

Transient and Spectral Fatigue Analysis for Random Base Excitation

Master's thesis in Applied Mechanics

ALBIN BÄCKSTRAND
PRITHVIRAJ MADHAVA ACHARYA

MASTER'S THESIS IN APPLIED MECHANICS

Transient and Spectral Fatigue Analysis for Random Base Excitation

ALBIN BÄCKSTRAND
PRITHVIRAJ MADHAVA ACHARYA

Department of Mechanics and Maritime Sciences
Division of Dynamics
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2019

Transient and Spectral Fatigue Analysis for Random Base Excitation
ALBIN BÄCKSTRAND
PRITHVIRAJ MADHAVA ACHARYA

© ALBIN BÄCKSTRAND , PRITHVIRAJ MADHAVA ACHARYA, 2019

Master's thesis 2019:37
Department of Mechanics and Maritime Sciences
Division of Dynamics
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone: +46 (0)31-772 1000

Cover:

Rainflow count algorithm performed on a stress signal at the top left image. PSD signal at the top right image. S-N curves at the bottom left image, and histogram plot generated from a density function at the bottom right image.

Chalmers Reproservice
Göteborg, Sweden 2019

Transient and Spectral Fatigue Analysis for Random Base Excitation
Master's thesis in Applied Mechanics
ALBIN BÄCKSTRAND
PRITHVIRAJ MADHAVA ACHARYA
Department of Mechanics and Maritime Sciences
Division of Dynamics
Chalmers University of Technology

ABSTRACT

This thesis work gives an insight on how to estimate the fatigue damage using transient and spectral fatigue analysis for both uniaxial and multiaxial stress states in case of random vibration base excitation. The transient method involves a rainflow count algorithm that counts the number of cycles causing the fatigue damage, while the spectral method is based on probabilistic assumptions from which an expected value of fatigue damage can be estimated. The purpose was to compare the fatigue damage obtained from the transient and spectral approach, and evaluate the performance of the spectral method. In this study, the Dirlik's empirical formula has been selected for the spectral method, partly because it has proven to be a good implementation in structural mechanics fatigue. In order to account multiaxial fatigue in the calculations, the authors decided to use Dang Van equivalent stress for the transient analysis. In addition to that, the cycles were counted applying Wang and Brown's method, which can be seen as a more general extension of the original rainflow count algorithm. In the spectral analysis, the well known equivalent von Mises stress (EVMS) has been selected in order to account for multiaxial spectral analysis. The two methods were studied in association with an air dryer component that is attached to a chassis frame of a truck. The air dryer is subjected to random vibration via the mounting interface. The vibration was simulated by acceleration base excitation. Both uniaxial and multiaxial base excitation were investigated and the fatigue life was estimated in three selected hotspots on the surface of the air dryer component. The hotspots were chosen based on modal analysis. The Dirlik's empirical formula was showing promising estimation of the fatigue life similar to the rainflow count. In most cases, the difference in fatigue life between the two methods was less than 30 % for both uniaxial and multiaxial stress. However, Dirlik's formula was mostly showing more conservative results compared to the rainflow count. The cause of this could either be errors in the calculations or too short input signals. In some cases the difference between the methods were more significant, showing 200 % difference in fatigue life. The authors believe that this is most likely caused by mid stress effects in the Dang Van equivalent stress.

Keywords: Spectral fatigue analysis, transient fatigue analysis, power spectral density (PSD), transfer function, base excitation, Dirlik's empirical formula, equivalent von Mises stress (EVMS), rainflow count, Wang and Brown's method, Dang Van equivalent stress, Palmgren-Miner rule.

PREFACE

We are two students studying Applied Mechanics, and our interests emphasize structural dynamics and vibration. This topic was proposed by Volvo Trucks. The reason behind picking this topic was obvious for us – to understand more on how to estimate fatigue damage of components due to random vibration. This topic seemed interesting to us as it involved probability theory, signal processing, structural dynamics and fatigue analysis. How to assess damage from the randomness using spectral methods in frequency domain was the main idea for the master thesis proposal. Therefore, we both decided to engage, covering all the relevant topics needed in the context.

An air dryer component was suggested by Volvo Group Trucks Technology as a test subject. As they suspected, it was an interesting component to investigate. The theoretical background related to spectral and transient fatigue approach is explained in the theory section and how to solve it is explained in the methodology section. Results and discussions are grouped together in a separate section.

For the reader, we have made detailed explanation on each topic with colorful figures and tables. MSC NASTRAN - version 2018.2.1 was used as a FE software. Pre-processing of the component was done in ANSA - version 2017.0.2 and the postprocessing was done by META - version 16.1.3. Most part of signal processing, calculation and fatigue were done in MATLAB - version R2019a. The thesis work is concluded with a future scope for the reader to work on similar topics. The appendix contains codes and interesting references that has been used. We made sure that enough time was contributed on this thesis work; and a sheer dedication has been put forth in bringing out this report such that it is easy for the reader to understand the concepts. We made sure that much of the related topics to this thesis work was covered.

ACKNOWLEDGEMENTS

This thesis work has been only possible because of the following people. We cannot describe our work without acknowledging your endless support and patience. We would like to express our sincere gratitude to **Martin Olofsson**, Mechanics Research at RISE (Research Institute of Sweden) for bringing up this topic as a thesis work at Volvo Group Trucks Technology. Your valuable lectures on statistics, signal processing, random vibration and PSD helped us understand the subject. We thank you for all the time that you have invested introducing the subject to us, and recommending us to Volvo GTT family.

We would like to thank our supervisor **Lars Lindén**, Principal Vehicle Analyst at Volvo Group Trucks Technology, for constant support throughout the thesis work. Your enormous time and assistance have made it smooth already from the legal documentation work to all the technical aspects of this thesis work. Clearing all our doubts we threw at you, we are lucky to have you as our supervisor. Your inputs on how to handle the Linux cluster, ANSA, META, MSC NASTRAN and topics related to solid mechanics, vibration fatigue were very valuable. We would also like thank our supervisor **Anders Nord**, Feature Leader Durability at Volvo Group Trucks Technology, for demonstrating excellent knowledge on the practical approach. You were an inspiration. Your inputs in connecting the simulation and reality gave us a good idea on this subject. Your valuable support on structural dynamics and durability has prevented us from some, now obvious, errors and loopholes. Our humble respect to you for educating us.

We would like to thank our examiner **Thomas Abrahamsson**, Professor at department of Applied Mechanics, Chalmers University of Technology. Your inputs and suggestions made us write this report more professionally. We thank you for taking up the responsibility on the documentation part at Chalmers University of Technology. We would like to acknowledge **Anders Ekberg**, Professor at department of Applied Mechanics, Chalmers University of Technology, for your support on uniaxial and multiaxial fatigue. Your suggested literatures and support materials were of great help.

Last but not least, we like to mention our Group Manager **Inge Johansson** and all other colleagues at department of Vehicle Analysis, Volvo Group Trucks Technology who has helped us directly or indirectly. We will cherish the time spent at your department. We would also like to thank our family and friends for their well wishes and moral support in helping us to carrying out this work.

NOMENCLATURE

$\mathcal{F}(\bullet)$	Fourier transform operator	\mathbf{S}, S_{ij}	stress PSD matrix
$\mathcal{F}^{-1}(\bullet)$	inverse Fourier transform operator	S_{vM}	EVMS (equivalent von Mises stress)
$\hat{\bullet}$	Fourier transform of an arbitrary quantity \bullet	m_k	k :th moment of area
$*$	convolution operator	$m_{\text{vM},k}$	k :th moment of area of the EVMS
t	time	f_m	mean frequency
f	frequency	E_0	expected number of zero up crossings per unit time
Δt	time step	E_p	expected number of peaks per unit time
T	time length	α_1, α_2	first and second order irregularity factors
f_s	sampling frequency	$E[\bullet]$	mean value operator
f_{ny}	Nyquist frequency	RMS	root-mean-square
$ \bullet $	absolute value	$\text{Var}(\bullet)$	variance operator
$\bar{\bullet}$	complex conjugate	$\text{Cov}(\bullet)$	covariance operator
$\text{Re}\{\bullet\}$	extract real part	ρ_{xy}	correlation factor between time signal x and y
$\text{Im}\{\bullet\}$	extract imaginary part	$\boldsymbol{\rho}, \rho_{ij}$	correlation matrix
$\arg\{\bullet\}$	extract argument	$p_X(x)$	density function of an arbitrary stochastic variable X
$\dot{\bullet}$	time derivative	μ_X	mean value of a normal distributed variable X
$\ddot{\bullet}$	second order time derivative	σ_X	standard deviation of a normal distributed variable X
a_1, a_2, a_3	input acceleration signals	\bullet^T	transpose of an arbitrary matrix \bullet
R_{xx}	auto-correlation of a time signal x	\bullet^\dagger	Hermitian conjugate of an arbitrary matrix \bullet
R_{xy}	cross-correlation of two time signals x and y	$:$	double contraction product
\mathbf{R}, R_{ij}	correlation matrix	$\det(\bullet)$	determinant operator
G_{xx}	PSD of a time signal x	$\text{eig}(\bullet)$	eigenvalue operator
G_{xy}	cross-PSD of time signals x and y		
\mathbf{G}, G_{ij}	PSD matrix		

\otimes	tensor product	σ_{ar}	reversible stress amplitude
$\underline{\otimes}$	modified tensor product	$\Delta\sigma$	stress range
H	transfer function	$\Delta\sigma_r$	reversible stress range
\mathbf{H}, H_{ij}	transfer function matrix	σ_m	mid stress
\mathbf{M}	global mass matrix	$\sigma_{dv,max}$	maximum Dang Van equivalent stress
\mathbf{C}	global damping matrix	N_f	number of cycles to failure
\mathbf{K}	global stiffness matrix	D	fatigue damage
\mathbf{f}	external load	T_{life}	fatigue life (in time)
\mathbf{f}_F	prescribed free load	σ_u	ultimate strength
\mathbf{f}_C	reaction load	N'_f	number of cycles to failure at the upper fatigue limit
\mathbf{f}'_F	external load when taking the prescribed motion into consideration	N_e	number of cycles to failure at the lower fatigue limit
\mathbf{q}	nodal variables, degrees of freedom (DOF)	$\Delta\sigma'_r$	stress range at the upper fatigue limit
\mathbf{q}_F	free DOF	$\Delta\sigma_{er}$	stress range at the lower fatigue limit
\mathbf{q}_C	constrained DOF (prescribed motion)	C, k	reversible S-N curve fitting parameters
$\boldsymbol{\sigma}, \sigma_{ij}$	stress tensor	c_{dv}	Dang Van parameter
$\underline{\boldsymbol{\sigma}}$	stress tensor in Voigt notation	m_e	bending fatigue limit factor
σ_h	hydrostatic stress	m_t	load type factor
$\boldsymbol{\sigma}^d, \sigma_{ij}^d$	deviatoric stress tensor	m_d	size factor
$\boldsymbol{\sigma}_{ij,mid}^d$	deviatoric mid stress tensor	m_s	surface finish factor
σ_{vM}	von Mises stress	m_o	factor that represents other effects
\mathbf{Q}	Q-matrix for the von Mises stress	m', m	total reduction factor for the upper and lower fatigue limit, respectively
τ_{Tr}	Tresca shear stress	ζ_i	modal damping coefficient for i :th eigenmode
δ_{ij}	Kronecker delta tensor	Φ_i	i :th eigenmode of an undamped system
$\Gamma(\bullet)$	gamma function	ω_i	eigenfrequency for i :th eigenmode of an undamped system
σ_a	stress amplitude		

CONTENTS

Abstract	i
Preface	iii
Acknowledgements	iii
Nomenclature	v
Contents	vii
1 Introduction	1
1.1 Purpose	1
1.2 Limitations	2
1.3 Air dryer Bracket – Example Structure	2
2 Theory	4
2.1 Fourier Transform	4
2.2 Power Spectral Density	5
2.2.1 Cross-PSD	7
2.3 Expected Value, Variance and Covariance	7
2.4 Random Vibration – Normal Distribution	10
2.4.1 Conservation of Normal Distribution	10
2.5 Transfer Functions	11
2.5.1 Filtering the Input Signal	13
2.5.2 Transfer Functions and PSD:s	13
2.6 Transfer Functions in Finite Element Model	14
2.6.1 Multiaxial Base Excitation – Prescribed Motion	15
2.6.2 Coupled and Uncoupled Interfaces	16
2.7 Modal Analysis	17
2.8 Fatigue Damage	17
2.8.1 Accumulated Damage and Rainflow Count	19
2.9 Multiaxial Fatigue	21
2.9.1 Rainflow Count – Multiaxial Stress	23
2.9.2 Equivalent Stress – Dang Van Criterion	23
2.9.3 Equivalent S-N Curve	24
2.10 Random Fatigue Analysis – Spectral Methods	24
2.10.1 Peak and Stress Range Distribution	25
2.10.2 Dirlik’s Empirical Formula	26
2.10.3 Account for Fatigue Limit	27
2.10.4 Equivalent PSD Stress – EVMS	27
3 Methodology	29
3.1 Generate Input Signals	31
3.1.1 Input PSD Matrix	32
3.2 Modal Analysis and Selection of Evaluation Points	33
3.2.1 Stress Singularity	35
3.3 Evaluate Transfer Functions	37
3.4 Stress PSD matrix, EVMS and Dirlik’s Empirical formula	38
3.5 Dynamic Response	40
3.6 Stress Time Signals and Rainflow Count	40
3.6.1 Rainflow Count – Halfcycles	41
3.6.2 Wang and Brown’s method – Multiaxial Rainflow Count	41
3.7 S-N Curve and Fatigue Parameters	43

4 Results and Discussion	48
4.1 Fatigue Life Comparison	48
4.2 Histogram Comparison	52
4.3 Some Interesting Properties of the Stress Signals	55
4.4 Observations of Wang and Brown's Method	55
4.5 Mid Stress Effects	55
4.6 Limitations of EVMS	60
4.7 Computation Time	61
5 Future Scope	62
A Code	63
A.1 MATLAB Code	63
A.1.1 Function <code>rainflow_count_halfcycles.m</code>	63
A.1.2 Function <code>multiaxial_rainflow_count.m</code>	64
A.1.3 Function <code>rainflow_count.m</code>	66
A.1.4 Function <code>Dirliks_formula.m</code>	67
A.1.5 Function <code>fatigue_EVMS_Dirlik.m</code>	68
A.1.6 Function <code>DangVan.m</code>	71
A.1.7 Function <code>fatigue_WangBrown_DangVan.m</code>	72
A.1.8 Function <code>time2PSDmatrix.m</code>	75
A.1.9 Function <code>Fourier2PSDmatrix.m</code>	76
A.1.10 Function <code>transfer_PSD.m</code>	76
A.1.11 Function <code>transfer_Fourier.m</code>	77
A.1.12 Function <code>file_scanner.m</code>	78
A.1.13 Main code example <code>main_example.m</code>	80
A.1.14 Generate white noise, example <code>generate_white_noise.m</code>	81
A.2 NASTRAN Code	82
A.2.1 Frequency Response	82
A.2.2 Transient Response	83
References	85

1 Introduction

Random vibration is an unwanted but inevitable feature that is commonly found in many structures. The vibration can, in the long run, cause serious fatigue damage, which often leads to mechanical failure. It is therefore important to validate the durability of different components in order to estimate their time in service. The fatigue damage can be estimated by implementing a finite element model of the component in question and calculate the dynamical stress response at selected points, and thereafter perform a fatigue analysis on each point respectively using a rainflow count algorithm. However, if the stress history is considered random, the fatigue damage can be estimated from a statistical point of view. If the probability density function for the stress is known, it is possible to estimate an expected value of the fatigue damage using probability theory. This kind of probabilistic methods are known as spectral methods and are easy and fast ways to estimate fatigue life without having to process long time signal data [19].

There are great varieties of spectral methods that are based on different probability distributions of the stress history. Most common is the assumption that the stress has static normal (Gaussian) distribution, which is a convenient assumption when dealing with random vibration. In this study, Dirlik's empirical formula was implemented, since it has proven to be a useful spectral method that gives acceptable results for both wide- and narrowband signals [13]. Dirlik's formula is in the family of spectral methods where normal distribution is considered.

All statistical information that are needed in the spectral methods can be extracted from the so called power spectral density (PSD) of the stress history. The PSD shows the power distribution of the stress along the frequency spectrum, and can be seen as a frequency domain representation of the stress history. The PSD can reveal much information about the stress history, such as the expected number of peaks per unit time or its root-mean-square (RMS). Hence, when using spectral methods to estimate the fatigue life, there is no need to calculate the stress history itself. Instead, the PSD stress can be obtained from a frequency response analysis. Also, there is no need to perform rainflow counting.

There are various methods in order to account for multiaxial fatigue in frequency domain. In this study we decided to implement the equivalent von Mises stress (EVMS). The EVMS has found a wide use in the random fatigue community, partly thanks to its simplicity but also due to its preservation of normal distribution. The purpose of the EVMS is to reduce a multiaxial PSD problem into a uniaxial PSD. This allows us to evaluate the fatigue damage using only one equivalent PSD [25].

This thesis work involves comparing the spectral method with the state-of-the-art transient method, for both uniaxial and multiaxial fatigue. In the transient method, the fatigue damage was calculated explicitly from the dynamical response using rainflow count. In order to account the multiaxial fatigue, the Dang Van equivalent stress was used. In addition, the Wang and Brown's method was implemented in order to extract stress cycles from multiple stress signals. Wang and Brown's method can be seen as an extended version of the original rainflow count algorithm, and is applicable on non-proportional loading cases [32].

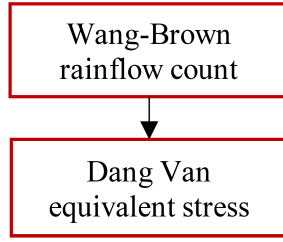
The study was made on a finite element structure representing an air dryer bracket component. The air dryer bracket is attached to a chassis frame of a truck. Irregularities from the road and the engine induces vibrations in the chassis frame that will affect the air dryer bracket accordingly. The vibration was simulated using random base excitation on the mounting interface with prescribed acceleration. The acceleration signals were generated in MATLAB from a normal distribution as a pre-process for the prescribed motion in the finite element analysis.

Both uniaxial and multiaxial base excitation was considered, and the transient and frequency responses were analyzed at selected hotspot elements on the surface of the air dryer bracket. The hotspots were selected based on modal analysis to hint on where large stresses are initiated when the eigenmodes are excited. The intension was not to do an extensive search for material points that experience largest stresses, but to find some interesting points to study. The finite element model was solved by MSC NASTRAN, and the transient response and frequency response were imported and processed in MATLAB for further fatigue analysis investigation.

1.1 Purpose

The purpose of the report is to compare the spectral and transient method for both uniaxial and multiaxial stress, along with uniaxial and multiaxial base excitation. In Fig. 1.1 the two methods are presented side by side just for clarification. In the transient approach the stress cycles causing fatigue damage are counted using rainflow count algorithm. In multiaxial fatigue, Wang and Brown's rainflow count was used in order

Transient Approach



Spectral Approach

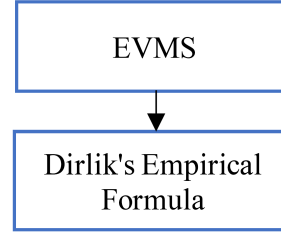


Figure 1.1: The transient and spectral approach for this study. Wang-Brown rainflow count method is used in the transient analysis in order to account for stresses in the multiaxial stress state. Dang Van equivalent stress is incorporated to convert the multiaxial stress state to equivalent uniaxial stress. A method called equivalent von Mises stress (EVMS) is used in spectral analysis to reduce multiaxial stress state to uniaxial stress PSD. The EVMS is then implemented to construct the probability density function in Dirlik's empirical formula.

to extract stress cycles from multiple stress components. In addition, the multiaxial stress state was reduced to an equivalent Dang Van stress before any further damage accumulation was performed. Similarly, in the spectral approach, the EVMS was used to convert multiple PSD stress signals into an equivalent PSD signal. The properties of the equivalent PSD signal were then applied in Dirlik's formula to construct a probability density function for the stress ranges.

The fatigue life was estimated in selected hotspot elements at the surface of the air dryer bracket using both the transient and the spectral method. The performance of Dirlik's formula and EVMS was evaluated by comparing the fatigue life from the rainflow count. A large portion of the study was dedicated in developing MATLAB code that could process the data from MCS NASTRAN solver and estimates the fatigue damage using both these methods.

1.2 Limitations

The project includes many complex features, and hence it is crucial to make some simplifications. These simplifications can be seen as limitations of the project and will affect the outcome. Linear isotropic material was considered, and the kinematic behaviour was assumed to be linear (small strain theory), meaning that the whole structure is treated as a linear system. The component was subjected to relatively small loadings which implies that the stress was not reaching the yield limit of the material. Because of this, only high cycle fatigue was considered throughout the project.

The component was assumed to have no surface irregularities or defects. No notch factors or any kind of stress raisers were used in this project. Also, fracture mechanics was not considered, and the concepts of crack propagation was omitted. In addition, no temperature or any sort of external effects such as corrosion was incorporated in the problem definition.

The study was limited to the methods proposed in Fig. 1.1. Dirlik's empirical formula assumes normal distributed stress history, and no mid stress effects. Mid stress correction in the spectral approach was not considered in this context.

1.3 Air dryer Bracket – Example Structure

To compare both transient and spectral method, an air dryer bracket is taken as an example structure. The air dryer is a component that is mounted on the chassis frame of a truck with the help of a bracket. The function of an air dryer is to dry the sucked air and prevent any adulteration that happens during its passage into the air compressor. The bracket that holds the air dryer is shown in Fig. 1.2.

The model is meshed in ANSA with total of 4340 elements, out of which 4240 are quadrilateral elements and 100 are triangular elements. The element size is not more than 10 mm and the material is steel with density equal to 7.85×10^{-6} kg/m³. The entire bracket except the bolt has Young's modulus $E = 210$ GPa,

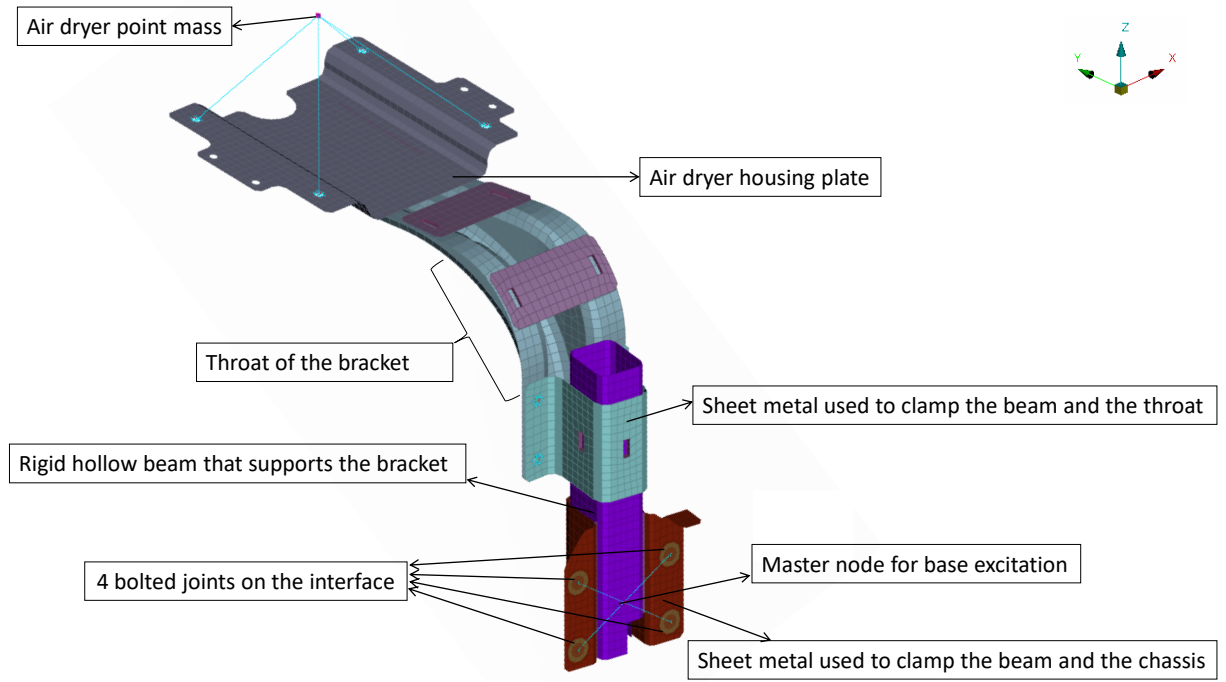


Figure 1.2: Schematic model of a bracket used by Volvo GTT to mount an air dryer. Several parts (colored differently) are linked together to form a L-shaped bracket. The bottom right extreme (brown color) is the interface that is bolted to the chassis. The hollow beam (purple color) in the centre gives a good strength and supports the bracket. The red node located at the top left corner of the bracket is a representation of a point mass of 9 kg that resembles the air dryer. The location of the point mass is decided based on the centre of mass of the air dryer.

and Poisson's ratio $\nu = 0.28$. The bolt on the other hand has Young's modulus $E = 205$ GPa, and Poisson's ratio $\nu = 0.3$.

In the model, the mass of the air dryer is represented by a point mass which is attached to the air dryer bracket with the help of multiple point constraints (MPC). The reader should be aware that the analysis focus on the bracket and not on the air dryer. This is why only the mass becomes an essential parameter rather than the air dryer geometry itself. Due to the presence of an air dryer point mass of 9 kg, the dynamics of the bracket is affected. On the other end of the bracket, the interface is rigidly fixed to the chassis. The four bolted joints are excited with identical prescribed motion. Therefore, the four joints are coupled into one master node to which a load is applied, or more precisely, to which base excitation is applied.

2 Theory

This chapter covers all the relevant theories that have been used or are related to the different subjects in the project. These theories provide a deeper understanding of each branch and gives a better aspect on the project. If the reader is not familiar with random fatigue theory or base excitation, this chapter should be convenient since it provides a lot of information and derivations. Also, some crucial limitations and assumptions are discussed in the following sections.

The theory chapter focuses mainly on the structural dynamics, uniaxial and multiaxial fatigue analysis, probability theory, Fourier analysis, power spectral density and base excitation. These theories are presented in a more general context and are also applied to the specific problems of this project.

2.1 Fourier Transform

Fourier analysis is a powerful tool that is widely used in science and engineering. It can be applied to various problems, especially for signal processing and in the process of solving differential equations [12]. In this context, the well-known Fourier transform will be commonly used throughout the theory chapter and is an important feature of the project. For that reason, it deserves a short introduction.

According to Fourier analysis, any periodic time signal can be represented as a superposition of sinusoidal waves with different amplitudes, frequencies and phases [14]. The Fourier transform is an excellent tool that can be used to map the frequency spectrum of the time signal, which will provide information about the amplitude distribution as well as the phase of the sinusoidal waves. The Fourier transform of an arbitrary time signal $x(t)$ is defined as

$$\hat{x}(f) = \mathcal{F}(x(t)) = \int_{-\infty}^{\infty} x(t) e^{-i2\pi ft} dt \quad (2.1)$$

where $\hat{x}(f)$ represents the Fourier transform of $x(t)$, and f is the frequency. From now on the Fourier transform of a quantity q will be denoted with a hat sign as \hat{q} . Further, the Fourier transform operator is denoted as $\mathcal{F}(\bullet)$, where the variable inside the bracket specifies the time dependent quantity. It follows directly from the definition in Eq. (2.1) that the Fourier transform is indeed a linear operator. The angular frequency $\omega = 2\pi f$ is commonly used as a transformation variable in many literatures, but the frequency f turns out to be a more convenient choice in this context.

The Fourier transform is in general complex valued, and it is sometimes conventional to present it in polar form as

$$\hat{x} = |\hat{x}(f)| e^{i \arg\{\hat{x}(f)\}} \quad (2.2)$$

The absolute value of the Fourier signal $|\hat{x}(f)|$, also called the amplitude spectrum, represents the amplitude distribution of the time signal. Similarly, the argument of the Fourier signal $\arg\{\hat{x}(f)\}$ holds information about the phase. However, the amplitude spectrum is usually more interesting, and it can reveal a great deal of information about the frequency composition of the time signal.

An illustrative example is presented in Fig. 2.1 where a time signal and its corresponding amplitude spectrum is shown. The time signal is fairly random, and it is hard to distinguish the sinusoidal waves and their frequencies. The amplitude spectrum, on the contrary, tells us that the amplitude concentration is largest close to the frequencies 20 Hz, 40 Hz and 80 Hz, which can be seen due to the distinct peaks. Hence, sinusoidal waves with frequencies close to 20 Hz, 40 Hz and 80 Hz will dominate the time signal, and their amplitudes, relatively speaking, are larger compared to other frequencies.

Note however that the frequency variable f can be both negative and positive since it is defined on the whole real axis. As shown in Fig. 2.1, the amplitude spectrum $|\hat{x}(f)|$ has associated sets of negative frequencies, which makes it symmetric. This is based on the fact that the time signal is real. From the definition of Fourier transform, it can be verified that the Fourier signal $\hat{x}(f)$ becomes complex symmetric when $x(t)$ is real, i.e. $\hat{x}(f) = \overline{\hat{x}(-f)}$, which indeed makes the amplitude spectrum symmetric $|\hat{x}(f)| = |\hat{x}(-f)|$. For that matter, it is enough to just analyze the positive frequency spectrum, since the negative frequency spectrum is just a mirrored copy. If we know one side of the spectrum, we also know the other side.

The Fourier transform does not distort or change the time signal. No information of the signal is lost, it is just transformed into frequency domain. Every unique time signal has a corresponding unique Fourier transform. Thus, it is possible to transform a signal back into time domain. This is done by employing the

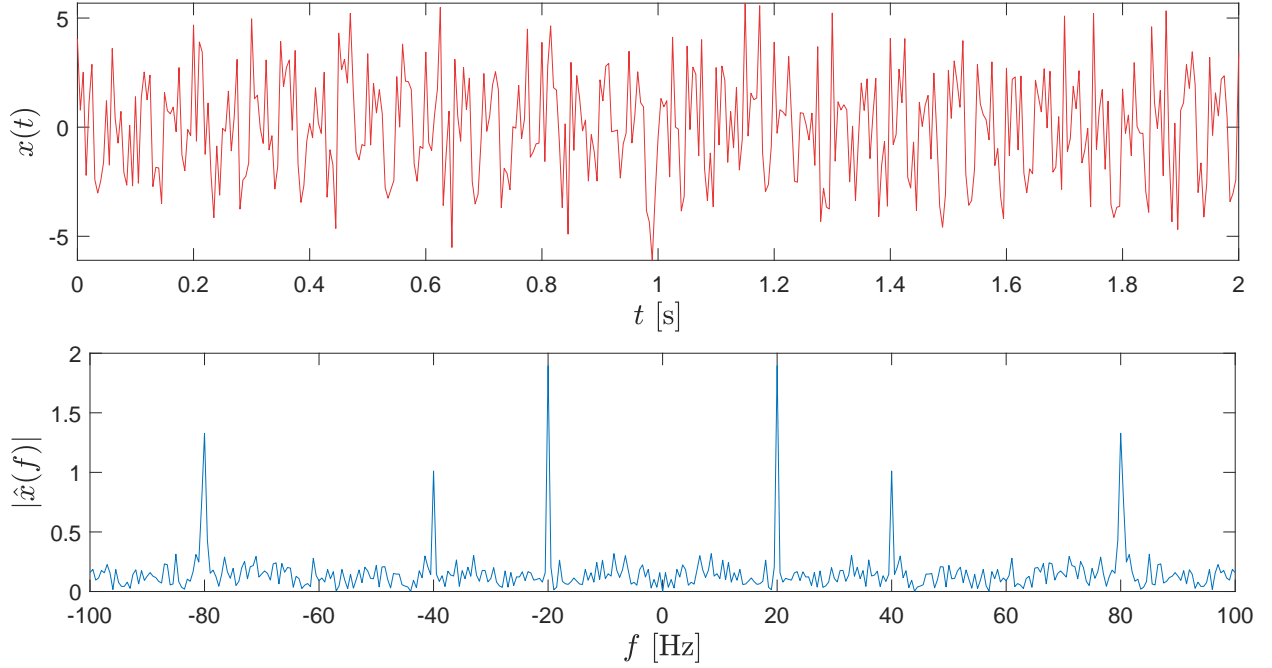


Figure 2.1: Time signal $x(t)$ and the corresponding amplitude spectrum $|\hat{x}(f)|$.

inverse Fourier transform, defined as

$$x(t) = \mathcal{F}^{-1}(\hat{x}(f)) = \int_{-\infty}^{\infty} \hat{x}(f) e^{i2\pi ft} df \quad (2.3)$$

where $\mathcal{F}^{-1}(\bullet)$ denotes the inverse Fourier transform operator.

From the Fourier transform follows a variety of important properties. Some of them are listed below and will be applied throughout the theory chapter [29].

$$\mathcal{F}(ax(t) + by(t)) = a\hat{x}(f) + b\hat{y}(f), \quad (2.4)$$

$$\mathcal{F}(\dot{x}(t)) = 2\pi fi \hat{x}(f), \quad (2.5)$$

$$\mathcal{F}(x * y(t)) = \hat{x}(f) \hat{y}(f), \quad (2.6)$$

$$\int_{-\infty}^{\infty} x(t) \overline{y(t)} dt = \int_{-\infty}^{\infty} \hat{x}(f) \overline{\hat{y}(f)} df \quad (2.7)$$

The first property in Eq. (2.4) declares the linearity of the Fourier transform operator. The second one in Eq. (2.5) shows the relation between the time derivative of a signal and its Fourier transform. Furthermore, the third relation in Eq. (2.6) explains the convolution property, where the operation $x * y(t)$ denotes the convolution between two signals $x(t)$ and $y(t)$, defined as

$$x * y(t) = \int_{-\infty}^{\infty} x(t - \tau) y(\tau) d\tau = \int_{-\infty}^{\infty} x(\tau) y(t - \tau) d\tau \quad (2.8)$$

The last property in Eq. (2.7) is the Parseval's theorem and will be used in relation to the power spectral density in Section 2.3.

2.2 Power Spectral Density

The power spectral density (PSD) is an important concept that is commonly associated with spectral analysis. It has found a wide use, especially in random fatigue, and plays a major role when performing fatigue analysis

in frequency domain [14]. The definition, however, varies depending on literature, but usually it is defined as the Fourier transform of the auto-correlation $R_{xx}(\tau)$. The auto-correlation of a real signal $x(t)$ is stated as

$$R_{xx}(\tau) = \int_{-\infty}^{\infty} x(t + \tau) x(t) dt \quad (2.9)$$

and is a measure on how well the signal $x(t)$ correlates with itself for some time lag τ [21]. The PSD of the signal $x(t)$ is now given as

$$G_{xx}(f) = \int_{-\infty}^{\infty} R_{xx}(\tau) e^{-i2\pi f\tau} d\tau \quad (2.10)$$

Substitute the auto-correlation Eq. (2.9) in Eq. (2.10) and using the definition of Fourier transform in Eq. (2.1) yields $G_{xx}(f) = |\hat{x}(f)|^2$. The PSD is usually scaled by the time length T of the signal, giving

$$G_{xx}(f) = \frac{1}{T} |\hat{x}(f)|^2 \quad (2.11)$$

According to Eq. (2.11) the PSD, also called auto-PSD, can be seen as the square of the amplitude spectrum. Hence, it does not provide any phase information about the signal $x(t)$, making it impossible to convert it back to a unique time signal $x(t)$. However, it is fairly easy to construct various time signals that share the same PSD signal.

For random noise, the phase information is often unnecessary, but the PSD, on the other hand, can reveal a great deal of information about the time signal. One important concept called moment of area can extract various information about the time signal $x(t)$. The definition can be stated as

$$m_k = \int_{-\infty}^{\infty} G_{xx}(f) |f|^k df \quad (2.12)$$

where k marks the moment degree [2]. For example, m_0 denotes the zero moment of area, m_1 the first moment of area, and so on. When the signal is real, the amplitude spectrum $|\hat{x}(f)|$ becomes symmetric (see Section 2.1), which makes the PSD symmetric. Because of that, the k :th moment of area can also be written as

$$m_k = 2 \int_0^{\infty} G_{xx}(f) f^k df, \quad (f \geq 0) \quad (2.13)$$

The latter is more common in literature, since only the positive side of the frequency spectrum is considered. However, different literatures scales of the Fourier transform and the PSD in different ways, which could lead to wrong results if the user is not being cautious. The moment of area in Eq. (2.12) and in Eq. (2.13) are defined according to the definition of Fourier transform and PSD in Eq. (2.1) and Eq. (2.10), respectively.

So, what can the moment of area m_k of a Gaussian signal $x(t)$ provide? Below follows some important properties.

$$\text{RMS} = \sqrt{m_0}, \quad (2.14)$$

$$f_m = \frac{m_1}{m_2}, \quad (2.15)$$

$$E_0 = \sqrt{\frac{m_2}{m_0}}, \quad (2.16)$$

$$E_p = \sqrt{\frac{m_4}{m_2}}, \quad (2.17)$$

$$\alpha_k = \frac{m_k}{\sqrt{m_0 m_{2k}}} \quad (2.18)$$

Here RMS denote the root-mean-square, f_m is the mean frequency, E_0 the expected number of zero up crossings per unit time, and E_p is the expected number of peaks per unit time [14]. The dimensionless constants α_k are called irregularity factors. The first and second order irregularity factors α_1 and α_2 ranges between 0 and 1, and are both measures of the bandwidth of the signal $x(t)$, where 0 indicates a wideband signal and 1 indicates a narrowband signal [2].

2.2.1 Cross-PSD

The PSD can also be calculated for two different time signals $x(t)$ and $y(t)$. Then it is called a cross-PSD and is defined as the Fourier transform of the cross-correlation $R_{xy}(\tau)$. The cross-correlation measures the correlation between the two signals for a time lag τ and is stated as

$$R_{xy}(\tau) = \int_{-\infty}^{\infty} x(t + \tau) y(t) dt \quad (2.19)$$

Consequently, the cross-PSD can be obtained in a similar fashion as in Eq. (2.10), leading to

$$G_{xy}(f) = \frac{1}{T} \hat{x}(f) \overline{\hat{y}(f)} \quad (2.20)$$

The cross-PSD is in general complex valued unlike the auto-PSD and carry information about the correlation between the signals $x(t)$ and $y(t)$.

For two time signals, there are four different sets of PSD:s, namely G_{xx} , G_{yy} , G_{xy} and G_{yx} . Similarly, for three time signals $x(t)$, $y(t)$ and $z(t)$ there are nine sets of PSD:s. They can all be arranged in a PSD matrix $\mathbf{G}(f)$ as [2]

$$\mathbf{G}(f) = \begin{bmatrix} G_{xx}(f) & G_{xy}(f) & G_{xz}(f) \\ G_{yx}(f) & G_{yy}(f) & G_{yz}(f) \\ G_{zx}(f) & G_{zy}(f) & G_{zz}(f) \end{bmatrix} \quad (2.21)$$

It follows directly from the definition in Eq. (2.20) that $G_{xy}(f) = \overline{G_{yx}(f)}$ which proves that the PSD matrix is complex symmetric, i.e. $\mathbf{G} = \overline{\mathbf{G}}^T = \mathbf{G}^\dagger$, where \dagger is the Hermitian operator. Consequently, the diagonal components of $\mathbf{G}(f)$ are always real, while the off-diagonal cross-PSD components are generally complex. In a similar manner, we can arrange a correlation matrix as $\mathbf{R}(\tau)$ [3]. Hence, by the definition of cross-PSD the PSD matrix can be obtained as

$$\mathbf{G}(f) = \frac{1}{T} \begin{bmatrix} |\hat{x}|^2 & \hat{x} \overline{\hat{y}} & \hat{x} \overline{\hat{z}} \\ \overline{\hat{y}} \hat{x} & |\hat{y}|^2 & \overline{\hat{z}} \hat{y} \\ \overline{\hat{z}} \hat{x} & \overline{\hat{z}} \hat{y} & |\hat{z}|^2 \end{bmatrix} = \frac{1}{T} \begin{bmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{bmatrix} \begin{bmatrix} \overline{\hat{x}} & \overline{\hat{y}} & \overline{\hat{z}} \end{bmatrix} \quad (2.22)$$

By arranging all the time signals in a vector $\mathbf{x}(t) = [x \ y \ z]^T$ with Fourier transform $\mathcal{F}(\mathbf{x}) = \hat{\mathbf{x}}(f) = [\hat{x} \ \hat{y} \ \hat{z}]^T$, the PSD matrix can be written in more compact form:

$$\mathbf{G}(f) = \frac{1}{T} \hat{\mathbf{x}}(f) \hat{\mathbf{x}}(f)^\dagger \quad (2.23)$$

However, the equation holds true for any number of time signals, not just three. The same equation can also be written using index notation as

$$G_{ij}(f) = \frac{1}{T} \hat{x}_i(f) \overline{\hat{x}_j(f)} \quad (2.24)$$

The stress PSD is something that will be investigated further in this chapter and is important in random fatigue. In general, multiaxial stress state, there are 6 stress components. The PSD matrix will then have $6 \times 6 = 36$ components. However, the number of non-zero stress components can be reduced depending on the geometry and load situation. In this study, plane stress will be considered giving 3 stress components and 3×3 PSD matrix [5].

