Problems

1. The Jacobi iterative method 1.9.4 can be written in the form $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n + D^{-1}(\boldsymbol{b} - A\boldsymbol{x}_n)$. Write an MPI-based subroutine JACOBI with arguments (A,IROW,JCOL,NZ,X,B,N), which iterates the Jacobi method to convergence, to solve an N by N (possibly nonsymmetric) linear system $A\boldsymbol{x} = \boldsymbol{b}$, with sparse matrix A distributed over the available processors. Solve the system 1.9.10, with $M = 40$ using a main program similar to Figure 6.2.5 to test your program. Run with 1 and 2 processors.

2. a. Use PLINEQ (Figure 6.2.1) to solve the linear system 1.9.10, and output the solution at the midpoint again, to check your answer. You can use the main program from Figure 6.2.5 to define the matrix AS in sparse format; then copy AS to an N by N full matrix A (inefficient for a sparse system such as this, of course), with columns also distributed over the available processors. You will need to remove the CALL MPI_INIT in PLINEQ since you are now initializing MPI in the main program.

   Run with 1, 2, 4, and 8 processors, with $M = 16$.

   b. In part (a), although no processor ever touches any columns but its own, you are still storing the entire N by N matrix on every processor. This can be avoided, as suggested in the text, by replacing every reference to $A(I, J)$ by $A(I, (J-1)/NPES + 1)$, in both the main program and PLINEQ. Then A can be dimensioned $A(N, (N-1)/NPES+1)$ and each processor will only store its own columns. However, NPES is not known until execution time, so you cannot dimension A in a DIMENSION statement; you should use the FORTRAN90 ALLOCATE statement to dynamically allocate space for A. This means

```
ALLOCATABLE A(:,:)
 .
CALL MPI_COMM_SIZE (MPI_COMM_WORLD,NPES,IERR)
ALLOCATE (A(N,(N-1)/NPES+1))
 .
```

   Make these modifications and retest your program. You should be able to solve problems with larger M now, using many processors.