COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

Degree in Industrial Technologies

Bachelor's or Master's final project

This is the title of your project

Author
Author's Name

Supervised by
Prof. Dr. Mc Great[a]
Prof. Dr. Mc Amazing[b]
[a] Institute of Greatness
[b] Amazing University

München 2022

Thank yous

And other important information

Abstract

Abstract content

Contents

List of Figures

List of Tables

Listings

List of Algorithms

List of Symbols

Notation	**Description**
T	Temperature
$\dot{\epsilon}^{cr}$	Creep strain rate
$\mathfrak{R}$	Universal gas constant
$\tilde{\sigma}$	Deviatoric stress component. Mises criterion
test	This is a test entry for glossaries

Acronyms

Notation	**Description**	**Page List**
BC	Boundary Condition	24
FEM	Finite Element Method	11, 13
HRP	Hot Radial Pressing	16, 24

Chapter 1

An overview of LaTeX

In this section, a few basic tools will be presented. In following sections more advance functionality and complex tools will be showcased. **Use this guide as an example and to your advantage!**

IMPORTANT: this template uses LuaLaTeX, a modern LaTeX engine. You should setup your editor to use LuaLaTeX, otherwise it will not be able to generate the document. Overleaf, for example, does not change the LaTeX engine automatically, so you have to do it yourself (it is very easy!).

IMPORTANT: read the documentation of the pacakges that you will use! Also, read general documentation about LaTeX! While the following guide below explains some of the most basic and cooler topics of LaTeX, the explanations are swallow. I do not cover details nor issues that may appear and how to fix them. Some really good resources are:

- LaTeX's Wikibook
- Overleaf's LaTeX resources
- The not so short introduction to LaTeX book
- The packages' manuals!
- Any searh engine.

1.1 Basics of LaTeX

1.1.1 Text styles

The following table showcases some of the more common text styles in LaTeX.

Style	Code	Ouput
Quotes	``` ``Quotes'' ```	“Quotes”
Boldface	`\textbf{Boldface}`	**Boldface**
Italics	`\textit{Italics}`	*Italics*
Emphasis	`\emph{Emphasis}`	*Emphasis*
Underline	`\underline{Underline}`	Underline
Typewriter	`\texttt{Typewriter}`	`Typewriter`
Small caps	`\textsc{small caps}`	SMALL CAPS
Mathematical	`$Mathematical^{\pi\cdot i}$`	$Mathematical^{\pi\cdot i}$
LaTeX Comments	`% Some text`	

Table 1.1: Text styles in LaTeX.

1.1.2 Structure of a LaTeX document

For this template, which is based in the `book` class, we have the following major sections:

1. `\part{}`: Parts are fully self-contained portions of information. They leave a full blank page with only the title of the part. **This is not used in this template and not recommended!**

2. `\chapter{}`: Your normal chapters, as you can see above. We are in the “*An overview of LaTeX*” chapter.

3. `\section{}`: Normal sections for a chapter. We are in “*Basics of LaTeX*” section.

4. `\subsection{}`: Subsections. We are in “*Structure of a LaTeX document*” subsection.

5. `\subsubsection{}`: Subsubsections. This level tends to be quite deep and will most likely not appear in the index unless we include `\setcounter{secnumdepth}{3}`[1] in the preamble[2].

6. `\paragraph{}`: One step deeper. By default paragraphs are not numbered.

You just have to write what you want between the `{}` for each command, and LaTeX does the rest. It typsets the titles/sections, it adds them to the table of contents and numbers them consistently!

[1]There is also `tocdepth`, which only affects the Table of Contents.

[2]The preamble is the part before `\begin{document}`, basically, the setup section.

1.1.3 Structure of a LaTeX paragraph

LaTeX gives us full control on how paragraphs appear in our text, but it is not obvious to know how to control such appearance.

Paragraphs can be separated by a simple empty line between themselves, with a double backslash `\\` or both. However, the results these methods produce is different. Lets take a look

```
This is a test without double backslash. Take a look at how the next paragraph is
    indented. This is the main difference with respect to the next method shown below.

This would be the beginning of the new paragraph. There is no blank line with the
    previous one.
```

This is a test without double backslash. Take a look at how the next paragraph is indented. This is the main difference with respect to the next method shown below.

This would be the beginning of the new paragraph. There is no blank line with the previous one.

```
Now, lets see how the new paragraph is formated when we end this paragraph with a double
     backslash (\verb|\\|) and without an empty line. \\
This would be the beginning of the new paragraph. This paragraph was not indented.
```

Now, lets see how the new paragraph is formated when we end this paragraph with a double backslash (\\) and without an empty line.
This would be the beginning of the new paragraph. This paragraph was not indented.

```
Now, lets see how the new paragraph is formated when we end this paragraph with a double
     backslash (\verb|\\|) and an empty line. \\

This would be the beginning of the new paragraph.
```

Now, lets see how the new paragraph is formated when we end this paragraph with a double backslash (\\) and an empty line.

This would be the beginning of the new paragraph.

1.1.4 Enumerations, bullet points and descriptions in LaTeX

Enumerated lists can be created with the `enumerate` environment.

1. First item.
2. Second one.
   (a) Going deeeeper.
3. Third.

```
\begin{enumerate}
        \item First item.
        \item Second one.
    \begin{enumerate}
            \item Going deeeeper.
    \end{enumerate}
        \item Third.
\end{enumerate}
```

Bullet points or lists can be created with the `itemize` environment. The structure is the same as the `enumerate` environment!

- First item.
- Second one.
  - Going deeeeper.
- Third.

```
\begin{itemize}
        \item First item.
        \item Second one.
    \begin{itemize}
            \item Going deeeeper.
    \end{itemize}
        \item Third.
\end{itemize}
```

Descriptions are quite nice is you need to descrive different concepts. They are created with the `description` environment and the `\item` entry requires the optional argument: `\item[Some text]`.

My favourite First item. Lets write some more text to see the full formatting of the `description` environment.

Continuation Second one.

