Chair of Mechanics

Wear of dry-running piston rod sealing rings: modelling and experiments

November 2018

Dipl.-Ing. Andreas Kaufmann, BSc

Affidavit

I declare on oath that I wrote this thesis independently, did not use other than the specified sources and aids, and did not otherwise use any unauthorized aids.

I declare that I have read, understood, and complied with the guidelines of the senate of the Montanuniversität Leoben for "Good Scientific Practice".

Furthermore, I declare that the electronic and printed version of the submitted thesis are identical, both, formally and with regard to content.

Andreas Kaufmann

Matriculation Number: 00835288

Date, 15.11.2018

Abstract

This work presents a predictive calculation tool for the wear of dry-running piston rod sealing elements, called “packing rings”, which are placed into the pressure packing of reciprocating compressors. Packing rings are performance determining components in two ways: i) they are key to an efficient operation during their lifetime and ii) they cause very expensive downtime and loss of production if failing unexpectedly. The rings’ duty is to seal the compression chamber towards the ambient. “Dry-running” applications are tribologically most challenging because lubrication must not be used.
Due to their working principle the packing rings are inevitably subject to wear. Proper sealing performance over the rings’ lifetime is only achieved by geometrically compensating wear.

Time and cost intensive laboratory testing is important and industry standard, but several quantities of interest (stress, strain, contact pressure, wear pattern, etc.) are hardly or not at all accessible via experiments. Finite element (FE) calculations allow to gain insight into these quantities and their evolution with time, which is key to better understanding and further improving packing rings.
The modelling and the simulation procedure are compared against analytical models and purpose-developed experiments proving the reliability and predictive capabilities of the numerical approach. This FE tool allows for the first time to bridge the gap between laboratory testing and real-world applications.

Kurzfassung

Diese Arbeit präsentiert eine Berechnungsmethode zur Verschleißvorhersage für trockenlaufende Kolbenstangendichtelemente, genannt “Packungsringe”. Packungsringe sind leistungs- und effizienzbestimmende Bauteile in reziprokierenden Kompressoren: zum einen müssen die Ringe Dichtheit gewährleisten, um einen Kompressor effizient betreiben zu können, zum anderen führt ein plötzliches Versagen der Dichtringe zu ungeplanten, kostenintensiven Stillständen und Produktionsausfällen. Die Ringe haben die Aufgabe, die Kompressionskammer von der Umgebung abzudichten. Am tribologisch anspruchsvollsten sind “trockenlaufende” Anwendungen. Hier ist die Verwendung von Schmierstoffen prozessbedingt untersagt.
Durch ihre Funktionsweise verschleißen Packungsringe zwangsläufig. Dieser Verschleiß muss geometrisch kompensiert werden, um über die gesamte Lebenszeit entsprechende Dichtwirkung zu zeigen.

Zeit- und kostenintensive Laborversuche sind wichtig und derzeit Industriestandard. Manche auslegungstechnisch relevanten Größen (Spannung, Dehnung, Kontaktdruck, Verschleißbild, etc.) sind jedoch kaum oder nicht experimentell zugänglich. Finite Element Rechnungen erlauben Einblick in diese Größen und deren Evolution mit der Zeit. Dadurch wird tiefgehenderes Verständnis erlangt und eine effiziente Weiterenwicklung der Packungsringe ermöglicht.
Die Verschleißberechnung und die dahinterliegenden Modelle werden mit analytischen Modellen und Experimenten verglichen, somit kann die Zuverlässigkeit und die Vorhersagekraft dieses Berechnungswerkzeuges gezeigt und erstmals eine Verbindung zwischen Labortests und echter Anwendung geschaffen werden.

Acknowledgments

First I thank HOERBIGER for the opportunity to conduct my PhD thesis. I was (and still am) allowed to work in an interesting, challenging field of research that connects thermodynamics, analytical and mechanical modelling, numerical simulation, test rig design, material testing and component testing.

Heartfelt thanks goes to my supervisor Prof. Thomas Antretter Head of the Institute of Mechanics. He turned out to be not only a supervisor, but a true mentor by combining his great in-depth simulation and modelling knowledge with the ability to find time for fruitful and directive discussions whenever necessary. I also thank all colleagues from the Institute of Mechanics for their help, especially Dr. Michael Toifl, Aleksandr Sannikov and Dr. Robert Kaiser. Special thanks goes to Karl Flicker, who helped occasionally setting up calculations on the cluster.

On the side of HOERBIGER I thank Dr. Bernhard Spiegl and Dr. Tino Lindner-Silwester who initiated and enabled this project. Special thanks to Dr. Lindner-Silwester and Dr. Matthias Kornfeld for creating a working environment, which allowed me to focus, to work without unnecessary (management) disturbances and to grow with the project. I thank them for many open, directive, fruitful and productive discussions and their effort especially with the analytical modelling. Due to their positive, encouraging critique during the recent years and while writing papers they enabled me to perform this work.
I thank the whole HOERBIGER R&D team for their support in gaining insight into compressor technology. I want to express my gratitude to the R&D workshop team (Martin Lagler, Alexander Jandl, Christopher Habetler, Martin Molnar, Dominik Ratschka) for their support preparing test specimen and packing rings, building test rigs, maintaining them and executing tests. Here special thanks goes to conscientious Alexander Jandl, who - despite occasional honest disagreements - simply made things work when they initially did not.
Heartfelt thanks goes to Dr. Marian Janko-Grasslober. He helped me a great number of times to re-focus and brainstorm. Thank you for your unbroken positive spirit and for encouraging me.

I thank Dr. Thomas Dick from tortuetec GmbH. His expertise in modelling and material testing improved this work.

At last I thank my family: my parents Johann and Christa for their huge support through all this years of studying. My brother Martin for his hospitality (at the beginning of this adventure) and the occasionally given input on programming. My beloved wife Margarita for her incredible tolerance and patience and my daughter, Sophia, who shows me - again and again - that the world is full of smaller and bigger wonders.

Contents

Chapter 1

Introduction

Sealing elements are performance determining components in reciprocating compressors in two ways: i) they are key to an efficient operation during their lifetime and ii) they cause very expensive downtime if they fail unexpectedly.

The sealing elements of a pressure packing, i.e., the so called “packing rings” are inevitably subject to wear, while sealing the compression chamber towards the ambient. In this work the packing rings will also be simply referred to as “rings”. Their working principle is “self-energizing”, i.e. their seal effect arises from the differential gas pressure to be sealed [44]. The wear has to be compensated by the ring geometry to ensure proper sealing performance over the rings’ lifetime.

In many applications packing rings are lubricated, which decreases wear and increases the rings’ lifetime. But there are applications where no gas contamination (e.g. by a lubricant) is tolerated, e.g. the food industry, when the gas is of highest purity or in chemical processes. Such non-lubricated applications are called “dry-running” or “non-lube” applications. They are tribologically very challenging, since the lack of lubrication poses greater demands on the rings’ material and design.

The tribology of each and every compressor packing is unique, even for packings in what is considered the “same” application. Field experience shows that sometimes out of two redundant machines one is a trouble maker, whereas the other one performs well. Nonetheless the lifetime of a dry-running packing ring is roughly between 2000 and 8000 h, which makes the rings often the maintenance interval determining component. Obviously there is a strong demand for longer lifetime of dry-running packing rings in the industry.

Common industry practise is to perform experiments on modified test compressors with purpose-designed highly instrumented packings. This allows to benchmark ring designs as well as materials to a certain extent. But by the nature of any tribological problem even very small occurring differences in the test set-up lead to (to a certain degree) different results. For example the contact surface preparation, the batch-to-batch difference of the same material grade or the manual finishing (deburring, lapping) of a packing ring have unknown influence on the tribological system. These experiments are important, but time and cost intensive. Furthermore, most experiments are a black-box to a certain extent. Not all quantities of interest can be measured during the runtime, like contact pressures or the wear pattern. The latter can only be determined at the end of a wear experiment.

This work presents a simulation tool, which bridges the gap between laboratory testing and real-world applications. Careful modelling combined with FE calculations allows to gain insight into many quantities which are hardly or not at all accessible via experiments: contact pressures, the wear pattern, stresses and strains. Additionally the simulation allows to track these quantities over time.
An analytical model with a simplified ring configuration validates the correct numerical implementation of the wear algorithm. Furthermore, the model is experimentally validated with the intention to prove the reliability and predictive power of the calculation tool.

Chapter 2

State of the art

This chapter briefly presents the state of the art of sealing elements in a reciprocating compressor with the focus on the packing rings, their geometry and their materials. Further detailed information can be found online on the homepage of any notable rings and packings producer or compressor packager, e.g. Ariel Corporation, Burckhardt Compression, Castanet-SA, Cook Compression, Compressor Products International (CPI), HOERBIGER, Howden-Thomassen, Kranz - Packungen Kuhrmeier GmbH, Stasskol GmbH.

Over time different compressor types have been developed, which can be split into two basic groups: positive displacement and dynamic compressors, see Figure 2.1. Both groups have subgroups with different working principles, advantages and problems [21, 27, 81]. This work focuses on reciprocating compressors.

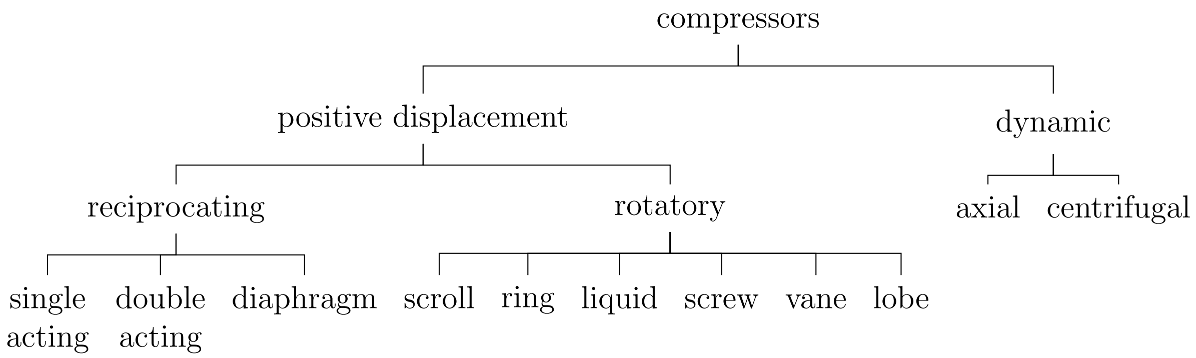


Figure 2.1. Compressor types

In reciprocating compressors the following *dynamic* sealing elements are used: valves, piston rings, packing rings and wiper rings. Figure 2.2 shows a compressor-cut. The valves, the piston with the piston rings and the rider rings, the rod with the pressure packing including the packing rings, the intermediate packing, and the wiper rings sealing are visible.

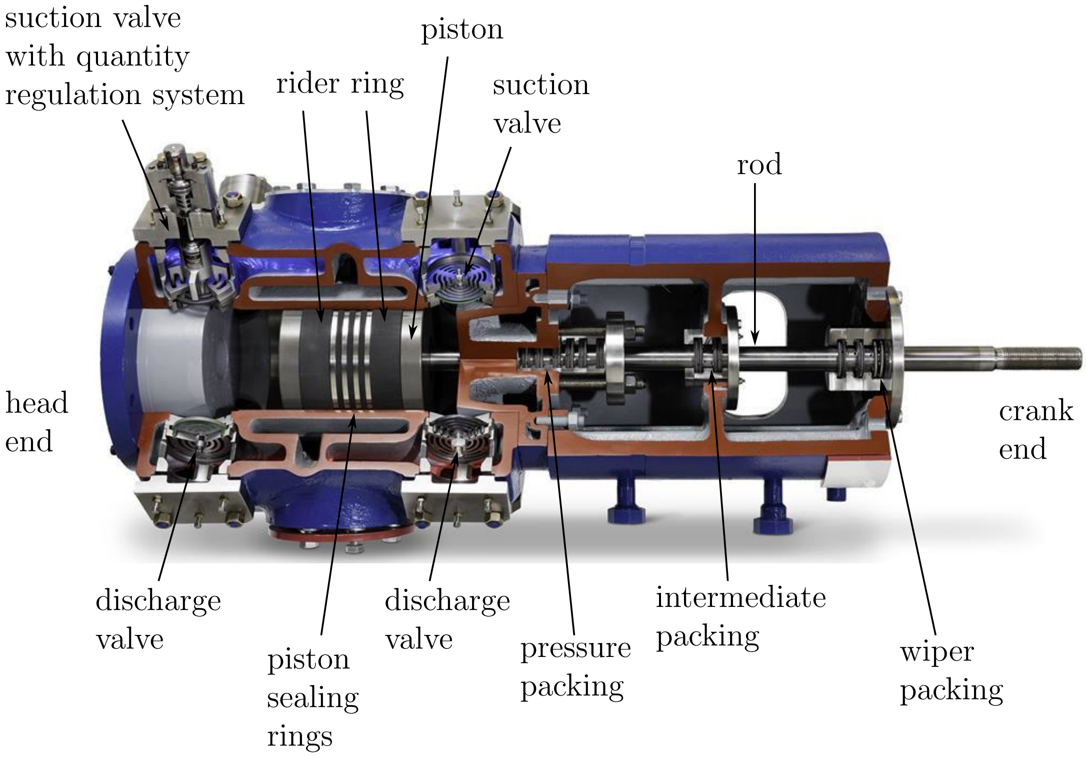


Figure 2.2. Cut of a compressor including the piston rings, the rider rings, the pressure packing, the intermediate packing and the wiper packing. All packings consist of multiple rings. The photo taken is from [63].

The term *dynamic* is valid for valves and rings, but the meaning is different. Valves move dynamically between seat and guard but seal statically. The valve kinematic is characterized by high accelerations and subsequent impacts. Only in one position, when the valve contacts the seat, the valve seals.
For rings *dynamic* refers to the relative motion between ring and rod (packing & wiper rings) or ring and liner (piston rings). Due to the manifold application areas of compressors the configurations of the rings within compressors vary with the application. The ring design and the number of rings vary depending on the gas type, the compression ratio, the end pressure, the materials used, the lubrication, and the gas delivery quantity. Wiper rings seal the crank case and prevent oil leakage. These rings work in lubricated or semi-lubricated mode and are mentioned for completeness only, since this work focuses on the wear of dry-running packing rings.

Piston and packing rings face a very similar situation during application. Both ring types have to seal the compression chamber. They have the same self-energized working principle inevitably leading to wear. Further both seal types work under difficult conditions: The gas pressure to be sealed varies periodically between suction and discharge pressure, the relative velocity changes in magnitude and direction and with it the contact situation changes permanently.

One of the major differences between packing and piston rings is that for non-vertical machines the piston rings have to carry the load of the piston, whereas the packing rings are pressed against the rod by the gas pressure and garter spring only. To achieve higher sealing performance and longer service time the piston rings consist of two groups: the piston sealing rings and the rider rings. The latter should only carry the piston weight without sealing. If a rider ring seals, it wears faster due to the higher contact pressure. When designing a rider ring, the axial width is determined by the contact pressure and the (limited) construction space. The contact pressure should be as low as possible to increase lifetime. Piston sealing rings seal at their outer diameter, whereas the packing rings seal at their inner diameter. Thus the design of rings is adapted to their function.

Another major difference between piston and packing rings is the possibility of cooling the contact partners during operation. A cooling circuit can be included into the cylinder surrounding the liner. This leads to an effective cooling system for the compression chamber and the piston rings. Cooling the packing housing is possible but by far not that effective. Packing rings, which are usually made out of polymeric materials, insulate the dynamic contact surface well [22]. Some attempts have been made to include a cooling system into the rod [74], but these systems are not commonly used or commercially available.

Leakage in the piston sealing system is in double-acting compressors less critical than in the packing sealing system, because the gas still remains inside the compressor, i.e. only efficiency is lost. Conversely, leakage in the packing sealing system leads to a loss of gas.

2.1 Piston rings

Piston sealing rings are cut into one, two or three pieces. A cut can be manufactured straight, under a certain angle, or with an overlap, Figure 2.3. The latter version allows to manufacture gas-tight piston rings. They are oversized in order to compensate wear. Bores on the inner diameter and a groove on the outer diameter lead to a pressure-balanced design with lower wear. Such a design is used for high compression ratios.

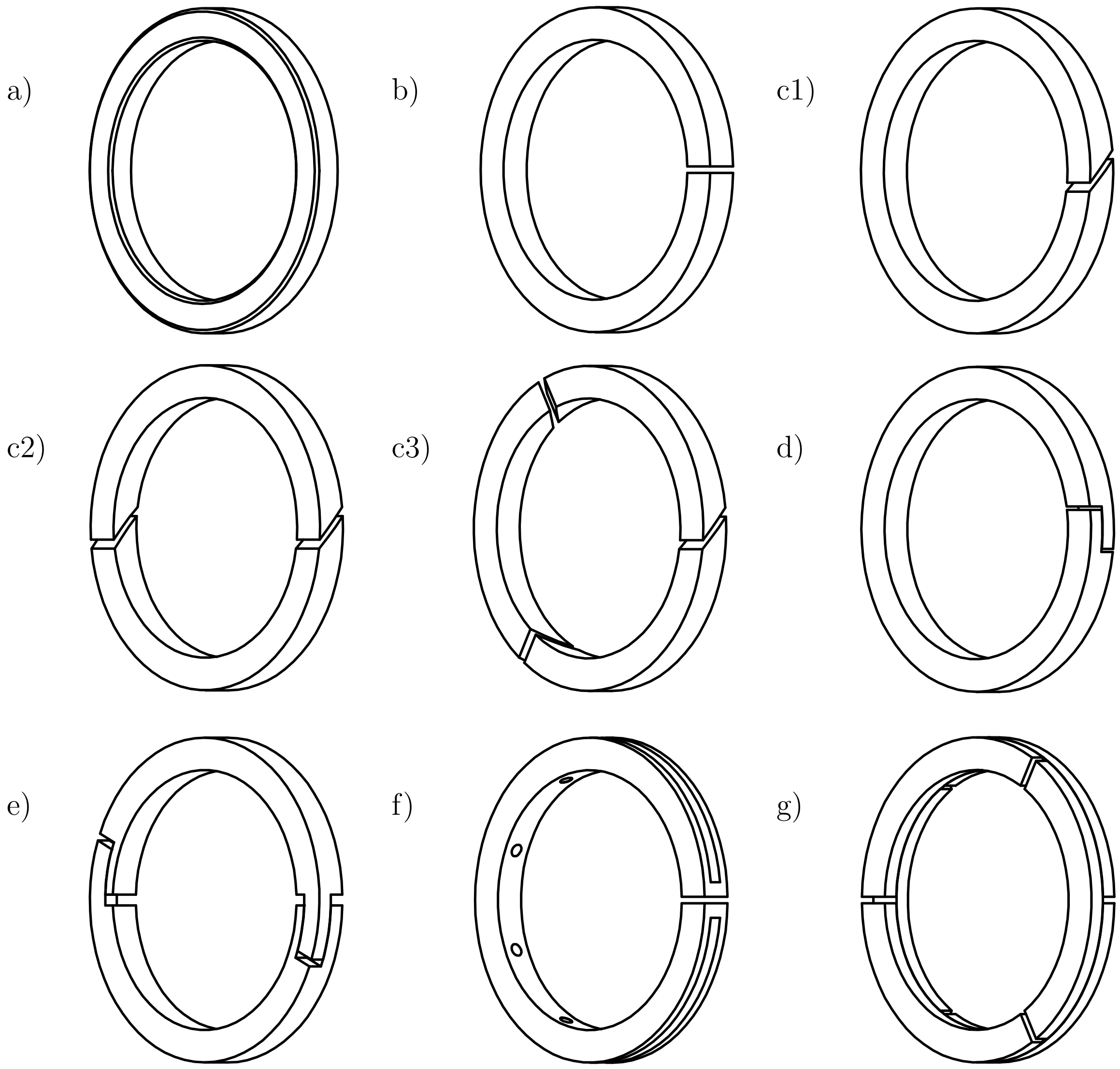


Figure 2.3. Piston ring designs: a) uncut ring, b) one piece radially straight-cut ring, c) angle-cut rings: c1) one piece, c2) two piece or c3) three piece design, d) one piece overlap/step-cut ring, e) two piece gas-tight cut ring, f) radially straight-cut ring with pressure-balancing groove, g) radially straight-cut ring pair.

The simplest (and cheapest) rider rings are solid rings, which are shrink-fitted onto the piston. To ensure easy installation the rider rings are cut either straight or with an angle. Radial and axial pressure relieve grooves avoid a too high pressure difference across the axial ring's width. Figure 2.4 presents an overview of typical single-piece rider rings. The rider rings are also available as two piece rings, radially or angularly cut. To prevent rotation of the ring an additional hole is provided for the peg on the piston.

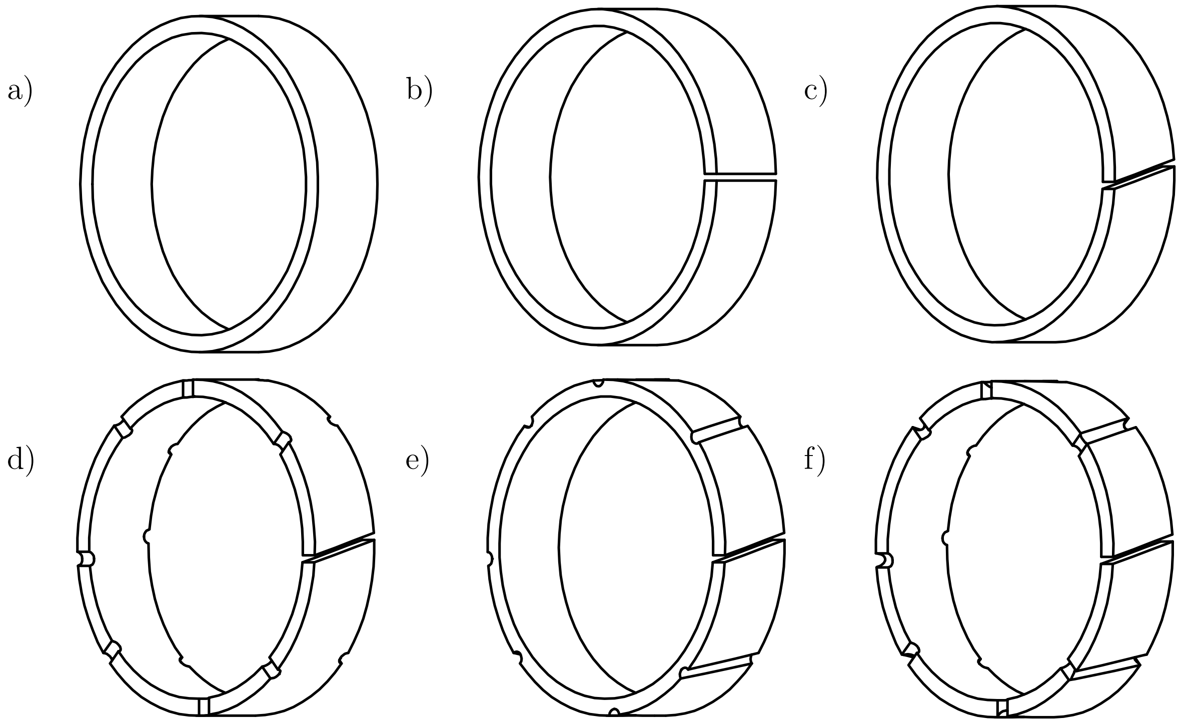


Figure 2.4. Rider ring designs for single piece rings: a) uncut ring, b) radially straight-cut ring, c) angle-cut ring, d) angular-cut ring with radial pressure-relieve grooves, e) angular-cut ring with axial pressure relieve grooves, f) angular-cut ring with radial and axial pressure relieve grooves. Similar versions of d) to f) can be found with the cut and the pressure relieve grooves in straight axial direction. Two-piece rings have another cut shifted by 180°.

2.2 Pressure packing rings

In the pressure packing several rings and/or ring pairs seal the compression chamber to the environment. The number of sealing elements within a packing is usually 3 to 8 [33], but even packing cases with more than 10 sealing elements are in the field. Additionally, depending on the pressure level, the first ring on the cylinder side is a pressure breaker with defined leakage gaps. Its name originates from its function to break down the dynamic pressure peaks, see Figure 2.2. Pressure breakers can be designed as radially contacting or non-contacting labyrinth seals depending on the application. The contacting designs are for light and the non-contacting design for heavy gases. They are either single-piece uncut rings or (mostly three-piece) radially-cut rings. The latter have either no bore clearance and axial gaps or they are designed as zero-gap with a bore clearance, i.e. the inner diameter of the pressure breaker is slightly bigger than the rod diameter. Pressure breakers can be single or double-acting. The former have radial slits on their high pressure side to relieve any pressure from the packing back into the cylinder. The shape and the number of the radial slits vary dependent on the diameter, operating conditions and manufacturer. The segments of a multi-piece pressure breaker are held together by a garter spring.

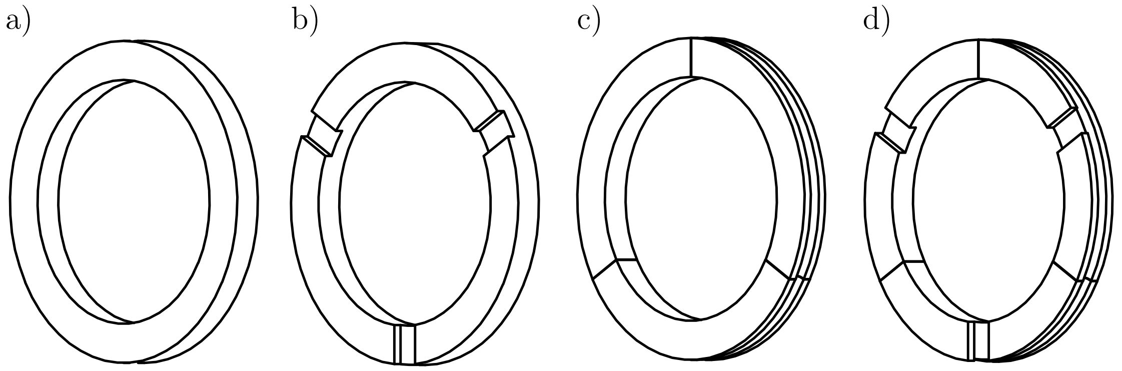


Figure 2.5. Most common pressure breakers: a) single piece, double acting b) single piece, single acting, and c) three piece, double acting d) three piece, single acting. The designs can have either i) bore clearance and no gap between the segments or ii) no bore clearance and gap between the segments for three piece rings. The bore clearance or gap are small compared to the ring's other dimensions.

The geometry of seal rings is often more complex. A single seal ring usually consists of several pieces held together by a garter spring, Figure 2.6. Standard ring groups are e.g. the single-acting radial-tangential ring pair (a+b or a+c in Figure 2.6) or the double-acting tangential-tangential ring pair (b+b or c+c). In contrast to these ring pairs the balanced cap design (BCD) ring (f) is a single ring, which allows to build shorter packings. Further designs and combinations of rings or ring pairs can be found in the literature [22, 27].
The geometric diversity and complexity of packing rings arises from the need of two basic features: Firstly, every cut or gap in the design has to be sealed, either by a second ring or by another ring segment. Secondly, the occurring wear has to be compensated to maintain the sealing performance [17, 23]. Most seal rings have an initial "wear gap", which closes during operation due to wear.
This work presents the wear of packing rings on the example of the BCD. Figure 2.7 shows the basic geometric features of a BCD ring, including the wear gap, the garter spring groove and the pressure balancing groove. The wear gap and the pressure balancing groove are designed such that when the wear stop is reached the pressure balancing groove has worn off entirely. The idea is that the ring is pressed adequately to the rod without losing sealing efficiency even when the ring's circumferential stiffness sharply increases due to closing of the wear gaps. If the pressure balancing groove has disappeared the ring is pressed more strongly against the rod.

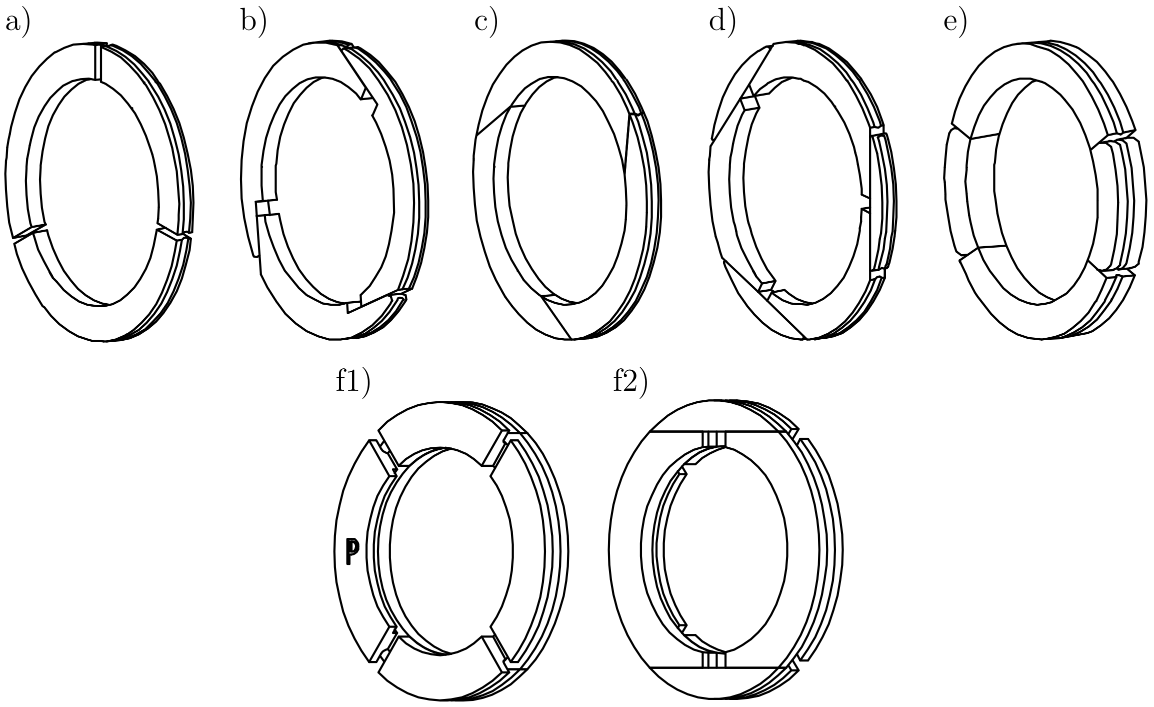


Figure 2.6. Overview of seal rings in a packing: a) 3 piece radial cut ring, b) 3 piece tangential cut ring with wear stop, c) 3 piece tangential to rod cut ring without wear stop, d) 6 piece tangential cut ring ("bridge"), e) 4 piece ring design f) balanced cap design (BCD) ring (f1 shows the pressure side, f2 the sealing side).

Current ring designs are mostly made from high-performance polymer-based composites, with highly temperature dependent material properties. The arising contact temperature in non-lubricated compressors has great influence on the packing ring stiffness. Between rod and packing housing is a clearance, which prevents contact and damage of these parts, but allows the rings to creep in with increasing temperature. This deformation can disable the ring's ability to compensate wear. A backup ring is used to reduce this clearance. This prevents creeping in most cases and increases the thermal coupling of the ring to the cooler packing housing. A backup ring is usually made from metal, but also high performance polymers are used.

Backup rings are usually either solid, i.e. single piece, uncut rings or three piece radially cut rings, see Figure 2.8. The rod-ring clearance is a few tenth of a millimetre. The radial cuts have to be finished precisely to prevent leakage. Generally the leakage of a packing ring (pair) is dominated by the leakage through manufacturing imperfections.

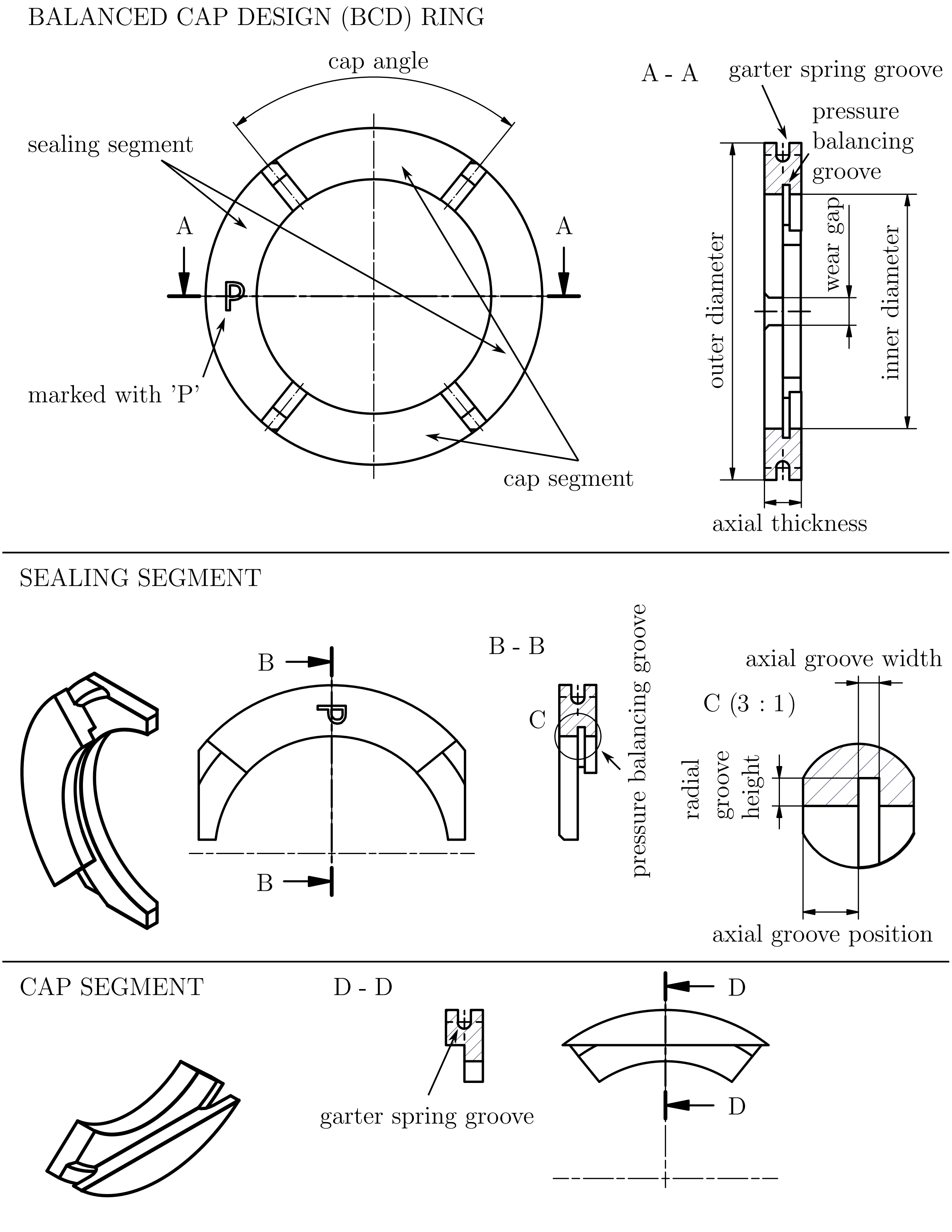


Figure 2.7. The BCD consists of two cap and two sealing segments. The wear gap, the pressure balancing groove and the garter spring groove are marked. A “P” marks the pressure side to prevent reverse installation.

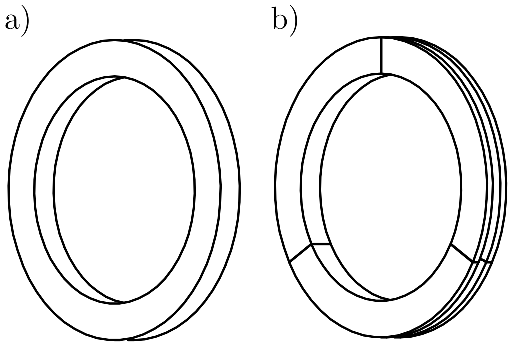


Figure 2.8. Overview of backup rings in a packing: a) solid (i.e. single piece, uncut) backup ring, b) 3 piece radial cut backup ring.

2.3 Wiper rings

Wiper rings (or scraper rings) have the task to seal the crank case. They prevent the oil from leaking along the rod into the pressure packing housing. These rings are working in lubricated or semi-lubricated mode only. Well designed wiper rings scrape the oil off the rod, which is then returned to the crank case. A bigger outer diameter allows to stop splashing oil from the crank case, Figure 2.9a), whereas a smaller radial dimension allows higher contact pressures on the scraping edges, Figure 2.9b).
The necessary contact pressure results from the garter springs only, since wiper rings are not loaded by any gas pressure. The contact pressure has to be high enough to scrape off the oil but not too high to prevent wear [27]. Since no gas pressure should act on a wiper ring, it may move axially inside its cup. This movement, induced by the rod, is called "shuttling". To prevent damage from shuttling some wiper rings or ring groups are side-loaded with springs.

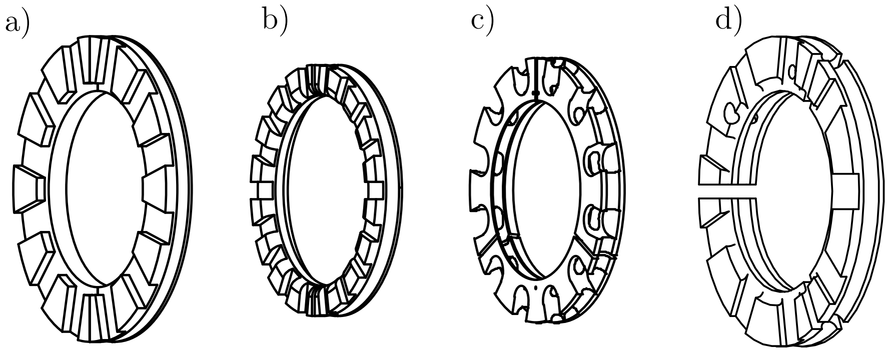


Figure 2.9. Oil wiper designs: a) oil wiper with bigger outer diameter to block splashing oil, b) oil wiper with smaller outer diameter to increase scraping performance, c) and d) profiled oil wiper ring with two scraping edges each.

2.4 Materials

Rings are made of various materials (metals, polymers and polymer compounds), depending on the operation conditions and their function. Metallic rings can be found in i) lubricated service in older compressors as cylinder rings, ii) in both lubricated and non-lubricated service as backup rings and iii) in rare instances even sealing rings are partly made of metal.
Today most compressors are equipped with polymer or polymer compound sealing rings. The change from metal to polymer based rings has the benefit of protecting the counter part (liner or rod) from wear. Liner and rod are much harder than a polymeric ring and thus hardly subject to wear during operation, which saves maintenance costs.

Polytetrafluoroethylene (PTFE) is an excellent tribological material. Its wear properties can be improved by orders of magnitude if used as a compound. Current compounds consist of a polymeric matrix like poylether ether ketone (PEEK), polyphenylene sulfide (PPS) or PTFE or a compound of these and fillers like bronze, carbon, carbon fibres (CFs), coke, graphite, glass fibres (GFs), molybdenum disulfide (MoS_2), etc. Often the compounds are highly filled, with a filler content of more than 40 wt%.

The literature review below gives an overview of relevant mechanical and tribological properties of pure PTFE, several PTFE compounds and the effect of fillers on a compound's tribology.

Many papers report on the mechanical and/or tribological behaviour of PTFE and/or PTFE compounds. The mechanical properties have been determined in tension and compression mode, with different specimen types and with different test parameters. The variety in the published tests makes comparison difficult, but helps to gain insight into the material's behaviour. Tribological investigations are even harder to compare, since the number of different parameters is greater. The tribological papers differ not only in the reported test methods, the specimen geometries but also in the test parameters, e.g. relative velocities ranging from $\mu m\,s^{-1}$ [9] to $m\,s^{-1}$ [71].

The literature can be grouped into several categories:

- The type of motion:
    - reciprocating (either circumferentially or linearly)
    - unidirectional (pin-on-disk or block-on-ring)
    - torsional.
- The counter surface is either
    - the same material as the specimen or
    - a different material, mostly steel.

The counter surface characteristics are hardly found to be equal. Although the surface preparation and finishing methods seem similar, the result is most of the time different e.g. surface roughness, surface hardness, surface profile.

- The specimen type varies with the test set-up. Bhushan [5, chapter 11] gives a general overview of wear testing set-ups including a short description. The test methods listed below are the set-ups used in the reviewed papers.
    - The pin-on-disk set-up seems the most common test method. Here the pin is placed one one side of the rotating or reciprocating disk. In this category different specimen types and set-ups are used:
        - Mostly the pin is made of the (polymeric) specimen material [3,6,7,38,42, 46,71,72,76,78,84],
        - sometimes the disk is ,made of the investigated polymeric material [37,69, 83] and
        - rarely both, pin and disk, are from the investigated material [34].

      A further sub-categorisation is based on the pin form:
        - usually the pin is cylindrical [6,7,38,42,46,69,71,72,76,78,84],
        - sometimes the contacting end is spherical [3,37,83].
    - In the block-on-ring (also pin-on-cylinder [5, chapter 11]) set-up a block is pressed against the outer diameter of the ring [39,41].
    - Other less popular set-ups are closer to an application:
        - the disk-on-disk set-up [34] (also called thrust washer [5, chapter 11])
        - ring-on-ring set-up [68]
        - bearing tester [73]
        - self-developed set-ups, which could be called “ring-on-rod” set-ups [12,47, 84].
- Test parameters used:
    - relative speed
    - applied load/contact pressure
    - temperature, which is hardly controlled but often measured
    - test atmosphere:
        - humidity
        - fluid (mainly gas) type
        - temperature
        - static gas pressure

The mechanical and tribological properties of pure PTFE are for many applications (also for rings and packings) not sufficiently stable. The low strength, high tendency to creep and high wear rate are problematic. These properties can be improved with fillers. Before addressing compounds and fillers several knowledgeable facts concerning pure PTFE are highlighted.

2.4.1 Pure PTFE

Domininghaus [20, pages 543-565] gives a good overview over the historical development, the mechanical and tribological properties and the different applications of PTFE. The reviewed papers are divided into two partly overlapping groups: firstly, papers which focus on the mechanical properties [11, 15, 34, 58, 59, 64, 79] and the effects of morphology and crystallinity [7, 15, 16, 38, 58, 59, 64] and secondly, papers which study friction [3, 6, 12, 39, 56, 69, 70, 77, 86] and wear [6, 12, 38, 39, 66, 69, 70, 76–78, 86]. In some references pure PTFE is used as a reference for the tested compounds.

In comparison to other commodity and engineering polymers, PTFE is a very heavy polymer. The theoretical values for the density of fully amorphous and fully crystalline PTFE are 2040 and 2300 $\mathrm{kg\,m^{-3}}$, respectively [59]. Usual density values of PTFE are between 2150 and 2200 $\mathrm{kg\,m^{-3}}$ [20, 59].

The transition temperatures of PTFE at atmospheric pressure are [20, 37, 59]

- -97 °C - glass transition temperature of amorphous PTFE
- 19 °C - crystal transformation: triclinic crystals below and hexagonal crystals above
- 30 °C - crystal transformation: hexagonal crystals below and pseudohexagonal crystals above
- 127 °C - glass transition temperature of amorphous PTFE from solid to under-cooled liquid state
- 313 °C - melting temperature for the amorphous phase
- 326 - 331 °C - melting temperature for the crystalline phase

The crystal transformation temperatures around room temperature have to be kept in mind when performing material testing.

The following sheds light on the mechanical properties of pure PTFE, considering the influence of hydrostatic pressure, the load-mode dependency of the mechanical properties, the effect of the strain rate and the effect of temperature. The end of this section presents some tribological aspects of PTFE.

The influence of hydrostatic pressure on the material properties was already investigated in the early 1970's [15, 64]. Christiansen et al. [15] investigated the effect of hydrostatic pressure on the stress-strain behaviour of PTFE. The effect on the yield stress is represented by the factor -0.07, which gives the gradient of the yield stress as a function of the hydrostatic pressure.
Utilizing this factor the influence of a hydrostatic pressure load on a sealing ring in a compressor can be estimated. The yield stress is 2.76 MPa at 54 °C, whereas a hydrostatic pressure of 5 MPa would add only 0.24 MPa (10 %).
The yield stress and the bulk modulus change over the tested hydrostatic pressure range due to a morphological change. A schematic phase diagram of PTFE can be found in

either [59, Fig. 1] which shows the four known phases of PTFE between 10 and 120 °C or [20, Bild 2-331] where only three solid phases are distinguished over a temperature range of -100 to 500 °C. Both show pressures up to 800 MPa. The triple point is at 70 °C and 450 MPa [20].

Sauer, Mears and Pae [64] found that the yield strength and the elastic modulus increases while the strain at fracture decreases with increasing hydrostatic pressure up to 517 MPa. The compressive yield stress increases linearly over the pressure range. The compressive elastic modulus increases linearly, but has a discontinuity near 550 MPa due to a phase transition. These increases are explained with the free volume theory: with increasing pressure PTFE loses free volume and becomes more brittle. The amorphous regions become more glass-like and less rubbery. The degree of cold drawing is reduced and strain hardening increases. The strain at fracture decreases. If a crack forms, no plastic tearing occurs and rupture follows. Sauer et al. stated that the Coulomb yield criterion predicts that the yield stress should be greater in compression than in tension, which fits to the presented measurements.

Rae and Dattelbaum [59] investigated PTFE's compressive properties and found that the Poisson's ratio varies between 0.4 and 0.5. A trend towards 0.5 was found for strains greater than 5 %, which agrees with earlier reported data by Sauer et al. [64]. Later Rae and Brown [58] published a comparison of compression and tension data in a low strain regime (max. 2 % strain). PTFE shows an asymmetric behaviour. Under compression the material is stiffer than under tension. The Poisson's ratio is inexplicably mode dependent, $\nu = 0.36$ under tension and $\nu = 0.46$ under compression.

The compressive tangent modulus is highly temperature dependent and considerably lower if measured at larger strains, approximately 500 MPa for large strains [59]. This lower compressive modulus is possibly better suitable for engineering purposes. Further Rae and Dattelbaum found that [59, (page 7624)] "During deformation, PTFE stores much more of the work done as structural change (≈ 30 %) than typical metals (< 10 %). This energy is presumably used in crystallographic and amorphous chain rearrangement resulting in less sample heating than a comparable metal and therefore lowering the thermal softening effect of 'self' heating."

Rae and Brown [58] published that the tensile properties of PTFE are highly strain rate and temperature dependent, whereas crystallinity plays a minor role. Higher loading rates and lower temperatures result in a stiffer material response. Rae and Brown give a detailed discussion on the temperature failure mechanisms of PTFE and their micro-structure dependency. They see the micro-structure of PTFE as "three component compound" consisting of crystalline, amorphous and quasi-ordered materials. The latter are either close to a crystalline region in the material or an amorphous region which rearranged under load. At low loading rates ($0.001\,\mathrm{s}^{-1}$) the strain hardening happens at the same rate in compression and tension (approximately $0.25\,\mathrm{MPa}\,\%^{-1}$ above a strain level of 10 %). The true stress level in compression is approximately 3 MPa higher than in tension above 10 % strain. The effect of hydrostatic pressure published by Christiansen [15] is approximately 1/10 of the difference in yield stress at this load level. Thus there has

to be another reason for this asymmetry.
From the data presented by Rae and Brown [58] (two PTFE types and a strain rate of 0.005 s^{-1}) the dependence of the stress level on the temperature is estimated. At 10 % strain the true stress at 25 °C is 11 MPa, at 150 °C it is 3 MPa. This gives a temperature dependency of 0.06 MPa °C^{-1}. The rate decreases with increasing temperature. A temperature change of 1 °C has approximately the same impact on the mechanical properties as 1 MPa hydrostatic pressure. This shows how sensitive PTFE reacts to temperature changes.
Cyclic compression tests [11] showed similar effects. The compression modulus depends on the loading rate and the ratcheting strain depends on the mean stress. The linear increase of the compression modulus as well as the higher ratcheting strains at low loading rates indicate that, if possible, PTFE is subject to creep and/or relaxation.

Some tribological aspects of pure PTFE, like the ability to form a transfer film, are discussed in the following section together with the tribological aspects of PTFE compounds. The following sheds some light on frictional behaviour μ of PTFE and the high dependency on the test parameters. The static and the dynamic coefficients of friction are nearly equal for PTFE, no stick-slip occurs [20, 73]. The temperature dependence of the coefficient of friction is presented by Pleskachevsky and Smurugov [56] for a PTFE-steel contact pair. Starting with $\mu = 0.23$ at room temperature the coefficient of friction drops to a minimum of $\mu = 0.13$ around 130 °C and then increases again to $\mu = 0.16$ at 170 °C (block-on-ring, load 400 N, speed 0.5 m s^{-1}).
Babuska et al. [3] presented the temperature dependency of PTFE against PTFE and against MoS_2 in bone-dry atmosphere under 1 N and 3 mm s^{-1} in a ball-on-flat set-up. The coefficient of friction increases from 0.15 at −150 °C to a maximum of 0.25 at −50 °C and then decreases to 0.07 at room temperature. Between 50 °C and 125 °C μ is approximately 0.05, then it increases again to 0.08 at 150 °C.
Blanchet and Kennedy [6] showed that PTFE against 316 stainless steel has a velocity dependent coefficient of friction, when loaded with 6.55 MPa. At approximately 0.01 m s^{-1} μ is 0.02, increasing v to 0.1 m s^{-1} gives μ=0.1 and at 0.2 m s^{-1} $\mu = 0.18$. Although the coefficient of friction does not significantly change at 0.1 m s^{-1}, the wear rate increases by two orders of magnitude.
Briscoe et al. [9] found that the PTFE-steel contact pair has a nearly constant coefficient of friction of 0.06 between 25 °C and 250 °C (load 10 N, speed 0.45 mm s^{-1}). At 0.16 m s^{-1} μ decreases from 0.35 to 0.15 when increasing the load from 25 to 75 N.
A decrease of μ from 0.23 to 0.17 was detected between room temperature and 90 °C by Smurugov et al. [70], (steel counterface, load 0.25 MPa, speed 0.125 m s^{-1}).
Shibo et al. [69] showed the motion dependency of the coefficient of friction with a steel ball on a PTFE disk. In ascending order μ is 0.1, 0.118 and 0.12 for unidirectional motion, reciprocating motion and torsion, respectively.
Tanaka and Kawakami [72] found that μ increases from 0.2 to 0.23 if the sliding speed is increased from 0.1 to 2.5 m s^{-1} for a load of 10 and 50 N.

So dependent on the test parameters the coefficient of friction varies for PTFE over an order of magnitude between 0.02 and 0.35 This shows the sensitivity of a PTFE based tribological system.

2.4.2 Fillers

Fillers and their effect on PTFE compounds have been studied over decades by many groups worldwide. The compounds can be divided into three categories, for each a list containing the filler type, the filler content and the literature source are noted. Table 2.1 gives an overview of the studied compounds with a single inorganic filler. Table 2.2 lists compounds with 2 or more inorganic fillers. Table 2.3 considers compounds of PTFE and another polymer. Polymeric compounds often use PTFE as a filler to improve the tribological behaviour. The presented compounds usually have a wear rate which is at least 1-2 orders of magnitude below the wear rate ($10^{-3}\mathrm{mm}^3\,\mathrm{N}^{-1}\,\mathrm{m}^{-1}$) of pure PTFE. Preferably, wear rates should be below $10^{-8}\mathrm{mm}^3\,\mathrm{N}^{-1}\,\mathrm{m}^{-1}$ (e.g. [10]).

The compounds presented in the literature usually have a lower coefficient of wear than pure PTFE. It is difficult to compare the presented compounds, since they differ in the filler type and content, the manufacturing processes as well as the test equipment and the test parameters. In the following only some key aspects are highlighted.

The fact that PTFE forms a transfer film on the counter surface makes this polymer attractive for tribological applications. The transfer film reduces friction and wear, but its formation and appearance is influenced by many parameters. It depends on the filler type and shape, the test parameters (load, velocity, type of motion) and the counter surface properties [6, 9, 12, 16, 37–39, 42, 56, 68, 69, 72, 77, 82, 83, 86].
The formation of the transfer film takes some time and is commonly referred to as "running-in" period. In this period the wear rate is usually higher than in the following steady-state [38, 75]. The running-in can take from several minutes [75] to a day [73]. Metals as filler promote the film formation [86]. The influence of the type of motion was studied by Shibo et al. [69], comparing unidirectional rotation, linear reciprocating and torsional motion. Under torsional motion the transfer film is thicker and fractured, here debris is not expelled as in the other two types of motion, where the pin does not stay at the same location.
Under water no transfer film is formed, thus the wear rate is significantly higher [38].
Close investigations on the formation and the effect of sliding speed and contact pressure have been carried out by Li et al. [42]. They found that a thinner film has a higher wear resistance. A thin film can be obtained by lower speeds and higher pressures, but the surface quality and wear resistance of the transfer film is higher at lower pressures. Furthermore, a certain surface roughness is conducive to a strongly bonded transfer film.
Tevrüz [73] found that the film thickness increases over time and with increasing load. Furthermore, a thick, continuous and smooth film gives a low coefficient of friction, which is in agreement with Conte and Igartua [16].
An investigation by Uruena et al. [77] showed the transfer film alone is not the reason for ultra low wear, it is a combination of polymer and transfer film. The wear of transfer films was tested with a micro-tribometer and is orders of magnitudes greater than the wear of the polymer/counter surface pair. The influence of the testing direction and the number of cycles forming the transfer films varied. The testing direction has no significant influence. After a certain number of over-travels a proper transfer film establishes. Its wear resistance does not show significant improvement with more frequent over-travels.

Table 2.1. PTFE compounds with a single inorganic filler

filler type	amount (wt%)	source	annotation
glass (fibre)	25	[6]	k increases with increasing velocity, GF prevents surface failure
	25	[16]	
	25	[38]	GFs perform better than glass beads (hollow or solid) and glass flakes
	15, 25	[37]	
	25	[72]	lowest k at high load
	40	[65]	significant counter surface damage by the fibres
graphite	15	[6]	flakes form, k is velocity independent
	15	[16]	
	0-50	[83]	above 40 % filler k increases sharply, lowest k at 20 %
	15	[72]	
carbon or	30	[84]	CF reinforced carbon matrix compounds, as well as CF compounds
carbon fibre	35	[73]	
	25	[16]	
bronze	40	[6]	flakes, k decreases with increasing velocity
	40	[72]	low k if the load is low
	60	[16]	
MoS_2	20	[72]	
	0-50	[83]	above 40 % filler k increases sharply, lowest k at 10 %
miscellaneous			
Al	5	[77]	aluminium particles
AlN	1-5	[79]	
$CaAlSi_3O_5$	n.a.	[75]	amount of man-made calcium aluminosilicate fibres not specified
Cu	30	[86]	
	45	[42]	
$K_2Ti_4O_9$	1-25	[68]	potassium titanate whiskers
$K_2Ti_6O_{13}$	1-25	[68]	potassium titanate whiskers
Ni	30	[86]	
$(OH)_8Si_4Al_4O_{10}$	0-15	[82]	ultra-fine kaolin particles
Pb	30	[86]	
Si_3N_4	1-5	[79]	
TiO_2	1-25	[68]	whiskers performed better than particles
	20	[72]	
UFD	1-10	[39]	ultra fine diamond (UFD)
ZrO_2	40	[72]	

Table 2.2. PTFE compounds with multiple inorganic fillers

filler type and amount (wt%)	source
carbon (18) + graphite (7)	[37]
CF (6) + graphite (8) + MoS_2 (2)	[47]
GF (5) + MoS_2 (5)	[71]
GF (15) + MoS_2 (3)	[16]
GF (20) + MoS_2 (5)	[37]
GF (20) + graphite (5)	[16]
modified GF (5) + Pb (60)	[12]

Table 2.3. PTFE compounds with another polymer (compound)

second polymer	amount (wt%)	source	annotation
FEP	35, 50	[7]	fluorinated ethylene propylene (FEP)
PA	70	[34]	polyamide (PA); mainly investigated glass fibre filled PA
	80	[60]	PA66
PEEK	0 - 100	[9]	poylether ether ketone (PEEK), 10-15 % PTFE in PEEK performs best
	0 - 100	[10]	optimum at 15-25 % PEEK in PTFE
	80	[65]	commercial grade: PEEK is already filled with 10 % carbon fibres, the 20 % are PTFE and graphite
	0 - 100	[46]	optimum at 10-20 % PTFE in PEEK, lowest wear rate at 5 % and lowest μ at 15 % PTFE in PEEK
PPDT fibres	2.5, 10	[37]	poly-*p*-phenyleneterephtalamide (PPDT)

Pleskachevsky and Smurugov [56] suggested that (pure) PTFE (against steel) has an operation point where most efficient self-lubrication occurs. At this operation point (temperature) the load bearing ability and tribological characteristics of the transfer film are at the optimum. This is in agreement with the findings of Blanchet and Kennedy [6]: the wear regime of pure PTFE can change abruptly from mild to severe wear, when increasing the sliding speed at a given contact pressure.
Smurugov et al. [70] presented an explanation for the thermally activated wear mechanisms of PTFE.

Invesigating different filler types, Tanaka and Kawakami [72] showed that the aspect ratio (l/r) of fillers is important. The higher the aspect ratio the higher is the filler's load-supporting action. This was confirmed by Khedkar et al. [37], adding that a higher volume

fraction of (glass) fibres increases the thermal stability and conductivity and effectively decreases the wear rate. Shi et al. [68] also reported that whisker-filled PTFE composites show a lower wear coefficient than compounds which are filled with particles of lower aspect ratio. Furthermore, they added that their high aspect ratio fillers prevents the dominant wear mechanism for pure PTFE: adhesion.
On the other hand Tanaka and Kawakami [72] reported that lamellar fillers as graphite or MoS_2 and particle like fillers like metal oxides have an extremely small aspect ratio or an aspect ratio about unity, respectively. These fillers do not have a significant load-supporting action. (This was confirmed by Blanchet and Kennedy [6].)

Additionally, the filler size is important [72]. Too small particles are not well incorporated in the matrix material and can easily be transferred to the transfer film on the counter part. Bigger fillers and fibres are firmly embedded into the matrix and can rub against the counter surface without being pulled from the matrix and thus reduce wear. On such "bigger" fillers like bronze or GF a PTFE film was found. This film reduces the wear of the fillers and thus the wear of the whole compound. Tanaka and Kawakami [72] concluded that the wear rate of PTFE compounds depends only weakly on the type and shape of the filler, if the filler has suitable wear resistance and size (i.e. several microns e.g. 30 µm).
Zhang et al. [86] used Cu, Pb and Ni powder as filler and reported an increase of the load-supporting action. It can be concluded that grain size of the metal powder is "big enough" to increase the wear resistance.
Klaas et al. [38] compared the effects of solid and hollow glass beads, glass flakes and GF as filler. They found that the compounds with solid glass beads, hollow glass beads or glass flakes produce a thicker transfer film and experience a higher wear rate than the compound with GF. They reported that the GFs polish the counter surface and produce a uniform transfer film at a low wear rate. The wear rate decreases further when adding a (not specified) solid lubricant. It can be guessed that the solid lubricant is either MoS_2 or graphite. This corresponds to the earlier findings of Cheng et al. [12] that the (modified) GFs increase the ability to form a transfer film and the transfer film's adhesion to the countersurface.
Xiang and Gu [82] confirmed Tanaka's statement concerning too small particles. They found that too many ultra-fine kaolin particles result in a non-uniform transfer film. This is likely due to abrasion by agglomerated particles.
Khedkar et. al [37] found that carbon particles as filler can plough the PTFE matrix, destroy the transfer film leading to three-body abrasion. Graphite does not hinder this process.

Tanaka and Kawakami [72] considered that compared to MoS_2, graphite adheres more strongly to the matrix and thus a PTFE+graphite compound exhibits less wear than a PTFE+MoS_2 compound. A benefit of MoS_2 was reported by Khedkar et al. [37] and Conte et al. [16]: MoS_2 may prevent the direct contact of GF and the transfer film and thus the abrasive effect of GF is reduced. So MoS_2 helps to reduce wear rate and maintain a low coefficient of friction. Conte et al. compared different compounds and found that the substitution of 5 % out of 25 % GF by the same amount of graphite results in lower wear. The substitution of 10 % GF by only 3 % MoS_2 outperforms the compound with

25 % GF.
Friedrich et al. [26] reported that carbon fibres perform better than glass fibres.

The coefficient of friction of PTFE based composites is generally independent on the filler type and similar to unfilled PTFE. This fact is ascribed to the formation of a transfer film [72]. The independence of the filler shape was confirmed by Shi et al. [68], adding that the crystal structure of the filler has a high influence on the coefficient of friction.
Yang et al. [84] presented an interesting study on packing ring materials. They claim that the coefficient of friction does not automatically mean lower frictional force and power consumption at a given constant load. They suggested that the coefficient of friction depends on the contact pressure, the sliding velocity, the temperature, the surface hardness and possibly other factors. Tzanakis et al. [75] studied the coefficient of friction of a PTFE compound filled with calcium aluminosilicate in detail. The speed was varied between 0.25 and 0.5 $\mathrm{m\,s^{-1}}$, the load between 0.4 and 1.6 N in 4 steps and the roughness of a high carbon steel between R_a 0.125 and 0.7 µm in 4 steps. The influence of these factors on the contact temperature is: i) Doubling the load increases the temperature by about 35 % of the initial value (in °C). ii) Doubling the sliding velocity increases the temperature by about 50 %. iii) The increase of temperature with roughness is more complicated: increasing the roughness profile from R_a 0.125 to 0.25 µm adds 17 % to the contact temperature regime, but from R_a 0.25 to 0.5 µm it adds 30 %, whereas an increase from 0.5 to 0.7 µm just adds 3 %.

Blanchet and Kennedy [6] found that fillers also govern the size and shape of wear fragments. Larger particles can turn a subsurface crack's trajectory to the sliding surface, leading to finer debris.
In contrast unfilled PTFE exhibits mild or severe wear, depending on the test parameters. In the severe wear regime debris stripes are found around the wear tracks. The morphology of the transferred material turns to fragmented sheets. These are removed from the contact zone by the oscillatory motion of the disk [6]. Similar results were found later [37].

Mamaev et al. [47] investigated the compressive behaviour of a PTFE compound after exposing it to air or oil for 3 or 6 months at 20 or 100 °C temperature and uniform compression of 4 or 16 MPa. None of the three independent factors (environment, temperature and pressure) has a dominating influence on the wear rate, but the combination of these factors has a significant impact. The elastic modulus, elongation at break and wear rate does not change significantly even after an exposure of 6 months (approximately 10 %). They concluded that their compound (c.f. Table 2.2) is fit for the service as a long-term piston sealing.

In combination with other polymers PTFE is used as a filler to decrease a compound's coefficient of friction and wear by forming a transfer film. In combination with the high-temperature polymer PEEK several studies have been published [9,10,46,65]. Each group reported that the compound performs tribologically better than the original materials. The lowest wear rate can be found at 5 % PTFE in PEEK, but the lowest coefficient of friction (0.25) is found at 15 % PTFE in PEEK, Lu et al. [46]. In comparison Briscoe et al. [9] found the lowest coefficient of friction to be about 0.1 (over a broad range of mixtures between 10 and 80 %) and showed that μ depends on the load and the operating conditions. They stated (and agreed with Lu et al.) that the optimum between mechanical and tribological properties is between 10 and 15 % PTFE in PEEK. The latter has the lower coefficient of friction, but is mechanically weaker. In contradiction Burris and Sawyer [10] found the optimum wear rate at 20 % PEEK in PTFE. The coefficient of friction varied between 0.135 and 0.111 for the compounds. They also presented the development of the wear rate and coefficient of friction over the reciprocating cycles: The former decreases, the latter increases.

2.5 Gas dryness

The choice of material for a non-lubricated application also depends on the dryness of the compressed gas. The moisture within a gas can work as lubricant or lead to significantly higher wear rates. Three different types of dryness are differentiated by the gas' dew point: wet, dry and bone-dry in Table 2.4.

Table 2.4. Gas dryness and typical examples [32]

name	dew point °C	example compression process
wet	> -20	air, chemical industry (CO_2)
dry	-20 > and > -40	air separation (O_2), chemical industry (CO_2), natural gas (turbine fuel gas)
bone-dry	< -40	air separation (Ar, He, N_2), natural gas (liquid natural gas boil off)

Chapter 3

Mechanisms governing wear

Archard's wear equation (3.1) is most frequently used in the literature when modelling wear. It assumes that the wear rate $\frac{\mathrm{d}w}{\mathrm{d}t}$ is proportional to the contact pressure p_c and the relative velocity v of the contact partners. The proportionality factor, the wear coefficient k, has to be determined experimentally.

$$\frac{\mathrm{d}w}{\mathrm{d}t} = kp_\mathrm{c}v \tag{3.1}$$

We assume that Archard's equation is valid for the wear of packing rings. The next section elaborates how to obtain the true, wear-relevant solid-solid contact pressure.
The relative velocity can directly be calculated from the crank mechanism, stroke and the running speed of the crank drive. For the wear calculation we assume that a mean velocity acts over time. Section 3.3 presents the considerations and thoughts on the time scales behind this assumption.

3.1 Contact pressure

The gas pressure on the cylinder side p_cyl and on the crank side p_crk load a packing ring, we assume $p_\mathrm{cyl} > p_\mathrm{crk}$. The pressure differential leads to contact between ring and rod and ring and cup. Due to this pressure differential the ring seals p_cyl from p_crk. This working principle is called "self-energizing", which is defined by Lindner-Silwester and Hold [44] as "the seal effect arises from the differential gas pressure to be sealed".
Figure 3.1a shows ring, cup and rod and additionally the gas pressure loads on a sealing ring. The contact pressure distribution $p_\mathrm{c,flat}$ at the sealing surfaces can be calculated using a FE solver, if the loads, the ring geometry, the material properties and the contact situation are known. This can be done in a straightforward way assuming that all contact surfaces are not only nominally but also microscopically perfectly flat. Hence, no gas can creep into the sealing surfaces, see Figure 3.1b.
In reality the surfaces of the contact partners are nominally, but not microscopically flat. Thus gas can creep into the gaps between the contact partners and a gas pressure drop

arises from p_{cyl} to p_{crk}. Figure 3.1c shows a gas pressure distribution p_{g} in the sealing surfaces assuming that surface roughness exists. The considerations concerning this gas pressure distribution are detailed in the following section 3.2.

The true solid-solid contact pressure $p_{\mathrm{c,rough}}$ can be calculated by taking all gas loads on the free *and* the sealing surfaces into account. The gas pressure distribution in the sealing surfaces p_{g} reduces the previously calculated contact pressure $p_{\mathrm{c,flat}}$

$$p_{\mathrm{c,rough}} = p_{\mathrm{c,flat}} - p_{\mathrm{g}}. \tag{3.2}$$

The resulting true solid-solid contact pressure $p_{\mathrm{c,rough}}$ is highlighted green in Figure 3.1d. Proportionally to $p_{\mathrm{c,rough}}$ the ring's dynamic sealing surface wears. No wear occurs between ring and cup, since the relative velocity of these parts is zero. Figure 3.1e shows the wear relevant contact pressure distribution $p_{\mathrm{c,rough}}$ acting on the ring.

For later use, convenience and the convention used in equation (3.1) the true contact pressure is renamed to

$$p_{\mathrm{c}} := p_{\mathrm{c,rough}}. \tag{3.3}$$

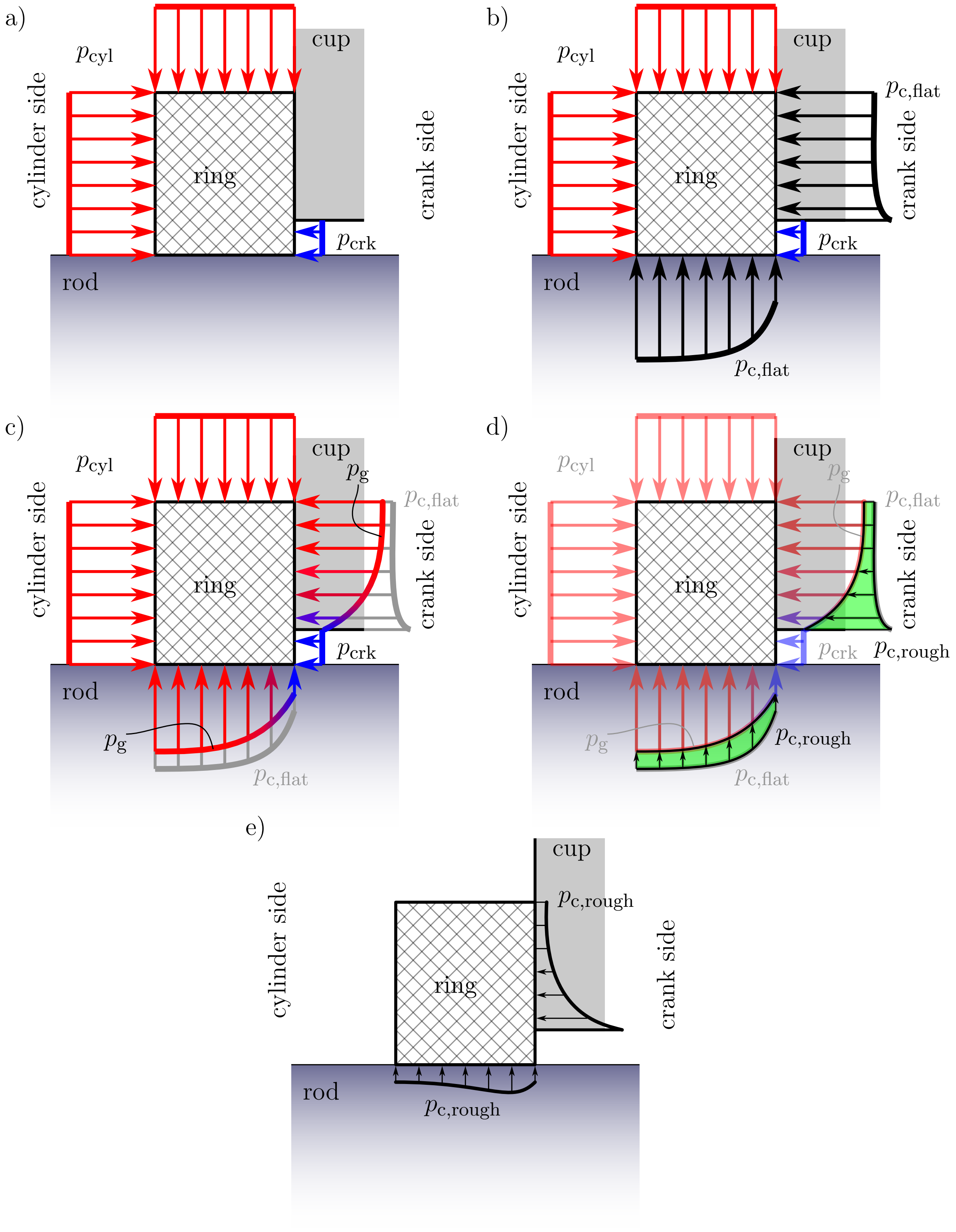


Figure 3.1. Mechanisms governing ring wear. a) Gas pressure load on ring p_{cyl} and p_{crk}. b) Contact pressure $p_{\mathrm{c,flat}}$ assuming perfectly flat contact surfaces. c) Gas pressure p_{g} in the sealing surfaces is added, which results from surface microroughness. d) The true solid-solid contact pressure $p_{\mathrm{c,rough}}$ (highlighted green) takes the gas pressure drop p_{g} into account. e) $p_{\mathrm{c,rough}}$ to which wear is proportional, if relative motion occurs.

3.2 Gas pressure distribution in a sealing surface

Here the investigation of Lindner-Silwester [43, section 2.3.1] concerning the gas pressure distribution in the sealing surfaces of packing rings is summarised:

On a microscopic level the surfaces of ring and rod are rough. The surface roughness results in a gap between the parts and allows gas to creep in. The gap is then replaced by an equivalent constant gap independent of x, y, see Figure 3.2. With a gap height of a few microns, an order of magnitude estimation reveals that inertia effects are negligible. Hence, the flow, commonly referred to as Hele-Shaw flow [80] in the incompressible case, is governed by the balance between frictional forces and pressure forces. In this limiting case the absolute gas pressure p_g is uniform over the gap height so that $p_\mathrm{g} = p_\mathrm{g}(x, y)$. When density variations are negligible which holds true for $\frac{p_\mathrm{cyl} - p_\mathrm{crk}}{p_\mathrm{cyl}} << 1$, the pressure distribution is governed by

$$\nabla^2 p_\mathrm{g} = 0. \tag{3.4}$$

When the pressure differential $p_\mathrm{cyl} - p_\mathrm{crk}$ across the ring can no longer be neglected compared to p_cyl, gas compressibility effects have to be taken into account. In the limiting case of negligible inertia, i.e. kinetic energy, the adiabatic flow under consideration is to a good approximation an isothermal one. For an ideal gas the density is then linearly proportional to the absolute pressure p_g, which together with the continuity equation leads to

$$\nabla^2 {p_\mathrm{g}}^2 = 0. \tag{3.5}$$

For the numerical implementation the fact will be taken advantage of that “The equations for many different physical situations have exactly the same appearance.” as is described in Volume II, chapter 12, section 1 “The same equations have the same solutions” [24]. Steady heat flow problems in isotropic, homogeneous solid bodies of constant thermal conductivity are also governed by Laplace’s equation

$$\nabla^2 T_g = 0. \tag{3.6}$$

Having the thermal conductivity vary in a linear way with the temperature T_g results in

$$\nabla^2 {T_g}^2 = 0. \tag{3.7}$$

This allows to use standard FE code to calculate the pressure distribution in the sealing surfaces by treating it as heat transfer problem.

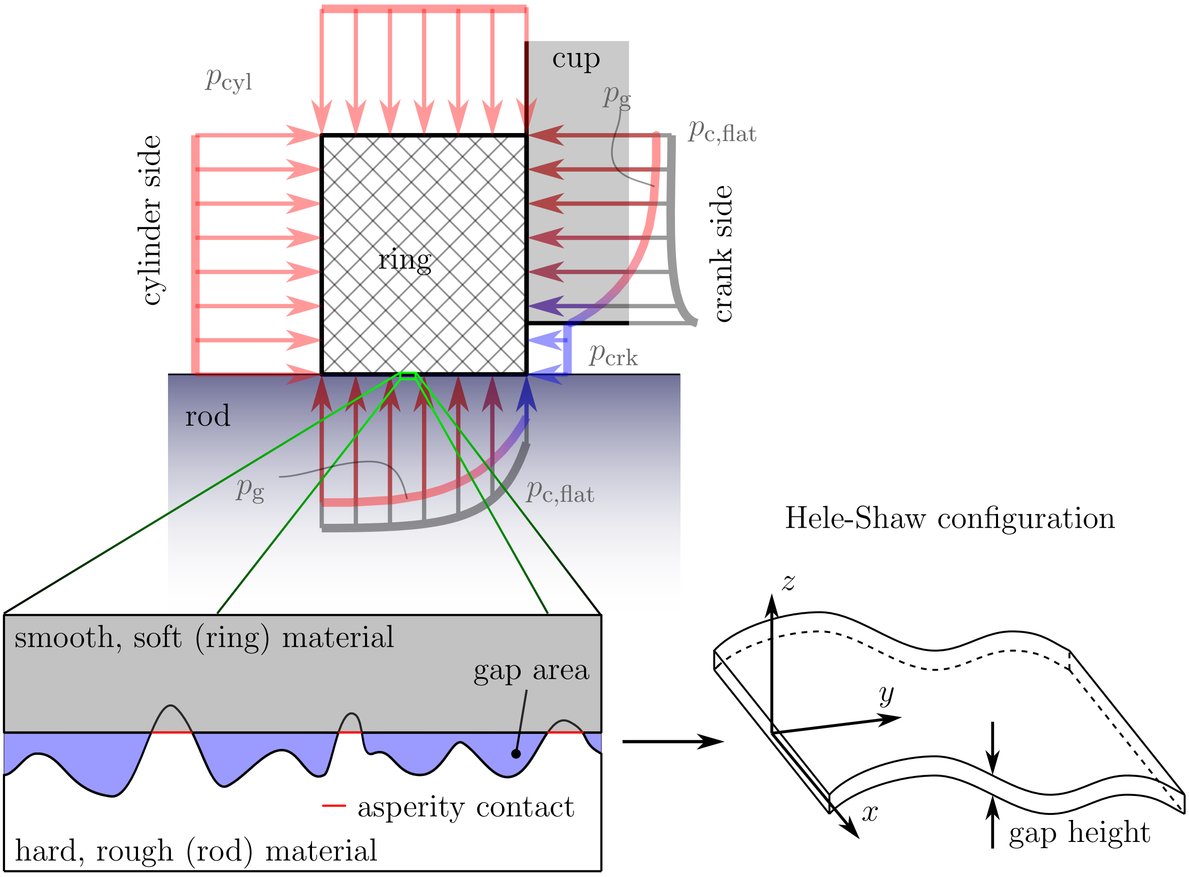


Figure 3.2. Zoom into the dynamic sealing surface. The softer ring contacts the harder rod, the surface roughness results in a gap. Here all surface roughness is depicted for the rod. The gap is assumed to be of constant height and allows to consider the gas flow in a Hele-Shaw configuration.

3.3 Consideration of time scales

This section presents the two physical time scales on which the wear process of rings occurs. A *fast time scale* t which is of the magnitude of a crank revolution and a *slow time scale* T which is of greater magnitude representing the wear process. The time scales are matched and two examples show that the wear law on the slow time scale is valid and allows to calculate wear over time correctly with highly reduced computational effort.

Note: In this section 3.3 and all its subsections the naming convention is slightly changed to increase readability: the tilde denotes dimensional quantities, all non-tilde quantities are non-dimensional.

Acknowledgement: This section was elaborated in cooperation with Dr. Matthias Kornfeld.

3.3.1 Background and idea of the model

While maintaining sealing performance the packing rings wear over millions of strokes. The wear is compensated geometrically and thus the sealing element geometry changes slowly over time. These changes are slow compared to the time scale in which the piston moves. Every single stroke involves a highly complex wear process due to the changes in velocity and pressure. But the wear due to a single stroke is barely detectable and a geometric change can hardly or not at all be measured.
The problem must be tackled with two different time scales: One is defined by the service interval of the sealing elements, the *slow time scale.* This slow time scale is important when considering long-term material effects (e.g. creep) and the wear of the sealing element. Furthermore, reciprocating compressors are running with several hundred (approximately 200-2000, application dependent) strokes per minute and with every stroke the sealing element wears. This process of wearing is physically occurring on the *fast time scale* of a single stroke. Nevertheless, in this work regardless of the time scale the wear problem is dealt with on a macroscopic level, similar to Archard's considerations [1].
The computational costs for modelling wear over the sealing element's lifetime are enormous, if every single stroke has to be resolved. Even with massive computational resources, such a calculation will take longer than the real-world wear process. Furthermore modelling contact and wear in FE software is complex and may lead to convergence issues. Thus it is beneficial or even necessary to reduce the complexity of the contact problem. The analytical model presented shows, how to calculate the wear process on the (slow) numerical time scale without resolving all real-time cycles occurring on the (fast) physical time scale.

For sealing elements the load on the outer diameter varies between two pressure levels (the suction $\tilde{p}_{\mathrm{s}}$ and the discharge pressure $\tilde{p}_{\mathrm{d}}$, dependent on the compressors working point), compression ratio, and delivery quantity. Additionally the gas pressure level might change over time. It is assumed that the difference between suction pressure and discharge pressure is constant even if the pressure level changes, Figure 3.3.

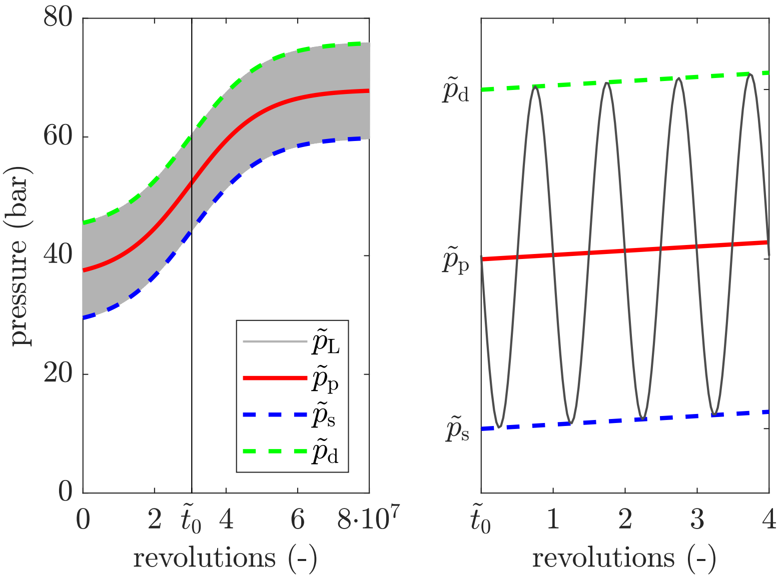


Figure 3.3. Schematic pressure function over time, for some cycles a zoom is shown on the right.

The load-pressure function $\tilde{p}_\mathrm{L}(\tilde{t})$ generally can be developed into $\tilde{p}_\mathrm{cycle}(\tilde{t})$ from the compression cycle and $\tilde{p}_\mathrm{p}(\tilde{t})$ depending on the process and application, e.g. filling a tank with gas. The latter can be expanded into a Taylor series, with the initial pressure $\tilde{p}_\mathrm{p}(\tilde{t}{=}\tilde{t}_0)$ and the pressure change over time

$$\begin{aligned}\tilde{p}_\mathrm{L}(\tilde{t}) &= \tilde{p}_\mathrm{cycle}(\tilde{t}) + \tilde{p}_\mathrm{p}(\tilde{t}) \\ &= \tilde{p}_\mathrm{cycle}(\tilde{t}) + \tilde{p}_\mathrm{p}(\tilde{t}_0) + \frac{\partial \tilde{p}_\mathrm{p}(\tilde{t})}{\partial \tilde{t}}\Big|_{\tilde{t}=\tilde{t}_0}(\tilde{t}-\tilde{t}_0) + \text{h.o.t.}\end{aligned} \tag{3.8}$$

where $\tilde{t}_0$ is usually set at $\tilde{t} = 0$.

Figure 3.4a shows schematically the gas-pressure for the compression chamber, which is equal to the gas pressure in the first cup. Additionally a possible gas pressure for the second cup is drawn, which depends on the sealing efficiency of the previous sealing element and is generally unknown [17,22]. The contact pressure $\tilde{p}_\mathrm{c}(\tilde{t})$ in the dynamic sealing surface depends on the gas pressure on the outer diameter and on the ring geometry, which changes due to wear. Evidently the sliding velocity is also a periodic function on the same time scale, Figure 3.4b.