2.3 Expected Value, Variance and Covariance

Consider a time dependent stochastic variable $X = X(t)$ with a corresponding normalized static density function $p_X(x)$. The expected value of X can then be calculated as

$$E[x] = \int_{-\infty}^{\infty} x p_X(x) dx \quad (2.25)$$

where $E[\bullet]$ denotes the mean value operator. The expected value of a stochastic time signal $x(t)$, can also be calculated as

$$E[x] = \frac{1}{T} \int_0^T x(t) dt \quad (2.26)$$

The mean square can be obtained in a similar fashion as

$$E[x^2] = \int_{-\infty}^{\infty} x^2 p_X(x) dx = \frac{1}{T} \int_0^T x(t)^2 dt \quad (2.27)$$

Further, the variance of a time signal $x(t)$ can be obtained as

$$\text{Var}(x) = E[(x - E[x])^2] = E[x^2] - E[x]^2 \quad (2.28)$$

and indicates how dispersed the stochastic variable is around its expected value [26].

From Parseval's theorem in Eq. (2.7), we can derive an important property that the PSD signal possess. The area under the whole PSD curve is equal to the mean square of the time signal, as

$$E[x^2] = \int_{-\infty}^{\infty} G_{xx}(f) df \quad (2.29)$$

In other words, the mean square is simply the zero moment of area m_0 . The square root of m_0 is then equal to the root-mean-square of the signal, already stated in Eq. (2.14).

The variance, however, does only measure the disperse of one stochastic time signal. The covariance operator $\text{Cov}(\bullet, \bullet)$ on the contrary measures the dispersion between two different stochastic variables. For two real signals $x(t)$ and $y(t)$, the covariance is defined as

$$\text{Cov}(x, y) = E[(x - E[x])(y - E[y])] = E[xy] - E[x]E[y] \quad (2.30)$$

Note that the random variables might be dependent and hence they could share a joint density function $p_{XY}(x, y)$. Then, the expected value of xy is obtained as

$$E[xy] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} xy p_{XY}(x, y) dx dy \quad (2.31)$$

If the random variables are independent, the joint density function can be subdivided as $p_{XY}(x, y) = p_X(x) p_Y(y)$. In that case it is easy to verify that $E[xy] = E[x]E[y]$, which in turn gives zero covariance [23].

In a similar way as in Eq. (2.26) the expected value of xy can also be obtained by integrating over time:

$$E[xy] = \frac{1}{T} \int_0^T x(t) y(t) dt \quad (2.32)$$

Again, adopting Parseval's theorem in Eq. (2.7) gives

$$E[xy] = \int_{-\infty}^{\infty} G_{xy}(f) df \quad (2.33)$$

The cross-PSD $G_{xy}(f)$, as mentioned before, is generally complex. However, when the time signals are considered real, the cross-PSD becomes complex symmetric, i.e. $G_{xy}(-f) = \overline{G_{xy}(f)}$. Consequently, it is enough to integrate the real value of $G_{xy}(f)$ along the positive side of the frequency spectrum

$$\int_{-\infty}^{\infty} G_{xy}(f) df = 2 \int_0^{\infty} \text{Re}\{G_{xy}(f)\} df \quad (2.34)$$

Eq. (2.34) suggests that the imaginary part of the cross-PSD does not contribute to the expected value product $E[xy]$. In general, when listing an arbitrary amount of real time signals in a vector $\mathbf{x}(t)$, the expected value of $\mathbf{x} \mathbf{x}^T$ is equal to the integral of the PSD matrix as

$$E[\mathbf{x} \mathbf{x}^T] = 2 \int_0^{\infty} \text{Re}\{\mathbf{G}(f)\} df \quad \left(E[x_i x_j] = 2 \int_0^{\infty} \text{Re}\{G_{ij}(f)\} df \right) \quad (2.35)$$

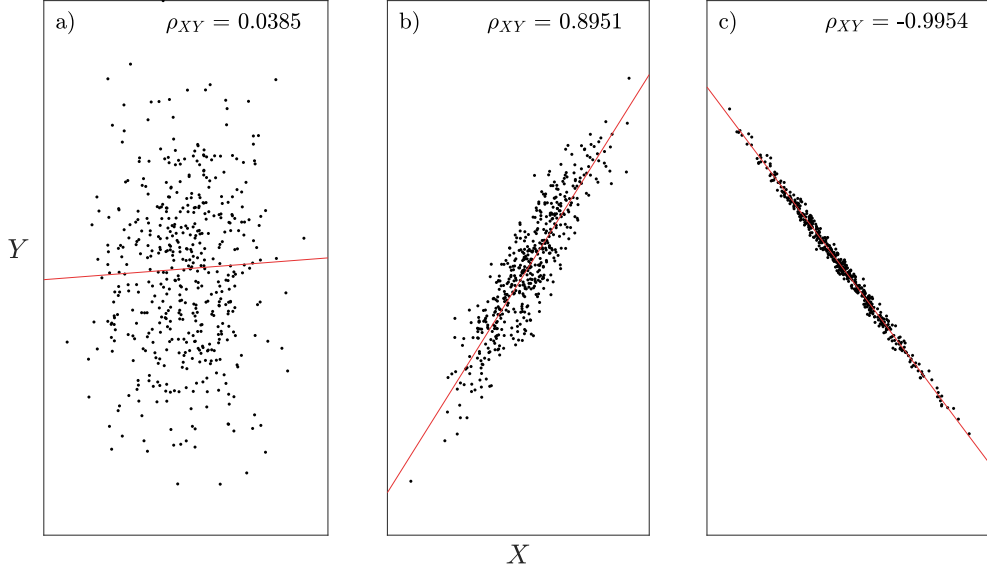


Figure 2.2: Scatter plots for two different stochastic variables X and Y . The red lines represent the optimized linear relation between them. Plot a) illustrates a bad correlation with a great disperse. Plot b) and c) on the other hand illustrates a more correlated relationship where the variables are more clearly linearly dependent.

The covariance can be scaled in order to make it independent of the variable units. The correlation coefficient ρ_{XY} is a convenient and a useful measure. The correlation coefficient can also be seen as a scaled version of the covariance, and it varies between -1 and 1 [26]. The definition is given as

$$\rho_{XY} = \frac{\text{Cov}(x, y)}{\sqrt{\text{Var}(x)\text{Var}(y)}} \quad (|\rho_{XY}| \leq 1) \quad (2.36)$$

A value close to 0 indicates that the variables are uncorrelated, while a value close to -1 or 1 indicates that they are closely correlated. In that case when $\rho_{XY} = \pm 1$, the relation between x and y is linear. For that reason, the correlation factor can also be seen as a measurement on how close two variables are to having a linear relationship.

In many situations, the expected value of $x(t)$ and $y(t)$ are treated to be zero. In this chapter, normal distribution with zero mean will be considered. Consequently, the variance and covariance can be simplified as $\text{Var}(x) = E[x^2]$ and $\text{Cov}(x, y) = E[xy]$, giving the correlation factor

$$\rho_{XY} = \frac{E[xy]}{\sqrt{E[x^2]E[y^2]}} \quad (2.37)$$

Here, the correlation factor reaches ± 1 if x and y are proportional. Indeed, when the signals are proportional with a constant c , i.e. $y = cx$, it follows from Eq. (2.37) that the correlation factor becomes $\rho_{XY} = c/\sqrt{c^2} = c/|c| = \text{sgn}(c) = \pm 1$. Positive correlation factor corresponds to a positive proportionality constant and vice versa. In Fig. 2.2 some scatter plots and their correlation factor are shown as illustration.

The correlation factor can also be written as a symmetric matrix $\boldsymbol{\rho}$, with a set of time signals $x_i(t)$:

$$\rho_{ij} = \frac{E[x_i x_j]}{\sqrt{E[x_i^2]E[x_j^2]}} \quad (2.38)$$

When $|\rho_{ij}| \approx 1$ for all i and j , we can expect all the time signals to be proportional. This is an important condition, especially in case of multiaxial fatigue (see Section 2.9). The time signals can then be expressed as $x_i(t) = c_i g(t)$ where c_i are proportionality constants and $g(t)$ is a dimensionless time signal. Note that the diagonal components in $\boldsymbol{\rho}$ are always 1, since the correlation factor between two identical signals are 1.

The correlation matrix $\boldsymbol{\rho}$ can be calculated directly from the PSD matrix $\mathbf{G}(f)$ using Eq. (2.35). Again, this proves that the PSD carry a lot of information about the signals and their correlations to one another.

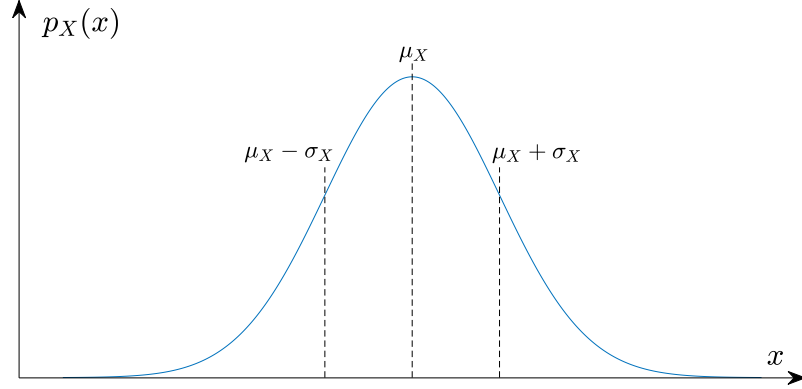


Figure 2.3: Density function of an arbitrary normal distribution $p_X(x)$ with mean μ_X and variance σ_X^2 .

2.4 Random Vibration – Normal Distribution

Random vibration is usually an unwanted feature in materials, and in many cases it is impossible to completely suppress it. The total vibration can be seen as a superposition of many different vibrations that are induced in different ways, for example the vibration from the road or the engine. The total random vibration is therefore simply the sum of many vibrations generated from different sources.

According to the Central limit theorem, the sum of (infinitely) many stochastic variables tends to follow a normal distribution [23]. In that sense, it is convenient to assume that the total random vibration, which usually consist of many different vibrations, also follows a normal distribution. This assumption is usually a good approximation and is commonly used in order to generate random noise. Luckily, the normal distribution has some interesting properties that simplifies the calculations.

The normalized density function $p_X(x)$ of a stochastic variable X that follows a normal distribution is given as

$$p_X(x) = \frac{1}{\sqrt{2\pi\sigma_X^2}} e^{-(x-\mu_X)^2/(2\sigma_X^2)} \quad (2.39)$$

where $\mu_X = E[x]$ denotes the expected value (mean value) and $\sigma_X = \sqrt{\text{Var}(x)}$ the standard deviation (square root of variance). In Fig. 2.3, a general normal distribution is shown as illustration.

2.4.1 Conservation of Normal Distribution

If $x(t)$ represents the displacement of a material point somewhere in a structure, the velocity and acceleration of the same point is simply stated $\dot{x}(t)$ and $\ddot{x}(t)$. If the displacement is following a stationary normal distribution, the velocity and acceleration will also follow a normal distribution. Differentiating a stationary stochastic variable is not possible. However, the time signal $x(t)$ itself is not completely random, it is a continuous function. When sampling the signal at two time instances t_k and $t_{k+1} = t_k + h$, for a given sampling interval h , the sampling data $x_{k+1} = x(t_{k+1})$ will indeed be stationary stochastic and random from $x_k = x(t_k)$, under the assumption that the sampling distance h is sufficiently large. A finite differentiation is thus undefined. However, the derivative can be calculated between two adjacent sampling points, giving $\dot{x}_k \approx (x(t_k + h) - x(t_k))/h$.

Using the fact that the two adjacent random displacement variables X_1 and X_2 are independent, the stochastic variable \dot{X} that denotes the velocity can be obtained as $\dot{X} = \frac{1}{h}(X_1 - X_2) = \frac{1}{h}X_1 - \frac{1}{h}X_2$. This is just a linear combination of two normally distributed variables X_1 and X_2 . One property that makes the normal distribution so tractable is that a linear combination of various independent normal distributed variables will in turn be normal distributed [26]. Hence, using this property, it is understood that the velocity and acceleration will also be normal distributed. In order words, the normal distribution is preserved when differentiating (and when integrating).

The preservation of normal distribution can also be applied to linear systems, since linear systems are nothing else than linear combinations of various differential orders. If the input signal is normally distributed, then a given output signal will be normally distributed as well. This was verified by the authors by studying a cart system with two degrees of freedom presented in Fig. 2.4 with a base excitation $u(t)$. The equation of

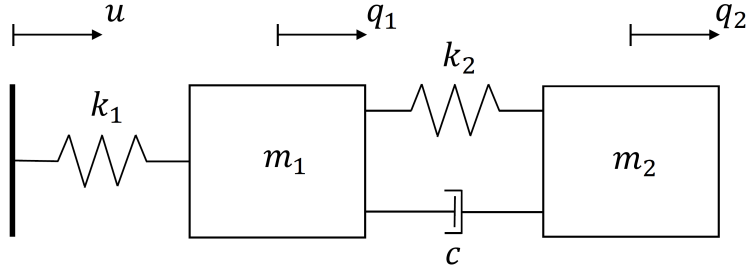


Figure 2.4: Mechanical system with two degrees of freedom, subjected to a base excitation $u(t)$. The mass of the carts are $m_1 = 1$ kg and $m_2 = 2$ kg, while the spring stiffness and damping coefficient are set to $k_1 = 1$ kN/m, $k_2 = 0.8$ kN/m and $c = 10$ Ns/m.

motion for this simple cart system is given as

$$\begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} c & -c \\ -c & c \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} + \begin{bmatrix} k_1 + k_2 & -k_2 \\ -k_2 & k_2 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} = \begin{bmatrix} k_1 \\ 0 \end{bmatrix} u \quad (2.40)$$

which can be stated on the more compact for

$$\mathbf{M} \ddot{\mathbf{q}} + \mathbf{C} \dot{\mathbf{q}} + \mathbf{K} \mathbf{q} = \mathbf{f} \quad (2.41)$$

In Fig. 2.5 the output response $q_2(t)$ is given when applying a normally distributed input signal $u(t)$ with zero mean. According to the histogram plots, the output signal also followed a normally distributed with zero mean just as suggested. The standard deviation, however, is not preserved, but is increased due to resonance of the system.

The time signal $q_2(t)$ seems to be less chaotic compared to the input $u(t)$. The reason for this is because the frequency spectrum is filtered. The frequencies near the systems eigenfrequencies will "survive", whereas the frequencies further away from the eigenfrequencies will be suppressed (see Section 2.5). For a less chaotic signal, it is sometimes crucial to increase the sampling interval in order to receive a stochastic sampling data. Every random continuous time signal appears to be more chaotic when increasing the sampling intervals.

The conservation of normal distribution can also be applied to larger DOF systems. For a continuum body, the finite element method can be used to discretize a body into nodes with certain degrees of freedom. The equation of motion can then be written in a similar fashion as Eq. (2.41).

2.5 Transfer Functions

The equations of motion can be converted to frequency domain using Fourier transform. Consider the equation for a single degree of freedom

$$m\ddot{q} + c\dot{q} + kq = p \quad (2.42)$$

where q is the displacement, m is the mass, c the damping coefficient, k the spring stiffness and p is the applied load. Taking the Fourier transform on both sides of the equation and applying the linear property in Eq. (2.4) and the derivative property in Eq. (2.5) gives

$$\mathcal{F}(m\ddot{x} + c\dot{x} + kx) = \mathcal{F}(p) \implies \hat{x}(f) = \frac{1}{\underbrace{k - m\omega^2 + ic\omega}_{= H(f)}} \hat{p}(f) \implies \hat{x}(f) = H(f) \hat{p}(f) \quad (2.43)$$

where $H(f)$ is the transfer function between the input load $\hat{p}(f)$ and the displacement $\hat{x}(f)$. Note that the transfer function is dependent on the angular frequency ω , but since $\omega = 2\pi f$, we can write it as function of the frequency f .

The transfer function can be obtained for any input and output signal. Generally, if $u(t)$ denotes the input and $x(t)$ the output, the transformation is simply stated as

$$\hat{x}(f) = H(f) \hat{u}(f) \quad (2.44)$$

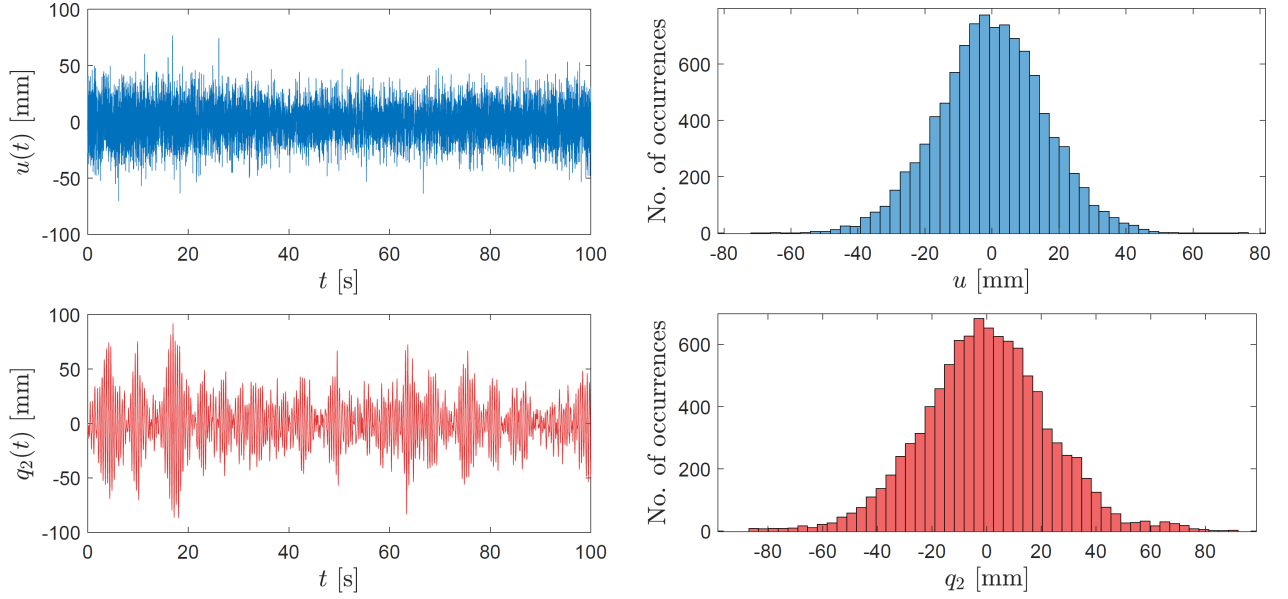


Figure 2.5: The input signal $u(t)$ and output signal $q_2(t)$, and their corresponding histogram plots. The input and output signal are both following a normal distribution just as the histograms suggests.

Multiplying the input frequency signal with the transfer function gives the frequency signal of the output. Hence, the transfer function holds information about amplitude amplification and phase difference between input and output. The absolute value of the transfer function $|H(f)|$ amplifies the input amplitude spectrum $|\hat{u}(f)|$, while the argument $\arg\{H(f)\}$ is simply changing the phase of the input signal. This can be verified by rewriting Eq. (2.44) in polar form which in turn gives

$$\begin{cases} |\hat{x}(f)| = |H(f)| |\hat{u}(f)|, \\ \arg\{\hat{x}(f)\} = \arg\{H(f)\} + \arg\{\hat{u}(f)\} \end{cases} \quad (2.45)$$

The absolute value is also called the magnification factor, since it magnifies or amplifies the input signal [4].

The transfer function is not dependent of the input signal as long as the system is linear. The same transfer function can therefore be used for various input signals, making it a fast and convenient tool when converting between input and output.

The transfer function is complex symmetric if the input and output has a real time signal. This can be proved by using the fact that both $\hat{u}(f)$ and $\hat{x}(f)$ are complex symmetric, giving $H(-f) = H(f)$. Hence, it is enough to just consider the positive side of the frequency spectrum when obtaining the transfer function.

The convolution property in Eq. (2.6) can be used to convert Eq. (2.44) into time domain, which yields

$$x(t) = h * u(t) \quad (2.46)$$

where $h(t)$ is the inverse Fourier transform of the transfer function, i.e. $\mathcal{F}^{-1}(H(f)) = h(t)$.

Consider a sinusoidal input $u(t) = U \cos(2\pi ft + \phi) = \text{Re}\{U e^{i(2\pi ft + \phi)}\}$. Inserting this in Eq. (2.46), it is easy to conclude that

$$x(t) = h * u(t) = |H(f)| U \cos(2\pi ft + \phi + \arg\{H(f)\}) \quad (2.47)$$

The absolute value of the transfer function works as an amplifier, while the argument of the transfer function is changing the phase. Note that the transfer function is dependent of the frequency of the input signal. Hence, the amplitude and phase of the output is directly related to the input frequency f .

Using a complex sinusoidal input signal $u(t) = e^{i(2\pi ft + \phi)}$ with unit amplitude gives an output signal with an amplitude equal to the transfer function itself. This can be verified in the same fashion as in Eq. (2.47). By doing this for various frequencies f , it is possible to map the frequency spectrum of the transfer function $H(f)$. This is usually referred to as sine sweep or swept sine. It should also be mentioned that it is enough to just sweep through the positive side of the frequency spectrum, since the transfer function is complex symmetric.

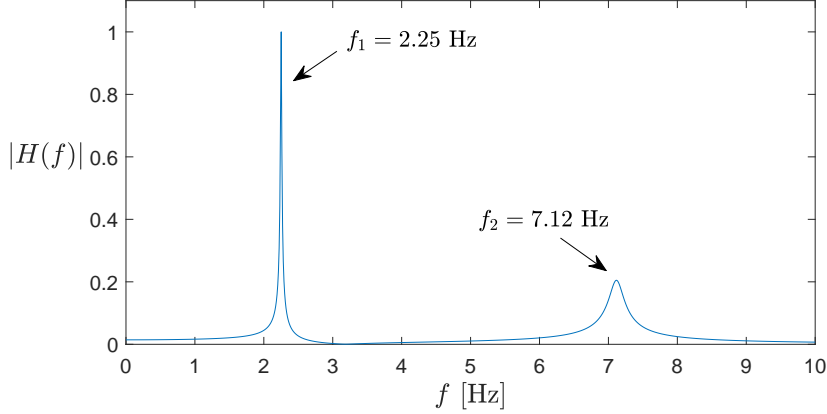


Figure 2.6: Normalized Transfer function for a two degrees of freedom cart system with eigenfrequencies $f_1 = 2.25$ Hz and $f_2 = 7.12$ Hz.

2.5.1 Filtering the Input Signal

Consider the spectrum transformation in Eq. (2.45). The amplitude of the transfer function $|H(f)|$ can be seen as a filter which suppress some parts of the input frequency spectrum when $|H(f)|$ is low, and amplify others when $|H(f)|$ is high. Because of this, certain frequencies are allowed to pass the filter more easily than others.

Usually for structural systems, the absolute value of the transfer functions has peaks located close to the eigenfrequencies, which means that the input amplitude spectrum is going to be amplified in a greater extent at frequencies located closer to the systems eigenfrequencies. An illustrative example of a transfer function is presented in Fig. 2.6.

If the input amplitude spectrum contains a high concentration at the eigenfrequencies, then it will excite a more violent vibration of the structure. On the contrary, if the structure does not contain eigenfrequencies close to frequencies in the input signal, the vibration of the structure will be smaller. The latter one is of course usually the most desirable option. However, it is sometimes hard to control the input signal – vibration will always be present. But the eigenfrequencies of the system can be modified by changing the geometry of the structure, or by adding control masses. Lower eigenfrequencies is usually more dangerous, since they will involve more movement of the structure. Indeed, when plotting $|H(f)|$ one commonly finds highest peaks at the first eigenfrequencies.

The damping of the system is also very crucial when considering the peaks of $|H(f)|$. If the damping is large, the vibration will be suppressed, especially at the eigenfrequencies. The system is losing energy due to friction and heat, and the peak in $|H(f)|$ will be less distinct. Higher damping results in smaller and wider peaks, which will give a wideband signal as output. On the contrary, for smaller damping the peaks are taller and more narrowed. The output is then a narrowband signal.

2.5.2 Transfer Functions and PSD:s

The transfer function can also be used to convert an input PSD to an output PSD. Substituting Eq. (2.44) in Eq. (2.11) gives

$$G_{xx}(f) = |H(f)|^2 G_{uu}(f) \quad (2.48)$$

where $|H(f)|^2$ denote PSD transfer function. To avoid confusion, the output PSD is from now on denoted S while the input PSD is denoted G . Also, the transfer functions are marked by indices $H_{xu}(f)$ where the left and right indices represent the output and input, respectively. Therefore Eq. (2.48) can be rewritten as

$$S_{xx}(f) = |H_{xu}(f)|^2 G_{uu}(f) \quad (2.49)$$

The transfer function for cross-PSD:s can be derived in a similar fashion. Assume that $\hat{x}(f) = H_{xu}(f) \hat{u}(f)$ and $\hat{y}(f) = H_{yv}(f) \hat{v}(f)$. By adopting Eq. (2.20), it is possible to state the output cross-PSD between x and y as

$$S_{xy}(f) = H_{xu}(f) \overline{H_{yv}(f)} G_{uv}(f) \quad (2.50)$$

where G_{uv} is the cross-PSD between the input signals u and v , and $H_{xu}(f) \overline{H_{yv}(f)}$ is the transfer function for cross-PSD:s. For two input and two output signals, there are a total of $2 \times 2 = 4$ transfer functions and $2^2 \times 2^2 = 16$ transfer functions between every input and output PSD. Similarly, for two input and three output signals there are $2 \times 3 = 6$ transfer functions and $2^2 \times 3^2 = 36$ PSD combinations.

Generally, if there are an arbitrary set of input signals $\{u_i\}_{i=1}^m$ and an arbitrary set of output signals $\{x_i\}_{i=1}^n$, the transformation between input and output PSD:s can be written in index notation as

$$S_{ij}(f) = H_{ik}(f) \overline{H_{jl}(f)} G_{kl}(f) \quad (2.51)$$

where $S_{ij}(f) = \frac{1}{T} \hat{x}_i(f) \overline{\hat{x}_j(f)}$ is the output PSD matrix, $G_{kl}(f) = \frac{1}{T} \hat{u}_k(f) \overline{\hat{u}_l(f)}$ is the input PSD matrix, and $H_{ij}(f)$ denotes the transfer function between input \hat{u}_j and output \hat{x}_i . The same equation can also be written in matrix notation as

$$\mathbf{S}(f) = \left(\mathbf{H}(f) \otimes \overline{\mathbf{H}(f)} \right) : \mathbf{G}(f) \quad (2.52)$$

where \otimes represent a modified tensor product defined as $\mathbf{A} \otimes \mathbf{B} = A_{ik} B_{jl} \mathbf{e}_i \otimes \mathbf{e}_j \otimes \mathbf{e}_k \otimes \mathbf{e}_l$, and $:$ is the double contraction operator.

2.6 Transfer Functions in Finite Element Model

The equation of motion for a linear discretized finite element system can generally be written as

$$\mathbf{M} \ddot{\mathbf{q}} + \mathbf{C} \dot{\mathbf{q}} + \mathbf{K} \mathbf{q} = \mathbf{f} \quad (2.53)$$

Here, \mathbf{M} , \mathbf{C} and \mathbf{K} are the global mass, damping and stiffness matrices, respectively. The vector \mathbf{q} stores all the nodal variables (degrees of freedom) in each node, and \mathbf{f} is the load vector including discretizations of both surface and volume loads [7]. Each node has in general 6 degrees of freedom; displacement in 3 directions, and rotation in 3 directions as illustrated in Fig. 2.7.

When treating discretized FE-systems, it is of great importance to separate the free (F) and constrained (C) degrees of freedom. The nodal variable vector can therefore be partitioned into a free and constrained DOF:s as $\mathbf{q}^T = [\mathbf{q}_F^T \quad \mathbf{q}_C^T]$. The equation of motion is then given as

$$\begin{bmatrix} \mathbf{M}_{FF} & \mathbf{M}_{FC} \\ \mathbf{M}_{CF} & \mathbf{M}_{CC} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}}_F \\ \ddot{\mathbf{q}}_C \end{bmatrix} + \begin{bmatrix} \mathbf{C}_{FF} & \mathbf{C}_{FC} \\ \mathbf{C}_{CF} & \mathbf{C}_{CC} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_F \\ \dot{\mathbf{q}}_C \end{bmatrix} + \begin{bmatrix} \mathbf{K}_{FF} & \mathbf{K}_{FC} \\ \mathbf{K}_{CF} & \mathbf{K}_{CC} \end{bmatrix} \begin{bmatrix} \mathbf{q}_F \\ \mathbf{q}_C \end{bmatrix} = \begin{bmatrix} \mathbf{f}_F \\ \mathbf{f}_C \end{bmatrix} \quad (2.54)$$

where \mathbf{f}_F is the free (known) load vector and \mathbf{f}_C is the constrained load vector that carry information about the reaction load at the constrained DOF:s. The matrix in Eq. (2.54) can be separated into two equations; one to solve the free degrees of freedom \mathbf{q}_F and another one to solve the constrained load vector \mathbf{f}_C . The equation of motion of the free DOF:s can be stated as

$$\mathbf{M}_{FF} \ddot{\mathbf{q}}_F + \mathbf{C}_{FF} \dot{\mathbf{q}}_F + \mathbf{K}_{FF} \mathbf{q}_F = \underbrace{\mathbf{f}_F - \mathbf{M}_{FC} \ddot{\mathbf{q}}_C - \mathbf{C}_{FC} \dot{\mathbf{q}}_C - \mathbf{K}_{FC} \mathbf{q}_C}_{=\mathbf{f}'_F} \quad (2.55)$$

The vector \mathbf{f}'_F denotes the modified load that contains the contribution from the constrained motion. Applying Fourier transform on Eq. (2.55), yield the frequency response

$$\hat{\mathbf{q}}_F(f) = \mathbf{H}_{FF}(f) \hat{\mathbf{f}}_F(f) + \mathbf{H}_{FC}(f) \hat{\mathbf{q}}_C(f) \quad (2.56)$$

with the transfer functions $\mathbf{H}_{FF}(f)$ and $\mathbf{H}_{FC}(f)$ defined as

$$\begin{cases} \mathbf{H}_{FF}(f) = (\mathbf{K}_{FF} - \omega^2 \mathbf{M}_{FF} + i\omega \mathbf{C}_{FF})^{-1} \\ \mathbf{H}_{FC}(f) = -\mathbf{H}_{FF}(f) (\mathbf{K}_{FC} - \omega^2 \mathbf{M}_{FC} + i\omega \mathbf{C}_{FC}) \end{cases} \quad (2.57)$$

When analyzing durability of the components which are subjected to an oscillating input load, it is usually of interest to examine the stress profile in the material. Using the linear property from kinematics and constitutive relation, it is possible to calculate the stress response directly once the DOF vector $\mathbf{q}^T = [\mathbf{q}_F^T \quad \mathbf{q}_C^T]$ is obtained. Similarly, we can construct a transfer function between the inputs $(\mathbf{f}_F, \mathbf{q}_C)$ and the output stress $\underline{\sigma}$ at a selected material point, where $\underline{\sigma}$ represents the Voigt notation of the stress tensor $\boldsymbol{\sigma}$.

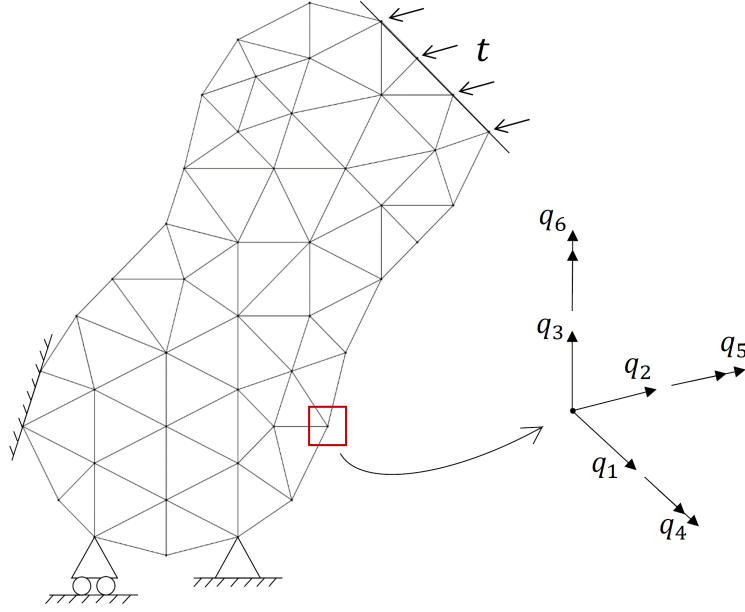


Figure 2.7: Example structure with natural and essential boundary conditions, where \mathbf{t} stands for traction load. The local degrees of freedom for an arbitrary node is shown.

Note that the transfer functions are complex as long as the system is damped. If not, there exist a set of real angular frequencies such that the matrix $\mathbf{H}_{FF}(f)$ becomes non-invertible. In that case the transfer functions in Eq. (2.57) are undefined. This happens when the frequency is equal to the system's eigenfrequencies (see Section 2.7), giving infinitely large peaks when plotting the transfer functions. However, if the damping is present, the matrix is invertible for all real frequencies, and the peaks in the transfer functions does no longer tends to infinity. In fact, larger damping will decrease the "height" of the peaks and make them broader.

2.6.1 Multiaxial Base Excitation – Prescribed Motion

It is sometimes convenient to present the load as prescribed motion $\mathbf{q}_C(t)$. If so, the free load vector is simply zero, $\mathbf{f}_F = \mathbf{0}$, giving the stress response

$$\hat{\mathbf{q}}_F(f) = \mathbf{H}_{FC}(f) \hat{\mathbf{q}}_C(f) \quad (2.58)$$

Usually, the prescribed motion is induced from another body, which will henceforth be call a master body. The component that is attached to the master body will be denoted slave body. The mass of the slave body is assumed to be negligible compared to the master body, which suggests that the master body is unaffected by the motion of the attached slave body. However, in practice this is not completely true. But as long as the mass of the master body is sufficiently large and its rigidity is high, the assumption is considered valid. In this project, the air dryer bracket component works as a slave body and is mounted on a truck chassis frame that represent the master body.

The interface surface between the slave and master body can be modelled in various ways. If the interface surface is sufficiently small, it is sometimes convenient to consider a rigid interface. This limitation is however not always practical, but for small interfaces and for stiff master bodies, this approximation works quite well, and is used in the air dryer example.

The translational vibration is usually more present than the vibration from the rotational DOF's. Henceforth, the rotational DOF:s at the interface surface are set to zero. This limitation is also considered for the bracket component. The interface surface is then allowed to translate in x, y, and z-direction but not allowed to rotate, which yields

$$\mathbf{q}_C(t) = \mathbf{R}_C \mathbf{u}(t) \quad (2.59)$$

Here \mathbf{R}_C reduction matrix containing zeros and ones, and reduce the constrained degrees of freedom from $\mathbf{q}_C(t)$ to $\mathbf{u}(t)$. The vector $\mathbf{u}(t)$ is the displacement vector representing the translative motion of the rigid interface surface.

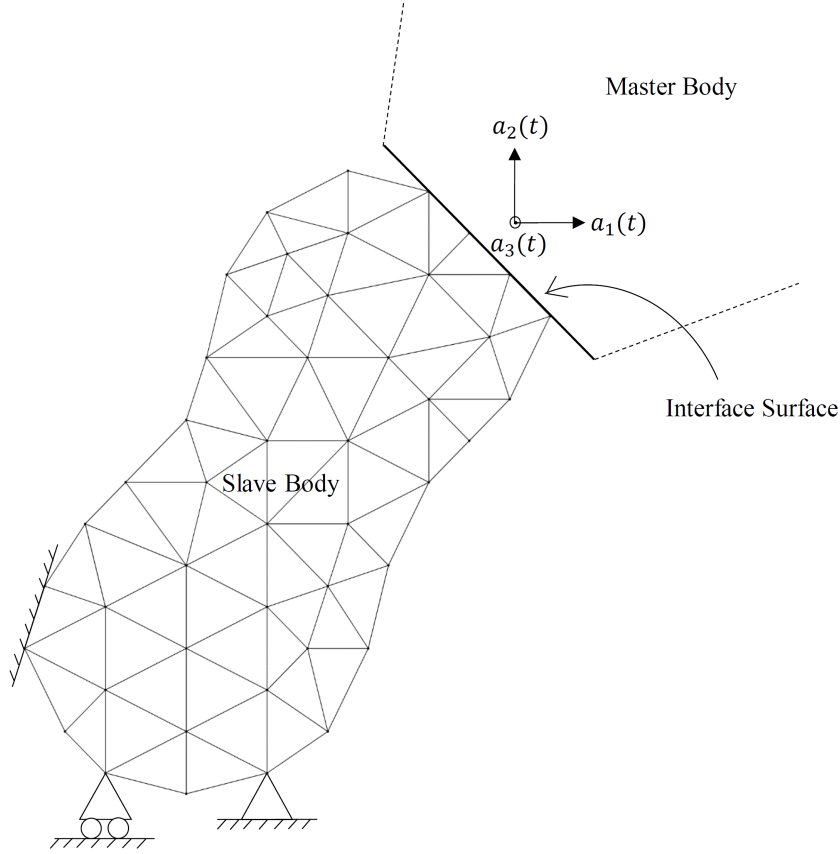


Figure 2.8: Example structure represented as a slave body attached to a larger master body via an interface. The master body induces motion to the slave body.

The prescribed motion of the interface can also be represented as velocity $\mathbf{v}(t) = \dot{\mathbf{u}}(t)$ or acceleration $\mathbf{a}(t) = \ddot{\mathbf{u}}(t)$. The latter one is commonly used and was also used for the air dryer bracket component in this project. By adopting the Fourier relation $\hat{\mathbf{a}}(f) = -(2\pi f)^2 \hat{\mathbf{u}}(f)$ and the reduction in Eq. (2.59), the stress response from the input acceleration can be written as

$$\hat{\underline{\sigma}}(f) = \mathbf{H}(f) \hat{\mathbf{a}}(f) \quad (2.60)$$

There are now 3 input signals and in general 6 output signals (6 stress components), giving a 6×3 transfer function matrix. The output stress PSD matrix $\mathbf{S}(f) = \frac{1}{T} \hat{\underline{\sigma}}(f) \hat{\underline{\sigma}}(f)^\dagger$ will then be a 6×6 matrix obtained from Eq. (2.23). Similarly, the input acceleration PSD matrix can be written as $\mathbf{G}(f) = \frac{1}{T} \hat{\mathbf{a}}(f) \hat{\mathbf{a}}(f)^\dagger$, giving a 3×3 matrix. The relation between $\mathbf{G}(f)$ and $\mathbf{S}(f)$ is then stated in the same way as in Eq. (2.51) and Eq. (2.52).

