```fortran
!  LamTube.f90
!
!
!
!*****************************************************************************
!
!  PROGRAM: CompTube
!
!  AUTHOR: T.H. Fronk
!
!  DATE: November 2011
!
!  PURPOSE:  Calculate Stresses or Strains For a Laminated Composite Tube
!
!*****************************************************************************
!   Modules
!
      Module Var
       implicit none
!
!      Variables
!
      type::lamina
         integer :: mat_type
         real (kind=kind(0.d0)):: thick
         real (kind=kind(0.d0)):: theta
      end type lamina

!
        integer :: i,j,k,l,n, Pflag, Tflag,  nl, nm, nrank, Smear_Flag,
L_Flag
        integer, allocatable, dimension (:) :: TIflag
        integer, parameter :: MaxLam=20
        character (len=70) :: title
      real (kind=kind(0.d0)) :: pi,sn, cs, sn2, cs2, sn4, cs4, Pin,
Pout,h,Ri,Ro
      real (kind=kind(0.d0)) ::  Num1, Num2, Num3, Num4, Num5, Num6
      real (kind=kind(0.d0)) :: epsx, Px, T, gammaxt
      real (kind=kind(0.d0)) :: Den, Den1, Den2, Den3, deltaT, wri, TT
      real (kind=kind(0.d0)) :: TItest, A1, A2
        real (kind=kind(0.d0)), allocatable, dimension(:,:) :: C
        real (kind=kind(0.d0)), allocatable, dimension(:,:,:) :: CB
        real (kind=kind(0.d0)), allocatable, dimension(:,:) :: alpha
        real (kind=kind(0.d0)), allocatable, dimension(:) :: thetar, Lamda,
Omega, Psi
        real (kind=kind(0.d0)), allocatable, dimension(:) :: Gamma, con1,
con3, con6
        real (kind=kind(0.d0)), allocatable, dimension(:) :: Sigmahat
        real (kind=kind(0.d0)), allocatable, dimension (:):: x, b, ELF, R,
rho, bb
      real (kind=kind(0.d0)), allocatable, dimension(:,:) :: a, KM, aa
      type (lamina), allocatable, dimension(:) :: lam
       real (kind=kind(0.d0)), allocatable, dimension(:,:) :: epsr, epst,
gamxt
         real (kind=kind(0.d0)), allocatable, dimension(:,:) :: sigx, sigt,
sigr, tauxt
         real (kind=kind(0.d0)) :: CBalpha6
```

```fortran
      real (kind=kind(0.d0)) :: Area, alphabarx, alphabarr, Nubarxt
      real (kind=kind(0.d0)) :: zetaPG, zetaTE, zetaPi, zetaDELT
      real (kind=kind(0.d0)) :: Ebarx, Gbarxt
     End module var

   Module PLib
   implicit none
     integer, parameter :: max_mats=30
     integer, dimension(max_mats) :: mat_id
     type :: materials
    character (len=32) :: description
    real (kind=kind(0.d0)) :: e1
      real (kind=kind(0.d0)) :: e2
      real (kind=kind(0.d0)) :: e3
      real (kind=kind(0.d0)) :: g12
      real (kind=kind(0.d0)) :: g13
      real (kind=kind(0.d0)) :: g23
      real (kind=kind(0.d0)) :: pr12
      real (kind=kind(0.d0)) :: pr21
      real (kind=kind(0.d0)) :: pr23
      real (kind=kind(0.d0)) :: pr32
      real (kind=kind(0.d0)) :: pr13
      real (kind=kind(0.d0)) :: pr31
      real (kind=kind(0.d0)) :: alpha1
      real (kind=kind(0.d0)) :: alpha2
     end type materials
     type (materials), dimension(max_mats) :: mat
     END MODULE PLib
!
!
!
     program LamTube
     use var
     USE PLib
     implicit none
!
!         Body of LamTube
!
   open(unit=7,file='LamTubeIn.txt',status='old')
   open(unit=8,file='LamTubeOut.txt')
   open(unit=9,file='Kmatrix.txt')
     pi=acos(-1.d0)

     CALL INPUT
     CALL KMATRIX
     WRITE(9,12)
     DO i=1,nrank
      DO j=1,nrank
       WRITE(9,11)i,j,KM(i,j)
      END DO
     END DO
 11   FORMAT(' ',1x,I2,4x,I2,2x,E13.6)
 12   FORMAT(' ','Row',2x,'Column',7x,'K(i,j)')
!
!
!
   IF(L_Flag.ne.0)THEN
```

```fortran
         CALL LOADS
         CALL SYMSOL
         If(Pflag.eq.1)then
          epsx=x(2*nl+1)
         ELSE
          Px=x(2*nl+1)
         end if
         If(Tflag.eq.1)then
          gammaxt=x(2*nl+2)
         else
          T=x(2*nl+2)
         end if
         wri=x(1)*r(0)**Lamda(1)+x(2)*r(0)**(-Lamda(1))+epsx*Gamma(1)*r(0)&
         +Omega(1)*gammaxt*r(0)**2.d0+Psi(1)*r(0)*deltaT
         CALL EPSSIG
         CALL OUTPUT
        END IF
        IF (Smear_Flag.ne.0)THEN
         CALL SMEAR
        END IF
        END
!
!
!=============================================================================
!
!     SUBROUTINES
!
!=============================================================================
        SUBROUTINE INPUT
        USE VAR
        USE PLib
        IMPLICIT NONE
        read(7,'(a70)')title
        read(7,*)nl,nm
        READ(7,*)(mat_id(i),i=1,nm)
!
!     Allocate Matrices
!