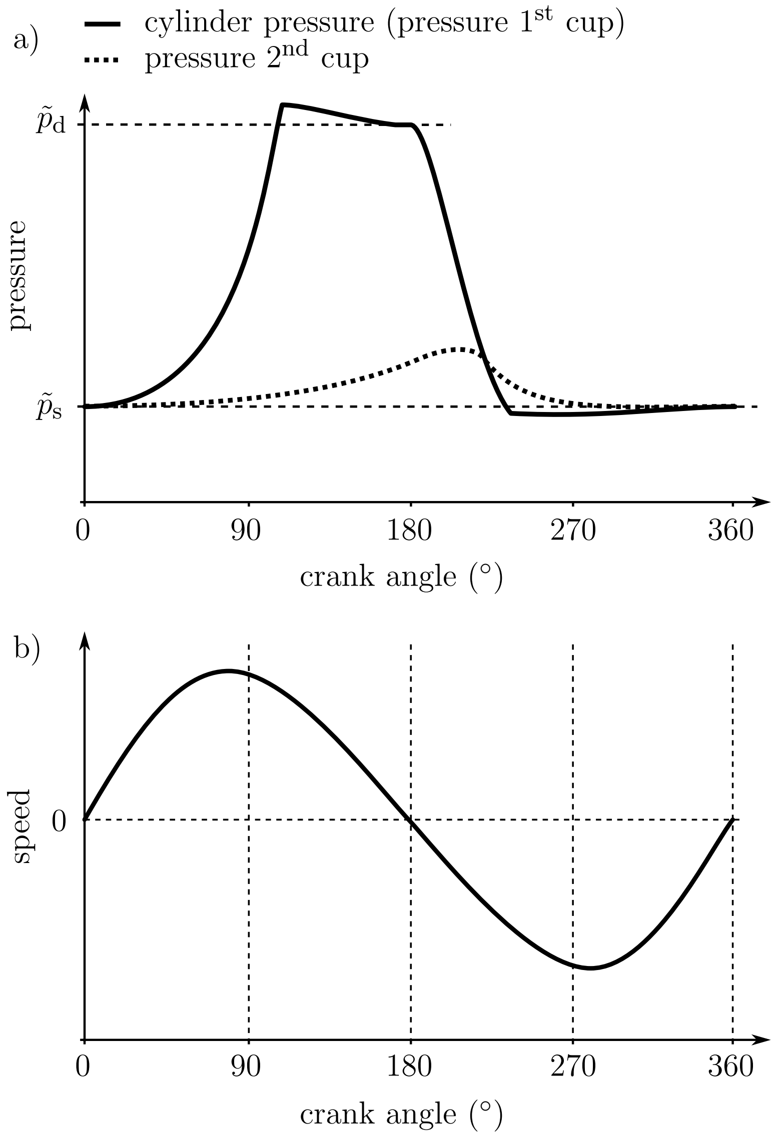


Figure 3.4. a) Pressure over crank angle for the first and second sealing element. b) Rod speed over crank angle. For both diagrams 0° is the head end centre point.

At the piston's turning points the contact conditions change from sliding to sticking and back to sliding into the other direction. This transition is not accounted for. Many different wear equations have been developed in the recent decades, most of them only for very special cases [50]. The oldest, best-known and still most frequently used wear equation is Archard's equation (3.9) [1]. (This equation is redefined here to be compliant to the naming convention.) It states that the wear rate $\frac{\mathrm{d}\tilde{w}(\tilde{t})}{\mathrm{d}\tilde{t}}$ is proportional to the contact pressure $\tilde{p}_\mathrm{c}(\tilde{t})$ and the sliding velocity $\tilde{v}(\tilde{t})$. The proportionality constant is the wear coefficient $\tilde{k}$.

$$\frac{\mathrm{d}\tilde{w}(\tilde{t})}{\mathrm{d}\tilde{t}} = \tilde{k}\tilde{p}_\mathrm{c}(\tilde{t})|\tilde{v}(\tilde{t})| \tag{3.9}$$

Archard's equation performs well under constant conditions. Thus the wear coefficient is often determined in a pin-on-disk experiment, with constant contact pressure and sliding velocity or as a parameter of these values e.g. [6, 7]. To account for reciprocating motion the wear coefficient can now be determined either as a parameter of velocity and pressure or for a given motion and different contact pressures.

Some authors have worked with similar approaches, like for quasi-steady-state sliding contacts by Lengiewicz et al. [40]. Lengiewicz et al. showed a concept for pin-on-flat configurations, namely for the ball-on-flat configuration, how to cope with the two inherited time scales of the wear-controlled problem under constant load, with both contact partners changing their shape due to wear. Peigney [55] analytically investigated a wear on half-plane problem with a rigid indenter on a solid continuum. He showed the asymptotically reached wear profile, if it exists for the investigated load. With his presented minimization approach wear problems with cyclic loadings are considered. Another procedure how to analytically tackle steady-state wear problems for elastic bodies in relative motion was presented by Páczelt and Mróz [52]. Páczelt and Mróz used a wear law where the influence of contact pressure and relative sliding velocity is non-linear. After calculating the contact pressure distribution and the wear velocity vector with the variation principle the shape of the contact surface is determined iteratively by the iterative optimization procedure. As extension of their method, Páczelt and Mróz mentioned that it can be expanded to i) oscillating sliding motion ii) a change in the contact area due to wear. These two poinst were addressed in a later work [53], where Páczelt and Mróz presented the implementation of reciprocating sliding motion (pin-on-flat configuration) using a variational principle and the p-version of finite elements.

Here a solution for wear problems using Archard's wear equation for periodic pressure and periodic sliding is presented. This enables to tackle long-term wear problems, like the wear of dynamic sealing elements, numerically efficiently.

3.3.2 Model

The wear process of sealing rings of a compressor packing is considered. The sealing elements are exposed to a reciprocating motion of the piston rod at the contact surface between the rings and the piston rod with a periodicity given by the speed of the compressor $\tilde{\omega}$. Moreover, the rings are pressed onto the piston rod through a gas pressure load $\tilde{p}_{\mathrm{L}}$ acting on the ring surfaces and eventually yields a time-varying contact pressure $\tilde{p}_{\mathrm{c}}(\tilde{t})$.

The specific wear coefficient $\tilde{k}$, reflecting the abrasion of sealing ring material from a macroscopic point of view, i.e. a height reduction per unit of time, is expected to be small compared to the radial ring height $\tilde{h}$ suggesting the introduction of a new time scale associated with the wear process.

The wear process and the pertaining suitable time scales for its description shall be described by means of matched asymptotic expansions. To this end non-dimensional quantities and formal asymptotic expansions are provided first. Subsequently the governing equations are discussed and an asymptotic matching of the time scales is performed.

3.3.2.1 Wear equation

The non-dimensional quantities, tilde denotes dimensional quantities, are defined as follows,

$$t = \tilde{t}\tilde{\omega}/2\pi \tag{3.10}$$

$$p_{\mathrm{c}} = \tilde{p}_{\mathrm{c}}(\tilde{t})/\tilde{p}_{\mathrm{s}}(\tilde{t}_0) \tag{3.11}$$

$$w = \tilde{w}(\tilde{t})/\tilde{h}(\tilde{t}_0) \tag{3.12}$$

$$v = \tilde{v}(\tilde{t})/\tilde{v}_{\mathrm{M}}, \text{with } \tilde{v}_{\mathrm{M}} = \int_0^{2\pi/\tilde{\omega}} |\tilde{v}(\tilde{t})| \mathrm{d}\tilde{t} \tag{3.13}$$

Here $\tilde{t}$ denotes the time, $\tilde{\omega}$ the angular speed of the compressor, $\tilde{p}_{\mathrm{c}}(\tilde{t})$ the contact pressure between the sealing ring and piston rod surface, $\tilde{p}_{\mathrm{s}}$ the suction pressure, $\tilde{w}(\tilde{t})$ the local abrasion of sealing ring material due to wear, $\tilde{h}$ the radial ring height, $\tilde{v}(\tilde{t})$ the piston rod velocity and $\tilde{v}_{\mathrm{M}}$ the mean piston rod velocity.

Archard's equation (3.9) can be written in the following form

$$\frac{\mathrm{d}w(t)}{\mathrm{d}t} = \frac{2\pi\tilde{k}\tilde{p}_{\mathrm{s}}\tilde{v}_{\mathrm{M}}}{\tilde{h}\tilde{\omega}} p_{\mathrm{c}}(t)|v(t)| = \varepsilon_t p_{\mathrm{c}}(t)|v(t)| \tag{3.14}$$

with the non-dimensional parameter $\varepsilon_t = (2\pi\tilde{k}\tilde{p}_{\mathrm{s}}\tilde{v}_{\mathrm{M}})/(\tilde{h}\tilde{\omega})$ accounting for all wear associated material properties from a macroscopic point of view. In the following, it will be assumed that $\varepsilon_t \ll 1$ which can be justified for typical values found for compressor applications as shown in Table 3.1. Here, values for ε_t are in the range of 10^{-14} to 10^{-4}.

Table 3.1. Estimation of ε_t

name	value range	dimension
suction pressure	$0.1 \leq \tilde{p}_{\mathrm{s}} \leq 10$	MPa
mean rod velocity	$0.1 \leq \tilde{v}_{\mathrm{M}} \leq 10$	$\mathrm{m\,s^{-1}}$
wear coefficient	$10^{-9} \leq \tilde{k} \leq 10^{-5}$	$\mathrm{mm^3\,N^{-1}\,m^{-1}}$
angular velocity	$5 \leq \tilde{\omega} \leq 30$	$\mathrm{s^{-1}}$
(radial) ring height	$5 \leq \tilde{h} \leq 50$	mm

3.3.2.2 Pressure load

The dimensionless pressure load $p_{\mathrm{L}}(t)$, which can be considered as a given function, is defined as follows:

$$p_{\mathrm{cycle}}(t) = \tilde{p}_{\mathrm{cycle}}(\tilde{t})/\tilde{p}_{\mathrm{s}}(\tilde{t}_0) \tag{3.15}$$

$$p_{\mathrm{p}}(t_0) = \tilde{p}_{\mathrm{p}}(\tilde{t}_0)/\tilde{p}_{\mathrm{s}}(\tilde{t}_0) \tag{3.16}$$

$$\frac{\partial p_{\mathrm{p}}(t)}{\partial t} = \frac{\partial \tilde{p}_{\mathrm{p}}(\tilde{t})}{\partial \tilde{t}} / \max\left(\frac{\partial \tilde{p}_{\mathrm{p}}(\tilde{t})}{\partial \tilde{t}}\right) \tag{3.17}$$

$$\begin{aligned} p_{\mathrm{L}}(t) &= p_{\mathrm{cycle}}(t) + p_{\mathrm{p}}(t_0) + \max\left(\frac{\partial \tilde{p}_{\mathrm{p}}(\tilde{t})}{\partial \tilde{t}}\right)\frac{2\pi}{\tilde{\omega}\tilde{p}_{\mathrm{s}}(\tilde{t}_0)}\frac{\partial p_{\mathrm{p}}(t)}{\partial t}t \\ &= p_{\mathrm{cycle}}(t) + p_{\mathrm{p}}(t_0) + \varepsilon_{\mathrm{L}}\frac{\partial p_{\mathrm{p}}(t)}{\partial t}t \end{aligned} \tag{3.18}$$

The pressure from the process is a function with two degrees of freedom: i) the initial pressure level $\tilde{p}_{\mathrm{p}}(\tilde{t}_0)$ and ii) the pressure gradient $\partial \tilde{p}_{\mathrm{p}}(\tilde{t})/\partial \tilde{t}$. The initial pressure level is related to the suction pressure level at $\tilde{p}_{\mathrm{s}}(\tilde{t}_0)$. To keep the information of the magnitude of the dimensional pressure gradient, the pressure gradient is made dimensionless with a characteristic gradient of the function, the maximum gradient: $\max(\partial \tilde{p}_{\mathrm{p}}(\tilde{t})/\partial \tilde{t})$. Thus the magnitude of the dimensionless gradient $\partial p_{\mathrm{p}}(t)/\partial t$ is of order 1 and the non-dimensional parameter $\varepsilon_{\mathrm{L}} = \max(\frac{\partial \tilde{p}_{\mathrm{p}}(\tilde{t})}{\partial \tilde{t}})\frac{2\pi}{\tilde{\omega}\tilde{p}_{\mathrm{s}}}$ accounts for the change of the process pressure load. From Table 3.2 values of ε_{L} can be determined, ranging between 10^{-9} and 10^{-2}.

Table 3.2. Estimation of ε_{L}

name	value range	dimension
suction pressure	$0.1 \leq \tilde{p}_{\mathrm{s}} \leq 10$	MPa
pressure change rate	$10^{-7} \leq \frac{\partial \tilde{p}_{\mathrm{p}}(\tilde{t})}{\partial \tilde{t}} \leq 10^{-3}$	$\mathrm{MPa\,s^{-1}}$
angular velocity	$5 \leq \tilde{\omega} \leq 30$	$\mathrm{s^{-1}}$

While the motion of the piston rod is exactly given by the crank mechanism and hence the piston rod velocity $v(t)$ is known a priori, the contact pressure $p_{\mathrm{c}}(t)$ depends on the periodically varying gas pressure load as well as on geometrical changes due to wear and therefore is not known in advance.

Thus, for subsequent analysis the contact pressure $p_{\mathrm{c}}(t)$ is decomposed into a periodic part $p_{\mathrm{per}}(t)$, associated with the gas pressure, and a non-periodic part $p_{\mathrm{non\,per}}(t)$, associated with the change of the contact pressure with time due to wear, as follows

$$p_{\mathrm{c}}(t) = p_{\mathrm{per}}(t) + p_{\mathrm{non\,per}}(t) \tag{3.19}$$

With the inspection analysis performed in the previous section a formal asymptotic expansion for the contact pressure and the wear can be formulated. For these two quantities the inspection analysis suggests the following scaling

$$p_{\mathrm{c}} = p_0(t) + \varepsilon_t p_1(t) + \mathcal{O}({\varepsilon_t}^2) \tag{3.20}$$

$$w = w_0(t) + \varepsilon_t w_1(t) + {\varepsilon_t}^2 w_2(t) + \mathcal{O}({\varepsilon_t}^3) \tag{3.21}$$

Inserting the asymptotic expansions (3.20) and (3.21) into Archard's equation (3.14) gives

$$\frac{\mathrm{d}}{\mathrm{d}t}(w_0(t) + \varepsilon_t w_1(t) + {\varepsilon_t}^2 w_2(t) + \ldots) = \varepsilon_t (p_0(t) + \varepsilon_t p_1(t) + \ldots)|v(t)| \tag{3.22}$$

which finally yields in leading order

$$\mathcal{O}(1) : \frac{\mathrm{d}}{\mathrm{d}t}(w_0(t)) = 0. \tag{3.23}$$

Evaluating also the second and third order approximations yields

$$\mathcal{O}(\varepsilon_t) : \frac{\mathrm{d}}{\mathrm{d}t}(w_1(t)) = p_0(t)|v(t)| \tag{3.24}$$

$$\mathcal{O}({\varepsilon_t}^2) : \frac{\mathrm{d}}{\mathrm{d}t}(w_2(t)) = p_1(t)|v(t)| \tag{3.25}$$

For later use expanding the non-periodic part of the decomposition $p_{\mathrm{non\,per}}(t)$ being a function of the contact pressure change due to i) wear of the contact area A_{c} (determined by the stiffness and the wear of the ring geometry) and ii) an external pressure load change over time $p_{\mathrm{L}}(t)$ gives

$$p_{\mathrm{non\,per}}(t) = 0 + \frac{\partial p_{\mathrm{c}}}{\partial A_{\mathrm{c}}} \frac{\partial A_{\mathrm{c}}}{\partial w}\Big|_{t=0} \int_0^t \frac{\partial w}{\partial \tau} \mathrm{d}\tau \, t + \frac{\partial p_{\mathrm{c}}}{\partial p_{\mathrm{L}}}\Big|_{t=0} \int_0^t \frac{\partial p_{\mathrm{L}}}{\partial \tau} \mathrm{d}\tau + \text{h.o.t.}\,, \tag{3.26}$$

where higher order terms can be neglected due to $(\int_0^t \frac{\partial w}{\partial \tau} \mathrm{d}\tau)^2 \ll \int_0^t \frac{\partial w}{\partial \tau} \mathrm{d}\tau$. The pressure load function $p_{\mathrm{L}}(t)$ must be a smooth function. Inserting equation (3.24) and the last term of (3.18) (the first two terms are considered in the periodic pressure $p_{\mathrm{per}}(t)$) into (3.26) finally yields

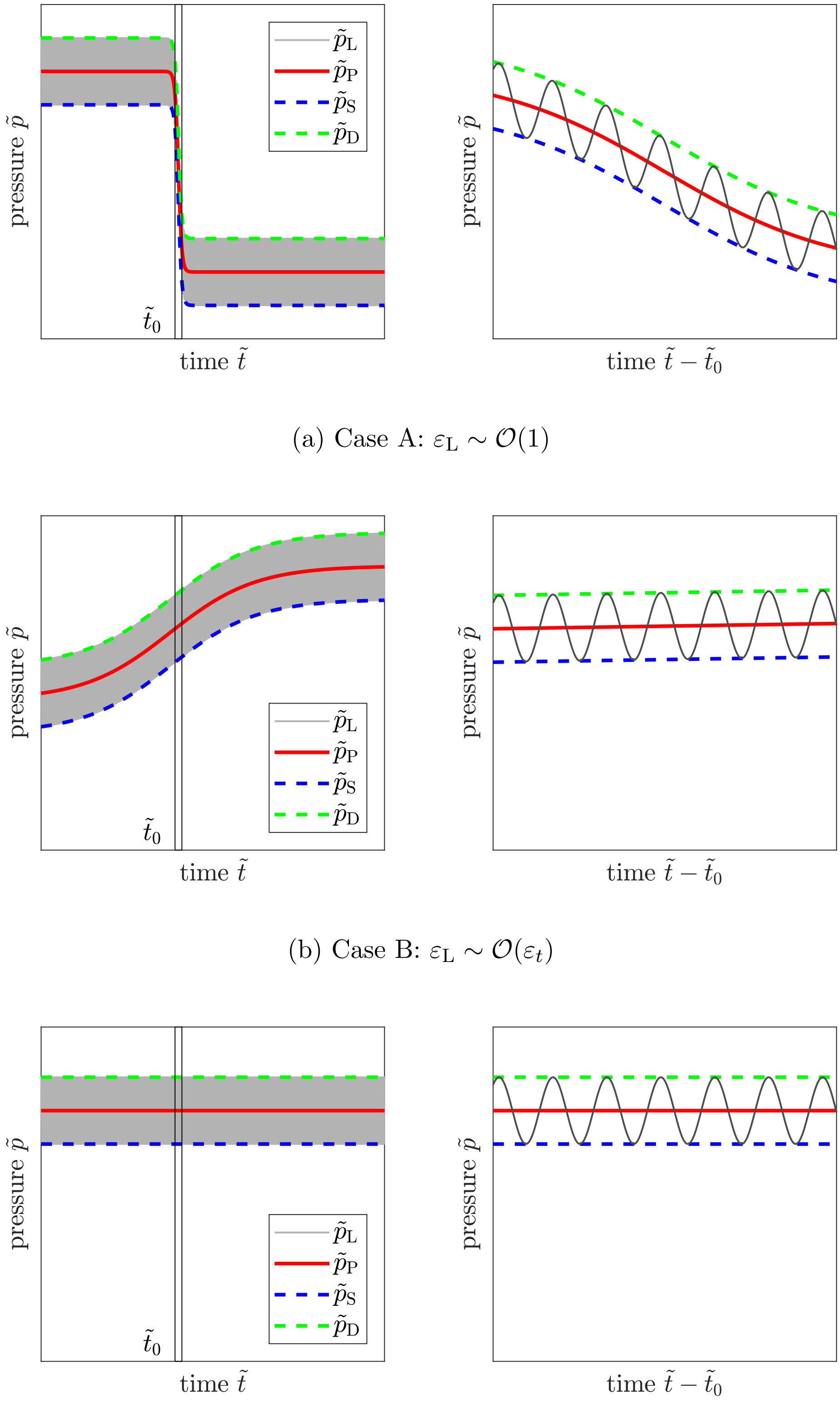
$$p_{\mathrm{non\,per}}(t) = \varepsilon_t \frac{\partial p_{\mathrm{c}}}{\partial A_{\mathrm{c}}} \frac{\partial A_{\mathrm{c}}}{\partial w}\Big|_{t=0} \int_0^t p_0(\tau)|v(\tau)| \mathrm{d}\tau + \varepsilon_{\mathrm{L}} \frac{\partial p_{\mathrm{c}}}{\partial p_{\mathrm{L}}}\Big|_{t=0} \int_0^t \frac{\partial p_{\mathrm{P}}}{\partial \tau} \mathrm{d}\tau. \tag{3.27}$$

Now there are three different cases to be considered case A) $\varepsilon_{\mathrm{L}} \sim \mathcal{O}(1)$, case B) $\varepsilon_{\mathrm{L}} \sim \mathcal{O}(\varepsilon_t)$ and case C) $\varepsilon_{\mathrm{L}} \ll \varepsilon_t$:

- Case A) $\varepsilon_{\mathrm{L}} \sim \mathcal{O}(1)$ indicates a significant pressure change within few compression cycles, e.g. an emergency shut-down of a compressor. Here the solution can be obtained piece-wise before and after the pressure change, Figure 3.5a.

- Case B) $\varepsilon_{\mathrm{L}} \sim \mathcal{O}(\varepsilon_t)$ could be the filling of a cavern with natural gas, a pipeline or a (big) gas storage, Figure 3.5b.

- Case C) $\varepsilon_{\mathrm{L}} \ll \varepsilon_t$ means the pressure on the ring is constant, Figure 3.5c.



(a) Case A: $\varepsilon_\mathrm{L} \sim \mathcal{O}(1)$

(b) Case B: $\varepsilon_\mathrm{L} \sim \mathcal{O}(\varepsilon_t)$

(c) Case C: $\varepsilon_\mathrm{L} \ll \varepsilon_t$

Figure 3.5. Schematic sketch of the different load changing situations.

The cases A) and C) are generic limiting cases of B), thus only case B) will be considered further: Inserting equation (3.27) into the decomposition (3.19) and comparison of the order of magnitudes of the expansion of the contact pressure (3.20) shows

$$\begin{aligned} p_0 &= p_{\mathrm{per}}(t), \\ p_1 &= \frac{\partial p_{\mathrm{c}}}{\partial A_{\mathrm{c}}}\frac{\partial A_{\mathrm{c}}}{\partial w}\Big|_{t=0}\int_0^t p_0(\tau)|v(\tau)|\mathrm{d}\tau + \frac{\partial p_{\mathrm{c}}}{\partial p_{\mathrm{L}}}\Big|_{t=0}\int_0^t \frac{\partial p_{\mathrm{P}}}{\partial \tau}\mathrm{d}\tau. \end{aligned} \tag{3.28}$$

Hence, equation (3.25) yields

$$\begin{aligned} \frac{\mathrm{d}w_2}{\mathrm{d}t} &= \frac{\partial p_{\mathrm{c}}}{\partial A_{\mathrm{c}}}\frac{\partial A_{\mathrm{c}}}{\partial w}\Big|_{t=0}\int_0^t \frac{\partial w}{\partial \tau}\mathrm{d}\tau \\ &= \frac{\partial p_{\mathrm{c}}}{\partial A_{\mathrm{c}}}\frac{\partial A_{\mathrm{c}}}{\partial w}\Big|_{t=0}\int_0^t p_0(\tau)|v(\tau)|\mathrm{d}\tau\, t|v(t)| \end{aligned} \tag{3.29}$$

3.3.2.3 Time scales

Wear analysis solely based on the leading order approximation based on equation (3.23) obviously does not predict any wear. Visible effects only appear if higher order terms are taken into account at longer times giving rise to introducing a second time scale, further referred to as *slow time scale* T, where the wear effects become significant. Here, the wear, to be precise, wear of the order of magnitude of the ring geometry, is zero on the fast time scale t

$$T = \varepsilon_t t. \tag{3.30}$$

Consequently, the asymptotic expansions for the contact pressure and wear on the time scale T can be written as

$$P(T) = P_0(T) + \varepsilon_t P_1(T) + \mathcal{O}({\varepsilon_t}^2) \tag{3.31}$$

$$W(T) = W_0(T) + \varepsilon_t W_1(T) + \mathcal{O}({\varepsilon_t}^2) \tag{3.32}$$

Here, the asymptotic expansion is denoted via capital letters to distinguish the expansion of the slow time scale quantities from the fast time scale quantities. Inserting the expansions (3.31) and (3.32) into Archard's equation (3.14), which has to hold true even on the slow time scale gives

$$\mathcal{O}(1) : \frac{\mathrm{d}}{\mathrm{d}T}(W_0(T)) = P_0(T)V(T) \tag{3.33}$$

$$\mathcal{O}(\varepsilon_t) : \frac{\mathrm{d}}{\mathrm{d}T}(W_1(T)) = P_1(T)V(T) \tag{3.34}$$

With the equations (3.23) to (3.25) and equation (3.33) and (3.34) the problem is properly posed. Here, the quantities on the t time scale contain the boundary conditions, namely the contact pressure and the rod velocity, while the wear is governed by the leading order equation (3.33) on the T time scale. Hence, to close the problem formulation the two time scales need to be matched properly.

To this end, the following matching condition is introduced, i.e. wear on the real (fast) time scale must accumulate to the same value as in the virtual (slow) time scale,

$$\lim_{\varepsilon_t \to 0} w(t = \mathcal{O}(\frac{1}{\varepsilon_t})) = W(T = \mathcal{O}(1)), \tag{3.35}$$

which matches the wear $w(t)$ on the fast time scale t for a sufficiently large time with the wear $W(T)$ on the slow time scale T where the time is an order one quantity. Inserting the equations (3.24) and (3.33) into the matching condition and integrating over an arbitrary time step ΔT finally yields

$$W_0(T = \Delta T) = \lim_{\varepsilon_t \to 0} \int_0^{\frac{\Delta T}{\varepsilon_t}} \varepsilon_t p_0(t)|v(t)|\mathrm{d}t = \int_0^{\Delta T} P_0(T)V(T)\mathrm{d}T \tag{3.36}$$

Note that only the leading order is non-zero due to ${\varepsilon_t}^2$, ${\varepsilon_t}^3$, ... ~ 0. Expanding the periodic part of the contact pressure and the piston rod velocity into a Fourier series

$$p_0(t) = S_0 + \sum_{n=1}^{\infty} \Big(S_{n,1}\cos(2\pi tn) + S_{n,2}\sin(2\pi tn)\Big) \tag{3.37}$$

$$|v(t)| = V_0 + \sum_{n=1}^{\infty} \Big(V_{n,1}\cos(2\pi tn) + V_{n,2}\sin(2\pi tn)\Big) \tag{3.38}$$

and introducing these series into the matching condition (3.36) gives

$$W_0(\Delta T) = \varepsilon_t \int_0^{\frac{\Delta T}{\varepsilon_t}} p_0(t)|v(t)\mathrm{d}t = (S_0 V_0 + C_0)\Delta T = S_0(V_0 + \frac{C_0}{S_0})\Delta T, \tag{3.39}$$

where only terms of $\mathcal{O}(1)$ are considered. Here S_0 and V_0 denote the mean quantities of the contact pressure and the absolute value of piston velocity, respectively. C_0 denotes a constant arising from the $\int \cos^2(2\pi tn)\mathrm{d}t$ and $\int \sin^2(2\pi tn)\mathrm{d}t$ terms of the Fourier series, whereas the other terms like $\int \cos(2\pi tn)\mathrm{d}t$, $\int \sin(2\pi tn)\mathrm{d}t$ or $\int \cos(2\pi tn)\sin(2\pi tn)\mathrm{d}t$ are negligible.

Comparing equation (3.39) with the leading order approximation (3.33) on the slow time scale shows that the contact pressure P_0 as well as the piston velocity V can be considered as constant on the slow time scale T and has to fulfil the following condition for proper matching of the time scales

$$P_0 = S_0 \tag{3.40}$$

$$V = V_0 + \frac{C_0}{S_0} \tag{3.41}$$

Thus, the leading order approximation of Archard's equation for reciprocating contact surface motion and periodically varying contact pressure on a slow time scale, where wear effects become significant, is found to be

$$\frac{\mathrm{d}W}{\mathrm{d}T} = S_0 K(S_0, V_0), \tag{3.42}$$

where $K(S_0, V_0)$ is a virtual wear parameter which can be determined directly by evaluating the terms in equation (3.41) and $S_0 = \int_0^1 p_0(t)\mathrm{d}t$ is the averaged contact pressure seen on the fast time scale. The approximation of Archard's law stated in equation (3.42) reflects the wear behaviour on a slow time scale derived from the governing equations formulated on the fast physical time scale. Thus, equation (3.42) is the proper formulation of a wear law for the implementation in numerical codes studying wear effects where two time scales are present.

3.3.3 Example: non-linear wear law

The asymptotic concept was presented with Archard's (linear) wear equation (3.9) as basis. Since the general non-linear wear equation form (e.g. [62]) can be written as

$$\Delta\tilde{w} = \tilde{k}_2 {\tilde{p}_\mathrm{c}}^a \tilde{v}^b \tilde{t}^c, \tag{3.43}$$

where $\tilde{k}_2$ is the wear coefficient, $\tilde{p}_\mathrm{c}$ the contact pressure, $\tilde{v}$ the relative velocity, $\tilde{t}$ the time of sliding and a, b and c are a set of parameters, which have to be (experimentally) derived for a contact pair.

To show the validity of the presented concept a non-linear wear equation (3.43) will be considered with the parameters $a = 2$, $b = c = 1$:

$$\frac{\mathrm{d}\tilde{w}(\tilde{t})}{\mathrm{d}\tilde{t}} = \tilde{k}_2 {\tilde{p}_\mathrm{c}}^2(\tilde{t})|\tilde{v}(\tilde{t})| \tag{3.44}$$

This choice of parameters makes an analytical solution amenable without enormous efforts.

The non-dimensional quantities are defined as shown above in the equations (3.10) to (3.13). Thus the wear equation can be rewritten into dimensionless quantities as follows

$$\frac{\mathrm{d}w(t)}{\mathrm{d}t} = \frac{2\pi\tilde{k}_2 {\tilde{p}_\mathrm{s}}^2 \tilde{v}_\mathrm{M}}{\tilde{h}\tilde{\omega}} {p_\mathrm{c}}^2(t)|v(t)| = \varepsilon_t {p_\mathrm{c}}^2(t)|v(t)| \tag{3.45}$$

Inserting the asymptotic expansions (3.20) and (3.21) into the non-linear wear equation (3.45) gives

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}(w_0(t)+&\varepsilon_t w_1(t) + {\varepsilon_t}^2 w_2(t) + ...) = \\ &= \varepsilon_t (p_0(t) + \varepsilon_t p_1(t) + ...)^2 |v(t)| \\ &= \varepsilon_t ({p_0}^2(t) + 2\varepsilon_t p_0(t) p_1(t) + {\varepsilon_t}^2 {p_1}^2(t) + ...)|v(t)| \end{aligned} \tag{3.46}$$

which finally yields in leading order

$$\mathcal{O}(1): \ \frac{\mathrm{d}}{\mathrm{d}t}(w_0(t)) = 0. \tag{3.47}$$

Evaluating also the second and third order approximations yields

$$\mathcal{O}(\varepsilon_t) : \frac{\mathrm{d}}{\mathrm{d}w_1}(t) = {p_0}^2(t)|v(t)| \tag{3.48}$$

$$\mathcal{O}({\varepsilon_t}^2) : \frac{\mathrm{d}}{\mathrm{d}w_2}(t) = 2p_0(t)p_1(t)|v(t)| \tag{3.49}$$

Consequently the slow time scale is derived as described in the equations (3.30) to (3.32). Leading to

$$\mathcal{O}(1) : \frac{\mathrm{d}}{\mathrm{d}T}(W_0(T)) = {P_0}^2(T)V(T) \tag{3.50}$$

$$\mathcal{O}(\varepsilon_t) : \frac{\mathrm{d}}{\mathrm{d}T}(W_1(T)) = 2P_0(T)P_1(T)V(T) \tag{3.51}$$

By using the matching condition (3.35) and integrating over an arbitrary time increment ΔT yields

$$W_0(t = \Delta T) = \lim_{\varepsilon_t \to 0} \int_0^{\frac{\Delta T}{\varepsilon_t}} \varepsilon_t {p_0}^2(t)|v(t)|\mathrm{d}t = \int_0^{\Delta T} {P_0}^2(T)V(T)\mathrm{d}T \tag{3.52}$$

Next the velocity term is expanded into a Fourier series as described in equation (3.38). The square of the periodic part of the contact pressure is expanded as follows

$${p_0}^2(t) = S_{0,\mathrm{nl}} + \sum_{n=1}^{\infty}(S_{n,1}\cos(2\pi tn) + S_{n,2}\sin(2\pi tn)) \tag{3.53}$$

Thus the matching condition yields

$$\begin{aligned} W_0(\Delta T) &= \varepsilon_t \int_0^{\frac{\Delta T}{\varepsilon_t}} {p_0}^2(t)|v(t)|\mathrm{d}t \\ &= (S_{0,\mathrm{nl}}V_0 + C_0)\Delta T \\ &= S_{0,\mathrm{nl}}(V_0 + \frac{C_0}{S_{0,\mathrm{nl}}})\Delta T \end{aligned} \tag{3.54}$$

where the result is similar to equation (3.39). Note that ${P_0}^2(T) = S_{0,\mathrm{nl}}$. For the linear and the non-linear wear equation the result looks formally equal, but the derived constants S_0 and $S_{0,\mathrm{nl}}$ are different. For the first case, Archard's wear equation (3.9), $S_0 = \int_0^{\frac{\Delta T}{\varepsilon_t}} p_0(t)\mathrm{d}t$, whereas for the non-linear case $S_0 = \int_0^{\frac{\Delta T}{\varepsilon_t}} {p_0}^2(t)\mathrm{d}t$. Note that in general $S_{0,\mathrm{nl}} \neq {S_0}^2$. Figure (3.14) now shows the difference between the linear and the non-linear wear law. It must be emphasized that in the non-linear wear case it is not valid to simply insert the mean value into Archard's equation.

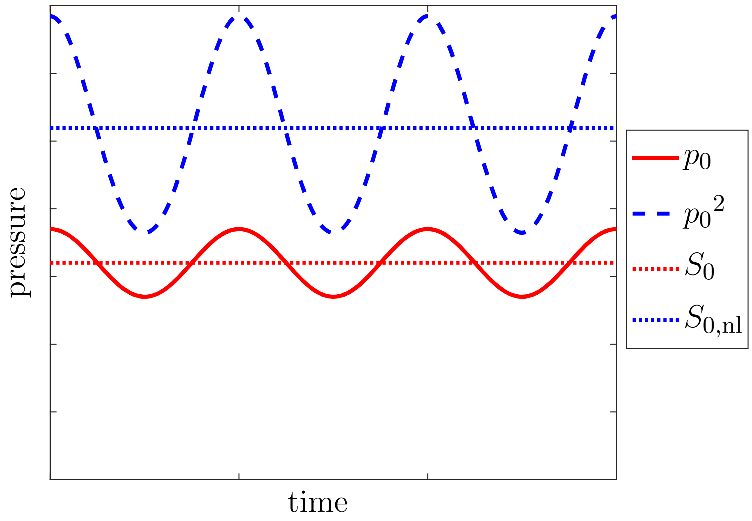


Figure 3.6. Comparison of the cyclic pressure for the linear (red) and non-linear (blue) wear equation including the resulting "mean" values for both models, S_0 and $S_{0,\mathrm{nl}}$ respectively.

3.3.4 Error estimation

The presented concept works well in a closed vicinity of the evolution point $T = 0$. The question arises at which time increment ΔT the approximation errors become no longer negligible.
The truncation error of this approach can be estimated by evaluating the higher order terms. Since the physical behaviour governed by the equations on the fast time scale must be reflected also on the slow time scale the $\mathcal{O}(\varepsilon_t)$-terms stated in equation (3.34) have to exhibit a similar form as the terms stated in equation (3.29)

$$\begin{aligned}\frac{\mathrm{d}W_1}{\mathrm{d}T} &= \frac{\partial p_\mathrm{c}}{\partial A_\mathrm{c}}\frac{\partial A_\mathrm{c}}{\partial W}\int_0^T \frac{\partial W}{\partial T}\mathrm{d}T V_0 + \frac{\partial p_\mathrm{c}}{\partial p_\mathrm{L}}\int_0^T \frac{\partial p_\mathrm{L}}{\partial T}\mathrm{d}T \\ &= \frac{\partial p_\mathrm{c}}{\partial A_\mathrm{c}}\frac{\partial A_\mathrm{c}}{\partial W} S_0 K(S_0, V_0) T + \frac{\partial p_\mathrm{c}}{\partial p_\mathrm{L}}\int_0^T \frac{\partial p_\mathrm{L}}{\partial T}\mathrm{d}T.\end{aligned} \tag{3.55}$$

Note: $\int_0^t \frac{\partial p_\mathrm{L}}{\partial t}\mathrm{d}t = \int_0^{\frac{T}{\varepsilon_t}} \frac{\partial p_\mathrm{L}}{\partial t}\mathrm{d}t = \int_0^T \frac{\partial p_\mathrm{L}}{\partial T}\frac{\partial T}{\partial t}\frac{1}{\varepsilon_t}\mathrm{d}T = \int_0^T \frac{\partial p_\mathrm{L}}{\partial T}\mathrm{d}T$, where $\frac{\partial T}{\partial t} = \varepsilon_t$ for case B.

Assuming piecewise constant terms $\frac{\partial p_\mathrm{c}}{\partial A_\mathrm{c}}$, $\frac{\partial A_\mathrm{c}}{\partial W}$ and $\frac{\partial p_\mathrm{c}}{\partial p_\mathrm{L}}$ and integrating equation (3.55) over an arbitrary time step ΔT finally yields

$$W_1 = \frac{\partial p_\mathrm{c}}{\partial A_\mathrm{c}}\frac{\partial A_\mathrm{c}}{\partial W} S_0 K(S_0, V_0)\frac{\Delta T^2}{2} + \frac{\partial p_\mathrm{c}}{\partial p_\mathrm{L}}\int_0^{\Delta T}\int_0^T \frac{\partial p_\mathrm{L}}{\partial \tau}\mathrm{d}\tau\,\mathrm{d}T. \tag{3.56}$$

The inner integration over τ is equivalent to the integration in (3.55), but for distinction of integrand and integration limit T was substituted by τ. Thus, the asymptotic series (3.32) can be written as

$$W = S_0 K(S_0 V_0)\Delta T(1 + \varepsilon_t \frac{\partial p_\mathrm{c}}{\partial A_\mathrm{c}} \frac{\partial A_\mathrm{c}}{\partial W} \frac{\Delta T}{2}) + \varepsilon_t \frac{\partial p_\mathrm{c}}{\partial p_\mathrm{L}} \int_0^{\Delta T} \int_0^T \frac{\partial p_\mathrm{L}}{\partial \tau} \mathrm{d}\tau \, \mathrm{d}T. \tag{3.57}$$

Comparing this result to the numerical implementation of Archard's equation as stated in equation (3.42) yields the following condition for the time step size

$$\varepsilon_t \frac{\partial p_\mathrm{c}}{\partial A_\mathrm{c}} \frac{\partial A_\mathrm{c}}{\partial W} \frac{\Delta T}{2} \ll 1 \text{ and } \varepsilon_t \frac{\partial p_\mathrm{c}}{\partial p_\mathrm{L}} \int_0^{\Delta T} \int_0^T \frac{\partial p_\mathrm{L}}{\partial \tau} \mathrm{d}\tau \, \mathrm{d}T \ll 1. \tag{3.58}$$

Here, the terms $\partial p_\mathrm{c}/\partial A_\mathrm{c}$ and $\partial A_\mathrm{c}/\partial W$ are assumed to be of order $\mathcal{O}(1)$. This assumption is violated for configurations where two bodies are in contact in a single point or line, e.g. the wear of a spherical pin, since the term $\partial A_\mathrm{c}/\partial W|_{(T=0)} \gg \mathcal{O}(1/\varepsilon_t)$ becomes very large. For such geometrical configurations the wear time scale overlaps with the physical time scale associated with the surface motion. Thus, the problem has to be treated first on the fast physical time scale until a finite and sufficiently large contact area is present, which allows switching to the slower wear time scale.
Furthermore, $\partial p_\mathrm{L}/\partial t \leq \mathcal{O}(\varepsilon_t)$ is fulfilled for the considered cases B and C.
For the problem formulation of packing rings the term $\frac{\partial p_\mathrm{c}}{\partial A_\mathrm{c}}$ can be further expanded into a term $\frac{\partial p_\mathrm{A}}{\partial A_\mathrm{c}}$ resulting from the change in contact area and a term $\frac{\partial p_\mathrm{f}}{\partial A_\mathrm{c}}$ resulting from the change in gas pressure acting on the ring surfaces.

$$\frac{\partial p_\mathrm{c}}{\partial A_\mathrm{c}} = \frac{\partial p_\mathrm{A}}{\partial A_\mathrm{c}} + \frac{\partial p_\mathrm{f}}{\partial A_\mathrm{c}} \tag{3.59}$$

Since for typical packing ring geometries the term $\frac{\partial p_\mathrm{A}}{\partial A_\mathrm{c}}$ is larger than the term $\frac{\partial p_\mathrm{f}}{\partial A_\mathrm{c}}$, the term $\frac{\partial p_\mathrm{A}}{\partial A_\mathrm{c}}$ determines the size of the time increment for updating the geometry. The integer of the ratio $\frac{\partial p_\mathrm{A}}{\partial A_\mathrm{c}}/\frac{\partial p_\mathrm{f}}{\partial A_\mathrm{c}}$ represents the calculation cycle after which the gas pressure distribution in the dynamic sealing surface has to be updated. Hence, condition (3.58) finally yields the following condition for the maximum time step size

$$\Delta T_\mathrm{max} \leq \frac{err_\mathrm{threshold}}{\varepsilon_t \frac{\partial p_\mathrm{A}}{\partial A_\mathrm{c}} \frac{\partial A_\mathrm{c}}{\partial W} + \varepsilon_t \frac{\partial p_\mathrm{f}}{\partial A_\mathrm{c}} \frac{\partial A_\mathrm{c}}{\partial W}}, \tag{3.60}$$

where the error threshold $err_\mathrm{threshold}$ is chosen sufficiently small.

3.3.5 Proof of concept

With a numerical experiment it can be shown, that the developed concept is valid. A calculation on the fast time scale is compared to a calculation on the slow time scale. Here a simplified sealing element configuration is considered, Figure 5.8, namely an uncut single piece packing ring of rectangular cross section [36]. This is closely elaborated in section 5.4. Note that the conclusions drawn here are valid for sealing elements of higher geometrical complexity as well. In equation (5.7) (from [36]) the change of the inner ring

radius due to wear over time is dependent of i) geometrical features: the outer ring radius, the rod diameter; ii) material parameters: the shear modulus and Poisson's ratio; iii) the pressure acting on the outer radius of the ring; iv) the wear coefficient and v) the relative velocity between the contacting surfaces.
For both time scales an Euler forward scheme is implemented to calculate the radial ring wear over time. On the fast time scale the piston rod movement is implemented as a sine function, which is a good approximation of the crank shaft motion, Figure 3.4b, and is resolved in 10° steps. On the slow time scale the wear-model relevant velocity is implemented as mean value of the absolute value of the rod velocity. The pressure on the outer diameter is held constant in both cases. This case is similar to the load situation of the last (crank end) packing ring of a piston rod sealing system [22].

$$w_{i+1} = w_i + \varepsilon_t v(t) p_{\mathrm{c}} \Delta t \tag{3.61}$$

$$W_{i+1} = W_i + v p_{\mathrm{c}} \Delta T \tag{3.62}$$

The results resolving the problem on the fast time scale as given in equation (3.29) as well as the numerical implementation as given by equation (3.62) are shown in Figure 3.7. The parameters for the validation are listed in Table 3.3. The rod diameter is denoted with $\tilde{D}$.

Table 3.3. Parameters for the validation

parameter	value and dimension	non-dimensional value
$\frac{\tilde{R}^{(\mathrm{o})}}{\tilde{R}^{(\mathrm{i})}}$	$\frac{33.4\,\mathrm{mm}}{25.4\,\mathrm{mm}}$	1.315
$\tilde{h} = \frac{\tilde{D}}{\tilde{R}^{(\mathrm{o})} - \tilde{R}^{(\mathrm{i})}}$	$\frac{50.8\,\mathrm{mm}}{8\,\mathrm{mm}}$	6.35
$\frac{\tilde{p}_{\mathrm{L}}}{\tilde{G}}$	$\frac{10\,\mathrm{MPa}}{500\,\mathrm{MPa}}$	0.02
ν	0.4	0.4
$\tilde{R}^{(\mathrm{i})}{}_0 = \frac{\tilde{D}}{2}$	25.4 mm	-
$\tilde{v}_{\mathrm{M}}$	$5\,\mathrm{m\,s^{-1}}$	-

For a more complex load case on the same ring, with a given, changing pressure load function the dimensionless pressure load and resulting contact pressure are depicted in Figure 3.8 and the resulting dimensionless wear in Figure 3.9.

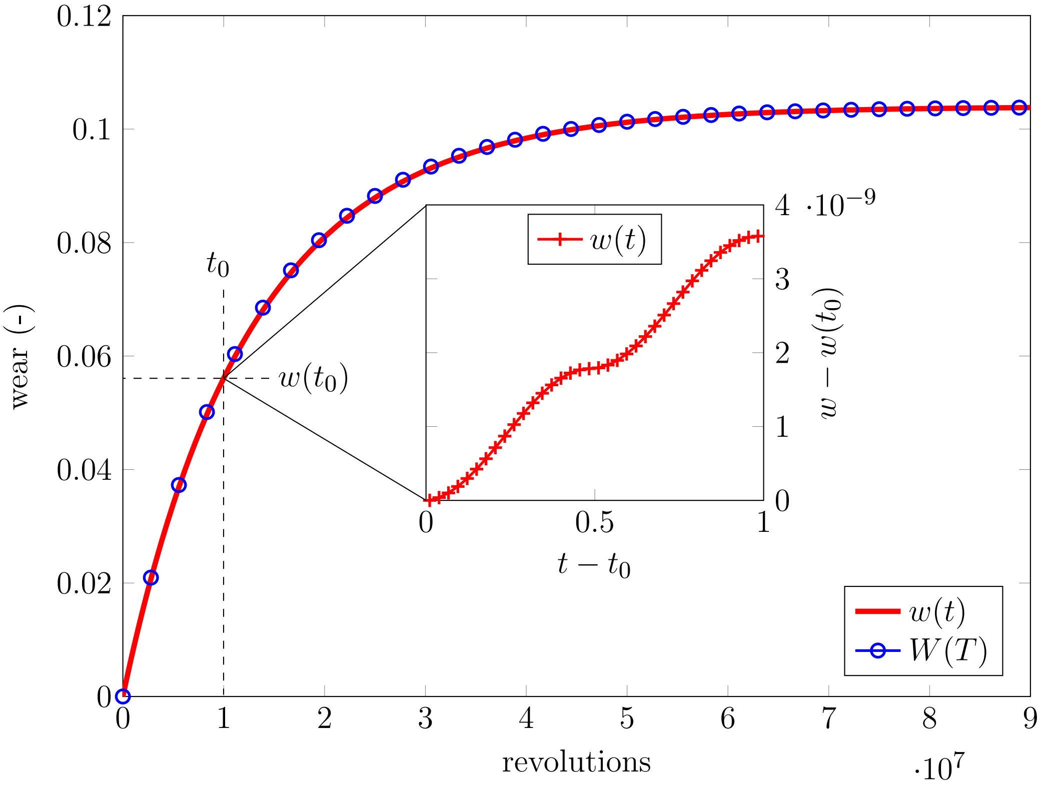


Figure 3.7. Validation of the model: The diagram shows the accumulated wear (per ring height) over the number of revolutions. The calculation of the fast time scale t is drawn as red line, every tenth calculated point on the slow time scale T is marked with a blue circle. The smaller plot shows the wear over the 10^7th revolution, here each of the 36 calculated points is marked with a red plus.

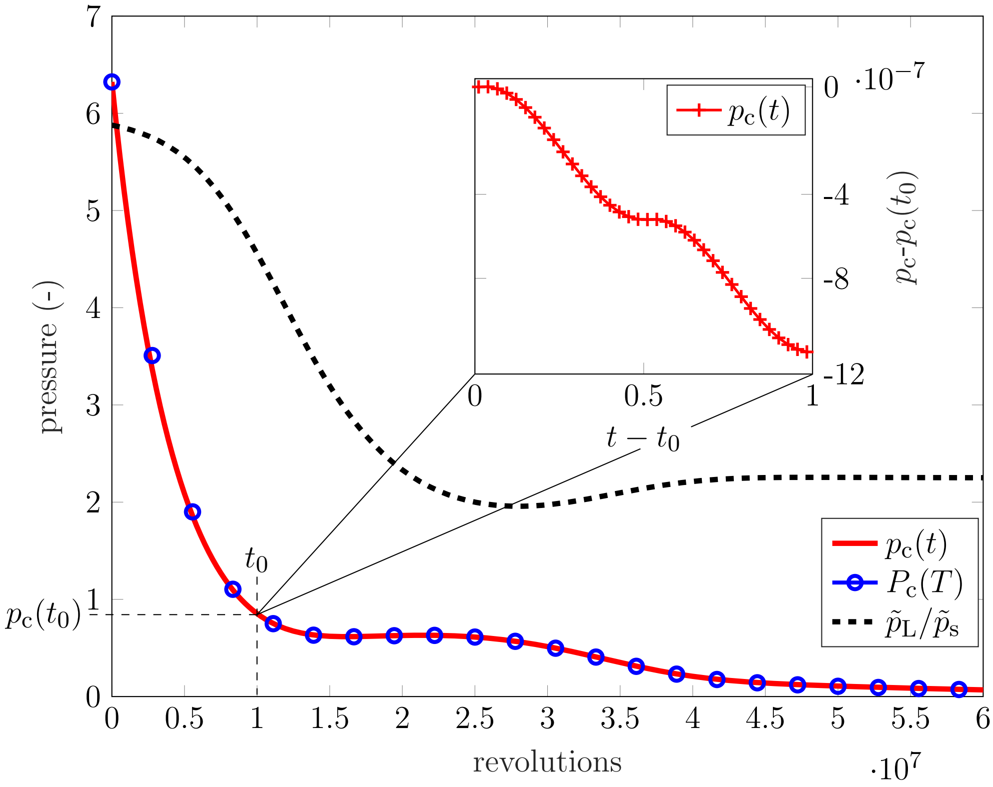


Figure 3.8. The change in the given pressure load (dotted line) and the resulting contact pressure over time, calculated on both time scales. The zoom shows the contact pressure of the 10^7th revolution on the fast time scale. A plus marks every calculated point in time.

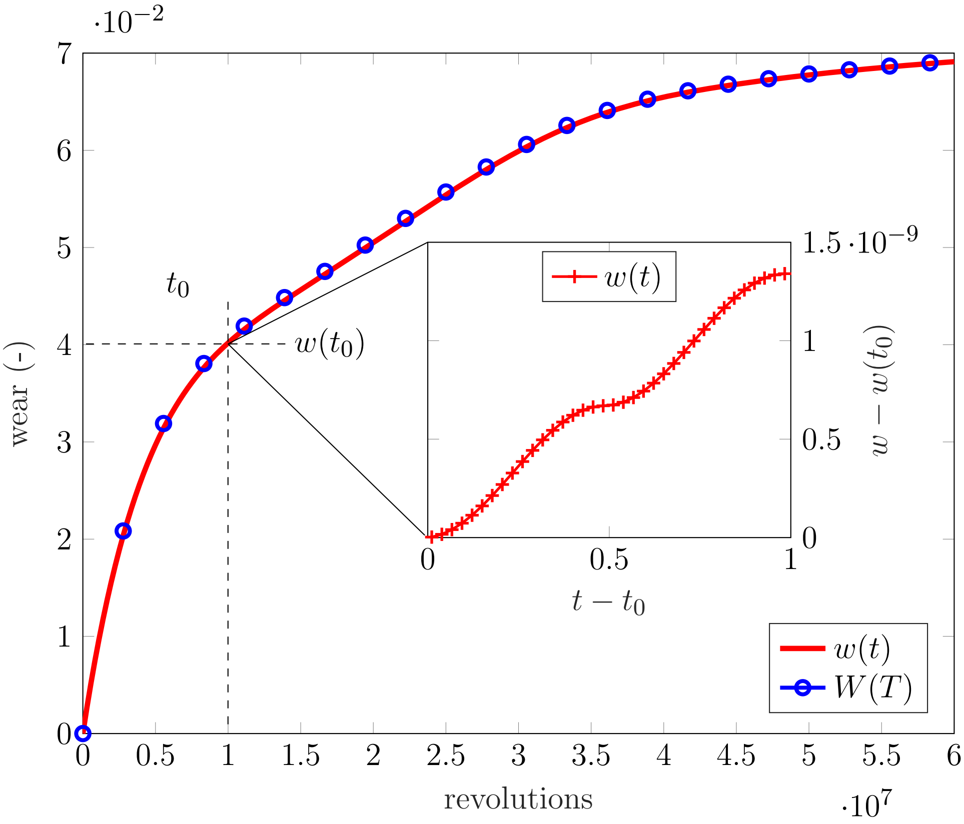


Figure 3.9. Time scale model comparison for the accumulated wear (per ring height) over the number of revolutions for the changing load presented in Figure 3.8. The calculation of the fast time scale t is drawn as red line, every tenth calculated point on the slow time scale T is marked with a blue circle. The smaller plot shows the wear over the 10^7th revolution, here each of the 36 calculated points is marked with a red plus.

Although contact pressure, sliding velocity and wear rates change within a single stroke on a fast time scale, it could be shown that by using an asymptotic approach wear can be calculated quantitatively correct on a slow time scale without considering every single stroke. It is possible to numerically implement wear only on the slow time scale without resolving the reciprocating motion of the contacting surfaces, because these quantities are analytically included into the wear coefficient on the slow time scale.

Through the introduction of the slow time scale the calculation times are reduced significantly even for the simplified one dimensional analytical wear consideration presented above. The wear calculation for example 1, Figure 3.7, on the fast time scale evaluates $3.24 \cdot 10^9$ points whereas on the slow time scale only 330 points are evaluated.

Chapter 4

Material

The goal to model packing ring wear efficiently can only be achieved by making necessary model restrictions and assumptions. This chapter presents the tribological modelling of a mounted packing ring as well as the thoughts and considerations for the mechanical modelling of the used packing ring material including all necessary test methods.

4.1 Tribological modelling

The fact that tribology is a system property is probably the greatest challenge in developing rings. Many interacting parameters influence how a ring (or packing) behaves. Some of the parameters can be controlled more easily (e.g. rod velocity, gas pressure, gas type), some with more effort and even then only to a certain extent (gas dryness, surface characteristics like roughness or profile) and some are nearly uncontrollable e.g. the rod temperature or leakage.

The contact layer where the tribo-chemical and mechanical interactions of asperities take place is inevitably part of the model, but not resolved in detail. It is important to understand that the calculated results in the contact layer, e.g. structural stresses and strains are not reliable. The high deformation gradients and the flash temperatures occurring during the wear process can currently not be evaluated in the same calculation where the ring wears in the magnitude of millimetres. An example for nano-scale modelling is the work of Barry et al. [4]. They studied only few PTFE molecule chains and found that the coefficient of friction increases with decreasing temperature (investigated between 25 and 300 K) for the different investigated molecular chain orientations.
The fact that the contact layer is not modelled in detail does not mean that it is disregarded or not of importance. The micro structure of ring and rod are not modelled, but the gas pressure distribution arising from that micro structure is a crucial ingredient and part of the model. The typical PTFE transfer film or its formation is not modelled explicitly, but the coefficient of friction or the wear coefficient is derived from configurations where a transfer film is present. This is valid since the formation of a quasi steady-state transfer

film is a short process compared to a ring's lifetime.

The frictional force of a packing ring design under given boundary conditions is a benchmark criterion. The lower the frictional force the more likely the ring runs cooler, wears more slowly and has a longer lifetime.
The frictional force can be derived from the contact pressure distribution in the dynamic sealing surface assuming Coulomb friction. To compare simulation and experiment of a loaded packing ring the coefficient of friction has to be determined as well as the gas pressure dependent frictional force. Both is possible with the testing device presented in the following.

4.1.1 Frictional force tester

A purpose-designed test rig allows to measure the frictional force of a packing ring mounted on a rod in a T-cup, which can be pressurized, see Figure 4.1. The static part of the frictional force tester is mounted on a standard testing machine of the type Zwick Roell Z010, company Zwick GmbH & Co. KG, Germany. Bolts prevent rotation of the frictional force tester, which are secured with nuts.

A rod is fixed to a 500 N load cell, precision class 0.05 %, which pulls or pushes the rod through the packing. The radial clearance of 1.6 mm between the ⌀50.8 mm (2") rod and the cup flanges is similar to the clearances in a compressor. The rods running surface is 350 mm long. Figure 4.2 names all parts of the frictional force tester. The tests are performed in a climate controlled laboratory at 20 °C.

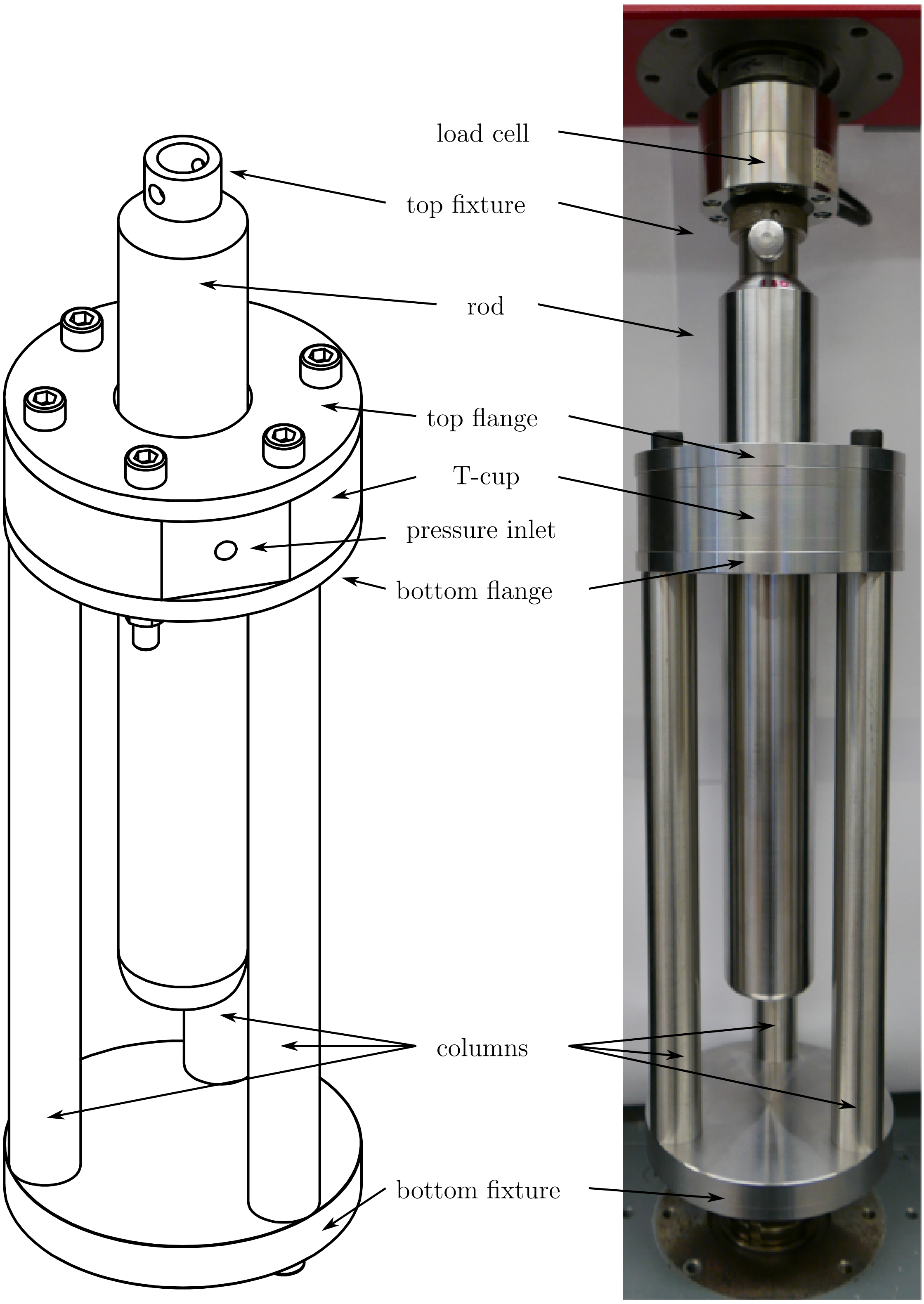


Figure 4.1. Friction force tester. Left: Sketch. Right: Photograph of the mounted friction force tester on the standard testing machine.

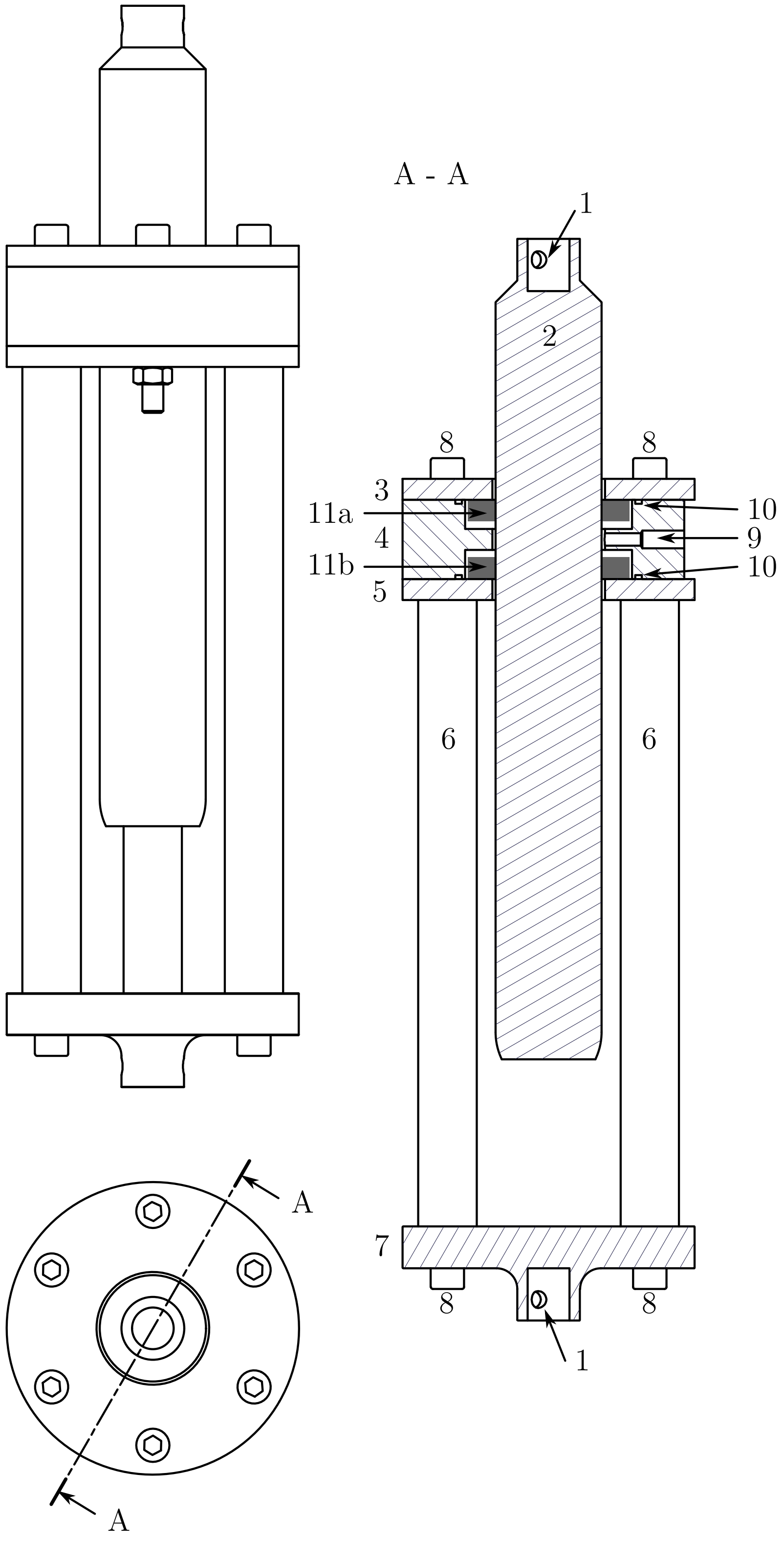


Figure 4.2. Side and top view of the friction testing device, including a cut, where all parts are named. (1) pin hole, fixture to the force measurement device, (2) rod, (3) top flange, (4) T-cup, (5) bottom flange, (6) columns, (7) bottom fixture to the testing machine, (8) screw, (9) pressure inlet, (10) O-ring groove, (11a) upper ring, (11b) lower ring.

4.1.2 Rod surface

The counter-face materials which are used most frequently in the industry are X20Cr13 steel (cylinder liner) and tungsten carbide (WC) coated steel. The rods where turned at the HOERBIGER Wien (HW) tool shop from X20Cr13 steel. The surface finishing of the steel rod was done at the HW. The high velocity oxygen fuel spraying (HVOF) coating and surface finishing of the WC coated rod was performed by TLBS GmbH, Vienna. A WC coating is commonly used for rods in non-lubricated service, consisting of WC/Co/Cr with 86/10/4 [31].The coated rod will further be referred to as 'WC rod'.
The surface of the rods was characterized with a Hommel-Etamic T8000 and a sensing arm of type TS1 KE5/90GD T1.9 D4/30 testing device. The roughness was determined on six (0-5) axial positions and at each on four circumferential positions for the steel rod. The WC rod is characterised on five (1-5) axial and four circumferential positions, see Figure 4.3. Position 0 is not WC coated.

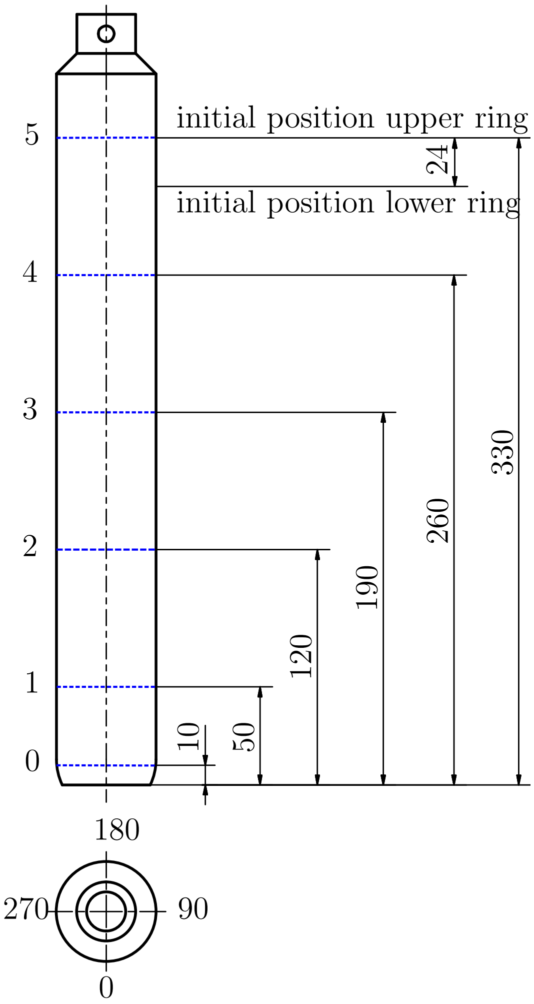


Figure 4.3. The blue dashed lines mark the roughness measurement positions. For the steel rod on all 6 positions a reading is taken. The WC rod is measured on position 1 to 5, since position 0 is not coated.

The roughness was determined according to the DIN EN ISO 4287 standard. On each position over a total measurement length of 4.8 mm five measurements were performed with 0.8 mm each. Usually the surface roughness for rods is specified with $R_\mathrm{a} = 0.3 \pm 0.05$µm [31]. The steel rod does not meet this standard, with a maximum of $R_\mathrm{a} = 0.8$ µm, Figure 4.4, but allows to check the influence of the roughness on the frictional force. The WC rod meets the standard very well, Figure 4.5.

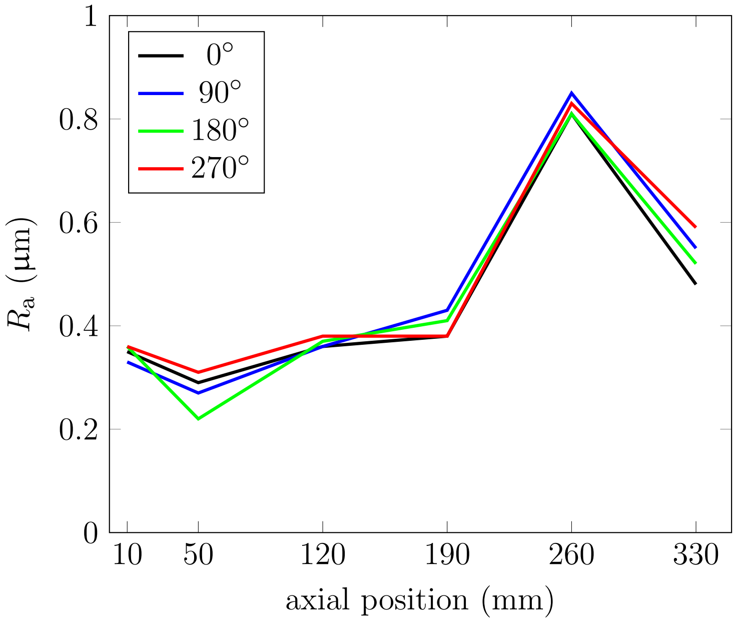


Figure 4.4. R_a measurements on the steel rod at the locations marked in Figure 4.3.

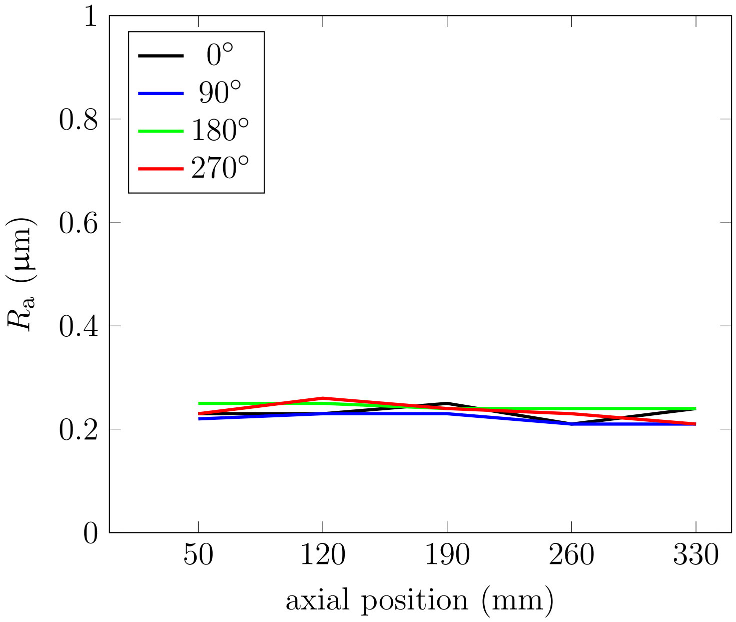


Figure 4.5. R_a measurements on the WC rod at the locations marked in Figure 4.3.

4.1.3 Coefficient of friction

To determine the coefficient of friction a special 20-piece ring design, named "segmented ring", is used, see Figure 4.6. For a sufficiently narrow ring segment the bending stiffness becomes negligible. That allows to measure the frictional force uninfluenced by the material's elastic properties. The segments are equally spaced around the rod, which is ensured by a mounting device. The segmented ring has two circumferential grooves on the outer diameter, in which garter springs can be placed to press the ring against the rod. This ring has no sealing properties, since it is used for the determination of the coefficient of friction only.

The following procedure is used when assembling the segmented ring and mounting it on a rod. The ring pieces are placed on a two-piece mounting device, consisting of a sleeve and a spacer, see Figure 4.7 (garter springs not depicted). Then the garter springs are mounted in the ring's grooves. The mounting device is pulled over the rod and the sleeve is removed, leaving the ring and the spacer on the rod, Figure 4.8. After carefully removing the spacer the ring pieces are mounted equally spaced on the rod, see Figure 4.9.

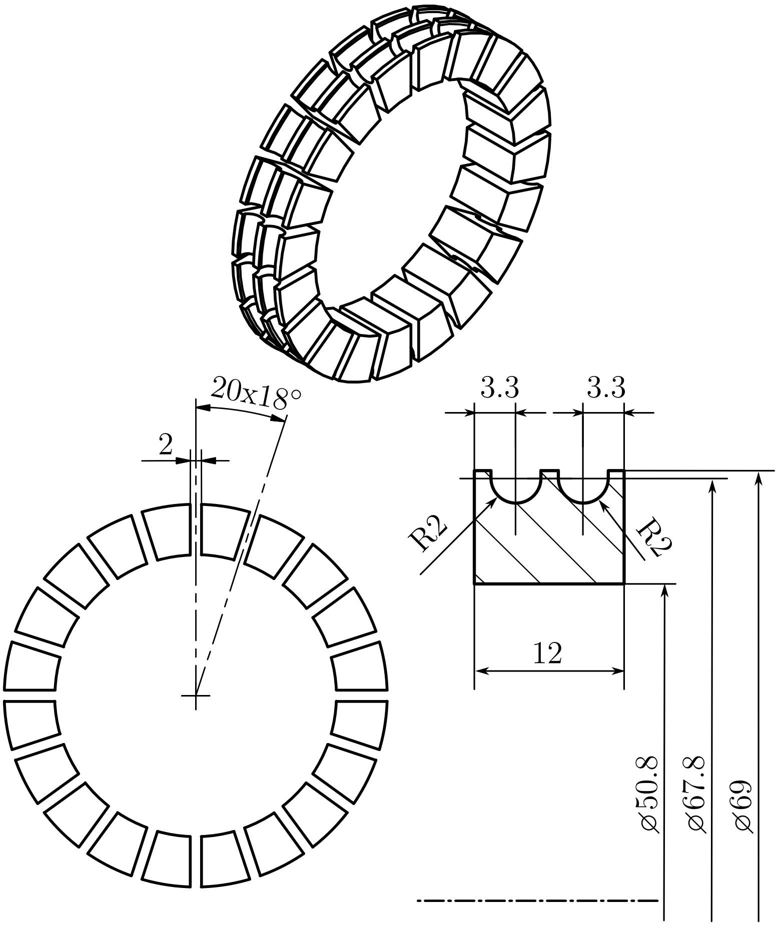


Figure 4.6. The 20 piece ring sketched in 3D and its drawing, including a cross section of a ring segment.

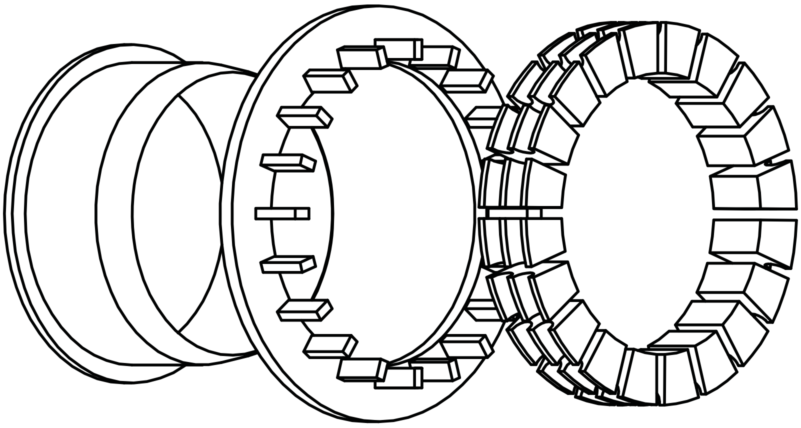


Figure 4.7. The mounting device used for the segmented ring. From left to right: Sleeve, spacer and segmented ring.

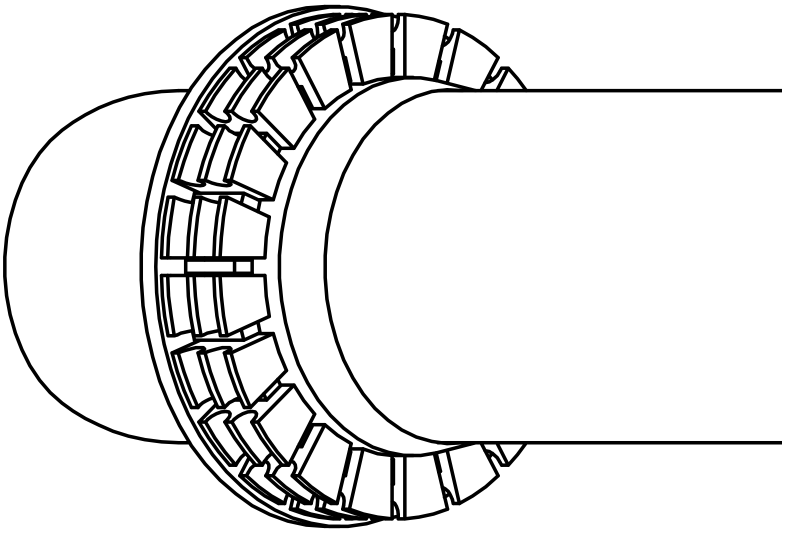


Figure 4.8. The mounting device with the assembled ring on a rod. The garter springs are not depicted.

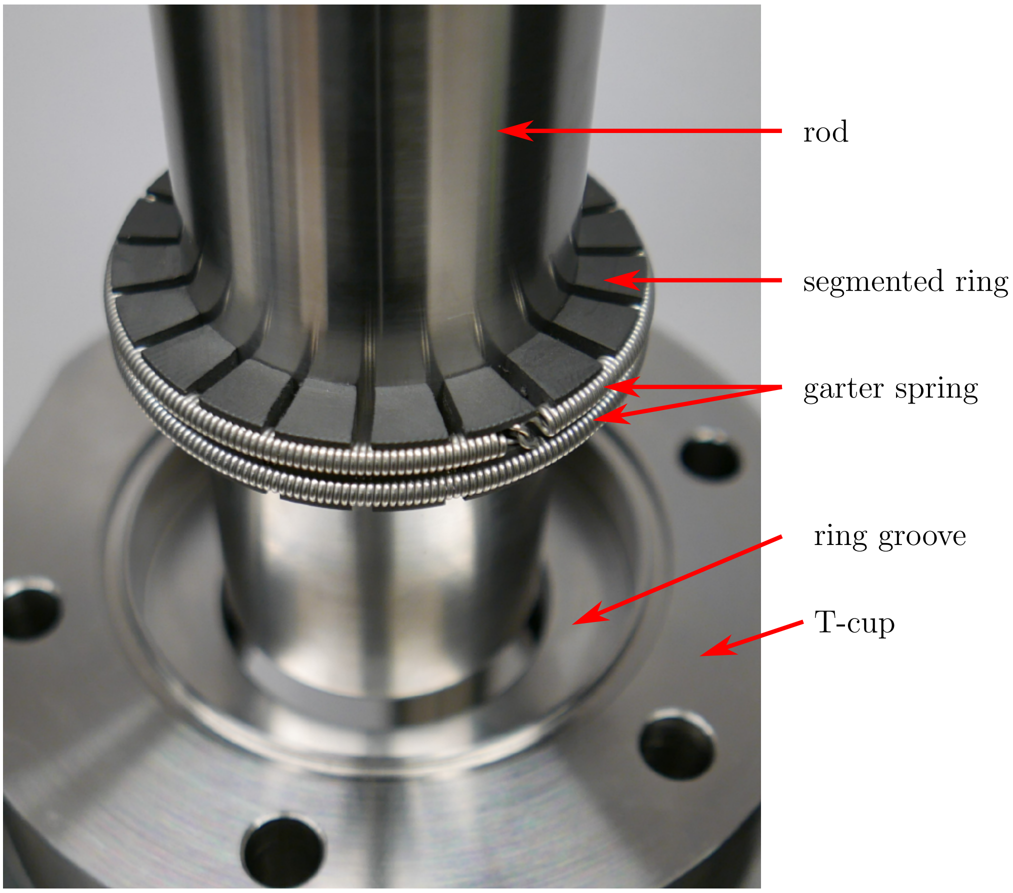


Figure 4.9. The segmented ring is mounted on a rod, held by two garter springs. In the background the T-cup with the upper ring groove is visible.

4.1.3.1 Normal force between rod and segmented ring

The coefficient of friction μ is calculated from the rod's measured pull-off force and the knowledge of the contact pressure applied by the garter springs. The friction is assumed to be of Coulomb type, equation (4.1). The contact force is referred to as F_{C} and the frictional force as F_{F}.

$$F_{\mathrm{F}} = \mu F_{\mathrm{C}} \tag{4.1}$$

The contact force resulting from the garter spring force is determined from the geometry of a single ring segment. The angle between the spring forces F_{S} is α. A force triangle and the law of cosines gives the contact force F_{C}.

$$F_{\mathrm{C}} = \sqrt{{F_{\mathrm{S}}}^2 + {F_{\mathrm{S}}}^2 - 2{F_{\mathrm{S}}}^2\cos(\alpha)} = F_{\mathrm{S}}\sqrt{2(1-\cos(\alpha))} \tag{4.2}$$

To get the total contact force $F_{\mathrm{C\,t}}$ between ring and rod the number of springs n_{S} and the number of ring segments n_{seg} has to be taken into account. We assume that the ring segments are all equal, thus a multiplication accounts for their number. Since garter springs sometimes differ in their offset value at the beginning of the elongation, the forces of the springs are summed up.

$$F_{\mathrm{C\,t}} = n_{\mathrm{seg}} \sqrt{2(1-\cos(\alpha))} \sum_{i=1}^{n_{\mathrm{S}}} F_{\mathrm{S}i} \tag{4.3}$$

Figure 4.10 shows a sketch of a single 18° ring segment from the segmented ring and the force triangle. A 2 mm milling cutter separates the segments. Thus the wrap of the garter spring (and the angle between the spring forces in the force triangle) is 14.62° at a diameter of 67.8 mm. This is the diameter of the centre of the garter springs.

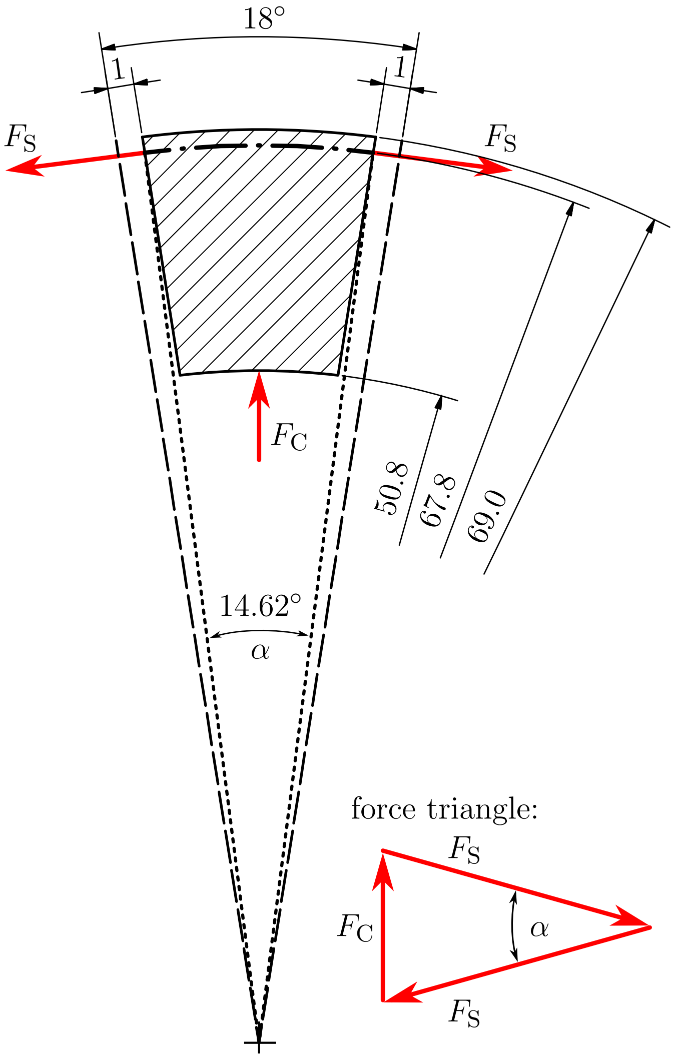


Figure 4.10. Sketch of a single ring segment and all acting forces. The force triangle for the determination of the contact force F_{S} is shown on the bottom.

