Math 586 Homework 3
Due May 5
By Marvyn Bailly (GitHub: MarvynB)

Problem 1 *Consider solving the following heat equation with "linked" boundary conditions*

$$\begin{cases} u_t = \frac{1}{2}u_{xx} \\ u(0,t) = su(1,t) \\ u_x(0,t) = u_x(1,t), \\ u(x,0) = \eta(x), \end{cases}$$

where $s \neq -1$. Recall that the MOL discretization with the standard second-order stencil can be written as

$$U'(t) = -\frac{1}{2h^2}AU(t) + \begin{bmatrix} \frac{U_0(t)}{2h^2} \\ 0 \\ \vdots \\ 0 \\ \frac{U_{m+1}(t)}{2h^2} \end{bmatrix}, \quad A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & \ddots & \ddots & \ddots \\ & & & -1 & 2 \end{bmatrix}.$$

The first boundary condition is naturally enforced via $U_0(t) = sU_{m+1}(t)$. Show that if we suppose

$$\frac{U_1(t) - U_0(t)}{h} = \frac{U_{m+1}(t) - U_m(t)}{h},$$

then the MOL system becomes

$$U'(t) = \frac{1}{2h^2}BU(t), \quad B = \begin{bmatrix} -2 + \frac{s}{1+s} & 1 & & & & \frac{s}{1+s} \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ \frac{1}{1+s} & & & & 1 & -2 + \frac{1}{1+s} \end{bmatrix}. \tag{1}$$

Solution.
Consider the heat equation with "linked" boundary conditions

$$\begin{cases} u_t = \frac{1}{2}u_{xx} \\ u(0,t) = su(1,t) \\ u_x(0,t) = u_x(1,t), \\ u(x,0) = \eta(x), \end{cases}$$

where $s \neq -1$. We will use the method of lines discretization with the standard second-order stencil which can be written as

$$U'(t) = -\frac{1}{2h^2}AU(t) + \begin{bmatrix} \frac{U_0(t)}{2h^2} \\ 0 \\ \vdots \\ 0 \\ \frac{U_{m+1}(t)}{2h^2} \end{bmatrix}, \quad A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & \ddots & \ddots & \ddots \\ & & & -1 & 2 \end{bmatrix}.$$

The first boundary condition is naturally enforced via $U_0(t) = sU_{m+1}(t)$. Now if we suppose

$$\frac{U_1(t) - U_0(t)}{h} = \frac{U_{m+1}(t) - U_m(t)}{h},$$

we can solve solve for U_0 and U_{m+1} in terms of U_1 and U_m. Observe that

$$\begin{aligned} \frac{U_1(t) - U_0(t)}{h} &= \frac{U_{m+1}(t) - U_m(t)}{h} \\ \iff U_1 - U_0 &= U_{m+1} - U_m \\ \iff U_1 - U_0 &= \frac{U_0}{s} - U_m \\ \iff U_0\left(\frac{s+1}{s}\right) &= U_1 + U_m \\ \iff U_0 &= \left(\frac{s}{s+1}\right)(U_1 + U_m). \end{aligned}$$

And similarly, we find that

$$\begin{aligned} \frac{U_1(t) - U_0(t)}{h} &= \frac{U_{m+1}(t) - U_m(t)}{h} \\ \iff U_1 - U_0 &= U_{m+1} - U_m \\ \iff U_1 - sU_{m+1} &= U_{m+1} - U_m \\ \iff U_{m+1} &= \left(\frac{1}{s+t}\right)(U_1 + U_m). \end{aligned}$$

Thus when we apply the finite-centered difference of the form

$$\frac{U_{j+1} - 2U_j + U_{j-1}}{h^2}$$

we no longer need to use the boundary conditions on the edge cases $j = 1, m$. For example when $j = 1$ we have

$$\frac{U_2 - 2U_1 + U_0}{h^2} = \frac{U_2 - 2U_1 + \left(\frac{s}{s+1}\right)(U_1 + U_m)}{h^2},$$

and when $j = m$ we have

$$\frac{U_{m+1} - 2U_m + U_{m-1}}{h^2} = \frac{\left(\frac{1}{s+t}\right)(U_1 + U_m) - 2U_m + U_{m-1}}{h^2}.$$

