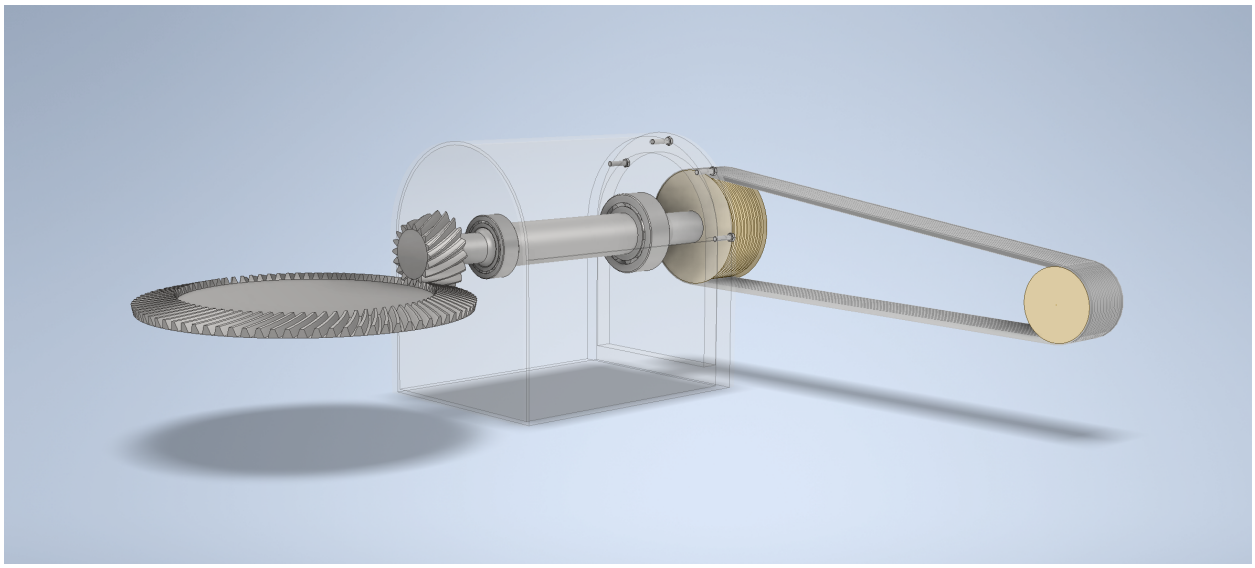
CHALMERS



Modernizing Machine Elements Design

A comparative study between theoretical frameworks and modern CAD software

Bachelor thesis in Mechanical engineering and Automation and Mechatronics

Olof Abrahamsson
Axel Månhammar Abrahamsson
Miika Nurmi
David Wessberg
Albin Wessman
Simon Wikell

Department of Industrial and Materials Science

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Bachelor's thesis 2025

Modernizing Machine Elements Design

A comparative study between theoretical frameworks and modern CAD software

Olof Abrahamsson
Axel Månhammar Abrahamsson
Miika Nurmi
David Wessberg
Albin Wessman
Simon Wikell

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Industrial and Materials Science
Division of Product Development
Chalmers University of Technology
Gothenburg, Sweden 2025

Modernizing Machine Elements Design
A Comparative Study Between Theoretical Frameworks and Modern CAD Software
Olof Abrahamsson
Axel Månhammar Abrahamsson
Miika Nurmi
David Wessberg
Albin Wessman
Simon Wikell

Supervisor: Kanishk Bhadani, Department of Industrial and Materials Science
Examiner: Magnus Evertsson, Department of Industrial and Materials Science

Bachelor's Thesis 2025
Department of Industrial and Materials Science
Division of Product Development
IMSX16-VT25-11
Chalmers University of Technology
SE-412 96 Gothenburg

Cover: Simplified model of a cone crusher drivetrain.

Gothenburg, Sweden 2025

Modernizing Machine Elements Design
A Comparative Study Between Theoretical Frameworks and Modern CAD Software

Olof Abrahamsson
Axel Månhammar Abrahamsson
Miika Nurmi
David Wessberg
Albin Wessman
Simon Wikell

Department of Industrial and Materials Science
Division of Product Development

Abstract

This bachelor thesis investigates the relationship between traditional theoretical frameworks and modern computer-aided design (CAD) software in the context of machine element design. As engineering workflows increasingly adopt automated digital tools, it becomes essential to evaluate whether these technologies uphold the transparency, educational value of manual calculations, and result validations. The study conducts a comparative analysis of selected machine elements using both manual calculations and various CAD platforms, assessing their performance based on precision, usability, and result traceability. A key aspect of the software investigation is the recognition that different CAD platforms are influenced by varying standards and theoretical foundations, often shaped by their regional or institutional origins. Although the industry strive to create common approaches and methods, many differences remain. The findings highlight discrepancies in calculation transparency, underscore the importance of foundational engineering knowledge, and offer guidance for effectively integrating CAD tools into academic use as well as professional practice. By bridging theoretical and digital methodologies, this research contributes to a more informed and balanced approach to modern mechanical design.

Sammandrag

Detta kandidatarbete undersöker sambandet mellan traditionella teoretiska beräkningsmetoder och modern datorstödd design (CAD) vid utformning av maskinelement. I takt med att ingenjörsprocesser i allt högre grad automatiseras och digitaliseras är det avgörande att utvärdera om dessa verktyg bibehåller den noggrannhet, transparens och pedagogiska nytta som manuella beräkningar erbjuder. Studien genomför en jämförande analys av utvalda maskinelement med både handberäkningar och olika CAD-program och utvärderar dessa utifrån precision, användarvänlighet och spårbarhet i resultaten. En central del av programvaruundersökningen är insikten att olika CAD-plattformar påverkas av skilda standarder och teoretiska grunder, vilka ofta formats av deras regionala eller institutionella ur-

sprung. Även om branschen strävar efter att skapa gemensamma arbetssätt och metoder kvarstår många olikheter. Resultaten belyser skillnader i beräkningarnas transparens, betonar vikten av grundläggande ingenjörskunskap och ger vägledning i hur CAD-verktyg effektivt kan integreras i utbildning och yrkesliv. Genom att förena teori med digitala metoder bidrar detta arbete till en mer informerad och balanserad syn på modern maskinkonstruktion.

Keywords: CAD, Machine Elements Design, MITCalc, MDesign, Autodesk Inventor, Bevel Gears, Belt Drives, Shafts, Bearings, Bolted joints, Design Accelerator

Acknowledgements

We would like to express our gratitude to our supervisor, Kanishk Bhadani, for his continuous support, valuable feedback, and insightful guidance throughout the course of this project. His expertise and encouragement were instrumental in shaping our work.

We would also like to thank our examiner, Magnus Evertsson, for his thoughtful comments and constructive critique that helped refine the quality of our thesis.

Gothenburg, May 2025
Olof Abrahamsson, Axel Månhammar Abrahamsson, Miika Nurmi, David Wessberg, Albin Wessman, Simon Wikell

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AI	Artificial Intelligence
ANSYS	Analysis System (commercial FEA software)
CAD	Computer-Aided Design
DIN	Deutsches Institut für Normung (German standards organization)
FEA	Finite Element Analysis
FEM	Finite Element Method
ISO	International Organization for Standardization
ML	Machine Learning
RMA	Rubber Manufacturers Association
SKF	Svenska Kullagerfabriken
SMS	Svensk Maskinstandard (Swedish Mechanical Standards)
SS	Svensk Standard

Nomenclature

Below is the nomenclature of variables that have been used in the report, with the used unit in brackets. The list is divided into separate lists for each machine element analysed in this project.

Shafts

F_r	Radial force acting perpendicular to the shaft's axis [N]
F_g	Gravitational force on the shaft [N]
F_a, F_{a2}	Axial force acting along the shaft's axis [N]
F_b	Radial force from the bearing on the right [N]
R_1, R_2	Reaction forces from the first and second bearing [N]
L_1–L_5	Shaft segment lengths [m]
D_1–D_5	Shaft segment diameters [m]
z_{cg}	Position of the center of gravity of the shaft [m]
$M(z)$	Bending moment as a function of position z [Nm]
$V(z)$	Shear force as a function of position z [N]
T	Torsional moment [Nm]
$A(z)$	Cross-sectional area at position z [m^2]
$I(z)$	Second moment of area at position z [m^4]
$J(z)$	Polar moment of inertia at position z [m^4]
$r(z)$	Radius of the shaft at position z [m]

$\sigma_b(z)$	Bending stress at position z [Pa]
$\tau_{\text{shear}}(z)$	Shear stress from transverse force at z [Pa]
$\tau_{\text{torsion}}(z)$	Shear stress from torsional load at z [Pa]
$\sigma_{\text{axial}}(z)$	Axial stress at position z [Pa]

σ_{vm}	von Mises equivalent stress [Pa]
n	Safety factor [–]
ρ	Material density [kg/m^3]
m	Total shaft mass [kg]
$\varphi(z)$	Angular deflection due to torque at z [rad]

Belt drives

i,j	Indices for distribution network buses [-]
F_0	Required preload force [N]
F_1	Tension force on the tight side of the belt [N]
F_2	Tension force on the slack side of the belt [N]
F_c	Centrifugal force acting on the belt [N]
F_{e1}	Effective tension force on tight side [N]
F_{e2}	Effective tension forces on slack sides [N]
$\mathrm{F}_{\mathrm{outer}}$	Axially applied external force [N]
L	Belt length [m]
M_1	Torque on the driving pulley [Nm]
M_2	Torque on the driven pulley [Nm]
R_1	Radius of the driving pulley [m]
R_2	Radius of the driven pulley [m]
Z	Required number of belts [-]

a	Axis distance between pulleys [m]
f_c	Correction factor for belt length [-]
f_d	Correction factor for wrap angle [-]
i	Speed ratio between the pulleys [-]
m'	Belt mass per unit length [kg/m]
p_d	Operating power [W]
p_r	Transmittable power per belt [W]
$\mathrm{v}_{\mathrm{belt}}$	Belt velocity [m/s]

α_1	Wrap angle on the driving pulley [rad]
α_2	Wrap angle on the driven pulley [rad]
β	Groove angle of the pulley [°]
χ	Load ratio between tight and slack side forces [-]
χ_{slip}	Load ratio considering belt slip [-]
$\Delta\alpha$	Deviation angle due to pulley offset [rad]
λ	Load factor (used in preload equations) [-]
μ	Friction coefficient (flat belt) [-]
μ_s	Apparent friction coefficient (V-belt) [-]
ω_1	Angular velocity of the driving pulley [rad/s]
ω_2	Angular velocity of the driven pulley [rad/s]
θ	Degree of utilization [-]

Bearings

C	Basic dynamic load rating [N]
F_a	Axial load [N]
F_r	Radial load [N]
L_{10}	Basic rating life [millions of revolutions]
L_{10h}	Basic rating life [hours]
L_{nm}	SKF-modified bearing life [millions of revolutions]
L_{nmh}	SKF-modified bearing life [hours]
P	Equivalent dynamic bearing load [N]
P_u	Fatigue load limit [N]
R	Reliability level (probability) [-]
X	Radial load factor [-]
Y_1	Axial load factor for light load [-]
Y_2	Axial load factor for heavy load [-]
a_1	Life adjustment factor for desired reliability level [-]
a_{SKF}	SKF life modification factor accounting for lubrication, contamination, and fatigue properties [-]
n	Rotational speed [rpm]
p	Life exponent (3 for ball bearings, 10/3 for roller bearings) [-]
η_c	Contamination factor [-]

κ	Lubrication condition of the bearing, i.e. viscosity ratio [-]
ν	Actual operating viscosity of the oil or the grease base oil [mm^2/s]
ν_1	Rated viscosity, function of the mean bearing diameter and rotational speed [mm^2/s]

Bolted Joints

A_{eq}	Active cross-sectional area of the clamped part that carries the load [mm^2]
c_k	Stiffness of the clamped part [N/mm]
c_s	Stiffness of the bolt [N/mm]
d_2	Pitch diameter of the thread [mm]
d_h	Inner diameter (hole diameter) in the clamped part [mm]
d_w	Outer diameter (washer or contact diameter) in the clamped part [mm]
E_k	Young's modulus of the clamped part [MPa]
E_s	Young's modulus of the screw material [MPa]
F_0	Preload force in the screw [N]
$F_{0,\text{ settling}}$	Total preload force including settlement compensation [N]
F_k	Force in the clamped parts [N]
F_N	External axial force acting on the joint [N]
F_s	Force absorbed by the bolt [N]
L_k	Length of the clamped parts [mm]
$M_{\text{tightening, settling}}$	Tightening moment including settlement compensation [Nm]
P	Thread pitch [mm]
r_m	Effective friction radius in the thread contact [mm]
x	Geometrical factor depending on clamping length and washer diameter [-]
δ_0	Total deformation length due to settlement [mm]
δ_k	Deformation in the clamped parts [mm]
δ_s	Deformation in the screw [mm]
μ_{thread}	Friction coefficient in the threads [-]

Bevel gears

Pinion gear is notated with subscript 1 and wheel gear is notated with subscript 2.

Geometry and force distribution

b_1, b_2	Face width [mm]
b_{e1}, be_{e2}	Face width from calculation point to outside [mm]
b_{i1}, b_{i2}	Face width from calculation point to inside [mm]
c	Clearance [mm]
c_{be2}	Face width factor [–]
c_{ham}	Mean addendum factor of wheel [–]
d_{ae1}, d_{ae2}	Outside diameter [mm]
d_{e1}, d_{e2}	Outer pitch diameter [mm]
d_{m1}, d_{m2}	Mean pitch diameter [mm]
F_{ax}	Axial force [N]
F_{mt1}, F_{mt2}	Tangential force at mean diameter [N]
F_{rad}	Radial force [N]
$f_{\alpha\mathrm{lim}}$	Influence factor of limit pressure angle [–]
h_{ae1}, h_{ae2}	Outer addendum [mm]
h_{am1}, h_{am2}	Mean addendum [mm]
h_{amc1}, h_{amc2}	Mean chordal addendum [mm]
h_{e1}, h_{e2}	Outer whole depth [mm]
h_{fe1}, h_{fe2}	Outer dedendum [mm]
h_{fi1}, h_{fi2}	Inner dedendum [mm]
h_{fm1}, h_{fm2}	Mean dedendum [mm]
h_m	Mean whole depth [mm]
h_{mw}	Mean working depth [mm]
h_{t1}	Pinion whole depth [mm]
j_{en}	Outer normal backlash [mm]
j_{et}	Outer transverse backlash [mm]
j_{mn}	Mean normal backlash [mm]
j_{mt}	Mean transverse backlash [mm]
k_c	Clearance factor [–]
k_d	Depth factor [–]

k_{hap}	Addendum coefficient [–]
k_t	Circular thickness factor [–]
m_{et}	Outer transverse module [mm]
m_{mn}	Mean normal module [mm]
n_1	Pinion speed [min^{-1}]
P	Power [W]
R_{e1}, R_{e2}	Outer cone distance [mm]
R_{i1}, R_{i2}	Inner cone distance [mm]
R_{m1}, R_{m2}	Mean cone distance [mm]
r_{c0}	Cutter radius [mm]
s_{mn1}, s_{mn2}	Mean normal circular tooth thickness [mm]
s_{mnc1}, s_{mnc2}	Mean normal chordal tooth thickness [mm]
t_{xo1}, txo_{xo2}	Crown to crossing point (hypoid) [mm]
x_{sm1}, x_{sm2}	Thickness modification coefficient (with backlash) [–]
T_1	Pinion torque [Nm]
t_{xi1}, t_{xi2}	Front crown to crossing point [mm]
t_{z1}, tz_{z2}	Pitch apex beyond crossing point [mm]
t_{zF1}, t_{zF2}	Face apex beyond crossing point [mm]
t_{zi1}, t_{zi2}	Crossing point to inside point along axis [mm]
t_{zm1}, t_{zm2}	Crossing point to mean point along axis [mm]
t_{zR1}, t_{zR2}	Root apex beyond crossing point [mm]
u	Gear ratio [–]
u_a	Equivalent gear ratio [–]
W_{m2}	Wheel mean slot width [mm]
x_{hm1}	Profile shift coefficient [–]
x_{smn}	Thickness modification coefficient (theoretical) [–]
z_0	Number of blade groups [–]
z_1, z_2	Number of teeth [–]
z_p	Number of crown gear teeth [–]
α_{dC}	Nominal design pressure angle on coast side [°]
α_{dD}	Nominal design pressure angle on drive side [°]
α_{eC}	Effective pressure angle on coast side [°]
α_{eD}	Effective pressure angle on drive side [°]
α_{nD}	Generated pressure angle on drive side [°]

α_{nC}	Generated pressure angle on coast side [°]
α_{lim}	Limit pressure angle [°]
β_{e1}, β_{e2}	Outer spiral angle [°]
β_{i1}, β_{i2}	Inner spiral angle [°]
β_{m1}, β_{m2}	Mean spiral angle [°]
δ_{a1}, δ_{a2}	Face angle [°]
δ_{f1}, δ_{f2}	Root angle [°]
δ_1, δ_2	Pitch angle [°]
θ_{a1}, θ_{a2}	Addendum angle [°]
θ_{f1}, θ_{f2}	Dedendum angle [°]
Σ	Shaft angle [°]
ε_β	Face contact ratio [–]
δ_{a1}, δ_{a2}	Face angle [°]
δ_{f1}, δ_{f2}	Root angle [°]
δ_1, δ_2	Pitch angle [°]
θ_{a1}, θ_{a2}	Addendum angle [°]
θ_{f1}, θ_{f2}	Dedendum angle [°]

Safety calculation

E_{modul}	Young's modulus [MPa]
F_{mt1}	Tangential load on pinion [N]
F_{ber}	Equivalent load for safety calculations [N]
K_1	Load factor [–]
K_{v1}	Dynamic factor [–]
$K_{H\alpha}$, $K_{H\beta}$	Load distribution factors for contact stress [–]
$K_{F\alpha}$, $K_{F\beta}$	Load distribution factors for bending stress [–]
K_L	Lubrication factor [–]
K_{HX}	Size factor for contact stress [–]
K_{HN}	Life factor for contact stress [–]
K_{HK}	Hardness combination factor [–]
K_{FX}	Size factor for bending stress [–]
K_{FN}	Life factor for bending stress [–]
m_{mn}	Normal module [mm]

$n_{1,\mathrm{rad}}$	Pinion rotational speed [rad/min]
R_e, R_i	Outer/inner pitch cone distances [mm]
S_H	Safety factor for pitting [–]
S_F	Safety factor for bending [–]
u	Gear ratio [–]
v_{mt1}	Mean tangential velocity [m/s]
x_{hm1}	Profile shift coefficient [–]
x_{vn1}	Virtual addendum modification [–]
Y_F	Tooth form factor [–]
Y_K	Tooth length factor [–]
Y_β	Helix angle factor [–]
Y_ε	Contact ratio factor (bending) [–]
Y_s	Stress correction factor [–]
Z_H	Geometry factor (Hertzian contact) [–]
Z_K	Cone length factor [–]
Z_M	Material factor [–]
Z_ε	Contact ratio factor (Hertzian) [–]
Z_R	Surface roughness factor [–]
Z_V	Lubrication film factor [–]
z_1, z_2	Number of teeth (pinion, gear) [–]
z_{v1}, z_{v2}	Virtual number of teeth [–]
z_{vn1}, z_{vn2}	Virtual normal number of teeth [–]
α_n	Normal pressure angle [°]
α_t	Effective transverse pressure angle [°]
β_m	Mean spiral angle [°]
β_{bm}	Virtual base helix angle [°]
σ_H	Contact (Hertzian) stress [MPa]
σ_{HP}	Allowable contact stress [MPa]
$\sigma_{H\mathrm{lim}}$	Contact fatigue limit [MPa]
σ_F	Bending stress [MPa]
σ_{FP}	Allowable bending stress [MPa]
σ_{Flim}	Bending fatigue limit [MPa]

Contents

List of Figures

List of Tables

1
Introduction

1.1 Background

Machine elements are universal in various fields, and essential in numerous applications, including machines, plants, and apparatus, forming the building blocks of mechanical systems (Engelmann et al., 2021). In the field of machine element design, there has been advancement within digital tools, particularly *Computer-Aided Design*, which will henceforth be referred to as CAD. Calculations that previously were time consuming can now be executed within seconds. This advancement has made it possible for engineers, both in academic and professional settings, to perform designs and analyses at a level of efficiency which is nearly impossible to execute with hand calculations alone. CAD-based calculations have become so simple that they do not require much engineering knowledge to operate, which in turn affect the stages of learning in an educational setting. (Ahmed & Rahman, 2016).

We are currently transitioning from traditional reliance on computers and conventional software tools to the integration of artificial intelligence (AI) and machine learning (ML) technologies. This shift make it possible to reduce the time required for product development, due to that calculations, design and testing can be done autonomously and iteratively. The transition enhances process efficiency and makes companies that adopt these technologies more competitive (Cooper & Brem, 2024). However there is a growing tendency to uncritically trust technology and forget its potential flaws, which can result in increased errors, inaccuracies, and a distorted comprehension of reality (Vuori et al., 2019). If this is the case with machine element design, there is a possibility that we could face problems. The selection of machine elements is crucial for the overall quality and durability of a system, and it is essential to ensure that the calculations are correct.

1.2 Objectives

This study aims to analyse the differences and similarities between theoretical calculations and CAD-based methods, as well as among the CAD-based methods themselves, in the context of machine element design. It focuses on bearings, shafts, bolted joints, bevel gears, and belt drives, evaluating the usability, transparency, and result verification of the CAD tools.

By establishing a comparative framework, this research will offer valuable insight into leveraging CAD software without compromising the rigour of traditional calculation techniques. The findings will support educational institutions and industries in improving training resources for the design and analysis of machine elements. The objectives listed:

- Compare outputs from software and hand calculations, regarding machine elements.
- Evaluate CAD tools in terms of:
  - Usability
  - Transparency
  - Result verification
  - Usage of standards
- Usage of CAD tools in an educational setting

1.3 Limitations

The study has certain limitations that have to be taken into account. These are important to be acknowledged and to have a complete understanding of the study and the results and their limitations. Limitations in this study include number of elements studied, and number of CAD software used for analysis.

- **Limited Number of Analysed Machine Elements**
  Due to time constraints, only a select number of machine elements were analysed. Choices must be made regarding which elements to include and the level of detail in their analysis. This limitation may impact the comprehensiveness of the study's findings.
- **Limited theoretical reference framework**
  The study is focused on the analysis of software, therefore a limited theoretical framework will be used. There is a wide variety of different standards to compare the results to, but for this study recognised European standards were chosen as the benchmark.
- **Restricted CAD Software Selection**
  The study was limited by the inability to use all relevant CAD software due to time constraints. Some software programs with potentially valuable tools and properties may be excluded, which influences the results and limit the study's ability to compare different analytical approaches.

By recognizing these limitations, the study ensures a more transparent approach and allows for informed interpretation of the results within the given constraints.

2
Theory

2.1 Machine Elements

In this field of study, a machine is referred to as a mechanical device that uses power to perform one or several specified functions, where its detachable components are machine elements (Mägi et al., 2017). Machine elements serve as the fundamental building blocks necessary for constructing a functional machine and are generally categorized as universal elements. They are systematically designed and arranged to operate in unison. Examples of machine elements include, bolts, bearings, keys, belts, gears and others. (Jiang, 2019). They have been employed by humans during thousands of years, for example wheels and grease-lubricated plain bearings. The modern, widely accepted definition of machine elements is largely influenced by German engineering traditions (Mägi et al., 2017).

2.2 Machine Elements Design

Design is a widely and common concept. In broad terms does it aims to create and execute a purposeful plan to meet industrial, commercial and social demands. In a mechanical engineering domain, the design process is innovative, problem-solving and iterative process that requires scientific knowledge as well as creative capability (Jiang, 2019). In order to determine what machine element design encompass and the importance of this subject we need to look into machine design.

Machine design is intended to create, develop or improve a new or already existing mechanical device. It's a result of an endeavour to meet the needs from a system or humans. The importance of machine design is firstly based on meeting the function of a system, strive to conserve resources, lower the environmental impact and make systems more durable. The machine design process is an extensive procedure that requires interactive and iterative processes as well as complex calculations. Figure 2.1, (Jiang, 2019, p. 8, figure 1.1), clarify the complex workflow that a design process requires, this includes the *Planning* and *Manufacturing and commercialization* part aswell.

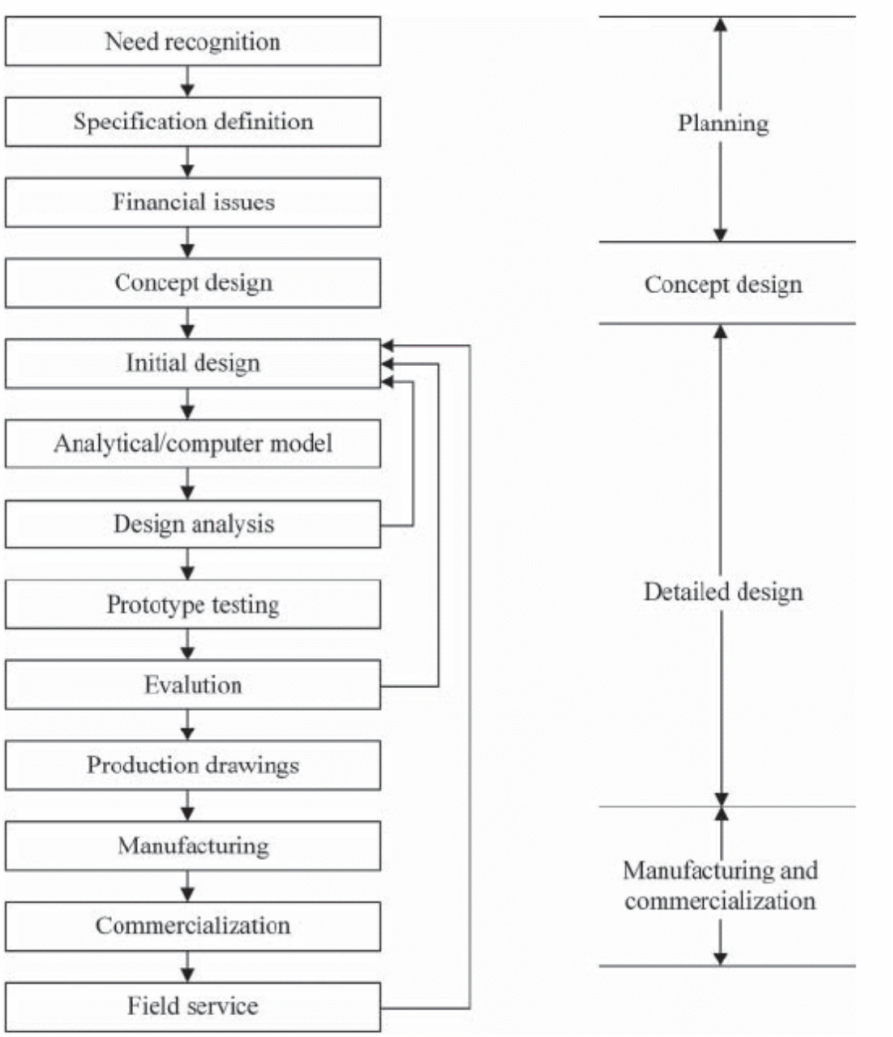


Figure 2.1: General flowchart of a product development process. (Jiang, 2019, p. 8, figure 1.1)

Machine Elements Design is the design of the components of a machine. Since machine elements are fundamental components of a machine, the design process is broad in scope. Each individual element must be properly designed to fulfill its function within the overall system. The most common criteria in the design of machine elements—and the primary reasons for undertaking such a process—are safety, durability, and efficiency.

2.3 Computer-aided design

Computer-aided design (CAD) is a computer-based tool for design modelling, design analysis and design review. Small and medium companies started to use CAD in the 1980s, although the development for this type of tool began already in the 1960s by the aeronautical and automotive industries. In the beginning was CAD limited to 2D-drawing, but nowadays 3D modeling is widespread due to the versatility of the software. In any kind of product development, it is an essential tool with relevant benefits such as cost and time optimization. It is used in almost every engineering application. (Lantada & Morgado, 2013)

CAD is useful across a wide range of industries. CAD allows for accurate design and modelling and reduces human error which ensures that measurements and geometries are exact compared to manual drafting, which is essential for engineering and manufacturing.

Designers can easily make changes without starting from scratch. This speeds up the design process and supports iterative development. It is easier to visualize concepts with 2D and 3D models, which also helps to identify problems earlier. For a seamless transition from design to production CAD files can be directly used in different manufacturing processes like 3D printing and CNC machining.

Generative design is a combination of parametric modeling and optimization to automatically create designs that meet specific goals. Historically, the use of this was limited because engineers needed to write their own software to use optimization. Plug-ins such as Grasshoppers Galapagos made optimization easier by using built-in algorithms such as Genetic Algorithms. This made it possible to explore and improve designs without needing to code. Generative design helps create lighter, stronger, and more efficient parts. While it still requires a lot of computing power, generative design is a promising tool for improving design performance and efficiency (Coenders, 2021).

In the context of machine element design, CAD software plays a critical role in supporting the selection and sizing of standard components such as bearings, bolts, gears, and shafts. While these elements are often predefined in standard libraries, their dimensions and tolerances depend on engineering calculations, including load, fit, and safety factors. Modern CAD tools assist in this by integrating simulation and analysis features that align with mechanical design requirements. (Metwalli, 2021)

2.4 Standards

A standard is a common solution of a frequent problem. It is a way to set consistent and transparent routines. One of the main reasons with standards is to increase the quality, efficiency and create a common approach so that the risk of misinterpretation decreases. Thanks to standards, companies, productions etc. do not need to come up with new things each time they change a product or similar (för Standarder (SIS), 2025).

Standards are beneficial by several reasons, firstly can it act like a common language between different parts, both when it comes to production and different stakeholders but also when it comes to procurements. Furthermore can standards ensure compatibility and efficiency.

For machine elements, the key properties are primarily defined by applicable standards. These standards typically cover dimensions, calculation methods, safety factors, and other relevant design criteria, (Engelmann et al., 2021).

Standardization can occur on different levels from several origins:

- International
  - ISO - **I**nternational **O**rganization for **S**tandardization
- within the European Union
  - EN - **E**uropäische **N**orm
    (eng: European Norm)
- National
  - DIN - **D**eutsches **I**nstitut für **N**ormung
    (eng: German Institute for Standardization)
  - SS - **S**vensk **S**tandard
    (eng: Swedish Standard)
  - ANSI - **A**merican **N**ational **S**tandards **I**nstitute
- National specific standards
  - SMS - **S**veriges **M**ekanikförbunds **S**tandardcenter
    (eng: Swedish Mechanical Association's Standards Center)
- Group standards
  - RMA - **R**ubber **M**anufacturing **A**ssociation
  - SKF - **S**venska **K**ullager **F**abriken
    (eng: Swedish Ball Bearing Factory)
  - VOLVO

Understanding the origin and scope of these standards is essential when working with CAD tools, as they often incorporate or rely on such standards for accurate modelling and analysis of machine elements.

2.5 Specified Machine elements

In this section, the machine elements included in the study are presented individually. For each element, a brief description of its function and typical applications is provided, followed by the necessary formulas and parameters to illustrate how the element is analysed within the context of the current application. The machine elements of choice are those included in the powertrain of a cone crusher i.e., a shaft, bearings, bolts, a belt drive and a bevel gear. (Rock Machinery, 2025)

2.5.1 Shafts

Figure 2.2: Shaft, (AI-generated image, ChatGPT – OpenAI, 2025).

Shafts are one of the most commonly used elements in machinery. Shafts often experience a combination of axial, bending, and torsional loads. Stress concentration factors for these loading conditions are usually obtained from standard tables and charts (Pedersen, 2018). In a mechanical system, it serves as a rotating component that carries gears, pulleys or couplings.

There are a few different kinds of shafts, e.g. transmission shafts, machine shafts and axles. A transmission shaft is used to transmit power between the source and the machine. A machine shaft is an integral part of the machine itself e.g. a camshaft. An axle or axle shaft is typically used to support rotating elements without transmitting torque (Sabhadiya, 2024).

When designing a shaft, it is essential to consider the loads and stresses it will experience which influence the required dimensions. A circular cross-section is most commonly used because it distributes stresses evenly and provide high strengths (see Figure 2.2). The choice of material is also critical, as it affects strength, toughness and fatigue life. Materials must be selected to withstand both static and dynamic loading conditions. In this work, the theoretical calculations were carried out using the standard assumptions and formulas provided in the KTH handbook (Alfredsson, 2014), which offers guidance for shaft dimensioning under combined loading conditions.

Note: The following equations are derived from DIN 743 (Deutsches Institut für Normung, 2012), which provides standardized formulas for the design and analysis of machine elements. These equations ensure consistency with established engineering practices and are applied in the context of shaft design and other related elements.

2.5.1.1 Calculations

The shaft configuration used for the calculations is shown in Figure 2.3. This geometry serves as the basis for determining the loading conditions, reaction forces, and stress distributions along the shaft.

