Advanced Numerical Methods
SF2520

Fall term 2021

Computer Exercise 3

Parabolic Equations

David Stuhrmann

19980508

dast9356@student.su.se

Sofia Nguyen

19980625

song0575@student.su.se

November 12, 2021

Part 1: Rescaling to dimensionless form

(a)

We start start with having the PDE

$$\rho C_p \frac{\partial T}{\partial t} = k \frac{\partial^2 T}{\partial y^2} \tag{1}$$

By changing the dependent variables according to

$$u = \frac{T}{T_0}, \quad x = \frac{y}{L}, \quad \tau = \frac{t}{t_P}$$

where u, x and τ are the scaled quantity relative to the characteristic unit of measure to be determined. Inserting this to (1) we receive

$$\rho C_p \frac{T_0}{t_P} \frac{\partial u}{\partial \tau} = k \frac{T_0}{L^2} \frac{\partial^2 u}{\partial x^2}$$

Divide both sides with the coefficients in front of the derivative of the right hand side which yields

$$\frac{\rho C_p L^2}{k\, t_P} \frac{\partial u}{\partial \tau} = \frac{\partial^2 u}{\partial x^2}$$

Set all of the coefficients equal to $1/a$ so

$$a = \frac{k\, t_P}{L^2 \rho C_P} \tag{2}$$

with the dimension:

$$[a] = \frac{\left[\frac{J}{msC}\right][s]}{[m^2]\left[\frac{kg}{m^3}\right]\left[\frac{J}{kgC}\right]} = [1] \tag{3}$$

of where the dimension of the variables are all cancelling each other and this thus shows that a is dimensionless and we also derived to the dimensionless equation

$$\frac{\partial u}{\partial \tau} = a \frac{\partial^2 u}{\partial x^2}$$

b)

For the fixed value $a = \frac{1}{4}$ choose reasonable values for L and t_p when all the other constants are given. For copper we have a density $\rho = 8960\,\frac{\mathrm{kg}}{\mathrm{m}^3}$, a specific heat capacity $C_p = 384.4\,\frac{\mathrm{J}}{\mathrm{kgK}}$ and a thermal conductivity $k = 400\,\frac{\mathrm{W}}{\mathrm{mK}}$. Then we get the following:

$$\frac{L^2}{t_p} = \frac{k}{a \rho C_p} \approx 0.000465 \frac{m^2}{s} \tag{4}$$

And if we choose $t_p = 30\,\mathrm{s}$ then $L = 0.118\,\mathrm{m}$. This would be something that is realizable in the lab.

Part 2: 1D problem with Explicit Euler and built-in MATLAB commands

(a)

Discretize the rescaled equation which is $u_\tau = a u_{xx}$. In the lecture we already saw this for the case $a = 1$. First setup the spacial points $x_j = j\Delta x$ for $j = 0, \ldots, n+1$ and $\Delta x = \frac{1}{n+1}$. Use the notation $u_j(t) = u(t, x_j)$. Then the approximation of $a u_{xx}$ for all $j = 1, \ldots, n$ is:

$$u_j'(\tau) = \frac{a}{(\Delta x)^2}(u_{j+1}(\tau) - 2u_j(\tau) + u_{j-1}(\tau)) \tag{5}$$

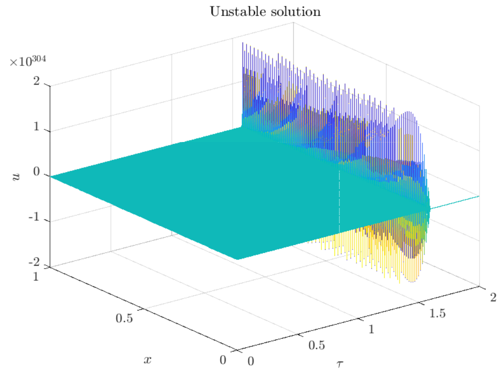


Figure 1: 3D mesh plot of the unstable solution $\Delta x = 0.01$, $\Delta\tau = 2.1 \cdot 10^{-4}$ and $\frac{\Delta\tau}{(\Delta x)^2} = 2.10$

