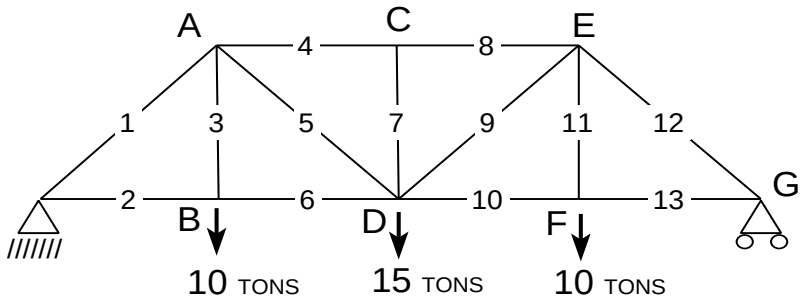
A Truss Problem

A typical task in structural engineering is to design a bridge to be strong enough to withstand a certain load. Consider the following *plane truss*, which is a set of metal bars connected by frictionless pin joints. ("Plane" refers to the fact that the truss is two-dimensional, not three-dimensional as it would be in reality.) The symbol at the left end of the truss indicates that it is fixed at that end, while the symbol at the right end indicates that the truss is free to move horizontally, but not vertically. The three arrows pointing down represent loads on the truss. These loads are 10 tons, 15 tons, and 10 tons respectively.



The problem is to solve a certain linear system of equations for the *internal forces* in the bars. A *positive* internal force indicates that the bar is being *extended* (pulled apart a little), by the load, while a *negative* internal force indicates that the bar is being *compressed*. It is assumed that, as long as the internal forces are not too big, bars will not be stretched or compressed more than a tiny amount: thus the structure does not collapse, but remains in equilibrium. By computing the internal forces, an engineer has more information as to whether the truss is indeed strong enough to withstand the load.

There are two linear equations for each internal joint in the truss, representing forces in the horizontal and vertical direction which must balance at the joints. Let us denote the internal forces by $x_1, x_2, \ldots, x_{13}$, corresponding to the numbers on the bars in the illustration. The balancing of forces at joint C in the *horizontal* direction gives the equation

$$x_4 = x_8$$

while the balancing of forces at joint C in the *vertical* direction gives simply

$$x_7 = 0.$$

The balancing of forces at joint B in the *horizontal* direction gives

$$x_2 = x_6$$

while the *vertical* direction at joint B gives

$$x_3 = 10.$$

The "10" comes from the 10 ton vertical load at joint B. The balancing of forces at joint A is a little more complicated, since it involves two bars oriented at an angle of 45 degrees as well as a horizontal and a vertical bar. Let $\alpha = \cos(\pi/4) = \sin(\pi/4) = \sqrt{2}/2$. Then the balancing of *horizontal* forces at joint A gives the equation

$$\alpha x_1 = x_4 + \alpha x_5$$

and the balancing of *vertical* forces at joint A gives

$$\alpha x_1 + x_3 + \alpha x_5 = 0.$$

There are also horizontal and vertical force equations at joints D, E and F which can be derived using the same ideas. These amount to 12 equations altogether. The 13th equation comes from the right end point G: since this end point is free to move horizontally, but not vertically, there is just one force equation, balancing the forces horizontally:

$$x_{13} + \alpha x_{12} = 0.$$

Thus, we have a total of 13 linear equations defining the 13 internal force variables.

There are several parts to the assignment, all important:

1. **Derive** the 13 linear equations in 13 variables. Write the equations using matrix notation, as $Ax = b$, and enter the matrix A and right-hand side vector b in a Matlab function. (This is better than working at the keyboard so you don't have to keep retyping if you make a mistake.)

2. **Solve** the system of linear equations, using the Matlab backslash operator: `x = A\b`. You can put this in the function too, and return x, the vector of internal forces, as an output parameter of the function. Print the solution vector `x` (you should compare it with what other classmates get to make sure you set up the system correctly).

