NTNU
Kunnskap for en bedre verden

Department of Structural Engineering

Master's thesis - TKT4950

Contributing to an open-source python package for concrete design

Covering shear for Model code 2010 and testing strengths and weaknesses

Authors:

Johan Røstad Brøndbo

Greger Alexander Wulfsberg

Date 11.06.2023

Department of Structural Engineering
Faculty of Engineering
NTNU- Norwegian University of Science and Technology

ACCESSIBILITY

MASTER THESIS 2023

SUBJECT AREA:	DATE:	NO. OF PAGES:
Concrete structures	11.06.2023	64

TITLE:

Contributing to an open-source python package for concrete design

Bidrag til en open-source python-pakke for betongkonstruksjoner

BY:

Johan Røstad Brøndbo & Greger Alexander Wulfsberg

SUMMARY:
Data programs is starting to become a big part of structural engineers every day. However, as of now, few open-source projects have been made to help replicating material behavior in a bigger scale. This is where fib now works on creating an open-source python package in GitHub for concrete design. This project can be of big help in saving time and increase the accuracy of how a structural engineer work. At the same time is awareness of how the code works and the consequences of using it.

This thesis takes upon the implementing of Model code 2010 chapter 7.3.3 - 7.3.5, including most of the shear equations. The equations are implemented as functions in the python package including a doc-string to elaboration for the inputs and objective of the function. The code was made with corresponding tests to minimize the possibility of errors. To further validate the implemented code, a benchmarking of the shear capacity without shear reinforcement was conducted. Here we analyzed how close different approximation methods in Model code 2010 and Eurocode 2 were to the actual capacity. There were also done research on the standard deviation of the different methods as well as changing parameters in the standard to see how it would affect the result. There are in total four files we have made, including plain shear, torsion, different casting time and punching. Each of these files also have corresponding tests.

A large part of the thesis is the benchmarking done for plain shear without shear reinforcements. This covers a small part of code, but can help users make an insightful choice when deciding to use a standard. Here we illustrate how much more conservative an approximation method is compared to another. We have also compared Model code with both Eurocode 2 - 2004 edition and the new Eurocode 2 that is soon to be released. Here we found weakness in the Model code 2010 when it comes to high strength concrete based on our tests. There can also be seen a trend of more conservative results when calculating with Model code 2010 compared to Eurocode 2

RESPONSIBLE TEACHER: Morten Engen

SUPERVISOR(S): Reignard Tan and Morten Engen

CARRIED OUT AT: Department of Structural Engineering, NTNU

Sammendrag

Dataprogrammer begynner å bli en større del av bygningsingeniørers hverdag. Foreløpig finnes det få open-source prosjekter som tar for seg materialbruk i stor skala. Derfor arbeider *fib - International Federation for Structural Concrete* for å lage en open-source python pakke i GitHub for betongkonstruksjoner. Et slikt prosjektet kan være til stor hjelp i å spare ingeniører tid, samtidig som det kan øke nøyaktigheten i arbeidet. Det er samtidig viktig å formidle forståelse for hvordan koden til programmet virker, og konsekvensene av måten en bruker programmet på.

Denne masteroppgaven tar på seg jobben å implementere de fleste skjærformlene fra Model code 2010 (kapitel 7.3.3-7.3.5). Formlene er laget i python med funksjoner og en doc-string med tekst til hver funksjon som forklarer hva funksjonen tar inn av variabler og hva den gjør. Funksjonene har tilsvarende tester som skal redusere risikoen for feil. For å vurdere koden som er skrevet er det gjort videre sammenligninger og undersøkelser av koden som dekker betong uten skjærarmering. Her undersøker vi hvor nærme de forskjellige tilnærmingsnivåene for Model code 2010 og Eurocode 2 er til den virkelige kapasiteten. Det blir sett på standardavvik for resultatene og hva en eventuell endring i Model code 2010 vil kunne ha å si for resultatene. Det er totalt laget fire filer, dette inkluderer ren skjær, torsjon, ulik settingstid og gjennomlokking. Disse filene har egne tester som korresponder til hver av funksjonene.

En stor del av oppgaven vår er referansemålingen som dekker skjær uten skjærarmering. Det dekker en liten del av kodens helhet i prosjektet, men forhåpentligvis hjelper referansemålingene eventuelle brukere å få innsikt i standarene og kodebruken. På den måten kan brukerne av koden gjøre opplyste valg når de tar i bruk koden. Her illustrerer vi hvor mye mer konservativ en tilnærmingsmetode er sammenlignet med en annen. Vi har også sammenlignet Model code 2010 med både Eurocode 2 - 2004-utgaven og den nye Eurocode 2 som snart skal lanseres. Her fant vi svakheter i Model code 2010 når det kommer til høytrykkfast betong basert på våre tester. Det kan også sees en trend med mer konservative resultater ved beregning med Model code 2010 sammenlignet med Eurokode 2 og den nyere utgaven av Eurokode 2.

Abstract

Data programs is starting to become a big part of structural engineers every day. However, as of now, few open-source projects have been made to help replicating material behvior on a bigger scale. This is where fib - *International Federation for Structural Concrete* now works on creating an open-source python package in GitHub for concrete design. This project can be of big help in saving time and increase the accuracy of how a structural engineer work. At the same time, it is important to be aware of how the code functions and the potential consequences of its utilization.

This thesis takes upon the implementing of Model code 2010 chapter 7.3.3 - 7.3.5, including most of the shear equations. The equations are implemented as functions in the python package containing a doc-string to elaboration for the inputs and objective of the function. The code was made with corresponding tests to minimize the possibility of errors. To further validate the implemented code, a benchmarking of the shear capacity without shear reinforcement was conducted. Here we analyzed how close different approximation methods in Model code 2010 and Eurocode 2 were to the actual capacity. There were also done research on the standard deviation of the different methods as well as changing parameters in the standard to see how it would affect the result. There are in total four files we have made, including plain shear, torsion, different casting time and punching. Each of these files also have corresponding tests.

A large part of the thesis is the benchmarking done for plain shear without shear reinforcements. This covers a small part of code, but can help users make an insightful choice when deciding to use a standard. Here we illustrate how much more conservative an approximation method is compared to another. We have also compared Model code 2010 with both Eurocode 2 - 2004 edition and the new Eurocode 2 that is soon to be released. Here we found weakness in the Model code 2010 when it comes to high strength concrete based on our tests. There can also be seen a trend of more conservative results when calculating with Model code 2010 compared to Eurocode 2 and the new version of Eurocode 2.

Preface

With a contribution of 3190 lines of code and over 300 commits, this concludes our master in Civil and Environmental engineering. The thesis was carried out for the department of structural engineering at the Norwegian university of Science and Technology, spring 2023. Even though the topic of programming is something we have done in a lower scale for a few of our subjects, it peeks our interest. To be able to join a large project as this one has been instructive for learning more about cooperating in git, coding in python and learning more about Model code 2010. It will be interesting to see further development and progression in the project.

A large motivation for this thesis was to become better at coding as well as to work in a bigger project in Git. We have never been a part of a project of this scale and wanted to learn how this type of work was carried out. At the same time as we wanted to develop our own abilities can we take pride in contribute to a project that aim at improving the construction industry.

During the work on our thesis, we have been supervised by Morten Engen and Reignard Tan and we would like to thank both of them for assistance and guidance during our meetings as well as being available when sending mail and feedback on our work during this period. This has been really helpful.

Table of Contents

1 Introduction

1.1 Motivation and background

Concrete has long been known for structural strength in compression, but have also a bad reputation regarding its CO_2 emission. The UN sustainability goals are set for 2030 where the thirteenth goal is to stop the climate changes. It is then imperative to cut climate changes where possible. A huge responsibility lies on the structural industry where concrete alone produces 8% of the global emissions [1]. With an workflow that relies more and more on computer programs will the need to stop and ensure structural safety and environmentally safety be essential. This thesis takes on the task of both: help the development of technology and enquiring the information to evaluate the technology advancements.

Open-source software is becoming increasingly popular among developers and businesses around the world. A survey done by WhiteSource [2] software shows that over 70% of companies use open-source in some ways. In addition, GitHub reports that 50% of developers contribute to open-source projects on GitHub, this represents more than 200 million repositories. Furthermore 99% of new software project rely on a part of open-source contribution. Open-source repositories also receive ten times more contributions than private ones [2].

In the field of structural engineering, traditional methods such as writing boards and calculations by hand were dominating for a long time. In the later years the development of BIM as well as the usage of different programs for mechanical calculations and structural analysis have been more popular. An area of potential improvement lies in the utilization of the hand-held laws and regulations called *standard*. What if the standards were more user friendly and accessible through a digital platform, such as an open-source python package?

One of the primary advantages of developing an open-source python package is the potential of significant time efficiency. Structural engineers would no longer need to manually search for relevant sections within the printed standards. Instead, they would navigate through the package and quickly accessing the information needed. There would be no need to implement equations in a program since they already exist in the python package. The workflow would allow the engineers to focus more on their main task such as analyzing

and designing structures, rather than spending time searching in the standard.

Another advantage is the potential for increased accuracy. Errors coming from misinterpretation or transcription can occur when using a manual copy of the standard. By providing a standardized and easily accessible python package for the standard, the risk of errors can be reduced. Engineers can then rely on their accuracy and consistency, improving the quality of their design and analysis.

Furthermore, an open-source python package has the potential to promote collaboration and knowledge sharing among structural engineers all over the world. To be able to access and contribute to the package, professionals with different backgrounds and from different regions could come together and make improvements. The collective effort could lead to a more updated standards to work with, as well as fostering a community, where engineers can share their ideas, experiences and provide feedback to continuously improve the package.

However, despite some potential advantages, an open-source python package can also cause some challenges. Adapting to a new way of working by using python and its functionalities to access attributes and functions used in the standard for concrete. This may require training and support to ensure a good transition. Additionally, some engineers may initially resist to the changes as they prefer their old way of working by using the hand-held standard.

1.2 Scope and limitations

The aim for this thesis is to contribute towards the large project by "International Federation for Structural Concrete" fib, on the development of an open-source python package. The package will consist of the formulas and models included in Eurocode 2 [3] and fib Model code 2010 [4]. The project has a long-term goal, and the work done in this thesis will be a contribution to the project. The final package will be available for everyone to use when it is done and the workflow is an open-source. The final project can be found by following this link: https://github.com/fib-international/structuralcodes/

The contribution on the thesis will cover the formulas in Model code 2010 for shear, specifically, chapter 7.3.3 - 7.3.5. This includes shear with and without shear reinforcement, torsion, punching shear and casting at different times. The thesis will not include prestressing. Model code 2010 is based on a "Level Of Approximation" LoA approach, which allows the user to specify the level of accuracy of the calculation based on the amount of input that is given. The approximation levels range from 1-4, whereas our contribution to the project covers LoA 1-3.

To ensure validation of the code made, we decided to add benchmarking to the thesis. This is done by comparing previous shear experiments [5] with Model code 2010 and Eurocode 2 as well as the newer version of Eurocode 2 which is not published yet. The goal for the benchmarking is to see if we can find any weakness in our code by sorting shear-data with internal moment arm, external moment, characteristic strength and load distance from support. We will then give an overall evaluation of the given standards as well as giving advice for improvements in the code.

2 Theoretical Framework

In this chapter we cover theory and background used in result and discussion. Concrete is large part of the thesis described in chapter 2.1 with some information about the material, environmental aspect, concrete impressive behaviour and high strength concrete. Furthermore we cover theory about shear to understand more about the shear tests we have based our thesis on. This is described in chapter 2.2. Standards are also an important aspect in our work, here we cover some information about the standards as well as the equations used in our benchmarks later in the results. This can be found in chapter 2.3. Digital development is described in chapter 2.4 and is our coding foundation, with the purpose of giving the reader more understanding of the coding done and what it means to work in a open-source project.

2.1 Concrete

Concrete is one of the most used materials in the building industry. The material is cheap and highly used worldwide because of its compression strength. Its composition consists of water, cement and aggregates. It normally consists of 65-75 % aggregates, 14-20 % water and 10-20 % cement as well as some percent air and admixtures. Cement combined with water are often called cement paste, which is mainly determined by the mass ratio between the water and cement or the w/c ratio. Often, instead of using a special concrete, it is possible to change the properties of the concrete by incorporating a suitable additive or an admixture. Admixtures are chemical agents added in small doses in the mixing stage to improve specific properties of the concrete, like an accelerator which accelerates the hardening phase or set-retarders which delay the setting time of the concrete. Additives are supplements to the concrete added in the cement manufacturing stage, usually added in larger doses. The most important additive in Norway is silica fume and fly ash. These are also called *pozzolans*. Pozzolans are often included in the mass ratio, m = w/(c + k*p) where k is the efficient factor for the actual property. [6]

The material is a heterogeneous material, which indicates that the property of the material is not consistent through the cross section. As a direct cause it is harder to replicate a test of the material. Concrete without reinforcement often fails in a brittle failure and has low tensile and strain capacity. A revolutionary improvement came when steel bars were added, allowing the bars to take the tensile forces. Reinforced concrete (RC) was then an

alternative to other materials capable of handling bending [7].

The environmental aspect of concrete becomes rapidly more important when it is the most common building material in the world. Cement production is the key element that produces CO2, where 2.5 billion tons are produced every year. This result comes from not taking into account the emissions from heating the raw materials (limestone), otherwise the overall emissions would be even higher. On a global basis, 2.5 billion tons make up 8% of all emissions. To produce this much cement, a large amount of water is needed. It is estimated that 10% of all industrial water is consumed in cement production [1].

2.1.1 Concrete behavior

The compressive strength of concrete is the most important trait. It is tested in two ways. The first test is to apply a force on a cube that is 100mm in all directions to get $f_{cm,cube}$. This number is the force divided on the side of the cube, which is $0.001m^2$. The second way of performing is the more common way, where one tests how much a cylinder with 100/150mm diameter and 200/300mm height can handle. This force divided on the cross section (πr^2) is then f_{cm}. Then subtracting 8 MPa to ensure sufficient strength. This would lead to a result that is conservative 95% of the time. The strength we end up with after subtracting is the characteristic strength, f_{ck}. f_{ck} is often given in MPa or N/mm^2

With f_{ck}, we have strength that holds 95% of the time. To take into account for long term and unfavorable effects, a normal formula to use is equation 1. Here α_{cc} takes into account the long term effects on the concrete strength and unfavourable effects on the way the load is applied while γ_c is the partial safety factor. Normally $\alpha_{cc} = 0.85$ and γ_c =1.5.

$$f_{cd} = \alpha_{cc} \cdot \frac{f_{ck}}{\gamma_c} \tag{1}$$

The flexural capacity of a reinforced concrete beam depends on the material property and geometrical property of the concrete and reinforcement. The strain distribution is linear for bending as long as plain sections remain plain. When loading a reinforced concrete beam, it will pass through three different states. These states describe the stress distribution across a section that is uncracked, cracked or close to failure in the ultimate state [8].

2.1.2 High strength concrete

The main intention of Model code 2010 is to contribute to the development of improved design methods and the use of improved structural materials, like *high strength concrete* HSC. HSC is described in Mode code 2010 as concrete with $fck > 50MPa$. The compressing strength regarding this value according to the Model code 2010 is measured on 150/300 mm cylinders in accordance to ISO 1920-3. [9]

In the thesis by Hallgren [8], there is a list summarizing the main differences between normal strength concrete and high strength concrete. The first observation described a more linear stress strain relationship to a higher percentage of the max stress, and that the high strength concrete reaches slightly higher strain at maximum stress. There was also observed a steeper slope after reaching maximum stress in the stress-strain diagram. Lastly, the HSC would have a lower ultimate strain. All these observations can be seen in figure 1.

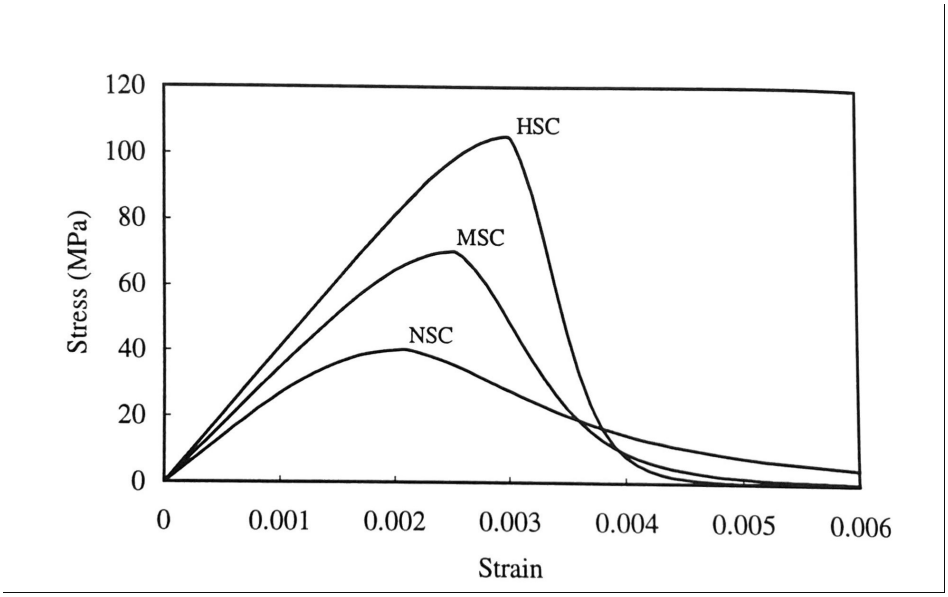


Figure 1: Shows typical stress-strain curves. With normal, medium and high strength concrete. From: [8]

The difference in stress-strain behavior with increasing strength also indicates a possible disadvantage with high strength concrete. The reduced ultimate strain combined with the more brittle behavior reduces the ductility of the concrete, i.e. the deformation capacity will be lower for high strength concrete.

