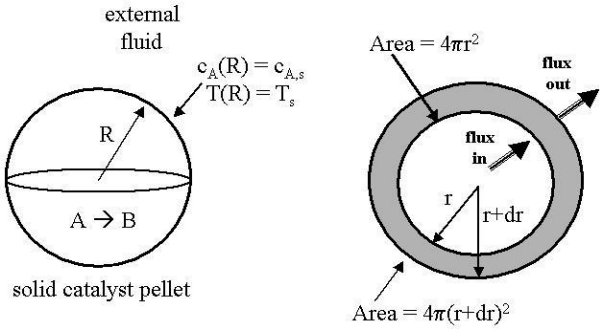
10.34. Numerical Methods Applied to Chemical Engineering
Homework #3. Nonlinear algebraic equations and matrix eigenvalue problems

SOLUTION

Problem 1. Modeling chemical reaction with diffusion in a catalyst pellet

In HW # 1, we considered the use of the finite difference method to convert a boundary value problem into a set of linear algebraic equations. In HW # 2, we considered the use of Newton's method to solve sets of nonlinear algebraic equations. In this problem, we combine these two approaches to solve a boundary value problem that contains a nonlinear term involving chemical reaction. This problem is inspired by the discussion of nonisothermal catalyst particles in section 3.9 of G. F. Froment and K. B. Bischoff, Chemical Reactor Analysis and Design, 2nd ed., Wiley, 1990.

We consider the case of a spherical catalyst pellet of radius *R* (see figure below).



We want to write a MATLAB program that will employ a finite difference method to solve for the dimensionless concentration profile and the effectiveness factor. The governing equation in dimensionless form is

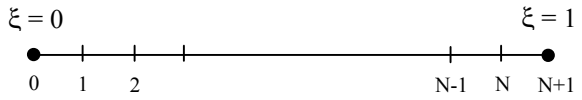
$$\frac{d}{d\xi}\left[\xi^2 \frac{d\Psi_A}{d\xi}\right] - \xi^2 \Phi^2 \exp\left[\frac{\gamma\beta(1-\Psi_A)}{1+\beta(1-\Psi_A)}\right]\Psi_A = 0 \qquad \text{(EQ 56)}$$

subject to the boundary conditions

$$\Psi_A(1) = 1 \qquad \left.\frac{d\Psi_A}{d\xi}\right|_{\xi=0} = 0 \qquad \text{(EQ 57)}$$

where Ψ_A is the dimensionless concentration inside the catalyst pellet and ξ is the dimensionless radius of the particle. The program should take three input parameters: Φ, γ, and β, which are defined in EQ (50), (51), and (54) of the assignment sheet, respectively.

We are going to solve this problem using finite differences. Let's number the grid points in the following way:



Discretizing the second derivative in (EQ 56) using 'central' finite differences

$$\left.\frac{d}{d\xi}\left[\xi^2\frac{d\Psi_A}{d\xi}\right]\right|_{\xi_j} = \frac{\xi_{j+1/2}^2\left(\left.\frac{d\Psi_A}{d\xi}\right|_{\xi_{j+1/2}}\right)-\xi_{j-1/2}^2\left(\left.\frac{d\Psi_A}{d\xi}\right|_{\xi_{j-1/2}}\right)}{\xi_{j+1/2}-\xi_{j-1/2}} =$$

$$= \frac{\xi_{j+1/2}^2\left(\frac{\Psi_{A,j+1}-\Psi_{A,j}}{\xi_{j+1}-\xi_j}\right)-\xi_{j-1/2}^2\left(\frac{\Psi_{A,j}-\Psi_{A,j-1}}{\xi_j-\xi_{j-1}}\right)}{\xi_{j+1/2}-\xi_{j-1/2}}$$

$$= A_{j,j-1}\Psi_{A,j-1} + A_{j,j}\Psi_{A,j} + A_{j,j+1}\Psi_{A,j+1} \qquad \text{(EQ 27}^*\text{)}$$

