Steven Wray
Math 550
Assignment 1

Let $u = u(x.t)$ be the solution of the inhomogeneous parabolic problem with inhomogeneous Dirichlet boundary conditions:

$$\frac{\partial u}{\partial t} - 0.1\frac{\partial^2 u}{\partial x^2} = f(x,t) \qquad \text{for } 2 < x < 3\,,\, 0 < t \leq 2$$

$$u(x,0) = v(x) \qquad \text{for } 2 < x < 3$$
$$u(2,t) = g_1(t),\ u(3,t) = g_2(t) \qquad \text{for } 0 < t \leq 2$$

The Crank-Nicolson Method

In order to describe this method, we will use a uniform spatial grid with N steps. Then the grid spacing is

$$h = \frac{3-2}{N} = \frac{1}{N}$$

and the grid points are located at $x_n = 2 + nh$ for $n = 0,1,\dots,N$. The temporal grid is also uniform with spacing

$$\Delta t = \frac{2-0}{M} = \frac{2}{M}.$$

Its grid points are $t_m = m\,\Delta t$ for $m = 0,1,\dots,M$.

For the Crank-Nicolson method, we will approximate the time derivative with $\bar{\delta}$ centered at $t_{m-1/2}$. So

$$\frac{\partial u}{\partial t}\left(x_n, t_{m-1/2}\right) \approx \bar{\delta}\left(x_n, t_{m-1/2}\right) \approx \frac{U_n^m - U_n^{m-1}}{\Delta t}$$

The space derivative is approximated with δ^2 also centered at $t_{m-1/2}$.

$$\frac{\partial^2 u}{\partial x^2}(x_n, t_{m-1/2}) \approx \delta^2(x_n, t_{m-1/2}) \approx \frac{U_{n+1}^{m-1/2} - 2U_n^{m-1/2} + U_{n-1}^{m-1/2}}{\mathrm{h}^2}$$

$$\approx \frac{1}{2}\left(\frac{U_{n+1}^{m} - 2U_n^{m} + U_{n-1}^{m}}{h^2} + \frac{U_{n+1}^{m-1} - 2U_n^{m-1} + U_{n-1}^{m-1}}{h^2}\right)$$

Substituting these approximations into the PDE gives

$$\frac{U_n^m - U_n^{m-1}}{\Delta t} - a\left(\frac{1}{2}\right)\left(\frac{U_{n+1}^{m} - 2U_n^{m} + U_{n-1}^{m}}{h^2} + \frac{U_{n+1}^{m-1} - 2U_n^{m-1} + U_{n-1}^{m-1}}{h^2}\right) = f_n^{m-1/2}$$

for $1 \le m \le M$ and $1 \le n \le N-1$. The initial conditions become

$$U_n^0 = V_n = v(x_n) \qquad \text{for } n = 0, \dots, N$$

and the boundary conditions become

$$U_0^m = g_1(t_m) \text{ and } U_N^m = g_2(t_m) \qquad \text{for } m = 1, \dots, M.$$

The source term is

$$f_n^{m-1/2} = f(x_n, t_{m-1/2}).$$

If we define $\alpha = \frac{a\,\Delta t}{h^2}$, the PDE can be rewritten as

$$U_n^m - U_n^{m-1} - \frac{1}{2}\alpha(U_{n+1}^m - 2U_n^m + U_{n-1}^m + U_{n+1}^{m-1} - 2U_n^{m-1} + U_{n-1}^{m-1}) = \Delta t\, f_n^{m-1/2}$$

and then move all of the known values to the right hand side of the equation

$$(1+\alpha)U_n^m - \frac{\alpha}{2}(U_{n+1}^m + U_{n-1}^m) = (1-\alpha)U_n^{m-1} + \frac{\alpha}{2}(U_{n+1}^{m-1} + U_{n-1}^{m-1}) + \Delta t\, f_n^{m-1/2}.$$

The same problem can be written in matrix-vector form using the vectors

$$\boldsymbol{U}^m = \begin{bmatrix} U_1^m \\ \vdots \\ U_{N-1}^m \end{bmatrix} \text{ and } \boldsymbol{f}^m = \begin{bmatrix} f_1^m \\ \vdots \\ f_{N-1}^m \end{bmatrix}$$

and the tridiagonal matrix

$$B_h = \begin{bmatrix} 2 & -1 & 0 & & & \\ -1 & 2 & -1 & & & \\ 0 & -1 & 2 & & & \\ & & & \ddots & \ddots & \\ & & & \ddots & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}.$$

In this notation, the system of equations becomes

$$(I + \alpha B_h)\boldsymbol{U}^m = (I - \alpha B_h)\boldsymbol{U}^{m-1} + \Delta t\, \boldsymbol{f}^{m-1/2}$$

and so our problem is a matrix-vector system of the form $A\boldsymbol{x} = \boldsymbol{b}$.

