9.5 剛床モデル

9.5.1 はじめに

SPACE（Ver.3.00）では、立体骨組の静的解析や動的解析を行っている。その際、通常の立体骨組構造では、床の扱いをどのようにするかが問題となる。以前のバージョンでは、床をブレースに置換して解析を行っていた。ここでは、立体骨組構造で良く用いられる剛床仮定を SPACE に組み込む手法について理論的に検討する。さらに、この節以降では、実際のプログラムを用いて剛床仮定を SPACE に組み込む方法を示すことにする。剛床の変換行列は、局所座標変換とは異なり、軸力と曲げモーメントが連成するため、釣合座標系を得るため全ての座標変換を変更する必要があり、多くのサブルーチンに影響する。そのため剛床仮定の機能を組み込むために、SPACE を大幅に変更することになる。ここでは、これらを全て検討しよう。

剛床仮定を SPACE に組み込むためには、以下の部分に変更を加えることになる。

１．入力仕様
２．座標変換式の保存
３．座標変換式、特に局所座標系との関係
４．剛性行列の釣合座標系への変換
５．質量行列の釣合座標系への変換
６．右辺項の釣合座標系への変換
７．釣合座標系から部材座標系に変換
８．釣合座標系から全体座標系に変換

9.5.2 座標変換

立体骨組の中で、床の剛性が他の部分に比較して著しく硬いとき、この床を剛床として扱うことができる。本節では、剛床モデルの理論的な処理手法について述べる。

ここでは、床は水平レベルにあるものとする。また、この床は剛であり、そのため、その床内にある代表点で床内の任意の節点の床面内の変位を表すことができるものとする。ただし、面外の変位に対しては、床内の節点は独自に挙動するものとする。

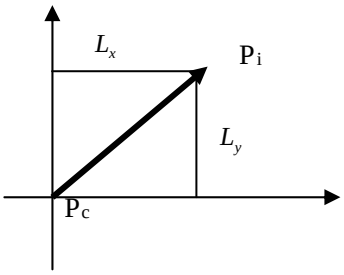
まず、代表節点と他の節点の関係を検討する。剛床モデルとして、代表節点と他の節点が長さ L の剛部材で連結しているとする。ここで、部材の代表節点側を c 端、他の節点を i 端とし、この剛部材の x 方向長さを L_x、y 方向長さを L_y とする。その剛部材をはさんで、全体座標系で表された両端の増分変位ベクトルを以下のように表す。

$$\left.\begin{aligned}\{\Delta u_i\}^T &= \{\Delta u_i, \Delta v_i, \Delta w_i, \Delta\theta_{xi}, \Delta\theta_{yi}, \Delta\theta_{zi}\} \\ \{\Delta u_c\}^T &= \{\Delta u_c, \Delta v_c, \Delta w_c, \Delta\theta_{xc}, \Delta\theta_{y_c}, \Delta\theta_{zc}\}\end{aligned}\right\} \cdots\cdots\cdots\cdots (9.1)$$

ここで、$\{\Delta u_i\}$は、任意節点側の増分変位であり、$\{\Delta u_c\}$は代表節点 c 側の増分変位である。また、同様に、全体座標系で表された両端の増分節点力ベクトルを次式で示す。

$$\left.\begin{aligned}\{\Delta f_i\}^T &= \{\Delta N_i, \Delta Q_{yi}, \Delta Q_{zi}, \Delta M_{xi}, \Delta M_{yi}, \Delta M_{zi}\} \\ \{\Delta f_c\}^T &= \{\Delta N_c, \Delta Q_{yc}, \Delta Q_{zc}, \Delta M_{xc}, \Delta M_{yc}, \Delta M_{zc}\}\end{aligned}\right\} \cdots\cdots\cdots (9.2)$$

剛床上の節点変位は、その床上の代表節点の変位で表すことになる。剛床上の他の節点変位は、この代表点における 2 方向の変位と、代表点の z 軸回りの回転による x 方向と y 方向の変位を考慮する。



$$\begin{Bmatrix}\Delta u_i \\ \Delta v_i \\ \Delta\theta_{zi}\end{Bmatrix} = \begin{bmatrix}1 & 0 & -L_y \\ 0 & 1 & L_x \\ 0 & 0 & 1\end{bmatrix}\begin{Bmatrix}\Delta u_c \\ \Delta v_c \\ \Delta\theta_{zc}\end{Bmatrix}$$

$$\{\Delta u_i\} = [R_u]\{\Delta u_c\} \qquad \cdots\cdots\cdots (9.3)$$

ここで、両者共に全体座標系で表された変位とする。また、上式の変位変換式では、z 軸の回転に関する幾何学的非線形性を考慮していない。

次に、節点力間の関係を示す。

$$\begin{Bmatrix}\Delta P_{xi} \\ \Delta P_{yi} \\ \Delta M_{zi}\end{Bmatrix} = \begin{bmatrix}1 & 0 & 0 \\ 0 & 1 & 0 \\ L_y & -L_x & 1\end{bmatrix}\begin{Bmatrix}\Delta P_{xc} \\ \Delta P_{yc} \\ \Delta M_{zc}\end{Bmatrix}$$

$$\{\Delta P_i\} = [R_f]\{\Delta P_c\} \qquad \cdots\cdots\cdots (9.4)$$

ここでも、両者の節点力は全体座標系で表されているものとする。上記した 2 つの変換行列には、次のような特異な関係が存在する。

$$[R_u][R_f]^T = [I]$$

$$\begin{bmatrix}1 & 0 & -L_y \\ 0 & 1 & L_x \\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & L_y \\ 0 & 1 & -L_x \\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1\end{bmatrix} \qquad \cdots\cdots\cdots (9.5)$$

したがって、

$$[R_u]^{-1} = [R_f]^T;\quad [R_f]^{-1} = [R_u]^T \qquad \cdots\cdots\cdots (9.6)$$

という関係が存在し、これらの関係は後で利用することになる。

剛性行列を全体座標系から釣合座標系に変換してみよう。ただし、ここでは、局所座標系は使用しないものとする。次式は、全体座標系で表された釣合式である。

$$\{\Delta P_i\} = [k]\{\Delta u_i\} \qquad \cdots\cdots\cdots(9.7)$$

式(9.3)と(9.4)を利用して、釣合座標系に変換する。

$$\left[\bar{R}_f\right]\{\Delta P_c\} = [k]\left[\bar{R}_u\right]\{\Delta u_c\} \qquad \cdots\cdots\cdots(9.8)$$

上式の左より、$\left[\bar{R}_f\right]^{-1}$を掛け、式(9.6)を用いると、

$$\{\Delta P_c\} = \left[\bar{R}_u\right]^T [k]\left[\bar{R}_u\right]\{\Delta u_c\} \qquad \cdots\cdots\cdots(9.9)$$

となり、釣合座標系における剛性行列が次のように得られる。

剛性行列の変換式

$$\left[\bar{k}\right] = \left[\bar{R}_u\right]^T [k]\left[\bar{R}_u\right]$$
$$\left[\bar{R}_u\right] = \begin{bmatrix} R_u & 0 \\ 0 & I \end{bmatrix} \qquad \cdots\cdots\cdots(9.10)$$

上式で、i 端は剛床位置にあり、 j 端は一般の節点としている。両者共に剛床位置である場合は、式(9.10)中の単位行列の替わりに剛床変換行列を用いることになる。ここで、$[R_u]$は以下のようである。

$$[R_u] = \begin{bmatrix} 1 & & & & & & -L_y \\ & 1 & & & & & L_x \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & 1 & \end{bmatrix} \qquad \cdots\cdots\cdots(9.11)$$

上式のL_xとL_yは、剛床代表点から当該節点までの x 方向、y 方向の長さを表す。

質量行列も、剛性行列と同様に全体座標系から釣合座標系に変換可能である。ここでも局所座標系は使用しないものとする。釣合座標系で表される質量行列は、次式で表される。

質量行列の変換式

$$\left[\bar{m}\right] = \left[\bar{R}_u\right]^T [m]\left[\bar{R}_u\right]$$
$$\left[\bar{R}_u\right] = \begin{bmatrix} R_u & 0 \\ 0 & I \end{bmatrix} \qquad \cdots\cdots\cdots(9.12)$$

右辺項ベクトルを釣合座標系に変換しよう。動的解析の右辺項は、ニューマークβ法を適用すると次式となる。

右辺項の変換式

$$\{f\}=\{G\}+\{g\} \quad \cdots\cdots\cdots(9.13)$$

$$\{G(y_{n+1},\Delta y_{n+1})\}=-\left\{\bar{f}_d\right\}-\left[K_T(y_n)\right]\{\Delta y_{n+1}\}+\left[K\right]\{y_{n+1}\} \quad \cdots\cdots\cdots(9.14)$$

$$\{g\}=-\left[M\right]\left[I\right]\left\{\ddot{u}_g\right\}+\{P_S\}-\{Q(y_n)\}-\left[\bar{C}\right]\{a\}-\left[K\right]\{\bar{b}\} \quad \cdots\cdots\cdots(9.15)$$

式(9.13)に示されるように、加速度は釣合座標系であるため、式(9.14)の線形剛性行列と接線剛性行列は釣合座標系に変換しておく必要がある。次に、式(9.15)の第１と２項は、各節点において全体座標系で求めた後、釣合座標系に変換する。したがって、ここでの質量行列は、全体座標系である必要がある。

第３項は、部材座標系の節点力から釣合座標系の節点力ベクトルに変換する。第４と５項は、$\{a\}$と$\{b\}$は釣合座標系で求められているため、減衰行列$[C]$と接線剛性行列$[K_T]$は、釣合座標系で求めておく必要がある。

9.5.3 入力仕様

本節では、剛床仮定に対する入力仕様を決めるが、現在の仕様を大きく変更することはできない。そこで、節点に関するデータ入力を次のように定義する。

節点データの変更点は、以下のようである。

１．節点の３次元座標の後のデータ入力項目を利用する。
２．局所座標系を使用する。ここで、剛床用の変換行列を局所座標変換行列と同じ領域にセットする。ただし、変換行列は、局所座標変換行列と異なるので変換サブルーチンは異なる。
３．節点の同一視を利用して、剛床の代表点の変位を参照する。

以下に、節点に関する入力仕様を示し、変更部分を提示する。

１）節点座標

この入力項目で剛床仮定用のデータ設定を行う

節点の３次元座標をセットする。節点数 NODE だけ繰り返す。

I	POSIT(1,I)	POSIT(2,I)	POSIT(3,I)	NDOUT(I)

I：節点番号
POSIT(1,I)：節点の X 座標（cm）
POSIT(2,I)：節点の Y 座標（cm）

POSIT(3,I)：節点の Z 座標（cm）

NDOUT(I)：通常節点：0

剛床用節点：0 以外の剛床グループ番号

剛床番号：0 は自由節点

正番号　剛床グループ番号

負番号　その絶対値番号の剛床グループの代表節点

この剛床の代表節点が設定されていない場合は、最も小さな節点番号の節点を代表節点とする。また、グループ内に 2 節点以上ない場合は、剛床処理は行わない。

２）局所座標系

節点に局所座標系を考慮する場合、その節点について、全体座標系から局所座標軸への回転角をセットする。このレコードは局所座標系を考慮する節点数(LOCOD*3) だけ繰り返す。**LOCOD=0 の場合は以下のデータを設定してはならない。**

剛床上の節点は、局所座標系を用いてはならない

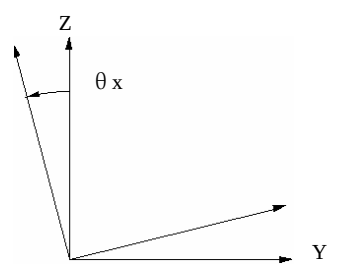
I	TLCX(I)	TLCY(I)	TLCZ(I)

I：局所座標系を考慮する節点番号

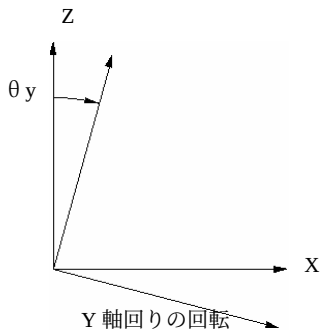
TLCX(I)：X 軸回りの回転角 θx（度）

TLCY(I)：Y 軸回りの回転角 θy（度）

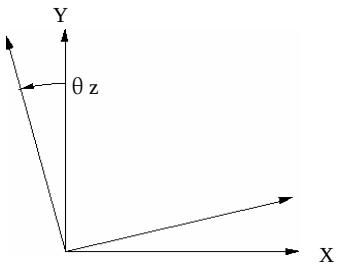
TLCZ(I)：Z 軸回りの回転角 θz（度）



X 軸回りの回転



Y 軸回りの回転



Z 軸回りの回転

図 9-2　局所座標（右手・右ねじの法則）

３）節点の拘束指標

節点の拘束条件の指標を入力し、拘束節点数(NRBOUND) だけ繰り返す。

I	IRE(1,I)	IRE(2,I)	IRE(3,I)	IRE(4,I)	IRE(5,I)	IRE(6,I)

I：拘束する節点番号

IRE(1,I)：X 方向の変位の拘束指標

IRE(2,I)：Y 方向の変位の拘束指標

IRE(3,I)：Z 方向の変位の拘束指標

IRE(4,I)：X 軸回りの回転の拘束指標

IRE(5,I)：Y 軸回りの回転の拘束指標

IRE(6,I)：Z 軸回りの回転の拘束指標

拘束指標は、次のようである。

IRE(*,I)=　0：自由

IRE(*,I)=－1：固定

他節点の自由度の変位と同一視する場合は、次のように拘束指標をセットする。

IRE(*,I)=－(10×K+L)：節点 K の自由度 L と同変位

ここで、方向 L とは、

=1：X 方向、=2：Y 方向、=3：Z 方向

=4：X 軸回りの回転、=5：Y 軸回りの回転、=6：Z 軸回りの回転

節点の拘束指標
０：自由
－１：固定
－(10×K+L)：
他節点の変位と同一視

剛床仮定を用いる節点は、システムが自動的に、x 方向変位、Y 方向変位、z 軸回りの回転に、他節点との同一視を行う拘束指標を設定する

9.5.4 他節点の変位と同一視を利用

剛床処理を行う場合、他節点の変位との同一視処理を利用する。従って、剛床が存在する場合は、他節点の変位との同一視を行うために、システム内で、当該節点の拘束条件に変更を加える。まず、剛床上の節点で、全体座標系での変位は、

$$\{u\}=\{u \quad v \quad w \quad \theta_x \quad \theta_y \quad \theta_z\}$$

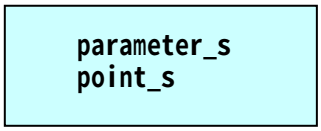
で表され。第 1、2、6 項目の変位を、代表点の変位の対応する変位と同一視する。これをシステム内のプログラムで変更する。例えば、代表節点が 10 で、剛床節点が 20 とすると、

```
20  -101   -102    0       0    0    -106
```

となる。

9.5.5 構造体の拡張

剛床処理を行うため、次のように２つの構造体の仕様拡張を行う。この構造体は、



であり、次のように変更される。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   parameter_s 構造体
C     ―――――――――――――――――――――――――――――――――――
c
c     解析パラメータ
      structure  / parameter_s/
      integer    n_unknown       !  全自由度
      integer    n_point         !  節点数
      integer    n_element       !  要素数
      integer    n_element_dll   !  DLL 用要素数
      integer    n_S_comp_model  !  任意型静的縮合モデル数
      integer    nE_New_Element  !  任意型静的縮合モデルに含まれる要素エレメント数
      integer    nM_New_Element  !  任意型静的縮合モデルに含まれる部材エレメント数
      integer    n_member        !  部材数
      integer    n_rot_axis      !  主軸回転部材数
      integer    n_local_coord   !  局所座標系を使用する節点数
      integer    n_boundary_p    !  境界節点数
      integer    nc_member       !  部材減衰機構を有する部材数
      integer    n_member_dll    !  DLL 用部材数
      integer    n_free          !  節点当たり解析自由度数
      integer    n_dim           !  解析次元数
      integer    n_skyline       !  スカイライン行列の領域数
      integer    n_sky_ave       !  平均バンド幅
      integer    n_gouyuka       !  剛床の有無
      end structure
c      record /parameter_s/ Parameter_C
c
C     ―――――――――――――――――――――――――――――――――――
C     ●   point_s 構造体
C     ―――――――――――――――――――――――――――――――――――
c
c     節点
      structure  / point_s/
      real*8   coord(3)          ! 3 次元の節点座標
      integer  irest(6)          ! 節点自由度の拘束表
      integer  local_ coord      ! 局所座標系の有無（ある場合は、回転行列の通し番号）
      integer  n_group_gouyuka   ! 剛床のグループ番号
      real*8   coord_local(3)    ! 局所座標系（全体座標系に対する角度）
      real*8   disp_initial(3)   ! 初期変位
      real*8   mass_1            ! 第一ステップ質量（ダミー）
      real*8   mass_2            ! 第二ステップ質量（ダミー）
      end structure
c     record /point_s/ Point
```

構造体要素の新規挿入

この２つの要素の仕様を拡張する

```
c     ALLOCATABLE ::Point(:)
c     ALLOCATE (Point(n_point))
c
```

上記２つの構造体要素の仕様を次のように拡張する。

```
integer   local_ coord      ! 局所座標系の有無：剛床節点の有無
      剛床節点の場合は-1 をセットする。
real*8    coord_local(3)    ! 剛床節点の場合、次の座標をセットする。
        coord_local(1)：Lx
        coord_local(2)：Ly
intger   n_group_gouyuka：剛床グループ番号
```

9.5.6 剛床データの設定

仕様に従って、剛床データが入力されるとき、まず、その入力データから、以下のサブルーチンを用いて解析に必要となる情報を構造体に設定する。最初に、構造データを入力するサブルーチンを以下のように変更する。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_structure
C     ―――――――――――――――――――――――――――――――
C     ●   構造データを入力し、データをダンプファイルに出力する。
C     ―――――――――――――――――――――――――――――――
      subroutine Get_structure(Point,Member,Element,Parameter_C,
     *           Model_type,ierr,
     *           S_comp_model,E_Fiber_work)
       .
       .
c――――――――――――――――――――――――――――――――★節点データ入力
      write(damp_out,1002)
 1002 format(///1h ,'      節点座標'/)
      do i=1,node
      read(5,*,err=9912,end=9918) ii,x,y,z,idm
      write(damp_out,'(i4,3f12.3,i4)')ii,x,y,z,idm  ! ii:節点番号 idm:剛床グループ番号
      Point(ii).coord(1)  = x
      Point(ii).coord(2)  = y
      Point(ii).coord(3)  = z
      Point(ii).n_group_gouyuka = idm
      end do
```

次のサブルーチンでは、剛床処理に必要となる情報を構造体に設定する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Set_gouyuka
C     ―――――――――――――――――――――――――――――――――
C     ●     剛床データの設定
C     ―――――――――――――――――――――――――――――――――
      subroutine Set_gouyuka(Parameter_C,Point )
C     ―――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / parameter_s  / Parameter_C
      record / point_s      / Point
      dimension Point(*)
     integer, ALLOCATABLE :: nx_yuka(:)
      node= Parameter_C.n_point
c――――――――――――――――――――――――★剛床の有無と剛床グループ番号の最大値チェック
      Parameter_C.n_gouyuka = 0
      n_g=0
      do i=1,node
      if(Point(i).n_group_gouyuka .ne. 0) then
      if(n_g.le.iabs(Point(i).n_group_gouyuka))
     * n_g = iabs(Point(i).n_group_gouyuka)
      endif
      enddo
      if(n_g.eq.0) return
      Parameter_C.n_gouyuka = 1
c――――――――――――――――――――――★剛床の代表点検索
      ALLOCATE (nx_yuka(n_g))
      do i=1,n_g
      nx_yuka(i)=0
      enddo
      do i=1,node
       if(Point(i).n_group_gouyuka .ne. 0) then
        n= Point(i).n_group_gouyuka
        if(n.gt.0 ) then
          if(nx_yuka(n) .eq. 0) nx_yuka(n)=-1
        else
          nx_yuka(-n)=I
          Point(j). n_group_gouyuka= 0
        endif
      endif
      enddo
c――――――――――――――――――――――★
      do i=1,n_g
      if(nx_yuka(i) .eq. – 1) then
       do j=1,node
        if(Point(j).n_group_gouyuka .eq. i) then
          nx_yuka(i)=j
          Point(j). n_group_gouyuka= 0
          goto 100
        endif
       enddo
       endif
100   continue
       enddo
```

```
c――――――――――――――――――――――★
      do i=1,n_g
      nx= nx_yuka(i)
      if(nx.gt.0) then
       do j=1,node
       if(i.eq. Point(j).n_group_gouyuka) then
         Point(j).coord_local(1) = Point(nx).coord(2) - Point(j).coord(2)
         Point(j).coord_local(2) = Point(j).coord(1)  - Point(nx).coord(1)
         nxx = -nx*10
          Point(i).irest(1)= nxx -1
          Point(i).irest(2)= nxx -2
          Point(i).irest(6)= nxx -6
       endif
       enddo
      endif
      enddo
      DEALLOCATE (nx_yuka)
      return
      end
```

上のコードで、Ly **は最初から負符号を付けてセットしているので注意されたい**。このため、理論式とコードの符号が異なることになる。

9.5.7 ベクトルの変換

節点における変位と節点力を変換するサブルーチンを示す。まず、釣合座標系の変位を全体座標系に変換するサブルーチンは、次式

$$u_i = u_c - L_y\theta_{zc};\quad u_c = u_i + L_y\theta_{zi}$$
$$v_i = v_c + L_x\theta_{zc};\quad v_c = v_i - L_x\theta_{zi}$$

を用いる。

```
C     ――――――――――――――――――――――――――――――――――
C     ●    SUBROUTINE /Set_gtrans_u
C     ――――――――――――――――――――――――――――――――――
C     ●     変位ベクトルの剛床座標変換
C     ――――――――――――――――――――――――――――――――――
      subroutine Set_gtrans_u(ik,u,aly,alx )
C     ――――――――――――――――――――――――――――――――――
C     ik=1 釣合座標系から全体座標系に変換
C     ――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      dimension u(6)
      if(ik.eq.1) then
      u(1)=u(1) + aly*u(6)
      u(2)=u(2) + alx*u(6)
      else
      u(1)=u(1) - aly*u(6)
      u(2)=u(2) - alx*u(6)
```

```
      endif
      return
      end
```

次に、節点力を変換するサブルーチンを以下に示す。全体座標系の節点力から釣合座標系に変換するサブルーチンは、次式

$$M_{cz} = M_z - L_y P_{xi} + L_x P_{yi};\quad M_{ci} = M_c + L_y P_{xc} - L_x P_{yc}$$

を用いる。

```
C     ―――――――――――――――――――――――――――――――――
C     ●    SUBROUTINE /Set_gtrans_f
C     ―――――――――――――――――――――――――――――――――
C     ●     力ベクトルの剛床座標変換
C     ―――――――――――――――――――――――――――――――――
      subroutine Set_gtrans_f(ik,f,aly,alx )
C     ―――――――――――――――――――――――――――――――――
C     ik=1 釣合座標系から全体座標系に変換
C     ―――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      dimension f(6)
      if(ik.eq.1) then
      f(6)=f(6) - aly*f(1) - alx*f(2)
      else

      f(6)=f(6) + aly*f(1) + alx*f(2)
      endif
      return
      end
```

9.5.8 剛性行列の変換

本節では、剛性剛列などの座標変換について述べる。変換する行列は全体座標系で与えられているものとし、釣合座標系に変換する式の詳細は、マニュアル理論編を参照されたい。変換後の剛性は

$$\left[\bar{k}\right] = [k] + [k_L]$$

$$[k_L] = \begin{bmatrix}
0 & & & k_{1,1}L_{1y} + k_{1,2}L_{1x} & & & & & k_{1,7}L_{2y} + k_{1,8}L_{2x} \\
& 0 & & k_{2,1}L_{1y} + k_{2,2}L_{1x} & & & & & k_{2,7}L_{2y} + k_{2,8}L_{2x} \\
& & 0 & k_{3,1}L_{1y} + k_{3,2}L_{1x} & & & & & k_{3,7}L_{2y} + k_{3,8}L_{2x} \\
& & & 0 & k_{4,1}L_{1y} + k_{4,2}L_{1x} & & & & k_{4,7}L_{2y} + k_{4,8}L_{2x} \\
& & & & 0 & k_{5,1}L_{1y} + k_{5,2}L_{1x} & & & k_{5,7}L_{2y} + k_{5,8}L_{2x} \\
& & & & & k_{6,1}L_{1y} + k_{6,2}L_{1x} & & & k_{6,7}L_{2y} + k_{6,8}L_{2x} \\
& & & & & k_{7,1}L_{1y} + k_{7,2}L_{1x} & 0 & & k_{7,7}L_{2y} + k_{7,8}L_{2x} \\
& & & & & k_{8,1}L_{1y} + k_{8,2}L_{1x} & & 0 & k_{8,7}L_{2y} + k_{8,8}L_{2x} \\
& & & & & k_{9,1}L_{1y} + k_{9,2}L_{1x} & & & 0 \quad k_{9,7}L_{2y} + k_{9,8}L_{2x} \\
& & & & & k_{10,1}L_{1y} + k_{10,2}L_{1x} & & & 0 \quad k_{10,7}L_{2y} + k_{10,8}L_{2x} \\
& & & & & k_{11,1}L_{1y} + k_{11,2}L_{1x} & & & 0 \quad k_{11,7}L_{2y} + k_{11,8}L_{2x} \\
& & & & & k_{12,1}L_{1y} + k_{12,2}L_{1x} & & & k_{12,7}L_{2y} + k_{12,8}L_{2x}
\end{bmatrix} +$$

$$
\begin{bmatrix}
0 & & & & & & & & & & & \\
 & 0 & & & & & & & & & & \\
 & & 0 & & & & & & & & & \\
 & & & 0 & & & & & & & & \\
 & & & & 0 & & & & & & & \\
 & & & & & 0 & & & & & & \\
k_{1,1}L_{1y} & k_{1,2}L_{1y} & k_{1,3}L_{1y} & k_{1,4}L_{1y} & k_{1,5}L_{1y} & k_{1,6}L_{1y} & k_{1,7}L_{1y} & k_{1,8}L_{1y} & k_{1,9}L_{1y} & k_{1,10}L_{1y} & k_{1,11}L_{1y} & k_{1,12}L_{1y} \\
+k_{2,1}L_{1x} & +k_{2,2}L_{1x} & +k_{2,3}L_{1x} & +k_{2,4}L_{1x} & +k_{2,5}L_{1x} & +k_{2,6}L_{1x} & +k_{2,7}L_{1x} & +k_{2,8}L_{1x} & +k_{2,9}L_{1x} & +k_{2,10}L_{1x} & +k_{2,11}L_{1x} & +k_{2,12}L_{1x} \\
 & & & & & & 0 & & & & & \\
 & & & & & & & 0 & & & & \\
 & & & & & & & & 0 & & & \\
 & & & & & & & & & 0 & & \\
 & & & & & & & & & & 0 & \\
k_{7,1}L_{2y} & k_{7,2}L_{2y} & k_{7,3}L_{2y} & k_{7,4}L_{2y} & k_{7,5}L_{2y} & k_{7,6}L_{1y} + & k_{7,7}L_{2y} & k_{8,8}L_{2y} & k_{7,9}L_{2y} & k_{7,10}L_{2y} & k_{7,11}L_{2y} & k_{7,12}L_{2y} \\
+k_{8,1}L_{2x} & +k_{8,2}L_{2x} & +k_{8,3}L_{2x} & +k_{8,4}L_{2x} & +k_{8,5}L_{2x} & k_{8,6}L_{1x} & +k_{8,7}L_{2x} & +k_{8,8}L_{2x} & +k_{8,9}L_{2x} & +k_{8,10}L_{2x} & +k_{8,11}L_{2x} & +k_{8,12}L_{2x}
\end{bmatrix} +
$$

$$
\begin{bmatrix}
0 & & & & & 0 & & & & & 0 & \\
 & 0 & & & & 0 & & & & & 0 & \\
 & & 0 & & & 0 & & & & & 0 & \\
 & & & 0 & & 0 & & & & & 0 & \\
 & & & & 0 & 0 & & & & & 0 & \\
 & & & & & L_{1y}^2k_{1,1} + 2L_{1y}k_{1,2}L_{1x} + L_{1x}^2k_{2,2} & & & & & L_{1y}k_{1,7}L_{2y} + L_{1y}k_{2,7}L_{2x} + L_{1x}k_{1,8}L_{2y} + L_{1x}k_{2,8}L_{2x} & \\
 & & & & & 0 & 0 & & & & 0 & \\
 & & & & & 0 & & 0 & & & 0 & \\
 & & & & & 0 & & & 0 & & 0 & \\
 & & & & & 0 & & & & 0 & 0 & \\
 & & & & & 0 & & & & & 0 & \\
 & & & & & L_{2y}k_{7,1}L_{1y} + L_{2y}k_{8,1}L_{1x} + L_{2x}k_{7,2}L_{1y} + L_{2x}k_{8,2}L_{1x} & & & & & L_{2y}^2k_{7,7} + 2L_{2y}k_{7,8}L_{2x} + L_{2x}^2k_{8,8} &
\end{bmatrix}
$$

で与えられる。

剛性行列を変換するサブルーチンを以下に示す。

```
C     ------------------------------------------------------------
C     ●    SUBROUTINE /Set_gtrans_k
C     ------------------------------------------------------------
C     ●     剛床データの設定
C     ------------------------------------------------------------
      subroutine Set_gtrans_k(ak,aly1,alx1, aly2,alx2 )
C     ------------------------------------------------------------
      implicit real*8(A-H,O-Z)
      dimension ak(12,12),u1(12),u2(12),v(3)
      do j=1,12
      u1(j)=aly1*ak(j,1)+alx1*ak(j,2)
      u2(j)=aly2*ak(j,7)+alx2*ak(j,8)
      enddo
      v(1)=aly1*u1(1)+alx1*u1(2)
      v(2)=aly1*u2(1)+alx1*u2(2)
      v(3)=aly2*u2(7)+alx2*u2(8)
      do j=1,12
      ak(6,j)=ak(6,j)+u1(j)
```

