第３章　部材モデルの履歴

3.1 はじめに

SPACE に組み込まれている部材モデルには、せん断型モデル、3 次元ケーブル弾塑性モデル、座屈を考慮したブレースモデル、Maxwell 免震モデルなど、部材の中にひとつのエレメントしか含まない単独部材モデルと、ファイバー断面、ＭＳ断面、塑性論アナロジー断面などによる複合エレメントを含む静的縮合モデルがある。本章では、単独の部材モデルの履歴特性と静的縮合モデルにおける塑性論アナロジー断面の履歴について解説する。なお、ファイバー断面の履歴モデルについては、次章で行うことにする。

3.2　3 次元ケーブル弾塑性モデル

3.2.1 組み込み位置

本節では、最も簡単な 3 次元ケーブル材の弾塑性履歴モデルについて解説する。SPACE に組み込まれた 3 次元ケーブル材の部材番号は、４であり、まず、この部材番号４に関連する element_s と member_s 構造体を以下に示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   3 次元ケーブル弾塑性モデル（モデル No.4）
C     ―――――――――――――――――――――――――――――――
c    要素数（モデル No.4  3 次元ケーブル弾塑性モデル）
c   element_s 構造体と同一
       structure / element_s4/
       integer  element_type   !  要素タイプ(6)
       integer  n_element      !  非線形要素番号
       real*8   AK_1           !   第一剛性
       real*8   AK_2           !   第二剛性
       real*8   A              !  断面積
       real*8   Q_1            !  引張第一折れ点
       real*8   Q_2            !  圧縮第一折れ点
       real*8   dm2            !  ダミー
       real*8   dm3            !  ダミー
       real*8   dm4            !  ダミー
       real*8   dm5            !  ダミー
       real*8   dm6            !  ダミー
       integer  nm_damp        !  部材減衰の有無(1)
       integer  nm_type        !  履歴モデルのタイプ
       integer  nm_section(5)  !  断面番号
       real*8   ANPt           !  引張軸方向耐力
       real*8   ANPc           !  圧縮軸方向耐力
       real*8   dmmm           !  ダミー
       real*8   dmm(3)         !  ダミー
       end structure
c      record /element_s4/ Element
```

```
c
C     ―――――――――――――――――――――――――――――――――――
C     ●   3 次元ケーブル弾塑性モデル用（モデル No.4）member_s 構造体
C     ―――――――――――――――――――――――――――――――――――
c     部材
      structure / member_s4/
      integer  nm_element          !  要素番号
      integer  element_type        !  要素タイプ
      integer  n_model             !  モデルの入れ物番号
      integer  n_model_type        !  モデル別の通し番号
      integer  n_element_type      !  要素タイプ別番号
      integer  analysis_3D         !  解析型 0:3D 1:2D(x-z) 2:2D(y-z)
      integer  nm_so               !  部材の層番号
      integer  nm_dll_element      !  DLL を用いた要素か（０；システム内要素、１：DLL 要素）
      integer  nm_point(2)         !  節点番号
      integer  irest(12)           !  部材両端の自由度番号表
      integer  istat_n             !  y 方向:履歴特性の状態(*)
      integer  idmm                !  ダミー
      integer  nm_analysis         !  部材解析種別
      integer  nm_group            !  部材グループ
      integer  nm_local_coord(2)   !  局所座標系の有無とその回転行列の番号
      integer  nm_damp             !  部材減衰の有無とその減衰行列の番号
      real*8   alength             !  長さ
      real*8   i_rigid_length      !  i 端剛域長さ
      real*8   j_rigid_length      !  j 端剛域長さ
      real*8   i_shear_G           !  i 端せん断剛性
      real*8   j_shear_G           !  j 端せん断剛性
      real*8   rot_x               !  部材主軸の回転角度（度）
      real*8   force(12)           !  部材両端の部材端力（釣合座標系）
      real*8   stress(6)           !  部材中央の応力（部材座標系）
      real*8   AKn_tan             !  接線剛性(*)
      real*8   u_past              !  部材変位
      real*8   dmn(18)             !  軸耐力:履歴特性で使用(*)
      real*8   AK_1                !  軸方向第一剛性
      real*8   AK_2                !  軸方向第二剛性
      integer   d_stat(3)          !  断面の弾塑性状態　(1) i 端 (2) j 端　(3) 中央
      real*8   an_vv(10)           !  部材軸力（計算用内部変位ｖ）
      real*8   an_ww(10)           !  部材軸力（計算用内部変位w）
      end structure
```

この部材モデルの履歴は、単純なのでデータの入出力仕様は標準の入力と出力で十分足りることになる。ここでは、このモデルとして組み込まれたサブルーチンについて説明する。まず、線形剛性行列は、サブルーチン Cal_stiff_linear()から以下のサブルーチンがコールされ、計算される。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_lin_stiff_M4
C       ―――――――――――――――――――――――――――――――――――
C     ●   Model_No.4 3 次元ケーブル弾塑性モデル
C     ―――――――――――――――――――――――――――――――――――
      subroutine Cal_lin_stiff_M4(Member,Element,ak_linear)
```

```
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s4    / Member
      record / element_s4   / Element
      dimension ak_linear(12,12)
c     ALLOCATABLE :: Member (:)
c     ALLOCATE (Member (n_member))
c           Element             structure
c           Member              structure
c           ak_linear           real*8   線形剛性行列
c-------------------------------------------------------
      do i=1,12
      do j=1,12
      ak_linear(j,i) = 0.0
      end do
      end do
      al=(Member.alength-Member.i_rigid_length
     *                                    - Member.j_rigid_length)
      ak=Element.AK_1*Element.A/al
      ak_linear(1,1)= ak
      ak_linear(1,7)=-ak
      ak_linear(7,7)= ak
      ak_linear(7,1)=-ak
c-------------------------------★履歴特性の初期設定
      Member.AKn_tan=ak
      Member.AK_1=ak
      Member.AK_2=Element.AK_2*Element.A/al
      Member.istat_n=-1
      Member.u_past=0.
      return
      end
```

線形剛性を求めるこのサブルーチンでは、単純な処理を行っている。まず、部材剛性行列をゼロクリアし、次に、軸方向剛性を次式を用いて計算し、部材剛性行列と部材の初期設定を行う。

$$k = \frac{E \cdot A}{L} \qquad \cdots\cdots(3.1)$$

次に、非線形剛性行列は、サブルーチン Get_nonlinear_stiff()より、次のサブルーチンがコールされ、計算される。

```
C     ------------------------------------------
C     ●   SUBROUTINE /Cal_nonlin_stiff_M4
C       ----------------------------------------
C     ●   Model_No.4 3 次元ケーブル弾塑性モデル
C     ------------------------------------------
      subroutine Cal_nonlin_stiff_M4(Member,Element,ak)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s4    / Member
      record / element_s4   / Element
      dimension ak(12,12)
```

```
C     ―――――――――――――――――――――――――――――――――
      do i=1,12
      do j=1,12
      ak(j,i) = 0.0
      end do
      end do
      akk=Member.AKn_tan
      ak(1,1)= akk
      ak(1,7)=-akk
      ak(7,7)= akk
      ak(7,1)=-akk
      return
      end
```

上記サブルーチンも非常に単純であり、理解は容易である。次に、応力計算を実行するサブルーチンを以下に示す。これも、簡単なコードであり、理解することは難しくない。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stress_M4
C       ―――――――――――――――――――――――――――――――――
C     ●   部材の応力計算(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_stress_M4(Member,Element,vv)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s4    / Member
      record / element_s4   / Element
      dimension vv(12)
C     ―――――――――――――――――――――――――――――――――
c           Element             structure
c           Member              structure
c           vv              real*8  増分変位
C     ―――――――――――――――――――――――――――――――――
      Member.stress(1)=Member.stress(1) + Member.AKn_tan*
     *                (vv(7)-vv(1))
      return
      end
```

部材の弾塑性状態をチェックするサブルーチン Check_stress()から以下のサブルーチンがコールされる。このサブルーチンが３次元ケーブル弾塑性モデルの履歴を管理しており、ここでは、ケーブルの特性を各種用意し、階層構造を用いて管理できるように設計されている。ただし、ここでは、履歴モデルとしては、スリップ型モデルのみが組み込まれている。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_check_stiff_M4
C       ―――――――――――――――――――――――――――――――――
C     ●   Model_No.4 3 次元ケーブル弾塑性モデル
```

```
C     ―――――――――――――――――――――――――――――――――――
      subroutine Cal_check_stiff_M4(Member,Element,vv,vpp,N_analysis)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s4    / Member
      record / element_s4   / Element
     dimension vv(12),vpp(12)
c―――――――――――――――――――――――――――――――――――――――――――
c     ALLOCATABLE :: Member (:)
c     ALLOCATE (Member (n_member))
c           Element             structure
c           Member              structure
c           ak_linear           real*8   線形剛性行列
c―――――――――――――――――――――――――――――――――――――――――――
      up= Member.u_past              ! 過去の変位
      if(N_analysis.eq.10.or.N_analysis.eq.8) then
      al=(Member.alength-Member.i_rigid_length
     *                                    - Member.j_rigid_length)
      du= vv(7)-vv(1)+( (vv(8)-vv(2))*(vpp(2)-vpp(2))+
     *       (vv(9)-vv(3))*(vpp(9)-vpp(3)) )/al
      else
      du=  vv(7)-vv(1)
      endif
      if(du.ne.0.) then
       if(N_analysis.le.8.or.Member.nm_analysis.eq.-1) then
c―――――――――――――――――――――――――――――――――★弾性解析
      Member.u_past=Member.u_past+du
      Member.stress(1)=Member.stress(1)+Member.AKn_tan*du
      else
c―――――――――――――――――――――――――――――――――★弾塑性解析
      No_rireki=Element.nm_type
      goto(5,10,20,30,40,50,60),No_rireki+1
   5  continue
c―――――――――――――――――――――――――――――――――★規定：スリップバイリニア
      call Slip_BiLinear(Member.istat_n,Member.AKn_tan,Member.dmn(1),
     *                   Member.dmn(2),Element,Member,up,du)
      Member.u_past=Member.u_past+du
      Member.stress(1)=Member.dmn(1)
c     write(76,'(a,2f15.6)') '  slip ',Member.stress(1),Member.u_past,du
      if(Member.istat_n.eq.1) then
      Member.d_stat(1)=2
      Member.d_stat(2)=2
      Member.d_stat(3)=2
      elseif(Member.istat_n.eq.2) then
      Member.d_stat(1)=1
      Member.d_stat(2)=1
      Member.d_stat(3)=1
      else
      Member.d_stat(1)=0
      Member.d_stat(2)=0
      Member.d_stat(3)=0
      endif
      goto 999
   10 continue
```

```
c―――――――――――――――――――――――――――――――――★規定：スリップバイリニア
      call Slip_BiLinear(Member.istat_n,Member.AKn_tan,Member.dmn(1),
     *                   Member.dmn(2),Element,Member,up,du)
       Member.u_past=Member.u_past+du
       Member.stress(1)=Member.dmn(1)
       if(Member.istat_n.eq.1) then
       Member.d_stat(1)=2
       Member.d_stat(2)=2
       Member.d_stat(3)=2
       elseif(Member.istat_n.eq.2) then
       Member.d_stat(1)=1
       Member.d_stat(2)=1
       Member.d_stat(3)=1
       else
       Member.d_stat(1)=0
       Member.d_stat(2)=0
       Member.d_stat(3)=0
       endif
       goto 999
       goto 999
    20 continue
c―――――――――――――――――――――――――――――――――★履歴モデル
       goto 999
    30 continue
c―――――――――――――――――――――――――――――――――★履歴モデル
       goto 999
    40 continue
c―――――――――――――――――――――――――――――――――★履歴モデル
       goto 999
    50 continue
c―――――――――――――――――――――――――――――――――★履歴モデル
       goto 999
    60 continue
c―――――――――――――――――――――――――――――――――★履歴モデル
       goto 999
   999 continue
       endif
       endif
       return
       end
```

