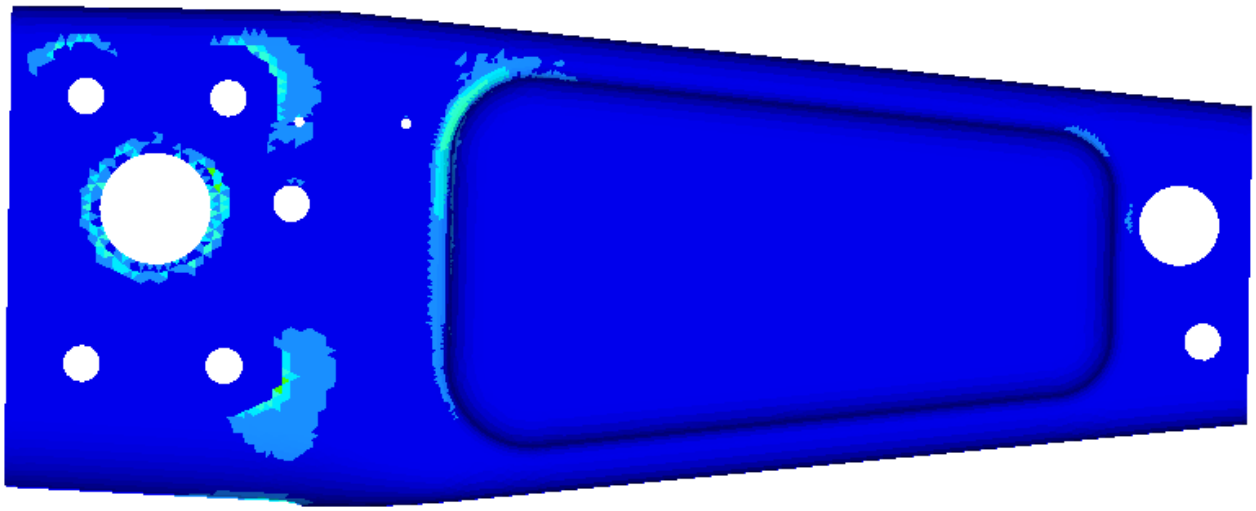
CHALMERS



Development of Force based Resonance Order Method employed on whole engine with combustion loads

Master's thesis in Applied Mechanics

ADAM ERIKSSON

Department of Applied Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2016

MASTER'S THESIS IN APPLIED MECHANICS

Development of Force based Resonance Order Method employed on whole engine with combustion loads

ADAM ERIKSSON

Department of Applied Mechanics

Division of Dynamics

CHALMERS UNIVERSITY OF TECHNOLOGY

Göteborg, Sweden 2016

Development of Force based Resonance Order Method employed on whole engine with combustion loads
ADAM ERIKSSON

Master's thesis 2016:15
ISSN 1652-8557
Department of Applied Mechanics
Division of Dynamics
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
Amplitude stress distribution in auxiliary component analyzed with the FROM.

Chalmers Reproservice
Göteborg, Sweden 2016

Development of Force based Resonance Order Method employed on whole engine with combustion loads
Master's thesis in Applied Mechanics
ADAM ERIKSSON
Department of Applied Mechanics
Division of Dynamics
Chalmers University of Technology

Abstract

This thesis regards the continued development of the Resonance Order Method, ROM, which is used at Scania in order to find critical engine speeds with regard to fatigue for auxiliary components fastened to the engine.

The ROM uses dynamic simulations in order to extract nodal excitations at a set of fixed engine speeds in the engines performance spectrum whereafter the excitation is transformed into the frequency domain by a Fourier transform. This allows for identification of the frequency components at what is known as engine orders.

By computing the eigenfrequencies of the auxiliary component to be analyzed, and matching these eigenfrequencies to the linear relations represented by the engine orders, it is possible to identify what is known as critical engine speeds which causes high amplitude stress. The result of running the ROM method is a maximum stress amplitude and a corresponding critical engine speed for each element in the analyzed component.

The purpose of this thesis is to further develop the ROM, generalize it, make it more accessible to the end users and to broaden the spectrum of what can be analyzed with the help of it. This is done by creating a whole engine model which is loaded by force loads at the crank shaft bearings as opposed to using nodal excitation at the fastening interface. This new method is called the Force based Resonance Order Method, FROM.

A methodology to create this whole engine model has been developed during this thesis, consisting of a suite of preprocessing-scripts that create and apply the force loads to the model and a set of post-processing scripts that format the results to be easily accessible. A comprehensive step-by-step user guide has been written in order to make the method accessible for potential users.

The FROM has also been verified by applying it to an bracket fastened to the engine and comparing the results to physical testing and other computational methods used at Scania. The verification could not be unambiguously finished within the designated time frame, but the tests performed indicate a successful adaption of the method.

Keywords: ROM, FROM, Fourier series, Engine orders, Abaqus, Fatigue life

Preface

This master thesis project has been conducted at NMBS, the group of Strength Analysis for engine development at Scania in Södertälje. The thesis regards the further development of the Resonance Order Method (henceforth ROM); used for computing critical engine speeds for auxiliary engine components with regard to fatigue. The examiner of the project has been Anders Boström, professor of mechanics at Chalmers University of Technology. David Norman at NMBS has been the supervisor at Scania. Sohail Iftikhar at NMDD and Fredrik Reuterswärd at NMBS have also aided the project greatly.

Abbreviations

Abaqus	-	*The FEA software primarily used by NMBS.*
NASTRAN	-	*The FEA software primarily used by NMDD for dynamic models.*
NMBS	-	*The group for strength analysis during engine development.*
NMDD	-	*The group for dynamic and acoustic analysis during engine development.*
ROM	-	*The Resonance Order Method. The original computational method.*
FROM	-	*The Force based Resonance Order Method. The method developed during this master thesis.*
DFT	-	*Discrete Fourier Transform. Fourier transform for finite data.*
IDFT	-	*Inverse Discrete Fourier Transform,. Transforms signal back to time-domain.*
TMF	-	*Thermo-Mechanical Fatigue, the phenomena where changes in temperature causes fatigue.*
ODB-file	-	*Output Data Base file. A file that contain the results of Abaqus computations*

Contents

1 Introduction

This chapter gives an introduction to the thesis, explaining the background and history of the computational method examined. The goal of the thesis, and why a new method will be developed, is also explained.

During the project a steering group, consisting of members from NMBS and NMDD who have worked with the ROM earlier, met at regular intervals to discuss the progress of the project and guide its development. This provided a good source of information and motivation to drive the project forwards. The progress of the thesis was also coordinated with the examiner once every month, often in conjunction with the steering group meeting.

1.1 Background

A truck engine can be described as consisting of a base engine which is then equipped with a set of auxiliary components to form the complete engine. The components are usually fastened to the engine with some kind of brackets. When the engine is running it vibrates and so do the auxiliary components. However, they do not vibrate in perfect harmony due to the inertia and elasticity of the different parts. This introduces stresses in the structure that ultimately might lead to fatigue. It was proposed that some engine speeds will produce vibrations that are particularly dangerous for each component, as they might excite the auxiliary component at one of its eigenfrequencies. Excitation at an eigenfrequency can be quite devastating over time as has been shown again and again throughout history. The breaking of an engine might not be as dramatic as the collapse of Tacoma Bridge, but to the owner of the truck it would feel that way. As the computational time needed to compute the transient excitation and stresses at all possible engine speeds would be immense, a method to find the most critical engine speeds is needed.

Thus the original Resonance Order Method (ROM) was first developed during a master thesis project in 2004 [Lar04]. The method was able to find critical engine speeds and corresponding stress fields by computing the dynamic response at a set of fixed engine speeds, transforming the response to the frequency domain using Fourier transform and then interpolating between the fixed engine speeds with regard to eigenfrequencies. NASTRAN was used as the FE solver. The method was based on an idea from Ola Jönsson at NMDD who further developed it in 2005 [Jön05]. The effects of Thermo-Mechanical Fatigue (TMF) on the method was also investigated in a master thesis by Josef Daelander in 2009 [Dae09] but has not been implemented in the method since. TMF calculations was, and still are, carried out separately.

A version of the method that utilizes Abaqus instead of NASTRAN as its solver and post-processor was created by Fredrik Reuterswärd in 2012 [Reu12], since Abaqus is the solver primarily used by NMBS. NMBS is the group of Strength Analysis at base engine development and the primary user of the method. The understanding of Abaqus models among the employees at NMBS is far greater than that of NASTRAN. Thus the migration to Abaqus increased the user friendliness, allowing the users more control over the models. The use of Abaqus also allows the ROM to be used to analyze significantly larger components as the data generated is an order of magnitude smaller than that which is generated by NASTRAN.

It should also be noted that the FE computations are performed faster when Abaqus is used as a solver since Scania's infrastructure is better suited for Abaqus than it is for NASTRAN. However, the Abaqus version of the ROM [Reu12] was dependant on the preprocessing conducted for the NASTRAN based ROM [Lar04]; it translated the input data from NASTRAN to Abaqus format before performing the computations.

These two versions of the ROM have utilized a dynamic simulation of the engine where a set of nodes has had to be specified for which to extract excitation data. Extracting the transient data for all nodes in the model is made virtually impossible due to the immense storage space this would require. The excitation data is employed on a model of an auxiliary component that is to be analyzed. This means that for each auxiliary component, and each change made to the auxiliary component, a new dynamic simulation has had to be performed if the stiffness and thus the eigenfrequencies has been altered.

1.2 Goal

The goal of this master thesis was to further develop the ROM to encompass the whole engine, thus creating a methodology where the dynamic simulations need only be performed once for each engine. The result should be a FE model of the whole engine where the load cases, defined by the dynamic simulations, does not change for each change made to the auxiliary components to be examined.

This was to be achieved by extracting the reaction forces in the crank shaft bearings caused by combustion loads, that act upon the engine and cause it to vibrate, from the dynamic simulation and then applying them to the FE model. This was to be performed automatically as much as possible by writing preprocessing scripts that can process the data. The forces would act in exactly the same way, independent of which auxiliary components are fastened to the engine and which are not, as long as the mass or stiffness is not changed too much. Thus an auxiliary component may be meshed independently whereafter it may be introduced to the base engine model. When the model is ran it should then compute the critical engine speeds and stress amplitudes for the aforementioned component. The computations were to be performed in Abaqus, were a post-processing script should be used to analyze and present the results in an intuitive manner.

2 Theory

The theory chapter covers the theoretical background of the thesis, which lays the groundwork for understanding the subsequent chapters. The exact way in which the ROM works is described in great detail as well as the mathematical reasoning behind the method.

2.1 Resonance Order Method

When an auxiliary component is mounted on an engine, vibrations will be transferred from the cylinder block to the bracket of the component. Due to the inertia of the component, stresses that may endanger the integrity of the component will be the result. If the auxiliary component has an eigenfrequency close to the excited frequency, resonance will occur which may lead to large stress amplitudes. Thus it is assumed that this is when the components will be the most vulnerable to fatigue.

The next logical step is then to compute at which engine speeds the resonance frequencies for each auxiliary component will be present. The problem that arises is computational speed. Full dynamic simulations of the engine is quite time consuming as the dynamical simulations require up to ten engine cycles to stabilize as shown by Ola Larsson [Lar04]. Thus it will be desirable to perform full dynamic simulations at as few engine speeds as possible in order to minimize simulation time. The ROM method uses a sweep of engine speeds evenly distributed across the performance spectrum of the engine; typically 800 to 2300 rpm, incremented by 100 rpm. Based on the dynamic response of the engine at these fixed engine speeds and an eigenvalue analysis the engine speeds generating eigenfrequencies can be computed.

The engine is fired in a cyclic pattern with a period of two crankshaft turns as it is a four-stroke engine, firing on every other revolution. The displacement of a node, or in this case the reaction force acting on a node, can then be described as a periodic function with regard to the crankshaft rotational angle, allowing for it to be Fourier transformed, thus identifying the frequency components present in the signal. The transformation places the signal in the frequency domain.

The movements of the engine can be divided into different engine orders depending on their period compared to the crankshaft, each corresponding to a frequency by which they are represented in the frequency domain. The lowest order represents something that occur once in each combustion cycle and is denoted as "half" an order as it occurs once every two crankshaft rotations, period time $T = \frac{1}{720}[\frac{1}{deg}]$. The rotation of the crankshaft is defined as being of the first order, period time $T = \frac{1}{360}[\frac{1}{deg}]$, and everything else is then related to that motion. Naturally there are movements in the engine that do not occur at the rather narrowly defined engine orders, but these are excluded as experience has shown that it is the movements occurring at the half and whole multiples

of the engine speed that contribute the most to fatigue. It is also known that there are some engine orders that are almost always the most critical for an engine, which is related to the number of cylinders.

In Figure 2.1 the first four engine orders have been plotted as slanted lines for the engine speed interval 800 to 2300 rpm. The vertical, red dashed lines, represent the amplitude of the frequency components that make up the complete signal for each engine speed. Thus there is one such dashed line for each engine order in each engine speed, representing the amplitude A of a complex load C as defined in Equation 2.1.

$$\begin{cases} C = a_n + i \cdot b_n \\ A = \sqrt{a_n^2 + b_n^2} \end{cases} \tag{2.1}$$

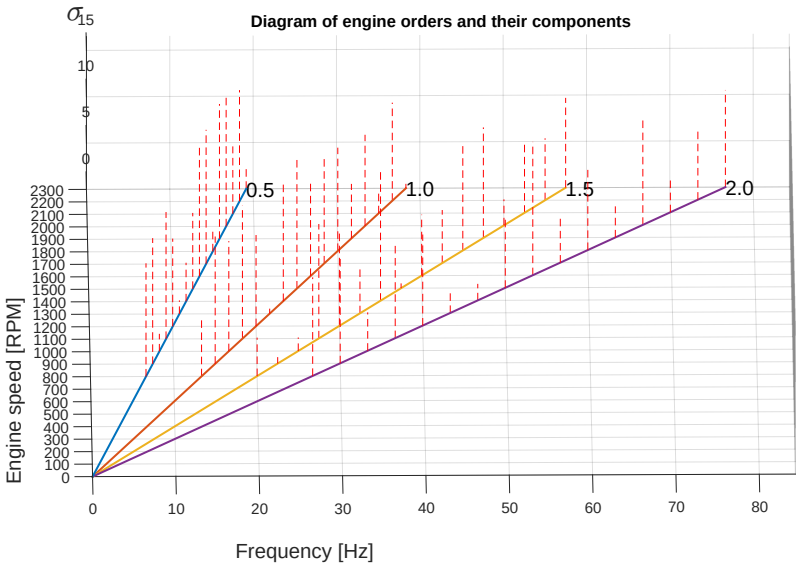


Figure 2.1: Diagram showing engine order structure.