A Symmetric and Positive Definite Matrix

The matrix $(I + \alpha B_h)$ defined above is symmetric and positive definite (SPD).

Recall that this matrix has dimensions $N-1 \times N-1$. A matrix of this type is SPD if and only if $\boldsymbol{v}^T(I + \alpha B_h)\boldsymbol{v} > 0$ for all vectors $\boldsymbol{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_{N-1} \end{bmatrix}$ in $\mathbb{R}^{N-1}$ such that $\boldsymbol{v} \neq \boldsymbol{0}$. Here we have

$$\boldsymbol{v}^T(I + \alpha B_h)\boldsymbol{v} = \begin{bmatrix} v_1 & \cdots & v_{N-1} \end{bmatrix} \begin{bmatrix} 1+2\alpha & -\alpha & 0 & & & \\ -\alpha & 1+2\alpha & -\alpha & & & \\ 0 & -\alpha & 1+2\alpha & & & \\ & & & \ddots & -\alpha & \\ & & & \ddots & 1+2\alpha & -\alpha \\ & & & & -\alpha & 1+2\alpha \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_{N-1} \end{bmatrix}$$

$$= \begin{bmatrix} v_1 & \cdots & v_{N-1} \end{bmatrix} \begin{bmatrix} (1+2\alpha)v_1 - \alpha v_2 \\ -\alpha v_1 + (1+2\alpha)v_2 - \alpha v_3 \\ -\alpha v_2 + (1+2\alpha)v_3 - \alpha v_4 \\ \vdots \\ -\alpha v_{N-2} + (1+2\alpha)v_{N-1} \end{bmatrix}$$

$$= v_1[(1+2\alpha)v_1 - \alpha v_2] + v_2[-\alpha v_1 + (1+2\alpha)v_2 - \alpha v_3] + v_3[-\alpha v_2 + (1+2\alpha)v_3 - \alpha v_4] + \\ \dots + v_{N-1}[-\alpha v_{N-2} + (1+2\alpha)v_{N-1}]$$

$$= (1+2\alpha)v_1^2 - 2\alpha v_1 v_2 + (1+2\alpha)v_2^2 - 2\alpha v_2 v_3 + (1+2\alpha)v_3^2 - 2\alpha v_3 v_4 + \\ \dots - 2\alpha v_{N-2} v_{N-1} + (1+2\alpha)v_N^2$$

$$= v_1^2 + v_2^2 + \cdots + v_{N-1}^2 + \alpha v_1^2 + \alpha(v_1^2 - 2v_1 v_2 + v_2^2) + \alpha(v_2^2 - 2v_1 v_2 + v_3^2) + \\ \dots + \alpha(v_{N-2}^2 - 2v_{N-2} v_{N-1} + v_{N-1}^2) + \alpha v_{N-1}^2$$

$$= \sum_{j=1}^{N-1} v_j^2 + \alpha v_1^2 + \alpha(v_1 - v_2)^2 + \alpha(v_2 - v_3)^2 + \cdots + \alpha(v_{N-2} - v_{N-1})^2 + v_{N-1}^2$$

$$= \sum_{j=1}^{N-1} v_j^2 + \alpha v_1^2 + \sum_{j=1}^{N-2} \alpha\left(v_j - v_{j+1}\right)^2 + \alpha v_{N-1}^2$$

In this last expression we have $\alpha = \frac{a\,\Delta t}{2} = \frac{0.1 \cdot \Delta t}{2} > 0$ since Δt is positive. All of the v_j's are real numbers, so their squares are nonnegative. Finally, the fact that $\boldsymbol{v} \neq \mathbf{0}$ means that $v_j \neq 0$ (and so that $v_j^2 > 0$) for at least one $j = 1, \dots, N-1$. So the sum above will be strictly positive for every $\boldsymbol{v} \neq \mathbf{0}$.

This means the matrix $I + \alpha B_h$ is SPD.

If a matrix is SPD, then is has a Cholesky factorization, $I + \alpha B_h = R^T R$ where R is some upper triangular matrix with positive numbers on the diagonal. The system described by a triangular matrix is easy to solve using forward or back substitution. If we put the system of our PDE into Cholesky form, we can solve for our vector $\boldsymbol{U}_m^n$ in an efficient manner.

