Justin Hexem
AMATH 586

Homework #2

1.) We will use Von Neumann analysis to find when this scheme is L^2 stable.

Let $U_j^n = g(\xi)^n e^{i\xi j\Delta x}$.

Plugging this into our scheme gives

$$\frac{1}{2\Delta t}\left[g(\xi)^{n+1}e^{i\xi j\Delta x} - g(\xi)^{n-1}e^{i\xi j\Delta x}\right] = \frac{1}{\Delta x^2}\left[g(\xi)^n e^{i\xi(j+1)\Delta x} - 2g(\xi)^n e^{i\xi j\Delta x} + g(\xi)^n e^{i\xi(j-1)\Delta x}\right]$$

$$\Leftrightarrow g(\xi)^2 e^{i\xi j\Delta x} - e^{i\xi j\Delta x} = 2\frac{\Delta t}{\Delta x^2}\left[g(\xi)e^{i\xi j\Delta x}e^{i\xi\Delta x} - 2g(\xi)e^{i\xi j\Delta x} + g(\xi)e^{i\xi j\Delta x}e^{-i\xi\Delta x}\right]$$

$$\Leftrightarrow g(\xi)^2 - 1 = 2\frac{\Delta t}{\Delta x^2}\left[e^{i\xi\Delta x} - 2 + e^{-i\xi\Delta x}\right]g(\xi)$$

$$\Leftrightarrow g(\xi)^2 - 1 = 2\frac{\Delta t}{\Delta x^2}\left[2\cos(\xi\Delta x) - 2\right]g(\xi)$$

$$\Leftrightarrow g(\xi)^2 + 4\frac{\Delta t}{\Delta x^2}\left[1 - \cos(\xi\Delta x)\right]g(\xi) - 1 = 0$$

$$\Leftrightarrow g(\xi) = \frac{-4\frac{\Delta t}{\Delta x^2}(\cos(\xi\Delta x) - 1) \pm \sqrt{16\left(\frac{\Delta t}{\Delta x^2}\right)^2(1-\cos(\xi\Delta x))^2 + 4}}{2}$$

$$= -2\frac{\Delta t}{\Delta x^2}(1-\cos(\xi\Delta x)) \pm \sqrt{4\left(\frac{\Delta t}{\Delta x^2}\right)^2(1-\cos^2(\xi\Delta x))^2 + 1}$$

For this scheme to be L^2 stable, we require $|g(\xi)| \leq 1$ for both values of $g(\xi)$. Observe that

$$-2\frac{\Delta t}{\Delta x^2}(1-\cos(\xi\Delta x)) \leq 0 \quad \& \quad \sqrt{4\left(\frac{\Delta t}{\Delta x^2}\right)^2(1-\cos(\xi\Delta x))^2 + 1} > 0.$$

This means the $g(\xi)$ with negative sign will be the largest in magnitude, so this scheme is stable if

$$\left|-2\frac{\Delta t}{\Delta x^2}(1-\cos(\xi\Delta x)) - \sqrt{4\left(\frac{\Delta t}{\Delta x^2}\right)^2(1-\cos(\xi\Delta x))^2 + 1}\right| \leq 1$$

$$\Rightarrow \left|2\frac{\Delta t}{\Delta x^2}(1-\cos(\xi\Delta x)) + \sqrt{4\left(\frac{\Delta t}{\Delta x^2}\right)^2(1-\cos(\xi\Delta x))^2 + 1}\right| \leq 1$$

Observe that if $1-\cos(\xi\Delta x) = 0$, then the LHS $= 1$. If $1-\cos(\xi\Delta x) > 0$, then the LHS > 1, so this scheme is only stable if $1-\cos(\xi\Delta x) = 0$. This means $\xi\Delta x = 0$, which implies $\Delta x = 0$. Thus this scheme is never L^2 stable.

2.) We will use Von Neumann analysis to find when this scheme is L^2 stable.

Let $U_j^n = g(\xi)^n e^{i\xi j\Delta x}$.