The response at the fixed engine speeds is then used to load the component in the FE-model and compute the resulting stresses. In the earlier versions of the ROM this response has been the excitation of specific nodes where the auxiliary component to be analyzed is fastened to the engine. In the FROM developed during this thesis work, the response used is the reaction forces acting on the engine block at the crank shaft bearings. In the dynamic simulations the whole engine is loaded with combustion pressure loads in the cylinders, which causes the reaction forces at the crank shaft bearings. This allows a model of the whole engine to be used where the load is applied away from the auxiliary components, which allows for a generalisation of the method.

The next step examines the eigenfrequencies identified for the auxiliary component, identifying which engine order that yield each particular frequency inside of the designated performance spectrum. The engine speeds corresponding to excitation at an eigenfrequency can then be computed since the response of each engine order, with regard to engine speed, is a linear function in the frequency domain as represented by the slanted lines in Figure 2.2. The identification of engine speeds associated with eigenfrequencies are represented by the dashed black lines. As each engine order line can be described as a linear function $RPM_N(f_N)$, the engine speed corresponding to an eigenfrequency is computed as:

$$RPM(f_{eig}) = \frac{RPM_N(f_{N,Max}) - RPM_N(f_{N,Min})}{f_{N,Max} - f_{N,Min}} \cdot f_{eig} \ [rpm] \tag{2.2}$$

$$RPM_N, RPM(f_{eig}) \in [800, 2300] \ [rpm] \tag{2.3}$$

$$f_N \in [\frac{800}{60} \cdot N, \frac{2300}{60} \cdot N] \ [Hz] \tag{2.4}$$

The response at each new engine speed, where the dashed black lines cross the slanted lines inside of the

analyzed spectrum, can be found by means of interpolation between the two closest responses computed during the first dynamic simulation. These are represented by the dashed vertical lines in Figure 2.2 for each engine order.

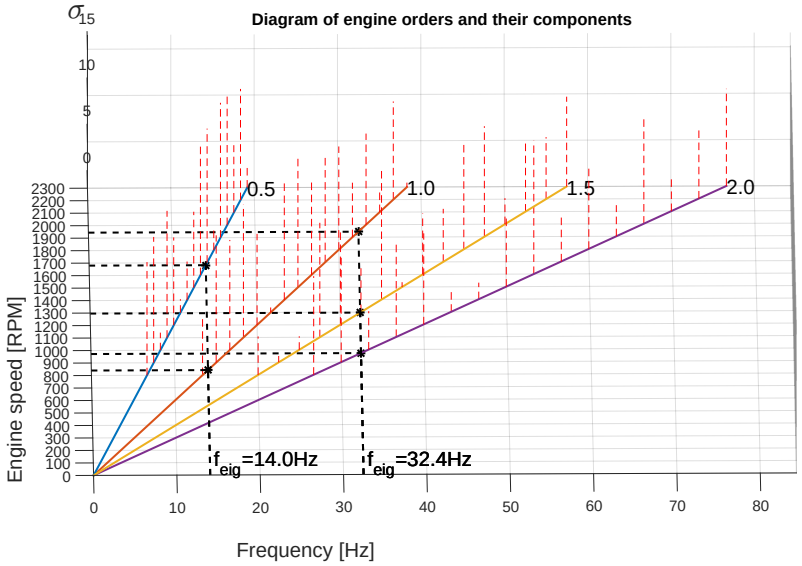


Figure 2.2: Waterfall diagram with eigenfrequencies.

In Figure 2.3 one slanted engine order line has been singled out and the interpolation path, blue dashed line, has been plotted between the two closest load points. The intersection path with the eigenfrequency causing the new engine speed and the corresponding load has been marked with a black dashed line. The interpolation method is described in detail in subsection 2.1.2.

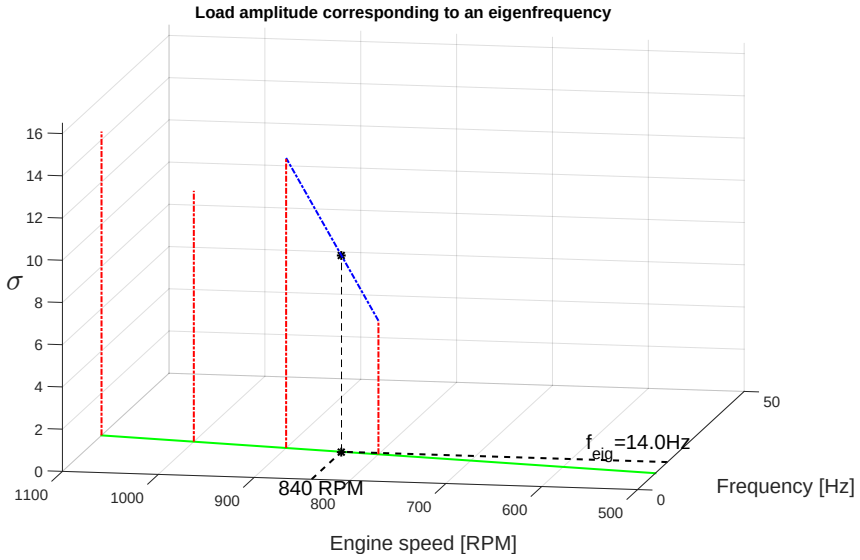


Figure 2.3: Interpolation between two data points inside an engine order line.

During the FE-analysis stresses are computed for each engine order at each engine speed, stresses that then can be assembled into a total stress for each engine speed by carrying out an inverse Fourier transform, transforming the data from the frequency domain back into the time domain. Thus the total response for each engine speed is computed using a complex component from each engine order to form a complete Fourier series for each engine speed. In relation to Figure 2.1 and Figure 2.2 this is roughly equal to a horizontal summation of the dashed amplitude lines. The procedure is described in detail in subsection 2.1.1.

2.1.1 Fourier Analysis

The forces extracted from Excite have been transformed into the frequency domain by employing a Discrete Fourier Transform to compute the Fourier constants a_n and b_n. An FFT algorithm is utilized with a Hanning window, but the exact transform used by Excite is unknown. The forces are extracted as complex numbers for each loaded degree of freedom, where the real part of the force represents the first Fourier constant a_n and the imaginary part represents the second constant b_n. The DFT is formulated as:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot \left(\cos\left(-2\pi k \frac{n}{N}\right) + i \cdot \sin\left(-2\pi k \frac{n}{N}\right)\right) \tag{2.5}$$

Where:

N	=	Number of time samples [-]
n	=	Current sample considered [-]
k	=	Current frequency considered [Hz]
X_k	=	Amount of frequency k in the signal (complex number)

The user of the ROM does not interact with the Fourier series directly at this point. Everything is carried out by the Excite software and the user only get a complex number for each load. However, as the complex forces load the engine during a Steady State Dynamic Analysis they give rise to stresses, complex stresses that also represent Fourier constants. The next step is therefore to transform the stresses back into the time domain, which is carried out by computing a finite Fourier series. Each pair of terms in the Fourier Series depends on a frequency, which in this application corresponds to the engine order frequencies as:

$$f = \frac{RPM}{60} \cdot eOrder \ [Hz] \tag{2.6}$$

$$\Omega = 2\pi \cdot f \ \left[\frac{rad}{s}\right] \tag{2.7}$$

$$f = \frac{1}{T} \tag{2.8}$$

$$\Omega = \frac{2\pi}{T} \tag{2.9}$$

The complete Fourier series is formulated as:

$$s(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \cos\left(\Omega n t\right) + b_n \sin\left(\Omega n t\right)\right] \tag{2.10}$$

$$a_n = \frac{2}{T} \cdot \int_{t_0}^{t_0+T} \left[s(t) \cos\left(\frac{2\pi n t}{T}\right)\right] dt \tag{2.11}$$

$$b_n = \frac{2}{T} \cdot \int_{t_0}^{t_0+T} \left[s(t) \sin\left(\frac{2\pi n t}{T}\right)\right] dt \tag{2.12}$$

Substitute Ω in Equation 2.10 in accordance with Equation 2.9 $\Rightarrow$

$$s(t) = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left[a_n \cos\left(\frac{2\pi n t}{T}\right) + b_n \sin\left(\frac{2\pi n t}{T}\right)\right] \tag{2.13}$$

In this master thesis a finite sum of terms has been used, more specifically the set of engine orders used in the ROM. Thus each engine order at each engine speed, spaced from 0.5 to 12 with an increment of 0.5 will correspond to a pair of Fourier constants a_n and b_n. The series used will therefor be:

$$s(t) = \frac{a_0}{2} + \sum_{n=1}^{N} \left[a_n \cos\left(\frac{2\pi n t}{T}\right) + b_n \sin\left(\frac{2\pi n t}{T}\right)\right] \tag{2.14}$$

The period time T represents two revolutions of the crank shaft, the definition of engine order 1. The time t can thus be expressed as dependant on crank shaft angle and period time as:

$$t = \frac{\phi}{720} \cdot T \tag{2.15}$$

The Fourier series will be computed for each stress component in each element for each engine speed at each crank shaft angle (time). Thus the final Fourier series [Lar04] used for computing the stresses for each element is formulated as:

$$\sigma_i^{rpm}(\phi) = \frac{a_{i0}^{rpm}}{2} + \sum_{n=1}^{N} \left[a_{in}^{rpm} \cos\left(\frac{2\pi n\phi}{720}\right) + b_{in}^{rpm} \sin\left(\frac{2\pi n\phi}{720}\right) \right] \tag{2.16}$$

Where:

$\sigma_i^{rpm}(\phi)$	=	Stress component i for engine speed rpm at crank shaft angle ϕ. [Pa]
ϕ	=	Current crank shaft angle considered. [deg]
n	=	Current sample considered, engine order. [-]
N	=	Total number of engine orders used in ROM. [-]
a_{in}^{rpm}	=	Real part of complex stress component i for engine order n in engine speed rpm.
b_{in}^{rpm}	=	Imaginary part of complex stress component i for engine order n in engine speed rpm.

This final Fourier series, Equation 2.16, is the one used in the ROM to transform the stresses back to the time domain during the post-processing. Once this is done the stresses in each element during two crank shaft turns are available for each engine speed, both the fixed ones of the original performance spectrum and those found by the eigenfrequency analysis. For each engine speed the amplitude stress in each element is computed as:

$$\sigma_a = \frac{\sigma_{1,max} - \sigma_{2,min}}{2} \tag{2.17}$$

The amplitude stresses, σ_a, are then compared to each other and the highest one is saved for each element along with what engine speed that caused it to occur. The information is written to the Abaqus Output Database File (ODB), as a new analysis step. The ODB file can be opened and the results viewed in a commercial post-processing program. During the thesis Abaqus Viewer and HyperView have been used.

2.1.2 Interpolation method

The loads are defined as loads curves in Abaqus, where the forces extracted from Excite act as data points defining the curve. Interpolation is automatically performed between the two proximate data points when a load is requested for an engine speed not contained in the initial sweep of fixed engine speeds. This is what will happen when an engine speed connected to an eigenfrequency has been identified as described in section 2.1. However, the interpolation method employed by Abaqus does not account for the behaviour in a correct way as it interpolates the real and imaginary parts of the data points respectively as shown in Figure 2.4. This interpolation method holds the potential to wreck the load curve if the data points are too widely spaced with regard to phase angle.

Since the loads are complex each one consists of a real and an imaginary part, $z = x + iy$. If linear interpolation is employed between the real and imaginary parts respectively, the interpolation path between two data points will be that of the blue line in Figure 2.4. The green and the red line represents the vectors going from origo to two different complex numbers, representing two consecutive data points on a load curve.

It is possible to perform the interpolation in a different manner where the two data points can be rewritten as absolute value and argument, or amplitude and phase, of the complex number z as shown in Equation 2.19.