2.2 Shear

In construction, it is crucial to ensure that the external loads on a concrete beam or slab remain below a critical point to avoid collapse or failure. If this critical point is exceeded, it can have fatal consequences for civilians and infrastructure. To fully understand the structural response of a construction, it is necessary to consider all possible combinations of the moment-, axial- and shear forces.

Development of a comparable theory for shear strength of reinforced concrete members has been a goal of a worldwide research effort for more than 40 years. A variable truss model based on a lower bound theory of plasticity has been developed and is incorporated in Eurocode 2 regarding members with significant shear reinforcement. For the potentially more dangerous case without shear reinforcement, Eurocode 2 uses a totally empirically procedure which is not supported by theory. There are two ways that shear can be transferred in a reinforced member: through flexural tension and compression forces. [10]

When talking about shear we need a load that is perpendicular to the beam/slab, or at least to some degree. This perpendicular force is called the shear force and often described as V. V can be derived as the perpendicular component of the force if the force is between 0 and 90 degrees. The shear stresses, τ, becomes most relevant as it is the shear force divided by the cross-sectional area. This relationship can be observed in figure 2 .

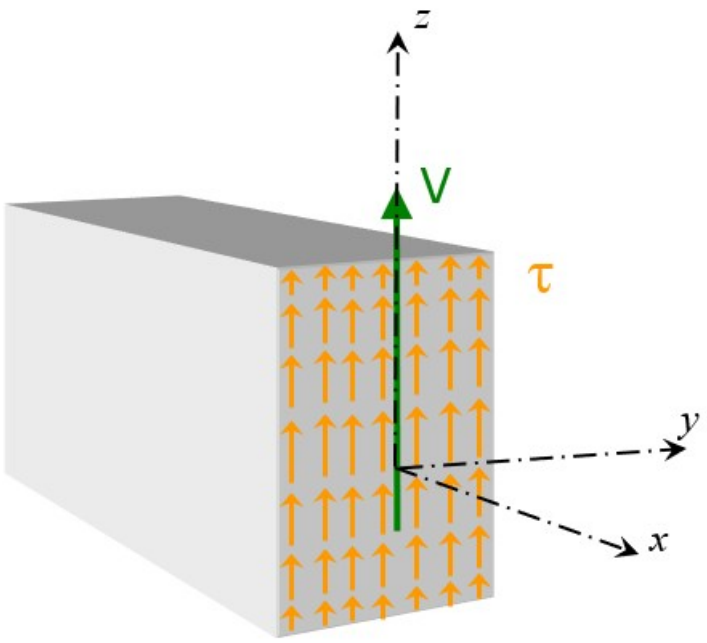


Figure 2: Shear load (V) and shear stresses (τ).*From* : [11]

2.2.1 A typical shear test

The span between the two-point loads is subjected to pure constant moment, whereas the span between support and point load is subjected to a constant shear with a linearly varying moment. A clear disadvantage by using this method is that the moment is changing along the shear span. This can make it unreliable to use the result of this test to develop a general theory for shear behavior [12]. Figure 3 shows a traditional four-point shear test.

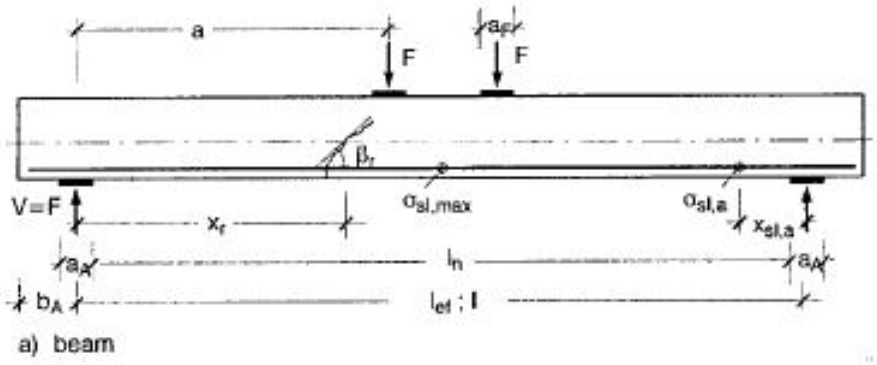


Figure 3: Typical shear test. From: [12]

With a shear test setup as shown in figure 3, the axial, shear and moment work differently in the different parts of the beam. There will be no axial forces if the support is pinned and roller, see figure 4.

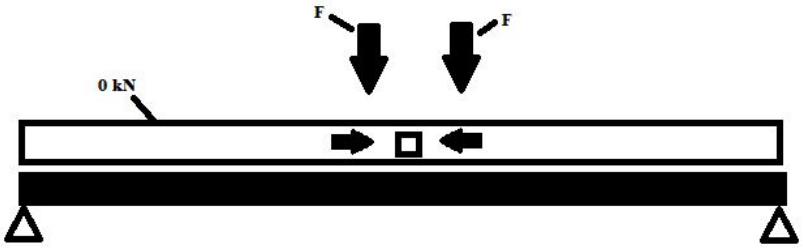


Figure 4: The axial forces in the beam

The shear forces will be the same between the support and load, as shown in figure 5.

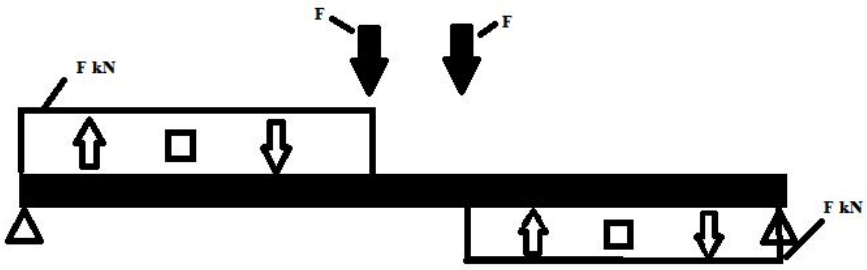


Figure 5: The shear forces in the beam

Even though figure 5 shows a constant shear force between the support and load can we expect the shear forces to be reduced close to the support. The moment is affected by the length of the beam and can thus be reduced by using a shorter beam, as shown in figure 6.

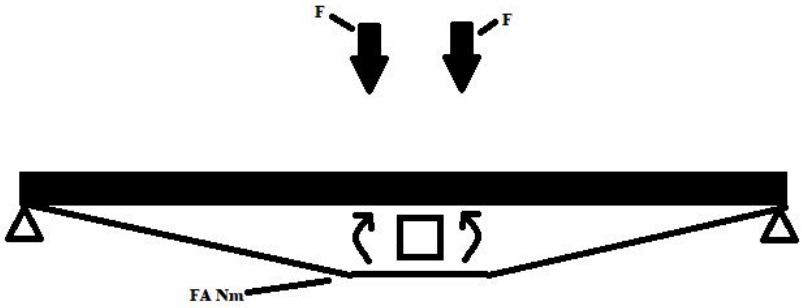
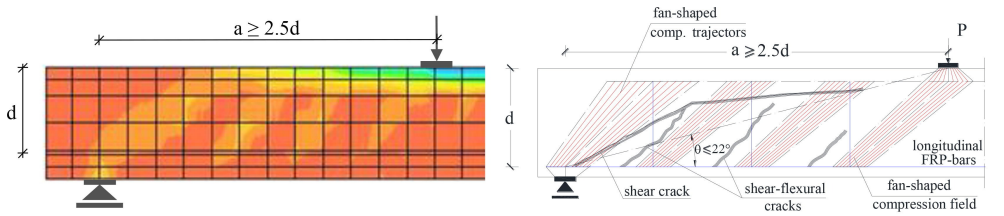


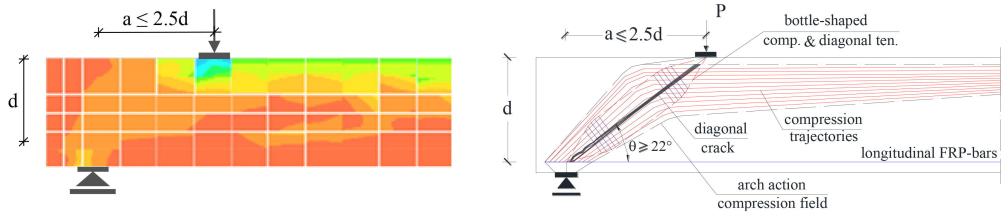
Figure 6: The moment in the beam where A is the distance from the support to the load

2.2.2 Factors that affect test performance

In a typical shear test described in figure 3 the beam is subjected to force applied parallel to the cross-section. The support conditions is a big indication on the accuracy and reliability of the test results. The effective depth ratio is defined as the distance between the load and support (a) divided by height from the reinforcement to the top of the beam (d). If (a/d) is less than 2.5, a strut starts to form between the support and the load, see figure 7. The normal shear formulas are given for scenarios where the (a/d) is higher than 2.5 [13].



a. shear mechanism for $a \geq 2.5d$



b. shear mechanism for $a \leq 2.5d$

Figure 7: Stresses and crack patterns for load close and far away from support. From: [13]

To transfer the forces that the concrete experiences to the reinforcement, a form of connection needs to take place. This is where bond and *anchorage* are used so solve the task with two different methods; *Bond* gradually takes the forces with friction along the surface of the reinforcement and anchorage to handle all the force from an anchor. The goal is to base the design rules on well-established results and calculation from science, but the complexity can often make it difficult to find the bond exact strength for the given case. Model code 2010 have hence derived simplified calculation rules to help the results become more reliable. For the bond method a lot of parameters come into play. The surface of the reinforcement η_1, the casting conditions η_2, bar diameter η_3 and the characteristic strength of steel η_4 are all used in Model code 2010 equation 6.1-20 to find the bond strength $f_{bd,0}$ [14]:

$$f_{bd,0} = \eta_1 \eta_2 \eta_3 \eta_4 \frac{(\frac{f_{ck}}{25})^{0.5}}{\gamma_{cb}} \tag{2}$$

2.3 Standards

Laws and regulation for the building industry, often called standards, are a common recipe for how something should be created or carried out [15]. They are everywhere in our day life, all from the text that this thesis is written in to a phone used to call with. In construction context, it is often how to calculate the capacity of a construction or a member.

2.3.1 Model code 2010

fib Model code 2010 [4] is a comprehensive code about concrete with the objective to serve as a base for other codes, and presents up-to-date research and developments to strive for the most optimal use of concrete. The code covers the whole life cycle of concrete, from design and assessment to maintenance and dismantling. It is made by the international federation for structural concrete fib, which is a not-for-profit organization.

The code consists of different levels of approximation. This is to give the required accuracy for the situation that the engineering projects are at. The approximation levels are sorted so that a higher level gives a more accurate result, but in return also need more advanced calculations and inputs. In a preliminary stage of a project or to test if a beam has sufficient strength, a low level of approximation may be all that is needed. However, if the stresses are close to the capacity or a demand of higher material efficiency, a higher level of approximation would be the recommended calculation [16].

To ensure a safety in choosing a lower level of approximation, a simplification of a formula is done with conservative assumptions. Due to this, we get consistent conservative answers when using lower level of approximation, when given less information about the material and cross section. There are situations where the demand for concrete will be the same, but most of the time the concrete use will increase. This also applies to the shear formulas in Model code 2010 [17].

Shear is covered in section 7.3.3 in Model code 2010, where equation 3 tells us that the shear resistance (V_{Rd}) is the shear resistance contributed from the concrete ($V_{Rd,c}$) plus the contribution from the shear reinforcement ($V_{Rd,s}$). This shear resistance must be greater or equal to the shear force (V_{Ed}).

$$V_{Rd} = V_{Rd,c} + V_{rd,s} \geq V_{Ed} \tag{3}$$

Furthermore, a lot of the formulas request the strain to be calculated for the shear resistance to be determined. This is longitudinal strain ϵ_x and formula 4 calculates it at the mid depth of the effective shear depth. E_s is the E-modulus to the reinforcement, while M_{ed}, V_{ed} and N_{ed} are the moment, shear force and axial force working in the cross section. δe is the eccentricity of the axial load due to imperfections in the construction.

$$\epsilon_x = \frac{1}{2E_s A_s}\left(\frac{M_{Ed}}{z} + V_{Ed} + N_{ed}\left(\frac{1}{2} \pm \frac{\delta e}{z}\right)\right) \tag{4}$$

Formulas for cross sections without shear reinforcement are covered in the Model code 2010 in chapter 7.3.3.2. Here we only have the shear resistance ($V_{Rd,c}$) because we do not have shear reinforcement. This is calculated from equation 5.

$$V_{Rd,c} = k_v \frac{\sqrt{f_{ck}}}{\gamma_c} z b_w \tag{5}$$

k_v is calculated based on what approximation level you choose. For approximation level 1, see equation 6. For approximation level 2, see equation 7 and 8.

$$k_v = \frac{180}{1000 + 1.25z} \tag{6}$$

$$k_{dg} = \frac{32}{16 + dg} \geq 0.75 \tag{7}$$

$$k_v = \frac{0.4}{1 + 1500\epsilon_x} \cdot \frac{1300}{1000 + k_{dg} z} \tag{8}$$

Here f_{ck} is the characteristic strength of concrete in MPa. γ_c is the material factor, often set to 1.5. z is the distances between the centerline of the compressive chord and the reinforcement. b_w is the width of the web of the cross section. dg is the maximum size of aggregate.

For equation 5, $\sqrt{f_{ck}}$ should not be greater than 8 MPa. If the concrete strength is greater than 70 MPa, dg in equation 7 shall be set to 0. The equation then becomes:

$$k_{dg} = \frac{32}{16+0} \geq 0.75 = 2 \tag{9}$$

2.3.2 Eurocode 2

Eurocode 2: Design of concrete structures [3], is a European standard that is widely known and commonly used when designing buildings. For constructions that do not have shear reinforcement, formula 6.2.a in 6.2.2 is given:

$$V_{Rd,c} = [C_{Rd,c}k(100\rho f_{ck})^{1/3}]b_w d \tag{10}$$

Here d is the distance from the reinforcement to the other edge in mm. $C_{rd,c}$ is given in equation 11, k in equation 12 and ρ in equation 13.

$$C_{Rd,c} = \frac{0.18}{\gamma_c} \tag{11}$$

$$k = 1 + \sqrt{\frac{200}{d}} \leq 2 \tag{12}$$

$$\rho = \frac{A_{sl}}{b_w d} \leq 0.02 \tag{13}$$

The work of producing a new Eurocode 2 is in progress [18]. Calculation for shear capacity is described in equation 8.27 in the new version of Eurocode 2 and is the following equation:

$$\tau_{Rd,c} = \frac{0.66}{\gamma_v} \cdot (100\rho_1 \cdot f_{ck} \cdot \frac{d_{dg}}{d})^{\frac{1}{3}} \tag{14}$$

Here it is important to note that equation 14 finds the allowed shear stress and not shear force. d_{dg} is the same as dg mentioned earlier. γ_v is not 1.5, as often used as material factor in concrete, but 1.4.

2.4 Digital development

2.4.1 Open-source

An open-source software refers to a software with a license that grants the user freedom to use, modify and distribute the software for any purpose [19]. This means that the source code is made public allowing anyone to modify and contribute to the software. This might be making the number of online developers or contributors infinite, this is called open collaboration. [19].

One of the principles of open-source software is the concept of a non-cost software, making the distribution to others free. This allows rapid development, flexibility and customization of the software. The typical open-source project utilizes a large amount of people to evaluate and improve code quality. The diversity of participants is important to this kind of collaboration, allowing people with different background, expertise, experience and resources to evaluate problems. This will increase the probability of finding more bugs and fixes; especially the hard ones [20].

One of the most popular licenses used for open-source software is the General Public License (GPL), with an approximate usage rate of 70%. The GPL is founded on four key principles [21]:

1. The right of individuals to **use software** for any purpose;
2. The right of individuals to **alter software** to meet individual needs
3. The right of individuals to **share software** with others
4. The right of individuals to **freely distribute** the changes one makes to software

In the context of software development, open-source often involves the use of open-source platforms and tools like GitHub. The open-source definition, published by the Open-Source initiative, provides criteria to determine whether a software license can be labeled as open-source. These criteria include requirement of free redistribution, availability of source code, allowance for derived works, integrity of the author's source code, no discrimination against person or groups, no discrimination against fields of endeavor, distribution of license and no restriction on other software, with the license being technology-neutral [22].

2.4.2 Python programming

Python is a versatile computer programming language often used to build software, websites, automate tasks and work with data analysis. It is a high-level programming language supporting structured-, object oriented- and functional programming. The programming language is one of the most popular programming languages out there because of its easy to learn syntax, versatility and beginner-friendliness. The syntax makes it more readable compared to its competitors. Python also supports packages and modules to encourage code reuse. The programming language has a large and active community, that contributes to the pool of modules and libraries within python and acts like a helpful resource for programmers.