Finally Third.

```
\begin{description}
        \item[My favourite] First item. Lets write some more text to see the full
    formatting of the \verb|description| environment.
        \item[Continuation] Second one.
        \item[Finally] Third.
\end{description}
```

1.1.5 Mathematical notation

LaTeX provides several way to include symbols and write mathematical formulas. The most basic way is to include mathematical notation or symbols into the text. This is known as *inline* and can be done with `$...$`. Whatever is between the $ symbols, is typeset in mathematical notation. This is an example: $2 = \frac{4}{2}$. This is produced using `$2 = \frac{4}{2}$`.

Another method is to write mathematical formulas in *display* mode, which is separated from the text. This can be done by wrapping the text in `\[...\]`. **This is not recommended** as the next method is better. Here is an example:

$$2 = \frac{4}{2}$$

Normally, the best way is to use mathematical environments. This environments will provide more functionality and generally number the equations and allows them to be labelled. Here are a few examples:

$$2 = \frac{4}{2} \tag{1.1}$$

The equation above, eq. (1.1), is produced by writing:

```
\begin{equation} \label{eq:simpleeq}
    2 = \frac{4}{2}
\end{equation}
```

Lets showcase more environments that help us write beautiful formulas! The `\begin{array}` environment helps us write vertically aligned formulas!

$$f(t) = \left\{ \begin{array}{lcc} A_0 + A \cdot e^{-\dfrac{t - t_0}{t_d}} & for & t \geq t_0 \\ A_0 & for & t < t_0 \end{array} \right. \tag{1.2}$$

```
\begin{equation} \label{eq:abaqus-exponential-decay}
    f(t) = \left\{
    \begin{array}{lcc}
        A_0 + A\cdot e^{-\dfrac{t - t_0}{t_d}} & for & t \geq t_0 \\
        A_0 & for & t < t_0
    \end{array}
    \right.
\end{equation}
```

The `\begin{aling}` environment may be easier to use, but it has a few quirks. Read the documentation[3] for more information.

$$a_{11} = b_{11} \qquad a_{12} = b_{12} \tag{1.3}$$

$$a_{21} = b_{21} \qquad a_{22} = b_{22} + c_{22} \tag{1.4}$$

```
\begin{align}
    a_{11}& =b_{11}&
    a_{12}& =b_{12}\\
    a_{21}& =b_{21}&
    a_{22}& =b_{22}+c_{22}
\end{align}
```

The `\begin{subequations}` allows us to have several formulas numbered into the same reference. As shown in eq. (1.5), with the first entry being eq. (1.5a).

$$\text{\texttt{XSYMM}} \equiv U1 = UR2 = UR3 = 0 \tag{1.5a}$$

$$\text{\texttt{ZSYMM}} \equiv U3 = UR1 = UR2 = 0 \tag{1.5b}$$

```
\begin{subequations} \label{eq:symmetry-bc}
    \begin{equation} \label{eq:x-symmetry-bc}
        \text{\texttt{XSYMM}} \equiv U1 = UR2 = UR3 = 0
    \end{equation}
```

[3] http://tug.ctan.org/info/short-math-guide/short-math-guide.pdf

```
    \begin{equation}
        \text{\texttt{ZSYMM}} \equiv U3 = UR1 = UR2 = 0
    \end{equation}
\end{subequations}
```

1.1.6 References

One of the strongest points of LaTeX is its wonderful and powerful referencing system. We can reference whatever we want by putting on a "tag" with the command `\label{xxx}`. Wherever the `\label` is, it will refer to it. You can see some examples above where we refered to a few equations by their labels, which are inside the `\begin{equation}` environment. This way, LaTeX knows automatically what type of thing they are referring.

The different types of references are shown in table 1.2.

Package	Command	Result
LaTeX	`\ref{eq:simpleeq}`	1.1
	`\pageref{eq:simpleeq}`	5
`hyperref`	`\autoref{eq:simpleeq}`	Equation 1.1
	`\autoref{fig:textstyles}`	Table 1.1
	`\autopageref{eq:simpleeq}`	page 5
`cleveref`	`\cref{eq:simpleeq}`	eq. (1.1)
	`\Cref{eq:simpleeq}`	Equation (1.1)
	`\cpageref{eq:simpleeq}`	page 5
	`\cref{eq:simpleeq,eq:symmetry-bc}`	eqs. (1.1) and (1.5)
	`\crefrange{eq:simpleeq}` `{eq:symmetry-bc}`	eqs. (1.1) to (1.5)

Table 1.2: Different reference mechanisms. **The author recommends `cleveref`!**. It is included in this template.

The style of references created by `cleveref` can be changed to use a full word by setting the `noabbrev` option in the `report.tex` file. The relevant line is `\usepackage[nameinlink]{cleveref}`.

1.1.7 Bibliography

Bibliography management is another strong point of LaTeX! We just need to add bibliographic entries to the bibliography database, which for this template it is the `main.bib` file. Here is what such an entry can look like:

```
@book{lovecraft2016el,
    author = {Lovecraft, H. P.},
    title = {El ćlerigo malvado y otros relatos},
    publisher = {Alianza Editorial},
    year = {2016},
    address = {Madrid},
    isbn = {9788491042105}
}
```

In order to cite the entry we just have to use `\cite{}` with the entry's identifier, like so `\cite{lovecraft2016el}` [1]. We can also have multiple cites in the same command, [1, 2] (`\cite{lovecraft2016el,norton_creep}`). It is that simple! They get automatically printed in the bibliography section.

IMPORTANT: this template uses `biblatex` as the management system, which is a powerful, flexible and modern tool. Therefore, you will need to run the `biber` command to build the bibliography after the first compilation of your document; then you will have to recompile the document after `biber` has run. Most editors do this by default.

You can also use third-party tools like Zotero[4] to manage your `.bib` database. Most bibliography management tools are capable of dealing with `.bib` entries!