3.2.2 スリップバイリニア型モデル

本節では、3 次元ケーブル材の弾塑性履歴モデルであるスリップバイリニア型の履歴特性について解説する。このケーブル材の弾塑性挙動は、図 3－1 に示すように、引張側の剛性は降伏点で折れ曲がるバイリニア型で表し、圧縮側は応力を維持できない、つまり、剛性がゼロとなる。ただし、プログラムでは、圧縮折れ点の軸力を与えられるようになっており、圧縮側も応力を負担可能となっている。この点を超えると剛性は

ゼロとなる。

例えば、弾性を示すistat:0 から、引張降伏軸力(Q_1)を超えると、状態：１(istat:1)となり、剛性は、K_2となる。除荷となると、再び、状態：0(istat:0)となり、剛性もK_1となる。次に、圧縮側耐力(Q_2)を超えると、状態：２(istat:2)となり、剛性はゼロとなる。

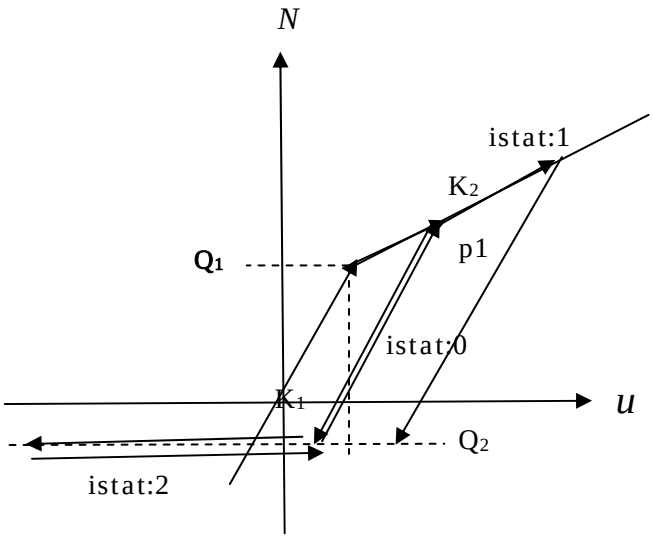

図 3-1 ケーブルのスリップバイリニア型履歴特性

上記の状態から、再度、増分変位が引張側に転じると、履歴は、そのまま、状態：２を維持し、スリップすることになる。状態：２から状態：０に移行する点は、直前で状態：０から状態：２に移った点であり、その位置を越えて増分変位が引張状態であると、履歴は弾性復帰して、状態：0 となる。

このように、ここで組み込まれたスリップバイリニア型モデルは、最も単純な履歴特性を有している。履歴特性を表すサブルーチンSlip_BiLinear()を以下に示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Slip_BiLinear
C       ―――――――――――――――――――――――――――――――
C     ●   Slip_BiLinear 履歴モデル
C     ―――――――――――――――――――――――――――――――――
      subroutine Slip_BiLinear(istat,AK,p,p1,Element,Member,up,du)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s4    / Member
      record / element_s4   / Element
     dimension vv(5)
C     ―――――――――――――――――――――――――――――――――
c     AK                :接線剛性
c     istat             :現在の状態(Work)
c     AK_1              :第一勾配
c     AK_2              :第二勾配
c     Q_1               :第一折れ点の軸力
c     Q_2               :圧縮折れ点の軸力
c     du                :増分変位
c     up                :増分前の変位
c     P                 :現在の軸力(Work)
c     P1                :istat=0 における反曲点上端(Work)
C     ―――――――――――――――――――――――――――――――――
      AK_1 =Member.AK_1           !第一勾配                          ! 1
      AK_2 =Member.AK_2           !第二勾配
      Q_1  =Element.Q_1           !第一折れ点の軸力
```

```
      Q_2  =Element.Q_2            !圧縮折れ点の軸力
c     write(76,'(a,10f17.5)') ' ak1,ak2,q1,q2',AK_1,AK_2,Q_1,Q_2
c     write(76,'(a,i4,10f17.6)') 'bracex',istat,p,ak,up,du,p1
  100 continue
      if(istat.eq.0) then                                                ! 2
      p = AK*du + p
       if(du.gt.0.) then                                                 ! 3
        if(p.lt.p1) return                                               ! 4
c-------------------------------★istat = 0 から istat=1 へ
        istat=1
        du2= (p - p1)/AK
        AK=AK_2
        p = p1 + AK*du2                !  p1:引張側の塑性応力として使用
       else                                                              ! 5
       if(p.gt.Q_2) return                                               ! 6
c-------------------------------★istat = 0 から istat=2 へ
        istat=2                                                          ! 7
        du2= (p - Q_2 )/AK
        AK=AK_1*0.000001
        p = Q_2
        p1=up - du + du2               !  p1:弾性復帰変位として使用
       endif

      elseif(istat.eq.1) then                                            ! 8
      p = AK*du + p
       if(du.ge.0.) return                                               ! 9
c-------------------------------★istat =1 から istat=0 へ
        istat=0                                                          ! 10
        px=AK*du
        p = p - px
        p1 = p
        AK=AK_1
        du=px/ak
        p = AK*du + p
       elseif(istat.eq.2) then                                           ! 11
       if(up+du.le.p1) return                                            ! 12
c-------------------------------★istat =2 から istat=0 へ
        istat=0                                                          ! 13
        du2 = up+du -p1
        px=AK*du2
        AK=AK_1
        du=p1-up+px/ak
        p1  =    Q_1+AK_2*(p1  -  Q_1/ak)             !    p1: 引 張 側 の 塑 性 応 力 と し て 使 用
      else                                                               ! 14
c-------------------------------★初期節点
       istat = 0
      AK=AK_1
      p1=Q_1
       p=0.
      goto 100
      endif
      return
      end
```

上記のサブルーチンの説明を、プログラムの右側にあるコメント番号に従って説明する。

1. この部材の第１剛性、第２剛性と引張側及び圧縮側の降伏軸力の値を構造体の成分から変数にセットする。
2. 現在の状態が０かどうかチェックする。もし０であれば、弾性状態であるとして増分応力を計算し、増分前の値に加える。
3. 増分変位が引張であるかどうかチェックし、引張の場合は、以下の処理を行い、そうでない場合は、処理５へ飛ぶ。
4. 増分後の応力が引張耐力ｐ1 を超えたかどうかチェックする。超えていない場合は、これで処理が終了となり、このサブルーチンより抜けることになる。引張耐力を超える場合は、以下の処理を行う。なお、この p1 は、初期設定では、Q1 とするが、塑性状態に入ると、最後の除荷点の軸力に設定される。ここでは、状態が０から１への移行処理を行う。まず、状態を istat=1 に変更する。次に、飛び越した部分の変位を求め、接線剛性として第２剛性をセットする。最後に、先に求めた飛び越した部分の変位と第２剛性とで、跳び越し部分の軸力を再評価して求め、p1 に足しこむ。
5. 以下の処理は、状態：0 で、増分変位が圧縮の場合である。
6. 増分後の軸力が圧縮耐力を超えているかどうかチェックする。超えていない場合は、処理を終了し、このサブルーチンから抜ける。
7. 状態：0 から状態：2 に、istat=2 として変更する。次に、跳び越した部分の変位を求める。剛性をゼロとするわけであるが、処理の都合上、この接線剛性を第１剛性の 0.000001 倍とする。増分後の軸力を圧縮側の耐力とする。最後に、状態：2 から状態：0 に復帰する場合の変位点を計算し、ワーク領域であるｐ1 にセットする。
8. 以後の処理は、状態：1 の場合である。まず、増分後の軸力を計算する。
9. 増分変位が引張である場合、処理は終了し、このサブルーチンより抜ける。
10. 増分変位が圧縮の場合、以下の処理を行う。ここでは、除荷であるため、状態：2 を状態：0、istat=0 に変更する。まず、増分軸力を求め、増分前の軸力を計算する。この増分前の軸力をワーク領域の p1 にセットする。次に、接線剛性を第１剛性に設定し、その接線剛性を用いて増分変位を計算する。さらに、その増分変位と接線剛性より増分軸力を再計算し、その値を増分前の軸力に足しこむ。

11. 状態が 2 であるかどうかチェックし、2 である場合は以下の処理を行う。
12. 増分後の変位が、最後に状態：２に入ったときの変位を飛び越しているかどうかチェックする。この時の変位 p1 は、7)で設定されている。
13. ここでは、状態：2 から状態：0 に移行する場合の処理を行う。まず、istat=0 に変更し、状態を変える。次に、跳び越した部分の変位を求め、この部分の軸力を求める。接線剛性を第 1 剛性にセットし直し、跳び越し部分の変位を求める。これらの値から、増分変位を計算し直す。さらに、状態：0 と状態：1 の交点 p1 を求め直す。
14. この部材モデルの初期設定を行う。状態を 0 に設定し、接線剛性を第 1 剛性に、ワーク領域のｐ1 を引張側の耐力 Q_1 に、最後に、軸力 p をゼロセットする。

3.3 3 次元軸力弾塑性モデル（座屈を考慮したトラスモデル）

3.3.1 組み込み法

本節では､座屈を考慮した3次元軸力弾塑性モデルについて解説する。ここで使用する構造体 element_s3 と member_s3 は、標準の要素に関する構造体と部材に関する構造体を書き直したものである。これらの構造体を以下に示す。このモデルでは、これ以外に特別に必要とする構造体は存在しない。