2.4.3 Git

Git is a free and open-source distributed version control system, designed to handle both small and large projects with speed and efficiency. Version control meaning a system that records changes to a set of files over time to recall specific versions later and distributed meaning the clients doesn't just checkout the latest changes of the files, but rather fully mirror the repository [23]. Git has three stages for files to reach: modified, staged and committed. *Committed* meaning that the data is stored in the local database with current changes. *Staged* is the change in current file or files are marked and ready to go into the next commit. *Modified* means that the changes in current file have been changed, but not committed to the database yet. There is a lot of ways utilizing git. Git uses a command line tool as well as many other graphical user interfaces with different capabilities. The command line is the only tool that can run all Git commands, most of the other graphical interfaces only implement some of the main functions for simplicity reasons.[23]

Nearly every version control has some kind of branching support. [23] Branching is a way of diverge from the main code, working in a own path without messing with the main code [23]. To understand the way git handles branches, it is important to understand how git stores data. Git stores data in snapshots rather than a series of changesets or differences. When a commit is made, Git stores the git commit object that points to the snapshot of the content that is staged. This object also stores the author's name, email address, the commit message as well as the pointer that points to the previous commits (its parent). There are zero parents for initial commit, one parent for normal

commit and multiple parents from a merge with two or more branches. Let's say there are a directory containing four files, every file is staged and commited. The staged files compute a checksum for each one, storing the content of the files in the git repository as a tree object storing groups of objects together [23]. Each new version of a file is called a *blob*, so there are now four blobs. When running git commit, Git then creates a commit object containing a pointer to the root of the tree and all commit metadata. The Git repository then represents four files containing six objects; three blobs, one tree of the list of content as blobs and one commit as shown in figure 8. A branch is simply a pointer to one of these commits. [23]

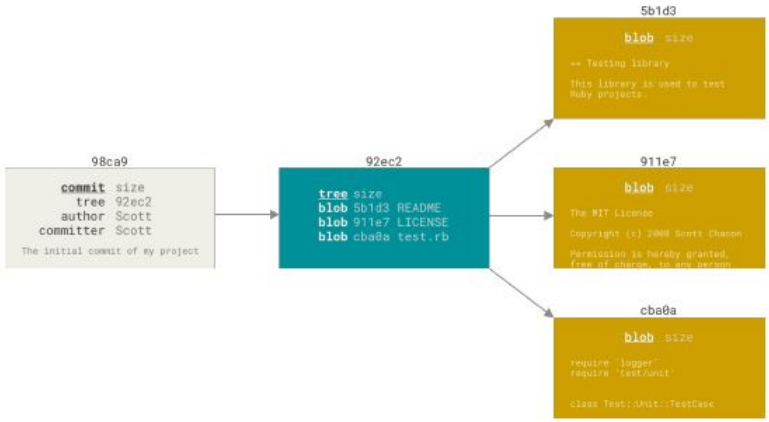


Figure 8: Commit tree Figure 9 from: [23]

When working on a branch there is always a need to merge the branch that is being worked on into the master branch. This is done by something called *merging*. Merging is done by checking out the branch that we want to merge with and collect the changes done in the branch into the master branch. This is done by the command git checkout and then use git merge command as illustrated in figure 9

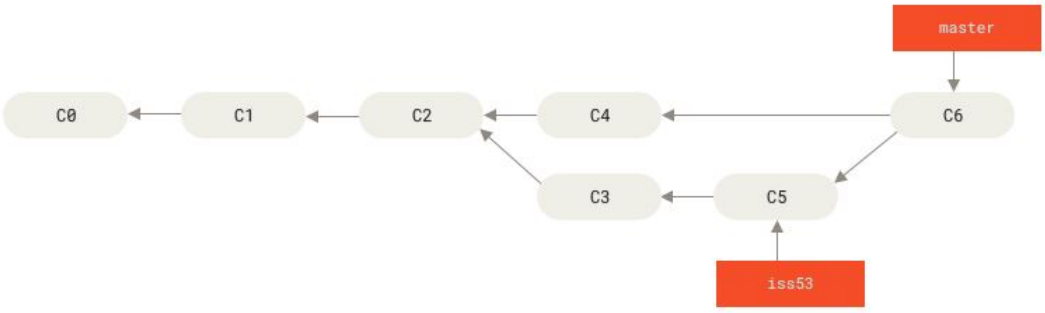


Figure 9: Merge tree Figure 25 from: [23]

3 Method

Arntzen and Tolsby [24] state that the research results are usually not considered absolutely true in the sense of being exact, universal or eternal knowledge. Rather, they are seen as a step towards greater understanding. Discussed in this chapter are methods for data collection, this includes literature study for the benchmarking part, collaboration in git and implementing of Model code 2010 with python.

3.1 Study design / Literature study

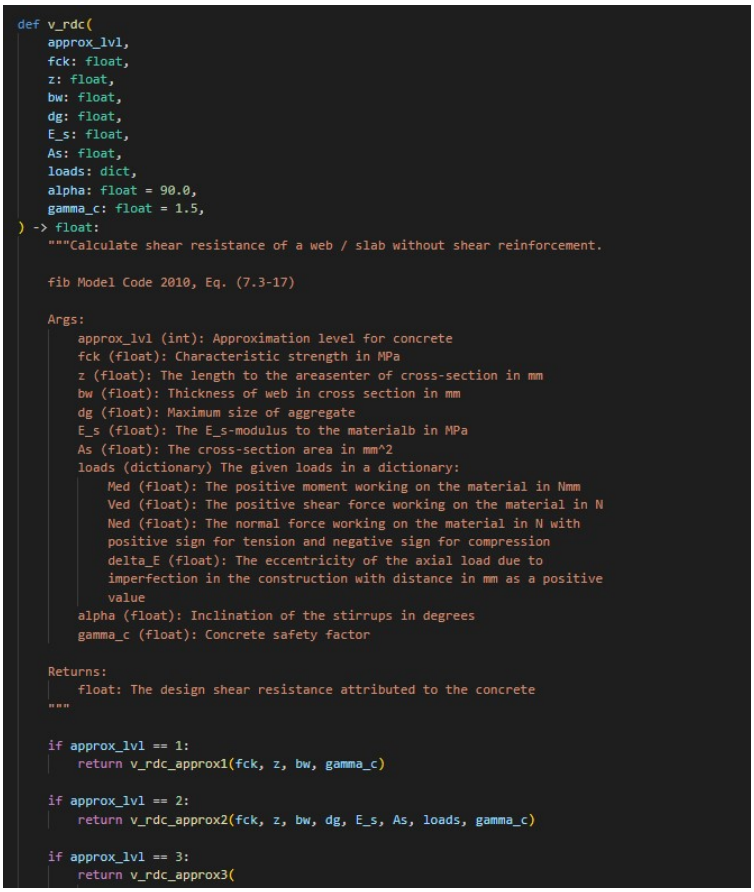
A literature study was done when benchmarking shear tests by searching for earlier thesis about "model code 2010" as well as "shear" and "concrete" in NTNU open. In the search for more information about the keywords, we also searched in multiple databases such as Oria, Scopus, Science direct and Compendex. This is a so called structured search [25]. During the literature search, we used techniques such as "forward snowballing" and "backward snowballing". These techniques were both used to dive into the citations of a relevant article and evaluate articles that have this article as a source [25]. This technique also allowed us to find different keywords relevant to our study, such as "high strength concrete" and "shear test anchorage" as well as finding the database for our benchmarking tests [5]. To narrow down the number of articles we did a study selection by only searching for articles after the year 2000. When evaluating relevant articles, we were also looking for a peer review marking. This indicates that the article has been evaluated by other scientists and approved. By choosing peer reviewed article we improved the credibility of the content in this thesis.

3.2 Collaboration in git

The collaboration took place on github, which is a code hosting platform for version control and collaboration that uses Git. The fib project is called "structuralcodes" and is an open-source project. This allows for contributions from all over the world with different backgrounds and specialties. The way the project works, is by making a copy of the project or a fork of the repository. This copy is the workspace for the project, and every contributor had his own copy that need to be in sync with the main repository. To ensure consistency of the code, formatters such as black, pylint and flake8 are used [26], these are used only for the aesthetics of the code. When the contributor is finished with a part of the code, it will then be sent to a pull request for the administrator and others to evaluate and comment on. While letting more participants in the project to discuss our code, we increased the chances of finding bugs in the code, which works like a peer reviewed article. The work we were doing was on the same repository and we utilized the branch mechanics to merge into our main, that again is merged to the original repository master branch.

3.3 Implementing Model code 2010 with python

We started to implement the formulas from chapter 7.3 in Model code 2010. Here we limeted the scope to cross sections without prestressing. This includes the formulas in section 2.5, equations 3 to 9 where we covered shear without shear reinforcement. Later, we did testing and benchmarking with these equations. There was also a need to make the loads (moment-, axial- and shear load) into a dictionary, which is a collection of keys / variables and values. The material factor for concrete was set to 1.5, and steel 1.15. It was also made possible to only insert the bare minimum of inputs if the user only wanted a quick calculation. The docstrings for each of the functions are written with the google format and we followed the guidelines in the repository [27]. The structure of the doctring in one of the functions can be seen in figure 10.



```python
def v_rdc(
    approx_lvl,
    fck: float,
    z: float,
    bw: float,
    dg: float,
    E_s: float,
    As: float,
    loads: dict,
    alpha: float = 90.0,
    gamma_c: float = 1.5,
) -> float:
    """Calculate shear resistance of a web / slab without shear reinforcement.

    fib Model Code 2010, Eq. (7.3-17)

    Args:
        approx_lvl (int): Approximation level for concrete
        fck (float): Characteristic strength in MPa
        z (float): The length to the areasenter of cross-section in mm
        bw (float): Thickness of web in cross section in mm
        dg (float): Maximum size of aggregate
        E_s (float): The E_s-modulus to the material in MPa
        As (float): The cross-section area in mm^2
        loads (dictionary) The given loads in a dictionary:
            Med (float): The positive moment working on the material in Nmm
            Ved (float): The positive shear force working on the material in N
            Ned (float): The normal force working on the material in N with
            positive sign for tension and negative sign for compression
            delta_E (float): The eccentricity of the axial load due to
            imperfection in the construction with distance in mm as a positive
            value
        alpha (float): Inclination of the stirrups in degrees
        gamma_c (float): Concrete safety factor

    Returns:
        float: The design shear resistance attributed to the concrete
    """

    if approx_lvl == 1:
        return v_rdc_approx1(fck, z, bw, gamma_c)

    if approx_lvl == 2:
        return v_rdc_approx2(fck, z, bw, dg, E_s, As, loads, gamma_c)

    if approx_lvl == 3:
        return v_rdc_approx3(
```

Figure 10: The structure of one the functions. Here; a function that redirects to another function with the corresponding approximation method

The process proceeded with implementing the equations for shear with shear reinforcements. Here the user could choose if they wanted stirrups or not and then the code would automatically calculate the right equations. This file would also contain Model code 2010 section 7.3.3.4, hollow core slabs. Both shear with and without shear reinforcement and hollow core slabs were made in the same python file.

We then started to make a python file for each of the categories: torsion, punching and different casting time. The structure for these files were the same as plain shear, as seen in figure 10. These files would not be as complex, however the file "_concrete_torsion.py" would need to import functions ($\epsilon_x, V_{rd,max}$) from the original shear file.

After finishing a function, a test was necessary. This was done by using pytest [28]. Here we defined the function and then selected a set of input values for the test. It is important what we chose as input values so that we could test all parts of the function. We also worked from the bottom to ensure that if a result from another function was used,

then that function would already be tested. After covering all the different parts of the functions, the manually calculations began. We calculated what we wanted the function to return. If these results were different from the test, then either we had made a mistake in our calculations or the function did not do as we wanted it to. This process made us correct errors before we had to commit the functions. As we tested V_{rdc}, six different inputs scenarios were needed to ensure correct return form the code. Here the rel_tol was set to 0.001. This meant that the function should calculate a result that was 0.1% or less from our manually calculated result.

```
@pytest.mark.parametrize(
    '''approx_lvl, fck, z, bw, dg, E_s, As, loads, alpha, gamma_c, expected''',
    [
        (1, 35, 180, 300, 0, 0, 0, create_load_dict(0, 0, 0, 0), 0, 1.5, 31294),
        (1, 35, 200, 300, 0, 0, 0, create_load_dict(0, 0, 0, 0), 0, 1.5, 34077),
        (2, 35, 140, 300, 16, 21e4, 2000, create_load_dict(40e6, 2e4, 1000, 50), 0, 1.5,
         48828),
        (2, 35, 140, 300, 32, 21e4, 2000, create_load_dict(40e6, 2e4, 1000, 50), 0, 1.5,
         50375),
        (3, 35, 200, 300, 32, 21e4, 2000, create_load_dict(40e6, 2e4, 1000, 50), 1.5, 1.5,
         67566),
        (3, 35, 200, 300, 32, 21e4, 2000, create_load_dict(40e6, 20e6, 1000, 50), 1.5, 1.5,
         0),
    ],
)
def test_v_rdc(
    approx_lvl, fck, z, bw, dg, E_s, As, loads, alpha, gamma_c, expected
):

    """Test the v_rdc function."""
    assert math.isclose(_concrete_shear.v_rdc(
                approx_lvl, fck, z, bw, dg, E_s, As, loads, alpha, gamma_c
                ),
        expected, rel_tol=0.001)
```

Figure 11: Shows how we tested if a function made the calculations we expected

3.4 Benchmark

When starting to benchmark, our goal needed to be well thought through, so that we knew in which direction we wanted to go. Here we decided that we wanted to compare the results from our code with Eurocode 2. There is also a new Eurocode 2 that is almost done that we will reference as the *new Eurocode 2*. We want to see which parameters about Model code 2010 is stronger or weaker than Eurocode 2, and visualise this. This would be done for the part of the code that did not include shear reinforcement. Approximation 2 has the aggregate size matters and therefore we decided to make two calculations with equations 7 and 8. The normal range in Europe is between 16 and 32mm, and equation 7 makes it so that a dg between 27 and 32mm will give the same result [29]. We therefore

calculated one with an aggregate size of 20 and one with an aggregate size of over 27. Hence a total of 5 different scenarios would be needed:

1. Model code 2010 with approximation 1
2. Model code 2010 with approximation 2 and a maximum aggregate seize of 20mm
3. Model code 2010 with approximation 2 and a maximum aggregate seize of 27mm or higher
4. Eurocode 2, 2004 edition.
5. New Eurocode 2, revised version.

We made an excel sheet, where we plotted all the data we got from ACI with rectangular cross sections [5]. Here we noted the: $width[mm]$, $height[mm]$, $d[mm]$ (distance from reinforcement to the edge), $z[mm]$ (the effective shear depth), $f_{1c}[MPa]$, $f_{ck}[MPa]$, $f_{1ct}[MPa]$, $A_{sl}[mm^2]$, $V[kN]$. z was assumed to be 0.9 of d. We went through the test inputs twice to ensure that we did not make any errors, see figure 12.

w [mm]	h [mm]	d [mm]	z [mm]	f1c [Mpa]	fck [Mpa]	f1ct [Mpa]	Asl [mm^2]	fsy [Mpa]	V [kN]
360	310	278	250	49.9	52.5	4	1570	536	**128**
360	310	278	250	49.9	52.5	4	1570	536	**119**
290	310	278	250	46.8	49.3	3.8	1570	536	**108**
290	310	278	250	43.9	46.2	3.7	1570	536	**81**
290	210	178	160	48.9	51.5	4	1570	536	**75**
290	310	278	250	56	58.9	4.2	800	536	**90**
127	254	203	183	59.3	62.4	4.3	1013	414	**58**
127	254	203	183	59.3	62.4	4.3	1013	414	**69**
127	254	203	183	59.3	62.4	4.3	1013	414	**69**
127	254	208	187	59.3	62.4	4.3	467	414	**49**
127	254	202	182	65.3	68.7	4.5	1289	414	**51**
127	254	202	182	65.3	68.7	4.5	1289	414	**69**
127	254	202	182	65.3	68.7	4.5	1289	414	**100**
127	254	208	187	65.3	68.7	4.5	594	414	**45**
127	254	208	187	65.3	68.7	4.5	594	414	**47**
127	254	208	187	65.3	68.7	4.5	594	414	**80**
127	254	184	166	62.7	66.0	4.4	1552	414	**54**
127	254	184	166	62.7	66.0	4.4	1552	414	**76**
127	254	184	166	62.7	66.0	4.4	1552	414	**69**

Figure 12: Inputs of tests

When the preconditions were set, the formulas for expected shear strength could be calculated. We first used excel for the calculations both Model code 2010 and Eurocode 2 to get an impression of the results. These results where the actual strength divided on the expected strength. Here we color coded the results to easier see what the results were, this can be seen in figure 13.

Ratio				
MC2010_approx1	MC2010_approx2	MC2010_approx3	EC2	EC2_new
2.136646567	1.548114863	1.504105544	1.32006	1.3861349
1.989176059	1.386042419	1.346640444	1.22895	1.2904644
2.313502947	1.531748491	1.48820443	1.31579	1.3816527
1.787639992	1.0307502	1.00144836	1.00592	1.0562791
2.253602258	1.520686098	1.491068393	1.29342	1.1015657
1.765399334	1.551625162	1.507516053	1.29525	1.3600829
3.227525997	2.258768487	2.209459944	1.87605	1.5270406
3.847345003	2.619165812	2.561989855	2.23632	1.8202959
3.847345003	2.51463343	2.459739397	2.23632	1.8202959
2.67711892	2.330017427	2.278079453	1.62383	1.6588484
2.834876664	1.663670531	1.627506894	1.61526	1.2108034
3.814902385	2.309087882	2.258894641	2.17366	1.6293819
5.542405352	3.879562889	3.795231826	3.15796	2.3672152
2.405993071	1.961928921	1.918195938	1.37334	1.3492171
2.524941043	1.855842984	1.814474742	1.44124	1.41592
4.330787528	4.213282952	4.119365465	2.47202	2.4285908
3.246174137	1.812293473	1.775959137	1.90156	1.2621674
4.519535262	2.635242087	2.582408607	2.64747	1.7572717
4.118994438	2.235579576	2.190758856	2.41284	1.6015347
2.464263192	1.65944453	1.62260674	1.42538	1.2377435
2.415412159	1.442730346	1.410703366	1.39713	1.2132068
2.464263192	1.433144439	1.401330256	1.42538	1.2377435
2.656294945	1.562341722	1.458522319	1.32487	1.6462947
2.226758913	1.331573392	1.24308881	1.19091	1.4798297
1.988186712	1.172637826	1.094714694	1.09421	1.3596735
1.574556327	0.941564575	0.878996527	0.9401	1.168172
1.463911828	1.293059131	1.293059131	0.81581	1.0137288
3.144201797	1.611198566	1.506156118	1.35403	1.6496914
1.986028219	1.659007866	1.548764886	1.33824	1.6629044

Figure 13: Colour coded excel results

Then the process started in python. With the excel file inputs uploaded, the code that we had written for shear could be used. We ensured that the results were the same as in the excel document and proceeded to remove the safety margin in the formulas. This was to observe two things. Firstly to see if the results are safe enough or too safe with the safety margin on, and secoundly to see how close the formulas were to the actual tests without additional safety factors. Removing the safety margin in the formulas included reducing the concrete material factor γ_c from 1.5 to 1 and using the anticipated concrete strength, that is 8 MPa higher than the characteristic strength.

