Assignment 2

Ludwig Tranheden

TMA372

2016/02/28

Contents

1 Problem 1 **2**
1.1 a) . . . 2
1.1.1 i) . . . 2
1.1.2 ii) . . . 2
1.2 b) . . . 3
1.3 c) . . . 3

2 Problem 2 - Application **4**
2.1 Real world application . . . 4
2.2 Mathematical model . . . 5
2.2.1 Another posteriori estimate . . . 7
2.3 Analytic aspects . . . 7
2.3.1 Analytic functions . . . 7
2.3.2 Analytic solution . . . 7
2.3.3 Simplified model . . . 8

3 Appendix A - Code **8**

4 Appendix B - Functions **9**

1 Problem 1

In this problem we consider the heat equation, that is equation 1.

$$\begin{cases} u_t - \Delta u = 0 & x \in \Omega,\ t > 0 \\ u = 0 & x \in \partial\Omega,\ t > 0 \\ u(x,0) = u_0 & x \in \Omega \end{cases} \tag{1}$$

1.1 a)

Prove the two following stability estimates:

i) $\|u\|^2 + \int_0^t \|\nabla u\|^2 ds \leq \|u_0\|^2$

ii) $\|\Delta u\| \leq \frac{1}{t}\|u_0\|$

1.1.1 i)

We begin by multiplying the DE with u and integrating over Ω obtaining.

$$\int_\Omega u_t u dx - \int_\Omega u \Delta u dx = 0$$

Using Greens formula, the boundary conditions, substituting t with s and integrating yields.

$$\frac{1}{2}\int_0^t \frac{d}{ds}\|u\|^2 ds \ + \ \int_0^t \|\nabla u\|^2 ds \Longrightarrow$$

$$\|u\|^2 + 2\int_0^t \|\nabla u\|^2 ds = \|u_0\|^2$$

And we have arrived at the result because the integral is positive, so it will give a positive contribution to the left hand side. Dividing it by 2 yields the inequality.

1.1.2 ii)

This time we begin with multiplying the DE with $t^2\Delta^2 u$ and in the same fashion integrate over Ω to obtain.

$$\frac{1}{2}\frac{d}{dt}(t^2\|\Delta u\|^2) - \int_\Omega t^2 \Delta^2 u \Delta u dx = t\|\Delta u\|^2$$

Next we substitute t for s and integrate.

$$\frac{1}{2}\int_0^t \frac{d}{ds}(s^2\|\Delta u\|^2) ds - \int_0^t s^2 \left(\int_\Omega \Delta^2 u \Delta u dx\right) ds = \int_0^t s\|\Delta u\|^2 ds \leq \frac{1}{4}\|u_0\|^2$$

The last inequality comes from multiplying the DE with $-t\Delta u$ and using the result of part i. Integrating we arrive at the wanted result.

$$\|\Delta u\| \leq \frac{1}{t}\|u_0\|$$

1.2 b)

In this task we want to see how the stability estimates are effected by substituting the operater $-\Delta u$ with $Au := -u_{xx} - 4u_{yy}$. The equation now takes the form of equation 2.

$$u_t + Au = 0 \qquad x \in \Omega \tag{2}$$

Repeating the procedure of part i of 1a we first acquire.

$$\frac{1}{2}\frac{d}{dt}\|u\|^2 + \int_\Omega uAudx = 0$$

Integrating we get the new estimate.

$$\|u\|^2 + 2\int_0^t (u, Au)ds = \|u_0\|^2$$

For the next part we multiply equation 2 with t^2A^2u simply repeating the procedure of part ii in 1a. That includes multiplying the equation with tAu to obtain a "supporting" estimate. Our result is.

$$\|Au\| \leq \frac{1}{t}\|u_0\|$$

1.3 c)

Now we are to solve the PDE by using Fourier series with $\Omega = [0, 1]$. To do this we use the technique called separation of variables, i.e $u(x, t) = T(t)X(x)$. This gives us two ordinary differential equations, 3 & 4.

$$X^{''}(x) = \lambda X(x) \tag{3}$$

$$T^{'}(t) = \lambda T(t) \tag{4}$$

Solving them gives us.