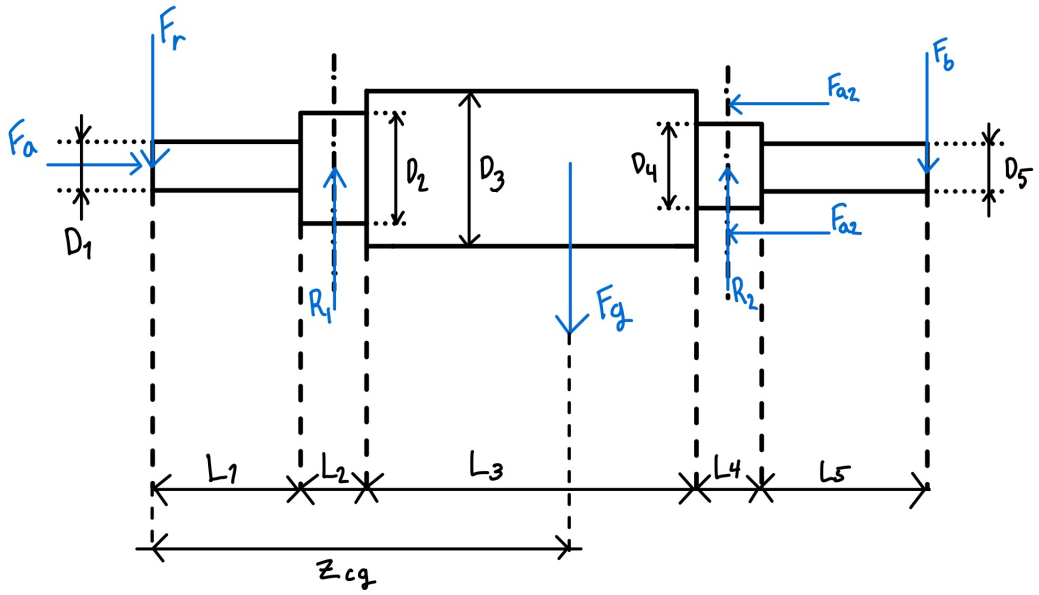


Figure 2.3: Geometry of the shaft with lengths, diameters and loads acting on it.

- F_r: Radial force, acting perpendicular to the shaft's axis.
- F_g: Gravitational force on the shaft.
- F_a: Applied axial force acting along the shaft's axis.
- F_{a2}: Reaction axial force balancing F_a, ensuring axial equilibrium.
- F_b: Radial force from the belt drive on the right.
- R_1 & R_2: Reaction forces from the first and second bearing. R_2 is fixed.
- L_1–L_5: The different lengths of each section of the shaft.
- D_1–D_5: The different diameters of each section of the shaft.
- z_{cg}: Position of the center of gravity of the shaft (from the left end), where the gravitational force $F_g = mg$ acts downward.

Force equilibrium equation

The force equilibrium equation ensures that the sum of all vertical forces acting on the system is zero. In this case, the total vertical reaction forces R_1 and R_2 are balanced by the applied forces: the radial force F_{r}, the gravitational force F_{g} and the radial force F_{b} . This is shown in Figure 2.3. By setting this equation to zero, the system's static equilibrium is maintained.

$$\uparrow y : R_1 + R_2 = F_r + F_g + F_b \tag{2.1}$$

Moment balance around R_1

The moment equilibrium around support R_1 considers all external forces acting on the shaft. The cutting force F_r creates a counterclockwise moment, while the support reaction R_2 and the tool force F_b produce clockwise moments. By setting the sum of these moments to zero, the unknown reaction R_2 can be determined.

$$\overset{\curvearrowleft}{R_1} : -F_r \left(L_1 + \frac{L_2}{2}\right) + R_2 \left(\frac{L_2}{2} + L_3 + \frac{L_4}{2}\right) - F_b \left(\frac{L_2}{2} + L_3 + L_4 + L_5\right) = 0 \tag{2.2}$$

Reaction Forces

Reaction forces calculate the reactions on R_1 and R_2, which act at the shaft supports. By applying static equilibrium, both force and moment balances determine these reactions based on the applied loads from cutting forces, tool forces, and the shaft's own weight. The shaft weight is modeled as a concentrated force F_g, applied at the center of gravity z_{cg}, see Figure 2.2, which represents the mass centroid of the shaft geometry. This accounts for the non-uniform distribution of shaft diameter and length across its sections.

$$R_2 = \frac{F_r \left(L_1 + \frac{L_2}{2}\right) + F_b \left(\frac{L_2}{2} + L_3 + L_4 + L_5\right) - F_g \cdot z_{\mathrm{cg}}}{\frac{L_2}{2} + L_3 + \frac{L_4}{2}} \tag{2.3}$$

$$R_1 = F_r + F_b + F_g - R_2 \tag{2.4}$$

Shear Force Distribution

The shear force distribution describes how transverse forces vary along the shaft's length. These functions are piecewise defined based on the positions of the external loads and reaction forces. This is a key step in performing section analysis and determining internal force behavior.

$$V_1(z) = F_r \tag{2.5}$$
$$V_2(z) = F_r - R_1 \tag{2.6}$$
$$V_3(z) = F_r - R_1 + F_g \tag{2.7}$$
$$V_4(z) = F_r - R_1 + F_g - R_2 \tag{2.8}$$

The application limits for the shear force functions are:

- $V_1(z)$ is valid for $0 \leq z < L_1$.
- $V_2(z)$ is valid for $L_1 \leq z < L_1 + \frac{L_2}{2}$.
- $V_3(z)$ is valid for $L_1 + \frac{L_2}{2} \leq z < z_{cg}$.
- $V_4(z)$ is valid for $z_{cg} \leq z \leq L$,

where z_{cg} is the position of the center of gravity, and L is the total shaft length.

Bending Moment Distribution

This section defines how the internal bending moment varies along the shaft. The bending moments are calculated using the principle of sections and depend on the location of the loads and supports. These moment equations are necessary for determining shaft deflection and stress levels.

$$M_1(z) = F_a \cdot \frac{d}{2} - F_r \cdot z \tag{2.9}$$
$$M_2(z) = M_1(z) + R_1 \cdot (z - (L1 + L2/2)) \tag{2.10}$$
$$M_3(z) = M_2(z) - F_g \cdot (z - z_{cg}) \tag{2.11}$$
$$M_4(z) = M_3(z) + R_2 \cdot (z - L1 - L2 - L3 - L4/2) \tag{2.12}$$

The application limits for the bending moment functions are:

- $M_1(z)$ is valid for $0 \leq z < L_1$.
- $M_2(z)$ is valid for $L_1 \leq z < L_1 + \frac{L_2}{2}$.
- $M_3(z)$ is valid for $L_1 + \frac{L_2}{2} \leq z < z_{cg}$.
- $M_4(z)$ is valid for $z_{cg} \leq z \leq L$,

where z_{cg} is the position of the center of gravity of the shaft, and L is the total shaft length.

Stress Evaluation Along the Shaft

To analyse the structural integrity of the shaft, the stress formulas below are applied point-wise along the length of the shaft. The shaft is divided into five sections, each with known diameter $d(z)$ and length. At each evaluation point z, the corresponding diameter is used to compute cross-sectional properties.
The calculated stresses are:

- $\sigma_b(z)$ — bending stress from internal moment $M(z)$
- $\tau_{\text{shear}}(z)$ — shear stress from transverse force $V(z)$
- $\tau_{\text{torsion}}(z)$ — shear stress from torsional load T
- $\sigma_{\text{axial}}(z)$ — axial stress from axial force F_{axial}

The following geometric property functions are used at each z:

$$A(z) = \frac{\pi}{4}d^2(z) \qquad \text{Cross-sectional area function}$$
$$I(z) = \frac{\pi}{64}d^4(z) \qquad \text{Second moment of area function}$$
$$J(z) = \frac{\pi}{32}d^4(z) \qquad \text{Polar moment of inertia function}$$
$$r(z) = \frac{d(z)}{2} \qquad \text{Radius function}$$

The shaft is modeled in code using a numerical loop over z. Depending on the segment (based on length), the local diameter $d(z)$ is selected. The bending moment $M(z)$ and transverse force $V(z)$ come from static analysis.

The axial stress is only evaluated for $z \leq L_{R2}$, where L_{R2} is the length to R_2 i.e. $L_1 + L_2 + L_3 + \frac{L_4}{2}$ (See fig. 2.3). Since the axial force is only applied over this portion of the shaft:

$$\sigma_{\text{axial}}(z) = \begin{cases} \frac{F_{\text{axial}}}{A(z)}, & \text{if } z \leq L_{R2} \\ 0, & \text{otherwise} \end{cases}$$

Axial Stress (Tension/Compression)
The axial stress is calculated based on the normal force acting along the shaft's axis. It is important for checking if the component can withstand pulling or compressing loads.

$$\sigma_{\text{axial}}(z) = \frac{F_{\text{axial}}}{A(z)} = \frac{F_{\text{axial}}}{\frac{\pi}{4}d^2(z)} \tag{2.13}$$

Torsional Stress
This expression estimates the shear stress due to torque. It's critical in shafts that transmit rotational power.

$$\tau_{\text{torsion}}(z) = \frac{T \cdot r(z)}{J(z)} = \frac{T \cdot \frac{d(z)}{2}}{\frac{\pi}{32}d^4(z)} = \frac{16T}{\pi d^3(z)} \tag{2.14}$$

Shear Stress from Transverse Force (YZ-plane)
This shear stress comes from a transverse force, often approximated for solid circular shafts using a factor $\frac{4}{3}$.

$$\tau_{\text{shear}}(z) = \frac{V(z)}{A(z)} = \frac{V(z)}{\frac{\pi}{4}d^2(z)} = \frac{4V(z)}{\pi d^2(z)} \tag{2.15}$$

Bending Stress (YZ-plane)
Bending stress arises from internal bending moments and varies with the distance from the neutral axis.

$$\sigma_b(z) = \frac{M(z) \cdot r(z)}{I(z)} = \frac{M(z) \cdot \frac{d(z)}{2}}{\frac{\pi}{64} d^4(z)} = \frac{32M(z)}{\pi d^3(z)} \tag{2.16}$$

Angular deflection from torsional load

The angular deflection is calculated pointwise along the shaft to determine how much it twists under the applied torque. The deflection at a given point z depends on the torque T, the shear modulus G, and the polar moment of inertia $J(z)$, which is already defined in the cross-sectional properties. The angular rotation is calculated using the following expression:

$$\varphi(z) = \frac{T \cdot L(z)}{G \cdot J(z)} \tag{2.17}$$

The shaft is divided into multiple segments with varying diameters. For each segment, the expression above is evaluated based on its length and diameter. The total angular deflection is then obtained by summing the contributions from all segments along the shaft.

Von Mises stress equation

The von Mises stress equation is used to determine whether a material subjected to a complex loading condition will yield. It combines the effects of bending, axial, and torsional stresses into one equivalent stress value, denoted as σ_{vm}. In this case, the equivalent stress is calculated from the bending stress σ_b, the axial stress σ_{axial}, and the torsional shear stress $\tau_{torsion}$. This relationship is shown in Equation 2.18. If the von Mises stress exceeds the yield strength of the material, plastic deformation is expected to occur.

$$\sigma_{vm} = \sqrt{\sigma_b^2 + \sigma_{axial}^2 + 3 \cdot \tau_{torsion}^2} \tag{2.18}$$

Safety factor equation

The safety factor is used to evaluate how much stronger a component is compared to the load it is subjected to. It is defined as the relationship between the allowable (or yield) stress of the material and the resulting stress in the component. In this case, the von Mises stress σ_{vm} is used as the stress measure. This relationship is shown in Equation 2.19. A safety factor greater than 1 indicates that the component can withstand the load without yielding or failing.

$$n = \frac{\sigma_{allow}}{\sigma_{vm}} \tag{2.19}$$

Mass of the Shaft

The total mass is calculated by summing the individual masses of each cylindrical section, using the following expression:

$$m = \rho \cdot \sum_{i=1}^{n} A_i \cdot L_i = \rho \cdot \sum_{i=1}^{n} \left(\frac{\pi d_i^2}{4} \cdot L_i \right) \tag{2.20}$$

where:

m	total mass [kg]
ρ	material density [kg/m^3]
A_i	cross-sectional area of section i
d_i	diameter of section i
L_i	length of section i
n	number of cylindrical sections

This approach provides a good estimate of the shaft's total mass, assuming constant density and ideal geometry. Deviations from the real mass could occur due to features like fillets, keyways, or hollow regions, which were not considered in this simplified model. Note: This equation is only applicable for solid shafts. For shafts with internal features such as holes or keyways, the mass calculation would need to account for these voids, which is not included in this simplified model.

2.5.2 Belt Drives

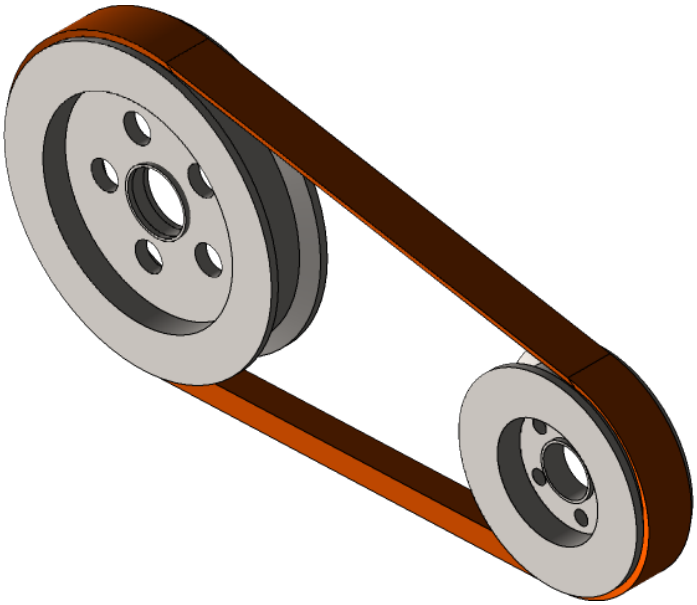


Figure 2.4: V-Belt with two different sized pulleys and one belt (Khanal, 2023)

A belt drive system is a mechanical system that is used to transmit power between rotating shafts, see figure 2.4. It does this using a flexible belt that goes over pulleys attached to the shafts. One of the shafts is usually powered by a motor and rotates the pulley, moving the belt, which in turn rotates the second shaft. Power is transmitted by frictional force between the belt and the pulleys, although there are some types of belt drive systems that use teeth in the belt and pulleys to transmit the power, hence avoiding slippage between the belt and pulley.

The belt is made of rubber and contains cords made of steel or polyester strands embedded in the rubber. The belt gets its axial stiffness from the steel or polyester cords, and friction is generated from the rubber which then transmits the power between the pulleys. (Wasfy & Wasfy, 2019)

There are many different types of belts for a belt drive system, e.g. round belt, V-belt, flat belt and timing/toothed belt. Round belts, or circular belts, has a circular cross section. They are almost only used for low-torque applications and designed to run with V-groove pulleys. V-belts are most commonly used to transmit a large amount of power between two pulleys. They have become the standard belt for power transmission since the V-shape of the belt helps with the alignment and reduces the slippage. Flat belts are also used to transmit power between two pulleys through friction. They are usually classified as small woven endless belts or high-strength flat belts. Timing belts have teeth that fit into matched-toothed pulleys. They are often used to transfer direct motion for timing and when properly stressed, they move at a constant speed and have no slippage. They are often used instead of chains or gears to avoid using lubrication bath and to keep noise down. (Sabhadiya, 2021)

2.5.2.1 Calculations

The following part will be divided into three different sections regarding calculations: Geometry of belt drive, force transmission in belt drive and preload. These three parts are essential when it comes to build a foundation for the design of a belt drive.

Geometry of belt drive

This part will cover the geometry of a belt drive, as a system, and relevant calculations that can be done regarding this. Figure 2.5 shows the most relevant dimensions for a belt drive with two pulleys. The following equations are from (Mägi et al., 2017)

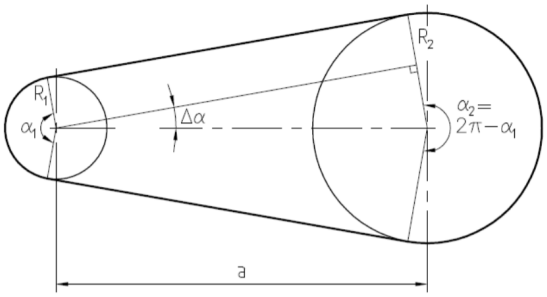


Figure 2.5: Geometry for belt drive with two different sized pulleys (Mägi et al., 2017)

- R_1 & R_2: Radius on the driving pulley and the driven pulley.
- α_1 & α_2: Wrapping angle of the driving and driven pulley.
- $\Delta\alpha$: Deviation from belt drive systems with parallel pulleys.
- a: Axis distance

Wrapping angle

The wrapping angle is the part of the pulley that is in contact with the belt. The wrapping angles is calculated as following:

$$sin(\Delta a) = \frac{R_2 - R_1}{a} \tag{2.21}$$

where

$$\alpha_1 = \pi - 2\Delta\alpha \tag{2.22}$$

$$\alpha_2 = \pi + 2\Delta\alpha \tag{2.23}$$

Belt length and Axis distance

How to calculate the belt length (L) and Axis distance (a) depends on which one is known. In order to calculate one of these, the other one needs to be known. If the axis distance is prescribed, $sin(\Delta a)$ can be directly calculated with Equation (2.21). The belt length can thereafter be calculated as following:

$$L = 2acos(\Delta\alpha) + R_1\alpha_1 + R_2\alpha_2 \tag{2.24}$$

If the belt length is prescribed and we want to calculate the axis distance, the method is a iterative process to calculate Δa in order to thereafter calculate the axis distance with Equation (2.21).

$$\Delta\alpha_{i+1} = arctan(1 + \Delta\alpha_i tan(\Delta\alpha_i)\frac{2(R_2 - R_1)}{L - \pi(R_2 + R_1)}) \tag{2.25}$$

Force transmission in belt drive

The figure below (2.6) shows the forces and torque on an operating belt drive during ideal conditions, it does not take conditions regarding slip and internal tensions and forces in the belt into consideration, this will be covered later on. The majority of the following equations are based on (Mägi et al., 2017), unless otherwise specified. Any deviations will be clearly indicated where applicable.

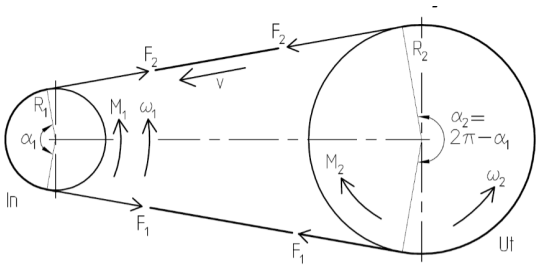


Figure 2.6: Forces an torque in an ideal belt drive (Mägi et al., 2017)

- M_1 & M_2: Torque on the driving and driven pulley.
- ω_1 & ω_2: Angular Velocity on the driving and driven pulley.
- F_1 & F_2: Tension forces
- v: Belt velocity.

The speed ratio is calculated as following:

$$i = \frac{\omega_1}{\omega_2} = \frac{R_2}{R_1} \tag{2.26}$$

Power is calculated as:

$$P = M_1\omega_1 = (F_2 - F_1)v \tag{2.27}$$

When forces will be calculated, note that variation of force from F_1 to F_2 is based on the friction between belt and pulley. Centrifugal force can be considered to be independent of the local curvature of the belt, and is calculated as following:

$$F_c = m'v_{belt}^2 \tag{2.28}$$

where m' is the belt weight and can be taken from ISO 4184. And v_{belt} is the speed of the belt and is calculated as following:

$$v_{belt} = 2R_{1,2}\pi v_{pulley}/60 \tag{2.29}$$

Note that R should be the radius of the same pulley used to determine the value of v_{pulley}.

With the speed of the belt can we now calculate the tensile force:

$$F_u = \frac{P}{v_{belt}} \tag{2.30}$$

If the system requires more than one belt, the centrifugal force is based on the number of belts. The amount of belts is a relevant parameter regarding design of belt drives system due to that the loads get distributed over the belts, i.e stronger system.

The number of belts is calculated as (Swedish Standards Institute, 1967):

$$Z = \frac{p_d}{p_r f_c f_d} \tag{2.31}$$

where the definitions of the input parameters are provided in the table below (2.1).

Table 2.1: Parameters for calculating required number of belts

p_d	Operating Power ($e \cdot p_d$), where e is the load factor
p_r	Transmittable power including additional power (which is taken from tables in SMS 2479 to 2485)
f_c	Correction factor for the belt length
f_d	Correction factor for the wrapping angle

F_1 and F_2 can be divided into centrifugal force and a effective force. Where the definition of F_e (Effective force) is:

$$F_{e,n} = F_n + F_c \tag{2.32}$$

In order to determine the effective force do we need to look into a a measure of the belt drive load which can be expressed with χ:

$$\chi = \frac{F_{e2}}{F_{e1}} \tag{2.33}$$

As can be seen, χ is the expression of the relation between the forces on the belts upper and lower side.

In order to determine the relation and taking the actual slip into consideration do we need to introduce χ_{slip}:

$$\chi_{slip} = (\frac{F_{e2}}{F_{e1}})_{slip} = e^{\mu_s \alpha} \tag{2.34}$$

μ is the slip coefficient for flat belts, in order to get μ_s which is for v-belts can we calculate it with:

$$\mu_s = \frac{\mu}{sin(\beta)} \tag{2.35}$$

Where β is the groove angle of the pulley. The groove angle can be taken from ISO 4183 and is based on the datum diameter of the pulley, note that if the pulleys got two different datum diameters, which results in two different sizes of groove angle, take the smaller one.

With belts with only controlled mikroslip, χ_{slip} is calculated:

$$\chi = e^{\theta \mu_s \alpha} \tag{2.36}$$

θ is the degree of utilization, which is a measure of how much of the belt's grip capacity is actually utilized.

By combining Equation (2.33) with the following:

$$\frac{F_{e2} - F_{e1}}{F_{e2} + F_{e1}} = \frac{\chi - 1}{\chi + 1} (= \lambda) \tag{2.37}$$

where $F_{e2} - F_{e1} = M/R$

Can F_{e1} and F_{e2} now be calculated as:

$$F_{e1} = \frac{M}{(\chi - 1)R} \tag{2.38}$$

$$F_{e2} = \chi F_{e1} \tag{2.39}$$

The condition for avoiding slip is as follows:

$$\frac{F_{e2}}{F_{e1}} < \chi_{slip} \tag{2.40}$$

Preload

Preload is a way to assure some pressure on the slack side of the belt, this condition must be fulfilled in ordet the get the belt drive to operate correctly (Mägi et al., 2017). There are several ways to assure a preload:

- Increase the axial distance between the pulleys by a given Δ a.
- Use a given axial load
- Tensioner pulley

- Weight of pivot-mounted drive unit
- Self-tensioning balanced drive unit
- Specified axial clamping force (V-belt)

Concerning this report will only the approaches regarding axial distance and axial load be addressed.

Given Δa

Note that the load factor, λ is the λ from Equation (2.37)

$$F_0 = \frac{M + 2\lambda R F_c}{R\lambda} \tag{2.41}$$

Given axialload

The axial load is, in this case, an outer force (F_{outer}) applied on on of the pulleys that's resulting in a required tension on the belt.

$$F_{outer} = F_{e1} + F_{e2} \tag{2.42}$$

in order to determine F_0 do we need to calculate a new load factor λ.

$$\lambda = \frac{M}{RF_{outer}} \tag{2.43}$$

(Mägi et al., 2017, p. 369) gives:

$$\lambda = \frac{M}{2R(F_0 - F_c)} \Rightarrow F_0 = \frac{M + 2R\lambda F_c}{2R\lambda} \tag{2.44}$$

By setting up an equilibrium equation over the belt drive system can we now calculate $F1$ and F_2, see fig 2.6, with the preload and tensile force.

$$F_1 = F_0 + \frac{F_u}{2} \tag{2.45}$$

$$F_2 = F_0 - \frac{F_u}{2} \tag{2.46}$$

This study does not consider stress-related conditioning effects or the load-bearing capacity of specific belt types. Standard belt performance criteria, such as long-term material behaviour under repeated loading or specific design ratings, are beyond the scope of this theoretical analysis.

2.5.3 Rolling Bearings

Rolling bearings are used to support and guide rotating or oscillating machine elements such as shafts, axles, or wheels, while minimizing friction. They also serve to transfer loads between different machine components. Thanks to their precise operation and minimal friction, rolling bearings allow for high rotational speeds and contribute to lower levels of noise, heat, energy use, and mechanical wear. They are also cost-effective and interchangeable, typically conforming to national or international dimensional standards.

There are two main types of rolling bearings, categorized by the shape of the rolling element: ball bearings and roller bearings.

Ball bearings use balls that make point contact with the bearing's ring raceways. When a load is applied, the contact point becomes an elliptical area. This small contact area results in low rolling friction, allowing ball bearings to operate at high speeds. However, the limited contact area also restricts their ability to carry heavy loads.

Roller bearings, in contrast, use rollers that make line contact with the ring raceways. As the load increases, the contact line expands and takes on a more rectangular shape. The larger contact area leads to higher friction but also allows roller bearings to support heavier loads than a ball bearing of the same size. However, this comes at the cost of lower permissible speeds. (SKF Group, 2018)

2.5.3.1 Calculation

The calculations presented in this section are based on the properties of spherical roller bearings, as the analysis primarily concerns this bearing type. The outer parameters affecting the bearing are radial load, axial load and rotational speed, as shown in Figure 2.7. The calculation of bearing life for spherical roller bearings follows SKF standards and involves several steps. The equations are from the book Maskinelement (Mägi et al., 2017), if nothing else is stated.

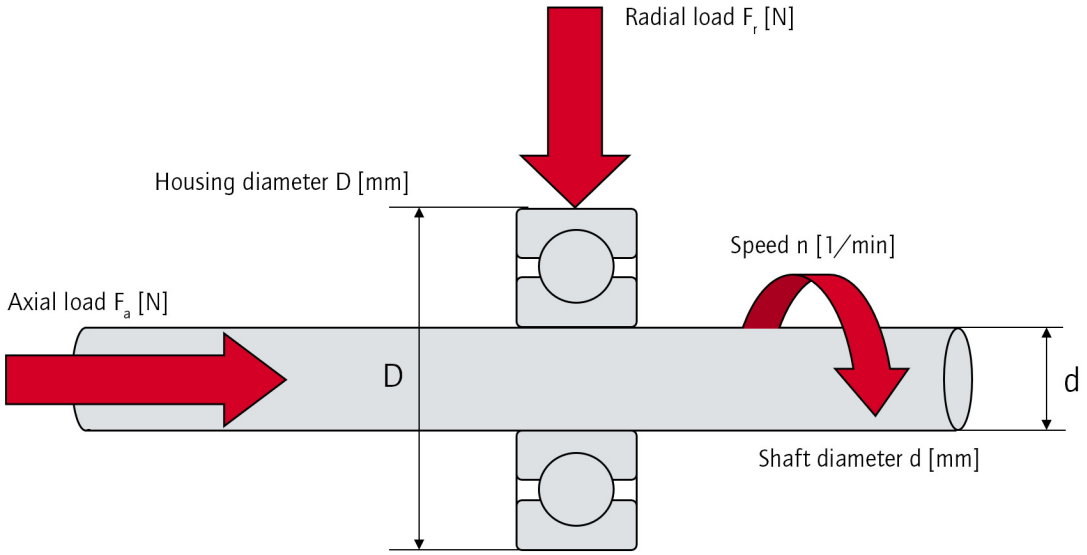


Figure 2.7: Free body diagram of a bearing (Findling Wälzlager GmbH, 2022)

First, the equivalent dynamic bearing load P is determined based on the relationship between the axial load F_a and the radial load F_r. If the ratio F_a/F_r is less than or equal to a given factor e, the equivalent load is calculated as:

$$P = F_r + Y_1 \cdot F_a \tag{2.47}$$

Otherwise, the following expression applies:

$$P = 0.67 \cdot F_r + Y_2 \cdot F_a \tag{2.48}$$

where Y_1 and Y_2 are load factors, and e is the load ratio factor, all of which are bearing specific and found in the SKF Rolling Bearings catalogue (SKF Group, 2018).

The basic rating life L_{10}, expressed in millions of revolutions, is then calculated using:

$$L_{10} = \left(\frac{C}{P}\right)^p \tag{2.49}$$

where C is the basic dynamic load rating in newtons (N), P is the equivalent dynamic load, and $p = \frac{10}{3}$ for spherical roller bearings.

To express the bearing life in operating hours, the following conversion is used:

$$L_{10h} = \frac{L_{10} \times 10^6}{60 \times n} \tag{2.50}$$

where n is the rotational speed in revolutions per minute (RPM).

To account for environmental conditions such as contamination, the contamination factor η_c is introduced as:

$$\eta_c \times \left(\frac{P_u}{P}\right) \tag{2.51}$$

where P_u is the fatigue load limit [N]. This value, together with the viscosity ratio κ, is used to determine the SKF life adjustment factor a_{SKF} from standard SKF diagrams.

The viscosity ratio κ is defined as:

$$\kappa = \frac{\nu}{\nu_1}$$

where ν is the actual operating viscosity of the lubricant, and ν_1 is the required (or rated) viscosity based on bearing size and speed. κ reflects the lubrication condition in the bearing and plays a critical role in estimating a_{SKF} (SKF Group, 2018).

The adjusted bearing life L_{nm}, considering reliability and operating conditions, is then calculated as:

$$L_{nm} = a_1 \times a_{SKF} \times L_{10} \tag{2.52}$$

The reliability factor a_1 accounts for the probability R that the bearing will reach or exceed its calculated basic life. a_1 can be determined using the following expression for reliability levels $R \geq 0.9$, which is typically the case in practical applications:

$$a_1 = 0.05 + 0.95 \left(\frac{\ln R}{\ln 0.9} \right)^{2/3} \tag{2.53}$$

This expression reflects the statistical distribution of fatigue life and enables the calculation of modified bearing life at any desired reliability level. For instance, at 95% reliability ($R = 0.95$), the factor becomes $a_1 \approx 0.62$, while at 99% reliability ($R = 0.99$), $a_1 \approx 0.21$. These reductions in life reflect the increased safety margins required for higher reliability.

Finally, the SKF-adjusted bearing life expressed in hours is:

$$L_{nmh} = \frac{L_{nm} \times 10^6}{60 \times n} \tag{2.54}$$

In summary, this procedure allows for the determination of both the basic and the adjusted bearing life, incorporating real-world influences such as contamination, lubrication quality, and required reliability levels in accordance with SKF and ISO methodologies.

This study does not account for varying load conditions, nor does it consider the effects of bearing systems involving multiple interacting elements. The influence of load variation over time or the distribution of forces across complex bearing assemblies falls outside the scope of this theoretical analysis.

2.5.4 Bolted joints

A bolt is a threaded fastener commonly used in mechanical assemblies to join two or more components together with a nut or by blind connection. Bolts are designed to withstand tensile and shear loads and play a critical role in ensuring structural integrity. When analysing bolts, it is important to consider parameters that affect their strength and function, such as geometry, material properties, applied loads, and the tightening moment.

2.5.4.1 Calculation

In this project, the calculations for the bolted joint are focused on analysing the preload force and tightening moment of the bolt to ensure the avoidance of leakage. The bolt of choice is an ISO M16 x 65 connecting two plates with a blind connection. All calculation formulas in this example are taken from the course literature (Mägi et al., 2017).

In order to make it possible to perform calculations on the bolt, a free-body diagram with all relevant forces acting on the bolt and the flange is required. The free body diagram in this case can be seen in Figure 2.8 below:

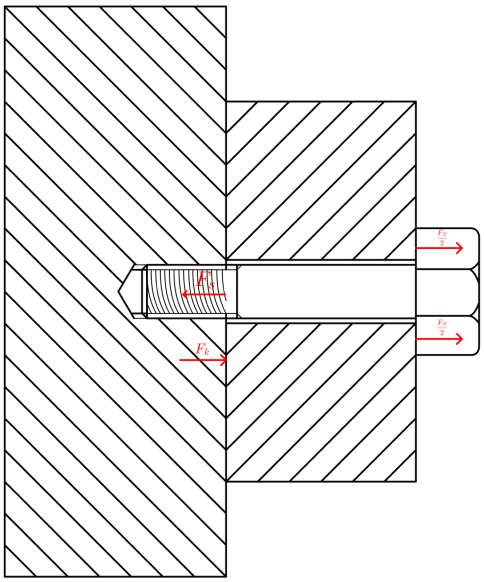


Figure 2.8: Free body diagram bolt

The equation for this free body diagram is as following:

$$\sum F_x = F_s - F_k - F_N = 0 \tag{2.55}$$

Now the resulting force in the screw F_s and the clamped part F_k can be determined:

$$F_s = F_0 + \frac{c_s}{c_s + c_k} F_N \tag{2.56}$$

$$F_k = F_0 - \frac{c_k}{c_s + c_k} F_N \tag{2.57}$$

To be able to calculate these forces, the stiffnesses need to be specified with following equations:

$$c_s = \frac{A_s E_s}{L_k} \tag{2.58}$$

$$c_k = \frac{A_{ekv} E_k}{L_k} \tag{2.59}$$

The area A_{ekv} is the active cross section area of the clamped part that takes the load, which is illustrated in Figure **??** below.

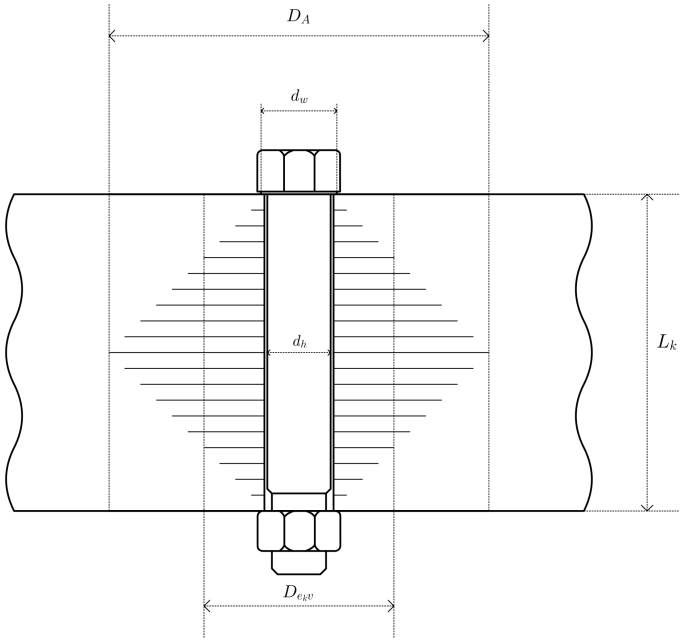


Figure 2.9: Illustration of A_{ekv} approximation

A_{ekv} can be calculated approximately in three different ways depending on how wide the clamped part is. The three different cases are A, B and C:

- A: $D_A > L_k + d_w \implies$

$$A_{ekv} = \frac{\pi}{4}(d_w^2 - d_h^2) + \frac{\pi}{8} L_k d_w ((x+1)^2 - 1) \tag{2.60}$$

  - If through hole with nut:

$$x = (\frac{L_k d_w}{(L_k + d_w)^2})^{\frac{1}{3}} \tag{2.61}$$

  - If screwed into material (no nut):

$$x = (\frac{L_k}{L_k + d_w})^{\frac{1}{5}} \tag{2.62}$$

- B: $D_A < L_k + d_w \implies$

$$A_{ekv} = \frac{\pi}{4}\left(d_w^2 - d_h^2\right) + \frac{\pi}{8}\left(D_A - d_w\right) d_w \left((x+1)^2 - 1\right) \tag{2.63}$$

  - If through hole:

$$x = (\frac{L_k d_w}{{D_A}^2})^{\frac{1}{3}} \tag{2.64}$$

  - If screwed into material (no nut):

$$x = (\frac{L_k}{D_A})^{\frac{1}{5}} \tag{2.65}$$

- C: $D_A \leq d_w \implies$

$$A_{ekv} = \frac{\pi}{4}({D_A}^2 - {d_h}^2) \tag{2.66}$$

In this study, case A was chosen for all calculaitons for bolts.