The results were then sorted out for different parameters: width, height, z, f_{ck}, A_{sl}, M_{ed}, and distance between support a. This was to see in what areas that the results were especially bad or good. Here we already had an idea from the color grading we did in excel. We also plotted a figure with the expected data on the y-axes and the test results on the x-axes to see the variance better.

3.5 Validation and reliability

After we got our results, the interpretation part began. After getting a visually overview from for evaluation of the data would help analysing and concluding. Here the expected results from the given method would be needed. This was done simply by calculating the average from the test results. While the spread of the data, how precis we can expect a given method to be, would be measured with standard deviation. The formula for standard deviation can be found in equation 15:

$$\sigma_{standard-deviation} = \sqrt{\frac{1}{N}\sum(x_i - \mu)^2} \tag{15}$$

Here N is the total tests, x_i is test result and μ is the average. This was calculated for both Model code 2010 and Euro code 2.

4 Results

In this chapter we cover the implementation of the python package with a link to the project and a link to our contribution as a merged commit. An illustration of the code has also been done by making a flowchart, this is showen in figure 14 to 17 in chapter 4.1 The results for the benchmarking will be shown in chapter 4.2 through 4.3 with and without safety margins as plot sorted by different variables. We decided to examine the data that caught our attention during the sorting process. These were values in the y-axis close to one, significant spread or any other trends. By looking at these trends it led to further investigation on high strength concrete and adjustments on variables in Model code 2010 shown in chapter 4.4 and 4.5. At the end of the chapter we decided to use standard deviation and coefficient of variation to make out a value for spread and consistency of the different standards. This is explained in chapter 4.6.

4.1 Code contribution to the fib project

The python project "fib - International Federation for Structural Concrete" can be found here: https://github.com/fib-international/structuralcodes/. It is work in constant progression, so the structure of the project may change after publishing of the thesis. Our functions can be found under structuralcodes/codes/mc2010 in project and the tests can be found under structuralcodes/tests. A direct link to our contribution is the link to the merged commit here: https://github.com/fib-international/structuralcodes/commit.

In the contribution to the project, we made both functions and tests. The functions cover general shear, punishing, casting at different time and torsion. The code was made before the test, so there was no test-driven development from our contribution. From an aesthetic point of view, we use so-called doc-strings to describe what the functions intentions are. The doc-string describes what the function does, where to find it in the Model code 2010, what arguments with desecration it takes and what it returns.

The code we have done as a contribution to the project ended up being 3190 lines for functions and tests spreading among over 300 commits. The functions itself is a default setup of a function in python with a name of the function behind def, some inputs in the parentheses and an end product or a return. These functions were as described in the method chapter, gone through by the developer and community as an open-source to ensure validity of the code. The tests associated with the functions used assert, which compares functions with input to the expected value.

To further illustrate the code and contribute to the project, we have decided to create some flowcharts seen in figure 14 to 17. The flowchart represents each of our files: shear capacity, torsion, punching and different casting time. It contains the functions used and the arrows shown in the flowchart are pointing in the direction of a function containing another function. The first chart, seen in figure 14, is for plain shear split in a direction with and without reinforcement. The code then takes approximation levels into consideration and this is done by an input in the code. At the end of the code, we return the resistance of web or slab called V_{rd}. The second chart takes torsion into account, it contains mainly $t_{rd,max}$, $v_{ed,ti}$ as well as approximation levels, this can be seen in figure 15. The third flowchart illustrated in figure 16 focuses mainly on the $psi_{punching}$ with branching out to $v_{rd,max,punching}$, $v_{rd,punching}$ and $v_{rds,punching}$. The last flowchart from figure 17 takes different casting times into consideration and only contains functions not dependent on

each other, those are tau_{edi}, $tau_{rdi,without,\ reinforcement}$ and $tau_{rdi,with,\ reinforcement}$.

To further try to explain the flowchart, we decided to explain every function used in table 1. This can also be followed in the code with the merged commit https://github.com/fib-international/structuralcodes/commit.

Function	**Purpose of the function**	**What equation in Model code**
V_{rd}	The shear resistance of a web or slab	7.3-11
V_{rdc}	The Shear resistance contribution of a web/slab from the concrete	7.3-17
$V_{rdc,approx1}$	The Shear resistance that is contributed from concrete with approximation method 1	(7.3-17) and (7.3-19)
$V_{rdc,approx2}$	The Shear resistance that is contributed from concrete with approximation method 2	(7.3-17), (7.3-20) and (7.3-21)
$V_{rdc,approx2}$	The Shear resistance that is contributed from concrete with approximation method 3	(7.3-17), (7.3-39) and (7.3-43)
V_{rds}	The shear resistance that shear reinforcement gives	(7.3-25) and (7.3-29)
$v_{rd,max}$	The maximum allowed shear resistance given the cross section	(7.3-24) and (7.3-26)
$v_{rd,max,approx1}$	The calculated shear resistance, with level 1 approximation	(7.3-37)
$v_{rd,max,approx2}$	The calculated shear resistance, with level 2 approximation	(7.3-24), (7.3-40) and (7.3-41)
$v_{rd,max,approx3}$	The calculated shear resistance, with level 3 approximation	(7.3-24), (7.3-40) and (7.3-41)
$v_{rd,ct}$	The shear resistance for a hollow core slab	(7.3-44) and (7.3-45)
$v_{rd,ct,approx1}$	The maximum shear force for level 1 approximation	(7.3-44)
$v_{rd,ct,approx2}$	The maximum shear force for level 2 approximation	(7.3-45), (7.3-46) and (7.3-47)
$epsilon_X$	The longitudinal strain	(7.3-16)
eta_{fc}	determin the strength reduction factor	(7.3-28)

Function	**Purpose of the function**	**What equation in Model code**
$v_{ed,ti}$	Shear force due to torsion	(7.3-53)
$t_{rd,max}$	The maximum allowed torsion	(7.3-56)
t_{rd}	Checks if the combination of torsion and shear is ok	(7.3-56)
b_0	Gives the general output for b_0, shear-resisting control perimeter.	(7.3-57)
m_{ed}	The acting bending moment acting in the support strip	(7.3-76), (7.3-71), (7.3-72), (7.3-73) and (7.3-74)
$psi_{punching}$	The rotation of the slab around the supported area	(7.3-70), (7.3-75) and (7.3-77)
$v_{rdc,punching}$	Punching resistance where the contribution is from the concrete	(7.3-61), (7.3-62) and (7.3-63)
$v_{rds,punching}$	Punching resistance where the contribution is from the reinforcement	(7.3-64) and (7.3-65)
$v_{rd,max,punching}$	The maximum value you can have for $v_{\mathrm{rd\_punching}}$	(7.3-68) and (7.3-69)
$v_{rd,punching}$	The total resistance for punching, both $V_{rd,c}$ and $V_{rd,s}$	(7.3-60)
tau_{edi}	Shear at the interface between concrete cast at different times	(7.3-49)
$tau_{rdi,without,reinforcement}$	Shear resistance without reinforcement at the intersection with different casting time	(7.3-50)
$tau_{rdi,with,reinforcement}$	Shear resistance with reinforcement at the intersection with different casting time	(7.3-51)

Table 1: An overview of the function, what they are and their placement in Model code 2010.

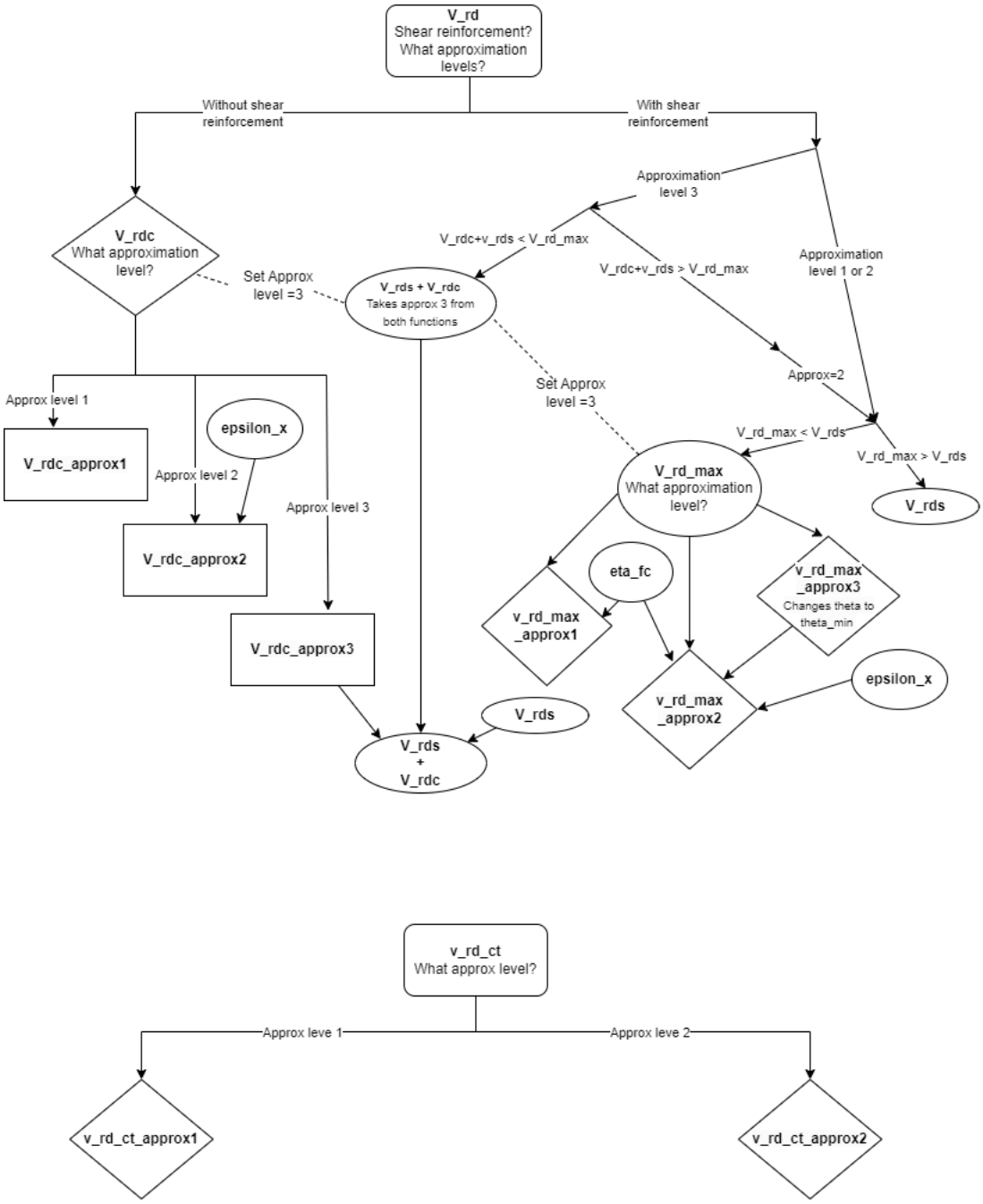


Figure 14: A flowchart showing the general shear functions and hollow core slab in the file named "_concrete_shear.py"

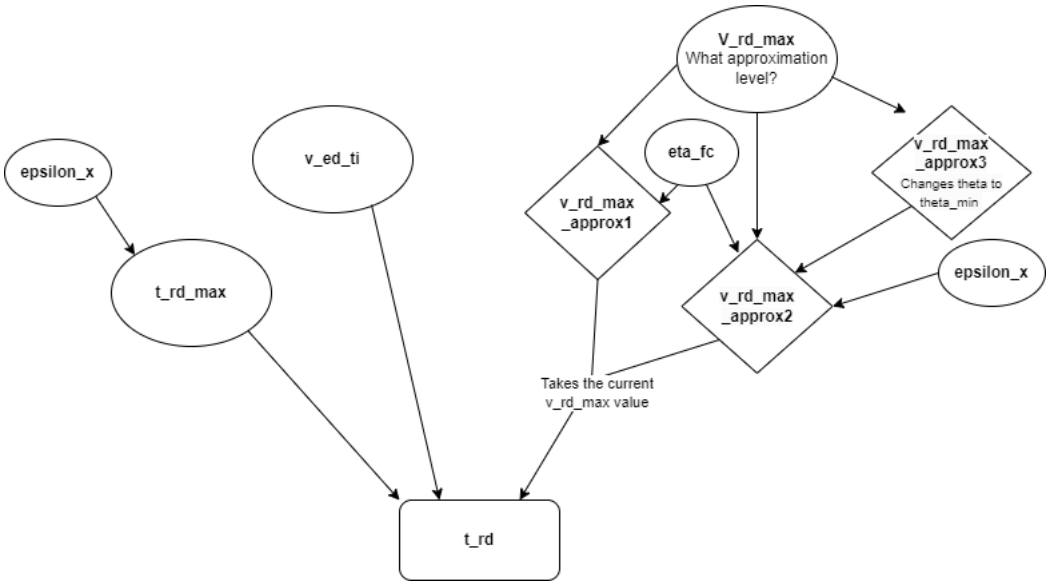


Figure 15: A flowchart showing the functions for torsion in the file named "_concrete_torsion.py"

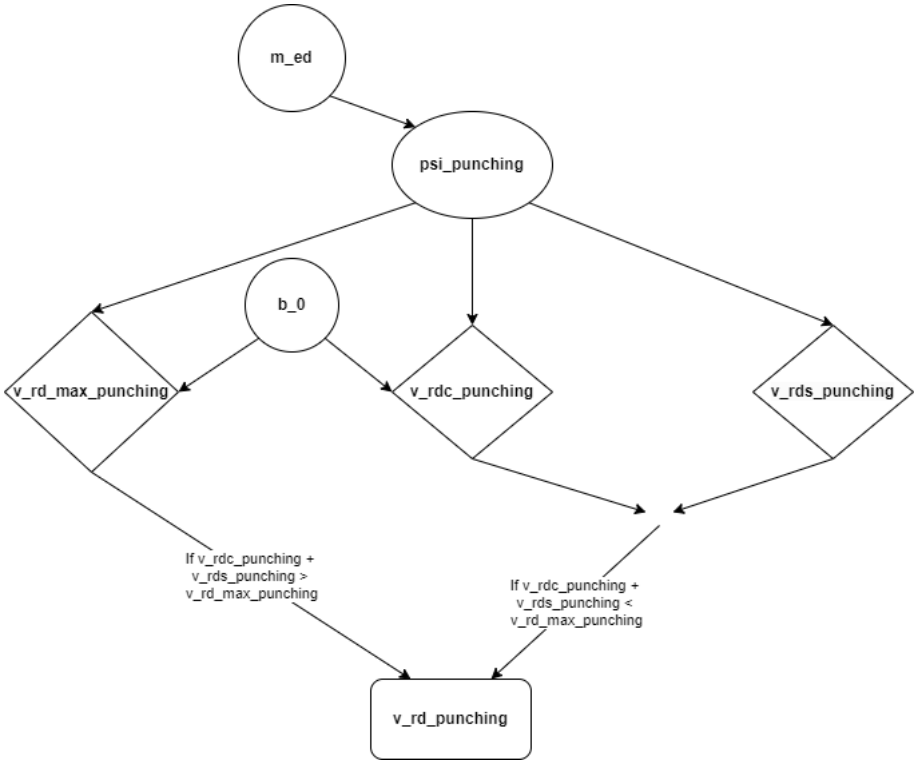


Figure 16: A flowchart showing the functions for punching in the file named "_concrete_punching.py"

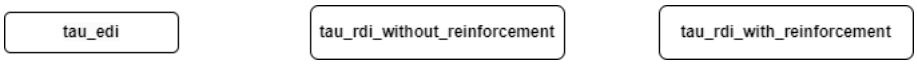


Figure 17: A flowchart showing the functions for different casting time in the file named "_concrete_interface_different_casting_times.py"

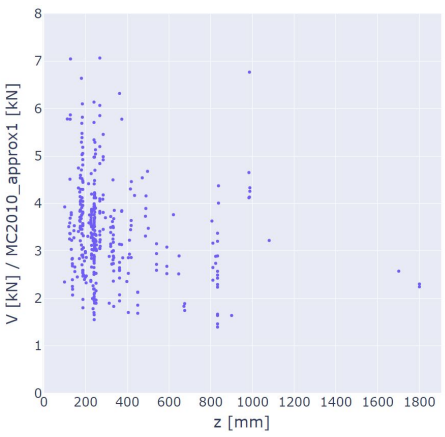
4.2 Benchmark with safety margin

The benchmarking was calculated for each of the standards, with specifications like *dg* and sorted by both internal moment arm, concrete strength, acting moment and distance from support. This was done both with and without safety margin. Results with safety margin can be seen in figure 18 to 21. Here all the tests are the actual results divided by the calculated expected capacity from the standards. This ratio is the unit on the y-axis, while the x-axis has different units depending on what it is sorted by. All figures have five sub figures, Model code 2010 approx 1, Model code 2010 approx 2 with dg = 20 Model code 2010 approx 2 with dg = 28, Eurocode 2 and the new Eurocode 2, in that order.

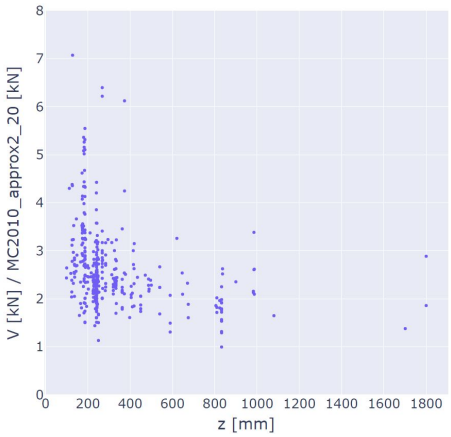
We also made a plot where we have the expected results on the y-axis and the actual results on the x-axis. This can be seen in figure 22. Here we also included a line, in grey, telling when expected results match the tests. In figure 22 a trendline is also added in the same color as the dots and shows a first-degree regression for the trend. Both axises have kN as unit.