Thus MOL discretization becomes

$$U'(t) = \frac{1}{2h^2} BU(t), \quad B = \begin{bmatrix} -2 + \frac{s}{1+s} & 1 & & & & \frac{s}{1+s} \\ 1 & -2 & 1 & & & \\ & 1 & -2 & 1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ \frac{1}{1+s} & & & & 1 & -2 + \frac{1}{1+s} \end{bmatrix},$$

where we also moved the minus sign into B. $\square$

Problem 2 *Apply the backward Euler method to* (1) *to give*

$$\left(I - \frac{k}{2h^2}B\right)U^{n+1} = U^n, \quad U^n = \begin{bmatrix} U_1^n \\ U_2^n \\ \vdots \\ U_m^n \end{bmatrix}, \tag{2}$$

and write a routine to solve the system (1) *with the initial condition*

$$\eta(x) = e^{-20(x-1/2)^2},$$

using $k = h$ *and* $h = 0.001$ *with* $s = 2$*. Plot the solution at times* $t = 0.001, 0.01, 0.1$*.*

Solution.
Recall that the backward Euler method is defined by

$$U^{n+1} = U^n + \frac{k}{2h^2}BU^{n+1}, \quad U^n = \begin{bmatrix} U_1^n \\ U_2^n \\ \vdots \\ U_m^n \end{bmatrix}.$$

rearranging the terms we get

$$U^{n+1}\left(I - \frac{k}{2h^2}B\right) = U^n.$$

Now we wish to write a routine to solve (1) with the initial condition

$$\eta(x) = e^{-20(x-1/2)^2},$$

using $k = h$ and $h = 0.001$ with $s = 2$. To create the routine, I used Julia and implemented the Backward Euler method using the following code:

```
function BE(m,uInit,T)
  h = 1/(m+1);
  k = h;
  s = 2;
  xs = (h:h:1—h)
  #create B
  B =  spzeros(m,m) + SymTridiagonal(fill(—2.0,m),fill(1.0,m—1));
  B[1,1] += s/(1+s);
  B[1,end] += s/(1+s);
  B[end,end] += 1/(1+s);
  B[end,1] += 1/(1+s);

```

```
13
14      #BE
15      u = uInit
16      for i = k:k:T
17          uNext = (I(m) - k/(2h^2) * B)\u
18          u = uNext
19      end
20      return (u,xs)
21  end
```

Running the backward Euler function for $t = 0.001, 0.01, 0.1$ and graphing the results, see Figure 1, we see that the solution behaves as expected as time evolves.

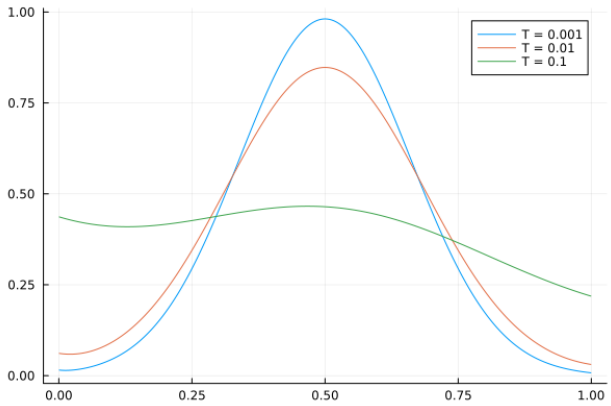


Figure 1: Backward Euler Solution to equation (1) at different times. Blue is the solution at $t = 0.001$, orange at $t = 0.01$, and green at $t = 0.1$.