        nrank=2*nl+2
        if(nl.gt.MaxLam)nl=MaxLam
        Allocate (lam(nl))
        Allocate (thetar(nl))
        Allocate (alpha(nl,6))
        Allocate (x(2*nl+2))
        Allocate (b(2*nl+2),bb(2*nl+2))
        Allocate (elf(2*nl+2))
        Allocate (Rho(2*nl+2))
        Allocate (a(2*nl+2,2*nl+2),aa(2*nl+2,2*nl+2))
        Allocate (KM(2*nl+2,2*nl+2))
        Allocate (C(nm,9))
        Allocate (CB(nl,6,6))
        Allocate (r(0:nl))
        Allocate (TIflag(nl))
        Allocate (Psi(nl))
        Allocate (Omega(nl))
        Allocate (Lamda(nl))
```

```
        Allocate (Sigmahat(nl))
        Allocate (Gamma(nl))
        Allocate (con1(nl))
        Allocate (con3(nl))
        Allocate (con6(nl))
        Allocate (epsr(nl,3),epst(nl,3),gamxt(nl,3))
        Allocate (sigx(nl,3),sigt(nl,3),sigr(nl,3),tauxt(nl,3))
!
!
!
      DO i=1,nm
       CALL PropLib(i)
      END DO
!
!
!
      alpha=0.d0
      tt=0.d0
       do i=1,nl
         read(7,*)lam(i)%mat_type,lam(i)%thick,lam(i)%theta
         thetar(i)=lam(i)%theta*acos(-1.d0)/180.d0
         tt=tt+lam(i)%thick
       end do
       do i=1,nl
         j=lam(i)%mat_type
         WRITE(6,*)i,j

alpha(i,1)=cos(thetar(i))**2*mat(j)%alpha1+sin(thetar(i))**2*mat(j)%alpha2

alpha(i,2)=sin(thetar(i))**2*mat(j)%alpha1+cos(thetar(i))**2*mat(j)%alpha2
       alpha(i,3)=mat(j)%alpha2
       alpha(i,6)=2.d0*cos(thetar(i))*sin(thetar(i))*(mat(j)%alpha1-
mat(j)%alpha2)
        end do

        CALL MATPROPS
        read(7,*)Ri
        r(0)=Ri
        do k=1,nl
         r(k)=r(k-1)+lam(k)%thick
        end do
        READ(7,*)L_Flag, Smear_Flag
!
!     L_flag = 1  Input Loads & Delta T
!     L_flag = 0  No Load Calculations
!     Smear_flag = 1 Calculate Smeared Properties
!     Smear_flag = 0 Do Not Calculate Smeared Properties
!
        IF(L_Flag.ne.0)THEN
        read(7,*)deltaT
!
!     Input Loads or Strains
!
!     Pflag = 1 = Input Px
!     Pflag = 0 = Input epsilonx
!     Tflag = 1 = Input T
!     Tflag = 0 = Input gammaxt
```

```
!
       Px=0.d0
         epsx=0.d0
         T=0.d0
         gammaxt=0.d0
         read(7,*)Pflag
         select case (Pflag)
       case(1)
        read(7,*)Px
       case(0)
        read(7,*)epsx
         end select
         read(7,*)Tflag
         select case (Tflag)
       case(1)
        read(7,*)T
       case(0)
        read(7,*)gammaxt
       end select
       read(7,*)Pin,Pout
         END IF
         close (7)
       END SUBROUTINE
!
!       Properties
!
       SUBROUTINE PropLib(i)
         USE PLib
         IMPLICIT NONE
         integer :: i
       SELECT CASE (mat_id(i))
         CASE(1)
!
!       Aluminum (1) Eng
!
        mat(i)%description='Aluminum Eng'
        mat(i)%E1=11.d06
          mat(i)%E2=11.d06
        mat(i)%pr12=0.33d0
        mat(i)%pr23=0.33d0
          mat(i)%g12=mat(i)%E1/(2.d0*(1.d0+mat(i)%pr12))
        mat(i)%alpha1=13.11d-06
          mat(i)%alpha2=13.11d-06
       CASE(11)
!
!       Aluminum (11) SI
!
        mat(i)%description='Aluminum SI'
        mat(i)%E1=69.d09
          mat(i)%E2=69.d09
        mat(i)%pr12=0.33d0
        mat(i)%pr23=0.33d0
          mat(i)%g12=mat(i)%E1/(2.d0*(1.d0+mat(i)%pr12))
        mat(i)%alpha1=23.4d-06
          mat(i)%alpha2=23.4d-06
       CASE(12)
         !
```

```
!       Aluminum (12) SI
!
        mat(i)%description='Aluminum SI 2'
        mat(i)%E1=72.4d09
          mat(i)%E2=72.4d09
        mat(i)%pr12=0.3d0
        mat(i)%pr23=0.3d0
          mat(i)%g12=27.846d09
        mat(i)%alpha1=22.5d-06
          mat(i)%alpha2=22.5d-06


       CASE(4)
!
!       Hyer Graphite (4) SI
!
        mat(i)%description='Hyer Graphite SI'
        mat(i)%E1=155.d09
          mat(i)%E2=12.0D09
        mat(i)%pr12=0.248d0
        mat(i)%pr23=0.458d0
          mat(i)%g12=4.4d09
        mat(i)%alpha1=-0.018d-06
          mat(i)%alpha2=24.3d-06
       CASE(15)
!
!       T300/5208 (15) Eng
!
         mat(i)%description='T300/5208 ENG'
         mat(i)%E1=19.2d06
           mat(i)%E2=1.56d06
         mat(i)%pr12=0.24d0
         mat(i)%pr23=0.59d0
           mat(i)%g12=0.82d06
         mat(i)%alpha1=-.43d-06
           mat(i)%alpha2=13.6d-06
       END SELECT

         mat(i)%pr21=(mat(i)%pr12*mat(i)%E2)/mat(i)%E1
       mat(i)%G23=mat(i)%E2/(2.d0*(1.d0+mat(i)%pr23))
       mat(i)%E3=mat(i)%E2
         mat(i)%pr32=mat(i)%pr23
       mat(i)%pr13=mat(i)%pr12
         mat(i)%pr31=mat(i)%pr21
       mat(i)%G13=mat(i)%G12
!