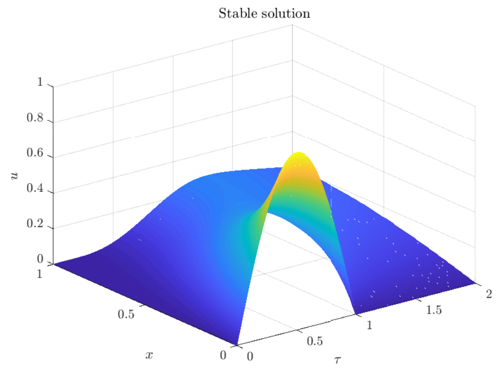


Figure 2: 3D mesh plot of the stable solution $\Delta x = 0.01$, $\Delta\tau = 1.9 \cdot 10^{-4}$ and $\frac{\Delta\tau}{(\Delta x)^2} = 1.90$

Next we deal with the boundary conditions. On the left side $x = 0$ the Dirichlet conditions yield:

$$u_1'(\tau) = \frac{a}{(\Delta x)^2}(u_2(\tau) - 2u_1(\tau) + \sin(\pi\tau)\theta(1-\tau)) \tag{6}$$

where θ is the Heaviside step function. On the right side use a ghost point to handle the Neumann conditions.

$$\begin{aligned} u_{n+1}'(t) &= \frac{a}{(\Delta x)^2}(d_{-1} + (d_0 - 2)u_n(t) + 2u_{n-1}(t)) &(7)\\ &= \frac{a}{(\Delta x)^2}(-2u_n(t) + 2u_{n-1}(t)) &(8)\end{aligned}$$

since $d_{-1} = d_0 = 0$ as $u_x(1,\tau) = 0$. All together this results in the system:

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}\tau} = A\boldsymbol{u} + \boldsymbol{b}(\tau) \tag{9}$$

where $\boldsymbol{u}(\tau) = (u_1(\tau), \ldots, u_{n+1}(\tau))^T \in \mathbb{R}^{n+1}$ and:

$$A = \frac{a}{(\Delta x)^2}\begin{pmatrix} -2 & 1 & & & \\ 1 & \ddots & \ddots & & \\ & & \ddots & \ddots & 1 \\ & & & 1 & -2 & 1 \\ & & & & 2 & -2 \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)} \tag{10}$$

$$\boldsymbol{b}(\tau) = \frac{a}{(\Delta x)^2}\begin{pmatrix} \sin(\pi\tau)\theta(1-\tau) \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{n+1} \tag{11}$$

The initial condition is just $\boldsymbol{u}(0) = (0, \ldots, 0)^T$. To solve the system with the explicit Euler method we reuse our implementation of the Euler method from CE1.

The stability of the method with the additional a now:

$$\frac{\Delta\tau}{(\Delta x)^2} \leq \frac{1}{2a} = 2 \tag{12}$$

We obtain an unstable solution when the ratio is 2.1 ($\Delta\tau = 0.01$, $\Delta x = 2.1 \cdot 10^{-3}$) as in figure 1 where you can also see that the values near $\tau = 2$ just explode. On the other hand figure 2 shows the stable solution with the ratio being 1.9 ($\Delta\tau = 0.01$, $\Delta x = 1.9 \cdot 10^{-3}$). Both figures show the temperature u over the grid of space $x \in [0, 1]$ and time $\tau \in [0, 2]$.

For the next plots we only focus on the stable solution we obtained before. Keeping the time fixed at $\tau = 1$ we plotted the spacial temperature profile of the rod for $x \in [0, 1]$. Is is shown in figure 3.

In figure 4 the spacial coordinate is fixed for $x = 0$ and $x = 1$ and the temperature profile at those points over time is shown.

