Mek4250 - Mandatory assignment 1
Andreas Thune
16.03.2016

Exercise 1

a) The H^p norm of a function $u(x,y)$ of two variables on $\Omega = (0,1)^2$ is defined as follows:

$$||u||^2_{H^p(\Omega)} = \sum_{i=0}^{p}\sum_{j=0}^{i}\binom{i}{j}||\frac{\partial^i u}{\partial^j x\partial^{i-j}y}||^2_{L^2(\Omega)} \tag{1}$$

In our case $u(x,y) = sin(k\pi x)cos(l\pi y)$. We easily see that the derivative of this function can be expressed with the following formula:

$$\frac{\partial^i u(x,y)}{\partial^j x\partial^i y} = (k\pi)^j(l\pi)^i f_j(k\pi x)g_j(l\pi y) \tag{2}$$

were f_j is the j-th derivative of $sin(x)$ and g_i is the i-th derivative of $cos(y)$. Now lets look at the L^2 norm of (2).

$$\begin{aligned}||\frac{\partial^i u(x,y)}{\partial^j x\partial^i y}||^2_{L^2(\Omega)} &= (k\pi)^{2j}(l\pi)^{2i}\int\int_\Omega f_j(k\pi x)^2 g_i(l\pi y)^2 dxdy\\ &= (k\pi)^{2j}(l\pi)^{2i}\int_0^1 f_j(k\pi x)^2dx\int_0^1 g_i(l\pi y)^2dy\end{aligned}$$

Since f_j^2 and g_i^2 are:

$$f_j(x)^2 = \begin{cases} sin^2(x) & \text{j even}\\ cos^2(x) & \text{j odd}\end{cases}$$

and

$$g_i(y)^2 = \begin{cases} cos^2(y) & \text{i even}\\ sin^2(y) & \text{i odd}\end{cases}$$

and since

$$\int_0^1 sin^2(l\pi y)dy = \int_0^1 cos^2(l\pi y)dy = \frac{1}{2}$$

we get the following expression for the square of the L^2 norm of a general derivative of u:

$$||\frac{\partial^i u(x,y)}{\partial^j x\partial^i y}||^2_{L^2(\Omega)} = \frac{1}{4}(k\pi)^{2j}(l\pi)^{2i}$$

If we plug this into (1), we get

$$||u||^2_{H^p(\Omega)} = \frac{1}{4}\sum_{i=0}^{p}\sum_{j=0}^{i}\binom{i}{j}(k\pi)^{2j}(l\pi)^{2(i-j)}$$

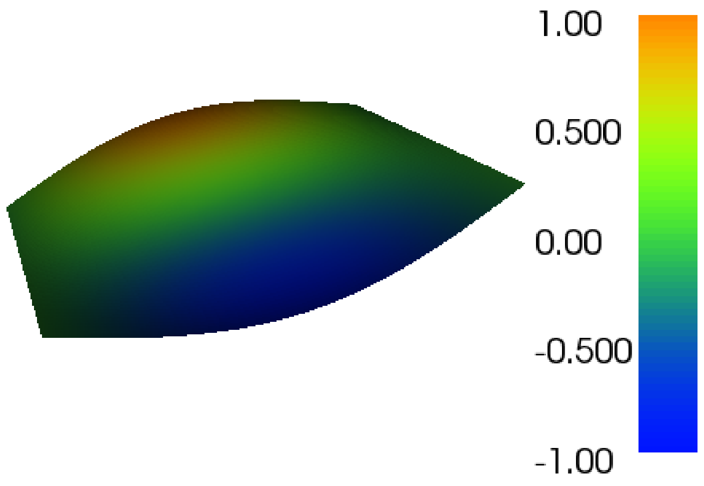

Figure 1: Plot of numerical solution u for $k = l = 1$ and $h = 64$. See that everything lies between -1 and 1 as one would expect from a product between sine and cosine functions. Notice also that the solution is smooth

b) To solve the equation numerically I need to derive the variational form of the equation. Since our boundary conditions are homogeneous Neumann and Dirichlet, I will ignore them completely. Our equation is then:

$$-\triangle u = f$$

Multiply this with testfunction v, and integrate over $\Omega = (0,1)^2$:

$$-\int_\Omega \triangle uvdx = \int_\Omega fvdx$$
$$\Longleftrightarrow \int_\Omega \triangledown u \cdot \triangledown vdx = \int_\Omega fvdx$$