Stability

To analyze stability, we look for solution of the form

$$U_n^m = W^m e^{i2\pi p x_n}$$

where p is an integer. Now substitute this expression into the formula that describes the Crank-Nicolson method for a homogeneous problem.

$$\left(\frac{W^m - W^{m-1}}{\Delta t}\right) e^{i2\pi p x_n} - \frac{a}{2h^2}(W^m + W^{m-1})\left(e^{i2\pi p x_{n+1}} - 2e^{i2\pi p x_n} + e^{i2\pi p x_{n-1}}\right) = 0$$

$$\left(\frac{W^m - W^{m-1}}{\Delta t}\right) e^{i2\pi p x_n} - \frac{a}{2h^2}(W^m + W^{m-1})\left(e^{i2\pi p h} - 2 + e^{-i2\pi p h}\right) e^{i2\pi p x_n} = 0$$

Divide out the common factor of $e^{i2\pi p x_n}$

$$\frac{W^m - W^{m-1}}{\Delta t} - \frac{a}{2h^2}(W^m + W^{m-1})\left(e^{i2\pi p h} - 2 + e^{-i2\pi p h}\right) = 0$$

and then simplify using an identity derived in class

$$\frac{W^m - W^{m-1}}{\Delta t} - \frac{a}{2h^2}(W^m + W^{m-1})(-4\sin^2(\pi p h)) = 0$$

$$h^2 W^m - h^2 W^{m-1} + 2a\,\Delta t \sin^2(\pi p h)\, W^m + 2a\,\Delta t \sin^2(\pi p h)\, W^{m-1} = 0$$

$$(h^2 + 2a\,\Delta t \sin^2(\pi p h)) W^m = (h^2 - 2a\,\Delta t \sin^2(\pi p h)) W^{m-1}$$

$$W^m = \frac{h^2 - 2a\,\Delta t \sin^2(\pi p h)}{h^2 + 2a\,\Delta t \sin^2(\pi p h)} W^{m-1}$$

Substituting in the values used in this problem gives

$$W^m = \frac{h^2 - 2a\,\Delta t \sin^2(\pi p h)}{h^2 + 2a\,\Delta t \sin^2(\pi p h)} W^{m-1}$$

The ratio in the last equation is the amplification factor, and the method is absolutely stable if

$$\left| \frac{h^2 - 2a\,\Delta t \sin^2(\pi p h)}{h^2 + 2a\,\Delta t \sin^2(\pi p h)} \right| \leq 1$$

$$\text{the same as } \left| \frac{1 - \frac{2a\,\Delta t}{h^2}\sin^2(\pi p h)}{1 + \frac{2a\,\Delta t}{h^2}\sin^2(\pi p h)} \right| \leq 1\,.$$

This last relationship is true since

$$\frac{2a\,\Delta t}{h^2}\sin^2(\pi p h) \geq 0.$$

So the Crank-Nicolson method is absolutely stable.

Truncation Error

If u is the exact solution of the PDE, then the truncation error for the Crank-Nicolson method is defined as

$$\tau_n^{m-1/2} = \frac{u_n^m - u_n^{m-1}}{\Delta t} - \frac{a}{2}\left(\frac{u_{n+1}^m - 2u_n^m + u_{n-1}^m}{h^2} + \frac{u_{n+1}^{m-1} - 2u_n^{m-1} + u_{n-1}^{m-1}}{h^2} \right) - f_n^{m-1/2}\,.$$

Assuming u is sufficiently smooth, we can extend this definition to any x and t in the domain.

$$\begin{aligned} \tau(x,t) = {} & \frac{u(x,t+\Delta t) - u(x,t-\Delta t)}{2\Delta t} \\ & - \frac{a}{2}\left(\frac{u(x+h,t+\Delta t) - 2u(x,t+\Delta t) + u(x-h,t+\Delta t)}{h^2} \right. \\ & \left. + \frac{u(x+h,t-\Delta t) - 2u(x,t-\Delta t) + u(x-h,t-\Delta t)}{h^2} \right) - f(x,t) \end{aligned}$$

$$= \bar{\delta}u(\,x,\cdot) - \frac{a}{2}(\delta^2(\cdot, t+\Delta t) + \delta^2(\cdot, t-\Delta t)) - \left(\frac{\partial u}{\partial t} - a\frac{\partial^2 u}{\partial x^2}\right)$$

$$= \left[\bar{\delta}u(\,x,\cdot) - \frac{\partial u}{\partial t}\right] - a\left[\frac{1}{2}(\delta^2(\cdot, t+\Delta t) + \delta^2(\cdot, t-\Delta t)) - \frac{\partial^2 u}{\partial x^2}\right]$$

Again assuming smoothness, we can simplify the spatial part of the equation as $\Delta t \to 0$.

$$\tau(x,t) = \left[\bar{\delta}u(\,x,\cdot) - \frac{\partial u}{\partial t}\right] - a\left[\delta^2(\cdot, t) - \frac{\partial^2 u}{\partial x^2}\right]$$

Now we will apply the following relationships (shown in class)

$$\bar{\delta}f(t) - f'(t) = \frac{1}{6}\Delta t^2 f'''(t) + O(\Delta t^4)$$

$$\delta^2 f(x) - f''(x) = \frac{1}{12}h^2 f^{(4)}(x) + O(h^4)$$

to get

$$\tau(x,t) = \frac{1}{6}\Delta t^2 \frac{\partial^3 u}{dt^3}(x,t) + O(\Delta t^4) - a\left[\frac{1}{12}h^2\frac{\partial^4 u}{\partial x^4} + O(h^4)\right]$$

$$= \frac{\Delta t^2}{6}\frac{\partial^3 u}{dt^3}(x,t) - \frac{ah^2}{12}\frac{\partial^4 u}{\partial x^4}(x,t) + O(\Delta t^4 + h^4)$$

$$= O(\Delta t^2 + h^2).$$

So in this method the truncation error for this method is of the same order with regard to both time and space.

