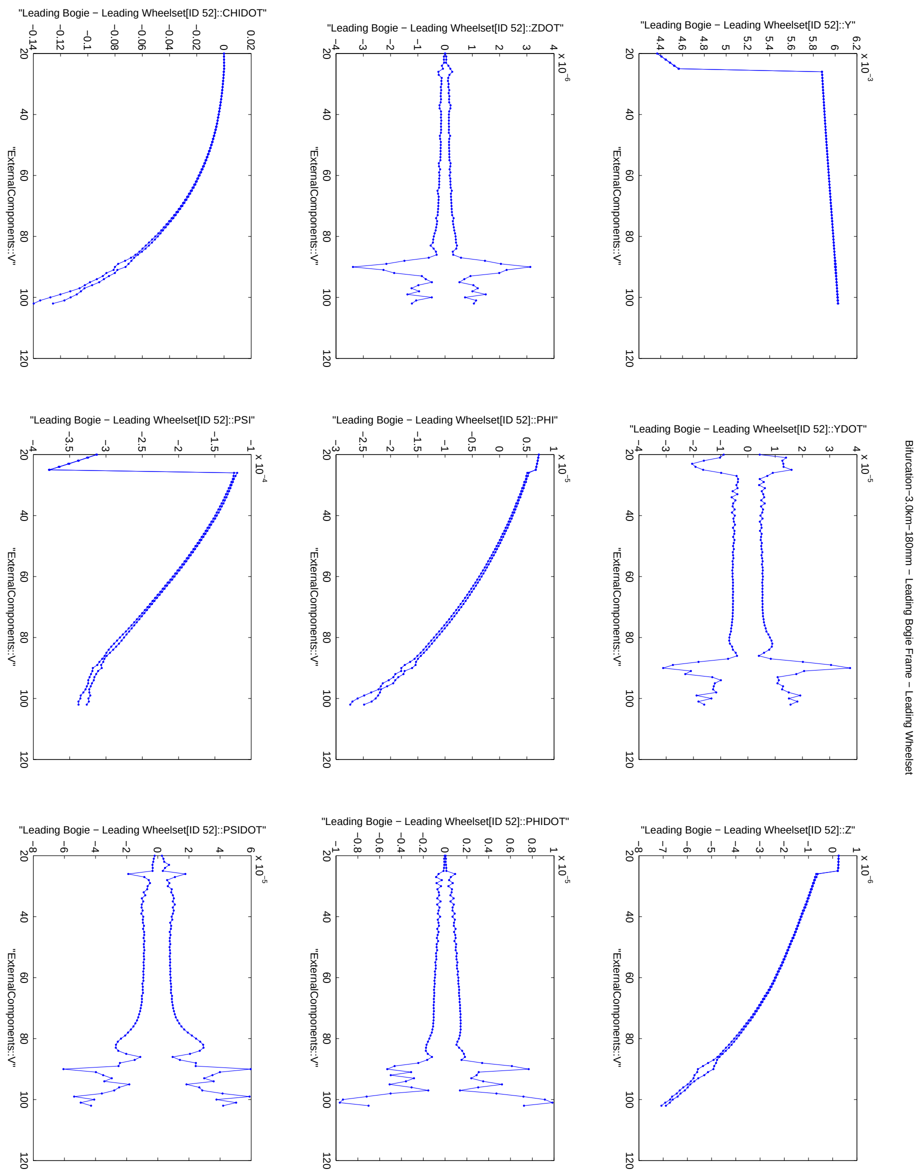


Figure F.131: AGV model: dynamics of the leading wheel set attached to the leading bogie frame at increasing speed on a curve of 3.0km with 180mm of cant.

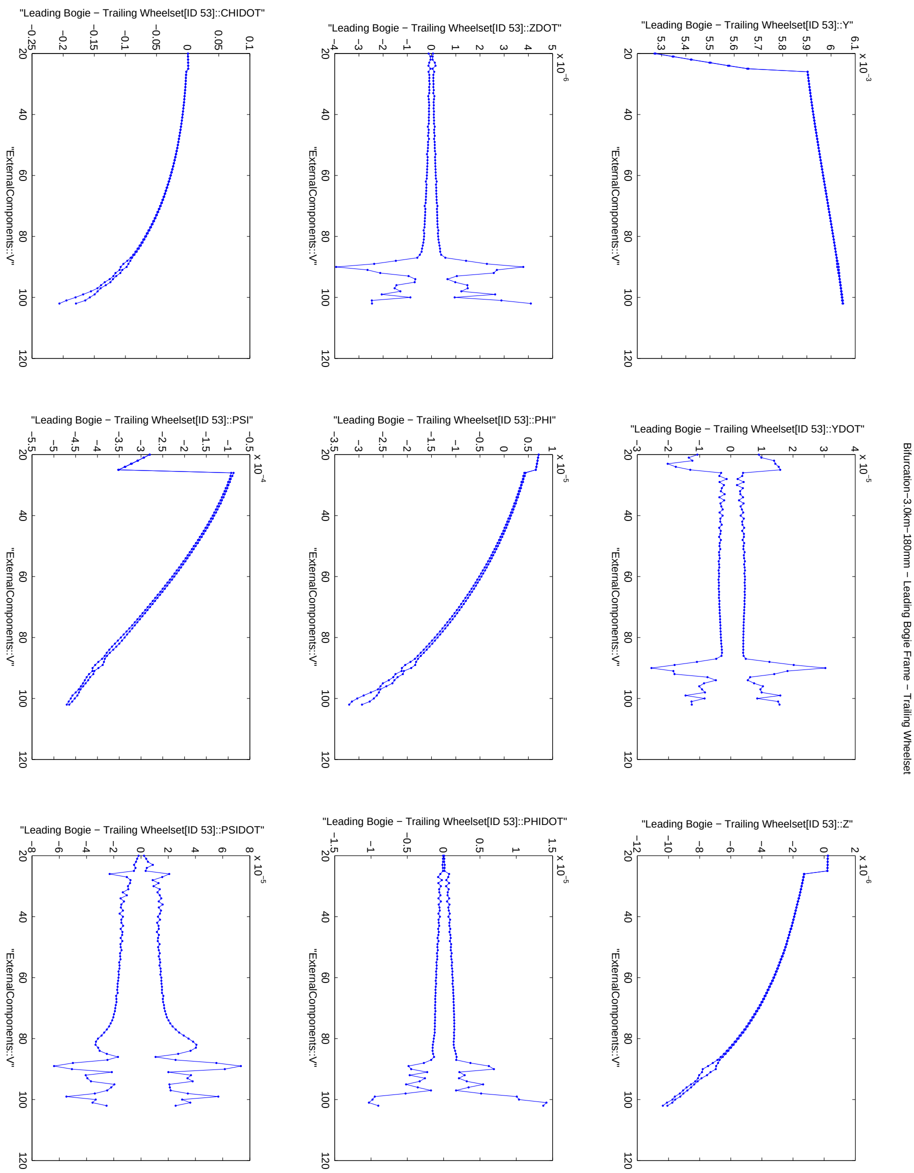


Figure F.132: AGV model: dynamics of the trailing wheel set attached to the leading bogie frame at increasing speed on a curve of 3.0km with 180mm of cant.

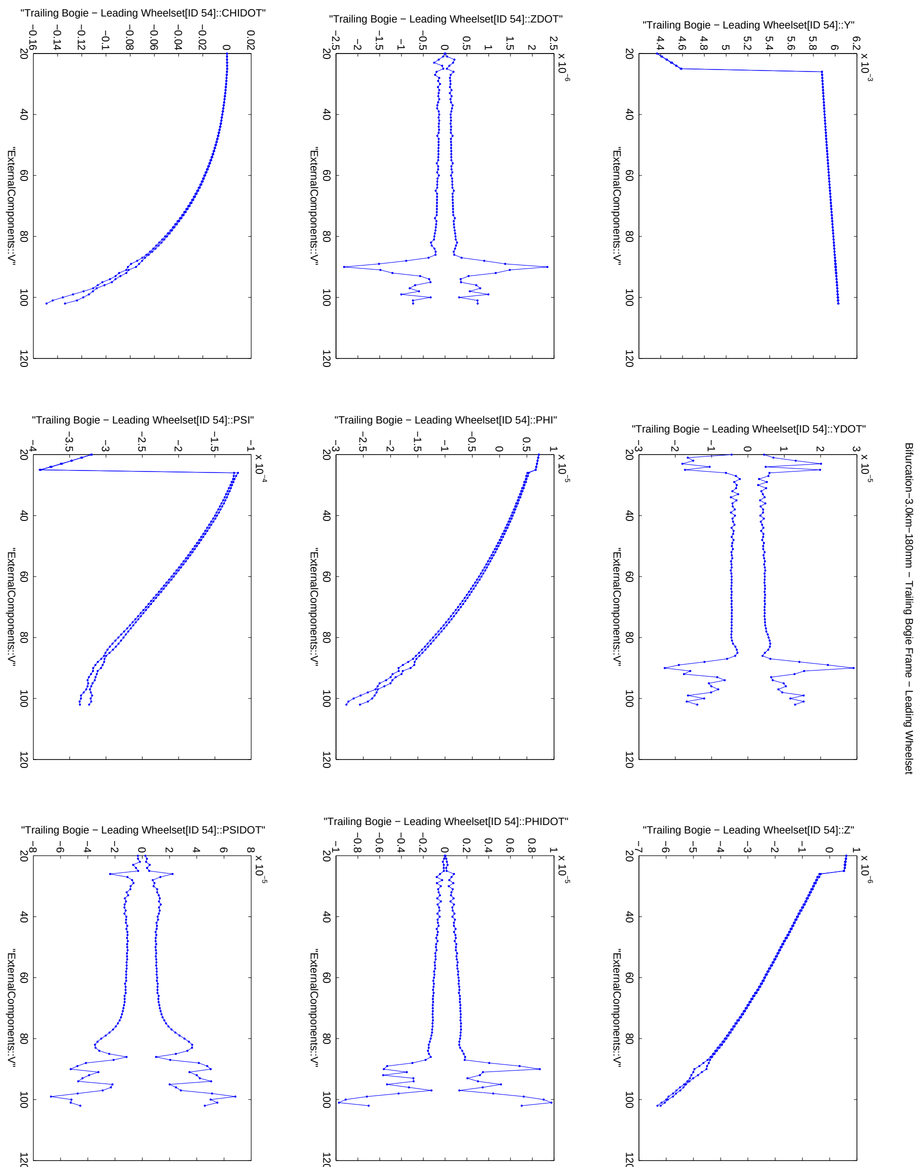


Figure F.133: AGV model: dynamics of the leading wheel set attached to the trailing bogie frame at increasing speed on a curve of 3.0km with 180mm of cant.

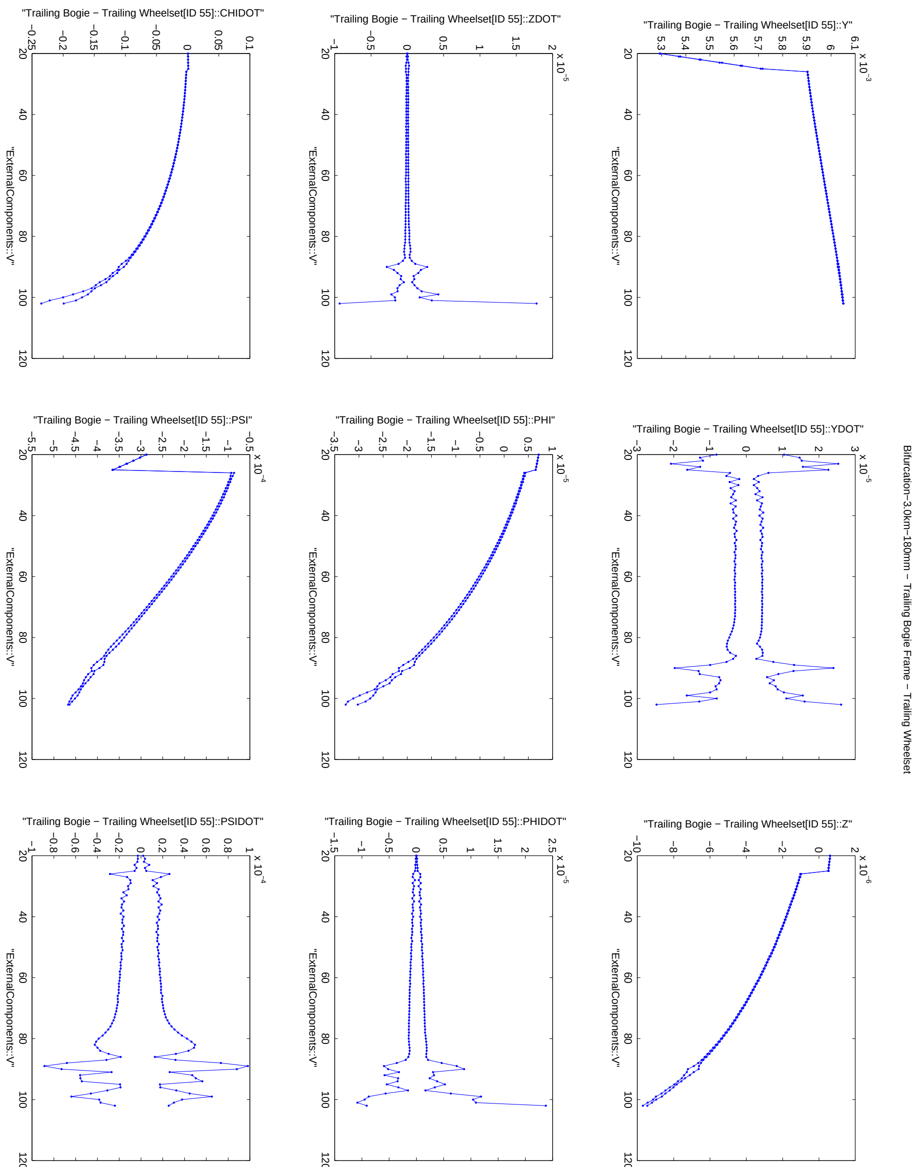


Figure F.134: AGV model: dynamics of the trailing wheel set attached to the trailing bogie frame at increasing speed on a curve of 3.0km with 180mm of cant.

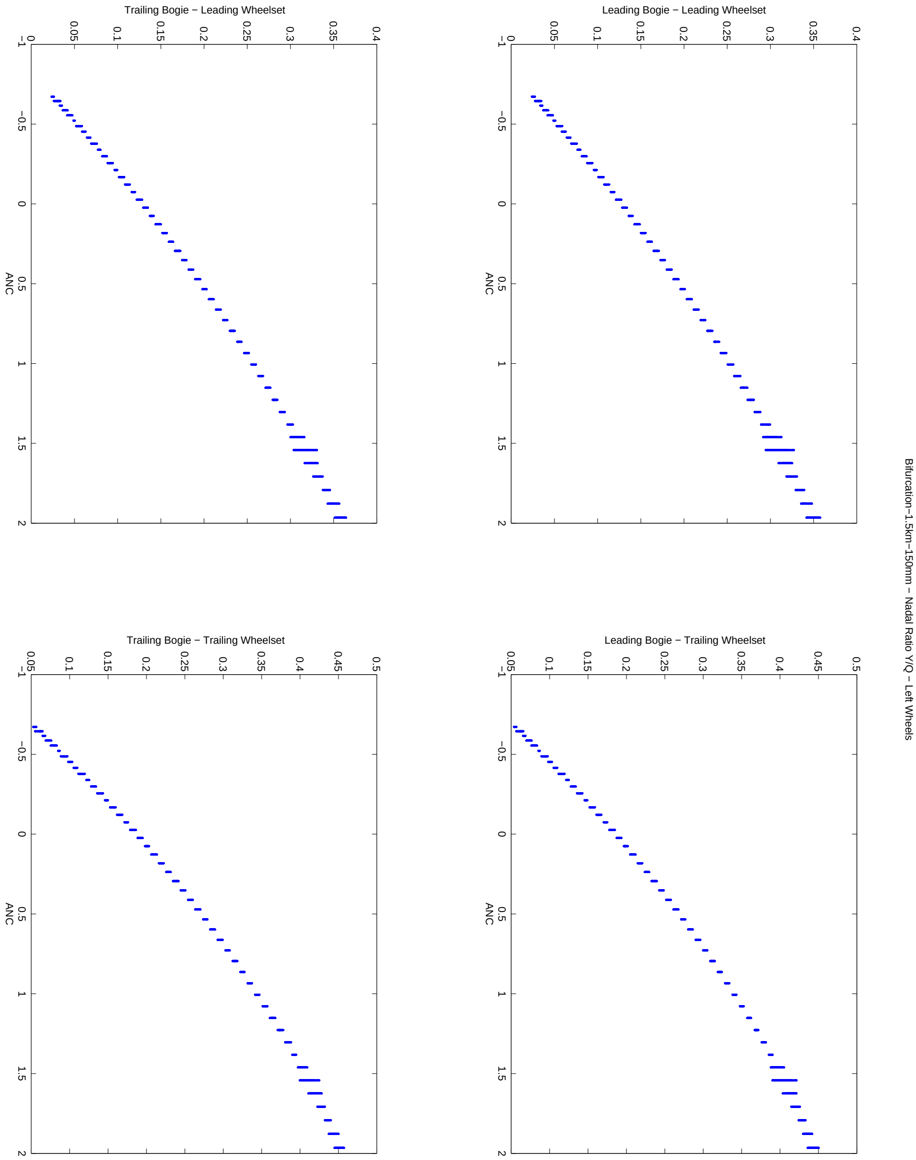


Figure F.135: AGV model: Nadal's ratio of the left wheels at increasing speed on a curve of 1.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

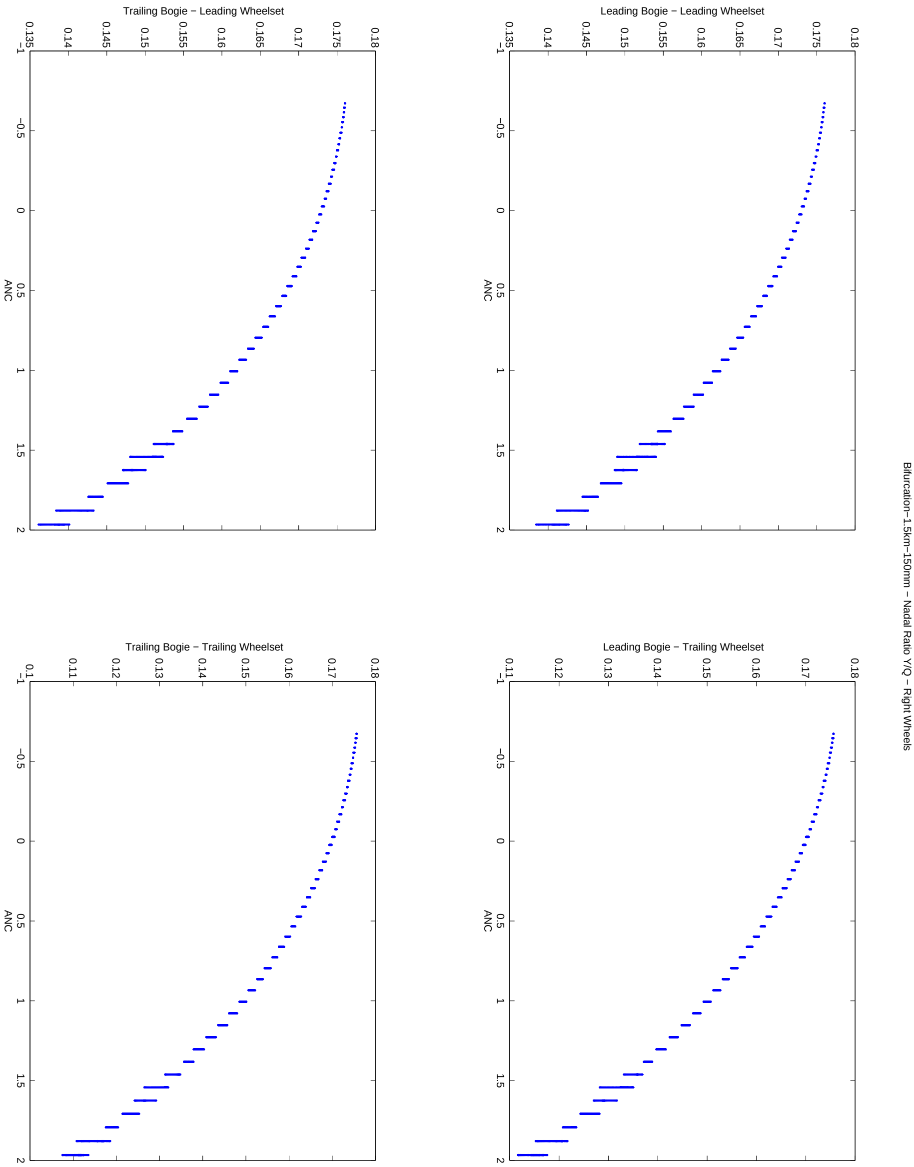


Figure F.136: AGV model: Nadal's ratio of the right wheels at increasing speed on a curve of 1.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

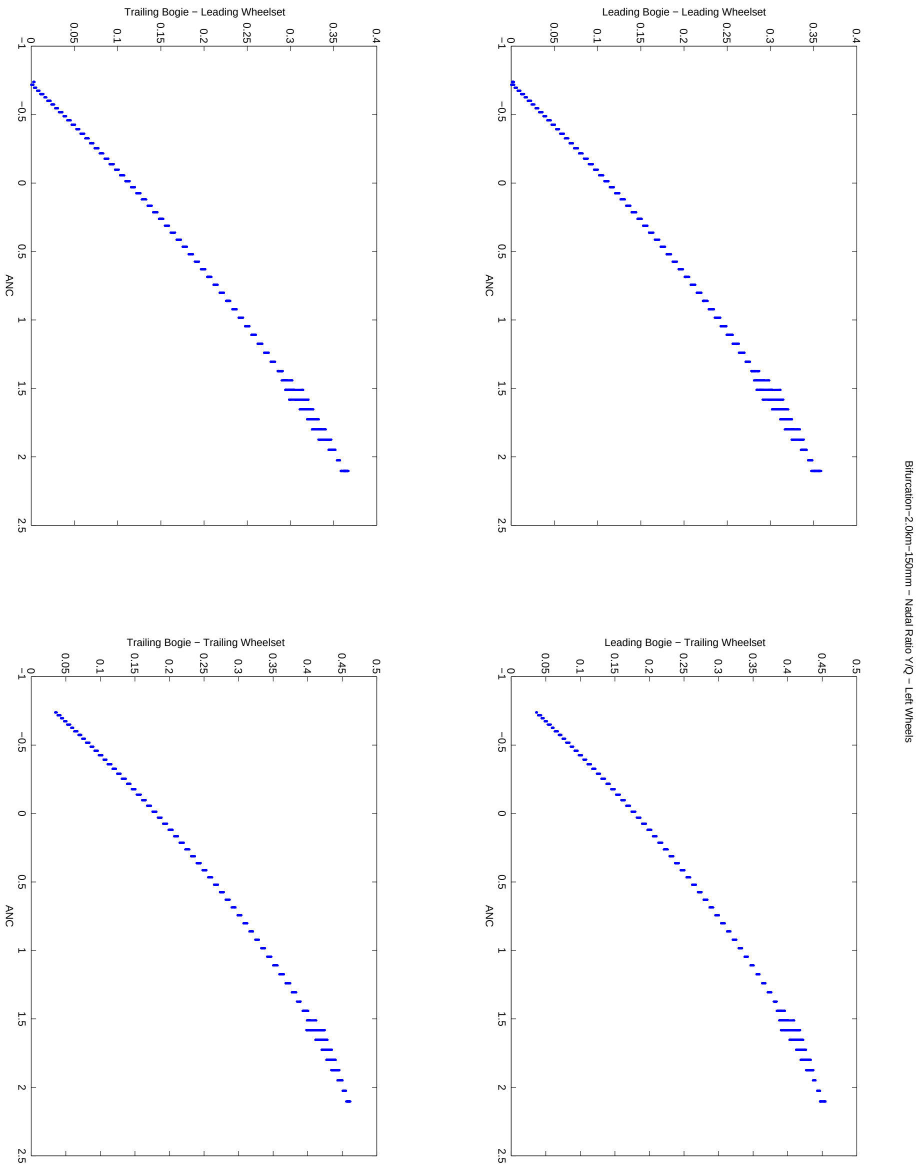


Figure F.137: AGV model: Nadal's ratio of the left wheels at increasing speed on a curve of 2.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

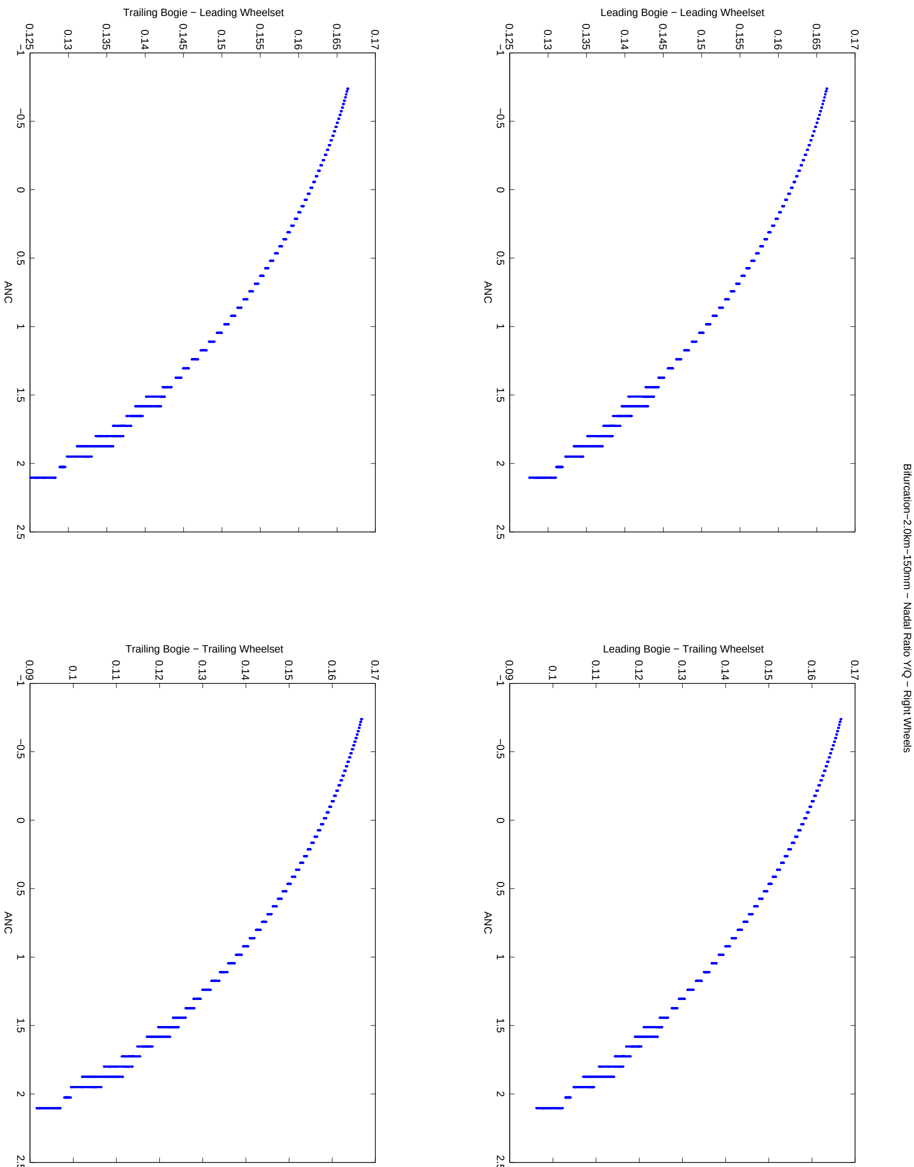


Figure F.138: AGV model: Nadal's ratio of the right wheels at increasing speed on a curve of 2.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