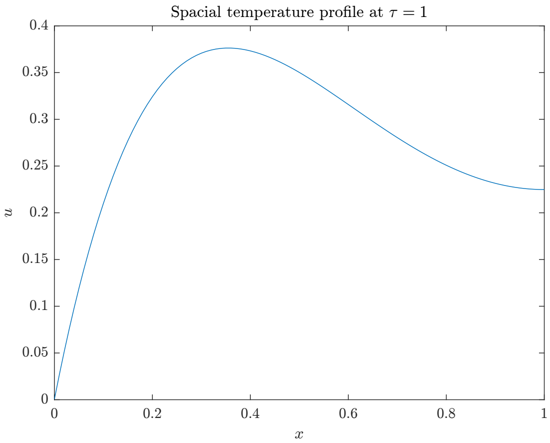


Figure 3: Spacial temperature profile for the stable solution for the time $\tau = 1$.

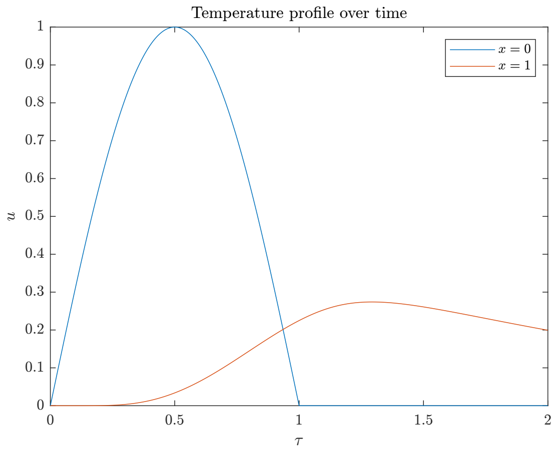


Figure 4: Temperature profile over time for the stable solution at the positions $x = 0$ and $x = 1$.

b)

In this part we compare the three methods `ode23`, `ode23s` and `ode23sJ` which is just `ode23s` with the Jacobian set to A from part (a). This last option is good because the solver needs the Jacobian during its computation and adding it as an argument reduces the need to compute it numerically and also reduces the induced errors during that numerical approximation.

For the spacial steps $N = 100, 200, 400$ we ran the three solvers on the previous problem. The results in step size and computation time are shown in table 1. The tolerances where `RelTol` $= 1 \times 10^{-3}$ and `AbsTol` $= 1 \times 10^{-6}$. The time measurement was done with Matlab's `tic`/`toc` functions.

Table 1: Comparison table of the three solvers for different spacial resolutions and time interval $[0, 2]$.

N	#time steps			computational time (s)		
	`ode23`	`ode23s`	`ode23sJ`	`ode23`	`ode23s`	`ode23sJ`
100	7958	135	73	0.087	0.143	0.009
200	31835	153	79	0.425	0.371	0.029
400	127347	156	80	3.82	1.63	0.034

We notice that the number of steps decrease drastically from `ode23` to `ode23s` by more than one order of magnitude. This behaviour is expected since the `ode23` uses an explicit method whose stability limit forces us to take a lot of steps. So the `ode23s` solver passes through stiff areas with far fewer steps than `ode23` and thus the results of the lower steps from `ode23s`. We can further reduce the needed steps by specifying the Jacobian in `ode23sJ` as the error of numerically computing the Jacobian is reduced.

The ode23 solver has the CFL number $\frac{\Delta t}{(\Delta x)^2}$. So doubling the number of steps in x halfes Δx so Δt has to be 4 times less which correspond to the 4 times for steps in time. This is accurate with the behaviour for the number of time steps of `ode23` seen in table 1.

The Jacobian of the function $f(t,u) = Au + b(t)$ w.r.t. u is A. Thus by specifying it the `ode23s` solver doesn't need to calculate the partial derivatives of f in every time step. This corresponds to the observed speedup. Also Matlab doesn't need to estimate the Jabobian using finite difference. which reduces the error and allows for fewer steps to meet the tolerances. In or ccase A is valid to use as the Jacobian due the form of f and A being independent of τ.

