7.4.6 モード変位と固有周期ファイル

本節では、固有値解析で求めた固有値と固有ベクトルを、出力仕様に合わせて振動数、固有周期、振動モード、刺激係数をファイルに出力する。SPACE では、固有値解析においてサブスペース法とヤコビ法を使い分けており、各々予備計算を含めて解析方法が異なっている。ただし、出力ファイルの仕様は同じである。ここで出力するファイルは次の２つ、

```
7   'mode_d'        42          17      F     !  ﾓｰﾄﾞ変位
8   'omeg_d'        43          18      F     !  固有周期、振動数、減衰定数、刺激係数
```

であり、どちらも ASKII ファイルである。固有周期や刺激係数を保存するファイル omeg_d は、第 7.3.8 節で示されている。ここでは、ファイル mode_d について説明する。ファイルは ASKII ファイルであり、単純な構成となっている。計算モード次数がｎ次である場合、2*n 個のグループで出力されており、第１段階、第２段階の順に、同じ仕様で連続して出力されている。ここでは、第１階の出力仕様を用いて説明する。第１行目はコメントであり、モード番号が示されている。２行目以降は、同じ仕様で、左から、節点番号、次に当該の節点に対する６個の自由度に関する全体座標系におけるモード変位（$u,v,w,\theta_x,\theta_y,\theta_z$）である。

```
 ***** MODE  1 *****
1   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
2   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
3   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
4   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
5   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
6   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
7   0.00000   0.05368   0.00351  -0.00015   0.00000   0.00000
8   0.00000   0.05414  -0.00873   0.00005   0.00000   0.00000
```

さらに、プログラムの中で、出力ユニット番号が 76 のファイルは、固有値解析の結果を出力する EOUTPUT である。また、サブスペース法による結果の出力は、Out_Eigen()、ヤコビ法では、Out_Eigen_J()サブルーチンである。ヤコビ法では一般固有値問題から標準固有値問題への変換を行っており、得られた固有ベクトルは、Trans_Eigen_V()サブルーチンを用いて変換している。これらのサブルーチンは、固有値問題に関する主サブルーチンである submain_dynamic_b()から以下のようにコールされている。

```
      c―――――――――――――――――――――――――★結果の出力
      call Out_Eigen(n_step,n_unknown,Parameter_C,Eigen_d,
     *          Point,Newmark_P,Eigen_Value,Eigen_Vector,Omega,
```

```
     *           gskymm,max_h_sky,rot_local,disp_point_m,
     *           tw_a,tw_b,ifl(7),iflz(7),ifl(8),iflz(8))

c----------★固有ベクトルの変換
      call Trans_Eigen_V(Eigen_Vector,n_unknown,gskymm,vec_w2,ier)

c----------★結果の出力
      call Out_Eigen_J(n_step,n_unknown,Parameter_C,Eigen_d,
     *           Point,Newmark_P,Eigen_Value,Eigen_Vector,Omega,
     *           gskymm,rot_local,disp_point_m,
     *           vec_w3,vec_w4,ifl(7),iflz(7),ifl(8),iflz(8))
```

このサブルーチンでは出力仕様に従って、少し計算を行っており、その部分について解説する。ただし、Out_Eigen_J()のプログラムコードは、Out_Eigen()とほとんど同じであり、ここでは、説明、記述共に省略する。サブルーチン Out_Eigen_J()を見たい方は付録を参照されたい。それでは、以下の説明を参照しながら、プログラムを順次理解していこう。

１．モード変位を保存する配列 disp_point_m をゼロクリアする。
２．振動モードの最大値を検索し、最大値を１とする規準化を行う。ここで、n_mode は解析モード数であり、振動モードは配列 Eigen_Vector に保存されている。
３．振動モードの最大値を１にセットする。
４．振動数 Omega を固有値 Eigen_Value から計算する。
５．レーリー減衰等に関する係数計算、ここでは、SPACE の仕様に従って第１と第２の２段階で係数を求める。
６．現在の仕様では、1：質量比例型、2：剛性比例型、3：レーリー型の３種が選択可能であり、各々に従って係数を計算する。
７．ここでは、振動数、固有周期等のタイトルをファイル'omeg_d'に出力する。
８．次に、刺激係数を求める。この計算は多少長いプログラムコードが必要となるので、注意して読まれたい。
  8.1 これ以後の一連の計算を解析モード数繰り返す。
  8.2 固有ベクトルをワーク領域である配列 disp に保存する。
  8.3 ３方向の刺激係数をゼロにセットする。
  8.4 サブルーチン Multm()を用いて、質量行列と固有ベクトルの行列積を求め、得られたベクトルを配列 w にセットする。
  8.5 上記の配列 w と固有ベクトルのスカラー積を取り、刺激係数の分母 t_dmd を計算する。

