CPS 5320 Practice Test

Name Key

1. If U is a function of $r = \sqrt{x^2 + y^2}$ only, use the chain rule to express $U_{xx} + U_{yy}$ in terms of U_{rr}, U_r only.

   answer: $U_{rr} + U_r/r$

2. To derive the beam bending equation, 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 $D(x)$ is the bending stiffness and $q(x)$ is an external vertical force. 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. From this we conclude that $f(\alpha) \equiv E(u + \alpha e)$ has a minimum at $\alpha = 0$, and thus $f'(0) = 0$. Write out the equation $f'(0) = 0$ and explain how this equation can be used to find a differential equation for u. (You don't need to actually derive the differential equation, just outline what needs to be done.)

   $\frac{df}{d\alpha}(0) = \int_0^L Du''e'' - eq \, dx = 0$. Do two integrations by parts to get $\int_0^L e[...]dx = 0$, since e is arbitrary, the expression in brackets must be 0.

3. Suppose we use Gauss elimination to solve an N by N linear system, and distribute the columns by "blocks" over the processors: each processor stores the entire matrix, but the first NB=N/NPES columns are only touched by processor 0, the second NB columns by processor 1, etc. How would you modify the innermost loop of DLINEQ, shown below:

```
      DO 25 K=I,N
         A(J,K) = A(J,K) - LJI*A(I,K)
25    CONTINUE
```

Use ITASK for the processor number, and NPES for the number of processors, and assume N is divisible by NPES. Hint: the limits are simpler now than when the columns are distributed cyclically, and remember that if $I1 > I2$, no trips through a loop "DO K=I1,I2" will be made.

answer:

```
      DO 25 K=max(I,ITASK*NB+1),ITASK*NB+NB
         A(J,K) = A(J,K) - LJI*A(I,K)
25    CONTINUE
```

who owns column I : "(I-1)/NB"

4. What is the output from the MPI Fortran program below, if run on NPES=3 processors?

```
      PARAMETER (N=12)
      DOUBLE PRECISION X(N),SUMI,SUM
      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
         IF (MOD(I-1,NPES).EQ.ITASK) THEN
            X(I) = I
         ENDIF
      ENDDO
C
      SUMI = 0
      DO I=ITASK+1,N,NPES
         SUMI = SUMI + X(I)
      ENDDO
      iroot = 1
      CALL MPI_REDUCE(SUMI,SUM,1,MPI_DOUBLE_PRECISION,
     &                MPI_SUM,iroot,MPI_COMM_WORLD,IERR)
      if (ITASK.EQ.iroot) PRINT *, SUMI,SUM
      CALL MPI_FINALIZE(IERR)
      STOP
      END
```

answer: 26,78

5. In deriving the Black-Scholes partial differential equation, we assumed what type of probability distribution for the price S at future time=T of an asset whose price is s at time=t? What did we assume for the mean and the standard deviation of this distribution? The asset volatility is σ_1, the strike price is E and the risk-free interest rate (rate of inflation, sort of) is r.

   answer: log normal distribution with mean $se^{r(T-t)}$ and standard deviation $\sigma = \sigma_1\sqrt{T-t}$.

6. Consider the diffusion partial differential equation:

$$C_t = \nabla \bullet [D\nabla \mathbf{C} - C\mathbf{v}] + q$$

a. What do $D, \mathbf{v}$, and q represent physically?
answer: D = diffusion coefficient, $\mathbf{v}$ = convection velocity field, q = generation rate due to sources/sinks.

b. Do solutions tend to be smoother when $D = 0$?
answer: no!

c. If $D(x, y)$ is a discontinuous function, for example, if it jumps from one value in one subregion to another in another subregion, which finite element method, Galerkin or collocation, is better able to handle this case, and why? Will the density C be continuous then? How about ∇C?
answer: Galerkin is better, for collocation we would have to write as $C_t = D\nabla^2 C + \nabla D \bullet \nabla C$... and ∇D is infinite at interface. C is continuous, but ∇C is not.

CPS 5320 (a)

Name Key

1. 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.

(3)

$U_x = U_\rho \rho_x = U_\rho \frac{x}{\rho}$ $\quad U_{xx} = \frac{\rho(U_\rho + x U_{\rho\rho}\frac{x}{\rho}) - x U_\rho \frac{x}{\rho}}{\rho^2}$

$U_{xx} + U_{yy} + U_{zz} = \frac{3U_\rho}{\rho} + \frac{x^2+y^2+z^2}{\rho^2} U_{\rho\rho} - \frac{x^2+y^2+z^2}{\rho^3} U_\rho = \frac{U_\rho}{\rho} + \frac{x^2}{\rho^2} U_{\rho\rho} - \frac{x^2}{\rho^3} U_\rho$

$= U_{\rho\rho} + \frac{2U_\rho}{\rho}$

2. The 2D steady-state Navier Stokes equations, at low Reynold's number are:

(3)

$$f1 + \mu(U_{xx} + U_{yy}) = P_x$$

$$f2 + \mu(V_{xx} + V_{yy}) = P_y$$

$$U_x + V_y = 0$$

where (U, V) is the fluid velocity vector, and μ, P are the fluid viscosity and pressure, and $(f1, f2)$ is the external force field.

If we define a stream function $\phi(x, y)$ such that $(U, V) = (\phi_y, -\phi_x)$, show that the last (divergence) equation is automatically satisfied, and find a system of two second order equations involving ϕ and the "vorticity" $\omega \equiv U_y - V_x$.

$U_x + V_y = \phi_{yx} - \phi_{xy} = 0$

$f1_y + \mu(U_{xxy} + U_{yyy}) = P_{xy}$

$f2_x + \mu(V_{xxx} + V_{yyx}) = P_{yx}$

$f1_y - f2_x + \mu\left((U_y - V_x)_{xx} + (U_y - V_x)_{yy}\right) = 0$

$f1_y - f2_x = \mu(\omega_{xx} + \omega_{yy})$

$\omega = \phi_{yy} + \phi_{xx}$

3. To derive the minimal surface equation, suppose $u(x,y)$ is the surface with $u = g(x,y)$ on the boundary $\partial\Omega$ of Ω which minimizes the surface area $SA(u) \equiv \int\int_\Omega \sqrt{1+u_x^2+u_y^2}\, dA$. Then if $e(x,y)$ is any smooth function with $e = 0$ on the boundary, $SA(u+\alpha e) \geq SA(u)$ for any α. From this we conclude that $f(\alpha) \equiv SA(u+\alpha e)$ has a minimum at $\alpha = 0$, and thus $f'(0) = 0$. Write out the equation $f'(0) = 0$ and explain how this equation can be used to find a partial differential equation for u. (You don't need to actually derive the partial differential equation, just outline what needs to be done.)