```
C     ―――――――――――――――――――――――――――――――――
C     ●　3 次元軸力弾塑性モデル（モデル No.3）
C     ―――――――――――――――――――――――――――――――――
c    要素数（モデル No.3　3 次元軸力弾塑性モデル：座屈耐力降下型）
c   element_s 構造体と同一
c
       structure / element_s3/
       integer  element_type   !  要素タイプ(6)
       integer  n_element      !  非線形要素番号
       real*8   E_1            !  第一弾性係数
       real*8   E_2            !  第二弾性係数
       real*8   A              !  断面積
       real*8   Iy             !  弱軸周り断面二次モーメント
       real*8   sigma          !  降伏応力度
       real*8   Qt_1           !  引張第一折れ点の軸力
       real*8   Qc_1           !  圧縮第一折れ点の軸力
       real*8   AK_1           !  第一剛性
       real*8   AK_2           !  第二剛性
       real*8   ramda          !  細長比
       integer  nm_damp        !  部材減衰の有無(1)
       integer  nm_type        !  履歴モデルのタイプ
```

```
        integer  nm_section(5)  !  断面番号
        real*8   ANPt           !  引張軸方向耐力
        real*8   ANPc           !  圧縮軸方向耐力
        real*8   Genkai         !  短柱＝１０　長柱＝０
        real*8   dmm(3)         !  ダミー
        end structure
c       record /element_s3/ Element
c
C     ―――――――――――――――――――――――――――――――――――
C     ●　３次元軸力弾塑性モデル用（モデル No.3_1）member_s 構造体
C     ―――――――――――――――――――――――――――――――――――
c
c     部材
      structure / member_s3x/
      integer  nm_element         !  要素番号
      integer  element_type       !  要素タイプ
      integer  n_model            !  モデルの入れ物番号
      integer  n_model_type       !  モデル別の通し番号
      integer  n_element_type     !  要素タイプ別番号
      integer  analysis_3D        !  解析型 0:3D 1:2D(x-z) 2:2D(y-z)
      integer  nm_so              !  部材の層番号
      integer  nm_dll_element     !  DLL を用いた要素か（０；システム内要素、１：DLL 要素）
      integer  nm_point(2)        !  節点番号
      integer  irest(12)          !  部材両端の自由度番号表
      integer  istat_n            !  y 方向:履歴特性の状態(*)
      integer  idmm               !  ダミー
      integer  nm_analysis        !  部材解析種別
      integer  nm_group           !  部材グループ
      integer  nm_local_coord(2)  !  局所座標系の有無とその回転行列の番号
      integer  nm_damp            !  部材減衰の有無とその減衰行列の番号
      real*8   alength            !  長さ
      real*8   i_rigid_length     !  i 端剛域長さ
      real*8   j_rigid_length     !  j 端剛域長さ
      real*8   i_shear_G          !  i 端せん断剛性
      real*8   j_shear_G          !  j 端せん断剛性
      real*8   rot_x              !  部材主軸の回転角度（度）
      real*8   force(12)          !  部材両端の部材端力（釣合座標系）
      real*8   stress(6)          !  部材中央の応力（部材座標系）
      real*8   AKn_tan            !  接線剛性(*)
      real*8   ramda              !  細長比
      real*8   AK_0               !  第一剛性
      real*8   AK_5               !  第二剛性
      real*8   Nt_0               !  引張り耐力応力
      real*8   u_0                !  引張り耐力変位
      real*8   n_E                !  オイラー荷重と降伏軸力の比
      real*8   K_1                !  第一剛性(無次元)
      real*8   K_2                !  第二剛性(無次元)
      real*8   u_past             !  全変位成分
      real*8   N_t                !  引張り耐力応力(無次元)
      real*8   N_c                !  圧縮耐力応力(無次元)
      real*8   d_A                !  A 基準点変位
      real*8   d_B                !  B 基準点変位
      real*8   d_C                !  C 基準点変位
      real*8   d_D                !  D 基準点変位
```

```
      real*8   d_P              !  P基準点変位
      real*8   d_Q              !  Q基準点変位
      real*8   N_A              !  A基準点軸力
      real*8   N_B              !  B基準点軸力
      real*8   N_P              !  P基準点軸力
      real*8   N_Q              !  Q基準点軸力
      integer   d_stat(3)       !  断面の弾塑性状態  (1) i端 (2) j端  (3) 中央
      real*8   dmmm             !  ワーク領域
      real*8   P(3)             !  ワーク領域
      real*8   Q(3)             !  ワーク領域
      real*8   nc               !  ワークエリア
      real*8   Genkai           !  ダミー
      real*8   alf              !  ワークエリア
      end structure
```

このモデルにおける入力と出力仕様は、標準の入力仕様の並びをこのモデル用に変更したものを使用しており、特別に入出力のコードを設計する必要はない。まず、このモデルの初期設定は、線形剛性行列を求めるサブルーチン Cal_stiff_linear()で、以下のサブルーチンがコールされる。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_lin_stiff_M3
C     ―――――――――――――――――――――――――――――――
C     ●   Model_No.3 3次元軸力弾塑性モデル
C     ―――――――――――――――――――――――――――――――
      subroutine Cal_lin_stiff_M3(Member,Element,ak_linear)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s3    / Member
      record / element_s3   / Element
      dimension ak_linear(12,12)
C     ―――――――――――――――――――――――――――――――
c     ALLOCATABLE :: Member (:)
c     ALLOCATE (Member (n_member))
c           Element             structure
c           Member              structure
c           ak_linear           real*8   線形剛性行列
c―――――――――――――――――――――――――――――――――――――――――
      al = Member.alength - Member.i_rigid_length
     *                                    - Member.j_rigid_length
      do i=1,12
      do j=1,12
      ak_linear(j,i) = 0.0
      end do
      end do
      ak=Element.E_1*Element.A/al
      ak_linear(1,1)= ak
      ak_linear(1,7)=-ak
      ak_linear(7,7)= ak
      ak_linear(7,1)=-ak
```

```
c――――――――――――――――――――――――――――――――★履歴特性の初期設定
      Member.AKn_tan=ak
      iet  = Element.nm_type+1
      goto(11,11,12,13,14,15,16),iet
  11  continue
c――――――――――――――――――――――――――――――――★履歴特性の初期設定
      call Set_init_Brace_1(1,Member,Element,ak,al)
      return
  12  continue
c――――――――――――――――――――――――――――――――★履歴特性の初期設定
      call Set_init_Brace_1(2,Member,Element,ak,al)
      return
c――――――――――――――――――――――――――――――――★履歴特性の初期設定
  13  continue
c――――――――――――――――――――――――――――――――★履歴特性の初期設定
      return
  14  continue
c――――――――――――――――――――――――――――――――★履歴特性の初期設定
      return
  15  continue
c――――――――――――――――――――――――――――――――★履歴特性の初期設定
      return
  16  continue
c――――――――――――――――――――――――――――――――★履歴特性の初期設定
      call Set_init_Brace5(Member,Element,ak,al)
      return
      end
```

上記のプログラムでは、線形の剛性行列として、軸方向剛性の部分にセットされている。さらに、このモデルの履歴特性として階層構造が見られ、各々の履歴番号に対応した初期設定サブルーチンがコールされている。これらのサブルーチンの内容は、次節で述べる。

次に、非線形剛性行列は、サブルーチン Get_nonlinear_stiff() より、次のサブルーチンがコールされ、計算される。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_nonlin_stiff_M3
C       ――――――――――――――――――――――――――――――――――――
C     ●   Model_No.3 3 次元軸力弾塑性モデル
C     ――――――――――――――――――――――――――――――――――――
      subroutine Cal_nonlin_stiff_M3(Member,Element,ak)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s3    / Member
      record / element_s3   / Element
      dimension ak(12,12)
C     ――――――――――――――――――――――――――――――――――――
      do i=1,12
      do j=1,12
      ak(j,i) = 0.0
      end do
```