8.6 刺激係数の分子を計算するために、上記の配列 w について方向別に和をとる。ただし、節点で釣合座標系（局所座標系）を用いている場合は、全体座標系に変換する必要があり、その処理を行った後、3 方向の和をとる。

8.7 最後に、分母である t_dmd がゼロでない場合は、3 方向の刺激係数を t_dmd で割ることによって、各々の刺激係数を求める。

９．得られた刺激係数をチェックするために、各固有ベクトルに対応する刺激係数を掛け、各節点の 3 方向毎に、全モードについて和を取る。無論、固有ベクトルが局所座標系を使用している場合は、全体座標系に座標変換する。全モードについてこの計算を行えば、その結果は各節点の 3 方向毎に全て１となるはずであるが、サブスペース法では、求められるモード数に制限があるため、全モードで処理することが不可能であり、１とはならない。ただし１に近い値が得られることになる。また、ヤコビ法では、全モードで求めるため、この値は１となる。この値は固有値解析や関連する処理が正確であることの検証となる。

10. 各振動数における減衰定数を、レーリー減衰などで定義した係数から求める。また、固有周期を計算する。

11. 得られた情報、つまりモード番号毎に、振動数、固有周期、減衰定数、3 方向の刺激係数をファイル'omeg_d'に出力する。

12. 振動モードをファイル'mode_d'に出力する。このとき、節点の座標が局所座標を使用している場合は、全体座標系に変換する。

13. 最後に、先に計算した刺激係数の検証結果を EOUTOUT ファイルに出力する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Out_Eigen
C     ―――――――――――――――――――――――――――――――――
C     ●    振動数と固有ベクトルの計算と出力
C     ―――――――――――――――――――――――――――――――――
      subroutine Out_Eigen(n_step,n_unknown,Parameter_C,Eigen_d,
     *           Point,Newmark_P,Eigen_Value,Eigen_Vector,Omega,
     *           gskymm,max_h_sky,rot_local,disp_point_m,
     *           disp,w,ifl1,iflz1,ifl2,iflz2)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record /eigen_d_s    / Eigen_d
      record /newmark_s    / Newmark_P
      record /Parameter_s  / Parameter_C
      record /point_s      / Point
      dimension  Point(*)
```

```
      dimension  Eigen_Value(*),Eigen_Vector(n_unknown,*)
      dimension  Omega(*),gskymm(*),max_h_sky(0:*),rot_local(3,3,*)
      dimension  disp(*),w(*),disp_point_m(3,3,*)      !ワーク領域
      dimension xx(3),yy(3),Shigeki(3),tdisp(6)