This is our variatonal form, that we write in FEniCS. To get it we use partial integration. We also need to find $f = -\triangle u$. Since we know u this is simple.

$$f = -\triangle u = -\triangle sin(k\pi x)cos(l\pi y)$$
$$= ((k\pi)^2 + (l\pi)^2)sin(k\pi x)cos(l\pi y)$$

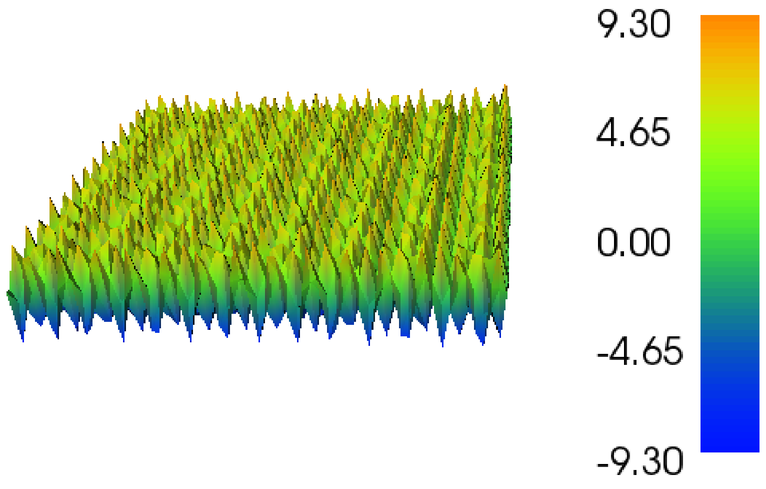

Figure 2: Plot of numerical solution u for $k = l = 100$ and $h = 64$. Now the values of u exceed both -1 and 1, and it also does not look smooth. This is due to aliasing.

In this exercise we were supposed to find the L^2 and H^1 error when solving our equation numerical for both elements of order 1 and 2, and for different values of k and l. To make it short I will only present my results for $k = 1$. The code for this exercise is added at the end, and the results are added at the end of the code. What we see from the error, is that it is small for k and l small, but big when k or l get big. This is especially true for the H^1 error. The reason for this is that big k and l means that our exact solution will "go up and down" very quickly, and since we are using a quite coarse mesh, we will get aliasing. To illustrate this I have added the plots of our numerical solutions for $k = l = 1$ and $k = l = 100$ when $\frac{1}{h} = 64$ and using $P1$ elements.

c) Since I implemented the norm in a), I have measured the convergence rate using all equations and errors, and also the convergence rate for each problem separately. This means I have tried to fit

$$\frac{||u - u_h||_{H^p}}{||u||_{H^{q+1}}} \quad \text{to} \quad Ch^\alpha \tag{3}$$

where u is exact solution to the equation for all l and k, and u_h is the same numerically. I also calculated convergence rate while holding k and l fixed, using the following:

$$||u - u_h||_{H^p} \text{ to } Ch^\alpha \tag{4}$$

Using $P1$ elements I got the following result when estimating convergence for (3):

	Convergence rate	Constant
L^2	1.776496	0.041508
H^1	0.911748	0.181145

For $P2$ the result was:

	Convergence rate	Constant
L^2	2.570886	0.000820
H^1	1.491997	0.003164

Theoretically we would expect

$$\frac{||u - u_h||_p}{||u||_{q+1}} \leq Ch^{q+1-p}$$

In the expression above q represent the order of the element used in our finite element method. We see that our numerical estimates of the convergence rate in all cases are lower than what we would expect. This could indeed be a result of the aliasing we get because of our coarse mesh. As I said I also checked convergence for (k, l) pairs separately, and this is the L^2 convergence rates I got for $P1$ elements :

	$k = 1$	$k = 10$	$k = 100$
$l = 1$	1.980241	1.665838	2.253570
$l = 10$	1.663255	1.190830	1.298707
$l = 100$	2.265711	1.367577	2.302735

One remark to these numbers, are that even though the convergence rate is good for k and l around 100, the constant is in these cases really big, and it is therefore simple to get good convergence. The bad convergence occurs around k and l equal to 10. This is probably because this is a borderline case where our mesh is almost good enough to avoid aliasing. The rest of the results is found in the code.

