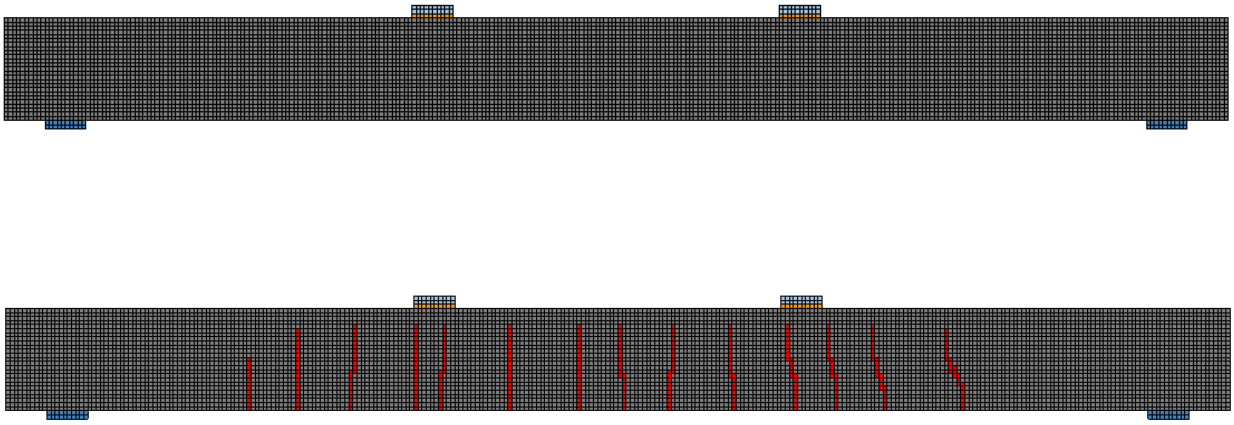
CHALMERS
UNIVERSITY OF TECHNOLOGY



Automating Crack Inclusion in Finite Element Modelling Using Data from Structural Health Monitoring

Refinement on Assessing the Structural Behaviour of Reinforced Concrete

Master's thesis in Master Programme Structural Engineering and Building Technology

MARCUS JANSÉN SVÄRD
OSKAR MAGNUSSON

DEPARTMENT OF ARCHITECTURE AND CIVIL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Master's thesis ACEX30
Gothenburg, Sweden 2021

Master's thesis ACEX30, 2021

Automating Crack Inclusion in Finite Element Modelling Using Data from Structural Health Monitoring

Refinement on Assessing the Structural Behaviour of Reinforced Concrete

MARCUS JANSÉN SVÄRD
OSKAR MAGNUSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Architecture and Civil Engineering
Division of Structural Engineering
Concrete Structures
Chalmers University of Technology
Gothenburg, Sweden 2021

Automating Crack Inclusion in Finite Element Modelling Using Data from Structural Health Monitoring
Refinement on Assessing the Structural Behaviour of Reinforced Concrete
MARCUS JANSÉN SVÄRD, OSKAR MAGNUSSON

Supervisor: Dr. Mattis Blomfors, Norconsult AB.
Supervisor: Dr. Carlos Gil Berrocal, Department of Architecture and Civil Engineering
Supervisor: Prof. Ignasi Fernandez, Department of Architecture and Civil Engineering
Examiner: Prof. Ignasi Fernandez, Department of Architecture and Civil Engineering

Master's Thesis 2021
Department of Architecture and Civil Engineering
Division of Structural Engineering
Concrete structures
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Example of a finite element mesh with and without crack inclusion. Red parts indicate cracked regions. Authors' own illustration.

Typeset in LaTeX
Department of Architecture and Civil Engineering
Gothenburg, Sweden 2021

Automating Crack Inclusion in Finite Element Modeling Using Data from Structural Health Monitoring
Refinement on Assessing the Structural Behaviour of Reinforced Concrete
Marcus Jansén Svärd
Oskar Magnusson
Department of Architecture and Civil Engineering
Chalmers University of Technology

Abstract

Inadequate monitoring of structural damage in reinforced concrete infrastructure is one of the most common causes of structural deterioration and at worst, failure, which entail unnecessary risks and costs, both financial and environmental. Furthermore, Finite Element (FE) tools of today use generalised methods to predict, rather than describe, the formation of cracks when evaluating reinforced concrete structures. In an attempt to circumscribe these problems, this thesis aimed to refine the structural assessment of reinforced concrete structures by automating crack inclusion, using data from Distributed Optical Fibre Sensors (DOFS), in FE Analysis (FEA). DOFS, installed inside reference test beams, were utilized to measure strain data during laboratory tests, which was then processed in a series of scripts and incorporated in an FE model. The scripts were written in such a way that the process of locating cracked elements and assigning these their corresponding weakened properties, was automated.

It was found that including cracks, translated from DOFS strain data, in FEA gave a far better numerical estimation of crack widths compared to a traditional model without crack inclusion. Moreover, the crack inclusion itself is highly functional and applicable, and conducting further laboratory test with DOFS can further improve the method. Furthermore, up to the load where strain data from DOFS was gathered, an improved representation of the structural behavior could be noted when comparing the FE model with crack inclusion to a traditional FE model without crack inclusion.

Keywords: Reinforced concrete, finite element analysis, structural health monitoring, distributed optical fibre sensors, automated crack inclusion, concrete beams.

Acknowledgements

This thesis was written and completed at the department of Architecture and Civil Engineering, at the Division of Structural Engineering within Concrete Structures at Chalmers University of Technology in collaboration with Norconsult AB.

We would first and foremost like to thank our thesis examiner Prof. Ignasi Fernandez and supervisor Dr. Carlos Gil Berrocal at the department of Architecture and Civil Engineering at Chalmers University of Technology. Not only for proposing this interesting and educational topic to us, but for always keeping their, in these peculiar times, fictitious door open at all times for all our questions and inquiries about the thesis work and for providing us with the material needed to accomplish our research.

We would also like to thank our supervisor Dr. Mattias Blomfors at Norconsult AB. Dr. Blomfors has been receptive to all our questions and enlightened us in an educational manner, being attentive with our full understanding at all times. Without their aid, the extent to which this thesis work was carried out would not have been possible and we will always be grateful for their assistance and mentorship.

We would also like to acknowledge Gustav Nilsson and Rasmus Öhman at the department of Architecture and Civil Engineering at Chalmers University of Technology as well as copy editor Sara Lowe, post graduate at the department of Culture, History and Philosophy at Åbo Academy, Finland as the second readers of this thesis. We are indebted to them for their valuable comments and constructive criticism, which helped improve this thesis.

Finally, we must express our profound gratitude to our respective parents and significant others for providing us with unfailing support and continuous encouragement throughout our years of study and through the process of completing and writing this thesis. This accomplishment would not have been possible without them.

Marcus Jansén Svärd, Gothenburg, June 2021
Oskar Magnusson, Gothenburg, June 2021

Notations

Abbreviations

DCA	Discrete Crack Approach
DIANAIE	DIANA Interactive Environment
DIC	Digital Image Correlation
DOFS	Distributed Optical Fibre Sensors
FE	Finite Element
FEA	Finite Element Analysis
FEC	Finite Element Model with Crack Inclusion
FEM	Finite Element Method
FET	Traditional Finite Element Model
OFDR	Optical Frequency Domain Reflectometry
PDE	Partial Differential Equation
RC	Reinforced Concrete
SCA	Smeared Crack Approach
SHM	Structural Health Monitoring
SLS	Serviceability Limit State
SRA	Strain Reading Anomalies
STA	Swedish Transport Administration
ULS	Ultimate Limit State
WEA	Weakened Element Approach

Roman upper case letters

A_g	Plastic elongation at maximum force
E_{cm}	Young's modulus of concrete
E_{s}	Modulus of elasticity of reinforcement steel
G_{C}	Compressive fracture energy of concrete
G_{F}	Tensile fracture energy of concrete
K_{ε}	Strain coefficient
K_{T}	Temperature coefficient

Roman lower case letters

f_{cm}	Mean compressive strength of concrete
f_{ct}	Tensile strength of concrete
f_{ctm}	Mean tensile strength of concrete
f_{t}	Residual tensile strength of concrete in weakened elements
f_{u}	Ultimate strength of reinforcement steel
f_{y}	Yield strength of reinforcement steel
h_{eq}	Equivalent length
k_{n}	Normal stiffness modulus
k_{t}	Shear stiffness modulus
$l^{-}_{\mathrm{l,i}}$	Left end of transmission length where slip occurs (crack divider), i denotes the i-th crack of the beam
$l^{+}_{\mathrm{l,i}}$	Right end transmission length where slip occurs (crack divider), i denotes the i-th crack of the beam
s_0	Linearised initial slip section
s_1, s_2, s_3	Relative slip displacement sections
w	Crack width / crack opening
$w_{\mathrm{cr,i}}$	Crack width of the i:th crack
w_{s}	Crack limit in tension softening curve
w_{ult}	Ultimate crack limit in tension softening curve
y_{bar1}	Y-coordinate of bar 1 in beam section
y_{mid}	Y-coordinate of middle layer in beam section

Greek letters

α	Exponent
$\Delta\gamma$	Change in wavelength
ΔT	Change in temperature
ε	Strain along the fibre
ε_1	Cracking strain
$\varepsilon_{\text{bar1}}$	Strain in bar 1
$\varepsilon^{\text{DOFS}}$	Strain measured with DOFS
$\varepsilon_{\text{bot}}^{\text{fict}}$	Fictitious strain in bottom strand of beam
$\hat{\varepsilon}$	Strain varying linearly between cracks
ε_{mid}	Strain in the middle layer of the beam section
$\varepsilon_{\text{wc,ult}}$	Ultimate strain of cracked element
γ	Reference wavelength
ν	Poisson's ratio
ρ	Reinforcement ratio
σ_{s}	Stress at w_{s}
τ	Shear-stress growth
τ_{f}	Ultimate shear stress
τ_{max}	Maximum stiffness stress

Contents

List of Figures

List of Tables

1
Introduction

Cracking is a phenomenon in reinforced concrete (RC) members that is not only expected but also necessary to maintain the estimated ductility and ultimate capacity in terms of tensile forces from design. Cracking is essential in this sense and its effects have predominantly no negative influence on the structural behaviour of an RC member, provided that the cracks are controlled correctly (Al-Emrani et al., 2013). Cracks and crack propagation are two of the most difficult aspects to model using the Finite Element Method (FEM). With methods used today, one must either have all the information about the cracks, (position, width etc.), or else constitutive relations will approximate both position and pattern of the cracks. In other words, modelling cracks precisely is almost impossible and therefore an approximation of their behaviour is generally used. This approximate localisation of cracks creates problems when considering the monitoring of the structural damage during the service life of an RC element.

Assessment methods for Structural Health Monitoring (SHM), are of great interest in this regard. Despite the affirmance - regarding the service life of an RC member - being of utmost importance, it is with today's monitoring techniques not possible to asses an RC structure to its full extent in this state. For any such method to be absolutely reliable, firstly the data collection i.e. the detection of cracks, their widths and positions, need to be gathered in real time. Secondly, and most importantly, the information needs to be translated and interpreted so that structural damage can be detected more or less instantly.

Furthermore, the lack of modern technologies to monitor RC structures entails other problems, mainly linked to the service of structures. The underlying idea is based on the increased capacity in the Serviceability Limit State (SLS), which implies that both sustainability and financial aspects have a great influence. To create a sustainable building industry, knowledge and tools are needed in order to achieve this, and the usage of SHM is one piece of the puzzle. Being able to prolong the service life of a structure, for example, will have a positive impact on total emissions caused by the bridge during its life-cycle , the total maintenance costs etc. To clarify these examples, the total emissions from a structure, such as a bridge, will be spread out over a longer time without a new bridge having to be constructed and maintenance or repair work can be conducted directly when the probability for structural damage is high, rather than as a reactive solution for already damaged elements. Further, the extent of the structural damage can be accounted for in a more exact manner which will lead to financial benefits in savings both in time and resources. As a

reference, the total cost increase of damage of bridges in Sweden has during the last 5 years amounted to SEK 300 million comparing to earlier estimates (Honauer & Ödeen, 2019).

Future demands will set high requirements on new constructions and even higher on existing ones, but by means of SHM, new structures can be built more efficiently and with more control over time whilst old structures can be modified and enhanced to meet the new demand (Berrocal, Fernandez, Bado et al., 2021).

1.1 Background

The demands on the world's infrastructure, new and existing, will increase due to population growth, urbanization and continued globalization. By the year 2050, the United Nations has predicted that 68% of the world's population will live in urbanized areas, which will increase both the need and performance requirements of existing and new infrastructure (United Nations, 2018). In addition to an expected increase of users, the evolution in the road bound freight transportation sector will increase load-carrying demands on bridges. The use of longer and heavier vehicles will become more common and streamline transportation, according to Chistidis and Leduc (2009), which will be beneficial both environmentally and financially. The number of trucks will decrease and though a single vehicle will emit more compared to a traditional one, the net emission rate will nevertheless decrease due to the increased cargo capacity. This would, however, in turn lead to the infrastructure being subjected to an increased quantity of loads of greater magnitude at more frequent intervals which in turn instigates a re-evaluation of bridge design. As aforementioned, the demand for higher load capacity is further complicated by the sustainability requirements - both environmental and financial - and thus, standards to meet all are imminent.

One of the most common causes of bridge failure is poor or even neglected maintenance and inspections (Lin & Yoda, 2017). The procedures are often costly and time-consuming, with significant risks of human error. According to the Swedish Transport Administration (STA), inspections of bridges under their auspices takes place at least every six years. The condition of the bridge population is said to be good due to rigorous maintenance helps prolong the service life of bridges in a satisfying manner. Furthermore, according to the STA, many of the existing bridges in Sweden are designed to handle such a load increase as aforementioned (Trafikverket, 2020). However, it can be argued that even though a responsible authority has a strict and well-functioning program for inspections and maintenance, mistakes and miscalculations can certainly occur. The structural damage may also be undetectable with methods prescribed and could thus be overlooked. During the time frame between inspections, the damage could then hypothetically develop during a six-year cycle before any preventive action is taken, at which point the damage might be irreversible.

Monitoring structural damage of existing and new infrastructure is paramount to

ensure improved control over the behaviour of the structure over time. As mentioned above, one such technology is SHM through Distributed Optical Fibre Sensors (DOFS), which for several reasons are of high interest as a remedy to this particular problem. The reason commonly invoked is that DOFS themselves are relatively cheap, even though the technology that performs the measurements is expensive. As a solution, the technology has high sustainability qualities and pose a less complicated installation- and monitoring process than other systems (de Sousa Barrias, 2018).

1.2 Aim and objectives

The aim of this Master's thesis was to develop a way of automatically implementing strain data from SHM into Finite Element Analysis (FEA) of RC structures and also investigate the performance of FEA with crack inclusion compared to a traditional Finite Element (FE) model.

To achieve the aim, our inquiry was carried out following three main objectives:

- Automate the inclusion of cracks from Distributed Optical Fibre Sensors into an FE model.
- Calibrate the model using experimental data.
- Analyse the performance of the FE model with crack inclusion and compare it with a traditional FE model.

1.3 Methodology

To fulfill the aim of the thesis, the effects of cracking in RC members was reviewed and systematised by studying existing literature. As the thesis is built upon previous works by Mattias Blomfors, PhD, and our supervisor Carlos G. Berrocal, these were further in depth reviewed and briefely summarised.

A Python script was developed which generates a reinforced concrete beam in the FE analysis tool DIANA (DIANA FEA, 2020) using a reference beam previously analysed by Berrocal, Fernandez, Bado et al. (2021). This script was named *Mesh Generator* and can be found in appendix A. As the name implies, the script generates the geometry, assigns all material parameters except for the cracked regions and generates the FE mesh for a specific structure.

Furthermore, a Matlab-script, named *Crack Translator*, was developed, which translates the strain data given by the DOFS into crack widths and locations over the beam. This script can be found in appendix B. The crack widths were then converted into weakened material properties using methods described by Blomfors (2021) and

were, along with the crack coordinates, exported into .csv-files. These files were then imported into another Python-script which processed this information and found the corresponding elements in the previously generated FE mesh in DIANA Interactive Environment (DIANAIE) based on the coordinates extracted using the information from the Crack Translator. These elements were then assigned the corresponding weakened material parameters. This script was named the *Crack Generator* and can be found in appendix C.

A third Python script was then developed which assigns specific analyses to the RC beam, namely the *Analysis Generator* which can be found in appendix D which, depending on the input arguments, conducted an analysis on either the FE model with crack inclusion or the one without. These FE analyses were then compared with laboratory tests of an RC reference beam where SHM had been included in terms of DOFS.

Along with the analysis that included cracks as weakened elements, a second analysis was conducted without crack inclusion, i.e. a traditional smeared crack approach. This was done to find and compare differences and similarities in the results between the two approaches.

Lastly, the *Mesh Generator* was modified to match the final behaviour of the beam. This was needed since the tests of the reference beams were conducted on two different occasions, hence the material parameters of the concrete needed to be updated to match the strength increase due to ageing.

In figure 1.1 is a flow chart describing the different parts of the thesis work.

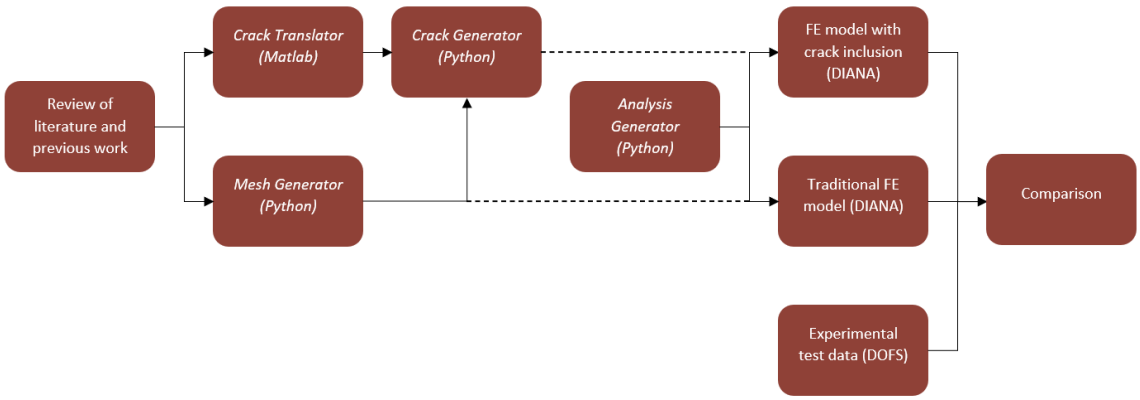


Figure 1.1: *Flow chart describing the different parts of the thesis work and how they link together.*

1.4 Specification of issue under investigation

Existing nonlinear FE crack models are insufficient for some applications. Currently there are two major methods applied to include cracking and its influence on the

structural behaviour; the smeared crack approach (SCA) and the discrete crack approach (DCA). Both have proven to provide results that are correct in many aspects but are limited in both their application and simulating all crack behaviours, such as pre-existing cracks (Blomfors, 2021) (for further reading see chapter 4). By providing a framework based on SHM data from laboratory tests to nonlinear FE models, in form of automated code, the intention was to improve the FE analysis by incorporating cracks and evaluate their influence on the behaviour of the beam.

As described in the methodology and background, the SHM data was provided from laboratory tests using DOFS, which has shown great potential to provide accurate data, that can be implemented in a reasonably uncomplicated approach (Berrocal, Fernandez, Bado et al., 2021).

The thesis is an extension of previous works by Berrocal, Fernandez, Bado et al. (2021) and Blomfors (2021), which constitute the theoretical framework of this thesis and what it seeks to accomplish, i.e., automate the raw data from DOFS into FEM based analysis to improve the performance of crack behaviour and in addition, present a powerful tool for damage assessment of RC structures in the future.

1.5 Scope

One major exclusion in the thesis is one of the most common causes of deterioration of RC members; reinforcement corrosion. This was done since the focus was cracking and crack behaviour due to external loading and automate crack information into FEA, and not the effects of corrosion or other strength reducing effects.

The RC beams were subjected to flexural tests in four-point bending; hence the predominant deformations were caused by bending, i.e. the beams were statically determined elements (simply supported), without any axial force.

The thesis is restricted to solely beam elements and other concrete elements such as slabs, are excluded from the analysis.

1.6 Limitations

Due to the complexity of the thesis, the relatively short time span limited FE-modelling to 2D. This consequently meant that no out-of-plane crack investigations were performed.

The DOFS data was accurately measured up to a load level of 30 kN in a pre-cracking test at an age of 14 days after the beam was cast. The full failure test was not conducted until an age of 507 days after casting, which consequently entails that no DOFS data was given for the load range between the aforementioned load level and failure. In extension, this means that cracking could not be validated through

the entire state II.

No experimental tensile tests were performed on the concrete used in the test beams. Therefore, the tensile strength properties were estimated through EN-1992-1-1 (2004). This added an uncertainty in estimating the tensile strength of concrete for the FE models.

Further, FE modelling was limited to utilizing Weakened Element Approach (WEA) and Smeared Crack Approach (SCA) for the same reason. Hence, no model based on the Discrete Crack Approach (DCA) was created and therefore no comparison with such could be made.

1.7 Target users

One group which clearly benefits from the methodology presented in this report are stakeholders in various infrastructure projects, such as bridges. With the development of the methods proposed, the evaluation of the structural damage state of RC structures becomes a more precise and dependable procedure and the maintenance plans will become more effective and thus, likely increase the service life of infrastructures. In addition to this, economic and environmental benefits will most likely arise, presenting a new perspective on how to handle ageing RC structures and offering the opportunity to act proactively rather than reactively. Risks, such as undesirable crack formation, can be detected at an early state before the damage has reached critical proportions.

Another group profiting from this development are the designers of RC structures, who will gain advantages not only due to future utilisation of DOFS itself, which will provide benefits as mentioned above, but also understanding crack behaviour and its applicability in FEA.

It is necessary to mention that even though with the conclusions presented in this thesis, further research and development is needed before any larger scale usage of the methodology will be applicable. Nevertheless, the thesis provides a new steppingstone for the next generation of damage controlling instruments/techniques, which will contribute to a more sustainable and financially viable infrastructure.

1.8 Structure of the thesis

- **Chapter 2**. Finite Element Modeling of Reinforced Concrete Structures

  Gives a general overview of FE-modelling, different material models and crack models that were used in the thesis.

- **Chapter 3**. Structural Health Monitoring

  In this chapter a deeper discussion on the characteristics of Structural Health

Monitoring, an in detail description of how DOFS and DIC operate.

- **Chapter 4**. Crack implementation

  Presents previous work on which the thesis is based and their findings. Further a discussion on what aspects of the two are going to be implemented in the scope of the thesis.

  Moreover, a step-wise description of the automated programs developed in Matlab and how these are transferred into the Diana FE software through generated Python scripts will be explicated. A description of the Diana 10.4 model is presented in terms of geometry, data of the used material models and mesh size. Finally, a description of the software analysis is presented.

- **Chapter 5**. Results and Calibration of Model Using Reference Beam

  Presents results from the FE analysis and comparing these to experimental test results.

- **Chapter 6**. Discussion

  Addresses the findings and discusses major differences, their origin and impact on the validation of the model.

- **Chapter 7**. Conclusions

  General conclusions drawn from the findings in the thesis is presented and recommendations on future improvements are discussed.

2

Finite Element Modelling of Reinforced Concrete Structures

The finite element method is a numerical method used to approximately solve differential equations describing some physical phenomenon. More precisely, FEM is performed by dividing a designated body into finite elements and linking these together using constitutive mathematical relationships. The relationship which acts over the body of elements, whether its nature of being structural or other, is described using Partial Differential Equations (PDE), and in order to solve these systems of equations more efficiently, numerical tools have been developed throughout the years (Harish, 2020).

The body of finite elements is in terms referred to as a mesh. Generally, the smaller the elements are, the better the approximation of the solution gets. This is also one of the drawbacks of FEA since a finer mesh size requires a lot of computational power and time to produce reliable results. Even so, FEA is widely used in most fields of engineering to evaluate and assess structures and their behaviour, both for global models and detailed analyses.

However, there are several factors which affect the outcome of FEA and not only the size of the elements in the mesh. Depending on what material(s) is or are being used and evaluated, several modeling choices have to be made and implemented into the FE model to gain a reliable solution. It is necessary that the structural response by these numerical models adequately reflects the the actual behaviour of the structure and to guarantee it, the FE models must be continuously checked and updated based on the results which are obtained (Harish, 2020) - in other words, the models must continously be refined and improved.

Creating FE models of RC structures in particular are a field which in many cases is generalised and the difficulties of modeling this material will be described in more depth in upcoming sections.

As stated in the introduction, RC structures crack for relatively small loads already in the serviceability state, unless pre-stressed. Micro-cracks appear and due to their nature, induce a non-linear response of the structure because of the non-linear material behaviour which is characteristic for concrete (Plos, 1996). For this reason, linear solution models are of limited value when a realistic representation of the behaviour is sought (even if they are of great value in design). In light of this, re-

liable non-linear relationships to describe RC structures in both serviceability and ultimate state are required to assure an accurate FE model.