4.1.3.2 Garter springs

The length of the garter springs is determined with a calliper. Spring 1 has a total length of 195.3 mm and an effective spring length of 188.3 mm. The effective spring length is the total spring length reduced by the length of the hooks on each spring end, see Figure 4.11. This effective spring length is used for the calculation of the spring elongation. Spring 2 has a total length of 195.6 mm with an effective spring length of 188.6 mm. The mounted springs are elongated to a total length of 213 mm, i.e. an elongation of 10 % of the effective spring length.

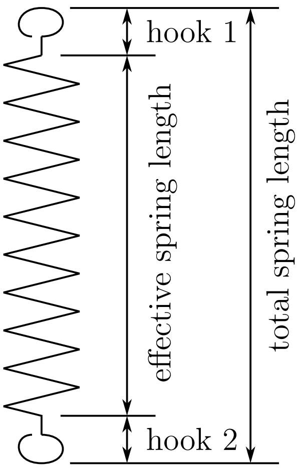


Figure 4.11. Sketch of a garter spring: total and effective spring length.

The spring force at an elongation of 18.7 and 19 mm is of interest for spring 1 and 2, respectively. The garter springs are tested on a standard testing machine. The springs are mounted loosely (no initial tension) on pins at the bottom fixture and at the load cell. The springs are elongated by 22 mm, i.e, 11.6 % of the initial spring length.

The force-displacement curves are evaluated above 0.5 N to remove the offset resulting from the loose mounting. Figure 4.12 shows the force-displacement curve for each garter spring. The marked point represents the mounted situation. The springs have the same slope after an initial offset. The offset itself is different for each spring.

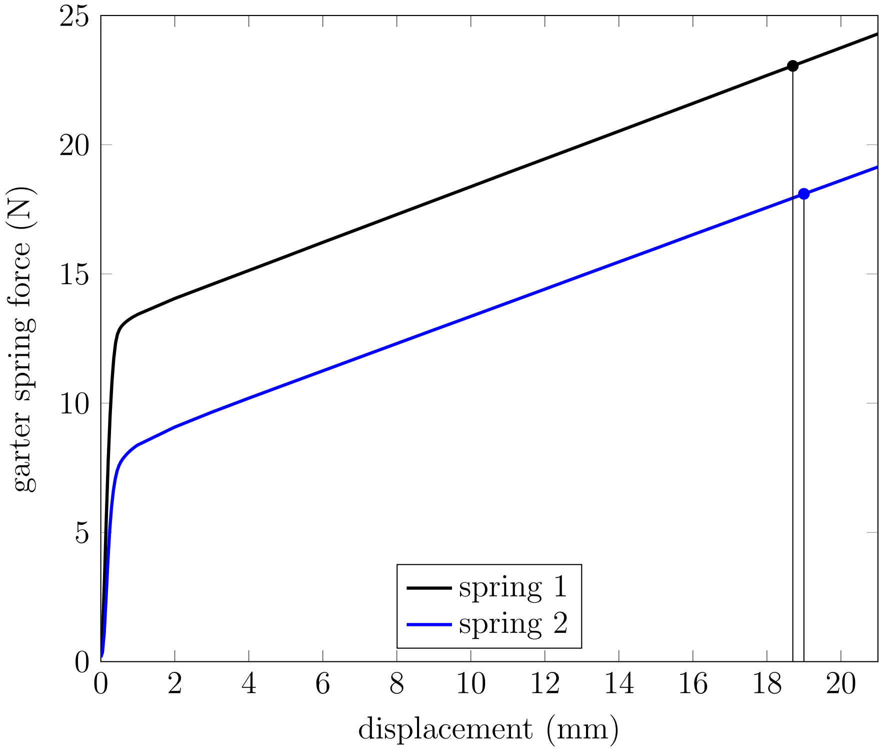


Figure 4.12. Force-displacement diagram of the two springs mounted on the segmented ring. The marked point represents the spring load in the mounted configuration.

When mounted on the segmented ring spring 1 acts with 23.1 N and spring 2 with 18.1 N on the ring. From the measurement the contact force F_C can be evaluated with equation (4.2) for a single ring segment

$$F_\mathrm{C} = (18.1 + 23.1)\sqrt{2(1 - \cos(14.62))} = 10.5\,\mathrm{N}$$

and with equation (4.3) the total contact force for 20 piece ring is

$$F_\mathrm{C\,t} = 209\,\mathrm{N}.$$

4.1.3.3 Measurement

Cleaning with ethanol removes any oil, grease or other contamination from ring and rod and ensures non-lubricated measurement conditions. The segmented ring is placed on the rod with the mounting device. The rod's and the ring's weight are considered by setting the load cell to zero with the ring mounted on the rod, both freely hanging from the load cell. The ring is located inside the upper recess of the T-cup.

The coefficient of friction is determined dependent on velocity. The rod speed is varied from 10 to 1000 mm min^{-1}. The latter is the testing machine's maximum speed. The total displacement of the rod depends on the relative velocity. For effective testing the total displacement is lower at lower speeds.

Each measurement consists of at least 5 cycles. Within a cycle the rod is first pulled upwards, then pushed downwards. The higher the relative velocity the higher the number of cycles, see Figure 4.13 and 4.14 and Table 4.1.

The coefficient of friction is determined from the mean contact force over the measurement distance, disregarding the first and last 10 mm of the travelled distance. Thus the result is not influenced by acceleration or deceleration of the rod or changes in contact mode (sliding/sticking) at the turning or the starting event.

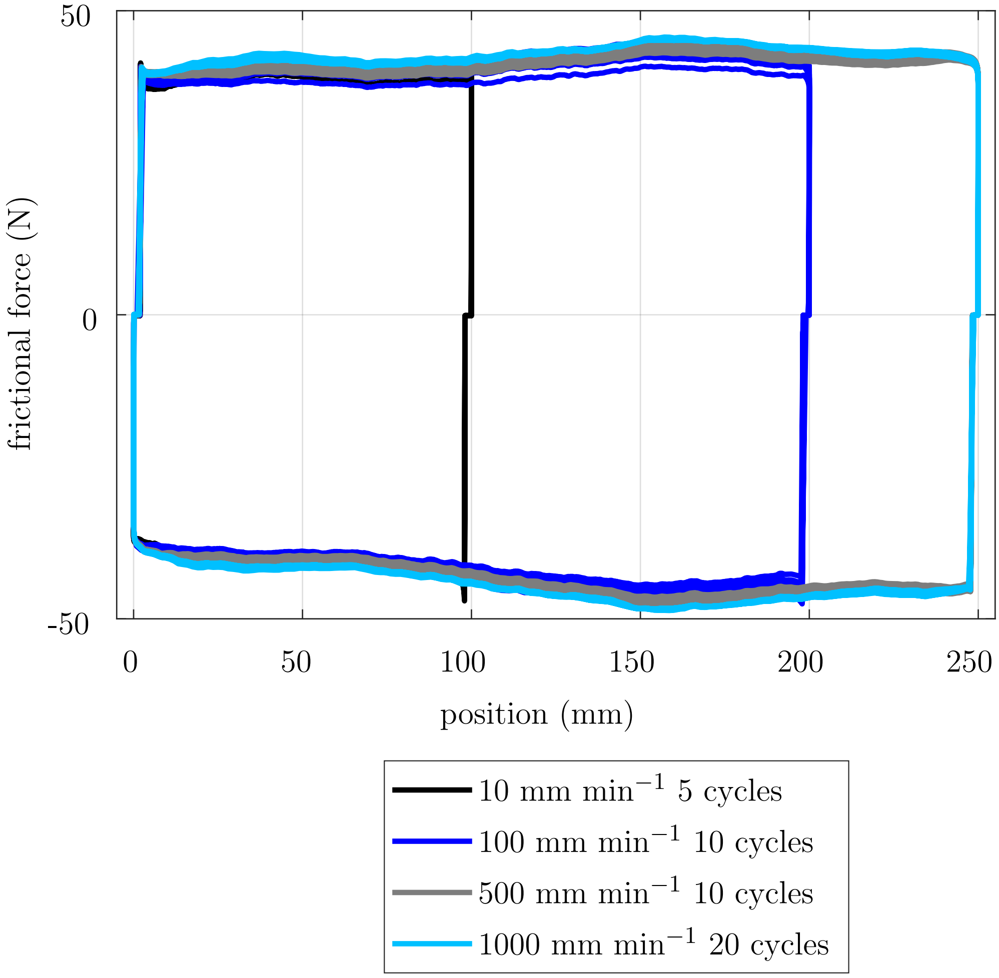


Figure 4.13. Frictional force measurement on the steel rod.

For each cycle the mean force in upward and downward direction is calculated and the coefficient of friction (μ_{up} and μ_{down}) is determined with equation (4.1). In table 4.1 the coefficients of friction is shown including the standard deviation. The measurement is highly reproducible over the performed measurement cycles, with a maximum standard deviation of 2.5 % for the steel rod and 0.42 % for the WC rod, when considering the only one direction of motion. We assume that the coefficient of friction is independent from the direction because both surfaces are smooth and have no orientation or surface profile.

Figure 4.15 presents the mean value calculated from both directions of motion including the standard deviation.

The measurements presented in Figure 4.13 show no significant influence of the roughness R_a (Figure 4.4) on the frictional force. The frictional force variation along the axial rod position is not coincident with the roughness profile measured. For the measurements on the WC rod with constant roughness, Figure 4.14, the frictional force shows more pronounced slip-stick over the whole measurement length than on the steel rod. In both measurements the axial clearance between ring and cup can be seen at the turning points. The coefficient of friction varies more strongly on the WC rod. It changes from 0.16 to 0.20 between 10 and 1000 mm min^{-1}, respectively. On the steel rod the coefficient increases from 0.19 to 0.21 over the same relative velocity range. Figure 4.15 and Table 4.1 show the detailed measurement data including the number of cycles and the standard deviation for both rods.

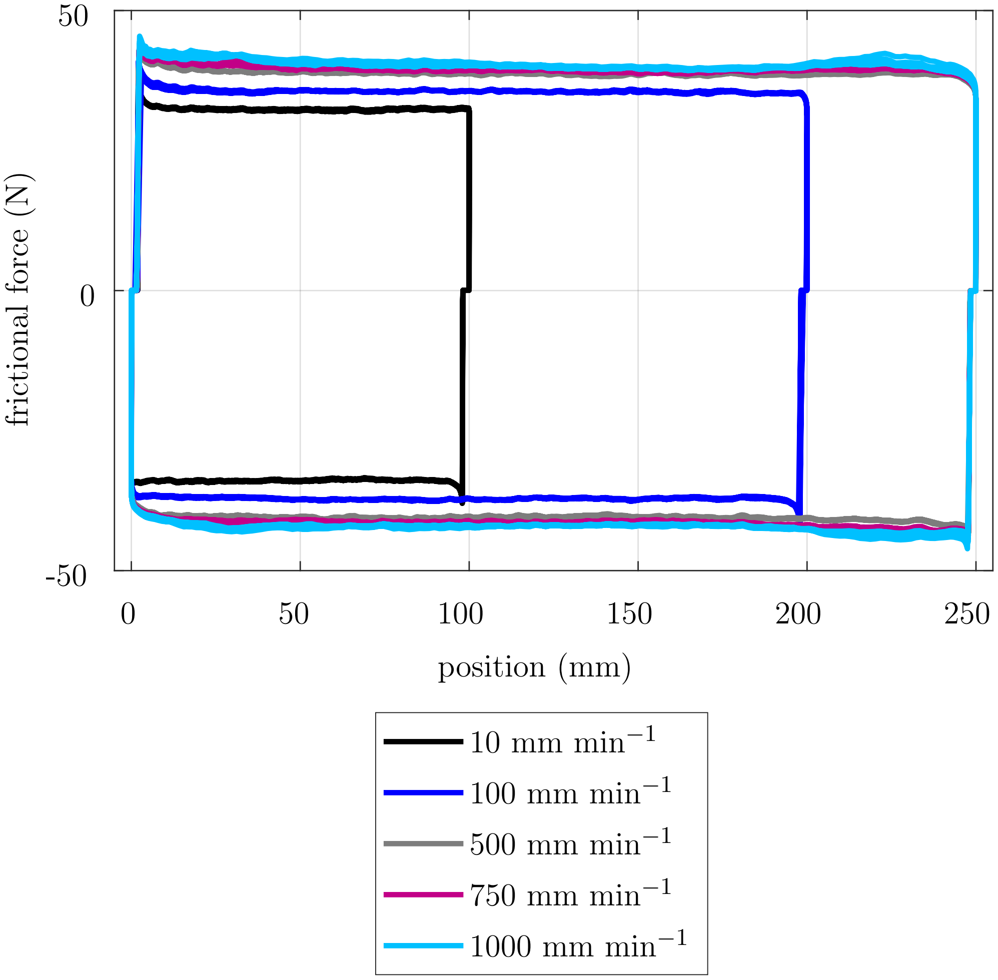


Figure 4.14. Frictional force measurement on the WC rod, for every rod speed 10 cycles were run.

Table 4.1. Detailed velocity dependent measurement data from the determination of the coefficient of friction for HY54 running on a steel rod or on a WC rod. The mean value and its standard deviation is given for the rod movement upwards, downwards and independent from the movement direction.

material	relative velocity (mm/min)	displacement (mm)	cycles	μ_{up}	μ_{down}	μ
steel HY54	10	100	5	0.1900 ± 0.0029	0.1933 ± 0.0019	0.1916 ± 0.0030
	100	200	10	0.1973 ± 0.0050	0.2061 ± 0.0042	0.2017 ± 0.0064
	500	250	10	0.2002 ± 0.0023	0.2094 ± 0.0022	0.2048 ± 0.0051
	1000	250	20	0.2048 ± 0.0005	0.2138 ± 0.0009	0.2093 ± 0.0045
WC/Co/Cr HY54	10	100	5	0.1547 ± 0.0005	0.1623 ± 0.0003	0.1585 ± 0.0038
	100	200	10	0.1701 ± 0.0002	0.1778 ± 0.0003	0.1739 ± 0.0039
	500	250	10	0.1861 ± 0.0004	0.1942 ± 0.0003	0.1901 ± 0.0041
	750	250	10	0.1891 ± 0.0003	0.1988 ± 0.0003	0.1940 ± 0.0049
	1000	250	10	0.1931 ± 0.0008	0.2023 ± 0.0006	0.1977 ± 0.0046

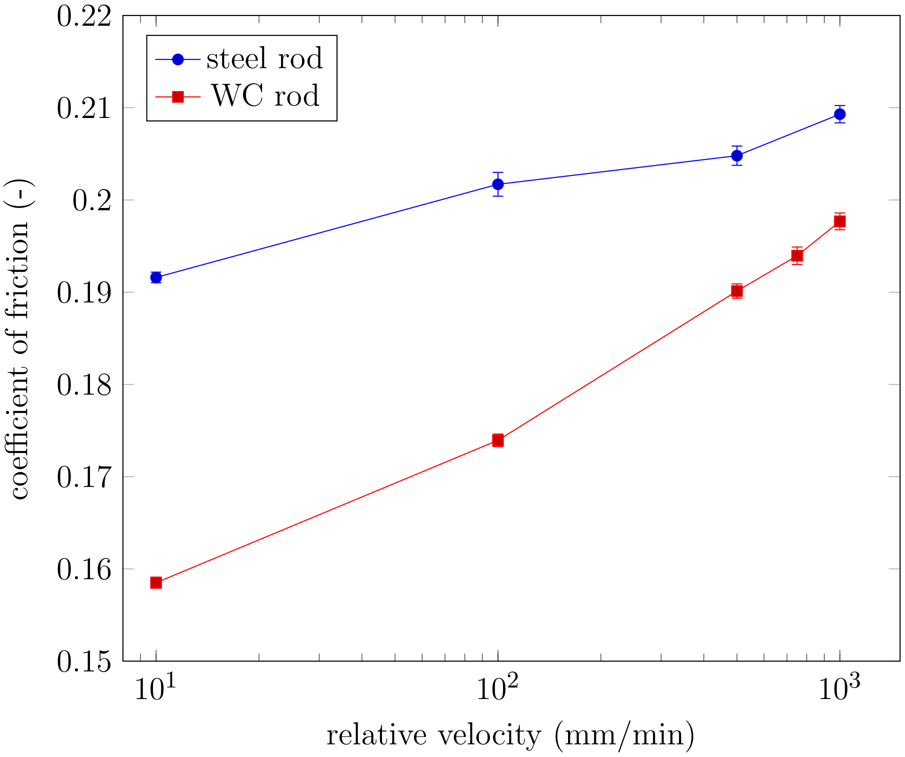


Figure 4.15. The coefficient of friction and its standard deviation over rod speed.

4.1.4 Wear coefficient

A purpose designed test rig, supplied by Phoenix Tribology LTD, Newbury, England, allows to determine the coefficient of wear. The test rig is of pin-on-flat type for linearly reciprocating motion. The test rig has four test stations on two sides. On each side the specimens are opposing each other, being separated by a heatable, moving block. On this block the exchangeable counter face is fixed. The cartridge heater in the block allows to set a minimum temperature for the counter face. The block temperature is controlled and tracked using an infrared sensor. If too much frictional heat is generated during a test, the block temperature becomes higher than the selected set value. Figure 4.16 gives an overview of one of the two equal sides of the test rig. Figure 4.17 shows more details. In this work the counter face is a plate of 1.4021 steel, 4 mm thick and coated with WC. The same coating as typically used as rod coating (WC/Co/Cr, 86/10/4 [31]).

The stroke during a test is fixed. For the presented results it is set to 50 mm. The frequency of the reciprocating motion can be set as high as 20 Hz. The stations are placed in two chambers. This allows tests under controlled gas and moisture atmosphere and easier temperature control. The stationary specimens are pressed against the counter face with a defined load up to 200 N. The range of contact pressures is increased by using a smaller and a bigger specimen type. The bigger (⌀18 mm) specimen directly fits into the hemispherical specimen holder. This specimen geometry is optimised to fit into the specimen holder without getting stuck during a test run, Figure 4.18a. The smaller specimen type has a steel adapter into which a cylindrical specimen of ⌀9 mm fits into, Figure 4.18b. The distance between specimen holder and reciprocating counter face is permanently tracked with a capacitive sensor to determine wear.

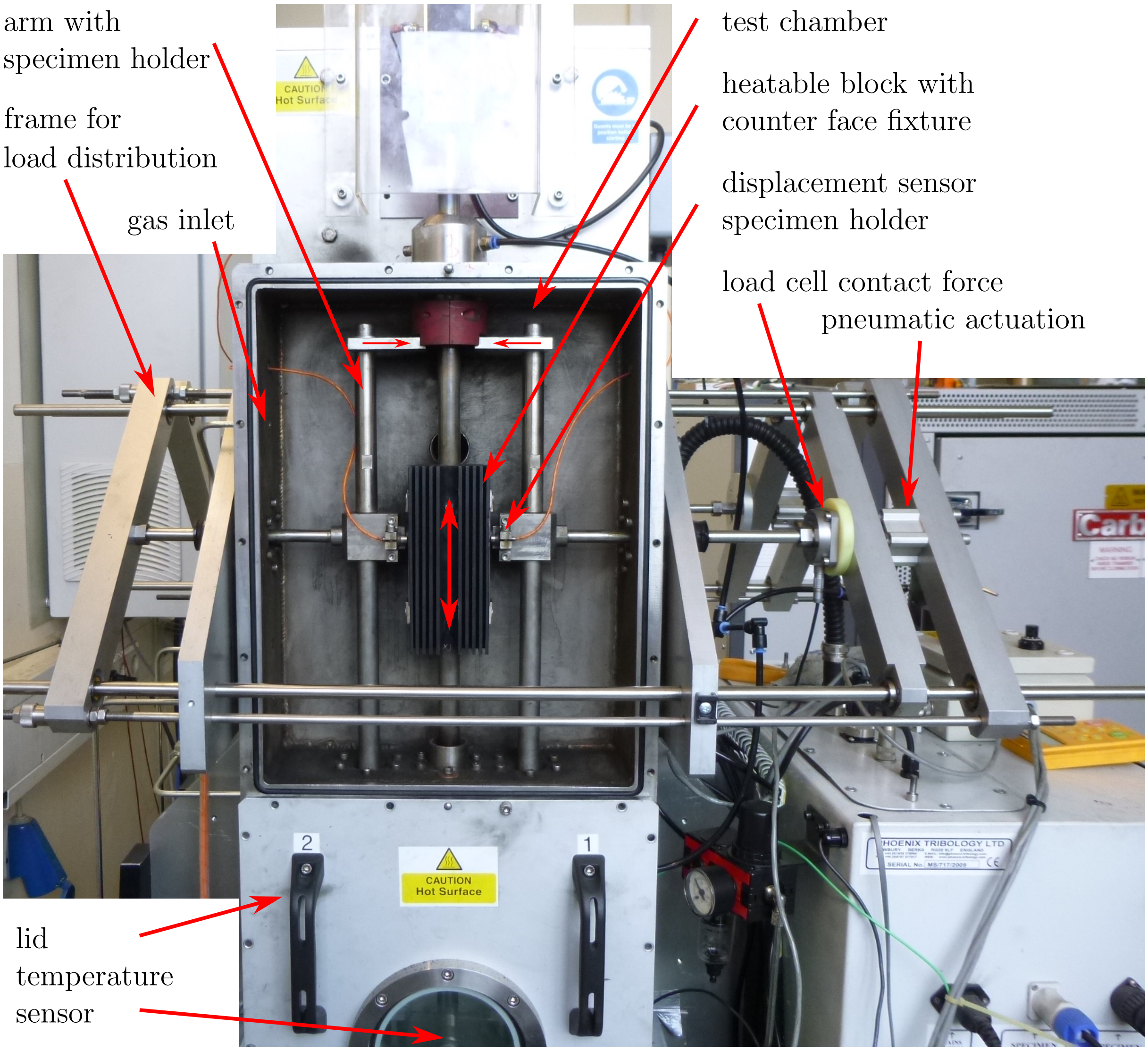


Figure 4.16. Test rig front view.

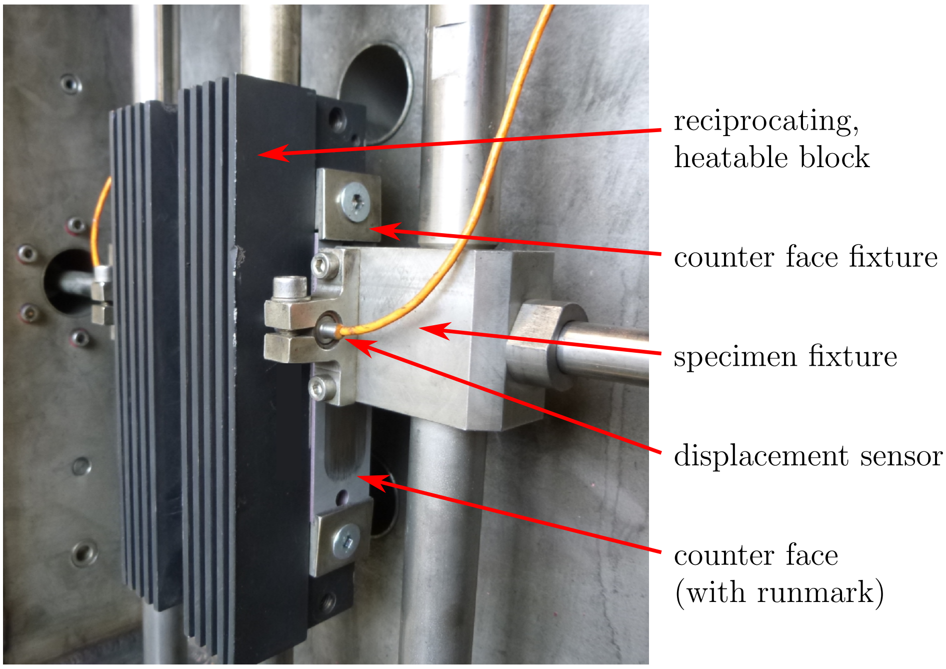


Figure 4.17. Test chamber showing the counter face fixed on the reciprocating, heatable block, the displacement sensor and the specimen fixture.

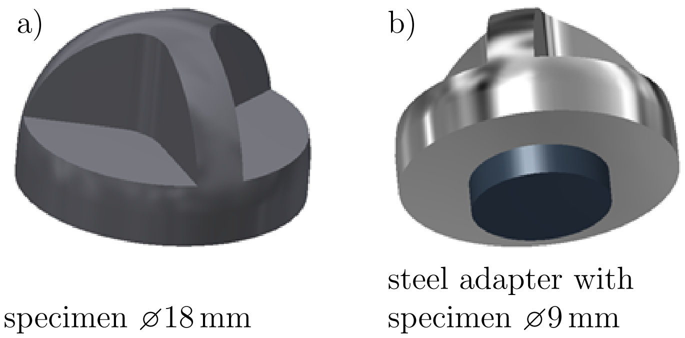


Figure 4.18. Specimen types: a) shows the bigger ⌀18 mm specimen, b) shows the smaller ⌀9 mm specimen in a steel adapter.

With a single test run four specimens can be tested simultaneously. Figure 4.19 shows four counter faces with run marks and the used specimen after a test. A test run of a material with high wear resistance takes several days to produce trustworthy results. During this time the two specimen holders of one side frequently shift their position symmetrically. Figure 4.20 shows the displacement data of all four stations. These shifts are compensated in the post processing by using the mean value of the displacement sensors of two opposing specimens of one side of the test rig, see Figure 4.21. From the mean values of the sides the wear coefficient is calculated with equation (4.4). The specimen's contacting area is A_s, p_c the contact pressure, v_M the mean velocity of the reciprocating motion. Δt is the time period over which the specimen height has worn by Δh_s. The wear coefficient for both sides and as mean value over all stations is given in Table 4.3. Table 4.2 lists the test parameters.

$$k = \frac{A_\mathrm{s} \Delta h_\mathrm{s}}{p_\mathrm{c} v_\mathrm{M} \Delta t} \tag{4.4}$$

Table 4.2. Test parameters.

parameter	value
block temperature	80 °C
contact pressure	1 MPa
mean relative velocity	$1\,\mathrm{m\,s^{-1}}$
gas	nitrogen
humidity	bond-dry

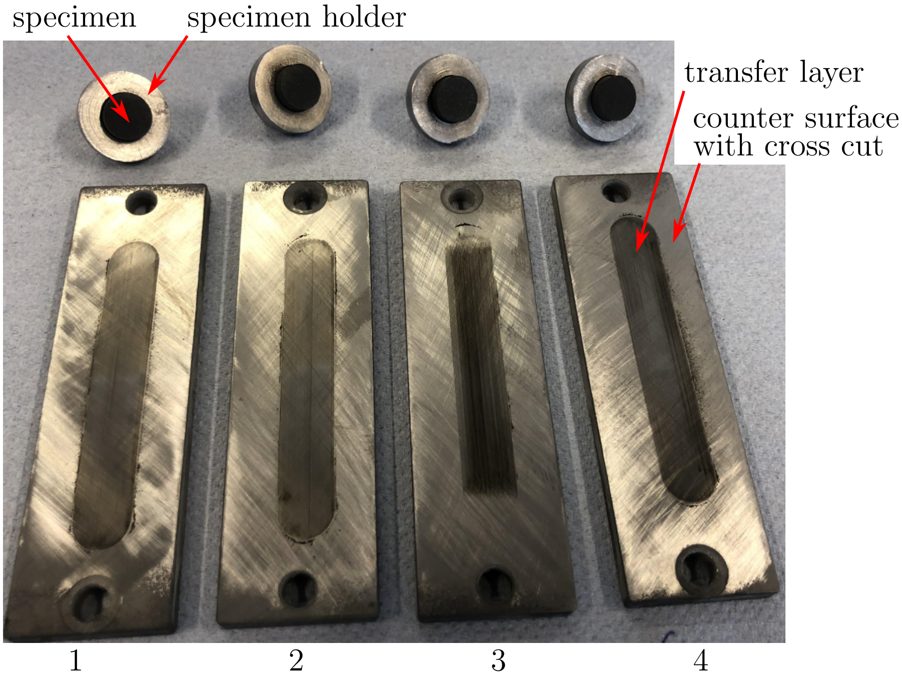


Figure 4.19. Post test: The specimens (black) are still in the steel adapter. On the counter surfaces a transfer layer is visible. The cross cut on the counter face is still visible around the transfer layer. The numbers below the picture are the labels of the stations.

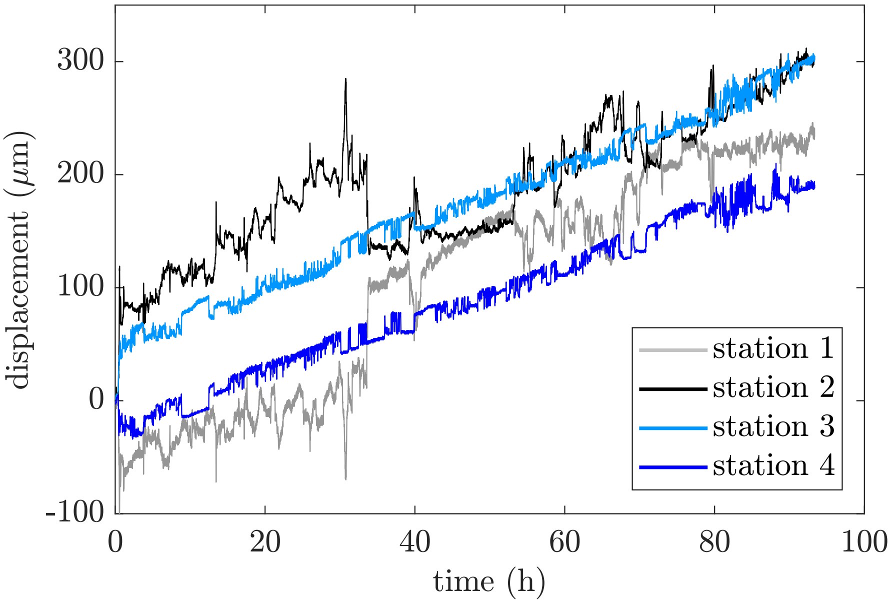


Figure 4.20. The displacement (wear) over time of 4 specimens occurring during the same run. Station 1 and 2 are on one side of the test rig, station 3 and 4 on the other side. The specimen holders on one side tend to symmetrically shift their position due to vibrations.

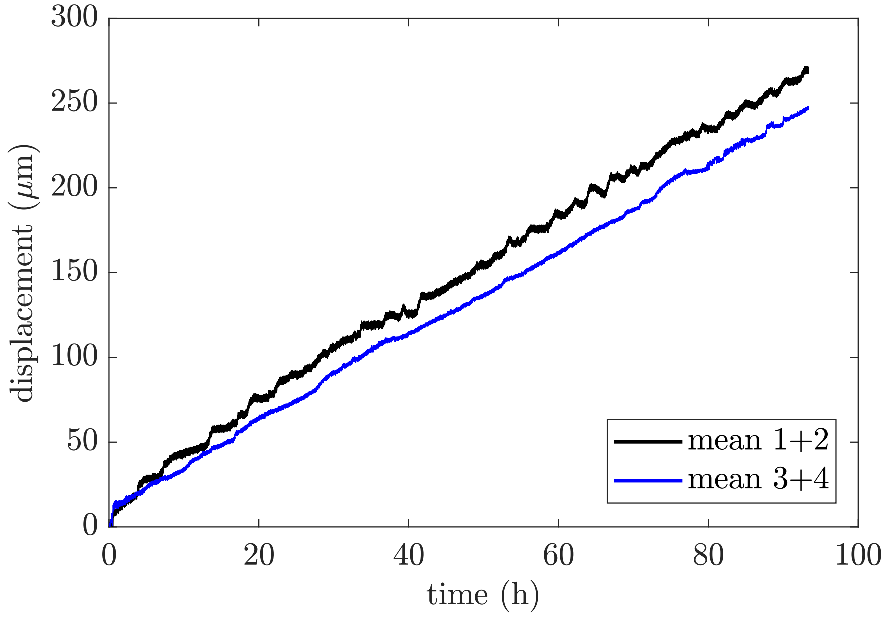


Figure 4.21. The mean displacement (wear) over time is given for each test side on the tribo rig.

Table 4.3. Resulting coefficient of wear.

stations	wear coefficient ($\mathrm{mm\,N^{-1}\,m^{-1}}$)
1+2	$7.48 \cdot 10^{-7}$
3+4	$6.90 \cdot 10^{-7}$
all	$7.2 \cdot 10^{-7}$

4.2 Mechanical modelling

This section motivates and explains the choice of the material model and discusses its features. Subsequently, the material characteristics are briefly described.

4.2.1 Basic considerations

The list below poses several questions, which help to choose an appropriate material model. Close consideration of the problem's boundary conditions, experiments and (field) experience help to find answers.

- What material characteristics have to be described by this model?
    - As long as a sealing element is operating constantly, i.e. a compressor running without stops, a material model is sufficient that correctly describes the loading process over time. It is then not necessary to differentiate between (visco-)elastic or plastic material behaviour, only the time dependency has to be modelled correctly.
    - If the material model has to describe starts and stops too, a true visco-elasto-plastic material model has to be used.
- In which ranges will the temperature, time, stresses and strains be?
    - The temperature boundary conditions (rod, packing and gas temperature) are known from field experience and laboratory testing (e.g. [43], section 6.3).
        - The rod temperature ranges between -150 and 180 °C. Higher temperatures - up to 350 °C - occur sometimes, often resulting in a failure of the packing rings and rod damage.
        - The packing temperature depends on the gas temperature and the surrounding (outside or room) temperature and if a packing cooling is installed. Another complication is that a packing can have a high temperature gradient from the inner to the outer diameter. The relevant temperature for

the sealing element in the packing is the inner wall temperature, which is assumed to be between 20 and 180 °C.

    - The gas temperature is assumed to be between suction temperature and discharge temperature, thus ranging between the extreme values $-180\,^\circ\mathrm{C}$ and $250\,^\circ\mathrm{C}$.

  - The lifetime of piston sealing rings is currently about one year for non-lube service and up to three years in lubricated service [57]. Understandably packing rings have to have equal or higher lifetimes.
  - The assumption of a constant loading situation and the known geometry allows to efficiently determine the stress and strain limits for the material model. The problem is strain controlled, since the sealing elements are wearing, thus deforming, until their end of life. The local strains and (bending) stresses in the ring segments can be estimated by a beam model.

- What other parameters influence the material behaviour?
  - The gas type and the gas moisture can have significant influence on the wear behaviour.
  - A material can be non-isotropic, e.g. stress-strain asymmetry between the tension and the compression regime.
- Which of the afore mentioned parameters have the biggest impact in the current problem?
  - Temperature concerning the stress-strain behaviour.
  - Moisture concerning the non-lubricated wear behaviour.
- Which material model is the fittest for the purpose?
  - This question is difficult to answer, since several aspects have to be considered and weighted:
    - the necessary complexity and accuracy of the material model,
    - the effort to obtain sufficient material data to calibrate the model
    - the numerical stability.

Choice: A linear elastic material model is far from the optimum fit, but it is numerically stable and a good choice for first estimations, analytical considerations and verification purposes. Furthermore, when the ring is in a thermally quasi-steady state and the ring wear is high, a linear elastic material model gives a reasonable response.

4.2.2 Analytical beam model

The packing ring's deformation is now reduced to a bending beam problem. A straight beam with a constant load over the beam length is assumed. Figure 4.22 sketches the model in the undeformed and deformed state, further the position of the neutral fibre and the boundary fibres. The typical conventions of the beam theory are applicable:

- The beam's cross section dimensions are small compared with the beam length.
- The cross section stays plane and is not deformed by shear.
- The beam is a Hookean body, i.e. linear elastic and isotropic.

Additionally, it is assumed that:

- The cross section is constant over the beam length.
- The load is constant over time.

Note that for the sake of simplicity the BCD's geometry is approximated with a straight beam. Using the theory of curved beams yields only slightly different results while adding much more complexity. The desired estimation of (strain) magnitudes can be derived from the theory of straight beams in a straightforward way.

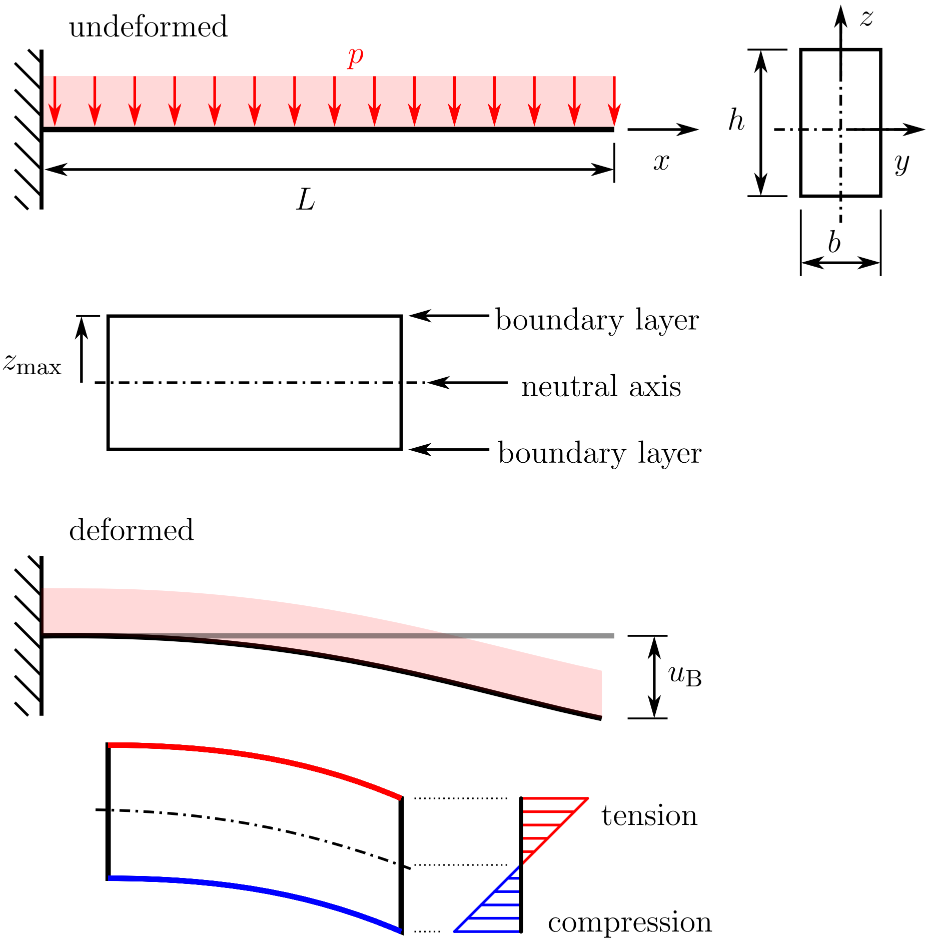


Figure 4.22. Beam model in the undeformed (top) and deformed (bottom) state. The beam dimensions L, b and h are derived from the BCD's geometry.

The beam dimensions are derived from the BCD ring geometry. Figure 4.23 presents the relevant dimensions of a BCD ring to develop the beam model: the outer diameter $D^{(\mathrm{o})}$, the inner diameter $D^{(\mathrm{i})}$, the radial ring height h and the two characteristic angles α and β.

Using a quarter model takes advantage of the BCD's symmetry and explains the assumption of a fixation on one side of the beam. The complex BCD geometry is reduced to a beam with rectangular, constant cross section. The medium ring diameter and α and β define the beam length, see equation (4.6). The beams cross section is equal to the full cross section of a BCD ring: The axial ring width b is constant and 8 mm, the radial ring height h depends on the inner ring diameter $D^{(\mathrm{i})}$ (rod size).

$$h = \frac{D^{(\mathrm{o})} - D^{(\mathrm{i})}}{2} \tag{4.5}$$

$$L = \frac{(\frac{D^{(\mathrm{i})}}{2} + \frac{h}{2})(\alpha + \frac{\beta}{2})\pi}{180} \tag{4.6}$$

Figure 4.24 presents the influence of the rod size (equal to $D^{(\mathrm{i})}$ of a virgin ring) on the BCD's other geometric features.

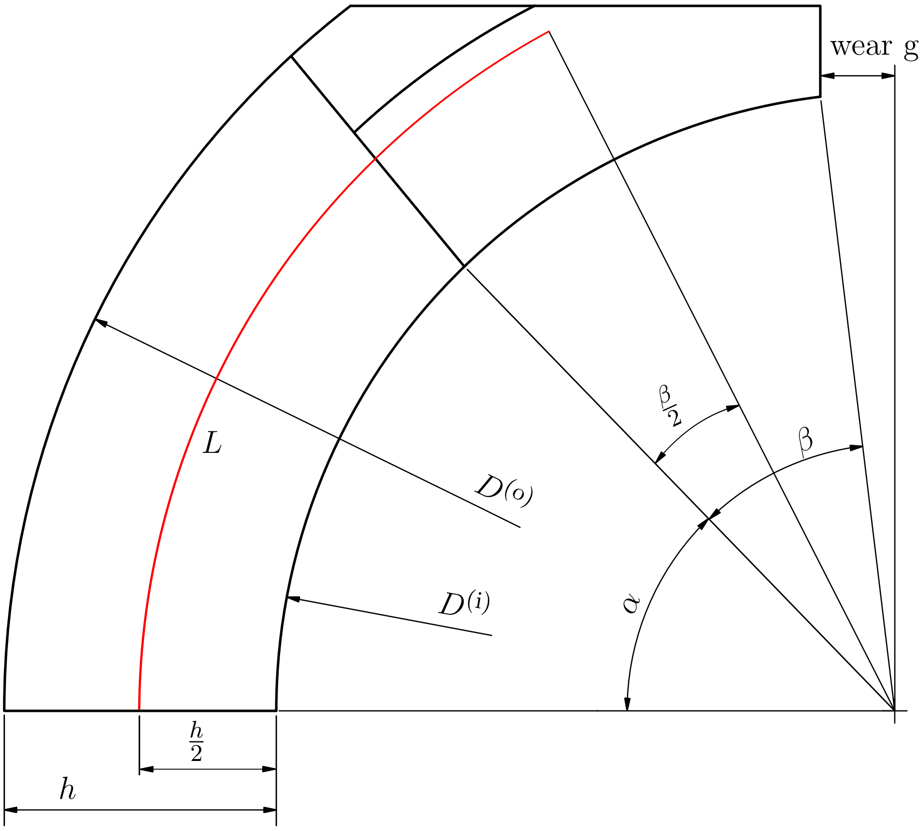


Figure 4.23. Geometric values needed to determine the size of the beam. L is calculated from the mean diameter and the angles α and β.

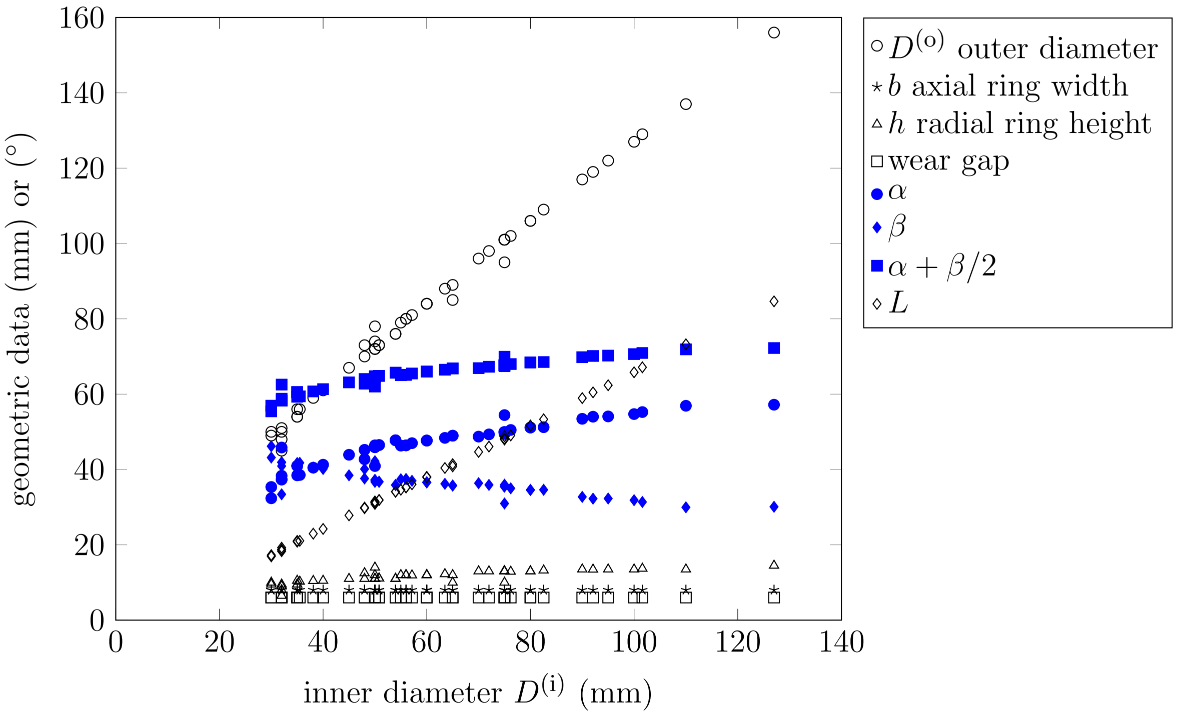


Figure 4.24. Dependence of the geometric BCD ring features on the rod size (equals $D^{(\mathrm{i})}$): the outer diameter $D^{(\mathrm{o})}$, the radial ring height h and the wear gap in (mm) and the angles α, β and their combination $\alpha + \beta/2$ in (°). A representative beam length L is calculated with equation (4.6).

The following is basic mechanics. The equations of the bending momentum $M_\mathrm{b}(x)$ (4.7), the deflection curve $d(x)$ (4.8), and the (maximum) deflection at the beam's free end u_B (4.9) are dependent on the pressure load p and the beam geometry. The wear and u_B are assumed to be of the same magnitude. The second momentum of area J is given for completeness in equation (4.10).

$$M_\mathrm{b}(x) = -p\frac{(L-x)^2}{2} \tag{4.7}$$

$$d(x) = \frac{p}{2EJ}\left(\frac{L^2x^2}{2} - \frac{Lx^3}{3} + \frac{x^4}{12}\right) \tag{4.8}$$

$$u_\mathrm{B} = d(x = L) = \frac{p}{2EJ}\left(\frac{L^2x^2}{2} - \frac{Lx^3}{3} + \frac{x^4}{12}\right) \tag{4.9}$$

$$J = \frac{bh^3}{12} \tag{4.10}$$

Since the modelled problem is strain controlled, the deflection at the beam end u_B is the only input parameter to evaluate the outer fibre elongation ε_b. Equation (4.11) allows to evaluate the needed pressure p to get a certain deflection u_B. The stress in the outer fibres σ_b can be calculated either from the bending momentum M_b and geometry parameter or from Hooke's law, when assuming elastic behaviour, equation (4.12).

Inserting equation (4.7) into (4.12) and substituting the pressure p as given in (4.11) results in the purely kinematic relationship between the outer fibre elongation ε_{b} and the deflection at the beam end u_{B}. The outer fibre strain is independent of the material. The beam height h has a linear relationship with the fibre strain via the distance between the neutral fibre and the outer fibre $z_{\max} = h/2$, see equation (4.13).

$$p = \frac{8u_{\mathrm{B}}EJ}{L^4} \tag{4.11}$$

$$\sigma_{\mathrm{b}} = \frac{M_{\mathrm{b}}}{J} z_{\max} = E\varepsilon_{\mathrm{b}} \tag{4.12}$$

$$\varepsilon_{\mathrm{b}} = \pm \frac{4u_{\mathrm{B}}(L-x)^2}{L^4} z_{\max} \tag{4.13}$$

The assumption in beam theory concerning the beam dimensions is violated, when considering small rod diameters. The full beam length for one sealing segment is $2L$. The ratio $2L/h$ gives an estimation how badly the assumption is violated. The full beam length ($2L$) is less than 5 times greater than the beam height h for a rod diameter smaller than 45 mm. In the worst case the ratio is 3.6 for a 30 mm rod.

Nonetheless, the evaluation of the maximum outer fibre strain at $x = 0$, equation (4.13), is performed for rod diameters between 30 and 127 mm. The ring dimensions are taken from existing rings. Between 30 and 60 mm rod diameter the evaluation is done in steps of 5 mm, beyond 60 mm in steps of 10 mm or bigger, see Figure 4.25. This allows to draw several conclusions:

- The strain range the material model has to perform accurately is up to 10 %.
- For bigger rings the strains are smaller by one magnitude, i.e. 1-2 % strain.
- Calculating wear of a few tenth of a millimetre will result in strains below 1 %. This means that the material model has to perform well in the range of small strains.
- A viable material model has to be well calibrated between 0.1 and 5 % strain, but has to be able to handle higher strain values as well (up to 10 or 12 %). Higher strain values are important when considering i) locally calculated strain maxima and ii) smaller rings.
- If wear is expected in the magnitude of millimetres sealing performance can only be ensured if the structure is compliant enough to adapt the ring's shape to the rod. Overly stiff materials might stop to seal or break.
- The BCD design is not scaling with the rod diameter. The axial width is constant and the radial height increases only slightly with the rod diameter. It is expected that the wear pattern of smaller and bigger rings looks differently because bigger rings have less bending stiffness.
- Equation (4.13) contains no material parameter because the problem is strain controlled. The conclusions drawn give already rough criteria what a sealing element material has to satisfy:

- The above stated strain levels have to be reached without rupture.
- The necessary pressure load to achieve the stated strains has to be compatible with the operating conditions. If the pressure in operation is by far bigger than necessary for the deformation the ring is pressed firmly to the rod and might wear away too quickly. If the pressure is smaller than needed for the deformation the ring might become leaky, if the contact pressure between rod and ring becomes too low.
- If stiff materials are to be used the ring design alone has to compensate wear.

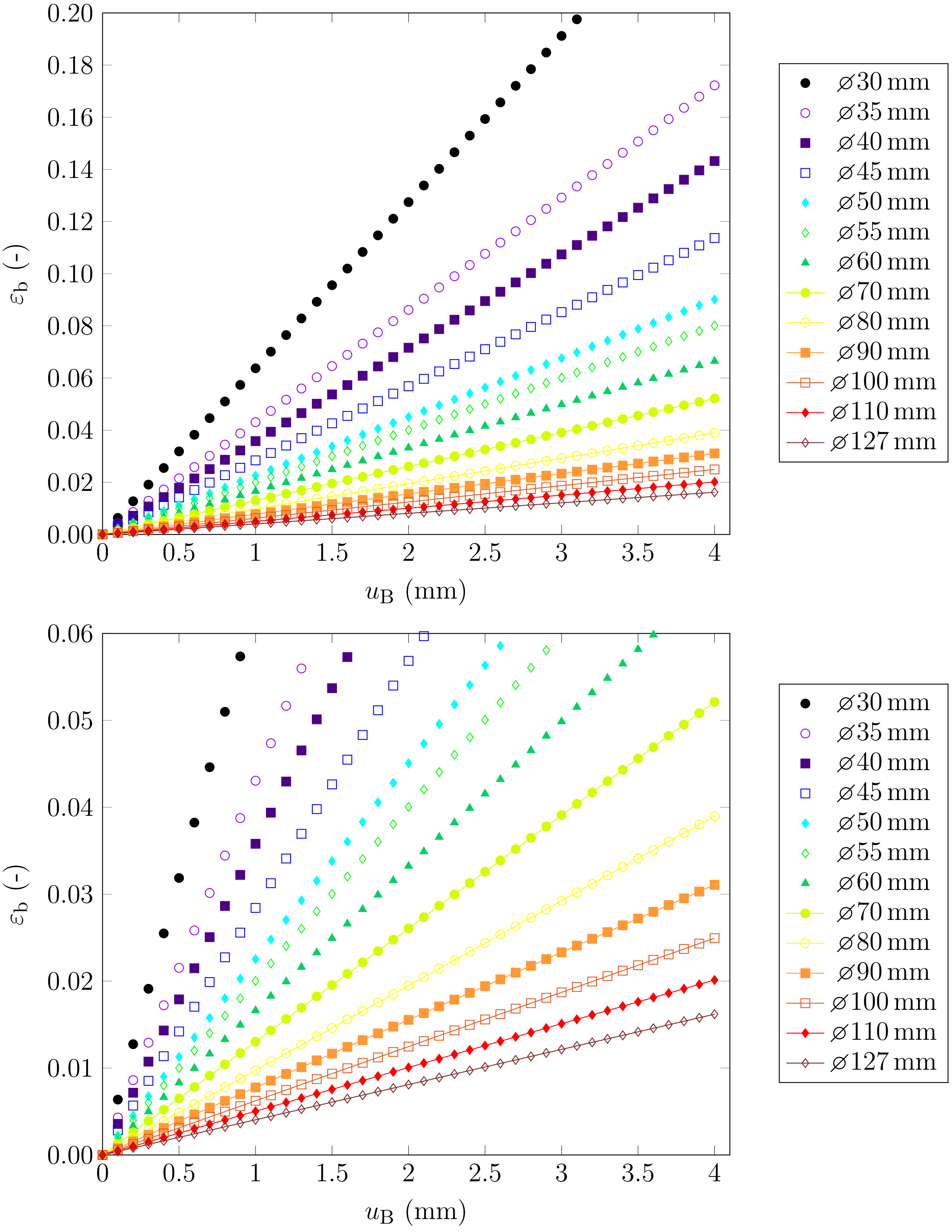


Figure 4.25. The outer fibre elongation over the deflection of the beam end for different inner (rod) diameters. The deflection represents the ring wear. Top: For ring diameters less than 40 mm the bending strains are bigger than 10 %, if a deflection (wear) of 3 mm is applied. Bottom: Zooms of the results between 0 and 6 % strain - the bigger the rod, the smaller the strains.

4.3 HY54

4.3.1 Material composition

The material used for the presented studies is a PTFE based compound with glass fibres and graphite. The trade name of this HOERBIGER grade is “HY54”. The material data sheet is included in the appendix 7. The grade is used in non-lubricated applications, except if the gas is bone-dry. If it is used in a non-lubricated bone-dry application the wear coefficient is significantly higher, which allows accelerated wear testing.

4.3.2 Material characterisation

The PTFE based compound is a rather soft material. The elastic modulus depends on the temperature, the load direction (compression or tension) and the strain rate. At room temperature (23 °C), the influence of the strain rate in tension is already significant. For the strain rates of 0.05, 0.5 and $5\,\%\,\mathrm{s}^{-1}$ the elastic moduli are 1425, 1651 and 1915 MPa, respectively.
Figure 4.26 shows the decay of the elastic modulus with increasing temperature for three strain rates. The average strain rate levels are 0.06, 0.6 and $6\,\%\,\mathrm{s}^{-1}$. The experiments were carefully carried out by Dr. Thomas Dick, tortutec GmbH, Vienna. Over all performed tests the mean Poisson’s ratio is $\nu = 0.4$.

Short-term 5 h creep tests show the material’s reaction to different temperature levels for compression and tension. Figure 4.27 shows the load mode and temperature dependency. From this data a non-linear Marlow model can be derived and implemented in the simulation model. This model includes the experimental time scale, which has to be considered, when interpreting the results. The implementation of this material model is still pending, but the data allows to get a first impression how the material behaves.

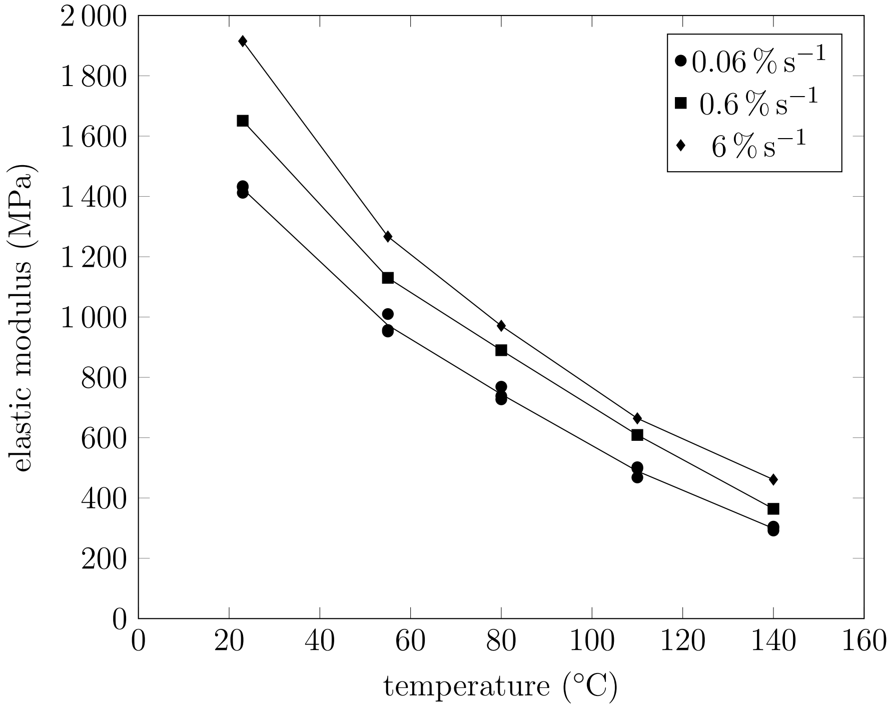


Figure 4.26. Elastic modulus dependent on strain rate and temperature. The lines connect the mean values of multiple data points for each temperature and strain rate.

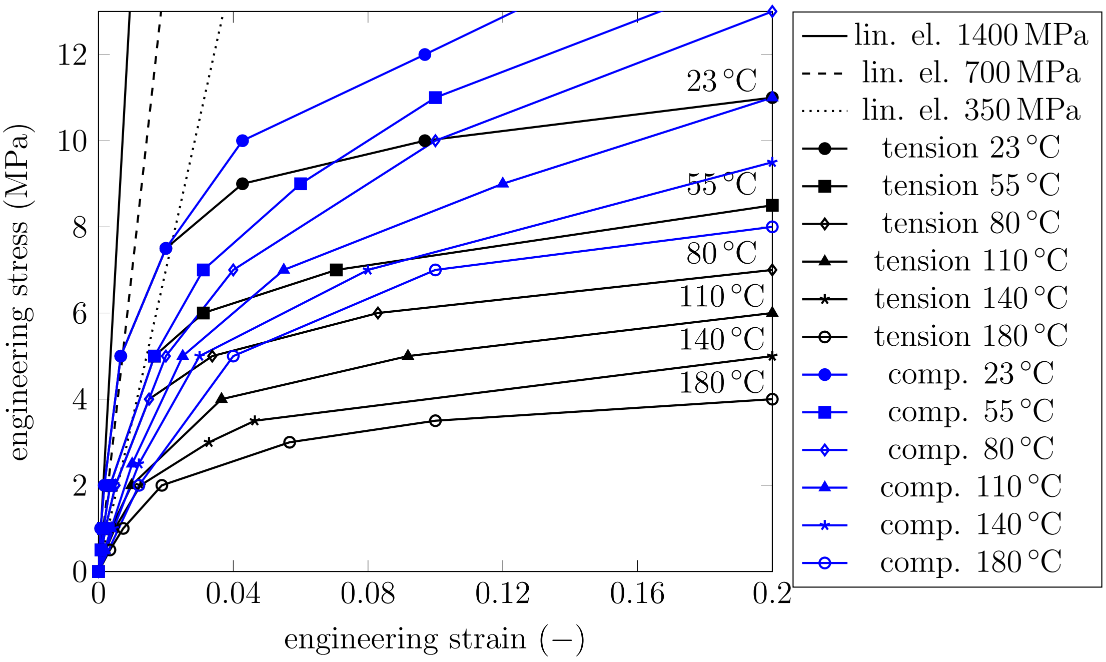


Figure 4.27. Non-linear elastic, temperature dependent material data derived from 5 h tension or compression creep tests. Elastic moduli of E = 350, 700 and 1400 MPa are added to the graph.

Chapter 5

Modelling

This chapter describes the choice of the wear algorithm, its implementation and all necessary additional calculations with or in connection with the commercial FE software package Abaqus, which is available at HW.
Then an analytical wear model of an uncut ring is derived to validate the numerical finite element implementation of the wear procedure.

5.1 Wear algorithm

Literature describes different methods how to implement wear: element deletion, remeshing-techniques or nodal displacement. Each of them has advantages and disadvantages.
Considering the following aspects allows to choose an appropriate implementation approach:

- The goal is to model wear of several millimetres, which may be more than half of the radial ring thickness.
- Packing rings contact with a rod of circular cross section. (This allows a highly regular mesh in the ring partitions near the rod.)
- The ring's sealing performance is only ensured if there is no leakage path from cylinder to crank side.
- The pressure drop from cylinder to crank side has to be considered. Therefore the contact surface has to be piecewise smooth, but may consist of several incoherent patches, which expand or contract over time due to wear.
- The computation time must be observed.

Element deletion was discarded, because by definition this method deletes elements, which creates non-physical leakage paths or holes in the contact surface. This type of wear modelling technique seems to be more applicable to erosive wear like in rock cutting/breaking [13, 14, 45, 85].
The procedure of *re-meshing* the geometry after every n$^{\text{th}}$ wear cycle can be used in two ways. First, the wear is calculated in post-processing and then the whole geometry is re-meshed. Second, the re-meshing is used only for the interior of the wearing solid body to obtain a better mesh (similar to mesh-smoothing, explained later in this section). This method includes mapping all results from the configuration prior to the re-meshing to the post-meshing configuration. Re-meshing is computationally expensive and by default only applicable for a limited number of element types in Abaqus, namely planar continuum triangles and quadrilaterals, shell triangles and shell quadrilaterals and 3D tetrahedrals from the Abaqus library. 4-node tetrahedral elements are overly stiff and not applicable in problems where bending and/or contact stresses are of importance. So this method is discarded as well.

The method chosen here is *nodal displacement*. The Fortran-coded user-defined Abaqus subroutine UMESHMOTION allows to *move nodes without causing stresses or strains*. In this subroutine the displacement of a node within an increment is user-defined, keeping the flux densities constant. The constant strain energy density results in the fact that an element can deform without stresses being caused. If a surface node moves "inside" the geometry, the body loses material, i.e. it wears. (The opposite works as well: If a surface node moves "outside" direction, material is generated.) Figure 5.1 shows how a single element wears proportional to the nodal contact forces, assuming constant velocity for the contacting nodes.

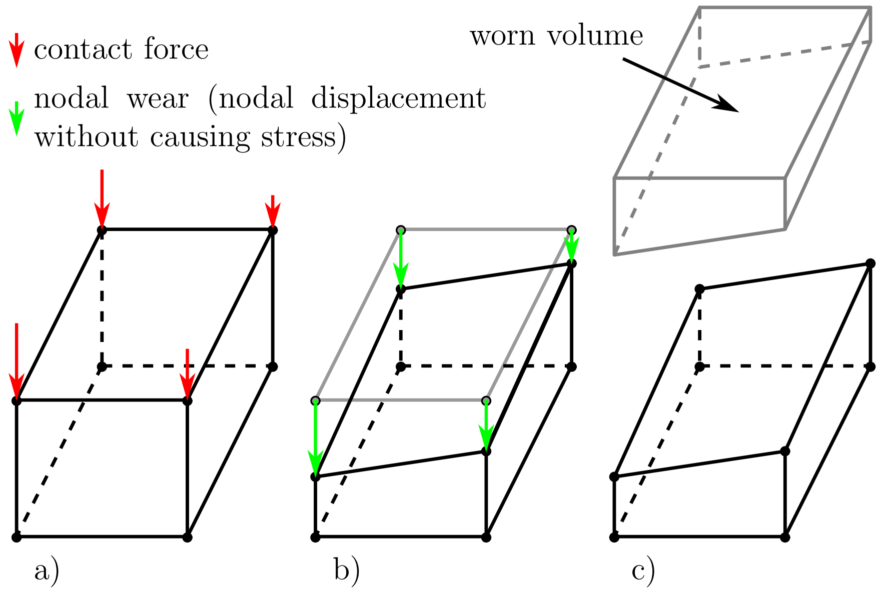
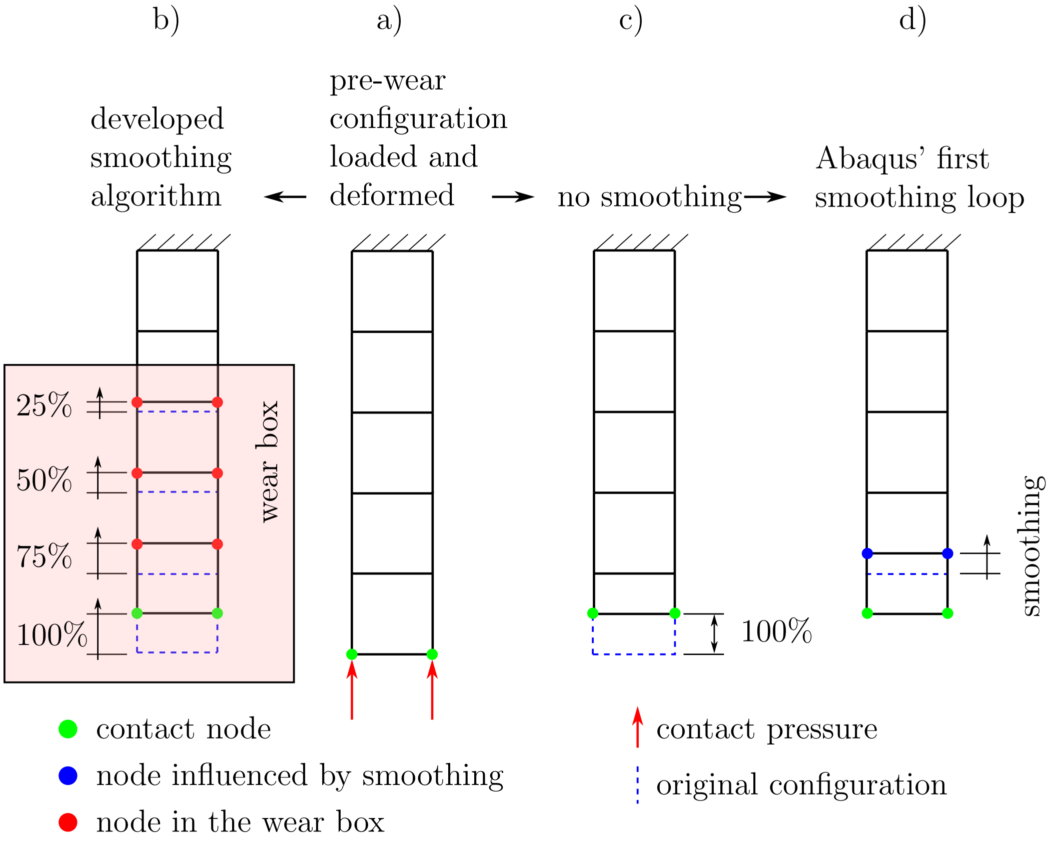


Figure 5.1. Working principle within a single wear increment: a) loaded and deformed configuration, b) displacement of nodes due to wear (proportional to contact force), c) bottom: worn configuration, top: worn material volume.

Wear routines following this principle were already presented in different applications using different FE codes: The software package Ansys was used by [2, 51]. The UMESHMOTION in Abaqus was used by [8, 48, 61, 67]. Hegadekatte et. al [28–30] used Abaqus for their wear calculations. They implemented a “global-incremental wear model“ as well as a “wear processor” (calculates wear in post-processing) and compared both methods to the Abaqus UMESHMOTION.

A natural wear limitation of this method is the element size. After several wear cycles and sufficient wear an element might be inverted by the algorithm. This causes the calculation to abort and has to be prevented.
Abaqus allows to use a built-in mesh-smoothing algorithm to prevent element inversion. This smoothing algorithm moves the nodes inside a meshed solid body and tries to keep the elements next to each other at similar size, see further [18]. The number of runs for the mesh-smoothing can be set manually.



b)
a)
c)
d)
developed smoothing algorithm
pre-wear configuration loaded and deformed
no smoothing
Abaqus' first smoothing loop
25%
50%
75%
100%
wear box
100%
smoothing
contact node
node influenced by smoothing
node in the wear box
contact pressure
original configuration

Figure 5.2. Smoothing algorithms: a) Starting from a loaded and deformed configuration, b) the developed smoothing algorithm moves all the nodes inside a wear box to prevent element distortion. c) Wear of a single element without smoothing. d) Effect of the first run of the Abaqus smoothing algorithm to reduce element distortion.

Abaqus' smoothing algorithm is too inefficient for this wear problem: With a low number

of smoothing runs the contacting elements wear away and start to invert, whereas the elements in the inner part of the mesh are not yet effected by the smoothing, see Figure 5.2. This is caused by the natural delay, which occurs when trying to keep the elements next to each other at similar size. With a higher number of loops of the smoothing algorithm turned out to be too time consuming.
The key to model wear of several element lengths (i.e. several mm) is to introduce a *wear-box*. The idea of a wear-box has already been exploited by Rezaei et al. [61] and Fouvry, Paulin et al. [25, 54]. Earlier Sui et al. [71] described this method, calling it "rezoning". Figure 5.2 shows a comparison between the smoothing algorithms.

5.2 Implementation of wear

This section explains how the wear simulation is set-up, how the chosen wear algorithm is implemented, and how the wear-box is defined. Figure 5.3 presents the quarter model of the BCD ring with the coordinate system, which is used in the following. Figure 5.4 shows a flowchart containing the simulation procedure.

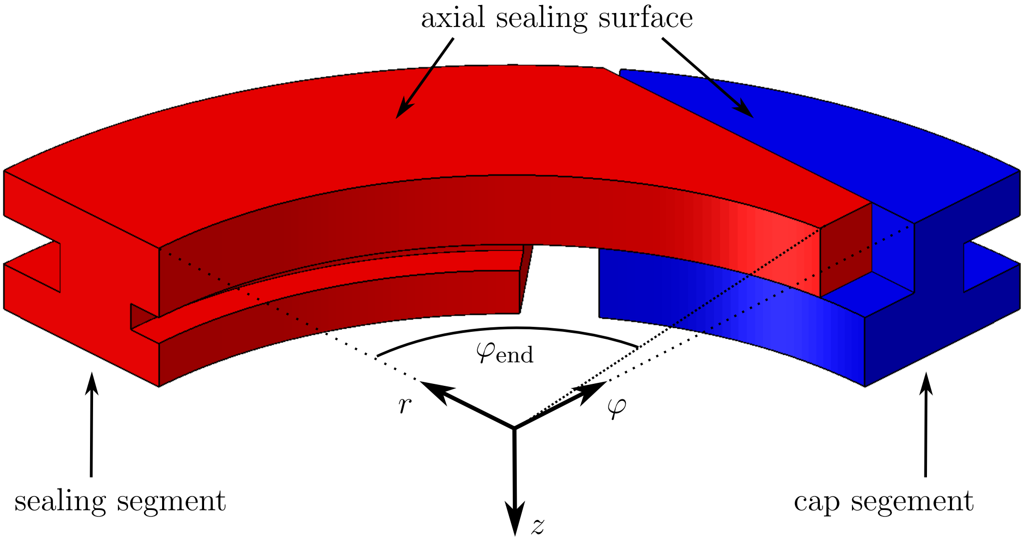


Figure 5.3. BCD quarter model: sealing segment (red) and cap segment (blue) viewed from the crank side. The used cylinder coordinate system originates in the rod axis and the ring's axial sealing surface. φ_{end} gives the position of the sealing segment's tip, which changes due to wear.

Many packing ring designs allow to build a radially expanding wear-box, which fully surrounds the rod. The size of the wear-box is defined as to achieve a maximum wear of 3 mm without re-meshing. The chosen approach requires:

- a regular mesh,

  - where the nodes are aligned in (nearly) radial direction.
  - where the specified number of neighbour nodes is constant for all nodes in the dynamic sealing surface.

- the knowledge of the node labels in the dynamic sealing surface.
- the knowledge of the node labels in the dynamic sealing surface are located on an edge or corner.
- the knowledge of the node labels of the neighbours of each node in the dynamic sealing surface.

A python script finds the neighbour nodes in a post-processing step after the ring has initially been loaded, see appendix 7. The script iterates over all nodes in the dynamic sealing surface. For each contact node and for each following neighbour node the script determines the elements, which are connected to the currently considered node. The correct neighbour node has to be geometrically determined from all nodes of the connected elements. The tolerances depend on the smallest distance between the current node and all surrounding neighbour nodes. The correct neighbour node is found if the following conditions are met:

1. The next neighbour node and the current node have the same axial coordinate (i.e. z-coordinate) (tolerance $= 0.1 \cdot$ smallest distance).
2. The next neighbour node has have a greater$r$-coordinate than the current node (tolerance $= 0.3 \cdot$ smallest distance).
3. The next neighbour node is the potential neighbour closest to the current node.

An additionally implemented sanity check assures that exactly one neighbour is found. If a neighbour node cannot be found or more than one neighbour is found, a warning is issued and the neighbour search has to be solved manually. Figure 5.5 presents a simplified flowchart of the script's logic. The algorithm works very well for regular meshes.

This geometric procedure is computationally expensive. An *advanced guess* is implemented additionally to reduce the search time for the next neighbour node. The geometry of a complex ring design like the BCD's is partitioned to be accessible to meshing. The regular mesh inside the wear-box is exploited by the algorithm because the node numbering within a partition is regular. Thus if the difference in node numbers between the neighbour nodes is known, the next neighbour can be guessed.
After finding all neighbours of the first few (usually 3) contacting nodes the two most frequently occurring node number increments are used to provide a suggestion of the next neighbour node number. The most frequently occurring node number increments are updated through the whole procedure.
If the first guess is wrong, a second guess is made. If it turns out wrong again, the geometric algorithm is used again. For each guess geometric checks ensure that the suggested

node is the correct neighbour node, according to given tolerances. The tolerances depend on the median of the distance between two neighbour nodes, the axial distance between the nodes and the fact that the neighbour has to be radially aligned. The dependence on the median of the distance gives the advantage that the algorithm can be used with different mesh sizes without further changes.
Finally, after all neighbour nodes have been determined, a sanity check provides the information if any node number occurs more than once. If so, a warning message and the surface nodes with duplicates are printed to a log file. The reason for a duplicate has to be investigated manually. (Wrong tolerances set? Mesh quality not high enough? = Nodes not aligned radially? Highly differing element sizes inside the wear-box?)
As a result of this algorithm all surface nodes and their neighbour nodes are written to a comma-separated values (csv) file for further processing. (The first column contains the contact nodes and in every subsequent row the contact node's neighbours are listed from the innermost to the outermost. The use of a text file is beneficial in two ways: it is human-readable and it can easily be read by any programming language (Fortran, Python, Bash).

With the ability to model wear over several element lengths the user-defined Fortran-coded subroutines become more complex. Implementing a MODULE (introduced in Fortran 95) instead of a COMMON BLOCK (used in Fortran 77) allows the independent subroutines to share data. The used subroutines and their main tasks are:

- UEXTERNALDB: This subroutine interacts with external files and is called at the beginning and at the end of every analysis as well as at the beginning and at the end of every increment. It is used to:
    - initialize the calculation at the beginning of the analysis:
        - allocate arrays,
        - read node sets,
        - read the file containing the neighbour nodes of each contact node,
        - read the pressure distribution in the dynamic sealing surface,
        - read the displacement (=wear) value of the nodes within the wear-box if the analysis is restarted after updating the pressure distribution in the dynamic sealing surface. This allows to directly track the total wear of the sealing element and not only the wear within the calculation between two updates of the dynamic sealing surface.
    - update the accumulated wear variable, to allow displaying the wear after every increment.
    - write the accumulated wear variable to a text file at the end of every increment.
    - write the accumulated wear variable to a text file which can be used for the restart of the analysis at the end of an analysis.
- URDFIL: This subroutine is used to gain access to the results file (every increment) and read the current coordinates of the nodes in contact with the rod.

- UMESHMOTION is the core subroutine, which allows to move nodes (every increment) independently from their underlying material. The movement does not create stresses or strains and gives the possibility to model wear. The subroutine is called for every node in ascending order. The contact nodes should have lower numbers than their neighbour nodes in the bulk, to guarantee that their position and wear status is already known and can be used as input for the calculation of the required displacement of the nodes in the wear-box.
  Internal routines provide access to necessary data to calculate wear. Then the nodal displacement is calculated dependent on a node's position: i.e. whether if it is a contact node, a neighbour node or not even within the wear-box. For nodes in the dynamic sealing surface the displacement results in material loss (wear) for the model, whereas the displacement of the bulk nodes prevents element distortion. Figure 5.6 shows the flowchart of the subroutine UMESHMOTION and can be summarized as follows:

  - GETNODETOELEMCONN returns all elements connected to the considered node and provides this information for the next internal routine.
  - GETVRMAVGATNODE returns the value of a chosen output. Here the contact stress 'CSTRESS' is used to obtain the nodal contact force for the considered node.
  - If the node is in contact with the rod, calculate the nodal wear dependent on Archard's equation (3.1). Dependent on the absolute radial wear of the contact node and the number of neighbours in the wear-box the displacement for each neighbour node is calculated and stored.
  - If the node is a neighbour node the stored displacement value is applied.
  - If the node is neither in contact nor a neighbour node the algorithm skips this node.

- DLOAD: This subroutine is used to specify the non-uniform gas-pressure distribution in the dynamic sealing surface, which has been determined by a separate preceding FE calculation. This is of paramount importance to obtain the correct solid-solid contact pressure for the wear calculation, as explained in chapter 3. The pressure values, which were determined at the nodal position, are mapped to the integration points of each element face in the dynamic sealing surface. This mapping is done in the isoparametric space where the position of each integration point is known in advance. A transformation from the isoparametric space to the model space is not necessary, since the position of the nodes or integration points does not change within this routine.

- UFIELD can be used to specify a pre-defined field variable or to display the nodal movement/wear in "Abaqus Viewer". Note that the subroutine displays the values from one increment prior to the current one.

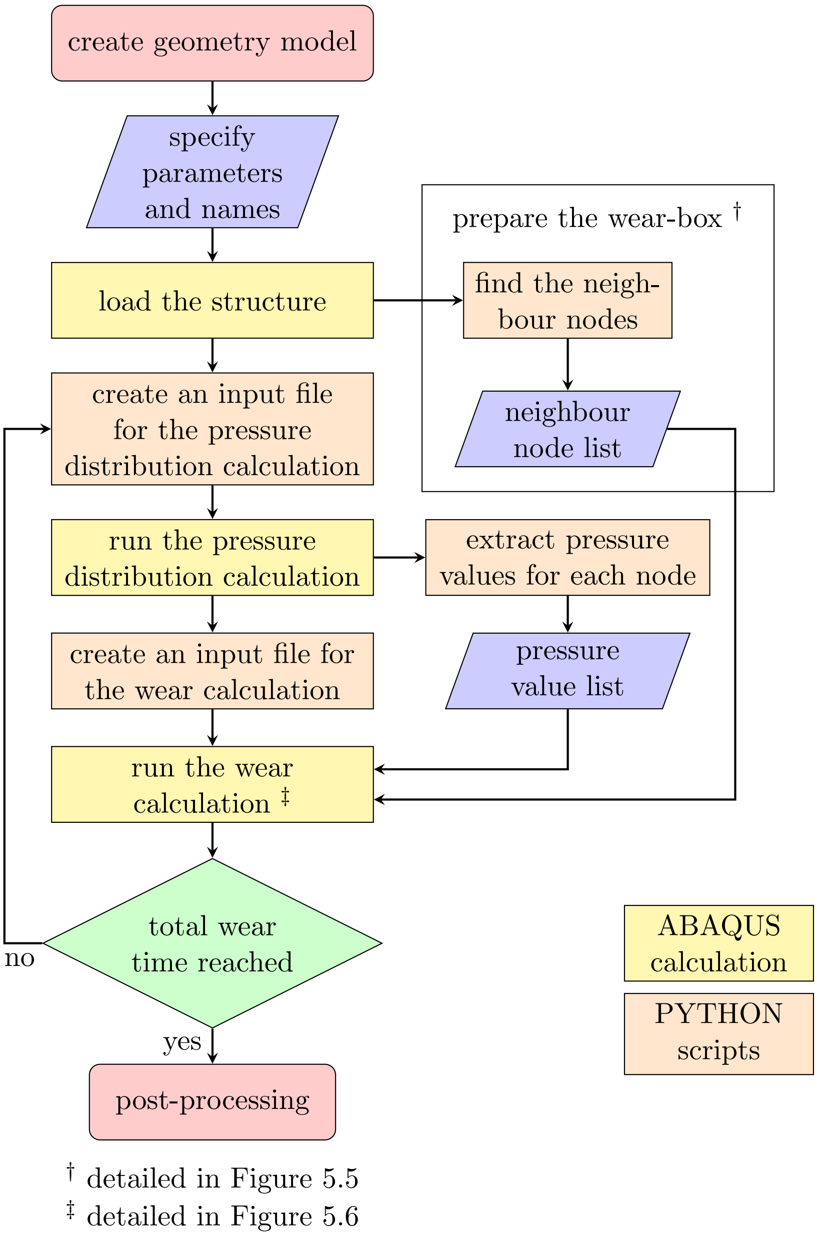
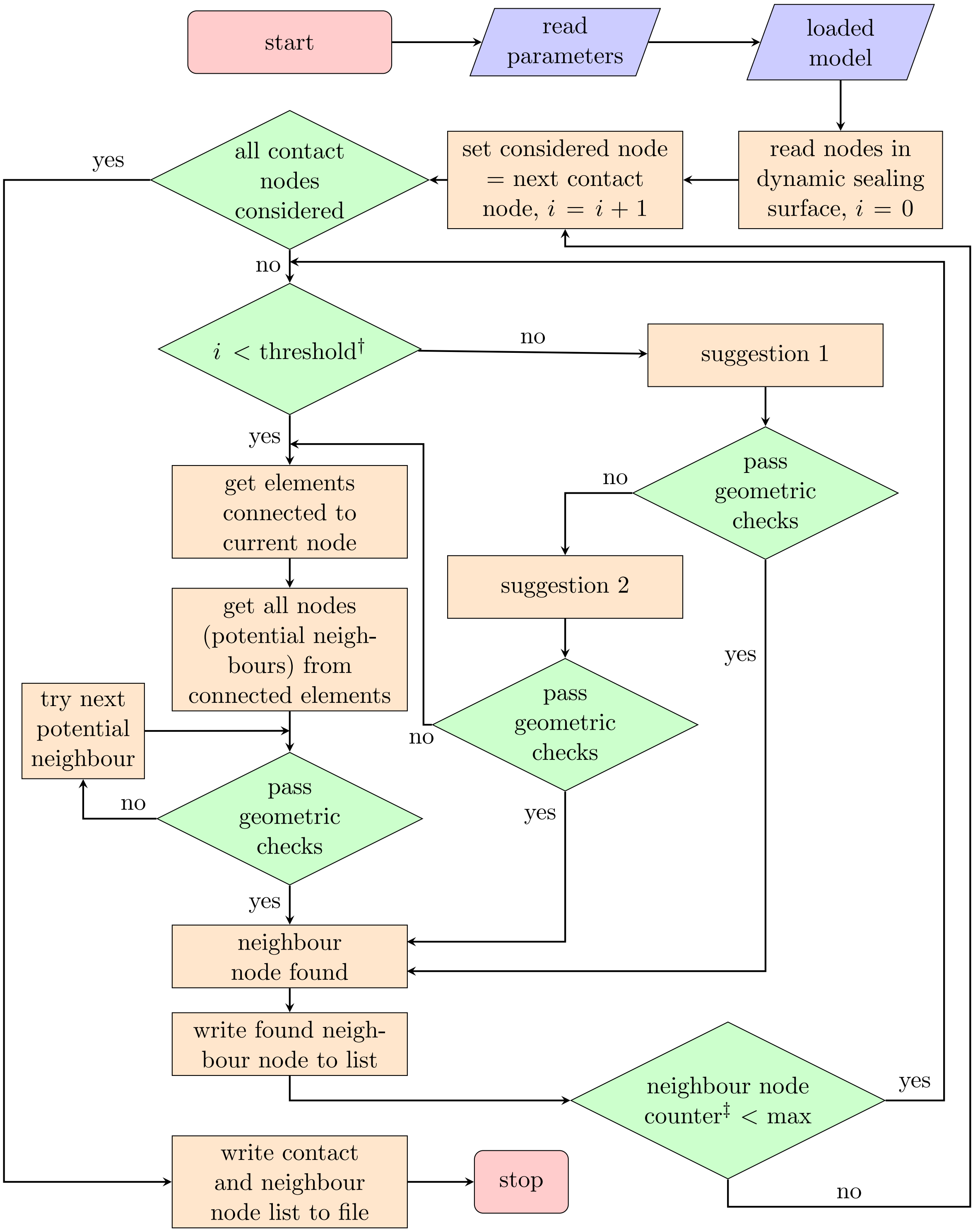


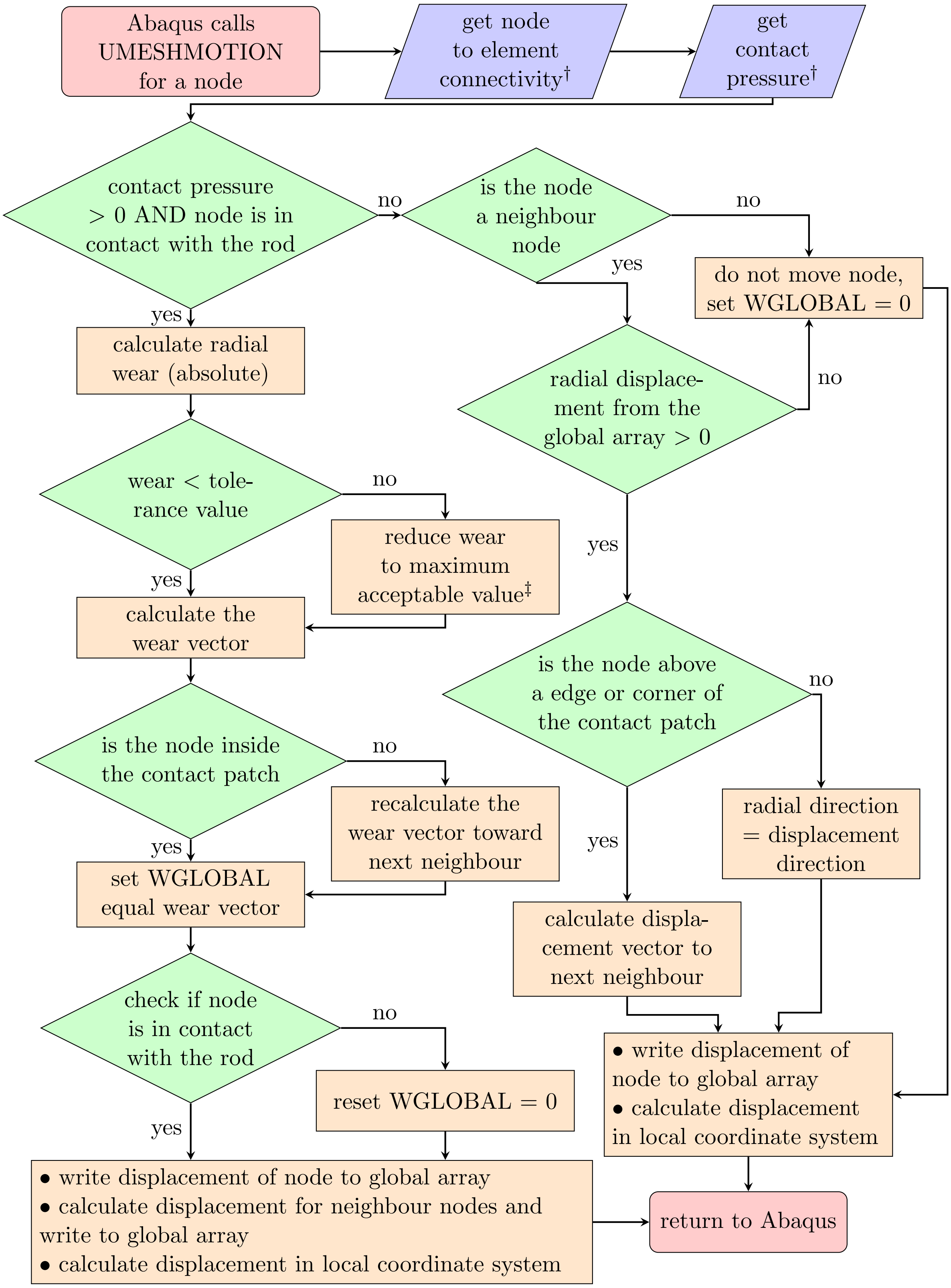
Figure 5.4. Flow chart of a wear calculation including the calculation of the pressure distribution in the dynamic sealing surface and setting up a wear-box [35].


start
read parameters
loaded model
read nodes in dynamic sealing surface, $i = 0$
set considered node = next contact node, $i = i + 1$
all contact nodes considered
yes
no
i < threshold†
no
suggestion 1
pass geometric checks
no
suggestion 2
yes
yes
get elements connected to current node
get all nodes (potential neighbours) from connected elements
try next potential neighbour
pass geometric checks
no
pass geometric checks
no
yes
yes
neighbour node found
write found neighbour node to list
neighbour node counter‡ < max
yes
no
write contact and neighbour node list to file
stop

† threshold is the number of runs required before suggesting

‡ each contact node has the same number of neighbour nodes in the whole wear-box

Figure 5.5. Flowchart of the algorithm to find all neighbour nodes in the wear-box. The counter i iterates over all nodes in the dynamic sealing surface.