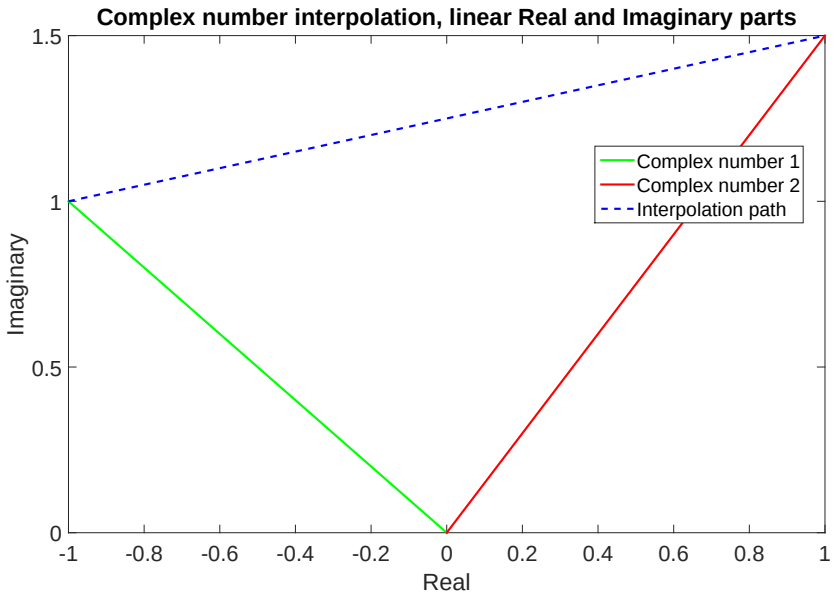


Figure 2.4: Interpolation of real and imaginary part.

$$r = |z| = \sqrt{x^2 + y^2} \tag{2.18}$$

$$\phi = \arg(z) = \begin{cases} \arctan(\frac{y}{x}) & if\ x > 0 \\ \arctan(\frac{y}{x}) + \pi & if\ x < 0\ and\ y >= 0 \\ \arctan(\frac{y}{x}) - \pi & if\ x < 0\ and\ y < 0 \\ \frac{\pi}{2} & if\ x = 0\ and\ y > 0 \\ -\frac{\pi}{2} & if\ x = 0\ and\ y < 0 \\ indeterminate & if\ x = 0\ and\ y = 0 \end{cases} \tag{2.19}$$

This is equivalent to writing the complex number on its polar form, where the amplitude and the phase of the two data points can be linearly interpolated and then returned to complex form as:

$$\begin{aligned} z &= r \cdot (\cos\phi + i \cdot \sin\phi) \\ Re(z) &= r \cdot \cos(\phi) \\ Im(z) &= r \cdot \sin(\phi) \end{aligned} \tag{2.20}$$

The interpolation path of the complex load between the data points are then as shown in Figure 2.5, a parabolic behavior, believed to represent the physical phenomena better than the interpolation between real and imaginary parts.

The reason to use the second interpolation method is that if interpolation of real and imaginary parts is used the amplitude sometimes dips in a non physical manner between the known data points on the load curve. Interpolation of amplitude and phase on the other hand will result in a linear transition between the data points as seen in Figure 2.6. The parabolic behaviour displayed in Figure 2.5 is thus only present when plotting the data in the complex plane, when examining the amplitude it displays a linear behaviour as desired.

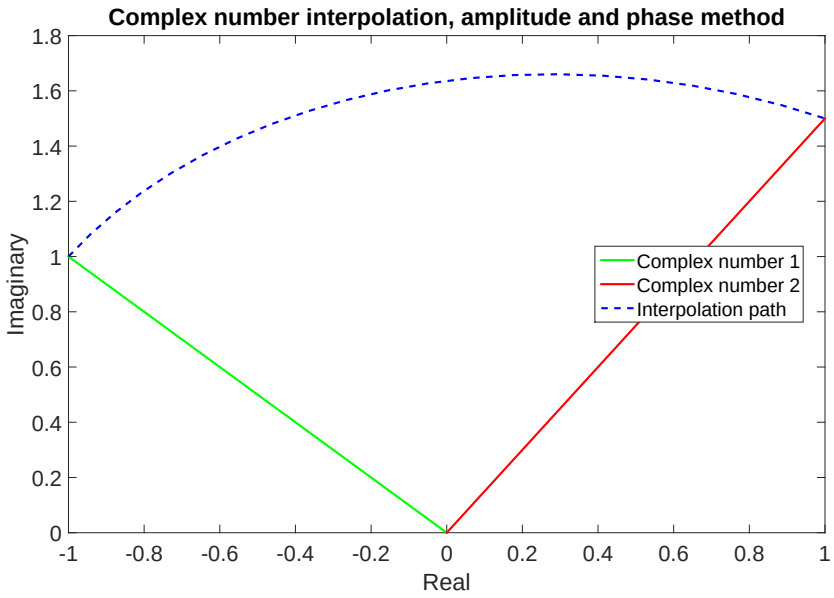


Figure 2.5: Interpolation of amplitude and phase.

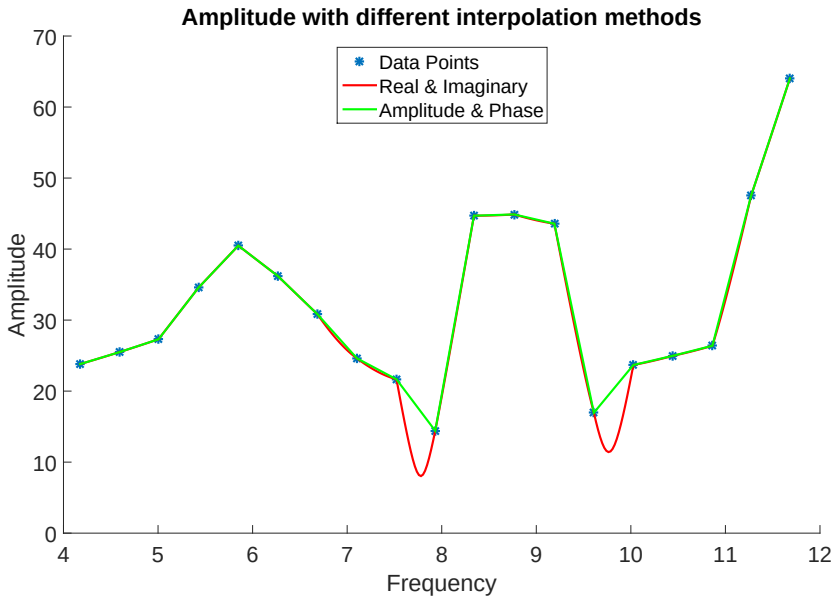


Figure 2.6: Amplitude comparison between interpolation methods.

Before performing the preprocessing interpolation the data points are located only at the fixed engine speeds. This means that when a new engine speed is found the corresponding load is found by interpolation between the two closest known data points, and there is a risk that it will be located at one of the dips shown in Figure 2.6. This occurs as Abaqus utilizes the interpolation method between real and imaginary parts. Correct amplitude is assured by increasing the resolution of the load curve using the amplitude and phase interpolation method during the preprocessing. Thus the load curve that is submitted to Abaqus has its data points much closer together and the dips in amplitude never occur.

2.1.3 Fatigue life computation

The amplitude-phase interpolation method is also employed in the post-processing to compute the stress of each engine order corresponding to engine speeds found by the eigenfrequency analysis. For each new engine speed the stress is only known for one engine order, where the dashed black line cross the slanted engine order line in Figure 2.2. Thus the other components, one for each engine order, need to be interpolated before the Fourier series can be computed, and the stress signal transformed back into the time domain.

3 Method

The methodology applied during the thesis project was to first perform a literature study in order to fully grasp how the ROM functions and to get an understanding regarding what has been done in the field previously. This was the main focus during the first two weeks of the thesis work. The literature study consisted of the reports by [Lar04], [Jön05], [Dan08], [Dae09], [Reu12]. During the same time introductions to the commercial software used at NMBS was also conducted as those were the tools to be used during the thesis work. The software in question were:

- **Abaqus** - The primary solver used by NMBS. Also the solver that is used during this thesis project.
- **Excite** - A Multi Body Dynamics (MBD) software used by NMDD to compute the dynamic behaviour of the engine. From this model the reaction forces are extracted. The Excite model is based around an FE-model on NASTRAN format.
- **HyperMesh** - A software for preprocessing.
- **HyperView** - A software for post-processing.

The FROM was developed using a L6 engine as its test object. When the verification was carried out a V8 engine was used in order to ensure that the method was general enough to handle different engines.

3.1 Preprocessing

Following the introduction the actual work on the project began, creating the preprocessing routines that would allow a user to setup a new engine model for ROM analysis. The first step was to extract the necessary data from the dynamic simulations performed in Excite. This was done by studying the structure of the input and output files as well as practical methods for extracting the relevant data from text files. The settings of Excite were examined in order to find the setup that would output the force load data in the frequency domain and on Abaqus format.

Also extracted from the Excite model was the FE-model of the engine, modeled in NASTRAN. This model was to be converted to Abaqus format, where the extracted force loads would be applied and the FE-analysis ran. It was decided that Abaqus would be used as the solver as it had previously been shown to yield far smaller amounts of data [Reu12], as well as being well known by the intended group of users. The smaller amounts of data generated are highly beneficial as the FROM encompass the whole engine.

The actual conversion of an FE-model from NASTRAN to Abaqus format is a bit tricky as the tools available introduce some interesting problems that the user has to mend by hand before running the model. Two tools for conversion were investigated and used during the thesis, one available through HyperMesh and one supplied by Abaqus. The conversion tool available directly from Abaqus seems to be the best one. It too produce some issues as the resulting model contains functions that are not supported by HyperMesh, the main preprocessing tool used at NMBS, but on the whole the model conversion is successful. Converting the model directly in HyperMesh is however not recommended; during the project the conversions carried out as such contained a wide array of issues where some element types were converted in a manner that broke the model.

Once the data had been extracted correctly the preprocessing scripts were written with the purpose of rearranging the loads from engine speed to engine order format and then defining the computation steps for

Abaqus. Some additional features were included in the preprocessing, such as the option to exclude nodes from the analysis with the intended use of allowing the user to define boundary conditions where needed. All of the preprocessing scripts are written in MATLAB.

The preprocessing defines a model which can be run, computing complex stresses corresponding to the complex loads in the frequency domain. For the results to make sense to the engineer using the method, the stress need to be transformed back into the time domain, and the main results identified. This is carried out during the post-processing, where the results are once again ordered in engine speed format, identifying an stress amplitude for each element and engine speed.

3.2 Post-processing

The post-processing script was not written during the thesis work since the script hereditary to the previous ROM for Abaqus [Reu12] already did all the things needed. Even if the FROM developed during this thesis is widely different from the old ROM, the results are the same, allowing for the same post-processing script to be run. It was decided that it would be unnecessary work to develop a new script, doing the exact same things. As Abaqus utilizes Python as a scripting language within itself, as well as providing functions for accessing output database files, the post-processing script is written in Python.

Even though the post-processing script was not created during the thesis project, it was edited to allow additional functionality and to upgrade the performance. The post-processing script was sped up by replacing the part of the program that assemble the time signal of the stress, performing the inverse Fourier transform. In the version developed in [Reu12] this is done with nested loops, which was replaced by vector operations. This increased the performance speed by approximately 250% which make a difference when large components are analyzed, as the calculation time is directly dependant on the amount of elements analyzed.

3.3 Conversion verification

When the model was converted from NASTRAN to Abaqus format there was need for verification, in order to ensure that the conversion had been successful. This was extra important as the conversion tools used was unreliable to some degree. The verification was performed by comparing the eigenvalues and eigenmodes of the two models. Some small difference were found between the models where the NASTRAN model found more eigenvalues within a given interval, thus comparing for example mode 28 of the two models would suggest a faulty conversion, while mode 27 would be identical to mode 28. The matching algorithm simply consisted of stepping through the eigenfrequencies found by the Abaqus simulation and then comparing them one by one to all the NASTRAN frequencies. The best matching frequency in the NASTRAN model was extracted and thus a new array was constructed with the same length as the number of Abaqus eigenfrequencies found. The comparison was performed with a MATLAB script.

The eigenmode shapes were compared between the converted and non-converted model using HyperView using visual inspection. The modes switching places mentioned previously could once again be seen during this comparison.

3.4 Result verification

The developed method was evaluated by examining an auxiliary component with the FROM method and then comparing the result of that to some physical testing. The physical testing consisted of an engine test where the complete engine was run in a test cell. Sensors were placed on and around the component and the engine was ran in a performance spectrum from 600 to 2400 rpm, incremented by 10 rpm per second.

By inspecting the response in the sensors placed on the auxiliary component it is possible to see which engine speeds that cause vibrations with the largest amplitude. This should approximately correspond to the largest stress amplitude, which is what is calculated by the FROM. The results of the physical testing should therefore

be able to indicate the critical engine speeds, which can be compared to the critical engine speeds computed using the FROM.

3.5 Model sensitivity

It is important to investigate the impact of the parameters used in the FROM. There are two parameters that the user of the method may designate that will determine the accuracy of the method; the number of engine orders considered for each engine speed and the number of engine speeds that will be used in the initial sweep of fixed engine speeds. Together these two criteria determine the amount of data points that will be used and therefor the accuracy and solution time.

3.5.1 Number of engine orders considered

The old ROM has been using a set of engine orders from 0.5 to 12 in the analysis, but it was not really motivated. It has been suggested that the use of half order steps are best on results of experiments and experience, showing that it is those multiples of the engine speed that are the most dangerous from a fatigue point of view. It is known that the higher frequencies will have a lower load amplitude in a Fourier series, thus there will be an upper limit where additional engine orders will not affect the stress computed.

For the new Force ROM the significance of the number of engine orders present in the analysis has been evaluated. It is desirable to use as few engine orders as possible in order to decrease the amount of steps in the FE computations and therefore the solution time. The investigation method used was to analyse the convergence of three different parts of the result in relation to the number of engine orders used in the post-processing. This is equal to truncating the Fourier series that assemble the total experienced by each element at each engine speed. The parts analyzed were:

- The maximum stress amplitude in the whole component.
- Which element that aforementioned maximum stress amplitude occurs in.
- Which engine speed that causes the maximum stress amplitude in the element.

By studying the convergence of these three parts it was assumed that the impact of the number of engine orders used could be understood. It should be noted that the number of engine orders needed for convergence likely will differ depending on what kind of engine is analyzed. In this sensitivity analysis a V6 engine has been utilized and that should be considered when examining the results; a V8 engine would likely need a few more engine orders to experience convergence.