2.1 Material models

Material models describe how a specific material behaves under different stress states and the way it plasticizes is described by a yield surface, a hardening rule and a flow rule (Plos, 1996). When the stress-strain relationship of a material behaves linearly it is still in its elastic domain and upon reaching its yield surface, it is considered to be in its plastic domain. Once plasticized, the hardening rule describes the change of the yield surface depending on the level of plastic strain and the flow rule operates the evolution of the plastic strain.

To describe the tensile behaviour of concrete, an exponential-type softening is preferred such as the *Hordijk* relationship shown in figure 2.1 (Hendriks et al., 2016). The main advantage with using Hordijk softening is that it will assure localised cracking which precludes large areas of diffuse cracking. The constitute relation for the theory is that the area under the stress-strain relationship in figure 2.1 should be equal to the tensile fracture energy, G_{F}, divided by the equivalent length, h_{eq}. Upon complete softening - when no stresses are virtually transmitted through the element where cracks have localised - the crack is considered as fully open.

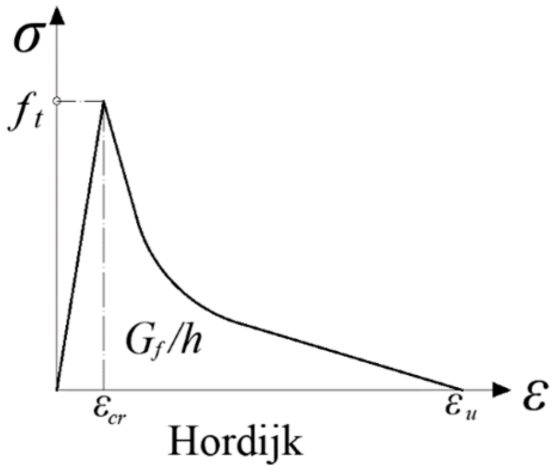


Figure 2.1: *Stress-strain relationship for the Hordijk softening curve (Gombosuren & Maki, 2020).*

Modelling the compressive behaviour of concrete can be quite complicated, in particular the post-peak behaviour. As recommended by Hendriks et al. (2016, p. 19), the compressive behaviour should be modelled such that the maximum compressive stress is limited with a parabolic stress-strain diagram with softening branch as per figure 2.2. The peak compressive strength of the parabolic curve should be

based on the compressive fracture energy, G_{C}, to reduce mesh size sensitivity during compressive strain localization.

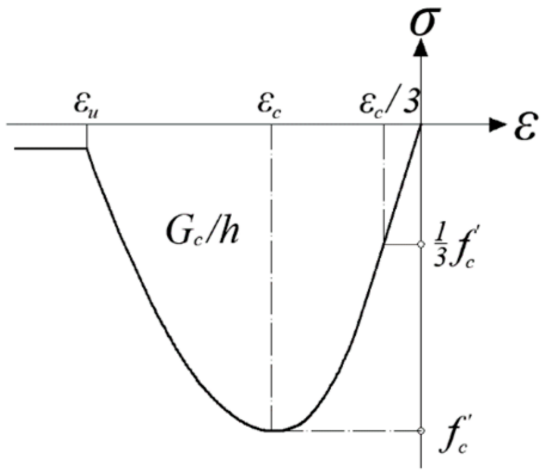


Figure 2.2: *Stress-strain relationship for the Parabolic softening curve (Gombosuren & Maki, 2020).*

2.1.1 Fracture mechanics

Even if an RC structure is not loaded in its Ultimate Limit State (ULS) on a macro-level, weak sections along the beam can develop micro-cracks where stresses reach beyond the material's tensile capacity, f_{ct}. These micro-cracks eventually localise with increasing load in the weakest section of the structure and form fractures, which eventually cause failure of the element (Plos, 1996).

To understand the process of fracture mechanics, the stress-displacement relationship for a homogeneous concrete material is translated by dividing it into two parts - one stress-strain relationship to describe the behaviour outside the fracture zone and a stress-crack opening displacement relation within the fracture zone which describes the deformations which occur in addition to the overall strains (Plos, 1996). This translation is illustrated in figure 2.3.

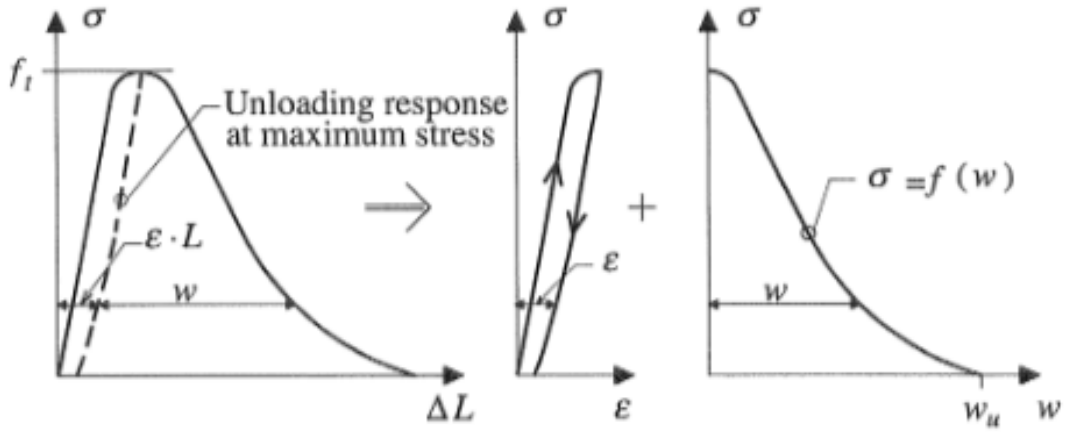


Figure 2.3: *Mean stress-displacement relation for a uniaxial tensile test of a homogeneous concrete specimen split into a general stress-strain relation and a stress-displacement relation in the fracture zone (Plos, 1996)*

Takeaways that can be made from this and implemented into the FE model is the *tensile fracture energy* of a material, described by Zandi (2020) as the *energy needed to completely break a unit area of material* and can be calculated as the area under the second part of the stress-displacement relation shown in figure 2.3. This is illustrated more clearly in figure 2.4. This parameter, along with compressive fracture energy, can further be numerically estimated in compliance with the guidelines provided by Hendriks et al. (2016) as:

$$G_F = 73 \cdot f_{cm}^{0.18}, \quad G_C = 250 \cdot G_F \tag{2.1}$$

where f_{cm} is the mean compressive strength of the concrete material.

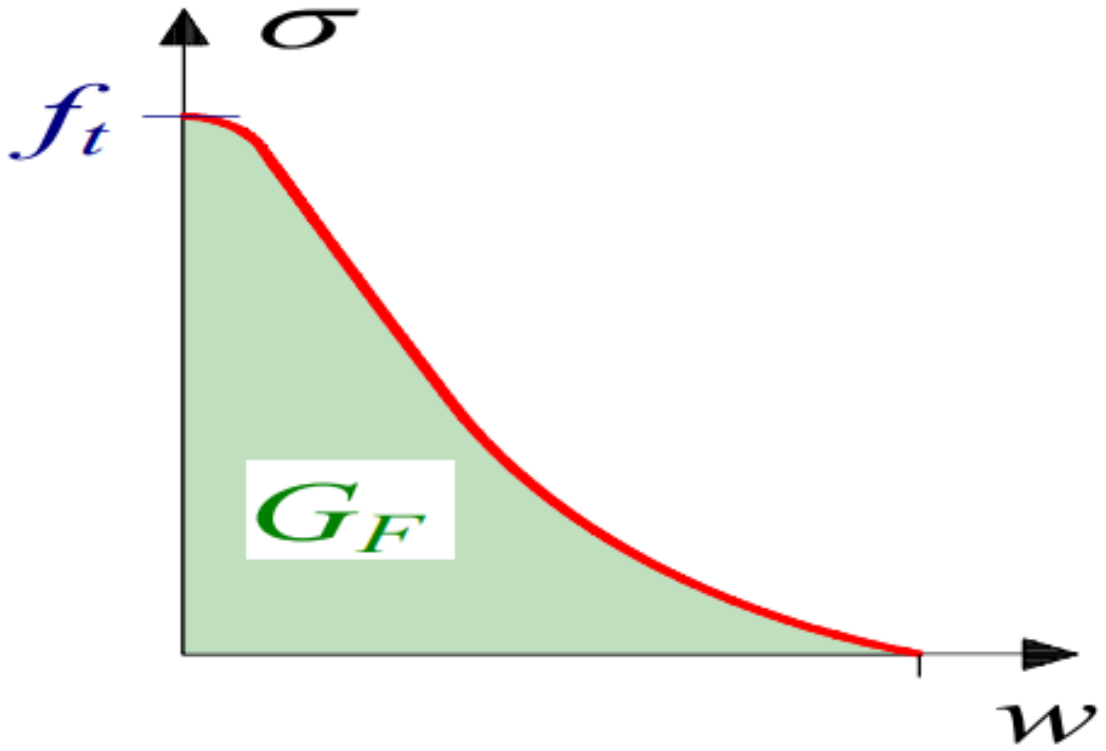


Figure 2.4: *Estimation of the tensile fracture energy, G_F, as the area under the graph of the stress-crack with opening relation (Zandi, 2020)*

The fracture energy can then be implemented as a material parameter in Diana, which enables the FE software to describe the crack opening behaviour in localised regions, using the SCA (Broo et al., 2008). Note that this approach does not model specific cracks but rather cracked regions. Additionally, the method requires parameters such as the concrete tensile strength along with the crack bandwidth, h. The latter is the length over which the deformation of the crack is smeared where the correlating cracking strain, ε_1, can be described using as the crack opening, w, divided by the crack band width in equation (2.2).

$$\varepsilon_1 \approx \frac{w}{h} \tag{2.2}$$

In order to describe specific cracks, the DCA, can be used. This, however, requires all the information about the cracks (position, magnitude etc.) since they have to be modelled beforehand using interfaces in the concrete element with weaker material properties.

2.2 Weakened element approach

An alternative way of implementing cracks is to adapt a method using the WEA, suggested by Blomfors (2021) where finite elements coinciding with a known, pre-existing crack are assigned weakened element properties in terms of decreased tensile strength and ultimate strain. This method is based on the smeared crack model as an already known crack is smeared over elements in the mesh to enable localisation. The weakening of elements is determined using a bi-linear, mode-I, stress-to-crack width relationship shown in figure 2.4, with the parameters as:

$$\sigma_{\mathrm{s}} = 0.25 \cdot f_{\mathrm{ctm}}; \quad w_{\mathrm{ult}} = \frac{5 \cdot G_{\mathrm{F}}}{f_{\mathrm{ctm}}}; \quad w_{\mathrm{s}} = 0.15 \cdot w_{\mathrm{ult}} \tag{2.3}$$

where f_{ctm} is the mean tensile strength of the concrete material. The curve can also be referred to as a *tension-softening curve* and can, depending on how exact of an estimation you want to make, be modelled as multi-linear and non-linear as well (Broo et al., 2008).

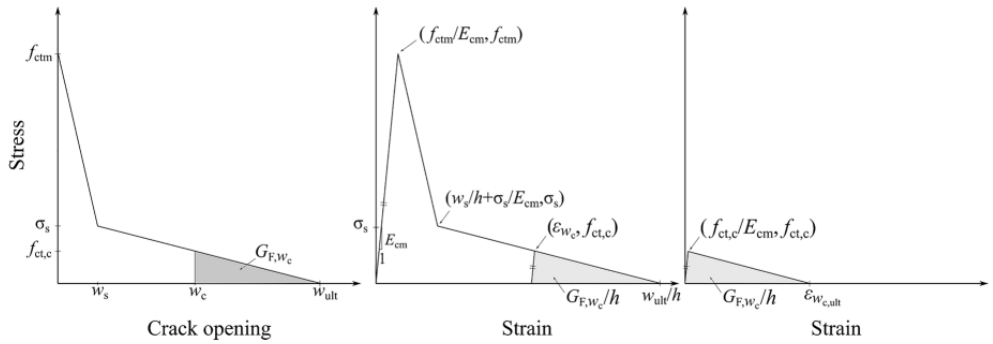


Figure 2.5: *Stress-to-crack opening relationship which is used for calculating the residual tensile strength (Blomfors, 2021)*

Using this bi-linear relationship, the tensile strength and remaining crack opening until stress-free cracking were derived based on the measured crack width, w, at a specific point and can be summarized as per Blomfors (2021):

$$f_{\mathrm{t}} = \begin{cases} f_{\mathrm{ctm}} - \frac{f_{\mathrm{ctm}} - \sigma_{\mathrm{s}}}{w_{\mathrm{s}}} \cdot w, & \text{for } w \leq w_{\mathrm{s}} \\ \sigma_{\mathrm{s}} - \frac{\sigma_{\mathrm{s}}}{w_{\mathrm{ult}} - w_{\mathrm{s}}} \cdot (w - w_{\mathrm{s}}), & \text{for } w_{\mathrm{s}} < w \leq w_{\mathrm{ult}} \\ 0, & \text{otherwise} \end{cases}$$

The ultimate strain, $\varepsilon_{w_{c,ult}}$, can further be described by setting the area under the stress-strain relationship equal to the residual fracture energy, G_{F,w_c}, divided by the crack bandwidth, h, which corresponds to:

$$\varepsilon_{\mathrm{w_{c,ult}}} = \frac{w_{\mathrm{ult}} - w}{h} \tag{2.4}$$

2.3 Bond-slip

Bond-slip refers to the behaviour of the interaction between the reinforcement bar and the surrounding concrete which it is embedded in. The stresses transferred between the two causes deformations in the concrete closest to the rebar and for deformed bars, the stresses are transferred mainly through its ribs (Plos, 1996).

According to Hendriks et al. (2016), excluding a bond-slip model for the reinforcement when using FEA is acceptable, but implementing it in the model will result in more accurate results. A commonly used model for bond-slip is one proposed by *fib Model Code for Concrete structures 2010, chapter 6*, and uses the bond-slip curve in figure 2.6, (FIB, 2013).

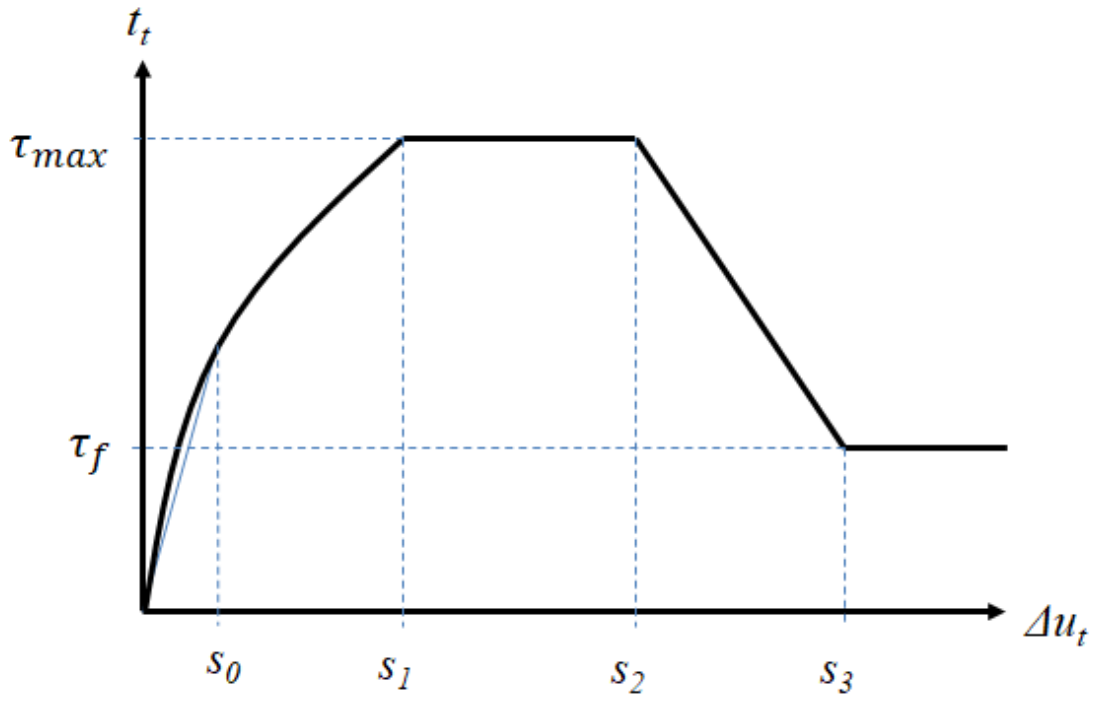


Figure 2.6: *Bond-slip back-bone curve of fib Model Code 2010 (DIANA FEA, 2020). The curve shown in the figure is for pull-out failure*

This bond-slip curve consists of four different sections; the first describes an exponential behaviour of the shear-stress growth, τ, from 0 to τ_{max} at a relative displacement of s_1. In the second section, the bond-slip stress remains constant at τ_{max} until the relative displacement s_2 has been reached. Upon reaching this displacement, the bond-stress decreases linearly until reaching the relative displacement s_3 where it is finally kept constant at τ_f (FIB, 2013).

3
Structural Health Monitoring

Before any deeper explanations on this subject are conducted, it should be noted that the thesis' focus is RC structures. Therefore, when any SHM systems or technologies are discussed in this chapter, it refers specifically to these structures. This, even though SHM is a subject that spans widely and can be applied throughout society in many different functions such as pipeline monitoring, fire security and communication networks (Ping et al., 2019). An industry where SHM is highly developed is in the aerospace industry, where SHM systems are utilized for aircraft monitoring (Sohn & Liu, 2016).

As mentioned above, it is also possible to note increasing demands on RC members, in particular bridge infrastructure and buildings, due to increasing vehicle loads, climate change factors etc., which have intensified the requirements on the ability to monitor the ongoing damage, in order to fully ensure their SLS behaviour and technical life span (Blomfors et al., 2020), (Berrocal, Fernandez, Bado et al., 2021), (Vardanega et al., 2015). The monitoring to ensure the serviceability and security of infrastructure due to the increased demands of performance, aim to prolong the service life of structures and make them more cost-efficient. Moreover, they serve as a steppingstone towards a more sustainable construction sector (Berrocal, Fernandez, Bado et al., 2021) since the knowledge of increasing infrastructure stock goes hand in hand with the maintenance, investment cost and emissions of these structures, all of which need to be limited.

SHM can be described, in a nutshell, as sensoring systems with one or several primary objectives, that collect and process data from structures, presenting the end-user with information about the structural condition of a structure. The output from the systems can be deflections, corrosion probability, crack propagation, strain measurements and chloride ion concentration (Vardanega et al., 2015). The output can then be used as a basis for action plans for maintenance and repairs of a structure.

3.1 Why is structural health monitoring needed?

Some questions arise from the aforementioned; why and how are the standards of monitoring today insufficient and what does this imply?

There are no clear answers to these questions, but a major deficiency is that current techniques, which detect structural damage mainly rely on visual and manual measurements, i.e. inspections which are not classed as SHM. These are not only time-consuming but also give far less reliable data and contain a lot of assumptions compared to more refined methods, based on impartial measurement techniques. This in return makes the data not only harder to implement in FE based analysis, but also highly ambiguous in terms of results. Not only does this make the assessment of large-scale structures impractical, it also makes updated models based on real time data impossible.

Due to the lack of well-developed techniques to monitor structural health, SLS damage assessment of RC members is an area where measurement are inadequate and need refinement. The difficulties are manifold, spanning from the sheer amount of data points needed to interpret potential risk zones to making the collected data sensible in an intuitive manner (Vardanega et al., 2015), (Blomfors et al., 2020),(Blomfors, 2021).

3.2 Categorisation and application

Current standards and codes, for example EN-1992-1-1 (2004), are up to date in some sense (considering e.g. the ultimate capacity state) but are not fully developed with regard to, for instance crack development (Blomfors, 2021). SHM has therefore become a promising technological feature to implement, in order to provide quantifiable data concerning the structural health of RC elements (Vardanega et al., 2015). There are a variety of different monitoring systems under development, but not many real case studies confirming their capabilities and as Vardanega et al. (2015, p.759), mention: What is often lacking is a clear statement of what value the system hopes to deliver.

This statement indicates that even if SHM systems are deployed and used today, the output needs to be much more intuitive for stakeholders to make correct decisions based on the data provided. It also highlights a key issue of trust between stakeholders and the manufacturers, since systems that do not provide sufficiently intuitive or understandable results are less likely to gather investments.

There is also the question of which type of SHM system that is actually required. The cost of implementing such systems is relatively high so there needs to be an understanding of what level the system should operate and what outputs are indispensable. Webb et al. (2014), categorises SHM systems in five different subcategories, based on their main purpose of monitoring, see table 3.1.

Table 3.1: *Structural Health Monitoring system objectives (Webb et al., 2014).*

Category	Category name	Description
1	Sensor deployment studies	Demonstrate new sensor or communication technologies.
2	Anomaly detection	Detect changes, either over time or direct.
3	Model validation	Comparison of a structure performance to performance given by structural analysis models.
4	Threshold check	Comparison of key parameters against thresholds to warn of potential problems.
5	Damage detection	Detection and location indicator of damage through structural identification and modal analysis.

It is emphasised that a system is not bound to one category but can be classified as a combination of any of these subcategories and therefore, one system can provide multiple checks and validations when analysing its output.

3.3 Distributed optical fibre sensors and digital image correlation

As aforementioned, there are a several SHM technologies available, however, this thesis focuses on DOFS, since the experimental results are given from this type of system. The reader is refereed to Vardanega et al. (2015) for an overview of other sensor systems.

3.3.1 DOFS - Characteristics

Fibre optical sensors have for a long time have been considered a promising prospect in the use for SHM. An optical fibre's properties are summarized by Adachi (2008, p.329) as: "..., corrosion-resistant, maintenance free, light weight, and having excellent durability".

Even though this is true, prior drawbacks to developed sensor technologies (e.g. fibre Bragg grating (FBG) and Fabry-Perot) such as the limited amount of data points that can be measured, have made them rather limited, depending on which application is considered. This might not be of great concern when studying small systems where any critical sections can be sufficiently covered. But considering larger and more complex structural systems where critical sections are harder to fully locate beforehand, this becomes a concern. The errors in the system could possibly have devastating outcomes since it could overlook structural damages. Another drawback is that a fibre, despite its excellent characteristics when installed, can be rather fragile during handling (Berrocal, Fernandez, Bado et al., 2021).

Nevertheless, improvements of fibre optics have produced DOFS and made it possible to utilize the fibre in its entirety, providing an unmatched spatial resolution which undermines the argument of uncertainty in terms of measurement points (Ping et al., 2019). DOFS measurements work on the basis of optical scattering and there are three types that are more prominent, Raman, Rayleigh and Brillouin (Ping et al., 2019); due to their ability to capture temperature and strain variations, the two latter are commonly employed in the construction sector (Berrocal, Fernandez, Bado et al., 2021).

3.3.1.1 Brillouin and Rayleigh scattering

Investigating Brillouin scattering more closely, this can be described as a backscattering process due to the interaction of light traveling within a medium, such as an optical fibre, and the medium itself leading to an energy exchange. This energy exchange depends on variations in the density and refractive-index, and occurs as the incident photons interact with e.g. phonons (acoustic waves), hence it is described as an inelastic scattering process. Furthermore, this implies that the scattered light's frequency will shift relative to the incident light frequency (Xin et al., 2019), see figure 3.1.

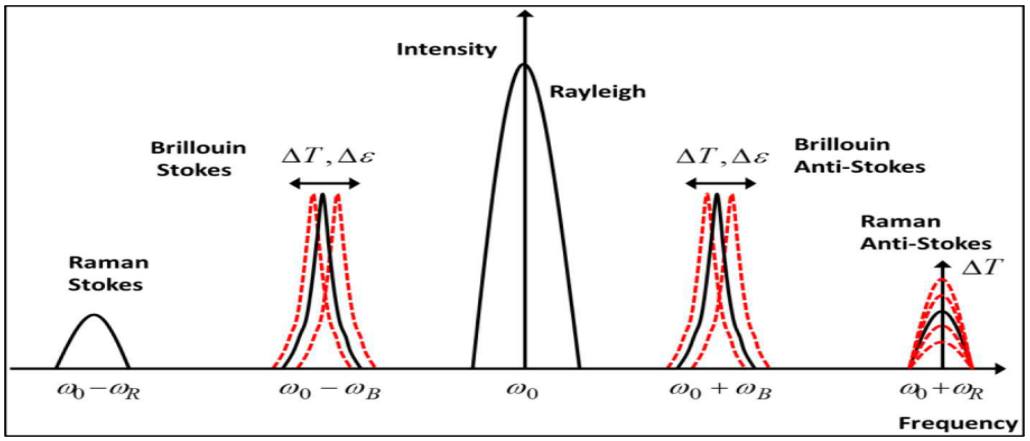


Figure 3.1: *Overview of the condensed down back-scattering processes employed in DOFS. The input frequency is denominated ω_0, and as can be seen the Brilloin scattering shows a clear shift in frequency, with non-linear characteristics. The Rayleigh scattering shows its elastic properties since no energy transfer has occurred, and so is linear (Muanenda et al., 2019).*

Different to the Brillouin scattering, Rayleigh scattering is what is called an elastic process - signified by the fact that there is no energy exchange between the incoming light and the fibre optic medium, thereby there is no relative frequency shift, as seen in figure 3.1.