```
      end do
      akk=Member.AKn_tan
      ak(1,1)= akk
      ak(1,7)=-akk
      ak(7,7)= akk
      ak(7,1)=-akk
      return
      end
```

部材の弾塑性状態をチェックするサブルーチン Check_stress()から以下のサブルーチンがコールされる。このサブルーチンが座屈を考慮した３次元トラスモデルの履歴を管理している。ここでは、トラスモデルの特性を各種用意し、階層構造を用いて管理できるように設計されている。ただし、履歴モデルの内容は次節で説明する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_check_stiff_M3
C     ―――――――――――――――――――――――――――――――――
C     ●   Model_No.3 3 次元軸力弾塑性モデル
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_check_stiff_M3(Member,Element,vv,vpp,
     *                              N_analysis)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s3    / Member
      record / element_s3   / Element
      dimension vv(12),vpp(12)
      real*8 st_x
C     ―――――――――――――――――――――――――――――――――
c     ALLOCATABLE :: Member (:)
c     ALLOCATE (Member (n_member))
c           Element             structure
c           Member              structure
c           ak_linear           real*8   線形剛性行列
c―――――――――――――――――――――――――――――――――――――――――
      du=vv(7)-vv(1)
      u =Member.u_past
      if(N_analysis.eq.8.or.N_analysis.eq.10) then
c      if()   ! 幾何学的非線形を考慮する場合
      al=(Member.alength-Member.i_rigid_length
     *                   - Member.j_rigid_length)
      du=du+( (vv(8)-vv(2))*(vpp(8)-vpp(2))+
     *        (vv(9)-vv(3))*(vpp(9)-vpp(3)) )/al
      endif
      st_x = Member.stress(1)
      if(du.ne.0.) then
       if(N_analysis.le.8.or.Member.nm_analysis.eq.-1) then
c―――――――――――――――――――――――――――――――★弾性解析
c       Member.stress(1)=Member.stress(1)+Member.AKn_tan*du
      else
      No_rireki = Element.nm_type
```

```
      goto(10,10,10,30,40,50,60),No_rireki+1
   10 continue
c――――――――――――――――――――――――――――――――★バイリニア：座屈考慮モデル
c――――――――――――――――★無次元量に変換
      ak = Member.AKn_tan/Member.AK_1
      call Bilinear_Bx(Member.istat_n,ak,Member,Element,du)
      Member.AKn_tan=ak*Member.AK_1
      Member.d_stat(3)=0
      if(Member.istat_n.eq.2.or.Member.istat_n.eq.3) then
      Member.d_stat(3)=2
      elseif(Member.istat_n.eq.5) then
      Member.d_stat(3)=1
      endif
      goto 999

   20 continue
c――――――――――――――――――――――――――――――――★履歴モデル
      goto 999
   30 continue
c――――――――――――――――――――――――――――――――★履歴モデル

      goto 999
   40 continue
c――――――――――――――――――――――――――――――――★履歴モデル

      goto 999
   50 continue
c――――――――――――――――――――――――――――――――★履歴モデル
c――――――――――――――――――――――――――――――――★バイリニア
      call Bilinear(Member.AKn_tan,Member.d_stat(3),
     *                    Element.AK_1,Element.AK_2,
     *                    Element.Qt_1,du,st_x,
     *                    Member.P(1))
      goto 999
   60 continue
c――――――――――――――――――――――――――――――――★履歴モデル
c――――――――――――――――――――――――――――――――★バイリニア：座屈考慮モデル
      call Bilinear_B(Member.AKn_tan,Member.istat_n,
     *                    Member.AK_1,Member.AK_2,
     *                    Element.Qc_1,Element.Qt_1,du,u,
     *                    st_x,Member.u_plastic,
     *                    Member.u_elastic,Member.ramda,
     *                    Member.dw,Member.delta,
     *                    Member.P(1),Member.P(2),Member.Genkai)

      Member.d_stat(3)=0
      if(Member.istat_n.eq.2.or.Member.istat_n.eq.3) then
      Member.d_stat(3)=2
      elseif(Member.istat_n.eq.5) then
      Member.d_stat(3)=1
      endif
      goto 999
  999 continue
      endif
```

```
        endif
        Member.u_past = Member.u_past + du
        return
        end
```

3.3.2 座屈を考慮したトラスモデル

本節では、座屈を考慮したトラスモデルの中で、文献 30,31 にしたがって作られた履歴特性について解説する。理論の詳細は文献を参照され、また、その使用上の制限条件も良く読んで理解されたい。

個材の圧縮耐力は、普通、材の細長比λをパラメータとして表され、図 3-2a のようになる。ここで、λが十分小さいか、十分大きいか、あるいはそのどちらでもない中間の大きさかによって、それぞれ降伏、弾性座屈、弾塑性座屈で材の載荷能力は決まってくる。また、個材の圧縮力と軸方向変位の関係は材の細長比によって異なり、一般に図 3-2b のような形となる。材の細長比が小さいか、あるいは大きい場合、最大耐力を超えて変位が進んでも、耐力の低下は緩やかである。しかし、材の細長比が限界細長比付近の場合、最大耐力に到達した後、耐力は急激に低減する。

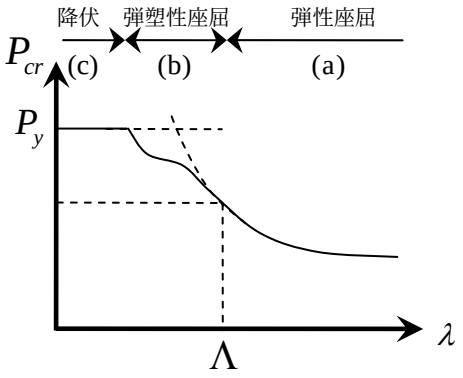

図 3-2a　部材の圧縮耐力と細長比の関係

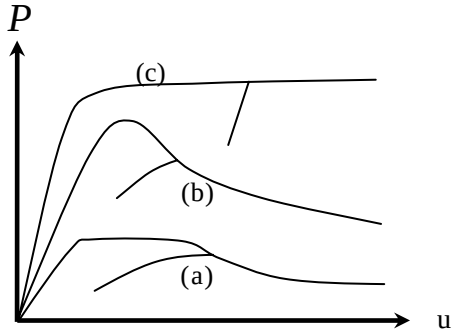

図 3-2b　部材の圧縮力と軸方向変位の関係

個材の座屈後、および降伏後も含めた履歴挙動に関する理論的研究では、塑性ヒンジの概念を導入したものと、有限要素法による直接解析によるものがある。また、実験的研究も数多く行われており、理論の検証と共に、個材の非弾性座屈挙動そのものが明らかにされつつある。軸方向力と軸方向変位の関係をモデル化し、部材の復元力特性を近似する解法はさまざまな形で提案されている。その中で最も単純なモデルは、直線型モデルで全ての領域を線形近似する。しかし、座屈後の挙動は直線で近似すると誤差が大きくなり、精密な非弾性領域の追跡ができない。コンピュータの能力が向上した今日では、弾性領域は直線、非弾性領域は曲線で近似することが多い。特に、双曲線関数や高次関数を用いる場合が多い。この方法では、処女載荷時の挙動は良い精度で近似できるが、繰返し載荷を受ける場合その誤差が大きくなっていくという欠点を有する。この問題は、除荷が起こった際の弾性領域で細長比が大きいものほど幾何学的非線形性の影響を強く受け、非線形弾性の履歴を描くためであると思われる。また、バウジンガー効果など材料の特性を無視したモデル化を行うとこの傾向が強くなる。

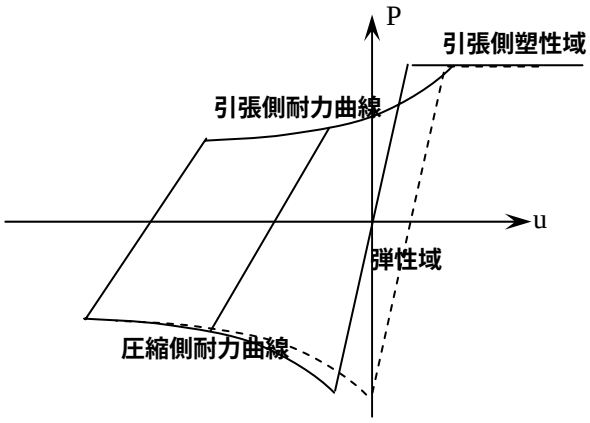

図 3-3　軸方向履歴モデル

ここでは、文献 30,31 にしたがって、SPACE に組み込まれた部材の復元力特性を、以下に示す。

繰返し軸方向力を受ける単一筋違の復元力特性を、図 3-3 のような一組の耐力曲線と弾性除荷直線および全断面引張塑性域直線で表現する。ここでは、この区分した領域ごとに説明しよう

(1) 圧縮側耐力曲線

耐力曲線は、引張側、圧縮側ともに、次式で与えられるものとする。

$$n = \frac{1}{(a \cdot \delta + b)^r} \qquad \cdots\cdots(3.2)$$

ここに、$n \equiv N / N_0$ は筋違軸力の大きさと降伏軸力の比、$\delta \equiv u / u_0$ は軸力変位と降伏軸方向変位の比であり、a および b は無次元 Euler 荷重 n_E の関数である。

$$n_E = \frac{\pi^2 E}{\lambda^2 \sigma_y} \qquad \cdots\cdots(3.3)$$

また、定数 r は $n^{-1/r}$ と δ がほぼ線形の関係になるように値を選択する。

実験で得られた代表的な圧縮側耐力曲線から、$r=1/2$ と定め、圧縮側耐力曲線を次式で与える。

$$n=\frac{1}{\sqrt{p_1\delta+p_2}} \qquad \cdots\cdots(3.4)$$

ここで、

$$p_1=\frac{10/n_E-1}{3}$$

$$p_2=\frac{4}{n_E}+0.6$$

とする。ただし、式が適用できるのは $n_E \le 10$ の場合に限られる。接線剛性は、式(3.4)から次式で与えられる。

$$ak=\frac{dn}{d\delta}=\frac{-0.5p_1}{(p_1\delta+p_2)^{\frac{3}{2}}} \qquad \cdots\cdots(3.5)$$

(2) 引張側耐力曲線

引張側耐力曲線は(0, 1)点を通る曲線として定義されるので、式(3.2)における b は、b=1 となる。実験で得られた結果によると、$(n^{-2/3}-1)$ と δ の関係は原点を通る直線で表されるため、$r=3/2$ とし、引張側耐力曲線を次式のように与える。

$$n=\frac{1}{(p_3\delta+1)^{\frac{3}{2}}} \qquad \cdots\cdots(3.6)$$

ただし、

$$p_3=\frac{1}{3.1\cdot n_E+1.4}$$

とする。引張側耐力曲線の接線剛性は、式(3.6)より、次式となる。

$$ak=\frac{dn}{d\delta}=-\frac{1.5p_3}{(p_3\delta+1)^{\frac{5}{2}}} \qquad \cdots\cdots(3.7)$$

(3) 引張側耐力曲線の移動則

引張側耐力曲線の途中から変位を逆転し、δ_a だけ圧縮側に変位させた後、再び変位を逆転させた場合の引張側耐力曲線の移動量 x を次のよ

うに定める。

$$x = \ln(q_1\bar{\delta}_a + 1) - q_2 s \geq 0 \qquad \cdots\cdots(3.8)$$

ただし、

$$q_1 = \frac{3 - 1/n_E}{10}$$

$$q_2 = \frac{0.115}{n_E} + 0.36$$

とする。

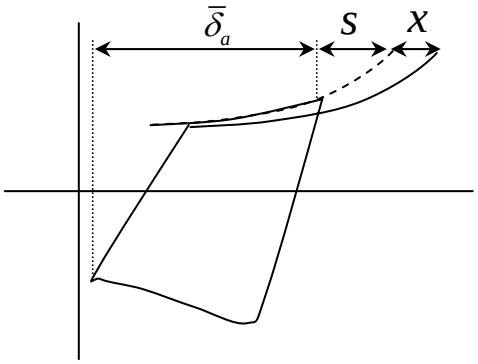

図 3-4　引張側耐力曲線の移動

(4) 圧縮側耐力曲線の移動則

圧縮側非弾性領域にある応力点において変位を逆転し、引張側耐力曲線と圧縮側耐力曲線の基準点の移動量 y は引張側塑性変形量 δ_b に比例するものとして扱う。

$$y / y_0 = \delta_b / \delta_{b0} \qquad \cdots\cdots(3.9)$$

(5) 初期剛性と座屈

処女載荷時の座屈軸力は、設計指針の基準式を用いるものとする。

$\lambda > \Lambda$ のとき

$$P_{cr} = \frac{9}{13}\frac{\pi^2}{\lambda^2} \qquad \cdots\cdots(3.10)$$

$\lambda \leq \Lambda$ のとき

$$P_{cr} = \frac{1 - 0.4\left(\frac{\lambda}{\Lambda}\right)^2}{1 + \frac{4}{9}\left(\frac{\lambda}{\Lambda}\right)^2} \qquad \cdots\cdots(3.11)$$