```
      ak(12,j)=ak(12,j)+u2(j)
      ak(j,6)=ak(j,6)+u1(j)
      ak(j,12)=ak(j,12)+u2(j)
      enddo
      ak(6,6)=ak(6,6)+v(1)
      ak(6,12)=ak(6,12)+v(2)
      ak(12,6)=ak(12,6)+v(2)
      ak(12,12)=ak(12,12)+v(3)
      return
      end
```

9.5.9 質量行列の変換

質量行列の変換は、一般的には、剛性行列と同じで良い。ただし、集中質量系では、全体座標系における質量行列が対角行列であり、しかも、回転慣性項を無視している。この集中質量の変換の詳細は、マニュアル理論編を参照されたい。

最終的に、変換後の質量行列は以下のようである。

$$[\bar{m}]=\begin{bmatrix} m_{11} & & & & & -m_{11}L_y \\ & m_{22} & & & & m_{22}L_x \\ & & m_{33} & & & \\ & & & 0 & & \\ & & & & 0 & \\ -m_{11}L_y & m_{22}L_x & & & & L_y m_{11}L_y + L_x m_{22}L_x \end{bmatrix}$$

このように、剛床を考慮すると、集中質量系であっても、回転慣性項を有することになる。

剛床を考慮する集中質量行列を求めるサブルーチンを以下の示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Set_gtrans_m
C     ―――――――――――――――――――――――――――――――――
C     ●    剛床データの設定
C     ―――――――――――――――――――――――――――――――――
      subroutine Set_gtrans_m(am1,am,aly1,alx1)
C     ―――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      dimension am(6,6)
      do I=1,6
      do j=1,6
      am(I,j)=0.
      enddo
      enddo
      am(1,1)=am1
```

```
      am(2,2)=am1
      am(3,3)=am1
      am(1,6)=am1*aly1
      am(2,6)=am1*alx1
      am(6,1)=am(1,6)
      am(6,2)=am(2,6)
      am(6,6)=am1*(aly1*aly1+alx1*alx1)
      return
      end
```

9.5.10 動的解析システムへの組み込み

9.5.10.1 係数行列などの変換

剛床仮定の処理を SPACE の動的解析に組み込むことにする。先に示したように、まず、構造体を拡張する。その後、構造データを入力するサブルーチンを変更する。これについては、既に先の節で説明した。

次に、動的解析の主サブルーチンの中で、構造データを入力した後、剛床に関する情報処理を行うサブルーチンをコールする。

```
c－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
c
c    ★          構造・荷重データを入力し、データの設定を行う
c
c－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
c－－－－－－－－－－－－－－－－－－－－－－－－－－－★基本構造データを入力(ok)
c     write(damp_out,*) ' Get_structure in'
      nfix=5
      nfi=1
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat =10
      call err_outf(ierr_dat)
      return
      endif
      call Get_structure(Point,Member,Element,Parameter_C,
     *                   Model_type,ierr,
     *                   S_comp_model,E_Fiber_work)
      close(nfix)
      if(ierr .ne. 0) then
      ierr_dat =iabs(ierr)
c                         err No. 12-19 使用
      call err_outf(ierr_dat)
      return
      endif
c     write(damp_out,*) ' Get_structure ok'
c－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
c
c    ★          剛床の設定
c
c－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
      call Set_gouyuka(Parameter_C,Point)
```

次に、剛性行列や質量行列に関する座標変換をみてみよう。まず、主サブルーチンの中で、次のように座標変換を行うサブルーチンをコールする。

```
c――――――――――――――――――――――――――――――――★部材の線形剛性計算(ok)
      call Cal_stiff_linear(Model_type,Element,Member,Parameter_C,
     *      ak_linear,E_model11,E_model_fiber,M_model11,M_model_fiber,
     *      E_model12,M_model12,E_model13,M_model13,E_model15,M_model15,
     *      E_model21,M_model21,E_model22,M_model22,
     *      E_model31,M_model31,E_model32,M_model32,
     *      E_model33,M_model33,
     *      Bilinear_work,Trilinear_work,Concrete_work,
     *      work1_elememt,work2_elememt,work1_member,work2_member,
     *      S_comp_model, E_modelx, M_modelx,
     *      E_fiber_work, M_fiber_work)
c     write(damp_out,*) ' Cal_stiff_linear Ok'
c――――――――――――――――――――――――――――――――★剛性の釣合座標系への変換(ok)
      call Rotate_stiffness(Parameter_C,ak_linear,rot_memb,Point,Member)
c     write(damp_out,*) ' Rotate_stiffness Ok'
c――――――――――――――――――――――――――――――――★部材の減衰行列計算(ok)
      if(Parameter_C.nc_member .ne. 0) then
      call Cal_damp_linear(Element,Member,Parameter_C,ac_member,
     *           E_model6_real,work1_elememt,
     *           work2_elememt,work1_member,work2_member)
c     write(damp_out,*) ' Cal_damp_linear Ok'
c――――――――――――――――――――――――――――――――★部材減衰行列の釣合座標系への変換(ok)
      call Rotate_damp(Parameter_C.n_member,ac_member,rot_memb,Member,Point,Parameter_C)
c     write(damp_out,*) ' Rotate_damp Ok'
      end if
c――――――――――――――――――――――――――――――――★節点集中質量セット(ok)
      call Set_mass(Point,Parameter_C,am_point)
c     write(damp_out,*) ' Set_mass Ok'
```

太文字で示したこれらのサブルーチンは以下のように変更される。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Rotate_stiffness
C     ――――――――――――――――――――――――――――――――――――
C     ●   部材行列の釣合系への座標変換(ok)
C     ――――――――――――――――――――――――――――――――――――
      subroutine Rotate_stiffness(Parameter_C,ak_linear,rot_memb,Point,Member)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record /parameter_s  / Parameter_C
      record / point_s     / Point
      record / member_s    / Member
      dimension Point(*),Member(*)
      dimension bk(12,12),rot_memb(3,3,2,*)
      dimension ak_linear(12,12,*)
      dimension alx(2),aly(2)
c――――――――――――――――――――――――――――――――――――――――――――
c     n_member                  : integer   部材数
```

```
c     ak_member(12,12,n_member)   : real*8    線形剛性行列
c     rot_memb(12,12,2,*)         : real*8    座標変換行列
c-----------------------------------------------------------------
      n_member = Parameter_C.N_member
c      write(76,*) ' 座標変換後の剛性：',n_member
      if(Parameter_C.n_gouyuka .eq.0) then

      do i=1,n_member
      call Rotate_K(ak_linear(1,1,i),rot_memb(1,1,1,i),
     *              rot_memb(1,1,2,i),bK)
      do  j=1,12
      do  k=1,12
      ak_linear(j,k,i)=bk(j,k)
      end do
      end do
      end do

      else
      do i=1,n_member
      call Rotate_K(ak_linear(1,1,i),rot_memb(1,1,1,i),
     *              rot_memb(1,1,2,i),bK)
c-------------------------- 剛床座標変換
      n_gouyuka=0
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
       n_gouyuka=1
       aly(k)= Point(i1).coord_local(1)
        alx(k)= Point(i1).coord_local(2)
       else
        aly(k)=0.
        alx(k)=0.
       endif
      enddo
      if(n_gouyuka.ne.0) call Set_gtrans_k(bk,aly(1),alx(1), aly(2),alx(2) )
      do  j=1,12
      do  k=1,12
      ak_linear(j,k,i)=bk(j,k)
      end do
      end do
c      write(76,*) ' member: ',i
c     do j=1,12
c      write(76,'(i4,12e10.3)')j,(bk(j,k),k=1,12)
c     enddo
      end do

      endif
      return
      end
```

部材減衰行列の変換用サブルーチン Rotate_damp()は、以下のように変更する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Rotate_damp
C       ―――――――――――――――――――――――――――――――――
C     ●   部材減衰行列の釣合座標系への変換(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Rotate_damp(n_member,ac_member,rot_memb,Member,Point,Parameter_C)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      dimension bk(12,12),rot_memb(3,3,2,*)
      dimension ac_member(12,12,*)
      record /parameter_s  / Parameter_C
      record / point_s     / Point
      record / member_s / Member
      dimension Member(*),Point(*)
      dimension alx(2),aly(2)
C     ―――――――――――――――――――――――――――――――――
c     n_member                  : integer   部材数
c     ac_member(12,12,n_member)   : real*8    減衰剛性行列
c     rot_memb(12,12,2,*)         : real*8    座標変換行列
c     Member                    : structure
C     ―――――――――――――――――――――――――――――――――
      if(n_member .eq.0) return
      if(Parameter_C.n_gouyuka .eq.0) then

      do i=1,n_member
      ij=Member(i).nm_damp
      if(ij.ne.0) then
      call Rotate_K(ac_member(1,1,ij),rot_memb(1,1,1,i),
     *             rot_memb(1,1,2,i),bK)
      do  j=1,12
      do  k=1,12
      ac_member(j,k,ij)=bk(j,k)
      end do
      end do
      end if
      end do

      else
      do i=1,n_member
      ij=Member(i).nm_damp
      if(ij.ne.0) then
      call Rotate_K(ac_member(1,1,ij),rot_memb(1,1,1,i),
     *             rot_memb(1,1,2,i),bK)
c―――――――――――――――――――――――― 剛床座標変換
      n_gouyuka=0
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
       n_gouyuka=1
        aly(k)= Point(i1).coord_local(1)
        alx(k)= Point(i1).coord_local(2)
       else
        aly(k)=0.
        alx(k)=0.
```

```
         endif
        enddo
        if(n_gouyuka.ne.0) call Set_gtrans_k(bk,aly(1),alx(1), aly(2),alx(2) )
        do  j=1,12
        do  k=1,12
        ac_member(j,k,ij)=bk(j,k)
        end do
        end do
        endif
        end do

        endif
        return
        end
```

次は、ニューマークβ法における左辺係数行列を作成する部分を見てみよう。ます、主サブルーチンを以下に示す。

```
c―――――――――――――――――――――――――――――――――――――――――
c
c    ★        第１と第２段階解析の最初にスカイライン行列を作成、分解する
c             [F] = [M] + μ1[C] + μ2[K]
c―――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――――★左辺係数行列の計算(ok)
c――――――――――――――――――――――――――――――――――★ステップ番号のセット(ok)
      if(istep.eq.1.or.istep.eq.Newmark_P.n2_step) then
      n_istep=1
      if(istep.eq.Newmark_P.n2_step) n_istep=2
c――――――――――――――――――――――――――――――――――★スカイライン行列のゼロセット(ok)
      n_skyline=Parameter_C.n_skyline
      call Set_sky_zero(gskym,n_skyline)
c     write(damp_out,*) ' Set_sky_zero ok'
c――――――――――――――――――――――――――――――――――★集中質量系の行列への足し込み
c――――――――――――――――――――――――――――――――――★レーリー減衰を含む
      call Build_sky_mm(n_istep,gskym,
     *          Point,n_point, am_point , rot_local,
     *          n_local_coord ,Newmark_P, max_h_sky)
c     write(damp_out,*) ' Build_sky_mm ok'
c――――――――――――――――――――――――――――――――――★部材の整合質量系の行列への足し込み(ok)
c――――――――――――――――――――――――――――――――――★レーリー減衰を含む
c――――――――――――――――――――――――――――――――――★部材の整合質量行列計算(ok*)
      if(Dynamic_load.load_mass .ne. 0) then
      call Cal_mass_linear(n_istep,Element,Member,Parameter_C,am_member,
     *           work1_elememt,work2_elememt,work1_member,work2_member,
     *           Dynamic_load.load_mass)
c     write(damp_out,*) ' Cal_mass_linear ok'
c――――――――――――――――――――――――――――――――――★整合質量の釣合座標系への変換(ok*)
      call Rotate_mass(n_istep,Element,Member,n_member,am_member,
     *                rot_memb,Dynamic_load.load_mass,Point)
c      write(damp_out,*) ' Rotate_mass ok'
c――――――――――――――――――――――――――――――――――★整合質量系の足し込み(ok*)
      call Build_sky_m(n_istep,gskym,n_skyline, Member,n_member,
```

```
     *                 am_member ,Newmark_P, max_h_sky,Element)
      endif
c       write(damp_out,*) ' Build_sky_m ok'
c――――――――――――――――――――――――――――★部材減衰系の足し込み(ok)
c――――――――――――――――――――――――――――★Maxwell 線形減衰を含む
      if(Parameter_C.nc_member .ne. 0) then
       call Build_sky_c(gskym,Member,n_member,
     *                ac_member ,Newmark_P, max_h_sky)
c        write(damp_out,*) ' Build_sky_c ok'
      endif
c――――――――――――――――――――――――――――★線形剛性の足し込み(ok)
c――――――――――――――――――――――――――――★レーリー減衰を含む
      call Build_sky_k(n_istep,gskym,n_skyline, Member,n_member,
     *                      Ak_linear, Newmark_P, max_h_sky)
c       write(damp_out,*) ' Build_sky_k ok'
c――――――――――――――――――――――――――――――――――★行列の LDU 分解(ok)
      n_skyline=Parameter_C.n_skyline
      call decomp_sky(n_skyline,n_unknown,n_unknown,max_h_sky,
     *               gskym,gskym_d,nwork,twork,iexit)
c     write(damp_out,'(a,12f12.3)')' gskym *'
c――――――――――――――――――――――――――――――――――★分解成功か？
```

最初に、集中質量系の係数行列への組み込みを行うサブルーチンを変更しよう。このサブルーチンは、

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Build_sky_mm
C       ――――――――――――――――――――――――――――――――――
C     ●    集中質量行列のスカイライン行列への組み込み(ok)
C     ――――――――――――――――――――――――――――――――――
      subroutine Build_sky_mm(n_istep,gskym,
     *          Point,n_point, am_point , rot_local,
     *          n_local_coord ,Newmark_P, max_h_sky)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      dimension am_point(2,*),gskym(*)
      dimension max_h_sky(0:*),rot_local(3,3,*)
      record / newmark_s   / Newmark_P
      record / point_s     / Point
      dimension Point(*)
      dimension amm(6,6)
c――――――――――――――――――――――――――――――――――――――――――――
c     n_istep                 :integer   第一ステップか第二ステップか
c     gskym(n_skyline)        :real*8    スカイライン行列
c     Point                   :structure
c     n_point                 :integer   節点数
c     am_point(2,n_point)     :real*8   節点集中質量
c     rot_local(3,3, *)       :real*8   全体座標系から局所座標への回転行列
c     n_local_coord           :integer    局所座標系を用いている節点数
c     Newmark_P               :structure
c     max_h_sky(n_unknown+1)  :integer     スカイライン行列の各列の高さ
c
```

```
c     集中系では、座標変換は必要でない。
c
c―――――――――――――――――――――――――――――――――――――――――――――
      if(n_istep.eq.1) then
      ik=1
      par=1. + Newmark_P.ddt*Newmark_P.alf1_1
      else
      ik=2
      par=1. + Newmark_P.ddt*Newmark_P.alf2_1
      endif

      do i=1,n_point
      am=par*am_point(ik,i)
      if(Point(i).n_group_gouyuka.ne.0) then
      call Set_gtrans_m(am,amm, Point(i).coord_local(1), Point(i).coord_local(2))
      do j=1,3
      irest = Point(i).irest(j)
      if(irest.gt.0) then
       i3=max_h_sky(irest)
       gskym(i3)=gskym(i3)+am
      endif
      enddo
      irest = Point(i).irest(6)
      if(irest.gt.0) then
       i3=max_h_sky(irest)
       gskym(i3)=gskym(i3)+amm(6,6)
      endif
       i1= Point(i).irest(1)
       if(i1.gt.0) then
       i2= Point(i).irest(6)
       if(i2.gt.0.and.i2.le.i1) then
        i3=max_h_sky(i1)-(i1-i2)
        gskym(i3)=gskym(i3)+amk(1,6)
       elseif(i2.gt.0.and.i1.le.i2) then
        i3=max_h_sky(i2)-(i2-i1)
        gskym(i3)=gskym(i3)+amk(1,6)
      endif
      endif
       i1= Point(i).irest(2)
       if(i1.gt.0) then
        i2= Point(i).irest(6)
        if(i2.gt.0.and.i2.le.i1) then
        i3=max_h_sky(i1)-(i1-i2)
        gskym(i3)=gskym(i3)+amk(2,6)
       elseif(i2.gt.0.and.i1.le.i2) then
        i3=max_h_sky(i2)-(i2-i1)
        gskym(i3)=gskym(i3)+amk(2,6)
      endif
      endif
      else
      do j=1,3
      irest = Point(i).irest(j)
      if(irest.gt.0) then
       i3=max_h_sky(irest)
```

```
      gskym(i3)=gskym(i3)+am
      endif
      enddo
      endif
      end do
      return
      end
```

次は、整合質量行列を座標変換するサブルーチンを変更する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Rotate_mass
C       ―――――――――――――――――――――――――――――――――
C     ●   部材質量行列の釣合系への座標変換(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Rotate_mass(nx,Element,Member,n_member,
     *                       am_member,rot_memb,load_mass,Point)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s    / Member
      record / element_s   / Element
      record / point_s     / Point
      dimension Point(*),alx(2),aly(2)
      dimension Member(*),Element(*)
      dimension bk(12,12),rot_memb(3,3,2,*)
      dimension am_member(12,12,*)
C     ―――――――――――――――――――――――――――――――――
c     n_member                   : integer   部材数
c     am_member(12,12,n_member)  : real*8    質量整合行列
c     rot_memb(12,12,2,*)        : real*8    座標変換行列
C     ―――――――――――――――――――――――――――――――――
      if(load_mass .eq. 0 ) return
c――――――――――――――――――――――――――――――――★質量行列の計算
      if(nx.eq.1) then
      ik=1
      else
      ik=2
      endif
      do i=1,n_member
      ie = Member(i).nm_element
      if(Element(ie).am(ik) .ne. 0.0) then
      call Rotate_K(am_member(1,1,i),rot_memb(1,1,1,i),
     *              rot_memb(1,1,2,i),bK)
c――――――――――――――――――――――――― 剛床座標変換
      n_gouyuka=0
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
       n_gouyuka=1
       aly(k)= Point(i1).coord_local(1)
       alx(k)= Point(i1).coord_local(2)
       else
       aly(k)=0.
```

```
        alx(k)=0.
        endif
       enddo
       if(n_gouyuka.ne.0) call Set_gtrans_k(bk,aly(1),alx(1),
      *                                     aly(2),alx(2) )
       do  j=1,12
       do  k=1,12
       am_member(j,k,i)=bk(j,k)
       end do
       end do
       endif
       end do
       return
       end
```

最後に、接線剛性を計算するサブルーチン Get_nonlinear_stiff()であるが、これは静的解析で変更部分を既に示しておいた。これをそのまま使用することになる。このサブルーチンコールは、

```
c――――――――――――――――――――――――――――――★接線剛性の計算(ok)
      call Get_nonlinear_stiff(Control.type_analysis,Point,Parameter_C,
     *     ak_nonlinear,Member,n_member,
     *     Model_type,Element,past_disp_point,disp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
     *     E_model21, M_model21,
     *     E_model22, M_model22,
     *     E_model31, M_model31,
     *     E_model32, M_model32,
     *     E_model33, M_model33,
     *     MSS_work,S_comp_model,E_modelx, M_modelx,
     *     E_fiber_work, M_fiber_work,
     *     work1_elememt,work2_elememt, work1_member, work2_member)
```

であり、サブルーチン自身は以下のようである。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_nonlinear_stiff
C     ――――――――――――――――――――――――――――――――――――
C     ●   接線剛性行列の計算(ok)
C     ――――――――――――――――――――――――――――――――――――
      subroutine Get_nonlinear_stiff(N_analysis,Point,Parameter_C,
     *     ak_nonlinear,Member,n_member,
     *     Model_type,Element,past_disp_point,disp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
```

```
      *     E_model15, M_model15,
      *     E_model21, M_model21,
      *     E_model22, M_model22,
      *     E_model31, M_model31,
      *     E_model32, M_model32,
      *     E_model33, M_model33,
      *     MSS_work,S_comp_model,E_modelx, M_modelx,
      *     E_fiber_work, M_fiber_work,
      *     work1_elememt,work2_elememt, work1_member, work2_member)
       implicit real*8(A-H,O-Z)
       include "submain.h"
       include "submainx.h"
       include "New_submain.h"
       record / member_s    / Member
      record / point_s     / Point
      record /parameter_s  / Parameter_C
       record / element_s   / Element
       record / n_model_s   / Model_type
c―――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
       record / S_comp_model_s    / S_comp_model
       record / E_modelx_s        / E_modelx
       record / M_modelx_s        / M_modelx
       record / E_fiber_work_s    / E_fiber_work
       record / M_fiber_work_s    / M_fiber_work
       dimension E_modelx(*),M_modelx(*),S_comp_model(*)
       dimension E_fiber_work(*),M_fiber_work (*)
      dimension Point(*),aly(2),alx(2)

        •
        •
  100 continue
c―――――――――――――――――――――――――――――――――★部材の接線剛性を釣合系に変換

       call Rotate_K(ak,rot_memb(1,1,1,i),
     *             rot_memb(1,1,2,i),ak_nonlinear(1,1,i))
c―――――――――――――――――――――――――――――――――★剛床の変換
       if(Parameter_C.n_gouyuka .ne.0) then
       n_gouyuka=0
        do k=1,2
        i1=Member(i).nm_point(k)
        if(Point(i1).n_group_gouyuka.ne.0) then
        n_gouyuka=1
         aly(k)= Point(i1).coord_local(1)
         alx(k)= Point(i1).coord_local(2)
        else
         aly(k)=0.
         alx(k)=0.
        endif
       enddo
       if(n_gouyuka.ne.0) call Set_gtrans_k(ak_nonlinear(1,1,i),aly(1),alx(1), aly(2),alx(2) )
       endif
```

9.5.10.2 変位ベクトルの変換

本節では、動的解析において、釣合座標系から部材座標系に変換する場合について説明する。この場合、まず剛床に関する変換を行い、その後、全体座標系から部材座標系に変換しなければならない。ここでは、SPACE で使用している多くの座標変換部分を取り出し、プログラムに変更を加えることになる。変位に関連して変更するサブルーチンは、以下のようである。

```
Set_preset_disp()
Cal_stress()
Check_stress()
Out_disp_vel_acc()
```

ここでは、これらのサブルーチンの変更部分を示すことにする。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Set_preset_disp
C     ―――――――――――――――――――――――――――――――――
C     ●   節点の変位、速度、加速度を出力(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Set_preset_disp(ihan,n_point,past_disp_point,
     *                           F_disp,Point,rot_local,Parameter_C)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / point_s  / Point
      record / parameter_s /Parameter_C
      dimension Point(*)
      dimension past_disp_point(*)
      dimension rot_local(3,3,*),v(6),vv(6)
      real*4    F_disp(3,*)
      if(ihan.ne.0) goto 900
      do i=1,n_point
      do j=1,3
      F_disp(j,i)=0.
      enddo
      enddo
      return
  900 continue
c―――――――――――――――――――――――――――――――――――――★局所座標系なし
      if(Parameter_C.n_local_coord.eq.0 .and.
     * Parameter_C.n_gouyuka .eq.0) then
c―――――――――――――――――――――――――――――――――――――――★変位 3
      do i=1,n_point
      do j=1,3
      ires= Point(i).irest(j)
      F_disp(j,i)=0.
      if(ires.ne.0) F_disp(j,i) = past_disp_point(ires)
      end do
```

```
c     write(76,'(a,i4,6f12.3)') ' set:',i,(F_disp(j,i),j=1,3)
      end do

c――――――――――――――――――――――――――――――――★座標変換あり
      else
c――――――――――――――――――――――――――――――――★変位 3
      do i=1,n_point
      ij=Point(i).local_coord
      if(ij.ne.0) then
      do j=1,3
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j ) = past_disp_point(ires)
      end do
      call trans_VT(v,vv,rot_local(1,1,ij))
      do j=1,3
      F_disp(j,i)=vv(j)
      enddo
c――――――――――――――――――――――――――――――――★座標変換なし
      elseif(Point(i).n_group_gouyuka.eq.0) then
c――――――――――――――――――――――――――――――――★変位 3
      do j=1,3
      ires= Point(i).irest(j)
      F_disp(j,i)=0.
      if(ires.ne.0) F_disp(j,i) = past_disp_point(ires)
      end do
c――――――――――――――――――――――――――――――――★剛床座標系あり
      else
      do j=1,3
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j ) = past_disp_point(ires)
      end do
      j=6
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j ) = past_disp_point(ires)
      call Set_gtrans_u(1,v,Point(i).coord_local(1),Point(i).coord_local(2))
      do j=1,3
      F_disp(j,i)=v(j)
      enddo

      endif
      end do

      endif
      return
      end
```

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stress
C     ――――――――――――――――――――――――――――――――――――
C     ●   部材内応力の計算(ok)
```

```
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_stress(Member,Point,n_member,Model_type,Element,
     *     past_disp_point,disp_point,rot_memb,
     *     E_model6_real,ak_nonlinear,N_analysis)
C     ―――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / point_s     / Point
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / E_model6_real_s   / E_model6_real
c―――――――――――――――――――――――――――――――――――――――
      dimension Member(*),Element(*),E_model6_real(*)
      dimension Point(*)
      dimension rot_memb(3,3,2,*),ak_nonlinear(12,12,*)
      dimension past_disp_point(*),disp_point(*),
     *          v(12),vv(12),vp(12),vpp(12),ak(12,12),vx(12),vpx(12)
       .
       .
      do i=1,n_member
c     write(76,'(a,i4)') ' mem: z',i
c―――――――――――――――――――――――――――――――――――――――★部材両端の変位取得
      do j=1,12
      ires=Member(i).irest(j)
      if(ires.gt.0) then
      vp(j)=past_disp_point(ires)
      v(j)=disp_point(ires)
      vpx(j)=vp(j)
      vx(j)=v(j)
      else
      v(j)=0.
      vp(j)=0.
      vpx(j)=0.
      vx(j)=0.
      endif
      enddo
c      write(76,'(a,12f12.6)')' v ',(v(j),j=1,12)
c      write(76,'(/i4,12f12.4)') i,(Member(i).stress(j),j=1,12)
c―――――――――――――――――――――――――――――――――――――――★変位を釣合系から部材座標系に変換
c―――――――――――――――――――――――★剛床座標変換
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
       k1=6*(k-1)+1
       call Set_gtrans_u(1,vx(k1),Point(i1).coord_local(1),
     *                                Point(i1).coord_local(2))
       call Set_gtrans_u(1,vpx(k1),Point(i1).coord_local(1),
     *                                Point(i1).coord_local(2))
       endif
      enddo
c―――――――――――――――――――――――★座標変換
      call RotateL_v(1,vx,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vv)
```

```
      call RotateL_v(1,vpx,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vpp
c―――――――――――――――――――――――――――――――――★要素及びモデルのセット
      mem = i
      iet  = Member(i).element_type
      iett=(iet-1)/10
      ie = Member(i).nm_element
      im = Element(ie).n_element
      imm = Member(i).n_element_type
      ien= Member(i).n_model_type
      if(Member(i).nm_dll_element .ne. 0) goto 9999   ! DLL 要素
      if(iett.eq.0)then
      goto(11,12,13,14,15,16,17,18,19,20),iet
   11 continue
c―――――――――――――――――――――――――――――――――★Model_No.1 通常の有限要素弾塑性モデル
      call Cal_stress_M1(Member(i),Element(ie),
     *     ak_nonlinear(1,1,i),v,rot_memb(1,1,1,i),rot_memb(1,1,2,i),
     *      Point)
      goto 100
   12 continue
c―――――――――――――――――――――――――――――――――★Model_No.2 3 次元せん断弾塑性モデル
      call Cal_stress_M2(Member(i),Element(ie),vv)
      goto 100
   13 continue
c―――――――――――――――――――――――――――――――――★Model_No.3 3 次元軸力弾塑性モデル
      call Cal_stress_M3(Member(i),Element(ie),
     *     ak_nonlinear(1,1,i),v,vv,vvp,
     *     rot_memb(1,1,1,i),rot_memb(1,1,2,i),
     *      Point)
      goto 100
   14 continue
       .
       .
```

上記のサブルーチン Cal_stress()中で、各部材モデルの応力を計算するサブルーチンにおいて、次のように座標変換する必要がある。代表的なサブルーチンを以下に示すが、他のサブルーチンにおいても、同様に変更する必要がある。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stress_M1
C       ―――――――――――――――――――――――――――――――――
C     ●   部材の応力計算(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_stress_M1(Member,Element,ak,vv,r1,r2,Point)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s    / Member
      record / element_s   / Element
      record / point_s     / Point
      dimension Point(*)
      dimension ak(12,12),vv(12),r1(3,3),r2(3,3),st(12),ss(12)
```

```
      do i=1,12
      s=0.
      do j=1,12
      s=s+ak(i,j)*vv(j)
      enddo
      st(i)=s
      enddo
c―――――――――――――――――――――――――――――――★全体座標から部材座標へ変換
c―――――――――――――――――――★剛床座標変換
       do k=1,2
        i1=Member.nm_point(k)
        if(Point(i1).n_group_gouyuka.ne.0) then
        k1=6*(k-1)+1
        call Set_gtrans_f(1,st(k1),Point(i1).coord_local(1),
     *                                Point(i1).coord_local(2))
        endif
      enddo
c――――――――――――――――――★全体座標から部材座標
      call RotateL_v(1,st,r1,r2,ss)
      do i=1,6
      Member.stress(i)=-ss(i)+Member.stress(i)
      enddo
      do i=7,12
      Member.stress(i)=ss(i)+Member.stress(i)
      enddo
c     write(76,'(6f12.2)') (Member.stress(j),j=1,6)
c
c―――――――――――――――――――――――――――――――★非線形応力の計算(ok)
c
c      ann=ak(1,1)/Member.alength*( (vp(8)-vp(2))*(vv(8)-vv(2))+
c     *           (vp(9)-vp(3))*(vv(9)-vv(3)) )
c     Member.stress(1)=Member.stress(1)+ ann
c     Member.stress(7)=Member.stress(7)+ ann

      return
      end
```

次に、サブルーチン Check_stress()の変更点は以下のようである。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Check_stress
C     ―――――――――――――――――――――――――――――――――
C     ●  部材の塑性状態をチェックする(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Check_stress(Control,N_analysis, Point,
     *     ak_nonlinear,Member,n_member,
     *     Model_type,Element,past_disp_point,disp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
```