3.5.2 Number of engine speeds considered

The old ROM traditionally used a sweep of fixed engine speeds in the dynamic simulations between 800 and 2300 rpm with an increment of 100 rpm. However, it is mentioned in the original master thesis that developed the method [Lar04], that doubling the increment between fixed engine speeds will not significantly affect the results of the method. As this has not been implemented in the method since, it had to be investigated whether this was possible or not as it may hold the potential for better results.

The results of the FROM using three sets of fixed engine speeds was used to test this. The first with the increment of 100 rpm as historically has been used, inside the interval of 500-2300 rpm. The second set was 500-2300 with an increment of 200 rpm and the last set was one incremented by 50 rpm. The same three parts of the results as were considered when examining the engine order sensitivity were used to compare the two models:

- How does the maximum stress amplitude in the whole model differ?
- Which element is subjected to the maximum stress amplitude?
- Which engine speed causes the maximum stress amplitude to occur?

Since the relation between engine speed and frequency is linear in each engine order, the rougher spectrum of fixed engine speeds should yield the same new engine speeds connected to eigenfrequencies as the more refined spectrum. The stresses computed at the non fixed engine speeds should not be the same, however, as the interpolation will be rougher. The interpolation scheme employed linearly interpolates the load amplitude to the new engine speeds, and this will be a worse estimation of the actual load the longer the distance between the fixed engine speeds. This happens as different engine orders will contain different fractions of the total amplitude signal at different engine speeds.

In his report from 2005 Ola Jönsson [Jön05] argues this exact point, that the computed stress amplitude will deviate from the exact stress but that the engine speed and critical element will be the same. He argues that the deviation is directly dependant on the increment size, as the error that occurs in the interpolation will increase with the increment. Thus a shorter increment would imply a smaller interpolation error. It should be noted that this point is made regarding the NASTRAN based ROM and not the Abaqus based one where a different interpolation method has been deployed.

The set that was incremented by 50 rpm during the convergence study had one problem as there did not exist any cylinder pressure load data for the engine at those engine speeds. An assumption was made that the cylinder pressure would differ very little with small increments in rpm, thus the same cylinder pressure was used to load the engine at 550 rpm as at 500 rpm and so forth. This introduced a small uncertainty into the test, but it is preferable to not being able to test it at all.

It should be noted that the amount of fixed engine speeds will not have as a big an impact on solution time for the new and improved FROM as in comparison to the old ROM. In the old ROM the dynamic simulations had to be rerun often in order to analyze different components on the same engine while the new FROM only need to run the dynamic simulation and preprocessing one time for each engine model. As such there is no reason to decrease the number of fixed engine speeds in order to save computation time, it will only be run once and it is unnecessary to risk the precision of the method for the sake of saving such a small amount of time.

4 Results

The results of the project were a functioning computation methodology, identifying the highest stress amplitude and what engine speed causes aforementioned stress in each element of an auxiliary component. It consists of both instructions on how to utilize commercial software as well as a set of scripts for pre- and post-processing. All of it coming together in a user friendly method. Part of the result is also a detailed, hands on, user guide, providing step by step instructions to the future users of the method.

4.1 Force based Resonance Order Method

The procedure of using the Force based Resonance Order Method, henceforth FROM, is quite simple. A complete dynamic model of the engine, modeled in Excite, are prepared and provided by the group specialised in engine dynamics, NMDD. Part of this model is a meshed FE-model of the engine and gearbox, typically modeled in NASTRAN. The FROM procedure is then:

- Dynamic response model in Excite are created by NMDD and supplied to an engineer performing strength analysis.
- A full dynamic response simulation should be run for a set of fixed engine speeds, covering the performance spectrum of the engine to be analysed. This has usually been done by NMDD when the model is supplied.
- The FE-model should be extracted from the Excite model and converted from NASTRAN to Abaqus format. This is done with the help of a commercial conversion tool, but some additional editing by hand are usually required from the engineer using the FROM as the conversion tools do not perform a good conversion every time.
- The full dynamic model should be run in Excite with a set of special settings in order to generate the nodal reaction forces in the crankshaft bearings. The forces are extracted in the frequency domain by

employing a FFT algorithm. This generate a set of files defining a load curve for each engine speed in the fixed engine speed range.

- The forces extracted are preprocessed using a set of MATLAB scripts in order to create the load steps necessary for the FROM. The preprocessor generate a set of files defining load curves for each loaded node and computation steps when the different load curves are applied to the model.
- The auxiliary component to be analysed is selected, meshed and incorporated into the FE-model. The parts of the component where stresses are to be analysed should be covered with very thin membrane elements as it is the stresses on the faces of the component that are of interest; it is on the surface that the fatigue cracks begin. Using membrane elements decreases the time needed to run the post-processing as it is directly dependent on the amount of elements in the stress analysis set.
- The FE-computations are run, performing steady state dynamic analysis for each step defined by the preprocessor. Stresses are computed and written to the membrane set defined.
- The post-processing script is run in order to assemble the stresses into a time signal once again. It also identifies maximum stress amplitude and critical engine speed for each element in the membrane stress-set.

When the method is used on a new engine model for the first time, all the steps above need to be performed as a first time setup. When additional auxiliary components, or new iterations of an auxiliary component, are analysed only the three last steps need be performed as the base engine model and the loads applied stay the same. This only holds as long as the changes made are small enough not to significantly change the mass or stiffness of the engine. Changes applied to an air filter bracket for example do not require the dynamic analysis to be rerun, but changes to the engine block would alter the dynamic response of the engine such that it would have to be treated as a completely new engine altogether.

4.1.1 Model conversion

During the course of the project it has become clear that the most time consuming part, in man hours, of using the FROM is the conversion of the NASTRAN based FE-model to Abaqus format. The reason for this is that the commercial conversion tools that exist are a bit flawed in their functionality when it comes to handling complex models with lots of unique parts. The conversion tools introduce errors in the model, errors that need to be remedied by hand which may be quite time consuming. The upside is that this step only need be performed once for each engine model, so in the long run it is a minor inconvenience.

The actual conversion from NASTRAN to Abaqus format may be performed in a variety of ways, using different commercial software. During this project two different conversion tools have been used; Abaqus *fromnastran*, available either in Abaqus CAE or invoked from the command line, and the conversion tool in HyperMesh. The best conversions were achieved by using the conversion tool from Abaqus themselves, and then by fixing any errors by hand. Due to the uncertainty in model conversion it is absolutely necessary to perform an eigenvalue and eigenmode comparison between the NASTRAN and Abaqus model. If the conversion has been successful, they should correspond very well to each other.

Some errors was encountered during the thesis work where the conversion was unsuccessful, thus resulting in a model that in the worst case did not work at all and that in the best case had different eigenvalues and modes than the original model. The most grievous errors were encountered using the HyperMesh conversion tool; In one instance beam elements used to connect different parts of the engine where converted to *HyperBeam* elements. This seemed OK at first glance, it is simply how they are modeled in HyperMesh. But when the FE-model was exported as an Abaqus input file the elements were exported without inertia, thus resulting in a model that could not run.

4.1.2 Preprocessing

The preprocessing step perform the setup of the model, extracting the reaction forces from the dynamic simulation model and then applying them in an appropriate manner onto the newly converted Abaqus FE-model. The loads are applied to the model by including the files generated by the preprocessing scripts to

the main model file. The loads are defined in these include-files in order for the model to be transparent and easily accessible for the user. The first part of the process is performed by hand as it involves interacting with commercial simulation software. The second part that formats the data, however, is automated in a set of MATLAB scripts that process the data from the dynamics simulation so that it may be applied to a FE-model.

4.1.2.1 Extracting force loads

The force load extraction is done directly through Excite, the software containing and running the dynamic model. The dynamic model is based around an FE-model of the engine on FE-format, the same model that is converted to the Abaqus FE-model later used. In the NASTRAN model a set of ASET nodes are defined, dictating where the reaction forces may be extracted. Inside Excite it is possible to create a *FE Analysis task* where it is possible to define the extraction of nodal reaction forces. Aa FFT along with a Hanning window is utilized during the extraction in order to transform the data to the frequency domain, as is required by the FROM. The transformation causes the reaction forces to be written on complex form.

Excite generates a set of files for each engine speed analysed, two files for each degree of freedom where a reaction force acts. One of the files contains the real part of the complex load and the other one contains the imaginary part. This amounts to a vast amount of files, several thousand for each engine speed. In the engine model that was used during the development of the method there were approximately 3500 loaded degrees of freedom and 16 engine speeds considered, totaling an amount of more than $112,000$ load defining files.

4.1.2.2 Processing force loads

The FROM requires load amplitude curves to be defined for each engine order, as opposed to being defined for each engine speed as is the case in the data extracted from Excite. Figure 4.1 shows a small section of the load defining file generated from Excite for one degree of freedom. The data is structured in pairs, where the leftmost value is a frequency corresponding to an engine order and the rightmost value is a load corresponding to that frequency. Figure 4.1 thus shows four such pairs of data, two on each line.

```
6.6788411e+00,     3.6055782e-01,  1.3357682e+01,     -8.1436396e-01,
3.3394207e+01,     3.6019740e+00,  4.0073048e+01,     -5.4209728e+00,
```

Figure 4.1: Force load file from excite, data structure.

The preprocessing scripts read each load file and identify the pairs that correspond to an engine order based on the frequency values. The corresponding load value is then saved into a data structure. When all the files have been read into the data structure it is printed to a new set of files which define all the load curves corresponding to a certain engine order. Practically this means that the $112,000$ original files are reduced to 24, though admittedly larger in size, files. The reduction in files is important as it both increases the readability of the data for the user and since it decreases the time needed to admit the model to the cluster for the following FE-computations. This is due to the fact that opening and copying a large amount of files is quite time consuming, especially when the files are located on a remote server and not on a local drive.

The loads are defined in Abaqus by two parts, one real and one imaginary, in the manner that is shown in Figure 4.2. The figure shows the complex load being defined for the second degree of freedom in node 100 with a weight factor of 1. The option **AMPLITUDE** tells Abaqus which load curve defines the load.

```
*CLOAD, REAL,  AMPLITUDE=Enginegearbox-100-2_R.5
100,2,1.
*CLOAD, IMAGINARY,  AMPLITUDE=Enginegearbox-100-2_I.5
100,2,1.
```

Figure 4.2: Defining the loads on Abaqus format.

A load curve is defined on the format shown in Figure 4.3, where the name corresponds to the one referenced in Figure 4.2. The load curve is defined as a table of data points, where each point consists of a frequency and

its corresponding load. These data points are the same ones as was extracted from Excite, shown in Figure 4.1, but they have been reordered so that the load is defined for an engine order. In the example a load curve for the half engine order is shown, and the first data point corresponds to the lowest engine speed used in the analysis in accordance with Equation 4.1.

$$f = \frac{rpm}{60} \cdot eOrder \ [Hz] \tag{4.1}$$

```
*AMPLITUDE, NAME=Enginegearbox-100-2_R.5, DEFINITION=TABULAR, VALUE=RELATIVE
4.175946e+00,         1.266490e+02,
4.217371e+00,         1.264593e+02,
4.258797e+00,         1.262690e+02,
```

Figure 4.3: Load curve definition after preprocessing.

As can be seen in Figure 4.3, the two following data points are quite close to the first, too close to represent the next couple of engine speeds used in the analysis. This is due to the phase and amplitude interpolation utilized in the preprocessing as described in subsection 2.1.2; The interpolation refines the load curve by creating additional data points in between the data points corresponding to the fixed engine speeds.

4.1.2.3 Defining FE computation steps

The preprocessor finishes its purpose by applying the newly defined loads to the FE-model. This is done by writing computation steps, two for each engine order. The first step defined, however, is an eigenfrequency computation, as shown in Figure 4.4, that is essential to the computations in the FROM method. As mentioned previously, excitation at an eigenfrequency is usually what is the most damaging to a auxiliary component, so it is of great importance to compute the stresses at the eigenfrequencies present within each engine order interval.

```
**************************************************
*STEP, PERTURBATION
*FREQUENCY, SIM, EIGENSOLVER=LANCZOS, NORMALIZATION = MASS
800,,,0,,,
*OUTPUT, FIELD
*ELEMENT OUTPUT, ELSET = stressSet, POSITION= CENTROIDAL
s
*NODE OUTPUT
u
*END STEP
**************************************************
```

Figure 4.4: Eigenfrequency step definition after preprocessing.

Before the eigenfrequencies are used, the model response at the fixed engine speeds are computed. This is defined in a computation step as shown in Figure 4.5, where the frequency interval $4.167 - 19.167 \ [Hz]$ depends on the lowest and highest fixed engine speed and what engine order is considered. The number 19 defines the number of fixed engine speeds that have been used inside the interval. The loads corresponding to the fixed engine speeds are extracted from the load curve defined in the *Cload_5* include file, which contains information as seen in Figure 4.2.

The response due to engine speeds giving rise to an eigenfrequency are computed by defining a computation step on the format shown in Figure 4.6. As can be seen the same frequency interval has been defined, but this time the **Interval** option has been set to eigenfrequency. This instructs Abaqus to compute the response at eigenfrequencies present in the defined interval. The number 2 instructs Abaqus to also include the end points of the interval in the analysis. As can be noted the same loads are included and used in this step as the one shown in Figure 4.5. The difference between the steps is only which points on the load curves that are used to load the model, whether they correspond to fixed or eigenfrequency caused engine speeds.

```
**************************************************
*STEP
*STEADY STATE DYNAMICS, INTERVAL = RANGE, FREQUENCY SCALE = LINEAR
4.166667e+00, 1.916667e+01, 19
*MODAL DAMPING, DEFINITION = MODE NUMBERS
1,800,0.01
*INCLUDE, INPUT =/home/aercn8/Examensarbete/exciteVer/Converted/Cload_5.txt
*OUTPUT, FIELD
*ELEMENT OUTPUT, ELSET = stressSet, POSITION= CENTROIDAL
S
*END STEP
**************************************************
```

Figure 4.5: FE computation step definition for fixed engine speeds.

```
**************************************************
*STEP
*STEADY STATE DYNAMICS, INTERVAL = EIGENFREQUENCY, FREQUENCY SCALE = LINEAR
4.166667e+00, 1.916667e+01, 2
*MODAL DAMPING, DEFINITION = MODE NUMBERS
1,800,0.01
*INCLUDE, INPUT =/home/aercn8/Examensarbete/exciteVer/Converted/Cload_5.txt
*OUTPUT, FIELD
*ELEMENT OUTPUT, ELSET =stressSet, POSITION= CENTROIDAL
S
*END STEP
**************************************************
```

Figure 4.6: FE computation step definition for engine speeds corresponding to eigenfrequencies.