2.6.2 Coupled and Uncoupled Interfaces

For some systems, there exists multiple interfaces that are exciting the component. The interfaces could be coupled, meaning that numerous interfaces are attached to the same master body. If the master body is considered stiff or if the interfaces are located close to one other, it is possible to couple them into a single interface motion. However, if the interfaces are attached to different master bodies, or if the interfaces are attached to a somewhat more flexible master body, the interfaces are no longer coupled. The motion presented in Eq. (2.59) can then be expanded as

$$\mathbf{q}_C(t) = \sum_{k=1}^n \mathbf{R}_{C,k} \mathbf{u}_k(t) = [\mathbf{R}_{C,1} \ \dots \ \mathbf{R}_{C,n}] \begin{bmatrix} \mathbf{u}_1(t) \\ \vdots \\ \mathbf{u}_n(t) \end{bmatrix} = \mathbf{R}_C \mathbf{u}(t) \quad (2.61)$$

where $\mathbf{u}_k(t)$ denotes the translative motion of the k :th interface. In this project, only one interface was taken into consideration. For more complex models with multiple interfaces, the number of input signals are highly increased, giving a larger transfer function matrix, but the approach is identical.

2.7 Modal Analysis

Modal analysis can be used to reveal interesting properties of a structure. The intension of modal analysis is to find the eigenfrequencies and its corresponding eigenmodes of a system.

For multi-DOF system, without exciting a motion to the structure, it is possible to encounter a state when all DOF:s are oscillating at the same frequency and same phase. The frequencies for which this can occur are termed eigenfrequencies. This critical state will often create severe damage since the vibration of the structure at those frequencies is often large. For a continuous system there are infinitely many eigenfrequencies, but the lowest frequencies are usually of most interest. A discretized FE system, however, does not cover all eigenfrequencies, but represent the lowest ones more accurately.

When an external load has a frequency content close to an eigenfrequency of the system, the vibration of the whole structure becomes excited, causing larger stresses in the structure which can lead to a reduced endurance and shorter fatigue life.

The equation of motion of the free DOF:s when no external load or prescribed motion is present is given as

$$\mathbf{M}_{FF} \ddot{\mathbf{q}}_F + \mathbf{C}_{FF} \dot{\mathbf{q}}_F + \mathbf{K}_{FF} \mathbf{q}_F = \mathbf{0} \quad (2.62)$$

For an undamped system, $\mathbf{C}_{FF} = \mathbf{0}$, the eigenmodes Φ_i and corresponding eigenfrequencies ω_i can be solved from the equation

$$(\mathbf{K}_{FF} - \omega_i^2 \mathbf{M}_{FF}) \Phi_i = \mathbf{0} \quad \Longleftrightarrow \quad \omega_i^2 = \frac{\Phi_i^T \mathbf{K}_{FF} \Phi_i}{\Phi_i^T \mathbf{M}_{FF} \Phi_i} \quad (2.63)$$

Now, we want to find frequencies ω_i such that there exists a non-trivial solution to Eq. (2.63), which occurs when the determinant is equal to zero:

$$\det(\mathbf{K}_{FF} - \omega_i^2 \mathbf{M}_{FF}) = 0 \quad \text{or} \quad \omega_i^2 = \text{eig}(\mathbf{K}_{FF} \mathbf{M}_{FF}^{-1}) \quad (2.64)$$

The eigenmodes Φ_i have in general an unknown magnitude, but are usually normalized by the mass matrix as

$$\Phi_i^T \mathbf{M}_{FF} \Phi_i = 1 \quad (2.65)$$

Also, due to linearity, the stress tensor has a corresponding eigenmode. By analyzing the stress modes, it is possible to map the stress distribution for different types of eigenfrequencies, and locate interesting hotspots where the oscillating stresses have large amplitudes. The magnitude of stress is however not of importance. Instead the stress distribution or the amplitude difference is something that should be taken into consideration.

In case of damping, the problem becomes more complicated. The damping matrix \mathbf{C}_{FF} is usually unknown, especially for FE models. The damping can be modelled in different ways, for example Rayleigh damping for which the global damping matrix can be written as linear combination of mass and stiffness matrix. In this case, however, the so-called modal damping method was used with a damping coefficient of $\zeta_i = 3\%$ for all eigenmodes i . Typical values of the modal damping coefficient lies in the range $1\% \leq \zeta_i \leq 10\%$ [7]. Each eigenmode has its own defined damping and is dependent on both material and geometry of the component. However, for simplicity, the authors assumed constant modal damping for all eigenmodes. Modal damping can be defined as

$$\Phi_i^T \mathbf{C}_{FF} \Phi_i = 2\zeta_i \omega_i \quad (2.66)$$

where Φ_i and ω_i again are the eigenmodes and eigenfrequencies for the undamped system, and Φ_i must be normalized according to Eq. (2.65).

When considering rigid interfaces, the eigenfrequencies of the structure will change. The motion of the interface DOF:s are limited which will result in a reduction $\mathbf{q}_C(t) = \mathbf{R}_C \mathbf{u}(t)$, discussed in Section 2.6.1.

2.8 Fatigue Damage

When a specimen is subjected to a repeated cyclic loading above some magnitude, it will eventually break even if the stress is far below the material yield limit. The oscillating load causes microscopical damage in the

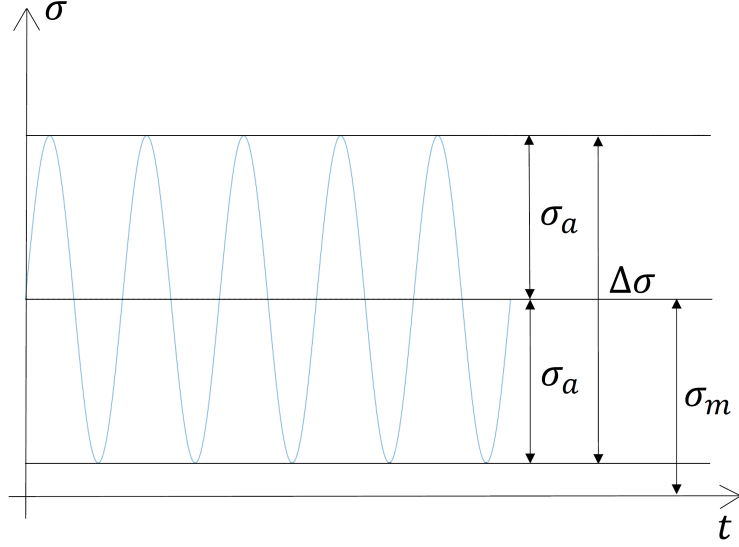


Figure 2.9: Cyclic loading for a given stress component σ . The amplitude stress is denoted σ_a , and the stress range as $\Delta\sigma = 2\sigma_a$. The mid stress σ_m should not be confused by the mean stress over time, since the cyclic loading does not need to have a sinusoidal nature. Instead, the mid stress is defined as the mean of the maximum and minimum stress.

material, and the damage will successively grow until the specimen reach mechanical failure. The failure is caused by microscopical cracks that are increasing in size.

This kind of failure is referred to as fatigue, which is a common problem for components that are subjected to vibrations. It is therefore of great importance to develop mathematical models that can be used to predict how much a component can endure before it breaks. Even though there are some well-established empirical models today, their purpose is very limited and should be used with great caution.

For simplicity, it is common to map the number of cycles to failure when applying constant amplitude stressing as shown in Fig. 2.9. The stress is then shifting between a maximum and minimum value σ_{\max} and σ_{\min} , respectively. The amplitude stress σ_a , stress range $\Delta\sigma$ and mid stress σ_m of the cyclic loading is then given as

$$\sigma_a = \frac{\sigma_{\max} - \sigma_{\min}}{2}, \quad \Delta\sigma = \sigma_{\max} - \sigma_{\min} = 2\sigma_a, \quad \sigma_m = \frac{\sigma_{\max} + \sigma_{\min}}{2} \quad (2.67)$$

In order to define the cyclic loading, we need 2 sets of parameters, usually (σ_a, σ_m) , $(\Delta\sigma, \sigma_m)$ or $(\sigma_{\min}, \sigma_{\max})$ [9]. Note that the constant amplitude stress history does not necessarily have to be shaped as sinusoidal wave as in Fig. 2.9, but can take any form as long as the amplitude is constant. In the special case when the mid stress is equal to zero, $\sigma_m = 0$, the load is called completely reversible. The amplitude and stress range are then denoted with a index r as σ_{ar} and $\Delta\sigma_r$.

The number of cycles to failure N_f is directly related to the properties of the cyclic loading. If the amplitude of the load is increased, we should expect fewer cycles before failure occurs. In case of reversible loading, the number of cycles to failure can be plotted versus the stress range $\Delta\sigma_r$ or amplitude stress σ_{ar} . This curve is usually referred to as a Wöhler curve or S-N curve and explains the durability property of a certain material. The S-N curve, however, is bounded to the given test specimen. The geometry of the specimen and the applied load also have an impact on the result. Hence, the S-N curve does not truly represent a material property – other effects are included as well. Also, some test specimen may have different defects in the material structure, which will affect the fatigue life. The great uncertainty of the S-N curve is a big problem, especially when applying the S-N curve on other components. Despite the uncertainty, the S-N curve can be used as a tool to approximate the fatigue life.

The data N_f and $\Delta\sigma_r$ can usually be fitted with a straight line in a log-log diagram, meaning that the relation is exponential:

$$\log(N_f) = \log(C) - k \log(\Delta\sigma_r) \quad \Longleftrightarrow \quad N_f = C \Delta\sigma_r^{-k} \quad (2.68)$$

Here, C and k are fitting constants that can be seen as material parameters. In Fig. 2.10 a typical reversible S-N curve is shown. The relation in Eq. (2.68) is only applicable for high cycle fatigue, and is not valid for N_f

below a certain limit N'_f (low cycle fatigue region). This corresponds to a stress range larger than $\Delta\sigma'_r$. The largest stress range possible is equal to two times the ultimate strength σ_u of the material, and the low cycle fatigue region is usually approximated with a straight line. For sufficiently large number of cycles, when N_f exceed the fatigue limit N_e , the relation is just a horizontal straight line as shown in Fig. 2.10. This corresponds to a stress range less than $\Delta\sigma_{er}$. Stress ranges below the fatigue limit will not contribute to fatigue damage.

Note that N'_f , N_e , $\Delta\sigma'_r$ and $\Delta\sigma_{er}$ are material parameters that can be used to obtain the fitting constants C and k . Using the logarithmic property in Eq. (2.68) yields the relation

$$k = \frac{\log(N_e/N'_f)}{\log(\Delta\sigma'_r/\Delta\sigma_{er})}, \quad C = \left(\frac{\Delta\sigma'_r \log(N_e)}{\Delta\sigma_{er} \log(N'_f)} \right)^{1/\log(\Delta\sigma'_r/\Delta\sigma_{er})} \quad (2.69)$$

The constant k is dimensionless and the unit of C is equal to the unit of stress to the power of k . Usually we consider the unit MPa for the stress which gives the unit $(\text{MPa})^k$ for C .

The upper and lower stress range limits can be estimated as $\Delta\sigma'_r = 2m'\sigma_u$ and $\Delta\sigma_{er} = 2m\sigma_u$. Here m and m' are reduction factors dependent on the load type, geometry, surface finish and other external conditions. The ratio between the stress range limits is then equal to the ratio between the reduction factors as $\Delta\sigma'_r/\Delta\sigma_{er} = m'/m$.

The reversible S-N curve is only applicable if the mid stress is equal to zero. In case when a non-zero mid stress is present, there are varieties of methods to construct an equivalent reversible stress range $\Delta\sigma_r$ from the non-reversible stress range $\Delta\sigma$ and mid stress σ_m . This can be stated in a general fashion as

$$\Delta\sigma_r = g(\Delta\sigma, \sigma_m) \quad (2.70)$$

where $g(\bullet, \bullet)$ is an arbitrary function that includes the mid stress effect. The function should of course fulfill that $g(\Delta\sigma, 0) = \Delta\sigma = \Delta\sigma_r$. The Morrow equation and the Smith, Watson and Topper (SWT) equation are commonly used, just to name a few [9]. The positive thing about constructing a relation in Eq. (2.70) is that the reversible S-N curve can be used for multiple mid stresses, without having to construct a S-N curve for each mid stress value. Inserting Eq. (2.70) into Eq. (2.68) gives the more general relation

$$N_f = C g(\Delta\sigma, \sigma_m)^{-k} \quad (2.71)$$

In this project, the mid stress effects were not taken into consideration. The stress range will contribute considerably more to fatigue damage, especially when treating stress histories with zero mean.

2.8.1 Accumulated Damage and Rainflow Count

The damage can be measured by counting the number of cycles N of a constant amplitude stressing $\Delta\sigma_r$ and divide it by the number of cycles to failure as

$$D = \frac{N}{N_f} = \frac{N}{C \Delta\sigma_r^{-k}} \quad (2.72)$$

where D represent the damage. When N exceed N_f , mechanical failure occurs and the damage will then exceed 1. It is also common to use safety factors which will restrict the failure condition. It is sometimes convenient to assume failure when D exceed 0.3 or 0.5. However, no safety factors were used in this project.

When accumulating the damage from various stress ranges it is common to add all the damages to get the total damage. This is called Palmgren-Miner rule and can be stated as

$$D = \sum_i \frac{N_i}{N_{f,i}} = \sum_i \frac{N_i}{C \Delta\sigma_{r,i}^{-k}} \quad (2.73)$$

where N_i is the counted cycles for a given stress range $\Delta\sigma_{r,i}$ with number of cycles to failure $N_{f,i}$. The Palmgren-Miner rule is the most common method to account for accumulated damage, probably thanks to its simplicity, but the rule has also been proven to be a reliable method for various cases [9]. The rule does not account for the history of the stress signal, i.e. the order in which the loads are applied. In reality, the order of the load cycles do matter and will affect the fatigue damage. There are methods that includes the cyclic order as well as other affects, but no other known method can compete with the simplicity of Palmgren-Miner rule.

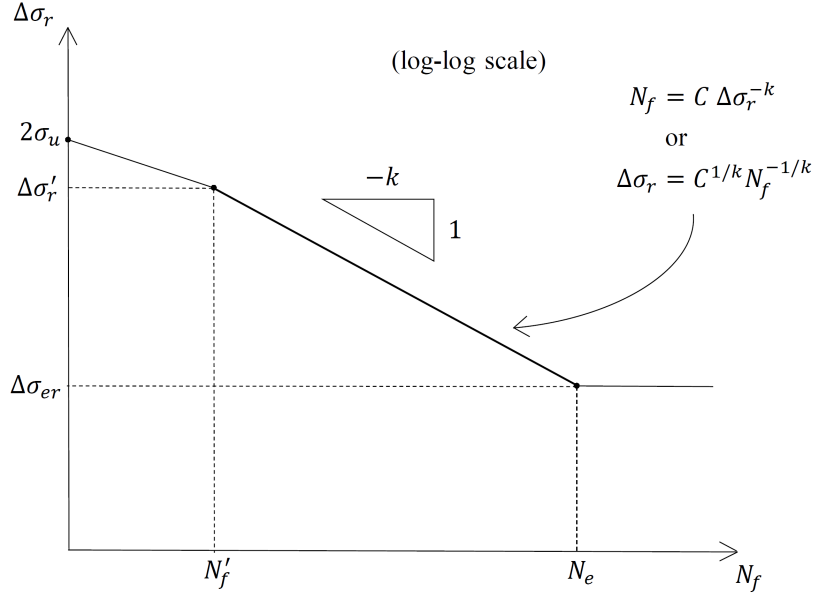


Figure 2.10: Reversible S-N curve with stress range versus number of cycles to failure. Both axes are logarithmic scaled. The linear property in Eq. (2.68) is defined between $N'_f \leq N_f \leq N_e$ and $\Delta\sigma_{er} \leq \Delta\sigma_r \leq \Delta\sigma'_r$, where N'_f is typically 10^3 and N_e is usually around 10^6 [16]. The ultimate strength σ_u is the largest stress that can be reached before fracture. For $N_f < N'_f$ the high cycle fatigue theory no longer applies. Usually, this region is approximated with a straight line from $\Delta\sigma'_r$ to $2\sigma_u$. However, for low cycle fatigue theory, this is not entirely true. For $N_f > N_e$, or when $\Delta\sigma_r < \Delta\sigma_{er}$, the cycles will no longer contribute to any fatigue damage.

For irregular stress signals the stress range and mid stress are varying from cycle to cycle. In case of random vibration, the stress components are highly irregular and does not follow a constant amplitude stressing. In that case, we need to distinguish each cycle from the signal. This can be done by adopting the rainflow count algorithm. There are various versions of rainflow count, some that are counting whole cycles and other that count halfcycles.

One way to perform rainflow count is to simply scan through the stress history for increasing time using 3 adjacent data points A, B and C. An example of this is shown in Fig. 2.11. The three data points represent a true cycle if $|\sigma_A - \sigma_B| \leq |\sigma_B - \sigma_C|$, where σ_A , σ_B and σ_C are the stress values at point A, B and C, respectively. The stress range, amplitude stress and mid stress for this particular cycle is then given as $\Delta\sigma = |\sigma_A - \sigma_B|$, $\sigma_a = |\sigma_A - \sigma_B|/2 = \Delta\sigma/2$ and $\sigma_m = (\sigma_A + \sigma_B)/2$. The point A and B are then discarded, and the scanning continues in a procedure that runs until all the cycles are found. If $|\sigma_A - \sigma_B| > |\sigma_B - \sigma_C|$, the set does not represent a true cycle, and the scanning proceeds without discarding A and B. The algorithm restart the scanning as long as there are cycles left to count, and stops when only one or two data points remains. Before applying this algorithm, it is convenient to reduce the stress signal so that it only contains local peaks and valleys. All points that does not represent a local maximum or minimum are unnecessary.

This algorithm is very fast and does only need local information in order to count a cycle (see MATLAB code in Section A.1.3) [10]. However, there are other rainflow count algorithms that are taking the whole stress signal into consideration when counting cycles, giving a more accurate result, yet a bit slower computational time. In this project, a halfcycle count algorithm was selected. For more information about this algorithm see Section A.1.1.

The cycle data from the rainflow count can later be used to accumulate the damage using Palmgren-Miner rule. However, the accumulation can be performed for one cycle at the time giving $N_i = 1$ or $N_i = 0.5$ in Eq. (2.73) if whole or halfcycles are counted.

If the length of the load is T , then it is possible to predict how long time T_{life} it takes to failure under the assumption that the same load sequence is used repeatedly [13]. This is referred to as fatigue life (in time) and is given as

$$\frac{T_{\text{life}}}{T} = \frac{1}{D} \quad \Longleftrightarrow \quad T_{\text{life}} = \frac{T}{D} \quad (2.74)$$

under the assumption that $D = 1$ represents the fatigue failure. The fatigue life T_{life} is however only valid if the

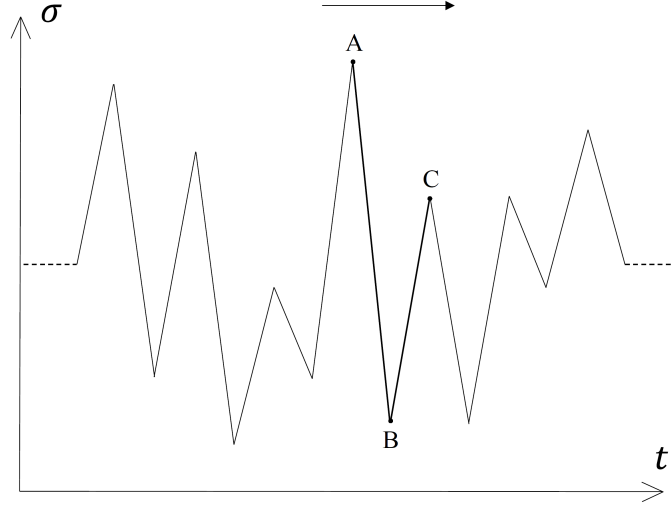


Figure 2.11: Stress signal with only the local maximums and minimum points present (peaks and valleys). The rainflow count algorithm is performed by scanning the stress signal from left to right by looking at three adjacent data points A, B and C for each step.

load is persistently acting. In practice, the load is not always present, and so the service time of a component is usually higher. Further, in practice the same load sequence is not usually repeated – every load sequence is unique.

The counted number of cycles can be saved in a histogram between certain stress range intervals $\Delta\sigma_{r,j} \leq \Delta\sigma_r < \Delta\sigma_{r,j+1}$ and mid stress intervals $\sigma_{m,j} \leq \sigma_m < \sigma_{m,j+1}$. It is then possible to visualize the number of cycles around certain stress ranges and mid stresses. In Fig. 2.12 a 2D histogram example is present with stress range and mid stress axes. The histogram can tell us a lot about the cycle spread and the distribution of $\Delta\sigma$ and σ_m .

2.9 Multiaxial Fatigue

In a general stress state, there are multiple stress components to consider when estimating fatigue damage. The theory behind this is called multiaxial fatigue and plays an important role when estimating the fatigue damage for multiaxial stress states. However, because of the complexity, this is a big challenge, especially if the stress signals are out-of-phase or non-proportional with respect to each other. Instead of constructing a variety of S-N curves for many specific multiaxial stress cases, it is more convenient to use the same uniaxial S-N curves even for multiaxial stresses. But in order to do so, it is essential to accumulate the contribution for each stress component. There are many established methods, and some involves finding an equivalent stress component that represents all the other components.

The special case when all stress components are in-phase can be stated as

$$\boldsymbol{\sigma}(t) = \mathbf{b} + \mathbf{c} g(t), \quad \sigma_{ij}(t) = b_{ij} + c_{ij} g(t) \quad (2.75)$$

where $\boldsymbol{\sigma}(t) = \sigma_{ij}(t) \mathbf{e}_i \otimes \mathbf{e}_j$ is the stress tensor, $\mathbf{b} = b_{ij} \mathbf{e}_i \otimes \mathbf{e}_j$ and $\mathbf{c} = c_{ij} \mathbf{e}_i \otimes \mathbf{e}_j$ are constant tensors, and $g(t)$ is a scalar time dependent function. This is usually termed in-phase loading and is an easy case of multiaxial fatigue since all peaks and valleys occur at the same instant of time. In case of zero mean stress, the constant \mathbf{b} is equal to zero. This case is called proportional loading [11]:

$$\boldsymbol{\sigma}(t) = \mathbf{c} g(t), \quad \sigma_{ij}(t) = c_{ij} g(t) \quad (2.76)$$

The stress components then have a proportional relation. So how can we prove that the stress components are proportional? The easiest choice would be to make a scatter plot between each component in the same way as in Fig. 2.2 and see if they follow a linear relationship, or estimate the correlation factor between each and every stress component using Eq. (2.38). Luckily, the correlation factors can also be estimated from the PSD matrix of the stress components $\mathbf{S}(f) = \frac{1}{T} \hat{\boldsymbol{\sigma}}(f) \hat{\boldsymbol{\sigma}}(f)^\dagger$ by using Eq. (2.35), where $\underline{\sigma}$ is the Voigt notation of the

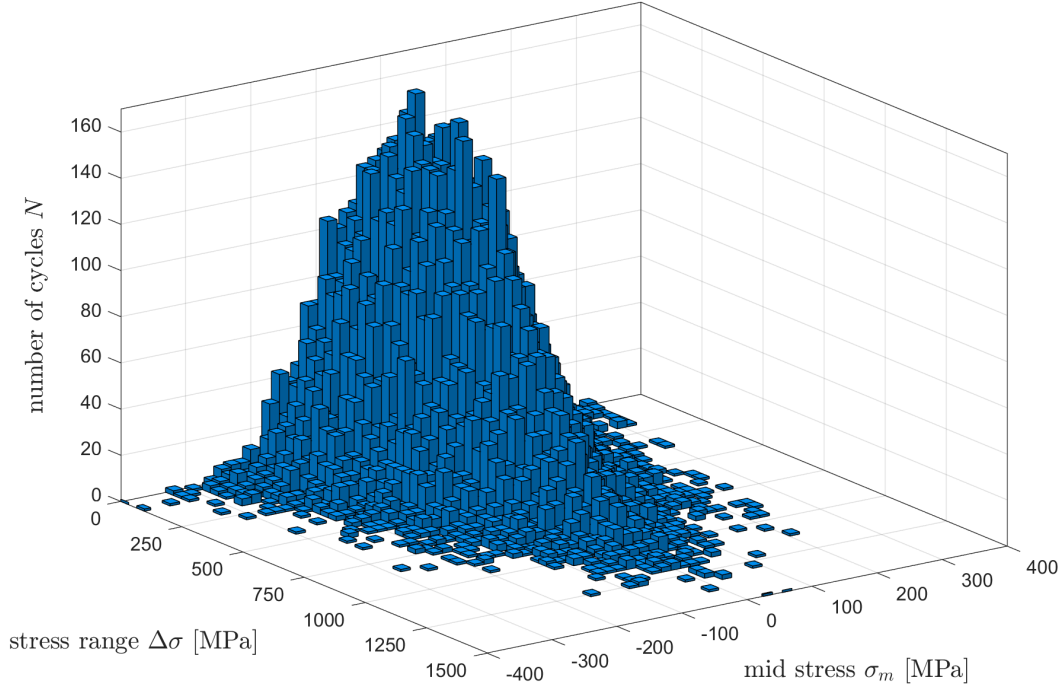


Figure 2.12: Histogram plot for stress range $\Delta\sigma$ and mid stress σ_m . The histogram was derived using the rainflow count data from a white noise stress signal.

stress tensor. Even though the stress components do not truly follow proportional loading, it is possible to approximate a proportional relation if the correlation factor is sufficiently close to ± 1 . In Fig. 2.2 optimized linear relations are obtained using least square, which can be used to approximate a proportional relationship $\boldsymbol{\sigma}(t) \approx \mathbf{c}g(t)$.

Note however that it is possible to reduce the stress tensor so that it only contains normal stresses (no shear stresses), simply by changing the coordinate system. The normal stresses are then called principal stresses σ_k with the corresponding principal directions \mathbf{n}_k , where $k = 1, 2, 3$. The principal stresses are the eigenvalues of the stress tensor $\boldsymbol{\sigma}$ and the principal direction are the eigenvectors to the eigenvalue problem

$$\boldsymbol{\sigma} \mathbf{n}_k = \sigma_k \mathbf{n}_k, \quad k = 1, 2, 3 \quad (2.77)$$

Here \mathbf{n}_k are orthonormal with respect to each other since the stress tensor is symmetric; i.e. $\mathbf{n}_i \mathbf{n}_j^T = \delta_{ij}$, where δ_{ij} is the Kronecker delta tensor. If the coordinate system has the basis vectors \mathbf{n}_k , the stress tensor becomes diagonal as

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} \quad (2.78)$$

The principal directions might change depending on the stress components, i.e. $\mathbf{n}_k = \mathbf{n}_k(t)$. This is a problem since we cannot observe the stress state in different coordinate system at every instant of time while tracking the fatigue damage. For tracking it must be assumed that the principal directions are constant. This only occurs if the stress tensor have proportional stress components, i.e. $\boldsymbol{\sigma}(t) = \mathbf{c}g(t)$. If the stress state is proportional, the principal stresses are also proportional and the principal directions have constant directions. If the motion is controlled by one static input load (or multiple input loads that are proportional with respect to each other), the stress state is indeed proportional, and fatigue damage estimation will be more accurate. However, this project also consider multiaxial base excitation, which could deliver out-of-phase stresses.

2.9.1 Rainflow Count – Multiaxial Stress

In case of out-of-phase loading, the stress components have peaks and valleys at different positions in time, making it hard to extract the cycles from multiple stress signals. This is a major issue, since we cannot really distinguish the cycles from different stress components. In order to count the cycles, one need to construct an equivalent stress (or strain) signal that can be used to extract the cycles. But even then, one should be aware of "hidden" peaks between the sample points, and it does not guarantee that a correct number of cycles will be counted.

In case of multiple stress signals the cycles have to be extracted by adopting a more sophisticated rainflow count algorithm. After doing some research, the authors decided to use Wang and Brown's rainflow count. The Wang and Brown's rainflow count uses a von Mises stress (strain) in order to extract the cycles. For more information about Wang and Brown's algorithm, the reader is referred to Section 3.6.2.

In case of proportional loading, all peaks and valleys will occur at the same instant of time for all stress components, making it possible to extract the cycles using the original rainflow count method proposed in Section 2.8.1. In fact, since $\sigma(t) = cg(t)$, it is possible to perform rainflow count on the signal $g(t)$ alone, and later multiply the cycle range Δg and mid value g_m with c_{ij} to receive the stress range and mid stress for all the stress components σ_{ij} individually.

2.9.2 Equivalent Stress – Dang Van Criterion

There are various methods to account multiaxial fatigue, and many of them regard an equivalent stress scalar that is representing the stress tensor. To reduce a complex multiaxial stress state into a scalar representation might not seem appropriate, but is essential in order to apply a uniaxial S-N curve when estimating the fatigue.

The Dang Van criterion was used in this project for the time domain approach, and can be seen as a critical plane criterion. The equivalent Dang Van stress for a given cycle in the interval $t \in \Delta t_c$ is given as

$$\sigma_{dv}(t) = \tau_{tr,a}(t) + c_{dv}\sigma_h(t) \quad (2.79)$$

where $\tau_{tr,a}(t)$ is the "varying amplitude" of the Tresca shear stress and $\sigma_h(t)$ is the hydrostatic stress. The constant c_{dv} is a fatigue parameter, usually derived from different uniaxial properties. The Dang Van criterion can now be formulated as

$$\sigma_{dv,max} = \max_{t \in \Delta t_c} [\sigma_{dv}(t)] \geq \sigma_{edv} \quad (2.80)$$

where $\sigma_{dv,max}$ is the maximum Dang Van stress in the given cycle interval $t \in \Delta t_c$ and σ_{edv} is the corresponding fatigue limit, i.e. if $\sigma_{dv,max}$ is less than σ_{edv} no fatigue damage will occur (in theory).

The varying amplitude of the Tresca shear stress and the hydrostatic stress is defined as

$$\begin{aligned} \tau_{tr,a}(t) &= \max \{ |\sigma_{1,a}^d(t) - \sigma_{2,a}^d(t)|, |\sigma_{1,a}^d(t) - \sigma_{3,a}^d(t)|, |\sigma_{2,a}^d(t) - \sigma_{3,a}^d(t)| \} / 2 \\ \sigma_h(t) &= \sigma_{kk}(t) / 3 = (\sigma_{11}(t) + \sigma_{22}(t) + \sigma_{33}(t)) / 3 \end{aligned} \quad (2.81)$$

where $\sigma_{k,a}^d(t)$ are the principal stresses of the varying amplitude deviatoric stress tensor $\sigma_{ij,a}^d(t)$. The operator $\max\{\bullet, \bullet, \bullet\}$ pick the largest value of the arguments for all $t \in \Delta t_c$. The varying amplitude of the deviatoric stress tensor can be calculated as

$$\begin{aligned} \sigma_{ij,a}^d(t) &= \sigma_{ij}^d(t) - \sigma_{ij,mid}^d \\ \sigma_{ij}^d(t) &= \sigma_{ij}(t) - \sigma_h(t) \delta_{ij} \end{aligned} \quad (2.82)$$

where $\sigma_{ij}^d(t)$ is the deviatoric stress tensor and $\sigma_{ij,mid}^d$ is a constant deviatoric mid stress tensor defined so that the amplitudes of $\sigma_{ij,a}^d(t)$ becomes as small as possible for all $t \in \Delta t_c$ [11]. This can be done by solving a optimization problem; find $\sigma_{ij,mid}^d$ such that $(\sigma_{ij,a}^d(t) \sigma_{ij,a}^d(t))^{1/2}$ becomes minimized for all $t \in \Delta t_c$. Sometimes it is convenient to minimize the maximum value of $(\sigma_{ij,a}^d(t) \sigma_{ij,a}^d(t))^{1/2}$. The problem can then be stated as

$$L(\sigma_{ij,mid}^d) \leq L(v_{ij}) \quad \text{or} \quad L(\sigma_{ij,mid}^d) = \min L(v_{ij}) \quad (2.83)$$

where

$$L(v_{ij}) = \max_{t \in \Delta t_c} \sqrt{(\sigma_{ij}^d(t) - v_{ij})(\sigma_{ij}^d(t) - v_{ij})} \quad (2.84)$$

However, the optimization may be very time consuming since it needs to be used for each extracted cycle path. To overcome this problem the authors decided to estimate the mid value simply by calculating the mean value of all data points inside the cycle path.

2.9.3 Equivalent S-N Curve

When applying an equivalent stress, one usually has to construct a corresponding equivalent S-N curve. The equivalent S-N curve can be seen as a generalization and should therefore be applicable for uniaxial stress as well. In case of Dang Van criterion, the parameters c_{dv} and σ_{edv} can be solved so that they coincide with the fatigue limit of the uniaxial S-N curve, and other fatigue data [11].

Consider the uniaxial constant reversible loading $\sigma_{11}(t) = \frac{\Delta\sigma_r}{2} \sin(t)$. The maximum Dang Van stress for reversible loading according to the definition in Eq. (2.80) is then

$$\sigma_{dv,max} = \left(\frac{1}{4} + \frac{c_{dv}}{6} \right) \Delta\sigma_r \iff \Delta\sigma_r = \left(\frac{1}{4} + \frac{c_{dv}}{6} \right)^{-1} \sigma_{dv,max} \quad (2.85)$$

This suggests that the number of cycles to failure versus maximum reversible Dang Van stress can be obtained by substituting Eq. (2.85) into Eq. (2.68), which yields

$$N_f = C \Delta\sigma_r^{-k} = C \left(\frac{1}{4} + \frac{c_{dv}}{6} \right)^k \sigma_{dv,max}^{-k} \quad (2.86)$$

The equivalent S-N curve then has the same slope as the uniaxial one, but shifted. The constant c_{dv} can be obtained by combining another uniaxial loading case such as reversible shear stress. In this study, however, the available fatigue property data of the selected material were limited, and so the Dang Van constant could not be estimated. Typical values lies around 1/3, and the Dang Van parameter was therefore set to $c_{dv} = 1/3$.

Note that the equivalent S-N curve is derived so that it coincides with the uniaxial one. This does not necessarily mean that the model will represent all multiaxial cases equally well. Modelling a more accurate equivalent S-N curve is of course much harder to accomplish and requires more fatigue parameters.

In case of multiaxial stress (or uniaxial stress for that matter), $\sigma_{dv,max}$ is calculated for each defined cycle using the procedure explained in Eq. 2.9.2 (easier for proportional loading). Further, the corresponding uniaxial reversible stress range $\Delta\sigma_r$ can be obtained from Eq. (2.85). Finally, the uniaxial S-N curve is used to find the number of cycles to failure N_f and the corresponding damage for each cycle.

The maximum Dang Van stress can be negative in certain cases. The equivalent S-N curve in Eq. (2.86) is however not defined for negative values, since this could evaluate to negative or complex fatigue life. If we assume that negative and positive mid stresses contribute to the same amount of damage (they usually don't) or if the mid stress effect is negligible, consider replacing $\sigma_{dv,max}$ by $|\sigma_{dv,max}|$. For a random stress signal with zero mean, most of the extracted maximum Dang Van values will indeed be positive, but for a more compressive stress profiles this could lead to large errors.

2.10 Random Fatigue Analysis – Spectral Methods

Random fatigue analysis is a powerful and convenient tool in case of random vibration. With the help of probability theory, it is possible to estimate fatigue life without having to perform rainflow count explicitly. Instead, semi-empirical methods can be applied using the moment of area properties from the PSD data [15]. However, one must know the density function of the stress signal before any theory can be applied with success. In Section 2.4 and 2.4.1, it has been discussed that the stress signal is usually normally distributed in case of random vibration due to the central limit theorem and the fact that normal distribution is preserved for linear systems when differentiating.

Consider $X = X(t)$ to be a stochastic variable that is representing the stress signal. Also, consider the first and second derivative as $\dot{X} = \dot{X}(t)$ and $\ddot{X} = \ddot{X}(t)$. The stochastic processes are assumed to be Gaussian stationary. The density function $p_X(x)$ of X is therefore defined according to Eq. (2.39). The mean and standard deviation is denoted μ_X and σ_X , respectively.

It can be shown using that $\dot{X} = 0$ and $\ddot{X} < 0$ for a local maximum (peak), that the expected number of peaks per unit time E_p , expected number of zero up crossing per unit time E_0 , and the irregularity factor α_2 are equal to [19]

$$E_p = \frac{\sigma_{\ddot{X}}}{2\pi\sigma_{\dot{X}}}, \quad E_0 = \frac{\sigma_{\dot{X}}}{2\pi\sigma_X}, \quad \alpha_2 = \frac{E_0}{E_p} = \frac{\sigma_{\dot{X}}^2}{\sigma_{\ddot{X}}\sigma_X} \quad (2.87)$$

where $\sigma_{\dot{X}}$ and $\sigma_{\ddot{X}}$ are standard deviation of the first and second derivative of X . The second order irregularity factor $\alpha_2 \in [0, 1]$, already defined in Eq. (2.18), can be seen as a bandwidth measurement, where $\alpha_2 = 0$ stands for extreme wideband and $\alpha_2 = 1$ extreme narrowband.

The values E_0 , E_p and α_2 can also be obtained from frequency domain using Eq. (2.16), (2.17), and (2.18). The PSD of a time derivative signal $\dot{x}(t)$ and $\ddot{x}(t)$ can be derived by using the Fourier property in Eq. (2.5), which yields the relation

$$\begin{aligned} G_{\dot{x}\dot{x}}(f) &= (2\pi f)^2 G_{xx}(f) \\ G_{\ddot{x}\ddot{x}}(f) &= (2\pi f)^4 G_{xx}(f) \end{aligned} \quad (2.88)$$

The standard deviations of X , \dot{X} and \ddot{X} in case of zero mean can therefore be calculated using the relation defined in Eq. (2.29) which gives

$$\begin{aligned} \sigma_X^2 &= E[x^2] = \int_{-\infty}^{\infty} G_{xx}(f) df \triangleq m_0 \\ \sigma_{\dot{X}}^2 &= E[\dot{x}^2] = (2\pi)^2 \int_{-\infty}^{\infty} G_{xx}(f) f^2 df \triangleq (2\pi)^2 m_2 \\ \sigma_{\ddot{X}}^2 &= E[\ddot{x}^2] = (2\pi)^4 \int_{-\infty}^{\infty} G_{xx}(f) f^4 df \triangleq (2\pi)^4 m_4 \\ E_p &= \frac{\sigma_{\ddot{X}}}{2\pi\sigma_{\dot{X}}} = \sqrt{\frac{\sigma_{\ddot{X}}^2}{(2\pi)^2\sigma_{\dot{X}}^2}} = \sqrt{\frac{m_4}{m_2}} \\ E_0 &= \frac{\sigma_{\dot{X}}}{2\pi\sigma_X} = \sqrt{\frac{\sigma_{\dot{X}}^2}{(2\pi)^2\sigma_X^2}} = \sqrt{\frac{m_2}{m_0}} \\ \alpha_2 &= \frac{E_0}{E_p} = \frac{m_2}{\sqrt{m_4 m_0}} \end{aligned} \quad (2.89)$$

The time derivative relation is also applicable on cross-PSD:s. This is important if you want to measure the correlation between the time derivative of two signals $x(t)$ and $y(t)$. This is really powerful since we can estimate the correlation between all derivatives if the cross-PSD:s are known. Below follows a general relation of the expected value of $x^{(p)} y^{(q)}$, where $p = 0, 1, 2 \dots$ and $q = 0, 1, 2, \dots$ are the time derivative orders. Using the same approach as in Eq. (2.88), it is easy to prove that

$$G_{x^{(p)} y^{(q)}}(f) = (2\pi f)^{p+q} G_{xy}(f) \quad (2.90)$$

which in turn gives

$$E[x^{(p)} y^{(q)}] = (2\pi)^{p+q} \int_{-\infty}^{\infty} G_{xy}(f) f^{p+q} df \quad (2.91)$$

Noteworthy, there are two interesting cases; when $p + q$ is even or odd. In the special case when $x \equiv y$ and $p + q$ is odd, the expected value is always zero due to the fact that $G_{xx}(f)$ is real and symmetric. This is a very important property since it suggests that $x(t)$ and $\dot{x}(t)$ have no correlation – they are independent. The same goes for $\dot{x}(t)$ and $\ddot{x}(t)$. Hence, they are uncoupled and not parts of a common density function.

