Aims: To solve the 2D Poisson equation and similar by central differences, and MATLAB’s \ operator, and to create awareness of sparse methods and use of spdiags (LN§ 4 on page 31).

1. Consider the 2D BVP

\[ \nabla^2 u(x, y) + 9u = 9xy, \quad 0 < x, y < 1, \quad (1) \]
\[ u(0, y) = u(x, 0) = 0, \quad (2) \]
\[ u(1, y) = u(x, 1) = 1. \quad (3) \]

Equation (1) is an inhomogeneous Helmholtz equation. Suppose that this BVP is to be approximated using 2nd order central differences applied at the 4 internal points of a uniform 2D grid \( x_i = ih, \quad (i = 0, \ldots, 3), \quad y_j = jh, \quad (j = 0, \ldots, 3) \).

(a) The grid is outlined below with nodes numbered similarly to Figure 14 on page 32. Write in the actual values of \( x_i \) and \( y_i \), alongside the generic labels \( x_0, \ldots, x_3 \), \( y_0, \ldots, y_3 \) shown in the diagram, and fill in the missing boundary values where “\( u = \ldots \)” appears. Also, write down the value of \( h \).

(b) Use equations LN(4.10) on page 32 to show that the central difference approximation to (1) is

\[ u_{i,j+1} + u_{i,j-1} + u_{i+1,j} + u_{i-1,j} - 3u_{i,j} = x_i y_j. \quad (4) \]

By representing the unknowns at nodes 1, 2, 3, 4 as \( v_1 = u_{11}, \quad v_2 = u_{21}, \quad v_3 = u_{12}, \quad v_4 = u_{22} \), and substituting BC’s (2), (3), into (4), show that the central difference approximations at nodes numbers 1, 2, 3 are as shown below, and complete the entry for node 4.

(c) Suppose the matrix form of the linear system in (b) is \( Mv = d \). Fill in the elements of \( M \) and \( d \) in the places provided below.
Assign values to \( M \) and \( d \) in MATLAB, without using \texttt{diag} or \texttt{spdiags}, and use \( \backslash \) to show that \( v_1 = 0.4667, v_2 = 0.7556, v_3 = 0.7556, v_4 = 1.0222 \).

These values for \( v_1, \ldots, v_4 \) obtained in (c) are to be compared with the exact solution at nodes 1, \ldots, 4, namely \( u_{\text{exact}} = 0.449, 0.759, 0.759, 1.041 \), shown here correct to 3 decimal places. Clearly higher \( N \) must be used if more accuracy is required. Higher \( N \) leads to much larger systems, and the use of \texttt{spdiags} may be required. The advantage of using \texttt{spdiags} is apparent in the next question.

2. *(a) Copy the files $\texttt{bvp2/poiss.m}$ and $\texttt{poiss.f.m}$ to your directory. These contain the Poisson backslash solver in LN§4.4 and the example function M-file in LN§4.6 for the steady state temperature problem

\[
\nabla^2 u(x, y) = 0, \quad 0 < x, y < 1,
\]

\[
u(0, y) = u(1, y) = u(x, 0) = 0, \quad u(x, 1) = 1.
\]

Run \texttt{poiss.m} with 10 \times 10 interior grid points (i.e. \( N = 10 \)), and check that it produces the output shown in LN Figure 16 on page 44. Make sure you understand the use of \texttt{spdiags} in \texttt{poiss.m} to construct the banded matrix \( M \).

(b) One global measure of the accuracy of the approximate solution \( u \) for a given \( N \), is the root mean square error defined by

\[
E_{\text{rms}} := \text{Rms}(E) = \sqrt{\frac{1}{A} \int_0^a \int_0^a E^2 \, dx \, dy}, \quad \text{where} \quad E = u - u_{\text{exact}},
\]

and where \( A = a^2 \) is the area of the square region where Poisson’s equation applies. Somewhat more useful is the relative rms error defined by \( E_{\text{rrms}} = \text{Rms}(E)/\text{Rms}(u_{\text{exact}}) \), where \( \text{Rms}(u_{\text{exact}}) \) is defined similarly to \( \text{Rms}(E) \). Modify your copy of \texttt{poiss.m} to include computation and output of \( E_{\text{rrms}} \). This can be done using the MATLAB trapezoidal rule integration function \texttt{trapz}, e.g. insert the following lines (where you complete the ‘...’), just before the ‘plotting section’. (See BVP Tutorial 2 or use \texttt{help trapz} in MATLAB if need be.)

\[
\texttt{Erms=sqrt(trapz(x,trapz(y,...))/(a^2));}
\]

\[
\texttt{uexactrms=sqrt(...(...,...,...,uexact.^2))/(a^2));}
\]

\[
\texttt{Errms=.../...;}
\]

\[
\texttt{disp(’ uexactrms Errms’)}
\]

\[
\texttt{disp([uexactrms,Errms])}
\]

