Natural frequency analysis of Modelica powertrain models

Garron Fish Mike Dempsey Juan Gabriel Delgado Neil Roberts
Claytex Services Ltd
Leamington Spa, United Kingdom
garron.fish, mike.dempsey, juan.delgado, neil.roberts @claytex.com

Abstract

The natural frequency analysis of complex powertrain models created in Modelica presents a number of problems. This paper presents the basic principles and some of the problems associated with carrying out this kind of analysis. As a result of this work, a new feature in the Powertrain Dynamics Library has been developed to automate these methods and provide the end-user with a simple set of functions to perform natural frequency analysis. Simple examples are used to illustrate the problems and solutions and a complex powertrain model is then analysed using the library.

Keywords: modal analysis; natural frequency; linearization; powertrain; NVH

1 Introduction

Modal analysis is the study of the dynamic response of a system at its resonance frequencies. Modal analysis is used in many fields for example in structural engineering to design buildings resistant to earthquakes [1] and in vehicle powertrain design to avoid poor NVH characteristics [2].

For a vehicle, modal analysis is carried out on all parts of the car to determine their natural frequencies. Care is taken to make sure that the natural frequencies of the parts in the car are all at distinct, separate frequencies. If the natural frequencies are not suitably separated this can lead to resonance across multiple parts of the car and a poor NVH characteristic.

A new feature has been introduced in the PTDynamics library [3] [4] to perform the natural frequency analysis of powertrain models created using this library. This paper highlights some of the problems involved with this type of analysis based on Modelica models and discusses some of the techniques developed to solve these.

To determine the natural frequencies of a model and the corresponding modal response we start by linearising the model at the required operating point. Linearisation of a model using Dymola returns the state-space representation of the model and from this the natural frequencies can be calculated. The natural frequencies are found when all damping in a model is removed.

2 Modal frequency analysis and Modelica models

2.1 Basic Principles

This section looks at the basic modal analysis principles applied to a spring mass network. The example of a spring mass network has been chosen so that the natural frequency of a model can be described. An unforced spring mass network can be represented by the following ordinary linear differential equation:

$$\boldsymbol{M\ddot{s} + C\dot{s} + Ks} = 0$$

It is common to calculate the natural frequency of the above equation with the damping term set to zero so the equation becomes:

$$\boldsymbol{M\ddot{s} + Ks} = 0 \qquad (1)$$

The natural frequency of the spring mass system can now be calculated from the roots of the above equation. The roots are the eigenvalues and eigenvectors of the equation.

To perform modal analysis on complex models we linearise these first which generates the state space representation of the model. The state space representation of a model is given by:

$$\begin{aligned} \boldsymbol{\dot{x}} &= \boldsymbol{Ax + Bu} \\ \boldsymbol{y} &= \boldsymbol{Cx + Du} \end{aligned} \qquad (2)$$

where:
A, **B**, **C** and **D** are matrices
u is the vector of inputs

y is the vector of outputs
x is the vector of states

To rearrange our simple spring-mass system in to state space form is done by transforming equation (1) in to the following form:

$$\ddot{\boldsymbol{s}} = -\boldsymbol{M}^{-1}\boldsymbol{K}\boldsymbol{s}$$

In this simple example, there are no inputs so the u term is dropped and there are no outputs so the equation for y is not required. The model is then reduced to:

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} \qquad (3)$$

where

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{S} \\ \dot{\boldsymbol{s}} \end{bmatrix}$$

and

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 \\ -\boldsymbol{M}^{-1}\boldsymbol{K} & 0 \end{bmatrix}$$

2.2 Eigenvalues and eigenvectors

For a given matrix A the eigenvalues and eigenvectors are calculated such that:

$$\boldsymbol{A}\boldsymbol{v} = \lambda\boldsymbol{v}$$

where:
$\boldsymbol{v}$ is the eigenvector associated with the eigenvalue
λ is an eigenvalue

The eigenvalue solutions, are the roots of:

$$[\lambda\boldsymbol{I} - \boldsymbol{A}]\boldsymbol{x} = \boldsymbol{0}$$

All the eigenvalues are included in vector $\boldsymbol{\lambda}$ that is referred to as the eigenvalues of A. The eigenvectors are combined row wise into matrix **v**. The eigenvectors and eigenvalues of this equation are calculated so that the natural frequency can be calculated as follows in section 2.3.

