INF5620 - Numerical methods for partial differential equations

Problem set 3

Even Marius Nordhagen

October 10, 2017

- For the Github repository containing programs and results, follow this link: `https://github.com/UiO-INF5620/INF5620-evenmn/tree/master/assignment_3`

1 Introduction

The aim of this exercise is to compute the deflection of a cable with sine functions and with a finite element method. The cable is described by the differential equation

$$T\omega''(x) = \ell(x). \tag{1}$$

We are going to find numerical solutions to the problem with mainly two methods: The least square method and the Galerkin method (also called projection method). For that we need a function space, which in the first part of this project is a set of sine functions. We will study how the solutions change when we vary the number of sine functions and also when we only look at sine functions with a even number of nodes in the interval $x \in [0, x_{max}]$. Moreover we are interested in the best possible approximation, so need to measure the error.

2 Theory and methods

As mentioned in the introduction, we are going to deal with the differential equation

$$T\omega''(x) = \ell(x) \tag{2}$$

where $\ell(x)$ is the vertical load per unit length. It is fixed in the end points $x = 0$ and $x = L$. If we assume that the load is uniformly distributed along the cable, the solution should be symmetric around the midpoint $x = L/2$. This means that

$$\omega(x_0 - h) = \omega(x_0 + h) \tag{3}$$

and the derivative at the midpoint is zero (which is obvious since the function is symmetric around that point). If we now study the half domain, this gives us another useful boundary condition: $\omega'(L/2) = 0$.

The next thing we want to do is to scale the equation such that we deal with dimensionless variables:

$$\bar{x} = \frac{x}{L/2}, \quad \bar{u} = \frac{\omega}{\omega_c} \tag{4}$$

where ω_c is a scaling parameter and $\bar{u}$ and $\bar{x}$ are dimensionless variables. If we choose $\omega_c = (1/4)\ell L^2/T$, we get the equation on a very simple form:

$$\frac{d^2\bar{u}}{d\bar{x}^2} = 1 \tag{5}$$

with $\Omega = [0, 1]$ and the boundary conditions $u(0) = 0$, $u'(1) = 0$. Observe that this is Poisson's equation $u''(x) = f(x)$ with $f(x) = -1$. I will omit the bars which indicate dimensionless variables since all the variables are dimensionless in the rest of the exercise.

2.1 Exact solution

An exact solution that matches both the equation and the boundary conditions is on the form

$$u_e(x) = \frac{1}{2}x^2 - x \tag{6}$$

which was found by guessing.

2.2 Infinite element method

2.2.1 The Least Square method

The idea behind the least square method is to find the approximation for a function such that the square norm of the residuals is minimized. We are given a specific Poisson equation, but to make the calculations more general (and avoid inner product with a constant, which is boring), I will derive the least square method for a general Poisson equation (recall Poisson's equation: $-u''(x) = f(x)$). The principle of least squares leads to the definition

$$\left(R, \frac{\partial R}{\partial c_i}\right) = 0, \quad i = 0, 1, ..., N \tag{7}$$

where R are the residuals

$$R = \frac{d^2}{dx^2}\left(\sum_j c_j \psi_j(x)\right) + f(x) = \sum_j c_j \psi_j^{''}(x) + f(x) \tag{8}$$

and the second object can be simplified to

$$\frac{\partial R}{\partial c_i} = \frac{\partial}{\partial c_i}\left(\sum_j c_j \psi_j^{''}(x) + f(x)\right) = \psi_i^{''}(x) \tag{9}$$

so we have $N+1$ equations with $N+1$ unknown C_j. Insert into the definition and you will get

$$\left(\sum_j c_j \psi_j^{''} + f, \psi_i^{''}\right) = 0 \quad \Rightarrow \quad \sum_j \left(\psi_j^{''}, \psi_i^{''}\right) c_j = -\left(f, \psi_i^{''}\right).$$

This is a linear system on the form $\sum_j A_{i,j} c_j = b_i$ with $A_{i,j} = \left(\psi_i^{''}, \psi_j^{''}\right)$ and $b_i = -\left(f, \psi_i^{''}\right)$.