On the contrary, the correlation between $x(t)$ and $\ddot{x}(t)$ is generally non-zero since $p + q = 2$ is even. In other words, we should expect a statistical dependency between X and \ddot{X} , meaning that they are parts of a coupled joint density function. The density function that includes all three stochastic variables X , \dot{X} and \ddot{X} can then be stated as

$$p_{X\dot{X}\ddot{X}}(u, v, w) = p_{X\ddot{X}}(u, w) p_{\dot{X}}(v) \quad (2.92)$$

2.10.1 Peak and Stress Range Distribution

With the distribution given by Eq. (2.92) it is possible to derive the stress peak density function $p_Y(u)$, where Y denote the stochastic variable representing stress peak values. Using the properties for peaks $\dot{X} = 0$ and $\ddot{X} < 0$, the probability for a peak below the stress value u to occur is stated as $P_Y(u) = P(X < u, \dot{X} = 0, \ddot{X} < 0)$, where $P_Y(u)$ is the cumulated density function of Y . Using probability theory [19], it can be shown that

$$p_Y(u) = p_X(u) \left(\sqrt{1 - \alpha_2^2} \exp\left(-\frac{\alpha_2^2(u - \mu_X)^2}{2\sigma_X^2(1 - \alpha_2^2)}\right) + \frac{\alpha_2(u - \mu_X)}{\sigma_X} \int_{-\infty}^{\alpha_2(u - \mu_X)/(\sigma_X \sqrt{1 - \alpha_2^2})} e^{-v^2/2} dv \right) \quad (2.93)$$

where α_2 is the irregularity factor stated in Eq. (2.89), $p_X(u)$ is the density function of the stress X , and μ_X and σ_X are the mean and standard deviation of X . In case of wideband excitation, the irregularity factor α_2 tends towards 0 which gives $p_Y(u) = p_X(u)$. The probability distribution for stress peaks is then equal to the probability distribution of stress values. Instead, for a narrowband, the irregularity factor tends to 1, which results in $p_Y(u) = \sqrt{2\pi}(u - \mu_X) p_X(u)/\sigma_X$.

The density function for stress range can be derived from the stress peak value Y . If $Z > 0$ represent the stochastic variable for stress range, it can be concluded that $Z = 2|Y - \mu_X|$. The cumulative density function of Z is then given as $P_Z(u) = P(Z < u) = P_Y(\mu_X + u/2) - P_Y(\mu_X - u/2)$, where $P_Y(u)$ is the cumulative density function for the stress peak value Y . Differentiating the cumulative density function gives the stress range density functions as

$$p_Z(u) = [p_Y(\mu_X + u/2) + p_Y(\mu_X - u/2)]/2 \quad u \geq 0 \quad (2.94)$$

In case of wideband excitation, $\alpha_2 = 0$, the stress range density function represents a normal distribution. On the other hand, for narrowband excitation $\alpha_2 = 1$ the density function for stress range is reduced to a Rayleigh distribution as

$$p_Z(u) = \frac{u}{4\sigma_X^2} \exp\left(-\frac{u^2}{8\sigma_X^2}\right) \quad u \geq 0 \quad (2.95)$$

According to Eq. (2.72) the expected damage for one cycle can be obtained as $E[D] = E[1/N_f] = E[Z^k]/C$, where Z is the stochastic variable corresponding to $\Delta\sigma_r$. In case of narrowband $\alpha_2 = 1$, the expected damage for one cycle can be obtained using Eq. (2.95) as

$$E[D] = \frac{1}{C} \int_0^\infty u^k p_Z(u) du = \frac{2^{3k/2} \sigma_X^k}{C} \Gamma(k/2 + 1) \quad (2.96)$$

where $\Gamma(\bullet)$ is the gamma function [19]. The total damage is then obtained by multiplying $E[D]$ by the expected number of peaks $E_p T$, where E_p is the expected number of peaks per unit time and T is the total length of the time signal. However, since $\alpha_2 = 1$ we know for sure that $E_p = E_0$. The expected fatigue life T_{life} in Eq. (2.74) is then obtained as

$$T_{\text{life}} = \frac{T}{E[D]E_0T} = \frac{C}{2^{3k/2} m_0^{(k-1)/2} m_2^{1/2} \Gamma(k/2 + 1)} \quad (2.97)$$

The fatigue life can therefore be obtained easily using Eq. (2.97) under the assumption that the narrowband stress signal is normally distributed. Only the zero and second order moment of area needs to be computed from the stress PSD in this special case. However, the same approach can be used to find the fatigue life for any α_2 , which then requires m_0 , m_2 and m_4 .

The number of cycles per stress range is given as $E_p T p_Z(u)$, and by integrating small stress range intervals Δu , it is possible to construct a histogram. The number of cycles N_i per interval $u_i - \Delta u/2 < u_i < u_i + \Delta u/2$ can then be approximated as

$$N_i = E_p T p_Z(u_i) \Delta u \quad (2.98)$$

It is also possible to derive a density function for mid stress σ_m . Studies have been made on how to include mid stress correction for spectral methods [22], but since mid stress effects are not considered in this work, no further explanation will be made on that topic.

2.10.2 Dirlik's Empirical Formula

There are various methods to construct a density function for stress range. Some are specialized for certain cases like Rayleigh distribution for narrowband excitation, and others are more general. Dirlik proposed a density function derived using Monte Carlo simulation and has proven to be easy and reliable for arbitrary narrow or wideband stress signals that are normal distributed. Dirlik's formula outperform many other available methods and is well suited for estimating the fatigue life of FE-structures, but does require that the load has a stationary normal distribution [28]. Nevertheless, the Dirlik's formula does not consider mid stress effects, making it well suited for zero mean loads. Dirlik's density function combines one exponential distribution and two Rayleigh distribution as follows [8]:

$$p_Z(u) = \frac{D_1}{2\sqrt{m_0}Q} \exp\left(-\frac{u}{2\sqrt{m_0}Q}\right) + \frac{D_2 u}{4m_0 R^2} \exp\left(-\frac{u^2}{8m_0 R^2}\right) + \frac{D_3 u}{4m_0} \exp\left(-\frac{u^2}{8m_0}\right) \quad (2.99)$$

where

$$\begin{aligned} D_1 &= \frac{2(x_m - \gamma^2)}{1 + \gamma^2}, \quad D_2 = \frac{1 - \gamma - D_1 + D_1^2}{1 - R}, \quad D_3 = 1 - D_1 - D_2, \quad Q = \frac{1.25(\gamma - D_3 - D_2 R)}{D_1}, \\ R &= \frac{\gamma - x_m - D_1^2}{1 - \gamma - D_1 + D_1^2}, \quad \gamma = \frac{m_2}{\sqrt{m_0 m_4}}, \quad x_m = \frac{m_1}{m_0} \sqrt{\frac{m_2}{m_4}} \end{aligned} \quad (2.100)$$

The only parameters needed to construct the Dirlik distribution are four moment of areas from the stress PSD; m_0 , m_1 , m_2 and m_4 . This makes it a very simple yet powerful method.

The expected damage per cycle $E[D]$ can be obtained in the same way as in Eq. (2.96) giving

$$E[D] = \frac{m_0^{k/2} [2^k Q^k D_1 \Gamma(k+1) + 2^{3k/2} (|R|^k D_2 + D_3) \Gamma(k/2+1)]}{C} \quad (2.101)$$

where $\Gamma(\bullet)$ is again the gamma function. The expected fatigue life is then obtained by taking the time length T divided by the total damage $E[D]E_p T$ as

$$T_{\text{life}} = \frac{C m_2^{1/2}}{m_4^{1/2} m_0^{k/2} [2^k Q^k D_1 \Gamma(k+1) + 2^{3k/2} (|R|^k D_2 + D_3) \Gamma(k/2+1)]} \quad (2.102)$$

Just as for any stress range distribution, it is possible to construct a histogram from the Dirlik's distribution using the method explained in Eq. (2.98).

2.10.3 Account for Fatigue Limit

As mentioned in Section 2.8, the stress range does not contribute to fatigue damage if it stays below the fatigue limit $\Delta\sigma_{er}$. Also, if the stress range exceed $\Delta\sigma'_r$, we enter the low cycle fatigue domain which can be approximated with a straight line (see Fig. 2.10). However, for a high cycle fatigue problem, the stress range should not exceed the upper limit. The density function then tends to zero when the stress range approach $\Delta\sigma'_r$. If we account for the lower fatigue limit in the calculations the integration of the density function should not be made from zero, since every stress range below the fatigue limit should not produce any damage. However, cycles with stress range below $\Delta\sigma_{er}$ will contribute very little to the overall fatigue damage, due to the fact that the S-N curve is decreasing exponentially. Hence, we still receive good results when integrating from 0 to ∞ as in Eq. (2.96) and Eq. (2.101).

2.10.4 Equivalent PSD Stress – EVMS

In case of multiaxial stress, one has to include multiple PSD:s when estimating fatigue. There are various methods for doing so, but perhaps the most common approach is to define a uniaxial equivalent PSD stress that represents the PSD of all stress components all together. The purpose with an equivalent stress PSD is to reduce the multiaxial stress state into a uniaxial stress state. A well-known method is using equivalent von Mises stress (EVMS) that was proposed by Preumont and co-workers. It has proven to be reliable and easy to apply. This method, however, should not be confused with the von Mises stress in time domain. Instead, it can be seen as a frequency domain version of the von Mises stress.

Before presenting the EVMS, consider the definition of von Mises stress $\sigma_{vM}(t)$ which can be presented as follows:

$$\begin{aligned} \sigma_{vM}^2 &= \frac{1}{2}(\sigma_{11} - \sigma_{22})^2 + \frac{1}{2}(\sigma_{22} - \sigma_{33})^2 + \frac{1}{2}(\sigma_{11} - \sigma_{33})^2 + 3(\sigma_{12}^2 + \sigma_{23}^2 + \sigma_{13}^2) \\ &= \underbrace{[\sigma_{11} \quad \sigma_{22} \quad \sigma_{33} \quad \sigma_{12} \quad \sigma_{23} \quad \sigma_{13}]}_{=\underline{\sigma}^T} \underbrace{\begin{bmatrix} 1 & -1/2 & -1/2 & 0 & 0 & 0 \\ -1/2 & 1 & -1/2 & 0 & 0 & 0 \\ -1/2 & -1/2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}}_{=\mathbf{Q}} \underbrace{\begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ \sigma_{12} \\ \sigma_{23} \\ \sigma_{13} \end{bmatrix}}_{=\underline{\sigma}} \\ &= \underline{\sigma}^T \mathbf{Q} \underline{\sigma} = \mathbf{Q} : (\underline{\sigma} \underline{\sigma}^T) \end{aligned} \quad (2.103)$$

Here $\underline{\sigma}$ is the stress tensor in Voigt notation, and \mathbf{Q} is a constant symmetric matrix referred to as the Q-matrix [27]. In many literatures this relation is commonly stated with a trace operator $\text{trace}\{\mathbf{Q} \cdot \underline{\sigma} \underline{\sigma}^T\}$ rather than the double dot product as in Eq. (2.103). These expressions are equivalent. The Q-matrix is presented in Eq. (2.103) for a general multiaxial stress state with 6 components. In this project however, only three local stresses σ_{11} , σ_{22} and σ_{12} are considered, giving a 3×3 Q-matrix.

The problem with the von Mises stress is that the relation to the stress components is non-linear. Also, the von Mises stress is only positive, which is a problem since the zero mean property no longer apply. The mean square of the von Mises stress, however, can be written as a linear combination of mean square of the other stress components as $E[\sigma_{\text{vM}}^2] = \mathbf{Q} : E[\underline{\sigma} \underline{\sigma}^T]$. Adopting Eq. (2.35) gives

$$E[\sigma_{\text{vM}}^2] = \mathbf{Q} : \int_{-\infty}^{\infty} \mathbf{S}(f) df \quad (2.104)$$

where $\mathbf{S}(f) = \frac{1}{T} \hat{\underline{\sigma}}(f) \hat{\underline{\sigma}}(f)^\dagger$ is the PSD matrix of the stress components. By introducing an equivalent PSD $S_{\text{vM}}(f)$ that has the mean square $E[\sigma_{\text{vM}}^2]$, it is possible to rewrite Eq. (2.104) as

$$\int_{-\infty}^{\infty} S_{\text{vM}}(f) df = \int_{-\infty}^{\infty} \mathbf{Q} : \mathbf{S}(f) df \quad (2.105)$$

which suggest that

$$S_{\text{vM}}(f) = \mathbf{Q} : \mathbf{S}(f) \quad (2.106)$$

This equation defines the EVMS of the stress PSD matrix $\mathbf{S}(f)$ [25]. Note that $S_{\text{vM}}(f)$ is not the PSD of the von Mises signal $\sigma_{\text{vM}}(t)$, however it shares the same mean square.

Once the EVMS is estimated, we can use the moment of area to receive important information to estimate fatigue damage. The moment of area of the EVMS can be obtained using the definition in Eq. (2.12). Also, if we consider the complex symmetry property in Eq. (2.34), the moment of area of the EVMS is given as [6]

$$m_{\text{vM},k} = 2 \int_0^{\infty} \text{Re}\{S_{\text{vM}}(f)\} f^k df = \mathbf{Q} : \mathbf{m}_k \quad (2.107)$$

where \mathbf{m}_k is the k:th moment of inertia matrix, defined as

$$\mathbf{m}_k = 2 \int_0^{\infty} \text{Re}\{\mathbf{S}(f)\} f^k df \quad (2.108)$$

The k:th moment of area of the EVMS $m_{\text{vM},k}$ can later be used to construct a density function, tentatively Dirlik's formula in Eq. (2.99). Note that \mathbf{m}_k is always real since the cross-PSD is complex symmetric. Also, the moment of area matrix is symmetric due to the fact that the PSD stress matrix fulfill $\mathbf{S}(f) = \mathbf{S}(f)^\dagger$.

As mentioned in Section 2.5.2, is possible to model a transfer function matrix from a PSD input matrix $\mathbf{G}(f)$ to a PSD output matrix $\mathbf{S}(f)$. By adopting Eq. (2.52) we can model a transfer function that takes us from input $\mathbf{G}(f)$ to the EVMS output S_{vM} . This can be derived using index notation, which gives

$$S_{\text{vM}}(f) = \mathbf{H}_{\text{vM}}(f) : \mathbf{G}(f) \quad (2.109)$$

where $\mathbf{H}_{\text{vM}}(f) = \mathbf{H}(f)^T \mathbf{Q} \overline{\mathbf{H}(f)}$ is the EVMS transfer function.

Before applying EVMS one must be aware of its limitations. The EVMS assumes that the S-N curve for shear stress has the same slope as the S-N curve of normal stress. Further, the fatigue limit ratio between the shear and normal stress should always be equal to $\sqrt{3}$. If $\Delta\tau_{er}$ denotes the reversible fatigue limit of shear stress and $\Delta\sigma_{er}$ the reversible fatigue limit for normal stress, and if k_τ is the slope of the shear S-N curve and k_σ is the slope of the normal S-N curve, the requirements can be stated as follows [2]

$$k_\sigma = k_\tau, \quad \Delta\sigma_{er}/\Delta\tau_{er} = \sqrt{3} \quad (2.110)$$

These requirements are approximately fulfilled for many isotropic materials, but not all. There are many suggested modifications of Preumont's EVMS that are less limited such as Braccisi's modification. However, these topics will not be encountered in this project.

3 Methodology

This chapter includes the methodologies and strategies that are incorporated in order to achieve the end results. A brief description on how to apply the theory is made. Application of the concepts that best mimics the real-world scenario is up to an engineer to decide based on his/her knowledge, competence and experience. Although there are plenty of other techniques to solve the random vibration induced durability of a component, the methodology regards the problem definition and its limitations stated before.

The chapter contains figures, tables and flow charts that guides the reader to understand how this thesis work was done. Two widely accepted approaches – the time domain analysis and the frequency domain analysis coupled with the random vibrations, and the transformation of multiaxial to uniaxial stress state to determine the life of the air dryer bracket – is discussed here.

Estimation of the fatigue damage are presented in the flow chart in Fig. 3.1 for both the transient and spectral approach. The dynamical and frequency response are solved using MSC NASTRAN solver for the selected evaluation points. The output data from NASTRAN is imported and processed in a MATLAB algorithm that estimates the fatigue life using both the transient and spectral method. The left path shows the straightforward transient approach with rainflow count. This approach is represented in time domain. On the other hand, the right path shows the spectral method with Dirlik's formula, and is instead represented in frequency domain. The input signals are either represented as time signals or PSD signals. Since the problem is linear, it is convenient to calculate the transfer functions in the frequency response for all evaluation points and then multiply the transfer function matrix with the input PSD signals.

When comparing two different methods, it is essential to start with a simple case, i.e. uniaxial stress. Hence, the first thing to analyse was uniaxial stress in case of uniaxial base excitation. Thereafter, uniaxial stress in case of multiaxial base excitation. Lastly, multiaxial stress in case of multiaxial base excitation.

Only three local stress components $\sigma_{11}(t)$, $\sigma_{22}(t)$ and $\sigma_{12}(t)$ are considered in this study due to the shell-like geometry of the bracket component. The vibration that are induced from the interface surface excites the eigenmodes of the bracket and will cause a bending motion, giving higher stresses at the very surface of the shell elements than deeper within the material. Therefore, cracks will most likely initiate at the surface. If the surface is free from traction, the condition reduces to a plane stress problem with only three stress components. If we only consider translative motion from the interface, the whole prescribed motion can be represented by only three input accelerations $a_1(t)$, $a_2(t)$ and $a_3(t)$ – one in each direction. According to Section 2.6.1, the transfer function matrix then becomes

$$\underline{\hat{\sigma}}(f) = \mathbf{H}(f) \hat{\mathbf{a}}(f) \implies \begin{bmatrix} \hat{\sigma}_{11}(f) \\ \hat{\sigma}_{22}(f) \\ \hat{\sigma}_{12}(f) \end{bmatrix} = \begin{bmatrix} H_{11}(f) & H_{12}(f) & H_{13}(f) \\ H_{21}(f) & H_{22}(f) & H_{23}(f) \\ H_{31}(f) & H_{32}(f) & H_{33}(f) \end{bmatrix} \begin{bmatrix} \hat{a}_1(f) \\ \hat{a}_2(f) \\ \hat{a}_3(f) \end{bmatrix} \quad (3.1)$$

Hence, there are in general three input and three output signals giving a 3×3 transfer function matrix, where the components $H_{ij}(f)$ corresponds to all 9 combinations of input to output.

A single input signal can still generate a multiaxial stress state. Therefore, it is interesting to study the fatigue life when applying one input load individually at the time, and then let them act all together. This leads to a total of 4 different loading scenarios. Also, for each loading scenario, it is interesting to estimate the fatigue life from each stress signal individually (uniaxial fatigue) and all together (multiaxial fatigue). The total number of stress scenarios for each loading scenario is then 4, giving $4 \times 4 = 16$ combinations in total. In Fig. 3.2 a visualization of the 16 cases are shown.

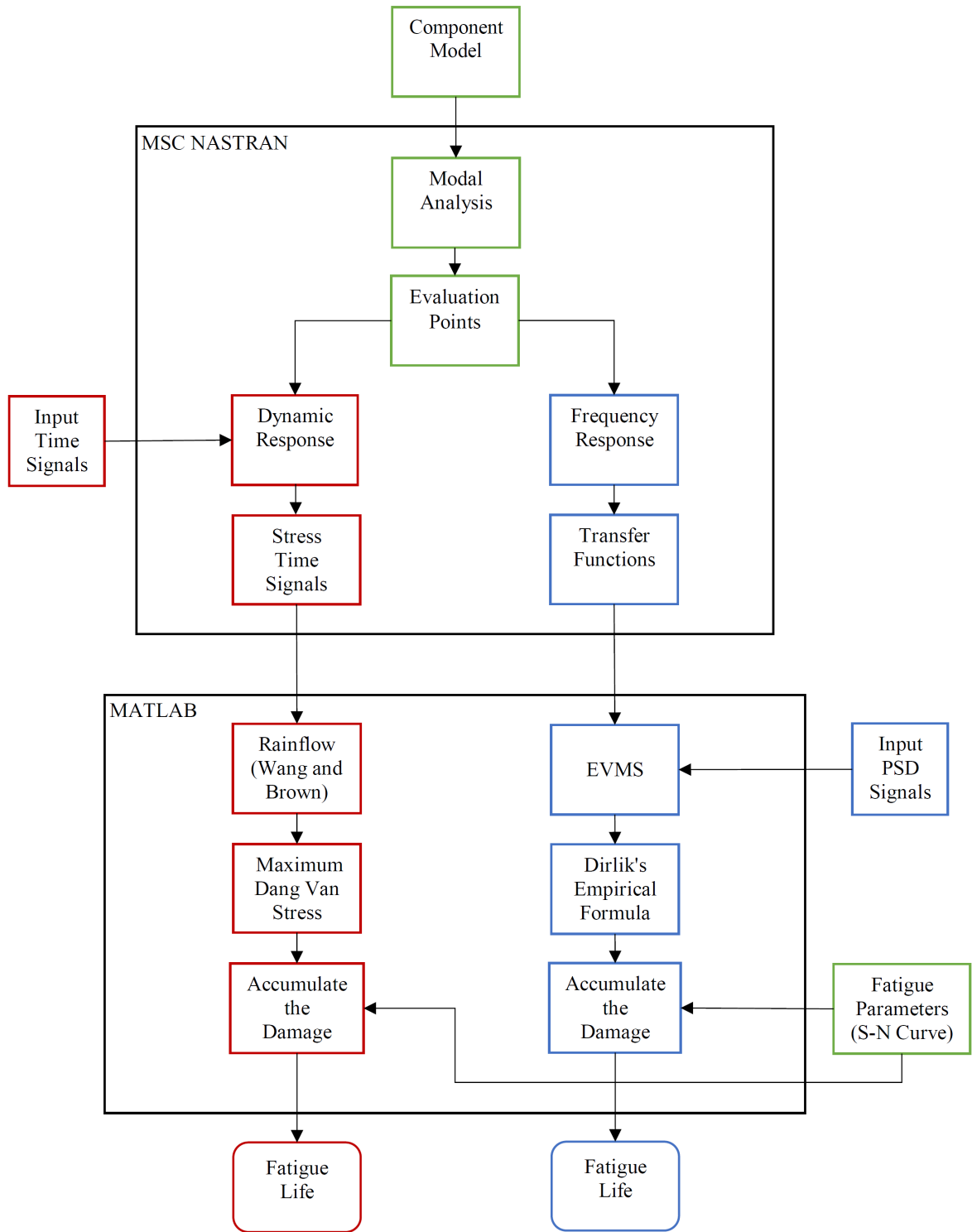


Figure 3.1: The flow chart depicts a step by step procedure in order to estimate the fatigue life in this study. The component model and the input signal are given. The input PSD Signals are derived from the input time signals by the use of FFT. Tasks that are solved in MATLAB and NASTRAN are grouped together for the reader to differentiate. Moreover, as the work proceeds, it gets split up into frequency domain (right sided path) and the time domain (left sided path). Two different results on fatigue life is obtained at the end. In case of uniaxial stress, the Wang and Brown's multiaxial rainflow count was replaced by an ordinary rainflow count, and the maximum Dang Van stress was ignored. The EVMS on the other hand works for both uniaxial and multiaxial stress state. The Fatigue parameters are given in Table 3.2.

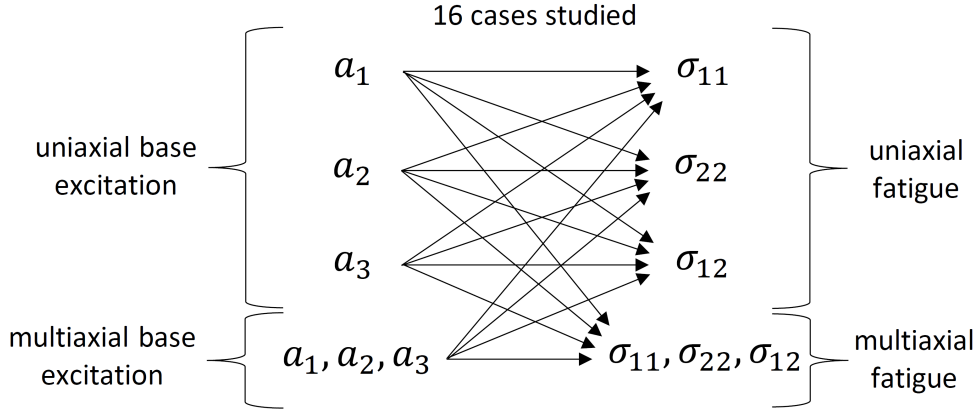


Figure 3.2: Different loading and stress cases that was studied in this project for three input accelerations a_1 , a_2 and a_3 , and three output stresses σ_{11} , σ_{22} and σ_{12} .

3.1 Generate Input Signals

Generating a signal that best represent the translative acceleration of the interface surface required that one knows what kind of vibration that is subjected to the studied component. The vibration is controlled by three prescribed accelerations $a_1(t)$, $a_2(t)$ and $a_3(t)$ in x, y and z-direction, respectively. The correlation between these accelerations are dependent on the structure that the component is mounted on, in this case the chassis frame of a truck. It is hard to estimate the vibration at the interface without knowing the eigenmodes of the entire structure and the source of vibration, for example vibration from the engine or the road profile. Nevertheless, the acceleration signals do not necessarily need to have the same magnitudes, phases and frequency spectrum, making the real-world translative vibration very complex.

The acceleration signals can be recorded from acceleration sensors that are attached close to the interface of interest while performing a dynamic test response. In this project, however, the enforced acceleration $a_1(t)$, $a_2(t)$ and $a_3(t)$ are not based on any measured data but was generated in MATLAB using normal distribution generator. As already mentioned in Section 2.4, the normal distribution is a good option when simulating random vibration due to the central limit theorem. The acceleration signals are considered uncorrelated (independent), with equal mean and standard deviation. This was done simply by generating three different signals from the same normal distribution density function. In reality, there is always a dependency between the vibration in x, y and z-direction and so there should exist a correlation between the input signals. But for simplicity we considered the case with no correlation. In Fig. 3.3 two scatter plots are shown for the acceleration vector $\mathbf{a}(t) = [a_1(t) \ a_2(t) \ a_3(t)]^T$ showing a perfect uncorrelation vibration and a case with correlated vibration.

If the generated time signals are too short, the rainflow count will not coincide well to Dirlik's empirical formula. Dirlik's empirical formula, and other spectral methods, are based on density functions and probability theory. In order to get a good comparison, one needs to count many cycles and therefore produce long time signals with many data points. Too short time signals will result in a very dispersed histogram that does not match the histogram from Dirlik's formula. A stochastic variables histogram tends to closely follow its corresponding density function only if many outcomes are considered (infinitely many outcomes in theory).

One should also remember that many spectral methods, Dirlik's formula included, are derived assuming normal distributed stress signals. If the stress signals are not normal distributed, the Dirlik's formula will not be valid. For that reason, the input load must also be normal distributed. Generating input signals from other density functions would most definitely result in a bad fatigue damage estimation in frequency domain. However, random vibration is usually represented by normal distribution, so it is unnecessary to pick other density functions.

The number of time steps used to generate the input time signals was set to $N_t = 200\ 000$, which is a generous amount that turned out to be sufficient for this purpose. The mean value μ_t was set to zero and standard deviation of $\sigma_t = 20\text{ m/s}^2$ was considered after doing some tests. The length of each time steps Δt is

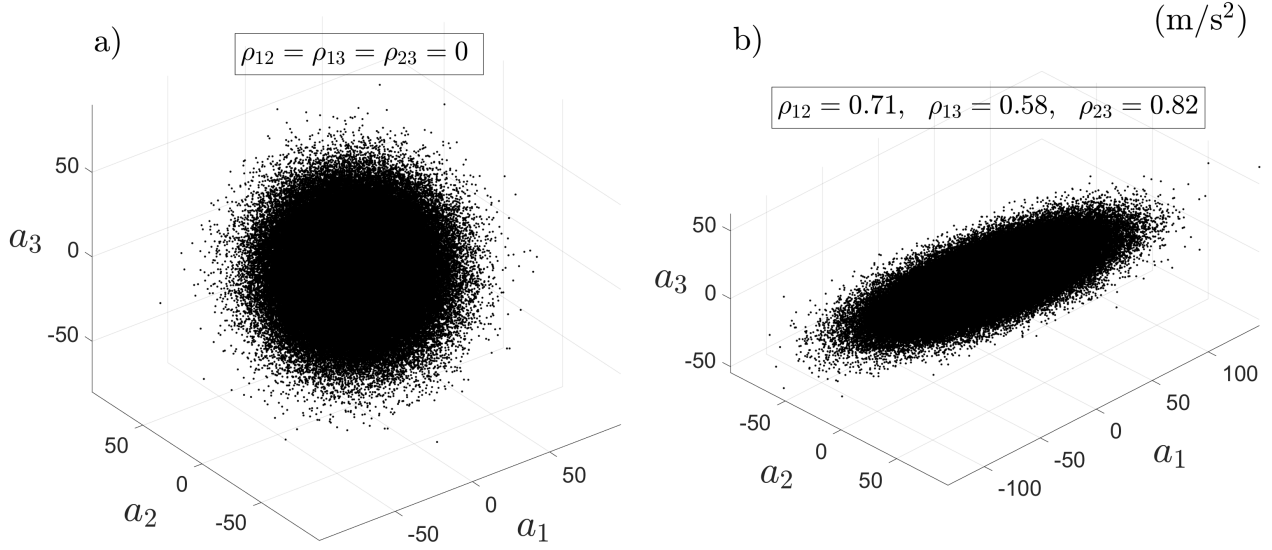


Figure 3.3: Scatter plots visualizing the correlation of the acceleration a_1 , a_2 and a_3 for two scenarios; independent a) and dependent b) vibration. In this project a) was used in case of multiaxial base excitation. The correlation factors ρ_{ij} between the acceleration signals are also shown for this particular example. In the special case when all the correlation factors are close to 1 or -1, the multiaxial base excitation reduces to a uniaxial base excitation.

directly related to the largest observable frequency, called Nyquist frequency f_{ny} given as

$$f_{\text{ny}} = \frac{f_s}{2} = \frac{1}{2\Delta t} \quad (3.2)$$

where $f_s = 1/\Delta t$ is the sampling frequency. Since the signal is represented by a discrete series of numbers, we cannot observe frequencies that have a time period less than $2\Delta t$, and $\pm f_{\text{ny}}$ will therefore be the limit of the frequency spectrum.

If we pick a smaller time step, it is possible to observe larger frequencies. But at the same time the length of the time signal $T = N_t \Delta t$ will decrease, and hence one has to increase the number of time steps N_t in order to preserve the length of the time signal. The choice of Nyquist frequency can be made by observing the transfer functions for the chosen hotspots. According to Fig. 3.10 the amplitude of the transfer functions have large peaks at the first two eigenfrequencies. Hence, a low frequency response is more severe. Large frequencies will only generate small stress ranges that are located far below the fatigue limit, while lower frequencies produce larger stress ranges. For that reason, a Nyquist frequency of $f_{\text{ny}} = 100$ Hz was estimated to be a good choice that enfolds the first 4 eigenfrequencies in the most severe region of the frequency spectrum. According to Eq. (3.2), the time step is then equal to $\Delta t = 0.005$ s and the time length becomes $T = 1000$ s.

$$N_t = 200\,000, \quad f_{\text{ny}} = 100 \text{ Hz} \implies \Delta t = 0.005 \text{ s}, \quad T = 1000 \text{ s} \quad (3.3)$$

In Fig. 3.4 an acceleration signal is generated using the properties explained above. All the input signals $a_1(t)$, $a_2(t)$ and $a_3(t)$ was generated in the same way.

3.1.1 Input PSD Matrix

From the input time signals, it is possible to construct the input PSD matrix using Eq. (2.23) which yields

$$\mathbf{G}(f) = \frac{1}{T} \hat{\mathbf{a}}(f) \hat{\mathbf{a}}(f)^\dagger = \frac{1}{T} \begin{bmatrix} |\hat{a}_1|^2 & \hat{a}_1 \bar{\hat{a}}_2 & \hat{a}_1 \bar{\hat{a}}_3 \\ \hat{a}_2 \bar{\hat{a}}_1 & |\hat{a}_2|^2 & \hat{a}_2 \bar{\hat{a}}_3 \\ \hat{a}_3 \bar{\hat{a}}_1 & \hat{a}_3 \bar{\hat{a}}_2 & |\hat{a}_3|^2 \end{bmatrix} \quad (3.4)$$

The discrete Fourier transform (DFT) signals $\hat{a}_1(f)$, $\hat{a}_2(f)$, and $\hat{a}_3(f)$ can be estimated from the discrete time signals using the fast Fourier transform (FFT) algorithm. The FFT algorithm is very fast and is recommended

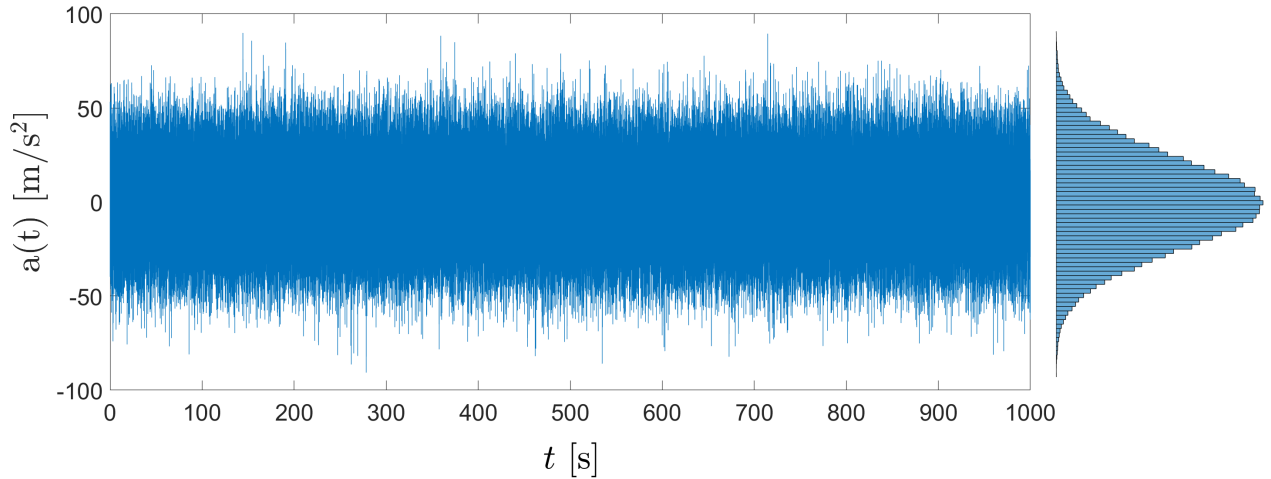


Figure 3.4: Acceleration input signal generated using normal distribution with zero mean, and standard deviation 20 m/s^2 . The number of time steps is $N_t = 200\,000$ and Nyquist frequency $f_{\text{ny}} = 100 \text{ Hz}$.

to use for very long time signals. However, the use of FFT does not always come without difficulties. For a very chaotic time signals as in Fig. 3.4, the FFT might give a very noisy DFT. This noise can be suppressed using various method. Perhaps one of the most established method is Welch's method, which splits the time signal into sequences. Each sequence is multiplied with a window function before being overlapped with adjacent sequences [30]. The number of sequences and overlapping points as well as the window function type are hard to decide in advance. Too many sequences or too many overlapping points might give a result that is filtering out even the desired information. Also, different window functions are more suitable than others in certain situations [31]. In this study a Hamming window function was used with 30 sequences and 50 % overlap ratio.

An infinitely long normal distributed time signal should in theory produce a constant PSD signal since all frequencies between $-f_{\text{ny}}$ and f_{ny} occurs the same number of times. This kind of random signal is called a white noise. In Fig. 3.5 the absolute value of the DFT is shown for the input signal in Fig. 3.4 in case of 30 sequences and 50 % overlap ratio using Welch's method. Other number of sequences were analysed as well, but it did not affect the result substantially. Also, a large number of sequences should be avoided for long time signals, since the computation time will be highly prolonged.

The PSD input matrix $\mathbf{G}(f)$ was generated directly using the MATLAB function `cpsd` which estimates the cross-PSD directly using Welch's method and Hamming window function by default.

3.2 Modal Analysis and Selection of Evaluation Points

In order to determine the life of the bracket, a closer look is taken at that location which encounters high stresses. These locations are termed as weak spots or hotspots. The hotspots are picked based on the general modal analysis (see Section 2.7). Therefore, choosing the correct hotspot is very critical here or one might end up picking wrong elements or elements with stress singularities. For the thesis work, the model data file is imported in META for postprocessing. In the NASTRAN code, 25 eigenfrequencies are requested from 0 Hz to 1000 Hz. In Table 3.1, the first seven eigenfrequencies are presented and are obtained from NASTRAN [20]. The table gives an idea about what frequencies the entire component oscillates more violently.

A typical shell element has top and bottom surfaces. The postprocessing in META is done for von Mises stress distribution across the bracket requesting to average the stresses between top and bottom surfaces for each element. It is important to have visual stress plots for each mode in order to pick the hotspots that undergoes uniaxial and multiaxial stress state. The most severe stress pattern that keeps repeating indicate points of interest for a good hotspot. It is stated in the earlier chapters that the lowest eigenmodes are the most dangerous ones and therefore only three modes are investigated. In Fig. 3.6 the von Mises stress distribution is shown for the first three eigenmodes. The stress distribution clearly follows a pattern in the first three modes and therefore element number 2562, 2873 and 3127 were picked for further durability tests. These elements experiences high stresses compared to the rest of the elements (excluding the stress singularities). Thus, the life of the bracket is now decided based on analysis of these three elements. This early stage reduction of a problem

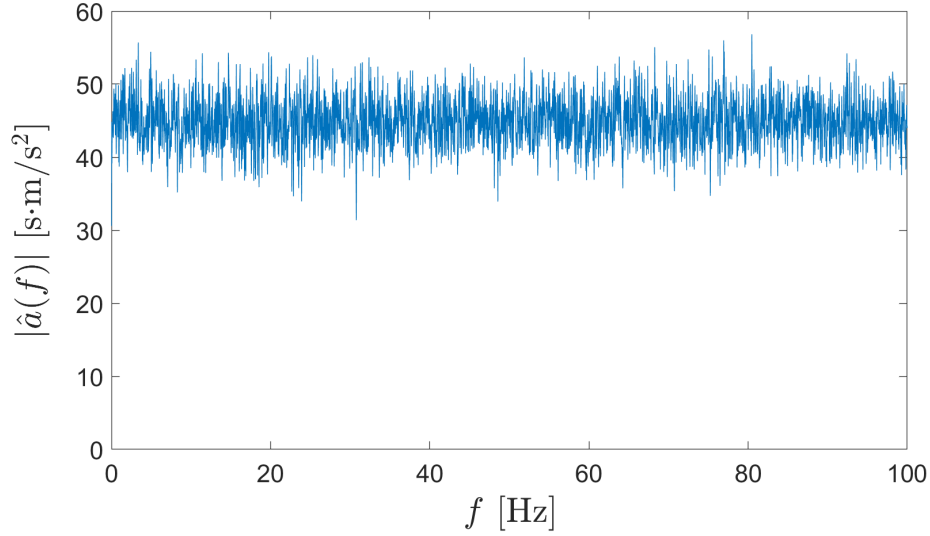


Figure 3.5: The amplitude spectrum of a normally distributed signal $a(t)$ with mean $\mu_t = 0$, standard deviation $\sigma_t = 20 \text{ m/s}^2$, Nyquist frequency $f_{\text{ny}} = 100 \text{ Hz}$ and time length $T = 1000 \text{ s}$. The DFT $\hat{a}(f)$ was obtained using FFT together with Welch’s method for 30 sequences and 50 % overlap ratio. Note that only the positive side of the frequency spectrum is shown. Indeed the amplitude spectrum is almost constant at $|\hat{a}(f)| = 45 \text{ s} \cdot \text{m/s}^2$ giving a constant PSD signal of $|\hat{a}(f)|^2/T = 2.025 \text{ m}^2/\text{s}^3$. The square root of the area under the curve is the simply obtained as $\sqrt{2f_{\text{ny}}|\hat{a}(f)|^2/T} \approx 20 \text{ m/s}^2$, which indeed is the standard deviation of the time signal $a(t)$.

