Department of Precision and Microsystems Engineering

Prediction of lifetime and failure mechanisms of rolling contacts in DOT Direct Drive Pump

Jeroen Houwen

Report no : MSD-2021.058
Coach : Ron van Ostayen
Professor : Ron van Ostayen, Gijs Druijf (DOT)
Specialisation : MSD
Type of report : Master Thesis
Date : 09/09/2021

Prediction of lifetime and failure mechanisms of rolling contacts in DOT Direct Drive Pump

Master Thesis

by

J. Houwen

to obtain the degree of Master of Science

at the Delft University of Technology,

to be defended publicly on Thursday September 9, 2021 at 14:00.

Student number:	4694015	
Project duration:	August 1, 2020 – September 9, 2021	
Thesis committee:	Dr. Ir. R.A.J. van Ostayen,	TU Delft, supervisor
	Dr. Ir. A. Aragon	TU Delft
Supervisors:	Ing. G. Druijf,	DOT B.V.
Honorable mention:	S. Lorenz,	Purdue University

This thesis is confidential and cannot be made public until September 9, 2023.

TU Delft

Abstract

In recent years, the increased need for renewable energy has put more interest in wind energy and therefore in making wind turbines more efficient and building them as efficiently and cheap as possible. Delft Offshore Turbines (DOT) is developing a drivetrain that replaces the heavy and expensive parts in the nacelle of a wind turbine and makes these parts more accessible and aims to improve the reliability of the wind turbines by replacing the parts that are most prone to failure.

Due to the innovative design of their drivetrain, several unknowns still have to be answered since this has never been tried before. One of the most important parts of this drivetrain requires rolling contacts that endure high contact stresses and have to operate for a long time in harsh conditions. These rolling contacts have to be tested and for this, DOT created the Roller Test Bench (RTB) to test different materials and material combinations until failure and to see what type of failure is occurring and what the lifetime of these rollers is. The rollers are a lot smaller compared to the rollers found in the drivetrain to reduce cost and testing time. The scaling of the dimensions has been done several times in different fields of research but it is unclear how this affects the results and how the results translate to the original rollers. In literature some papers have stated that the results scale in a certain way but this has not been proven or the statements have not been argumented in a clear way such that these statements hold no ground.

A rolling contact has several ways that it can fail; wear, deformation, corrosion or fatigue are most commonly found. From the literature study performed in this thesis it has been noted that the main mode of failure in the RTB is rolling contact fatigue since the rollers are smooth, have no lubrication applied, are free of external contaminants and have no significant amount of friction on the surface. With the stress applied and the material used it is also noted that the fatigue will occur after a high amount of cycles and in literature this is also called high-cycle rolling contact fatigue. Experimental testing of such problems takes a very long time, even if the rollers are smaller and are run faster through the load cycles.

With both problems stated, the lack of information on the influence of scaling on fatigue life and the testing of high cycle fatigue, the goal of this thesis was found and it was to find a method to predict the lifetime and wear mechanisms of rolling contacts in the DOT Direct Drive Pump. This method could then be used to figure out how the scaling affects the fatigue life for rolling contacts and make a possible statement of its influence.

From the literature study a numerical modelling method was found that incorporated the microstructure and included both crack initiation and propagation and was able to model high cycle fatigue. This model suited the problem the best and was proven to provide results that agree well with previously performed experiments and their experimental results. This method implemented a Voronoi tessellation to reproduce the microstructure and applied continuum damage mechanics to calculate the damage evolution and apply the damage to the microstructure. A jump-in-cycle method was applied to improve the efficiency of the model and greatly reduce the time to model until failure. The created model is able to reproduce subsurface initiated fatigue and with that produce a scatter and L_{10} life comparable with previous models and experimental results. The model was validated by comparing the stress distribution, crack formation and the initiation and total lives with other models and data. The model was off the required range since the models uses an homogeneous isotropic material which causes the scatter to be lower.

With the model created and validated, it was applied to several different half-widths to test how the scaling affects the fatigue life. From the results it was noted that for a larger roller size the fatigue life decreases but it has to be noted that further tests, experimental and numerical, have to be done and improvements to the model have to be made such that at some point a correlation can be made between the scaling and the fatigue life. Further recommendations were provided on improving the model and with that creating more reliable results such that a better prediction can be made on how the fatigue life is affected but also to predict

the life of a rolling contact.

Acknowledgements

In this thesis research on rolling contacts and more specifically rolling contact fatigue is performed for the DOT Direct Drive Pump together with Delft Offshore Turbines (DOT) and the TU Delft. The project was conducted as part of the curriculum requirements for obtaining a master degree in High Tech Engineering at the faculty of Mechanical Engineering of the TU Delft. The project was initiated with an internship at DOT where I designed a smaller test bench for testing rolling contacts and studied the subject and its shortcomings in literature. DOT and the TU Delft have been working together on several projects and I would like to thank both parties for giving me the opportunity to learn about the subject and perform this research.

At DOT I would like to thank Gijs Druijf for all the time and support he has given me over the past year. He taught me a different way of thinking and working and he was always willing to help and give his opinion. His critical view and experience was extremely valuable to the project.
At the TU Delft I would like to thank Ron van Ostayen who supported and guided me the whole year. His experience and knowledge helped me a lot and helped me steer in the right direction to reach this result. His opinion and different view whenever I was overthinking a specific subject helped me a lot and working with Ron was a pleasure. I would also like to thank Jos van Driel who at the start of my project, when an experimental setup was still part of the project, was helpful and even surprised me when he had already worked out several things without me knowing. His input was incredible and a pleasure to work with Jos.

I would also like to show my gratitude towards Steven Lorenz who is a research assistant at the Mechanical Engineering Tribology Laboratory of Purdue University. Steven helped me in building the model and gave me advise and his opinion on certain choices while building the model. Steven helped me without hesitation and had all the patience in the world to answer my (sometimes not so smart) questions. I cannot explain how grateful I am for his help. With that I also would like to thank professor Sadeghi who linked me with Steven and helped me answer some doubts I had about different modelling methods. I also would like to thank professor Anders Ekberg of Chalmers university for providing me with his knowledge about the models he created. Many others have had a hand in this project and I am grateful for all the help.

Finally I would like to thank Jolien for the support, the love, easing my mind in stressful times and for believing in me. I would like to thank my roommates and friends for supporting me and being such amazing people. Last but not least I would like to thank my parents. They have supported me emotionally and financially during my studies and they helped me become who I am and where I am today. They supported and even encouraged me to do the things that I dreamed of and were always there when I needed it the most. Because of my parents, you are all able to read this work and I cannot express in words what all their support means to me. Dank jullie wel!

Contents

1 Introduction

In this section, some background will be given on the origin of this project and on the parties involved. Later on, the aim of the project will be explained and the lay-out of the thesis will be outlined.

1.1. Delft Offshore Turbines - DOT

DOT is a start-up company based in Delft which is part of De Oude Bibliotheek (DOB) and both have a focus on the offshore energy sector. DOB is a knowledge institute which offers training courses and seminars to professionals who are interested in offshore related subjects. DOT is the innovator and focusses on offshore wind turbines and mainly on seawater-hydraulic power transmission.

In the past couple years, the need for renewable energy has increased [38] and therefore the size of the wind turbines has been increasing, especially offshore where the wind speed tend to be faster and steadier and because they do not take up building or farming space [23]. A wind turbine is made out of a rotor, a generator with sometimes a gearbox and the surrounding structure, the housing on top of a wind turbine which encapsulates the generator and gearbox is the nacelle. Most wind turbines have a gearbox that turns the slow rotation of the blades to a higher speed suitable for the generator but some have a special generator that is able to generate electricity with the lower rotational speed of the blades. The increase in size of wind turbines means that the rotational speed at the shaft is even lower but the torque is higher and that leads to increased stresses on the gearbox and the other components in a conventional wind turbine. This also leads to an increase in size and mass of the components in the nacelle and with that more supporting structure for the wind turbine is required but also increasing the price of these components and the wind turbine.Knowing this it is logical to assume that reducing the mass in the nacelle could mean a reduction in the price for production and placement of these wind turbines.

To solve this problem, DOT is developing a drive train that is able to work with high torque and low RPM and is able to deliver high amounts of power but with a lower mass. The main parts of this system are a direct drive pump and a Pelton turbine which drives the generator to create electricity. The direct drive pump, as the name states, is directly driven by the turbine rotor which in turn rules out the gearbox which is prone to failure. This pumps sea water at a high pressure to the Pelton turbine which in turn drive the generator. The Pelton turbine and the generator are placed in the middle of a group of wind turbines and is connected to these wind turbines as seen in Fig. 1.1. The centralisation of the generator also reduces the cost of the wind turbines and makes it easier to assemble the turbines since the generator is expensive and is one of the heavier parts of the turbine. Placing it lower also makes it easier to maintain and not as many have to be maintained since one generator is used for multiple turbines.

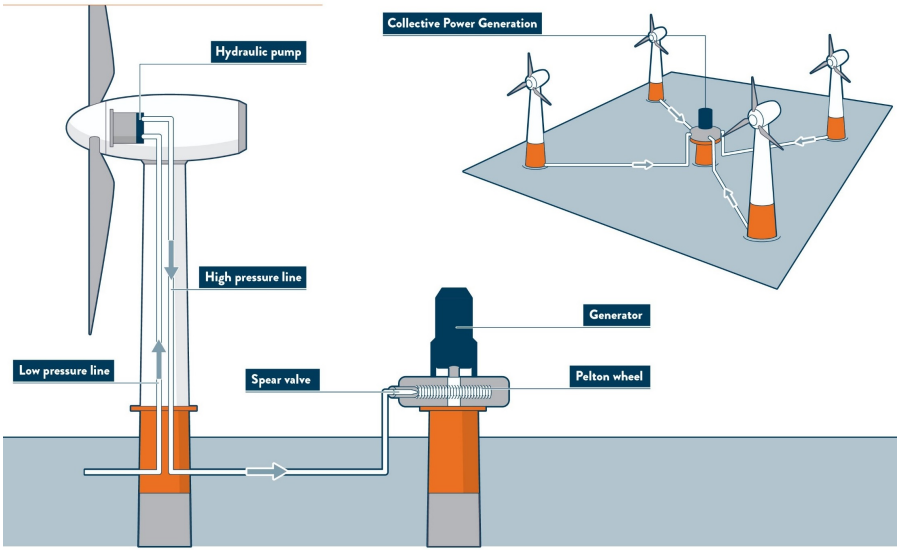


Figure 1.1: *Schematic of the innovative concept from DOT [42]*

The pump they are developing is a radial piston pump with the outer ring, which is connected to the turbine rotor, consisting of lobes and the inner part consisting of several cylindrical roller and piston combinations. The roller rolls over the surface of the outer ring and the lobes on this outer ring act the same as the camshaft in an engine, making the roller and piston move up and down creating a pumping movement. While there are already pumps that can deliver several megawatts, these are oil-based systems and due to cost and ecological reasons these pumps are not desirable and therefore a sea-water drive pump has to be developed. For an oil-based system, the whole system needs to be closed off and the oil has to be transported to the wind turbine which increases the cost while for a seawater system, it is already there and if something breaks on the system it would not spill any oil into the sea and just seawater.

The Modular project was the first project that developed a working pump and was able to put out 40kW but the market requires a higher output (3 MW and more) and therefore the project DOT 3000 was started which is focussed on developing a 3MW direct drive pump.

1.2. DOT 3000

DOT 3000 is the follow up project of the Modular project which focusses on creating a larger pump with a higher power output. In this project all the components are larger but the stresses are also higher compared to the Modular project. Since the stresses on all the components are higher, a more in depth look at some of these parts has to be taken since the risk of failure increases. Because the cylindrical rollers in the pump, rolling over the surface of the outer ring, causes the pistons to translate and therefore make the pump work, these roller play a crucial role in the pump. For this reason DOT created the RTB (Fig. 1.2) where they test the rollers at a smaller size compared to the Direct Drive Pump (DDP). These rollers are tested for material hardening, wear, corrosion and fatigue on the contact area and due to the smaller size of the rollers, the test can be done at a faster rate since the rollers can be run at higher RPM. The reason they are tested for wear, corrosion and fatigue is due to the harsh environment the rollers are ran in. The contact surfaces are not lubricated and will experience corrosion of the sea water and wear due to particles that are in the water. Since the rollers operate in such a tough environment and they are required to run for years on end, it is important to test these rollers and to test for the best material combination for this application.

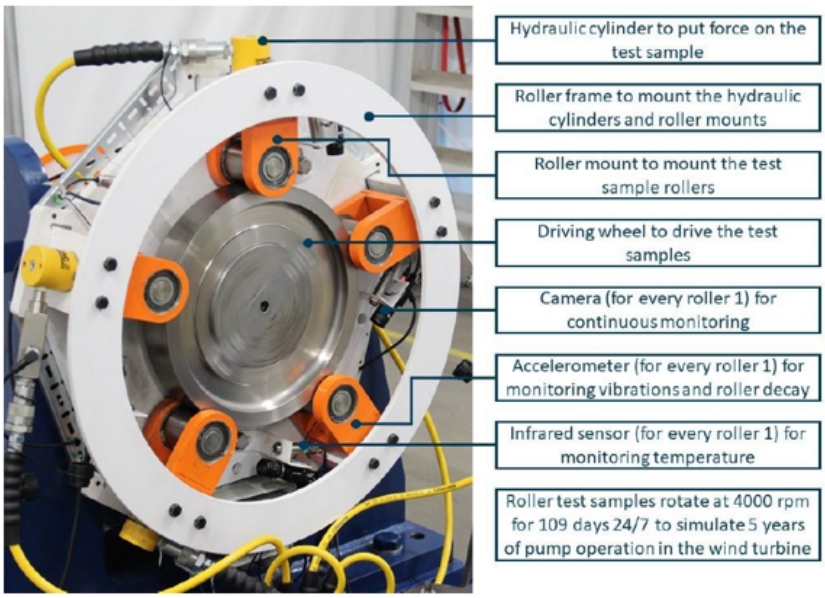


Figure 1.2: *Schematic of the roller test bench created by DOT during the DOT 3000 project*

The RTB is build to do endurance roller test and to simulate five years of continuous loading without maintenance and with this test simulating the behaviour of the rollers on the outer ring in the pump. The idea behind the RTB is to create a relatively small test bench which can be easily used and where the rollers can be quickly changed or where other parameters can be added or removed to see the effect of these parameters. As seen on Fig. 1.2, the setup consists out of five smaller rollers and one large roller which is the driving wheel. The smaller rollers are pressed onto the larger wheel at a specified force such that the contact pressure is equal to the one in the real life pump. The smaller roller surfaces are regularly checked visually for signs of failure or failure initiation and during the testing, the RTB measures the vibrations of the rollers to check if the surface is causing excessive vibrations and therefore signs of failure.

1.3. Problem Definition

Endurance testing of a rolling contact takes a long time and is expensive and even if the rollers are smaller and are run at a faster rate, it still takes a very long time to see any signs of failure. Since the rollers in the RTB are not lubricated and are tested in a clean environment, the rollers will most probably fail due to fatigue and more precisely high cycle rolling contact fatigue. With high cycle fatigue, it is expected that the roller will fail after more than 10^4 load cycles but in the case of the DOT DDP it will probably be around 10^{12} cycles. For this reason, downscaling of the rollers is interesting since the rollers can be tested at a faster rate but also multiple test benches can be made at a lower cost. This is done in the RTB but also in the railway industry where the downscaling of train wheels and rails have been applied extensively to test the dynamics and fatigue behaviour of train wheels and rails. While the scaled benches give a good idea what the behaviour of certain material combinations will be like, it is still unclear how the scaling of the dimensions influences the fatigue life and therefore it is unclear how the results from the RTB translate to reality. Some papers have stated that the reduction in dimensions reduces the fatigue life but no clear relation between the two was given or this was not validated experimentally [20][28][55]. Therefore it is still unclear what happens with the fatigue life when the dimensions are scaled but the contact stress is kept the same.

Another important thing to note when testing these rollers is the variability of the results due to the heterogeneous material structure. Fatigue is the formation of cracks in the material and can originate on the surface or in the material. This process is highly dependent on the material structure since cracks are formed on the weak points in the material like grain boundaries, dislocations and other irregularities. Due to these factors, a scatter can be seen in the fatigue life results of rollers and it is unknown when a roller will fail which is a phenomenon that is also found in bearings. The way bearing manufacturers solve this is by giving bearings an L_{10} life, the amount of cycles until 10% of a population of bearings have failed, which is found by

testing a large set of bearings until failure. Therefore to get an accurate prediction on the fatigue life of these rollers, the rollers have to be tested several times such that a life prediction can be made. As was said earlier, only testing once takes a long time so testing multiple times will become unreasonable if only one test bench is used.

1.4. Aim of the Project

Based on the problems stated in the previous section, the aim of the project is to provide a method to predict the lifetime and failure mechanisms of the rolling contacts in the DOT DDP. Besides providing this method, an overview of the mechanisms behind the failure and behaviour of the rolling contacts is given to better understand the workings of rolling contacts and with the gathered knowledge be able to prevent failure and extend the lifetime of the rolling contacts. The results that are gathered from the provided method are then used to analyse the influence of scaling on the fatigue life of rolling contacts and a recommendation will be made based on the retrieved results.

Research Objective

How to predict lifetime and failure mechanisms of rolling contacts in DOT Direct Drive Pump?

Background questions

Which failure mechanisms are expected to occur in rolling contacts?
How can lifetime be predicted for rolling contacts?
What are the main failure mechanisms of rolling contacts in DOT Direct Drive Pumps and how can the lifetime be predicted for these rolling contacts?

Additional questions

How does scaling influence lifetime and wear mechanisms of rolling contacts?
How to numerically model failure mechanisms and predict the lifetime of rolling contacts?

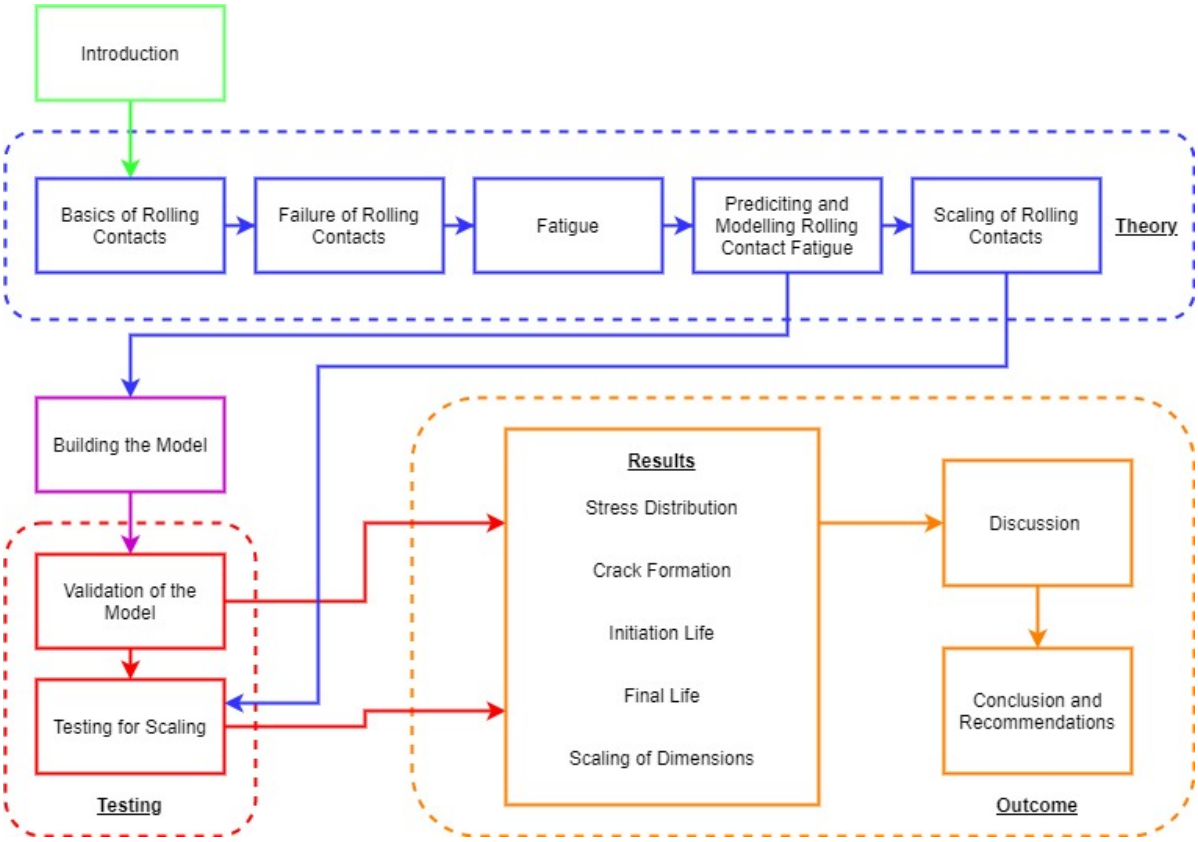
1.5. Outline of the Thesis

This section presents an overview of the structure of this thesis. The report starts by providing background theory on the fundamentals of rolling contacts and the main modes of failure applicable to the DOT Direct Drive Pump. This provides the basis for the following part of the literature study which is focussed on fatigue and crack behaviour since this is assumed to be the main mode of failure in the DOT DDP. Knowing this, the methods to calculate rolling contact fatigue life are presented and a method is chosen that fits the problem the best. These methods range from probabilistic methods to a numerical approach that implements the material structure. The chosen method is discussed in detail and a validation approach is presented. At the end of the literature study, the literature around scaling is presented and discussed and the reason for further research is explained.

After the literature study, the chosen method is worked out in detail such that it is understood properly and can be applied in the used software such that a correct and fitting model is build. The assumptions made to make this model are discussed in the end and after building the model, the validation is done and compared to the existing numerical models and experimental data. Subsequently, the model is used to estimate the fatigue life for several differently sized rolling contacts and the relation between the different sizes is discussed and translated to real life.

To sum this all up, the paper consists out of three parts:

1. **Theory** : Literature study and theoretical background on rolling contacts, rolling contact failure and methods to predict fatigue life
2. **Building the Model** : The application of the method and detailed explanation of the building procedure
3. **Testing** : Validation of the chosen method and the presentation of the testing procedure
4. **Outcome** : The results are presented and a possible correlation is discussed. Further improvements and further work is suggested


Introduction
Basics of Rolling Contacts
Failure of Rolling Contacts
Fatigue
Prediciting and Modelling Rolling Contact Fatigue
Scaling of Rolling Contacts
Theory
Building the Model
Validation of the Model
Testing for Scaling
Testing
Results
Stress Distribution
Crack Formation
Initiation Life
Final Life
Scaling of Dimensions
Discussion
Conclusion and Recommendations
Outcome

2

Background Information

In this chapter, the theory of static and rolling contacts, the failure mechanisms of rolling contacts, modelling and prediction methods and the scaling is discussed based on literature found on the subject. The main focus will be on line contacts and in the latter part of this chapter, the focus will be on high cycle fatigue. Firstly, the basics of contact mechanics and rolling contacts relevant to the subject are discussed. After that, the focus will be on possible modes of failure, followed up by the section on predicting fatigue life. Lastly, the modelling technique applied in this thesis is discussed and at the end, the literature around scaling of rolling contacts is discussed.

2.1. Introduction

The basics of rolling contacts starts with contact mechanics itself which Hertz [21] created the basis off early on [30][61]. From the theory he created, many other researchers followed his theory with probably the most important and well known one being K.L.Johnson [31] who is also the author of the book "Contact Mechanics". With the basis of contact mechanics created, a further focus was put on rolling contacts and since rolling contacts are extensively used, the subject has been researched a lot in many different fields and on many different aspects and is an interesting and widely researched topic. Rolling contacts can be found on many places like bearings, train wheels and even the pen you write with, and therefore it is still such a topic of interest.

Since the 1940s, bearing manufacturers have been increasingly interested in understanding the bearing behaviour and the cause of failure of bearings. With the improvement of lubricants and an increased understanding of material behaviour and alloy composition, the failure of these bearings has gone from mainly wear to mainly fatigue originated. Since fatigue in rolling contacts is different to regular fatigue due to its multiaxial and non-proportional behaviour, new methods had to be developed to test this behaviour but also predict the fatigue life of the rolling contact. The first model created that predicted rolling contact fatigue life was the probabilistic engineering life model formulated by Lundberg and Palmgren [35]. This work has had an influence on almost all the models that followed after that, each one adapting their (Lundberg and Palmgren) method or adding to their method for improvement or adjusting to the fit ones research. Other important models were created by Ioannides and Harris [24] and by Harris and Barnsby [18] and the current ISO standards also use a modified version of the Lundberg-Palmgren model.

Later on, more and more research models were created that applied a deterministic approach meaning that the physical principals were taken into account to predict the failure process of a rolling contact. Due to the complexity of rolling contact fatigue, many models focussed on either crack initiation or propagation and were often based on a homogeneous material description. Some of the most important research was done by Ekberg [10] who implemented the Dang Van model and Jiang and Sehitoglu [29] who applied an elastoplastic finite element model. Since these models lacked the microstructural behaviour due to their assumption of a homogeneous material, it lacked important information and a model was created that implemented the microstructure through the implementation of the Voronoi tessellation. The model and the research on these

models is led by Sadeghi and has been giving promising results regarding predicting fatigue life and simulating real life RCF behaviour in numerical models. A review on all the important models created can be found in the work of Sadeghi et al. [54]. Raje and Sadeghi [50] set the basis of this modelling technique and this was further refined and applied to study different types of behaviour in the following years in which the research was led by Sadeghi.

2.2. Contact Mechanics

It is important to start off with understanding the basics of contact mechanics because the formulas found in this section will be the basis of all the calculations in the following sections. Also having a good understanding of the stress distribution through the material will be important when looking at the possible failure mechanism in rolling contacts.

2.2.1. Hertzian Contact Stress

The Hertzian contact theory is the classical theory of contact mechanics where the focus is on non-adhesive contacts where no tension force is allowed to occur on the contact area. Hertz sets several boundary conditions to determine the solution to Hertzian contact problems.

1. The bodies are in frictionless contact
2. The deformations and localized stresses are within the elastic limit and the stresses must disappear moving away from the contact area i.e. the strains are small
3. The surface is an infinitely large half-space
4. The surfaces are continuous and non-conforming

2.2.2. Contact between Two Cylinders

In this project the focus will be on contact between two cylinders with parallel axes (Fig. 2.1). This is also called a line contact in which the contact area resembles a line if no deformation is assumed. In reality, due to elastic deformation, this is a rectangle with a length L and a half-width b which can be calculated with formula 2.1 [70].

$$b = \sqrt{\frac{4F\left[\frac{1-\upsilon_1^2}{E_1} + \frac{1-\upsilon_2^2}{E_2}\right]}{\pi L\left(\frac{1}{R_1} + \frac{1}{R_2}\right)}} \tag{2.1}$$

where F = Force pressing on the cylinders
υ_1, υ_2 = Poisson's ratio for the materials
E_1, E_2 = Young's modulus of the materials
L = Contact length
R_1, R_2 = Radii of the cylinders

The half-width can then be used to calculate the maximum contact pressure which can be found on the centerline of the contact strip as seen in figure 2.1.

2.2.3. Contact Pressure and Stress Components

Formula 2.2 calculates this maximum pressure from the normal force applied on the cylinders. The formulas 2.1 and 2.2 were both determined by Hertz and the contact stress was therefore called the Hertz contact stress.

$$P_{\max} = \frac{2F}{\pi bL} \tag{2.2}$$

where F = Force pressing on the cylinders
b = Half-width on the contact area
L = Contact length

The contact pressure or Hertzian pressure has a certain distribution along the contact surface which can be seen in Fig. 2.1. The formula can be seen below. This differs between all the different types of contact but this is for line contacts.

$$P(x) = P_{max}\sqrt{1 - \left(\frac{x}{b}\right)^2} \tag{2.3}$$

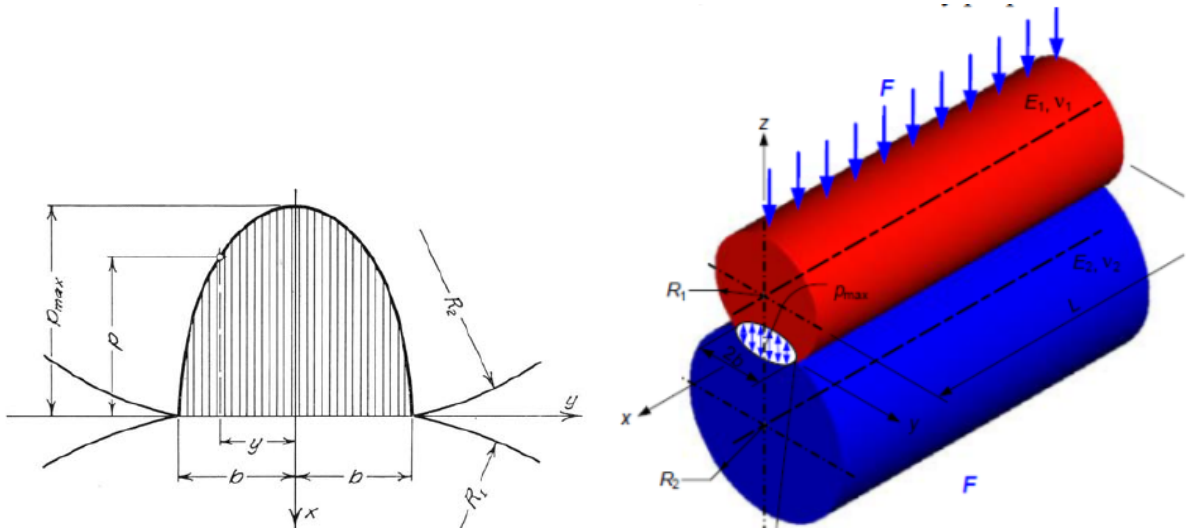


Figure 2.1: *On the left: Pressure Distribution on Contact Surface [49]. On the right: Two Cylinders with Parallel Axes [70]. The x-axis on the left is the same as the z-axis on the right*

Later work on contact mechanics extended Hertz' work and determined the distribution of stress components along the x-axis. Figure 2.2 shows the stress components σ_x, σ_y, σ_z and $\tau_{45°}$ for a line contact. The stresses approach zero with increasing depth and the maximum shear stress can be found at a depth of $x \cong 0.78b$ from the contact surface and the magnitude of the maximum shear stress is $\tau_{max} \cong 0.3P_{max}$ [49] [31] while the maximum normal stresses can be found on the surface. $\tau_{45°}$ is the shear stress component which lies on the bisecting plane between the x-axis and the y-axis. This component can be calculated with the σ_x and σ_y as seen in formula 2.9. The state of stress in a material point, under general loading, is described by the following six components

$$\sigma_{xx}, \sigma_{yy}, \sigma_{zz} \tag{2.4}$$

$$\tau_{xy} = \tau_{yx}, \tau_{yz} = \tau_{zy}, \tau_{xz} = \tau_{zx} \tag{2.5}$$

$$\sigma_1 = \sigma_x = -\frac{P_{\max}}{\sqrt{1+(z/b)^2}} \tag{2.6}$$

$$\sigma_3 = \sigma_y = -P_{\max}\left(\frac{1+2(z/b)^2}{\sqrt{1+(z/b)^2}} - 2|z/b|\right) \tag{2.7}$$

$$\sigma_2 = \sigma_z = -2\upsilon P_{\max}\left(\sqrt{1+(z/b)^2} - |z/b|\right) \tag{2.8}$$

$$\tau_{45°} = \frac{|\sigma_x - \sigma_y|}{2} \tag{2.9}$$

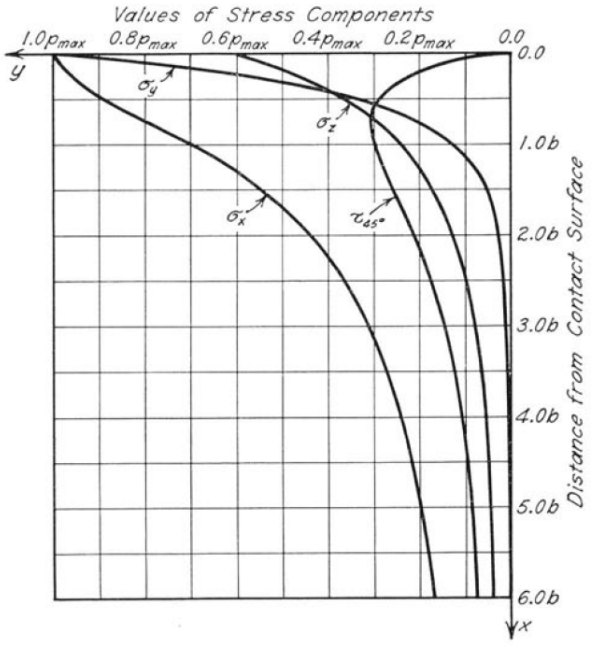


Figure 2.2: *Distribution of Stresses in Contact Zone Along Line of Symmetry [49]*

At the point of maximum shear stress, the inelastic yielding will take place and will happen when the shear stress exceeds the critical shear stress for a given material [63]. The shear stress can be calculated in two different ways, Tresca shear stress or von Mises effective stress. The Tresca shear stress is calculated with σ_1 and σ_3 being the largest and smallest principal stress respectively and is the one used above.

$$\tau_{max} = \frac{\sigma_1 - \sigma_3}{2} \tag{2.10}$$

The von Mises effective stress is defined in the following equation and can be seen as the root mean square value of the shear stress in a material point

$$\sigma_{\mathrm{vM}} = \frac{1}{\sqrt{2}}\sqrt{(\sigma_1-\sigma_2)^2+(\sigma_2-\sigma_3)^2+(\sigma_3-\sigma_1)^2} \tag{2.11}$$

The hydrostatic stress or volumetric stress (σ_h) in the material is the mean value of the normal stresses acting in a material point calculated with

$$\sigma_h \equiv \frac{\sigma_{ii}}{3} = \frac{\left(\sigma_{xx}+\sigma_{yy}+\sigma_{zz}\right)}{3} \tag{2.12}$$

The hydrostatic stress can be employed in many different forms in equivalent stress criteria but normally in one of the following ways, time dependent value $\sigma_h(t)$, maximum value during a cycle $\sigma_{h,max}$ or mid value during a cycle $\sigma_{h,med}$. The hydrostatic stress is a scalar and is stress invariant.

2.3. Rolling Contacts

Up to this point only stationary behaviour of a line contact has been discussed but in this paper the focus is on rolling contacts which behave differently compared to stationary contacts. It was assumed in this paper that there was no relative motion between the surfaces and that the rolling was pure and therefore the tangential component was negligible, in other words, no friction is assumed throughout the whole paper. But in rolling contacts, the amount of tangential force can play an important role in the material response and is briefly investigated in this section. This will become important when differentiating between the failure mechanisms and to decide which failure mechanism is more prominent in the DOT DDP.

2.3.1. Orthogonal Shear Stress

The maximum orthogonal shear stress is equal to $\tau_{maxorth} \cong \pm 0.256 \cdot P_{max}$ and is found at a depth of $0.5 \cdot b$. The maximum shear stress and maximum orthogonal shear stress differ in several ways. The maximum shear stress is found further from the contact surface and is larger but the orthogonal shear stress has two components and therefore $\Delta\tau_{maxorth} \cong 0.512 \cdot P_{max}$ which is larger compared to maximum shear stress. During rolling, the material experiences this full shear stress reversal while the other principal stresses such as the maximum shear stress remains compressive and do not show this reversal.

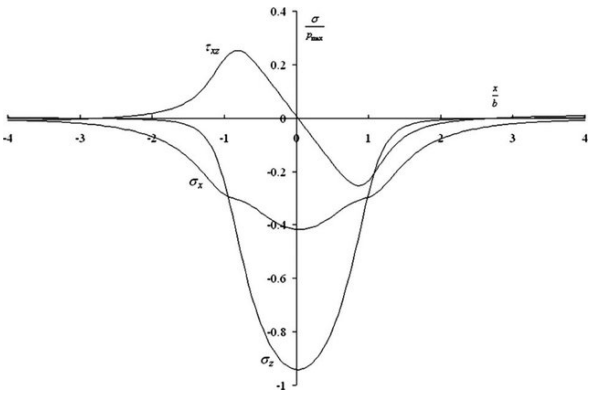


Figure 2.3: *Subsurface stress at the depth of maximum orthogonal shear stress in a Hertzian line contact [54]*

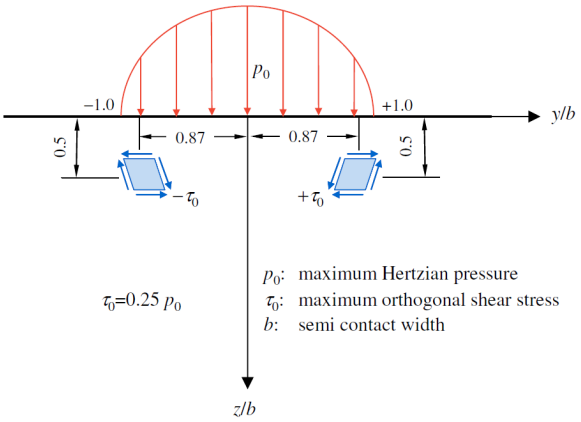


Figure 2.4: *Location and magnitude of the maximum orthogonal shear stress τ_0 [13]*

2.3.2. Rolling with Friction

The tangential force between the two surfaces causes a relative motion between these surfaces and causes slip which results in friction. This tangential force can simply be calculated with the basic formula for friction $F_x = \mu F_N$. The addition of this tangential component causes a change in the magnitude and location of the maximum shear stress. In Figure 2.5, the contours of the shear stress for a point contact are shown with $\mu = \frac{F_{tan}}{F}$. When $\mu = 1/9$, the maximum shearing stress for a line contact occurs at the surface and not underneath the surface which occurs during free rolling [63]. It is important to note that in the the RTB, the friction is negligible and therefore friction will not be included but for later research this might become more important.

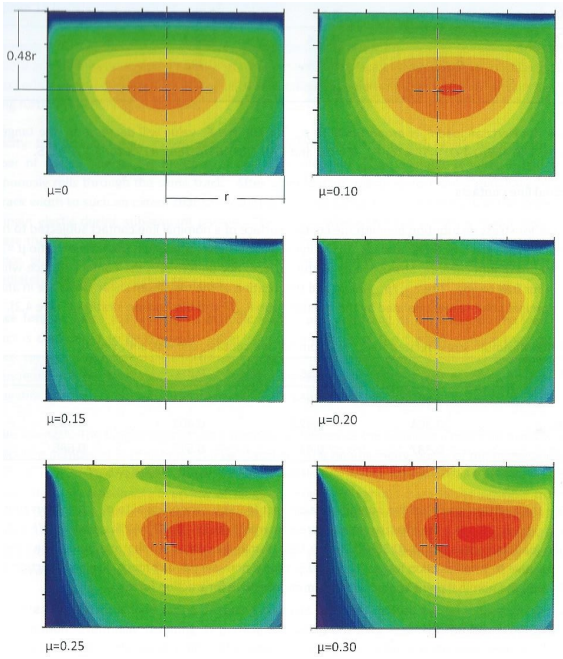


Figure 2.5: *Contours of shear stress beneath a nominal point contact with $\nu = 0.3$ [63]*

2.3.3. Rolling Resistance

Rolling resistance is mainly made up of hysteresis loss and slip, but another factor that can increase resistance is plastic deformation. During rolling, indentation happens at the leading edge of the contact area while relaxation happens at the trailing edge. The rate of deformation that occurs in the material is not equal to the rate of recovery since energy is transferred to heat during the deformation which is called elastic hysteresis loss. Another type of loss is micro slip where the contact surface under load deforms and forms an area of slip and stick (Fig. 2.6). Micro slip is also called Reynolds slip and also occurs in rolling contacts without tractive forces.

Figure 2.6: *Reynolds slip, central stick zone [63]*

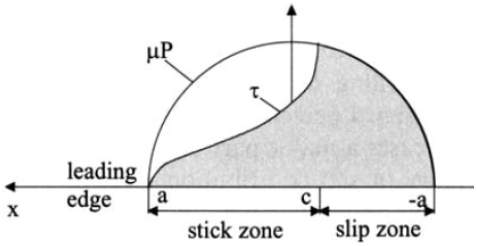


Figure 2.7: *The solid line μP shows the maximum tangential stress possible in the contact area [58]*

In the book Contact Mechanics and Friction it is stated that:"For a driven or braking wheel there is always a sticking domain that exists in the leading edge and a slip domain that exists in the trailing edge."[48]. The slip contributes to part of the rolling resistance that a rolling object experiences.

2.4. Failure of Rolling Contacts

After getting to grips on the basics of static and rolling contacts, it is possible get an idea of what the main failure modes are in a rolling contact. The failure modes of rolling contacts can be split up in a couple groups namely rolling contact fatigue, wear, plastic flow and bulk failures. The main failure mechanism for most rolling contacts is rolling contact fatigue while other minor failure modes are wear and plastic deformation [60] [63]. The causes of these failure modes will be discussed in this section.

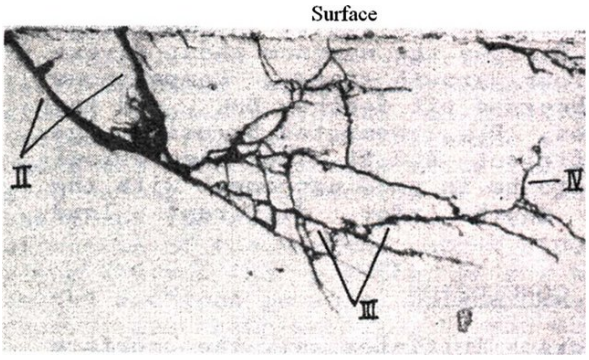


Figure 2.8: *Subsurface cracks in rolling contact fatigue [54]*

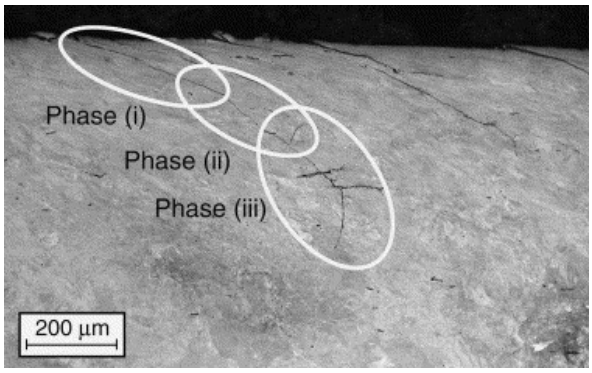


Figure 2.9: *Three phases of the life of a surface originated crack during rolling contact fatigue [53]*

2.4.1. Rolling Contact Fatigue (RCF)

Fatigue is the formation of damage in the material due to repeated loading of an element until the accumulation of damage forms a crack which grows to a size that causes the material to fail. The fatigue strength of materials subjected to repeated loading is known to depend not only on the maximum stress but also upon the manner in which the stresses vary during a loading cycle [49]. The differences between classical fatigue and RCF make it impossible to directly apply results of classical fatigue to RCF. The differences are listed in the paper of Sadeghi et al. [54] but the main difference is that RCF is a multiaxial non proportional phenomenon. Non proportional loading means that the stress components in the material do not change in the same proportion to each other which means that the principal directions change and the magnitudes of the principal stresses vary during the load cycle [17]. This can be seen in Fig. 2.3 where the orthogonal shear stress τ_{xy} completely reverses while the other principal stresses (σ_x and σ_z) remain compressive.

Depending on the type of loading, the initiation of rolling contact fatigue can be found on two different places, subsurface or surface initiated fatigue. A tangential component can change the magnitude and location of the maximum stress and therefore the location of inelastic yielding or crack initiation changes (Fig. 2.5) which was discussed in the previous section (Subsection 2.3.2). Subsurface initiated fatigue is the normal form of RCF and happens due to material fatigue on places with no or low traction, therefore the maximum shear stress and the maximum orthogonal shear stress can be found below the surface where the yielding will occur. Surface initiated fatigue initiates in places with high tangential loads which causes the shear stress to move to the surface. The cracks will progress transversely to the surface and can cause spalling and flaking of the surface.

2.4.2. Deformation and Wear

Wear can be corrosive or frictional wear in which the surface material is slowly removed but it can also be abrasive wear caused by contaminants which damage the surface and thus increase the wear on the surface. These types of failure are not the main cause of failure in this project and depend mainly on external factors or are caused by extreme slip. Another factor of failure might be plastic deformation which is caused by extreme stress. Metallic materials deform plastically after a critical stress is exceeded and when a material is loaded in tension up to a critical stress, failure will occur after a certain deformation. In contrast, if the elastic limit is exceeded while two metallic parts are compressed, then these part will be welded together [48]. For this case it is assumed the material mainly deforms elastically and does not enter the inelastic domain.

2.5. Fatigue

In this section, fatigue is studied in more detail since it is the main mode of failure in the DOT DDP and with that, the source of fatigue and the types of fatigue are discussed. Fatigue was chosen as the main mode of failure due to the contact being loaded elastically and because the surface is free of lubrication and other contaminants. As was stated earlier, it is assumed that no friction occurs on the surface which will be important when deciding what the fatigue type is.

At first, the types of fatigue life is discussed and with that a short introduction to the most important concepts of fatigue which are material and crack behaviour. Material behaviour is strongly influenced by the load and the possible types of behaviour are discussed in the section on *Shakedown* 2.5.2. The behaviour of the material in turn influences the type of failure and the crack behaviour. This crack behaviour is discussed in detail in the sections *Crack Formation* 2.5.3 and *Crack Modes* 2.5.4 which will be important to understand when trying to predict the fatigue life.

2.5.1. Fatigue Life

It is important to differentiate low and high cycle fatigue life since the material response is completely different and it is useful to know during finite element modelling. Low cycle fatigue is fatigue that occurs up to 10^4 cycles and it is mostly found to be due to plastic shakedown or ratcheting. High cycle fatigue occurs from 10^4 load cycles and up and is predominantly in the elastic domain. For this project, high cycle fatigue is of main interest since the material is loaded elastically.

Even if there is no global plasticity, there will always be some weak zone that experiences a stress above the yield limit and deforms plastically. During the life of the rolling contact surface, irreversible deformation and cracks form due to these weak zones [12]. These cracks go through three stages: crack initiation, crack propagation and final fracture (Fig. 2.10 and 2.14). During the high cycle fatigue life, crack initiation takes up most of the fatigue life.

In the high cycle fatigue regime, two fatigue domains corresponding to finite and infinite lifetime can be considered. From observations at the macroscopic scale, the material is in elastic shakedown but looking at the grains (mesoscopic scale), it is agreed upon that elastic shakedown only occurs in the case of infinite lifetime. For finite lifetime, the orientation of the grains causes plastic shakedown or ratcheting to occur which leads to failure after a finite number of cycles. The stress concentration due to this mesoscopic failure is the origin of the initiation of a macroscopic crack associated with failure on the macroscopic scale [22].

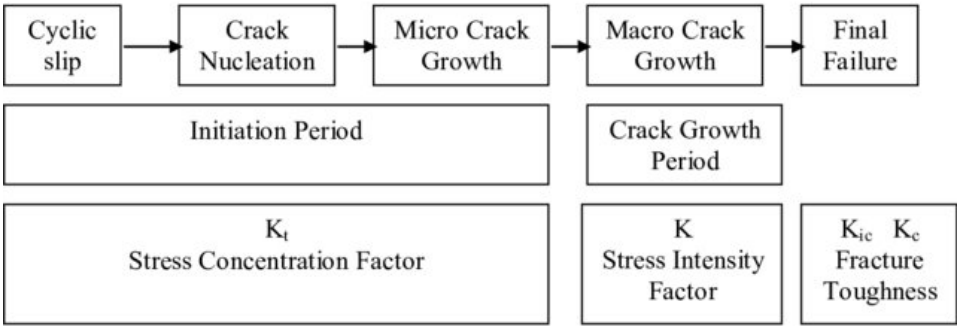


Figure 2.10: *The different phases of fatigue life [2]*

2.5.2. Shakedown

The material response during cyclic loading is strongly influenced by the magnitude of the load but also by the material hardening, residual stress state and the change in contact conditions due to deformation or wear [53]. In Fig. 2.11 the cyclic deformation of a material is shown under different magnitudes of applied load in which four different responses can be seen, a) elastic, b) elastic shakedown, c) plastic shakedown and d) ratcheting [12].