!
!
       RETURN
         END
!
!      STIFFNESS AND COMPLIANCE MATRICES
!
       SUBROUTINE MATPROPS
         Use var
         USE PLib
         Implicit None
```

```fortran
      do i=1,nm
       DEN=(1.d0-2.d0*mat(i)%pr12*mat(i)%pr21-mat(i)%pr23*mat(i)%pr32-
2.d0*mat(i)%pr21*&
         mat(i)%pr32*mat(i)%pr13)/(mat(i)%E1*mat(i)%E2*mat(i)%E3)
         DEN1=mat(i)%E2*mat(i)%E3*DEN
         DEN2=mat(i)%E1*mat(i)%E3*DEN
         DEN3=mat(i)%E1*mat(i)%E2*DEN
         Num1=1.d0-mat(i)%pr23*mat(i)%pr32
         Num2=mat(i)%pr21+mat(i)%pr23*mat(i)%pr31
         Num3=mat(i)%pr31+mat(i)%pr21*mat(i)%pr32
       Num4=1.d0-mat(i)%pr13*mat(i)%pr31
         Num5=mat(i)%pr32+mat(i)%pr12*mat(i)%pr31
       Num6=1.d0-mat(i)%pr12*mat(i)%pr21
         C(i,1)=Num1/Den1
         C(i,2)=Num2/Den1
         C(i,3)=Num3/Den1
         C(i,4)=Num4/Den2
         C(i,5)=Num5/Den2
         C(i,6)=Num6/Den3
         C(i,7)=mat(i)%G23
         C(i,8)=mat(i)%G13
         C(i,9)=mat(i)%G12
      end do
      CB=0.d0
      do k=1,nl
       cs=cos(thetar(k))
         sn=sin(thetar(k))
         i=lam(k)%mat_type
       cs4=cs**4
       sn4=sn**4
         cs2=cs**2
         sn2=sn**2
         CB(K,1,1)=cs4*C(i,1)+2.d0*cs2*sn2*(C(i,2)+2.d0*C(i,9))+sn4*C(i,4)
         CB(K,1,2)=cs2*sn2*(C(i,1)+C(i,4)-4.d0*C(i,9))+(sn4+cs4)*C(i,2)
         CB(K,1,3)=cs2*C(i,3)+sn2*C(i,5)
         CB(K,1,6)=sn*cs*(cs2*(C(i,1)-C(i,2)-2.d0*C(i,9))+sn2*(C(i,2)-
C(i,4)+2.d0*C(i,9)))
         CB(K,2,2)=sn4*C(i,1)+2.d0*cs2*sn2*(C(i,2)+2.d0*C(i,9))+cs4*C(i,4)
       CB(K,2,3)=sn2*C(i,3)+cs2*C(i,5)
         CB(K,2,6)=sn*cs*(sn2*(C(i,1)-C(i,2)-2.d0*C(i,9))+cs2*(C(i,2)-
C(i,4)+2.d0*C(i,9)))
         CB(K,3,3)=C(i,6)
         CB(K,3,6)=cs*sn*(C(i,3)-C(i,5))
         CB(K,4,4)=cs2*C(i,7)+sn2*C(i,8)
         CB(K,4,5)=sn*cs*(C(i,8)-C(i,7))
         CB(K,5,5)=sn2*C(i,7)+cs2*C(i,8)
         CB(K,6,6)=sn2*cs2*(C(i,1)-2.d0*C(i,2)+C(i,4))+C(i,9)*(sn2-cs2)**2
       DO i=1,5
        DO j=i+1,6
           CB(k,j,i)=CB(k,i,j)
        end do
       end do
        end do
!
!    Calculate lamina constants
!
```

```
      Do k=1,nl
         Lamda(k)=sqrt(CB(K,2,2)/CB(K,3,3))
         TItest=dabs(CB(K,3,3)-CB(K,2,2))
       If(TItest.lt.1.0D-08)then
          TIFlag(k)=1
          Gamma(k)=0.d0
          Omega(k)=0.d0
          Sigmahat(k)=0.d0
          Psi(k)=0.d0
       else
          TIFlag(k)=0
          Gamma(k)=(CB(K,1,2)-CB(K,1,3))/(CB(K,3,3)-CB(K,2,2))
        Omega(k)=(CB(K,2,6)-2.d0*CB(K,3,6))/(4.d0*CB(K,3,3)-CB(K,2,2))
          Sigmahat(k)=0.d0
          do i=1,6
           Sigmahat(k)=Sigmahat(k)+(CB(K,i,3)-CB(K,i,2))*alpha(k,i)
          end do
          Psi(k)=Sigmahat(k)/(CB(K,3,3)-CB(K,2,2))
       End IF
      end do
        RETURN
        END
!.............................................................................
..
!
!
!   Assemble Coefficient Matrix
!
!
      SUBROUTINE KMATRIX
        USE var
        implicit none
        KM=0.d0
!
!    Find Preliminary KM Matrix and Vector ELF
!
!