2.2.2 The Galerkin method

For Galerkin's method (or the projection method) we start with

$$\left(R, v\right) = 0, \quad \Rightarrow \quad \left(R, \psi_i\right) = 0 \tag{10}$$

where $v \in \forall$ and $\psi_i(x)$ is the function space. The approach is quite similar to the least square method, and also here we obtain a linear system:

$$\sum_j \left(\psi_i, \psi_j^{''}\right) c_j = -\left(f, \psi_i\right) \tag{11}$$

so with the same form as for the least square method we have $A_{i,j} = \left(\psi_i, \psi_j^{''}\right)$ and $b_i = -\left(f, \psi_i\right)$.

2.2.3 Function space

The purpose of this exercise is to find the best approximation to the given differential equation using sine functions, which means that our function space needs to consist of sine functions. We are going to look at two function spaces, one that contains of sine functions with (in the principle) all frequencies, and one that only contains of sine functions with a even number of nodes. The latter is expressed by

$$\psi_i(x) = \sin\left((2i+1)\frac{\pi}{2}x\right) \tag{12}$$

and first one is expressed by

$$\psi_i(x) = \sin\left((i+1)\frac{\pi}{2}x\right). \tag{13}$$

One advantage of these function spaces is that they have orthogonal function spaces, which gives us a diagonal A-matrix. This makes it very easy to find A^{-1} when we are calculating $C_i = A_i^{-1}b_i$. The first one also satisfies the boundary conditions when we only look at half of the domain, while the last one satisfies the boundary conditions for all the domain.

2.3 Finite elements

The idea behind the finite element method that.. It can be implemented in several ways, both simple and complicated, where the same theory lies behind. In this problem I will choose a rather simple implementation: Initially we again have Poisson's equation $-u''(x) = f$ with $x \in [0, L]$ and the boundary conditions $u(0) = u(L) = 0$. We will now divide the domain $\Omega = [0, L]$ uniformly into N_e elements where the length of an element is $h = L/N_e$. Every element takes N_n nodes and every node i has its own basis function φ_i which has value 1 at that node and value 0 at all the other nodes. In most cases we have $N_n = 2$ nodes per element, denoted by P1 elements, but we can also have more. We use Galerkin method on inner product form to obtain the linear system $\sum_j A_{i,j}c_j = b_i$ where

$$A_{i,j} = (\psi_i', \psi_j'), \qquad b_i = (f, \psi_i).$$

To avoid boundary problems, we will ignore the basis functions that live in the first and last element which are non-zero at the end-points. We when get $\psi_i = \varphi_{i+1}$ and so on:

$$A_{i,j} = (\varphi_{i+1}', \varphi_{j+1}'), \qquad b_i = (f, \varphi_{i+1}). \tag{14}$$

Assume linear basis functions

$$\begin{aligned} x \in [N_{i-1}, N_i]: &\quad \varphi_i(x) = (1-i) + hx \\ x \in [N_i, N_{i+1}]: &\quad \varphi_i(x) = (1+i) - hx \end{aligned}$$

which are 1 at node i and 0 at the neighbor nodes. This gives $\varphi_i'(x) = h$ for $x \in [N_{i-1}, N_i]$ and $\varphi_i'(x) = -h$ for $x \in [N_i, N_{i+1}]$. By plugging this into equation (**??**) this gives the linear system

$$\frac{1}{h}\begin{bmatrix} 2 & -1 & \dots & 0 \\ -1 & 2 & \ddots & 0 \\ \vdots & \ddots & \ddots & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ \vdots \\ c_N \end{bmatrix} = \begin{bmatrix} f \cdot h \\ \vdots \\ \vdots \\ f \cdot h \end{bmatrix} \tag{15}$$

which can be verified by the second derivative formula.

3 Results

3.1 Infinite element method

I will first use the least square method to find a numerical solution to the problem, and then compare the results to the numerical solutions found by the Galerkin method. All the results are found numerically. For the least square method I have used the example program *approx1D.py* written by H. P. Langtangen, and for the Galerkin method I made my own function based on the theory above. This function can be found in *approx_galarkin.py*, while the rest of the code used here is stored in the python file *cable_sin.py*.