以上が、文献に従った履歴曲線の大まかな説明である。以後、具体的に、プログラムコードに従って、その内容を説明する。

最初に、この履歴を追跡するために、各種の初期設定が必要となる。このサブルーチンは、前節で示した線形剛性を計算するサブルーチン Cal_lin_stiff_M3()でコールされる。以下に、初期設定サブルーチン Set_init_Brace_1()を示す。その内容は、コードの右に示したコメントで理解できるものと思われる。また、履歴を追うための代表点の計算は、図 3-5b を参照しながら理解されたい。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Set_init_Brace_1
C     ―――――――――――――――――――――――――――――――――――
C     ●   Model_No.3_1 3 次元軸力弾塑性モデル
C     ―――――――――――――――――――――――――――――――――――
      subroutine Set_init_Brace_1(ibuckl,Member,Element,ak,al)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s3x   / Member
      record / element_s3   / Element
      data Pi/3.1415926/
C     ―――――――――――――――――――――――――――――――――――
c     real*8   ramda              !  細長比
c     real*8   AK_0               !  第一剛性
c     real*8   AK_5               !  第二剛性
c     real*8   Nt_0               !  引張り耐力応力
c     real*8   u_0                !  引張り耐力変位
c     real*8   n_E                !  オイラー荷重と降伏軸力の比
c     real*8   K_1                !  第一剛性(無次元)
c     real*8   K_2                !  第二剛性(無次元)
c     real*8   u_past             !  全変位成分
c      real*8   N_t               !  引張り耐力応力(無次元)
c      real*8   N_c               !  圧縮耐力応力(無次元)
c      real*8   d_A               !  A 基準点変位
c      real*8   d_B               !  B 基準点変位
c      real*8   d_C               !  C 基準点変位
c      real*8   d_D               !  D 基準点変位
c      real*8   d_P               !  P 基準点変位
c      real*8   d_Q               !  Q 基準点変位
c      real*8   N_A               !  A 基準点軸力
c      real*8   N_B               !  B 基準点軸力
c      real*8   N_P               !  P 基準点軸力
c      real*8   N_Q               !  Q 基準点軸力
c      integer  d_stat(3)         !  断面の弾塑性状態　(1) i 端 (2) j 端　(3) 中央
c      real*8   P(3)              !  ワーク領域
c      real*8   Q(3)              !  ワーク領域
c      real*8   nc                !  ワークエリア
c      real*8   Genkai            !  ダミー
c               ibuckl   :1:座屈耐力は学会規準　2：論文に準拠
C     ―――――――――――――――――――――――――――――――――――
         Member.AK_0 = ak                                    ! 剛性の初期設定
         Member.AK_5 = Element.E_2*Element.A/al
                                                             ! 状態変数の初期化
          Member.d_stat(3)=0
          Member.istat_n  =0
C     ―――――――――――――――――――
          Member.ramda = al*sqrt(Element.A/Element.Iy)       ! 細長比の計算
          Member.Nt_0=Element.sigma*Element.A                !  引張り耐力応力
          Member.u_0 =Element.sigma*al/Element.E_1           !  引張り耐力変位
          Member.n_E =Pi*Pi*Element.E_1/
     *         (Member.ramda*Member.ramda*Element.sigma)     !  オイラー荷重と降伏軸力の比
          Member.K_1= 1.                                     ! 第一剛性(無次元)
```