Berrocal, Fernandez, Bado et al. (2021, p.2) discuss the limitations and advantages of different processes, according to their relevance to measure strain. Comparing Rayleigh and Brillouin scattering DOFS, the latter are capable of measurements up to 300 km whereas Rayleigh based DOFS' (Optical Frequency Domain Reflectometry (OFDR)) range lies up to 70 m. However, this limitation is compensated with a spatial resolution of 0.65 mm which, compared to the spatial resolution of Brillouin scattering based DOFS, that lies in the range of about 1 m, clearly demonstrates its applicability in detecting strain variations and so crack detection in a RC element (Barrias et al., 2019). Forth, the actual strain measurement procedure for Rayleigh OFDR will be discussed.

When subjected to load and/or temperature variations, strain variations will arise in an RC member. If then a laser light at a certain frequency is emitted through the fibre optic, providing a baseline of the backscatter, the relative frequency shift of the backscatter can be measured along the cable, hence detect local fluctuations (Barrias et al., 2019), see figure 3.2. These can then be transformed as mechanical strain through equation 3.1, (Drake et al., 2018):

$$\frac{\Delta\gamma}{\gamma} = K_{\mathrm{T}} \cdot \Delta T + K_{\varepsilon} \cdot \varepsilon \tag{3.1}$$

where $\Delta\gamma$ represents the change in wavelength relative to the reference wavelength, γ, ε is the strain along the fibre and $\Delta\ T$ is the change in temperature. K_T and

K_ε are known coefficients for the temperature and strain, respectively (Drake et al., 2018). However, in an environment where the temperature is known to be stable during testing, the temperature has little or no effect and can therefore be neglected in terms of strain measurements (Berrocal, Fernandez, Bado et al., 2021).

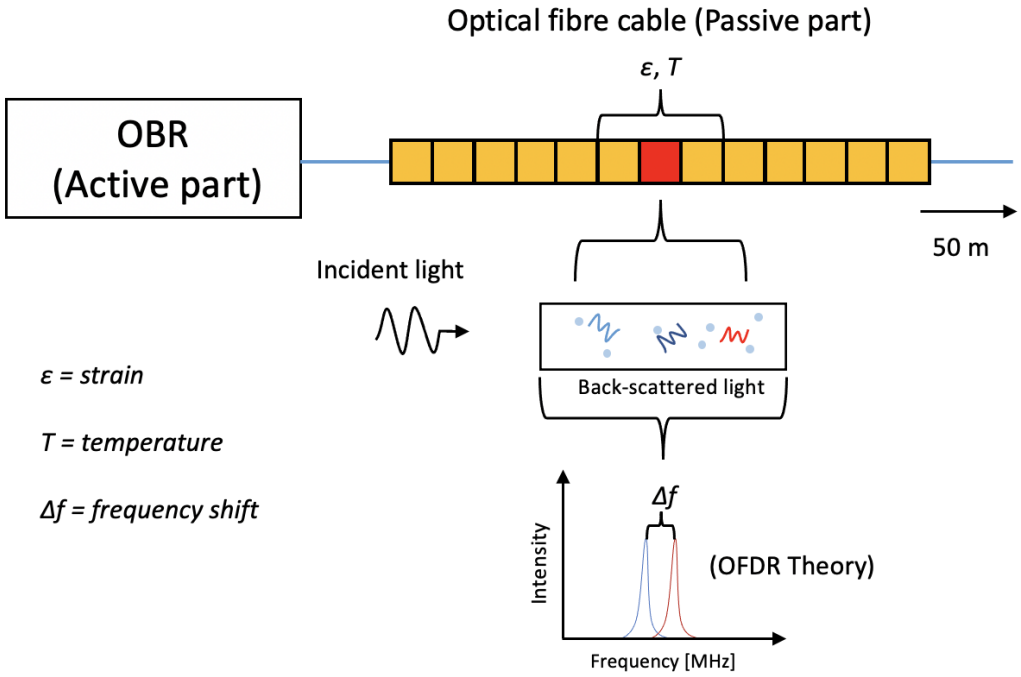


Figure 3.2: *Schematic view of how the Rayleigh scattering-based DOFS operate. As can be seen, it is the relative shift of the backscattering that detects strain and temperature fluctuations. After idea by (Barrias et al., 2019).*

3.3.1.2 Installation techniques and improvement areas

There are several installation techniques for SHM systems based on DOFS in RC structures. The choice of installation technique is mainly influenced by the structure, the desired output and at which specific time in the casting process the installation is carried out (Berrocal, Fernandez, Bado et al., 2021). Frequently used methods are:

- Attaching the fibre on surface of concrete (when hardened)
- Embedding the fibre in the concrete when casting
- Attaching the fibre to the reinforcement in prior to casting

where the two latter are mainly used in new structures. One could, however, also use them in e.g. rehabilitation work of a structure, where the concrete cover is removed and later recasted or directly on the surface of an old structure, like the Göta älv-bridge in Gothenburg, see Herlin (2011) for reference.

Even though an extensive number of experiments and research have been conducted using DOFS and the results have confirmed their capabilities, there are some issues

that need further development. For example, the adhesive used to bond the fibre to either the surface of the concrete or the reinforcement and also the type of coating of the fibre have a large influence on the measured/predicted crack width. This is because the coating has lower stiffness than the encased fibre, which makes the strain transfer between the RC member and the optical fibre localise in a smeared manner rather than in the desired, confined position. Work has been done to handle this phenomenon, e.g. a mechanical model based on shear lag theory, but further development is needed to account for all bonding situations (Berrocal, Fernandez & Rempling, 2021a).

Another example is so called Strain Reading Anomalies (SRA) that have been shown to manifest in surface- and reinforcement bonded DOFS, when e.g. too delicate fibre configurations are installed, which could lead to either under- or overestimation of strain effect when analysed (Berrocal, Fernandez, Bado et al., 2021).

Regardless, the technology overall has great potential and the development points towards better models when applying it to RC members, not least the newly published article by Brault and Hoult (2019).

3.4 Ethical dilemmas when improving structural health monitoring

Thus far, this section of the chapter has presented the outline of what defines SHM. In this section, possible ethical downsides of improved and wider utilisation of SHM will be discussed.

The technical benefits of using SHM are substantial - the precise measurements and machine learning algorithms which after improvement over time, will eventually minimize human interaction, perhaps both in monitoring and in decision-making. The sustainability gains and the pros of being able to act proactively rather than reactively has also previously been discussed, and made the beneficial aspects tangible.

Even though the pros are clear, the transition might not be as smooth as it would seem. As discussed in section 3.2, the research field of SHM, runs the risk getting too abstract and yield too little gains to the stakeholders, which could develop into further trust issues towards this new technology. This distrust could, in turn, lead lasting use of old, manual techniques to a larger extent than desired. Hence, the transparency of the research and applicability of the products developed are vital, both to engage in understanding throughout the industry and to raise demand for the technology.

The implementation will, as mentioned, overtake manual labour and an expertise will become obsolete. This will also have social consequences (unemployment, lower income, educational differences) but merely during a transition period. This being said, both the industry and the world at large, are approaching a more automated

reality which society must adapt to, at all levels, and if companies/schools are vigilant and willing to educate, the transaction will be smoother than if corporate profits must come first.

4

Crack Implementation

This chapter will present previous research conducted with the objective of enabling crack inclusion in FEM. Moreover, the process of developing the Matlab and Python scripts to automate the translation and implementation of raw SHM data into cracks into the FE mesh in Diana will be thoroughly presented. All scripts have been based on the beam configuration shown in figure 4.1, since DOFS data was available for that specific beam and it could therefore be used for creating the Matlab-script which translates the strain data into cracks.

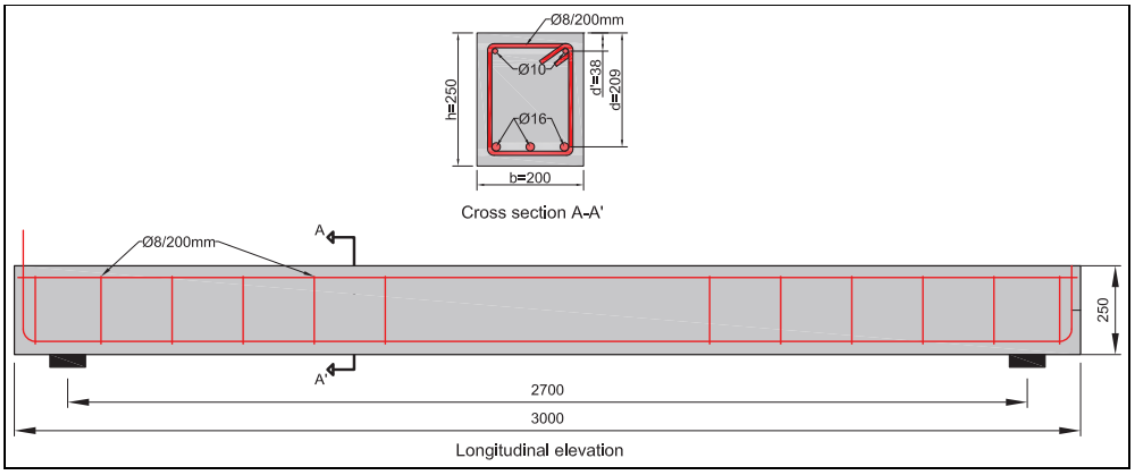


Figure 4.1: *Geometry and reinforcement layout for the beam used for debugging the scripts. All measurements given in mm. Source: Berrocal, Fernandez, Bado et al. (2021)*

4.1 Previous research in the field of automating crack inclusion in FEM

As mentioned in chapter 1, the thesis has been mainly based on previous research published by Berrocal, Fernandez, Bado et al. (2021) and Blomfors (2021). The fields which the two concern are highly related, but the methods used to collect and implement crack data differ.

The approach by Blomfors (2021), was to through a visual inspection, detect and categorize cracks throughout the duration of the experiments (i.e. restrained shrinkage, tensile loading, and bending cracks) and record their features closely. Thus,

this data, based on the coordinates and widths of the cracks, could be transformed into material and spatial characteristics, that were capable of being implemented into an FE model to improve the performance of the analysis. The inclusion of cracks was done both with the WEA, introducing weakened elements according to the metric measurements of the crack and their widths, and the DCA, introducing crack planes in the model with interface elements that consider the residual tensile strength. In his PhD thesis, Dr. Blomfors concluded that the WEA resulted in less errors and anomalies, hence this was presented as the most promising solution of the two (Blomfors, 2021).

However, there are points brought up in the thesis further improving the method, like incorporating sensors such as fibre optics, to measure cracks more precisely. The context to this was the limitations of the visual inspection, where only surface cracks are detectable and hence, measurable, unlike measurement sensors that can be installed within a structure.

Both the method and output of the report by Berrocal, Fernandez, Bado et al. (2021) are different. Here, the measurements from the experiments were done by utilizing DOFS, thus implementing a more sophisticated measurement technique as Blomfors (2021) invoked.

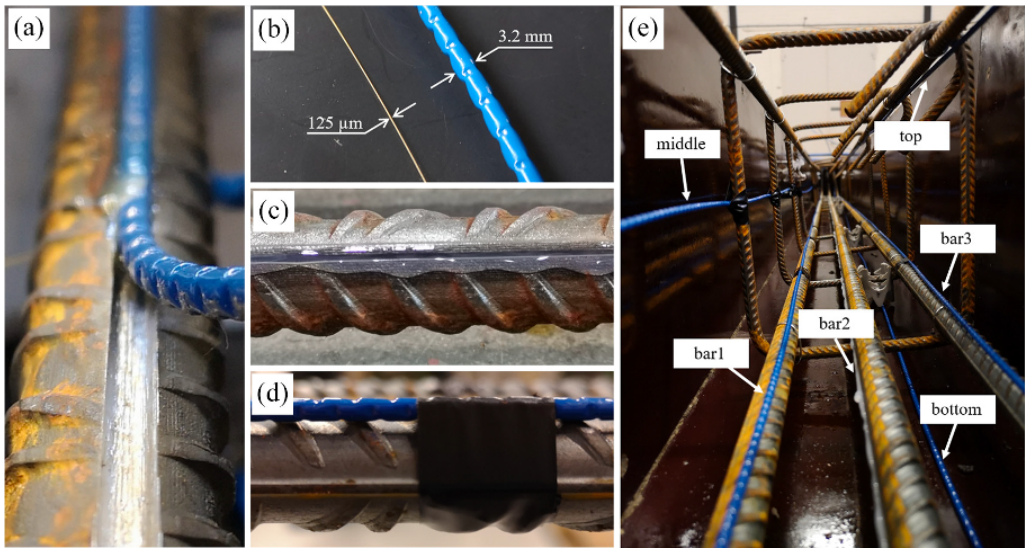


Figure 4.2: *Installation of the optical fibre sensors: (a) installation of robust DOFS cable in a reinforcement bar by inserting it into a previously milled groove; (b) comparison of thin and robust DOFS; (c) installation of thin DOFS on the surface of a reinforcement bar by bonding it with cyanoacrylate adhesive and protecting it with silicone; (d) installation of robust DOFS on the surface of a reinforcement bar by mechanically anchoring the cable to the reinforcement with electric tape; and (e) multi-layer configuration of embedded DOFS in the beam specimens, (Berrocal, Fernandez, Bado et al., 2021)*

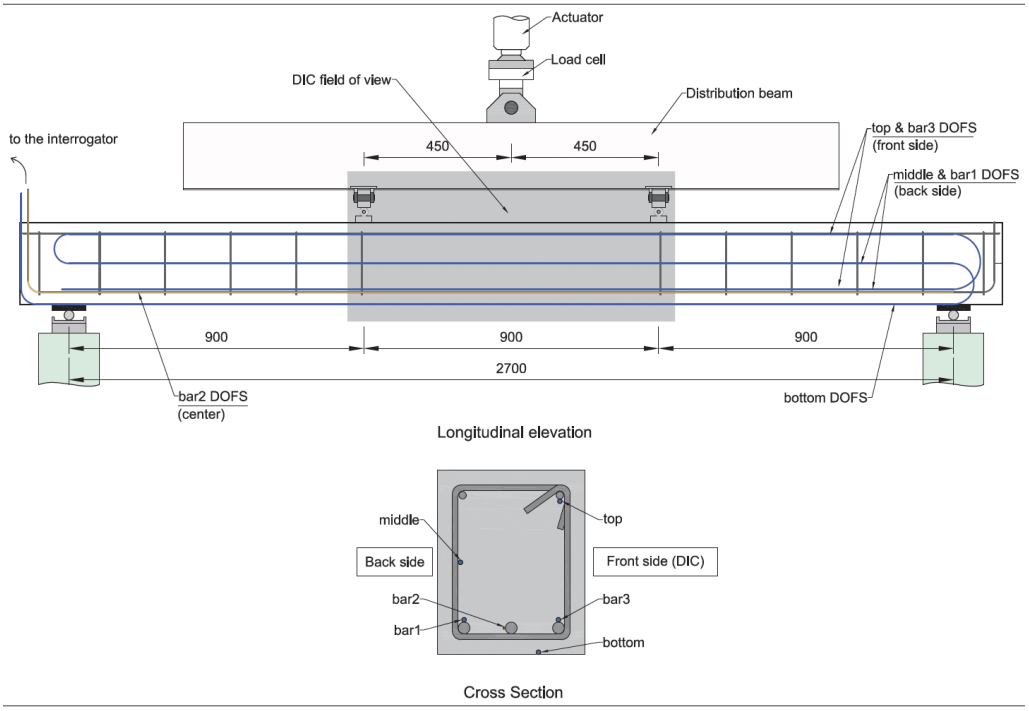


Figure 4.3: *Loading setup and DOFS installation configuration for the RC beam specimens (Berrocal, Fernandez, Bado et al., 2021).*

By installing Rayleigh-based scattering DOFS (described in section 3.3.1.1) embedded in the concrete and bounded to the reinforcement, as seen in figure 4.2, the strain could be measured both throughout the length of the fibres and at different levels of the cross-section of the beam. Figure 4.3 illustrates the loading setup for the testing, i.e. a four-point bending test setup. When testing this method, Berrocal, Fernandez, Bado et al. (2021) could conclude that by post-processing the data, both the crack widths and the positions of the cracks could be accurately predicted, even though some cracks were shown to go undetected. The ratio of undetected cracks was relatively small though, which lead the authors to deem the method very useful. To calculate the crack widths, equation 4.1 was utilized (see (Berrocal, Fernandez & Rempling, 2021b) for full explanation). Basically, the equation considers both the actual strains measured, assumes a linear behaviour of the strain between the cracks and also takes into account transferred stress due to bond between the concrete and reinforcement.

$$w_{cr,i} = \int_{-l_{l,i}^{-}}^{l_{l,i}^{+}} \varepsilon^{DOFS}(x)dx \; - \rho\alpha \left[\int_{-l_{l,i}^{-}}^{l_{l,i}^{+}} \hat{\varepsilon}(x) - \varepsilon^{DOFS}(x)dx \right] \tag{4.1}$$

where

α	Modular ratio: $E_\mathrm{s}/E_\mathrm{c}$
$\varepsilon^{\mathrm{DOFS}}$	Strain measured with DOFS
$\hat{\varepsilon}$	Strain, varying linearly between cracks
$l_{\mathrm{l,i}}^{-}$	Left end of transmission length where slip occurs (i:th crack)
$l_{\mathrm{l,i}}^{+}$	Right end of transmission length where slip occurs (i:th crack)
ρ	Reinforcement ratio: $\frac{A_\mathrm{s}}{b \cdot d}$
$w_{\mathrm{cr,i}}$	Crack width of the i:th crack

However, the report was not aimed towards the possibility of explicitly carrying out any further FE analysis, in other words not to enhance the assessment toolbox, but to analyse whether DOFS could be a useful tool in crack and deflection detection. Furthermore, the post-processing provided easily comprehensible plots that could become of use for stakeholders.

Nevertheless, data collected through DOFS proved to be accurate in the report and showed great potential for coupling the two works. This would minimize time consumption of visual inspections along with providing more accurate data, updated in real time.

4.1.1 Collection and post-processing of DOFS data

In the following two sections, a more in-depth analysis of which aspects of the previous works were of great interest to further develop the thesis is presented, starting with DOFS data collection and how to post-process this into a form usable in the application of FEA.

Berrocal, Fernandez, Bado et al. (2021), present the measurement- and detection strategy for bending cracks as the analysis of the strain profiles of robust DOFS. Compared to thin fibres, the robust DOFS employed were shown to yield more converged and smooth strain profiles with less noise (SRA). The crack locations were then confirmed by comparing the strain profiles from the DOFS to the DIC results, and it could be concluded that strain peaks corresponded to crack locations.

To calculate the crack widths, equation 4.1 was utilized, but to visualize the crack pattern, a post-processing of the strain profiles was needed due to the limited amount of data points measured along the height of the specimen beams. Two additional strain profiles were created in order to further improve the resolution of the cracks (see figure 4.4). The intermediate profile was created by taking the mean value between two known points, whereas the bottom profile was calculated by extrapolating the strain under the Euler-Bernoulli theory assumption that plane sections

remain plane. The latter can be represented as:

$$\epsilon_{bot}^{fict}(x) = \epsilon_{bar1}(x) - \frac{\epsilon_{mid}(x) - \epsilon_{bar1}(x)}{y_{mid} - y_{bar1}} y_{bar1} \tag{4.2}$$

where $\epsilon_{bar1}(x)$ and $\epsilon_{mid}(x)$ represent the measured strains and y_{mid} and y_{bar1} the DOFS positions in the beam.

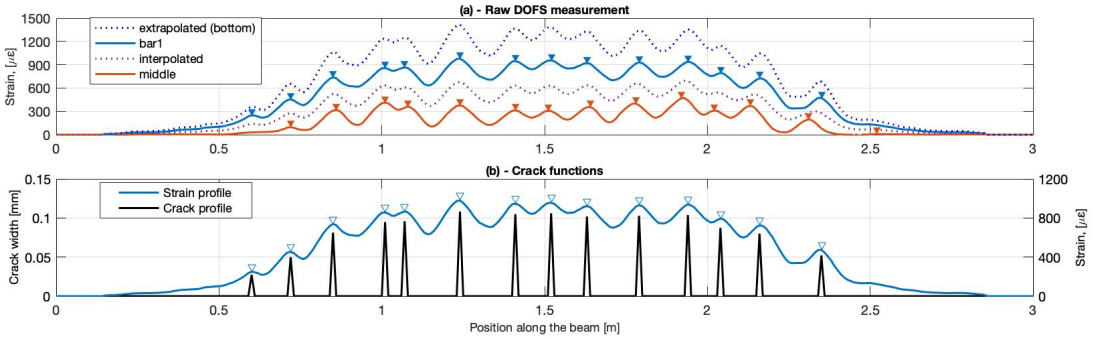


Figure 4.4: *(a) Graph of the known and fictitious strain profiles. (b) Graph of the crack functions containing location and widths(Berrocal, Fernandez, Bado et al., 2021).*

Coupling all the strain profiles with equation 4.1 and the gained knowledge of the positions of the cracks as aforementioned, crack functions were able to be formed using a Gaussian curve function, see figure 4.4. As a last step, a mesh grid was created onto which these crack functions containing all the data from of the cracks (location and widths), were allocated. Furthermore, by interpolating the crack functions between the nodes of the mesh, contour plots could then be generated as illustrated in figure 4.5.

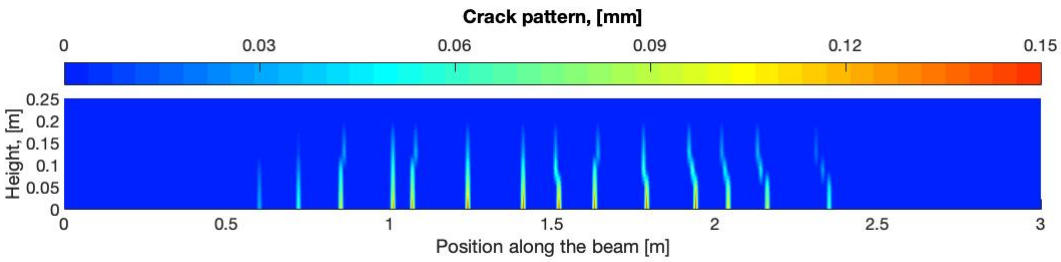


Figure 4.5: *Contour plot after post-processing displaying crack pattern of one of the experiment specimens (Berrocal, Fernandez, Bado et al., 2021).*

4.1.2 Pre-existing crack inclusion, weakened elements and FE-modeling

Blomfors (2021) had, as mentioned previously, an analog measuring technique where all cracks were measured by visual inspection. The aim of the research was not to

show the ability to perfectly monitor crack propagation, but to visualize the importance of the inclusion of pre-existing cracks in FE-modeling and analysis (Blomfors, 2021) in ULS. Blomfors (2021) had two main approaches in terms of modelling as discussed, WEA and DCA.

As described in section 2.2, the basic idea of the WEA is to localize and weaken the tensile properties of elements in an FE mesh in relation to the measured cracks width, and is based on the SCA. Since the position and magnitude of pre-existing cracks were known and the goal was to localize the cracks in the weakened elements (i.e. smear the cracks over the weakened elements), the crack bandwidth, h, can be set to the element length, see figure 4.6. This was done since the SCA supposes that a crack localises over an element in the FE mesh.

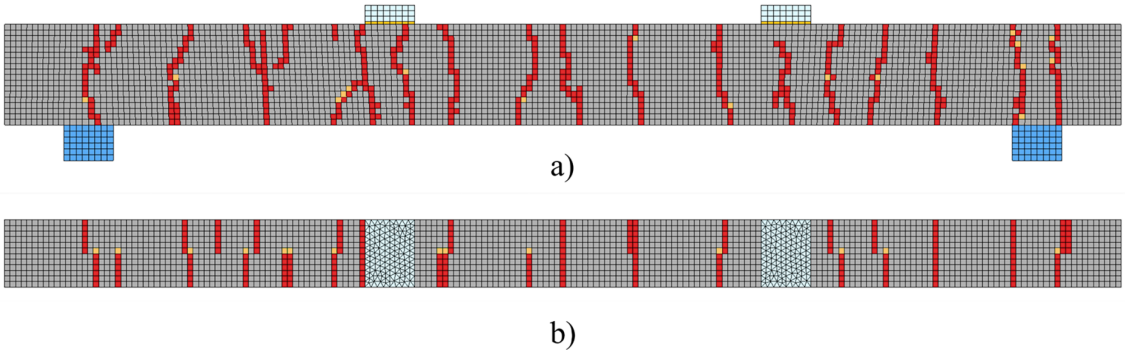


Figure 4.6: *Mesh of FEM beam with weakened elements in Blomfors (2021).*

DCA is fundamentally different as described in section 2.1.1. There is no weakening of tensile properties in elements but the pre-existing cracks are defined in FE model by creating discrete, interface elements with weakened properties, defining the average crack width of the pre-existing crack, see figure 4.7.