1.1.8 Tables, images and floating environments

Probably, the part of LATEX that causes the most confusion among new users, are the so called *floating envrionments*. **Tables, images, algorithms, etc are floating envrionments**. This means that **LATEX can position them where it sees fit, not where they are written by the user.** In reality, LATEX is trying to optimise your document's layout and leave as little empty space as possible.

Sooo... How do we solve LATEX moving our floating environments? Here are a few solutions:

- We don't solve it. LATEX referencing tools allow us to easily point the reader to the table, image, etc. Therefore, it is not that problematic that the *floats* may not be where we put them!

- We can ask LATEX to try to place the image where it appears in our document. This is done with the *"here"* `[h]` placement modifier. More on placement modifiers later. **This is not a definitive solution.** This will just tell LATEX to try hard to do what we are asking. There is the `[h!]` modifier, which is even stronger.

[4]https://www.zotero.org/

- **A really good solution is to use** `\FloatBarrier`. It comes from the `placeins` package, included in this template. `\FloatBarrier` forces LaTeX to put all floating environment that have already appeared before the position where `\FloatBarrier` appears. This is very useful to force LaTeX to put all floats before another section that may not be related to the topic of those floats. Here is an example:

```
\section{Some topic}

\begin{figure}
    XXX
\end{figure}

\begin{table}
    XXX
\end{table}

\FloatBarrier % All previous floats will appear before this point.

\section{Some unrelated topic}
XXX
```

- We can use the placement modifier `[H]` to force the float to appear *HERE*. This is provided by the `float` package. However, **this solution is not recommended!** It can lead to some wierd and nasty document layouts!

Now, how do we actually include figures, tables, etc? They all follow the same structure, here are some examples:

Figures are declared in the `figure` environment (*SHOCK!*). You can see the image rendered in fig. 1.1.

```
\begin{figure}
    \centering % Center image horizontally
    \includegraphics[keepaspectratio, trim = 1050 12 150 30, clip, width=0.5\
    linewidth, height=0.3\textheight]{Images/monoblock-material-overview-mesh.png
    }
    \caption[Overview of \glsentryname{FEM} mesh used for the final analysis.]{
    Overview of \glsxtrshort{FEM} mesh used for the final analysis.}
    \label{fig:monoblock-overview-mesh}
\end{figure}
```

The key here is `\includegraphics`, it is what loads the graphics and allows us to set its properties. The above example is rather complex, most times you do not need these many options. Nonetheless, here is what they do:

`keepaspectratio` Keeps the size ratio of the image. Very useful if you set `height` and `width` at the same time. `\includegraphics` will use the most restrictive length to size the image.

`trim` It allows us to trim/cut the image. It cuts X amount of pixels from the `left, bottom, right, top`. This is useful if your image is too large and you only care about a small portion of it.

`clip` Only show the trimmed image.

`width and height` Sets the maximum size with respect to the width and height. We use `\linewidth` and `\textheight` to limit the size of the image in the page by using the page's natural lengths.

Then we have `\caption`, which is what adds the text to the image. We use `\caption[]` here, to modify the text that will appear in the "List of Figures", as I do not want my acronym `FEM` to be linked there, and therefore I use `\glsentryname` to control that. But more about acronyms and glossaries in section 1.2 Finally, we have `\label{}` is is what allows us to give an identifier to our image so that we can reference it.

Tables are fairly easy to do once we get used to their nature. It uses the `table` floating environment with another environment that allows us to type tabulated data. A basic example is given below and shown in table 1.3.

```
\begin{table}
    \centering % To center the table
    \begin{tabular}{lcr}
      \toprule
      Heading 1 & Heading 2 & Heading 3 \\
      \midrule
      Left aligned & Center aligned & Right aligned \\
      \cmidrule{2-3} % Example of a defined size rule
      Some info & Some info & Some info \\
      \bottomrule
    \end{tabular}
    \caption{Example of a table.}
    \label{tab:example-table}
\end{table}
```

IMPORTANT: the `&` symbol is used as a column separator in all alignment environments! Go back to the mathematical environments and see if you can see its use there.

The column alignment options for the tabular environment can be `l`, `c`, `r`, `m` or `p` among others. They refer to left, center, right alignment and the `m` and `p` refer to a limited size column whose vertical alignment is either centered or natural. You could use them as `m{0.3\linewidth}` for example. If you want to aling the text horizontally with `m` or `p` you would write `>{\centering\arraybackslash}m{0.3\linewidth}`. You can change the `\centering` for a `\raggedright` for a right aligned column. But this is getting too advance! One final bit of knowledge about very long tables and dynamicly sized columns. This template includes the package `xltabular`, which includes the well-known `tabularx` environment and merges it with the `longtable` environment (for tables that can span more than one page) generating its own `xltabular` envrionment. Please read the documentation of the `tabularx`, `longtable` and `xltabular` packages if you need to build complex tables!

There are also two packages `multicolumn` and `multirow` that allows the content of a cell to span several columns/rows. These packages are included in this template. Take a look at their documentation.

Also, **there are online tools to help you generate LaTeX tables from Excel sheets**. One such example (which I am not very familiar with nor endorse) is Tableconvert.

Finally, if you have `.csv` files or similar and you want to print them in your LaTeX document, you can see section 1.5, which shows how table 1.5 was automatically generated using `pgfplotstable`.

Heading 1	Heading 2	Heading 3
Left aligned	Center aligned	Right aligned
Some info	Some info	Some info

Table 1.3: Example of a table.

See section 1.6.1 for an example on how to have several labeled pictures, tables, etc in a single environment.

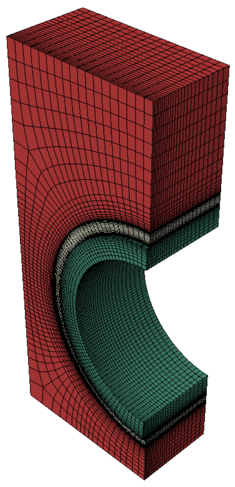

Figure 1.1: Overview of FEM mesh used for the final analysis.

1.2 Glossaries

Creating glossaries in LATEX is surprisingly easy. However, it does require a bit of understanding.