```
      *     E_model21, M_model21,
      *     E_model22, M_model22,
      *     E_model31, M_model31,
      *     E_model32, M_model32,
      *     E_model33, M_model33,
      *     MSS_work,
      *     Bilinear_work,Trilinear_work,Concrete_work,RO_work,
      *     work1_elememt,work2_elememt, work1_member, work2_member,
      *     S_comp_model,E_modelx,M_modelx,
      *     E_fiber_work,M_fiber_work)
       implicit real*8(A-H,O-Z)
       include "submain.h"
       include "submainx.h"
       include "New_submain.h"
       record /control_s    / Control
       record / member_s    / Member
       record / point_s     / Point
       record / element_s   / Element
       record / n_model_s   / Model_type
       record / Bilinear_work_s    / Bilinear_work
       record / Trilinear_work_s   / Trilinear_work
       record / Concrete_work_s    / Concrete_work
c――――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
       record / S_comp_model_s    / S_comp_model
       record / E_modelx_s        / E_modelx
       record / M_modelx_s        / M_modelx
       record / E_fiber_work_s    / E_fiber_work
       record / M_fiber_work_s    / M_fiber_work
       dimension E_modelx(*),M_modelx(*),S_comp_model(*)
       dimension E_fiber_work(*),M_fiber_work (*)
       dimension Point(*)
       dimension vx(12),vpx(12)
        .
        .
       do i=1,n_member
c      write(76,'(a,2i4)') ' memb : ',i,Member(i).element_type
c       kk=10
c       write(76,'(a,i4,3f12.3)') 'non c',kk,Member(kk).stress(7),
c      *  Member(kk).stress(8)
c――――――――――――――――――――――――――――――――――★部材両端の変位取得
       do j=1,12
       ires=Member(i).irest(j)
       if(ires.gt.0) then
       v(j)=disp_point(ires)
       vp(j)=past_disp_point(ires)
       vx(j)=v(j)
       vpx(j)=vp(j)
       else
       v(j)=0.
       vp(j)=0.
       vx(j)=0.
       vpx(j)=0.
       endif
       enddo
```

```
c         write(76,'(a,i3,12e12.5)') 'v',i,(v(j),j=1,12)
c        write(76,'(a,i3,12e12.5)') 'vp',i,(vp(j),j=1,12)
c－－－－－－－－－－－－－－－－－－－－－－－－－－－★部材両端の節点力のゼロセット
     do j=1,12
     f(j)=0.
     enddo
c－－－－－－－－－－－－－－－－－－－－－－－－－－－★変位を釣合系から部材座標系に変換
c－－－－－－－－－－－－－－－－★剛床座標変換
      do k=1,2
      i1=Member(i).nm_point(k)
      if(Point(i1).n_group_gouyuka.ne.0) then
      k1=6*(k-1)+1
      call Set_gtrans_u(1,vx(k1),Point(i1).coord_local(1),Point(i1).coord_local(2))
      call Set_gtrans_u(1,vpx(k1),Point(i1).coord_local(1),Point(i1).coord_local(2))
      endif
     enddo
c－－－－－－－－－－－－－－－－★座標変換
     call RotateL_v(1,vx,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vv)
     call RotateL_v(1,vpx,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vpp)
c         write(76,'(a,i3,12e12.5)') 'vv',i,(vv(j),j=1,12)
c        write(76,'(a,i3,12e12.5)') 'vpp',i,(vpp(j),j=1,12)
      ・
      ・
```

```
C     －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
C     ●    SUBROUTINE /Out_disp_vel_acc
C       －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
C     ●    節点の変位、速度、加速度を出力(ok)
C     －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
      subroutine Out_disp_vel_acc(Point,n_point,Parameter_C,
     *     past_disp_point, past_vel_point, past_acc_point,
     *     rot_local,ifl,iflz,vacc,i_print)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / point_s     / Point
      record / parameter_s /Parameter_C
      dimension Point(*)
      dimension past_disp_point(*),past_vel_point(*),past_acc_point(*)
      dimension rot_local(3,3,*),vacc(6),vv(6),vvv(6)
      dimension ifl(16),iflz(16)
      real*4    v(6)
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            :real*8 地震加速度
c     i_print         :integer 出力制御変数 0:ファイル出力あり
C     －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
      if(i_print.ne.0) return
```

```
c      write(76,*) ' 変位：', n_point
c――――――――――――――――――――――――――――――――――★局所座標系なし
      if(Parameter_C.n_local_coord.eq.0 .and.
     *          Parameter_C.n_gouyuka .eq.0) then
c――――――――――――――――――――――――――――――――――★変位 3
      do i=1,n_point
      do j=1,3
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j) = past_disp_point(ires)
      end do
      if(ifl(3).eq.1) write(iflz(3)) (v(j),j=1,3)
c     write(76,'(i4,3e12.4,2i4)') i,(v(j),j=1,3)
      end do
c――――――――――――――――――――――――――――――――――★速度 11
c      write(76,*) ' 速度：', n_point
      do i=1,n_point
      do j=1,3
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j) = past_vel_point(ires)
      end do
      if(ifl(11).eq.1) write(iflz(11)) (v(j),j=1,3)

c     write(76,'(i4,6e12.4)') i,(v(j),j=1,3)
      end do
c――――――――――――――――――――――――――――――――――★相対加速度 10
c――――――――――――――――――――――――――――――――――★絶対加速度 13
c      write(76,*) ' 加速度：', n_point
      do i=1,n_point
      do j=1,3
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j) = past_acc_point(ires)
      end do
      if(ifl(10).eq.1) write(iflz(10)) (v(j),j=1,3)
c     write(76,'(i4,6e12.4)') i,(v(j),j=1,3)
      do j=1,3
      v(j)=v(j)+vacc(j)
      enddo
      if(ifl(13).eq.1) write(iflz(13)) (v(j),j=1,3)
c     write(76,'(i4,6e12.4)') i,(v(j),j=1,3)
      end do
c――――――――――――――――――――――――――――――――――★座標変換あり
      else
c――――――――――――――――――――――――――――――――――★変位 3
      do i=1,n_point
      ij=Point(i).local_coord
      if(ij.ne.0) then
c――――――――――――――――――――――――――――――――――★局所座標系あり
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_disp_point(ires)
```

```
      end do
      call trans_VT(vv,vvv,rot_local(1,1,ij))
      do j=1,3
      v(j)=vvv(j)
      enddo
c―――――――――――――――――――――――――――――――★座標変換なし
      elseif(Point(i).n_group_gouyuka .eq.0) then
      do j=1,3
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j) = past_disp_point(ires)
      end do
      else
c―――――――――――――――――――――――――――――――★剛床座標系あり
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_disp_point(ires)
      end do
      j=6
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_disp_point(ires)
      call Set_gtrans_u(1,vv,Point(i).coord_local(1),
     *                       Point(i).coord_local(2))
      do j=1,3
      v(j)=vv(j)
      enddo
      endif
      if(ifl(3).eq.1) write(iflz(3)) (v(j),j=1,3)
      end do
c―――――――――――――――――――――――――――――――★速度 11
      do i=1,n_point
      ij=Point(i).local_coord
      if(ij.ne.0) then
c―――――――――――――――――――――――――――――――★局所座標系あり
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_vel_point(ires)
      enddo
      call trans_VT(vv,vvv,rot_local(1,1,ij))
      do j=1,3
      v(j)=vvv(j)
      enddo
c―――――――――――――――――――――――――――――――★座標変換なし
      elseif(Point(i).n_group_gouyuka .eq.0) then
      do j=1,3
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j) = past_vel_point(ires)
      end do
      else
c―――――――――――――――――――――――――――――――★剛床座標系あり
```

```
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_vel_point(ires)
      enddo
      j=6
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_vel_point(ires)
      call Set_gtrans_u(1,vv,Point(i).coord_local(1),
     *                       Point(i).coord_local(2))
      do j=1,3
      v(j)=vv(j)
      enddo
      endif
      if(ifl(11).eq.1) write(iflz(11)) (v(j),j=1,3)
      end do
c―――――――――――――――――――――――――――――★相対加速度 10
c―――――――――――――――――――――――――――――★絶対加速度 13
      do i=1,n_point
      ij=Point(i).local_coord
      if(ij.ne.0) then
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_acc_point(ires)
      end do
      call trans_VT(vv,vvv,rot_local(1,1,ij))
      do j=1,3
      v(j)=vvv(j)
      enddo
      if(ifl(10).eq.1) write(iflz(10)) (v(j),j=1,3)
      do j=1,3
      v(j)=vvv(j)+vacc(j)
      enddo
c―――――――――――――――――――――――――――――★座標変換なし
      elseif(Point(i).n_group_gouyuka .eq.0) then
      do j=1,3
      ires= Point(i).irest(j)
      v(j)=0.
      if(ires.ne.0) v(j) = past_acc_point(ires)
      end do
      if(ifl(10).eq.1) write(iflz(10)) (v(j),j=1,3)
      do j=1,3
      v(j)=v(j)+vacc(j)
      enddo
      else
c―――――――――――――――――――――――――――――★剛床座標系あり
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_acc_point(ires)
      end do
      j=6
```

```
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_acc_point(ires)
      call Set_gtrans_u(1,vv,Point(i).coord_local(1),
     *                       Point(i).coord_local(2))
      do j=1,3
      v(j)=vv(j)
      enddo
      if(ifl(10).eq.1) write(iflz(10)) (v(j),j=1,3)
      do j=1,3
      v(j)=vv(j)+vacc(j)
      enddo
      endif
      if(ifl(13).eq.1) write(iflz(13)) (v(j),j=1,3)
      end do
      endif
      return
      end
```

```
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,rot_local)
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(6)
      dimension past_disp_point(*),past_vel_point(*),past_acc_point(*)
      dimension vv(6),vvv(6)
      dimension rot_local(3,3,*)
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
c――――――――――――――――――――――――――――――――――★変位最大値
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_disp_point(ires)
```

```
      enddo

      ij=Point(i).local_coord
      if(ij.ne.0) then
c―――――――――――――――――――――――――――――――――★局所座標系あり
      call trans_VT(vv,vvv,rot_local(1,1,ij))
      do j=1,3
      vv(j)=vvv(j)
      enddo
c―――――――――――――――――――――――――――――――――★座標変換なし
      elseif(Point(i).n_group_gouyuka .ne.0) then
c―――――――――――――――――――――――――――――――――★剛床座標系あり
      j=6
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_disp_point(ires)
      call Set_gtrans_u(1,vv,Point(i).coord_local(1),
     *                       Point(i).coord_local(2))
      endif
c―――――――――――――――――――――――――――――――――★最大値チェック
      do j=1,3
      aa = vv(j)
      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
      if(Max_disp(i).disp_point(j+3).gt.aa)
     *                     Max_disp(i).disp_point(j+3)=aa
      enddo
c―――――――――――――――――――――――――――――――――★速度最大値
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_vel_point(ires)
      enddo

      ij=Point(i).local_coord
      if(ij.ne.0) then
c―――――――――――――――――――――――――――――――――★局所座標系あり
      call trans_VT(vv,vvv,rot_local(1,1,ij))
      do j=1,3
      vv(j)=vvv(j)
      enddo
c―――――――――――――――――――――――――――――――――★座標変換なし
      elseif(Point(i).n_group_gouyuka .ne.0) then
c―――――――――――――――――――――――――――――――――★剛床座標系あり
      j=6
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_vel_point(ires)
      call Set_gtrans_u(1,vv,Point(i).coord_local(1),
     *                       Point(i).coord_local(2))
```

```
      endif
c―――――――――――――――――――――――――――――――――★最大値チェック
      do j=1,3
      aa = vv(j)
      if(Max_disp(i).vel_point(j).lt.aa) Max_disp(i).vel_point(j)=aa
      if(Max_disp(i).vel_point(j+3).gt.aa) Max_disp(i).vel_point(j+3)=aa
      enddo
c―――――――――――――――――――――――――――――――――★加速度最大値
      do j=1,3
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_acc_point(ires)
      enddo

      ij=Point(i).local_coord
      if(ij.ne.0) then
c―――――――――――――――――――――――――――――――――★局所座標系あり
      call trans_VT(vv,vvv,rot_local(1,1,ij))
      do j=1,3
      vv(j)=vvv(j)
      enddo
c―――――――――――――――――――――――――――――――――★座標変換なし
      elseif(Point(i).n_group_gouyuka .ne.0) then
c―――――――――――――――――――――――――――――――――★剛床座標系あり
      j=6
      ires= Point(i).irest(j)
      vv(j)=0.
      if(ires.ne.0) vv(j) = past_acc_point(ires)
      call Set_gtrans_u(1,vv,Point(i).coord_local(1),
     *                       Point(i).coord_local(2))
      endif
c―――――――――――――――――――――――――――――――――★最大値チェック
      do j=1,3
      aa=vv(j)
      if(Max_disp(i).acc_point(j).lt.aa) Max_disp(i).acc_point(j)=aa
      if(Max_disp(i).acc_point(j+3).gt.aa) Max_disp(i).acc_point(j+3)=aa
      enddo

c―――――――――――――――――――――――――――――――――★絶対加速度最大値
チェック
      do j=1,3
      aa = vv(j)+vacc(j)
      if(vv(j).ne.0.) then
      if(Max_disp(i).ab_acc_point(j).lt.aa)
     *     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)
     *     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
      end if
```

```
      end do
      end do

      return
      end
```

9.5.10.3 応力と節点力ベクトルの変換

最初に、静的荷重を設定するサブルーチンを変更する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Set_point_load
C       ―――――――――――――――――――――――――――――――――
C     ●   節点荷重データを釣合系の座標に変換し、セットする(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Set_point_load(fll_static_point,Parameter_C,
     *                Dynamic_load,Point,fld_static,rot_local)
C     ―――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / parameter_s    / Parameter_C
      record /Dynamic_load_s/ Dynamic_load
      record / point_s        / Point
      dimension Point(*),fll_static_point(3,6,*),fld_static(3,*)
      dimension rot_local(3,3,*),p(3),q(3) ,pp(6)
       •
       •
      do i=1,3
      if(Dynamic_load.load_point(i) .ne. 0) then
      do j=1,n_point
      if(Point(j).local_coord .eq. 0 .and.
     *  Point(j).n_group_gouyuka.eq.0) then
      do k=1,6
      ires=Point(j).irest(k)
      if(ires .ne. 0)
     *  fld_static(i,ires) = fld_static(i,ires)+fll_static_point(i,k,j)
      end do
c―――――――――――――――――――――――――――――★局所座標
      elseif(Point(j).local_coord .ne. 0 )then
      do k=1,3
      p(k) = fll_static_point(i,k,j)
      end do
      ip=Point(j).local_coord
      call trans_V(p,q,rot_local(1,1,ip))
      do k=1,3
      ires=Point(j).irest(k)
      if(ires .gt. 0)
     *  fld_static(i,ires) = fld_static(i,ires)+q(k)
      end do

      do k=1,3
      p(k) = fll_static_point(i,k+3,j)
```

```
      end do
      ip=Point(j).local_coord
      call trans_V(p,q,rot_local(1,1,ip))
      do k=1,3
      ires=Point(j).irest(k+3)
      if(ires .gt. 0)
     *  fld_static(i,ires) = fld_static(i,ires)+q(k)
      end do
c-----------------------------★剛床座標
      else
      do k=1,6
      pp(k) = fll_static_point(i,k,j)
      end do
      call Set_gtrans_f(2,pp,
     *     Point(j).coord_local(1),Point(j).coord_local(2))
      do k=1,6
      ires=Point(j).irest(k)
      if(ires .ne. 0)
     *  fld_static(i,ires) = fld_static(i,ires)+pp(k)
      end do
      endif
      end do

      endif
      end do
      write(76,'(//a,i4)') ' 静的荷重: ',Parameter_C.n_unknown
      write(76,*) 'load_point(1)',Dynamic_load.load_point(1)
      write(76,*) 'load_point(2)',Dynamic_load.load_point(2)
      write(76,*) 'load_point(3)',Dynamic_load.load_point(3)
      do j=1,Parameter_C.n_unknown
      write(76,'(i4,3f10.2)')j,(fld_static(i,j),i=1,3)
      end do
      return
      end
```

次に、右辺項ベクトルを作成する部分に関するサブルーチンを検討しよう。右辺項ベクトルは、

$$\{f\} = \{G\} + \{g\}$$
$$\{G(y_{n+1}, \Delta y_{n+1})\} = -\left\{\bar{f}_d\right\} - \left[K_T(y_n)\right]\{\Delta y_{n+1}\} + [K]\{y_{n+1}\}$$
$$\{g\} = -[M][I]\{\ddot{u}_g\} + \{P_S\} - \{Q(y_n)\} - \left[\bar{C}\right]\{a\} - [K]\left\{\bar{b}\right\}$$

である。ここで、再度、右辺項の変換規則について整理して置こう。なお、線形の剛性行列、接線剛性行列、整合質量行列、部材減衰行列は、剛床変換を含めた釣合座標系に変換された後、保存される。部材両端の節点力も釣合座標系に変換した後、保存される。静的荷重ベクトルも釣合座標系に変換された後、保存される。一方、集中質量は、対角行列であり、行列の形式で保存されていない。作成する場合は、全体座標系で

求める。

右辺項を求める場合、注意すべき点は慣性項である。質量行列に掛け算する地震加速度は、全体座標系で求められているため、集中質量系では、全体座標系の質量行列と掛け算した後、慣性項を釣合座標系に変換する。また、整合質量系は、既に釣合座標系に変換されているため、地震加速度ベクトルを全体座標系から釣合座標系に変換した後、掛け算して慣性項を求める。以下に、変換規則をまとめる。

１） Maxwell モデルの非線形項$\{\bar{f}_d\}$は、まず、釣合座標系の変位を全体座標系に変換した後、部材座標系に変換し、これよりモデルの非線形項を計算する。その後、非線形項を全体座標系に変換した後、剛床変換を用いて釣合座標系に変換する。
２） 接線剛性と線形剛性は、共に剛性行列を計算するときに釣合座標系に変換されており、このまま座標変換する必要はない。
３） 慣性項$[M][I]\{\ddot{u}_g\}$は、集中質量、整合質量共に全体座標系で求めておき、地震加速度と掛け算した後、慣性力を釣合座標系に変換する。
４） 擬似的静的荷重$\{\bar{f}_d\}$は、荷重ベクトルが既に釣合座標系に変換されているため、反復内では、座標変換を行う必要がない。
５） 不釣合い力$\{Q(y_n)\}$は、部材両端の節点力を求めた後、一端全体座標系に変換し、その後、剛床がある場合、釣合座標系に変換する。
６） 線形の剛性行列は、既に釣合座標系に変換されているため、そのまま、変位ベクトルと掛け算すればよい。
７） 質量に関連する減衰項は、集中質量系では、一端、$\{a\}$ベクトルを全体座標系に変換し、その後、質量行列と掛け算を行い、その結果を釣合座標系に変換する。整合質量行列は、既に釣合座標系に変換されているため、そのまま、$\{a\}$ベクトルと掛け算を行えば良い。

最初に、主サブルーチンの中で、右辺項ベクトルを求めるルーチンを以下に示す。

```
c―――――――――――――――――――――――――――――――――――――――
c
c    ★          動的解析（反復解法）
c
c―――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――★右辺の定数ベクトルゼロセット(ok)
      call Clear_vec(n_unknown,ld_point )
c        write(damp_out,*) ' Clear_vec ok'
c――――――――――――――――――――――――――――――★Work(a,b)ベクトルのセット(ok)
```

```
      call Set_a_b_vec(n_unknown,a_vector,b_vector,Newmark_P,
     *                 past_vel_point, past_acc_point)
c-----------------------------------★部材節点力のセット(ok)
      call Get_pointforce_ld(ld_point,Member,n_member)
c                   write(damp_out,'(a,6f16.5)') 'Get_pointforce_ld ok'
c      write(76,'(10e12.4)')(ld_point(i),i=1,n_unknown)
c-----------------------------------★地震加速度セット(ok)
      acc1=Get_Acc(T,1,acc_earth,Dynamic_load,Newmark_P.f1_T)
      acc2=Get_Acc(T,2,acc_earth,Dynamic_load,Newmark_P.f1_T)
      acc3=Get_Acc(T,3,acc_earth,Dynamic_load,Newmark_P.f1_T)
c      write(damp_out,'(a,4f10.3,i4)')'Get_Acc ok'
c-------------------------------★集中質量に関する慣性項
      call Add_earth1_ld(n_istep,acc1,acc2,acc3,ld_point,
     *          Point,n_point,am_point,rot_local,Parameter_C)
c         write(damp_out,*) ' Add_earth1_ld ok'
c-------------------------------★整合質量に関する慣性項
      call Add_earth2_ld(acc1,acc2,acc3,ld_point,
     *    Member,n_member,am_member,rot_local,Parameter_C,Dynamic_load)
c      write(damp_out,*) ' Add_earth2_ld ok'
c-----------------------------------★節点荷重のセット(ok)
      p1=Get_Ps(T,1,fdd_point,Dynamic_load)
      p2=Get_Ps(T,2,fdd_point,Dynamic_load)
      p3=Get_Ps(T,3,fdd_point,Dynamic_load)
c     write(damp_out,'(a,4f10.3)') ' Get_Ps ok'
      call Add_point_ld(p1,p2,p3,ld_point,n_unknown,
     *       Dynamic_load,fld_static)
c      write(damp_out,*) ' Add_point_ld ok'
c-----------------------------------★線形減衰項計算(ok)
c-------------------------------★集中質量(ok)
c      call Add_damp1_ld(n_istep,ld_point,Point,n_point,
c     *       past_disp_point,past_vel_point,past_acc_point,
c     *       am_point,Newmark_P,Parameter_C,rot_local)
      call Add_damp1_ld_ex(n_istep,ld_point,Point,n_point,
     *       a_vector,am_point,Newmark_P,Parameter_C,rot_local)
c     write(damp_out,*) ' Add_damp1_ld ok'
c-------------------------------★整合質量(ok)
c      call Add_damp2_ld(n_istep,ld_point,Member,n_member,
c     *       past_disp_point,past_vel_point,past_acc_point,
c     *       am_member,Newmark_P,Element,Dynamic_load.load_mass)
      call Add_damp2_ld_ex(n_istep,ld_point,Member,n_member,a_vector,
     *       am_member,Newmark_P,Element,Dynamic_load.load_mass)
c     write(damp_out,*) ' Add_damp2_ld ok'
c-------------------------------★部材減衰(ok)
c-------------------------------★Maxwell 線形減衰を含む
c      call Add_damp3_ld(n_istep,ld_point,Member,n_member,
c     *       past_disp_point,past_vel_point,past_acc_point,
c     *       ac_member,Newmark_P,Model_type.n_m_damp)
      call Add_damp3_ld_ex(n_istep,ld_point,Member,n_member,
     *       ac_member,a_vector,Element,rot_memb,E_model6_real,
     *       Newmark_P,Model_type.n_m_damp,Point)
c     write(damp_out,*) ' Add_damp3_ld ok'
c-----------------------------------★線形剛性項計算(ok)
c                                                  レーリー減衰も含む
      call Add_stiff1_ld(n_istep,ld_point,Member,n_member,
```

```
     *       past_disp_point,past_vel_point,past_acc_point,
     *       ak_linear,Newmark_P)
c      write(damp_out,*) ' Add_stiff1_ld ok'
c--------------------------------★線形剛性によるレーリー減衰（解析２）
c      call Add_stiff1_ld_ex(n_istep,ld_point,Member,n_member,
c     *       a_vector,ak_linear,Newmark_P)
c     write(damp_out,*) ' Add_stiff1_ld ok'
c--------------------------------★接線剛性に関する増分ベクトル（解析２）
c      call Add_tan_stiff_ld(ld_point,
c     *       Member,n_member,b_vector,ak_nonlinear)
c     write(damp_out,*) ' Add_stiff1_ld ok'
c--------------------------------★Δｔ秒後の変位と速度を予測(ok)
      n_err_roop = 0
 9991 continue
      nx =0
      call Estimate_disp_vel(nx, n_unknown,
     *       est_disp_point, est_vel_point, est_ddisp_point,
     *       past_disp_point, past_vel_point, past_acc_point,
     *       result_disp_point, result_vel_point,
     *       past_dacc_point,result_acc_point,Newmark_P)
c     write(damp_out,*) ' Estimate_disp_vel ok'
c-----------------------------------------------------------
c
c    ★        反復計算開始
c
c-----------------------------------------------------------
      n_roop=Newmark_P.max_repeat
      do iroop=1,n_roop
c     write(damp_out,*) ' 反復回数：' ,iroop
c--------------------------------★反復に関連する右辺ベクトルのゼロセット(ok)
      call Clear_vec(n_unknown,ld_point_repeat)
c         write(damp_out,*) ' Clear_vec ok'
c--------------------------------★線形剛性に関するベクトル(ok)
      call Add_stiff2_ld(ld_point_repeat,
     *       Member,n_member,est_disp_point,ak_linear)
c     write(damp_out,*) ' Add_stiff2_ld ok'
c--------------------------------★接線剛性に関する増分ベクトル(ok)
      call Add_tan_stiff_ld(ld_point_repeat,
     *       Member,n_member,est_ddisp_point,ak_nonlinear)
c     write(damp_out,*) ' Add_tan_stiff_ld ok'
c--------------------------------★線形剛性に関するベクトル(解析２)
c      call Add_stiff2x_ld(ld_point_repeat,
c     *       Member,n_member,result_acc_point,ak_linear,Newmark_P)
c     write(damp_out,*) ' Add_stiff2_ld ok'
c--------------------------------★接線剛性に関する増分ベクトル(解析２)
c      call Add_tan_stiff2_ld(ld_point_repeat,
c     *       Member,n_member,result_acc_point,ak_nonlinear,Newmark_P)
c     write(damp_out,*) ' Add_tan_stiff_ld ok'
c--------------------------------★Maxwell 型モデルの計算(ok)
      call Add_fdd_ld(ld_point_repeat,E_model6_real,Element,
     *       Member,n_member,est_vel_point,rot_memb)
c     write(damp_out,*) ' Add_fdd_ld ok'
c--------------------------------★右辺２項の和を取る(ok)
      call add_vec(n_unknown,ld_point_repeat,ld_point)
```

```
c     write(damp_out,*) ' add_vec ok'
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★線形方程式を解く(ok)
c      write(76,'(a)')' ld_point_repeat'
c      write(76,'(10e12.4')(ld_point_repeat(i),i=1,n_unknown)
      n_skyline=Parameter_C.n_skyline
      call solve_sky(n_skyline,n_unknown,n_unknown,
     *             max_h_sky,gskym,gskym_d,
     *             nwork,twork,ld_point_repeat,result_acc_point)
c      write(76,'(a)')' result_acc_point'
c      write(76,'(10e12.4')(result_acc_point(i),i=1,n_unknown)
c     write(damp_out,*) ' solve_sky ok'
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★β法に基づき加速度より変位と速度を計算(ok)
      call Cal_disp_vel(n_unknown,
     * result_disp_point, result_vel_point, result_acc_point,
     * past_disp_point, past_vel_point, past_acc_point,
     * Newmark_P)
c     write(damp_out,*) ' Cal_disp_vel ok'
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★収束したかチェック(ok)
      if(ICheck_error(iroop,n_point,Point,n_unknown,result_disp_point,
     *     est_disp_point, Newmark_P) .eq. 0) goto 9980
c     write(damp_out,*) ' Check_error ok'
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★次の増分値を予測(ok)
      nx=iroop
      call Estimate_disp_vel(nx, n_unknown,
     *       est_disp_point, est_vel_point, est_ddisp_point,
     *       past_disp_point, past_vel_point, past_acc_point,
     *       result_disp_point, result_vel_point,
     *       past_dacc_point,result_acc_point, Newmark_P)
c     write(damp_out,*) ' Estimate_disp_vel ok'
      end do
```

上記のルーチンの中で変更しなければならないサブルーチンは、以下の
ようである。

```
Get_pointforce_ld()
Add_earth1_ld()
Add_earth2_ld()
Add_damp1_ld_ex
Add_damp3_ld_ex
```

これらのサブルーチンを以下のように変更する。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_pointforce_ld
C       ――――――――――――――――――――――――――――――――――――
C     ●   部材節点力のセット(ok)
C     ――――――――――――――――――――――――――――――――――――
      subroutine Get_pointforce_ld(ld_point,Member,n_member,
     *           Parameter_C,Point)
```

```
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s   / Member
      record / point_s     / Point
      record /parameter_s  / Parameter_C
      dimension Member(*),Point(*)
      real*8 ld_point(*),ff(12)
C     ――――――――――――――――――――――――――――――――――――
c           ld_point      :real*8 右辺項
c           Member        :structure
c           n_member      :integer 部材数
C     ――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★座標変換なし
      if(Parameter_C.n_gouyuka .eq.0) then
      do i=1,n_member
c                       Maxwell  Model の応力は、他で考慮するのでここでは、無視する。
      if(Member(i).element_type.ne.6) then
      do j=1,12
      i1 = Member(i).irest(j)
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - Member(i).force(j)
      end do
      endif
c     write(76,'(i3,12e10.3)') i,(Member(i).force(j),j=1,12)
      end do
      else
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★剛床座標変換あり
      do i=1,n_member
      if(Member(i).element_type.ne.6) then
      n_gouyuka=0
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) n_gouyuka=1
       enddo
      if(n_gouyuka .ne. 0) then
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――― 剛床座標変換あり
      do j=1,12
      ff(j)=Member(i).force(j)
      enddo
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
        k1=6*(k-1)+1
        call Set_gtrans_f(2,ff(k1),
     *     Point(i1).coord_local(1),Point(i1).coord_local(2))
        endif
      enddo
      do j=1,12
      i1 = Member(i).irest(j)
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - ff(j)
      end do
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――― 剛床座標変換なし
      else
      do j=1,12
      i1 = Member(i).irest(j)
```