!       Row 1
        KM(1,1)=(CB(1,2,3)+Lamda(1)*CB(1,3,3))*ri**(Lamda(1)-1.d0)
      KM(1,2)=(CB(1,2,3)-Lamda(1)*CB(1,3,3))*ri**(-Lamda(1)-1.d0)
      KM(1,2*nl+1)=CB(1,1,3)+(CB(1,2,3)+CB(1,3,3))*Gamma(1)
        KM(1,2*nl+2)=ri*((CB(1,2,3)+2.d0*CB(1,3,3))*Omega(1)+CB(1,3,6))
!     Even Rows
      do k=1,nl-1
       KM(2*k,2*k-1)=r(k)**Lamda(k)
       KM(2*k,2*k)=r(k)**(-Lamda(k))
         KM(2*k,2*k+1)=-r(k)**Lamda(k+1)
         KM(2*k,2*k+2)=-r(k)**(-Lamda(k+1))
         KM(2*k,2*nl+1)=(Gamma(k)-Gamma(k+1))*r(k)
         KM(2*k,2*nl+2)=(Omega(k)-Omega(k+1))*r(k)**2
      end do
!     Odd Rows
      do k=1,nl-1
       KM(2*k+1,2*k-1)=(CB(k,2,3)+Lamda(k)*CB(k,3,3))*r(k)**(Lamda(k)-1)
       KM(2*k+1,2*k)=(CB(k,2,3)-Lamda(k)*CB(k,3,3))*r(k)**(-Lamda(k)-1)
         KM(2*k+1,2*k+1)=-
(CB(k+1,2,3)+Lamda(k+1)*CB(k+1,3,3))*r(k)**(Lamda(k+1)-1)
```

```
          KM(2*k+1,2*k+2)=-(CB(k+1,2,3)-Lamda(k+1)*CB(k+1,3,3))*r(k)**(-
Lamda(k+1)-1)
          KM(2*k+1,2*nl+1)=CB(k,1,3)+Gamma(k)*(CB(k,2,3)+CB(k,3,3))&
                         -(CB(k+1,1,3)+Gamma(k+1)*(CB(k+1,2,3)+CB(k+1,3,3)))
          KM(2*k+1,2*nl+2)=((CB(k,2,3)+2.d0*CB(k,3,3))*Omega(k)+CB(k,3,6)&
                      -
((CB(k+1,2,3)+2.d0*CB(k+1,3,3))*Omega(k+1)+CB(k+1,3,6)))*r(k)
      end do
!     2N Equation
      KM(2*nl,2*nl-1)=(CB(nl,2,3)+Lamda(nl)*CB(nl,3,3))*r(nl)**(Lamda(nl)-
1.d0)
      KM(2*nl,2*nl)=(CB(nl,2,3)-Lamda(nl)*CB(nl,3,3))*r(nl)**(-Lamda(nl)-
1.d0)
        KM(2*nl,2*nl+1)=CB(nl,1,3)+Gamma(nl)*(CB(nl,2,3)+CB(nl,3,3))

KM(2*nl,2*nl+2)=((CB(nl,2,3)+2.d0*CB(nl,3,3))*Omega(nl)+CB(nl,3,6))*r(nl)
!     2N+1 Equation
      do k=1,nl
       KM(2*nl+1,2*k-1)=2.d0*pi*(CB(k,1,2)+Lamda(k)*CB(k,1,3))&
                         *(r(k)**(lamda(k)+1.d0)-r(k-
1)**(lamda(k)+1.d0))/(Lamda(k)+1.d0)
       IF(TIFlag(k).eq.1)then
          KM(2*nl+1,2*k)=0.d0
         else
        KM(2*nl+1,2*k)=2.d0*pi*(CB(k,1,2)-Lamda(k)*CB(k,1,3))&
                         *(r(k)**(-lamda(k)+1.d0)-r(k-1)**(-
lamda(k)+1.d0))/(-Lamda(k)+1.d0)
       END IF
       KM(2*nl+1,2*nl+1)=2.d0*pi*(CB(k,1,1)+Gamma(k)*(CB(k,1,3)+CB(k,1,2)))&
                         *(r(k)**2.d0-r(k-1)**2.d0)/2.d0+KM(2*nl+1,2*nl+1)

KM(2*nl+1,2*nl+2)=2.d0*pi*((CB(k,1,2)+2.d0*CB(k,1,3))*Omega(k)+CB(k,1,6))&
                         *(r(k)**3.d0-r(k-1)**3.d0)/3.d0+KM(2*nl+1,2*nl+2)
        end do

!     2N+2 Equation
      do k=1,nl
         KM(2*nl+2,2*k-1)=2.d0*pi*(CB(k,2,6)+Lamda(k)*CB(k,3,6))&
                         *(r(k)**(Lamda(k)+2.d0)-r(k-
1)**(Lamda(k)+2.d0))/(Lamda(k)+2.d0)
       KM(2*nl+2,2*k)=2.d0*pi*(CB(k,2,6)-Lamda(k)*CB(k,3,6))&
                         *(r(k)**(-Lamda(k)+2.d0)-r(k-1)**(-
Lamda(k)+2.d0))/(-Lamda(k)+2.d0)
       KM(2*nl+2,2*nl+1)= 2.d0*pi*(CB(k,1,6)+(CB(k,2,6)+CB(k,3,6))*Gamma(k))&
                         *(r(k)**3.d0-r(k-1)**3.d0)/3.d0+KM(2*nl+2,2*nl+1)
       KM(2*nl+2,2*nl+2)=
2.d0*pi*(CB(k,6,6)+(CB(k,2,6)+2.d0*CB(k,3,6))*Omega(k))&
                         *(r(k)**4.d0-r(k-1)**4.d0)/4.d0+KM(2*nl+2,2*nl+2)
      end do
        Con1=0.d0
        Con3=0.d0
        Con6=0.d0
        do k=1,nl
         DO i=1,6
          con1(k)=CB(k,i,1)*alpha(k,i)+con1(k)
          con3(k)=CB(k,i,3)*alpha(k,i)+con3(k)
          con6(k)=CB(k,i,6)*alpha(k,i)+con6(k)
```

```
        end do
       end do
!
!      End Forces
!
       Rho=0.d0
         Rho(1)=((CB(1,2,3)+CB(1,3,3))*Psi(1)-con3(1))
         Do k=1,nl-1
          Rho(2*k)=(Psi(k)-Psi(k+1))*r(k)
          Rho(2*k+1)=((CB(k,2,3)+CB(k,3,3))*Psi(k)-con3(k))&
                   -((CB(k+1,2,3)+CB(k+1,3,3))*Psi(k+1)-con3(k+1))
       End Do
         Rho(2*nl)=((CB(nl,2,3)+CB(nl,3,3))*Psi(nl)-con3(nl))
         Do k=1,nl
          Rho(2*nl+1)=((CB(k,1,2)+CB(k,1,3))*Psi(k)-con1(k))*&
                     pi*(r(k)**2-r(k-1)**2)+Rho(2*nl+1)
          Rho(2*nl+2)=((CB(k,2,6)+CB(k,3,6))*Psi(k)-con6(k))*&
                      2.d0*pi*(r(k)**3-r(k-1)**3)/3.d0+Rho(2*nl+2)
       End Do
        aa=Km
          bb=Elf

         RETURN
         END
!
!     Apply Loads or Strains and/or DELTA T
!
       SUBROUTINE LOADS
         USE var
         implicit none
!
!      Apply delta T and construct Force vector
!
         do i=1,2*nl+2
        elf(i)=-rho(i)*deltaT
         end do
         ELF(1)=elf(1)-Pin
       ELF(2*nl)=elf(2*nl)-Pout
       ELF(2*nl+1)=elf(2*nl+1)+Px
       ELF(2*nl+2)=elf(2*nl+2)+T
!
!    If Loads are given instead of strains the KM matrix and elf vector are
updated
!
       IF(Pflag.eq.0)then
        do i=1,2*nl+2
         ELF(i)=ELF(i)-KM(i,2*nl+1)*epsx
         KM(i,2*nl+1)=0.d0
        end do
          KM(2*nl+1,2*nl+1)=-1.d0
       END IF

         IF(Tflag.eq.0)then
        do i=1,2*nl+2
         ELF(i)=ELF(i)-KM(i,2*nl+2)*gammaxt
         KM(i,2*nl+2)=0.d0
        end do
```

```fortran
         KM(2*nl+2,2*nl+2)=-1.d0
      END IF
!
!
!
        a=Km
        b=Elf
        RETURN
        END
!
!......................................................................
..
!
!
!   Solve System of Equations
!
!
        SUBROUTINE SYMSOL
        use var
        implicit none
        integer :: ii, jj, i1
!       integer :: nrank
        integer :: mcol, icol
        integer, dimension(nrank) :: id
      real (kind=kind(0.d0)) :: amax
!
!     SCALING
!
        do i=1,nrank
         amax=dabs(a(i,1))
         do j=2,nrank
        if(dabs(a(i,j)).gt.amax)amax=DABS(a(i,j))
       end do
       DO j=1,nrank
      a(i,j)=a(i,j)/amax
     end do
     b(i)=b(i)/amax
    end do
!
!
!
       do i=1,nrank
      id(i)=i
     end do
!
!     PIVOTING
!
      mcol=nrank-1
      do j=1,mcol
       amax=a(id(j),j)
       icol=id(j)
       do i=j+1,nrank
        if(dabs(a(id(i),j)).GT.dabs(amax))then
         amax=a(id(i),j)
         id(j)=id(i)
         id(i)=icol
         icol=id(j)
```

```fortran
         end if
      end do
!
!       ELIMINATION
!
        i1=id(j)
        do  i=j+1,nrank
         ii=id(i)
         b(ii)=b(ii)-b(i1)*a(ii,j)/a(i1,j)
         do  jj=j+1,nrank
        a(ii,jj)=a(ii,jj)-a(ii,j)*a(i1,jj)/a(i1,j)
       end do
        end do
     end do
!
!      BACK SUBSTITUTION
!
       x(nrank)=b(id(nrank))/a(id(nrank),nrank)
       DO i=nrank-1,1,-1
        ii=id(i)
        x(i)=b(ii)/a(ii,i)
        DO j=i+1,nrank
       x(i)=x(i)-(a(ii,j)*x(j))/a(ii,i)
      end do
     end do
       RETURN
       END
!............................................................................
..
!
!
!  Find the Strains & Stresses
!
!
       SUBROUTINE EPSSIG
         USE Var
         Implicit NONE
         REAL (kind=kind(0.d0)) :: rmid
!
!       STRAINS
!
       epsr=0.d0
         epst=0.d0
         gamxt=0.d0
         do k=1,nl
           A1=x(2*k-1)
             A2=x(2*k)
             rmid=(r(k)+r(k-1))/2.d0
         epsr(k,1)=lamda(k)*A1*r(k-1)**(lamda(k)-1)-lamda(k)*A2*r(k-1)**(-
lamda(k)-1)&
                      +Gamma(k)*epsx+2.d0*Omega(k)*gammaxt*r(k-
1)+Psi(k)*deltaT
         epsr(k,2)=lamda(k)*A1*rmid**(lamda(k)-1)-lamda(k)*A2*rmid**(-
lamda(k)-1)&
                      +Gamma(k)*epsx+2.d0*Omega(k)*gammaxt*rmid+Psi(k)*deltaT
             epsr(k,3)=lamda(k)*A1*r(k)**(lamda(k)-1)-lamda(k)*A2*r(k)**(-
lamda(k)-1)&
```

```fortran
                    +Gamma(k)*epsx+2.d0*Omega(k)*gammaxt*r(k)+Psi(k)*deltaT
      epst(k,1)=A1*r(k-1)**(lamda(k)-1)+A2*r(k-1)**(-lamda(k)-1)&
                    +Gamma(k)*epsx+Omega(k)*gammaxt*r(k-1)+Psi(k)*deltaT
      epst(k,2)=A1*rmid**(lamda(k)-1)+A2*rmid**(-lamda(k)-1)&
                    +Gamma(k)*epsx+Omega(k)*gammaxt*rmid+Psi(k)*deltaT
          epst(k,3)=A1*r(k)**(lamda(k)-1)+A2*r(k)**(-lamda(k)-1)&
                    +Gamma(k)*epsx+Omega(k)*gammaxt*r(k)+Psi(k)*deltaT
      gamxt(k,1)=gammaxt*r(k-1)
          gamxt(k,2)=gammaxt*rmid
          gamxt(k,2)=gammaxt*r(k)
    end do
!