Testing an Exact Solution

Suppose the exact solution of the parabolic problem is

$$u(x,t) = [\sin(4x) + \cos(2x)][\cos(t) + \sin(t)].$$

So then we can take the necessary partials to calculate

$$f(x,t) = u_t - 0.1u_{xx}$$

$$= [\sin(4x) + \cos(2x)][-\sin(t) + \cos(\mathrm{t})] - 0.1[-16\sin(4x) - 4\ \cos(2x)][\cos(t) + \sin(t)]$$

The initial condition is

$$v(x) = u(x, 0) = \sin(4x) + \cos(2x) \text{ for } 2 \leq x \leq 3$$

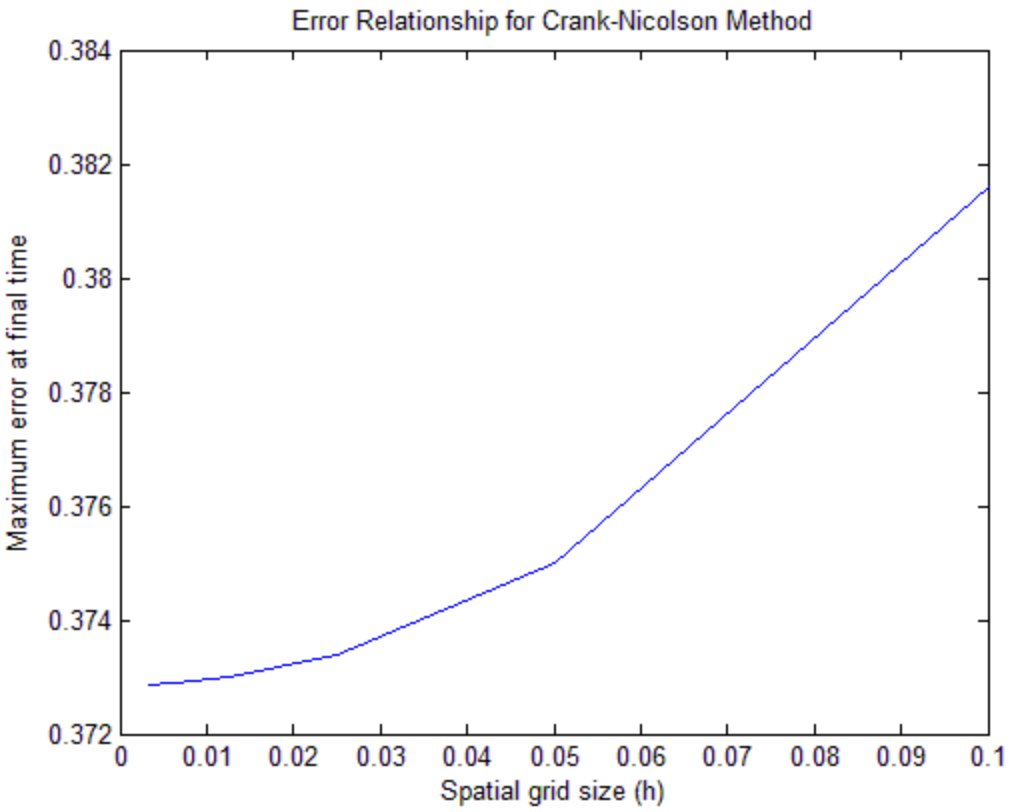
and the boundary conditions are

$$g_1(t) = u(2, t) = [\sin(8) + \cos(4)][\cos(t) + \sin(t)]$$
$$g_2(t) = u(3, t) = [\sin(12) + \cos(6)][\cos(t) + \sin(t)]$$

