```
c
c     Main Program to Solve System of Non-Linear Equations by Newton-Raphson Method
c
      PARAMETER (N = 2)
      REAL UNKNON_G(N),UNKNON(N)
      OPEN(11,FILE='output.txt')
c
c     Provide an initial guess UNKNON
      UNKNON_G(1) = 1.5
      UNKNON_G(2) = 3.5
c
c     call Newton to solve the non-linear equations iteratively
      call Newton(N,UNKNON_G,UNKNON)
c
c     write the solution to newton.txt
  11  do i = 1, N
        write(11,'(e20.10)') UNKNON(i)
      enddo
      end
c
c
c
c
c
c
c
c     Subroutine Newton to to solve a system of algebraic equations iteratively
      subroutine Newton(N,UNKNON_G,UNKNON)
      REAL A(N,N),UNKNON_G(N),UNKNON(N),B(N)
      integer ipvt(N)
c
      itmax = 10
      tol   = 1.e-7
c     initialize matrix A and vector B
      do i = 1, N
      UNKNON(i) = UNKNON_G(i)
      B(i) = 0.0
      do j = 1, N
         A(i,j)=0.0
      enddo
      enddo
c
c     Start Iterative Solution
      iter = 0
  10  iter = iter + 1
      if (iter .gt. itmax) go to 12
c
c     Input system of nonlinear equations
c     on return B contains the residuals
      call residuals(N,UNKNON,B)
c
c     Check the norm of the residuals
      rnorm = 0.0
      do i = 1, N
      rnorm = rnorm + B(i)**2
      enddo
      rnorm = sqrt(rnorm)
      write(*,'(3x,a,i3,5x,a,e20.10)')
     !                    'iteration #',iter, 'Tolerance =', rnorm
      if (rnorm .lt. tol) return
c
c     Evaluate the Jacobian Matrix
c     IFLAG = 0 ==> analatical jacobian
c     IFLAG = 1 ==> numerical jacobian
      IFLAG = 0
      call jacobian(IFLAG,N,UNKNON,A)
c
c     LU-decompose matrix A
c     Note that matrix A will contains on return the L and U matrixes (original is destroyed)
      call ludcmp(A,N,N,ipvt,d)
c
c     Solve the equations: A .(Xiter+1-Xiter) =-B, by backsubstitution
```

```
c     Note that vector B will contains the unknowns X (original is destroyed)
c     for Newton Raphson method B = Xiter+1 - Xiter
      do i = 1, N
        B(i) = -B(i)
      enddo
c
      call lubksb(A,N,N,ipvt,B)
c
      do i= 1, N
        UNKNON(i) = UNKNON(i)+B(i)
      enddo
      go to 10
  12  write(11,*) 'No Convergence !'
      write(*,*)  'No Convergence !'
      stop
      return
      end
c
c
c
c
c
c
c
c
c
c     Subroutine Residuals to Input System of Non-Linear Equations
      subroutine residuals(N,UNKNON,R)
      REAL UNKNON(N),R(N)
c
      do i = 1, N
      R(i) = 0.0
      enddo
c
      R(1) = UNKNON(1)**2+UNKNON(1)*UNKNON(2)-10.0
      R(2) = UNKNON(2)+3.0*UNKNON(1)*UNKNON(2)**2-57.0
c
      return
      end
c
c
c
c
c
c
c
c
c     Subroutine jacobian to Evaluate the Jacobian Matrix
      subroutine jacobian(IFLAG,N,UNKNON,A)
      REAL A(N,N),UNKNON(N),UNKNONJ(N),FUN(N),FUNJ(N)
      parameter (epsl=1.e-5)
c
      do i = 1, N
      do j = 1, N
        A(i,j)=0.0
      enddo
      enddo
c
      if (IFLAG .gt. 0) go to 10
c
c     Evaluate the jacobian matrix analatically
      A(1,1) = 2.0*UNKNON(1)+UNKNON(2)
      A(1,2) = UNKNON(1)
      A(2,1) = 3.0*UNKNON(2)**2
      A(2,2) = 1.0+6.0*UNKNON(1)*UNKNON(2)
      return
c
c     Evaluate the jacobian matrix numerically
 10   do i = 1, N
        UNKNONJ(i)=UNKNON(i)
      enddo
      call residuals(N,UNKNON,FUN)
```

```
      do i = 1, N
        diff  = max1(epsl,abs(epsl*UNKNON(i)))
        UNKNONJ(i) = UNKNON(i)+diff
        call residuals(N,UNKNONJ,FUNJ)
        do j = 1, N
          A(j,i) = (FUNJ(j)-FUN(j))/diff
        enddo
        UNKNONJ(i)=UNKNON(i)
      enddo
      return
      end
c
      SUBROUTINE ludcmp(a,n,np,indx,d)
      INTEGER n,np,indx(n),NMAX
      REAL d,a(np,np),TINY
      PARAMETER (NMAX=500,TINY=1.0e-20)
      INTEGER i,imax,j,k
      REAL aamax,dum,sum,vv(NMAX)
      d=1.
      do 12 i=1,n
        aamax=0.
        do 11 j=1,n
          if (abs(a(i,j)).gt.aamax) aamax=abs(a(i,j))
11      continue
        if (aamax.eq.0.) pause 'singular matrix in ludcmp'
        vv(i)=1./aamax
12    continue
      do 19 j=1,n
        do 14 i=1,j-1
          sum=a(i,j)
          do 13 k=1,i-1
            sum=sum-a(i,k)*a(k,j)
13        continue
          a(i,j)=sum
14      continue
        aamax=0.
        do 16 i=j,n

          sum=a(i,j)
          do 15 k=1,j-1
            sum=sum-a(i,k)*a(k,j)
15        continue
          a(i,j)=sum
          dum=vv(i)*abs(sum)
          if (dum.ge.aamax) then
            imax=i
            aamax=dum
          endif
16      continue
        if (j.ne.imax)then
          do 17 k=1,n
            dum=a(imax,k)
            a(imax,k)=a(j,k)
            a(j,k)=dum
17        continue
          d=-d
          vv(imax)=vv(j)
        endif
        indx(j)=imax
        if(a(j,j).eq.0.)a(j,j)=TINY
        if(j.ne.n)then
          dum=1./a(j,j)

          do 18 i=j+1,n
            a(i,j)=a(i,j)*dum
18        continue
        endif
19    continue
      return
      END
c
      SUBROUTINE lubksb(a,n,np,indx,b)
```

```
      INTEGER n,np,indx(n)
      REAL a(np,np),b(n)
      INTEGER i,ii,j,ll
      REAL sum
      ii=0
      do 12 i=1,n
        ll=indx(i)
        sum=b(ll)
        b(ll)=b(i)
        if (ii.ne.0)then
          do 11 j=ii,i-1
            sum=sum-a(i,j)*b(j)
11        continue
        else if (sum.ne.0.) then
          ii=i
        endif
        b(i)=sum
12    continue
      do 14 i=n,1,-1
        sum=b(i)
        do 13 j=i+1,n
          sum=sum-a(i,j)*b(j)
13      continue
        b(i)=sum/a(i,i)
14    continue
      return
      END
```