□

Problem 3 *In the next two problems, you will use the heat equation to assist with a statistics problem.*

- *Consider data points $X_1, X_2, \ldots, X_N, \ldots$ each being a real number arising from a repeated experiment. We may want to know what probability distribution (if any) they come from. One way of coming up with an approximation to the density is to use*

$$\frac{1}{N}\sum_{j=1}^{N}\frac{1}{\sqrt{2\pi t}}\exp\left(-\frac{(x-X_j)^2}{2t}\right), \quad t>0. \tag{3}$$

Use normally distributed random data (`X = randn(n)` *in* `Julia`, `X = randn(n,1)` *in* `Matlab` *and* `X = numpy.random.randn(n,1)` *in* `Python`*) with $n = 10000$ and plot this function for $t = 0.001, 0.01, 0.1, 1, 10$ and compare it with the true probabilty density function for the data: $\rho(x) = \frac{1}{\sqrt{2\pi}}e^{-x^2/2}$. Visually, which "time" t gives the best approximation?*

Note: The solution of the heat equation $u_t = \frac{1}{2}u_{xx}$ with initial condition $u(x,0) = \delta(x)$ where δ is the standard Dirac delta function is given by $u(x,t) = \frac{1}{\sqrt{2\pi t}}\exp\left(-\frac{x^2}{2t}\right)$. So (3) can be seen as the solution of $u_t = \frac{1}{2}u_{xx}$ with

$$u(x,0) = \frac{1}{N}\sum_{j=1}^{N}\delta(x-X_j).$$

- *The previous approach works well if the underlying distribution is smooth and decays exponentially in both directions. But there are physical situations within cell biology, in particular, where the density should only be non-zero on a finite interval $[0,1]$ and satisfy some natural boundary conditions:*

$$\rho(0) = s\rho(1), \quad \rho'(0) = \rho'(1).$$

An example of such a function for $s = 2$ is given by

$$\rho(x) = -\frac{2}{3}x + \frac{4}{3} + \frac{1}{2}\sin(2\pi x).$$

Code to generate $X_1, X_2, \ldots, X_N, \ldots$ with this probability density in our three languages is given at the end of the homework. Repeat the calculation in the previous part with this data, $X_1, X_2, \ldots$.

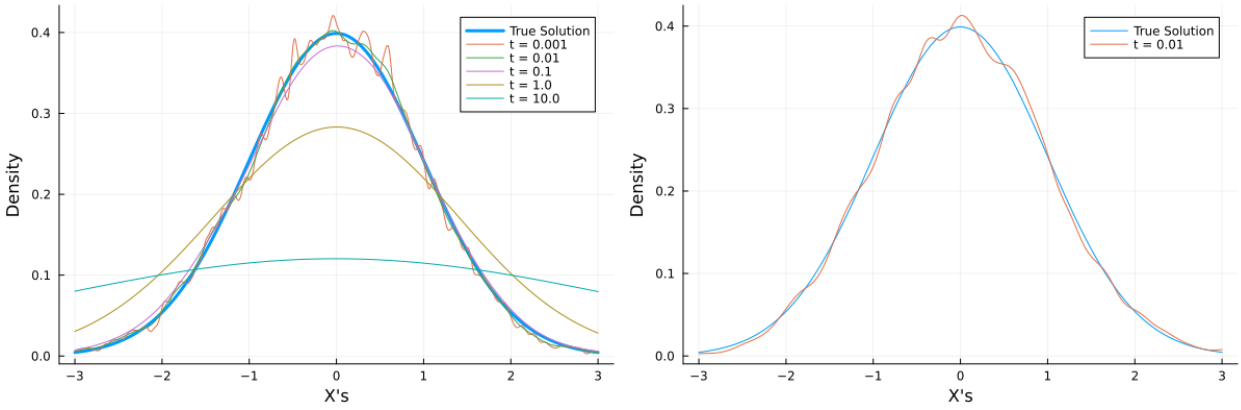
Solution.
we wish to create a routine to approximate the density of normally distributed data points using equation 3. To achieve this, I will use Julia, where `randn(n)` is used to generate n normally distributed random data points. We will use $n = 10000$. I implemented 3 and the true solution and their plotting in Julia with the following code:

```
getTrueDensity = (x) -> 1/(sqrt(2 * pi))*exp(-x^2/2)
getApprox = (n,xs,t,points) -> map(xs -> sum(point -> begin (1/n)*(1/(
    sqrt(2*pi*t)))*exp(-(xs - point)^2/(2t)) end, points), xs)

n = 10000;
points = randn(n);
xs = -3:0.01:3;
ts = [0.001,0.01,0.1,1,10];

graph = plot(xs,map(getTrueDensity,xs),xlabel="X's",ylabel="Density",
    label="True Solution");
for i=1:length(ts)
    graph = plot!(xs,getApprox(n,xs,ts[i],points),label="t = "*string(ts
        [i]));
end
```