Elastic

In a, there is no global yielding since the load is below the elastic limit σ_y and therefore the global deformation is elastic. At dislocations in the material or due to surface roughness, peak stresses occur and there might occur local yielding of the material. If the stress in the material is above the fatigue limit σ_{fl}, fatigue cracks may initiate.

Elastic Shakedown

In b, the material experiences elastic shakedown since the material is loaded above the elastic limit σ_y but below the elastic shakedown limit σ_{el}. Initially, yielding occurs and the residual stresses in the material rise and these reduce the plasticity in the material. If plastic hardening occurs, the elastic limit increases and the material will return to behave elastically. Therefore if the shakedown period is short, the majority of the fatigue life will be spend in the elastic domain.

Plastic Shakedown

In c, the material experiences plastic shakedown and is loaded above the elastic shakedown limit σ_{el} but below the plastic shakedown limit σ_{pl}. The material experiences plastic flow in both tension and compression but after a couple cycles, it goes into a closed loop of plastic shakedown. During plastic shakedown, plastic deformation occurs but there is no net accumulation of plastic strain and the amount of cycles until fatigue failure occurs is low. The plastic strain causes dislocation pile-ups, shear band formation and crack formation between grains.

Ratcheting
In d, the material is loaded above the plastic shakedown limit σ_{pl} and experiences ratcheting. The residual stresses, hardening and reversed plastic flow is not enough to prevent the plastic deformation of the material and the accumulation of plastic strain in the material causes the component to fail. The deformation grows during each load cycle and the amount of cycles until fatigue failure occurs due to the plastic deformation is low [12].

In Figure 2.12, the different types of material responses are shown with the load factor p_0/k on the ordinate and the friction coefficient λ on the abscissa. The load factor is made up of the maximum normal contact pressure p_0 and the material shear yield strength k. The map shows the position of the fatigue damage and in which domain the loading occurs [53].

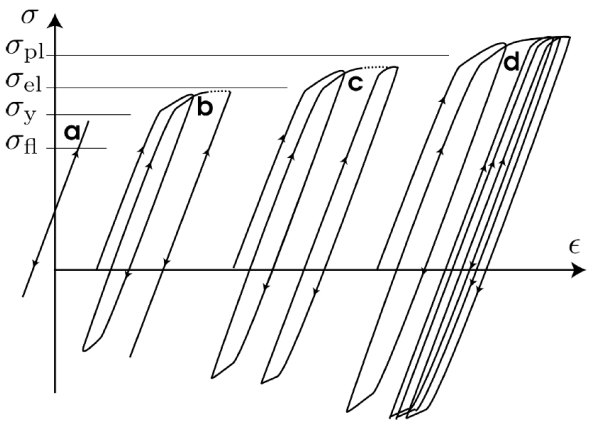


Figure 2.11: *Different responses of a metallic material to cyclic loading. X-axis with strain (ϵ), y-axis with stress (σ) [12]*

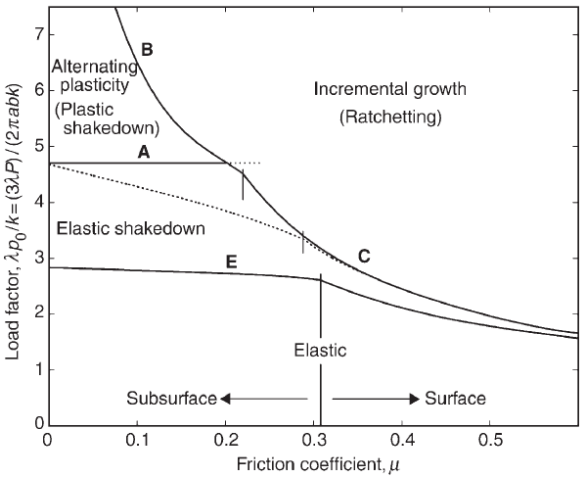


Figure 2.12: *A shakedown map for a general three-dimensional non-conformal rolling–sliding contact. X-axis with friction coefficient (μ), y-axis with load factor ($\frac{\lambda * p_0}{k}$) [53].*

2.5.3. Crack Formation

The whole process of crack formation is shown in Fig. 2.10 and is explained in detail in this section. Crack formation starts off with crack initiation (Stage I) which consists out of three phases, cyclic slip, crack nucleation and small crack growth and is also known as the non-continuum fatigue crack growth mechanism. The crack nucleation is on the order of grain size while crack propagation is of the order of several grain diameters and therefore this process depends on the material microstructure, the stress ratio and the environment [62]. A fatigue crack can initiate in two different ways, due to stress concentrations or due to dislocations in the metallic crystals. The stress concentrations are formed due to the inhomogeneous nature of the material and the formation of cracks can be promoted by the residual stresses in the material [12]. The dislocations on the other hand cause inhomogeneity and weaken the metallic structure and this can cause the crystal planes to slip. During the load cycles, the dislocations in the material start to propagate plastically which causes the dislocations to coalesce and the crystal planes to slip. The repetition of the cycles causes the coalescence of dislocation to form a crack.

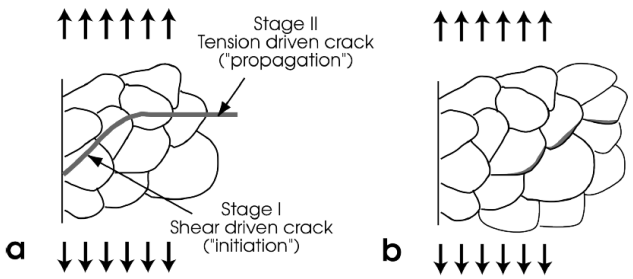


Figure 2.13: *a. Initiation from slip bands b. Initiation and growth between the grains [12]*

The crack initiation stage is mainly a shear stress driven but it has been a highly discussed topic which type of shear stress is the main cause of crack initiation. Lundberg and Palmgren supposed that the crack initiation occurs at the same depth as the maximum orthogonal stress occurred at and therefore the orthogonal shear stress being the most detrimental to the fatigue process [35][41]. Experiments have also shown that orthogonal shear stress is the most damaging stress component when talking about rolling contact fatigue because of its alternating nature [3][33].

In stage II (crack propagation), this crack will propagate through the material with stable crack growth, also called power growth, perpendicular to the load direction. In many cases, this stage is normal stress driven but in rolling contacts, the normal stresses are compressive and inhibit crack opening and the shear stress along the grain boundaries is the damage causing stress [27]. If the ensemble average of cracks is looked at, the generalized form of Paris' law (Eq. 2.13) can be used. It gives a linear relationship between crack rate $\frac{da}{dN}$ and stress intensity factor K in a bi-logarithmic scale[36]. As seen in Fig. 2.14, the stage has a lower bound, fatigue crack threshold ΔK_{th}, and an upper bound, fracture toughness of the material ΔK_C (or ΔK_{IC}).

Stage III is the unstable crack growth stage, the crack growth accelerates and ΔK_C (or ΔK_{IC}) is approached by ΔK_{max}. This stage is sensitive to the microstructure, load ratio and the stress state and in this stage the crack moves to the surface and the surface experiences failure [12][54][62]. Most papers also assume that these cracks initiate and propagate along the grain boundaries as these are seen as the weak planes in the material.

$$\frac{da}{dN} = C\Delta k_{eff}^{n} \tag{2.13}$$

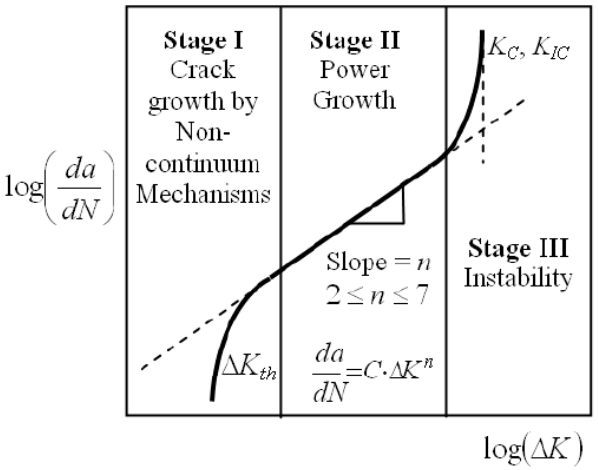


Figure 2.14: *Fatigue crack propagation stages. Y-axis crack growth per cycle $\frac{da}{dN}$ (mm/cycle) and x-axis the stress intensity factor range ΔK. ΔK_C is the plane stress fracture toughness, ΔK_{IC} is the plane stain fracture toughness. Stage I is the crack initiation phase, stage II the crack propagation stage and stage III is final fracture. [62]*

2.5.4. Crack Modes

Since the loading is multiaxial, the crack will experience several loading modes. There are three different modes, modes I, II and III but for a two-dimensional fatigue study only mixed mode I and II loading is important since mode III is out of plane and therefore $K_{III} = 0$. Mode I loading occurs most frequently in most cases and is the mode that produces the most damage and is often called the opening mode. Mode II is the shearing of the crack faces due to in-plane shear stresses but it is generally seen as the more unstable crack propagation mode and therefore in mixed mode I and II loading the crack will typically go to pure mode I loading instead. Mode I is also seen as the compressive stress and when the crack is closed, the mode I stress intensity factor (SIF) is 0 and during a rolling contact cycle the SIF for mode II varies a lot. It is in the absence of strong shear tractions acting at the contact site that mode I crack growth is suppressed by the compressive hydrostatic component of the stress field and mode II crack growth is expected [3]. Therefore the most important stress that affects the rolling contact fatigue life is shear stress.

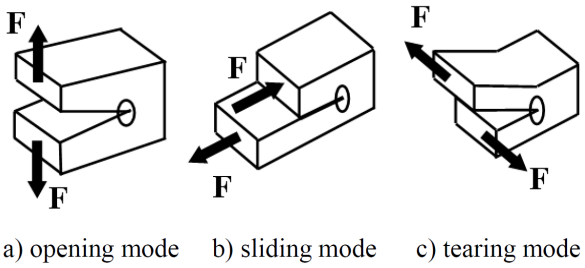


Figure 2.15: *The different loading modes of a crack [62]*

2.6. Predicting Fatigue Life

Since fatigue is the predominant mode of failure in the rolling elements of the DOT DDP, the life of the rollers is governed by its RCF life [54] but it is impractical to experimentally test the life and therefore a reliable life calculation is a proper substitute to testing or to make a proper prediction about the life. Therefore, the prediction of RCF life has been and still is a subject of great research interest. Due to the localized nature of contact pressure, the effect of the heterogeneous microstructure and the presence of microstructural defects is enhanced. These factors are different in each roller since the microstructure is never the same and this causes the scatter in the fatigue life for, what seems, identical rollers. Several methods have been proposed to estimate lives of rolling contacts under RCF which have been summarized in the work of Sadeghi [54]. Early models [16] were focussed on developing formulas based on extensive full-scale fatigue tests of bearings for determining their service life and loading performance. The development of these models was expensive and took a very long time. Lundberg and Palmgren later developed a theory that laid the basis of several other models and is still used in the ISO 281 bearing load-life equations. These models account for the scatter in fatigue life by employing a Weibull probability distribution function [3]. Later models were more deterministic and focussed on a homogeneous material structure and looked for the critical stress by applying a critical plane method. More recent models have been focussing on the physical mechanisms that cause RCF and this approach has been led by professor Sadeghi and his research group and utilizes a randomly generated microstructure and applies damage mechanics to simulate the damage induced on the material structure each load cycle. The models have been able to give life estimations, life scatter and spall profiles that compare well with experimental data [3]. The models discussed above can be classified into probabilistic engineering models, combined methods and continuum damage mechanics (CDM) models.

The probabilistic models include variables that are obtained from extensive experimental testing and do not directly consider the behaviour of materials under contact loading and the residual stress and strain in the contact areas. The other two are theoretical and require complete stress-strain behaviour of the contact surfaces and take the mechanics of failure into account. Some models focus on just initiation or propagation while others account for both mechanisms and put it into one model. Some models assume homogeneous material structure while other models assume a heterogeneous material structure and therefore differ in the processes they focus on. Some implement microscale processes while others assume an existing microcrack in the model and employ fracture mechanics to predict the propagation of the crack. The combination of two different models is shortly discussed in the combined methods section while the CDM approach focusses on both mechanisms at once.

2.6.1. Probabilistic Engineering Models

Probabilistic engineering models give a solid base for predicting the fatigue life of different types of bearings. These models can be adjusted to the DDP and RTB to give a fatigue life prediction but these are fairly general and do not implement specific material or geometric features. Many models have been developed, each with a different focus and approach, but many of them are based on the Lundberg-Palmgren (LP) theory. This section discusses a couple of these models with a focus on analysing the DDP.

The first basis for calculating bearing life was provided by LP who supposed that crack initiation occurs subsurface due to the repetition of a maximum shear stress and at the weak points in the material. They hypothesized that these weak points were stochastically distributed in the material and a Weibull statistical

strength theory was applied to determine the probability of survival (S) from subsurface initiated fatigue. The implementation of the Weibull slope into the model considers the scatter found in the experimental data on bearing lives. A Weibull slope is the representation of the scatter that is found in a dataset, the higher the slope, the lower the scatter.

$$\ln \frac{1}{S} = A \frac{N^e \tau_0^c V}{z_0^h} \tag{2.14}$$

where τ_0 = Maximum orthogonal shear stress in the contact
A = Constant (Determined experimentally)
c = Stress criterion exponent (Determined experimentally)
e = Weibull slope
h = Depth exponent (Determined experimentally)
N = Fatigue life
S = Probability of survival
V = Stressed volume of material, $V = az_0(2\pi r_r)$ with a, z_0 and r_r the width, depth and length
z_0 = Depth of τ_0 (mm)

The LP equation (Eq. 2.14) relates the critical stress-life exponent to the Weibull slope e for a given S and bearing dimensions. Therefore the stress life equation depends on the scatter of the experimental bearing life data.

$$N \propto \frac{1}{\tau_0^{((c-h+1)/e)}} \tag{2.15}$$

To calculate the fatigue life, the formula (Eq. 2.16) for basic rating life is commonly used in the industry (ISO 281) and has been used together with the LP theory. While this theory has a very solid basis to calculate fatigue life, it does not take into account surface initiated failure, lubrication and the physical phenomenon of RCF and the geometry of the rollers. Therefore many other models were developed, each with a different focus or purpose, thus not all of them as relevant.

$$L_{10} = \left(\frac{C}{P}\right)^p \tag{2.16}$$

where C = Basic dynamic load rating
L_{10} = Basic rating life with 10% failure probability
P = Equivalent dynamic bearing load
p = 3 for ball bearing, 10/3 for roller bearings and 4 for perfect line contacts

Other models that were focussed on subsurface initiated fatigue were Chiu et al. [8], Ioannides and Harris [24], Harris and Barnsby [18], Miyashita et al. [39], Zaretski [6], etc. The model from Chiu et al. was based on a crack propagation law and attributed spalling to material defects and applied a severity distribution to the size and physical nature. The model from Ioannides and Harris modified the LP model and put the focus on crack initiation. The Ioannides-Harris (IH) model implemented discrete material volumes with individual probabilities of survival and added a stress threshold below which no failure would occur. The Harris and Barnsby model modified the IH model to a more generalized model and applies a single stress life factor to modify fatigue life, predicted by the LP theory. The model from Miyashita studied RCF of sintered alloys using FEM and RCF tests. They estimated the location of the maximum shear stress and this coincided with the crack initiation depth observed in the experiment. Zaretsky modified the LP model such that the stress-life relation was not dependent on the Weibull slope e and the dependence on the depth term was removed [54].

The Ioanniddes and Harris (IH) model is based on the LP theory and applies modifications to the formula (Eq. 2.14) formed by LP. It assumes a discrete material volume with individual probabilities of survival and these discrete volumes are integrated to obtain the overall risk of failure for the contact. There is also a stress threshold σ_u which can be compared to the fatigue limit below which no failure will occur. Applying these modifications to Eq. 2.14 gives the following equation.

$$\ln \frac{1}{S} = AN^e \int_V \frac{(\sigma - \sigma_u)^c}{z^h} dV, \quad \sigma > \sigma_u \tag{2.17}$$

where σ = Stress at depth z (Not limited to orthogonal shear stress, can be other stress measure)
σ_u = Stress threshold
z = Depth of maximum Hertzian shear stress (mm)

The load-life equation was also modified to implement the modifications from IH into Eq.2.16 and can be seen below.

$$L_{10} = \frac{A}{\left[1 - \left(\frac{P_u}{P}\right)^w\right]^{c/e}} \left(\frac{C}{P}\right)^p, \quad P > P_u \tag{2.18}$$

where P_u = Load corresponding to stress threshold
w = Constant (Determined experimentally)

The idea behind the probabilistic models is applying the Weibull strength theory with including the material microstructural characteristics. The resulting lives are based on the scatter of experimental data which follow the Weibull distribution and is therefore not influenced by material inhomogeneity or the random material structure [54].

2.6.2. Combined Methods Approach

In the combined methods approach, a crack initiation and a crack propagation model are combined to calculate the fatigue life. For crack initiation in a non-proportional multiaxial loaded surface, the stress-based, strain-based and energy-based are all possible modelling methods (Fig. 2.16). After the application of these methods, Linear Elastic Fracture Mechanics (LEFM) can be applied to analyse the crack propagation and calculate the total life [53].

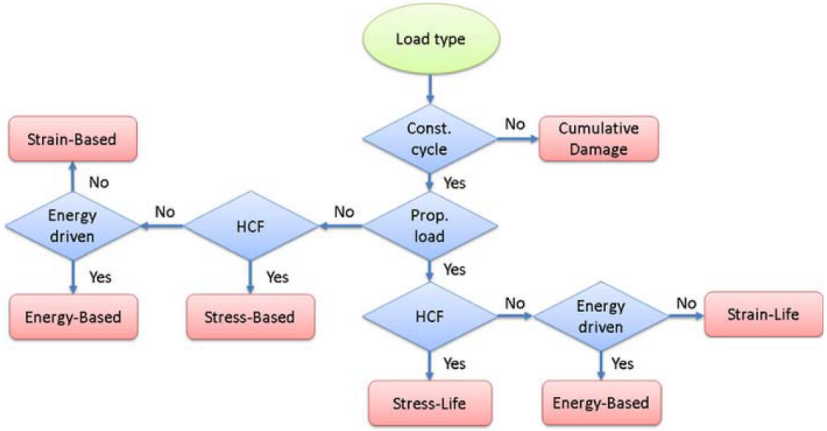


Figure 2.16: *Fatigue Model Selection [9]*

The stress and strain based methods are both critical plane methods which examine different orientations in space in order to find a critical plane on which fatigue initiation occurs. The critical plane is a plane that, depending on the method, maximizes a stress or strain expression for a certain orientation in a point. The energy-based method results in a lifetime prediction in terms of cycles to fatigue and provides a dissipated fatigue energy density. The stress-based models aim at high-cycle fatigue and result in a fatigue usage factor and result in no life prediction. The strain-based models aim at low-cycle fatigue and result in a lifetime prediction in terms of cycles to fatigue, the model is mainly aimed at the initial plastic deformation of the material. Since the the focus is on high-cycle fatigue ($> 10^4$) and gathering data on the effect of scaling on the amount of load cycles to failure, the stress-based and energy based method are discussed in this section.

Stress-Based Fatigue Models
The stress-based fatigue model can be evaluated with the Findley, Matake, Normal Stress or Dang Van model. The Dang Van model takes compressive stress into account and can study multiaxial fatigue with rotating directions of principal stresses which makes it suitable for contact fatigue analysis. Dang Van is based on the parameters that have the highest influence on crack initiation. The importance of shear stress on crack nucleation is combined with the influence of hydrostatic stress on crack propagation in the Dang Van model via

$$\begin{aligned} \tau_{\mathrm{DV1}} &= \tau_{\mathrm{a}}(t) + a_{\mathrm{DV}}\sigma_{\mathrm{h}}(t) > \tau_{\mathrm{e}} \\ \tau_{\mathrm{DV2}} &= \tau_{\mathrm{a}}(t) - a_{\mathrm{DV}}\sigma_{\mathrm{h}}(t) < -\tau_{\mathrm{e}} \end{aligned} \tag{2.19}$$

where $\tau_{\mathrm{a}}(t)$ is the time-dependent value of shear stress, $\sigma_{\mathrm{h}}(t)$ is the time-dependent value of hydrostatic stress, τ_{e} represents the fatigue limit in pure shear and a_{DV} is a material constant representing the influence of hydrostatic stress. It states that fatigue initiation occurs during a stress cycle if the combination of $\tau_{\mathrm{a}}(t)$ of a shear stress (on the most damaging shear plane) and the value $\sigma_{\mathrm{h}}(t)$ of the hydrostatic stress at the considered material point fulfils one or both of the two inequalities above during some time portion(s) of the full stress cycle [11].

From the inequalities (Eq. 2.19), the values τ_{DV1} and τ_{DV2} are calculated and applied in a Wöhler diagram. With this diagram, the cycles to crack initiation can be found for each critical plane. The method first looks for the plane with crack initiation and identifies its position and orientation. Secondly, it calculates the time to crack initiation on that plane. The Palmgren-Miner rule states that the crack initiation occurs when the total accumulated amount of damage D reaches one [11] and following this rule, as seen below, calculates and accumulates the damage at specific material points. For each shear plane of these points, the largest damage during a cycle is added to the accumulated damage.

$$D = \sum_{i=1}^{m} D_i = \sum_{i=1}^{m} \frac{n_i}{N_i} \tag{2.20}$$

With N_i, the fatigue life for a certain stress and n_i the amount of stress cycles taking place. After the life until crack initiation is found, a linear elastic fracture mechanics (LEFM) model can be applied to calculate the crack propagation life since it follows a linear relationship between $log\frac{da}{dN}$ and $log\Delta K$ (Fig. 2.14). A simple method to use is Paris' Law.

$$\frac{da}{dN} = C_p(\Delta K)^{mp} \tag{2.21}$$

The curve fitting parameters C_p and mp are easy to obtain and therefore this method is easy to apply but this method does not take the material structure and its randomness into account and is also mainly focussed on crack initiation.

Reference [11] uses the Dang Van fatigue criterion together with the Palmgren-Miner damage accumulation law to study triaxial fatigue with rotating directions of principal stresses. The Hertzian contact pressures are analytically found, and the corresponding subsurface stresses are calculated using a numerical integration scheme starting from the exact point force solutions of Boussinesq and Cerruti. This paper also applies the Wöhler diagram with the equivalent Dang Van stresses for the calculation of the stress cycles to failure. The method is later on further developed but it still does not account of the heterogeneous microstructure [10].

Energy-Based Fatigue Models
The energy-based fatigue model can be evaluated with the Morow or Darveaux model. The models depend on energy dissipation, meaning that the energy cannot be restored since it is dissipated into the material. The Morow model is used to model failure defined as the crack initiation and relates the plastic strain on the microscopic level to the movement of dislocations. The Darveaux Model separates fatigue life into crack initiation and crack propagation. Both models give the amount of load cycles to failure but requires the calculation of the dissipation of creep, plastic or viscoplastic energy which are not applicable in this project. It seems that the energy-based fatigue models do not suffice the needs of this research and therefore other methods have to be applied and combined.

Alternative Fatigue Models
The WF method is an alternative method which mainly focusses on the propagation of subsurface cracks by placing cracks into the material and calculating the speed at which this crack propagates. Subsurface cracks, typically keep parallel to the contact surface for most of their growth, and all geometric parameters, unless crack length/depth ratio, do not change. Since the focus is on propagation while the majority of the rolling contact life is in the crack initiation phase, this model does not fit the paper well.

2.6.3. Continuum Damage Mechanics Method
The first work on CDM was found in the works of Kachanov and Rabotnov who considered the creep of metals and has been the basis for plenty of research in the following years. CDM takes various damage processes in materials into account and is able to describe heterogeneous microstructural behaviour during straining of the material and structures at the macro scale. Several effect can be characterized and presented by continuum damage mechanics like elastic and plastic deformation but also other material behaviour can be described even if the structure is not homogeneous due to micro defects or varying grain shapes and sizes [5]. The method discussed below is a method developed and refined by Sadeghi and his research group [27][34][50][54].

The focus of developing a damage theory is predicting the lifetime of a structure. Damage mechanics theory starts with developing a damage parameter since damage is not directly measured with stress or strain. The damage variable D represents the gradual deterioration of materials before crack initiation and is introduced into the material constitutive stress-strain relationship [50]. The value $D = 0$ corresponds to a pristine material and $D = 1$ is a fully damaged material and this damage is generally seen as crack initiation, a defect of which the size is large compared to the already existing material defects. This microcrack is assumed to initiate when the accumulated damage equals unity ($D = 1$) in an element [54] and is thought to nucleate along the grain slip bands making the problem highly dependent on the microstructure [37]. Evolution of damage is calculated separately by a damage parameter, and is included in the FE analysis by updating the material stiffness (locally) at the occurrence of fatigue damage [53]. In the current study, the material is assumed to be isotropic, the material properties to be homogeneous and the grain boundaries are assumed as the weak planes where the cracks nucleate. Since this model applies the microstructure of the material, it can take all the factors that influence fatigue into account and therefore it is the preferred modelling technique.

2.7. Modelling of Rolling Contact Fatigue
After discussing the possible life prediction methods, the CDM method came out on top since it takes the microstructure into account and has shown promising results. This section goes over the theory and principles used for the CDM model and talks in detail about the decisions made in this paper and in previous research. In the next chapter the implementation of the model in the software and the assumptions made to build the model are discussed. The software used in this paper is Matlab and COMSOL and specifically COMSOL Livelink which makes it possible to extend the programming from Matlab into COMSOL seamlessly. Matlab is used for doing the calculations, programming and building the model while COMSOL is used for the FEM analysis.

2.7.1. Continuum Damage Mechanics Model
The model introduced here incorporates cyclic damage accumulation and progressive degradation of material properties with rolling contact cycling [50] and this degradation due to the repeated loading of the surface will cause nucleation and propagation of microcracks until failure occurs. For each load cycle, the model has to solve the stress distribution through the material microstructure because damage evolution is a stress-based formulation. Therefore the materials stress strain response needs to incorporate the material degradation (damage) of each pass and this is done by modifying the constitutive equation to incorporate damage which can be seen below.

$$\sigma_{ij} = C_{ijkl}\,(I_{klmn} - D_{klmn})\,\varepsilon_{mn} \tag{2.22}$$

where σ_{ij} is the stress tensor, C_{ijkl} is the material stiffness tensor containing the elastic constants, I_{klmn} is the identity matrix, D_{klmn} is the damage tensor and ϵ_{kl} is the strain tensor. Since the material is considered to be isotropic, the damage tensor becomes a scalar and the equation (Eq. 2.22) simplifies to the following.

$$\sigma_{ij} = C_{ijkl}(1 - D)\varepsilon_{kl} \tag{2.23}$$

The addition of the damage term in the equation causes a reduction in stiffness of the material with D ranging from zero (undamaged) to one (completely damaged).

The evolution of damage for high cycle fatigue can be related to the stress level at that point through the following non-linear equation.

$$\frac{dD}{dN} = f(\sigma, D) \tag{2.24}$$

Where N is the number of stress cycles and σ is a stress measure. The commonly used equation for the evolution of damage for one-dimensional fatigue based on Eq. 2.24 is seen below.

$$\frac{dD}{dN} = \left[\frac{\Delta\sigma}{\sigma_r(1-D)}\right]^m \tag{2.25}$$

where

$$\sigma_r = M_0\left(1 - b\frac{\sigma_m}{\sigma_u}\right) \tag{2.26}$$

where N is the amount of cycles, $\Delta\sigma$ is the stress range of the damage causing stress, σ_m the mean stress, σ_u is the ultimate stress and M_0, b and m are experimentally identified material parameters with σ_r a function of the mean stress [27]. For rolling contacts, the shear stress reversal causes the formation and propagation of subsurface cracks and the normal compressive stress does not cause any damage. Since the shear stress is the main cause of damage for rolling contact fatigue, Eq. 2.25 is adapted to the following form.

$$\frac{dD}{dN} = \left[\frac{\Delta\tau}{\sigma_r(1-D)}\right]^m \tag{2.27}$$

where $\Delta\tau$ is the range of shear stress calculated along the grain edges and σ_r is the resistance stress which controls the ability of a material to resist damage accumulation. σ_r and m are material specific parameters which can be obtained from torsional fatigue tests (Fig. 2.18). These results are used since rolling contact fatigue and torsional fatigue are caused by the same shear stress reversal in the material and the results for torsional fatigue tests are widely available.

The shear stress along this grain edge is the plane shear stress and by resolving the shear stress along the grain edges, the directionality of this edge is included. This is important to include since it is assumed that the grain boundaries are the weak planes and therefore the orientation of the edge and thus the grain shape has an impact on fatigue. The differing shape and sizes of the grains causes some grains to be more resistant to damage accumulation than others. The plane shear stress is calculated with the following equation.

$$\tau_{x'y'} = -\frac{1}{2}(\sigma_x - \sigma_y)\sin(2\theta) + \tau_{xy}\cos(2\theta) \tag{2.28}$$

The angle (θ) is the angle that the normal of the edge makes to the x-axis which is parallel to the loaded surface and σ_x, σ_y and τ_{xy} are the principal stresses.

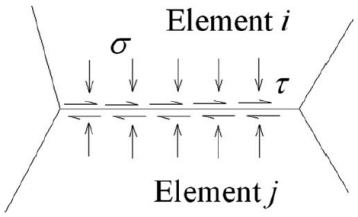


Figure 2.17: *Stress components along the grain boundaries [50]*

2.7.2. Material Fatigue Damage Properties

The parameters σ_r and m are calculated from the experimental data from torsional fatigue tests [59]. The data of these tests is placed in an S-N curve as seen in Fig. 2.18 and is used later on to find certain material constants. To begin, Eq. 2.25 is integrated for cycle to failure ($D = 1$) which can be seen below.

$$\int_0^{N_f} dN = \int_0^1 \left\{\frac{\sigma_r(1-D)}{\Delta\tau}\right\}^m dD \Rightarrow N_f = \left[\frac{\sigma_r}{\Delta\tau}\right]^m \left(\frac{(1-D)^{m+1}}{-(m+1)}\right)_0^1 \tag{2.29}$$

which results in this equation

$$N_f = \frac{1}{(m+1)} \left[\frac{\sigma_r}{\Delta\tau} \right]^m \tag{2.30}$$

Since the model is focussed on high cycle fatigue, Basquin's law [1] is considered

$$\sigma = C(N_f)^a \tag{2.31}$$

Rewriting this for the shear stress reversal $\Delta\tau$, gives us the stress-life relation for torsional fatigue where $\sigma = \frac{\Delta\tau}{2}$ and $C = \sigma_f$

$$\frac{\Delta\tau}{2} = \sigma_f (N_f)^a \Rightarrow N_f = \left(\frac{2\sigma_f}{\Delta\tau} \right)^{-\frac{1}{a}} \tag{2.32}$$

with σ_f and a being material constants and for 100Cr6 these are $\sigma_f = 2409MPa$ and $a = -0.099$. Filling in and computing Eqs. 2.30 and 2.33 gives

$$m = -\frac{1}{a}, \quad \sigma_r = 2\sigma_f \left(1 - \frac{1}{a} \right)^{-a} \tag{2.33}$$

and results in $\sigma_r = 6.113GPa$ and $m = 10.1$ for 100Cr6.

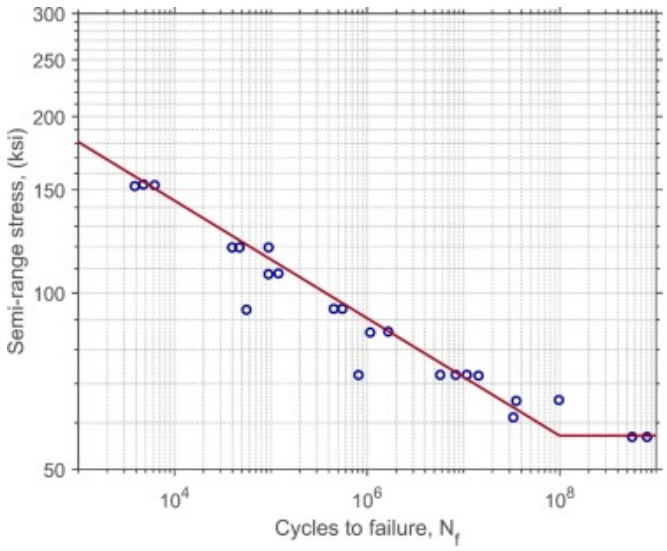


Figure 2.18: *S-N curve for bearing steel AISI-52100 or 100Cr6 [34][59]*

2.7.3. Jump-In-Cycles

Since the problem is a high cycle fatigue problem, it is computationally expensive to run all the load cycles individually. Therefore the procedure developed by Lemaitre [32], the 'jump-in-cycles' method, is used to speed up the modelling time. The method makes a jump in the cycles which depends on the maximum damage evolution that an edge in the material experiences and calculates the jump with the set increment in damage (ΔD). This method has been used for several other damage mechanics simulations ranging from of course rolling contact fatigue [3][27][34][50][57][67] to tensile fatigue [4] and torsion fatigue [65][66]. The stress field through the material is assumed to remain unchanged for a finite number of cycles ΔN, which is called the loading block. Over this block of cycles, the damage in each element is constant and equal to D_j^i where j is the element and i the loading block. As seen in Fig. 2.19, the damage evolution $\left(\frac{dD}{dN} \right)_j^i$ is assumed to be piecewise linear. The damage evolution curve is not predetermined but it is calculated with the results from the numerical simulations of the stress-damage coupled model.

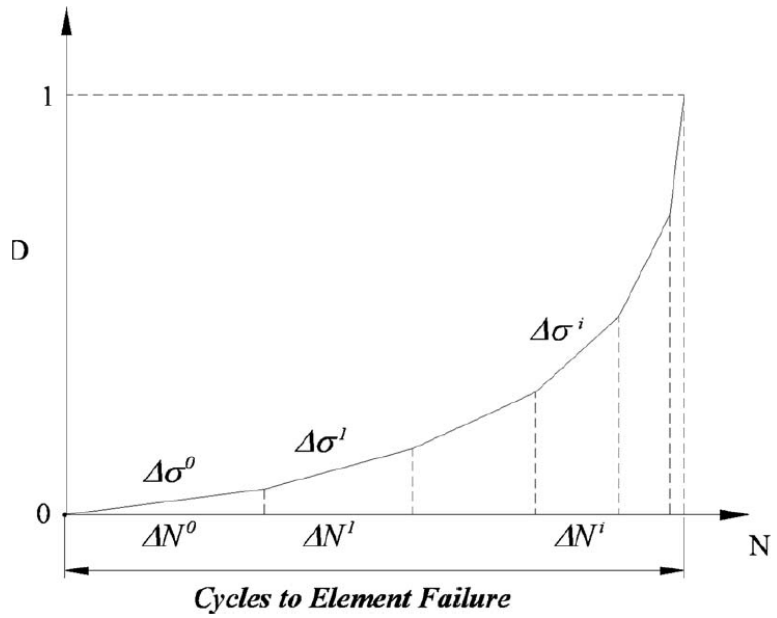


Figure 2.19: *Jump-in-cycles method: damage evolution is assumed to be piecewise linear with respect to the number of cycles [27]*

During each loading block, the material model and damage accumulation from the previous loading block is used to model the critical stress distribution running through the material. With the stress distribution calculated, the average plane shear stress (Eq. 2.28) acting along the grain boundaries at each step of the load cycle is determined and used to calculate the damage evolution rate (Eq. 2.27). For the first loading block, the material is undamaged and therefore damage for all grains is 0 ($D_j^1 = 0$). When the damage evolution is calculated for all the grain boundaries, the boundary with the maximum damage evolution is determined as the critical element.

$$\left(\frac{dD}{dN}\right)^i_{\text{crit}} = \text{Max}\left|\left(\frac{dD}{dN}\right)^i_j\right| \tag{2.34}$$

The critical element determines the number of cycles ΔN^i spend in that loading block with a constant maximum increment in damage (ΔD) over the block of cycles.

$$\Delta N^i = \frac{\Delta D}{\left(\frac{dD}{dN}\right)^i_{\text{crit}}} \tag{2.35}$$

With the jump in cycles calculated, the total number of cycles can be updated. This jump in cycles is the same for all the boundaries.

$$N^{i+1} = N^i + \Delta N^i \tag{2.36}$$

With both the damage evolution rate and the jump in cycles calculated, the damage for all the boundaries can be updated.

$$D_j^{i+1} = D_j^i + \left(\frac{dD}{dN}\right)^i_j \Delta N^i \tag{2.37}$$

2.7.4. Material Degradation

In reality, material degradation occurs through the formation of cracks and voids which has an influence on the stress-strain behaviour of the material. The continuum damage mechanics approach applied in this model is able to apply the influence of these failure mechanisms to the stress-strain material response through a damage parameter. In this model, the degradation of the material is applied to the stress-strain constitutive equations of the grain but the more accurate approach would be by applying the damage to the elements along the grain boundaries. This is a better representation of the degradation of the model since this is the area where failure occurs. The damage of the boundaries of a grain is averaged to get the damage of that grain D_{grain} which in this model represents the degradation of the material. The choice of using the

damage of the boundaries is an authors choice and is done based on the capabilities of the used software but also on previous decisions. Firstly, this choice links back to the use of plane shear stress which adds the orientation of a boundary and therefore the shape of a grain to the calculation of damage. Secondly, in COMSOL it is not possible to calculate the damage of a single finite element and therefore, the damage of a whole grain is calculated. This damage has to be applied to the stress strain relationship such that the material response is changed with the material degradation and this is done by implementing the damage parameter into the stress-strain relationship. Expanding Eq. 2.23 to Hooke's law for isotropic materials gives the following.

$$\sigma = \lambda \operatorname{tr}(\varepsilon) I + 2\mu\varepsilon = \mathrm{c} : \varepsilon \tag{2.38}$$

Where the two Lamé parameters, λ and μ are a function of Young's modulus (E) and Poisson's ration (ν). The Poisson's ratio is not affected by the damage while the Young's modulus is and by implementing the effective stress concept, the Young's modulus is replaced with $\tilde{E} = E_0(1 - D_{grain})$ in both Lamé parameters. Implementing the damage parameter into the Lamé parameters gives the following functions.

$$\lambda = \frac{E_0(1 - D_{grain})\nu}{(1+\nu)(1-2\nu)} \tag{2.39}$$

$$\mu = \frac{E_0(1 - D_{grain})}{2 * (1+\nu)} \tag{2.40}$$

This procedure is repeated until the damage in a boundary reaches 1 and at that moment this boundary is fully damaged.

2.7.5. Crack Formation

Once a boundary is fully damaged, a microcrack is introduced along this grain edge. When the crack is compressed, the crack faces act upon each other and a friction component starts to act on the faces. A friction coefficient of 0.4 has been added between the two faces to account for potential traction forces [27][41].

The cracked boundary will be excluded from further damage evolution equations since it is already completely damaged so no further damage evolution can occur. For the calculation of the damage of a grain, the cracked boundary will be used and the damage of this boundary will be set to one. Some papers set this value to zero, to one and some even remove it but this depends on the way the model is built and the reasoning behind it. The reason that the cracked boundary's value is set to one instead of the other values goes back to the material degradation equations (Eq. 2.39 and 2.40). If the value of the damaged boundary would be set to zero or even be removed, the material degradation would be reduced and the material would 'heal' and when the value is set to one, the material continues to degrade.

The initiation phase is the phase until a microcrack nucleates, after that the material is in the crack propagation phase. The propagation of the cracks is modelled by the coalescence of microcracks which form long cracks and are propagated by mode II loading [27][46]. The failure occurs when the crack reaches the surface and after this crack reaches the surface, it does not take long for a piece of material to break off and therefore a spall to create in the domain.

2.7.6. Voronoi Tessellation

Fatigue is a phenomenon that occurs on the microstructural level of a material and therefore to create a model that comes close to reality, the grain structure has to be implemented in the model. Bearing materials and other high strength steels consist of grains with differing shapes and sizes. To recreate this structure, researchers [14][15][40] developed the Voronoi cell finite element method (VCFEM). This method proved to create solid results and therefore was used and improved upon by many other researchers. Therefore this model also implements the Voronoi tessellation with random size and shape to simulate the material grains.

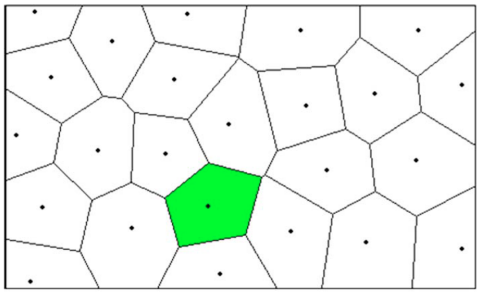


Figure 2.20: *Domain with seed points and and the Voronoi tessellation simulating the grain structure of a metal [26]*

Imagine a set of seed points that are placed in a plane (Fig. 2.20). The number of points in the domain is finite and these points do not overlap. When looking at the distance of random points in this domain to the seed points, some will be closer to a certain seed than another seed point. The points closest to a certain seed point are assigned to that seed point and when a point is as close to one seed point as to another, the point is assigned to both seed points and a boundary is formed. The points assigned to a certain seed point form a region while the points assigned to two or more make the boundary of those regions. The regions form a tessellation because they cover every location of the plane. These regions are called the Voronoi cells or Voronoi polygons and represent a grain and the combination of regions is called the Voronoi tessellation or Voronoi diagram.

The reasoning behind the use of the Voronoi tessellation is that it closely resembles the grain structure and grain growth of metals. Consider the crystallization process of a one-phase metal with random nucleation points (seed points). If all the grains start to grow simultaneously from the nucleation point and at the same rate, they will produce a microstructure that resembles a Voronoi tessellation [26]. For the material that is used in this paper, 100Cr6 or AISI 52100, the grains have a mean diameter of 10µm which is normally distributed and therefore the seed points are placed a mean of 10µm apart.To define the boundaries of the region of interest, the seed points are mirrored around boundaries of the region of interest. This causes the box to be defined by the Voronoi cells itself.

2.7.7. Grain Improvement

Since the Voronoi structure is randomly created, very small boundaries in between some of the grains occur. These small boundaries are not seen at first sight but when meshing the structure, they become visible since the mesh is very concentrated around those boundaries. To improve the structure, reduce the modelling time and create a more even mesh, these smaller boundaries are removed. A minimum distance requirement is set and the method looks for boundaries with a length below this set requirement. If the small boundary is found in the structure, the middle point of that boundary is used as the vertex of the surrounding grains. The other points are removed and thus the small boundary is also removed. When these small edges are found on the edge of the domain or adjacent to the edge of the domain, the vertices of that edge are moved such that they are at the minimum distance that is required. The removal of these small edges reduces the calculation time since there are less edge to retrieve data from.

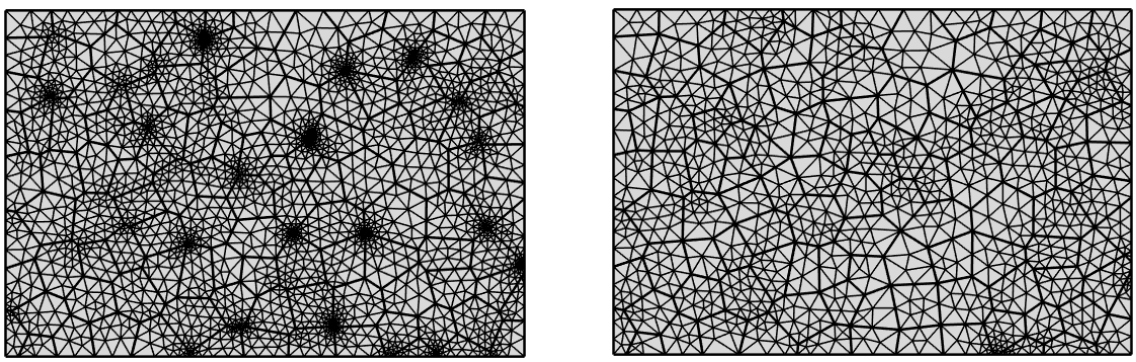


Figure 2.21: *Comparison between the unchanged grain structure (Left) and the refined grain structure (Right)*

2.8. Model Validation

In order to know if the created model gives results that are similar or equal to experimental data, the model has to be validated. The validation can be done in several ways and will be explained in this section. The methods that are used for the validation are found in several papers [27][34][56] but also some methods are created during this project.

2.8.1. Stress Distribution

The first and most basic form of validating the model is checking if the stress distribution of the principal stresses in the material closely follow the theoretical stress distributions calculated by the analytical Hertz contact formulas. The data is also compared to the experimental data of Chen et al. [7] who observed crack initiation and Raje et al. [51] who manipulated the original Lundberg-Palmgren theory such that the relative live N_f can be related to the critical stress quantity τ. Raje assumed that q and r are the same exponents as in the original Lundberg-Palmgren theory with $q = 10.33$ and $r = 2.33$. With this formula, relative life (N) can be given of the measured depth divided by the half-width $\frac{z}{b}$ to the maximum orthogonal shear stress reversal divided by the maximum pressure $\frac{\Delta\tau_{max}}{P_{max}}$.

$$N \sim \frac{z^r}{\tau^q} \tag{2.41}$$

To test the effect of the microstructure on the stress distribution, 40 different microstructures (Voronoi distributions) are used with similar dimensions. The same load and material parameters are used and the Hertzian pressure distribution is applied to the semi-infinite domain. This is done once with a static load and with a moving load to see if the locations and magnitudes of the principal stress do not change noticeably when moving the load. The data retrieved from the 40 different microstructures is evaluated and compared to the theoretical values and the experimental values. The comparison was done by checking if the average of the data was close to the theoretical values and if the values showed a comparable scatter to the scatter of the experimental data. This is done by comparing the range of the data sets and fitting the data in a Weibull plot using Eq. 2.41 and checking if the slope is in the range of 0.7-3.5 set by the experimental data from Harris [18].

2.8.2. Crack Formation

Another method to validate the workings of the model is to check at what depth the cracks initiate and how the cracks propagate through the material. From experimental data it is seen that the crack initiation occurs at the depth of maximum orthogonal stress. The following cracks nucleate parallel to the surface which has been noted from experimental observations and with further cycling these crack coalesce and form distinct trajectories. After a while these cracks will propagate to the surface and eventually reach the surface and create a spall [50]. This has been observed experimentally by using metallographic examinations of rolling contacts with the focus on the failed/spalled area. If the crack in the model is formed in the same way as it propagates in real life, it can be said that the model simulates reality fairly well.

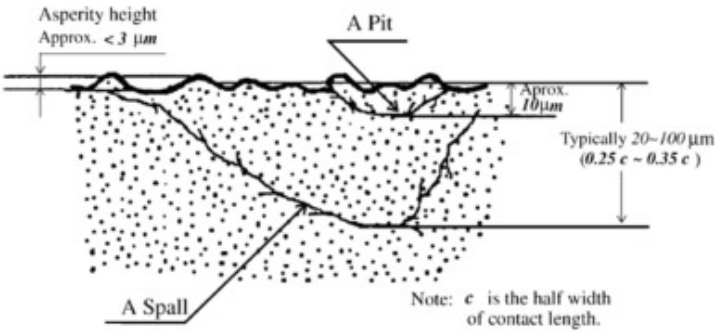


Figure 2.22: *Pit and spall formation and appearance [68]*

2.8.3. Experimental Data

The last and most important method for validating the model is to compare the initiation life and total life results from the numerical model to the experimental data created by Harris and Barnsby [18] and Harris and Kotzalas [19] but also to numerical models and the model created by Lundberg and Palmgren [35]. Firstly,

the Weibull slope of the created model is compared to the different types of data, the Harris and Barnsby dataset created a slope (β) with a range between $0.51 \leq \beta \leq 5.7$ while the Harris and Kotzalas range is between $0.7 \leq \beta \leq 3.5$. Since the range of the latter also fits in-between the former, the aim is to get the slope inside the $0.7 \leq \beta \leq 3.5$ range. Secondly, the L_{10} life is discussed because the L_{10} life needs to be in the same range as the numerical models, the Lundberg-Palmgren model and the experimental data. The L_{10} life is the amount of load cycles it takes for 90% of a group to survive and therefore 10% to fail. This is a common method to describe how long a bearing will probably last. Jalalahmadi et al. [27] compared the results from their model to the L_{10} life results from the data of Harris and Barnsby but also to the Lundberg-Palmgren model and the model created by Raje et al. [50].