Table 3.1: First seven eigenmodes and its respective eigenfrequencies acquired from MSC NASTRAN.

Mode number	Eigenfrequencies in Hz
1	17.9
2	19.5
3	46.5
4	90.4
5	139
6	148
7	449

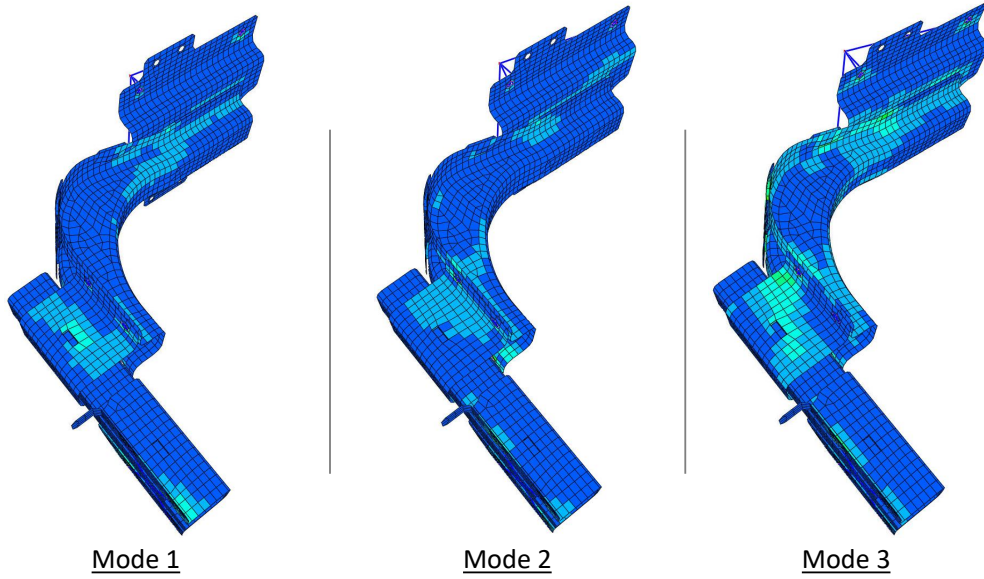


Figure 3.6: Image of a bracket with von Mises stress distribution for different modes. From left to right: Stress plot for mode 1 at 17.88 Hz, stress plot for mode 2 at 19.47 Hz and stress plot for mode 3 at 46.54 Hz. The model is analysed and captured in META postprocessing. The stresses are averaged between top and bottom surfaces for all the elements.

from an entire component to one single element allows to focus on more detailed analysis on the hotspot. The location of three hotspots used in this work is shown in Fig. 3.7.

The air dryer bracket model contains mostly shell elements [24]. A shell element is shown in more detail in Fig. 3.8, with a top and bottom surfaces. Fatigue failure initiates at the very surface of the component, since the stress varies linearly through the thickness of the shell element in case of bending. The maximum and the minimum stress values can therefore only occur at the surface (top or bottom of the element).

3.2.1 Stress Singularity

For a finite element model, the solution converges upon mesh refinement. It is intriguing how some elements display much higher stress values than their actual values. This is due to the large stress gradient in rapidly varying geometry. The elements associated to this never converge upon refinement of mesh. These elements are termed as stress singularities and they are mostly found around sharp corners, where there is a sudden change in geometry [17]. Choosing these elements as hotspots will clearly lead to an incorrect fatigue life. However, the elements that are located at some distance away from these singularity points presents valid stress results. An example of stress singularity found in the air dryer bracket is shown in Fig. 3.9.

The geometrical model is designed in such a way that the two bodies are held in contact with each other with a help of SPC (Single Point Constraint). These SPC's acts as bolt joints. The point of interest is along the throat of the bracket (see Fig. 1.2), which is sufficiently far away from the singularity point. If the point of interest is close to the singularity, then it is advised to use elastic-perfectly plastic material model and then the stresses can only reach up to the yield limit, beyond which yielding of material occurs. This is the plasticity side of the problem and this work does not deal with anything beyond yield point. Another way to deal with singularity is by extrapolating sufficiently far from the singularity where the stress gradient is small. Also, locally refining the mesh close the singularity gives better resolution on the stresses and shrinks the singularity region itself [18].

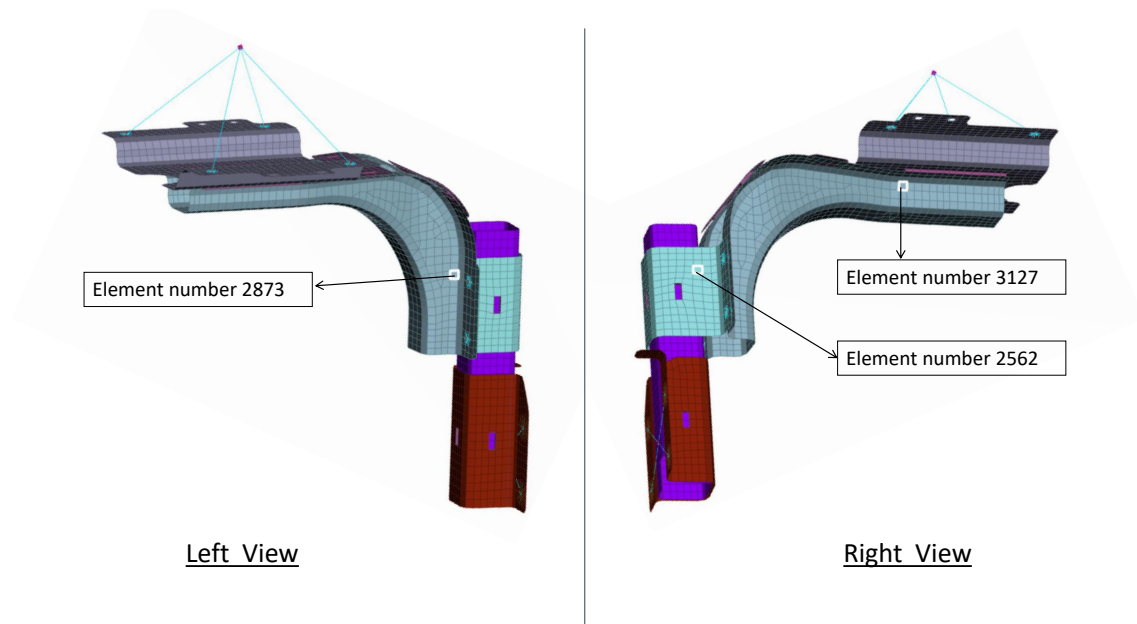


Figure 3.7: Elements that are exposed to higher stresses based on modal analysis. These elements are highlighted and captured in ANSA pre-processor and were picked at three distinctive locations on the bracket. Element 2873 and 2562 are located on the either side of the throat of the bracket and element 3127 is located underneath the air dryer.

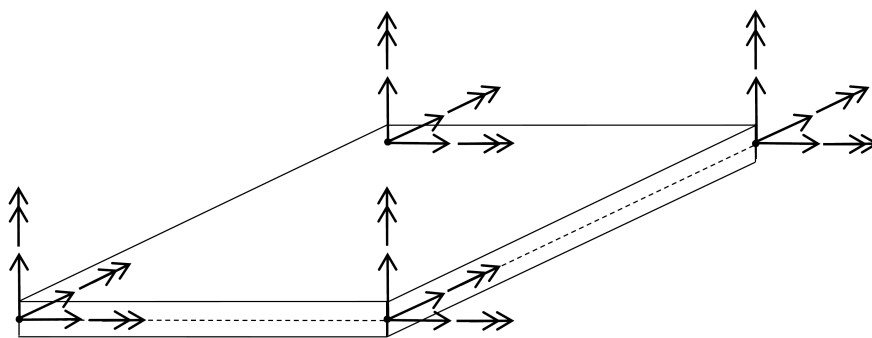


Figure 3.8: Illustration of a single shell element. The shell element has two surfaces, top and bottom. Each node at the four corners of the element has six degrees of freedom, i.e. displacement in x, y and z directions and rotation about x, y and z axes. The shell element does not exhibit normal stresses with respect to the surface, since no external traction load is applied (free surface). Therefore, it can only have three stress components (plane stress) in its local element coordinate system; two normal stresses and one shear stress.

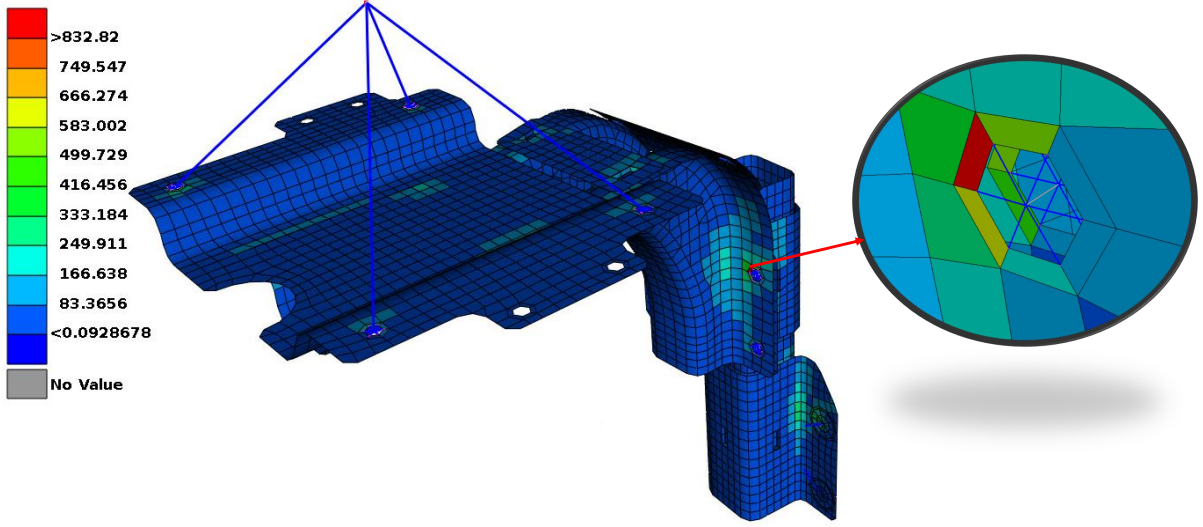


Figure 3.9: Singularity point that arose for mode 1. The zoomed image on the right is the element number Q4, 3016 (red element) is a point with high stresses around the bolt joint.

3.3 Evaluate Transfer Functions

The transfer function matrix $\mathbf{H}(f)$ introduced in Eq. (3.1) for a given hotspot, can be obtained if the global matrices \mathbf{M} , \mathbf{C} and \mathbf{K} are known. We also need information about what DOF:s that are free and constrained. However in this study we are only interested in obtaining the transfer functions for specific hotspots, and therefore it is more convenient to perform swept sine to receive the transfer functions directly from each stress component σ_{11} , σ_{22} and σ_{12} . The sine sweep method was carried out in NASTRAN for each selected hotspot element 2562, 2873 and 3127. NASTRAN solves the equation of motion in frequency domain when applying a given sinusoidal input load $a_1(t) = e^{i2\pi ft} \text{ m/s}^2$. The complex output stresses in the given element of interest is then equal to the transfer functions themselves, i.e. $\hat{\sigma}_{11}(f) = H_{11}(f)$, $\hat{\sigma}_{22}(f) = H_{21}(f)$ and $\hat{\sigma}_{12}(f) = H_{31}(f)$.

The problem is linear, which makes it possible to perform sine sweep on each acceleration signal $a_1(t)$, $a_2(t)$ and $a_3(t)$ individually as follows:

$$\begin{aligned} a_1(t) = e^{i2\pi ft} \text{ m/s}^2 &\implies \hat{\sigma}_{11}(f) = H_{11}(f), \quad \hat{\sigma}_{22}(f) = H_{21}(f), \quad \hat{\sigma}_{12}(f) = H_{31}(f) \\ a_2(t) = e^{i2\pi ft} \text{ m/s}^2 &\implies \hat{\sigma}_{11}(f) = H_{12}(f), \quad \hat{\sigma}_{22}(f) = H_{22}(f), \quad \hat{\sigma}_{12}(f) = H_{32}(f) \\ a_3(t) = e^{i2\pi ft} \text{ m/s}^2 &\implies \hat{\sigma}_{11}(f) = H_{13}(f), \quad \hat{\sigma}_{22}(f) = H_{23}(f), \quad \hat{\sigma}_{12}(f) = H_{33}(f) \end{aligned} \quad (3.5)$$

The total stress when all enforced acceleration signals are applied simultaneously is simply given by superimposing all contributions, which yields

$$\begin{cases} \hat{\sigma}_{11}(f) = H_{11}(f) \hat{a}_1(f) + H_{12}(f) \hat{a}_2(f) + H_{13}(f) \hat{a}_3(f) \\ \hat{\sigma}_{22}(f) = H_{21}(f) \hat{a}_1(f) + H_{22}(f) \hat{a}_2(f) + H_{23}(f) \hat{a}_3(f) \\ \hat{\sigma}_{33}(f) = H_{31}(f) \hat{a}_1(f) + H_{32}(f) \hat{a}_2(f) + H_{33}(f) \hat{a}_3(f) \end{cases} \implies \hat{\underline{\sigma}}(f) = \mathbf{H}(f) \hat{\underline{a}}(f) \quad (3.6)$$

The transfer functions $H_{ij}(f)$ are in this case transferring the signal from acceleration 1 m/s^2 to stress in MPa, which gives the transfer function unit $\text{MPa} \cdot \text{s}^2/\text{m}$. Hence if the input signal has the unit m/s^2 , the output stress will have the unit MPa.

The magnitude of all transfer function $H_{ij}(f)$ are shown in Fig. 3.10 for element number 2562, 2873 and 3127. Each element has a top and bottom stress (see Section 3.2). Only the most severe surface stress of the element was selected, either top or bottom. Evaluating the stress in the centre of the element or evaluating the

mean stress of the top and bottom is a bad option, since the fatigue damage will become highly reduced. The largest stress values are found on the surface; either the top or bottom of the element. As the transfer functions suggests in Fig. 3.10, the frequency response is more severe for the first 2 eigenfrequencies, which can be seen as large peaks. This is not surprising, since the air dryer bracket will then undergo a swaying motion which results in large deformations and high stress values. Higher frequencies of the input signal will produce stress signals with small amplitudes and therefore small stress ranges. On the contrary, lower frequencies close to the first two eigenfrequencies of the input signal will produce stress signals with large amplitudes and therefore large stress ranges that contributes more to fatigue damage.

In Fig. 3.10 we can also distinguish which input loads that are most severe. In element 2873, for example, the largest peak in $|H_{11}(f)|$ is found at the first eigenfrequency of 17.88 Hz, which means that an acceleration load in x-direction with frequencies close to 17.88 Hz will generate the largest stress amplitude for stress component σ_{11} . The peaks are almost always located close the eigenfrequencies of the structure. The height of the peaks are however directly related to the damping of the structure. A structure with no damping will result in infinitely large peaks in the transfer functions, and a larger damping will suppress the peaks and smoothens out the transfer function (see Section 2.5). In this study, the so called modal damping method was used with a damping coefficient of $\zeta_i = 3\%$ for all eigenmodes i , which is a serious limitation. However, since the input loads have a Nyquist frequency of 100 Hz, only the first 4 eigenmodes will be excited.

3.4 Stress PSD matrix, EVMS and Dirlik's Empirical formula

Once the transfer functions $H_{ij}(f)$ are obtained and the input PSD matrix $G_{ij}(f) = \hat{a}_i(f) \overline{\hat{a}_j(f)}/T$ generated, we can estimate the output PSD matrix $S_{ij}(f)$ directly using Eq. (2.51). Alternatively, it is possible to estimate the Fourier transform of the stress components as $\hat{\underline{\sigma}}(f) = \mathbf{H}(f) \hat{\mathbf{a}}(f)$ and then use $\mathbf{S}(f) = \hat{\underline{\sigma}}(f) \hat{\underline{\sigma}}(f)^\dagger / T$. In case of multiaxial fatigue, the stress PSD matrix is reduced to an EVMS signal $S_{\text{VM}}(f)$ using Eq. (2.106).

From the EVMS PSD it is possible to generate the k :th EVMS moment of area $m_{\text{VM},k}$, which are used as parameters in order to construct the Dirlik's density function for reversible stress range in Eq. (2.99) and thereafter estimate the fatigue life directly using Eq. (2.102) if the fatigue limit is not considered. If the fatigue limit is considered, Eq. (2.102) is not valid, and then it is better to construct a histogram from the Dirlik's density function using Eq. (2.98) and thereafter apply the reversible uniaxial S-N curve and Palmgren-Miner rule.

Below follows a stepwise explanation on how to estimate fatigue life using Dirlik's empirical formula as spectral method. The same approach can be applied to other spectral methods as well.

1. Fourier transform of stress components, $\hat{\underline{\sigma}}(f) = \mathbf{H}(f) \hat{\mathbf{a}}(f)$
2. PSD stress matrix, $\mathbf{S}(f) = \frac{1}{T} \hat{\underline{\sigma}}(f) \hat{\underline{\sigma}}(f)^\dagger$
3. EVMS signal, $S_{\text{VM}}(f) = \mathbf{Q} : \mathbf{S}(f)$
4. EVMS moment of area, $m_{\text{VM},k} = 2 \int_0^\infty \text{Re}\{S_{\text{VM}}(f)\} f^k df$
5. Construct Dirlik's density function $p_Z(u)$ in Eq. (2.99) using $m_{\text{VM},0}, m_{\text{VM},1}, m_{\text{VM},2}$ and $m_{\text{VM},4}$
6. Estimated number of cycles for each stress range (histogram), $N_i = E_p T p_Z(u_i) \Delta u$
7. Apply uniaxial reversible S-N curve and Palmgren-Miner rule, $T_{\text{life}} = \frac{T}{D} = \frac{CT}{\sum_i N_i u_i^k}$

Here u_i denotes reversible stress range values ($u_i = \Delta \sigma_{r,i}$). When fatigue limit is not considered, step 5-7 can be replaced by Eq. (2.102). Step 1-3 can also be replaced by $S_{\text{VM}}(f) = Q_{ij} H_{ik}(f) \overline{H_{jl}(f)} G_{kl}(f)$. Note that the same algorithm can be applied for uniaxial stress state, since $S_{\text{VM}}(f)$ is then equal to the PSD of the uniaxial stress.

The input and output PSD:s $\mathbf{G}(f)$ and $\mathbf{S}(f)$ are always good to determine, since they bear information about the input and output signal such as covariance and correlation. Also, the PSD matrices contain the information needed in order to construct density function such as Dirlik's empirical formula. Note that the transient data can also be used in spectral methods. However, in that case we need to estimate the variance of first and second order time derivative of the stress signals. Numerically differentiating a stress signal multiple times is not a good idea, especially if the data points are sparse, since it could lead to large errors (loss of

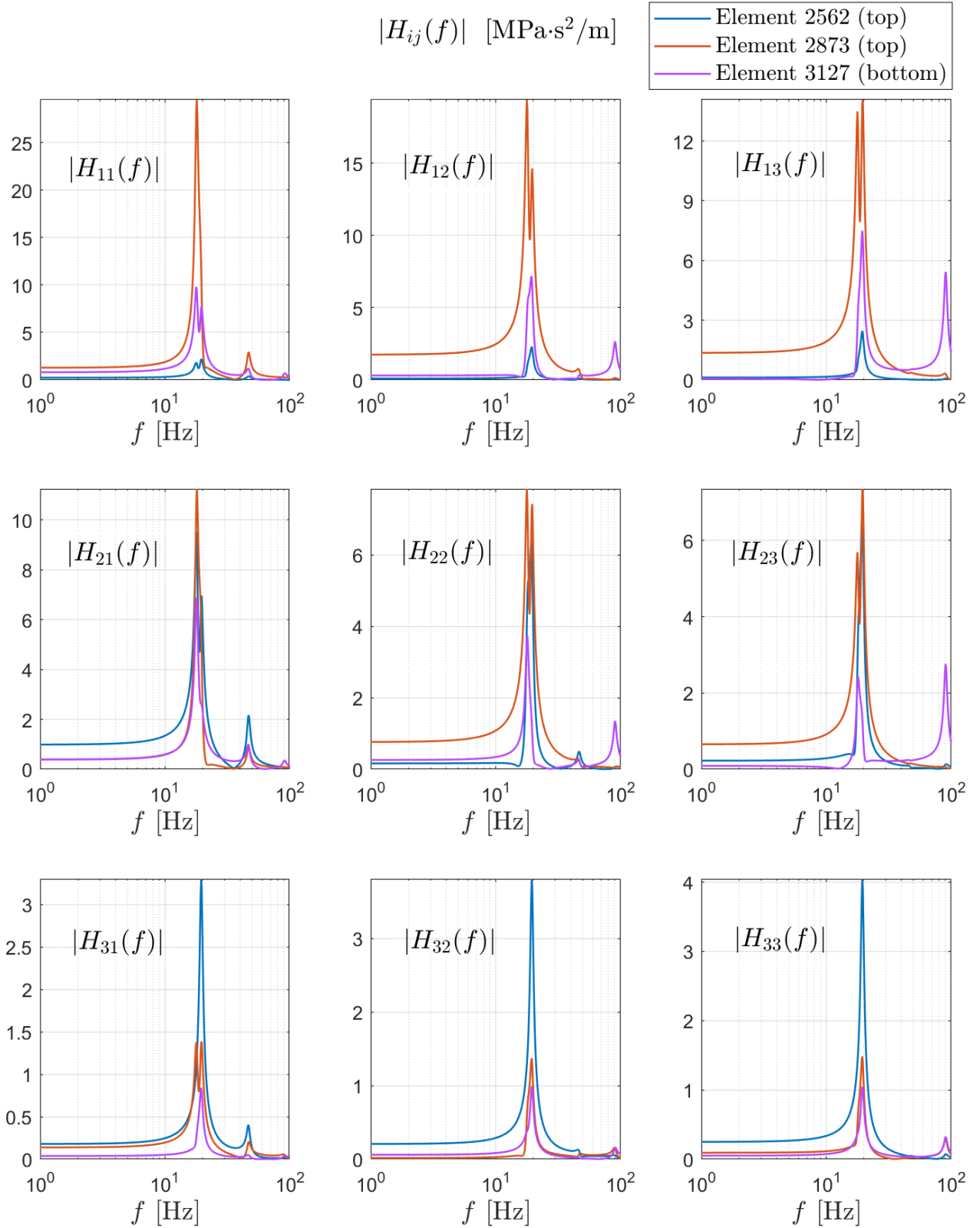


Figure 3.10: Magnitude of the transfer function components $H_{ij}(f)$ for each element hotspots 2562, 2873 and 3127. Only the positive frequency spectrum is presented from 1 to 100 Hz with logarithmic scale. The transfer functions are showing large peaks for the first two eigenfrequencies. Beyond 100 Hz, the transfer functions are showing very small magnitudes, which indicates that lower frequency response is more critical for the selected hotspots.

information). Time derivative in frequency domain only leads to a frequency factor f , and so the accuracy is preserved (see Section 2.10).

The expected damage and fatigue life are just two scalar values; and comparing the frequency domain approach to time domain approach with a few scalar values is not very convincing. A better way to compare results is to visualise with help of a histogram plot. By doing so, one gets a better understanding on how the damage predictions differs. The histogram plot depicts the contribution of damage for each stress cycle.

3.5 Dynamic Response

The transient stress signals $\sigma_{11}(t)$, $\sigma_{22}(t)$ and $\sigma_{12}(t)$ in the given hotspots 2562, 2873 and 3127 was obtained using dynamic response in NASTRAN with prescribed acceleration $a_1(t)$, $a_2(t)$ and $a_3(t)$ for the translative DOF:s at the interface. Due to linearity, we can apply one load at the time and superimpose the stress contribution for each loading case. NASTRAN solves the dynamic response explicit using linear combination of eigenmodes [20].

Solving a linear system using linear combination of eigenmodes are very fast compared to a direct integration method, especially if we have many DOF:s. To illustrate this method, consider the equation of motion of the free DOF:s in Eq. (2.55). Now, assume that ω_i are the undamped eigenfrequencies and Φ_i are the corresponding undamped eigenmodes of the system, already introduced in Section 2.7. The solutions to the equation of motion is then $\Phi_i \eta_i(t)$ where $\eta_i(t)$ is a time dependent scalar function. Substituting this in Eq. (2.55) and multiplying by Φ_i^T yields

$$\ddot{\eta}_i(t) + 2\zeta_i \omega_i \dot{\eta}_i(t) + \omega_i^2 \eta_i(t) = \Phi_i^T \mathbf{f}'_F(t) \quad (3.7)$$

where $\mathbf{f}'_F(t)$ is the time dependent load vector including both force load $\mathbf{f}_F(t)$ and prescribed motion \mathbf{q}_C . The general solution can then be obtained by superimposing all solutions $\Phi_i \eta_i(t)$ as [4]

$$\mathbf{q}_F = \sum_{i=1}^n \Phi_i \eta_i(t) \quad (3.8)$$

The exact solution $\mathbf{q}_F(t)$ requires a linear combination of all eigenmodes. However, for low frequency response, only the first eigenmodes are excited. In that case, an accurate solution can be obtained by calculating the linear combination of only the first eigenmodes in Eq. (3.8). Once, \mathbf{q}_F is known, the stress response can be calculated using kinematic and constitutive relations.

3.6 Stress Time Signals and Rainflow Count

Once the stress time signals are obtained according to Section 3.5, it is possible to count the cycles using rainflow count algorithm and then accumulate the damage. For a uniaxial stress state, the procedure is straightforward and can be summarized as follows (assuming no mid stress effects):

1. Rainflow count algorithm, halfcycles. Extract stress ranges, $\Delta\sigma_{r,i}$
2. Reversible uniaxial S-N curve, $N_{f,i} = C \Delta\sigma_{r,i}^{-k}$
3. Palmgren-Miner rule, $T_{\text{life}} = \frac{T}{D} = \frac{T}{\sum_i 0.5/N_{f,i}} = \frac{2CT}{\sum_i \Delta\sigma_{r,i}^k}$

Note that the halfcycles are counted in this case. If whole cycles are counted, consider replacing 0.5 by 1, and remove factor 2. The rainflow halfcycle count is presented in Section A.1.1.

In case of multiaxial stress, the cycles must be extracted using a more general rainflow count method. The Wang and Brown method was used for this purpose. See Section 3.6.2 for more explanation. Moreover, we need an equivalent stress in order to account the damage distribution from all stress components. In this case maximum Dang Van stress has been used (see Section 2.9.2). The stepwise procedure for multiaxial fatigue can

then be summarized as

1. Multiaxial rainflow count, Wang and Brown's method, halfcycles. Save all cycle paths.
2. Calculate maximum Dang Van stress $\sigma_{dv,max,i}$ for each extracted cycle path.
3. Convert from maximum Dang Van stress to reversible stress range, $\Delta\sigma_{r,i} = \left(\frac{1}{4} + \frac{c_{dv}}{6}\right)^{-1} \sigma_{dv,max,i}$
4. Reversible uniaxial S-N curve, $N_{f,i} = C\Delta\sigma_{r,i}^{-k}$
5. Palmgren-Miner rule, $T_{life} = \frac{T}{D} = \frac{T}{\sum_i 0.5/N_{f,i}} = \frac{2CT}{\sum_i \Delta\sigma_{r,i}^k}$

Note that the same approach can be used but with replacing Dang Van or Wang and Brown with other methods. In case of proportional loading, step 1 can be replaced by uniaxial rainflow count. Again, factor 0.5 and 2 in step 5 is present due to halfcycle count. Step 3 is derived from maximum Dang Van stress in case of reversible uniaxial loading. Step 4-5 is identical to step 2-3 for the uniaxial stress state.

3.6.1 Rainflow Count – Halfcycles

Counting halfcycles using rainflow count algorithm is straightforward and is performed by scanning through the signal for increasing time. Before applying rainflow count, however, it is convenient to remove all points that does not represent a local maximum or minimum since those points will not contribute to any damage. In Fig. 3.11 the rainflow count has been performed on a stress sequence just for demonstration. The algorithm starts from the extreme left point and scans through the signal until it reaches the last but one point (the extreme right point is not considered). For a signal with n number of peaks and valleys, $n - 1$ halfcycles will be counted. Each cycle path starts from a peak or valley and stops when the algorithm has scanned throughout the whole signal.

If the starting point of the cycle path represent a local minimum, the cycle path is only allowed to increase. Hence, the stress value of the cycle path is only updated if the stress signal is increasing. If not, the cycle path will follow a constant straight line as shown in Fig. 3.11. On the contrary, if the starting point represent a local maximum, the cycle path is only allowed to decrease. The cycle path stops if one of the four following conditions are fulfilled:

1. The cycle path reaches the end of the sequence.
2. The starting point is a local minimum and the cycle path encounters a stress value that is less than the starting point.
3. The starting point is a local maximum and the cycle path encounters a stress value that is larger than the starting point.
4. The cycle path encounters an additional point, i.e. interfere with another cycle.

The additional points in Fig. 3.11 are denoted F', I' and L', and have to be saved and updated for each cycle count in order to make sure that the cycles are not interfering with one another. If σ_L and σ_R denotes the end point stresses for a given cycle, the stress range and mid stress is obtained as

$$\Delta\sigma = |\sigma_L - \sigma_R|, \quad \sigma_m = (\sigma_L + \sigma_R)/2 \quad (3.9)$$

For more information about the rainflow count algorithm, see MATLAB function `rainflow_count_halfcycles` in Section A.1.1.

3.6.2 Wang and Brown's method – Multiaxial Rainflow Count

Wang and Brown's method can be applied in order to extract cycles from a multiaxial stress state using rainflow count on an equivalent von Mises stress (or strain) signal. The von Mises stress can be written on Voigt notation according to Eq. (2.103) which yields

$$\sigma_{vM}(t) = \sqrt{\underline{\sigma}(t)^T \mathbf{Q} \underline{\sigma}(t)} \quad (3.10)$$

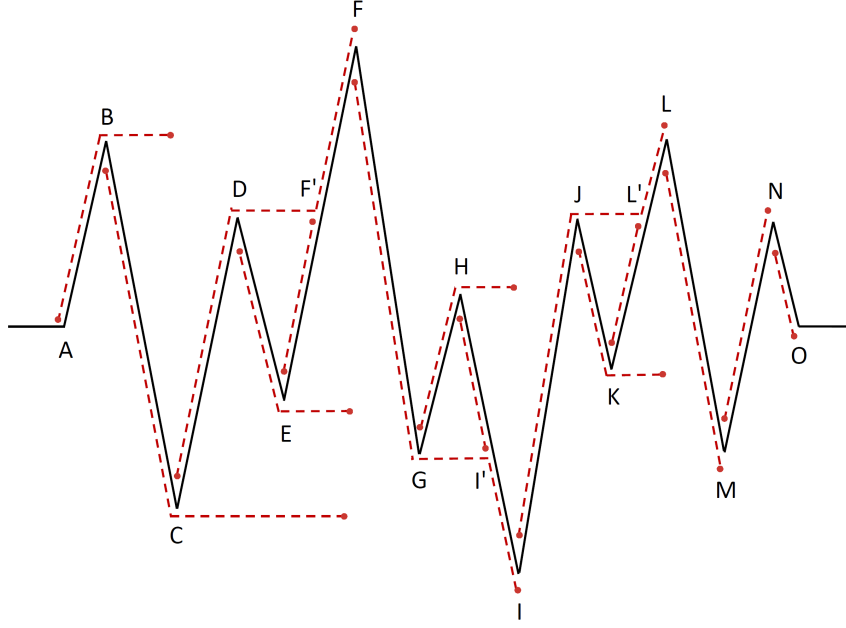


Figure 3.11: Rainflow count algorithm that counts halfcycles for an arbitrary stress signal. Each cycle path is marked with dashed lines. The stress signal only contains peaks and valleys. Each peak and valley are marked with letters from A to O. Additional points such as F', I' and L' are added if the cycle takes a path between two adjacent points.

where \mathbf{Q} is the Q-matrix already defined in Eq. (2.103). The major problem with the von Mises stress is the fact that it is always positive. To overcome this problem while performing rainflow count, one can use something called relative von Mises stress which subtracts the stress values at the starting point of the cycle path for all stress components [32]. If the stress tensor of the starting point is denoted $\underline{\sigma}_A$, the relative von Mises stress with respect to that point is defined as

$$\sigma_{vM,A}(t) = \sqrt{(\underline{\sigma}(t) - \underline{\sigma}_A)^T \mathbf{Q} (\underline{\sigma}(t) - \underline{\sigma}_A)} \quad (3.11)$$

The Wang and Brown's method is similar to the halfcycle count in Section 3.6.1. The only difference is that the rainflow count algorithm is applied on the relative von Mises stress in Eq. (3.11). However, the relative von Mises is equal to zero at the starting point making it a local minimum point. For that reason, condition 2 and 3 in Section 3.6.1 can be suppressed, since it will never be used in this context. Since the cycles are not allowed interfering with each other, it is essential to keep track of the occupied cycle paths. A cycle can never take a path that has already been taken and stops when one of the two conditions are fulfilled:

1. The cycle path reaches the end of the sequence.
2. The cycle path encounters an already existing cycle path.

Note that, unlike the uniaxial rainflow count, the stress signals must not be reduced into peaks and valleys, since the von Mises stresses can have peak values in regions where the stress components are not having any peaks. No reduction can be made while having multiple stress components, since it could lead to loss of important information. For similar reason, the whole path of the extracted cycles needs to be saved, not only the first and last value of the cycle path [1].

In Fig. 3.12 the rainflow count for a relative von Mises stress is illustrated. The relative von Mises stress has been estimated with respect to the starting point every time a new cycle path is initiated. All cycle paths are unique, which means that no interference is allowed.

When the cycle paths are following the von Mises stress, the stress values are well defined. However, when the cycle paths are taking a constant route, for example B-F' and D-F'' in step 1 and 3 in Fig. 3.12, the stress values are undefined and need to be projected. The relative von Mises stress is then constant, which means that the von Mises hyper ellipsoid is constant in time. Consider two points L and R in Fig. 3.13. The stress values in point L and R are then well defined and located on the hyper ellipsoid, and the linear path between L

and R have to be projected on the surface of the hyper ellipsoid, since the relative von Mises stress must be constant along that path [1]. The path do not have to be linear, but in order to avoid too conservative paths, the shortest distance between R and L along the ellipsoid surface is considered, i.e. a projected straight line.

If $\underline{\sigma}_A$ denotes the starting point for the given cycle path, the relative von Mises stress along L-R is given by

$$\sqrt{(\underline{\sigma}_L - \underline{\sigma}_A)^T \mathbf{Q} (\underline{\sigma}_L - \underline{\sigma}_A)} = \sqrt{(\underline{\sigma}_R - \underline{\sigma}_A)^T \mathbf{Q} (\underline{\sigma}_R - \underline{\sigma}_A)} = C_{LR} \quad (3.12)$$

where $\underline{\sigma}_L$ is the stress tensor at point L and $\underline{\sigma}_R$ is the stress tensor at point R. The surface equation for the hyper ellipsoid is thus

$$(\underline{\sigma} - \underline{\sigma}_A)^T \mathbf{Q} (\underline{\sigma} - \underline{\sigma}_A) = C_{LR}^2 \quad (3.13)$$

with the centre located at point $\underline{\sigma}_A$. Further, a straight line from point L to R in stress space can be stated as

$$\underline{\sigma} = \underline{\sigma}_L + s(\underline{\sigma}_R - \underline{\sigma}_L) \quad (3.14)$$

where s is the interpolation factor varying between $0 \leq s \leq 1$ along L-R. However, as illustrated in Fig. 3.13, the stress tensor has to be scaled so that Eq. (3.13) is fulfilled. In other words, the straight line has to be projected on the hyper ellipsoid surface, yielding

$$\underline{\sigma} = \frac{C_{LR}}{\sqrt{(\underline{\sigma}_L - \underline{\sigma}_A + s(\underline{\sigma}_R - \underline{\sigma}_L))^T \mathbf{Q} (\underline{\sigma}_L - \underline{\sigma}_A + s(\underline{\sigma}_R - \underline{\sigma}_L))}} (\underline{\sigma}_L + s(\underline{\sigma}_R - \underline{\sigma}_L)) \quad (3.15)$$

This can easily be verified by substitution of Eq. (3.15) into Eq. (3.13). Every time the cycle path is following a constant straight line similarly to the example in Fig. 3.13, the stress values along that path is obtained from Eq. (3.15). The interpolation factors s are chosen according to the resolution of the stress history.

In order to verify the multiaxial rainflow count, it is convenient to compare the result to the uniaxial rainflow count when only one stress signal is considered. In Fig. 3.14 rainflow count has been made on a single stress component using uniaxial halfcycle count and Wang-Brown's method.

For more information about the authors version of Wang and Brown's method, the reader is referred to the MATLAB function `multiaxial_rainflow_count` in Section A.1.2.

3.7 S-N Curve and Fatigue Parameters

In order to quantify the life of the bracket, S-N curve is constructed. As discussed in the theory, the slope of the S-N curve is extremely sensitive and a slight change in the slope could lead to a significant shift in life. Deriving a true S-N curve is important when simulating the fatigue life, however the S-N curve models usually comes with great uncertainty (see Section 2.8).

The material chosen is a SAE 1045 steel material with ultimate strength $\sigma_u = 621$ MPa. The lower and upper stress range limit is then given as $\Delta\sigma_{er} = 2m\sigma_u$ and $\Delta\sigma'_r = 2m'\sigma_u$, where the factor m and m' are the reduction factor. The value of m depends upon many parameters such as surface effects, temperature, loading type, size factors, corrosion etc., that reduces the fatigue limit of the material. The value of m' is typically around 0.7 and 0.9 and sets the upper limit of the high cycle fatigue region. The reduction factor m can be factorized as

$$m = m_e m_t m_d m_s m_o \quad (3.16)$$

where m_e is the bending fatigue limit factor, m_t is the load type factor, m_d is the size (stress gradient) factor, m_s is the surface finish factor and m_o denote other effects such as corrosion and heat.

The Eq. (3.16) includes all the crucial parameters that are chosen based on Juvinall (2006) [9]. The bending fatigue limit factor is equal to $m_e = 0.5$ for $\sigma_u \leq 1400$ MPa. Also due to the location of the 9.0 kg air dryer, bending loads are applied on the bracket and for this reason the load type factor was set to $m_t = 0.8$ and the size factor to $m_d = 0.9$. The material is considered to be hot rolled with the surface finish factor of $m_s = 0.7$. Other effects like corrosion, humidity and temperature is assumed to have no affect on the bracket and thus $m_o = 1$. Further, the factor m' is considered to be 0.9 for the case of bending according to Juvinall (2006) [9]. The Dang Van parameter in case of multiaxial fatigue was set to $c_{dv} = 1/3$, which is more or less a standard value if the shear S-N curve is unknown.