With these variables defined, it is possible to determine the preload force needed to prevent leakage. The preload force is taken from equation 2.56, and since the preload force is to prevent leakage, the clamped force can be set to 0:

$$F_0 = F_k + \frac{c_k}{c_k + c_s} F_N = \frac{c_k}{c_k + c_s} F_N \tag{2.67}$$

However, when a bolt is preloaded, permanent deformation occurs due to imperfections in the contact surfaces between the clamped parts and the bolt, this is called settling. To compensate for this, an additional preload is applied. The total amount of preload with settlement included is calculated through the equation below:

$$F_{0,settling} = F_0 + \frac{c_s c_k}{c_s + c_k} \delta_0 \tag{2.68}$$

δ_0 is the total length of deformation and is obtained through the following equation:

$$\delta_0 = \delta_s + |\delta_k| \tag{2.69}$$

With the preload force determined, the tightening moment required to reach this force can be calculated through the following equation:

$$M_{\mathrm{t}} = F_{0,\ \mathrm{settling}} \left(\frac{P}{2\pi} + \mu_{\mathrm{thread}} \left(0.58 d_2 + r_m \right) \right) \tag{2.70}$$

2.5.5 Bevel gears

Bevel gears, as seen in Figure 2.10, are used to transmit power between shafts positioned at almost any angle or speed. The most appropriate type of bevel gear for a particular application depends on the mounting setup, space constraints, and the conditions in which it will operate according to International Organization for Standardization, 2016 in ISO 23509.

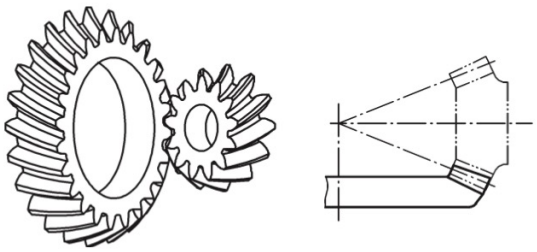


Figure 2.10: Spiral bevel gear, (ISO 23509 International Organization for Standardization, 2016)

Helical bevel gears feature curved, angled teeth where contact initiates at one end and gradually moves along the tooth surface. Although they engage in a manner similar to straight bevel gears, their overlapping tooth engagement results in smoother power transmission. This leads to reduced noise and vibration, particularly at higher operating speeds. Additionally, the tooth surfaces of spiral bevel gears can be precision-machined for enhanced performance according to International Organization for Standardization, 2016 in ISO 23509.

Bevel gears can be employed in both speed-reducing and speed-increasing systems. The necessary gear ratio should be calculated by the designer based on the specified input speed and the desired output speed according to International Organization for Standardization, 2016 in ISO 23509.

Geometry calculations for bevel gears

The geometry of both the pinion (usually the smaller gear, notated with subscript 1) and the wheel (usually the larger gear, notated with subscript 2) must be established to enable accurate durability calculations, as the gear geometry directly affects the load-carrying capacity of the teeth. The calculations presented for gear geometry are based on Method 0 from ISO 23509, International Organization for Standardization, 2016, which is applicable to non-hypoid bevel gears i.e. bevel gears where the centrelines of the two shafts intersect.

Determination of pitch cone parameters

Gear ratio is dependent on the number of teeth on the pinion and wheel. Equation (2.71) describes this relationship.

$$u = \frac{z_2}{z_1} \qquad \text{Gear ratio} \qquad (2.71)$$

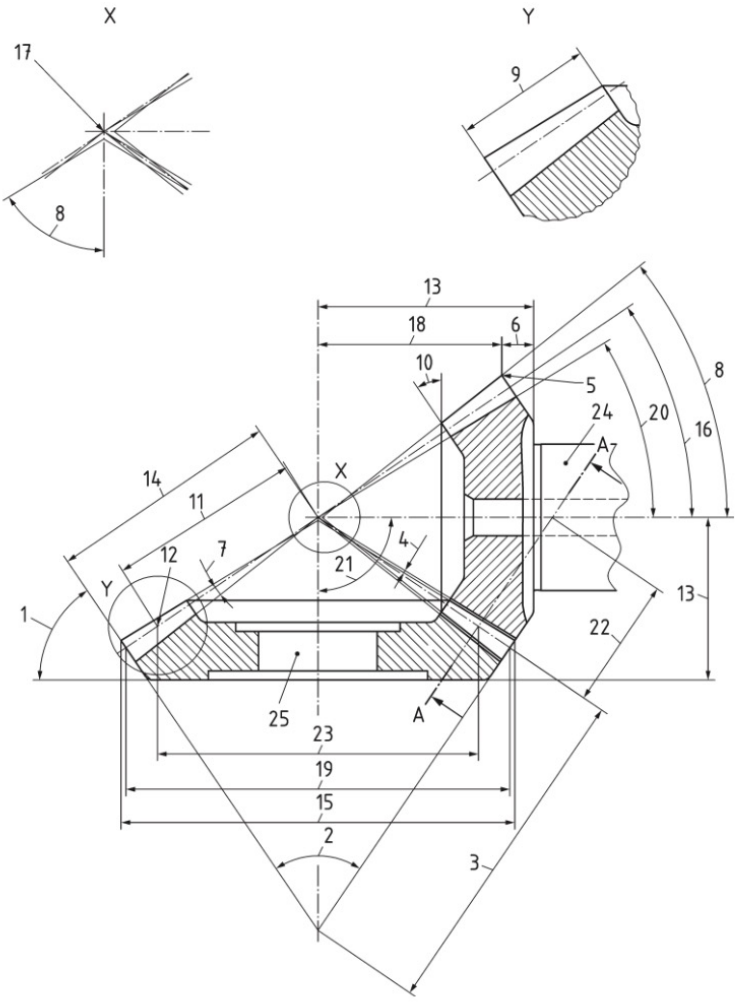
Speed of the wheel and pinion has the same relation as gear ratio, Equation (2.72).

$$n_1 = n_2 \cdot u \qquad \text{Pinion speed} \qquad (2.72)$$

Pitch angles, the angles of imaginary pitch cones that roll on each other without slipping, can be calculated with Equations (2.73), and (2.74). These angles can be seen in Figure 2.11 along with other geometry parameters that will be introduced later.

$$\delta_1 = \arctan\left(\frac{\sin(\Sigma)}{\cos(\Sigma) + u}\right) \qquad \text{Face angle, pinion} \qquad (2.73)$$

$$\delta_2 = \Sigma - \delta_1 \qquad \text{Face angle, wheel} \qquad (2.74)$$



Key

1	back angle	10	front angle	19	outer pitch diameter, d_{e1}, d_{e2}
2	back cone angle	11	mean cone distance, R_m	20	root angle, δ_{f1}, δ_{f2}
3	back cone distance	12	mean point	21	shaft angle, Σ
4	clearance, c	13	mounting distance	22	equivalent pitch radius
5	crown point	14	outer cone distance, R_e	23	mean pitch diameter, d_{m1}, d_{m2}
6	crown to back	15	outside diameter, d_{ae1}, d_{ae2}	24	pinion
7	dedendum angle, θ_{f1}, θ_{f2}	16	pitch angle, δ_1, δ_2	25	wheel
8	face angle δ_{a1}, δ_{a2}	17	pitch cone apex		
9	face width, b	18	crown to crossing point, t_{xo1}, t_{xo2}		

Figure 2.11: Bevel gear nomenclature for axial plane, (ISO 23509 International Organization for Standardization, 2016)

Once the outer pitch diameters d_{e1} and d_{e2} are determined with (2.75), the outer transverse modules m_{et1} and m_{et2}, measured perpendicular to the axis of rotation, can be calculated using the number of teeth on each gear with (2.76).

$$d_{e1,2} = 2R_e \sin(\delta_{1,2}) \qquad \text{Outer pitch diameter} \qquad (2.75)$$

$$m_{et1,2} = \frac{d_{e1,2}}{z_{1,2}} \qquad \text{Outer transverse module} \qquad (2.76)$$

The outer cone distance R_e can be calculated using the known outer pitch diameter of the wheel d_{e2} and the pitch angle δ_2, Equation (2.77)

$$R_e = \frac{d_{e2}}{2\sin(\delta_2)} \qquad \text{Outer cone distance} \tag{2.77}$$

Face width is normally either the outer transverse module times ten or 30% of outer cone distance, whichever is less. This can also vary depending on the case. The convention is described with Equations (2.78), (2.79), and (2.80).

$$b_{\text{met}} = m_{et2} \cdot 10 \tag{2.78}$$

$$b_{Re} = 0.3 \cdot R_e \tag{2.79}$$

$$b = \min(b_{\text{met}}, b_{Re}) \qquad \text{Face width (selected)} \tag{2.80}$$

Inner cone distance is calculated by subtracting the face width form the outer cone distance, Equation (2.81). The mean cone distance is the mean from the outer and inner cone distance or just half of the face width subtracted from the external cone width, Equation (2.82). These can be but are not always the same for both the pinion and wheel as the cones have the same length sides but different pitch angles.

$$R_i = R_e - b \qquad \text{Inner cone distance} \tag{2.81}$$

$$R_m = R_e - \frac{b}{2} \qquad \text{Mean cone distance} \tag{2.82}$$

The mean spiral angle β_m, which defines the helical orientation of the gear teeth, is determined iteratively to satisfy the required overlap contact ratio ϵ_β. This relationship is governed by Equation (2.83), which includes K_Z defined in Equation (2.84). The iteration proceeds until the calculated contact ratio matches the design requirement.

$$\epsilon_\beta = \frac{1}{\pi m_{et2}} \left(K_Z \tan(\beta_m) - \frac{K_Z^3}{3} \tan^3(\beta_m) \right) R_e \qquad \text{Overlap contact ratio} \tag{2.83}$$

$$K_Z = \frac{b}{R_e} \cdot \frac{2 - \frac{b}{R_e}}{2\left(1 - \frac{b}{R_e}\right)} \qquad \text{Overlap factor} \tag{2.84}$$

Determination of basic data

The inner and mean pitch diameters for both the pinion and wheel are calculated using basic trigonometric relationships. The inner pitch diameter is based on the inner cone distance and the pitch angle, as shown in Equation (2.85). The mean pitch diameter is then the average of the outer and inner pitch diameters, as shown in Equation (2.86).

$$d_{i1,2} = 2R_i \sin(\delta_{1,2}) \qquad \text{Inner pitch diameter} \tag{2.85}$$

$$d_{m1,2} = \frac{d_{e1,2} + d_{i1,2}}{2} \qquad \text{Mean pitch diameter} \tag{2.86}$$

The mean normal module m_{mn}, which is the average module measured perpendicular to the tooth flank and accounts for the spiral angle, is calculated using Equation (2.87). This parameter is essential for evaluating tooth geometry and load distribution.

$$m_{mn} = \frac{2R_m \sin(\delta_2)\cos(\beta_m)}{z_2} \qquad \text{Mean normal module} \tag{2.87}$$

The crossing point of the pitch cones, referred to as the apex, marks the intersection of the axes of the pinion and the wheel. From this point, the distances along the gear axes to the mean pitch circle and the pitch apex are critical for defining the gear geometry. The distance from the apex to the mean point on the opposing gear cone is calculated using Equation (2.88), while the total distance from the apex to the base of the gear cone (referred to here as the pitch apex) is computed with Equation (2.89).

$$t_{zm1,2} = \frac{d_{m2,1}}{2} \qquad \text{Distance from apex to mean pitch point} \tag{2.88}$$

$$t_{z1,2} = R_m \cos(\delta_{1,2}) - t_{zm1,2} \qquad \text{Apex to base of gear cone} \tag{2.89}$$

For bevel gears where the pinion has twelve teeth or more, the mean addendum factor c_{ham} is calculated using the equivalent gear ratio u_a, as defined in Equation (2.90). This ratio is then used to compute the addendum factor according to Equation (2.91).

$$u_a = \sqrt{\frac{\cos(\delta_1)\cdot\tan(\delta_2)}{\cos(\delta_2)}} \qquad \text{Equivalent ratio} \tag{2.90}$$

$$c_{\text{ham}} = 0.210 + \frac{0.290}{u_a^2} \qquad \text{Mean addendum factor, } z_1 \geqslant 12 \tag{2.91}$$

$$\tag{2.92}$$

The circular thickness factor k_t, which accounts for both the gear ratio and the number of pinion teeth, is determined using Equation (2.93).

$$k_t = -0.088 + 0.092u - 0.004u^2 + 0.0016(z_1 - 30)(u - 1) \qquad \text{Circular thickness factor} \tag{2.93}$$

These values are subsequently used to compute tooth geometry corrections. The profile shift coefficient x_{hm1} is given by Equation (2.94), and the mean normal circular tooth thickness x_{smn} by Equation (2.95).

$$x_{hm1} = k_d\left(\frac{1}{2} - c_{\text{ham}}\right) \qquad \text{Profile shift coefficient} \tag{2.94}$$

$$x_{smn} = \frac{k_t}{2} \qquad \text{Mean normal circular tooth thickness} \tag{2.95}$$

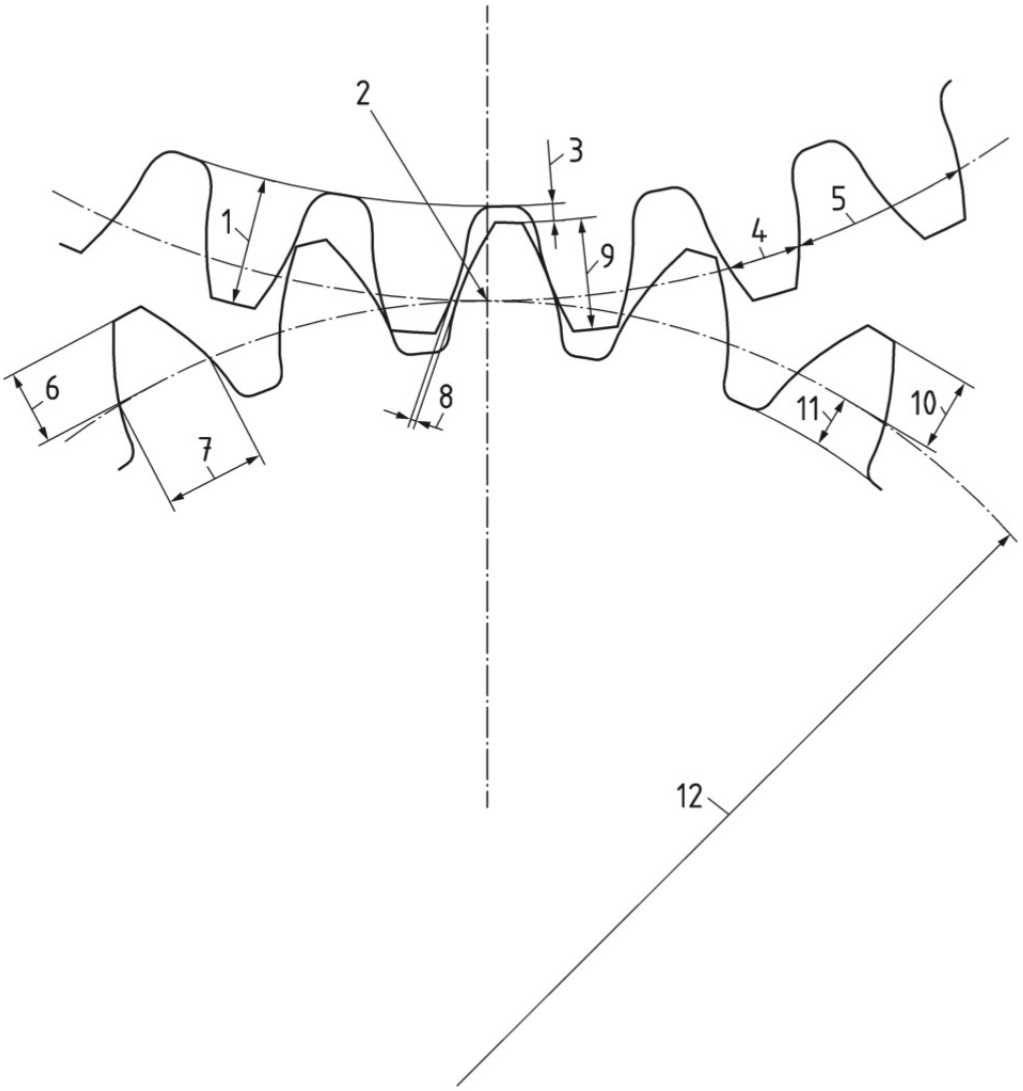
Face width adjustments are made using the following formulas. The face width from the calculation point to the outside is given by Equation (2.96), and the face width from the calculation point to the inside by Equation (2.97).

$$b_e = b \cdot c_{be} \quad \text{Face width from calculation point to outside} \tag{2.96}$$
$$b_i = b \cdot c_{be} \quad \text{Face width from calculation point to inside} \tag{2.97}$$

Determination of tooth depth at calculation point

The equations below are used to calculate tooth depth and modifications at the calculation point. The mean working depth is given by Equation (2.98), the mean addendum for the wheel by Equation (2.99), and the mean dedendum for the wheel by Equation (2.100). Similarly, the mean addendum for the pinion is given by Equation (2.101), while the mean dedendum for the pinion is given by Equation (2.102). Clearance is calculated using Equation (2.103), and the mean whole depth is given by Equation (2.104). These can be visulised in Figure 2.12

$$h_{mw} = 2 \cdot m_{mn} \cdot k_{\text{hap}} \quad \text{Mean working depth} \tag{2.98}$$
$$h_{am2} = m_{mn} \cdot (k_{\text{hap}} - x_{hm1}) \quad \text{Mean addendum, wheel} \tag{2.99}$$
$$h_{fm2} = m_{mn} \cdot (k_{\text{hfp}} + x_{hm1}) \quad \text{Mean dedendum} \tag{2.100}$$
$$h_{am1} = m_{mn} \cdot (k_{\text{hap}} + x_{hm1}) \quad \text{Mean addendum, pinion} \tag{2.101}$$
$$h_{fm1} = m_{mn} \cdot (k_{\text{hfp}} - x_{hm1}) \quad \text{Mean dedendum, pinion} \tag{2.102}$$
$$c = m_{mn} \cdot (k_{\text{hfp}} - k_{\text{hap}}) \quad \text{Clearance} \tag{2.103}$$
$$h_m = h_{am1} + h_{fm1} \quad \text{Mean whole depth} \tag{2.104}$$



Key

1	whole depth, h_m	5	circular pitch	9	working depth, h_{mw}
2	pitch point	6	chordal addendum	10	addendum, h_{am}
3	clearance, c	7	chordal thickness	11	dedendum, h_{fm}
4	circular thickness	8	backlash	12	equivalent pitch radius

Figure 2.12: Bevel gear nomenclature for mean transverse section, (ISO 23509 International Organization for Standardization, 2016)

The addendum and dedendum angles for the wheel are determined based on the mean cone distance and the corresponding tooth depths. The sum of the dedendum angles is calculated using Equation (2.105), which adds the angular contributions from both the pinion and the wheel. The wheel's addendum angle is given by Equation (2.106), and its dedendum angle is obtained by subtracting the addendum angle from the total, as shown in Equation (2.107).

$$\theta_{\Sigma f_s} = \tan^{-1}\left(\frac{h_{fm1}}{R_m}\right) + \tan^{-1}\left(\frac{h_{fm2}}{R_m}\right) \quad \text{Sum of dedendum angles} \quad (2.105)$$

$$\theta_{a2} = \tan^{-1}\left(\frac{h_{fm1}}{R_m}\right) \quad \text{Addendum angle, wheel} \quad (2.106)$$

$$\theta_{f2} = \theta_{\Sigma f_s} - \theta_{a2} \quad \text{Dedendum angle, wheel} \quad (2.107)$$

Determination of root angles and face angles

The face and root angles for the wheel, which correspond to the angles from the pitch cone to the tip and root of the teeth respectively, are calculated using Equations (2.108), and (2.109). These are based on the pitch angle δ_2 and the previously determined addendum and dedendum angles.

$$\delta_{a2} = \delta_2 + \theta_{a2} \quad \text{Face angle, wheel} \quad (2.108)$$

$$\delta_{f2} = \delta_2 - \theta_{f2} \quad \text{Root angle, wheel} \quad (2.109)$$

The corresponding face and root angles for the pinion are derived from the geometry of the mating wheel and are shown in Equations (2.110), and (2.111). The pinion's addendum and dedendum angles are then calculated using Equations (2.112), and (2.113), respectively.

$$\delta_{a1} = \arcsin\left(\cos(\delta_{f2})\right) \quad \text{Face angle, pinion} \quad (2.110)$$

$$\delta_{f1} = \arcsin\left(\cos(\delta_{a2})\right) \quad \text{Root angle, pinion} \quad (2.111)$$

$$\theta_{a1} = \delta_{a1} - \delta_1 \quad \text{Addendum angle, pinion} \quad (2.112)$$

$$\theta_{f1} = \delta_1 - \delta_{f1} \quad \text{Dedendum angle, pinion} \quad (2.113)$$

The apex positions of the face and root cone elements for both the wheel and the pinion are essential for defining the gear geometry and locating the boundaries of the tooth. The wheel face and root apex positions are given by Equations (2.114), and (2.115), respectively, and depend on the mean cone radius R_m, the addendum and dedendum heights, and their associated angles.

$$t_{zF2} = t_{z2} - \frac{R_m \cdot \sin(\theta_{a2}) - h_{am2} \cdot \cos(\theta_{a2})}{\sin(\delta_{a2})} \quad \text{Wheel face apex} \quad (2.114)$$

$$t_{zR2} = t_{z2} + \frac{R_m \cdot \sin(\theta_{f2}) - h_{fm2} \cdot \cos(\theta_{f2})}{\sin(\delta_{f2})} \quad \text{Wheel root apex} \quad (2.115)$$

The corresponding pinion apex positions are computed using the results from the wheel side and the clearance c, as shown in Equations (2.116) and (2.117).

$$t_{zF1} = \frac{-t_{zR2} \cdot \sin(\delta_{f2}) - c}{\sin(\delta_{a1})} \quad \text{Pinion face apex} \quad (2.116)$$

$$t_{zR1} = \frac{-t_{zF2} \cdot \sin(\delta_{a2}) - c}{\sin(\delta_{f1})} \quad \text{Pinion root apex} \quad (2.117)$$

Determination of inner and outer spiral angles

The distances from the apex to the outer and inner boundaries of the pinion cone, denoted as R_{e21} and R_{i21}, are calculated based on the mean cone distance R_m and the adjusted face widths b_e and b_i. These define the full span of the pinion cone and are given by Equations (2.118) and (2.119).

$$R_{e21} = \sqrt{R_m^2 + b_e^2 + 2R_m b_e} \quad \text{Cone distance at outer pinion boundary} \tag{2.118}$$

$$R_{i21} = \sqrt{R_m^2 + b_i^2 - 2R_m b_i} \quad \text{Cone distance at inner pinion boundary} \tag{2.119}$$

The spiral angles at the boundaries of the pinion and wheel are calculated with Equations (2.120) and (2.121) for the outer and inner boundaries, respectively.

$$\beta_e = \arcsin\left(\frac{2R_m r_{c0}\sin(\beta_m) - R_m^2 + R_{e21}^2}{2R_{e21} r_{c0}}\right) \quad \text{Spiral angle at outer pinion boundary} \tag{2.120}$$

$$\beta_i = \arcsin\left(\frac{2R_m r_{c0}\sin(\beta_m) - R_m^2 + R_{i21}^2}{2R_{i21} r_{c0}}\right) \quad \text{Spiral angle at inner pinion boundary} \tag{2.121}$$

Tooth Depth at Inner and Outer Boundaries

The outer mean tooth depths for both the pinion and the wheel are calculated by extending the addendum and dedendum from the mean cone point using the tangent of the respective angles. These are given in Equations (2.122) to (2.124).

$$h_{ae_{1,2}} = h_{am_{1,2}} + b_e \cdot \tan(\theta_{a_{1,2}}) \quad \text{Outer addendum, pinion and wheel} \tag{2.122}$$

$$h_{fe_{1,2}} = h_{fm_{1,2}} + b_e \cdot \tan(\theta_{f_{1,2}}) \quad \text{Outer dedendum, pinion and wheel} \tag{2.123}$$

$$h_{e_{1,2}} = h_{ae_{1,2}} + h_{fe_{1,2}} \quad \text{Outer whole depth, pinion and wheel} \tag{2.124}$$

The inner mean tooth depths for both the pinion and the wheel are determined similarly to the outer depths, except the face width component is subtracted due to the direction of taper. These are given in Equations (2.125) to (2.127).

$$h_{ai_{1,2}} = h_{am_{1,2}} - b_i \cdot \tan(\theta_{a_{1,2}}) \quad \text{Inner addendum, pinion and wheel} \tag{2.125}$$

$$h_{fi_{1,2}} = h_{fm_{1,2}} - b_i \cdot \tan(\theta_{f_{1,2}}) \quad \text{Inner dedendum, pinion and wheel} \tag{2.126}$$

$$h_{i_{1,2}} = h_{ai_{1,2}} + h_{fi_{1,2}} \quad \text{Inner whole depth, pinion and wheel} \tag{2.127}$$

Tooth thickness calculation

The thickness modification coefficients for the pinion and the wheel are calculated using Equation (2.128) and Equation (2.129), respectively.

$$x_{sm1} = x_{smn} - \frac{j_{en}}{4m_{mn}\cos(\alpha_n)} \cdot \frac{R_m\cos(\beta_m)}{R_e\cos(\beta_{e2})} \quad \text{Thickness modification coefficient, pinion} \quad (2.128)$$

$$x_{sm2} = -x_{smn} - \frac{j_{en}}{4m_{mn}\cos(\alpha_n)} \cdot \frac{R_m\cos(\beta_m)}{R_e\cos(\beta_{e2})} \quad \text{Thickness modification coefficient, wheel} \quad (2.129)$$

The mean normal circular tooth thickness for the pinion and the wheel is determined using Equation (2.130) and Equation (2.131), respectively. These expressions account for both the thickness modification coefficients and the influence of the profile shift coefficient x_{hm1} along with the normal pressure angle α_n.

$$s_{mn1} = \frac{\pi}{2}m_{mn} + 2m_{mn}\left(x_{sm1} + x_{hm1}\tan(\alpha_n)\right) \quad \text{Mean normal circular thickness, pinion} \quad (2.130)$$

$$s_{mn2} = \frac{\pi}{2}m_{mn} + 2m_{mn}\left(x_{sm2} - x_{hm1}\tan(\alpha_n)\right) \quad \text{Mean normal circular thickness, wheel} \quad (2.131)$$

The mean transverse circular tooth thickness for both the pinion and the wheel is calculated using Equation (2.132) and Equation (2.133), respectively. These values are derived from the mean normal thickness and account for the spiral angle β_m.

$$s_{mt1} = \frac{s_{mn1}}{\cos(\beta_m)} \quad \text{Mean transverse circular thickness, pinion} \quad (2.132)$$

$$s_{mt2} = \frac{s_{mn2}}{\cos(\beta_m)} \quad \text{Mean transverse circular thickness, wheel} \quad (2.133)$$

The mean normal pitch diameters for the pinion and the wheel are calculated using Equation (2.134). This formula accounts for the effects of the spiral angle β_m, the normal pressure angle α_n, and the pitch angles δ_1 and δ_2.

$$d_{mn1,2} = \frac{d_{m1,2}}{\left(1 - \sin^2(\beta_m)\cos^2(\alpha_n)\right)\cos(\delta_{1,2})} \quad \text{Mean normal pitch diameter, pinion and wheel} \quad (2.134)$$

The mean normal chordal tooth thickness for the pinion and wheel are calculated by Equation (2.135).

$$s_{mnc1,2} = d_{mn1,2} \cdot \sin\left(\frac{s_{mn1,2}}{d_{mn1,2}}\right) \quad \text{Pinion and Wheel, mean normal chordal tooth thickness} \quad (2.135)$$

The mean normal chordal tooth thickness for the pinion and wheel are calculated by Equation (2.136). This formula uses the mean normal pitch diameters $d_{mn1,2}$, along with the corresponding normal tooth thicknesses $s_{mn1,2}$ and s_{mn2}.

$$h_{amc1,2} = h_{am1,2} + \frac{1}{2} d_{m1,2} \cos(\delta_1, 2) \left(1 - \cos\left(\frac{s_{mn1,2}}{d_{mn1,2}}\right)\right) \quad \text{Pinion and Wheel, mean chordal addendum} \quad (2.136)$$

Remaining gear dimensions

The pitch cone distances for the pinion are calculated with Equations (2.137) and (2.138)

$$R_{e1} = R_m + b_e \quad \text{Outer pitch cone distance} \quad (2.137)$$

$$R_{i1} = R_m - b_i \quad \text{Inner pitch cone distance} \quad (2.138)$$

The diameters for the pinion and wheel are given by Equations (2.139) - (2.142).

$$d_{ae1,2} = d_{e1,2} + 2h_{ae1,2} \cos(\delta_{1,2}) \quad \text{Outer tip diameter} \quad (2.139)$$

$$d_{fe1,2} = d_{e1,2} - 2h_{fe1,2} \cos(\delta_{1,2}) \quad \text{Outer root diameter} \quad (2.140)$$

$$d_{ai1,2} = d_{i1,2} + 2h_{ai1,2} \cos(\delta_{1,2}) \quad \text{Inner tip diameter} \quad (2.141)$$

$$d_{fi1,2} = d_{i1,2} - 2h_{fi1,2} \cos(\delta_{1,2}) \quad \text{Inner root diameter} \quad (2.142)$$

The pitch apex positions for both the pinion and the wheel are calculated as follows Equations (2.143) and (2.144).

$$t_{xo1,2} = t_{zm1,2} + b_e \cos(\delta_1, 2) - h_{ae1,2} \sin(\delta_{1,2}) \quad \text{Pitch cone apex to crown} \quad (2.143)$$

$$t_{xi1,2} = t_{zm1,2} - b_i \cos(\delta_{1,2}) - h_{ai1,2} \sin(\delta_{1,2}) \quad \text{Front crown to crossing point} \quad (2.144)$$

The tooth height for the pinion is calculated by:

$$h_{t1} = \frac{t_{zF1} + t_{xo1}}{\cos(\delta_{a1})} \cdot \sin(\theta_{a1} + \theta_{f1}) - (t_{zR1} - t_{zF1}) \cdot \sin(\delta_{f1}) \quad (2.145)$$

Force calculaitons for bevel gears

Force distribution in a bevel gear assembly depends on the direction of rotation in comparison to the spiral angle, the normal pressure angle and the size of the gears. In this section the he analysis of forces according to ISO 23509, International Organization for Standardization, 2016, will be presented.

Torque

Torque is the input power divided by the roatation speed in radians, as seen in Equation (2.146).

$$T_{1,2} = \frac{P}{n_{1,2\text{rad}}} \quad \text{Torque} \quad (2.146)$$

Distribution of Forces

The distribution of forces in the bevel gear pair is illustrated in Figure 2.13. The direction of the forces depends on the sense of rotation. When rotating in one direction, the gears push each other apart, as shown on the left side of the figure. When rotating in the opposite direction, they pull each other together, as shown on the right. These opposing force distributions result in reversed axial and radial force components and must be considered when analysing load paths and bearing reactions.