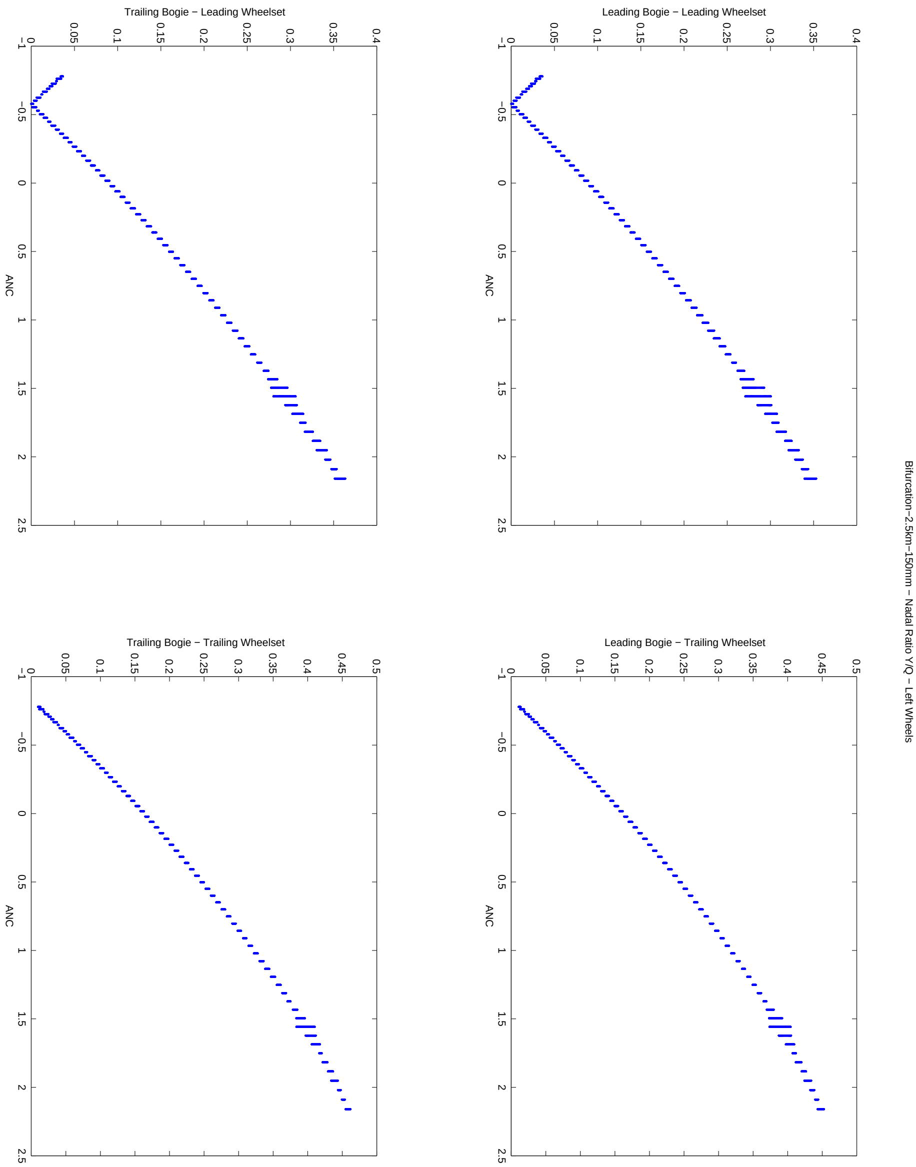


Figure F.139: AGV model: Nadal's ratio of the left wheels at increasing speed on a curve of 2.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

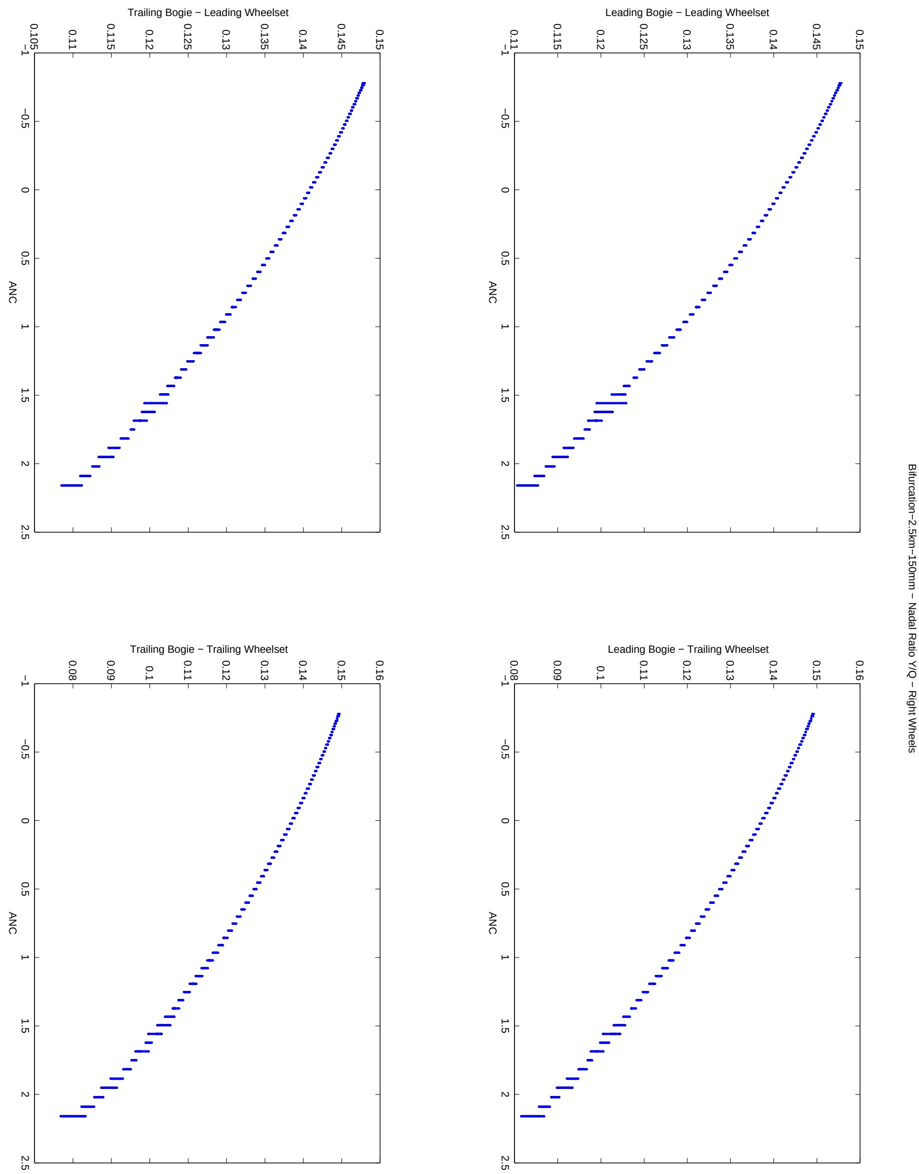


Figure F.140: AGV model: Nadal's ratio of the right wheels at increasing speed on a curve of 2.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

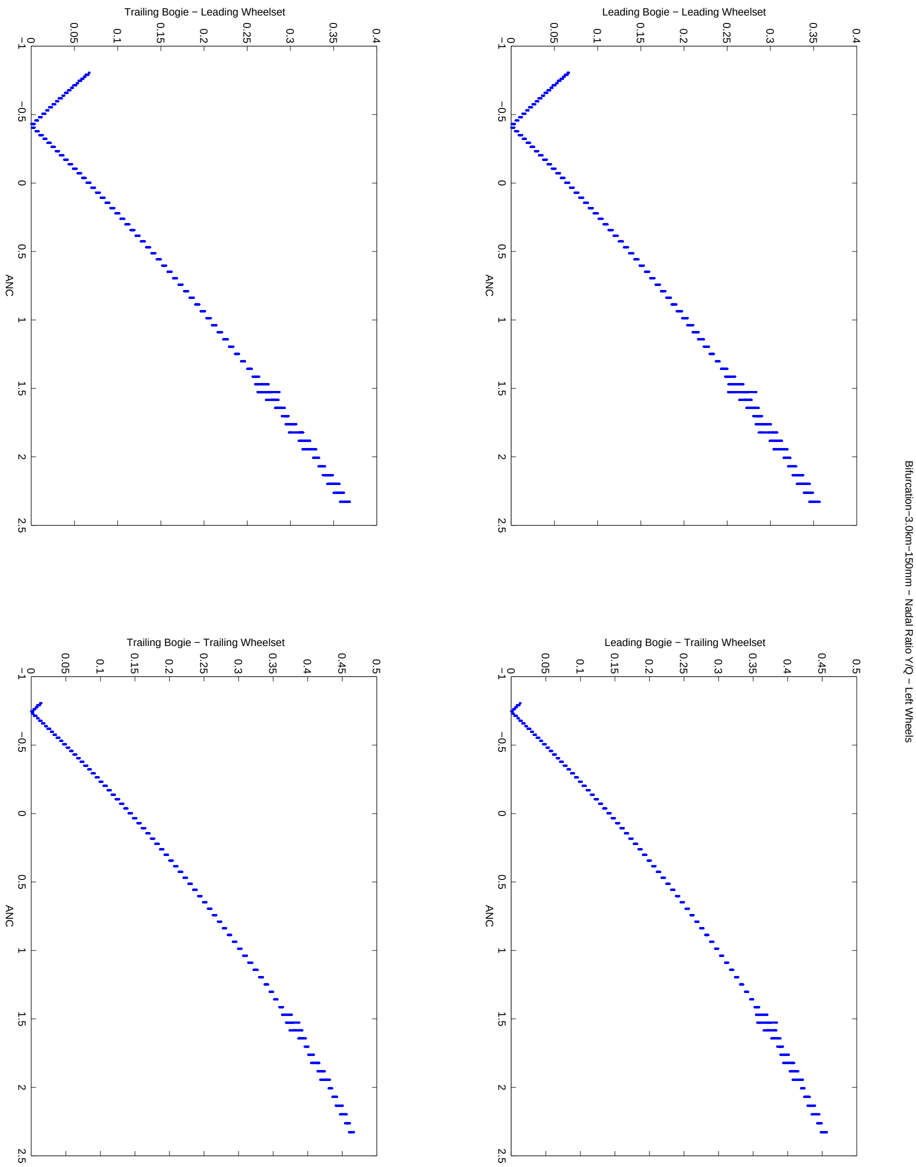


Figure F.141: AGV model: Nadal's ratio of the left wheels at increasing speed on a curve of 3.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

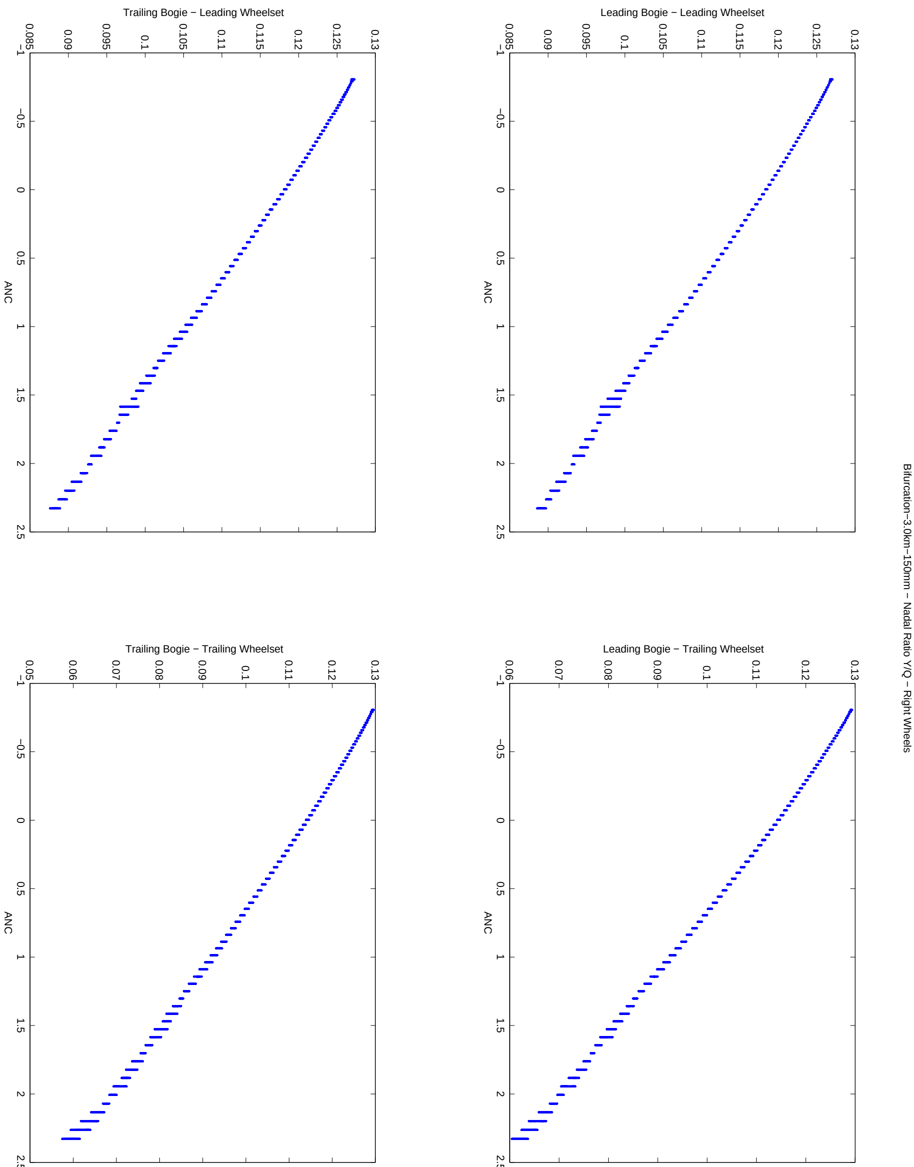


Figure F.142: AGV model: Nadal's ratio of the right wheels at increasing speed on a curve of 3.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

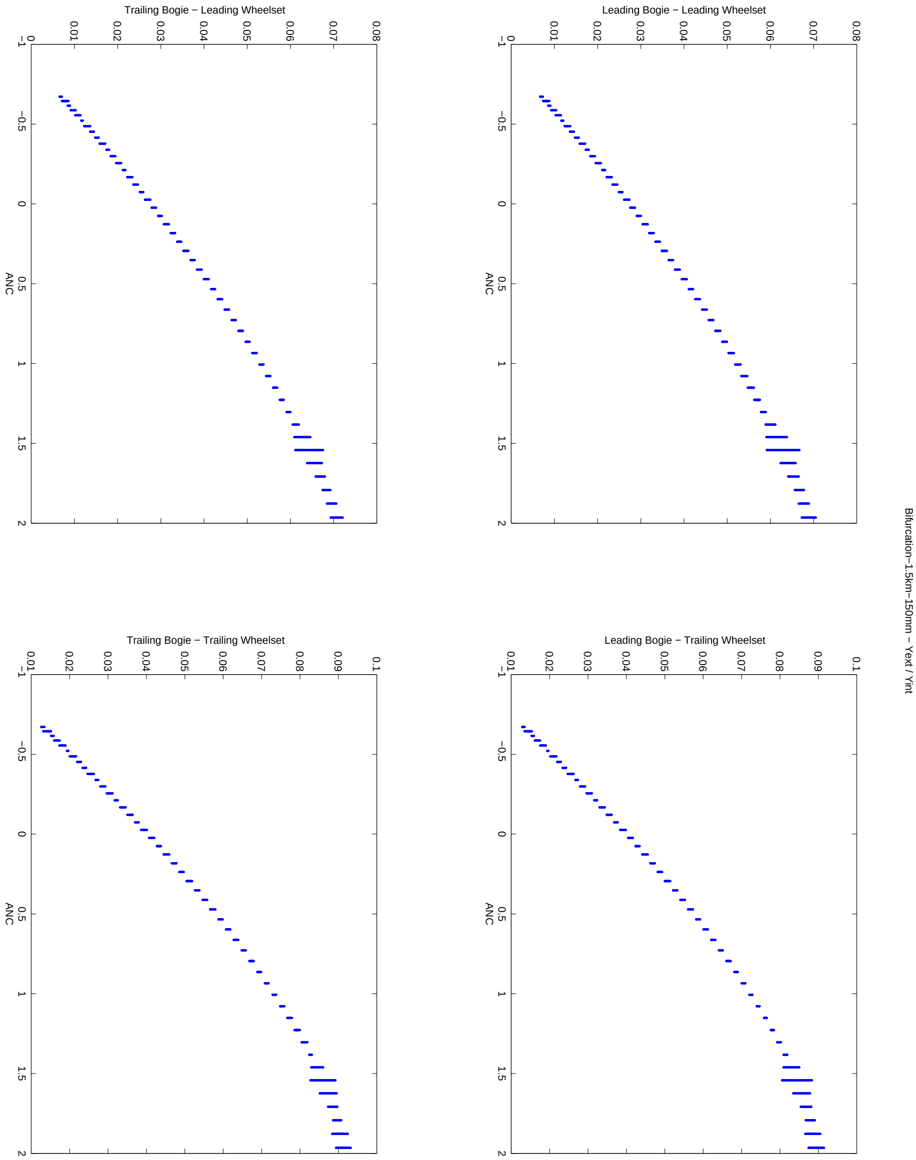


Figure F.143: AGV model: ratio between lateral forces on the outer and the inner wheels $\frac{Y_{ext}}{Y_{int}}$ at increasing speed on a curve of 1.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

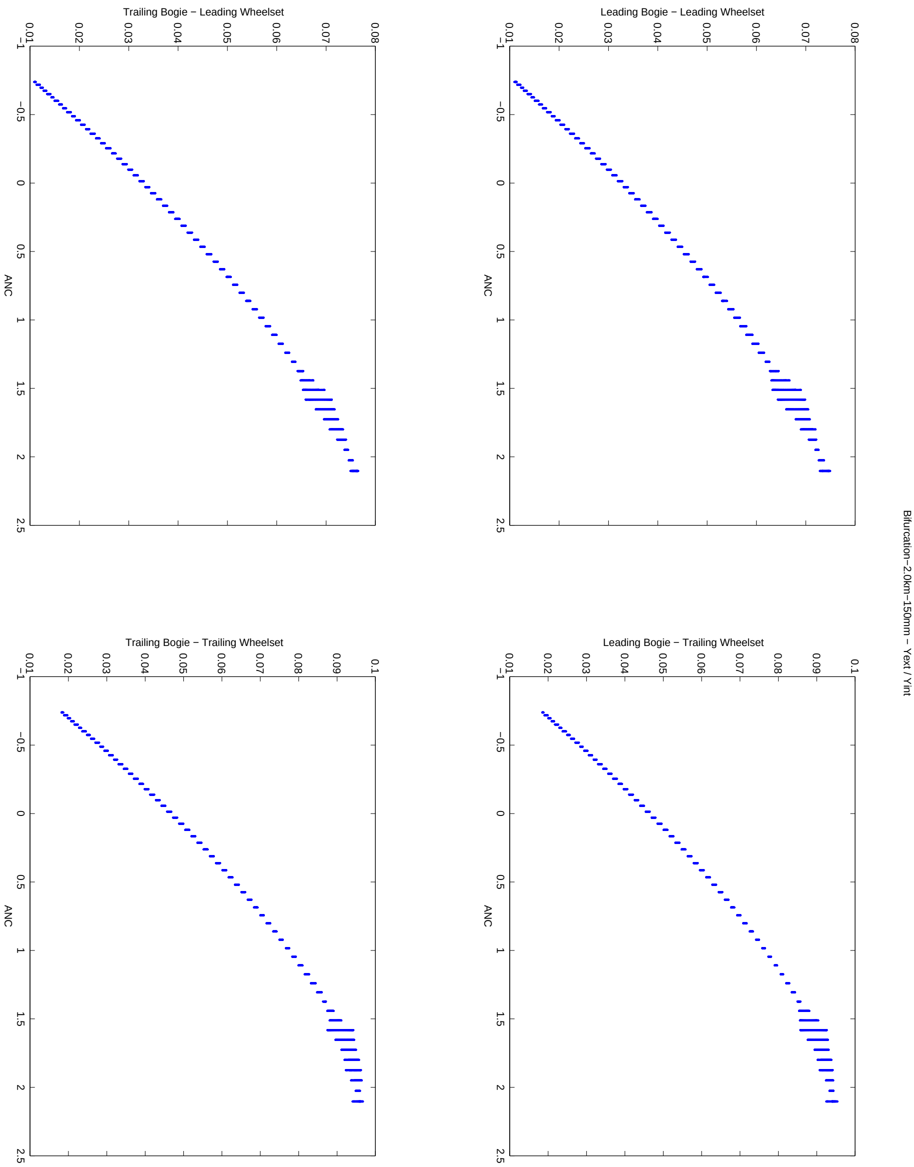


Figure F.144: AGV model: ratio between lateral forces on the outer and the inner wheels $\frac{Y_{ext}}{Y_{int}}$ at increasing speed on a curve of 2.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

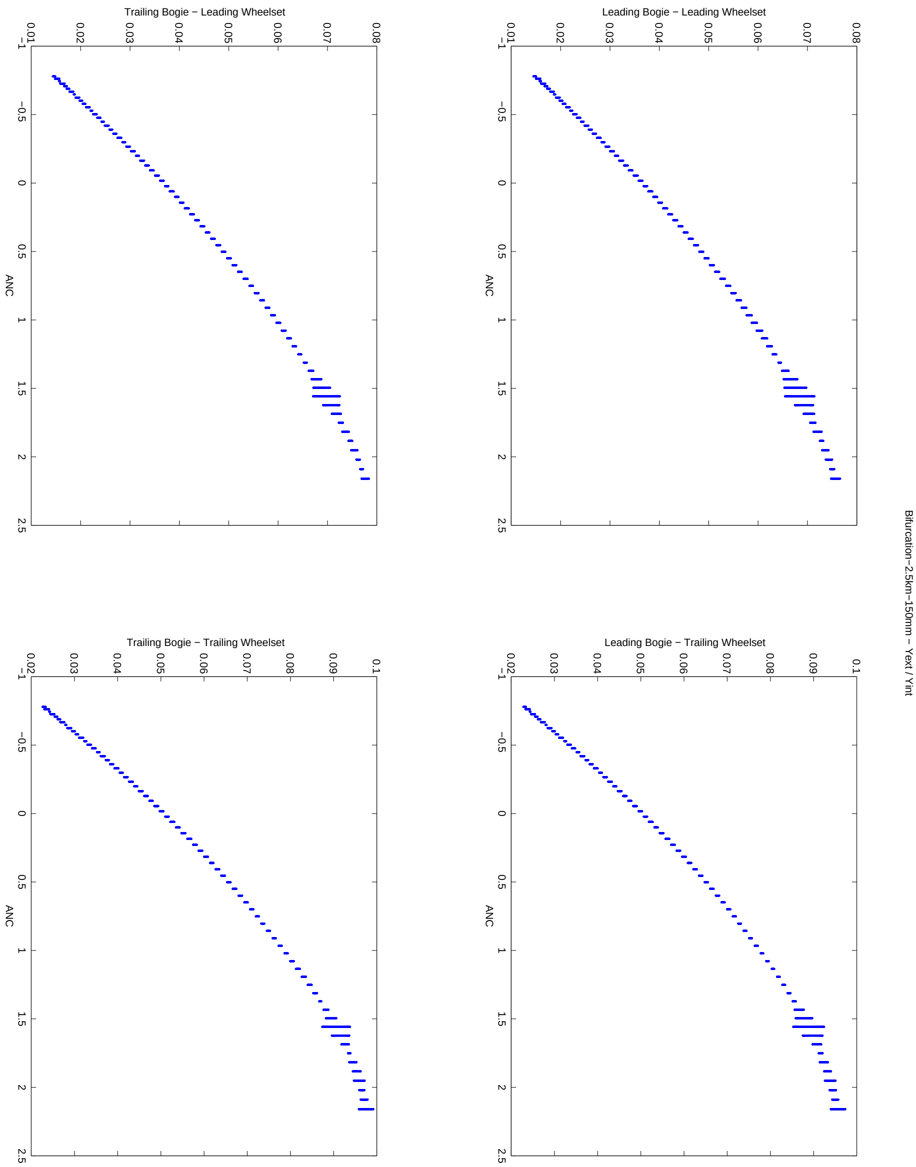


Figure F.145: AGV model: ratio between lateral forces on the outer and the inner wheels $\frac{Y_{ext}}{Y_{int}}$ at increasing speed on a curve of 2.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