The fatigue parameters are listen in Table 3.2 and the constructed uniaxial and multiaxial S-N curve are presented in Fig. 3.15. Any stress range below the fatigue limit will not contribute any damage according to

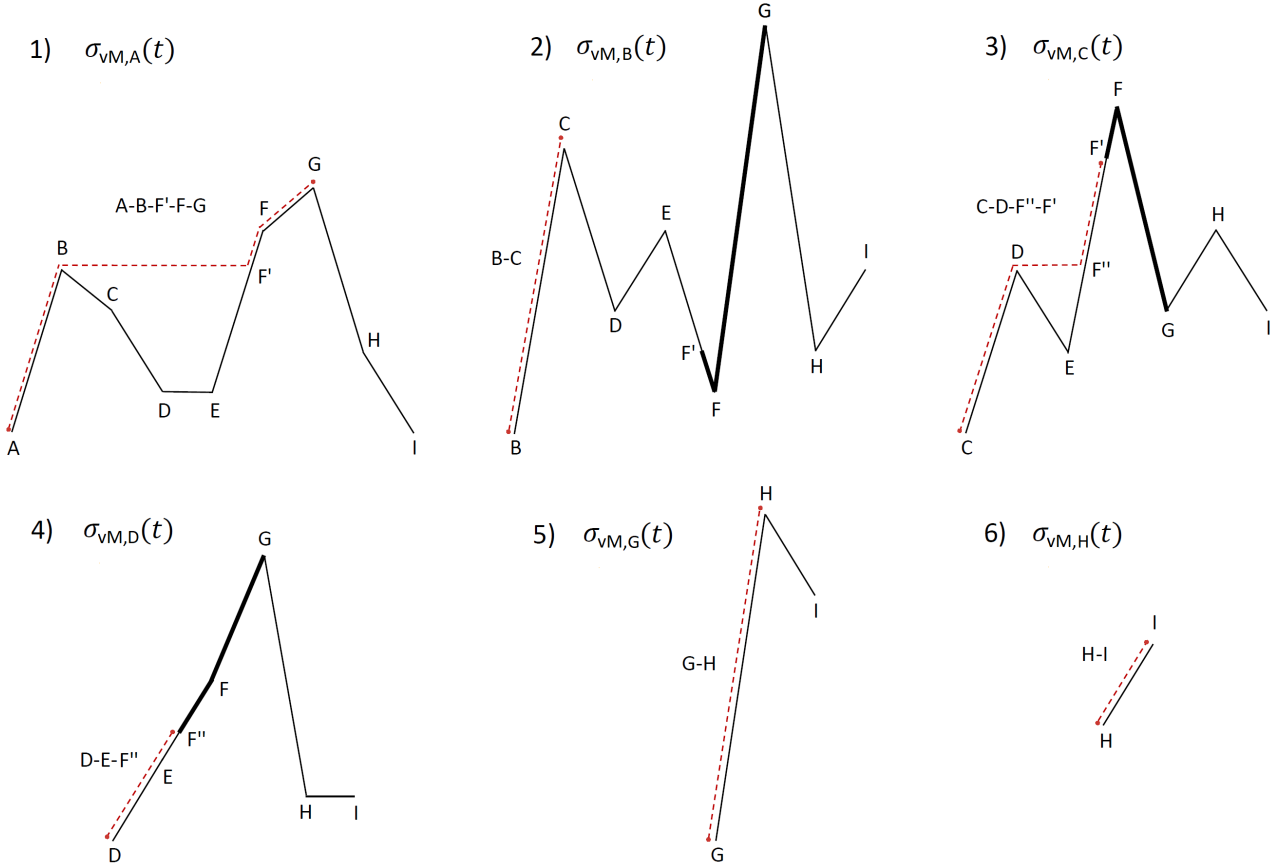


Figure 3.12: Visual stepwise example on how to perform rainflow count for a relative von Mises stress signal with data points A-I. A new relative von Mises stress signal needs to be calculated every time a new cycle path is initiated. The relative von Mises stress with respect to the first starting point A is simply given as $\sigma_{vM,A}(t) = \sqrt{(\underline{\sigma}(t) - \underline{\sigma}_A)^T \mathbf{Q} (\underline{\sigma}(t) - \underline{\sigma}_A)}$, and for the second starting point B $\sigma_{vM,B}(t) = \sqrt{(\underline{\sigma}(t) - \underline{\sigma}_B)^T \mathbf{Q} (\underline{\sigma}(t) - \underline{\sigma}_B)}$, where $\underline{\sigma}_A$ and $\underline{\sigma}_B$ are the stress values in point A and B, respectively. The same goes for cycle 3, 4, 5 and 6. The cycle paths are not allowed to decrease or interfere with other cycles. Each halfcycle is presented with dashed lines. The bold lines indicate occupied cycle paths.

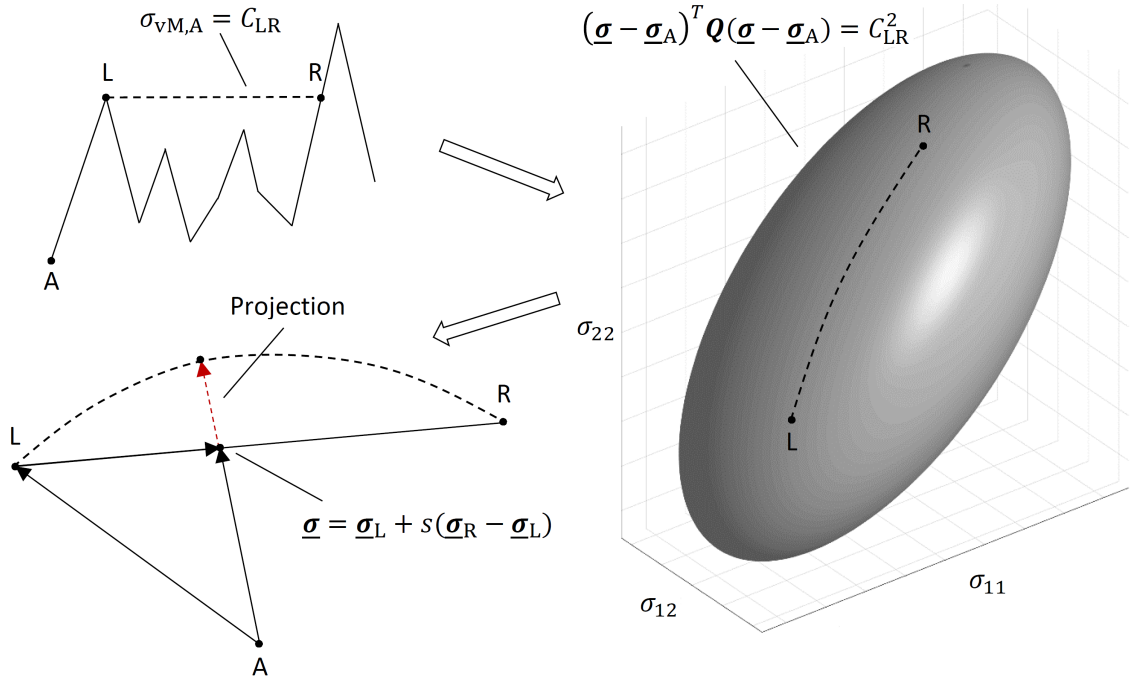


Figure 3.13: Example showing the constant relative von Mises path from point L to R. The L-R path can be projected on the surface of the von Mises hyper ellipsoid in stress space. The red dashed arrow illustrates the projection of the straight line in Eq. (3.14) on the ellipsoid in Eq. (3.13). The von Mises ellipsoid is presented for plane stress, $\underline{\sigma} = [\sigma_{11} \ \sigma_{22} \ \sigma_{12}]^T$, but same procedure is applicable for lower or higher dimensions (in general 6 dimensional stress space).

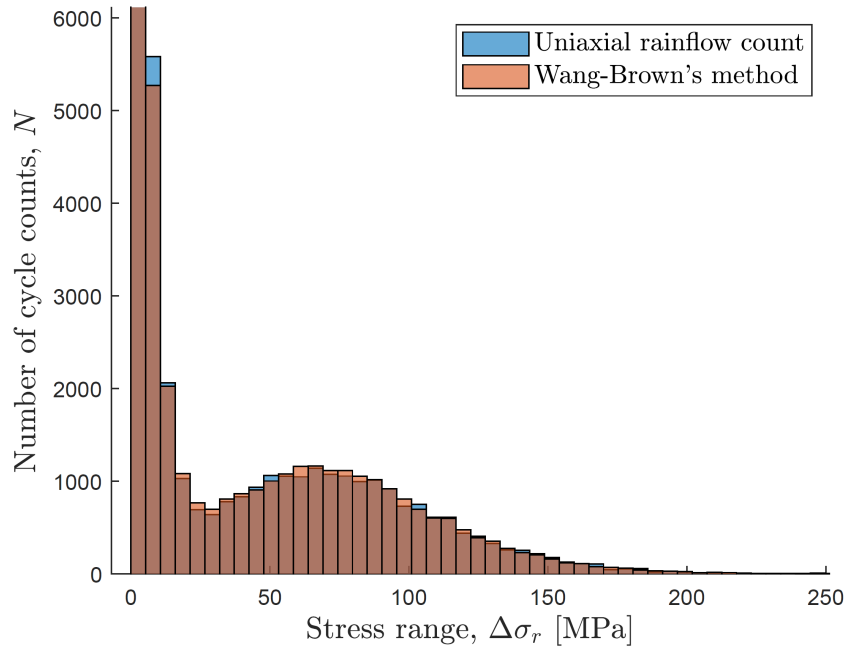


Figure 3.14: Histogram plots comparing uniaxial rainflow count and Wang and Brown's method for a single stress signal σ_{11} , in order to verify the performance of the Wang and Brown's algorithm. Note that the histogram plots are transparent; the blue histogram is visible behind the red.

Table 3.2: Fatigue parameters and its corresponding values used in this thesis work in order to construct the S-N curve. The fundamental fatigue parameters are decided according to Juvinall (2016) [9], and the rest are derived.

Parameter	Value	Derivation
σ_u	621 MPa	–
m_e	0.5	–
m_t	0.8	–
m_d	0.9	–
m_s	0.7	–
m_o	1.0	–
m'	0.9	–
m	0.252	Eq. (3.16)
c_{dv}	1/3	–
$\Delta\sigma'_r$	1120 MPa	$\Delta\sigma'_r = 2m'\sigma_u$
$\Delta\sigma_{er}$	312 MPa	$\Delta\sigma_{er} = 2m\sigma_u$
N'_f	10^3	–
N_e	10^6	–
C	$3.48 \times 10^{19} (\text{MPa})^k$	Eq. (2.69)
k	5.43	Eq. (2.69)

this model. However, for stress signals with various amplitudes, the fatigue limit tends to be less distinct and does not truly follow a constant straight line. In other words, there are no clear limit for which the stress ranges that do not produce any fatigue damage. Therefore, fatigue damage can be initiated even if the stress ranges are located below the defined fatigue limit in 3.15. If many counted stress ranges are located below the fatigue limit one should consider this effect by manipulating the straight line at $N_f > N_e$. On the contrary, if sufficiently many stress ranges are above the fatigue limit, these will neglect the fatigue damage caused by the stress ranges below the fatigue limit. The modelled fatigue limit in Fig. 3.15 is then valid.

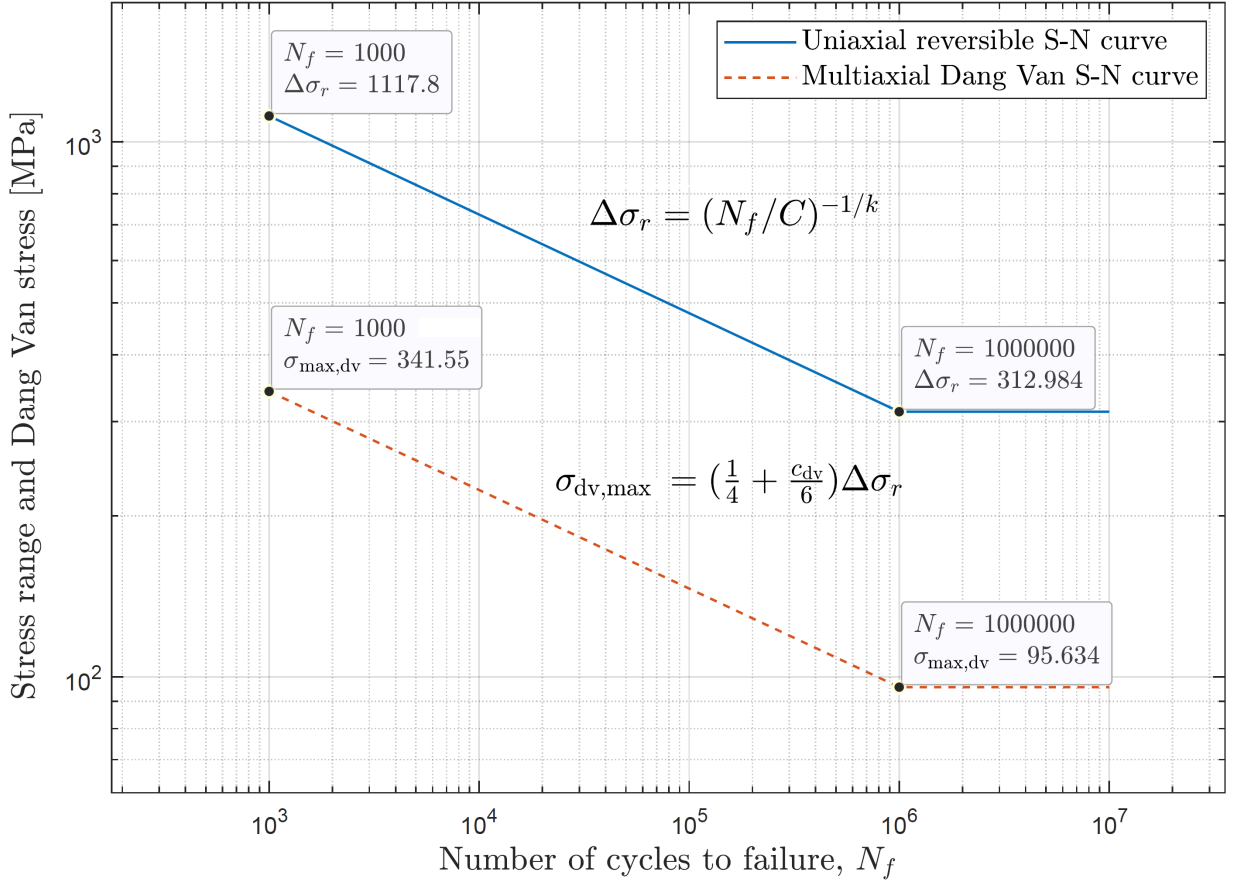


Figure 3.15: Generated reversible S-N curve for the selected steel material with all parameters presented in Table 3.2. The equivalent Dang Van S-N curve is also presented. The S-N curves are assumed to be constant beyond the fatigue limit. Also, the low cycle fatigue $N_f \leq 10^3$ is not presented and will not be considered in this project. (The stress ranges will not exceed the upper stress range limit in this case. In situations when it does, one should apply a low cycle fatigue model instead.)

4 Results and Discussion

In this chapter, results are discussed including figures and tables. Particularly, the fatigue life and damage are compared for the spectral method (Dirlik's formula and EVMS) and transient method (Dang Van, and Wang-Brown). Histograms are also provided to differentiate between the two methods. Time and PSD signals of the stress components, their correlation to one another and other interesting observations are discussed.

4.1 Fatigue Life Comparison

The final step of estimating the life of the selected hotspots are calculated in MATLAB. The 16 cases shown in Fig. 3.2 are noted down in the tabular for each hotspot. The fatigue life from the two methods are displayed side by side in years, days or hours in Table 4.1 when excluding the fatigue limit and in Table 4.2 when the fatigue limit is included in the calculations. Based on the generated input signals, many of the stress ranges are found below the fatigue limit. Therefore, in Table 4.2, many of the loading cases experience infinite fatigue life since almost all stress ranges does not exceed the fatigue limit. In order to make a more thorough comparison, the authors decided to suppress the fatigue limit and extrapolate the S-N curve which gave the result in Table 4.1. An illustrative example is shown in Fig. 4.1. This approach certainly gives a bit too conservative life estimation. However, both methods are applied to the same S-N curve model, so a comparison between the methods can still be made.

According to Table 4.2 and 4.2, the fatigue life between the transient and spectral approach are matching surprisingly well for both uniaxial and multiaxial fatigue. The most critical load scenario of the four loading cases is when all independent acceleration loads a_1 , a_2 and a_3 are applied at the same time. Element 2873 will experience most damage and hence a shortest fatigue life. This could already be predicted from the transfer function plots in Fig. 3.10, since $|H_{11}(f)|$ for element 2873 has the largest peak at the first eigenfrequency.

In all three selected hotspot elements, the shear stress signal σ_{12} does not contribute much to fatigue damage compared to the normal stresses σ_{11} and σ_{22} . This proves that the hotspots only experience a small amount of shear, and a higher amount of tension and compression. Normal stresses exist more at the surface of the component for the lower eigenmodes, since the vibration is causing the air dryer bracket to bend. For element 2562, stress component σ_{22} is contributing most to fatigue damage, while in element 2873 and 3127, stress component σ_{11} is causing most fatigue damage.

The damage ratio between the transient and spectral method for all 16 cases is presented in Table 4.3. For element 2562 the ratio lies close to 1 for both uniaxial and multiaxial fatigue. The largest ratio is equal to 1.30 in case of multiaxial fatigue when only a_1 is applied. Element 2873 is also showing promising results with around 0.7 ratio for the uniaxial cases. However, for multiaxial cases the ratio reaches as much as 3.3 which is a significant different compared to the uniaxial cases. The damage ratio varies mostly in element 3127 and lies in the region 0.068–3.2.

Overall the spectral method is showing similar damage estimate as the transient method, but there are some cases when damage differs significantly. This is the result of that small differences in stress range results in a large difference in damage due to the exponential nature of the S-N curve. In fact, the slope of the S-N curve is playing an important role. A large value of the fitting constant k will give a S-N curve that is sensitive to errors, and a small value of k instead gives a S-N curve that is more resistant to errors. Hence, in order to compare methods, it is essential to know the slope of the S-N curve that has been used. If D_1 and D_2 denote the damage for **one** cycle with the corresponding stress ranges $\Delta\sigma_{r1}$ and $\Delta\sigma_{r2}$, the damage ratio is obtained as

$$\frac{D_1}{D_2} = \left(\frac{\Delta\sigma_1}{\Delta\sigma_2} \right)^k \quad (4.1)$$

The damage ratio for a particular cycle is then equal to the ratio of the corresponding stress ranges to the power of k .

As can be seen in Table 4.1, 4.2 and 4.3, the spectral method is more conservative than the transient method for most cases, both for uniaxial and multiaxial stress state.

Table 4.1: The fatigue life T_{life} presented for all 16 cases in Fig. 3.2 when excluding the fatigue limit. The fatigue life is presented in unit time; either in hours (h), days (d) or years (y). The spectral method (Dirlik's formula and EVMS) is presented without brackets, and the transient method (Dang Van and Wang Brown) are presented inside the brackets.

	a_1	a_2	a_3	a_1, a_2, a_3
Element 2562 (top)				
σ_{11}	680 y (960 y)	2 500 y (3 400 y)	1 800 y (2 400 y)	69 y (92 y)
σ_{22}	110 d (160 d)	6.5 y (8.4 y)	7.6 y (9.7 y)	29 d (40 d)
σ_{12}	280 y (390 y)	140 y (190 y)	100 y (140 y)	7.8 y (10 y)
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	95 d (74 d)	1.4 y (1.8 y)	1.3 y (1.5 y)	13 d (13 d)
Element 2873 (top)				
σ_{11}	1.3 d (1.6 d)	3.0 d (3.9 d)	9.2 d (13 d)	3.6 h (4.6 h)
σ_{22}	220 d (280 d)	230 d (300 d)	1.4 y (1.9 y)	15 d (19 d)
σ_{12}	5 800 y (8 200 y)	23 000 y (39 000 y)	13 000 y (29 000 y)	550 y (940 y)
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	57 h (22 h)	6.3 d (1.9 d)	18 d (5.9 d)	7.1 h (2.3 h)
Element 3127 (bottom)				
σ_{11}	91 d (82 d)	230 d (540 d)	21 d (170 d)	3.1 d (8.0 d)
σ_{22}	5.0 y (1.6 y)	33 y (38 y)	3.2 y (47 y)	130 d (150 d)
σ_{12}	53 000 y (20 000 y)	110 000 y (440 000 y)	35 000 y (70 000 y)	5 000 y (7 300 y)
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	18 d (27 d)	140 d (240 d)	30 d (77 d)	1.7 d (3.2 d)

Table 4.2: The fatigue life T_{life} presented for all 16 cases in Fig. 3.2 when including the fatigue limit. The fatigue life is presented in unit time; either in hours (h), days (d) or years (y). The spectral method (Dirlik's formula and EVMS) is presented without brackets, and the transient method (Dang Van and Wang Brown) are presented inside the brackets. All the input signals a_1 , a_2 and a_3 are generated according to Section 3.1. If the stress ranges do not exceed the fatigue limit, the fatigue life tends to infinity, $T_{\text{life}} \rightarrow \infty$. If so, the fatigue life is left blank.

	a_1	a_2	a_3	a_1, a_2, a_3
Element 2562 (top)				
σ_{11}	— (—)	— (—)	— (—)	— (—)
σ_{22}	190 y (—)	— (—)	— (—)	1.5 y (3.8 y)
σ_{12}	— (—)	— (—)	— (—)	— (—)
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	88 y (—)	— (—)	— (—)	66 d (74 d)
Element 2873 (top)				
σ_{11}	39 h (54 h)	120 h (180 h)	33 d (69 d)	4.0 h (5.2 h)
σ_{22}	— (—)	— (—)	— (—)	87 d (160 d)
σ_{12}	— (—)	— (—)	— (—)	— (—)
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	91 h (27 h)	370 h (68 h)	140 d (17 d)	7.7 h (2.8 h)
Element 3127 (bottom)				
σ_{11}	— (—)	— (—)	4.9 y (—)	8.4 d (33 d)
σ_{22}	— (—)	— (—)	— (—)	3 800 y (—)
σ_{12}	— (—)	— (—)	— (—)	— (—)
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	140 d (770 d)	— (—)	11 y (—)	2.9 d (6.5 d)

Table 4.3: The damage ratio $D_{\text{rain}}/D_{\text{dir}}$ presented for all 16 cases in Fig. 3.2 when excluding the fatigue limit. Damage D_{dir} denote the damage from Dirlik's formula, and D_{rain} is the damage from the rainflow count.

	a_1	a_2	a_3	a_1, a_2, a_3
Element 2562 (top)				
σ_{11}	0.72	0.75	0.75	0.75
σ_{22}	0.71	0.78	0.78	0.73
σ_{12}	0.71	0.74	0.74	0.75
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	1.30	0.76	0.89	0.99
Element 2873 (top)				
σ_{11}	0.77	0.78	0.73	0.77
σ_{22}	0.78	0.76	0.72	0.76
σ_{12}	0.71	0.59	0.43	0.58
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	2.6	3.3	3.1	3.0
Element 3127 (bottom)				
σ_{11}	1.1	0.43	0.12	0.39
σ_{22}	3.2	0.87	0.068	0.84
σ_{12}	2.7	0.25	0.50	0.68
$\sigma_{11}, \sigma_{22}, \sigma_{12}$	0.65	0.59	0.39	0.54

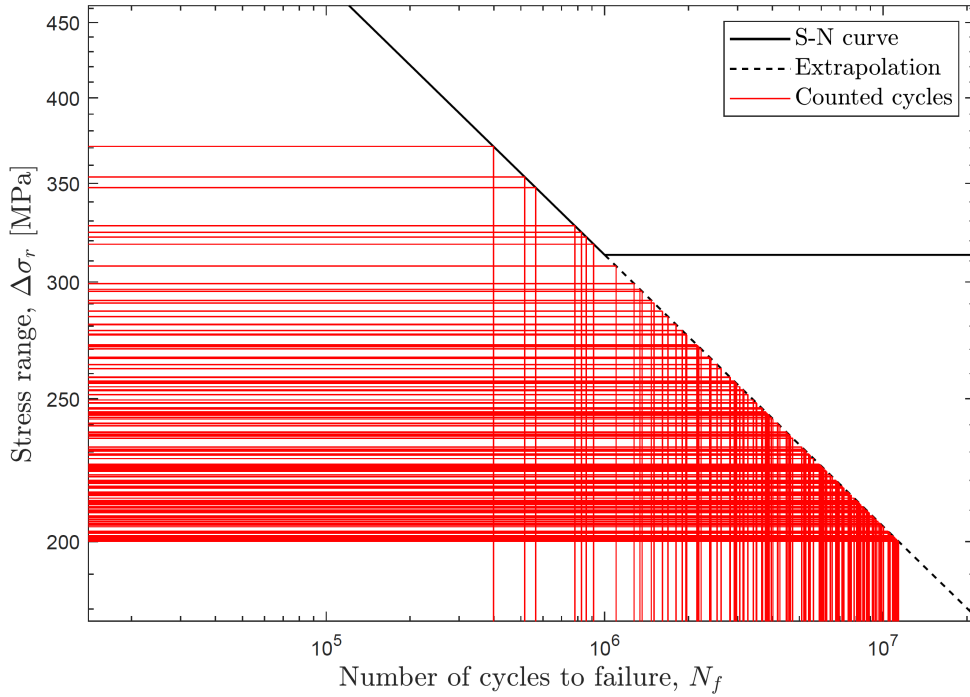


Figure 4.1: An example showing an extrapolated version of the S-N curve. Many counted cycles are located below the fatigue limit in this case. It is therefore convenient to neglect the fatigue limit. The red lines represent some cycle counts for element 2562.

4.2 Histogram Comparison

In order to make a better comparison between the transient and spectral method, their generated histogram plots are presented in Fig. 4.2 and 4.3 for element 2562 and 2873 when all input loads are applied simultaneously. The histograms are showing the number of counted cycles for given stress ranges and are presented for each stress component individually (uniaxial) and all stress components combined (multiaxial).

Interestingly the expected number of counted cycles from Dirlik's formula coincides well with the counted cycles from rainflow count for larger stress ranges. On the other hand, for smaller stress ranges, we can see a significant difference. This difference might seem to be the main cause of the deviation in fatigue damage between the two methods. However, lower stress ranges contribute very little to fatigue damage compared to the larger ones due to the exponential nature of the S-N curve. The difference in fatigue damage will then be smaller for smaller stress ranges and larger for larger stress ranges. In fact, the fatigue damage is more sensitive for deviations at the larger stress ranges, i.e. the error is increasing with the stress range.

The large deviations at the small stress ranges are therefore not as severe as they might seem. Luckily, the Dirlik's formula is showing a promising results for the larger stress ranges. This is however not a coincidence. The parameters in Dirlik's density function are derived based on Monte-Carlo simulations and are designed in a way so that the fatigue damage is matching the fatigue damage of the rainflow count [8]. Therefore, the Dirlik's formula will show a better result for larger stress ranges as shown in Fig. 4.2 and 4.3.

In order to compare spectral and transient methods, it is important that the time signals are sufficiently long. Spectral methods are based on probability theory and density function. If the time signals are too short, the counted cycles will not follow the distribution of the probability density function, giving a large deviation in fatigue life. The counted cycles have to be many. Spectral methods are therefore more suitable for high cycle fatigue compared to low cycle fatigue, where much fewer cycles are being counted. In Fig. 4.2 and 4.3, we can see that the probability that larger stress ranges are being counted is highly reduced. If the time signals are too short, very few cycles with large stress ranges will be counted. In this study however, the generated input signals had a length of $T = 1000$ s and sampling frequency of $f_s = 200$ Hz (Nyquist frequency $f_{ny} = 100$ Hz) which provided many cycle counts with large stress ranges. If the time signals were much shorter, we should expect a larger difference in fatigue life between Dirlik's formula and rainflow count.

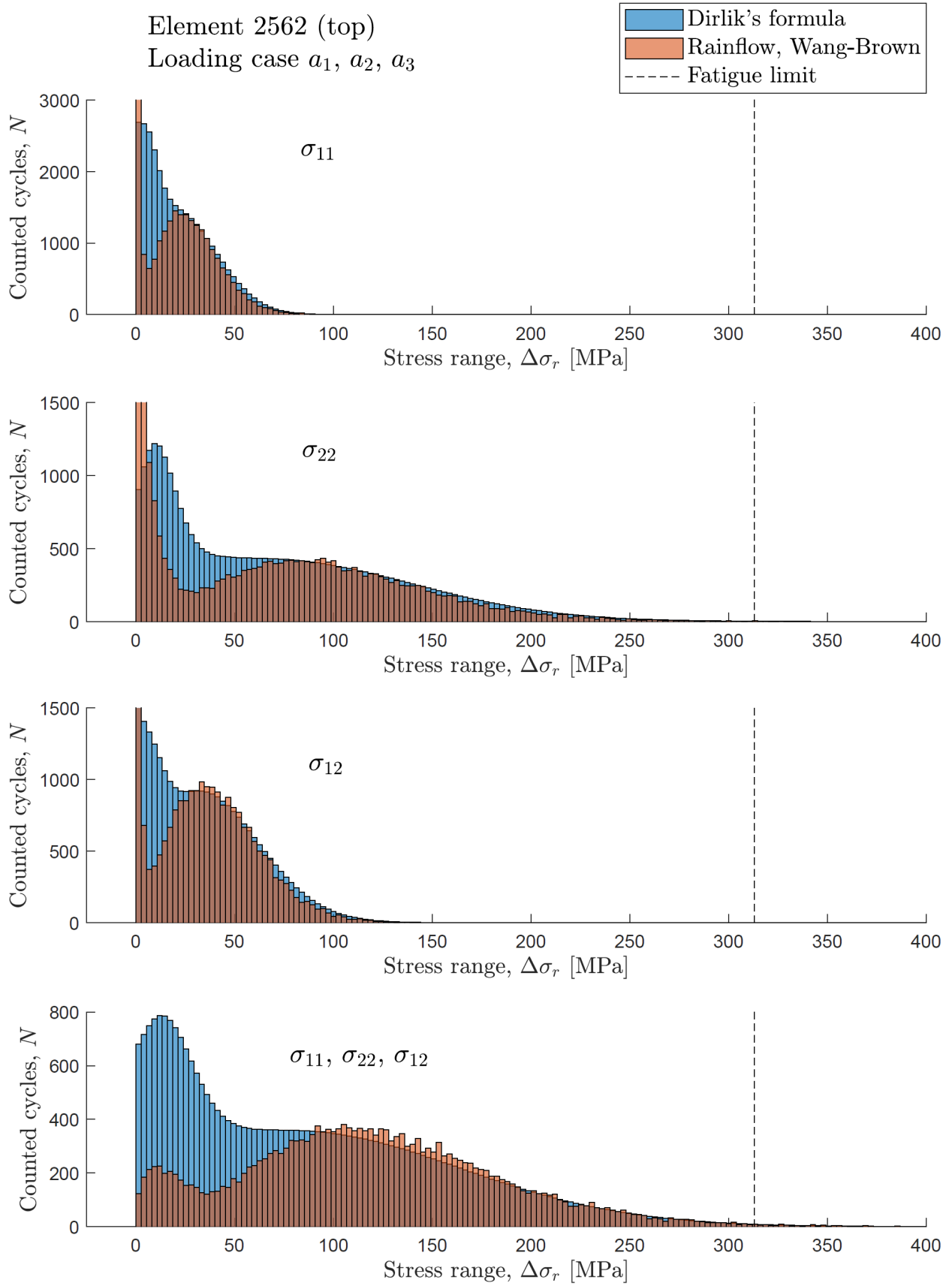


Figure 4.2: Counted cycles N versus stress range $\Delta\sigma_r$ obtained from Dirlik's empirical formula and rainflow count algorithm. The stress signals are taken from element 2562 (top) when all input loads are applied simultaneously.

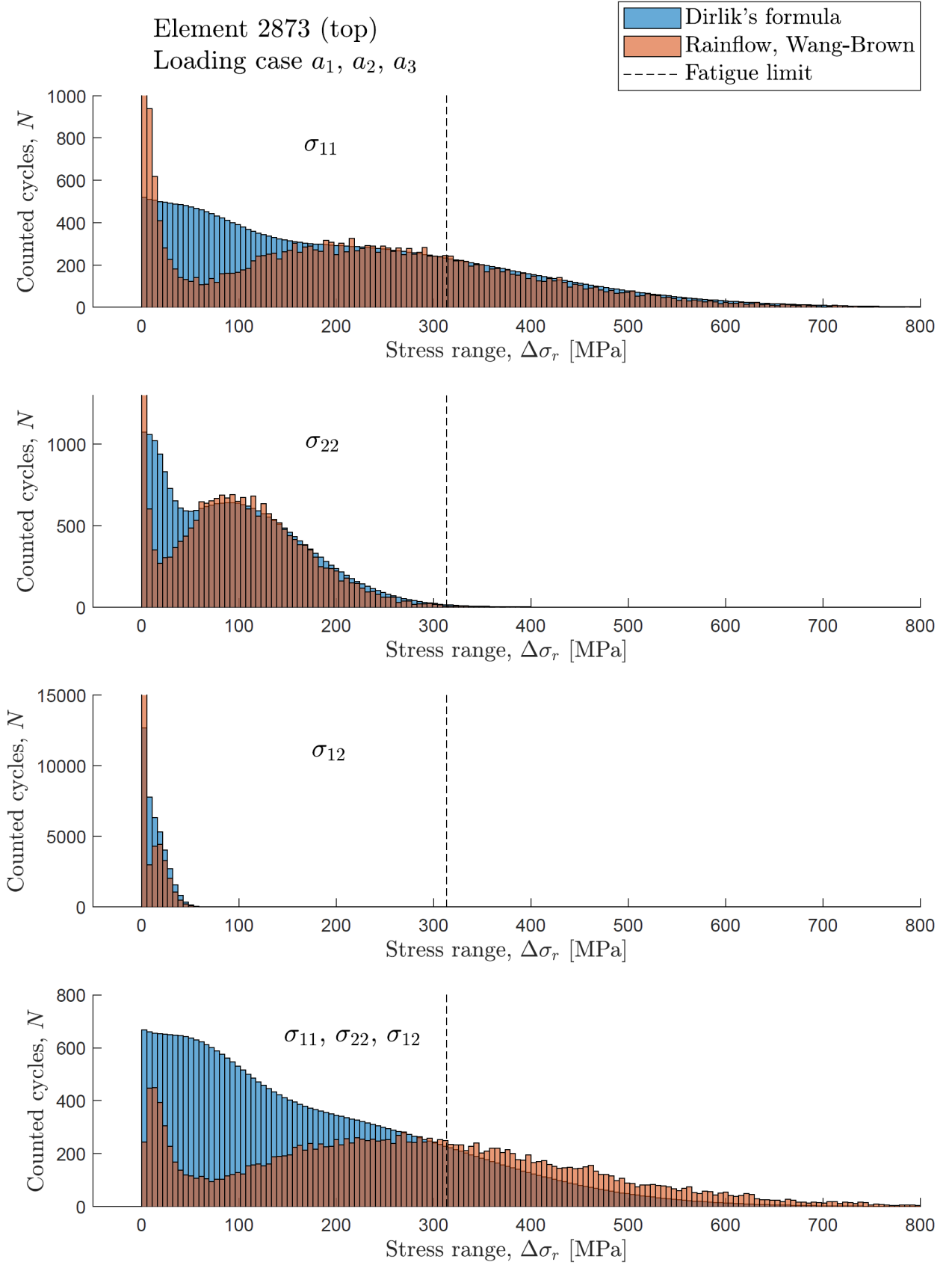


Figure 4.3: Counted cycles N versus stress range $\Delta\sigma_r$ obtained from Dirlik's empirical formula and rainflow count algorithm. The stress signals are taken from element 2873 (top) when all input loads are applied simultaneously.

4.3 Some Interesting Properties of the Stress Signals

Due to linearity, the normal distribution is preserved from input to output, giving normal distributed output stresses with zero mean (see Section 2.4.1). The RMS of the stress components and the von Mises stress is presented in Table 4.4 for all hotspot elements and all loading cases. Comparing Table 4.4 and 4.1 it can be concluded that the RMS have a direct impact on the fatigue life. A larger RMS gives a greater dispersion around the zero mean and therefore larger stress ranges and shorter fatigue life. Note that the RMS of the von Mises stress σ_{vM} is equal to the RMS of the EVMS signal S_{vM} (see definition of EVMS in Section 2.10.4).

Furthermore, in Table 4.5 the correlation factor between the stress components are presented. The correlation between the stress signals are important when considering multiaxial fatigue. A good correlation between the stress components usually results in larger fatigue damage since the peaks and valleys from each stress component are coinciding instead of suppressing each other. In terms of von Mises stress, the worst-case scenario is found when the correlation factor is equal to -1 between all normal stress components, under the assumption that the stress components have zero mean. This can be verified by analysing the mean square of the von Mises stress in case of plane stress:

$$E[\sigma_{\text{vM}}^2] = E[\sigma_{11}^2] + E[\sigma_{22}^2] - \underbrace{E[\sigma_{11}\sigma_{22}]}_{\text{correlation dependency}} + 3E[\sigma_{12}^2] \quad (4.2)$$

Note that the third term in Eq. (4.2) is dependent on the correlation between the normal stresses σ_{11} and σ_{22} . If no correlation exists, the third term is negligible. If the correlation is positive, the mean square of the von Mises stress is decreased. Instead, if the correlation is negative, the mean square are increased. According to Fig. 4.5, stress components σ_{11} and σ_{22} are very close for having correlation factor equal to 1 in element 2873 (top), and so the RMS of the von Mises stress is reduced.

In Fig. 4.4, 4.5 and 4.6 the stress time signals and corresponding PSD signals are shown for element 2562, 2873 and 3127, respectively. The figures are providing some information about the dynamic and frequency response in the selected hotspots when all input loads are applied simultaneously. In Fig. 4.5 we can clearly see that the normal stresses σ_{11} and σ_{22} are almost proportional. The diagonal PSD:s have their peaks are located at the eigenfrequencies of the bracket component. The peaks are relatively thin, which indicates narrowband response. The bandwidth is more precisely estimated in Table 4.6 using the irregularity factors α_1 and α_2 . As mentioned before, irregularity factors close to 0 indicates wideband, while 1 indicates narrowband. The Dirlik's formula works for all irregularities. However, other spectral methods are bounded to perfect narrowband or wideband.

4.4 Observations of Wang and Brown's Method

It should be noted that Wang and Brown's rainflow count algorithm was only validated for uniaxial stress cases, since the authors did not have access to other data. The test was done for many different stress signals, and the authors noted that the algorithm seemed to work much better for narrowband stresses. For wideband stresses the difference between Wang-Brown rainflow count and uniaxial rainflow count became more significant. To be sure that our implementation of Wang and Brown's algorithm does indeed work for multiaxial stress, a deeper study has to be made. According to Table 4.3 the difference in fatigue damage between the rainflow count and Dirlik's formula were looking promising for multiaxial stress at element 2562 and 3127, but not for element 2873. It is hard to confirm that the large difference in element 2873 is due to Wang and Brown's rainflow count since it seems to work for the other elements.

We tried many different projection methods on the von Mises hyper ellipsoid; a both straight line as shown in Fig. 3.13, but also projection of the original path. The projection seemed to give too conservative results, partly because the mid stress of the cycle path was highly increased. This have a direct impact on the maximum Dang Van stress. In this study, order to suppress the impact of mid stress effects, we decided to not project a straight line, but instead to discard the idea of projection.