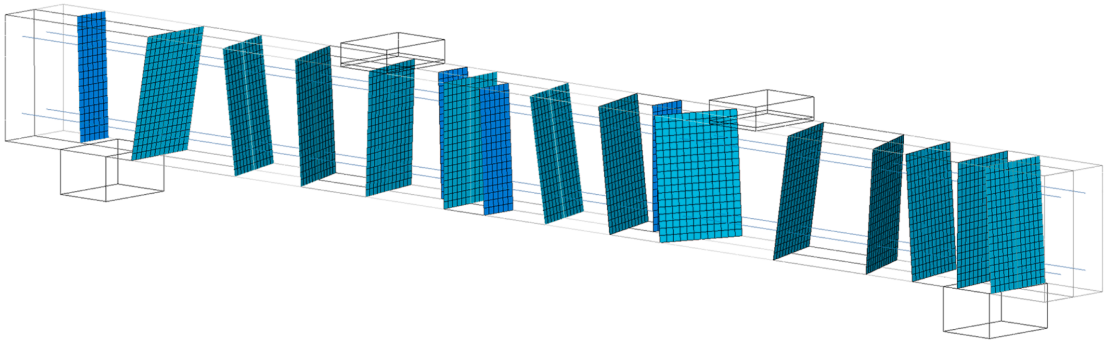


Figure 4.7: *3D view of discrete crack elements in Blomfors (2021).*

What Dr. Blomfors concluded in his dissertation was that the WEA was more straightforward and demanded less computational time. The results were also shown to have better correlation to the experimental results in terms of ultimate capacity, but the measured crack widths were not explicitly validated between experimental values and FE results. A confirmation on how well the method performs in SLS is therefore missing and is not validated when considering cracks (Blomfors, 2021).

The main implementations from the report and dissertation are therefore the coupling of DOFS data into FE modelling and analysis and the validation of the crack widths between the pre-existing cracks from measurements and the FEA performed according to the WEA.

4.2 Strain to crack translation

The information from the DOFS were extracted from a reference beam in terms of strains over the length of each individual fibre, measured in a pre-cracking test up to 30 kN at an age of 14 days after casting of the beam. See chapter 5 for further presentation of the reference beam.

To translate the strain data from the DOFS into cracks, a program originally created by Berrocal, Fernandez, Bado et al. (2021) was utilized. The Matlab script produced by the aforementioned creates a strain profile over the height of the beam by inter- and extrapolating in between and outside of the DOFS layout previously shown in figure 4.3 and then converts them into crack widths using equation 4.1. These values were smeared and more widely interpolated over a mesh grid made in Matlab which gave the appearance shown in figure 4.5, where the cracks' prevalences are shown in the form of a contour plot.

However, the script written by Berrocal, Fernandez, Bado et al. (2021) was made with the objective of showing the formation of cracks in an educational manner by the previously mentioned smearing of crack widths. It therefore had to be modified accordingly with regard to the objective of this thesis. The cracks were, instead of being smeared, extracted in terms of exact nodal positions along with their respective crack width and allocated into two separate matrices using the aforementioned mesh grid. A note to be made is that the size of each element in the mesh grid created in Matlab has to coincide with the mesh size to be used later in DIANA when implementing the crack information. To illustrate the extraction of nodal positions and crack widths, the scatter-plot shown in figure 4.8 was created.

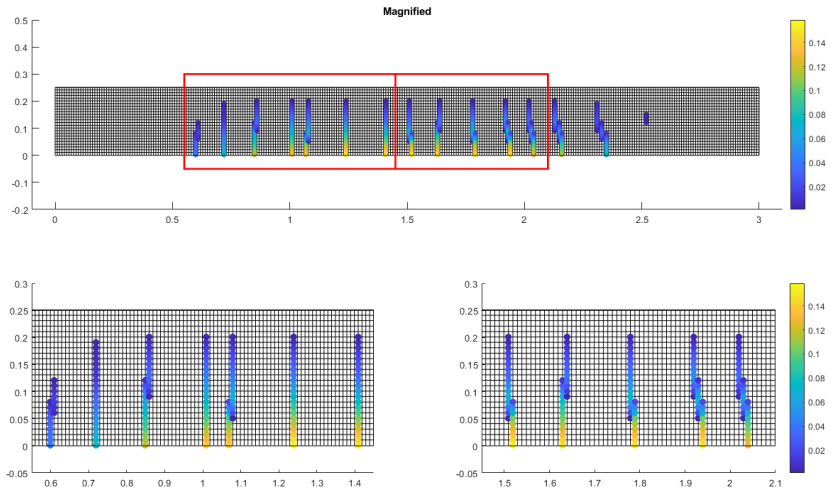


Figure 4.8: *Top: Scatter plot of the cracks' nodal positions and their magnitudes described by the colour bar. Bottom left: Magnification of cracks to the left of the middle of the beam. Bottom right: Magnification of the cracks to the right of the middle of the beam.*

One problem that arose, which might not be as evident judging by figure 4.8, was that the DOFS lacked in sensitivity for smaller mesh sizes which generated voids in the crack interpolation for inclined cracks at certain locations. This is more evident if looking at figure 4.10. This was solved by introducing the for-loop shown in figure 4.9.

```
%% Find voids in inclined cracks and interpolate
k = 0;
for i = 1:max(row)
    for j = 1:max(col)
        if fcrack(i,j) > 0
            if (fcrack(i,j+1) == 0 && fcrack(i,j+2) > 0)
                k = k + 1;
                voidCrack(k, 1:2) = [i, j+1];
                if ( fcrack(i,j+2) > fcrack(i,j) )
                    fcrack(i,j+1) = fcrack(i,j)+(fcrack(i,j+2)-fcrack(i,j))/2;
                else
                    fcrack(i,j+1) = fcrack(i,j+2) + (fcrack(i,j) - fcrack(i,j+2)/2);
                end
            elseif (fcrack(i,j-1) == 0 && fcrack(i,j-2) > 0)
                k = k + 1;
                voidCrack(k, 1:2) = [i, j-1];
                if ( frack(i,j-2) > fcrack(i,j) )
                    fcrack(i,j-1) = fcrack(i,j)+(fcrack(i,j-2)-fcrack(i,j))/2;
                else
                    fcrack(i,j-1) = fcrack(i,j-2) + (fcrack(i,j) - fcrack(i,j-2)/2);
                end
            end
        end
    end
end
```

Figure 4.9: *For-loop from the Matlab script which checks (in the direction of the length of the beam) if voids are present in either side of a crack position and interpolates linearly in between if detected.*

Where $fcrack$ in figure 4.9 is an m x n-matrix where the crack widths are allocated into their respective nodal position by row (m) and column (n) and row and col are vectors containing indices describing the row and column position for each crack width in the $fcrack$-matrix, respectively.

The for-loop computes the iteration by going through the matrix row by row and column by column and upon finding a crack, checks if there are void elements in between two seemingly adjacent crack widths for that specific crack. If such a void is located, the for-loop interpolates a crack width magnitude linearly between them, using the contiguous crack magnitudes of the void cell. This optimization was needed to assure continuous cracks. To illustrate the optimization made, the interpolated crack widths and their positions are shown in figure 4.10.

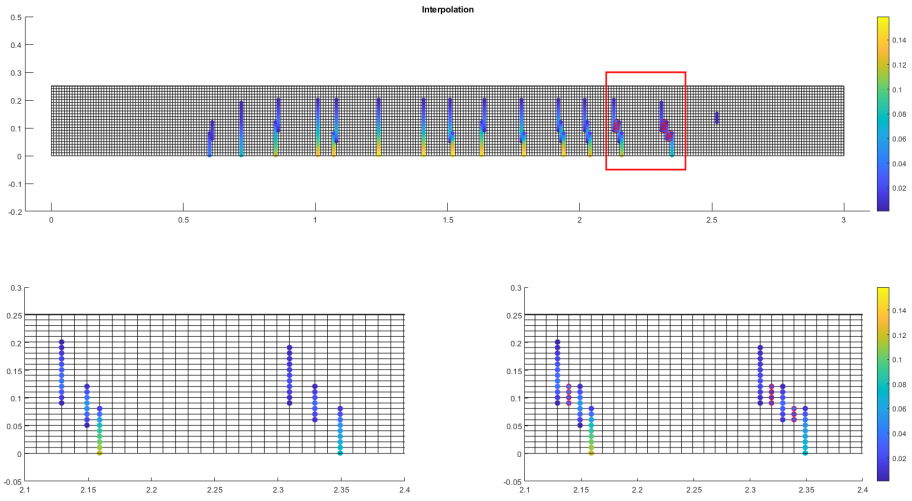


Figure 4.10: *Top: The original crack allocation complemented with interpolated values marked with red dots. Bottom left: Magnification of inclined cracks containing voids. Bottom right: Magnification containing interpolated values which are marked with red dots.*

After solving the problem with discontinuous cracks, the next optimization which had to be made was to solve the thickness of the inclined cracks which arose when interpolating them. If the crack information gathered from figure 4.10 was to be inserted into an FE software as are, the local strength of the concrete would be highly underestimated and would result in an improper estimation of the concrete's structural behaviour. This is because the implementation into the FE software is made by reduced tensile strength of the cracked elements in the mesh. Since the mesh size used for computing the scatter plots was set to 10 mm, reducing as many as three contiguous elements in a single row - as illustrated in figure 4.10 - would be equivalent to modelling a crack with a magnitude of 30 mm. Henceforth, the crack profiles were adjusted using the Matlab code shown in figure 4.11.

```
% Perform the morph of cracks
I = fcrack;
J = bwmorph(bwskel(logical(I)),'diag');
fcrack = J.*fcrack;
```

Figure 4.11: *Matlab script which reduces the thickness of the inclined cracks.*

The built-in Matlab-functions *bwmorph* and *bwskel* convert all thick crack-profiles into slimmer prevalences using morphological operations on each respective binary image of the provided crack-matrix, I, which contains the same information as the aforementioned $fcrack$-matrix. The order of the functions is that *bwskel* initially finds the skeleton of each crack, using an erosion function. This function removes all external pixels of the crack profiles until nothing but one pixel wide lines remain. Furthermore, to make the crack profiles continuous again, the function *bwmorph*

applies a filling to the profile, depending on the input argument given to the function. In this case the input argument *diag* was used since a diagonal filling is desired for the inclined cracks. This binary information is then stored into a matrix J which in terms only contains zeros and ones to describe the binary image of the new morphed cracks. This matrix is in turn multiplied with the original $frack$-matrix to formulate the updated $fcrack$-matrix which now contains the morphed cracks. Scattering the new crack profiles then gives the plot shown in figure 4.12, where the morphed cracks are illustrated. The reader is referred to the MathWorks documentation for further reading, see MathWorks (2021b) and MathWorks (2021a).

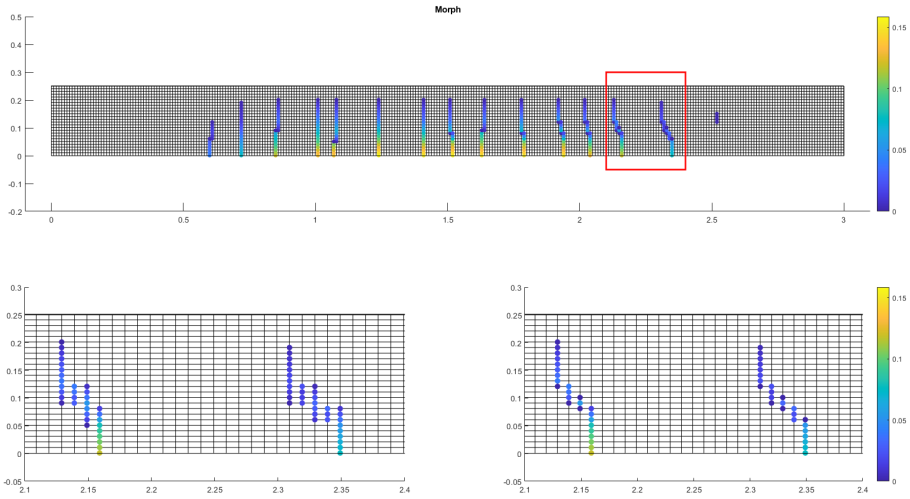


Figure 4.12: *Top: The new crack profiles after using morphological operation. Bottom left: Magnification of the crack profiles before morphological operations. Bottom right: Magnification of the crack profiles after morphological operations.*

Since the nodal cracks which are being handled in the Matlab script are numerically scattered and interpolated following the propagation of cracks retrieved from the DOFS, the crack information becomes smeared over a number of nodes to represent the inclination of the crack. These cracked nodes in turn do not individually represent the actual crack width but rather the smearing of a single crack width in a single element row, meaning the sum of the cracked nodes represent the actual crack width. This becomes problematic when performing morphological operations on the crack profiles since it removes vital information about the cracks which is not transferred to the updated crack profiles, therefore overestimating the concrete strength in inclined sections. This can be seen clearly when looking at the magnified parts of the beam in figure 4.12. While considering the colour bar, the figure shows that cracked nodes are removed but the magnitude of the remaining nodes are not increased in value.

To solve this, a for-loop was initiated which counter checks the original crack-matrix with the updated one, containing morphed cracks, to check which values have been removed, following the morphing of cracks. Upon finding removed values, it simply

adds them to the remaining node in the updated crack-matrix, row-wise, to ensure a less improper handling of the cracks. This is shown more clearly in figure 4.13 where the colour bar more clearly illustrates the increased magnitude in the remaining nodes.

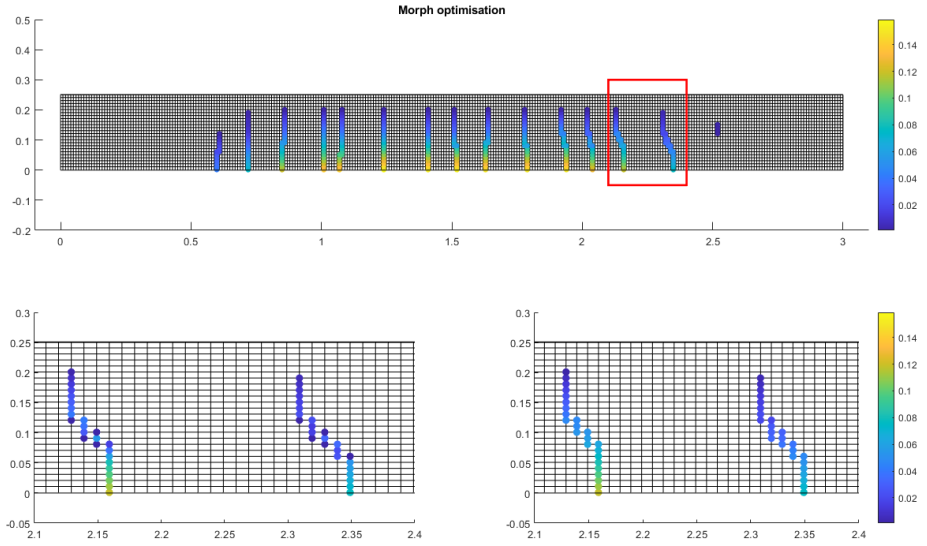


Figure 4.13: *Top: The new crack profiles after complementing the morphed cracks. Bottom left: Magnification of the morphed crack profiles. Bottom right: Magnification of the optimised morphed crack profiles.*

The final product of the Matlab script gives the two matrices shown in equation (4.3). Where N is a n x 3-matrix containing the nodal positions for each crack in a Cartesian manner of (x, y, z)-coordinates and the other called *CSI* (*crack Strength Input*) is a n x 2-matrix containing the corresponding reduced tensile strength and ultimate strain of the crack, where n is the number of cracked nodes in the mesh grid. The final Matlab script was named *Crack Translator* and can be found in appendix B.

$$N = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix}, \quad CSI = \begin{bmatrix} f_{t,1} & \varepsilon_{ult,1} \\ \vdots & \vdots \\ f_{t,n} & \varepsilon_{ult,n} \end{bmatrix} \tag{4.3}$$

4.3 Implementation of cracks into FE software

The implementation of the crack information created in Matlab into the FE software Diana was done by creating three separate scripts in Python. One that creates the geometry, material properties and mesh of the RC beam, the *Mesh Generator*, another that implements the cracks into the generated mesh, the *Crack Generator* and finally the *Analysis Generator*, which creates the analysis for the structure. The scripts in Python were all written using commands native to the DIANAIE, which allows Python-scripts to be run directly through the command console.

The reason the program code was divided into three separate scripts was to save computational time, mainly because the crack implementation takes quite some time to run. This generates two base models, one with and one without crack inclusion, on which the analysis is based. Having them separated make it easier to choose if a traditional FE analysis or an analysis based on the WEA is to be performed. As they are saved as models, new analyses can be run without having to do the entire process of recreating the geometry, mesh generation and crack implementation every time.

4.3.1 Mesh generator

The script was built in a way which enables easy modification of geometrical parameters of the concrete and reinforcement, material parameters etc. All geometry specific parameters are stated in the beginning of the script and all positions and input parameters such as reinforcement areas etc. are calculated and implemented automatically using several for-loops of different natures throughout the script. See appendix A for the full script.

In addition to generating the mesh, several important steps needed to be included in the mesh generator. The material model for the concrete was chosen as a total strain based crack model with a rotating crack orientation and Hordijk and Parabolic softening to describe the tensile and compressive behaviour, respectively (see section 2.1 for reference). Material parameters used for the concrete are presented in table 4.1. Furthermore, the reinforcement steel was modelled using Von Mises plasticity with strain hardening. Subsequently, bond-slip between the reinforcement and concrete was considered and implemented using guidelines set by fib Model Code 2010 (FIB, 2013) with parameters as per table 4.2. These parameters was provided by the supervision team of the thesis.

Table 4.1: *Material properties for the concrete material.*

Material property	Value
Linear material properties	
Young's modulus, E_{cm} [GPa]	37
Poisson's ratio, ν [-]	0.15
Tensile behaviour; Hordijk	
Tensile strength of concrete, f_{ctm} [MPa]	3.71
Mode-I tensile fracture energy, G_F [N/m]	151.6
Crack bandwidth, h [m]	0.01
Compressive behaviour; Parabolic	
Compressive strength of concrete, f_{cm} [MPa]	58
Compressive fracture energy, G_C [N/m]	37094

Table 4.2: *Material properties for the reinforcement steel.*

Material property	Value
Reinforcement properties	
Yield strength, f_y [MPa]	527.4
Ultimate strength, f_u [MPa]	649.1
Plastic elongation at maximum force, A_g [%]	10.7
Modulus of elasticity, E_s [GPa]	199.3
Poisson's ratio, ν [-]	0.3
Bond-slip failure model; CEB-FIB 2010	
Normal stiffness modulus, k_n [N/m^3]	$1{\cdot}10^{14}$
Shear stiffness modulus, k_t [N/m^3]	$1{\cdot}10^{11}$
Maximum stiffness stress, τ_{max} [MPa]	20
Ultimate shear stress, τ_f [MPa]	8
Linearised initial slip section, s_0 [m]	$1{\cdot}10^{-5}$
Relative slip section, s_1 [m]	0.001
Relative slip section, s_2 [m]	0.002
Relative slip section, s_3 [m]	0.008
Exponent, α [-]	0.4

Running this script using quadrilateral mesh elements with a size of 10 mm in Diana yielded the geometry shown in figure 4.14 with the corresponding mesh illustrated in figure 4.15.

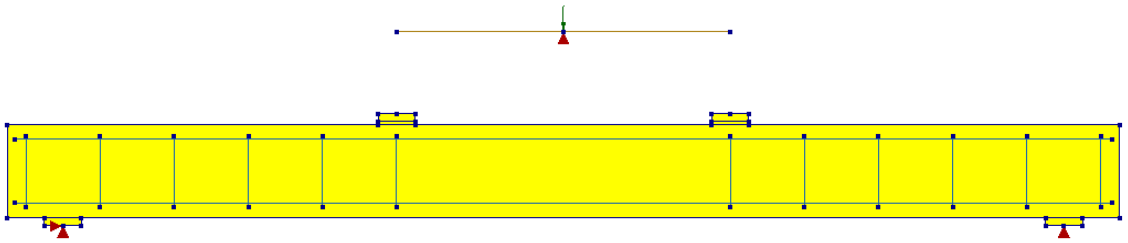


Figure 4.14: *Geometry of the beam where the reinforcement and stirrup layout, supports, load plates and load distribution beam are illustrated.*

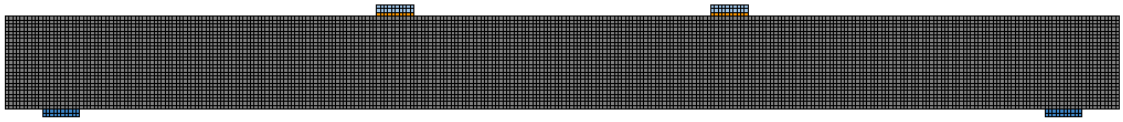


Figure 4.15: *Final mesh used for implementation of cracks.*

4.3.2 Crack generator

To enable easy management of the crack positions and magnitudes produced in Matlab, the two matrices described using equation (4.3); N and CSI, were exported into two separate .csv-files. These were then imported into the Python script where a for-loop was initiated which went through the matrices row by row and first found and selected the element in the mesh from the *Mesh Generator* which lie closest to the information given from the N-matrix. Once an element was found, information about the strength reduction was extracted from the CSI-matrix and assigned to an altered version of the original concrete material which in turn was assigned to the selected element. This course of actions was repeated for all crack-positions provided through the CSI- and N-matrices and the final implementation of cracks was produced, shown in figure 4.16. See appendix C for the full script.

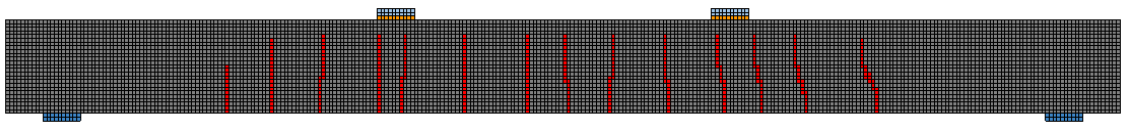


Figure 4.16: *Updated mesh with weakened elements in red.*

4.3.3 Analysis generator

The *Analysis Generator*, as the name implies, adds analyses based on the users choice to the FE model with crack inclusion created with the *Crack Generator*. In this particular case, it applies a structural nonlinear analysis to the model and increases the load (in this case prescribed deformation) up until failure of the beam. The post-processing of the results is then done manually by the user. In this case, crack widths in the formerly included crack regions (see figure 4.16 for reference), displacement in the middle of the beam and reaction forces was gathered to be compared in upcoming chapters. See appendix D for the full script.

5

Results and Calibration of Model Using Reference Beam

Since the inclusion of cracks in Diana was made using SHM - where exact crack widths could be extracted - one could argue that the cracks which propagates when running FE analysis on the model containing cracks should be exactly the same as the ones measured with the optical fibres when evaluated under the same loading. To check this, another test beam, *PC1*, with the same geometry and reinforcement layout as shown in figure 4.1, with concrete parameters as per table 5.1, was investigated. The acquisition of DOFS data was conducted by laboratory testing of PC1 at an age of 14 days after casting, as illustrated in figure 5.1, and material tests were only made on the on the compressive strength. The other strength parameters therefore had to be estimated using EN-1992-1-1 (2004). The concrete parameters were based on the 28-day cube compressive strength, $f_{cm,cube28}$, and are presented in table 5.1 with the calculations in appendix F.1. Moreover, no testing of the reinforcement was made, either, so it was assumed to have the same characteristics as the ones presented in Dr. Blomfors' thesis (see Blomfors (2021) for reference). Furthermore, in figure 5.2, the crack profiles translated from the DOFS data of the reference beam, PC1, are illustrated.