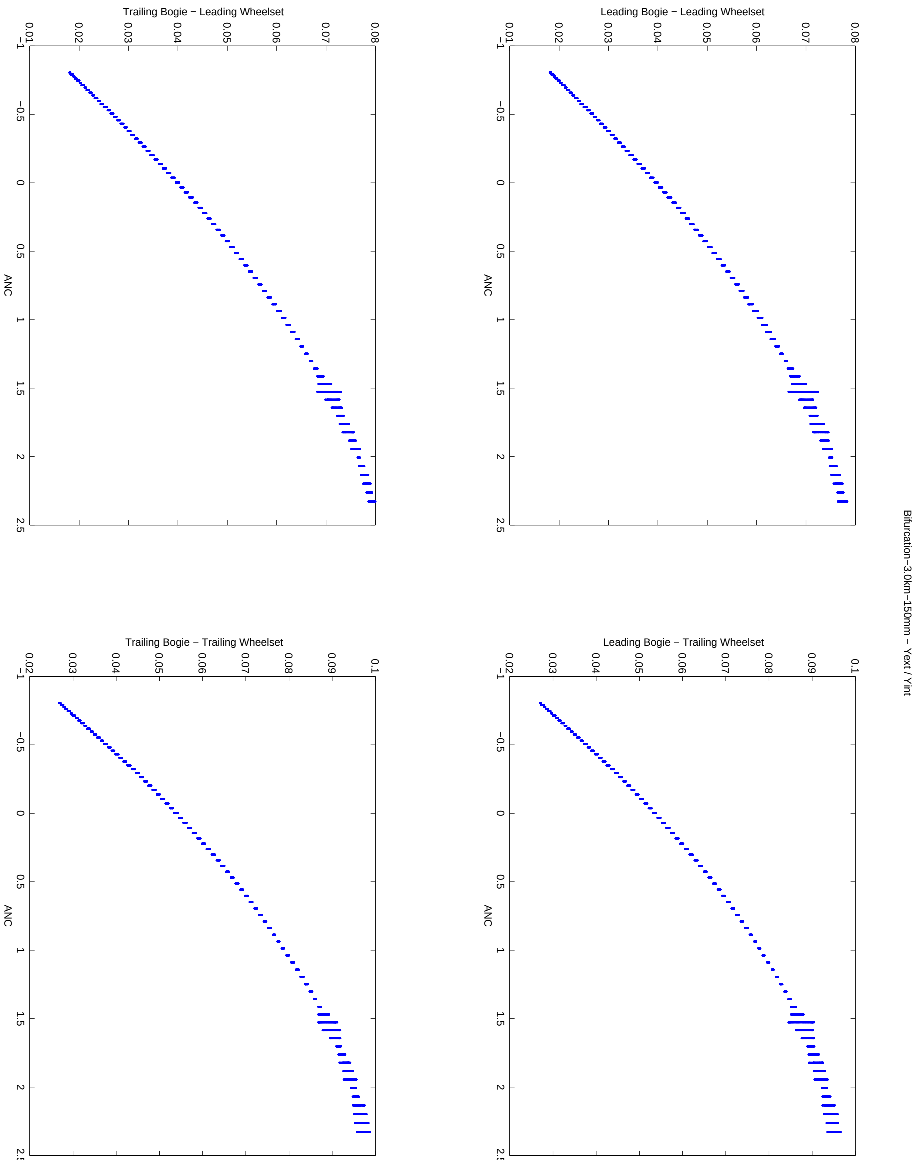


Figure F.146: AGV model: ratio between lateral forces on the outer and the inner wheels $\frac{Y_{ext}}{Y_{int}}$ at increasing speed on a curve of 3.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

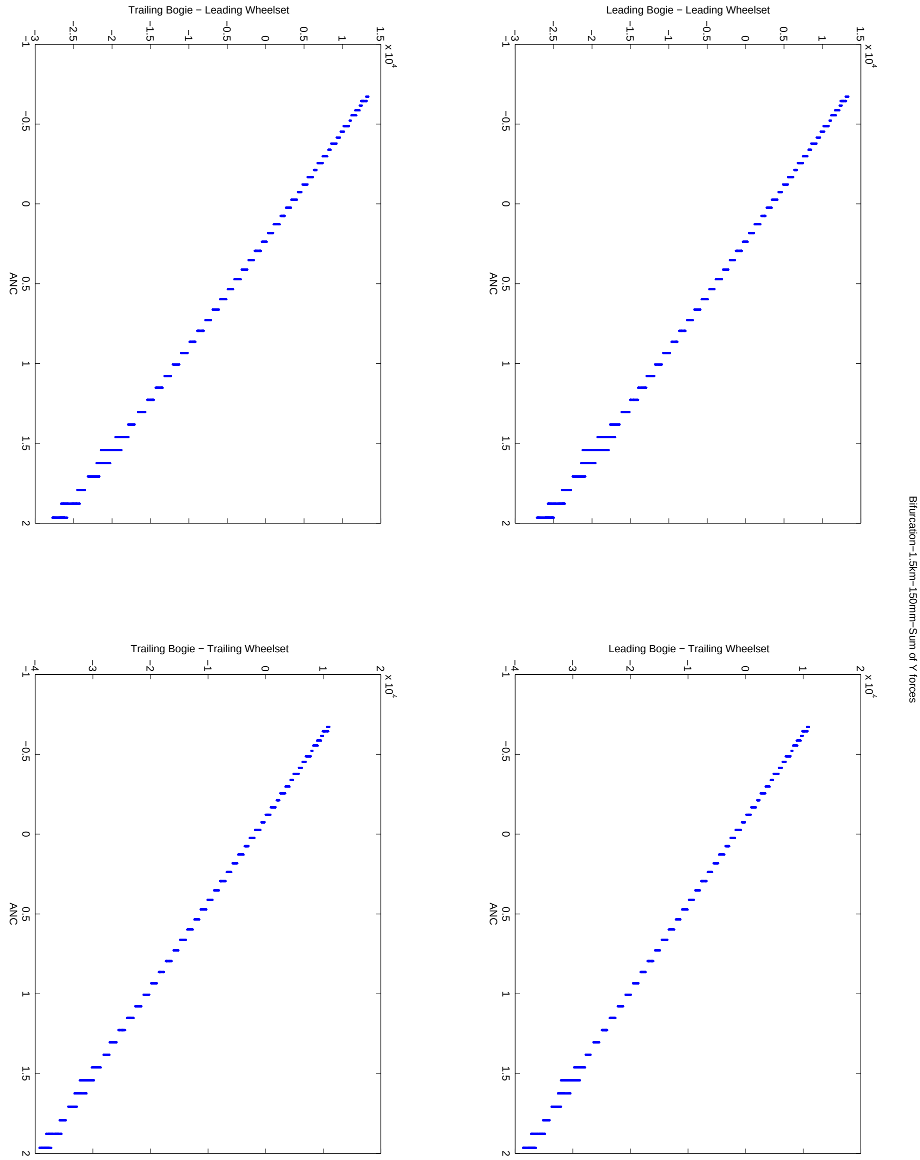


Figure F.147: AGV model: sum of the lateral forces (internal plus external) per wheel set at increasing speed on a curve of 1.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

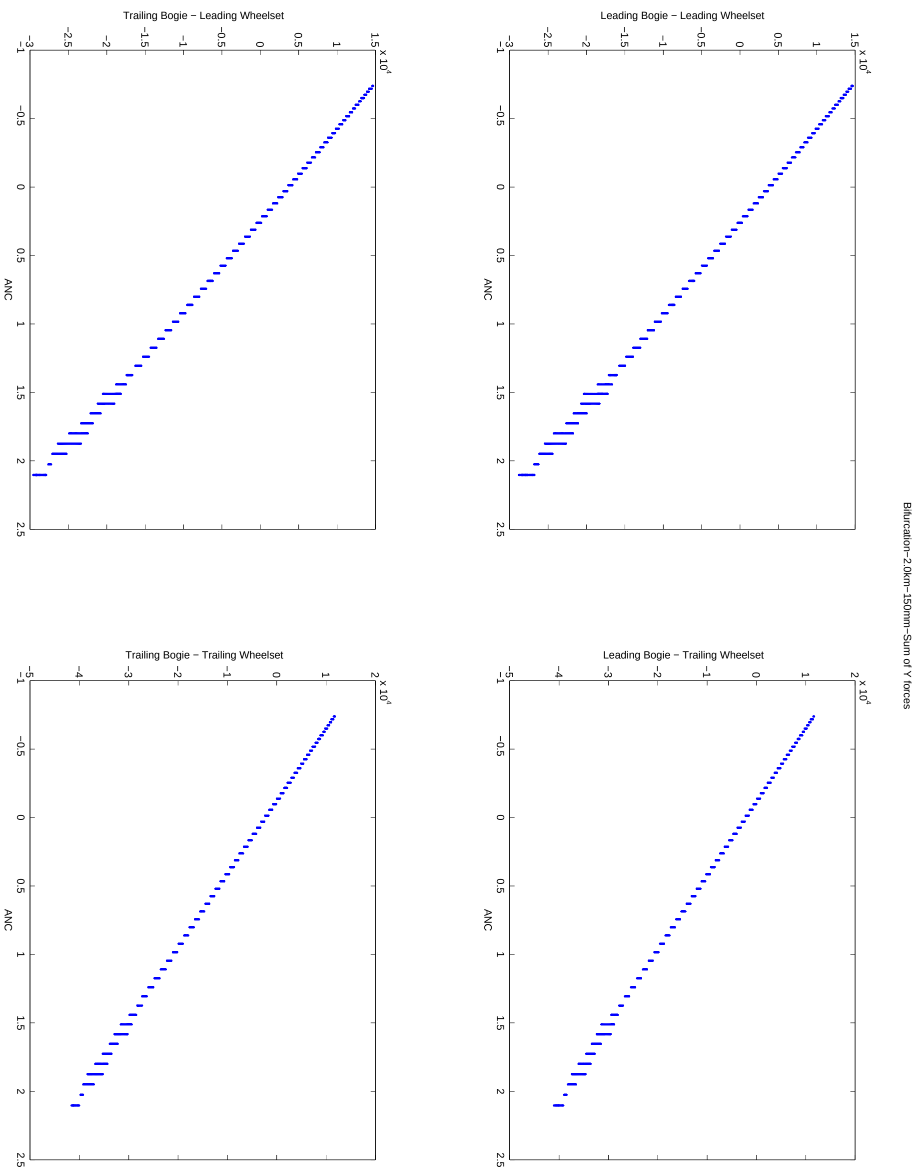


Figure F.148: AGV model: sum of the lateral forces (internal plus external) per wheel set at increasing speed on a curve of 2.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

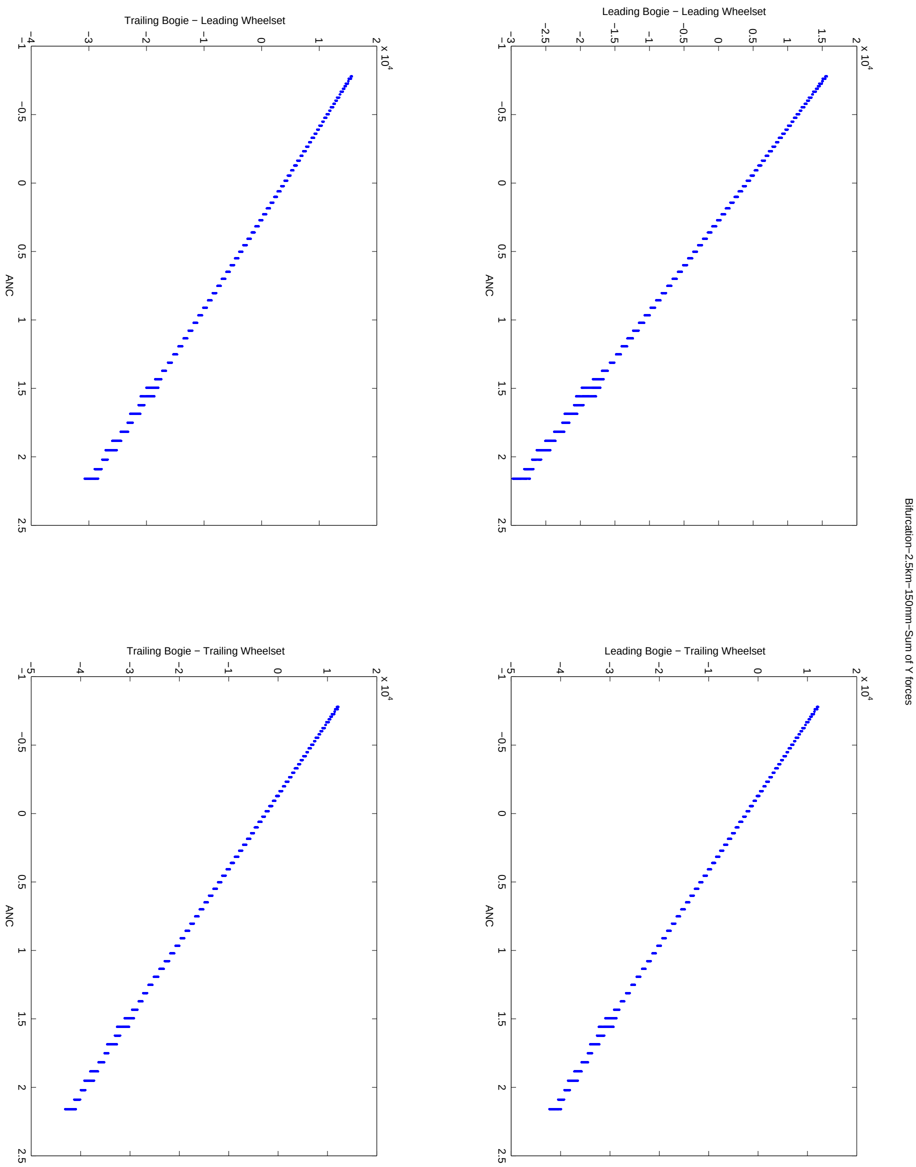


Figure F.149: AGV model: sum of the lateral forces (internal plus external) per wheel set at increasing speed on a curve of 2.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

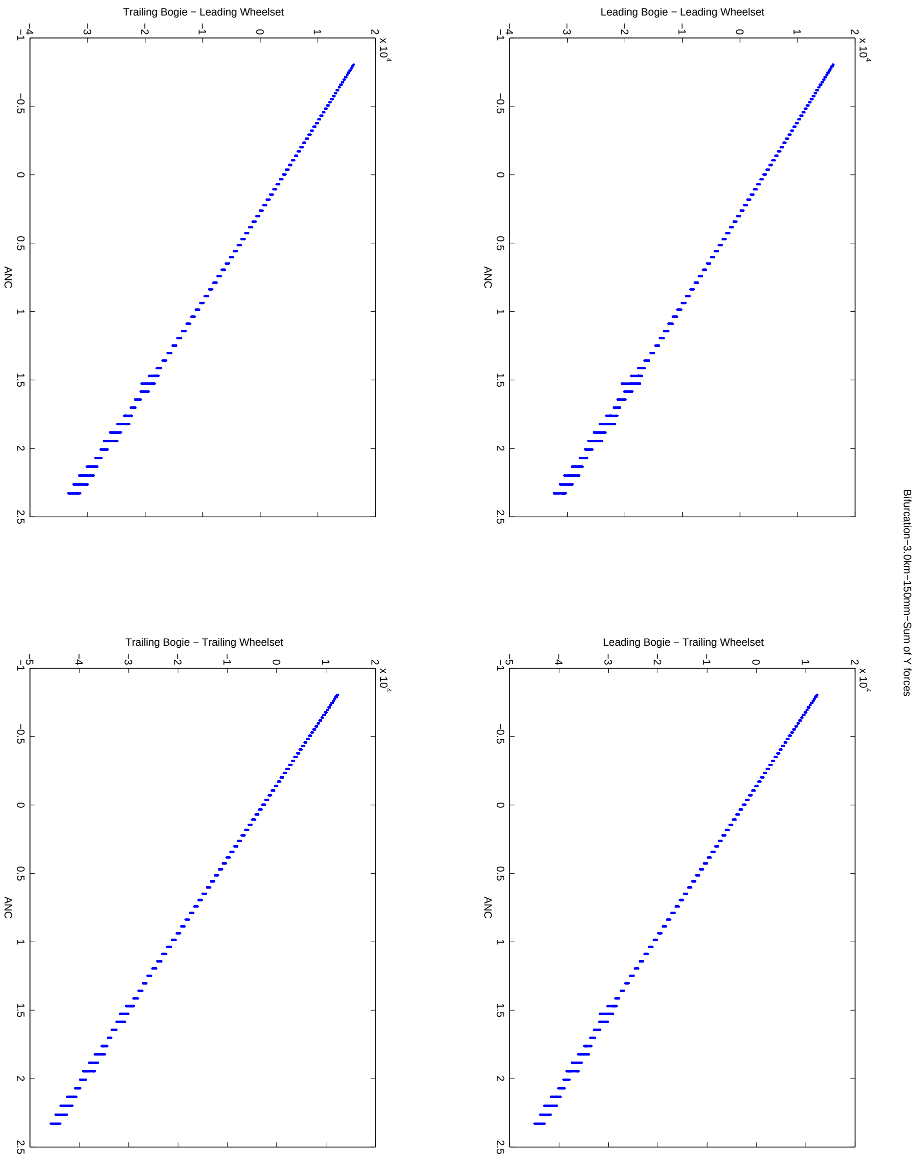


Figure F.150: AGV model: sum of the lateral forces (internal plus external) per wheel set at increasing speed on a curve of 3.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

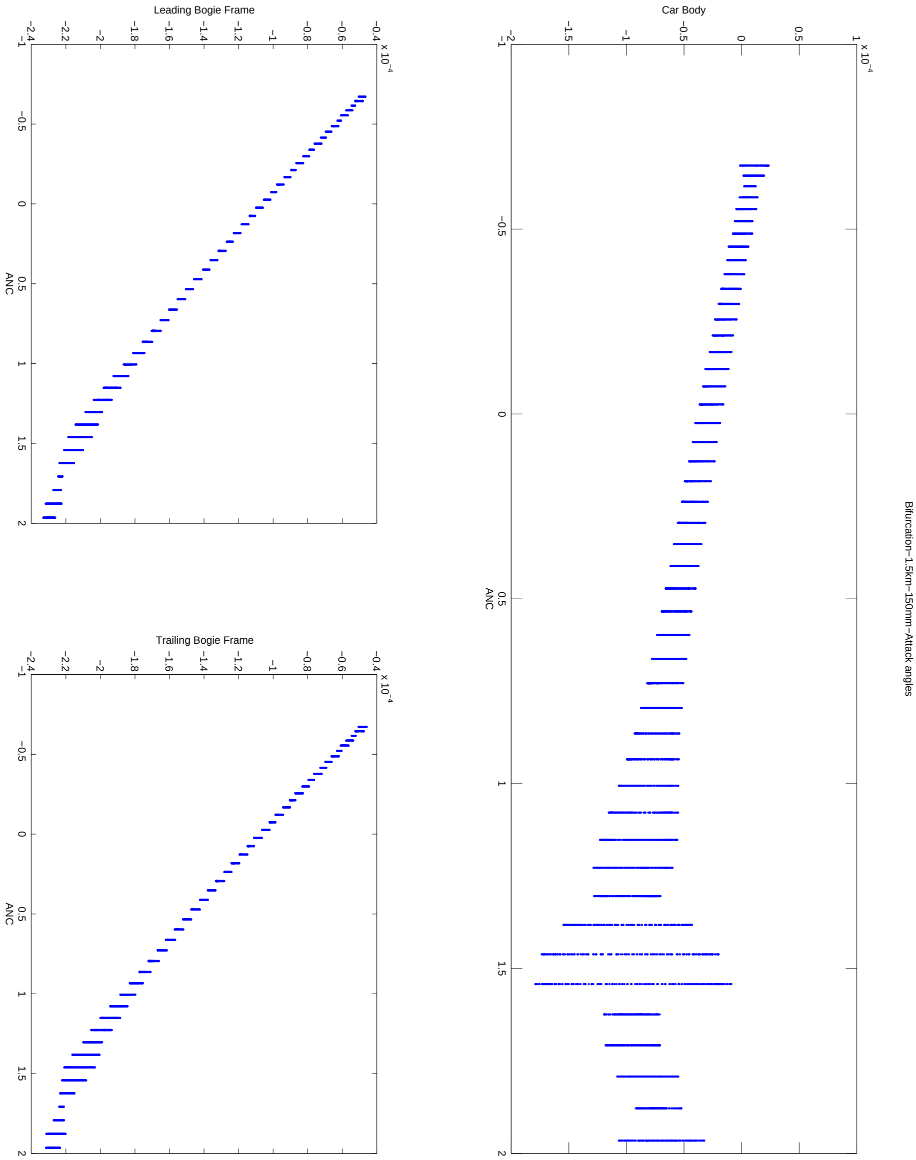


Figure F.151: AGV model: evolution of the attack angles of the car body and the bogie frames at increasing speed on a curve of 1.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

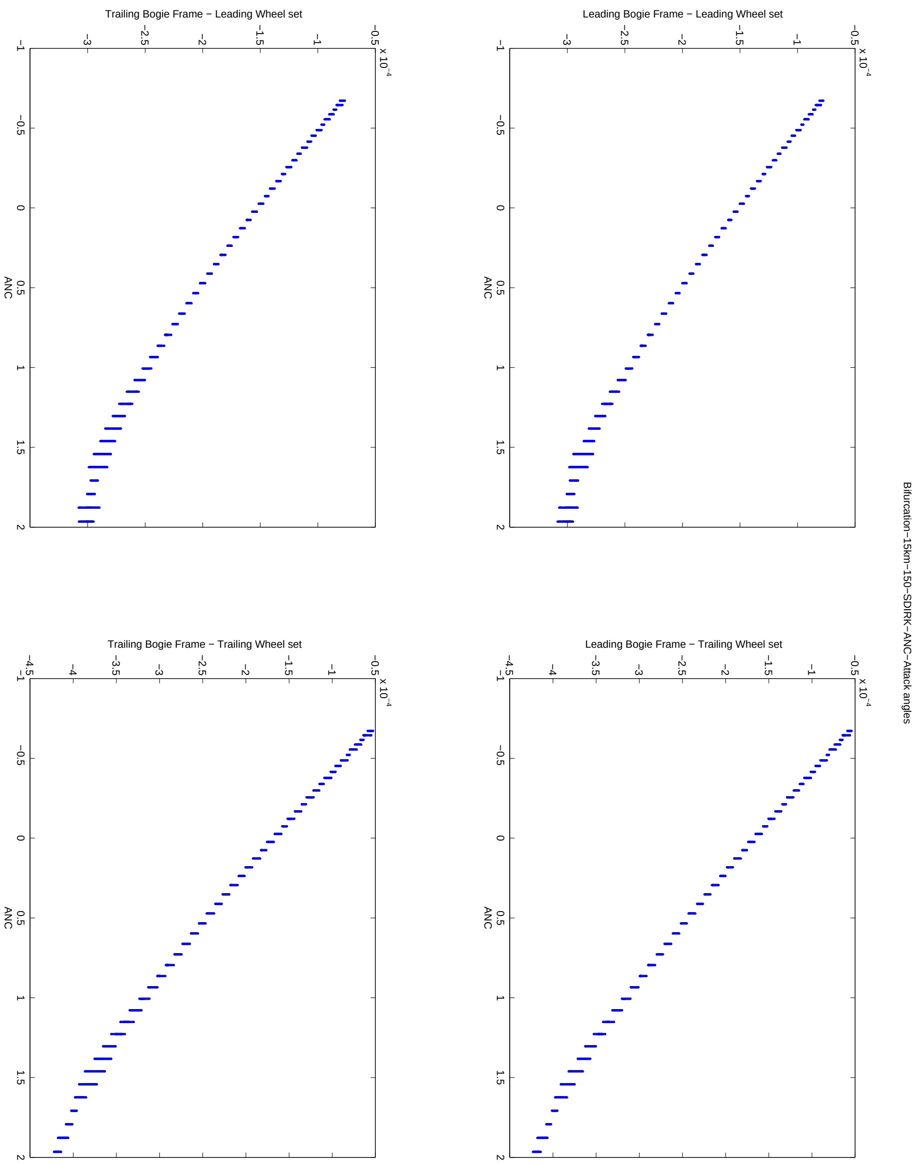


Figure F.152: AGV model: evolution of the attack angles of all the wheel sets at increasing speed on a curve of 1.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

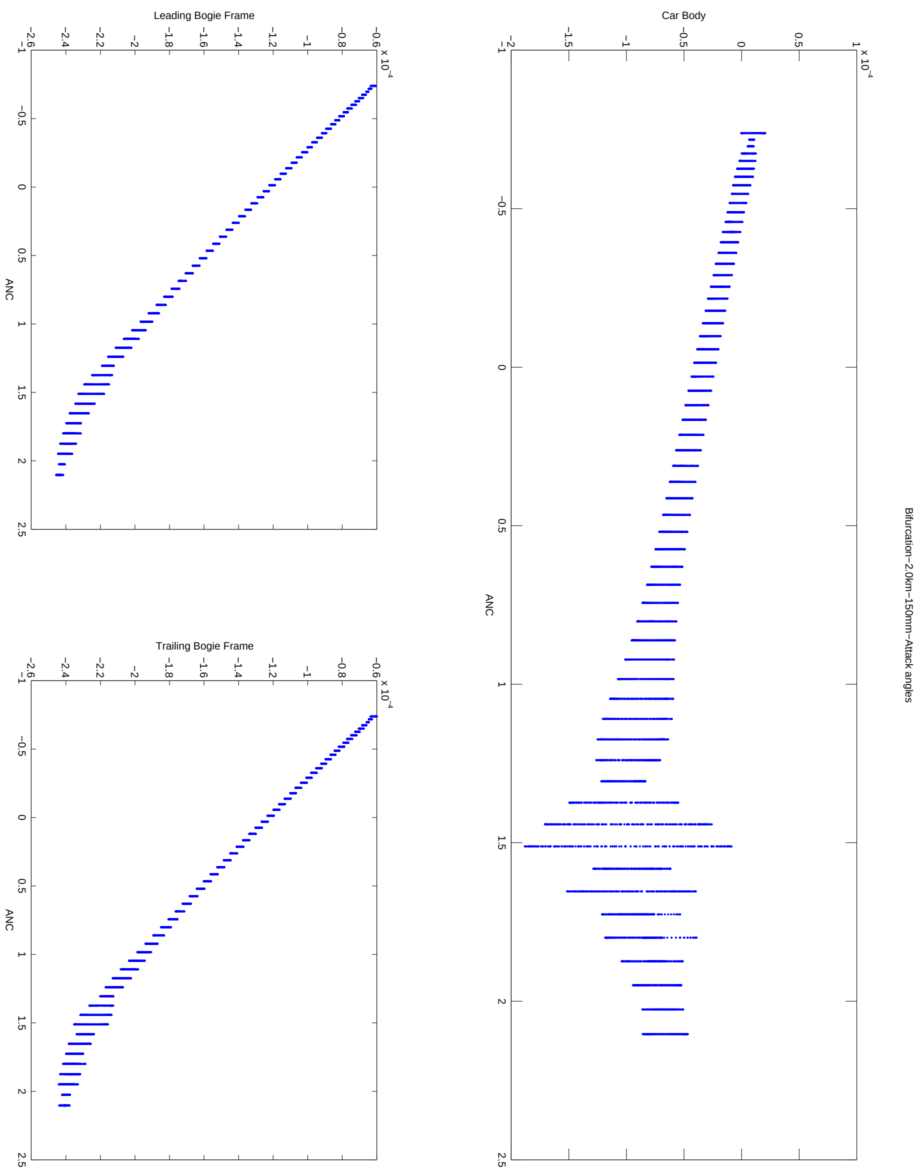


Figure F.153: AGV model: evolution of the attack angles of the car body and the bogie frames at increasing speed on a curve of 2.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

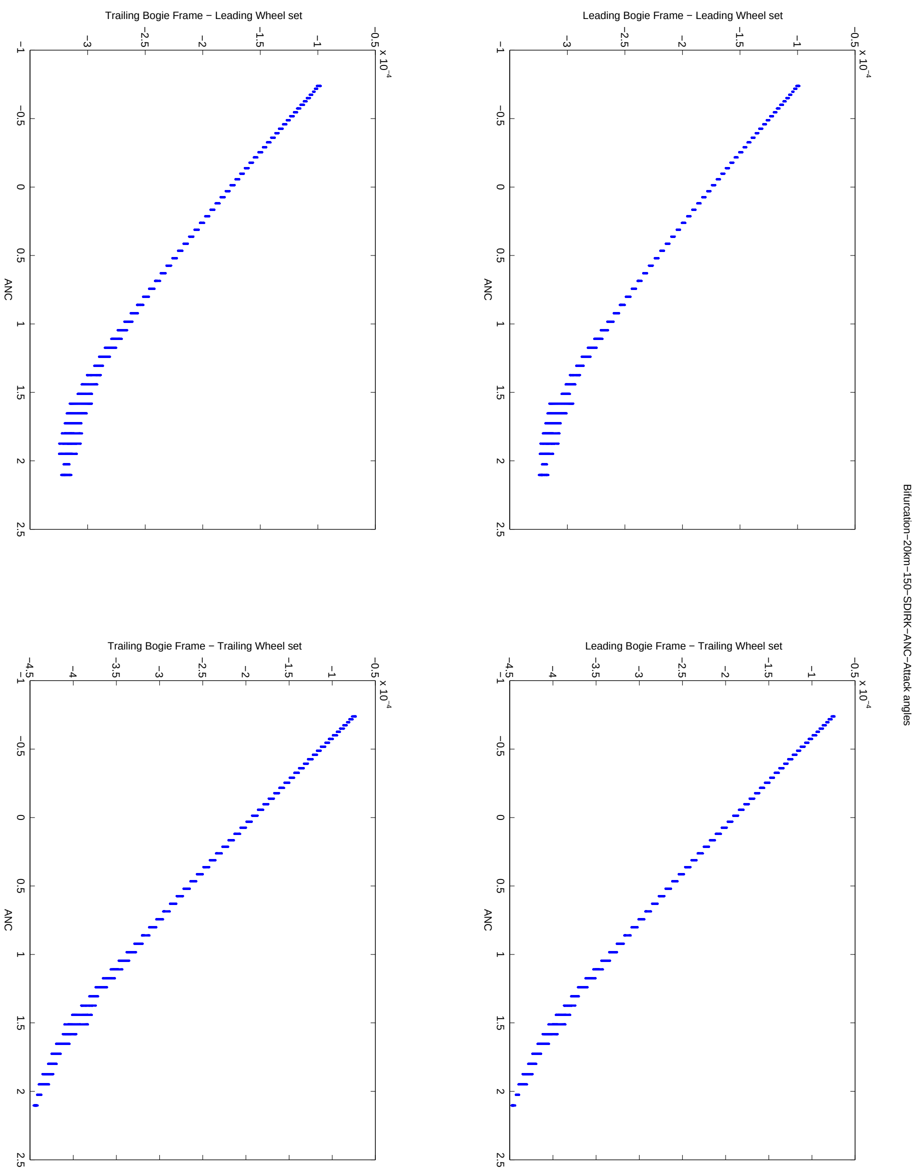


Figure F.154: AGV model: evolution of the attack angles of all the wheel sets at increasing speed on a curve of 2.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

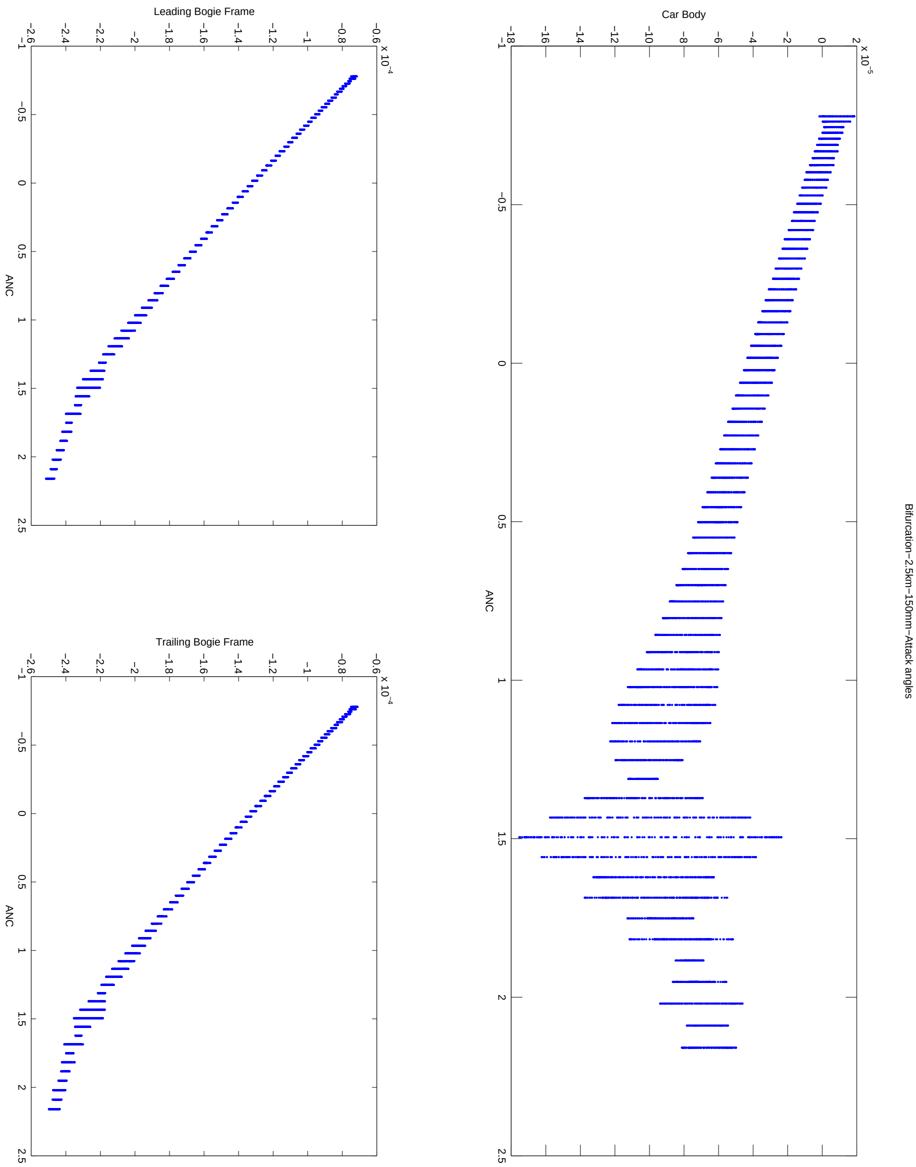


Figure F.155: AGV model: evolution of the attack angles of the car body and the bogie frames at increasing speed on a curve of 2.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

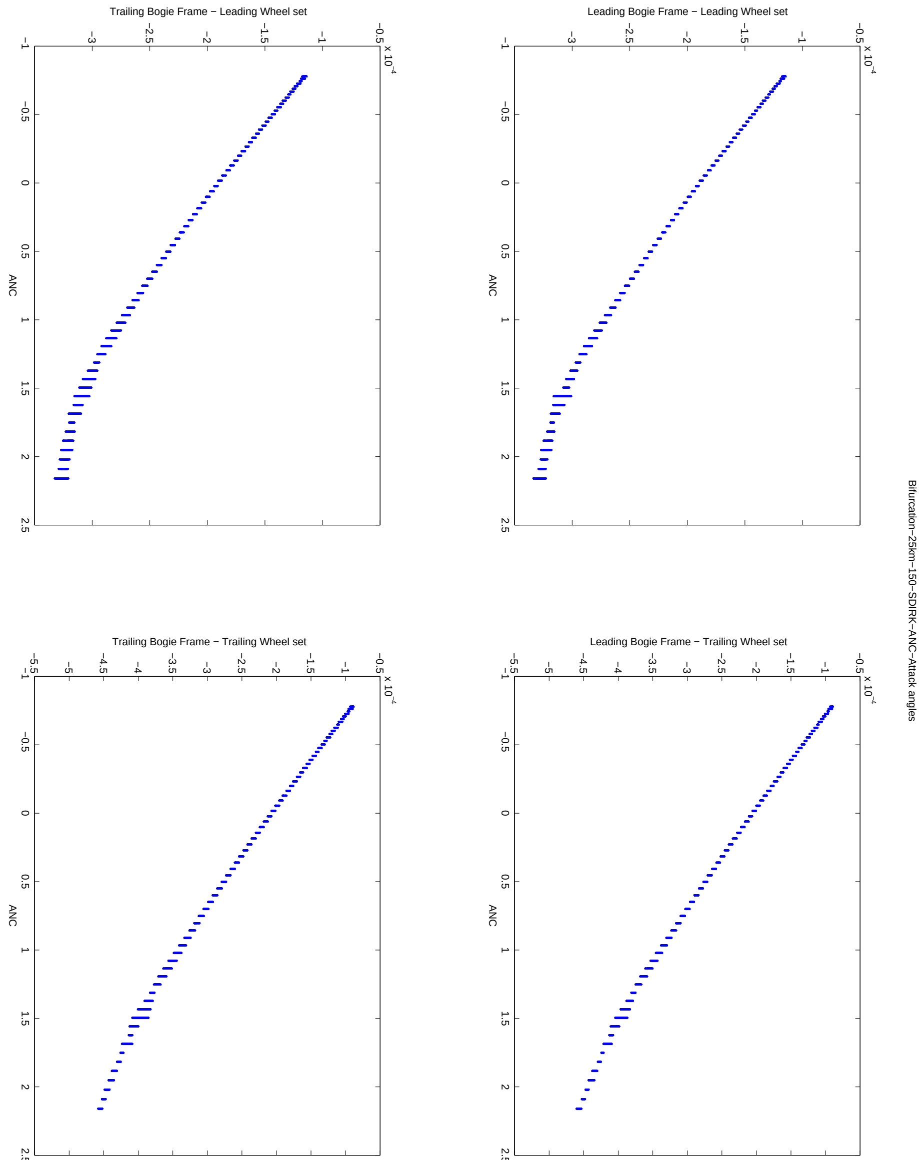


Figure F.156: AGV model: evolution of the attack angles of all the wheel sets at increasing speed on a curve of 2.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

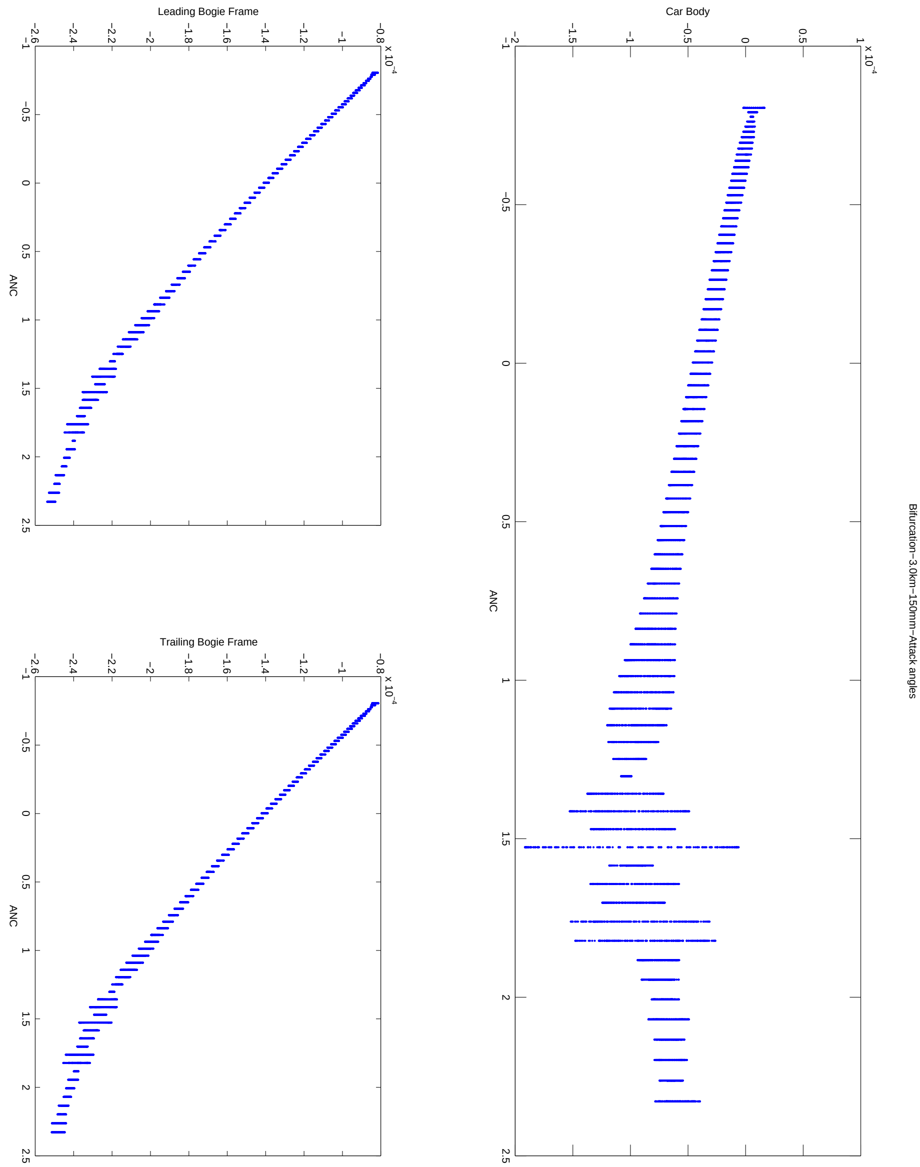


Figure F.157: AGV model: evolution of the attack angles of the car body and the bogie frames at increasing speed on a curve of 3.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

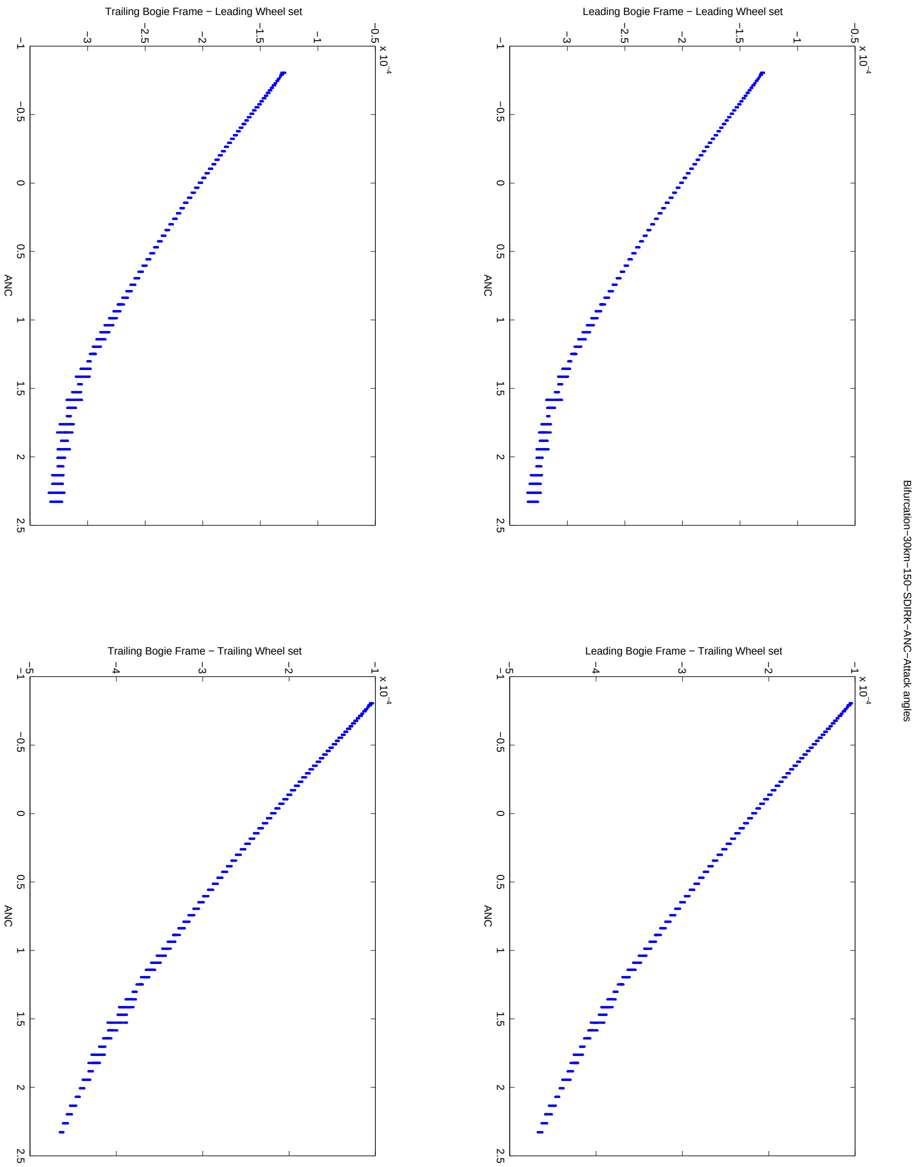


Figure F.158: AGV model: evolution of the attack angles of all the wheel sets at increasing speed on a curve of 3.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

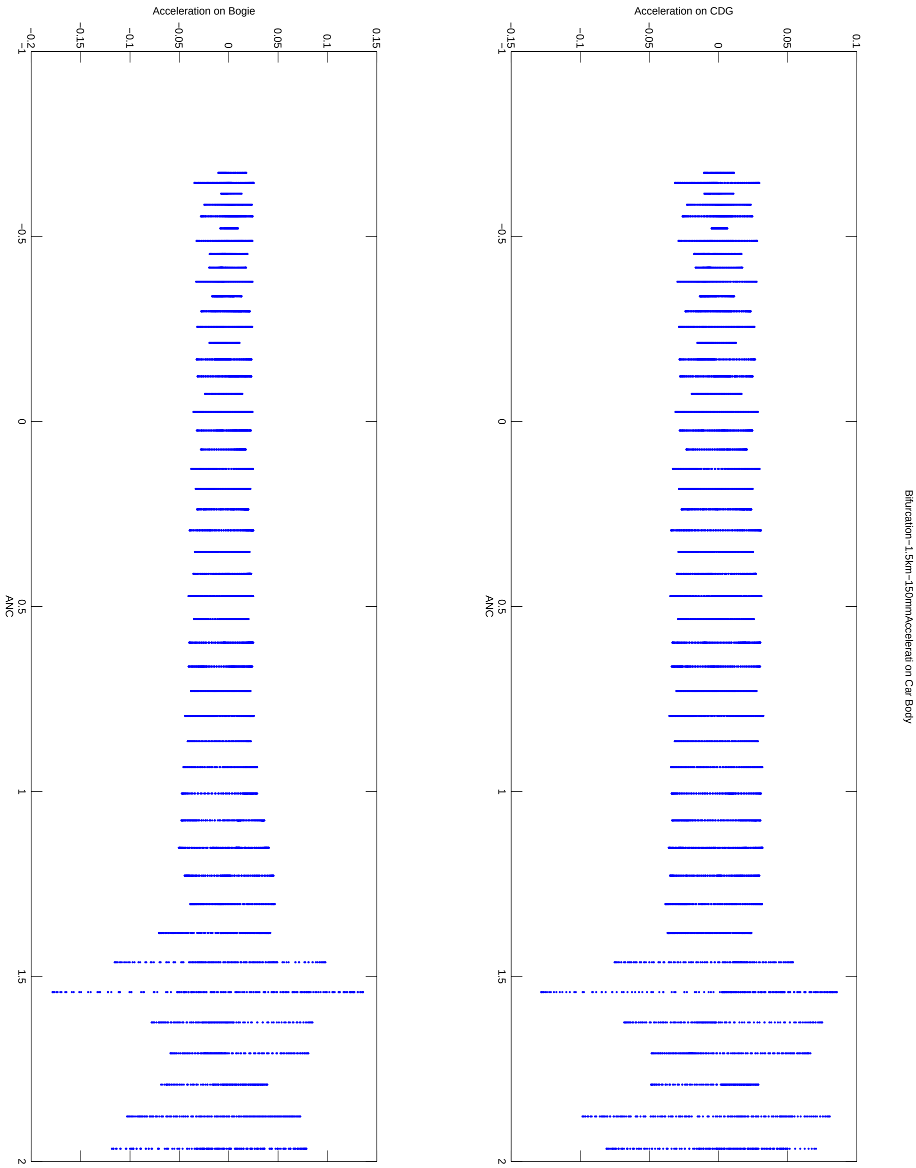


Figure F.159: AGV model: acceleration on the car body and over the position of the bogie frame at increasing speed on a curve of 1.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

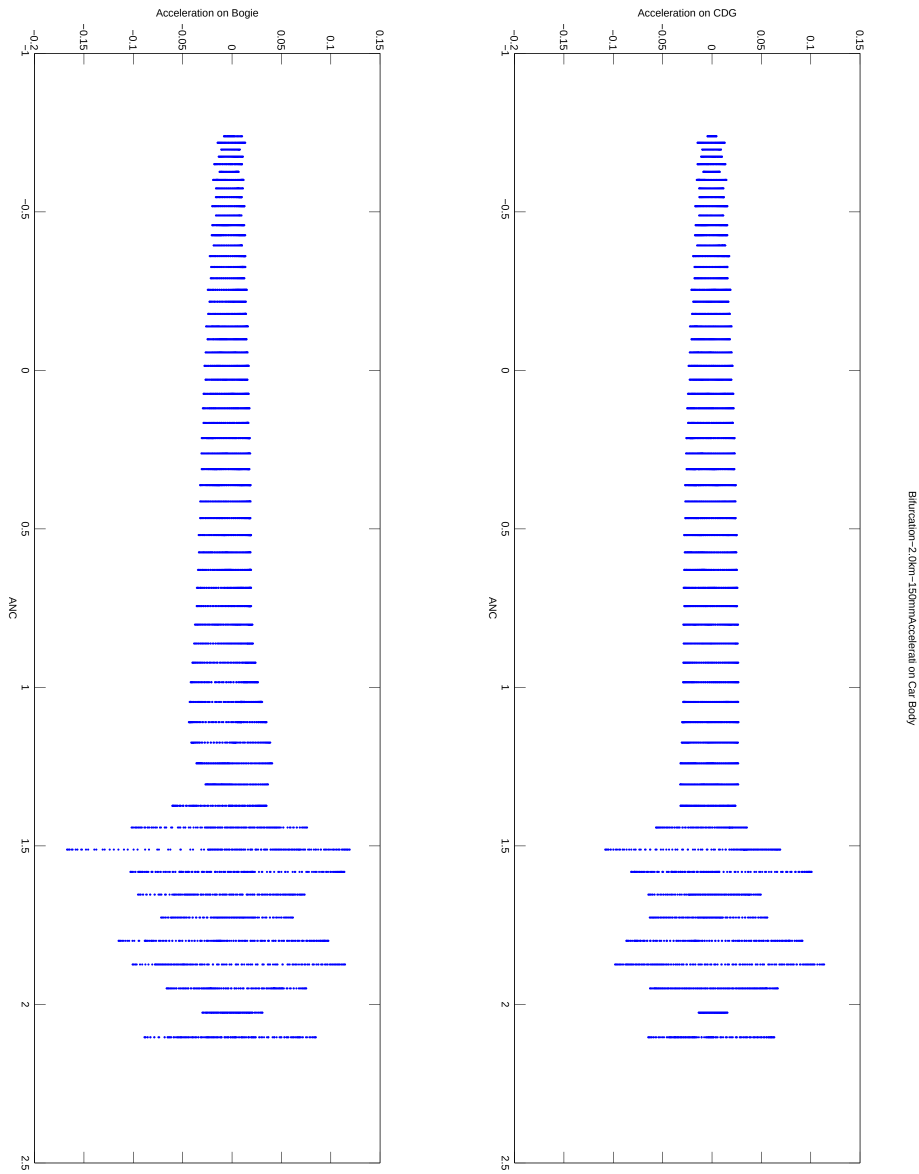


Figure F.160: AGV model: acceleration on the car body and over the position of the bogie frame at increasing speed on a curve of 2.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

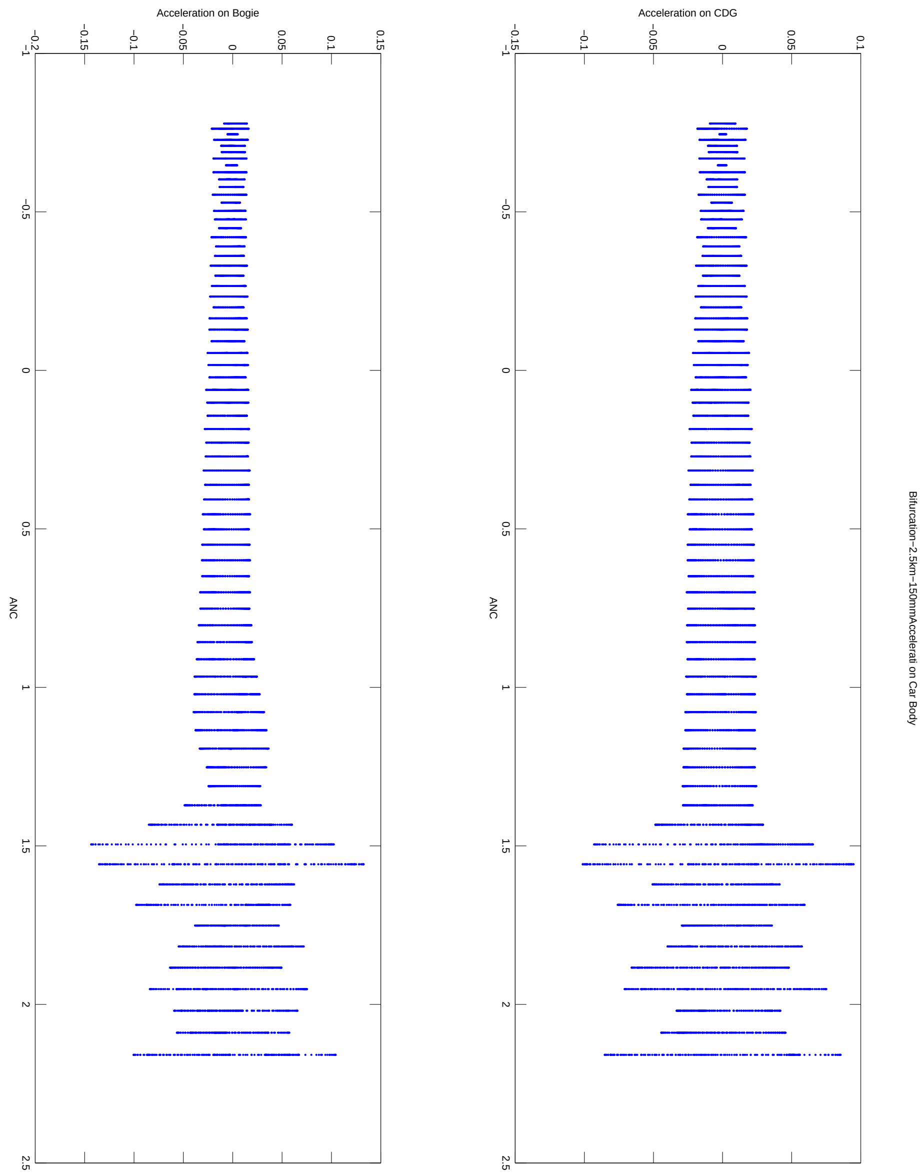


Figure F.161: AGV model: acceleration on the car body and over the position of the bogie frame at increasing speed on a curve of 2.5km with 150mm of cant. The values are plotted against the uncompensated acceleration.