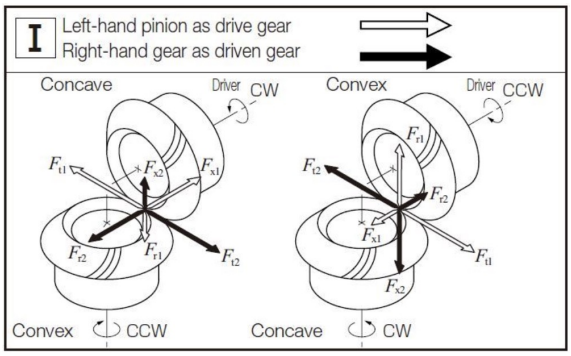


Figure 2.13: Bevel gear force distribution, (KHK Gears, 2024)

The tangential force on the pinion at the mean point of contact is calculated using a basic torque–radius relationship, as shown in Equation (2.147). Since bevel gear dimensions are typically given in millimeters, the radius is converted to meters by dividing by 10^3. The tangential force on the wheel has the same magnitude but acts in the opposite direction, as given in Equation (2.148).

$$F_{mt1} = \frac{T_1}{\frac{d_{m1}}{2} \cdot 10^{-3}} \qquad \text{Tangential force on pinion} \tag{2.147}$$

$$F_{mt2} = F_{mt1} \qquad \text{Tangential force on wheel (equal magnitude)} \tag{2.148}$$

The tangential force can be resolved into components, the first being the *axial force*, which acts along the axis of the shaft. This component is influenced by the normal pressure angle α_n, the spiral angle β_m, and the pitch angle δ. Depending on the direction of rotation, the gears may either push each other apart or pull each other together, resulting in different axial force directions. The axial forces for the pinion and the wheel, when the gears are pushing each other apart, are calculated using Equations (2.149) and (2.150), respectively. These forces act along the gear shafts and depend on the normal pressure angle α_n, the spiral angle β_m, and the pitch angles δ_1 and δ_2.

$$F_{ax1} = \left(\frac{\tan(\alpha_n)\sin(\delta_1)}{\cos(\beta_m)} + \tan(\beta_m)\cos(\delta_1) \right) F_{mt1} \qquad \text{Axial force on pinion} \tag{2.149}$$

$$F_{ax2} = \left(\frac{\tan(\alpha_n)\sin(\delta_2)}{\cos(\beta_m)} - \tan(\beta_m)\cos(\delta_2) \right) F_{mt1} \qquad \text{Axial force on wheel} \tag{2.150}$$

The tangential force can also be resolved into a second component, the *radial force*, which acts perpendicular to the shaft and points toward or away from the gear center, depending on the geometry and direction of rotation. This force component is influenced by the normal pressure angle α_n, the spiral angle β_m, and the pitch angle δ. As with the axial force, the direction of rotation affects whether the gears tend to pull together or push apart, altering the orientation of the radial force. The radial forces for the pinion and the wheel, when the gears are pushing each other apart, are calculated using Equations (2.151) and (2.152), respectively.

$$F_{rad1} = \left(\frac{\tan(\alpha_n)\cos(\delta_1)}{\cos(\beta_m)} - \tan(\beta_m)\sin(\delta_1) \right) F_{mt1} \quad \text{Radial force on pinion} \quad (2.151)$$

$$F_{rad2} = \left(\frac{\tan(\alpha_n)\cos(\delta_2)}{\cos(\beta_m)} + \tan(\beta_m)\sin(\delta_2) \right) F_{mt1} \quad \text{Radial force on wheel} \quad (2.152)$$

Calculations for Safety

Pitting and tooth fracture safety factors can be calculated using the Swedish standard SS 1862 (SIS - Standardiseringskommissionen i Sverige, 1978). This standard employs a methodology similar to the ISO 10300 series (International Organization for Standardization, 2023), but offers a more compact and simplified implementation. SS 1862 provides specific formulas for calculating both the actual and allowable loads related to flank pressure (pitting) and tooth bending, and can also be used to derive the corresponding safety factors.

To begin the calculations, an equivalent force F_{ber} is determined according to Equation (2.155). This force is an amplified version of the tangential load F_{mt1}, accounting for dynamic effects and load conditions. The dynamic factor K_{v1} is calculated using Equation (2.154), which in turn depends on the mean tangential velocity v_{mt1}, given by Equation (2.153). The load factor K_1 is selected from a table in SS 1862 based on the specific load conditions and support configuration.

$$v_{mt1} = \frac{d_{m1} \cdot n_{1,\text{rad}}}{2000} \quad \text{Mean tangential speed} \quad (2.153)$$

$$K_{v1} = \sqrt{\frac{78 + 14\sqrt{v_{mt1}}}{78}} \quad \text{Dynamic factor} \quad (2.154)$$

$$F_{\text{ber}} = F_{mt1} \cdot K_1 \cdot K_{v1} \quad \text{Equivalent load} \quad (2.155)$$

Contact Stress (Hertzian Pressure)

The contact stress, also referred to as the Hertzian pressure σ_H, is calculated according to the methodology outlined in SS 1862 SIS - Standardiseringskommissionen i Sverige, 1978, which is based on Hertzian contact theory. This theory evaluates the localized stress in the contact zone between mating gear teeth.

The calculation begins with the determination of the effective transverse pressure angle α_t, which adjusts the normal pressure angle α_n for the influence of the spiral angle β_m, as shown in Equation (2.156). This angle is then used to compute the geometry factor Z_H, which accounts for the curvature of the contacting surfaces (Equation (2.157)).

The cone length factor Z_K (Equation (2.158)) accounts for the geometry of bevel gears and their conical tooth engagement, based on the outer (R_e) and inner (R_i) pitch cone distances. The material factor Z_M (Equation (2.159)) reflects the elastic modulus of the gear material. The contact ratio factor Z_ε can normally be put to one (Equation (2.160)).

The final contact stress σ_H is calculated using Equation (2.161), which incorporates the previously discussed factors along with the equivalent force F_{ber}, face width b, mean pitch diameter d_{m1}, gear ratio u, and load distribution factors $K_{H\alpha}$ and $K_{H\beta}$.

$$\alpha_t = \arctan\left(\frac{\tan(\alpha_n)}{\cos(\beta_m)}\right) \quad \text{Effective pressure angle} \quad (2.156)$$

$$Z_H = \sqrt{\frac{2\cos(\beta_m)}{\sin(2\alpha_t)}} \quad \text{Geometry factor} \quad (2.157)$$

$$Z_K = \frac{2R_e}{R_e + R_i} \quad \text{Cone length factor} \quad (2.158)$$

$$Z_M = \sqrt{0.35 \cdot E_{\text{modul}}} \quad \text{Material factor} \quad (2.159)$$

$$Z_\varepsilon = 1 \quad \text{Contact ratio factor} \quad (2.160)$$

$$\sigma_H = Z_H Z_K Z_M Z_\varepsilon \sqrt{\frac{F_{\text{ber}} K_{H\alpha} K_{H\beta} \sqrt{u^2+1}}{b d_{m1} u}} \quad \text{Hertzian contact stress} \quad (2.161)$$

Allowable Contact Stress

The allowable contact stress σ_{HP} is the maximum permissible Hertzian pressure that the gear teeth can endure under given conditions. This value is determined based on material properties, lubrication conditions, surface finish, expected service life, and a specified safety factor. The calculation follows the method described in SS 1862 SIS - Standardiseringskommissionen i Sverige, 1978 and incorporates several modifying factors, as shown in Equation (2.162).

The base value $\sigma_{H\text{lim}}$ represents the material's contact fatigue limit. This is adjusted by the following factors:

- K_L: **Lubrication factor** – Accounts for the quality and type of lubrication used in the gear system.
- Z_R: **Surface roughness factor** – Reflects the impact of surface finish on the fatigue strength of the teeth.
- Z_V: **Hydrodynamic lubrication film factor** – Considers the effect of the lubricant film supporting the gears during operation.
- K_{HX}: **Size factor for contact stress** – Adjusts for dimensional effects related to the contact area.
- K_{HN}: **Life factor for contact stress** – Accounts for the expected number of load cycles over the gears' operational life.
- K_{HK}: **Hardness combination factor** – Reflects the difference in hardness between mating gears, impacting the resistance to pitting and wear.
- S_H: **Safety factor for pitting resistance** – A design factor applied to ensure a margin of safety against pitting failure.

These factors are typically set according to standard recommendations or chosen based on design requirements. When no special adjustments are required, each factor may be set to unity.

$$\sigma_{HP} = \frac{\sigma_{H\text{lim}} \cdot K_L \cdot Z_R \cdot Z_V \cdot K_{HX} \cdot K_{HN} \cdot K_{HK}}{S_H} \quad \text{Allowable contact stress} \quad (2.162)$$

Bending Stress

The bending stress σ_F in the gear tooth root is calculated based on the applied equivalent load F_{ber}, the face width b, and the normal module m_{mn}. This stress is influenced by several correction and geometry-dependent factors. The formula according to SS 1862 is given in Equation (2.163).

$$\sigma_F = \frac{Y_F \cdot Y_K \cdot Y_\beta \cdot Y_\varepsilon \cdot F_{\text{ber}} \cdot K_{F\alpha} \cdot K_{F\beta}}{b \cdot m_{mn}} \quad \text{Bending stress} \quad (2.163)$$

The influencing factors are:

- Y_F: **Tooth form factor** – Reflects the geometry of the tooth root and can be determined from standard tables (e.g., Table 8.4 in SS 1862).
- Y_K: **Tooth length factor** – Accounts for the effective load distribution along the tooth length:
$$Y_K = \left(\frac{2R_e}{R_e + R_i}\right)^2$$
- Y_β: **Helix angle factor** – Compensates for the load-sharing effects in helical gears:
$$Y_\beta = \begin{cases} 1 - \frac{\beta_m^\circ}{120} & \text{if } \beta_m \leq 30^\circ \\ 0.75 & \text{otherwise} \end{cases}$$
- Y_ε: **Contact ratio factor** – Set to 1.0 in this analysis unless otherwise noted.
- $K_{F\alpha}$: **Transverse load factor for bending** – Takes into account uneven load distribution along the path of contact. Set to 1.0 for simplified cases.
- $K_{F\beta}$: **Face load factor for bending** – Reflects load variation along the face width. Equal to $K_{H\beta}$, the corresponding factor for contact stress.

Virtual Gear Geometry To determine the tooth form factor Y_F used in bending stress calculations, the bevel gear geometry must be translated into equivalent spur gear parameters. This is achieved through the concept of virtual gear geometry, allowing empirical formulas and standard tables developed for spur gears to be applied to bevel gears. The transformation includes calculating a virtual addendum modification x_{vn1}, a virtual number of teeth z_{v1} and z_{v2}, as well as a virtual base helix angle β_{bm}. These are defined in Equations (2.164)–(2.167).

$$x_{vn1} = \frac{x_{hm1}}{\cos(\beta_m)} \quad \text{Virtual addendum modification} \tag{2.164}$$

$$z_{v1} = \frac{z_1\sqrt{1+u^2}}{2}, \quad z_{v2} = z_2\sqrt{1+u^2} \quad \text{Virtual number of teeth (pinion and wheel)} \tag{2.165}$$

$$\beta_{bm} = \arctan(\tan(\beta_m)\cos(\alpha_t)) \quad \text{Virtual base helix angle} \tag{2.166}$$

$$z_{vn1,2} = \frac{z_{v1,2}}{\cos(\beta_m)\cos^2(\beta_{bm})} \quad \text{Virtual number of teeth (normal)} \tag{2.167}$$

These virtual quantities serve to approximate the geometry of the bevel gear teeth as if they were spur gears, enabling the use of standardized empirical formulas and lookup tables for stress evaluation.

Allowable Bending Stress

To ensure that the gear teeth do not fail due to fatigue at the root, the allowable bending stress σ_{FP} must be calculated. This stress represents the maximum stress the tooth root can safely endure under repeated loading and is determined using Equation (2.168).

The calculation incorporates the material's fatigue strength σ_{Flim}, a safety factor S_F, and various correction and application factors as defined in SS 1862 SIS - Standardiseringskommissionen i Sverige, 1978. These include the stress correction factor Y_s, the size factor K_{FX}, and the life factor K_{FN}, all of which account for geometric effects, component dimensions, and expected service life, respectively.

$$\sigma_{FP} = \frac{\sigma_{Flim} Y_s K_{FX} K_{FN}}{S_F} \tag{2.168}$$

Where:

- σ_{Flim} — Bending fatigue limit of the material (in MPa)
- Y_s — Stress correction factor (accounts for geometry and stress concentrations)
- K_{FX} — Size factor for bending stress
- K_{FN} — Life factor for bending stress (adjusts for desired service life)
- S_F — Safety factor for bending fatigue (typically > 1)

These factors are defined in **SS 1862** and are selected based on material properties, gear geometry, operational lifetime, and load conditions. For a safe design, the calculated bending stress σ_F must not exceed σ_{FP}.

3
Method

This section outlines the methodology used to compare theoretical calculations with CAD-software calculations, and within the CAD-software themselves. The method followed a structured workflow (see Figure 3.1) that were divided into three phases: concept building, operational, and evaluation.

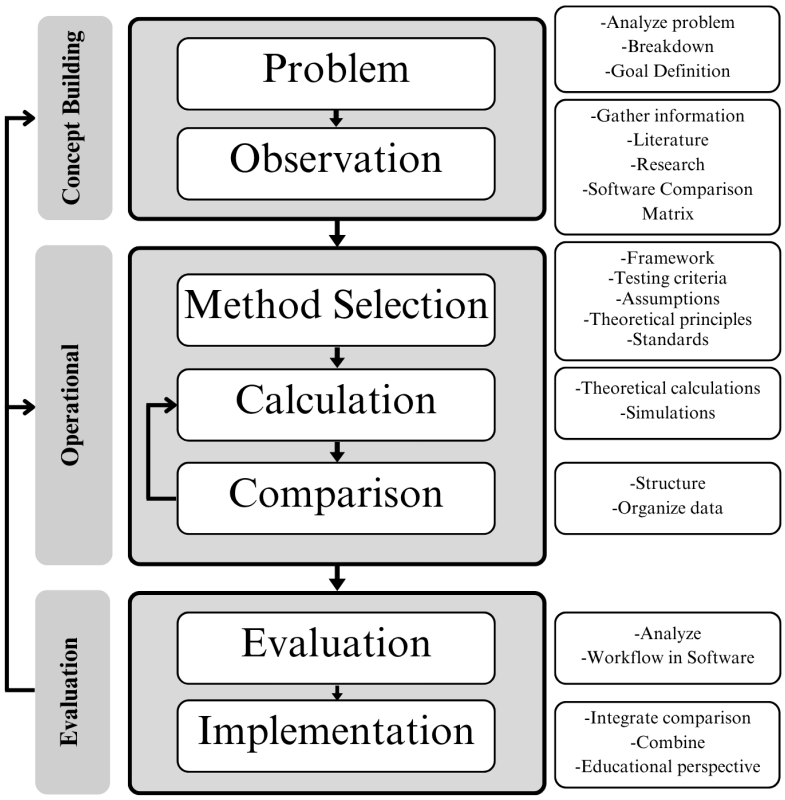


Figure 3.1: Methodology worflow - the workflow throughout the study

3.1 Concept building

The first phase, Concept Building, focused on analysing the problem and create a problem definition. In the observation part, a literature review was conducted, focusing on mechanical principles, machine elements, and available CAD software. To determine which CAD software to be used in the study, a CAD software comparison matrix was created. The matrix covered relevant software that occurred based on research and discussion. Furthermore, an analysis was conducted that addressed whether and to what extent the software was capable of doing calculations on different machine elements. Whether the software was capable or not depended on if the program had specific tools for the elements in question and not just executed Finite Element Analysis (FEA), i.e, did the software considered the machine elements as elements or just as solid object? If the software considered them as elements, does that mean it is also able to execute relevant calculations, design procedures, etc.?

For the software that qualified based on the matrix mentioned above it was evaluated what kind of input data the software required to perform the calculations, i.e. how much information does the user need beforehand to perform the calculations. How well the respective help desks for using the software was built and also what kind of output data was received after the calculations was performed and how did it present the output, lists, 2D or/and 3D. This was systematically done by conducting a thorough review of each program with each element and documenting the results of what were discovered in an excel sheet.

The review was based and constructed to answer the following questions.

- Required input data: What data is necessary to perform calculations?
- Other input options: Are additional, optional parameters available?
- Output data/results: What results are generated, and in what format?
- Result presentation: How are the results visualized (tables, diagram, 3D models)?
- Calculation capabilities: Can the software perform extended or iterative analysis?
- CAD integration: Is it possible to create or import CAD models?
  - If yes, can the model be created manually in the program?
  - Can external CAD files be imported for analysis?
- Design vs. analysis: Does the software support design from input values or evaluate existing design?

3.2 Operational

In the first part, method selection, a methodological framework was created based on findings from the concept phase. Testing criteria were established, assumptions clarified, and relevant theoretical principles selected. As a result, a specific case definition was created for each machine element. These case definitions included both the necessary input values, such as operating time, applied loads, safety factors, and stress levels - among others. As well as a structured list of parameters to be investigated. Each case were designed to serve as a consistent and elementary reference for analysing single-element calculations, both through manual methods and using CAD tools.

The second step, calculations, contained both theoretical calculations and simulations in the selected software. Firstly, hand calculations in Python were done using standard formulas from engineering textbooks and standards. The assumptions made during the hand calculations, such as material properties, simplifications, and safety factors, were clearly noted. The results from the hand calculations were used as a reference point to compare against the calculations in the CAD software. Then, a simplified model was created in the CAD software, and the program's calculation feature was used to obtain relevant values. The results were clearly noted.

In the comparison, the results were organized side-by-side to analyse how closely the software outputs aligned between themselves and with theory, to further be able to analyse differences and similarities.

3.3 Evaluation

This phase included Evaluation and Implementation. The first one covered investigation regarding discrepancies, particularly in terms of variation in results, usage of different standards or assumptions in the software. As well as the influence of default settings or software-specific simplifications, transparency and user control over the calculations.

The last part, implementation, covered how software tools could be used in education, not just for design, but to support learning and understanding of mechanical theory.

This structured method allowed for an objective and repeatable approach when analysing different machine elements and comparing performance. It furthermore ensured that the project stayed structured, relevant, and aligned with both engineering practice and educational value.

4
Results

The following section presents the results of the study. It begins by introducing the software tools used, including their capabilities, functionality, and application within the project. This is followed by the results related to the machine elements.

4.1 CAD software functionalities

The research for relevant CAD software to conduct the study on resulted in eleven different CAD software. After a deeper analysis of the software, with help of the comparison matrix, see Appendix A, five of the eleven software were deemed capable of performing calculations on all of the chosen machine elements. Due to complications in acquiring licenses for some of the software and the time limitations for doing so, three of the five software were chosen to do the study on. The software chosen were:

- MITCalc
- Inventor
- MDesign

Inventor is a CAD software, and MDesign and MITCalc are more computational software that uses input data to provide output data from its calculations.

4.1.1 MITCalc

MITCalc is a plug-in software for several CAD programs, such as Autodesk, SolidWorks and Solid Edge. It's a program with the goal to make day-to-day calculations faster and easier by implementing a collection of manufacturing, engineering and technical calculations. MITCalc is excel based and can compute calculations in several different areas, everything from aerodynamics and fluid mechanics to machine elements and welding. To see the completed list of possible calculations, see (MITCalc Team, 2025). Most of the calculations supports visualizations tools, such as 2D and even, in some cases, 3D, this is where the CAD software comes in hand.

The general workflow in MITCalc is structured into three main sections: the input section, results section, and supplement section, with more specific subsections where the user are able to insert the input, make changes and view results, for a visualisation of the programs user interface, see Appendix B. This structure is consistent across most modules within the software. On most of the sections and subsections, MITCalc provides help sections with short descriptions that covers information and relevant points of considerations to take into account. MITCalc does also provide an hyperlink to its website that contains more information, among others, the used standards.

Input section

In the Input section, the user is required to enter the necessary design and operating parameters. This section is typically divided into subcategories such as *Loading*, *Operational*, and *Design* parameters, though the specific structure may vary depending on the type of calculation being performed. While this section primarily focuses on user input, it also includes preliminary internal calculations to verify and guide the input process.

Results section

The results section presents the output from the core calculations. The computed values are generally fixed and cannot be altered manually by the user, with some exceptions. This section serves to provide results mostly based on the input data provided, but also by taking values from relevant tables, mostly from implemented standards.

Supplement section

The Supplement section combines both input and output elements, offering the user the opportunity to make additional modifications or refinements to the model. It also includes the Graphic Output/CAD System functionality, which allows the user to export the project into either a 2D drawing or a 3D model for further use within a CAD environment.

4.1.2 Inventor

Autodesk Inventor is a 2D and 3D CAD software that enables users to create, simulate, and document digital prototypes. The software supports both top-down and bottom-up design strategies, allowing the user to either develop individual parts and then assemble them into a complete assembly, or to design parts directly within the assembly environment.

The software also allows for the import of standardized components across a wide range of machine elements through the integrated Design Accelerator module. For a complete list of supported machine elements, see their website (Autodesk, 2023). Within the Design Accelerator, users may either begin by inputting operational loads and receive design suggestions for standardized components capable of withstanding these loads, or start with a predefined component and then perform an analysis using the built-in calculation tool. To initiate the analysis, the user selects the component to be evaluated, upon which a calculation dialogue appears. For a clearer understanding of this interface, see Appendix C.

Within the calculation dialogue, Inventor visually illustrates for some components what each parameter represents, often through simplified diagrams such as free-body sketches or analogous representations for other machine elements. This helps the user's understanding of the computational model and the meaning of each input variable. Additional explanations for all parameters, some accompanied by illustrative figures, are available on Inventor's help page (Autodesk, 2025).

The calculation dialogue is capable of identifying parameters that exceed material properties or other value errors, based on standardized dimensions and material characteristics. Furthermore, the Design Accelerator can perform fatigue analysis on components subjected to cyclic loading.

4.1.3 MDesign

MDesign is a calculation software intended for engineers and designers working within the fields of mechanical and plant engineering. It supports the calculation, verification, and documentation of various machine elements based on established engineering standards.

The general workflow begins with the selection of the desired component category, such as bearings, shafts, or belt drives. Once a module is selected, the software prompts the user to input relevant design parameters, including dimensions, material properties, operating speeds, and loads. After the required data is entered, the calculation are performed by the software.

The output varies depending on the type of component being analysed. For elements like rolling bearings and belt drives, MDesign generates a list of suitable component variants that satisfies the specified input conditions and complies with applicable standards. These results include references to relevant standards as well as calculated values, such as bearing life or belt speed.

For other components, such as shafts or bolted joints, the software provides primarily calculated values rather than suggestions for standardized parts. These values typically included stress levels, safety factors, and deformation, based on the specified load cases and input geometry.

The user interface of MDesign is reasonably straightforward and easy to navigate. Input fields are clearly labelled and logically organized according to the selected machine element. Additionally, the software includes supplementary help features, such as tables of standard values and explanation fields, which aid understanding of the required input parameters and support efficient use without prior experience of the software. The user interface during bearing life calculations can be seen in Appendix D.

Overall, the output from MDesign consists either of a selection of standardized component options that fulfill the given requirements, or a set of calculated performance metrics for the analysed component. The structure and content of the results is determined by the nature of the machine element and the calculation method applied within the software.

4.2 Calculation Results

The calculations for the selected machine elements performed on the aforementioned CAD tools in Section 4.1 as well as by hand following the theory in Section 2.5. The calculations results are presented in this section.

4.2.1 Shafts

The goal with the following calculations regarding shafts was to determine key stress and deformation parameters, as well as basic physical properties.
The analysis aimed to calculate the following parameters:

- Bending stress
- Shear stress
- Torsional stress
- Axial (tension) stress
- Safety factor (based on von Mises stress)
- Angular deflection
- Ideal shaft mass

The shaft was divided into five sections with varying lengths and diameters. External forces and moments were applied based on a simplified mechanical scenario.
Material properties such as Young's modulus and Poisson's ratio were assumed according to typical structural steel standards. The calculations were based on cross-sectional analysis, superposition of stresses, and simplified static loading conditions, see Table 4.1. Results were later used as a benchmark for comparison with automated outputs.

Table 4.1: Case definition for Shaft

Parameter	Value	Unit
Section 1 length (l_1)	140	mm
Section 2 length (l_2)	56	mm
Section 3 length (l_3)	480	mm
Section 4 length (l_4)	80	mm
Section 5 length (l_5)	210	mm
Shaft diameters	85, 110, 125, 110, 100	mm
Center of gravity z_{cg}	485.9	mm
Torque (T)	961.56	Nm
Axial force (F_{axial})	5237.84	N
Belt force (F_a)	4927.76	N
Cutting force (F_r)	3359.07	N
Mass (shaft + load)	48.1144	kg
Gravitational acceleration (g)	9.81	m/s^2
Weight force (F_g)	472.5	N
Tool support force (F_b)	29236	N
Material density (ρ)	7850	kg/m^3
Allowable stress (σ_{allow})	335	MPa

Table 4.2 below compares the results of shaft calculations obtained from different methods: hand calculations, MITcalc, Inventor, and MDesign. The parameters include various stress types, safety factors, angular deflection, and mass, which are crucial for evaluating the performance and design of the shaft. The result graphs are presented in the Appendix E. The values obtained from the hand calculations are generated using a custom Python script developed for shaft analysis, see Appendix F.

Table 4.2: Results from hand calculations and software - Shafts

Variable	Hand Calc	MITcalc	Inventor	MDesign	Unit
Max Bending Stress	61.129	62.537	62.673	62.537	MPa
Max Shear Stress	3.722	3.700	3.739	3.7	MPa
Max Torsional Stress	7.974	7.953	7.953	7.593	MPa
Max Tension Stress	0.923	0.868	0.868	0.9	MPa
Equivalent Stress	62.65	63.107	63.572	63.107	MPa
Safety Factor	5.35	5.63	–	5.63	—
Angular Deflection	0.0539	0.0807	0.05	0.0807	deg
Mass	75.570	75.57	75.666	75.62	kg

MITCalc

MITCalc results were very close to the hand calculations, with only minor deviations well within acceptable limits. The software follows standard DIN743 (Deutsches Institut für Normung, 2012) and performs iterative calculations based on user input. However, transparency is somewhat limited, as it is not always clear how intermediate values are derived.

An important characteristic of MITCalc is that the user must follow a predefined input structure. The tool requires fixed design parameters to be entered such as:

- Transmitted power
- Shaft speed
- Shaft geometry
- Dimensions
- Loading conditions in a specific sequence

This means that the user does not have complete freedom to define all values independently; instead, they are somewhat restricted by the tool's internal logic.

Once the input data is provided, the software offers suggestions for suitable materials and corresponding ultimate tensile strengths, further guiding the design process according to established mechanical standards. The shear force and bending moment diagrams generated by MITCalc are presented in Figure 4.1 and Figure 4.2, respectively. One noticeable difference in the results is the angular deflection, where the values differ quite significantly. The hand calculation yields a deflection of 0.0539°, while MITCalc reports a larger value of 0.0807°. This discrepancy can be attributed to several factors.

Firstly, the hand calculation is based on a simplified model that assumes a constant torque applied across the entire shaft length. In contrast, MITCalc likely considers a more realistic load distribution, where the torque may vary depending on the positions of pulleys, gears, or other torque-transmitting elements along the shaft. For MITCalc's load distribution see Appendix G.

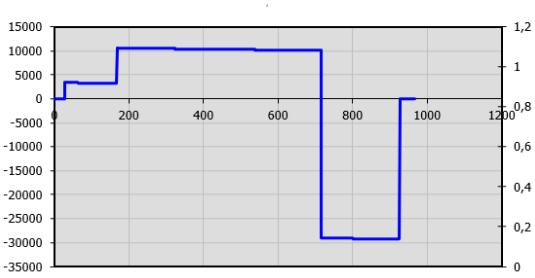


Figure 4.1: Shear Force Diagram from MITCalc.

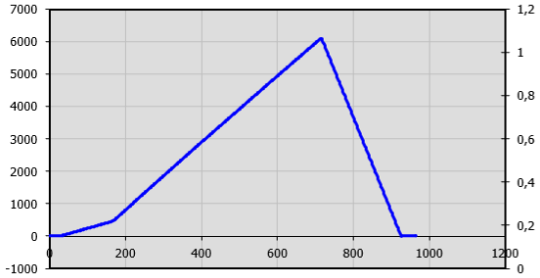


Figure 4.2: Bending Moment Diagram from MITcalc.

Secondly, in the hand calculation, the shaft is divided into a few discrete sections, each with a constant diameter. MITCalc, on the other hand, may use a finer segmentation or even a continuous geometry profile, which allows for a more precise computation of the polar moment of inertia along the shaft. Since the angular deflection is directly dependent on both the shaft's geometry and the applied torque, such modeling differences can have a significant impact.

Additionally, stress concentration factors are also considered in the MITCalc calculation, which can further contribute to differences between the numerical and hand calculations. MITCalc applies standard values for stress concentration factors such as the notch factor (k_t), surface factor (k_s), and size factor (k_d) based on the shaft geometry, fillet radii, surface finish, and dimensions. These factors amplify the local stresses at critical points, leading to slightly higher calculated stresses and deflections compared to the more idealized hand calculations where such concentration effects may not have been fully considered.

Autodesk Inventor

Inventor's Shaft Component Generator allows the user to build a shaft step by step by combining cylindrical sections of different geometries and then applying external loads such as axial forces, radial forces, and torques. Unlike MITCalc, where the user inputs predefined parameters into a structured form, Inventor offers more flexibility by enabling the user to freely define the geometry and loading conditions directly within the model. Each section of the shaft can be visually adjusted, and a full strength analysis can then be performed, where the software calculates bending and torsional stresses, resulting deflections, and safety factors. The results are presented through diagrams showing shear forces, bending moments, and stress distributions along the shaft, giving the user clear insight into how each section is affected. The shear force and bending moment distributions can be seen in the shear force diagram (see Figure 4.3) and the bending moment diagram (see Figure 4.4), respectively. Material properties such as Young's modulus, shear modulus, and density can be manually adjusted or selected from a list, and they directly influence the outcome of the analysis. While Inventor does not explicitly follow a fixed standard like MITCalc does, it provides greater freedom in modeling. However, it is the user's responsibility to ensure the shaft's design complies with practical and standard mechanical guidelines, as the CAD environment does not inherently restrict values according to predefined mechanical standards. In this way, Inventor functions as a more open-ended analysis tool where the user retains control over both design and veryficaton.

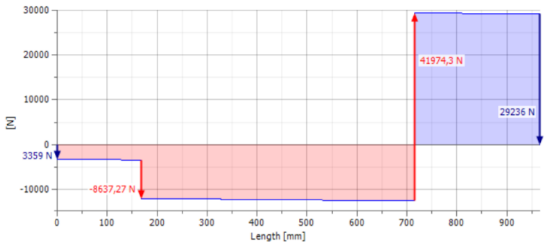


Figure 4.3: Shear Force Diagram from Autodesk Inventor.

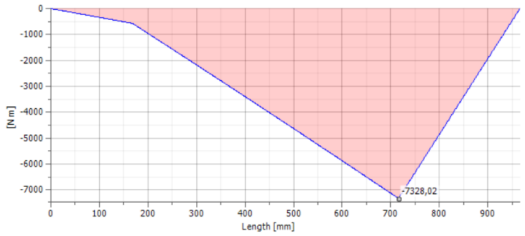


Figure 4.4: Bending Moment Diagram from Autodesk Inventor.

MDesign

MDesign and MITCalc are both specialized programs for machine element calculations, but they differ in their approach and scope, particularly when analysing shafts. Both rely on established engineering standards, DIN 743 (Deutsches Institut für Normung, 2012) in the case of MDesign which provide a solid framework for evaluating both static and dynamic strength.

In MDesign, all input and analysis are performed within a highly structured interface where each parameter is carefully defined. The software accounts for the influence of geometric discontinuities (notches), surface roughness, material treatments, and the nature of applied loads in a way that reflects advanced strength of materials theory. The user defines the shaft's geometry, loading conditions, and material properties according to a step-by-step flow, and MDesign then calculates safety factors against failure, fatigue, and plastic deformation. Results are presented both numerically and graphically, including diagrams of bending moments, torsion, and shear force distributions along the shaft, as well as identifying critical stress locations.

One noticeable difference between the methods is the result for angular deflection. The hand calculation yields a value of 0.0539°, whereas MDesign reports a significantly higher value of 0.0807°, which is consistent with the result from MITCalc.

The same reasoning applied to MITCalc can therefore be extended to MDesign — the discrepancy is likely due to simplified assumptions about the torque distribution along the shaft. While the hand calculation is based on a more detailed, section-by-section analysis, MDesign likely assumes a constant torque load, which can lead to an overestimation of the angular deflection.

Additionally, this difference may partly be attributed to the fact that the hand calculation was based on the standard assumptions provided in the KTH handbook, while MDesign applies a different base case for deflection calculations, possibly considering additional effects or slightly different loading conditions. As such, the base assumptions used for the deflection part may not have been identical between the two methods.

4.2.2 Belt Drives

For belt drive, the calculations were based on a standard belt drive setup, including two pulleys of different diameters and a belt of known material. Assumptions were stated, such as neglecting belt elasticity and assuming constant belt velocity.

The general problem formulation that was created for belt drive was defined as follows (4.3):

Table 4.3: Case definition for belt drive

Parameter	**Value**	**Unit**
Transferred power	75	kW
Velocity (Driving pulley)	1500	RPM
Velocity (Driven pulley)	500	RPM
Efficiency	95%	—
Center distance pulley	< 1	m
Operating time	8h/day	—
Type of driving unit (Loading)	Heavy shock	—
Type of driven machine (Loading)	Medium Duty	—

Seeking:

- What belt type profile should be used
- Number of belts
- Pulley dimensions (Wrapping angle)
- Belt length
- Required Preload
- Forces

In order to evaluate which belt type to use, a load factor was chosen based on the *Type of driving unit*, *Type of driven machine* and *Operating time*. SMS 2476 was used as a standard, this includes a table from where the load factor could be determine. furthermore, from a graph (see Figure 4.5) based on velocity of driving pulley and the operating power, the belt type profile could be determined. belt type C fitted the requirements best and got chosen. In order to establish the diameter of the driving pulley, SMS 2492 was used, the standard shows an interval of possible diameters, based on a tolerance value K, which can be taken from SMS 2493. By looking into SMS 2492 the diameter of the bigger pulley could be determined based on the speed ratio and the diameter of the smaller pulley.

The subsequent hand calculations regarding wrapping angle, belt length, required preload and forces followed the same workflow as presented in the theory part (Section 2.5.2.1). The length of the belt got established based on the fact that the axial distance pulley should not exceed 1 meter.