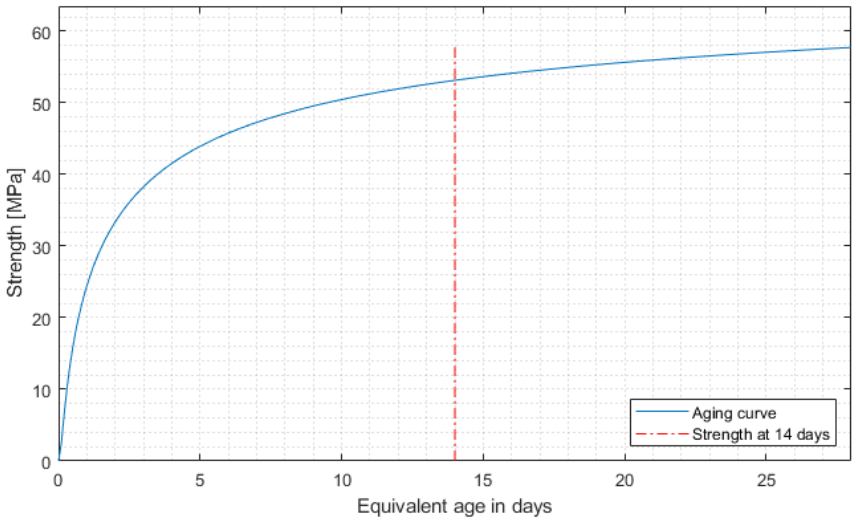


Figure 5.1: *Curve describing the increase in strength of concrete due to aging. Strength at 14 days indicated.*

Table 5.1: *Material properties for concrete and reinforcement used in PC1. Concrete strength determined by testing at 28 days and adjusted with aging effects due to pre-cracking tests being carried out at 14 days strength. Standard deviations given in parenthesis.*

Material property	**Value**
Concrete properties	
Compressive strength, $f_{\mathrm{cm,cube}}$ [MPa]	68.2 (3.9)
Compressive strength, f_{cm} [MPa]	53.152
Tensile strength, f_{ctm} [MPa]	3.733
Young's modulus, E_{cm} [GPa]	36.314
Tensile fracture energy, G_{F} [N/m]	147.77
Ultimate concrete strain, $\varepsilon_{\mathrm{c,u}}$ [%]	0.389
Reinforcement properties	
Yield strength, f_{y} [MPa]	527.4
Ultimate strength, f_{u} [MPa]	649.1
Plastic elongation at maximum force, A_{g} [%]	10.7
Modulus of elasticity, E_{s} [%]	199.3

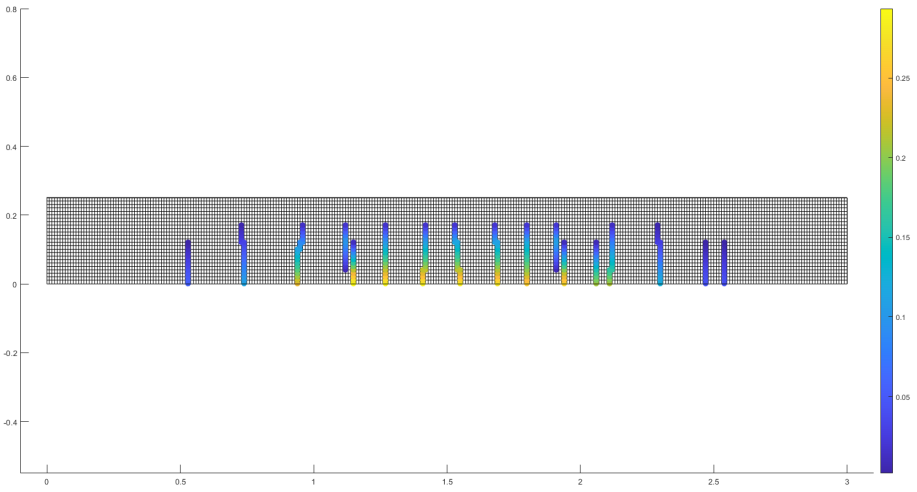


Figure 5.2: *Crack profiles for reference beam PC1 generated using the Crack Translator and DOFS data.*

5.1 Comparison of crack propagation

The calibration was done by comparing crack widths gathered from the DOFS with the same cracks which had propagated in Diana when running the FE model with Crack Inclusion (FEC). The cracks *C6, C7, C8* and *C9* (illustrated in figure 5.3) were deemed to be most suitable for comparison since they were well developed bending cracks. Comparing the crack widths in each element of the FE mesh to the corresponding cracked node from the DOFS data gave the results shown in table 5.2.

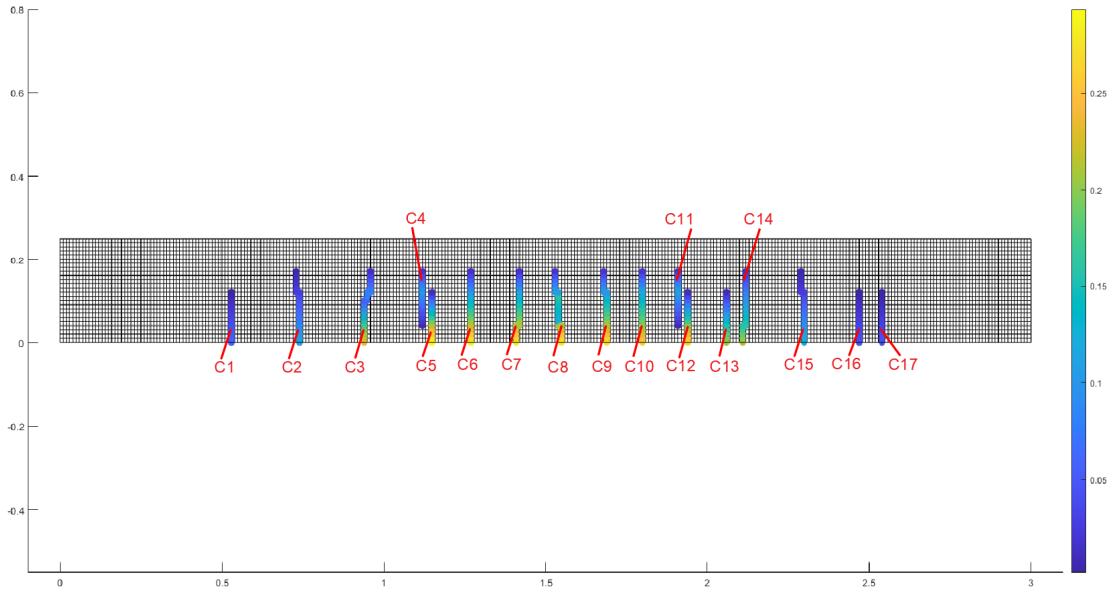


Figure 5.3: *Numbering of crack profiles, where crack indices C6, C7, C8 and C9 have been used to perform calibrations on the Matlab script which calculates the weakened element properties.*

Table 5.2: *Crack widths for the crack profiles C6, C7, C8 and C9 in figure 5.3 from the bottom of the beam and up after the first FE analysis. FE values for each crack profile in the left column and DOFS values in the right along with the average difference for each crack profile in the bottom. All values in mm unless stated otherwise.*

C6		**C7**		**C8**		**C9**	
FE	**DOFS**	**FE**	**DOFS**	**FE**	**DOFS**	**FE**	**DOFS**
0.228	0.287	0.245	0.284	0.240	0.281	0.216	0.278
0.212	0.269	0.226	0.267	0.221	0.264	0.201	0.261
0.195	0.253	0.206	0.249	0.203	0.247	0.184	0.244
0.175	0.235	0.176	0.232	0.153	0.230	0.167	0.227
0.163	0.219	0.137	0.218	0.157	0.216	0.155	0.213
0.157	0.205	0.127	0.204	0.160	0.204	0.147	0.200
0.147	0.190	0.152	0.191	0.151	0.191	0.137	0.188
0.135	0.175	0.140	0.178	0.141	0.179	0.126	0.175
0.123	0.161	0.130	0.164	0.131	0.166	0.114	0.162
0.114	0.146	0.129	0.151	0.130	0.154	0.106	0.149
0.100	0.132	0.119	0.138	0.135	0.142	0.095	0.136
0.085	0.117					0.088	0.124
Diff:	−23.6%	**Diff:**	−21.5%	**Diff:**	−21.8%	**Diff:**	−26.9%

Judging by the average differences in table 5.2, the model beam shows a stiffer behaviour than the reference beam as crack widths are predominately smaller in the FEC compared to the DOFS data. One reason could be that the FE model overestimates the concrete strength in the cracked sections.

Even if some discrepancies can be expected between FE analyses and real-life cases, a first attempt of a combined sensitivity analysis of input parameters and calibration of the model was performed. This was done with the ambition of fine tuning the Matlab script which calculates the weakened material properties which are inserted as cracks in the FE model. The model presented in this section will, in upcoming sections, be referred to as the *base case*.

5.1.1 First calibration

The first parameter altered was the tensile fracture energy, G_F. Originally, the fracture energy was determined using the fib Model Code 2010 (FIB, 2013) (see equation 2.1) to 147.77 N/m, and since it is a parameter which describes the energy required to obtain a fully formed crack, it was decreased to 100 N/m in an attempt to reduce the underestimation of crack widths. Lowering the fracture energy was done based on the idea that the amount of energy needed to be put into the system for the concrete stresses to reach the tensile strength, f_{ctm}, would be lower. Hence, with the same amount of force applied as before, the crack widths in the FEC model should, theoretically, increase due to the decrease of fracture energy. Consequently, the compressive fracture energy, G_C, is connected to the tensile fracture energy as per equation 2.1, and was altered accordingly.

This alteration gave the crack widths shown in table 5.3 where the same crack profiles as in table 5.2 were compared in terms of crack widths from FEC and DOFS data. Furthermore, in table 5.3, the factor *Imp* was introduced, which describes how much the average difference for each crack profile has improved after the calibration.

Table 5.3: *Crack widths for the crack profiles C6, C7, C8 and C9 in figure 5.3 from the bottom of the beam and up after the first calibration of the FE analysis. FE values for each crack profile in the left column and DOFS values in the right along with the average difference for each crack profile in the bottom. All values in mm unless stated otherwise.*

C6		**C7**		**C8**		**C9**	
FE	**DOFS**	**FE**	**DOFS**	**FE**	**DOFS**	**FE**	**DOFS**
0.230	0.287	0.249	0.284	0.245	0.281	0.218	0.278
0.215	0.269	0.229	0.267	0.225	0.264	0.203	0.261
0.197	0.253	0.208	0.249	0.206	0.247	0.186	0.244
0.176	0.235	0.180	0.232	0.156	0.230	0.169	0.227
0.164	0.219	0.169	0.218	0.161	0.216	0.157	0.213
0.159	0.205	0.163	0.204	0.163	0.204	0.150	0.200
0.149	0.190	0.155	0.191	0.153	0.191	0.140	0.188
0.137	0.175	0.143	0.178	0.143	0.179	0.129	0.175
0.125	0.161	0.130	0.164	0.130	0.166	0.117	0.162
0.112	0.146	0.117	0.151	0.117	0.154	0.105	0.149
0.099	0.132	0.103	0.138	0.104	0.142	0.093	0.136
0.086	0.117					0.088	0.124
Diff:	−23.1%	**Diff:**	−19.5%	**Diff:**	−21.3%	**Diff:**	−26.2%
Imp:	2.1%	**Imp:**	9.2%	**Imp:**	2.2%	**Imp:**	2.4%

Lowering the fracture energy gave an average improvement of 4% in the crack profiles, but even if the crack width in the FEC model became more in line with the DOFS data, the differences are still notably large, which entail further calibrations.

5.1.2 Second calibration

Similarly, as prior to the first calibration, it could be noted that the FEC was still overestimating the concrete strength. Consequently, some material properties of the concrete were once again lowered. This time the tensile strength of the concrete was lowered, since this was the parameter that was most uncertain due to the lack of tensile testing. It was decreased to a seemingly low value to further check the sensitivity of the model with regard to this parameter. The material properties for the concrete used in the second calibration are presented in table 5.4. Doing these alterations gave the updated crack widths presented in table 5.5.

Table 5.4: *Material properties of the concrete used in the second calibration of the FEC model. All parameters are lower bound values determined from laboratory tests on the concrete material.*

Material property	**Value**	**Note**
Concrete properties		
Compressive strength, f_{cm} [MPa]	53.152	Unchanged
Tensile strength, f_{ctm} [MPa]	3	Estimated
Young's modulus, E_{cm} [GPa]	36.314	Unchanged
Tensile fracture energy, G_{F} [N/m]	100	Estimated
Compressive fracture energy, G_{C} [N/m]	25000	Equation 2.1
Ultimate concrete strain, $\varepsilon_{\mathrm{c,u}}$ [%]	0.389	Unchanged

Table 5.5: *Crack widths for the crack profiles C6, C7, C8 and C9 in figure 5.3 from the bottom of the beam and up after the second calibration of the FE analysis. FE values for each crack profile in the left column and DOFS values in the right along with the average difference for each crack profile in the bottom. All values in mm unless stated otherwise.*

C6		**C7**		**C8**		**C9**	
FE	**DOFS**	**FE**	**DOFS**	**FE**	**DOFS**	**FE**	**DOFS**
0.278	0.287	0.296	0.284	0.290	0.281	0.256	0.278
0.251	0.269	0.260	0.267	0.256	0.264	0.233	0.261
0.221	0.253	0.220	0.249	0.217	0.247	0.207	0.244
0.149	0.235	0.157	0.232	0.163	0.230	0.155	0.227
0.137	0.219	0.173	0.218	0.147	0.216	0.143	0.213
0.179	0.205	0.174	0.204	0.189	0.204	0.167	0.200
0.173	0.190	0.178	0.191	0.175	0.191	0.159	0.188
0.162	0.175	0.166	0.178	0.164	0.179	0.148	0.175
0.149	0.161	0.152	0.164	0.151	0.166	0.135	0.162
0.134	0.146	0.137	0.151	0.136	0.154	0.121	0.149
0.121	0.132	0.112	0.138	0.123	0.142	0.107	0.136
0.110	0.117					0.109	0.124
Diff:	−13.1%	**Diff:**	−11.5%	**Diff:**	−12.0%	**Diff:**	−17.9%
Imp:	43.2%	**Imp:**	47.9%	**Imp:**	49.8%	**Imp:**	31.7%

Lowering the strength parameters of the concrete showed a great improvement in terms of crack widths, where the average improvement over the evaluated crack profiles became 43.2% and the average difference, relative to the DOFS data, decreased to 13.6%.

5.2 Final model

In light of the presented results in section 5.1, the model presented in section 5.1.2 was chosen as the final, due to the vast improvements in terms of crack widths. To illustrate the weakening of strength parameters for the base case, calibration 1 and calibration 2, the different tension softening curves for each case are presented in figure 5.4.

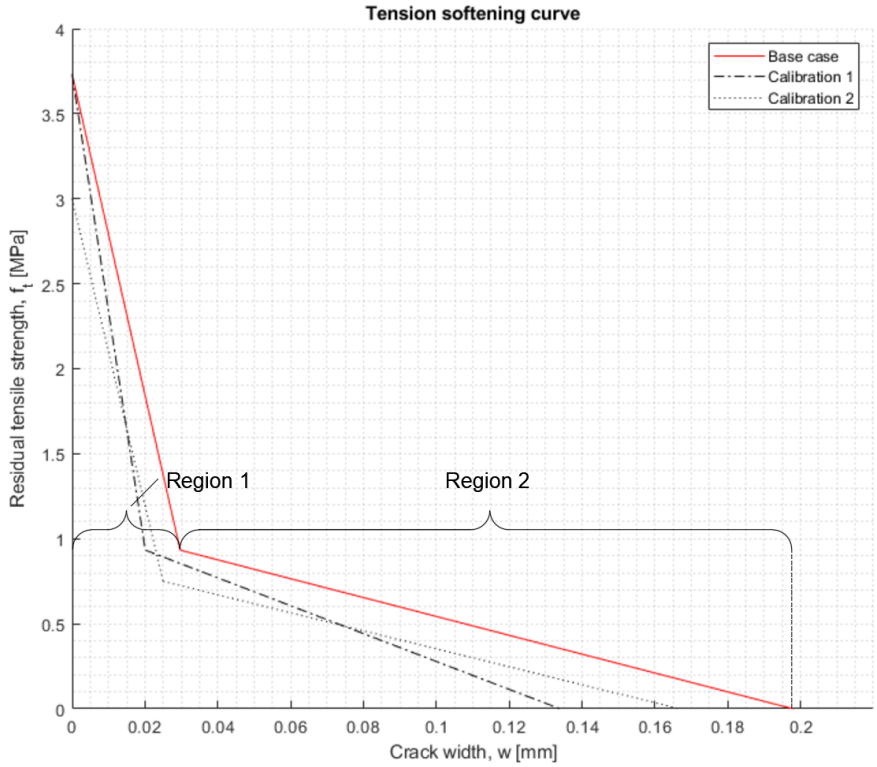


Figure 5.4: *Tension softening curves used for determining the residual tensile strength for each element in the FE mesh for the three different cases. Region 1 and 2 are relative depending on the respective softening curve but has only been visualized on the base case.*

To put the tension softening curves into perspective, the amount of cracked elements in each linear region of the curves are presented in table 5.6. Additionally, the *relative weakening* for each of the three curves was evaluated in terms of the average reduction of each cracked element compared to the original tensile strength. This gives an indication of the difference in how much the concrete properties in the cracked elements have been weakened in comparison to the sound concrete elements.

Table 5.6: *Number of cracks in each region of the tension softening curve in figure 5.4 where region 1 refers to the region before the kink-point and region 2 to the region after and up to the ultimate crack limit. Fully opened refers to the part beyond region 2.*

	Base case	**Calibration 1**	**Calibration 2**
Region 1	52	35	44
Region 2	178	148	165
Fully opened	47	94	68
Relative weakening	79.4 %	85.4 %	82.14 %

5.3 Standard finite element crack width comparison

To see how the final model performed in terms of predicting crack widths, a Traditional FE model (FET) was built with the *Mesh Generator* with the concrete parameters given in table 5.4. This was run with an analysis based on the SCA. Since the formation of cracks in the FET analysis did not exactly match the weakened element's positions from the FEC, the closest cracked elements to the studied cracks in the calibration stage (see figure 5.3) were studied.

Since the cracks from the FET analysis did not form as predictably as in the FEC analysis, only the bottom element crack widths were extracted and compared - results presented in table 5.7.

Table 5.7: *Crack widths for the crack profiles C6, C7, C8 and C9 in figure 5.3 in bottom most strand of the beam. FEC and FET analysis results presented with the difference relative to the DOFS in parentheses (%). All values in mm unless stated otherwise.*

C6		**C7**		**C8**		**C9**	
DOFS	0.287	**DOFS**	0.284	**DOFS**	0.281	**DOFS**	0.278
FEC	0.278 (−3.1%)	**FEC**	0.296 (4.2%)	**FEC**	0.290 (3.2%)	**FEC**	0.256 (−7.9%)
FET	0.132 (−54%)	**FET**	0.144 (−49%)	**FET**	0.128 (−54%)	**FET**	0.122 (−56%)

From table 5.7, it can be concluded that implementing DOFS data through the WEA greatly improves the prediction of crack widths compared to FET analysis. The inclusion of the already existing cracks make the model more precise so that cracks form as in reality to a higher degree, in combination with more accurate localisation of strains in the beam.

5.4 Beam behaviour

After an additional curing of 493 days after the pre-cracking test, the reference beam *PC1* was loaded to failure in a four-point bending test. This entailed that, in order to compare the FE models with the reference beam, further development of the scripts had to be conducted to account for the increase in concrete strength due to ageing of the uncracked concrete. The *Mesh Generator* was modified to reflect the ageing of sound concrete as stated in EN-1992-1-1 (2004), by creating another concrete material and assigning all uncracked elements in the FE mesh this strengthened concrete, see table 5.8. The weakened (cracked) elements, on the other hand, were modelled as before, using the same material properties as given in the final crack calibration (see table 5.4).

The calculation of the aged concrete strength was based on material tests made on concrete specimens at an age of 293 days (see appendix E) where all specific material properties - except for the tensile strength - had been tested. The aged concrete properties are presented in table 5.8 and the calculations can be found in appendix F.2.

The FEC model was then subjected to the same analysis and load as for the crack calibration (a prescribed deformation) until failure occurred. In addition, an FET model with concrete properties as per table 5.8, was also analysed and loaded to failure, to compare the behaviour of the two models. The load-displacement curves for PC1 from experiments and the FEC and FET analyses are presented in figure 5.5

Table 5.8: *Material parameters used for the sound concrete elements in load-displacement comparison. The concrete is strengthened in accordance with EN-1992-1-1 (2004)(*) to the given material parameters given in appendix E.*

Material property	**Value**	**Note**
Concrete properties		
Young's modulus, E_{cm} [GPa]	33.783	*
Compressive strength, f_{cm} [MPa]	75.013	*
Tensile strength, f_{ctm} [MPa]	4.592	*
Tensile fracture energy, G_{F} [N/m]	100	Unchanged
Compressive fracture energy, G_{C} [N/m]	25000	Unchanged
Ultimate strain, $\varepsilon_{\mathrm{c,u}}$ [%]	0.2875	*

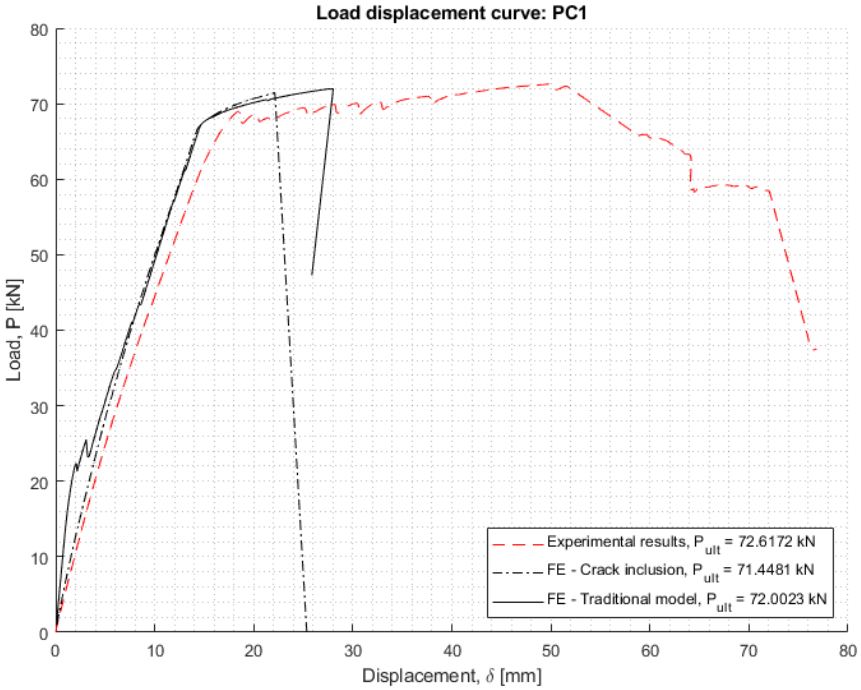


Figure 5.5: *Load-displacement curves for the reference beam PC1, FEC and FET model. Ultimate load for the three cases indicated in the legend.*

As can be seen in figure 5.5, the FE models resulted in a much stiffer behaviour in terms of flexural stiffness. The FET model shows an uncracked behaviour up to ~12 kN, which is expected. After cracking of the FET model, the behaviour of the two FE models coincide after a load level of ~40 kN, and exhibit more or less the same behaviour up to failure, with a slightly higher ultimate capacity for FET model. Another note to be made is the brittle failure of the FE models, whereas *PC1* has a more ductile failure. It was shown in the analyses that the failure mode was due to localisation of high stress concentrations under the loading plates, which caused crushing of concrete. The analysis then diverged which led to ultimate failure of the beam.

6
Discussion

It was shown in chapter 5 that there were differences between the FEC model and reference beam in terms of crack widths, pre- and post-yielding behaviour and crack propagation. As stated, some discrepancies can always be expected when comparing FEA and real life cases, but when evaluating the results, some reasons for the discrepancies can be established.

The reasons for the differences between the FEC model and the reference beam are manifold, since the propagation of cracks and their magnitude is affected by lots of different factors such as material models, parameter and the way SHM is included in the model to name a few. But performing the calibration of the crack inclusion scripts gave a good indication of the sensitivity of certain material parameters. It was observed that reducing the tensile fracture energy of the concrete, G_F, gave a slight improvement in terms of crack widths in the FE analysis. However, when reducing the tensile strength of the concrete, f_{ctm} - in addition to the fracture energy - the crack widths correlated considerably well. This could be explained by the difference in the tension softening curves in figure 5.4, which were used to determine the residual tensile strength of the cracked elements in the FEC model. These curves indicate that lowering the fracture energy decreases the crack width limit, w_s, at the kink-point, making the slope in region 1 steeper, ergo, making the reduction greater, whereas lowering the tensile strength mainly lowers the corresponding residual tensile strength at w_s. Lowering the tensile strength, however, does not necessarily entail a decrease in
w_s (see equation 2.3 and figure 2.5 for reference).

Furthermore, by quantifying the number of cracks in each respective region of the tension softening curves in table 5.6, indicate that the first calibration contained a higher amount of more greatly reduced elements as well as a larger relative weakening than the second calibration. The reason for the second calibration, however, generating a better resemblance in terms of crack widths in the FE analysis can, once again, be explained by looking at figure 5.4, where even if the relative weakening is slightly larger for the first calibration, the second calibration is based on a lower tensile strength, making the weakening far greater than for the first calibration explaining the vastly improved results for this case.

In light of these arguments, it is evident that the tension softening curve for calibration 2 greatly improved the FE results in terms of crack widths, which leads to the next topic of discussion; the tensile strength of the concrete. As stated in chapter 5,

the testing of the tensile strength for the concrete used in reference beam PC1 was never done, leading to it being estimated. The sensitivity of this particular parameter shows that it could be preferable to conduct tests for the tensile strength of concrete, to not rely solely on estimations from EN-1992-1-1 (2004). Furthermore, if the tensile strength had been tested, the tension softening curve could have been fitted exactly to the test data and this source of error could have been ignored.

Another peculiar behaviour, is the load-displacement curves in figure 5.5, where the FEC model shows a considerably good resemblance to the reference beam for lower loads, with regard to the stiffness. However, for increasing load, the FEC model starts to coincide with the FET model instead. The fact that the two models exhibit similar behaviour for higher loads could be explained by the fact that after cracking of the FET model, the two models have the similar material parameters, since the cracked elements of the FEC are based on the concrete parameters of sound concrete and are both based on the SCA. This does, however, not explain the deviation from the results from the experiments.