      n_point=Parameter_C.n_point                      ! 節点数
      n_local_coord = Parameter_C.n_local_coord
      n_mode = Eigen_d.n_modes                         ! モード数
      load_mass=Eigen_d.load_mass
      pi = 3.14159265D0
c―――――――――――――――――――――――★モード変位領域をゼロクリア
      do i=1,n_point                                                  ! 1
      do j=1,3
      do k=1,3
      disp_point_m(k,j,i)=0.
      enddo
      enddo
      enddo
c―――――――――――――――――――――――★振動モードの最大値検索
      do 10 imode=1,n_mode                                            ! 2
      xmax = 0.D0
      do i=1,n_unknown
      if (dabs(Eigen_Vector(i,imode)).gt.xmax)
     &           xmax = dabs(Eigen_Vector(i,imode))
      enddo
c―――――――――――――――――――――――★振動モードの最大を１にセット
      if(xmax.ne.0.) then                                             ! 3
      do i=1,n_unknown
      Eigen_Vector(i,imode) = Eigen_Vector(i,imode)/xmax
      enddo
      endif
c―――――――――――――――――――――――★振動数計算
      Omega(imode) = dsqrt(Eigen_Value(imode))                        ! 4
   10 enddo
c―――――――――――――――――――――――★レーリー減衰等に関する係数計算
      A0 = 0.0D0                                                      ! 5
      A1 = 0.0D0
      n_damp_1=Newmark_P.n_damp_1         ! 第１指定モード番号
      n_damp_2=Newmark_P.n_damp_2         ! 第２指定モード番号
      n_damp_type=Newmark_P.n_damp_type
c―――――――――――――――――――――――★ステップで減衰定数を変更
      if(n_step.eq.1) then
      h1=Newmark_P.alf1_1
      h2=Newmark_P.alf1_2
      else
      h1=Newmark_P.alf2_1
      h2=Newmark_P.alf2_2
      endif
c―――――――――――――――――――――――★質量比例型
      if (n_damp_type.eq.1 ) then                                     ! 6
      A0 = 2.0*h1*Omega(n_damp_1)
c―――――――――――――――――――――――★剛性比例型
      elseif (n_damp_type.EQ.2 ) then
      A1 = 2.0*h1/Omega(n_damp_1)
```

```
c――――――――――――――――――――――★レーリー型
      elseif (n_damp_type.EQ.3 ) then
      AA = Omega(n_damp_2)*Omega(n_damp_2)-
     *     Omega(n_damp_1)*Omega(n_damp_1)
        if ( AA.EQ.0.0D0 ) then
          write(76,'(''**** AA IS 0 !! ****  '')')
        else
          A0 = 2.0*Omega(n_damp_1)*Omega(n_damp_2)
     *           *(h1*Omega(n_damp_2)-h2*Omega(n_damp_1))/AA
          A1 = 2.0*(h2*Omega(n_damp_2)-h1*Omega(n_damp_1))/AA
        endif
      endif
c――――――――――――――――――――――――――★振動数、固有周期等のタイトル出力
      if (ifl2.eq.1 ) write(iflz2,535) n_mode                                  ! 7
  535 format(I6/2X,'MODE       FREQ.        PERIOD           DAMPING',
     * '         BETA(x)          BETA(y)',
     *          '         BETA(z)')
  536 format(//2X,'MODE       FREQ.        PERIOD           DAMPING',
     * '         BETA(x)          BETA(y)',
     *          '         BETA(z)')
c――――――――――――――――――――――――――★刺激係数、固有周期等計算          ! 8
      do 100 imode=1,n_mode                                                    ! 8.1
      do i=1,n_unknown
      disp(i) = Eigen_Vector(i,imode)                                          ! 8.2
      enddo
c――――――――――――――――――――――――――★刺激係数のゼロセット
      do j=1,3
      Shigeki(j)=0.                                                           ! 8.3
      enddo
      call Multm(load_mass,gskymm,disp,w,max_h_sky,n_unknown)                  ! 8.4
      t_dmd = 0.0
      do i = 1, n_unknown
      t_dmd = t_dmd + disp(i)*w(i)                                             ! 8.5
      enddo
c――――――――――――――――――――――――――★刺激係数
      if(n_local_coord.eq.0) then                                              ! 8.6
         do i= 1, n_point                                  ! 局所座標がない場合の処理
         do j=1,3
          irest =Point(i).irest(j)
         if(irest.gt.0) Shigeki(j) = w(irest) + Shigeki(j)
         enddo
          enddo
      else                                               ! 局所座標がある場合の処理
        do i= 1, n_point
         local_coord=Point(i).local_coord
         if(local_coord.eq.0) then
           do j=1,3
           irest =Point(i).irest(j)
          if(irest.gt.0)  Shigeki(j) = w(irest) + Shigeki(j)
           enddo
         else
           do j=1,3
           xx(j)=0.
           irest =Point(i).irest(j)
```

