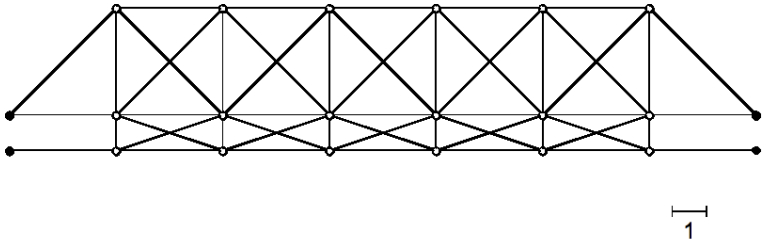
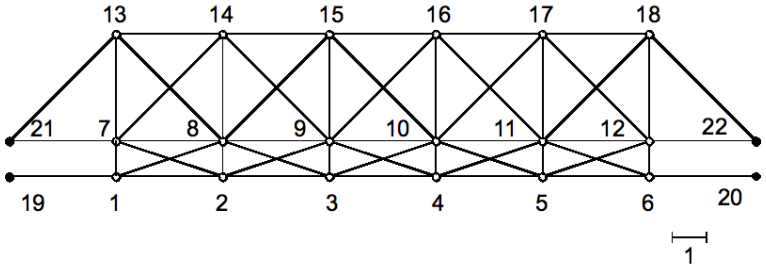
Introduction

In the civil engineering sciences bridges are often modeled as so-called *trusses*. Formally, a *truss* is a set of points P_i which are connected by edges $E_k = (P_i, P_j)$. In that respect a truss is just a directed graph. The points resemble so-called *joints* and the edges resemble so-called *members* of the truss. We give an example of such a truss in the next picture:



The joints are enumerated as follows:



The first $n = 18$ joints $P_1, \ldots, P_{18}$ are free to move. These joints are called *free joints*. The additional $p = 4$ joints $P_{19}, \ldots, P_{22}$ are assumed to be fixed and are thus called *fixed joints*. In the above truss we have $m = 55$ members in total.

We want to know, how the bridge reacts to loads. One particular load that has to be considered, is the gravitational force. It can be modeled by force vectors F_i that act on each free joint P_i. In this exercise, we assume the force vectors $F_1, \ldots, F_{18}$ are given by

$$F_i = \begin{pmatrix} f_i \\ g_i \end{pmatrix} = \begin{pmatrix} 0 \\ -9.81 \end{pmatrix} \qquad i = 1, \ldots, n. \tag{1}$$

The force acting on the joints will result in a displacement ΔP_i of each free joint P_i. The position of each joint is given by an x- and a y-coordinate, i.e.,

$$P_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} \qquad i = 1, \ldots, n. \tag{2}$$

The displacements ΔP_i are given by a horizontal and a vertical component.

$$\Delta P_i = \begin{pmatrix} \Delta x_i \\ \Delta y_i \end{pmatrix} \qquad i = 1, \ldots, n. \tag{3}$$

The truss itself can be modeled by a so-called *stiffness matrix* $A \in \mathbb{R}^{2n \times 2n}$. This matrix determines, how the bridge will react to loads. If we define the vector $b \in \mathbb{R}^{2n}$ by

$$b = \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_n \end{pmatrix} = \begin{pmatrix} f_1 \\ g_1 \\ f_2 \\ g_2 \\ \vdots \\ f_n \\ g_n \end{pmatrix} \tag{4}$$

and solve the linear system

$$Au = b, \tag{5}$$

for $u \in \mathbb{R}^{2n}$, then the displacements are given by

$$u = \begin{pmatrix} \Delta P_1 \\ \Delta P_2 \\ \vdots \\ \Delta P_n \end{pmatrix} = \begin{pmatrix} \Delta x_1 \\ \Delta y_1 \\ \Delta x_2 \\ \Delta y_2 \\ \vdots \\ \Delta x_n \\ \Delta y_n \end{pmatrix} \tag{6}$$

Exercise

Simulate and investigate the reaction of the above truss to the forces $F_1, \ldots, F_{18}$ acting on the free joints using the C-XSC library.

1. Download the C-XSC library from the following homepage:

   `http://www.math.uni-wuppertal.de/~xsc/`

   and install it by running the `install_cxsc` script. Make sure you choose the static library installation. If you choose the dynamic library installation, you might have problems when compiling your solution. If you do, look at the provided examples to adapt the compilation.