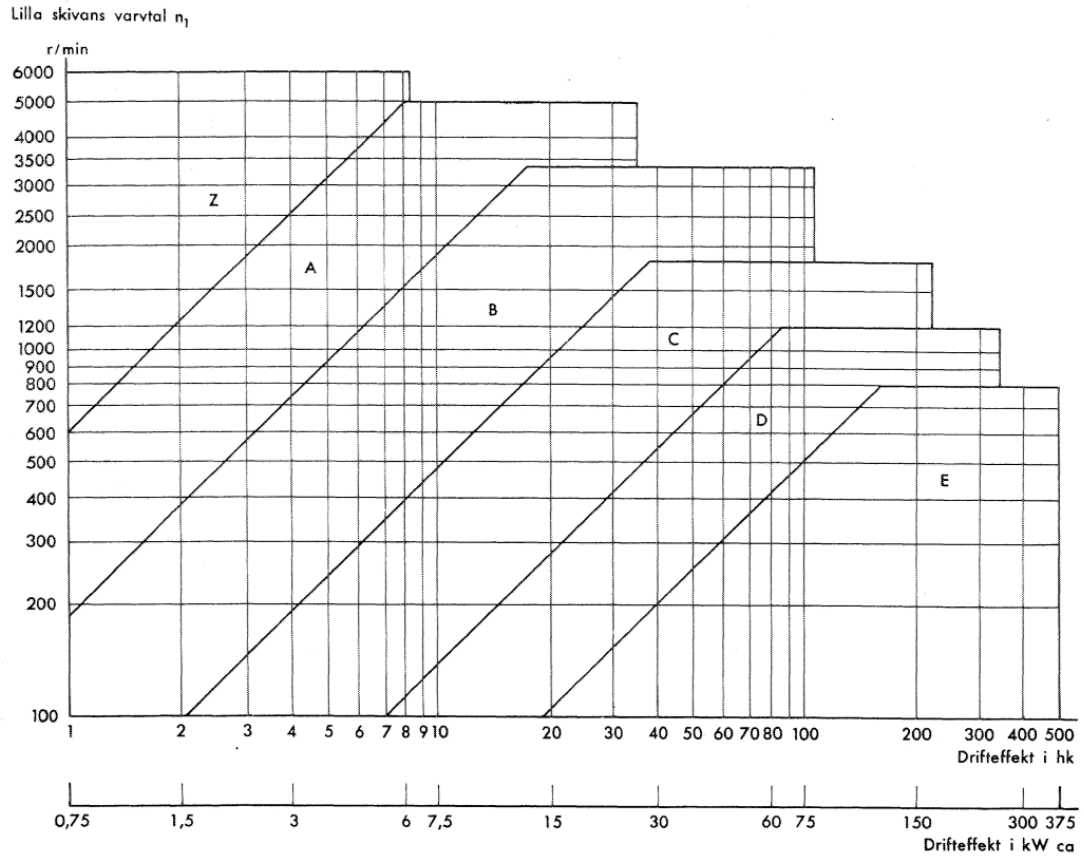


Figure 4.5: Choose belt profile (Swedish Standards Institute, 1967, p. 4, graph 3)

In the following will the results from the case definition raised above be presented. Table 4.4 represents the results from: Hand calculations, MITCalc, Inventor and MDesign. Thereafter will the results be analysed, divided into the software. Note that hand calculations were conducted with a python code, see Appendix H

Table 4.4: Results from hand and software calculations - belt drive

Variable	**Hand calculations**	**MITCalc**	**Inventor**	**MDesign**	**unit**
Transferred power	71.25	71.25	71.25	—	kW
Speed ratio	3	3	3	3.17	—
Torque Pulley 1	477.46	477.5	477.46	—	Nm
Torque Pulley 2	1360.77	1360.88	1378.91	—	Nm
Belt length	3.65	3.623	3.602	3.602	m
Axis distance	—	—	0.97418	1.039	m
Wrapping angle 1	150.45	151.36	149.65	153	Deg
Wrapping angle 2	209.54	208.64	210.35	—	Deg
Number of belts	10	12	$11.09 \Rightarrow 12$	9	—
Tensile force (F_u)	3819.72	3819.72	3819.72	—	N
Friction coefficient	0.756025	0.83	—	—	—
Effective force (F_{e1})	1740.91	—	—	—	N
Effective force (F_{e2})	5560.63	—	—	—	N
Belt speed	19.635	19.63	19.635	17.59	m/s
Centrifugal force (F_c)	1232.78	1387.91	1156.59	—	N
Pre load [1]	4883.55	4704.06 [2]	—	—	N
Pre load [3]	3212.86		—	—	N
F_1	6793.41	6613.92	6589.766	—	N
F_2	2973.69	2794.20	2770.048	—	N
Velocity Driven pulley				473	RPM

[1] Calculated pre load based on given axial load.
[2] Not based on any specific approach regarding calculation of pre load.
[3] Calculated pre load based on axial distance

Results from MITcalc

MITCalc initially recommended belt type SPB, this is a parameter where the user got the possibility to change if wanted. This was done, and changed to belt type C, due to that the evaluation of differences, regarding calculations, between the programs and the hand calculations should, advantageously, be based on the same belt type. MITCalc also initially recommended 200 mm as the diameter of the smaller pulley, due to that this only is a value that can be taken from a table and somewhere be chosen based on one's discretion, this got changed by user to 250 (same as the hand calculations).

Something noteworthy is that MITCalc only presented one value for preload, even though there are several ways to compute this. Compared to the two pre load values within hand calculations can it be noticed that the calculated value for pre load with a given axial load is much closer to the one computed in MITCalc.

MITcalc used American- (ANSI), German- (DIN) and international standards (ISO). The list of used standards were as followed:

- ANSI/RMA
  - IP-20
  - IP-22
  - IP-23
- DIN
  - 7753
  - 2211
  - 2215
- ISO 4148

Results from Inventor

Inventor requires different kinds of input, the first big difference is that it is not able to, by specific input, generate a belt type proposal. Which belt type to use is one input that's required from the user to choose. Amount of belts and length of these was also a required input from the user. Due to that Number of belts was a input data from the user, the following results are based on 10 belts. Inventor did a calculation on this, it says that required amount of belts is 11.09 $\Rightarrow$ 12. When trying to use this as the input value for number of belts, the program was not able to execute calculations due to "Insufficient number of belts". For the input parameter belt length, the user selected a value from a predefined list of standardized lengths. Consequently, the axial distance became a calculated output variable (see Section 2.5.2.1). Since one of the design constraints was that the axial distance must not exceed 1 meter, it was necessary to continuously monitor this value and iterate the selection of belt length to ensure compliance with the requirement. This was also the case for MDesign.

Regarding the belt length, the values are 3.65 m and 3.623 m for the hand calculations and MITCalc, respectively. In both Inventor and MDesign, the result is 3.602 m. This identical value comes from a standard, and the user was not able to modify it. While the hand calculations and MITCalc compute the belt length, Inventor and MDesign instead selected, based on input data, a matching value from standardised belt lengths.

Results from MDesign

The software gave several belt type suggestions with results for each and one. Results for 22/C (DIN 2215) were the chosen one for this study. MDesign could not perform force calculations for belt drive and the amount of possible input was limited. Note that the results from MDesign included some of the input parameters, with changed values. The last value, velocity Driven pulley in table 4.4 were an input value but got changed by the program.

The reason why it presents some of the input with changed value as results is that MDesign is very fixed to standards.

Note that the result for belt velocity differ from the results in the other software and hand calculations.

- Hand calculations: 19.635
- MITCalc: 19.63
- Inventor: 19.635
- MDesign: 17.59

According to Equation (2.29) can it be seen that the velocity of pulley is a factor in the equation for the velocity of the belt. Due to that the velocity of the driven pulley was one of the input values that got changed when MDesign computed the results, the belt velocity contained another value, in comparison to the other software that still calculated with 500 rpm (which was the input value).

One noticeable different between MDesign and the other software is the amount of results, especially results concerning forces. MDesign only compute a few results and the majority of these are whitin the field of geometry and dimensions. One of the reason to this could be the fact that MDesign, for belt drives, just computed lists of standard and values within these as a result.
The used standards and literature within MDesign were:

- DIN
  - 2211-1
  - 2215
  - 2217-1
  - 323-1
  - 323-2
- Roloff/Matek, 24. Auflage

4.2.3 Bearings

For bearings the calculations were mainly on bearing life of a singular bearing setup. Since most software use this type of calculation, this was done to simplify the comparisons between different software and Python-based calculations. In Inventor the possibility of calculating the bearing life adjusted according to SKF-standards was offered, so this was done to further enable a comparison between the software and the Python calculations.

Two spherical roller bearings with the correct dimensions were selected from the SKF rolling bearings catalogue (SKF Group, 2018). For the locating bearing, which needs to take both axial and radial forces, the bearing with number 22322 E was chosen. For the floating bearing, which only needs to take radial forces, the bearing 23122 CC/W33 was chosen.

The parameters then selected for further analysis were:

- P (equivalent dynamic bearing load)
- L_{10} (nominal bearing life)
- L_{nm} (SKF adjusted bearing life)

With the selected parameters for comparison calculations were preformed by hand with Python, and in MDesign, Inventor and MITCalc respectively for one bearing at a time. In Inventor a CAD-model of the shaft was made and the bearings were taken from the built in SKF catalogue that Inventor offers, the needed input data were entered and the calculations were preformed. In a similar manner SKF bearings were selected from built in catalogues in MDesign and MITCalc, the remaining input data was entered and an output of results was generated. The previoulsy chosen parameters for comparison was evaluated to find key similarities and differences and to analyse the software against eachother. The input, together with reasonable arbitrary values for κ and η_c used to determine a_{SKF}, for the calculation is presented in Table 4.5:

Table 4.5: Bearing data for life calculation

Parameter	**22322 E**	**23122 CC/W33**	**Unit**
Outer diameter	240	180	mm
Inner diameter	110	110	mm
Width	80	56	mm
Radial load F_r	45532.0	12253.1	N
Axial load F_a	5237.8	0	N
Dynamic load rating C	989000	450000	N
Load factor e	0.33	0.3	—
Factor Y_1	2.0	2.3	—
Factor Y_2	3.0	3.4	–
Fatigue load limit P_u	100000	61000	N
Speed	1440	1440	RPM
Reliability [%]	99	99	—
Viscosity ratio κ	2.5	2.5	—
Contamination factor η_c	0.5	0.5	—
SKF life modification factor a_{SKF}	0.3	10	—

Calculation and software results

This section presents the results from the bearing calculations. The numerical results are presented in Table 4.6 followed by descriptions of the software, the software are presented in the following order: Python, MDesign, Inventor and MITCalc.

Table 4.6: Results from hand calculations and software - Bearing life calculation

Variable	**Hand Caculations**	**MDesign**	**Inventor**	**MITCalc**	**Unit**
Bearing 22322 E					
P	56007.60	56008	56008.00	56006.60	N
L_{10}	14338.30	14338.30	14338.00	14335.64	Mrev
L_{10h}	165952.58	165952.60	165951.00	165962.00	h
L_{nm}	1068.20	–	899.00	–	Mrev
L_{nmh}	12363.38	–	10410.00	–	h
Bearing 23122 CC/W33					
P	12253.10	12253	12253.00	12253.10	N
L_{10}	164647.49	164647.50	164646.00	164093.15	Mrev
L_{10h}	1905642.27	1905642.30	1905619.00	1905642.00	h
L_{nm}	408871.86	–	342470.00	–	Mrev
L_{nmh}	4732312.22	–	3984602.00	–	h

Python Calculations

Calculations were preformed in python according to the theory presented in Section 2.5.3.1, the code in its entirety can be seen in Appendix I.

MDesign

MDesign provides two distinct types of calculations: design and verification. The design calculation produces an output in the form of a list of potential bearings capable of withstanding the specified loads. For both the locating and the floating bearing, one of the suggested bearings had already been selected for the earlier Python-based calculations. These same bearings were subsequently chosen for the verification calculation.

The verification process allows for the selection of a bearing from the previously generated list, with the corresponding bearing data automatically generated as input parameters.

Inventor

Autodesk Inventor, similar to MDesign, provides the functionality to select bearings from a predefined catalogue, automatically inserting the corresponding bearing data as input parameters. As a 3D CAD software, it required the creation of a shaft onto which the bearings could be mounted. This was done by modelling the shaft according to the defined shaft dimensions and placing 3D representations of the selected bearings from the built in SKF-catalogue onto it. The shaft with the mounted bearings can be seen in Figure 4.6. Calculations were then performed using Inventor's *Design Accelerator* feature.

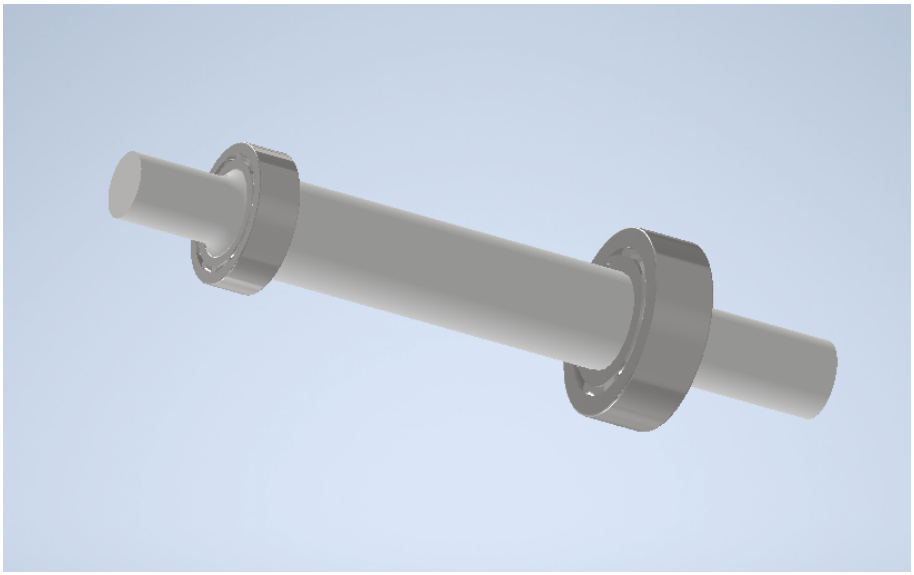


Figure 4.6: Shaft with bearings, Inventor

Furthermore, Inventor offers the option to apply the SKF adjusted life calculation method. This was used with the same input values for a_{SKF} and a_1 as in the Python calculation, thus supporting a more consistent comparison between the software tools.

MITCalc

Similar to the previously mentioned software tools, MITCalc provides the capability to select bearings from a predefined catalogue, which allowed for the selection of the same bearings used in the other calculations with the bearing data automatically entered as input.

Observations

The nominal bearing life results are nearly identical across all tools, confirming consistent implementation of the same ISO standard used in Maskinelement (Mägi et al., 2017). However, noticeable differences appear in the SKF-modified life values. This is likely due to the commercial software tools internally accounting for additional influencing factors, such as lubrication conditions and contamination levels, when determining the life modification factor a_{SKF}. In contrast, the Python calculations used manually defined values for these parameters, which may not capture the full range of inputs considered by the software.

MDesign includes functionality for calculating an adjusted bearing life based on factors such as oil viscosity. However, the method applied does not follow the SKF standard used in the Python calculations and in Autodesk Inventor, and was therefore excluded from the comparison. MITCalc does provide the option to perform SKF-adjusted life calculations, but this feature was not tested due to time constraints.

4.2.4 bolted joints

To be able to compare the results between the programs, a specific case with input was defined as in Table 4.7.

Table 4.7: Case definition for bolted joints

Geometry		**Unit**
Type of bolt:	M16	—
Thread type:	Coarse	—
Clamped part length	27	mm
Washer (Yes/No):	No	—
Connection type:	Blind connection	—
Loads		
Static load (Yes/No)	Yes	—
Axial load	4984.4	N
Radial load	0	N
Materials		
Material name:	EN C45E	—
Bolt Young's Modulus	207000	MPa
Clamped part Young's Modulus	207000	MPa
Friction factors		
Thread friction factor:	0.18	—
Seating head bearing area friction factor:	0.18	—
Plastic deformation due to settlement	6.1	µm

Note: The bolt is an ISO 4014 M16 × 65 mm bolt.

Seeking:

- Preload force without settling
- Preload force with settling
- Tightening moment
- Stiffness in bolt
- Stiffness in clamped part

Table 4.8 below summarizes the results obtained from the hand calculation, as well as those generated by MITCalc, Inventor and MDesign.

Hand Calculation

The hand calculation was performed in a python code based on the method used theory section (see Section 2.5.4 *Bolted joints*). The python code is inserted in the appendix (see Appendix J). The results obtained from this method were used as reference for the programs results.

Table 4.8: Results from hand calculations and software - bolted joint

Variable	Hand Calc.	MITCalc	Inventor	MDesign	Unit
Preload force without settling ($F_{0,I}$)	4347	9378	3747	5594	N
Preload force with settling ($F_{0,II}$)	10187	12965	4744	8391	N
Required tightening moment with settling (M_t)	36.99	45.37	18.51	21.26	Nm
Stiffness in bolt (c_s)	1152536	667000	882817	610000	N/mm
Stiffness in clamped part (c_k)	5654131	4956000	2674037	736000	N/mm

Results from MITCalc

For bolt calculations, MITCalc offers two different approaches: *Method of Pressure Cones* and *Method of Equivalent Tube*. In order to be able to use the hand calculation as a reference, the *Method of Equivalent Tube* was selected. After that, the user can choose which method to implement for calculating the equivalent tube length. The three available methods are Rötscher, VDI 2230 (Verein Deutscher Ingenieure), and Fritscher. Since the hand calculation uses VDI 2230, this method was also selected in MITCalc.

After entering the input data from the defined case, MITCalc proposed various bolts capable of withstanding the applied loads. In this case, an ISO M16 bolt with the corresponding dimensions, presented in Table 4.7 *Bolt case definition* was used. With these parameters entered, MITCalc produced the results presented in Table 4.8.

The MITCalc values in the table differ significantly from the reference calculation. Unfortunately, due to a lack of transparency in MITCalc, no in-depth technical analysis could be performed, as the equations used by the software were never found.

Results from Inventor

In Inventor, the user first creates an assembly, modeling the clamped part, and then selects a point to serve as the center of the hole where the bolt is to be placed. The user then selects a bolt to be analysed. Here the M16 bolt presented in the case definition was selected as well. The bolt is then inserted as a new part in the assembly. Once the bolt had been inserted, calculations could be carried out. This is done by entering loads, friction values, safety factors and material properties, after which the software performs the calculations. Inventor provides the output in the form of tables. These results are presented in Table 4.8.

Inventor does not reference its equations to any specific standard. However, explanations of all variables and equations are provided in their help page (Autodesk, 2025), enabling the user to analyse the calculations in greater depth. Based on this information, the calculation was first performed manually (see Appendix K) and subsequently implemented in Python (see Appendix L).

The variation in preload between the hand calculation and Inventor methods is mainly due to the different ways the bolt and clamped part stiffness are calculated. In the hand calculation, the bolt stiffness c_s is determined using the axial stiffness formula as presented in Equation (2.58) from Section 2.5.4.1 *Calculation*:

$$c_s = \frac{E \cdot A_s}{L}$$

where A_s is the tensile stress area based on the core diameter of the bolt. This approach assumes a direct axial load and uniform material behavior. In contrast, Inventor estimates bolt stiffness using an extended effective length and correction factors, and then inverts this value to obtain stiffness as can be seen in the equation below. The equation below is taken from equation A in Appendix K:

$$c_1 = \left(\frac{(L + 0.8d) \cdot 4}{E \cdot \pi d_s^2} \right)^{-1}$$

where the increased effective length and the factor of four account for additional deformation in the threaded portion of the bolt. This results in a lower bolt stiffness.

The clamped part stiffness in the hand calculation is computed using a detailed geometric representation of the contact region A_{ekv}:

$$c_k = \frac{E \cdot A_{\text{ekv}}}{L}$$

where

$$A_{ekv} = \frac{\pi}{4}(d_w^2 - d_h^2) + \frac{\pi}{8} L_k d_w ((x+1)^2 - 1)$$

Inventor however, uses a conical expansion model and calculates the equivalent area as in the equation below, taking the material stiffness into account using a constant a, which Inventor recommend to set as 10 for steel. This equation is taken from equation B in Appendix K:

$$A_{\text{eq}} = \frac{\pi}{4} \left(\left(1.5d + \frac{L}{a} \right)^2 - 1.05d^2 \right)$$

which is then used to calculate stiffness with the following formula:

$$c_2 = \left(\frac{L}{E \cdot A_{\text{eq}}} \right)^{-1}$$

This approach accounts for the spread of load through the clamped material but simplifies the geometry, resulting in a lower stiffness value. The Inventor method, which relies on empirical formulas and geometric assumptions, results in lower calculated stiffnesses than those produced by the more idealized hand calculation approach.

Regarding the preload with settlement ($F_{0,II}$), Inventor deviated significantly from the hand calculation. This is partly due to the difference in stiffness values, but also because Inventor does not calculate the exact material deformation δ_{pl} as in Equation (2.68) in section 2.5.4.1 *Calculation*. Instead it applies a safety factor ψ, which Inventor recommends to set as 1.2 (see Appendix K and Inventor's help page (Autodesk, 2025)).

The tightening moment differs significantly between the hand calculation and the Inventor method, primarily due to differences in the preload force and the torque model used. In the hand calculation, the tightening moment is estimated using the following formula:

$$M_t = F_0 \left(\frac{P}{2\pi} + \mu \left(0.58 d_2 + r_m \right) \right)$$

This formula assumes constant friction and approximates the thread helix and under-head friction contributions using coefficients. In contrast, Inventor uses a more detailed formula that considers the thread geometry and friction:

$$M_t = F_0 \left(\frac{d_2}{2} \cdot \tan \left(\frac{p}{\pi d_2} + \frac{f_1}{\cos(30^\circ)} \right) + 0.7 \cdot d \cdot f_2 \right)$$

Here, the term involving the tangent accounts for the combined effect of the thread helix angle and the thread friction coefficient f_1, while the under-head friction is considered separately with a factor of 0.7 and a friction coefficient f_2.

Despite the more refined torque model, the Inventor method yields a significantly lower tightening moment. This is primarily due to the lower preload force computed in Inventor, which directly affects the tightening moment calculation. The hand calculation applies a higher preload, especially when including settling, and thereby results in a substantially greater tightening moment.

4.2.4.1 Results from MDesign

For bolts, MDesign functioned in approximately the same manner as MITCalc, as both are purely calculation-based programs, in contrast to Inventor, which is a CAD program. The user interface was similar to that of MITCalc, with raw input and output data, but lacked visualizations, making the software less intuitive to work with. Unlike MITCalc, only one method was available in MDesign, namely VDI 2230. Once all input data had been entered and the calculation was performed. The resulting values can be seen in Table 4.8.

These values also differ significantly from the reference calculation, but as with MITCalc, the transparency of the calculation process was insufficient. The equations used by the software to generate the results could not be found, making it impossible to analyse the calculation.

4.2.5 Bevel gears

The objective was to design a bevel gear pair for a stone crusher and to evaluate the resulting force distribution and safety factors. The calculations were based on a predefined set of requirements, summarised in Table 4.9.

Table 4.9: Bevel gear design requirements

Parameter	Value	Unit
Output speed	320	RPM
Target gear ratio	4.5	—
Driving power	145	kW
Maximum wheel size	868.81	mm
Material (wheel)	Carbon steel	—
Operation	Stone crusher	—
Shaft orientation	Perpendicular	—

These requirements were used to determine the gear geometry, ensuring the bevel gears were dimensioned to meet the operational demands. The calculations were first done by hand and then with the selected software. The geometry of both the pinion and the wheel was calculated, forming the basis for subsequent calculations of force distribution and safety factors, and these calculations were preformed as outlined in Section 2.5.5. This was done by implement the equations in a python code, which can be seen in Appendix N. For the safety calculations SS 1862 standard was selected for its similarity to ISO 10300 in methodology, while offering a more streamlined approach suited to the scope of this thesis.

While bevel gear design and machine element design in general typically involve iterative refinement, this study was limited to a single design iteration. Consequently, no further adjustments were made to optimise the design if it was found to be over- or under-dimensioned. This did however not affect the comparative aspect of the study as the resulting configuration was assessed for compliance with performance and safety requirements.

The results of the hand calculations are presented in a comprehensive table in Appendix M, alongside the corresponding results from the selected software tools. Since the software required additional input parameters beyond those listed in the initial requirements (Table 4.9), the remaining input values were taken from the hand calculation results. A complete overview of all input parameters is provided in Table 4.10.

Table 4.10: Input for bevel gear calculations in programs

Description	Parameter	MITCalc	MDesign	Inventor	Unit
Gear ratio	u	4.5	–	4.5	–
Angle of shaft axes	$\sum$	90	90	90	deg
Number of teeth (pinion)	z_1	18	18	18	–
Number of teeth (wheel)	z_2	–	81	–	–
Outer pitch diameter (wheel)	d_{e2}	868.81	868.81	–	mm
Face width (gear)	b	107.26	107.26	107.26	mm
Mean spiral angle	β_m	29	29	29	deg
Nominal pressure angle	α	20	20	–	deg
Tangential pressure angle	α_t	–	–	22.59	deg
Transferred power	P	145000	145000	145000	W
Transferred torque	T	–	961.56	–	Nm
Pinion speed	n_1	1440	1440	1440	rpm
Mean normal module	m_{mn}	8.25	–	–	mm
Outer transverse module	m_{et2}	10.73	–	10.73	mm
Addendum factor (pinion)	k_{hap}	1.00	1.00	–	–
Dedendum factor (pinion)	k_{hfp}	1.25	1.25	–	–
Profile shift coefficient	x_{hm1}	0.5514	0.551	0.551	–
Thickness mod. coeff.	x_{smn}	0.0889	0.0889	–	–
Thickness mod. coeff. (theoretical)	x_{sm1}	–	–	0.0742	–
Cutter radius	r_{c0}	170	170	–	mm
Tooth form factor	j_{en}	0.41	0.41	–	–
Addendum angle (wheel)	θ_{a2}	0.844	0.844	–	deg
Dedendum angle (wheel)	θ_{f2}	2.175	2.175	–	deg
Grinding allowance	q	0.2867	0.2867	–	–
Cutter edge ratio	f_{aP0}	0.25	0.25	–	–
Application factor	K_1	–	1.75	1.75	–
Required life time	L_h	–	20000	–	h
Manufacturing quality grade	q	–	10	–	–
Mounting factor	$K_{HV\text{-}be}$	–	1.32	–	–
Protuberance on cutter	p_r	-	0.5133	–	–

Material properties and load factors were selected according to guidelines in ISO23509 and SS1862. Case-hardened steel with a hardness of 55 HRC was used for both the pinion and the wheel.

As mentioned earlier, the complete set of results from the hand calculations, along with those from the selected software tools, is presented in Appendix M. Additionally, specific results for force distribution and safety factors are shown in Table 4.11 and Table 4.12, respectively.

Table 4.11: Results for the Force and Torque Calculations from Hand Calculation, MITCalc, Mdesign, and Inventor

Variable	Hand calc.	MITCalc	Mdesign	Inventor	Unit
F_{ax1}	7150.9	7151.5	7150.9	7150.9	N
F_{ax2}	3239.1	3239.3	3239.1	3239.1	N
F_{mt1}	11325.8	11326.6	11325.8	11325.8	N
F_{mt2}	11325.8	11326.6	11325.8	11325.8	N
F_{rad1}	3239.1	3239.3	3239.1	3239.1	N
F_{rad2}	7150.9	7151.5	7150.9	7150.9	N
T_1	961.6	961.6	961.6	961.6	Nm

Table 4.12: Results for the Safety Factors from Hand Calculation, MITCalc, Mdesign, and Inventor

Variable	Hand calc.	MITCalc	Mdesign	Inventor
S_H	1.7795	–	1.549	2.993
S_F	4.0837	–	2.147	1.362

The following sections provide analysis of the results obtained from the software-based bevel gear design calculations. These results are organised according to the sequence of the design process: geometric calculations, force distribution, and safety factor assessments.

The analysis follows the order of MITCalc, MDesign, and Inventor. A comparison is made between the hand calculations and the results generated by the CAD software tools to evaluate consistency and identify any deviations. Emphasis is placed on how closely the results align with the relevant standards, ISO 23509 and SS 1862. The aim is to assess whether the software outputs are consistent with the hand-calculated values and verify the reliability of the selected tools.

MITCalc

The bevel gear geometry was calculated using the software tool MITCalc, which allows the user to input fixed design parameters and, where applicable, suggests recommended values, but user-defined inputs can override the suggested defaults where needed. The recommended values were not considered in this study as the aim was to have as similar as possible input to both the hand and CAD calculations. The software does not allow complete freedom in setting all input values, as some are constrained by the tool's internal logic. As an example the number of teeth for the pinion and the gear ratio are required and there is no flexibility to input the number of teeth for both the pinion and the wheel.

The bevel gear module in MITCalc is structured in two main stages for Method 0, non-hypoid gears. First, a *Preliminary / Approximate Design of Geometrical Parameters* is performed based on the initial input data. Once sufficient data is provided, the tool proceeds to the second stage, generating the final geometrical output. The software calculates key parameters for tooth profiling and force distribution but does not include safety factor calculations related to pitting or tooth root breakage.

Geometry calculations Based on these parameters, MITCalc generated a complete geometrical design. The full list of output values, compared to the hand calculations, is included in Appendix M. It can be concluded that the values for the geometrical calculations in MITCalc correspond well with the hand calculations according to ISO 23509.

Force distribution calculations As mentioned earlier MITCalc also does the force distribution calculation. There is an option to choose which way the spiral angle are oriented, like in Figure 2.13 but the spiral angle is changed instead of rotation direction. For these calculations the *Left-hand method* from MITCalc was chosen, e.i. the gears push each other apart. This resulted in the results displayed in Tabel 4.11

Some irregularities can be identified, such as the tangential force on pinion, F_{mt1}. For the hand calculations, the force is 11325.8 N, while for MITCalc it is 11326.6 N, which is a small discrepancy for a simple calculation (Equation (2.147)). This is most likely due to rounding when converting from revolutions per second to radians per second. It is a small error, but it also affects the rest of the force calculations slightly.

Conclusion for MITCalc on bevel gears Overall, it can be said that in this test, MITCalc performed well when compared to the standard based hand calculations for calculating the geometry and forces according to ISO 23509. It is as expected, as MITCalc also refers to this standard in their help section (MITCalc, 2024). It did make some small rounding errors, though, which is worth noting. And like mentioned earlier, it does not calculate for safety factors.

MDesign

MDesign was used in this study to generate a complete geometric design and perform safety calculations for the bevel gear pair. The software outputs a detailed report including all input and output values, as well as references to the standards and calculation methods applied.

In MDesign, it is possible to select between performing calculations for geometry only, geometry and safety factors, or a full analysis including tooth flank fracture. For this study, the geometry and safety factor calculation mode was chosen to allow for comparison with manual calculations.

For bevel gear analysis, MDesign references the ISO 23509 and ISO 10300 standards. The input required for MITCalc calculations is presented in Table 4.10. Based on the defined input, MDesign generated a complete geometrical and safety analysis. A full list of output values and their comparison to hand calculations can be found in Appendix M.

Geometry calculations As seen from the results, it can be concluded that the values for the geometrical calculations in MDesign correspond well with the hand calculations done according to ISO 23509. MDesign did however not calculate for the *pitch cone apex to crown* (t_{xo}), *front crown to crossing point* (t_{xi}) or pinion whole depth (h_{t1}) as seen in Appendix M.

Force distribution calculations MDesign also performs the force distribution calculation, but there is no option to choose if the gears are pushing or pulling each other apart, as there was in MITCalc. MDesign calculates with the push configuration, and the results from the calculations can be seen in Table 4.11. The values seem similar, which indicates that the implementation of these calculations inside MDesign has been done correctly.

Safety factor calculations The results from the safety factor calculations in MDesign are shown in Table 4.12, alongside the results from the hand calculations. While the values are not identical, the safety factor for pitting, S_H, is similar for both methods and falls within the same range. However, the safety factor for tooth breakage, S_F, is significantly higher in the hand calculations.

MDesign refers to the ISO 10300 standard for its calculations, which provides a robust and widely accepted framework for stress analysis when implemented appropriately. In contrast, the application of ISO 10300 in the hand calculations was outside the scope of this study. Therefore, the older and more general Swedish standard SS 1862 was used instead. This difference in methodological approach likely accounts for the observed discrepancies in the safety factor results. Additionally, MDesign applies more multiplication factors for stress calculations while for hand calculations a lot of them were set to unity. For example, in the hand calculations, the lubrication factor was assumed to be unity, whereas in MDesign, it was mandatory to specify a particular oil, potentially leading to more different values for the resulting safety factors. With this in mind the results are fairly similar.

Conclusion for MDesign on bevel gears Overall MDesign perfomed well when compared to the standard based hand calculations. This together with its transparency citing the used standards makes it a capable software for bevel gear calculations.

Inventor

The gear calculations in Autodesk Inventor were performed using the *Component Generator* feature. For this study, the *size type* was set to **module** instead of diametral pitch, as the latter uses imperial units, which were not applicable in this metric-based analysis.

The *gear ratio* was selected as the defining input rather than directly inputting the number of teeth on the wheel. This approach allowed the program to automatically determine the number of wheel teeth based on the specified gear ratio and the number of teeth on the pinion.

Load data could be entered in three different ways, since speed, power, and torque are interdependent. For this study, the most relevant known quantities were selected accordingly.

In this study, *the check calculation* mode was chosen, as it suited the goal of evaluating the performance of already roughly dimensioned gear design by completing the geometry and making safety calculations on the created design. Unless necessary, *load factors* were left at their default values.

Geometry calculations Based on the defined input, Inventor generated a geometrical and safety analysis. A full list of output values and their comparison to hand calculations can be found in Appendix M. It can be concluded that the values for the geometrical calculations in Inventor do not correspond well with the hand calculations according to ISO 23509. This is most likely due to it being an American software company. Even thought they refer to standards like ISO6336:1996 for the safety factor calculation the geometry does not get even closely match the values of ISO 23509 method. Inventor also gives far fewer parameters for the geometry compared to ISO 23509.

On the other hand when the geometry has been calculated a 3D CAD model is generated, see Figure 4.7.