Plotting the solutions with $t = 0.001, 0.01, 0.1, 1, 10$ compared to the true solution, as seen in figure 2, we see that $t = 0.01$ gives the best approximation.



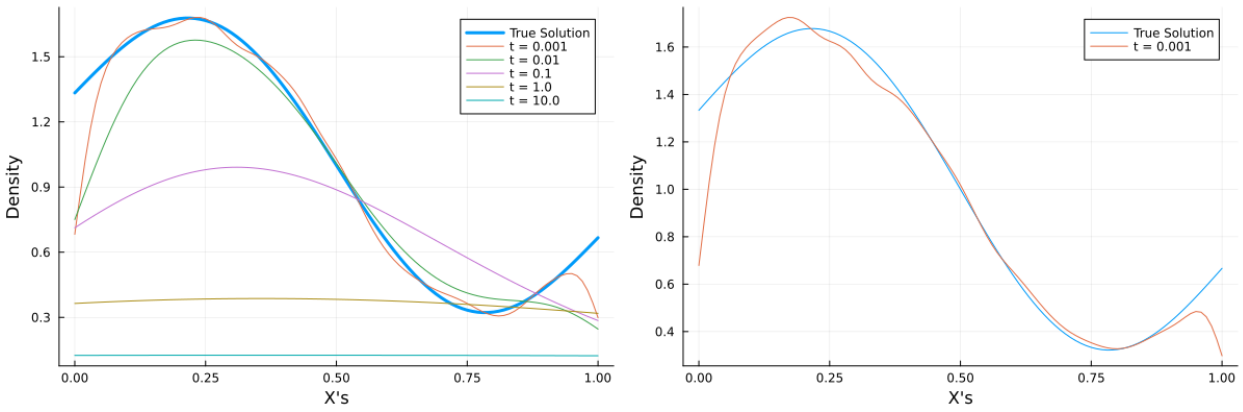
(a) (b)

Figure 2: Plotting solution from 3 compared to the true solution of normally distributed data. In (a), we compared the true solution with $t = 0.001, 0.01, 0.1, 1, 10$ while in (b) we see that $t = 0.01$ gives the closest approximation to the true solution.

We repeat the process but now using data points sampled from `prand` which has distribution of the form

$$\rho(x) = -\frac{2x}{3} + \frac{4}{3} + \frac{1}{2}\sin(2\pi x). \tag{4}$$

Plotting the approximate solutions for $t = 0.001, 0.01, 0.1, 1, 10$ compared to the true solution, as seen in figure 3, we see that $t = 0.001$ now best approximates the true solution.



(a) (b)

Figure 3: Plotting solution from 3 compared to the true solution where the data is generated from prand. In (a), we compared the true solution with $t = 0.001, 0.01, 0.1, 1, 10$ while in (b) we see that $t = 0.001$ gives the closest approximation to the true solution.

```
function prand(m)
  p = x -> -(2.0/3)* x .+ 4.0/3 .+ .5*sin.(2* pi * x )
  B = 1.7
  out = fill(0.,m)
      for j = 1: m
          u = 10.
          y = 0.
          while u >= p(y)/ B
              y = rand()
              u = rand()
          end
      out[j] = y
  end
  return out
end
```

□

Problem 4 *Consider binning data $X_1, X_2, \ldots, X_N$, $X_j \in (0,1)$ as follows:*