For this template, the entries for the glossaries and acronyms (which are just a different type of glossary) are loaded from the `glossaries.tex` and `acronyms.tex` files. This is just to keep things organised. So, lets define some entries for our glossary and acronym list.

For glossary entries we use the `\newglossaryentry{}{...}` command. Here is an example:

```
\newglossaryentry{identifier}
{
    name={name}, % Mandatory, what gets printed
    description={description of the entry}, % Mandatory, description that appears in the
     glossary index
    plural={plural-name}, % Optional, in case the plural is more complex
    sort={alphanumeric entry}, % Optional, how should the entry be sorted
    symbol={\ensuremath{associated symbol}}, % Optional, prints the symbol of the entry
    with \glssymbol{identifier}
}
```

For acronyms, we could use the code above, but there is a simpler and more direct way of doing it with `\newabbreviation[]{}{}{}`. Here is how it works:

```
\newabbreviation{Identifier}{ACRONYM}{Description Of Acronym}
```

`\newabbreviation[]` supports a long set of options. For example, the `[longplural={...}]` option allows us to write the plural form in case it is more refined. There are many other options. The abbreviation functionality is provided by the `glossaries-extra` package.

Once your own personal entries have been created, you can use them with the following commands.

Type	Command	Result
Glossary	`\gls{test}`	test
	`\Gls{test}`	Test
	`\glspl{test}`	tests
	`\Glspl{test}`	Tests
	`\glsentryname{test}`	test
	↑ this does not produce a link	
Acronyms	`\glsxtrshort{FEM}`	FEM
	`\glsxtrshortpl{FEM}`	FEMs
	`\glsxtrlong{FEM}`	Finite Element Method
	`\glsxtrfull{FEM}`	Finite Element Method (FEM)
	`\glsentryname{FEM}`	FEM
	`\gls{FEM}`	Finite Element Method (FEM)
What?	`\gls{FEM}`	FEM

Table 1.4: Glossary and acronym types.

Wait, what happened in the second `\gls{FEM}` entry in the acronym section? Why did it produce a different result (`\glsxtrshort{FEM}`) when compared to the first one (`\glsxtrfull{FEM}`)?

Simple. This template uses the `\setabbreviationstyle[acronym]{long-short}` style. The first time an acronym is used, it will show the full form. After that, the short form is used. All of this automatically! Isn't this magical? If you would like to show always the short form, you can delete that line from the `report.tex` or use `\setabbreviationstyle[acronym]{short-nolong}` (or any other style that you like!).

IMPORTANT: in order to show the list of glossaries and acronyms in their table of contents, you will have to run `makeglossaires`. Some editors will do that automatically for you, as they will detect you have a glossary in your document.

1.3 Automatic loading and formatting of code

The package included in this template, `listings`, allows us to format code into our document. You have already seen it in action multiple times! All the code examples in this introduction have been generated by `listings`.

The package has two ways of formatting code in our document. The first method, **which is not ideal,** is to explicitly write/copy the code into the `lstlisting` environment. Here is an example:

Listing 1.1: Hello world in Ada

```ada
with Ada.Text_IO; use Ada.Text_IO;

procedure Hello_World is
begin
    Put_Line ("Hello World");
end Hello_World;
```

```latex
\begin{lstlisting}[language=Ada, caption={Hello world in Ada}, label={lst:hello-world-
    ada}]
with Ada.Text_IO; use Ada.Text_IO;

procedure Hello_World is
begin
    Put_Line ("Hello World");
end Hello_World;
\end{lstlisting}
```

A slightly better way is to read the file that has already been written and typeset it directly. This can be achieved using `\lstinputlisting`. Here is an example:

Listing 1.2: Abaqus' `CREEP` subroutine example for several materials.

```fortran
C
C MAIN CREEP SUBROUTINE, CALLED BY ABAQUS
C
      SUBROUTINE CREEP(DECRA,DESWA,STATEV,SERD,EC,ESW,P,QTILD,
     1 TEMP,DTEMP,PREDEF,DPRED,TIME,DTIME,CMNAME,LEXIMP,LEND,
     2 COORDS,NSTATV,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
C
      INCLUDE 'ABA_PARAM.INC'
C
      CHARACTER*80 CMNAME
C
      DIMENSION DECRA(5),DESWA(5),STATEV(*),PREDEF(*),DPRED(*),
     1 TIME(3),COORDS(*),EC(2),ESW(2)
C
```

```
C DOCUMENTATION FOR THIS SUBROUTINE CAN BE FOUND IN
C https://abaqus-docs.mit.edu/2017/English/SIMACAESUBRefMap/simasub-c-creep.htm
C
C
C MATERIAL INFORMATION
C NAMES OF THE MATERIALS CAN BE CHANGED BELOW IN THEIR DEFINITION
C
C WE BREAK UP THE CREEP SUBROUTINE INTO SEVERAL OTHERS TO ALLOW
C FOR MULTIPLE MATERIALS TO BE DEFINED
C
C SELECT MATERIAL TO COMPUTE CREEP
C
C USE "MODERN" FORTRAN 95 FEATURES
C
C DEFINE MATERIAL NAMES
C
C
      CHARACTER, PARAMETER :: CU_NAME = "ITER Cu You-harden for WPDIV phase II"
      CHARACTER, PARAMETER :: CUCRZR_NAME = "CuCrZr_ITER_A"
C
C SELECT SUBROUTINE DEPENDING ON THE MATERIAL BEING COMPUTED
C
      IF (CMNAME(1:LEN(CU_NAME)) .EQ. CU_NAME) THEN
            CALL CREEP_CU(DECRA,DESWA,STATEV,SERD,EC,ESW,P,QTILD,
     1 TEMP,DTEMP,PREDEF,DPRED,TIME,DTIME,CMNAME,LEXIMP,LEND,
     2 COORDS,NSTATV,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
C            WRITE(7,*) "CREEP SUBROUTINE: COMPUTED CU CREEP"
      ELSE IF(CMNAME(1:LEN(CUCRZR_NAME)) .EQ. CUCRZR_NAME) THEN
            CALL CREEP_CUCRZR(DECRA,DESWA,STATEV,SERD,EC,ESW,P,QTILD,
     1 TEMP,DTEMP,PREDEF,DPRED,TIME,DTIME,CMNAME,LEXIMP,LEND,
     2 COORDS,NSTATV,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
C            WRITE(7,*) "CREEP SUBROUTINE: COMPUTED CUCRZR CREEP"
      ELSE
C
C MATERIAL NOT IN THE SELECTION
C WRITE ERROR
C THE ERROR GETS PRINTED IN THE .MSG FILE
C FOR MORE INFORMATION ON THE FILE DESCRIPTOR NUMBERS USED BY
C ABAQUS, SEE
C https://abaqus-docs.mit.edu/2017/English/SIMACAEEXCRefMap/simaexc-c-unitnumbers.htm
C
            WRITE(7,*) "CREEP SUBROUTINE: MATERIAL NOT FOUND"
            WRITE(7,*) "MATERIAL NAME THAT WAS RECEIVED"
            WRITE(7,*) CNAME
            WRITE(7,*) "PLEASE, FIX THE NAME OF THE MATERIAL IN THE FORTRAN CODE"
      END IF
C
      RETURN
      END
```

```
\lstinputlisting
[
language={[77]Fortran},
firstline=154,
caption={Abaqus' \texttt{CREEP} subroutine example for several materials.},
label={lst:subroutine-creep-materials}
]
{Data/Cu_CuCrZr_creep_secondary.for}
```