```
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - Member(i).force(j)
c     if(i1.gt.0) ld_point(i1)=ld_point(i1) + Member(i).force(j)
      end do
      endif

      endif
      end do
      endif
      return
      end
```

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Add_earth1_ld
C       ――――――――――――――――――――――――――――――――――――
C     ●   集中質量行列に関する地震荷重(ok)
C     ――――――――――――――――――――――――――――――――――――
      subroutine Add_earth1_ld(n_istep,acc1,acc2,acc3,ld_point,
     *           Point,n_point,am_point,rot_local,Parameter_C)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      dimension am_point(2,*)
      dimension rot_local(3,3,*)
      record / point_s     / Point
      record / parameter_s / Parameter_C
      dimension Point(*)
      dimension amk(6),amkk(3)
      real*8 ld_point(*)
c――――――――――――――――――――――――――――――――――――――――――――
c     n_istep                 :integer  第一ステップか第二ステップか
c     acc1,acc2,acc3          :real*8   ｘ、ｙ、ｚ方向の地震荷重
c     ld_point(*)             :real*8   右辺荷重項
c     Point                   :structure
c     n_point                 :integer  節点数
c     am_point(2,n_point)     :real*8   節点集中質量
c     rot_local(3,3, *)       :real*8   全体座標系から局所座標への回転行列
c     Parameter_C             :structure
c――――――――――――――――――――――――――――――――――――――――――――
      if(n_istep.eq.1) then
      ik=1
      else
      ik=2
      endif
c――――――――――――――――――――――――――――――――――――――――――――★座標変換あり
      if(Parameter_C.n_local_coord.ne.0
     *    .or. Parameter_C.n_gouyuka .ne. 0) then
      do i=1,n_point
      am=am_point(ik,i)
      amk(1)=am * acc1
      amk(2)=am * acc2
      amk(3)=am * acc3
c――――――――――――――――――――――――――――――――――――――――――――★局所座標系あり
      if(Point(i).local_coord.ne.0) then
      ij=Point(i).local_coord
```

```
      call RotateTLs_v(2,amk,rot_local(1,1,ij),amkk)
      do  j=1, 3
      irest=Point(i).irest(j)
      if(irest.gt.0) ld_point(irest)=ld_point(irest) - amkk(j)
      end do
c―――――――――――――――――――――――――――――――――★剛床座標系あり
      elseif(Point(i).n_group_gouyuka.ne.0) then
      amk(6)=0.
      call Set_gtrans_f(2,amk, Point(i).coord_local(1),
     *                         Point(i).coord_local(2) )
      do  j=1, 3
      irest=Point(i).irest(j)
      if(irest.gt.0) ld_point(irest)=ld_point(irest) - amk(j)
      end do
      irest=Point(i).irest(6)
      if(irest.gt.0) ld_point(irest)=ld_point(irest) - amk(6)
c―――――――――――――――――――――――――――――――――★座標変換なし
      else
      do  j=1,3
      irest =Point(i).irest(j)
      if(irest.gt.0) ld_point(irest)=ld_point(irest) - amk(j )
      end do
      endif
      end do

c―――――――――――――――――――――――――――――――――★座標変換なし
      else
      do i=1,n_point
      am=am_point(ik,i)
      amk(1)=am * acc1
      amk(2)=am * acc2
      amk(3)=am * acc3
      do  j=1, 3
      irest=Point(i).irest(j)
      if(irest.gt.0) ld_point(irest)=ld_point(irest) - amk(j)
      end do
      end do

      end if
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Add_earth2_ld
C       ―――――――――――――――――――――――――――――――――――――
C     ●   整合質量質量行列に関する地震荷重(ok)
C     ―――――――――――――――――――――――――――――――――――――
      subroutine Add_earth2_ld(Point,acc1,acc2,acc3,ld_point,
     *    Member,n_member,am_member,rot_local,Parameter_C,Dynamic_load)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      dimension am_member(12,12,*)
      dimension rot_local(3,3,*)
```

```
      record / member_s     / Member
      record / parameter_s / Parameter_C
      record / dynamic_load_s/ Dynamic_load
      record / point_s     / Point
      dimension Point(*)
      dimension Member(*)
      dimension amk(12),amkk(6),accx(3)
      real*8 ld_point(*)
c-----------------------------------------------------------------
c     n_istep                 :integer  第一ステップか第二ステップか
c     acc1,acc2,acc3          :real*8   ｘ、ｙ、ｚ方向の地震荷重
c     ld_point(*)             :real*8   右辺荷重項
c     Point                   :structure
c     n_point                 :integer  節点数
c     am_member               :real*8  節点整合質量
c     rot_local(3,3,*)        :real*8  全体座標系から局所座標への回転行列
c     Parameter_C             :structure
c-----------------------------------------------------------------
      if(Dynamic_load.load_mass .eq.0 ) return
      if(Parameter_C.n_local_coord.eq.0
     *    .and. Parameter_C.n_gouyuka .eq. 0) then
c-------------------------------★座標変換なし
      do i=1,n_member
      do j=1,12
      amk(j)=acc1*am_member(j,1,i)+acc1*am_member(j,7,i)
     *      +acc2*am_member(j,2,i)+acc2*am_member(j,8,i)
     *      +acc3*am_member(j,3,i)+acc3*am_member(j,9,i)
      enddo
      do  j=1, 12
      i1=Member(i).irest(j)
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - amk(j)
      end do
      end do
c-------------------------------★座標変換あり
      else
      accx(1)=acc1
      accx(2)=acc2
      accx(3)=acc3
      do i=1,n_member
      n_henkann=0
      do k=1,2
      i1= Member(i).nm_point(k)
      if(Member(i).nm_local_coord(k).ne.0 .or.
     *   Point(i1).nm_group_gouyuka .ne. 0) n_henkann=1
      enddo
c-------------------------------★座標変換なし
      if(n_henkann .eq. 0) then
      do j=1,12
      amk(j)=acc1*am_member(j,1,i)+acc1*am_member(j,7,i)
     *      +acc2*am_member(j,2,i)+acc2*am_member(j,8,i)
     *      +acc3*am_member(j,3,i)+acc3*am_member(j,9,i)
      enddo
      do  j=1, 12
      i1=Member(i).irest(j)
```

```
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - amk(j)
      end do

      else
c―――――――――――――――――――――――――――――――――★局所座標変換あり
      do j=1,12
      amk(j)=0.
      amkk(j)=0.
      enddo
      do j=1,3
      if(accx(j).ne.0.) then
      amkk(j)=accx(j)
      amkk(j+6)=accx(j)
      do k=1,2
      i1= Member(i).nm_point(k)
      if(Member(i).nm_local_coord(k).ne.0 )then
      ij= Member(i).nm_local_coord(k)
      k1=(k-1)*6+1
      call RotateTLs_vm(2,amk(k1),rot_local(1,1,ij))
c―――――――――――――――――――――――――――――――――★剛床座標系あり
      if(Point(i1).nm_group_gouyuka.ne.0) then
      k1=(k-1)*6+1
      call Set_gtrans_f(2,amkk(k1), Point(i1).coord_local(1), Point(i1).coord_local(2) )
      endif
      enddo
c―――――――――――――――――――――――――――――――――★質量行列と掛け算
      do k=1,12
      sum=0.
      do kk=1,12
      sum=sum+ am_member(k,kk,i)*amkk(kk)
      enddo
      amk(k)=amk(k)+sum
      enddo
      endif
      enddo
c―――――――――――――――――――――――――――――――――★荷重項へセット
      do  j=1, 12
      i1=Member(i).irest(j)
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - amk(j)
      end do
      endif
      enddo

      end if
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Add_damp1_ld_ex
C       ―――――――――――――――――――――――――――――――――――
C     ●   減衰・集中質量による減衰項のセット(ok)
C     ―――――――――――――――――――――――――――――――――――
      subroutine Add_damp1_ld_ex(nx,ld_point,Point,n_point,
```

```
     *       a_vector,am_point,Newmark_P,Parameter_C,rot_local)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / newmark_s   / Newmark_P
      record / parameter_s /Parameter_C
      record / point_s     /Point
      dimension Point(*)
      real*8 ld_point(*),am_point(2,*),rot_local(3,3,*)
      dimension u(6),amm(3),uu(6)
      dimension a_vector(*)
c-----------------------------------------------------------------
c     nx                      :integer  第一ステップか第二ステップか
c     ld_point(*)             :real*8   右辺荷重項
c     Point                   :structure
c     n_point                 :integer  節点数
c     a_vector                :real*8   a vactor
c     am_point(2,n_point)     :real*8   節点集中質量
c     Newmark_P               :structure
c     Parameter_C             :structure
c     rot_local(3,3,*)        :real*8   全体座標系から局所座標への回転行列
c-----------------------------------------------------------------
      if(nx.eq.1) then
      a= Newmark_P.alf1_1
      ik=1
      else
      a= Newmark_P.alf2_1
      ik=2
      endif
c-------------------------------★座標変換あり
      if(Parameter_C.n_local_coord.ne.0
     *    .or. Parameter_C.n_gouyuka .ne. 0) then
      do i=1,n_point
      am=am_point(ik,i)
      amm(1)=am
      amm(2)=am
      amm(3)=am
      if(am.ne.0.) then
      do j=1,3
      irest = Point(i).irest(j)
      if(irest.gt.0) then
      u(j)=a*a_vector(irest)
      else
      u(j)=0.0
      endif
      end do
c-------------------------------★局所座標系あり
      if(Point(i).local_coord.ne.0) then
      ij=Point(i).local_coord
c-------------------------------★局所座標変換は変換前の質量剛列と同一
      do j=1,3
      u(j)=amm(j)*u(j)
      enddo
       do  j=1, 3
      i1=Point(i).irest(j)
```

```
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - u(j)
      end do
c―――――――――――――――――――――――――――――――――★剛床座標系あり
      elseif(Point(i).n_group_gouyuka.ne.0) then
      j=6
      irest = Point(i).irest(j)
      if(irest.gt.0) then
      u(j)=a*a_vector(irest)
      else
      u(j)=0.0
      endif
      call Set_gtrans_u(1,u, Point(i).coord_local(1),
     *                       Point(i).coord_local(2) )
      do j=1,3
      uu(j)=amm(j)*u(j)
      enddo
      uu(6)=0.
      call Set_gtrans_f(2,uu, Point(i).coord_local(1),
     *                        Point(i).coord_local(2) )
       do  j=1, 3
      i1=Point(i).irest(j)
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - uu(j)
      end do
      i1=Point(i).irest(6)
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - uu(6)

c―――――――――――――――――――――――――――――――――★座標変換なし
      else
      do  j=1, 3
      i1=Point(i).irest(j)
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - amm(j)*u(j)
      end do
      endif
      endif
      end do
c―――――――――――――――――――――――――――――――――★座標変換なし
      else
      do i=1,n_point
      am=am_point(ik,i)
      amm(1)=am
      amm(2)=am
      amm(3)=am
      if(am.ne.0.) then
      do j=1,3
      irest = Point(i).irest(j)
      if(irest.ne.0) then
      u(j)=a*a_vector(irest)
      else
      u(j)=0.0
      endif
      end do
      do  j=1, 3
      i1=Point(i).irest(j)
      if(i1.gt.0) ld_point(i1)=ld_point(i1) - amm(j)*u(j)
```

```
      end do
      endif
      end do
c-----------------------------------------------------------------------
      endif
      return
      end
```

```
C     ------------------------------------------
C     ●   SUBROUTINE /Add_damp3_ld_ex
C       ----------------------------------------
C     ●   部材減衰による減衰項のセット(ok)
C     ------------------------------------------
      subroutine Add_damp3_ld_ex(nx,ld_point,Member,n_member,ac_member,
     *       a_vector,Element,rot_memb,E_model6_real,Newmark_P,n_damp,Point)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / newmark_s    /Newmark_P
      record / member_s     /Member
      record / point_s      /Point
      record / element_s    / Element
      record /E_model6_real_s / E_model6_real
      dimension Member(*),Element(*),Point(*)
      real*8 ld_point(*),ac_nonlinear(12,12),bk(12,12)
      dimension rot_memb(3,3,2,*),ac_member(12,12,*)
      dimension u(12),aly(2),alx(2)
      dimension a_vector(*)
      dimension E_model6_real(*)
c-----------------------------------------------------------------------
c     nx                      :integer  第一ステップか第二ステップか
c     ld_point(*)             :real*8   右辺荷重項
c     Point                   :structure
c     n_point                 :integer  節点数
c     a_vector                :real*8   a ベクトル
c     ac_point(2,n_point)     :real*8  部材減衰行列（釣合系）
c     Newmark_P               :structure
c     Parameter_C             :structure
c     n_damp                  :integer  部材減衰の部材がありか
c-----------------------------------------------------------------------
      if(n_damp.eq.0) return
c-----------------------------------------------------------------------
      do i=1,n_member
      ij = Member(i).nm_damp
      if(ij .ne. 0) then
      do j=1,12
      irest = Member(i).irest(j)
      if(irest.gt.0) then
      u(j)=  a_vector(irest)
      else
      u(j)=0.0
      endif
      end do
```

```
c――――――――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル
      if(Member(i).element_type.eq.6) then
      ien= Member(i).n_model_type
      ie = Member(i).nm_element
      ii=Element(ie).nm_type
      call Cal_nonlin_maxwelldamp(E_model6_real(ien),
     *                         bk,ii)
      call Rotate_K(bk,rot_memb(1,1,1,i),
     *              rot_memb(1,1,2,i),ac_nonlinear)
c――――――――――――――――――――――――――――――――――――★剛床変換チェック
      n_gouyuka=0
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
       n_gouyuka=1
        aly(k)= Point(i1).coord_local(1)
        alx(k)= Point(i1).coord_local(2)
       else
        aly(k)=0.
        alx(k)=0.
       endif
      enddo
      if(n_gouyuka.ne.0) call Set_gtrans_k(ac_nonlinear(1,1,i),aly(1),alx(1), aly(2),alx(2) )

c――――――――――――――――――――――――――――――――――――★その他の減衰部材
      else
      do j=1,12
      do k=1,12
      ac_nonlinear(j,k)=ac_member(j,k,ij)
      enddo
      enddo
      endif
c――――――――――――――――――――――――――――――――
      do  j=1,12
      irest=Member(i).irest(j)
      if(irest.gt.0) then
      sum=0.
      do k=1,12
      sum=sum+ac_nonlinear(j,k)*u(k)
      enddo
      ld_point(irest)=ld_point(irest) - sum
      endif
      end do
      endif
      enddo
c――――――――――――――――――――――――――――――――――――――――――――――――――

      return
      end
```

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Add_fdd_ld
C       ――――――――――――――――――――――――――――――――――
```

```
C     ●   Maxwell 減衰項に関するベクトルを加える。(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Add_fdd_ld(ld_point_repeat,E_model6_real,Element,
     *       Member,n_member,est_vel_point,rot_memb,Point)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s   / Member
      record / point_s    / Point
      record / e_model6_real_s / E_model6_real
      record /element_s   / Element
      dimension Member(*),E_model6_real(*),Element(*)
      dimension Point(*)
      real*8 ld_point_repeat(*),est_vel_point(*)
      dimension rot_memb(3,3,2,*)
      dimension av(12),ud(12),vpp(12)
c―――――――――――――――――――――――――――――――――――――――
c     ld_point_repeat(*)    : real*8   線形右辺項ベクトル
c     Member                :structure
c     n_member              :integer   部材数
c     est_vel_point         :real*8    予測節点速度
c―――――――――――――――――――――――――――――――――――――――
      do i=1,n_member
      mem = i
      iet  = Member(i).element_type
      iett=(iet-1)/10
      ie = Member(i).nm_element
c――――――――――――――――――――――――――――――――★部材減衰を持っているか
      if( Element(ie).nm_damp .ne. 0) then
      ien= Member(i).n_model_type
c      write(76,'(a,3i4)') ' mem:',i,Member(i).nm_point(1),
c     *                    Member(i).nm_point(2)
      do  j=1,12
      irest = Member(i).irest(j)
      if(irest.ne.0) then
      ud(j)=est_vel_point(irest)
      else
      ud(j)=0.
      endif
      enddo
c――――――――――――――――――――★剛床座標変換
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
       k1=6*(k-1)+1
       call Set_gtrans_u(1,ud(k1),Point(i1).coord_local(1),
     *                                  Point(i1).coord_local(2))
       endif
      enddo
c――――――――――――――――――――★座標変換
c     write(76,'(6e12.4,3x,6e12.4)') ( ud(j),j=1,12)
      call RotateL_v(1,ud,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vpp)
c     write(76,'(6e12.4,3x,6e12.4)') ( vpp(j),j=1,12)
      if(Member(i).nm_dll_element .ne. 0) goto 9999   ! DLL 要素
      if(iett.eq.0)then
```

```
      goto(11,12,13,14,15,16,17,18,19,20),iet
   11 continue
c――――――――――――――――――――――――――――――――――★Model_No.1 通常の有限要素弾塑性モデル

      goto 100
   12 continue
c――――――――――――――――――――――――――――――――――★Model_No.2 3 次元せん断弾塑性モデル

      goto 100
   13 continue
c――――――――――――――――――――――――――――――――――★Model_No.3 3 次元軸力弾塑性モデル

      goto 100
   14 continue
c――――――――――――――――――――――――――――――――――★Model_No.4 3 次元ケーブル弾塑性モデル

      goto 100
   15 continue
c――――――――――――――――――――――――――――――――――★Model_No.5 3 次元免振モデル

      goto 100
   16 continue
c――――――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル
      ii=Element(ie).nm_type
      call Cal_force_maxwelldamp(vpp,E_model6_real(ien),av,ii,i)
      goto 100
   17 continue
      ・
      ・
  100 continue
c――――――――――――――――――――――――――――――――――★右変更への追加
      call RotateL_v(2,av,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vpp)
c――――――――――――――――――――――――★剛床座標変換
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
        k1=6*(k-1)+1
        call Set_gtrans_f(2,vpp(k1),
     *     Point(i1).coord_local(1),Point(i1).coord_local(2))
        endif
      enddo
      do  j=1,12
      irest = Member(i).irest(j)
      if(irest.ne.0) then
      ld_point_repeat(irest)=ld_point_repeat(irest) - vpp(j)
      end if
      end do
      endif
      end do
      return
      end
```

9.5.10.4 動的固有問題解析への組み込み

動的固有問題解析においても、座標変換を行なわなければならない。固有問題解析用の主サブルーチン submain_dynamic_b()で、まず、変更しなければならないサブルーチンを示す。

```
Rotate_stiffness()
Build_sky_mm_E
Rotate_mass()
Out_Eigen
```

ただし、Rotate_stiffness()、Rotate_mass()は既に変更を行っている。

まず、主サブルーチンで挿入するサブルーチンは、以下のようである。

```
c―――――――――――――――――――――――――――――――――――――――
c
c    ★        構造・荷重データを入力し、データの設定を行う
c
c―――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――★基本構造データを入力(ok)
c      open (damp_out,FILE='EOUTPUT')
      nfix=5
      nfi=1
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat =10
      return
      endif
c                          write(damp_out,*) ' Get_structure in Ok'
      call Get_structure(Point,Member,Element,Parameter_C,
     *                   Model_type,ierr,
     *                   S_comp_model,E_Fiber_work)
      close(nfix)
c                            write(damp_out,*) ' Get_structure Ok'
      if(ierr .ne. 0) return
c―――――――――――――――――――――――――――――――――――――――
c
c    ★        剛床の設定
c
c―――――――――――――――――――――――――――――――――――――――
      call Set_gouyuka(Parameter_C,Point)


      if(Eigen_d.load_mass .ne. 0.or. Parameter_C.n_gouyuka.ne.0) then
      load_mass_x=1
      else
      load_mass_x=0
      endif
      call Build_sky_mm_E(n_step,gskymm,
     *          Point,n_point,am_point,rot_local,
```

```
     *           n_local_coord,max_h_sky,load_mass_x)
c       write(damp_out,*) ' Build_sky_mm ok'
c     write(76,'(12e12.4)')(gskymm(j),j=1,n_unknown)

c―――――――――――――――――――――――――――――――――――――――
c
c    ★          サブスペース法の計算用データをセットする
c
c―――――――――――――――――――――――――――――――――――――――
      nroot = Eigen_d.n_modes
      nc = min(2*nroot,nroot+8)
      nnc = nc*(nc+1)/2
      nnm=Parameter_C.n_unknown+1
      IFSS=0  !Ｓｔｕｒｍ列による検定のためのフラグ 0 : 検定しない
      IFPR=0  !反復の間にプリントするためのフラグ 0 : プリントしない
      if(Eigen_d.load_mass .ne. 0 .or. Parameter_C.n_gouyuka .ne.0 ) then
      nm_skyline=n_skyline
      load_mass=1
      else
      nm_skyline=n_unknown
      load_mass=0
      endif
```

次に、刺激係数を求める式について検討する。第ｉ次モードの刺激係数は、次式で与えられる。

$$\beta_{xi}=\frac{\varphi_i^T MI_x}{\varphi_i^T M\varphi_i};\quad \beta_{yi}=\frac{\varphi_i^T MI_y}{\varphi_i^T M\varphi_i};\quad \beta_{zi}=\frac{\varphi_i^T MI_z}{\varphi_i^T M\varphi_i}$$

上式は全体座標系で表されたものであり、釣合座標系で評価された振動モードや質量行列ではない。ここで、式(9.3)を用いて、上式を釣合座標系に変換する。ただし、以下の式は、各節点について評価する。まず、刺激係数の分母であるが、

$$\varphi_i^T M\varphi_i=\phi_i^T R_u^T MR_u\phi_i$$

となり、釣合座標系で表した質量行列をｍとすると、

$$m=R_u^T MR_u$$

であり、したがって、

$$\varphi_i^T M\varphi_i=\phi_i^T m\phi_i$$

となり、刺激係数の分母は、全体座標系で評価しても、釣合座標系で評価しても同じ値となる。ここで、φ_i は全体座標系の第ｉ次振動モードであり、ϕ_i は釣合座標系におけるそれである。

次に、刺激係数における各分子を検討しよう。まず、x 方向の刺激係

数の分子を座標変換する。

$$\varphi_i^T M I_x = \phi_i^T R_u^T M R_u R_u^{-1} I_x$$

上式で、全体座標系から釣合座標系で評価した質量行列を用いると

$$\varphi_i^T M I_x = \phi_i^T m R_u^{-1} I_x$$

となる。ここで、式(9.6)を用いると、

$$\varphi_i^T M I_x = \phi_i^T m R_f^T I_x$$

であり、この式は、節点で評価した後、和を取って求めても良い。釣合座標系で計算した $\phi_i^T m$ を任意の節点で取り出したものを w_i とすると、

$$\varphi_i^T M I_x = \sum w_i R_f^T I_x$$

となり、これを書き下すと

$$w_i R_f^T I_x = \begin{pmatrix} w_{xi} & w_{yi} & w_{zi} & w_{\theta xi} & w_{\theta yi} & w_{\theta yi} \end{pmatrix} \begin{bmatrix} 1 & & & & & L_y \\ & 1 & & & & -L_x \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix} \begin{Bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

$$= w_{xi}$$

となる。他のｙ方向、ｚ方向の刺激係数分子も同様となる。ただし、節点毎に上記の分子を計算して和を取ると、他の節点の自由度と同一視する変位は、2 重に評価することになる。そこで、釣合座標系で使用された節点自由度のみ評価するするように **Copy_restraint_point()**サブルーチンを用いて同一視する自由度を取り除く。

次に、変更すべき 2 つのサブルーチンを示すことにする。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Build_sky_mm_E
C       ―――――――――――――――――――――――――――――――――
C     ●     集中質量行列のスカイライン行列への組み込み（固有問題用）
C     ―――――――――――――――――――――――――――――――――
      subroutine Build_sky_mm_E(n_istep,gskym,
     *           Point,n_point, am_point , rot_local,
     *           n_local_coord ,max_h_sky,load_mass)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      dimension am_point(2,*),gskym(*)
```

```
      dimension max_h_sky(0:*),rot_local(3,3,*)
      record / point_s     / Point
      dimension Point(*)
      dimension amm(6,6)
c-----------------------------------------------------------------
c     gskym(n_skyline)          :real*8    スカイライン行列
c     Point                     :structure
c     n_point                   :integer   節点数
c     am_point(2,n_point)       :real*8   節点集中質量
c     rot_local(3,3, *)         :real*8   全体座標系から局所座標への回転行列
c     n_local_coord             :integer    局所座標系を用いている節点数
c     Newmark_P                 :structure
c     max_h_sky(n_unknown+1)    :integer     スカイライン行列の各列の高さ
c     load_mass                 :integer    部材分布質量の使用
c
c     集中系では、座標変換は必要でない。
c
c-----------------------------------------------------------------
      if(n_istep.eq.1) then
      ik=1
      else
      ik=2
      endif

      if(load_mass.eq.0) then
c-------------------------★集中質量系の行列への組み込み
      do i=1,n_point
      am=am_point(ik,i)
      do j=1,3
      irest = Point(i).irest(j)
      if(irest.gt.0) then
      gskym(irest)=gskym(irest)+am
      endif
      enddo
      enddo
      else
c-------------------------★整合質量系の行列への組み込み
      do i=1,n_point
      am=am_point(ik,i)
c------------------------- 剛床座標変換
      if(Point(i).n_group_gouyuka.ne.0) then
      call Set_gtrans_m(am,amm, Point(i).coord_local(1),
     *                          Point(i).coord_local(2))
      do j=1,3
      irest = Point(i).irest(j)
      if(irest.gt.0) then
      i3=max_h_sky(irest)
      gskym(i3)=gskym(i3)+am
      endif
      enddo
      irest = Point(i).irest(6)
      if(irest.gt.0) then
      i3=max_h_sky(irest)
      gskym(i3)=gskym(i3)+amm(6,6)
```

```
      endif

      i1= Point(i).irest(1)
      if(i1.gt.0) then
      i2= Point(i).irest(6)
       if(i2.gt.0.and.i2.le.i1) then
       i3=max_h_sky(i1)-(i1-i2)
       gskym(i3)=gskym(i3)+amm(1,6)
       elseif(i2.gt.0.and.i1.le.i2) then
       i3=max_h_sky(i2)-(i2-i1)
       gskym(i3)=gskym(i3)+amm(1,6)
      endif
     endif
      i1= Point(i).irest(2)
      if(i1.gt.0) then
      i2= Point(i).irest(6)
      if(i2.gt.0.and.i2.le.i1) then
      i3=max_h_sky(i1)-(i1-i2)
      gskym(i3)=gskym(i3)+amm(2,6)
      elseif(i2.gt.0.and.i1.le.i2) then
      i3=max_h_sky(i2)-(i2-i1)
      gskym(i3)=gskym(i3)+amm(2,6)
       endif
       endif
     else
     do j=1,3
     irest = Point(i).irest(j)
     if(irest.gt.0) then
     i3=max_h_sky(irest)
     gskym(i3)=gskym(i3)+am
     endif
     enddo
     endif

     end do
     endif
     return
     end
```

```
C    ―――――――――――――――――――――――――――――――――
C    ●   SUBROUTINE /Copy_restraint_point
C    ―――――――――――――――――――――――――――――――――
C    ●    刺激係数を計算するための節点拘束表の設定
C    ―――――――――――――――――――――――――――――――――
     subroutine Copy_restraint_point(Parameter_C,Point,Control,
    *                                Point_rest)
     implicit real*8(A-H,O-Z)
     include "submain.h"
     record /control_s     / Control
     record / parameter_s  / Parameter_C
     record / point_s      / Point
     dimension Point(*)
     integer Point_rest(6,*)
```

```
C     ―――――――――――――――――――――――――――――――――――
      do i=1,Parameter_C.n_point
      do j=1,6
      Point_rest(j,i)=0
      enddo
      enddo
      do ii=1,Parameter_C.n_unknown
      do i=1,Parameter_C.n_point
      do j=1,6
      if(ii.eq.Point(i).irest(j)) then
      Point_rest(j,i)=1
      goto 100
      endif
      enddo
      enddo
  100 continue
      enddo
      write(76,'(//a,2i4)') ' 刺激係数用拘束表 ',Parameter_C.n_point,
     *Parameter_C.n_unknown
      do i=1,Parameter_C.n_point
      write(76,'(7i8)') i,(Point_rest(j,i),j=1,6)
      enddo
      return
      end
```

```
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,Point_rest)
      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(6),yy(6),Shigeki(3),tdisp(6)
      integer Point_rest(6,*)
C     ―――――――――――――――――――――――――――――――――――
      n_point=Parameter_C.n_point
      n_local_coord = Parameter_C.n_local_coord
      n_mode = Eigen_d.n_modes
      load_mass=Eigen_d.load_mass
      n_gouyuka = Parameter_C.n_gouyuka
      if(n_gouyuka .ne. 0) load_mass=1
```

```
      pi = 3.14159265D0
c―――――――――――――――――――――――――★刺激係数＊振動モード
      do i=1,n_point
      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
      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
       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))

   10 enddo
c―――――――――――――――――――――――――★レーリー減衰等に関する係数計算
      A0 = 0.0D0
      A1 = 0.0D0
      n_damp_1=Newmark_P.n_damp_1
      n_damp_2=Newmark_P.n_damp_2
      n_damp_type=Newmark_P.n_damp_type
      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
      if (n_damp_type.eq.1 ) then
      A0 = 2.0*h1*Omega(n_damp_1)
      elseif (n_damp_type.EQ.2 ) then
      A1 = 2.0*h1/Omega(n_damp_1)
      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
  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      write(76,535) n_mode
c――――――――――――――――――――――――――★刺激係数、固有周期等計算
      do 100 imode=1,n_mode
      do i=1,n_unknown
      disp(i) = Eigen_Vector(i,imode)
      enddo
c――――――――――――――――――――――――――★刺激係数のゼロセット
      do j=1,3
      Shigeki(j)=0.
      enddo
      call Multm(load_mass,gskymm,disp,w,max_h_sky,n_unknown)
c――――――――――――――――――――――――――★刺激係数の分母計算
      t_dmd = 0.0
      do i = 1, n_unknown
      t_dmd = t_dmd + disp(i)*w(i)
      enddo
c     write(76,'(a,e16.5)') ' 刺激係数分母',t_dmd
c――――――――――――――――――――――――――★刺激係数
      if(n_local_coord.eq.0.and.n_gouyuka.eq.0) then
         do i= 1, n_point
         do j=1,3
         irest =Point(i).irest(j)
         if(irest.gt.0.and.Point_rest(j,i).ne.0)
     *               Shigeki(j) = w(irest) + Shigeki(j)
         enddo
        enddo
      else
c――――――――――――――――――――――――――★座標変換
        do i= 1, n_point
         local_coord=Point(i).local_coord
         if(local_coord.eq.0.and.Point(i).n_group_gouyuka.eq.0) then
           do j=1,3
           irest =Point(i).irest(j)
           if(irest.gt.0.and.Point_rest(j,i).ne.0)
     *              Shigeki(j) = w(irest) + Shigeki(j)
           enddo
         elseif(Point(i).n_group_gouyuka.ne.0) then
c――――――――――――――――――――――――――★剛床変換
           do j=1,3
           yy(j)=0.
           irest =Point(i).irest(j)
           if(irest.gt.0.and.Point_rest(j,i).ne.0)  yy(j) = w(irest)
           enddo
c          j=6
c          irest =Point(i).irest(j)
```

```
c                                        yy(j)=0.
c          if(irest.gt.0)  yy(j) = w(irest)
c     write(76,'(a,i4,4f12.4)') ' yy2',i,(yy(j),j=1,3),yy(6)
c          call Set_gtrans_f(1,yy, Point(i).coord_local(1),
c     *                      Point(i).coord_local(2) )

           do j= 1, 3
           Shigeki(j) = yy(j) + Shigeki(j)
           enddo
         else
c----------------------------------★局所座標変換
           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
           if(Point_rest(j,i).ne.0) Shigeki(j) = yy(j) + Shigeki(j)
           enddo
         endif
        enddo

      endif
c     write(76,'(a,3e16.5)') ' 刺激係数分子',(Shigeki(j),j=1,3)
c----------------------------------★刺激係数の計算
      if(t_dmd.ne.0.)then
      do j=1,3
      Shigeki(j)=Shigeki(j)/t_dmd
      enddo
      endif
c----------------------------------★刺激係数のチェック
      if(n_local_coord.eq.0 .and. n_gouyuka.eq.0) then
         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
c----------------------------------★座標変換
      else
        do i= 1, n_point
         local_coord=Point(i).local_coord
         if(local_coord.eq.0.and.Point(i).n_group_gouyuka.eq.0) then
           do j=1,3
            yy(j)=0.
           irest =Point(i).irest(j)
           if(irest.gt.0) yy(j) = disp(irest)
```

```
          enddo
        elseif(Point(i).n_group_gouyuka .ne.0) then
c―――――――――――――――――――――――――★剛床変換
          do j=1,3
          yy(j)=0.
          irest =Point(i).irest(j)
          if(irest.gt.0)  yy(j) = disp(irest)
          enddo
          j=6
          irest =Point(i).irest(j)
           yy(j)=0.
          if(irest.gt.0)  yy(j) =  disp(irest)
           call Set_gtrans_u(1,yy, Point(i).coord_local(1),
     *                       Point(i).coord_local(2) )
        else
c―――――――――――――――――――――――――★局所座標変換
          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
      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)
      endif

      if ( ifl1.EQ.1 ) then
c―――――――――――――――――――――――――――――★振動モードの出力
```

```
      write(iflz1,1000) imode
 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
c      write(76,1000) imode
       do i=1,n_unknown
       disp(i) = Eigen_Vector(i,imode)
       enddo

       if(n_local_coord.eq.0.and. n_gouyuka.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
c――――――――――――――――――――――――★座標変換
       else
         do i= 1, n_point
          local_coord=Point(i).local_coord
          if(local_coord.eq.0.and.Point(i).n_group_gouyuka .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)
          elseif(Point(i).n_group_gouyuka.ne.0) then
c――――――――――――――――――――――――★剛床変換
            do j=1,6
            tdisp(j)=0.
            irest =Point(i).irest(j)
            if(irest.gt.0)  tdisp(j) = disp(irest)
            enddo
             call Set_gtrans_u(1,tdisp, Point(i).coord_local(1),
     *                        Point(i).coord_local(2) )
                              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'
      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
      return
      end
```

9.6 塑性変形倍率

9.6.1 はじめに

SPACE（Ver.3.00）では、立体骨組みの静的解析や動的解析を行っている。その際、妥当な耐震性能評価が重要となる。そこで、3 次元構造物の地震応答解析を行い、各部材の塑性変形倍率及び累積塑性変形倍率、すなわち部材に吸収された塑性ひずみエネルギーを評価する。本節で、SPACE で実行している部材レベルの塑性変形倍率及び累積変形倍率の計算手法を定義し、処理方法を解説する。

9.6.2 塑性変形率

断面内の微小部分における塑性変形率は、１軸応力状態で評価すると、

$$\mu = \frac{\varepsilon_x}{\varepsilon_y} \qquad \cdots\cdots\cdots\cdots(9.16)$$

で表される。ここで、ε_x は軸方向ひずみであり、ε_y は降伏点ひずみを表す。しかし、断面全体で塑性変形率を評価することは、かなり困難を伴う。

まず、単純曲げの状態について考えよう。この場合、次式のように塑性ヒンジが生じたときの曲率 ϕ_P を用いて評価する。

$$\mu = \frac{\phi}{\phi_P} \qquad \cdots\cdots\cdots\cdots(9.17)$$

線形部分の曲げモーメントと曲率の関係は、

$$M = EI\phi$$

で表されることから、曲率 ϕ_P は次式で与えられる。

$$\phi_P = \frac{M_P}{EI} \qquad \cdots\cdots\cdots\cdots(9.18)$$

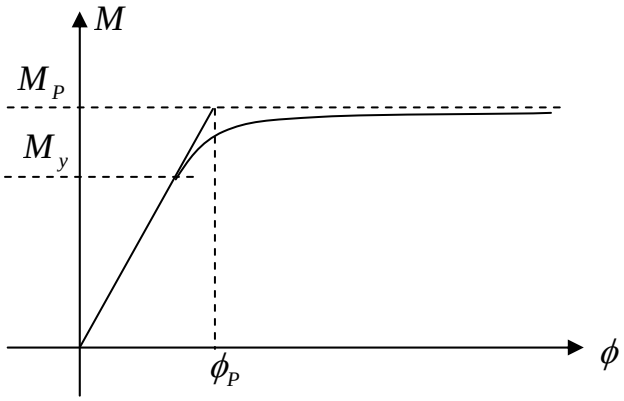


図 9-3　曲げモーメントと曲率の関係

次に、軸力が存在する場合で、しかも 2 軸の曲げモーメントが発生している場合の塑性変形率を検討しよう。SPACE では、塑性論アナロジーモデルの塑性関数として、次の 3 つの関数が用いられている。

$$\left.\begin{aligned} f &= \left(\frac{N}{N_p}\right)^2 + \sqrt{\left(\frac{M_y}{M_{yp}}\right)^2 + \left(\frac{M_z}{M_{zp}}\right)^2} - 1 \\ f &= \left(\frac{N}{N_p}\right)^2 + \left(\frac{M_y}{M_{yp}}\right)^2 + \left(\frac{M_z}{M_{zp}}\right)^2 - 1 \\ f &= \left|\frac{N}{N_p}\right| + \sqrt{\left(\frac{M_y}{M_{yp}}\right)^2 + \left(\frac{M_z}{M_{zp}}\right)^2} - 1 \end{aligned}\right\} \cdots\cdots\cdots\cdots(9.19)$$

上記の３つの関数を応用して、塑性変形倍率 μ を評価する関数を次のように定義する。

$$\left.\begin{aligned} \mu &= \left(\frac{\varepsilon_x}{\varepsilon_y}\right)^2 + \sqrt{\left(\frac{\phi_y}{\phi_{yp}}\right)^2 + \left(\frac{\phi_z}{\phi_{zp}}\right)^2} \\ \mu &= \sqrt{\left(\frac{\varepsilon_x}{\varepsilon_y}\right)^2 + \left(\frac{\phi_y}{\phi_{yp}}\right)^2 + \left(\frac{\phi_z}{\phi_{zp}}\right)^2} \\ \mu &= \left|\frac{\varepsilon_x}{\varepsilon_y}\right| + \sqrt{\left(\frac{\phi_y}{\phi_{yp}}\right)^2 + \left(\frac{\phi_z}{\phi_{zp}}\right)^2} \end{aligned}\right\} \cdots\cdots\cdots\cdots(9.20)$$

ここで、ϕ_{yp} と ϕ_{zp} は、塑性ヒンジが生じたときの各々ｙ軸とｚ軸に関する曲率を表す。

塑性関数が任意断面について、精度良く塑性状態を表すことが可能で

あると、式(9.20)で塑性変形率を精度良く表すことができる。なお、ϕ_{yp} と ϕ_{zp} は、次式で求める。

$$\varepsilon_P = \frac{N_P}{EA};\quad \phi_{yP} = \frac{M_{yP}}{EI_y};\quad \phi_{zP} = \frac{M_{zP}}{EI_z} \qquad \cdots\cdots\cdots\cdots(9.21)$$

9.6.3 累積塑性変形率

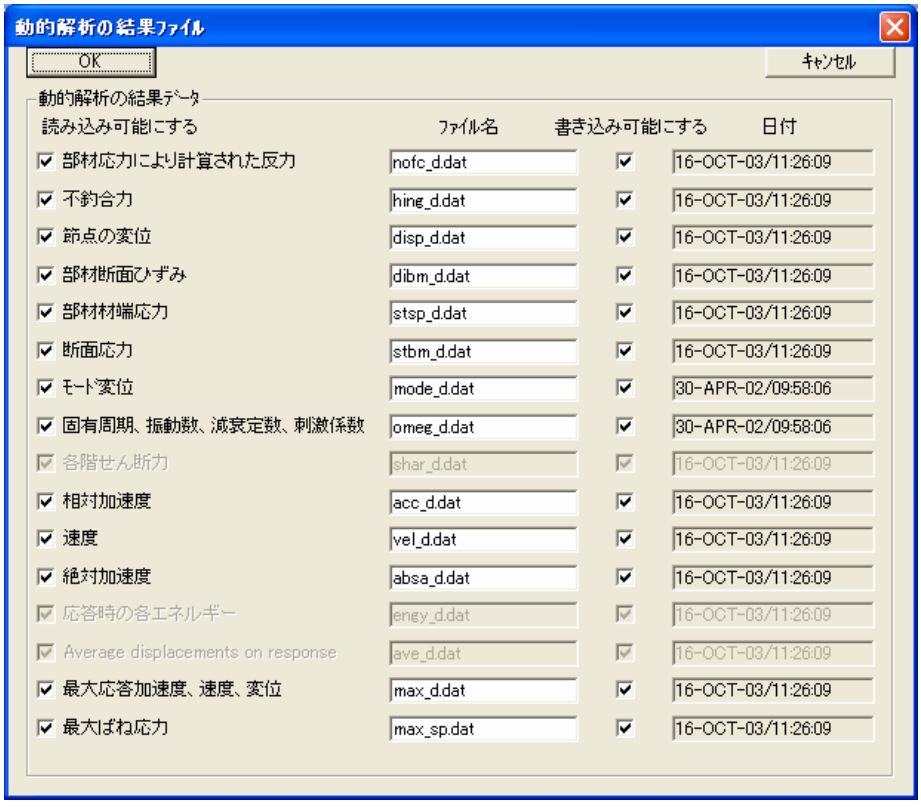
累積塑性変形率は、塑性変形率を利用して、以下のように定義する。

$$\eta = \sum(\mu_{\max} - 1) \qquad \cdots\cdots\cdots\cdots(9.22)$$

ここで、η は累積塑性変形率であり、$\mu_{\max}$ は各振動振幅時における最大塑性変形率を表す。ただし、値が 1 以下の場合、つまり、弾性状態の場合は参入しないものとする。また、累積塑性回数とは、式(9.22)で、η を数えた回数とする。

9.6.4 動的解析におけるひずみの出力

動的解析で、次のサブルーチン Out_Strain()を用いて、部材断面のひずみを出力する。出力するファイルは、以下に示す SPACE のダイアログ



断面内のひずみを出力するファイル

を用いて行う。上記ダイアログの中で、「部材断面ひずみ」が新たに設けられたファイルであり、ここで入出力許可並びにファイル名を設定する。このファイルに与えられたキィワードは、「dibm_d」である。

動的解析における主サブルーチンでは、以下のようにこのサブルーチンをコールする。

```
c------------------------------------------------------------
c
c    ★        解析結果を出力するファイル群をオープンする(ok)
c
c------------------------------------------------------------
      call flcheck(ifl,ifly,iflz,ierr,Control.type_analysis)
       if(ierr.ne.0) then
      ierr_dat =14
       endif
       write(damp_out,*) ' Out_Strain ok' ,n_member
c-----------------------------------★出力指定した断面がファイバー要素
c-----------------------------------★かどうかチェック(ok)
      call out_section_check(Member,Element,
     *               Parameter_C.n_member,No_section,Out_section,
     *               ifl,iflz,i_print,S_comp_model,E_modelx,
     *               M_modelx,E_fiber_work,M_fiber_work)
c       write(damp_out,*) ' out_section_check ok' ,n_member
c-----------------------------------★断面ひずみの個数出力
      call Out_Strain(0,Member,Element,E_model11,M_model11,
     *               E_model12,M_model12,E_model13,M_model13,
     *               E_model15,M_model15,
     *               E_model21,M_model21,E_model22,M_model22,
     *               E_model31,M_model31,E_model32,M_model32,
     *               E_model33,M_model33,
     *               E_model_fiber,M_model_fiber,
     *               Parameter_C.n_member,ifl,iflz,i_print,
     *               S_comp_model, E_modelx, M_modelx,
     *               E_fiber_work, M_fiber_work)
c       write(damp_out,*) ' Out_Strain ok' ,n_member
       .
       .
c-----------------------------------★断面ファイバー応力を出力
      call Out_Fiber(Member,Element,E_model11,M_model11,
     *               E_model12,M_model12,E_model13,M_model13,
     *               E_model15,M_model15,
     *               E_model21,M_model21,E_model22,M_model22,
     *               E_model31,M_model31,E_model32,M_model32,
     *               E_model33,M_model33,
     *               E_model_fiber,M_model_fiber,
     *               n_member,ifl,iflz,i_print,Out_section,
     *               S_comp_model, E_modelx, M_modelx,
     *               E_fiber_work, M_fiber_work)
c      write(damp_out,*) ' Out_stress ok'
c-----------------------------------★断面ひずみを出力
      call Out_Strain(1,Member,Element,E_model11,M_model11,
     *               E_model12,M_model12,E_model13,M_model13,
```

```
     *              E_model15,M_model15,
     *              E_model21,M_model21,E_model22,M_model22,
     *              E_model31,M_model31,E_model32,M_model32,
     *              E_model33,M_model33,
     *              E_model_fiber,M_model_fiber,
     *              n_member,ifl,iflz,i_print,
     *              S_comp_model, E_modelx, M_modelx,
     *              E_fiber_work, M_fiber_work)
      c      write(damp_out,*) ' Out_stress ok'
      c―――――――――――――――――――――――――――――――――★応力等の最大値セット
            call Get_max_stress(Member,n_member,Max_stress)
```

最初のサブルーチンコールでは、初期設定を行い、次のサブルーチンコールで部材のひずみがファイルに出力される。次に、このサブルーチンの内容を示そう。

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Out_Strain
C       ――――――――――――――――――――――――――――――――
C     ●   部材の断面ひずみを出力(ok)
C     ――――――――――――――――――――――――――――――――――
      subroutine Out_Strain(nx_han,Member,Element,E_model11,M_model11,
     *            E_model12,M_model12,E_model13,M_model13,
     *            E_model15,M_model15,
     *            E_model21,M_model21,E_model22,M_model22,
     *            E_model31,M_model31,E_model32,M_model32,
     *            E_model33,M_model33,
     *            E_model_fiber,M_model_fiber,
     *            n_member,ifl,iflz,i_print,
     *            S_comp_model, E_modelx, M_modelx,
     *            E_fiber_work, M_fiber_work)

      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
c―――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
     record / Member_s       / Member
     record / Element_s      / Element
      record / M_model11_s    / M_model11
      record / E_model11_s    / E_model11
      record / M_model12_s    / M_model12
      record / E_model12_s    / E_model12
      record / M_model13_s    / M_model13
      record / E_model13_s    / E_model13
      record / M_model15_s    / M_model15
      record / E_model15_s    / E_model15
      record / M_model21_s    / M_model21
```

```
      record / E_model21_s    / E_model21
      record / M_model22_s    / M_model22
      record / E_model22_s    / E_model22
      record / M_model31_s    / M_model31
      record / E_model31_s    / E_model31
      record / M_model32_s    / M_model32
      record / E_model32_s    / E_model32
      record / M_model33_s    / M_model33
      record / E_model33_s    / E_model33
      record / M_model_fiber_s   / M_model_fiber
      record / E_model_fiber_s   / E_model_fiber
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension ifl(16),iflz(16)
      dimension Member(*),Element(*),M_model11(*),E_model11(*),
     *          M_model12(*),E_model12(*),M_model13(*),E_model13(*),
     *          M_model15(*),E_model15(*)
      dimension M_model21(*),E_model21(*),M_model22(*),E_model22(*)
      dimension M_model31(*),E_model31(*),M_model32(*),E_model32(*)
      dimension M_model33(*),E_model33(*)
      dimension mxtype(100),myytype(4)
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)
      dimension E_fiber_work(*),M_fiber_work(*)
      data mxtype/1,1,1,1,1,1,1,1,1,1, 1,3,1,3,1,1,3,3,3,1,
     3           1,3,1,1,1,1,1,1,1,1, 1,3,1,1,1,1,1,1,1,1,
     5           1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,
     7           1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,
     9           1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1/
      data myytype/2,4,3,5/
      real*4    v(100),vf(3),vfx
      dimension vff(3)
      write(76,'(a,3i4)') ' out_strain',nx_han,ifl(4),n_member
c     i_print         :integer 出力制御変数 0:ファイル出力あり
c-----------------------------★全部材の断面個数検索、出力
      if(nx_han.eq.0.and.ifl(4).ne.0) then
      do i=1,n_member
      vfx=0.
c-------------------------------★断面ひずみの個数
      ie = Member(i).nm_element
      imm= Element(ie).n_element              ! 要素番号
      immm= Member(i).n_model_type              ! モデルタイプ別番号
      iet  = Member(i).element_type
      ax=(Element(ie).A*Element(ie).E)
      if(ax.ne.0.)then
      vff(1)=Element(ie).ANP/ax
      else
      vff(1)=0.
      endif
      ax=(Element(ie).RIy*Element(ie).E)
      if(ax.ne.0.)then
      vff(2)=Element(ie).AMPY/ax
      else
      vff(2)=0.
      endif
      ax=(Element(ie).RIz*Element(ie).E)
```

```
      if(ax.ne.0.)then
      vff(3)=Element(ie).AMPZ/ax
      else
      vff(3)=0.
      endif
c      write(76,'(i4,5f12.4)') i,vfx,(vff(j),j=1,3)
      if(iet.eq.11) then
      vfx=2.1
      elseif(iet.eq.12) then
      vfx=3.1
      elseif(iet.eq.15) then
      vfx=2.1
      elseif(iet.eq.13) then
      vfx=2.1
      elseif(iet.eq.14) then
      vfx=3.1
      elseif(iet.eq.16) then
      vfx=2.1
      elseif(iet.eq.17) then
      vfx=1.1
      elseif(iet.eq.18.or.iet.eq.19) then
      vfx=2.1
      elseif((iet-1)/10.eq.5.or. (iet-1)/10.eq.6) then
      i_comp= iet - 50                                                    ! 1
      n_out_stress  = S_comp_model(i_comp).n_out_stress
      do m=1, n_out_stress
      kk = S_comp_model(i_comp).n_out_stress_x(m)       ! 出力エレメント番号      3
      if(kk.ne.0) then
      vfx=vfx+1.
      endif
      enddo
      vfx=vfx+0.2
      endif
      n_sec=vfx
      if(n_sec.ne.0) then
      do j=1,3
      if(vff(j).ne.0.) vff(j)=1./vff(j)
      vf(j)=vff(j)
      enddo
      else
      do j=1,3
      vf(j)=0.
      enddo
      endif
      write(iflz(4))vfx,(vf(j),j=1,3)
      enddo
c―――――――――――――――――――――――――――――★断面ひずみ
      else
      if(i_print.ne.0) return
      if(ifl(4).eq.0) return
      do i=1,n_member
c―――――――――――――――――――――――――――――★断面ひずみ
      ie = Member(i).nm_element
      imm= Element(ie).n_element              ! 要素番号
```

```
      immm= Member(i).n_model_type              ! モデルタイプ別番号
      iet  = Member(i).element_type
      if(iet.eq.11) then
c―――――――――――――――――――――――★モデル１１
      vf(1) = M_model11(immm).d_epsi_x_1          !  軸方向歪
      vf(2) = M_model11(immm).d_epsi_y_1          !  ｙ軸に関する曲げ歪
      vf(3) = M_model11(immm).d_epsi_z_1          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model11(immm).d_epsi_x_2          !  軸方向歪
      vf(2) = M_model11(immm).d_epsi_y_2          !  ｙ軸に関する曲げ歪
      vf(3) = M_model11(immm).d_epsi_z_2          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)

      elseif(iet.eq.12) then
c―――――――――――――――――――――――★モデル１２
      vf(1) = M_model12(immm).d_epsi_x_1          !  軸方向歪
      vf(2) = M_model12(immm).d_epsi_y_1          !  ｙ軸に関する曲げ歪
      vf(3) = M_model12(immm).d_epsi_z_1          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model12(immm).d_epsi_x_2          !  軸方向歪
      vf(2) = M_model12(immm).d_epsi_y_2          !  ｙ軸に関する曲げ歪
      vf(3) = M_model12(immm).d_epsi_z_2          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model12(immm).d_epsi_x_c          !  軸方向歪
      vf(2) = M_model12(immm).d_epsi_y_c          !  ｙ軸に関する曲げ歪
      vf(3) = M_model12(immm).d_epsi_z_c          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)

      elseif(iet.eq.15) then
c―――――――――――――――――――――――★モデル１５

      vf(1) = M_model15(immm).d_epsi_x_1          !  軸方向歪
      vf(2) = M_model15(immm).d_epsi_y_1          !  ｙ軸に関する曲げ歪
      vf(3) = M_model15(immm).d_epsi_z_1          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model15(immm).d_epsi_x_2          !  軸方向歪
      vf(2) = M_model15(immm).d_epsi_y_2          !  ｙ軸に関する曲げ歪
      vf(3) = M_model15(immm).d_epsi_z_2          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)

      elseif(iet.eq.13) then
c―――――――――――――――――――――――★モデル２１
      vf(1) = M_model21(immm).d_epsi_x_1          !  軸方向歪
      vf(2) = M_model21(immm).d_epsi_y_1          !  ｙ軸に関する曲げ歪
      vf(3) = M_model21(immm).d_epsi_z_1          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model21(immm).d_epsi_x_2          !  軸方向歪
      vf(2) = M_model21(immm).d_epsi_y_2          !  ｙ軸に関する曲げ歪
      vf(3) = M_model21(immm).d_epsi_z_2          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)

      elseif(iet.eq.14) then
c―――――――――――――――――――――――★モデル２２
      vf(1) = M_model22(immm).d_epsi_x_1          !  軸方向歪
```

```
      vf(2) = M_model22(immm).d_epsi_y_1          !  ｙ軸に関する曲げ歪
      vf(3) = M_model22(immm).d_epsi_z_1          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model22(immm).d_epsi_x_2          !  軸方向歪
      vf(2) = M_model22(immm).d_epsi_y_2          !  ｙ軸に関する曲げ歪
      vf(3) = M_model22(immm).d_epsi_z_2          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model22(immm).d_epsi_x_c          !  軸方向歪
      vf(2) = M_model22(immm).d_epsi_y_c          !  ｙ軸に関する曲げ歪
      vf(3) = M_model22(immm).d_epsi_z_c          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)

      elseif(iet.eq.16) then
c-----------------------★モデル３１
      vf(1) = M_model31(immm).d_epsi_x_1          !  軸方向歪
      vf(2) = M_model31(immm).d_epsi_y_1          !  ｙ軸に関する曲げ歪
      vf(3) = M_model31(immm).d_epsi_z_1          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model31(immm).d_epsi_x_2          !  軸方向歪
      vf(2) = M_model31(immm).d_epsi_y_2          !  ｙ軸に関する曲げ歪
      vf(3) = M_model31(immm).d_epsi_z_2          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)

      elseif(iet.eq.17) then
c-----------------------★モデル３２
      vf(1) = M_model32(immm).d_epsi_x_1          !  軸方向歪
      vf(2) = M_model32(immm).d_epsi_y_1          !  ｙ軸に関する曲げ歪
      vf(3) = M_model32(immm).d_epsi_z_1          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model32(immm).d_epsi_x_2          !  軸方向歪
      vf(2) = M_model32(immm).d_epsi_y_2          !  ｙ軸に関する曲げ歪
      vf(3) = M_model32(immm).d_epsi_z_2          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)
      vf(1) = M_model32(immm).d_epsi_x_c          !  軸方向歪
      vf(2) = M_model32(immm).d_epsi_y_c          !  ｙ軸に関する曲げ歪
      vf(3) = M_model32(immm).d_epsi_z_c          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)

      elseif(iet.eq.18.or.iet.eq.19) then
c-----------------------★モデル１３
      vf(1) = M_model13(immm).d_epsi_x_1          !  軸方向歪
      vf(2) = M_model13(immm).d_epsi_y_1          !  ｙ軸に関する曲げ歪
      vf(3) = M_model13(immm).d_epsi_z_1          !  ｚ軸に関する曲げ歪
      write(iflz(4))(vf(k),k=1,3)

      elseif((iet-1)/10.eq.5.or. (iet-1)/10.eq.6) then
c-----------------------★新規モデル 51-70
      i_comp= iet - 50                                                  ! 1
      n_out_stress  = S_comp_model(i_comp).n_out_stress
      nmmx= Member(i).n_model_type -1                ! M_Fiber_work の開始番号   2
      nmx = Element(ie).n_section(1) -1              ! E_Fiber_work の開始番号
      do m=1, n_out_stress
      kk = S_comp_model(i_comp).n_out_stress_x(m)      ! 出力エレメント番号       3
      if(kk.ne.0) then
```

```
        immmx= M_Fiber_work(nmmx+kk).nm_section          ! 内部エレメント番号
        vf(1) = M_modelx(immmx).d_epsi_x                 !  軸方向歪
        vf(2) = M_modelx(immmx).d_epsi_y                 !  y 軸に関する曲げ歪
        vf(3) = M_modelx(immmx).d_epsi_z                 !  z 軸に関する曲げ歪
        write(iflz(4))(vf(k),k=1,3)                      ! ひずみ              ! 9
        endif
        enddo
        endif
c――――――――――――――――――――――――――――――★断面ひずみ出力終わり
        enddo
       endif
       return
       end
```

断面のひずみを出力するファイルの仕様について説明する。断面内のひずみは、全ての部材について出力しているわけではない。部材モデルの弾塑性状態を評価する特殊な位置で、軸方向ひずみ、ｙ軸の曲げひずみ、z 軸の曲げひずみを単精度で出力する。さらに、他のデータと同様に、ユーザーが設定した時間ごとに、結果が出力される。したがって、解析増分ごとにひずみが出力されないので注意されたい。

このファイルの最初は、全部材について次の情報が出力される。

1. 当該部材のひずみ出力位置数（ただし、実数で出力）
2. 式(9.21)で計算される塑性ひずみと塑性ヒンジが生じたときの各々ｙ軸とｚ軸に関する曲率の逆数

上記の２．の値は、塑性率を計算するために利用する。1. の値は、次に示すひずみを入力する際用いることになる。ひずみは、次のように部材モデルの弾塑性状態を評価する特殊な位置で、軸方向ひずみ、ｙ軸の曲げひずみ、z 軸の曲げひずみを単精度で出力する。

1. 軸方向ひずみ
2. ｙ軸の曲げひずみ
3. z 軸の曲げひずみ

9.6.6 ひずみの入力と塑性率計算

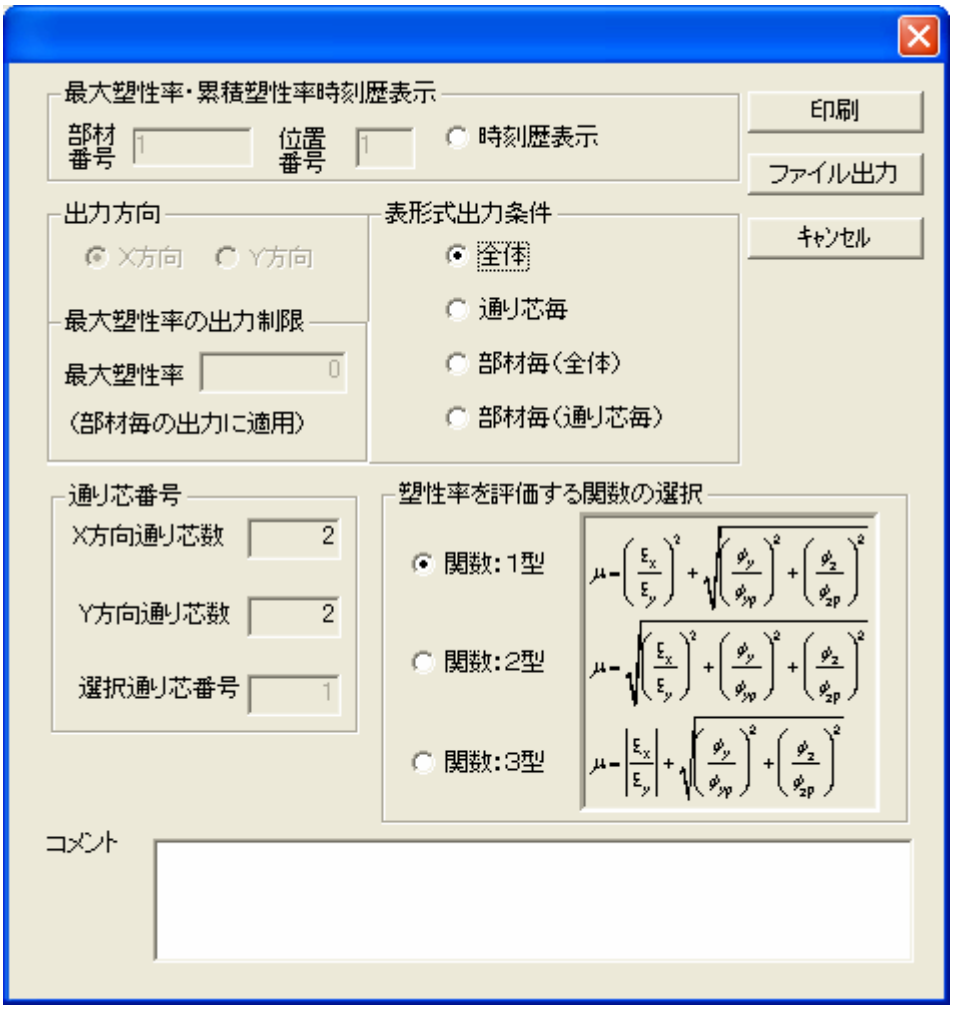
部材ひずみの最大塑性率などの処理は、プレゼンターで行う。最初に、プレゼンターで、各種の処理を行うためのデータ設定を行うダイアログを示す。

ここでは、次に示す 5 種の最大塑性率などの処理を行う。

1. 全フレームについて、最大塑性率、最大累積塑性率など求める。
2. 指定したフレーム毎に、最大塑性率、最大累積塑性率など求める。

3．全フレームについて、部材毎に最大塑性率や最大累積塑性率を求める。
4．指定したフレームの全部材について、最大塑性率や最大累積塑性率を求める。
5．指定した部材の塑性率、累積塑性率などを時刻歴で、グラフ表示する。

プレゼンターで使用するダイアログを次に示す。



上記のダイアログでデータを設定した後、動的解析で求めた各時刻におけるひずみを読むことになる。処理プログラムは C++で記述されており、関数 OnfilePrprint()において、書類として出力される 5 つの処理が行われる。この処理の選択は、ユーザーがダイアログで設定するが、プログラムでは、変数 m_radio_jyouken の値を用いて行われる。

```
void CDlg_sosei::OnFilePrPrint()
{
     UpdateData(TRUE);
//-------------------------------------------------------------
//     kai     : 階数
//     ifrm    : フレーム最大数
//     ix      : Ｘ方向フレーム数
//     iy      : Ｙ方向フレーム数
//     kn      : 階名
//     sn      : 層名
//     xn      : Ｘ方向フレーム名
//     yn      : Ｙ方向フレーム名
//     zh      : 階高
//     IERR    : エラー番号  0 : 無   1 : リードプロテクト   2 : ファイルなし
//-------------------------------------------------------------
// info.dat 基本データ読み込み
//        ikn = new char[5*ikai]; //階名
//        isn = new char[5*I_so]; //層名
//        ixn = new char[5*ifrm]; //Ｘフレーム名
//        iyn = new char[5*ifrm]; //Ｙフレーム名
//        izh = new float[ikai];  //階高
switch(m_radio_jyouken){
     case 0://全体（表）

     break;
     case 1://通り芯毎（表）
        if(m_radio_tori == 0){
                if(m_edit_tori_set > iix || m_edit_tori_set < 1){
     MessageBox("x 方向通り芯番号入力エラー");
     return;
                }
        }else{
        if(m_edit_tori_set > iiy || m_edit_tori_set < 1){
     MessageBox("y 方向通り芯番号入力エラー");
     return;
                }
        }
     break;
     case 2://部材毎：全体（表）
     break;
     case 3://部材毎：通り芯毎（表）
        if(m_radio_tori == 0){
                if(m_edit_tori_set > iix || m_edit_tori_set < 1){
     MessageBox("x 方向通り芯番号入力エラー");
     return;
                }
        }else{
        if(m_edit_tori_set > iiy || m_edit_tori_set < 1){
     MessageBox("y 方向通り芯番号入力エラー");
     return;
                }
        }
     break;
     case 4://部材毎：時刻歴（グラフ）
```

```
        break;
        default:
         MessageBox("デフォルト");
}
        int ihan=0;
        int nfl=39;// 部材ひずみファイル番号
         int n_type = m_radio_type;//関数型（in）
         int n_member = mnbsb;//部材数
         float dt =    F_nstep_k;//増分時間
         int n_time =   F_all_step;//増分分割数
         int n_radio_tori = m_radio_tori + 1;//通り芯方向 0:x 方向  1:y 方向
         int n_edit_tori_set = m_edit_tori_set;//通り芯番号
         int n_sosu =I_so;
         int m_radio_J = m_radio_jyouken;
         float m_edit_s_soseix = m_edit_s_sosei;
         int n_mem =  m_edit_n_member;
         int n_ichi=  m_edit_nx_member;

         IP_strain = new int[10*n_member];
       nx_member = new int[n_member];
       P_strain = new double[35*n_member];
       Sec_epsy = new double[3*n_member];
        if(m_radio_jyouken < 4){
      CAL_P_STRAIN(&ihan,&nfl,&n_type,&n_member,P_strain,
            IP_strain,nx_member,Sec_epsy,&dt,&n_time);
            if(ihan == 0 ){
            }else{
        MessageBox("最大塑性率は入力失敗");
        return;
        }}
        if(m_radio_jyouken < 2){
       S_hyou = new float[3*6*n_sosu];
       IS_hyou = new int[3*13*n_sosu];
        }
        if(m_radio_jyouken > 1 && m_radio_jyouken < 4){
       S_hyou = new float[3*n_member];
       IS_hyou = new int[6*n_member];
        }
        if(m_radio_jyouken == 4){
       S_hyou = new float[5*n_time];
       IS_hyou = new int[n_time];
        }
        int i_out=0;
        char path_name;
        int istrlen;
switch(m_radio_jyouken){
        case 0://全体（表）
      SET_HYO_STRAIN(&i_out,&path_name,&istrlen,&m_radio_J,&n_member,
            F_iso,P_strain,IP_strain,
            nx_member,S_hyou,IS_hyou,&n_radio_tori,&n_edit_tori_set,&n_sosu);
        I_print = 13;//最大塑性率
        break;
        case 1://通り芯毎（表）
```

```
        I_print = 13;//最大塑性率
    SET_HYO_STRAIN(&i_out,&path_name,&istrlen,&m_radio_J,&n_member,
          F_iso,P_strain,IP_strain,
           nx_member,S_hyou,IS_hyou,&n_radio_tori,&n_edit_tori_set,&n_sosu);
        break;
        case 2://部材毎：全体（表）
        I_print = 14;//最大塑性率
    SET_MEMBER_STRAIN(&i_out,&path_name,&istrlen,&m_radio_J,&n_member,
          F_iso,P_strain,IP_strain,
           nx_member,S_hyou,IS_hyou,&n_radio_tori,&n_edit_tori_set,&m_edit_s_soseix);
        break;
        case 3://部材毎：通り芯毎（表）
        I_print = 14;//最大塑性率
    SET_MEMBER_STRAIN(&i_out,&path_name,&istrlen,&m_radio_J,&n_member,
          F_iso,P_strain,IP_strain,
           nx_member,S_hyou,IS_hyou,&n_radio_tori,&n_edit_tori_set,&m_edit_s_soseix);
        break;
        case 4://部材時刻歴（グラフ）
        I_print = 15;//最大塑性率
    CAL_MEMBER_P_STRAIN(&i_out,&path_name,&istrlen,&n_mem,&n_ichi,&ihan,
         &nfl,&n_type,&n_member,S_hyou,nx_member,Sec_epsy,&n_time,&dt);
        break;
}
        CDialog::OnOK();
}
```

上記プログラムでは、最初に、ダイアログで入力したデータに矛盾がないかチェックした後、部材断面のひずみを入力し、各部材の最大塑性率を求めるサブルーチン Cal_P_Strain()をコールする。このサブルーチンは、処理１から４の場合使用され、塑性率などの時刻歴表示の処理５ではコールしない。

次に、これらの処理に必要となるワーク領域を動的に確保する。その後、５つの処理に分類して、書類として出力する表や図形用データを求める。これらの処理は次の３つのサブルーチンによって行われる。

1. SET_HYO_STRAIN（処理１、処理２）
2. SET_MEMBER_STRAIN（処理３、処理４）
3. CAL_MEMBER_P_STRAIN（処理５）

最初に、サブルーチン Cal_P_Strain()を検討しよう。このサブルーチンでは、まず、全部材について、ひずみの出力個数と塑性ひずみと塑性ヒンジが生じたときの各々ｙ軸と ｚ 軸に関する曲率の逆数を入力し、配列 nx_member と Sec_epsy にセットする。次に、処理結果を設定する配列 P_strain と IP_strain をゼロセットする。これ以降、出力時刻毎

にひずみを入力した後、サブルーチン check_strain()を用いて、最大塑性率などをチェックする。

サブルーチン check_strain()では、ひずみを各塑性ひずみで割り、軸方向、２つの曲げに関する塑性率を計算する。この値を元にユーザーが選択した計算式を用いて、断面塑性率を求める。次からは、最大塑性率、累積塑性率、累積損傷回数を計算する。

それでは、この２つのサブルーチンを示そう。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE Cal_P_Strain
C       ―――――――――――――――――――――――――――――――――
C     ●   各部材の塑性率と最大累積塑性率の計算
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_P_Strain(ihan,nfl,n_type,n_member,P_strain,
     *                        IP_strain,nx_member,Sec_epsy,dt,n_time)
      implicit real*8(A-H,O-Z)
      common /sf01/fnfile,jdfile,kdfile,iidat,title,lengf,ltitle,timex
      CHARACTER fnfile(60)*50,title*50,timex*20(60)
      integer*4 jdfile(60),kdfile(60),lengf(60)
      dimension P_strain(7,5,*), IP_strain(2,5,*), Sec_epsy(3,*),
     *          nx_member(*)
      real*4  eps(3),vf,dt
c―――――――――――――――――――――――――――――★
c     nfx:              ひずみ入力ファイル番号
c     n_time:           時間個数
c     dt:               増分時間
c     n_type:           塑性率を計算する型
c     n_member:         部材数
c     P_strain:         1：最大塑性率　2：時刻　3:最大累積塑性率　4:増分前の塑性率
c     IP_strain:        1：累積回数　　2：回数のためのワーク領域
c     nx_member:        部材内塑性計算を行う個数
c     Sec_epsy:         塑性時ひずみの逆数
c―――――――――――――――――――――――――――――★断面個数入力
      ihan=1
      nfx=98
      open(unit=nfx,file=fnfile(NFL),FORM='UNFORMATTED',
     *    status='old',err=199)
c      write(77,'(a,2i4,f12.4)')fnfile(NFL),n_member,n_time,dt
      do i=1, n_member
      read(nfx,end=1000) vf,(eps(j),j=1,3)
c     write(77,'(i4,4f12.4)') i,vf,(eps(j),j=1,3)
      nx_member(i)=vf
      do j=1,3
      Sec_epsy(j,i)=eps(j)
      enddo
      do k=1,5
       do j=1,7
       P_strain(j,k,i)=0.
       enddo
       do j=1,2
```

```
        IP_strain(j,k,i)=0.
       enddo
      enddo
     enddo
c―――――――――――――――――――――――――――――★断面個数入力
      tt=0.
      do nt=1, n_time
      tt=dt*nt
        do i=1,n_member
        n_sec= nx_member(i)
        if(n_sec.ne.0) then
         do j=1,n_sec
           read(nfx,end=1001) (eps(k),k=1,3)
c     write(77,'(3i4,4f12.4)') nt,i,j,(eps(k),k=1,3)
          call check_strain(n_type ,tt,eps, Sec_epsy(1,i),
     *      P_strain(1,j,i),IP_strain(1,j,i),i)
         enddo
        endif
       enddo
      enddo
 1000 continue
      close(nfx)
c     write(77,'(a)')' shuuryou '
      ihan=0
      return
 1001 continue
      close(nfx)
c     write(77,'(a)')' データ不足 '
      ihan=0
      return
 199  continue
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE / check_strain
C       ―――――――――――――――――――――――――――――――――
C     ●   塑性率と最大累積塑性率の計算
C     ―――――――――――――――――――――――――――――――――
      subroutine check_strain(n_type,tt,eps,Sec_epsy,P_strain,
     *    IP_strain,nm)
      implicit real*8(A-H,O-Z)
      dimension P_strain(7),Sec_epsy(3),IP_strain(2),ss(3)
      real*4 eps(3)
      do i=1,3
      ss(i)=eps(i)*Sec_epsy(i)
      P_strain(i+4)=eps(i)
      enddo
      goto(11,12,13), n_type
11    continue
      P_strainx= ss(1)**2+dsqrt(ss(2)**2 + ss(3)**2)
      goto 100
12    continue
      P_strainx= dsqrt(ss(1)**2+ ss(2)**2 + ss(3)**2)
```

```
      goto 100
13    continue
      P_strainx= dabs(ss(1))+dsqrt(ss(2)**2+ ss(3)**2)
      goto 100
100   continue
c―――――――――――――――――――――――――――――――★最大塑性率
      if(P_strainx.gt. P_strain(1))then
      P_strain(1) = P_strainx
      P_strain(2) = tt
      endif
c―――――――――――――――――――――――――――――――★累積塑性率
      if(P_strainx .ge. P_strain(4)) then
c―――――――――――――――――――――――――――――――★ひずみ増加
       if(P_strain(4).gt.1. .and. P_strainx .gt.1.) then
        P_strain(3)= P_strain(3)+ P_strainx - P_strain(4)
       if(IP_strain(2).le.0) IP_strain(2)=0
        IP_strain(2)= IP_strain(2)+1
       elseif(P_strainx .gt.1.) then
        P_strain(3)= P_strain(3)+ P_strainx - 1.
       if(IP_strain(2).le.0) IP_strain(2)=0
        IP_strain(2)= IP_strain(2)+1
       endif
c―――――――――――――――――――――――――――――――★ひずみ減少
      else
      if(P_strain(4).gt.1.) then
      if(IP_strain(2).gt.0.) then
c―――――――――――――――――――――――――――――――★累積損傷回数チェック
        IP_strain(1)= IP_strain(1)+1
        IP_strain(2)=-1
        else
        IP_strain(2)= IP_strain(2)-1
        endif
       endif

      endif
      if(nm.eq.18)then
c      write(77,'(a,3i4,4f12.5)') ' 累積 ',nm,IP_strain(1),IP_strain(2),
c     *  P_strain(4),P_strainx
      endif
      P_strain(4) = P_strainx
      return
      end
```

上記のサブルーチンで計算した各部材の最大塑性率、最大累積塑性率などを用いて、出力用の表をセットする。

サブルーチン Set_Hyo_strain()では、まず、表となる配列 s_hyou と is_hyou をゼロセットする。次に、全部材について、サブルーチン Cal_hyo_strain()を用いて塑性率などの最大値をチェックし、表内の適切な位置にセットする。ここで、当該部材内でひずみが出力されている

か、配列 nx_member を用いてチェックし、出力していない場合は、その部材をスキップする。サブルーチン check_floor()を用いて、当該部材のコード iso(i)から部材種別（はり、柱、ブレースなど）や層、通り芯番号などを取得する。この情報を元に得られた最大値を表にセットすることになる。最後は、得られた表を指定したファイルに出力する。

サブルーチン Cal_Hyo_strain()では、各層、部材種別毎に、塑性率などの最大値をチェックする。

ここで説明した３つのサブルーチンの内容を以下に示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE / Set_Hyo_strain
C       ―――――――――――――――――――――――――――――――
C     ●   塑性率と最大累積塑性率の表作成
C     ―――――――――――――――――――――――――――――――
      subroutine Set_Hyo_strain(i_out,ifn,ilen,
     *                          m_radio_J,n_member,iso,P_strain,
     *                          IP_strain,nx_member,S_hyou,IS_hyou,
     *                           n_radio_tori,n_edit_tori_set,jjso)
      implicit real*8(A-H,O-Z)
      dimension P_strain(7,5,*), IP_strain(2,5,*),
     *          nx_member(*),iso(*)
      dimension IS_hyou(3,13,*)
      real*4 S_hyou(3,6,*)
      character FN*200,fnn*1(200)
      integer*1 ifn(200)
      integer*4 ilen
      equivalence (fnn,fn)

      do i=1,jjso
      do j=1,6
      do k=1,3
      s_hyou(k,j,i)=0.
      enddo
      enddo
      enddo
      do i=1,jjso
      do j=1,13
      do k=1,3
      is_hyou(k,j,i)=0
      enddo
      enddo
      enddo

      do i=1,n_member
      if(nx_member(i).ne.0) then
         call check_floor(iso(i),ix,iy,ibb,jso,itype)
      do j=1,nx_member(i)
c     write(77,'(7i4)') i,j,jso,itype,ix,iy,ibb
      call Cal_hyo_strain(i,j,m_radio_J,ix,iy,itype,
     *                    n_radio_tori,n_edit_tori_set,
```

```
     *                    P_strain(1,j,i),IP_strain(1,j,i),
     *                    IS_hyou(1,1,jso),S_hyou(1,1,jso))
      enddo
      endif
      enddo
c-----------------------------★ファイル出力
      if(i_out.eq.1) then
      fn=' '
      do i=1,ilen
      fnn(i)=char(ifn(i))
      enddo
      open(77,file=FN,err=810)

      write(77,'(a,3i4)') ' 通り芯番号',m_radio_J,
     *     n_radio_tori,n_edit_tori_set
      write(77,'(/a/a,a)') ' 最大塑性率',
     *'                柱                                        はり'
     *,'                                  ブレース'
      do i=jjso,1,-1
      write(77,'(i4,3(f10.3,a,f8.3,a,i5,i2,2x,2i3,3x))')
     *    i,((S_hyou(k,1,i),' (',
     *    S_hyou(k,2,i),' )',
     *    IS_hyou(k,8,i),IS_hyou(k,9,i),
     *    IS_hyou(k,1,i),IS_hyou(k,2,i)),k=1,3)
      enddo
      write(77,'(/a/a,a)') ' 最大累積塑性率',
     *'                柱                              はり',
     *'                     ブレース'
      do i=jjso,1,-1
      write(77,'(i4,3(f10.3,i5,i2,2x,2i3,3x))') i,((S_hyou(k,3,i),
     *    IS_hyou(k,10,i),IS_hyou(k,11,i),
     *    IS_hyou(k,3,i),IS_hyou(k,4,i)),k=1,3)
      enddo
      write(77,'(/a/a,a)') ' 最大累積回数',
     *'              柱                    はり',
     *'               ブレース'
      do i=jjso,1,-1
      write(77,'(i4,3(i4,i5,i2,2x,2i3,3x))') i,((IS_hyou(k,5,i),
     *    IS_hyou(k,12,i),IS_hyou(k,13,i),
     *    IS_hyou(k,6,i),IS_hyou(k,7,i)),k=1,3)
      enddo
      close(77)
      return
 810  continue
      write(77,'(a)') ' 入力ファイル名エラー'
      endif

      return
      end
```

```
C     ------------------------------------
C     ●   SUBROUTINE / Cal_Hyo_strain
C       ------------------------------------
```

```
C     ●   塑性率と最大累積塑性率の表作成
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_hyo_strain(i_mem,j_mem,m_radio_J,ix,iy,itype,
     *                    n_radio_tori,n_edit_tori_set,
     *                    P_strain,IP_strain,IS_hyou,S_hyou)
      implicit real*8(A-H,O-Z)
      dimension P_strain(7),IP_strain(2)
      dimension IS_hyou(3,13)
      real*4    S_hyou(3,6)
C     ―――――――――――――――――――――――――――――――――
c   i:1 はしら  2:はり 3:ブレース
c   S_hyou(i,1):最大塑性率
c   S_hyou(i,2):その時刻
c   S_hyou(i,3):最大累積塑性率
c   IS_hyou(i,1):最大塑性率発生 x 方向通り芯番号
c   IS_hyou(i,2):最大塑性率発生 y 方向通り芯番号
c   IS_hyou(i,3):最大累積塑性率発生 x 方向通り芯番号
c   IS_hyou(i,4):最大累積塑性率発生 y 方向通り芯番号
c   IS_hyou(i,5):最大累積塑性回数
c   IS_hyou(i,6):最大累積塑性回数発生 x 方向通り芯番号
c   IS_hyou(i,7):最大累積塑性回数発生 y 方向通り芯番号
c
c
      if(m_radio_J.eq.0)then
c     write(77,'(3i4,3f12.3,i4)') itype,ix,iy,P_strain(1),
c     *  P_strain(2),P_strain(3),IP_strain(1)
C     ―――――――――――――――――――――――全部材について
      goto(11,12,12,14),itype
C     ―――――――――――――――――柱
  11  continue
      ihashira=1
C     ―――――――――――――塑性率
      if(P_strain(1).gt.S_hyou(ihashira,1))then
      S_hyou(ihashira,1)=P_strain(1)
      S_hyou(ihashira,2)=P_strain(2)
      IS_hyou(ihashira,1)=ix
      IS_hyou(ihashira,2)=iy
      S_hyou(ihashira,4)=P_strain(5)
      S_hyou(ihashira,5)=P_strain(6)
      S_hyou(ihashira,6)=P_strain(7)
      IS_hyou(ihashira,8)=i_mem
      IS_hyou(ihashira,9)=j_mem
c     write(77,'(a,i4,3f12.3)')' 塑性率',ihashira,
c     *   S_hyou(ihashira,1),S_hyou(ihashira,2)
      endif
C     ―――――――――――――累積塑性率
      if(P_strain(3).gt.S_hyou(ihashira,3))then
      S_hyou(ihashira,3)=P_strain(3)
      IS_hyou(ihashira,3)=ix
      IS_hyou(ihashira,4)=iy
      IS_hyou(ihashira,10)=i_mem
      IS_hyou(ihashira,11)=j_mem
c     write(77,'(a,i4,3f12.3)')' 累積率',ihashira,
c     *   S_hyou(ihashira,3)
```

```
      endif
C     ――――――――――――――塑性回数
      if(IP_strain(1).gt.IS_hyou(ihashira,5))then
      iS_hyou(ihashira,5)=IP_strain(1)
      IS_hyou(ihashira,6)=ix
      IS_hyou(ihashira,7)=iy
      IS_hyou(ihashira,12)=i_mem
      IS_hyou(ihashira,13)=j_mem
c     write(77,'(a,2i4)')' 塑性率',ihashira,
c     *   iS_hyou(ihashira,5)
      endif
      return
C     ――――――――――――――――――はり
  12  continue
      ihashira=2
C     ――――――――――――塑性率
      if(P_strain(1).gt.S_hyou(ihashira,1))then
      S_hyou(ihashira,1)=P_strain(1)
      S_hyou(ihashira,2)=P_strain(2)
      IS_hyou(ihashira,1)=ix
      IS_hyou(ihashira,2)=iy
      S_hyou(ihashira,4)=P_strain(5)
      S_hyou(ihashira,5)=P_strain(6)
      S_hyou(ihashira,6)=P_strain(7)
      IS_hyou(ihashira,8)=i_mem
      IS_hyou(ihashira,9)=j_mem
c     write(77,'(a,i4,3f12.3)')' 塑性率',ihashira,
c     *   S_hyou(ihashira,1),S_hyou(ihashira,2)
      endif
C     ――――――――――――累積塑性率
      if(P_strain(3).gt.S_hyou(ihashira,3))then
      S_hyou(ihashira,3)=P_strain(3)
      IS_hyou(ihashira,3)=ix
      IS_hyou(ihashira,4)=iy
      IS_hyou(ihashira,10)=i_mem
      IS_hyou(ihashira,11)=j_mem
c     write(77,'(a,i4,3f12.3)')' 累積率',ihashira,
c     *   S_hyou(ihashira,3)
      endif
C     ――――――――――――――塑性回数
      if(IP_strain(1).gt.IS_hyou(ihashira,5))then
      iS_hyou(ihashira,5)=IP_strain(1)
      IS_hyou(ihashira,6)=ix
      IS_hyou(ihashira,7)=iy
      IS_hyou(ihashira,12)=i_mem
      IS_hyou(ihashira,13)=j_mem
c     write(77,'(a,2i4)')' 塑性率',ihashira,
c     *   iS_hyou(ihashira,5)
      endif
      return

C     ――――――――――――――――――ブレース
  14  continue
      ihashira=3
```

```
C     ――――――――――――――塑性率
      if(P_strain(1).gt.S_hyou(ihashira,1))then
      S_hyou(ihashira,1)=P_strain(1)
      S_hyou(ihashira,2)=P_strain(2)
      IS_hyou(ihashira,1)=ix
      IS_hyou(ihashira,2)=iy
      S_hyou(ihashira,4)=P_strain(5)
      S_hyou(ihashira,5)=P_strain(6)
      S_hyou(ihashira,6)=P_strain(7)
      IS_hyou(ihashira,8)=i_mem
      IS_hyou(ihashira,9)=j_mem
c     write(77,'(a,i4,3f12.3)')' 塑性率',ihashira,
c     *   S_hyou(ihashira,1),S_hyou(ihashira,2)
      endif
C     ――――――――――――――累積塑性率
      if(P_strain(3).gt.S_hyou(ihashira,3))then
      S_hyou(ihashira,3)=P_strain(3)
      IS_hyou(ihashira,3)=ix
      IS_hyou(ihashira,4)=iy
      IS_hyou(ihashira,10)=i_mem
      IS_hyou(ihashira,11)=j_mem
c     write(77,'(a,i4,3f12.3)')' 累積率',ihashira,
c     *   S_hyou(ihashira,3)
      endif
C     ――――――――――――――塑性回数
      if(IP_strain(1).gt.IS_hyou(ihashira,5))then
      iS_hyou(ihashira,5)=IP_strain(1)
      IS_hyou(ihashira,6)=ix
      IS_hyou(ihashira,7)=iy
      IS_hyou(ihashira,12)=i_mem
      IS_hyou(ihashira,13)=j_mem
c     write(77,'(a,2i4)')' 塑性率',ihashira,
c     *   iS_hyou(ihashira,5)
      endif
      return
C     ――――――――――――――――――――――――通り芯について
      else
       if(n_radio_tori .eq. 1 .and. ix .eq. n_edit_tori_set) goto 110
       if(n_radio_tori .eq. 2 .and. iy .eq. n_edit_tori_set) goto 110
      return
 110  continue
c     write(77,'(3i4,3f12.3,i4)') itype,ix,iy,P_strain(1),
c     *  P_strain(2),P_strain(3),IP_strain(1)
C     ――――――――――――――――――――――――全部材について
      goto(21,22,22,24),itype
C     ――――――――――――――――――柱
  21  continue
      ihashira=1
C     ――――――――――――――塑性率
      if(P_strain(1).gt.S_hyou(ihashira,1))then
      S_hyou(ihashira,1)=P_strain(1)
      S_hyou(ihashira,2)=P_strain(2)
      IS_hyou(ihashira,1)=ix
      IS_hyou(ihashira,2)=iy
```

```
      S_hyou(ihashira,4)=P_strain(5)
      S_hyou(ihashira,5)=P_strain(6)
      S_hyou(ihashira,6)=P_strain(7)
      IS_hyou(ihashira,8)=i_mem
      IS_hyou(ihashira,9)=j_mem
c     write(77,'(a,i4,3f12.3)')' 塑性率',ihashira,
c     *   S_hyou(ihashira,1),S_hyou(ihashira,2)
      endif
C     ――――――――――――――累積塑性率
      if(P_strain(3).gt.S_hyou(ihashira,3))then
      S_hyou(ihashira,3)=P_strain(3)
      IS_hyou(ihashira,3)=ix
      IS_hyou(ihashira,4)=iy
      IS_hyou(ihashira,10)=i_mem
      IS_hyou(ihashira,11)=j_mem
c     write(77,'(a,i4,3f12.3)')' 累積率',ihashira,
c     *   S_hyou(ihashira,3)
      endif
C     ――――――――――――――塑性回数
      if(IP_strain(1).gt.IS_hyou(ihashira,5))then
      iS_hyou(ihashira,5)=IP_strain(1)
      IS_hyou(ihashira,6)=ix
      IS_hyou(ihashira,7)=iy
      IS_hyou(ihashira,12)=i_mem
      IS_hyou(ihashira,13)=j_mem
c     write(77,'(a,2i4)')' 塑性率',ihashira,
c     *   iS_hyou(ihashira,5)
      endif
      return
C     ――――――――――――――――――はり
   22 continue
      ihashira=2
C     ――――――――――――――塑性率
      if(P_strain(1).gt.S_hyou(ihashira,1))then
      S_hyou(ihashira,1)=P_strain(1)
      S_hyou(ihashira,2)=P_strain(2)
      IS_hyou(ihashira,1)=ix
      IS_hyou(ihashira,2)=iy
      S_hyou(ihashira,4)=P_strain(5)
      S_hyou(ihashira,5)=P_strain(6)
      S_hyou(ihashira,6)=P_strain(7)
      IS_hyou(ihashira,8)=i_mem
      IS_hyou(ihashira,9)=j_mem
c     write(77,'(a,i4,3f12.3)')' 塑性率',ihashira,
c     *   S_hyou(ihashira,1),S_hyou(ihashira,2)
      endif
C     ――――――――――――――累積塑性率
      if(P_strain(3).gt.S_hyou(ihashira,3))then
      S_hyou(ihashira,3)=P_strain(3)
      IS_hyou(ihashira,3)=ix
      IS_hyou(ihashira,4)=iy
      IS_hyou(ihashira,10)=i_mem
      IS_hyou(ihashira,11)=j_mem
c     write(77,'(a,i4,3f12.3)')' 累積率',ihashira,
```

```
c     *   S_hyou(ihashira,3)
      endif
C     ――――――――――――――塑性回数
      if(IP_strain(1).gt.IS_hyou(ihashira,5))then
      iS_hyou(ihashira,5)=IP_strain(1)
      IS_hyou(ihashira,6)=ix
      IS_hyou(ihashira,7)=iy
      IS_hyou(ihashira,12)=i_mem
      IS_hyou(ihashira,13)=j_mem
c     write(77,'(a,2i4)')' 塑性率',ihashira,
c     *   iS_hyou(ihashira,5)
      endif
      return
C     ――――――――――――――――――ブレース
  24  continue
      ihashira=3
C     ――――――――――――――塑性率
      if(P_strain(1).gt.S_hyou(ihashira,1))then
      S_hyou(ihashira,1)=P_strain(1)
      S_hyou(ihashira,2)=P_strain(2)
      IS_hyou(ihashira,1)=ix
      IS_hyou(ihashira,2)=iy
      S_hyou(ihashira,4)=P_strain(5)
      S_hyou(ihashira,5)=P_strain(6)
      S_hyou(ihashira,6)=P_strain(7)
      IS_hyou(ihashira,8)=i_mem
      IS_hyou(ihashira,9)=j_mem
c     write(77,'(a,i4,3f12.3)')' 塑性率',ihashira,
c     *   S_hyou(ihashira,1),S_hyou(ihashira,2)
      endif
C     ――――――――――――――累積塑性率
      if(P_strain(3).gt.S_hyou(ihashira,3))then
      S_hyou(ihashira,3)=P_strain(3)
      IS_hyou(ihashira,3)=ix
      IS_hyou(ihashira,4)=iy
      IS_hyou(ihashira,10)=i_mem
      IS_hyou(ihashira,11)=j_mem
c     write(77,'(a,i4,3f12.3)')' 累積率',ihashira,
c     *   S_hyou(ihashira,3)
      endif
C     ――――――――――――――塑性回数
      if(IP_strain(1).gt.IS_hyou(ihashira,5))then
      iS_hyou(ihashira,5)=IP_strain(1)
      IS_hyou(ihashira,6)=ix
      IS_hyou(ihashira,7)=iy
      IS_hyou(ihashira,12)=i_mem
      IS_hyou(ihashira,13)=j_mem
c     write(77,'(a,2i4)')' 塑性率',ihashira,
c     *   iS_hyou(ihashira,5)
      endif
      return
      endif
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――
*         SUBROUTINE check_floor                          *
C     ―――――――――――――――――――――――――――――――――
      subroutine check_floor(iso,ix,iy,ibb,jso,itype)
      ix=0    !ｘ方向通り番号
      iy=0    !ｙ方向通り番号
      ibb=0   !部材内通り番号
      jso=0   !層
      itype= 0!部材型
      if(iabs(iso).lt.100000) return
      iiso = iso/10000000
      goto(9,10,11,12,13,14),iiso+4
   9  continue
c                  せん断
      itype= 6
      goto 100

  10  continue
c                  ブレース
      itype= 4
      goto 100
  11  continue
c                 柱
      itype= 1
      goto 100
  12  continue
c                  制振ダンパー
      itype= 5
      goto 100
  13  continue
c                  ｘ方向梁
      itype= 2
      goto 100
  14  continue
c                  ｙ方向梁
      itype= 3
      goto 100

 100  continue
      iso1=iabs(iso)
      iiso=iso1-(iso1/10000000)*10000000
      jso=iiso/100000
      iiso=iiso-jso*100000
      iy = iiso/1000
      iiso= iiso - iy*1000
      ix = iiso/10
      ibb= iiso - ix*10

      return
      end
```

次は、全フレーム、あるいはユーザーが指定したフレームについて、部材毎の最大塑性率、最大累積塑性率などを求めるサブルーチンを示す。ここでは、ダイアログを用いて、最大塑性率の最低値を指定することで、出力部材の数を少なくすることができる。処理内容は、上記のサブルーチンとほとんど同じであり、理解することは難しくないであろう。

該当するサブルーチン Set_member_strain()と Cal_member_strain()を以下に示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE / Set_member_strain
C       ―――――――――――――――――――――――――――――――
C     ●   塑性率と最大累積塑性率の表作成
C     ―――――――――――――――――――――――――――――――――
      subroutine Set_member_strain(i_out,ifn,ilen,
     *                         m_radio_J,n_member,iso,P_strain,
     *                         IP_strain,nx_member,S_hyou,IS_hyou,
     *                          n_radio_tori,n_edit_tori_set,m_edit_s)
      implicit real*8(A-H,O-Z)
      dimension P_strain(7,5,*), IP_strain(2,5,*),
     *          nx_member(*),iso(*)
      dimension IS_hyou(6,*)
      real*4 S_hyou(3,*),m_edit_s
      character FN*200,fnn*1(200)
      integer*1 ifn(200)
      integer*4 ilen
      equivalence (fnn,fn)

      do i=1,n_member
      do j=1,3
      s_hyou(j,i)=0.
      enddo
      enddo
      do i=1,n_member
      do j=1,6
      is_hyou(j,i)=0
      enddo
      enddo
      write(77,'(a,3i4)') ' 通り芯番号',m_radio_J,
     *     n_radio_tori,n_edit_tori_set
      do i=1,n_member
      if(nx_member(i).ne.0) then
         call check_floor(iso(i),ix,iy,ibb,jso,itype)
       IS_hyou(6,i)=itype
       IS_hyou(3,i)=ibb
       if(m_radio_J .eq. 2) goto 110      ! 全部材出力
       if(n_radio_tori .eq. 1 .and. ix .eq. n_edit_tori_set) goto 110 ! 通り芯出力
       if(n_radio_tori .eq. 2 .and. iy .eq. n_edit_tori_set) goto 110
        nx_member(i)=0
      goto 112
  110 continue
      do j=1,nx_member(i)
```

```
c     write(77,'(7i4)') i,j,jso,itype,ix,iy,ibb
         call Cal_member_strain(nx,j,ix,iy,itype,
     *                     P_strain(1,j,i),IP_strain(1,j,i),
     *                     IS_hyou(1,i),S_hyou(1,i))
      enddo
       IS_hyou(4,i)=nx

      if(m_edit_s.gt.S_hyou(1,i)) nx_member(i)=0  ! 最大塑性率の制限チェック
 112  continue
      endif
      enddo
c-------------------------------★ファイル出力
      if(i_out.eq.1) then
      fn=' '
      do i=1,ilen
      fnn(i)=char(ifn(i))
      enddo
      open(77,file=FN,err=810)
      write(77,'(/a/)')'部材塑性率の表示'
      do j=1,4
      if(j.eq.1) write(77,'(/a/a,a)')'  柱',' 部材番号     ix    iy',
     *'      最大塑性率   発生時刻   累積塑性率   発生回数'
      if(j.eq.2) write(77,'(/a/a,a)')'  x方向はり',
     *'  部材番号     ix    iy',
     *'      最大塑性率   発生時刻   累積塑性率   発生回数'
      if(j.eq.3) write(77,'(/a/a,a)')'  y方向はり',
     *'  部材番号     ix    iy',
     *'      最大塑性率   発生時刻   累積塑性率   発生回数'
      if(j.eq.4) write(77,'(/a/a,a)')'  ブレース',
     *'  部材番号     ix    iy',
     *'      最大塑性率   発生時刻   累積塑性率   発生回数'
      do i=1,n_member
c     write(77,'(4i4)') i,j,IS_hyou(9,i),nx_member(i)
      if(nx_member(i).ne.0.and.j.eq.IS_hyou(6,i)) then
c     if(nx_member(i).ne.0) then

      write(77,'(2i4,3x,3i4,3x,3f10.3,i6)')
     *    i,IS_hyou(4,i),IS_hyou(1,i),IS_hyou(2,i),IS_hyou(3,i),
     *    S_hyou(1,i),S_hyou(2,i),S_hyou(3,i),IS_hyou(5,i)
      endif
      enddo
      enddo
      close(77)
      return
 810  continue
      write(77,'(a)') ' 入力ファイル名エラー'
      endif
      return
      end
```

```
C     ------------------------------------
C     ●   SUBROUTINE / Cal_member_strain
C       ------------------------------------
```

```
C     ●　塑性率と最大累積塑性率の表作成
C     ―――――――――――――――――――――――――――――――
      subroutine Cal_member_strain(nxx,jx,ix,iy,itype,
     *                    P_strain,IP_strain,IS_hyou,S_hyou)

      implicit real*8(A-H,O-Z)
      dimension P_strain(7),IP_strain(2)
      dimension IS_hyou(6)
      real*4    S_hyou(3)
C     ―――――――――――――――――――――――――――――――
c   i:1 はしら  2:はり 3:ブレース
c   S_hyou(i,1):最大塑性率
c   S_hyou(i,2):その時刻
c   S_hyou(i,3):最大累積塑性率
c   IS_hyou(i,1):最大塑性率発生 x 方向通り芯番号
c   IS_hyou(i,2):最大塑性率発生 y 方向通り芯番号
c   IS_hyou(i,3):最大累積塑性率発生 x 方向通り芯番号
c   IS_hyou(i,4):最大累積塑性率発生 y 方向通り芯番号
c   IS_hyou(i,5):最大累積塑性回数
c   IS_hyou(i,6):最大累積塑性回数発生 x 方向通り芯番号
c   IS_hyou(i,7):最大累積塑性回数発生 y 方向通り芯番号
c
c

C     ―――――――――――――塑性率
      if(P_strain(1).gt.S_hyou(1))then
      S_hyou(1)=P_strain(1)
      S_hyou(2)=P_strain(2)
      IS_hyou(1)=ix
      IS_hyou(2)=iy
      nxx=jx
c     write(77,'(a,2i4,3f12.3)')' 塑性率',jx,itype,
c     *   S_hyou(1),S_hyou(2)
      endif
C     ―――――――――――――累積塑性率
      if(P_strain(3).gt.S_hyou(3))then
      S_hyou(3)=P_strain(3)
c     write(77,'(a,2i4,3f12.3)')' 累積率',jx,itype,
c     *   S_hyou(3)
      endif
C     ―――――――――――――塑性回数
      if(IP_strain(1).gt.IS_hyou(5))then
      iS_hyou(5)=IP_strain(1)
c     write(77,'(a,3i4)')' 回数',jx,itype,
c     *   iS_hyou(5)
      endif

      return
      end
```

次は、部材の塑性率と累積塑性率の時刻履歴を求めるサブルーチンを示す。この部材は、ダイアログで指定することになる。塑性率などと同

様に、３つのひずみの時刻歴も配列 S_hyo にセットする。

サブルーチン Cal_member_P_strain()では、まず、ファイルから、全部材について、ひずみの出力個数と塑性ひずみと塑性ヒンジが生じたときの各々ｙ軸と ｚ 軸に関する曲率の逆数を入力し、配列 nx_member と Sec_epsy にセットする。次に、処理結果を設定する配列 S_hyo をゼロセットする。これ以降、出力時刻毎にひずみを入力した後、サブルーチン check_member_strain ()を用いて、最大塑性率、累世塑性率を計算し、時刻歴として配列 S_hyo にセットする。最後に、ユーザーが指定したファイルにデータを出力する。

ここで、上で説明した２つのサブルーチンを示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_member_P_strain
C       ―――――――――――――――――――――――――――――――
C     ●   任意部材の塑性率と最大累積塑性率の履歴
C     ―――――――――――――――――――――――――――――――
      subroutine Cal_member_P_strain(i_out,ifn,ilen,n_mem,n_ichi,ihan,
     *    nfl,n_type,n_member,S_hyo,nx_member,Sec_epsy,n_time,dt)

      implicit real*8(A-H,O-Z)
      common /sf01/fnfile,jdfile,kdfile,iidat,title,lengf,ltitle,timex
      CHARACTER fnfile(60)*50,title*50,timex*20(60)
      integer*4 jdfile(60),kdfile(60),lengf(60)
      dimension Sec_epsy(3,*),nx_member(*),ssx(3)
      real*4  eps(3),vf,S_hyo(5,*),dt
      character FN*200,fnn*1(200)
      integer*1 ifn(200)
      integer*4 ilen
      equivalence (fnn,fn)
c―――――――――――――――――――――――――――――★
c     n_mem:           計算したい部材
c     n_ichi:          部材の位置
c     nfx:             ひずみ入力ファイル番号
c     n_time:          時間個数
c     n_type:          塑性率を計算する型
c     n_member:        部材数
c     S_hyo:        1：最大塑性率　2:最大累積塑性率
c     nx_member:       部材内塑性計算を行う個数
c     Sec_epsy:        塑性時ひずみの逆数
c―――――――――――――――――――――――――――――★断面個数入力
      ihan=1
      nfx=98
      open(unit=nfx,file=fnfile(NFL),FORM='UNFORMATTED',
     *    status='old',err=199)
      ihan=2
      if(n_mem.le.0 .or. n_mem .gt. n_member) return
      do i=1, n_member
      read(nfx) vf,(eps(j),j=1,3)
      do j=1,3
```

```
      Sec_epsy(j,i)=eps(j)
      enddo
      nx_member(i)=vf
      enddo
      ihan=1
      if(nx_member(n_mem).lt. n_ichi) return
      do i=1, n_time
      do j=1,2
      S_hyo(j,i)=0.
      enddo
      enddo

      ihan=0
c-----------------------------★断面個数入力
      S_hyox=0.
      S_hyoy=0.
      do nt=1, n_time
        do i=1,n_member
        n_sec= nx_member(i)
        if(n_sec.ne.0) then
         do j=1,n_sec
           read(nfx,end=1000) (eps(k),k=1,3)
          if(n_mem .eq.i .and. n_ichi .eq. j) then
           call check_member_strain(n_type , eps, Sec_epsy(1,i),
     *                              S_hyo(1,nt), S_hyox,S_hyoy)
           S_hyox= S_hyo(1,nt)
           S_hyoy= S_hyo(2,nt)
          do k=1,3
          S_hyo(K+2,nt)=eps(k)*Sec_epsy(1,i)
          enddo
           endif
         enddo
        endif
       enddo
      enddo
 1000 continue
c-----------------------------★ファイル出力
      if(i_out.eq.1) then
      fn=' '
      do i=1,ilen
      fnn(i)=char(ifn(i))
      enddo
      open(77,file=FN,err=199)

      write(77,'(a,i4,a,i4,a,i4)')
     * ' 時刻歴出力:部材番号：',n_mem,'  位置：',
     * n_ichi,'  分割数：',n_time
      do j=1,3
      ssx(j)=0.
      if( Sec_epsy(j,n_mem).ne.0.) ssx(j)=1./Sec_epsy(j,n_mem)
      enddo
      write(77,'(/a,f12.8,a,f12.8,a,f12.8/)')
     * ' εy:',ssx(1),' Φyp:',ssx(2),' Φzp:',ssx(3)
      write(77,'(a,a)')
```

```
     *'        時刻       塑性率    累積塑性率',
     *'      εx         Φy         Φz'
      do i=1,n_time
      tt=dt*i
      write(77,'(f12.4,2f12.5,3f12.8)') tt,(S_hyo(j,i),j=1,5)
      enddo
      close(77)
      return
  199 continue
      write(77,'(a)') ' 入力ファイル名エラー'
      endif
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE / check_strain
C       ―――――――――――――――――――――――――――――――――――
C     ●   塑性率と最大累積塑性率の計算
C     ―――――――――――――――――――――――――――――――――――
      subroutine check_member _strain(n_type,eps,Sec_epsy,
     *                                S_hyo, S_hyox,S_hyoy)
      implicit real*8(A-H,O-Z)
      dimension Sec_epsy(3), ss(3)
      real*4 eps(3),S_hyo(5)
      do i=1,3
      ss(i)=eps(i)*Sec_epsy(i)
      enddo
      goto(11,12,13), n_type
11    continue
      S_hyo(1) = ss(1)**2+dsqrt(ss(2)**2 + ss(3)**2)
      goto 100
12    continue
      S_hyo(1) = dsqrt(ss(1)**2+ ss(2)**2 + ss(3)**2)
      goto 100
13    continue
      S_hyo(1) = dabs(ss(1))+dsqrt(ss(2)**2+ ss(3)**2)
      goto 100
100   continue
      S_hyo(2)=S_hyoy
c―――――――――――――――――――――――――――――★累積塑性率計算
      if(S_hyo(1).gt. S_hyox) then
       if(S_hyox.gt.1. .and. S_hyo(1).gt.1.) then
        S_hyo(2)= S_hyo(2)+ S_hyo(1) - S_hyox
       elseif(S_hyo(1).gt.1.) then
        S_hyo(2)= S_hyo(2)+ S_hyo(1) - 1.
      endif
      endif
      return
      end
```

前節で説明した 5 つの処理が全て終了すると、CDialog::OnOK()関数がコールされる。

```
   SET_MEMBER_STRAIN(&i_out,&path_name,&istrlen,&m_radio_J,&n_member,
         F_iso,P_strain,IP_strain,
          nx_member,S_hyou,IS_hyou,&n_radio_tori,&n_edit_tori_set,&m_edit_s_soseix);
      break;
      case 3://部材毎：通り芯毎（表）
      I_print = 14;//最大塑性率
   SET_MEMBER_STRAIN(&i_out,&path_name,&istrlen,&m_radio_J,&n_member,
         F_iso,P_strain,IP_strain,
          nx_member,S_hyou,IS_hyou,&n_radio_tori,&n_edit_tori_set,&m_edit_s_soseix);
      break;
      case 4://部材時刻歴（グラフ）
      I_print = 15;//最大塑性率
   CAL_MEMBER_P_STRAIN(&i_out,&path_name,&istrlen,&n_mem,&n_ichi,&ihan,
        &nfl,&n_type,&n_member,S_hyou,nx_member,Sec_epsy,&n_time,&dt);
      break;
}
      CDialog::OnOK();
```

関数 Cdialog::OnOK()が実行されると、後で示す CSf40View::OnDraw ()が自動的にコールされることになる。関数 OnDraw()関数が実行され、最初に、関数 void CSf40View::OnDlgPrint()がコールされる。この関数によって、プリンター出力の準備が行われる。まず、この関数を見てみよう。この関数では、変数 I_print の値によって、異なるメッセージが発せられる。ひとつは ID_FILE_PRINT であり、他は ID_FILE_PRINT_PREVIEW である。前者は、直接プリンターに書類が出力される。後者は、プレビューで出力書類が画面に表示されることになる。

```
 void CSf40View::OnDlgPrint()
 {
     if(dlg_print.DoModal() == IDOK){
     if(I_print < 100){
 //AfxGetMainWnd()->PostMessage(WM_COMMAND,ID_FILE_PRINT);
     Flag_dialog = 1;
 //  if(I_print == 10 || I_print == 11|| I_print == 13|| I_print == 14|| I_print == 15){
     if(I_print == 10 || I_print == 11){
  AfxGetMainWnd()->PostMessage(WM_COMMAND,ID_FILE_PRINT);
 //I_print = 0;
     }
     else{
  AfxGetMainWnd()->PostMessage(WM_COMMAND,ID_FILE_PRINT_PREVIEW);
 //I_print = 0;
     }
     }
     }
     else{
```

```
        AfxGetMainWnd()->PostMessage(WM_CLOSE);
        }
    }
```

メッセージは次のメッセージマップによって関数 OnPreparePrinting() が自動的に実行され、プリンターの準備が行われることになる。

```
BEGIN_MESSAGE_MAP(CSf40View, CView)
    //{{AFX_MSG_MAP(CSf40View)
    ON_COMMAND(ID_DLG_PRINT, OnDlgPrint)
    ON_COMMAND(ID_JISUU, OnJisuu)
    ON_WM_RBUTTONDOWN()
    ON_WM_LBUTTONDOWN()
    ON_WM_MOUSEMOVE()
    ON_WM_LBUTTONUP()
    ON_COMMAND(ID_BUTTON_PRINT, OnButtonPrint)
    ON_WM_DESTROY()
    ON_COMMAND(ID_BUTTON_PROPERTY, OnButtonProperty)
    ON_WM_LBUTTONDBLCLK()
    //}}AFX_MSG_MAP
    // 標準印刷コマンド
    ON_COMMAND(ID_FILE_PRINT, CView::OnFilePrint)
//  ON_COMMAND(ID_FILE_PRINT, CView::OnFilePrintPreview)
    ON_COMMAND(ID_FILE_PRINT_DIRECT, CView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_PREVIEW, CView::OnFilePrintPreview)
END_MESSAGE_MAP()
```

プリンターの準備が行われた後、具体的に図や表を出力するルーチンをコールすることになる。最大塑性率などを出力するルーチンは、ケース 13 から 15 でとなる。

```
void CSf40View::OnDraw(CDC* pDC)
{
    CSf40Doc* pDoc = GetDocument();
    ASSERT_VALID(pDoc);

    HWND hwnd = this->GetSafeHwnd();

// TODO: この場所にネイティブ データ用の描画コードを追加します。
    if(Flag_dialog == 0){
    Flag_dialog = 1;
    OnDlgPrint();
    }
    if(I_print >= 100){
//  MessageBox("3");
    F_print_hen =0;
    disp_mem_persp(pDC);
    }
    else if(pDC->IsPrinting()){

    switch(I_print){
    case 1://固有値解析結果（表）
```

```
    print_koyu_hyo(pDC);
    break;
  case 2://応答最大層間変形角及び層間変位
    print_souhen_hyo(pDC);
    break;
  case 3://応答最大層せん断力及び層せん断力係数
    print_sousen_hyo(pDC);
    break;
  case 4://応答最大加速度・速度・変位
      print_kasoku_hyo(pDC);
      break;
  case 5://モード図
      if(iim == 1) print_mode_zu(pDC);
  //  else  pre4_disp_mem_persp(hwnd,ii);
      break;
  case 6://応答最大層間変形角・応答最大加速度・応答最大転倒モーメント・応答最大せん
断力のグラフ
    print_graph(pDC);
    break;
  case 7://Ｑ－δ図（静的）
    print_qd(pDC);
    break;
  case 8://制振装置応答最大相対（速度・変位）
    print_vd(pDC);
    break;
  case 9://解析波形とスペクトル
    print_fft_spc(pDC);
     break;
  case 10://応答最大加速度・速度・変位（図）
    disp_mem_persp_pr(hwnd,pDC);
    break;
  case 11://モード図（フレームモデル）
    disp_mem_persp_pr(hwnd,pDC);
    break;
  case 12://伏図・軸組図
    print_fj(pDC);
    break;
  case 13://最大塑性率・最大累積塑性率（全体、通り芯毎）
    print_soseiritu_hyo(pDC);
    break;
  case 14://全部材あるいは通り芯に関する部材毎の最大塑性率・最大累積塑性率
    print_buzai_soseiritu_hyo(pDC);
    break;
  case 15://任意部材の最大塑性率・最大累積塑性率の時刻歴図
    print_soseiritu_graph(pDC);
    break;
    default:
    break;
  }
  }
  }
```

次に、3 つの関数を示す。その内容は、図や表を出力するルーチンで

あり、簡単なので、理解するのは容易であろう。

```
//-------------------------------------------
//  最大塑性率・最大累積塑性率（全体、通り芯毎）
//-------------------------------------------
void CSf40View::print_soseiritu_hyo(CDC *pDC)
{
      CFont NewFont, *oldFont;
      CPen  NewPen,  *oldPen;
      CString m;
//     char buff[300];
      HWND hWnd = this->GetSafeHwnd();
    CRect rcFrame;
      ::GetClientRect(hWnd ,&rcFrame);
    int fontbase= (rectPage.right-rectPage.left)/58;
//                                        データ表示
      int iy=rectPage.top ;
      int ix=rectPage.left ;
      int F_ipy = fontbase*0.9;
      int x1= 12*fontbase;
      int icc=0;
      int iccc=0;
      int i,ic,jc;
      int xxx=1000;
      int n_k = 0;
      int nn_k = 0;
      int n_page=35;
      pDC->SetMapMode(MM_TWIPS);
      NewPen.CreatePen(PS_SOLID,20,RGB(0,0,0));
      oldFont= pr_font_style(pDC, 1);
      oldPen= pDC->SelectObject(&NewPen);
      for(ic=1;ic<=ipage;ic++){
      pDC->StartPage(); //  印刷ページ開始
// タイトル
      if(dlg_print.dlg_sosei.m_radio_jyouken == 0){
      pr_title(pDC, "最大塑性率・最大累積塑性率（全フレーム)");
      }else{
      pr_title(pDC, "最大塑性率・最大累積塑性率（通り芯)");
      }
// 条件
      pr_condition(pDC,dlg_print.dlg_sosei.m_radio_type,dlg_print.dlg_sosei.m_edit_sosei_hyo, 100);
      pr_font_style(pDC, 4);
      if(dlg_print.dlg_sosei.m_radio_jyouken == 0){
      m.Format(" 全フレーム、全部材の最大値を表示" );
      pDC->TextOut(8000-xxx, -(1780), m);
      }else{
      if(dlg_print.dlg_sosei.m_radio_tori == 0 ){
      m.Format("x 方向:            フレーム",
      dlg_print.dlg_sosei.m_edit_tori_set);
      pDC->TextOut(8000-xxx, -(1780), m);
      pDC->TextOut(9000-xxx,-(1780),_T(&ixn[(dlg_print.dlg_sosei.m_edit_tori_set-1)*5]));
      }else{
      m.Format("y 方向:          フレーム",dlg_print.dlg_sosei.m_edit_tori_set);
      pDC->TextOut(8000-xxx, -(1780), m);
```

```
        pDC->TextOut(9000-xxx,-(1780),_T(&iyn[(dlg_print.dlg_sosei.m_edit_tori_set-1)*5]));
        }
        }
        m.Format(" 部材：部材番号" );
        pDC->TextOut(8000-xxx, -(2080), m);
        m.Format(" 位置：１：i 端　２：j 端　３：部材中央" );
        pDC->TextOut(8000-xxx, -(2380), m);
    // 最大塑性率
        pr_font_style(pDC, 4);
        jc=0;
        pDC->TextOut(3300-xxx, -4560-300* jc, _T("最大塑性率"));
        jc=jc+1;
        int n_type = 0;
        pDC->TextOut(2480-xxx, -4640-300* jc, _T("   階"));
        pDC->TextOut(3160-xxx, -4640-300* jc, _T("      柱　(生起時刻)　部材 位置"));
        pDC->TextOut(5960-xxx, -4640-300* jc, _T("      はり(生起時刻)　部材 位置"));
        pDC->TextOut(8760-xxx,-4640-300*  jc, _T("  ブレース(生起時刻)　部材 位置"));
        pDC->MoveTo(2480-xxx, -4560-300* jc);pDC->LineTo(11560-xxx,-4560-300* jc);
        pDC->MoveTo(2480-xxx, -4860-300* jc);pDC->LineTo(11560-xxx, -4860-300* jc);
        pDC->MoveTo(2480-xxx, -4560-300* jc);pDC->LineTo(2480-xxx, -4860-300* jc);
        pDC->MoveTo(3160-xxx,-4560-300*jc);pDC->LineTo(3160-xxx, -4860-300* jc);
        pDC->MoveTo(5960-xxx, -4560-300* jc);pDC->LineTo(5960-xxx, -4860-300* jc);
        pDC->MoveTo(8760-xxx, -4560-300* jc);pDC->LineTo(8760-xxx, -4860-300* jc);
        pDC->MoveTo(11560-xxx,-4560-300* jc);pDC->LineTo(11560-xxx,-4860-300* jc);
        for(i=I_so;i>0;i--){
          n_k=(i-1)*18;
          nn_k=(i-1)*39;
          if(jc > n_page ) {
            pDC->EndPage();
           jc=-14;
            pDC->MoveTo(2480-xxx,-(4860+300*jc));pDC->LineTo(11560-xxx,-(4860+300* jc));
          }
          if(i == I_so){
          m.Format("      Rf");pDC->TextOut(2200-xxx, -(4920+300*jc), m);
          }else{
          m.Format("%7df",i);pDC->TextOut(2200-xxx, -(4920+300*jc), m);
          }
          if(S_hyou[n_k+3] == 0.){
          m.Format("―");pDC->TextOut(4500-xxx, -(4920+300*jc), m);
          }else{
          m.Format("%8.2f (%6.2f ) %6d %2d",S_hyou[n_k],S_hyou[n_k+3],IS_hyou[nn_k+21],IS_hyou[nn_k+24]);
          pDC->TextOut(3200-xxx, -(4920+300*jc), m);
          }
          if(S_hyou[n_k+4] == 0.){
          m.Format("―");pDC->TextOut(7300-xxx, -(4920+300*jc), m);
          }else{
          m.Format("%8.2f (%6.2f ) %6d %2d" ,S_hyou[n_k+1],S_hyou[n_k+4],IS_hyou[nn_k+22],IS_hyou[nn_k+25]);
          pDC->TextOut(6000-xxx, -(4920+300*jc), m);
          }
          if(S_hyou[n_k+5] == 0.){
          m.Format("―");pDC->TextOut(10100-xxx, -(4920+300*jc), m);
          }else{
          m.Format("%8.2f (%6.2f ) %6d %2d",S_hyou[n_k+2],S_hyou[n_k+5],IS_hyou[nn_k+23],IS_hyou[nn_k+26]);
          pDC->TextOut(8800-xxx, -(4920+300*jc), m);
```

```
        }
        pDC->MoveTo(2480-xxx,-(4860+300*(jc+1)));pDC->LineTo(11560-xxx,-(4860+300*(jc+1)));
        pDC->MoveTo(2480-xxx, -(4860+300* jc   ));pDC->LineTo(2480-xxx, -(4860+300*(jc+1)));
        pDC->MoveTo(3160-xxx, -(4860+300* jc   ));pDC->LineTo(3160-xxx, -(4860+300*(jc+1)));
        pDC->MoveTo(5960-xxx, -(4860+300* jc   ));pDC->LineTo(5960-xxx, -(4860+300*(jc+1)));
        pDC->MoveTo(8760-xxx, -(4860+300* jc   ));pDC->LineTo(8760-xxx, -(4860+300*(jc+1)));
       pDC->MoveTo(11560-xxx,-(4860+300* jc   ));pDC->LineTo(11560-xxx,-(4860+300*(jc+1)));
        jc=jc+1;
        }
// 最大累積塑性率
        if(jc +3> n_page ) {
        pDC->EndPage();
        jc=-14;
        }
       jc=jc+2;
        pDC->TextOut(3300-xxx, -4560-300* jc, _T("最大累積塑性率"));
        jc=jc+1;
        pDC->TextOut(2480-xxx, -4640-300* jc, _T("   階"));
       pDC->TextOut(3160-xxx, -4640-300* jc, _T("      柱              部材 位置"));
        pDC->TextOut(5960-xxx, -4640-300* jc, _T("      はり             部材 位置"));
        pDC->TextOut(8760-xxx,-4640-300*  jc, _T("  ブレース            部材 位置"));
        pDC->MoveTo(2480-xxx, -4560-300* jc);pDC->LineTo(11560-xxx,-4560-300* jc);
        pDC->MoveTo(2480-xxx, -4860-300* jc);pDC->LineTo(11560-xxx, -4860-300* jc);
        pDC->MoveTo(2480-xxx, -4560-300* jc);pDC->LineTo(2480-xxx, -4860-300* jc);
        pDC->MoveTo(3160-xxx, -4560-300* jc);pDC->LineTo(3160-xxx, -4860-300* jc);
        pDC->MoveTo(5960-xxx, -4560-300* jc);pDC->LineTo(5960-xxx, -4860-300* jc);
        pDC->MoveTo(8760-xxx, -4560-300* jc);pDC->LineTo(8760-xxx, -4860-300* jc);
        pDC->MoveTo(11560-xxx,-4560-300* jc);pDC->LineTo(11560-xxx,-4860-300* jc);
        for(i=I_so;i>0;i--){
        n_k=(i-1)*18+6;
        nn_k=(i-1)*39;
        if(jc > n_page ) {
        pDC->EndPage();
        jc=-14;
        pDC->MoveTo(2480-xxx, -(4860+300* jc   ));pDC->LineTo(11560-xxx,-(4860+300* jc   ));
        }
        if(i == I_so){
        m.Format("      Rf");pDC->TextOut(2200-xxx, -(4920+300*jc), m);
        }else{
        m.Format("%7df",i);pDC->TextOut(2200-xxx, -(4920+300*jc), m);
        }
        if(S_hyou[n_k-3] == 0.){
        m.Format("―");pDC->TextOut(4500-xxx, -(4920+300*jc), m);
        }else{
        m.Format("%8.2f           %6d %2d",S_hyou[n_k],IS_hyou[nn_k+27],IS_hyou[nn_k+30]);
        pDC->TextOut(3200-xxx, -(4920+300*jc), m);
        }
        if(S_hyou[n_k-2] == 0.){
        m.Format("―");pDC->TextOut(7300-xxx, -(4920+300*jc), m);
        }else{
       m.Format("%8.2f           %6d %2d" ,S_hyou[n_k+1],IS_hyou[nn_k+28],IS_hyou[nn_k+31]);
        pDC->TextOut(6000-xxx, -(4920+300*jc), m);
        }
        if(S_hyou[n_k-1] == 0.){
```

```
        m.Format("―");pDC->TextOut(10100-xxx, -(4920+300*jc), m);
        }else{
        m.Format("%8.2f          %6d %2d",S_hyou[n_k+2],IS_hyou[nn_k+29],IS_hyou[nn_k+32]);
        pDC->TextOut(8800-xxx, -(4920+300*jc), m);
        pDC->MoveTo(2480-xxx,-(4860+300*(jc+1)));pDC->LineTo(11560-xxx,-(4860+300*(jc+1)));
        pDC->MoveTo(2480-xxx, -(4860+300* jc   ));pDC->LineTo(2480-xxx, -(4860+300*(jc+1)));
        pDC->MoveTo(3160-xxx, -(4860+300* jc   ));pDC->LineTo(3160-xxx, -(4860+300*(jc+1)));
        pDC->MoveTo(5960-xxx, -(4860+300* jc   ));pDC->LineTo(5960-xxx, -(4860+300*(jc+1)));
        pDC->MoveTo(8760-xxx, -(4860+300* jc   ));pDC->LineTo(8760-xxx, -(4860+300*(jc+1)));
        pDC->MoveTo(11560-xxx,-(4860+300*jc));pDC->LineTo(11560-xxx,-(4860+300*(jc+1)));
        jc=jc+1;
    //  累積回数
        if(jc +3> n_page ) {
        pDC->EndPage();
        jc=-14;
        }
        jc=jc+2;
        pDC->TextOut(3300-xxx, -4560-300* jc, _T("累積塑性回数"));
        jc=jc+1;
        pDC->TextOut(2480-xxx, -4640-300* jc, _T("   階"));
        pDC->TextOut(3160-xxx, -4640-300* jc, _T("      柱              部材 位置"));
        pDC->TextOut(5960-xxx, -4640-300* jc, _T("      はり            部材 位置"));
        pDC->TextOut(8760-xxx,-4640-300*  jc, _T("  ブレース          部材 位置"));
        pDC->MoveTo(2480-xxx, -4560-300* jc);pDC->LineTo(11560-xxx,-4560-300* jc);
        pDC->MoveTo(2480-xxx, -4860-300* jc);pDC->LineTo(11560-xxx, -4860-300* jc);
        pDC->MoveTo(2480-xxx, -4560-300* jc);pDC->LineTo(2480-xxx, -4860-300* jc);
        pDC->MoveTo(3160-xxx, -4560-300* jc);pDC->LineTo(3160-xxx, -4860-300* jc);
        pDC->MoveTo(5960-xxx, -4560-300* jc);pDC->LineTo(5960-xxx, -4860-300* jc);
        pDC->MoveTo(8760-xxx, -4560-300* jc);pDC->LineTo(8760-xxx, -4860-300* jc);
        pDC->MoveTo(11560-xxx,-4560-300* jc);pDC->LineTo(11560-xxx,-4860-300* jc);
        for(i=I_so;i>0;i--){
          nn_k=(i-1)*39;
          n_k=(i-1)*18+6;
          if(jc > n_page ) {
          pDC->EndPage();
          jc=-14;
          pDC->MoveTo(2480-xxx, -(4860+300* jc   ));pDC->LineTo(11560-xxx,-(4860+300* jc   ));
          }
          if(i == I_so){
          m.Format("      Rf");pDC->TextOut(2200-xxx, -(4920+300*jc), m);
          }else{
          m.Format("%7df",i);pDC->TextOut(2200-xxx, -(4920+300*jc), m);
          }
          if(S_hyou[n_k-3] == 0.){
          m.Format("―");pDC->TextOut(4500-xxx, -(4920+300*jc), m);
          }else{
         m.Format("%8d           %6d %2d",IS_hyou[nn_k+12],IS_hyou[nn_k+33],IS_hyou[nn_k+36]);
          pDC->TextOut(3200-xxx, -(4920+300*jc), m);
          }
          if(S_hyou[n_k-2] == 0.){
          m.Format("―");pDC->TextOut(7300-xxx, -(4920+300*jc), m);
          }else{
         m.Format("%8d          %6d %2d" ,IS_hyou[nn_k+13],IS_hyou[nn_k+34],IS_hyou[nn_k+37]);
          pDC->TextOut(6000-xxx, -(4920+300*jc), m);
```

```
          }
          if(S_hyou[n_k-1] == 0.){
          m.Format("―");pDC->TextOut(10100-xxx, -(4920+300*jc), m);
          }else{
          m.Format("%8d           %6d %2d",IS_hyou[nn_k+14],IS_hyou[nn_k+35],IS_hyou[nn_k+38]);
          pDC->TextOut(8800-xxx, -(4920+300*jc), m);
          }
          pDC->MoveTo(2480-xxx,-(4860+300*(jc+1)));pDC->LineTo(11560-xxx,-(4860+300*(jc+1)));
          pDC->MoveTo(2480-xxx, -(4860+300* jc   ));pDC->LineTo(2480-xxx, -(4860+300*(jc+1)));
          pDC->MoveTo(3160-xxx, -(4860+300* jc   ));pDC->LineTo(3160-xxx, -(4860+300*(jc+1)));
          pDC->MoveTo(5960-xxx, -(4860+300* jc   ));pDC->LineTo(5960-xxx, -(4860+300*(jc+1)));
          pDC->MoveTo(8760-xxx, -(4860+300* jc   ));pDC->LineTo(8760-xxx, -(4860+300*(jc+1)));
          pDC->MoveTo(11560-xxx,-(4860+300*jc));pDC->LineTo(11560-xxx,-(4860+300*(jc+1)));
          jc=jc+1;
          }
          pDC->EndPage();                                  //  印刷ページ終了
          }
       pDC->EndDoc();                                  //  印刷ジョブページ終了
       pDC->SelectObject(oldFont);
       pDC->SelectObject(oldPen);
}
```

```
//-------------------------------------------
//  全部材あるいは通り芯に関する部材毎の最大塑性率・最大累積塑性率
//-------------------------------------------
void CSf40View::print_buzai_soseiritu_hyo(CDC *pDC)
{
       CFont NewFont, *oldFont;
       CPen  NewPen,  *oldPen;
       CString m;
       int i,j,ic,jc;
       int xxx=1000;
       int n_k = 0;
       int nn_k = 0;
       int n_page=35;
       int  n_out;
       pDC->SetMapMode(MM_TWIPS);
       fontbase= 50;
       NewPen.CreatePen(PS_SOLID,20,RGB(0,0,0));
       oldFont= pr_font_style(pDC, 1);
       oldPen       = pDC->SelectObject(&NewPen);
       char buff[300];
       pDC->StartPage();                //  印刷ページ開始
// タイトル
          if(dlg_print.dlg_sosei.m_radio_jyouken == 2){
          pr_title(pDC, "最大累積塑性率（全フレーム)");
          }else{
          pr_title(pDC, "最大累積塑性率（通り芯)");
          }
// 条件
          pr_condition(pDC, dlg_print.dlg_sosei.m_radio_type,
                   dlg_print.dlg_sosei.m_edit_sosei_hyo, 100);
          pr_font_style(pDC, 4);
```

```
          if(dlg_print.dlg_sosei.m_radio_jyouken == 2){
          m.Format(" 全フレーム、全部材の最大値を表示" );
             pDC->TextOut(8000-xxx, -(1780), m);
          }else{
           if(dlg_print.dlg_sosei.m_radio_tori == 0 ){
              m.Format("x 方向:             フレーム",
                   dlg_print.dlg_sosei.m_edit_tori_set);
              pDC->TextOut(8000-xxx, -(1780), m);
                   pDC->TextOut(9000-xxx,-(1780),_T(&ixn[(dlg_print.dlg_sosei.m_edit_tori_set-1)*5]));
            }else{
              m.Format("y 方向:             フレーム",
              dlg_print.dlg_sosei.m_edit_tori_set);
              pDC->TextOut(8000-xxx, -(1780), m);
              pDC->TextOut(9000-xxx, -(1780), _T(&iyn[(dlg_print.dlg_sosei.m_edit_tori_set-1)*5]));
            }
          }
             m.Format(" 部材：部材番号" );
             pDC->TextOut(8000-xxx, -(2080), m);
             m.Format(" 位置：１：i 端　２：j 端　３：部材中央" );
             pDC->TextOut(8000-xxx, -(2380), m);
             if(dlg_print.dlg_sosei.m_edit_s_sosei > 0.){
             m.Format(" 表示制限：最大塑性率　＞　%6.2f" ,dlg_print.dlg_sosei.m_edit_s_sosei);
             pDC->TextOut(8000-xxx, -(2780), m);
             }
// 最大塑性率
          jc=0;
          for(j=0;j<4;j++){
          if(jc +3> n_page ) {
          pDC->EndPage();
          jc=-14;
          }
// 出力個数チェック
          n_out=0;
          for(i=0;i<mnbsb;i++){
          nn_k=i*6;
          if(nx_member[i] != 0 && j+1 == IS_hyou[nn_k+5]) n_out++;
          }
          if(n_out != 0){
// タイトル表示
          pr_font_style(pDC, 2);
          jc=jc+1;
          if(j == 0 ) {
          pDC->TextOut(2150-xxx, -4560-300* jc, _T("(1) 柱"));
          }
          if(j == 1 ) {
          jc=jc+1;
          pDC->TextOut(2150-xxx, -4560-300* jc, _T("(2) x 方向はり"));
          }
          if(j == 2 ) {
          jc=jc+1;
          pDC->TextOut(2150-xxx, -4560-300* jc, _T("(3) ｙ方向はり"));
          }
          if(j == 3) {
          jc=jc+1;
```

```
          pDC->TextOut(2150-xxx, -4560-300* jc, _T("(4) ブレース"));
          }
           pr_font_style(pDC, 4);
           jc=jc+1;
           int n_type = 0;
           pDC->TextOut(2650-xxx, -4640-300* jc, _T(" 部材 位置 x 通り y 通り"));
           pDC->TextOut(4950-xxx, -4640-300* jc, _T("最大塑性率　(生起時刻：秒)"));
           pDC->TextOut(8150-xxx, -4640-300* jc, _T("最大累積塑性率"));
           pDC->TextOut(10300-xxx,-4640-300* jc, _T("累積塑性回数"));
           pDC->MoveTo(2480-xxx, -4560-300* jc); pDC->LineTo(11560-xxx,-4560-300* jc);
           pDC->MoveTo(2480-xxx, -4860-300* jc); pDC->LineTo(11560-xxx, -4860-300* jc);
           pDC->MoveTo(2480-xxx, -4560-300* jc); pDC->LineTo(2480-xxx, -4860-300* jc);
           pDC->MoveTo(4620-xxx, -4560-300* jc); pDC->LineTo(4620-xxx, -4860-300* jc);
           pDC->MoveTo(7520-xxx, -4560-300* jc); pDC->LineTo(7520-xxx, -4860-300* jc);
           pDC->MoveTo(10040-xxx, -4560-300* jc);         pDC->LineTo(10040-xxx, -4860-300* jc);
           pDC->MoveTo(11560-xxx,-4560-300* jc); pDC->LineTo(11560-xxx,-4860-300* jc);
           for(i=0;i<mnbsb;i++){
           if(jc > n_page ) {
            pDC->EndPage();
           jc=-14;
           pDC->MoveTo(2480-xxx, -(4860+300* jc   ));     pDC->LineTo(11560-xxx,-(4860+300* jc   ));
           }
          n_k=i*3;
          nn_k=i*6;
           if(nx_member[i] != 0 && j+1 == IS_hyou[nn_k+5]){
           m.Format("%4d",i+1);pDC->TextOut(3500-xxx-12*fontbase, -(4920+300*jc), m);
           m.Format("%2d",IS_hyou[nn_k+3]);pDC->TextOut(3500-xxx-4*fontbase, -(4920+300*jc), m);
           m.Format("%3d",IS_hyou[nn_k]);pDC->TextOut(3500-xxx+2*fontbase, -(4920+300*jc), m);
           m.Format("%3d",IS_hyou[nn_k+1]);pDC->TextOut(3500-xxx+10*fontbase, -(4920+300*jc), m);
           m.Format("%8.2f",S_hyou[n_k]);pDC->TextOut(5500-xxx, -(4920+300*jc), m);
           m.Format("(%6.2f )",S_hyou[n_k+1]);pDC->TextOut(6320-xxx, -(4920+300*jc), m);
           m.Format("%8.2f",S_hyou[n_k+2]);pDC->TextOut(8780-xxx, -(4920+300*jc), m);
           m.Format("%8d",IS_hyou[nn_k+4]);pDC->TextOut(10240-xxx, -(4920+300*jc), m);
           pDC->MoveTo(2480-xxx, -(4860+300*(jc+1)));pDC->LineTo(11560-xxx,-(4860+300*(jc+1)));
           pDC->MoveTo(2480-xxx, -(4860+300* jc   ));pDC->LineTo(2480-xxx, -(4860+300*(jc+1)));
           pDC->MoveTo(4620-xxx, -(4860+300* jc   ));pDC->LineTo(4620-xxx, -(4860+300*(jc+1)));
//         pDC->MoveTo(5120-xxx, -(4860+300* jc   ));pDC->LineTo(5120-xxx, -(4860+300*(jc+1)));
           pDC->MoveTo(7520-xxx, -(4860+300* jc   ));pDC->LineTo(7520-xxx, -(4860+300*(jc+1)));
           pDC->MoveTo(10040-xxx, -(4860+300* jc   ));pDC->LineTo(10040-xxx, -(4860+300*(jc+1)));
           pDC->MoveTo(11560-xxx,-(4860+300* jc   ));pDC->LineTo(11560-xxx,-(4860+300*(jc+1)));
          jc=jc+1;
           }}
           }}
       pDC->EndPage();
       pDC->EndDoc();                              //  印刷ジョブページ終了
       pDC->SetTextAlign(TA_LEFT);
       pDC->SelectObject(oldFont);
       pDC->SelectObject(oldPen);
}
```

```
//-------------------------------------------
//  任意部材の最大塑性率・最大累積塑性率の時刻歴図
//-------------------------------------------
```

```
void CSf40View::print_soseiritu_graph(CDC *pDC)
{
      CFont NewFont, *oldFont;
      CPen  NewPen,  *oldPen;
      CString m;
      int i,ic,jc;
      int xxx=1000;
      int ii_x;
       float ss_eps[3];
      pDC->SetMapMode(MM_TWIPS);
      NewPen.CreatePen(PS_SOLID,20,RGB(0,0,0));
      oldFont= pr_font_style(pDC, 1);
      oldPen       = pDC->SelectObject(&NewPen);
                    char buff[300];
//    float bairitu=2;
//    bairitu =  setpos(iww,iwh,rcFrame.right,rcFrame.bottom);
         pDC->StartPage();           //  印刷ページ開始
// タイトル
         pr_title(pDC, "部材の最大塑性率・最大累積塑性率の時刻歴図");
// 条件
         pr_condition(pDC, dlg_print.dlg_sosei.m_radio_type,
                  dlg_print.dlg_sosei.m_edit_sosei_hyo, 100);
         pr_font_style(pDC, 4);
         m.Format("部材番号:%8d  位置:%2d ",dlg_print.dlg_sosei.m_edit_n_member,
                  dlg_print.dlg_sosei.m_edit_nx_member);
            pDC->TextOut(8000-xxx, -(1480), m);
            m.Format(" (位置：１：i 端　２：j 端　３：部材中央)" );
            pDC->TextOut(8000-xxx, -(1780), m);
            for(int n_m=0;n_m<3;n_m++){
           ss_eps[n_m]=0.;
           if(Sec_epsy[n_m+3*(dlg_print.dlg_sosei.m_edit_n_member-1)] != 0.)
            ss_eps[n_m]=1./Sec_epsy[n_m+3*(dlg_print.dlg_sosei.m_edit_n_member-1)] ;
            }
            m.Format(" 塑性ひずみ εp: %8.6f",ss_eps[0]);
            pDC->TextOut(8000-xxx, -(2080), m);
            m.Format("           Φyp: %8.6f",ss_eps[1]);
            pDC->TextOut(8000-xxx, -(2380), m);
            m.Format("           Φzp: %8.6f",ss_eps[2]);
            pDC->TextOut(8000-xxx, -(2680), m);
//    rectPage.right=4500;
//    rectPage.left=500;
//    rectPage.top=1000;
//    rectPage.bottom=5000;
       F_pos_pr[1][1] = 4060-300* 25;
       F_pos_pr[1][0] = 4060;
       F_pos_pr[0][1] = 2480;
       F_pos_pr[0][0] = 11560;
      int ar_w = (F_pos_pr[0][0] - F_pos_pr[0][1]);
      int ar_h = (F_pos_pr[1][1] - F_pos_pr[1][0])/8.;
      fontbase= (ar_w)/50;
      char buffer[200];
      int x_axi;
      float y_axi;
      float load_data_max,ld_dt;
```

```
        float m_wave_max_v,m_wave_delt_v,ar_bi_x;
        int m_wave_time;
        int m_wave_max=F_all_step;
        m_wave_max_v=1.;
        float ar_sx;
        float ar_sy;
        jc=0;
        ar_sy = F_pos_pr[1][0] + ar_h*7.5  ;
//      図：
       for(int n_graph = 0 ;n_graph < 2; n_graph++){
      jc=jc+1;
 //図形中心設定
       ar_sx = F_pos_pr[0][1] - 5*fontbase ;
       ar_sy = ar_sy + ar_h*3. ;
       pDC->SelectObject (&NewPen_pr[0]);
       pDC->MoveTo ( ar_sx, ar_sy);
        pDC->LineTo ( ar_sx + ar_w, ar_sy);
        pDC->MoveTo ( ar_sx, ar_sy );
        pDC->LineTo ( ar_sx, ar_sy - 2*ar_h);
        pDC->SelectObject (&NewPen_pr[0]);
        pDC->MoveTo ( ar_sx,        ar_sy- 2*ar_h);
        pDC->LineTo ( ar_sx + ar_w, ar_sy- 2*ar_h);
        pDC->LineTo ( ar_sx + ar_w, ar_sy );
        pDC->LineTo ( ar_sx,        ar_sy );
        pDC->LineTo ( ar_sx,        ar_sy- 2*ar_h);

        int ist, jst;
        float ar_dtt = (float) ar_w /m_wave_max;
        float ar_bi  = (float)2*ar_h;
        int nn = 5;
// 図形軸表示
        SAIDAI_X(S_hyou,&m_wave_max,&nn,&n_graph,
             &m_wave_data_max,&load_data_max,&ld_dt);
        int iij =load_data_max/ld_dt+ 0.01;
        ar_bi = (float)ar_h*2 /load_data_max;
        for ( i=0;i <iij+1; i++)
        {
          jst = ar_sy - ld_dt*ar_bi * (i);
          y_axi= (i)*ld_dt;
          if(ld_dt >= 0.1 && ld_dt < 1.) sprintf(buffer,"%3.1f",y_axi);
          if(ld_dt >= 1 && ld_dt < 990.) sprintf(buffer,"%3.0f",y_axi);
          if(ld_dt >= 990 ) sprintf(buffer,"%5.0f",y_axi);
          if(ld_dt >= 990 ){
          pDC->TextOut(ar_sx-4*fontbase,jst+50,buffer);
          }else{
          pDC->TextOut(ar_sx-3*fontbase,jst+50,buffer);
          }
          pDC->MoveTo ( ar_sx, jst);
         pDC->LineTo ( ar_sx+ar_w, jst);
          }
// 図形 x 軸タイトル表示
         sprintf(buffer,"Time(sec.)");
         pDC->TextOut(ar_sx+ ar_w- 7.*fontbase,ar_sy-0.5*fontbase,buffer);
        if(F_all_time > 10.) {
```

```
          iij=F_all_time+0.01;
          ar_bi_x=ar_dtt/F_delt_cl;
        for ( i=0;i < iij; i++)
        {
            ii_x=((i+1)/5)*5;
            jst = ar_sx + ar_bi_x * (i+1);
            if(i+1 == ii_x ) {

            pDC->MoveTo ( jst, ar_sy+200);
            pDC->LineTo ( jst, ar_sy);
            pDC->MoveTo ( jst, ar_sy-2*ar_h-200);
            pDC->LineTo ( jst, ar_sy-2*ar_h);
            sprintf(buffer,"%2d",i+1);
            pDC->TextOut(jst-0.7*fontbase,ar_sy+200,buffer);
            }else{
           pDC->MoveTo ( jst, ar_sy+100);
           pDC->LineTo ( jst, ar_sy);
           pDC->MoveTo ( jst, ar_sy-2*ar_h-100);
           pDC->LineTo ( jst, ar_sy-2*ar_h);
        }
        }
        }else{
           iij=F_all_time*2+0.01;
           ar_bi_x=ar_dtt/F_delt_cl*0.5;
        for ( i=0;i <iij; i++)
        {
            ii_x=((i+1)/2)*2;
            jst = ar_sx + ar_bi_x * (i+1);
            if(i+1 == ii_x ) {
           pDC->MoveTo ( jst, ar_sy+200);
           pDC->LineTo ( jst, ar_sy);
           pDC->MoveTo ( jst, ar_sy-2*ar_h-200);
           pDC->LineTo ( jst, ar_sy-2*ar_h);
           sprintf(buffer,"%2d",(i+1)/2);
           pDC->TextOut(jst-0.7*fontbase,ar_sy+200,buffer);
            }else{
           pDC->MoveTo ( jst, ar_sy+100);
           pDC->LineTo ( jst, ar_sy);
           pDC->MoveTo ( jst, ar_sy-2*ar_h-100);
           pDC->LineTo ( jst, ar_sy-2*ar_h);
            }
        }}
  // 図形 y 軸タイトル表示
           if(n_graph == 0){
          sprintf(buffer,"塑性率");
           pDC->TextOut(ar_sx +ar_w*0.5- 3.*fontbase, ar_sy-0.5*fontbase,buffer);
           }
           if(n_graph == 1){
          sprintf(buffer,"累積塑性率");
           pDC->TextOut(ar_sx +ar_w*0.5- 4.*fontbase, ar_sy-0.5*fontbase,buffer);
           }
          int j;
          pDC->SelectObject (&NewPen_pr[3]);
           j=n_graph;
```

```
          int istt = ar_sx;
         int jstt = -S_hyou[j]*ar_bi+ ar_sy;
          pDC->MoveTo ( istt, jstt);
         for(  i = 1; i < m_wave_max; i++)
       {
           j=5*i+n_graph;
           ist = ar_sx + ar_dtt * (float)i;
           jst = -S_hyou[j]*ar_bi + ar_sy;
           pDC->LineTo ( ist, jst);
          }
            ar_sy = ar_sy - ar_h*0.5 ;
       }
      ar_sy = ar_sy - ar_h*0.5 ;
      for(n_graph = 2 ;n_graph < 5; n_graph++){
      jc=jc+1;
  //図形中心設定
        ar_sx = F_pos_pr[0][1] - 5*fontbase ;
        ar_sy = ar_sy + ar_h*2.5 ;
       pDC->SelectObject (&NewPen_pr[0]);
       pDC->MoveTo ( ar_sx, ar_sy);
      pDC->LineTo ( ar_sx + ar_w, ar_sy);
      pDC->MoveTo ( ar_sx, ar_sy + ar_h);
      pDC->LineTo ( ar_sx, ar_sy - ar_h);
      pDC->SelectObject (&NewPen_pr[0]);
      pDC->MoveTo ( ar_sx,        ar_sy- ar_h);
      pDC->LineTo ( ar_sx + ar_w, ar_sy- ar_h);
      pDC->LineTo ( ar_sx + ar_w, ar_sy + ar_h);
      pDC->LineTo ( ar_sx,        ar_sy + ar_h);
      pDC->LineTo ( ar_sx,        ar_sy- ar_h);
      int ist, jst;
      float ar_dtt = (float) ar_w /m_wave_max;
      float ar_bi  = (float)ar_h;
      int nn = 5;
 // 図形軸表示
      SAIDAI_X(S_hyou,&m_wave_max,&nn,&n_graph,
           &m_wave_data_max,&load_data_max,&ld_dt);
       int iij =load_data_max/ld_dt+ 0.01;
       ar_bi = (float)ar_h /load_data_max;
       jst = ar_sy ;
       y_axi=0.;
       if(ld_dt >= 0.1 && ld_dt < 1.) sprintf(buffer,"%3.1f",y_axi);
       if(ld_dt >= 1 && ld_dt < 990.)  sprintf(buffer,"%3.0f",y_axi);
       if(ld_dt >= 990 ) sprintf(buffer,"%5.0f",y_axi);
        pDC->TextOut(ar_sx-3*fontbase,jst+50,buffer);
      for ( i=0;i <iij; i++)
      {
          jst = ar_sy - ld_dt*ar_bi * (i+1);
          ist = ar_sy + ld_dt*ar_bi * (i+1);
          y_axi= (i+1)*ld_dt;
          if(ld_dt >= 0.1 && ld_dt < 1.) sprintf(buffer,"%3.1f",y_axi);
          if(ld_dt >= 1 && ld_dt < 990.)  sprintf(buffer,"%3.0f",y_axi);
          if(ld_dt >= 990 ) sprintf(buffer,"%5.0f",y_axi);
          if(ld_dt >= 990 ){
          pDC->TextOut(ar_sx-4*fontbase,jst+50,buffer);
```

```
            }else{
            pDC->TextOut(ar_sx-3*fontbase,jst+50,buffer);
            }
             y_axi= -y_axi;
            if(ld_dt >= 0.1 && ld_dt < 1.) {
            sprintf(buffer,"%3.1f",y_axi);
            pDC->TextOut(ar_sx-3.5*fontbase,ist+50,buffer);
            }
            if(ld_dt >= 1){
            if(ld_dt >= 1 && ld_dt < 990.) {
            sprintf(buffer,"%3.0f",y_axi);
            pDC->TextOut(ar_sx-3*fontbase,ist+50,buffer);
            }
            if(ld_dt >= 990 ) {
            sprintf(buffer,"%5.0f",y_axi);
            pDC->TextOut(ar_sx-4*fontbase,ist+50,buffer);
            }}
           pDC->MoveTo ( ar_sx, jst);
           pDC->LineTo ( ar_sx+300, jst);
           pDC->MoveTo ( ar_sx, ist);
           pDC->LineTo ( ar_sx+300, ist);
           pDC->MoveTo ( ar_sx+ar_w-300, jst);
           pDC->LineTo ( ar_sx+ar_w, jst);
           pDC->MoveTo ( ar_sx+ar_w-300, ist);
           pDC->LineTo ( ar_sx+ar_w, ist);
         }
  // 図形軸タイトル表示
           sprintf(buffer,"Time(sec.)");
          pDC->TextOut(ar_sx+ ar_w- 7.*fontbase,ar_sy+ar_h-0.5*fontbase,buffer);
          if(F_all_time > 10.) {
          iij=F_all_time+0.01;
          ar_bi_x=ar_dtt/F_delt_cl;
         for ( i=0;i < iij; i++)
         {
          i_x=((i+1)/5)*5;
          jst = ar_sx + ar_bi_x * (i+1);
          if(i+1 == ii_x ) {
            pDC->MoveTo ( jst, ar_sy+200);
            pDC->LineTo ( jst, ar_sy-200);
            sprintf(buffer,"%2d",i+1);
            pDC->TextOut(jst-0.7*fontbase,ar_sy+200,buffer);
            }else{
             pDC->MoveTo ( jst, ar_sy+100);
             pDC->LineTo ( jst, ar_sy-100);
              }
        }
        }else{
            iij=F_all_time*2+0.01;
            ar_bi_x=ar_dtt/F_delt_cl*0.5;
        for ( i=0;i <iij; i++)
        {
            ii_x=((i+1)/2)*2;
            jst = ar_sx + ar_bi_x * (i+1);
            if(i+1 == ii_x ) {
```

```
            pDC->MoveTo ( jst, ar_sy+200);
            pDC->LineTo ( jst, ar_sy-200);
            sprintf(buffer,"%2d",(i+1)/2);
            pDC->TextOut(jst-0.7*fontbase,ar_sy+200,buffer);
             }else{
            pDC->MoveTo ( jst, ar_sy+100);
            pDC->LineTo ( jst, ar_sy-100);
             }
          }
          }
            if(n_graph == 2){
              sprintf(buffer,"εx/εp ：軸方向ひずみ");
             pDC->TextOut(ar_sx +ar_w*0.5- 9.*fontbase, ar_sy+ar_h-0.5*fontbase,buffer);
            }
            if(n_graph == 3){
              sprintf(buffer,"Φy/Φyp：曲げひずみ");
             pDC->TextOut(ar_sx +ar_w*0.5- 8.*fontbase, ar_sy+ar_h-0.5*fontbase,buffer);
            }
                     if(n_graph == 4){
           sprintf(buffer,"Φz/Φzp：曲げひずみ");
            pDC->TextOut(ar_sx +ar_w*0.5- 8.*fontbase, ar_sy+ar_h-0.5*fontbase,buffer);
            }
        int j;
    // 図形出力
            pDC->SelectObject (&NewPen_pr[3]);
            j=n_graph;
           int istt = ar_sx;
           int jstt = -S_hyou[j]*ar_bi+ ar_sy;
            pDC->MoveTo ( istt, jstt);
           for(  i = 1; i < m_wave_max; i++)
           {
            j=5*i+n_graph;
            ist = ar_sx + ar_dtt * (float)i;
            jst = -S_hyou[j]*ar_bi + ar_sy;
            pDC->LineTo ( ist, jst);
           }
        }  //  グラフ出力終了
        pDC->EndPage();                  //  印刷ページ終了
        pDC->EndDoc();                           //  印刷ジョブページ終了
        pDC->SetTextAlign(TA_LEFT);
        pDC->SelectObject(oldFont);
        pDC->SelectObject(oldPen);
    }
```