1. Periodic tridiagonal. In class, we saw how to write a Gaussian elimination routine for a tridiagonal matrix. Here we build on that, with one little twist. Consider the system below:

\[
\begin{bmatrix}
  b_0 & c_0 & a_0 \\
  a_1 & b_1 & c_1 \\
  & a_2 & b_2 & c_2 \\
  & & \\n  & & \\n  & & \\
  & & \\
  c_N & a_{N-1} & b_{N-1} & c_{N-1} & a_N & b_N \\
\end{bmatrix}
\begin{bmatrix}
  x_0 \\
  x_1 \\
  x_2 \\
  \\
  \\
  x_N \\
\end{bmatrix}
= 
\begin{bmatrix}
  d_0 \\
  d_1 \\
  d_2 \\
  \\
  \\
  d_N \\
\end{bmatrix}
\] (1)

Now there are two additional elements, in the far corners. Also note that as written, this is an \(N+1 \times N+1\) matrix. This type of matrix arises when you enforce periodic boundary conditions on a system (for example, if we had done so in the cubic splines instead of the natural boundary conditions).

We can solve this type of linear system efficiently using a method called the Thomas algorithm. First we eliminate the last row and last column. This gives an \(N \times N\) system of the form:

\[
\begin{bmatrix}
  b_0 & c_0 & a_0 \\
  a_1 & b_1 & c_1 \\
  & a_2 & b_2 & c_2 \\
  & & \\n  & & \\n  & & \\
  & & \\
  a_{N-2} & b_{N-2} & c_{N-2} & a_{N-1} & b_N \\
\end{bmatrix}
\begin{bmatrix}
  x_0 \\
  x_1 \\
  x_2 \\
  \\
  \\
  x_N-2 \\
\end{bmatrix}
= 
\begin{bmatrix}
  d_0 \\
  d_1 \\
  d_2 \\
  \\
  \\
  d_N-2 \\
\end{bmatrix}
+ 
\begin{bmatrix}
  -a_0 \\
  0 \\
  0 \\
  \\
  \\
  -c_{N-1} \\
\end{bmatrix}
\begin{bmatrix}
  x_N \\
\end{bmatrix}
\] (2)

Here \(\eta\) arises from looking at which terms in the first \(N\) rows of the original matrix involve the \(x_N\) element in the solution vector that we are cutting out. Note that the \(\eta\) vector multiplies \(x_N\).

This system is linear, so we can imagine a solution of the form:

\[
\tilde{x} = x^{(a)} + x^{(\beta)}x_N
\] (3)

Substituting this into our new linear system, we see:

\[
A^c\tilde{x} = A^c x^{(a)} + A^c x^{(\beta)}x_N = \tilde{d} + \eta x_N
\] (4)
This means that we can solve
\[ A^c x^{(a)} = \tilde{d} \] (5)
\[ A^c x^{(\beta)} = \eta \] (6)
to get \( x^{(a)} \) and \( x^{(\beta)} \).

(a) With \( x^{(a)} \) and \( x^{(\beta)} \) found, we can find \( x_N \) by looking at the last row of our original system:
\[ c_N x_0 + a_N x_{N-1} + b_N x_N = d_N \] (7)
Substitute in the definition of \( x_0 \) and \( x_{N-1} \) from our vector \( \tilde{x} \) and show that
\[ x_N = \frac{d_N - c_N x_0^{(a)} - a_N x_{N-1}^{(a)}}{c_N x_0^{(\beta)} + a_N x_{N-1}^{(\beta)} + b_N} \] (8)

(b) We can now get the remaining values of \( x \) as \( \tilde{x} = x^{(a)} + x^{(\beta)} x_N \).

Write a code to solve this periodic tridiagonal system. Test it by writing a corresponding matrix-vector multiply code and find the solution for a matrix of the form \( b_i = 4 \) and \( a_i = c_i = 1 \) for \( i = 0, \ldots, N \).

2. (Matrix inverse) (based on Garcia) An iterative method for constructing the matrix inverse can be found via Newton’s method. A single step in the iteration appears as:
\[ X^{(k+1)} = 2X^{(k)} - X^{(k)}AX^{(k)} \] (9)
where \( A \) is the matrix whose inverse we seek and \( X^{(k)} \) is our current guess for the inverse.

Find the inverse of the following matrix using this technique:
\[
A = \begin{pmatrix}
4 & 3 & 4 & 10 \\
2 & -7 & 3 & 0 \\
-2 & 11 & 1 & 3 \\
3 & -4 & 0 & 2
\end{pmatrix}
\] (10)
You will need to supply an initial guess and a desired tolerance. Be careful with your initial guess—some choices will diverge. Think about the operations that are taking place in the first iteration to help you devise a good initial guess for \( A^{-1} \). It will probably help to put some prints in there (or work out a \( 2 \times 2 \) case analytically).