!   STRESSES
!
     sigx=0.d0
       sigt=0.d0
       sigr=0.d0
       tauxt=0.d0
       DO k=1,nl
        rmid=(r(k)+r(k-1))/2.d0
        sigx(k,1)=(epsx-alpha(k,1)*deltaT)*CB(k,1,1)+&
                (epst(k,1)-alpha(k,2)*deltaT)*CB(k,1,2)+&
                      (epsr(k,1)-alpha(k,3)*deltaT)*CB(k,1,3)+&
              (gamxt(k,1)-alpha(k,6)*deltaT)*CB(k,1,6)
      sigx(k,2)=(epsx-alpha(k,1)*deltaT)*CB(k,1,1)+&
                (epst(k,2)-alpha(k,2)*deltaT)*CB(k,1,2)+&
                      (epsr(k,2)-alpha(k,3)*deltaT)*CB(k,1,3)+&
              (gamxt(k,2)-alpha(k,6)*deltaT)*CB(k,1,6)
      sigx(k,3)=(epsx-alpha(k,1)*deltaT)*CB(k,1,1)+&
                (epst(k,3)-alpha(k,2)*deltaT)*CB(k,1,2)+&
                      (epsr(k,3)-alpha(k,3)*deltaT)*CB(k,1,3)+&
              (gamxt(k,3)-alpha(k,6)*deltaT)*CB(k,1,6)
        sigt(k,1)=(epsx-alpha(k,1)*deltaT)*CB(k,1,2)+&
                (epst(k,1)-alpha(k,2)*deltaT)*CB(k,2,2)+&
                      (epsr(k,1)-alpha(k,3)*deltaT)*CB(k,2,3)+&
              (gamxt(k,1)-alpha(k,6)*deltaT)*CB(k,2,6)
      sigt(k,2)=(epsx-alpha(k,1)*deltaT)*CB(k,1,2)+&
                (epst(k,2)-alpha(k,2)*deltaT)*CB(k,2,2)+&
                      (epsr(k,2)-alpha(k,3)*deltaT)*CB(k,2,3)+&
              (gamxt(k,2)-alpha(k,6)*deltaT)*CB(k,2,6)
      sigt(k,3)=(epsx-alpha(k,1)*deltaT)*CB(k,1,2)+&
                (epst(k,3)-alpha(k,2)*deltaT)*CB(k,2,2)+&
                      (epsr(k,3)-alpha(k,3)*deltaT)*CB(k,2,3)+&
              (gamxt(k,3)-alpha(k,6)*deltaT)*CB(k,2,6)
        sigr(k,1)=(epsx-alpha(k,1)*deltaT)*CB(k,1,3)+&
                (epst(k,1)-alpha(k,2)*deltaT)*CB(k,3,2)+&
                      (epsr(k,1)-alpha(k,3)*deltaT)*CB(k,3,3)+&
              (gamxt(k,1)-alpha(k,6)*deltaT)*CB(k,3,6)
      sigr(k,2)=(epsx-alpha(k,1)*deltaT)*CB(k,1,3)+&
                (epst(k,2)-alpha(k,2)*deltaT)*CB(k,3,2)+&
                      (epsr(k,2)-alpha(k,3)*deltaT)*CB(k,3,3)+&
              (gamxt(k,2)-alpha(k,6)*deltaT)*CB(k,3,6)
      sigr(k,3)=(epsx-alpha(k,1)*deltaT)*CB(k,1,3)+&
                (epst(k,3)-alpha(k,2)*deltaT)*CB(k,3,2)+&
                      (epsr(k,3)-alpha(k,3)*deltaT)*CB(k,3,3)+&
              (gamxt(k,3)-alpha(k,6)*deltaT)*CB(k,3,6)
              CBalpha6=0.d0
```

```
                        DO i=1,6
                         CBalpha6=CBalpha6+CB(k,i,6)*alpha(k,i)
                END DO
          tauxt(k,1)=epsx*(CB(k,1,6)+(CB(k,2,6)+CB(k,3,6))*Gamma(k))+&

gammaxt*(CB(k,6,6)+(CB(k,2,6)+2.d0*CB(k,3,6))*Omega(k))*r(k-1)+&
                 ((CB(k,2,6)+CB(k,3,6))*Psi(k)-CBalpha6)*deltaT+&
                 (CB(k,2,6)+CB(k,3,6)*lamda(k))*A1*r(k-1)**(lamda(k)-1.d0)+&
                         (CB(k,2,6)-lamda(k)*CB(k,3,6))*A2*r(k-1)**(-
lamda(k)-1.d0)
          tauxt(k,2)=epsx*(CB(k,1,6)+(CB(k,2,6)+CB(k,3,6))*Gamma(k))+&

gammaxt*(CB(k,6,6)+(CB(k,2,6)+2.d0*CB(k,3,6))*Omega(k))*rmid+&
                 ((CB(k,2,6)+CB(k,3,6))*Psi(k)-CBalpha6)*deltaT+&
                 (CB(k,2,6)+CB(k,3,6)*lamda(k))*A1*rmid**(lamda(k)-1.d0)+&
                         (CB(k,2,6)-lamda(k)*CB(k,3,6))*A2*rmid**(-lamda(k)-
1.d0)
            tauxt(k,3)=epsx*(CB(k,1,6)+(CB(k,2,6)+CB(k,3,6))*Gamma(k))+&

gammaxt*(CB(k,6,6)+(CB(k,2,6)+2.d0*CB(k,3,6))*Omega(k))*r(k)+&
                 ((CB(k,2,6)+CB(k,3,6))*Psi(k)-CBalpha6)*deltaT+&
                 (CB(k,2,6)+CB(k,3,6)*lamda(k))*A1*r(k)**(lamda(k)-1.d0)+&
                         (CB(k,2,6)-lamda(k)*CB(k,3,6))*A2*r(k)**(-lamda(k)-
1.d0)
      END DO
      return
        END
!-----------------------------------------------------------------------------
----
!