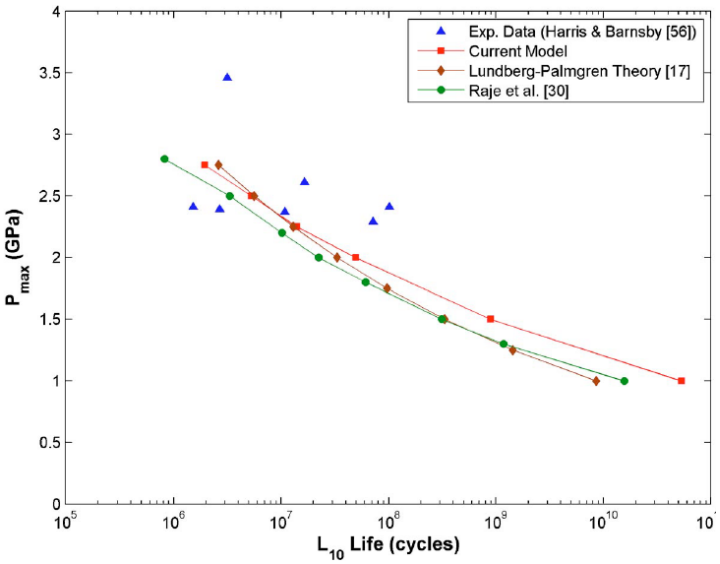


Figure 2.23: *The comparison of the P_{max} versus L_{10} curve obtained in the current model (Model of Jalalahmadi) and the existing results for AISI 52100 steel [27]*

The numerical model is run with 30 different microstructures and the initiation and total life data is fit into a 2-parameter Weibull plot. The scatter or Weibull slope (β) of the total life data has to be in the range set by the experimental data of Harris and Barnsby and Harris and Kotzalas while the initiation life can have a slope inside a range of 10-30. The validation of checking the slope of the initiation life is a suggestion made by Lorenz S. who has done a lot of research on the subject and is a research assistant of professor Sadeghi. This method reduces the initial validation time a lot because the model does not have to be run until failure. It is important to be in this range since this shows that the scatter from the model has a similar scatter as the experimental data therefore implying that the model comes close to reality. It is important to note that while the figure above (Fig. 2.23) shows that the presented models have an L_{10} life for a $P_{max} = 2.5GPa$ similar to the experimental data, it is unsure if this is still the case for lower values of P_{max}.

2.9. Scaling of Boundary Conditions in Rolling Contacts

As was discussed earlier, RCF is the main mode of failure of rolling contacts. The experimental testing of a high-cycle fatigue problem is a lengthy process and difficult to gather results from in real life even when testing happens at high speeds. The experimental tests look at crack development and surface behaviour while measuring the fatigue life of these contacts. Due to the duration of these tests a method to obtain realistic conditions during testing while also reducing the testing time and the cost of testing needs to be developed. This can be done by scaling certain parameters while keeping others the same, for example the scaling of the dimensions of the rolling contact and keeping the Hertzian pressure the same. The reduction in the dimensions provide the ability to do cheaper and faster testing by faster rotation or being able to create multiple test benches and do multiple experiments at the same time. The results from the scaled test bench should then be related to the real life tests is some way. This could be done by using a scaling factor or keeping the values of the results the same but it is not clear from literature what the correct procedure is. The subject of scaling parameters for rolling contact testing has been researched a lot and mainly on the

interaction of wheel-rail contacts. While this has been researched a lot, the focus was mainly on the dynamic interaction between wheel and rail, the stability and dynamics of vehicles and impact induced rolling contact fatigue [28][44]. In the paper of Naeimi et al. [44] a list is given of all the available test rigs for wheel rail contact studies and the scale of these models. In this section, the literature on scaling is presented but first the industry standard for bearing life is converted to get an idea what the effect of scaling might be on the fatigue life.

2.9.1. Scaling with ISO 281

A first method of calculating how scaling might influence the fatigue life is using the standards that already have been created for the industry, the ISO 281 standards. Following the ISO 281 standards, the basic bearing life is given by

$$L_{10} = \left(\frac{C_r}{P_r}\right)^p \tag{2.42}$$

where P_r is the dynamic equivalent radial load which is equal to the load applied and is related to the contact width in the following way.

$$P_{\max} = \frac{2F}{\pi bL} \tag{2.43}$$

with the half-width b calculated with the following equation and transformed to calculate the force.

$$E' = \frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2} \quad \& \quad R' = \frac{1}{R_1} + \frac{1}{R_2} \tag{2.44}$$

$$b = \sqrt{\frac{4FE'}{\pi LR'}} \quad \rightarrow \quad F = \frac{b^2\pi LR'}{4E'} \tag{2.45}$$

when placing the equation for F (Eq. 2.45) into the equation for P_{max} (Eq. 2.43) to following equation is formed.

$$P_{\max} = \frac{bR'}{2E'} \tag{2.46}$$

from the equations Eq. 2.43 and Eq. 2.46, for the same P_{max}, it can be seen that the half-width b is inversely proportional to R' and therefore proportional to the radius of the rollers and that $b \cdot L$ is proportional to F.

$$b \propto (R')^{-1} \tag{2.47}$$

$$b \propto R_1, R_2 \tag{2.48}$$

$$bL \propto F \tag{2.49}$$

C_r is the basic dynamic radial load rating and for radial roller bearings it is given as follows.

$$C_{\mathrm{r}} = b_{\mathrm{m}} f_{\mathrm{c}} \left(i L_{\mathrm{we}} \cos\alpha\right)^{7/9} Z^{3/4} D_{\mathrm{we}}^{29/27} \tag{2.50}$$

in which f_c a function is of $\frac{D_{we}cos\alpha}{D_{pw}}$. Here, D_{we} is the roller diameter, D_{pw} is the pitch diameter and α is the nominal contact angle. In C_r, b_m is the bearing type factor, i the number of rows of rolling elements, L_{we} is the effective roller length and Z the number of rolling elements. In this case, D_{we} and L_{we} both scale equally, $\alpha = 0$, and thus $P_r = F$ and from Eq. 2.48 it is known that the half-width is directly proportional to the roller radius. Knowing this, P_r and C_r can be expressed in the scaled parameters L and b.

$$\begin{aligned} C_r \propto L_{we}^{7/9} D_{we}^{29/27} &\rightarrow C_r \propto L^{7/9} b^{29/27} \\ P_r = F \quad &\rightarrow P_r \propto bL \end{aligned} \tag{2.51}$$

with these relations worked out, the L_{10} life can be worked out for the scaled parameters

$$L_{10} = \left(\frac{C_r}{P_r}\right)^{10/3} \rightarrow L_{10} \propto \left(\frac{L^{\frac{7}{9}} b^{\frac{29}{27}}}{Lb}\right)^{10/3} \rightarrow L_{10} \propto L^{\frac{-20}{21}} b^{\frac{20}{81}} \tag{2.52}$$

With this known, the ratio between the L_{10} lives between scaled rollers can be written as the following equation

$$\frac{L_{10,2}}{L_{10,1}} = \left(\frac{L_2}{L_1}\right)^{\frac{-20}{21}} \left(\frac{b_2}{b_1}\right)^{\frac{20}{81}} \tag{2.53}$$

From this equation, several things can be said about the fatigue life when the roller dimensions are scaled and the maximum pressure is kept the same. When both the length and the radii are scaled equally to a smaller size, the fatigue life of the smaller roller increases but when the length is kept the same and only the radius is decreased, the fatigue life decreases.

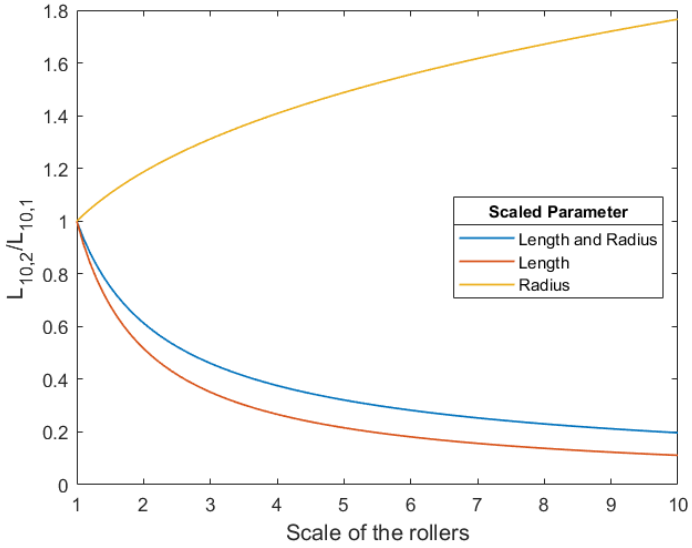


Figure 2.24: *Scaling of the roller dimensions and their effect on the L_{10} life based on the ISO 281 bearing life equation*

2.9.2. Scaling in Literature

In Ref. [55], RCF is tested experimentally with the aim of recreating real life conditions in the laboratory. The focus of the paper was on illustrating an experimental approach for rolling contact fatigue and shelling for simulating actual railroad conditions in the laboratory by scaling boundary conditions, thus providing a realistic evaluation of the contact fatigue life of railway steels. They started with the theory of similitude in which the same Hertzian pressure and materials are used in the lab as are used in real life. The size of the objects is reduced to a certain size in which the railroad wheel is the smaller wheel and the rail is simulated with a larger wheel. During testing, data on the contact area, crack density and crack length were collected. From the data it was concluded that applying the similitude design approach while applying a reduced scale of the actual contact surface gives a realistic picture of what occurs in the field. The cracks formed in their test bench, which were surface initiated cracks, followed the same path compared to the real crack (Fig. 2.25). Therefore they concluded that the similitude design approach worked out well for giving a realistic picture of the processes occurring in the field. It is also stated that the life of the model can be compared to the life of the actual wheel but that the number of cycles of the model wheel is three orders of magnitude lower compared to the actual wheel. Due to the lack of data on how the objects are scaled and how the fatigue life is influenced, it is unclear how the scaling of the contact area relates to the scaling of the load cycles.

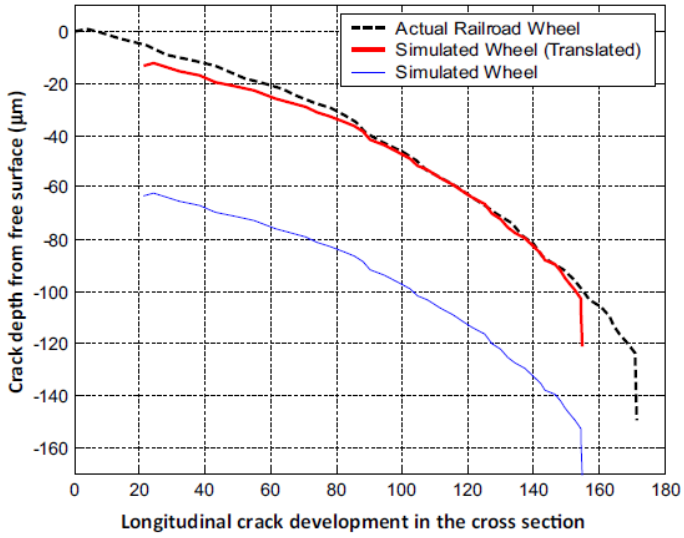


Figure 2.25: *Comparison of real crack pattern (depicted in Fig. 1) and crack pattern generated in the experiments (shown in Fig. 12); the y-axis represents the depth of the crack starting from the free surface, while the x-axis represent the longitudinal crack development of the crack starting from the initiation site[55].*

In Ref. [20], the focus was on developing a similarity theory for railway dynamics and focussed on small scale bogies. It is stated in the beginning that the advantages of small-scale test models is the high speed that can be maintained and that the time scale is reduced by the scaled amount and therefore everything occurs that amount faster. But other benefits were the increased safety and the cheaper and easier modification of the test samples. It is important to note that the method developed applies the similarity theory and thus the method is based on conservation of materials and stresses. The method developed is explained below with e being the scaling factor.
For the dimensions

$$\text{Length} : l = eL$$
$$\text{Surface} : s = e^2 S$$
$$\text{Volume} : v = e^3 V$$

Since the material stays the same, the density will also be equal therefore

$$\text{Mass} : m = e^3 M \tag{2.54}$$

And since the similitude approach is applied, the stresses applied are kept the same and thus σ = f/s = F/S

$$\text{Force} : f = e^2 F \tag{2.55}$$

The last parameter to be defined is time and appears in the Newton law through acceleration.

$$\frac{f}{F} = e^2 = \frac{m * a}{M * A} = \frac{e^3 * M * a}{M * A} \Rightarrow a = e^{-1} A \tag{2.56}$$

thus working further on this gives

$$\frac{A}{a} = e = \frac{L/T^2}{eL/t^2} = \frac{t^2}{eT^2} \Rightarrow t = eT \tag{2.57}$$

The paper states that the time scale is reduced and therefore everything 'ages' four times faster which would be very interesting. This does not take any material parameters into account and therefore it seems unreasonable to state that material degradation will occur 4 times faster.

In Ref. [28] (pg. 41), an exact representation of the contact area and its elasticity is studied in which the stress is also kept the same during scaling. This paper focusses more on the dynamics and not on the wear and fatigue life but is still interesting for its focus on the contact area and because it is applies the same scaling method that is applied in this project. In the paper, like the other two discussed before, the time also scales with the length and the derivation is described in the paper on page 41.

The papers of Heliot [20] and Sciammarella [55] state that reducing the size of the rollers while maintaining the stress and material properties, reduces the amount of load cycles for fatigue or that the object age quicker. While the approach is very different in both papers, one approaches the problem experimentally while the other theoretically, the reasoning and information is scarce and is not validated in other papers. The derivation where the ISO standard is used also states a reduction is fatigue life but only for the reduction of radius and it states an increase in fatigue life when both length and radius is scaled. Since the Roller Test Bench of DOT also follows these principles, it is important to validate this method of similitude by numerical modelling. Due to the lack of information on the influence of scaling on rolling contact fatigue life and the influence it can have on roller test benches, this becomes an interesting topic of research. Fatigue and certainly rolling contact fatigue are complex phenomenon that have statistical characteristics due to the heterogeneous material structure, therefore this problem needs to be answered with a closer look on the material structure. While the material structure does not change, other parameter like critical stress depth and amount of affected grains changes.

3

Building the Model

In this chapter, the building steps and assumptions made to build the model are talked about in detail. From creating the structure in Matlab to simulating in COMSOL and analysing back in MATLAB, every step with the code included in the appendix is discussed and the full code can be found in appendix F.

3.1. Microstructure Topology Model

As was discussed earlier, the accurate representation of the microstructure is important in the investigation of rolling contact fatigue since it adds the effect of fatigue life scatter, fatigue damage initiation and propagation [3]. The Voronoi tessellation is applied in this model since it has been proven to represent the microstructure of 100Cr6 well. In this section, a detailed explanation of all the steps taken to build the structure is given.

3.1.1. Point Selection

In order to create the Voronoi tessellation that represents the microstructure, seed points have to be created that are placed at a certain distance from each other. For the material used in this paper, 100Cr6, the grains have a mean diameter of 10µm with a standard deviation of 2.5 which is represented with a normal distribution. Therefore the seed points have to be at a distance with mean 10µm which is normally distributed. The size of the domain with the microstructure is $3 \cdot b$ wide by $1.5 \cdot b$ high with b being the Hertzian contact half-width. The Matlab code used to create the seed points is found in appendix E.1.1.

Firstly, the domain of interest is filled with points and from these points a seed point is chosen and saved for later use. A distance requirement is created with the normal random number generator *normrnd* and the distance between the seed point and the points around it is calculated. If the distance between a seed point and a point is below the distance requirement, the point is removed. After inspecting all the points around the seed point, a new seed point is chosen and the whole process is repeated until all the remaining points are seed points and no points can be removed anymore.

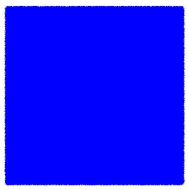


Figure 3.1: *Domain full of points*

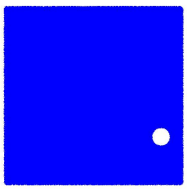


Figure 3.2: *Domain with one seed point*

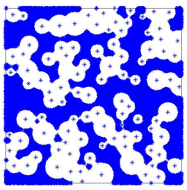


Figure 3.3: *Domain with 100 seed points*

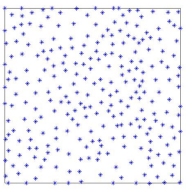


Figure 3.4: *Domain with only seed points*

The Voronoi tessellation is created with Matlab command *voronoin* which returns the Voronoi vertices (v) and the Voronoi vertex combination for each cell (c) for all the seed points. The created structure has unbound cells at the outer edges of the specified domain which have to be bound to create a structure with

surfaces that can be constrained and have the pressure distribution applied to. This is simply done by mirroring all the seed points in the domain about the edges of the domain as seen below. It can be seen that with the use of *voronoi* the mirrored points places boundaries along the required edges of the domain.

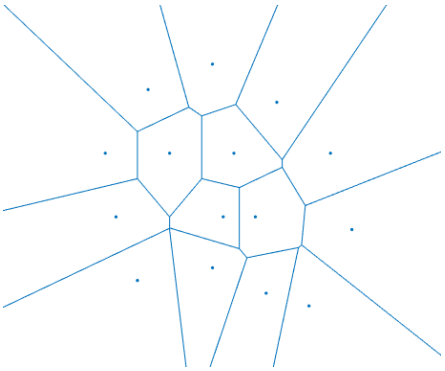


Figure 3.5: *Voronoi tessellation without a mirror about the domain edges*

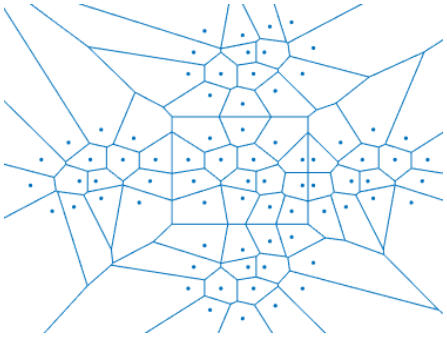


Figure 3.6: *Voronoi tessellation with a mirror about the domain edges*

3.1.2. Removing Small Edges

After creating the Voronoi tessellation in Matlab, it seems at first that all the grains have normally sized edges but when zooming into some of the grain corners, it can be seen that several very small edges are created. If this would be implemented in COMSOL, it would cause mesh concentrations around these points and therefore a method is created to remove the very small edges. This reduces the computational effort and creates a more uniform mesh [45]. The Matlab code for this edge removing method can be found in appendix E.1.3.

3.1.3. Random Micro Structure

Every time a new simulation is started, a new microstructure is created and since the seed points are placed differently to each other, a different Voronoi tessellation is created. But the structures do have similarities since they follow a normal distribution for the diameter of the grain. As seen in the figures below, the edges per cell on the left and the area of each cell on the right form a normal distribution with most of the cells having five, six or seven edges which was also found in other research papers [25] [27].

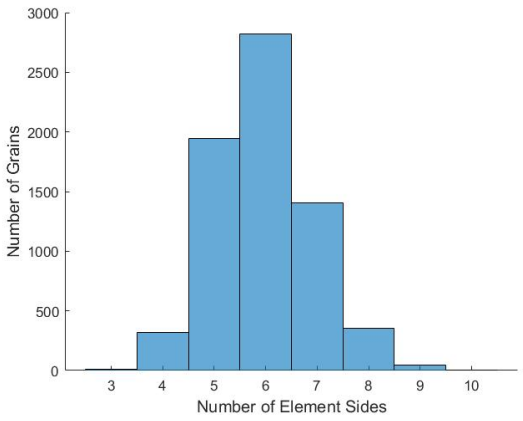


Figure 3.7: *Histogram of the number of sides created per Voronoi cell created in a domain*

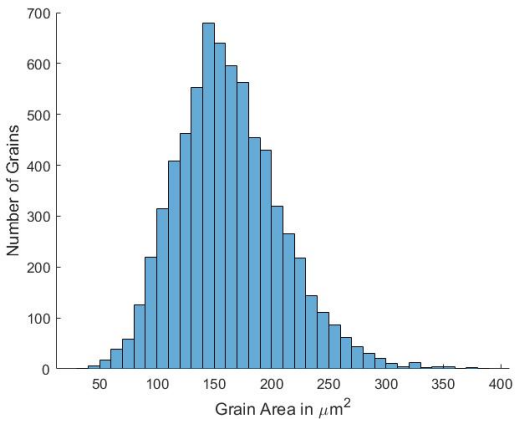


Figure 3.8: *Histogram of the areas of all the Voronoi cells created in a domain*

3.2. COMSOL Model

In this model, COMSOL is used as the FEM software while Matlab is used for most of the calculations. Matlab and COMSOL can be used together with the use of COMSOL Livelink such that Matlab is able to steer COMSOL and the results of COMSOL can be easily transferred to Matlab. Through COMSOL Livelink, Matlab is able to create a COMSOL model such that each time a similar model can be created and used. In the model, the majority of the work is done in Matlab but COMSOL is used for its finite element capabilities and is just a vehicle that delivers the information of the state of stress in the material.

3.2.1. Create Model

While a detailed user guide for Livelink is made [43], it can still be quite challenging and overwhelming to start building a model. There are several methods that can be used that make creating a model in Matlab quite a bit easier. With the first method, a model can be build in COMSOL at first and saved as Matlab code but this can be quite a large file and it does not aid in understanding the capabilities of Livelink or what some parts of the code do. The second method is more focussed on understanding what each part of the code does. After building the model in COMSOL, when right clicking on a part in the *Model Builder*, the option to *Copy as Code to Clipboard* presents itself. This gives the option to copy certain actions such that it becomes clear what each part does.

When building the model, it is important to start of with *import com.comsol.model.** and *import com.comsol.model.util** since these import the COMSOL class. After doing this, the model can be created by using *ModelUtil.create('Modelname')* and after this, features can be adjusted and added depending on the model you are creating. In this project, a 2D model is built of which the code can be seen in appendix E.2.1. After that, the parameters that are important for the COMSOL model are added which can be seen in appendix E.2.2.

3.2.2. Create Geometry

The microstructure is created cell by cell by placing the vertices (v) of each cell in the *Polygon* geometry tool of COMSOL. In this project, this seemed to be the easiest method of placing the in Matlab created geometry into COMSOL. Only the cells in the bounded domain are chosen to create a proper structure since these boundaries can be constrained. A *Union* is used on the grains such that a single geometry is created while still keeping the interior boundaries. The code on how this was done can be found in appendix E.2.3.

After creating the microstructure, the surrounding geometry has to be created which is a box that does not contain any grains. If the whole domain would be made out of Voronoi cells and would be analysed, the time to compute the results would take a long time since every grain edge is analysed and the computational time increases with an increase in the amount of edges. The surrounding box is $10 \cdot b$ wide by $6 \cdot b$ high with b being the Hertzian contact half-width (Fig. 3.10). The *Difference* geometry tool in COMSOL is used on the microstructure and the box to account for small imperfections created in Matlab and a *Union* is used on the microstructure and the box to create a single geometry.

The size of the domain dimensions is chosen based on when the principal stresses are not influenced by the constrained edges. Meaning that shorter dimensions had an influence on the stress distribution in the region of interest but with larger dimensions, the constrained edges did not have any influence anymore. This can be seen in the figure on the left below where the stress increases again around the outer edges compared to the figure on the right where the stresses do not show such an increase.

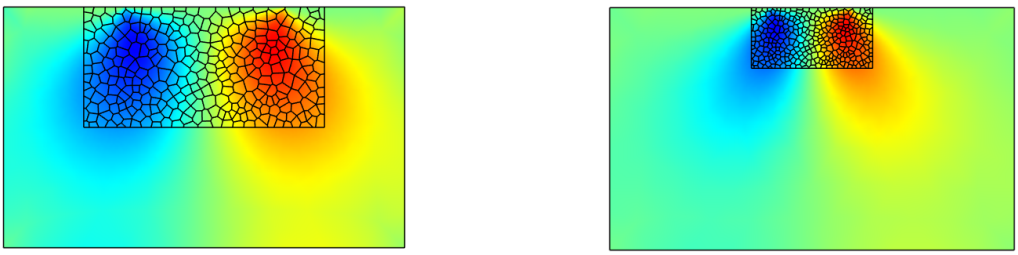


Figure 3.9: *The left model has a width of* $5 \cdot b$ *and a height of* $3 \cdot b$ *and a* $\Delta\tau_{max} = 0.4923 \cdot P_{max}$. *The right model has a width of* $10 \cdot b$ *and a height of* $6 \cdot b$ *and a* $\Delta\tau_{max} = 0.4985 \cdot P_{max}$.

3.2.3. Explicit Grain Selection

While the vertices assigned to each grain is known in Matlab, COMSOL has its own way of numbering the grains and edges in the structure. Therefore, a way of assigning edges to grains in COMSOL has to be found such that later on, the retrieved data can be used for each individual grain. In this paper, an *Explicit Selection* in the *Definitions* node is created for each grain. A *Domain* selection is made as the *Input Entity* but the output of the selection is set to *Adjacent Boundaries* and this is set to *Exterior Boundaries* which will output the boundary numbers of a certain grain. The code for this can be found in appendix E.2.5 but it is important to note that these lines of code were not found with the Copy to Clipboard method but by converting the whole model to Matlab code.

3.2.4. Boundary and RVE Selection

For the same reason as in the previous section, a selection has to be made of the boundaries in COMSOL due to its way of numbering elements. A box selection is created of the four boundaries of which three will be constrained and one will have a load applied to it. Another selection is made of the top boundary of the microstructure which will be used to stop the model once a crack reaches the surface. This selection is different to the other selections since this selection also includes adjacent edges to the boundary while the other selections are focussed on just the outer boundary. Another difference between the crack selection and the boundary selection is that the crack selection is created in the *Definitions* part while the other selections are made in the *Geometry*. The difference between the *Definitions* selection and the *Geometry* selection is that values can be retrieved to Matlab from the *Definitions* selection while this is not possible with the *Geometry* selection. The use for this will become clear when using the software and later in the model. The code for this can be seen in appendix E.2.6.

Only a small part of the domain has the material microstructure and in this domain, a subregion is created called the Representative Volume Element (RVE). The RVE is the region of focus where the material is critically stressed and this is used to reduce the computation time. The reason that the RVE is used, is to focus on the stressed area and to not have an abrupt stop of the grain structure. The Voronoi tessellation domain has a width of $3 \cdot b$ and a height of $1.5 \cdot b$ while the RVE has a width of $2 \cdot b$ and a height of $1 \cdot b$ (Fig. 3.10) which is large enough since the depth of maximum shear stress reversal is around $0.5 \cdot b$. To create this domain, a selection in the *Definitions* section of the boundaries and the grains is made such that the values can later be used to speed up the model. The code can be found in appendix E.2.7.

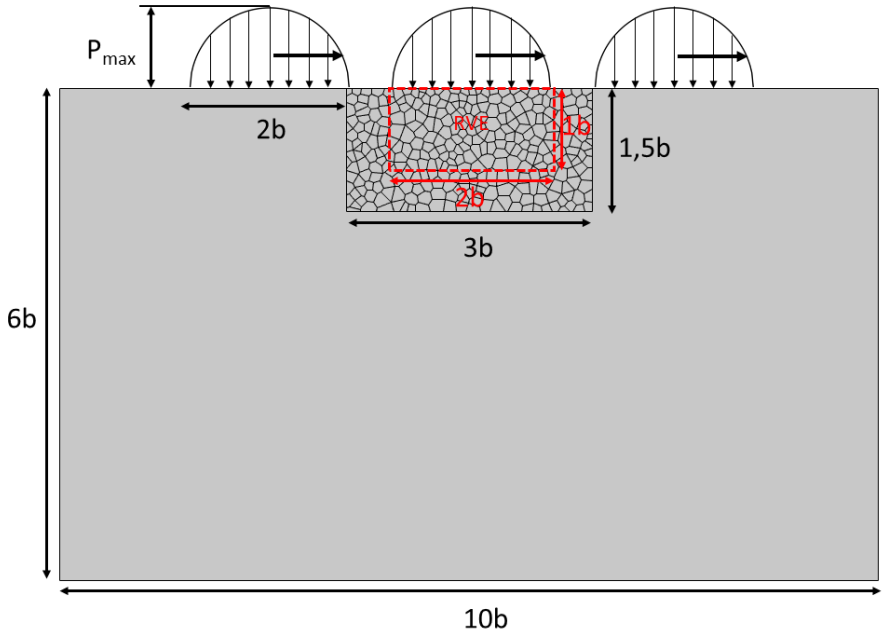


Figure 3.10: *Computational domain with Voronoi tessellation*

3.2.5. Stress Distribution

In the *Definitions* section, an analytic function is created to create the Hertzian stress distribution that will be applied to the surface (Eq. 2.3). The function goes from $-b$ to b and to make the function work in COMSOL, two if statements are used to make sure that the values in the square root never become negative. Later on, while creating the model, it was noticed that this was not sufficient because COMSOL had trouble solving this function and therefore the absolute value of the values under the square root was taken. The code can be found in appendix E.2.8.

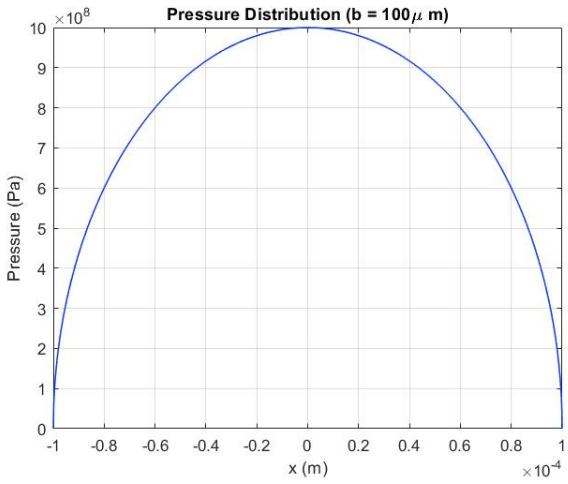


Figure 3.11: *Hertzian pressure distribution created in COMSOL*

3.2.6. Physics and Material Properties

In the current model, the *Solid Mechanics* physics that have to be added and an important change has to be made to the *Linear Elastic Material* node. Since Lamé parameters are used, the node with *Young's modulus and Poisson's ratio* has to be changed to *Lamé parameters*. This can be seen in appendix E.2.9. After adding the physics and changing to Lamé parameters, each grain gets its own material properties assigned to it such that during the damage calculations, each grain can degrade at its own rate. While adding the density to each grain is fairly simple, it requires a couple more steps to add the Lamé parameters with Livelink. Firstly, the Lamé property group has to be created and after that the Lamé parameters can be added. This is shown in more detail in appendix E.2.10.

3.2.7. Constraints and Load

After adding the physics and creating the stress distribution, the boundaries can be constrained and a load can be applied to the top boundary or contact surface. The bottom edge and two side edges of this domain are constrained and a Hertzian pressure distribution (Eq. 2.3) is placed on the top edge. The function that was created earlier is added as a boundary load and a parameter is added to the function such that the load can be moved over the surface. This added parameter will later be used again in the auxiliary sweep. The boundary selections that were created earlier are used in this part to quickly select the correct boundaries. All the details can be found in appendix E.2.11.

3.2.8. Meshing and Study

After adding all the constraints and the load, the structure is meshed with a simple COMSOL created mesh (Appendix E.2.12). After that, the study is created and a *Stationary* study is chosen with an *Auxiliary Sweep* and is used to move the load over the surface. A load cycle is simulated by moving the Hertzian pressure distribution over the semi-infinite domain with the middle of the load starting at $-2 \cdot b$ and going to $2 \cdot b$ with 40 steps. This causes the RVE to go through a full shear stress reversal.

3.3. CDM Model

In this section, the details on retrieving the data but also implementing material degradation and crack formation are discussed. An explanation of the formulas that are used and the reasoning behind certain decisions is discussed in section 2.7.

3.3.1. Start Model

To start of the model, a while loop is created that will stop the model once a crack reaches the surface. For this while loop, the earlier created crack selection of the edges adjacent to the top boundary is used. After the while loop, the study is run and therefore COMSOL calculates the stresses in the material. These two lines of code can be found in appendix E.3.2.

3.3.2. Line Average

The RCF crack initiation and propagation are restricted to the grain boundaries in this study which has been proposed by many researchers [27][34][64] and therefore the stress along these grain boundaries has to be found and calculated. The resolved shear stress is averaged at each grain boundary to remove the mesh dependency of the simulations and these averaged values are used to characterize the grain boundary response during the load cycle [64].

The first time the while loop is ran, a *Line Average* is created for every edge in the RVE. The *Line Average* is used to calculate the plane shear stress along the edges of the grains in the RVE. It seemed at first that COMSOL variable *solid.stt* was the correct stress to use but the problem with this variable is that it never becomes negative. Therefore the equation for plane shear (Eq. 2.28) has to be calculated in COMSOL such that the correct shear values can be retrieved in Matlab. The formula applied in COMSOL is *-0.5*(solid.sx-solid.sy)*sin(2*atan2(nY,nX))+solid.sxy*cos(2*atan2(nY,nX))* where *nX* and *nY* are the normal direction coordinates of the edge which are used to calculate the angle of the normal of the edge to the x-axis. While there are several methods of retrieving the plane shear stress into Matlab, the fastest way is to create an *Average Line* evaluation in the *Derived Values* section of COMSOL. The code to create these evaluations can be found in appendix E.3.3.

3.3.3. Retrieve Data

The retrieval of data from COMSOL to Matlab is the most time consuming part of the whole model. For each edge in the RVE, the plane shear stress of each step is calculated and Fig. 3.12 is an ideal representation of what the stress profile looks like during a load cycle. This will differ from edge to edge since each edge has a different orientation but for each edge the maximum and minimum shear stress is used to calculate the total shear stress reversal. There are several ways of getting the data to Matlab but the method seen in appendix E.3.4 and E.3.3 were the fastest methods found in this project.

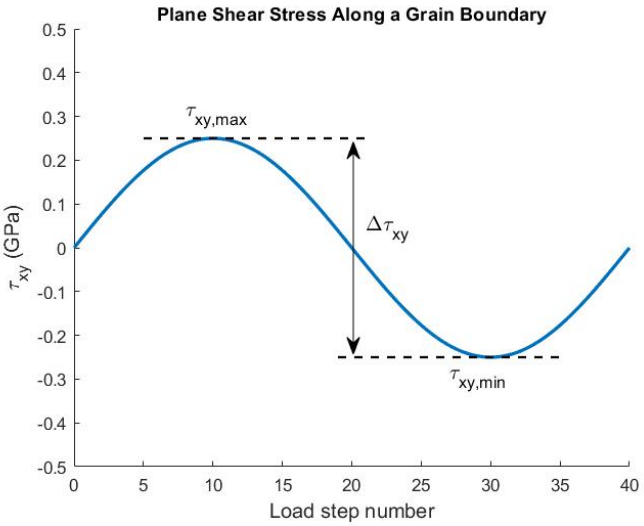


Figure 3.12: *Plane shear stress reversal along a grain boundary*

3.3.4. Damage Rate

After calculating the shear stress reversal of each edge in the RVE, the damage evolution rate is calculated with the formula (Eq. 2.27) found in section 2.7.1. After the calculation of the damage evolution, the jump-in-cycle method is applied as seen in section 2.7.3 to calculate the cycles and the damage that each edge incurs. As discussed in the previous chapter, to reduce the calculation times, the jump-in-cycle method is implemented in the Matlab code and to keep the amount of jumps low and the calculations fast but the accuracy high, $\Delta D = 0.2$ is chosen for each block of cycles. This is a simple Matlab code and does not include any Livelink coding and can be found in appendix E.3.5.

3.3.5. Crack Creation

A boundary fails once the damage reaches one which means that a crack occurs. When a boundary is fully damaged, the damage in that edge is set to Not a Number (NaN). NaN is a member used in Matlab to represent missing or in this case removed values. In this case it is used to keep the order of the dataset intact and therefore the model easier to use but it also makes it easier to keep these members out of the following calculations. The failed boundaries are not used in further damage evolution calculations but are still used in the grain damage calculation which will be discussed in the following section.

A crack can be formed in several ways in COMSOL but for this model the *From Geometry* node in the *Crack Surface* section of the *Crack* node is selected. This causes the single boundary to be split in two crack faces which are represented by different boundaries. This is important since a friction component has to be added between the crack faces. This is done by creating a *Contact Pair* of the crack faces and adding a *Contact* node in the physics interface. In the *Contact* node, a *Friction* sub node can be added in which the friction coefficient of 0.4 can be added. The *Crack* and *Contact* node are added once when the first boundary fails while a new *Contact Pair* is added each time a boundary fails. The whole process is repeated such that cracks continue to form in the structure and coalesce to form larger cracks. The process is stopped when a crack reaches the surface and thus failure is assumed. The detailed code can be found in appendix E.3.6.

3.3.6. Damage of Grain

After the calculation of the damage of a grain edge, the damage of each grain is calculated. Because COMSOL numbers the edges and grains in a different way compared to how the grains are introduced in the model from Matlab, the explicit selection that was created earlier is used to find the edges that belong to each grain. The damage of each grain is calculated by averaging the damage of the boundaries of the grain and this damage is then applied to the material parameters of that grain with Eq. 2.39 and 2.40. These values are then fed back to COMSOL and the process can be started all over again.

For the grain damage calculations, Matlab converts the NaN values of the boundaries to ones while still keeping the NaN for the other calculations. When all the boundaries of the grain have failed, the damage of the grain becomes one which means that the Lamé parameters become zero. In this model, the Lamé parameters are set to one when this happens since setting these to zero would cause calculation problems in the model. The code can be found in appendix E.3.7. The evolution of the damage and the material degradation of a grain and the increase in load cycles over the simulation time can be seen below.

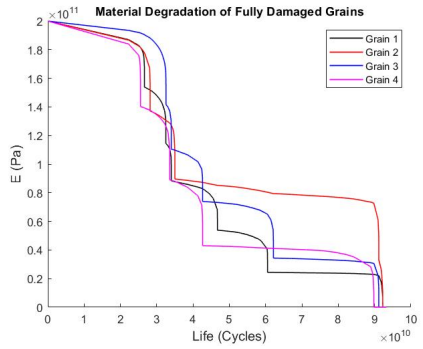


Figure 3.13: *Material degradation of several grains that were fully damaged when failure occurred*

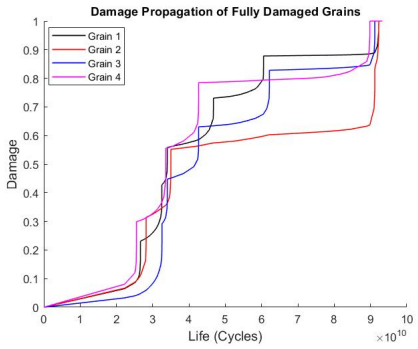


Figure 3.14: *Damage propagation of several grains that were fully damaged when failure occurred*

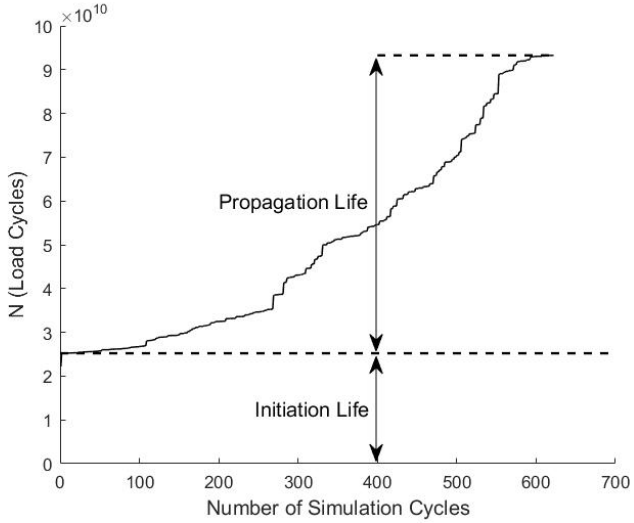


Figure 3.15: *Load cycle progression compared to the amount of simulation cycles*

3.4. Parameters

Material parameters	
Parameter	Value
Mean grain diameter (μm)	10
Grain diameter standard deviation	2.5
Elastic Modulus (GPa)	200
Poisson's Ratio	0.3
Resistance Stress (σ_r) (GPa)	6.113
m	10.1

Table 3.1: *Material parameters used in the model*

Validation parameters	
Parameter	Value
Maximum pressure (P_{max})(GPa)	1 GPa
Half-width (b)(μm)	100
Crack Friction Coefficient	0.4
Damage evolution (ΔD)	0.2

Table 3.2: *Validation parameters used in during validation*

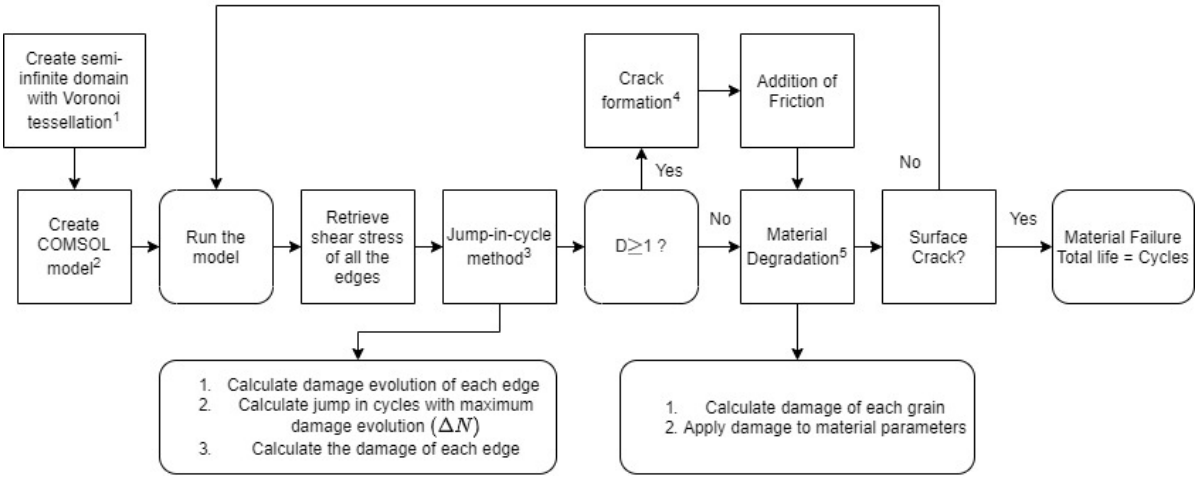


Figure 3.16: *Development of the model. A thorough explanation of these methods can be found in the following sections. 1) Sections 2.7.6 and 3.1 2) Section 3.2 3) Section 2.7.3 4) Sections 2.7.5 and 3.2 5) Section 2.7.4*

3.5. Factors taken into Account

In this model several factor are assumed to be able to create a model in a reasonable time and to reduce the computational effort but also some choices were made to adapt to the capabilities of the software that is used. Some of these can be explained fairly easily or where choices from previous researchers while others require some testing and reasoning behind the choices made. While in the previous sections most of these things are already discussed, this section will summarize all of the decisions made. First the general assumptions are discussed and after that the similarities and the differences to other models will be explained.

3.5.1. General Assumptions

While building this model, it was assumed that the material properties were homogeneous and that the material behaves isotropically. It was also assumed that the surface was smooth, that there were no initial material defects and that there was no wear and no traction. Inhomogeneous material properties, anisotropy, surface roughness and the different types of failure require further development and complication of the model with a lengthy validation process and therefore these were excluded from this study. Since the surface experiences a negligible amount of friction it is fair to assume to keep it out of the model. Surface roughness is an important factor to develop if a follow up project is done with the method and model developed in this paper since surface roughness, just like grain size, does not change with scaling and causes peak pressures in the material. The surface roughness was implemented in the paper of Lorenz [34] and this research can be the basis for further development. The application of anisotropy was discussed by Paulson [47] and Vijay [64] in a lot of detail while the implementation of inhomogeneous material properties and material defects is discussed by several researchers [26][27][50][51][52][56]. The heterogeneous material properties assume a normal distribution with the mean being the Young's Modulus ($E = 200GPa$) with a standard deviation of $20GPa$.

3.5.2. Similarities

In almost all papers that apply this method and therefore also this one, assume that the cracks form along the grain boundaries (intergranular failure). These papers also assume that a friction coefficient of 0.4 has to be applied between the crack faces which has been researched by Jalalahmadi et al. [27]. Since the FEM software that the other papers used (Abaqus) was different to what is used in this paper (COMSOL), it was unclear how a crack would behave in COMSOL. Therefore, the model was first ran without the friction and a slit was introduced in COMSOL and after that, the model with friction along the crack faces was ran. It can be seen in the figure below 3.17 that the frictionless model has no significant increase in fatigue life while the one with friction shows a more desirable behaviour. This was also noted in the paper of Jalalahmadi et al. [27] and therefore the friction was applied in the model.

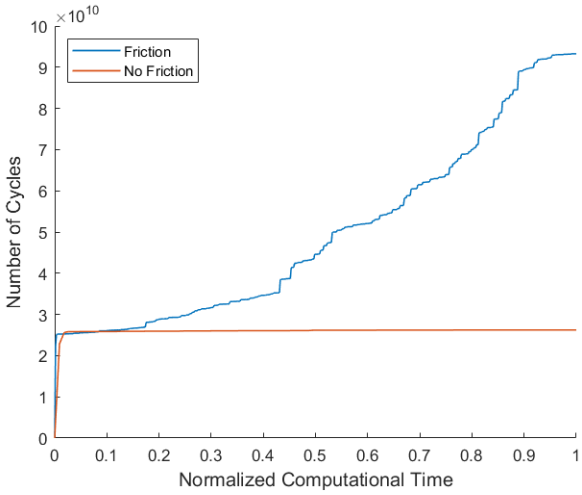


Figure 3.17: *The difference between the friction model and the frictionless model for $b = 100\mu m$ and $P_{max} = 1GPa$*

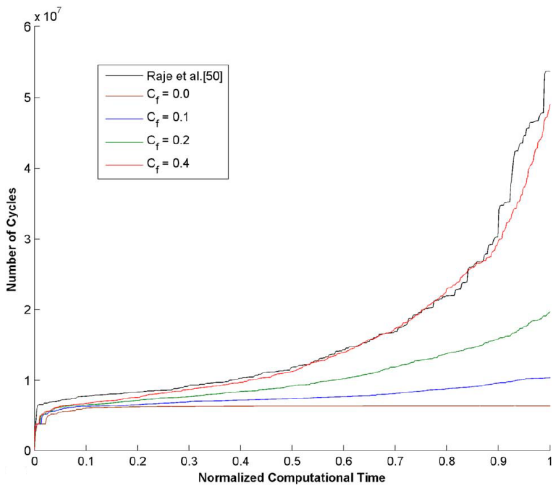


Figure 3.18: *Comparison of the life curves with different friction parameters for $b = 100\mu m$ and $P_{max} = 2GPa$ [27]*

To reduce the computational effort of the model, several decisions were made and techniques were implemented of which the most important ones were the jump-in-cycle method and the RVE [34]. The jump-in-cycle method makes sure that not every cycle has to be modelled and calculated, reducing the time it takes to get to failure. ΔD was chosen to be 0.2 and this decision is based on several papers that researched the effect of different ΔD values [27][52]. It was shown that a smaller ΔD did not improve the final result and only increased the time to get a result and that a larger ΔD greatly reduced the accuracy. The RVE reduced the computational effort since less edges and therefore less data had to be transferred from COMSOL to Matlab. Another factor that had influence on the computational effort is the removal of the small edges because this again reduces the amount of edges in the model but also improved the mesh.

