3.2 Determine the root of $f(x) = x - 2e^{-x}$ by:

(a) Using the bisection method. Start with $a = 0$ and $b = 1$, and carry out the first three iterations.

Order	a	f(a)	b	f(b)	x_{NS}	$f(x_{NS})$	subinterval
1st loop	0	-2	1	0.26424	0.5	-0.71306	x_{NS1} and b
2nd loop	0.5	-0.71306	1	0.26424	0.75	-0.19473	x_{NS2} and b
3rd loop	0.75	-0.19473	1	0.26424	0.875	0.041276	a and x_{NS3}

(b) Using the secant method. Start with the two points, $x_1 = 0$ and $x_2 = 1$, and carry out the first three iterations.

$$x_3 = x_2 - \frac{f(x_2)(x_1 - x_2)}{f(x_1) - f(x_2)} = 1 - \frac{0.26424 * (0-1)}{(-2) - 0.26424} = 1 - \frac{0.26424}{2.26424} = 1 - 0.11670 = 0.88330$$

$$x_4 = x_3 - \frac{f(x_3)(x_2 - x_3)}{f(x_2) - f(x_3)} = 0.88330 - \frac{0.056467 * (1 - 0.88330)}{0.26424 - 0.056467} = 0.88330 - \frac{0.0065897}{0.20777} = 0.88330 - 0.031716 = 0.85158$$

$$x_5 = x_4 - \frac{f(x_4)(x_3 - x_4)}{f(x_3) - f(x_4)} = 0.85158 - \frac{(-0.0019003) * (0.88330 - 0.85158)}{0.056467 - (-0.0019003)} = 0.85158 + \frac{0.000060278}{0.058367} = 0.85158 + 0.0010327 = 0.85261$$

(c) Using Newton's method. Start at $x_1 = 1$ and carry out the first three iterations.

$$f'(x) = 1 + 2e^{-x}$$

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} = 1 - \frac{f(1)}{f'(1)} = 1 - \frac{0.26424}{1.73576} = 1 - 0.15223 = 0.84777$$

$$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)} = 0.84777 - \frac{f(0.84777)}{f'(0.84777)} = 0.84777 - \frac{(-0.0089683)}{1.85674} = 0.84777 + 0.0048301 = 0.85260$$

$$x_4 = x_3 - \frac{f(x_3)}{f'(x_3)} = 0.85260 - \frac{f(0.85260)}{f'(0.85260)} = 0.85260 - \frac{(-0.000010193)}{1.85261} = 0.85260 + 0.0000055020 = 0.85260$$

3.4 The lateral surface area, S, of a cone is given by:

$$\mathbf{S} = \boldsymbol{\pi}\mathbf{r}\sqrt{\boldsymbol{r}^2 + \boldsymbol{h}^2}$$

Where $\mathbf{r}$ is the radius of the base and $\mathbf{h}$ is the height. Determine the radius of a cone that has a surface area of $\mathbf{1800\ m^2}$ and a height of $\mathbf{25\ m}$. Solve by using the fixed-point iteration method with $\mathbf{r} = \frac{\mathbf{S}}{\boldsymbol{\pi}\sqrt{\boldsymbol{r}^2+\boldsymbol{h}^2}}$ as the iteration function. Start with $\mathbf{r_0 = 17\ m}$ and calculate the first four iterations.

$$\mathrm{r}_{\mathrm{i}+1} = \frac{S}{\pi\sqrt{r_i^2 + h^2}}$$

$$\mathrm{r}_1 = \frac{S}{\pi\sqrt{r_0^2 + h^2}} = \frac{1800}{\pi\sqrt{17^2 + 25^2}} = 18.952$$

$$\mathrm{r}_2 = \frac{1800}{\pi\sqrt{r_1^2 + h^2}} = \frac{1800}{\pi\sqrt{18.952^2 + 25^2}} = 18.264$$

$$\mathrm{r}_3 = \frac{1800}{\pi\sqrt{r_2^2 + h^2}} = \frac{1800}{\pi\sqrt{18.264^2 + 25^2}} = 18.506$$

$$\mathrm{r}_4 = \frac{1800}{\pi\sqrt{r_3^2 + h^2}} = \frac{1800}{\pi\sqrt{18.506^2 + 25^2}} = 18.421$$

3.13 Solve the following system of nonlinear equations:

$$\mathbf{-2x^3 + 3y^2 + 42 = 0}$$

$$\mathbf{5x^2 + 3y^3 - 69 = 0}$$

(a) Use Newton's method. Start at $\mathbf{x = 1, y = 1}$, and carry out the first five iterations.