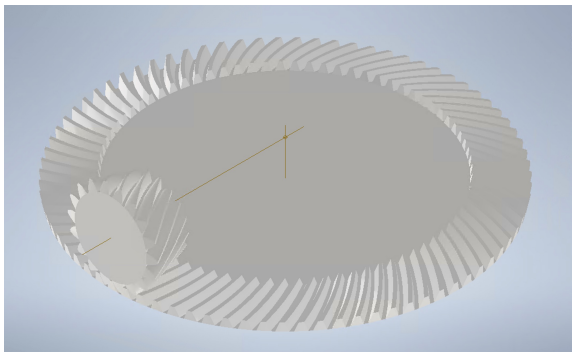


Figure 4.7: Generated spiral bevel gear from Inventor

Force distribution calculations Inventor also performs the force distribution calculations, but there is no option to choose if the gears are pushing or pulling each other apart, as there was in MITCalc. Inventor calculates with the push configuration, and the results from the calculations can be seen in Table 4.11.

The force values calculated by Autodesk Inventor closely align with those derived from ISO 23509. This suggests that the force distribution calculations in Inventor are reasonably accurate. This is likely due to the nature of force distribution as it relies on trigonometrical calculations base on e.g. the spiral and torque which are inputs to the calculations.

Safety Factor Calculations Autodesk Inventor provides several options for selecting the standard or methodology used in gear strength calculations:

- **ANSI/AGMA 2001-D04:2005** – A widely adopted American standard for evaluating bending and pitting strength in spur and helical gears.
- **Legacy ANSI** – Earlier American standards that formed the foundation for modern AGMA methods.
- **Bach (simple design)** – A simplified German design method primarily used for preliminary or educational applications.
- **Merritt (complex design)** – A comprehensive approach incorporating load distribution and dynamic effects, based on the work of Joseph Merritt.
- **ISO 6336:1996** – An internationally recognized standard for calculating gear load capacity, widely used across industry sectors.
- **DIN 3991:1988** – A German standard for evaluating the strength of cylindrical gears, influential in the development of ISO 6336.
- **ČSN 01 4686:1988** – A national Czech standard for gear strength, similar in structure to DIN-based methodologies.

For this study, **ISO 6336:1996** was selected within Inventor as the calculation basis. This is a standard for spur and helical gears which is not the same as bevel gear. The resulting safety factors are compared with the hand calculation results in Table 4.12.

As shown, the results differ significantly: Inventor predicts a notably higher safety factor for pitting (S_H) and a much lower value for tooth breakage (S_F). These discrepancies are likely due to differences in the calculation methods, including how geometry and load factors are treated, as well as the differing assumptions inherent in the standards used. For instance, ISO 6336 is not a bevel gear specific standard likely causing discrepancies. If Inventor were to use AGMA standard behind the users back, it would partly explain the root safety factor discrepancies as this standard has been found to be on the conservative side (Solutions, 2022), but not the discrepancies for pitting safety factors.

Conclusion for Inventor on bevel gears Overall, Autodesk Inventor's results show noticeable deviations from the methodologies and outcomes of ISO 23509 and SS 1862, and are unlikely to be fully consistent with ISO 10300 either. As the calculated geometry differs a lot from the geometry derived by hand this most likely causes the safety calculations to deviate as well making it hard to make any conclusion on the reliability of those.

5
Discussion

Modern CAD-integrated tools enable iterative approaches to the design of machine elements, which enhances both result verification and efficiency. Features such as multi-pass adaptive analysis allow for repeated recalculations and refinements, ultimately leading to more optimized design proposals compared to purely manual calculations.

5.1 MITcalc

MITCalc demonstrates strong computational capabilities, offering a wide range of predefined calculation modules for machine elements and more. These modules are beneficial for providing quick estimations during early design phases.

The first notable advantage of the software is that it allows the user to perform both advanced, in-depth analyses and calculations, as well as simpler and more basic computations depending on the needs of the task. The software offers the possibility to choose the amount of input data depending on the desired amount and detail in outputs and calculations. In general, more advanced computations that provide a broader range of output parameters require a greater amount of input data. This is advantageous from an educational perspective, particularly in terms of education across various levels. In other words, MITCalc can be used in teaching at multiple levels, depending on the depth and difficulty required.

MITCalc enables the opportunity for a user to make some changes regarding results from calculations. For example, when working with belt drives. Even though MITCalc gives a design proposal, based on standards, the user got a possibility to make some changes.

For certain parameters within these standards, the user has the option to select values within a specified range; this is also the case when working within MITCalc. This is not always the case in the other software. Regarding the values for the belt length, see Section 4.2.2, and the fact that the values were the same in Inventor and MDesign. This is, as stated, due to that it's taken from a standard and the user was not able to change it in the program, even tough the standards got an interval of possible, well-adapted values. This could limit the freedom of design for a user and the potential for innovative and well-adapted solutions. On the other hand, even though a standard got a wide-range of possible values, just some may fit the purpose (given input). The programs that just gives a fixed value based on the standards, has probably done this by an iterative procedure checking which value fits the best, if this is the case, this could help a design process to be more efficient. If this is the case, the whole "*Detailed design*"-part within the product development process (see Figure 2.1) could be more efficient, which could be beneficial for a company in a competitive and cost-effective way.

One more notable thing within MITCalc is regarding results verification. By checking the results in MITCalc in relation to the results from hand calculations and the others software can it be seen that the results seems to be relevant and valid. On the other hand can it be noticed that it even though got some drawbacks. Regarding preload it can be inferred (see *Preload* under Section 2.5.2.1) that a preload can be assured in six different ways. The hand calculations computed values for two of these ways. The results were 4883.55 N for preload with a given axial load and 3212 N with a given axial distance. MITCalc only computet one value for preload (= 4704.06 N) and did not clarify how this was computed. The result seems to be close to the calculated pre load with a given axial load, but this can only be confirmed if a user has made these calculations beforehand, which than undermines the whole purpose of using such a calculation tool as MITCalc. From a constructor perspective could it be essential to understand how this calculation were made and based on, in order to make a decision of how a preload should be executed.

However, despite its computational power, MITCalc presents certain challenges in terms of usability and transparency. One noticeable drawback is the limited availability of documentation and clear instructional resources, which can create a barrier for users, especially new ones. Even though the software got some clear figures to facilitate and enhance understanding. The software could be improved by providing clearer explanations of the meaning and purpose of each parameter. And the visualization tools could be clearer. These two points are particularly highlighted in comparison to the other software.

Additionally, the workflow within the software can be difficult to follow, particularly when trying to trace the logic behind the step-by-step calculations. This may reduce transparency and hinder understanding of how the final values are obtained. These two drawbacks could, in turn, may pose difficulties in educational settings. Although MITCalc offers an all-encompassing help section, both on the website and within each subsection in the program, the information provided is limited in depth.

Regarding the usage of standards, MITCalc combines different standards, both national and group standards. For each and one of the machine elements, MITCalc does just lists all the used standards without any explanation regarding when and where they are used, this adversely affects the transparency. In calculations that rely on standardized values, it is evident, based on what we have investigated, that MITCalc performs the computations using appropriate and valid data from relevant standards. However, the process lacks transparency; the software does not clearly indicate how these values are derived or from which specific sources within the standards they are obtained.

The lack of transparency is not necessarily a disadvantage. Using the program requires some fundamental knowledge. Which, from an educational perspective require, or even encourage students to learn what is necessary. If that is the case, MITCalc can serve as a valuable complement to education.

5.2 Inventor

Inventor supports calculation methods for a wide range of machine elements, as described in section 4.1.2 *Inventor*. Since Inventor has 3D-capabilities, the component that is to be analysed is or can be inserted as a 3D-part in an assembly. This can be a great advantage for educational purposes. It provides the user with a clear view of what is being calculated, in contrast to MITCalc and MDesign, which function purely as numerical calculators with input and output values.

Having the capability to analyse multiple components in the same assembly is also something unique compared to MITCalc and MDesign. This enables the user to create wide systems of different machine elements, which can be particularly useful when designing larger systems.

On the other hand, the fact that Inventor is a 3D-CAD program could also be a drawback since it requires some knowledge of 3D modelling and its typical user interface. MITCalc and MDesign have simpler user-interfaces with raw inputs and outputs, which can be beneficial for user friendliness and also educational purposes since it requires little previous knowledge of similar user interfaces.

Another advantage in Inventor is that the calculation methods that it uses are easily accessible and explained on Inventor's help page, which indicates a high level of transparency and makes the software highly intuitive and user-friendly. This also enhances the software's suitability for use in educational contexts.

However, the calculation methods are not referenced to specific standards. Instead, they are linked to what Inventor refers to as the *Engineer's Handbook*, which appears to be an in-house repository of calculation methods and engineering knowledge developed by Inventor. Although some of these methods may be based on established standards, it is difficult, if not impossible, to determine exactly which ones. Furthermore, there is no guarantee that the methods are standard-based at all, which makes it challenging to verify their validity. This reduces the overall transparency of the software, which negatively affects its suitability for educational use.

An exception for this was the SKF calculation method for the nominal bearing life of spherical roller bearings, which was specifically referenced in the bearing calculation. For more information, see section 4.2.3 *Bearings*.

5.3 MDesign

MDesign is a calculation-oriented tool, structured primarily around standardized input and output procedures. It is particularly aimed at users requiring compliance with established norms, such as VDI 2230 for bolt connections, ISO 10300 for bevel gears, or ISO 281 for rolling bearings. The software simplifies the design process by offering predefined templates and suggesting components based on accepted standards, while performing the necessary calculations.

One of MDesign's notable strengths is its structured approach to standard integration. The software clearly references applicable standards in both the input and output stages, allowing users to trace where and how these standards are applied.

Compared to MITCalc, MDesign provides a more systematic link between design parameters and normative documents. In the context of this project, it proved useful not only as a calculation tool but also as a means of identifying applicable standards based on the given requirements. Although MDesign does not disclose its internal equations, it references the engineering standards that serve as the basis for its calculations. This enables users to trace the methodologies by consulting the cited documents, such as those published by ISO or DIN. While this does not offer full computational transparency, it provides a degree of traceability that supports verification and builds confidence in the results. This feature is particularly valuable in regulated industries where compliance with standards is critical. Despite its limitations, MDesign has demonstrated reliability in producing standard-compliant results efficiently. Its integration of standards and structured workflow makes it a practical tool for preliminary design phases and contexts where adherence to established norms is essential.

In addition to its practical applications, MDesign could also hold a value in an educational setting. By consistently linking calculations to established engineering standards, it helps students understand the role and importance of standardized methods in mechanical design. The structured workflow and clear references to norms encourage good engineering practices and reinforce the necessity of adhering to documented guidelines, which is especially beneficial for those learning design methodologies and regulatory frameworks.

A notable limitation of MDesign lies in its restrictive flexibility. Unlike MITCalc, which permits manual adjustment of a broad range of parameters, MDesign may lock certain values after an initial calculation. For example, in belt drive calculations, the software automatically altered the driven pulley's speed, resulting in a belt velocity that differed from both the user-defined input and the manual calculations. This behavior reduces user control and may restrict the ability to tailor designs to specific needs. However, this limitation is not consistent across all modules. For instance, in bearing calculations the software allows greater flexibility to adjust parameters and explore alternative results. In terms of output, MDesign provides limited insight into intermediate steps.

Results are primarily displayed as final component selections and calculated values, with little visibility into the equations or methods used. While this issue is also present in MITCalc, MDesign tends to offer even fewer details, particularly in cases involving complex or dynamic loading. Although the relevant standards used in the calculations are clearly listed, the lack of direct access to the equations reduces overall transparency.

5.4 Advantages and disadvantages

This section provides a summary of the advantages and disadvantages of each software, presented in table 5.1.

Table 5.1: Advantages and disadvantages of each software

Tool	Advantages	Disadvantages
MITCalc	- Offers a large library - Follows standards	- Lack of transparency
Inventor	- 3D Visualization - Directly linked to geometry - Detailed help section	- No clear reference to standards - Requires knowledge of 3D-CAD software user interface
MDesign	- Follows established engineering standards - Solid and comprehensive technical documentation	- Requires very precise input - Does not display exact calculations or intermediate steps, reducing transparency.

6
Conclusion

This study found that traditional hand calculations and modern CAD based methods generally converge on similar outcomes for machine element design, yet they differ in process and transparency. Manual theoretical calculations provide a clear, step-by-step insight into the design, serving as a transparent baseline. The CAD tools, when supplied with the same input parameters, produced results largely in line with these classical calculations, confirming that modern software can adhere to established engineering principles. However, some discrepancies emerged in specific output values due to differences in built-in assumptions and calculation approaches. These variations underscore that while CAD software can match the precision of theoretical methods in most cases, the reasoning behind its results can sometimes be unclear. This highlights the importance of an engineer's understanding of the underlying theory.

Each of the evaluated software tools (MITCalc, MDesign, and Autodesk Inventor) demonstrated unique strengths and practical applicability. MITCalc and MDesign are dedicated calculation programs that closely follow recognized standards and yield output very close to the manually calculated values. Their formula-based approach ensures rigorous compliance with design guidelines, although for more advanced output data they need more in-data, and some intermediate steps remain hidden, making certain aspects of their calculations hard to understand. Autodesk Inventor's design accelerator, in contrast, offers an integrated and user-friendly 3D environment that streamlines the design process. It allows quick application of calculations directly on CAD models, making iterative design more convenient. However, this convenience comes at a cost: the calculation process in Inventor is less transparent. The software handles much of the computation internally, giving engineers only limited visibility into the formulas or standards being applied. All three tools can significantly speed up the design process and reliably handle complex calculations. Even so, it remains essential to verify their outputs. Engineers should always cross-check critical results (either through manual calculations or with a second software tool), especially whenever an outcome seems unexpected. This practice helps catch any inconsistencies due to different default parameters, interpretations of standards, or potential input errors before a design is finalized.

Based on these findings, a balanced approach is recommended for both education and professional practice. Educational institutions should integrate modern CAD tools into their curriculum to familiarize students with industry relevant software, while simultaneously reinforcing fundamental hand calculation techniques. By working with both approaches, students can appreciate the speed and convenience of CAD software without losing sight of core engineering principles. For example, instructors might use software generated results as a starting point and then ask students to verify those outcomes through classical theory. In industry, engineers can likewise leverage these tools to improve efficiency in everyday design tasks. However, it is crucial that companies also foster a culture of thorough understanding and verification. This means providing training to reinforce fundamental knowledge and encouraging practices such as independent verification of software results or peer reviews of critical calculations.

In summary, CAD software should be seen as a powerful complement to, rather than a replacement for traditional calculation methods. When used wisely, it enables faster and more efficient design workflows while still upholding the rigor and transparency that are the hallmarks of sound engineering practice.

6.1 Further research

While this study provides a detailed comparison between theoretical frameworks and CAD based methods for several common machine elements, there are multiple avenues for further research that could deepen and broaden its findings. One promising direction would be to expand the analysis to additional components such as springs, couplings, chain drives, and welded joints. This would help determine whether observed patterns, such as differences in transparency, flexibility, and standard enforcement, persist in a wider variety of design scenarios. In addition, evaluating complete mechanical assemblies rather than single elements could shed light on how compound design complexity affects the reliability of the software and interpretation of the results.

Another important area for future research involves the growing integration of AI-driven and cloud-based CAD platforms, such as Autodesk Fusion 360 with generative design or Onshape. These tools promise faster and more optimized solutions, but often sacrifice user control and calculation traceability. A comparative study of these newer platforms, particularly in terms of transparency, decision logic, and integration with engineering standards, would be highly relevant as the field evolves.

Conducting a comparative study that includes the CAD software with FEA capabilities, overlooked in this study, would be an interesting extension. How much do the results from FEA-based simulations differ from those obtained using standard-based calculations? Are some machine elements more suited for FEA than others? Another area that could be researched more is to look even deeper into the software already analyzed, since there could be features or options in the programs that could have been missed during the course of this project.

Bibliography

Ahmed, E., & Rahman, M. Impact of software usage on fundamental engineering courses [Cited by: 0]. In: Cited by: 0. 2016. https://doi.org/10.1145/2910925.2910942

Alfredsson, B. (Ed.). (2014). *Handbok och formelsamling i hållfasthetslära* (11, omarbetade upplagan) [Andra tryckningen, 2016]. Institutionen för hållfasthetslära, KTH.

Autodesk. (2023). *Inventor 2023 help - fasteners and calculations* [Accessed: 2025-05-13]. https://help.autodesk.com/view/INVNTOR/2023/ENU/?guid=GUID-24104648-BE41-49C7-9DD0-1AF2BAFC102E

Autodesk. (2025). *Inventor 2025 help - fasteners and calculations* [Accessed: 2025-05-13]. https://help.autodesk.com/view/INVNTOR/2025/ENU/?guid=GUID-C92EEE54-5A24-436B-B809-88EB1A7737B1

Coenders, J. L. (2021). Next generation parametric design [Cited by: 5]. *Journal of the International Association for Shell and Spatial Structures*, *62*(2), 153–166. https://doi.org/10.20898/j.iass.2021.018

Cooper, R. G., & Brem, A. M. (2024). Insights for managers about ai adoption in new product development [Cited by: 2]. *Research Technology Management*, *67*(6), 39–46. https://doi.org/10.1080/08956308.2024.2418734

Deutsches Institut für Normung. (2012, December). DIN 743-1:2012-12 – Calculation of load capacity of shafts and axles – Part 1: General [Accessed 2025-05-12].

Engelmann, F., Grote, K.-H., & Guthmann, T. (2021). Machine elements [Cited by: 0]. *Springer Handbooks*, 503–628. https://doi.org/10.1007/978-3-030-47035-7_15

Findling Wälzlager GmbH. (2022). Thrust bearings – product overview [Accessed: 2025-06-02]. https://www.findling.com/en/products/product-overview/thrust-bearings

för Standarder (SIS), S. I. (2025). Vad är en standard? [Accessed: 2025-04-01]. https://www.sis.se/standarder/vad-ar-en-standard/

International Organization for Standardization. (2016, November). ISO 23509:2016, Bevel and hypoid gear geometry (2nd ed.) [Accessed 2025-04-01].

International Organization for Standardization. (2023, August). ISO 10300-1:2023, Calculation of load capacity of bevel gears – Part 1: Factors (3rd ed.) [Accessed 2025-04-01].

Jiang, W. (2019). *Analysis and design of machine elements*. Wiley Online Library.

Khanal, R. (2023, November). V belt and pulley assembly | belt and pulley mechanism [Accessed: 2025-06-05].

KHK Gears. (2024). Gear forces – technical reference [Accessed: 2025-05-08].

Lantada, A. D., & Morgado, P. L. (2013). Computer-aided design (cad) technologies for biodevices. In *Handbook on advanced design and manufacturing technologies for biomedical devices* (pp. 59–74). Springer.

Mägi, M., Melkersson, K., & Evertsson, M. (2017). *Maskinelement*. Studentlitteratur.

Metwalli, S. M. (2021). *Machine design with cad and optimization*. John Wiley & Sons.

MITCalc. (2024). *Bevel gear calculation* [Accessed: 2025-05-07]. https://www.mitcalc.com/doc/gear2/help/en/gear2.htm

MITCalc Team. (2025). Mitcalc - mechanical, industrial and technical calculations [Accessed: 2025-05-01].

Pedersen, N. L. (2018). Aspects of stress in optimal shaft shoulder fillet [Cited by: 4; All Open Access, Green Open Access]. *Journal of Strain Analysis for Engineering Design*, *53*(5), 285–294. https://doi.org/10.1177/0309324718763514

Rock Machinery. (2025). *Sandvik® superior cone crushers* [Accessed: 2025-06-05]. https://www.rockmachinery.com/crusherparts-superior.htm

Sabhadiya, S. (2021). *What is belt drive? – types and materials of belt* [Accessed: 2025-04-22]. https://www.theengineeringchoice.com/what-is-belt-drive/

Sabhadiya, S. (2024). *Shafts: Definition, types, and application* [Accessed: 2025-05-08]. https://www.theengineeringchoice.com/what-is-mechanical-shaft/

SIS - Standardiseringskommissionen i Sverige. (1978, April). SS 1862: Gängsystem – Metrisk gänga [Utgåva 3. Standarden utarbetad av SMS, Sveriges Mekanstandardisering. Första giltighetsdag: 1978-04-01].

SKF Group. (2018). *Rolling bearings* [PUB BU/P1 17000/1 EN, October 2018]. SKF. https://www.skf.com/go/17000

Solutions, G. (2022). *Bevel gear strength rating: The appropriate combination of fem with rating standards* [Accessed: 2025-06-05]. https://gearsolutions.com/features/bevel-gear-strength-rating-the-appropriate-combination-of-fem-with-rating-standards/

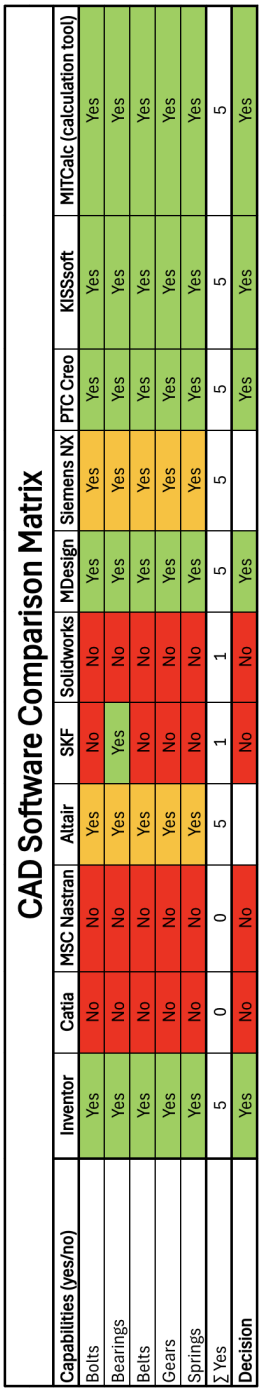
Swedish Standards Institute. (1967). SMS 2476: Power Transmission — Belt Drives [Accessed via SIS Abonnemang, www.sis.se].

Vuori, V., Helander, N., & Okkonen, J. (2019). Digitalization in knowledge work: The dream of enhanced performance. *Cognition, Technology & Work*, *21*(2), 237–252.

Wasfy, T. M., & Wasfy, H. M. (2019). Effect of rubber material properties on steady-state flat belt response [Cited by: 1]. *Proceedings of the ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE)*, *6*. https://doi.org/10.1115/DETC2019-98166

A

CAD software Comparison Matrix



CAD Software Comparison Matrix											
Capabilities (yes/no)	Inventor	Catia	MSC Nastran	Altair	SKF	Solidworks	MDesign	Siemens NX	PTC Creo	KISSsoft	MITCalc (calculation tool)
Bolts	Yes	No	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes
Bearings	Yes	No	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
Belts	Yes	No	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes
Gears	Yes	No	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes
Springs	Yes	No	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes
∑Yes	5	0	0	5	1	1	5	5	5	5	5
Decision	Yes	No	No		No	No	Yes		Yes	Yes	Yes

Figure A.1: CAD Software Comparison Matrix

B

MITCalc User interface

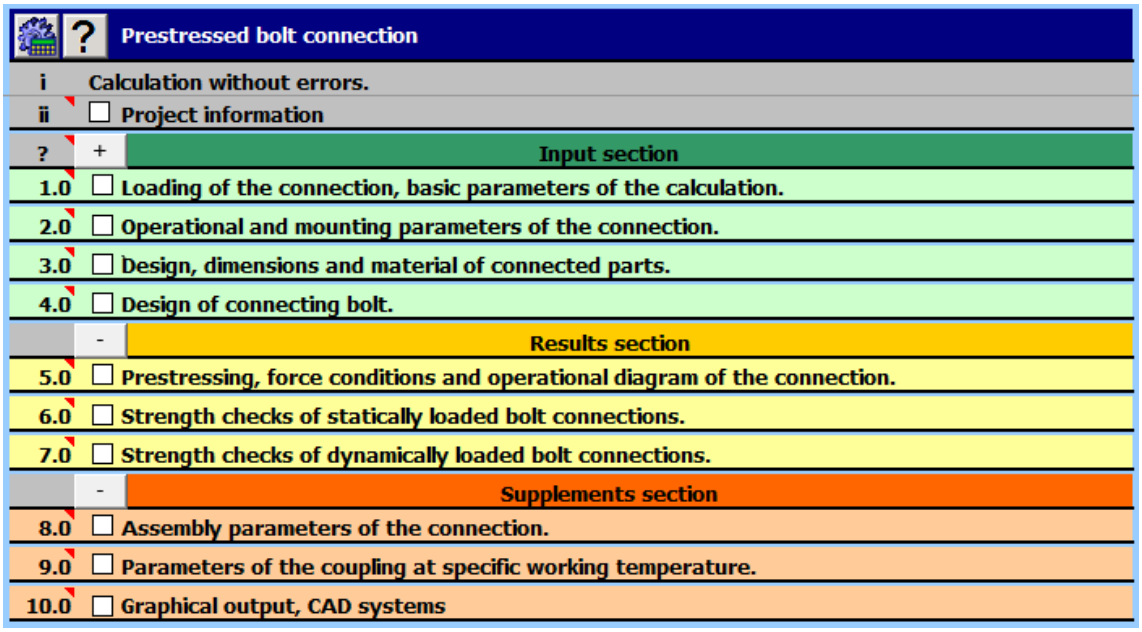


Figure B.1: MITCalcs user interface

C

Inventor User Interface

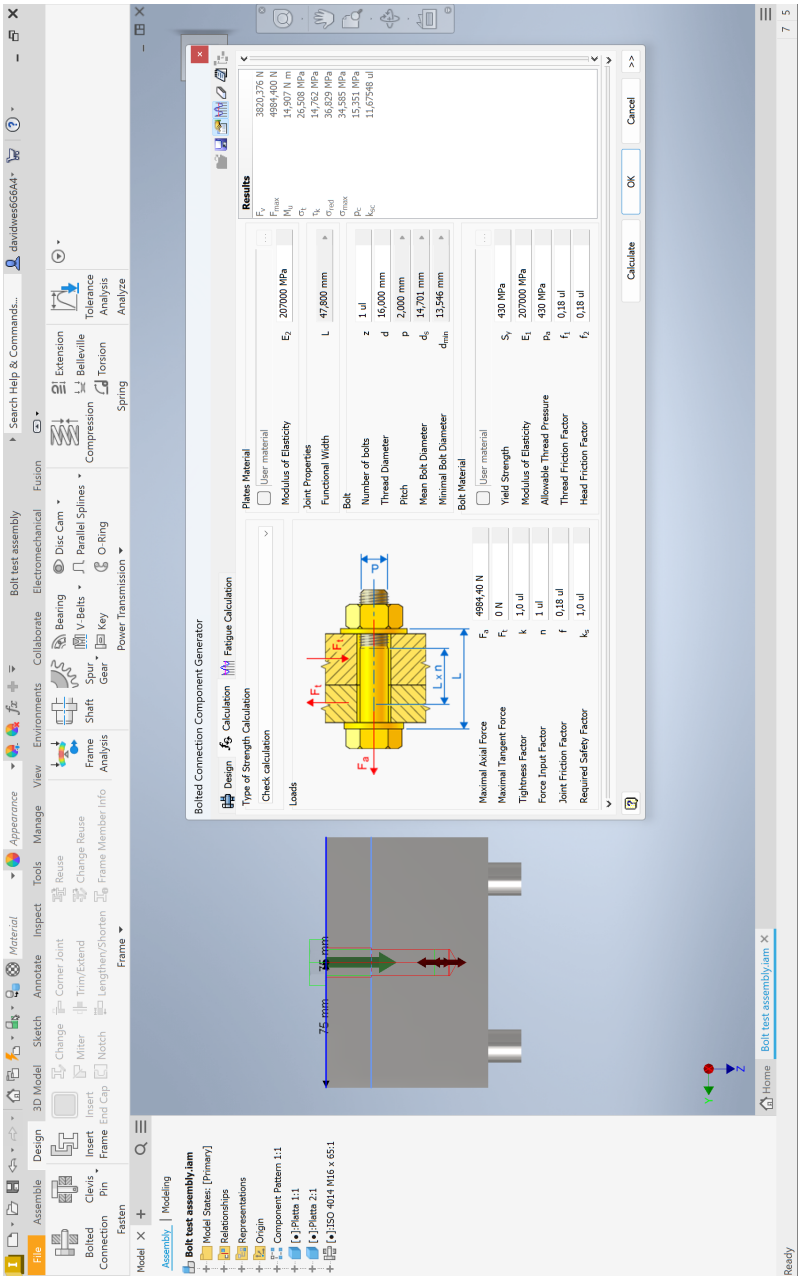


Figure C.1: Inventor design accelerator user interface.

D

MDesign User interface

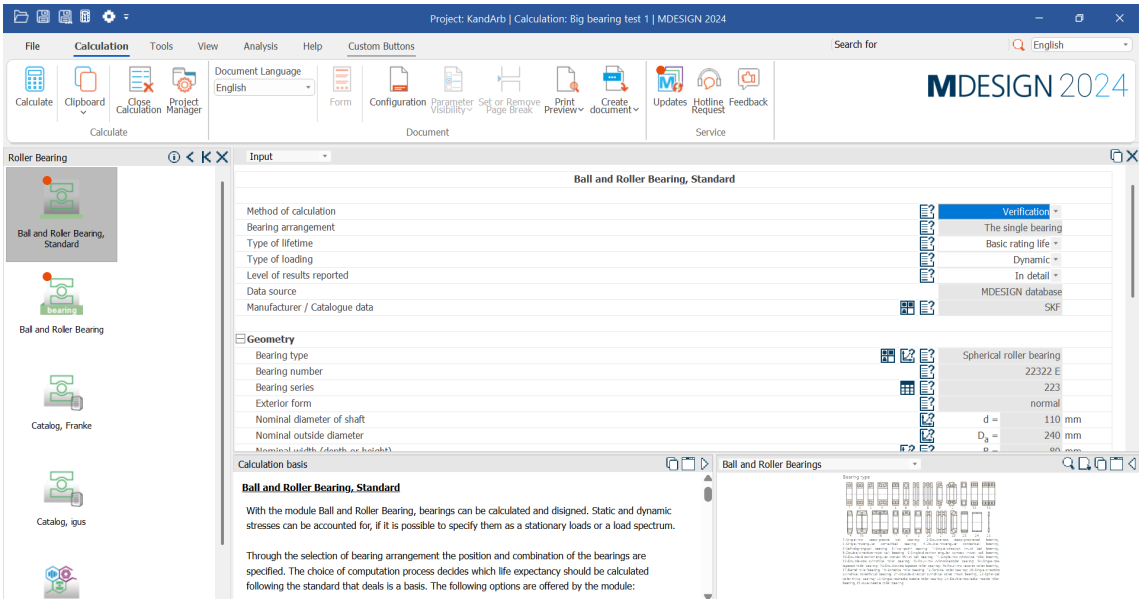


Figure D.1: MDesign user interface

E

Result Graphs for shafts

E.0.1 Graphs for hand calculations

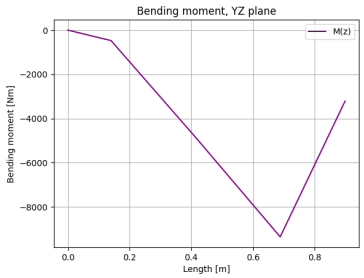


Figure E.1: Distribution of von Mises stress along the shaft.

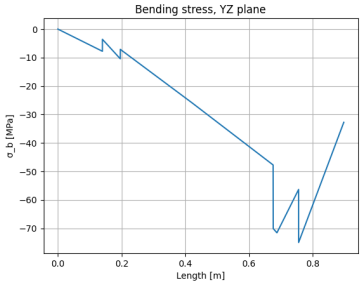


Figure E.2: Distribution of bending stress along the shaft.

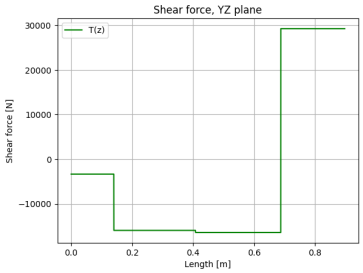


Figure E.3: Distribution of shear force along the shaft.

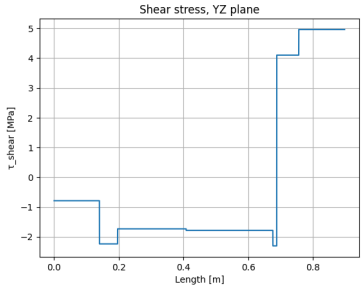


Figure E.4: Distribution of shear stress along the shaft.

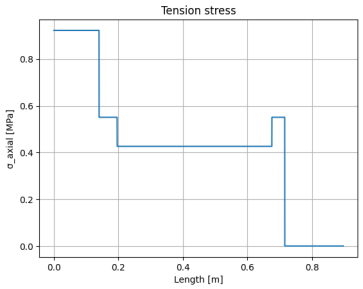


Figure E.5: Distribution of tension stress along the shaft.

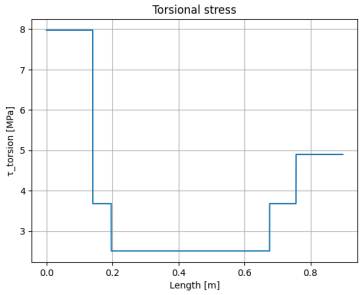


Figure E.6: Distribution of torsional stress along the shaft.

E.0.2 Inventor graphs

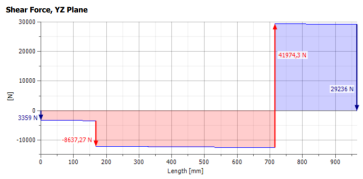


Figure E.7: Shear Force YZ plane, Autodesk Inventor.

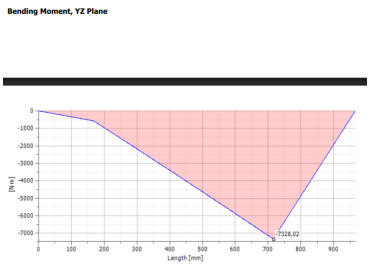


Figure E.8: Bending Moment YZ plane, Autodesk Inventor.

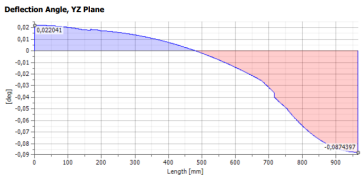


Figure E.9: Deflection angle YZ plane, Autodesk Inventor.

