For this final self-study you are going to continue on from what you completed in the lab, hopefully this task will also help with your coursework for MECH1206. Again, any feedback on Scriptcheck would be appreciated.

This task will build on the lab 7 exercise you undertook, solving the equations for a 13 member pin-jointed truss bridge, that is you should be able to utilise some of the code you've developed already to complete this task.

Download test_bridge.m and breakingLoadTests.csv from the VLE to your local working directory.

Background

MATLAB allows us to very quickly solve the underlying sets of simultaneous equations used for calculating the member forces of a pin-jointed truss bridge. In the lab exercise you should have solved the equations to determine the internal member forces and external reaction forces when the bridge shown in Figure 1 was loaded with 50N loading at the points C, E and G. We are now going to expand on this to check if the bridge will actually fail under different loading conditions, with different values of truss angle, θ and different breaking loads for the tensile and compressive members.

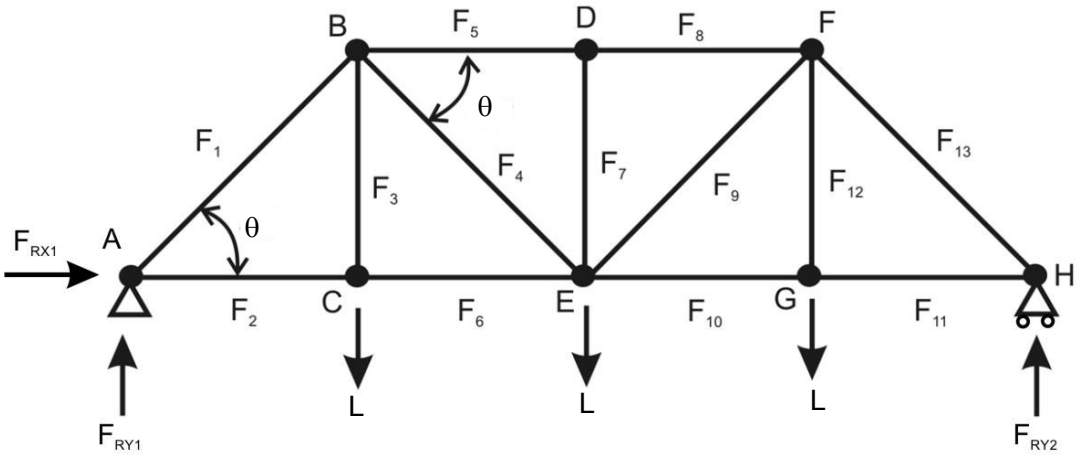

Figure 1. Diagram of the pin jointed truss bridge.

Objectives

This exercise should allow you to demonstrate the following skills:

- File I/O
- Solving simultaneous equations
- Indexing
- Using MATLAB to do basic mathematics

Requirements

Your task is to write a user-defined function that will test a pin-jointed truss bridge for a given test load and return the number of members that would have broken under that load (in reality, a bridge may fail as soon as a single member gives way). There are a number of variables that need to be used within your code, these allow for a variation in truss angle, θ, breaking loads and external loading, L. A file is given that contains breaking load data for 25 different bridges.

The program should output a value of -1 if a negative load (an upwards force in this case) is applied or if the truss angle θ is outside of the range 15^o to 75^o. Otherwise it should return the number of members where the calculated loads exceed those which the member can withstand. Note that some values will be negative if the member is in compression and your code will have to deal with this.

The function definition should be ***exactly*** as follows and should go at the start of your m-file (do not change the names of the input or output arguments):

```
function brokenMembers = breakingBridges(loadFile, delimiter, trussAngle, testNo, testLoad)
```

loadFile will pass the name of the file that holds the sets of breaking load information. The file breakingLoadTests.csv can be found on the VLE, download it and see how it is structured. It contains 25 rows, each one for a separate test, and 13 columns, one for each of the truss members. The values are the breaking loads in N, note a negative value is a compressive load.
delimiter will pass the delimiter used in the file. In this example a .csv file is used but we may want to use other delimiters (such as tab) in the future.
trussAngle contains the value for θ in degrees and should be used to adjust the values in the coefficient matrix.
testNo contains a numerical value between 1 and 25 to select the row of data from breakingLoadTests.csv that should be used for the test and *testLoad* is the total load that will be applied to the bridge, evenly distributed.

Your code should calculate the member forces in each of the 13 members when the structure is subjected to the *testLoad*. (bear in mind this is the total force applied to the structure, if is evenly distributed across the struts at C, E and G. Also that you are just modelling half the bridge).
It should compare the member forces to the breaking loads given by the data in the .csv for the particular *testNo* (remembering some member forces and breaking loads will be negative) and determine how many breaking loads have been exceeded.

The function should return a variable, brokenMembers, that contains a numerical value between -1 and 13.

You can quickly check your code by using the test_bridge.m program available on the VLE. This will undertake two tests on your function using different parameters and let you know if they are correct. Type `help test_bridge` for info. When you've passed this test, go on to ScriptCheck below.

Hints

Hopefully when creating your coefficient matrix in lab 7, you used cosd(θ) for the horizontal components and sind(θ) for the vertical components and kept them as variables rather than hard coding the numerical values. This will make it easier to adjust for different truss angles. Also remember there are no breaking loads for the reaction forces, so remove these before you undertake the comparison.

ScriptCheck

Follow these steps to evaluate your solution:

1. When you have completed your function copy the whole of your script including all comments.
2. Open a web browser and go to scriptcheck.leeds.ac.uk.
3. From here you will need to logon. To do this enter your University e-mail address and the password is your student ID number.

4. Once logged on select *Exercises* from the left hand menu and then *Break Bridges* followed by *Submit Solution*.
5. Paste your code into the text box and click *Submit.*
6. When the button appears, click *View Feedback.* This will present your results, letting you know how accurate your code is, number of lines of code and comments and will provide some feedback comments about your code.
7. You can then select *View Solutions* from the left hand menu and see your result against the results of the cohort.
8. If your code is incorrect, you can resubmit by going back to step 4.