One explanation for the differences on terms of stiffness with the experimental test, could be a parameter that was never altered, but included in the models. The bond-slip relationship between the concrete and reinforcement has a great influence on the crack width and by extension, also the rigidity of the model. Calibration of the bond-slip parameters in the models could therefore have had a major importance, but it was overlooked in the process of trying to understand the sensitivity of other material parameters. The configuration of bond-slip that the model is based on at the present time, could have a stiffening influence due to its influence on tension stiffening. Analyses showed the tension stiffening behaviour in the uncracked sections between cracks as stresses built up around the reinforcement. However, there were only a few new cracks introduced to the FEC, which indicates that the concrete bond stresses in these sections never reached its tensile strength, f_{ct}. As Blomfors (2021) mentions, this modification is expected to result in a less stiff behaviour, considering the load-displacement relationship.

The divergence of the models in ULS (where a brittle behaviour was noted for both analyses) can be considered as a result of the compressive softening branch not being properly introduced in the model. A calibration of this parameter to a more ductile branch, could in all likeliness have improved the model in terms of post-yielding behaviour, since it would reduce the stress concentrations that caused failure at the loading plates. With this modification, the FE models would probably fail due to bending rather than crushing of the concrete. Furthermore, even if the ultimate capacities of the models in terms of ultimate load coincided reasonably well to the experimental results, see figure 5.5, the steepness of the curves indicate that if failure had not occurred at this state of loading, the ultimate load would have been slightly higher still. As mentioned, this in turn entails further modification of the reinforcement material parameters, since these govern the behaviour to a much higher degree in post-yielding stage of the beam.

7
Conclusion

To sum up everything that has been stated so far, it can be concluded that the aim of the thesis was achieved and it can be observed that up to the load of where which DOFS data had been measured, the estimation through the FEC model in terms of structural behaviour was improved compared to the FET model. Furthermore, it was proven that, even though there were some deviations in the crack widths estimated by the FEC model compared to the laboratory tests, they gave a much better estimation when comparing to an FET model.

In terms of the failure mode of the RC beam, both the FEC and FET analyses showed a brittle behaviour when reaching the ultimate load and probably failed earlier than expected. They also showed a slightly stiffer behaviour but behaved well considering State II as a whole, but some aspects might not have been introduced properly in the model.

As discussed in chapter 6, further sensitivity studies of some of the material parameters need to be conducted in the future in order to better match the results from experimental tests, particularly the bond-slip relationship between concrete and reinforcement and the compressive softening branch of concrete. As of now, the concrete material parameters are the only ones looked at when calibrating the model, but with new parameters altered, these might have to be calibrated once again, due to changes of the model behaviour.

Even if further improvements can be made, DOFS can still be used to localise cracks and now, further characterise their magnitude and effect on the structure, improving the assessment of structural damage of new structures. This was made evident by the crack widths presented in table 5.7, where (even if the crack widths determined using the FEC model differed slightly from the ones measured using DOFS) it was shown that a great improvement had been made compared to an FET model, in terms of estimating the magnitude of cracks. Furthermore, using crack inclusion as described in this thesis, enables cracks in an FEC model to localise as predicted and actions can be taken in terms of prolonging a structure's service life; granting a great improvement from an economic and environmental point of view.

7.1 Future aspects

As both discussed and concluded, the discrepancies in terms of material data need to be limited, if further tests where crack inclusion in FEA is researched. One suggestion could be to test parameters of the concrete material on the same day as loading tests for a e.g. four-point bending, to exclude errors due to ageing calculations. It is yet to be determined if this would result in an improved FE analysis, but it would, however, give enough data to preclude certain source of errors. In terms of the testing set-up, DOFS need to be included during the entire loading cycle of the reference beams so all data can be registered and included.

Further, 3D-models need to be evaluated as well as different crack inclusion approaches (such as the DCA) to ensure the reliability of the crack inclusion if used in real-life cases. As of now, only 2D-models may be evaluated using the weakened element approach in conjunction with the smeared crack approach, but the foundation of including cracks has been laid by the creation of the Matlab- and Python-scripts presented throughout this thesis.

Finally, using a meshing tool with more capabilities than the DIANA built-in mesher, could be beneficial when considering crack inclusion, translated from DOFS. For example this could be done by constructing the mesh using a general mesher and import it into DIANA through a NASTRAN file - which the DIANA meshing editor is capable to translate (TNO DIANA BV, 2008). This could, eventually, improve the behaviour of the model, since a finer mesh would make the implementation of cracks as weakened elements resemble the real-life propagation of cracks more precisely. As of now, even if the mesh which was used in this thesis is quite fine, some detected cracked elements might only coincide with the extracted crack profile every so slightly, meaning that an exaggerated weakening would be applied to such an element. If the mesh could be refined, iteratively, the prevalence of the crack could be captured more precisely and would diminish the risk of unnecessary weakening of the surrounding concrete. A coarse illustration of this course of actions is illustrated in figure 7.1 where the mesh around the crack, extracted from DOFS, is iteratively refined to more exactly capture its prevalence; notice the difference in mesh size between 7.1b and 7.1e.

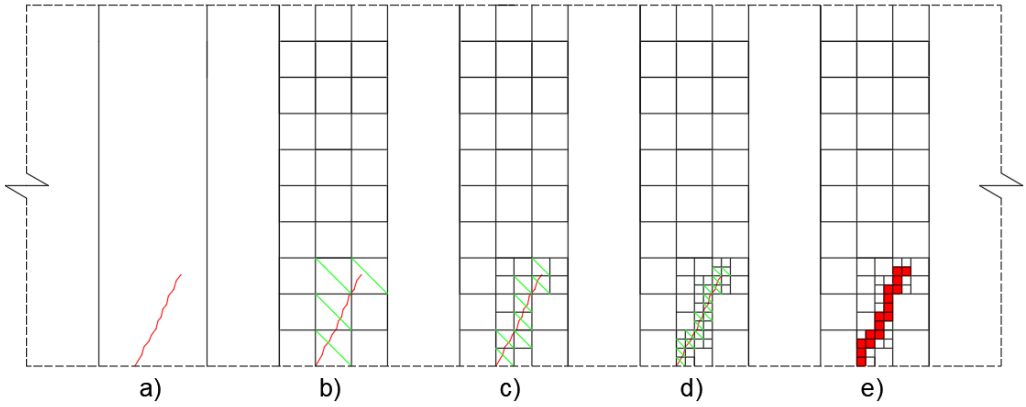


Figure 7.1: *a) Arbitrary crack profile in a section of a beam. b) Initial mesh and detection of cracked elements (green lines). c) Refinement of detected cracked elements and new detection of such. d) Further refinement and detection of cracked elements. e) Final detection and application of weakened elements properties to cracked elements. Could, theoretically, be refined indefinitely.*

This mesh optimisation was discussed in the early stages of the thesis but could not be achieved, unfortunately, since refinement of regions in the mesh using DIANA was not possible with the version used in this thesis.

Bibliography

Adachi, S. (2008). Distributed optical fiber sensors and their applications. *2008 SICE Annual Conference*, 329–333. https://doi.org/10.1109/SICE.2008.4654674 (cited on page 20)

Al-Emrani, M., Engström, B., Johansson, M. & Johansson, P. (2013). *Bärande konstruktioner del 1* (6., [rev.] uppl.). Chalmers tekniska högskola. (Cited on page 1).

Barrias, A., Casas, J. R. & Villalba, S. (2019). Shm of reinforced concrete elements by rayleigh backscattering dofs. *Frontiers in Built Environment*, *5*(30). https://doi.org/10.3389/fbuil.2019.00030 (cited on pages 21, 22)

Berrocal, C. G., Fernandez, I., Bado, M. F., Casas, J. R. & Rempling, R. (2021). Assessment and visualization of performance indicators of reinforced concrete beams by distributed optical fibre sensing. *Structural Health Monitoring*, 1–18 (cited on pages 2, 3, 5, 17, 20–23, 25–29, 32).

Berrocal, C. G., Fernandez, I. & Rempling, R. (2021a). Crack monitoring in reinforced concrete beams by distributed optical fiber sensors. *Structure and Infrastructure Engineering*, *17*(1), 124–139. https://doi.org/10.1080/15732479.2020.1731558 (cited on page 23)

Berrocal, C. G., Fernandez, I. & Rempling, R. (2021b). Crack monitoring in reinforced concrete beams by distributed optical fiber sensors. *Structure and Infrastructure Engineering*, *17*(1), 124–139. https://doi.org/10.1080/15732479.2020.1731558 (cited on page 27)

Blomfors, M. (2021). *Incorporation of pre-existing longitudinal cracks in finite element analyses of reinforced concrete beams without transverse reinforcement* (Doctoral dissertation). Chalmers University of Technology. Sven Hultins gata 6. (Cited on pages 3, 5, 14, 18, 25, 26, 29–31, 41, 54).

Blomfors, M., Lundgren, K. & Zandi, K. (2020). *Incorporation of pre-existing longitudinal cracks in finite element analyses of corroded reinforced concrete beams failing in anchorage* (Doctoral dissertation). Chalmers University of Technology. Sven Hultins gata 6. (Cited on pages 17, 18).

Brault, A. & Hoult, N. (2019). Monitoring reinforced concrete serviceability performance using fiber-optic sensors. *Structural Journal*, *116*(1), 57–70. https://doi.org/10.14359/51710870 (cited on page 23)

Broo, H., Lundgren, K. & Plos, M. (2008). *A guide to non-linear finite element modelling of shear and torsion in concrete bridges* (technical report) [Report RTD:2008:18, ISSN: 1652-9162]. Chalmers University of Technology, Department of Civil and Environmental Engineering, Division of Structural Engineering, Concrete Structures. (Cited on pages 13, 14).

Chistidis, P. & Leduc, G. (2009). *Longer and heavier vehicles for freight transport* (technical report) [ISSN: 1018-5593]. European Commission, Joint Research Centre, Institute for Prospective Technological Studies. https://doi.org/10.2791/12276. (Cited on page 2)

de Sousa Barrias, A. J. (2018). *Development of optical fibre distributed sensing for shm of bridges and large scale structures* (Doctoral dissertation) [Department of Civil and Environmental Engineering]. Universitat Politècnica de Catalunya (UPC). K2M Building, S1 Floor, S103-S104 Office North Campus, C / Jordi Girona, 08034 Barcelona. (Cited on page 3).

DIANA FEA. (2020). DIANA FEA: User's manual. https://dianafea.com/manuals/d104/Diana.html (cited on pages 3, 15)

Drake, D. A., Sullivan, R. W. & Wilson, J. C. (2018). Distributed strain sensing from different optical fiber configurations. *Inventions*, *3*(4). https://doi.org/:10.3390/inventions3040067 (cited on pages 21, 22)

EN-1992-1-1. (2004). *Eurocode 2: Design of concrete structures - part 1-1: General-rules and rules for buildings* (technical report) [Ref. No. EN 1992-1-1:2004: E]. European Committee for Standardization. ue de Stassart, 36 B-1050 Brussels. (Cited on pages 6, 18, 41, 50, 54, XXXV).

FIB. (2013). *Fib model code for concrete structures* (1st edition). International Federation for Structural Concrete. (Cited on pages 15, 37, 45).

Gombosuren, D. & Maki, T. (2020). Prediction of joint shear deformation index of rc beam–column joints. *Buildings*, (176). https://doi.org/10.3390/buildings10100176 (cited on pages 10, 11)

Harish, A. (2020). Finite element method - what is it? fem and fea explained. *Simscale Blog* (cited on page 9).

Hendriks, M., de Boer, A. & Belletti, B. (2016). *Guidelines for nonlinear finite element analysis of concrete structures* (technical report Number 2) [Report RTD:1016-1:2016]. Rijkswaterstaat Ministry of Infrastructure and the Environment. (Cited on pages 10, 12, 15).

Herlin, A. (2011). *Göta älvbron, övervakning med optisk fiber* (Master's thesis). Uppsala universitet. Villavägen 16, 752 36 Uppsala. (Cited on page 22).

Honauer, U. & Ödeen, S. (2019). *Underhållsplan 2019–2022* (technical report) [ISBN: 978-91-7725-416-4]. Trafikverket. (Cited on page 2).

Lin, W. & Yoda, T. (2017). Bridge engineering, chapter fourteen - repair, strengthening, and replacement. Butterworth-Heinemann. https://doi.org/10.1016/B978-0-12-804432-2.00014-1. (Cited on page 2)

MathWorks. (2021a). *Bwmorph* [Accessed May 17, 2021]. https://se.mathworks.com/help/images/ref/bwmorph.html. (Cited on page 35)

MathWorks. (2021b). *Bwskel* [Accessed May 17, 2021]. https://se.mathworks.com/help/images/ref/bwskel.html?s_tid=doc_ta. (Cited on page 35)

Muanenda, Y., Oton, C. J. & Pasquale, F. D. (2019). Application of raman and brillouin scattering phenomena in distributed optical fiber sensing. *Frontiers in Physics*, *7*(155). https://doi.org/10.3389/fphy.2019.00155 (cited on page 21)

Ping, L., Mudabbir, B., Nageswara, L. & Bo, L. (2019). Distributed optical fiber sensing: Review and perspective. *Applied Physics Reviews*. https://doi.org//10.1063/1.5113955 (cited on pages 17, 20)

Plos, M. *Finite element analyses of reinforced concrete structures* [*Department of Structural Engineering, Concrete Structures*]. Compendium for the course *Concrete Structures - Advanced Course* at Chalmers Univserity of Technology. *Department of Structural Engineering, Concrete Structures*. Göteborg, 1996, April (cited on pages 9, 10, 12, 15).

Sohn, H. & Liu, P. (2016). Structural health monitoring (shm) in aerospace structures. Woodhead Publishing. https://doi.org/10.1016/B978-0-08-100148-6.00012-3. (Cited on page 17)

TNO DIANA BV. (2008). *Diana - finite element analysis* (1st edition) [Accessed 26 May, 2021]. https://dianafea.com/manuals/d93/Applic/node85.html. (Cited on page 56)

Trafikverket. (2020). *Skötsel av broar* [Accessed April 8, 2021]. https://www.trafikverket.se/resa-och-trafik/underhall-av-vag-och-jarnvag/Sa-skoter-vi-broar-och-tunnlar/Skotsel-av-broar/. (Cited on page 2)

United Nations. (2018). *68% of the world population projected to live in urban areas by 2050, says un* [Accessed April 7, 2021]. https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html. (Cited on page 2)

Vardanega, P., Webb, G., Fidler, P. & Middleton, C. (2015). Innovative bridge design handbook - construction, rehabilitation and maintenance. Butterworth Heinemann. http://dx.doi.org/10.1016/B978-0-12-800058-8.00029-3. (Cited on pages 17, 18, 20)

Webb, G., Vardanega, P. & Middleton, C. (2014). Categories of shm deployments: Technologies and capabilities. *Journal of Bridge Engineering*, *20*. https://doi.org/10.1061/(ASCE)BE.1943-5592.0000735 (cited on pages 18, 19)

Xin, L., Thomas, P. J. & Hellevang, J. O. (2019). A review of methods for fibre-optic distributed chemical sensing. *Sensors*. https://doi.org/10.3390/s19132876 (cited on page 20)

Zandi, K. (2020). Guidelines for nonlinear finite element analysis. (Cited on pages 12, 13).

A

Python script - MeshGenerator

The Python script which creates the geometry, reinforcement layout and mesh for the Diana model as well as assigning all relevant material models and parameters.

Z:\Exjobb\Beams\DOFS data\MeshGenerator.py 1

```
# Mesh Generator
# Part of Master thesis in structural engineering
# FE analyses of bending tests in DIANA
# Beam: PC1
# Authors: Marcus Jansén Svärd & Oskar Magnusson
# Date: 10.05.2021

##############################  IMPORT PYTHON MODULES  ##############################
import math as np
import collections
import time
import os

################################  DEFINE GEOMETRY  ################################

L = 3
h = 0.25
j = 0.8
D= 0.016
Ds = 0.008

b = h*j
name = 'PC1'
folder = "Z:/Exjobb/Beams/DOFS Data/" + name
newProject(folder + "/" + name + "/Models/" + name, 1000)

nbars = 3
nbarsp = 2
nstirrups = np.ceil(0.3*L/0.2)+1
if D < 0.016:
    Dp = D-0.002
else:
    Dp = 0.01

As = np.pi*D**2/4*nbars
Asp = np.pi*Dp**2/4*nbarsp
Ass = np.pi*8**2/4*2

us = np.pi*D*nbars
usp = np.pi*Dp*nbarsp
uss = np.pi*8*2

rho = As/(b*(h/2))

alf = 6.8
d = h-0.033-D/2 # 28
dp = 0.045+Dp/2 # 40 - 50
A = b/2
B = (alf-1)*Asp + alf*As
C = -((alf-1)*Asp*dp + alf*As*d)

x = (-B + np.sqrt(B**2-4*A*C))/(2*A)
Ifis = b*x**3/3 + (alf-1)*Asp*(x-dp)**2 + alf*As*(d-x)**2

Mu = 550000000*As*0.9*d
Fu = Mu/(0.3*L)
```

Z:\Exjobb\Beams\DOFS data\MeshGenerator.py 2

```
Delta_max = 0.6*Fu*0.3*L*(3*(0.9*L)**2 - 4*(0.3*L)**2)/(24*33000000000*Ifis)

print(x)
print(name)
print([nbars, nbarsp, D, As, rho*100])

newProject("Z:/Exjobb/" + name, 100)
setModelAnalysisAspects(["STRUCT"])
setModelDimension("2D")
setDefaultMeshOrder("LINEAR")
setDefaultMesherType("TETTRIA")
createSheet("Concrete",
             [[0, 0, 0],[L, 0, 0],[L, h, 0],[0, h, 0]])
createSheet("Left_support_plate",
             [[0.05*L-0.05, -0.065, 0],[0.05*L+0.05, -0.065, 0],
              [0.05*L+0.05, 0, 0],[0.05*L-0.05, 0, 0]])
createSheet("Right_support_plate",
             [[0.95*L-0.05, -0.065, 0],[0.95*L+0.05, -0.065, 0],
              [0.95*L+0.05, 0, 0],[0.95*L-0.05, 0, 0]])
createSheet("Left_wooden_sheet",
             [[0.35*L-0.05, h, 0],[0.35*L+0.05, h, 0],
              [0.35*L+0.05, h+0.01, 0],[0.35*L-0.05, h+0.01, 0]])
createSheet("Right_wooden_sheet",
             [[0.65*L-0.05, h, 0],[0.65*L+0.05, h, 0],
              [0.65*L+0.05, h+0.01, 0],[0.65*L-0.05, h+0.01, 0]])
createSheet("Left_load_plate",
             [[0.35*L-0.05, h+0.01, 0],[0.35*L+0.05, h+0.01, 0],
              [0.35*L+0.05, h+0.03, 0],[0.35*L-0.05, h+0.03, 0]])
createSheet("Right_load_plate",
             [[0.65*L-0.05, h+0.01, 0],[0.65*L+0.05, h+0.01, 0],
              [0.65*L+0.05, h+0.03, 0],[0.65*L-0.05, h+0.03, 0]])
createLine("Line 1", [0.35*L, h+0.1, 0], [0.65*L, h+0.1, 0])
createPointBody("point 1", [L/2, 2*h, 0])
projection("SHAPEEDGE", "Line 1", [[L/2, h+0.1, 0]], ["point 1"], [0, -1, 0], True)
removeShape(["point 1"])
createPointBody("point 2", [0.05*L, -0.2, 0])
createPointBody("point 3", [0.95*L, -0.2, 0])
createPointBody("point 4", [0.35*L, h*2, 0])
createPointBody("point 5", [0.65*L, h*2, 0])
projection("SHAPEEDGE", "Left_support_plate",
            [[0.05*L, -0.065, 0]], ["point 2"], [0, 1, 0], True)
removeShape(["point 2"])
projection("SHAPEEDGE", "Right_support_plate",
            [[0.95*L, -0.065, 0]], ["point 3"], [0, 1, 0], True)
removeShape(["point 3" ] )
projection("SHAPEEDGE", "Left_load_plate",
            [[0.35*L, h+0.03, 0]], ["point 4"], [0, -1, 0], True)
removeShape(["point 4" ] )
projection("SHAPEEDGE", "Right_load_plate",
            [[0.65*L, h+0.03, 0]], ["point 5"], [0, -1, 0], True)
removeShape(["point 5" ] )
createLine("Bottom_reinf",
            [0.02, h - d, 0], [L-0.02, 0.028+D/2, 0])
createLine("Top_reinf",
            [0.02, h - dp, 0], [L-0.02, h-0.028-Dp/2, 0])
```

```
for i in range(1, int(nstirrups) + 1):
    createLine("Stirrup_" + str(i),
        [L / 60 + (i - 1) * 0.2, h - d - D/2, 0],
        [L / 60 + (i - 1) * 0.2, h - dp + Dp/2, 0])
    createLine("Stirrup_" + str(2 * nstirrups - (i - 1)),
        [59 * L / 60 - (i - 1) * 0.2, h - d - D/2, 0],
        [59 * L / 60 - (i - 1) * 0.2, h - dp + Dp/2, 0])

################################  ADD SUPPORTS  ##########################
addSet("GEOMETRYSUPPORTSET", "Supports" )
createPointSupport("Support_left", "Supports" )
setParameter("GEOMETRYSUPPORT", "Support_left", "AXES", [1, 2 ])
setParameter("GEOMETRYSUPPORT", "Support_left", "TRANSL", [1, 1, 0])
setParameter("GEOMETRYSUPPORT", "Support_left", "ROTATI", [0, 0, 0])
attach("GEOMETRYSUPPORT", "Support_left", "Left_support_plate",
        [[0.05*L, -0.065, 0]])
createPointSupport( "Support_right", "Supports" )
setParameter("GEOMETRYSUPPORT", "Support_right", "AXES", [1, 2])
setParameter("GEOMETRYSUPPORT", "Support_right", "TRANSL", [0, 1, 0])
setParameter("GEOMETRYSUPPORT", "Support_right", "ROTATI", [0, 0, 0])
attach("GEOMETRYSUPPORT", "Support_right", "Right_support_plate",
        [[0.95*L, -0.065, 0]])
addSet("GEOMETRYSUPPORTSET", "Load_plate" )
createPointSupport("Support_1", "Load_plate" )
setParameter("GEOMETRYSUPPORT", "Support_1", "AXES", [1, 2])
setParameter("GEOMETRYSUPPORT", "Support_1", "TRANSL", [0, 1, 0])
setParameter("GEOMETRYSUPPORT", "Support_1", "ROTATI", [0, 0, 0])
attach("GEOMETRYSUPPORT", "Support_1", "Line 1",
       [[0.5*L, h+0.1, 0]])

################################  ADD TYINGS  ###############################
addSet("GEOMETRYTYINGSET", "Support_plates")
createPointTying("Left_tying", "Support_plates")
setParameter("GEOMETRYTYING", "Left_tying", "AXES", [1, 2])
setParameter("GEOMETRYTYING", "Left_tying", "TRANSL", [0, 1, 0])
setParameter("GEOMETRYTYING", "Left_tying", "ROTATI", [0, 0, 0])
attachTo("GEOMETRYTYING", "Left_tying", "SLAVE", "Left_load_plate",
         [[0.35*L, h+0.03, 0]])
attachTo("GEOMETRYTYING", "Left_tying", "MASTER", "Line 1",
         [[0.35*L, h+0.1, 0]])
createPointTying("Right_tying", "Support_plates")
setParameter("GEOMETRYTYING", "Right_tying", "AXES", [1, 2 ])
setParameter("GEOMETRYTYING", "Right_tying", "TRANSL", [0, 1, 0])
setParameter("GEOMETRYTYING", "Right_tying", "ROTATI", [0, 0, 0])
attachTo("GEOMETRYTYING", "Right_tying", "SLAVE", "Right_load_plate",
         [[0.65*L, h+0.03, 0]])
attachTo("GEOMETRYTYING", "Right_tying", "MASTER", "Line 1",
         [[0.65*L, h+0.1, 0]])

############################  MATERIAL DEFINITIONS  ##########################
## Concrete material at 507 days
#addMaterial("Concrete", "CONCR", "TSCR", [])
#setParameter(MATERIAL, "Concrete", "LINEAR/ELASTI/YOUNG", 3.3783e+10)
#setParameter(MATERIAL, "Concrete", "LINEAR/ELASTI/POISON", 0.15)
#setParameter(MATERIAL, "Concrete", "LINEAR/MASS/DENSIT", 2300)
#setParameter(MATERIAL, "Concrete", "MODTYP/TOTCRK", "ROTATE")
```