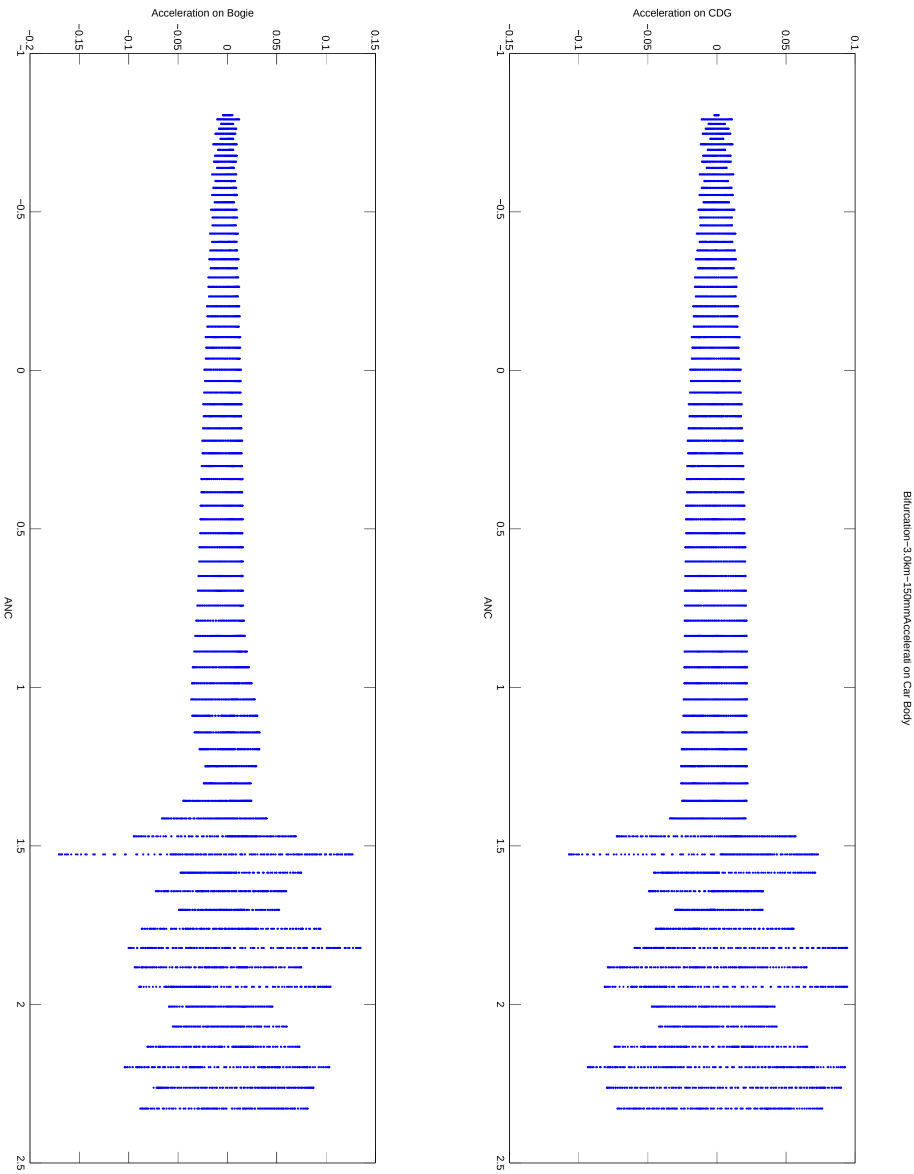


Figure F.162: AGV model: acceleration on the car body and over the position of the bogie frame at increasing speed on a curve of 3.0km with 150mm of cant. The values are plotted against the uncompensated acceleration.

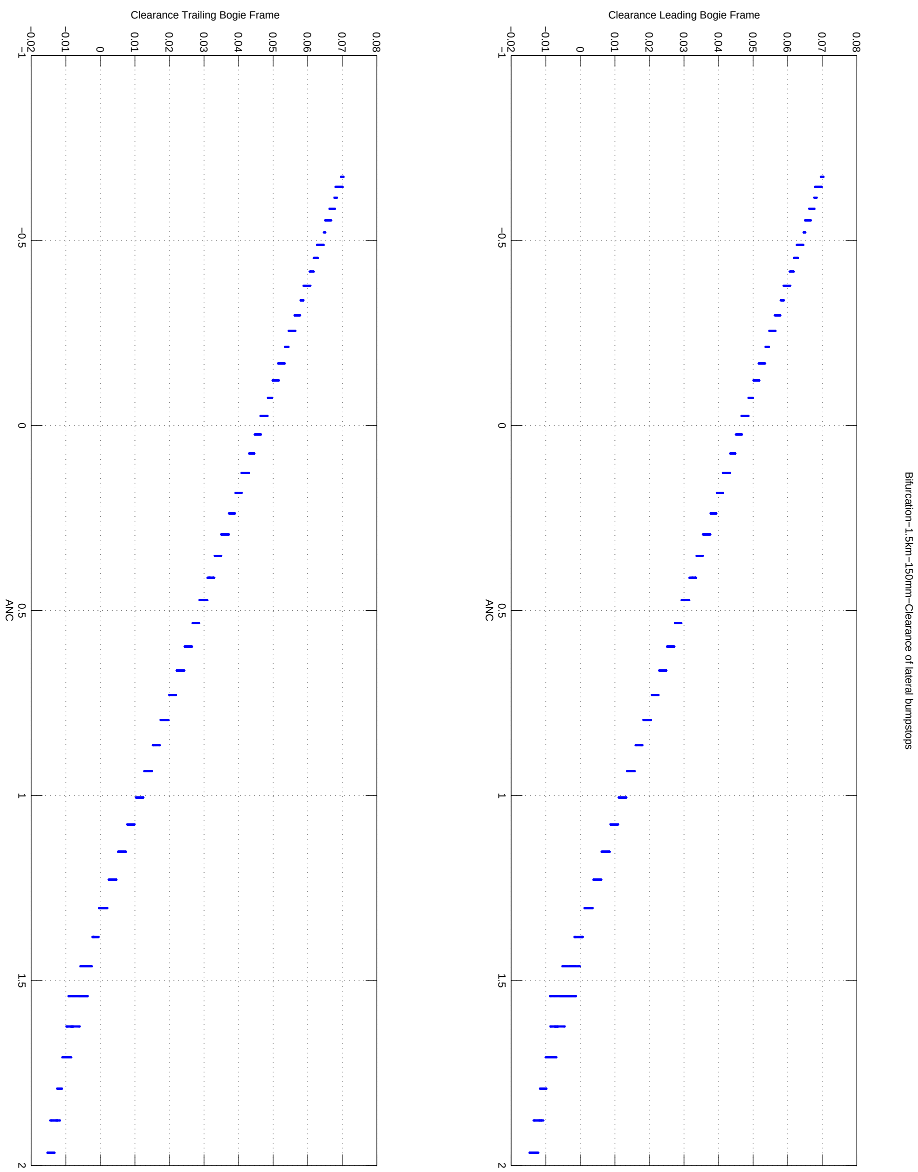


Figure F.163: AGV model: clearance of the lateral bumpstop in the secondary suspension against the uncompensated acceleration at increasing speed on a curve of 1.5km with 150mm of cant.

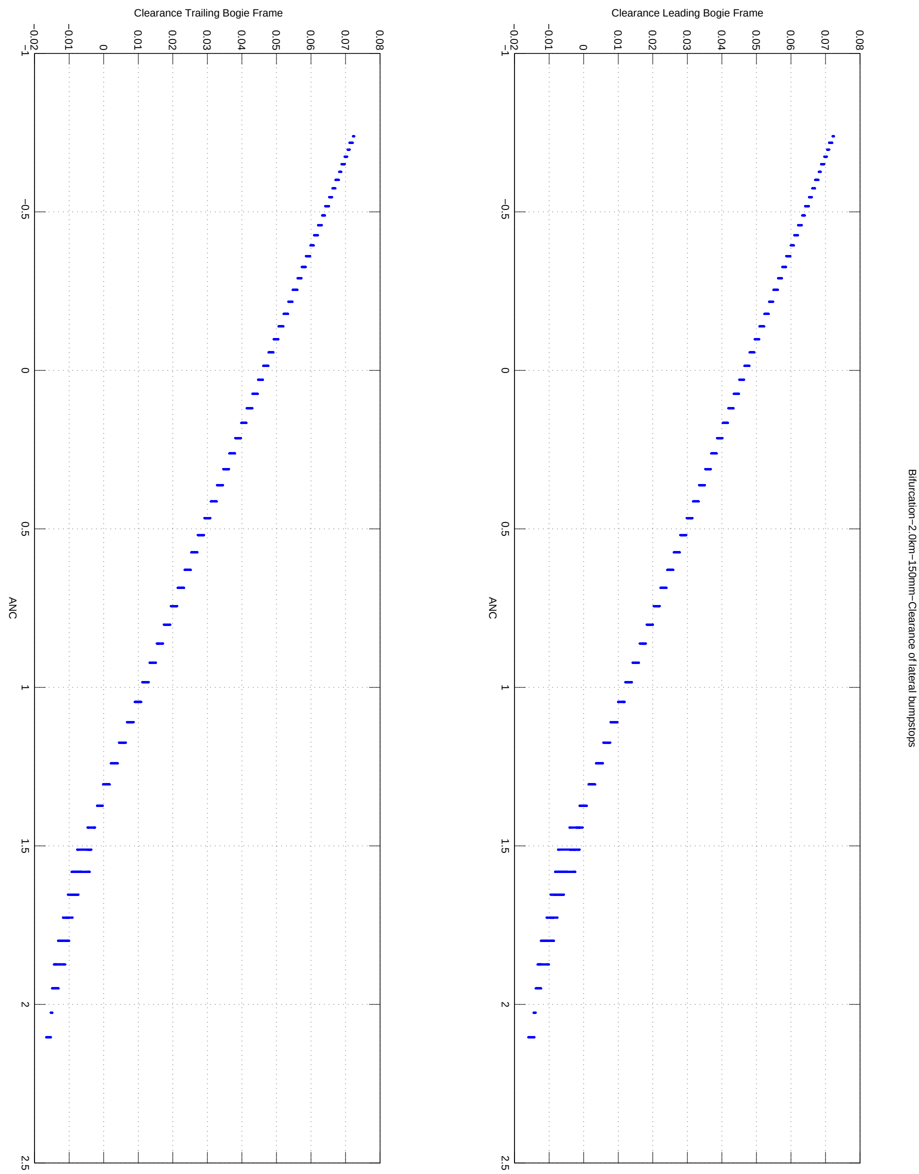


Figure F.164: AGV model: clearance of the lateral bumpstop in the secondary suspension against the uncompensated acceleration at increasing speed on a curve of 2.0km with 150mm of cant.

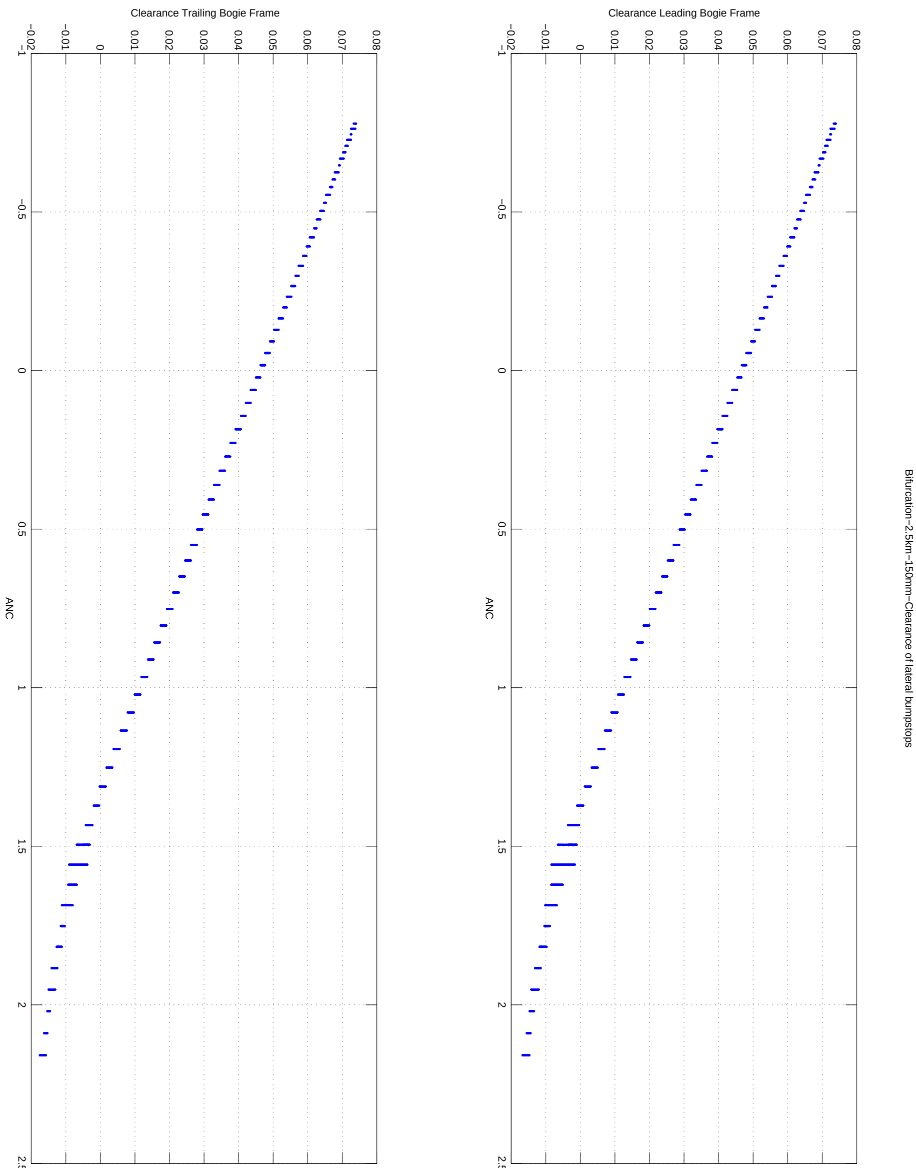


Figure F.165: AGV model: clearance of the lateral bumpstop in the secondary suspension against the uncompensated acceleration at increasing speed on a curve of 2.5km with 150mm of cant.

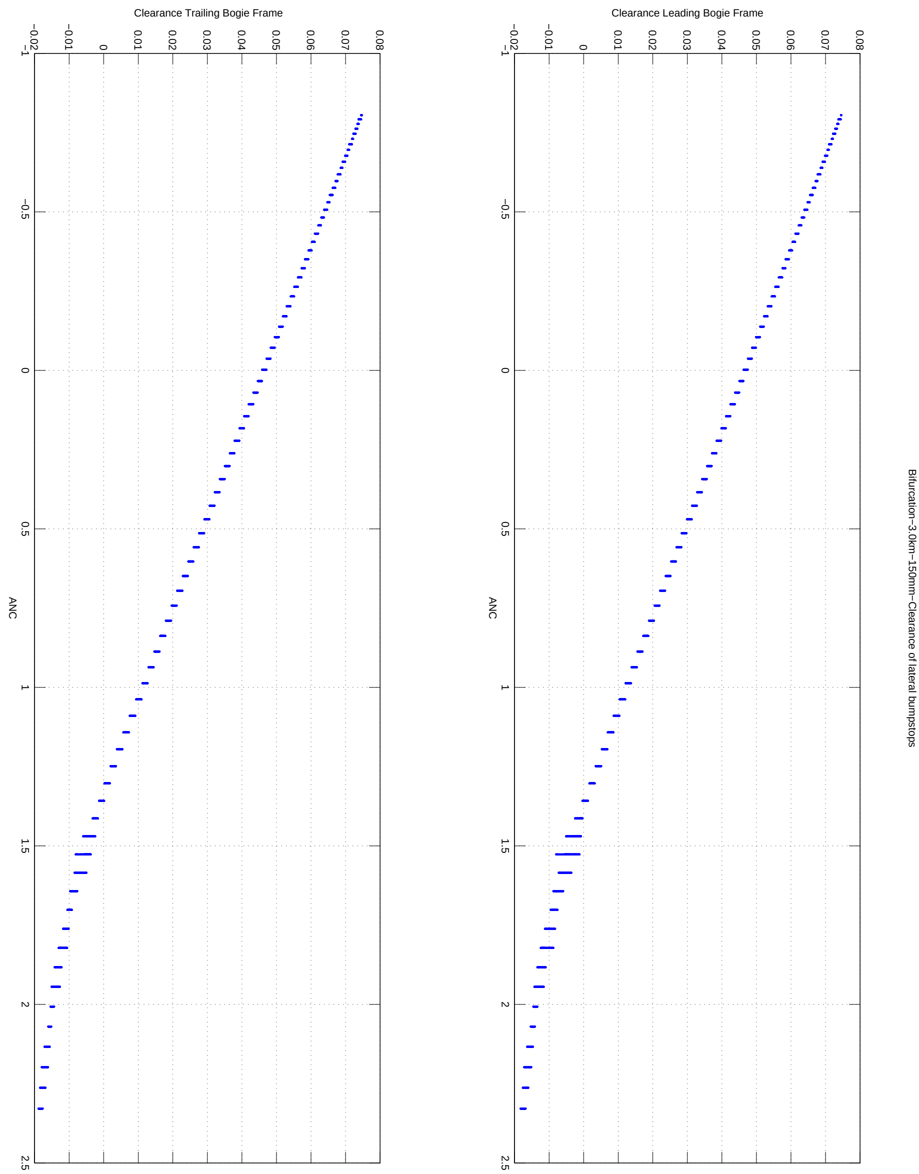


Figure F.166: AGV model: clearance of the lateral bumpstop in the secondary suspension against the uncompensated acceleration at increasing speed on a curve of 3.0km with 150mm of cant.

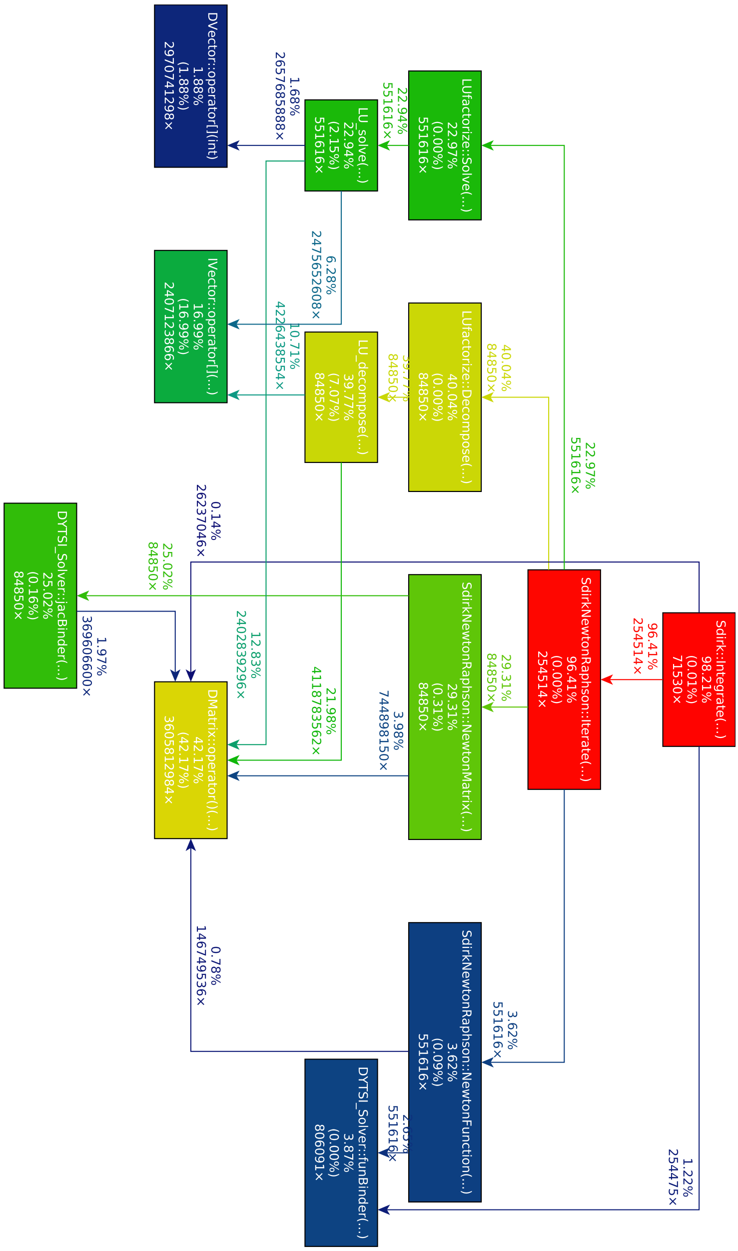


Figure F.167: Profiling: call diagram of DYTSI when an SDIRK method is used for performing a transient analysis of a train running on a transient curve for 20s and on constant curve for other 20s. The continuation of the diagram is in figure F.168 and the functions `funBinder` and `jacBinder` link the two parts.

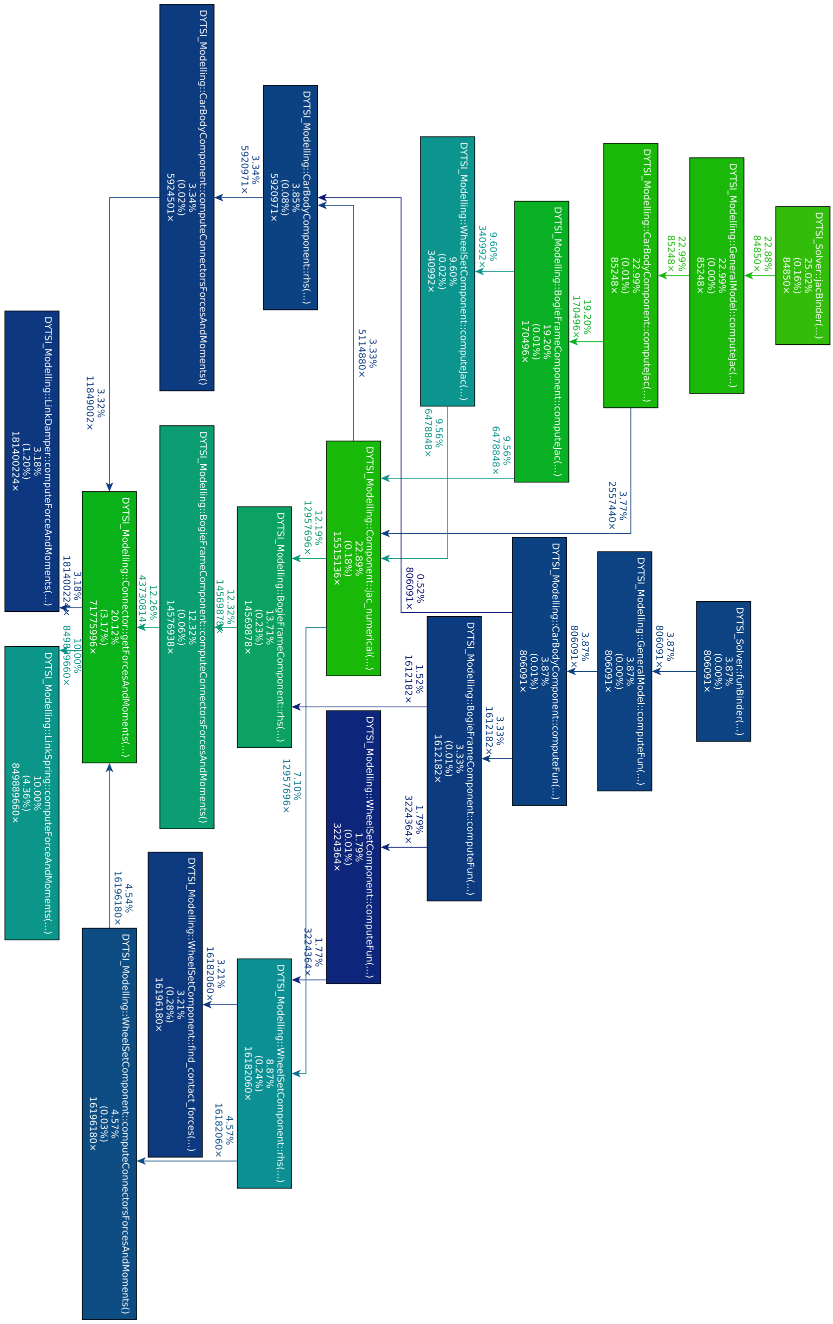


Figure F.168: Profiling: second part of the call diagram (see fig.F.167).

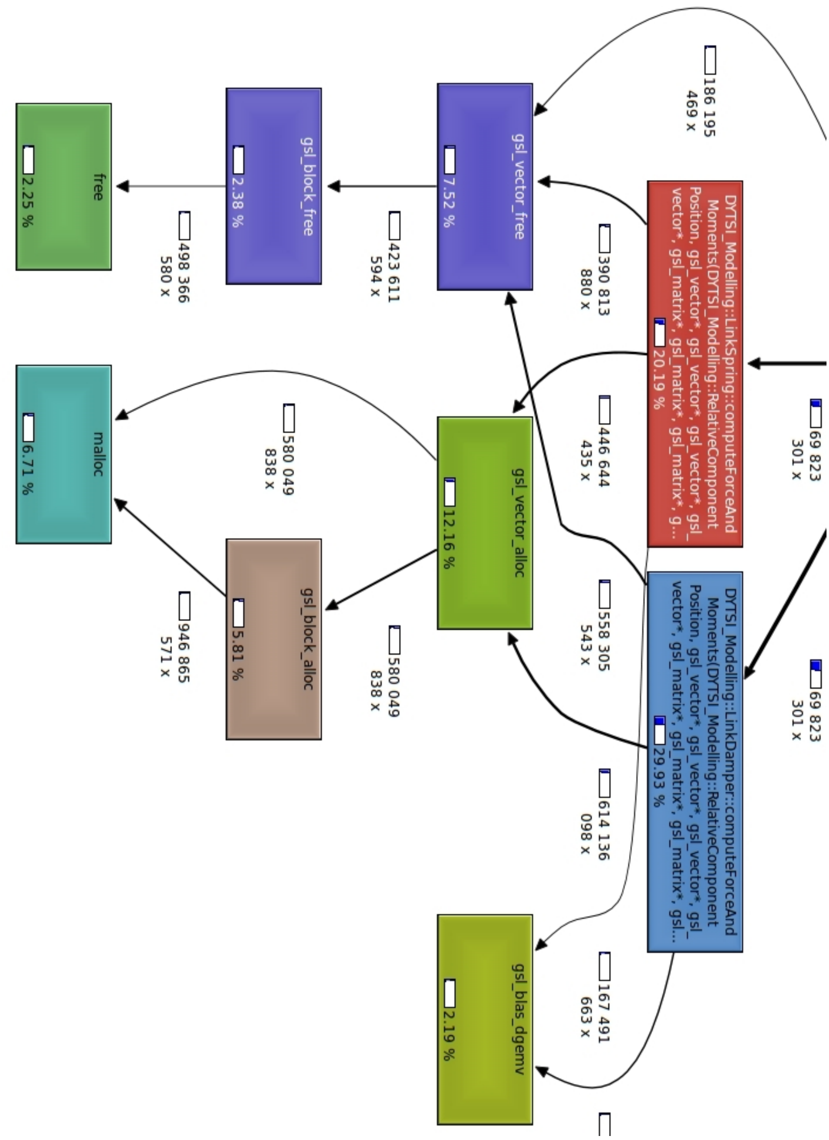


Figure F.169: Profiling: particular of the callgrind call diagram about the memory allocation for the computation of the forces and moments.

