Two dimensional heat equation

Deep Ray, Ritesh Kumar, Praveen. C, Mythily Ramaswamy, J.-P. Raymond

IFCAM Summer School on Numerics and Control of PDE
22 July - 2 August 2013
IISc, Bangalore
http://praveen.cfdlab.net/teaching/control2013

1 The PDE model

Let $z = z(x, y, t)$ denote the temperature. The shifted 2-D heat equation is given by

$$z_t = \mu\Delta z + \omega z, \quad (x, y) \in \Omega = (0, 1) \times (0, 1), \qquad t \in [0, T]$$

with boundary conditions, see figure (1)

$$z(x, 0, t) = z(x, 1, t) = 0, \quad z(1, y, t) = u(y, t), \qquad \frac{\partial z}{\partial x}(0, y, t) = 0$$

and initial condition

$$z(x, y, 0) = z_0(x, y)$$

Here $\omega \geq 0$ and $\mu > 0$. Let us denote the Dirichlet part of the boundary by Γ_D

$$\Gamma_D = \{y = 0\} \cup \{y = 1\} \cup \{x = 1\}$$

the Neumann part as

$$\Gamma_N = \{x = 0\}$$

and the part on which the control is applied as

$$\Gamma_c = \{x = 1\}$$

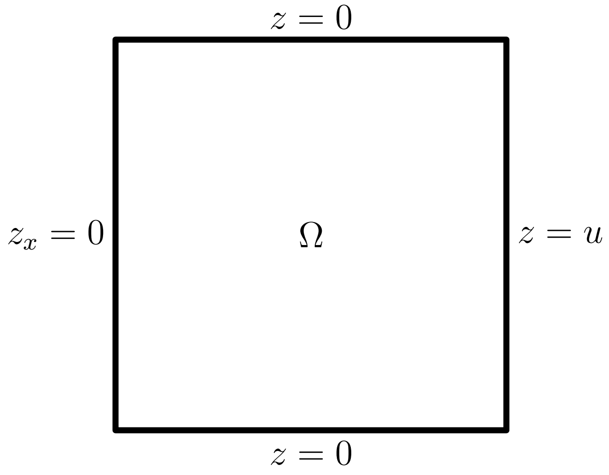

Figure 1: Problem definition

1.1 Observations

We will measure an average value of the temperature on strips along the left vertical boundary

$$I_i = [a_i, b_i], \qquad i = 1, 2, 3$$

as shown in figure (2). Thus the observations are

$$y_i(t) = \frac{1}{b_i - a_i} \int_{a_i}^{b_i} z(0, y, t) \mathrm{d}y, \qquad i = 1, 2, 3 \tag{1}$$

Note: Do not confuse the observation y_i with the spatial coordinate y.

1.2 Weak formulation

We assume $z_0 \in L^2(\Omega)$. We wish to find $z \in L^2(0, T; H^1(\Omega))$ such that

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega z(t)\phi = -\mu \int_\Omega \nabla z \cdot \nabla \phi + \omega \int_\Omega z\phi, \quad \forall \phi \in H^1_{\Gamma_D}(\Omega)$$

$$z(x, 0, t) = z(x, 1, t) = 0, \quad z(1, y, t) = u(y, t)$$

$$(z(0), \phi)_{L^2} = (z_0, \phi)_{L^2}$$

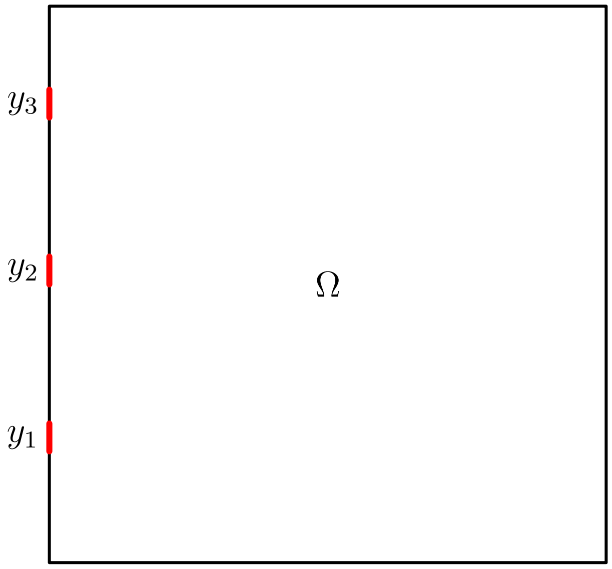

Figure 2: Observations

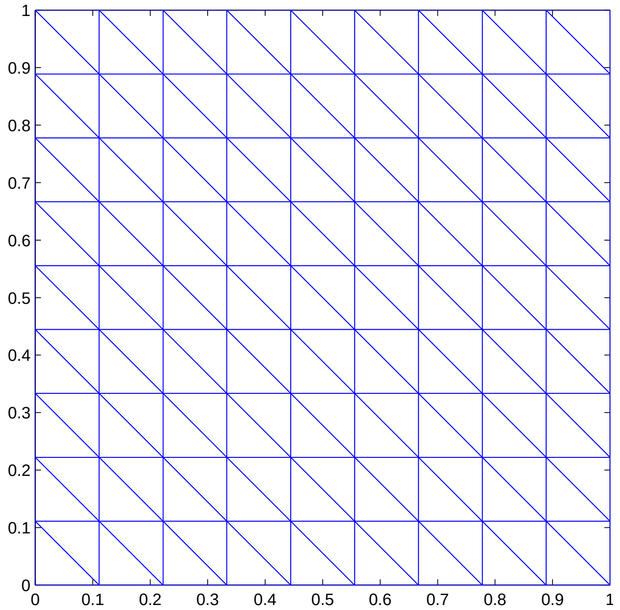

Figure 3: Example of a finite element mesh

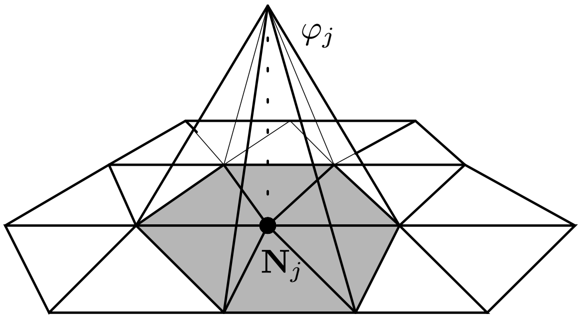

Figure 4: Piecewise affine basis functions

2 FEM approximation

Consider a division of Ω into disjoint triangles as shown in figure (3). We will assume that the vertices of the mesh are numbered in some manner. Let us define the following sets of vertices

$$\begin{aligned} N_c &= \text{vertices on } \Gamma_c \\ N_d &= \text{vertices on } \{y=0\} \cup \{y=1\} \\ N_f &= \text{remaining vertices (unknown degrees of freedom)} \end{aligned}$$

For each vertex i, define the piecewise affine functions $\phi_i(x,y)$, see figure (4), with the property that

$$\phi_i(x_j, y_j) = \delta_{ij}$$

Then the finite element solution is of the form

$$z(x,y,t) = \sum_{j \in N_f} z_j(t)\phi_j(x,y) + \sum_{j \in N_c} u_j(t)\phi_j(x,y), \qquad z(x_i, y_i, t) = z_i(t), \quad \forall i \in N_f$$

This function already satisfies the Dirichlet boundary conditions

$$z(x,0,t) = z(x,1,t) = 0, \qquad z(1,y,t) = \sum_{j \in N_c} u_j(t)\phi_j(1,y)$$

We will take the control to be of the form

$$u(y,t) = v(t)\sin(\pi y) \qquad \Longrightarrow \qquad u_j(t) = v(t)\sin(\pi y_j)$$

This is a one dimensional control which is compatible with the other boundary control

$$u(0,t) = u(1,t) = 0$$

Then the finite element solution is of the form

$$z(x,y,t) = \sum_{j \in N_f} z_j(t)\phi_j(x,y) + v(t) \sum_{j \in N_c} \sin(\pi y_j)\phi_j(x,y)$$

The approximate weak formulation is given as

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega z(t)\phi_i = -\mu \int_\Omega \nabla z \cdot \nabla \phi_i + \omega \int_\Omega z\phi_i, \quad \forall i \in N_f$$

i.e.,

$$\begin{aligned} & \sum_{j \in N_f} \frac{\mathrm{d}z_j}{\mathrm{d}t} \int_\Omega \phi_j\phi_i + \frac{\mathrm{d}v}{\mathrm{d}t} \sum_{j \in N_c} \sin(\pi y_j) \int_\Omega \phi_j\phi_i \\ = \quad & -\mu \sum_{j \in N_f} z_j \int_\Omega \nabla\phi_j \cdot \nabla\phi_i - \mu v \sum_{j \in N_c} \sin(\pi y_j) \int_\Omega \nabla\phi_j \cdot \nabla\phi_i \\ & +\omega \sum_{j \in N_f} z_j \int_\Omega \phi_j\phi_i + \omega v \sum_{j \in N_c} \sin(\pi y_j) \int_\Omega \phi_j\phi_i, \qquad \forall i \in N_f \end{aligned}$$

In order to simplify the presentation we will ignore the term containing $\frac{\mathrm{d}v}{\mathrm{d}t}$. This term vanishes if we use the trapezoidal rule for integration. Then we can write the FEM formulation as

$$\begin{aligned} & \sum_{j \in N_f} \frac{\mathrm{d}z_j}{\mathrm{d}t} \int_\Omega \phi_j\phi_i \\ = \quad & \sum_{j \in N_f} z_j \left[-\mu \int_\Omega \nabla\phi_j \cdot \nabla\phi_i + \omega \int_\Omega \phi_j\phi_i \right] + \\ & v \sum_{j \in N_c} \left[-\mu \sin(\pi y_j) \int_\Omega \nabla\phi_j \cdot \nabla\phi_i + \omega \sin(\pi y_j) \int_\Omega \phi_j\phi_i \right] \qquad \forall i \in N_f \end{aligned}$$

This can be written as a system of ordinary differential equations

$$\mathbf{M}\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}t} = \mathbf{A}\mathbf{z} + \mathbf{B}v$$

where $\mathbf{z}$ denotes all the unknown degrees of freedom corresponding to N_f.

2.1 Grid information

The grid information consists of following files

- `coordinates.dat`: contains x, y coordinates of all the vertices coordinates.dat

1	0	0
2	1	0
3	1.59	0
4	2	1
5	3	1.41
6	3	2
7	3	3
8	2	3
9	1	3
10	0	3
11	0	2
12	0	1
13	1	1
14	1	2
15	2	2

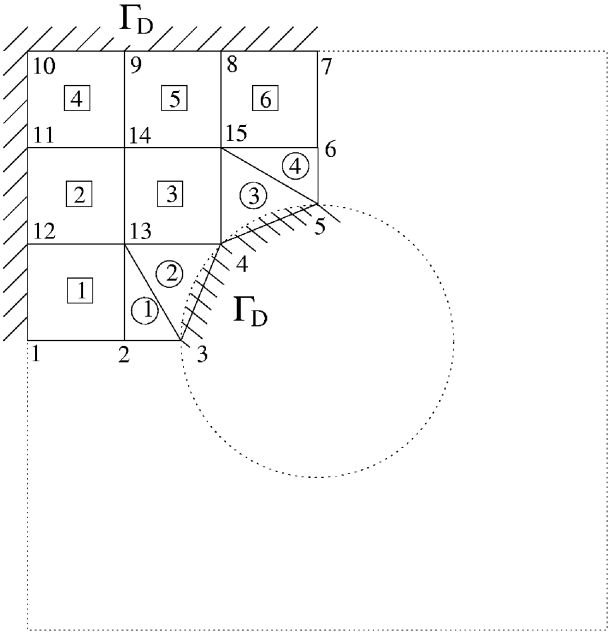

Figure 1. Example of a mesh.