Z:\Exjobb\Beams\DOFS data\MeshGenerator.py 4

```
#setParameter(MATERIAL, "Concrete", "TENSIL/TENSTR", 4592000)
#setParameter(MATERIAL, "Concrete", "TENSIL/EPSULT", 0.02875)
#setParameter(MATERIAL, "Concrete", "COMPRS/COMCRV", "PARABO")
#setParameter(MATERIAL, "Concrete", "TENSIL/TENCRV", "HORDYK")
#setParameter(MATERIAL, "Concrete", "TENSIL/GF1", 100)
#setParameter(MATERIAL, "Concrete", "TENSIL/POISRE/POIRED", "DAMAGE")
#setParameter(MATERIAL, "Concrete", "TENSIL/POISRE/POIRED", "NONE")
#setParameter(MATERIAL, "Concrete", "TENSIL/CBSPEC", "USER")
#setParameter(MATERIAL, "Concrete", "TENSIL/CRACKB", 0.010)
#setParameter(MATERIAL, "Concrete", "COMPRS/COMSTR", 75013000)
#setParameter(MATERIAL, "Concrete", "COMPRS/GC", 25000)

## Concrete material at 14 days
addMaterial("Concrete", "CONCR", "TSCR", [])
setParameter(MATERIAL, "Concrete", "LINEAR/ELASTI/YOUNG", 3.6414e+10)
setParameter(MATERIAL, "Concrete", "LINEAR/ELASTI/POISON", 0.15)
setParameter(MATERIAL, "Concrete", "LINEAR/MASS/DENSIT", 2300)
setParameter(MATERIAL, "Concrete", "MODTYP/TOTCRK", "ROTATE")
setParameter(MATERIAL, "Concrete", "TENSIL/TENSTR", 3000000)
setParameter(MATERIAL, "Concrete", "TENSIL/EPSULT", 0.003892)
setParameter(MATERIAL, "Concrete", "COMPRS/COMCRV", "PARABO")
setParameter(MATERIAL, "Concrete", "TENSIL/TENCRV", "HORDYK")
setParameter(MATERIAL, "Concrete", "TENSIL/GF1", 100)
setParameter(MATERIAL, "Concrete", "TENSIL/POISRE/POIRED", "DAMAGE")
setParameter(MATERIAL, "Concrete", "TENSIL/POISRE/POIRED", "NONE")
setParameter(MATERIAL, "Concrete", "TENSIL/CBSPEC", "USER")
setParameter(MATERIAL, "Concrete", "TENSIL/CRACKB", 0.010)
setParameter(MATERIAL, "Concrete", "COMPRS/COMSTR", 53152000)
setParameter(MATERIAL, "Concrete", "COMPRS/GC", 25000)


addGeometry("Beam", "SHEET", "MEMBRA", [])
setParameter("GEOMET", "Beam", "THICK", b)
setElementClassType("SHAPE", ["Concrete"], "MEMBRA")
assignMaterial("Concrete", "SHAPE", ["Concrete"])
assignGeometry("Beam", "SHAPE", ["Concrete"])

## Reinforcement steel Material
addMaterial("ReinforcementSteel", "REINFO", "REBOND", [])
setParameter("MATERIAL", "ReinforcementSteel", "REBARS/ELASTI/YOUNG",
             1.984e+11)
setParameter("MATERIAL", "ReinforcementSteel","REBARS/POISON/POISON",
             0.3)
setParameter("MATERIAL", "ReinforcementSteel", "REBARS/MASS/DENSIT",
             7850)
setParameter("MATERIAL", "ReinforcementSteel", "RESLIP/DSNY",
             1e+14)
setParameter("MATERIAL", "ReinforcementSteel", "RESLIP/DSSX",
             1e+11)
setParameter("MATERIAL","ReinforcementSteel","RESLIP/SHFTYP","BONDS6")
setParameter("MATERIAL", "ReinforcementSteel", "RESLIP/BONDS6/SLPVAL",
              [20000000, 8000000, 1e-05, 0.001, 0.002, 0.008, 0.4])
setParameter("MATERIAL", "ReinforcementSteel", "REBARS/PLATYP", "VMISES")
setParameter(MATERIAL, "ReinforcementSteel", "REBARS/PLASTI/TRESSH", "KAPSIG")
setParameter(MATERIAL, "ReinforcementSteel", "REBARS/PLASTI/KAPSIG",
             [0, 5.274e+8, 0.107123, 6.491e+8])
```

```
addGeometry("Bottom_reinf", "RELINE", "REBAR", [])
setParameter("GEOMET", "Bottom_reinf", "REITYP", "REITRU")
setParameter("GEOMET", "Bottom_reinf", "REITRU/CROSSE", As)
setParameter("GEOMET", "Bottom_reinf", "REITRU/PERIME", us)
addElementData("Bottom_reinf")
setShapeType("REINFORCEMENTSHAPE", "Bottom_reinf")
setReinforcementType("REINFORCEMENTSHAPE", "Bottom_reinf",
                     "TRUSS_BOND_SLIP")
assignMaterial("ReinforcementSteel", "REINFORCEMENTSHAPE",
               ["Bottom_reinf"])
assignGeometry("Bottom_reinf", "REINFORCEMENTSHAPE",
               ["Bottom_reinf"])
assignElementData("Bottom_reinf", "REINFORCEMENTSHAPE",
                  ["Bottom_reinf"])
setParameter("DATA", "Bottom_reinf", "INTERF", "TRUSS")

addGeometry("Top_reinf", "RELINE", "REBAR", [])
setParameter("GEOMET", "Top_reinf", "REITYP", "REITRU")
setParameter("GEOMET", "Top_reinf", "REITRU/CROSSE", Asp)
setParameter("GEOMET", "Top_reinf", "REITRU/PERIME", usp)
addElementData("Top_reinf")
setShapeType("REINFORCEMENTSHAPE", "Top_reinf")
setReinforcementType("REINFORCEMENTSHAPE", "Top_reinf",
                     "TRUSS_BOND_SLIP")
assignMaterial("ReinforcementSteel", "REINFORCEMENTSHAPE",
               ["Top_reinf"])
assignGeometry("Top_reinf", "REINFORCEMENTSHAPE", ["Top_reinf"])
assignElementData("Top_reinf", "REINFORCEMENTSHAPE", ["Top_reinf"])
setParameter("DATA", "Top_reinf", "INTERF", "TRUSS")

addGeometry("Stirrups", "RELINE", "REBAR", [])
setParameter("GEOMET", "Stirrups", "REITYP", "REITRU")
setParameter("GEOMET", "Stirrups", "REITRU/CROSSE", Ass)
setParameter("GEOMET", "Stirrups", "REITRU/PERIME", uss)
addElementData("Stirrups")

for i in range(1,int(nstirrups)*2+1):
    setShapeType("REINFORCEMENTSHAPE", "Stirrup_" + str(i))
    setReinforcementType("REINFORCEMENTSHAPE", "Stirrup_" + str(i),
                         "TRUSS_BOND_SLIP")
    assignMaterial("ReinforcementSteel", "REINFORCEMENTSHAPE",
                   ["Stirrup_" + str(i)])
    assignGeometry("Stirrups", "REINFORCEMENTSHAPE",
                   ["Stirrup_" + str(i)])
    assignElementData("Stirrups", "REINFORCEMENTSHAPE",
                      ["Stirrup_" + str(i)])

setParameter("DATA", "Stirrups", "INTERF", "TRUSS")

## Steel plates
addMaterial("Steel", "MCSTEL", "ISOTRO", [])
setParameter("MATERIAL", "Steel", "LINEAR/ELASTI/YOUNG", 2e+11)
setParameter("MATERIAL", "Steel", "LINEAR/ELASTI/POISON", 0.3)
setParameter("MATERIAL", "Steel", "LINEAR/MASS/DENSIT", 7850)
addGeometry("Plate thickness", "SHEET", "MEMBRA", [])
```

```
setParameter("GEOMET", "Plate thickness", "THICK", b)
setElementClassType("SHAPE", ["Left_load_plate", "Left_support_plate",
            "Right_load_plate", "Right_support_plate"], "MEMBRA")
assignMaterial("Steel", "SHAPE", ["Left_load_plate",
            "Left_support_plate", "Right_load_plate",
            "Right_support_plate"])
assignGeometry("Plate thickness", "SHAPE", ["Left_load_plate",
            "Left_support_plate", "Right_load_plate",
            "Right_support_plate"])

## Wooden sheets
addMaterial("Wood", "MCSTEL", "ISOTRO", [])
setParameter("MATERIAL", "Wood", "LINEAR/ELASTI/YOUNG", 2e+10)
setParameter("MATERIAL", "Wood", "LINEAR/ELASTI/POISON", 0.3)
setParameter("MATERIAL", "Wood", "LINEAR/MASS/DENSIT", 900)
setElementClassType("SHAPE", ["Left_wooden_sheet", "Right_wooden_sheet"],
                    "MEMBRA")
assignMaterial("Wood", "SHAPE", ["Left_wooden_sheet",
                    "Right_wooden_sheet"])
assignGeometry("Plate thickness", "SHAPE",
               ["Left_wooden_sheet", "Right_wooden_sheet" ])

## Dummy beam
addMaterial("Dummy Material", "MCSTEL", "ISOTRO", [])
setParameter("MATERIAL", "Dummy Material", "LINEAR/ELASTI/YOUNG", 1e+12)
setParameter("MATERIAL", "Dummy Material", "LINEAR/ELASTI/POISON", 0.2)
setParameter("MATERIAL", "Dummy Material", "LINEAR/MASS/DENSIT", 1)
addGeometry("Dummy Geometry", "LINE", "CLS2B2", [])
setParameter("GEOMET", "Dummy Geometry", "SHAPE/RECTAN", [0.5, 0.2])
setElementClassType("SHAPE", [ "Line 1" ], "CLS2B2")
assignMaterial("Dummy Material", "SHAPE", ["Line 1"])
assignGeometry("Dummy Geometry", "SHAPE", ["Line 1"])

################################  LOAD DEFINITIONS  ################################
addSet("GEOMETRYLOADSET", "Displacement")
createPointLoad("Disp", "Displacement")
setParameter("GEOMETRYLOAD", "Disp", "LODTYP", "DEFORM")
setParameter("GEOMETRYLOAD", "Disp", "DEFORM/SUPP", "Support_1")
setParameter("GEOMETRYLOAD", "Disp", "DEFORM/TR/VALUE", -0.001)
setParameter("GEOMETRYLOAD", "Disp", "DEFORM/TR/DIRECT", 2)
attach("GEOMETRYLOAD", "Disp", "Line 1", [[ L/2, h+0.1, 0 ]])

#################################### MESHING ####################################
elementSetString = ["Concrete", "Top_reinf", "Left_support_plate",
                "Bottom_reinf", "Right_support_plate", "Right_load_plate",
                "Line 1", "Right_wooden_sheet", "Left_wooden_sheet",
                "Left_load_plate"]
for i in range(1,int(nstirrups)*2+1):
    elementSetString.append('Stirrup_'+str(i))
setElementSize(elementSetString, 0.01, -1, True)
setMesherType(elementSetString, "HEXQUAD")

generateMesh([])

setColor(ELEMENTSET, "#3d85c6", ["Left_support_plate", "Right_support_plate"])
setColor(ELEMENTSET, "#9fc5e8", ["Left_load_plate", "Right_load_plate"])
```

Z:\Exjobb\Beams\DOFS data\MeshGenerator.py 7

```
setColor(ELEMENTSET, "#ff9900", ["Right_wooden_sheet", "Left_wooden_sheet"])
setColor(ELEMENTSET, "#9fc5e8", ["Line 1"])
exportModel(folder + "/Models/" + name + ".dat", 5)
```

B

Matlab script - CrackTranslator

The Matlab script which was used to find, allocate and quantify the number and magnitude of cracks. The part which was originally written by Berrocal, C. denoted as well as the part which was written by the authors.

Table of Contents


```
clear all
close all
clc
```

Load DOFS data and translate into cracks

```
% Author: Carlos Gil Berrocal

% Load DOFS data from 4-point bending test.
load beam1_filtered_data_modified.mat

tt = 1;

% Allocate the specific levels of the DOFS strains
bot.strain = mid_ip(tt,:)*1.25;
bar1.strain = bar1_ip(tt,:);
mid.strain = mid_ip(tt,:);
top.strain = top_ip(tt,:);

[~,~,bar1f] = fitSpline(x,bar1.strain,0.999999);
[~,~,midf] = fitSpline(x,mid.strain,0.999999);

mid.crack = strain2crack_modified(x,midf);
bot.crack = strain2crack_modified(x,bar1f*1.25);
bar1.crack = strain2crack_modified(x,bar1f);
top.crack = top.strain*0;

%Fill the parts outside the supports with zeros
strain = [bot.strain; bar1.strain; mid.strain; top.strain];
strain = [zeros(4,15) strain zeros(4,15)]';
S = strain(:);

crack = [bot.crack; bar1.crack; mid.crack; top.crack];
crack = [zeros(4,15) crack zeros(4,15)]';
C = crack(:);
```

```
%Specify location of nodes where data is measured
meshsize = 0.01;
s_x = 0:meshsize:3;
s_y = [0 0.033 0.125 0.18];
[S_X,S_Y] = meshgrid(s_x,s_y);
S_X = S_X'; S_X = S_X(:);
S_Y = S_Y'; S_Y = S_Y(:);

%Create interpolant function
Fstrain = scatteredInterpolant(S_X, S_Y, S);
Fcrack = scatteredInterpolant(S_X, S_Y, C);

%Create a uniform mesh grid of nodes
z=0:0.01:3;
y=0:0.01:0.25;
[Z,Y] = meshgrid(z,y);

%Interpolate
fstrain = Fstrain(Z,Y);
fcrack=Fcrack(Z,Y);
w_max = 0.2935;
fcrack = fcrack * w_max/max(max(fcrack));
```

Find and allocate the nodal coordinates and crack widths for each invidvidual crack

```
% Part of master thesis in structural engineering
% Calculation of weakened element properties from DOFS data
%
% Beam: PC1
% Authors: Marcus Jansen Svard, Oskar Magnusson
%
% Date: May 10th, 2021

% Set threshold value for what to consider not as cracked
tol = 1e-3;

% Remove all negligible crack widths
fcrack(fcrack < tol) = 0;
fcrackUI = fcrack;

% Extract matrix positions for each nodal crack
[row, col] = find(fcrack);

% Save uninterpolated values for scatter-plot
[rowI, colI] = find(fcrack);
```

Find voids in inclined cracks and interpolate

```
k = 0;
voidCrack =[0,0];
```

```
% This loop covers inclined cracks with only one intermediate void
 element
for i = 1:max(row)
    for j = 1:max(col)
        if fcrack(i,j) > 0
            if (fcrack(i,j+1) == 0 && fcrack(i,j+2) > 0)
                k = k + 1;
                voidCrack(k, 1:2) = [i, j+1];
                if ( fcrack(i,j+2) > fcrack(i,j) )
                    fcrack(i,j+1) = fcrack(i,j)+(fcrack(i,j+2)-
fcrack(i,j))/2;
                else
                    fcrack(i,j+1) = fcrack(i,j+2) + (fcrack(i,j) -
 fcrack(i,j+2))/2;
                end
            elseif (fcrack(i,j-1) == 0 && fcrack(i,j-2) > 0)
                k = k + 1;
                voidCrack(k, 1:2) = [i, j-1];
                if ( frack(i,j-2) > fcrack(i,j) )
                    fcrack(i,j-1) = fcrack(i,j)+(fcrack(i,j-2)-
fcrack(i,j))/2;
                else
                    fcrack(i,j-1) = fcrack(i,j-2) + (fcrack(i,j) -
 fcrack(i,j-2))/2;
                end
            end
        end
    end
end

% Choose row and column indices to not interpolate (different
 dependant on
% case)
iNot1 = [5, 6, 7, 8, 9, 10];
jNot1 = [96, 97];

for j = 1:length(jNot1)
    for i = 1:length(iNot1)
        fcrack(iNot1(i),jNot1(j)) = 0;
    end
end

% Save matrix for plotting purposes
fcrackI = fcrack;
```

Perform morphological operations to remove "thick" cracks

```
% Save unmorphed values for scatter-plot
[rowM, colM] = find(fcrack);
XM = meshsize * (colM - 1) - tol;
```

```
XMc = meshsize * (voidCrack(:,2) - 1);

YM = meshsize * (rowM - 1) + tol;
YMc = meshsize * (voidCrack(:,1) - 1);

for i = 1:length(rowM)
    WM(i,1) = fcrack(rowM(i), colM(i));
end

% Perform the morph of cracks
I = fcrack;
K = bwskel(logical(I));
fcrackM = K.*fcrack; % Save matrix for plotting purposes
J = bwmorph(bwskel(logical(I)),'diag');
fcrack = J.*fcrack;

% Save semi-morphed values for scatter-plot
[rowM2, colM2] = find(fcrackM);
XM2 = meshsize * (colM2 - 1) - tol;
YM2 = meshsize * (rowM2 - 1) + tol;

for i = 1:length(rowM2)
    WM2(i,1) = fcrackM(rowM2(i), colM2(i));
end
```

Readjust crack widths for inclined cracks

```
% Extract matrix positions with new interpolated values
[row, col] = find(fcrack);

% Choose range of sensitivity
range = -2:2; % j

% For straight parts of the crack
for i = 1:max(row)
    for j = 1:max(col)
        if fcrack(i,j) > 0
            if (fcrack(i,j+1) == 0 && fcrack(i,j-1) == 0)
                fcrack(i,j) = sum(fcrackUI(i,j-range));
            end
        end
    end
end

% For inclined parts of the crack
for i = 1:max(row)
    for j = 1:max(col)
        if (fcrack(i,j) > 0 && fcrack(i,j+1) > 0)
            fcrack(i,j) = sum(fcrackUI(i,j-range));
            fcrack(i,j+1) = fcrack(i,j);
        end
    end
end
```

Translate row and column positions into x- and y-coordinates

```
% Extract matrix positions with new interpolated values
[row, col] = find(fcrack);

X = meshsize * (col - 1) - tol;
XI = meshsize * (colI - 1);       % for uninterpolated scatter-plot
Xc = meshsize * (voidCrack(:,2) - 1);

Y = meshsize * (row - 1) + tol;
YI = meshsize * (rowI - 1) + tol; % For uninterpolated scatter-plot
Yc = meshsize * (voidCrack(:,1) - 1) + tol;

% Extract all nodal crack widths from the fcrack matrix
for i = 1:length(row)
    W(i,1) = fcrack(row(i), col(i));
end

for i = 1:length(rowI)
    WI(i,1) = fcrack(rowI(i), colI(i)); % Save for plotting purposes
end

% Create a Z-coordinate for the nodes
Z = zeros(length(X),1);

% Assemble into global matrix
filteredCrack(:,1) = X;
filteredCrack(:,2) = Y;
filteredCrack(:,3) = Z;
filteredCrack(:,4) = W;

% Vectors to plot the mesh grid used
% z=z-tol;
% y=y+tol;
```

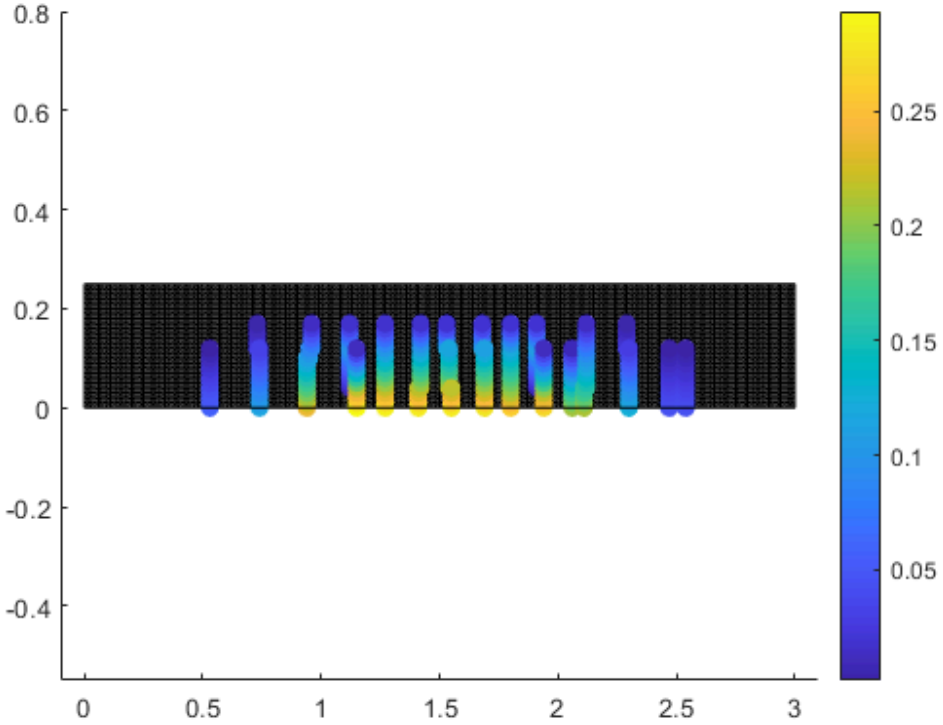
Plot for confirmation of crack positions

```
figure(2)
pointsize = 25;
hold on
for i = 2:length(y)
    plot([z(1), z(end)], [y(i), y(i)], 'k')
end

for i = 2:length(z)
    plot([z(i), z(i)], [y(1), y(end)], 'k')
end

scatter(X+tol, Y-tol, pointsize*2, W, 'filled')
cbar = colorbar;
```

```
rectangle('Position', [0 0 3.000 0.250])
axis([-0.100 3.100 -0.550 0.800])
```



Calculate weakened element properties

```
% fcm = 72;              % Mean compressive strength, 507 days [MPa]
fcm = 53.152;            % Mean compressive strength, 14  days [MPa]
% fctm = 4.592;          % Tensile strength, 507 days          [MPa]
fctm = 3.0;              % Tensile strength, 14  days          [MPa]
% Gf = 100;              % Tensile fracture energy, 507 days   [N/m]
Gf = 100;                % Tensile fracture energy, 14  days   [N/m]
stress_limit = 0.01;     %                           [MPa]
strain_limit = 1e-4;     %                           [-]
crack_bandwidth = 10;    % Crack bandwidth           [mm]
alpha_s = 0.25;
beta_s = 0.15;
sigma_s = alpha_s*fctm;
w_c = 5*Gf/(fctm*1000);  % w_ult
w_s = beta_s * w_c;

steps1 = linspace(0,w_s);
steps2 = linspace(w_s,w_c);

% Plot tension softening curve
figure(8)
title('Crack positions on tension softening curve')
```

```
hold on
plot(steps1,fctm - (fctm - sigma_s)/w_s*steps1);
plot(steps2,sigma_s - (sigma_s-0)/(w_c - w_s)*(steps2 - steps1(end)));
grid minor
xlabel('Crack width, w [mm]')
ylabel('Residual tensile strength, f_t [MPa]')
axis([0 w_max*1.1 0 fctm*1.1])

crackNames = cell(length(W),2);

for i = 1:length(W)
    crackNames{i,1} = string(i);
    crackNames{i,2} = W(i);
end

% Pre-allocate to calculate amount of cracks in each region of the
 curve
% and calculation of relative weakening
r1 = 0;
r2 = 0;
r3 = 0;
output = zeros(length(crackNames),2);
K = zeros(length(crackNames),1);
L = zeros(length(crackNames),1);

crackStrengthInput = cell(length(crackNames),1);
for i=1:length(crackNames)
    crack_width = crackNames{i,2};
    if crack_width <= w_s
        r1 = r1 + 1;
        stress = fctm-(fctm - sigma_s)/w_s*crack_width;
        ult_strain =(w_c - crack_width)/crack_bandwidth;
    elseif crack_width > w_s && crack_width < w_c
        r2 = r2 + 1;
        stress = sigma_s - (sigma_s-0)/(w_c-w_s)*(crack_width-
steps1(end));
        ult_strain =(w_c - crack_width)/crack_bandwidth;
    else
        r3 = r3 + 1;
        stress = stress_limit;
        ult_strain = strain_limit;
    end

    scatter(crack_width,stress)
    crackStrengthInput{i} = {crackNames{i,1},[stress*1e6,ult_strain]};

    % Store values to calculate relative weakening
    output(i,1:2) = crackStrengthInput{i,1}{1,2};
    K(i) = output(i,1);
    L(i) = (fctm * 1e6 - K(i))/(fctm * 1e6);
end

% Calculate relative weakening
RelW = mean(L);
```