Plugging this into our scheme gives

$$\frac{1}{\Delta t}\left[g(\xi)^{n+1}e^{i\xi j\Delta x} - g(\xi)^n e^{i\xi j\Delta x}\right] + \frac{b}{2\Delta x}\left[g(\xi)^{n+1}e^{i\xi(j+1)\Delta x} - g(\xi)^{n+1}e^{i\xi(j-1)\Delta x}\right]$$
$$= \frac{a}{\Delta x^2}\left[g(\xi)^{n+1}e^{i\xi(j+1)\Delta x} - 2g(\xi)^{n+1}e^{i\xi j\Delta x} + g(\xi)^{n+1}e^{i\xi(j-1)\Delta x}\right]$$

$$\Leftrightarrow \frac{1}{\Delta t}\left[g(\xi) - 1\right] + \frac{b}{2\Delta x}\left[g(\xi)e^{i\xi\Delta x} - g(\xi)e^{-i\xi\Delta x}\right] = \frac{a}{\Delta x^2}\left[g(\xi)e^{i\xi\Delta x} - 2g(\xi) + g(\xi)e^{-i\xi\Delta x}\right]$$

$$\Leftrightarrow \frac{1}{\Delta t}g(\xi) - \frac{1}{\Delta t} + \frac{b}{\Delta x}\cos(\xi\Delta x)g(\xi) = \frac{2a}{\Delta x^2}\left[\cos(\xi\Delta x) - 1\right]g(\xi)$$

$$\Leftrightarrow \left[\frac{1}{\Delta t} + \frac{b}{\Delta x}\cos(\xi\Delta x) - \frac{2a}{\Delta x^2}\cos(\xi\Delta x) + \frac{2a}{\Delta x^2}\right]g(\xi) = \frac{1}{\Delta t}$$

$$\Leftrightarrow g(\xi) = \frac{1}{\left[1 + 2a\frac{\Delta t}{\Delta x^2} + \left(b\frac{\Delta t}{\Delta x} - 2a\frac{\Delta t}{\Delta x^2}\right)\cos(\xi\Delta x)\right]}$$

The scheme is L^2 stable when $|g(\xi)| \leq 1$, so this means

$$|g(\xi)| = \left|\frac{1}{\left[1 + 2a\frac{\Delta t}{\Delta x^2} + \left(b\frac{\Delta t}{\Delta x} - 2a\frac{\Delta t}{\Delta x^2}\right)\cos(\xi\Delta x)\right]}\right| \leq 1$$

$$\Leftrightarrow 1 \leq \left|1 + 2a\frac{\Delta t}{\Delta x^2} + \left(b\frac{\Delta t}{\Delta x^2} - 2a\frac{\Delta t}{\Delta x^2}\right)\cos(\xi\Delta x)\right|$$

$$\leq |1| + \left|2a\frac{\Delta t}{\Delta x^2}\right| + \left|b\frac{\Delta t}{\Delta x^2} - 2a\frac{\Delta t}{\Delta x^2}\right||\cos(\xi\Delta x)|$$

$$\leq 1 + 2a\frac{\Delta t}{\Delta x^2} + \left|b\frac{\Delta t}{\Delta x^2} - 2a\frac{\Delta t}{\Delta x^2}\right|$$

$$\Leftrightarrow 0 \leq 2a\frac{\Delta t}{\Delta x^2} + \left|b\frac{\Delta t}{\Delta x^2} - 2a\frac{\Delta t}{\Delta x^2}\right|$$

Since $a > 0$, this means $2a\frac{\Delta t}{\Delta x^2} > 0$. Also, we see that $\left|b\frac{\Delta t}{\Delta x} - 2a\frac{\Delta t}{\Delta x^2}\right| \geq 0$. This means our condition for L^2 stability holds for all $a, \Delta t, \Delta x > 0$ and for all $b \in \mathbb{R}$, so this scheme is unconditionally L^2 stable.

```
>> Problem3ExpEuler

ErrorOrderForL2NormExplicitEuler =

    2.0066


ErrorOrderForLinfNormExplicitEuler =

    1.9997

>> Problem3ImpEuler

ErrorOrderForL2NormImplicitEuler =

    1.0062


ErrorOrderForLinfNormImplicitEuler =

    1.0164

>> Problem3CN

ErrorOrderForL2NormCN =

    2.0075


ErrorOrderForLinfNormCN =

    2.0133

>>
```