$$X(x) = c_1 sin(\sqrt{-\lambda}x) + c_2 cos(\sqrt{-\lambda}x)$$

$$T(t) = c_3 e^{\lambda t}$$

Using the boundary conditions and discarding trivial solutions we get.

$$u(x, t) = 2ic_1c_3 sin(n\pi x)e^{-n^2\pi^2 t} = csin(n\pi x)e^{-n^2\pi^2 t}$$

And the fourier series as.

$$u(x, t) = \sum_{n=1}^{\infty} c_n sin(n\pi x)e^{-n^2\pi^2 t}$$

Using the initial value condition we can acquire the constant c_n.

$$u(x, 0) = \sum_{n=1}^{\infty} c_n sin(n\pi x) = u_0(x)$$

So c_n is given by the sine series coefficients of $u_0(x)$. Regarding the solution we notice that the x-part is bounded and periodic on Ω and the t part decay very quickly.

To acquire the smoothing estimate we calculate $t\|\Delta u(x,t)\|$.

$$t\|\Delta u(x,t)\| = \|\sum_{n=1}^{\infty} c_n t\pi^2 n^2 sin(n\pi x)e^{-n^2\pi^2 t}\|$$

Using the fact that $xe^{-x} < 1$, $\forall x > 0$ gives us.

$$t\|\Delta u(x,t)\| < \|\sum_{n=1}^{\infty} c_n sin(n\pi x)\| = \|u_0\| \implies$$

$$\|\Delta u\| < \frac{\|u_0\|}{t}$$

And we are done.

2 Problem 2 - Application

The application to be investigated is the fluid flow through a 2d nuzzle.

2.1 Real world application

The flow through a, for example, nuzzle have a wide spectrum of applications. Usually it's purpose is to somehow control the velocity and direction of the flow of the fluid. The fluids velocity increases but on the expense of it's pressure energy. A simple example is a water hose or a sprinkler. When the cross sectional area of the nuzzle decreases the velocity of the fluid, in this case water, increases. One particular type of nuzzle is the "A de Laval nozzle", a tube pinched in the middle. It's application is to accelerate a hot pressurized gas passing through to a higher velocity. The "A de Laval nozzle" is for example used in some types of rocket engine nozzles.

2.2 Mathematical model

Imagine that we have the nuzzle domain illustrated in figure 1. The velocity of

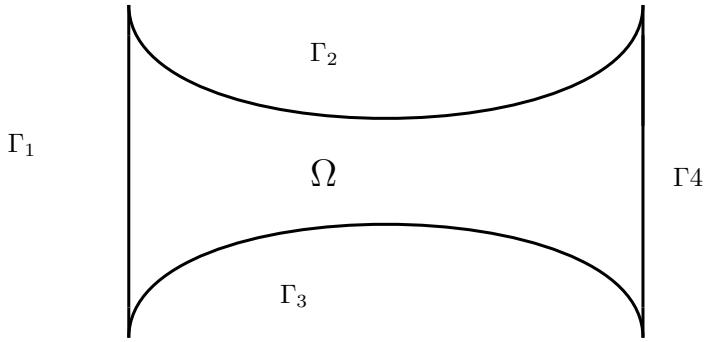

Figure 1: Nuzzle domain (principal)

the fluid is described by $u = (u_1, u_2)$. Using the assumption that the fluid is incompressible and irrotational we can express u by the potential ϕ. Combining this with the incompressibility equation we acquire a Laplace equation for ϕ: $\nabla(\nabla\phi) = \Delta\phi = 0$. We approximate the differential equation by setting up suitable boundary conditions. We first assume that the fluid enters the nozzle with a velocity of 3 (Unit not relevant) and that there is not any velocity in the normal direction of the horizontal walls. Further that the potential is zero along the right wall. These assumptions yields the PDE show in equation 5.

$$\begin{cases} \Delta\phi = 0 & x \in \Omega \\ \frac{d\phi}{dn} = 0 & x \in \Gamma_2 \cup \Gamma_3 \\ \frac{d\phi}{dn} = 3 & x \in \Gamma_1 \\ \phi = 0 & x \in \Gamma_4 \end{cases} \tag{5}$$

We solve this using the 2D cG(1) method implemented in matlab. Using greens we get the following FEM formulation.