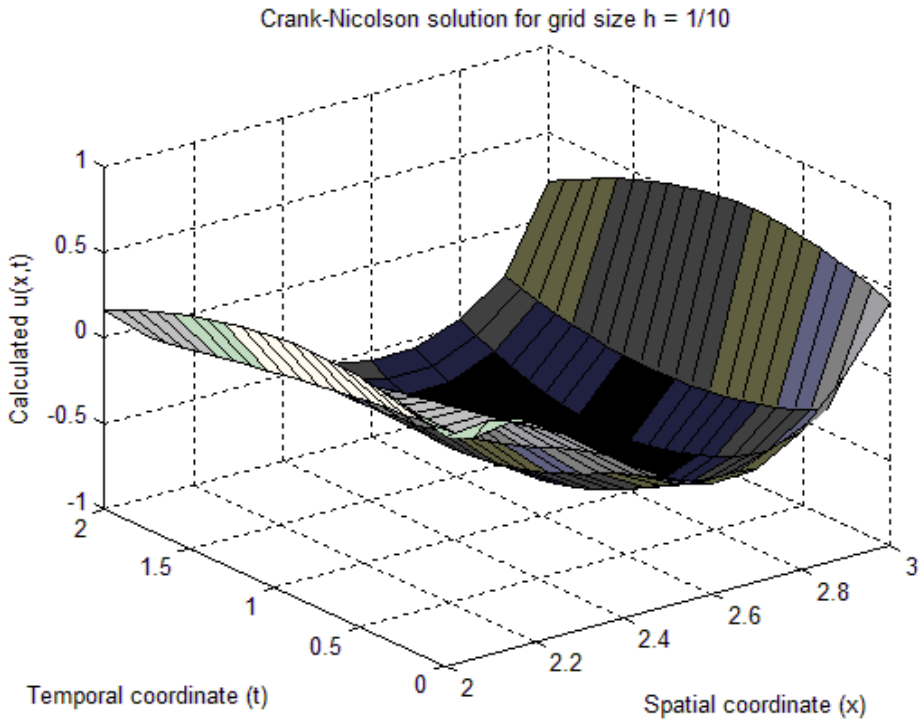
both on $0 < t \leq 2$.