em3ExpEuler.m | Problem3ImpEuler.m | Problem3CN.m | +

```
xmin = -5;
xmax = 5;
Tmax = 3;
v = @(t, x) (1 ./ sqrt(4 * pi * t)) .* exp(-((x-2).^2) ./ (4 * t));

u0 = @(x) v(1, x);
uNegBdry = @(t) v(t + 1, -5);
uPosBdry = @(t) v(t + 1, 5);

Nvals = [20 40 80 160];
L2errors = zeros(1, length(Nvals));
LinfErrors = zeros(1, length(Nvals));

for iter = 1:length(Nvals)
    N = Nvals(iter);
    dx = (xmax - xmin) / N;
    dt = 0.4 * dx * dx;
    dt = Tmax / (ceil(Tmax / dt));

    xvals = ((0:N) * dx) + xmin;
    tvals = (0:(Tmax / dt)) * dt;
    [T, X] = meshgrid(tvals, xvals);

    U = zeros(length(xvals), length(tvals));
    U(:, 1) = u0(xvals);
    U(1, :) = uNegBdry(tvals);
    U(end, :) = uPosBdry(tvals);

    for k = 2:length(tvals)
        for j = 2:(length(xvals)-1)
            U(j, k) = (dt / (dx^2)) * (U(j+1, k-1) - 2*U(j, k-1) + U(j-1, k-1)) + U(j, k-1);
        end
    end

    trueSol = v(T+1, X);
    error = trueSol(:, end) - U(:, end);
    L2errors(iter) = sqrt(dx)*norm(error, 2);
    LinfErrors(iter) = norm(error, "inf");
end
dxvals = (xmax - xmin) ./ Nvals;
L2coeff = polyfit(log(dxvals), log(L2errors), 1);
LinfCoeff = polyfit(log(dxvals), log(LinfErrors), 1);
ErrorOrderForL2NormExplicitEuler = L2coeff(1)
ErrorOrderForLinfNormExplicitEuler = LinfCoeff(1)
```

```
xmin = -5;
xmax = 5;
Tmax = 3;

v = @(t, x) (1 ./ sqrt(4 * pi * t)) .* exp(-((x-2).^2) ./ (4 * t));

u0 = @(x) v(1, x);
uNegBdry = @(t) v(t + 1, -5);
uPosBdry = @(t) v(t + 1, 5);

Nvals = [20 40 80 160];
L2errors = zeros(1, length(Nvals));
LinfErrors = zeros(1, length(Nvals));

for iter = 1:length(Nvals)

    N = Nvals(iter);
    dx = (xmax - xmin) / N;
    dt = dx;
    dt = Tmax / (ceil(Tmax / dt));

    xvals = ((0:N) * dx) + xmin;
    tvals = (0:(Tmax / dt)) * dt;

    [T, X] = meshgrid(tvals, xvals);

    U = zeros(length(xvals)-2, length(tvals));
    U(:, 1) = u0(xvals(2:end-1));

    mainDiag = 1 + 2 * (dt/(dx^2)) * ones(1, N-1);
    subDiags = -(dt / (dx^2)) * ones(1, N-2);
    A = diag(mainDiag, 0) + diag(subDiags, -1) + diag(subDiags, 1);

    for j = 2:length(tvals)
        Uprev = U(:, j-1);
        Uprev(1) = Uprev(1) + (dt / (dx^2)) * uNegBdry(tvals(j));
        Uprev(end) = Uprev(end) + (dt / (dx^2)) * uPosBdry(tvals(j));

        U(:, j) = A \ Uprev;
    end

    Ufull = [uNegBdry(tvals); U; uPosBdry(tvals)];

    trueSol = v(T+1, X);
    error = abs(trueSol(:, end) - Ufull(:, end));
```

```
    for j = 2:length(tvals)
        Uprev = U(:, j-1);
        Uprev(1) = Uprev(1) + (dt / (dx^2)) * uNegBdry(tvals(j));
        Uprev(end) = Uprev(end) + (dt / (dx^2)) * uPosBdry(tvals(j));

        U(:, j) = A \ Uprev;
    end