where

$$A_{j,j-1} = \frac{\xi_{j-1/2}^2}{\left(\xi_j-\xi_{j-1}\right)\left(\xi_{j+1/2}-\xi_{j-1/2}\right)}$$

$$A_{j,j} = \left(-\frac{1}{\xi_{j+1/2}-\xi_{j-1/2}}\right)\left[\frac{\xi_{j+1/2}^2}{\xi_{j+1}-\xi_j}+\frac{\xi_{j-1/2}^2}{\xi_j-\xi_{j-1}}\right]$$

$$A_{j,j+1} = \frac{\xi_{j+1/2}^2}{\left(\xi_{j+1}-\xi_j\right)\left(\xi_{j+1/2}-\xi_{j-1/2}\right)}$$

The nonlinear algebraic equation for an interior grid point j, $j \in [2, N-1]$, takes the form

$$A_{j,j-1}\Psi_{A,j-1} + A_{j,j}\Psi_{A,j} + A_{j,j+1}\Psi_{A,j+1} - \xi_j^2\Phi^2\exp\left[\frac{\gamma\beta(1-\Psi_{A,j})}{1+\beta(1-\Psi_{A,j})}\right]\Psi_{A,j} = 0 \qquad \text{(EQ 28}^*\text{)}$$

As described on page 5 of the assignment sheet, some modification of the above equation is required for the first and last grid points. At the last grid point, $j = N$, the discretized form of the differential equation is

$$A_{N,N-1}\Psi_{A,N-1} + A_{N,N}\Psi_{A,N} + A_{N,N+1}\Psi_{A,N+1} - \xi_N^2\Phi^2\exp\left[\frac{\gamma\beta(1-\Psi_{A,N})}{1+\beta(1-\Psi_{A,N})}\right]\Psi_{A,N} = 0 \qquad \text{(EQ 30}^*\text{)}$$

where $\Psi_{A,N+1} = \Psi_A\big|_{\xi=1} = 1$ from the boundary condition.

The modification for grid point $j = 1$ is slightly more involved and is described in detail on pp. 6-7 of the assignment sheet. Using the second boundary condition, we arrived at the relation

$$\Psi_{A,0} = \frac{4\Psi_{A,1} - \Psi_{A,2}}{3} \qquad \text{(EQ 42}^*\text{)}$$

and the algebraic equation therefore became

$$\left[A_{1,1} + \frac{4}{3}A_{1,0}\right]\Psi_{A,1} + \left[A_{1,2} - \frac{1}{3}A_{1,0}\right]\Psi_{A,2} - \xi_1^2\Phi^2 \exp\left[\frac{\gamma\beta(1-\Psi_{A,1})}{1+\beta(1-\Psi_{A,1})}\right]\Psi_{A,1} = 0 \qquad \text{(EQ 43}^*\text{)}$$

As was mentioned in the hints for this assignment, the fsolve() routine is executed much faster if the Jacobian matrix is calculated analytically and supplied to the subroutine. The general form of the Jacobian is

$$Jac = \begin{bmatrix} df_1/dx_1 & df_1/dx_2 & \dots & \dots & df_1/dx_N \\ df_2/dx_1 & df_2/dx_2 & \dots & \dots & df_2/dx_N \\ \vdots & & & & \\ \vdots & & & & \\ df_N/dx_1 & df_N/dx_2 & & & df_N/dx_N \end{bmatrix}$$

In our case, $[f_1 \ldots f_N]$ will be expressions in (EQ 28*), (EQ 30*), and (EQ 43*) and $[x_1 \ldots x_N]$ are $[\Psi_{A,1} \ldots \Psi_{A,N}]$. Given that these three equations are at most functions of three different $\Psi_{A,j}$, the Jacobian matrix will have at most three non-zero entries in each row. For completeness, let's find the entries of the Jacobian for an interior point j.