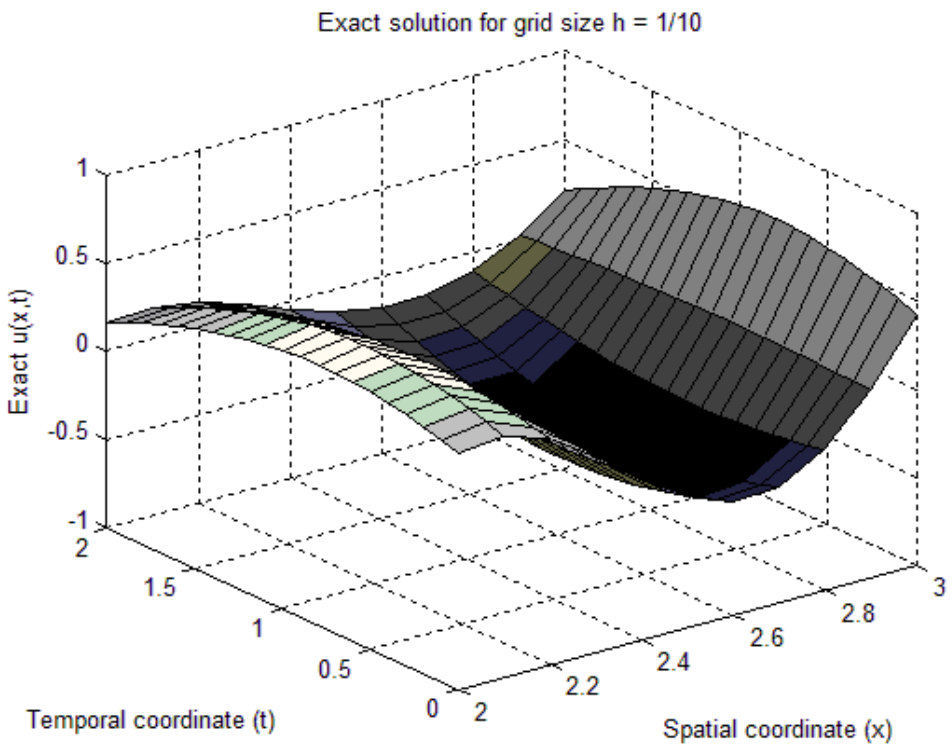
Error Analysis

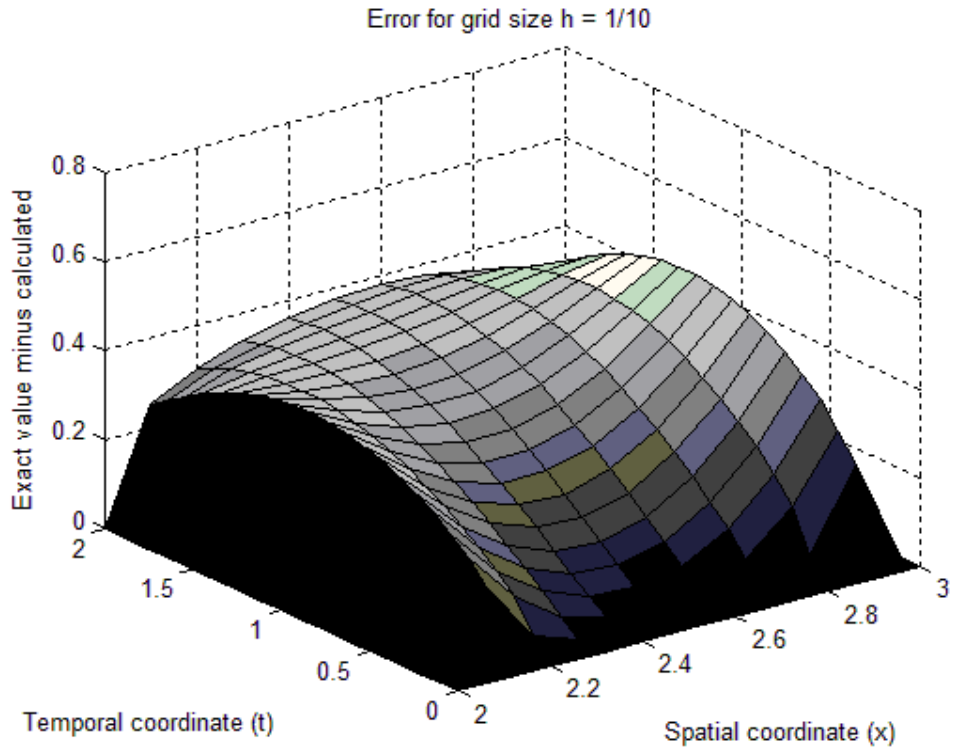
Using the test function above, the calculated solution was compared to the exact solution for a range of grid spacings. The figure below shows the maximum error at the final timestep for different trials.



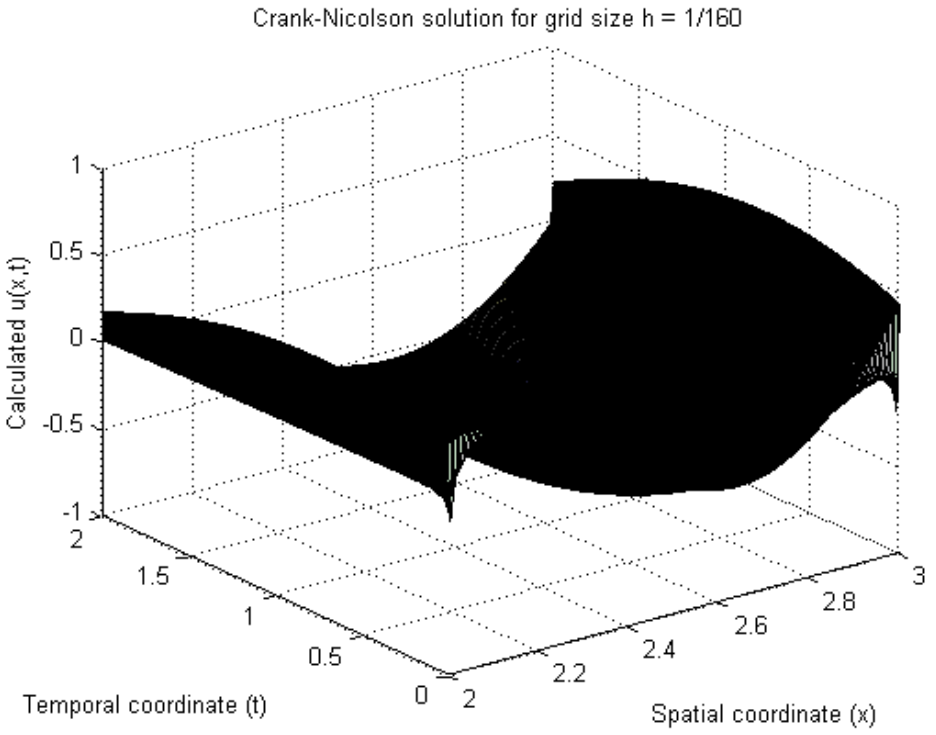
For the largest grid spacing, the Crank-Nicolson method produced the following result:

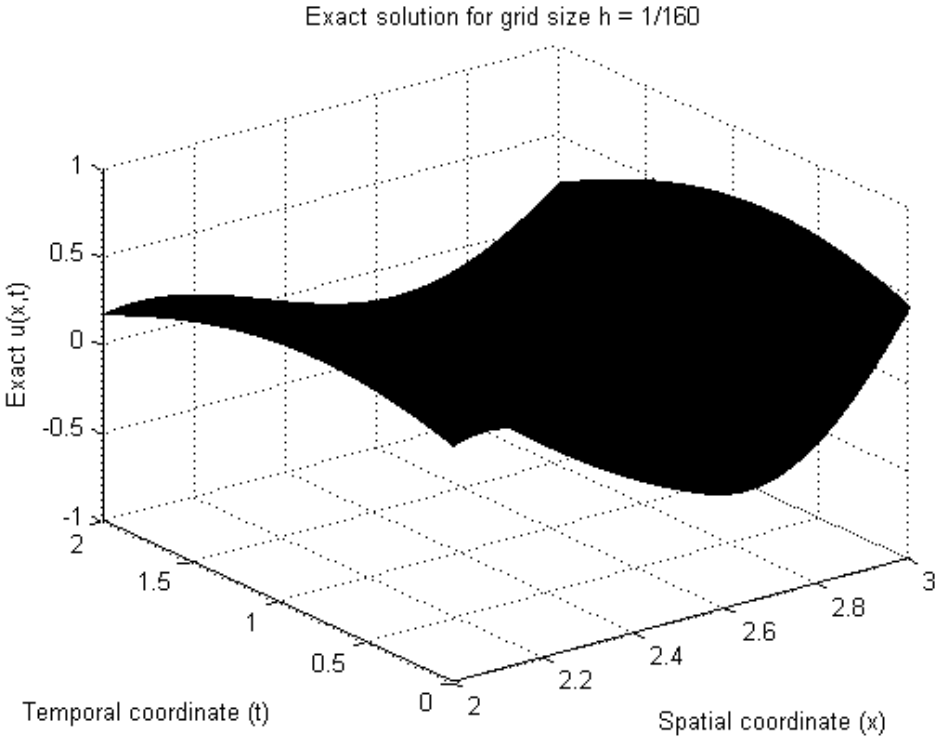


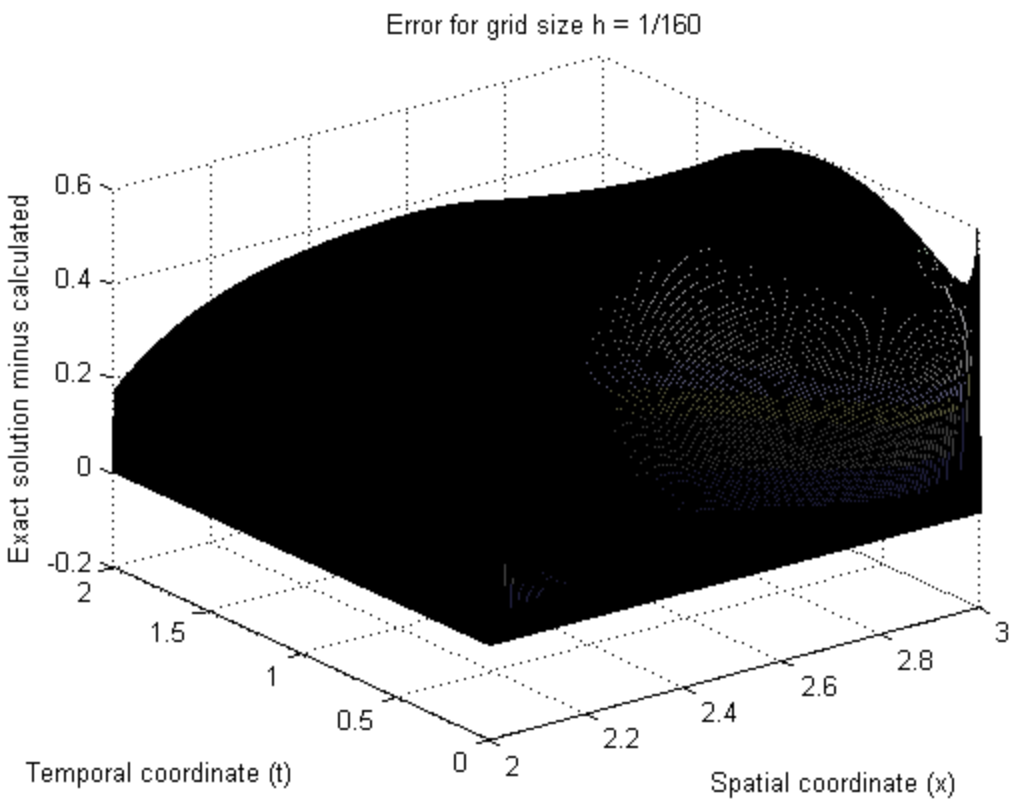

Exact solution for grid size h = 1/10
Exact u(x,t)
1
0.5
0
-0.5
-1
2
1.5
1
0.5
0
Temporal coordinate (t)
2
2.2
2.4
2.6
2.8
3
Spatial coordinate (x)


Error for grid size h = 1/10
Exact value minus calculated
0.8
0.6
0.4
0.2
0
2
1.5
1
0.5
0
Temporal coordinate (t)
2
2.2
2.4
2.6
2.8
3
Spatial coordinate (x)

Using a finer grid gave a different result:







Conclusions

The Crank-Nicolson method is a good approximation for the PDE in this example. A qualitative examination of the plots show that the estimated solution is a reasonable match to the exact function. Quantitatively, the graph shows that maximum error in the approximation decreases as the grid spacing decreases. The effect becomes more pronounced as more time passes.

The Crank-Nicoloson method is absolutely stable. This gives it an advantage over the explicit Euler method. CN also has convergence of order $O(\Delta t^2 + h^2)$. In this respect it better than the implicit Euler method.