4.5 Mid Stress Effects

As mentioned in Section 2.10.2, mid stress effect are completely omitted in Dirlik's formula. The maximum Dang Van stress on the other hand does include mid stress effects, even though the equivalent S-N curve is

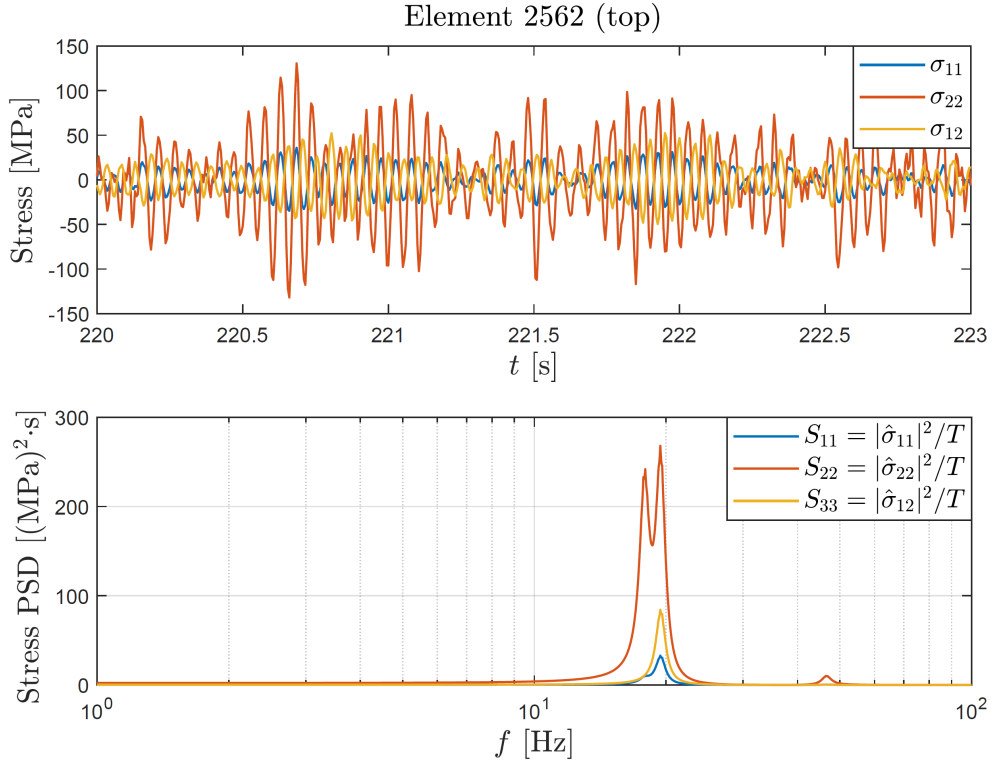


Figure 4.4: Stress time signals and corresponding diagonal PSD stresses for element 2562 when all input loads are applied simultaneously. The time signal are shown for an interval of 3.0 s. The PSD output stresses are based on constant power RMS inputs.

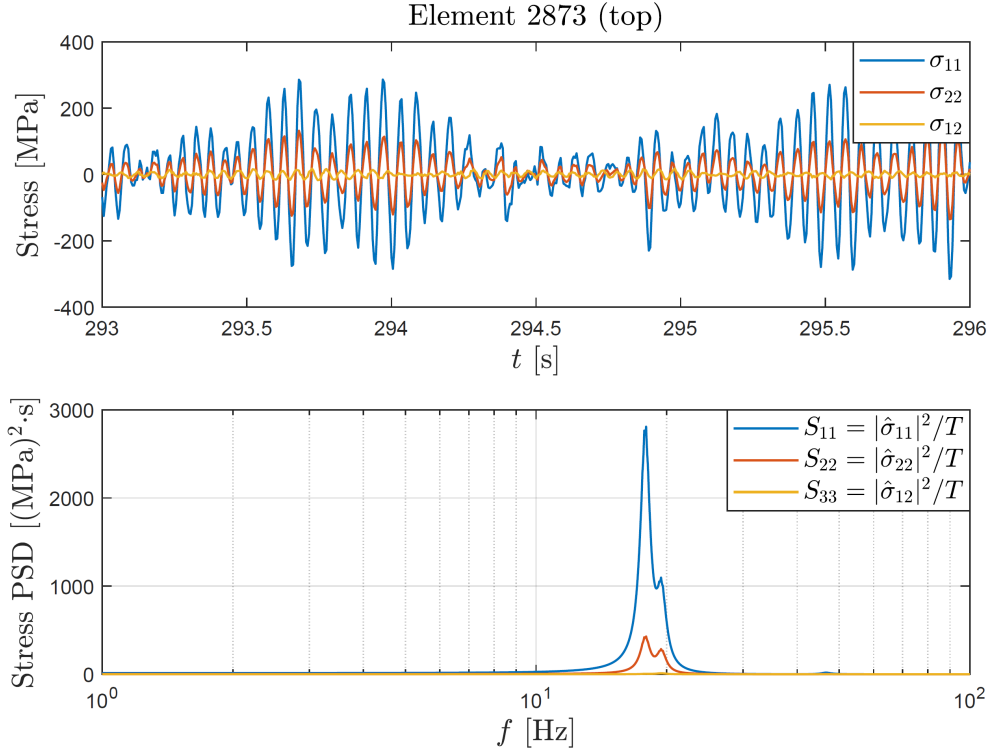


Figure 4.5: Stress time signals and corresponding diagonal PSD stresses for element 2873 when all input loads are applied simultaneously. The time signal are shown for an interval of 3.0 s. The PSD output stresses are based on constant power RMS inputs.

Table 4.4: The RMS (root-mean-square) of stresses σ_{11} , σ_{22} , σ_{12} and σ_{vM} for all loading cases and all hotspot elements. All stress components are normal distributed with zero mean. The RMS was derived from the stress PSD matrix. Similar values were obtained when deriving the RMS from the time signals.

	a_1	a_2	a_3	a_1, a_2, a_3
Element 2562 (top)				
σ_{11}	7.8 MPa	6.1 MPa	6.5 MPa	12 MPa
σ_{22}	32 MPa	18 MPa	18 MPa	41 MPa
σ_{12}	9.1 MPa	10 MPa	11 MPa	18 MPa
σ_{vM} (and S_{vM})	33 MPa	25 MPa	25 MPa	48 MPa
Element 2873 (top)				
σ_{11}	75 MPa	64 MPa	52 MPa	110 MPa
σ_{22}	29 MPa	29 MPa	25 MPa	47 MPa
σ_{12}	5.2 MPa	3.9 MPa	4.3 MPa	7.8 MPa
σ_{vM} (and S_{vM})	66 MPa	56 MPa	45 MPa	98 MPa
Element 3127 (bottom)				
σ_{11}	33 MPa	25 MPa	37 MPa	56 MPa
σ_{22}	19 MPa	12 MPa	17 MPa	28 MPa
σ_{12}	2.3 MPa	2.9 MPa	3.4 MPa	5.0 MPa
σ_{vM} (and S_{vM})	45 MPa	28 MPa	35 MPa	64 MPa

Table 4.5: The correlation factor between stress pairs $(\sigma_{11}, \sigma_{22})$, $(\sigma_{11}, \sigma_{12})$ and $(\sigma_{22}, \sigma_{12})$, for all loading cases and selected hotspot elements. A correlation factor close to 1 or -1 represents positive or negative proportionality between the stress pairs. The correlation factors are derived from the stress PSD matrix.

	a_1	a_2	a_3	a_1, a_2, a_3
Element 2562 (top)				
σ_{11}, σ_{22}	0.96	0.93	0.97	0.94
σ_{11}, σ_{12}	-0.81	-0.94	-0.98	-0.90
σ_{22}, σ_{12}	-0.64	-0.74	-0.91	-0.70
Element 2873 (top)				
σ_{11}, σ_{22}	0.99	0.99	0.99	0.99
σ_{11}, σ_{12}	-0.53	0.35	0.61	0.016
σ_{22}, σ_{12}	-0.38	0.47	0.69	0.18
Element 3127 (bottom)				
σ_{11}, σ_{22}	-0.90	0.026	0.67	-0.11
σ_{11}, σ_{12}	-0.66	-0.73	-0.86	-0.76
σ_{22}, σ_{12}	0.31	-0.25	-0.45	-0.14

Table 4.6: The irregularity factors α_1 and α_2 corresponding to the stress signals σ_{11} , σ_{22} and σ_{12} , and the EVMS signal S_{vM} for all loading cases and selected hotspot elements. Irregularity factors close to 0 indicates wideband and 1 indicates narrowband. The irregularity factors are derived from the stress PSD matrix. The data α_1 is presented without the brackets followed by α_2 within the brackets. The factor α_1 is more sensitive compared to α_2 and will therefore always estimate a larger irregularity.

	a_1	a_2	a_3	a_1, a_2, a_3
Element 2562 (top)				
σ_{11}	0.93 (0.67)	0.98 (0.76)	0.96 (0.61)	0.95 (0.67)
σ_{22}	0.91 (0.63)	0.98 (0.79)	0.98 (0.74)	0.94 (0.66)
σ_{12}	0.96 (0.76)	0.97 (0.74)	0.98 (0.78)	0.97 (0.76)
S_{vM}	0.92 (0.65)	0.98 (0.76)	0.98 (0.76)	0.95 (0.69)
Element 2873 (top)				
σ_{11}	0.95 (0.64)	0.98 (0.88)	0.96 (0.71)	0.96 (0.70)
σ_{22}	0.97 (0.72)	0.97 (0.83)	0.97 (0.83)	0.97 (0.78)
σ_{12}	0.91 (0.54)	0.82 (0.46)	0.76 (0.49)	0.80 (0.43)
S_{vM}	0.95 (0.63)	0.97 (0.76)	0.94 (0.55)	0.95 (0.63)
Element 3127 (bottom)				
σ_{11}	0.89 (0.45)	0.79 (0.64)	0.90 (0.84)	0.80 (0.65)
σ_{22}	0.90 (0.49)	0.78 (0.64)	0.95 (0.92)	0.79 (0.65)
σ_{12}	0.92 (0.48)	0.77 (0.43)	0.78 (0.59)	0.76 (0.50)
S_{vM}	0.92 (0.53)	0.76 (0.53)	0.86 (0.77)	0.76 (0.54)

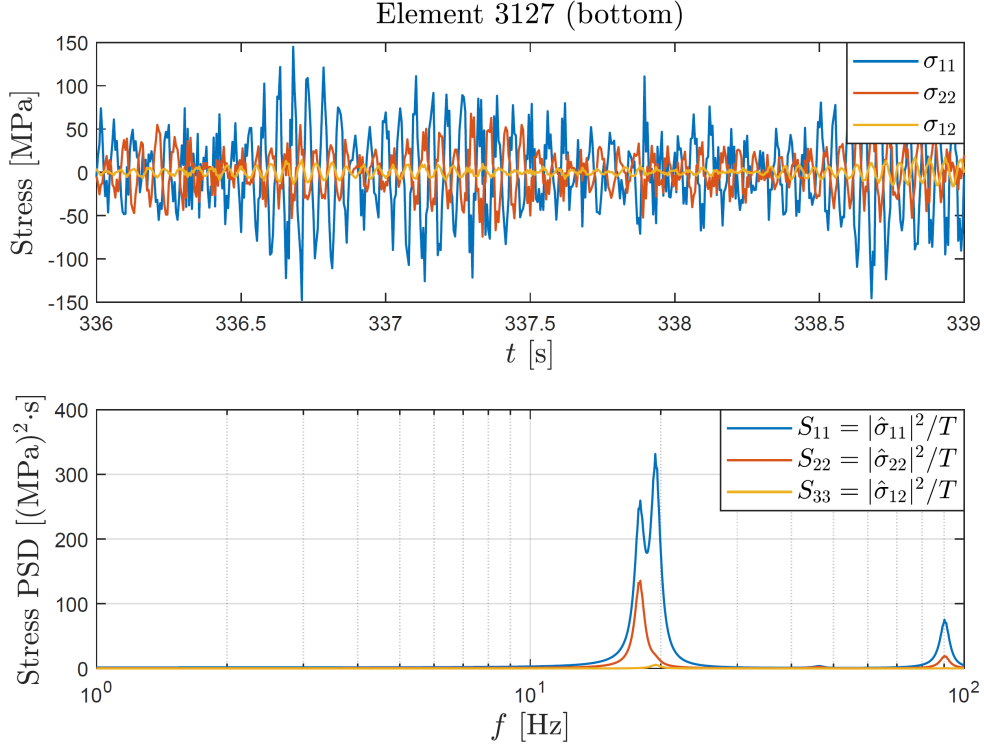


Figure 4.6: Stress time signals and corresponding diagonal PSD stresses for element 3127 when all input loads are applied simultaneously. The time signal are shown for an interval of 3.0 s. The PSD output stresses are based on constant power RMS inputs.

derived assuming reversible stress ranges. This is a big issue and could lead to large differences in fatigue damage. We tried to overcome this problem by suppressing the mid stress contribution in the maximum Dang Van stress, but this does not necessarily solve the problem since the extracted cycle paths might be very long and complex. It is therefore hard to find a convenient mid stress level for each stress component. This is probably the main reason why the difference in fatigue damage prediction is large in some cases.

4.6 Limitations of EVMS

The EVMS is only applicable with success if the ratio between the fatigue limit in the reversible normal S-N curve, $\Delta\sigma_{er}$, and reversible shear S-N curve, $\Delta\tau_{er}$, is equal to $\sqrt{3}$, and if their slopes are equal, $k_\sigma = k_\tau$. A Study done by D. Benasciutti [2] shows that the error in fatigue damage increases if the ratio $\Delta\sigma_{er}/\Delta\tau_{er}$ is further away from $\sqrt{3}$.

According to Dang Van equivalent stress definition (2.79), the Dang Van fatigue limit σ_{edv} can be expressed in terms of $\Delta\sigma_{er}$ and $\Delta\tau_{er}$ giving

$$\begin{cases} \sigma_{edv} = \left(\frac{1}{4} + \frac{c_{dv}}{6}\right) \Delta\sigma_{er} \\ \sigma_{edv} = \frac{1}{2} \Delta\tau_{er} \end{cases} \implies \frac{\Delta\sigma_{er}}{\Delta\tau_{er}} = \frac{6}{3 + 2c_{dv}} \quad (4.3)$$

The Dang Van parameter $c_{dv} = 1/3$ gives a ratio of $\Delta\sigma_{er}/\Delta\tau_{er} = 1.636$, which is very close to $\sqrt{3}$. Hence, it turned out that $c_{dv} = 1/3$ was not really a bad assumption when comparing Dang Van to EVMS. A value of $c_{dv} = 0.232$ gives exactly $\sqrt{3}$ ratio. We tried to implement $c_{dv} = 0.232$ in the calculations. Sure enough, this improved the results, but not by much.

4.7 Computation Time

Since some calculations were done using different FE tools, it is hard to clock both the methods and come down to a point of comparison. In the spectral method, the transfer functions need to be calculated only once in order to perform many realizations. This approach saves a lot of time. But in the transient analysis, entire time history simulations need to be repeated. Also, the computational time depends on the capacity of the CPU that is in use. The NASTRAN runs in this thesis work were remotely run in a HPC (High Performance Computing) that had a clock frequency of 3500 MHz and 16 CPU's. For the spectral analysis, stresses and eigenmodes were calculated in one batch, which gave an average solution run time of only 20 s. For the transient analysis, only stresses were requested which gave an average run time of over 2 min. The time recorded in each domain is only for the 3 hotspots. If an entire model was considered, the transient analysis would definitely take much longer time than the spectral analysis.

In the MATLAB runs we spotted similar observations. The transient approach had to process stress signals with 200 000 data points, while for the spectral approach the frequency response could be presented with less than 1 000 data points. This significance in data size was one of the reasons why the spectral method outperforms the rainflow count in terms of computational efforts. Even though, the uniaxial rainflow count algorithm was proven to be very fast, the PSD:s only need to be integrated in order to receive the moment of areas. The Wang and Brown method, however, was much slower, partly due to the fact that a relative von Mises stress has to be calculated for each cycle path. Also, every cycle path needed to be saved, which results in a larger data storage.

A larger difference in computation time would be present if a larger amount of hotspot elements were considered. As for the entire FE-models, these reduction methods are commonly used to reduce computational effort which will eventually give a less accurate result.

5 Future Scope

This chapter suggests for future scope. There is no straightforward method to calculate random fatigue as the problem description may involve many required conditions like wide band or narrow band spectrum, linearity or nonlinearity material model, coupled with low cycle or high cycle fatigue. Or the problem definition could include material properties such as ductility, brittle or also composite materials etc., for which many researchers have contributed to both transient and spectral domain of random vibration using various methods. There is a lot to explore and investigate in random vibration fatigue.

The limitations mentioned in the Section 1.2 can be a good start for the future work. Small strain theory was assumed here, and the entire analysis was done within the elastic limit. If the stresses were above the yield limit, then the fatigue assessment in both the domains should be done using low cycle fatigue. This could be a topic to include in a future study. In addition to that, for a random vibration, a study on crack propagation using probability theory can be an interesting topic to investigate along with fatigue analysis. Also, the mid stress effects were omitted in this work. Including the mid stress effects, for example in the welded structure, where residual stresses are present, can be seen as an extension for the fatigue assessment. There are ways to apply mid stress corrections to spectral methods, and it could also be an interesting topic to investigate further [22].

The air dryer bracket was a simple example structure considered in the thesis work that had only one interface to excite. The four bolted joints were connected to one master node on which an acceleration input load was applied. Considering a complex structure or a CVM (complete vehicle model) with multiple interfaces that are independent of each other can be very interesting to study. The simulation of vibration on the air dryer bracket was done considering only the translative motion. Including the rotational motion of the bracket in the study can be seen as an extension to this thesis work. Furthermore, as a part of validation study, the component can be physically tested with a dynamic shaker and the results from the simulations can then be compared with it. Also, for a FE model, a parameter study on a mesh refinement, element size, element type for selection of good hotspots can itself have a dedicated study.

Coming to the properties of the input signal, various other spectral methods apart from Dirlik's formula can be studied for a non-Gaussian distribution. There are many spectral methods dedicated for wide band and narrow band separately. Based on the bandwidth size, one can always decide which spectral methods to incorporate in the calculation. As part of future scope, one can incorporate various multiaxial methods (for example critical plane approach) for spectral and transient analysis; and compare them with each other to decide which method gives the best result.

A Code

All code that have been developed for the project are presented in this Chapter; both for MATLAB and MSC NASTRAN. The code include verbose comments. No further explanations will be made.

A.1 MATLAB Code

MATLAB version R2019a was used for this purpose. All developed MATLAB functions are presented. In order to understand the code, one should understand the theory first. Many of the variables are presented with the same variable name as in the theory chapter 2.

A.1.1 Function rainflow_count_halfcycles.m

```
function [sr,sm] = rainflow_count_halfcycles(s)
% [sr,sm] = rainflow_count_halfcycles(s)
%
% PURPOSE: The function calculates the stress range and mid stress for each
%           extracted halfcycle. The halfcycles are extracted using rainflow
%           count algorithm.
%
% INPUT: - s           Stress history saved in a column vector.
%
% OUTPUTS: - sr         Stress range for each stress cycle saved in a column
%                  vector.
%           - sm         Mid stress for each stress cycle saved in a column
%                  vector.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last Modified: 2019-05-20
%

% reduce signal to only peak and valleys:
bol = [true;~(s(1:end-2) <= s(2:end-1) & s(2:end-1) <= s(3:end)) & ...
      ~(s(1:end-2) > s(2:end-1) & s(2:end-1) > s(3:end));true];
s = s(bol);

N = size(s,1); % number of data points
alpha = ones(1,N-1); % interpolation factors

sr = zeros(N-1,1); % stress range values for each cycle
sm = zeros(N-1,1); % mid stress values for each cycles

for i = 1:N-1 % starting point for each cycle
    j = i;
    s1 = s(i); % most left stress value
    s2 = s1; % most right stress value
    bol = s(i) < s(i+1); % is the starting point a maximum (true) or minimum (false)
    while j < N
        % if stress value is larger/smaller than the previous one, update s2
        if (s2 < s(j+1) && bol) || (s(j+1) < s2 && ~bol)
            alphas = (s2-s(j))/(s(j+1)-s(j));
            alpha2 = alpha(j);
            s2 = s(j)+alpha(j)*(s(j+1)-s(j));
            alpha(j) = alphas;
            % if additional point is reached, stop cycle
            if alpha2 ~= 1
                j = N-1;
            end
            % if encounter a smaller/larger stress than the starting point, stop cycle
        elseif (s(j+1) < s1 && bol) || (s1 < s(j+1) && ~bol)
            j = N-1;
        end
        j = j+1;
    end
end
```

```

    sr(i) = abs(s1-s2); % save stress range
    sm(i) = (s1+s2)/2; % save mid stress
end

```

A.1.2 Function `multiaxial_rainflow_count.m`

```

function [s_cycles,s_ind] = multiaxial_rainflow_count(s,s_select)
% function [s_cycles,s_ind] = multiaxial_rainflow_count(s,s_select)
%
% PURPOSE: This function extracts cycles from a multiaxial stress state
%          using Wang and Brown's rainflow count algorithm. The algorithm
%          counts halfcycles using a relative von Mises stress.
%
% INPUTS: - s = [s11 s22 s33 s12 s23 s13]
%          The Voigt notation of the stress tensor. The data points for
%          each stress component are saved column wise. The order in which
%          the stress components are presented is important. If only
%          a portion of stress components are considered, remove all stress
%          components that are of no use. In case of plane stress for example
%          consider s = [s11 s22 s12], or in case of uniaxial stress
%          consider s = s11.
%
%          - s_select = [1 x 6 boolean]
%          A 1x6 boolean matrix that represents the selected stress components
%          in the Voigt notation s = [s11 s22 s33 s12 s23 s13].
%          True indicates that the stress components are considered, while
%          false means that the stress components are removed. For example,
%          in case of plane stress s11, s22 and s12 consider
%          s_select = [true true false true false false]. Instead, in case
%          of uniaxial stress s11 set s_select = [true false false false false false].
%
% OUTPUTS: - s_cycles = [n x 1 cells]
%          The stress values for all stress components for each cycle.
%          For example the stress values along cycles path number i is
%          saved in s_cycles{i} = [s11 s22 s33 s12 s23 s13].
%
%          - s_ind = [n x 1 cells]
%          The indices for each cycle. For example at cycle path number i
%          the indices are saved as a row vector in s_ind{i}. If the indices
%          are decimals, they represent interpolated data points.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-06-02
%

N = size(s,1); % number of data points
N_stress = size(s,2); % number of stress components

% Q-matrix reduction:
Q = [ 1 -1/2 -1/2 0 0 0
      -1/2 1 -1/2 0 0 0
      -1/2 -1/2 1 0 0 0
        0 0 0 3 0 0
        0 0 0 0 3 0
        0 0 0 0 0 3];
Q = Q(s_select,s_select);

alpha = ones(1,N-1); % interpolation factors
s_ind = cell(N-1,1); % path index
track = true(1,N-1); % keep track of false cycles

% count all cycles:
for i = 1:N-1 % starting point for each cycle
    j = i;
    count = 1; % counter
    if alpha(i) == 0
        track(i) = false;
    else
        s_ind{i}(count) = i;
    end
end

```

```

end
if 1 < i && length(s.ind{i-1}) == 1
    track(i-1) = false;
end
sc = 0; % current stress value
while j < N
    s1 = sqrt((s(j,:)-s(i,:))*Q*(s(j,:)-s(i,:))'); % vM stress at j
    s2 = sqrt((s(j+1,:)-s(i,:))*Q*(s(j+1,:)-s(i,:))'); % vM stress at j+1
    if sc < s2 % if the vM stress is increasing do this:
        alpha1 = (sc-s1)/(s2-s1); % interpolation factor
        alpha2 = alpha(j); % old interpolation factor
        sc = s2; % update current stress value
        if alpha2 == 0 % the path from j to j+1 is occupied
            j = N-1;
        elseif alpha1 == 0 && alpha2 == 1 % the path from j to j+1 is free
            count = count+1;
            s.ind{i}(count) = j+1;
            alpha(j) = 0;
        elseif alpha1 == 0 && alpha2 ~= 1 % the path from j to j+1 is partly free
            count = count+1;
            s.ind{i}(count) = j+alpha2;
            alpha(j) = 0;
            j = N-1;
        elseif alpha1 ~= 0 && alpha2 == 1 % the path from j to j+1 is free
            count = count+2; % but the cycle path does not include j
            s.ind{i}(count-1) = j + alpha1;
            s.ind{i}(count) = j + 1;
            alpha(j) = alpha1;
        elseif alpha1 < alpha2 % the path from j to j+1 is partly free
            count = count+2; % but the cycle path does not include j
            s.ind{i}(count-1) = j + alpha1;
            s.ind{i}(count) = j + alpha2;
            alpha(j) = alpha1;
            j = N-1;
        else % if none of the above are fulfilled, stop cycle path
            j = N-1;
        end
    end
    j = j+1; % update j
end
end

% remove false cycles:
s.ind = s.ind(track);

% calculate the stresses in each cycle path:
N.cycles = length(s.ind); % number of cycles that has been extracted
s.cycles = cell(N.cycles,1); % cycle path stress values
for i = 1:N.cycles
    ind_c = s.ind{i};
    n = length(ind_c);
    s.cycles{i} = zeros(fix(ind_c(end))-ind_c(1)+1+length(ind_c(mod(ind_c,1)~=0)),N.stress);
    count = 1;
    s.cycles{i}(count,:) = s(ind_c(1,:));
    for j = 1:n-1
        sL = s(fix(ind_c(j)),:)+mod(ind_c(j),1)*(s(ceil(ind_c(j)),:)-s(fix(ind_c(j)),:));
        sR = s(fix(ind_c(j+1)),:)+mod(ind_c(j+1),1)*(s(ceil(ind_c(j+1)),:)-s(fix(ind_c(j+1)),:));
        C_LR = sqrt((sL-s(ind_c(1),:))*Q*(sL-s(ind_c(1),:))');
        delta_ind = ind_c(j+1)-ind_c(j);
        if delta_ind <= 1 % if two data point are adjacent do this
            count = count+1;
            s.cycles{i}(count,:) = sR;
        else % if not, project the path on the hyper ellipsoid
            k = ind_c(j)+1;
            while k < ind_c(j+1)
                count = count+1;
                f = (k-ind_c(j))/delta_ind;
                vM_c = sqrt((sL+f*(sR-sL)-s(ind_c(1),:))*Q*(sL+f*(sR-sL)-s(ind_c(1),:))');

                % Choose one of the following options:
            end
        end
    end
end

```

```

        % project straight line on hyper ellipsoid from L to R
        %s_cycles{i}(count,:) = (sL+f*(sR-sL))*C_LR/vM_c;

        % project ordinary cycle path on hyper ellipsoid from L to R
        %s_cycles{i}(count,:) = s(k,:)*C_LR/vM_current;

        % straight non-projected line
        %s_cycles{i}(count,:) = sL+f*(sR-sL);

        % ordinary non-projected cycle path from L to R
        s_cycles{i}(count,:) = s(k,:);

        k = k+1;
    end
    count = count+1;
    s_cycles{i}(count,:) = sR; % the last stress values in the cycle path
end
end
end

```

A.1.3 Function rainflow_count.m

```

function [sr,sm] = rainflow_count(s)
% function [sr,sm] = rainflow_count(s)
%
% PURPOSE: Calculate amplitude stresses and mean stresses for a given
%          stress history s. The algorithm are counting whole cycles.
%
% INPUT: - s      Input stress data (column vector)
%
% OUTPUT: - sr     Column vector containing all extracted stress ranges.
%          - sm     Column vector containing all extracted mean stresses.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-04-08
%

% Reduce the signal into just peaks and valleys:
bol = [true;~(s(1:end-2) <= s(2:end-1) & s(2:end-1) <= s(3:end)) & ...
       ~(s(1:end-2) > s(2:end-1) & s(2:end-1) > s(3:end));true];
s = s(bol);

% Uneven number of data points:
if mod(length(s),2) == 0
    s(end) = [];
end

% Rearrange stress vector (if necessary):
[~,ind_max] = max(s);
if ind_max ~= 1 || s(1) ~= s(end)
    s = [s(ind_max:end);s(2:ind_max)];
end

% Find amplitude and mean stresses for each cycle:
N = length(s); % number of data points
N_cycles = (N-1)/2; % number of cycles
sr = zeros(N_cycles,1); % stress amplitude
sm = zeros(N_cycles,1); % mid stress
track = true(1,N); % track which points that are discarded
num_points = N; % current number of points

counter = 1;
while(num_points >= 3)
    i = 1;
    while(i+2 <= N)
        while ~track(i) && i+2 < N
            i = i+1;
        end
        j = i+1;
    end
end

```

```

while ~track(j) && j+1 < N
    j = j+1;
end
k = j+1;
while ~track(k) && k < N
    k = k+1;
end

if abs(s(i)-s(j)) <= abs(s(j)-s(k)) && track(i) && track(j) && track(k)
    sr(counter) = abs(s(i)-s(j));
    sm(counter) = (s(i)+s(j))/2;
    track(i) = false;
    track(j) = false;
    num_points = num_points-2;
    counter = counter+1;
end
i = i+1;
end
end

```

A.1.4 Function Dirliks_formula.m

```

function [cycle_counts,sr_edges,T_life,T_life_red] = Dirliks_formula(m,T,SN_data,nbins)
% [cycle_counts,sr_edges,T_life,T_life_red] = Dirliks_formula(m,T,SN_data,nbins)
%
% PURPOSE: This function estimates the number of stress cycles
%           within given stress intervals and fatigue life using moment of
%           area for an uniaxial PSD stress in frequency domain. The probability
%           density function between the number of cycles and stress values
%           is obtained using Dirlik's empirical formula.
%
% INPUT: - m = [m0 m1 m2 m4]           Moment of areas (0, 1, 2, 4 order).
%         - T           Total time elapsed.
%         - SN_data = [sr_min sr_max C k] Fatigue parameters parameters
%                                           where sr_min and sr_max are the
%                                           upper and lower stress range
%                                           limit. C and k are fitting
%                                           parameters; Nf = C*sr^(-k).
%         - nbins       Number of bins (number of stress range intervals).
%
% OUTPUT: - cycle_counts   Number of cycles for each stress range (nbins x 1).
%         - sr_edges       Stress range edges (nbins+1 x 1).
%         - T_life         Fatigue life excluding the limits.
%         - T_life_red     Fatigue life including the limits.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-04-22
%
% Moments of area:
m0 = m(1);
m1 = m(2);
m2 = m(3);
m4 = m(4);

% Expected number of peaks per unit time:
E_p = sqrt(m4/m2);

% Constants in Dirlik's formula:
x_m = (m1/m0)*sqrt(m2/m4);
gamma1 = m2/sqrt(m0*m4);
D1 = 2*(x_m-gamma1^2)/(1+gamma1^2);
R = (gamma1-x_m-D1^2)/(1-gamma1-D1+D1^2);
D2 = (1-gamma1-D1+D1^2)/(1-R);
D3 = 1-D1-D2;
Q = 1.25*(gamma1-D3-D2*R)/D1;

% Load fatigue parameters:
sr_min = SN_data(1);

```

```

sr_max = SN.data(2);
C = SN.data(3);
k = SN.data(4);

% Determine bin_count and sr_edges:
sr_edges = linspace(sr_min,sr_max,nbins+1)';
cycle_counts = zeros(nbins,1);
delta_u = (sr_max-sr_min)/nbins;
for i = 1:nbins
    u = sr_min + delta_u*(i-1/2);
    Z = u/(2*sqrt(m0));
    p = (D1/Q*exp(-Z/Q) + D2*Z/R^2*exp(-Z^2/(2*R^2)) + D3*Z*exp(-Z^2/2))/(2*sqrt(m0));
    cycle_counts(i) = E.p*T*p*delta_u;
end

% Fatigue life:
T.life = C*m2^0.5/(m4^0.5*m0^(k/2)*(2^k*Q^k*D1*gamma(k+1)+2^(3*k/2)*(abs(R)^k*D2+D3)*gamma(k/2+1)));

% Reduced fatigue life:
sr = (sr_edges(1:end-1)+sr_edges(2:end))/2;
T.life_red = C*T/sum(cycle_counts.*sr.^k);

```

A.1.5 Function fatigue_EVMS_Dirlik.m

```

function [T.life,T.life_red,corr_mat,alpha,sr_edges,cycle_counts] = ...
    fatigue_EVMS_Dirlik(S,f,s_select,T,SN.data,nbins)
% [T.life,T.life_red,corr_mat,alpha,sr_edges,cycle_counts] = ...
%     fatigue_EVMS_Dirlik(S,f,s_select,T,SN.data,nbins)
%
% PURPOSE: This function estimates the expected number of cycle counts and
% calculate the expected fatigue life for a
% stress PSD matrix S. The fatigue life is
% calculated using Dirlik's empirical formula for each stress component
% individually (uniaxial stress) and all together (multiaxial stress).
% In case of multiaxial stress, the stress PSD is reduced using
% "Equivalent von Mises stress" (EVMS).
%
% INPUTS: - S = [S.11 S.12 ... S.1n S.21 S.22 ... S.2n ... S.3n ... S.nn]
% Stress PSD matrix (square matrix). The stress PSD components
% are saved column wise. Note that the stress PSD matrix have to
% represent a square matrix with n^2 elements. Also, the PSD:s
% must only be defined on the positive side of the frequency
% spectrum.
%
% - f
% The frequency data saved in a column vector for all the PSD:s.
% No negative frequencies are allowed. The column vector needs to
% have the same length as the columns in S.
%
% - s_select = [1 x 6 boolean]
% A 1x6 boolean matrix that represents the selected stress components
% in the Voigt notation s = [s11 s22 s33 s12 s23 s13].
% True indicates that the stress components are considered, while
% false means that the stress components are removed. For example,
% in case of plane stress s11, s22 and s12 consider
% s_select = [true true false true false false]. Instead in case
% of uniaxial stress s11 set s_select = [true false false false false false].
%
% - T (scalar)
% Total time elapsed (time length of the stress components)
%
% - SN.data = [sr_min sr_max C k]
% Row vector containing all fatigue data necessary for this
% purpose, where sr_min is the lower stress range limit, sr_max
% the upper stress range limit, C and k are fitting parameters
% to the reversible S-N curve such that Nf = C*sr^(-k).
%
% - nbins (scalar)
% Number of nbins, i.e. the number of stress range intervals

```



```

%           for the histogram plots.
%
% OUTPUTS: - T_life = [T_1 T_2 T_3 T_4 T_5 T_6 T_all] ([1 x N+1])
%           The fatigue life for each stress components individually and
%           all together when no restrictions are made. They are stored in
%           a row vector. The values T1-T6 corresponds to each stress
%           component, and T_all is the fatigue life for the multiaxial
%           stress state. In case of plane stress for example, T_life contains
%           3+1 components, T_life = [T_1 T_2 T_4 T_all].
%
%           - T_life_red = [T_1 T_2 T_3 T_4 T_5 T_6 T_all] ([1 x N+1])
%           The same as T_life, but does consider the stress range limits
%           sr_min and sr_max. Everything outside sr_min <= sr <= sr_max
%           will not contribute to any fatigue damage.
%
%           - corr_mat = [N x N]
%           Correlation matrix between all stress components. The diagonal
%           is always equal to 1. The correlation matrix is also
%           symmetric. (Number of stress components N).
%
%           - alpha = [alpha1_1 alpha1_2 ... alpha1_5 alpha1_6 alpha1_all
%                      alpha2_1 alpha2_2 ... alpha2_5 alpha2_6 alpha2_all]
%           The first and second irregularity factor for each stress component
%           and the EVMS. The first order irregularity factor is saved in the
%           first row, and the second order irregularity factor in the
%           second row.
%
%           - sr_egdes = [nbins+1 x 1]
%           The stress range edges for the histograms
%
%           - cycle_counts = [nbins x N+1]
%           Number of cycles that has been counted in each bin for each
%           stress component (uniaxial stress) and also for the multiaxial
%           stress. All the counted cycles are saved column wise.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-06-13
%

N = sqrt(size(S,2)); % number of stress components
T_life = zeros(1,N+1); % fatigue life
T_life_red = zeros(1,N+1); % fatigue life when including the limits
cycle_counts = zeros(nbins,N); % counted cycles

cov_mat = zeros(N); % covariance matrix of output (assume zero mean)
for i = 1:N
    for j = 1:N
        cov_mat(i,j) = 2*trapz(f,real(S(:,N*(i-1)+j))));
    end
end

s_RMS = sqrt(diag(cov_mat))'; % standard deviation of output stresses

corr_mat = zeros(N); % correlation matrix of output (assume zero mean)
for i = 1:N
    for j = 1:N
        corr_mat(i,j) = cov_mat(i,j)/sqrt(cov_mat(i,i)*cov_mat(j,j));
    end
end

s_ind = [11 22 33 12 23 13]; % stress indices
s_ind = s_ind(s_select); %

% 0, 1, 2, 4 moment of area:
m0 = 2*trapz(f,real(S));
m1 = 2*trapz(f,real(S).*f);
m2 = 2*trapz(f,real(S).*f.^2);
m4 = 2*trapz(f,real(S).*f.^4);

% Q-matrix:
Q = [ 1 -1/2 -1/2 0 0 0

```

```

-1/2  1  -1/2  0  0  0
-1/2 -1/2  1  0  0  0
  0    0    0  3  0  0
  0    0    0  0  3  0
  0    0    0  0  0  3];
Q = Q(s.select,s.select); % reduced Q-matrix

% uniaxial PSD, Dirlik's formula:
figure
for i = 1:N
    m = [m0(N*(i-1)+i),m1(N*(i-1)+i),m2(N*(i-1)+i),m4(N*(i-1)+i)];
    [cycle_counts(:,i),sr_edges,T_life(i),T_life_red(i)] = Dirliks_formula(m,T,SN_data,nbins);
    subplot(N+1,1,i);
    hold on
    histogram('binEdges',sr_edges','binCounts',cycle_counts(:,i));
    xlabel('stress range')
    ylabel('expected nr of cycles')
    title(['Stress component \sigma_{',num2str(s_ind(i)),'}'])
end

% EVMS PSD, Dirlik's formula:
m0_vM = 0;
m1_vM = 0;
m2_vM = 0;
m4_vM = 0;
for i = 1:N
    for j = 1:N
        m0_vM = m0_vM + Q(i,j)*m0(N*(i-1)+j);
        m1_vM = m1_vM + Q(i,j)*m1(N*(i-1)+j);
        m2_vM = m2_vM + Q(i,j)*m2(N*(i-1)+j);
        m4_vM = m4_vM + Q(i,j)*m4(N*(i-1)+j);
    end
end
m = [m0_vM,m1_vM,m2_vM,m4_vM];
[cycle_counts(:,end),sr_edges,T_life(N+1),T_life_red(N+1)] = Dirliks_formula(m,T,SN_data,nbins);
subplot(N+1,1,N+1);
hold on
histogram('binEdges',sr_edges','binCounts',cycle_counts(:,end));
xlabel('stress range')
ylabel('expected nr of cycles')
title('All stress component')

% Irregularity factors:
alpha = zeros(2,N+1);
alpha(1,1:end-1) = m1(1:N+1:end)/sqrt(m0(1:N+1:end).*m2(1:N+1:end));
alpha(2,1:end-1) = m2(1:N+1:end)/sqrt(m0(1:N+1:end).*m4(1:N+1:end));
alpha(1,end) = m1_vM/sqrt(m0_vM*m2_vM);
alpha(2,end) = m2_vM/sqrt(m0_vM*m4_vM);

disp(' ')
disp('-----fatigue-EVMS-Dirlik-----')
disp(' ')
disp('RMS of output signals and EVMS: ')
disp([s_RMS,sqrt(m0_vM)])
disp('Correlation matrix of output signals: ')
disp(corr_mat)
disp('Irregularity factors: ')
disp('alpha.1 = ')
disp(alpha(1,:))
disp('alpha.2 = ')
disp(alpha(2,:))
disp('T_life (hours): ')
disp(T_life/(60*60))
disp('T_life.red (hours): ')
disp(T_life_red/(60*60))
disp('-----end-----')
disp(' ')

```

A.1.6 Function DangVan.m

```

function [s_dv_max,s_dv] = DangVan(s,s_select,c_dv,midEff)
% function [s_dv_max,s_dv] = DangVan(s,s_select,c_dv,midEff)
%
% PURPOSE: This function calculates the Dang Van equivalent stress for
%           given stress cycle.
%
% INPUTS:  - s           The Voigt notation of a stress tensor for all
%                       instants in time in a given stress cycle. It is defined
%                       as a n x 6 matrix where n is the number of data points.
%                       s(i,:) = [s11 s22 s33 s12 s23 s13].
%
%           - s_select   Selection of stress components (true or false). For
%                       example s_select = [true true false true false false]
%                       selects stress component s11, s22 and s12.
%
%           - c_dv       Coefficient in Dang Van criterion.
%
%           - midEff     Mid stress effects? Yes (true) or No (false).
%
% OUTPUT:  - s_dv_max    Maximum Dang Van equivalent stress.
%
%           - s_dv       Dang Van equivalent stress for each instant in time.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-04-25
%

N = size(s,1); % number of data points

s_ext = zeros(N,6); % extended voight notation
s_ext(:,s_select) = s; %

s_dv = zeros(N,1); % initiate Dang Van equivalent stress

% hydrostatic stress:
sh = (s_ext(:,1)+s_ext(:,2)+s_ext(:,3))/3;

% deviatoric stress:
sd = s_ext;
sd(:,1:3) = sd(:,1:3) - sh;

% Find appropriate mid value of the deviatoric stress by minimizing J2.
% Two options:

% find smid by calculating the mean of sd (fast):
smid = mean(sd);

% find smid so that the amplitude von Mises stress is minimized (slow):
% smid = fminsearch(@(smid) J2(smid,sd), (max(sd)+min(sd))/2);

smid_tensor = [smid(1) smid(4) smid(6) %
               smid(4) smid(2) smid(5) %
               smid(6) smid(5) smid(3)]; % mid value of deviatoric stress

% Loop through all time data:
for i = 1:size(s,1)
    s_tensor = [s_ext(i,1) s_ext(i,4) s_ext(i,6) %
                s_ext(i,4) s_ext(i,2) s_ext(i,5) %
                s_ext(i,6) s_ext(i,5) s_ext(i,3)]; % stress tensor

    sd_tensor = s_tensor - sh(i)*diag([1 1 1]); % deviatoric stress tensor

    sa_tensor = sd_tensor - smid_tensor; % deviatoric "amplitude" stress

    pr = eig(sa_tensor); % principal deviatoric stresses

    tau_Tr = max(abs([pr(1)-pr(2),pr(1)-pr(3),pr(2)-pr(3)]))/2; % maximum shear

```

```

    s_dv(i) = tau.Tr + c_dv*sh(i); % Dang Van equivalent stress
end

if ~midEff
    s_dv = s_dv - c_dv*mean(sh); % suppress mid stress effects if midEff = true
end

s_dv_max = max(s_dv); % maximum value of Dang Van equivalent stress
if s_dv_max < 0
    warning('The maximum Dang Van stress is negative')
end

end

function s_vM = J2(smld,sd)
% function s_vM = J2(smld,sd)
%
% PURPOSE: Find the maximum "von Mises" stress for sd-smld
%
% INPUT: - smld      mid deviatoric stress (1x6)
%         - sd        deviatoric stress (nx6)
%
% OUTPUT: - s_vM      the maximum "von Mises" stress for sd-smld
%
sa = sd - smld; % deviatoric "amplitude" stress

sJ2 = zeros(size(sa,1),1);
for i=1:size(sa,1)
    s = [sa(i,1) sa(i,4) sa(i,6)
         sa(i,4) sa(i,2) sa(i,5)
         sa(i,6) sa(i,5) sa(i,3)];
    sJ2(i) = sqrt(3/2*sum(sum(s.*s)));
end
s_vM = max(sJ2);
end

```

A.1.7 Function fatigue_WangBrown_DangVan.m

```

function [Tlife,Tlife_red,corr_mat,sr_edges,cycle_counts] = ...
    fatigue.WangBrown_DangVan(s,s_select,T,SN_data,nbins)
% function [Tlife,Tlife_red,corr_mat,sr_edges,cycle_counts] =
%         fatigue.WangBrown_DangVan(s,s_select,T,SN_data,nbins)
%
% PURPOSE: This function counts cycles and estimates fatigue life for a
%           set of stress components stored in s. The fatigue life is
%           calculated for each stress component individually (uniaxial stress)
%           and all together (multiaxial stress). The multiaxial stress
%           are handled by Wang and Brown's method and Dang Van equivalent
%           stress.
%
% INPUTS: - s = [s11 s22 s33 s12 s23 s13]
%           Stress tensor in Voigt notation. The data for each stress
%           components are stored column wise. The order in which
%           the stress components are presented is important. If only
%           a portion of stress components are considered, remove all stress
%           components that are of no use. In case of plane stress for example
%           consider s = [s11 s22 s12], or in case of uniaxial stress
%           consider s = s11.
%
%           - s_select = [1 x 6 boolean]
%           A 1x6 boolean matrix that represents the selected stress components
%           in the Voigt notation s = [s11 s22 s33 s12 s23 s13].
%           True indicates that the stress components are considered, while
%           False means that the stress components are removed. For example,
%           in case of plane stress s11, s22 and s12 consider
%           s_select = [true true false true false false]. Instead in case
%           of uniaxial stress s11 set s_select = [true false false false false false].