```
          Member.K_2= Member.AK_5/ak                          ! 第二剛性(無次元)
          if(Member.K_2.eq.0.) Member.K_2=0.00001
          Member.u_past = 0.                                  !  全変位成分
          Member.N_t    = 1.                                  ! 引張り耐力応力(無次元)
         Buckl_limit = Pi * SQRT(Element.E_1/(0.6*Element.sigma))   ! 限界細長比の計算
         if(Member.ramda.lt. Buckl_limit) then                ! 弾塑性座屈
             ram_a=(Member.ramda/Buckl_limit)**2
            Member.N_c = -(1.-0.4*ram_a)/(1.+4.*ram_a/9.)
            Member.Genkai = 10.
         else                                                 ! 弾性座屈
            Member.N_c = -(Pi*Pi*Element.E_1)/
     *                    (1.44*Member.ramda*Member.ramda)
            Member.Genkai = 0.
         endif
              AnE =  Member.n_E
              Member.P(1)=(10./AnE-1.)/3.                          !  ワーク領域
              Member.P(2)=4./AnE +0.6                              !  ワーク領域
              Member.P(3)=1/(3.1*AnE+1.4)                          !  ワーク領域
              Member.Q(1)=(3.-1./AnE)/10.                          !  ワーク領域
              Member.Q(2)=1.15/AnE+3.6                             !  ワーク領域
              Member.Q(3)=0.3*dsqrt(AnE)+0.24                      !  ワーク領域
             call set_nc( Member.nc,Member.P(1),Member.P(2))       !  ワークエリア
              if( ibuckl.eq.2) Member.N_c = -Member.nc
c―――――――――――――――――――――――★最大点の調整
              Member.alf=dabs(Member.N_c/Member.nc)
              Member.d_A= 1.               !  A基準点変位
              Member.d_B= Member.N_c       !  B基準点変位
              Member.d_C= 1.               !  C基準点変位
              Member.d_D= Member.N_c       !  D基準点変位
              Member.d_P= 1.               !  P基準点変位
              Member.d_Q= Member.N_c       !  Q基準点変位
              Member.N_A= 1.               !  A基準点軸力
              Member.N_B= Member.N_c       !  B基準点軸力
              Member.N_P= 1.               !  P基準点軸力
              Member.N_Q= Member.N_c       !  Q基準点軸力
c      write(76,'(a)') '  member tokusei '
c     write(76,'(a,f12.3)') 'Member.ramda ' ,Member.ramda
c     write(76,'(a,f12.3)') 'Member.Nt_0  ' ,Member.Nt_0
c     write(76,'(a,f12.3)') 'Member.u_0   ' ,Member.u_0
c     write(76,'(a,f12.3)') 'Member.n_E   ' ,Member.n_E
c     write(76,'(a,f12.3)') 'Member.K_1   ' ,Member.K_1
c     write(76,'(a,f12.6)') 'Member.K_2   ' ,Member.K_2
c     write(76,'(a,f12.3)') 'Buckl_limit  ' ,Buckl_limit
c     write(76,'(a,f12.3)') 'Member.N_c   ' ,Member.N_c
c     write(76,'(a,f12.3)') 'Member.p(1)  ' ,Member.p(1)
c     write(76,'(a,f12.3)') 'Member.p(2)  ' ,Member.p(2)
c     write(76,'(a,f12.3)') 'Member.p(3)  ' ,Member.p(3)
c     write(76,'(a,f12.3)') 'Member.q(1)  ' ,Member.q(1)
c     write(76,'(a,f12.3)') 'Member.q(2)  ' ,Member.q(2)
c     write(76,'(a,f12.3)') 'Member.q(3)  ' ,Member.q(3)
c     write(76,'(a,f12.3)') 'Member.nc    ' ,Member.nc
c     write(76,'(a,f12.3)') 'Member.alf   ' ,Member.alf
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Set_nc
C     ―――――――――――――――――――――――――――――――――
C     ●
C     ―――――――――――――――――――――――――――――――――
      subroutine Set_nc(anc,P1,P2)
      implicit real*8(A-H,O-Z)
      anc=0.
      ancc=0.1
      err=1.0d-6
      rate=0.5
      f0 = -1.
 1000 continue
      anc=anc + ancc
      f1=(P1*anc+P2)*anc*anc - 1.
      if(dabs(f1). lt. err) goto 2000
      if(f0*f1 .lt. 0.0) then
        ancc = -ancc*rate
      endif
      f0=f1
      goto 1000
 2000 continue
      return
      end
```

次に、履歴を追跡するサブルーチン BILINEAR_Bx()について説明する。この履歴は、軸力を受けてトラスもしくはブレースが座屈する挙動を表す。その挙動は、部材の細長比および断面形状によって異なるが、ここでは、細長比によってのみ変化するように設定されている。図 3-5 は、細長比λ=120 の弾塑性座屈を生じるトラス材の履歴を表す。

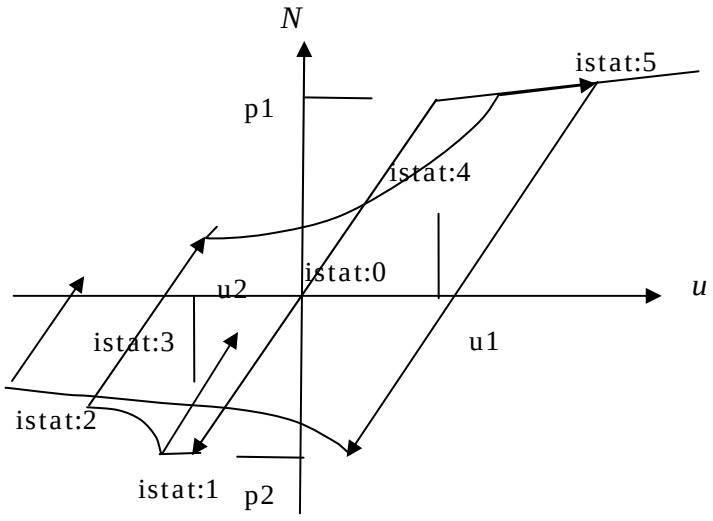

図 3-5a 座屈を考慮したトラスモデルの履歴特性

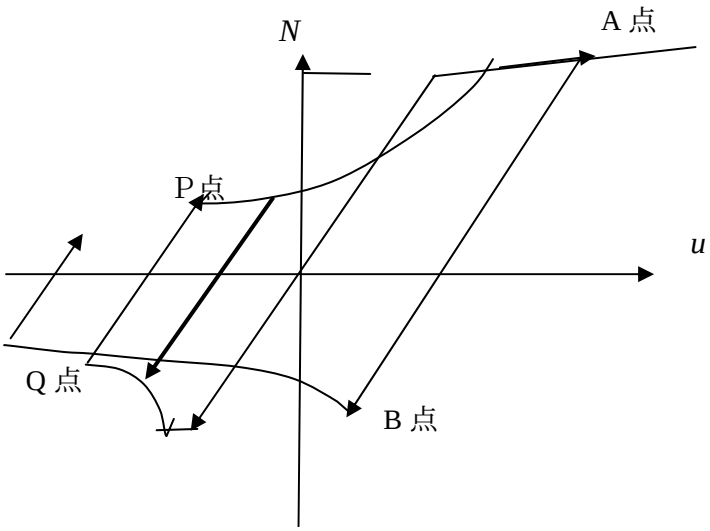

図 3-5b 代表点の位置

図3-5bの中で示される点は、この履歴を追跡するための代表点であり、文献によると、A 点は引張側基準点、B 点を圧縮側基準点という。

以下にサブルーチン BILINEAR_Bx()を示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE   /  BILINEAR_Bx
C     ―――――――――――――――――――――――――――――――――
C     ●   軸方向力履歴特性モデル（柴田、中村、若林の理論）
C     ―――――――――――――――――――――――――――――――――
      subroutine BILINEAR_Bx(istat,ak,Member,Element,du)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s3x    / Member
      record / element_s3    / Element
c――――――――――――――――――――――――――――――――★ 変位増分がなかったら戻る
      if(du.eq.0) return                                              ! 1
c――――――――――――――――――――――――――――――――★ 値を無次元量に変換
      delt = ( Member.u_past + du ) / Member.u_0                       ! 2
      d_delt = du/ Member.u_0
      Stress_x = Member.stress(1)/Member.Nt_0
C     ―――――――――――――――――――――――――――――――――
      selectcase (istat)                                              ! 3
c――――――――――――――――――――――――――――――――★ state = 0    弾性範囲
      case (0)                                                        ! 4
        if(d_delt.gt.0.0)then                                         ! 5
          if(Stress_x.lt.Member.N_t) return                           ! 6
c――――――――――――――――――――――――――――――――★ 引張り塑性
c――――――――――――――――――――――――――――――――★増分変位調節
          dp = ak*d_delt                                              ! 7
          d_over=d_delt*( Stress_x - Member.N_t)/dp
```

```
c――――――――――――――――――――――★増分変位調節
          istat=5                                                    ! 8
          ak= Member.K_2
          Stress_x = Member.N_t +ak*d_over
          Member.stress(1) = Stress_x * Member.Nt_0
      else                                                           ! 9
          if(Stress_x.gt.Member.N_c) return
c――――――――――――――――――――――★ 塑性座屈
          istat = 2
          dd= Member.d_B + Member.nc - delt
          ff=Member.P(1)*dd + Member.P(2)
          f1=dsqrt(ff)
c――――――――――――――――――――――★最大点の調整 alf
          ak = - 0.5*Member.alf*Member.P(1)/(ff*f1)
      endif
      return
c――――――――――――――――――――――――――――――★ state = 1      座屈後弾性変形範囲
      case (1)                                                       ! 10
         if(d_delt.lt.0)then                    !歪進行
         else                                   !歪逆転
         endif
      return
c――――――――――――――――――――――――――――――★ state = 2      非弾性圧縮側崩壊曲線
      case (2)                                                       ! 11
        if(d_delt.lt.0.0 )then                      !歪進行           ! 12
          dd= Member.d_B + Member.nc -  delt                          ! 13
          ff=Member.P(1)*dd+Member.P(2)
          f1=dsqrt(ff)
c――――――――――――――――――――――★最大点の調整 alf
          ak = - 0.5*Member.alf*Member.P(1)/(ff*f1)
        else                                    !歪逆転              ! 14
c――――――――――――――――――――――★増分変位調節
          Stress_x= Stress_x - ak*d_delt                             ! 15
          delt=delt - d_delt
c――――――――――――――――――――――★増分変位調節
          istat=3                                                    ! 16
          Member.d_Q = delt
          Member.N_Q = Stress_x
c――――――――――――――――――――――★ A 点移動計算
          x=Dlog(Member.Q(1)*(Member.d_D - delt) + 1.0)               ! 17
     *             -  Member.Q(2)*(Member.d_B - Member.d_D)
          if(x .le. 0.) x = 0.0
          Member.d_A = Member.d_C + x
c――――――――――――――――――――――★ P 点の計算
          Member.d_P = Member.d_A - (Member.d_B - delt)*Member.Q(3)   ! 18
          dd = Member.d_A  - Member.d_P
          ff = Member.P(3)*dd + 1.0
          f1 = dsqrt(ff)
          Member.N_P = 1./(ff*f1)
c――――――――――――――――――――――★ 接線剛性計算
          ddp= ak*d_delt                                             ! 19
          akx=ak
          ak = (Member.N_P - Member.N_Q)/(Member.d_P - Member.d_Q)
c――――――――――――――――――――――★増分変位調節
```

```
          if(dabs(akx).lt.dabs(ak)) then                                ! 20
              d_delt = 0.
              if(ak.ne.0.) d_delt=ddp/ak
          else                                                          ! 21
               ddp= ak*d_delt
          endif
          Stress_x= Stress_x + ddp                                      ! 22
          du= d_delt*Member.u_0
          Member.stress(1) = Stress_x * Member.Nt_0
c―――――――――――――――――――――――★増分変位調節
      endif
      return
c――――――――――――――――――――――――――――――――★ state = 3        座屈後除荷弾性範囲
      case (3)                                                          ! 23
        if(d_delt.gt.0.0 )then              ! 歪進行                   ! 24
            if(Stress_x .lt. Member.N_P) return                         ! 25
            istat=4                                                     ! 26
            dd= Member.d_A  - delt
            ff=Member.P(3)*dd + 1.
            f1=dsqrt(ff)
            ak =   1.5*Member.P(3)/(ff*ff*f1)
        else                                ! 歪逆行                   ! 27
            if(Stress_x .gt. Member.N_Q) return
            istat=2
            dd= Member.d_B + Member.nc - delt
            ff=Member.P(1)*dd + Member.P(2)
            f1=dsqrt(ff)
c―――――――――――――――――――――――★最大点の調整 alf
            ak = - 0.5*Member.alf*Member.P(1)/(ff*f1)
        endif
      return
c――――――――――――――――――――――――――――――――★ state = 4        引張側崩壊曲線
      case (4)                                                          ! 28
        if(d_delt.lt.0)then                 ! 歪逆転                   ! 29
c―――――――――――――――――――――――★増分変位調節
            Stress_x= Stress_x - ak*d_delt
            delt=delt - d_delt
c―――――――――――――――――――――――★増分変位調節
            istat=3                                                     ! 30
            Member.d_C = Member.d_A
c―――――――――――――――――――――――――★ B 点移動計算
            yyx=(Member.d_A - 1. + Member.N_c - Member.d_B)              ! 31
            y = (Member.d_A - 1. + Member.N_c - Member.d_B)*
     *           (delt - Member.d_P)/(Member.d_A - Member.d_P)
            Member.d_B = Member.d_B + y
c―――――――――――――――――――――――――★ P 点の設定
            Member.d_P = delt                                           ! 32
            Member.N_P = Stress_x
c―――――――――――――――――――――――――★ Q 点の計算
            Member.d_Q = Member.d_B -                                   ! 33
     *                   (Member.d_A - delt)/Member.Q(3)
            Member.d_D = Member.d_Q
            dd = Member.d_B  + Member.nc - Member.d_Q
            ff = Member.P(1)*dd + Member.P(2)
```

```
            f1 = dsqrt(ff)
c―――――――――――――――――――――――――★最大点の調整
            Member.N_Q = -Member.alf/(f1)                                  ! 34
c―――――――――――――――――――――――――★ 接線剛性計算
            ddp= ak*d_delt                                                 ! 35
            akx=ak
            ak = (Member.N_P - Member.N_Q)/(Member.d_P - Member.d_Q)
c―――――――――――――――――――――――――★増分変位調節
            if(dabs(akx).lt.dabs(ak)) then                                 ! 36
               d_delt = 0.
               if(ak.ne.0.) d_delt=ddp/ak
             else
                ddp= ak*d_delt                                             ! 37
             endif
             Stress_x= Stress_x + ddp                                      ! 38
             du= d_delt*Member.u_0
             Member.stress(1) = Stress_x * Member.Nt_0
c―――――――――――――――――――――――――★増分変位調節
          else                                                   ! 歪進行 ! 39
             if(delt.lt.Member.d_A) then
                  dd= Member.d_A  - delt
                  ff=Member.P(3)*dd + 1.
                  f1=dsqrt(ff)
                  ak =   1.5*Member.P(3)/(ff*ff*f1)
             else                                                          ! 40
c―――――――――――――――――――――――――★増分変位調節
                  dp = ak*d_delt
                  d_over=d_delt*( Stress_x - Member.N_t)/dp
c―――――――――――――――――――――――――★増分変位調節
                  istat = 5                                                ! 41
                  ak= Member.K_2
                  Stress_x = Member.N_t +ak*d_over
                  Member.stress(1) = Stress_x * Member.Nt_0
            endif
          endif
      return
c―――――――――――――――――――――――――――――――――★ state = 5        引張側降伏範囲
      case (5)                                                             ! 42
        if(d_delt.gt.0) return                                             ! 43
c―――――――――――――――――――――――――★増分変位調節
            Stress_x= Stress_x - ak*d_delt                                 ! 44
            du=0.
c―――――――――――――――――――――――――★増分変位調節
            istat =0                                                       ! 45
            ak = Member.K_1
            Member.d_A = delt
            Member.d_C = delt
            Member.d_P = delt
            Member.d_B = delt - 1. - member.nc
            Member.d_D =      Member.d_B
            Member.d_Q =      Member.d_B
            Member.N_P = 1.
            Member.N_Q = -Member.nc
            Member.N_c = -Member.nc
```

```
            Member.alf=dabs(Member.N_c/Member.nc)
          return
        endselect
        return
        end
```