APPENDIX: SOURCE CODE

Crank-Nicolson.m

```
function [ x, t, U ] = Crank_Nicolson( vstring, fstring, g1string, ...
                                       g2string, a, A, B, N, T, M )
% Crank-Nicolson method for solution of a 1D parabolic problem

% Input parameters
%   vstring: initial condition
%   fstring: driving term
%   g1string: boundary condition at lower endpoint
%   g2string: boundary condition at upper endpoint
%   a: positive parameter in heat equation
%   A: lower endpoint of space domain
%   B: upper endpoint
%   N: number of mesh points in the spatial grid
%   T: upper end of time scale - here we assume time starts at 0
%   M: number of steps in time grid

% Output
%   x: vector of space gridpoints
%   t: vector of time gridpoints
%   U: array of approximate solutions to the PDE at gridpoints

dt = T/M;                               % timestep
t = dt * [0:M];

h = (B - A)/N;                          % space step
x = A + h*[0:N]';

U = zeros(N+1, M+1);
U(:,1) = feval(vstring, x);             % fill in initial condition

% Use sparse matrix setup to make code more efficient
Bh = (1/h^2) * spdiags([-ones(N-1,1),2*ones(N-1,1),-ones(N-1,1)], ...
                        [-1,0,1], N-1, N-1);
R = chol(speye(N-1) + (a*dt/2)*Bh);

n = 2:N;
for m = 2:M+1
   b = (speye(N-1) - (a*dt/2)*Bh) * U(n,m-1) + ...
       dt * feval(fstring, x(n), (1/2)* (t(m-1)+t(m)));
   b = R'\b;
   U(n,m) = R\b;
   U(1,m) = feval(g1string, t(m));
   U(N+1,m) = feval(g2string, t(m));
end

end
```

assignment_1_vi.m

```
function [] = assignment_1_vi()
% Math 550, assignment 1
% Problem (vi)

% Store table data
table_data = zeros(6,4);

for j = 1:6

    k = j - 1.0;      % parameter for mesh
    N = 10.0 * 2.0^k;
    M = 2.0 * 10.0 * 2.0^k;     % twice as many for this dimension

    [x,t,U] = Crank_Nicolson('initial_data', 'source_term', ...
        'left_boundary_data', 'right_boundary_data', 0.1, 2, 3, N, 2, M);

    % Calculate  maximum error at final time
    exact = exact_data(2,3,N,2,M);
    max_err = max( abs( U(:,end)- exact(:,end) ) );
    % Store data for display in table
    table_data(j,1) = 0.1 / 2.0^(k);       % h
    table_data(j,2) = 0.1 / 2.0^(k);       % delta t
    table_data(j,3) = max_err;              % e infinity, delta t, h
    table_data(j,4) = 2 * (0.1 / 2.0^(k))^2;    % order of convergence

    % Store calculated values for 3D plots if k=0 or k=4
    if k == 0
        x0 = x;
        t0 = t;
        U0 = U;
        exact0 = exact;
    elseif k == 4
        x4 = x;
        t4 = t;
        U4 = U;
        exact4 = exact;
    end
end

% display table of error data
disp('    Delta t     h       max err     EOC');
disp(table_data);

% Make 2D plot of error relationship
figure(1);
plot(table_data(:,2), table_data(:,3))
title('Error Relationship for Crank-Nicolson Method')
xlabel('Spatial grid size (h)')
ylabel('Maximum error at final time')

% Make the sets of 3D plots
figure(2);
```

```
colormap(gray);
surf(x0, t0, U0');
title('Crank-Nicolson solution for grid size h = 1/10')
xlabel('Spatial coordinate (x)')
ylabel('Temporal coordinate (t)')
zlabel('Calculated u(x,t)')

figure(3);
colormap(gray);
surf(x0, t0, exact0');
title('Exact solution for grid size h = 1/10')
xlabel('Spatial coordinate (x)')
ylabel('Temporal coordinate (t)')
zlabel('Exact u(x,t)')

figure(4);
colormap(gray);
surf(x0, t0, exact0' - U0');
title('Error for grid size h = 1/10')
xlabel('Spatial coordinate (x)')
ylabel('Temporal coordinate (t)')
zlabel('Exact value minus calculated')

figure(5);
colormap(gray);
surf(x4, t4, U4');
title('Crank-Nicolson solution for grid size h = 1/160')
xlabel('Spatial coordinate (x)')
ylabel('Temporal coordinate (t)')
zlabel('Calculated u(x,t)')

figure(6);
colormap(gray);
surf(x4, t4, exact4');
title('Exact solution for grid size h = 1/160')
xlabel('Spatial coordinate (x)')
ylabel('Temporal coordinate (t)')
zlabel('Exact u(x,t)')

figure(7);
colormap(gray);
surf(x4, t4, exact4' - U4');
title('Error for grid size h = 1/160')
xlabel('Spatial coordinate (x)')
ylabel('Temporal coordinate (t)')
zlabel('Exact solution minus calculated')

end
```