The nonlinear equations are given by:

$$\mathrm{f}_1(x, y) = -2x^3 + 3y^2 + 42 = 0$$

$$\mathrm{f}_2(x, y) = 5x^2 + 3y^3 - 69 = 0$$

The partial derivatives in the equations are given by:

$$\frac{\partial \mathrm{f}_1}{\partial \mathrm{x}} = -6x^2 \ \text{ and } \ \frac{\partial \mathrm{f}_1}{\partial \mathrm{y}} = 6y$$

$$\frac{\partial \mathrm{f}_2}{\partial \mathrm{x}} = 10x \ \ and \ \ \frac{\partial \mathrm{f}_2}{\partial \mathrm{y}} = 9y^2$$

The Jacobian is given by:

$$\mathrm{J}(\mathrm{f}_1, \mathrm{f}_2) = \det \begin{bmatrix} \frac{\partial \mathrm{f}_1}{\partial \mathrm{x}} & \frac{\partial \mathrm{f}_1}{\partial \mathrm{y}} \\ \frac{\partial \mathrm{f}_2}{\partial \mathrm{x}} & \frac{\partial \mathrm{f}_2}{\partial \mathrm{y}} \end{bmatrix} = \det \begin{bmatrix} -6\mathrm{x}^2 & 6\mathrm{y} \\ 10\mathrm{x} & 9\mathrm{y}^2 \end{bmatrix} = -54x^2y^2 - 60xy$$

The initial guess is $\mathrm{x}_0 = 1, y_0 = 1$. The iterations start. $\Delta\mathrm{x}$ and $\Delta\mathrm{y}$ are determined by:

$$\Delta\mathrm{x} = \frac{-f_1(x_i, y_i)\dfrac{\partial f_2}{\partial y} + f_2(x_i, y_i)\dfrac{\partial f_1}{\partial y}}{J(f_1(x_i, y_i), f_2(x_i, y_i))}$$

$$\Delta\mathrm{y} = \frac{-f_2(x_i, y_i)\dfrac{\partial f_1}{\partial x} + f_1(x_i, y_i)\dfrac{\partial f_2}{\partial x}}{J(f_1(x_i, y_i), f_2(x_i, y_i))}$$

$\mathrm{x}_{\mathrm{i+1}} = x_i + \Delta x$, and $\mathrm{y}_{\mathrm{i+1}} = y_i + \Delta y$ are determined.

$$\mathrm{x}_1 = x_0 + \Delta x = 1.0000 + 3.9298 = 4.9298, y_1 = y_0 + \Delta y = 1.0000 - 0.5614 = 0.4386$$

$$\mathrm{x}_2 = x_1 + \Delta x = 4.9298 - 1.0307 = 3.8992, y_2 = y_1 + \Delta y = 0.4386 + 5.2832 = 5.7218$$

$$\mathrm{x}_3 = x_2 + \Delta x = 3.8992 + 0.2060 = 4.1051, y_3 = y_2 + \Delta y = 5.7218 - 1.8694 = 3.8524$$

$$\mathrm{x}_4 = x_3 + \Delta x = 4.1051 - 0.4919 = 3.6132, y_4 = y_3 + \Delta y = 3.8524 - 1.1594 = 2.6930$$

$$\mathrm{x}_5 = x_4 + \Delta x = 3.6132 - 0.3601 = 3.2531, y_5 = y_4 + \Delta y = 2.6930 - 0.5605 = 2.1326$$

3.18 Determining the natural logarithm of a number p, $\ln p$, is the same as finding a solution to the equation $f(x) = e^x - p = 0$. Write a MATLAB user-defined function that determines the natural logarithm of a number by solving the equation using the bisection method. Name the function `X=Ln(p)`. The output argument `X` is the value of $\ln p$, and the input argument `p` is the number whose natural logarithm is determined. The program should include the following features:

*** The starting values of a and b are $a = e^0$ and $b = p$, respectively, if $b > e^1$, and $a = -1/p$ and $b = e^0$, respectively if $b < e^1$.**