The figures from 18 to 22 have a safety margin that include 8 MPa lower strength on the concrete than what we can expect and a material factor $\gamma_c = 1.5$, see section 2.1.1. This means that the results in this section are what we can expect to get when compared to what the industry does.

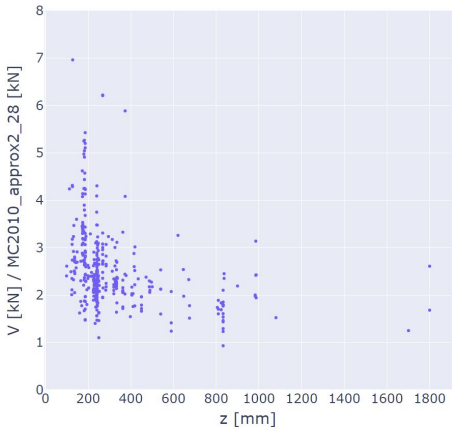
There were also made similar dot diagrams for width, height and reinforcement amount. These were not added due to high similarities to other diagrams, such as height and z.



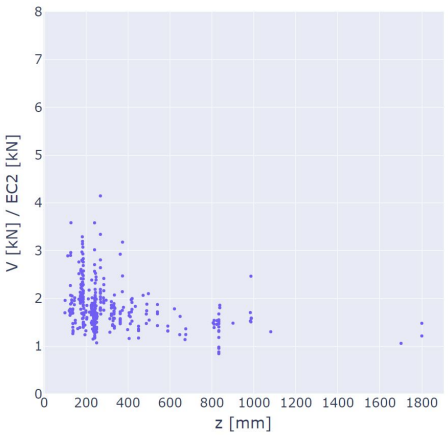
(a) Model code 2010 approx method 1 sorted by z



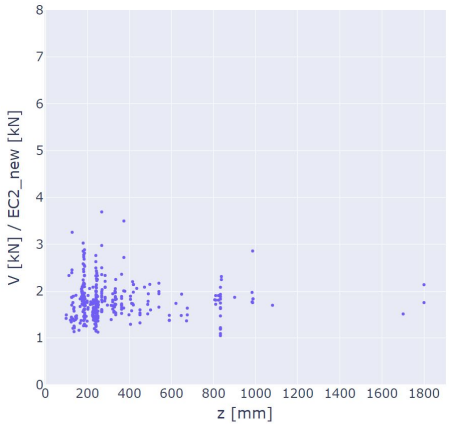
(b) Model code 2010 approx method 2 dg=20 sorted by z



(c) Model code 2010 approx method 2 dg=28 sorted by z

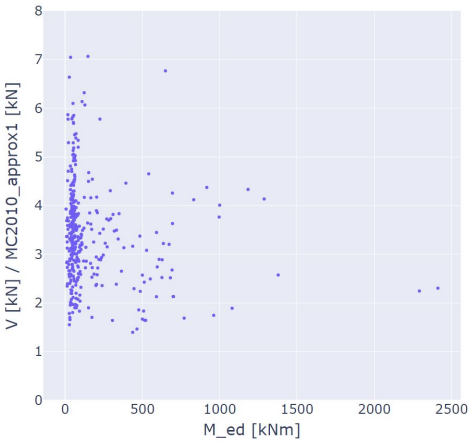


(d) EC2 sorted by z

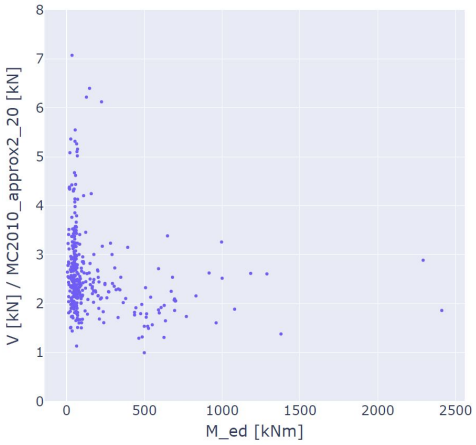


(e) New EC2 sorted by z

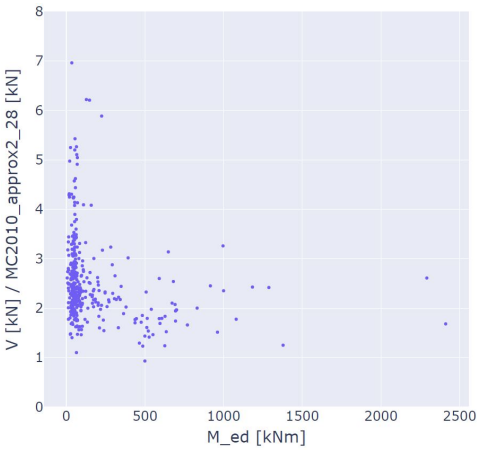
Figure 18: Sorted by internal moment arm, z



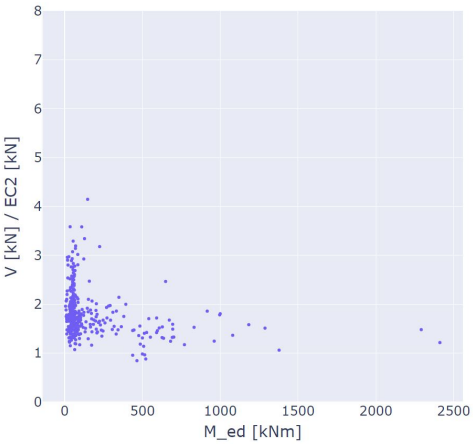
(a) Model code 2010 approx method 1 sorted by M_{ed}



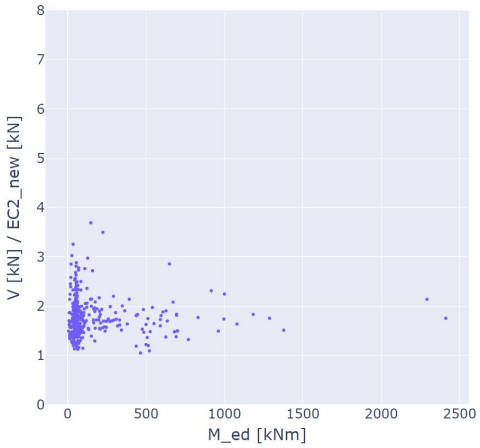
(b) Model code 2010 approx method 2 dg=20 sorted by M_{ed}



(c) Model code 2010 approx method 2 dg=28 sorted by M_{ed}

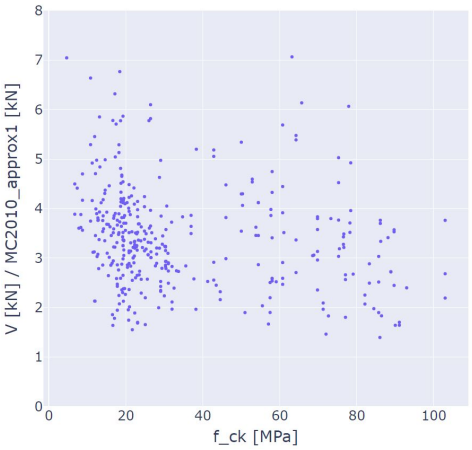


(d) EC2 sorted by M_{ed}

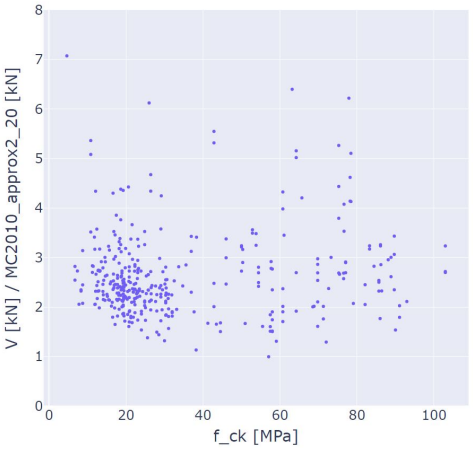


(e) New EC2 sorted by M_{ed}

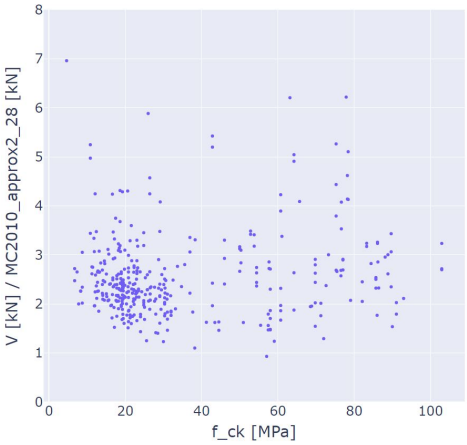
Figure 19: Sorted by external moment M_{ed}



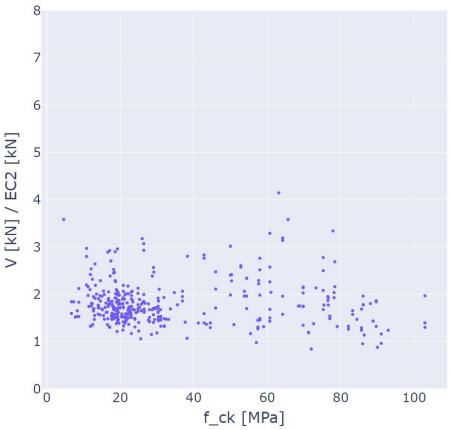
(a) Model code 2010 approx method 1 sorted by f_{ck}



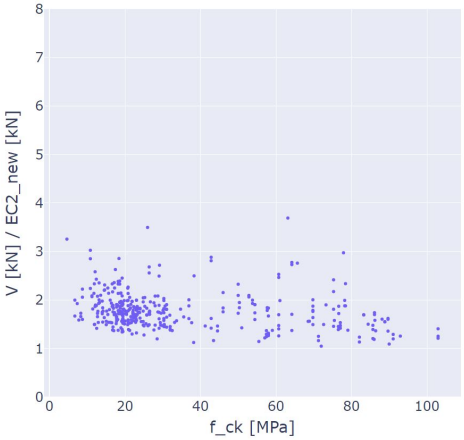
(b) Model code 2010 approx method 2 dg=20 sorted by f_{ck}



(c) Model code 2010 approx method 2 dg=28 sorted by f_{ck}

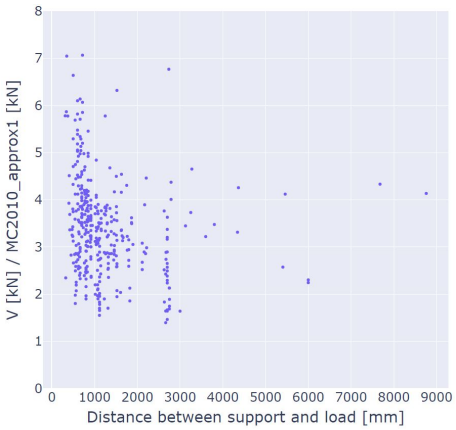


(d) EC2 sorted by f_{ck}

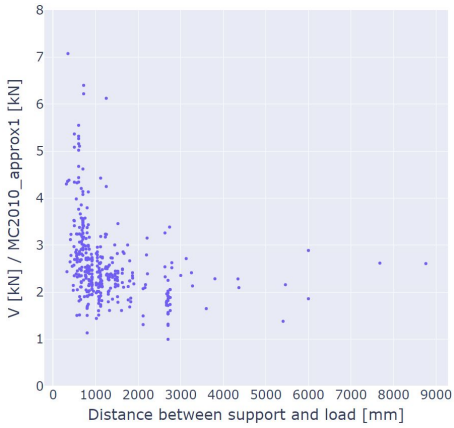


(e) New EC2 sorted by f_{ck}

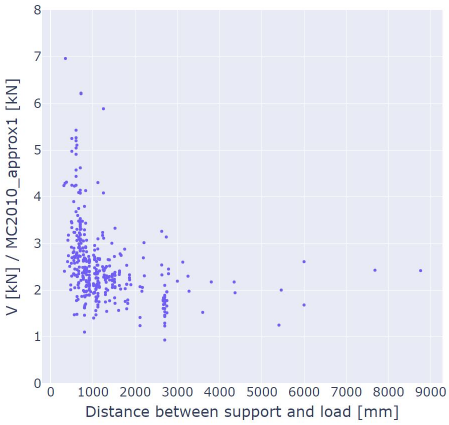
Figure 20: Sorted by characteristic strength, f_{ck}



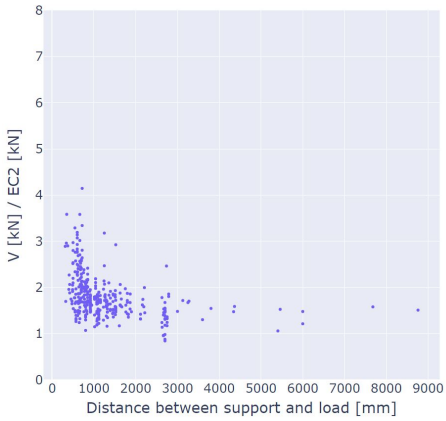
(a) Model code 2010 approx method 1 sorted by distance from support, a



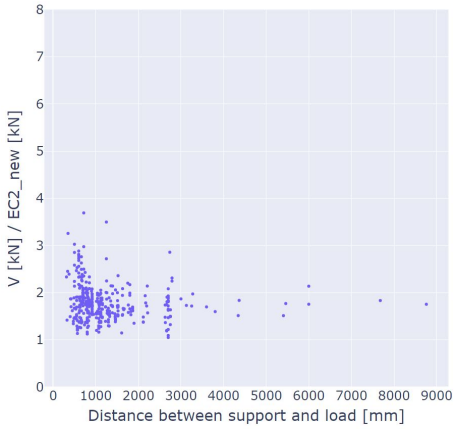
(b) Model code 2010 approx method 2 dg=20 sorted by distance from support, a



(c) Model code 2010 approx method 2 dg=28 sorted by distance from support, a

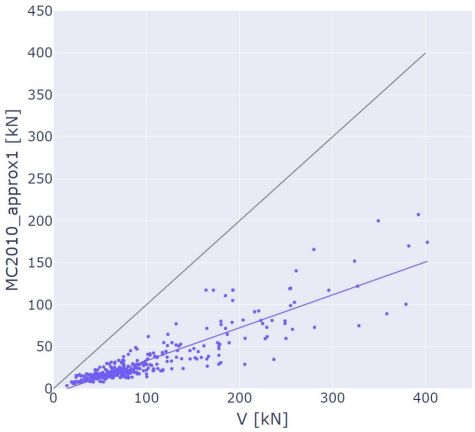


(d) EC2 sorted by distance from support, a

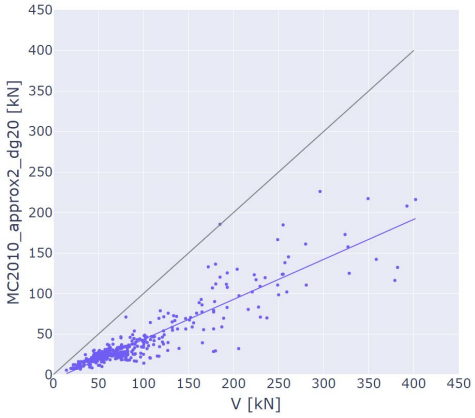


(e) New EC2 sorted by distance from support, a

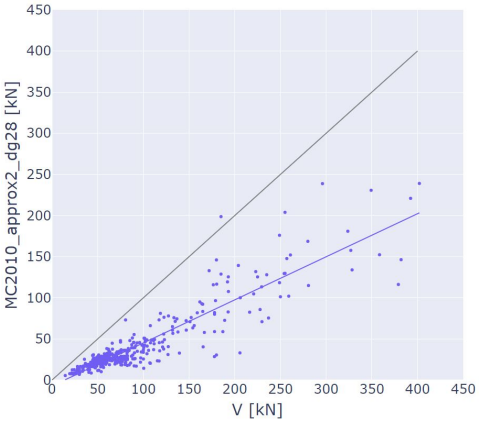
Figure 21: Sorted by distance from support, a



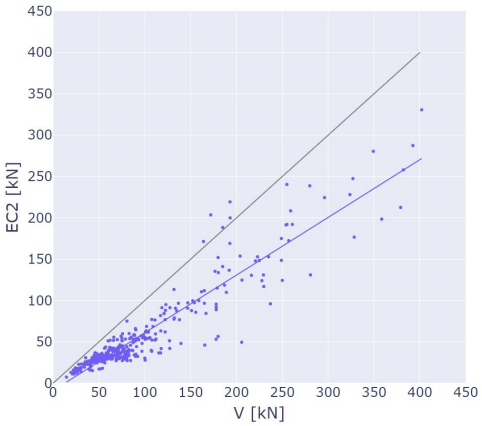
(a) Model code 2010 approx method 1 compared directly with test results



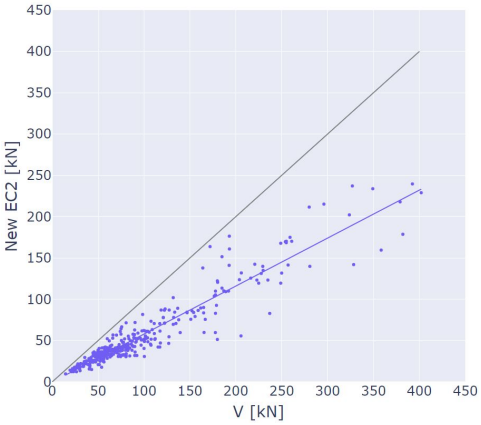
(b) Model code 2010 approx method 2 dg=20 compared directly with test results



(c) Model code 2010 approx method 2 dg=28 compared directly with test results



(d) EC2 compared directly with test results

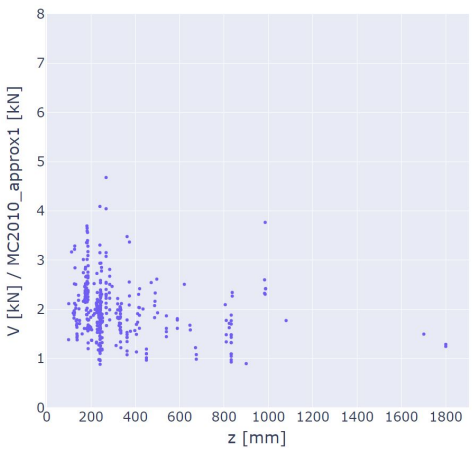


(e) New EC2 compared directly with test results

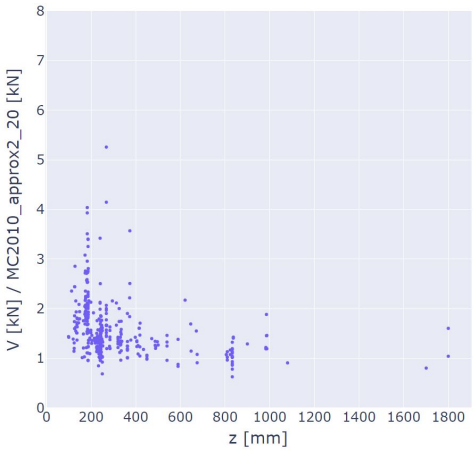
Figure 22: Direct comparison with the test results

4.3 Benchmarking without safety margin

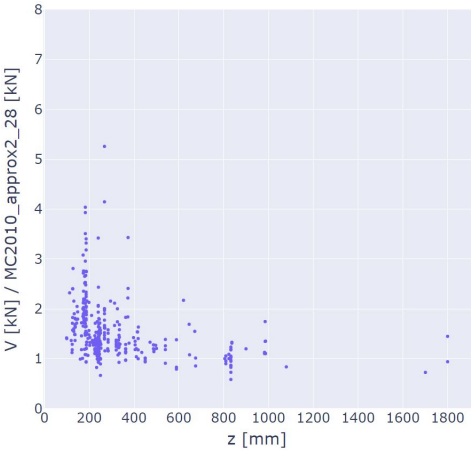
The results in this section have similarities to the previous section. The difference being that we excluded the safety factor, this being γ_c. At the same time calculating with the expected strength of concrete. This is 8 MPa higher than in the previous section. Plotted results without safety margin can be seen in figure 23 to 26. Then in figure 27, a direct comparison was made, where we without the safety margin should get results to align with the grey line.