$$Jac(j, j-1) = \frac{df_j}{d\Psi_{A,j-1}} = A_{j,j-1}$$

$$Jac(j, j+1) = \frac{df_j}{d\Psi_{A,j+1}} = A_{j,j+1}$$

$$Jac(j, j) = \frac{df_j}{d\Psi_{A,j}} = A_{j,j} - \xi_j^2\Phi^2 \frac{d}{d\Psi_{A,j}}\left\{\exp\left[\frac{\gamma\beta(1-\Psi_{A,j})}{1+\beta(1-\Psi_{A,j})}\right]\Psi_{A,j}\right\}$$

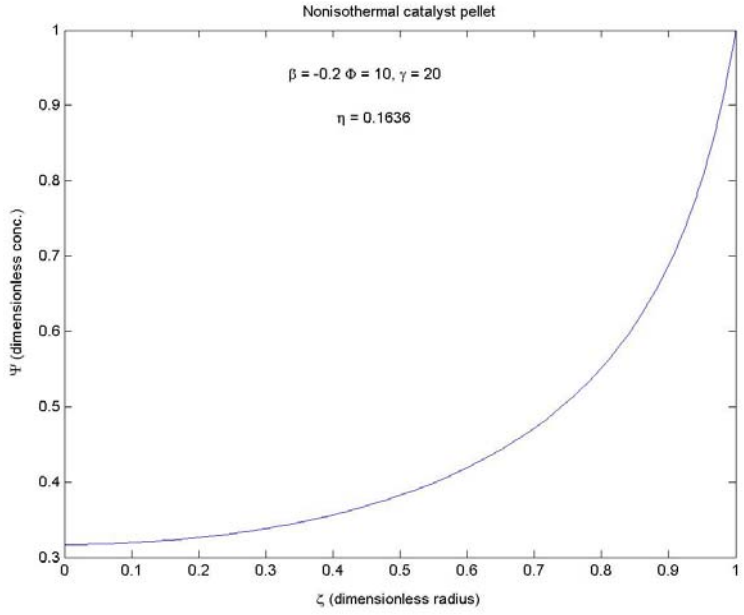
$$= A_{j,j} - \xi_j^2\Phi^2\left(\Theta + \Psi_{A,j}\frac{d\Theta}{d\Psi_{A,j}}\right)$$

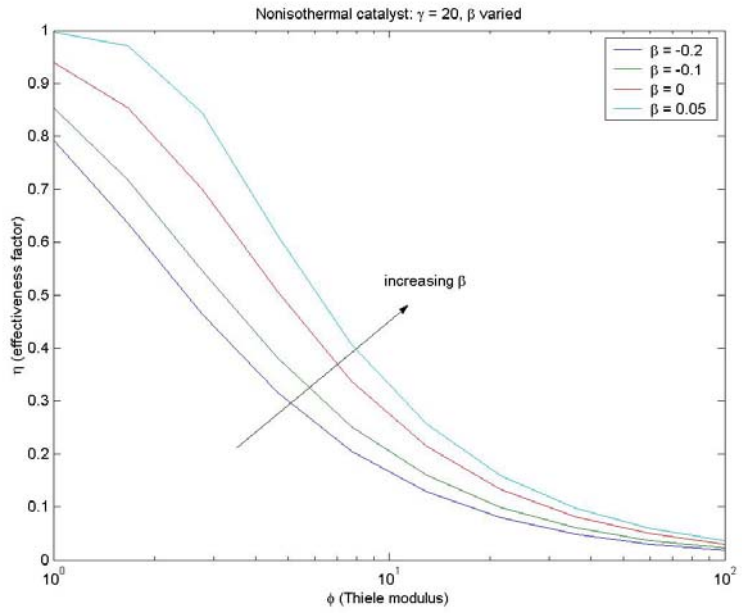
where

$$\Theta = \exp\left[\frac{\gamma\beta(1-\Psi_{A,j})}{1+\beta(1-\Psi_{A,j})}\right] \qquad \text{and therefore}$$

$$\frac{d\Theta}{d\Psi_{A,j}} = \Theta\left\{\frac{(-\gamma\beta)}{1+\beta(1-\Psi_{A,j})} + \frac{\gamma\beta^2(1-\Psi_{A,j})}{\left[1+\beta(1-\Psi_{A,j})\right]^2}\right\}$$

At this point, we are ready to code in the problem in order to calculate the concentration profile. Once that is done, this profile can be used to calculate the effectiveness factor from (EQ 61) using the trapz() function. The first of the attached plots is a concentration profile for a particular set of input parameters, and the second plot is the master plot required by Question A.2.





```
% PROBLEM 1 CODE
% catalyst_nonisothermal_scan_v2.m
%
% This MATLAB program plots the effectiveness
% factor vs. the Thiele modulus for a catalyst
% pellet with first order reaction at steady
% state in nonisothermal operation.
% No external mass or heat transfer resistance
% is included, and Diriclet boundary conditions
% are employed at the particle surface.
%
% To solve the dimensionless PDE's, finite
% differences are used with a non-uniform
% grid (except near r=0 for discretized
% symmetry boundary condition).
%
% The dimensionless input parameters are :
% Phi_TM = Thiele modulus
%        = product of particle radius and
%          the square root of the rate
%          constant divided by the diffusivity
%        = measure of importance of diffusion
%          limitations
% gamma = activation energy / (R*Ts), where
%         Ts is the known surface temperature
% beta = Da*(-DH)*Cs/lambda/Ts, where Da is
%         the diffusivity, DH is the heat of
%         reaction, Cs is the known surface
%         concentration, lambda is the thermal
%         conductivity, and Ts is the known
%         surface temperature
%      = measure of importance of heat generation
%        by reaction vs. thermal conductivity
%
% Written K. Beers. MIT ChE. 9/21/2002
% v2. 9/21/2002. modified from catalyst_nonisothermal_scan_v2.m
%          to compute the Jacobian analystically, saving
%          time required to perform finite differences
%
%  COMPARISON :  beta = 0, gamma = 0,
%               phi_TM_min = 1e-2, phi_TM_max = 1e2,
%               num_TM = 20, dz = 0.01, dz_fine = 0.001
%          catalyst_nonisothermal_scan.m: CPU = 222.56
%          catalyst_nonisothermal_scan_v2.m: CPU = 22.612
%                (BUT HAD WARNINGS SINCE USED SPARSE
%                   JACOBIAN, STILL WORKS)

function iflag_main = catalyst_nonisothermal_scan()

cpu_start = cputime;

iflag_main = 0;


% First, have the user input the simulation parameters.
%disp('Enter simulation parameters, ...');
```

```
% problem parameters
%beta = input('Enter beta (row vector) : ');
beta = [-0.2;-0.1;0;0.05]
num_beta = length(beta);
gamma = input('Enter gamma : ');

% Thiele modulus bounds (log scale)
phi_TM_min = input('Enter min. Thiele modulus : ');
phi_TM_max = input('Enter max. Thiele modulus : ');
num_TM = input('Enter number of Thiele modulus points : ');
phi_TM = logspace(log10(phi_TM_min),log10(phi_TM_max),num_TM);

% number of physical grid points, use simple uniform scale with fine grid near surface
dz = input('Enter interior grid point spacing : ');
dz_fine = input('Enter surface (0.9-1.0) spacing : ');
z_grid = [ [dz : dz : 0.9-dz], [0.9 : dz_fine : 1.0-dz_fine] ];
num_grid = length(z_grid);

% We now scan the values of the Thiele modulus, starting at the lowest value.  For each value of