- *Find Y_i so that Y_i is the number of data points X_j that lie in the interval $[ih, (i+1)h) = [x_i, x_{i+1})$.*
- *Set $U_i^0 = \frac{Y_i}{hN}$.*

With $N = m$, $h = 0.0001$, $k = 10h$, $s = 2$, generate $X_1, \ldots, X_N$ using the `prand` *function, and bin the data to get the initial condition U_i^0, $i = 1, 2, \ldots, m$ for the MOL discretization (1). Solve with this initial condition using your code from* ***Problem 2*** *to times $t = 0.001, 0.01, 0.1$. Compare this with Part 2 of Problem 3.*

Solution.
We now wish to use the Backward Euler method we implement in Problem 2 to approximate the distribution of the `prand` function. We will use the same code from Problem 2 with $N = m, h = 0.0001, k = 10h,$ and $s = 2$. Next, we generate $X_1, \ldots, X_N$ data points using the `prand` function and bin the data as described in the problem statement. We used the following Julia code to bin our data:

```
function binData(data)
  Y = zeros(length(data))
  for i=1:length(data)
      for j=1:m
          if data[i] >= j*h && data[i] < (j +1)*h
              Y[j] += 1
          end
      end
  end
  uInit = Y./(h*m)
  return uInit
end
```

We now run Backward Euler with the new initial condition for times $t = 0.001, 0.01, 0.1$ and compare them to the true solution given by Equation 4. Observing the graph, seen in Figure 4, we see that the solutions are similar to those seen in Figure 3a expect the Backward Euler method handles the boundary condition at $x = 1$ significantly better than equation 3.

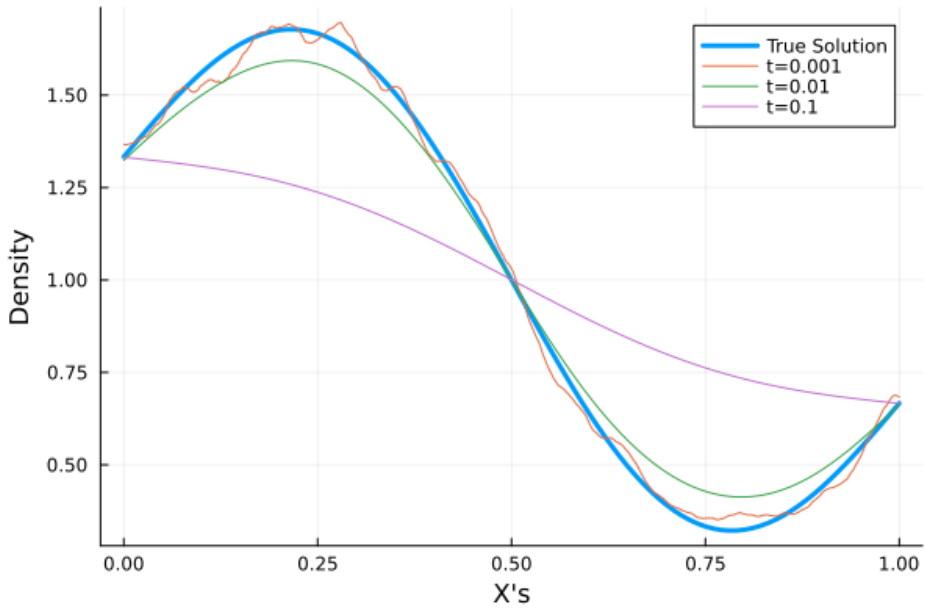


Figure 4: Backward Euler in conjugation with binning is used to approximate the distribution of data generated from prand. The true solution shown in blue is graphed from 4.

□

Problem 5 *Suppose the following is true: For $s > 0$, if y is a vector with non-negative entries and $\left(I - \frac{k}{2h^2}B\right)x = y$ then x has non-negative entries.*

For $s > 0$, establish the following:

- $\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} B = 0$ *and therefore* $\sum_j U_j^n = \sum_j U_j^0$ *for all* n.
- *Show that* (2) *is Lax-Richtmyer stable in the* 1*-norm,* $\|u\|_1 = h\sum_{i=1}^m |u_i|$.