3.5.3. Differences

Other factors were implemented to make the model work in COMSOL because each FEM software has different capabilities and therefore the calculation of damage and which elements were affected was different. Some models are able to change the properties of individual mesh elements but COMSOL does not have such capabilities which means that in this model, a whole grain will degrade. Other models are able to focus the damage to the grain boundary by creating elements along this grain boundary [64]. Older models [27] applied a centroidal mesh which was also not possible in COMSOL and for that reason a fine mesh was created by COMSOL. Due to the previous two 'short comings' of COMSOL, the material degradation was done through individual grain degradation. The grain damage was calculated with the damage from the grain boundaries which in turn was calculated from the shear stress along these grain boundaries which was averaged to remove the mesh dependency. After a boundary was fully damaged, some models set the damage of that boundary to zero or to one but in this model the value was set to NaN. This was done to exclude the boundary for the damage evolution equation and to easily change the value to one when calculating the damage of the grain. If the damage was set to zero or kept at NaN, it would have an adverse effect on the material degradation.

4

Testing

4.1. Model Validation

While this model applies the same principals and methods that are used in the models created by Sadeghi and his research group, the difference in modelling software means that certain adaptations had to be made. The different software and the applied adaptations means that this model will behave differently to other similar models and therefore it has to be validated to make sure that it simulates reality or comes close to reality. For the validation process, the same parameters are used as in the paper of Lorenz [34] who used a $P_{max} = 1GPa$ and a $b = 100\mu m$. The validation is done in several ways to make sure that the most important factors are done correctly.

4.1.1. Stress Distribution

A Hertzian pressure distribution was applied to the contact surface and moved over the contact surface in 40 steps and for one test the load is placed in the middle of the microstructure and the principal stresses are measured. For the stationary case, the stress is measured along the centerline which lays in the middle of the pressure distribution. From the 40 tested domains, the maximum shear stress reversal and the depth of each step is measured and for each test all these 40 step values are averaged and plotted in the figure below. The centerline stresses of the contact for one domain is measured and these should follow the theory fairly close since the material properties are assumed to be homogeneous and the material is assumed to behave isotropically. After that the relative life is calculated with Eq. 2.41 and placed in a Weibull plot to find the slope of these points.

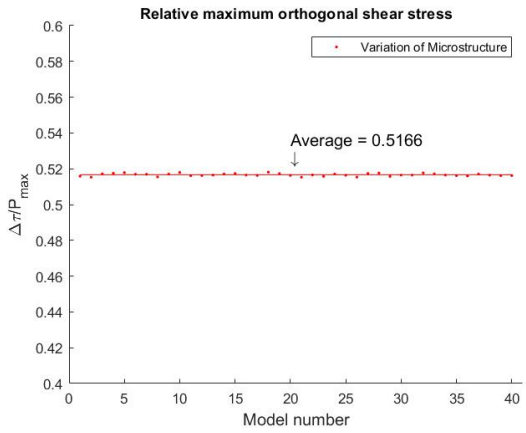


Figure 4.1: *Average shear stress reversals for each loadcycle, the values vary more during the loadcycle*

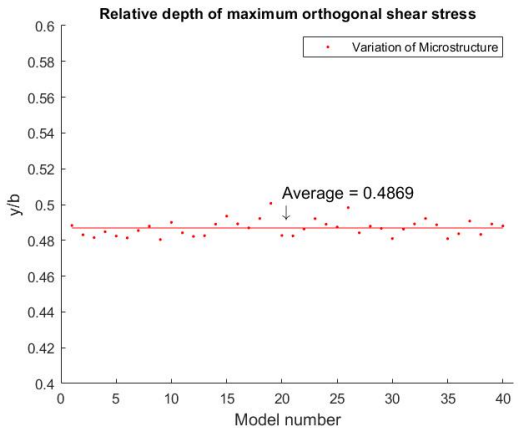


Figure 4.2: *Relative depth of the maximum orthogonal shear stress*

In Fig. 4.1, the average values for the 40 test for the maximum orthogonal shear stress are plotted relative to the maximum pressure. The stress comes very close to the theory which is $\Delta\tau_{max} = 0.512 \cdot P_{max}$ and over the 40 domains during the load cycle the values ranged from $0.496 \cdot P_{max}$ to $0.592 \cdot P_{max}$ with an average of $0.517 \cdot P_{max}$. In Fig. 4.2, the average depth over the whole load cycle of these stresses relative to the half-width can be seen. The depth of the maximum orthogonal shear stress comes close to the theory which is $z = 0.5 \cdot b$ and over the 40 domains during the load cycle it ranges from $0.480 \cdot b$ to $0.501 \cdot b$ with an average of $0.487 \cdot b$. Chen at al. observed for the depth of the critical stress, where the cracks initiate, a range of $0.33 \leq (y/b) \leq 0.64$ with an average of 0.522 and Raje et al. obtained a similar range of $0.36 \leq (y/b) \leq 0.66$ with an average of 0.5. For Raje et al. the stresses ranged from $0.492 \leq \Delta\tau / P_{max} \leq 0.524$ with an average of 0.5025. When comparing the values obtained in this model, it can be seen that the range of the depth is smaller compared to Raje and Chen but that the range of the stress is similar.

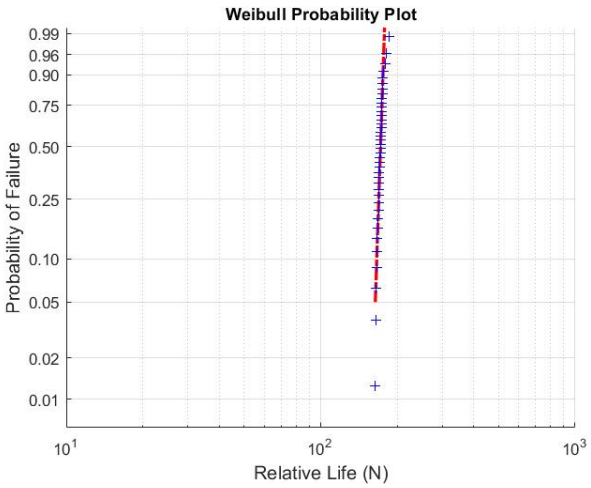


Figure 4.3: *Weibull plot of the relative life created with Eq. 2.41*

The relative life plot is created with the formula from Raje and shows a Weibull plot with a slope of 33.7 while experimental data resulted in a range for the slope between 0.7 and 3.5 and the slope found in the paper of Raje is 3.36. Since the slope is high, it means that the scatter is low which is to be expected with an homogeneous isotropic material and can also be attributed to the improvement of modelling techniques and software over the years. The scatter can be attributed to the size of the mesh and will vary with the shape of the microstructure. The larger the mesh size, the larger the scatter in these values will be but it is more desirable to get more accurate values and therefore the grains are finely meshed.

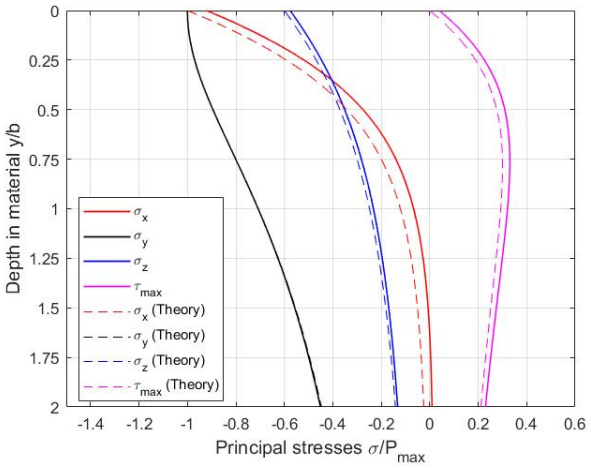


Figure 4.4: *Principal centerline stresses in the material compared to the theory*

It can be seen in the principal stress plots that the centerline stress in the material follows the theory very well, with σ_y following the theory perfectly. From all these results it can be noted that the pressure is applied properly to the material and because of that, the principal stresses follow the theory closely. The change in microstructure does not change the behaviour of the critical stresses because of the assumption of an homogeneous isotropic material but the values are in the correct range. Since a fine mesh is applied to the material, the scatter in the critical stress values is low and therefore the created slope from the formula of Raje is far from the experimental data.

4.1.2. Crack Formation

The figures shown below, show the crack formation and propagation until failure occurs for a surface with a Hertzian pressure distribution applied to it with $P_{max} = 1GPa$. The figures (Fig. 4.5 and 4.9) show that the crack initiates in the region of maximum shear stress reversal which Lundberg and Palmgren assumed and has been proven by several experimental observations like the one from Chen at al. [7]. In Fig. 4.9 it can also be seen that the scatter in the depths of crack initiation comes closer to the experimental observations of Chen et al. [7] than the values for the depth found in the previous section. The cracks will continue to form and as seen in Fig. 4.6, these cracks will mainly be oriented parallel to the surface which is also observed in experiments [46][50][69]. These cracks start to coalesce as seen in Fig. 4.7 and at some point these cracks will propagate to the surface and form a spall which can be seen in 4.8. The formation of cracks and the depth of crack initiation is consistent with experimental observations which means that the model is one step closer to reality.

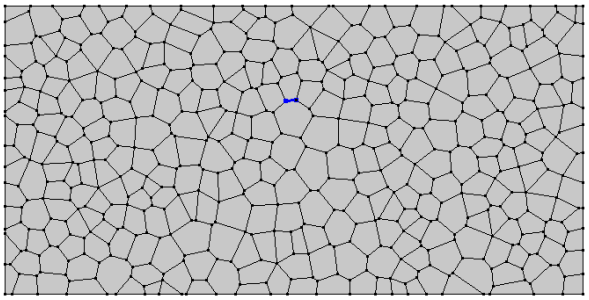


Figure 4.5: *Crack initiation after* $2.6193 \cdot 10^{10}$ *cycles*

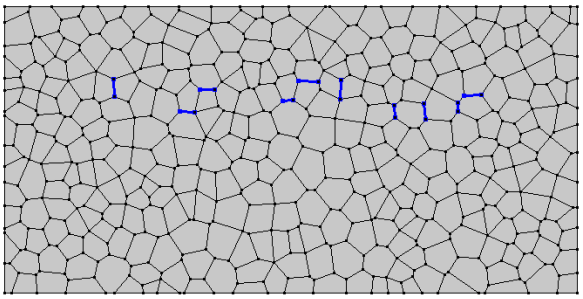


Figure 4.6: *Crack formation after* $2.6318 \cdot 10^{10}$ *cycles*

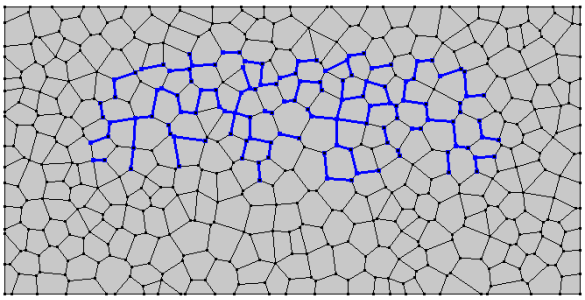


Figure 4.7: *Crack formation after* $3.9065 \cdot 10^{10}$ *cycles*

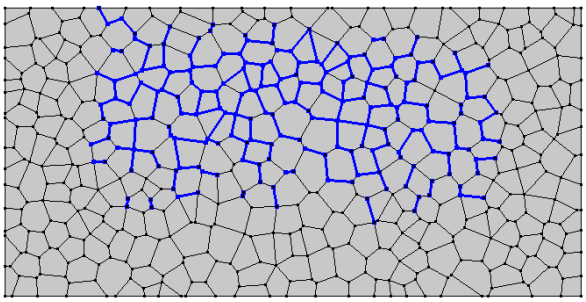


Figure 4.8: *Crack formation after* $6.9734 \cdot 10^{10}$ *cycles, failure has occurred*

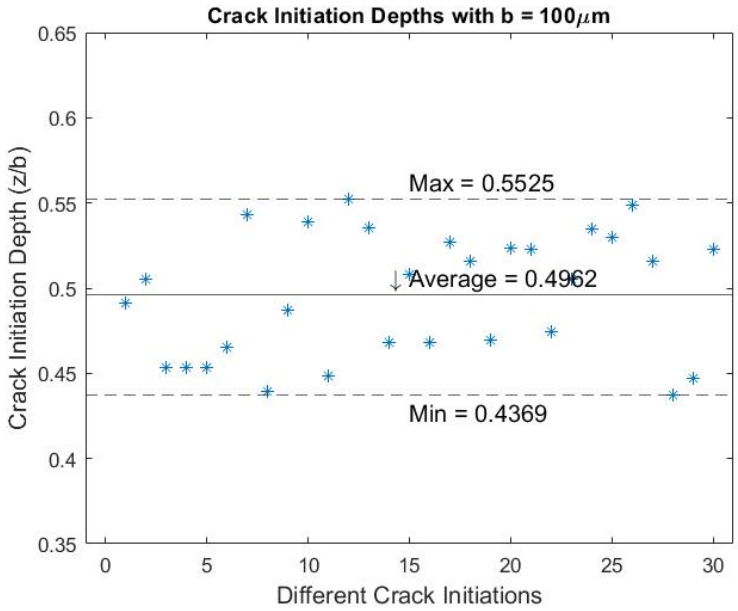


Figure 4.9: *Crack initiation depths relative to the half-width for* $b = 100\mu m$

4.1.3. Experimental Data

The first test to check if the model is build correctly and creates results that fit in the required range is the initiation life test. For this test, the model is run until a crack is initiated and the amount of load cycles until this crack initiation is recorded. This is a quick way to check if the model produces a scatter that comes close to the experimental results and the range provided by Lorenz. Fig. 4.10 represents the Weibull plot created from 30 different microstructures with a $P_{max} = 1GPa$ and Fig. 4.11 represents the Weibull plot created from 30 different microstructures with a $P_{max} = 2GPa$. It can be seen in these figures that the slope obtained from the model ($e = 53$ and $e = 62$ respectively) is fairly high compared to where it is supposed te be at (10-30). While this is on the high side, the initiation life values are in the range that they are supposed to be in.

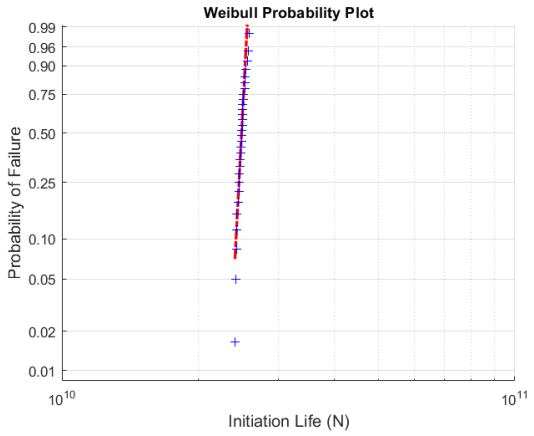


Figure 4.10: *Weibull life plot of the initiation lives for different microstructures with* $P_{max} = 1GPa$

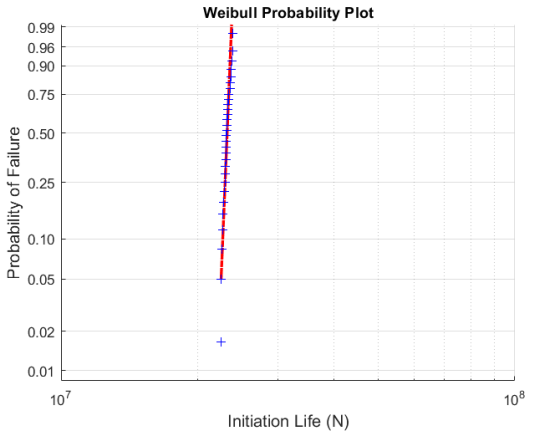


Figure 4.11: *Weibull life plot of the initiation lives for different microstructures with* $P_{max} = 2GPa$

For the final validation of the model, 30 different microstructures were modelled with a half-width $b = 100\mu m$ and a maximum pressure $P_{max} = 1GPa$. The scatter of this data is plotted in Fig. 4.12 together with the Lundberg-Palmgren model and four other numerical models. In Table 4.1 the slope and the L_{10} life is compared of the models with the current model and with the experimentally determined slopes from Harris and Barnsby and Harris and Kotzalas. It can be seen that the current model compares fairly well to the other models but that the slope is still high (6.75) and is not in the range set by the experimental results. The L_{10} life is

in the range that it needs to be in (around 10^{10} and close to the value of Jalalahmadi) and therefore it can be said that this model is acceptable for further use. The high slope and therefore low scatter might be explained by the lack of a frictional component on the surface and the use of initially homogeneous material properties and the use of an isotropic material which have been noted in other models as the cause of a higher slope [56]. Further refinement of the model might be able to reduce the slope and get it in the range of the experimental data.

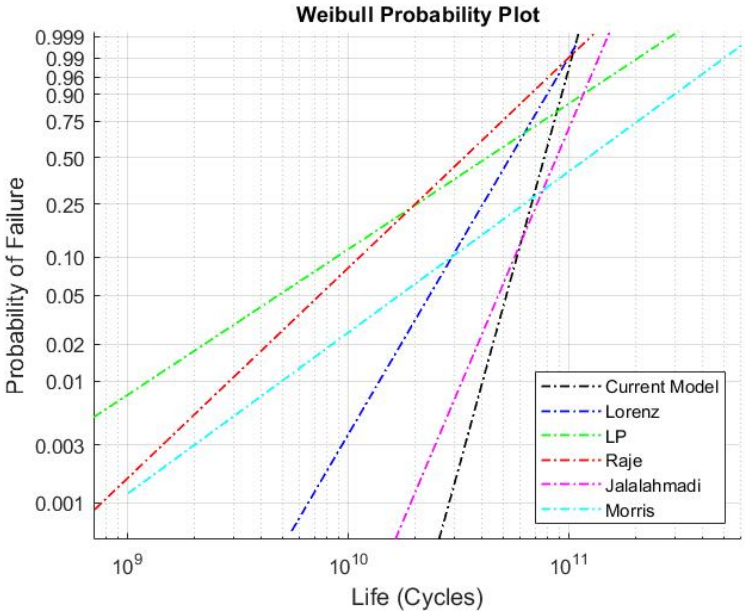


Figure 4.12: *Fatigue life distribution for $P_{max} = 1GPa$*

Model	Weibull slope, e	L_{10} life (Cycles)
Current Model	6.75	$5.83 \cdot 10^{10}$
Lundberg-Palmgren [35]	1.14	$0.86 \cdot 10^{10}$
Lorenz et al. [34]	3.20	$3.08 \cdot 10^{10}$
Raje et al. [50]	1.72	$1.06 \cdot 10^{10}$
Jalalahmadi and Sadeghi [27]	3.94	$5.31 \cdot 10^{10}$
Morris et al. [41]	1.36	$3.00 \cdot 10^{10}$
Harris and Barnsby [18]	$0.51 \leq e \leq 5.7$	
Harris and Kotzalas [19]	$0.7 \leq e \leq 3.5$	

Table 4.1: *Weibull slopes and L_{10} lives of previously created numerical models and the Lundberg-Palmgren method*

4.2. Testing Procedure

After the validation of the model, the model can be used to test what the effects of scaling the dimensions are on the fatigue life. First the models with the different half-widths are validated because it is unknown how the model behaves when smaller half-widths are used. The results of the scaled models can then be compared to each other to look what the effect of scaling might be. The contact pressure applied to the model during all the tests is kept the same as the stress used in the validation tests ($P_{max} = 1GPa$) but the dimensions are changed compared to the validation model. The way the dimensions are changed is by changing the contact half-width (b) used in the model because a smaller half-width means smaller rollers when the stress is kept the same. It is important to note that in this model, the time to model increases almost exponentially with the increase in half-width. For this reason, a smaller half-width is preferred for further testing since this reduces the testing time quite a lot and for that reason no testing is done with half-widths that are larger than $100\mu m$.

Since the model creates a scatter due to the varying microstructure, a large set of data points has to be created and placed in a Weibull plot to see what the distribution and the L_{10} life is. To create a large enough dataset, the model is run 30 times for each half-width such that it creates representative dataset to reality. The testing is started with a half-width of $b = 50\mu m$ and is moved up in increments of 10μm. When the testing is completed, the different Weibull plots and the L_{10} lives are looked at and evaluated to see what effect the scaling has on the fatigue life.

5

Outcome

This chapter presents and discusses the results that were gathered from the numerical model for the different contact half-widths. The fatigue lives and slopes that were found for the different half-widths are compared and also used for further validation of the model. The stress distribution, crack initiation and formation and other factors are investigated for each half-width and compared with the different half-widths. This is done to study the effect of scaling and investigate the differences between the different sized structures and to study what the behaviour of the material might be. After that, a conclusion is formulated about the gathered results and about the observations and statements made in the discussion. At last several recommendations are made for further work and different research that might apply similar techniques.

5.1. Results

This section presents the results and gives a possible explanation about why the results are what they are. First the models with the different half-widths are validated in the same manner as the original model ($b = 100\mu m$) to see how the models compare to other models and the first model. This is important to do since the model might behave differently with different half-widths. The results are also used to further investigate the behaviour of the scaled structures.

5.1.1. Stress Distribution

First, the stress distribution in the material is tested by moving a Hertzian pressure distribution over the surface. The depth and the magnitude of the stress is measured while moving the load and these values are averaged. This is repeated 40 times ans each time with a different microstructure and this is done for each half-width. The stresses and depths for each half-width can be found in appendix A but a summary and comparison between the half-widths can be found below.

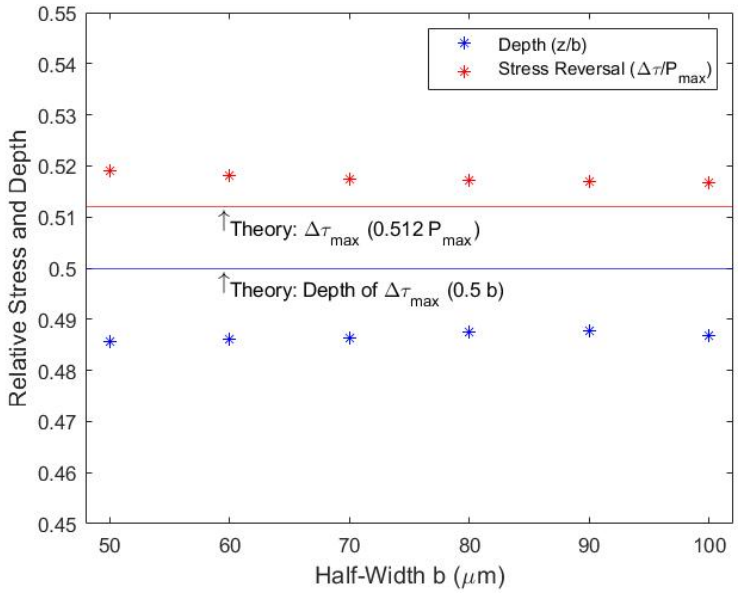


Figure 5.1: *Relative maximum shear stress reversal and its depth of each half-width compared to theory*

Half-Width b (μm)	Shear Stress Depth (z/b)	Shear Stress Reversal ($\Delta\tau/P_{max}$)
50	0.4857	0.5191
60	0.4861	0.5182
70	0.4864	0.5175
80	0.4874	0.5171
90	0.4877	0.5169
100	0.4869	0.5166

Table 5.1: *Depth and range of the critical stress relative to the half-width (b) and the maximum pressure (P_{max})*

Both Fig. 5.1 and Table 5.1 show that the stress depths and the magnitude of the stresses are close to the theory and do not deviate much from the first validated model. In the stress magnitude it is noted that there is a slight trend towards the theory with an increase in half-width and at first this can also be seen with the depth but with $b = 100\mu m$, the depth moves away from the trend and lowers slightly. Appendix A shows that the stresses have a very small increase in the scatter in depths and magnitudes for the smaller half-widths which might be due to having less elements in the microstructure and therefore having a less refined mesh compared to the larger half-width models. This less refined mesh causes the values to differ a bit more and therefore the stresses are influenced more by the boundaries of the domain and this might explain why the values come closer to theory with a larger half-width. The centerline stresses seen in appendix A.2 follow the theory very closely and deviate in a similar manner to the theory and to the other half-widths.

5.1.2. Crack Formation

In appendix B several examples of the crack formation from beginning to end for each half-width can be found with the according load cycles at which the crack formation is shown at. The crack initiation depth of the different half-widths is shown below and the average depth and the range are compared in Table 5.2 and in Fig. 5.8.

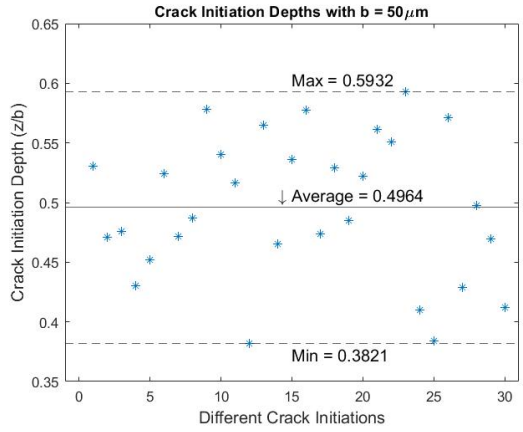


Figure 5.2: *Crack initiation depths relative to the half-width for b = 50μm*

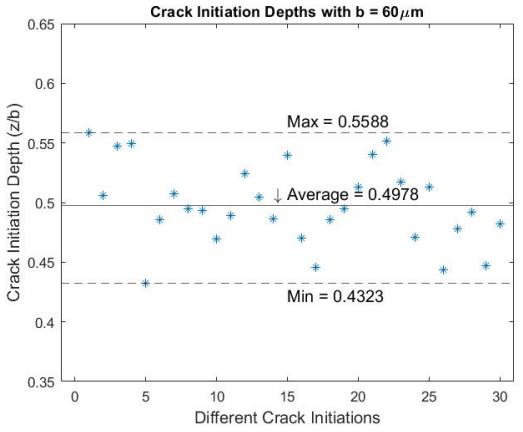


Figure 5.3: *Crack initiation depths relative to the half-width for b = 60μm*

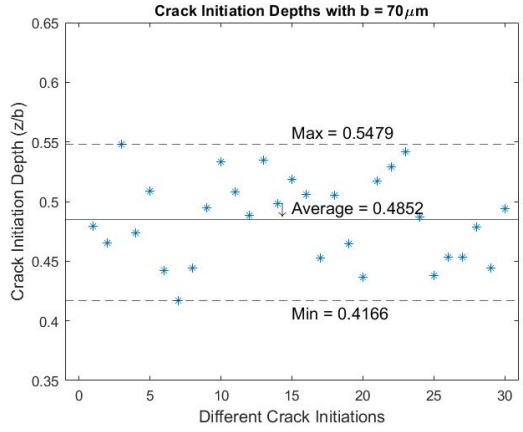


Figure 5.4: *Crack initiation depths relative to the half-width for b = 70μm*

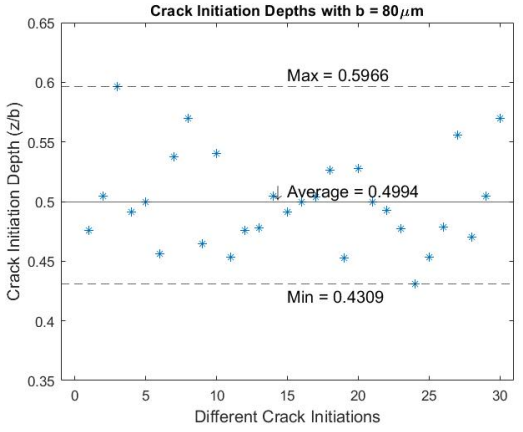


Figure 5.5: *Crack initiation depths relative to the half-width for b = 80μm*

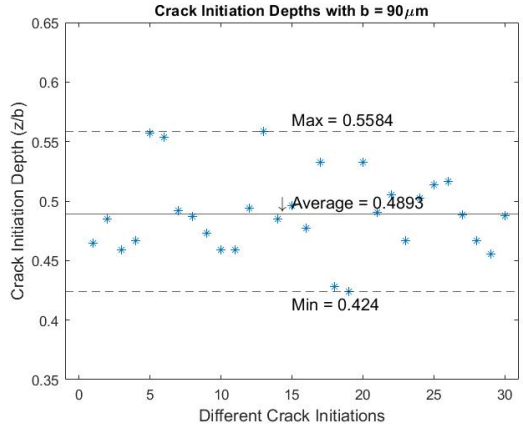


Figure 5.6: *Crack initiation depths relative to the half-width for b = 90μm*

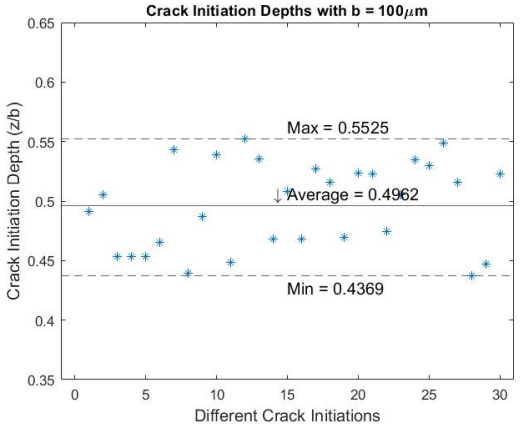


Figure 5.7: *Crack initiation depths relative to the half-width for b = 100μm*

Half-Width b (μm)	Average Crack Initiation Depth (z/b)	Crack Initiation Depth Range
50	0.4964	0.211
60	0.4978	0.127
70	0.4852	0.131
80	0.4994	0.166
90	0.4893	0.134
100	0.4921	0.116

Table 5.2: *Average depth of the crack initiation and average initiation life of the different half-widths*

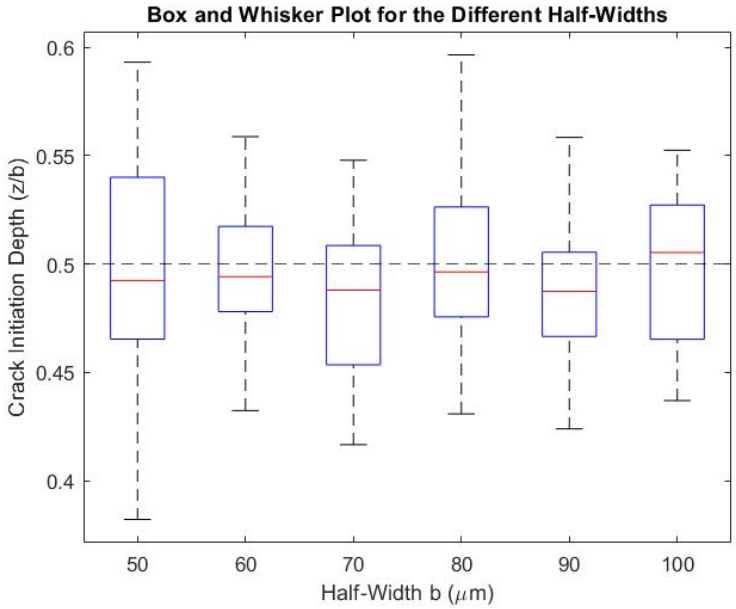


Figure 5.8: *Box and whisker plot to show the variation in the data for the different half-widths with the theoretical value of* $0.5 \cdot b$ *shown*

It can be seen that the average crack initiation depth is close to the theoretical depth of maximum shear stress reversal and close to what has been observed experimentally and that the range is comparable to what was found in the results of Chen et al [7]. It has to be noted that the range of the crack initiation varies from half-width to half-width, $50\mu m$ and $80\mu m$ show a larger range in the extremes compared to the other half-widths while being quite similar in the inter-quartile range as seen in Fig 5.8. The inter-quartile range of the $60\mu m$ dataset is compared to the other plots quite small which means that the data has a lower amount of scatter. This scatter will vary with each set of test results due to the random creation of the microstructure and therefore these results could have been completely different. It can be seen in appendix B that the crack formation for the different half-widths happens in a similar manner. The first crack initiates around the same depth relative to the half-width and then the first couple cracks that follow form parallel to the surface. These cracks start to coalesce and spread in the material until a crack reaches the surface and the material fails.

5.1.3. Fatigue Life

The last validation step is checking the initiation life and total life Weibull plots and their slope with experimental data but also with previous models and the current model. This testing delivers the data that later is used to check how the scaling affects the fatigue life. The Weibull plots with the initiation and propagation for each half-width and the initiation and propagation for each microstructure can be found in appendix C. Below each Weibull plot, a bar plot with the combination of the initiation and total life can be seen and further down, the combination of the initiation life and after the full life Weibull plots of all the half-widths can be seen. In this section the combination of the initiation and full life can be found and a table that summarizes

the slopes and L_{10} lives with a comparison between the initiation and full life.

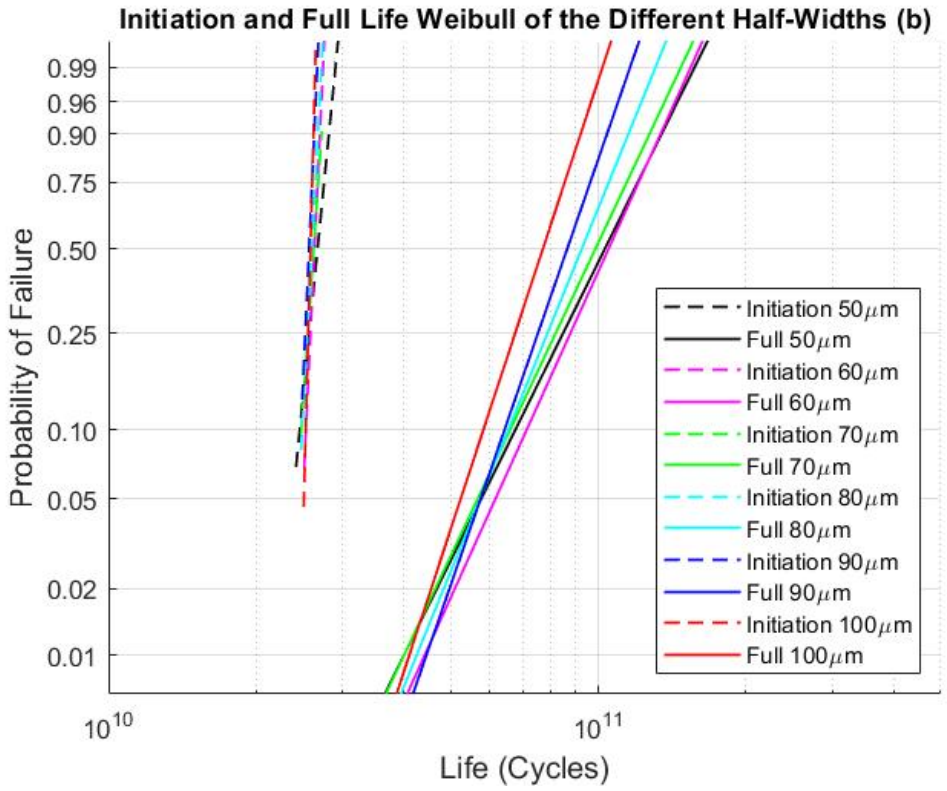


Figure 5.9: *Weibull Slopes of the initiation and full lives of all the half-widths*

Half-Width b (μm)	Initiation Life		Total Life		% Initiation Life to Total Life
	Weibull Slope, e	L_{10} Life (Cycles)	Weibull Slope, e	L_{10} Life (Cycles)	
50	14.73	$2.46 \cdot 10^{10}$	4.62	$6.86 \cdot 10^{10}$	35.9
60	25.43	$2.53 \cdot 10^{10}$	4.85	$7.11 \cdot 10^{10}$	35.6
70	43.18	$2.47 \cdot 10^{10}$	4.67	$6.61 \cdot 10^{10}$	37.4
80	37.78	$2.49 \cdot 10^{10}$	5.39	$6.54 \cdot 10^{10}$	38.1
90	42.82	$2.49 \cdot 10^{10}$	6.26	$6.39 \cdot 10^{10}$	39.0
100	53.15	$2.53 \cdot 10^{10}$	6.75	$5.83 \cdot 10^{10}$	43.4

Table 5.3: *Weibull slopes and L_{10} lives of the initiation and full life of all the different half-widths*

From Table 5.3 and appendix C, it can be seen that the Weibull plots shows an increase in slope for both initiation and total life with an increase in half-width meaning that the scatter decreases. In Fig. 5.9 the Weibull plots of all the data can be seen and the difference in slope and L_{10} can be seen here. This can be attributed to the fact that with a larger half-width, there are more grains and therefore more weak planes in the critically stressed area. This means that there is a higher chance of an edge/weak plane with an orientation that is more susceptible to damage and therefore endures higher critical stress. It can be said that the results therefore become more deterministic with larger half-widths thus increasing the Weibull slope.

5.1.4. Scaling

From the values seen in Table 5.3, it can be seen that the initiation lives do not change with a change in half-width and that there is no correlation between these values while it can be seen that the total L_{10} life

decreases with an increase in half-width (Fig. 5.10). In Fig. 5.10 it can be seen that a line with a slope of -0.02 can be drawn between the points showing that there is a decrease in L_{10} life and Fig. 5.11 shows the same but it is clear that these point do not follow the lines seen in the figure. The difference in the initiation life and total life of the different half-widths is shown in even further detail in the bar plot in Fig. 5.12. The decrease could be attributed to what was said previously about larger half-widths having more grains and edges in the critically stressed domain. Since there are more edges in the critically stressed domain, more edges can fail and the material can degrade faster while with the smaller half-widths only a couple edges are in the critical domain and therefore less edges endure damage and the same edges endure a higher amount of damage. The critical stress therefore does not affect as many grain and therefore the material does not degrade as quickly.

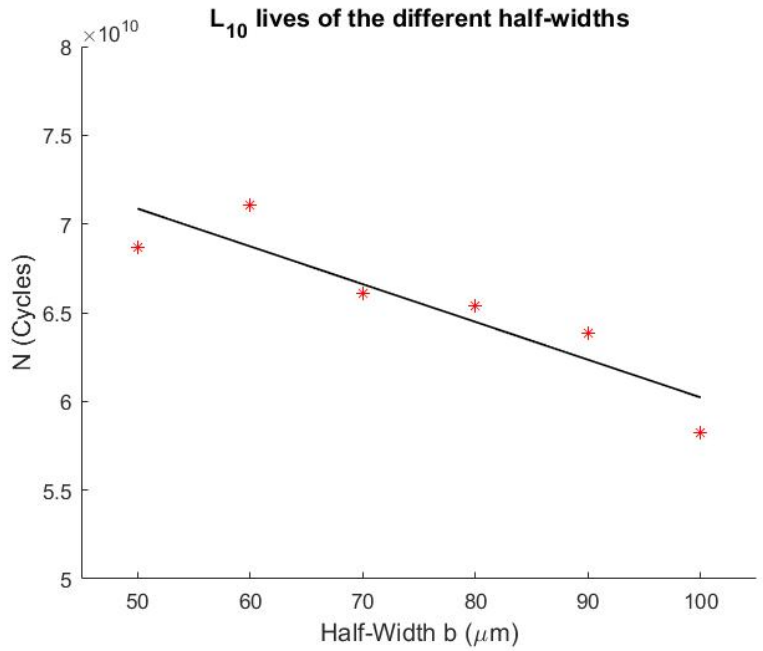


Figure 5.10: *The different L_{10} lives for the different half-widths with a trend line with slope -0.02 and $R^2 = 0.774$*

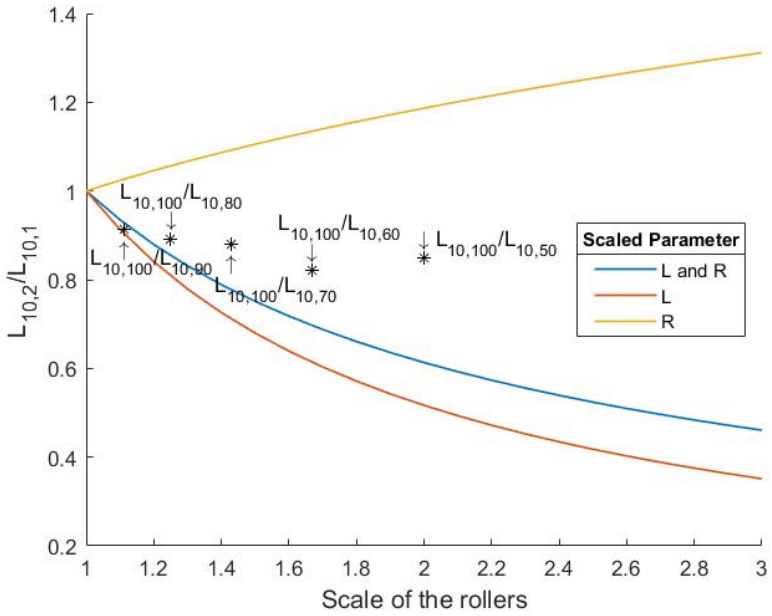


Figure 5.11: *L_{10} lives of all the tests compared to the ISO scaling*

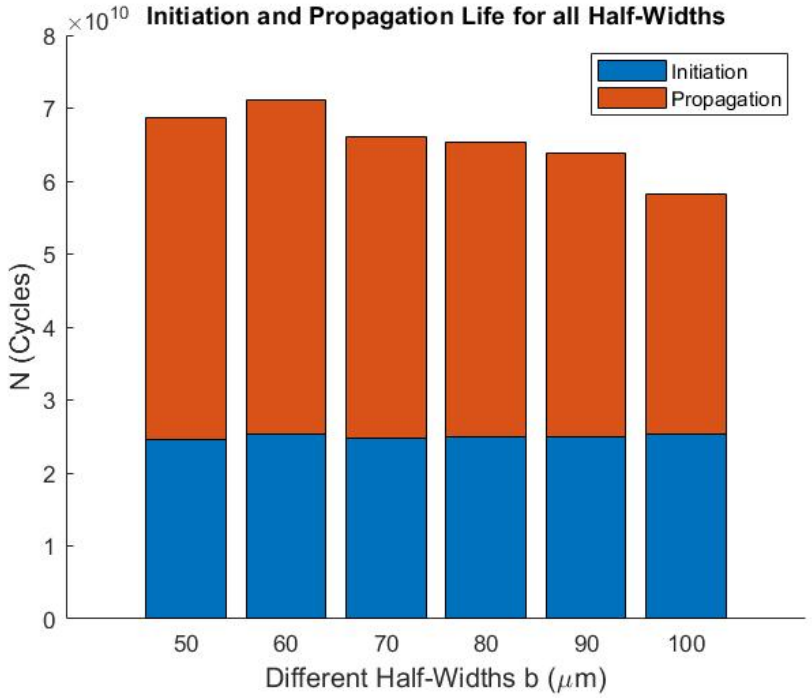


Figure 5.12: *Bar plot of the L_{10} initiation and full lives of all the half-widths*

5.2. Discussion

In this section, an answer is given to the questions that were asked and to the gap in the research that was found during the literature research. The research objective was

> How to predict lifetime and failure mechanisms of rolling contacts in DOT Direct Drive Pump?

Before answering this question, other questions were stated of which the answers lead up to the answer of the research objective.

The first question that has to be asked is, what failure mechanism is expected to occur in rolling contacts and it was found that there a several modes of failure like wear, corrosion, plastic deformation and fatigue. This all depends on the environment and the loading conditions that the contact experiences and some failures mechanisms can even be caused by another failure mechanism. The way failure occurs and the time until failure can vary a lot between these mechanisms and it is therefore important to figure out what the main failure mechanism is for the rolling contacts in the DOT DDP which has been discussed in chapter 2.4. This is done by looking at the environment and the loading conditions for that surface and in the case of this project, the loads are below the elastic limit and it is assumed that the contacts are dry and free of contamination. For these reasons it was determined that the main failure mechanism gravitated to rolling contact fatigue as the main mode of failure which has been discussed in further detail in the literature study in chapter 2.5. In this chapter, the different types of fatigue and the factors that influence fatigue life are discussed.

The follow up question to determining and understand the failure mechanism is, how is the lifetime of a rolling contact predicted, which depends on the failure mechanism and therefore in this case it is important to know how the fatigue life is predicted? Several methods have been developed over time, some making predictions depending on experimental results and some focussing on a homogeneous material structure which has been discussed in chapter 2.6. This paper has the aim to simulate reality as closely as possible due to the microstructural dependency of fatigue and therefore a method has to be chosen that fits this requirement.

The continuum damage mechanics method developed by Sadeghi and his research group has been chosen since this method proved to fit the purpose the best. It implements both the crack initiation and propagation and the model has a focus on high cycle fatigue but this method also focusses on materials that are of interest for DOT. This method has been discussed in detail in section 2.7 and worked out in further detail in chapter 3. The method was also proven to deliver results that are similar to reality and fit the range set by experimental results very well in previous research. Since the method simulates the microstructure, the results show a scatter that follows a Weibull plot and therefore the model has to be run several times to get an accurate prediction of the slope and with that the L_{10} life. This gives a prediction of the amount of load cycles required for 90% of the rolling contacts to survive.

Another question asked which is important for DOT but which can also prove significant to many other fields of research is how the scaling of a rolling contact influences the fatigue life? This question is asked since experimental testing of rolling contacts, of which fatigue is the main mode of failure, takes a long time and is expensive. The experimental testing of rolling contact fatigue is expensive because the test has to be performed several times to get an accurate life prediction and therefore a new test sample has to be created every time. If the the sample is large, it is most of the time expensive and the testing has to be performed at a slow rate for safety reasons. Therefore DOT and many other researchers have created test benches with samples that are smaller compared to the object of interest but it was still unclear how the results from the scaled sample translated to the original sample. An even faster and cheaper method to test for rolling contact fatigue is creating a numerical model that is able to simulate the fatigue life and the behaviour of the rolling contact surface. The model that was presented earlier has been used to test the influence of scaling since it is able to simulate the fatigue behaviour of a rolling contact but also steps are taken to create a numerical model for DOT which can accelerate the testing. The results from the model are discussed below.

5.2.1. Numerical Model

It can be seen in the results that the created model follows reality fairly well when taking into account that several assumptions have been made while making this model. The model is close to the range that was set by Harris and the total life is of the same magnitude as the other data from models and the data from Lundberg and Palmgren. As was said earlier, the scatter of this model is not in the range of the experimental data from Harris et al because the model does not include heterogeneous material properties, anisotropy, surface roughness, etc..

From the different half-width tests it can be seen that the smaller half-widths show a higher amount of scatter and that the total L_{10} life is higher for smaller half-widths. As was stated in the results, this could be due to less elements being in the critical domain and therefore being less prone to damage and therefore increasing the fatigue life.

The assumptions made by Sciammarella et al [55] who stated from their experimental observations that the fatigue life is reduced for smaller rollers and the theoretical assumptions made by Heliot [20] and Jaschinski et al [28] who also assumed a reduction in life with the reduction in size do not agree with the results found in this paper. The results found in this research show a decrease in fatigue life for larger rollers which can also be seen from the theoretical derivation of the ISO 281 formula for rolling bearings when scaling the length and radius but the results found during testing certainly do not follow this line.

While these results show a decrease in fatigue life for the larger rollers there are several things that have to be noted. Firstly, the model implements a 2D structure and therefore the length is not directly included and while the length is still found in the half-width equation, a 3D model will be able to directly include both parameters.

Secondly, the range of the tested half-widths goes from $50\mu m$ to $100\mu m$ in steps of $10\mu m$ and this upper limit was chosen due to time constraints. For a fairly quick computer it takes at least 12 hours to run one test for $b = 100\mu m$ which means that it would take 15 days to get a complete data set. The preference in this thesis was on creating a complete data set of 30 tests for each half-width instead of only a couple tests for a lot of half-widths such that the accuracy of the used results could be ensured. Due to the set range for the half-widths, it is unclear if this downward trend continues for larger roller dimensions.

Another factor to note is that while there is a trend in the data, some of the half-widths tested show a deviation of this line and to increase the accuracy of the found trend line slope, more tests have to be done for each half-width to further improve the results but also more half-widths have to be tested. With the increase of tests and data sets, a better assumption can be made and with that create an even better and more accurate understanding on the influence of scaling. The further improvement of the model will also bring the model closer to reality and therefore produce even better results and thus giving a more accurate prediction of the fatigue life but will also be able to produce results quicker.

5.3. Conclusion

When looking back at the research objective of finding a method to predict the lifetime and failure mechanisms of rolling contacts in the DOT Direct Drive Pump:

> The developed CDM method is able to give a prediction of the fatigue life and is able to simulate crack formation behaviour comparable to what happens in reality. The model can be adapted such that different materials and roller sizes can be applied and a valid prediction can be made on how the roller will behave. It is important to further develop the model and validate experimentally such that the prediction can come closer to reality

When looking at the influence of scaling on the fatigue life, the results showed a decrease in fatigue life with an increase in half-width. When applying this knowledge to scaled roller test bench results it can be said that the smaller rollers will perform for more load cycles if the roller dimensions are scaled equally but the time to test can still be shorter since the smaller rollers can be ran at a faster rate and multiple test benches can be made for the same price that larger test bench can be made. It is difficult to already make a statement about how many more cycles a smaller roller will endure because in the creation of the model several assumption have been made and it is unclear if the same trend continues for larger roller dimension. Therefore further tests, experimental and numerical, have to be done and improvements to the model have to be made such that at some point a correlation can be made between the scaling and the fatigue life. Further experimental testing will also give a better idea of how close the developed model actually is to reality and will help the investigation for finding the influence of scaling on fatigue life.

5.4. Recommendation

While this model is fairly detailed compared to other modelling methods, there is still plenty of room for improvement such that the model can come even closer to reality. In this section several recommendations are made that can be implemented in possible further work or for different research with a similar approach and a priority list is created which can be applied in following research.

5.4.1. Accuracy of the Model

The most important improvements can be made in making the model more accurate and similar to reality. Several factors are assumed in this model for efficiency but also due to a lack of time or missing information. In this thesis, the model assumes an homogeneous isotropic material, the next step should be the implementation of anisotropy and heterogeneous material parameters since this is a key factor in improving the scatter in the results. Besides that, it is important to further verify the accuracy of the model and the best way to do this is experimentally and comparing to older data sets. With the experimental validation, it is possible to check how close the results of the numerical model are to reality and by doing this validation, a better understanding of the results is created.

The next addition to the model should be the implementation of surface roughness since this does not scale with the size and therefore it will have an influence on the results of scaled models. The implementation of the surface roughness can also be used to test certain surface finishes and which could translate to the manufacturing of the rollers.

Friction was assumed to be negligible in the model and material defects were not implemented in this paper but for the refinement of the model it is important to add these. Friction can cause the stress to move closer to the surface and therefore alter the depth of crack initiation or even cause the crack to form on the surface. The latter is not the case in this problem since the friction was low but to be able to use the model to test different types of situations, like failure occurring in a bearing that causes the roller to endure higher friction levels, it might be interesting to implement friction and be able to change it easily.

The last couple assumptions that require some further in depth research is adding grain failure instead of only assuming grain boundary failure but also creating a 3D model which is computationally expensive but very interesting to implement instead of assuming a 2D structure.

5.4.2. Computational Efficiency

A more efficient model improves the usability and reduces the time to model from beginning to failure and therefore larger models can be created. While the jump-in-cycle method improved the efficiency of the model to a great extend, it is noticed that the jump in cycles can go to almost zero when an edge is about to break since the damage on that edge is high. The small jump in cycles increases the amount of jumps required until failure occurs and thus the time to model increases. An improved jump-in-cycle method should be implemented that neglects these small jumps and therefore reduces the time to model. A modified jump-in-cycle method has been discussed in a couple models that also implement a 3D model and has been proven to reduce the computational time immensely [3][67].

Another factor that reduces the efficiency of the model is the retrieval of data from COMSOL to Matlab. Therefore further research has to be done to create a method that reduces the time of this data retrieval. Other improvement of the model were the mesh improvement and with that grain boundary improvement which reduce simulation time but also improve the results that are retrieved from the model and improve the crack formation. While the method that was used worked fairly well, it still can use some tweaks such that small angles in the structure are avoided.

5.4.3. Accuracy of the Results

To improve the results in further work, even bigger sample sizes for each half-width should be taken, more half-widths and a larger range for the half-widths should be tested but first the improvement of the model itself should be done such that the model gives results that are closer to reality.

5.4.4. Additional Work

Further work on the model could include implementing and studying material treatments and improving the usability such that different materials and different parameters can easily be implemented. While this sounds fairly easy, each material has a differently shaped structure and therefore a study has to be done on creating a similar structure at first and after that further research has to be done on the implementation of this material in the model. This could for example be research on the friction between the crack faces.

After the implementation of a 3D model and the other recommendations it could be interesting to implement full rollers because in the current model a semi-infinite domain is used and a prescribed stress distribution is placed onto the surface. With the implementation of the full roller, it is easier to add certain design choices and see what the effect of these choices are on the roller and the fatigue life.

5.4.5. Priority List

The following priority list is based on personal experience and can vary from researcher to researcher but the list focusses on creating a more accurate model by improving the scatter and the total life results. These will be ranked based on ease of implementation and effect of the implementation.

1. Heterogeneous material structure.
    - This is quite an easy addition since the model already assigns each grain individual material properties. The reason it has not been added in this model is due to time constraints.
2. Anisotropy.
    - Adding anisotropy has been studied by a couple researchers before [47][64] and COMSOL has the capabilities of adding anisotropy but this requires quite a bit of further research.
3. Experimental Validation.
    - The experimental validation will prove how accurate the model is but this requires a tedious and long testing process. Therefore it is important to test efficiently and document properly such that the results can be used widely throughout the research, for example testing the material combination, validation of the numerical model and testing for scaling.
4. Surface roughness.
    - Surface roughness has been studied by Lorenz [34] and can be used to further study the effect of surface roughness on the fatigue life. This in turn can be used to see which surface roughness might be enough to get a sufficient lifetime and therefore possibly reduce the manufacturing cost.
5. Improved Computational Efficiency.
    - Improving the jump-in-cycle method and the retrieval of data will reduce the modelling time such that the effect of changes made to the model can be studied faster and adjusted if needed.
6. Friction.
    - Since the friction is low, it was not taken into account but it can be seen in the section on Rolling with Friction 2.3.2 that when it is considered, it can have a major effect on the stress distribution in the material.

5.5. Roadbook

While this research can be used by other researchers, the purpose of the project was predicting the failure mechanism and life of the DOT Direct Drive Pump. The method and mechanism found here is based on friction and contaminant free contact which is based on the Rolling Test Bench and causes the main mode of failure to be subsurface rolling contact fatigue. This section will discuss the steps that have to be taken to get a better prediction and to create a method to find the best material combination for the purpose.

At first, it is important to identify the most prominent failure method in the DDP by asking the question, are these contacts going to be free of contaminants or is the amount negligible and how much friction will actually be on the surface. If the friction and the amount of contaminants is negligible like in the RTB, the development of the provided method can be continued. If the friction is high but the contaminants low, the same modelling method can be applied but friction has to be added and another method has to be developed to see the influence of the friction. If the contaminants are high, a different method has to be developed that is able to include these contaminants and the failure method has to be researched in detail.

Basing the following steps on the RTB on which the model used in this project is focussed on, the first step that has to be taken is further developing and refining the model of which the priority list can be seen in the previous section. When the developed method is refined to a point that it able to output values that mimic reality, this method can be applied to seek for the best material combination together with experiments. The method does not have to put out values that are completely the same but if the deviation is known compared to reality, the model data can be adjusted to what it is supposed to be.

Bibliography

[1] O.H. Basquin. The exponential law of endurance tests. *American Society for Testing and Materials Proceedings*, 10:625–630, 1910.

[2] Sabah M. Beden, Shahrum Abdullah, and Ahmad Kamal Ariffin. Review of fatigue crack propagation models for metallic components. *European Journal of Scientific Research*, 28(3):364–397, 2009.

[3] John A. R. Bomidi, Nick Weinzapfel, Farshid Sadeghi, Alexander Liebel, and Joerg Weber. An Improved Approach for 3D Rolling Contact Fatigue Simulations with Microstructure Topology. *Tribology Transactions*, 56(3):385–399, may 2013. ISSN 1040-2004. doi: 10.1080/10402004.2012.754072. URL https://www.tandfonline.com/doi/abs/10.1080/10402004.2012.754072.

[4] John A.R. Bomidi, Nick Weinzapfel, Chin-Pei Wang, and Farshid Sadeghi. Experimental and numerical investigation of fatigue of thin tensile specimen. *International Journal of Fatigue*, 44:116–130, nov 2012. ISSN 01421123. doi: 10.1016/j.ijfatigue.2012.05.013. URL http://dx.doi.org/10.1016/j.ijfatigue.2012.05.013.

[5] J. L. Chaboche. Continuum Damage Mechanics: Part I - General Concepts. *Journal of Applied Mechanics, Transactions ASME*, 55(1):59–64, 1988. ISSN 00218936. URL http://appliedmechanics.asmedigitalcollection.asme.org/.

[6] Jennifer K. Chalsma and E.V. Zaretsky. Design for life, plan for death. *Machine Design*, 66(15):57–59, 1994.

[7] Qing Chen, Eryu Shao, Dongmei Zhao, Juwen Guo, and Zhonghe Fan. Measurement of the critical size of inclusions initiating contact fatigue cracks and its application in bearing steel. *Wear*, 147(2):285–294, jul 1991. ISSN 00431648. doi: 10.1016/0043-1648(91)90186-X. URL https://linkinghub.elsevier.com/retrieve/pii/004316489190186X.

[8] Y.P. Chiu, T.E. Tallian, and J.I. McCool. An engineering model of spalling fatigue failure in rolling contact. *Wear*, 17(5-6):433–446, may 1971. ISSN 00431648. doi: 10.1016/0043-1648(71)90049-4. URL https://linkinghub.elsevier.com/retrieve/pii/0043164871900494.

[9] COMSOL. Fatigue Module User's Guide. *Training Manual*, pages 1–98, 2005.

[10] A. Ekberg. Rolling contact fatigue of railway wheels—a parametric study. *Wear*, 211(2):280–288, nov 1997. ISSN 00431648. doi: 10.1016/S0043-1648(97)00106-3. URL https://linkinghub.elsevier.com/retrieve/pii/S0043164897001063.

[11] A. Ekberg, H. Bjarnehed, and R. Lundbéan. A FATIGUE LIFE MODEL FOR GENERAL ROLLING CONTACT WITH APPLICATION TO WHEEL/RAIL DAMAGE. *Fatigue & Fracture of Engineering Materials and Structures*, 18(10):1189–1199, oct 1995. ISSN 8756-758X. doi: 10.1111/j.1460-2695.1995.tb00847.x. URL https://onlinelibrary.wiley.com/doi/10.1111/j.1460-2695.1995.tb00847.x.

[12] Anders Ekberg. *Multiaxial Fatigue*. Chalmers University of Technology, Gothenburg, 5.6 edition, 2020.

[13] Antonio Gabelli, Junbiao Lai, Thore Lund, Karin Rydén, Ingemar Strandell, and Guillermo E. Morales-Espejel. The fatigue limit of bearing steels – Part II: Characterization for life rating standards. *International Journal of Fatigue*, 38(February 2018):169–180, may 2012. ISSN 01421123. doi: 10.1016/j.ijfatigue.2011.12.006. URL https://linkinghub.elsevier.com/retrieve/pii/S0142112311003288.

[14] Somnath Ghosh and R.L. Mallett. Voronoi cell finite elements. *Computers & Structures*, 50(1):33–46, 1994.

[15] Somnath Ghosh and Suresh Moorthy. Elastic-plastic analysis of arbitrary heterogeneous materials with

the voronoi cell finite element method. *Computer Methods in Applied Mechanics and Engineering*, 121 (1-4):373–409, 1995.

[16] J Goodman. Roller and ball bearings. In *Minutes of the Proceedings of the Institution of Civil Engineers*, volume 189, pages 82–127. Thomas Telford-ICE Virtual Library, 1912.

[17] Dave Hannes and B. Alfredsson. A Parametric Investigation of Surface Initiated Rolling Contact Fatigue Using the Asperity Point Load Mechanism. *Key Engineering Materials*, 577-578(85):45–48, sep 2013. ISSN 1662-9795. doi: 10.4028/www.scientific.net/KEM.577-578.45. URL https://www.scientific.net/KEM.577-578.45.

[18] T. A. Harris and R. M. Barnsby. Life ratings for ball and roller bearings. *Proceedings of the Institution of Mechanical Engineers, Part J: Journal of Engineering Tribology*, 215(6):577–595, jun 2001. ISSN 1350-6501. doi: 10.1243/1350650011543817. URL http://journals.sagepub.com/doi/10.1243/1350650011543817.

[19] Tedric A. Harris and Michael N. Kotzalas. *Rolling Bearing Analysis - 2 Volume Set*. CRC Press, nov 2006. ISBN 9781482275148. doi: 10.1201/9781482275148. URL https://www.taylorfrancis.com/books/9780429621024.

[20] Coralie Heliot. SMALL-SCALE TEST METHOD FOR RAILWAY DYNAMICS. *Vehicle System Dynamics*, 15(sup1):197–207, jan 1986. ISSN 0042-3114. doi: 10.1080/00423118608969136. URL http://www.tandfonline.com/doi/abs/10.1080/00423118608969136.

[21] Heinrich Hertz. On the contact of elastic solids. *Z. Reine Angew. Mathematik*, 92:156–171, 1881.

[22] Felix Hofmann, Gratiela Bertolino, Andrei Constantinescu, and Mohamed Ferjani. Numerical exploration of the Dang Van high cycle fatigue criterion: application to gradient effects. *Journal of Mechanics of Materials and Structures*, 4(2):293–308, apr 2009. ISSN 1559-3959. doi: 10.2140/jomms.2009.4.293. URL http://msp.org/jomms/2009/4-2/p08.xhtml.

[23] American Geosciences Institute. What are the advantages and disadvantages of offshore wind farms?, 2019. URL https://www.americangeosciences.org/critical-issues/faq/what-are-advantages-and-disadvantages-offshore-wind-farms.

[24] E. Ioannides and T. A. Harris. A New Fatigue Life Model for Rolling Bearings. *Journal of Tribology*, 107(3):367–377, jul 1985. ISSN 0742-4787. doi: 10.1115/1.3261081. URL https://asmedigitalcollection.asme.org/tribology/article/107/3/367/437635/A-New-Fatigue-Life-Model-for-Rolling-Bearings.

[25] Osamu Ito and E.R. Fuller. Computer modelling of anisotropic grain microstructure in two dimensions. *Acta Metallurgica et Materialia*, 41(1):191–198, jan 1993. ISSN 09567151. doi: 10.1016/0956-7151(93)90350-2. URL https://linkinghub.elsevier.com/retrieve/pii/0956715193903502.

[26] Behrooz Jalalahmadi and Farshid Sadeghi. A Voronoi Finite Element Study of Fatigue Life Scatter in Rolling Contacts. *Journal of Tribology*, 131(2):1–15, apr 2009. ISSN 0742-4787. doi: 10.1115/1.3063818. URL https://asmedigitalcollection.asme.org/tribology/article/doi/10.1115/1.3063818/451525/A-Voronoi-Finite-Element-Study-of-Fatigue-Life.

[27] Behrooz Jalalahmadi and Farshid Sadeghi. A Voronoi FE Fatigue Damage Model for Life Scatter in Rolling Contacts. *Journal of Tribology*, 132(2):1–14, apr 2010. ISSN 0742-4787. doi: 10.1115/1.4001012. URL https://asmedigitalcollection.asme.org/tribology/article/doi/10.1115/1.4001012/468645/A-Voronoi-FE-Fatigue-Damage-Model-for-Life-Scatter.

[28] Alfred Jaschinski, Hugues Chollet, Simon Iwnicki, Alan Wickens, and Jurgen Wurzen. The Application of Roller Rigs to Railway Vehicle Dynamics. *Vehicle System Dynamics*, 31(5-6):345–392, jun 1999. ISSN 0042-3114. doi: 10.1076/vesd.31.5.345.8360. URL http://www.tandfonline.com/doi/abs/10.1076/vesd.31.5.345.8360.

[29] Yanyao Jiang and Huseyin Sehitoglu. A model for rolling contact failure. *Wear*, 224(1):38–49, January 1999. ISSN 0043-1648. doi: 10.1016/S0043-1648(98)00311-1. Funding Information: The work is sup-

ported by Association of American Railroads and the work on Hadfield steel is partially sponsored by the National Science Foundation, Mechanics and Materials Program, Grant No. CMS 94-14525.

[30] K. L. Johnson. *Contact Mechanics*. Cambridge University Press, may 1985. ISBN 9780521255769. doi: 10.1017/CBO9781139171731. URL https://www.cambridge.org/core/product/identifier/9781139171731/type/book.

[31] K. L. Johnson. Contact Mechanics, 1989. ISSN 0742-4787.

[32] Jean Lemaitre. *A Course on Damage Mechanics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996. ISBN 978-3-540-60980-3. doi: 10.1007/978-3-642-18255-6. URL http://link.springer.com/10.1007/978-3-642-18255-6.

[33] WE Littmann and PM Ku. The mechanism of contact fatigue. *Interdisciplinary approach to the lubrication of concentrated contacts*, pages 309–377, 1969.

[34] Steven J. Lorenz, Farshid Sadeghi, Hitesh K. Trivedi, Lewis Rosado, Mathew S. Kirsch, and Chinpei Wang. A continuum damage mechanics finite element model for investigating effects of surface roughness on rolling contact fatigue. *International Journal of Fatigue*, 143(September 2020):105986, feb 2021. ISSN 01421123. doi: 10.1016/j.ijfatigue.2020.105986. URL https://linkinghub.elsevier.com/retrieve/pii/S0142112320305181.

[35] Gustaf Lundberg and Arvid Palmgren. Dynamic Capacity of Rolling Bearings. *Journal of Applied Mechanics*, 16(2):165–172, jun 1949. ISSN 0021-8936. doi: 10.1115/1.4009930. URL https://asmedigitalcollection.asme.org/appliedmechanics/article/16/2/165/1106338/Dynamic-Capacity-of-Rolling-Bearings.

[36] A. MAZZÙ. A numerical approach to subsurface crack propagation assessment in rolling contact. *Fatigue & Fracture of Engineering Materials & Structures*, 36(6):548–564, jun 2013. ISSN 8756758X. doi: 10.1111/ffe.12024. URL https://onlinelibrary.wiley.com/doi/10.1111/ffe.12024.

[37] K. J. MILLER. THE SHORT CRACK PROBLEM. *Fatigue & Fracture of Engineering Materials and Structures*, 5(3):223–232, jul 1982. ISSN 8756-758X. doi: 10.1111/j.1460-2695.1982.tb01250.x. URL https://onlinelibrary.wiley.com/doi/10.1111/j.1460-2695.1982.tb01250.x.

[38] Ministerie. Zon en wind op zee zorgen voor groei duurzame energie. *Klimaatakkoord*, 8 2021. URL https://www.klimaatakkoord.nl/actueel/nieuws/2021/08/03/maandbericht-energieopwek.nl.

[39] Yukio MIYASHITA, Yoshihiro YOSHIMURA, Jin-Quan XU, Makoto HORIKOSHI, and Yoshiharu MUTOH. Subsurface Crack Propagation in Rolling Contact Fatigue of Sintered Alloy. *JSME International Journal Series A*, 46(3):341–347, 2003. ISSN 1344-7912. doi: 10.1299/jsmea.46.341. URL http://www.jstage.jst.go.jp/article/jsmea/46/3/46{_}3{_}341/{_}article.

[40] S. MOORTHY and S. GHOSH. A MODEL FOR ANALYSIS OF ARBITRARY COMPOSITE AND POROUS MICROSTRUCTURES WITH VORONOI CELL FINITE ELEMENTS. *International Journal for Numerical Methods in Engineering*, 39(14):2363–2398, jul 1996. ISSN 0029-5981. doi: 10.1002/(SICI)1097-0207(19960730)39:14<2363::AID-NME958>3.0.CO;2-D. URL https://onlinelibrary.wiley.com/doi/10.1002/(SICI)1097-0207(19960730)39:14{%}3C2363::AID-NME958{%}3E3.0.CO;2-D.

[41] Dallin Morris, Farshid Sadeghi, Yong-Ching Chen, Chinpei Wang, and Ben Wang. Predicting Material Performance in Rolling Contact Fatigue via Torsional Fatigue. *Tribology Transactions*, 62(4):614–625, jul 2019. ISSN 1040-2004. doi: 10.1080/10402004.2019.1587557. URL https://doi.org/10.1080/10402004.2019.1587557.

[42] Sebastiaan Paul Mulders, Niels Frederik Boudewijn Diepeveen, and Jan-Willem van Wingerden. Control design, implementation, and evaluation for an in-field 500 kW wind turbine with a fixed-displacement hydraulic drivetrain. *Wind Energy Science*, 3(2):615–638, sep 2018. ISSN 2366-7451. doi: 10.5194/wes-3-615-2018. URL https://wes.copernicus.org/articles/3/615/2018/.

[43] COMSOL Multiphysics. LiveLink for MATLAB User's Guide. *Version 4.3b*, page 282, 2013.

[44] Meysam Naeimi, Zili Li, Roumen H. Petrov, Jilt Sietsma, and Rolf Dollevoet. Development of a New Downscale Setup for Wheel-Rail Contact Experiments under Impact Loading Conditions. *Experimental Techniques*, 42(1):1–17, feb 2018. ISSN 0732-8818. doi: 10.1007/s40799-017-0216-z. URL http://link.springer.com/10.1007/s40799-017-0216-z.

[45] Mikael Nygårds and Peter Gudmundson. Three-dimensional periodic Voronoi grain models and micromechanical FE-simulations of a two-phase steel. *Computational Materials Science*, 24(4):513–519, jul 2002. ISSN 09270256. doi: 10.1016/S0927-0256(02)00156-8. URL https://linkinghub.elsevier.com/retrieve/pii/S0927025602001568.

[46] A. Otsuka, H. Sugawara, and M. Shomura. A TEST METHOD FOR MODE II FATIGUE CRACK GROWTH RELATING TO A MODEL FOR ROLLING CONTACT FATIGUE. *Fatigue & Fracture of Engineering Materials and Structures*, 19(10):1265–1275, oct 1996. ISSN 8756-758X. doi: 10.1111/j.1460-2695.1996.tb00949.x. URL https://onlinelibrary.wiley.com/doi/10.1111/j.1460-2695.1996.tb00949.x.

[47] Neil R. Paulson, John A.R. Bomidi, Farshid Sadeghi, and Ryan D. Evans. Effects of crystal elasticity on rolling contact fatigue. *International Journal of Fatigue*, 61:67–75, apr 2014. ISSN 01421123. doi: 10.1016/j.ijfatigue.2013.12.005. URL http://dx.doi.org/10.1016/j.ijfatigue.2013.12.005.

[48] Valentin L. Popov. *Contact Mechanics and Friction*. Springer Berlin Heidelberg, Berlin, Heidelberg, nov 2010. ISBN 978-3-642-10802-0. doi: 10.1007/978-3-642-10803-7. URL http://link.springer.com/10.1007/978-3-642-10803-7.

[49] Eugene I. Radzimovsky. Stress Distribution and Strength Condition of Two Rolling Cylinders Pressed Together. *University of Illinois Engineering Experiment Station*, 50(44):46, 1953. doi: 3500-11-52-50311. URL http://hdl.handle.net/2142/4380.

[50] Nihar Raje, Farshid Sadeghi, and Richard G. Rateick. A Statistical Damage Mechanics Model for Subsurface Initiated Spalling in Rolling Contacts. *Journal of Tribology*, 130(4):1–11, oct 2008. ISSN 0742-4787. doi: 10.1115/1.2959109. URL https://asmedigitalcollection.asme.org/tribology/article/doi/10.1115/1.2959109/470905/A-Statistical-Damage-Mechanics-Model-for.

[51] Nihar Raje, Farshid Sadeghi, Richard G. Rateick, and Michael R. Hoeprich. A Numerical Model for Life Scatter in Rolling Element Bearings. *Journal of Tribology*, 130(1):1–10, jan 2008. ISSN 0742-4787. doi: 10.1115/1.2806163. URL https://asmedigitalcollection.asme.org/tribology/article/doi/10.1115/1.2806163/462299/A-Numerical-Model-for-Life-Scatter-in-Rolling.

[52] Nihar Raje, Trevor S. Slack, and Farshid Sadeghi. A discrete damage mechanics model for high cycle fatigue in polycrystalline materials subject to rolling contact. *International Journal of Fatigue*, 31(2):346–360, feb 2009. ISSN 01421123. doi: 10.1016/j.ijfatigue.2008.08.006. URL http://dx.doi.org/10.1016/j.ijfatigue.2008.08.006https://linkinghub.elsevier.com/retrieve/pii/S0142112308001953.

[53] J Ringsberg. Life prediction of rolling contact fatigue crack initiation. *International Journal of Fatigue*, 23(7):575–586, aug 2001. ISSN 01421123. doi: 10.1016/S0142-1123(01)00024-X. URL https://linkinghub.elsevier.com/retrieve/pii/S014211230100024X.

[54] Farshid Sadeghi, Behrooz Jalalahmadi, Trevor S. Slack, Nihar Raje, and Nagaraj K. Arakere. A Review of Rolling Contact Fatigue. *Journal of Tribology*, 131(4):1–15, oct 2009. ISSN 0742-4787. doi: 10.1115/1.3209132. URL https://asmedigitalcollection.asme.org/tribology/article/131/4/041403/471354/A-Review-of-Rolling-Contact-Fatigue.

[55] C.A. Sciammarella, R.J.S. Chen, P. Gallo, F. Berto, and L. Lamberti. Experimental evaluation of rolling contact fatigue in railroad wheels. *International Journal of Fatigue*, 91:158–170, oct 2016. ISSN 01421123. doi: 10.1016/j.ijfatigue.2016.05.035. URL http://dx.doi.org/10.1016/j.ijfatigue.2016.05.035.

[56] Trevor Slack and Farshid Sadeghi. Explicit finite element modeling of subsurface initiated spalling in rolling contacts. *Tribology International*, 43(9):1693–1702, sep 2010. ISSN 0301679X. doi: 10.1016/j.triboint.2010.03.019. URL http://dx.doi.org/10.1016/j.triboint.2010.03.019.

[57] Trevor Slack and Farshid Sadeghi. Cohesive zone modeling of intergranular fatigue damage in rolling contacts. *Tribology International*, 44(7-8):797–804, jul 2011. ISSN 0301679X. doi: 10.1016/j.triboint.2011.02.003. URL http://dx.doi.org/10.1016/j.triboint.2011.02.003.

[58] T.A Stolarski and S Tobe. *Rolling Contacts*. John Wiley & Sons, Ltd, Chichester, UK, dec 2000. ISBN 9781118903001. doi: 10.1002/9781118903001. URL http://doi.wiley.com/10.1002/9781118903001.

[59] Haakon Styri. Fatigue strength of ball bearing races and heat-treated 52100 steel specimens. In *Proceedings-American Society for Testing and Materials*, volume 51, page 682. American Society for Testing and Materials, 1951.

[60] T. E. Tallian. On Competing Failure Modes in Rolling Contact. *A S L E Transactions*, 10(4):418–439, jan 1967. ISSN 0569-8197. doi: 10.1080/05698196708972201. URL https://doi.org/10.1080/05698196708972201.

[61] S.P. Timoshenko and J.N. Goodier. Theory of Elasticity, 1970.

[62] Lucian Mircea Tudose and Claudiu Ovidiu Popa. STRESS INTENSITY FACTORS ANALYSIS ON CRACKS IN THE HERTZIAN STRESSES FIELD OF TEETH GEARS. *The 10th International Conference on Tribology*, 1(January):1–8, 2016.

[63] Anton van Beek. *Advanced Engineering Design:"Lifetime performance and reliability"*. Delft University of Technology, Netherlands, 2015. ISBN 978-90-810406-1-7.

[64] Akhil Vijay and Farshid Sadeghi. A continuum damage mechanics framework for modeling the effect of crystalline anisotropy on rolling contact fatigue. *Tribology International*, 140(April), dec 2019. ISSN 0301679X. doi: 10.1016/j.triboint.2019.105845. URL https://linkinghub.elsevier.com/retrieve/pii/S0301679X19303524.

[65] Akhil Vijay and Farshid Sadeghi. An anisotropic damage model for tensile fatigue. *Fatigue & Fracture of Engineering Materials & Structures*, 42(1):129–142, jan 2019. ISSN 8756758X. doi: 10.1111/ffe.12877. URL https://onlinelibrary.wiley.com/doi/10.1111/ffe.12877.

[66] Anurag Warhadpande, Behrooz Jalalahmadi, Trevor Slack, and Farshid Sadeghi. A new finite element fatigue modeling approach for life scatter in tensile steel specimens. *International Journal of Fatigue*, 32(4):685–697, apr 2010. ISSN 01421123. doi: 10.1016/j.ijfatigue.2009.10.003. URL http://dx.doi.org/10.1016/j.ijfatigue.2009.10.003.

[67] Nick Weinzapfel and Farshid Sadeghi. Numerical modeling of sub-surface initiated spalling in rolling contacts. *Tribology International*, 59(November 2020):210–221, mar 2013. ISSN 0301679X. doi: 10.1016/j.triboint.2012.03.006. URL https://linkinghub.elsevier.com/retrieve/pii/S0301679X1200103X.

[68] Yuan Yin, Yun-Xia Chen, and Le Liu. Lifetime prediction for the subsurface crack propagation using three-dimensional dynamic FEA model. *Mechanical Systems and Signal Processing*, 87(October 2016):54–70, mar 2017. ISSN 08883270. doi: 10.1016/j.ymssp.2016.09.033. URL https://linkinghub.elsevier.com/retrieve/pii/S0888327016303855.

[69] T Yoshioka. Detection of rolling contact sub-surface fatigue cracks using acoustic emission technique. *Lubrication Engineering; (United States)*. ISSN 0024-7154. URL https://www.osti.gov/biblio/6293354.

[70] Xiaoyin Zhu. Tutorial on Hertz Contact Stress. *Opti 521*, pages 1–8, 2012.

A

Stress Distribution

A.1. Shear Stress Reversal

A.1.1. $50\mu m$

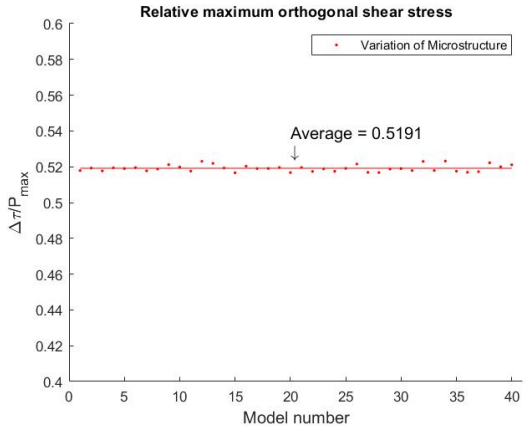


Figure A.1: *Average shear stress reversals for each loadcycle, the values vary more during the loadcycle*

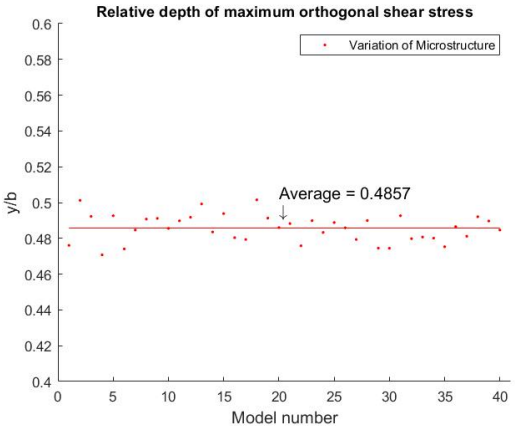


Figure A.2: *Relative depth of the maximum orthogonal shear stress*

A.1.2. $60\mu m$

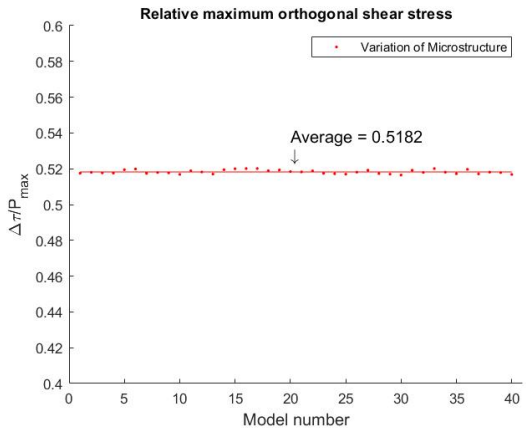


Figure A.3: *Average shear stress reversals for each loadcycle, the values vary more during the loadcycle*

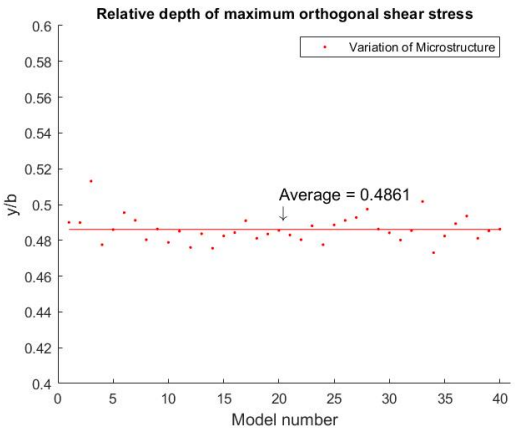


Figure A.4: *Relative depth of the maximum orthogonal shear stress*

A.1.3. $70\mu m$

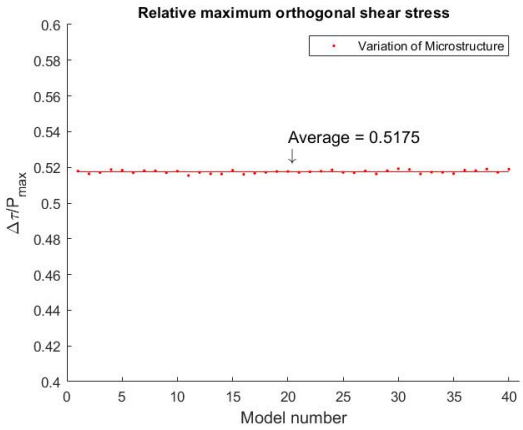


Figure A.5: *Average shear stress reversals for each loadcycle, the values vary more during the loadcycle*

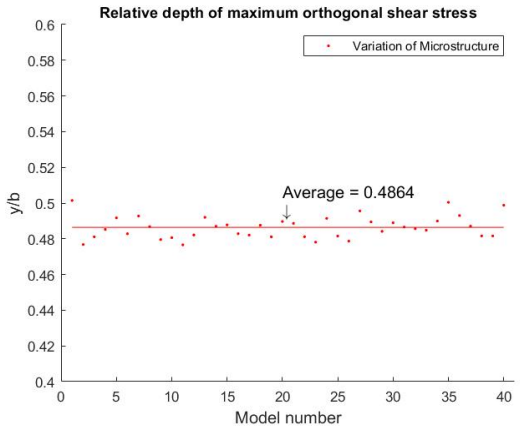


Figure A.6: *Relative depth of the maximum orthogonal shear stress*

A.1.4. $80\mu m$

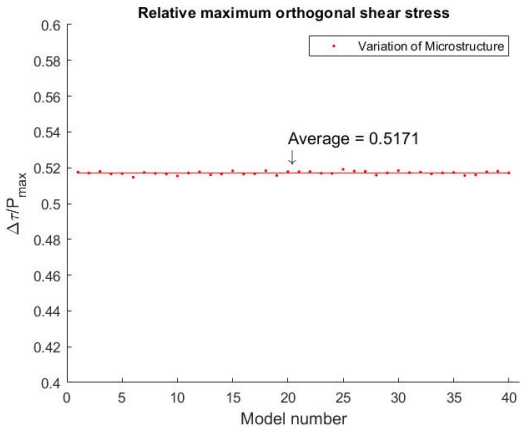


Figure A.7: *Average shear stress reversals for each loadcycle, the values vary more during the loadcycle*

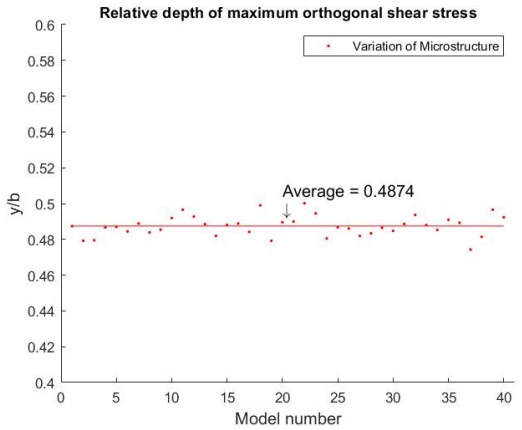


Figure A.8: *Relative depth of the maximum orthogonal shear stress*

A.1.5. $90\mu m$

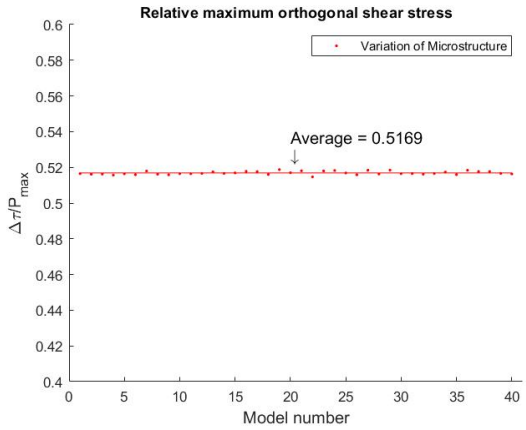


Figure A.9: *Average shear stress reversals for each loadcycle, the values vary more during the loadcycle*

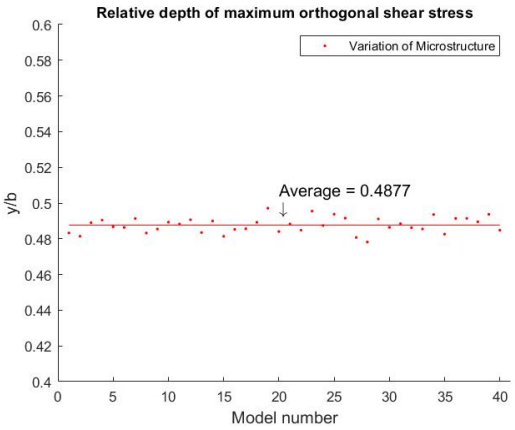


Figure A.10: *Relative depth of the maximum orthogonal shear stress*

A.1.6. $100\mu m$

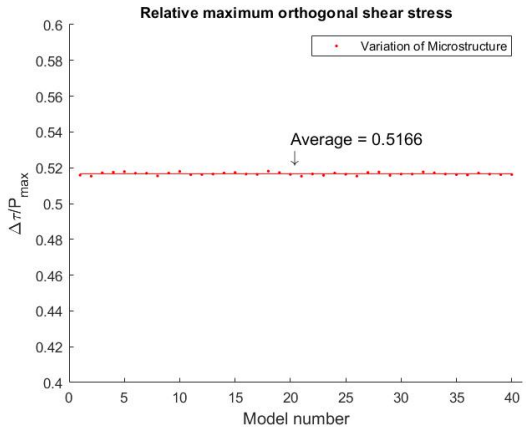


Figure A.11: *Average shear stress reversals for each loadcycle, the values vary more during the loadcycle*

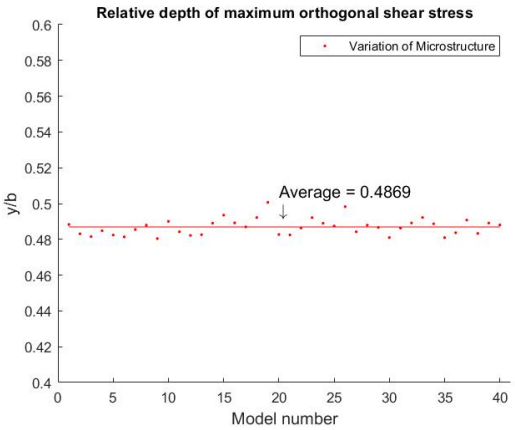


Figure A.12: *Relative depth of the maximum orthogonal shear stress*

A.2. Principal Stresses

A.2.1. $50\mu m$

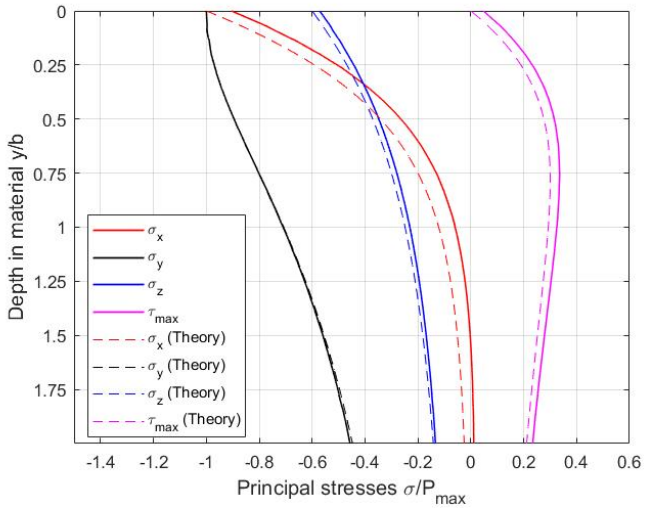


Figure A.13: *Principal centerline stresses in the material compared to the theory*

A.2.2. $60\mu m$

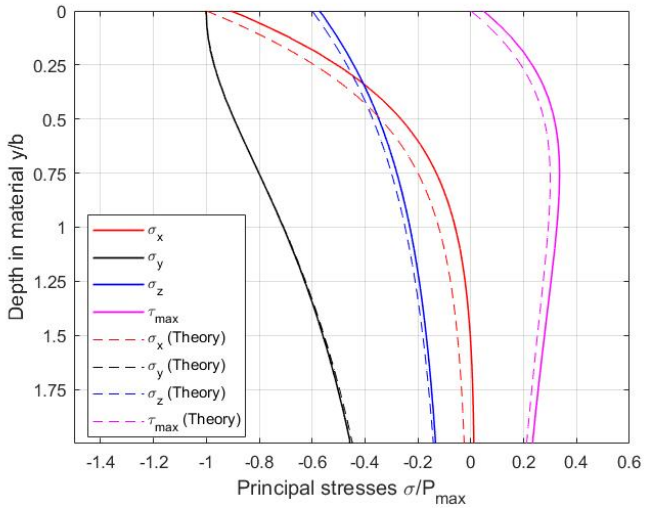


Figure A.14: *Principal centerline stresses in the material compared to the theory*

A.2.3. $70\mu m$

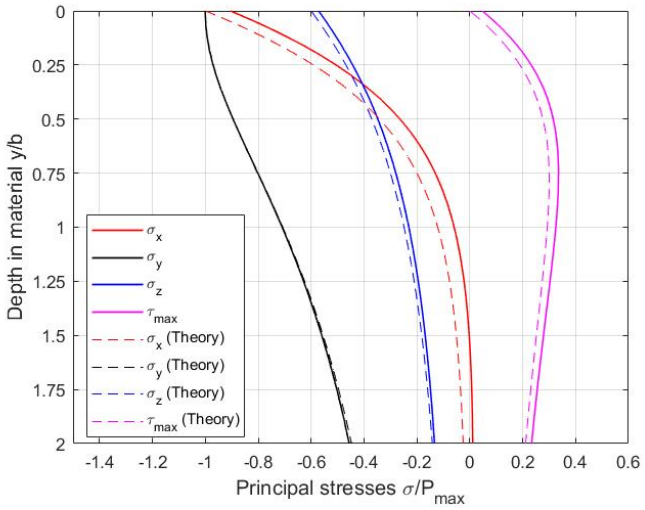


Figure A.15: *Principal centerline stresses in the material compared to the theory*

A.2.4. $80\mu m$

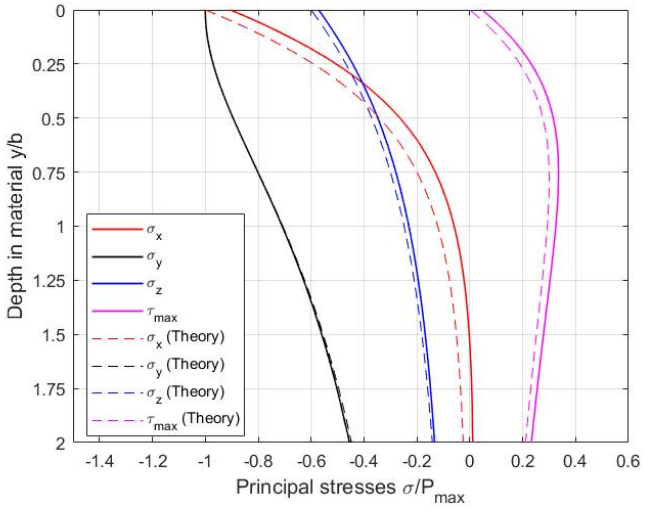


Figure A.16: *Principal centerline stresses in the material compared to the theory*

A.2.5. $90\mu m$

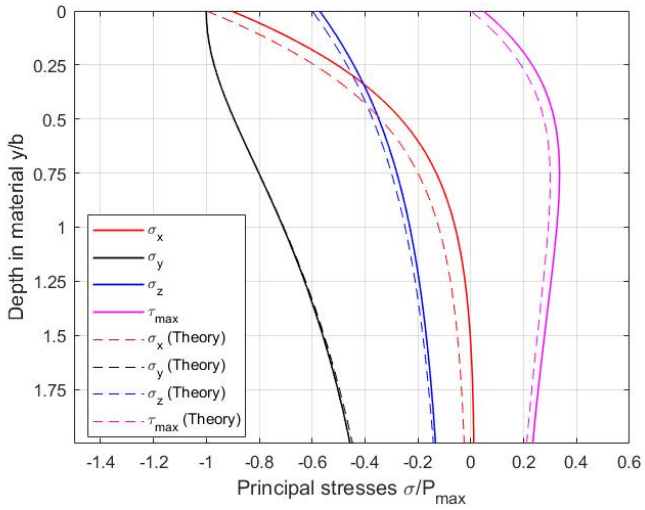


Figure A.17: *Principal centerline stresses in the material compared to the theory*

A.2.6. $100\mu m$

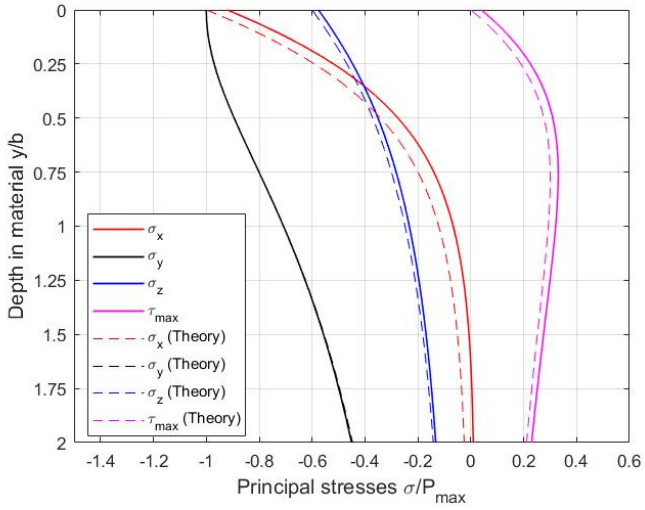


Figure A.18: *Principal centerline stresses in the material compared to the theory*