Solving the parabolic PDE by semi-discretizations leads to stiff ODE systems which gives us a severe stability. We have seen before that in this case implicit methods should be used when handling stiff ODE's, therefore `ode23s` and `ode23sJ` are preferred. Since the right side of the ODE system is given in terms $Au + b$ we can further speed up the computiation by specifying the Jacobian, so the solver `ode23sJ` performs the best.

Part 3: 2D problem with Crank-Nicolson

We are solving the PDE with the boundary condition and initial condition:

$$\frac{\partial u}{\partial \tau}(x,y,\tau) = \Delta u(x,y,\tau) + f(x,y) \tag{13}$$

$$f(x,y) = 6000\mathrm{e}^{-5(x-1)^2-10(y-1.5)^2} \tag{14}$$

$$u(0,y) = 40, \quad u(5,y) = 400 \quad \forall y \in [0,2] \tag{15}$$

$$\frac{\partial u}{\partial y}(x,0) = \frac{\partial u}{\partial y}(x,2) = 0 \quad \forall x \in [0,5] \tag{16}$$

$$u(x,y,0) = u_0(x,y) = 40 + 72x \tag{17}$$

with the Crank-Nicolson method up to $\tau = 10$ with a time step of $h = 0.1$. We end up to need to solve a linear equation of (17) with

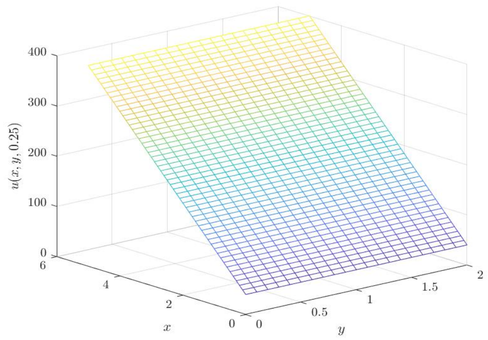
$$(I + \tfrac{\Delta\tau}{2}A)\mathbf{u}^{n+1} = (I + \tfrac{\Delta\tau}{2}A)\mathbf{u}^n + \tfrac{\Delta\tau}{2}(\mathbf{b}(\tau_n) + \mathbf{b}(\tau_{n+1})) \tag{18}$$

where the matrix A is reused from the computer exercise 2 part 2.

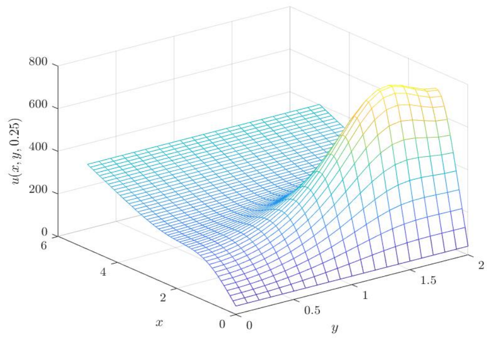
$$A = \begin{pmatrix} T & -2I & & & \\ -I & \ddots & -I & & \\ & \ddots & \ddots & \ddots & \\ & & -I & \ddots & -I \\ & & & -2I & T \end{pmatrix}, \quad T = \begin{pmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{pmatrix} \tag{19}$$

and I is the identity matrix. We also reuse the definition of $\boldsymbol{b}$ which contains the forcing terms $f_{i,j} = f(x_i, y_j)$ plus the boundary conditions. N and M are determined by h accordingly.