```
            if(irest.gt.0)  xx(j) = w(irest)
            enddo
             call trans_VT8(xx,yy,rot_local(1,1,local_coord))
            do j= 1, 3
            Shigeki(j) = yy(j) + Shigeki(j)
            enddo
           endif
          enddo

        endif
        if(t_dmd.ne.0.)then
        do j=1,3
        Shigeki(j)=Shigeki(j)/t_dmd                                    ! 8.7
        enddo
        endif
c――――――――――――――――――――――――★刺激係数のチェック
        if(n_local_coord.eq.0) then                                    ! 9
           do i= 1, n_point
           do j=1,3
           yy(j)=0.
            irest =Point(i).irest(j)
           if(irest.gt.0) yy(j) = disp(irest)
           enddo
        do j=1,3
        do k=1,3
        disp_point_m(k,j,i)=disp_point_m(k,j,i)+Shigeki(j)*yy(k)
        enddo
        enddo
           enddo
        else
          do i= 1, n_point
           local_coord=Point(i).local_coord
           if(local_coord.eq.0) then
             do j=1,3
              yy(j)=0.
             irest =Point(i).irest(j)
             if(irest.gt.0) yy(j) = disp(irest)
             enddo
           else
             do j=1,3
             xx(j)=0.
             irest =Point(i).irest(j)
             if(irest.gt.0)  xx(j) = disp(irest)
             enddo
             call trans_VT8(xx,yy,rot_local(1,1,local_coord))
           endif

        do j=1,3
        do k=1,3
        disp_point_m(k,j,i)=disp_point_m(k,j,i)+Shigeki(j)*yy(k)
        enddo
        enddo

        enddo
```

```
      endif
c―――――――――――――――――――――――――★振動数、固有周期等の出力
      hh0  = 0.0D0                                                    ! 10
      if (ifl2.eq.1 ) then
      if (n_damp_type.eq.1.and.Omega(imode).ne.0.0D0 )
     *                 hh0 = A0/(2.0*Omega(imode))
      if (n_damp_type.eq.2 ) hh0 = A1*Omega(imode)/2.0
      if (n_damp_type.eq.3.and.Omega(imode).ne.0.0D0 )
     *                 hh0 = (A0/Omega(imode)+A1*Omega(imode))/2.0
      T_period = 0.0D0
      if (Omega(imode).ne.0.0D0 ) T_period = 2.0*PI/Omega(imode)
      write(76,536)
      write(76,'(I5,6e16.8)') imode,Omega(imode),T_period,
     *                        hh0,(Shigeki(j),J=1,3)
      write(iflz2,'(I5,6e16.8)') imode,Omega(imode),T_period,
     *                        hh0,(Shigeki(j),J=1,3)                  ! 11
      endif
      if ( ifl1.EQ.1 ) then
c―――――――――――――――――――――――――★振動モードの出力
      write(iflz1,1000) imode                                         ! 12
 1000 FORMAT(5X,'***** MODE',I3,' *****')
  200  format(I5,6F10.5)
 1001 FORMAT(5X,'***** MODE',I3,' *****'/
     *'N_point    u         v         w  ',
     *          '    theta_x   theta_y   theta_z')
       write(76,1001) imode
      do i=1,n_unknown
      disp(i) = Eigen_Vector(i,imode)
      enddo
      if(n_local_coord.eq.0) then
         do i= 1, n_point
         do j=1,6
          tdisp(j)=0.
          irest =Point(i).irest(j)
         if(irest.gt.0) tdisp(j) = disp(irest)
         enddo
      write(iflz1,200) i,(tdisp(j),j=1,6)
      write(76,200) i,(tdisp(j),j=1,6)
          enddo
      else
        do i= 1, n_point
         local_coord=Point(i).local_coord
         if(local_coord.eq.0) then
           do j=1,6
           tdisp(j)=0.
           irest =Point(i).irest(j)
          if(irest.gt.0)  tdisp(j) = disp(irest)
           enddo
            write(iflz1,200) i,(tdisp(j),j=1,6)
            write(76,200) i,(tdisp(j),j=1,6)
        else
           do j=1,6
           tdisp(j)=0.
           irest =Point(i).irest(j)
```