B

Crack Formation

B.0.1. $50\mu m$

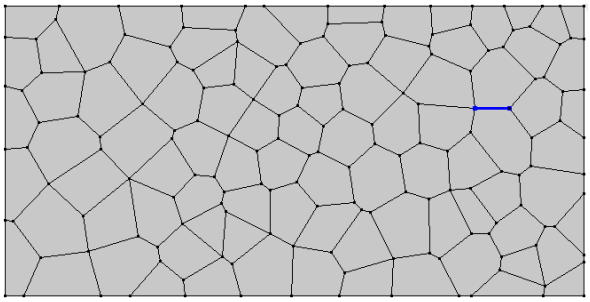


Figure B.1: *Crack initiation after* $2.7510 * 10^{10}$ *cycles*

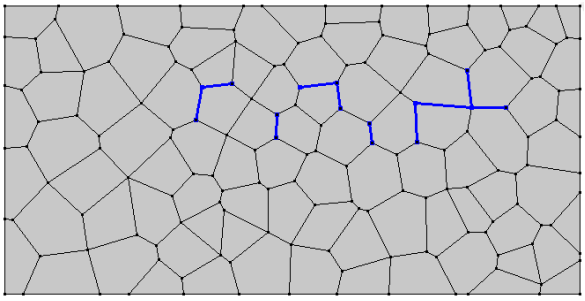


Figure B.2: *Crack formation after* $2.8439 * 10^{10}$ *cycles*

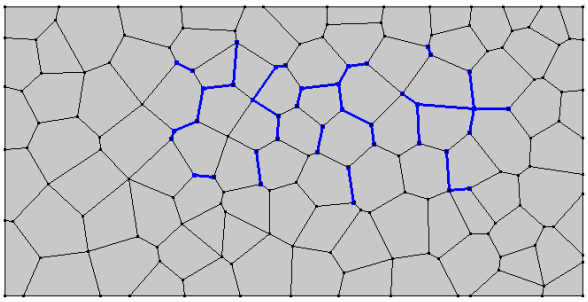


Figure B.3: *Crack formation after* $5.6743 * 10^{10}$ *cycles*

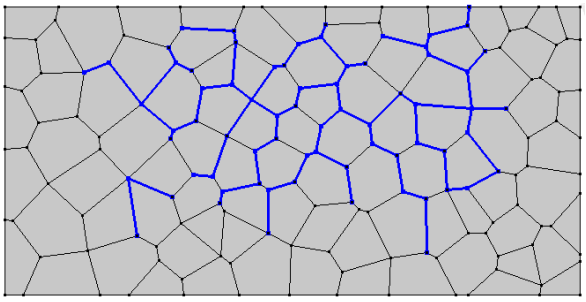


Figure B.4: *Crack formation after* $1.3533 * 10^{11}$ *cycles, failure has occurred*

B.0.2. $60\mu m$

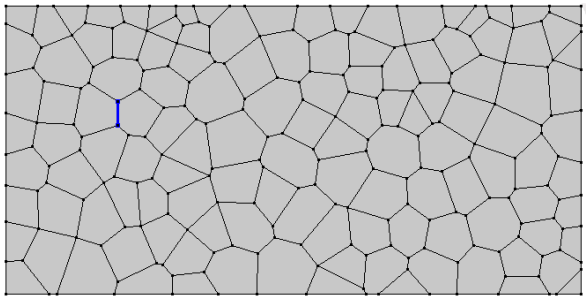


Figure B.5: *Crack initiation after* $2.6101 * 10^{10}$ *cycles*

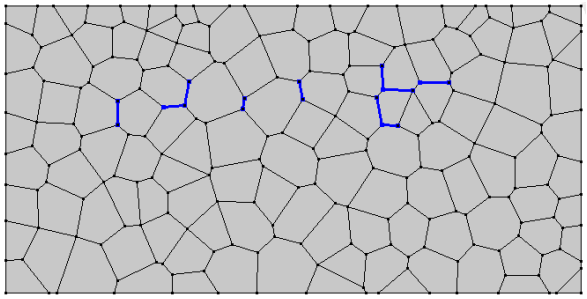


Figure B.6: *Crack formation after* $2.7362 * 10^{10}$ *cycles*

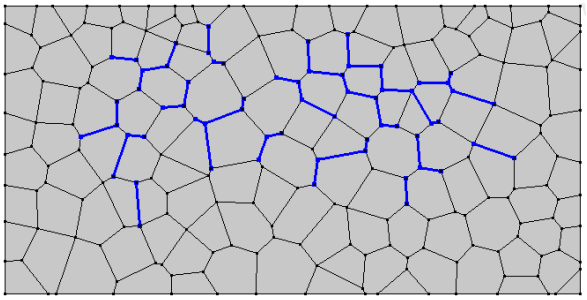


Figure B.7: *Crack formation after* $5.7339 * 10^{10}$ *cycles*

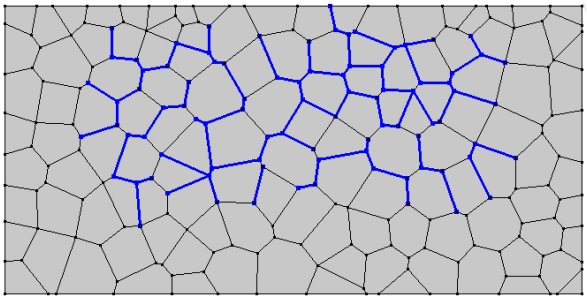


Figure B.8: *Crack formation after* $1.1074 * 10^{11}$ *cycles, failure has occurred*

B.0.3. $70\mu m$

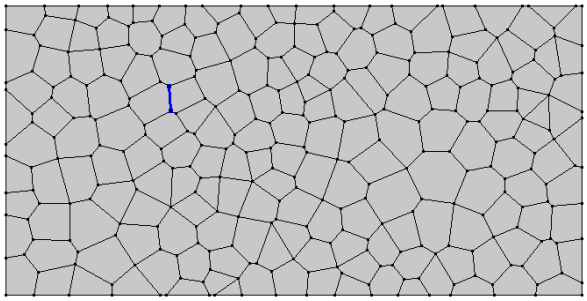


Figure B.9: *Crack initiation after* $2.6479 * 10^{10}$ *cycles*

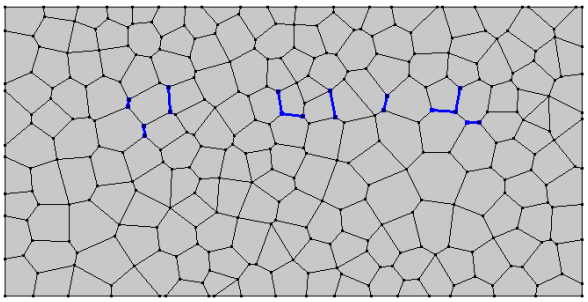


Figure B.10: *Crack formation after* $2.7609 * 10^{10}$ *cycles*

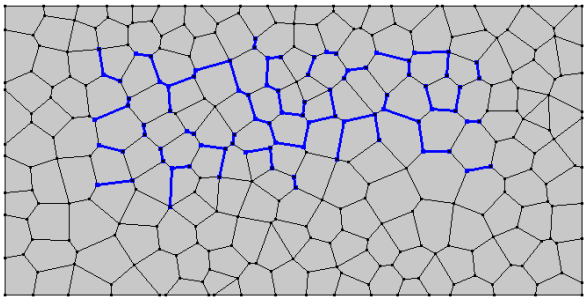


Figure B.11: *Crack formation after* $4.9853 * 10^{10}$ *cycles*

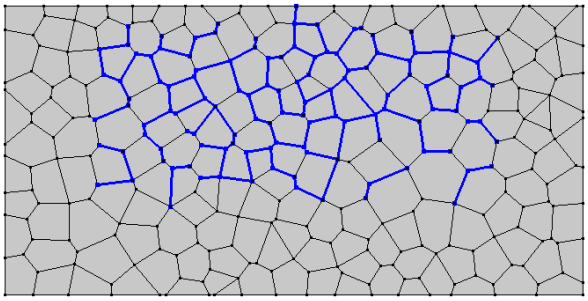


Figure B.12: *Crack formation after* $1.1477 * 10^{11}$ *cycles, failure has occurred*

B.0.4. $80\mu m$

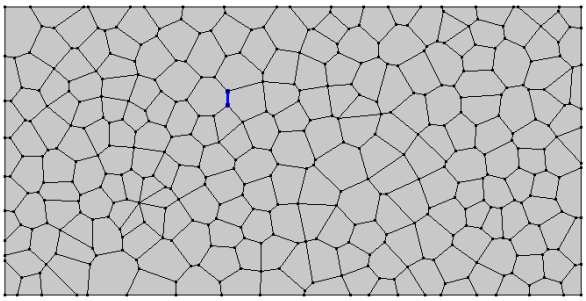


Figure B.13: *Crack initiation after* $2.5125 * 10^{10}$ *cycles*

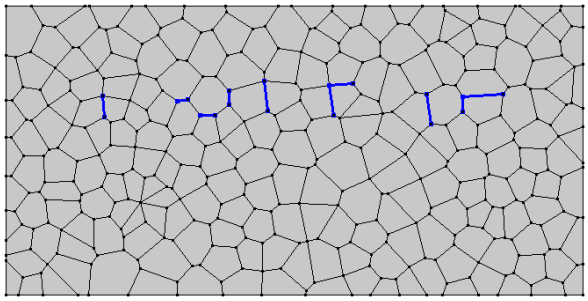


Figure B.14: *Crack formation after* $2.5423 * 10^{10}$ *cycles*

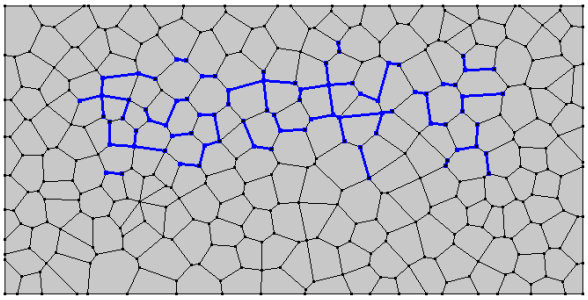


Figure B.15: *Crack formation after* $3.7156 * 10^{10}$ *cycles*

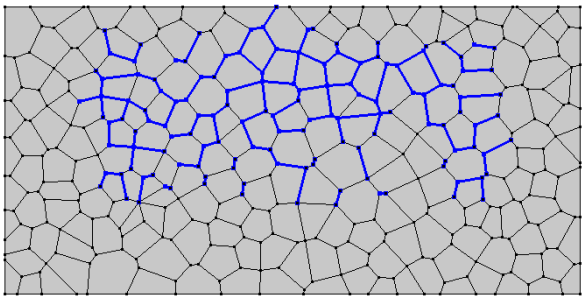


Figure B.16: *Crack formation after* $6.9289 * 10^{10}$ *cycles, failure has occurred*

B.0.5. $90\mu m$

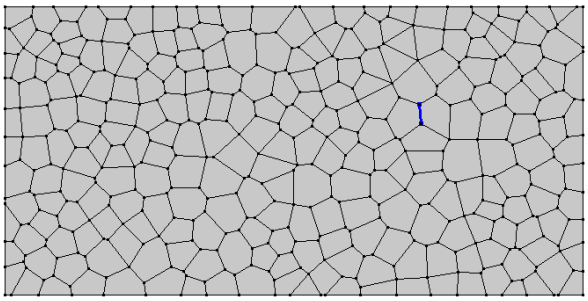


Figure B.17: *Crack initiation after* $2.6588 * 10^{10}$ *cycles*

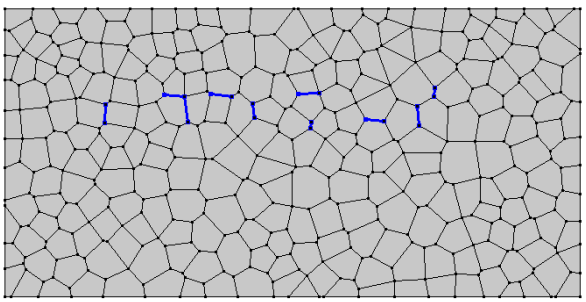


Figure B.18: *Crack formation after* $2.6691 * 10^{10}$ *cycles*

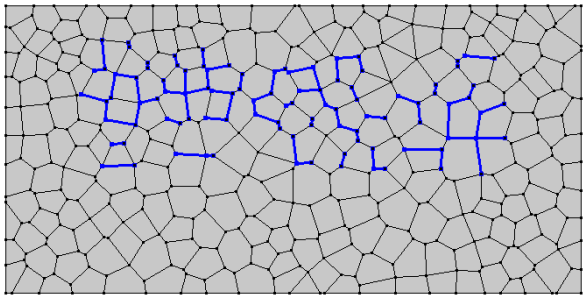


Figure B.19: *Crack formation after* $3.6985 * 10^{10}$ *cycles*

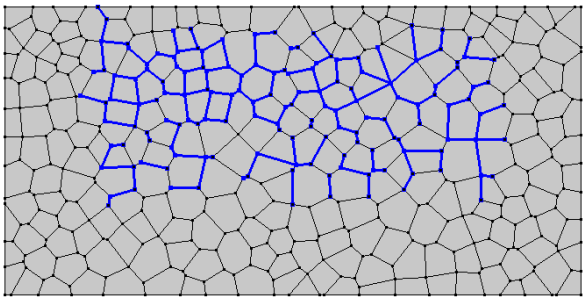


Figure B.20: *Crack formation after* $5.7763 * 10^{10}$ *cycles, failure has occurred*

B.0.6. $100\mu m$

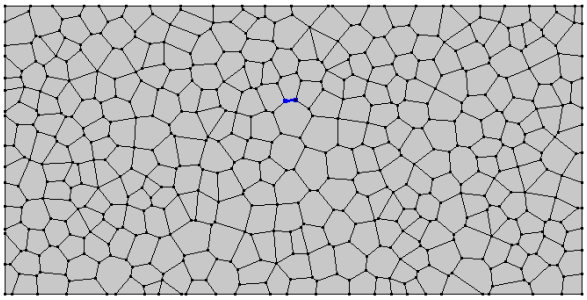


Figure B.21: *Crack initiation after* $2.6193 * 10^{10}$ *cycles*

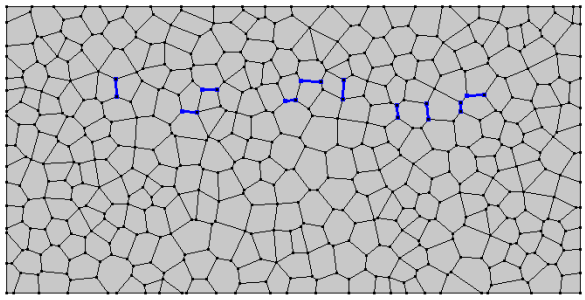


Figure B.22: *Crack formation after* $2.6319 * 10^{10}$ *cycles*

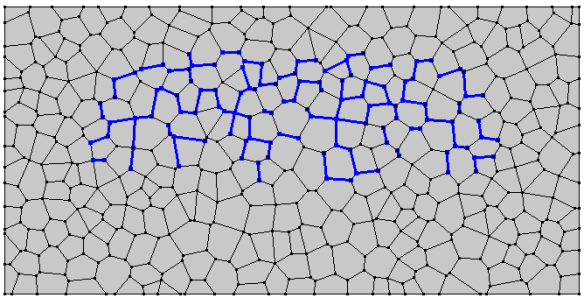


Figure B.23: *Crack formation after* $3.9065 * 10^{10}$ *cycles*

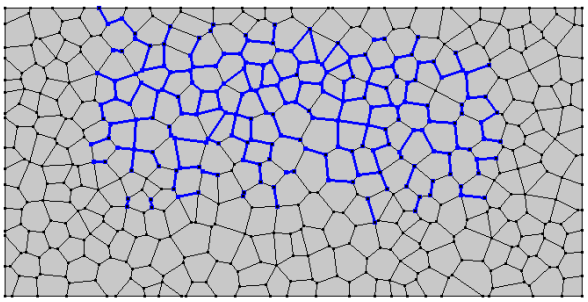


Figure B.24: *Crack formation after* $6.9734 * 10^{10}$ *cycles, failure has occurred*

C

Fatigue Life

C.1. Individual Plots

C.1.1. $50\mu m$

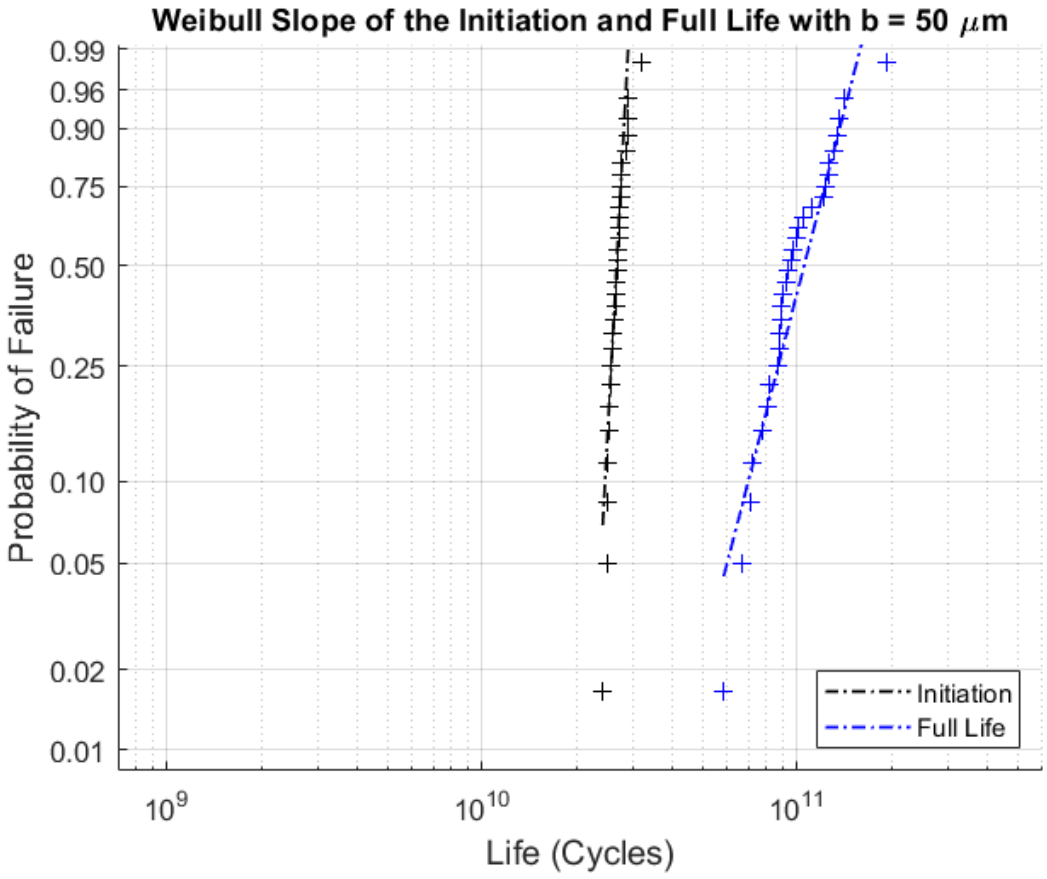


Figure C.1: *Weibull plot* $b = 50\mu m$

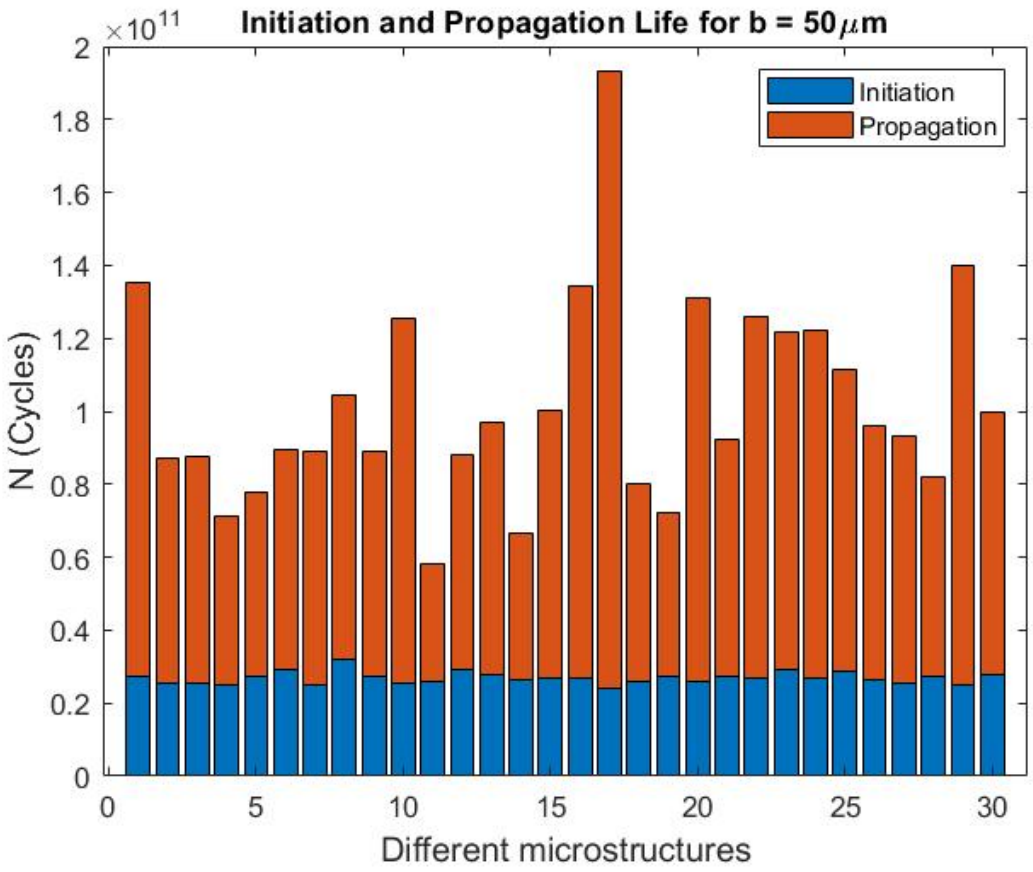


Figure C.2: *Fatigue lives for the different microstructures with* $b = 50\mu m$

C.1.2. $60\mu m$

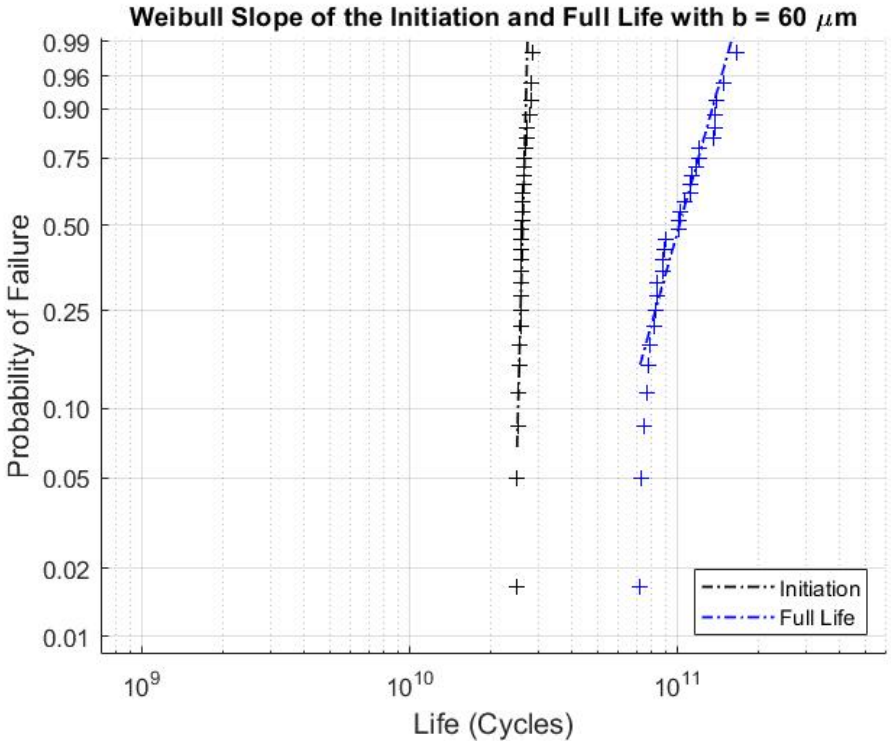


Figure C.3: *Weibull plot* $b = 60\mu m$

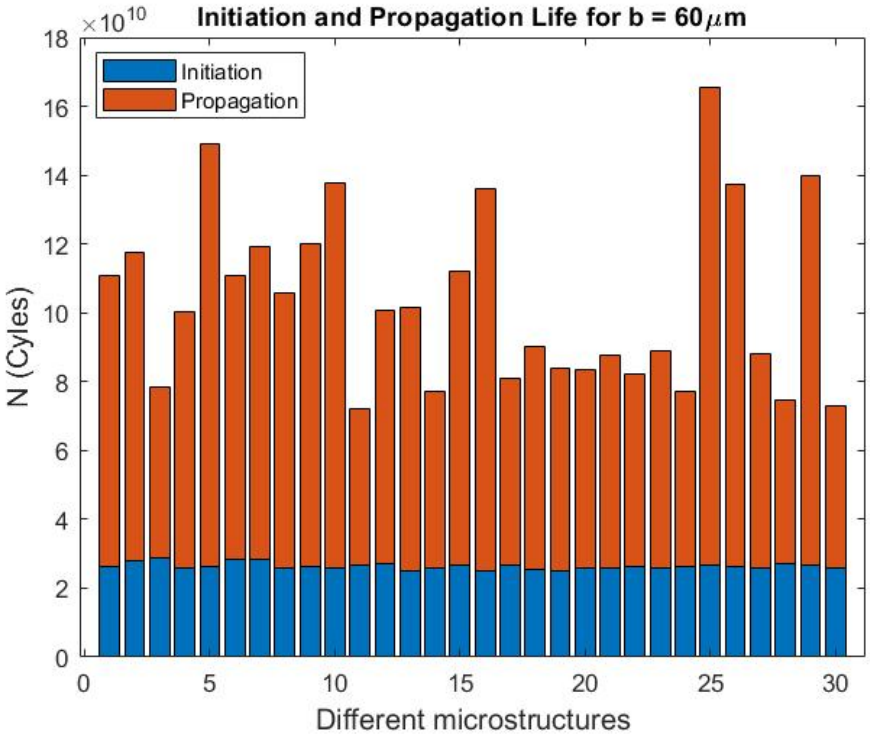


Figure C.4: *Fatigue lives for the different microstructures with* $b = 60\mu m$

C.1.3. $70\mu m$

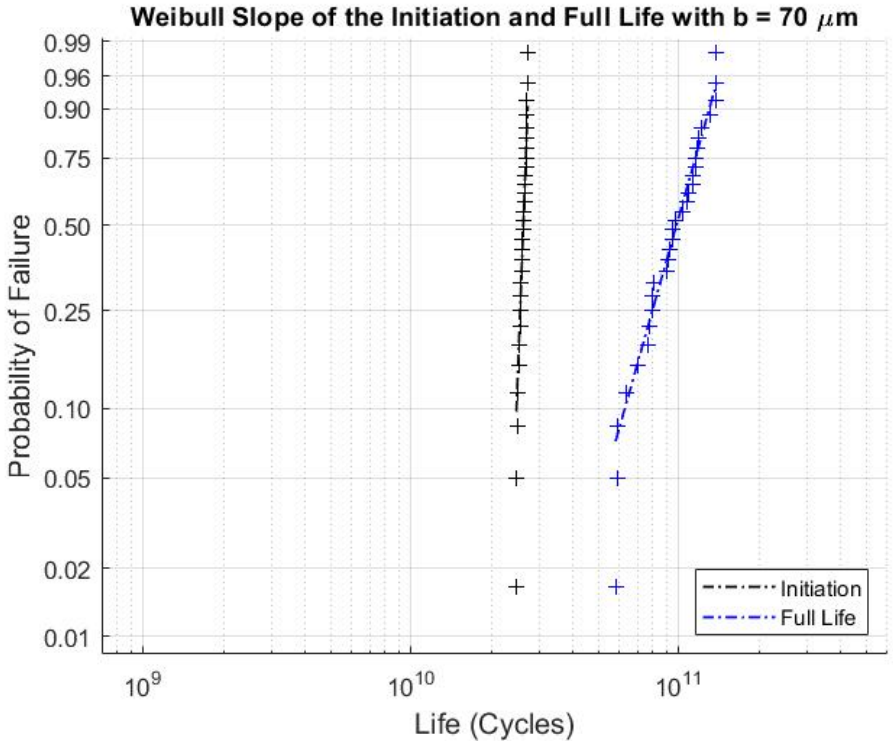


Figure C.5: *Weibull plot $b = 70\mu m$*

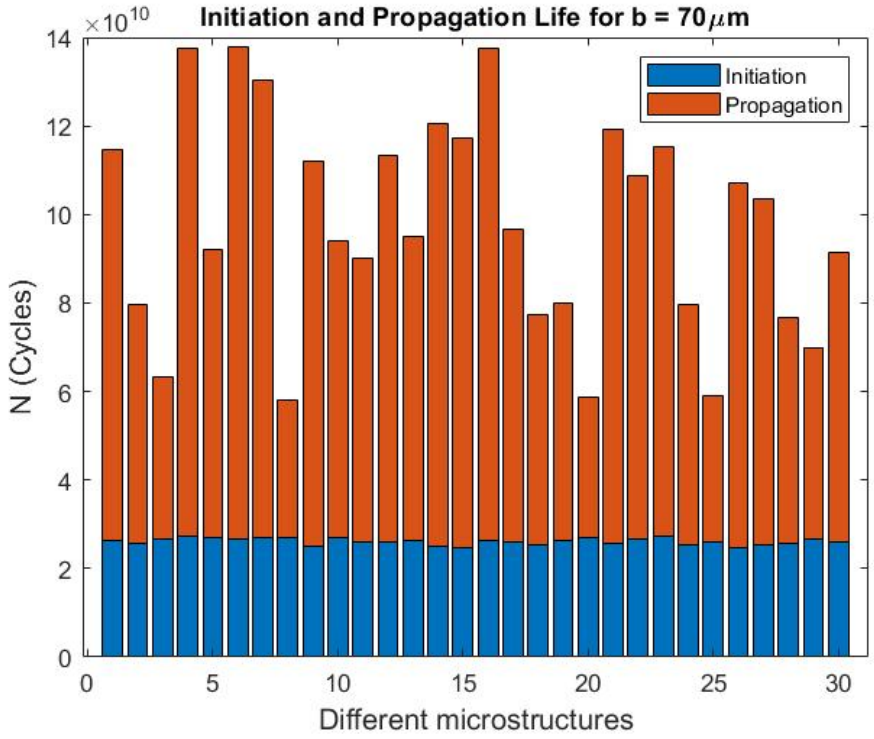


Figure C.6: *Fatigue lives for the different microstructures with $b = 70\mu m$*

C.1.4. $80\mu m$

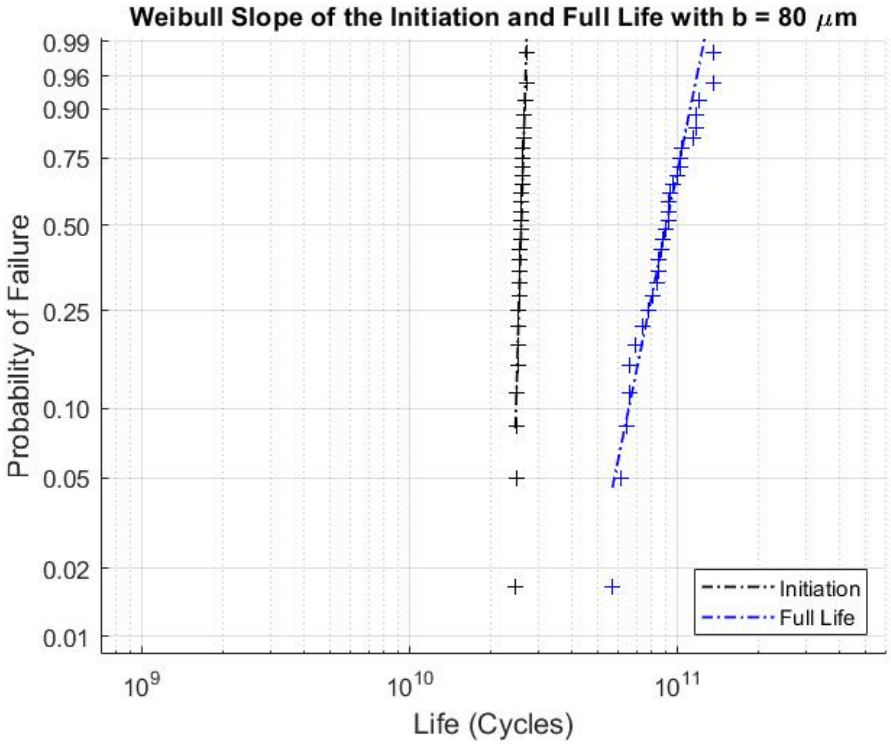


Figure C.7: *Weibull plot* $b = 80\mu m$

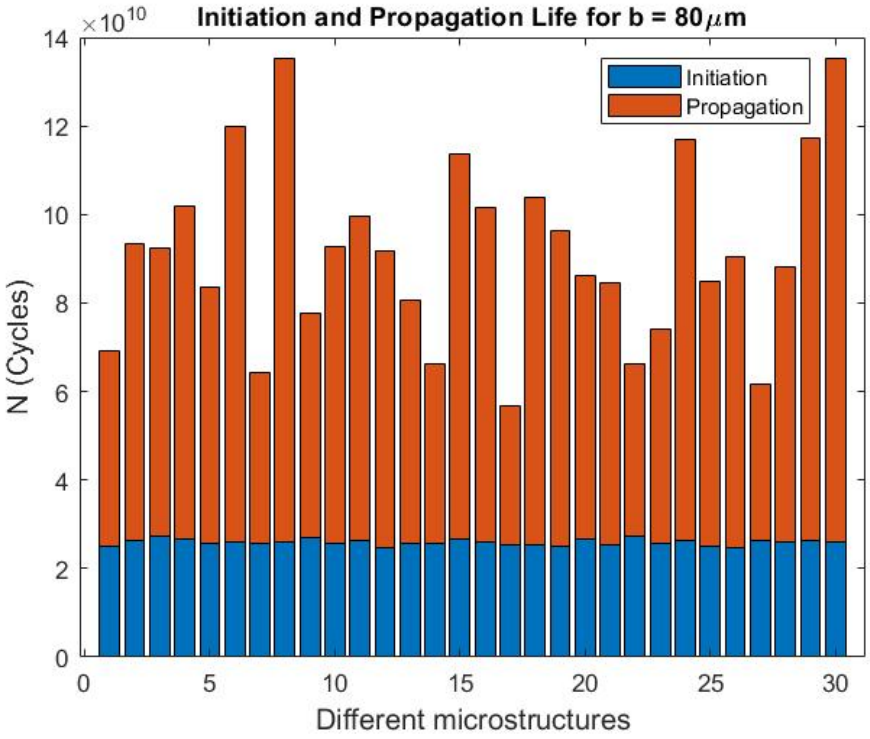


Figure C.8: *Fatigue lives for the different microstructures with* $b = 80\mu m$

C.1.5. $90\mu m$

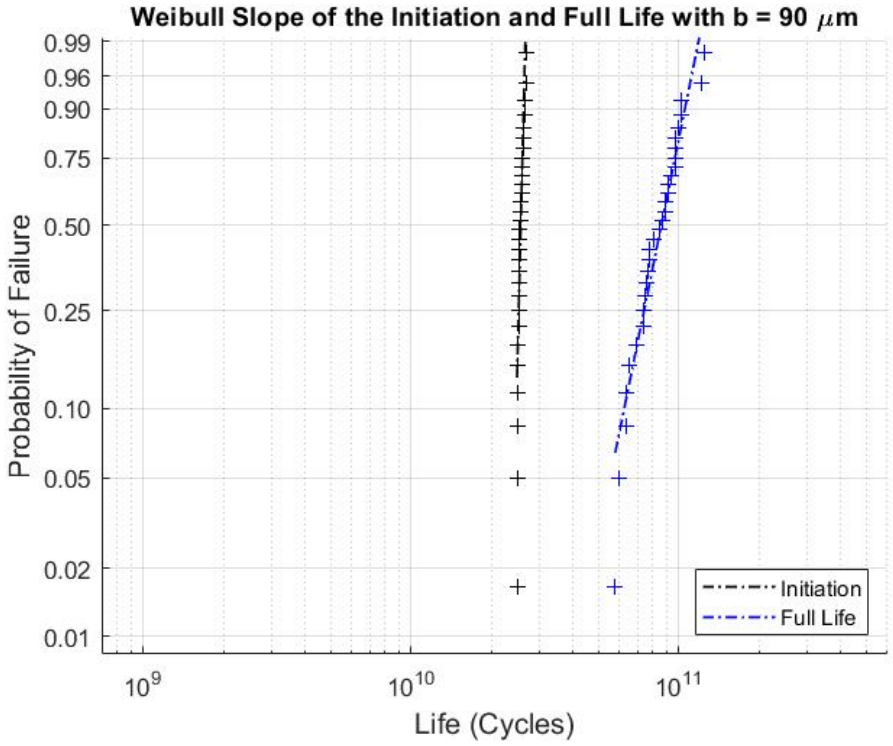


Figure C.9: *Weibull plot $b = 90\mu m$*

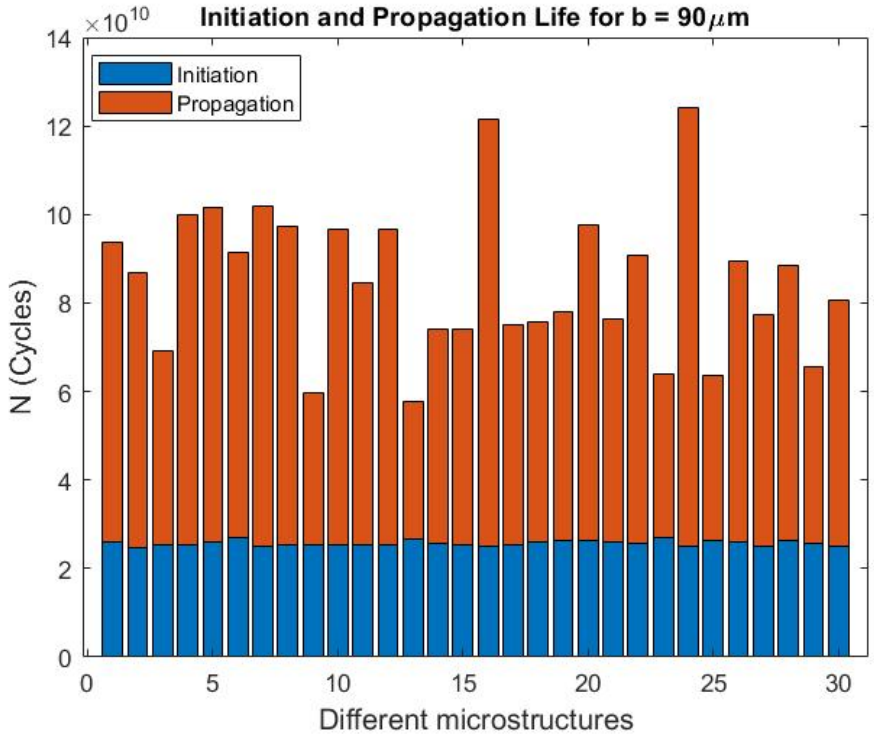


Figure C.10: *Fatigue lives for the different microstructures with $b = 90\mu m$*

C.1.6. $100\mu m$

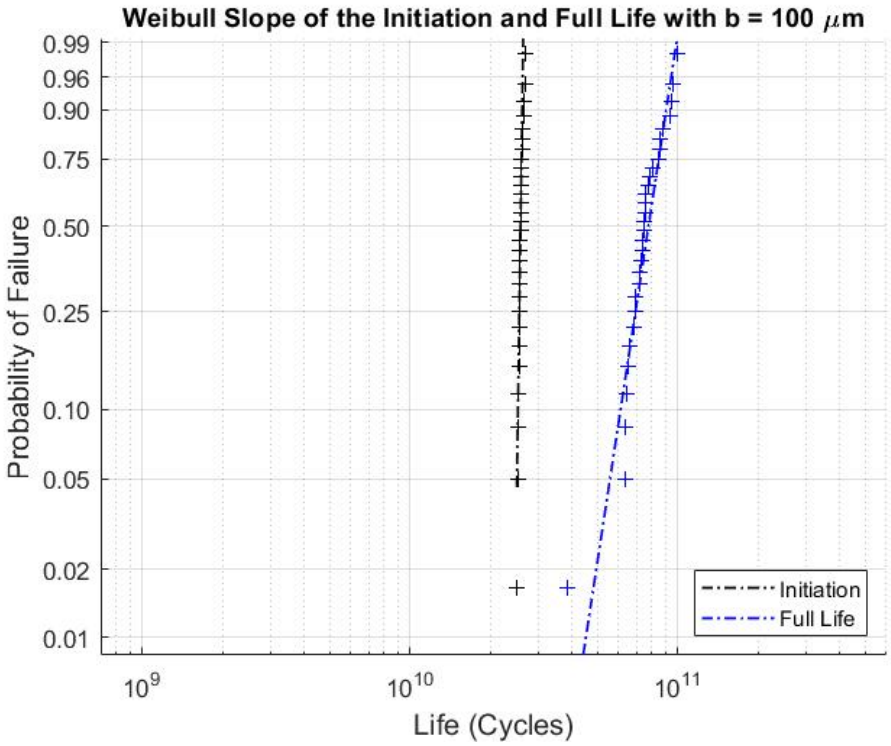


Figure C.11: *Weibull plot* $b = 100\mu m$

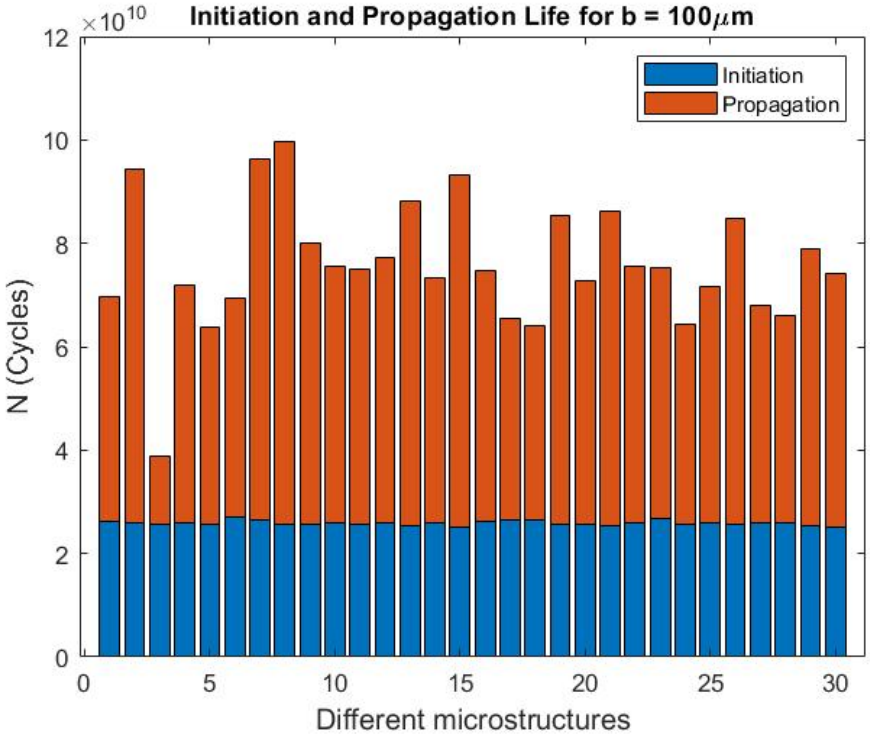


Figure C.12: *Fatigue lives for the different microstructures with* $b = 100\mu m$

C.2. Combined Plots

C.2.1. Initiation Life of All Structures

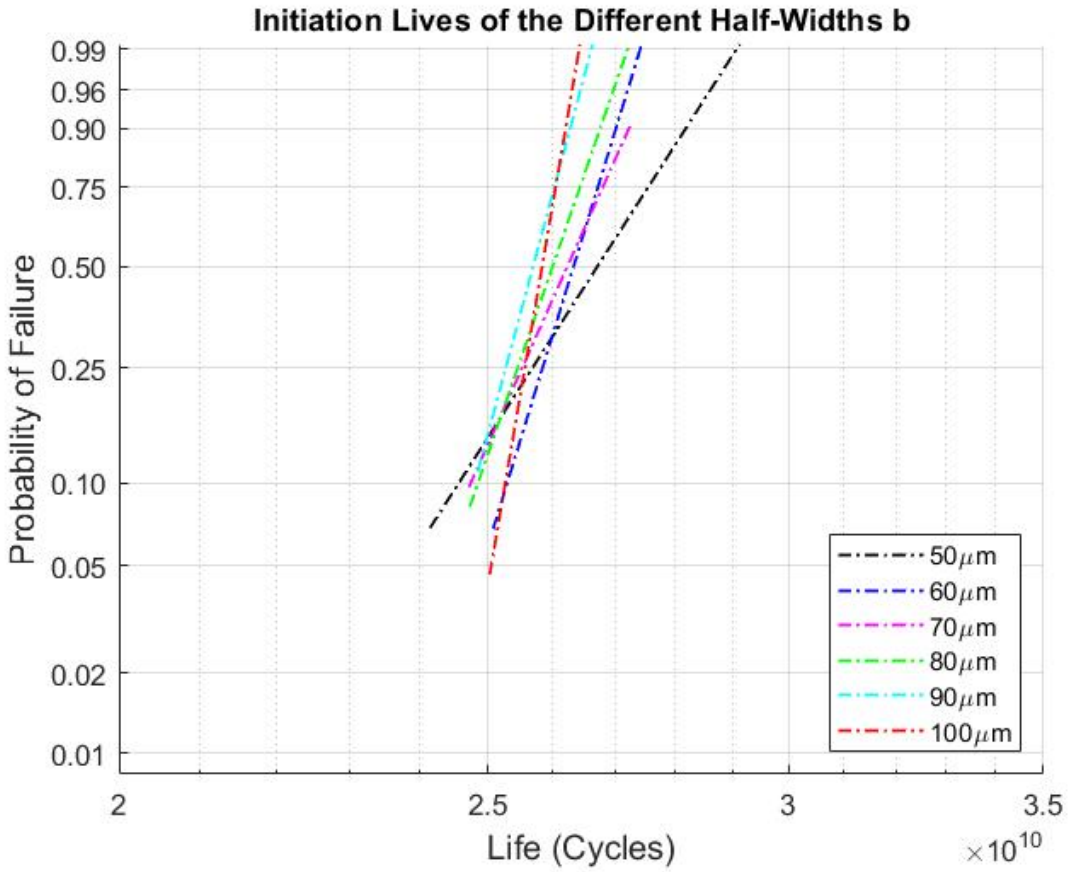


Figure C.13: *Initiation life Weibull plots of all the half-widths*

C.2.2. Full Life of All Structures

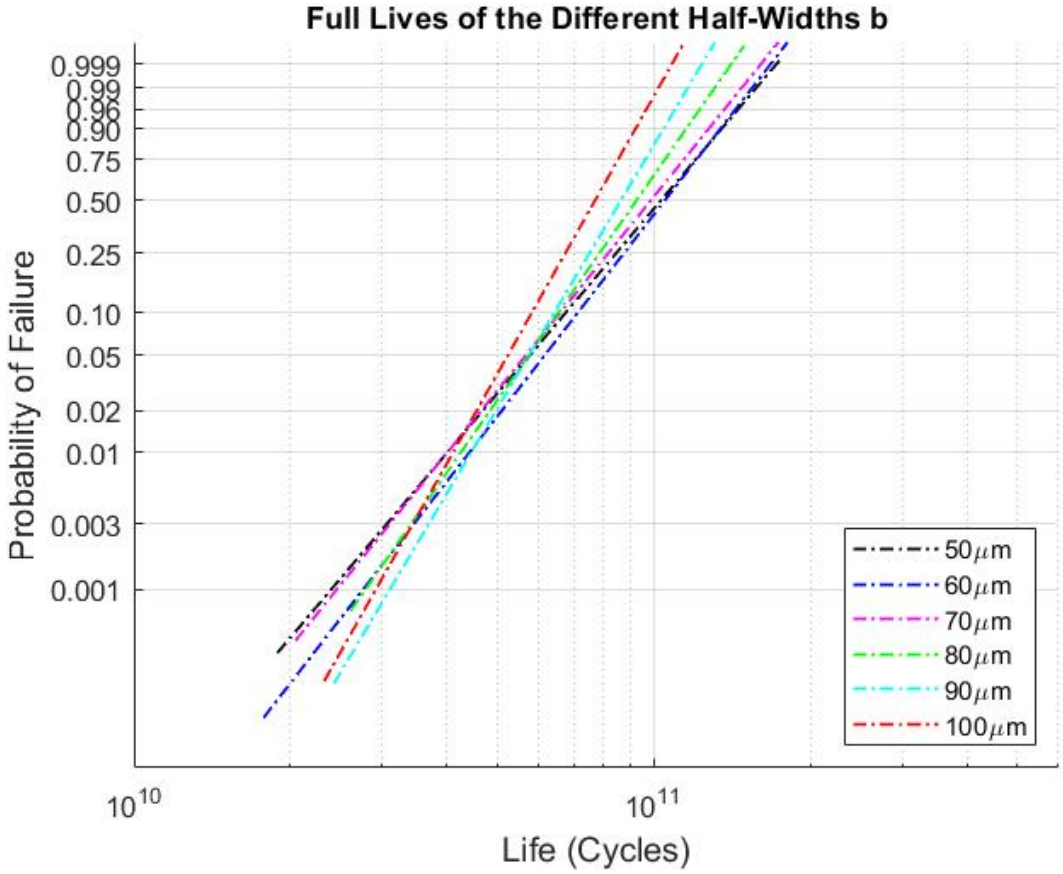


Figure C.14: *Full Life Weibull plots of all the half-widths*

D

Simulation Differences

D.1. Loadcycle Progression

D.1.1. $50\mu m$

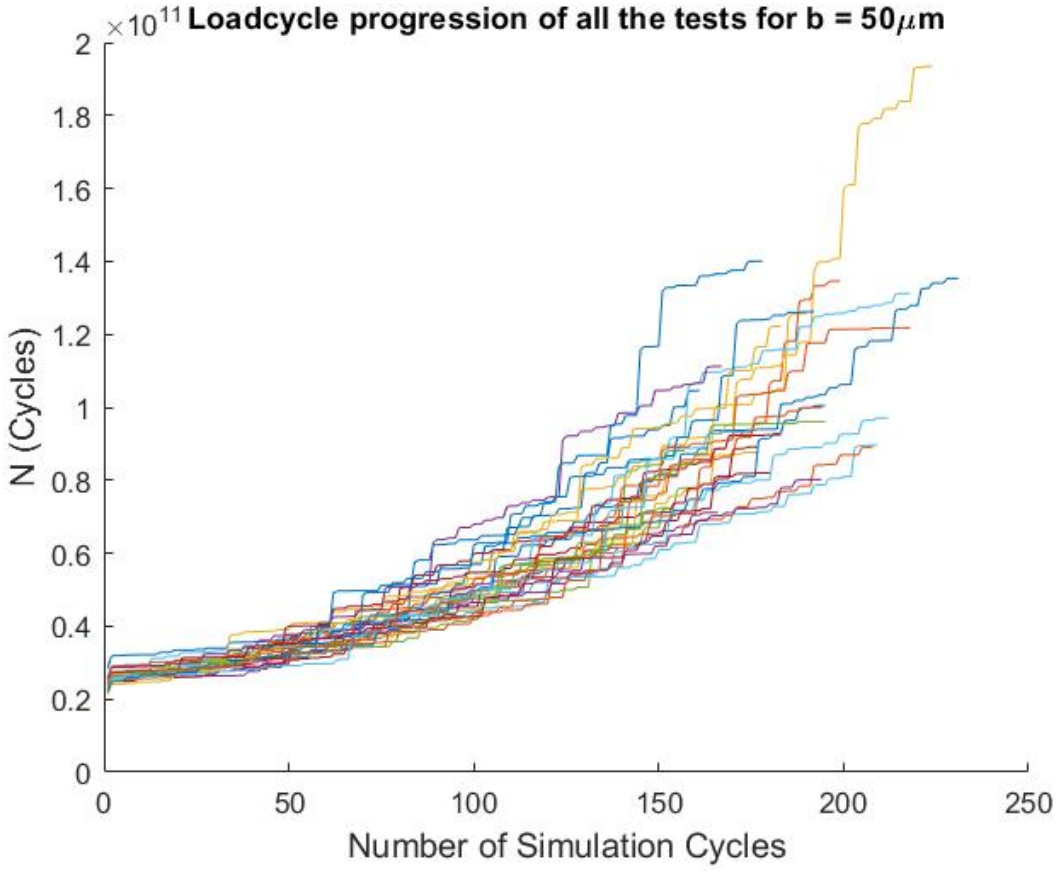


Figure D.1: *Loadcycle progression of the different microstructures with* $b = 50\mu m$

D.1.2. $60\mu m$

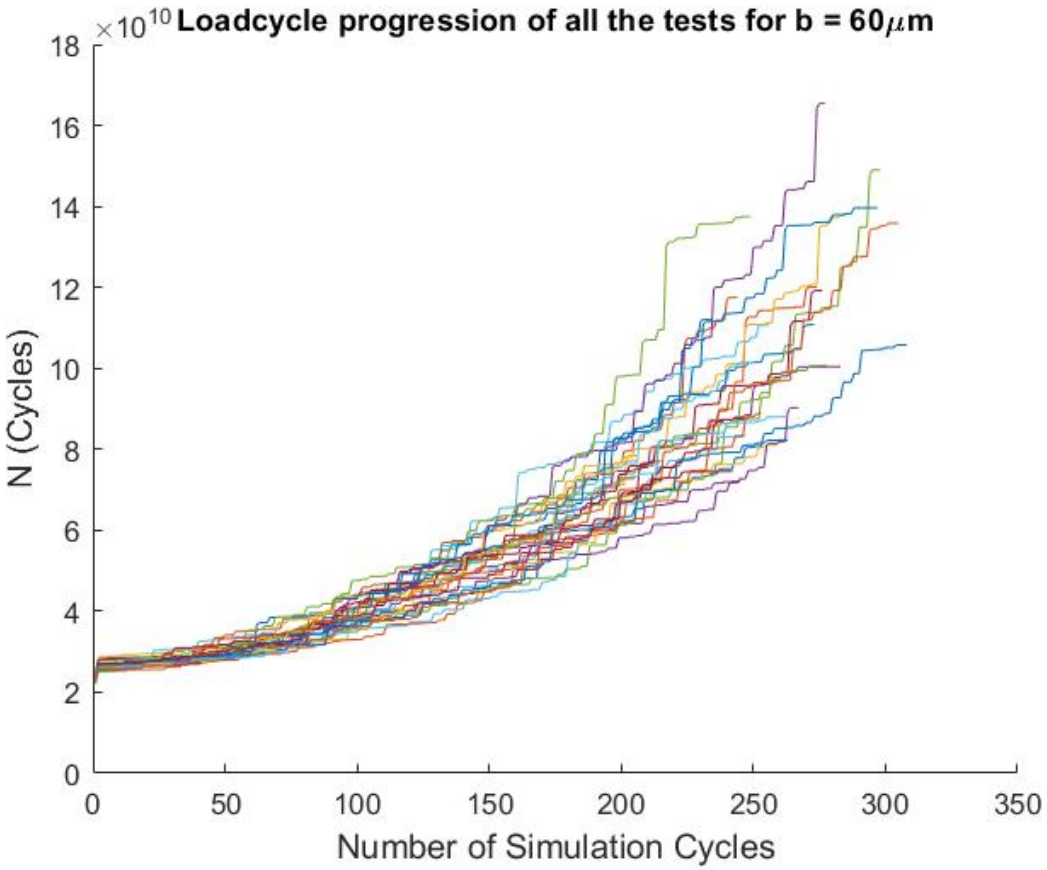


Figure D.2: *Loadcycle progression of the different microstructures with* $b = 60\mu m$

D.1.3. $70\mu m$

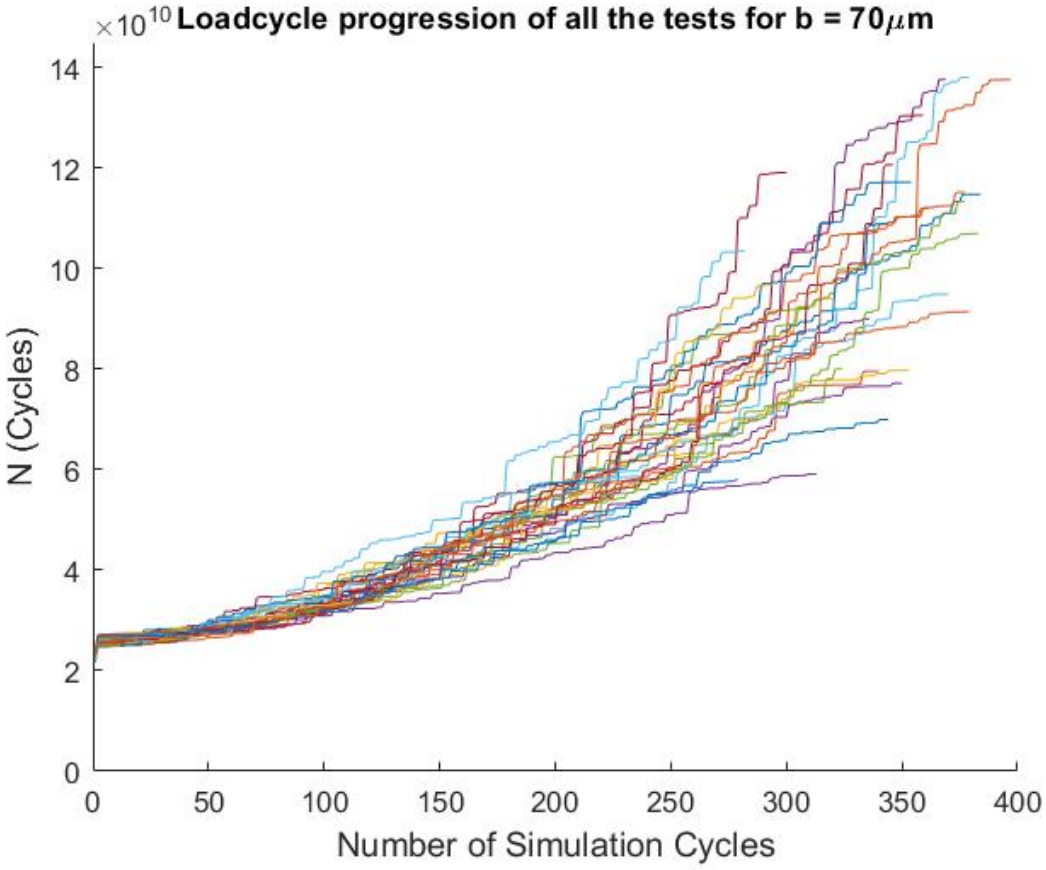


Figure D.3: *Loadcycle progression of the different microstructures with* $b = 70\mu m$

D.1.4. $80\mu m$

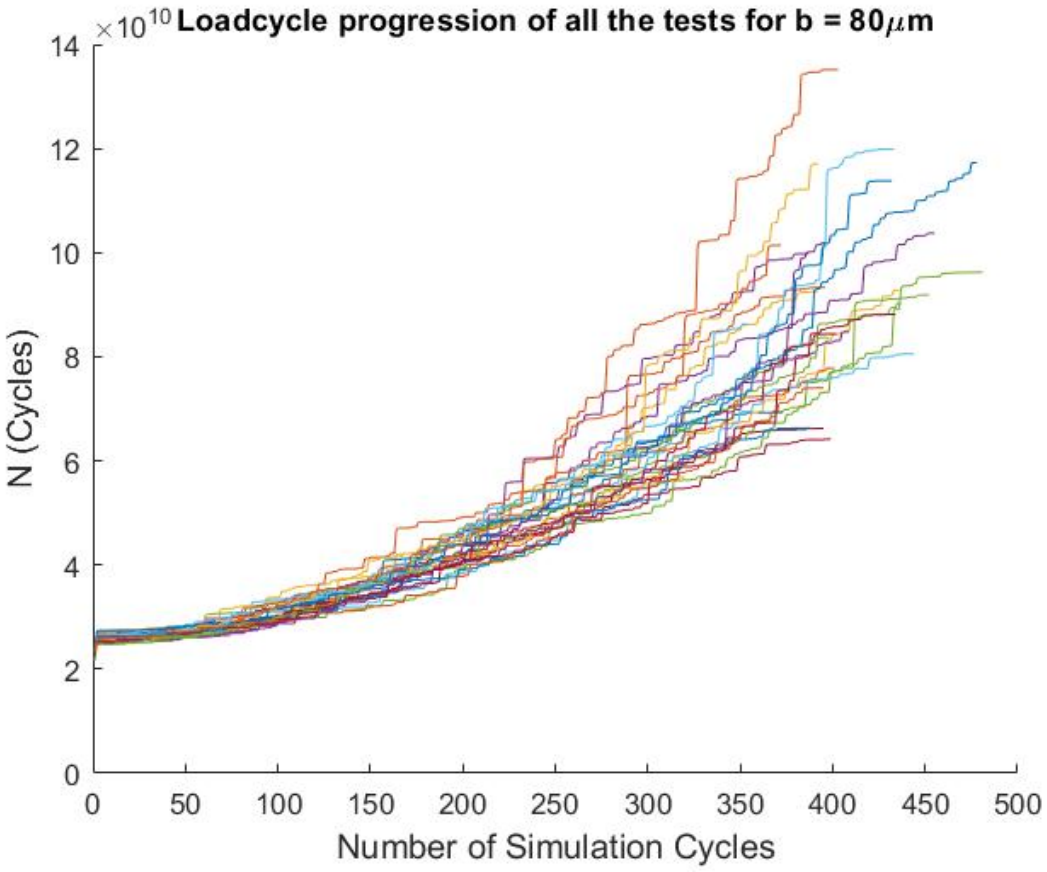


Figure D.4: *Loadcycle progression of the different microstructures with* $b = 80\mu m$

D.1.5. $90\mu m$

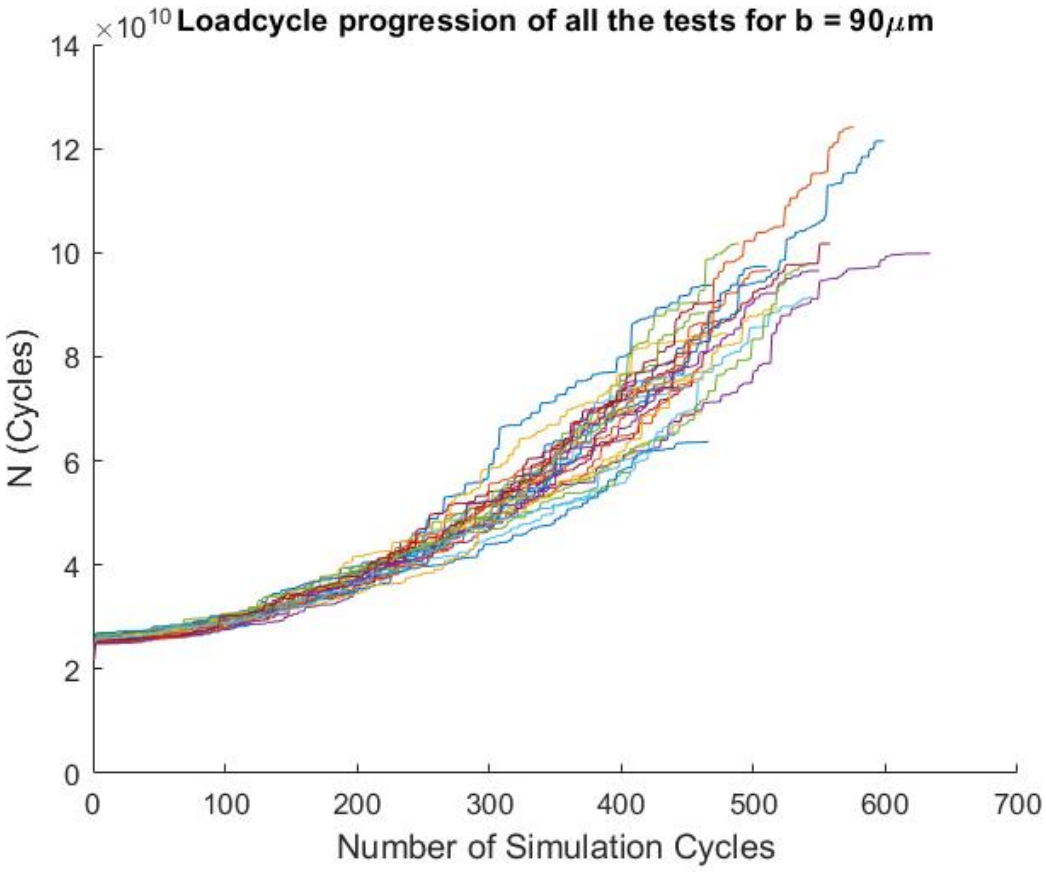


Figure D.5: *Loadcycle progression of the different microstructures with* $b = 90\mu m$

D.1.6. $100\mu m$

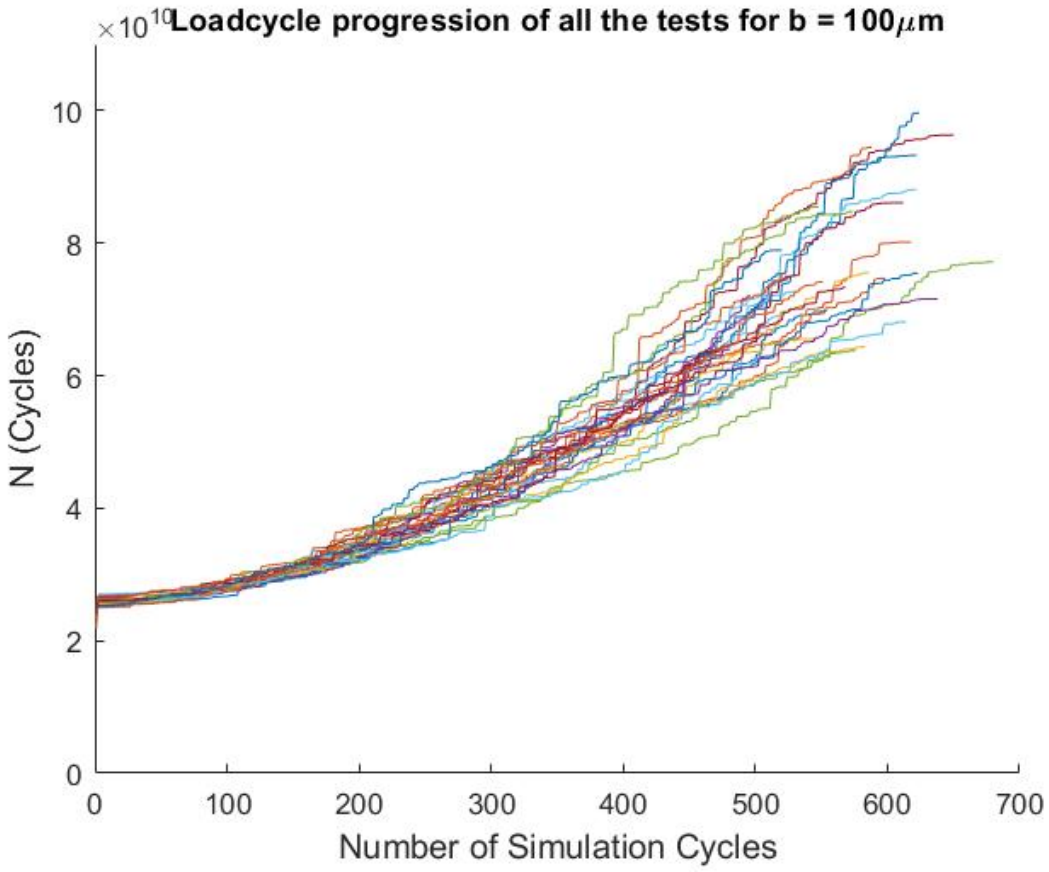


Figure D.6: *Loadcycle progression of the different microstructures with* $b = 100\mu m$

D.2. Computational Effort

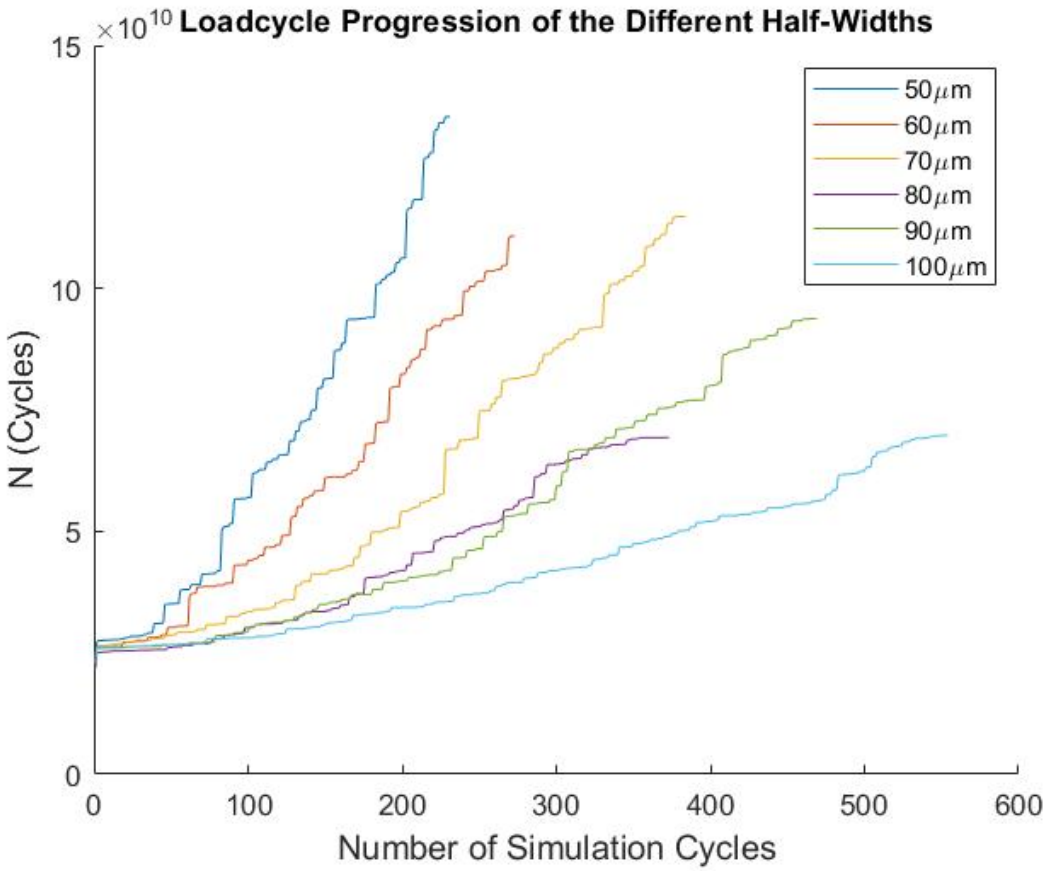


Figure D.7: *Loadcycle progression of the different half-widths*

Half-Width b (μm)	Average Time per Cycle (s)	Average Amount of Cycles (Cycles)	Average Total Time (h)
50	20	185	1.03
60	28	265	2.06
70	34	344	3.24
80	45	401	5.02
90	60	490	8.17
100	75	574	11.96

Table D.1: *Computational effort required to simulate the full fatigue life of different microstructures*

E

Building the Model

E.1. Microstructure Topology Model

E.1.1. Point Selection

```
%% Dimensions of domain
Width  = 3*(b*10^6);
Height = 1.5*(b*10^6);
Box_Width  = 10*(b*10^6);
Box_Height =  6*(b*10^6);

% Distribution
Mean_Grain_Diameter = 10;
Distribution        = 2.5;

% Initial points placed in domain
n_Points = 280000;
xPoint = Width*rand(n_Points,1);
yPoint = Height*rand(n_Points,1);
Random_Points = [xPoint,yPoint];

% Define counter for how many actually get placed.
numPoints = 0;

% Define fail safe, how many iterations you want to keep trying before quitting.
maxIterations = 1000 * n_Points;
loopCounter = 1;

% Allocate space to variables
x = nan(numPoints,1);
y = nan(numPoints,1);
Norm_Distance = nan(numPoints,1);

%% Point Selection
while numPoints < n_Points && loopCounter < maxIterations && numPoints < ...
    length(Random_Points)
numPoints = numPoints + 1;
xChosen = Random_Points(numPoints,1);
yChosen = Random_Points(numPoints,2);
x(numPoints,1) = xChosen;
y(numPoints,1) = yChosen;

% The grain diameter of the chosen point
Norm_Distance(numPoints,1) = abs(normrnd(Mean_Grain_Diameter,Distribution));
% Calculate the distance of all points to the chosen point
Distances = sqrt((Random_Points(:,1)-xChosen) .^ 2 + (Random_Points(:,2) - ...
    yChosen) .^ 2);
closePair = Distances ≤ Norm_Distance(numPoints);

```

```
43      Random_Points(closePair == 1,:) = nan; % Remove all points inside certain distance
44      Random_Points = [x(1:numPoints) y(1:numPoints); Random_Points(numPoints+1:end,1) ...
            Random_Points(numPoints+1:end,2)];
45      Random_Points = rmmissing(Random_Points);
46
47      % Increment the loop counter.
48      loopCounter = loopCounter + 1;
49      end
50      P = [x,y];
```

E.1.2. Mirror Points

```
1       %% Mirror all the points
2       Right = [2*Width - x ,            y];
3       Left  = [0       - x ,            y];
4       Up    = [          x , 2*Height - y];
5       Low   = [          x , 0        - y];
6       P_Mirror = [P;Right;Left;Up;Low];
7       [v,c] = voronoin(P_Mirror);
```

E.1.3. Removing Small Edges

```
1         %% Remove small vertices for better mesh
2       min_Dist = 2;
3       cnt = 1;
4       tol = 1.e-10;
5       rnd = 5;
6
7       edges = [0,Width,Height];
8       LLcorner = [0,0];
9       LUcorner = [0,Height];
10      RLcorner = [Width,0];
11      RUcorner = [Width,Height];
12
13      for i = 1:length(P)
14      % All the points of the grain
15      VPoint = v(c{i},:);
16      c_new  = c{i};
17      cntB = 0;
18      % Distance between the points
19      Dist = pdist(VPoint);
20      % Find points that are too close to each other
21      if any(Dist < min_Dist)
22      cnt = cnt + 1;
23      % Make square out of Dist
24      Z = squareform(Dist);
25      % row and col
26      [row,col] = find(Z < min_Dist & Z ≠ 0);
27      ClosePoints = VPoint(col,:);
28      [¬,idx] = unique(sort(ClosePoints,2),'rows','stable');
29      ClosePoints = ClosePoints(idx,:);
30      repCol = ismember(VPoint,ClosePoints,'rows');
31      B = [row,col];
32      [¬,idx] = unique(sort(B,2),'rows','stable');
33      B = B(idx,:);
34      row = B(:,1);
35      col = B(:,2);
36      for o = 1:size(B,1)
37      ClP = round(VPoint(B(o,:)',:),rnd);
38      %% If VPoint has edge members
39      if any(any(ismembertol(ClP,edges)))
40      %% Check if VPoint has corner points
41      % Left Lower Corner %
42      if any(ismembertol(ClP,LLcorner,'ByRow',true))
43      valChange = not(ismembertol(ClP,LLcorner,'ByRow',true));
44      CloseP = ClP(valChange,:);
45      if ismembertol(CloseP(1,2),0)          % Is it on the x-axis?
```

```
46      CloseP = [min_Dist, 0];
47      else                                % Is it on the y-axis?
48      CloseP = [0, min_Dist];
49      end
50      ClP(valChange,:) = CloseP;
51      VPoint(B(o,:)',:) = ClP;
52      % Left Upper Corner %
53      elseif any(ismembertol(ClP,LUcorner,'ByRow',true))
54      valChange = not(ismembertol(ClP,LUcorner,'ByRow',true));
55      CloseP = ClP(valChange,:);
56      if ismembertol(CloseP(1,2),Height)    % Is it on the x-axis?
57      CloseP = [min_Dist, Height];
58      else                                % Is it on the y-axis?
59      CloseP = [0, Height - min_Dist];
60      end
61      ClP(valChange,:) = CloseP;
62      VPoint(B(o,:)',:) = ClP;
63      % Right Lower Corner %
64      elseif any(ismembertol(ClP,RLcorner,'ByRow',true))
65      valChange = not(ismembertol(ClP,RLcorner,'ByRow',true));
66      CloseP = ClP(valChange,:);
67      if ismembertol(CloseP(1,2),0)         % Is it on the x-axis?
68      CloseP = [Width - min_Dist, 0];
69      else                                % Is it on the y-axis?
70      CloseP = [Width, min_Dist];
71      end
72      ClP(valChange,:) = CloseP;
73      VPoint(B(o,:)',:) = ClP;
74      % Right Upper Corner %
75      elseif any(ismembertol(ClP,RUcorner,'ByRow',true))
76      valChange = not(ismembertol(ClP,RUcorner,'ByRow',true));
77      CloseP = ClP(valChange,:);
78      if ismembertol(CloseP(1,2),Height)    % Is it on the x-axis?
79      CloseP = [Width - min_Dist, Height];
80      else                                % Is it on the x-axis?
81      CloseP = [Width, Height - min_Dist];
82      end
83      ClP(valChange,:)  = CloseP;
84      VPoint(B(o,:)',:) = ClP;
85      % Check if adjacent or on edge %
86      %% Bottom edge y = 0
87      elseif any(ismembertol(ClP(:,2),0,'ByRow',true))
88      m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
89      d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
90      t = min_Dist/d;
91      if abs(m) < tol % on line
92      [maxClP,maxvalClP] = max(ClP(:,1));
93      [minClP,minvalClP] = min(ClP(:,1));
94      ClP(maxvalClP,1) = ClP(minvalClP,1) + min_Dist;
95      elseif abs(m) == Inf % Perpendicular to line
96      valChange = not(ismembertol(ClP(:,2),0,'ByRow',true));
97      ClP(valChange,2) = ClP(not(valChange),2) + min_Dist;
98      else
99      valChange = not(ismembertol(ClP(:,2),0,'ByRow',true));
100     ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
101     ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
102     end
103     VPoint(B(o,:)',:) = ClP;
104     %% Left edge   x = 0
105     elseif any(ismembertol(ClP(:,1),0,'ByRow',true))
106     m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
107     d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
108     t = min_Dist/d;
109     if abs(m) == Inf  % on line
110     [maxClP,maxvalClP] = max(ClP(:,2));
111     [minClP,minvalClP] = min(ClP(:,2));
112     ClP(maxvalClP,2) = ClP(minvalClP,2) + min_Dist;
113     elseif abs(m) < tol % Perpendicular to line
114     valChange = not(ismembertol(ClP(:,1),0,'ByRow',true));
115     ClP(valChange,1) = ClP(not(valChange),1) + min_Dist;
116     else
```

```
117     valChange = not(ismembertol(ClP(:,1),0,'ByRow',true));
118     ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
119     ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
120     end
121     VPoint(B(o,:)',:) = ClP;
122     %% Top edge y = Height
123     elseif any(ismembertol(ClP(:,2),Height,'ByRow',true))
124     m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
125     d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
126     t = min_Dist/d;
127     if abs(m) < tol % on line
128     [maxClP,maxvalClP] = max(ClP(:,1));
129     [minClP,minvalClP] = min(ClP(:,1));
130     ClP(maxvalClP,1) = ClP(minvalClP,1) + min_Dist;
131     elseif abs(m) == Inf % Perpendicular to line
132     valChange = not(ismembertol(ClP(:,2),Height,'ByRow',true));
133     ClP(valChange,2) = ClP(not(valChange),2) - min_Dist;
134     else
135     valChange = not(ismembertol(ClP(:,2),Height,'ByRow',true));
136     ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
137     ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
138     end
139     VPoint(B(o,:)',:) = ClP;
140     %% Right edge x = Width
141     elseif any(ismembertol(ClP(:,1),Width,'ByRow',true))
142     m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
143     d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
144     t = min_Dist/d;
145     if abs(m) == Inf % on line
146     [maxClP,maxvalClP] = max(ClP(:,2));
147     [minClP,minvalClP] = min(ClP(:,2));
148     ClP(maxvalClP,2) = ClP(minvalClP,2) + min_Dist;
149     elseif abs(m) < tol % Perpendicular to line
150     valChange = not(ismembertol(ClP(:,1),Width,'ByRow',true));
151     ClP(valChange,1) = ClP(not(valChange),1) - min_Dist;
152     else
153     valChange = not(ismembertol(ClP(:,1),Width,'ByRow',true));
154     ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
155     ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
156     end
157     VPoint(B(o,:)',:) = ClP;
158     end
159     %% ClP has no edge values
160     % Does it have multiple small lines?
161     elseif sum(any(ismember(B,B(o,:)),2)) > 1 % If true, close lines adjacent
162     cntB = cntB + 1;
163     if cntB > 1
164     continue
165     end
166     [C,ia,ic] = unique(B);
167     B_counts = accumarray(ic,1);
168     numCount = [B_counts,C];
169     % Find point in the middle
170     A = C(B_counts == 2);
171     if size(A,1) > 1 % The end points of the adjacent line also within min dist
172     G = VPoint(A,:);
173     Gdist = pdist(G);
174     ZG = squareform(Gdist);
175     [maxValue,linearIndexesOfMaxes] = max(ZG(:));
176     [rowsOfMaxes, colsOfMaxes] = find(ZG == maxValue);
177     VPoint(A(colsOfMaxes),:) = nan;
178     c_keep = c_new(A(not(ismember(A,A(colsOfMaxes)))));
179     c_del  = c_new(A(colsOfMaxes));
180     c_new(A(colsOfMaxes))  = nan;
181     VPoint(A(colsOfMaxes)) = nan;
182     else
183     Q = B(any(ismember(B,B(o,:)),2),:);
184     Q = Q(Q ≠ A);
185     c_keep = c_new(A);
186     c_del  = c_new(Q);
187     c_new(Q) = nan;
```

```
188     VPoint(Q,:) = nan;
189     end
190     for j = 1:length(P)
191     if any(ismember(c{j},c_del)) && j ≠ i % if cell has point(s) to delete
192     c_other = c{j};
193     [La,Lb] = ismember(c_other,c_del);
194     [La1,Lb1] = ismember(c_other,c_keep);
195     if any(ismember(c{j},c_keep)) % if cell has point(s) to keep, only delete values ...
            any(ismember(c{j},c_keep(nonzeros(Lb))))
196     c_other(La) = [];
197     else  % if cell does not have point(s) to keep, only replace values
198     c_other(La) = c_keep;
199     end
200     c{j} = c_other;
201     end
202     end
203     else
204     %% Create new points
205     ClP(2,1) = (ClP(1,1) + ClP(2,1))/ 2;
206     ClP(2,2) = (ClP(1,2) + ClP(2,2))/ 2;
207     % Replace the first point with the new value and delete the other point
208     ClP(1,:) = nan;
209     VPoint(B(o,:)',:) = ClP;
210     % Delete the other c value
211     c_keep = c_new(B(o,2));
212     c_del  = c_new(B(o,1));
213     c_new(B(o,1)) = nan;
214     % Find the other cells with these values
215     for j = 1:length(P)
216     if any(ismember(c{j},c_del)) && j ≠ i % if cell has point(s) to delete
217     c_other = c{j};
218     [La,Lb] = ismember(c_other,c_del);
219     [La1,Lb1] = ismember(c_other,c_keep);
220     if any(ismember(c{j},c_keep(nonzeros(Lb)))) % if cell has point(s) to keep, only ...
            delete values
221     c_other(La) = [];
222     c{j} = c_other;
223     else  % if cell does not have point(s) to keep, only replace values
224     c_other(La) = c_keep(nonzeros(Lb));
225     c{j} = c_other;
226     end
227     end
228     end
229     end
230     % Replace old values
231     c{i} = rmmissing(c_new);
232     v(c{i},:) = rmmissing(VPoint);
233     end
234     end
235     end
```

E.2. COMSOL Model

E.2.1. Create Model

```
1     import com.comsol.model.*
2     import com.comsol.model.util.*
3     model = ModelUtil.create('Model1');
4     comp1 = model.component.create('comp1',true);
5     geom1 = comp1.geom.create('geom1',2);
6     model.component("comp1").geom("geom1").lengthUnit("um");
```

E.2.2. Add Parameters

```
    model.param().set("b", [num2str(b) '[m]']);
    model.param().set("P_max", [num2str(P_max) '[Pa]']);
    model.param().set("step", 1);
    model.param().set("nsteps", nsteps);
```

E.2.3. Create Grains

```
    % Create the grains in COMSOL
    for q = 1:length(P)
    V = v(c{q},:);
    tag = ['pol' num2str(q)];
    geom1.feature.create(tag,'Polygon');
    geom1.feature(tag).set('x', V(:,1)).set('y', V(:,2));
    end
    geom1.run

    % Union of all the grains
    uni1 = geom1.feature.create('uni1','Union');
    uni1.selection('input').all();
    geom1.feature('uni1').selection('input');
```

E.2.4. Creating Computational Domain

```
% Move union
model.component("comp1").geom("geom1").create("mov1", "Move");
model.component("comp1").geom("geom1").feature("mov1").selection("input").set("uni1");
model.component("comp1").geom("geom1").feature("mov1").set("displx", (Box_Width  - ...
    Width)/2);
model.component("comp1").geom("geom1").feature("mov1").set("disply",  Box_Height - ...
    Height);
% Create boxes around Voronoi
% Create Box
model.component("comp1").geom("geom1").create("r1", "Rectangle");
model.component("comp1").geom("geom1").feature("r1").set("size",[Box_Width, ...
    Box_Height]);
geom1.run
% Difference of Voronoi and Box
model.component("comp1").geom("geom1").create("dif1", "Difference");
model.component("comp1").geom("geom1").feature("dif1").selection("input").set("r1");
model.component("comp1").geom("geom1").feature("dif1").selection("input2").set("mov1");
model.component("comp1").geom("geom1").feature("dif1").set("keep", true);

% Union of Box with voronoi
uni2 = geom1.feature.create('uni2','Union');
uni2.selection('input').set({'mov1' 'r1'});

geom1.run
```

E.2.5. Explicit Grain Selection

```
    for p = 1:length(P)+1
    seltag = ['sel' num2str(p)];
    model.component("comp1").selection.create(seltag, "Explicit");
    model.component("comp1").selection(seltag).geom(2);
    model.component("comp1").selection(seltag).geom('geom1', 2, 1, {'exterior'});
    model.component("comp1").selection(seltag).set(p);
    end
```

E.2.6. Boundary Selection

```
     % Create selection for boundary and crack
```

```
    % Fix all bounds except top
    % LEFT
    model.component("comp1").geom("geom1").create("edge_L", "BoxSelection");
    model.component("comp1").geom("geom1").feature("edge_L").set("xmin", -0.001);
    model.component("comp1").geom("geom1").feature("edge_L").set("ymin", -0.001);
    model.component("comp1").geom("geom1").feature("edge_L").set("ymax", Box_Height + ...
        0.001);
    model.component("comp1").geom("geom1").feature("edge_L").set("xmax", 0.001);
    model.component("comp1").geom("geom1").feature("edge_L").set("groupcontang", false);
    model.component("comp1").geom("geom1").feature("edge_L").set("condition", "inside");
    model.component("comp1").geom("geom1").feature("edge_L").set("entitydim", 1);
    % RIGHT
    model.component("comp1").geom("geom1").create("edge_R", "BoxSelection");
    model.component("comp1").geom("geom1").feature("edge_R").set("xmin", Box_Width - ...
        0.001);
    model.component("comp1").geom("geom1").feature("edge_R").set("ymin", -0.001);
    model.component("comp1").geom("geom1").feature("edge_R").set("ymax", Box_Height + ...
        0.001);
    model.component("comp1").geom("geom1").feature("edge_R").set("xmax", Box_Width  + ...
        0.001);
    model.component("comp1").geom("geom1").feature("edge_R").set("groupcontang", false);
    model.component("comp1").geom("geom1").feature("edge_R").set("condition", "inside");
    model.component("comp1").geom("geom1").feature("edge_R").set("entitydim", 1);
    % DOWN
    model.component("comp1").geom("geom1").create("edge_D", "BoxSelection");
    model.component("comp1").geom("geom1").feature("edge_D").set("xmin", -0.001);
    model.component("comp1").geom("geom1").feature("edge_D").set("ymin", -0.001);
    model.component("comp1").geom("geom1").feature("edge_D").set("ymax", +0.001);
    model.component("comp1").geom("geom1").feature("edge_D").set("xmax", Box_Width + ...
        0.001);
    model.component("comp1").geom("geom1").feature("edge_D").set("groupcontang", false);
    model.component("comp1").geom("geom1").feature("edge_D").set("condition", "inside");
    model.component("comp1").geom("geom1").feature("edge_D").set("entitydim", 1);
    % UP
    model.component("comp1").geom("geom1").create("edge_U", "BoxSelection");
    model.component("comp1").geom("geom1").feature("edge_U").set("xmin", - 0.001);
    model.component("comp1").geom("geom1").feature("edge_U").set("ymin", Box_Height - ...
        0.001);
    model.component("comp1").geom("geom1").feature("edge_U").set("ymax", Box_Height + ...
        0.001);
    model.component("comp1").geom("geom1").feature("edge_U").set("xmax", Box_Width  + ...
        0.001);
    model.component("comp1").geom("geom1").feature("edge_U").set("groupcontang", false);
    model.component("comp1").geom("geom1").feature("edge_U").set("condition", "inside");
    model.component("comp1").geom("geom1").feature("edge_U").set("entitydim", 1);
    % Crack
    model.component("comp1").selection.create("box1", "Box");
    model.component("comp1").selection("box1").set("entitydim", 1);
    model.component("comp1").selection("box1").set("xmin", (Box_Width-Width)/2 - 0.001);
    model.component("comp1").selection("box1").set("ymin", Box_Height - 0.001);
    model.component("comp1").selection("box1").set("ymax", Box_Height + 0.001);
    model.component("comp1").selection("box1").set("xmax", (Box_Width-Width)/2 + ...
        Width + 0.001);
    model.component("comp1").selection("box1").set("condition", "intersects");

    Surface = model.selection("box1").entities();
    Crack   = zeros(geom1.getNEdges,1);
```

E.2.7. RVE Selection

```
 % RVE Edges
model.component("comp1").selection.create("RVE_Edge", "Box");
model.component("comp1").selection("RVE_Edge").set("entitydim", 1);
model.component("comp1").selection("RVE_Edge").set("xmin", Box_Width/2 - b*10^6 - ...
    0.001);
model.component("comp1").selection("RVE_Edge").set("ymin", Box_Height  - b*10^6 - ...
    0.001);
model.component("comp1").selection("RVE_Edge").set("ymax", Box_Height + 0.001);
model.component("comp1").selection("RVE_Edge").set("xmax", Box_Width/2 + b*10^6 + ...
    0.001);
model.component("comp1").selection("RVE_Edge").set("condition", "intersects");
RVE_Edges = model.selection("RVE_Edge").entities();


% RVE Grains
model.component("comp1").selection.create("RVE_Grain", "Box");
model.component("comp1").selection("RVE_Grain").set("entitydim", 2);
model.component("comp1").selection("RVE_Grain").set("xmin", Box_Width/2 - b*10^6 ...
    - 0.001);
model.component("comp1").selection("RVE_Grain").set("ymin", Box_Height  - b*10^6 ...
    - 0.001);
model.component("comp1").selection("RVE_Grain").set("ymax", Box_Height + 0.001);
model.component("comp1").selection("RVE_Grain").set("xmax", Box_Width/2 + b*10^6 ...
    + 0.001);
model.component("comp1").selection("RVE_Grain").set("condition", "intersects");
RVE_Grain = model.selection("RVE_Grain").entities();
```

E.2.8. Stress Distribution

```
model.component("comp1").func().create("an1", "Analytic");
model.component("comp1").func("an1").set("fununit", "Pa");
model.component("comp1").func("an1").set("argunit", "m");
model.component("comp1").func("an1").setIndex("plotargs", "b", 0, 2);
model.component("comp1").func("an1").setIndex("plotargs", "-b", 0, 1);
model.component("comp1").func("an1").set("funcname", "P_dist");
model.component("comp1").func("an1").set("expr", ...
    "if(x/b>-1,if(x/b<1,P_max*sqrt(abs(1-(x/b)^2)),0),0)");
```

E.2.9. Physics

```
 phys = comp1.physics.create('solid', 'SolidMechanics', 'geom1');
model.component("comp1").physics("solid").feature("lemm1").set("IsotropicOption", ...
    "Lame");
```

E.2.10. Material Parameters

```
E_0 = 200*10^9;
nu  = 0.3;

lambdaLame_0 = (E_0*nu)/((1+nu)*(1-2*nu));
mhuLame_0    =  E_0/(2*(1+nu));
lambdaLame = lambdaLame_0*ones(size(P,1)+1,1);
mhuLame    = mhuLame_0*ones(size(P,1)+1,1);

for o = 1:length(P)+1
mattag = ['mat' num2str(o)];
mat = model.component('comp1').material.create(mattag);
model.component('comp1').material(mattag).selection.set(o);
mat.materialmodel('def').set('density', 7800);
model.component('comp1').material(mattag).propertyGroup().create('Lame', 'Lame ...
    parameters'); % Change e in Lame parameters to e with acute accent
model.component('comp1').material(mattag).propertyGroup('Lame').set('lambLame', ...
    lambdaLame_0);
model.component('comp1').material(mattag).propertyGroup('Lame').set('muLame', ...
    mhuLame_0);
end
```

E.2.11. Constraints and Load

```
 % Create fixed constraint
model.component("comp1").physics("solid").create("fix1", "Fixed", 1);
model.component("comp1").physics("solid").feature("fix1").selection()...
.named("geom1_edge_L");
model.component("comp1").physics("solid").create("fix2", "Fixed", 1);
model.component("comp1").physics("solid").feature("fix2").selection()...
.named("geom1_edge_R");
model.component("comp1").physics("solid").create("fix3", "Fixed", 1);
model.component("comp1").physics("solid").feature("fix3").selection()...
.named("geom1_edge_D");

% Add boundary load
model.component("comp1").physics("solid").create("bndl1", "BoundaryLoad", 1);
model.component("comp1").physics("solid").feature("bndl1").selection().named("geom1_edge_U");
model.component('comp1').physics('solid').feature('bndl1').set('LoadType', ...
    'FollowerPressure');
model.component("comp1").physics("solid").feature("bndl1").set("FollowerPressure", ...
    "P_dist(X-step)");
```

E.2.12. Meshing

```
mesh1 = comp1.mesh.create('mesh1');
ftri1 = mesh1.feature.create('ftri1','FreeTri');
mesh1.feature('size').set('hauto',6);
mesh1.run;
```