If you are going to be printing code of the same programming language constantly, it is easier to set the language to be used globally. This can be done with `\lstset{language=[LaTeX]TeX}` for example.

The colours, style, etc of the listings are defined in the `loaded-thesis.sty` file, which is the main body of this template. You can modify it to your liking!

1.4 Creating beautiful plots in 2D and 3D

Oh... This is an interesting and impresive part. But I must warn new LaTeX users: doing plots in LaTeX with `pgfplots` is not a trivial thing and takes a bit of practice! Read its documentation, as it is an incredibly capable package!

1.4.1 Plots with formulas

We can create simple plots by directly writing the formula. The first part of this example is the configuration of the `axis`, which is the section that controls the axis, style, etc of our plot. Once it is configured, we can start adding plots using the different `\addplot` command styles. Lets see an example:

```
\begin{figure}[h]
    \centering
    \begin{tikzpicture} % PGFplots uses TikZ to draw the plots
        \begin{axis}[ % Define the actual plot parameters
            title={Title of the plot},
            width=0.95\textwidth,
            height=0.4\textheight,
            xlabel={\glsxtrshort{HRP} time [\unit{\hour}]},
            ylabel={Temperature [\unit{\celsius}]},
            grid=major, % How fine we want the grid
            enlarge x limits=false, % By default, the axis are enlarged
```

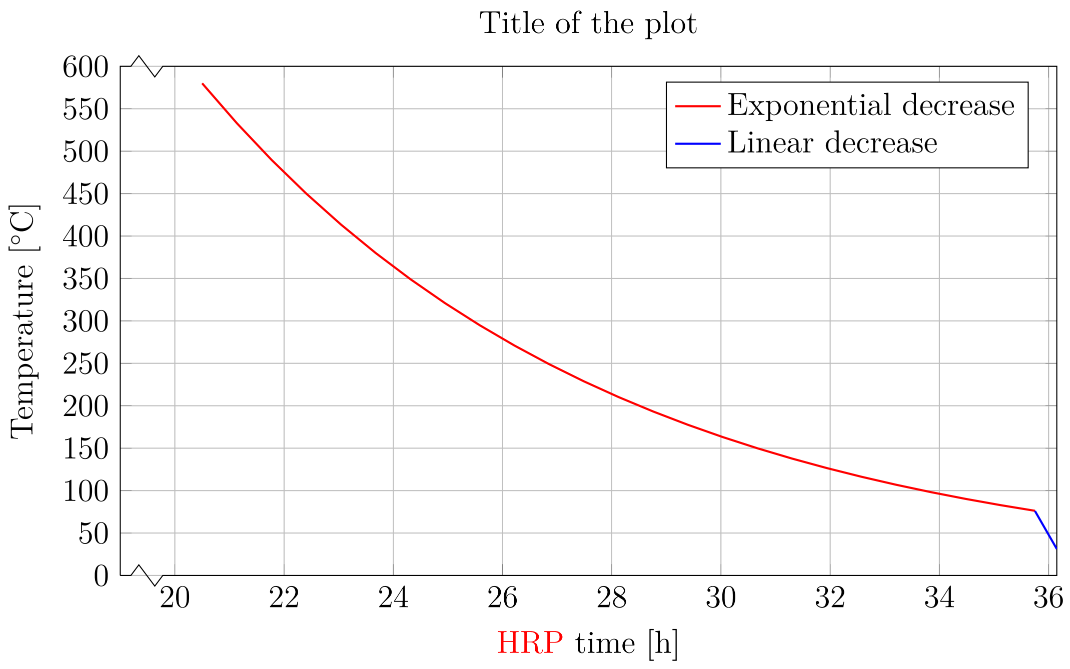

Figure 1.2: Caption of our plot

```
        axis x discontinuity=crunch, % Our domain does not start at zero, add
discontinuity
        xmin=19, % Start value for X
        xtickmin=20, % First tick for X
        ymin=0,
        ymax=600,
        ytick distance=50, % Interval for which the labels are printed
        legend pos=north east, % Were the legend should go
        legend cell align=left,
        ylabel near ticks,
        ]

        % Start adding lines!
        \addplot [
        thick, % Thickness of the line
        red, % Colour of the line
        domain=20.5:35.75 % For what domain of X the expression should be computed
        ] % And now, the actual formula
        {580*exp(-(x-20.5)/(27050/60/60))};
        % IMPORTANT, notice the trailing semicolon (;)!
        % Add line to legend
        \addlegendentry{Exponential decrease}

        \addplot [
            thick,
            blue,
            domain=35.75:36.15
            ]
            {-112*x+4080};
            \addlegendentry{Linear decrease}
        \end{axis}
    \end{tikzpicture}
    \caption{Caption of our plot}
    \label{fig:temperature-time-transient}
\end{figure}
```

We can also do 3D plots! The plot in fig. 1.3 uses glossaries in its entries, and parametised values at the top. **However, this parametrisation of values is not recommended,** as it forces `pgfplots` to use a slower engine.