(c) Run \texttt{poiss.m} with \( N = 5 \) and \( N = 10 \), and record the values of \( E_{\text{rms}} \). Determine the minimum \( N \), \( N_{\text{min}} \) say, such that \( E_{\text{rms}} < 0.5\% \), and \( E_{\text{rrms}} < 0.2\% \). [Ans: \( E_{\text{rms}} \approx 0.009, 0.005 \), \( N_{\text{min}} \approx 11, 29 \). In this problem the discontinuous boundary conditions at (0,1) and (1,1) mean that it takes significantly greater \( N \) to reduce \( E_{\text{rms}} \) significantly. There always remains a non-zero error near the top corners.]

(d) For the case \( N = 29 \) in (c), and without using MATLAB, write down the dimensions of the matrix \( M \) in the linear system that is being solved. Use a calculator to find the total number of elements in \( M \), i.e. the number of interior grid points being used; and calculate the number of bytes that would be used if MATLAB was storing \( M \) in full.

(Note that MATLAB uses 8 bytes, i.e. double precision, to store a real number.)

Use the MATLAB command $\texttt{whos M}$ to find out how many elements of \( M \) are non-zero, and the actual number of bytes used by MATLAB to store these elements together with their addresses. Given that the density of a sparse matrix is defined as the fraction of non-zero elements, calculate the density of \( M \). (Do this with a calculator, or in MATLAB via the command
\[
nnz(M)/\text{prod(size}(M))\text{, which uses the inbuilt ‘number of non-zeros’ function } \text{nnz.}.
\]

[Ans: 707,281 elements, approx. 5.7Mbytes; 4089 non-zero elements which together with addresses occupy only about 52Kbytes, density approx. 0.6\%, i.e. very sparse.]

**Additional Practice Exercises**

3. For the problem in Q1, give MATLAB commands for constructing \(M\) using (i) \texttt{diag} as on LNP.23, and (ii) \texttt{spdiags} as on LNP.29. Check that use of these commands in MATLAB does reproduce the same \(M\) as assigned in Q1(c). (Of course \texttt{spdiags} would only be used in practice for larger \(N\).)

4. *(a) Use a single \texttt{diag} command to construct the \(20 \times 20\) diagonal matrix \(A\), with elements

\[
A_{ij} = \begin{cases} 
0 & (i \neq j), \\
1 & (i = j).
\end{cases}
\]

(b) Use a single \texttt{spdiags} command to construct the same matrix \(A\) as in (a).

5. Consider the following BVP:

\[
\nabla^2 u(x, y) = -2, \quad 0 < x, y < 1, \\
u(0, y) = 0, \quad u(1, y) = \sinh(\pi)\sin(\pi y), \quad 0 \leq y \leq 1, \\
u(x, 0) = u(x, 1) = x(1 - x), \quad 0 \leq x \leq 1 \\
u_{\text{exact}} = \sinh(\pi x)\sin(\pi y) + x(1 - x). 
\]

Use \texttt{poiss.m} and \texttt{poiss.f.m} from Q2 to again find \(E_{\text{rms}}\) for \(N = 5, 10\), but now find \(N_{\text{min}}\) such that \(E_{\text{rms}} < 0.1\%.\) To reuse \texttt{poiss.f.m} for this and other problems, use a block \texttt{if} construct along the lines

\[
\text{if } \text{fno} == 1 \\
\quad \text{(define function number 1)} \\
\text{elseif } \text{fno} == 2 \\
\quad \ldots \\
\text{end } %\text{End if construct}
\]

Then read the function number \texttt{fno} into \texttt{poiss.m} via an \texttt{input} statement, and place \texttt{fno} in \texttt{global} statements in both \texttt{poiss.m} and \texttt{poiss.f.m}.

[Ans: \(E_{\text{rms}} \approx 0.01, 0.004, N_{\text{min}} \approx 21\). You should observe better accuracy compared to Q2 for the same \(N\) values as \(N\) increases, since the BC’s here are not discontinuous.]

6. * Modify \texttt{poiss.m} so that it applies to the inhomogeneous Helmholtz equation

\[
\nabla^2 u + ku = f,
\]

which arises in certain forced vibration problems, and temperature problems with heat sources. Let \(k\) be entered via an \texttt{input} statement. Check your program by running it with \(k = 1\) and the same BC’s and \(u_{\text{exact}}\) as in Q5. [Hint: you need to determine the appropriate \(f\). You can reuse \texttt{poiss.f.m} from Q5 by adding in another \texttt{elseif} sub-block. Your program should show agreement between the exact and approximate solutions.]