3.1.1 The Least Square method

The first thing we want to study is how the coefficient ratio C_{i+1}/C_i is changing when $i \to N$. With $N = 10$ and using the only "odd" sine functions (function space from equation (**??**)) we get the coefficients and ratios presented in table (**??**).

Table 1: The coefficients when we have 11 sine function ($N = 10$) and the ratios between them.

$i \in [0, N]$	Coefficient	Ratio C_{i+1}/C_i
0	-0.516024550931	
1	-0.0191120204049	1/27
2	-0.00412819640745	27/125
3	-0.0015044447549	125/343
4	-0.000707852607587	343/729
5	-0.000387696882743	729/1331
6	-0.000234876900742	1331/2197
7	-0.000152896163239	2197/3375
8	-0.000105032475256	3375/4913
9	-7.52332046845e-05	4913/6859
10	-5.57201761075e-05	6859/9261

We observe that the ratios follow a pattern, given by the formula $(2i-1)^3/(2i+1)^3$.

An important aspect in numerical calculations is to know the error at any time. We expect the error to be largest in the midpoint since this point is furthest away from the boundaries. We should also have the largest error when $N = 0$, so we measure the error at $x = 1$ with one sine function. The exact cable position here is $u_e(x = 1) = -1/2$, the approximated position is $u(x = 1) = -16/\pi^3$, and we therefore get an absolute error $E \simeq 0.01602$.

We also want to visualize the numerical solution u for various number of sine functions, and the result in found in figure (**??**).

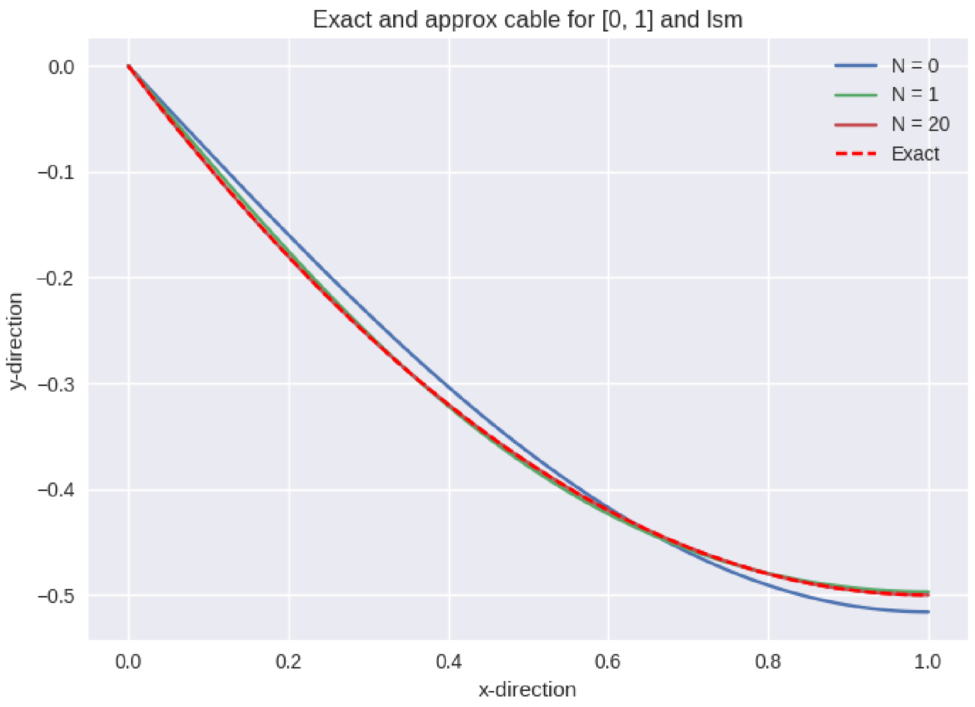

Figure 1: Here you can see the approximations when we have 1, 2 and 21 sine functions. The dashed line is the exact solution

3.2 The Galerkin method

The first thing we want to study is how the coefficient ratio C_{i+1}/C_i is changing when $i \to N$. With $N = 10$ we get the coefficients and ratios presented in table (**??**).

