Lennart Edsberg

Matematik, KTH

Applied Numerical Methods I, SF2520
Autumn 2015; deadline for 5 credit point: November 6th 2015

Computer Lab 4
Partial differential equation of parabolic type

A metallic rod of length L $[m]$ is initially of temperature $T = 0$ $[C]$. At time $t = 0$ a heat pulse of temperature $T = T_0$ and duration t_P $[s]$ hits the left end (at $x = 0$) of the rod. At the right end (at $x = L$) the rod is isolated. After some time the rod will therefore be warmer in the right end and then cool off again. The following partial differential equation can be set up for the heat diffusion process through the rod:

$$\rho C_p \frac{\partial T}{\partial t} = k \frac{\partial^2 T}{\partial x^2}, \quad t > 0, \quad 0 < x \leq L,$$

The boundary conditions are

$$T(0,t) = \begin{cases} T_0, & 0 \leq t \leq t_P \\ 0, & t > t_P \end{cases}, \qquad\qquad \frac{\partial T}{\partial x}(L,t) = 0$$

and the initial condition is

$$T(x,0) = 0, \quad 0 < x \leq L$$

In the PDE, ρ is the density $[kg/m^3]$, C_P the heat kapacity $[J/kg \cdot C]$ and k is the thermal conductivity $[J/m \cdot s \cdot C]$ of the rod.

The purpose of this lab is to

* scale the problem to dimensionless form
* discretize the scaled problem with the Method of Lines (MoL)
* investigate stability properties of Euler's explicit method
* comparison between an explicit and an implicit adaptive method
* show how an implicit method can be made more efficient
* visualize the result in a 2- and 3-dimensional plot

This lab consists of the following parts:

a) Show that with the new variables u, ξ and τ defined by

$$T = T_0 u, \qquad x = L\xi, \qquad t = t_P \tau$$

the problem can be transformed (scaled) into the following dimensionless form

$$\frac{\partial u}{\partial \tau} = a\frac{\partial^2 u}{\partial \xi^2}, \quad \tau > 0, \quad 0 < \xi < 1,$$

with boundary conditions

$$u(0,\tau) = \begin{cases} 1, & 0 \le \tau \le 1 \\ 0, & \tau > 1 \end{cases}, \qquad \frac{\partial u}{\partial \xi}(1,\tau) = 0$$

and initial condition

$$u(\xi, 0) = 0, \quad 0 < \xi \le 1$$

Show that the only remaining parameter a is dimensionless. From now on assume that a has the numerical value $a = 1$.

b) Discretize in space (the ξ-variable) using the constant stepsize h and central differences to obtain an ODE-system

$$\frac{d\mathbf{u}}{d\tau} = A\mathbf{u} + \mathbf{b}(\tau), \qquad \mathbf{u}(0) = \mathbf{u}_0,$$

where $\mathbf{u}_0$ is the zero vector. Show the dimensions and structures of A, $\mathbf{b}(\tau)$ and $\mathbf{u}_0$. Show also the discretized grid of the ξ-axis you have used and how the gridpoints are numbered.

c) Numerical part: Discretize the ODE-system in b) with Euler's explicit method. Use constant time step Δt. To make your calculations efficient you should write your code so that the vector $A\mathbf{u}+\mathbf{b}$ is formed directly. Store the whole approximate solution (including initial and boundary conditions) in a large matrix U, as

$$U = \begin{pmatrix} 0 & x & x & x & \dots & x \\ 0 & x & x & x & \dots & x \\ 0 & x & x & x & \dots & x \\ . & . & . & . & \dots & x \\ . & . & . & . & \dots & x \\ . & . & . & . & \dots & x \\ 0 & x & x & x & \dots & x \\ 1 & 1 & 1 & 1 & \dots & 0 \end{pmatrix}$$

Graphical part: Use e.g. `surf` to draw a 3D-plot of the solution. Experiment with different values of the discretization step $\Delta x = h$ and Δt and study stability. Submit one graph showing a stable solution and one graph with an unstable solution. Present the values of Δx, Δt and $\Delta t/\Delta x^2$ in the two cases.

d) In this part of the lab you shall compare the explicit method ode23 and the implicit method ode23s, suitable for stiff problems, in the Matlab library. The two functions shall be run under similar conditions (same problem as in d), same tolerance) and for three sizes of the stepsize h corresponding to $N = 10, 20, 40$ grid-points on the ξ-axis.

The comparison shall comprise

1) the number of time steps needed to reach $\tau = 2$

2) the cpu-time needed to do each computation

3) the maximal timestep that each method could take

Collect your statistics in three tables:

	$timesteps$		$cpu - time$		h_{tmax}	
N	$ode23$	$ode23s$	$ode23$	$ode23s$	$ode23$	$ode23s$
10						
20						
40						

e) (This part is not compulsory, 1 extra credit if done correctly) The conclusion from d) is that the number of time-steps is considerably smaller when using a stiff method. However the default implementation is not efficient for stiff problems coming from parabolic PDEs. The main reason is that all the linear systems of equations $Ax = b$ that are to be solved in each time step need a lot of number crunching since they are based on using the backslash-operator, i.e. Gaussian elimination on a full matrix is performed each time $Ax = b$ is solved. However, the system matrix of these equations is tridiagonal, which is not considered in the default implementation of ode23s. With the Matlab function odeset, options can be set by the user in order to make the computation more efficient. Do `help odeset` and study the options for 'Jacobian' and 'Jpattern'. Also do `help odefile` and look at the example. With the help of these options, the number of flops can be reduced considerably. Experiment with these three options and collect statistics regarding the number of floating point operations as was done in d).

f) Visualize the result of a successful computation from c) in graphs. One graph shall show 2-dimensional plots of $u(\tau, \xi)$ $0 \leq \xi \leq 1$ at four timepoints $\tau \approx 0.5, 1, 1.5, 2$. Another graph shall show a 3-dimensional plot of u as function of $0 \leq \tau \leq 2$ and $0 \leq \xi \leq 1$, hence initial and boundary conditions should be included in the plot.

g) Submit the results as a small report. Parts a) and b) may be hand written. In c) submit the graphs with explanations and in d) submit the program and the three tables with comments on the results, in e) (not compulsory) the program and the table plus comments and in f) one graph with 4 2-D plots (use subplot!!) and one graph with a 3-D plot. Of course for all parts there should be text explaining what you have done, *not only graphs and programs*!

Conclude your results by answering the following questions:

If we want efficient numerical calculations for parabolic problems:

Q1: What type of ODE-method should be used, stiff or non-stiff?

Q2: What structure does the jacobian of the ODE-system have, banded or sparse in another way?