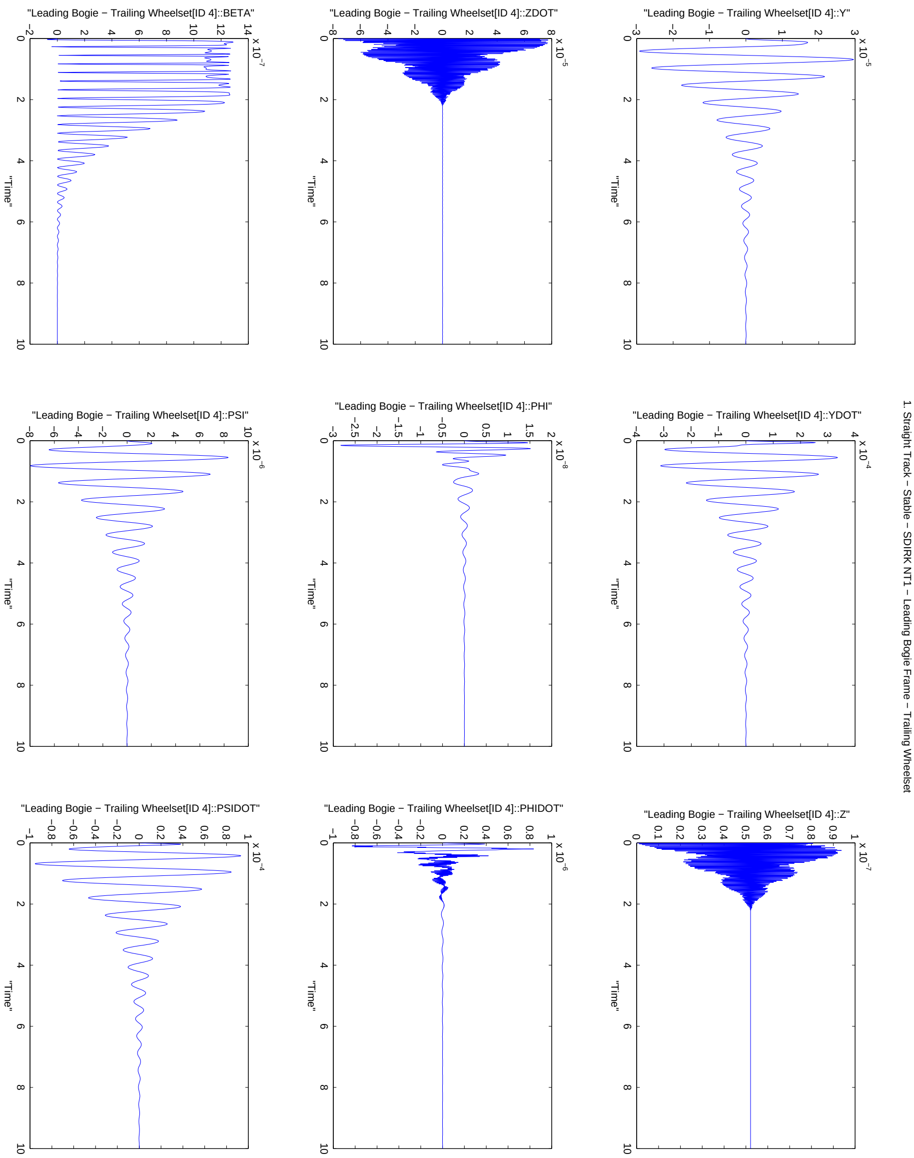


Figure F.170: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK NT1 solver on the test case 1 (transient on straight track with speed 40m/s - not hunting).

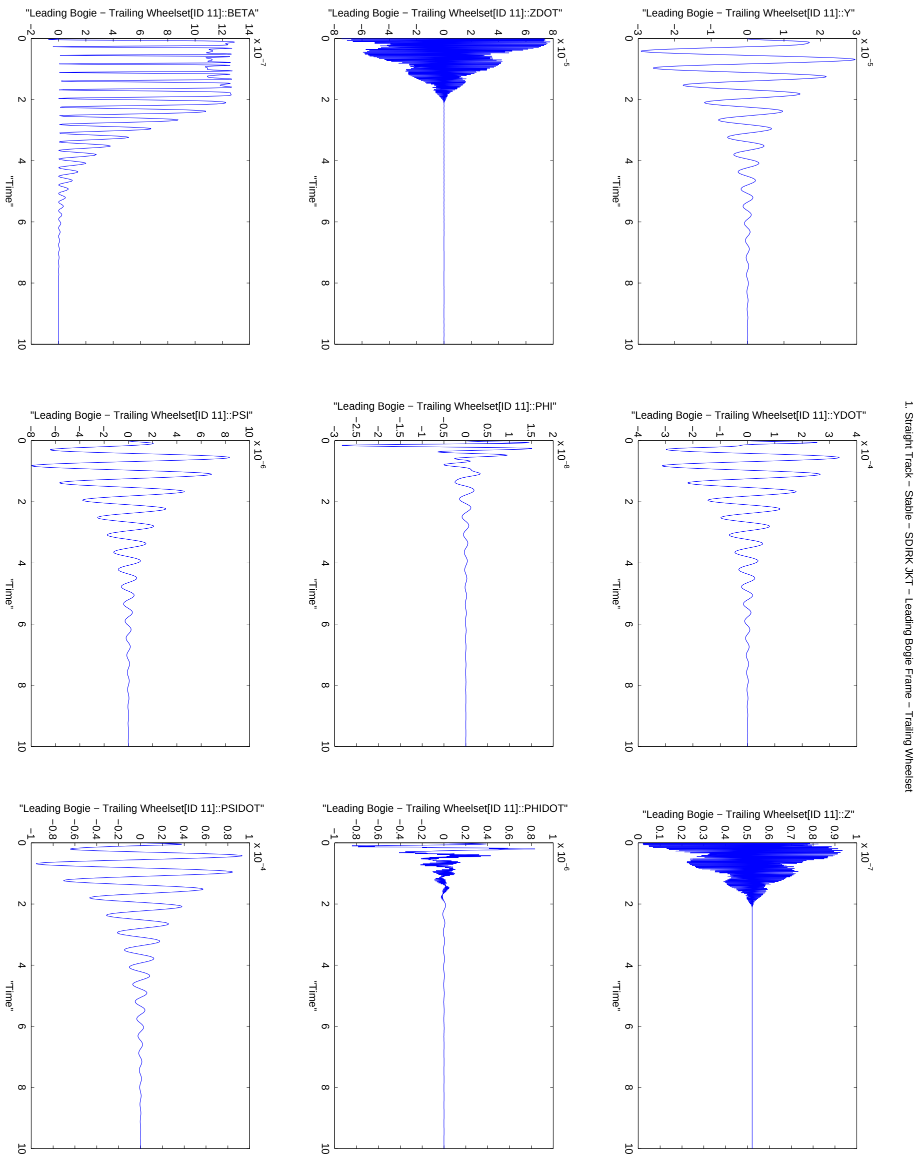


Figure F.171: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK JKT solver on the test case 1 (transient on straight track with speed 40m/s - not hunting).

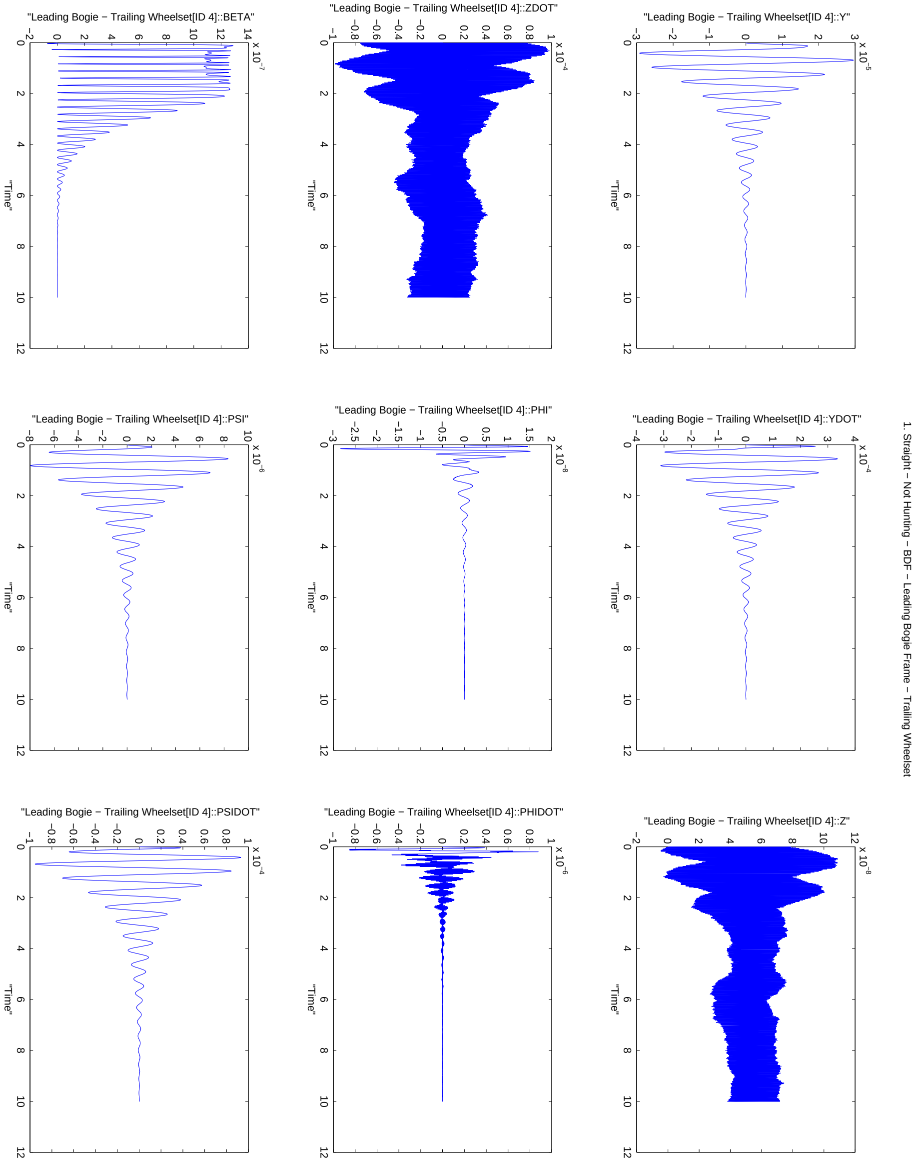


Figure F.172: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the BDF solver on the test case 1 (transient on straight track with speed 40m/s - not hunting).

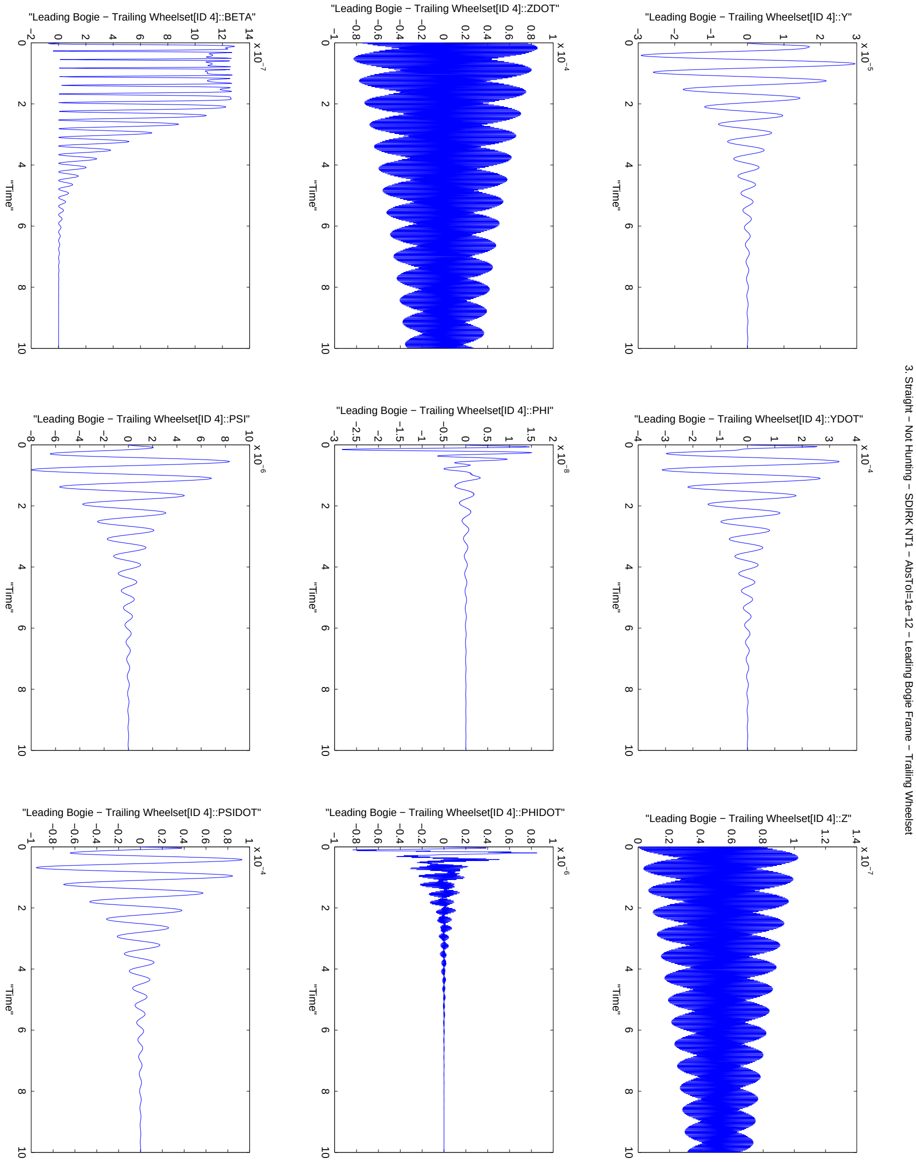


Figure F.173: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK NT1 solver (with reduced tolerance 10^{-12}) on the test case 1 (transient on straight track with speed 40m/s - not hunting).

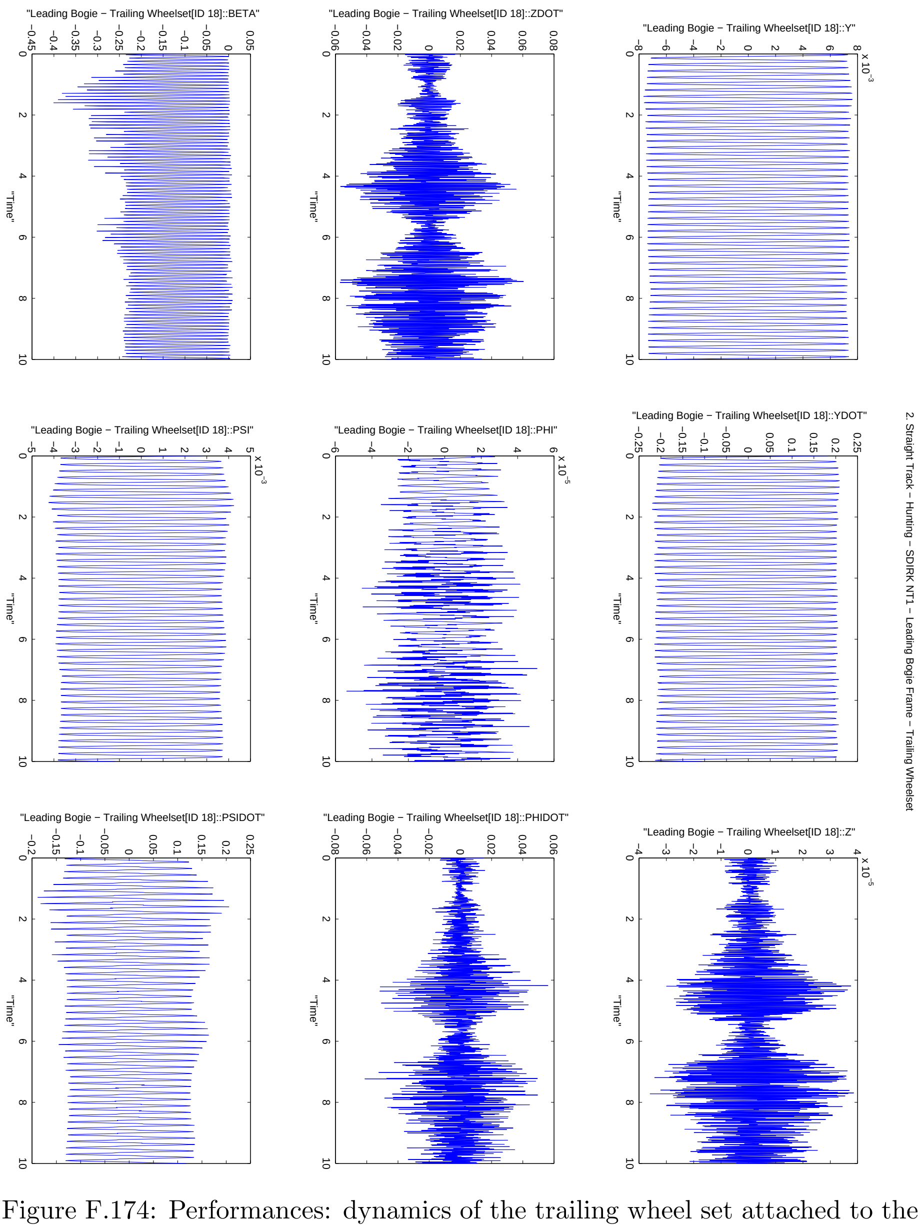


Figure F.174: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK NT1 solver on the test case 2 (transient on straight track with speed 60m/s - hunting).

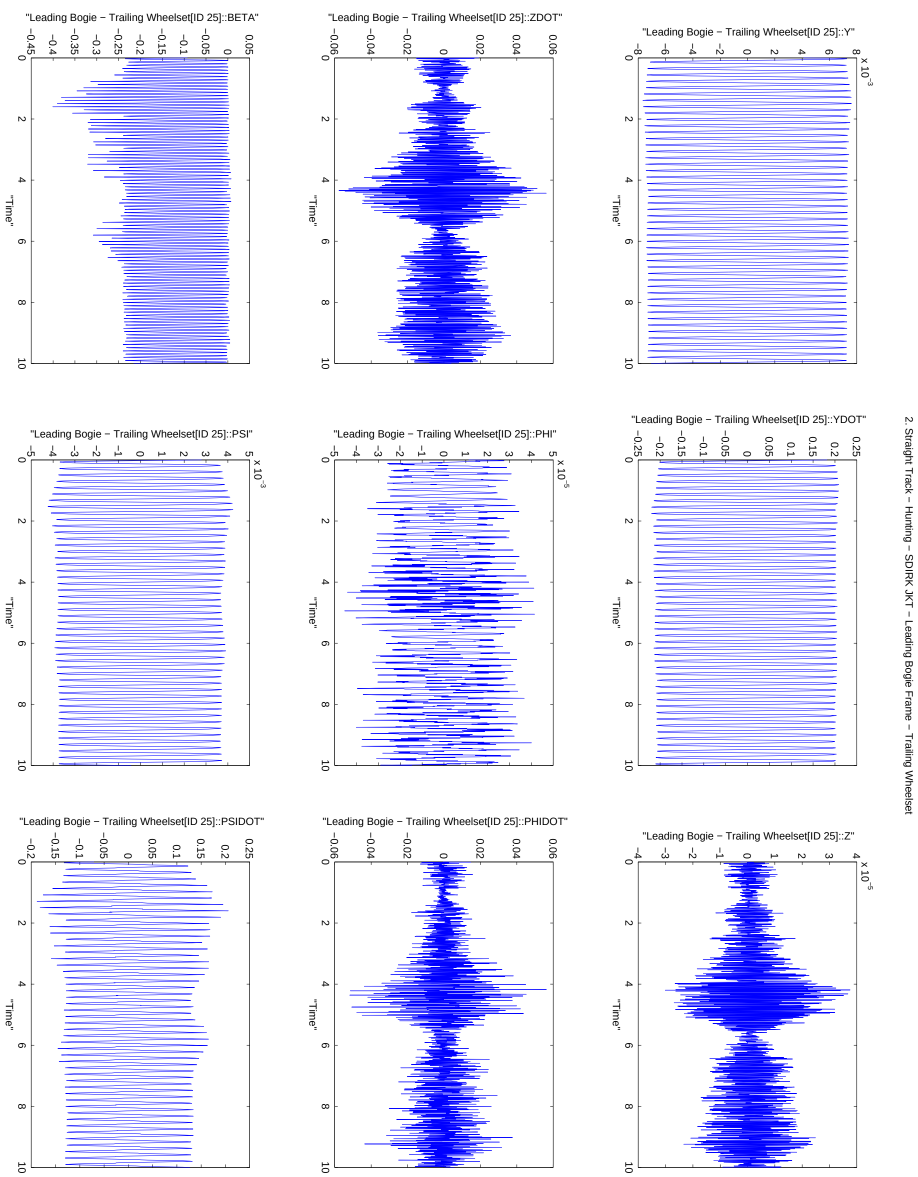


Figure F.175: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK JKT solver on the test case 2 (transient on straight track with speed 60m/s - hunting).

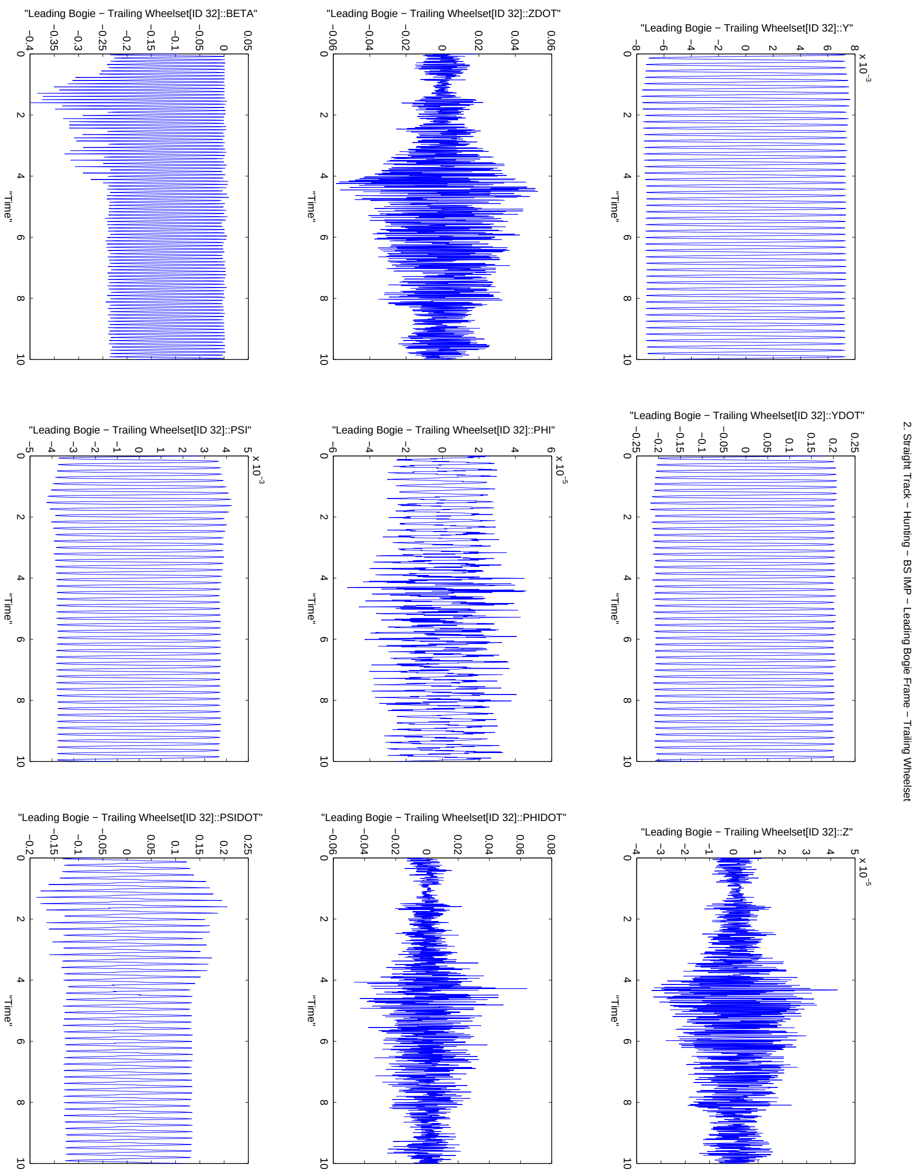


Figure F.176: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the Bulirsch-Stoer solver on the test case 2 (transient on straight track with speed 60m/s - hunting).

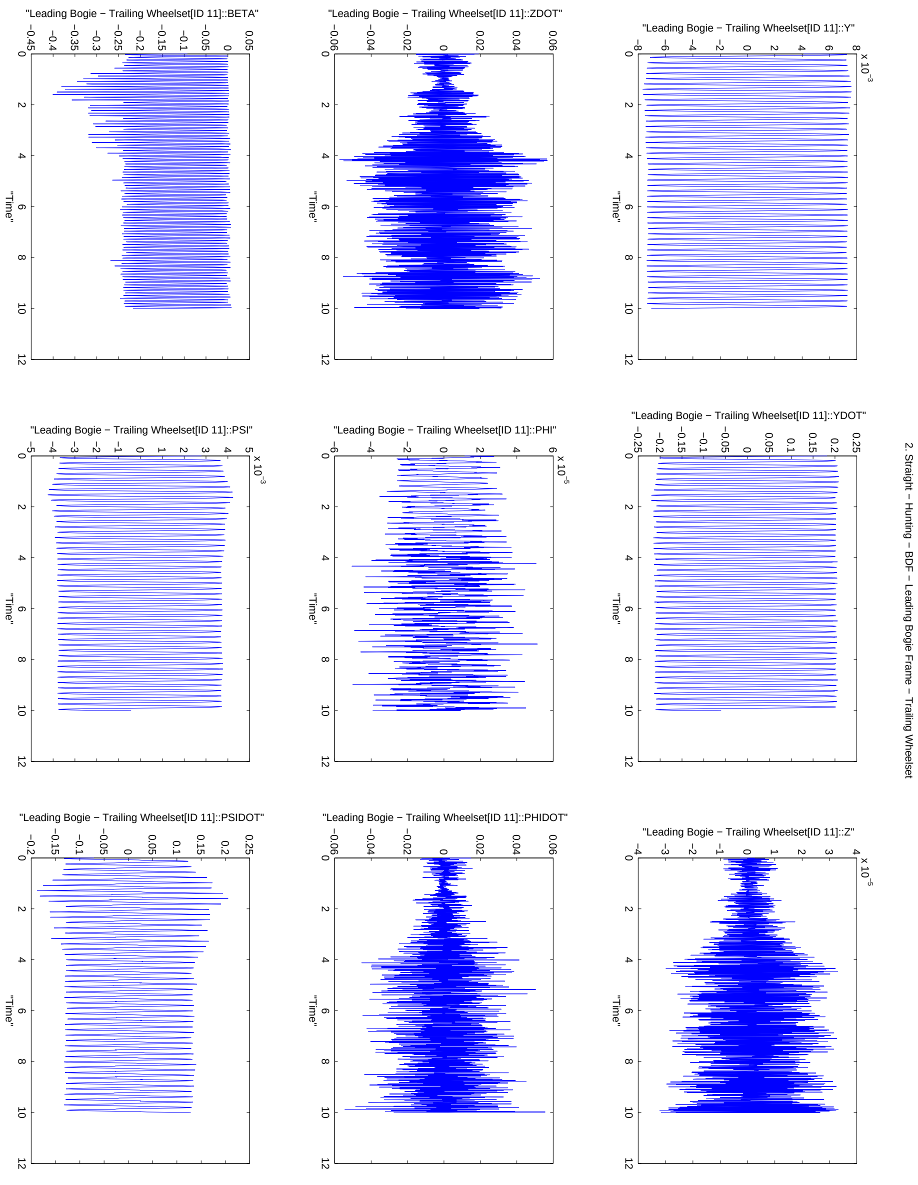


Figure F.177: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the BDF solver on the test case 2 (transient on straight track with speed 60m/s - hunting).