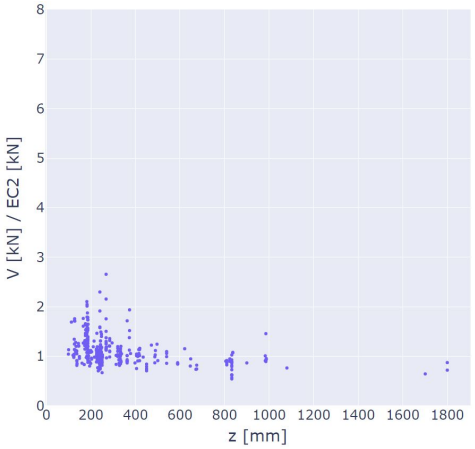
(a) Model code 2010 approx method 1 sorted by z



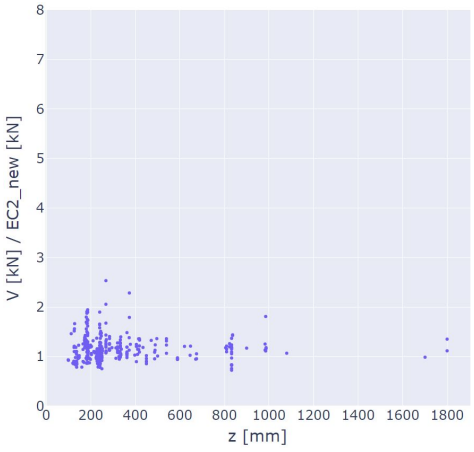
(b) Model code 2010 approx method 2 dg=20 sorted by z



(c) Model code 2010 approx method 2 dg=28 sorted by z

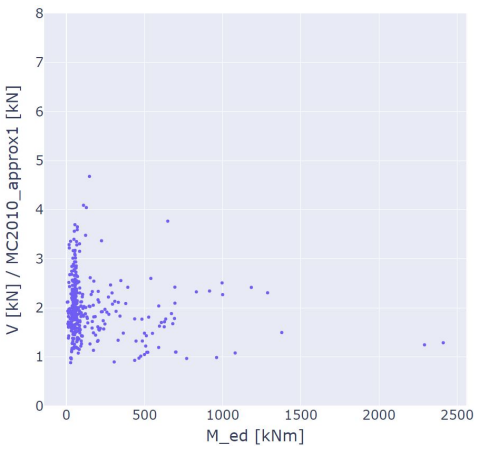


(d) EC2 sorted by z

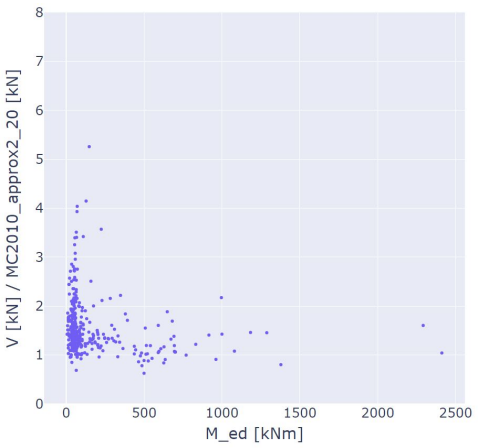


(e) New EC2 sorted by z

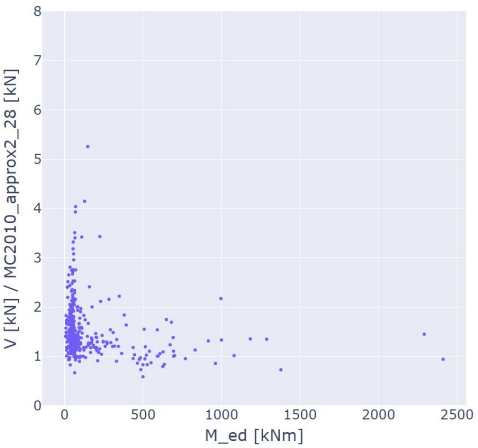
Figure 23: Sorted by internal moment arm, z



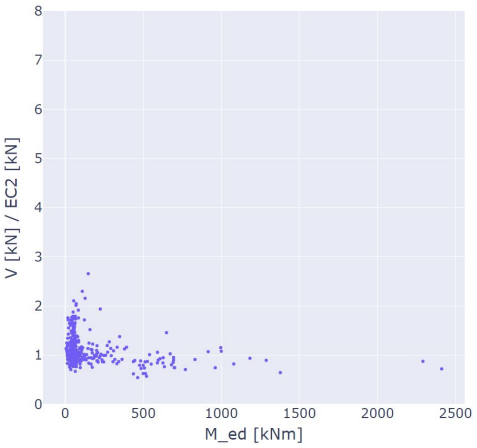
(a) Model code 2010 approx method 1 sorted by M_{ed}



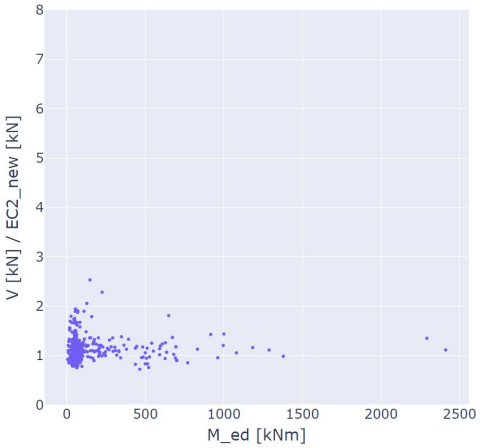
(b) Model code 2010 approx method 2 dg=20 sorted by M_{ed}



(c) Model code 2010 approx method 2 dg=28 sorted by M_{ed}

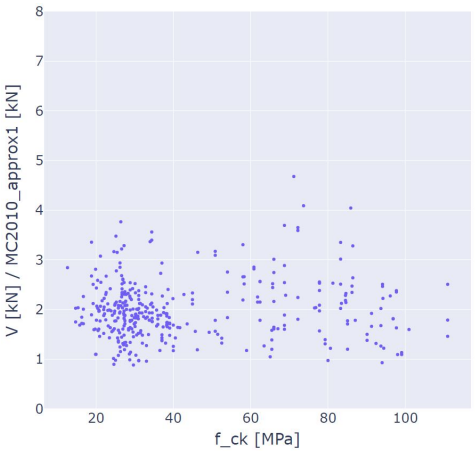


(d) EC2 sorted by M_{ed}

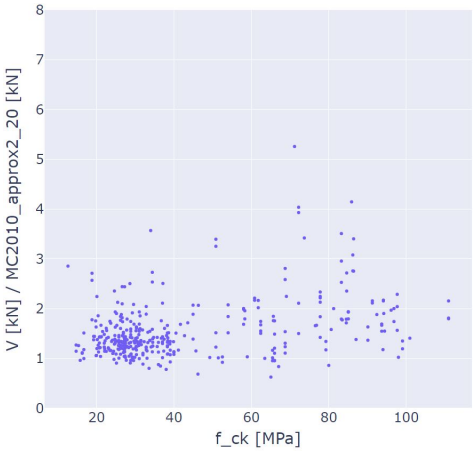


(e) New EC2 sorted by M_{ed}

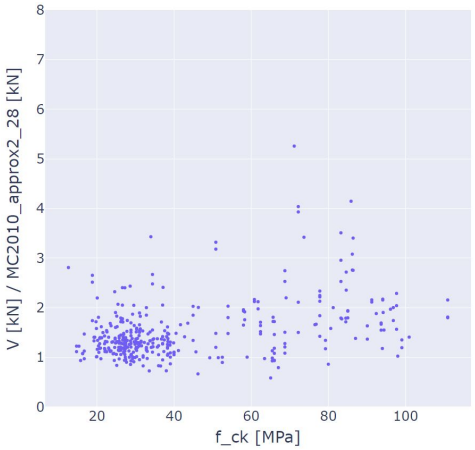
Figure 24: Sorted by external moment m_{ed}



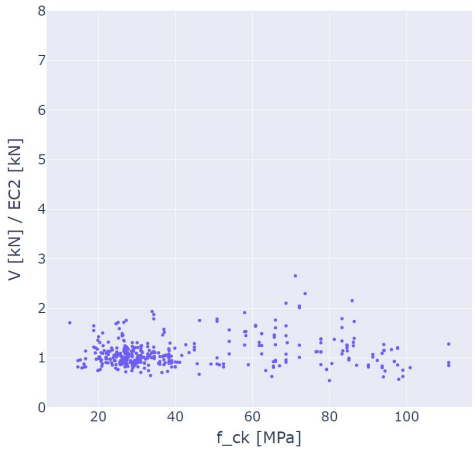
(a) Model code 2010 approx method 1 sorted by f_{ck}



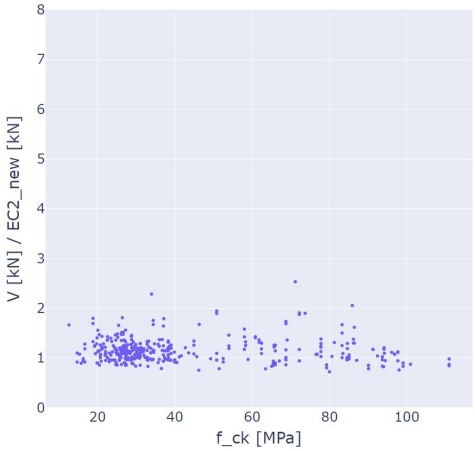
(b) Model code 2010 approx method 2 dg=20 sorted by f_{ck}



(c) Model code 2010 approx method 2 dg=28 sorted by f_{ck}

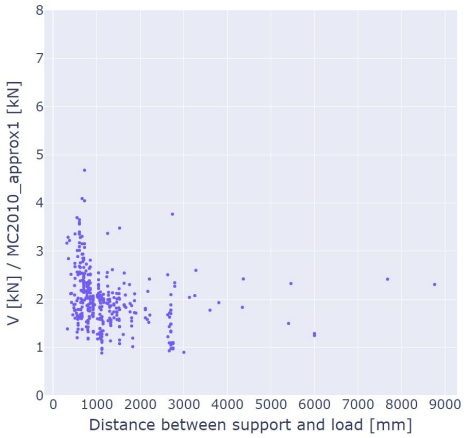


(d) EC2 sorted by f_{ck}

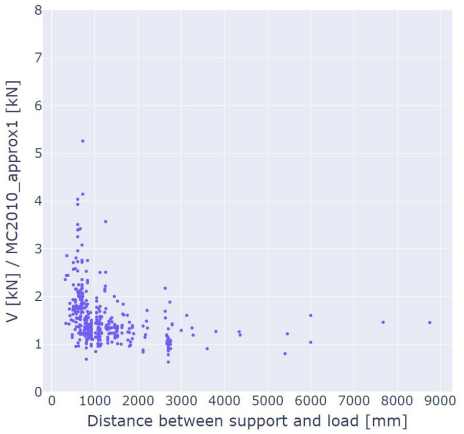


(e) New EC2 sorted by f_{ck}

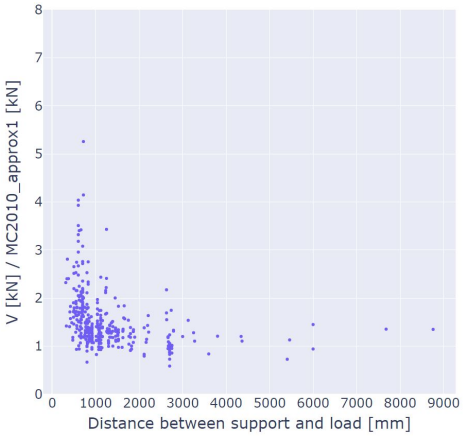
Figure 25: Sorted by characteristic strength, f_{ck}



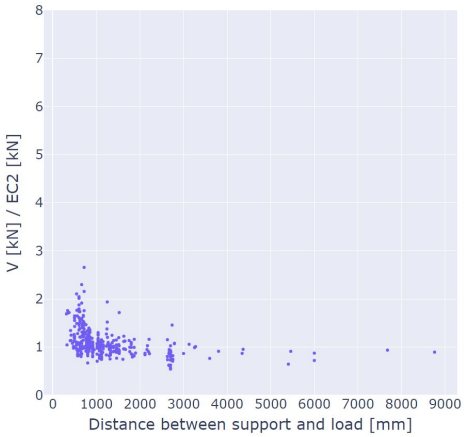
(a) Model code 2010 approx method 1 sorted by distance from support, a



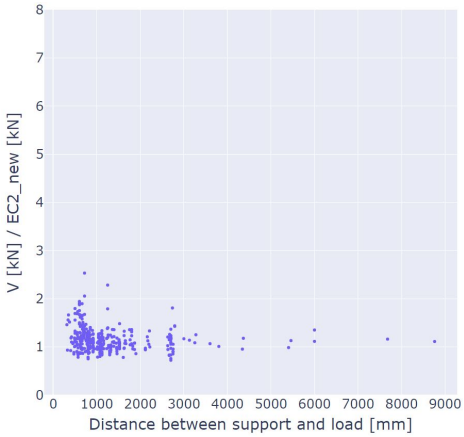
(b) Model code 2010 approx method 2 dg=20 sorted by distance from support, a



(c) Model code 2010 approx method 2 dg=28 sorted by distance from support, a

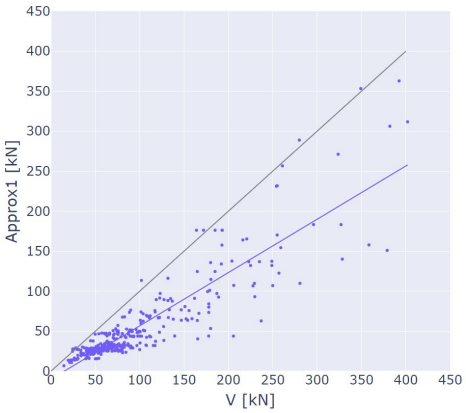


(d) EC2 sorted by distance from support, a

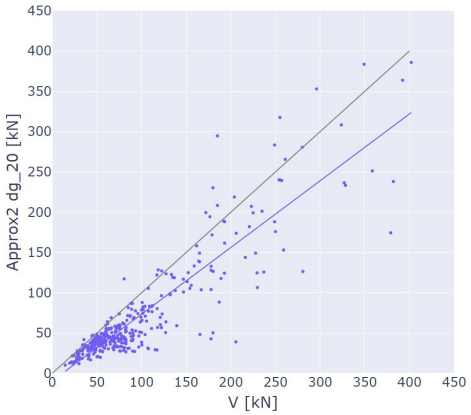


(e) New EC2 sorted by distance from support, a

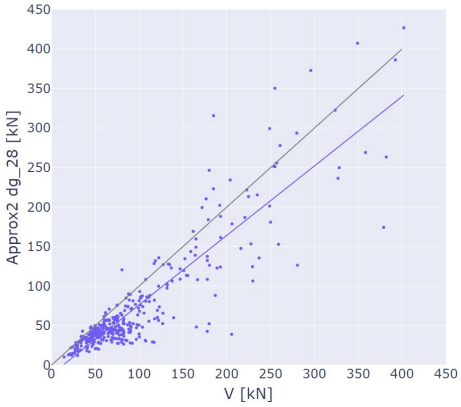
Figure 26: Sorted by distance from support, a



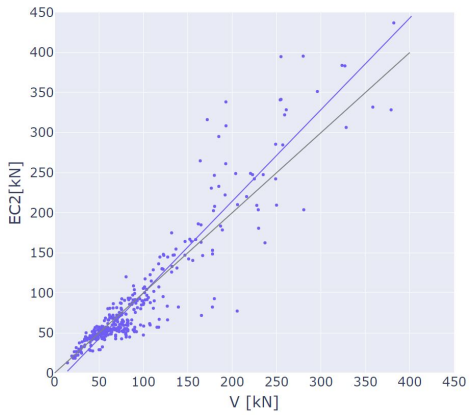
(a) Model code 2010 approx method 1 compared directly with test results



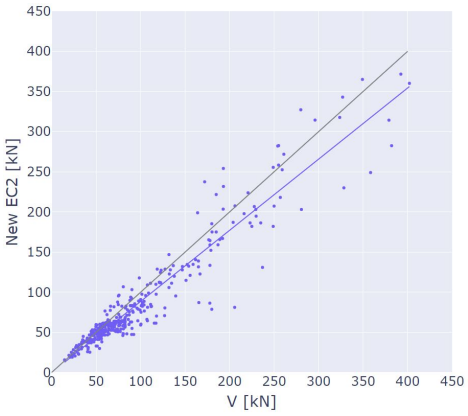
(b) Model code 2010 approx method 2 dg=20 compared directly with test results



(c) Model code 2010 approx method 2 dg=28 compared directly with test results



(d) EC2 compared directly with test results



(e) New EC2 compared directly with test results

Figure 27: Direct comparison with the test results

4.4 High strength concrete

To easier see what the results say about high strength concrete, a plot was made with results side by side in different color for the higher strength concrete. This has been done for figure 28 and 29. Here we divided the results at 55 MPa and gave these values a color red. The trendlines for figure 28 are made from the corresponding dots.