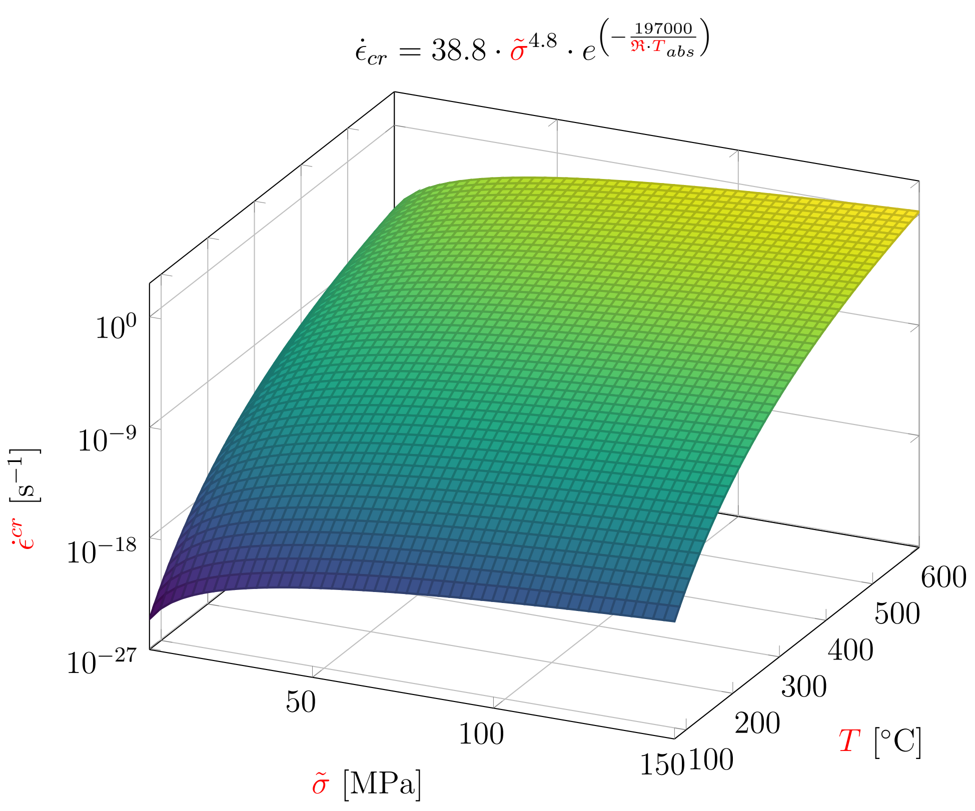

Figure 1.3: Yup, 3D plots are that easy! And this one is parametised

```
\begin{figure}[h]
    % Define values for plots
```

```latex
    % Variables
    \newcommand\Nexp{4.8}
    \newcommand\Mexp{0.0}
    \newcommand\Acoeff{38.8}
    \newcommand\SIGMAMIN{5}
    \newcommand\SIGMAMAX{150}
    % Min Temp 100C
    \newcommand\TEMPMIN{75}
    % Max Temp 600C
    \newcommand\TEMPMAX{600}
    % Self-diffusion energy
    \newcommand\Q{197000}
    \centering
    \begin{tikzpicture}
        \begin{axis}[
            title={$\dot{\epsilon}_{cr} = \Acoeff \cdot \gls{stressdevia}^{\Nexp} \cdot
    e^{\left(-\frac{\Q}{\gls{gasconstant}\cdot \gls{temperature}_{abs}}\right)}$},
            width=0.8\textwidth,
            xlabel={\gls{stressdevia} [\unit{\mega\pascal}]},
            ylabel={\gls{temperature} [\unit{\celsius}]},
            ytick distance=100,
            xtick distance=50,
            zlabel={\gls{creepstrainrate}\ [\unit{\per\second}]},
            zmode=log, % Logarithmic axis!
            grid=major,
            colormap/viridis, % Select colour
            % view={0}{90}, % View from the top
            ]

            \addplot3 [
            thick,
            surf,
            domain=\SIGMAMIN:\SIGMAMAX,
            y domain=\TEMPMIN:\TEMPMAX,
            samples=50,
            samples y=50,
            ]
            % Formula as a function of X and Y
            {\Acoeff*x^\Nexp*exp(-\Q/(8.314*(y+273.15)))};
        \end{axis}
    \end{tikzpicture}
    \caption{Yup, 3D plots are that easy! And this one is parametised}
    \label{fig:3d-plot}
\end{figure}
```

1.4.2 Plots with data from files

`PGFplots` can easily read, plot and even transform data from files. The data needs to be provided in a somewhat tabulated manner. Your typical `.cvs` or `.dat` files will do fine. This is done with the `\addplot [] table [] {}` command. The first set of options refers to the plotting style, the second one, to the reading and interpretation of data; finally, the argument points to the file.

`PGFplots` can modify the data that it reads. This allows a lot of really nice tricks; like reading `X` and `Y` coordinate data and using them to generate `Z` values with a formula for a 3D plot.

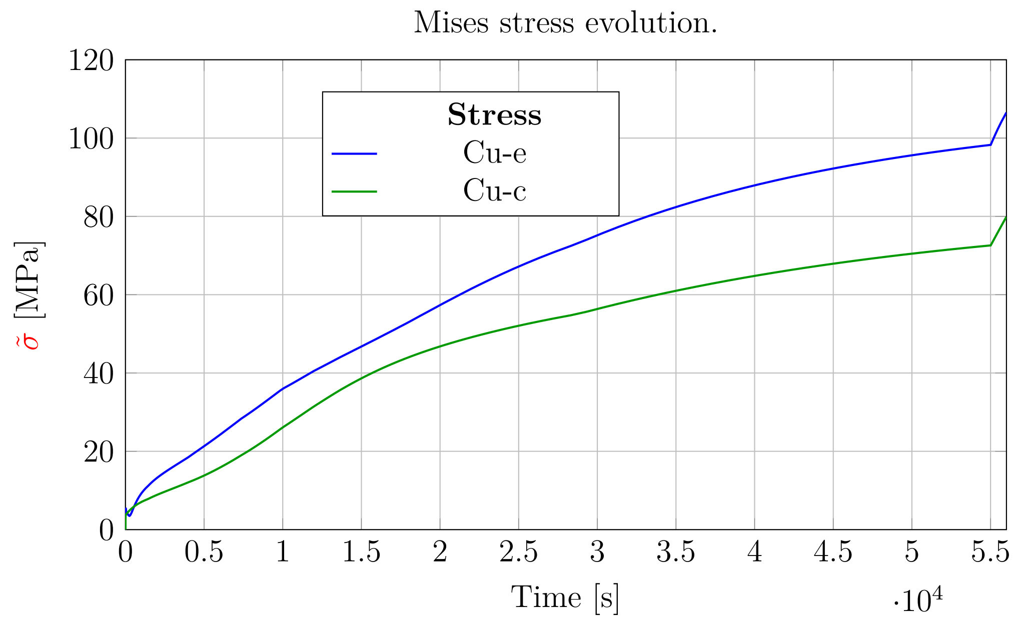

Figure 1.4: Stress history.

```
\begin{figure}[h]
    \centering
    \begin{tikzpicture}
        \pgfplotsset{set layers}
        \begin{axis}[
            title={Mises stress evolution.},
            width=0.9\textwidth,
            height=0.375\textheight,
            xlabel={Time [\unit{\second}]},
            ylabel={\gls{stressdevia} [\unit{\mega\pascal}]},
            enlargelimits=false,
            grid=major,
            ymax=120,
            % Carefully position the legend
            legend style={
                at={(axis cs:12500, 80)}, anchor=south west
            }
            ]

            % Add a legend title
            \addlegendimage{empty legend}
            \addlegendentry[text width=0.2\textwidth, align=center]{\textbf{Stress}}
            \addplot [
            thick,
            blue
            ] table
            % Now we setup the columns that are read and the type of separator
            [col sep=semicolon, x index=0, y index = 3, header=false]
            % And the file with the data
            {Data/Creep-Cu-external-PEEQ-S-TEMP.csv};
            \addlegendentry{Cu-e}

            \addplot [
            thick,
            color=green!60!black
            ] table
            [col sep=semicolon, x index=0, y index = 3, header=false]
            {Data/Creep-Cu-internal-PEEQ-S-TEMP.csv};
            \addlegendentry{Cu-c}
        \end{axis}
    \end{tikzpicture}
    \caption{Stress history.}
    \label{fig:stress-evolution}
\end{figure}
```

1.4.3 Groupped plots

1.5 Automatic formatting of table data

The following table, table 1.5, is formatted using the following general setup for `pgfplotstable`. The following LaTeX-`pgfplotstable` is only needed once, and it applies to “all” the automatically loaded table.

```
% Configure the general setting of pgfplotstable
\pgfplotstableset{
    every odd row/.style={
        before row={\rowcolor{gray!20}}
    },
    every head row/.style={
        before row=\toprule,
        after row=\midrule,
        % Don't print the row name or the row index!
        output empty row
    },
    every last row/.style={
        after row=\bottomrule
    },
    header=false,
    format=file,
    col sep=tab,
    search path={Data},
    font={\small}
}
```

And then the actual loading of the table. The following code setups the header (names, columns, etc) and then loads the data.

```
\begin{table}
    \newcommand{\prop}{Expansion}
    \newcommand{\propunit}{[\unit{\milli\meter\per\celsius\per\milli\meter}]}
    \centering
    \pgfplotstabletypeset[
    every head row/.append style={
        before row={
            \toprule
            \multicolumn{2}{c}{\glsentryname{Cu-OFHC}} \\
            \midrule
            \multirow{2}{\widthof{\propunit}}{\centering \prop\ \propunit} & \multirow
    {2}{\widthof{Temperature}}{\centering Temperature [\unit{\celsius}]} \\
            \\
        },
    },
    ]{ITER Cu You-harden for WPDIV phase II_\prop_f_T.txt}
    \caption{Automatically formatted table using \texttt{pgfplotstable}.}
    \label{tab:automatic-reading-csv}
\end{table}
```

Whats even cooler is that `pgfplotstable` uses the package `siunitx` to format the values as it is included in this template!

OFHC-Cu	
Expansion [mm °C^{-1} mm^{-1}]	Temperature [°C]
$1.68 \cdot 10^{-5}$	20
$1.7 \cdot 10^{-5}$	50
$1.72 \cdot 10^{-5}$	100
$1.74 \cdot 10^{-5}$	150
$1.76 \cdot 10^{-5}$	200
$1.78 \cdot 10^{-5}$	250
$1.79 \cdot 10^{-5}$	300
$1.81 \cdot 10^{-5}$	350
$1.82 \cdot 10^{-5}$	400
$1.84 \cdot 10^{-5}$	450
$1.85 \cdot 10^{-5}$	500
$1.87 \cdot 10^{-5}$	550
$1.88 \cdot 10^{-5}$	600
$1.9 \cdot 10^{-5}$	650
$1.91 \cdot 10^{-5}$	700
$1.93 \cdot 10^{-5}$	750
$1.96 \cdot 10^{-5}$	800
$1.98 \cdot 10^{-5}$	850
$2.01 \cdot 10^{-5}$	900

Table 1.5: Automatically formatted table using `pgfplotstable`.

1.6 Some extra bits of knowledge

1.6.1 Subcaptions, multiple floats together

1.6.2 How do I prevent LaTeX from splitting a word, number, etc?

LaTeX will automatically break some words or numbers in order to have a nice layout of the paragraph. However, sometime we don't want that. This can be solved with `\mbox{XXX}`. This, however, may generate unexpecte behaviour! For example: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX.

1.6.3 Can I put two figures, tables, etc; side to side? One on the left page and another thing on the right page.

Yes! With the package `dpfloat`, included in the template. **However, it can generate a lot of errors** if LATEX cannot put enough text around the floating environments[5]. The easiest solution is to write a bit more text or place the floats more deeply into your text.

Here is a demonstration of an example from my thesis. Inside your `figure, table, etc`, use the `leftfullpage` environment around the contents to place something in the left page. Then in the next float environment, which should be written directly next to the previous one, use the `fullpage` environment. Here is an example with fig. 1.5 and table 1.6.

```
\begin{figure}
    \begin{leftfullpage} % Place in the left page
    \centering
    \def\svgwidth{0.6\linewidth}
    \import{Data}{BCs.pdf_tex} % Import LaTeX graphics generated by Inkscape!
    \caption[\glsentryname{FEM} boundary conditions.]{Boundary conditions applied on the
     monoblock.}
    \label{fig:boundary-conditions}
    \end{leftfullpage}
\end{figure}
```

```
\begin{table}
    \begin{fullpage} % Put it in a right page
            \centering
            \footnotesize
            \begin{tabularx}{\linewidth}{>{\raggedright\arraybackslash}X >{\centering\
    arraybackslash}X >{\centering\arraybackslash}X >{\centering\arraybackslash}X >{\
    centering\arraybackslash}X >{\centering\arraybackslash}X}
              \toprule
              & Pressure increase & Exponential drop of temperature & Rapid drop of
    temperature & Cutting of cooling pipe & Cutting of the part \\
              \cmidrule(r){2-4} \cmidrule(l){5-6}
              Duration & \qty{10}{\second} & \qty{55000}{\second} & \qty{1000}{\second}
    & \qty{1}{\second} & \qty{1}{\second} \\
              \midrule
              Temperature  & \qty{580}{\celsius} & \qtyrange[range-units=single
    ]{580}{76}{\celsius} & \qtyrange[range-units=single]{76}{20}{\celsius} & \qty{20}{\
    celsius} & \qty{20}{\celsius} \\
              \midrule
```

[5]The error tends to be `Output loop---100 consecutive dead cycles.`

```
                .
                .
                .
                \bottomrule
            \end{tabularx}
            \caption{Simulated steps and their boundary conditions.}
            \label{tab:fem-steps}
    \end{fullpage}
\end{table}
```

1.6.4 How can I edit graphics in LaTeX?

In the `figure` explanation at the beginning of the chapter I show how you can clip the graphics in LaTeX. However, if you would like to add text, arrows and similar things to your graphics, **the best way is to use Inkscape**. It is a vector/image manipulation program that can ouput files to be loaded and typsetted by LaTeX! You can see fig. 1.5, which is generated with Inkscape (file named `BCs.svg` in the `Data` directory). This forum post explains how to do it.

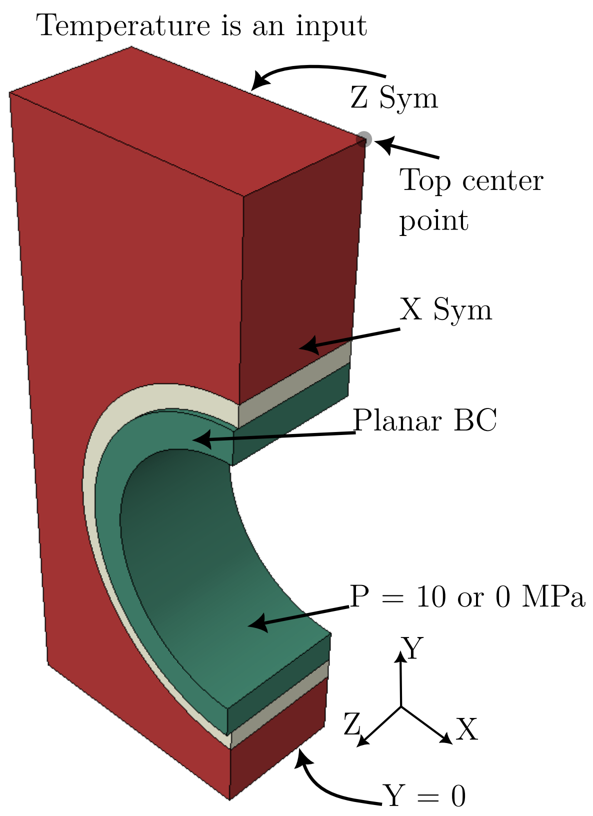

Figure 1.5: Boundary conditions applied on the monoblock.

	Pressure increase	Exponential drop of temperature	Rapid drop of temperature	Cutting of cooling pipe	Cutting of the part
Duration	10 s	55 000 s	1000 s	1 s	1 s
Temperature	580 °C	580 to 76 °C	76 to 20 °C	20 °C	20 °C
HRP pressure	0 to 10 MPa	10 MPa	10 to 0 MPa	—	—
Tie constrains	✓	✓	✓	✓	✓
X Symmetry plane	✓	✓	✓	✓	✓
Z Symmetry plane	✓	✓	✓	✓	✓
Y = 0 bottom plane	✓	✓	✓	✓	—
Planar BC	✓	✓	✓	—	—
Vertical fix of top and bottom center point	—	—	—	—	✓

Table 1.6: Simulated steps and their boundary conditions.

Bibliography

[1] H. P. Lovecraft. *El clérigo malvado y otros relatos*. Madrid: Alianza Editorial, 2016. ISBN: 9788491042105.

[2] F. H Norton. *The creep of steel at high temperatures*. McGraw-Hill, 1929. URL: https://archive.org/details/creepofsteelathi00nort.

Appendix A

This is an appendix