```
            if(irest.gt.0)  tdisp(j) = disp(irest)
            enddo
            call trans_VT8(tdisp(1),xx,rot_local(1,1,local_coord))
            call trans_VT8(tdisp(4),yy,rot_local(1,1,local_coord))
              write(iflz1,200) i,(xx(j),j=1,3),(yy(j),j=1,3)
              write(76,200) i,(xx(j),j=1,3),(yy(j),j=1,3)
           endif
          enddo
        endif
        endif
  100 continue
c―――――――――――――――――――――――――★刺激係数のチェック出力
       write(76,'(//a)') ' 刺激係数と振動モードチェック'                    ! 13
       write(76,1002)
 1002 format('N_point','               x_direction',
     *                '               y_direction',
     *                '              ud_direction'/
     *       '             u         v         w   ',
     *              '      u         v         w   ',
     *              '      u         v         w')
       do i= 1, n_point
       write(76,'(i6,10f10.4)') i,((disp_point_m(k,j,i),k=1,3),j=1,3)
       enddo
       return
       end
```

7.4.7 最大変位、加速度、速度ファイル

本節では、計算過程で得た最大変位、最大速度、最大加速度を出力するファイルの仕様とそのプログラムコードについて説明する。上記の最大値は、増分時間毎の計算過程で常にチェックしており、最大値を入れ替えている。しかし、変位や速度、加速度のファイルへの出力は、使用者が設定した間隔で成されており、プレゼンターなどでこの時刻歴ファイルを用いて最大値を求めた場合と、ここでチェックしている最大値とは多少異なる場合がある。

最大値のチェックは、以下のサブルーチンコールで行われ、そのサブルーチンの内容は、後で説明する。

```
c―――――――――――――――――――――――――――――――★変位、速度、加速度の最大値セット(ok)
       call Get_max_disp(Max_disp,Parameter_C.n_point,Point,
      *      past_disp_point, past_vel_point, past_acc_point,
      *      d_max_v,id_max_v,vacc)
```

動的解析が全て終了した後、最大値を出力するサブルーチンコールが行われ、得られた変位、速度、加速度の最大値が節点ごとに ASKII ファイルとして出力される。このファイルの一部を以下に示す。このファイルは、同じ仕様で４つのグループに分けられており、相対加速

度、絶対加速度、速度、変位の順に出力されている。その仕様は、以下に示すように、最初の１行がコメント行であり、２行目以降は、左よりNODE:、節点番号、次に続く３個が負方向の３方向最大値、次の３つが正方向の最大値である。以下の例は全体座標系における x、y、z 方向の相対加速度を示す。

```
     ***** MAXIMUM RELATION ACCELERATION *****
NODE:  1  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
NODE:  2  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
NODE:  3  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
NODE:  4  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
NODE:  5  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
NODE:  6  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
NODE:  7  0.4359E+03  0.0000E+00  0.1074E+02 -0.4708E+03  0.0000E+00 -0.1590E+02
NODE:  8  0.4394E+03  0.0000E+00  0.1434E+03 -0.4833E+03  0.0000E+00 -0.1437E+03
NODE:  9  0.4424E+03  0.0000E+00  0.7903E+02 -0.4865E+03  0.0000E+00 -0.7530E+02
NODE: 10  0.4442E+03  0.0000E+00  0.1544E+02 -0.4879E+03  0.0000E+00 -0.1925E+02
```

これらのサブルーチンは、いずれも submain_dynamic_a()から、以下のように呼ばれている。