*** The iterations should stop when the tolerance is smaller than 1×10^{-6}.**

*** The number of iterations should be limited to 100. If a solution is not obtained in 100 iterations, the function stops and displays an error message.**

*** If zero or negative number is entered for `p`, the program stops and displays an error message.**

Use the function `Ln` to determine the natural logarithm of (a) 510, (b) 1.35, (c) 1, and (d) -7.

```
#  2017-2 Mathematical Models for Engineering Problems and Differential Equations
#          Numerical Methods for Engineers and Scientists
#          Chapter 3 Problem 18
#          Written by Choe Hyeong Jin, Dept. of Computer Science, Univ. of Seoul

clear all
function Ln (p)
          if p <= 0
                    disp('Error: p is zero or a negative number.\n');
                    return;
          end
          F = @ (x) exp(x) - p;
          if p > exp(1)
                    a = exp(0); b = p;
          else
                    a = -1/p; b = exp(0);
          end
          imax = 100; tol = 0.000001;
          Fa = F(a); Fb = F(b);
          if Fa*Fb > 0
                    disp('Error: The function has the same sign at points a and b.');
          else
                    disp('iteration        a            b     (xNS) Solution    f(xNS)    Tolerance');
                    for i = 1:imax
                              xNS = (a + b)/2;
                              toli = (b - a)/2;
                              FxNS = F(xNS);
                              fprintf('%9i %11.6f %11.6f %11.6f %11.6f %11.6f\n', i, a, b, xNS, FxNS, toli);
                              if FxNS == 0
                                        fprintf('An exact solution x = %11.6f was found\n', xNS);
                                        break;
                              end
```

```
                        if toli < tol
                                break;
                        end
                        if i == imax
                                fprintf('Solution was not obtained in %i iterations\n', imax);
                                break;
                        end
                        if F(a)*FxNS < 0
                                b = xNS;
                        else
                                a = xNS;
                        end
                end
        end
endfunction
```

3.19 A new method for solving a nonlinear equation $f(x) = 0$ is proposed. The method is similar to the bisection method. The solution starts by finding an interval $[a, b]$ that brackets the solution. The first estimate of the solution is the midpoint between $x = a$ and $x = b$. Then the interval $[a, b]$ is divided into four equal sections. The section that contains the root is taken as the new interval for the next iteration.

Write a MATLAB user-defined function that solves a nonlinear equation with the proposed new method. Name the function `Xs=QuadSecRoot(Fun, a, b)`, where the output argument `Xs` is the solution. The input argument `Fun` is a name for the function that calculates f(x) for a given x(it is a dummy name for the function that is imported into `QuadSecRoot`), a and `b` are two points that bracket the root. The iteration should stop when the tolerance is smaller than $10^{-6}x_{NS}$(x_{NS} is the current estimate of the solution).

Use the user-defined `QuadSecRoot` function to solve the equations in Problems 3.2 and 3.3. For the initial values of a and b, take the values that are listed in part (a) of the problems.

```
#  2017-2 Mathematical Models for Engineering Problems and Differential Equations
#         Numerical Methods for Engineers and Scientists
#         Chapter 3 Problem 19
#         Written by Choe Hyeong Jin, Dept. of Computer Science, Univ. of Seoul

function Xs = QuadSecRoot(F, a, b)
         imax = 1000; tol = abs((b - a)/2);
         Fa = F(a); Fb = F(b);
         if Fa*Fb > 0
                  disp('Error: The function has the same sign at points a and b.');
         else
                  for i = 1:imax
                           xNS = [0, 0, 0, 0, 0];
                           FxNS = [0, 0, 0, 0, 0];
                           gap = (b-a)/4;
                           for j = 1:5
                                    xNS(j) = a + gap * (j-1);
                                    FxNS(j) = F(xNS(j));
                           end
                           tol = 0.000001*xNS(j);
                           toli = (b-a)/2;
                           if toli < tol
                                    break;
                           end
                           for j = 1:5
                                    if FxNS(j) == 0
                                             fprintf('An exact solution x = %11.6f was found\n',
xNS(j));
                                             Xs = xNS(j);
                                             return;
                                    end
                           end
                           for j = 2:5
                                    if FxNS(j-1) * FxNS(j) < 0
                                             a = xNS(j-1);
                                             b = xNS(j);
                                             break;
                                    end
                           end
                           if i == imax
```

```
                        fprintf('Solution was not obtained in %i iterations\n', imax);
                        break;
                    end
                end
                Xs = xNS(3);
            end
endfunction
```

4.15 Carry out the first three iterations of the solution of the following system of equations using the Gauss-Seidel iterative method. For the first guess of the solution, take the value of all the unknowns to be zero.

$$\mathbf{8x_1 + 2x_2 + 3x_3 = 51}$$

$$\mathbf{2x_1 + 5x_2 + x_3 = 23}$$

$$\mathbf{-3x_1 + x_2 + 6x_3 = 20}$$

First, the equations are written in an explicit form.

$$x_1 = [51 - (2x_2 + 3x_3)]/8$$

$$x_2 = [23 - (2x_1 + x_3)]/5$$

$$x_3 = [20 - (-3x_1 + x_2)]/6$$

As a starting point, the initial value of all the unknowns, $x_1^{(1)}, x_2^{(1)}, x_3^{(1)}$ is assumed to be zero. The first three iterations are calculated manually.

The second estimate of the solution $(k = 2)$ is calculated in the first iteration. The values that are substituted for x_i in the right-hand side of the equations are the most recent known values. This means that when the first equation is used to calculate $x_1^{(2)}$, all the x_i values are zero. Then, when the second equation is used to calculate $x_2^{(2)}$, the new value $x_1^{(2)}$ is substituted for x_1, but the older value $x_3^{(1)}$ is substituted for x_3 and so on:

$$x_1^{(2)} = \frac{[51 - (2 \cdot 0 + 3 \cdot 0)]}{8} = 6.375$$

$$x_2^{(2)} = \frac{[23 - (2 \cdot 6.375 + 0)]}{5} = 2.050$$

$$x_3^{(2)} = \frac{[20 - (-3 \cdot 6.375 + 2.050)]}{6} = 6.1792$$

The third estimate of the solution $(k = 3)$ is calculated in the second iteration:

$$x_1^{(3)} = \frac{[51 - (2 \cdot 2.050 + 3 \cdot 6.1792)]}{8} = 3.5453$$

$$x_2^{(3)} = \frac{[23 - (2 \cdot 3.5453 + 6.1792)]}{5} = 1.9460$$

$$x_3^{(3)} = \frac{[20 - (-3 \cdot 3.5453 + 1.9460)]}{6} = 4.7816$$

The fourth estimate of the solution $(k = 4)$ is calculated in the third iteration:

$$x_1^{(4)} = \frac{[51 - (2 \cdot 1.9460 + 3 \cdot 4.7816)]}{8} = 4.0954$$

$$x_2^{(4)} = \frac{[23 - (2 \cdot 4.0954 + 4.7816)]}{5} = 2.0055$$

$$x_3^{(4)} = \frac{[20 - (-3 \cdot 4.0954 + 2.0055)]}{6} = 5.0468$$

4.17 Find the condition number of the matrix in Problem 4.13 using the infinity norm.

$$\begin{bmatrix} \mathbf{10} & \mathbf{12} & \mathbf{0} \\ \mathbf{0} & \mathbf{2} & \mathbf{-0.5} \\ \mathbf{0.5} & \mathbf{1} & \mathbf{-2} \end{bmatrix}$$

$$[\mathrm{a}] = \begin{bmatrix} 10 & 12 & 0 \\ 0 & 2 & -0.5 \\ 0.5 & 1 & -2 \end{bmatrix}, [a]^{-1} = \begin{bmatrix} 10 & 12 & 0 \\ 0 & 2 & -0.5 \\ 0.5 & 1 & -2 \end{bmatrix}^{-1} = \begin{bmatrix} 0.0921053 & -0.6315789 & 0.1578947 \\ 0.0065789 & 0.5263158 & -0.1315789 \\ 0.0263158 & 0.1052632 & -0.5263158 \end{bmatrix}$$

The infinity norm of [a] is calculated by using the equation below:

$$\|[a]\|_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} |a_{ij}|$$

$$\|[a]\|_\infty = \max_{1 \le i \le 3} \sum_{j=1}^{3} |a_{ij}| = \max[|10| + |12| + |0|, |0| + |2| + |-0.5|, |0.5| + |1| + |-2|] = \max[22, 2.5, 3.5] = 22$$

$$\begin{aligned} \|[a]^{-1}\|_\infty &= \max_{1 \le i \le 3} \sum_{j=1}^{3} |a_{ij}^{-1}| \\ &= \max[|0.0921053| + |-0.6315789| + |0.1578947|, |0.0065789| + |0.5263158| \\ &+ |-0.1315789|, |0.0263158| + |0.1052632| + |-0.5263158|] = \max[0.88158, 0.66447, 0.65789] \\ &= 0.88158 \end{aligned}$$

The condition number of the matrix [a] is calculated by using Eq. (4.86):

$$\therefore \mathrm{Cond}[\mathrm{a}] = \|[a]\| \|[a]^{-1}\| = 22 \cdot 0.88158 = 19.395$$

4.19 Find the condition number of the matrix in Problem 4.13 using the 1-norm.

$$\begin{bmatrix} \mathbf{10} & \mathbf{12} & \mathbf{0} \\ \mathbf{0} & \mathbf{2} & \mathbf{-0.5} \\ \mathbf{0.5} & \mathbf{1} & \mathbf{-2} \end{bmatrix}$$

$$[\mathrm{a}] = \begin{bmatrix} 10 & 12 & 0 \\ 0 & 2 & -0.5 \\ 0.5 & 1 & -2 \end{bmatrix}, [a]^{-1} = \begin{bmatrix} 10 & 12 & 0 \\ 0 & 2 & -0.5 \\ 0.5 & 1 & -2 \end{bmatrix}^{-1} = \begin{bmatrix} 0.0921053 & -0.6315789 & 0.1578947 \\ 0.0065789 & 0.5263158 & -0.1315789 \\ 0.0263158 & 0.1052632 & -0.5263158 \end{bmatrix}$$

The 1-norm of [a] is calculated by using the equation below:

$$\|[a]\|_1 = \max_{1 \le j \le n} \sum_{i=1}^{n} |a_{ij}|$$

$$\|[a]\|_1 = \max_{1 \le j \le 3} \sum_{i=1}^{3} |a_{ij}| = \max[|10| + |0| + |0.5|, |12| + |2| + |1|, |0| + |-0.5| + |-2|] = \max[10.5, 15, 2.5] = 15$$

$$\begin{aligned} \|[a]^{-1}\|_1 = \max_{1 \le j \le 3} \sum_{i=1}^{3} |a_{ij}^{-1}| \\ &= \max[|0.0921053| + |0.0065789| + |0.0263158|, |-0.6315789| + |0.5263158| \\ &+ |0.1052632|, |0.1578947| + |-0.1315789| + |-0.5263158|] = \max[0.12500, 1.2632, 0.81579] \\ &= 1.2632 \end{aligned}$$

The condition number of the matrix [a] is calculated by using Eq. (4.86):

$$\therefore \mathrm{Cond}[\mathrm{a}] = \|[a]\| \|[a]^{-1}\| = 15 \cdot 1.2632 = 18.948$$

4.27 Write a user-defined MATLAB function that calculates the condition number of an $(\mathbf{n} \times \mathbf{n})$ matrix by using the 1-norm. For the function name and arguments use `c = CondNumb_One(A)`, where `A` is the matrix and `c` is the value of the condition number. Within the function, use the user-defined functions `Inverse` from Problem 4.24 and `OneNorm` from Problem 4.25. Use the function `CondNumb_One` for calculating the condition number of the matrices in Problem 4.25.

(a) The matrix $A = \begin{bmatrix} -2 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -1.5 \end{bmatrix}$, (b) The matrix $B = \begin{bmatrix} 4 & -1 & 0 & 1 & 0 \\ -1 & 4 & -1 & 0 & 1 \\ 0 & -1 & 4 & -1 & 0 \\ 1 & 0 & -1 & 4 & -1 \\ 0 & 1 & 0 & -1 & 4 \end{bmatrix}$.

```
#  2017-2 Mathematical Models for Engineering Problems and Differential Equations
#       Numerical Methods for Engineers and Scientists
#       Chapter 4 Problem 27
#       Written by Choe Hyeong Jin, Dept. of Computer Science, Univ. of Seoul

function c = CondNumb_One(A)
        c = norm(A, 1) * norm(inv(A), 1);
endfunction
```