In this paper we will use Matlab to numerically solve the heat equation (also known as diffusion equation) a partial differential equation that describes many physical precesses including conductive heat flow or the diffusion of an impurity in a motionless fluid.

In three-dimensional medium the heat equation is:

$$\frac{\partial u}{\partial t} = k * (\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2})$$

Here *u* is a function of *(x,t,y,z)* that represents temeperature at time at position *(x,y,z)*

The costant k depends on the material involved,it is called the thermal conductivity in the case of heat flow and diffusion coefficient in the case of diffusion.To simplify matters let us assume that the medium is one-dimensional.This could represent   heat flow in a thin insulated wire or rod

Then partial differential equation becomes

$$\frac{\partial u}{\partial t} = k * \frac{\partial^2 u}{\partial x^2}$$

where u is temperature at time t a distance x along the wire

u=u(x,t)

A finite difference solution

To solve this partial differential equation we need both initial conditions of the form $u(x, t = 0) = f(x)$,where $f(x)$ gives the temperature distirbution in the wire at time 0, and boundary conditions at the endpoints of the wire,call them $x = a \; and \; x = b$

We choose so- called Dirichlet boundary conditions

$u(x = a, t) = L_a; u(x = b, t) = L_b$ which correspond to the temperature being held steady at valuesL_a and L_b at the two endpoints

Though an exact solution is available in this scenario, let us instead illustrate the numerical method of finite differences.

To begin with ,on computer we can only keep track of the temperature u

at discrete set of times and discrete set of positions.

Let times be $0, \Delta t, 2\Delta t, \dots, n\Delta t$,and let the positions $a, a + \Delta x, \dots, J\Delta x = b$

let $u_j^n = u(a + J\Delta t, n\Delta t)$.Rewriting the partila differential equation in terms of finite- difference approximations to the derivatives,we get

$\frac{\partial u}{\partial t} = \frac{u_J^{n+1} - u_J^n}{\Delta t};$

$$k * \frac{\partial^2 u}{\partial x^2} = k * \frac{u_{J+1}^n - 2u_J^n + u_{J-1}^n}{\Delta x^2}$$

So we get

$\frac{u_J^{n+1} - u_J^n}{\Delta t} = k * \frac{u_{J+1}^n - 2u_J^n + u_{J-1}^n}{\Delta x^2}$

Thus if for a particualr n,we know the values of u_J^n for all j we can solve equation above to find u_J^{n+1} for each j;

$$u_J^{n+1} = u_J^n - \frac{k * \Delta t}{\Delta x^2}(u_{J+1}^n - 2u_J^n + u_{J-1}^n)$$

after some algebric manipulation we get

u_J^{n+1}=$\left(u_{J+1}^n - u_{J-1}^n\right) * l + (1 - 2 * l) * u_J^n$

where $l = \frac{k*\Delta t}{\Delta x^2}$

In other words ,this equation tell us how to find the temperature ditribution at time step n+1 given the temperature distribution at time step n

Thus our numerical implementation of the heat equation is a discretized version of the microscopic description of diffusion we gave initially, that heat energy spreads due to random interactions between nearby particles.

The following M-file which we have named heat.m

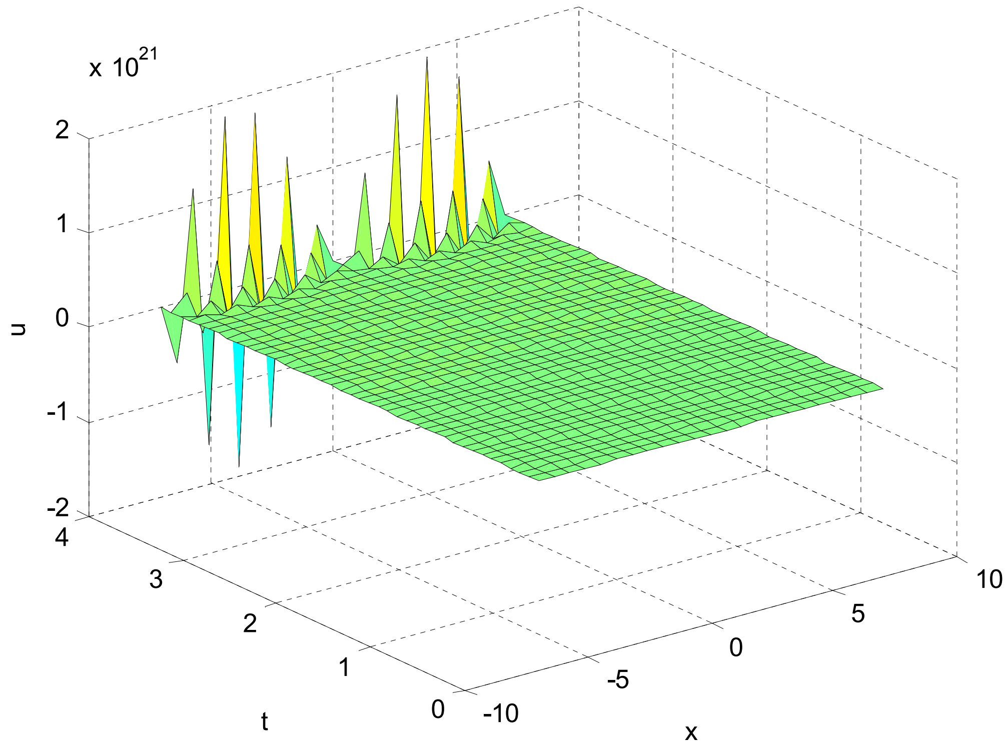
```
function u = heat(k, x, t, init, bdry)
% solve the 1D heat equation on the rectangle described
by
% vectors x and t with u(x, t(1)) = init and Dirichlet
% boundary conditions
% u(x(1), t) = bdry(1), u(x(end), t) = bdry(2).
J = length(x);
N = length(t);
dx = mean(diff(x));
dt = mean(diff(t));
l = k*dt/dx^2;
u = zeros(N,J);
u(1, :) = init;
for n = 1:N-1
u(n+1, 2:J-1) = l*(u(n, 3:J) + u(n, 1:J-2)) +...
(1 - 2*l)*u(n, 2:J-1);
u(n+1, 1) = bdry(1);
u(n+1, J) = bdry(2);
end


end
```

The function takes as inputs the value of k vectors of t and and x values

a vector int and a vector bdry containing a pair of boundary values

Let’s use the M-file above to solve the one-dimensional heat equation with k=5 on the interval−10≤x≤10 from time 0 to time 4, using boundary temperatures 16 and 26, and initial temperature distribution of 16 forx<0 and 26 forx>0. You can imagine that two separate wires of length 10 with different temperatures are joined at time 0 at positionx=0, and each of their far ends remains in an environment that holds it at its initial temperature. We must choose values for t and x; let’s try Δt=0.1 and Δx=0.6,

```
tvals = linspace(0, 4, 41);

xvals = linspace(-7, 7, 23);

init = 21 + 7*sin(xvals);

uvals = heat(5, xvals, tvals, init, [16 26]);

surf(xvals, tvals, uvals)

xlabel x; ylabel t; zlabel u
```



Here we used surf to show the entire solutionu(x,t).

Conclusions

The output is clearly unrealistic; notice the scale on the *u* axis! The numerical solution of partial differential equations is fraught with dangers, and instability like that seen above is a common problem with finite difference schemes. For many partial differential equations a finite difference scheme will not work at all, but for the heat equation and similar equations it will work well with proper choice ofΔt and Δx