CPS 5320

Name _____

1. Derive the beam bending equation as follows. Suppose $u(x)$ is the height of the beam with $u(0) = g_0, u'(0) = h_0, u(L) = g_1, u'(L) = h_1$, which minimizes the energy

$$E(u) \equiv \int_0^L \frac{1}{2} D[u'']^2 - uq \; dx,$$

where D (constant) is the bending stiffness and $q(x)$ is an external vertical force (per unit length). Then if $e(x)$ is any smooth function with $e(0) = e'(0) = e(L) = e'(L) = 0$ on the boundary, $E(u + \alpha e) \geq E(u)$ for any α, thus $f(\alpha) \equiv E(u+\alpha e) = \int_0^L \frac{1}{2} D[u''+\alpha e'']^2 - [u+\alpha e]q \; dx$ should have a minimum at $\alpha = 0$ and so $\frac{df}{d\alpha}(0)$ should be zero. Do two integrations by parts to get the integral in this equation into a form where it is clear what differential equation must be satisfied by $u(x)$.

2. In the program PLINEQ that we studied in class, which uses Gaussian elimination to solve a linear system, the columns of the matrix A are distributed cyclically over the processors: 0,1,2,...,NPES-1,0,1,2,...,NPES-1,.... Suppose we modify PLINEQ so that the columns are distributed by "blocks," with the first NB=N/NPES columns on processor 0, the next NB columns on processor 1, etc: 0,0,0,...,0,1,1,1,...,1,2,... (Assume that N is an integer multiple of NPES.) Would you expect the new program to run faster than the original PLINEQ? How much faster or slower (assume NPES is large, but N is much larger)? Show calculations to justify your answer.

3. If U is a function of $\rho = \sqrt{x^2 + y^2 + z^2}$ only, use the chain rule to express $U_{xx} + U_{yy} + U_{zz}$ in terms of $U_{\rho\rho}, U_\rho$ only.

4. A technique sometimes used in image processing is to solve the nonlinear diffusion equation $U_t = \nabla \bullet [\nabla U/(1 + (U_x^2 + U_y^2)/\lambda^2)]$, with a noisy image as the initial condition and zero flux boundary conditions. Why can this diminish random noise without destroying the real picture? What will happen if the diffusion equation is solved too far in time?

5. If the stream function approach is used for a 2D incompressible fluid flow problem, what are the appropriate boundary conditions for "no slip" (U=V=0) in terms of the stream function and vorticity, ϕ, ω ($U = \phi_y, V = -\phi_x, \omega = U_y - V_x$)? What are the appropiate boundary conditions for "free slip" ($V = U_y = 0$ on y=constant boundaries, $U = V_x = 0$ on x=constant boundaries)?

6. If 95% of a program is parallelizable, about what speed-up factor should be expected when going from 1 to 8 processors? (Assume the communication time is negligible).

7. If the MPI Fortran program below is run on NPES=3 processors, what will be output for B, on every processor?

```
      PARAMETER (N=4)
      DOUBLE PRECISION A(N),B(N)
      INCLUDE 'mpif.h'
C                     INITIALIZE MPI
      CALL MPI_INIT (IERR)
C                     NPES = NUMBER OF PROCESSORS
      CALL MPI_COMM_SIZE (MPI_COMM_WORLD,NPES,IERR)
C                     ITASK = MY PROCESSOR NUMBER
      CALL MPI_COMM_RANK (MPI_COMM_WORLD,ITASK,IERR)
      DO I=1,N
         A(I) = 10*ITASK + I
         B(I) = A(I)
      ENDDO
C
      iroot = 0
      CALL MPI_BCAST(A,N,MPI_DOUBLE_PRECISION,iroot,
     &                MPI_COMM_WORLD,IERR)
      CALL MPI_REDUCE(B,A,N,MPI_DOUBLE_PRECISION,
     &                MPI_SUM,iroot,MPI_COMM_WORLD,IERR)
      CALL MPI_ALLREDUCE(A,B,N,MPI_DOUBLE_PRECISION,
     &                MPI_SUM,MPI_COMM_WORLD,IERR)
      PRINT *, B
      CALL MPI_FINALIZE(IERR)
      STOP
      END
```