2. Make yourself familiar with the data types `real`, `rvector`, `rmatrix`, `interval`, `ivector` and `imatrix`.

3. Download the following files from the homepage of the lecture:

   `Makefile`, `main.cpp`, `truss.hpp`, `truss.cpp`, `truss.txt`

   and copy them into your source folder. A call to the function `stiffness` defined in `truss.hpp` will return the stiffness matrix A of the above truss as an `rmatrix`. The file `main.cpp` contains a main function that uses `stiffness` to load A into a variable.

4. Solve the linear equation system (5). Represent u and b as an `rvector`. Implement the Gauss elimination to solve the system in the `main.cpp` file. The Gauss elimination is given by the following pseudo-code:

```
INPUT     A: matrix of dimension n-by-n
          b: vector of dimension n
OUTPUT    u: vector of dimension n, solution of A*u = b
VARIABLES y: vector of dimension n

ALGORITHM

  FOR i = 1,...,n
    FOR j = i,...,n
      FOR k = 1,...,i-1
        A[i][j] = A[i][j] - A[i][k] * A[k][j]
      END
    END
    FOR j = i+1,...,n
      FOR k = 1,...,i-1
        A[j][i] = A[j][i] - A[j][k] * A[k][i]
      END
      A[j][i] = A[j][i] / A[i][i]
    END
  END
  FOR i = 1,...,n
    y[i] = b[i]
    FOR k = 1,...,i-1
      y[i] = y[i] - A[i][k] * y[k]
    END
  END
```

```
FOR i = n,...,1
  x[i] = y[i]
  FOR k = i+1,...,n
    x[i] = x[i] - A[i][k] * x[k]
  END
  x[i] = x[i] / A[i][i]
END
```

Implement this algorithm using the C-XSC type `rvector`. What is the maximal vertical displacement of of the joints, and for which joints is the maximal displacement attained. What does that mean?

5. Due to measurement uncertainty and round-off errors, the result obtained in the previous part of the exercise can actually be inaccurate. Study a worst-case scenario, in which all entries of the vector b are uncertain by 0.1% and in which the components of A are subject to computational round-off errors.

   To study this worst case scenario, represent A as an interval matrix $[A]$ and b as an interval vector $[b]$ using the C-XSC types `imatrix` and `ivector`. Solve the linear interval system

   $$[A][x] = [b] \tag{7}$$

   for the interval vector $[x]$ by implementing the Gauss elimination in interval arithmetic. What maximal vertical displacement do you get by this? How do you judge the quality of the interval solution $[x]$?

6. Usually, naive implementations of standard algorithms in interval arithmetic yield solution intervals that are too big. This is also the case for the Gauss elimination. Download the following files and copy them into your source folder.

   `ilss.hpp`, `ilss.cpp`, `lss_aprx.hpp`, `lss_aprx.cpp`

   Modify the program target in the Makefile to include the object files related to those ones (`ilss.cpp` and `lss_aprx.cpp`) so that your compilation keeps working.

   Study the worst-case scenario in the previous section again, using the function `LSS` defined in `ilss.hpp`. The function `LSS` solves linear interval systems in a more appropriate way. What maximal vertical displacement do you get now? How good is the interval solution $[x]$?

Delivery format

You have to hand your solution using the PACA system available at:

`http://paca.ime.usp.br/`

1. Software source code

   You should package all the files required to compile and run your solution. This will typically include the following files:

   `Makefile`, `main.cpp`, `truss.hpp`, `truss.cpp`, `truss.txt`, `ilss.hpp`, `ilss.cpp`, `lss_aprx.hpp`, `lss_aprx.cpp`

   If you have any other file required, remember to pack it as well. Please DO NOT include generated binaries or temporary files (such as .o files). One must be able to just run `make all` to compile your program.

2. A work report in PDF

   To answer the questions of the exercise, you should produce a work report document. It should contain an evaluation of the work required, of your own work and of the result. It should also present the answers for all the questions of the exercise and a critical analysis of the advantages and disadvantages of this computing concept.

You should hand over a single tarball (Gzipped tar file) that contain all those files. The name of that file must be in the following format:

your-name-ep1.tar.gz

and it should extract to a folder of the same name (without the extensions) that contain all the files.

Heb pret!