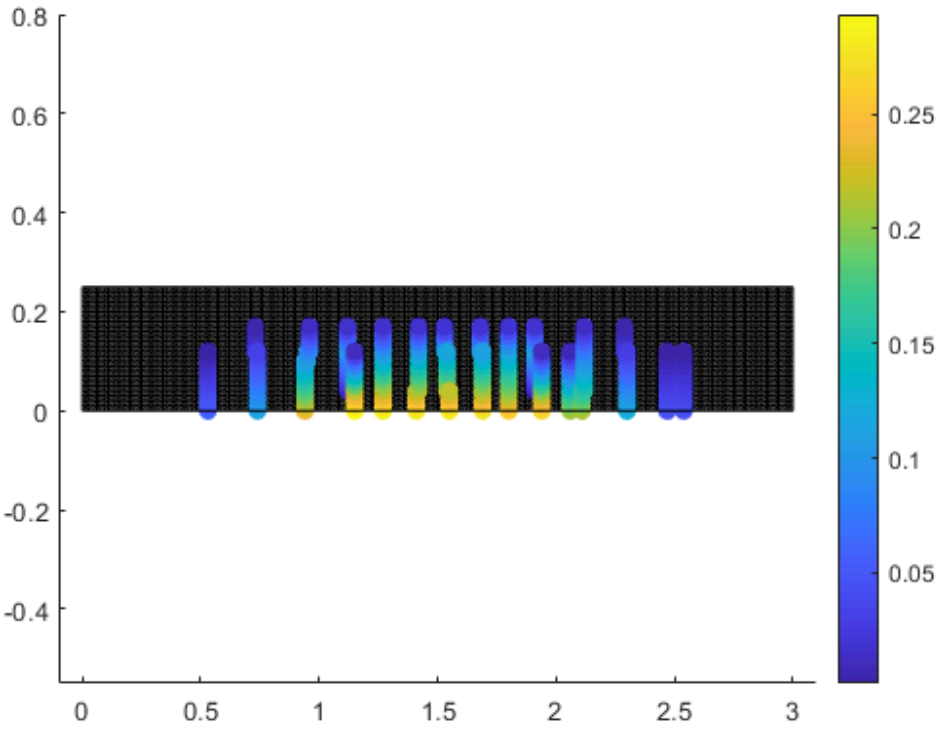
```
% Plot table
column1 = [r1, r2, r3, RelW*100];
column2 = ['#', '#', '#', '%'];
label = {'Region 1', 'Region 2', 'Fully opened', 'Relative
 weakening'};
TensionSoftening = table(column1', column2', 'VariableNames',
 {'Value' 'Unit'}, 'rownames', label)
```

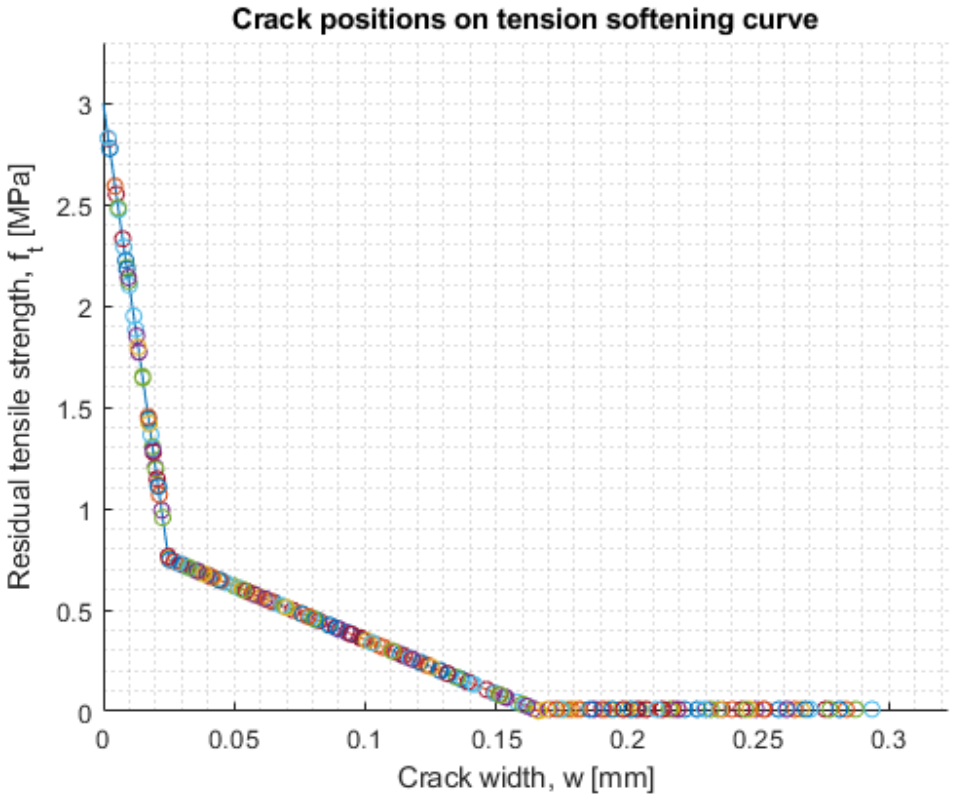
```
TensionSoftening =

  4×2 table

                         Value      Unit
                         _____      ____

    Region 1                44       #
    Region 2               165       #
    Fully opened            68       #
    Relative weakening    82.141     %
```





Write reduced strength and corresponding co-ordinates to .csv

```
output=[];
OP = zeros(length(crackStrengthInput),2);
for i=1:length(crackStrengthInput)
    output=[];
    output = [output;crackStrengthInput{i}{2}];
    OP(i,:) = output;
end

% Weakened element properties
csvwrite('CrackStrengthInput.csv',OP) % CSI-matrix

% Nodes
nodes = filteredCrack(:,1:3); % N-matrix
csvwrite('Nodes.csv', nodes);
```

Published with MATLAB® R2019b

B.1 Matlab script - strain2crack_modifed

Script required to run the Crack Translator, written by Berrocal, C.

```
function [ynorm, peaks, locs] = strain2crack_modified(pos, eps)

cutoff = 90;
k = 1/9000;
xx = pos;
yy = eps;

[~, locs] = findpeaks(yy,xx,'MinPeakHeight',cutoff);

% figure(1)
% subplot(5,1,1)
% hold on
% findpeaks(yy,xx/1000,'MinPeakHeight',cutoff);
% ax=gca;
% axPos = ax.Position;
% ax.Position([1 3]) = [0.05, 0.85];
% box on
% grid off
% title('fibre optic strain [microstrains]')
%
%
dlocs = diff(locs); ind_locs = find(dlocs < 5);

if isempty(ind_locs) == 0
    for j = numel(ind_locs):-1:1
        i = ind_locs(j);
        locs(i) = mean(locs(i:i+1));
        locs(i+1) = [];
    end
end


ynorm = zeros(size(xx));

for i=1:numel(locs)
    dist = 1/max(normpdf(xx,locs(i),10))*normpdf(xx,locs(i),1);
    ynorm = ynorm + dist;
end
ynorm = yy'.*ynorm*k;

[peaks,locs] =
 findpeaks(ynorm,'MinPeakHeight',cutoff*k,'MinPeakDistance',0.006);

% subplot(5,1,2)
% hold on
% plot((xx+150)/1000,ynorm)
% plot((xx+150)/1000,yy*k,'.')
% ax=gca;
% axPos = ax.Position;
% ax.Position([1 3]) = [0.05, 0.9];
```

Published with MATLAB® R2019b

B.2 Matlab script - fitSpline

Script required to run the Crack Translator, written by Berrocal, C.

```
function [fitresult, gof, curve] = fitSpline(pos, eps, SmoothingParam)

if nargin < 3
    SmoothingParam = 0.999999;
end

[xData, yData] = prepareCurveData( pos, eps );

% Set up fittype and options.
ft = fittype( 'smoothingspline' );
opts = fitoptions( 'Method', 'SmoothingSpline' );
% opts.SmoothingParam = 0.99999;
opts.Normalize = 'on';
opts.SmoothingParam = SmoothingParam;

% Fit model to data.
[fitresult, gof] = fit( xData, yData, ft, opts );

% Plot fit with data.
% figure( 'Name', 'untitled fit 1' );
% h = plot( fitresult, xData, yData );
% legend( h, 'x vs. pos', 'fit curve', 'Location', 'NorthEast' );
% % Label axes
% xlabel pos
% ylabel x
% grid on

curve = feval(fitresult,pos);
```

Published with MATLAB® R2019b

C

Python script - CrackGenerator

The Python script which loads the weakened element properties produced by the *CrackTranslator* in appendix B and assigns them to the corresponding elements in the FE mesh created by the *Mesh Generator* in appendix A.

Z:\Exjobb\Beams\DOFS data\CrackGenerator.py 1

```
# Crack Generator
# Part of Master thesis in structural engineering
# FE analyses of bending tests in DIANA
# Beam: PC1
# Authors: Marcus Jansén Svärd & Oskar Magnusson
# Date: 10.05.2021

##############################  IMPORT PYTHON MODULES  ##############################
import csv

################################  NAME FOR IMPORT  ################################
name = 'PC1'
folder = "Z:/Exjobb/Beams/DOFS data/" + name

importModel(folder + '/Models/' + name + ".dat")

w = 0.200
addGeometry("Concrete_weakened", "SHEET", "MEMBRA", [])
setParameter("GEOMET", "Concrete_weakened", "THICK", w)
addElementData("Concrete_weakened")

nodes = "Z:/Exjobb/Beams/DOFS data/" + name + "/Nodes.csv"
crackStrengthInputFile = "Z:/Exjobb/Beams/DOFS Data/" + name + "/
  CrackStrengthInput.csv"
with open(nodes, 'r') as f:
    data = list(csv.reader(f, delimiter=','))
    for key in range(1, len(data) + 1):
        elemList = []
        print(key)

        with open(crackStrengthInputFile, 'r') as f:
            reader = csv.reader(f, delimiter=',', quoting=csv.QUOTE_NONNUMERIC)
            crackStrengthInput = list(reader)

        crackStrengthInput = crackStrengthInput[key-1]
        copy(MATERIAL, "Concrete", "Cracked " + str(key))
        setParameter(MATERIAL, "Cracked " + str(key), "MODTYP/TOTCRK", "FIXED")
        setParameter(MATERIAL, "Cracked " + str(key), "TENSIL/TENCRV", "LINEPS")
        setParameter(MATERIAL, "Cracked " + str(key), "TENSIL/TENSTR",
          crackStrengthInput[0])
        setParameter(MATERIAL, "Cracked " + str(key), "TENSIL/EPSULT",
          crackStrengthInput[1])
        setParameter(MATERIAL, "Cracked " + str(key), "TENSIL/POISRE/POIRED",
          "NONE")
        setParameter(MATERIAL, "Cracked " + str(key), "SHEAR/SHRCRV", "CONSTA")
        setParameter(MATERIAL, "Cracked " + str(key), "SHEAR/BETA", 0.01)

        addSet(ELEMENTSET, "Element set 1")
        rename(ELEMENTSET, "Element set 1", "Cracked " + str(key))
        setColor(ELEMENT, "#ff0000", ["Cracked " + str(key)])

        with open(nodes, 'r') as f1:
            reader = csv.reader(f1, delimiter=',', quoting=csv.QUOTE_NONNUMERIC)
            crackCoordinates = list(reader)
        elemList.append(crackCoordinates[key - 1])
        select(ELEMENT, findNearestElements(elemList))
```

```
        print(findNearestElements(elemList))

        deselectByMaterial(ELEMENT, ["Steel", "Wood"])
        moveToElementSet("Cracked " + str(key), selectedElementIds())
        assignMaterialToElements("Cracked " + str(key), ["Cracked " + str(key)])
        assignGeometry("Concrete_weakened", ELEMENTSET, ["Cracked " + str(key)])
        assignElementData("Concrete_weakened", ELEMENTSET, ["Cracked " + str(key)])
        clearSelection(ELEMENT)

setColor( ELEMENTSET, "#3d85c6", [ "Left_support_plate","Right_support_plate" ] )
setColor( ELEMENTSET, "#9fc5e8", [ "Left_load_plate" ,"Right_load_plate" ] )
setColor( ELEMENTSET, "#ff9900", [ "Right_wooden_sheet", "Left_wooden_sheet" ] )
setColor( ELEMENTSET, "#9fc5e8", [ "Line 1" ] )

################################  NAME FOR EXPORT  ################################
name2 = name + 'Cracked'
exportModel(folder + "/Models/" + name2 + ".dat", 5)
```

D

Python script - AnalysisGenerator

The Python script which adds the FE analysis to either the FEC or FET model created by the *Mesh* and *Crack Generator* in appendices A and C, repesctively.

Z:\Exjobb\Beams\DOFS data\AnalysisGenerator.py 1

```
# Analysis Generator
# Part of Master thesis in structural engineering
# FE analyses of bending tests in DIANA
# Beam: PC1
# Authors: Marcus Jansén Svärd & Oskar Magnusson
# Date: 10.05.2021

##############################  IMPORT PYTHON MODULES  ##############################
import math as np
import csv

################################  NAME FOR IMPORT  ################################
name = 'PC1'
folder = "Z:/Exjobb/Beams/DOFS data/" + name

## Choose which model to analyse - with or without crack inclusion
importModel(folder + '/Models/' + name + "Cracked" + ".dat") # FEC model
#importModel(folder + '/Models/' + name + ".dat")            # FET model

nodes = "Z:/Exjobb/Beams/DOFS data/" + name + "/Nodes.csv"
crackStrengthInputFile = "Z:/Exjobb/Beams/DOFS Data/" + name + "/CrackStrengthInput.csv"
with open(nodes, 'r') as f:
    data = list(csv.reader(f, delimiter=','))
    for key in range(1, len(data) + 1):
        select(ELEMENT, "Cracked " + str(key))
        setColor(ELEMENT, "#ff0000", ["Cracked " + str(key)])

setColor( ELEMENTSET, "#3d85c6", [ "Left_support_plate","Right_support_plate" ] )
setColor( ELEMENTSET, "#9fc5e8", [ "Left_load_plate" ,"Right_load_plate" ] )
setColor( ELEMENTSET, "#ff9900", [ "Right_wooden_sheet", "Left_wooden_sheet" ] )
setColor( ELEMENTSET, "#9fc5e8", [ "Line 1" ] )


################################  ADD ANALYSES  ################################
max_steps = np.floor(Delta_max / 0.0001)
ramp_upp_1 = np.floor(max_steps * 0.5)
ramp_upp_2 = np.floor(max_steps * 0.7)
addAnalysis("Analysis1")
addAnalysisCommand("Analysis1", "NONLIN", "Structural nonlinear")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "EXECUT(1)/LOAD/STEPS/EXPLIC/SIZES", "0.1(800)")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "EXECUT(1)/ITERAT/MAXITE", 250)
addAnalysisCommandDetail("Analysis1", "Structural nonlinear", "EXECUT(1)/ITERAT/CONVER/ENERGY")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "EXECUT(1)/ITERAT/CONVER/ENERGY", True)

dispNode = findNearestNodes([[L / 2, 0, 0]])
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/FILE", name + "_Disp")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/DEVICE", "TABULA")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/LAYOUT/LINPAG", 1000000)
```

Z:\Exjobb\Beams\DOFS data\AnalysisGenerator.py 2

```
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/LAYOUT/
  COLLIN", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/SELECT/
  MODSEL", "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/SELECT/
  NODES(1)/RNGNRS", dispNode)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/SELECT/
  ELEMEN(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/SELECT/
  REINFO(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/SELTYP",
  "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)/USER/
  DISPLA(1)/TOTAL/TRANSL/GLOBAL/COMP",
                           ["Y"])
renameAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(1)",
  "Disp")

nodeList = []
for i in range(0, L * 200 + 1):
    nodeList.append([i / 200, 0.028, 0])

NodeRange = findNearestNodes(nodeList)

setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/FILE",
  name + "_dispXX")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/DEVICE",
  "TABULA")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/LAYOUT/
  LINPAG", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/LAYOUT/
  COLLIN", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/SELECT/
  MODSEL", "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/SELECT/
  NODES(1)/RNGNRS", NodeRange)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/SELECT/
  ELEMEN(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/SELECT/
  REINFO(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/SELTYP",
  "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/USER/
  DISPLA(1)/TOTAL/TRANSL/GLOBAL/COMP",
                           ["X"])
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)/USER/
  DISPLA(1)/TOTAL/TRANSL/GLOBAL/COORDI",
                           True)
renameAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(2)",
  "dispXX")

setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/FILE",
  name + "_Top_reinf")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/DEVICE",
  "TABULA")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/LAYOUT/
```

```
  LINPAG", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/LAYOUT/
  COLLIN", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/SELECT/
  MODSEL", "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/SELECT/
  NODES(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/SELECT/
  ELEMEN(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/SELECT/
  REINFO(1)/RNGNRS", "\"Top_reinf\"")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/SELTYP",
  "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/USER/
  STRAIN(1)/TOTAL/GREEN/GLOBAL/COMP",
                           ["XX"])
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/USER/
  STRAIN(1)/TOTAL/GREEN/GLOBAL/COORDI",
                           True)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)/USER/
  STRAIN(1)/TOTAL/GREEN/GLOBAL/LOCATI",
                           "CENTER")
renameAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(3)",
  "Top_reinf")

setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/FILE",
  name + "_Bot_reinf")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/DEVICE",
  "TABULA")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/LAYOUT/
  LINPAG", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/LAYOUT/
  COLLIN", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/SELECT/
  MODSEL", "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/SELECT/
  NODES(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/SELECT/
  ELEMEN(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/SELECT/
  REINFO(1)/RNGNRS", "\"Bottom_reinf\"")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/SELTYP",
  "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/USER/
  STRAIN(1)/TOTAL/GREEN/GLOBAL/COMP",
                           ["XX"])
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/USER/
  STRAIN(1)/TOTAL/GREEN/GLOBAL/COORDI",
                           True)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)/USER/
  STRAIN(1)/TOTAL/GREEN/GLOBAL/LOCATI",
                           "CENTER")
renameAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(4)",
  "Bot_reinf")

reacNode = findNearestNodes([[0.05 * L, -0.065, 0]])
```

Z:\Exjobb\Beams\DOFS data\AnalysisGenerator.py 4

```
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/FILE",
  name + "_Load")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/DEVICE",
  "TABULA")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/LAYOUT/
  LINPAG", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/LAYOUT/
  COLLIN", 1000000)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/SELECT/
  MODSEL", "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/SELECT/
  NODES(1)/RNGNRS", reacNode)
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/SELECT/
  ELEMEN(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/SELECT/
  REINFO(1)/RNGNRS", "NONE")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/SELTYP",
  "USER")
setAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)/USER/FORCE
  (5)/REACTI/TRANSL/GLOBAL/COMP",
                           ["Y"])
renameAnalysisCommandDetail("Analysis1", "Structural nonlinear", "OUTPUT(5)",
  "Load")

setAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/SELTYP",
  "USER" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  DISPLA(1)/TOTAL/TRANSL/GLOBAL" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  FORCE(1)/REACTI/TRANSL/GLOBAL" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  STRAIN(1)/TOTAL/GREEN/PRINCI" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  STRAIN(2)/TOTAL/GREEN/GLOBAL" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  STRAIN(3)/CRACK/GREEN" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  STRAIN(4)/CRKWDT/GREEN/PRINCI" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  STRESS(1)/TOTAL/CAUCHY/GLOBAL" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  STRESS(2)/TOTAL/CAUCHY/PRINCI" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  STRESS(3)/TOTAL/FORCE/LOCAL" )
addAnalysisCommandDetail( "Analysis1", "Structural nonlinear", "OUTPUT(6)/USER/
  STRESS(4)/TOTAL/MOMENT/LOCAL" )

runSolver(["Analysis1"])
# closeProject()
```

E

Material test data

Material test data for the concrete used in reference beam *PC1*, provided by the supervisors. For testing at 28 days, only the $f_{\mathrm{cm,cube}}$ was provided and for testing at 213 days, no value for f_{ctm} was provided.

f.cm.28 day	**cube 150**				
Casting date	2019-11-19				
Testing date	2019-12-17				-43788
No	f.cm	Mean	SD	COV	
1	65,01	68,2	3,9		5,7%
2	67,05				
3	72,52				

f.cm.test date	**cube 150**				
Casting date	2019-11-19				
Testing date	2020-09-07				
No	f.cm	Mean	SD	COV	
1	80,55	78,7	4,0		5,0%
2	74,14				
3	81,39				

f.cm.test date	**cylinder 100, height 200**				
Casting date	2019-11-19				
Testing date	2020-09-07				
No	f.cm	Mean	SD	COV	
1	79,46	71,5	8,5		11,9%
2	62,53				
3	72,4				

E.cm.test date	**cylinder 100, height 200**				
Casting date	2019-11-19				
Testing date	2020-09-07				
No	E.cm	Mean	SD	COV	
1	32,71	33,3	2,6		7,7%
2	36,08	39,03177			
3	31,02	43,05308	1,1932672		
		37,01515	3,0739314		

G_F	**cube 150**				
Casting date	2019-11-19				
Testing date	2020-09-04				
No	G.F	Mean	STD	CoV	
	N/m				
1	124,2	134	10,7		8,0%
2	149,2				
3	129,5	1,110818			

F

Material properties - Calculations

Calculations to estimate the concrete strength at 14 and 507 days of curing in appendix F.1 and F.2, respectively. Estimated using EN-1992-1-1 (2004).

F.1 14 days of curing

Test date: 2019-12-17

$f_{cm.cube} := 68.2MPa$

$f_{cm} := \frac{f_{cm.cube}}{1.2} + \frac{1MPa}{1.1} = 57.742 \cdot MPa$

$f_{ctm} := 2.12 \cdot \ln\left(1 + \frac{f_{cm}}{10MPa}\right) \cdot 1MPa = 4.056 \cdot MPa$ [SS-EN-1992-1-1: Table 3.1]

$E_{cm} := 22\left[\frac{(f_{cm})^{0.3}}{(10MPa)^{0.3}}\right] \cdot 1GPa = 37.228 \cdot GPa$ [SS-EN-1992-1-1: Table 3.1]

$G_F := 73\frac{N}{m} \cdot \left(\frac{f_{cm}}{1MPa}\right)^{0.18} = 151.493 \cdot \frac{N}{m}$ [CEB-FIB Model Code 2010]

Experimental date: 2019-12-03 $t := 14$ (days)

$s := 0.2$

$\beta_{cc} := \exp\left[s \cdot \left[1 - \left(\frac{28^{0.5}}{t^{0.5}}\right)\right]\right] = 0.92$ [SS-EN-1992-1-1: Eq. 3.2]

$$\alpha := \begin{cases} 1 & \text{if } t < 28 \quad = 1 \\ \frac{2}{3} & \text{otherwise} \end{cases}$$

$f_{cm.2} := \beta_{cc} \cdot f_{cm} = 53.152 \cdot MPa$ [SS-EN-1992-1-1: Eq. 3.1]

$f_{ctm.2} := f_{ctm} \cdot \beta_{cc}^{\alpha} = 3.733 \cdot MPa$ [SS-EN-1992-1-1: Eq. 3.4]

$E_{cm.2} := E_{cm} \cdot \left(\frac{f_{cm.2}^{0.3}}{f_{cm}^{0.3}}\right) = 36.314 \cdot GPa$ [SS-EN-1992-1-1: Eq. 3.5]

$G_{F.2} := G_F \cdot \left(\frac{f_{cm.2}^{0.3}}{f_{cm}^{0.3}}\right) = 147.774 \cdot \frac{N}{m}$ $G_{C.2} := 250 \cdot G_F = 3.787 \times 10^4 \cdot \frac{N}{m}$

$\varepsilon_{c.ult.2} := \frac{1}{100} \cdot \left[2.8 + 27 \cdot \left[\frac{(98MPa - f_{cm.2})^4}{(100MPa)^4}\right]\right] = 3.892 \cdot \%$ [SS-EN-1992-1-1: Table. 3.1]

F.2 507 days of curing

Test date: 2020-09-07 **Cast date:** 2019-12-17 $t_1 := 293$

$f_{cm} := 71.5MPa$ $E_{cm} := 33.3GPa$ $G_F := 134\frac{N}{m}$ Measured material parameters

$f_{ctm} := 2.12 \cdot \ln\left(1 + \frac{f_{cm}}{10MPa}\right) \cdot 1MPa = 4.448 \cdot MPa$ [SS-EN-1992-1-1: Table 3.1]

Experimental date: 2021-04-09 $t_2 := 507$ (days)

$s := 0.2$

$\beta_{cc} := \exp\left[s \cdot \left[1 - \left(\frac{t_1^{0.5}}{t_2^{0.5}}\right)\right]\right] = 1.049$ [SS-EN-1992-1-1: Eq. 3.2]

$$\alpha := \begin{cases} 1 & \text{if } t_2 < t_1 \\ \frac{2}{3} & \text{otherwise} \end{cases} = 0.667$$

$f_{cm.2} := \beta_{cc} \cdot f_{cm} = 75.013 \cdot MPa$ [SS-EN-1992-1-1: Eq. 3.1]

$f_{ctm.2} := f_{ctm} \cdot \beta_{cc}^{\alpha} = 4.592 \cdot MPa$ [SS-EN-1992-1-1: Eq. 3.4]

$E_{cm.2} := E_{cm} \cdot \left(\frac{f_{cm.2}^{0.3}}{f_{cm}^{0.3}}\right) = 33.783 \cdot GPa$ [SS-EN-1992-1-1: Eq. 3.5]

$G_{F.2} := G_F \cdot \left(\frac{f_{cm.2}^{0.3}}{f_{cm}^{0.3}}\right) = 135.942 \cdot \frac{N}{m}$ $G_{C.2} := 250 \cdot G_F = 3.35 \times 10^4 \cdot \frac{N}{m}$

$\varepsilon_{c.ult.2} := \frac{1}{100} \cdot \left[2.8 + 27 \cdot \left[\frac{(98MPa - f_{cm.2})^4}{(100MPa)^4}\right]\right] = 2.875 \cdot \%$ [SS-EN-1992-1-1: Table. 3.1]

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY