Two dimensional heat equation

Praveen. C, Deep Ray, Jean-Pierre Raymond

July 17, 2013

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

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

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

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

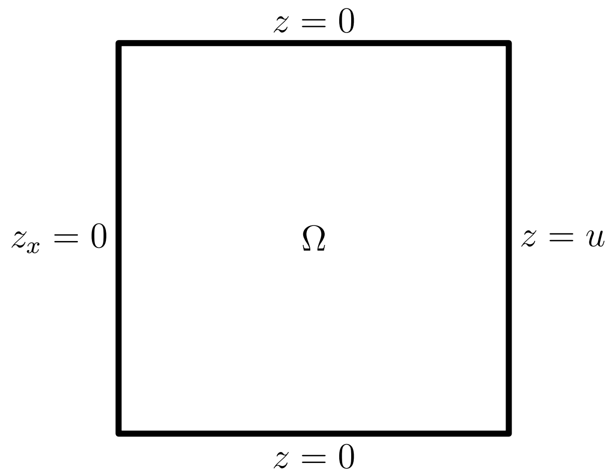

Figure 1: Problem definition

1.1 Observations

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

$$I_i = [a_i, b_i]$$

as shown in figure. Thus the observations are

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

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

$$\frac{\mathrm{d}}{\mathrm{d}t}(z(t), \phi)_{L^2} = -\mu \int_\Omega \nabla z \cdot \nabla \phi \mathrm{d}x + \omega \int_\Omega z \phi \mathrm{d}x, \quad \forall \phi \in H^1_{\Gamma_D}(\Omega)$$

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

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

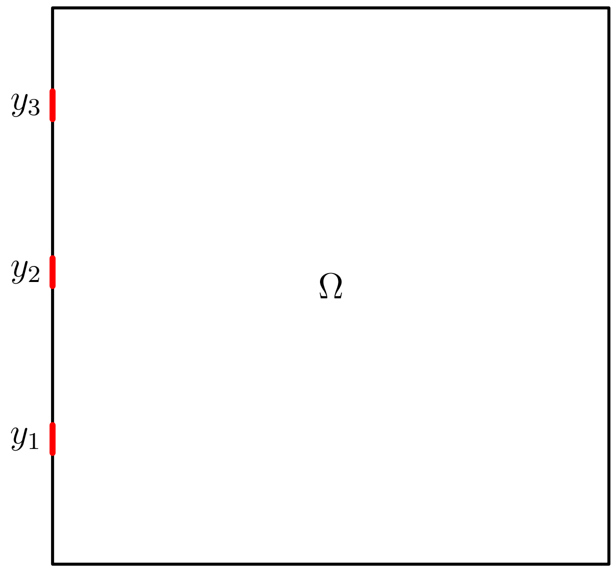

Figure 2: Observations

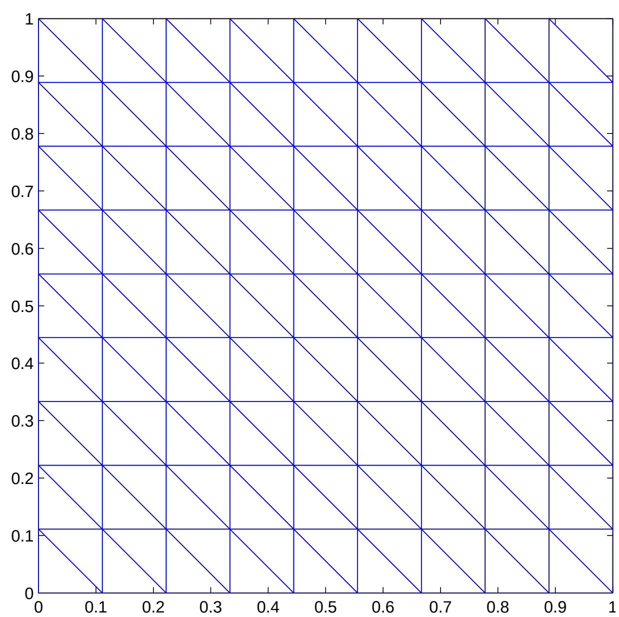

Figure 3: Example of a finite element mesh

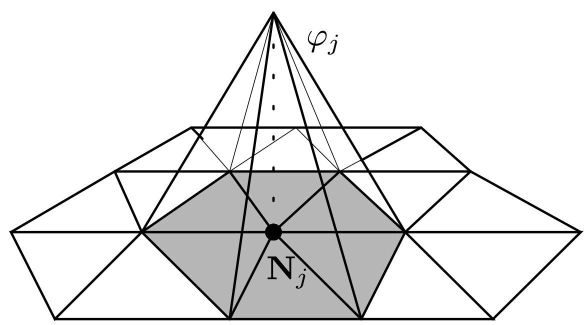

Figure 4: Piecewise affine basis functions

2 FEM approximation

Consider a division of Ω into disjoint triangles as shown in figure (xxx). 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 $\phi_i(x,y)$ with the property that

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

We will take the control to be of the form

$$u(y,t) = v(t)\sin(\pi y)$$

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

$$\frac{\mathrm{d}}{\mathrm{d}t}(z(t), \phi_i)_{L^2} = -\mu \int_\Omega \nabla z \cdot \nabla \phi_i \mathrm{d}x + \omega \int_\Omega z \phi_i \mathrm{d}x, \quad \forall i \in N_f$$

i.e.,

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

In order to simplify the presentation we will ignore the term containing $\frac{\mathrm{d}v}{\mathrm{d}t}$[1] and then we can write the FEM formulation as

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

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 in the set N_f.

2.1 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

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

[1] This term vanishes if we use the trapezoidal rule for integration.

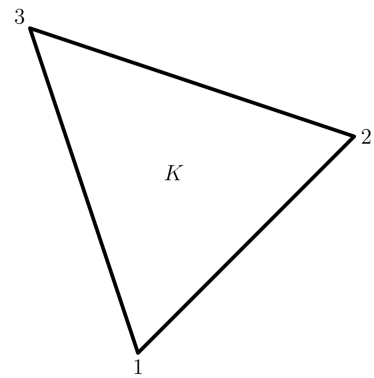

Figure 5: Triangle

and similarly the stiffness matrix is computed as

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

The integrals on each triangle K will be evaluated exactly. For a triangle K with vertices labelled 1, 2, 3 as in figure (5), the local mass and stiffness matrices are given by

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

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

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 $\mathbf{A}$

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`

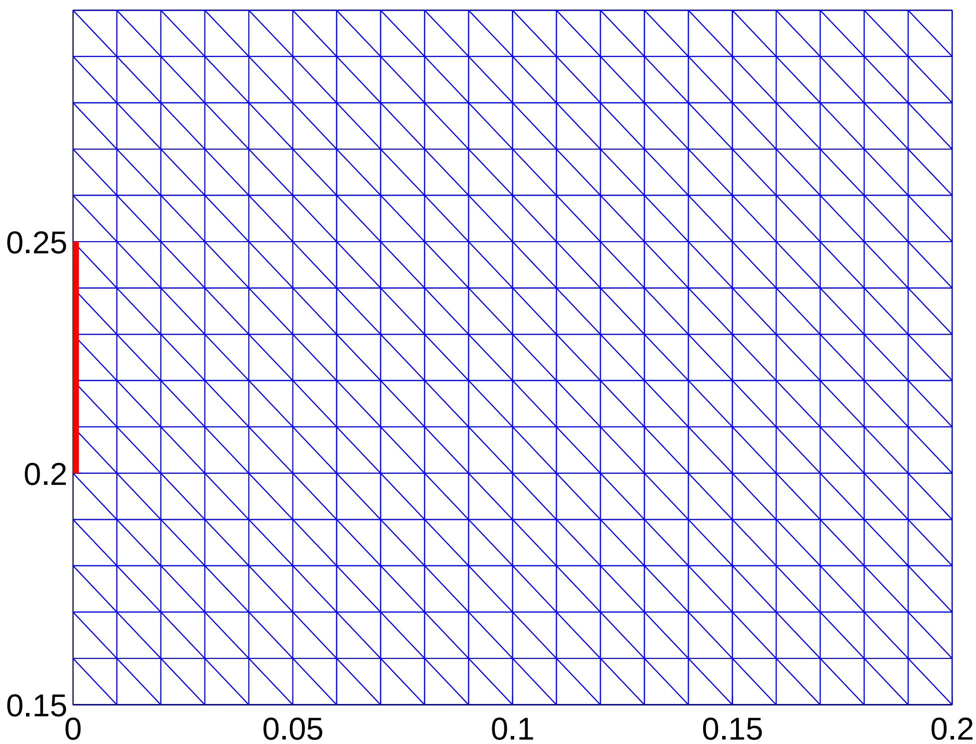

Figure 6: Triangular mesh with 100 edges on each side. The figure shows a zoomed view near the left boundary. Notice that the first observation zone $\{x = 0, 0.2 \le y \le 0.25\}$ is exactly covered by five boundary edges.

2.2 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 (6). Let E_i denote the edges on $[a_i, b_i]$. Then

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

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

2.3 Grid information

The grid information consists of following files

- `coordinates.dat`: contains x, y coordinates of all the vertices
- `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 (7), (8). 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)
```

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.

3 List of matlab programs

The programs are under the directory `heat_2d` 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

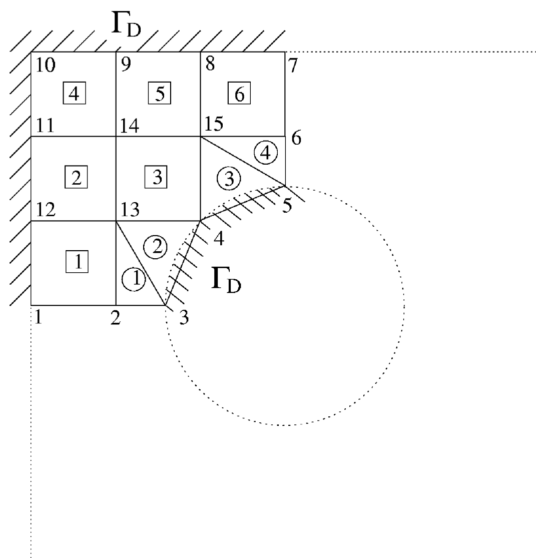

Figure 1. Example of a mesh.

Figure 7: Example of a mesh

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 8: Structure of mesh files

4 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 (9). We will use the unstable eigenfunction as the initial condition for the heat equation. This initial condition satisfies the following boundary conditions

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

5 Excercises

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

2. The value of ω is set in the file `parameters.m`. Set $\omega = 0$ and calculate the eigenvalue with largest real part using `eigs` function. Is this eigenvalue stable or unstable ?

3. With $\omega = 0$, run the program. 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 wrt time.

4. Set $\omega = 0.4$ and caculate the eigenvalue with largest real part using `eigs` function. Compare it with exact eigenvalue given above.

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

6 Estimation and feedback control

7 Excercises

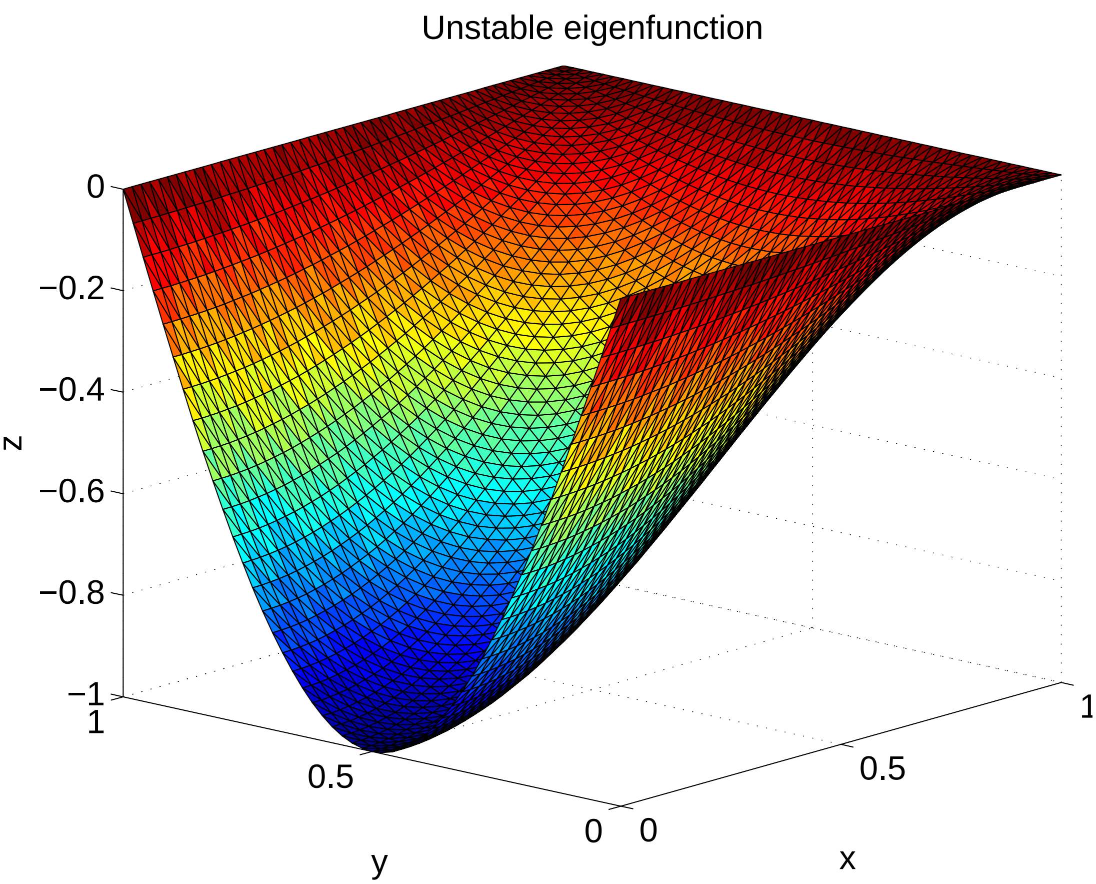

Figure 9: Unstable eigenfunction