Abaqus calls UMESHMOTION for a node
get node to element connectivity†
get contact pressure†
contact pressure > 0 AND node is in contact with the rod
no
is the node a neighbour node
no
yes
do not move node, set WGLOBAL = 0
yes
calculate radial wear (absolute)
radial displacement from the global array > 0
no
wear < tolerance value
no
reduce wear to maximum acceptable value‡
yes
yes
calculate the wear vector
is the node above a edge or corner of the contact patch
no
is the node inside the contact patch
no
recalculate the wear vector toward next neighbour
yes
radial direction = displacement direction
yes
set WGLOBAL equal wear vector
calculate displacement vector to next neighbour
check if node is in contact with the rod
no
• write displacement of node to global array
• calculate displacement in local coordinate system
reset WGLOBAL = 0
yes
• write displacement of node to global array
• calculate displacement for neighbour nodes and write to global array
• calculate displacement in local coordinate system
return to Abaqus

† provided by internal routines within UMESHMOTION

‡ so that the node does not leave the contact surface

Figure 5.6. Flow chart for the calculation of the nodal displacement (wear) with the user-defined subroutine UMESHMOTION. The subroutine is called for each node in the model in ascending order.

5.3 Numerical implementation of the gas pressure distribution in the dynamic sealing surface

The gas pressure drop across the dynamic sealing surface has significant impact on the ring wear. The gas pressure drop supports the structure against the outer pressure load. This reduces the solid-solid contact pressure and thus the wear, as described in chapter 3. The gas pressure distribution is modelled in an Abaqus sub-calculation before the wear calculation starts. The gas pressure distribution is updated periodically, since the dynamic sealing surface changes due to wear over time. The sub-calculation is derived from the loaded main model and consists of the dynamic sealing surface only. The gas pressure distribution in the axial sealing surface is not modelled, since no wear occurs here.
A Python script reads the nodes from the Abaqus output database file, which are in the dynamic sealing surface, and rebuilds the dynamic sealing surface with shell elements of the type DS4. Next the script applies the boundary conditions to node sets specified in the wear calculation. The boundary conditions are the high and low pressure values for the cylinder and crank side, respectively, see Figure 5.7.

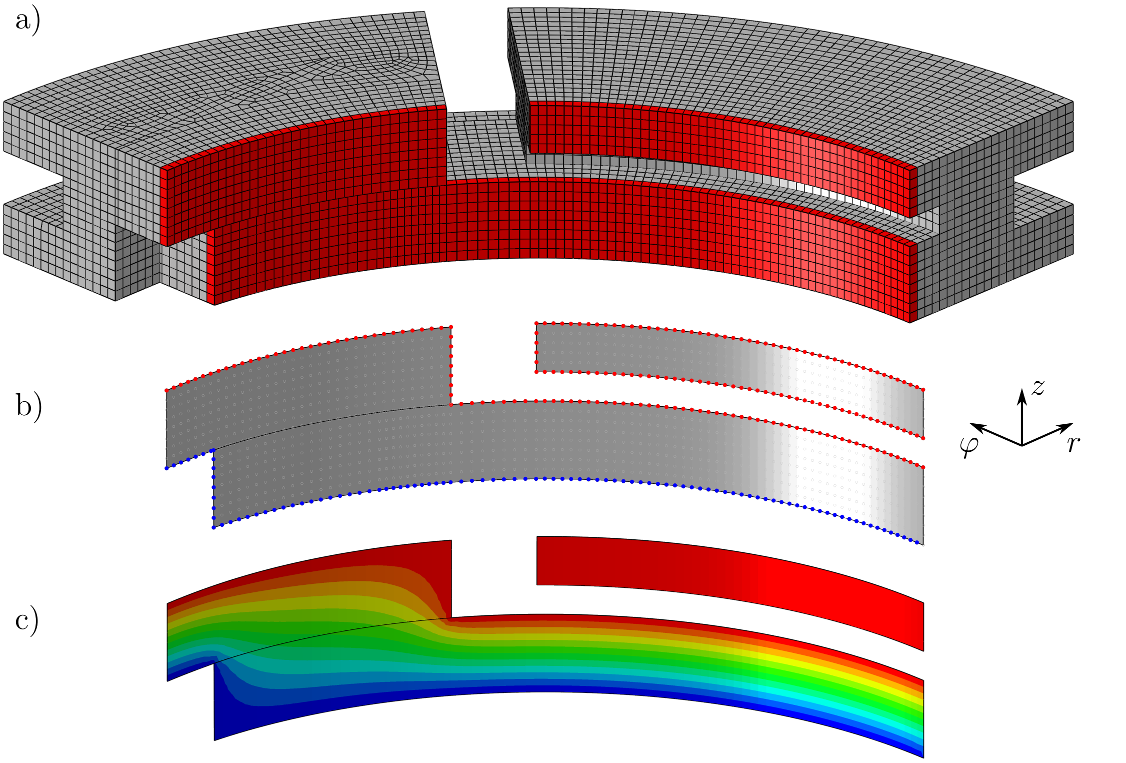


Figure 5.7. Pressure distribution calculation: a) 3D model of the loaded ring geometry viewed from cylinder side. The red elements are extracted and the nodes on the inner diameter are used to build the dynamic sealing surface model. b) Dynamic sealing surface model derived from a). On the highlighted nodes boundary conditions are applied: red - high value, blue - low value. c) Resulting (incompressible) gas pressure distribution.

The formal similarity of the gas pressure drop equation and the heat transfer equation is exploited, as explained in section 3.2. For the incompressible gas pressure distribution a steady-state heat transfer calculation is set-up. For the compressible gas pressure distribution the coefficient of thermal conductivity varies linearly with temperature.
The heat transfer analysis is implemented in the Abaqus software package for different element types (e.g. DS4). Calculating a temperature field with given boundary conditions (high and low temperature) is done quickly and returns a temperature distribution over the modelled dynamic sealing surface. UEXTERNALDB reads the temperature distribution and the DLOAD applies the nodal (temperature) values as nodal loads.

5.4 Analytical model

This section presents an analytical model, which allows to verify the developed wear algorithm and its numerical implementation. The content of this section and most of the figures are published in [36].
The following four independent aspects are considered when modelling wear:

- Ring geometry: The garter spring is ignored, since the force it applies is much smaller than the loading due to the gas pressure. For simplicity, the following discussion focuses on a simpler geometry: an uncut (i.e., single-piece) packing ring of rectangular cross section. Axial symmetry allows easier and quicker numerical modelling. It also makes the problem amenable to an analytical solution.
- Wear: according to Archard's equation (3.1).
- Gas pressure drop across the dynamic sealing surface, see section 3, is ignored in the analytical model. We assume perfectly smooth surfaces.
- Material behaviour: Since the primary focus is on verifying the numerical wear algorithm, we use the simplest possible material model, Hooke's law, disregarding any material ageing effects. Stress-strain measurements at 23 °C give the values for the elastic modulus $E = 1400\,\mathrm{MPa}$ and the Poisson's ratio $\nu = 0.4$.

5.4.1 Model

To make the problem amenable to an analytical solution, we consider a simplified configuration where a constant pressure p_o acts on the outer diameter $D^{(\mathrm{o})} = 2R^{(\mathrm{o})}$ of a linear-elastic axisymmetric ring of rectangular cross-section (axial width b, radial height h), i.e. the problem is axisymmetric in the r-z-plane. This pressure pushes the ring against a rod sliding at constant speed (Figure 5.8).
Friction is ignored, as are gas pressure effects over the contact surface. Furthermore, the ring is assumed to be in a state of plane strain in the r-φ-plane, so that stresses and

strains are functions only of the radial coordinate r. Under these assumptions, ring wear does not vary along the axial direction z.

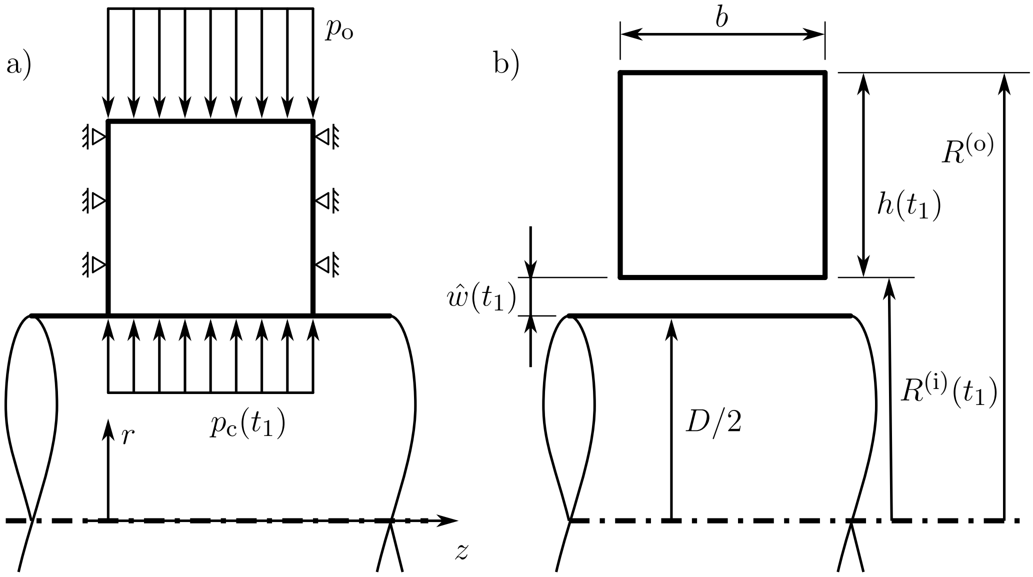


Figure 5.8. Worn axisymmetric ring at $t = t_1$. a) Loaded worn ring (pressure p_o acting on the outer diameter, contact pressure p_c acting on the inner diameter), assumed to be in state of plane strain. b) Dimensions of the worn ring in the undeformed state.

In the initial state $t = 0$, the inner diameter of the undeformed ring $D^{(\mathrm{i})} = 2R^{(\mathrm{i})}$ is equal to the rod diameter D. The pressure p_o gives rise to a contact pressure p_c which in turn leads to wear in accordance with equation (3.1). Due to wear, $R^{(\mathrm{i})}$ grows with time and so does the radial displacement $u^{(\mathrm{i})} = u(r = R^{(\mathrm{i})})$ required to satisfy the contact condition $D/2 = R^{(\mathrm{i})} + u^{(\mathrm{i})}$. Linear elasticity theory [87] provides the radial displacement field:

$$u(r) = \frac{r}{2G}\left(p_\mathrm{c}\frac{1 - 2\nu + \left(\frac{R^{(\mathrm{o})}}{r}\right)^2}{\left(\frac{R^{(\mathrm{o})}}{R^{(\mathrm{i})}}\right)^2 - 1} - p_\mathrm{o}\frac{1 - 2\nu + \left(\frac{R^{(\mathrm{i})}}{r}\right)^2}{1 - \left(\frac{R^{(\mathrm{i})}}{R^{(\mathrm{o})}}\right)^2} \right) \tag{5.1}$$

Evaluated at $r = R^{(\mathrm{i})}$, equation (5.1) gives the elastic radial displacement $u(\mathrm{i})$ of the ring's inner diameter when loaded by p_o and p_c on its outer and inner diameter, respectively. G denotes the shear modulus and ν is the Poisson's ratio.

At the first instant of time, the rod diameter is equal to the undeformed inner diameter of the ring, so that $u^{(\mathrm{i})}(t = 0) = 0$. Together with equation (5.1), this gives the initial contact pressure $p_\mathrm{c}(t = 0)$ and thus the initial wear rate $\frac{\mathrm{d}w}{\mathrm{d}t}(t = 0)$. However, the rate at which $R^{(\mathrm{i})}$ grows is not given by $\frac{\mathrm{d}w}{\mathrm{d}t}$, since equation (3.1) gives the rate at which *deformed* material is worn away. The following consideration shows how to derive the proper relationship between $\frac{\mathrm{d}R^{(\mathrm{i})}}{\mathrm{d}t}$ and $\frac{\mathrm{d}w}{\mathrm{d}t}$.

During a time increment $\mathrm{d}t$, an infinitesimal layer of material of thickness $\mathrm{d}w$ is worn away according to equation (3.1), this infinitesimal cylindrical shell being under stress. The thickness $\mathrm{d}\hat{w}$ of this infinitesimal shell in its undeformed state is related to $\mathrm{d}w$ via the radial strain ε_{rr} according to:

$$\hat{w} = \frac{\mathrm{d}w}{1 + \varepsilon_{rr}(r = R^{(\mathrm{i})})} \tag{5.2}$$

where the radial strain of the infinitesimal shell follows from equation (5.1) according to:

$$\varepsilon_{rr}(r = R^{(\mathrm{i})}) = \frac{\partial u}{\partial r}\Big|_{r=R^{(\mathrm{i})}} \tag{5.3}$$

Noting that the incremental change $\mathrm{d}R^{(\mathrm{i})}$ of the inner radius is equal to $\mathrm{d}\hat{w}$ gives:

$$\frac{\mathrm{d}R^{(\mathrm{i})}}{\mathrm{d}t} = \frac{k\nu p_{\mathrm{c}}}{1 + \frac{\partial u}{\partial r}\Big|_{r=R^{(\mathrm{i})}}} \tag{5.4}$$

which, with when we express p_{c} in terms of $R^{(\mathrm{i})}$ via equation (5.1),

$$p_{\mathrm{c}} = \frac{G\Big(\frac{D}{R^{(\mathrm{i})}} - 2\Big) + p_{\mathrm{o}}\frac{2-(1-\nu)}{1-\Big(\frac{1}{\rho}\Big)^2}}{\frac{1-2\nu+\rho^2}{\rho^2-1}} \tag{5.5}$$

and the derivation of $\frac{\partial u}{\partial r}$,

$$\frac{\partial u(r)}{\partial r} = \frac{1}{2G}\left(p_{\mathrm{c}}\frac{1-2\nu-\left(\frac{R^{(\mathrm{o})}}{r}\right)^2}{\rho^2-1} - p_{\mathrm{o}}\frac{1-2\nu-\left(\frac{R^{(\mathrm{i})}}{r}\right)^2}{1-\left(\frac{1}{\rho}\right)^2}\right) \tag{5.6}$$

finally leads to the non-linear first-order differential equation:

$$\frac{\mathrm{d}R^{(\mathrm{i})}}{\mathrm{d}t} = \frac{\frac{k\nu(\rho^2-1)}{1-2\nu+\rho^2}\left(G\left(\frac{D}{R^{(\mathrm{i})}}-2\right) + p_{\mathrm{o}}\frac{2(1-\nu)}{1-(\frac{1}{\rho})^2}\right)}{1 + \frac{1}{2G}\left(\frac{\left(G\left(\frac{D}{R^{(\mathrm{i})}}-2\right) + p_{\mathrm{o}}\frac{2(1-\nu)}{1-(\frac{1}{\rho})^2}(1-2\nu-\rho^2)\right)}{1-2\nu+\rho^2} + p_{\mathrm{o}}\frac{2\nu}{1-(\frac{1}{\rho})^2}\right)} \tag{5.7}$$

Equation (5.7) governs how the inner radius grows with time due to wear, ρ denoting the ratio $\frac{R^{(\mathrm{o})}}{R^{(\mathrm{i})}}$. It can be solved incrementally by means of an Euler forward scheme.

5.4.2 Numerical implementation and analytical results

The numerical implementation of the Euler forward scheme for the analytical model was done within the commercially available software Matlab. The flowchart Figure 5.9 shows the calculation steps as well as the parameters, which are chosen for the comparison with the finite element models. Using the Matlab code gives the accumulated wear in the deformed state w and the contact pressure p_c over time.

The ring is pressed against the rod during the wear process, if the pressure p_o is set to zero after the ring is already worn, a gap between the ring and the rod can be noticed. This gap $(R^{(\mathrm{i})} - D/2)$ can be evaluated as well and turns out to be *smaller* than the originally worn distance w. This can be seen in Figure 5.10 and Figure 5.11.

The numerical implementation of the analytical model enables to test quickly the effect of different geometry parameters on the wear process. Exemplary two cases are presented. Case 1 is a case where the ring stiffness is high enough that a wear limit is reached. Case 2 is the case where the ring wears fully away. The parameters for those cases are in Table 5.1, where the parameters equal for both cases are written only for case 1. The cases differ only in the radial dimension of the rings (case 1: 8 mm, case 2: 5 mm).

Table 5.1. Exemplary cases for ring wear.

parameter	case 1	case 2
$R^{(\mathrm{i})}$	28.58 mm	
$R^{(\mathrm{o})}$	36.58 mm	33.58 mm
D	28.58 mm	
b	8 mm	
G	500 MPa	
ν	0.4	
p_o	10 MPa	
kv	2×10^{-7} mm^3/Ns	

In case 1 the wear rate of the ring decreases over time proportional to the decreasing contact pressure. The wear stops after approximately 3×10^6 s. The accumulated wear in the deformed state will be referred to as *ultimate wear limit* w_∞. An ultimate wear limit can only be reached if the contact pressure drops to zero and there is still some radial ring height left.

In case 2 no ultimate wear limit is reached, the ring wears away fully. Although the pressure drops to a minimum value (less than 5 % of the initial value), it never reaches zero. The ring wears further, with a low wear rate and constantly loses stiffness. At a certain point the loss in stiffness gives rise to an increasing contact pressure. With the contact pressure the wear rate increases and the ring wears faster until it is worn away.

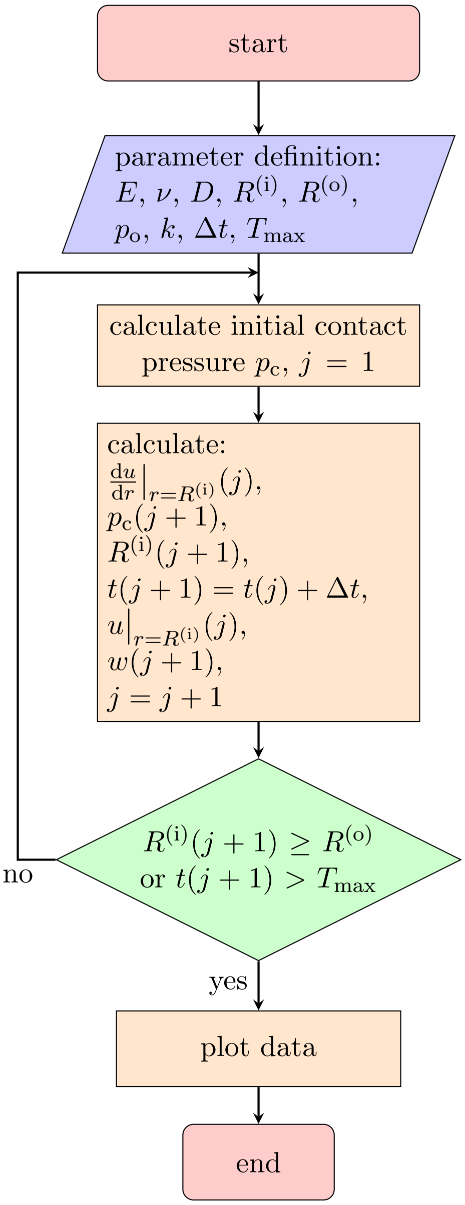

start
parameter definition:
E, ν, D, $R^{(\mathrm{i})}$, $R^{(\mathrm{o})}$,
p_{o}, k, Δt, $T_{\max}$
calculate initial contact pressure p_{c}, $j = 1$
calculate:
$\frac{\mathrm{d}u}{\mathrm{d}r}\big|_{r=R^{(\mathrm{i})}}(j)$,
$p_{\mathrm{c}}(j+1)$,
$R^{(\mathrm{i})}(j+1)$,
$t(j+1) = t(j) + \Delta t$,
$u\big|_{r=R^{(\mathrm{i})}}(j)$,
$w(j+1)$,
$j = j + 1$
$R^{(\mathrm{i})}(j+1) \geq R^{(\mathrm{o})}$
or $t(j+1) > T_{\max}$
no
yes
plot data
end

Figure 5.9. Numerical implementation of the analytical model to calculate the wear on the inner surface of a plane strain uncut ring.

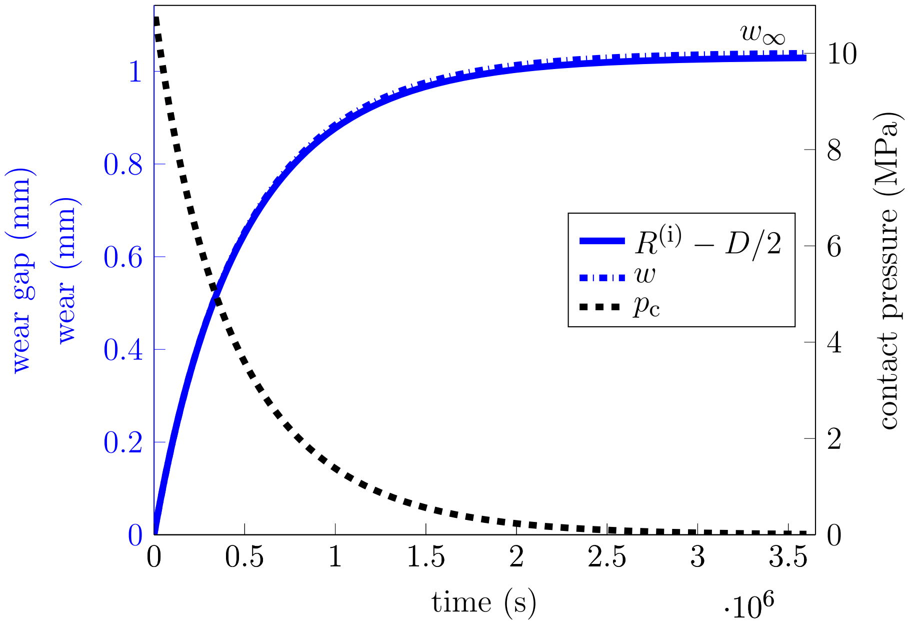


Figure 5.10. Case 1: Change of the wear gap ($R^{(\mathrm{i})} - D/2$), the accumulated wear w and the contact pressure p_c over time for a ring with $h_0 = 8\,\mathrm{mm}$ on a rod of diameter $D = 57.16\,\mathrm{mm}$. The ring wear stops before the ring is fully worn away.

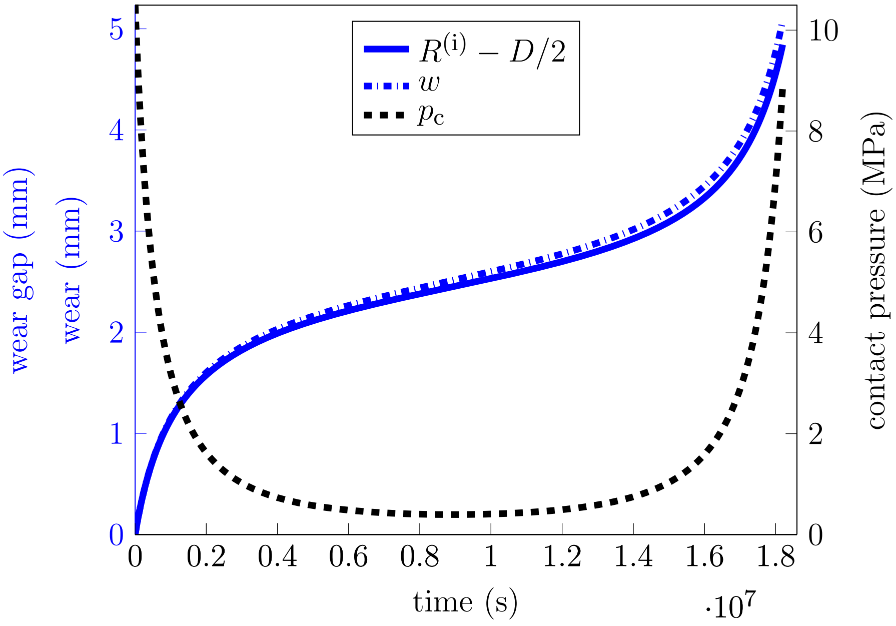


Figure 5.11. Case 2: Change of the wear gap ($R^{(\mathrm{i})} - D/2$), the accumulated wear w and the contact pressure p_c over time for a ring with $h_0 = 5\,\mathrm{mm}$ on a rod of diameter $D = 57.16\,\mathrm{mm}$. The ring wear continues until the ring is fully worn away.

Equation (5.7) was solved via numerical integration. For the parameters given in Table 5.1, Figure 5.10 shows how the accumulated wear $w(t) = \int_0^t \mathrm{d}w(\tau)/\mathrm{d}\tau \,\mathrm{d}\tau$, the wear gap $(R^{(\mathrm{i})} - D/2)$ in the undeformed state, and the contact pressure p_c change with time. For each value of the pressure loading p_o there is a certain value h_crit of the initial radial ring height $h_0 = h(t = 0)$ that determines whether the wear process comes to an end before the ring is fully worn away (Figure 5.10) $h_0 > h_\mathrm{crit}$ or continues until the ring is fully worn away (Figure 5.11) $h_0 < h_\mathrm{crit}$. Note that Figure 5.11 has a smaller initial ring height h_0h=5 mm, with $R^{(\mathrm{o})}$=33.58 mm, the other parameters are equal to those used in Figure 5.10.

The wear process comes to an end when either the contact pressure p_c becomes zero or when the ring has completely worn away. For the configuration under consideration where $R^{(\mathrm{i})}(t = 0) = D/2$, there is always a finite initial value $p_\mathrm{c}(t = 0)$ of the contact pressure so that the ring initially starts wearing away with $R^{(\mathrm{i})}$ increasing correspondingly.

From equation (5.1) the following relationship between the contact pressure p_c and the inner ring radius $R^{(\mathrm{i})}$ can be derived:

$$p_\mathrm{c} = \left(\left(\frac{D}{2} - R^{(\mathrm{i})} \right) \frac{2G}{R^{(\mathrm{i})}} + \frac{2p_\mathrm{o}(1-\nu)}{1 - \left(\frac{R^{(\mathrm{i})}}{R^{(\mathrm{o})}} \right)^2} \right) \frac{\left(\frac{R^{(\mathrm{o})}}{R^{(\mathrm{i})}} \right)^2 - 1}{1 - 2\nu + \left(\frac{R^{(\mathrm{o})}}{R^{(\mathrm{i})}} \right)^2} \qquad (5.8)$$

This relationship is depicted in Figure 5.12 for the geometrical and material parameters given in Table 5.1 case 1 for different values of p_o. Case A shows how the contact pressure p_c varies with $R^{(\mathrm{i})}$ for $p_\mathrm{o} = 30\,\mathrm{MPa}$. As can be seen, the contact pressure first decreases with increasing $R^{(\mathrm{i})}$ until the further loss in structural ring stiffness requires smaller and smaller pressure differentials $|p_\mathrm{o} - p_\mathrm{c}|$ to radially displace the inner ring diameter by the amount $R^{(\mathrm{i})} - D/2$. Since the contact pressure stays positive over the entire range $D/2 < R^{(\mathrm{i})} < R^{(\mathrm{o})}$, the ring fully wears away in case A. Applying a ring loading of $p_\mathrm{o} = 10\,\mathrm{MPa}$, which is the configuration considered for Figure 5.10, leads to case C, where the ring wear stops at $R^{(\mathrm{i})} = 29.6\,\mathrm{mm}$. These two regimes A and C are separated by the limiting case B ($p_\mathrm{o} = 21.2\,\mathrm{MPa}$) where the contact pressure only vanishes at a single value of $R^{(\mathrm{i})}$.

Figure 5.13 shows in dimensionless form how the ultimate accumulated wear $w_\infty = \lim\limits_{t\to\infty} w(t)$, referred to the initial ring height h_0, varies with dimensionless pressure loading p_o/G for rings of different aspect ratios h_0/D. The critical ring height h_crit for a certain pressure loading p_o/G is given by the highest value of h_0/D that leads to $w_\infty/h_0 = 1$. Values for h_0/D between 0.01 and 1 are evaluated.

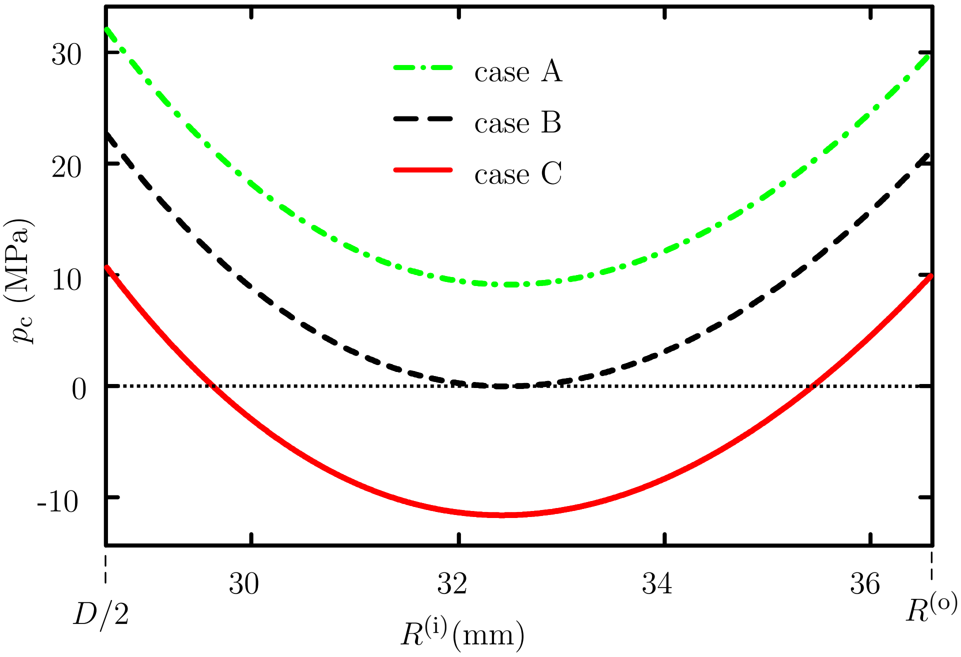


Figure 5.12. Variation of contact pressure p_c with inner radius over $D/2 < R^{(i)} < R^{(o)}$ for different pressure loadings p_o (case A: $p_o = 30\,\text{MPa}$, case B: $p_o = 21.2\,\text{MPa}$ and case C: $p_o = 10\,\text{MPa}$)

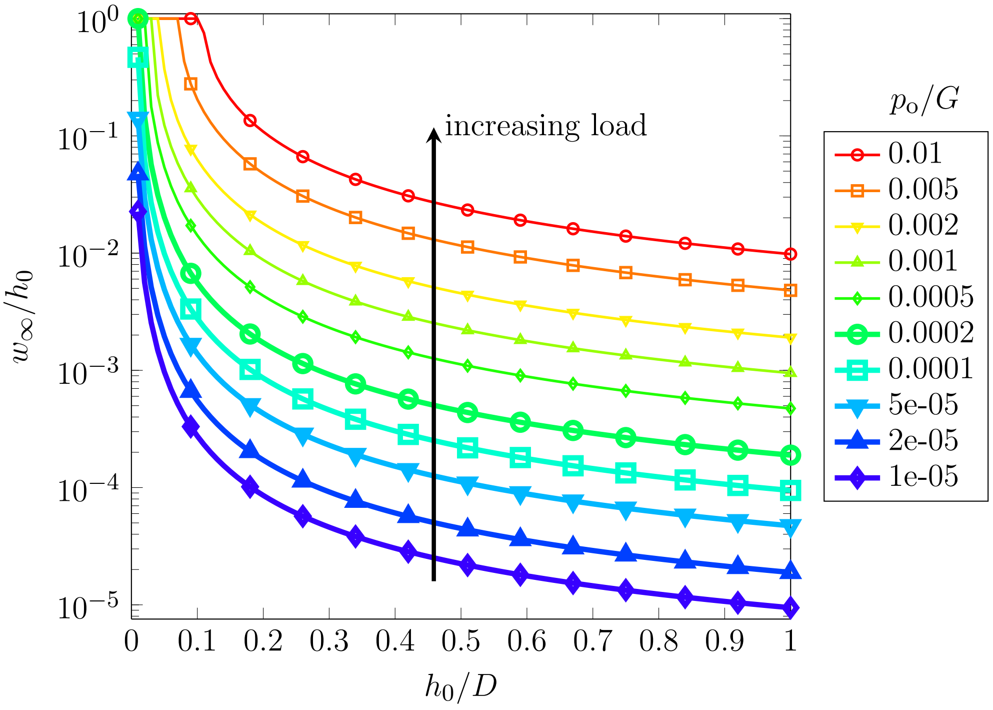


Figure 5.13. Ultimate wear w_∞ referred to initial ring height h_0 for rings of different aspect ratios h_0/D that are loaded by the non-dimensional pressure p_o/G.

5.4.3 Validation

Table 5.1, case 1 lists the parameters for the verification. To verify the user-defined wear algorithms developed in FORTRAN, the analytical model is now compared with two FE models created according to the clamping situation given in Figure 5.8. In both FE models the nodes on the axial surfaces of the rings are constrained such that they are fixed in axial direction.

a) A 2D axisymmetric FE model. The axisymmetric cross section is modelled with 50x50 CAX4 elements.The wear-box expands radially over 40 elements. Note that for validation purposes a single row of elements would be sufficient (due to the plane strain condition in r-φ). However, the same model has already been set up for some more general, non plane-strain cases.

b) A quarter symmetrical 3D FE model of an uncut single piece ring. The cross section is modelled with 16x11 C3D8 elements. The radial dimension of the outer 5 elements is twice the radial dimension of the inner 6 elements. The smaller, inner elements are within the wear-box.

Figure 5.14 shows the mesh of the ring cross section. The wear-box is highlighted red. Note: The size of the wear-box does not influence the result.

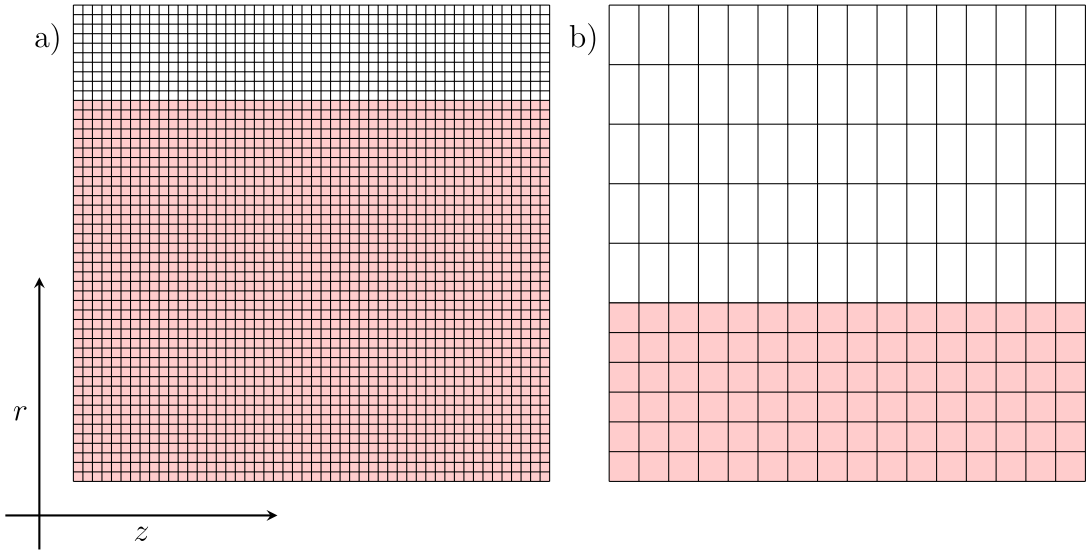


Figure 5.14. Cross section of the algorithm verification models: a) the 2D axialsymmetric model meshed with 50x50 CAX4 elements and b) the cross section of the 3D model meshed with C3D8 elements. The wear-box is shaded red.

Figure 5.15 shows how the contact pressure and the accumulated wear evolve over time. The data from the analytical model is drawn as solid line (as presented in Figure 5.10). The results of the numerical 2D and 3D FE calculation are marked with symbols. The 2D calculation and the analytical model are in excellent agreement. For the 3D calculation the agreement is good. A deviation from the analytic model can be seen only for the wear with progressing time. This may be due to the fact that the analytical model is formulated in geometrically linear theory, whereas the FE implementation takes geometric non-linearities into account.

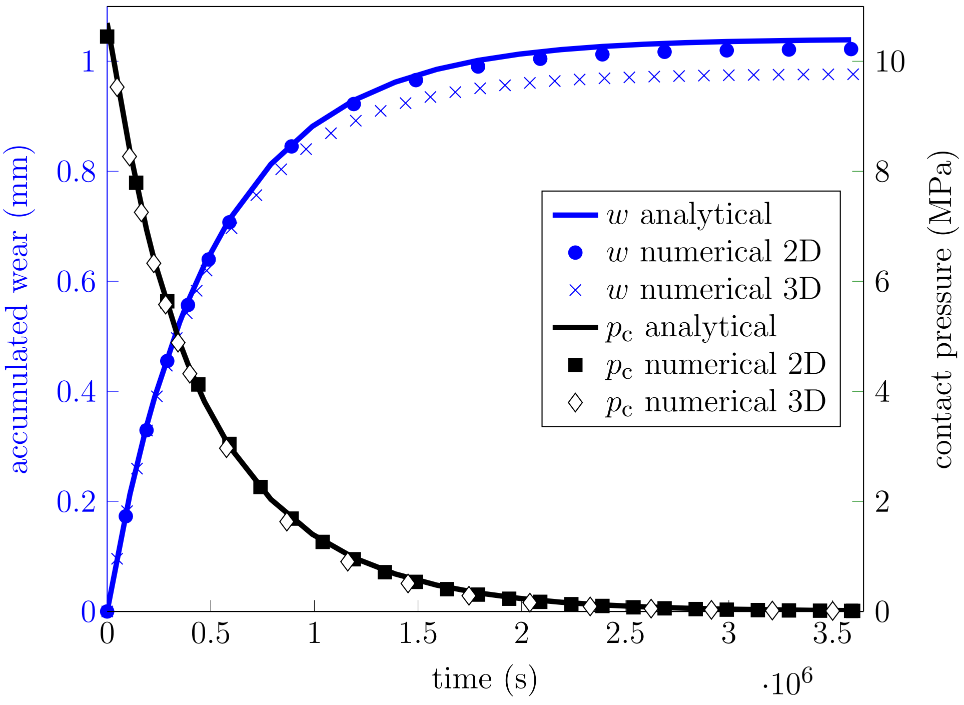


Figure 5.15. Verification of the wear algorithm of the numerical FE model against the analytical (numerically integrated) model for the wear in the deformed configuration w and the contact pressure p_c for an uncut ring.

5.5 Results

In this section the wear of BCD rings is investigated for different rod sizes and wear gaps. The main geometric features are given in Figure 2.7, with a detailed description in section 2.2. The fact that the geometry of the BCD rings are not scaling with the rod diameter was discussed as well, see section 4.2.2 and Figure 4.24, which already leads to interesting effects even when assuming linear elastic material behaviour.

5.5.1 Model set-up and parameter variation

The BCD ring is quarter-symmetrical. This symmetry is utilized to reduce the model size. The BCD geometry was adapted slightly to facilitate meshing and remove non-physical, numerically problematic stress peaks. The following measures were taken:

- The round garter spring groove on the outer diameter has been changed to rectangular shape, see Figure 5.16.
- The chamfer on the sealing segment's tips has been removed.
- Two edges have been trimmed on the cap segment, to remove non-physical stress peaks:
    - The edge between the surface, which contacts with the sealing segment, and the garter spring groove.
    - The edge between the surface, which contacts with the sealing element, and the surface of the outer diameter of the cap segment, see Figure 5.16.

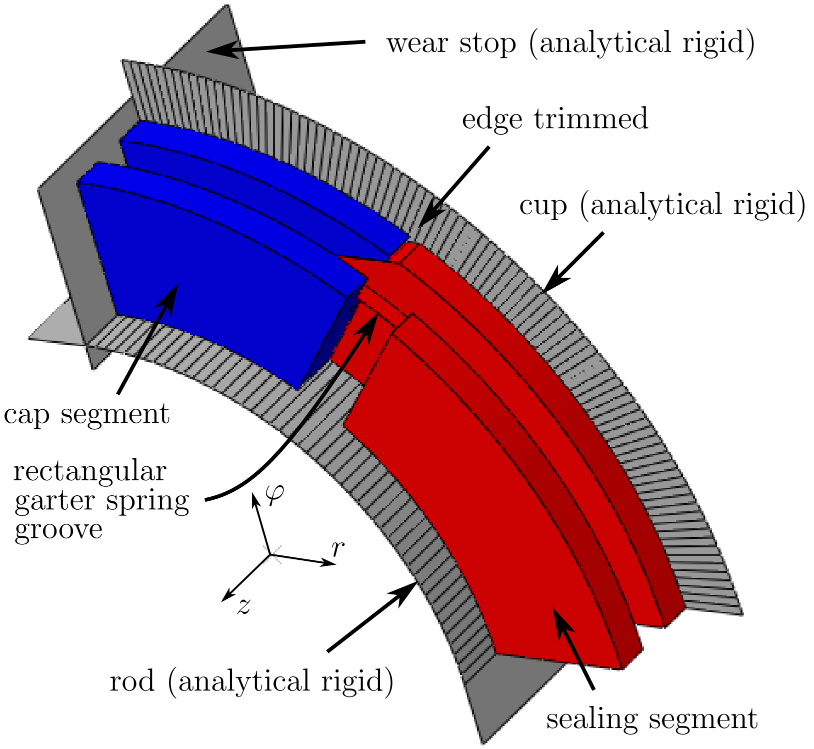


Figure 5.16. BCD ring model set-up for a 50 mm rod. Analytical rigid bodies are displayed gray, the sealing segment red and the cap segment blue. Edges of the cap segment are trimmed to avoid non-physical high contact pressures, which can lead to numerical instability. The garter spring groove is modelled as rectangular groove to allow meshing without tetrahedrons or distortion.

Relative motion of the sealing segment and the cap segment occurs due to wear. During the simulation the gas pressure is not applied to the newly created free surface, thus leading to non-physical locally high bending stresses, if the two forementioned edges jut

out.[1] Trimming the edges reduces this unintended problematic behaviour.
The garter spring is not modelled because its effect on the contact pressure (usually 0.03 MPa) is negligible compared to the contact pressure resulting from the gas pressure.

Figure 5.16 shows the quarter-model of a BCD for a 50 mm rod, the rigid bodies are shaded grey. Rod and cup are modelled as analytical-rigid parts, the radial clearance between rod and cup is 1 mm. Additionally another rigid part is used as wear stop for the sealing element. The wear stop is fixed in space and only contacts with the sealing segment after the ring has worn sufficiently. It is assumed that the BCD's quarter-symmetry is not broken while wearing. Figure 5.7 shows the meshed model.
The BCD geometry is manually partitioned to make it accessible to the Abaqus meshing algorithm, to avoid mesh distortion at the beginning (and through) the analysis as far as possible and to allow to use a wear-box. The mesh consists of C3D8 and wedge elements of the type C3D6. The wedges are used only outside the wear-box. Figure 5.17a shows the wear-box coloured yellow. The C3D8 elements inside have an edge length of approximately 0.5 mm. The wear-box is designed such that a maximum wear of 3 mm can be calculated. Figure 5.17b shows the developed view of the dynamic sealing surface of the virgin BCD the quarter model. The pressure balancing groove and the wear gap are marked.

[1] Abaqus offers a pressure penetration algorithm, which was not yet mature and could not be used to map the pressure physically correctly in the wear algorithm.

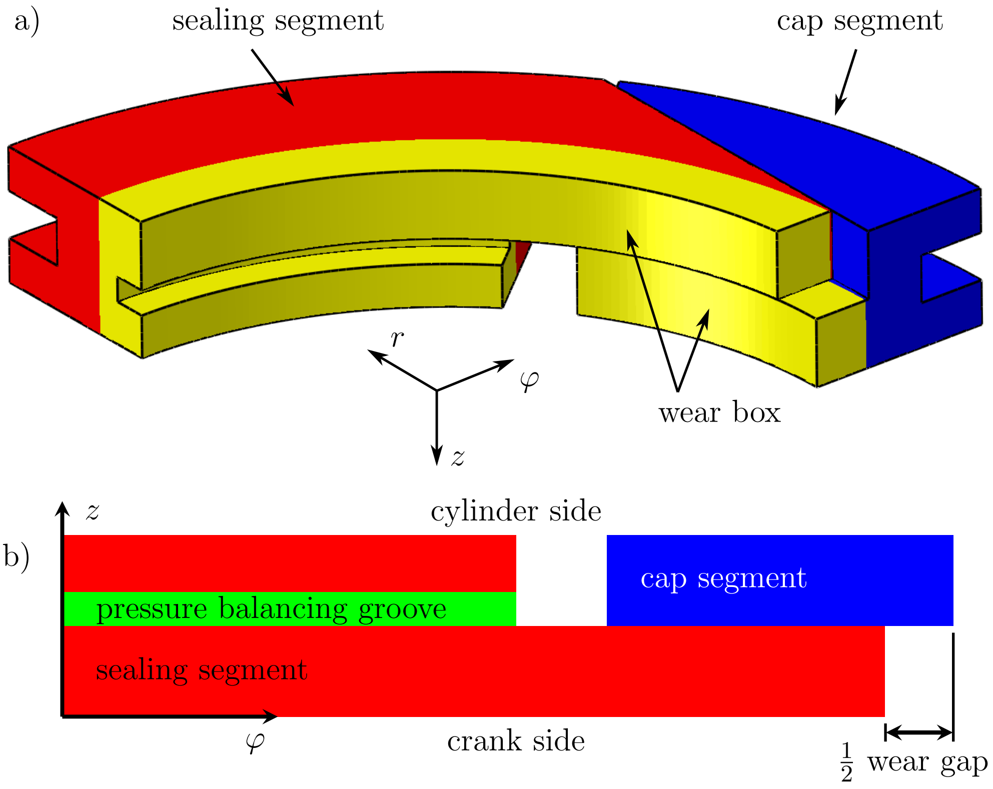


Figure 5.17. BCD ring for the 50 mm rod, quarter-model: a) The wear-box is coloured yellow for the sealing (red) and the cap (blue) segment. b) The developed view of the dynamic sealing surface with the view direction being the negative radial direction. The pressure balancing groove area is coloured green. On the lower right the clearance between the sealing segment and the wear stop, half the wear gap, is marked.

The wear simulation of the BCD rings allows to investigate the effect of the rod diameter, the pressure balancing groove and the size of the wear gap. The range of rod sizes is represented by three rod diameters: 28.58 mm, 50 mm and 100 mm. Not all ring dimensions scale with the rod diameter, cf. Figure 4.24. Adding the pressure balancing groove unloads the ring partially and thus reduces the contact pressure and the wear rate. For each rod size a model without a pressure balancing groove is investigated. The size of the wear gap is 6 mm in the current configuration of a standard BCD ring. The wear gap is varied in three steps of 6 mm, 4 mm and 2 mm for the configuration with the pressure balancing groove. The calculations of the BCD without the pressure balancing groove has a wear gap of 6 mm. Table 5.2 lists the 12 considered cases.

Table 5.2. Cases considered while screening the wear behaviour of the BCD ring.

case name	rod diameter	pressure balancing groove	wear gap
A1	28.58 mm	yes	6 mm
A2		yes	4 mm
A3		yes	2 mm
A4		no	6 mm
B1	50 mm	yes	6 mm
B2		yes	4 mm
B3		yes	2 mm
B4		no	6 mm
C1	100 mm	yes	6 mm
C2		yes	4 mm
C3		yes	2 mm
C4		no	6 mm

The linear elastic calculations are time independent. The material properties are represented by $E = 1400\,\mathrm{MPa}$ and $\nu = 0.4$. The gas pressure load is set to 10 MPa, which is very high. The gas is assumed incompressible. The rod velocity is $1\,\mathrm{m\,s^{-1}}$, the wear coefficient is $2.8 \times 10^{-7}\,\mathrm{mm^3\,m^{-1}\,MPa^{-1}}$.
Note: In these linear elastic calculations the point in time when a certain wear pattern occurs solely depends on the chosen wear coefficient. The resulting times can be used to compare different configurations, but for a lifetime prediction the coefficient of wear of an analysed tribo-system has to be known.
Here the maximum time increment is 600 s. Every 1.2×10^5 s (equals at least 200 wear increments) the gas pressure distribution in the dynamic sealing surface is evaluated and updated.

5.5.2 Evaluation procedure

After a wear calculation finishes post-processing gives the following results, which are explained in detail below for a BCD ring mounted on a 50 mm rod:

- the wear pattern at a certain point in time,
- the evolution of wear over time,
- the contact force over time and
- the closing of the wear gap over time.

The wear pattern of the whole dynamic sealing surface can be plotted as 3D surface. Below the 3D wear pattern a black patch represents the dynamic sealing surface. These patches also indicate the remaining wear gap. If a ring reaches the wear stop, the black patch of the sealing segment expands from 0 to 90°. Figure 5.18 represents the end of a wear calculation procedure, when a maximum wear of more than 3.2 mm is achieved and the wear-box is locally fully consumed. The wear stop has been reached and the wear gap is gone.

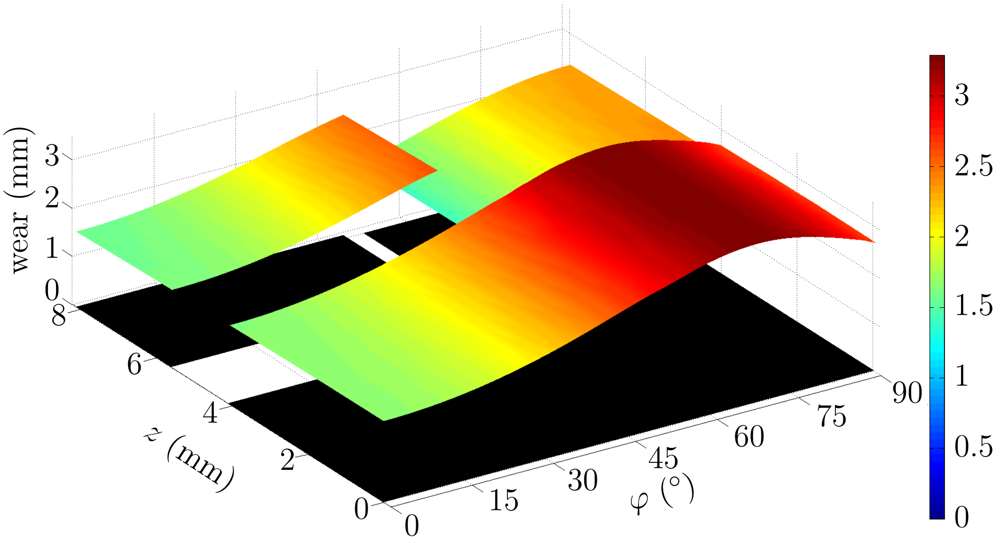


Figure 5.18. Wear pattern of a (linear elastic) wear calculation of BCD ring on a 50 mm rod after 23.6 d of continuous operation.

These surfaces can be produced for different points in time and give insight how the whole ring wears.

The wear history is presented in Figure 5.19a to f, where six wear patterns are depicted at different times. The time increment between the graphs is 6×10^5 sec (166.67 h). Between these points in time the pressure drop is updated 5 times. It is evident, that the axial wear gradient is small compared to the gradients in the circumferential direction. Thus the wear data of one of the three patches of the surface plot can be represented well by the data along a node path, z = const..

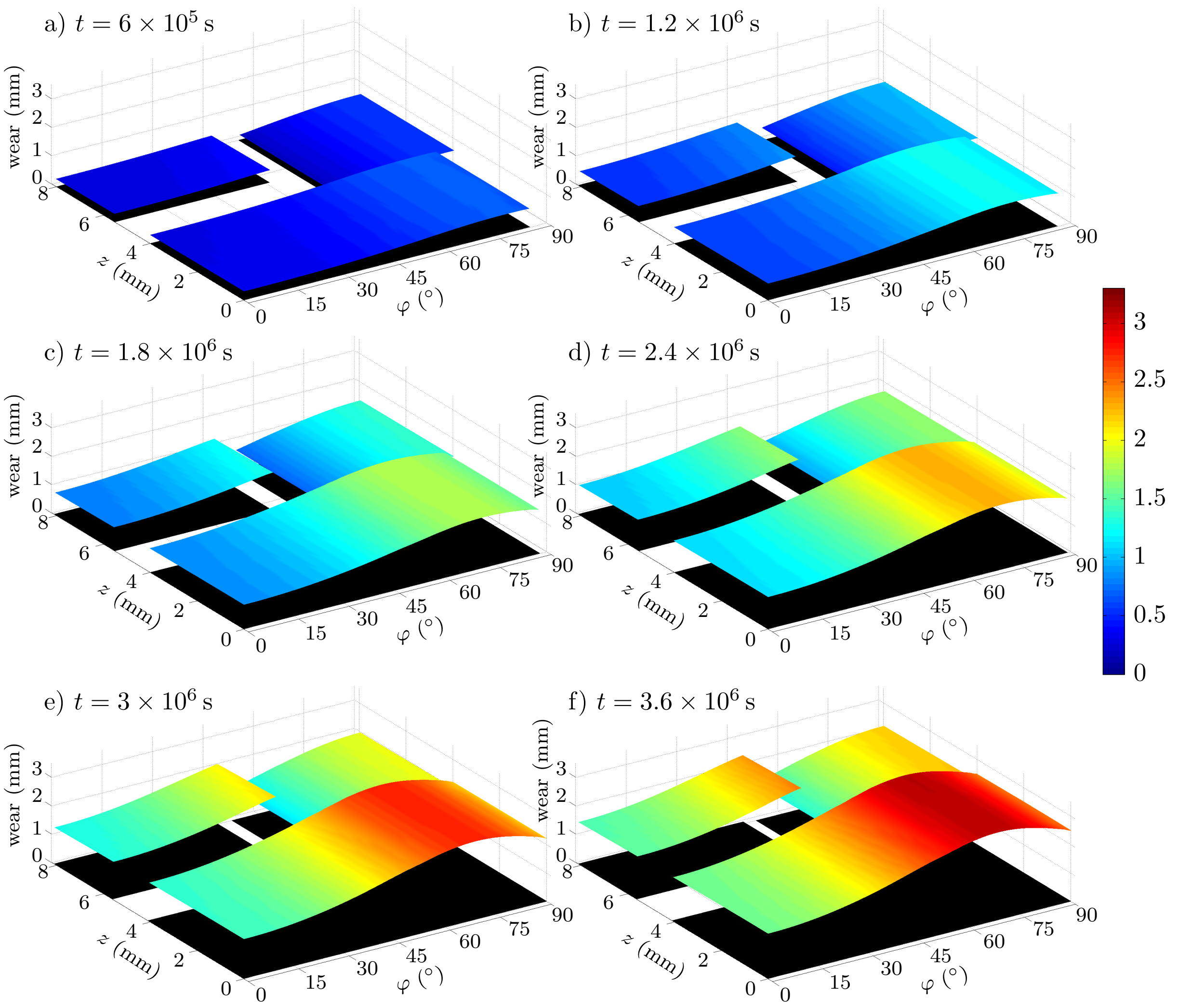


Figure 5.19. Wear pattern of the BCD ring quarter model on a 50 mm rod after 6 points in time. Where a) to f) marks advancing time from 6×10^5 s (166.67 h) to 3.6×10^6 s (1000 h).

Next the wear evolution for $z = 2$ mm is considered, starting at the middle of a sealing segment with $\varphi = 0°$ and running to the tip of the sealing segment (where φ_{end} changes over time due to the wear). Figure 5.20 gives a graphical explanation: a) The wear pattern at a certain point in time. The pink plane at $z = 2$ mm represents the position of the node path, which is depicted schematically in b). The cut between the pink plane and the 3D wear surface gives the wear of this node path, which can be tracked over time, see c).

This representation allows to compare with other configurations. A line is drawn from 6×10^4 s every 1.2×10^5 s, for easier visualisation and comparison every 10th line ($\Delta t = 1.2 \times 10^6$ s) is coloured red. Within c) there is a miniature sketch, similar to b), of the virgin-state dynamic sealing surface, where the position of the node path is drawn as a red line. The sketch also gives the size of the wear gap and shows the existence of the

pressure balancing groove. Additionally the rod diameter is given.
The wear gap closes as the sealing segment wears, this can be seen on the right end of the abscissa in Figure 5.20c. The distance between the wear contour line and the right side of the plot (90°) represents the size of the remaining wear gap.

The simulation also provides the contact force of every contacting node. The contact force for the calculated ring is obtained by integrating over all nodes in the dynamic sealing surface, without regard of the surface normal. This integrated contact force is displayed as a result. Note that the contact force for a whole ring is four times this value. The rod pull-off force or the frictional force of a ring is proportional to that contact force. The proportionality factor is μ, when assuming Coulomb friction.
Comparing the direction independent, sum of all nodal contact forces has a big advantage: The result is valid independent of the tribological system.

The closing of the wear gap can be monitored, using the Abaqus variable "COPEN". It gives the contact opening distance, the distance between two potentially contacting partners. The distance between a node on the surface of the sealing segment's tip and the rigid stop is evaluated. Here the node, which is in contact with the rod and at φ_{end} and $z = 2\,\text{mm}$ is chosen, see Figure 5.21. Note that the real wear gap is twice as big as the calculated (and plotted) distance from the node to the wear stop.

Figure 5.22 shows the closing of the wear gap over time, considering the distance of the tip of one sealing segment to the wear stop. Further it shows the development of the contact force of the quarter model over time. When the wear gap closes (0 mm gap) the ring's structural stiffness increases instantly and significantly. The ring then nearly behaves like an uncut, solid ring. This is the reason for the discontinuity in the contact force curve. The contact force decreases faster after the closing event because then the ring's structural stiffness takes on more and more of the gas pressure load.

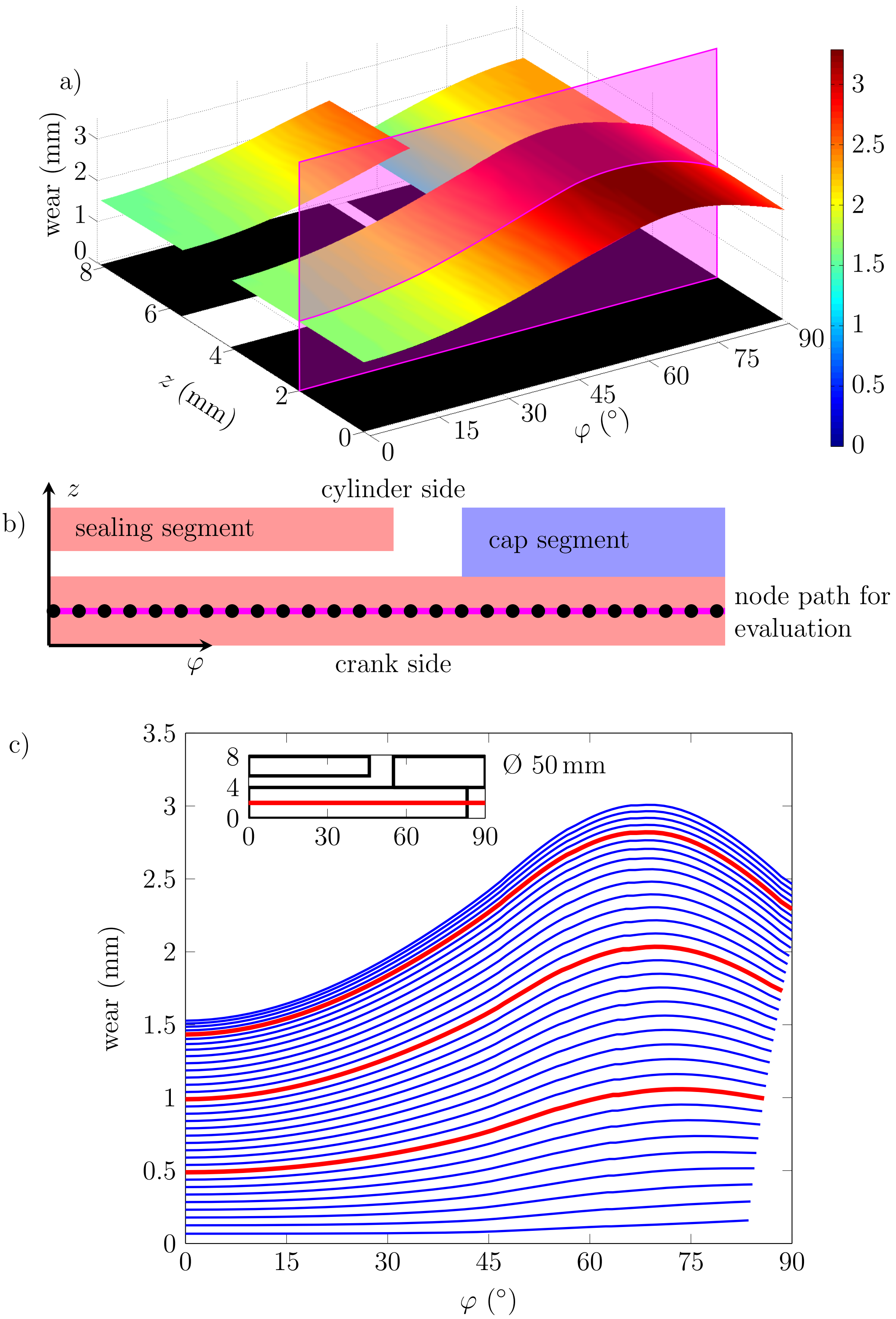


Figure 5.20. a) Wear pattern of a linear elastic wear calculation of BCD ring quarter model on a 50 mm rod after 23.6 d. The pink plane represents the position of a node path in tangential direction 2 mm from the cup face, schematically shown in b). c) presents the evolution of the wear of the picked node path. Blue lines are drawn for time increments of $\Delta t = 1.2 \times 10^6$ s. The red lines indicate every 10th line.

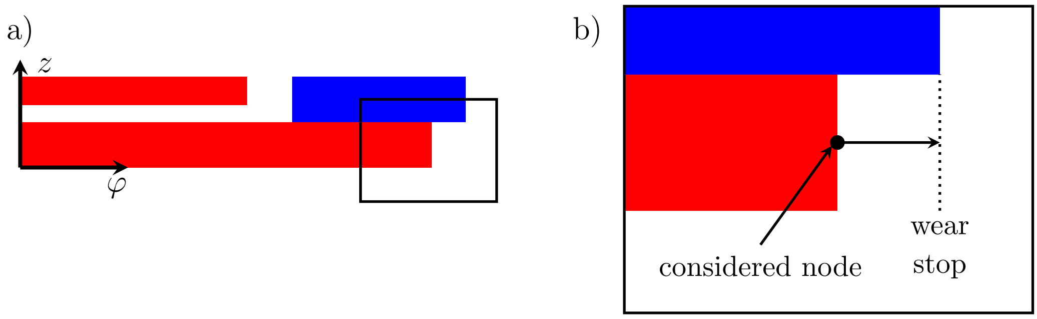


Figure 5.21. Position of the node, which is used to measure the wear gap. a) shows the dynamic sealing surface, the sealing segment's contact area is coloured red and the cap's area blue. In b) the black framed detail of a) is magnified. The wear stop is indicated by a dotted line. The distance from the considered node to the wear stop is monitored over time.

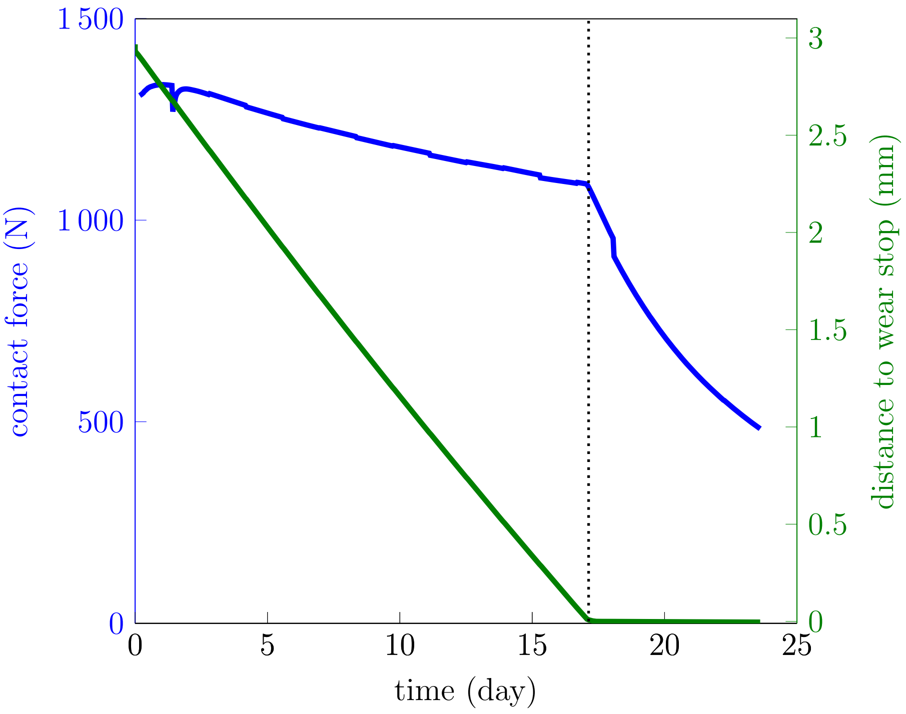


Figure 5.22. The evolution of the contact force (blue) and the wear gap (green) are shown over time. When the wear gap closes (0 mm gap) after 17.1 days, the contact force shows a discontinuity. After gap closing the contact force decreases faster because from then on the ring has a higher structural stiffness in circumferential direction which allows it to carry a greater part of the external pressure load.

5.5.3 Calculations with incompressible gas

The wear pattern and then the contact force and the wear gap are analysed in the following for the variations given in Table 5.2. The plots comparing these configurations are arranged equally for each rod diameter as shown in Table 5.3.

Table 5.3. Plot arrangement in the consecutive figures. The symbol ’*’ represents a wildcard character to be substituted by the letters A, B or C. The number in the case name can be cross-referenced to Table 5.2. Case *4 is placed below case *1 which allows to better see the effects of the pressure balancing groove.

case name	*1	case name	*2
configuration	standard	configuration	modified
pressure balancing groove	yes	pressure balancing groove	yes
wear gap	6 mm	wear gap	4 mm
case name	*4	case name	*3
configuration	modified	configuration	modified
pressure balancing groove	no	pressure balancing groove	yes
wear gap	6 mm	wear gap	2 mm

Figure 5.23 shows that the wear pattern of the 28.58 mm rod reaches an ultimate wear limit. The structure is stiff enough to carry the whole gas pressure load, which gives high lifetime. This limit is reached earlier in time and the total amount of wear is lower if the wear gap is smaller, i.e. the wear values are arranged in the following order A3<A2<A1. Interestingly the sealing segments of case A1 and A2 do not stop wearing at $\varphi = 0°$. The effect of the pressure balancing groove is visible when comparing wear patterns at equal times. For example the second red line of A4 starts at 1.59 mm at 0° and has a maximum of 1.84 mm at 58°. The equivalent line for A1 starts at 1.35 mm at 0° and has a maximum of 1.79 mm at 59°. The form of the wear pattern also differs. The ring without pressure balancing groove has a more uniform pattern than the ring with the pressure balancing groove.

The ring without pressure balancing groove (A4) shows the highest total wear not at approximately 60° but at 0°. This is unique. The rings from cases A1 and A2 show similar trends, but not that pronounced. Their wear pattern evens out when reaching their saturation pattern. This indicates a change in structural stiffness that shifts the loading situation such that the tips of the sealing segment are unloaded.

Figure 5.24 shows the evolution of the contact force and the closing of the wear gap for the 28.58 mm rod. Any discontinuities in the curves of the contact force are either due to the update of the gas pressure distribution in the dynamic sealing surface or the load rearrangement after the wear gap closes. The proof that an ultimate wear level has been reached for the cases A2 and A3 is found in the contact force. In these cases the contact force nearly drops to zero. (Note: No information can be given if the rings would still be sealing.) The closing speed of the wear gap is not constant over time. It slows down with increasing time. This is more pronounced when the wear gap is bigger. The closing speed with pressure balancing groove is 15 % lower than without pressure balancing groove. The contact force of the ring with pressure balancing groove is 15 % lower than without pressure balancing groove at the beginning of the calculation. This accordance is used as a sanity check: Lower contact pressure results in slower wear.

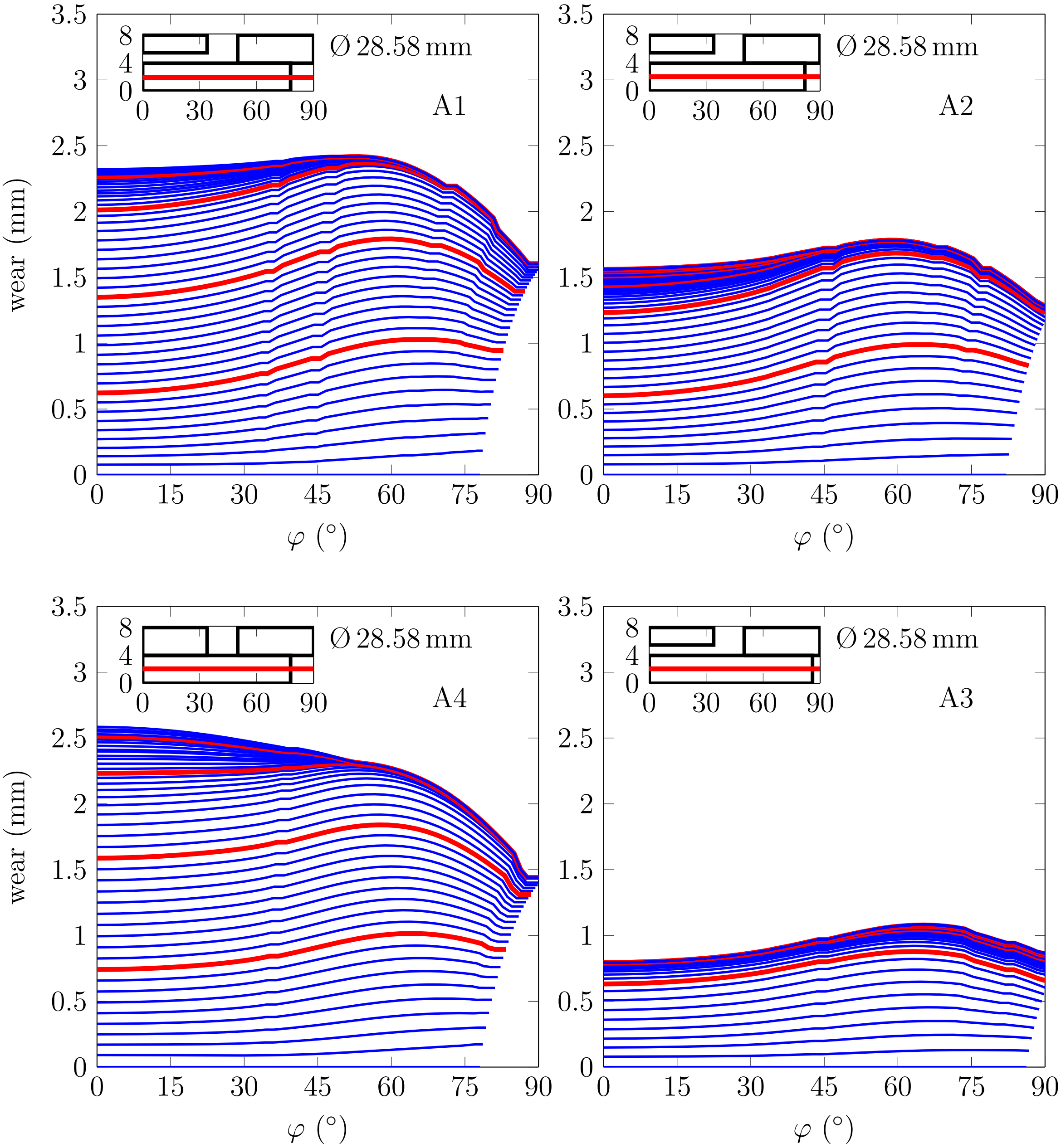


Figure 5.23. Wear pattern evolution. The tangential node path at $z = 2\,\mathrm{mm}$. The rod diameter = 28.58 mm. The case names are A1, A2, A3 and A4 given clockwise starting at the upper left plot.

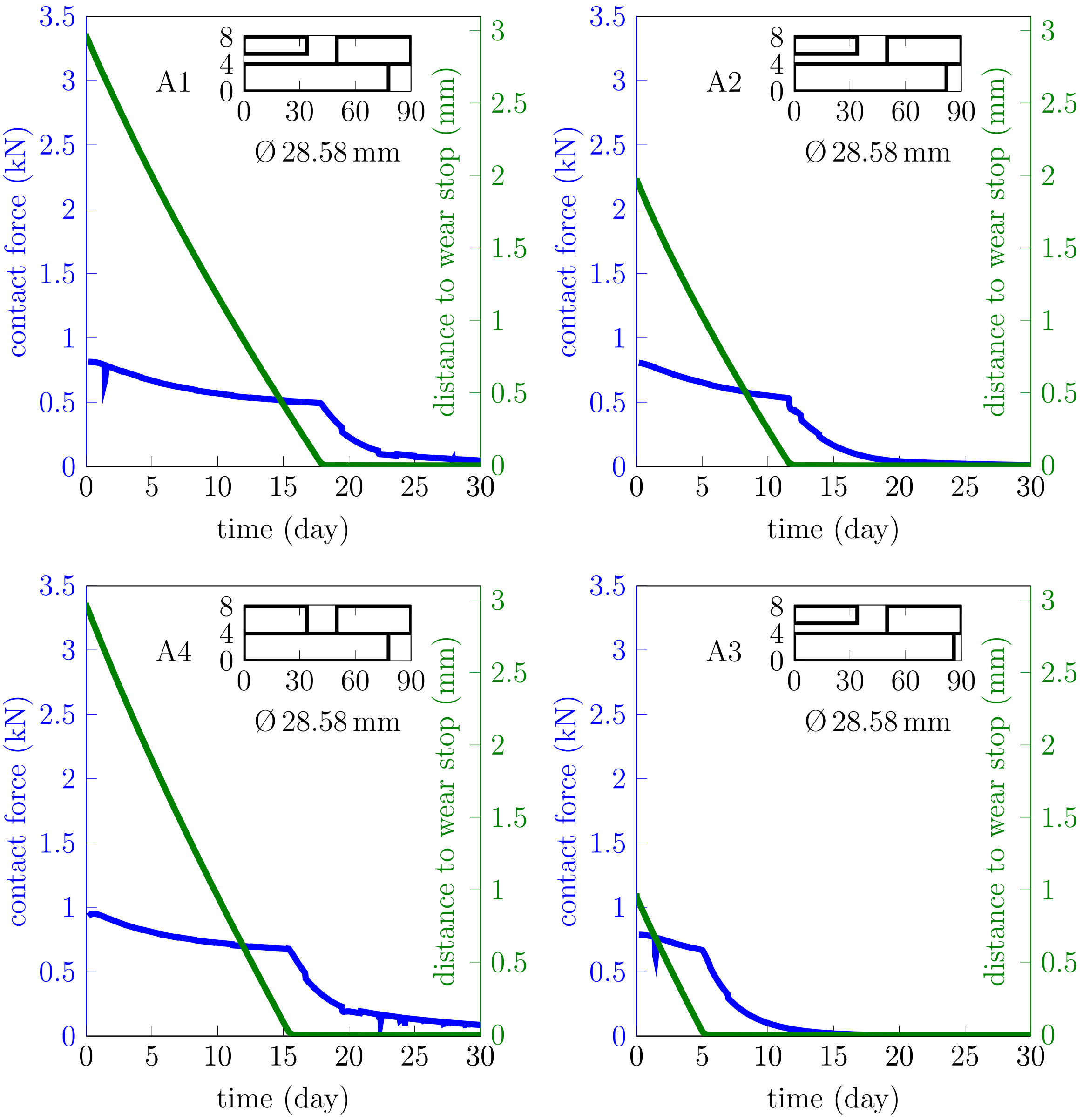


Figure 5.24. Evolution of the integral contact force (blue) and the closing of the wear gap (green) for the four investigated configurations A1 to A4, which are placed clockwise starting at the top left, respectively. Rod diameter 28.58 mm.

Figure 5.25 shows the wear patterns of BCD calculation for the 50 mm rod. Locally the ring wears more than 3 mm, if the wear gap is 6 mm. This is independent of the pressure balancing groove. In these cases the wear-box is fully consumed locally. Cases B2 and B3, with 4 and 2 mm wear gap, respectively, reach an ultimate wear level in the central part of the sealing segment from $\varphi = 0$ to approximately 30°. The tips of the sealing segment continue wearing. In the cases B1 and B4 the the wear rate drops towards the end of the calculation.

The effect of the pressure balancing groove is similar to the BCDs on the smaller rod.

Without the groove the ring wears more uniformly. The second red line of B4 starts at 1.52 mm at 0° and has a maximum of 2.11 mm at 67°. The equivalent line for B1 starts at 0.99 mm at 0° and has a maximum of 2.03 mm at 70°. The shape of the wear pattern also differs.

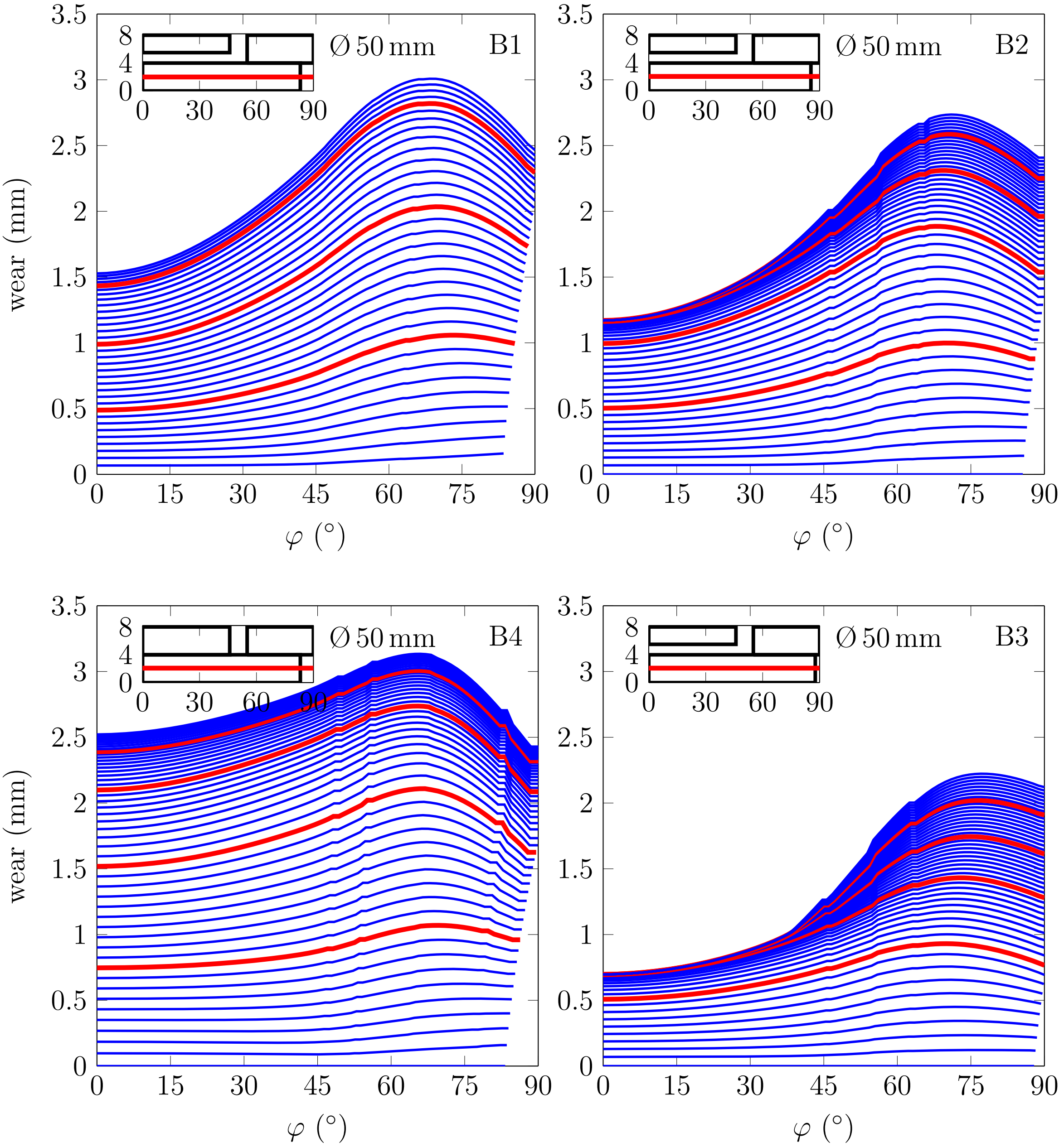


Figure 5.25. Wear pattern evolution. The tangential node path is at $z = 2$ mm. The rod diameter = 50 mm. The case names are B1 to B4 given clockwise starting at the top left plot.

Figure 5.26 presents the wear gap size and the contact force over time of the quarter model of the 50 mm rod. The discontinuity after 1.4 days results from the update of the gas pressure distribution in the dynamic sealing surface. The calculation of A1 stops after 23.6 days, because the wear-box is fully consumed. As mentioned, a quasi steady-state is reached for B2 and B3, when these rings reach an ultimate wear level. Then the contact force reaches a nearly constant level after the wear gap has closed and the structure has worn to some extent.

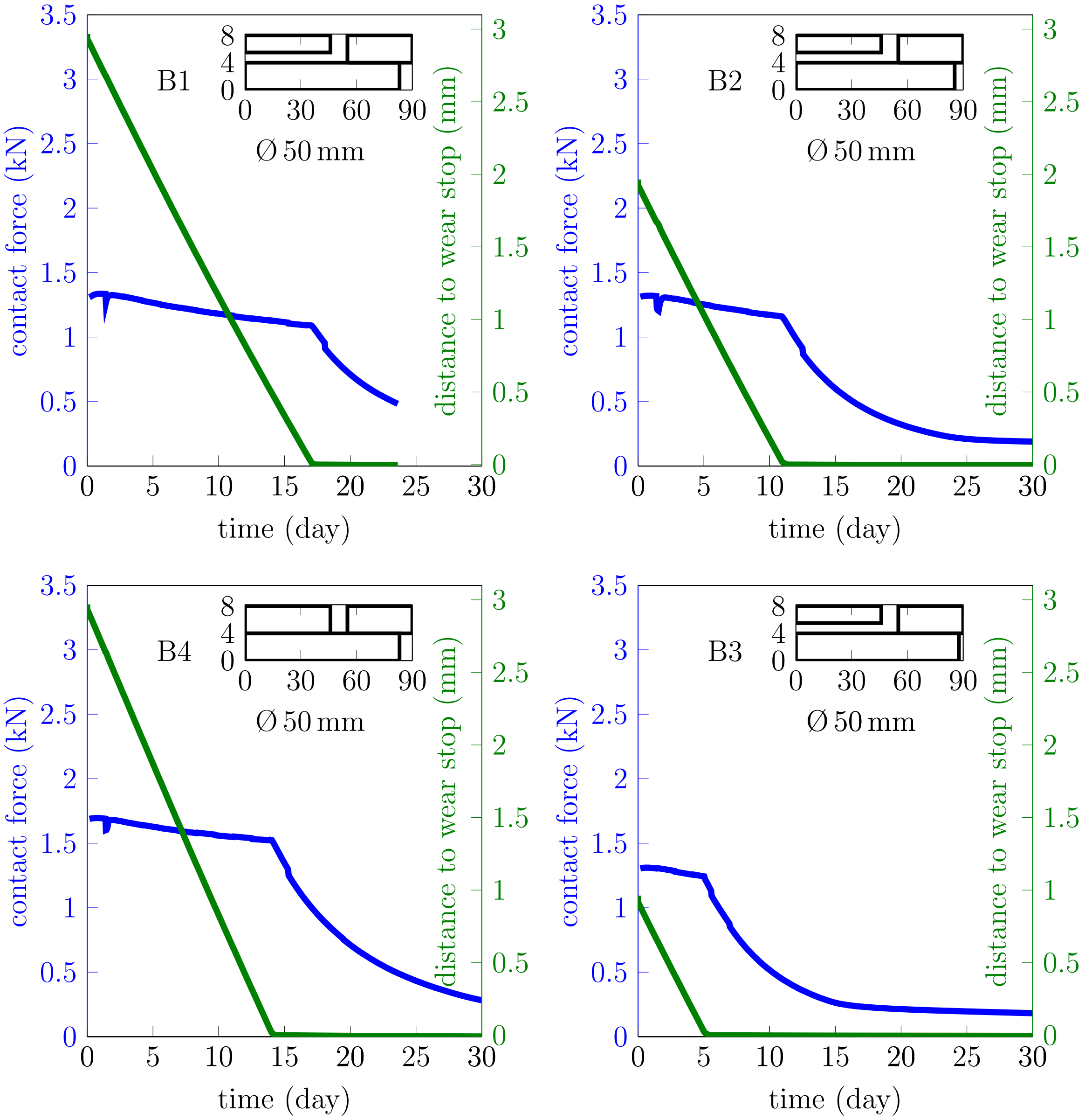


Figure 5.26. Evolution of the integral contact force of the BCD quarter-model and the rod (blue) and the closing of the wear gap (green) for the four investigated BCD configurations B1 to B4. Rod diameter 50 mm.

The closing speed of the wear gap is nearly constant over time. The closing speed with pressure balancing groove is 20 % slower than with pressure balancing groove. The contact force of the ring without the pressure balancing groove is 23 % lower than without pressure balancing groove at the beginning of the calculation.

The BCD rings on the 100 mm rod wear until the wear-box is consumed locally. Figure 5.27 shows that only B3, with 2 mm wear gap shows signs of reaching an ultimate wear level at $\varphi = 0°$.

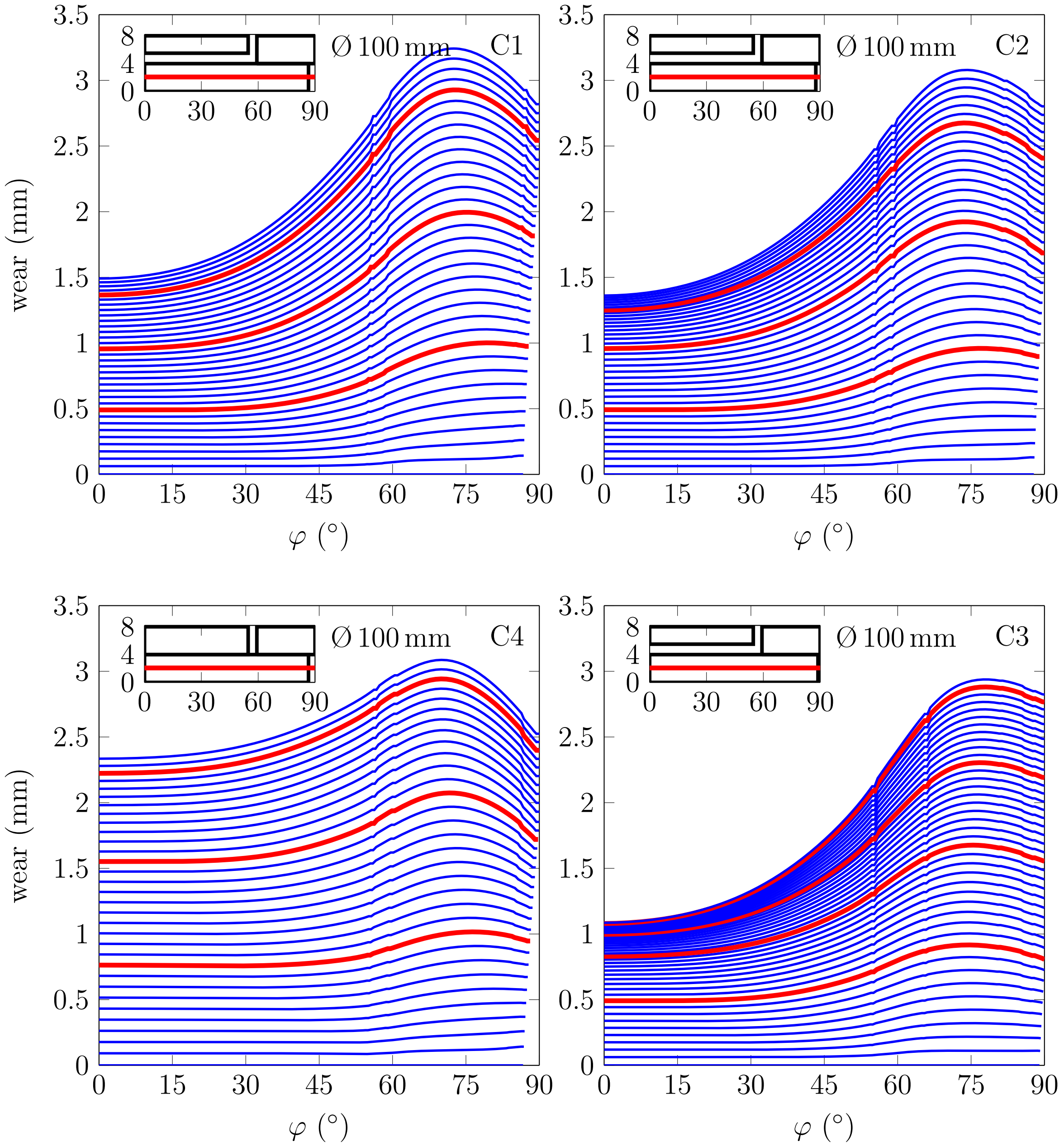


Figure 5.27. Wear pattern evolution. The tangential node path is at $z = 2$ mm. The rod diameter = 100 mm. The case names are C1 to C4 given clockwise starting at the top left plot.

Figure 5.28 shows that all rings were worn to the limits of the wear-box within 30 days of wear time. The closing speed of the wear gap is nearly constant over time. The closing speed with the pressure balancing groove is 30 % slower than without pressure balancing groove. The contact force of the ring with pressure balancing groove is 28 % lower than without pressure balancing groove at the beginning of the calculation.

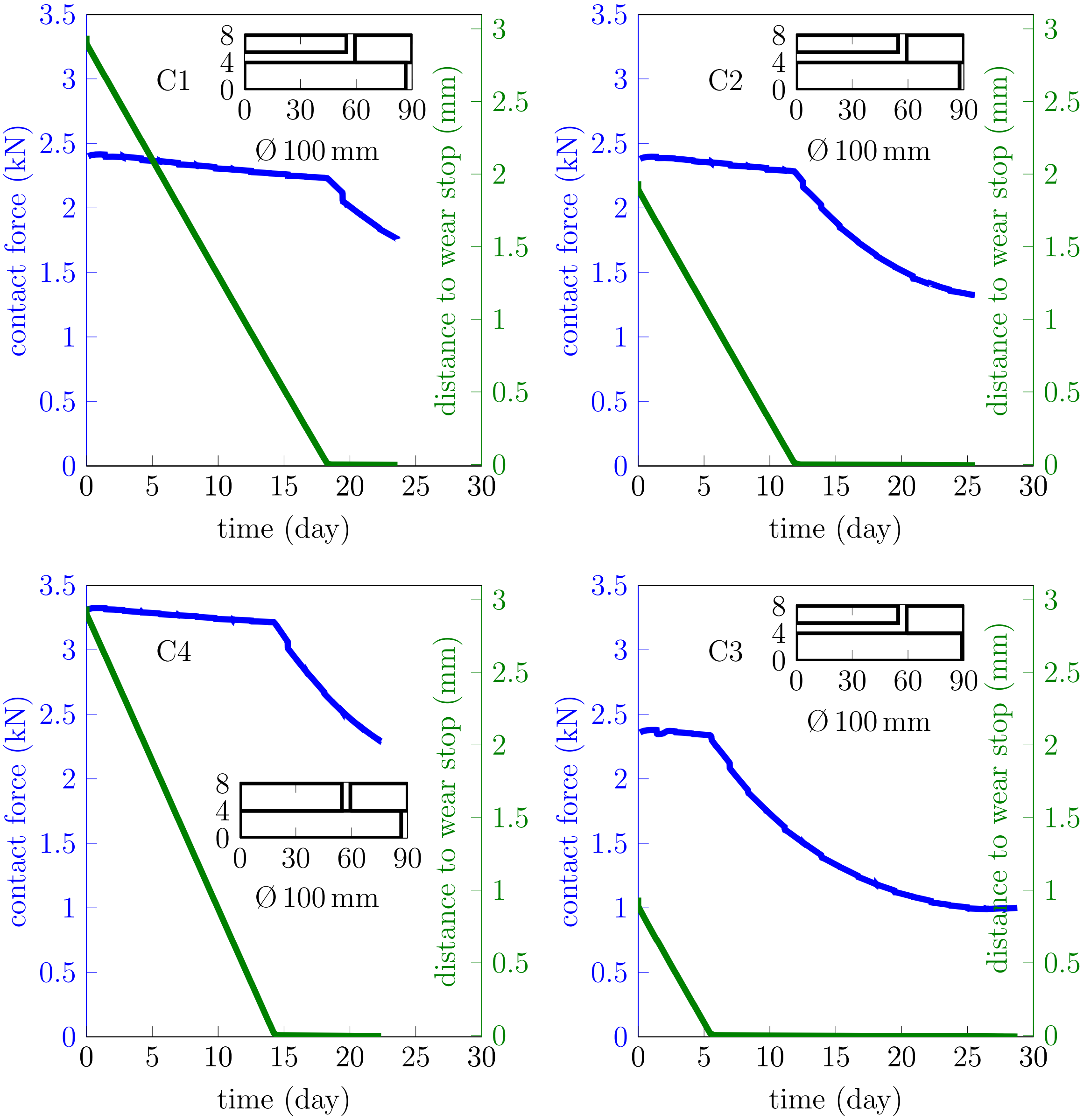


Figure 5.28. Evolution of the integral contact force of the BCD quarter-model and the rod (blue) and the closing of the wear gap (green) for the four investigated BCD configurations C1 to C4. The rod diameter = 100 mm.

Comparison of different diameters

A comparison of different diameters shows that with decreasing initial wear gap the position of the maximum wear $\varphi_{\max}$ shifts towards 90°. If the wear gap is smaller, the sealing segments are longer. Thus the cap segment is supported over a greater length and the position of maximum wear shifts. Table 5.4 presents the maximum position $\varphi_{\max}$ of wear found in the plots of Figure 5.23, 5.25 and 5.27. For case A4 from Figure 5.23 two values are given, since the wear maximum shifts from 50° to 0° when the wear gap is closed and the structure wears further. A significant increase between the small and medium diameter occurs. The difference between the medium and big rod are smaller. This is mainly because the BCD does not scale with the rod diameter: For small rods the cap segment seals a bigger circumferential section than for a bigger rod, see Figure 4.24.

Table 5.4. Circumferential position $\varphi_{\max}$ of the maximum wear. For A4 two values are given, since the maximum wear at the calculation's end occurs at 0°. For comparability the maximum position of the 3rd red line is given. Case *4 has no pressure balancing groove. The symbol '*' represents a wildcard character to be substituted by the letters A, B or C.

case name	wear gap (mm)	circumferential position of $\varphi_{\max}$ (°) for the rod diameter (case)		
		28.58 mm (A)	50 mm (B)	100 mm (C)
*3	2	65.3	77.0	77.5
*2	4	58.3	71.0	74.3
*1	6	53.5	69.6	72.5
*4	6	(0), 50	66.0	70.0

Figure 5.29 compares how the wear gap closes for all variations. The time of the wear gap closing depends on the existance of the pressure balancing groove and the initial wear gap size, but not significantly on the rod size. The closing speed is constant for the biggest rod size, but for the small and medium rod it decreases over time. This effect is more pronounced for the small rod. Table 5.5 gives the times when the wear gap closes for all 12 compared calculations. Increasing the wear gap from 2 to 4 to 6 mm increases the time until the wear gap closes roughly in the same proportion, e.g. for the 100 mm rod from 6 to 12 to 19 days.

The pressure balancing groove increases the time until the wear gap closes for increasing rod diameters by 15, 20 and 30 %, respectively. This is in accordance to the increase of the contact force. Table 5.6 gives the contact force of the loaded virgin rings. Since the contact area between ring and rod increases linearly with the diameter, the contact forces grow roughly with the rod diameter.

A bigger wear gap results in slightly higher contact forces. The reason is that the dynamic sealing surface becomes smaller and thus some of the support of the gas pressure distribution is lost as the wear gap increases.

The effect of the pressure balancing groove is much more pronounced than changing the

size of the wear gap. As the rod diameter increases the groove becomes circumferentially longer and has a higher impact. Thus the introduction of a pressure balancing groove decreases the contact force by 14, 22.5 and 27.5 % for the small, medium and large rod, respectively.
The decrease of the contact force over time corresponds to the closing speed of the wear gap. On the small rod the contact force decreases faster at the beginning. This effect is slightly visible on the medium rod, but cannot be seen at the big rod.

Table 5.5. Time in days until the wear gap is closed, for all investigated variations. Case *4 has no pressure balancing groove.

case name	wear gap (mm)	wear gap closing time (days) for the rod diameter (case)		
		28.58 mm (A)	50 mm (B)	100 mm (C)
*3	2	5.4	5.6	5.9
*2	4	12.0	11.5	12.3
*1	6	18.1	17.3	18.7
*4	6	15.7	14.4	14.4

Table 5.6. Contact force of the ring variations. Case *4 has no pressure balancing groove.

case name	wear gap (mm)	contact force of virgin ring (N) for the rod diameter (case)		
		28.58 mm (A)	50 mm (B)	100 mm (C)
*3	2	787	1304	2359
*2	4	807	1312	2380
*1	6	815	1309	2400
*4	6	930	1690	3313

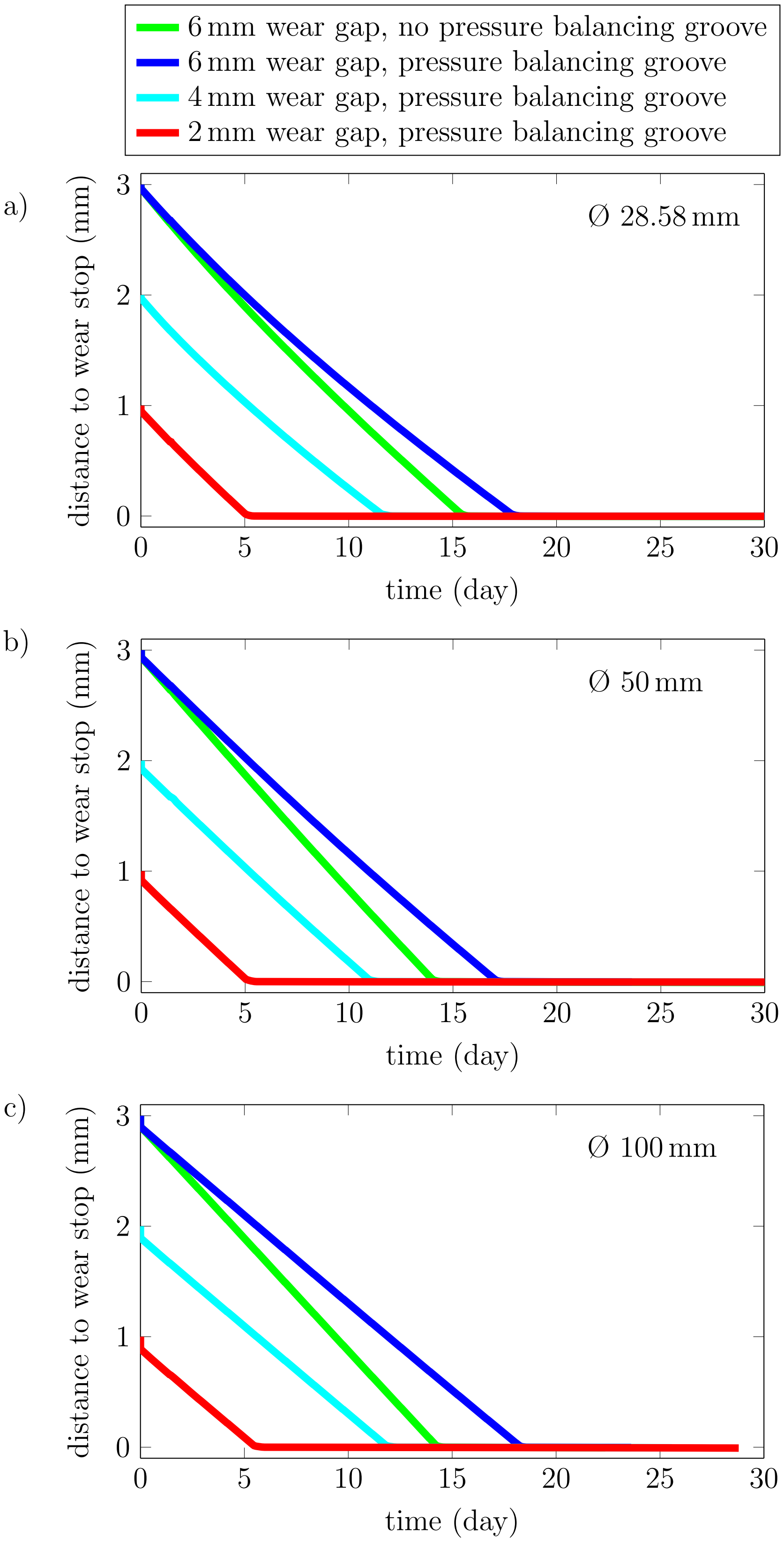


Figure 5.29. Evolution of the wear gap for the four quarter-models and the three rod diameters: a) 28.58 mm, b) 50 mm and c) 100 mm.

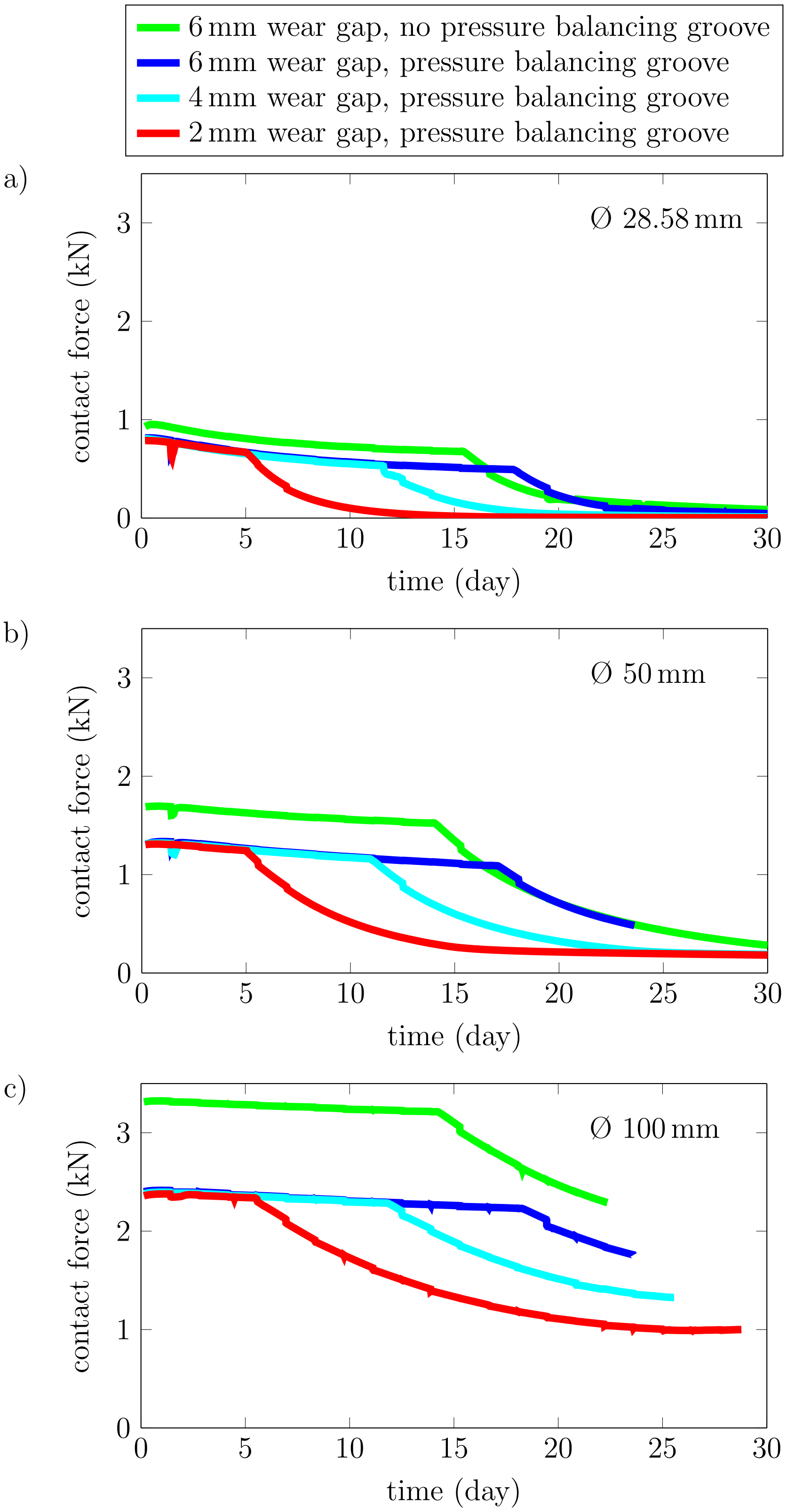


Figure 5.30. Evolution of the integral direction independent contact force over time for the four quarter-models on the three rod diameters a) 28.58 mm, b) 50 mm and c) 100 mm.

5.5.4 Calculations with compressible gas

This section presents the effect of the gas compressibility on the frictional force and the wear of the BCD ring mounted on a 50.8 mm (2") rod. The compressibility effects of the gas in the dynamic sealing surface is becoming more pronounced when the pressure differential becomes larger and the absolute pressure on the crank side smaller.

A series of numerical simulations investigates the wear pattern dependent on the gas pressure. The material is assumed to be linear elastic with $E = 1400\,\mathrm{MPa}, \nu = 0.4$. The gas pressure load varies between 11 and 101 bar in steps of 10 bar. The pressure on the crank side is ambient pressure (1 bar). Additionally a calculation with 6 bar is performed for the comparison with the experiments. Each calculation works with a quarter model using the available symmetry boundary conditions. The gas pressure drop in the dynamic sealing surface is calculated for each gas pressure level with and without taking into account compressibility effects. Figure 5.31 shows the axial gas pressure drop in the dynamic sealing surface at $\varphi = 10°$ and $\varphi = 70°$ for different pressure levels.

The calculations are evaluated according to section 5.5.2. The integral contact pressure is calculated i) after loading the structure, thus without the gas pressure distribution in the dynamic sealing surface and ii) after adding the gas pressure in the dynamic sealing surface and starting the wear algorithm. For the latter case Figure 5.32 shows the contact force over time assuming a) incompressible gas and b) compressible gas.

For comparison with the experiment the results of the quarter model are scaled up to a full BCD model, see Table 5.7 and Figure 5.33. The dependence of the frictional force on the gas pressure p_g is linear. The modelling of the gas pressure distribution is crucial to the correct implementation of a wear calculation, since it drastically reduces the effective contact pressure leading to wear. The implementation of the compressible gas pressure distribution results in lower effective contact pressure, i.e. 19.8 % for the calculation with 5 bar differential pressure, 24.7 % with 10 bar differential pressure, 27.4 % with 20 bar and then slowly increasing to 30.6 % with 100 bar differential pressure.

Figure 5.34 shows the effect of the gas compressibility on wear for three loads: a) 3 MPa, b) 6 MPa and c) 10 MPa. The crank end pressure is at ambient pressure (0.1 MPa). As expected, the rings wear more slowly if the compressible gas properties are taken into account in the pressure distribution of the dynamic sealing surface. Additionally the wear pattern becomes more uniform, since the pressure distribution is effected more in the region where cap segment touches the sealing segment.

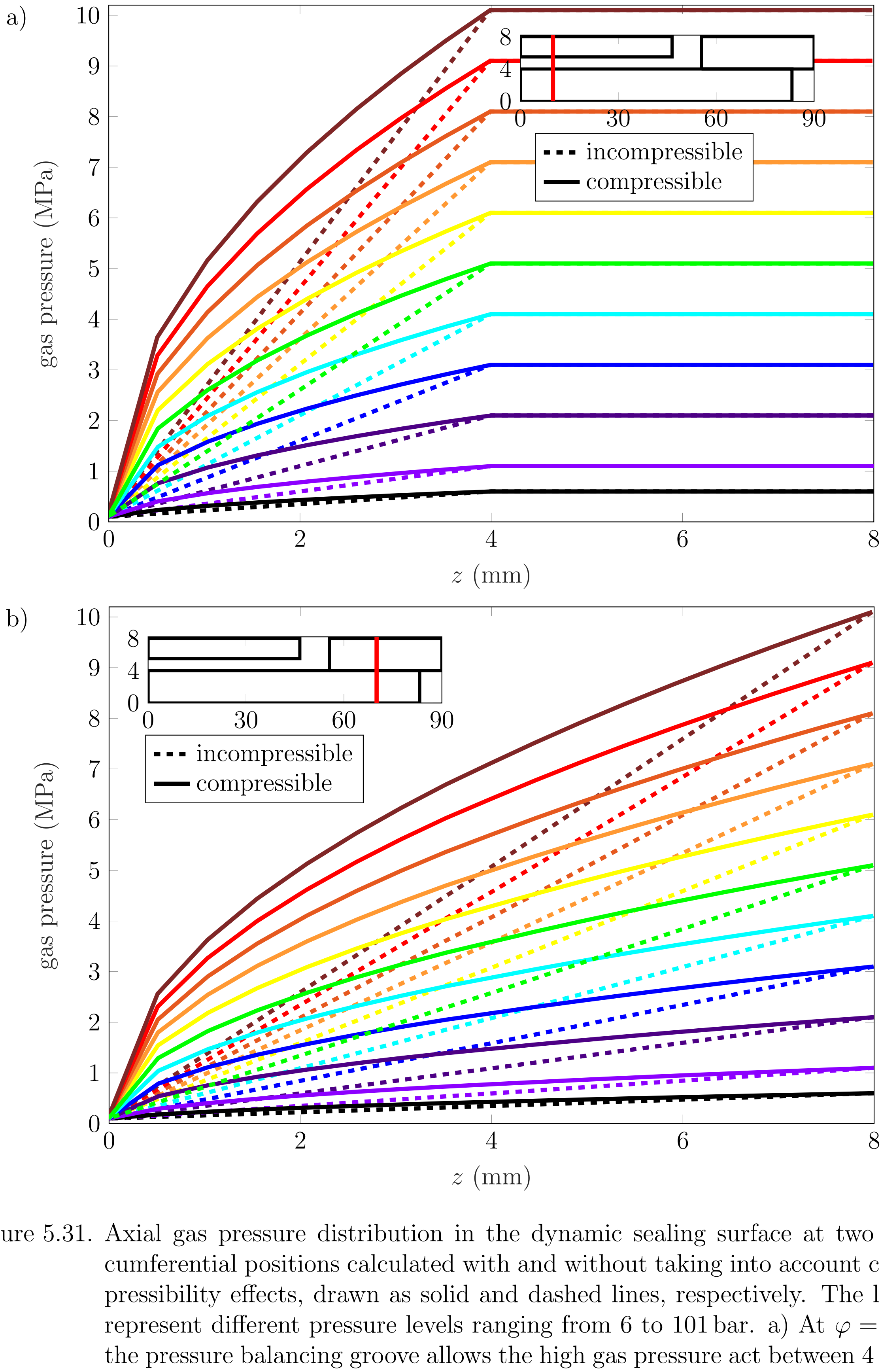


Figure 5.31. Axial gas pressure distribution in the dynamic sealing surface at two circumferential positions calculated with and without taking into account compressibility effects, drawn as solid and dashed lines, respectively. The lines represent different pressure levels ranging from 6 to 101 bar. a) At $\varphi = 10°$ the pressure balancing groove allows the high gas pressure act between 4 and 8 mm from the cup face. b) At $\varphi = 70°$ the gas pressure decreases over cap and sealing surface.

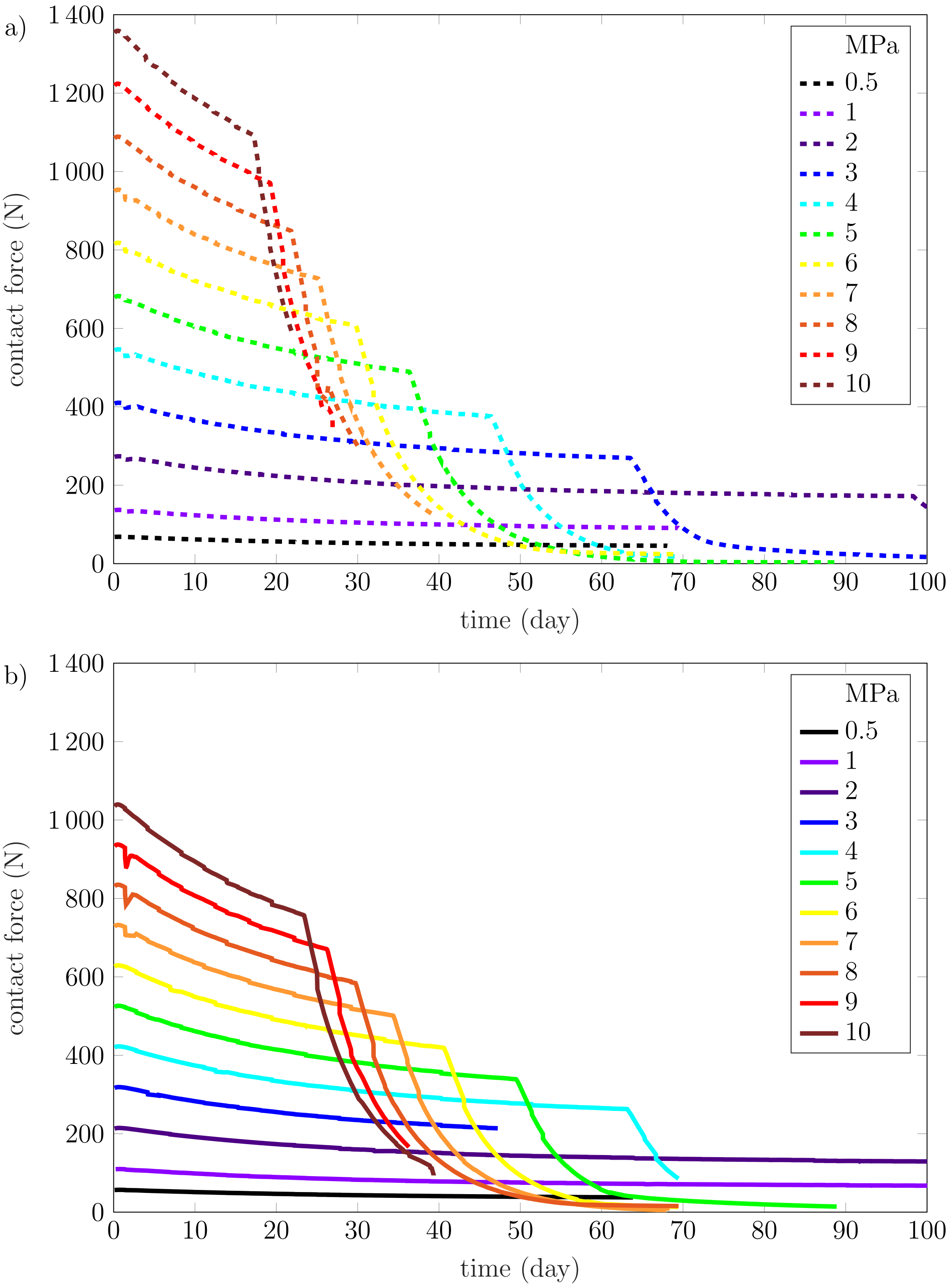


Figure 5.32. Evolution of the contact force over time for different differential pressures (MPa). The values given are valid for the quarter model. The crank end pressure is ambient, the pressure load varies from 0.5 to 10 MPa. a) shows the contact force assuming an incompressible gas pressure distribution in the dynamic sealing surface. b) gives the results assuming compressible gas properties.

Table 5.7. Contact force F_C for a single (full) BCD ring on a 50.8 mm rod, depending on the head-end gas pressure. The contact pressure is listed for i) perfectly flat surfaces assuming no gas pressure in the sealing surface $F_\text{C, flat}$, ii) an incompressible gas pressure distribution $F_\text{C, incompressible}$ and iii) a compressible gas pressure distribution $F_\text{C, compressible}$ acting in the dynamic sealing surface.

p_g (head end) (bar)	p_g (crank end) (bar)	$F_\text{C, flat}$ (N)	$F_\text{C, incompressible}$ (N)	$F_\text{C, compressible}$ (N)
6	1	584	272	227
11	1	1167	544	436
21	1	2334	1096	860
31	1	3499	1640	1276
41	1	4664	2184	1688
51	1	5829	2728	2104
61	1	6992	3272	2516
71	1	8155	3812	2928
81	1	9317	4356	3340
91	1	10478	4896	3748
101	1	11638	5432	4160

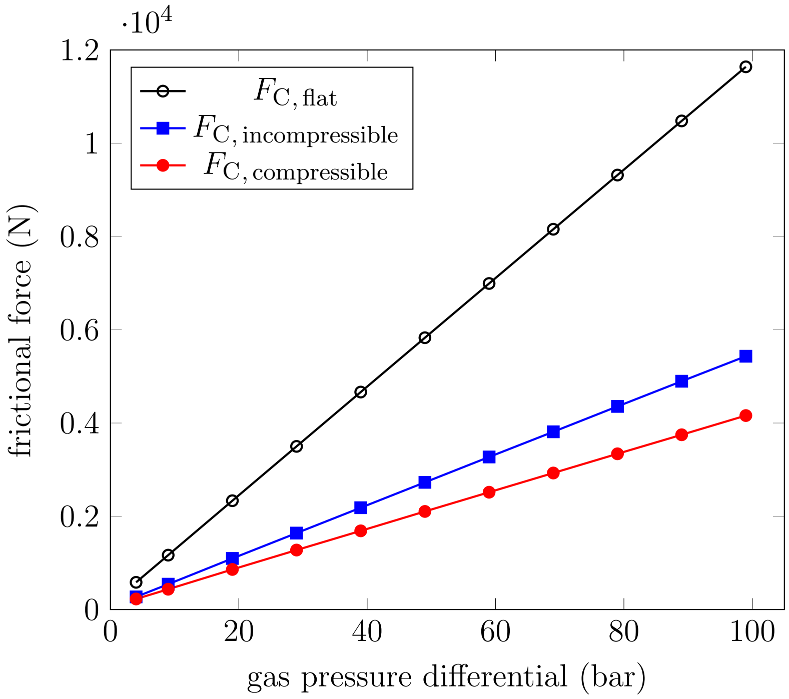


Figure 5.33. Contact force F_C of a single BCD ring on a 50.8 mm rod, depending on the sealed gas pressure differential and the type of gas pressure distribution calculation. In the dynamic sealing surface the gas is assumed to be i) non existent, ii) incompressible and iii) compressible.

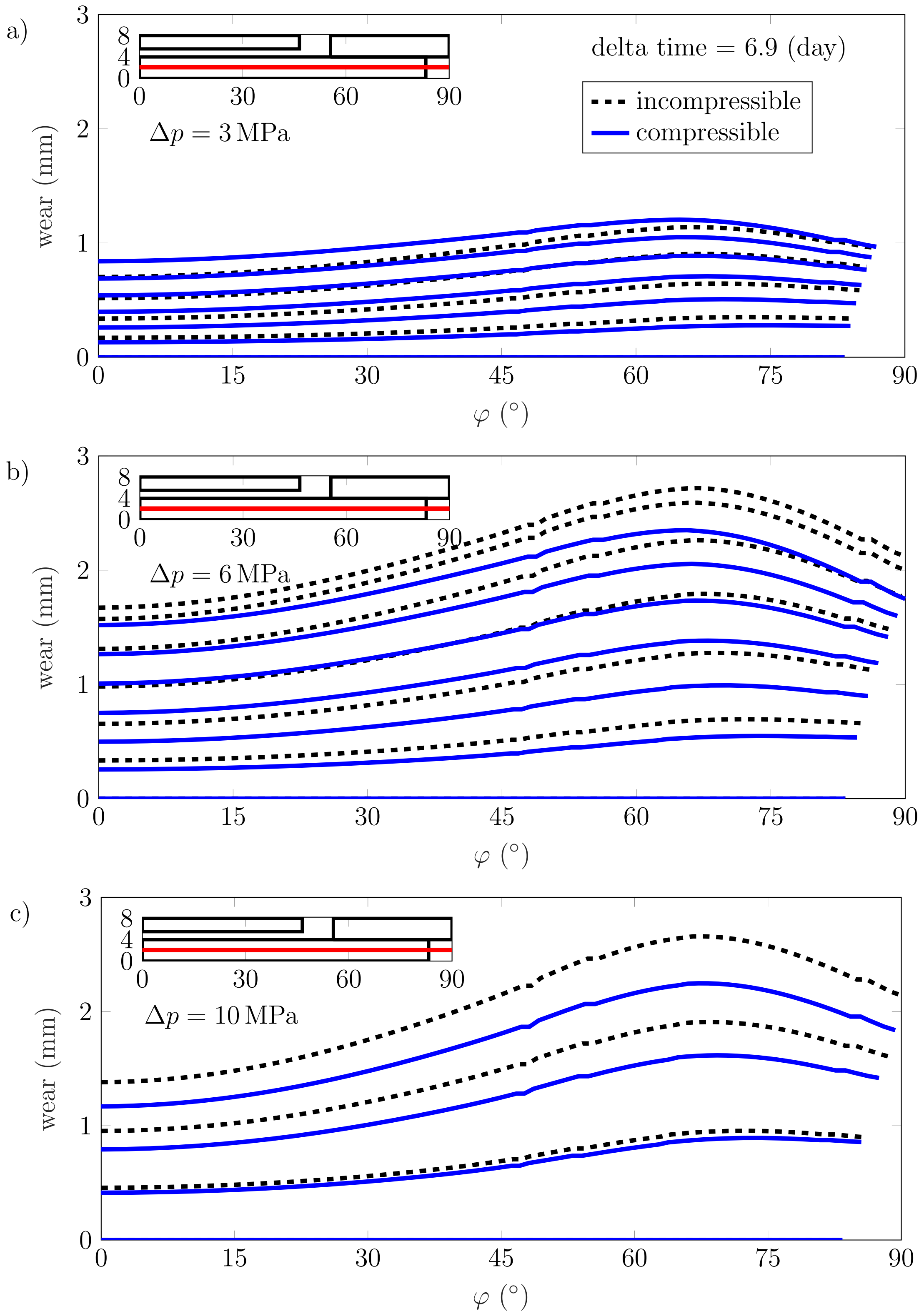


Figure 5.34. Wear pattern evolution. Black dashed lines represent calculations with incompressible gas properties. Blue solid lines represent the calculations with compressible gas properties. The plots show three different pressure differentials: a) 3 MPa, b) 6 MPa and c) 10 MPa. The crank end pressure is at ambient pressure (0.1 MPa).

5.5.5 Calculations with different elastic moduli

This section presents calculations with different elastic moduli, which represent different temperature levels. Figure 4.26 in section 4.3 presents the dependency of the elastic modulus on the temperature. The elastic modulus ranges from 350 MPa to 1400 MPa between 140 °C and 23 °C. The three elastic moduli chosen for comparison are 350, 700 and 1400 MPa representing 140, 80 and 23 °C, respectively.

The considered BCD is mounted on a 50.8 mm (2”) rod. The gas pressure distribution takes the compressibility of the gas into account. Two different pressure loads are used for this evaluation: 3 and 6 MPa. The pressure on the crank side is ambient pressure. The code pertaining to the 6 MPa calculation is provided in the appendix 7.

Figure 5.35 presents the effect of the elastic modulus on the contact forces over time for two load levels: 3 and 6 MPa. At the beginning the high modulus rings show slightly higher contact forces. The contact pressure decreases with increasing wear. A higher elastic modulus results in lower contact pressure, since the ring segments carry a larger portion of the load. This is independent of the load level, but becomes more pronounced with increasing load. The closing of the wear gap is only slightly effected, see Figure 5.35b. Higher elastic modulus means longer time until the wear gap closes. Figure 5.36 shows the evolution of the wear pattern of the sealing segments. Sealing segments with lower elastic modulus wear faster at the segment tip because the segment carries less load, the wear distribution becomes less uniform.

This observation allows the following notion: A hot-running ring will wear faster at the sealing segment’s tips. If the tips are consumed too fast, the segments are very thin when the wear gap closes. The tips might be too weak to guarantee for enough structural stability of the ring or they might even get stuck in the clearance between backup and rod. Both cases likely lead to ring failure. This underlines that lower temperatures are crucial for a ring to perform well over a long time.

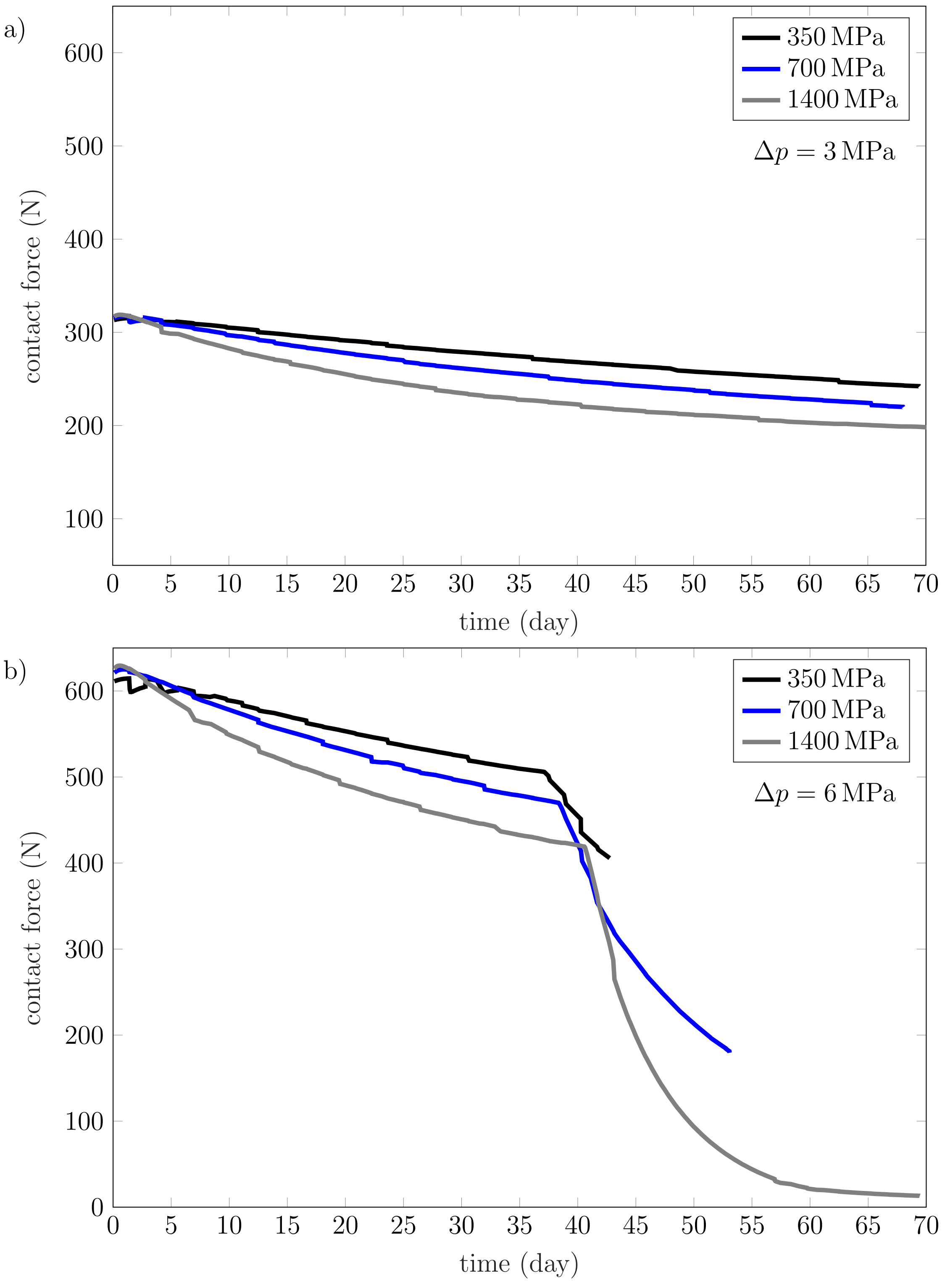


Figure 5.35. The evolution of the contact force over time and its dependency on the elastic modulus is shown for two different load levels a) 3 MPa and b) 6 MPa. The crank end pressure is ambient pressure (0.1 MPa). The elastic modulus is given in the legend.

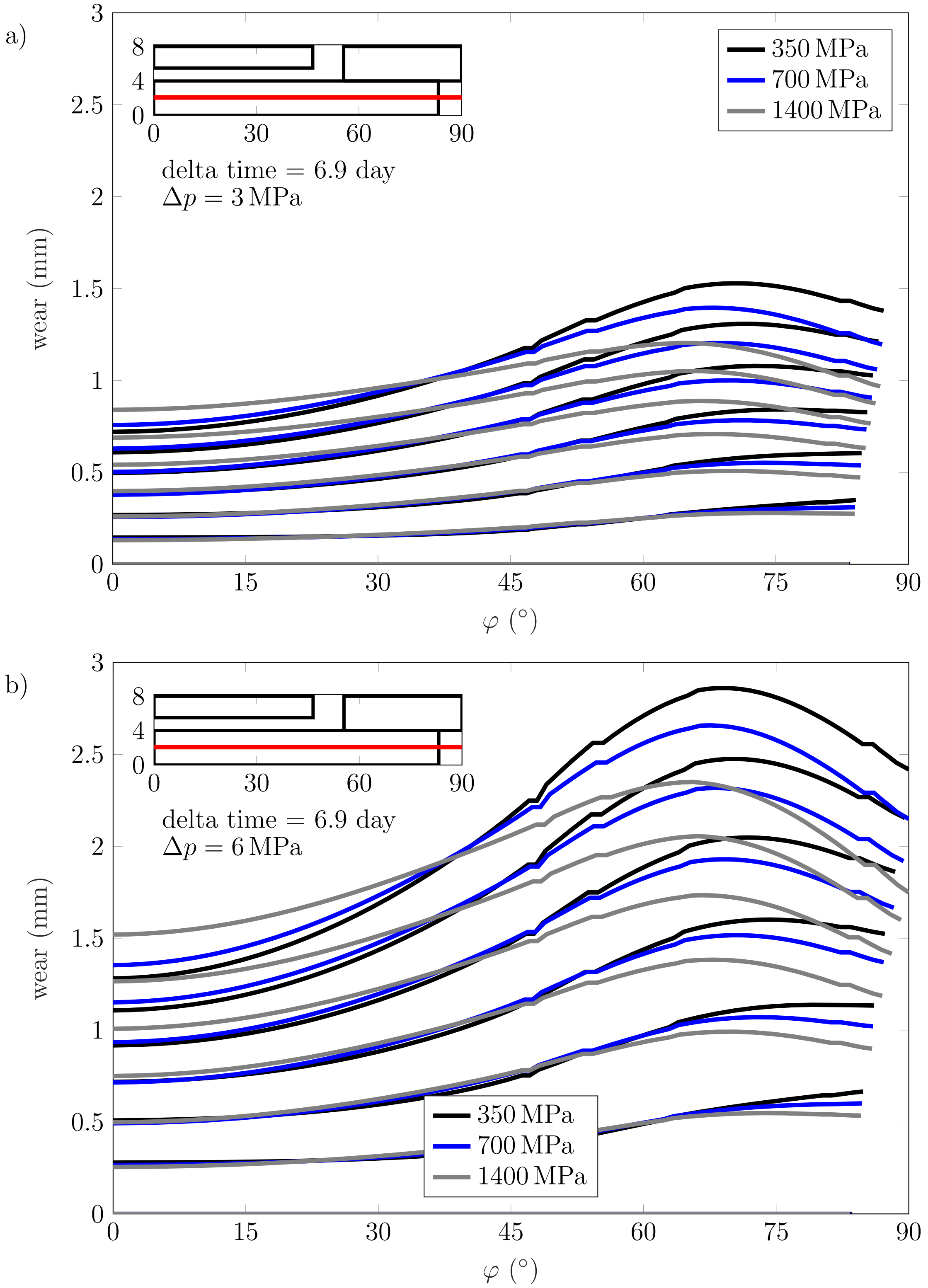


Figure 5.36. The wear pattern evolution and its dependency on the elastic modulus is shown for two different load levels a) 3 MPa and b) 6 MPa. The crank end pressure is ambient pressure (0.1 MPa). The elastic modulus is given in the legend.

Chapter 6

Experimental validation

In a compressor the packing consists of several cups each filled with a sealing element. It is unknown how the pressure is broken down by the sealing elements and how the pressure in the cups changes with time [17, 22]. Field experience has shown that sometimes two compressors of the same type, at the same site, next to each other (redundant machines) behave unexplainably. One runs smoothly, the other one is a trouble-maker. Finding the difference is often a tedious procedure and sometimes the reason for the problems is not detected.

To avoid uncontrollable irregularities as much as possible a test packing has been developed to compare ring designs, to evaluate ring features and to examine the effect of different operating conditions. Additionally a precise wear measurement procedure is presented for the BCD ring.

Parts of the content and some of the figures of this chapter are published in [35].

6.1 Test rig

A crank drive named “short block BS102”, manufactured by LMF (Leobersdorfer Maschinenfabrik, Leobersdorf, Austria), has been prepared to work with different rod diameters and test packings. It is possible to work between 365 and 911 rpm. The drive works like a boxer (engine) with horizontally opposing rods. The stroke is 98 mm. Since different rods might be used on the opposite sides at the same time, compensation weights are place on the lighter rod. The motor is placed below the crank case on the frame. Motor and drive are connected via a belt. Figure 6.1 shows a schematic sketch of the test rig.

In comparison to real compressors no cylinders and packing boxes are mounted directly to the crank drive as standard. With the presented set-up it is not possible to compress gas. There is no need for compression since the last ring in a pressure packing is usually loaded by static (suction) pressure, which is represented by the applied static pressure on the test rig.

On the test rig different gases can be used:

- Compressed air, 13 bar from the production plant (HW). This pressurised air has a dew point of 3 °C (5.95 $\mathrm{g\,m^{-3}}$). This gas is condsidered wet.
- Compressed air, 30 bar from the production plant (HW) with a dew point of −40 °C (0.117 $\mathrm{g\,m^{-3}}$). This gas is bone-dry.
- Bone-dry nitrogen from the production plant's 25 $\mathrm{m^3}$ liquid nitrogen storage tank, with 30 bar. The gas purity is nitrogen 5.0 ($\geq 99.999\,\%$ nitrogen).
- Any other non-corrosive, non-toxic gas which is supplied by a bundle of pressure bottles.

The test rig is placed indoors and the test room's oxygen content is constantly monitored to prevent asphyxia.
The short block is mounted on a steel frame which has a bench with guidings on each side i.e. for each throw. The rod is fixed at the crank head and passes a standard set of oil wipers. Outside the crank case and the wiper packing a compensation weight is placed onto the rod if necessary. Then there is approximately 400 mm rod length as test space for a test packing and instead of a piston a guiding guides the rod's free end.

6.1.1 Packing

The test packing consists of two massive steel plates between which a T-cup and flanges are placed. The T-cup has two recesses each with a width of 12.2 mm, where a sealing element is mounted on the rod. The hollow rod has an outer diameter of 50.8 mm and an inner diameter of 35 mm. The hollow design reduces the rod weight and so the accelerated masses on the test rig. Figure 6.1 shows a schematic sketch. Although no cylinder is used the sides of the packing are referred to as "cylinder" and "crank" side. Figure 6.2 shows a cut of the test packing.

The temperature is measured on 8 positions during a test. The packing temperature is monitored on four positions with thermocouples of type K. 1) and 2) are placed at a radial position of 31.9 and 35.9 mm at the packing's cylinder end side, respectively. Sensor 3) is placed in the middle of the T-cup. The tip of the sensor is in contact with the gas. Position 4) is in the crank end flange. 5) the room temperature is tracked on the cylinder side of the packing. 6) A pyrometer (impac 140, LumaSense Technologies GmbH, Frankfurt, Germany) directly measures the rod temperature through a quartz glass in the axial middle of the T-cup. The pyrometer works above 75 °C. Below this value the from the rod emitted energy is too low for this measurement principle. 7) and 8) measure the temperatures of the packing cooling water, the inlet and outlet, respectively. A heat meter measures the effect of the packing cooling using the temperatures 7) and 8). The cooling water from the packing cooling (secondary cooling circuit) is cooled via a heat exchanger with the plant's primary cooling circuit. The primary circuit is cooled outdoors, thus the cooling water temperature of the primary circuit depends on the outside temperature.

First tests showed that the packing cooling is not sufficient to regulate the rod temperature especially at elevated loads and/or rod speeds. Further improvement of the cooling was necessary to achieve rod temperatures similar to field and testing experience. So additionally the hollow rod is cooled via pressurised air on the inside. The air is guided with a lance into the hollow rod and blown off into ambient. This allows to work with higher pressures and rod speeds.

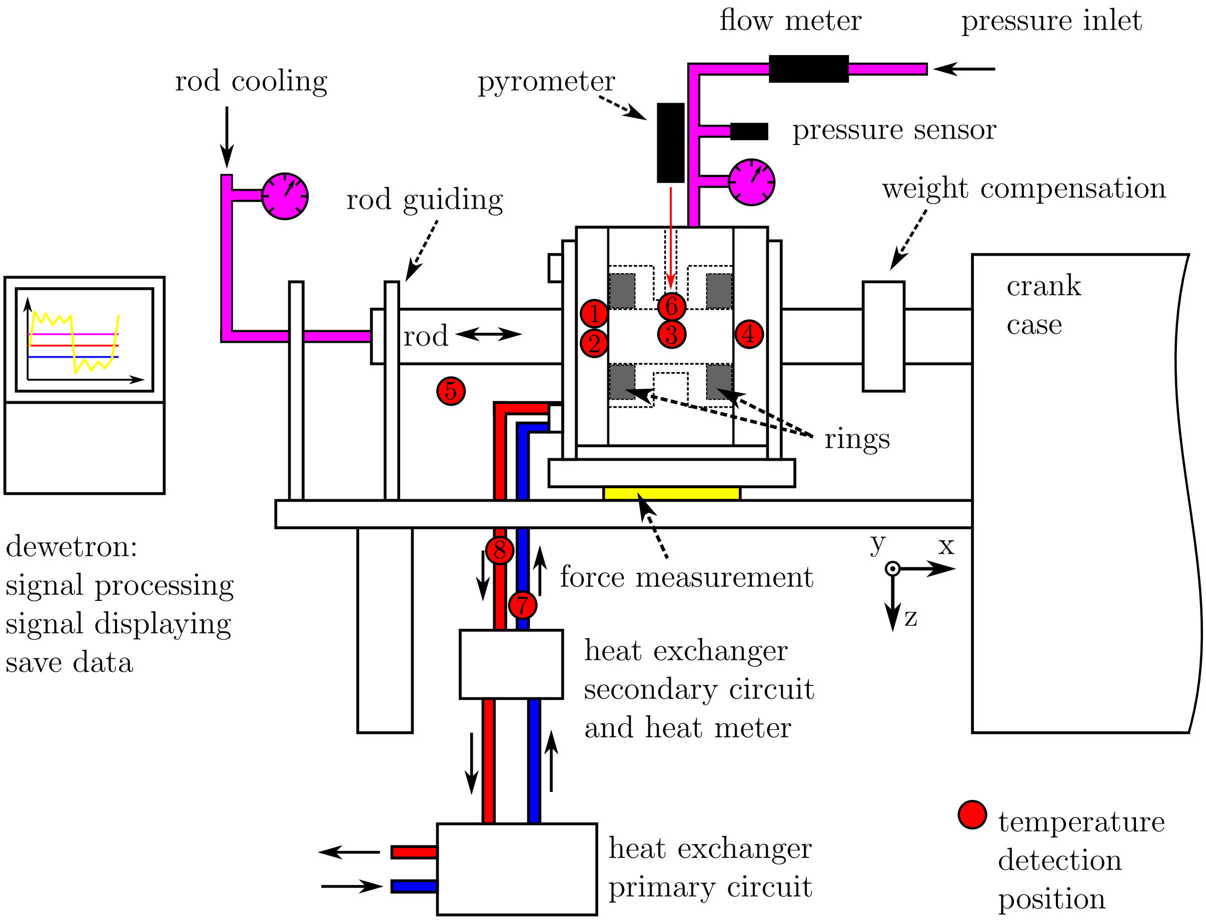


Figure 6.1. Shematic drawing of the test rig, the mounted packing and the 8 temperature measurement positions. [35]

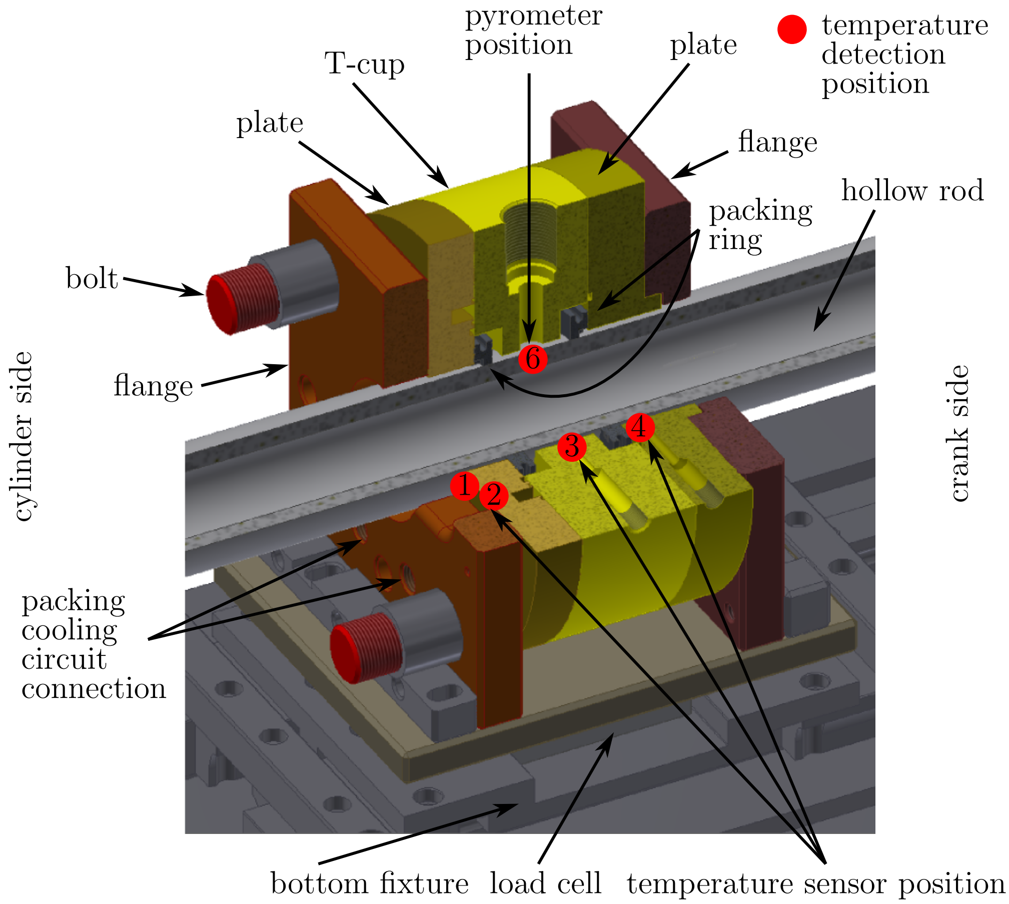


Figure 6.2. Cut through the packing. The temperature measurement positions are marked and labeled according to Figure 6.1.

The packing ring is loaded by static gas pressure. Since the pipe connecting the nitrogen or pressurized air tank with the test rig is longer than 25 m, it is assumed that the gas has ambient temperature at the inlet. Before the pressure inlet in the axial centre of the T-cup a mass flow meter and a pressure sensor are installed. The mass flow meter monitors the flow into the packing, i.e., the leakage of both rings. The rings' leakage can hardly be separated, since it leaks into ambient.
The whole packing is mounted on a steel plate which is fixed to a 3D force measurement device. It allows to measure not only the frictional force occurring between rod an ring, but also any of the perpendicular forces.

A Dewetron (DEWE-5000) processes and displays all measured data. Every minute the signals of 300 ms are saved. After conversion using the Software DeweSoft [19], Matlab [49] post-processes the data.

6.2 Test procedure

Prior to each test the rod, the backup rings and the packing rings are prepared as follows: The rod used for the packing ring tests is made of steel (1.4021) and coated with WC. The coating is according to industrial standard and consists of WC/Co/Cr with a ratio of 86/10/4 [31]. The rod's surface is cleaned and finished to a roughness of R_a=0.3 µm.
A static leakage test is performed at 6 bar prior to mounting a ring on the rod. This test is a standard production control test performed for all BCD rings. The leakage limit is 16 L min^{-1} for a BCD ring made from HY54. No significant changes could be found between the measurement after production and before mounting. Between ring production and a test a maximum of half a year passed.
A 3D tactile coordinate measurement machine precisely scans the ring geometry before and after a test. The comparison of the measurements gives the ring wear.
The sealing elements are placed in the T-cup and their initial position is noted to detect sealing element rotation. In none of more than 30 tests rotation was detected. A bronze backup ring is used. The axial thickness of the backup ring varies with the packing ring thickness, such that the axial clearance is never below 0.2 mm. Prior to each test the backup ring is lapped (industrial standard procedure).

6.2.1 Wear measurement

The BCD ring's geometry is measured before and after the packing ring test. The initial configuration of the virgin ring segments is compared to the worn configuration. Wear inevitably leads to a distortion of the unmounted packing ring due to residual stresses. In order to separate wear related shape changes from residual stress related shape changes a sophisticated measurement procedure has been devised.

The ring is disassembled and each ring segment is separately mounted on a steel fixture, Figure 6.3. The measurement procedure measures always a sealing and a cap segment at the same time. Double sided adhesive tape fixes the segments at the fixture during measurement.
To distinguish between the two sealing segments and between the two cap segments one of each is marked with “P” before the test on the pressure side. Figure 6.4 shows the chosen coordinate system and the segment's reference points. All wear results displayed are using this coordinate system.

A 3D coordinate measurement machine (CMM) (DeMeet 404, manufacturer Schut Geometrical Metrology, Groningen, The Netherlands) provides a detailed 3D scan of the whole ring geometry. The density of the data points is highest on the inner diameter, where the ring wears. Figure 4.9 shows two mounted segments and the xy, yz and xz planes used to set the origin for the measurement. The measurement routine compensates axial misalignment by setting the coordinate system on the ring's top plane.

Matlab post-processes the exported raw data. The measured configuration before the

wear test is referred to as “pre” and after the test as “post” configuration. Virtual reorientation of each segment compensates minor misalignment resulting from the manual mounting. Figure 6.5 shows the raw data of a pre measurement.
As next post-processing step the data is shifted and rotated. The origin is set between the edges, which are calculated by cutting the radial planes and the outer diameter for the cap segment or by cutting the planes, which are parallel to the $y = 0$ plane, and the outer diameter for the sealing segment. Then the data is rotated so that these edges are in the $x = 0$ plane. After aligning the data, the inner and outer diameter are fitted with circles for each measured axial (z) level and the planes' positions are recalculated. In the pre configuration the centre points of the inner and outer diameter are equal (concentric), see Figure 6.6. After the test run, due to wear and deformation, the centre points differ such, that the centre points of the inner diameter are closer to the ring segment, Figure 6.7. The significant wear on the sealing segment's ends is visible, when comparing the pre and the post configuration.
The cap segment deforms plastically during the test. The cap segment's surface contacting the sealing segments is bent, Figure 6.8. The sealing segment's deformation of the originally parallel planes is visible, they are deformed towards the centre points, Figure 6.9.
Then the radial ring dimension is evaluated dependent on the axial and circumferential position. From the measurement points a mean outer diameter is calculated. The distance from the measurement points of the inner diameter to the mean outer diameter is calculated to obtain the local radial ring thickness. The post-test local radial ring thickness is subtracted from the pre-test local radial ring thickness, giving the local wear. Figure 6.10 shows the wear pattern of the ring segments over the axial and angular position. To determine if a gradient over the axial height exists, the data is plotted as line plot, Figure 6.11, where for each z-coordinate a line is drawn. The wear measurement and analysis procedure is repeated twice for a single BCD ring, so the full wear is determined and can be displayed within one plot, see Figure 6.12.

The total wear volume of a ring is derived from the measurement. The total wear volume is hard to measure directly with contacting measurement because edges and corners are hardly accessible and not well reproducibly measurable, especially if a geometry changes unpredictably due to wear and deformation. Thus the measured wear volume is extrapolated by a factor created from the ratio of the measurement area and the real contacting area. The extrapolation in axial direction is reasonable, since the change in the wear pattern in axial direction is small (Figures 6.10 and 6.11). The extrapolation in circumferential direction is small since this direction is scanned with high resolution and nearly to the segment ends. The up-scaling factor is approximately 1.4. Table 6.1 presents the measured and calculated wear volume data for the full BCD ring, from which the sealing segment and the cap segment without “P” are presented below.

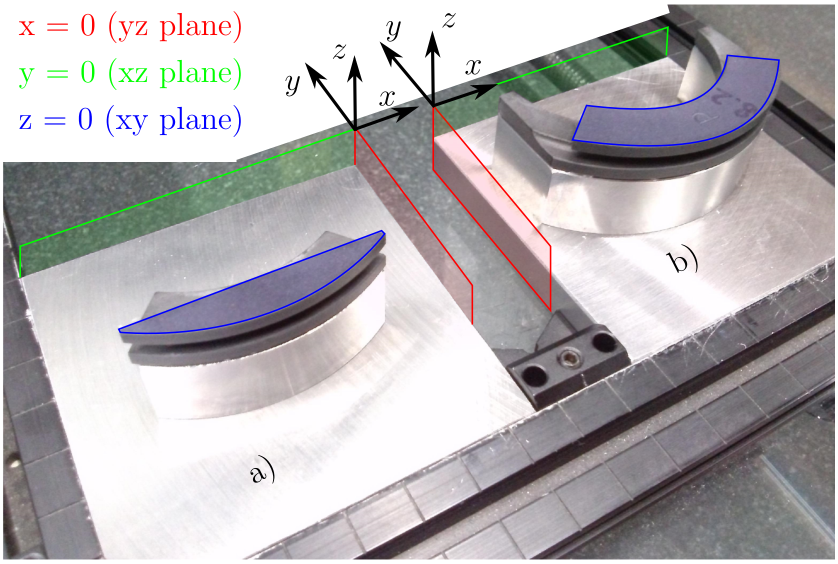


Figure 6.3. a) a cap segment and b) a sealing segment ready for 3D scanning. The segments are mounted on a steel fixture on the CMM. For both segments the origin is set at the intersection of the three planes drawn in red (yz plane), green (xz plane) and blue (xy plane).

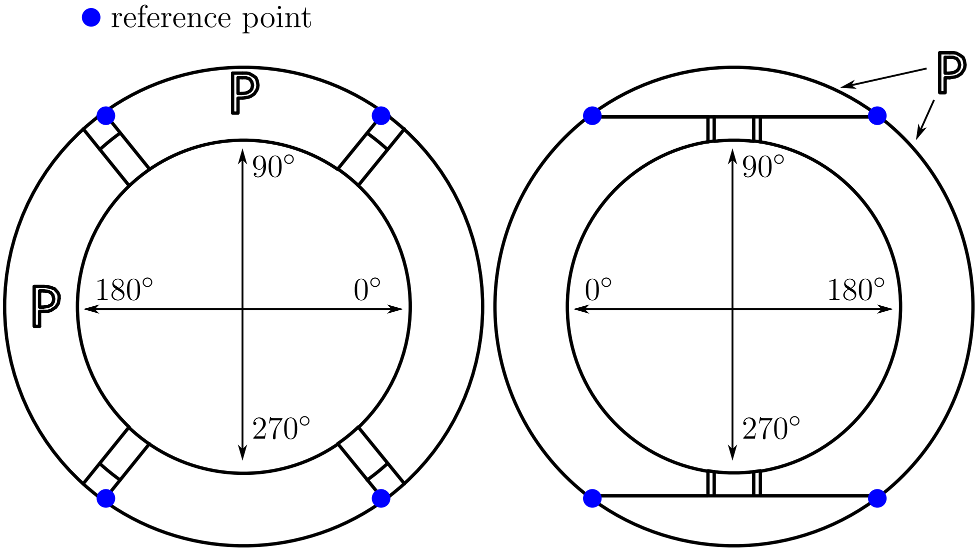
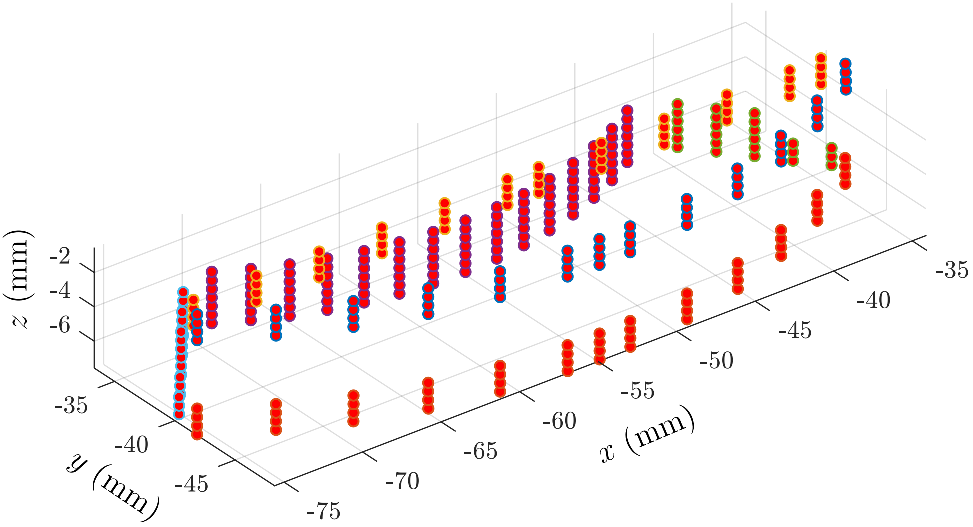
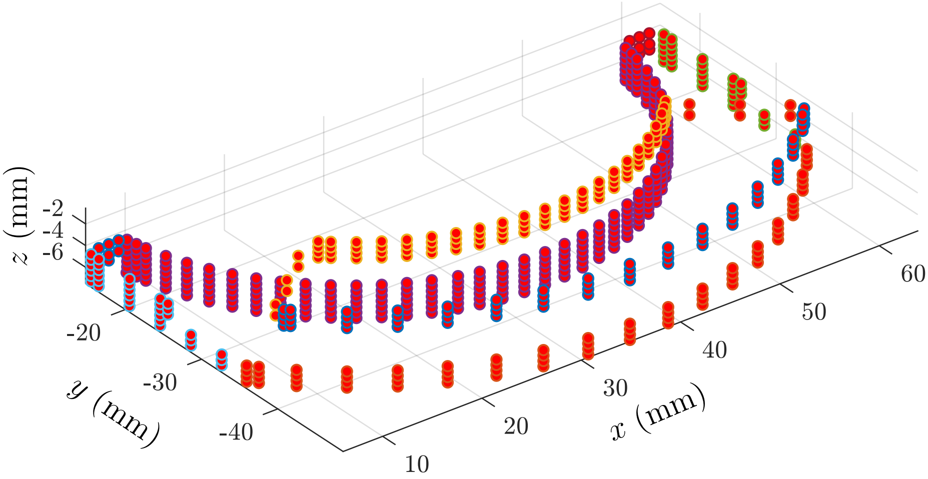


Figure 6.4. Sketch of the BCD ring geometry. Left: One cap and one sealing segment are marked with “P” on the high pressure (cylinder) side. The location of the reference points is shown. Right: Low pressure (crank) side of the BCD, the two marked segments are on the upper right. Both sketches show the coordinate system used for displaying results.

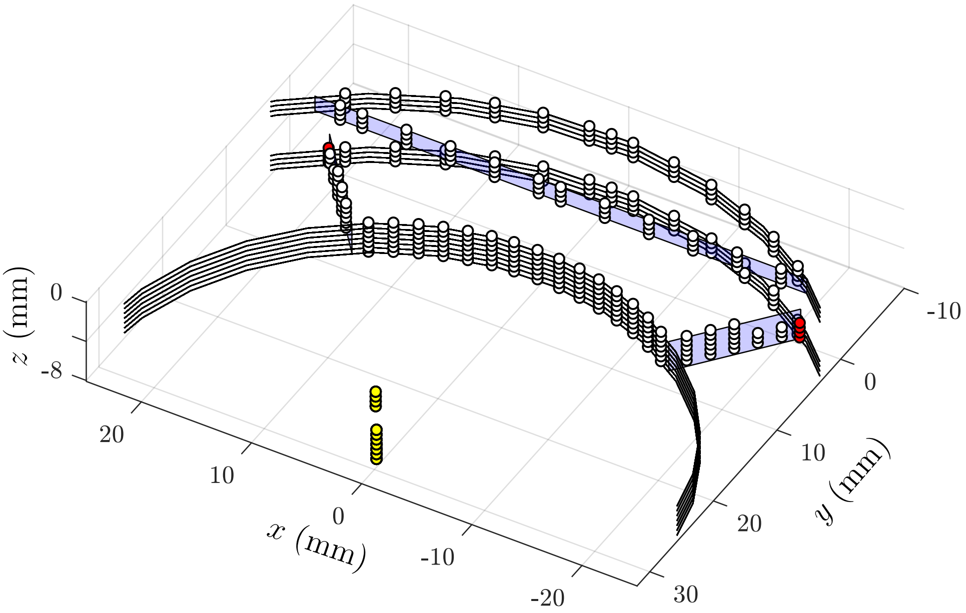


(a) Cap segment.

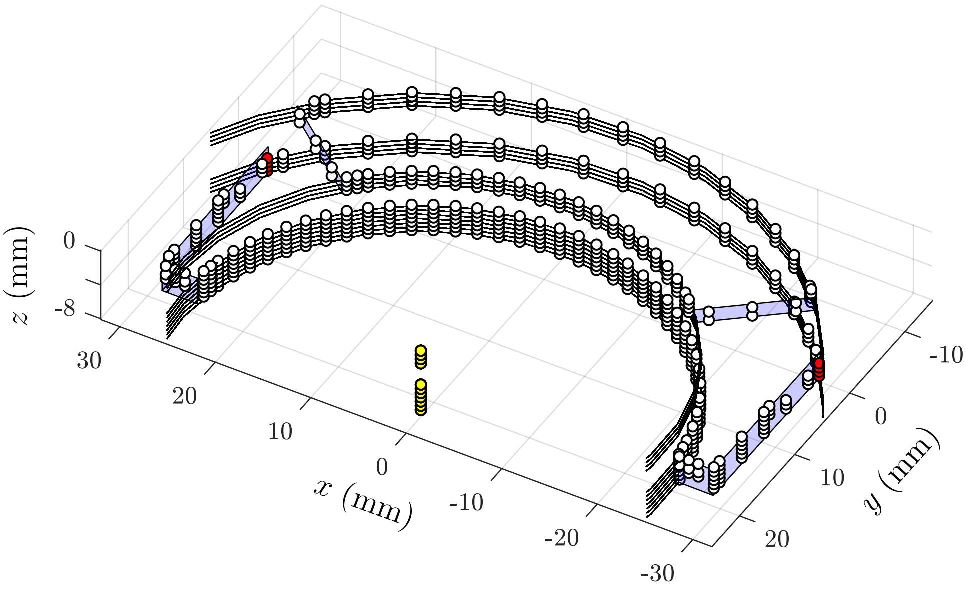


(b) Sealing segment.

Figure 6.5. Raw measurement data from BCD ring segments. The origin is from the CMM.

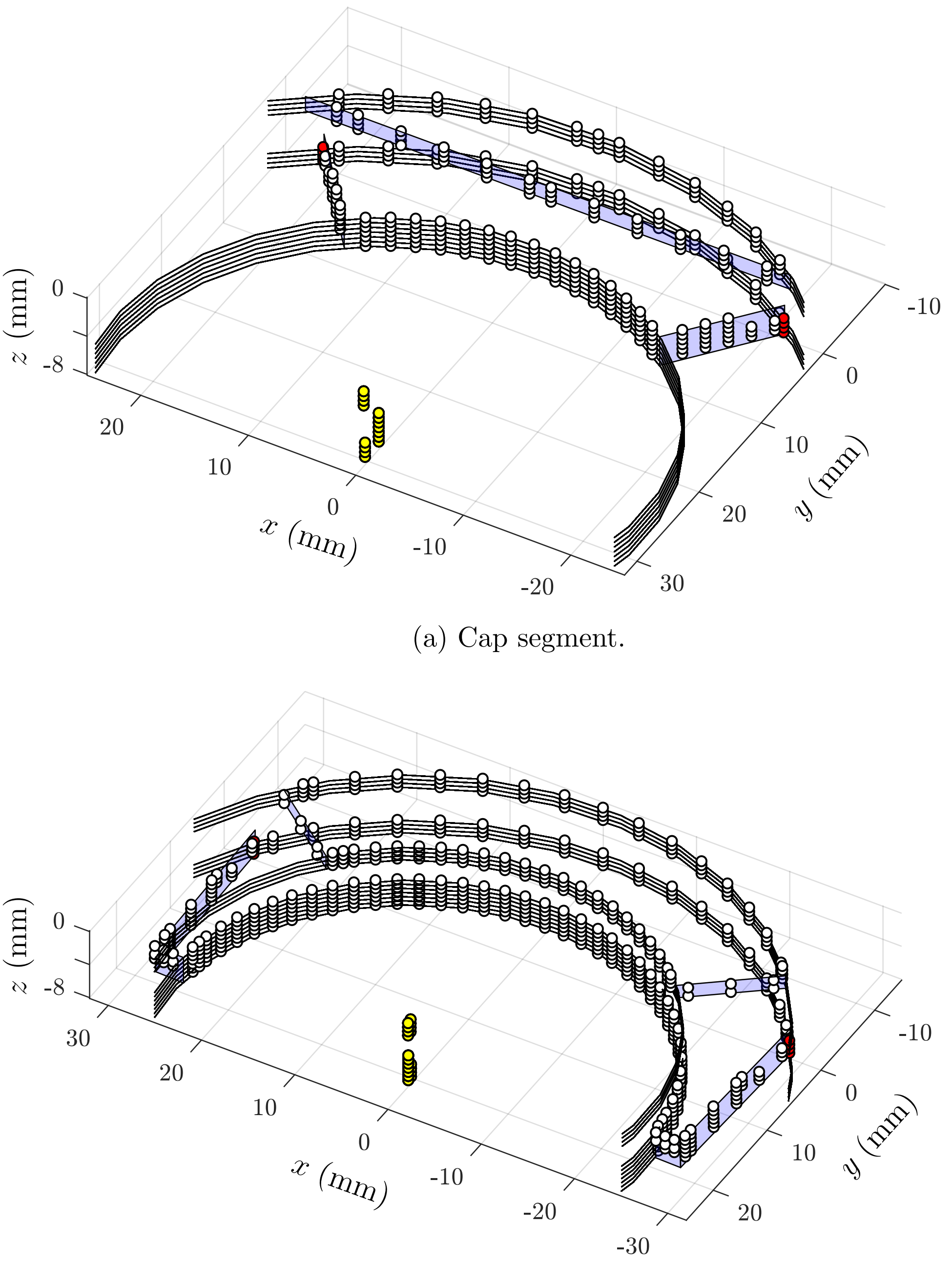


(a) Cap segment.



(b) Sealing segment.

Figure 6.6. “Pre” configuration: Shifted and rotated data from 3D scans. The origins are reset between the calculated reference points. The fitted diameters are drawn as black line, their centre points as yellow filled circles, the fitted planes are shaded blue. Each calculated reference point is represented by a filled red circle and results from the intersection of a fitted outer diameter and a fitted plane.

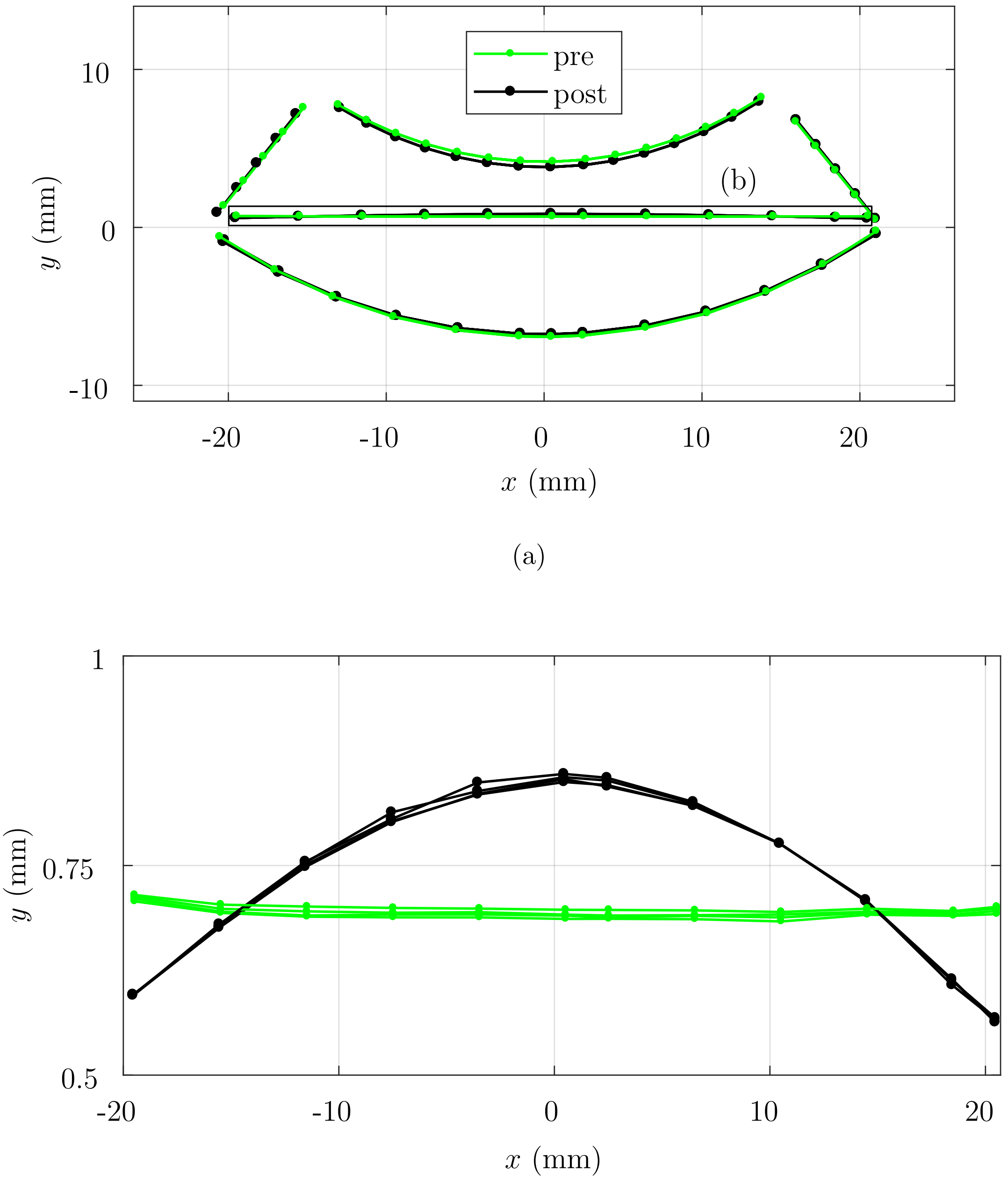


(a) Cap segment.

(b) Sealing segment.

Figure 6.7. "Post" configuration: Shifted and rotated data from 3D scans with the origins are reset between the calculated reference points on the edges. The fitted diameters are drawn as black line, their centre points as yellow filled circles, the fitted planes are shaded blue. Each calculated reference point is represented by a filled red circle and results from the intersection of a fitted outer diameter and a fitted plane.

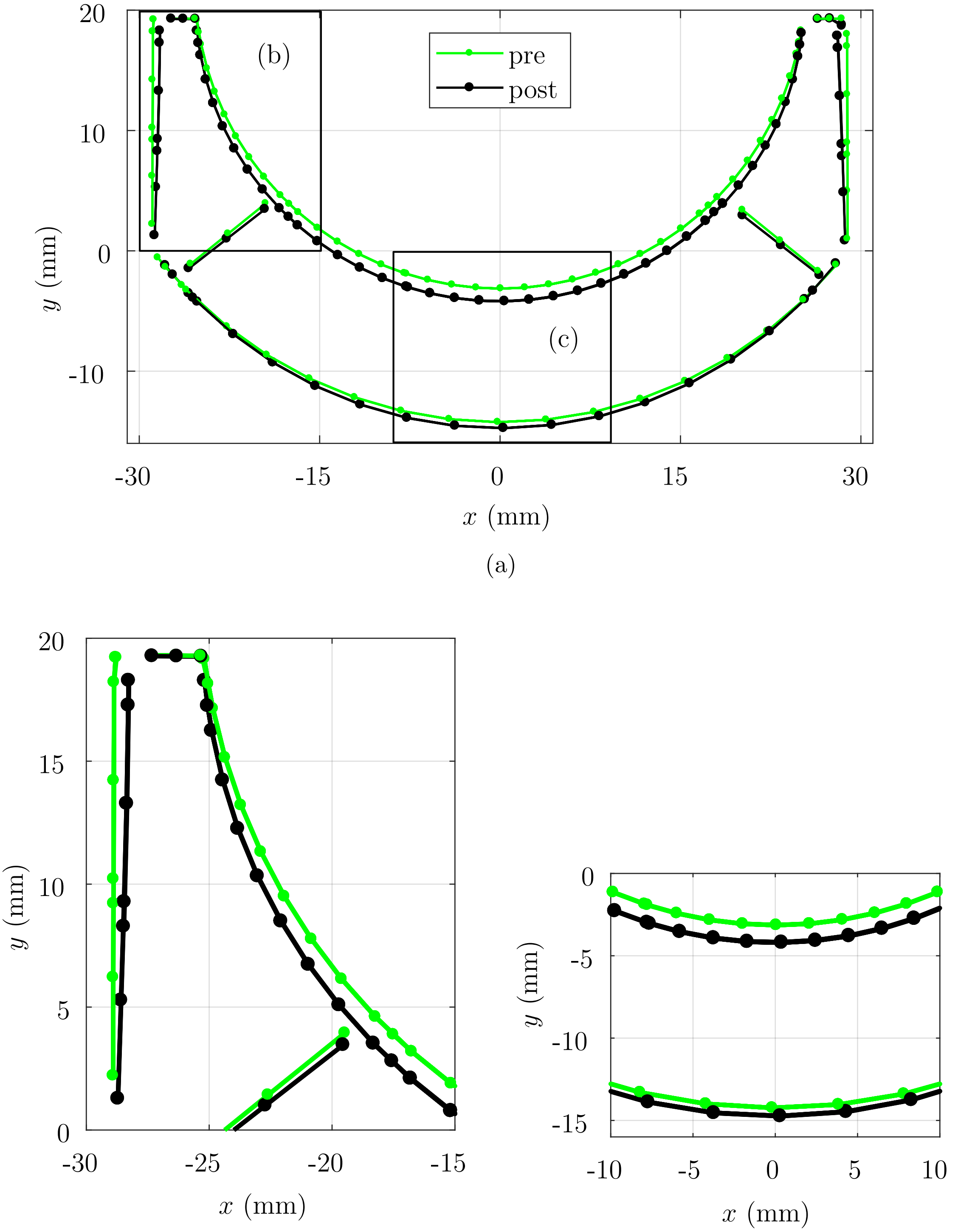
The Figures 6.8 and 6.9 present the difference resulting from wear and plastic deformation between pre and post configuration for the cap and sealing segment, respectively. The pre configuration is drawn in green, the post configuration in black. The dots represent measurement points.



(a)

(b)

Figure 6.8. (a) Overview of the pre and post test data of a BCD cap segment. (b) The deformation of the cap segment facing the contact area contacting the sealing segments.

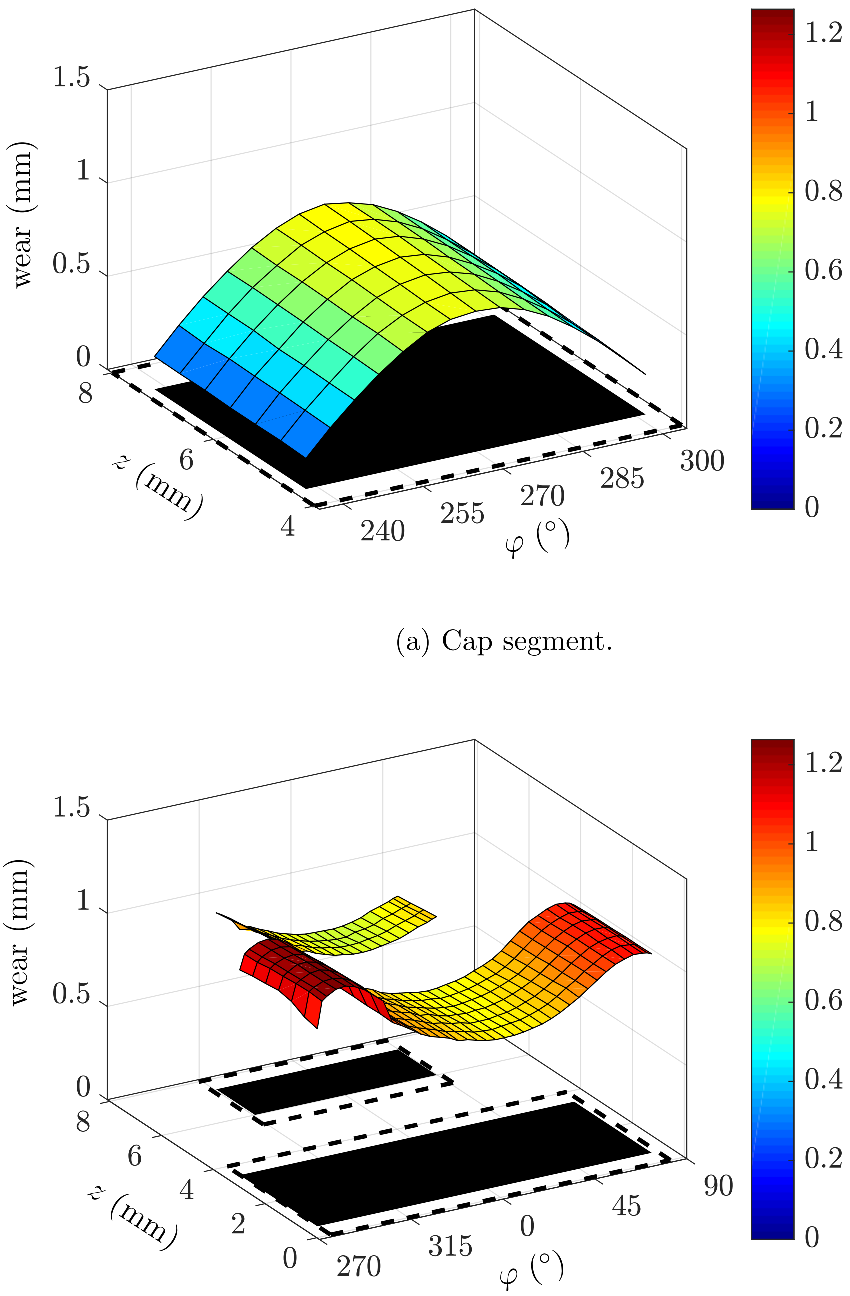


(a)

(b)

(c)

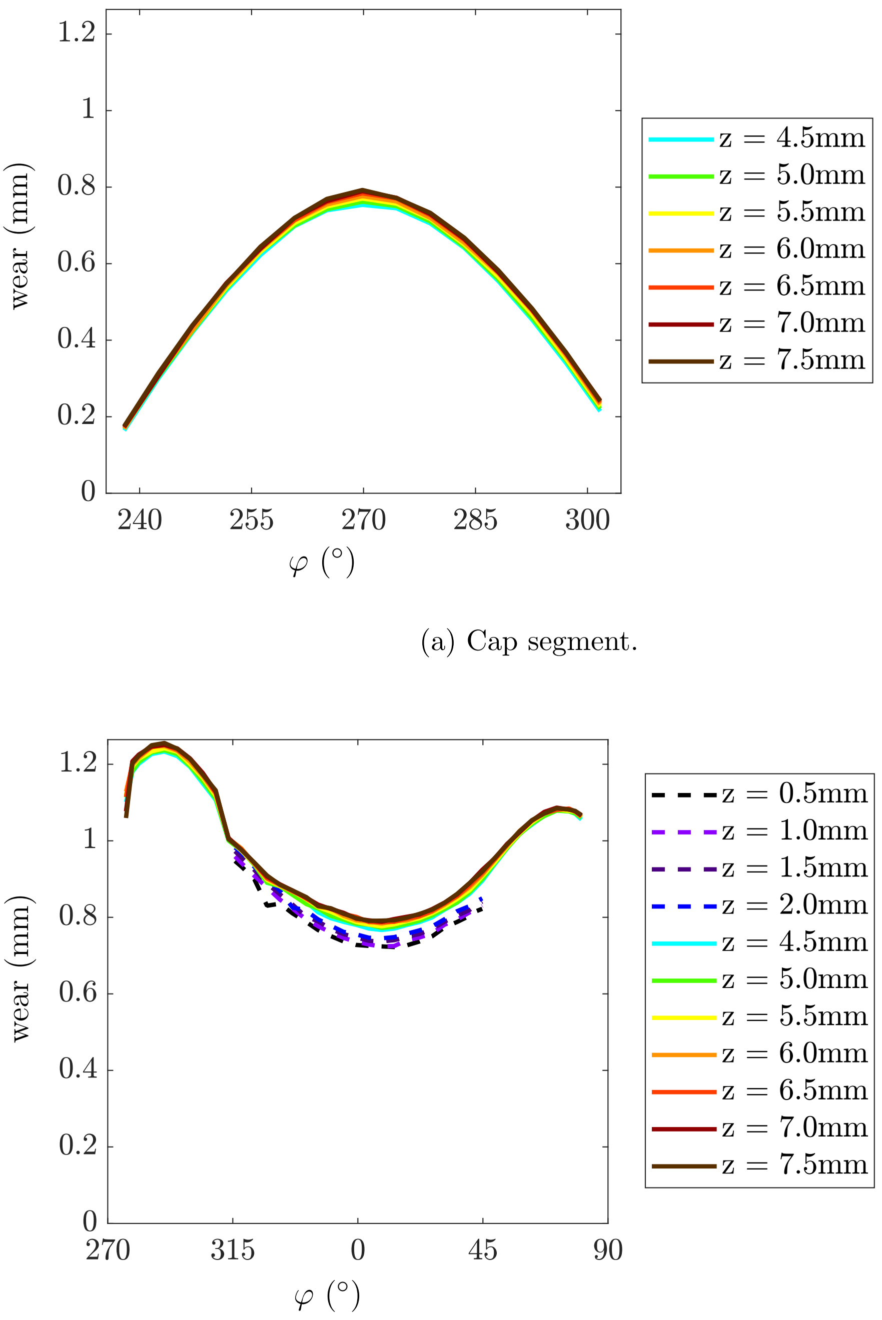
Figure 6.9. (a) gives an overview of the pre and post test data of a BCD sealing segment and the position of (b) and (c), which are displayed in higher magnification. (b) The deformation at the sealing segment's end is of the same order of magnitude as the wear. (c) The change of the inner diameter also causes a permanent deformation of the outer diameter.



(a) Cap segment.

(b) Sealing segment.

Figure 6.10. The wear pattern of the ring segments results from subtracting the post configuration from the pre configuration.



(a) Cap segment.

(b) Sealing segment.

Figure 6.11. Wear pattern of the ring segments viewed in axial direction. The data is the same as in 6.10. Each line represents an axial position given in the legend. The dashed lines in (b) are the part of the sealing segment which is between the high pressure side of the ring and the pressure balancing groove.

Table 6.1. Detailed wear and wear volume data. The pressure balancing groove splits a sealing segment's contact area in two sections. One section is on the head end side (high pressure side) of the pressure balancing groove, the other on the crank end side. The results for these areas are provided separately.

variable	value	unit
area data		
theoretical area virgin state cap segment	138.3	mm^2
theoretical area virgin state sealing segment head end	103.1	mm^2
theoretical area virgin state sealing segment crank end	295.2	mm^2
total theoretical BCD area virgin state	1073.3	mm^2
measured area cap segment P	90.1	mm^2
measured area sealing segment P head end	60.3	mm^2
measured area sealing segment P crank end	221.0	mm^2
measured area cap segment	89.3	mm^2
measured area sealing segment head end	60.1	mm^2
measured area sealing segment crank end	220.2	mm^2
wear volume data		
measured wear volume cap segment P	32.55	mm^3
measured wear volume sealing segment P head end	36.58	mm^3
measured wear volume sealing segment P crank end	163.95	mm^3
measured wear volume cap segment	55.37	mm^3
measured wear volume sealing segment head end	40.40	mm^3
measured wear volume sealing segment crank end	181.39	mm^3
up-scaled wear volume cap segment P	49.96	mm^3
up-scaled wear volume sealing segment P head end	62.49	mm^3
up-scaled wear volume sealing segment P crank end	219.08	mm^3
up-scaled wear volume cap segment	85.77	mm^3
up-scaled wear volume sealing segment head end	69.28	mm^3
up-scaled wear volume sealing segment crank end	243.18	mm^3
total measured wear volume	510.25	mm^3
total up-scaled wear	729.76	mm^3
radial wear data		
mean radial wear cap segment P	0.361	mm
mean radial wear sealing segment P head end	0.606	mm
mean radial wear sealing segment P crank end	0.742	mm
mean radial wear cap segment	0.620	mm
mean radial wear sealing segment head end	0.672	mm
mean radial wear sealing segment crank end	0.824	mm
mean radial wear	0.689	mm

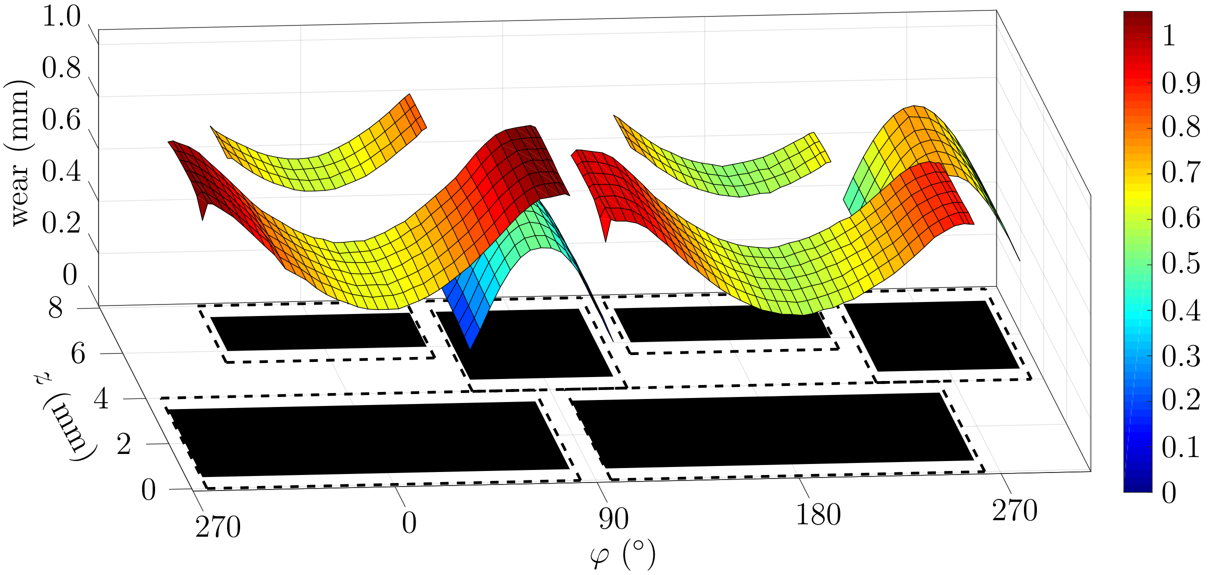


Figure 6.12. Wear of all ring segments. The segments marked with “P” are the cap segment at 90° and the sealing segment at 180°.

6.2.2 Measurement reproducibility

Repeated measurement of the same ring proves the reproducibility and capability of the 3D scanning method and the virtual aligning process:
A BCD ring is mounted, scanned and demounted twice. Figure 6.13 presents the difference between the two scans for a whole BCD ring. A difference greater than 0.01 mm occurs only at the circumferential ends of the scanned areas. These artefacts from mounting occur since the measurement procedure tries to evaluate the inner diameter in circumferential position as far as possible. The circumferential length of the inner diameter increases due to wear, thus the measurement procedure for a virgin ring requires careful mounting at the fixture.
Figure 6.14 shows the same data, while the viewing direction is the axial direction. For each ring segment a separate graph presents the high repeatability and data quality of this method. The differences of approximately ±0.01 mm is in the range of the sensitivity of the CMM.

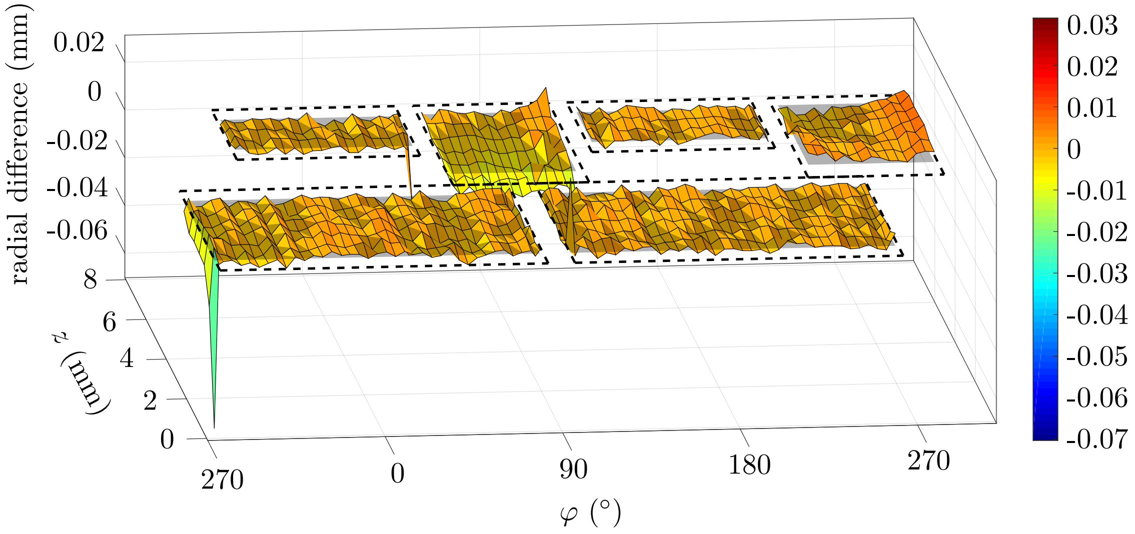


Figure 6.13. Reproducibility test: The difference between two measurements of a single BCD ring is depicted.

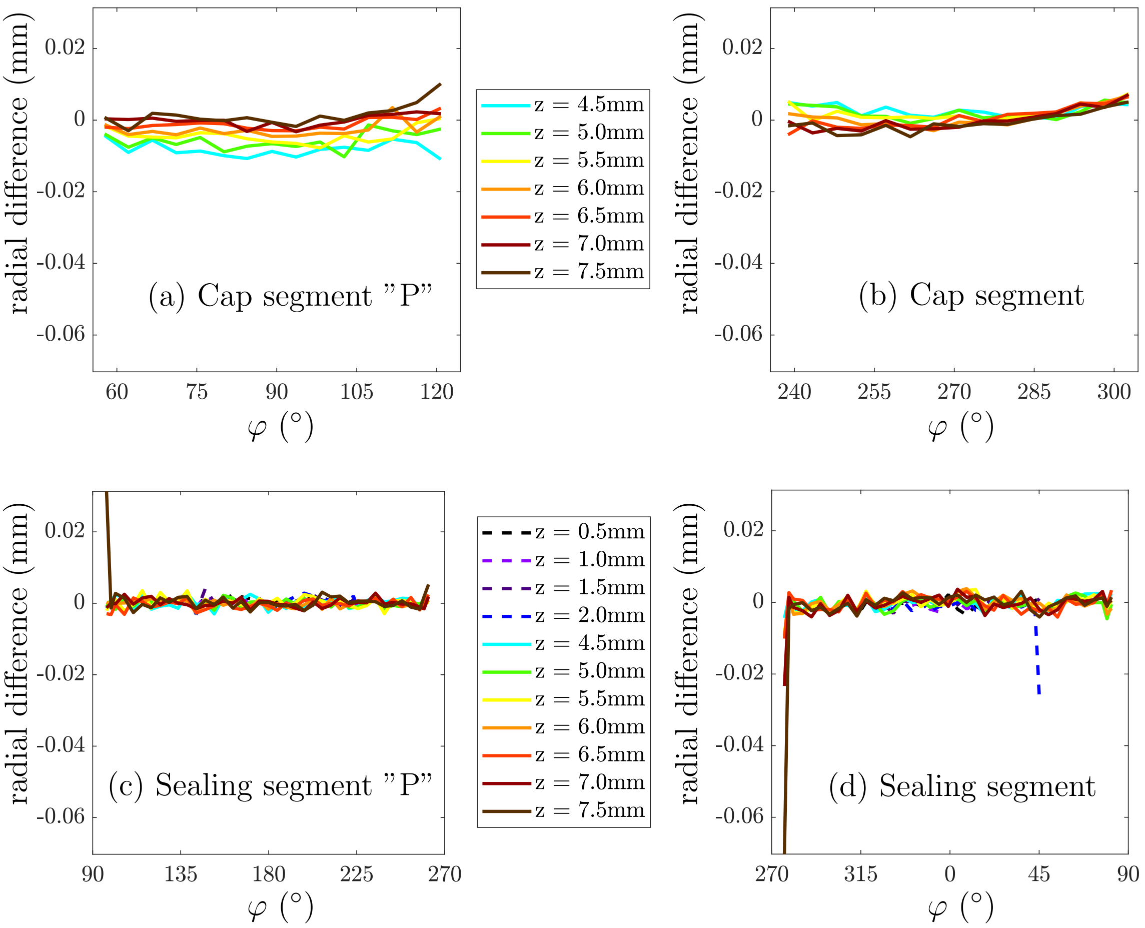


Figure 6.14. Reproducibility test: The difference between two measurements of a single BCD ring are presented. The viewing direction is the axial direction. For each segment (a) to (d) the data is drawn separately.

6.3 Test results

Out of a test series two tests are chosen for discussion. The first test, named A, shows two things: Firstly what can happen if a ring operates outside it's design limits i.e. becomes too hot. Secondly a reason why several rings are used in a packing. In the second test run, named B, rings experience common operating conditions. Test B is used later to compare simulation and experiment.

6.3.1 Test A

Table 6.2 gives the parameters for test A. The rings in this test are named A1 (head end) and A2 (crank end). Note that the used ring material (HY54) is not designed for bone-dry operating conditions, thus the wear rates are significantly higher. The combination of 10 bar gas pressure and $2.15\,\mathrm{m\,s^{-1}}$ rod speed led to high rod temperatures. In comparison to a real compressor the rod is not cooled by the fresh and cool gas, which enters the compression chamber during suction. The cooling inside the rod is turned on, but does not stop the rod temperature from rising above the material design temperature 150 °C.

Figure 6.15 shows the averaged rod and packing temperature, the leakage[1] for both rings and the gas pressure over time. The temperatures are averaged over the rod length or the whole packing housing. The test duration is 192 h.
The temperature is very high within few minutes and stays above 150 °C. After only 17 h the rings fail the first time with leakage rates above $200\,\mathrm{l\,min^{-1}}$. Since the pressure is only regulated, but not controlled, the high leakage is accompanied by a pressure drop. Nonetheless, the test was continued to check if the rings seal again. After failing, the leakage rate dropped continuously and 6 h later the rings sealed fully for another 10 h. This pattern of sealing and failing repeated unexpectedly regularly with this set of parameters. For closer consideration of such a cycle, view Figure 6.16, where the grey highlighted area of Figure 6.15 is zoomed.

Photos of the worn rings can be seen in Figure 6.17 and Figure 6.18 for the rings A1 and A2, respectively. Both rings have segments sticking to each other after the test. The segments seem welded together and snap apart when applying force.
A microscopy of the cap segments shows significant difference between a cap segment with and without a sealing segment sticking to it, see Figure 6.19. The cap segment which did not stick after test shows the usual homogeneity of HY54. Whereas the sticking segment appears to be covered in fibrils, which are assumed to be PTFE fibrils, generated during breaking the segments apart. The fibrils cover the cap segment from the wear gap towards the cap segment's ends. The density of fibrils seems highest near the wear gap and decreases towards the segment's end. This is in accordance to the calculated contact pressure which is highest at the sealing segments ends.

[1] The leakage rates given in this section are in standard litre per minute. The term "standard" refers to the standard conditions of the gas, i.e. 101 325 Pa(=1 atm) and 273.15 K(=0 °C).

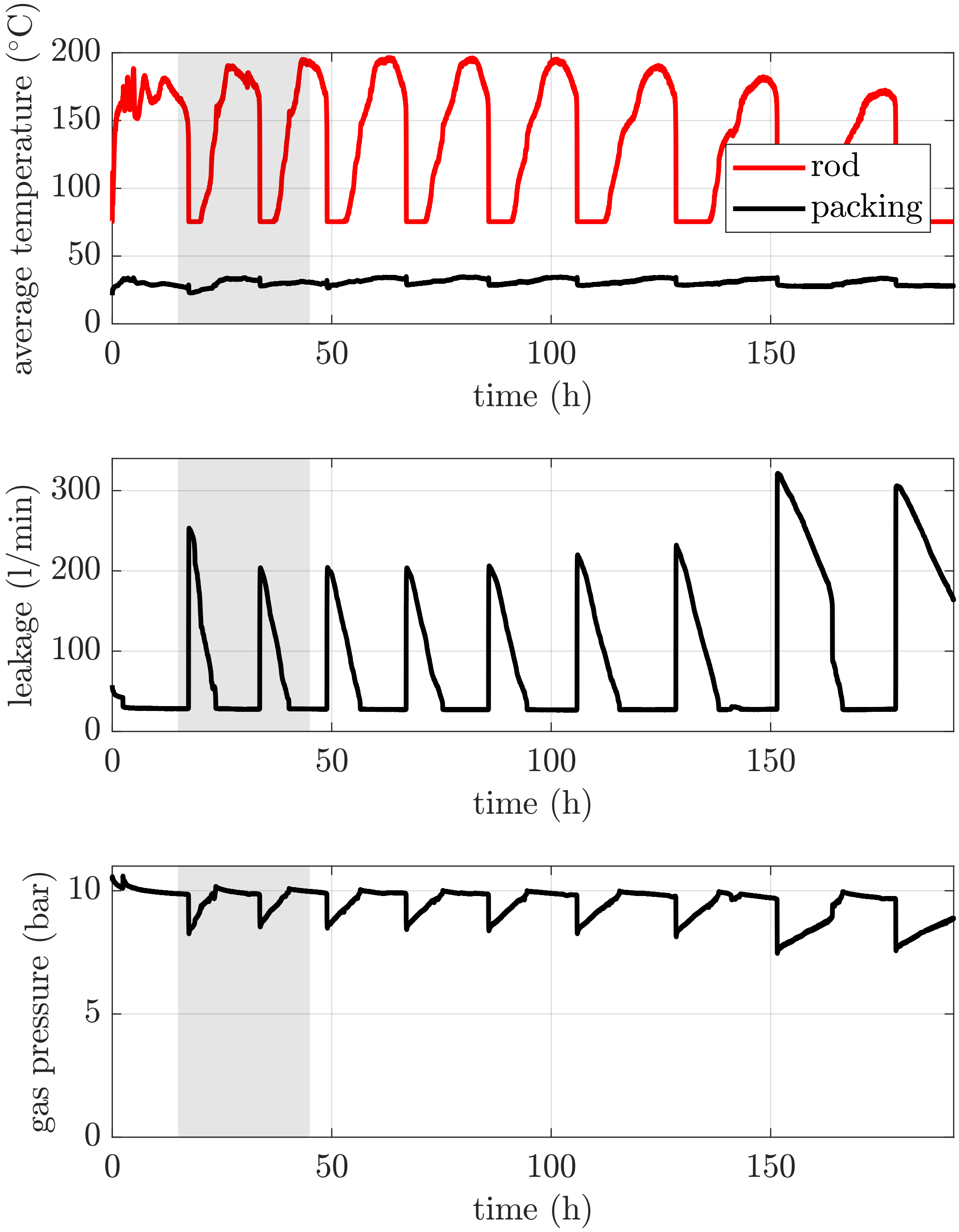


Figure 6.15. Protocol of the averaged rod and packing temperature, the leakage and the gas pressure over time for test A. The area highlighted in grey between the test time of 15 and 45 h is zoomed and shown in Figure 6.16.

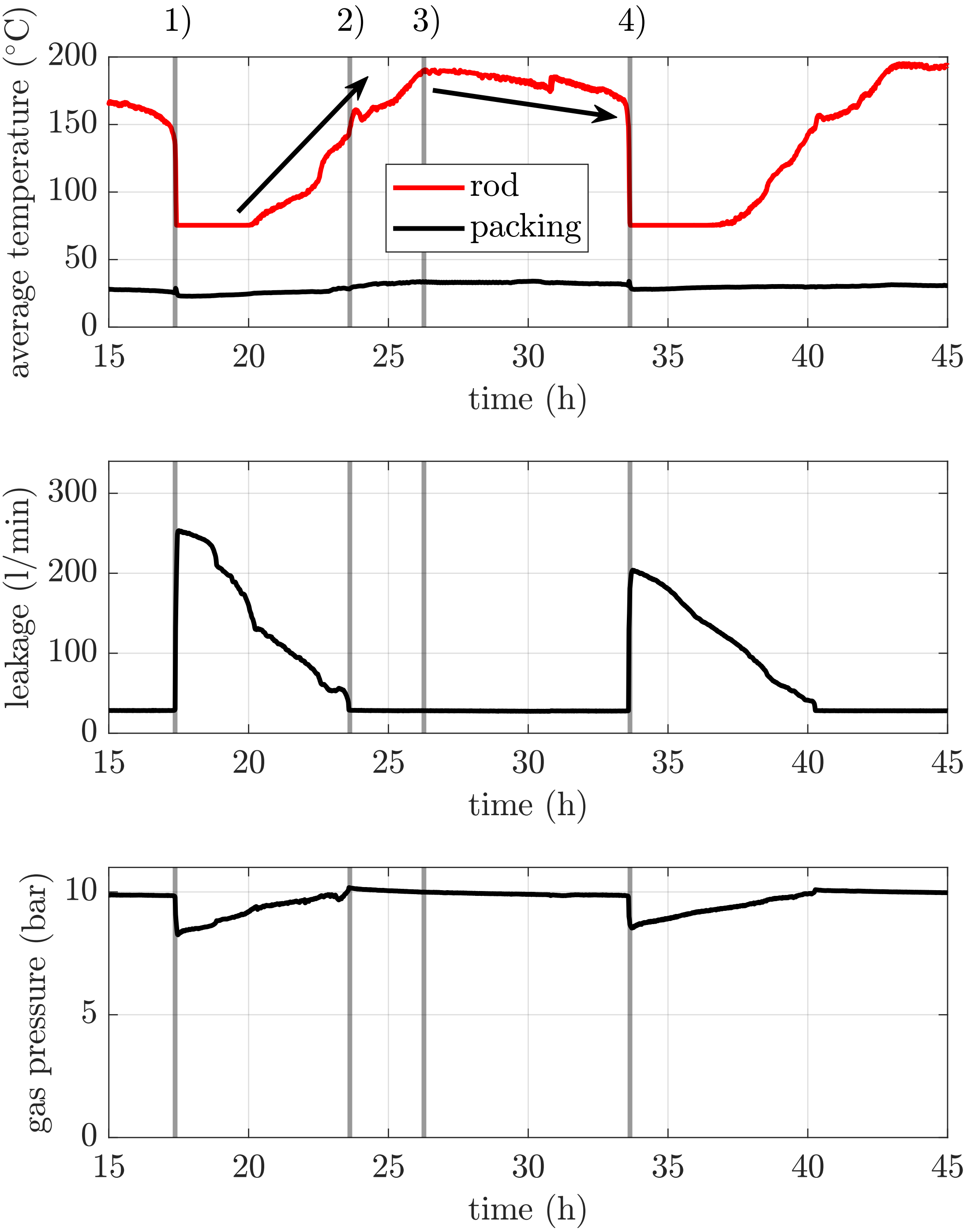


Figure 6.16. Zoom from Figure 6.15. The rings fail and seal repeatedly. Position 1) marks the failing of the rings. The high leakage is accompanied by pressure loss and an instant drop of the rod temperature. The leakage rate drops between 1) and 2) until the rings seal again. Meanwhile the rod temperature increases until a maximum at position 3) is reached, the rings are still sealing. After 3) the rod temperature slowly decreases, while the rings seal until the rings fails again at 4).

Table 6.2. Test parameters for test A.

parameter	value	unit
gas type	nitrogen	
gas dryness	bone-dry	
pressure	10	bar
average rod speed	2.15	$\mathrm{m\,s^{-1}}$
ring material	HY54	
backup material	bronze	
rod inside cooling	yes	
static leakage test at 6 bar		
ring A1	8.6	$\mathrm{l\,min^{-1}}$
ring A2	13.2	$\mathrm{l\,min^{-1}}$

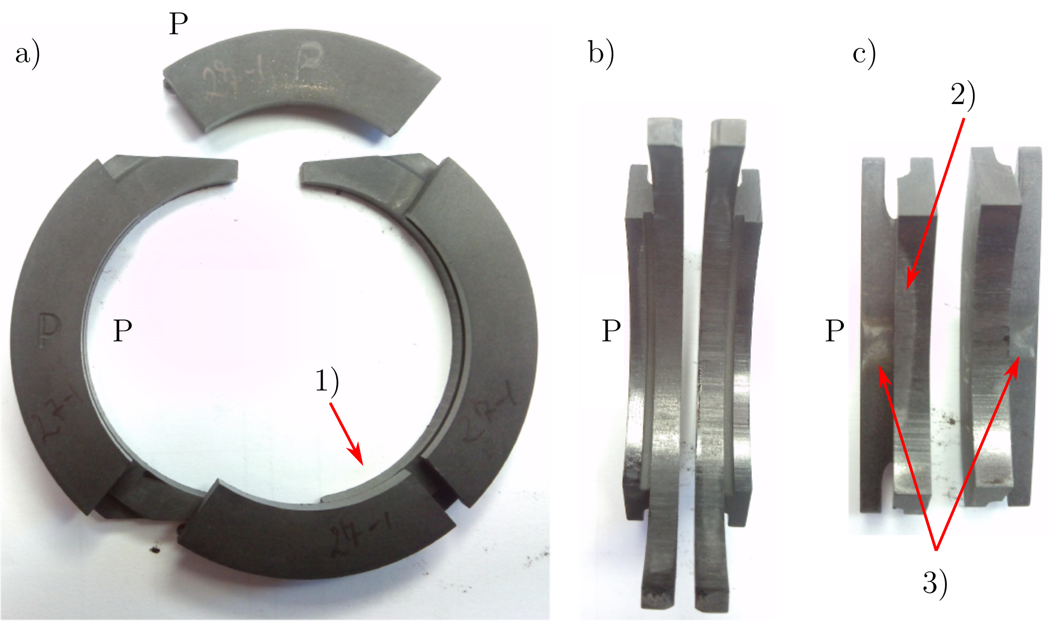


Figure 6.17. Worn ring A1: a) shows the worn ring form the pressure side, note that two segments are sticking together at position 1). To distinguish between equal segments, the "P"-segments are marked, according to the convention presented above, see Figure 6.4. In b) and c) the dynamic contact surfaces are depicted for the sealing segments and the cap segments, respectively. The contact area of the sealing segments is very smooth and of homogeneous appearance. The P-cap segment in c) shows inhomogeneous wear. A region appearing brighter is marked with 2). On the caps pressure marks of the sealing segments can be seen, 3). The area of the open wear gap appears brighter and seems to be covered in either debris or dirt.

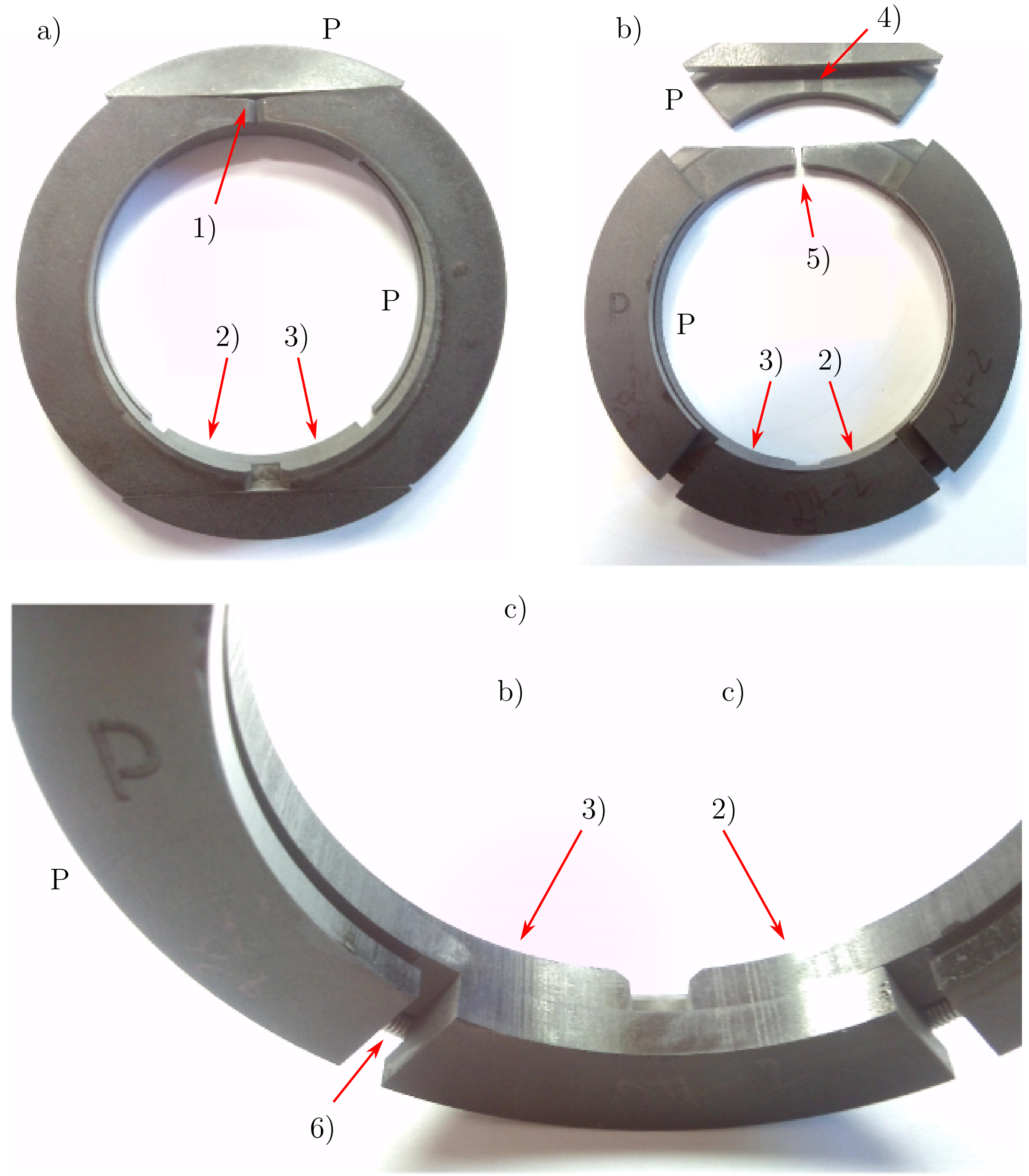


Figure 6.18. Worn ring A2. To distinguish between equal segments, the “P”-segments are marked, according to the convention presented above, see Figure 6.4. a) shows the ring from the crank side, the garter spring is still mounted. A gap between the P-cap segment and the sealing segments is visible, 1). The sealing segments are touching only in one place, since they are sticking to the cap segment 2) and 3). In b) the garter spring is removed and the P-cap segment is taken from the other segments. It shows pressure marks of the sealing segments. The wear gap 4) is visible between the pressure marks. It is (naturally) bigger than the wear gap of the unmounted, unloaded wear gap at 5). Picture c) was taken before dismounting the garter spring (similar to a). The spring is visible at 6). The garter spring force is not sufficient to separate the sticking segments in 2) and 3).

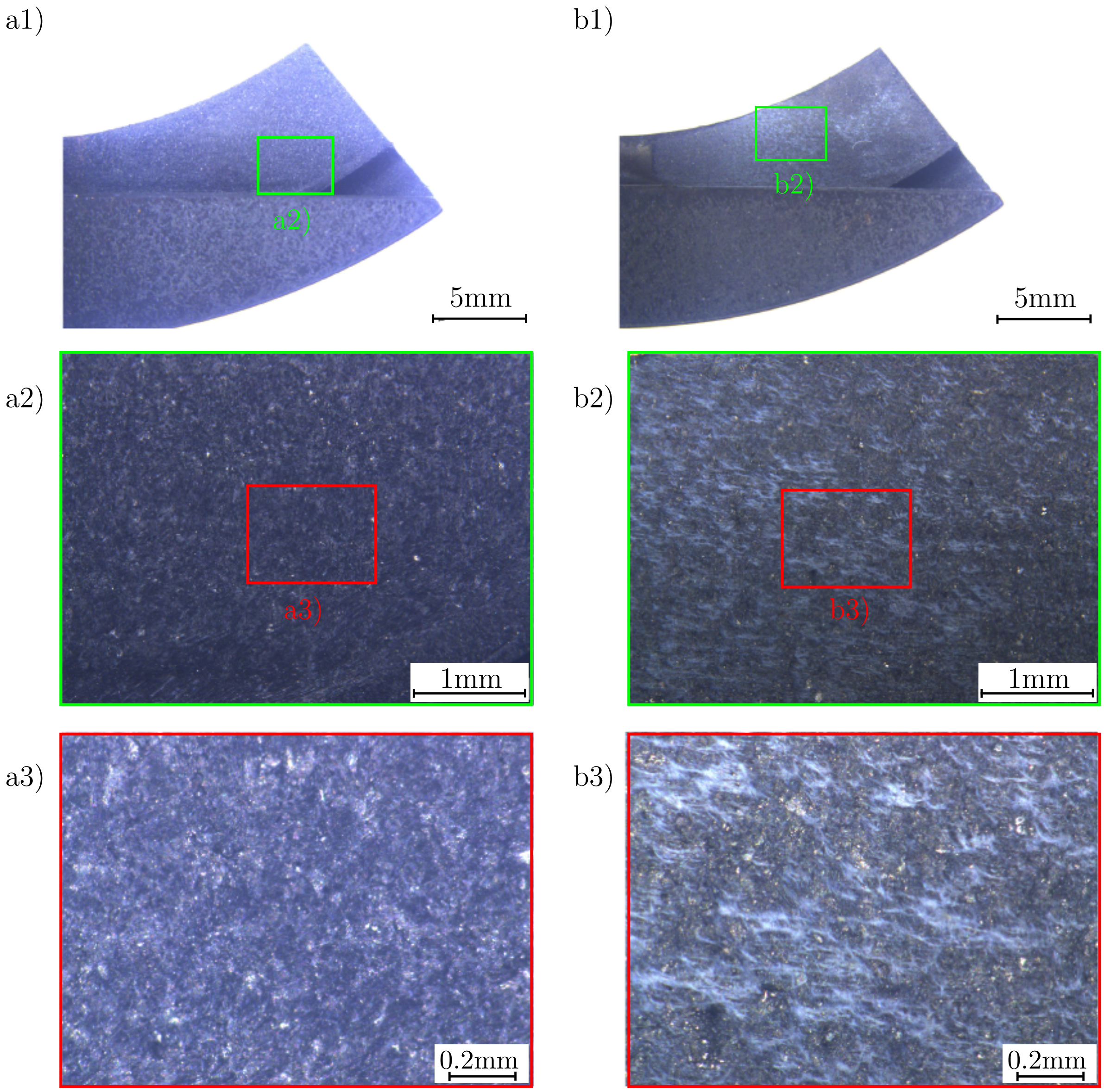


Figure 6.19. Microscopy of a cap segment after test A. The left pictures a1) to a3) show a cap segment in different magnifications. No sticking was present between cap and sealing segment at the test end. The right pictures b1) to b3) show a cap surface after the corresponding sticking sealing segment has been broken away. The higher magnifications b2) and b3) show that the surface is covered with thin white fibrils. It is assumed that these are PTFE fibrils.

The wear measurement results in the Figures 6.20 and 6.12 for ring A1 and A2, respectively. The tables 7.1 and 6.1 give detailed data of the ring segment's wear, the former is in the appendix, the latter in section 6.2.1. Although the photographs of A1 (Fig. 6.17) indicate that a certain section of the P-cap segment is worn differently, this cannot be seen in the wear measurement. It is interesting that, although the rings failed and sealed repeatedly the wear profile is very smooth and shows the expected pattern: The maximum appears at the sealing segment's ends and in the circumferential middle of the cap segments. Nevertheless this test is not fit to be compared to a steady-state simulation, since the rod and thus the material temperature changed frequently and over a great range.

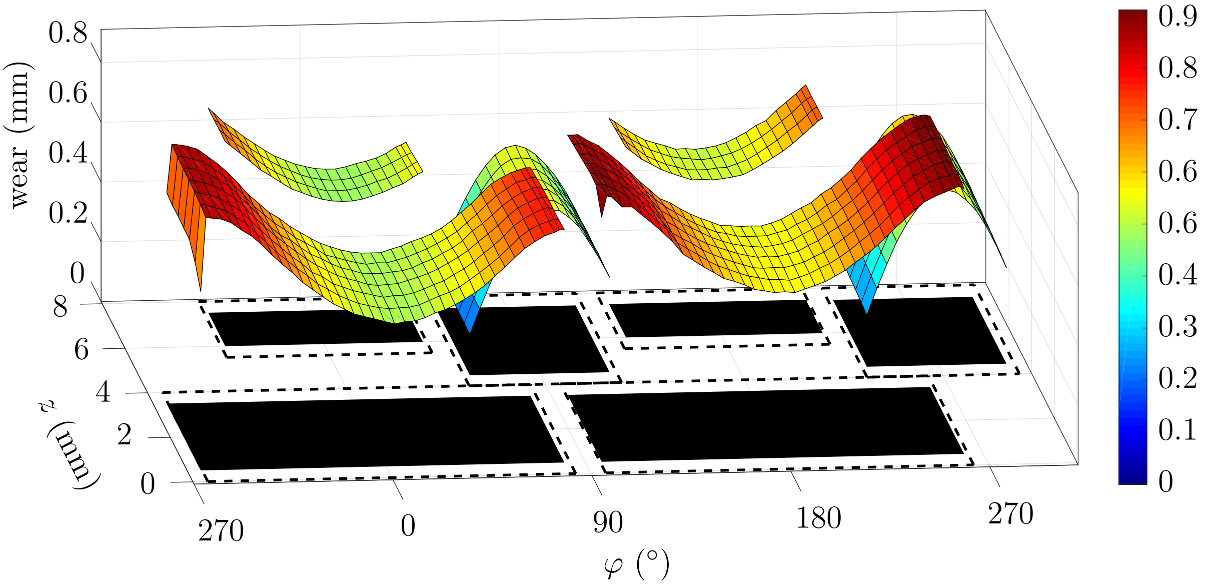


Figure 6.20. Wear of the ring A1.

The fact that a packing consists of several rings indicates that such a behaviour of sealing and failing appears independently from the material, design and manufacturer. Figure 6.21 summarizes a theory, which explains this process of repeated sealing and failing. After starting a test/compressor the rings seal and thus the temperature of rod and ring increases. With increasing temperature the elastic modulus and the structural stiffness decrease. This leads to higher contact forces since a more compliant structure carries less load. The frictional force and thus the frictional heating rises along with the contact forces, leading to a further increase in temperature. This cycle self-accelerates until a quasi-stationary temperature level is reached. The material fails and the ring extrudes between rod and backup ring/cup and is consumed by wear or until the ring segments start to stick together.
It is assumed that a certain material dependent combination of temperature, contact pressure and time allows the ring segments to stick together. If these conditions are met, the ring segments stick together, the ring's structural stiffness increases significantly and allows to carry more of the gas pressure load than before. This reduces the contact force and with it the frictional heating, resulting in a decrease in rod and ring temperature. Figure 6.16 shows clearly: the rod temperature increases between 1) and 3) and then slowly decreases until 4). The temperature decreases more slowly than it has previously increased. This is explained as follows: As soon as the segments stick together the structure carries an ever increasing portion of the load as it slowly wears away. If the sticking segments break loose again, they would geometrically compensate the wear, just like they

are designed to do. Here the segments stick firmly enough together and cannot compensate wear. The contact pressure decreases and thus the temperature drops from 190 °C to 170 °C.

In contrast to the previous (thermally upwards) cycle this cycle has the potential to enormously self-accelerate. If the contact pressure is reduced enough due to ring wear a leakage path may form. This gas flow cools ring and rod. The colder the ring becomes the bigger the leakage path may become, which accelerates the cooling further. Thus at a certain point, Figure 6.16 position 4), the rod temperature drops within minutes below the lower measurement limit 75 °C. Here in Figure 6.16 at 4) it took less than 4 min for the rod temperature to drop from 150 °C below 75 °C.

Finally with the low rod temperature, ring and rod are not conform any more and high leakage occurs. The ring fails sealing, but over time conformity can again be achieved due to wear, closing leakage paths until the ring seals again. Over time the sticking segments may also break loose again. This is then a starting point for a next cycle.

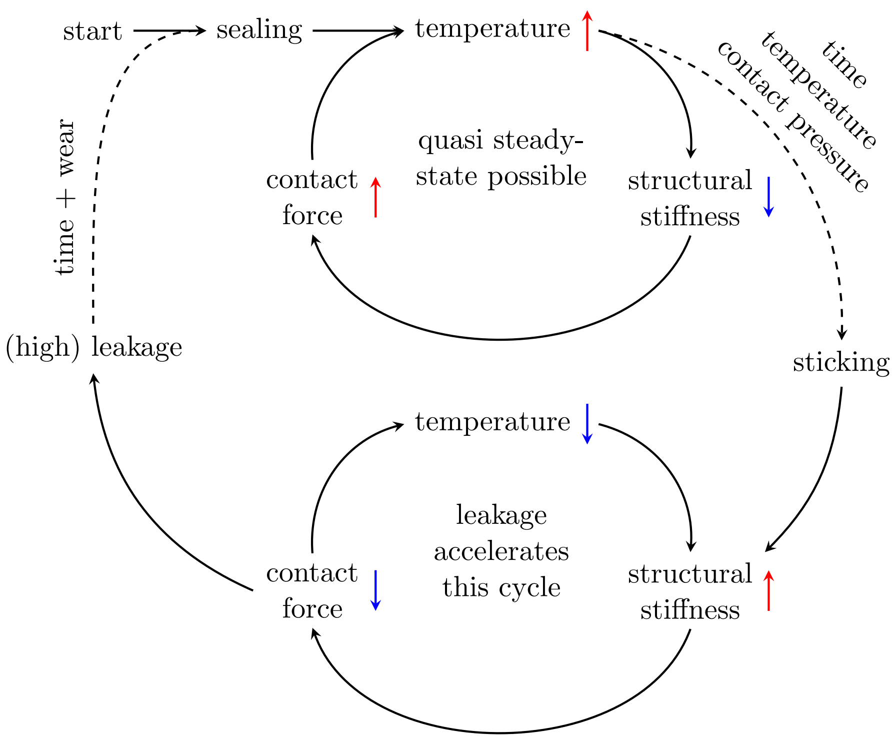


Figure 6.21. Possible life scheme of a ring where the segments start sticking together. The solid lines show cogent correlations. The dashed lines show possible correlations. A proper ring seals at the beginning and enters a loop: it heats up, loses structural stiffness, which increases the contact forces and again the temperature. This cycle may stabilize into a quasi steady-state. A combination of temperature, contact pressure and time can lead to sticking, breaking the quasi steady-state. Sticking increases the structural stiffness, thus decreases the contact forces leading to lower temperatures and again to higher structural stiffness. This cycle accelerates if leakage occurs. If a ring cools down rapidly the ring usually fails to seal due to thermally induced strains, which open leakage paths. Nevertheless the ring can start sealing again, if time and wear create conformity of ring and rod.

6.3.2 Test B

Test B has been performed with a gas pressure of 30 bar, a mean rod speed of 1.2 m s^{-1} and with bone-dry nitrogen. Table 6.3 shows the test parameters.

Table 6.3. Test B - test parameters

parameter	value	unit
gas type	nitrogen	
gas dryness	bone-dry	
pressure	30	bar
average rod speed	1.2	$\mathrm{m\,s^{-1}}$
ring material	HY54	
backup material	bronze	
rod inside cooling	yes	
static leakage test at 6 bar		
ring B1	17	$\mathrm{l\,min^{-1}}$
ring B2	12	$\mathrm{l\,min^{-1}}$

Figure 6.22 presents the averaged rod and packing temperature, the leakage for both rings and the gas pressure over time. The temperatures are averaged over the rod length or the whole packing housing. The test duration is 359 h. The test shows an average rod temperature of 84 °C. The temperature profile is very constant over the whole test time. The maximum rod temperature is 94 °C. During the first 285 h the rings sealed well, the leakage is always below 60 $\mathrm{l\,min^{-1}}$ for both rings. Then the rings start to lose sealing performance. On average the leakage goes up once every 83 min in the last 75 h. Figure 6.23 is a zoom of the grey highlighted area of Figure 6.22. Four losses in sealing performance between the test time of 325 and 330 h have been monitored. A measurement point has been taken every 3 min. Each leakage is represented by a single data point only. These events are very short and no change in the rod temperature is measured, nonetheless the pressure drops according to the increased leakage.

Figure 6.24 presents images of the disassembled rings after test B. The rings show no sticking. The cap segments show pressure marks of the sealing segments. The dynamic sealing surfaces appear homogeneous and smooth. The Figures 6.25 and 6.26 show the results of wear measurement for the rings B1 and B2, respectively. The tables 7.2 and 7.3, in the appendix, give detailed data of the ring segment's wear. In comparison to ring B1, the ring B2 shows a very symmetrical wear pattern. The P-cap segment of ring B1 is nearly virgin, the maximum wear is 0.1 mm and the average radial wear is 0.08 mm, whereas the other cap segment shows 0.35 mm average radial wear, with a maximum in the segment's middle of 0.62 mm. In comparison the cap segments of ring B2 show a maximum wear of 0.74 mm (P-cap) and 0.79 mm (other cap segment). This indicates that the sealing segments were sticking slightly to the P-cap segment of B1. Thus it is assumed that the ring B1, especially the region of the P-cap segment is the reason for repeated short-term leakage. A slight sticking is assumed since the ring starts to seal again quickly after leaking.

Ring B2 is suitable for a comparison with the developed calculation. The rod and packing temperatures stayed constant over the test duration and no signs of sticking or other symmetry breaking effects are visible. The comparison follows in section 6.4.2.

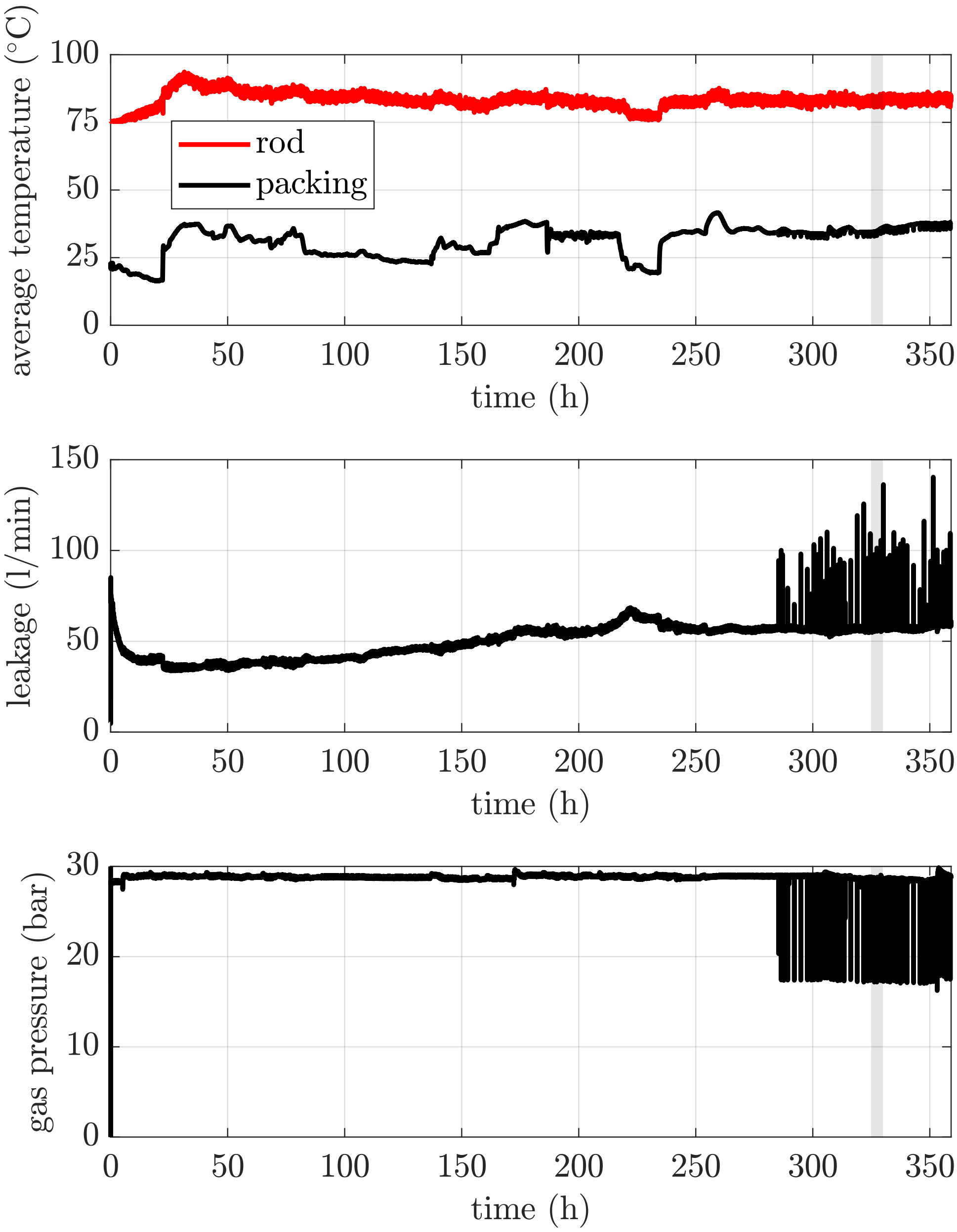


Figure 6.22. Protocol of the averaged rod and packing temperature, the leakage of both rings and the gas pressure over time for test B. Figure 6.23 presents a zoom into the grey highlighted area.

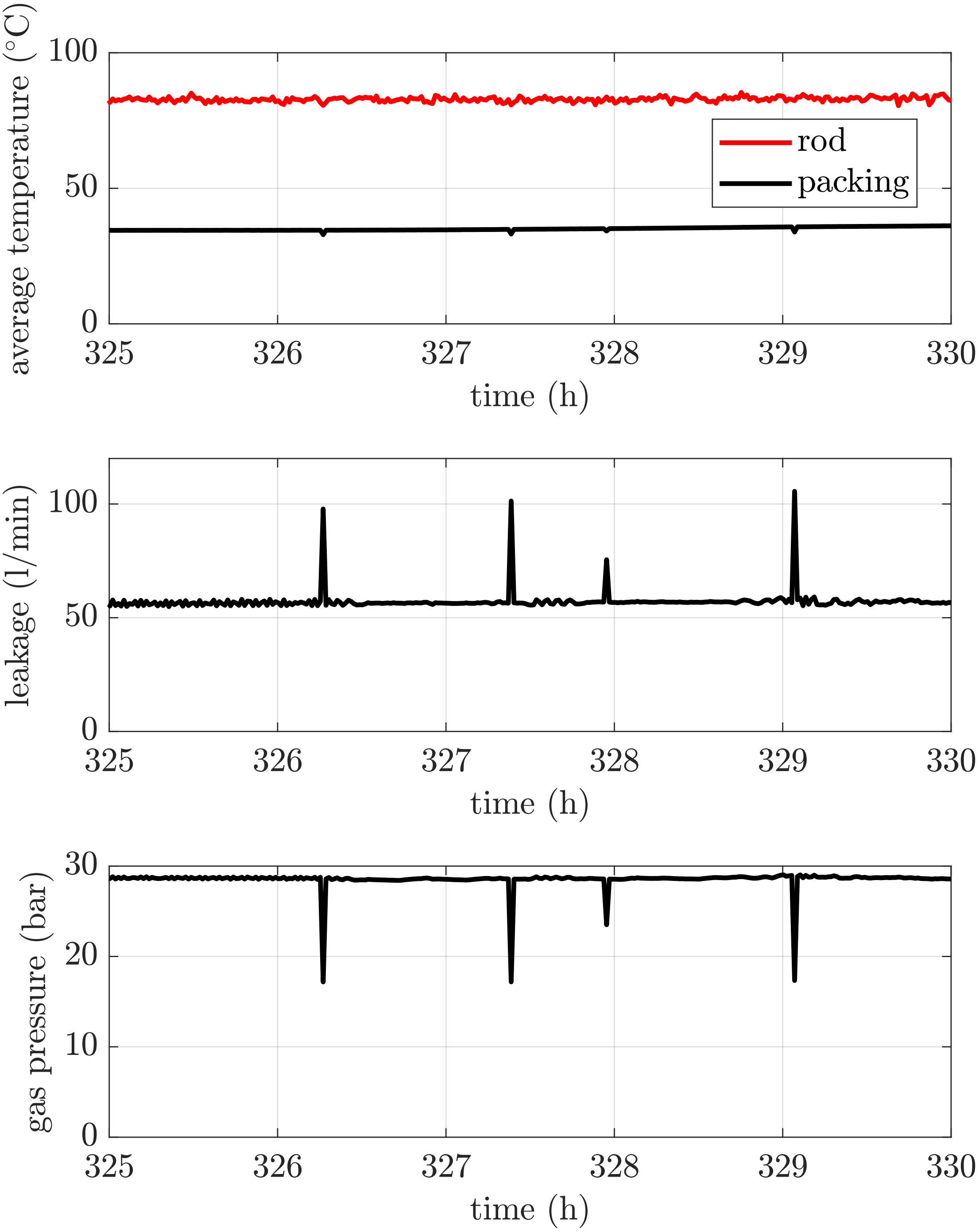


Figure 6.23. Zoom from Figure 6.22. The rings fail four times for a very short period, each time an according pressure drop is monitored. The rod temperature stays constant since the rings start to seal again quickly after leaking.

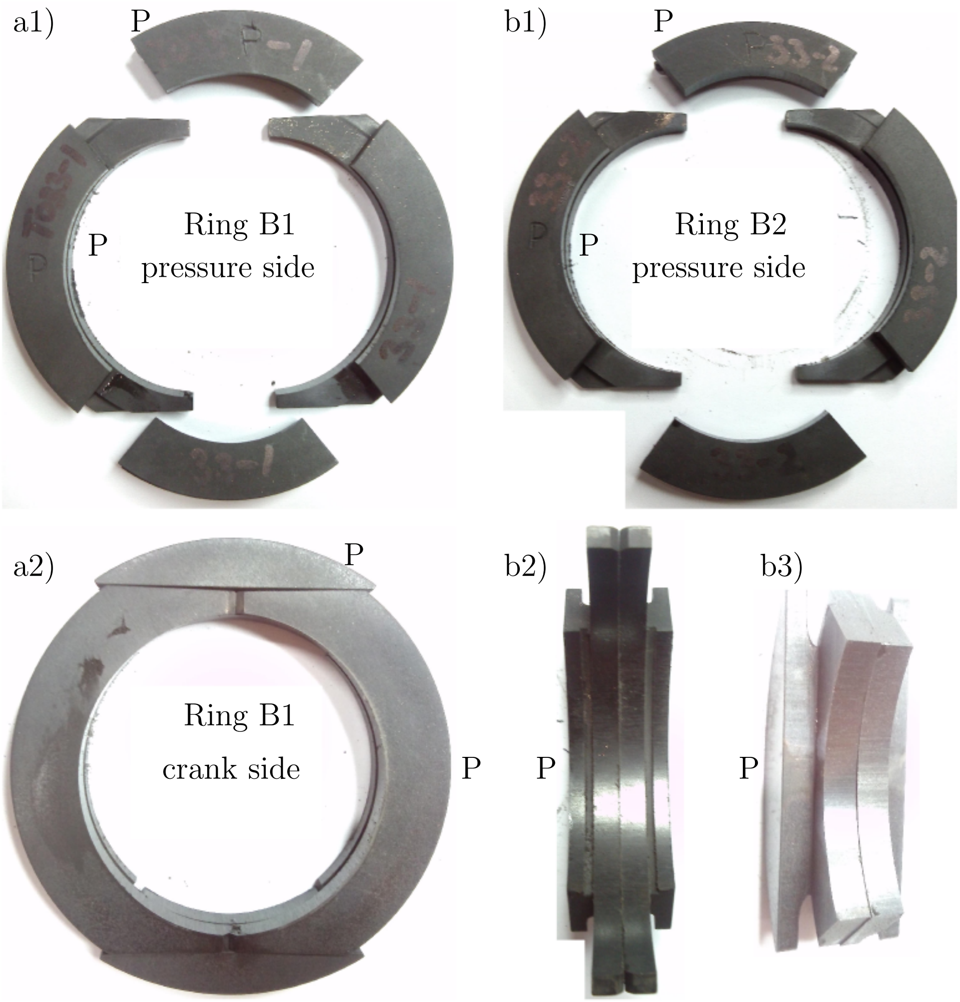


Figure 6.24. Worn rings B1 and B2, a) and b), respectively. A “P” marks the P-segments, according to the convention in Figure 6.4. The dynamic contact surface appears smooth and homogeneous. Their ring segments were not sticking together when the test ended and the rings were unmounted. Nonetheless a pressure mark of the sealing segments can be seen on the cap segments in b3). [35]

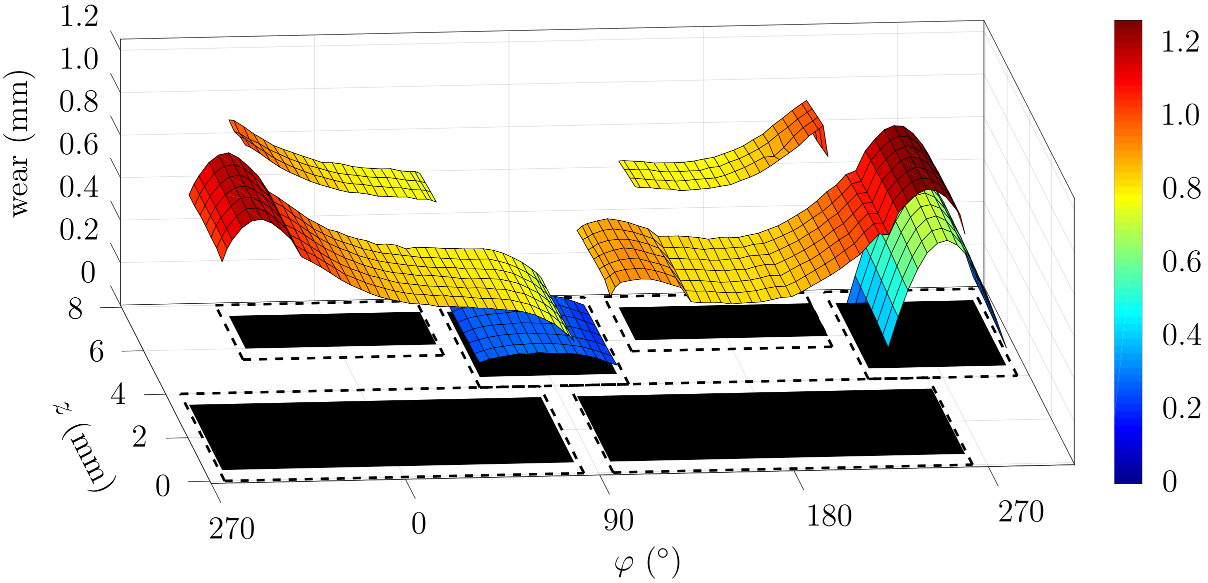


Figure 6.25. Wear of the ring B1. The ring shows an irregular pattern. The P-cap segment is nearly in its virgin condition, with a maximum wear of 0.10 mm. This indicates that the sealing segments were sticking to the P-cap segment.

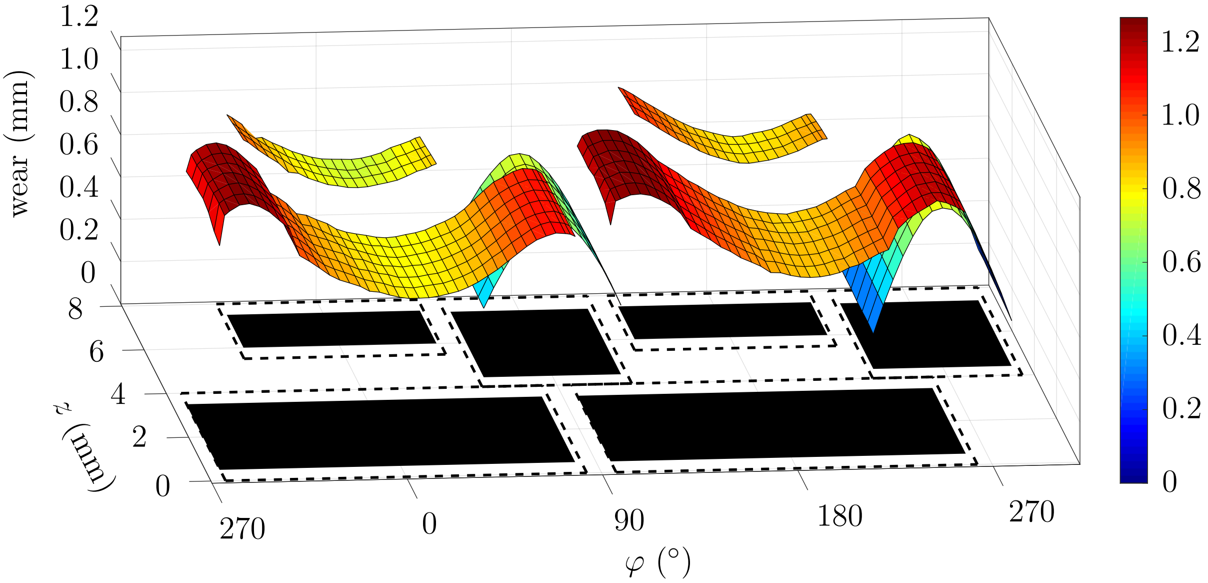


Figure 6.26. Wear of the ring B2. The segments are equally worn. The pattern looks as expected and is used for comparison with a calculation.

6.4 Comparison of experiment and calculation

6.4.1 Frictional force

On the testing device presented in 4.2 a pressure inlet in the T-cup allows to fill the T-cup with pressurized gas (i.e. air). So packing rings can be tested when sealing a pressure differential. Such a test allows to measure the effect of the gas pressure distribution in the dynamic sealing surface and to verify the numerically correct implementation.

6.4.1.1 Measurement

The BCD rings are mounted on the rod after cleaning with ethanol. The force is set to zero while the rod with the mounted packing rings is hanging freely from the load cell. Then the packing is fully assembled and pressurized. A certain initial pressure differential eases the initial sealing, thus the high pressure levels are measured first.
Before the first test run starts, the rod is manually moved up and down for several millimetres to ensure that the rings are sealing. The measurements are performed similar to the situation in a real packing. To allow the pressurized rings to properly adjust to their duty, they run several cycles with maximum speed (1000 mm min^{-1}). Then the series of measurements starts with a stroke of 200 mm and at least 4 cycles, a speed of 500 and 1000 mm min^{-1} and different pressure levels between 0 and 10 bar, Figure 6.27, 6.28 and 6.29. A manometer before the inlet measures the gas pressure behind the pressure control valve.

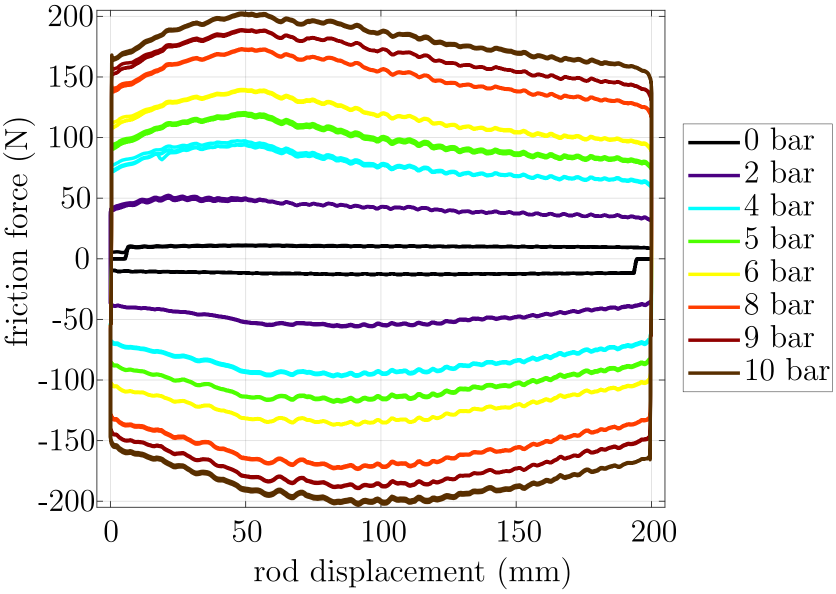


Figure 6.27. Frictional force measurement with a BCD ring (under given load) on the steel rod at a relative velocity of 500 mm min^{-1}. Each pressure level was tested with 4 cycles.

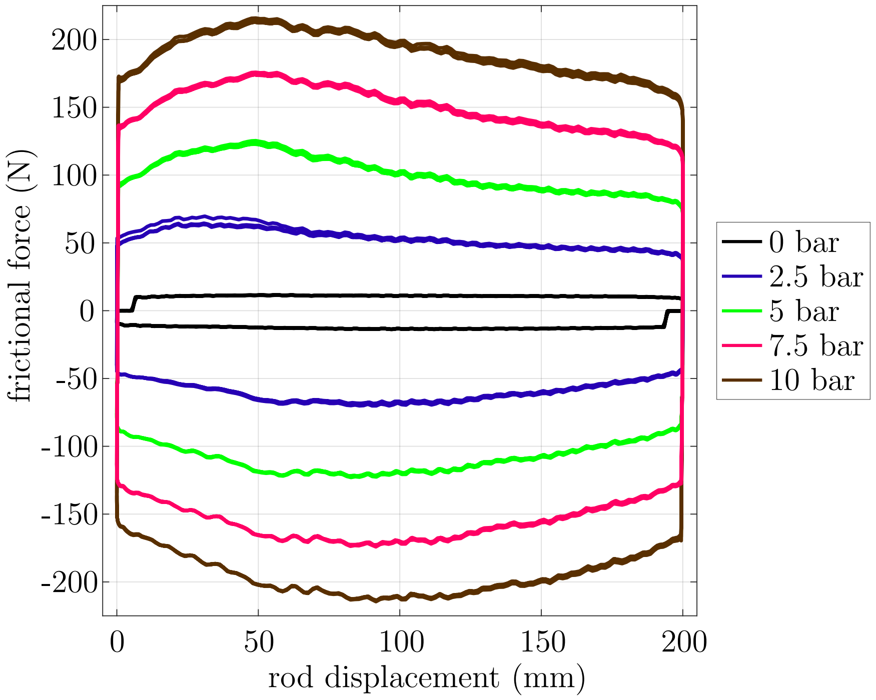


Figure 6.28. Friction force measurement with a BCD ring on the steel rod at a relative velocity of 1000 mm min^{-1}. For each pressure level 4 cycles were run.

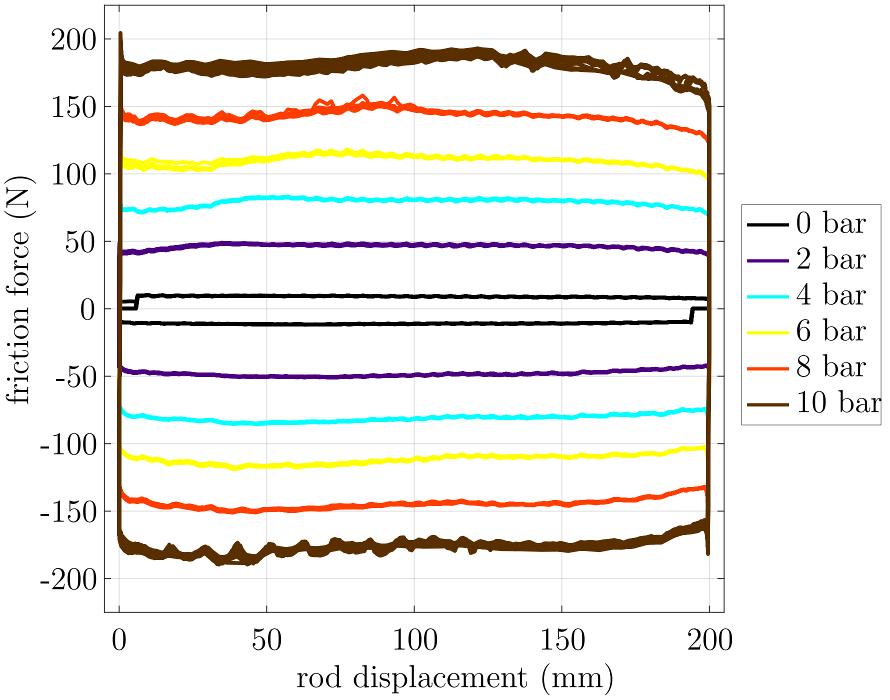


Figure 6.29. Friction force measurement with a BCD ring on the WC rod at a relative velocity of 500 mm min^{-1}. For each pressure level a minimum of 5 cycles were run.