    Ufull = [uNegBdry(tvals); U; uPosBdry(tvals)];

    trueSol = v(T+1, X);
    error = abs(trueSol(:, end) - Ufull(:, end));
    L2errors(iter) = sqrt(dx)*norm(error, 2);
    LinfErrors(iter) = norm(error, "inf");

end

dxvals = (xmax - xmin) ./ Nvals;

L2coeff = polyfit(log(dxvals), log(L2errors), 1);
LinfCoeff = polyfit(log(dxvals), log(LinfErrors), 1);
ErrorOrderForL2NormImplicitEuler = L2coeff(1)
ErrorOrderForLinfNormImplicitEuler = LinfCoeff(1)
```

```
xmin = -5;
xmax = 5;
Tmax = 3;

v = @(t, x) (1 ./ sqrt(4 * pi * t)) .* exp(-((x-2).^2) ./ (4 * t));

u0 = @(x) v(1, x);
uNegBdry = @(t) v(t + 1, -5);
uPosBdry = @(t) v(t + 1, 5);

Nvals = [20 40 80 160];
L2errors = zeros(1, length(Nvals));
LinfErrors = zeros(1, length(Nvals));

for iter = 1:length(Nvals)

    N = Nvals(iter);
    dx = (xmax - xmin) / N;
    dt = dx;
    dt = Tmax / (ceil(Tmax / dt));

    xvals = ((0:N) * dx) + xmin;
    tvals = (0:(Tmax / dt)) * dt;

    [T, X] = meshgrid(tvals, xvals);

    U = zeros(length(xvals)-2, length(tvals));
    U(:, 1) = u0(xvals(2:end-1));

    mainDiag = 1 + (dt/(dx^2)) * ones(1, N-1);
    subDiags = -0.5 * (dt / (dx^2)) * ones(1, N-2);
    A = diag(mainDiag, 0) + diag(subDiags, -1) + diag(subDiags, 1);

    for j = 2:length(tvals)
        Uprev = (1 - (dt / (dx^2))) * U(:, j-1) + 0.5 * (dt / (dx^2)) * [U(2:end, j-1); 0] + 0.5 * (dt / (dx^2)) * [0; U(1:end-1, j-1)];
        Uprev(1) = Uprev(1) + 0.5 * (dt / (dx^2)) * uNegBdry(tvals(j)) + 0.5 * (dt / (dx^2)) * uNegBdry(tvals(j-1));
        Uprev(end) = Uprev(end) + 0.5 * (dt / (dx^2)) * uPosBdry(tvals(j)) + 0.5 * (dt / (dx^2)) * uPosBdry(tvals(j-1));

        U(:, j) = A \ Uprev;
    end

    Ufull = [uNegBdry(tvals); U; uPosBdry(tvals)];

    trueSol = v(T+1, X);
    error = abs(trueSol(:, end) - Ufull(:, end));
```

```
    for j = 2:length(tvals)
        Uprev = (1 - (dt / (dx^2))) * U(:, j-1) + 0.5 * (dt / (dx^2)) * [U(2:end, j-1); 0] + 0.5 * (dt / (dx^2)) * [0; U(1:end-1, j-1)];
        Uprev(1) = Uprev(1) + 0.5 * (dt / (dx^2)) * uNegBdry(tvals(j)) + 0.5 * (dt / (dx^2)) * uNegBdry(tvals(j-1));
        Uprev(end) = Uprev(end) + 0.5 * (dt / (dx^2)) * uPosBdry(tvals(j)) + 0.5 * (dt / (dx^2)) * uPosBdry(tvals(j-1));

        U(:, j) = A \ Uprev;
    end

    Ufull = [uNegBdry(tvals); U; uPosBdry(tvals)];

    trueSol = v(T+1, X);
    error = abs(trueSol(:, end) - Ufull(:, end));
    L2errors(iter) = sqrt(dx)*norm(error, 2);
    LinfErrors(iter) = norm(error, "inf");

end

dxvals = (xmax - xmin) ./ Nvals;

L2coeff = polyfit(log(dxvals), log(L2errors), 1);
LinfCoeff = polyfit(log(dxvals), log(LinfErrors), 1);
ErrorOrderForL2NormCN = L2coeff(1)
ErrorOrderForLinfNormCN = LinfCoeff(1)
```