!     Find Laminated Tube Smeared Properties
!
!
      SUBROUTINE SMEAR
      USE var
        IMPLICIT NONE
        real (kind=kind(0.d0)) :: Jo
!
!      SMEARED PROPERTIES
!
        Area=pi*(r(nl)**2-Ri**2)
            Jo=0.5d0*pi*(r(nl)**4-Ri**4)
            Pflag=1
            Tflag=1
            Pin=0.d0
            Pout=0.d0
            Km=aa
          Elf=bb
!
!       CASE A  Px.ne.0
!
            Px=1.d0
            T=0.d0
        deltaT=0.d0
        CALL LOADS
            CALL SYMSOL
            epsx=x(2*nl+1)
```

```
            WRITE(6,*)'epsx=',epsx
            gammaxt=x(2*nl+2)
            wri=x(1)*r(0)**Lamda(1)+x(2)*r(0)**(-
Lamda(1))+epsx*Gamma(1)*r(0)&
              +Omega(1)*gammaxt*r(0)**2.d0+Psi(1)*r(0)*deltaT
        Ebarx=Px/(epsx*Area)
            WRITE(6,*)'eps=',epsx,area,px
            zetaPG=gammaxt*Ri/epsx !shear / axial eleongation due to AXIAL
Load, Px
        Nubarxt=-wri/(epsx*Ri)
!
!       CASE B Tx.ne.0
!
            Px=0.d0
            T=100.d0
        deltaT=0.d0
            CALL LOADS
            CALL SYMSOL
            epsx=x(2*nl+1)
            gammaxt=x(2*nl+2)
            wri=x(1)*r(0)**Lamda(1)+x(2)*r(0)**(-
Lamda(1))+epsx*Gamma(1)*r(0)&
              +Omega(1)*gammaxt*r(0)**2.d0+Psi(1)*r(0)*deltaT

            Gbarxt=T/(gammaxt*Jo)
            zetaTE=epsx/(gammaxt*Ri) !axial eleongation / shear due to Torque
!
!       CASE C  delta T .ne. 0
!
            Px=0.d0
            T=0.d0
        deltaT=100.d0
        CALL LOADS
            CALL SYMSOL
            epsx=x(2*nl+1)
            gammaxt=x(2*nl+2)
            wri=x(1)*r(0)**Lamda(1)+x(2)*r(0)**(-
Lamda(1))+epsx*Gamma(1)*r(0)&
              +Omega(1)*gammaxt*r(0)**2.d0+Psi(1)*r(0)*deltaT
            alphabarx=epsx/deltaT
        alphabarr=wri/(Ri*deltaT)
            zetaDELT=gammaxt*Ri/epsx !Shear / axial eleongation due to a
delta T
!
!           CASE D Pi .ne. 0
!
        Px=0.d0
            T=0.d0
            deltaT=0.d0
            Pin=10.d0
        CALL LOADS
            CALL SYMSOL
            epsx=x(2*nl+1)
            gammaxt=x(2*nl+2)
            wri=x(1)*r(0)**Lamda(1)+x(2)*r(0)**(-
Lamda(1))+epsx*Gamma(1)*r(0)&
              +Omega(1)*gammaxt*r(0)**2.d0+Psi(1)*r(0)*deltaT
```

```
        zetaPi=gammaxt*Ri/epsx !Shear / axial eleongation due to a Internal
Pressure, Pi
!
!
!
        WRITE(8,*)
        WRITE(8,*)
        WRITE(8,*)'SMEARED PROPERTIES'
        WRITE(8,*)
        write(8,32)Ebarx
        WRITE(8,33)Nubarxt
      write(8,34)Gbarxt
        WRITE(8,35)zetaPG
        WRITE(8,36)zetaTE
        WRITE(8,37)zetaDELT
        WRITE(8,38)zetaPi
        WRITE(8,39)alphabarx
        WRITE(8,40)alphabarr

 32   format(' ','Ex=',e13.6)
 33   format(' ','Nuxt=',e13.6)
 34   format(' ','Gxt=',e13.6)
 35   format(' ','zeta PG =',e13.6)
 36   format(' ','zeta TE =',e13.6)
 37   format(' ','zeta DELT=',e13.6)
 38   format(' ','zeta Pi =',e13.6)
 39   format(' ','alpha x =',e13.6)
 40   format(' ','alpha r =',e13.6)
      RETURN
        END
!-----------------------------------------------------------------------------
--
!
!     Write Output
!
!
      SUBROUTINE Output
        Use var
        Use Plib
        implicit none
        REAL (kind=kind(0.d0)) :: rmid
!
!      Echo Input
!
        write(8,*)title
        write(8,2)
        WRITE(8,*)'Laminate Stacking Sequence'
        WRITE(8,*)
        WRITE(8,32)
        DO k=1,nl
         write(8,33)lam(k)%mat_type,lam(k)%thick,lam(k)%theta
      end do
        write(8,2)
        write(8,*)'NRANK= ',nrank
        WRITE(8,36)Ri
      write(8,1)
        Do i=1,nm
```

```
        write(8,6)mat(i)%description
        write(8,7)mat(i)%e1, mat(i)%e2,  mat(i)%g12, mat(i)%pr12, mat(i)%pr23
          write(8,17)mat(i)%alpha1, mat(i)%alpha2
          write(8,1)
      end do
      Write(8,2)
      If(Pflag.eq.1)then
        Write(8,22)Px
      else
        Write(8,21)epsx
      end if
        If(Tflag.eq.1)then
        Write(8,24)T
      else
        Write(8,23)gammaxt
      end if
        write(8,25)Pin
        Write(8,26)Pout
        Write(8,27)deltaT
        write(8,2)
!