Figure 5: Example of a mesh

- `elements3.dat`: contains verices forming each triangle
- `dirichlet.dat`: contains boundary edges on the dirichlet boundary
- `neumann.dat`: contains boundary edges on the neumann boundary

An example of this type of data is given in figures (5), (6). In each file, the first column is the serial number.

Note: In our current programs we use a mesh consisting of only triangles. Also, the serial numbers are not stored in the files.

The domain Ω is the unit square. The program `square.m` generates the mesh and creates the above four files. You have to specify the number of vertices on the side of the square. E.g., to have 11 points on each side, you do the following in matlab

```
>> square(11)
```

elements3.dat

1	2	3	13
2	3	4	13
3	4	5	15
4	5	6	15

elements4.dat

1	1	2	13	12
2	12	13	14	11
3	13	4	15	14
4	11	14	9	10
5	14	15	8	9
6	15	6	7	8

neumann.dat

1	5	6
2	6	7
3	1	2
4	2	3

dirichlet.dat

1	3	4
2	4	5
3	7	8
4	8	9
5	9	10
6	10	11
7	11	12
8	12	1

Figure 6: Structure of mesh files

When you run the above program, you can see a picture of the mesh. Examine the four files created by this program. For our actual computations, we will use 101 points on each side of the square. In this case, there are 5 edges which exactly cover each of the three observation zones.

2.2 Finite element assembly

The finite element basis functions have compact support. Hence we can compute the integrals

$$\int_\Omega \phi_i \phi_j \qquad \text{and} \qquad \int_\Omega \nabla\phi_i \cdot \nabla\phi_j$$

by adding the contributions from a small number of triangles. For example, the elements of the mass matrix can be computed as

$$\int_\Omega \phi_i \phi_j = \sum_{K\,:\,i,j\in K} \int_K \phi_i \phi_j$$

and similarly the stiffness matrix is computed as

$$\int_\Omega \nabla\phi_i \cdot \nabla\phi_j = \sum_{K\,:\,i,j\in K} \int_K \nabla\phi_i \cdot \nabla\phi_j$$

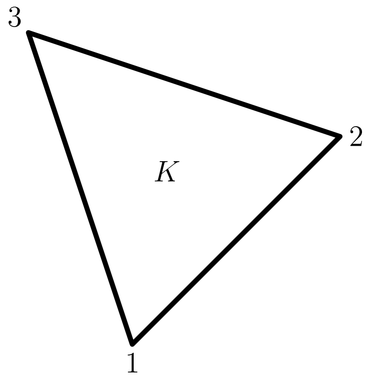

Figure 7: Triangle

The integrals on each triangle K will be evaluated exactly. For a triangle K with vertices labelled 1, 2, 3 as in figure (7), the local mass and stiffness matrices are given by

$$M^K = \begin{bmatrix} \int_K \phi_1\phi_1 & \int_K \phi_1\phi_2 & \int_\Omega \phi_1\phi_3 \\ \int_K \phi_2\phi_1 & \int_K \phi_2\phi_2 & \int_\Omega \phi_2\phi_3 \\ \int_K \phi_3\phi_1 & \int_K \phi_3\phi_2 & \int_\Omega \phi_3\phi_3 \end{bmatrix} = \frac{1}{24}\det\begin{bmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

$$A^K = \frac{1}{2}\det\begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix} GG^\top \qquad \text{where} \qquad G = \begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{bmatrix}^{-1} \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The numerical process can be summarized by the following algorithm:

Set $\mathbf{M} = 0$, $\mathbf{A} = 0$.
For each triangle K in the mesh

- Compute M^K and A^K
- Add the contributions from M^K into $\mathbf{M}$ and from A^K into $\mathbf{A}$

For the j'th triangle, the mapping between the local index to global index is given below

Local index	Global index
1	`elements3(j,1)`
2	`elements3(j,2)`
3	`elements3(j,3)`

```
For $j = 1, 2, \ldots,$ Ntri
    For $k = 1, 2, 3$
```

For $l = 1, 2, 3$

$$\begin{aligned}
\mathbf{A}(elements3(j,k), elements3(j,l)) \quad &+= \quad A^{K_j}(k,l) \\
\mathbf{M}(elements3(j,k), elements3(j,l)) \quad &+= \quad M^{K_j}(k,l)
\end{aligned}$$

More details on the assembly process can be found in this paper

Jochen Alberty, Carsten Carstensen and Stefan A. Funken: "Remarks around 50 lines of Matlab: short finite element implementation", Numerical Algorithms 20 (1999) 117-137
`http://math.tifrbng.res.in/~praveen/notes/control2013/acf.pdf`

3 Parameters and initial condition

Let us take

$$\mu = \frac{1}{50}, \qquad \omega = 0.4$$

Then the heat equation has one unstable eigenvalue given by

$$\lambda = -\frac{\pi^2}{40} + \omega = 0.015325988997$$

The corresponding eigenfunction is

$$\phi(x,y) = \cos(\pi x/2)\sin(\pi y)$$

which is shown in figure (8). We will use the unstable eigenfunction as the initial condition for the heat equation. This initial condition satisfies the following boundary conditions

$$\phi(x,0) = \phi(x,1) = \phi(1,y) = 0, \qquad \phi_x(0,y) = 0$$

4 Excercises I

The solution of heat equation is implemented in `heat2d.m`. For the case without control, we use zero boundary conditions on $x = 1$ which corresponds to setting $v = 0$ or equivalently we can set $\mathbf{K} = 0$ since $v = -\mathbf{K}\mathbf{z}$.

1. Generate the mesh by running the `square` program and use 41 or 81 or 101 points on each side of the square.

2. Study the program `get_matrices.m` which performs the matrix assembly.

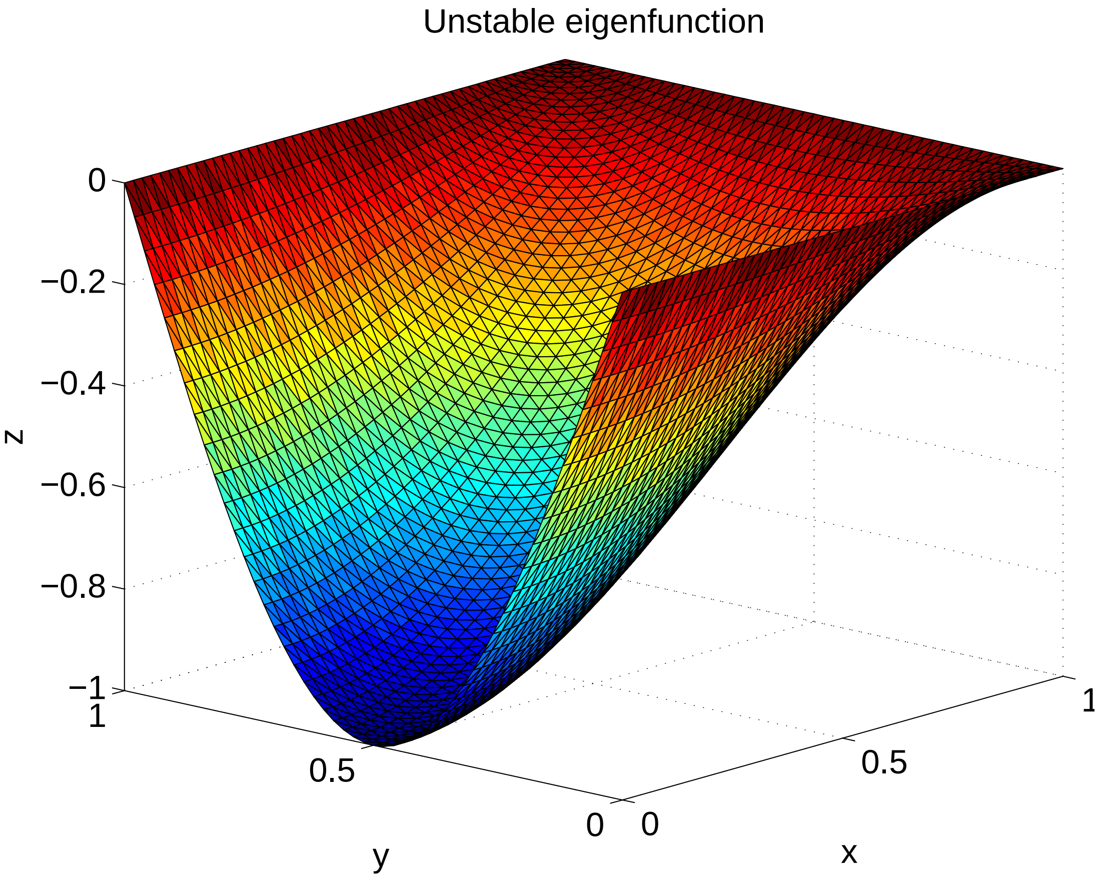

Figure 8: Unstable eigenfunction

3. The value of ω is set in the file `parameters.m`. Set $\omega = 0$ and calculate five eigenvalues with largest real part using `eigs` function. Is there an unstable eigenvalue ? Use the program `check_eig.m`.

4. Study the program `heat2d.m` which solves the heat equation.

5. With $\omega = 0$, run the program.

   ```
   >> heat2d(0) % argument 0 = no control => v=0 or K=0
   ```

   Observe that the energy is decreasing with time. The energy is plotted with linear scale for time and log scale for energy. The straight line behaviour indicates that the energy is decaying exponentially with time.

6. Set $\omega = 0.4$ and calculate five eigenvalues with largest real part using `eigs` function. Is there an unstable eigenvalue ? Compare it with exact unstable eigenvalue given above.

7. With $\omega = 0.4$, run the program. Observe that the energy is increasing exponentially with time.

5 Feedback control

The feedback control is determined by solving the minimal norm control problem using the matrices $\mathbf{M}$, $\mathbf{A}$ and $\mathbf{B}$. Since the dimensions of these matrices are very large, we determine the feedback matrix $\mathbf{K}$ by considering only the unstable components of the system as we did in the case of 1-D heat equation. This is implemented in the program `heat2d.m`.

6 Excercises II

1. Set $\omega = 0.4$. Use the program `check_eig.m` to compute eigenvalues. There should be one unstable eigenvalue.

2. Set $\omega = 0.4$. Use Hautus criterion to check that $(\mathbf{M}, \mathbf{A}, \mathbf{B})$ is stabilizable.

3. For $\omega = 0.4$ run `heat2d.m`.

   ```
   >> heat2d(1)
   ```

   After $\mathbf{K}$ is computed inside `heat2d.m`, check that $(\mathbf{M}, \mathbf{A} - \mathbf{BK})$ is stable by computing it eigenvalue (only a few eigenvalues with largest real part using `eigs`). Does the solution become stable with this feedback ?

4. Save the matrix $\mathbf{K}$ obtained above. Set $\omega = 0$ and load $\mathbf{K}$ into `heat2d.m` instead of evaluating it. Run it and observe the decay rate of the solution. Is it better than the case without control ?

5. Try to solve the Riccati equation for the full system and check how much time it takes. Use $\omega = 0.4$

   ```
   >> square(41)
   >> globals;
   >> parameters;
   >> read_data();
   >> [M,A,B,Q,H] = get_matrices();
   >> R = 1;
   >> [P,E,K] = care(full(A),full(B),full(Q),full(R),[],full(M));
   ```

   Use this feedback operator $\mathbf{K}$ and solve the heat equation.