```
c―――――――――――――――――――――――――――――★最大変位、速度、加速度の出力
      call Out_max_disp(Max_disp,Parameter_C.n_point,
     *                  Point,ifl(12),iflz(12))
```

上記２つのサブルーチンを具体的に示す。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_max_disp
C       ――――――――――――――――――――――――――――――――――――
C     ●   最大変位最大変位、最大加速度を求める(ok)
C     ――――――――――――――――――――――――――――――――――――
      subroutine Get_max_disp(Max_disp,n_point,Point,
     *      past_disp_point, past_vel_point, past_acc_point,
     *      d_max_v,id_max_v,vacc)
C     ――――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / point_s  / Point
     record / max_disp_s    / Max_disp
     dimension Max_disp(*),Point(*),vacc(3)
     dimension past_disp_point(*),past_vel_point(*),past_acc_point(*)
C     ――――――――――――――――――――――――――――――――――――
c     Max_disp        :structure 最大値
c     n_point         :integer   節点数
c     Point           :structure
c     past_disp_point :real*8 計算結果の変位
c     past_vel_point  :real*8 計算結果の速度
c     past_acc_point  :real*8 計算結果の加速度
```

```
c     vacc(3)         :real*8 地震加速度
C     ―――――――――――――――――――――――――――――――――――――
       d_max_v=0.
      do i=1,n_point                                                  ! 1
      do j=1,3
      ires= Point(i).irest(j)
      if(ires.ne.0) then                                              ! 2
      aa = past_disp_point(ires)
      if(abs(aa).gt.d_max_v)then
      d_max_v=abs(aa)
      id_max_v=i
      endif
      if(Max_disp(i).disp_point(j).lt.aa) Max_disp(i).disp_point(j)=aa  ! 3
      if(Max_disp(i).disp_point(j+3).gt.aa) Max_disp(i).disp_point(j+3)=aa  ! 4
      aa = past_vel_point(ires)
      if(Max_disp(i).vel_point(j).lt.aa) Max_disp(i).vel_point(j)=aa   ! 5
      if(Max_disp(i).vel_point(j+3).gt.aa) Max_disp(i).vel_point(j+3)=aa
      aa = past_acc_point(ires)
      if(Max_disp(i).acc_point(j).lt.aa) Max_disp(i).acc_point(j)=aa   ! 6
      if(Max_disp(i).acc_point(j+3).gt.aa) Max_disp(i).acc_point(j+3)=aa
      aa = past_acc_point(ires)+vacc(j)
      if(Max_disp(i).ab_acc_point(j).lt.aa)                           ! 7
     *     Max_disp(i).ab_acc_point(j)=aa
      if(Max_disp(i).ab_acc_point(j+3).gt.aa)
     *     Max_disp(i).ab_acc_point(j+3)=aa
      else
      aa = vacc(j)
      if(Max_disp(i).ab_acc_point(j).lt.aa)                           ! 8
     *     Max_disp(i).ab_acc_point(j)=aa
      if(Max_disp(i).ab_acc_point(j+3).gt.aa)                         ! 9
     *     Max_disp(i).ab_acc_point(j+3)=aa
    end if
    end do
    end do
    return
    end
C     ―――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Out_max_disp
C       ―――――――――――――――――――――――――――――――――――――
C     ●   最大変位、最大速度、最大加速度を出力する(ok)
C     ―――――――――――――――――――――――――――――――――――――
      subroutine Out_max_disp(Max_disp,n_point,Point,
     *                        ifl,iflz)
      implicit real*8(A-H,O-Z)
    include "submain.h"
    record / point_s  / Point
    record / max_disp_s    / Max_disp
    dimension Max_disp(*),Point(*)
C     ―――――――――――――――――――――――――――――――――――――
c     Max_disp        :structure 最大値
c     n_point         :integer   節点数
c     Point           :structure
C     ―――――――――――――――――――――――――――――――――――――
      if(ifl.ne.1) return                                             ! 10
```

```
      write(iflz,10)                                                    ! 11
   10 format(5X,'***** MAXIMUM RELATION ACCELERATION *****')
      do i=1,n_point
      write(iflz,25) I,(Max_disp(i).acc_point(j),j=1,6)                 ! 12
      enddo
   25 format('NODE:',I3,6E12.4)
C
      write(iflz,15)                                                    ! 13
   15 format(5X,'***** MAXIMUM ABSOLUTE ACCELERATION *****')
      do i=1,n_point
      write(iflz,27) I,(Max_disp(i).ab_acc_point(j),j=1,6)              ! 14
      enddo
   27 format('NODE:',I3,6E12.4)
C
      write(iflz,30)                                                    ! 15
   30 format(5X,'***** MAXIMUM VELOCITY *****')
      do i=1,n_point
      write(iflz,45) I,(Max_disp(i).vel_point(j),j=1,6)                 ! 16
      enddo
   45 format('NODE:',I3,6E12.4)
C
      write(iflz,50)                                                    ! 17
   50 format(5X,'***** MAXIMUM DISPLACEMENT *****')
      do i=1,n_point
      write(iflz,65) I,(Max_disp(i).disp_point(j),j=1,6)                ! 18
      enddo
   65 format('NODE:',I3,6E12.4)
C
    return
    end
```