!
!
        Do i=1,nm
       Write(8,2)
         Write(8,*)'3D Stiffness Matrix C'
         Write(8,1)
         write(8,16)C(i,1),C(i,2),C(i,3),0.d0,0.d0,0.d0
         write(8,16)C(i,2),C(i,4),C(i,5),0.d0,0.d0,0.d0
         write(8,16)C(i,3),C(i,5),C(i,6),0.d0,0.d0,0.d0
         write(8,16)0.d0,0.d0,0.d0,C(i,7),0.d0,0.d0
         write(8,16)0.d0,0.d0,0.d0,0.d0,C(i,8),0.d0
         write(8,16)0.d0,0.d0,0.d0,0.d0,0.d0,C(i,9)
         Write(8,2)
      END DO
!
!
!
        DO k=1,nl
       Write(8,*)'Layer #',k
       Write(8,*)'C Bar Matrix'
       do i=1,6
        write(8,3)(CB(k,i,j),j=1,6)
       end do
         write(8,1)
         Write(8,*)'Off-axis CTE'
         do j=1,6
          Write(8,19)j,alpha(k,j)
       end do
         write(8,2)
      END DO
!
!
!
      Write(8,*)'Lamina Constants'
        WRITE(8,*)
        WRITE(8,34)
```

```
       Do k=1,nl
        WRITE(8,35)k,Lamda(k),Gamma(k),Omega(k),Sigmahat(k),Psi(k)
       END Do
         WRITE(8,2)
!
!
!
         Write(8,*)'Km Matrix'
!       Do i=1,10
!        Write(8,13)(KM(i,j),j=1,10)
!       end do
       Write(8,2)
       Write(8,*)'Rho & Elf Terms'
         write(8,15)
         Do i=1,2*nl+2
          Write(8,14)i,Rho(i),Elf(i)
       end do
         write(8,2)
!
!
!

        write(8,4)
        If(Pflag.eq.1)then
       Write(8,21)x(2*nl+1)
      else
         Write(8,22)x(2*nl+1)
      end if
        If(Tflag.eq.1)then
       Write(8,23)x(2*nl+2)
      else
       Write(8,24)x(2*nl+2)
      end if
        Write(8,2)
        write(8,*)'w(Ri)=',wri
!
!
!
         WRITE(8,2)
         WRITE(8,28)
         DO k=1,nl
          rmid=(r(k)+r(k-1))/2.d0
          WRITE(8,29)k,r(k-1),epsr(k,1),epst(k,1),gamxt(k,1)
          WRITE(8,29)k,rmid,epsr(k,2),epst(k,2),gamxt(k,2)
          WRITE(8,29)k,r(k),epsr(k,3),epst(k,3),gamxt(k,3)
       END DO
!
!
!
         WRITE(8,2)
         WRITE(8,30)
         DO k=1,nl
          rmid=(r(k)+r(k-1))/2.d0
          WRITE(8,31)k,r(k-1),sigx(k,1),sigt(k,1),sigr(k,1),tauxt(k,1)
          WRITE(8,31)k,rmid,sigx(k,2),sigt(k,2),sigr(k,2),tauxt(k,2)
        WRITE(8,31)k,r(k),sigx(k,3),sigt(k,3),sigr(k,3),tauxt(k,3)
       END DO
```

```
!
!
!
 1     format(' ',/)
 2     format(' ',//)
 3     format(' ',6(2x,e9.2))
 4     format(' ',5X,'SOLUTION')
 5     format(' ','A1= ',d13.6)
 6     format(' ','Material',3x,a32)
 7     format(' ','E1=',en11.2,8x,'E2=',en9.2,10x,'G12=',&
                  en11.2,8x,'PR12=',f6.4,4x,'PR23=',f6.4)
 8     format(' ','Lamda=',2x,E11.2)
 9     format(' ','Gamma=',E11.2)
 10    format(' ','Omega=',2x,E11.2)
 11    format(' ','Sigmahat=',2x,En11.2)
 12    format(' ','Psi=',2x,En11.2)
 13    format(' ','|',10(2x,e11.4),1x,'|')
 14    format(' ',2x,i2,4x,e11.4,4x,e11.4)
 15    format(' ','Index',9x,'R',13x,'Elf')
 16    format(' ','|',6(2x,e11.4),2x,'|')
 17    format(' ','alpha1= ',e11.2,4x,'alpha2= ',e11.2)
 18    format(' ','THETA= ',f8.4)
 19    format(' ',i1,2x,en13.2)
 20    format(' ','A2= ',e13.6)
 21    format(' ','epsx= ',e15.6)
 22    format(' ','Px= ',e13.6)
 23    format(' ','gammaxt=',en15.6)
 24    format(' ','T=',e13.6)
 25    format(' ','Pin=',e13.6)
 26    format(' ','Pout=',e13.6)
 27    format(' ','delta T=',e13.6)
 28    format(' ','Lamina',9x,'r',10x,'epsilon r',6x,'epsilon t',6x,'gamma
xt')
 29    format(' ',2x,i2,4x,e13.6,2x,e13.6,2x,e13.6,2x,e13.6)
 30    format(' ','Lamina',9x,'r',11x,'sigma x',8x,'sigma t',8x,'sigma
r',8x,'tau xt')
 31    format(' ',2x,i2,4x,e13.6,2x,e13.6,2x,e13.6,2x,e13.6,2x,e13.6)
 32    format(' ','Material Type',2x,'Lamina Thickness',2x,'Lamina Angle')
 33    format(' ', 5x,i2,9x,en13.6,4x,f8.4)
 34    format('
','Lamina',6x,'lamda',10x,'Gamma',10x,'Omega',9x,'Sigmahat',9x,'Psi')
 35    format(' ',2x,i2,2x,5(2x,e13.6))
 36    format(' ','Ri= ',en15.6)
       RETURN
       END
```