Exercise 2

a) Assume our solution is on the form $u(x, y) = X(x)Y(y)$. If we plug this into our equation and assume that both X and Y are nonzero, we get:

$$-\mu(X''Y + XY'') + X'Y = 0 \iff \frac{-\mu X'' + X'}{X} - \mu\frac{Y''}{Y} = 0$$
$$\iff -\mu X'' + X' = \lambda X \text{ and } \mu Y'' = \lambda Y$$

Now lets look at the boundary conditions, starting withe the Dirichlet conditions:

$$u(0,y) = 0 \iff X(0)Y(y) = 0 \Rightarrow X(0) = 0$$

Since $Y(y) = 0$ would be a contradiction.

$$u(1,y) = 1 \iff X(1)Y(y) = 1 \Rightarrow Y(y) = 1/X(1)$$

This means that $Y(y)$ is a constant. This does not contradict our Neumann boundary conditions, since they say that the y-derivative is zero at $y = 0$ and $y = 1$. This means that our PDE is really an ODE on the form:

$$\begin{cases} -\mu X''(x) + X'(x) = 0 \\ X(0) = 0,\ X(1) = 1 \end{cases}$$

This gives us:

$$\mu X'(x) = X(x) + C \iff (X(x)e^{\frac{-x}{\mu}})' = Ce^{\frac{-x}{\mu}} \tag{5}$$

$$\iff X(x) = C' + De^{\frac{x}{\mu}} \tag{6}$$

Our boundary terms yields

$$C' = -D$$
$$C' + De^{\frac{1}{\mu}} = 1$$

The solution to this system is:

$$C' = \frac{1}{1 - e^{\frac{1}{\mu}}}$$
$$D = -\frac{1}{1 - e^{\frac{1}{\mu}}}$$

Putting this into the general solution gives us:

$$X(x) = \frac{1 - e^{\frac{x}{\mu}}}{1 - e^{\frac{1}{\mu}}}$$

and

$$u(x,y) = \frac{1 - e^{\frac{x}{\mu}}}{1 - e^{\frac{1}{\mu}}}$$

b)To solve the equation numerically I need to derive the variational form of the equation. Since our boundary conditions are a mix of homogeneous

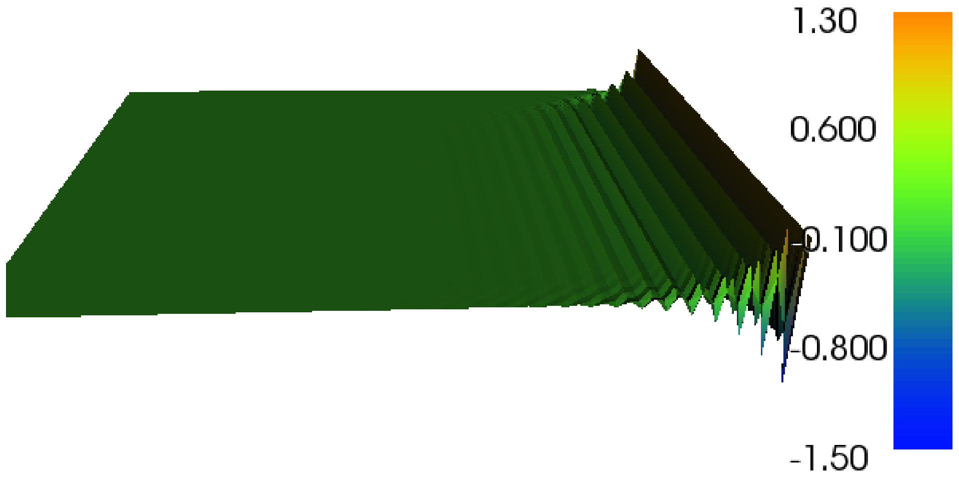

Figure 3: Plot of numerical solution u for $\mu = 0.001$ and $h = 64$, using normal Galerkin method. Notice oscillations close to $x = 1$.

Neumann and non-homogeneous Dirichlet, I will ignore them completely. Our equation is then:

$$-\mu \triangle u + u_x = f$$

Multiply this with testfunction v, and integrate over $\Omega = (0,1)^2$

$$\begin{aligned} &\int_\Omega (-\mu \triangle u + u_x) v dx = \int_\Omega f v dx \\ \iff &\int_\Omega \bigtriangledown u \cdot \bigtriangledown v dx + \int_\Omega u_x v dx = \int_\Omega f v dx \end{aligned}$$

This is our variatonal form, that we write in FEniCS. To get it we use partial integration.