Table 6.4 presents the test results and Figure 6.30 the resulting frictional forces. The coefficient of friction is higher on the steel rod than on the WC rod and increases with relative velocity.

The contact force F_C for the ring is calculated from the measured frictional force F_F and the coefficient of friction μ. F_C allows to compare the different measurement series, Figure 6.31. Note that F_C is independent of the tribological system, hence different ring designs can be compared even if the tribological system is unknown. Through this representation of the data the effect of the pressure load and the gas pressure distribution in the dynamic sealing surface already contains all the microscopic effects of any tribological system.

A linear fit represents all data points very well. This indicates that the gas pressure distribution is equally present for each measurement series. Any deviation from the fit results mainly from the manual pressure regulation.

Table 6.4. Contact force F_C from a single BCD ring on a 50.8 mm rod, depending on the gas pressure.

material	relative velocity (mm/min)	μ	pressure (bar)	F_F (N)	F_C (N)
steel HY54	500	0.205	10	92.6	452
	500	0.205	9	85.6	418
	500	0.205	8	77.7	379
	500	0.205	6	60.7	296
	500	0.205	5	51.4	251
	500	0.205	4	41.3	202
	500	0.205	2	22.9	112
	500	0.205	0	5.6	27
	1000	0.208	10	97.5	466
	1000	0.208	7.5	77.5	371
	1000	0.208	5	53.3	255
	1000	0.208	2.5	28.8	138
	1000	0.208	0	5.9	28
WC/Co/Cr HY54	500	0.19	10	87.7	469
	500	0.19	8	71.9	378
	500	0.19	6	55.3	291
	500	0.19	4	40.1	211
	500	0.19	2	23.7	125
	500	0.19	0	5.0	26

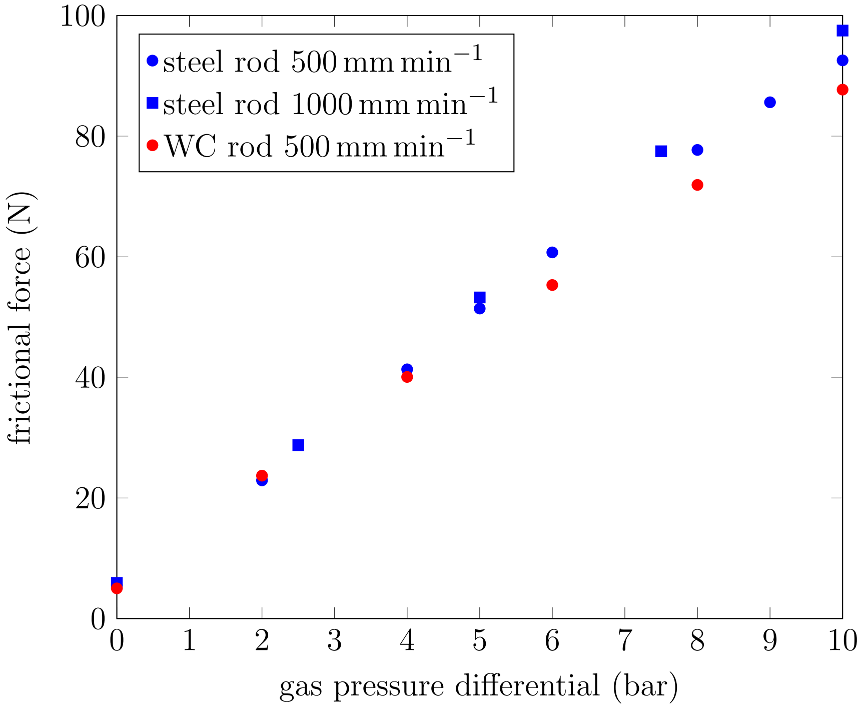


Figure 6.30. Frictional force F_F of a single BCD ring on a 50.8 mm rod, depending on the sealed gas pressure differential.

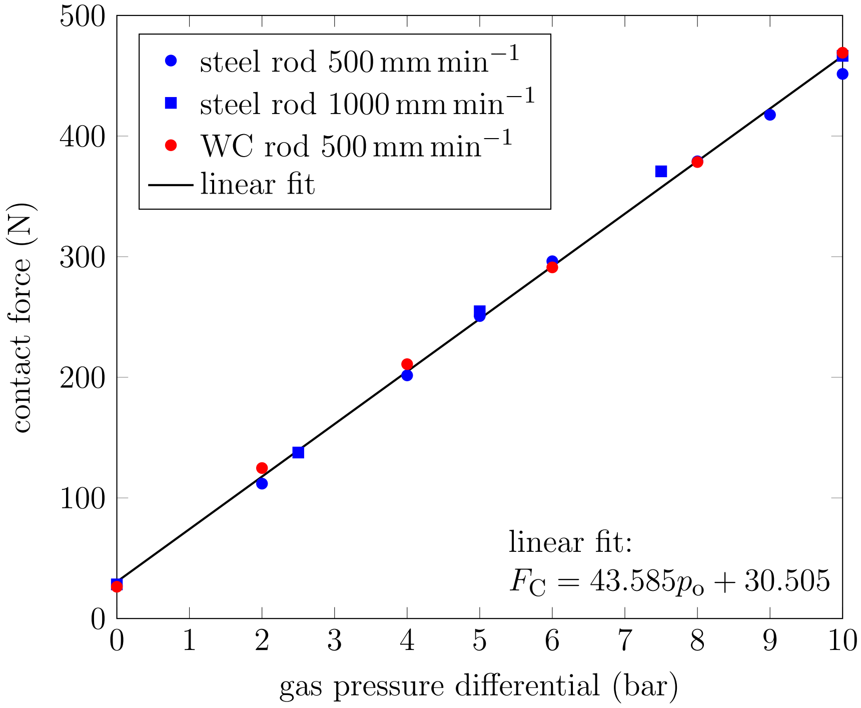


Figure 6.31. Contact force F_C from a single BCD ring on a 50.8 mm rod, depending on the gas pressure differential. The linear fit approximates all data points very well.

6.4.1.2 Comparison with calculation

The numerical calculations presented in section 5.5.4 show the effect of the incompressible or compressible gas pressure distribution in the dynamic sealing surface.

The simulation including the compressible gas pressure distribution in the dynamic sealing surface is compared to the experimentally determined contact force. Since the effect of the garter spring is small, it is disregarded in the calculation. For a comparison the experimental contact force results are reduced by the contact force resulting from the garter spring. The calculation is in excellent agreement with the experiment, measured in the 0 bar run.

This allows to compare the results from the numerical simulation with the experiments performed, see Figure 6.32. The excellent agreement between experiments and calculation validates the formulation of the compressible gas pressure distribution in the dynamic sealing surface by Lindner-Silwester [43] and its numerical implementation.

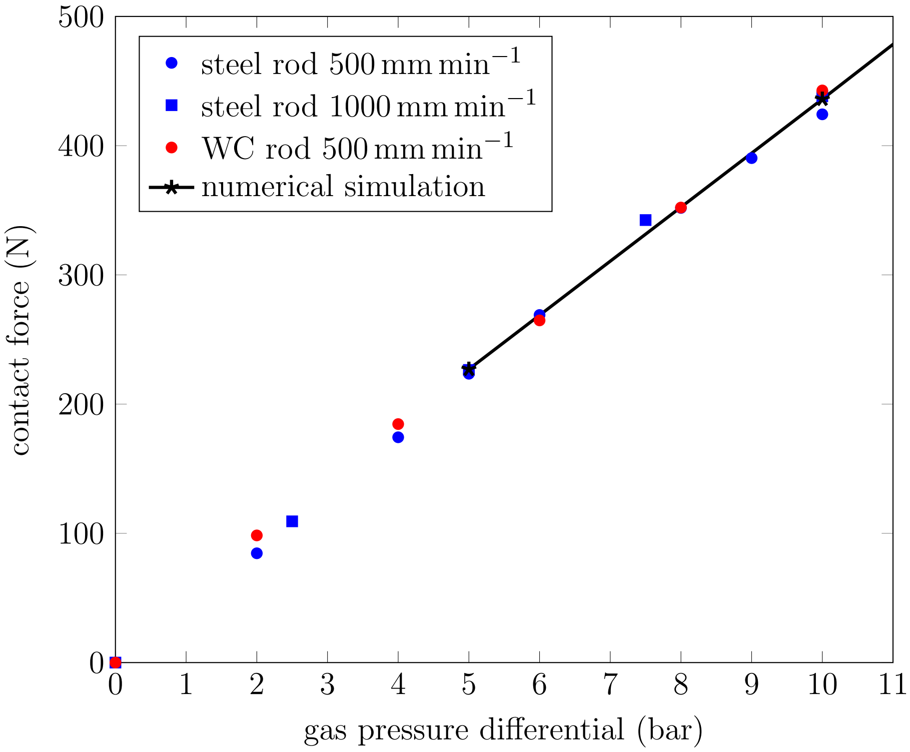


Figure 6.32. Comparison of simulation and experiment: Contact force F_C from a single BCD ring on a 50.8 mm rod, depending on the sealed gas pressure differential. The experimental data is corrected by the contact force resulting from the garter spring load.

Now the frictional force of a single BCD on a WC rod with Ø=50.8 mm is calculated dependent on the sealed pressure differential. The crank end pressure is ambient. The determined coefficient of friction 0.19 leads to the forces in Figure 6.33.

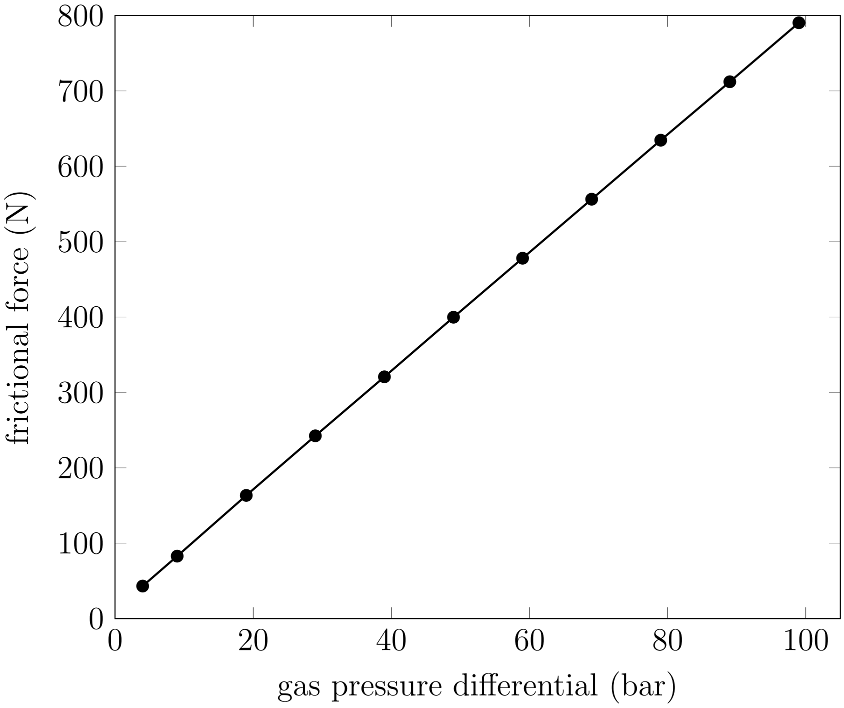


Figure 6.33. Frictional force F_{F} for a single BCD ring on a WC rod with Ø=50.8 mm, depending on the sealed gas pressure difference. The gas pressure distribution is calculated with compressible gas. The coefficient of friction is 0.19.

6.4.2 Wear

The results presented in this section have been published in [35].

A comparison of the wear of a packing ring between calculation and experiment is only possible if the wear coefficient is known and can be inserted into the calculation. During the experiment, section 6.3.2 (Test B), two BCD rings where loaded with 29.8 bar for 359 h (15 days). The mean rod speed was 1.19 m s^{-1}. The maximum temperature during the test was 94 °C, the average temperature 84 °C.
In a tribological test (see section 4.1.4, Table 4.3) the coefficient of wear is determined at 80 °C giving $k = 7.2 \cdot 10^{-7}\,\mathrm{mm^3\,km^{-1}\,N^{-1}}$, which is used as an input parameter for the calculation.
Figure 6.26 shows the wear of the ring segments. From these results the total wear volume of each segment can be computed. The up-scaled wear data for the segments is again given in Table 6.5. These measurements are in good agreement with the simulation results corresponding to the same wear time of 15 days. The calculation yields a wear volume of 466 mm^3 for the sealing segment and 92 mm^3 for the cap segment. These values are between 13 and 21 % higher than the measured ones. This deviation shows that the wear coefficient used in the calculation was slightly too high.
In experiments and in calculations the wear pattern does not significantly change in axial direction, see e.g. Figure 6.26 and Figure 5.19, respectively. Figure 6.34 from [35] gives

a wear comparison at the nodal axial positions 2.08 and 6.44 mm from the cup face (as explained in Figure 5.20). Calculation and measurement are in very good agreement, which confirms the predictive power of the developed simulation model. It is important to note that the coefficient of wear k has been determined in an independent experiment and the calculation is free of adjustment or fitting parameters.

Table 6.5. Measured (and up-scaled) wear volumes for the segments of a BCD ring after a 359 h test and the calculation result.

segment type	position	total wear volume	
		experiment	calculation
	(°)	(mm^3)	(mm^3)
seal	0	368	466
cap "P"	90	80	92
seal "P"	180	398	466
cap	270	76	92
all		921	1118

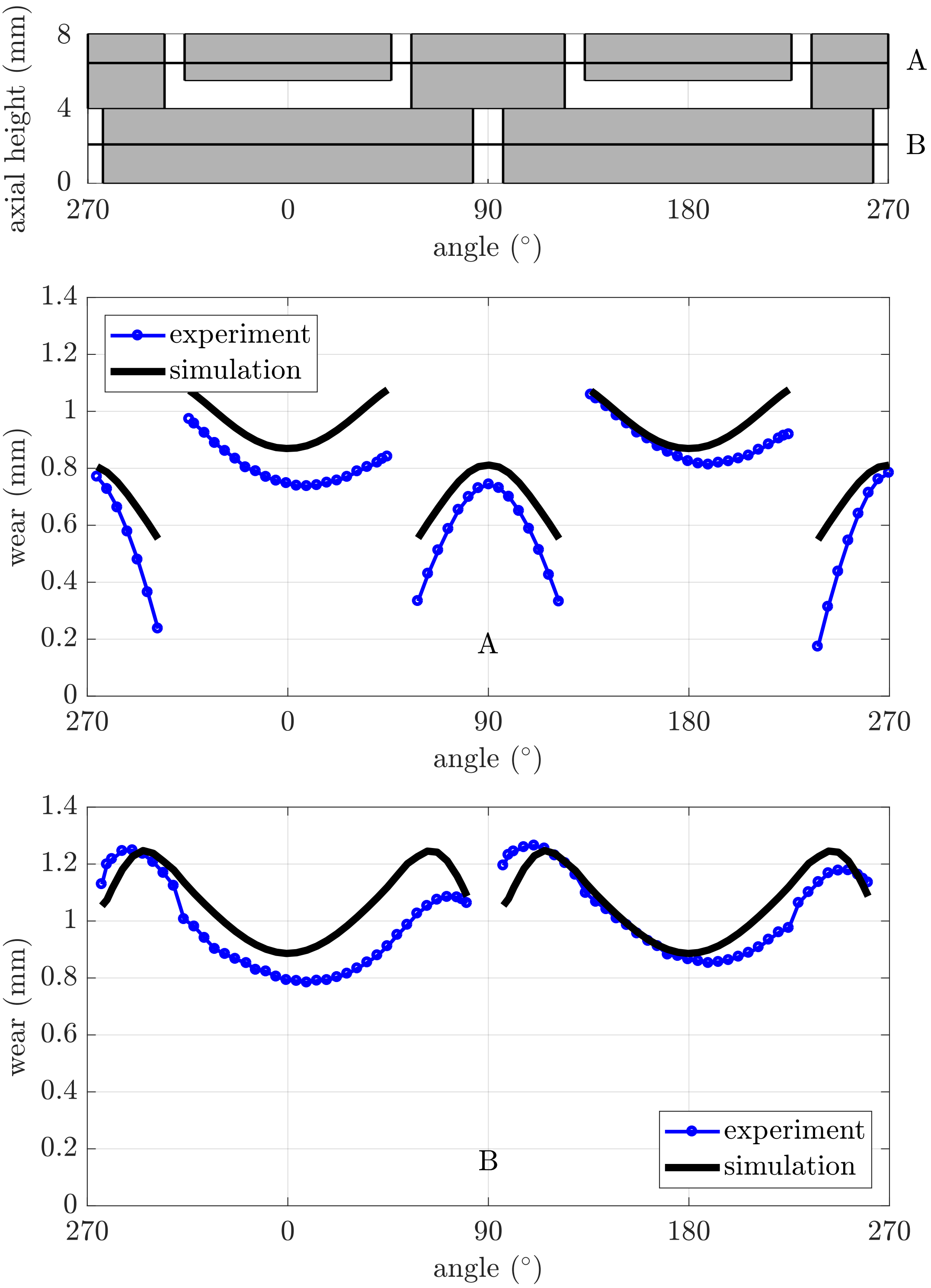


Figure 6.34. Calculated and measured wear for a BCD ring [35]. Top: Sketch of the ring's dynamic sealing surface and the paths A and B at which the comparison is made. Middle: Path A, representing the cap segment of the BCD ring. Bottom: Path B, representing the sealing section of the sealing segment of the ring.

Chapter 7

Conclusion and Outlook

The presented calculation model for packing ring wear bridges for the first time the gap between laboratory testing and real-world applications.
The calculation procedure is able to handle the wear of packing rings up to several millimetres and gives insight into the wear process. It further allows to monitor quantities, which are hardly or not at all measurable, e.g. stresses, strains, contact pressures, the wear pattern and the evolution of these quantities.
The numerical implementation was validated against an analytical model, derived for a simplified ring configuration. The developed theory, the assumptions and the model of the ring wear were successfully validated against measurements performed on purpose-developed test equipment. All material parameters were obtained from independent tests. In none of the comparisons an adjustment parameter is used. The agreement between the frictional-force measurements and the calculation is excellent. The agreement between the wear tests and the calculation is good, qualitatively and quantitatively. This shows the predictive power of the simulation and opens windows of opportunity to gain a sound understanding of packing ring wear.

This calculation procedure can now be used i) to benchmark ring designs without the unpredictability of any tribologic system, setting the wear coefficient and all other boundary conditions equally. ii) to investigate how symmetry breaking effects like ring segments sticking together or rod-cup misalignments influence the ring wear. iii) to investigate how stopping and restarting a compressor effects the packing rings. To this end a material model of higher complexity will be needed. iv) as part of the simulation of a whole packing case with several interacting rings. Here further calculations have to be implemented.

This work lays the basis for future computer-aided ring designs improving the efficiency of compressors.

Appendix

HY54

HOERBIGER
because performance counts

The properties of HY54 provide a very versatile material grade that has been successfully applied in a multitude of applications, with gas conditions from wet through to dry, in both lubricated and non-lubricated service. This material has shown itself to be a broad range problem solving material where standard carbon filled PTFE materials do not give desired lifetime. Successful experience in compressor applications covers air, methane, ethylene, propylene, helium, hydrogen, and nitrogen and other mixed gas process streams.

Air

Industrial Gases

Natural Gas

Refinery

Olefins

Alcohols

Chemicals

Refrigeration

Physical Properties

Property	Method	Value
COTE - Radial x 10-6/C (20-200 °C)	ASTM D696	73
COTE - Axial x 10-6/C (20-200 °C)	ASTM D696	107
Maximum operating temperature (°C)	-	200
Density (g/cm3)	ASTM D792-00	2.07 ± 0.05
Shore D Hardness	ASTM D2240-04	64 ± 2
Tensile strength at break (MPa)	ASTM D638-03	15.2 ± 0.9
Elongation at break (%)	ASTM D638-03	86.4 ± 0.5

Operating range

Max. Gas Temperature (°C)		Max. Pressure (bar)			
Discharge	Design	Packing Discharge		Cylinder Ring Diff.	
		Non-Lube	Lube	Non-Lube	Lube
200	150	100	175	50	100

Operating restriction for oxygen-service: Compression ratio up to 3 and max. temperature 225°C

All values are approximate and subject to change without notification.
The maximum material design temperature is calculated by considering suction and discharge conditions, machine speed, cooling and loading. Typically: $T_{design} = T_{suction} + \frac{2}{3}(T_{discharge} - T_{suction})$.
Additional operating conditions need to be considered when making material selections.
The data presented are guidelines only; consult HOERBIGER to ensure the correct material is specified.

Issue 2015 04

Table 7.1. Detailed wear and wear volume data for ring A1.

variable	value	unit
area data		
theoretical area virgin state cap segment	138.3	mm^2
theoretical area virgin state sealing segment head end	103.1	mm^2
theoretical area virgin state sealing segment crank end	295.2	mm^2
total theoretical BCD area virgin state	1073.3	mm^2
measured area cap segment P	89.1	mm^2
measured area sealing segment P head end	60.2	mm^2
measured area sealing segment P crank end	220.6	mm^2
measured area cap segment	89.0	mm^2
measured area sealing segment head end	60.5	mm^2
measured area sealing segment crank end	221.3	mm^2
wear volume data		
measured wear volume cap segment P	37.57	mm^3
measured wear volume sealing segment P head end	34.65	mm^3
measured wear volume sealing segment P crank end	156.37	mm^3
measured wear volume cap segment	41.99	mm^3
measured wear volume sealing segment head end	32.39	mm^3
measured wear volume sealing segment crank end	141.59	mm^3
up-scaled wear volume cap segment P	58.30	mm^3
up-scaled wear volume sealing segment P head end	59.28	mm^3
up-scaled wear volume sealing segment P crank end	209.25	mm^3
up-scaled wear volume cap segment	65.24	mm^3
up-scaled wear volume sealing segment head end	55.17	mm^3
up-scaled wear volume sealing segment crank end	188.92	mm^3
total measured wear volume	444.57	mm^3
total up-scaled wear	636.17	mm^3
radial wear data		
mean radial wear cap segment P	0.421	mm
mean radial wear sealing segment P head end	0.575	mm
mean radial wear sealing segment P crank end	0.709	mm
mean radial wear cap segment	0.472	mm
mean radial wear sealing segment head end	0.535	mm
mean radial wear sealing segment crank end	0.640	mm
mean radial wear	0.600	mm

Table 7.2. Detailed wear and wear volume data for ring B1.

variable	value	unit
area data		
theoretical area virgin state cap segment	138.3	mm^2
theoretical area virgin state sealing segment head end	103.1	mm^2
theoretical area virgin state sealing segment crank end	295.2	mm^2
total theoretical BCD area virgin state	1073.3	mm^2
measured area cap segment P	89.6	mm^2
measured area sealing segment P head end	60.4	mm^2
measured area sealing segment P crank end	221.1	mm^2
measured area cap segment	89.9	mm^2
measured area sealing segment head end	59.8	mm^2
measured area sealing segment crank end	219.4	mm^2
wear volume data		
measured wear volume cap segment P	6.85	mm^3
measured wear volume sealing segment P head end	45.97	mm^3
measured wear volume sealing segment P crank end	197.45	mm^3
measured wear volume cap segment	31.51	mm^3
measured wear volume sealing segment head end	45.32	mm^3
measured wear volume sealing segment crank end	185.65	mm^3
up-scaled wear volume cap segment P	10.57	mm^3
up-scaled wear volume sealing segment P head end	78.47	mm^3
up-scaled wear volume sealing segment P crank end	263.68	mm^3
up-scaled wear volume cap segment	48.51	mm^3
up-scaled wear volume sealing segment head end	78.09	mm^3
up-scaled wear volume sealing segment crank end	249.82	mm^3
total measured wear volume	512.75	mm^3
total up-scaled wear	729.14	mm^3
radial wear data		
mean radial wear cap segment P	0.076	mm
mean radial wear sealing segment P head end	0.761	mm
mean radial wear sealing segment P crank end	0.893	mm
mean radial wear cap segment	0.351	mm
mean radial wear sealing segment head end	0.758	mm
mean radial wear sealing segment crank end	0.846	mm
mean radial wear	0.693	mm

Table 7.3. Detailed wear and wear volume data for ring B2.

variable	value	unit
area data		
theoretical area virgin state cap segment	138.3	mm^2
theoretical area virgin state sealing segment head end	103.1	mm^2
theoretical area virgin state sealing segment crank end	295.2	mm^2
total theoretical BCD area virgin state	1073.3	mm^2
measured area cap segment P	89.0	mm^2
measured area sealing segment P head end	60.1	mm^2
measured area sealing segment P crank end	220.2	mm^2
measured area cap segment	89.9	mm^2
measured area sealing segment head end	60.2	mm^2
measured area sealing segment crank end	220.5	mm^2
wear volume data		
measured wear volume cap segment P	51.45	mm^3
measured wear volume sealing segment P head end	53.22	mm^3
measured wear volume sealing segment P crank end	228.53	mm^3
measured wear volume cap segment	49.30	mm^3
measured wear volume sealing segment head end	48.15	mm^3
measured wear volume sealing segment crank end	212.82	mm^3
up-scaled wear volume cap segment P	79.93	mm^3
up-scaled wear volume sealing segment P head end	91.28	mm^3
up-scaled wear volume sealing segment P crank end	306.39	mm^3
up-scaled wear volume cap segment	75.88	mm^3
up-scaled wear volume sealing segment head end	82.48	mm^3
up-scaled wear volume sealing segment crank end	284.97	mm^3
total measured wear volume	643.46	mm^3
total up-scaled wear	920.92	mm^3
radial wear data		
mean radial wear cap segment P	0.578	mm
mean radial wear sealing segment P head end	0.886	mm
mean radial wear sealing segment P crank end	1.038	mm
mean radial wear cap segment	0.549	mm
mean radial wear sealing segment head end	0.800	mm
mean radial wear sealing segment crank end	0.965	mm
mean radial wear	0.870	mm

Developed code

Start calculation

This file contains all necessary input parameters to call the (Main_BCD_V04.py) backbone file, which starts a wear calculation including all sub calculations and scripts.

```python
#!/usr/bin/python
import os
os.system("python -u Main_BCD_V04.py \
--inp1 ID2Z_BCD_V11 \
--user1 UMM3D_BCD_V03_user1 \
--startcounter 0 \
--counter 51 \
--jobmain Main_2Z_BCD_V11 \
--jobsub Sub_2Z_BCD_V11_therm \
--inpmain Main_2Z_BCD_V11 \
--inpsub Sub_2Z_BCD_V11 \
--usermain UMM3D_BCD_V05.f \
--inpgenmain Main_Input_Generator_BCD_V04 \
--inpgensub Sub_Input_Generator_BCD_V05 \
--tempfilename Knoten_Temp \
--contactNodeSetName INNEN \
--wearElementSetName INNEN_ADA_MESH \
--neighborNumber 6 \
--inc 10 \
--highpressuretemp 6.1 \
--lowpressuretemp 0.1 \
--modelType 3D \
--abaquspath /opt/abaqus/Commands/abq6142 \
--cpus 1 \
| tee 0_LOG.log")
# The option -u in conjunction with a Python file makes Python work without buffering!
```

Backbone of the calculation

This file (Main_BCD_V04.py) starts the initial loading, then the neighbour nodes are searched and the first gas pressure distribution in the dynamic sealing surface is calculated before starting the wear calculation. The wear calculation and the pressure calculation alternate until the ring is worn away or a maximum number of wear calculations (wear time) is reached.

```
import sys
import os
import argparse                          # to parse arguments
from datetime import datetime
os.system('python -V')
start_time = datetime.now()
print('Starting time = '+str(start_time))
# Class containing all Parameters which are parsed
class Parameters:
    pass
parameter = Parameters()                 # instance of the class Parameters
# Parsing arguments
parser = argparse.ArgumentParser(description='Coupled Abaqus Jobs including UserDefinedSubroutines.')
# Adding all necessary and possible arguments to the parser.
parser.add_argument('--inp1', type=str, required=True, help='Name of the first input file, providing the geometry (including sets) and a first (dummy) step.')
parser.add_argument('--user1', type=str, required=False, help='Name of the first Fortran file, containing the UserDefinedSubroutines for the first(dummy/initialization) calculation.')
parser.add_argument('--counter', type=int, required=True, help='Counter counts the number of fully run cycles.')
parser.add_argument('--startcounter',type=int,required=False, help='This is the first value of the counter, so an analysis can be restarted at any step.')
parser.add_argument('--jobmain', type=str, required=True, help='Name of the main job containing the global model. This model is used repeatedly.')
parser.add_argument('--jobsub',  type=str, required=True, help='Name of the sub job containing the local model. This model is used repeatedly.')
parser.add_argument('--inpmain', type=str, required=True, help='Name of the input file for the main job.')
parser.add_argument('--inpsub',  type=str, required=True, help='Name of the input file for the sub model job.')
parser.add_argument('--usermain', type=str, required=True, help='Name of the Fortran file, containing the UserDefinedSubroutines for the main model.')
parser.add_argument('--usersub', type=str, required=False, help='Name of the Fortran file, containing the UserDefinedSubroutines for the sub model.')
parser.add_argument('--inpgenmain', type=str, required=True, help='Name of the input file generator for the main model.')
parser.add_argument('--inpgensub', type=str, required=True, help='Name of the input file generator for the sub model.')
parser.add_argument('--tempfilename', type=str,required=True, help='Name of the *.txt file where all temperature values for all nodes are stored.')
parser.add_argument('--contactNodeSetName', type=str, required=True, help='Name of the node set containing all contact nodes (in CAPITAL LETTERS). Will be passed to find_node_neighbors.py.')
parser.add_argument('--wearElementSetName', type=str, required=True, help='Name of the element set which is affected by mesh smoothing via UMeshMotion later and used by find_node_neighbors.py.')
parser.add_argument('--neighborNumber', type=int, required=True, help='Number of the neighbors which will be searched behind each contact node and used for the mesh smoothing algorithm.')
parser.add_argument('--highpressuretemp', type=float, required=True, help='Pressure of the cylinder side. This is the higher pressure. The maximum is 10 MPa (100bar). This maximum was agreed upon on 13.08.2015. This value is also the higher temperature in the sub calculation/model. The ring is loaded with 0 pressure on the crank side. On the cylinder side the load is highpressuretemp-lowpressuretemp. This gives the advantage that the pressure on the crank side mus t not be specified and applied on changing element surfaces.')
parser.add_argument('--lowpressuretemp', type=float, required=True, help='Pressure of the crank side. This is the lower pressure. The minimum is 0.1 MPa (1bar). This value is the lower temperature in the sub calculation/model. The ring is loaded with 0 pressure on the crank side. On the cylinder side the load is highpressuretemp-lowpressuretemp. This gives the advantage that the pressure on the crank side mus t not be specified and applied on changing element surfaces.')
```

```python
parser.add_argument('--modelType', type=str, required=True, help='Two model types are accepted: 3D and AX. 3D means a 3D Simulation with the z-axis being the length axis of the rod, AX means axisymmetric simulation, where x is the radial direction, y in rod direction.')
parser.add_argument('--abaquspath', type=str, required=True, help='Name of the abaqus executable.')
parser.add_argument('--cpus', type=int, required=True, help='Number of cpus which shall be used for the Abaqus calculation.')
parser.add_argument('--inc', type=float,required=True, help='This is the increment which is maximaly used when calculating the wear.')
args = parser.parse_args(namespace=parameter)
# Start of plausibility checks
if parameter.user1 == None:
    print('No user subroutine was given for the initialization. If a subroutine for the main part of the analysis was given, this subroutine is used instead.')
    if parameter.usermain != None:
        print('The usual subroutine is used.')
        parameter.user1 = parameter.usermain
        print(parameter.user1)

if parameter.startcounter == None or parameter.startcounter == 0:
    print('No startcounter or startcounter = 0 was given. Analysis starts at time zero with the initialization.')
if parameter.startcounter > parameter.counter:
    sys.exit('\n\n\n ERROR - The startcounter is greater or equal than the counter.\nReconsider the input values. \n\n')

if parameter.highpressuretemp > 10:
    sys.exit('\n\n\n ERROR - The given pressure value is greater than 10 MPa (100 bar). This is out of the application range. \n Reconsider the given input. \n\n')
elif parameter.highpressuretemp < 0:
    sys.exit('\n\n\n ERROR - The given pressure value is smaller than 0 MPa (0 bar). This is out of the application range. \n Reconsider the given input. \n\n')

if parameter.lowpressuretemp < 0.1:
    sys.exit('\n\n\n ERROR - The given pressure value is smaller than 0.1 MPa (1 bar). This is out of the application range. \n Reconsider the given input. \n\n')
elif parameter.lowpressuretemp > parameter.highpressuretemp:
    sys.exit('\n\n\n ERROR - The given (low) pressure value is bigger
     ↪  than the (high) pressure value. \n Reconsider the given input.
     ↪  \n\n')

# End of plausibility checks

print('Start of the Coupled Analysis\n')
path_working_dir = os.path.dirname(os.path.abspath(__file__))
# The directory of the current __file__ is cut down to the directory
 ↪  without the filename.
print('The current working direcotry is: '+str(path_working_dir))
if parameter.startcounter == 0:
    print('The Coupled Analysis begins with the initialization.')
    parameter.startcounter = 1
    # Start the first Abaqus job:
    print('Start the 1st Abaqus job:\nName of the
     ↪  job:\t\t'+str(parameter.jobmain)+'\nName of the input
     ↪  file:\t\t'+str(parameter.inp1)+'.inp\n\n')
    if 0 != os.system(parameter.abaquspath+"
     ↪  job="+str(parameter.jobmain)+"_0"+"
     ↪  inp="+str(parameter.inp1)+".inp"+"
     ↪  user="+str(parameter.user1)+" cpus="+str(parameter.cpus)+"
     ↪  interactive | tee "+str(parameter.jobmain)+"_0.log"):
        sys.exit("Error at Calculation _0")
    print("\n1st Abaqus job finished. \n")
    if 0 != os.system(parameter.abaquspath+' python -u get_set_info.py
     ↪  --path '+str(path_working_dir)+' --jobname
     ↪  '+str(parameter.jobmain)+' --neighborNumber
     ↪  '+str(parameter.neighborNumber)):
        sys.exit('Error during get_set_info.py.')
    if False ==
     ↪  os.path.isfile('neighbor_nodes_'+parameter.jobmain+'.txt'):
        # parameter.neighborNumber +1  because one more node is
         ↪  needed to give the wear direction for the edge nodes
        if 0 != os.system(parameter.abaquspath+' python -u
         ↪  find_node_neighbors.py --path '+str(path_working_dir)+'
         ↪  --jobname '+str(parameter.jobmain)+' --contactNodeSetName
         ↪  '+str(parameter.contactNodeSetName)+' --wearElementSetName
         ↪  '+parameter.wearElementSetName+' --neighborNumber
         ↪  '+str(parameter.neighborNumber+1)+' --modelType
         ↪  '+str(parameter.modelType)):
            sys.exit('Error during find_node_neighbors.py.')
    else:
```

```python
        print('\nThe neighbor node file exists, the search is skipped.\n\n')
else:
    print('This is a restart analysis. The calculation loop starts at loop number '+str(parameter.startcounter)+'.')
    i = parameter.startcounter

print('Starting the calculation loop:\n')
# Loop to gain a solution for the coupled 'temperature' - wear problem:
for i in range(parameter.startcounter,parameter.counter,1):
    print('\nLoop number '+str(i)+' :\nGeneration of the input file for the sub model.')
    if 0 != os.system(parameter.abaquspath+' python '+str(parameter.inpgensub)+'.py'+' --jobmain '+str(parameter.jobmain)+' --jobsub '+str(parameter.jobsub)+' --inpsub '+str(parameter.inpsub)+' --counter '+str(i-1)+' --path '+str(path_working_dir)+' --inpgensub '+str(parameter.inpgensub)+' --highpressuretemp '+str(parameter.highpressuretemp)+' --lowpressuretemp '+str(parameter.lowpressuretemp)):
        sys.exit('Error in generation of the Submodel in loop number '+str(i)+'.')
    print('\nDone. Continuing with the sub model Abaqus job.\n')
    if 0 != os.system(parameter.abaquspath+' job='+str(parameter.jobsub)+'_'+str(i)+' inp='+str(parameter.inpsub)+'_'+str(i-1)+' interactive | tee '+str(parameter.jobsub)+'_'+str(i)+'.log'):
        sys.exit('Error in calculation of the Submodel in loop number '+str(i)+'.')
    print('Done. Continuing with the generation of the input file for the main model.\n')
    if 0 != os.system(parameter.abaquspath+' python '+str(parameter.inpgenmain)+'.py'+' --jobmain '+str(parameter.jobmain)+' --jobsub '+str(parameter.jobsub)+' --inpmain '+str(parameter.inpmain)+' --tempfilename '+str(parameter.tempfilename)+' --counter '+str(i)+' --path '+str(path_working_dir)+' --inpgenmain '+str(parameter.inpgenmain)+' --wearElementSetName '+str(parameter.wearElementSetName)+' --inc '+str(parameter.inc)+' --highpressuretemp '+str(parameter.highpressuretemp)+' --lowpressuretemp '+str(parameter.lowpressuretemp)):
        sys.exit('Error in generation of the Mainmodel in loop number '+str(i)+'.')
    if 0 != os.system(parameter.abaquspath+' job='+str(parameter.jobmain)+'_'+str(i)+' inp='+str(parameter.inpmain)+'_'+str(i)+' user='+str(parameter.usermain)+' oldjob='+str(parameter.jobmain)+'_'+str(i-1)+' interactive | tee '+str(parameter.jobmain)+'_'+str(i)+'.log'):
        sys.exit('Error in calculation of the Mainmodel in loop number '+str(i)+'.')
    print('Done. Loop number '+str(i)+' finished. Current time: '+str(datetime.now())+'\n\n')

end_time = datetime.now()
print('Analysis done. Calculation finished. Current date and time: '+str(end_time))
delta_time = end_time-start_time
print('The analysis finished in '+str(delta_time))
```

Initial FORTRAN subroutine

The definition of UFILD is necessary already in the initial loading of the structure, only then UFIELD can be used in the following (restart-) wear calculations.

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC    UFIELD - Userdefined FIELD Variables
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE UFIELD(FIELD,KFIELD,NSECPT,KSTEP,KINC,TIME,NODE,
     1 COORDS,TEMP,DTEMP,NFIELD)
C
      INCLUDE 'ABA_PARAM.INC'
C
      DIMENSION FIELD(NSECPT,NFIELD), TIME(2), COORDS(3),
     1 TEMP(NSECPT), DTEMP(NSECPT)
C
      FIELD = 0.0
      RETURN
      END
```

FORTRAN subroutines to calculate wear

Below are all necessary FORTRAN subroutines, which allow to calculate the wear of a packing ring, based on the mechanisms governing this process.

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC       MODUL is used to share data between
CCCCC       different user-defined subroutines
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      MODULE Information

      CHARACTER*256 JOBNAME

      CHARACTER*80, DIMENSION(:), allocatable ::
     * node_set_names, element_set_names

      INTEGER  neighbor_node_nr, number_of_elements,
     *         number_of_node_sets, number_of_nodes,
     *         number_of_element_sets, num_contact_nodes,
     *         num_edge_nodes, LENJOBNAME

      INTEGER, DIMENSION(:), allocatable ::
     *   node_set_count, i_element_set_count,
     *   nodes_temp_array

      INTEGER, DIMENSION(:,:), allocatable ::
     *   N2EConn, nodes_in_contact, neighbors

      DOUBLE PRECISION, DIMENSION(:), allocatable ::
     *   temp_array

      DOUBLE PRECISION, DIMENSION(:,:), allocatable ::
     *  coords_nodes_in_contact, uvarm_neighbor_nodes,
     *  coords_allnodes, disp_neighbors

      DOUBLE PRECISION, DIMENSION(:,:,:), allocatable ::
     *  applied_disp

C   Information:
C      N2EConn             = Array that gives the all elements and
C                            their connected nodes
```

```
C      neighbor_node_nr    = number of neighbors of the contact node
C                            determined by the python script
C      number_of_elements  = number of all elements the model contains
C      number_of_nodes     = number of all   nodes  the model contains
C      number_of_element_sets
C      number_of_node_sets
C      num_contact_nodes   = number of nodes in contact
C                            between rod and ring (cup not considered)
C      num_edge_nodes      = number of nodes placed on an edge

C      coords_allnodes     = containing the coordinates of all nodes
C                            (:,1) - x-coords
C                            (:,2) - y-coords
C                            (:,3) - z-coords
C      JOBNAME             = name of the job, read from parameter file
C      LENJOBNAME          = length in characters of the JOBNAME

C  Following 2 arrays have the same index for 1 data-pair

C     nodes_temp_array    = containing the node numbers of the nodes
C                           which have temperature values in temp_array
C     temp_array          = containing all the temperature values
C                           for the nodes in the nodes_temp_array

C     node_set_names      = Array containing all  node   set names
C     node_set_count      = number of nodes in a set

C     element_set_names   = Array containing all element set names
C     i_element_set_count = number of elements in a set

C     nodes_in_contact    = array containing all nodes which are in
C                           contact with the rod (the nodes in contact
C                           with the cup are not included here ! )
C                           the node numbers are at index 1; index 2
C                           marks if the node is on the boarder of a
C                           surface (1=edge)or in the surface (=0)
C
C     coords_nodes_in_contact (:,1) = X
C                             (:,2) = Y
C                             (:,3) = Z
C     neighbors            = array containing all neighbor nodes (:,1)
C                            are the contact nodes and then (:,2) too
C                            (:,num_neigh) neighbors which are aligned
C                            radially - calculated by a python script
```

```
C                            and read by fortran
C     disp_neighbors       = displacement of the neighbors, in radial
C                            direction  (:,2 -> neighbor_node_nr)
C     applied_disp         = displacement of the neighbors
C                            (:,2 -> neighbor_node_nr; :); this variable
C                            is passed to UEXTERNALDB and then to UVARM
C                            for display of the displacment; the third
C                            index is for the x, y and z amount of the
C                            wear (z is 0)
C     uvarm_neighbor_nodes = array containing the sum of all the
C                            displacement which a node (neighbor or
C                            contact) experiences over time
C
      save
      END MODULE Information

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC       Subroutine UEXTERNALDB  allocate and initialize all
CCCCC                               global variables
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE UEXTERNALDB(LOP, LRESTART, TIME, DTIME, KSTEP, KINC)
C  Initialization is called only once and at the first run
      USE Information
      INCLUDE 'ABA_PARAM.INC'

C LOP = 0       the subroutine is called at the start of the analysis.
C LOP = 1       the subroutine is called at the start of the current
C               analysis increment. The subroutine can be called
C               multiple times at the beginning of an analysis
C               increment if the increment fails to converge and a
C               smaller time increment is required.
C LOP = 2       the subroutine is called at the end of the current
C               analysis increment. When LOP=2, all information that you
C               need to restart the analysis should be written to
C               external files.
C LOP = 3       the subroutine is called at the end of the analysis.
C LOP = 4       the subroutine is called at the beginning of a restart
C               analysis. When LOP=4, all necessary external files
C               should be opened and properly positioned and all
C               information required for the restart should be read
C               from the external files.
```

```
      DIMENSION TIME(2)
      CHARACTER*256 zeile, OUTDIR, filename
      CHARACTER*18  total_time
      CHARACTER zahl*5
C local variables:
      INTEGER       node_number, merker, nr_of_cols, problem
C problem ... an integer error code returned by OPEN
      INTEGER, DIMENSION (:,:),  allocatable ::
     *              N2E_mixed
      DOUBLE PRECISION temperature, coordinates(3)
C ex ... if file exists = .TRUE.
      LOGICAL ex

      WRITE(*,*) 'starte UEXTERNALDB mit LOP = ',LOP
      WRITE(*,*) 'LRESTART = ',LRESTART,'   TIME = ',TIME
      WRITE(*,*) 'DTIME = ',DTIME,'    KSTEP = ',KSTEP,'  KINC =',KINC

      IF ((LOP.EQ.4).AND.(KSTEP.EQ.0).AND.(KINC.EQ.0)) THEN
      CALL GETOUTDIR(OUTDIR,LENOUTDIR)
      WRITE(*,*) 'current working directory:', TRIM(OUTDIR)
      WRITE(*,*) '======= INITIALISIERUNG PHASE ========'

      node_number = 0
      merker = 0
      nr_of_cols = 0
      temperature = 0.0
      coordinates = 0.0

C Opening of the parameter file - this file includes all parameters
C which are needed for the FORTRAN code, the variables are saved to the
C module "Information".
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):) ='/parameter.txt'
      OPEN(150,FILE=filename,STATUS='OLD')
      DO i = 1,4,1
        READ(150,*) zeile
        WRITE(*,*) 'read has read the line: ', zeile
        IF (INDEX(zeile,'JOBNAME').EQ.1) THEN
           READ(150,*) JOBNAME
           READ(150,*) LENJOBNAME
           WRITE(*,*) 'jobname = ',TRIM(JOBNAME), ' CHAR: ', LENJOBNAME
        ELSEIF (INDEX(zeile,'NODE_SETS').EQ.1) THEN
           READ(150,*) number_of_node_sets
           WRITE(*,*) 'number of node sets ', number_of_node_sets
        ELSEIF (INDEX(zeile,'ELEMENT_SETS').EQ.1) THEN
```

```
           READ(150,*) number_of_element_sets
           WRITE(*,*) 'number of element sets ', number_of_element_sets
        ELSEIF (INDEX(zeile,'NEIGHBOR_NODE_NUMBER').EQ.1) THEN
           READ(150,*) neighbor_node_nr
           WRITE(*,*) 'neighbor nodes = ', neighbor_node_nr
        ENDIF
      ENDDO
      CLOSE(150)
      filename = ''
      ALLOCATE(node_set_names(number_of_node_sets))
      ALLOCATE(node_set_count(number_of_node_sets))
      ALLOCATE(element_set_names(number_of_element_sets))
      ALLOCATE(i_element_set_count(number_of_element_sets))
      node_set_names(1:number_of_node_sets) = ''
      node_set_count(1:number_of_node_sets) = 0
      element_set_names(1:number_of_element_sets) = ''
      i_element_set_count(1:number_of_element_sets) = 0

C Reading the set names
C *** NODE SETS ***
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):) ='/set_names_'//TRIM(JOBNAME)//'.txt'
      OPEN(151,FILE=filename,STATUS='OLD')
      READ(151,*) zeile
      IF (INDEX(zeile,'NODE_SET_NAMES').EQ.1) THEN
         DO i = 1,number_of_node_sets,1
            READ(151,*) node_set_names(i)
            READ(151,*) node_set_count(i)
            IF ((INDEX(node_set_names(i),'INNEN').EQ.1) .AND.
     *         (INDEX(node_set_names(i),'INNEN_ADA_MESH').NE.1)) THEN
               num_contact_nodes = node_set_count(i)
               WRITE(*,*) 'Number of contact nodes',num_contact_nodes
            ELSEIF (INDEX(node_set_names(i),'ALLNODES').EQ.1) THEN
               number_of_nodes = node_set_count(i)
               ALLOCATE(coords_allnodes(number_of_nodes,3))
               coords_allnodes(1:number_of_nodes,1:3) = 0.0
               WRITE(*,*) 'Total number of nodes:',number_of_nodes
             ELSEIF (INDEX(node_set_names(i),'KANTE').EQ.1) THEN
               num_edge_nodes = node_set_count(i)
               WRITE(*,*) 'Number of edge nodes', num_edge_nodes
            ENDIF
         ENDDO
      ELSE
         WRITE(*,*) 'ERROR - LOP 4 USXTERNALDB - reading set names'
      ENDIF
```

```
C *** ELEMENT SETS ***
      READ(151,*) zeile
      IF (INDEX(zeile,'ELEMENT_SET_NAMES').EQ.1) THEN
         DO i = 1,number_of_element_sets,1
            READ(151,*) element_set_names(i)
            READ(151,*) i_element_set_count(i)
            IF (INDEX(element_set_names(i),'EL_ALL').EQ.1) THEN
                number_of_elements = i_element_set_count(i)
            ENDIF
         ENDDO
      ELSE
         WRITE(*,*) 'ERROR - LOP 4 USXTERNALDB - reading element names'
      ENDIF
      CLOSE(151)
C Reading the neighbors nodes
C allocating neighbors: neighbor node nr +1 (contact nodes)
C +1 (neighbors to give direction for the nodes on an edge)
      filename = ''
      ALLOCATE(neighbors(num_contact_nodes,neighbor_node_nr+2))
      ALLOCATE(disp_neighbors(num_contact_nodes,neighbor_node_nr+1))
      ALLOCATE(applied_disp(num_contact_nodes,neighbor_node_nr+1,3))
      neighbors(1:num_contact_nodes,1:neighbor_node_nr+1) = 0
      disp_neighbors(1:num_contact_nodes,neighbor_node_nr+1) = 0.0
      applied_disp(1:num_contact_nodes,1:neighbor_node_nr+1,3) = 0.0
C Reading contact nodes and their neighbors from *.txt file,
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):)='/neighbor_nodes_'//TRIM(JOBNAME)//'.txt'
      OPEN(152,FILE=filename,STATUS='OLD')
      DO i = 1,num_contact_nodes,1
         READ(152,*) neighbors(i,1:neighbor_node_nr+2)
C         WRITE(*,*) 'Reading neighbors: ', neighbors(i,:)
      ENDDO
C START   Sanity check: the neighbor array must be free of zeros
      DO i = 1,num_contact_nodes,1
C         WRITE(*,*) neighbors(i,:)
         DO j = 1,neighbor_node_nr+2,1
            IF (neighbors(i,j).EQ.0) THEN
               WRITE(*,*) 'neighbors(',i,',',j,')=',neighbors(i,j)
            ENDIF
         ENDDO
      ENDDO
C END    Sanity check
      WRITE(*,*) 'Neighbors read.'
      filename = ''
      CLOSE(152)
```

```
C Reading temperature of contact nodes
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):) ='/Knoten_Temp.txt'
      ALLOCATE(temp_array(num_contact_nodes))
      ALLOCATE(nodes_temp_array(num_contact_nodes))
      temp_array(1:num_contact_nodes) = 0.0
      nodes_temp_array(1:num_contact_nodes) = 0
      OPEN(153,FILE=filename,STATUS='OLD')
      DO i = 1,num_contact_nodes,1
         READ(153,*) node_number, temperature
         nodes_temp_array(i) = node_number
         temp_array(i) = temperature
C         WRITE(*,*) 'Reading temp-node file: ',
C     *                 node_number, temperature
      ENDDO
      WRITE(*,*) 'Temp-file read.'
      node_number = 0
      i = 0
      filename = ''
      CLOSE(153)
C Reading contact nodes and their coordinates
      ALLOCATE(nodes_in_contact(num_contact_nodes,2))
      ALLOCATE(coords_nodes_in_contact(num_contact_nodes,3))
      nodes_in_contact(1:num_contact_nodes,2) = 0
      coords_nodes_in_contact(1:num_contact_nodes,3) = 0.0
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):) ='/node_INNEN.txt'
      OPEN(154,FILE=filename,STATUS='OLD')
      DO i = 1,num_contact_nodes,1
         READ(154,*) node_number, coordinates(:)
         IF (nodes_temp_array(i).EQ.node_number) THEN
            nodes_in_contact(i,1) = node_number
            nodes_in_contact(i,2) = 0
            coords_nodes_in_contact(i,1:3) = coordinates(:)
C            WRITE(*,*) 'nr: ',i,' node: ',
C    *            node_number,coords_nodes_in_contact(i,1:3)
         ELSE
           WRITE(*,*)'ERROR: nodes_in_contact and nodes_temp_array do',
     *'not contain the same sequence of nodes! - Further calculations',
     *' may be erroneous!'
         ENDIF
      ENDDO
      WRITE(*,*) 'Contact nodes ring-rod read.'
      filename = ''
      CLOSE(154)
```

```fortran

C Adding Information if tcontact node is an edge node or within a
C surface
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):) ='/node_KANTEN.txt'
      OPEN(155,FILE=filename,STATUS='OLD')
      READ(155,*) node_number, coordinates(:)
      i = 1
      j = 1
      DO WHILE (i.LE.num_edge_nodes)
C         WRITE(*,*) nodes_in_contact(j,:)
         IF (nodes_in_contact(j,1).EQ.node_number) THEN
            nodes_in_contact(j,2) = 1
C            WRITE(*,*) nodes_in_contact(j,:)
            i = i + 1
            j = j - 1
            IF (i.LE.num_edge_nodes) THEN
               READ(155,*) node_number, coordinates(:)
            ENDIF
         ENDIF
         j = j + 1
      ENDDO
      filename = ''
      CLOSE(155)

C Reading all element numbers with their connectivities
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):) ='/element_EL_ALL.txt'
      OPEN(155,FILE=filename,STATUS='OLD')
      merker = 0
      DO i = 1, number_of_element_sets, 1
        WRITE(*,*) 'set_name = ', element_set_names(i)
        IF (INDEX(element_set_names(i),'EL_ALL').EQ.1) THEN
           merker = i
           WRITE(*,*)'EL_ALL has',i_element_set_count(merker),'Element.'
           ALLOCATE(N2EConn(i_element_set_count(merker),9))
           ALLOCATE(N2E_mixed(i_element_set_count(merker),9))
           N2EConn(1:i_element_set_count(merker),1:9) = 0
           N2E_mixed(1:i_element_set_count(merker),1:9) = 0
        ENDIF
      ENDDO
      IF (merker.EQ.0) THEN
         WRITE(*,*) 'ERROR - EL_ALL NICHT GEFUNDEN'
      ELSE
        DO i = 1,i_element_set_count(merker), 1
```

```
          READ(155,*) N2E_mixed(i,:)
        ENDDO
        DO i = 1,i_element_set_count(merker), 1
          N2EConn(N2E_mixed(i,1),:)=N2E_mixed(i,:)
        ENDDO
        deallocate(N2E_mixed)
      ENDIF
      WRITE(*,*) 'All elements read.'
      filename = ''
      CLOSE(155)

C Reading the UVARM1 file
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):) ='/UVARM1.txt'
      OPEN(156,FILE=filename,STATUS='OLD')
      ALLOCATE(uvarm_neighbor_nodes(num_contact_nodes,
     *         neighbor_node_nr+1))
      uvarm_neighbor_nodes(1:num_contact_nodes,
     * 1:neighbor_node_nr+1) = 0.0
      DO i = 1, num_contact_nodes, 1
         READ(156,*) uvarm_neighbor_nodes(i,:)
      ENDDO
      WRITE(*,*) 'UVARM read.'
      filename = ''
      CLOSE(156)

      WRITE(*,*) '====== INITIALISIERUNG ENDE =========='
      ENDIF ! LOP = 4 END OF INITIALIZING

C => always update the uvarm!
      IF ((LOP.EQ.1).OR.(LOP.EQ.2).OR.(LOP.EQ.3).OR.(LOP.EQ.4)) THEN
      WRITE(*,*) 'Updating UVARM.'
      DO i = 1, num_contact_nodes, 1
         DO j = 1, neighbor_node_nr+1, 1
                uvarm_neighbor_nodes(i,j) =
     1             uvarm_neighbor_nodes(i,j) +
     2             sqrt(applied_disp(i,j,1)**2 +
     3                  applied_disp(i,j,2)**2 +
     4                  applied_disp(i,j,3)**2)
                applied_disp(i,j,:) = 0.0
         ENDDO
C      WRITE(*,*) neighbors(i,1), uvarm_neighbor_nodes(i,1)
      ENDDO
      WRITE(*,*) 'UVARM updated.'
      ENDIF
```

```
C LOP = 2:      end of current analysis increment
      IF (LOP.EQ.2) THEN
      filename = ''
      total_time = ''
      WRITE(*,*) 'Writing UVARM to file.'
C     Format of total_time and zahl dependent on their value
C      WRITE(*,*) 'TIME(2)', TIME(2)

      IF (TIME(2).LT.10) THEN
         WRITE(total_time,'(f7.4)') TIME(2)
      ELSEIF ((TIME(2).GE.10)       .AND.(TIME(2).LT.100)) THEN
         WRITE(total_time,'(f8.4)') TIME(2)
      ELSEIF ((TIME(2).GE.100)      .AND.(TIME(2).LT.1000)) THEN
         WRITE(total_time,'(f9.3)') TIME(2)
      ELSEIF ((TIME(2).GE.1000)     .AND.(TIME(2).LT.10000)) THEN
         WRITE(total_time,'(f10.3)') TIME(2)
      ELSEIF ((TIME(2).GE.10000)    .AND.(TIME(2).LT.100000)) THEN
         WRITE(total_time,'(f11.2)') TIME(2)
      ELSEIF ((TIME(2).GE.100000)   .AND.(TIME(2).LT.1000000)) THEN
         WRITE(total_time,'(f12.1)') TIME(2)
      ELSEIF ((TIME(2).GE.1000000)  .AND.(TIME(2).LT.10000000)) THEN
         WRITE(total_time,'(f13.0)') TIME(2)
      ELSEIF ((TIME(2).GE.10000000) .AND.(TIME(2).LT.100000000)) THEN
         WRITE(total_time,'(f14.0)') TIME(2)
      ENDIF

C      WRITE(*,*) 'total_time >', total_time,'<'

      nr_of_cols = neighbor_node_nr + 1

      IF ((nr_of_cols.GT.1).AND.(nr_of_cols.LT.10)) THEN
         WRITE(zahl,'(I1)') nr_of_cols
      ELSEIF((nr_of_cols.GT.9).AND.(nr_of_cols.LT.100))THEN
         WRITE(zahl,'(I2)') nr_of_cols
      ELSEIF (nr_of_cols.GT.99) THEN
         WRITE(zahl,'(I3)') nr_of_cols
      ELSEIF((neighbor_node_nr.EQ.0).OR.(neighbor_node_nr.GT.999)) THEN
         WRITE(*,*) 'ERROR - neighbor_node_nr either=0 or >999'
      ENDIF

C      WRITE(*,*) 'zahl', zahl
C      WRITE(*,*) 'TIME(2)', TIME(2)
C      WRITE(*,*) 'JOBNAME', TRIM(JOBNAME)
C      WRITE(*,*) 'total_time',TRIM(total_time)
```

```

      CALL GETOUTDIR(OUTDIR,LENOUTDIR)
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):)='/UVARM1_'//TRIM(JOBNAME)//'_'//
     1 TRIM(ADJUSTL(total_time))//'.txt'

C       WRITE(*,*) 'filename: ',TRIM(filename)

C Check if the filename exists - added 27.2.2017:
C (if a 'NEW' file is opened and this file already exists an error
C occurs and Abaqus stops, thus the existing file is deleted and
C rewritten - this minor inaccuracy should hardly be noticable because
C this should only happen if the increment steps is small and the
C absolute calculation time is big)

      OPEN(159,FILE=TRIM(filename),STATUS='REPLACE')
      DO i = 1, num_contact_nodes, 1
         WRITE(159,"("//TRIM(ADJUSTL(zahl))//"(F14.10,:,',  '))")
     *           uvarm_neighbor_nodes(i,:)
C           WRITE(*,*) uvarm_neighbor_nodes(i,:)
      ENDDO
      zahl = ''
      filename = ''
      CLOSE(159)
      WRITE(*,*) 'UVARM written to file.'

      ENDIF ! LOP = 2 END OF CURRENT INCREMENT

      IF (LOP.EQ.3) THEN
      filename = ''
      WRITE(*,*) 'LOP 3: Writing UVARM to UVARM1.txt'
      CALL GETOUTDIR(OUTDIR,LENOUTDIR)
      filename(1:LENOUTDIR) = OUTDIR(1:LENOUTDIR)
      filename((LENOUTDIR+1):) ='/UVARM1.txt'
      OPEN(157,FILE=TRIM(filename),STATUS='REPLACE')
      nr_of_cols = neighbor_node_nr + 1
      IF ((nr_of_cols.GT.1).AND.(nr_of_cols.LT.10)) THEN
         WRITE(zahl,'(I1)') nr_of_cols
      ELSEIF((nr_of_cols.GT.9).AND.(nr_of_cols.LT.100))THEN
         WRITE(zahl,'(I2)') nr_of_cols
      ELSEIF (nr_of_cols.GT.99) THEN
         WRITE(zahl,'(I3)') nr_of_cols
      ELSEIF((neighbor_node_nr.EQ.0).OR.(neighbor_node_nr.GT.999)) THEN
         WRITE(*,*) 'ERROR - neighbor_node_nr either=0 or >1000'
      ENDIF
      DO i = 1, num_contact_nodes, 1
         WRITE(157,"("//TRIM(ADJUSTL(zahl))//"(F16.12,:,',  '))")
     *           uvarm_neighbor_nodes(i,:)
      ENDDO
      filename = ''
      zahl = ''
      CLOSE(157)
      WRITE(*,*) 'LOP 3: UVARM to UVARM1.txt written.'
      ENDIF ! LOP = 3 END OF CURRENT ANALYSIS


      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC       Subroutine URDFIL to read from the result file
CCCCC                         or to write into it
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE URDFIL(LSTOP, LOVRWRT, KSTEP, KINC, DTIME, TIME)

      USE Information
      INCLUDE 'ABA_PARAM.INC'

      DIMENSION ARRAY(513),JRRAY(NPRECD,513),TIME(2)
      EQUIVALENCE (ARRAY(1),JRRAY(1,1))

      INTEGER index1

      index1 = 0

C      WRITE(*,*) 'URDFIL - START'
C      WRITE(*,*)'STEP=',KSTEP,'INC=',KINC,'TIME INC=',DTIME
C      WRITE(*,*)'STEP TIME=',TIME(1),'TOTAL TIME=',TIME(2)

      LSTOP = 0
C No overwriting with lovrwrt = 0, overwriting with = 1
      LOVRWRT = 0

      CALL POSFIL(KSTEP,KINC,ARRAY,JRCD)
      DO K1 = 1, 999999
         JRRAY(1,:) = 0
         CALL DBFILE(0,ARRAY,JRCD)
         IF (JRCD.NE.0) GO TO 101
         KEY = JRRAY(1,2)
C KEY = 107:    CURRENT NODE COORDINATES
         IF (KEY.EQ.107) THEN
C           WRITE(*,*) 'Node Nr: JRRAY = ',JRRAY(1,3)
C           WRITE(*,*) 'Coords : ARRAY = ',ARRAY(4:6)
           coords_allnodes(JRRAY(1,3),1:3) = ARRAY(4:6)
           CALL FINDITEMINLIST(nodes_in_contact, num_contact_nodes,
     *                         index1, JRRAY(1,3))
           IF (index1.NE.0) THEN
              coords_nodes_in_contact(index1,1:3) = ARRAY(4:6)
C              WRITE(*,*) 'Coords von Node', JRRAY(1,3),
C     *                    coords_nodes_in_contact(index1,:)
           ENDIF
           index1 = 0
         ENDIF
      ENDDO

 101  CONTINUE
C      WRITE(*,*) 'ENDE URDFIL'
C      WRITE(*,*) 'neighbors (1:2,1)', neighbors(1:2,1)
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC       Subroutine UMESHMOTION user-defined node movement
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE UMESHMOTION(UREF,ULOCAL,NODE,NNDOF,
     *            LNODETYPE,ALOCAL,NDIM,TIME,DTIME,PNEWDT,
     *            KSTEP,KINC,KMESHSWEEP,JMATYP,JGVBLOCK,LSMOOTH)
      USE Information
      INCLUDE 'ABA_PARAM.INC'
C - Variables UMESHMOTION
      DIMENSION ULOCAL(NDIM)
      INTEGER NODE
C JELEMLIST(*)
      DIMENSION ALOCAL(NDIM,*),TIME(2)
      DIMENSION JMATYP(*),JGVBLOCK(*)
C - Variables GETNODETOELEMCONN
      PARAMETER ( MAXNELEMS = 100)
      DIMENSION JELEMLIST(MAXNELEMS),JELEMTYPE(MAXNELEMS)
```

```
C - Variables GETVRMAVGATNODE
      DIMENSION ARRAY(15)

C ... other Variables

      DOUBLE PRECISION WGLOBAL(NDIM),WLOCAL(NDIM),
     *      cpress, norm_coord(3), radius, radial_wear,
     *      coords_n(3), coords_k(3), betrag, wear_vector(3),
     *      wear_in_norm
      INTEGER ind1(9), ind2(9), isp(2), izei(2),
     *        i_edge_flag

C    betrag      ... Betrag des Vektors von coords_k nach coords_n
C    coords_k    ... Koordinaten des Kontaktknoten (bzw. des Knotens der
C                    naeher an der Kolbenstange liegt)
C    coords_n    ... Koordinaten des Nachbarknoten (bzw. des Knotens der
C                    weiter weg von der Kolbenstange liegt)
C    cpress      ... Contact pressure at the current NODE
C    i_edge_flag ... Flag if a node is on the edge or within a surface
C    ind1, ind2  --> Indices returned from FIND Subroutines
C    isp         ... uebergabe der Spalten Nr von (1) bis (2) dann
C                    INDEXFIND ausgefuehrt
C    izei        ... uebergabe der Zeilen Nr von (1) bis (2) an
C                    INDEXFIND
C    norm_coord  ... normierte Koordinaten der Verschleiszrichtung
C                    (1 = x), (2 = y), (3 = z ist
C                    nicht von belangen fuer Knoten in der
C                    Kontaktflaeche, wir jedoch beruecksichtigt bei
C                    Eckknoten)
C   radius       ... benoetigt zum Normieren der Koordinaten
C   radial_wear  ... absoluter Verschleiszbetrag in radialer Richtung
C   wear_vector  ... Verschleiszvektor (Gibt due Richtung des
C                    Verschleiszes an und ist NICHT GLEICH dem
C                    Verscheibungsvektors WGLOBAL. WGLOBAL kann in
C                    eine andere Richtung zeigen, wenn ein Knoten an
C                    einer Kante sitzt.)
C   wear_in_norm ... Inproduct of wear_vector and norm_coord

C setzen des Koordinatensystem auf 0 = global, 1 = lokal
C       Write(*,*) 'entering umeshmotion'
       LRTN = 0
       UREF = 1
       JTYP = 0
       JRCD = 0
       NELEMS = MAXNELEMS
```

```fortran
       JELEMLIST = 0
       JELEMTYPE = 0
       WGLOBAL = 0.0

       ind1(1:9) = 0
       ind2(1:9) = 0
       isp(1:2) = 0
       izei(1:2) = 0
C if i_edge_flag is not reset it raises error
       i_edge_flag = 2
       betrag = 0.0
       norm_coord = 0.0
       cpress = 0.0
       coords_n = 0.0
       coords_k = 0.0
       radius = 0.0
       radial_wear = 0.0
       wear_vector = 0.0

C ACHTUNG: in dieser Subroutine wird nur das im Inputfile definierte
C Rechengebiet betrachtet !!! (*ADAPTIVE MESH)
C      WRITE(*,*) '+++++++++++++++++++++++++++++++++++++++++++++++'
C      WRITE(*,*) 'wir sind bei Knoten ', NODE
C      WRITE(*,*) 'LNODETYPE = ', LNODETYPE

C      WRITE(*,*) 'bevor GETNODE... NELEMS = ',NELEMS

      CALL GETNODETOELEMCONN(NODE, NELEMS, JELEMLIST, JELEMTYPE,
     *                       JRCD, JGVBLOCK)
C      WRITE(*,*) 'get node to element connectivity gives:'
C      WRITE(*,*) 'node' , NODE
C      WRITE(*,*) 'NELEMS' , NELEMS
C      WRITE(*,*) 'jelemlist', JELEMLIST
C      WRITE(*,*) 'jelemtype', JELEMTYPE(1:NELEMS)
C      WRITE(*,*) 'jrcd', JRCD
C      WRITE(*,*) '-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-'

C if an error pops up in the .msg file which says:
C ***ERROR: SYSTEM ERROR IN shr_getvmave -- INCONSISTENT VARIABLE
C LENGTHS.
C
C it is highly likely that in the *Surface Interaction the option
C Tracking Thickness was not set at all or set to a too low number
C example of the line:
C *Surface Interaction, name=FRICTIONLESS, TRACKING THICKNESS = 5
```

```

      CALL GETVRMAVGATNODE(NODE,JTYP,'CSTRESS',ARRAY,JRCD,
     *                     JELEMLIST,NELEMS,JMATYP,JGVBLOCK)

      cpress = ARRAY(1)
C      WRITE(*,*) 'Kontaktdruck fuer Node ',NODE,' = ', cpress
C      WRITE(*,*) 'Kontaktdruckarray fuer Node ',NODE,' = ',array(1:9)

C for umeshmotion node has to see contact pressure and be in the
C nodes_in_contact set (so nodes which are in contact with the cup are
C not considered)
      izei(1) = 1
      izei(2) = num_contact_nodes
      isp(1) = 1
      isp(2) = neighbor_node_nr+1
      CALL FINDINDEX(neighbors(izei(1):izei(2),isp(1):isp(2)),
     *               izei, isp, ind1, ind2, NODE)
      i_edge_flag = nodes_in_contact(ind1(1),2)

      IF ((cpress.GT.0.0).AND.(ind1(1).NE.0).AND.(ind2(1).EQ.1)) THEN
C         WRITE(*,*) 'contact node'
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C Berechnungen der Verschiebungen fuer das manuelle Smoothing der
C hinteren Nachbarknoten, um weniger Mesh-Distortion zu haben.

C Abhaengig ob es sich um einen Knoten an der Kante oder in der Flaeche
C handelt, wird der Absolutbetrag des Verschleiszes aufgebracht:
C
C Abschaetzung des Verschleisfaktors von etwa 1 micromenter/Stunde
C was etwa 2.8 * 10 hoch -7 mm/sec
          radial_wear = 2.8E-7*cpress*DTIME

C   CHECK START    if wear is to big => nodes can leave the surface
C                  reasonable limit 0.005 mm (for the increment)
          IF (radial_wear.GT.0.007) THEN
             WRITE(*,*) 'WEAR VERRINGERT von ', radial_wear,
     *                  ' auf  0.007.'
             radial_wear = 0.007
          ENDIF
C   CHECK END

C calculation of the wear_vector (in radial direction):
C Durch die Koordinaten des Knoten kann die Verschleiszrichtung
C bestimmt werden. diese ist in radialer Richtung. (Koordinaten
C werden normiert (durch Betrag (x,y, nicht z) dividiert) und
C damit der Verschleiszbetrag multipliziert, dieser Betrag hat
C dem Name: 'radius')
          radius = SQRT(coords_nodes_in_contact(ind1(1),1)**2
     1                 +coords_nodes_in_contact(ind1(1),2)**2)
          DO i = 1, 2, 1
             norm_coord(i)=coords_nodes_in_contact(ind1(1),i)/radius
             wear_vector(i) = radial_wear*norm_coord(i)
          ENDDO
          wear_vector(3) = 0.0


C --> IN DER FLAECHE
C es muessen nur noch die Werte von oben uebernommen werden
          IF (i_edge_flag.EQ.0) THEN
             WGLOBAL = wear_vector
C --> AN DER KANTE
          ELSEIF (i_edge_flag.EQ.1) THEN
C normierten Vektor zum naechsten Nachbarn bestimmen:
C Koordinaten Kontaktknoten
              coords_k(1:3)=coords_allnodes(neighbors(ind1(1),1),1:3)
C Koordinaten Nachbarknoten
              coords_n(1:3)=coords_allnodes(neighbors(ind1(1),2),1:3)
C normierter Vektor von Kontaktknoten zum Nachbarknoten
              betrag = SQRT((coords_n(1)-coords_k(1))**2+
     *                      (coords_n(2)-coords_k(2))**2+
     *                      (coords_n(3)-coords_k(3))**2)
              IF (betrag.LT.0.0001) THEN
                 WRITE(*,*) 'ERROR - variable betrag zu klein', betrag
                 WRITE(*,*) 'NODE', NODE, coords_k
                 WRITE(*,*) 'neighbor',neighbors(ind1(1),2),coords_n
              ENDIF
              DO i = 1, NDIM
                 norm_coord(i) = (coords_n(i)-coords_k(i))/betrag
              ENDDO
C Berechnung Verschleiszvektor:
C wear_vector(3) := 0
C    => kein Beitrag im Inprodukt aus der dritten Komponente
              wear_in_norm = wear_vector(1)*norm_coord(1)+
     *                       wear_vector(2)*norm_coord(2)
              DO i = 1, NDIM
C               WRITE(*,*) 'radial_wear = ', radial_wear
              WGLOBAL(i)=radial_wear**2/(wear_in_norm)*norm_coord(i)
C               WRITE(*,*) 'WGLOBAL = ', WGLOBAL
              ENDDO
          ELSE
             WRITE(*,*) 'ERROR - i_edge_flag = ',i_edge_flag
          ENDIF

C CHECK START           IF  NODES  LEAVE THE SURFACE

C Following additional IF statement is necessary to prevent nodes
C from wearing if they do not have contact to the (rigid) rod any more:
C IF the radial node coordinate is bigger than 0.0005 the radius of the
C rod there is no more movement of the node performed.
          IF (radius.GT.25.41) THEN
              WGLOBAL(1:NDIM) = 0.0
C Also radial_wear has to be 0. To stop the neighbors from moving.
              radial_wear = 0.0
              WRITE(*,*) NODE,'WEAR WAS SET TO 0; radius=',radius
          ENDIF
          IF (radius.GT.25.409) THEN
              WRITE(*,*) 'Node ',NODE,' has left the rigid:'
              WRITE(*,*) 'Radius = ', radius
          ENDIF
C CHECK END   ----------------------------------------------------------

C uebernehmen der aufgebrachten Verschiebung in das globale Array
C applied_disp
          DO i = 1, NDIM, 1
             applied_disp(ind1(1),1,i) = WGLOBAL(i)
          ENDDO

C Calculate the radial wear for the neighbor nodes:
          DO i = 1, neighbor_node_nr
             disp_neighbors(ind1(1),i+1) = radial_wear *
     *       (neighbor_node_nr+1-i)/(neighbor_node_nr+1)
          ENDDO
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C Aufbringen der Verschiebung, wenn der Knoten ein hinterer Nachbar ist

      ELSEIF ((ind1(1).NE.0).AND.
     *       ((ind2(1).GT.1).AND.(ind2(1).LE.neighbor_node_nr+1))) THEN
C Datenbereitstellung:
C         WRITE(*,*) 'neighbor detected'
         i_edge_flag = nodes_in_contact(ind1(1),2)
         radial_wear = disp_neighbors(ind1(1),ind2(1))
         disp_neighbors(ind1(1),ind2(1)) = 0.0

         IF (radial_wear.GT.0) THEN
```

```
            radius = SQRT(
     1             coords_allnodes(neighbors(ind1(1),ind2(1)),1)**2
     2           + coords_allnodes(neighbors(ind1(1),ind2(1)),2)**2)
            DO i = 1, 2, 1
               wear_vector(i) = radial_wear *
     *           coords_allnodes(neighbors(ind1(1),ind2(1)),i)/radius
            ENDDO
            wear_vector(3) = 0.0

C wieder Differentiation ob der Knoten ein Kantenknoten ist oder nicht:
C --> IN DER FLAECHE:
            IF (i_edge_flag.EQ.0) THEN
               DO i = 1, NDIM, 1
                  WGLOBAL(i) = wear_vector(i)
               ENDDO

C --> AN DER KANTE:
            ELSEIF (i_edge_flag.EQ.1) THEN
C normierten Vektor zum naechsten Nachbarn bestimmen:
C Koordinaten vom aktuellen Nachbarknoten
             coords_k(1:3)=coords_allnodes(neighbors(ind1(1),ind2(1)),:)
C Koordinaten vom naechsten Nachbarknoten
             coords_n(1:3)=
     *                   coords_allnodes(neighbors(ind1(1),ind2(1)+1),:)
C normierter Vektor von Knoten k zum Knoten n
               betrag = SQRT((coords_n(1)-coords_k(1))**2
     1                      + (coords_n(2)-coords_k(2))**2
     2                      + (coords_n(3)-coords_k(3))**2)
               DO i = 1, NDIM
                  norm_coord(i) = (coords_n(i)-coords_k(i))/betrag
               ENDDO
C Berechnung Verschleisz:
C wear_vector(3) := 0
C    => kein Beitrag im Inprodukt aus der dritten Komponente
               wear_in_norm = wear_vector(1)*norm_coord(1)+
     *                        wear_vector(2)*norm_coord(2)
               DO i = 1, NDIM
                 WGLOBAL(i)=radial_wear**2/(wear_in_norm)*norm_coord(i)
               ENDDO
            ELSE
                WRITE(*,*) 'ERROR - i_edge_flag = ', i_edge_flag
            ENDIF
C uebertrage den Verschleiszbetrag zur Darstellung im UVARM
            DO i = 1, NDIM, 1
```

```
               applied_disp(ind1(1),ind2(1),i) = WGLOBAL(i)
            ENDDO
         ELSE
           WGLOBAL = 0.0
         ENDIF
      ENDIF

C      WRITE(*,*) 'WGLOBAL @ NODE',NODE, ':', WGLOBAL(:)

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C Transformation globale Verschleiszrichtung in die
C => lokale Verschleiszrichtung
C WLOCAL - wird berechnet => Berechnung  ULOCAL (braucht Abaqus)
C WGLOBAL - von mir vorgegeben
C ALOCAL - von UMESHMOTION vorgegeben:
      DO i = 1, NDIM
        WLOCAL(i) = 0
        DO j = 1, NDIM
          WLOCAL(i) = WLOCAL(i) + WGLOBAL(j) * ALOCAL(j,i)
        END DO
      END DO
      DO i = 1, NDIM
        ULOCAL(i) = ULOCAL(i) + WLOCAL(i)
      END DO


C      WRITE(*,*) NODE, 'WLOCAL', WLOCAL(:)
C      WRITE(*,*) 'ULOCAL', ULOCAL(:)
C      WRITE(*,*) 'Ende UMESHMOTION'

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC          Subroutine DLOAD to define user-defined loads
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE DLOAD(F,KSTEP,KINC,TIME,NOEL,NPT,LAYER,KSPT,
     *                   COORDS,JLTYP,SNAME)

      USE Information
      INCLUDE 'ABA_PARAM.INC'

      DIMENSION TIME(2), COORDS(3)
      CHARACTER*80 SNAME
      DOUBLE PRECISION pressure(4),coords_node(4,3), cipt(2), F

      INTEGER  ind1, ind2, izei, nodes(4), izaehl
C JLTYP = 0 ... Surface-based load
      JLTYP = 0
      nodes = 0
      pressure = 0D0
      coords_node = 0D0
      cipt = 0D0        !coordinateds of the integration point
C                        in the unity element
      F = 0D0
C F = 1 MPa = 10 bar
C      WRITE(*,*) ' '
C      WRITE(*,*) 'Was ist mit DLOAD?', NOEL, N2EConn(NOEL,1)

C Searching the Nodes which are connected to the Element, checking if
C they are in contact with the rod => if they are in the array
C nodes_in_contact
      izaehl = 0
      DO j = 2, 9
         izei = num_contact_nodes
         ind1 = 0
         IF (N2EConn(NOEL,j).NE.0) THEN
         CALL FINDITEMINLIST(nodes_in_contact,izei,ind1,N2EConn(NOEL,j))
           IF (ind1.NE.0) THEN
               izaehl = izaehl + 1
               nodes(izaehl) = N2EConn(NOEL,j)
               coords_node(izaehl,:) = coords_nodes_in_contact(ind1,:)
               pressure(izaehl) = temp_array(ind1)
            ENDIF
         ENDIF
      ENDDO


      IF (izaehl.NE.4) THEN
      DO j = 2, 9
         izei = num_contact_nodes
         ind1 = 0
         IF (N2EConn(NOEL,j).NE.0) THEN
         WRITE(*,*) 'NOEL = ', NOEL
         WRITE(*,*) 'j = ', j
         WRITE(*,*) 'N2EConn(NOEL,j)', N2EConn(NOEL,j)
         CALL FINDITEMINLIST(nodes_in_contact,izei,ind1,N2EConn(NOEL,j))
           IF (ind1.NE.0) THEN
               izaehl = izaehl + 1
               WRITE(*,*) 'izaehl = ', izaehl
               nodes(izaehl) = N2EConn(NOEL,j)
               WRITE(*,*) 'nodes(izaehl) = ', nodes(izaehl)
               coords_node(izaehl,:) = coords_nodes_in_contact(ind1,:)
               WRITE(*,*) 'coords_node (izaehl,:)',coords_node(izaehl,:)
               pressure(izaehl) = temp_array(ind1)
               WRITE(*,*) 'pressure(izaehl) = ', pressure(izaehl)
            ENDIF
         ENDIF
      ENDDO
         WRITE(*,*)'E R R O R - W R O N G NODE NUMBER: ', izaehl
         WRITE(*,*) 'NOEL = ', NOEL
         WRITE(*,*) 'N2EConn(NOEL,2:9)', N2EConn(NOEL,2:9)
         WRITE(*,*) '-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-'
      ENDIF


C Coordinates of the integration point depending on the NPT (number of
C the integration point).
C    coordinates: s (=x)   t (=y)
      IF (NPT.EQ.1) THEN
         cipt(1) = -0.57735026918963
         cipt(2) =                      -0.57735026918963
      ELSEIF (NPT.EQ.2) THEN
         cipt(1) =  0.57735026918963
         cipt(2) =                      -0.57735026918963
      ELSEIF (NPT.EQ.3) THEN
         cipt(1) =  0.57735026918963
         cipt(2) =                       0.57735026918963
      ELSEIF (NPT.EQ.4) THEN
         cipt(1) = -0.57735026918963
         cipt(2) =                       0.57735026918963
      ELSE
         WRITE(*,*) 'ERROR Integration point number not 1, 2, 3 or 4!'
      ENDIF
C Solving the trial function (Ansatzfunktion) in the pictorial space:
      F = (pressure(1)*(1 - cipt(1) - cipt(2) + cipt(1)*cipt(2))
     1    +pressure(2)*(1 + cipt(1) - cipt(2) - cipt(1)*cipt(2))
     2    +pressure(3)*(1 + cipt(1) + cipt(2) + cipt(1)*cipt(2))
     3    +pressure(4)*(1 - cipt(1) + cipt(2) - cipt(1)*cipt(2)))/4
      RETURN
      END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC   UFIELD - user-definded field variable
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE UFIELD(FIELD,KFIELD,NSECPT,KSTEP,KINC,TIME,NODE,
     1 COORDS,TEMP,DTEMP,NFIELD)
C
      USE Information
      INCLUDE 'ABA_PARAM.INC'
C
      INTEGER   izei(2),isp(2),ind1(9),ind2(9)

      DIMENSION FIELD(NSECPT,NFIELD), TIME(2), COORDS(3),
     1 TEMP(NSECPT), DTEMP(NSECPT)
C
C      WRITE(*,*) 'ENTERING UFIELD'
C      WRITE(*,*) 'KFIELD=',KFIELD
C      WRITE(*,*) 'NSECPT=',NSECPT
C      WRITE(*,*) 'TEMP=',  TEMP
C      WRITE(*,*) 'NFIELD=',NFIELD
C      WRITE(*,*) 'FIELD=',FIELD
C      WRITE(*,*) '---------------------------'

      izei(1) = 1
      izei(2) = num_contact_nodes
      isp(1) = 1
      isp(2) = neighbor_node_nr+1
      ind1 = 0
      ind2 = 0
      CALL FINDINDEX(neighbors(izei(1):izei(2),isp(1):isp(2)),
     *               izei, isp, ind1, ind2, NODE)
      IF ((ind1(1).NE.0).AND.(ind2(1).NE.0)) THEN
         FIELD(1,1) = uvarm_neighbor_nodes(ind1(1),ind2(1))
C         WRITE(*,*) 'FIELD=',FIELD
      ELSE
         FIELD(1,1) = 0.0
      ENDIF

      RETURN
      END
```

```



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC    FINDINDEX - finds and returns the index where a searched
CCCCC                value is in an array
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE FINDINDEX(iarray, izeile, ispalte, index1, index2,
     *                     iwert)

      INTEGER izeile(2),ispalte(2),
     *        iarray(izeile(1):izeile(2),ispalte(1):ispalte(2)),
     *        index1(9), index2(9), iwert, izaehler

      index1=0
      index2=0
      izaehler=1

      DO i = ispalte(1), ispalte(2), 1
         DO j = izeile(1), izeile(2), 1
            IF (iarray(j,i).EQ.iwert) THEN
C      WRITE(*,*) 'INDEX gefunden: Zeilennr, Spaltennr',i,j,iarray(j,i)
               index1(izaehler) = j
               index2(izaehler) = i
               izaehler = izaehler + 1
            ENDIF
C         WRITE(*,*) 'FINDINDEX,Zeil,Spalt,WERT',j,',',i,',',iarray(j,i)
         ENDDO
      ENDDO
C wird der Wert nicht gefunden, so werden index1 und index2 = 0 gesetzt
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCC
CCCCC    FINDITEMINLIST - finds and returns index of a searched
CCCCC                     value on a 1D array
CCCCC                     (only for integer values AND the value
CCCCC                     must only occur once within the array)
CCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

```
      SUBROUTINE FINDITEMINLIST(iarray, length, index1, iwert)

      INTEGER length,iarray(1:length),
     *        index1, iwert

      index1    = 0
      DO i = 1, length, 1
         IF (iarray(i).EQ.iwert) THEN
            index1 = i
C            WRITE(*,*) 'Wert ',iwert,' auf Pos ',index1
         ENDIF
      ENDDO
      END
```

Obtain information about sets

This code generates an information file, which has to be provided to the FORTRAN wear routine.