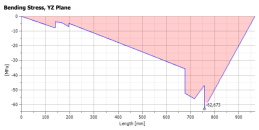


Figure E.10: Bending stress YZ plane, Autodesk Inventor.

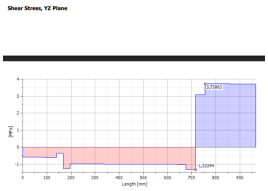


Figure E.11: Shear stress YZ plane, Autodesk Inventor.

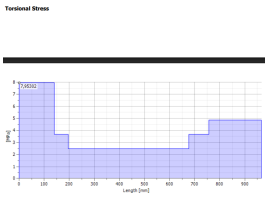


Figure E.12: Torsional Stress, inventor, Autodesk Inventor.

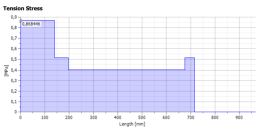


Figure E.13: Tension stress, Autodesk Inventor.

E.0.3 Graphs for MITcalc

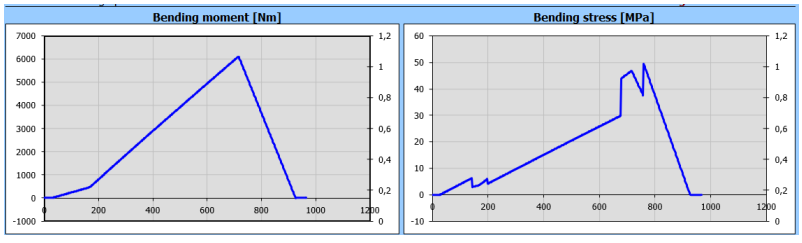


Figure E.14: Bending moment and Bending stress, MITCalc.

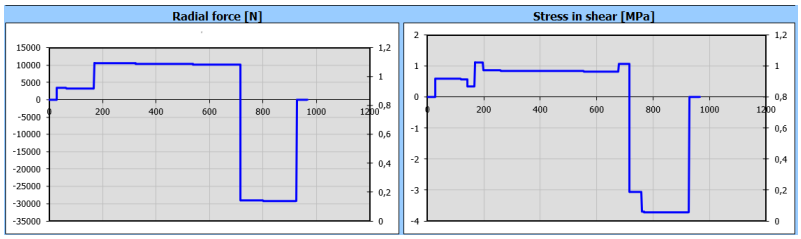


Figure E.15: Radial force and Stress in shear, MITCalc.

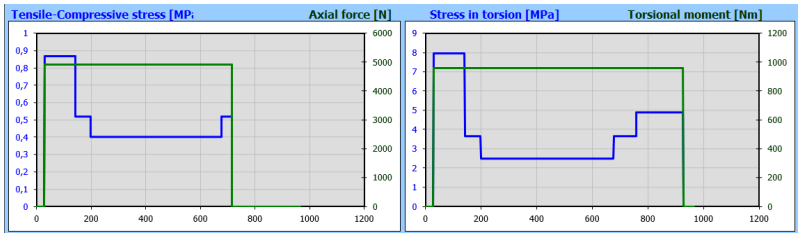


Figure E.16: Tensile stress and Torsion stress, MITCalc.

E.0.4 Graphs for MDesign

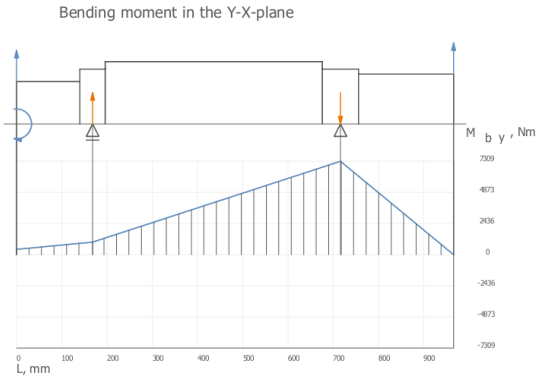


Figure E.17: Bending moment in the Y-X-plane, MDesign.

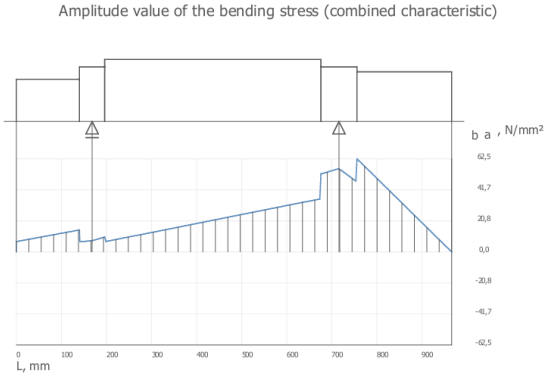


Figure E.18: Amplitude of bending stress, MDesign.

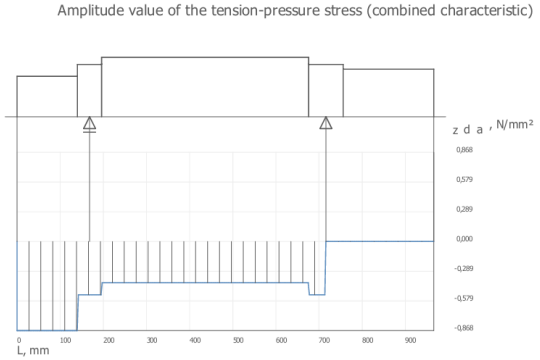


Figure E.19: Amplitude of tension-pressure stress, MDesign.

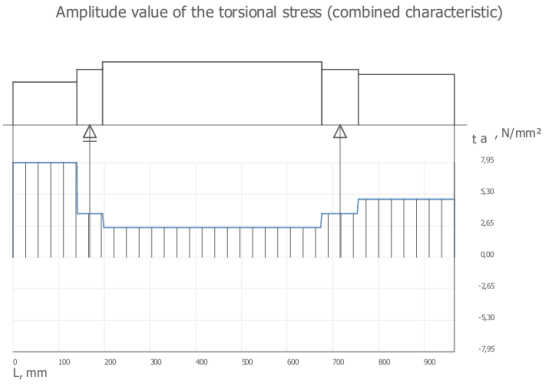


Figure E.20: Amplitude of torsional stress, MDesign.

F

Shaft Python Code

```
import numpy as np
import matplotlib.pyplot as plt

# Geometry
l1 = 140e-3
l2 = 56e-3
l3 = 480e-3
l4 = 80e-3
l5 = 210e-3
a = l1
b = (l2+l3)/2
c = (l3+l4)/2
d = l5
z_total = a + b + c + d

# Forces
Fa = 4927.76
Fr = 3359.07
m = 48.1144
g = 9.81
Fg = m*g
Fb = 29236
tyngdpunkt = 485.9e-3

# Reaction forces
R2 = (Fr * (l1 + l2/2) + Fb * (l2/2 + l3 + l4 + l5) - Fg *
    tyngdpunkt) / (l2/2 + l3 + l4/2)
R1 = Fr + Fb + Fg - R2

# Shear and moment functions
def T1(z): return -Fr
def T2(z): return -(Fr - R1)
def T3(z): return -(Fr - R1 + Fg)
def T4(z): return -(Fr - R1 + Fg - R2)

def M1(z): return -Fr * z
def M2(z): return -Fr * z + R1 * (z - a)
```

```
def M3(z): return M2(z) - Fg * (z - a - b)
def M4(z): return M3(z) + R2 * (z - a - b - c)

# Discretize z
tol = 0.0001
z1 = np.arange(0, a, tol)
z2 = np.arange(a, a + b, tol)
z3 = np.arange(a + b, a + b + c, tol)
z4 = np.arange(a + b + c, z_total, tol)

# Shear and moment results
T_list = np.concatenate([[T1(z) for z in z1], [T2(z) for z
    in z2], [T3(z) for z in z3], [T4(z) for z in z4]])
M_list = np.concatenate([[M1(z) for z in z1], [M2(z) for z
    in z2], [M3(z) for z in z3], [M4(z) for z in z4]])
z = np.concatenate([z1, z2, z3, z4])

# Plot shear force
plt.plot(z, T_list, label='T(z)', color='green')
plt.title('Shear force, YZ plane')
plt.xlabel('Length [m]')
plt.ylabel('Shear force [N]')
plt.grid()
plt.legend()
plt.show()

# Plot bending moment
plt.plot(z, M_list, label='M(z)', color='purple')
plt.title('Bending moment, YZ plane')
plt.xlabel('Length [m]')
plt.ylabel('Bending moment [Nm]')
plt.grid()
plt.legend()
plt.show()
```

```
# Stress Analysis
lengths = [l1, l2, l3, l4, l5]
diameters = [85e-3, 110e-3, 125e-3, 110e-3, 100e-3]
T = 961.56
F_axial = 5237.84

sigma_b_list = []
tau_shear_list = []
tau_torsion_list = []
sigma_axial_list = []
boundaries = np.cumsum([0] + lengths)
```

```
for zi, Mi, Ti in zip(z, M_list, T_list):
    if boundaries[0] <= zi < boundaries[1]: d_local =
       diameters[0]
    elif boundaries[1] <= zi < boundaries[2]: d_local =
       diameters[1]
    elif boundaries[2] <= zi < boundaries[3]: d_local =
       diameters[2]
    elif boundaries[3] <= zi < boundaries[4]: d_local =
       diameters[3]
    else: d_local = diameters[4]

    A = (np.pi * d_local**2) / 4
    I = (np.pi * d_local**4) / 64
    J = (np.pi * d_local**4) / 32
    c_local = d_local / 2

    sigma_b = (Mi * c_local) / I
    tau_shear = Ti / A
    tau_torsion = (T * c_local) / J
    sigma_axial = F_axial / A if zi <= 0.716 else 0

    sigma_b_list.append(sigma_b)
    tau_shear_list.append(tau_shear)
    tau_torsion_list.append(tau_torsion)
    sigma_axial_list.append(sigma_axial)

sigma_b_list = np.array(sigma_b_list)
tau_shear_list = np.array(tau_shear_list)
tau_torsion_list = np.array(tau_torsion_list)
sigma_axial_list = np.array(sigma_axial_list)

# Von Mises
sigma_von_mises = np.sqrt(
    sigma_b_list**2 + sigma_axial_list**2 + 3 *
       tau_torsion_list**2
)
sigma_tillaten = 335e6
sigma_vm_max = np.max(sigma_von_mises)
safety_factor = sigma_tillaten / sigma_vm_max

print(f"Max von Mises Stress: {sigma_vm_max/1e6:.3f} MPa")
print(f"Safety Factor: {safety_factor:.2f}")
```

```
# Angular deflection
G = 80e9
```

```
phi_R1 = 0.0
phi_R2 = 0.0
for L, d in zip(lengths, diameters):
    J = (np.pi * d**4) / 32
    phi = T * L / (G * J)
    phi_R2 += phi

print("\n--- Angular deflection ---")
print(f"  (R1) = {phi_R1:.6f} rad = {phi_R1 * 180/np.pi:.4f}
    ")
print(f"  (R2) = {phi_R2:.6f} rad = {phi_R2 * 180/np.pi:.4f}
    ")
```

```
# Shaft mass
rho = 7850
mass = 0
for L, d in zip(lengths, diameters):
    A = np.pi * (d ** 2) / 4
    V = A * L
    mass += rho * V

print(f"Ideal shaft mass: {mass:.3f} kg")
```

G

MITCalc load shaft

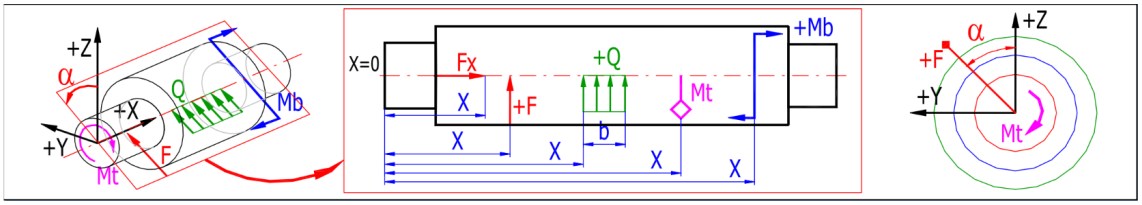


Figure G.1: MITCalc's shaft load distribution

H

Belt drive Python code

```
import numpy as np
from tabulate import tabulate

p = 75000 # Tranferred power
v1 = 1500 # Speed of pulley 1 (rpm)
v2 = 500  # Speed of pulley 2 (rpm)
Eta = 0.95 # Efficiency of the belt drive
Conv_factor = 60/(2*np.pi)
def manner_of_loading(p, v1, v2, Eta, Conv_factor):
    P_dist_pulley = (p*Eta) # Power with
                            # considering efficiency
    Speed_Ratio = v1/v2
    torque_1 = p*Conv_factor/v1 # Rad/s for pulley 1
    torque_2 = P_dist_pulley*Conv_factor/v2 # Rad/s
        for pulley 2
    return P_dist_pulley, Speed_Ratio, torque_1, torque_2

p_dist_pulley, Speed_Ratio, torque_1, torque_2 = \
    manner_of_loading(p, v1, v2, Eta, Conv_factor)
print("Power considering efficiency: ", p_dist_pulley)
print("Speed ratio: ", Speed_Ratio)
print("Torque for pulley 1: ", torque_1)
print("Torque for pulley 2: ", torque_2)
```

```
P_dist_pulley, Speed_Ratio, torque_1, torque_2 = \
    manner_of_loading(p, v1, v2, Eta, Conv_factor)
R1 = 0.125 # Radius Pulley 1
R2 = 0.380 # Radius Pulley 2
L = None
axis_distance = 1
tol = 1e-8
max_iter = 1000
def design(R1, R2, axis_distance, L, tol):

    if axis_distance:
```

```
        d_alpha = np.arcsin((R2 - R1)/axis_distance)
        alpha1 = np.pi - 2*d_alpha
        alpha2 = np.pi + 2*d_alpha
        L = (2*(R2 - R1)/np.tan(d_alpha)) + np.pi*(
            R2 + R1) + 2*(R2 - R1)*d_alpha
    else:
        pass

    if L:
        delta_alpha_i = 0.1
        for iteration in range(max_iter):
        # Compute the next iteration using the given equation
            delta_alpha_next = np.arctan(
                (1 + (delta_alpha_i * np.tan(delta_alpha_i)))
                  * ((2 * (R2 - R1)) /
                     (L - (np.pi * (R2 + R1))))
            )

            # Check for convergence
            if abs(delta_alpha_next - delta_alpha_i) < tol:
                break
            delta_alpha_i = delta_alpha_next    # Update for
                                                # the next
                                                # iteration
            axis_distance = (R2 - R1) / np.sin(delta_alpha_i)
        # Print the final result
    else:
        pass
    alpha1 = alpha1 * 180/np.pi
    alpha2 = alpha2 * 180/np.pi
    return L, alpha1, alpha2, \
        axis_distance, delta_alpha_i, iteration, d_alpha

L, alpha1, alpha2, axis_distance, delta_alpha_i, \
    iteration, d_alpha = \
    design(R1, R2, axis_distance, L, tol)

print(f"Belt length with axis distance: \
    {axis_distance:.2f} is {L:.2f}")
print(f"Axis distance with belt length: \
    {L:.2f} is {axis_distance:.2f}")
print(f"Converged delta_alpha: \
    {delta_alpha_i:.6f} after {iteration+1} iterations")
print(alpha1, alpha2)
```

```
def angle_correction_factor():
    """
    This calculates the Angle of wrap correction factor,
    Due to that we take standard values from tables,
    wee need to interpolate.

    ACF1: The lower Angle correction factor
    ACF2: The higher Angle correction factor
    alpha1: The lower angle of wrap
    alpha2: The higher angle of wrap
    alpha: The angle of wrap

    From SMS 2476 Table 5
    """
    ACF1 = 0.93
    ACF2 = 0.92
    alpha1 = 151
    alpha2 = 148
    alpha_real = alpha1
    ACF = ACF1 + ((alpha1 - alpha1)/(alpha2 - alpha1)) * (
        ACF2 - ACF1
    )
    return ACF
ACF = angle_correction_factor()
print(f"Angle correction factor: {ACF}")
```

```
def length_correction_factor():
    """
    This calculates the length correction factor,
    Due to that we take standard values from tables,
    wee need to interpolate.

    LCF1: The lower Length correction factor
    LCF2: The higher Length correction factor
    L1: The lower length of the belt
    L2: The higher length of the belt
    Lx: The length of the belt

    From SMS 2476 Table 4
    """
    LCF1 = 1
    LCF2 = 1.02
    L1 = 3600
```

```
    L2 = 3800
    Lx = 3650
    LCF = LCF1 + ((Lx - L1)/(L2 - L1)) * (LCF2 - LCF1)
    return LCF
LCF = length_correction_factor()
print(f"Length correction factor: {LCF}")

---------------------------------------------------------------


def effekt_factor():
    """
    Transferred power factor, Due to that we take
    standard values from tables, wee need to interpolate.

    FF1: The lower Effect factor
    FF2: The higher Effect factor
    Speed1: The lower speed of the faster pulley
    Speed2: The higher speed of the faster pulley
    Standards:
        SMS 2479
        .
        .
        .
        SMS 2485
    """

    FF1 = 0
    FF2 = 9.19
    Speed1 = 1800
    Speed2 = 1500
    FF = FF1 + ((v1 - Speed1)/(Speed2 - Speed1)) * (
        FF2 - FF1
    )
    return FF
# FF = effekt_factor()
FF = 9.19
print(f"Effect factor: {FF}")

---------------------------------------------------------------


def added_effekt_factor():
    """
    If necessary, this interpolates the
    first value for the addition for Pr
    """
```

```
    TFF1 = 0.70
    TFF2 = 0.76
    Speed1 = 2400
    Speed2 = 2600
    TFF = TFF1 + ((v1 - Speed1) / (Speed2 - Speed1)) * (
        TFF2 - TFF1
    )
    return TFF
# TFF = added_effekt_factor()
TFF = 1.38
print(f"Effect factor: {TFF}")

-------------------------------------------------------------


L, alpha1, alpha2, axis_distance, delta_alpha_i, \
iteration, d_alpha = design(R1, R2, axis_distance, L, tol)

mu = 0.4 #Friction coefficient
m_dot = 0.324
e = 1.3 # Service factor (SMS 2476)
beta = 34   # Groove angle for smaller pulley,
            # due to that datum diameter < 315 (ISO 4183)
omega = 0.42

def forces(mu, m_dot, e, beta):
    v_belt = (2 * R1 * v1 * np.pi) / 60 # Belt speed

    Fu = p / v_belt # Tensile Froce

    # F2 = Fu * (1/(np.exp(mu * alpha1) - 1))

    F1_ideel = ((R2 * torque_1) - (torque_2 * R1)) / (
        R1 - R2 * R1
    ) # F1 froom the book

    F2_ideel = (torque_1 + F1_ideel * R1) / R1 # F2 from book

    pd = (e * p)*10**-3 # Pd (operating power) from SMS 2476

    pr = FF + TFF   # Transmittable power
                    # including additional
                    # power (which is taken from
                # tables in SMS 2479 to 2485)

    num_belts = pd / (pr * LCF * ACF)   # Number of belts
                                        # from SMS 2476
```

```
    Fc = num_belts * m_dot * (v_belt)**2 # Fc from
            # Maskinelement book, note that Fc
            # needs to multiplied with number of belts

    mu_s = mu / np.sin(beta) # Apparent coefficient of
        # friction from Maskinelement book,
        # this is calculated from the groove angle
        # and the friction coefficient

    chi = np.exp(omega * mu_s * alpha2*(np.pi/180)) # value
        # for load on belt drive, the angle of erapping
        # is from the larger

    chi_slip = np.exp(mu_s * alpha2*(np.pi/180)) # value
        # for load on belt drive,
        # the angle of erapping is from the larger, slip

    F_e1 = torque_1 / ((chi - 1)*R1) # Effective force 1
                        # (see page. 362 Maskinelement)

    F_e2 = F_e1 * chi # Effective force 2

    return Fu, mu_s, chi_slip, chi, F_e1, \
        F_e2, Fc, v_belt, num_belts

Fu, mu_s, chi_slip, chi, F_e1, F_e2, Fc, \
    v_belt, num_belts = forces(mu, m_dot, e, beta)


data = [
    ["Tensile force", Fu],
    ["Friction coefficient", mu_s],
    ["Effective force 1", F_e1],
    ["Effective force 2", F_e2],
    ["Number of belts", num_belts],
    ["Belt speed", v_belt],
    ["Centrifugal force", Fc],
]
print(tabulate(data, headers=["Description", "Value"], \
    tablefmt="fancy_grid", colalign=("left", "left"))
)

Fu, mu_s, chi_slip, chi, F_e1, F_e2, \
    Fc, v_belt, num_belts = forces(mu, m_dot, e, beta)
```

```
def prestressing():
    """
    There are different ways to calculate the prestressing
    force, redarging the case and what data you already got.
    The following code will determine the different cases.
    """

    """ Given axial load: """
    F_outer = F_e1 + F_e2
    lam1 = torque_1 / (R1 * F_outer)
    F_0_axiall= (torque_1 + (2 * R1 * Fc * lam1)) / (
        2 * R1 * lam1
    ) # From figure 10.19 in Maskinelement book

    """ Given Axial distance: """
    lam2 = (chi - 1) / (chi + 1)
    F_0_axiald = (torque_1 + (2 * lam2 * R1 * Fc)) / (
        R2 * lam2
    )

    """
    Tensioner Pulley (Note that we do not need a
    prestress force, the tensioner pulley
    solves this and creating new effective forces):
    """
    xi = 10 # Angle that the tensioner
            # pulley creates when applied
    tp_mass = 10
    g = 9.82

    F_e1_tp = (tp_mass * g) / (2 * np.sin(xi)) # Effective
        # force on the belt side where the tensioner
        # pully is (Note that this is a constant)

    F_e2_tp = (torque_1 / R1) + F_e1_tp # Effective force
                        # on the other side of the belt

    chi_tp = F_e2_tp / F_e1_tp # Value for load on belt drive
    lam_tp = (chi - 1) / (chi + 1)

    # F_0 = (F_e1 + F_e2)*np.cos(d_alpha)

    # lam = (chi_slip - 1)/(chi_slip + 1)
```

```
    return lam1, lam2, F_0_axiall, F_0_axiald

lam1, lam2, F_0_axiall, F_0_axiald = prestressing()
data = [
    ["Load ratio (Given Axial load)", lam1],
    ["Load ratio (Given Axial distance)", lam2],
    ["Pre load (Given Axial load)", F_0_axiall],
    ["Pre load (Given Axial distance)", F_0_axiald],
]
print(tabulate(data, headers=["Description", "Value"], \
    tablefmt="fancy_grid", colalign=("left", "left"))
)

-------------------------------------------------------------


def froces1and2():
    """
    This calculates the forces on the belt,
    both the driven and the driving side.
    The following two equations are from
    how MITcalc calculates F1 and F2
    """

    F1_new = F_0_axiall + Fu / 2
    F2_new = F_0_axiall - Fu / 2
    return F1_new, F2_new
F1_new, F2_new = froces1and2()

print(f"F1_new: {F1_new:.2f} N")
print(f"F2_new: {F2_new:.2f} N")

rho = 10
def radialforce(rho):
    F_ax_driven = ((np.cos(beta)/(2 * mu)) * (
        F_e2 - F_e1)) + (0.5 * (1 - omega
        ) * alpha2 * (1 / np.tan(beta + rho))) * F_e1

    F_ax_driving = (0.5 * alpha1 * (
        1 / np.tan(beta + rho))) * F_e2

    return F_ax_driven, F_ax_driving
F_ax_driven, F_ax_driving = radialforce(rho)

print(f"F_ax: {F_ax_driving*10**-3:.2f} N")
```

I

Bearing Life Python Code

```
import math

def equivalent_bearing_load(Fr, Fa, e, Y1, Y2):
    """
    Calculates the equivalent bearing load for spherical
       roller bearings according to SKF standard.
    """
    if Fa / Fr <= e:
        return Fr + Y1 * Fa
    else:
        return 0.67 * Fr + Y2 * Fa

def calculate_nominal_life(C, P, p=10/3):
    """
    Calculates the nominal bearing life (L10) in million
       revolutions.
    Uses p = 10/3 for spherical roller bearings.
    """
    return (C / P) ** p

def convert_life_to_hours(L10, speed_rpm):
    """
    Converts bearing life from million revolutions to
       operating hours.
    """
    return (L10 * 1_000_000) / (60 * speed_rpm)

def calculate_a1(reliability):
    """
    Calculates the reliability factor a1 based on desired
       reliability (e.g., 0.99 for 99%).
    """
    return 0.05 + 0.95 * ((math.log(reliability) / math.log
       (0.9)) ** (2/3))

# Input data, big bearing 22322E, small bearing 23122 CC/W33
Fr_small = 12253.1
```

```
Fr_big = 45532.0
Fa_small = 0
Fa_big = 5237.8
C_small = 450000
C_big = 989000
e_small = 0.3
e_big = 0.33
Y1_small = 2.3
Y2_small = 3.4
Y1_big = 2
Y2_big = 3
Pu_small = 61000
Pu_big = 100000
speed_rpm = 1440
reliability = 0.99 # for 99% reliability

# Calculate a1 from reliability
a1 = calculate_a1(reliability)

# Equivalent bearing load calculation
P_small = equivalent_bearing_load(Fr_small, Fa_small,
    e_small, Y1_small, Y2_small)
P_big = equivalent_bearing_load(Fr_big, Fa_big, e_big,
    Y1_big, Y2_big)

# Nominal life calculation
L10_small = calculate_nominal_life(C_small, P_small)
L10_big = calculate_nominal_life(C_big, P_big)

L10_small_hours = convert_life_to_hours(L10_small, speed_rpm
    )
L10_big_hours = convert_life_to_hours(L10_big, speed_rpm)

print(f"Big bearing data:")
print(f"Equivalent bearing load (P): {P_big:.2f} N")
print(f"Nominal bearing life (L10) in million revolutions: {
    L10_big:.2f}")
print(f"Nominal bearing life in operating hours: {
    L10_big_hours:.2f} hours")
print(f"Small bearing data:")
print(f"Equivalent bearing load (P): {P_small:.2f} N")
print(f"Nominal bearing life (L10) in million revolutions: {
    L10_small:.2f}")
print(f"Nominal bearing life in operating hours: {
    L10_small_hours:.2f} hours")
```

```
etha_c = 0.5
product_small = etha_c * (Pu_small / P_small)
product_big = etha_c * (Pu_big / P_big)

# Using arbitrary values for kappa = v/v1 because of the
    lack of knowledge on which oil is used
kappa = 2.5

# Values from SKF graph for chosen kappa
a_SKF_small = 10
a_SKF_big = 0.3

SKF_life_small = a1 * a_SKF_small * L10_small
SKF_life_hours_small = SKF_life_small * 1_000_000 / (60 *
    speed_rpm)
SKF_life_big = a1 * a_SKF_big * L10_big
SKF_life_hours_big = SKF_life_big * 1_000_000 / (60 *
    speed_rpm)

print(f"Life adjustment factor a1: {a1:.4f}")
print(f"Data for big bearing: ")
print(f"Bearing life according to SKF life formula: {
    SKF_life_big:.4f} million revolutions")
print(f"Bearing life in hours: {SKF_life_hours_big:.4f}")
print(f"Data for small bearing: ")
print(f"Bearing life according to SKF life formula: {
    SKF_life_small:.4f} million revolutions")
print(f"Bearing life in hours: {SKF_life_hours_small:.4f}")
```

J

Bolted Joint Python Code

```
import math

#The following calculation is based on Maegi, M., Melkersson
    , K., & Evertsson, M. (2017). Maskinelement (1:2 edition)
    .

#Define variables
E_s = 207e9   #E-Module [Pa] for screw (Broberg et al.
    (2017), Handbok och formelsamling i haallfasthetslaera,
    3:rd edition)
E_k = 207e9   #E-Module [Pa] for flange (Broberg et al.
    (2017), Handbok och formelsamling i haallfasthetslaera,
    3:rd edition)

#Following values are taken from following standards: coarse
     threads (SS-ISO 273),
#spanner sizes and contact diameters for nuts (SS-ISO
    4014/4017/4018 respective 4032/4034)
#(Maegi et al. (2017), p.58)
L_k = 27*10**-3 #[m]
d_1 = 13.835*10**-3 #[m]
d_2 = 14.701*10**-3 #[m]
d_w = 22*10**-3 #[m]
d_h = 17.5*10**-3 #[m]
d_0 = 16*10**-3 #[m]
P = 2e-3 #[m]

friction_thread = 0.18 #table 2.2 (Maegi et al. (2017), p
    .67)

N = 1 #Number of bolts

F_N = 4984.40/N

A_s = (math.pi*(d_1**2))/4 #Bolt cross section area

x = (L_k / (L_k + d_w)) ** (1/5) #Blind connection type
```

```
#Equivalent cross section area
A_ekv = (math.pi/4) * (d_w**2 - d_h**2) + (math.pi/8) * L_k
    * d_w * ((x + 1)**2 - 1)
#A_ekv = 0.0003487874703831729   #Test with same A_eq as in
    Inventor
print('A_ekv =', A_ekv, 'm^2')

#Stiffnesses
C_s = ((A_s * E_s) / L_k)*10**-3 #screw stiffness adjusted
    to N/mm, (Maegi et al. (2017), p.69)
C_k = ((A_ekv * E_k) / L_k)*10**-3 #flange stiffness
    adjusted to N/mm, (Maegi et al. (2017), p.70)

print('C_s =', C_s, 'N/mm')
print('C_k =', C_k, 'N/mm')

F_0 = F_N*((C_k)/(C_s+C_k))*1.05   #(Maegi et al. (2017), p
    .72), 5% safety factor to prevent
F_k = F_0-(C_k/(C_s+C_k))*F_N    #(Maegi et al. (2017), p
    .72)
F_s = F_0+(C_s/(C_s+C_k))*F_N   #(Maegi et al. (2017), p.72)

print('F_0=', F_0, 'N')

#settling

#Approximated values from Ritkompendium_Ytfinhet.pdf
#R_a1 = 1.2e-6 #Bult-flaens
#R_a2 = 1.2e-6 #Flaens-flaens
#R_a3 = 2.4e-6 #Gaengor

#(Maegi et al. (2017), p.75)
#H1 = 4*1.2e-6 #Bult-flaens
#H2 = 4*1.2e-6 #Flaens-flaens
#H3 = 4*2.4e-6 #Gaengor

#deformation in different parts
delta_pl1 = 2e-3 #Bult-flaens, (Maegi et al. (2017), p.75),
    figure 2.23, graph "saettningen per kontaktpar foer
    axialbelastning F_N"
delta_pl2 = 2e-3 #Flaens-flaens, (Maegi et al. (2017), p.75)
    , figure 2.23, graph "saettningen per kontaktpar foer
    axialbelastning F_N"
delta_pl3 = 2.1e-3 #Gaengor, (Maegi et al. (2017), p.75),
    figure 2.23, graph "saettningen i gaengorna"
```

```
delta_pl = delta_pl1 + delta_pl2 + delta_pl3 #total
    deformation
print('delta_pl=', delta_pl, 'mm')

F0_settling = delta_pl*(C_s*C_k)/(C_s+C_k) + F_0

print('F0_settling=', F0_settling, 'N')

#Tightening moment

r_m = (d_w + d_h) / 4 #average bolt radius

M_tightnening_settling = F0_settling * (P/(2*math.pi) +
    friction_thread*(0.58*d_2 + r_m))

print('Tightnening moment=', M_tightnening_settling, 'Nm')
```

K

Inventor Manual Calculation Summorised

$$C_{10} = \frac{4\,(L + 0.8d)}{E_1 \cdot \pi \cdot d_s^2} \qquad \text{Equation (A)}$$

$$C_{20} = \frac{L}{E_2 \cdot \frac{\pi}{4}\left(\left(1.5d + \frac{L}{a}\right)^2 - 1.05d^2\right)} \qquad \text{Equation (B)} \qquad \Rightarrow A_{eq} = \frac{\pi}{4}\left(\left(1.5d + \frac{L}{a}\right)^2 - 1.05d^2\right)$$

Where:

E_1 : Bolt E- modulus
E_2 : Clamped part E- modulus
L : Clamped part length
d_s : Mean bolt diameter
d : Thread diameter
a : Material constant for elastic deformation

Where:

A_{eq} : Equivalent cylinder area

$$C_1 = C_{10} + (1 - n)\,C_{20} \qquad \text{Equation (C)}$$

$$C_2 = n \cdot C_{20} \qquad \text{Equation (D)}$$

Where:

C_1 : Bolt stiffness
C_2 : Clamped part stiffness

$$F_{\max} = \frac{\psi}{z}\left(F_a + \frac{F_t}{f}\right) = \frac{1}{1}\left(4984.4 + \frac{0}{0.18}\right) = 4984.4N \qquad \text{Equation (E)}$$

$$F_0 = F_{max} - \frac{C_2}{C_1 + C_2} \cdot \frac{F_a}{z} \qquad \text{Equation (F)}$$

Where:

ψ : Tightness factor
F_a : Maximum operational force
$F_{\max}$: Working force in the joint
F_0 : Preload force
z : Number of bolts
F_t : Tangential force
f : Joint friktion factor

$$M_u = F_0\left(\frac{d_2}{2}\tan\left(\frac{p}{\pi d_2} + \frac{f_1}{cos\,(30^o)}\right) + 0.7 \cdot d \cdot f_2\right) \qquad \text{Equation (G)}$$

Where:

M_u : Tightening moment
d_2 : Pitch diameter of the thread
p : Thread pitch
f_1 : Thread friktion factor
f_2 : Friction factor in the contact surface of the bolt