2.3 Frequency and damping

The natural frequency is calculated from the eigenvalues as [5]:

$$\boldsymbol{f} = \frac{|\boldsymbol{\lambda}|}{2\pi}$$

where:
|| is the complex norm
$\boldsymbol{f}$ is frequency in Hertz

The complex norm is the sum of the squares of the real and imaginary parts all square rooted. There is also a damping term that is associated with each eigenvalue. In the case where the damping has been set to zero, this term will be zero and will not influence the natural frequencies of the model. The damping term can be calculated with the following equation [6]:

$$\zeta = \begin{Bmatrix} 0, |\lambda| = 0 \\ \dfrac{\mathrm{Re}(\lambda)}{|\lambda|}, |\lambda| \neq 0 \end{Bmatrix}$$

where:
Re() is the real part of a complex number

The frequency that a model with damping oscillates at without being driven by an outside force is referred to as the *damped frequency* and using eigenvalue analysis this is calculated as :

$$w_d = w_n\sqrt{1-\zeta^2}$$

2.4 Issues for complex Modelica models

The current analysis described above can be easily performed on a spring mass network but it is not as easy to implement this on a complex Modelica model. A number of issues arise when trying to apply this process using a Modelica tool such as Dymola.

A complex model will contain a large number of state variables and we would normally expect to find many states that do not have any effect on the natural frequency response of the physical states of the model. For example, states within a driver model or control system that do not directly influence the physical response of the system. These states should be removed from the analysis to reduce the time taken to do the analysis.

Some Modelica tools are able to compile models using dynamic state selection. Currently models that use dynamic states cannot be analysed and a fixed set of states needs to be applied to the model. This has to be done by the user before starting the frequency analysis.

In the simple spring mass network presented so far we have not considered the possibility of the relative state of the spring being selected as a state rather than the position of the mass. Modelica tools are able to select a set of states from a model and in many cases they will select relative states rather than absolute states. Whilst the natural frequencies of the

system are unaffected by the choice of state variable it is preferable in this type of analysis to use the absolute states of the system. Using the absolute states makes the interpretation of the modal response easier as the points of interest become physical points such as the driveshaft ends or pinion gear rather than relative states such as the driveshaft twist or relative angle between pinion and crown wheels.

Further problems are observed when component models that utilize the standard Modelica friction model are included for analysis. The behaviour of the slip/stick friction models is not linearized in the expected manner and modifications to the analysis have to be made around these components.

To calculate the natural frequencies the damping terms have to be removed from the model but without the damping often models will not simulate. This causes a problem for the initialisation of models and when the model needs to be analysed under different operating conditions, for example, in different gears or under different loading conditions where springs are compressed to different parts of their non-linear force curve.

3 Implementation in the Powertrain Dynamics Library

The Powetrain Dynamics (PTDynamics) library is used to create complex MultiBody models of powertrains in a user friendly and efficient manner. A new feature has been introduced to determine the natural frequencies of these powertrain models. A number of issues are present that make performing the natural frequency analysis difficult when working with Modelica models (refer to Section 2.1). This section describes some of the methods implemented in to the linearization functions available in the PTDynamics library that are used to overcome these issues.

3.1 Relative states

The natural frequencies of the model are typically calculated for positional states (i.e. position or angular position). However when a model is created using Modelica, the modelling tool can choose to select relative states (such as spring extension) rather than positional states (such as the position of ends of the spring). When this is detected in a model the relative states are converted in to positional states before linearizing the model.

The first step in the analysis process is to determine the states used in the model which is done by translating the model and analysing the list of selected states. If relative states are detected then the model has to be modified by adding outputs that measure the positions either side of the component with the relative states, see Figure 1. The model can then be linearized and the resulting A matrix manipulated to transform the relative state in to a positional state. Within the PTDynamics library a precise naming convention is used to enable the automatic detection of relative and absolute states from the variable names.

By only making the transformation from relative to positional state in the linearized model we do not affect how the original model simulates. This means that we can still use the original model to get the system to the desired operating point and then linearize it. If we forced the user to only use

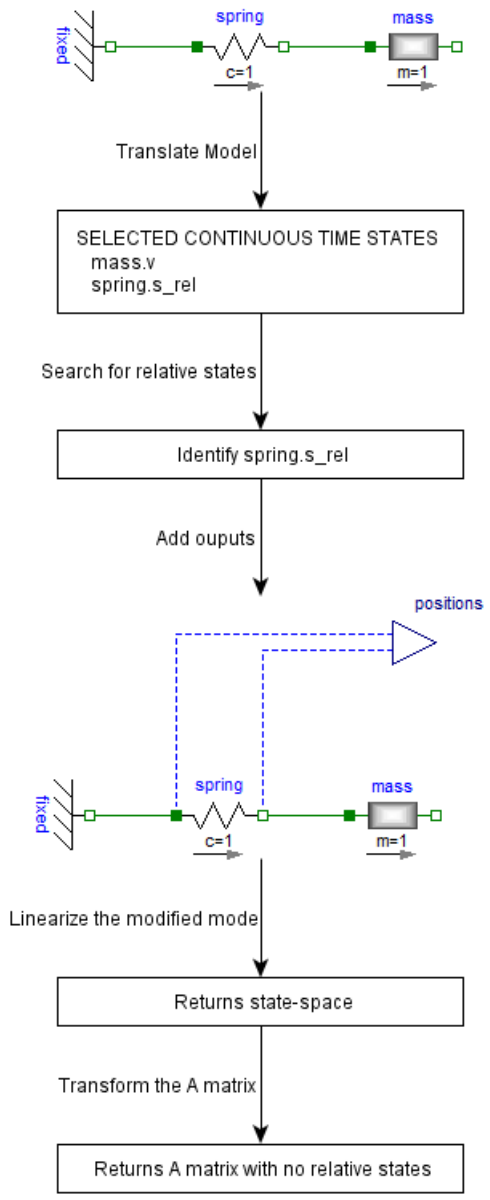

Figure 1: Process to convert relative states to positional states

positional states in the model we may introduce slight differences in to the model due to the different equation solutions required and we could impact the simulation time.

When the modified model, with the added outputs, is linearized, the resulting state space representation includes these outputs in the **C** matrix. This matrix relates the position outputs to the states in the model. Each relative state will generate two outputs but only one of these outputs will be related to the relative state by the **C** matrix. This state is used to replace the relative state.

Using the spring mass model as an example we can see how this manipulation of the A matrix should be performed. Linearizing the modified model gives the following:

$$A = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

$$state\ names = \{mass.v, spring.s\_rel\}$$
$$output\ names = \{position1, position2\}$$

From the C matrix it is seen that **position2** is related to spring.s_rel as:

$$\mathbf{position2} = \boldsymbol{C}[2,:]\boldsymbol{x}$$

where:
$\boldsymbol{x}$ are the states of the model

A transformation matrix is now created that transforms $\boldsymbol{x}$ to a set of states that does not contain relative states. In this example the transformation matrix would be:

$$\boldsymbol{T} = \begin{bmatrix} 1 & 0 \\ \boldsymbol{C}[2,1] & \boldsymbol{C}[2,2] \end{bmatrix}$$

$$\boldsymbol{x}_{pos} = T\boldsymbol{x} \qquad (4)$$

Replacing $\boldsymbol{x}$ in (3) with $\boldsymbol{x}_{pos}$ from (4) gives:

$$\dot{\boldsymbol{x}}_{\boldsymbol{pos}} = \boldsymbol{TAT^{-1}x_{pos}}$$

A drawback of this method is that it can select a state that is only associated with a position and not directly with an actual mass or inertia state. Figure 2 illustrates a case where this behaviour is present. The user currently has to review the selections made during the analysis process to ensure that these situations are avoided.