7 Estimation and control

We now consider the case where we have partial observation as discussed in the previous sections. Thus we need to solve the estimation problem. This requires the evauation

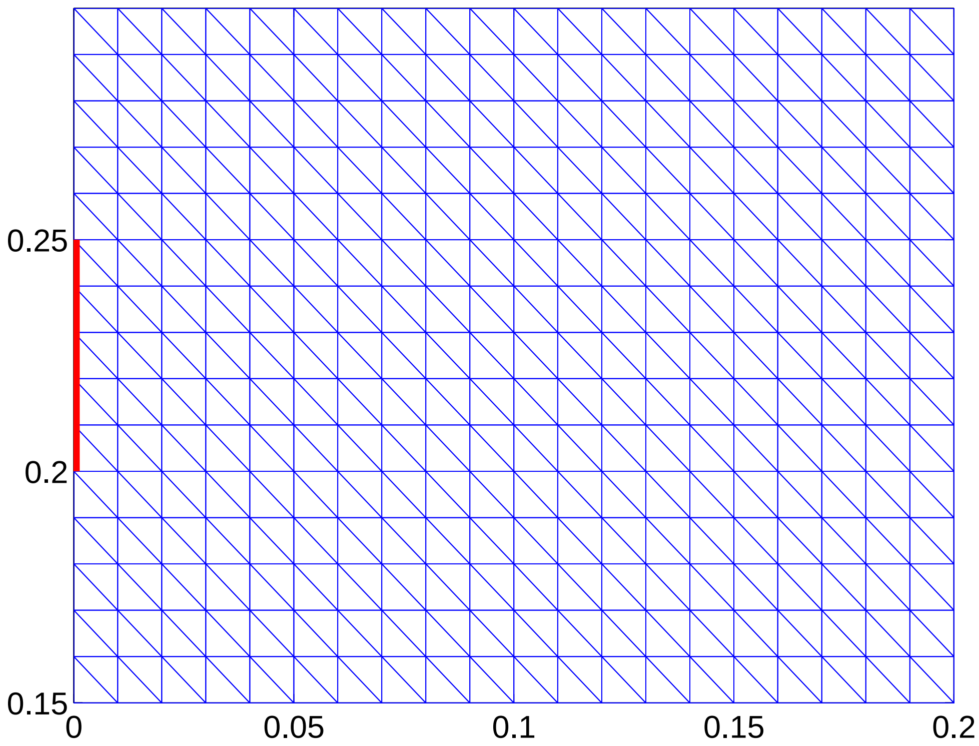

Figure 9: Triangular mesh with 100 edges on each side. The figure shows a zoomed view near the left boundary. Notice that the first observation zone $\{x = 0, 0.2 \leq y \leq 0.25\}$ is exactly covered by five boundary edges.

of the filtering gain matrix $\mathbf{L}$. Once again, the large dimensions of the system matrices forces us to evaluate $\mathbf{L}$ using only the unstable components.

7.1 Computing the observation

We will assume that the intervals $[a_i, b_i]$ on which the observation is computed is exactly covered by the edges of the finite element mesh, see figure (9). Let E_i denote the edges on $[a_i, b_i]$. Then

$$\begin{aligned} y_i &= \frac{1}{b_i - a_i} \int_{a_i}^{b_i} z(0, y, t) \mathrm{d}y, \qquad i = 1, 2, 3 \\ &= \frac{1}{b_i - a_i} \sum_{e \in E_i} \int_e z(0, y, t) \mathrm{d}y \\ &= \frac{1}{b_i - a_i} \sum_{e \in E_i} \frac{1}{2} (z_{e_1} + z_{e_2}) |e| \end{aligned}$$

The set of observations can be written as

$$\mathbf{y} = \mathbf{H}\mathbf{z}$$

The observation zones are defined by the following parameters

Value/i	1	2	3
a_i	0.20	0.50	0.80
b_i	0.25	0.55	0.85

7.2 Excercises III

The control and estimation problem is implemented in `heat2d_est.m`.

1. For $\omega = 0.4$ run `heat2d_est.m`; compare the true solution and estimated solution by computing the error norm between the two solutions.

2. Use $\omega = 0.4$; we now want to achieve better stabilization, so we add a further shift of 0.5 in which case there are three unstable eigenvalues

   ```
   alpha=0.5;
   opts.p=50; % number of Lanczos vectors
   [V,D] = eigs(A+alpha*M,M,10,'lr',opts);
   ee=diag(D)
   nu = 3;
   V = V(:,1:nu);
   D = D(1:nu,1:nu);
   ```

   Note that we add the shift `alpha` only to compute the feedback gain.