PX390, 22-Nov-2022, Assignment 4: Euler-Bernouli beam oscillations

November 22, 2022

Consider oscillations on a beam displaced a small amount from its equilibrium position. The beam is treated as long and narrow. We seek the displacement $w(x,t)$ of the beam from equilibrium as a function of position x along the beam and time t, given a forcing $q(x)\sin(\omega t)$ periodic in time. The left and right end of the beam are at $x = 0$ and $x = L$ respectively.

The motion of a beam subjected to this time-varying force may be solved via:

$$-\mu(x)\frac{\partial^2}{\partial t^2}w(x,t) = -\frac{\partial^2}{\partial x^2}\left(K(x)\frac{\partial^2}{\partial x^2}\right)w(x,t) + q(x)\sin(\omega t) \tag{1}$$

where $K(x)$ is the flexural rigidity (resistance to bending), and $\mu(x)$ is the mass density. The inhomogeneous solution is found by choosing $w(x,t) = w(x)sin(\omega t)$ leading to

$$\mu(x)\omega^2 w(x) = -\frac{\partial^2}{\partial x^2}\left(K(x)\frac{\partial^2}{\partial x^2}\right)w(x) + q(x). \tag{2}$$

Your task is to find numerical solution $w(x)$ of equation (2), given the boundary conditions, coefficients $K(x)$, $\mu(x)$ as well as the source term $q(x)$. The coefficient $K(x)$ and mass density $\mu(x)$ are taken to be positive. Note that this is a fourth order differential equation. The best approach is to descretise the higher order derivative in the form it is written, that is, write a finite difference for the expression $[K_k w_k'']''$, where primes denote the finite difference derivative.

We consider only a cantilevered beam, fixed at one end and free to oscillate at the other, which corresponds to the following boundary conditions:

$$\text{at } x = 0\text{:} \quad w(x) = 0 \quad \text{and} \quad \frac{\partial w(x)}{\partial x} = 0 \tag{3}$$

$$\text{at } x = L\text{:} \quad \frac{\partial^2 w(x)}{\partial x^2} = 0 \quad \text{and} \quad \frac{\partial^3 w(x)}{\partial x^3} = 0 \tag{4}$$

You are required to use the LAPACK library to invert matrices, so the compiler flags are different this time. This library is installed on the lab computers and on vonneumann. We will not provide any support for installing these libraries

on your own computers, it might be quicker just to log in to vonneumann to test your program. In order to compile your program (here, named prog4.c) on vonneumann you need to execute the following commands:

1. module purge
2. module load intel impi imkl
3. gcc prog4.c -lmkl -liomp5 -lm

You also need to add

```
#include <mkl_lapacke.h>
```

in your source code. As before, you should test your code with simple test cases. We will, again, be marking it mostly based on how well it reproduces certain tests.

1 Suggestions on getting started

- The easiest way to call LAPACK is to use the matrix example codes *band_utility.c* as a framework for setting the values of the matrix at particular locations and solving the inverse problem.
- It may be helpful to solve for the force as a function of displacement to check whether your matrix is appropriate. It should be banded and symmetric.
- There are analytical solutions to these equations for simple cases.
- The boundary conditions require an extension of the ideas shown in the lectures for second order derivatives. You can extend the domain to include 'ghost points' outside the domain and use Taylor expansion to apply the boundary conditions at the grid points within the domain of your numerical solution.

2 Specification

The input and output x grid has N grid points, which will include the end of the domain; this grid will be taken to run from $x_0 = 0$ to $x_{N-1} = L$. The grid points you use in setting up the matrix problem are up to you. Ideally, for the sake of efficiency, the matrix should be a narrow banded matrix.

3 Input

The input parameters will be read from a file called 'input.txt'.

1. L: length of the beam, right boundary of the x domain

2. N: number of grid points for the x domain

3. ω: angular frequency of oscillations

I will leave you to determine whether these should be read as integers or floating point numbers based on their meaning. Note that the example input file does not necessarily have decimal points on all the values that should be floating point. It is sufficient to use scanf to read in these parameters and simply test that the reads were successful: I am not specifically forbidding the use of other ways of reading input but in the past, roughly 50% of students attempting a more complicated input scheme got it wrong.

You may assume $N > 4$, $L > 0$, and $\mu, K > 0$. The functions $\mu(x)$, $K(x)$ and $q(x)$ (in that order) will be given at the N grid points as three columns of double precision numbers in a file called 'coefficients.txt'.

There are example input files and output file on the assignment page.

4 Output:

The code should output results to a file 'output.txt', with x and $w(x)$ in two column format for the grid points $x_0, x_1, ...x_{N-1}$.