$$\sum_{j=1}^{N} \xi_j \Big(\int_\Omega \nabla\varphi_j \nabla\varphi_i dx \Big) = 3 \int_{\Gamma_1} \varphi_i ds \qquad \Longrightarrow$$

$$S\xi = 0 * b + r$$

Where i ranges from 1 to N and the φ are the standard basis functions. One can deduce a posterior error estimate assuming that because the boundary Γ_1 is a straight line the integral over the difference of a function and it's interpolation, in this case $e - \pi_h e$, is zero. Following the calculations of Eriksson et. al. (1996, p.343-346) and "A Posteriori Based Adaptive Mesh Refinement"[1], in my

[1] https://www.cambro.umu.se/access/content/group/69cbe5bd-67a8-41d8-9bc0-be47f2291b61/r5fzlto0.pdf

case with setting $f \equiv 0$, you arrive at a estimate similar to the one used in matlabs built in function pdejmps that calculates the errors over all triangles. The estimates derived is (the latter is the one matlab uses):

$$\|\nabla(\phi - \phi_h)\|^2_{L^2_\Omega} \leq C \sum_k h_k^2 \|\Delta\phi_h\|_{L^2_k} + \frac{1}{2} h_k \| \left[n \nabla \phi_h \right] \|^2_{L^2_{\partial k}}$$

and

$$\eta_k = h_k^2 \|\Delta\phi_h\|^2_k + \frac{1}{2} h_k \| \left[n \nabla \phi_h \right] \|^2_{\partial k}$$

The first equation is the overall error while η_k is the contribution element k gives to the total error. $h_k = diam(k)$ and for two neighboring elements k^+ and k^- sharing edge E and with outward unit normal pointing from k^+ to k^-: $\left[n \nabla \phi_h \right] = (\nabla \phi_h|_{k^-} - \nabla \phi_h|_{k^+}) n$. But to sum over triangles rather then edges we redistribute the jump between the two elements sharing E, hence the factor $\frac{1}{2}$.

So looping throw the calculations refining the mesh when the desired tolerance for a triangle is not fulfilled will produce a acceptable solution. Refining the mesh 5 times yields the meshes displayed below together with the plot of the potential and the gradient plot of the velocity. As expected the velocity increases and we loose pressure energy when the, in the 2D case, cross sectional length decreases. My matlab solution for the problem is displayed in Appendix A.

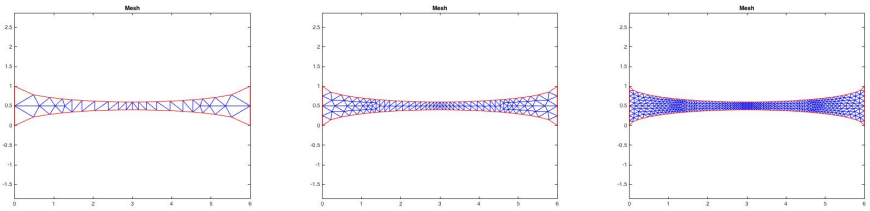

Figure 2: Mesh 1,2 and 5

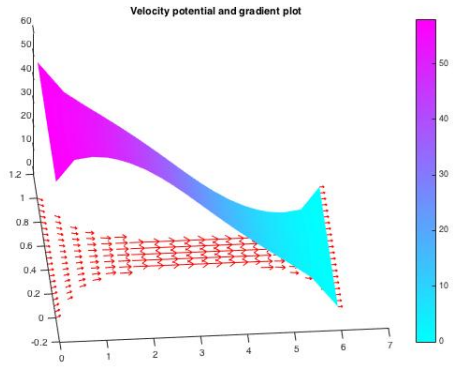

Figure 3: Velocity potential and gradient plot

2.2.1 Another posteriori estimate

One can derive another posteriori estimate by using the dual problem and the fact that the error is zero at he boundary. With the help of galerkin orthogonality, greens and the interpolation error you arrive at the estimate.

$$\|e\| \leq C_\Omega \|h^2 \Delta_h \phi_h\|$$

Where Δ_h is the discrete laplacian.

2.3 Analytic aspects

2.3.1 Analytic functions

A interesting aspect of the laplace equation is it's role in the analysis of analytic functions. Because a condition for a function to be analytic is that it satisfies Cauchy-Riemann equations. Which directly leads to that both the real and imaginary part satisfies the laplace equation.