Table 2: The coefficients when we have 11 sine function ($N = 10$) and the ratios between them.

$i \in [0, N]$	Coefficient	Ratio C_{i+1}/C_i
0	-0.516024550931	
1	-0.0191120204049	1/27
2	-0.00412819640745	27/125
3	-0.0015044447549	125/343
4	-0.000707852607587	343/729
5	-0.000387696882743	729/1331
6	-0.000234876900742	1331/2197
7	-0.000152896163239	2197/3375
8	-0.000105032475256	3375/4913
9	-7.52332046845e-05	4913/6859
10	-5.57201761075e-05	6859/9261

Also here we observe that the ratios follow the formula $(2i-1)^3/(2i+1)^3$, and the results are actually identical to the Least Square method results.

We are interested in the error for this method as well, and incredibly the error is exactly the same at $x = 1$ with one sine function: $E \simeq 0.01602$.

We also want to visualize the numerical solution u for various number of sine functions, and the result is found in figure (**??**)

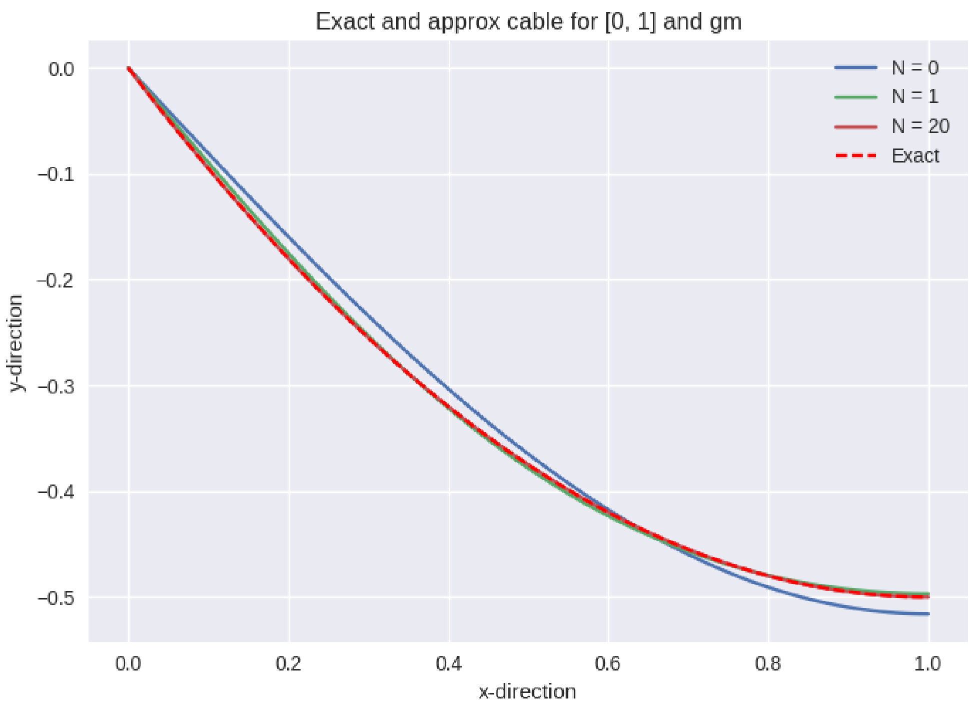

Figure 2: Here you can see the approximations when we have 1, 2 and 21 sine functions. The dashed line is the exact solution,

If we now change to a function space with all sines (that one in equation (**??**)), we get the result in figure (**??**). As we can see, the approximation is way better when we only have "odd" sines. I have also visualized the cable in entirety in figure (**??**).

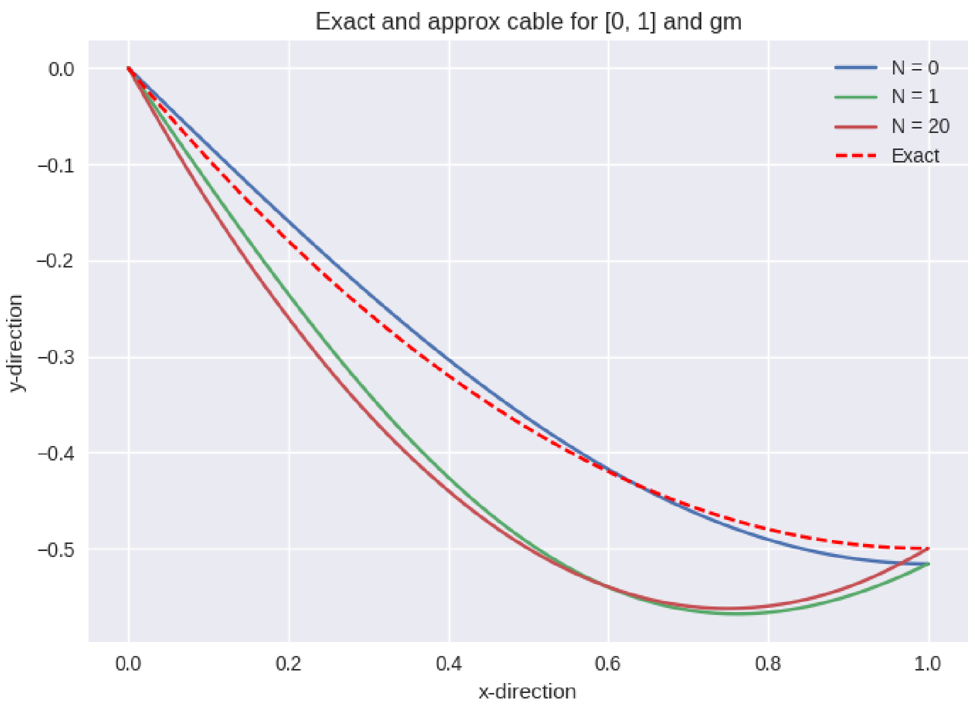

Figure 3: Here you can see the approximations when we have 1, 2 and 21 sine functions. The dashed line is the exact solution,

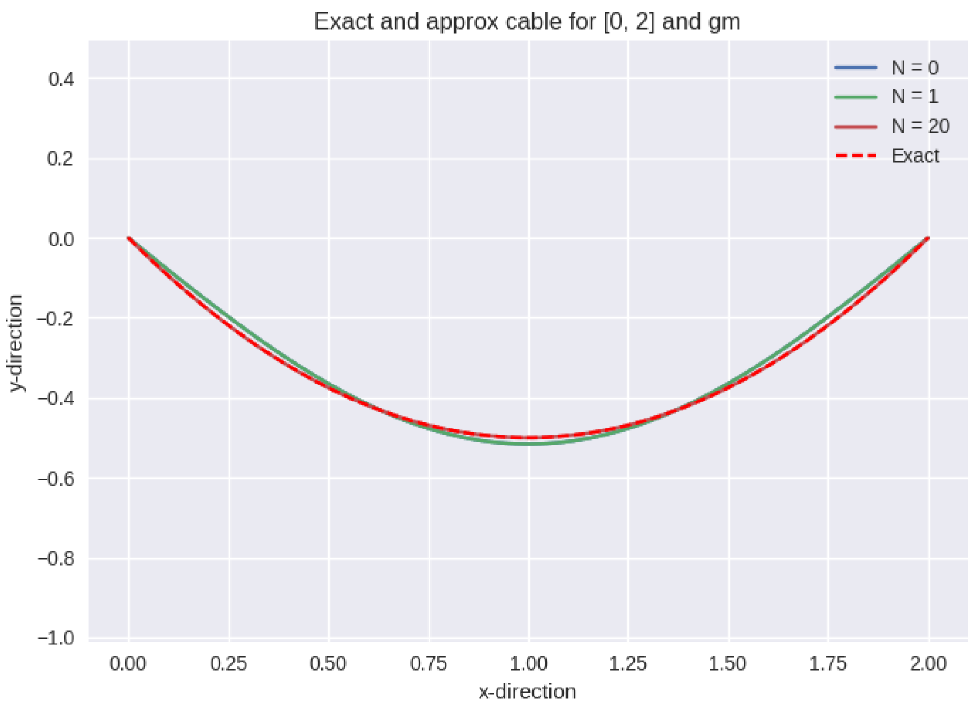

Figure 4: Here you can see the approximations when we have 1, 2 and 21 sine functions. The dashed line is the exact solution,