When the preprocessing has been finished and the files discussed in this section thus have been created, the user should include the step definition file into their FE-model. This file then takes care of everything else, referring to the load defining files and defining every necessary computation.

4.1.3 FE computations

As previously mentioned the solver used for the FE-computations in the FROM is Abaqus, to which the model is submitted once the preprocessing has been finished. The solver is equipped with parallel processing which makes it very suitable for use on a cluster. This significantly reduces the solution time compared to using the 4 or 8 cores available on a standard desktop computer. Unfortunately the full power of using a cluster is not available when using the FROM as the computational step that computes the eigenfrequencies is not compatible with multi-host execution.

Multi-host execution is when several processors, each containing several cores, handle the computations. This means that even though a cluster is used, only 16 cores partake in the computations. If an eigenfrequency computation is not part of the model the cluster support 64 cores to be used in parallel. There are other perks of using the cluster too: The memory available is higher, allowing for larger models to be analyzed, and the user also retain access to the power of the local computer to perform other tasks simultaneously.

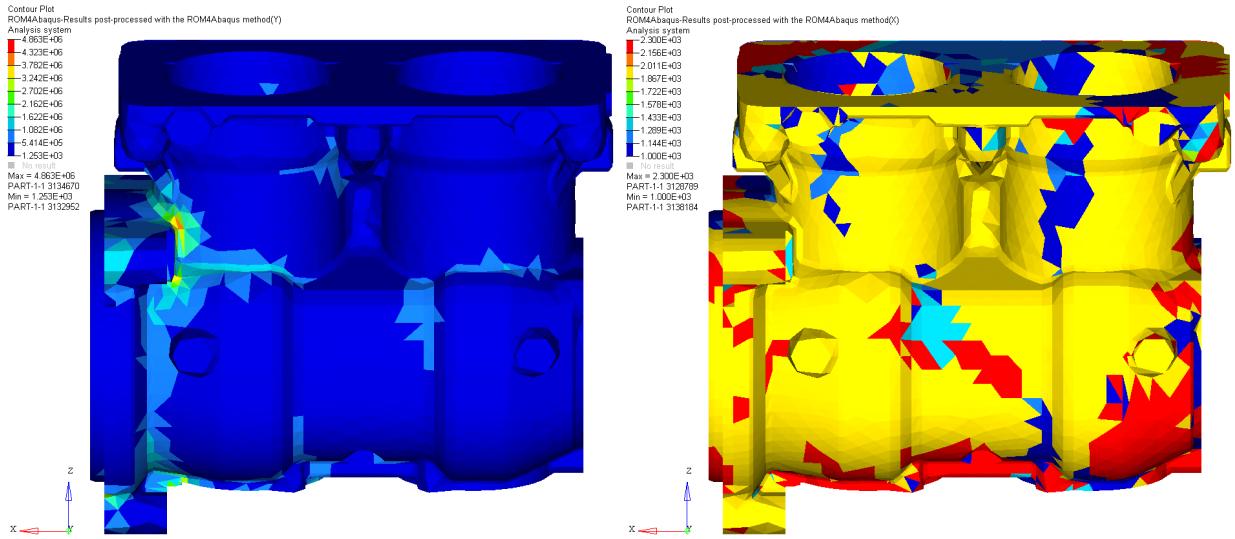
4.1.4 Post-processing

As has been mentioned earlier the analysis is set up in a way that when it is done, stresses have been computed in each element in a designated set. In each of these elements, a complex stress has been computed for each engine speed in each engine order. To recap, there are two computational steps for each engine order; one for the fixed engine speeds and one for the engine speeds corresponding to eigenfrequencies.

What is done in the post-processing is that the total stress is assembled for each element in the analysis set. This is equal to transforming the stresses from the frequency domain back into the time domain, where it is possible to more intuitively interpret the results. The transformation and assembling of the stresses is performed by using the Fourier sum shown in Equation 2.16 in subsection 2.1.1. The result will be one set of stresses for each engine speed in each element. Then the stress amplitude at each of those engine speeds are computed and the highest one is saved in each element along with the corresponding engine speed.

The performance of the post-processing calculations are greatly increased by utilizing a cluster as they are compatible with multi-host computations. This is possible as the computations of stresses for each element are independent of the calculation of stresses in the other elements, thus allowing for as much parallelization as the host can provide. The cluster that has been used during the thesis project have 12 cores available for running python scripts, making it quite fast, especially in comparison to running it on a 4 core standard computer.

When ran, the post-processing script reads the data from the ODB-file created by Abaqus during FE computations, and compare the stress amplitudes caused on each element by the different engine speeds to each other. The highest stress amplitude computed, along with the engine speed causing it, is saved in a new step in the ODB-file. The results can thereafter be viewed in a post-processor of the users choice. In Figure 4.7 the results on a auxiliary component have been plotted using HyperView.



(a) Max stress amplitude in each element, 1.25e-3 to 4.66 MPa. (b) Critical engine speed for each element, 1000 to 2300 rpm.

Figure 4.7: Post-processed results with ROM4Abaqus.

As can be seen in the left part of Figure 4.7, displaying the critical stress in each element, there is a small stress concentration at a radius on the leftmost side of the component. By then looking at the right side figure, showing the corresponding engine speeds, it can be seen that the same engine speed causes the critical stresses in that area of the component as the whole area have the same color. Actually it is one engine speed that is critical for most of the component, marked in strong yellow in the figure.

When an interesting engine speed has been identified it is sometimes desirable to examine how the auxiliary component will move when excited at that speed. To this end two more scripts are used. The first is another MATLAB script much resembling the ones used in the preprocessing as it defines a set of computational steps for Abaqus where the displacement of the component is examined at the chosen engine speed. This new step defining file is used with the FE-model where Abaqus will compute the motion. The operation is quite a bit

faster than the stress computation as it only need to consider one engine speed, and thus one load point, in each step.

On top of that there is only half as many steps performed as the steps computing the response at different eigenfrequencies have been removed. When the Abaqus calculation is finished there is also a post-processing script that once again assembles the signal, now displacement instead of stress, into the time domain. This is also quite a bit faster as only one engine speed is analyzed.

It was suggested during the project that it might be desirable to examine other physical quantities than stress and deformation. The strain and the nodal acceleration might also be of interest, thus two additional preprocessing scripts where created which defines computational steps that compute those physical quantities. The post-processing script also had to be modified accordingly. The changes made was minor as the method is still carried out in the same way; the difference is that the scripts has to call different objects during the computations.

The addition of these scripts broaden the potential use of the method as it provides a user with access to more data that might be of interest. For example one might consider the study of a gasket made of rubber, where the stresses might not be of the highest interest. Instead it might be the strain and deformation of the material that is most important.

4.2 Verification

Verification of the models' validity has been carried out continuously during the thesis project to ensure that the model is stable and that subsequently the results are trustworthy. The conversion of the FE-model from NASTRAN to Abaqus is an integral part of using the method and there are quite a few things that might go wrong there, thus the associated verification step needs to be taken each time the method is used. If the conversion is unsuccessful it does not matter if the rest of the method works perfectly.

There was also the more obvious need for verification of the method itself, ensuring that the user will be able to trust and use the results. This second verification step is a bit more complex than the first. Optimally it would be possible to verify the method against both the old method and physical testing, both of which have been done in this project.

4.2.1 Verification of model conversion

The first verification stage occurred during the conversion of the FE-model from NASTRAN to Abaqus format. As the engine model is quite large and contains many parts it was important to ensure the integrity of the model post conversion. This was done by performing an eigenvalue and eigenmode analysis of the NASTRAN and the Abaqus version of the model. It was assumed that a successful conversion would yield the same eigenvalues and mode shapes regardless of solver.

In both models the first 200 eigenfrequencies were computed using the Lanczos method, [16], and was then plotted for comparison in MATLAB. When the FROM is used not all of these modes are used, but some extra modes are computed in order to ensure the stability of the modes that are actually used. When the method is run the maximum frequency that is considered is associated with the maximum fixed engine speed and the highest engine order used. Specifically this would be 460 Hz in a normal run. As can be seen in Figure 4.8 the conversion seem to have been quite successful even though the lines diverge from one another slightly.

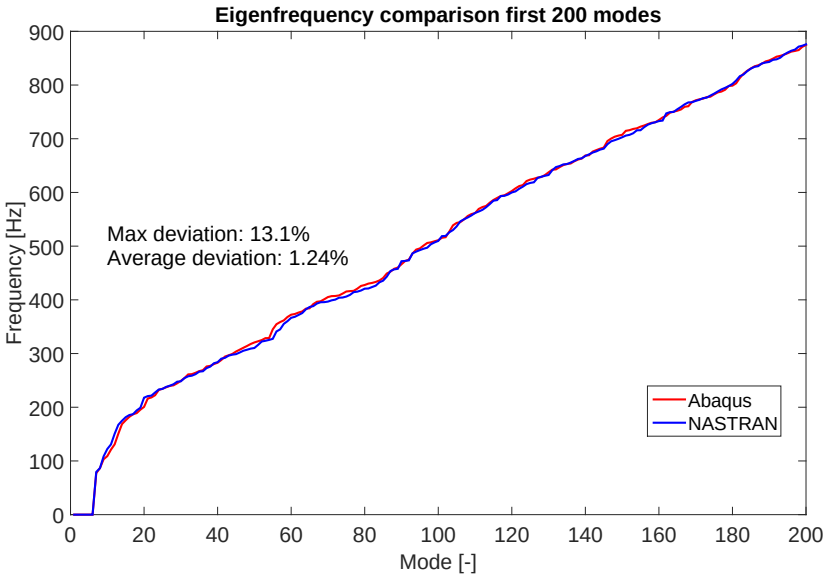


Figure 4.8: Eigenvalue comparison to validate converted model.

However, this comparison becomes somewhat skewed as the models find some different eigenvalues, distorting the order of the values in the comparison. For example the Abaqus model might find two eigenvalues very close to each other where the NASTRAN model only finds one, causing gaps to appear in the comparison when there actually is an eigenvalue matching much better, only that it appear at another mode number. When this is taken into account and the data is shifted to compensate, the match between the models become somewhat better as can be seen in Figure 4.9.

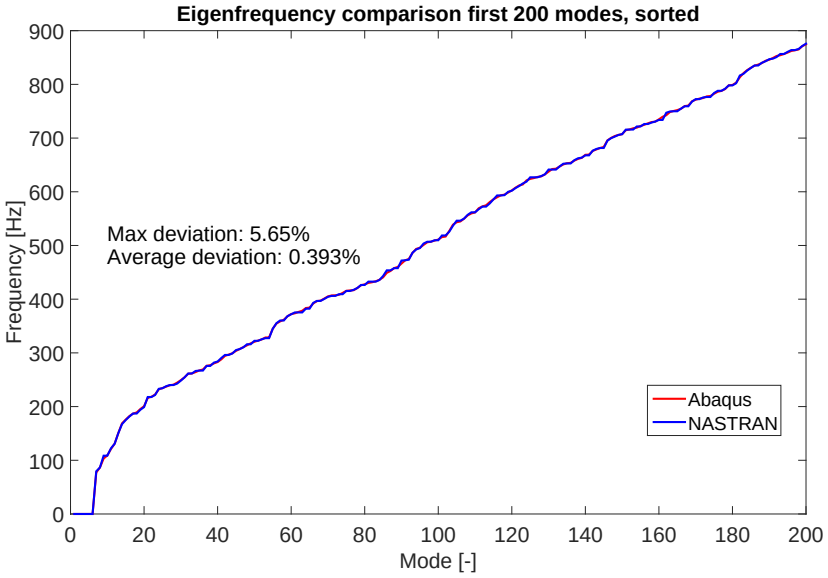


Figure 4.9: Eigenvalue comparison, closest match.

The mode shapes of the first 30 modes, including the rigid body motions, were compared by visual inspection in HyperView and it could be seen that they corresponded well between the models when the shift in mode numbers were taken into account. Thus mode 11 of the Abaqus model might be an exact match for mode 12 in the NASTRAN model. It was decided that this was a good enough correlation between the models to continue with the method.

4.2.2 FROM computation

To ensure that the FROM computes the same things as the old ROM a comparison was made, where the same auxiliary component was analysed using both methods. The component chosen for analysis was a bracket sustaining the actuator which controls the exhaust brake on a V8 engine, as highlighted in Figure 4.10. The actuator bracket was chosen as it has previously been studied and was therefor already meshed, it would also be fairly simple to carry out physical testing on it. It also had the added benefit of already being prepared with a fine mesh from the previous study, shortening the time needed to set up the model.