(3)

$$f(\alpha) = \iint \left(1 + (u_x+\alpha e_x)^2 + (u_y+\alpha e_y)^2\right)^{\frac{1}{2}} dA$$

$$f'(\alpha) = \iint \tfrac{1}{2}\left(1+(u_x+\alpha e_x)^2+(u_y+\alpha e_y)^2\right)^{-\frac{1}{2}} \left(2(u_x+\alpha e_x)e_x + 2(u_y+\alpha e_y)e_y\right) dA$$

$$f'(0) = \iint \frac{u_x e_x + u_y e_y}{\sqrt{1+u_x^2+u_y^2}}\, dA$$

Do integration by parts to get $\iint e[\ \text{PDE}\ ]\, dA$

$\Rightarrow$ PDE $= 0$

4. The diffusion/convection/reaction equation is:

$$C_t = \nabla \bullet [D\nabla \mathbf{C} - C\mathbf{v}] + q$$

where $C(x,y,z,t)$ is the density of a substance, and $\mathbf{v} \equiv (U,V,W)$ is the velocity of the medium. If there is no diffusion ($D = 0$) and no reaction (source or sink) terms ($q = 0$), so only convection is operative, we get the "continuity" equation:

$$C_t + \nabla \bullet [C\mathbf{v}] = 0$$

If we follow a given point $(X(t), Y(t), Z(t))$ as it moves with the fluid

(so $(X', Y', Z') = (U, V, W)$), and define the density at this moving point as $C_0(t) \equiv C(X(t), Y(t), Z(t), t)$, show, using the chain rule and the continuity equation, that

$$C_0'(t) = -(U_x + V_y + W_z)C_0(t).$$

3

This means that an incompressible fluid ($C_0'(t) = 0$) satisfies the divergence equation $U_x + V_y + W_z = 0$.

$$C_0'(t) = C_x u + C_y v + C_z w + C_t$$

$$= C_t + (Cu)_x + (Cv)_y + (Cw)_z - C_0 u_x - C_0 v_y - C_0 w_z$$

$$C_0' = C_t + \nabla \cdot (C\vec{v}) - C_0(u_x + v_y + w_z)$$

$$C_0'(t) = -C_0(u_x + v_y + w_z)$$

5. Chris Sewell, in his lecture on *Visualization and Data Analysis in HPC* talked about visualization with VTK, ParaView and OpenGL. Rank these three from lowest to highest level. (Lowest level means most flexible and requiring most attention to detail.)

1

OpenGL, VTK, ParaView

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

```
      PARAMETER (N=5)
      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_MAX,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
```

A
0: 1 2 3 4 5
1: 11 12 13 14 15
2: 21 22 23 24 25

21 22 23 24 25
1 2 3 4 5
1 2 3 4 5

B
1 2 3 4 5
11 12 13 14 15
21 22 23 24 25

23 26 29 32 35

7. Explain why it is more efficient to distribute the columns of a matrix cyclically (0,1,2,...,NPES-1,0,1,2,...NPES-1,...) than in blocks (0,0,...0,1,1,...1,2,2,...2,...) when doing Gaussian elimination.

(2) After first $NB = \frac{N}{NPES}$ column zeroed, first processor is idle rest of forward elimination. When last NB column zeroed all processors except last are idle.

8. You are offered an option to buy an asset at price E at time T. If $V(t, s)$ is the value of the option at time t if the asset price is s at that time, V satisfies the Black-Scholes partial differential equation. What are the appropriate initial/boundary conditions for V?

(2)
$$\begin{cases} V(T,S) = \max(S-E, 0) \\ V(t,0) = 0 \\ V_S(t, S_{max}) = 1 \end{cases}$$

CPS 5320 (6)

Name Key

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 $u(x)$.

3

$$f(\alpha) = \int_0^L \tfrac{1}{2} D (u'' + \alpha e'')^2 - (u+\alpha e) q \, dx$$

$$f'(\alpha) = \int_0^L D(u'' + \alpha e'') e'' - e q \, dx$$

$$f'(0) = \int_0^L D u'' e'' - eq \, dx = 0$$

$$\int_0^L D(u'' e')' - D(u''' e)' + D u'''' e - eq \, dx = 0$$

$$= D u'' e' \Big]_0^L - D u''' e \Big]_0^L + \int_0^L (D u'''' - q) e \, dx$$

$$\Rightarrow \boxed{D u'''' - q = 0}$$

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.

Original PLINEQ: $\frac{1}{3}N^3/NPES$ since work distributed evenly

3

new PLINEQ: $\sum_{i=1}^{N-1}\sum_{j=i+1}^{N} NB \cong \frac{1}{2}N^2 NB = \frac{1}{2}\frac{N^3}{NPES}$ (last processor)

new block version 50% more work

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.

$$U_x = U_\rho \rho_x = U_\rho \frac{x}{\rho}$$

3

$$U_{xx} = (U_{\rho\rho}\rho_x)\rho_x + U_\rho \frac{\rho - x\rho_x}{\rho^2} = U_{\rho\rho}\frac{x^2}{\rho^2} + U_\rho\left(\frac{1}{\rho} - \frac{x^2}{\rho^3}\right)$$

$$U_{xx}+U_{yy}+U_{zz} = U_{\rho\rho}\left(\frac{x^2+y^2+z^2}{\rho^2}\right) + U_\rho\left(\frac{3}{\rho} - \frac{x^2+y^2+z^2}{\rho^3}\right)$$

$$= U_{\rho\rho} + \frac{2}{\rho}U_\rho$$

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?