1. この時刻における節点全体の最大変位を求めるために、最大値を入れる変数 d_max_y をゼロクリアする。
2. その節点における３方向変位の拘束状態を調べ、拘束なしであれば以降の処理を行う。最大変位とその位置である節点番号 id_max_y を調査する。
3. ここからは時刻歴における各節点の最大値をチェックする。まず、変位について行う。負の方向の最大変位をチェックし、最大値ならば値を変更する。
4. 正の方向の最大変位をチェックし、最大値ならば値を変更する。
5. 負の方向の最大速度をチェックし、最大値ならば値を変更する。同じく、正の方向の最大速度をチェックし、最大値ならば値を変更する。
6. 負の方向の最大相対加速度をチェックし、最大値ならば値を変更する。同じく、正の方向の最大相対加速度をチェックし、最大値ならば値を変更する。

7. 負の方向の最大絶対加速度をチェックし、最大値ならば値を変更する。同じく、正の方向の最大絶対加速度をチェックし、最大値ならば値を変更する。次に、その節点の相対加速度に３方向別に地震加速度を加え、絶対加速度を計算する。
8. 節点拘束がある場合は、節点に働く負方向地震加速度をチェックし、最大値ならば値を変更する。
9. 同じく、正方向地震加速度をチェックし、最大値ならば値を変更する。
10. 次に、サブルーチン Out_max_disp()について説明する。まず、出力要求がない場合はこのサブルーチンから直ちに戻る。
11. 最初に、相対加速度の最大値を出力する。出力ユニット番号は iflz である。ここでは、相対加速度であることのコメント行を出力する。
12. 全節点について、最大相対加速度を仕様に従って出力する。最初は節点番号、次の３つが負方向の最大値、最後の３つが正方向の最大値である。
13. ここでは、絶対加速度であることのコメント行を出力する。
14. 全節点について、最大絶対加速度を仕様に従って出力する。最初は節点番号、次の３つが負方向の最大値、最後の３つが正方向の最大値である。
15. 速度であることのコメント行を出力する。
16. 全節点について、最大速度を仕様に従って出力する。最初は節点番号、次の３つが負方向の最大値、最後の３つが正方向の最大値である。
17. 変位であることのコメント行を出力する。
18. 全節点について、最大変位を仕様に従って出力する。最初は節点番号、次の３つが負方向の最大値、最後の３つが正方向の最大値である。

7.4.8 最大応力ファイル

本節では、計算過程で得た最大応力を出力するファイルの仕様とそのプログラムコードについて説明する。上記の最大値は、増分時間毎の計算過程で常にチェックしており、最大値を入れ替えている。しかし、応力のファイルへの出力は、使用者が設定した間隔で成されており、プレゼンターなどでこの時刻歴ファイルを用いて最大値を求めた場合と、ここでチェックしている最大値とは多少異なる場合がある。

最大値のチェックは、最初に示したサブルーチンコールで行われ、その内容は、後で説明する。また、動的解析が全て終了した後、次に示すサブルーチンコールが行われ、得られた応力の最大値が節点ごとにASKII ファイルとして出力される。

```
c―――――――――――――――――――――――――――――――★応力等の最大値セット
      call Get_max_stress(Member,n_member,Max_stress)
c―――――――――――――――――――――――――――――――★最大応力等の出力
      call Out_max_stress(Max_stress,Parameter_C.n_member,
     *                    ifl(16),iflz(16))
```