E.2.13. Study

```
std = model.study.create('std1');
std.feature.create('stat', 'Stationary');
model.study("std1").feature("stat").set("useparam", true);
model.study("std1").feature("stat").setIndex("punit", "m", 0);
model.study("std1").feature("stat").setIndex("plistarr", ...
    "range(3*b,(7*b-(3*b))/(nsteps-1),7*b)", 0);
model.study("std1").feature("stat").setIndex("pname", "step", 0);
```

E.3. CDM Model

E.3.1. Properties and Parameters

```
 m     = 10.1;
tau_r = 6.113 * 10^9;
D_del = 0.2;
D_edge           = zeros(geom1.getNEdges,1);
del_shear        = zeros(geom1.getNEdges,1);
Crack            = zeros(geom1.getNEdges,1);
D_grain          = zeros(length(P)+1,1);
del_sheargrain   = zeros(length(P)+1,1);
Dloop            = zeros(length(P)+1,1);
Dloopedge        = zeros(geom1.getNEdges,1);
s                = 1;
Ncrack           = 0;
N                = 0;
conlist          = [];
CrackInitiation  = [];
Cycle            = [];
```

E.3.2. Start Model

```
while any(ismember(Crack,Surface)) ~= 1
% Run study
model.study("std1").run();
```

E.3.3. Line Average

```
if s == 1
%% Line Average
% Edge
for l = 1:geom1.getNEdges
if ismember(l,RVE_Edges)
% Create the average values for edges in RVE
avtag = ['av' num2str(l)];
model.result.numerical().create(avtag, 'AvLine');
model.result.numerical(avtag).setIndex('expr', ...
    '-0.5*(solid.sx-solid.sy)*sin(2*atan2(nY,nX))+solid.sxy*cos(2*atan2(nY,nX))', 0);
model.result.numerical(avtag).selection().set(l);
else
end
end
end
```

E.3.4. Retrieve Data

```
for i = 1:geom1.getNEdges
if ismember(i,RVE_Edges)
if isnan(D_edge(i))
del_shear(i,:) = nan;
else
avtag  = ['av' num2str(i)];
% Retrieve data
av_val = model.result().numerical(avtag).computeResult();
% Calculate shear stress difference during one loadcycle
del_shear(i,:) = max(av_val(1,:)) - min(av_val(1,:));
end
else
end
end
```

E.3.5. Damage Rate

```
    D_rate  = (del_shear./(tau_r*(1-D_edge))).^m;
    % Max damage rate
    [D_ratemax,I] = max(D_rate);
    % Calculate jump in cycles
    N_del = D_del/D_ratemax;
    N     = N + N_del
    Cycle(s) = N;
    % Damage on all edges
    D_edge = D_edge + D_rate*N_del;
    Dloopedge(:,s) = D_edge;
```

E.3.6. Crack Creation

```
    for j = 1:length(D_edge)
    if ismember(j,RVE_Edges)
    if D_edge(j) ≥ 1 % If damage larger than 1
    Ncrack = Ncrack + 1;
    if Ncrack == 1
    % Crack initiation
    CrackInitiation = N;
    % Add Crack
    model.component("comp1").physics("solid").create("crack1", "Crack", 1);
    % Settings of crack node
    model.component("comp1").physics("solid").feature("crack1").set("CrackSurface", ...
        "FromGeometry");
    % Add Contact node for friction
    model.component("comp1").physics("solid").create("cnt1", "SolidContact", 1);
    % Settings of Contact node
    model.component("comp1").physics("solid").feature("cnt1").create("fric1", ...
        "Friction", 1);
    model.component("comp1").physics("solid").feature("cnt1").feature("fric1")...
    .set("FrictionModel", "Coulomb");
    model.component("comp1").physics("solid").feature("cnt1").feature("fric1")...
    .set("mu_fric", 0.4);
    end
    % Save the crack and crack order
    Crack(j) = j;
    Crackorder(Ncrack,:) = [j,N];
    % Fill in crack node
    model.component("comp1").physics("solid").feature("crack1")...
    .selection.set(nonzeros(Crack));
    model.component("comp1").physics("solid").feature("crack1")...
    .selection("Face1").set(nonzeros(Crack));
    D_edge(j) = nan;
    % Create contact selection
    contag = ['p' num2str(Ncrack)];
    conlist = [conlist;{contag}];
    model.component('comp1').pair.create(contag,'Contact');
    model.component('comp1').pair(contag).source.set(j);
    model.component('comp1').pair(contag).destination.set(j);
    % Add to contact node
    model.component("comp1").physics("solid").feature("cnt1").set("pairs", conlist);
    end
    else
    end
    end
```

E.3.7. Damage of Grain

```
      % Grain data gathering
    for k = 1:length(P)+1
    if ismember(k,RVE_Grain)
    % 1 is the outer box, can be skipped
    seltag = ['sel' num2str(k)];
    mattag = ['mat' num2str(k)];
    faces = model.selection(seltag).entities();
    D_edges = D_edge(faces);
    D_edges(isnan(D_edges)) = 1;
    D_grain = mean(D_edges);
    lambdaLame(k) = (E_0*(1-D_grain)*nu)/((1+nu)*(1-2*nu));
    mhuLame(k)    = (E_0*(1-D_grain))/(2*(1+nu));
    if D_grain == 1
    lambdaLame(k) =  1;
    mhuLame(k)    =  1;
    end
    Dloop(k,s) = D_grain;
    mat = model.component("comp1").material(mattag);
    model.component('comp1').material(mattag).propertyGroup('Lame').set('lambLame', ...
        lambdaLame(k));
    model.component('comp1').material(mattag).propertyGroup('Lame').set('muLame', ...
        mhuLame(k));
    else
    end
    end
```

F

Full Code

```
clc;            % Clear command window.
clear;          % Delete all variables.
close all;      % Close all figure windows except those created by imtool.
%% filename
for init = 1:30 %CHECK IF V AND C ARE DIFFERENT, RESTARTING MATLAB RESTARTS THE ...
    RANDOM GENERATOR AND THIS COULD MEAN A REPETITION IN MICROSTRUCTURE
 if init ≠ 1
  ModelUtil.remove('Model');
 end
 filename = ['Testingdata/100 um/Test100um.',num2str(init),'.mat'];
 %% Other parameters
 % Parametric sweep
 nsteps = 40;
 %% BELANGRIJK
 b = 100*10^(-6);
 P_max = 1*10^9;
 %% Create Voronoi
 try
  %% Dimensions of domain
  Width  = 3*(b*10^6);
  Height = 1.5*(b*10^6);
  Box_Width  = 10*(b*10^6);
  Box_Height =  6*(b*10^6);

  % Distribution
  Mean_Grain_Diameter = 10;
  Distribution        = 2.5;

  % Initial points placed in domain
  n_Points = 280000;
  xPoint = Width*rand(n_Points,1);
  yPoint = Height*rand(n_Points,1);
  Random_Points = [xPoint,yPoint];

  % Define counter for how many actually get placed.
  numPoints = 0;

  % Define fail safe, how many iterations you want to keep trying before quitting.
  maxIterations = 1000 * n_Points;
  loopCounter = 1;

  % Allocate space to variables
  x = nan(numPoints,1);
  y = nan(numPoints,1);
  Norm_Distance = nan(numPoints,1);

  %% Point Selection
```

```
47  while numPoints < n_Points && loopCounter < maxIterations && numPoints < ...
        length(Random_Points)
48   numPoints = numPoints + 1;
49   xChosen = Random_Points(numPoints,1);
50   yChosen = Random_Points(numPoints,2);
51   x(numPoints,1) = xChosen;
52   y(numPoints,1) = yChosen;
53
54   % The grain diameter of the chosen point
55   Norm_Distance(numPoints,1) = abs(normrnd(Mean_Grain_Diameter,Distribution));
56   % Calculate the distance of all points to the chosen point
57   Distances = sqrt((Random_Points(:,1)-xChosen) .^ 2 + (Random_Points(:,2) - ...
         yChosen) .^ 2);
58   closePair = Distances ≤ Norm_Distance(numPoints);
59
60   Random_Points(closePair == 1,:) = nan; % Remove all points inside certain distance
61   Random_Points = [x(1:numPoints) y(1:numPoints); Random_Points(numPoints+1:end,1) ...
         Random_Points(numPoints+1:end,2)];
62   Random_Points = rmmissing(Random_Points);
63
64   % Increment the loop counter.
65   loopCounter = loopCounter + 1;
66  end
67  P = [x,y];
68  %% Mirror all the points
69  Right = [2*Width - x ,            y];
70  Left  = [0       - x ,            y];
71  Up    = [          x , 2*Height - y];
72  Low   = [          x , 0        - y];
73  P_Mirror = [P;Right;Left;Up;Low];
74  [v,c] = voronoin(P_Mirror);
75  %% Remove small vertices for better mesh
76  min_Dist = 2;
77  cnt = 1;
78  tol = 1.e-10;
79  rnd = 5;
80
81  edges = [0,Width,Height];
82  LLcorner = [0,0];
83  LUcorner = [0,Height];
84  RLcorner = [Width,0];
85  RUcorner = [Width,Height];
86
87  for i = 1:length(P)
88   % All the points of the grain
89   VPoint = v(c{i},:);
90   c_new  = c{i};
91   cntB = 0;
92   % Distance between the points
93   Dist = pdist(VPoint);
94   % Find points that are too close to each other
95   if any(Dist < min_Dist)
96    cnt = cnt + 1;
97    % Make square out of Dist
98    Z = squareform(Dist);
99    % row and col
100   [row,col] = find(Z < min_Dist & Z ≠ 0);
101   ClosePoints = VPoint(col,:);
102   [¬,idx] = unique(sort(ClosePoints,2),'rows','stable');
103   ClosePoints = ClosePoints(idx,:);
104   repCol = ismember(VPoint,ClosePoints,'rows');
105   B = [row,col];
106   [¬,idx] = unique(sort(B,2),'rows','stable');
107   B = B(idx,:);
108   row = B(:,1);
109   col = B(:,2);
110   for o = 1:size(B,1)
111    ClP = round(VPoint(B(o,:)',:),rnd);
112    %% If VPoint has edge members
113    if any(any(ismembertol(ClP,edges)))
114     %% Check if VPoint has corner points
```

```
115         % Left Lower Corner %
116         if any(ismembertol(ClP,LLcorner,'ByRow',true))
117          valChange = not(ismembertol(ClP,LLcorner,'ByRow',true));
118          CloseP = ClP(valChange,:);
119          if ismembertol(CloseP(1,2),0)          % Is it on the x-axis?
120           CloseP = [min_Dist, 0];
121          else                                   % Is it on the y-axis?
122           CloseP = [0, min_Dist];
123          end
124          ClP(valChange,:) = CloseP;
125          VPoint(B(o,:)',:) = ClP;
126          % Left Upper Corner %
127          elseif any(ismembertol(ClP,LUcorner,'ByRow',true))
128           valChange = not(ismembertol(ClP,LUcorner,'ByRow',true));
129           CloseP = ClP(valChange,:);
130           if ismembertol(CloseP(1,2),Height)    % Is it on the x-axis?
131            CloseP = [min_Dist, Height];
132           else                                  % Is it on the y-axis?
133            CloseP = [0, Height - min_Dist];
134           end
135           ClP(valChange,:) = CloseP;
136           VPoint(B(o,:)',:) = ClP;
137           % Right Lower Corner %
138          elseif any(ismembertol(ClP,RLcorner,'ByRow',true))
139           valChange = not(ismembertol(ClP,RLcorner,'ByRow',true));
140           CloseP = ClP(valChange,:);
141           if ismembertol(CloseP(1,2),0)          % Is it on the x-axis?
142            CloseP = [Width - min_Dist, 0];
143           else                                  % Is it on the y-axis?
144            CloseP = [Width, min_Dist];
145           end
146           ClP(valChange,:) = CloseP;
147           VPoint(B(o,:)',:) = ClP;
148           % Right Upper Corner %
149          elseif any(ismembertol(ClP,RUcorner,'ByRow',true))
150           valChange = not(ismembertol(ClP,RUcorner,'ByRow',true));
151           CloseP = ClP(valChange,:);
152           if ismembertol(CloseP(1,2),Height)    % Is it on the x-axis?
153            CloseP = [Width - min_Dist, Height];
154           else                                  % Is it on the x-axis?
155            CloseP = [Width, Height - min_Dist];
156           end
157           ClP(valChange,:)  = CloseP;
158           VPoint(B(o,:)',:) = ClP;
159           % Check if adjacent or on edge %
160           %% Bottom edge y = 0
161          elseif any(ismembertol(ClP(:,2),0,'ByRow',true))
162           m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
163           d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
164           t = min_Dist/d;
165           if abs(m) < tol % on line
166            [maxClP,maxvalClP] = max(ClP(:,1));
167            [minClP,minvalClP] = min(ClP(:,1));
168            ClP(maxvalClP,1) = ClP(minvalClP,1) + min_Dist;
169           elseif abs(m) == Inf % Perpendicular to line
170            valChange = not(ismembertol(ClP(:,2),0,'ByRow',true));
171            ClP(valChange,2) = ClP(not(valChange),2) + min_Dist;
172           else
173            valChange = not(ismembertol(ClP(:,2),0,'ByRow',true));
174            ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
175            ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
176           end
177           VPoint(B(o,:)',:) = ClP;
178           %% Left edge   x = 0
179          elseif any(ismembertol(ClP(:,1),0,'ByRow',true))
180           m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
181           d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
182           t = min_Dist/d;
183           if abs(m) == Inf  % on line
184            [maxClP,maxvalClP] = max(ClP(:,2));
185            [minClP,minvalClP] = min(ClP(:,2));
```

```
            ClP(maxvalClP,2) = ClP(minvalClP,2) + min_Dist;
          elseif abs(m) < tol % Perpendicular to line
            valChange = not(ismembertol(ClP(:,1),0,'ByRow',true));
            ClP(valChange,1) = ClP(not(valChange),1) + min_Dist;
          else
            valChange = not(ismembertol(ClP(:,1),0,'ByRow',true));
            ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
            ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
          end
          VPoint(B(o,:)',:) = ClP;
          %% Top edge y = Height
        elseif any(ismembertol(ClP(:,2),Height,'ByRow',true))
          m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
          d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
          t = min_Dist/d;
          if abs(m) < tol % on line
            [maxClP,maxvalClP] = max(ClP(:,1));
            [minClP,minvalClP] = min(ClP(:,1));
            ClP(maxvalClP,1) = ClP(minvalClP,1) + min_Dist;
          elseif abs(m) == Inf % Perpendicular to line
            valChange = not(ismembertol(ClP(:,2),Height,'ByRow',true));
            ClP(valChange,2) = ClP(not(valChange),2) - min_Dist;
          else
            valChange = not(ismembertol(ClP(:,2),Height,'ByRow',true));
            ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
            ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
          end
          VPoint(B(o,:)',:) = ClP;
          %% Right edge x = Width
        elseif any(ismembertol(ClP(:,1),Width,'ByRow',true))
          m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
          d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
          t = min_Dist/d;
          if abs(m) == Inf % on line
            [maxClP,maxvalClP] = max(ClP(:,2));
            [minClP,minvalClP] = min(ClP(:,2));
            ClP(maxvalClP,2) = ClP(minvalClP,2) + min_Dist;
          elseif abs(m) < tol % Perpendicular to line
            valChange = not(ismembertol(ClP(:,1),Width,'ByRow',true));
            ClP(valChange,1) = ClP(not(valChange),1) - min_Dist;
          else
            valChange = not(ismembertol(ClP(:,1),Width,'ByRow',true));
            ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
            ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
          end
          VPoint(B(o,:)',:) = ClP;
        end
        %% ClP has no edge values
        % Does it have multiple small lines?
      elseif sum(any(ismember(B,B(o,:)),2)) > 1 % If true, close lines adjacent
        cntB = cntB + 1;
        if cntB > 1
          continue
        end
        [C,ia,ic] = unique(B);
        B_counts = accumarray(ic,1);
        numCount = [B_counts,C];
        % Find point in the middle
        A = C(B_counts == 2);
        if size(A,1) > 1 % The end points of the adjacent line also within min dist
          G = VPoint(A,:);
          Gdist = pdist(G);
          ZG = squareform(Gdist);
          [maxValue,linearIndexesOfMaxes] = max(ZG(:));
          [rowsOfMaxes, colsOfMaxes] = find(ZG == maxValue);
          VPoint(A(colsOfMaxes),:) = nan;
          c_keep = c_new(A(not(ismember(A,A(colsOfMaxes)))));
          c_del  = c_new(A(colsOfMaxes));
          c_new(A(colsOfMaxes))  = nan;
          VPoint(A(colsOfMaxes)) = nan;
        else
```

```
257         Q = B(any(ismember(B,B(o,:)),2),:);
258         Q = Q(Q ≠ A);
259         c_keep = c_new(A);
260         c_del  = c_new(Q);
261         c_new(Q) = nan;
262         VPoint(Q,:) = nan;
263        end
264        for j = 1:length(P)
265         if any(ismember(c{j},c_del)) && j ≠ i % if cell has point(s) to delete
266          c_other = c{j};
267          [La,Lb] = ismember(c_other,c_del);
268          [La1,Lb1] = ismember(c_other,c_keep);
269          if any(ismember(c{j},c_keep)) % if cell has point(s) to keep, only delete ...
                 values any(ismember(c{j},c_keep(nonzeros(Lb))))
270           c_other(La) = [];
271          else  % if cell does not have point(s) to keep, only replace values
272           c_other(La) = c_keep;
273          end
274          c{j} = c_other;
275         end
276        end
277       else
278        %% Create new points
279        ClP(2,1) = (ClP(1,1) + ClP(2,1))/ 2;
280        ClP(2,2) = (ClP(1,2) + ClP(2,2))/ 2;
281        % Replace the first point with the new value and delete the other point
282        ClP(1,:) = nan;
283        VPoint(B(o,:)',:) = ClP;
284        % Delete the other c value
285        c_keep = c_new(B(o,2));
286        c_del  = c_new(B(o,1));
287        c_new(B(o,1)) = nan;
288        % Find the other cells with these values
289        for j = 1:length(P)
290         if any(ismember(c{j},c_del)) && j ≠ i % if cell has point(s) to delete
291          c_other = c{j};
292          [La,Lb] = ismember(c_other,c_del);
293          [La1,Lb1] = ismember(c_other,c_keep);
294          if any(ismember(c{j},c_keep(nonzeros(Lb)))) % if cell has point(s) to keep, ...
                 only delete values
295           c_other(La) = [];
296           c{j} = c_other;
297          else  % if cell does not have point(s) to keep, only replace values
298           c_other(La) = c_keep(nonzeros(Lb));
299           c{j} = c_other;
300          end
301         end
302        end
303       end
304       % Replace old values
305       c{i} = rmmissing(c_new);
306       v(c{i},:) = rmmissing(VPoint);
307      end
308     end
309    end
310  catch
311    %% Dimensions of domain
312    Width  = 3*(b*10^6);
313    Height = 1.5*(b*10^6);
314    Box_Width  = 10*(b*10^6);
315    Box_Height =  6*(b*10^6);
316
317    % Distribution
318    Mean_Grain_Diameter = 10;
319    Distribution        = 2.5;
320
321    % Initial points placed in domain
322    n_Points = 280000;
323    xPoint = Width*rand(n_Points,1);
324    yPoint = Height*rand(n_Points,1);
325    Random_Points = [xPoint,yPoint];
```

```
326
327     % Define counter for how many actually get placed.
328     numPoints = 0;
329
330     % Define fail safe, how many iterations you want to keep trying before quitting.
331     maxIterations = 1000 * n_Points;
332     loopCounter = 1;
333
334     % Allocate space to variables
335     x = nan(numPoints,1);
336     y = nan(numPoints,1);
337     Norm_Distance = nan(numPoints,1);
338
339     %% Point Selection
340     while numPoints < n_Points && loopCounter < maxIterations && numPoints < ...
            length(Random_Points)
341      numPoints = numPoints + 1;
342      xChosen = Random_Points(numPoints,1);
343      yChosen = Random_Points(numPoints,2);
344      x(numPoints,1) = xChosen;
345      y(numPoints,1) = yChosen;
346
347      % The grain diameter of the chosen point
348      Norm_Distance(numPoints,1) = abs(normrnd(Mean_Grain_Diameter,Distribution));
349      % Calculate the distance of all points to the chosen point
350      Distances = sqrt((Random_Points(:,1)-xChosen) .^ 2 + (Random_Points(:,2) - ...
            yChosen) .^ 2);
351      closePair = Distances ≤ Norm_Distance(numPoints);
352
353      Random_Points(closePair == 1,:) = nan; % Remove all points inside certain distance
354      Random_Points = [x(1:numPoints) y(1:numPoints); Random_Points(numPoints+1:end,1) ...
            Random_Points(numPoints+1:end,2)];
355      Random_Points = rmmissing(Random_Points);
356
357      % Increment the loop counter.
358      loopCounter = loopCounter + 1;
359     end
360     P = [x,y];
361     %% Mirror all the points
362     Right = [2*Width - x ,            y];
363     Left  = [0       - x ,            y];
364     Up    = [          x , 2*Height - y];
365     Low   = [          x , 0        - y];
366     P_Mirror = [P;Right;Left;Up;Low];
367     [v,c] = voronoin(P_Mirror);
368     %% Remove small vertices for better mesh
369     min_Dist = 2;
370     cnt = 1;
371     tol = 1.e-10;
372     rnd = 5;
373
374     edges = [0,Width,Height];
375     LLcorner = [0,0];
376     LUcorner = [0,Height];
377     RLcorner = [Width,0];
378     RUcorner = [Width,Height];
379
380     for i = 1:length(P)
381      % All the points of the grain
382      VPoint = v(c{i},:);
383      c_new  = c{i};
384      cntB = 0;
385      % Distance between the points
386      Dist = pdist(VPoint);
387      % Find points that are too close to each other
388      if any(Dist < min_Dist)
389       cnt = cnt + 1;
390       % Make square out of Dist
391       Z = squareform(Dist);
392       % row and col
393       [row,col] = find(Z < min_Dist & Z ≠ 0);
```

```
394       ClosePoints = VPoint(col,:);
395       [¬,idx] = unique(sort(ClosePoints,2),'rows','stable');
396       ClosePoints = ClosePoints(idx,:);
397       repCol = ismember(VPoint,ClosePoints,'rows');
398       B = [row,col];
399       [¬,idx] = unique(sort(B,2),'rows','stable');
400       B = B(idx,:);
401       row = B(:,1);
402       col = B(:,2);
403       for o = 1:size(B,1)
404        ClP = round(VPoint(B(o,:)',:),rnd);
405        %% If VPoint has edge members
406        if any(any(ismembertol(ClP,edges)))
407         %% Check if VPoint has corner points
408         % Left Lower Corner %
409         if any(ismembertol(ClP,LLcorner,'ByRow',true))
410          valChange = not(ismembertol(ClP,LLcorner,'ByRow',true));
411          CloseP = ClP(valChange,:);
412          if ismembertol(CloseP(1,2),0)          % Is it on the x-axis?
413           CloseP = [min_Dist, 0];
414          else                                   % Is it on the y-axis?
415           CloseP = [0, min_Dist];
416          end
417          ClP(valChange,:) = CloseP;
418          VPoint(B(o,:)',:) = ClP;
419          % Left Upper Corner %
420          elseif any(ismembertol(ClP,LUcorner,'ByRow',true))
421           valChange = not(ismembertol(ClP,LUcorner,'ByRow',true));
422           CloseP = ClP(valChange,:);
423           if ismembertol(CloseP(1,2),Height)    % Is it on the x-axis?
424            CloseP = [min_Dist, Height];
425           else                                  % Is it on the y-axis?
426            CloseP = [0, Height - min_Dist];
427           end
428           ClP(valChange,:) = CloseP;
429           VPoint(B(o,:)',:) = ClP;
430           % Right Lower Corner %
431          elseif any(ismembertol(ClP,RLcorner,'ByRow',true))
432           valChange = not(ismembertol(ClP,RLcorner,'ByRow',true));
433           CloseP = ClP(valChange,:);
434           if ismembertol(CloseP(1,2),0)          % Is it on the x-axis?
435            CloseP = [Width - min_Dist, 0];
436           else                                   % Is it on the y-axis?
437            CloseP = [Width, min_Dist];
438           end
439           ClP(valChange,:) = CloseP;
440           VPoint(B(o,:)',:) = ClP;
441           % Right Upper Corner %
442          elseif any(ismembertol(ClP,RUcorner,'ByRow',true))
443           valChange = not(ismembertol(ClP,RUcorner,'ByRow',true));
444           CloseP = ClP(valChange,:);
445           if ismembertol(CloseP(1,2),Height)    % Is it on the x-axis?
446            CloseP = [Width - min_Dist, Height];
447           else                                  % Is it on the x-axis?
448            CloseP = [Width, Height - min_Dist];
449           end
450           ClP(valChange,:)  = CloseP;
451           VPoint(B(o,:)',:) = ClP;
452           % Check if adjacent or on edge %
453           %% Bottom edge y = 0
454          elseif any(ismembertol(ClP(:,2),0,'ByRow',true))
455           m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
456           d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
457           t = min_Dist/d;
458           if abs(m) < tol % on line
459            [maxClP,maxvalClP] = max(ClP(:,1));
460            [minClP,minvalClP] = min(ClP(:,1));
461            ClP(maxvalClP,1) = ClP(minvalClP,1) + min_Dist;
462           elseif abs(m) == Inf % Perpendicular to line
463            valChange = not(ismembertol(ClP(:,2),0,'ByRow',true));
464            ClP(valChange,2) = ClP(not(valChange),2) + min_Dist;
```

```
465          else
466           valChange = not(ismembertol(ClP(:,2),0,'ByRow',true));
467           ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
468           ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
469          end
470          VPoint(B(o,:)',:) = ClP;
471          %% Left edge   x = 0
472         elseif any(ismembertol(ClP(:,1),0,'ByRow',true))
473          m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
474          d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
475          t = min_Dist/d;
476          if abs(m) == Inf  % on line
477           [maxClP,maxvalClP] = max(ClP(:,2));
478           [minClP,minvalClP] = min(ClP(:,2));
479           ClP(maxvalClP,2) = ClP(minvalClP,2) + min_Dist;
480          elseif abs(m) < tol % Perpendicular to line
481           valChange = not(ismembertol(ClP(:,1),0,'ByRow',true));
482           ClP(valChange,1) = ClP(not(valChange),1) + min_Dist;
483          else
484           valChange = not(ismembertol(ClP(:,1),0,'ByRow',true));
485           ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
486           ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
487          end
488          VPoint(B(o,:)',:) = ClP;
489          %% Top edge y = Height
490         elseif any(ismembertol(ClP(:,2),Height,'ByRow',true))
491          m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
492          d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
493          t = min_Dist/d;
494          if abs(m) < tol % on line
495           [maxClP,maxvalClP] = max(ClP(:,1));
496           [minClP,minvalClP] = min(ClP(:,1));
497           ClP(maxvalClP,1) = ClP(minvalClP,1) + min_Dist;
498          elseif abs(m) == Inf % Perpendicular to line
499           valChange = not(ismembertol(ClP(:,2),Height,'ByRow',true));
500           ClP(valChange,2) = ClP(not(valChange),2) - min_Dist;
501          else
502           valChange = not(ismembertol(ClP(:,2),Height,'ByRow',true));
503           ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
504           ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
505          end
506          VPoint(B(o,:)',:) = ClP;
507          %% Right edge x = Width
508         elseif any(ismembertol(ClP(:,1),Width,'ByRow',true))
509          m = (ClP(2,2) - ClP(1,2))/(ClP(2,1) - ClP(1,1));
510          d = sqrt((ClP(2,1) - ClP(1,1))^2 + (ClP(2,2) - ClP(1,2))^2);
511          t = min_Dist/d;
512          if abs(m) == Inf % on line
513           [maxClP,maxvalClP] = max(ClP(:,2));
514           [minClP,minvalClP] = min(ClP(:,2));
515           ClP(maxvalClP,2) = ClP(minvalClP,2) + min_Dist;
516          elseif abs(m) < tol % Perpendicular to line
517           valChange = not(ismembertol(ClP(:,1),Width,'ByRow',true));
518           ClP(valChange,1) = ClP(not(valChange),1) - min_Dist;
519          else
520           valChange = not(ismembertol(ClP(:,1),Width,'ByRow',true));
521           ClP(valChange,1) = (1-t)*ClP(not(valChange),1) + ClP(valChange,1)*t;
522           ClP(valChange,2) = (1-t)*ClP(not(valChange),2) + ClP(valChange,2)*t;
523          end
524          VPoint(B(o,:)',:) = ClP;
525        end
526        %% ClP has no edge values
527        % Does it have multiple small lines?
528       elseif sum(any(ismember(B,B(o,:)),2)) > 1 % If true, close lines adjacent
529        cntB = cntB + 1;
530        if cntB > 1
531         continue
532        end
533        [C,ia,ic] = unique(B);
534        B_counts = accumarray(ic,1);
535        numCount = [B_counts,C];
```

```
536         % Find point in the middle
537         A = C(B_counts == 2);
538         if size(A,1) > 1 % The end points of the adjacent line also within min dist
539          G = VPoint(A,:);
540          Gdist = pdist(G);
541          ZG = squareform(Gdist);
542          [maxValue,linearIndexesOfMaxes] = max(ZG(:));
543          [rowsOfMaxes, colsOfMaxes] = find(ZG == maxValue);
544          VPoint(A(colsOfMaxes),:) = nan;
545          c_keep = c_new(A(not(ismember(A,A(colsOfMaxes)))));
546          c_del  = c_new(A(colsOfMaxes));
547          c_new(A(colsOfMaxes))  = nan;
548          VPoint(A(colsOfMaxes)) = nan;
549         else
550          Q = B(any(ismember(B,B(o,:)),2),:);
551          Q = Q(Q ≠ A);
552          c_keep = c_new(A);
553          c_del  = c_new(Q);
554          c_new(Q) = nan;
555          VPoint(Q,:) = nan;
556         end
557         for j = 1:length(P)
558          if any(ismember(c{j},c_del)) && j ≠ i % if cell has point(s) to delete
559           c_other = c{j};
560           [La,Lb] = ismember(c_other,c_del);
561           [La1,Lb1] = ismember(c_other,c_keep);
562           if any(ismember(c{j},c_keep)) % if cell has point(s) to keep, only delete ...
                  values any(ismember(c{j},c_keep(nonzeros(Lb))))
563            c_other(La) = [];
564           else  % if cell does not have point(s) to keep, only replace values
565            c_other(La) = c_keep;
566           end
567           c{j} = c_other;
568          end
569         end
570        else
571         %% Create new points
572         ClP(2,1) = (ClP(1,1) + ClP(2,1))/ 2;
573         ClP(2,2) = (ClP(1,2) + ClP(2,2))/ 2;
574         % Replace the first point with the new value and delete the other point
575         ClP(1,:) = nan;
576         VPoint(B(o,:)',:) = ClP;
577         % Delete the other c value
578         c_keep = c_new(B(o,2));
579         c_del  = c_new(B(o,1));
580         c_new(B(o,1)) = nan;
581         % Find the other cells with these values
582         for j = 1:length(P)
583          if any(ismember(c{j},c_del)) && j ≠ i % if cell has point(s) to delete
584           c_other = c{j};
585           [La,Lb] = ismember(c_other,c_del);
586           [La1,Lb1] = ismember(c_other,c_keep);
587           if any(ismember(c{j},c_keep(nonzeros(Lb)))) % if cell has point(s) to keep, ...
                  only delete values
588            c_other(La) = [];
589            c{j} = c_other;
590           else  % if cell does not have point(s) to keep, only replace values
591            c_other(La) = c_keep(nonzeros(Lb));
592            c{j} = c_other;
593           end
594          end
595         end
596        end
597        % Replace old values
598        c{i} = rmmissing(c_new);
599        v(c{i},:) = rmmissing(VPoint);
600       end
601      end
602     end
603    end
604    %% START COMSOL
```

```
import com.comsol.model.*
import com.comsol.model.util.*
model = ModelUtil.create('Model1');
comp1 = model.component.create('comp1',true);
geom1 = comp1.geom.create('geom1',2);
model.component("comp1").geom("geom1").lengthUnit("um");
% Parameters
model.param().set("b", [num2str(b) '[m]']);
model.param().set("P_max", [num2str(P_max) '[Pa]']);
model.param().set("step", 1);
model.param().set("nsteps", nsteps);
% Create the grains in COMSOL
for q = 1:length(P)
 V = v(c{q},:);
 tag = ['pol' num2str(q)];
 geom1.feature.create(tag,'Polygon');
 geom1.feature(tag).set('x', V(:,1)).set('y', V(:,2));
end
geom1.run

% Union of all the grains
uni1 = geom1.feature.create('uni1','Union');
uni1.selection('input').all();
geom1.feature('uni1').selection('input');
% Move union
model.component("comp1").geom("geom1").create("mov1", "Move");
model.component("comp1").geom("geom1").feature("mov1").selection("input").set("uni1");
model.component("comp1").geom("geom1").feature("mov1").set("displx", (Box_Width  - ...
    Width)/2);
model.component("comp1").geom("geom1").feature("mov1").set("disply",  Box_Height - ...
    Height);
% Create boxes around Voronoi
% Create Box
model.component("comp1").geom("geom1").create("r1", "Rectangle");
model.component("comp1").geom("geom1").feature("r1").set("size",[Box_Width, ...
    Box_Height]);
geom1.run
% Difference of Voronoi and Box
model.component("comp1").geom("geom1").create("dif1", "Difference");
model.component("comp1").geom("geom1").feature("dif1").selection("input").set("r1");
model.component("comp1").geom("geom1").feature("dif1").selection("input2").set("mov1");
model.component("comp1").geom("geom1").feature("dif1").set("keep", true);

% Union of Box with voronoi
uni2 = geom1.feature.create('uni2','Union');
uni2.selection('input').set({'mov1' 'r1'});

geom1.run

% Create explicit grain selection
for p = 1:length(P)+1
 seltag = ['sel' num2str(p)];
 model.component("comp1").selection.create(seltag, "Explicit");
 model.component("comp1").selection(seltag).geom(2);
 model.component("comp1").selection(seltag).geom('geom1', 2, 1, {'exterior'});
 model.component("comp1").selection(seltag).set(p);
end
% Create selection for boundary and crack
% Fix all bounds except top
% LEFT
model.component("comp1").geom("geom1").create("edge_L", "BoxSelection");
model.component("comp1").geom("geom1").feature("edge_L").set("xmin", -0.001);
model.component("comp1").geom("geom1").feature("edge_L").set("ymin", -0.001);
model.component("comp1").geom("geom1").feature("edge_L").set("ymax", Box_Height + ...
    0.001);
model.component("comp1").geom("geom1").feature("edge_L").set("xmax", 0.001);
model.component("comp1").geom("geom1").feature("edge_L").set("groupcontang", false);
model.component("comp1").geom("geom1").feature("edge_L").set("condition", "inside");
model.component("comp1").geom("geom1").feature("edge_L").set("entitydim", 1);
% RIGHT
model.component("comp1").geom("geom1").create("edge_R", "BoxSelection");
```

```
model.component("comp1").geom("geom1").feature("edge_R").set("xmin", Box_Width - 0.001);
model.component("comp1").geom("geom1").feature("edge_R").set("ymin", -0.001);
model.component("comp1").geom("geom1").feature("edge_R").set("ymax", Box_Height + ...
    0.001);
model.component("comp1").geom("geom1").feature("edge_R").set("xmax", Box_Width  + ...
    0.001);
model.component("comp1").geom("geom1").feature("edge_R").set("groupcontang", false);
model.component("comp1").geom("geom1").feature("edge_R").set("condition", "inside");
model.component("comp1").geom("geom1").feature("edge_R").set("entitydim", 1);
% DOWN
model.component("comp1").geom("geom1").create("edge_D", "BoxSelection");
model.component("comp1").geom("geom1").feature("edge_D").set("xmin", -0.001);
model.component("comp1").geom("geom1").feature("edge_D").set("ymin", -0.001);
model.component("comp1").geom("geom1").feature("edge_D").set("ymax", +0.001);
model.component("comp1").geom("geom1").feature("edge_D").set("xmax", Box_Width + 0.001);
model.component("comp1").geom("geom1").feature("edge_D").set("groupcontang", false);
model.component("comp1").geom("geom1").feature("edge_D").set("condition", "inside");
model.component("comp1").geom("geom1").feature("edge_D").set("entitydim", 1);
% UP
model.component("comp1").geom("geom1").create("edge_U", "BoxSelection");
model.component("comp1").geom("geom1").feature("edge_U").set("xmin", - 0.001);
model.component("comp1").geom("geom1").feature("edge_U").set("ymin", Box_Height - ...
    0.001);
model.component("comp1").geom("geom1").feature("edge_U").set("ymax", Box_Height + ...
    0.001);
model.component("comp1").geom("geom1").feature("edge_U").set("xmax", Box_Width  + ...
    0.001);
model.component("comp1").geom("geom1").feature("edge_U").set("groupcontang", false);
model.component("comp1").geom("geom1").feature("edge_U").set("condition", "inside");
model.component("comp1").geom("geom1").feature("edge_U").set("entitydim", 1);
% Crack
model.component("comp1").selection.create("box1", "Box");
model.component("comp1").selection("box1").set("entitydim", 1);
model.component("comp1").selection("box1").set("xmin", (Box_Width-Width)/2 - 0.001);
model.component("comp1").selection("box1").set("ymin", Box_Height - 0.001);
model.component("comp1").selection("box1").set("ymax", Box_Height + 0.001);
model.component("comp1").selection("box1").set("xmax", (Box_Width-Width)/2 + Width + ...
    0.001);
model.component("comp1").selection("box1").set("condition", "intersects");

Surface = model.selection("box1").entities();
Crack   = zeros(geom1.getNEdges,1);

% RVE Edges
model.component("comp1").selection.create("RVE_Edge", "Box");
model.component("comp1").selection("RVE_Edge").set("entitydim", 1);
model.component("comp1").selection("RVE_Edge").set("xmin", Box_Width/2 - b*10^6 - ...
    0.001);
model.component("comp1").selection("RVE_Edge").set("ymin", Box_Height  - b*10^6 - ...
    0.001);
model.component("comp1").selection("RVE_Edge").set("ymax", Box_Height + 0.001);
model.component("comp1").selection("RVE_Edge").set("xmax", Box_Width/2 + b*10^6 + ...
    0.001);
model.component("comp1").selection("RVE_Edge").set("condition", "intersects");
RVE_Edges = model.selection("RVE_Edge").entities();


% RVE Grains
model.component("comp1").selection.create("RVE_Grain", "Box");
model.component("comp1").selection("RVE_Grain").set("entitydim", 2);
model.component("comp1").selection("RVE_Grain").set("xmin", Box_Width/2 - b*10^6 - ...
    0.001);
model.component("comp1").selection("RVE_Grain").set("ymin", Box_Height  - b*10^6 - ...
    0.001);
model.component("comp1").selection("RVE_Grain").set("ymax", Box_Height + 0.001);
model.component("comp1").selection("RVE_Grain").set("xmax", Box_Width/2 + b*10^6 + ...
    0.001);
model.component("comp1").selection("RVE_Grain").set("condition", "intersects");
RVE_Grain = model.selection("RVE_Grain").entities();

% Definitions
```

```
731 model.component("comp1").func().create("an1", "Analytic");
732 model.component("comp1").func("an1").set("fununit", "Pa");
733 model.component("comp1").func("an1").set("argunit", "m");
734 model.component("comp1").func("an1").setIndex("plotargs", "b", 0, 2);
735 model.component("comp1").func("an1").setIndex("plotargs", "-b", 0, 1);
736 model.component("comp1").func("an1").set("funcname", "P_dist");
737 model.component("comp1").func("an1").set("expr", ...
        "if(x/b>-1,if(x/b<1,P_max*sqrt(abs(1-(x/b)^2)),0),0)");
738 % Add physics
739 phys = comp1.physics.create('solid', 'SolidMechanics', 'geom1');
740 model.component("comp1").physics("solid").feature("lemm1").set("IsotropicOption", ...
        "Lame");
741 % Add Material
742 E_0 = 200*10^9;
743 nu  = 0.3;
744
745 lambdaLame_0 = (E_0*nu)/((1+nu)*(1-2*nu));
746 mhuLame_0    =  E_0/(2*(1+nu));
747 lambdaLame = lambdaLame_0*ones(size(P,1)+1,1);
748 mhuLame    = mhuLame_0*ones(size(P,1)+1,1);
749
750 for o = 1:length(P)+1
751  mattag = ['mat' num2str(o)];
752  mat = model.component('comp1').material.create(mattag);
753  model.component('comp1').material(mattag).selection.set(o);
754  mat.materialmodel('def').set('density', 7800);
755  model.component('comp1').material(mattag).propertyGroup().create('Lame', 'Lame ...
         parameters'); % Change e in Lame parameters to e with acute accent
756  model.component('comp1').material(mattag).propertyGroup('Lame').set('lambLame', ...
         lambdaLame_0);
757  model.component('comp1').material(mattag).propertyGroup('Lame').set('muLame', ...
         mhuLame_0);
758 end
759 % Create fixed constraint
760 model.component("comp1").physics("solid").create("fix1", "Fixed", 1);
761 model.component("comp1").physics("solid").feature("fix1").selection()...
762     .named("geom1_edge_L");
763 model.component("comp1").physics("solid").create("fix2", "Fixed", 1);
764 model.component("comp1").physics("solid").feature("fix2").selection()...
765     .named("geom1_edge_R");
766 model.component("comp1").physics("solid").create("fix3", "Fixed", 1);
767 model.component("comp1").physics("solid").feature("fix3").selection()...
768     .named("geom1_edge_D");
769
770 % Add boundary load
771 model.component("comp1").physics("solid").create("bndl1", "BoundaryLoad", 1);
772 model.component("comp1").physics("solid").feature("bndl1").selection().named("geom1_edge_U");
773 model.component('comp1').physics('solid').feature('bndl1').set('LoadType', ...
        'FollowerPressure');
774 model.component("comp1").physics("solid").feature("bndl1").set("FollowerPressure", ...
        "P_dist(X-step)");
775 % Meshing
776 mesh1 = comp1.mesh.create('mesh1');
777 ftri1 = mesh1.feature.create('ftri1','FreeTri');
778 mesh1.feature('size').set('hauto',6);
779 mesh1.run;
780 % Adding study
781 std = model.study.create('std1');
782 std.feature.create('stat', 'Stationary');
783 model.study("std1").feature("stat").set("useparam", true);
784 model.study("std1").feature("stat").setIndex("punit", "m", 0);
785 model.study("std1").feature("stat").setIndex("plistarr", ...
        "range(3*b,(7*b-(3*b))/(nsteps-1),7*b)", 0);
786 model.study("std1").feature("stat").setIndex("pname", "step", 0);
787 %% Stop the study when crack reaches surface
788 % Properties
789 m     = 10.1;
790 tau_r = 6.113 * 10^9;
791 D_del = 0.2;
792
793 D_edge          = zeros(geom1.getNEdges,1);
```

```
del_shear       = zeros(geom1.getNEdges,1);
Crack           = zeros(geom1.getNEdges,1);
D_grain         = zeros(length(P)+1,1);
del_sheargrain  = zeros(length(P)+1,1);
Dloop           = zeros(length(P)+1,1);
Dloopedge       = zeros(geom1.getNEdges,1);

s               = 1;
Ncrack          = 0;
N               = 0;
conlist         = [];
CrackInitiation = [];
Cycle           = [];

while any(ismember(Crack,Surface)) ≠ 1
 tic
 % Run study
 model.study("std1").run();
 if s == 1
  %% Line Average
  % Edge
  for l = 1:geom1.getNEdges
   if ismember(l,RVE_Edges)
    % Create the average values for edges in RVE
    avtag = ['av' num2str(l)];
    model.result.numerical().create(avtag, 'AvLine');
    model.result.numerical(avtag).setIndex('expr', ...
        '-0.5*(solid.sx-solid.sy)*sin(2*atan2(nY,nX))+solid.sxy*cos(2*atan2(nY,nX))', ...
        0);
    model.result.numerical(avtag).selection().set(l);
   else
   end
  end
 end
 % Retrieve all data from edges
 for i = 1:geom1.getNEdges
  if ismember(i,RVE_Edges)
   if isnan(D_edge(i))
    del_shear(i,:) = nan;
   else
    avtag  = ['av' num2str(i)];
    % Retrieve data
    av_val = model.result().numerical(avtag).computeResult();
    % Calculate shear stress difference during one loadcycle
    del_shear(i,:) = max(av_val(1,:)) - min(av_val(1,:));
   end
  else
  end
 end
 %% Damage rate of all edges
 D_rate  = (del_shear./(tau_r*(1-D_edge))).^m;
 % Max damage rate
 [D_ratemax,I] = max(D_rate);
 % Calculate jump in cycles
 N_del = D_del/D_ratemax;
 N     = N + N_del
 Cycle(s) = N;
 % Damage on all edges
 D_edge = D_edge + D_rate*N_del;
 Dloopedge(:,s) = D_edge;
 %% Crack creation
 for j = 1:length(D_edge)
  if ismember(j,RVE_Edges)
   if D_edge(j) ≥ 1 % If damage larger than 1
    Ncrack = Ncrack + 1;
    if Ncrack == 1
     % Crack initiation
     CrackInitiation = N;
     % Add Crack
     model.component("comp1").physics("solid").create("crack1", "Crack", 1);
     % Settings of crack node
```

```
        model.component("comp1").physics("solid").feature("crack1").set("CrackSurface", ...
            "FromGeometry");
        % Add Contact node for friction
        model.component("comp1").physics("solid").create("cnt1", "SolidContact", 1);
        % Settings of Contact node
        model.component("comp1").physics("solid").feature("cnt1").create("fric1", ...
            "Friction", 1);
        model.component("comp1").physics("solid").feature("cnt1").feature("fric1")...
            .set("FrictionModel", "Coulomb");
        model.component("comp1").physics("solid").feature("cnt1").feature("fric1")...
            .set("mu_fric", 0.4);
       end
       % Save the crack and crack order
       Crack(j) = j;
       Crackorder(Ncrack,:) = [j,N];
       % Fill in crack node
       model.component("comp1").physics("solid").feature("crack1")...
           .selection.set(nonzeros(Crack));
       model.component("comp1").physics("solid").feature("crack1")...
           .selection("Face1").set(nonzeros(Crack));
       D_edge(j) = nan;
       % Create contact selection
       contag = ['p' num2str(Ncrack)];
       conlist = [conlist;{contag}];
       model.component('comp1').pair.create(contag,'Contact');
       model.component('comp1').pair(contag).source.set(j);
       model.component('comp1').pair(contag).destination.set(j);
       % Add to contact node
       model.component("comp1").physics("solid").feature("cnt1").set("pairs", conlist);
      end
     else
     end
    end
    %% Damage of grain
    % Grain data gathering
    for k = 1:length(P)+1
     if ismember(k,RVE_Grain)
      % 1 is the outer box, can be skipped
      seltag = ['sel' num2str(k)];
      mattag = ['mat' num2str(k)];
      faces = model.selection(seltag).entities();
      D_edges = D_edge(faces);
      D_edges(isnan(D_edges)) = 1;
      D_grain = mean(D_edges);
      lambdaLame(k) = (E_0*(1-D_grain)*nu)/((1+nu)*(1-2*nu));
      mhuLame(k)    = (E_0*(1-D_grain))/(2*(1+nu));
      if D_grain == 1
       lambdaLame(k) =  1;
       mhuLame(k)    =  1;
      end
      Dloop(k,s) = D_grain;
      mat = model.component("comp1").material(mattag);
      model.component('comp1').material(mattag).propertyGroup('Lame').set('lambLame', ...
          lambdaLame(k));
      model.component('comp1').material(mattag).propertyGroup('Lame').set('muLame', ...
          mhuLame(k));
     else
     end
    end
    s = s+1
    toc
  end
  save(filename,'P','P_Mirror','v','c','Box_Width','Width','Box_Height','Height','E_0',...
  'P_max','b','D_del','Cycle','N','CrackInitiation','Crack','Crackorder',...
  'Dloop','Dloopedge','nsteps')
end
```