3.3 Finite element method

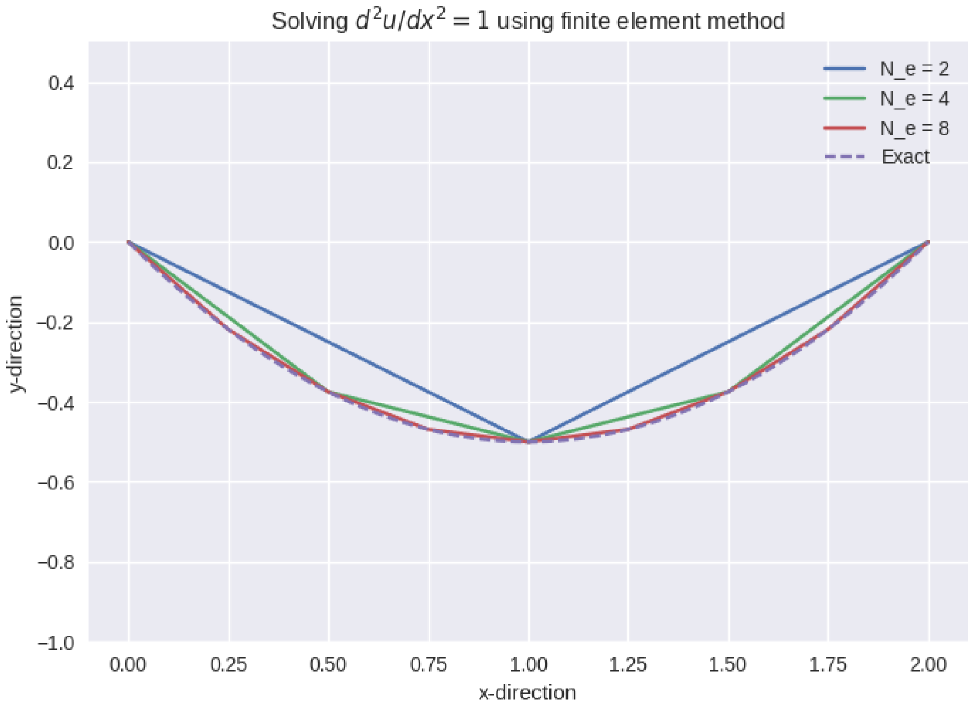

Figure 5: Here you can see the approximation using 2, 4 and 8 P1 linear elements. The exact solution from the theory part is plotted as a dashed line

4 Discussion and conclusion

For both the least square method and the Galerkin method we found some mysterious coefficient ratios. If we take a closer look at A_i (which is a diagonal matrix, hence not two indexes) and b_i for the two methods, the results might be more obvious. For the least square method, we will find C_i on the form

$$C_i = A_i^{-1} b_i = c \cdot (2i+1)^{-3}$$

where c is a constant, which gives $C_i/C_{i-1} = (2i-1)/(2i+1)$. In our case we also get exactly the same result with Galerkin method as Least Square method, but this does not apply in general.

We found the absolute error at the midpoint to be exactly the same for both of the methods. Apparently the methods give the same result in this case, but I am not sure why. The plots look very similar, but from the coefficient tables we can see that the approximations not are identical.

We have seen that the approximation when we only look at half the cable is good for both the least square method and the Galerkin method when we use "odd" sines, and it is getting better when we increase the number of sine functions, as expected. On the other hand when we approximate half of the cable with all sine functions, the approximation is not so good, and tends to be worse for increasing N. As mentioned in the theory part, this happens because the function space does not meet the boundary conditions. Anyway we can see that a function space with all sines gives a good approximation when we study all the domain, since the functions now meet the boundary conditions.