% the Thiele modulus, we need to solve the set of nonlinear algebraic equations obtained by finite
% difference disrectization of the governing equations. We start at the lowest value of the Thiele
% modulus, and use as an initial guess a completely uniform concentation profile.  In the limit of
% the Thiele modulus goes to zero, this initial guess is correct.  For each subsequent calculations,
% we use the final result of the last calculation as an initial guess.

% allocate space to store the effectiveness factor
eta_eff = zeros(num_beta,num_TM);
figure;

for k_beta = 1:num_beta;

% form initial guess - uniform concentration
psi_guess = ones(size(z_grid));

% set some options flags for fsolve()
options = optimset('TolFun',1e-8, 'Display','off', 'LargeScale','off',...
   'Jacobian','on');

% iterate over each value of the Thiele modulus
do_plot = 0;
for i_TM = 1:num_TM

   % compute dimensionless profile
   [psi,fval,exiflag] = ...
      fsolve(@catalyst_nonisothermal_calc_f, ...
      psi_guess, options, ...
      phi_TM(i_TM), beta(k_beta), gamma, z_grid);

   if(do_plot)
      figure;
      plot(z_grid,psi);
      xlabel('\zeta (dimensionless radius)');
      ylabel('\Psi (dimensionless conc.)');
      title('Nonisothermal catalyst pellet');
   end
```

```
   % compute effectiveness factor
      eta_eff(k_beta,i_TM) = calc_eta_eff(psi, ...
      phi_TM(i_TM), beta(k_beta), gamma, z_grid)
%       disp(['eta = ' int2str(eta_eff(k_beta,i_TM))]);

end

% Now, plot out the results
semilogx(phi_TM,eta_eff);
%gtext(num2str(beta(k_beta)));
hold on;

end
legend('\beta = -0.2','\beta = -0.1','\beta = 0','\beta = 0.05')

xlabel('\phi (Thiele modulus)');
ylabel('\eta (effectiveness factor)');
title(['Nonisothermal catalyst: ', '\gamma = ', num2str(gamma), ' \beta varied']);

iflag_main = 1;
cpu_end = cputime;

cpu_elapsed = cpu_end - cpu_start;
disp(['Elapsed CPU time : ', num2str(cpu_elapsed)]);

return;


% ============================================================
% This routines calculates the function values for the
% set of equations that sovle for the dimensionless
% concentration profile within the catalyst pellet
% for a nonisothermal first order reaction.
% K. Beers. MIT ChE. 9/21/2002

function [fval,Jac] = catalyst_nonisothermal_calc_f(psi, ...
   phi_TM, beta, gamma, z_grid);

num_grid = length(z_grid);

% allocate space for function vector
% and Jacobian
fval = zeros(size(psi));
Jac = spalloc(num_grid,num_grid,3*num_grid);

% For each point except the first and last, compute
% the value of the function.
for j=2:(num_grid-1)

   % estimate the local second derivative in
   % spherical coordinates
   z_mid_up = (z_grid(j+1)+z_grid(j))/2;
   z_mid_lo = (z_grid(j)+z_grid(j-1))/2;
   dz_mid = z_mid_up - z_mid_lo;
   dz_up = z_grid(j+1) - z_grid(j);
```

```
   dz_lo = z_grid(j) - z_grid(j-1);

   A_lo = (z_mid_lo^2)/(dz_lo*dz_mid);
   A_mid = (-1/dz_mid)*((z_mid_up^2)/dz_up + (z_mid_lo^2)/dz_lo);
   A_up = (z_mid_up^2)/(dz_up*dz_mid);

   second_deriv = A_lo*psi(j-1) + A_mid*psi(j) +A_up*psi(j+1);

   % compute nonlinear reaction term
   theta = exp(  gamma*beta*(1-psi(j)) / (1 + beta*(1-psi(j))));
   rxn = (z_grid(j)^2)*(phi_TM^2)*theta*psi(j);

   % compute function value
   fval(j) = second_deriv - rxn;