```

```

%
% - T (scalar)
%     Total time elapsed (time length of the stress components)
%
% - SN_data = [sr_min sr_max C k c_dv]
%     Row vector containing all fatigue data necessary for this
%     purpose, where sr_min is the lower stress range limit, sr_max
%     the upper stress range limit, C and k are fitting parameters
%     to the reversible S-N curve such that  $N_f = C \cdot sr^{-k}$ , and c_dv
%     is the Dang Van parameter.
%
% - nbins (scalar)
%     Number of nbins, i.e. the number of stress range intervals
%     for the histogram plots.
%
% OUTPUTS: - T_life = [T_1 T_2 T_3 T_4 T_5 T_6 T_all] ([1 x N+1])
%     The fatigue life for each stress components individually and
%     all together when no restrictions are made. They are stored in
%     a row vector. The values T1-T6 corresponds to each stress
%     component in s, and T_all is the fatigue life for the multiaxial
%     stress state. In case of plane stress for example, T_life contains
%     3+1 components, T_life = [T_1 T_2 T_4 T_all].
%
% - T_life_red = [T_1 T_2 T_3 T_4 T_5 T_6 T_all] ([1 x N+1])
%     The same as T_life, but does consider the stress range limits
%     sr_min and sr_max. Everything outside  $sr_{min} \leq sr \leq sr_{max}$ 
%     will not contribute to any fatigue damage.
%
% - corr_mat = [N x N]
%     Correlation matrix between all stress components. The diagonal
%     is always equal to 1. The correlation matrix is also
%     symmetric. (Number of stress components N).
%
% - sr_egdes = [nbins+1 x 1]
%     The stress range edges for the histograms
%
% - cycle_counts = [nbins x N+1]
%     Number of cycles that has been counted in each bin for each
%     stress component (uniaxial stress) and also for the multiaxial
%     stress. All the counted cycles are saved column wise.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-06-13
%

N = size(s,2); % number of stress components
T_life = zeros(1,N+1); % fatigue life
T_life_red = zeros(1,N+1); % fatigue lifes when including the limits
cycle_counts = zeros(nbins,N); % counted cycles

s_RMS = zeros(1,N); % RMS of each stress components
cov_mat = zeros(N); % covariance matrix
for i = 1:N
    s_RMS(i) = sqrt(mean(s(:,i).*s(:,i)));
    for j = 1:N
        cov_mat(i,j) = mean(s(:,i).*s(:,j))-mean(s(:,i))*mean(s(:,j));
    end
end

corr_mat = zeros(N); % correlation matrix
for i = 1:N
    for j = 1:N
        corr_mat(i,j) = cov_mat(i,j)/sqrt(cov_mat(i,i)*cov_mat(j,j));
    end
end

s_ind = [11 22 33 12 23 13]; % stress indices
s_ind = s_ind(s_select); %

count = 0;
c_pro = zeros(N*(N-1)/2,1);

```

```

figure
for i = 1:N-1
    for j = i+1:N
        count = count + 1;

        % optimized proportional constants, least square method:
        c_pro(count) = (s(:,i)'*s(:,i))\ (s(:,i)'*s(:,j));

        % scatter plot
        subplot(N*(N-1)/2,1,count)
        plot(s(:,i),s(:,j),'.k')
        hold on
        plot(1.5*[min(s(:,i)) max(s(:,i))],c_pro(count)*1.5*[min(s(:,i)) max(s(:,i))], '--r')
        xlabel(['\sigma-',num2str(s_ind(i)),'])')
        ylabel(['\sigma-',num2str(s_ind(j)),'])')
        title(['Scatter plot, stress elements ', ...
            '\sigma-',num2str(s_ind(i)),'] and ', '\sigma-',num2str(s_ind(j)),'])')
    end
end

% load fatigue parameters
sr_min = SN_data(1);
sr_max = SN_data(2);
C = SN_data(3);
k = SN_data(4);
c_dv = SN_data(5);
sr_edges = linspace(sr_min,sr_max,nbins+1);

% Uniaxial rainflow count:
figure
for i = 1:N
    sr = rainflow_count_halfcycles(s(:,i));
    cycle_counts(:,i) = histcounts(sr,sr_edges)/2;
    subplot(N+1,1,i)
    hold on
    histogram('BinEdges',sr_edges,'BinCounts',cycle_counts(:,i))
    title(['Stress component \sigma-',num2str(s_ind(i)),'])')
    xlabel('stress range')
    ylabel('counted cycles')
    T_life(i) = 2*C*T/sum(sr.^k);
    T_life_red(i) = 2*C*T/sum(sr(sr_min <= sr & sr <= sr_max).^k);
end

% Multiaxial rainflow count, Wang and Brown's method:
s_cycles = multiaxial_rainflow_count(s,s_select);

% Maximum Dang Van stress for each extracted cycle:
N_cycles = length(s_cycles); % number of counted cycles
s_dv_max = zeros(N_cycles,1); % maximum Dang Van stress saved in s_dv_max
for i = 1:N_cycles
    s_dv_max(i) = DangVan(s_cycles{i},s_select,c_dv,false);
end

sr = abs(s_dv_max)/(1/4+c_dv/6); % convert from maximum Dang Van to reversible stress

% fatigue life from multiaxial stress:
T_life(N+1) = 2*C*T/sum(sr.^k);
T_life_red(N+1) = 2*C*T/sum(sr(sr_min <= sr & sr <= sr_max).^k);

cycle_counts(:,end) = histcounts(sr,sr_edges)/2;
subplot(N+1,1,N+1)
hold on
histogram('BinEdges',sr_edges,'BinCounts',cycle_counts(:,end))
title('All stress components')
xlabel('stress range')
ylabel('counted cycles')

disp(' ')
disp('-----fatigue.WangBrown.DangVan-----')
disp(' ')
disp('RMS of all stress components: ')

```

```

disp(s_RMS)
disp('Correlation matrix: ')
disp(corr_mat)
disp('T_life (hours): ')
disp(T_life/(60*60))
disp('T_life.red (hours): ')
disp(T_life.red/(60*60))
disp('-----end-----')
disp(' ')

```

A.1.8 Function time2PSDmatrix.m

```

function [G,f] = time2PSDmatrix(u,T,win)
% [G,f] = PSD_matrix(u,T,win)
%
% PURPOSE: Estimates the power spectral density (PSD) matrix G of the time
%           signals u. The PSD is calculated using Welch's method with Hamming
%           window function. For example in case of 3 signals we receive 3x3
%           PSD matrix:
%
%           [u_1      [G_11 G_12 G_13
%             u_2      ---->  G =  G_21 G_22 G_23
%             u_3]      G_31 G_32 G_33]
%
% INPUTS: - u = [u_1 u_2 ... u_n]
%           Time signals stored column by column (real time signals).
%
%           - T (scalar)
%           Total time elapsed (time length of the signals u)
%
%           - win = [N_win D_over Nf]
%           Row vector containing window parameters. N_win is the number of
%           windows that the signals are divided into, D_over is the overlapping
%           ratio (between 0 and 1), and Nf is the number of frequency data
%           points of the PSD matrix. Only the positive frequency spectrum
%           is considered (if the signals u are real).
%
% OUTPUTS: - G = [G_11 G_12 ... G_1n G_21 G_22 ... G_2n ... G_3n ... G_nn]
%           Power spectral density matrix. All components are saved column
%           by column. For example a 3x3 PSD matrix gives
%           G = [G_11 G_12 G_13 G_21 G_22 G_23 G_31 G_32 G_33].
%
%           - f
%           Frequency data saved as a column vector.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-06-12
%

Nt = size(u,1); % number of data points
N = size(u,2); % number of signals

dt = T/(Nt-1); % time step
f_s = 1/dt; % sampling frequency
f_ny = f_s/2; % Nyquist frequency

N_win = win(1); % number of windows
D_over = win(2); % overlap ratio
Nf = win(3); % number of frequency data points

f = linspace(0,f_ny,Nf)'; % frequency data

N_samp = round(Nt/N_win); % number of samples in each window
N_over = round(D_over*N_samp); % number of overlap samples

G = zeros(Nf,N^2); % PSD matrix
for i = 1:N
    for j = 1:N
        G(:,N*(i-1)+j) = cpsd(u(:,i),u(:,j),N_samp,N_over,2*Nf-1)*10/(2*pi*f_ny);
    end
end

```

```

end
end

```

A.1.9 Function Fourier2PSDmatrix.m

```

function G = Fourier2PSDmatrix(U,T)
% G = Fourier2PSDmatrix(U,T)
%
% PURPOSE: Estimates the power spectral density (PSD) matrix G of the Fourier
%           signals U. For example in case of 3 signals we receive 3x3
%           PSD matrix:
%
%           [U_1      [G_11 G_12 G_13
%           U_2  ---->  G =  G_21 G_22 G_23
%           U_3]      G_31 G_32 G_33]
%
% INPUTS: - U = [U_1 U_2 ... U_n]
%           Fourier signals stored column by column. Make sure that the Fourier
%           signals are scaled in correct way.
%
%           - T (scalar)
%           Total time elapsed (time length of the signals u)
%
% OUTPUTS - G = [G_11 G_12 ... G_1n G_21 G_22 ... G_2n ... G_3n ... G_nn]
%           Power spectral density matrix. All components are saved column
%           by column. For example a 3x3 PSD matrix gives
%           G = [G_11 G_12 G_13 G_21 G_22 G_23 G_31 G_32 G_33].
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-06-12
%

Nf = size(U,1); % number of data points
N = size(U,2); % number of signals

G = zeros(Nf,N^2); % PSD matrix
for i = 1:N
    for j = 1:N
        G(:,N*(i-1)+j) = U(:,i).*U(:,j)/T;
    end
end

```

A.1.10 Function transfer_PSD.m

```

function [S,f_out] = transfer_PSD(H,f_trans,G,f_in)
% [X,f_out] = transfer_Fourier(H,f_trans,U,f_in)
%
% PURPOSE: Calculates the output PSD matrix S when the input PSD matrix G and
%           the transfer function matrix H is known.
%
%           S_ij = H_ik*conj(H_jl)*G_kl
%
%           Note however that any number of input and output signals are
%           valid (the transfer function matrix does not have to be a
%           square matrix). Also, note that the number of components in H
%           divided by the number of input signals should be equal to the number
%           of output signals. If H is defined in the wrong way, it could lead to
%           severe errors. All data points outside the defined stress interval
%           of the transfer functions are not being considered.
%
% INPUTS: - H = [H_11 H_12 ... H_1n H_21 H_22 ... H_2n ... H_3n ... H_mn]
%           Transfer function matrix where all components are saved
%           column wise. For example 3x2 gives H = [H_11 H_12 H_21 H_22 H_31 H_32].
%
%           - f_trans
%           Frequency data corresponding to the transfer function. Saved in

```



```

%           a column vector.
%
%           - G = [G_11 G_12 ... G_1n G_21 G_22 ... G_2n ... G_3n ... G_nn]
%               Input signal where each component is saved column wise. Needs
%               to be a square matrix with n^2 components.
%
%           - f_in
%               Frequency data corresponding to the input signals.
%
% OUTPUTS: - S = [S_11 S_12 ... S_1m S_21 S_22 ... S_2m ... S_3m ... S_mm]
%               Output signal where each component is saved column wise. Needs
%               to be a square matrix with m^2 components.
%
%           - f_out
%               Frequency data corresponding to the input signals.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-06-12
%

N_in = sqrt(size(G,2)); % number of input signals
N_out = size(H,2)/N_in; % number of output signals

f_out = f_in(f_trans(1) <= f_in & f_in <= f_trans(end)); % output frequency data
S = zeros(length(f_out),N_out^2); % output signals

G = G(f_trans(1) <= f_in & f_in <= f_trans(end),:); % reduced input signals
H = interp1(f_trans,H,f_out); % interpolated transfer functions

for i = 1:N_out
    for j = 1:N_out
        for k = 1:N_in
            for l = 1:N_in
                S(:,N_out*(i-1)+j) = S(:,N_out*(i-1)+j)+...
                    H(:,N_in*(i-1)+k).*conj(H(:,N_in*(j-1)+1)).*G(:,N_in*(k-1)+1);
            end
        end
    end
end
end
end

```

A.1.11 Function transfer_Fourier.m

```

function [X,f_out] = transfer_Fourier(H,f_trans,U,f_in)
% [X,f_out] = transfer_Fourier(H,f_trans,U,f_in)
%
% PURPOSE: Calculates the output signals X when the input signals U and the
%           transfer function matrix H are known;  $X = H \cdot U$ . For example
%           in case of 3 inputs and 3 outputs we have that:
%
%           [X_1      [H_11 H_12 H_13      [U_1
%            X_2      H_21 H_22 H_23      *   U_2
%            X_3]      H_31 H_32 H_33]      U_3]
%
%           Note however that any number of input and output signals are
%           valid (the transfer function matrix does not have to be a
%           square matrix). Note however that the number of input signals
%           must be equal to the number of columns in the transfer function
%           matrix. Also, note that the number of components in H divided by the
%           number of input signals should be equal to the number of output
%           signals. If H is defined in the wrong way, it could lead to
%           severe errors. All data points outside the defined stress interval
%           of the transfer functions are not being considered.
%
%
% INPUTS: - H = [H_11 H_12 ... H_1n H_21 H_22 ... H_2n ... H_3n ... H_mn]
%           Transfer function matrix where all components are saved
%           column wise. For example 3x2 gives H = [H_11 H_12 H_21 H_22 H_31 H_32].
%
%           - f_trans

```

```

%           Frequency data corresponding to the transfer function. Saved in
%           a column vector.
%
%           - U = [U_1 U_2 ... U_n]
%           Input signal where each component is saved column wise.
%
%           - f_in
%           Frequency data corresponding to the input signals.
%
% OUTPUTS: - X = [X_1 X_2 ... X_m]
%           Output signal where each component is saved column wise.
%
%           - f_out
%           Frequency data corresponding to the input signals.
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-06-12
%

N_in = size(U,2); % number of input signals
N_out = size(H,2)/N_in; % number of output signals

f_out = f_in(f_trans(1) <= f_in & f_in <= f_trans(end)); % output frequency data
X = zeros(length(f_out),N_out); % output signals

U = U(f_trans(1) <= f_in & f_in <= f_trans(end),:); % reduced input signals
H = interp1(f_trans,H,f_out); % interpolated transfer functions

for i = 1:N_out
    for j = 1:N_in
        X(:,i) = H(:,N_out*(i-1)+j).*U(:,j);
    end
end

```

A.1.12 Function file_scanner.m

```

function M = file_scanner(file_path,line_interval,filter_sequence)
% M = file_scanner(file_path,line_interval,filter_sequence)
%
% PURPOSE:
% Read data (float numbers) in a text file (support almost all text files)
% and store it in matrix M. The function is limited to data that is
% following a sequence pattern. The function is very sensitive. If the
% filter sequence is defined in the wrong way, or if the data in the text
% file is not following a sequence pattern, the scanner will not work!
%
% INPUT:
%
% - file_path
%   Path of the file (string).
%
% - line_interval = [min_line max_line]
%   Lines in the file that are to be read, where min_line is the minimum line
%   number and max_line is the maximum line number. If max_line = inf, the
%   function will scan to the very end of the file (if possible).
%
% - filter_sequence
%   Cell containing the numbering order for all float number that you want to
%   save in each row of the sequence. If you want to ignore a row in the
%   sequence, type 0. Note! The function can only save float numbers!
%
% OUTPUT:
%
% - M
%   Matrix containing the float data in the given file. The data is saved
%   column by column.
%
% Example file:
% -----
% 1      ***** Transfer funtion data *****

```

```

% 2      frequency  real/imag  transfer function
% 3          0      real      15e-8
% 4
% 5          imag      3.85e-6
% 6      5.78      real      200
% 7
% 8          imag      -87
% 9          ***** End *****
% -----
% This file contain 9 lines. We are interested to store the frequency
% values and the real and imaginary values of the transfer function in
% three separate columns in M. The first two lines and the last line in the
% text file are not of interested, only the lines in the interval
% line_interval = [3 8] will be accounted for.
%
% The function is reading the file line by line. Each sequence contains 3
% rows, and the middle row is not of interest. The second and fourth term
% in the first row as well as the third term in the third row are of interest.
% The filter_sequence should then be stated as filter_sequence = {[2 4],0,3}.
% The output then becomes M = [ 0      15e-8  3.85e-6
%                               5.78    200    -87 ]
%
% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-04-06
%
fid = fopen(file_path,'r');

if line_interval(2) == inf
    str = '';
    counter = -1;
    while ischar(str)
        str = fgetl(fid);
        counter = counter+1;
    end
    fclose(fid);
    m_inf = counter;
    fid = fopen(file_path,'r');
end

counter = 1;
str = '';
while counter < line_interval(1) && ischar(str)
    str = fgetl(fid);
    counter = counter+1;
end

m = (line_interval(2)-line_interval(1)+1)/length(filter_sequence);

n = 0;
for i = 1:length(filter_sequence)
    filter_sequence{i} = sort(filter_sequence{i});
    if filter_sequence{i} ~= 0
        n = n + length(filter_sequence{i})(filter_sequence{i} >= 0));
    end
end

if line_interval(2) == inf
    M = zeros(m_inf,n);
else
    M = zeros(m,n);
end

row = 1;
while row <= m
    counter1 = 1;
    for i = 1:length(filter_sequence)
        if filter_sequence{i} == 0
            fgetl(fid);
        else
            str = fgetl(fid);
            if str == -1

```

```

        fclose(fid);
        return
    end
    format = '';
    counter2 = 0;
    k = 1;
    for j = 1:filter.sequence{i}(end)
        if filter.sequence{i}(counter2+1) == j
            format(k:k+1) = '%f';
            k = k+2;
            counter2 = counter2+1;
        else
            format(k:k+2) = '%*s';
            k = k+3;
        end
    end
    row_cell = textscan(str,format);
    for p = 1:counter2
        if isempty(row_cell{p})
            fclose(fid);
            return
        end
        M(row,counter1+p-1) = row_cell{p}(1);
    end
    counter1 = counter1 + counter2;
end
end
row = row + 1;
end
fclose(fid);

```

A.1.13 Main code example main_example.m

```

% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last modified: 2019-06-14
%% Material data:
sigma_u = 621; % ultimate strength (MPa)
m_e = 0.5; %
m_t = 0.8; %
m_d = 0.9; %
m_s = 0.7; %
m_o = 1; %
m_prime = 0.9; % reduction factors
Nf_prime = 1e3; % lower cycle limit
Ne = 1e6; % upper cycle limit (fatigue limit)

m = m_e*m_t*m_d*m_s*m_o; % total reduction factor at fatigue limit
sr_prime = 2*m_prime*sigma_u; % upper stress range limit
sr_e = 2*m*sigma_u; % lower stress range limit (fatigue limit)

% S-N parameters Nf = C*sr^(-k):
C = (sr_prime*log(Ne)/sr_e*log(Nf_prime))^(1/log(sr_prime/sr_e));
k = log(Ne/Nf_prime)/log(sr_prime/sr_e);

c_dv = 1/3; % Dang Van parameter

% set stress range limit (does not have to be sr_e and sr_prime):
sr_min = 0;
sr_max = 800;

SN_data1 = [sr_min,sr_max,C,k]; % save uniaxial S-N data
SN_data2 = [sr_min,sr_max,C,k,c_dv]; % save multiaxial S-N data
nbins = 150; % number of bins in histograms

% selection of stress components in this study:
s.select = [true true false true false false]; % output stress components s11, s22, s12

%% Frequency Domain, Spectral approach:
% Load transfer functions H, input signals u and time length T1.

```

```

% How to import the data is up to the reader, but make sure that
% everything is saved column wise. Check the function fatigue_EVMS_Dirlik
% to see how the data should be saved!

% For example, load data from Nastran using the f06 file. The author to this
% code created a file scanner function file_scanner that can scan through
% any text file. However, you need to know what lines that are to be scanned.
% (See file_scanner function for more information).

% - Example 1:
% If the input time signals u and the transfer functions H are known,
% calculate the stress PSD matrix as follows:
win = [30,0.5,round(size(u,1)/10)]; % welch window data (problem dependent)
[G,f] = time2PSDmatrix(u,Tl,win); % input PSD matrix
[S,f] = transfer_PSD(H,f_trans,G,f); % output PSD matrix

% - Example 2:
% If the Fourier transform of the inputs U and the transfer functions H
% are known, calculate the stress PSD matrix as follows:
[X,f] = transfer_Fourier(H,f_trans,U,f)
% S = Fourier2PSDmatrix(X,T)

% - Example 3:
% If the Fourier transform of the stress components X are known (complex
% stress response in frequency domain), calculate the stress PSD matrix as
% follows:
% S = Fourier2PSDmatrix(X,T)

% Dirlik's formula and EVMS (calculates fatigue life):
[T.life1,T.life_red1,corr_mat1,alpha] = fatigue_EVMS_Dirlik(S,f,s_select,Tl,SN_data1,nbins);

D1 = Tl./T.life1; %
D.red1 = Tl./T.life_red1; % if you want the fatigue damage

%% Time Domain, rainflow count:

% Load stress time signals s and the time length T2.
% How to import the data is up to the reader, but make sure that
% everything is saved column wise. Check the function fatigue_WangBrown_DangVan
% to see how the data should be saved!

% For example, load data from Nastran using the f06 file. The author to this
% code created a file scanner function file_scanner that can scan through
% any text file. However, you need to know what lines that are to be scanned.
% (See file_scanner function for more information).

% Wang and Brown's method and Dang Van equivalent stress (calculates fatigue life):
[T.life2,T.life_red2,corr_mat2] = fatigue_WangBrown_DangVan(s,s_select,T2,SN_data2,nbins);

D2 = T2./T.life2; %
D.red2 = T2./T.life_red2; % if you want the fatigue damage

```

A.1.14 Generate white noise, example generate_white_noise.m

```

% Author: Albin Backstrand, albinbackstrand@gmail.com
% Last Modified: 2019-04-10
%% Generate white noise (acceleration in mm/s^2)
N.in = 3; % number of input signals
mu = 0; % mean value (mm/s^2)
dev = 20*1e3; % standard deviation (mm/s^2)
f.ny = 100; % Nyquist frequency (largest observable frequency)
N = 2e5; % number of time steps

f.s = 2*f.ny; % sampling frequency
dt = 1/f.s; % time step
T = N*dt; % length of time signal

% Generate white noise acceleration:
t = linspace(0,T,N+1)';

```

```

a = zeros(N+1,N_in);
for i = 1:N_in
    a(:,i) = random('Normal',mu,dev,N+1,1);
end

% Save data:
data = [t,a];
str = ['input.signals/acc-',num2str(f_ny),'_',num2str(dev),'_',num2str(dt),'_',num2str(N),'_txt'];
save(str,'data','-ascii')

plot(t,a)

```

A.2 NASTRAN Code

MSC NASTRAN version 2018.2.1 was used for this purpose. A standard script was implemented with the help of NASTRAN *Quick Reference Guide* [20].

A.2.1 Frequency Response

Following script generates transfer functions for each stress component in selected hotspot elements 2562, 2873 and 3127. Sinusoidal base excitation in x, y and z directions are solved separately in 3 subcases.

```

SOL 111 $ Solution number (111 - modal frequency analysis)
TIME 3.15e+07 $ Maximum allowable execution time (seconds)
CEND $ End of the executive control section

MAXLINES = 1800000 $ Maximum amount of lines in the output files
LINE = 160000 $ Print lines at each page

TITLE = FREQUENCY RESPONSE

SPC = 1 $ Single point constrained (SID = 1)
SDAMPING = 66 $ Damping property of the structure (SID = 66)
DLOAD = 101 $ Dynamic loading condition - frequency or transient response (SID = 101)
FREQ = 604 $ Frequency data options (SID = 604)

SET 15 = 2562, 2873, 3127 $ Element set saved in SID = 15 (element number 2562, 2873 and 3127)
SET 16 = 6 $ Node set saved in SID = 16 (node number 6, the interface)

STRESS(SORT2, PSDF) = 15 $ Stress output at SET 15

METHOD = 50

SUBCASE 1
TITLE = Acceleration in x-direction
SPC = 1
DLOAD = 101

SUBCASE 2
TITLE = Acceleration in y-direction
SPC = 1
DLOAD = 102

SUBCASE 3
TITLE = Acceleration in z-direction
SPC = 1
DLOAD = 103

BEGIN BULK $ Begin bulk section

ECHOON $ Activate printed echo

TABDMP1, 66, CRIT, , , , , , + $ Damping parameter vs freq table (extrapolation)
+, 1.0, 0.03, 2.0, 0.03, ENDT $ Constant modal damping parameter, 3 percent

$ Parameters:
PARAM, WTMASS, 1.0E-3
PARAM, AUTOSPC, YES

```

```

PARAM, EPZERO, 10.0E-6
PARAM, GRDPNT, 0
PARAM, POST, -2
PARAM, PRTMAXIM, YES

EIGRL, 50, -0.1, 500.0, 25 $ Compute the first 25 eigenmodes (below 500 Hz)

SPC1, 1, 123456, 6 $ Constrain node number 6 (all DOF constrained; 123456)

$ Enforced sinusiodal acceleration in x-direction (unit 1.0 mm/s^2), table 51:
RLOAD1, 101, 11, , , 51, ACCE
SPCD, 11, 6, 1, 1.0

$ Enforced sinusiodal acceleration in y-direction (unit 1.0 mm/s^2), table 51:
RLOAD1, 102, 12, , , 51, ACCE
SPCD, 12, 6, 2, 1.0

$ Enforced sinusiodal acceleration in z-direction (unit 1.0 mm/s^2), table 51:
RLOAD1, 103, 13, , , 51, ACCE
SPCD, 13, 6, 3, 1.0

TABLED1, 51, , , , , , , + , $ Amplitude vs frequency (extrapolation)
+, 0.0, 1.0, 50.0, 1.0, ENDT $ Constant amplitude of 1.0 mm/s^2

$ Select more output frequencies - equally spaced based on log scale.
FREQ2, 604, 0.1, 2000.0, 1000

ECHO00FF $ Deactivate printed echo

INCLUDE 'airdryer.dat' $ Mesh data and material parameters of the airdryer bracket

ENDDATA $ End of document

```

A.2.2 Transient Response

Following script generates first 25 eigenfrequencies of the air dryer bracket and also stress time signals for each of the selected hotspot element 2562, 2873 and 3127. Base excitation in x, y, and z directions are solved separately in 3 subcases. The 4:th subcase is requested to calculate stress signals when all the loads are applied simultaneously.

```

SOL 112 $ Solution number (112 - modal transient analysis)
TIME 3.15e+07 $ Maximum allowable execution time
CEND $ End of the executive control section

MAXLINES = 80000000 $ Maximum amount of lines in the output files
LINE = 2000000 $ Print lines at each page

TITLE = TRANSIENT RESPONSE ANALYSIS

ECHO = PUNCH $ Controls the echo (printout). Add a punch file to the output files

SPC = 1 $ Single point constrained (SID = 1)
SDAMPING = 66 $ Damping property of the structure (SID = 66)
DLOAD = 101 $ Dynamic loading condition - frequency or transient response (SID = 101)
TSTEP = 20 $ Transient time data options (SID = 20)

SET 15 = 2562, 2873, 3127 $ Elemet set saved in SID = 15 (element number 2562, 2873 and 3127)
SET 16 = 6 $ Node set saved in SID = 16 (node number 6, interface)

STRESS(SORT2, PRINT) = 15 $ Stress output at SET 15

METHOD = 50

SUBCASE 1
TITLE = Enforced acceleration in x-direction
SPC = 1
DLOAD = 101
TSTEP = 20

```

```

SUBCASE 2
TITLE = Enforced acceleration in y-direction
SPC = 1
DLOAD = 102
TSTEP = 20

SUBCASE 3
TITLE = Enforced acceleration in z-direction
SPC = 1
DLOAD = 103
TSTEP = 20

SUBCASE 4
TITLE = Enforced acceleration in all directions
SPC = 1
DLOAD = 104
TSTEP = 20

BEGIN BULK $ Begin bulk section

ECHOON $ Activate printed echo
EIGRL, 50, -0.1, 500.0, 25 $ Compute the first 25 eigenmodes (below 500 Hz)

TABDMP1, 66, CRIT, , , , , + $ Damping parameter vs freq table (extrapolation)
+, 1.0, 0.03, 2.0, 0.03, ENDT $ Constant damping parameter 3%

$ Parameters:
PARAM, WTMASS, 1.0E-3
PARAM, AUTOSPC, YES
PARAM, EPZERO, 10.0E-6
PARAM, GRDPNT, 0
PARAM, POST, -2
PARAM, PRTMAXIM, YES

SPC1, 1, 123456, 6 $ Constrain node number 6 (all DOF constained; 123456)

$ Enforced acceleration in x-direction of node number 6 (unit 1.0 mm/s^2), table 51
TLOAD1, 101, 11, , 3, 51
SPCD, 11, 6, 1, 1.0

$ Enforced acceleration in y-direction of node number 6 (unit 1.0 mm/s^2), table 52
TLOAD1, 102, 12, , 3, 52
SPCD, 12, 6, 2, 1.0

$ Enforced acceleration in z-direction of node number 6 (unit 1.0 mm/s^2), table 53
TLOAD1, 103, 13, , 3, 53
SPCD, 13, 6, 3, 1.0

$ Enforced acceleration in all directions of node number 6 (unit 1.0 mm/s^2), table 51-53
TLOAD1, 104, 14, , 3, 51
SPCD, 14, 6, 1, 1.0
TLOAD1, 104, 15, , 3, 52
SPCD, 15, 6, 2, 1.0
TLOAD1, 104, 16, , 3, 53
SPCD, 16, 6, 3, 1.0

TSTEP, 20, 2000000, 0.0005, 10 $ Transient time data

ECHOOFF $ Deactivate printed echo

INCLUDE 'trlm_accX_100_20000_0.005_200000.dat' $ Table 51 data (MATLAB data)

INCLUDE 'trlm_accY_100_20000_0.005_200000.dat' $ Table 52 data (MATLAB data)

INCLUDE 'trlm_accZ_100_20000_0.005_200000.dat' $ Table 53 data (MATLAB data)

INCLUDE 'airdryer.dat' $ Mesh data and material parameters of the airdryer bracket

ENDDATA $ End of document

```


References

- [1] M. Antonio and J. Castro. An improved multiaxial rainflow algorithm for non-proportional stress or strain histories - Part II: The modified Wang-Brown method. *International Journal of Fatigue* (2011).
- [2] D. Benasciutti. Some analytical expressions to measure the accuracy of the "equivalent von Mises stress" in vibration multiaxial fatigue. *Journal of Sound and Vibration* **333** (2014).
- [3] D. Benasciutti, F. Sherratt, and A. Cristofori. Recent developments in frequency domain multi-axial fatigue analysis. *International Journal of Fatigue* **91** (2016), 397–413.
- [4] V. Berbyuk. *Structural Dynamics Control*. 2nd ed. Chalmers University of Technology, 2014.
- [5] M. Bonte et al. Determining the von Mises stress power spectral density for frequency domain fatigue analysis including out-of-phase stress components. *JOURNAL OF SOUND AND VIBRATION* **302** (2007), 379–386. DOI: 10.1016/j.jsv.2006.11.025.
- [6] C. Braccesi, F. Cianetti, and L. Tomassini. An innovative modal approach for frequency domain stress recovery and fatigue damage evaluation. *International Journal of Fatigue* **91.2** (2016), 382–396. DOI: <https://doi.org/10.1016/j.ijfatigue.2016.02.028>.
- [7] R. R. Craig Jr. and A. J. Kurdila. *Fundamentals of Structural Dynamics*. 2nd ed. John Wiley Sons, Inc., 2006. ISBN: 9780471430445.
- [8] T. Dirlik. Application of computers in fatigue analysis (1985). URL: <http://wrap.warwick.ac.uk/2949/>.
- [9] N. E. Dowling. *Mechanical behaviour of Materials*. Fourth International Edition. Pearson Education Limited, 2013. ISBN: 0-273-76455-1.
- [10] S. D. Downing and D. F. Socie. Simple Rainflow Counting Algorithms. *International Journal of Fatigue* **4**.Issue 1 (1982), 31–40. DOI: 10.1016/0142-1123(82)90018-4.
- [11] A. Ekberg. *Multiaxial Fatigue*. 5.3. Chalmers University of Technology, 2018.
- [12] G. B. Folland. *Fourier Analysis and Its Applications*. American Mathematical Society, 2009. ISBN: 978-0-8218-4790-9.
- [13] A. Halfpenny. A frequency domain approach for fatigue life estimation from Finite Element Analysis. *International Conference on Damage Assessment of Structure (DAMAS 99)* (n.d.).
- [14] A. Halfpenny. What is the 'Frequency Domain' and how do we use a PSD? (n.d.).
- [15] T. Irvine. Estimating Fatigue Damage from Stress Power Spectral Density Functions (2014).
- [16] R. C. Juvinall and K. M. Marshek. *Fundamentals of Machine Component Design*. Sixth edition. John Wiley Sons, Inc., 2017. ISBN: 978-1-118-98768-1(PBK).
- [17] N. Kikuchi and J. Oden. *Contact problems in elasticity. A study of variational inequalities and Finite Element Methods*. Society for Industrial Mathematics, 1995. ISBN: 9780898714685.
- [18] P. M. Kurowski. *Finite Element Analysis for Design Engineers*. SAE International, 2004. ISBN: 0-7680-1140-X.
- [19] L. D. Lutes and S. Sarkani. *Random Vibrations-Analysis of Structural and Mechanical Systems*. Elsevier Ltd., 2004. ISBN: 0-7506-7765-1.
- [20] *MSC Nastran Documentation. Quick Reference Guide*. Version 2018.2. MSC. Software Corporation, 2018.
- [21] D. E. Newland. *An Introduction to Random Vibrations and Spectral Analysis*. Longman Group Limited, 1975. ISBN: 0 582 46335-1.
- [22] A. Nieslony and M. Böhm. Frequency-domain fatigue life estimation with mean stress correction. *International Journal of Fatigue* **91.2** (2016), 373–381. DOI: <https://doi.org/10.1016/j.ijfatigue.2016.02.031>.
- [23] P. Olofsson and M. Andersson. *Probability, Statistics, and Stochastic Processes*. 2nd ed. John Wiley Sons, Inc., 2012. ISBN: 9780470889749.
- [24] N. Ottosen and H. Petersson. *Introduction to the Finite Element Method*. Pearson Prentice Hall, 1992. ISBN: 978-0-13-473877-2.
- [25] X. Pitoiset and A. Preumont. Spectral methods for multiaxial random fatigue analysis of metallic structures. *International Journal of Fatigue* **22** (2000), 541–550.
- [26] J. A. Rice. *Mathematical Statistics and Data Analysis*. 3rd ed. Brooks/Cole, Cengage Learning, 2007. ISBN: 978-0-495-11868-8.
- [27] D. J. Segalman et al. An Efficient Method for Calculating RMS von Mises Stress in a Random Vibration Environment (1998).
- [28] F. Sherratt, N. W. M. Bishop, and T. Dirlik. Predicting Fatigue Life from Frequency Domain Data. *Engineering Integrity* **18** (2005), 12–16.

- [29] E. M. Stein and R. Shakarchi. *Fourier Analysis: An Introduction*. illustrated ed. Princeton University Press, 2003. ISBN: 9780691113845.
- [30] P. Stoica and R. Moses. *Spectral Analysis of Signals*. Prentice Hall, Inc., 2005. ISBN: 0-13-113956-8.
- [31] Understanding FFTs and Windowing (2019, revised). URL: <http://www.ni.com/en-us/innovations/white-papers/06/understanding-ffts-and-windowing.html>.
- [32] C. H. Wang and M. W. Brown. Life Prediction Techniques for Variable Amplitude Multiaxial Fatigue-Part 1: Theories. *Journal of Engineering Materials and Technology* **118** (1996), 367–370.