これらのサブルーチンは、いずれも submain_dynamic_a()から呼ばれている。以下に、２つのサブルーチンの内容を示す。

```
C     ―――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_max_stress
C       ―――――――――――――――――――――――――――――――――――
C     ●   最大応力を求める(ok)
C     ―――――――――――――――――――――――――――――――――――――
      subroutine Get_max_stress(Member,n_member,Max_stress)
      implicit real*8(A-H,O-Z)
       include "submain.h"
       record / member_s  / Member
       record / max_stress_s    / Max_stress
       dimension Member(*),Max_stress(*)
C     ―――――――――――――――――――――――――――――――――――――
c     Max_stress       :structure 最大値
c     n_member         :integer   節点数
C     ―――――――――――――――――――――――――――――――――――――
       do i=1,n_member                                                  ! 1
       do j=1,18                                                        ! 2
      aa = Member(i).stress(j)
       if(Max_stress(i).stress_max_member(j).lt.aa)
     *           Max_stress(i).stress_max_member(j)=aa                  ! 3
       if(Max_stress(i).stress_min_member(j).gt.aa)
     *           Max_stress(i).stress_min_member(j)=aa                  ! 4
       end do
       end do
       return
       end
C     ―――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Out_max_stress
C       ―――――――――――――――――――――――――――――――――――
C     ●   最大応力を出力する(ok)
C     ―――――――――――――――――――――――――――――――――――――
       subroutine Out_max_stress(Max_stress,n_member,
     *                           ifl,iflz)
       implicit real*8(A-H,O-Z)
       include "submain.h"
       record / max_stress_s    / Max_stress
       dimension Max_stress(*)
```

```
      dimension ff(10)
C     ――――――――――――――――――――――――――――――――――――
c     Max_stress        :structure 最大値
c     n_member          :integer   節点数
C     ――――――――――――――――――――――――――――――――――――
      if(ifl.ne.1) return
      WRITE(iflz,510)                                                  ! 5
  510 FORMAT(5X,'***** MAXIMUM fay *****')
       do i=1,5
      ff(i)=0.
      enddo
      DO i=1, n_member                                                 ! 6
        WRITE(iflz,530) I, (ff(j), j=1,5)
  530   FORMAT(I5,5e12.4)
      enddo
C
      WRITE(iflz,511)
  511 FORMAT(5X,'***** MAXIMUM AXIAL FORCE *****')
      DO i=1, n_member                                                 ! 7
        WRITE(iflz,531) i, Max_stress(i).stress_max_member(1),
     *                  Max_stress(i).stress_min_member(1)
  531   FORMAT(I5,2e12.4)
      enddo
        WRITE(iflz,512)
  512 FORMAT(5X,'***** MAXIMUM BENDING MOMENT *****')
      DO i=1, n_member                                                 ! 8
        WRITE(iflz,532) i, Max_stress(i).stress_max_member(5),
     *                     Max_stress(i).stress_max_member(6),
     *                     Max_stress(i).stress_max_member(11),
     *                     Max_stress(i).stress_max_member(12),
     *                     Max_stress(i).stress_max_member(17),
     *                     Max_stress(i).stress_max_member(18),
     *                     Max_stress(i).stress_max_member(17),
     *                     Max_stress(i).stress_max_member(18),
     *                     Max_stress(i).stress_max_member(17),
     *                     Max_stress(i).stress_max_member(18)
         WRITE(iflz,532) i, Max_stress(i).stress_min_member(5),
     *                     Max_stress(i).stress_min_member(6),
     *                     Max_stress(i).stress_min_member(11),
     *                     Max_stress(i).stress_min_member(12),
     *                     Max_stress(i).stress_min_member(17),
     *                     Max_stress(i).stress_min_member(18),
     *                     Max_stress(i).stress_min_member(17),
     *                     Max_stress(i).stress_min_member(18),
     *                     Max_stress(i).stress_min_member(17),
     *                     Max_stress(i).stress_min_member(18)
  532   FORMAT(I5,10e12.4)
      enddo
C
       return
       end
```

1．応力の最大値をチェックするために、部材全部について以下の処理を行う。
2．部材両端と中央点における 6 つの合応力について、最大値をチェックする。
3．応力の最小値を求める。
4．応力の最大値を求める。
5．現在は、塑性関数値の最大値は求めていないため、ここでは、全部材について、ゼロを出力する。
6．軸力の最小値と最大値を全部材について出力する。
7．全部材について、両端及び中央の両軸に関する曲げモーメントの最小値と最大値を出力する。