上記のサブルーチンの説明を、プログラムの右側にあるコメント番号に従って説明する。

1. 変位増分がない場合は、このサブルーチンから抜ける。
2. 増分変位 d_delt、増分後の変位 delt、軸力 Stress_x を、引張側の第１折れ点の変位 Member.u_0 と軸力 Member.Nt_0 で割り、無次元量とする。
3. パラメータ istat を用いて、現在の状態に従って処理を分類する。
4. **ケース：０**　以降では、弾性状態の処理を行う。
5. 増分変位が正（引張側）か負（圧縮側）かをチェックし、以降では引張側の処理を行う。
6. 現在の応力が引張側の耐力 Member.N_t（無次元量）より小さい場合は、このサブルーチンを抜ける。大きい場合は、以下の処理を行う。
7. ここで、応力の超過分を調整する。まず、増分応力 dp を求める。さらに、超過分の変位を求める。
8. 状態を 5（引張側塑性状態）にする。接線剛性 ak に引張側第 2 剛性をセットする。無次元軸力 Stress_x をセットする。軸力を構造体成分にセットする。
9. 以降では、弾性圧縮側の処理を行う。軸力が圧縮側の座屈耐力より大きい場合は、このサブルーチンから抜ける。ここでは、全て塑性座屈としており、状態を 2 とする。状態：2 における初期接線剛性 ak を、次式より計算する。なお、圧縮側の座屈耐力式は、

$$n = 1/(p_1\delta + p_2)^{1/2}$$
$$p_1 = (10n_E - 1)/3; \quad p_2 = 4/n_E + 0.6$$

   であり、p1 と p2 は既に初期設定で計算し、構造体にセットされている。また、上式の接線勾配である接線剛性は次式で表される。

$$ak = -0.5\alpha p_1/(p_1\delta + p_2)^{3/2}$$

   ここで、α は最大点を調節する係数である。
10. **ケース：１**　状態：１の挙動を処理する。ここは、弾性座屈後の弾性挙動を示す。現在、この部分の処理を無視する。
11. **ケース：２**　以降で、状態：２の非弾性圧縮側崩壊挙動を処理する。

12. ここで、増分ひずみの正負をチェックし、以降で、負のひずみが進行する場合の処理を行う。
13. 圧縮側の座屈耐力式を用いて、接線剛性 ak を求める。なお、圧縮側の座屈耐力式は、

$$n = 1/(p_3\delta + 1)^{1/2}$$
$$p_3 = 1/(3.1n_E + 1.4)$$

であり、p3 は、既に初期設定で計算し、構造体にセットされている。また、上式の接線勾配である接線剛性は次式で表される。

$$ak = 0.5\alpha p_3/(p_3\delta + 1)^{3/2}$$

14. 以降では除荷によってひずみが逆転する場合について処理を行う。
15. 増分前の応力と変位を求める
16. 状態を 3 にセットする。除荷点の変位と軸力を構造体にセットする。
17. 次式を用いて、Ａ点の移動量を計算する。

$$x = \ln(q_1\bar{\delta}_a + 1) - q_2 s$$

18. 次式を用いて、Ｐ点の位置を計算する。

$$n_P = 1/(p_3\delta + 1)^{1/2}$$

19. Ｐ点とＱ点の値を用いて直線式を作り、その勾配より接線剛性を計算する。
20. ここでは、除荷前と後の剛性の大きさによって、増分変位と増分変位を調節する。調整後に、増分後応力と増分変位を求める。実際の増分後軸力を構造体にセットする。増分前の剛性が小さいとき、増分変位を調節する。
21. 増分後の剛性が小さいとき、増分応力を調節する。
22. 調整後に、増分後応力と増分変位を求める。実際の増分後軸力を構造体にセットする。
23. **ケース：３**　以降で、状態：３の座屈後除荷弾性挙動を処理する。
24. 増分ひずみが正か負かチェックし、正の場合はひずみが進行し、以下の処理を行う。負の場合の処理は処理 27 で行う。
25. ひずみが進行した場合、軸力がＰ点の軸力より小さい場合は、このサブルーチンより抜ける。Ｐ点を超えて大きくなった場合は、以下の処理を行う。
26. 状態を４にする。圧縮側の座屈耐力式より、次式を用いて接線剛性を計算する。

$$ak = \frac{dn}{d\delta} = -\frac{1.5 p_3}{(p_3 \delta + 1)^{\frac{5}{2}}}$$

27. ひずみが逆行する場合の処理を行う。まず、増分後軸力が Q 点の軸力より大きい場合はサブルーチンを抜ける。Q 点の軸力を超えて小さくなった場合は、以下の処理を行う。状態を 2 とし、処理 9 で説明した接線剛性を計算するための処理を行う。
28. **ケース：４**　以降で、状態：4 の引張側の崩壊挙動を処理する。
29. 増分ひずみが正か負かチェックし、負の場合はひずみが逆行し、以下の処理を行う。負の場合の処理は処理 39 で行う。
30. 増分前の軸力と変位を求める。
31. 次式を用いて、B 点の移動量 y を計算し、B 点を設定し直す。

$$y / y_0 = \delta_b / \delta_{b0}$$

32. P 点の変位と軸力をセットする。
33. Q 点の変位を求める。
34. Q 点の軸力を求める。
35. P 点と Q 点の値を用いて直線式を作り、その勾配より接線剛性を計算する。
36. ここでは、除荷前と後の剛性の大きさによって、増分変位と増分変位を調節する。調整後に、増分後応力と増分変位を求める。実際の増分後軸力を構造体にセットする。増分前の剛性が小さいとき、増分変位を調節する。
37. 増分後の剛性が小さいとき、増分応力を調節する。
38. 調整後に、増分後応力と増分変位を求める。実際の増分後軸力を構造体にセットする。
39. ひずみが進行する場合の処理を行う。増分後の変位が A 点の変位を超えるかどうかチェックし、超えない場合は以下のように接線剛性を計算する。超える場合は、処理 40 を行う。
40. 超過した部分の変位を求める。
41. 状態を 5 とする。接線剛性を引張側の第 2 剛性とする。増分後の軸力を計算し、実際の軸力を構造体にセットする。
42. **ケース：５**　以降で、状態：5 の引張側の降伏挙動を処理する。
43. 増分ひずみが正か負かチェックし、正の場合はひずみが進行するため、このサブルーチンを抜ける。
44. ひずみが逆行し、弾性復帰のための処理を行う。まず、増分前の軸力と変位を求める。

45. 状態を 0 とする。接線剛性を第１剛性とする。各ワーク点の変位と軸力をセットする。

3.4 MSS 免震弾塑性モデル

3.4.1 組み込み法

本節では、MSS(Multi Spring System)モデルの履歴について解説する。このモデルは、せん断剛性を円周上に配置した仮想的なモデルであり、円周上に置かれたスプリングは、円周面からの垂線方向の変位に対して、挙動するものとしている。このモデルは、立体骨組み用の免震デバイスの履歴に適している。SPACE では、このせん断モデルに弾性軸方向剛性を付加して、MSS モデルとしている。

最初に、このモデルを SPACE に組み込むための準備を、主サブルーチン submain_dynamic_a()で、次のように設定する。ここでは、まず、このモデルで必要となる構造体を動的配列として宣言し、次にモデルの個数分、動的領域を確保する。この動的領域を使用して、モデルの初期設定をサブルーチン Cal_MSS_dat()で行う。後は、他のモデルと同様に、線形剛性、非線形剛性、応力の弾塑性チェックの部分で、このモデルが行うべき処理を実行するサブルーチンを組み込むことになる。このモデルを静的解析に組み込むとき、サブルーチン Cal_unb_stress()でも必要になるので忘れないこと、また、このモデルの入出力仕様は、すべて標準仕様を用いており、この中で全てのデータが設定されることになる。

入力仕様は以下のようである。

(3)M_TYPE=5：MSS 免震弾塑性モデル（軸方向弾性、水平方向バイリニア弾塑性）

M_TYPE	E	GK1	A	GK2	QY	DUMMY	
5						0	
DUMMY	DUMMY	DUMMY	DUMMY	ANP	AQPV	AQPW	NM_TYPE
0	0	0	0				

E：ヤング率（tf/cm²）(軸方向剛性計算用)
GK1：水平方向せん断用第一剛性（tf/cm）
A：免震装置の断面積（軸力方向）（cm²）
GK2：水平方向せん断用第二剛性（tf/cm）
QY：水平剛性第一折れ点のせん断力（tf）
DUMMY：ダミー
ANP：表示用基準化軸力（tf）
AQPV：表示用基準化 y 方向せん断力（tf）

MSS 部材の軸方向弾性剛性は、A*E/L で計算される。

履歴モデルに０を設定すると、規定値をセットすることになり、規定値は、各要素タイプで異なった履歴モデルが設定されている。

AQPW：表示用基準化ｚ方向せん断力（tf）
（計算には関係なし。プレゼンテーションでこの変数を使用
している。）
NM_TYPE：１：バイリニア、２：デグレーディング型バイリニア

以下に、主サブルーチンで、上記に関連する部分を取り出して、表示する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /submain_dynamic_a
C       ―――――――――――――――――――――――――――――――――
C     ●   動的解析主プログラム（反復解法＋陰解法）Ver.3.00
C     ―――――――――――――――――――――――――――――――――
      ・
      ・
c―――――――――――――――――――――――――――――――★Model_No.5 3 次元 MSS 免振モデル
      record / MSS_work_s   / MSS_work
c―――――――――――――――――――――――――――――――★Model_No.5 3 次元免振モデル
      ALLOCATABLE :: MSS_work(:)
c―――――――――――――――――――――――――――――――★Model_No.5 3 次元免振モデル
      save MSS_work
c―――――――――――――――――――――――――――――――★Model_No.5 3 次元免振モデル
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
        ALLOCATE (MSS_work(n))
      endif
      ・
      ・
c―――――――――――――――――――――――――――――――★MSS モデルの初期設定
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      call Cal_MSS_dat(Member,Element,Model_type,
     *                 MSS_work,Parameter_C)
      endif
      ・
      ・
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―――――――――――――――――――――――――――――――――★ファイバー応力セット
      call Check_stress(Control,Control.type_analysis,Point,
     *     ak_nonlinear,Member,n_member,Model_type,
     *     Element,past_disp_point,est_ddisp_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)
       .
       .
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 )
       .
       .
      n=Model_type.n_m_model(5)
         if(n.ne.0) then
      DEALLOCATE (MSS_work )            !ok
      endif
       .
       .
```

このモデルで使用する構造体を以下に示す。構造体 element_s5 と member_s5 は、標準の構造体 element_s と member_s と同一領域であり、このモデル用に設計したものである。また、構造体 MSS_work_s は、このモデルのために新たに設定したもので、ワーク領域として使用される。この構造体は、上記のようにこのモデル数分動的領域を確保され、計算

終了後解放されることになる。