Solution.
Let $s > 0$. We assume that if y has non-negative entries and $(I - \frac{k}{2h^2}B)x = y$ (same B from Equation 1), then x has non-negative entries. We first wish to show that $\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} B = 0$. Observe that

$$\left(\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} B\right)_1 = -2 + \frac{s}{1+s} + 1 + \frac{1}{1+s} = -1 + \frac{s}{1+s} + \frac{1}{1+s} = 0,$$

and

$$\left(\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} B\right)_m = \frac{s}{1+s} + 1 - 2 + \frac{1}{1+s} = -1 + 1 = 0,$$

and

$$\left(\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} B\right)_j = -2 + 1 + 1 = 0,$$

for $1 < j < m$. Thus we have that

$$\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} B = 0.$$

To show $\sum_j U_j^n = \sum_j U_j^0$, note that using $\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} B = 0$ we have

$$\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}(I - \frac{k}{2h^2}B) = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} - \frac{k}{2h^2}\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} B = \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}.$$

Since $(I - \frac{k}{2h^2}B)U^{n+1} = U^n$, for $n = 0, 1, 2, \ldots$, inductively we get that

$$\left(I - \frac{k}{2h^2}B\right)^n U^n = U^0, \quad \forall n.$$

Observe that multiplying both sides by $\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}$ yields

$$\begin{aligned}\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} \cdot U^0 &= \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} \cdot \left(I - \frac{k}{2h^2}B\right)^n U^n \\ &= \left(\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}\left(I - \frac{k}{2h^2}B\right)\right) \cdot \left(I - \frac{k}{2h^2}B\right)^{n-1} U^n \\ &= \begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix} \cdot \left(I - \frac{k}{2h^2}B\right)^{n-1} U^n\end{aligned}$$

$$
\begin{aligned}
&\vdots \\
&= \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \cdot \left(I - \frac{k}{2h^2} B \right) U^n \\
&= \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \cdot U^n.
\end{aligned}
$$

Therefore we have shown that

$$
\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \cdot U^0 = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \cdot U^n \implies \sum_j U_j^0 = \sum_j U_j^n.
$$

Next we wish to show that equation 2 is Lax-Richtmyer stable in the $1-$norm. Recall that Lax-Richtmyer stability is defined as

$$
\left\| \left(I - \frac{k}{2h^2} B \right)^{-n} \right\|_1 \leq c_T,
$$

where $c_T > 0$. Let's begin by showing that $\left(I - \frac{k}{2h^2} B\right)$ is invertible by showing that the kernel is trivial. Let x be in the null space of $\left(I - \frac{k}{2h^2} B\right)$, then by definition

$$
\left(I - \frac{k}{2h^2} B \right) x = 0,
$$

and since 0 has nonzero entries, we also know x has nonzero entries by our first assumption. Next multiplying both sides by $\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}$ yields

$$
\begin{aligned}
0 &= \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \left(I - \frac{k}{2h^2} B \right) x \\
&= \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} Ix - \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} Bx \\
&= \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} x \\
&= \sum_j x_j,
\end{aligned}
$$

and since $x_j > 0$ for all j and $\sum_j x_j = 0$, then $x_j = 0$ for all j. Thus the kernel is trivial and the inverse exists. Let $A = \left(I - \frac{k}{2h^2} B\right)^{-1}$. Then observe that for an arbitiary u^0 we have

$$
\begin{aligned}
\|A^n u^0\|_1 &= \|A^n \sum_j u_j^0 e_j\|_1 \\
&\leq \sum_j \|A^n u_j^0 e_j\|_1 \\
&= h \sum_j |u_j^0| \sum_i (|A^n e_j|)_i,
\end{aligned}
$$

where e_j has zeros in all entries expect for jth entry being one. Thus e_j has nonnegative entries. Since $\sum_i (A^n u^0)_i = \sum_i (u^n)_i = \sum_i (u^0)_i$, letting $e_j = u^0$ gives that $\sum_i (|A^n e_j|)_i = \sum_i (|e_j|)_i = 1$ for all j. Thus we have that

$$\|A^n u^0\|_1 \leq h \sum_j |u_j^0| \sum_i (|Ae_j|)_i = h \sum_j |u_j^0| = \|u^0\|_1.$$

This implies that

$$\|A^n\|_1 \leq \frac{\|A^n u^0\|_1}{\|u^0\|_1} \leq 1,$$

therefore if we let $c_T = 1$ we have that

$$\|A^n\|_1 < c_T$$

and thus equation 2 is L-R stable.

□

Problem 6 *Consider the bi-infinite matrix*

$$L = \begin{bmatrix} \ddots & \vdots & \vdots & \vdots & \iddots \\ \cdots & L_{-1,-1} & L_{-1,0} & L_{-1,1} & \cdots \\ \cdots & L_{0,-1} & L_{0,0} & L_{0,1} & \cdots \\ \cdots & L_{1,-1} & L_{1,0} & L_{1,1} & \cdots \\ \iddots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

Suppose the matrix L defines a bounded linear operator on $\ell^2(\mathbb{Z})$ via matrix-vector multiplication with

$$\ell^2(\mathbb{Z}) \ni V = \begin{bmatrix} \vdots \\ V_{-1} \\ V_0 \\ V_1 \\ \vdots \end{bmatrix}.$$

Show that L is translation invariant if and only if $L_{i,j} = c_{i-j}$ for a sequence $(c_j)_{j=-\infty}^{\infty}$, i.e., L is a Toeplitz matrix. Hint: Apply L to the standard basis vectors.

Solution.
Consider the bi-infinite matrix

$$L = \begin{bmatrix} \ddots & \vdots & \vdots & \vdots & \iddots \\ \cdots & L_{-1,-1} & L_{-1,0} & L_{-1,1} & \cdots \\ \cdots & L_{0,-1} & L_{0,0} & L_{0,1} & \cdots \\ \cdots & L_{1,-1} & L_{1,0} & L_{1,1} & \cdots \\ \iddots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

Suppose the matrix L defines a bounded linear operator on $\ell^2(\mathbb{Z})$ via matrix-vector multiplication with

$$\ell^2(\mathbb{Z}) \ni V = \begin{bmatrix} \vdots \\ V_{-1} \\ V_0 \\ V_1 \\ \vdots \end{bmatrix}.$$

Let e_j be the standard basis vector where the jth element is one and all others are zero and let $(L)_k$ denote the kth row of L. Then

$$(Le_j)_i = (L_j)_i = L_{i,j}.$$

Now consider the shift operator S_l where $l \in \mathbb{Z}$. Then $S_l e_j = e_{j+l}$ and note that due to the construction of L, $(S_{-l}L)_i = (L)_{i+l}$. Then we have

$$(S_{-l}LS_l e_j)_i = (LS_l e_j)_{i+l} = (Le_{j+l})_{i+l} = L_{i+l,j+l}.$$

Since the e_j's form a basis for $\ell^2(\mathbb{Z})$, any $V \in \ell^2(\mathbb{Z})$ can be decomposed into a sum of e_j's. Thus we have that

$$LV = (S_{-l}LS_l)V \iff L_{i,j} = L_{i+l,j+l}.$$

Therefore L is shift-invariant if and only if $L_{i,j} = L_{i+l,j+l}$ which is the definition of a Toeplitz matrix, i.e. $L_{i,j} = c_{i-j}$ for a sequence $(c_j)_{j=-\infty}^{\infty}$.

□

Problem 7 *A challenge (extra credit, total score cannot exceed 20): In the notation of Problem 2, for $s > 0$, establish:*

- *If y is a vector with non-negative entries and $\left(I - \frac{k}{2h^2}B\right)x = y$ then x has non-negative entries.*

Note that this shows that if $\sum_j U_j^0 = 1$ then at each step n we can interpret U_j^n as the evolution of a probability distribution.

Solution. This is a solution $\square$