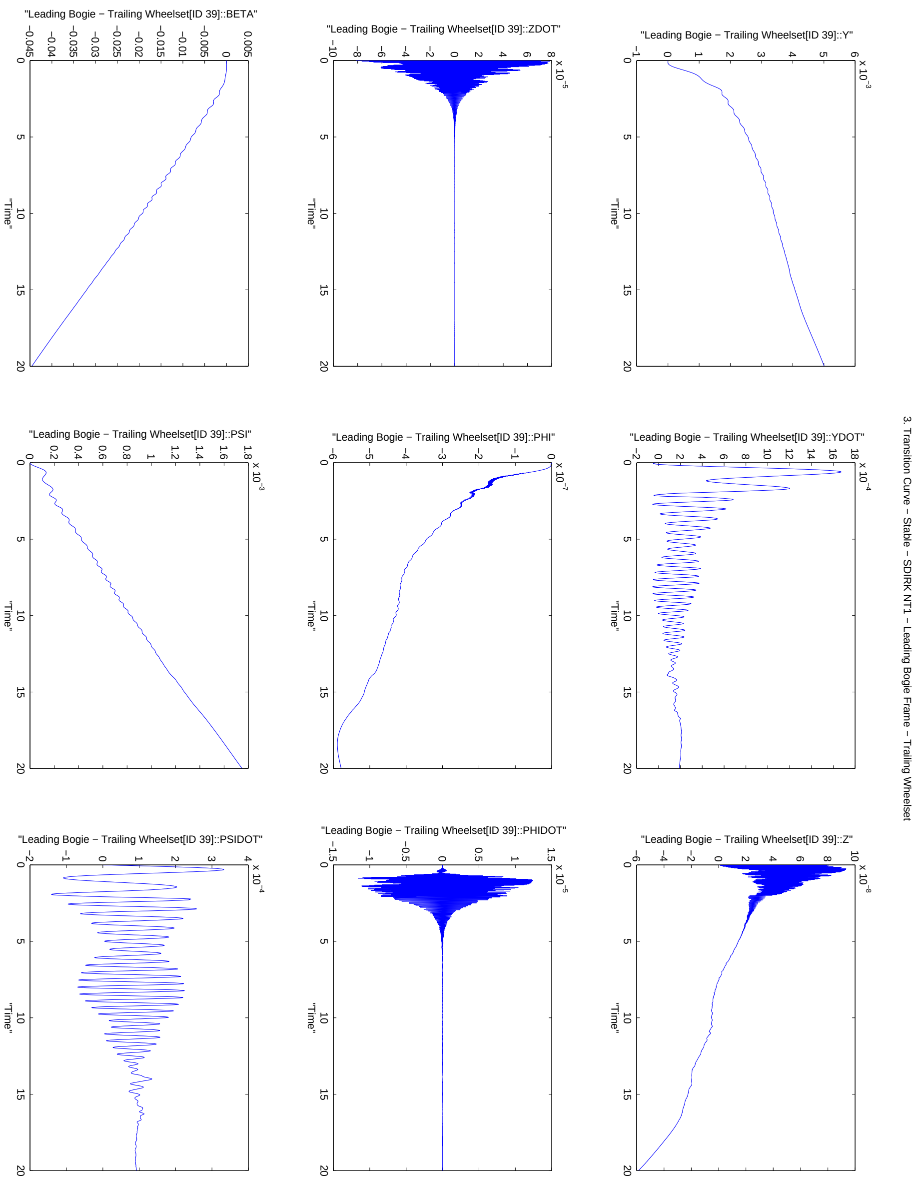


Figure F.178: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK NT1 solver on the test case 3 (transient on transition curve with speed 20m/s - not hunting).

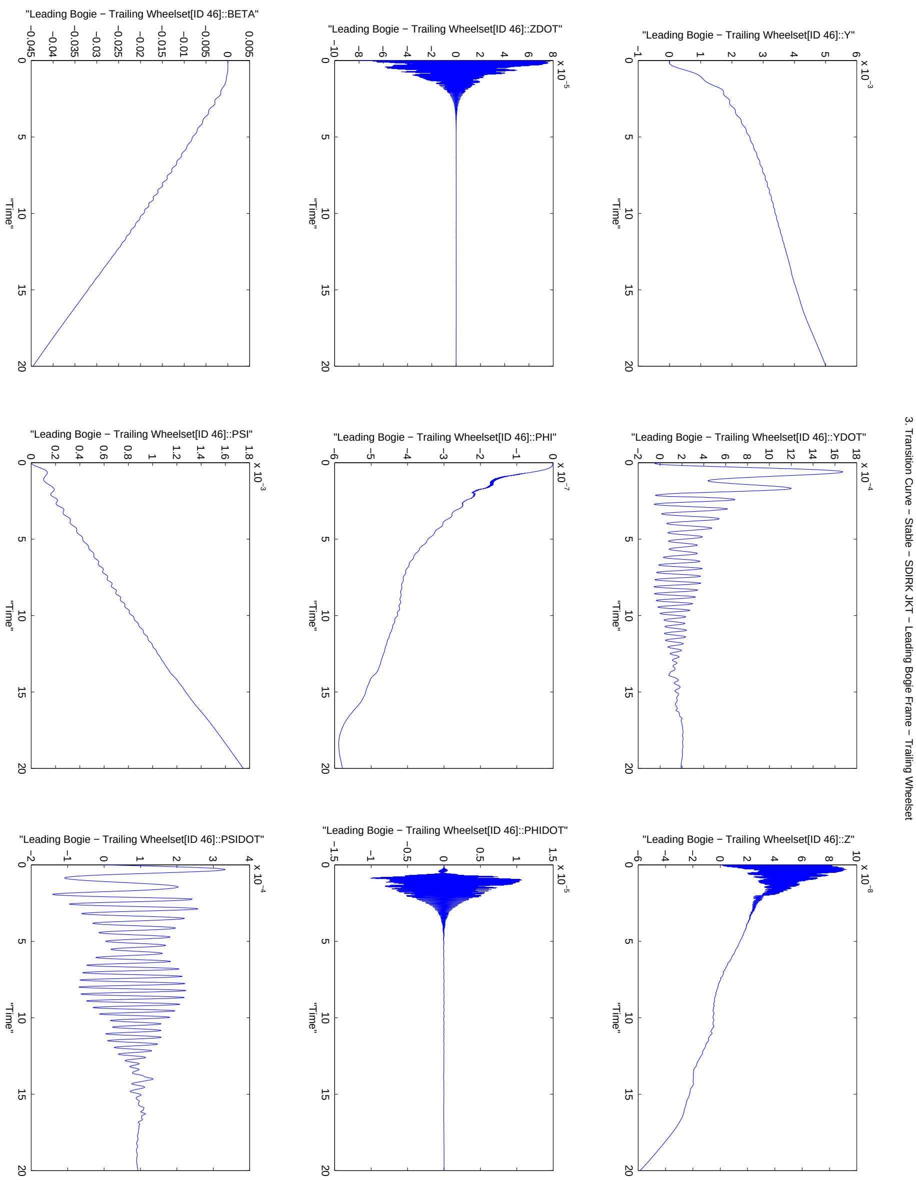


Figure F.179: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK JKT solver on the test case 3 (transient on transition curve with speed 20m/s - not hunting).

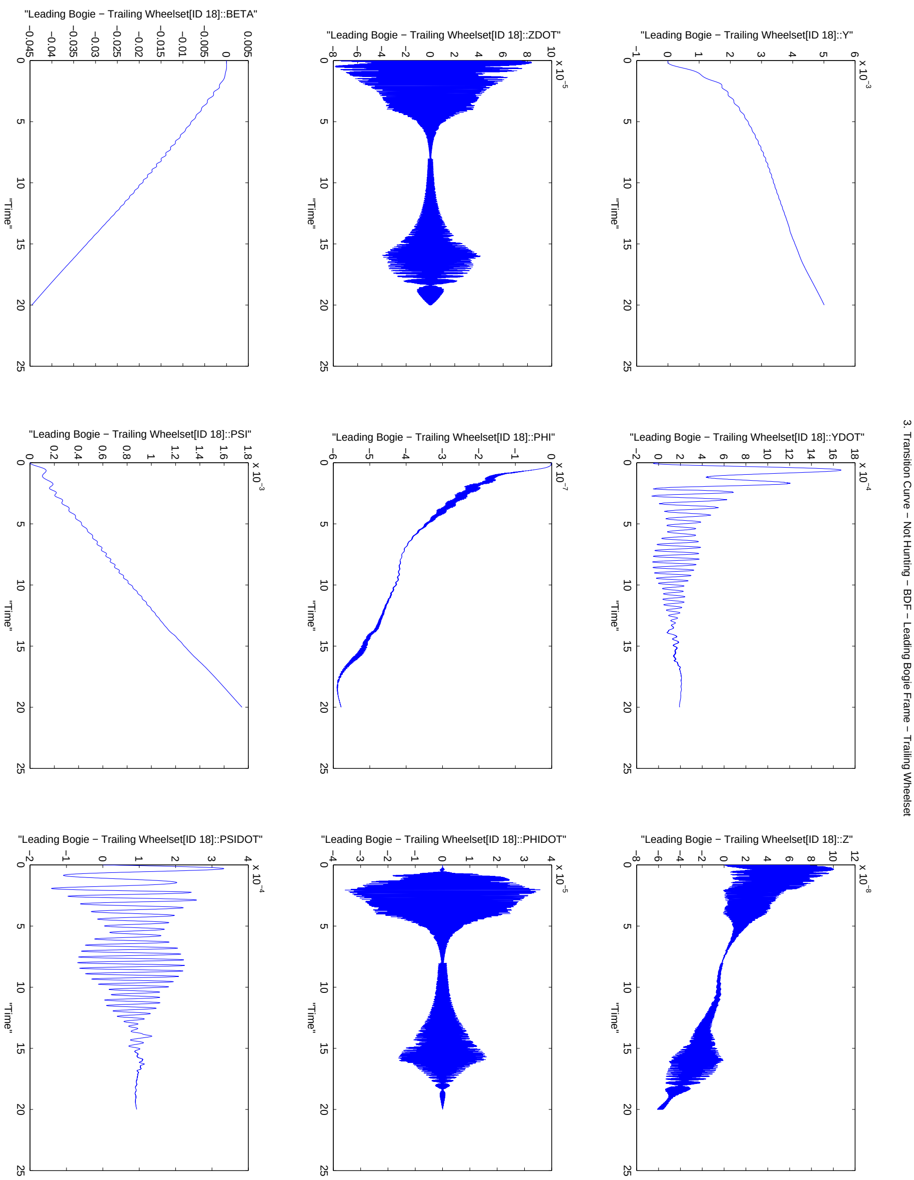


Figure F.180: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the BDF solver on the test case 3 (transient on transition curve with speed 20m/s - not hunting).

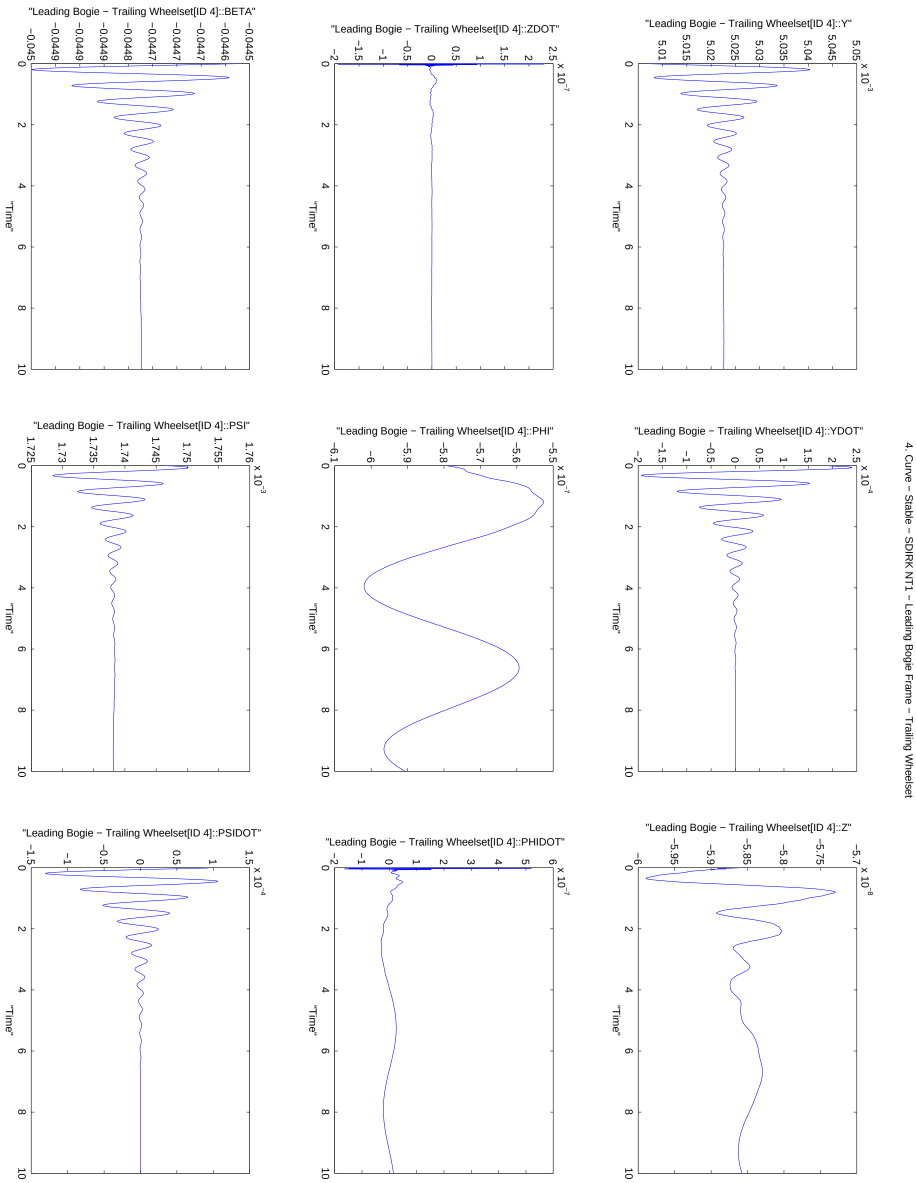


Figure F.181: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK NT1 solver on the test case 4 (transient on curved track with speed 20m/s - not hunting).

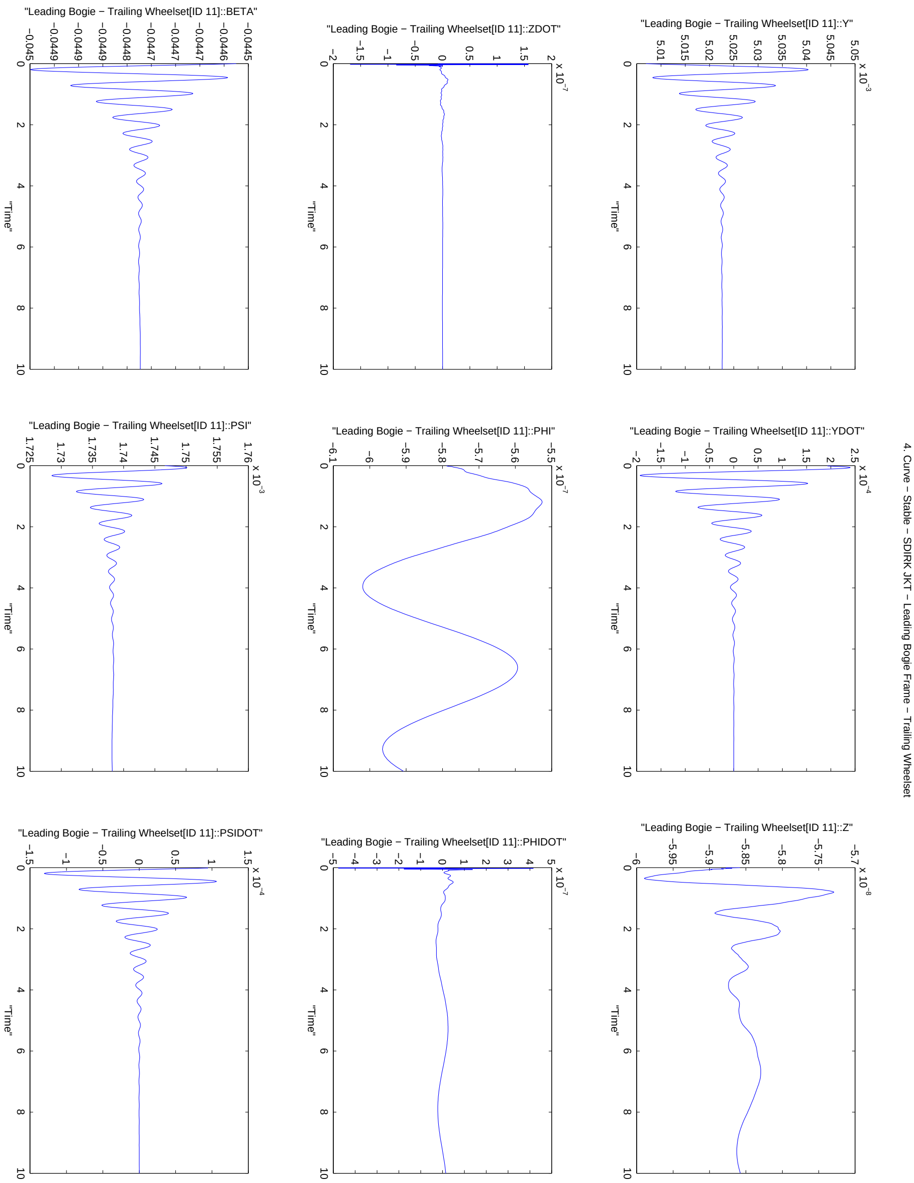


Figure F.182: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK JKT solver on the test case 4 (transient on curved track with speed 20m/s - not hunting).

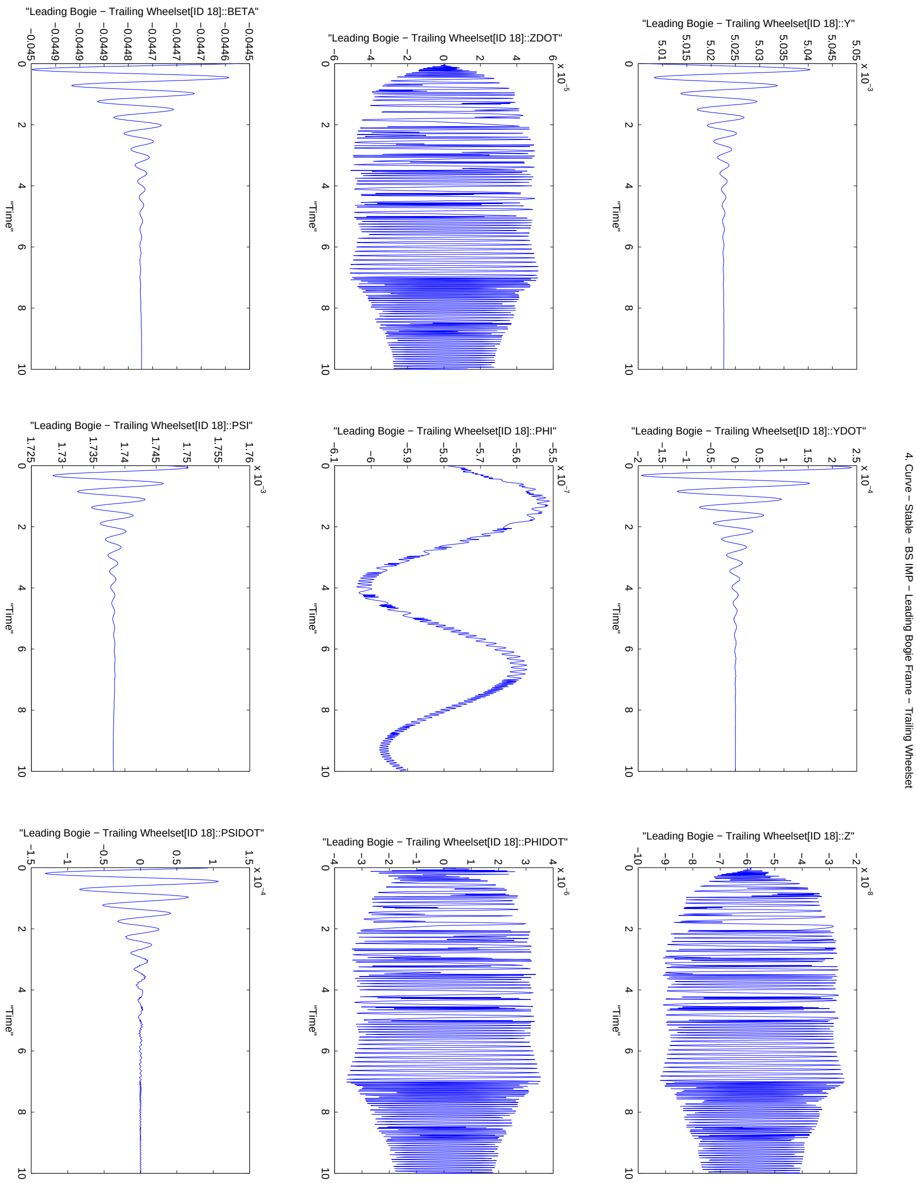


Figure F.183: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the Bulirsch-Stoer solver on the test case 4 (transient on curved track with speed 20m/s - not hunting).

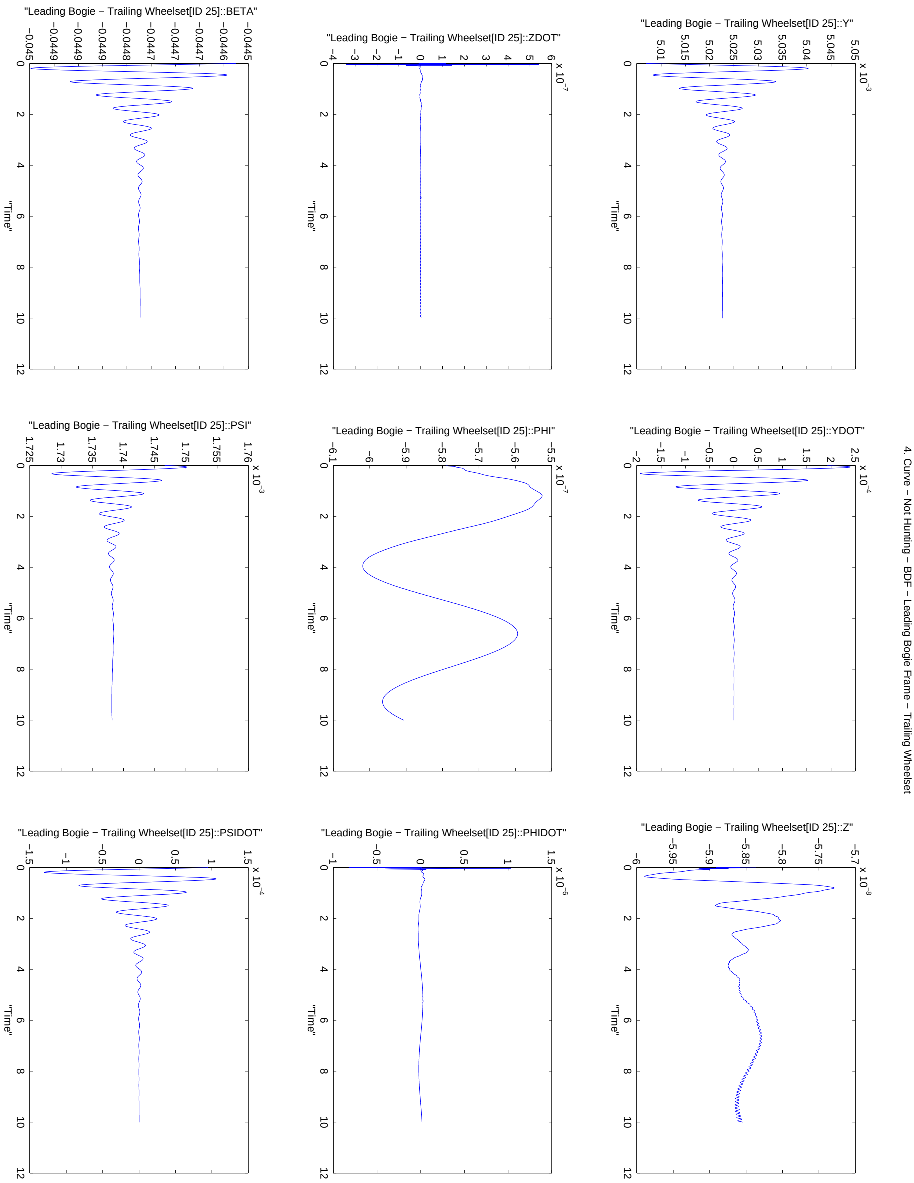


Figure F.184: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the BDF solver on the test case 4 (transient on curved track with speed 20m/s - not hunting).