3

D is small near steep gradients, assumed to be real
but larger near small gradients, assumed to be noise
so smooths out noise more than sharp, real, features
After a long time, U eventually becomes constant destroying picture!

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)?

3

no slip:

$\phi_x = \phi_y = 0$

so

ϕ = constant

$\frac{\partial \phi}{\partial n} = 0$

free slip:

$V_x = 0$

y = constant: $V = 0 \Rightarrow \phi_x = 0$

$\Rightarrow \phi$ = constant

$\omega = U_y - V_x = 0 - 0 = 0$

$\omega = 0$

similarly on x = constant

(also general boundary)

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).

3

$$\frac{1}{\frac{f_p}{NPES} + 1 - f_p} = \frac{1}{\frac{.95}{8} + .05} = 5.9$$

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
```

3

35
40
45
50

CPS 5320(c)

Name key

1. If the density of a fluid is $\rho(t,x,y,z)$ then if we equate the rate of change of fluid mass in an arbitrary subregion S to the rate at which it is entering/leaving, we get

   $\frac{d}{dt}\iiint_S \rho = \iint_{\partial S} \rho \mathbf{v} \bullet (-\mathbf{n})$

   where $\mathbf{v} = (\mathbf{U}, \mathbf{V}, \mathbf{W})$ is the fluid velocity and $\mathbf{n}$ is the unit outward normal, and ∂S is the boundary of S.

   a. Use the divergence theorem to derive a partial differential equation for the fluid density, and show that if the fluid is incompressible (ie, the density $\rho(t, x(t), y(t), z(t))$ of a moving piece of fluid is constant) then the fluid velocity satisfies $U_x + V_y + W_z = 0$.

10

$$\iiint \rho_t = \iiint -\nabla\cdot(\rho\vec{v}) \qquad \rho_t = -\nabla\cdot(\rho\vec{v})$$

$$\rho_t + (\rho u)_x + (\rho v)_y + (\rho w)_z = 0$$

$$\underbrace{\rho_t + \rho_x u + \rho_y v + \rho_z w}_{0} + \rho(u_x + v_y + w_z) = 0$$

$$\Rightarrow u_x + v_y + w_z = 0$$

   b. If the fluid is not quite incompressible, but it takes a large pressure (P) to make a small change in volume, what can the divergence equation be replaced by? This equation is the one used by the penalty method.

10

$$p = -\alpha(u_x + v_y + w_z)$$

α = large number

2. Explain why it is more efficient to distribute the columns of a matrix cyclically (0,1,2,...,NPES-1,0,1,2,...NPES-1,...) than in blocks (0,0,...0,1,1,...1,2,2,...2,...) when doing Gaussian elimination, while it doesn't matter as much when programming the Simplex method.

10

For GE, toward the end only the last columns, and then the last processors if block distribution is done, are active. For the Simplex method, all columns are active throughout.

3. You are offered an option ("put" option) to sell an asset at price E at time T. If $V(t,s)$ is the value of the option at time t if the asset price is s at that time, V satisfies the Black-Scholes partial differential equation. What are the appropriate initial/boundary conditions for V?

10

$V(T,s) = \max(E-s,0)$

$V_s(t,0) = -1$ or $V(t,0) = Ee^{-r(T-t)}$

$V(t,\infty) = 0$

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

```
      PARAMETER (N=5)
      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 = 1
      CALL MPI_BCAST(A,N,MPI_DOUBLE_PRECISION,iroot,
     &                MPI_COMM_WORLD,IERR)
      CALL MPI_REDUCE(B,A,N,MPI_DOUBLE_PRECISION,
     &                MPI_MAX,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
```

10

43 46 49 52 55

5. a. One of your homework problems was to solve the beam problem $\nabla^2(\nabla^2 u) = -\delta(x,y)$ in the unit disk. You were to calculate the boundary integral of $\frac{\partial M}{\partial n}$, where $M \equiv \nabla^2 u$ to check your answer. What should this integral be, no matter what boundary conditions were used? Show your work.

10

$$\iint \nabla^2 M = \iint -\delta(x,y) = -1$$

$$\int \nabla M \cdot n = -1$$

$$\int \frac{\partial M}{\partial n} = -1$$

b. What are the simply supported boundary conditions for this problem?

$u=0$, $M=0$

4

c. What are the clamped boundary conditions?

$u=0$, $\frac{\partial u}{\partial n}=0$

4

d. What are the free and unsupported boundary conditions?

$M=0$, $\frac{\partial M}{\partial n}=0$

4

6. If 99% of a program is parallelizable, about what speed-up factor should be expected when going from 1 to 16 processors? (Assume the communication time is negligible). What is the maximum speed-up possible, with many processors?

10

$$\frac{1}{\frac{0.99}{16} + 0.01} = 13.91 \text{ speedup with 16 procs}$$

$$\frac{1}{\frac{0.99}{\infty} + 0.01} = 100 \text{ max speedup}$$

7. The Schroedinger equation is $\nabla^2\phi - V(x,y,z)\phi = -E\phi$, with $\phi = 0$ at $\rho \equiv \sqrt{x^2+y^2+z^2} = \infty$, where $V(x,y,z)$ is the potential energy of the field.

   a. In a hydrogen atom, $V(x,y,z) = -1/\rho$. Reduce the Schroedinger equation to a 1D equation, and tell which eigenvalues (if not all of them) we can find using this 1D version.

$$\frac{\partial^2\varphi}{\partial\rho^2} + \frac{2}{\rho}\frac{\partial\varphi}{\partial\rho} + \frac{1}{\rho}\varphi = -E\varphi$$

10

only eigenvalues with spherically symmetric eigenfunctions

   b. In an H_2^+ molecule,

$$V(x,y,z) = -1/\sqrt{x^2+y^2+(z+1)^2} - 1/\sqrt{x^2+y^2+(z-1)^2}.$$

   Reduce the Schroedinger equation to an axisymmetric equation and tell which eigenvalues (if not all of them) we can find using this 2D version.

10 $\frac{\partial^2\varphi}{\partial z^2} + \frac{\partial^2\varphi}{\partial r^2} + \frac{1}{r}\frac{\partial\varphi}{\partial r} + \left(\frac{1}{\sqrt{r^2+(z+1)^2}} + \frac{1}{\sqrt{r^2+(z-1)^2}}\right)\varphi = -E\varphi$

only eigenvalues with axisymmetric eigenfunctions

2017

CPS 5320 (March 24) Qualifier (Theory)

1. To derive the minimal surface equation, suppose $u(x,y)$ is the surface with $u = g(x,y)$ on the boundary $\partial\Omega$ of Ω which minimizes the surface area $SA(u) \equiv \int\int_\Omega \sqrt{1+u_x^2+u_y^2}\, dA$. Then if $e(x,y)$ is any smooth function with $e = 0$ on the boundary, $SA(u+\alpha e) \geq SA(u)$ for any α. From this we conclude that $f(\alpha) \equiv SA(u+\alpha e)$ has a minimum at $\alpha = 0$, and thus $f'(0) = 0$. Write out the equation $f'(0) = 0$ and then derive a partial partial differential equation for u.

   Hint: you can use the multidimensional integration by parts formula:

$$\iint_\Omega \nabla u \bullet \mathbf{v} = \int_{\partial\Omega} u\mathbf{v} \bullet n - \iint_\Omega u\nabla \bullet \mathbf{v}$$

   where u is a scalar function, $\mathbf{v}$ is a vector function.

$$f(\alpha) = SA(u+\alpha e) = \iint \left[1 + (u_x+\alpha e_x)^2 + (u_y+\alpha e_y)^2\right]^{\frac{1}{2}} dA$$

$$f'(\alpha) = \iint \left[1 + (u_x+\alpha e_x)^2 + (u_y+\alpha e_y)^2\right]^{-\frac{1}{2}} \left[(u_x+\alpha e_x)e_x + (u_y+\alpha e_y)e_y\right] dA$$

$$f'(0) = \iint \frac{u_x e_x + u_y e_y}{\sqrt{1+u_x^2+u_y^2}}\, dA = 0$$

$$\iint \nabla\cdot\left(e\frac{\nabla u}{\sqrt{\ }}\right) - e\nabla\cdot\left(\frac{\nabla u}{\sqrt{\ }}\right) dA = \int_{\partial R} e\frac{\frac{\partial u}{\partial n}}{\sqrt{\ }}\, ds - \iint e\nabla\cdot\left(\frac{\nabla u}{\sqrt{\ }}\right)$$

$$\iint e\,\nabla\cdot\left(\frac{\nabla u}{\sqrt{1+u_x^2+u_y^2}}\right) dA = 0 \quad \text{all } e \quad \Rightarrow \quad \nabla\cdot\left(\frac{\nabla u}{\sqrt{1+u_x^2+u_y^2}}\right) = 0$$

2. Solve this minimal surface equation using PDE2D, in the region $1 < r < 2, 0 < \theta < \pi/2$, with $U = (r-1)(2-r)$ on the boundary, where r, θ are polar coordinates. Output the integral of U over the region (should be about 0.101).

3. a. Modify the ?????s in the following code so that each processor initializes the elements of N-vectors X and Y only on its processors, and calculates the dot product X and Y in parallel, when the elements of X and Y are distributed cyclically over the NPES processors, that is, in the order
0,1,2,...,NPES-1,0,1,2...,NPES-1,0,1,2,...,NPES-1,...

```
      PARAMETER (N=12000000)
      DOUBLE PRECISION X(N),Y(N),SUM,SUMI
      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)
      NB = N/NPES
      DO I=1,N
         IF (?????.EQ.ITASK) THEN
            X(I) = I
            Y(I) = I+2
         ENDIF
      ENDDO
C
      SUMI = 0.0
      DO I=?????
         SUMI = SUMI + X(I)*Y(I)
      ENDDO
      CALL MPI_ALLREDUCE(SUMI,SUM,1,MPI_DOUBLE_PRECISION,
     &                   MPI_SUM,MPI_COMM_WORLD,IERR)
      PRINT *, SUM
      CALL MPI_FINALIZE(IERR)
      STOP
      END
```

b. Same question, but now assume the elements of X and Y are distributed over the processors by blocks, that is, in the order (you may assume N is divisible by NPES):
0,0,...0,1,1,...,1,2,2,...2,...

```
      PARAMETER (N=12000000)
      DOUBLE PRECISION X(N),Y(N),SUM,SUMI
      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)
      NB = N/NPES
      DO I=1,N
         IF (?????.EQ.ITASK) THEN
            X(I) = I
            Y(I) = I+2
         ENDIF
      ENDDO
C
      SUMI = 0.0
      DO I=?????
         SUMI = SUMI + X(I)*Y(I)
      ENDDO
      CALL MPI_ALLREDUCE(SUMI,SUM,1,MPI_DOUBLE_PRECISION,
     &                   MPI_SUM,MPI_COMM_WORLD,IERR)
      PRINT *, SUM
      CALL MPI_FINALIZE(IERR)
      STOP
      END
```

(I-1)/NB

ITASK*NB+1, ITASK*NB+NB

c. Would you expect either code (a) or (b) to be significantly faster than the other? Explain. about the same

4. What will be the output from the program below, if it is run on NPES=3 processors?