The code is attached at the end together with the results. Could add, that to be able to define the exact solution for small μ, I did the following approximation trick:

$$u(x,y) = \frac{1 - e^{\frac{x}{\mu}}}{1 - e^{\frac{1}{\mu}}} = \frac{e^{\frac{1}{\mu}}}{e^{\frac{1}{\mu}}} \frac{e^{-\frac{1}{\mu}} - e^{\frac{x-1}{\mu}}}{e^{-\frac{1}{\mu}} - 1} = \frac{e^{-\frac{1}{\mu}} - e^{\frac{x-1}{\mu}}}{e^{-\frac{1}{\mu}} - 1} \approx e^{\frac{x-1}{\mu}}$$

This makes sense since $e^{-\frac{1}{\mu}} \approx 0$ when μ is small.

As we see from the error results we get big errors in both norms when μ is small. To illustrate what happens I have plotted the numerical solution for $\mu = 0.001$ and $h = 64$. What we observe is oscillations close to $x = 1$, where we now that the exact solution grows very fast.

c) Lets look at the convergence rate α and constant C, I get from L^2 and H^1 for different μ.

	$\mu = 1$	$\mu = 0.1$	$\mu = 0.01$	$\mu = 0.001$	$\mu = 0.0001$
L^2 α	1.999763	1.975224	1.467166	1.299193	1.802562
L^2 C	0.044866	0.735446	3.339322	12.052507	301.737979
H^1 α	0.999856	0.978303	0.462191	0.184035	0.698340
H^1 C	0.212219	4.229330	19.327090	44.796474	989.689148

As we see from these results, we get what we would expect for big μ, but when we let μ get smaller, the convergence rate gets worse and the constant gets big. See that the convergence gets better again for the last μ value, but this is because error is so big that the convergence rate almost doesn't matter. The problem is that we get big oscillation near $x = 1$. We see this by plotting, or that the $H1$ error gets big.

d) Want to implement the streamline upwind Petrov-Galerkin method. The notes describe how to do this when we use first order Lagrange elements. The only thing we have to add to our variational form is the term:

$$\beta \int_{\Omega} (v \cdot \bigtriangledown u)(v \cdot \bigtriangledown w) dx = \beta \int_{\Omega} u_x w_x dx$$

An important question is how to choose β. I choose to use $\beta \approx h$. This seemed to work. However, when I use a changing β, I can no longer use the sd norm to calculate the convergence rate of the method.

As above the code is attached below, and the error and convergence of the method is in the code as well. See that the L^2 error and H^1 error is much smaller for SUPG method then for the regular Galerkin method when μ is small. However, the convergence rate for L^2 is now a lot worse, and for H^1, the H^1 norm actually increases. To illustrate how much better SUPG is I have added a plot of the solution to the same problem I plotted above using SUPG. This looks a lot closer to the true solution, then what what we got earlier, i.e. no numerical artifacts.

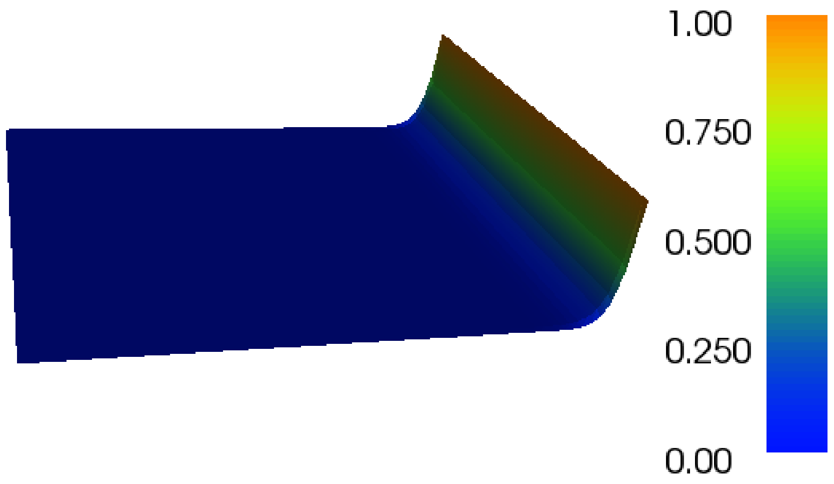

Figure 4: Plot of numerical solution u for $\mu = 0.001$ and $h = 64$ using the SUPG method. Notice that the graph is smooth, even at $x = 1$.