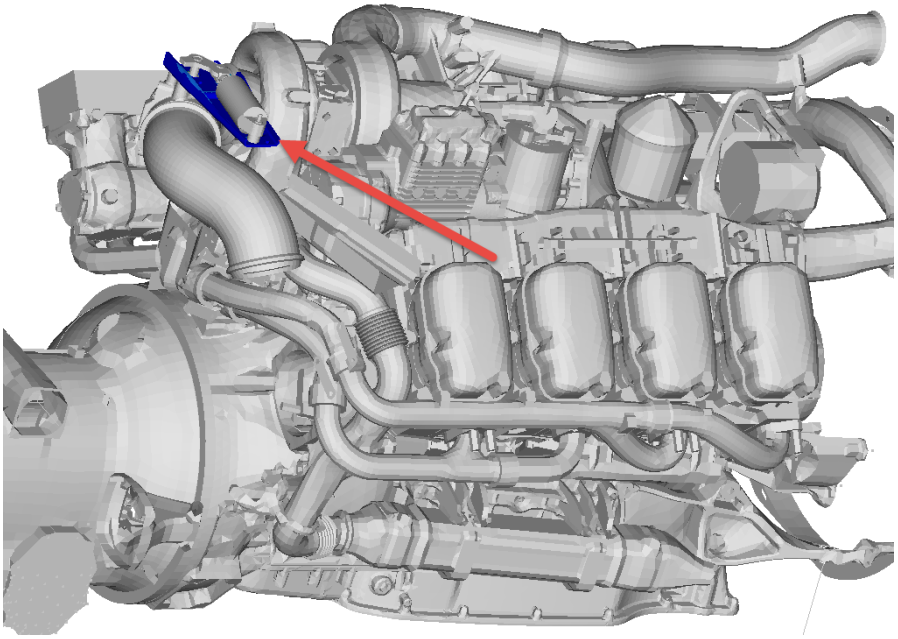
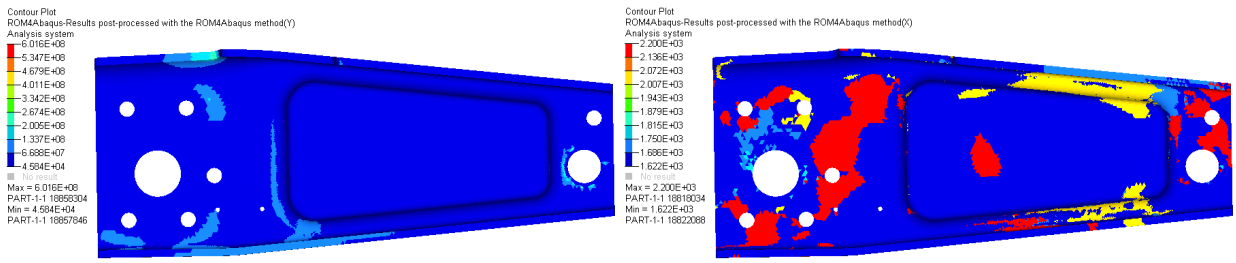


Figure 4.10: Location of the examined actuator bracket on the engine.

The FROM FE-model contained the bracket when exported from Excite, however, it was very roughly meshed, only supplying the correct stiffness and inertia to the whole model. So in order to be able to analyze the bracket with regard to stress, a much finer mesh was required. The initial, rough, mesh was removed and a finely meshed version from a previous project was attached in its place whereafter the FROM was used to analyze it.

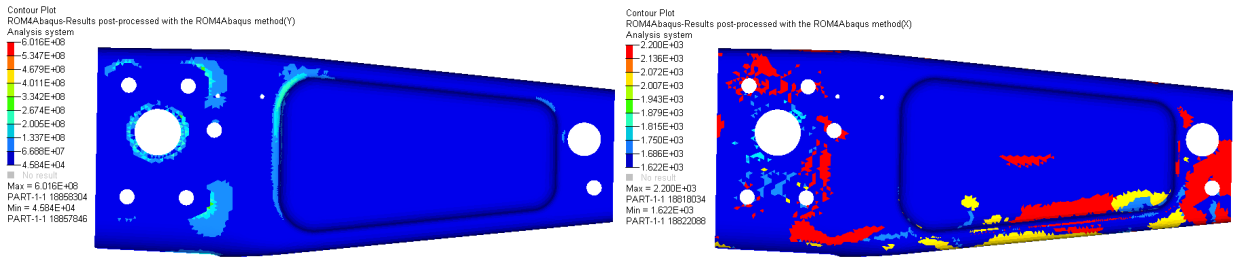
The critical stresses experienced by the elements on top of the bracket is displayed in Figure 4.11 where the elements have been colored individually in correspondence with the highest stress amplitude experienced. Theoretically the critical stress could occur at a different engine speed in each element. However, this is not the case as can be seen to the right in Figure 4.11, where the elements have been colored in accordance with what engine speed causes the critical stress for that element.



(a) Max stress amplitude in each element,45.8e-3 to 601 MPa. (b) Critical engine speed for each element, 1622 to 2200 rpm.

Figure 4.11: Critical stress distribution in actuator bracket top using the FROM.

As can be seen in Figure 4.11 the critical engine speed varies in fields across the bracket, indicating that different parts of the component will be weaker against the vibrations caused by specific engine speeds. Figure 4.12 shows the bottom of the actuator bracket where it can be seen that the largest critical stress occur at the corner of the countersink in the bracket and on the edges where the bracket is fastened to the exhaust pipe. The engine speed distribution should be compared to the stress distribution as shown in Figure 4.12 where it is possible to identify the areas experiencing high stresses in the left side figure and which engine speed caused said stresses in the right side figure. Thus it is possible to determine which engine speed or speeds that should be further investigated.



(a) Max stress amplitude in each element, 45.8e-3 to 301 MPa. (b) Critical engine speed for each element, 1622 to 2200 rpm.

Figure 4.12: Critical stress distribution in actuator bracket bottom using the FROM.

4.2.3 Physical testing

Sensor placement during the physical testing is displayed in Figure 4.13 where four sensors have been marked with arrows indicating the directions of measurement in each point. However, the result of all four sensors are not used in the verification analysis. Only the three located directly on the bracket and on the actuator cylinder are considered.

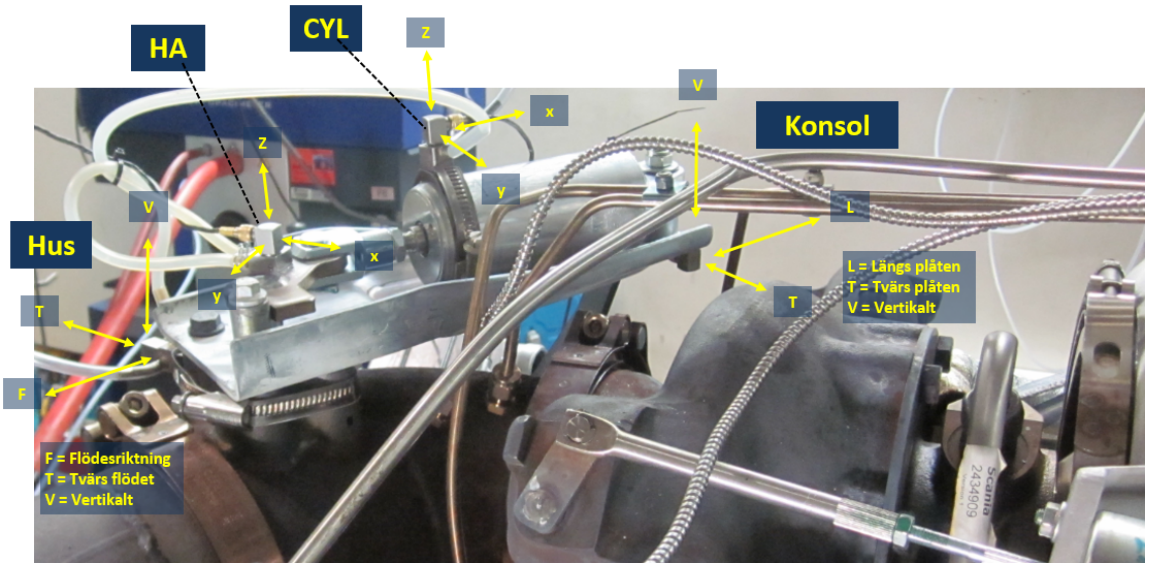


Figure 4.13: Sensor placement at actuator bracket during testing.

The tests can regrettably not compute an exact stress amplitude and corresponding engine speed in the manner of which the ROM does. What is done is that the excitation of the sensors are recorded during the test, thus providing an indication as to what engine speed might be the most dangerous. It is however important to note that it is only the excitation, or deformation, that is recorded and not the stress in the component. The critical engine speed that was computed using the FROM has been plotted in the figures as a dashed line to mark the excitation at that precise engine speed.

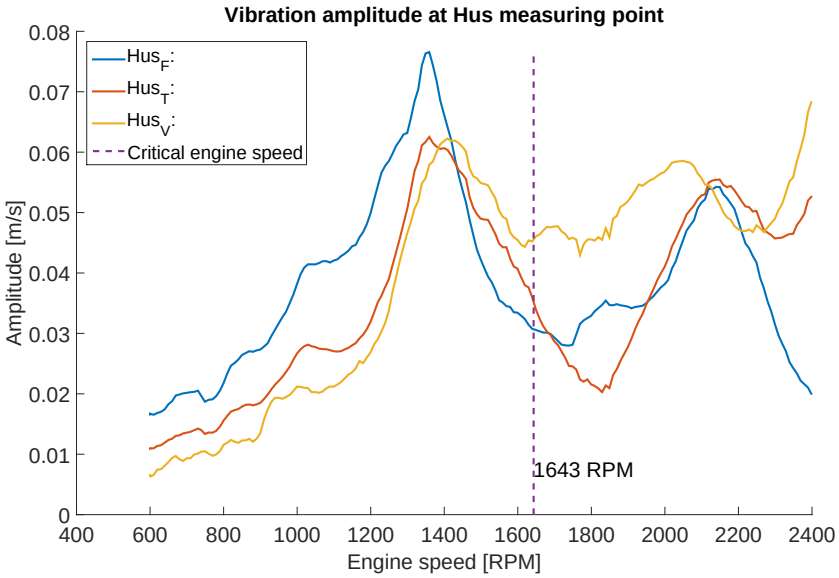


Figure 4.14: Excitation measured by sensor located at exhaust pipe.

Figure 4.14 shows the excitation measured by the sensor placed at the exhaust pipe. This can be viewed as a measure of the excitation that is induced into the actuator bracket whereas Figure 4.15 display the excitation at the outer edge of the bracket. The response shown in Figure 4.15 might be a more accurate measure of the vibrations acting on the bracket. It can be seen that the critical engine speed computed using the FROM hits one of the peaks in excitation experienced by the bracket in the transverse direction. However, it should be noted that the vibration in all three directions peaks at a lower engine speed, indicating that the FROM model might be a little stiff.

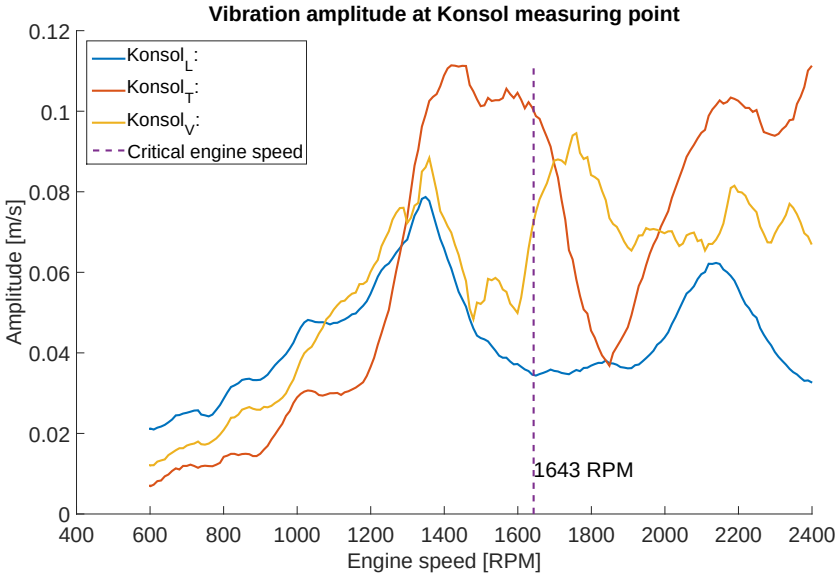


Figure 4.15: Excitation measured by sensor located at outermost edge of actuator bracket.

Furthermore, the excitation at the cylinder located on the bracket is plotted in Figure 4.16. This sensor is not placed directly on the bracket, but the results might still be interesting as the position of the sensor is quite close to the countersink corner on the bracket where the highest stresses were observed, shown in Figure 4.12. As can be seen the critical engine speed correlate fairly well with the peaks in all three directional excitations. Once again, however, it can be seen that there is a common peak at a lower interval, 1300-1500 rpm.

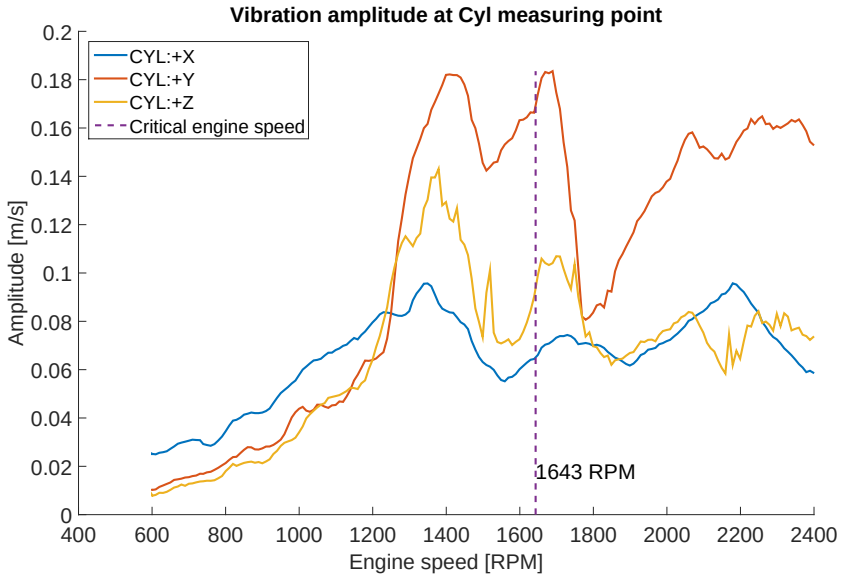


Figure 4.16: Excitation measured by sensor located at actuator cylinder.