```
      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 = 2
      CALL MPI_BCAST(A,N,MPI_DOUBLE_PRECISION,iroot,
     &                MPI_COMM_WORLD,IERR)
      CALL MPI_REDUCE(A,B,N,MPI_DOUBLE_PRECISION,
     &                MPI_SUM,iroot,MPI_COMM_WORLD,IERR)
      PRINT *, ITASK,B
      CALL MPI_FINALIZE(IERR)
      STOP
      END
```

after MPI_REDUCE

B

0	1	2	3	4
1	11	12	13	14
2	63	66	69	72

Initial:

A

0	1	2	3	4
1	11	12	13	14
2	21	22	23	24

B

1	2	3	4
11	12	13	14
4	22	23	24

after MPI_BCAST

A

0	21	22	23	24
1	21	22	23	24
2	21	22	23	24

B

1	2	3	4
11	12	13	14
21	22	23	24

CPS 5320 Theory-Based Qualifier Exam

09:00 am – 10:30 am on August 22, 2019

Name: Key

Student ID #: _____

Please read the following instructions carefully

1. This is a closed book exam.
2. The total time for this exam is 1 hour 30 minutes.
3. The exam is worth a total of 100 points.
4. You are permitted to use a simple (non-graphing, non-programmable) calculator. Cell phones, laptops, and all other web-enabled devices are not allowed.
5. Show sufficient work for full credit.

Number	Maximum Points	Earned Points
I	20	
II	20	
III	20	
IV	20	
V	20	
Total	100	

I. What application did we see which involved the "divergence" equation $U_x + V_y + W_z = 0$, and what physical property does this equation enforce? For an "axisymmetric" problem, $U = R(r,z)cos(\theta), V = R(r,z)sin(\theta), W = W(r,z)$, where $r = \sqrt{x^2+y^2}, \theta = tan^{-1}(y/x)$ are polar (or cylindrical) coordinates. Use the chain rule to convert the divergence equation to axisymmetric form, where only derivatives of R, W with respect to r, z appear.

$r_x = \frac{x}{r} = cos\theta$

$r_y = \frac{y}{r} = sin\theta$

$\theta_x = \frac{-y}{r^2} = \frac{-sin\theta}{r}$

$\theta_y = \frac{x}{r^2} = \frac{cos\theta}{r}$

incompressible fluid flow; enforces incompressibility

$U_x = R_r r_x cos\theta + R(-sin\theta)\theta_x$

$= R_r cos^2\theta + R\frac{sin^2\theta}{r}$

$V_y = R_r r_y sin\theta + R cos\theta \theta_y$

$= R_r sin^2\theta + R\frac{cos^2\theta}{r}$

$U_x + V_y + W_z = R_r + \frac{R}{r} + W_z$

II. Consider the elastic plate problem, $\nabla^2(\nabla^2 U) = q(x,y)$, with boundary conditions $M = 0, \frac{\partial M}{\partial n} = 0$, where $M = \nabla^2 U$ is the bending moment. Integrate both sides of the PDE and use the divergence theorem to find a condition on the load $q(x,y)$ which must be satisfied for this steady-state problem to have a solution. Why does this condition make sense physically? (Hint: what do the boundary conditions model?) Show that even if this condition is satisfied, the solution is not unique.

$\iint \nabla^2 M = \iint q$ $\iint q$ must be 0

$0 = \int \frac{\partial M}{\partial n} = \iint q$

1) must be no net force up or down for equilibrium

2) add $+C$ to U still a solution since M unchanged

III. (a) A "call" option gives you the option to buy an asset at price E, at time T. What are the initial/final conditions and what are the boundary conditions, for the Black-Scholes equation for the value $V(s,t)$ of this option, where s is the price of the asset at time t.

$V(s,T) = \max(s-E, 0)$

$V(0,t) = 0$

$V_s(\infty,t) = 1$

(b) Same question, but for a "put" option, which gives you the option to sell the asset at price E, at time T.

$V(s,T) = \max(0, E-s)$

$V_s(0,t) = -1$ (or $V(0,t) = E\,e^{-r(T-t)}$)

$V(\infty,t) = 0$

IV. The damped membrane equation we studied was:

$$\rho u_{tt} + b u_t = T\nabla^2 u + f(x,y)$$

where ρ, b, T and $f(x,y)$ are the density (per unit area),frictional coefficient, tension and load. The total energy (kinetic plus potential) of the membrane is

$$E(t) = \iint_\Omega \left[\frac{1}{2}\rho u_t^2 + \frac{T}{2}\nabla u \cdot \nabla u - fu\right] dA$$

Show that if $u = g(x,y)$ on the boundary, the total energy is nonincreasing, and constant only if $b = 0$ or u has reached a steady-state. (Note that $f(x,y)$ and $g(x,y)$ are assumed to not be functions of time.)

$$E'(t) = \iint_\Omega \left(\rho u_t u_{tt} + T\nabla u \cdot \nabla u_t - f u_t\right) dA$$

$$= \iint_\Omega \left[\rho u_t u_{tt} - T\nabla^2 u\, u_t + T\nabla\cdot(u_t \nabla u) - f u_t\right) dA$$

$$= \iint_\Omega u_t\left(\rho u_{tt} - T\nabla^2 u - f\right) dA + \int_{\partial\Omega} T u_t \frac{\partial u}{\partial n}$$ (the last term $= 0$)

$$= \iint_\Omega u_t(-b u_t)\, dA = -\iint_\Omega b u_t^2\, dA \le 0$$

$(=0$ only if $b=0$ or $u_t \equiv 0)$

V. The program PBACK below solves a nonsingular upper triangular system using back substitution, with the columns of U distributed cyclically over the processors, ie., 0,1,2,...,NPES-1,0,1,2,...,NPES-1,0,1,2,... Although each processor actually stores the entire matrix, it operates only on "its" columns. Modify PBACK (just mark your changes on the paper copy) 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,... You may assume that N will always be an integer multiple of NPES. This can be done by modifying only 2 lines. Hints: There is now a simpler way to specify the limits in loop 10.

```
      SUBROUTINE PBACK(U,X,B,N)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C                                SOLVE U*X=B USING BACK SUBSTITUTION.
      DOUBLE PRECISION U(N,N),X(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 (0,1,...,NPES-1).
C                                I WILL NEVER TOUCH ANY COLUMNS OF A EXCEPT
C                                MY COLUMNS
      CALL MPI_COMM_RANK (MPI_COMM_WORLD,ITASK,IERR)
      NB = N/NPES
      DO 20 I=N,1,-1
C                                IO IS FIRST COLUMN >= I+1 THAT BELONGS
C                                TO ME
        LO = (I+NPES-(ITASK+1))/NPES
        IO = ITASK+1+LO*NPES
        SUMI = 0.0
        DO 10 J=IO,N,NPES
           SUMI = SUMI + U(I,J)*X(J)
10      CONTINUE
        CALL MPI_ALLREDUCE(SUMI,SUM,1,MPI_DOUBLE_PRECISION,
     &   MPI_SUM,MPI_COMM_WORLD,IERR)
C                                JTASK IS PROCESSOR THAT OWNS U(I,I)
        JTASK = MOD(I-1,NPES)
C                                IF JTASK IS ME, CALCULATE X(I)
        IF (ITASK.EQ.JTASK) THEN
           X(I) = (B(I)-SUM)/U(I,I)
        ENDIF
C                                RECEIVE X(I) FROM PROCESSOR JTASK
        CALL MPI_BCAST(X(I),1,MPI_DOUBLE_PRECISION,JTASK,
     &  MPI_COMM_WORLD,IERR)
20    CONTINUE
```

MAX (I+1, ITASK*NB+1), ITASK*NB+NB

(I-1)/NB

```
      CALL MPI_FINALIZE(IERR)
      RETURN
      END
```

Should the program run faster or slower now, or about the same, when $1 << NPES << N$? How much faster or slower (if any)? Explain.

twice as slow:

cyclical distribution $\sum_{i=1}^{N} \frac{N-i}{NPES} \cong \frac{1}{2}\frac{N^2}{NPES}$

1 art processor, block dist. $\sum_{i=1}^{N} NB = N \cdot NB = \frac{N^2}{NPES}$

CPS 5320 Theory-Based Qualifier Exam

0?:00 am – ?:00 am on January 16, 2020

Name: Key

Student ID #: _____

Please read the following instructions carefully

1. This is a closed book exam.
2. The total time for this exam is 1 hour 30 minutes.
3. The exam is worth a total of 100 points.
4. You are permitted to use a simple (non-graphing, non-programmable) calculator. Cell phones, laptops, and all other web-enabled devices are not allowed.
5. Show sufficient work for full credit.

Number	Maximum Points	Earned Points
I	20	
II	10	
III	15	
IV	15	
V	20	
VI	20	
Total	100	

I. At low Reynold's number the 2D Navier Stokes equations reduce to:

$$f1 + \mu(U_{xx} + U_{yy}) = \rho U_t + P_x$$
$$f2 + \mu(V_{xx} + V_{yy}) = \rho V_t + P_y$$
$$U_x + V_y = 0$$

where $(U(x,y,t), V(x,y,t))$ is the fluid velocity vector, and $\mu, \rho, P(x,y,t)$ are the fluid viscosity, density and pressure, and $(f1(x,y,t), f2(x,y,t))$ is the external force field vector.

If we define a stream function $\phi(x,y,t)$ such that $(U,V) = (\phi_y, -\phi_x)$, show that the last (divergence) equation is automatically satisfied, and find the "stream function" formulation which consists of a system of two second order equations involving only ϕ and the "vorticity" $\omega \equiv U_y - V_x$. (Hint: eliminate P from the first two equations.)

$\omega = \phi_{xx} + \phi_{yy}$

$U_x + V_y = \phi_{yx} - \phi_{xy} = 0$

$(f1)_y - (f2)_x + \mu(\omega_{xx} + \omega_{yy}) = \rho\, \omega_t$

II. If 99% of a program is parallelizable, what is the highest speed-up factor that could be expected when going from 1 to 16 processors? (Assume the communication time is negligible).

Amdahl's law

$$\frac{1}{(0.99/16) + 0.01} = 13.91$$

III. Consider the heat equation $\rho C_p T_t = \nabla \bullet [\kappa \nabla \mathbf{T} - \rho C_p T \mathbf{v}] + q$

a. Here $T(x, y, z, t)$, ρ and C_p represent temperature, density and specific heat of the medium. What do $\kappa(x, y, z)$, $\mathbf{v}(x, y, z)$, and $q(x, y, z, t)$ represent physically?

K = conductivity
$\vec{V}$ = convection velocity
q = source/sink generation rate

b. If $\kappa(x, y, z)$ is a discontinuous function, which finite element method, Galerkin or collocation, is better able to handle this case, and why?

Galerkin. $\nabla \cdot (K \nabla T) = K \nabla^2 T + (\nabla K) \cdot \nabla T$

collocation bc to handle → infinite at interface

c. If $\kappa > 0$, the temperature can be specified on the entire boundary. If $\kappa = 0$, on what part of the boundary could the temperature be specified?

part where $\vec{V} \cdot \vec{n} < 0$, ie convection inward

IV. To use PDE2D to solve a PDE in the 3D region above $z = 0$ and below $z = 4 - x^2 - y^2$, you need to describe this paraboloid as $(X(p1, p2, p3), Y(p1, p2, p3), Z(p1, p2, p3))$, with constant limits on the parameters $p1, p2, p3$. Give a possible parameterization, with limits, of this region.

$X = p1 \cdot \cos(p2)$
$y = p1 \cdot \sin(p2)$
$z = p3 \cdot (4 - p1^2)$

$0 \le p1 \le 2$
$0 \le p2 \le 2\pi$
$0 \le p3 \le 1$

or

$x = \sqrt{4 - p3}(\cos p2)\, p1$
$y = \sqrt{4 - p3}(\sin p2)\, p1$
$z = p3$

$0 \le p1 \le 1$
$0 \le p2 \le 2\pi$
$0 \le p3 \le 4$

V. To derive the minimal surface equation, suppose $u(x,y)$ is the surface with $u = g(x,y)$ on the boundary $\partial\Omega$ of Ω which minimizes the surface area $SA(u) \equiv \int\int_\Omega \sqrt{1+u_x^2+u_y^2}\, dA$. Then if $e(x,y)$ is any smooth function with $e = 0$ on the boundary, $SA(u+\alpha e) \geq SA(u)$ for any α. From this we conclude that $f(\alpha) \equiv SA(u+\alpha e)$ has a minimum at $\alpha = 0$, and thus $f'(0) = 0$. Write out the equation $f'(0) = 0$ and then derive a partial differential equation for u.

Hint: you can use the multidimensional integration by parts formula:

$$\iint_\Omega \nabla w \bullet \mathbf{v} = \int_{\partial\Omega} w\mathbf{v} \bullet n - \iint_\Omega w\nabla \bullet \mathbf{v}$$

where w is a scalar function, $\mathbf{v}$ is a vector function.

$$f(\alpha) = SA(u+\alpha e) = \iint \sqrt{1+(u_x+\alpha e_x)^2+(u_y+\alpha e_y)^2}\, dA$$

$$f'(\alpha) = \iint \left(1+(u_x+\alpha e_x)^2+(u_y+\alpha e_y)^2\right)^{-\frac{1}{2}} \left[(u_x+\alpha e_x)e_x + (u_y+\alpha e_y)e_y\right] dA$$

$$0 = f'(0) = \iint \frac{u_x e_x + u_y e_y}{\sqrt{1+u_x^2+u_y^2}}\, dA$$

$$0 = \iint \nabla e \cdot \frac{\nabla u}{\sqrt{1+u_x^2+u_y^2}}\, dA$$

$$= \int_{\partial\Omega} e \frac{\nabla u \cdot n}{\sqrt{1+u_x^2+u_y^2}} - \iint_\Omega e\, \nabla \cdot \left(\frac{\nabla u}{\sqrt{1+u_x^2+u_y^2}}\right) dA$$

$$\Rightarrow \quad 0 = \nabla \cdot \left(\frac{\nabla u}{\sqrt{1+u_x^2+u_y^2}}\right)$$

VI. The subroutine MMUL below computes C=A*B, where A is a matrix and B is a vector, and only one processor is used (NPES=1).

```
      SUBROUTINE MMUL(A,B,C,N,NPES,ITASK)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION A(N,*),B(N),C(N),CI(N)
      include 'mpif.h'
      DO I=1,N
         CI(I) = 0
         DO J=1,N
            CI(I) = CI(I) + A(I,J)*B(J)
         ENDDO
      ENDDO
      CALL MPI_ALLREDUCE(CI,C,N,MPI_DOUBLE_PRECISION,
     &           MPI_SUM,MPI_COMM_WORLD,IERR)
      RETURN
      END
```

a. Modify MMUL, by changing only one line, so that it runs efficiently on NPES processors, if the columns of A are stored cyclically, 0,1,2,...NPES-1,0,1,2,...NPES-1,0,1,2... but each processor actually stores the whole matrix, that is, A is dimensioned A(N,N) in the calling program. Write the modified line below:

DO J = ITASK+1, N, NPES

b. Changing only one more line, modify MMUL so that each processor only stores its columns, that is, A can be dimensioned A(N,(N-1)/NPES+1) (using ALLOCATE) in the calling program. Write the modified line below:

CI(I) = CI(I) + A(I, (J-1)/NPES+1) * B(J)

CPS 5320 Qualifier Exam

9:00 am – 12:00 noon on Friday, Aug 20, 2021

Name: Key

Student ID #: _____

Please read the following instructions carefully

1. This is a closed book exam.
2. The total time for this exam is 3 hours.
3. The exam is worth a total of 100 points.
4. You are permitted to use a simple (non-graphing, non-programmable) calculator. Cell phones, laptops, and all other web-enabled devices are not allowed.
5. Show sufficient work for full credit.

Number	Maximum Points	Earned Points
Ia	5	
b	5	
c	5	
d	5	
e	5	
II	10	
IIIa	10	
b	5	
c	5	
d	5	
IVa	10	
b	5	
V	10	
VIa	5	
b	5	
VII	5	
Total	100	

I. Consider the following MPI Fortran program:

```
      PARAMETER (N=100)
      INTEGER X(N),SUMI,SUM
      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)
      SUM = 0
      DO I=ITASK+1,N,NPES
        X(I) = I**2
      ENDDO
C
      SUMI = 0
      DO I=ITASK+1,N,NPES
        SUMI = SUMI + X(I)
      ENDDO
      iroot = 0
      CALL MPI_REDUCE(SUMI,SUM,1,MPI_INTEGER,
     &                MPI_SUM,iroot,MPI_COMM_WORLD,IERR)
      CALL MPI_BCAST(SUM,1,MPI_INTEGER,iroot,MPI_COMM_WORLD,IERR)
      IF (ITASK.EQ.1) PRINT *, SUM
      CALL MPI_FINALIZE(IERR)
      STOP
      END
```

ALLOCATABLE X(:)

ALLOCATE (X((N-1)/NPES+1))

X((I-1)/NPES+1)

a. What is the output of this program (approximately), if run on several processors?

$$\frac{N(N+1)(2N+1)}{6} = 338350 \approx \frac{1}{3}N^3$$

b. Show exactly how you could replace the MPI_REDUCE and MPI_BCAST calls by a single MPI call that has the same effect.

CALL MPI_ALLREDUCE (SUMI, SUM, 1, MPI_INTEGER, MPI_SUM, MPI_COMM_WORLD, IERR)

c. Modify the program of [a.] so that each processor only stores its elements of the vector X. Mark your changes on the listing above.

(See above)

d. Complete the lines LIM1= and LIM2= so that the program below does the same calculations, but with the elements of X distributed by blocks, that is, the first processor keeps up with the first NB elements of X, etc. (You can assume N is a multiple of NPES.)