3. **Graphically display** the solution, using `plot`. After plotting the first line of the plot you need to execute `hold on` before continuing with plotting. The figure shown should look something like the one above, but instead of labeling the bars with numbers, *plot the bars with variable thickness*, corresponding to the magnitude of the internal force in the bar. Type `help plot` to see how to get lines of variable thickness (toward the end of the help info). Use one color for positive forces (bars being extended by the load), and another color for negative forces (bars being compressed by the load). Thick bars correspond to bars under great stress from the load, while thin bars correspond to bars under little stress. For a zero or very nearly zero force, you could use a dotted line. There is no need to label the joints. Look at which bars are under extension and which are under compression: does it make sense, bearing in mind where the load is?

4. **Experiment** with different choices for the load. This changes only the vector b, not the matrix A. You can pass the load vector (length 3) to your first function as a parameter. Try loads of 10,20,30 tons instead of 10,15,10, and try some other loads too. Choose a couple of loads that give interesting pictures and include them in your submission. Do negative loads make sense (try them)?

5. **Generalize** by writing a new function that sets up and solves the equations for a *variable-sized truss, with k sections exactly like the section ABCDEF instead of one*, where k is an input parameter to the function. You do *not* need to plot the resulting trusses. What you need to do is write code to *automatically* set up the matrix A and vector b defining the system of equations for a variable-sized truss. This will require some careful thought. Start by sketching the larger truss on paper and carefully writing down the relevant equations systematically; working together with a classmate for this part is particularly recommended, but don't forget to acknowledge his/her help in your comments. Make sure you number the variables in the appropriate order, so that you

recover your original answer when $k = 1$. The load is a vector whose length depends on k (what is it?) and should be provided to the function as a second input parameter. Make sure you include plenty of comments explaining the code. Test your function for the case $k = 1$, $k = 2$ and $k = 3$ and carefully look at the output to see if it makes sense before going on to the next part.

6. **Solve** the new system using a load vector that increases regularly like this from left to right: $10, 20, 30, \ldots$. Print the computed internal force vector x for $k = 10$ (as numbers, you don't need to plot these values).

7. **Sparsity:** Use `spy` to display the nonzero entries in the matrix A for $k = 10$. Is there a pattern? Does it make sense? Do the same for A^{-1} (the *inverse* of A, computed by `inv(A)`) *and* the L and U factors obtained from `[L,U] = lu(A)`. By looking at L, seeing how close it is to being lower triangular as opposed to permuted lower triangular, you can see whether pivoting occurs in the LU factorization; does it? How sparse are A^{-1}, L and U, compared with the sparsity of A? Answer this in some detail, comparing the four different *spy* plots.

8. **Timing Comparsion:** Experiment with how long it takes to solve the system of equations for k that is *large enough that timing comparisons are meaningful* (use `timeit`, **not** `cputime`). Compare the following:

   - `x=A\b`
   - getting x by first computing A^{-1} and multiplying it onto b
   - getting x by first computing the L and U factors with `lu` and then solving two triangular systems using `\`. (This is what is actually going on when you type `x=A\b` as explained in class, so the timing should be about the same.)
   - the same 3 again, but with A set up as a *sparse* matrix; type `help sparse` for information as to how this works; we will also discuss this in class. This means editing your function that sets up A accordingly; add an input parameter that determines whether A is to be set up as a dense or sparse matrix.

   How do the execution times compare? Does this relate to the sparsity displayed in the `spy` plots?

9. **Timing When k is Increased:** Compare the timings for the k which you reported in the previous paragraph and the timings for larger values of k. Do this just for solving with `x=A\b`, but for the two cases: A is dense, and A is sparse. Plot the running times as a function of k, using whatever format you think is informative. You can plot both running times on one plot using `legend`.

Submit: listings of the Matlab functions, several plots of the original truss under various loads, the plots generated by `spy` for $k = 10$, the derivation of the equations, especially for the generalized truss, the output x for the original truss and the generalized truss with $k = 10$, the plot of running times as a function of k and answers to all the questions above. Carefully document everything, showing which pictures go with which loads, etc. Staple the pages together. Print color plots if possible; however, black and white printing is OK as long as the bars under compression and under extension are clearly distinguished.