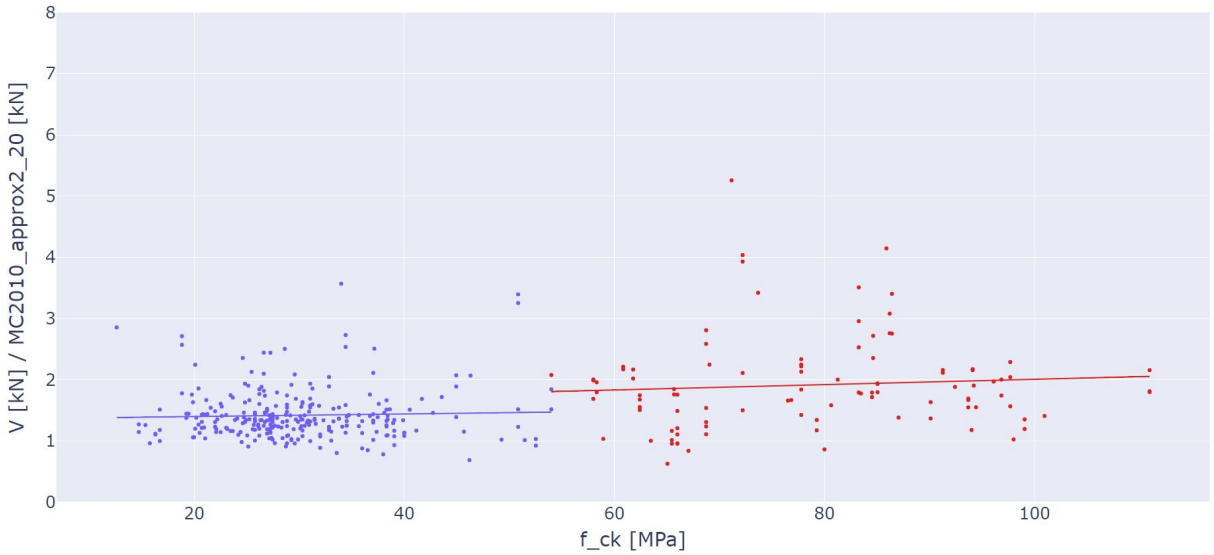


Figure 28: Shows the same as in figure 25 approx 2 dg = 20, but with red dots and trendline for concrete over 55 MPa

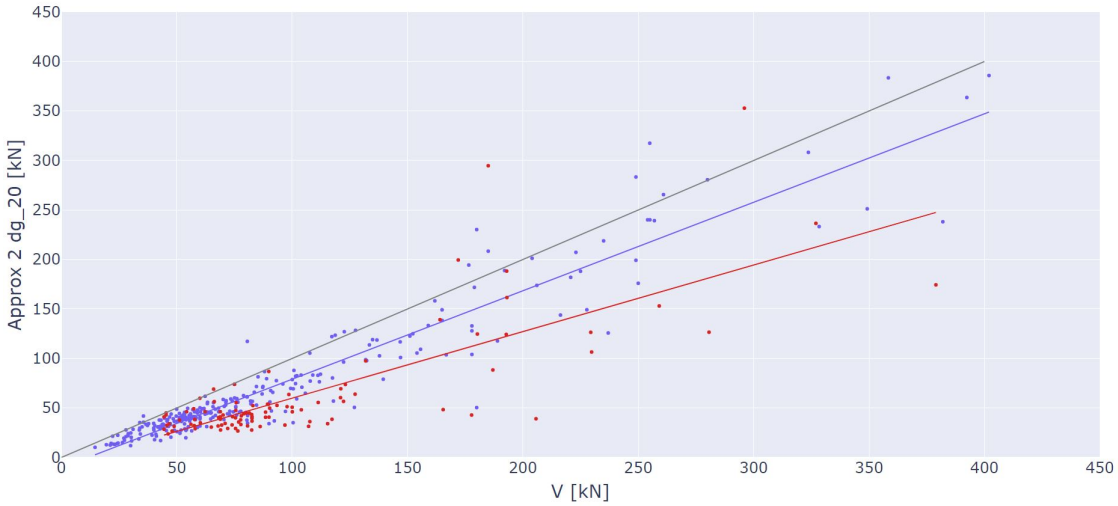


Figure 29: Shows the same as in figure 27 but with red dots and trendline for concrete over 55 MPa

4.5 Adjustment on variable for Model code 2010

To test what effect the rule that concrete strength over 70 MPa should be calculated with a maximum aggregate seize of 0, a plot was made. This can be directly compared with figure 29 which includes the rule.

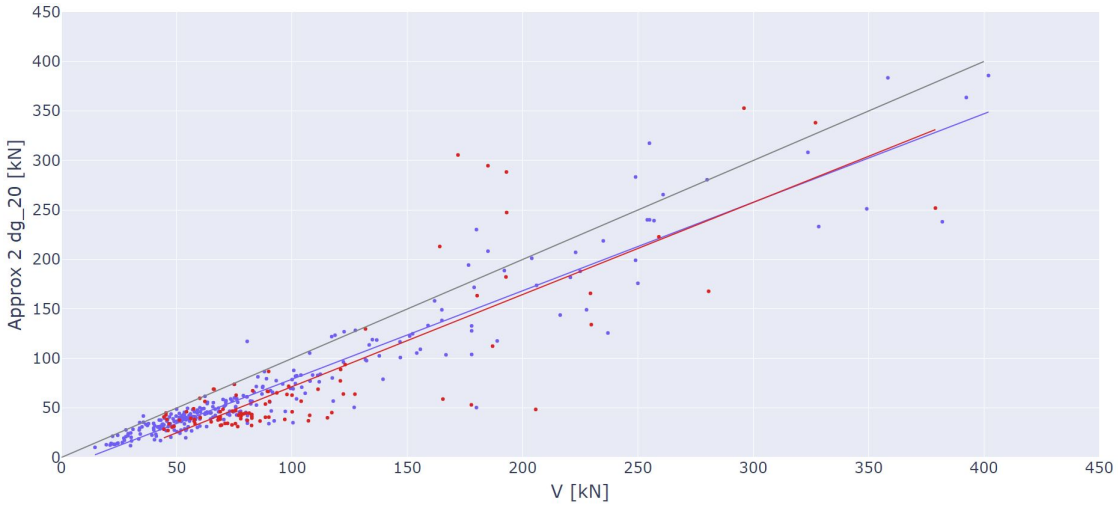


Figure 30: A plot of the approx 2 dg=20 results, but without the rule of dg=0 for concrete strength over 70 MPa

4.6 Validation and reliability

When it comes to interpreting the data, one often needs to look at what the expected ratio will be when using this model. This was calculated with the average results. At the same time the spread of the data points tells us how close to the expected outcome we will come. Here we use standard deviation and coefficient of variation to calculate. The Coefficient of variation is the Standard deviation divided by the average.

4.6.1 With safety margin

	MC_1	**MC_2_dg=20**	**MC_2_dg=28**	**EC2**	**New_EC2**
Expected value	2.985	2.326	2.270	1.645	1.779
Standard deviation	0.881	0.855	0.8620	0.431	0.145
Coefficient of variation	0.295	0.367	0.379	0.262	0.082

Table 2: The expected value and variation with safety margin

4.6.2 Without safety margin

	MC_1	**MC_2_dg=20**	**MC_2_dg=28**	**EC2**	**New_EC2**
Expected value	1.990	1.488	1.514	1.100	1.155
Standard deviation	0.346	0.248	0.331	0.083	0.059
Coefficient of variation	0.174	0.167	0.219	0.075	0.051

Table 3: The expected value and variation without safety margin

4.6.3 High strength concrete without safety margin

The results for Model code 2010 were analyzed further and were therefore also divided for high and normal strength concrete. This was done for Model code 2010 dg = 20, and the results can be found in the table below.

	Concrete <55 MPa	**Concrete >55 MPa**
Expected value	1.420	1.915
Standard deviation	0.167	0.583
Coefficient of variation	0.118	0.304

Table 4: The expected value and variation divided between normal and high strength concrete

4.6.4 Without the high strength adaptation

Here are the reliability results without equation 9, where dg is set to 0 for concrete with strength over 70. These results are as table 4, without safety margin and done for approximation 2, with dg = 20.

	Concrete <55 MPa	**Concrete >55 MPa**
Expected value	1.420	1.676
Standard deviation	0.167	0.423
Coefficient of variation	0.118	0.253

Table 5: The expected value and variation divided between normal and high strength concrete

5 Discussion

This chapter discusses and evaluates the content of our results as well as highlighting what we consider as important finds. We will discuss our contribution to the project, and how we chose to solve the tasks that appeared when helping with the development of fib. For the benchmarking and comparison part of the result, it is more open for discussion because of the interpretation of the graphs. Do we see any clutterings or anything worth noticing? Are some of the results too conservative and how do we evaluate the differences in the data with and without safety margins?

5.1 The contribution to the fib project

We expect our work to have made a significant contribution to the ongoing fib project, by making the foundation of further development, with our interpretation of Model code 2010. This comes from good feedback and a clear understanding of the standards while developing the code. This is still only a fraction of the projects end result, but still a step in the right direction. Our code seen in the light of the validation will be proven useful for any user of the code that wishes to further understand what the code returns. It is easy to get lost on a project as big as this one, and few users of the project will understand the code we have written to the same degree as us. This is why we have also benchmarked the results that our code provides and made comparisons with Eurocode 2. This will make the code more reliable. We also focused on following the google format that the project is written in. This is required to make it easier for other developers to understand what we have done and to keep consistency through the project as an open-source.

The work on the benchmarking will also help further work as well as confirming the quality of our code. The tests are pretty simple but make sure that the functions do what they are intended to do. This will be of help when more code and concepts are included in the projects. Our tests have been calculated by hand to see if we get the same value, which reduces the risk of mistakes dramaticly. When creating the tests, our goal was to try to test all parts of the code, this was partly done for our contribution. A large amount of inputs made it time consuming to test all edge cases, so we decided to test the main intention of the function. At a later stage, the tests will need changes because of the introduction of load, material and cross section classes.

The fact that our work is done as an open-source project was part of the reason why we chose this exact task. Work done in this way is honest work. It is made for any individual, where they have the right to alter the software or share it. This way of working has a mindset that can be recognized in NTNU's slogan "knowledge for a better world".

5.2 Evaluating the sorted results

We did a sorting by z, m_{ed}, f_{ck} as well as the distance from support, found in section 4.2 and 4.3 to try to identify strengths, weaknesses or irregularities. As explained earlier in 3.4 we divide our sorting date by variables we found interesting. For each sorting we evaluate each calculation done for the specific standard. When further evaluating these images, we have chosen to evaluate these at three points in particular:

- Where the main parts of the data are located. Is it close to 1? How conservative can we expect this to be?
- The spread of data. Are the data points close to each other or are they all over the chart?
- Do we see any other trends? Is there more inconsistency when reaching a certain value? Is there something that does not make sense?

Evaluating the internal moment arm z and the moment m_{ed} which have quite similar figures (see figure 18 and 19). Here we did not find anything particularly interesting to report. In the figures we can tell that most of the data have internal moment arms under 400mm and external moment close to zero. With many tests done for these areas a larger spread can also be expected. There are less data points for higher z and m_{ed} and as a result there is the same spread compared to tests as we can see. The expected value seems reasonable compared to what the average found to be in table 3. No area stands out with any irregularities.

When looking at the distance from support a we see tendencies to more conservative values for lower value of a. This can be because of the large amount of tests for small a, so we would expect a larger variation in the tests. This might also be because of the concept described in section 2.2.2, where the load forces are transferred directly to support instead of being handled as a shear problem. We do not see any other trends regarding the distance from support a.

When looking at the plotted results that are seen in figure 27 we mainly look at the spread of the data points. It is however easy to see that the trendline is far off the optimal line in approximation method 1. The spread can be confirmed in table 2, to go in decreasing order: approximation 1, approximation 2, Eurocode 2 and New Eurocode 2.

5.3 High strength concrete

Upon further observation, we found a significant amount of variation and high mean value considering high strength concrete. Surpassing 50-60 MPa, a significant contrast between Model code 2010 and Eurocode 2 was found. This may occur because of the fact that dg in equation 7 is set to 0 when dealing with concrete higher than 70 MPa. By comparing the results from figure 28 and 29 considering high strength concrete compared to concrete with lower strength (<55MPa), we fond tendencies to high strength being more conservative while looking at the trend lines from figure 29 for Model code 2010 approx 2 dg = 20.

Another factor to take into consideration is the amount of tests done for higher strength concrete. By looking at figure 29 we can observe that there are more tests in a clutter for the lower strength concretes then for high strength concrete. This makes the result from higher strength less reliable than for lower strength concrete. Table 4 confirms more spread for high strength concrete, even though there are fewer tests. Fewer test have less probability of having extreme results in either direction, but this can still be seen here. This might be because the difference in stress-strain condition, showen in figure 1. The fact that high strength concrete has a more brittle behavior with reduced ductility can be a cause to why we get more extreme results with more spread. There is also the fact that higher strength concrete is newer and has had less testing. The formulas for shear are complex and are in to a high degree based on empiric knowledge. It is to assume that is the reason why these parts show worse results.

We see a drastic change when we go past 50-60 MPa, for the Model code 2010 compared with Eurocode 2. By comparing the results from figure 28 and 29 considering high strength concrete to concrete with lower strength (<55MPa), it is easy to draw the conclusion that generally high strength concrete is more conservative in Model code 2010. When removing the condition on equation 7, the results in figure 30 appear. Here we can see that the trendlines seem to align for normal- and high strength concrete. Tables 4 and 5 show clearly the difference between normal and high strength concrete, as well as better results

when removing the condition on equation 7. Removing the rule does not make the Model code 2010 results for high strength concrete as good as normal strength concrete, but the results indicate an improvement.

5.4 Comparison of standards

When evaluating the standards based on the results from equation 5 in Model code 2010, equation 10 in Eurocode 2 and equation 14 in new Eurocode 2 to form an opinion on how close to the experiments that standards get. This is best done by using the capacities without the safety margins and the images in section 4.2. To get capacities close to the experiments, we want the values on the y-axis to be close to 1. From comparing these finds in shear, we can see that both Eurocode 2 and the new Eurocode 2 closer to the experiments than Model code 2010 on the placement of the main data and the spread. Model code 2010 approximation 1 is most conservative from these comparisons. This is expected because approximation level 1 is mainly used in the early stage with less inputs, it is also explained in section 2.3.1.

The main intention of the benchmarking is to make it clear that the users of a future python package can be mindful of which standard they use and how safe the specific code is. For shear without shear reinforcement the results show that a use of Eurocode 2 could be to prefer. This is not to say that a quick calculation with approx 1 would be wrong, but it would most likely be a lot more conservative than Eurocode 2 or new Eurocode 2. These conclusions should not be taken as a general rule for the other sections of Model code 2010 without further research.

5.5 Environmental changes by the use of digital standards

The future is unknown and that includes the construction industry. There will probably be new software, taller and more complex constructions, but the foundation may very well be based on the same basics. A 100 million dollar building may use a γ_c that is 1.5 or 1.4 a small and simple choice, but with huge emissions consequences. With more solutions more problems often follow. We are right now on the edge of putting away hand held rules and putting our faith in a program. This is the time to evaluate if the programs should look the same as the rules we have today or maybe adjust more to the future.

Our testing shows a tendency of leading in the safe direction. This situation would have been a dream 50 years ago. However, with a new environmental aspect, the safest approx may not be optimal. With 2.5 billion tons of CO2 coming from cement production an assessment of what a "safe future" is, should be well thought through. To achieving greater safety the use of more concrete is needed, which, in turn, results in increased carbon dioxide emissions First of all, the building needs to avoid collapse, but if there is any time to look through the basics for what the machines will calculate as "safe", it is now.

Our benchmarking illustrate what it means to use the code we have provided. This can help future users to choose the right calculation method. These are choices that are difficult to make if not the developers have already done an evaluation of what using this method will mean. With the aspect of both as a safe construction and a safe future in mind.

5.6 Source of error

The tests were taken from "Shear database for reinforced concrete members without shear reinforcement" [5]. Which is considered a common database when talking about shear, but not without uncertainties. The article itself tells us that some of the dimensions necessarily are lacking. " (...) these dimensions were not specified. In these cases, it was necessary to measure these dimensions from photos or drawings." This tells us that some of the tests might have had a breaking point higher or lower than what the shear capacity tells us. With the given information a certainty about sufficient anchorage/bond cannot be guaranteed, and may give some results with a low breaking point.