```
      PARAMETER (N=100)
      INTEGER X,SUMI,SUM
      ALLOCATABLE X(:)
      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)
      NB = N/NPES
      ALLOCATE (X(N))
      LIM1 =
      LIM2 =
      SUM = 0
      DO I=LIM1,LIM2
         X(I) = I**2
      ENDDO
C
      SUMI = 0
      DO I=LIM1,LIM2
         SUMI = SUMI + X(I)
      ENDDO
      iroot = 0
      CALL MPI_REDUCE(SUMI,SUM,1,MPI_INTEGER,
     &                 MPI_SUM,iroot,MPI_COMM_WORLD,IERR)
      CALL MPI_BCAST(SUM,1,MPI_INTEGER,iroot,MPI_COMM_WORLD,IERR)
      IF (ITASK.EQ.1) PRINT *, SUM
      CALL MPI_FINALIZE(IERR)
      STOP
      END
```

X(NB)

ITASK×NB+1

ITASK×NB+NB (2)

IME=1

IME

IME=IME+1

IME=1

IME

e. Now modify the program of [d.] so that each processor only stores its elements of X. Mark your changes on the listing above.

(see above)

alternative: replace X(I) by X(MOD(I-1,NB)+1)

II. When the forward elimination phase of Gauss elimination is parallelized, distributing the columns cyclically (as in [Ia.]) was more efficient than distributing by blocks (as in [Id.]). Why? What about the back substitution phase, is cyclic distribution better for that than block distribution also? What about the programs of problem I above?

When column $I > NB$ is active, some processors are no longer used, if block distribution is used.
Yes, better also for back substitution, same reason.
Doesn't matter for Σi^2 (problem I)

III. The convection/diffusion equation is $U_t = \nabla \cdot [D\nabla U - \mathbf{V}U] + q$, where $U(x, y, z, t)$ is the density of a substance, $q(x, y, z, t)$ is the generation rate, $\mathbf{V}(x, y, z, t)$ is the convection velocity vector and $D(x, y, z, t)$ is the diffusion coefficient.

a. Derive this equation by setting the rate of change of mass of the substance in an arbitrary subregion S equal to the rate the substance is crossing the boundary into S plus the rate it is generated by sources/sinks. The diffusive flux is $-D\nabla U$ by Fick's law, and the convective flux is $\mathbf{V}U$. (Hint: you will need to use the divergence theorem.)

$$\frac{d}{dt}\iiint_S u = \iint_{\partial S} (-D\nabla u + \vec{v}u)\cdot[-n] + \iiint_S q$$

$$\iiint_S u_t = \iiint_S \nabla\cdot(D\nabla u - \vec{v}u) + q$$

so

$$u_t = \nabla\cdot(D\nabla u - \vec{v}u) + q$$

b. If $D = 0$, on what part of the boundary should U be specified?

part where $V \cdot n < 0$, ie, where flux is inward

c. What is the boundary condition on a boundary where there is no flux? (Assume $\mathbf{V} = \mathbf{0}$)

$$\frac{\partial u}{\partial n} = \nabla u \cdot n = 0$$

d. If D is discontinuous, which finite element method is better, Galerkin or collocation?

Galerkin

IV. The 2D Navier Stokes equations are:

$$\rho(U_t + UU_x + VU_y) = f1 + \mu(U_{xx} + U_{yy}) - P_x$$
$$\rho(V_t + UV_x + VV_y) = f2 + \mu(V_{xx} + V_{yy}) - P_y$$
$$U_x + V_y = 0$$

where (U, V) is the fluid velocity vector, and ρ, μ, P are the fluid density, viscosity and pressure, and $(f1, f2)$ is the external force field.

a. If we define a stream function $\phi(x, y, t)$ such that $(U, V) = (\phi_y, -\phi_x)$, show that the last (divergence) equation is automatically satisfied, and find a system of two second order equations involving ϕ and the "vorticity" $\omega \equiv U_y - V_x$.

$$\omega = \phi_{xx} + \phi_{yy}$$

$$\rho \omega_t + \rho(\phi_y \omega_x - \phi_x \omega_y) + (f_2)_x - (f_1)_y$$
$$= \mu(\omega_{xx} + \omega_{yy})$$

b. The alternative for solving these equations is the "penalty" method. Explain how the Navier-Stokes equations are modified if the penalty method is used. Does the penalty method work for 3D problems also?

p replaced by $-\alpha(u_x+v_y)$, α = large number

yes, works in 3D also

V. If the height of an elastic membrane $u(x,y,t)$ satisfies the PDE
$\rho u_{tt} + bu_t = T\nabla^2 u$

where b is the damping coefficient, T is the tension, ρ is the density, then the energy of the membrane (kinetic plus potential) is given by:

$E(t) = \iint_\Omega \left[\frac{1}{2}\rho u_t^2 + \frac{T}{2}\nabla u \cdot \nabla u\right] dA$

Show that if the membrane is fixed on the boundary, or satisfies the boundary condition $\frac{\partial u}{\partial n} = 0$, then the energy continually decreases if $b > 0$ and is constant if $b = 0$, (Hint: use $\nabla \cdot (w\nabla u) = w\nabla^2 u + \nabla w \cdot \nabla u$.)

$$E'(t) = \iint_\Omega \rho u_t u_{tt} + T\nabla u \cdot \nabla u_t$$

$$= \iint_\Omega \rho u_t u_{tt} + T\nabla\cdot(u_t \nabla u) - T u_t \nabla^2 u$$

$$= \int_{\partial\Omega} T u_t \nabla u \cdot \vec{n}^{\,0} + \iint_\Omega u_t(\rho u_{tt} - T\nabla^2 u)$$

$$= \iint_\Omega u_t(-bu_t) = -\iint_\Omega b u_t^2 \le 0$$

$E'(t) < 0$ if $b > 0$

$E'(t) = 0$ if $b = 0$

VI. The Black-Scholes equation for the value V(s,t) of an option is:

$$V_t + \frac{1}{2}\sigma_1^2 s^2 V_{ss} + rsV_s - rV = 0$$

a. This is qualitatively similar to a diffusion or heat equation, with one major difference, which affects how the initial conditions are treated. What is the major difference and how does that affect the initial conditions? What happens if you try to treat the initial conditions as done for the diffusion equation?

Diffusion coefficient $\left(-\frac{1}{2}\sigma_1^2 s^2\right)$ is negative, so final condition given, must integrate backward. Otherwise unstable

b. The Black-Scholes equation is derived assuming what type of probability distribution for the future price of the asset, and what is the mean of this distribution? Qualitatively, how does this distribution change as time passes?

log normal distribution $\mu = se^{rt}$, ie, current price adjusted for inflation, σ increases with time

VII. If PDE2D is used to solve a 3D PDE problem in the cone $0 \le z \le 2 - \sqrt{x^2 + y^2}$, the cone must be parameterized as $(X(p1, p2, p3), Y(p1, p2, p3), Z(p1, p2, p3))$, with constant limits on the parameters $p1, p2, p3$. Give a possible parameterization, including the parameter limits.

$X = \rho 1\,(2 - \rho 3)\cos(\rho 2)$

$y = \rho 1\,(2 - \rho 3)\sin(\rho 2)$

$z = \rho 3$

$0 \le \rho 1 \le 1$

$0 \le \rho 2 \le 2\pi$

$0 \le \rho 3 \le 2$

other parameterizations possible