   % compute Jacobian elements
   Jac(j,j-1) = A_lo;
   Jac(j,j+1) = A_up;
   dtheta_dpsi = theta* ( ...
      (-gamma*beta)/(1+beta*(1-psi(j))) + ...
      (gamma*beta^2*(1-psi(j)))/((1+beta*(1-psi(j)))^2));
   drxn_dpsi = (z_grid(j)^2)*(phi_TM^2) * (theta + psi(j)*dtheta_dpsi);
   Jac(j,j) = A_mid - drxn_dpsi;

end


% Now, calculate function value for last point, employing known concentration at the
% catalyst surface.
z_mid_up = (1+z_grid(num_grid))/2;
z_mid_lo = (z_grid(num_grid) + ...
   z_grid(num_grid-1))/2;
dz_mid = z_mid_up - z_mid_lo;
dz_up = 1 - z_grid(num_grid);
dz_lo = z_grid(num_grid) - z_grid(num_grid-1);
A_lo = (z_mid_lo^2)/(dz_lo*dz_mid);
A_mid = (-1/dz_mid)*((z_mid_up^2)/dz_up + ...
                (z_mid_lo^2)/dz_lo);
A_up = (z_mid_up^2)/(dz_up*dz_mid);
second_deriv = A_lo*psi(num_grid-1) + ...
   A_mid*psi(num_grid) + A_up;
% compute nonlinear reaction term
theta = exp(  gamma*beta*(1-psi(num_grid)) / ...
             (1 + beta*(1-psi(num_grid))));
rxn = (z_grid(num_grid)^2)*(phi_TM^2)*theta*psi(num_grid);
% compute function value
fval(num_grid) = second_deriv - rxn;
% compute Jacobian elements
j = num_grid;
Jac(j,j-1) = A_lo;
dtheta_dpsi = theta* ( ...
   (-gamma*beta)/(1+beta*(1-psi(j))) + ...
   (gamma*beta^2*(1-psi(j)))/((1+beta*(1-psi(j)))^2));
drxn_dpsi = (z_grid(j)^2)*(phi_TM^2) * ...
   (theta + psi(j)*dtheta_dpsi);
Jac(j,j) = A_mid - drxn_dpsi;
```

```
% Then, calculate function value for
% first point, using symmetry boundary
% condition.
z_mid_up = (z_grid(2)+z_grid(1))/2;
z_mid_lo = (z_grid(1))/2;
dz_mid = z_mid_up - z_mid_lo;
dz_up = z_grid(2) - z_grid(1);
dz_lo = z_grid(1);
A_lo = (z_mid_lo^2)/(dz_lo*dz_mid);
A_mid = (-1/dz_mid)*((z_mid_up^2)/dz_up + ...
                (z_mid_lo^2)/dz_lo);
A_up = (z_mid_up^2)/(dz_up*dz_mid);
second_deriv = (A_mid+4*A_lo/3)*psi(1) + ...
   (A_up - A_lo/3)*psi(2);
% compute nonlinear reaction term
theta = exp(  gamma*beta*(1-psi(1)) / ...
            (1 + beta*(1-psi(1))));
rxn = (z_grid(1)^2)*(phi_TM^2)*theta*psi(1);
% compute function value
fval(1) = second_deriv - rxn;
% compute Jacobian elements
j = 1;
Jac(j,j+1) = A_up - A_lo/3;
dtheta_dpsi = theta* ( ...
   (-gamma*beta)/(1+beta*(1-psi(j))) + ...
   (gamma*beta^2*(1-psi(j)))/((1+beta*(1-psi(j)))^2));
drxn_dpsi = (z_grid(j)^2)*(phi_TM^2) * ...
   (theta + psi(j)*dtheta_dpsi);
Jac(j,j) = A_mid + 4*A_lo/3; - drxn_dpsi;


return;


% ========================================================
% This routine computes the effectiveness factor from
% the dimensionless concentration profile for a
% catalyst particle with a nonisothermal first order
% reaction.
% K. Beers. MIT ChE. 9/21/2002
function eta_eff = calc_eta_eff(psi, ...
       phi_TM, beta, gamma, z_grid);

% compute the augmented z grid including the end
% points
z_aug = [0, z_grid, 1];
psi_aug = [0, psi, 1];
psi_aug(1) = (4*psi(1)-psi(2))/3;

% Now, compute on this augmented grid the values
% of the integrand.
integrand = zeros(size(psi_aug));
for j=1:length(integrand)
   var1 = gamma*beta*(1-psi_aug(j)) / ...
      (1 + beta*(1-psi_aug(j)));
   integrand(j) = 3*psi_aug(j)*(z_aug(j)^2)*exp(var1);
```

```
end

% Use the trapezoid rule to estimate the value of the
% integral.

eta_eff = trapz(z_aug,integrand)

return;
```

Problem 2. Eigenvalues of 3x3 matrices

In this problem we are given the following three 3x3 matrices:

$$A=\begin{bmatrix} 2 & 1 & -1 \\ 1 & 3 & 0 \\ -1 & 0 & 4 \end{bmatrix} \qquad B=\begin{bmatrix} 1 & 2 & -3 \\ -2 & -3 & 4 \\ 3 & -4 & 5 \end{bmatrix} \qquad C=\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

<u>Question 2.A</u>. Which of these matrices can you tell by inspection, without computation, must have all eigenvalues be real numbers?

Since $A = A^{\mathrm{T}}$ and $C = C^{\mathrm{T}}$, matrices A and C will have real eigenvalues.

<u>Question 2.B</u>. Using Gershogen's theorem, what are the bounds on the possible values of the eigenvalues of A? Is it possible that any of the eigenvalues of A are negative?

From Eqn. (3.1.2-9) of the class notes,

$$r_1 = |A_{12}| + |A_{13}| = 2$$
$$r_2 = |A_{21}| + |A_{23}| = 1$$
$$r_3 = |A_{31}| + |A_{32}| = 1$$

Using Eqn. (3.1.2-10), we can get the bounds on the three eigenvalues.

$$|\lambda_1 - A_{11}| \le r_1$$
$$|\lambda_1 - 2| \le 2 \qquad \Rightarrow 0 \le \lambda_1 \le 4$$

$$|\lambda_2 - A_{22}| \le r_2$$
$$|\lambda_2 - 3| \le 1 \qquad \Rightarrow 2 \le \lambda_2 \le 4$$

$$|\lambda_3 - A_{33}| \le r_3$$
$$|\lambda_3 - 4| \le 1 \qquad \Rightarrow 3 \le \lambda_3 \le 5$$

Based on the limits, we see that all of the eigenvalues of A will be positive.

<u>Question 2.C</u>. Which of these matrices are normal?

A real matrix D is said to be normal if the following condition holds : $DD^{\mathrm{T}} = D^{\mathrm{T}}D$. As a special case, we see that a real, symmetric matrix has $D^{\mathrm{T}} = D$, so that it is obviously

normal. Having said this, based on our answer to Q.2.A., matrices *A* and *C* are normal. What about matrix *B*?

$$BB^T = \begin{pmatrix} 14 & -20 & -20 \\ -20 & 29 & 26 \\ -20 & 26 & 50 \end{pmatrix} \text{ and}$$

$$B^TB = \begin{pmatrix} 14 & -4 & 4 \\ -4 & 29 & -38 \\ 4 & -38 & 50 \end{pmatrix}$$

Since the condition stated above does not hold, matrix *B* is NOT normal.

<u>Question 2.D</u>. For every matrix that you know has all real eigenvalues, compute the eigenvalues and the normalized (unit-length) eigenvectors by hand. Show your calculations.

Let's look at matrix *A* first. To find the eigenvalues, we need to derive a characteristic polynomial in λ's, which is done in the following way.

$$\det(A-\lambda I) = \left| \begin{pmatrix} 2-\lambda & 1 & -1 \\ 1 & 3-\lambda & 0 \\ -1 & 0 & 4-\lambda \end{pmatrix} \right| = 0$$

The characteristic polynomial is then

$$(2-\lambda)(3-\lambda)(4-\lambda) - 1(4-\lambda) + (-1)(0+(3-\lambda)) = 0$$

Rearranging, we get

$$-\lambda^3 + 9\lambda^2 - 24\lambda + 17 = 0$$

Solving for the three eigenvalues, we get λ_1=1.1206, λ_2=3.3473, and λ_3=4.5321.

Let's put expression A$\boldsymbol{v}$ = λ$\boldsymbol{v}$, where $\boldsymbol{v}$ is the eigenvector corresponding to a particular eigenvalue, in the equation form.

$$\begin{bmatrix} 2 & 1 & -1 \\ 1 & 3 & 0 \\ -1 & 0 & 4 \end{bmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \lambda \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

$$2v_1 + v_2 - v_3 = \lambda v_1 \qquad (1)$$
$$v_1 + 3v_2 \qquad = \lambda v_2 \qquad (2)$$
$$-v_1 \quad + 4v_3 = \lambda v_3 \qquad (3)$$
$$v_1^2 + v_2^2 + v_3^2 = 1 \qquad (4)$$

Equation (4) ensures that the eigenvector has unit length. From (2),
$v_1 = -v_3(4-\lambda) \qquad (5)$

Adding (1) and (2), we get

$$3v_2 + 4v_3 = \lambda(v_2 + v_3)$$
$$\Rightarrow v_2 = -v_3\left[\frac{4-\lambda}{3-\lambda}\right] \qquad (6)$$

Substituting (5) and (6) into (4),

$$\left[-v_3(4-\lambda)\right]^2 + \left[-v_3\left(\frac{4-\lambda}{3-\lambda}\right)\right]^2 + v_3^2 = 1 \qquad (7)$$

Solving for v_3 from (7)

$$v_3 = \left[(4-\lambda)^2 + \left(\frac{4-\lambda}{3-\lambda}\right)^2 + 1\right]^{-1/2} \qquad (8)$$

Thus, we use Eqns. (8), (5), and (6) to get expressions for the three components of the eigenvector for a given value of λ. The results are:

$\lambda_1 = 1.1206$ $\qquad$ $\lambda_2 = 3.3473$ $\qquad$ $\lambda_3 = 4.5321$

$$\underset{-1}{v} = \begin{pmatrix} 0.8440 \\ -0.4491 \\ 0.2931 \end{pmatrix} \qquad \underset{-2}{v} = \begin{pmatrix} 0.2931 \\ 0.8440 \\ 0.4491 \end{pmatrix} \qquad \underset{-3}{v} = \begin{pmatrix} -0.4491 \\ -0.2931 \\ 0.8440 \end{pmatrix}$$

Now let's look at matrix *C*.

$$\det(C - \lambda I) = \left\| \begin{pmatrix} 1-\lambda & 0 & 0 \\ 0 & -\lambda & 1 \\ 0 & 1 & -\lambda \end{pmatrix} \right\| = 0$$

and the characteristic polynomial is

$$(\lambda^2 - 1)(1-\lambda) = 0$$
$$\Rightarrow \lambda_1 = 1,\ \lambda_2 = 1, \text{ and } \lambda_3 = -1$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = \lambda \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$$

We can apply exactly the same methodology here as we did for matrix *A*, but given the simple structure of matrix *C*, it might be quicker to find eigenvalues by inspection. For λ=1, we can take $v_1 = 1$, and $v_2 = v_3 = 0$. Another option for this repeated root is $v_1 = 0$ and $v_2 = v_3 = (1/2)^{0.5} = 0.7071$. For λ = -1, we can choose $v_1 = 0$ and $v_2 = (1/2)^{0.5} = 0.7071$ and $v_3 = -v_2$. In summary,

$$\lambda_1 = 1 \qquad \lambda_2 = 1 \qquad \lambda_3 = -1$$
$$\underset{-1}{v} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \qquad \underset{-2}{v} = \begin{pmatrix} 0 \\ 0.7071 \\ 0.7071 \end{pmatrix} \qquad \underset{-3}{v} = \begin{pmatrix} 0 \\ -0.7071 \\ 0.7071 \end{pmatrix}$$