```
import argparse          # ability to use and parse arguments
import sys
from odbAccess import *  # ability to work with an Abaqus - ODB file

class Parameters:
    pass
parameter = Parameters() # instance of the class Parameters
# Parsing arguments
parser = argparse.ArgumentParser(description="Writing the input file for the Abaqus analysis of a sub model.")
parser.add_argument('--path',               type=str, required=True, help='Path of the working directory.')
parser.add_argument('--jobname',            type=str, required=True, help='Name of the job, where the neighbor nodes will be searched.')
parser.add_argument('--neighborNumber',     type=int, required=True, help='Number of neighbor nodes.' )
args = parser.parse_args(namespace=parameter)
#
odb = openOdb(path=parameter.path+"/"+parameter.jobname+'_0.odb')
print('get_set_info.py - START.')
parameter_file = open('parameter.txt', 'w')
parameter_file.write('JOBNAME\n'+str(parameter.jobname)+'\n'+str(len(parameter.jobname))+'\n')
```

```python
parameter_file.write('NEIGHBOR_NODE_NUMBER\n'+str(parameter.neighborNumber)+'\n')
set_file = open('set_names_'+str(parameter.jobname)+'.txt', 'w')
# Writing all node set names and the number of nodes this set contains to a file (set_file.txt)
# All nodes of a node set are written to a separate file (including coordinates @ beginning)
set_file.write('NODE_SET_NAMES\n')
lastFrame = odb.steps['Verschleiss'].frames[-1]
set_counter = 0
max_node_nr = 0

for node_sets in odb.rootAssembly.instances['PART-1-1'].nodeSets.keys():
    set_counter += 1
    print('Working on node set: '+str(node_sets))
    set_file.write(str(node_sets)+'\n')
    node_set_file = open('node_'+str(node_sets)+'.txt','w')
    node_set_temp_array = [] # array for cross checking the node number => no double nodes
    node_counter = 0
    for nodes in odb.rootAssembly.instances['PART-1-1'].nodeSets[node_sets].nodes:
        node_counter += 1
        node_set_temp_array.append(nodes.label)
        if nodes.label > max_node_nr:
           max_node_nr = nodes.label
    for allnodes in lastFrame.fieldOutputs['COORD'].values:
        if (allnodes.nodeLabel in node_set_temp_array):
            node_set_file.write(str(allnodes.nodeLabel)+', '+', '.join(map(str,allnodes.data[0:3]))+'\n')
            node_set_temp_array.remove(allnodes.nodeLabel)
    if str(node_sets) == 'INNEN':
        contact_node_nr = node_counter
    if str(node_sets) == 'ALLNODES':
        if node_counter < max_node_nr:
           sys.exit('\n\n\n ERROR \n\n The maximum node number of ALLNODES is bigger than the node_counter. \n max_node_nr = '+str(max_node_nr)+'; node_counter = '+str(node_counter)+' \n This can lead to a memory leakage in the FORTRAN subroutines! \n\n Correction of the error by changing the ALLNODES node set: \n In ALLNODES all the nodes of the model (also reference points) have to be included. \n\n')
    set_file.write(str(node_counter)+'\n')
    node_set_file.close()
parameter_file.write('NODE_SETS,\n'+str(set_counter)+'\n')
set_counter = 0
# Writing all element set names and the number of elements this set contains to a file (set_file.txt)
# All elements of a element set and its connectivities are written to a separate file (including connectivities)
set_file.write('ELEMENT_SET_NAMES\n')
for element_sets in odb.rootAssembly.instances['PART-1-1'].elementSets.keys():
    set_counter += 1
    print('Working on element set: '+str(element_sets))
    set_file.write(str(element_sets)+'\n')
    element_set_file = open('element_'+str(element_sets)+'.txt','w')
    element_counter = 0
    for elements in odb.rootAssembly.instances['PART-1-1'].elementSets[element_sets].elements:
        if str(elements.type) == 'C3D8':
            element_counter += 1
            if element_counter == 1:
                element_set_file.write(str(elements.label)+',\t')
            else:
                element_set_file.write('\n'+str(elements.label)+',\t')
            for conn in elements.connectivity:
                element_set_file.write(str(conn)+',\t')
        elif str(elements.type) == 'C3D6':
            element_counter += 1
            if element_counter == 1:
                element_set_file.write(str(elements.label)+',\t')
            else:
                element_set_file.write('\n'+str(elements.label)+',\t')
            for conn in range(0,8):
                element_set_file.write('0'+',\t') #C3D6 werden nicht die Nachbarn, sondern Nullen eingetragen, damit bemerkt wird, wenn ein Wedge-Element im 'adaptive-mesh sektor' ist
            print('\n WARNING: \n Elements of type C3D6 detected! They must not be in the neighbor node section!')
        else:
            print('\n WARNING: \n Elements of type '+str(elements.type)+' detected!')
    set_file.write(str(element_counter)+'\n')
    element_set_file.close()
```

```
parameter_file.write('ELEMENT_SETS,\n'+str(set_counter)+'\n')
set_file.close()
# Prepare empty file for the UVARM Variable - containing the right amount of zeros:
uvarm_file = open('UVARM1.txt','w')
for i in range(0,contact_node_nr,1):
    for j in range(0,parameter.neighborNumber+1,1):
        uvarm_file.write('0.0,\t')
    uvarm_file.write('\n')
uvarm_file.close()
parameter_file.close()
print('get_set_info.py - DONE.\n\n')
```

Find the neighbour nodes

This Python script finds the neighbour nodes of the contact nodes.

```
import sys                          # operating system
import argparse                     # ability to use and parse arguments
import collections                  # for counting arguments etc. in a list
import numpy                        # for calculation of median
from odbAccess import *             # ability to work with an Abaqus - ODB file
from math import sqrt
from datetime import datetime
from collections import Counter

start_time = datetime.now()
print("Starting time = "+str(start_time))

class Parameters:
    pass
parameter = Parameters()    # instance of the class Parameters
# Parsing arguments
parser = argparse.ArgumentParser(description="Writing the input file for the Abaqus analysis of a sub model.")
parser.add_argument('--path',                 type=str, required=True, help='Path of the working directory.')
parser.add_argument('--jobname',              type=str, required=True, help='Name of the job, where the neighbor nodes will be searched.')
```

```python
parser.add_argument('--contactNodeSetName', type=str, required=True, help='Name of the contact node set. These nodes will get neighbors.')
parser.add_argument('--wearElementSetName', type=str, required=True, help='Name of the element set which is affected by mesh smoothing via UMeshMotion later.')
parser.add_argument('--neighborNumber',     type=int, required=True, help='Number of neighbor nodes, which will be calculated here.' )
parser.add_argument('--modelType',          type=str, required=True, help='Two model types are accepted: 3D and AX. 3D means a 3D Simulation with the z-axis being the length axis of the rod, AX means axisymmetrical simulation, where x is the radial direction, y in rod direction.')
args = parser.parse_args(namespace=parameter)
#
odb = openOdb(path=parameter.path+"/"+parameter.jobname+'_0.odb')
print('find_node_neighbors.py - START.')
if parameter.contactNodeSetName not in odb.rootAssembly.instances['PART-1-1'].nodeSets.keys():
    print('\n\n\n * * * E R R O R in find_node_neighbors.py * * * \n\n\n'+'find_node_neighbors.py cannot find the given node set called: '+parameter.contactNodeSetName+'\n\n')
elif 'INNEN_ADA_MESH' not in odb.rootAssembly.instances['PART-1-1'].elementSets.keys():
    print('\n\n\n * * * E R R O R in find_node_neighbors.py * * * \n\n\n'+'find_node_neighbors.py cannot find the given element set called: INNEN_ADA_MESH \n\n')

#print('neighbor number = '+str(parameter.neighborNumber))

if parameter.modelType == 'AX':
    sys.exit('THIS NEIGHBORNODE SEARCH IS FOR 3D MODELS ONLY!')

# the following dictionaries work only for models containing 1 instance named ['PART-1-1']
# creating a dictionary for all nodes of the model, the node.label is the key, the node object the value
node_dictionary = {}
for all_nodes in odb.rootAssembly.instances['PART-1-1'].nodeSets['INNEN_ADA_MESH'].nodes:
    node_dictionary.update({all_nodes.label:all_nodes})
# creating a dictionary for all elements, the node.label is the key, the node object the value
element_dictionary = {}
```

```
for all_elements in odb.rootAssembly.instances['PART-1-1'].
 ↪  elementSets['INNEN_ADA_MESH'].elements:
    element_dictionary.update({all_elements.label:all_elements})
# creating a list which contains all contact nodes and their
 ↪  neighbors
# 1st entry = contact node
# the following entries per sub-list are the neighbors
neighbors = []
#list containing the increment of numbers between the node and the
 ↪  next neighbor
delta_node_number = []
#gap/distance between nodes (total gap), used for proof if a
 ↪  suggestion concerning the node neighbor is correct
delta_gap_between_nodes = []
# Finding neighbors in radial direction:
# This procedure works only if the node numbers of the whole model
 ↪  go from 1 to n ?!
# This procedure works for problems where the nodes are placed
 ↪  cylindrically in space. The z coordinate is not considered in the
 ↪  choosing phase, because
# the correct neighbor node has to have the same z-coordinate (+-2%)
 ↪  as the current_node.
contact_node_counter = -1                                 #
 ↪  contact_node_counter counts the contact nodes for positioning
 ↪  correctly in the array: neighbors
suggestion = False                                        # suggestion is a
 ↪  boolean which determinates if a suggestion is going to be made or
 ↪  not
first_time = True                                         # first_time is a
 ↪  boolean which determinates if an action is performed the 1st
 ↪  time
suggestion_1_counter = 0
suggestion_2_counter = 0
for contact_nodes in odb.rootAssembly.instances['PART-1-1'].
 ↪  nodeSets[parameter.contactNodeSetName].nodes:
    contact_node_counter += 1
#
#     print('contact_node_counter = '+str(contact_node_counter))
#     print('current contact node considered =
 ↪  '+str(contact_nodes.label))
    current_node = contact_nodes
     ↪  # current node is one of the contact nodes
    i = 0
    neighbors.append([contact_nodes.label])

    while i < parameter.neighborNumber:
     ↪  # repeat for the number of neighbor nodes per contact node
#        print('Searching neighbor for Node
 ↪  '+str(current_node.label))
# after 2 runs of searching the whole database a suggestion for the
 ↪  next neighbor node can be made, to speed up the process of
 ↪  searching
# the most_common_delta_node_number as well as the 2nd most common
 ↪  node number are recalculated every 10th loop
        if (contact_node_counter == 3 or contact_node_counter%10==0)
         ↪  and contact_node_counter != 0:
            counted_delta_node_numbers = Counter(delta_node_number)
            most_common_delta_node_number = [0,0]
            most_common_delta_node_number[0] =
             ↪  counted_delta_node_numbers.most_common()[0][0]
             ↪  # calculation of the most common increment between
             ↪  the neighbor nodes
            most_common_delta_node_number[1] =
             ↪  counted_delta_node_numbers.most_common()[1][0]
             ↪  # calculation of the second most common increment
             ↪  between the neighbor nodes
            median_displacement =
             ↪  numpy.median(delta_gap_between_nodes)
             ↪  # calculation of the median of the gaps between the
             ↪  nodes (average node displacement)
#            print('the most common increments are
 ↪  ='+str(most_common_delta_node_number[0])+' and
 ↪  '+str(most_common_delta_node_number[1])+'   the displacement
 ↪  median = '+str(median_displacement))
        if contact_node_counter > 2:
            suggestion = True                                        #
             ↪  <==  # this is the switch which allows to use the
             ↪  suggestion modus for faster searching

# SUGGESTION OF A NEIGHBOR NODE FOR SEARCH ACCELERATION:
# this algorithm is designed for meshes which have their node
 ↪  neighbors radially.
# in case of a 3D model
# the following suggestion is only valid for a ring, where the
 ↪  rotation axis is the Z axis and the neighbors have the same z
 ↪  coordinate
        if (suggestion == True) and (parameter.modelType == '3D'):
            first_time = True
            most_common_delta = most_common_delta_node_number[0]
            while suggestion == True:
#                print('Current node = '+str(current_node.label))
                if first_time == False:
                    most_common_delta =
                    ↪  most_common_delta_node_number[1]
                    ↪  # Second guess - if the most common guess
                    ↪  goes wrong - second common is tried

                if (current_node.label + most_common_delta) in
                ↪  node_dictionary:
                    suggested_neighbor =
                    ↪  node_dictionary[current_node.label +
                    ↪  most_common_delta]         # suggested
                    ↪  neighbor = current node number + median of
                    ↪  the increment of node numbers

                   # print('Suggested node =
                   ↪  '+str(suggested_neighbor.label))
                   # print('a suggestion for 3D is
                   ↪  made:'+str(suggested_neighbor.label))

                    # further the z-coordinate of a standard ring is
                    ↪  as a maximum 8mm, so if this coordinate is
                    ↪  bigger than 8.3 an input-error occured
                    if current_node.coordinates[2] > 8.3:
                         sys.exit('\n\nERROR - the z (axial)
                         ↪  coordinate of the ring was bigger or
                         ↪  equal to 8.3mm. \n Check if the model has
                         ↪  the z-axis as rod-axis. This might be an
                         ↪  input error. \n If a ring of thickness
                         ↪  greater than 8mm shall be calculated,
                         ↪  change the limit accordingly.\n\n')
                    # the suggestion of a node can only be correct if
                    ↪  the z (axial) coordinate is equal +- 0.01
                    if abs(current_node.coordinates[2] -
                    ↪  suggested_neighbor.coordinates[2]) < 0.01:
                        length_current_node =
                        ↪  sqrt(current_node.coordinates[0]**2+
                        ↪  current_node.coordinates[1]**2)
                        ↪  # lenght of the coordinate vector of the
                        ↪  current_node
                        length_sugg_node     = sqrt(suggested_neighbor.coordinates[0]**2+ suggested_neighbor.coordinates[1]**2)  # lenght of the coordinate vector of the suggested_neighbor
                        # print('x coord current: '+ str(current_node.coordinates[0])+' suggestion: '+str(suggested_neighbor.coordinates[0]))
                        # print('y coord current: '+ str(current_node.coordinates[1])+' suggestion: '+str(suggested_neighbor.coordinates[1]))
                        # print('z coord current: '+ str(current_node.coordinates[2])+' suggestion: '+str(suggested_neighbor.coordinates[2]))
                        # print('length current: '+ str(length_current_node)+' suggestion: '+str(length_sugg_node))
                        # print('x coord normiert: current '+ str(current_node.coordinates[0]/ length_current_node)+ '   suggestion: '+str(suggested_neighbor.coordinates[0]/ length_sugg_node))
                        # print('y coord normiert: current '+ str(current_node.coordinates[1]/ length_current_node)+ '   suggestion: '+str(suggested_neighbor.coordinates[1]/ length_sugg_node))

                        # test if the node and the neighbor have the same unity vector radially (the z-coordinate is not considered,
                        # but was considered previously: the z-coordinate also must be equal), tolerance = 0.01
                        # further the radial distance between the suggested and the current node must not be greater than 1.2*median_displacement and not smaller than 0.8*median_displacement
                        # further the radial distance of the neighbor must be greater than the radial distance of the neighbor
```

```
                         distance_radial = sqrt((current_node.coordinates[0]-suggested_neighbor.coordinates[0])**2+(current_node.coordinates[1]-suggested_neighbor.coordinates[1])**2)
                         distance_zaxis_to_current_node = sqrt((current_node.coordinates[0])**2+(current_node.coordinates[1])**2)
                         distance_zaxis_to_suggested_neighbor = sqrt((suggested_neighbor.coordinates[0])**2+(suggested_neighbor.coordinates[1])**2)
                        # print('radial distance between the current and suggsted node = '+str(distance_radial))
                         if (abs(current_node.coordinates[0]/length_current_node-suggested_neighbor.coordinates[0]/length_sugg_node) < 0.012) and \
                            (abs(current_node.coordinates[1]/length_current_node-suggested_neighbor.coordinates[1]/length_sugg_node) < 0.012) and \
                            (distance_radial < 1.5*median_displacement) and \
                            (distance_radial > 0.8*median_displacement) and \
                            (distance_zaxis_to_current_node < distance_zaxis_to_suggested_neighbor) and \
                            (abs(distance_zaxis_to_suggested_neighbor-distance_zaxis_to_current_node) > 0.45*median_displacement):

                            # update the distance between nodes and the delta in node numbers
                             delta_coordinates = current_node.coordinates-suggested_neighbor.coordinates
                             delta_gap_between_nodes.append(sqrt(delta_coordinates[0]**2 + delta_coordinates[1]**2 + delta_coordinates[2]**2))
                             delta_node_number.append( -
                              ↪  current_node.label +
                              ↪  suggested_neighbor.label)

                             # taking the suggested node for the next
                             ↪  neighbor
                             current_node = node_dictionary
                              ↪  [suggested_neighbor.label]
                             neighbors[contact_node_counter].
                              ↪  append(current_node.label)

                             # additional information
                             if first_time == True:
                                 # print('suggestion accepted - 1st
                                  ↪  guess was right!')
                                 suggestion_1_counter += 1
                             else:
                                 # print('suggestion accepted - 2nd
                                  ↪  guess was right!')
                                 suggestion_2_counter += 1
                             i = i + 1
                             suggestion = True
                             break

                         else:
                             print('Suggestion was wrong - (node
                              ↪  '+str(suggested_neighbor.label)+')')

                             # ========DEBUGGING=========
                             # if (abs(current_node.coordinates[0]
                         ↪  /length_current_node-
                         ↪  suggested_neighbor.coordinates[0]/
                         ↪  length_sugg_node) >= 0.025):
                         #       print('Possibility 1 x coord:
                         ↪  '+str(abs(current_node.coordinates[0]/
                         ↪  length_current_node-
                         ↪  suggested_neighbor.coordinates[0]/
                         ↪  length_sugg_node))+' > 0.01 =
                         ↪  tolerance.')
                         # if (abs(current_node.coordinates[1]/
                         ↪  length_current_node-
                         ↪  suggested_neighbor.coordinates[1]/
                         ↪  length_sugg_node) >= 0.025):
```

```
                           #    print('Possibility 2 y coord:
                           ↪  '+str(abs(current_node.coordinates[1]/
                           ↪  length_current_node-
                           ↪  suggested_neighbor.coordinates[1]/
                           ↪  length_sugg_node))+' > 0.01 =
                           ↪  tolerance.')
                           # if (distance_radial >=
                           ↪  1.5*median_displacement) or
                           ↪  (distance_radial <=
                           ↪  0.8*median_displacement):
                           #      print('Possibility 3 distance
                           ↪  between nodes: 0.8*median_displacement
                           ↪  = '+ str(0.8*median_displacement)+ ' <
                           ↪  '+str(distance_radial)+' <
                           ↪  '+str(1.5*median_displacement)+' =
                           ↪  1.5*median_displacement.')
                           # if distance_zaxis_to_current_node >=
                           ↪  distance_zaxis_to_suggested_neighbor:
                           #      print('Possibility 4 radial
                           ↪  position of inner node = '+
                           ↪  str(distance_zaxis_to_current_node)+'
                           ↪  > '+ str(distance_zaxis_to
                           ↪  _suggested_neighbor)+' = outer node.
                           ↪  Contradiction!')
                           # if abs(
                           ↪  distance_zaxis_to_suggested_neighbor
                           ↪  -distance_zaxis_to_current_node) <=
                           ↪  0.45*median_displacement:
                           #      print('Possibility 5 the radial
                           ↪  position ('+str(
                           ↪  distance_zaxis_to_suggested_neighbor
                           ↪  -distance_zaxis_to_current_node)+')
                           ↪  was smaller than
                           ↪  0.3*median_displacement =
                           ↪  '+str(0.45*median_displacement))
                           # ========DEBUGGING=========
                            if first_time == True:
                                first_time = False
                            else:
                                suggestion = False

                    else:
                        # print('Suggestion (node
                        ↪   '+str(suggested_neighbor.label)+') was
                        ↪   wrong - z coordinates not equal.')
                        #============================
                        # print('delta z =
                        ↪   '+str(abs(current_node.coordinates[2] -
                        ↪   suggested_neighbor.coordinates[2]))+' >
                        ↪   0.01 = tolerance')
                        #============================

                        if first_time == True:
                            first_time = False
                        else:
                            suggestion = False

                else:
                   # print('Suggested node not available.')

                    if first_time == True:
                        first_time = False
                    else:
                        suggestion = False

        if suggestion == False:
           # print('looking for neighbor of node
           ↪   '+str(current_node.label))
           # print('this is the '+str(i)+' neighbor of the contact
           ↪   node '+str(contact_nodes.label))
            i = i + 1
            ↪   # counter for the neighbor nodes
            current_element_list = {}
            ↪   # contains all elements which are in connected to the
            ↪   current_node
            for elements in odb.rootAssembly.instances['PART-1-1'].
            ↪   elementSets[parameter.wearElementSetName].elements:
                if current_node.label in elements.connectivity:
                ↪   # checking if the elements of the
                ↪   wearElementSetName are connected with the current
                ↪   node
                   # print('element found '+str(elements.label))
                    current_element_list.update(
                    ↪   {elements.label:elements})          # in
                    ↪   connected => updating the dictionary
##care much output:
```

```
                # print('current element list = '+str(current_element_list))
                neighbor_node_list = {} # contains all nodes of the elements which are connected to the current_node, except the current_node
            for element in current_element_list:
##care much output:
                # print('element of current_element_list: '+str(current_element_list[element]))
                for n in current_element_list[element].connectivity:
                    # print('current connectivity element investigated = '+str(current_element_list[element].label)+' -> Node: '+str(n))
                    if n != current_node.label:
                        neighbor_node_list.update({n:node_dictionary[n]})
                        # print('Possible neighbors of Node: '+str(current_node.label)+' are the nodes '+str(neighbor_node_list.keys()))

# Finding the neighbor node through the coordinates
# the 3D MODEL

            if parameter.modelType == '3D':
                smallest_distance = 0                              # calculation of the smallest distance between the current_node and the other neighbor nodes
                for all_nodes in neighbor_node_list:               # when iterating over a dictionary the keys are returned
                    distance = sqrt((current_node.coordinates[0]-neighbor_node_list[all_nodes].coordinates[0])**2+(current_node.coordinates[1]-neighbor_node_list[all_nodes].coordinates[1])**2+(current_node.coordinates[2]-neighbor_node_list[all_nodes].coordinates[2])**2)
                    if (smallest_distance != 0) and (smallest_distance > distance):
                        smallest_distance = distance
                    elif smallest_distance == 0:
                        smallest_distance = distance
                # print('smallest_distance between current_node and
                ↪   the nearest neighbor:'+str(smallest_distance))
                # if the radial distance to the rotation center
                ↪   (z-axis assumed) of the current_node +
                ↪   0.1*smallest_distance is bigger than the radial
                ↪   distance form the rotation center (z-axis
                ↪   assumed) of the neighbor node,
                # the neighbor node is not further away from the
                ↪   rotation axis than the current_node => will not
                ↪   be the next neighbor => pop item from dictionary
                # further if the z-coordinate is not == z-coordinate
                ↪   of the current_node +- 0.1*smallest_distance is
                ↪   bigger than the radial distance form the rotation
                ↪   center (z-axis assumed) of the neighbor node =>
                ↪   will not be the next neighbor => pop item from
                ↪   dictionary
                for all_nodes in neighbor_node_list.keys():
                    if (neighbor_node_list[all_nodes].
                    ↪   coordinates[0]**2
                    ↪   +neighbor_node_list[all_nodes].
                    ↪   coordinates[1]**2)**0.5 <
                    ↪   ((current_node.coordinates[0]**2
                    ↪   +current_node.coordinates[1]**2)**0.5
                    ↪   +0.3*smallest_distance):
                        del neighbor_node_list[all_nodes]
                    elif (neighbor_node_list[all_nodes].
                    ↪   coordinates[2] - current_node.coordinates[2])
                    ↪   > 0.1*smallest_distance:
                        del neighbor_node_list[all_nodes]
                # print('after poping out not wanted neighbors =>
                ↪   neighbor_node_list = '+str(neighbor_node_list))
                # the neighbor with the smallest distance to the
                ↪   current_node should be the wanted neighbor node
                smallest_distance = 0
                j = 0
                while j < 2:
                    j = j + 1
                    for all_nodes in neighbor_node_list.keys():
                        distance = sqrt((current_node.coordinates[0]-
                         ↪  neighbor_node_list[all_nodes].
                         ↪  coordinates[0])**2+
                         ↪  (current_node.coordinates[1]-
                         ↪  neighbor_node_list[all_nodes].
                         ↪  coordinates[1])**2+
                         ↪  (current_node.coordinates[2]-
                         ↪  neighbor_node_list[all_nodes].
                         ↪  coordinates[2])**2)
                        if (smallest_distance != 0) and
                         ↪  (smallest_distance > distance):
                            smallest_distance = distance
                        elif (smallest_distance != 0) and
                         ↪  (smallest_distance < distance):
                            # print('eliminated node:
                             ↪  '+str(neighbor_node_list[all_nodes]))
                            del neighbor_node_list[all_nodes]
                        elif smallest_distance == 0:
                            smallest_distance = distance


# ERRORS:
# A)    if there is more than 1 key in the neighbor_node_list left
# B)    if there is no key in the neigbhor_node_list
#    => the watchdog has to raise some ERROR
            watchdog = 0
            for key in neighbor_node_list.keys():
                if type(key) == int:
                    watchdog += 1
                    # ========DEBUGGING=========
                    # print(' THE WINNER OF THE NEIGHBOR NODE CONTEST
                     ↪  FOR NODE '+str(current_node.label)+' IS ....
                     ↪  ' + str(neighbor_node_list[key].label))
                    # print('coords current
                     ↪  '+str(current_node.coordinates)+'  coords
                     ↪  neighbor
                     ↪  '+str(neighbor_node_list[key].coordinates))
                    # print('coords current quadrat
                     ↪  '+str(current_node.coordinates**2)+'  coords
                     ↪  neighbor quadrat '+
                     ↪  str(neighbor_node_list[key].coordinates**2))
                    # print('delta coords quadrat
                     ↪  '+str((current_node.coordinates-
                     ↪  neighbor_node_list[key].coordinates)**2))
                    # ========DEBUGGING=========
                    a = (current_node.coordinates-neighbor_node_list[key].coordinates)**2
                    # print('sum of coords quadrat '+str(a[0]+a[1]+a[2]))
                    b = sqrt(a[0]+a[1]+a[2])
                    # print('delta distance = sqrt(sum)'+str(b))
                    # gap/displacement of the current and the neighbor node
                    delta_coordinates = current_node.coordinates-neighbor_node_list[key].coordinates
                    delta_gap_between_nodes.append(sqrt(delta_coordinates[0]**2 + delta_coordinates[1]**2 + delta_coordinates[2]**2))
##care much output:
                    # print(delta_gap_between_nodes)

                    # difference in node numbers between the current and the neighbor node
                    # print('current node label:'+str(current_node.label))
                    # print('neighbor node list:'+str(neighbor_node_list[key].label))
                    # print('delta node number = '+str(-current_node.label + neighbor_node_list[key].label))
                    delta_node_number.append( - current_node.label + neighbor_node_list[key].label)
##care much output:
                    # print('delta node number ='+str(delta_node_number))

                current_node = neighbor_node_list[key]
                neighbors[contact_node_counter].append(current_node.label)
            if watchdog > 1:
                sys.exit('\n\n\nERROR - there was more than 1 neighbor found - ALARM: watchdog = '+str(watchdog)+'\n\n\n')
            elif watchdog == 0:
                sys.exit('\n\n\nERROR - there was no neighbor node found for node '+str(current_node.label)+'\n Hint: maybe the region for the neighbor nodes does not contain the next wanted neighbor.\n\n')
            elif watchdog == 1:
               if suggestion == False and contact_node_counter > 2:
                      if current_node.label == suggested_neighbor.label:
                             # print('although the suggestion was initially considered false it turned out to be correct!\nSwitching suggestion-mode on again. Setting: suggestion = True')
                             suggestion = True


# After finding all neighbor nodes they are printed to a file: neighbor_nodes_JOBNAME.txt
node_file = open('neighbor_nodes_'+str(parameter.jobname)+'.txt', 'w')

# Check if there are no duplicates in the list, which contains the neighbor nodes:
# 1st putting all node numbers into one array:
all_node_numbers = []

for lines in neighbors:
    for nodes in lines:
        node_file.write(str(nodes)+',\t')
        all_node_numbers.append(nodes)
    node_file.write('\n')
node_file.close()

# Sort the list with all neigbors and check for duplicates:
# If any duplicates are found print a warning message. (Thank you stackoverflow - question 9835762)
def list_duplicates(seq):
  seen = set()
  seen_add = seen.add
  # adds all elements it doesn't know yet to seen and all other to seen_twice
  seen_twice = set( x for x in seq if x in seen or seen_add(x) )
  # turn the set into a list (as requested)
  return list( seen_twice )

```

```
def list_without_duplicates(seq):
  seen = set()
  seen_add = seen.add
  return [ x for x in seq if not (x in seen or seen_add(x))]

my_duplicates = []
my_duplicates = list_duplicates(all_node_numbers)

contact_nodes_without_clear_neighbors = []

if my_duplicates:
   print("\n\n\n  + + + + +   WARNING - NODE NR DUPLICATES FOUND   + +
    ↪   + + + \n\n Following node numbers occure more than once:
    ↪   \n"+str(sorted(my_duplicates))+"\n\n HINT: Adjust the settings
    ↪   of the tolerances which are used to determine if a suggested
    ↪   node is the real neighbor node. (The tolerances of the advanced
    ↪   guess as well as the tolerances of the 3D neighbor node
    ↪   search.\n\n")
   for lines in neighbors:
      for nodes in lines:
         for doubles in my_duplicates:
            if doubles == nodes:
               contact_nodes_without_clear_neighbors.append(lines[0])
   contact_nodes_without_clear_neighbors =
    ↪   list_without_duplicates(contact_nodes_without_clear_neighbors)
   contact_nodes_without_clear_neighbors.sort()
   print('The contact nodes with unclear neighbors are listed below:
    ↪   \n'+str(contact_nodes_without_clear_neighbors))
   print('The median displacement was '+str(median_displacement)+'.
    ↪   The minimum and maximum displacement are
    ↪   '+str(min(delta_gap_between_nodes))+';
    ↪   '+str(max(delta_gap_between_nodes))+'.')

# determination where the mistake is:
# the delta in node numbers is printed into the file
 ↪   'delta_neighbor_node_nr_JOBNAME.txt'
# to get the delta the node nr of the previous node has to be
 ↪   subtracted from the current node number
# example:  contact node, neighbor1, neighbor2, neighbor3   => delta
 ↪   (regular pattern, 1st row = 0, nothing to subtract)
#           123,          1030,       1040,       1050     0,    0,    0,
 ↪   0
#           124,          1033,       1043,       1053     1,    3,    3,
 ↪   3
```

```
#           125,          1036,      1046,      1056    1,   3,   3, 3
# through the deltas a pattern shall be visualized - the pattern will be broken by the false neighbors

# the file contains following information:
# delta between contact nodes, delta between contact and neighbor, n times delta between neighbors
   delta = []
   delta_line = []
   for i in range(0,len(neighbors)):
      for j in range(0,len(neighbors[0])):
         if i == 0:
            delta_line.append(0)
         else:
            delta_line.append(neighbors[i][j]-neighbors[i-1][j])
      delta.append(delta_line)
      delta_line = []

   delta_node_nr_file = open('delta_neighbor_node_nr_'+str(parameter.jobname)+'.txt', 'w')

   for line in delta:
       for nodes in line:
           delta_node_nr_file.write(str(nodes)+',\t')
       delta_node_nr_file.write('\n')
   delta_node_nr_file.close()
   print("\n For easier checking the difference between a contact node and its neighbors and the previous contact node and its neighbors is calculated and printed to the file called: delta_neighbor_node_nr_"+str(parameter.jobname)+".txt \n")

   # determination which nodes are in the adaptive mesh segement, but were not used in as neighbor nodes
   # this only makes sense  IF all nodes in the mesh segment should be used as neighbor nodes. (e.g.: If just 3 out of 5 possible neighbors are used ther will be a lot of output)

else:
   print("No duplicates in the node numbers were found, every node appears only once. \n Detailed data below.\n")

end_time = datetime.now()
```

```
print("Neighbor nodes found. Current date and time: "+str(end_time))
delta_time = end_time-start_time
print("The search lasted "+str(delta_time))
print("\nAdditional information:")
contact_node_counter = contact_node_counter + 1  # +1 because counter started at 0
print("Number of contact nodes: \t"+str(contact_node_counter))
print("Number of neighbors: \t"+str(parameter.neighborNumber*contact_node_counter))
print("Number of total suggestions = (contact node number-3) * number of neighbors = "+str((contact_node_counter-3)*parameter.neighborNumber))
print("Number of true suggestions: \t"+str(suggestion_1_counter+suggestion_2_counter))
print('Resulting from: 1st guesses: \t'+str(suggestion_1_counter)+'\t and 2nd guesses: \t'+str(suggestion_2_counter))
print("Most common deltas in node numbers (delta, incidence): \t"+str(counted_delta_node_numbers.most_common()[0])+', \t'+str(counted_delta_node_numbers.most_common()[1])+', \t'+str(counted_delta_node_numbers.most_common()[2]))
print('The script find_node_neighbors.py finished successfully, writing all neighbor nodes of the contact nodes to neighbor_nodes_'+str(parameter.jobname)+'.txt')
```

Input generator for the wear calculation

This code generates an input file for the next Abaqus wear restart calculation.

```
# This Phyton file shall generate a new knotenTempTxt.txt and a new input/restart file:
#
print("enter Main_Input_Generator_BCD_V04.py")
import argparse
from odbAccess import *

class Parameters:
    pass
parameter = Parameters() # instance of the class Parameters
# Parsing arguments
parser = argparse.ArgumentParser(description="Writing the input file for the Abaqus analysis of a sub model.")
# Adding all necessary and possible arguments to the parser.
```

```
parser.add_argument('--path', type=str, required=True, help='Path of the working directory.')
parser.add_argument('--counter', type=int, required=True, help='Counter counts the number of fully run cycles.')
parser.add_argument('--jobmain', type=str, required=True, help='Name of the sub job containing the local model. This model is used repeatedly.')
parser.add_argument('--jobsub', type=str, required=True, help='Name of the sub job containing the local model. This model is used repeatedly.')
parser.add_argument('--inpmain', type=str, required=True, help='Name of the input file for the sub model job.')
parser.add_argument('--inpgenmain', type=str, required=True, help='Name of the input file generator for the sub model.')
parser.add_argument('--tempfilename', type=str, required=True, help='Name of the *.txt file where all temperature values for all nodes are stored.')
parser.add_argument('--wearElementSetName', type=str, required=True, help='Name of the element set which is affected by mesh smoothing via UMeshMotion later.')
parser.add_argument('--inc', type=float, required=True, help='This is the increment which is maximaly used when calculating the wear.')
parser.add_argument('--highpressuretemp', type=float, required=True, help='Pressure of the cylinder side. This is the higher pressure. The maximum is 10 MPa (100bar). This maximum was agreed upon on 13.08.2015. This value is considered as the high temperature of the submodel calculation.')
parser.add_argument('--lowpressuretemp', type=float, required=True, help='Pressure of the crank side. This is the lower pressure. The minimum is 0 MPa (0bar). This value is considered as the low temperature of the submodel calculation.')

args = parser.parse_args(namespace=parameter)
odb = openOdb(path=parameter.path+"/"+parameter.jobsub+'_'+str(parameter.counter)+'.odb')
odbmain = openOdb(path=parameter.path+'/'+parameter.jobmain+'_'+str(parameter.counter-1)+'.odb')
# print(parameter.path+"/"+parameter.jobsub+'_'+str(parameter.counter)+'.odb')

# Write all Node-Temperatures to a file.
knotenTempTxt = open(parameter.path+'/'+parameter.tempfilename+'.txt', 'w')
lastFrame = odb.steps['thermal_analysis'].frames[-1]
```

```
for nt11 in lastFrame.fieldOutputs['NT11'].values:
    # print('Node = %d, Temp. = %6.4f'  %(nt11.nodeLabel, nt11.data))
    for nodes in odbmain.rootAssembly.instances['PART-1-1'].nodeSets['INNEN'].nodes:
        if nt11.nodeLabel == nodes.label:
            # The temp or pressure value is offset so that the distribution falls from highpressuretemp-lowpressuretemp to 0.
            temp_value = nt11.data-parameter.lowpressuretemp
            if temp_value < 0:   # correct rounding/truncation errors
               temp_value = 0.0
               print('Temperature / Pressure value < 0 - reset to 0 for node '+str(nt11.nodeLabel)+' with the corrected value '+str(temp_value)+'\nCorrected by '+str(parameter.lowpressurevalue)+'.\n')
            knotenTempTxt.write('%d, %8.6f\n'  %(nt11.nodeLabel, nt11.data-parameter.lowpressuretemp))
print('\nINFORMATION:\nThe pressure (temperature) distribution has been offset (reduced) by the low pressure (temperature) value: '+str(parameter.lowpressuretemp)+'\n\n')
knotenTempTxt.close()
#
#
print('Generation of knotenTempTxt successfully finished!')
# ----- Node Temperatures written -----

print('Start Main-Restart-File generation.')
restartFile = open(str(parameter.path)+'/'+str(parameter.inpmain)+'_'+str(parameter.counter)+'.inp', 'w')
restartFile.write('**This file is generated through the Phyton file '+str(parameter.inpgenmain)+'.py.\n')
restartFile.write('*HEADING\n')
restartFile.write('**Restart the Analysis\n')
restartFile.write('*RESTART, WRITE, FREQUENCY=1, OVERLAY, READ, END STEP\n')
restartFile.write('** AMPLITUDE=STEP applies the load instantaneously at the start of the step\n')
restartFile.write('*STEP, NAME = Verschleiss, nlgeom = YES, AMPLITUDE = STEP, INC = 2000\n')
restartFile.write('*VISCO, STABILIZE, CETOL=0.01\n')
restartFile.write(str(parameter.inc)+' , 120000, 1e-3, '+str(100*parameter.inc)+'\n')
```

```
# only write the adaptive mesh controls once:
if (parameter.counter==1):
    restartFile.write('*ADAPTIVE MESH CONTROLS, NAME=Ada-1\n0.,1.\n')
restartFile.write('*ADAPTIVE MESH, ELSET='+str(parameter.wearElementSetName)+', CONTROLS=Ada-1, OP=MOD, FREQUENCY=1, MESH SWEEPS=1\n')
restartFile.write('*ADAPTIVE MESH CONSTRAINT, CONSTRAINT TYPE=SPATIAL, TYPE=DISPLACEMENT, USER\n')
restartFile.write(str(parameter.wearElementSetName)+'\n')
restartFile.write('*DSLOAD\n')
restartFile.write('CAP_2Z_SURF_PRESSURE_CAP_HIGH, P, '+str(parameter.highpressuretemp-parameter.lowpressuretemp)+'\n')
restartFile.write('SS_2Z_SURF_PRESSURE_SS_HIGH, P, '+str(parameter.highpressuretemp-parameter.lowpressuretemp)+'\n')
restartFile.write('TEMP_ALL, PNU, 1\n')
restartFile.write('*FIELD, USER\n')
restartFile.write('INNEN_ADA_MESH\n')
restartFile.write('*OUTPUT, HISTORY, FREQUENCY = 1\n')
restartFile.write('*ENERGY OUTPUT, ELSET = EL_ALL, VARIABLE = ALL\n')
restartFile.write('*OUTPUT, FIELD, FREQUENCY = 1\n')
restartFile.write('*ELEMENT OUTPUT, ELSET = EL_ALL\n')
restartFile.write('S, P, FV1, E, LE\n')
restartFile.write('*NODE OUTPUT, NSET = ALLNODES\n')
restartFile.write('CF, RF, U, COORD\n')
restartFile.write('*CONTACT OUTPUT, NSET = ALLNODES\n')
restartFile.write('CDISP, CSTATUS, CSTRESS, CFORCE\n')
restartFile.write('*NODE FILE, NSET = ALLNODES\n')
restartFile.write('COORD\n')
restartFile.write('*CONTACT FILE, NSET = INNEN\n')
restartFile.write('CSTRESS\n')
restartFile.write('*END STEP\n')
restartFile.close()
print('Main-Restart-File successfully written.')
exit()
```

Input generator gas pressure sub-calculation

This file creates an input file for the Abaqus calculation of the gas pressure distribution in the dynamic sealing surface.

```
#!/usr/bin/env python
# This Phyton file shall generate a new input file for the SUBMODEL:
#
```

```python
print("enter Sub_Input_Generator_BCD_V05.py")
import os
import sys
import argparse
import math
from odbAccess import *
#
class Parameters:
    pass
parameter = Parameters()    # instance of the class Parameters
# Parsing arguments
parser = argparse.ArgumentParser(description="Writing the input file for the Abaqus analysis of a sub model.")
# Adding all necessary and possible arguments to the parser.
parser.add_argument('--path', type=str, required=True, help='Path of the working directory.')
parser.add_argument('--counter', type=int, required=True, help='Counter counts the number of fully run cycles.')
parser.add_argument('--jobmain', type=str, required=True, help='Name of the main job containing the global model. This model is used repeatedly.')
parser.add_argument('--jobsub', type=str, required=True, help='Name of the sub job containing the local model. This model is used repeatedly.')
parser.add_argument('--inpsub', type=str, required=True, help='Name of the input file for the sub model job.')
parser.add_argument('--usersub', type=str, required=False,help='Name of the Fortran file, containing the UserDefinedSubroutines for the sub model.')
parser.add_argument('--inpgensub', type=str, required=True, help='Name of the input file generator for the sub model.')
parser.add_argument('--highpressuretemp', type=float, required=True, help='Pressure of the cylinder side. This is the higher pressure. The maximum is 10 MPa (100bar). This maximum was agreed upon on 13.08.2015. This valau is considered as the high temperature of the submodel calculation.')
parser.add_argument('--lowpressuretemp', type=float, required=True, help='Pressure of the crank side. This is the lower pressure. The minimum is 0 MPa (0bar). This value is also the lower temperature in the sub calculation/model. This value has to be smaller than the highpressuretemp.')


args = parser.parse_args(namespace=parameter)
```

```
odb = openOdb(path=str(parameter.path)+"/"+str(parameter.jobmain)+
 ↪  "_"+str(parameter.counter)+'.odb')
#    GENERATION OF THE SUBMODEL INPUT FILE
inp = open(parameter.inpsub+'_'+str(parameter.counter)+'.inp', 'w')
inp.write('**This file is generated through the Phyton file
 ↪  '+str(parameter.inpgensub)+'.py.\n')
inp.write('*HEADING\n')
inp.write('*NODE\n')

# writing the desired nodeSets form the odb (main model) to the
 ↪  input file:
# P R E P E R A T I O N   O F   T H E   D A T A
# 1. for the CAP-Segement
nodeList_contact_nodes_CAP = []
nodeDict_contact_nodes_CAP = {}
for allnodes in
 ↪  odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_CAP'].nodes:
        nodeList_contact_nodes_CAP.append(allnodes.label)
        nodeDict_contact_nodes_CAP.update({allnodes.label:allnodes})

# 2. for the Sealing-Segment (SS)
nodeList_contact_nodes_SS = []
nodeDict_contact_nodes_SS = {}
for allnodes in
 ↪  odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_SS'].nodes:
    nodeList_contact_nodes_SS.append(allnodes.label)
    nodeDict_contact_nodes_SS.update({allnodes.label:allnodes})

lastFrame = odb.steps['Verschleiss'].frames[-1]

nodeDict_all_nodes = {}
# Dictionary which contains all nodes used in the sub-model
 ↪  calculation and their coordinates
# To get the coordinates use nodeDict_all_nodes[node nr][0/1/2] with
 ↪  (0=x,1=y,2=z)
# Searching for the nodes and their coordinates of the desired
 ↪  element-sets
for nodes in lastFrame.fieldOutputs['COORD'].values:
    if nodes.nodeLabel in nodeList_contact_nodes_CAP:
        inp.write(str(nodes.nodeLabel)+', '+',
         ↪  '.join(map(str,nodes.data[0:3]))+'\n')
        nodeDict_all_nodes.update({nodes.nodeLabel:nodes.data})
    elif nodes.nodeLabel in nodeList_contact_nodes_SS:
        inp.write(str(nodes.nodeLabel)+', '+', '.join(map(str,nodes.data[0:3]))+'\n')
        nodeDict_all_nodes.update({nodes.nodeLabel:nodes.data})

# Getting the current position of the reference nodes for the boundary condition:
# 1. CAP
if len(odb.rootAssembly.instances['PART-1-1'].nodeSets['CAP_REF_POS_TEMP'].nodes) > 1:
    sys.exit('\n\nERROR in Sub_Input_Generator -\n There is more than one node in the CAP_REF_POS_TEMP set.\n\n')
else:
   cap_ref_pos_nr = odb.rootAssembly.instances['PART-1-1'].nodeSets['CAP_REF_POS_TEMP'].nodes[0].label

# 2. SS
if len(odb.rootAssembly.instances['PART-1-1'].nodeSets['SS_REF_POS_TEMP'].nodes) > 1:
    sys.exit('\n\nERROR in Sub_Input_Generator -\n There is more than one node in the SS_REF_POS_TEMP set.\n\n')
else:
    ss_ref_pos_nr = odb.rootAssembly.instances['PART-1-1'].nodeSets['SS_REF_POS_TEMP'].nodes[0].label

# sanity checks:
# no more than one node in the REF_POS_TEMP sets => test performed above
# the z-coordinates of the two REF_POS_TEMP nodes have to be (nearly) equal
if abs(nodeDict_all_nodes[cap_ref_pos_nr][2] - nodeDict_all_nodes[ss_ref_pos_nr][2]) > 0.15:
   if parameter.counter == 1:
    sys.exit('\n\nERROR in Sub_Input_Generator -\n The z-coordinates of the REF_POS_TEMP nodes are not equal. Difference = '+str(abs(nodeDict_all_nodes[cap_ref_pos_nr][2] - nodeDict_all_nodes[ss_ref_pos_nr][2]))+'\n\n')
   else:
    print('\n\nWARNING from Sub_Input_Generator -\n The z-coordinates of the REF_POS_TEMP nodes are not equal. Difference = '+str(abs(nodeDict_all_nodes[cap_ref_pos_nr][2] - nodeDict_all_nodes[ss_ref_pos_nr][2]))+'\n\n')

# Write the node sets to the (sub-) input file
inp.write('*NSET, NSET = NODES_CAP')
```

```
k=-1
for allnodes in odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_CAP'].nodes:
    k+=1
    if (k%16) == 0:
        inp.write("\n"+str(allnodes.label)+", ")
    else:
        inp.write(str(allnodes.label)+", ")

inp.write('\n*NSET, NSET = NODES_SS')
k=-1
for allnodes in odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_SS'].nodes:
    k+=1
    if (k%16) == 0:
        inp.write("\n"+str(allnodes.label)+", ")
    else:
        inp.write(str(allnodes.label)+", ")

# here the algorithm checks if the nodes which were initially selected as nodes for the boundary conditions are still valid:

# a) for the CYLINDER SIDE (higher temperature):
# it is checked if the length of the vector from the SS_Ref_Pos to the CAP_Ref_Pos is smaller than the length of the vector from CAP_Ref_Pos to the current node
# vector from the SS to CAP
SS_to_CAP = nodeDict_all_nodes[cap_ref_pos_nr] - nodeDict_all_nodes[ss_ref_pos_nr]
# length of the vector
abs_SS_to_CAP = (SS_to_CAP[0]**2 + SS_to_CAP[1]**2 + SS_to_CAP[2]**2)**0.5

inp.write('\n*NSET, NSET = TEMP_CYLINDER_SIDE')
k=-1
for allnodes in odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_SS'].nodes:
    print_now = False
    if abs(nodeDict_all_nodes[allnodes.label][2] - nodeDict_all_nodes[cap_ref_pos_nr][2]) < 0.1:
                # if the node is in the contact plane of SS and CAP
        SS_to_allnodes = nodeDict_all_nodes[allnodes.label] - nodeDict_all_nodes[ss_ref_pos_nr]
```

```
                                # create vector from the reference node to the current node
        if abs_SS_to_CAP <= (SS_to_allnodes[0]**2 + SS_to_allnodes[1]**2 + SS_to_allnodes[2]**2)**0.5:
                                # and if the node is further away than CAP_Ref_Pos from SS_Ref_Pos the node is a node which has to be in the boundary condition set
            print_now = True
    else:
       for nodes in odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_CYLINDER_SIDE'].nodes:
           if allnodes.label == nodes.label: # the node is not in the contact plane of SS and CAP, but in the TEMP_CYLINDER_SIDE set, it shall carry the boundary condition as initially decided
              print_now = True
              break        # for acceleration --> if node is found in set, there is no need to search further

    if print_now == True:
        k+=1
        if (k%16) == 0:
           inp.write("\n"+str(allnodes.label)+", ")
        else:
           inp.write(str(allnodes.label)+", ")

# Further add all nodes, which are in the TEMP_CYLINDER_SIDE set, but on the CAP Segement
for allnodes in odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_CAP'].nodes:
    print_now  = False
    for nodes in odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_CYLINDER_SIDE'].nodes:
        if allnodes.label == nodes.label:
           print_now = True
           break        # for acceleration (as above)

    if print_now == True:
        k+=1
        if (k%16) == 0:
           inp.write("\n"+str(allnodes.label)+", ")
        else:
           inp.write(str(allnodes.label)+", ")
```

```


# b) for the CRANK SIDE (lower temperature):
move_to_isolation_set = []
inp.write('\n*NSET, NSET = TEMP_CRANK_SIDE')
k=-1
for allnodes in
 ↪  odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_CAP'].nodes:
    print_now = False
    if abs(nodeDict_all_nodes[allnodes.label][2] -
     ↪  nodeDict_all_nodes[ss_ref_pos_nr][2]) < 0.1:
        CAP_to_allnodes = nodeDict_all_nodes[allnodes.label] -
         ↪  nodeDict_all_nodes[cap_ref_pos_nr]
        if abs_SS_to_CAP <= (CAP_to_allnodes[0]**2 +
         ↪  CAP_to_allnodes[1]**2 + CAP_to_allnodes[2]**2)**0.5:
            # abs_SS_to_CAP == abs_CAP_to_SS (PER DEFINITION!)
            print_now = True
    else:
        for nodes in odb.rootAssembly.instances['PART-1-1'].
         ↪  nodeSets['TEMP_CRANK_SIDE'].nodes:
            if allnodes.label == nodes.label:
               print_now = True
               break

    if print_now == True:
        k+=1
        if (k%16) == 0:
            inp.write("\n"+str(allnodes.label)+", ")
        else:
            inp.write(str(allnodes.label)+", ")

for allnodes in
 ↪  odb.rootAssembly.instances['PART-1-1'].nodeSets['TEMP_SS'].nodes:
    print_now = False
    for nodes in odb.rootAssembly.instances['PART-1-1'].
     ↪  nodeSets['TEMP_CRANK_SIDE'].nodes:
        if allnodes.label == nodes.label:
           print_now = True
           break
    # if the sealing segments touch each other the nodes touching
     ↪  loose their boundary condition and the temperature will be
     ↪  calculated there
    # this happens when either x or y coord is equal to (nearly) 0
    if nodeDict_all_nodes[allnodes.label][0] < 0.0001 or nodeDict_all_nodes[allnodes.label][1] < 0.0001:
        print_now = False
        #move_to_isolation_set.append(allnodes.label)

    if print_now == True:
        k+=1
        if (k%16) == 0:
            inp.write("\n"+str(allnodes.label)+", ")
        else:
            inp.write(str(allnodes.label)+", ")

# generation of elements from these nodes - improve for 3D
inp.write('\n** DS4 is a heat transfer 4 node quadrilateral shell element')
inp.write('\n*ELEMENT, TYPE = DS4, ELSET = EL_CAP')
# Below the element definition of all shell elements are defined. Therefore the elements from the Main-file are searched for nodes which are in contact with the rod (node-set-name 'TEMP_SS' and 'TEMP_CAP' see above). So 4 out of 8 nodes are derived and then ordered:
# 1st the smallest z-coordinates on the dictionary (nodeList_temporary) keys 1,2 and then the higher z values 3,4. Afterwards the angle between the node and the xz-plane is calculated and the dictionary is reordered:
# key 1 = smaller z, smaller angle
# key 2 = smaller z, higher angle
# key 3 = higher z, smaller angle
# key 4 = higher z, higher angle
# At last the nodes are written into the SUB-input file in following order: key: 1,3,4,2

# 1st this is done for the CAP-Segment
for elInnen in odb.rootAssembly.instances['PART-1-1'].elementSets['EL_TEMP_CAP'].elements:
    nodeList_temporary = {}
                # temporary dictionary containing the nodes of a shell element
    counter = 0 # counter => has to be 4 to generate a valid element
    for i in range(0,8,1):
        if elInnen.connectivity[i] in nodeList_contact_nodes_CAP:
            counter += 1
            nodeList_temporary.update({counter: nodeDict_contact_nodes_CAP[elInnen.connectivity[i]]})
```

```
if counter != 4:
            # check if counter == 4 else => ERROR-WARNING
    print('ERROR - there were not 4 but '+str(counter)+' nodes
    ↪  found which are in contact. Something went wrong!
    ↪  (sub_inp_generator)')
else:
    j=0
    while j < 4:
        j += 1
        current_node = nodeList_temporary[j]
        # current_node can be swapped if the z-coordinate of the
        ↪  i-node is smaller than that of the current one
        # print('j='+str(j))
        for i in range(j+1,5,1):
            if (abs(current_node.coordinates[2] -
            ↪  nodeList_temporary[i].coordinates[2]) > 0.0001)
            ↪  and ((current_node.coordinates[2] -
            ↪  nodeList_temporary[i].coordinates[2]) > 0):
              nodeList_temporary.update({j:nodeList_temporary[i]})
              nodeList_temporary.update({i:current_node})
              current_node = None
              # print('swap bei i='+str(i)+' und j='+str(j))
              j=0
              # if a swap is performed reset the counter i and j
              ↪  to rerun the swapping process
            break
        for i in range(1,4,2):
         # print(str(i)) # another reordering of the z-pair with
         ↪  smaller z values concerning the angle between the
         ↪  node and the xz plane
        if math.atan(nodeList_temporary[i].coordinates[1]/
        ↪  nodeList_temporary[i].coordinates[0]) >
        ↪  math.atan(nodeList_temporary[i+1].coordinates[1]/
        ↪  nodeList_temporary[i+1].coordinates[0]):
          # print('swap')
                current_node = nodeList_temporary[i]
            nodeList_temporary.update({i:nodeList_temporary[i+1]})
            nodeList_temporary.update({i+1:current_node})
    # print nodes in the right order to get a valid, not twisted
    ↪  element:
    inp.write("\n"+str(elInnen.label))
    inp.write(',\t'+str(nodeList_temporary[1].label))
    inp.write(',\t'+str(nodeList_temporary[2].label))
    inp.write(',\t'+str(nodeList_temporary[4].label))
```

```
        inp.write(',\t'+str(nodeList_temporary[3].label))

inp.write('\n*ELEMENT, TYPE = DS4, ELSET = EL_SS')
# then this is done for the Sealing-Segement
for elInnen in odb.rootAssembly.instances['PART-1-1'].
↪   elementSets['EL_TEMP_SS'].elements:
    nodeList_temporary = {}
                # temporary dictionary containing the nodes of a
                 ↪  shell element
    counter = 0
                # counter => has to be 4 to generate a valid element
    for i in range(0,8,1):
        if elInnen.connectivity[i] in nodeList_contact_nodes_SS:
            counter += 1
            nodeList_temporary.update({counter:
             ↪  nodeDict_contact_nodes_SS[elInnen.connectivity[i]]})
    if counter != 4:
                # check if counter == 4 else => ERROR-WARNING
        print('ERROR - there were not 4 but '+str(counter)+' nodes
         ↪  found which are in contact. Something went wrong!
         ↪  (sub_inp_generator)')
    else:
        j=0
        while j < 4:
            j += 1
            current_node = nodeList_temporary[j]
            # current_node can be swapped if the z-coordinate of the
             ↪  i-node is smaller than that of the current one
            # print('j='+str(j))
            for i in range(j+1,5,1):
                if (abs(current_node.coordinates[2] -
                 ↪  nodeList_temporary[i].coordinates[2]) > 0.0001)
                 ↪  and ((current_node.coordinates[2] -
                 ↪  nodeList_temporary[i].coordinates[2]) > 0):
                    nodeList_temporary.update(
                     ↪  {j:nodeList_temporary[i]})
                    nodeList_temporary.update({i:current_node})
                    current_node = None
                    # print('swap bei i='+str(i)+' und j='+str(j))
                    j=0
                    # if a swap is performed reset the counter i and
                     ↪  j to rerun the swapping process
                break
            for i in range(1,4,2):
```

```
            # print(str(i))
            # another reordering of the z-pair with smaller z values concerning the angle between the node and the xz plane
            if math.atan(nodeList_temporary[i].coordinates[1]/nodeList_temporary[i].coordinates[0]) > math.atan(nodeList_temporary[i+1].coordinates[1]/nodeList_temporary[i+1].coordinates[0]):
              # print('swap')
                    current_node = nodeList_temporary[i]
                nodeList_temporary.update({i:nodeList_temporary[i+1]})
                nodeList_temporary.update({i+1:current_node})
        # print nodes in the right order to get a valid, not twisted element:
        inp.write("\n"+str(elInnen.label))
        inp.write(',\t'+str(nodeList_temporary[1].label))
        inp.write(',\t'+str(nodeList_temporary[2].label))
        inp.write(',\t'+str(nodeList_temporary[4].label))
        inp.write(',\t'+str(nodeList_temporary[3].label))

inp.write('\n*SHELL SECTION, ELSET = EL_CAP, MATERIAL = imaginaryMaterial\n1.,1')
inp.write('\n*SHELL SECTION, ELSET = EL_SS, MATERIAL = imaginaryMaterial\n1.,1')
inp.write('\n*SURFACE, TYPE=ELEMENT, NAME=SURF_TEMP_CAP')
inp.write('\nEL_CAP, SPOS')
inp.write('\n*SURFACE, TYPE=ELEMENT, NAME=SURF_TEMP_SS')
inp.write('\nEL_SS, SPOS')
inp.write('\n*TIE, NAME=TIE_TEMP_SURFS, ADJUST=YES, TYPE=SURFACE TO SURFACE')
inp.write('\nSURF_TEMP_SS, SURF_TEMP_CAP')
inp.write('\n**Material Definition\n')
inp.write('*MATERIAL, NAME=imaginaryMaterial\n')
inp.write('*CONDUCTIVITY\n')
inp.write('0.0001, 0.0001\n')
inp.write('1., 1.\n')
inp.write('1000., 1000.\n')
inp.write('** PHYSICAL CONSTANTS\n')
inp.write('*PHYSICAL CONSTANTS, ABSOLUTE ZERO=0.\n')
inp.write('*INITIAL CONDITIONS, TYPE = TEMPERATURE\n')
inp.write('NODES_CAP, '+str(parameter.highpressuretemp/2)+'\n')
inp.write('NODES_SS,  '+str(parameter.highpressuretemp/2)+'\n')
inp.write('** STEP DEFINITION\n')
inp.write('*STEP, NAME = thermal_analysis, NLGEOM = NO\n')
inp.write('*HEAT TRANSFER, STEADY STATE\n')
inp.write('1., 1., 1.\n')
inp.write('*BOUNDARY\n')
inp.write('TEMP_CYLINDER_SIDE, 11, 11,
↪   '+str(parameter.highpressuretemp)+'\n')
inp.write('TEMP_CRANK_SIDE,    11, 11,
↪   '+str(parameter.lowpressuretemp)+'\n')
inp.write('*OUTPUT, FIELD, FREQUENCY = 1\n')
inp.write('*ELEMENT OUTPUT\n')
inp.write('TEMP\n')
inp.write('*NODE OUTPUT\n')
inp.write('NT\n')
inp.write('*END STEP\n')
inp.close()
```

Acronyms and Symbols

Acronyms

BCD balanced cap design

CF carbon fibre

CMM 3D coordinate measurement machine

FE finite element

FEP fluorinated ethylene propylene

GF glass fibre

HVOF high velocity oxygen fuel spraying

HW HOERBIGER Wien

MoS_2 molybdenum disulfide

PA polyamide

PEEK poylether ether ketone

PPDT poly-*p*-phenyleneterephtalamide

PPS polyphenylene sulfide

PTFE polytetrafluoroethylene

UFD ultra fine diamond

WC tungsten carbide

List of Symbols

The quantities, which seem multiply defined, are all needed to distinguish between dimensionless and dimensional quantities or to differentiate between the integrand and the integration limit.

α angle

A_c contact area

A_s contact area specimen

β angle

b axial ring width

C_0 integration constant

d beam deflection

D rod diameter

$\tilde{D}$ rod diameter

$D^{(\mathrm{i})}$ inner diameter

$D^{(\mathrm{o})}$ outer diameter

ε_b outer fibre strain

ε_L non-dimensional load parameter

ε_{rr} radial strain

ε_t non-dimensional wear parameter

E elastic modulus

$err_\mathrm{threshold}$ error threshold

F_C contact force

$F_{\mathrm{C\,t}}$ total contact force over all ring segments

F_F frictional force

F_S spring force

G shear modulus

$\tilde{G}$ shear modulus

h radial ring height

$\tilde{h}$ (radial) ring height

h_0 original radial ring height

h_{crit} critical radial ring height

h_{s} specimen height

J second momentum of area

k wear coefficient

$\tilde{k}$ wear coefficient

$\tilde{k}_2$ wear coefficient

K virtual wear coefficient

l length

L beam length

μ friction coefficient

M_{b} bending momentum

n_{S} number of springs

n_{seg} number of ring segments

ν Poisson's ratio

$\tilde{\omega}$ angular velocity

p pressure

p_0 pressure asymtotic expansion, first part, fast time scale

p_1 pressure asymtotic expansion, second part, fast time scale

p_{c} contact pressure

$\tilde{p}_{\mathrm{c}}$ contact pressure

$p_{\mathrm{c,flat}}$ contact pressure assuming microscopically perfectly flat surfaces

$p_{\mathrm{c,rough}}$ contact pressure reduced by the gas pressure in the dynamic sealing surface which occurrs due to microscopical surface roughness

p_{crk} crank side pressure

p_{cycle} pressure during a compression cycle

$\tilde{p}_{\mathrm{cycle}}$ pressure during a compression cycle

p_{cyl} cylinder side pressure

$\tilde{p}_{\mathrm{d}}$ discharge pressure

p_{g} absolute gas pressure

p_{L} gas pressure load

φ_{end} angular position of the sealing segment's tip

$\varphi_{\max}$ angular position of the wear maximum

$\tilde{p}_{\mathrm{L}}$ gas pressure load

$p_{\mathrm{non\,per}}$ non-periodic pressure

p_{o} pressure on the outer diameter

p_{p} process pressure

$\tilde{p}_{\mathrm{p}}$ process pressure

p_{per} periodic pressure

$\tilde{p}_{\mathrm{s}}$ suction pressure

P pressure, slow time scale

P_0 pressure asymtotic expansion, first part, slow time scale

P_1 pressure asymtotic expansion, second part, slow time scale

P_{c} contact pressure fast time scale

ρ ratio of $\frac{R^{(\mathrm{o})}}{R^{(\mathrm{i})}}$

r radial coordinate

R_{a} roughness (arithmetical mean deviation of the assessed profil)

$R^{(\mathrm{i})}$ inner radius of a ring

$\tilde{R}^{(\mathrm{i})}$ inner radius of a ring

$R^{(\mathrm{o})}$ outer radius of a ring

$\tilde{R}^{(\mathrm{o})}$ outer radius of a ring

σ_{b} outer fibre strain

S_0 fourier coefficent

$S_{0,\mathrm{nl}}$ fourier coefficent

$S_{n,1}$ fourier coefficent

$S_{n,2}$ fourier coefficent

τ time

t time

Δt time increment

$\tilde{t}$ time

t_0 time zero

$\tilde{t}_0$ time zero

T time

T_g temperature

T_{max} maximum time in calculation

u radial displacement

u_{B} deflection at the beam end

v velocity

$\tilde{v}$ velocity

v_{M} mean velocity

$\tilde{v}_{\mathrm{M}}$ mean rod velocity

V_0 fourier coefficent

$V_{n,1}$ fourier coefficent

$V_{n,2}$ fourier coefficent

V velocity, slow time scale

w wear in the deformed state

$\hat{w}$ wear in the undeformed state

$\tilde{w}$ wear in the deformed state

w_{∞} ultimate wear limit (deformed state)

w_0 wear asymtotic expansion, first part, fast time scale

w_1 wear asymtotic expansion, second part, fast time scale

w_2 wear asymtotic expansion, third part, fast time scale

W wear slow time scale

W_0 wear asymtotic expansion, first part, slow time scale

W_1 wear asymtotic expansion, second part, slow time scale

z axial coordinate

z_{max} maximum distance from neutral fibre to outer fibre

$\frac{\partial p_{\mathrm{A}}}{\partial A_{\mathrm{c}}}$ contact pressure change due to area change

$\frac{\partial p_{\mathrm{f}}}{\partial A_{\mathrm{c}}}$ contact pressure change due to fluid pressure change

List of Figures

List of Tables

Bibliography

[1] J. F. Archard. Contact and rubbing of flat surfaces. *Journal of Applied Physics*, 24(8):981, 1953.

[2] M. A. Ashraf, R. Ahmed, O. Ali, N. H. Faisal, A. M. El-Sherik, and M. F. A. Goosen. Finite element modeling of sliding wear in a composite alloy using a free-mesh. *Journal of Tribology*, 137(3):031605–031605–15, 2015.

[3] T. F. Babuska, A. A. Pitenis, M. R. Jones, B. L. Nation, W. G. Sawyer, and N. Argibay. Temperature-dependent friction and wear behavior of PTFE and MoS2. *Tribology Letters*, 63(2):15, 2016.

[4] P. R. Barry, P. Y. Chiu, S. S. Perry, W. G. Sawyer, S. B. Sinnott, and S. R. Phillpot. Effect of temperature on the friction and wear of PTFE by atomic-level simulation. *Tribology Letters*, 58(3):856, 2015.

[5] B. Bhushan. *Introduction to tribology.* John Wiley & Sons, Ltd, Hoboken and New Jersey, second edition edition, 2013.

[6] T. A. Blanchet and F. E. Kennedy. Sliding wear mechanism of polytetrafluoroethylene (PTFE) and PTFE composites. *Wear*, 153(1):229–243, 1992.

[7] T. A. Blanchet and Y.-L. Peng. Wear resistant irradiated FEP/unirradiated PTFE composites. *Wear*, 214(2):186–191, 1998.

[8] E. M. Bortoleto, A. C. Rovani, V. Seriacopi, F. J. Profito, D. C. Zachariadis, I. F. Machado, A. Sinatora, and R. M. Souza. Experimental and numerical analysis of dry contact in the pin on disc test. *Wear of Materials 2013*, 301(1):19–26, 2013.

[9] B. J. Briscoe, L. H. Yao, and T. A. Stolarski. The friction and wear of poly(tetrafluoroethylene)-poly (etheretherketone) composites: An initial appraisal of the optimum composition. *Wear*, 108(4):357–374, 1986.

[10] D. L. Burris and Sawyer, W. G. A low friction and ultra low wear rate PEEK/PTFE composite. *Wear*, 261(3):410–418, 2006.

[11] X. Chen and S. Hui. Ratcheting behavior of PTFE under cyclic compression. *Polymer Testing*, 24(7):829–833, 2005.

[12] X.-H. Cheng, Y.-J. Xue, and C.-Y. Xie. Friction and wear of rare-earth modified glass-fiber filled PTFE composites in dry reciprocating sliding motion with impact loads. *Wear*, 253(7–8):869–877, 2002.

[13] J.-W. Cho, S. Jeon, H.-Y. Jeong, and S.-H. Chang. Evaluation of cutting efficiency during TBM disc cutter excavation within a korean granitic rock using linear-cutting-machine testing and photogrammetric measurement. *Tunnelling and Underground Space Technology*, 35:37–54, 2013.

[14] J.-W. Cho, S. Jeon, S.-H. Yu, and S.-H. Chang. Optimum spacing of TBM disc cutters: A numerical simulation using the three-dimensional dynamic fracturing method. *Tunnelling and Underground Space Technology*, 25(3):230–244, 2010.

[15] A. W. Christiansen, E. Baer, and S. V. Radcliffe. The mechanical behaviour of polymers under high pressure. *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics*, 24(188):451–467, 1971.

[16] M. Conte and A. Igartua. Study of PTFE composites tribological behavior. *Wear*, 296(1):568–574, 2012.

[17] E. Cummings. Mechanical packing design and theory of operation, 2011.

[18] Dassault Systmes. Abaqus documentation, 2017.

[19] DEWESoft. *version 7.0.6.* DEWESoft, Trbovlje and Slovenia, 2012.

[20] H. Domininghaus, P. Elsner, P. Eyerer, and T. Hirth. *Kunststoffe: Eigenschaften und Anwendungen; mit 275 Tabellen.* Springer, Heidelberg [u.a], 8., neu bearb. und erw. aufl edition, 2012.

[21] W. Eifler, E. Schlücker, U. Spicher, and G. Will. *Küttner Kolbenmaschinen - Kolbenpumpen, Kolbenverdichter: Kolbenpumpen, Kolbenverdichter, Brennkraftmaschinen.* Vieweg+Teubner Verlag, 7 edition, 2009.

[22] N. Feistel. *Beitrag zum Betriebsverhalten trocken laufender Dichtsysteme zur Abdichtung der Arbeitsräume von Kreuzkopfkompressoren.* PhD thesis, Universität Erlangen, Erlangen and Nürnberg, 2001.

[23] N. Feistel. Performance improvement of dry-running sealing systems by optimization of wear compensation. *Conference of the European Forum for Reciprocating Compressors*, 9:239–247, 2014.

[24] R. P. Feynman, R. B. Leighton, and M. Sands, editors. *The Feynman Lectures on Physics.* Addison Wesley, Boston, 2 edition, 2005.

[25] S. Fouvry, C. Paulin, and T. Liskiewicz. Application of an energy wear approach to quantify fretting contact durability: Introduction of a wear energy capacity concept. *Tribology at the Interface: Proceedings of the 33rd Leeds-Lyon Symposium on Tribology (Leeds, 2006)*, 40(10):1428–1440, 2007.

[26] K. Friedrich, Z. Lu, and A. M. Hager. Recent advances in polymer composites' tribology. *Wear*, 190(2):139–144, 1995.

[27] P. C. Hanlon. *Compressor Handbook*. McGraw-Hill, USA, 2001.

[28] V. Hegadekatte, N. Huber, and O. Kraft. Finite element based simulation of dry sliding wear. *Modelling and Simulation in Materials Science and Engineering*, 13(1):57–75, 2005.

[29] V. Hegadekatte, N. Huber, and O. Kraft. Modeling and simulation of wear in a pin on disc tribometer. *Tribology Letters*, 24(1):51–60, 2006.

[30] V. Hegadekatte, S. Kurzenhäuser, N. Huber, and O. Kraft. A predictive modeling scheme for wear in tribometers. *Tribological Contacts and Component Life: Proceedings of the 34th Leeds-Lyon Symposium on Tribology*, 41(11):1020–1031, 2008.

[31] HOERBIGER Compression Technology. Hoerbiger specification, piston rod reconditioning, 2008.

[32] HOERBIGER Compression Technology. Material selection matrix, 2016.

[33] HOERBIGER Compression Technology. Packing ring standard, 29.11.2017.

[34] C. J. Hooke, S. N. Kukureka, P. Liao, M. Rao, and Y. K. Chen. The friction and wear of polymers in non-conformal contacts. *Wear*, 200(1–2):83–94, 1996.

[35] A. Kaufmann and T. Lindner-Silwester. New insight into the wear of packing rings: model, calculation, experiment. *Conference of the European Forum for Reciprocating Compressors*, 11:250–259, 2018.

[36] A. Kaufmann, T. Lindner-Silwester, and T. Antretter. Modelling dry wear of piston rod sealing elements of reciprocating compressors considering gas pressure drop across the dynamic sealing surface. *Journal of Tribology*, 140(4):042201–042201–7, 2018.

[37] J. Khedkar, I. Negulescu, and E. I. Meletis. Sliding wear behavior of PTFE composites. *Wear*, 252(5–6):361–369, 2002.

[38] N. V. Klaas, K. Marcus, and C. Kellock. The tribological behaviour of glass filled polytetrafluoroethylene. *Special Issue in Memory of Professor Tony Ball*, 38(9):824–833, 2005.

[39] S.-Q. Lai, L. Yue, T.-S. Li, and Z.-M. Hu. The friction and wear properties of polytetrafluoroethylene filled with ultrafine diamond. *Wear*, 260(4):462–468, 2006.

[40] J. Lengiewicz and S. Stupkiewicz. Efficient model of evolution of wear in quasi-steady-state sliding contacts. *Wear*, 303(1–2):611–621, 2013.

[41] C. Li and F. Yan. A comparative investigation of the wear behavior of PTFE and PI under dry sliding and simulated sand-dust conditions. *Wear*, 266(7–8):632–638, 2009.

[42] H.L. Li, Z.W. Yin, D. Jiang, L.Y. Jin, and Y.Q. Cui. A study of the tribological behavior of transfer films of PTFE composites formed under different loads, speeds and morphologies of the counterface. *Wear*, 328–329:17–27, 2015.

[43] T. Lindner-Silwester. Advances in fundamental understanding of the dynamic sealing action in packing systems. *Conference of the European Forum for Reciprocating Compressors*, 5:40–50, 2007.

[44] T. Lindner-Silwester and C. Hold. The BCD packing ring - a new high performance design. *Conference of the European Forum for Reciprocating Compressors*, 7:112–119, 2010.

[45] X. Liu, S. Liu, and H. Ji. Numerical research on rock breaking performance of water jet based on sph. *Powder Technology*, 286:181–192, 2015.

[46] Z. P. Lu and K. Friedrich. On sliding friction and wear of PEEK and its composites. *10th International Conference on Wear of Materials*, 181–183, Part 2(0):624–631, 1995.

[47] O. A. Mamaev, Y. K. Mashkov, R. I. Kosarenko, and V. P. Pivovarov. Durability of a polymer composite material and service life of packing elements of contact movable sealing devices. *Journal of Friction and Wear*, 29(2):127–132, 2008.

[48] F. J. Martínez, M. Canales, S. Izquierdo, M. A. Jiménez, and M. A. Martínez. Finite element implementation and validation of wear modelling in sliding polymer–metal contacts. *Wear*, 284–285(0):52–64, 2012.

[49] MATLAB. *version 9.2.0 (R2017a)*. The MathWorks Inc., Natick and Massachusetts and USA, 2017.

[50] H. C. Meng and K. C. Ludema. Wear models and predictive equations: their form and content. *10th International Conference on Wear of Materials*, 181–183, Part 2(0):443–457, 1995.

[51] S. Mukras, N. H. Kim, W. G. Sawyer, D. B. Jackson, and L. W. Bergquist. Numerical integration schemes and parallel computation for wear prediction using finite element method. *Wear*, 266(7–8):822–831, 2009.

[52] I. Páczelt and Z. Mróz. On the analysis of steady-state sliding wear processes. *Tribology International*, 42(2):275–283, 2009.

[53] I. Páczelt and Z. Mróz. Solution of wear problems for monotonic and periodic sliding with p-version of finite element method. *Higher Order Finite Element and Isogeometric Methods*, 249–252:75–103, 2012.

[54] C. Paulin, S. Fouvry, and C. Meunier. Finite element modelling of fretting wear surface evolution: Application to a Ti–6A1–4V contact. *Wear*, 264(1–2):26–36, 2008.

[55] M. Peigney. Simulating wear under cyclic loading by a minimization approach. *International Journal of Solids and Structures*, 41(24–25):6783–6799, 2004.

[56] Y. M. Pleskachevsky and V. A. Smurugov. Thermal fluctuations at PTFE friction and transfer. *Wear*, 209(1–2):123–127, 1997.

[57] C. D. Radcliff. Development of piston rings for reciprocating compressors. In *Compressors and their systems 2001*, pages 201–212, London, 2001. Professional Engineering Publishers.

[58] P. J. Rae and E. N. Brown. The properties of poly(tetrafluoroethylene) (PTFE) in tension. *Controlled Macromolecular Synthesis and Controlled Architectures - Supramolecular Polymer Assemblies*, 46(19):8128–8140, 2005.

[59] P. J. Rae and D. M. Dattelbaum. The properties of poly(tetrafluoroethylene) (PTFE) in compression. *Polymer*, 45(22):7615–7625, 2004.

[60] M. Rao, C. J. Hooke, S. N. Kukureka, P. Liao, and Y. K. Chen. The effect of PTFE on the friction and wear behavior of polymers in rolling-sliding contact. *Polymer Engineering & Science*, 38(12):1946–1958, 1998.

[61] A. Rezaei, W. van Paepegem, P. de Baets, W. Ost, and J. Degrieck. Adaptive finite element simulation of wear evolution in radial sliding bearings. *Wear*, 296(1–2):660–671, 2012.

[62] S. K. Rhee. Wear equation for polymers sliding against metal surfaces. *Wear*, 16(6):431–445, 1970.

[63] T. C. Robinson. Reciprocating compressor cutaway display, 2014. Photo is used with the permission from HOERBIGER.

[64] J. A. Sauer, D. R. Mears, and K. D. Pae. Effects of hydrostatic pressure on the mechanical behaviour of polytetrafluoroethylene and polycarbonate. *European Polymer Journal*, 6(7):1015–1032, 1970.

[65] A. Schelling and H. H. Kausch. The influence of long term reciprocating dry friction on the wear behaviour of short fibre reinforced composite materials. *Tribology of composite materials*, pages 227–238, 1990.

[66] T. L. Schmitz, J. E. Action, D. L. Burris, W. G. Sawyer, and J. C. Ziegert. Measurement uncertainty in tribological wear rate testing. *technical papers-society of manufacturing engineers-all series-*, 2004.

[67] X. Shen, Y. Liu, L. Cao, and X. Chen. Numerical simulation of sliding wear for self-lubricating spherical plain bearings. *Journal of Materials Research and Technology*, 1(1):8–12, 2012.

[68] Y. J. Shi, X. Feng, H. Y. Wang, C. Liu, and X. H. Lu. Effects of filler crystal structure and shape on the tribological properties of PTFE composites. *Tribology International*, 40(7):1195–1203, 2007.

[69] W. Shibo, C. Niu, and B. Teng. Tribological behavior of polytetrafluoroethylene: Effect of sliding motion. *Journal of Tribology*, 139(1):011301–011301–7, 2016.

[70] V. A. Smurugov, A. I. Senatrev, V. G. Savkin, V. V. Biran, and A. I. Sviridyonok. On PTFE transfer and thermoactivation mechanism of wear. *Wear*, 158(1–2):61–69, 1992.

[71] H. Sui, H. Pohl, U. Schomburg, G. Upper, and S. Heine. Wear and friction of PTFE seals. *Wear*, 224(2):175–182, 1999.

[72] K. Tanaka and S. Kawakami. Effect of various fillers on the friction and wear of polytetrafluoroethylene-based composites. *Wear*, 79(2):221–234, 1982.

[73] T. Tevrüz. Tribological behaviours of carbon filled polytetrafluoroethylene (PTFE) dry journal bearings. *Wear*, 221(1):61–68, 1998.

[74] C. Thomas. *Innenkühlung der Kolbenstange von trockenlaufenden Kolbenverdichtern.* TUDpress, Dresden, 2014.

[75] I. Tzanakis, M. Conte, M. Hadfield, and T. A. Stolarski. Experimental and analytical thermal study of PTFE composite sliding against high carbon steel as a function of the surface roughness, sliding velocity and applied load. *Wear*, 303(1–2):154–168, 2013.

[76] Y. Uchiyama and K. Tanaka. Wear laws for polytetrafluoroethylene. *Wear*, 58(2):223–235, 1980.

[77] J. M. Urueña, A. A. Pitenis, K. L. Harris, and W. G. Sawyer. Evolution and wear of fluoropolymer transfer films. *Tribology Letters*, 57(1):9, 2015.

[78] E. Vanswijgenhoven, S. Cutulic, K. Kenis, G. de Regter, O. Crols, and P. Pennings. The effect of pneumatic gas pressure on the wear behaviour of PTFE. *Wear*, 264(7–8):494–498, 2008.

[79] Z. Wang, K. Kou, Z. Liu, D. Zhang, H. Bi, M. Chao, and Q. Zhao. Structural characterization and high-temperature compressive creep of PTFE-based composites filled with inorganic nanoparticles. *Polymers for Advanced Technologies*, 23(3):545–550, 2012.

[80] Wikipedia contributors. Hele-Shaw flow, 2018. [online; accessed 29-September-2018].

[81] Wikipedia contributors. Compressor, 5 July 2018 07:47 UTC. [online; accessed 20-July-2018].

[82] D. Xiang and C. Gu. A study on the friction and wear behavior of PTFE filled with ultra-fine kaolin particulates. *Materials Letters*, 60(5):689–692, 2006.

[83] F. Yan, Q. Xue, and S. Yang. Debris formation process of PTFE and its composites. *Journal of Applied Polymer Science*, 61(7):1223–1229, 1996.

[84] X.-B. Yang, X.-Q. Jin, Z.-M. Du, T.-S. Cui, and S.-K. Yang. Frictional behavior investigation on three types of PTFE composites under oil-free sliding conditions. *Industrial Lubrication and Tribology*, 61(5):254–260, 2009.

[85] B. Yu and A. W. Khair. Numerical modeling of rock ridge breakage in rotary cutting. In *1st Canada - U.S. Rock Mechanics Symposium*, page 8, ARMA, 2007. American Rock Mechanics Association.

[86] Z.-Z. Zhang, Q.-J. Xue, W.-M. Liu, and W.-C. Shen. Friction and wear properties of metal powder filled PTFE composites under oil lubricated conditions. *Wear*, 210(1–2):151–156, 1997.

[87] F. Ziegler. *Mechanics of solids and fluids*. Springer-Verlag, New York, 2nd ed edition, 1995.