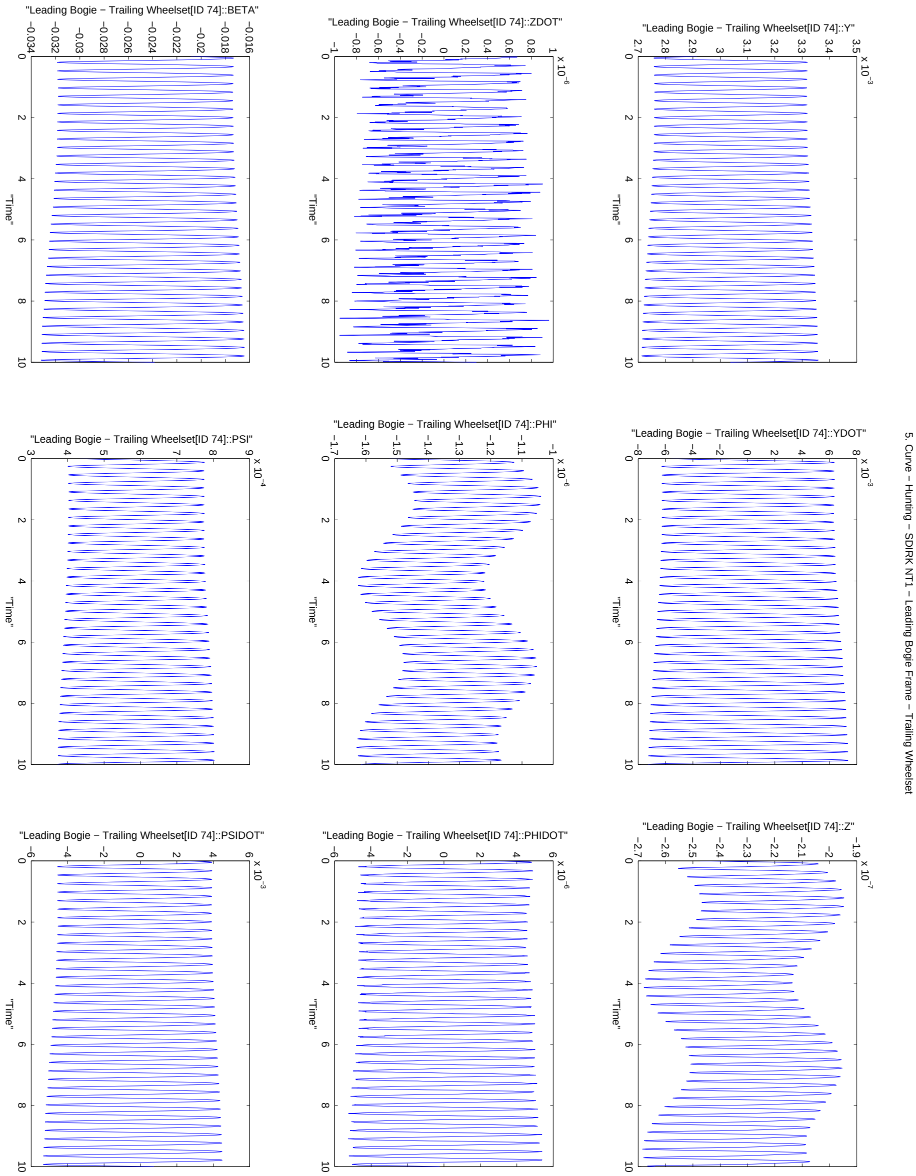


Figure F.185: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK NT1 solver on the test case 5 (transient on curved track with speed 35m/s - hunting).

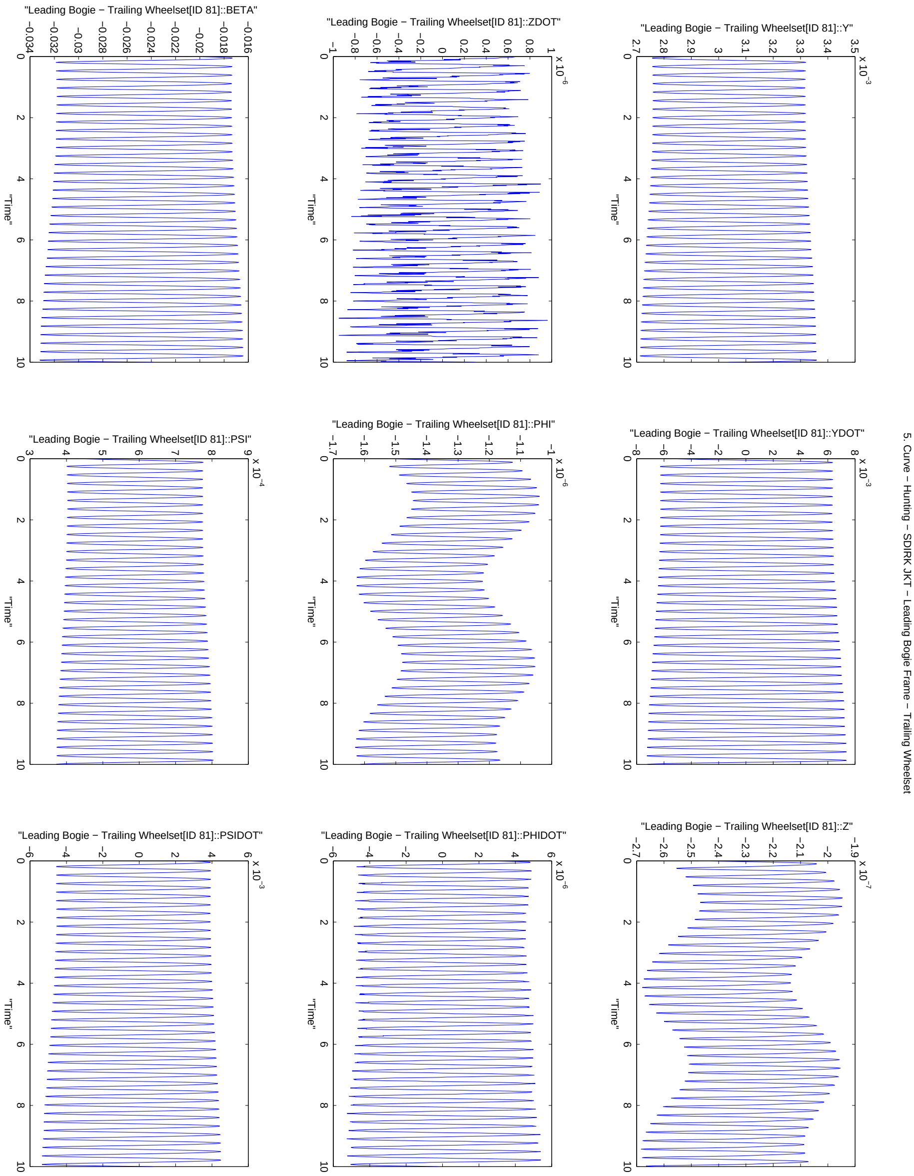


Figure F.186: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the SDIRK JKT solver on the test case 5 (transient on curved track with speed 35m/s - hunting).

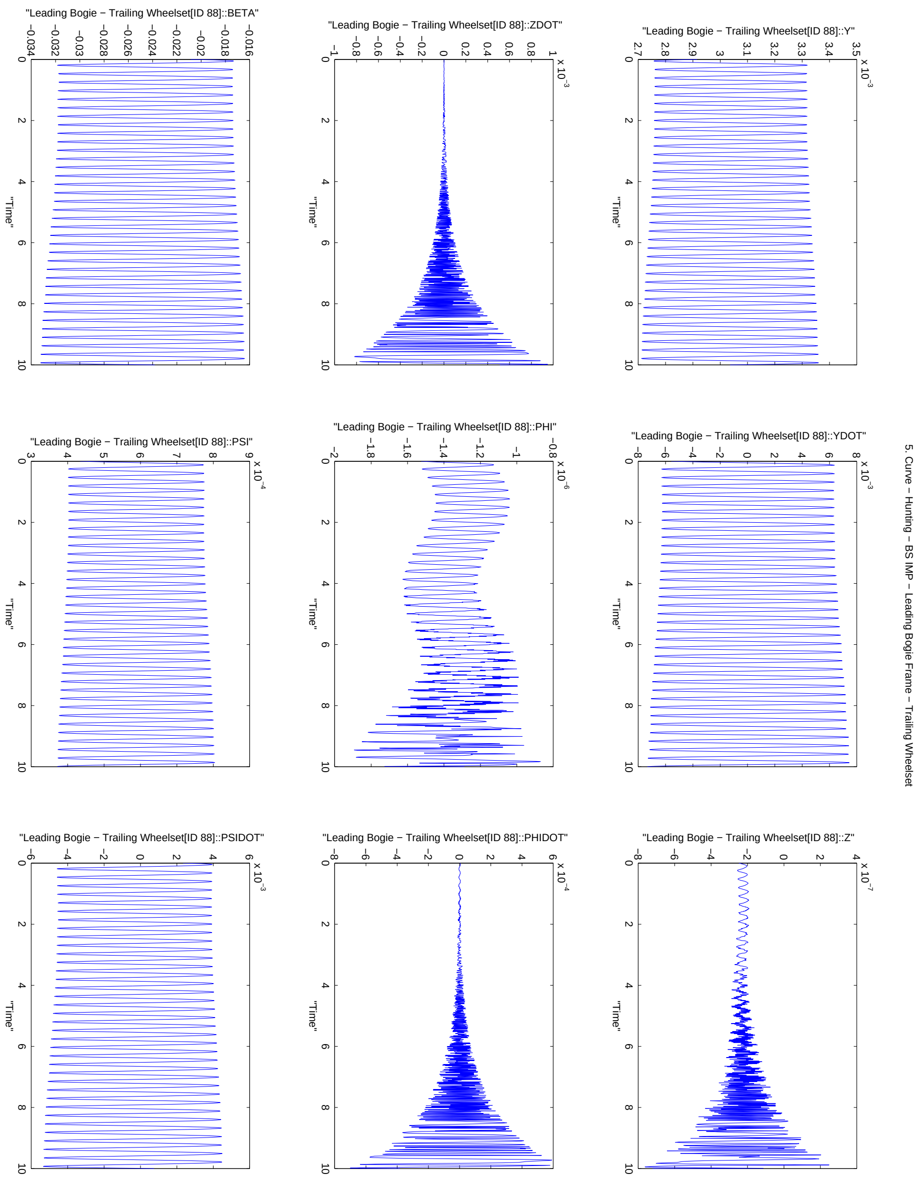


Figure F.187: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the Bulirsch-Stoer solver on the test case 5 (transient on curved track with speed 35m/s - hunting).

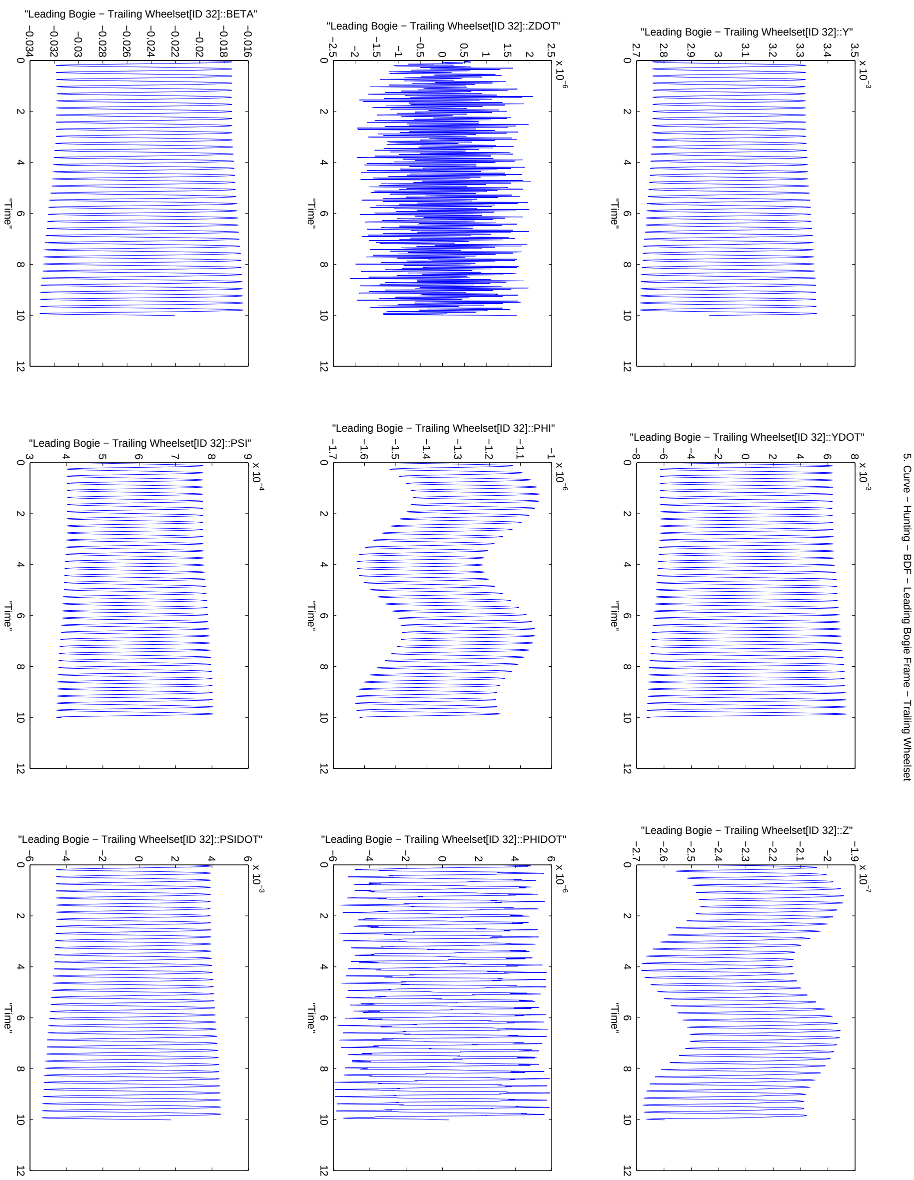


Figure F.188: Performances: dynamics of the trailing wheel set attached to the leading bogie frame obtained using the BDF solver on the test case 5 (transient on curved track with speed 35m/s - hunting).

Appendix G

Scripts

Listing G.1: `constRSGEO.m`

```
function RSGEO = constRSGEO(L1,L2,L3,L4,normalLoad)
    RSGEO = zeros(size(L1,1),13);

    % Displacement
    RSGEO(:,1) = L1(:,1);

    % Normal Forces
    RSGEO(:,2) = normalLoad*cos(L2(:,2));

    % Contact angle
    RSGEO(:,3) = L2(:,2);

    % Contact patch
    RSGEO(:,4) = L3(:,6);
    RSGEO(:,5) = L3(:,7);

    % Distance center wheelset - cp
    RSGEO(:,6) = L1(:,4) + 0.75;

    % Rolling radius
    RSGEO(:,7) = L2(:,6);

    % C11
```

```
    RSGEO(:,8) = (L4(:,2)*2*(1+0.27))./(L3(:,6).*L3(:,7)*2.1
        e11);

    % C22
    RSGEO(:,9) = (L4(:,4)*2*(1+0.27))./(L3(:,6).*L3(:,7)*2.1
        e11);

    % C23
    RSGEO(:,10) = (L4(:,6)*2*(1+0.27))./((L3(:,6).*L3(:,7))
        .^(3/2)*2.1e11);

    % Vertical displacement of CP on the rail
    RSGEO(:,11) = L1(:,11);

    % Penetration
    RSGEO(:,12) = -L3(:,2);

end
```

Listing G.2: `constRSGEO2.m`

```
%%
% CONSTRSGEO2
% Input:
%   - T3, T4:     tables for one contact point of the right
    wheel as given by
%                 RSGEO 2.99
%   - normalLoad: load on the vertical direction for each
    wheel
%   - latRunningTread:   lateral distance from the center of
    the wheel set to
%                        the running tread of the wheel
%   - upsilon: Poisson's Ratio
%   - E: Young's modulus
%
function RSGEO = constRSGEO2(T3,T4,latRunningTread,upsilon,E
    )
    RSGEO = zeros(size(T3,1),13);

    % Displacement
    RSGEO(:,1) = T3(:,1)*10^-3;

    % Normal force
    RSGEO(:,2) = T4(:,2)*10^3 .* cos(atan(T3(:,13)));

    % Contact angle
    RSGEO(:,3) = atan(T3(:,13));

```

```
    % Contact patch (a,b)
    RSGEO(:,4) = T4(:,7)*10^-3;
    RSGEO(:,5) = T4(:,8)*10^-3;

    % Distance Center wheelset - CP
    RSGEO(:,6) = T3(:,3)*10^-3 + latRunningTread;

    % Rolling radius
    RSGEO(:,7) = T3(:,21)*10^-3;

    % C11
    RSGEO(:,8) = (T4(:,13) * 2 * (1 + upsilon))./(T4(:,7)
        *10^-3 .* T4(:,8)*10^-3 * E);

    % C22
    RSGEO(:,9) = (T4(:,14) * 2 * (1 + upsilon))./(T4(:,7)
        *10^-3 .* T4(:,8)*10^-3 * E);

    % C23
    RSGEO(:,10) = (T4(:,15) * 2 * (1 + upsilon))./((T4(:,7)
        *10^-3 .* T4(:,8)*10^-3).^(2/3) * 1e-6^(2/3) * E);

    % Vertical displacement of CP on the rail
    RSGEO(:,11) = T3(:,7)*10^-3;

    % Penetration
    RSGEO(:,12) = -T4(:,3);
end
```

Listing G.3: continuationRSGEO

```
function RSGEOs = continuationRSGEO(RSGin)

    RSGEOs = RSGin;

    dim1 = size(RSGin{1},1);
    dim2 = size(RSGin{1},2);

    % Initialize the RSGEO tables
    for i = 1:length(RSGin)
        if ((dim1 ~= size(RSGin{i},1)) || (dim2 ~= size(
            RSGin{i},2)))
            error('cont:Err','The dimensions of the rsgeo
                tables do not agree');
        end
        RSGEOs{i} = zeros(size(RSGin{i}));
        RSGEOs{i}(1,:) = RSGin{i}(1,:);
    end
```

```

    % Continue the tables with the nearest values following
        the coordinates
    % given by column 6
    colCheck = 6;
    for i = 2:dim1
        markingVector = zeros(length(RSGin),1);

        for j = 1:length(RSGin)
            idxMin = 0;
            min = inf;
            for k = 1:length(RSGEOs)
                if (markingVector(k) == 0)
                    if ( abs(RSGEOs{k}(i-1,colCheck)-RSGin{j
                        }(i,colCheck) ) < min )
                        idxMin = k;
                    end
                end
            end

            markingVector(idxMin) = 1;
            RSGEOs{idxMin}(i,:) = RSGin{j}(i,:);

        end

    end

end
```

Listing G.4: plotRSGEO

```
function [] = plotRSGEO(TABS)
    text = {'rh(51)','Normalforce','DATAN(TANDJA(1,1))','a
        (1,1)','b(1,1)','BJA(2,1,1)+0,75','ROJ(1,1)','c11','
        c22','c23','BJA(3,1,1)','Penetration','BJA(2,1,1)
        +0.75+rh(51)'};

    paperSizeHoriz = 1357;
    paperSizeVert = 960;

    f0 = figure();
    set(f0, 'Position', [0 0 paperSizeHoriz paperSizeVert] )
        ;
    c = {'-b','--r','-.g','.k'};
    for j=1:length(TABS)
        tab = TABS{j};
        for i=1:12
            subplot(3,4,i);
```

```
            hold on;
            plot(tab(:,1),tab(:,i),c{j});
            ylabel(text{i});
            hold off;
        end
    end
end
```

Listing G.5: `writeRSGEO`

```
function [] = writeRSGEO(filename,RSGEO)

    dlmwrite(filename,RSGEO, 'delimiter',' ', 'precision', '
        %1.10e');

end
```

Listing G.6: `mergeRSGEO.sh`

```
#!/bin/bash

EXPECTED_ARGS=2
E_BADARGS=65

if [ $# -lt $EXPECTED_ARGS ]
then
  echo "Usage: `basename $0` [output] [inputs...]"
  exit $E_BADARGS
fi


i=0

for f in $@
do
    if [ $i -lt 1 ]
    then
   let i=$i+1
    else
   echo '#' >> $1
   echo '"rh(51)" "Normalforce"  "DATAN(TANDJA(1,1))" "a
       (1,1)" "b(1,1)" "BJA(2,1,1)+0,75" "ROJ(1,1)"  "c11" "
       c22" "c23" "BJA(3,1,1)"   "Penetration"  "BJA(2,1,1)
       +0.75+rh(51)"' >> $1
   cat ${f} >> $1
    fi
done
```

```
echo >> $1
```

Bibliography

[ABS07] Evert Andersson, Mats Berg, and Sebastian Stichel. Rail Vehicle Dynamics. Kungl Tekniska Högskolan (KTH), 2007. ISBN 978-91-7415-272-2

[Aki70] Hiroshi Akima. *A New Method of Interpolation and Smooth Curve Fitting Based on Local Procedures*. Journal of the ACM, vol. 17(4), (1970), pp. 589–602. ISSN 00045411. doi:10.1145/321607.321609

[ANF11] ANFOR. *French standard - Wheels tread profile - EN13715*. Tech. rep., 2011

[BB89] PN Brown and GD Byrne. *VODE: A variable coefficient ODE solver*. SIAM J. Sci. Stat.

[BD83] G Bader and P Deuflhard. *A semi-implicit mid-point rule for stiff systems of ordinary differential equations*. Numerische Mathematik, vol. 41(3), (1983), pp. 373–398. ISSN 0029-599X. doi:10.1007/BF01418331

[BH75] G. D. Byrne and a. C. Hindmarsh. *A Polyalgorithm for the Numerical Solution of Ordinary Differential Equations*. ACM Transactions on Mathematical Software, vol. 1(1), (1975), pp. 71–96. ISSN 00983500. doi:10.1145/355626.355636

[Big10] Daniele Bigoni. *Curve dynamics on railway vehicles*. Tech. rep., IMM, Technical University of Denmark, 2010

[Bre73] Richard Peirce Brent. Algorithms for Minimization Without Derivatives. Prentice-Hall, Englewood Cliffs, 1973. ISBN 0130223352

[BS66] Roland Bulirsch and Josef Stoer. *Numerical treatment of ordinary differential equations by extrapolation methods.* Numerische Mathematik, vol. 8(1), (1966), pp. 1–13. ISSN 0029-599X. doi:10.1007/BF02165234

[But64] J. C. Butcher. *Implicit Runge-Kutta Processes.* Mathematics of Computation, vol. 18(85), (1964), p. 50. ISSN 00255718. doi:10.2307/2003405

[CEN05] CEN. Railway applications. Testing for the acceptance of running characteristics of railway vehicles. Testing of running behavior and stationary tests - EN 14363. Brussels, 2005. ISBN 0580481522

[Chr01] Lasse Engbo Christiansen. *The Dynamics of a Railway Vehicle on a Disturbed Track.* Tech. rep., Technical University of Denmark, 2001

[Dah63] Germund G. Dahlquist. *A special stability problem for linear multistep methods.* BIT, vol. 3(1), (1963), pp. 27–43. ISSN 0006-3835. doi:10.1007/BF01963532

[GSL] GSL. *GNU Scientific Library*

[HBG05] AC Hindmarsh, PN Brown, and KE Grant. *SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers.* ACM Transactions on, vol. 31(3), (2005), pp. 363–396

[Her95] H. Hertz. *Uber die Berührung Fester Elastischer Körper: Gesammelte Werke.* Band I, Leipzig, vol. 171, (1895), pp. 156–171

[HNrW91] E Hairer, S P Nø rsett, and G Wanner. Solving Ordinary Differential Equations I: Nonstiff Problems. Springer Series in Computational Mathematics. Springer-Verlag, second rev edn., 1991

[Hof06] Mark Hoffmann. Dynamics of European two-axle freight wagons. Ph.D. thesis, IMM, The Technical University of Denmark, 2006

[HP02] Mark Hoffmann and Dan Erik Petersen. *Curving Dynamics of Railway Vehicles.* Tech. rep., Technical University of Denmark, 2002

[Kal90] J J Kalker. Three-Dimensional Elastic Bodies in Rolling Contact. Kluwer Academic Publishers, 1990

[Kal91] JJ Kalker. *Wheel-rail rolling contact theory.* Wear, vol. 144(1-2), (1991), pp. 243–261

[Kli82] R Klinger. *Über del Lauf der Eisenbahnwagen auf gerader Bahn.* Organ Fortschr Eisenbahnwesens XX, vol. 4, (1882), pp. 113–123

[KM10] Walter Kik and Dirk Moelle. *ACRadSchiene - To create or Approximate Wheel/Rail profiles - Tutorial*

[KTJr08] Morten Rode Kristensen, Per Grove Thomsen, and John Bagterp Jø rgensen. *A family of ESDIRK integration methods*. SIAM Journal on Scientific Computing, pp. 1–22

[KV] J.J. Kalker and E.A.H. Vollebregt. *CONTACT*

[Law67] J. Douglas Lawson. *Generalized Runge-Kutta Processes for Stable Systems with Large Lipschitz Constants*. SIAM Journal on Numerical Analysis, vol. 4(3), (1967), p. 372. ISSN 00361429. doi:10.1137/0704033

[Lev07] Randall Leveque. Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems (Classics in Applied Mathematics). SIAM, Society for Industrial and Applied Mathematics, 2007. ISBN 0898716292, 357 pp.

[Nad08] M.J. Nadal. *Locomotives a Vapeur, Collection Encyclopedie Scientifique*. Biblioteque de Mecanique Appliquee et Genie, Paris, 1908

[NB95] A H Nayfeh and B Balachandran. Applied nonlinear dynamics. John Wiley & Sons, Inc., 1995

[NT93] Hans Bruun Nielsen and Per Grove Thomsen. Hæfte 66 - Numeriske Metoder for Sædvanlige differentialligninger. Numerisk Istitut, DTH, 1993

[Os98] Erik Ø stergaard. *Documentation for the SDIRK C++ Solver*. Tech. Rep. 2, Technical University of Denmark, Jul 1998

[Pro07] The Apache XML Project. *Xerces-C++ XML Parser*, 2007

[PS91] J.P. Pascal and Gilles Sauvage. *New method for reducing the multi-contact wheel/rail problem to one equivalent contact patch*. In The dynamics of vehicles on roads and on tracks, Proceedings of 12th IAVSD Symposium, p. 700. SWETS & ZEITLINGER inc. Berwin, Pa, 1991

[SÖ2] Gustaf Söderlind. *Automatic control and adaptive time-stepping*. Numerical Algorithms, (31), (2002), pp. 281–310

[Sha83] LF Shampine. *Smoothing the extrapolated midpoint rule*. Numerische Mathematik, vol. 175, (1983), pp. 165–175

[SHE84] Z Y Shen, J K Hedrick, and J A Elkins. *A comparison of alternative creep-force models for rail vehicle dynamic analysis*. In 8th IAVSD Symposium, pp. 591–605, 1984

[Ste21] G. Stephenson. *Observations on Edge and Tram Railways*

[Tru99] Hans True. *On the Theory of Nonlinear Dynamics and its Applications in Vehicle Systems Dynamics*. Vehicle System Dynamics, vol. 31(5-6), (1999), pp. 393–421. ISSN 0042-3114. doi: 10.1076/vesd.31.5.393.8361

[UIC05] UIC. Testing and approval of railway vehicles from the point of view of their dynamic behaviour - Safety - Track fatigue - Ride quality - Code 518 OR. Paris, 2 edn., 2005. ISBN 978-2-7461-1642-9, 129 pp.