2.3.2 Analytic solution

A analytic solition to $\Delta\phi = 0$ for $x \neq 0$ if $\phi : \mathbb{R}^2 \to \mathbb{R}$ is $\phi(x) = log(|x|^{-1})$.

$$\frac{\partial^2}{\partial x_1^2}\phi = -\frac{1}{|x|^2} + \frac{2x_1^2}{|x|^4}$$

$$\frac{\partial^2}{\partial x_2^2}\phi = -\frac{1}{|x|^2} + \frac{2x_2^2}{|x|^4} \implies$$

$$\Delta\phi = \frac{\partial^2}{\partial x_1^2}\phi + \frac{\partial^2}{\partial x_2^2}\phi = 0$$

2.3.3 Simplified model

To be able to solve the DE we simplify the geometry to a rectangle and alter the boundary conditions slightly.

$$\begin{cases} \Delta\phi = 0 \qquad 0 \leq x \leq L, 0 \leq y \leq H \\ \phi_y(x, H) = \phi_y(x, 0) = 0 \\ \phi_x(0, y) = 3 \\ \phi(L, y) = 0 \end{cases} \tag{6}$$

A very simple, and not very interesting solution to this DE is $\phi(x, y) = 3x - 3L$. One can also solve equation 6 with some further modifications to the boundary conditions by a rather involved Fourier series.

3 Appendix A - Code

Listing 1: FEM

```
load Nozzleinfo
kappa = 0.01;
ITER = 5; %Number of iterations
h = 20; %Initial hmax
dl = decsg(gd,sf,ns);   %Geometry
[p,e,t] = initmesh(dl,'hmax',h); %Create initial mesh


for i=1:ITER
    figure(i); clf
    pdemesh(p,e,t)
    title('Mesh')
    axis equal
    hold on
    pause(1)

    % Assemble
    Nt = size(t,2); % #Triangles
    Np = size(p,2); % # Nodal points
    Ne = size(e,2); % #Boundaryedges
    A = sparse(Np,Np); % Inititate Stiffnessmatrix
    b = zeros(Np,1); % Rigth hand side
    for i=1:Nt
        n = t(1:3,i); % Cornerpoints triangle i
        x = p(1,n); % x coordinates for corner
        y = p(2,n); % y coordinates for corner
        dy = [y(2)-y(3); y(3)-y(1); y(1)-y(2)]; %Distance between y-coordinate corners
        dx = [x(3)-x(2); x(1)-x(3); x(2)-x(1)]; %Distace between x-coordinate corners
        area = Mypolyarea(x,y); %Area of current triangle
        A(n,n) = A(n,n) + (dx*dx'+dy*dy')/4/area;
    end
    r = zeros(Np,1);
```

```
    for i=1:size(e,2)
        if(e(5,i)==2)   %Neumann condition boundary 1
            n = e(1:2,i);
            x = p(1,n);
            y = p(2,n);
            length_i = sqrt((x(1)-x(2))^2+(y(1)-y(2))^2);
            r(n) = r(n) + 3*length_i/2;
        elseif (e(5,i)==1)  %Dirichlet condition boundary 4
            n = e(1:2,i);
            A(n,n) = 1e6;
        end
    end
    b = r+b;

    U = A\b; %Solve
    errf = pdejmps(p,t,-1,0,0,U,1,1,1); %Error
    tol = kappa*max(errf); %Tolerans
    elements = find(errf>tol); %Find elements with large error contribution
    [p,e,t] = refinemesh(dl,p,e,t,elements); %Refine the mesh
end
S=sprintf('Calculations finished');
disp(S)
```

4 Appendix B - Functions

Listing 2: Mypolyarea

```
%Returns area of the polygon created by connecting the
% (x,y) coordinates of the input using a formula based on greens theorem
function Area = Mypolyarea(x,y)
if length(x)~=length(y)
    error('Error. \nInvalid input argument, lengths must be equal %s.',class(x,y))
end
Area = 0;
for i=1:length(x)
    if i == length(x)
        k = 1;
    else
        k = i+1;
    end
    Area = Area + (x(i)*y(k) - x(k)*y(i));
end
Area = abs(Area)/2;
```