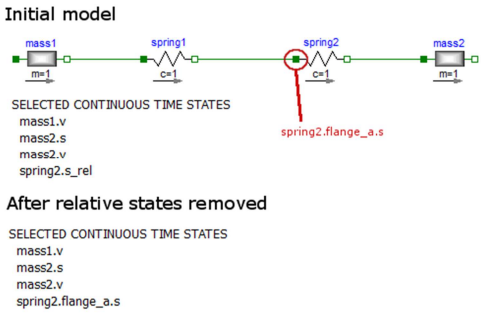

Figure 2: The initial states of the model include spring2.s_rel, this state is replaced with spring2.flange_a.s that is a state without a mass

3.2 Friction components

A number of component models such as clutches and brakes use the Modelica Standard Library coulomb friction model [7] that handles the stuck and sliding modes in a clean way using state events. When this is linearized using the built-in Dymola function the model is sometimes linearized as if in the slipping mode regardless of the actual state of the component. A method has been developed to adjust the model and resulting state space model to correctly account for the friction state.

Figure 3 shows an overview of the automatic process that is used to overcome this using the PTDynamics library. First the model is translated and the names of the selected states are analysed to determine if there are any states that relate to friction and to determine what state the friction model is in at the instant that the model is being linearized at.

If the friction model is in the stuck mode then it is necessary to join the positional states in the A matrix that are either side of the frictional component. To be able to join states in the A matrix it is necessary to calculate the mass/inertia of the states being joined together. This is done by adding torque inputs to the corresponding positional states either side of the friction component.

In the example shown in Figure 3, we would detect the friction states within the clutch and then modify the model. In addition to adding a torque input either side of the friction model we also need to add position outputs either side of the friction model so that we can join the states in the locked mode.

After the modified model is linearized the **B** matrix is used to determine the mass of the states. This information together with the state space **C** matrix can then be used to update the **A** matrix by joining the states on either side of the friction component.

The mass of the states is determined as follows, the basic equation describing a spring mass system that contains a force is:

$$\boldsymbol{ma = ks + dv + F}$$

where:

$\boldsymbol{m}$ is mass

$\boldsymbol{s}$ is position

$\boldsymbol{v}$ is velocity

$\boldsymbol{a}$ is acceleration

$\boldsymbol{k}$ is stiffness

$\boldsymbol{d}$ is damping vector

$\boldsymbol{F}$ is the applied force

In the example shown in Figure 3, the positional states that the clutch is connected to are independent

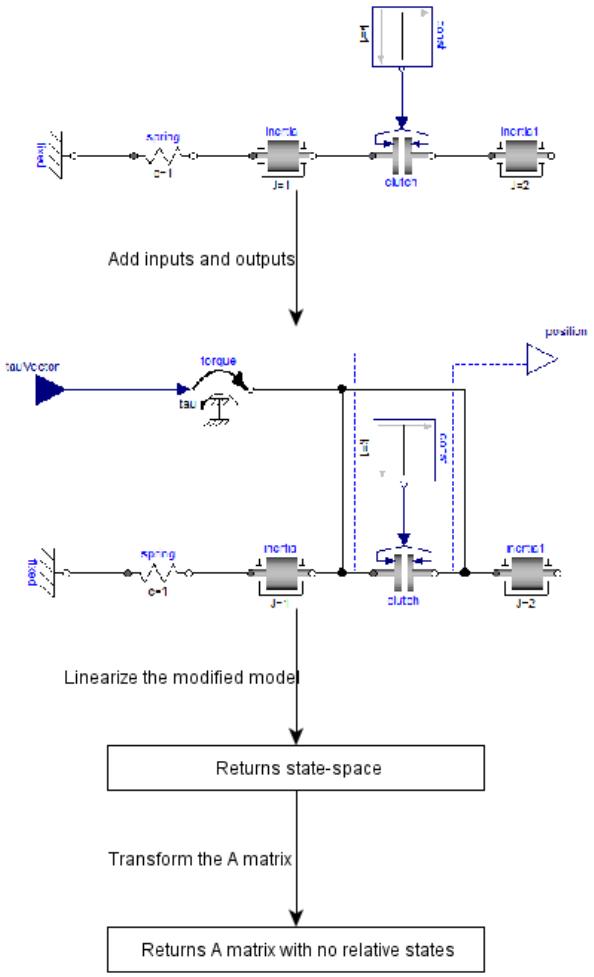

Figure 3: Process to handle friction components

which means the following equation can be used to describe both states that need to be joined together and rearranged as:

$$\ddot{s} = m^{-1}ks + m^{-1}d\dot{s} + m^{-1}F$$

The state space representation of this equation is:

$$\begin{bmatrix}\dot{s}\\ \ddot{s}\end{bmatrix} = \begin{bmatrix}0 & 1\\ m^{-1}k & m^{-1}d\end{bmatrix}\begin{bmatrix}s\\ \dot{s}\end{bmatrix} + \begin{bmatrix}0\\ m^{-1}\end{bmatrix}F \quad (5)$$

From (2) and (5), we can determine that the state space **B** matrix is equal to $\begin{bmatrix}0\\ m^{-1}\end{bmatrix}$, so the mass/inertia for the states to be joined can be calculated. Using the example shown in Figure 3, we get the following values when linearising the modified model.

$$A = \begin{bmatrix}0 & 1 & 0 & 0\\ -1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0\end{bmatrix}$$

$$B = \begin{bmatrix}0 & 0\\ 1 & 0\\ 0 & 0\\ 0 & 0.5\end{bmatrix}$$

$$C = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}$$

$$state\ names = \{i1.phi, i1.w, i2.phi, i2.w\}$$
$$input\ names = \{tau1, tau2\}$$
$$output\ names = \{position1, position2\}$$

Using the **B** matrix we can determine the inertia of the two bodies either side of the clutch.

$$m_1 = \frac{1}{B_{2,1}} = 1$$
$$m_2 = \frac{1}{B_{4,2}} = 2$$

To modify the A matrix we use the B matrix to determine the rows in the A matrix that should be combined. The C matrix is then used to determine the columns that need to be combined. After combining the rows and columns we can remove the redundant rows and columns from the A matrix.

In this example we find that the 2nd and 4th rows need to be combined as well as the 1st and 3rd columns which results in:

$$A = \begin{bmatrix}0+0 & 1\\ \left(\frac{(-1+0)*m_1 + 0*m_2}{m_1+m_2}\right) & \left(\frac{(0+0)*m_1 + 0*m_2}{m_1+m_2}\right)\end{bmatrix}$$

$$A = \begin{bmatrix}0 & 1\\ 0.333 & 0\end{bmatrix}$$

To include damping effects when joining states using this method the columns corresponding to the rows determined from the B matrix need to be added together as well.

There is a known limitation of the joining method demonstrated here and used in the PTDynamics library in that the states being joined together must be independent states. This means that the positional state must not be dependent on other positional states. An example of a component that has dependent states is a planetary gear where the rotational states of the three shafts are dependent on each other. To overcome this limitation a flexible shaft has to be connected between a clutch and a planetary gear in a gearbox to be able to join the states on either side of the clutch using this method.

4 Applications

4.1 Simple example

This simple example contains three inertias with the first two separated by a clutch and the second and third inertia separated by a spring as shown in Figure 4. A ramp input is used to actuate the clutch and goes from 0 at 0s to 1 at 1s. The response for the clutch state and the speeds of the inertias either side are shown in Figure 5.

If the model is linearized at t=0s, i.e. when the clutch is open we find the natural frequency is at 5.29Hz. If the model is linearized at t=2s, when the clutch is locked, the natural frequency occurs at 2.20Hz.

The change in frequency occurs because the total effective inertia on the left hand side of the spring has changed. Without using the method to join the states either side of the clutch the built in functions report no change in the natural frequency despite the change in configuration of the model.

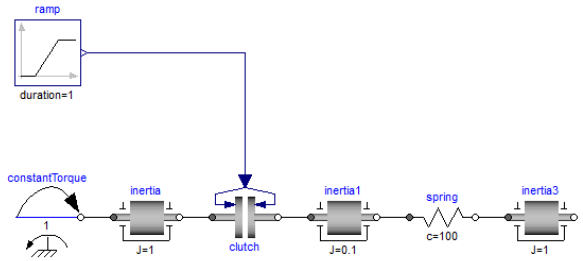

Figure 4. Simple model that contains a clutch and a spring

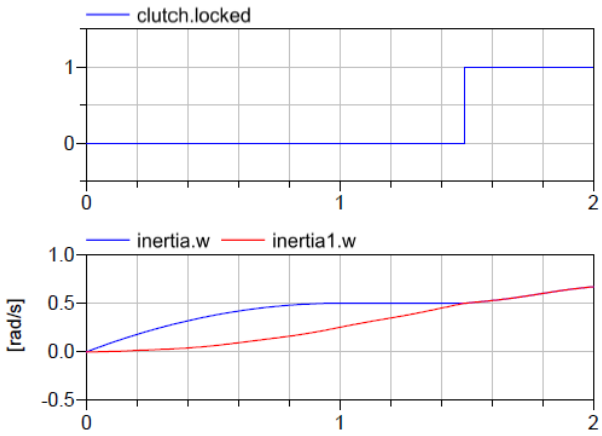

Figure 5. Plots of locked and angular velocity of inertia and inertia1 in the Simple model in Figure 4.

4.2 Full vehicle example

A model of a front engine, rear-wheel drive vehicle with a manual transmission was constructed using the PTDynamics library it fully test the new functions and methods. The model is shown in Figure 6. The engine model is a simple mapped engine model but the transmission and driveline are more detailed. Figure 7 shows the gearset model from within the transmission. The gearset and driveline models include torsional compliance in a number of the shafts but are rigidly mounted within the chassis. Overall this model has a good torsional representation of the powertrain system and would be suitable for studying driveability events such as tip-in and tip-out.

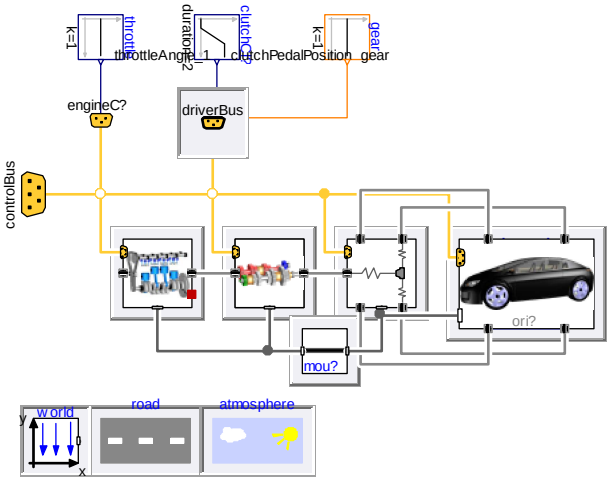

Figure 6. PTDynamics vehicle example that is linearized

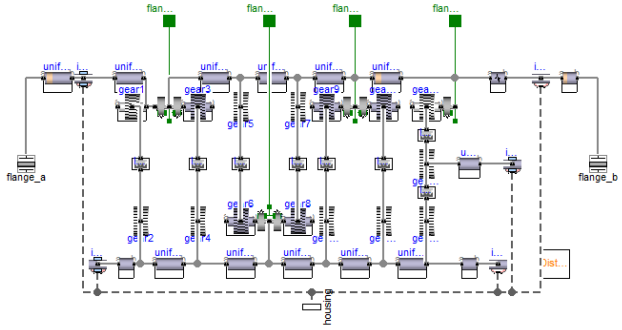

Figure 7. Gearset used in vehicle example.

The chassis model doesn't include suspension but the tyres do include a slip model based on the well-known Pacejka tyre model. This required the development of a method that relates the wheel rotation to the chassis movement. This was necessary because the slip models are based on velocity relationships but for this type of analysis we need the relationships to be based on position. The method developed assumes that the ratio between the wheel rotation and the chassis motion is a fixed ratio at the instance that linearization occurs. The details of this method are not described in this paper.

The model was linearized and the following natural frequencies are found (in Hz): 5.1, 35, 124, 266 and 343. The 5.1Hz response is the shuffle frequency of the vehicle and the modal response is shown in Figure 8. The x-axis of the modal response plots is an integer that corresponds to the states listed in Table 1. The magnitudes are normalised with respect to the variable with the largest displacement.

The modal response shows that at this frequency there is very little motion of the chassis but the whole powertrain is moving out of phase with the chassis and at relatively large displacements.

No.	State
1	transmission.clutch.drivenPlate.flange_a.flange.phi
2	transmission.gearset.uniformShaft10.body_a.phi
3	transmission.gearset.uniformShaft.body_a.phi
4	driveline.rearDifferential.pinion.phi
5	driveline.rearDifferential.differentialAssembly.outputGear_2.phi
6	chassis.motion.prismatic_x.s

Table 1. States of simple vehicle. Each number corresponds to a state. The number in the legends in Figure 8 corresponds to the number in this table.

It is also possible to generate Bode diagrams for different inputs and outputs of the vehicle model. The example shown in Figure 8 is the bode diagram generated when engine torque is an input to the system and the differential pinion gear rotation angle is the output. The Bode plotting function in Modelica_LinearSystems2 is used to generate the actual plot.

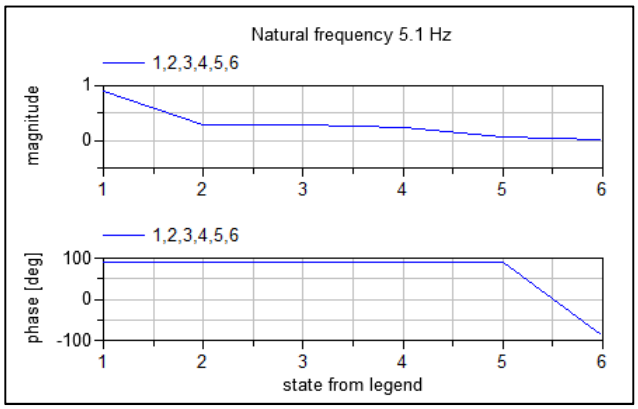

Figure 8. Modal response of the vehicle model at 5.1Hz. The magnitude and phase of the different states are plotted. Each state is assigned to a position along the x axis as determined by the legend. The numbers in the legends correspond to the states in Table .

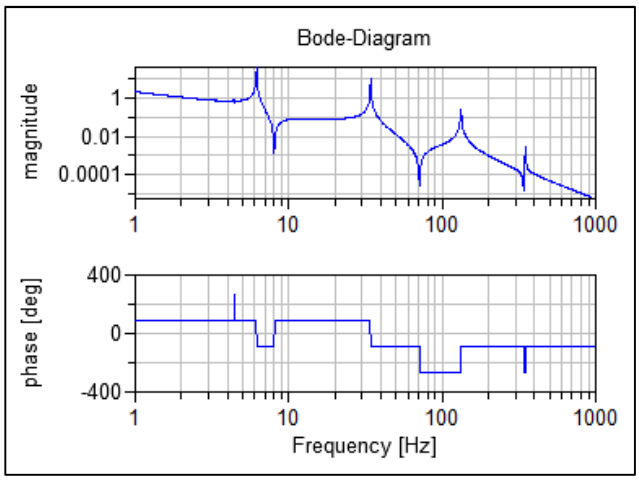

Figure 9. Bode diagram with Engine torque as the input and differential pinion position as the output.

5 Conclusion

A new method for determining the natural frequencies and modal responses of complex Modelica models has been developed and introduced as a new feature in the Powertrain Dynamics library. This feature includes automated methods to handle the problems with relative states and friction components as described in this paper in addition to other methods to handle further problem areas such as tyre slip models. The feature will be further improved to provide animation of the modal response of the powertrain to aid the understanding of the natural frequencies of the powertrain system.

References

[1] M. Paz, “International handbook of earthquake engineering,” in *International handbook of earthquake engineering*, London, Chapman & Hall, 1994, p. 283.

[2] H. Gagnon and G. Gagnon, “Identifictation of powertrain noise sources using sound intensity and modal analysis techniques,” *Proceeding of SPIE, the International Society for Optical Engineering*, vol. 3089, no. 2, pp. 2016-2020, 1997.

[3] N. Roberts and M. Dempsey, “Predicting the launch feel of automatic and dual clutch transmissions,” in *Proceedings 9th Modelica Conference 2012*, Munich, 2012.

[4] M. Dempsey and A. Picarelli, “Investigating the MultiBody dynamics of the complete powertrain system,” in *Proceedings 7th Modelica Conference*, Como, 2009.

[5] MSC Software, “MD Nastran 2010, Dynamic Analysis User's Gude,” 2010, p. 47.

[6] A. Mohamed, “Modelling, simulation and identification,” Rijeka, Sciyo, 2010, p. 68.

[7] M. Otter, H. Elmqvist and S. E. Mattsson, “Hybrid Modeling in Modelica based on Synchronous Data Flow Principle,” in *CACSD'99*, Hawaii, 1999.