Figure K.1: Manual calculation with Inventor's methodology for bolts

L

Bolted Joint - Inventor Python Code

```
#This code calculates the preload, stiffnesses and the
    tightening moment of a bolt, based on Inventor's
    methodology.

import math

#Material input
E_1 = 207e9
E_2 = 207e9
a = 10  #material constant for steel in inventor's help page

#Dimensions input
d = 16e-3
d_s = 14.701e-3
d_1 = 13.835e-3
d_2 = 14.701e-3
p = 2e-3
L = 27e-3
z=1

#Friction input
f = 0.18
f_1 = 0.18
f_2 = 0.18

#Forces input
F_a = 4984.4
F_t = 0
n = 1
k = 1   #tightness factor, also named \psi. Set to 1.2 when
    considering settling


#Stiffnesses
c_10 = (L + 0.8 * d) * 4 / (E_1 * math.pi * d_s**2)
```

```
c_20 = L / (E_2 * (math.pi / 4) * (((1.5 * d + L / a)**2) -
    1.05 * d**2))
c_1 = (c_10 + (1-n) * c_20)
c_2 = (n * c_20)

#Equivalent cylinder area analysis
A_eq = (math.pi / 4) * (((1.5 * d + L / a)**2) - 1.05 * d
    **2)
print('A_eq =', A_eq, 'm^2')

#Forces calculation
F_max = (k/z) * (F_a + (F_t/f))
F_0 = F_max-(c_2/(c_1+c_2))*F_a/z

#Tightening moment calculation
Mu = F_0 * ((d_2 / 2) * math.tan((p / (math.pi * d_2)) + (
    f_1 / math.cos(math.radians(30)))) + 0.7 * d * f_2)

print('c_1 =', (1/c_1)*10**-3, 'N/mm')
print('c_2 =', (1/c_2)*10**-3, 'N/mm')
print('F_0 =', F_0, 'N')
print('Tightening moment =', Mu, 'Nm')
```

M

Results from all software and hand calculations - Bevel gear

Table M.1: Bevel gear results from hand and software calcualtions in alphabetical order

Parameter	Value	MITCalc	Mdesign	Inventor
alpha_d	20	20	20	
alpha_e	20			
alpha_n	20		20	20
alpha_t	22,59			22,5945
b	107,26	107,26	107,26	107,26
b_e	53,63	53,63	53,63	
b_i	53,63	53,63	53,63	
b_met	107,26			
b_Re	133,50			
beta_bm	27,10			27,1017
beta_e1	46,2897	46,2897	46,29	25,2382
beta_e2	46,2897	46,2897	46,29	25,2382
beta_i1	12,784	12,784	12,784	
beta_i2	12,784	12,784	12,784	
beta_m	29	29	29	29
c	2,0626	2,0627	2,063	
c_be	0,5	0,5	0,5	
c_ham	0,2243	0,2422		
d_ae1	222,0339	222,0347	222,029	225,555
d_ae2	870,7552	870,7545	870,757	870,894
d_ai1	167,5456	167,5461	167,539	171,189
d_ai2	660,6580	660,6575	660,659	660,979
d_e1	193,068	193,068	193,068	193,068
d_e2	868,8065	868,806	868,8065	868,806
d_fe1	180,2721	180,2727	180,266	179,485
d_fe2	861,4747	861,4741	861,476	860,656
d_fi1	136,8204	136,8209	136,815	
d_fi2	653,8302	653,8297	653,832	
d_i1	146,5322	146,5321	146,532	

d_i2	659,3947	659,3944	659,395	
d_m1	169,8	169,8	169,8	169,8
d_m2	764,1006	764,1002	764,101	764,1
d_mn1	219,498	219,498	219,498	
d_mn2	4444,8374	4444,8349	4444,837	
delta_1	12,5288	12,5288	12,529	12,5288
delta_2	77,4712	77,4712	77,471	77,4712
delta_a1	14,7036	14,7036	14,704	14,6703
delta_a2	78,315	78,315	78,315	78,0908
delta_f1	11,685	11,685	11,685	11,6331
delta_f2	75,2964	75,2964	75,296	75,054
E_modul	206000			
epsilon_beta	2,00003136			
f_aP0			0,25	
F_ax1	7150,931	7151,5	7150,931	7150,935
F_ax2	3239,073	3239,3	3239,073	3239,075
F_ber	25399,3			
F_mt1	11325,799	11326,6	11325,799	11325,806
F_mt2	11325,799	11326,6	11325,799	11325,806
F_rad1	3239,073	3239,3	3239,073	3239,075
F_rad2	7150,931	7151,5	7150,931	7150,935
h_ae1	14,8362	14,8366	14,833	
h_ae2	4,4914	4,4911	4,495	
h_ai1	10,7630	10,7633	10,76	
h_ai2	2,9117	2,9114	2,914	
h_am1	12,800	12,8	12,797	
h_am2	3,7016	3,7012	3,705	
h_amc1	13,0627	13,1401	13,137	12,638
h_amc2	3,702	3,702	3,705	2,838
h_e1	21,3903	21,3903	21,391	
h_e2	21,3903	21,3903	21,391	
h_fe1	6,5541	6,5537	6,557	
h_fe2	16,8989	16,8993	16,896	
h_fi1	4,974	4,974	4,977	
h_fi2	12,826	12,826	12,822	
h_fm1	5,7642	5,7639	5,767	
h_fm2	14,8623	14,8626	14,859	
h_i1	15,7373	15,7374	15,737	
h_i2	15,7373	15,7374	15,737	
h_m	18,5638	18,5639	18,564	23,597
h_mw	16,5012	16,5012	16,501	
h_t1	21,3867	21,3867		
j_en	0,41	0,41	0,41	
K_1	1,75		1,75	1,75

k_c	1,25	0,125		
k_d	2	2		
K_Falpha	1			
K_Fbeta	1,25			
K_FN	1			
K_FX	1			
K_Halpha	1			
k_hap	1	1	1	
K_Hbeta	1,25			
k_hfp	1,25	1,25	1,25	
K_HK	1			
K_HN	1			
K_HV-be			1,32	
K_HX	1			
K_L	1			
k_t	0,1778	0,1778		
K_v1	1,281		1,523	
K_Z	0,279			
L_h			20000	
m_et2	10,726	10,726	10,726	10,726
m_mn	8,2506	8,2506	8,251	9,702
n1	1440	1440	1440	1440
n2	320	320		
P	145000	145000	145000	145000
p_r			0,5133	
pressure_angle	20	20	20	20
q			0,2867	
quality			10	
r_c0	170	170	170	
R_e	445	444,9998	445	445
R_e1	445			
R_e21	445	444,9998		
R_i	337,7399	337,7398	337,74	
R_i1	337,7399			
R_i21	337,7399	337,7398		
R_m	391,3700	391,3698		391,37
S_F	4,084		2,147	1,362
S_F1			2,451	3,961
S_F2			2,147	1,362
S_H	1,779		1,549	2,993
S_H1	1,779		1,549	2,993
S_H2			1,622	2,993
s_mn1	17,4955	17,4958	17,495	
s_mn2	7,9388	7,9385	7,939	

s_mnc1	17,4770	17,4773	17,477	19,235
s_mnc2	7,9388	7,9385	7,939	9,488
s_mt1	20,0036	20,0039	20,003	
s_mt2	9,0768	9,0766	9,077	
sigma_F	77,3816	-	299,035	-
sigma_Flim	316	-	732,806	-
sigma_FP	77,381	-	276,531	-
sigma_H	640,649		1114,718	
sigma_Hlim	1140		1726,794	
sigma_HP	640,629		1059,383	
sigma_large	90	90	90	90
T_1	961,56	961,63	961,56	961,56
t_xi1	327,3625	327,3623		326,958
t_xi2	70,424	70,424		69,701
t_xo1	431,1848	431,1845		430,793
t_xo2	92,1496	92,1499		91,836
t_z1	0	0		
t_z2	0	0		
t_zF1	-8,1265	-8,1265		
t_zF2	-2,1061	-2,1061		
t_zm1	382,0503	382,0501		
t_zm2	84,9	84,9		
t_zR1	-0,0011	-0,0011		
t_zR2	-4E-15	0		
theta_a1	2,1748	-8,1265	0	
theta_a2	0,8438	0,8438	0,844	
theta_f1	0,8438	-8,1265	0	
theta_f2	2,1748	2,1748	2,175	
theta_sum_fs	3,019			
theta_sum_fs_deg	3,019			
u	4,5	4,5	4,5	4,5
u_target	4,511	4,5		4,5
v_mt1	12,80			
x_hm1	0,5514	0,5514	0,551	0,551
x_sm1	0,0742	0,0742	0,074	0,0742
x_sm2	-0,1036	-0,1036	-0,104	
x_smn	0,0889	0,0889	0,089	
x_vn1	0,6304			
Y_beta	0,7583			
Y_epsilon	1			
Y_F	2,2		2,04	
Y_K	1,2928			
Y_s	1			
z_1	18	18	18	18

z_2	81	81	81	81
Z_epsilon	1			
Z_H	1,5702			
Z_K	1,1370			
Z_M	268,51		190,2	
Z_R	1			
Z_V	1			
z_v1	18,439			27,65
z_v2	373,392			558,093
z_vn1	26,604			
z_vn2	538,730			

N

Bevel gear Python code

```
import numpy as np
import pandas as pd

#indata for indata
u_target = 4.511  # gearratio

## INDATA
sigma_large = 90 #degrees shaft angle

sigma_large_rad = np.deg2rad(sigma_large) #convert to rad
    for further calculations
z_1 = 18 #Number of pinion teeth

n2 = 320 #rpm cone crusher

P = 145e3 #Power

#R_e = 445 #outer cone distance
pressure_angle = 20 # alpha_d degrees

## According to recommendartion in ISO23509:2016 Figure B.4:
z_2 = int(z_1 * u_target) #Number of wheel teeth, rounded
d_e2 = 868.8065

u = z_2 / z_1 #Actual gearratio after teeth selection

## Rotation speed
n1 = n2 * u #rpm shaft
n1_rad = n1/60*2*np.pi #rad/s shaft

## Angles
delta_1 = (np.arctan(np.sin(sigma_large_rad)/(np.cos(
    sigma_large_rad) + u))) #pitch angle pinion
delta_2 = (sigma_large_rad - delta_1) #pitch angle wheel
alpha_d = np.deg2rad(pressure_angle) #Nominal design
    pressure angle in rad ISO23509 C.2.1
```

```
## Dimensions
R_e = d_e2 / (np.sin(delta_2) * 2)
d_e1 = R_e * np.sin(delta_1) * 2 #pinion outer pitch
    diameter

m_et2 = d_e2 / z_2  #outer transverse module

#tooth width, ISO23509 B.6
b_met = m_et2 * 10
b_Re = 0.3 * R_e

if b_met < b_Re:
  b = b_met
else:
  b = b_Re

R_i = R_e - b #inner cone distance
R_m = R_e - b / 2 #mean cone distance

#Calculation for spiral angle, ISO 23509 B.7.2
K_Z = b/R_e * ((2-b / R_e) / (2 * (1-b / R_e)))
beta_m = 0.01 #small start value
epsilon_beta = 0

while epsilon_beta <= 2: #iterate till epsilon is no longer
    under 2
  epsilon_beta = 1 / (np.pi * m_et2) * (K_Z * np.tan(beta_m)
      - K_Z**3 / 3
                                        * np.tan(beta_m)**3)
                                          * R_e
  beta_m += 0.00001

beta_m_deg = round(np.rad2deg(beta_m),0) #round to integrer
    for simplicity
beta_m = np.deg2rad(beta_m_deg) #convert to rad for further
    calculations

d_i1 = R_i * np.sin(delta_1) * 2 #Pinion inner pitch
    diameter
d_m1 = (d_e1 + d_i1)/2 #Pinion mean pitch diameter
d_i2 = R_i * np.sin(delta_2) * 2 #Wheel inner pitch diameter
d_m2 = (d_e2 + d_i2)/2 #Wheel mean pitch diameter

m_mn = 2 * R_m * np.sin(delta_2) * np.cos(beta_m) / z_2 #
    Mean normal module
```

```
alpha_lim = 0
alpha_n = alpha_d
alpha_e = alpha_n - alpha_lim

t_zm2 = d_m1 * np.sin(delta_2) / (2 * np.cos(delta_1)) #
    crossing point
t_zm1 = d_m2 / 2 #crossing point

t_z1 = R_m * np.cos(delta_1) - t_zm1 #pitch apex beyond
    crossing point along axis
t_z2 = R_m * np.cos(delta_2) - t_zm2 #pitch apex beyond
    crossing point along axis

## Torque
T_1 = P / n1_rad #Torque

#Tooth depth components ISO23509
k_hap = 1 #Addendum factor
k_hfp = 1.25 #Dedendum factor
k_d = 2 #Depth factor Table C.1
k_c = 0.125 #clearance factor
u_a = np.sqrt(np.cos(delta_1)*np.tan(delta_2)/np.cos(delta_2
    ))
c_ham = 0.210 + 0.290 / u_a**2
k_t = -0.088 + 0.092*u - 0.004*u**2 + 0.0016*(z_1-30)*(u-1)
    #thickness factor
x_hm1 = k_d * (1/2 - c_ham)
x_smn = k_t / 2
c_be = 0.5 #face width factor A.3

b_e = b * c_be #face width from calculation point to outside
b_i = b * c_be#face width from calculation point to inside

#cutter
r_c0 = 170 #cutter radius Table E.1

#Determination of tooth depth at calculation point F.2.4
h_mw = 2 * m_mn * k_hap #mean working depth
h_am2 = m_mn*(k_hap - x_hm1) #mean addendum, wheel
h_fm2 = m_mn*(k_hfp + x_hm1) #mean dedendum, wheel
h_am1 = m_mn*(k_hap + x_hm1) #mean addendum, pinion
h_fm1 = m_mn*(k_hfp - x_hm1) #mean dedendum, pinion
c = m_mn * (k_hfp - k_hap) #clearance
h_m = h_am1 + h_fm1 #mean whole depth

#Addendum angle and dedendum angle of wheel, standard C.5
```

```
theta_sum_fs = np.arctan(h_fm1/R_m)+np.arctan(h_fm2/R_m) #
    sum of dedendum angles
theta_a2 = np.arctan(h_fm1/R_m) #addedum angle, wheel
theta_f2 = theta_sum_fs - theta_a2 #dedendum angle, wheel
# Convert radians to degrees
theta_sum_fs_deg = np.rad2deg(theta_sum_fs)
theta_a2_deg = np.rad2deg(theta_a2)
theta_f2_deg = np.rad2deg(theta_f2)

#Determination of root angles and face angles F.2
delta_a2 = delta_2 + theta_a2 #Face angle, wheel
delta_f2 = delta_2 - theta_f2 #Root angle, wheel
delta_a1 = np.arcsin(np.cos(delta_f2)) #Face angle, pinion
delta_f1 = np.arcsin(np.cos(delta_a2)) #Root angle, pinion
theta_a1 = delta_a1 - delta_1 #Addedum angle, pinion
theta_f1 = delta_1 - delta_f1 #Dedendum angle, pinion
#wheel face apex beyond crossing point along wheel axis
t_zF2 = t_z2 - (R_m * np.sin(theta_a2) - h_am2 *
                  np.cos(theta_a2)) / (np.sin(delta_a2))
#wheel root apex beyond crossing point along wheel axis
t_zR2 = t_z2 + (R_m * np.sin(theta_f2) - h_fm2 *
                  np.cos(theta_f2)) / (np.sin(delta_f2))
#pinion face apex beyond crossing point along wheel axis
t_zF1 = ( -t_zR2 * np.sin(delta_f2) - c) / np.sin(delta_a1)
#pinion root apex beyond crossing point along wheel axis
t_zR1 = ( -t_zF2 * np.sin(delta_a2) - c) / np.sin(delta_f1)

#Determination of inner and outer spiral angles F.2.7
#pinion
R_e21 = np.sqrt(R_m**2 + b_e**2 + 2 * R_m * b_e) #pinion
    cone distance of outer pinion boundary point
R_i21 = np.sqrt(R_m**2 + b_i**2 - 2 * R_m * b_i) #pinion
    cone distance of inner pinion boundary point
beta_e1 = np.arcsin((2 * R_m * r_c0 * np.sin(beta_m) - R_m
    **2 + R_e21**2)/(2 * R_e21 * r_c0)) #Wheel spiral angle
    at outer boundary point
beta_i1 = np.arcsin((2 * R_m * r_c0 * np.sin(beta_m) - R_m
    **2 + R_i21**2)/(2 * R_i21 * r_c0)) #Wheel spiral angle
    at inner boundary point

#wheel
beta_e2 = np.arcsin((2 * R_m * r_c0 * np.sin(beta_m) - R_m
    **2 + R_e21**2)/(2 *
                     R_e21 * r_c0)) #Outer wheel spiral angle
beta_i2 = np.arcsin((2 * R_m * r_c0 * np.sin(beta_m) - R_m
    **2 + R_i21**2)/(2 *
```

```
                    R_i21 * r_c0)) #Inner wheel spiral angle

#Determination of tooth depth F.2.8
h_ae1 = h_am1 + b_e * np.tan(theta_a1) #Outer mean addendum,
    pinion
h_fe1 = h_fm1 + b_e * np.tan(theta_f1) #Outer mean dedendum,
    pinion
h_ae2 = h_am2 + b_e * np.tan(theta_a2) #Outer mean addendum,
    wheel
h_fe2 = h_fm2 + b_e * np.tan(theta_f2) #Outer mean dedendum,
    wheel
h_e1 = h_ae1 + h_fe1 #Outer mean whole depth, pinion
h_e2 = h_ae2 + h_fe2 #Outer mean whole depth, wheel
h_ai1 = h_am1 - b_i * np.tan(theta_a1) #Inner mean addendum,
h_ai2 = h_am2 - b_i * np.tan(theta_a2) #Inner mean addendum,
h_fi1 = h_fm1 - b_i * np.tan(theta_f1) #Inner mean dedendum,
h_fi2 = h_fm2 - b_i * np.tan(theta_f2) #Inner mean dedendum,
h_i1 = h_ai1 + h_fi1 #Inner mean whole depth, pinion
h_i2 = h_ai2 + h_fi2 #Inner mean whole depth, wheel

#get minium normal backlash
def get_min_backlash(module, grade_group):
    # Define the table as ranges and corresponding backlash
       values
    table = [
        ((25.00, 20.00), (0.61, 0.81)),
        ((20.00, 16.00), (0.51, 0.69)),
        ((16.00, 12.00), (0.38, 0.51)),
        ((12.00, 10.00), (0.30, 0.41)),
        ((10.00, 8.00),  (0.25, 0.33)),
        ((8.00, 6.00),   (0.20, 0.25)),
        ((6.00, 5.00),   (0.15, 0.20)),
        ((5.00, 4.00),   (0.13, 0.15)),
        ((4.00, 3.00),   (0.10, 0.13)),
        ((3.00, 2.50),   (0.08, 0.10)),
        ((2.50, 2.00),   (0.05, 0.08)),
        ((2.00, 1.50),   (0.05, 0.08)),
        ((1.50, 1.25),   (0.03, 0.05)),
        ((1.25, 1.00),   (0.03, 0.05))
    ]

    # Loop through the table and find the matching range
    for (upper, lower), (val_4to7, val_8to12) in table:
        if upper >= module > lower:
            return val_4to7 if grade_group == 0 else
               val_8to12
```

```
    return None  # If module is out of defined range
j_en = get_min_backlash(m_et2,1)
print('Minimum normal backlash:', j_en)

## Determination of tooth thickness

#Thickness modification coefficient, pinion
x_sm1 = x_smn - j_en * 1 / (4 * m_mn * np.cos(alpha_n)) * (
    R_m * np.cos(beta_m))/(R_e * np.cos(beta_e2))

#Mean normal circular tooth thickness, pinion
s_mn1 = 0.5 * m_mn * np.pi + 2 * m_mn * (x_sm1 + x_hm1 * np.
    tan(alpha_n))

#Thickness modification coefficient, wheel
x_sm2 = - x_smn - j_en * 1 / (4 * m_mn * np.cos(alpha_n)) *
    (
    R_m * np.cos(beta_m))/(R_e * np.cos(beta_e2))

#Mean normal circular tooth thickness, wheel
s_mn2 = 0.5 * m_mn * np.pi + 2 * m_mn * (x_sm2 - x_hm1 * np.
    tan(alpha_n))

#Mean transverse circular thickness
s_mt1 = s_mn1 / np.cos(beta_m)
s_mt2 = s_mn2 / np.cos(beta_m)

#Mean normal diameter
d_mn1 = d_m1 / ((1 - np.sin(beta_m)**2 * np.cos(alpha_n)**2)
    * np.cos(delta_1))
d_mn2 = d_m2 / ((1 - np.sin(beta_m)**2 * np.cos(alpha_n)**2)
    * np.cos(delta_2))

#Mean normal chordal tooth thickness
s_mnc1 = d_mn1 * np.sin(s_mn1 / d_mn1)
s_mnc2 = d_mn2 * np.sin(s_mn2 / d_mn2)

#Mean chordal addendum
h_amc1 = h_am1 + 0.5 * d_m1 * np.cos(delta_1) * (1 - np.cos(
    s_mn1 / d_mn1))
h_amc2 = h_am2 + 0.5 * d_m2 * np.cos(delta_2) * (1 - np.cos(
    s_mn2 / d_mn2))

#Determination of remaining gear dimensions
R_e1 = R_m + b_e #Outer pitch cone distance, pinion
```

```
R_i1 = R_m - b_i #Inner pitch cone distance, pinion
d_ae1 = d_e1 + 2 * h_ae1 * np.cos(delta_1) #Outer pitch
    diameter, pinion
d_ae2 = d_e2 + 2 * h_ae2 * np.cos(delta_2) #Outer pitch
    diameter, wheel
d_fe1 = d_e1 - 2 * h_fe1 * np.cos(delta_1) #Inner pitch
    diameter, pinion
d_fe2 = d_e2 - 2 * h_fe2 * np.cos(delta_2) #Inner pitch
    diameter, wheel
d_ai1 = d_i1 + 2 * h_ai1 * np.cos(delta_1) #Inner pitch
    diameter, pinion
d_ai2 = d_i2 + 2 * h_ai2 * np.cos(delta_2) #Inner pitch
    diameter, wheel
d_fi1 = d_i1 - 2 * h_fi1 * np.cos(delta_1) #Outer pitch
    diameter, pinion
d_fi2 = d_i2 - 2 * h_fi2 * np.cos(delta_2) #Outer pitch
    diameter, wheel
t_xo1 = t_zm1 + b_e * np.cos(delta_1) - h_ae1 * np.sin(
    delta_1) #Pitch apex beyond crossing point, pinion
t_xo2 = t_zm2 + b_e * np.cos(delta_2) - h_ae2 * np.sin(
    delta_2) #Pitch apex beyond crossing point, wheel
t_xi1 = t_zm1 - b_i * np.cos(delta_1) - h_ai1 * np.sin(
    delta_1) #Pitch apex beyond crossing point, pinion
t_xi2 = t_zm2 - b_i * np.cos(delta_2) - h_ai2 * np.sin(
    delta_2) #Pitch apex beyond crossing point, wheel
h_t1 = (t_zF1 + t_xo1) / np.cos(delta_a1) * np.sin(theta_a1
    +
                                theta_f1) - (t_zR1 - t_zF1)
                                    * np.sin(delta_f1)

## Analysis of Forces ##
#Tangential Forces
F_mt1 = T_1 * 2000 / d_m1 #On pinion according to ISO23509
F_mt2 = F_mt1 #Same mean spiral angle
#Axial forces
F_ax1 = (np.tan(alpha_n)*np.sin(delta_1)/np.cos(beta_m)+np.
    tan(beta_m)*np.cos(delta_1))*F_mt1
F_ax2 = (np.tan(alpha_n)*np.sin(delta_2)/np.cos(beta_m)-np.
    tan(beta_m)*np.cos(delta_2))*F_mt1
#Radial force
F_rad1 = (np.tan(alpha_n)*np.cos(delta_1)/np.cos(beta_m)-np.
    tan(beta_m)*np.sin(delta_1))*F_mt1
F_rad2 = (np.tan(alpha_n)*np.cos(delta_2)/np.cos(beta_m)+np.
    tan(beta_m)*np.sin(delta_2))*F_mt1

E_modul = 206000 #MPa
```

```
sigma_Hlim = 1140 #N/mm^2

#SS 1862
v_mt1 = d_m1 * n1_rad / 2000
K_1 = 1.75 #Load factor from tabel 1
K_v1 = np.sqrt((78 + 14 * np.sqrt(v_mt1))/78)
F_ber = F_mt1 * K_1 * K_v1

alpha_t = np.arctan(np.tan(alpha_n)/np.cos(beta_m)) #
    Effective pressure angle in transverse section according
    to inventor
Z_H = np.sqrt(2*np.cos(beta_m)/np.sin(2*alpha_t)) #Form
    factor for contact stress at the rolling point
Z_K = 2 * R_e / (R_e + R_i) #Cone length factor for contact
    stress
Z_M = np.sqrt(0.35*E_modul) #Material factor
Z_epsilon = 1 #Contact ratio factor
K_Halpha = 1
K_Hbeta = 1.25
#Load-carrying capacity with respect to flank pressure
    accroding to Hertz
sigma_H = Z_H * Z_K * Z_M * Z_epsilon * np.sqrt(F_ber *
                K_Halpha * K_Hbeta * np.sqrt(u**2 + 1)/(b *
                   d_m1 * u))

#allowed flank pressure

K_L = 1 #lubrication factor
Z_R = 1 #ytjaemnhetsfaktor
Z_V = 1 #hydrodynamisk smoerjfilmsfaktor
K_HX = 1 #dimensionsfaktor foer yttryck
K_HN = 1 #livslaengdsfaktor foer yttryck
K_HK = 1 #haerdhetskombinationsfaktor
S_H = 1 #safty factor for flank pressure

sigma_HP = sigma_Hlim * K_L * Z_R * Z_V * K_HX * K_HN * K_HK
    / S_H
while sigma_H <= sigma_HP:
    S_H += 0.0001
    sigma_H = Z_H * Z_K * Z_M * Z_epsilon * np.sqrt(F_ber *
                    K_Halpha * K_Hbeta * np.sqrt(u**2 + 1)/(
                       b * d_m1 * u))
    sigma_HP = sigma_Hlim * K_L * Z_R * Z_V * K_HX * K_HN *
       K_HK / S_H
```

```
x_vn1 = x_hm1 / np.cos(beta_m) #Apparent profile shift
    coefficient of the virtual gear
z_v1 = z_1 * np.sqrt(1 + u**2) / u #Virtual number of teeth
    pinion
z_v2 = z_2 * np.sqrt(1 + u**2) #Virtual number of teeth
    wheel
beta_bm = np.arctan(np.tan(beta_m) * np.cos(alpha_t)) #base
    helix angle at the mean radius
z_vn1 = z_v1 / (np.cos(beta_m) * np.cos(beta_bm)**2)
z_vn2 = z_v2 / (np.cos(beta_m) * np.cos(beta_bm)**2)

Y_F = 2.2 #tooth form factor for bending from tabel 8.4
Y_K = (2 * R_e / (R_e + R_i))**2 #tooth lenght coefficient
if beta_m <= np.deg2rad(30): #helical factor
  Y_beta = 1 - np.rad2deg(beta_m) / 120
else:
  Y_beta = 0.75

Y_epsilon = 1 #Contact ratio factor for bending
K_Falpha = 1 #Transverse load factor for bending stress
K_Fbeta = K_Hbeta #Face load factor for bending stress

sigma_F = Y_F * Y_K * Y_beta * Y_epsilon * (F_ber *
                K_Falpha * K_Fbeta) / (b * m_mn) #calculated
                    bendingstress
sigma_Flim = 316
Y_s = 1
S_F = 1 #start value
K_FX = 1
K_FN = 1
sigma_FP = sigma_Flim * Y_s * K_FX * K_FN / S_F

while sigma_F <= sigma_FP:
    S_F += 0.0001
    sigma_F = Y_F * Y_K * Y_beta * Y_epsilon * (F_ber *
                K_Falpha * K_Fbeta) / (b * m_mn) #calculated
                    bendingstress
    sigma_FP = sigma_Flim * Y_s * K_FX * K_FN / S_F


parameters = [
    "u_target", "sigma_large", "sigma_large_rad", "z_1", "
       d_e2", "n2", "P", "R_e", "pressure_angle",
    "z_2", "u", "n1", "n1_rad", "delta_1", "delta_2", "
       alpha_d", "d_e1", "m_et2", "b_met", "b_Re", "b",
```

```
    "R_i", "R_m", "K_Z", "beta_m", "epsilon_beta", "
       beta_m_deg", "d_i1", "d_m1", "d_i2", "d_m2", "m_mn",
    "alpha_lim", "alpha_n", "alpha_e", "t_zm2", "t_zm1", "
       t_z1", "t_z2", "T_1", "k_hap", "k_hfp", "k_d",
    "k_c", "u_a", "c_ham", "k_t", "x_hm1", "x_smn", "c_be",
       "b_e", "b_i", "r_c0", "h_mw", "h_am2",
    "h_fm2", "h_am1", "h_fm1", "c", "h_m", "theta_sum_fs", "
       theta_a2", "theta_f2", "theta_sum_fs_deg",
    "theta_a2_deg", "theta_f2_deg", "delta_a2", "delta_f2",
       "delta_a1", "delta_f1", "theta_a1",
    "theta_f1", "t_zF2", "t_zR2", "t_zF1", "t_zR1", "R_e21",
        "R_i21", "beta_e1", "beta_i1", "beta_e2",
    "beta_i2", "h_ae1", "h_fe1", "h_ae2", "h_fe2", "h_e1", "
       h_e2", "h_ai1", "h_ai2", "h_fi1", "h_fi2",
    "h_i1", "h_i2", "j_en", "x_sm1", "s_mn1", "x_sm2", "
       s_mn2", "s_mt1", "s_mt2", "d_mn1", "d_mn2",
    "s_mnc1", "s_mnc2", "h_amc1", "h_amc2", "R_e1", "R_i1",
       "d_ae1", "d_ae2", "d_fe1", "d_fe2", "d_ai1",
    "d_ai2", "d_fi1", "d_fi2", "t_xo1", "t_xo2", "t_xi1", "
       t_xi2", "h_t1", "F_mt1", "F_mt2", "F_ax1",
    "F_ax2", "F_rad1", "F_rad2", "E_modul", "sigma_Hlim", "
       v_mt1", "K_1", "K_v1", "F_ber", "alpha_t",
    "Z_H", "Z_K", "Z_M", "Z_epsilon", "K_Halpha", "K_Hbeta",
        "sigma_H", "K_L", "Z_R", "Z_V", "K_HX",
    "K_HN", "K_HK", "S_H", "sigma_HP", "x_vn1", "z_v1", "
       z_v2", "beta_bm", "z_vn1", "z_vn2", "Y_F",
    "Y_K", "Y_beta", "Y_epsilon", "K_Falpha", "K_Fbeta", "
       sigma_F", "sigma_Flim", "Y_s", "S_F",
    "K_FX", "K_FN", "sigma_FP"
]

value_list = [
    u_target, sigma_large, sigma_large_rad, z_1, d_e2, n2, P
       , R_e, pressure_angle,
    z_2, u, n1, n1_rad,
    np.rad2deg(delta_1), np.rad2deg(delta_2), np.rad2deg(
       alpha_d),
    d_e1, m_et2, b_met, b_Re, b,
    R_i, R_m, K_Z,
    np.rad2deg(beta_m), epsilon_beta, beta_m_deg,
    d_i1, d_m1, d_i2, d_m2, m_mn,
    np.rad2deg(alpha_lim), np.rad2deg(alpha_n), np.rad2deg(
       alpha_e),
    t_zm2, t_zm1, t_z1, t_z2,
    T_1, k_hap, k_hfp, k_d, k_c, u_a, c_ham, k_t, x_hm1,
       x_smn,
```

```
    c_be, b_e, b_i, r_c0, h_mw, h_am2, h_fm2, h_am1, h_fm1,
    c, h_m,
    np.rad2deg(theta_sum_fs), np.rad2deg(theta_a2), np.
        rad2deg(theta_f2),
    theta_sum_fs_deg, theta_a2_deg, theta_f2_deg,
    np.rad2deg(delta_a2), np.rad2deg(delta_f2), np.rad2deg(
        delta_a1), np.rad2deg(delta_f1),
    np.rad2deg(theta_a1), np.rad2deg(theta_f1),
    t_zF2, t_zR2, t_zF1, t_zR1,
    R_e21, R_i21,
    np.rad2deg(beta_e1), np.rad2deg(beta_i1), np.rad2deg(
        beta_e2), np.rad2deg(beta_i2),
    h_ae1, h_fe1, h_ae2, h_fe2, h_e1, h_e2,
    h_ai1, h_ai2, h_fi1, h_fi2, h_i1, h_i2,
    j_en, x_sm1, s_mn1, x_sm2, s_mn2, s_mt1, s_mt2, d_mn1,
        d_mn2,
    s_mnc1, s_mnc2, h_amc1, h_amc2, R_e1, R_i1,
    d_ae1, d_ae2, d_fe1, d_fe2, d_ai1, d_ai2, d_fi1, d_fi2,
    t_xo1, t_xo2, t_xi1, t_xi2,
    h_t1, F_mt1, F_mt2, F_ax1, F_ax2, F_rad1, F_rad2,
    E_modul, sigma_Hlim, v_mt1, K_1, K_v1, F_ber,
    np.rad2deg(alpha_t),
    Z_H, Z_K, Z_M, Z_epsilon,
    K_Halpha, K_Hbeta, sigma_H, K_L, Z_R, Z_V, K_HX, K_HN,
        K_HK, S_H, sigma_HP,
    x_vn1, z_v1, z_v2,
    np.rad2deg(beta_bm),
    z_vn1, z_vn2,
    Y_F, Y_K, Y_beta, Y_epsilon,
    K_Falpha, K_Fbeta, sigma_F, sigma_Flim, Y_s, S_F, K_FX,
        K_FN, sigma_FP
]


# Create the DataFrame
data = {
    'Parameter': parameters,
    'Value': value_list,
}

df = pd.DataFrame(data)

# Formatting options for better readability
pd.set_option("display.precision", 4)

# Display the DataFrame
```

```
print(df)
df
```

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS