CMEA - Exercise

Jonas Lauener (jlauener@student.ethz.ch)

ETH Zürich

Week 01 - 24.09.2020

Exercise Website

https://www.jolau.ch/eth/cmea/

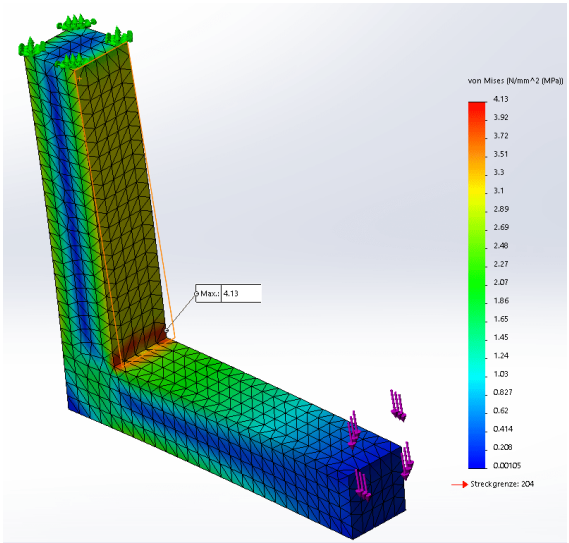
About me

- 1st semester of Master of Science in Robotics, Systems and Control
- Bachelor's degree in mechanical engineering
  - Focus Project VariLeg enhanced (exoskeleton)

- Apprenticeship as software developer

Goal of course

- Learn about numerical ODE solver $\rightarrow$ i.e. used in FEM solver



- More importantly: learn about numerical algorithms and how to implement them in C++

About exercises

- Usually pen-and-paper derivation and following implementation in C++
- Exercises are important for the graded projects
  - Projects are usually variation of preceding exercises
- Results are plotted with Matlab or Python
  - Data export functions from C++ and plotting scripts are given to you
  - Personally always used Python for plotting

Software Editor/IDE

- IDE is the software to write and run your code
- Recommending CLion
  - Works well the provided exercise templates
  - Free for students:
    `https://www.jetbrains.com/shop/eform/students`
  - Available for Windows, Mac and Linux:
    `https://www.jetbrains.com/clion/download/`
- Install and configure C++ compiler in CLion:
  - Windows: `https://www.jetbrains.com/help/clion/how-to-use-wsl-development-environment-in-clion.html#wsl-general`
  - macOS: `https://www.jetbrains.com/help/clion/quick-tutorial-on-configuring-clion-on-macos.html`
- Free to use other IDEs...

Important Resources

- Guide to "Compiling, running and plotting": `https://moodle-app2.let.ethz.ch/pluginfile.php/934075/course/section/93940/compiling.html`
  - Install guides to alternative IDEs
  - Plotting workflow
- Eigen Quick Reference: `http://eigen.tuxfamily.org/dox/group__QuickRefPage.html`
  - C++ library used for linear algebra
  - Widely used in industry
- C++ tutorials and reference: `https://www.learncpp.com/`

Data to Plot Workflow: C++ side

```cpp
#include "writer.hpp" // made for CMEA II
#include <math.h>

int main() {
    std::vector<double> sineValues(1000);

    for (int i = 0; i < sineValues.size();
        ++i) {
        sineValues[i] = sin(3.14 * i /
            1000.0);
    }

    writeToFile("sine_values.txt",
        sineValues);
}
```

Data to Plot Workflow: Python side

```
1 import numpy
2 import pylab
3
4 sine = numpy.loadtxt("sine_values.txt")
5
6 x = numpy.linspace(0, 1, len(sine))
7 pylab.plot(x, sine)
8 pylab.show()
```