When conducting shear tests a short beam is necessary to ensure high shear stresses without exceeding the capacity for the moment. We found around 5% of the tests to be under 2.5 a/d (distance from support divide on height from reinforcement to the other edge). This can cause a path for the forces to go directly to the support, see figure 7. This may give the test more capacity than the general rules imply. Those occurrences are hard to tell, but may explain some of the conservative values found in figure 26 with short distance from the support.

The shear tests are based on a support that does not carry moment, where one is a roller and one is pinned. It can be hard to make this perfect, especially the roller, where friction quickly comes in. From the lack of data from [5], we cannot guarantee that this was satisfied when dealing with the shear data set.

Human error can happen when dealing with large amount of data, but being two can help with that. The data from the shear tests, we got from a pdf, which was copied and cross checked. We needed to cross check because we found some errors while copying the data. After one of us inputted the data, the other cross-checked it to confirm the accuracy of the inputs. The places where we had different inputs were reviewed closer. All data points that had a very high or low values were also examined more. A total of 374 tests were gone through with 10 variables, so some error in the data could have occurred.

5.7 Further work

The contribution we have done to fib is a small part of the overall product. It is then natural to continue the work on the fib project as further work. The contribution needed would depend on the . As told earlier the project is an open-source, so the project is under constant improvement and everyone has the opportunity to contribute. A specific contributing example could be to continue where we left off, chapter 7.3.6 "Design with stress fields and strut-and-tie models". There is also an opportunity to progress on our work on shear by including prestressing in the code. Making the code more compatible introducing classes for cross-section, loads and prestressing of some sort is also some work we would love to see be done. This would compress the overall code and most likely make changes easier later.

By continuing the validation, it would be interesting to do benchmarking on a larger sample size for high strength concrete, to further evaluate the results. We would also like to see benchmarkings done for the other parts we have coded. At a later stage it would also be interesting to include a typical moment and axial test in the benchmarking. While using the final package, it would also be an interesting aspect to make an overview letting the user see the benchmarkings that have been done to further evaluate and choose what standard to use with respect to conservativity and accessibility.

6 Conclusion

In this thesis, development of functions and tests regarding shear for Model code 2010 for fib - International Federation for Structural Concrete has been done. The testing and validating of the code is our interpretation on the Model code 2010 and what the fib project expected at this stage in time based on feedback on our code. This is done as an open-source project, so this will be a contribution to everyone that works with structural engineering and would like to use the package. The way we have chosen to work with this project is by testing and validating the code with our supervisor, making the code more robust and minimizes potential errors. This code lays a basis for what to come in the projects and serves as a good start for further development.

During the benchmarking done by using existing shear-tests [5], we compared the sorting of different variables and primarily focused on evaluating the laboratory test divided by the prediction of the standard. The goal was to achieve a value close to one, indicating a high level of accuracy. Additionally, we analyzed the spread of the data and looked for any unusual trends. While varying the internal moment arm z and the moment m_{ed}, we found no significant finds while looking at the data. However, by examining the distance from support a, we discovered more conservative values for lower values of a. The trend may be attributed to a large sample size for lower values of a or a instance where the tests directly transferred the forces to the support instead of handling it as a shear problem.

The last variable we investigated was the sorting of high strength concrete, which proved to be particularly interesting. By utilizing the shear tests [5] and analyzing the spread in concrete strength, our graphs in figure 25 revealed that higher strength concrete have more variation then for lower strength concrete in the tested standards. Which, in turn gives bigger differences between the test prediction and the standards prediction. We see these tendencies especially for Model code 2010, and to some degree Eurocode 2 and the newer version of Eurocode 2. To further investigate the high strength concrete we plotted trendlines for f_{ck} higher or lower then 55MPa in chapter 4.4, which further showed the difference in spread being higher for high strength concrete.

The main findings from comparing the standards Model Code 2010, Eurocode 2, and the new Eurocode 2 indicate that overall, both Eurocode 2 and the new Eurocode 2 outperform the Model Code 2010 in terms of shear without shear reinforcement. This comes to show when Model code 2010 often gives results more on the conservative side compared to Eurocode 2 and new Eurocode 2. This is comfimed in section 4.6 where both the expected value and the spread improves when going from Model code 2010 to Eurocode 2, and then again when moving from Eurocode 2 to new Eurocode 2.

Bibliography

[1] *Building the modern world: Concrete and our environment*. https://www.sciencemuseum.org.uk/objects-and-stories/everyday-wonders/building-modern-world-concrete-and-our-environment. Accessed: 2023-05-10.

[2] *The Most Surprising Opensource Statistics And Trends in 2023*. https://blog.gitnux.com/opensource-statistics/. Accessed: 2023-05-10.

[3] European committee for standardization. 'Eurocode 2 Design of concrete structures Part 1-1: General rules and rules for buildings'. In: (2004).

[4] International Federation for Structural Concrete (fib). 'Model Code 2010 Final draft Volume 2'. In: (2012).

[5] Karl-Heinz Reineck et al. 'Shear database for reinforced concrete members without shear reinforcement'. In: *Structural Journal* 100.2 (2003), pp. 240–249.

[6] S.Smeplass S.jacobsen M.Maage. *Concrete technology*. Norwegian University of Science and Technology, 2022.

[7] Abid A Shah and Yuri Ribakov. 'Recent trends in steel fibered high-strength concrete'. In: *Materials & Design* 32.8-9 (2011), pp. 4122–4151.

[8] Mikael Hallgren. 'Flexural and shear capacity of reinforced high strength concrete beams without stirrups'. PhD thesis. KTH Royal Institute of Technology, 1994.

[9] Féderation internationale du béton (fib). *Towards a rational understanding of shear in beams and slabs*. Féderation internationale du béton (fib), 2018.

[10] Michael P Collins et al. 'An adequate theory for the shear strength of reinforced concrete structures'. In: *Magazine of Concrete Research* 60.9 (2008), pp. 635–650.

[11] Jan Arve Øverli. *TKT4123 Mekanikk 2*. TKT4123 Mekanikk 2 [Power Point]. 2019.

[12] Evan C Bentz, Frank J Vecchio and Michael P Collins. 'Simplified modified compression field theory for calculating shear strength of reinforced concrete elements'. In: *ACI structural journal* 103.4 (2006), p. 614.

[13] Mohamed A El Zareef, Moussa S Elbisy and Moataz Badawi. 'Evaluation of code provisions predicting the concrete shear strength of FRP-reinforced members without shear reinforcement'. In: *Composite Structures* 275 (2021), p. 114430.

[14] John Cairns. 'Bond and anchorage of embedded steel reinforcement in fib Model Code 2010'. In: *Structural Concrete* 16.1 (2015), pp. 45–55.

[15] *Hva er en standard?* https://standard.no/standardisering/hva-er-en-standard/. Accessed: 2023-06-07.

[16] Aurelio Muttoni and Miguel Fernández Ruiz. 'Levels-of-approximation approach in codes of practice'. In: *Structural Engineering International* 22.2 (2012), pp. 190–194.

[17] Viktor Sigrist et al. 'Background to the fib Model Code 2010 shear provisions–part I: beams and slabs'. In: *Structural Concrete* 14.3 (2013), pp. 195–203.

[18] European committee for standardization. 'Eurocode 2: Design of concrete structures — Part 1-1: General rules and rules for buildings, bridges and civil engineering structures'. In: (2022).

[19] Andrew M St Laurent. *Understanding open source and free software licensing: guide to navigating licensing issues in existing & new software.* " O'Reilly Media, Inc.", 2004.

[20] Jing Wang, Patrick C Shih and John M Carroll. 'Revisiting Linus's law: Benefits and challenges of open source software peer review'. In: *International Journal of Human-Computer Studies* 77 (2015), pp. 52–65.

[21] James Edward Corbly. 'The free software alternative: Freeware, open source software, and libraries'. In: *Information Technology and Libraries* 33.3 (2014), pp. 65–75.

[22] *The Open Source Definition.* https://opensource.org/osd/. Accessed: 2023-04-24.

[23] Scott Chacon and Ben Straub. *Pro git.* Springer Nature, 2014.

[24] Erik Arntzen and June Tolsby. 'Studenten som forsker i utdanning og yrke: vitenskapelig tenkning og metodebruk'. In: (2010).

[25] Yan Xue et al. 'Multi-sector partnerships in the urban development context: A scoping review'. In: *Journal of cleaner production* 268 (2020), p. 122291.

[26] Contributors only. *black.* URL: https://github.com/psf/black?fbclid=IwAR0gwkLcVjxcGsa2BvV2H-fi7A7sm9jD5mfFj-KmX9kh2CuHPn-mcu-0hT8. (accessed: 06.06.2023).

[27] Rob Ruana. *Example Google Style Python Docstrings.* URL: https://sphinxcontrib-napoleon.readthedocs.io/en/latest/example_google.html. (accessed: 07.06.2023).

[28] Holger Krekel et al. *pytest 7.3.1.* URL: https://pypi.org/project/pytest/. (accessed: 06.06.2023).

[29] Stefan Jacobsen. *TKT4215 Betongteknologi 1.* TKT 4215 Concrete Technology 1 [Power Point]. 2022.

A Python approx 2 dg=20 sorting and plotting

How we plotted the results with the use of the code from Model code 2010 approximation 1.

```
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go

df = pd.read_excel('inputs_bench.xlsx')

### -----model code----
def epsilon_x(
    E_s: float,
    As: float,
    z: float,
    loads: dict,
) -> float:
    """Calculate the longitudinal strain from a distance z

    fib Model Code 2010, eq. (7.3-16)

    Args:
        E (float): The E-modulus to the material in MPa
        As (float): The cross-section area of reinforcement in mm^2
        z: (float): The effective shear depth in mm
        loads (dictionary) The given loads in a dictionary:
            Med (Float): The positive moment working on the material in Nmm
            Ved (float): The positive shear force working on the material in N
            Ned (float): The normal force working on the material in N with
            positive sign for tension and negative sign for compression
            delta_E (float): The eccentricity of the axial load due to
            imperfection in the construction with distance in mm as a positive
            value in compression direction

    Returns:
        float: The longitudinal strain"""

    return max(
```

```
        (
            (1 / (2 * E_s * As))
            * (
                (abs(loads.get('Med')) / z)
                + abs(loads.get('Ved'))
                + loads.get('Ned') * ((1 / 2) + (loads.get('delta_e') / z))
            )
        ),
        0,
    )


def v_rdc(
    approx_lvl,
    fck: float,
    z: float,
    bw: float,
    dg: float,
    E_s: float,
    As: float,
    loads: dict,
    alpha: float = 90.0,
    gamma_c: float = 1.5,
) -> float:
    """Calculate shear resistance of a web / slab without shear reinforcement.

    fib Model Code 2010, Eq. (7.3-17)

    Args:
        approx_lvl (int): Approximation level for concrete
        fck (float): Characteristic strength in MPa
        z (float): The length to the areasenter of cross-section in mm
        bw (float): Thickness of web in cross section in mm
        dg (float): Maximum size of aggregate
        E_s (float): The E_s-modulus to the materialb in MPa
        As (float): The cross-section area in mm^2
        loads (dictionary) The given loads in a dictionary:
            Med (float): The positive moment working on the material in Nmm
            Ved (float): The positive shear force working on the material in N
            Ned (float): The normal force working on the material in N with
```

```
            positive sign for tension and negative sign for compression
            delta_E (float): The eccentricity of the axial load due to
            imperfection in the construction with distance in mm as a positive
            value
        alpha (float): Inclination of the stirrups in degrees
        gamma_c (float): Concrete safety factor

    Returns:
        float: The design shear resistance attributed to the concrete
    """

    if approx_lvl == 1:
        return v_rdc_approx1(fck, z, bw, gamma_c)

    if approx_lvl == 2:
        return v_rdc_approx2(fck, z, bw, dg, E_s, As, loads, gamma_c)

    raise ValueError("Invalid approx level")


def v_rdc_approx1(
    fck: float,
    z: float,
    bw: float,
    gamma_c: float = 1.5,
) -> float:
    """Calculate shear resistance for concrete with approx level 1

    For members with no segnificant axal load, with fyk <= 600 Mpa,
    fck <= 70 Mpa and with maximum aggrigate size of no less then 10mm.

    fib Model Code 2010, Eq. (7.3-17) and (7.3-19)

    Args:
        fck (float): The characteristic compressive strength in MPa.
        z (float): The length to the areasenter of cross-section in mm
        bw (float): Thickness of web in cross section
        gamma_c (float): Safety factor for concrete
```

```
    Returns:
        float: Design shear resistance without shear reinforcement
    """
    fsqr = min(fck**0.5, 8)
    kv = 180 / (1000 + 1.25 * z)

    return (kv * fsqr * z * bw) / gamma_c


def v_rdc_approx2(
    fck: float,
    z: float,
    bw: float,
    dg: float,
    E_s: float,
    As: float,
    loads: dict,
    gamma_c: float = 1.5,
) -> float:
    """Calculate shear resistance for concrete with approx level 2

    In higher strength concrete and light-weight aggregate concretes,
    the fracture surface may go through the aggregate particles,
    rather then around, reducing the crack roughness

    fib Model Code 2010, Eq. (7.3-17), (7.3-20) and (7.3-21)

    Args:
        fck (float): Characteristic strength in MPa
        z (float): The length to the areasenter of cross-section in mm
        bw (float): Thickness of web in cross section
        dg (float): Maximum size of aggregate
        E_s (float): The E_s-modulus to the materialb in MPa
        As (float): The cross-section area in mm^2
        loads (dictionary) The given loads in a dictionary:
            Med (float): The positive moment working on the material in Nmm
            Ved (float): The positive shear force working on the material in N
            Ned (float): The normal force working on the material in N with
            positive sign for tension and negative sign for compression
```

```
            delta_E (float): The eccentricity of the axial load due to
            imperfection in the construction with distance in mm as a positive
            value
        gamma_c (float): Concrete safety factor

    Returns:
        float: Design shear resistance without shear reinforcement
    """

    if fck > 70:
        dg = 0
    fsqr = min(fck**0.5, 8)
    epsilonx = epsilon_x(E_s, As, z, loads)
    k_dg = max(32 / (16 + dg), 0.75)
    kv = (0.4 / (1 + 1500 * epsilonx)) * (1300 / (1000 + k_dg * z))

    return (kv * fsqr * z * bw)
### -----model code end----

b_list = df['b [mm]'].tolist()
h_list = df['h [mm]'].tolist()
d_list = df['d [mm]'].tolist()
z_list = df['z [mm]'].tolist()
f1c_list = df['f1c [Mpa]'].tolist()
fck_list = df['fck [Mpa]'].tolist()
Asl_list = df['Asl [mm^2]'].tolist()
f_cd_list = df['f_cd'].tolist()
M_rd_list = df['M_rd'].tolist()
M_ed_list = df['M_ed'].tolist()
V_list = df['V [kN]'].tolist()
a_list = df['a'].tolist()

MC2010_approx2_dg20=[]


def create_load_dict(Med: float, Ved: float, Ned: float, delta_e: float) -> dict:
    """returns dictionary assosiated with loads"""
    dictionary = {'Med': Med, 'Ved': Ved, 'Ned': Ned, 'delta_e': delta_e}
```

```
    return dictionary

fck_list = [x + 8 for x in fck_list]

for i in range(len(df)):
    MC2010_approx2_dg20.append(v_rdc(2, fck_list[i], z_list[i], b_list[i], 20,
        205000, Asl_list[i],
        create_load_dict(M_ed_list[i]*1000000,V_list[i]*1000,0,0))/1000)


approx2_20 = [k/j for k,j in zip(V_list, MC2010_approx2_dg20)]


# # <---------- Plot data MC2010_approx1---------->
x, y = zip(*sorted(zip(b_list, approx2_20)))
fig = px.scatter(x=x, y=y)
fig.update_layout(xaxis_title="Width [mm]", yaxis_title="V [kN] /
    MC2010_approx2_20[kN]")
fig.update_layout(yaxis_range=[0,8])
fig.show()
```

B Test code

Here is one of the test. Where the V_{rdc} function is confirmed to give the right values.

```
import math

import pytest

from structuralcodes.codes.mc2010 import _concrete_shear


def create_load_dict(Med: float, Ved: float, Ned: float, delta_e: float) -> dict:
    """returns dictionary assosiated with loads"""
    dictionary = {'Med': Med, 'Ved': Ved, 'Ned': Ned, 'delta_e': delta_e}
    return dictionary

@pytest.mark.parametrize(
```

```
    '''approx_lvl, fck, z, bw, dg, E_s, As, loads, alpha, gamma_c, expected''',
    [
        (1, 35, 180, 300, 0, 0, 0, create_load_dict(0, 0, 0, 0), 0, 1.5, 31294),
        (1, 35, 200, 300, 0, 0, 0, create_load_dict(0, 0, 0, 0), 0, 1.5, 34077),
        (2, 35, 140, 300, 16, 21e4, 2000, create_load_dict(40e6, 2e4, 1000, 50),
            0, 1.5,
         48828),
        (2, 35, 140, 300, 32, 21e4, 2000, create_load_dict(40e6, 2e4, 1000, 50),
            0, 1.5,
         50375),
        (3, 35, 200, 300, 32, 21e4, 2000, create_load_dict(40e6, 2e4, 1000, 50),
            1.5, 1.5,
         67566),
        (3, 35, 200, 300, 32, 21e4, 2000, create_load_dict(40e6, 20e6, 1000, 50),
            1.5, 1.5,
         0),
    ],
)
def test_v_rdc(
    approx_lvl, fck, z, bw, dg, E_s, As, loads, alpha, gamma_c, expected
):

    """Test the v_rdc function."""
    assert math.isclose(_concrete_shear.v_rdc(
              approx_lvl, fck, z, bw, dg, E_s, As, loads, alpha, gamma_c
              ),
       expected, rel_tol=0.001)
```