4.3 Method sensitivity to engine orders used

As the old ROM has utilized engine orders 0.5 to 12 in its computations but has not motivated why this interval was used, a sensitivity analysis was performed. The convergence of three important measurements were investigated to establish how many engine orders that need to be considered to ensure validity of the method. The number of engine orders used corresponds to how many frequencies are considered when the Fourier sum is computed. Decreasing the number of engine orders are equal to truncating Equation 2.16.

The maximum stress amplitude experienced by the component can be seen in Figure 4.17. It suggests that the maximum stress experienced by the component converges somewhere around seven engine orders.

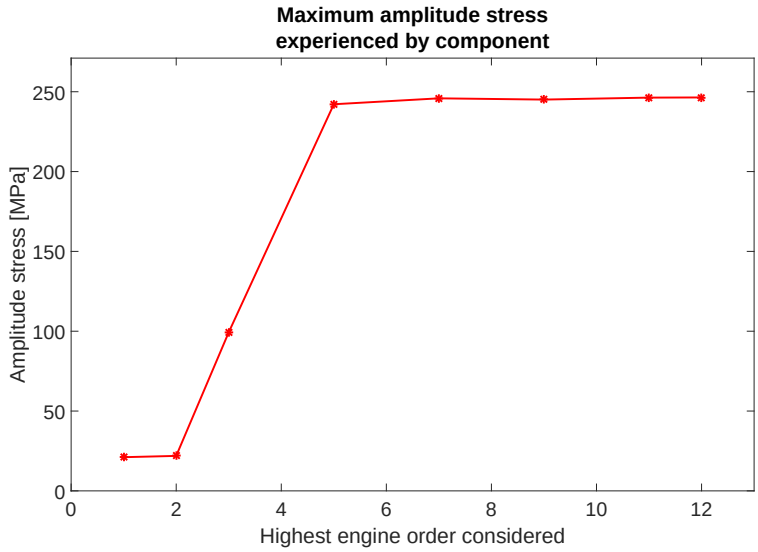


Figure 4.17: Maximum stress amplitude experienced depending on number of engine orders considered.

The convergence of the results can be further confirmed by viewing Figure 4.18 where the element number of the element that is experiencing the above mentioned maximum stress amplitude is examined. This is

equivalent to identifying the most critical element on the examined component. As can be seen the convergence to the critical element occur as early as when using three to four engine orders.

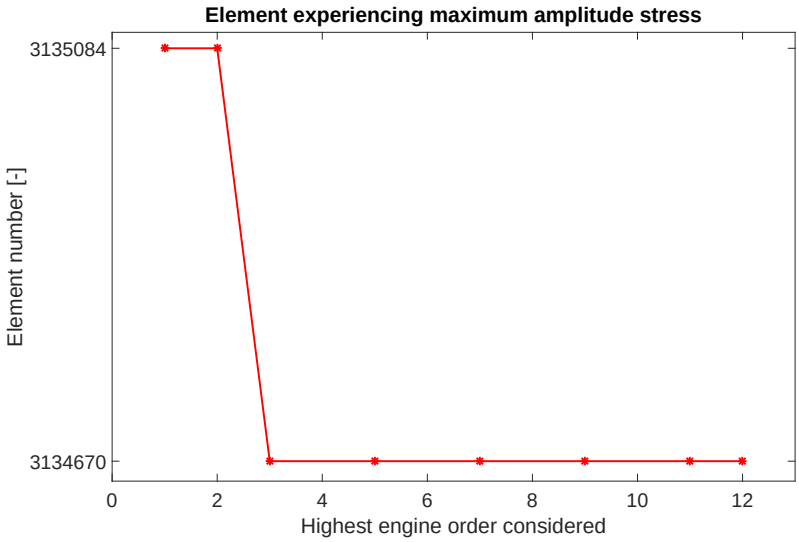


Figure 4.18: Element experiencing maximum stress amplitude depending on number of engine orders considered.

The critical engine speed that causes the maximal stress amplitude seems to converge somewhere around six to seven engine orders as seen in Figure 4.19 where an engine speed caused by an eigenfrequency arises as the critical one, exactly as might be expected. The engine speeds shown corresponds to the converged maximum stress amplitude in Figure 4.17 in the converged to element from Figure 4.18.

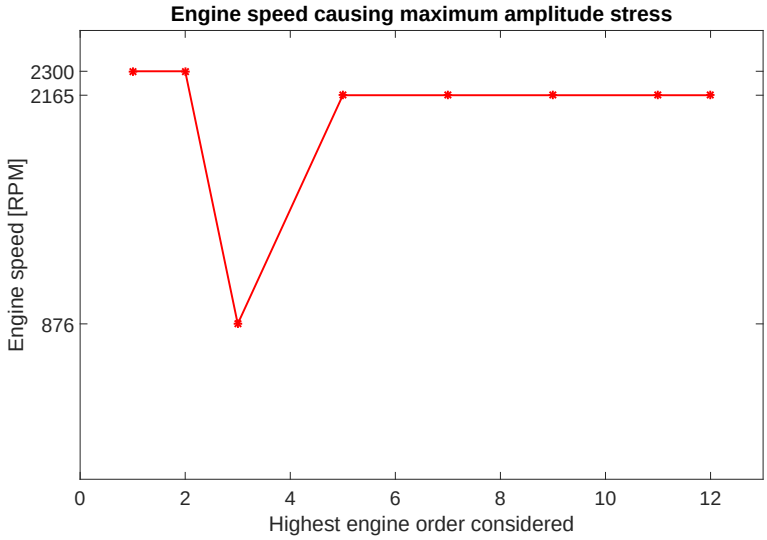


Figure 4.19: Engine speed causing maximum stress amplitude depending on number of engine orders considered.

The critical engine speed distribution can further be seen in Figure 4.20. It can be seen that using 12 engine orders instead of 7 does not make much of a difference while the difference between using 5 or 7 orders are significant. In the figure the elements has been colored depending on what engine speed is the most critical for each element.

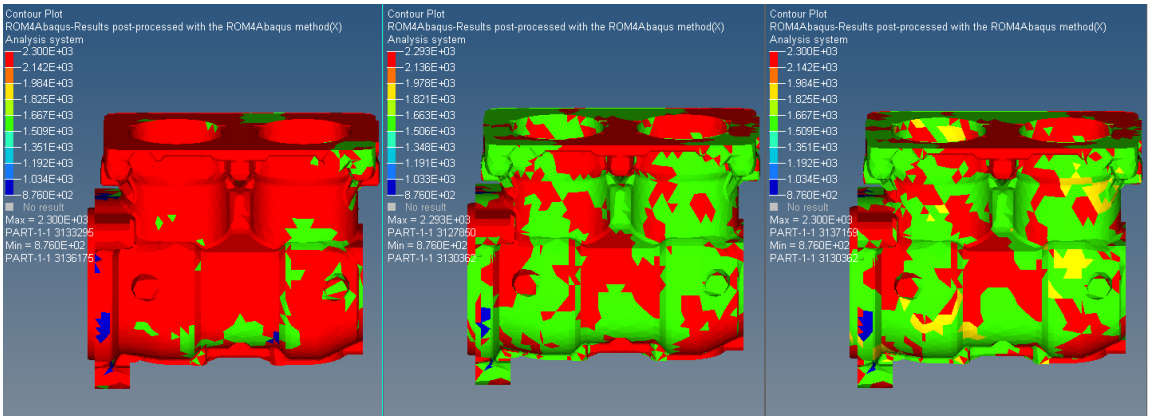
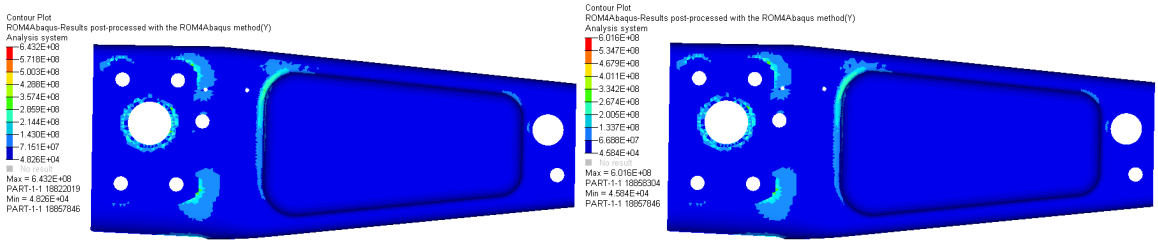


Figure 4.20: Critical engine speed distribution. Engine orders up to 5 to the left,up to 7 in the middle, and up to 12 on the right, 876 to 2300 rpm.

4.3.1 Method sensitivity to number of fixed engine speeds

The most critical element, and the critical stress caused by the critical engine speed in that element was investigated for the same component using three different sets of fixed engine speeds as described in subsection 3.5.2. The two sets were incremented by 50, 100 and 200 rpm respectively, making the third set twice as sparse as the normal one and the first set twice as dense. In Figure 4.21 the stress distribution on the bottom of the bracket is displayed, and as can be seen the results hardly differ at all. The amplitude of the critical stresses actually differs a bit between the two models, for each element the stress differs somewhere between five and seven percent where the sparse set of fixed engine speeds yield higher stresses. The maximum critical stress occurs at the inner radius of one of the holes in the bracket for both sets, in a single element where the surrounding elements does not experience a stress similar to that. It is quite likely that this is due to the element being located at the border of a boundary condition, thus experiencing a non realistic load.



(a) Fixed engine speeds incremented by 200 rpm, 46.2e-3 to 643 MPa.

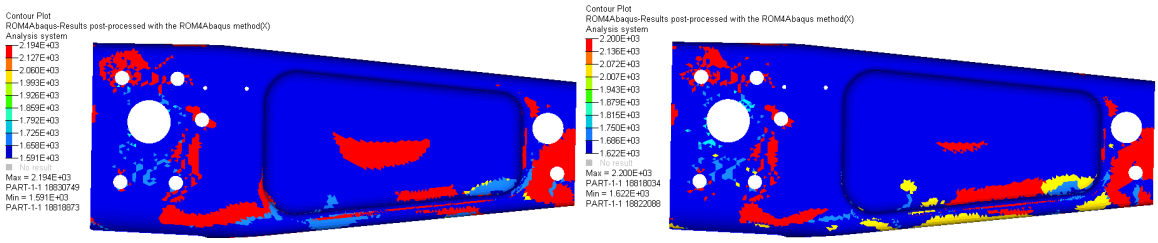
(b) Fixed engine speeds incremented by 100 rpm, 45.8e-3 to 601 MPa.

Figure 4.21: Critical stress comparison between using fixed engine speed increments of 100 or 200 rpm.

Studying what engine speed causes the critical stress in each element however reveals more of a difference as can be seen in Figure 4.22. The lowest critical engine speed, that affects the majority of the bracket differ only slightly between the two models however. When the fixed engine speeds are incremented by 200 rpm it is 1637 rpm and when they are incremented by 100 rpm the critical engine speed is found at 1643 rpm.

An additional comparison was made with a set of engine speeds incremented by 50 rpm in order to observe convergence in relation to decrease in step size. It should once again be noted that for this analysis an assumption was made regarding the cylinder pressure loads in the dynamic model as mentioned in subsection 3.5.2.

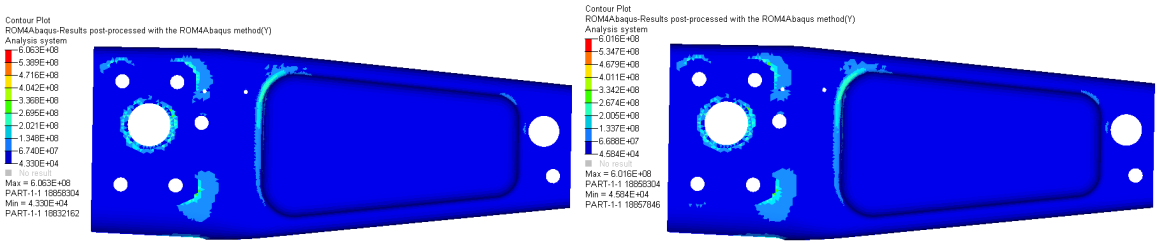
As can be seen in Figure 4.23, the difference between using 50 or 100 rpm increments regarding the stress distribution is as minor as it is between 100 and 200 rpm increments. The difference in stress amplitude is also minor; with an increment of 50 rpm the maximum stress amplitude in the bracket is 0.8% higher than with an increment of 100 rpm. At the countersink corner the stress amplitude is around 5% lower when increments of 50 rpm are used.



(a) Fixed engine speeds incremented by 200 rpm, 1591 to 2194 rpm.

(b) Fixed engine speeds incremented by 100 rpm, 1622 to 2200 rpm.

Figure 4.22: Critical engine speed comparison between using fixed engine speed increments of 100 or 200 rpm.

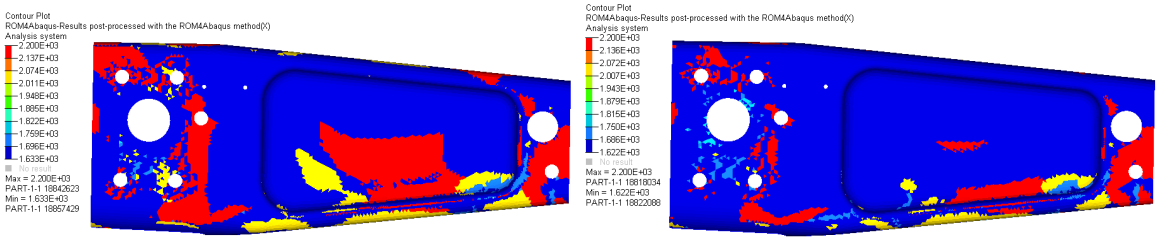


(a) Fixed engine speeds incremented by 50 rpm, 43.3e-3 to 606 MPa.

(b) Fixed engine speeds incremented by 100 rpm, 45.8e-3 to 601 MPa.

Figure 4.23: Critical stress comparison between using fixed engine speed increments of 50 or 100 rpm.

Figure 4.24 display the critical engine speed for each element in the analyzed bracket. As can be seen the major critical engine speed, marked by deep blue, is not as dominant when an increment of 50 rpm is used. However, the elements that experience another engine speed to be the most critical are those that do not experience a very high stress amplitude in the first place. The elements located at the countersink corner have converged at 1643 rpm as the critical engine speed.



(a) Fixed engine speeds incremented by 50 rpm, 1633 to 2200 rpm.

(b) Fixed engine speeds incremented by 100 rpm, 1622 to 2200 rpm.

Figure 4.24: Critical engine speed comparison between using fixed engine speed increments of 50 or 100 rpm.

In conclusion it can be said that changing the increment between the fixed engine speeds does not yield very much of a difference on the result. As the cylinder pressure is not recorded at increments of 50 rpm, but at increments of 100 rpm, this will continue to be used.

4.4 Step-by-step user guide

Part of the development of the method during the thesis work has been to make it easily accessible for new users. The whole purpose of the method is that it should be used in order to simplify the work of engineers and supply them with additional insights on which they may base their decisions. To this end a very detailed user guide has been written during the project which detail the exact methodology employed in using the method.

Step-by-step the whole procedure is described, from receiving the initial Excite model from NMDD to where the user can view the stress and critical engine speed distribution as seen in Figure 4.21. The instructions are complemented with images and screenshots from the commercial software used in the method.

The guide is structured in a way that allows it to be used for reference too if the user does not need to perform the whole FROM from beginning to end. An example for this would be an engineer who wants to analyze an auxiliary component on an engine which have been analyzed before, thus the whole engine FE model have been set up by another engineer. Then it is possible to refer to the user guide in order to get instructions on how to access the already prepared model, simply adding the new component to it and starting the analysis.

5 Discussion

In this chapter the results are discussed and compared to the goals set at the beginning of the project. The future of the method is also discussed, what needs to be performed and how it might come to use for the employees as Scania.

5.1 Conversion validity

As was shown in subsection 4.2.1 the conversion of the FE-model from NASTRAN to Abaqus was deemed successful enough to continue to use it for further analysis. But it was not a one hundred percent perfect match between the conversions. One could assume that the eigenvalues would be exactly the same between the two models, but it was not. The models that have been used during this project have been very complex, they contain many different parts that are connected with different rigid connections and contact conditions.

The cause for the difference in eigenvalues is probably due to that Abaqus and NASTRAN model some of these things differently, that some part of the model becomes more or less constrained than it was in the original model. One big error of this kind was encountered during the thesis work was a combined point mass and inertia element that was used in the NASTRAN model at several locations. The same type of element does not exist in Abaqus, and the conversion tool solved this by creating two point elements in the same location; one for the mass and one for the inertia.

The problem was that in the NASTRAN model the combined element was constrained to a single node that in turn was connected to the rest of the model, thus effectively adding the point mass and inertia to the full model. In Abaqus two nodes were created in the same location, where the mass element was connected to one and the inertia element connected to the other. But only one of the nodes, with the same node number as the one in NASTRAN, retained its connection to the rest of the model.

The new node in the Abaqus model was not connected to anything and therefore it had 6 unconstrained degrees of freedom, which caused numerical singularities when the model was submitted for solving on the cluster. This was a huge error and the results of it was obvious, even though it was hard to find the cause for it. It is deemed quite likely that the slight deviation between the models in the eigenvalue comparison is due to a similar problem, were the conversion tool is not able to exactly replicate the modeling, but that the results of the error is not as obvious.

One way to mitigate this problem would of course be to use NASTRAN as the solver instead of Abaqus, as was done in the first versions of the ROM, in order to completely forgo the conversion step of the method. This should however be avoided if possible, to use Abaqus as the solver was one of the main requirements of the project. As was shown in [Reu12] using NASTRAN yields output files with a size of an order of magnitude

larger than the corresponding Abaqus files. This means that using NASTRAN to solve the FROM for the whole engine, especially when analyzing a large auxiliary component, would yield files of immense size which should be avoided. In the same report, [Reu12], it was also shown that the solution speed using Abaqus was almost twice as fast as compared to using NASTRAN.

5.2 Sensitivity to number of engine orders used

From the sensitivity investigation performed, section 4.3, it may be concluded that most likely the choice of using 0.5 to 12 engine orders are a conservative one. Thus the computation time may be reduced by reducing the number of engine orders used to for example seven or eight, without changing the result of the method. The sensitivity analysis was carried out on and straight 6 cylinder engine and the results should be considered with that in mind. If the method is used on an engine with more cylinders, for example and V8 engine, the number of orders required for convergence would be higher. This might have been the original motivation for using orders up to the twelfth, even though it is not stated in the original report.

The time saved by using less engine orders are significant as the relation between the number of engine orders and the number of computation steps in Abaqus is linear. Therefore it is also a linear dependency to the solution time, if half an engine order is removed from the analysis two computational steps are removed. The first computation step that computes the eigenfrequencies will always be there, and that corresponds to as much as ten percent of the solution time, depending on the amount of elements in the model.

It should be noted that the sensitivity analysis was only performed on one component and it may be that other components are more sensitive to the higher engine orders. Evaluating the method fully in order to optimize its performance would take a while as a series of tests with a high accuracy would be necessary to be able to trust the result completely. Different auxiliary components, each finely meshed and located at different parts of the engine, should go through the same set of tests to find the convergence of the method with regard to the number of engine orders used.

5.3 Number of fixed engine speeds considered in the method

The study regarding the effect on the precision of the method, with regard to the number of fixed engine orders used, showed that a more sparse set of fixed engine speeds yielded different results than the more refined set, as would be expected. It would indeed improve the computation time for running the method, but only the first time when all the preprocessing is carried out. At large the performance increase seen is minor in comparison with the decrease in the quality of the results. As the computation time of the dynamic analysis is not a major contributor to the total solution time in the FROM, the time that can be saved by using a sparser set of fixed engine speeds is not worth the trade-off in precision.

Regarding the denser set of engine speeds it is not recommended either as the cylinder pressure load is only measured every hundredth rpm. By using a denser set another interpolation is added to the method which seems unnecessary. It is deemed better to keep using the loads that are confirmed. If a precision issue with the method is found in the future then it might be worth looking into the possibility of recording the aforementioned cylinder pressure with shorter increments in engine speed.

5.4 User friendliness/accessibility

During the thesis project there has been an underlying ideal that has been guiding the work. To not just develop a computational method that works, but to develop one that is easy to use in a way that invites new users to utilize and get use from it. This is evident in the premise of the method; to generalize it in such a manner that the complicated part, the preprocessing, need only be carried out once for each engine model. This generalization allows one person with a good understanding of the method to convert and prepare the different engine models in development, and then a lot of other people might benefit from it by using the converted and prepared model to analyze auxiliary components that they are working with.

This is possible since once the model has been converted and preprocessed for the FROM, the only thing needed by the user is to mesh, import and incorporate their component into the main FE-model in order to use the method. Thus no additional skills than the most basic will be required. This will hopefully ensure that the method becomes well used, which of course is the most primal goal of the thesis; to develop something that contribute to the work of engineers.

In order to introduce a new user to using the method a detailed user guide has been written during the thesis that thoroughly explain each step that needs to be taken in order to fully use the FROM. It will serve both as an practical introduction to using the method and as a reference work if there are any uncertainties while using the method.

All code that has been written during the project has been thoroughly commented, allowing for someone else to access and make changes or develop it further in the future. By utilizing a transparent method during the whole project it ensures that it does not become some obscure thing that only one or two persons are able to use, as is the case with many old methods used by companies. Transparency and knowledge regarding the inner workings of the methods used by the engineers ensures that they are comfortable with both using the methods and trusting the results.

5.5 Sources of error

Naturally there are a few possible sources of error that may effect the results seen in the project. There might both be errors with the commercial software used, the code written or the way some things are modeled. The most likely sources of error are described in this section along with mitigation propositions.

5.5.1 FE-model and model conversion

One possible source of error that need to be considered when using a whole engine model in the method is the constraining of the model to the ground. Physically there are engine brackets where the engine is fastened to the frame of the vehicle, and these need to be modeled in some manner. It would not be a good representation of reality to constrain the brackets directly to ground, as it would be much stiffer than in reality. Instead the interface is modeled as a combination of springs and dampers.

The possible error is with what constants these springs and dampers are modeled. There exist very little consistent data on the exact values which makes it quite perilous. Possibly the damping should even be nonlinear to correctly represent reality. The stiffness and damping effect the behaviour of the whole model as different amounts of the vibrations in the engine might be absorbed at the mounts. If the real damping is higher than the modeled one, it is possible that an engine speed that seem to be critical when running the FROM is not in the real world application if the main frequency is absorbed by the higher damping.

The most prominent source of error when using the FROM is with no doubt the conversion from NASTRAN to Abaqus format as has been discussed earlier. The model of the whole engine is very complex, thus there are a lot of parts where an error might occur. If the conversion tools are used on a model of a single meshed part they work well, it is the complexity of a large assembly model that invites the erroneous behaviour.

5.5.2 Physical testing

The physical testing done in order to verify the FROM would in conclusion seem to indicate a lower engine speed as the critical one than the one found by the use of the FROM. This might have several explanations. Assuming that the FROM is correct one might consider that the sensors only measure the excitation in a few points, and that this does not necessarily correspond directly to the stress, which is what is considered in the FROM. Thus it might be that even though the amplitude of the excitation is at its largest at a lower rpm, the highest stresses are experienced at another engine speed. For example, the countersink radius might be the most affected when the vibration happen in a certain direction, even though the amplitude of the excitation is lower than at some other mode.

Then there might be a problem with the FE-model where the bracket is fastened to the complete engine. In reality the bracket is bolted to the exhaust pipe, and in the FE-model this is modeled by using a TIE-contact. The TIE-contact is a rigid connection between two surfaces and it might be that it causes the bracket to become too stiff. If it is stiffer it might have higher eigenfrequencies, which is usually caused by higher engine speeds.

5.6 Future of the FROM

Hopefully the FROM will be used in a variety of projects, both for analyzing the components of already existing engines and to aid in the development of new engine platforms. The whole purpose of the new FROM in comparison to the old ROM is that it should be more accessible for the end user as they will only have to use one commercial software, Abaqus, in order to utilize the method.

Another possible use of the method has been discussed during the project, and that is the possibility of examining components that are experiencing sliding. This would be possible to do in the force loaded model as it does not inherently prohibit motion of any of the auxiliary components. Take for example the gasket between the oil sump and the engine block, this would be very hard to analyze in the old ROM as the oil sump would have to be locked at a certain amount of nodes were the excitation is applied, thus preventing sliding.

In the FROM on the other hand it would be possible to model the sliding condition between the fodder and the other components as the load on the model is applied on another part of the engine, the connecting rod bearings. Thus the load does not directly interfere with the component to be analyzed, which is a better representation of reality. In the beginning of the project it was decided that this would be a good alternative use of the method, but sadly there has not been time enough to test this approach during the project. It is thus recommended that this is done in the future; it is very effective if the same method, or a big part of it, can be used in order to analyze something different as this potentially saves a lot of development time.

Before the method is employed for use however, it is necessary to perform more verification testing. A comparison to the ROM was started during the thesis but could not be finished in time and is therefore not included in this report. Before the FROM is used to analyze components in the day to day work this verification and possibly more need to be finished.

Some improvements have been made during the project, for example the switch from nested loop summation to vector operations in the post-processing script. Still there are areas where additional improvements may be applied. One possible improvement would be to separate the eigenfrequency computation performed in the first step of the FE-computations. If the eigenfrequency analysis could be performed separately, and then be included into the rest of the computations it might be possible to better utilize the potential of the cluster where the computations are performed. As was mentioned in subsection 4.1.3 the current version of the FROM can access a limited amount of cores on the cluster due to the eigenfrequency step. If the steps could be separated then the major part of the FE-computations could be carried out using four times as many cores as it do currently, thus decreasing the solution time.

References

[16] *Abaqus Analysis User's Guide, 6.3.5 Natural frequency extraction.* 6.14. 2016.

[Dae09] J. Daelander. "Vibrationsutmattning av avgassamlare - En utveckling av ROM-metoden". MA thesis. KTH, 2009.

[Dan08] M. Danielsson. *SNABBARE OCH ROBUSTARE UTMATTNINGSUTVÄRDERING MED ROM-METODEN.* Tech. rep. m87. SCANIA, NMBS, 2008.

[Jön05] O. Jönsson. *Beräkningsrapport, Verifiering av interpoleringsmetod för att hitta värsta lastfallen för påhängskomponenter.* Tech. rep. M 81/186. SCANIA; NMBS, 2005.

[Lar04] O. Larsson. "Beräkningsmetodik: Dynamik hos motormonterade påhängskomponenter". MA thesis. KTH, 2004.

[Reu12] F. Reuterswärd. *Metodbeskrivning: ROM-metoden för Abaqus.* Tech. rep. 7009559. SCANIA, NMBS, 2012.