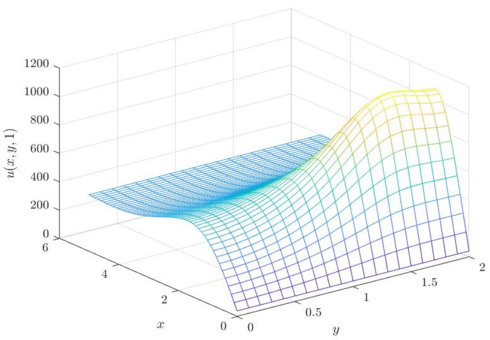
$$\boldsymbol{b} = \begin{pmatrix} f_{1,0} + \frac{40}{h^2} \\ f_{2,0} \\ \vdots \\ f_{N-2,0} \\ f_{N-1,0} + \frac{400}{h^2} \\ f_{1,1} + \frac{40}{h^2} \\ f_{2,1} \\ \vdots \\ f_{N-2,M-1} \\ f_{N-1,M-1} + \frac{400}{h^2} \\ f_{1,M} + \frac{40}{h^2} \\ f_{2,M} \\ \vdots \\ f_{N-2,M} \\ f_{N-1,M} + \frac{400}{h^2} \end{pmatrix} \tag{20}$$



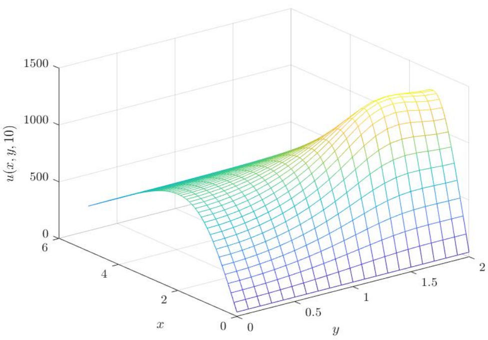
(a) Solution of the PDE at $\tau = 0$



(b) Solution of the PDE at $\tau = 0.25$



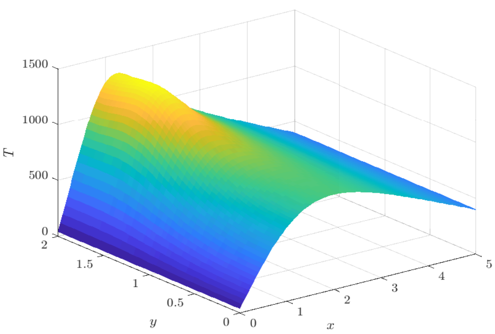
(c) Solution of the PDE at $\tau = 1$



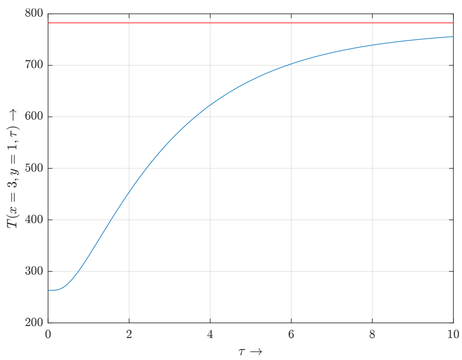
(d) Solution of the PDE at $\tau = 10$

Figure 5: Solution of the PDE at fixed times is the spacial temperature profile.

The four plots showing the temperature profile at different points in time are shown in figure 5. The times are $\tau = 0, 0.25, 1, 10$. Comparing figure 5d with the plot (see figure 6a which we computed in previous exercise sheet 2 part 2, we see that both are comparable since we approach the steady state solution. This approach towards the steady state solution is also shown strongly in the plot of the temperature at the point $x = 3$, $y = 1$ for the different times τ, see figure 6b. During the time evolution we see that the initial condition is overcome by the forcing term f which has the heating peak at $x = 1$, $y = 1.5$.



(a) Solution of the PDE from previous project with time step $h = 0.5$



(b) The temperature solved with Crank-Nicolson at point $u(3, 1)$ for $\tau = [0, 10]$ with the temperature of 782.43 marked as the red line.

Figure 6: Figures for comparing solution of the PDE with the regular finite difference method and the specified Crank-Nicolson method.

The initial data is a linear function connecting the boundary conditions on the left and right side. Since will have a time evolution anyway it is well suited for this problem. We solved this task with the Crank-Nicolson due to us working on a parabolic PDE with two spacial dimensions and the Crank-Nicolson method as an implicit 2nd order method is well suited for that and thus gives a good accuracy and stability.