```
C     ―――――――――――――――――――――――――――――――――
C     ●　MSS 免震弾塑性モデル用（モデル No.5）element_s 構造体
C     ―――――――――――――――――――――――――――――――――
c     要素
      structure / element_s5/
      integer  element_type   !  要素タイプ
      integer  n_element      !  非線形要素番号
      real*8   E              !  ヤング係数
      real*8   GK1            !  第一剛性
      real*8   A              !  断面積
      real*8   GK2            !  第二剛性
      real*8   QY             !  折れ点のせん断力
      real*8   damy(5)        !  ダミー
      integer  nm_damp        !  部材減衰の有無
      integer  nm_type        !  履歴タイプ１:バイリニア、2:デグレーディング型バイリニア
      integer  n_section(5)   !  断面番号
      integer  nm_section(5)  !  ファイバー数
      real*8   ANP            !  軸方向耐力
      real*8   AMPY           !  y 軸塑性モーメント
      real*8   AMPZ           !  z 軸塑性モーメント
      real*8   dmm(3)         !  ダミー
      real*8   i_rigid_length       !  i 端剛域長さ
      real*8   j_rigid_length       !  j 端剛域長さ
      real*8   i_shear_G            !  i 端せん断剛性
      real*8   j_shear_G            !  j 端せん断剛性
      end structure
c     record /element_s5/ Element
c     ALLOCATABLE ::Element(:)
c     ALLOCATE (Element(n_element))
c
C     ―――――――――――――――――――――――――――――――――
C     ●　MSS 免震弾塑性モデル用（モデル No.5）member_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / member_s5/
      integer  nm_element         !  要素番号（入力した要素番号を示す）
      integer  element_type       !  要素タイプ番号
      integer  n_model            !  モデルの入れ物番号
      integer  n_model_type       !  モデル別の通し番号
      integer  n_element_type     !  要素タイプ別通し番号
      integer  analysis_3D        !  解析型 0:3D 1:2D(x-z) 2:2D(y-z)
      integer  nm_so              !  部材の層番号
      integer  nm_dll_element     !  DLL を用いた要素か（０；システム内要素、１：DLL 要素）
      integer  nm_point(2)        !  節点番号
      integer  irest(12)          !  部材両端の自由度番号表
      integer  ijp(2)             !  両端節点への結合状況（0:剛結合 1:ピン結合）
      integer  nm_analysis        !  部材解析種別（-1:弾性解析、その他：通常解析）
      integer  nm_group           !  部材グループ
      integer  nm_local_coord(2)  !  局所座標系の有無とその回転行列の番号
      integer  nm_damp            !  部材減衰の有無とその減衰行列の番号
      real*8   alength            !  長さ
```

```
      real*8   i_rigid_length     !  i 端剛域長さ
      real*8   j_rigid_length     !  j 端剛域長さ
      real*8   i_shear_G          !  i 端せん断剛性
      real*8   j_shear_G          !  j 端せん断剛性
      real*8   rot_x              !  部材主軸の回転角度（度）
      real*8   force(12)          !  部材両端の部材端力（釣合座標系）
      real*8   stress(12)         !  部材両端（部材座標系）
      real*8   rkb(6)             !  mss モデルで使用
      real*8   an_stress(10)      !  部材軸力（部材座標系）
      integer   d_stat(3)         !  断面の弾塑性状態  (1) i 端 (2) j 端  (3) 中央
      real*8   an_vv(10)          !  部材軸力（計算用内部変位ｖ）
      real*8   an_ww(10)          !  部材軸力（計算用内部変位w）
      end structure
c     record / member_s / Member
c     ALLOCATABLE :: Member (:)
c     ALLOCATE (Member (n_member))
c
C     ―――――――――――――――――――――――――――――――
C     ●   MSS モデル   MSS モデル：立体免震要素 用  Work エリア構造体
C     ―――――――――――――――――――――――――――――――
c
c     部材
      structure / MSS_work_s/
      integer  istat(16)
      real*8   d_smm(7,16)              !  ワークエリア
      end structure
c       record / MSS_work_s  /  MSS_work
c
```

このモデルの初期設定や履歴の追跡などの具体的な処理については、次節で解説する。ここでは、これらの処理を実行するサブルーチンがどの位置に組み込まれているかを中心に述べる。まず、線形剛性行列を計算するサブルーチン Cal_stiff_linear()で､以下のサブルーチンがコールされる。ここでは、線形の剛性行列が作成される。軸方向剛性 Member.rkb(1)(A11)とせん断剛性 Member.rkb(2)(rk)を用いて、剛性が計算される。円周上に配置された各々のせん断剛性は、次式を用いて、y 方向とｚ方向の部材のせん断剛性として、A22、A23、A33 として各々評価される。

$$\left.\begin{aligned} A22 &= \sum rk \cdot \cos\theta_i \cos\theta_i \\ A23 &= \sum rk \cdot \cos\theta_i \sin\theta_i \\ A33 &= \sum rk \cdot \sin\theta_i \sin\theta_i \end{aligned}\right\} \cdots\cdots (3.12)$$

ここで、座標系は部材座標系であり、また、A23 はｙ方向とｚ方向せん断剛性の連成項を表す。上式を用いて線形剛性行列が、サブルーチン Cal_lin_stiff_M5()で作成される。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_lin_stiff_M5
C       ―――――――――――――――――――――――――――――――――――
C     ●   線形剛性行列の計算(ok)
C     ―――――――――――――――――――――――――――――――――――
      subroutine Cal_lin_stiff_M5(Member,sm,cosin,nsprg)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s5    / Member
      dimension sm(12,12),cosin(2,16)
C     ―――――――――――――――――――――――――――――――――――
      do j = 1, 12
      do i = 1, 12
      sm(i,j) = 0.0D0
      enddo
      enddo
c――――――――――――――――――――――――――――――――★
      A22 = 0.0D0
      A23 = 0.0D0
      A33 = 0.0D0
      rk = Member.rkb(2)
      A11= Member.rkb(1)
      DO i = 1, nsprg
       c = cosin(1,i )
       s = cosin(2,i )
        A22 = A22 + rk*C*C
        A23 = A23 + rk*S*C
        A33 = A33 + rk*S*S
      enddo
c――――――――――――――――――――――――――――――――★
      sm(1,1) =  A11
      sm(7,1) = -A11
      sm(1,7) = -A11
      sm(7,7) =  A11
      sm(2,2) =  A22
      sm(2,3) =  A23
      sm(3,2) =  A23
      sm(3,3) =  A33
      sm(8,8) =  A22
      sm(8,9) =  A23
      sm(9,8) =  A23
      sm(9,9) =  A33
      sm(8,2) = -A22
      sm(8,3) = -A23
      sm(9,2) = -A23
      sm(9,3) = -A33
      sm(2,8) = -A22
      sm(2,9) = -A23
      sm(3,8) = -A23
      sm(3,9) = -A33
      return
      return
      end
```

次に、非線形剛性行列は、サブルーチン Get_nonlinear_stiff()より、次のサブルーチンがコールされ、計算される。ここでの処理も、円周上の各せん断剛性が個々異なることを除いて、線形剛性行列を計算する処理方法と全く同じである。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_nonlin_stiff_M5
C       ―――――――――――――――――――――――――――――――――
C     ●   Model_No.5 3 次元免振モデル（MSS モデル）
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_nonlin_stiff_M5(Member,Element,n_sprg,cosin,
     *                               MSS_work,ak)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s5    / Member
      record / element_s5   / Element
      record / MSS_work_s     / MSS_work
     dimension ak(12,12),cosin(2,16)
C     ―――――――――――――――――――――――――――――――――
      A11=Member.rkb(1)
      call MSS_tan_K(cosin,n_sprg,A11,MSS_work.d_smm(1,1),ak)
     return
     end
```

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /MSS_tan_K
C       ―――――――――――――――――――――――――――――――――
C     ●   免震装置 MSS モデルのデータを初期セットする(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine MSS_tan_K(cosin,nsprg,A11,dsmm,sm)
      implicit real*8(A-H,O-Z)
      dimension sm(12,12),dsmm(7,16),cosin(2,16)
      do i = 1, 12
      do j = 1, 12
      sm(i,j) = 0.0D0
      enddo
      enddo
c――――――――――――――――――――――――――――――――――――――★
      A22 = 0.0D0
      A23 = 0.0D0
      A33 = 0.0D0
      DO i = 1, nsprg
       rk =dsmm(7,i )
       c = cosin(1,i )
       s = cosin(2,i )
        A22 = A22 + rk*C*C
        A23 = A23 + rk*S*C
        A33 = A33 + rk*S*S
      enddo
c――――――――――――――――――――――――――――――――――――――★
      sm(1,1) =  A11
```

```
      sm(7,1) = -A11
      sm(1,7) = -A11
      sm(7,7) =  A11
      sm(2,2) =  A22
      sm(2,3) =  A23
      sm(3,2) =  A23
      sm(3,3) =  A33
      sm(8,8) =  A22
      sm(8,9) =  A23
      sm(9,8) =  A23
      sm(9,9) =  A33
      sm(8,2) = -A22
      sm(8,3) = -A23
      sm(9,2) = -A23
      sm(9,3) = -A33
      sm(2,8) = -A22
      sm(2,9) = -A23
      sm(3,8) = -A23
      sm(3,9) = -A33
      return
      end
```

部材の弾塑性状態をチェックするサブルーチン Check_stress()から以下のサブルーチンがコールされる。このサブルーチンが MSS モデルの履歴を管理している。ただし、履歴モデルの内容は次節で説明する。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_check_stiff_M5
C       ―――――――――――――――――――――――――――――――
C     ●   Model_No.2 3 次元せん断弾塑性モデル
C     ―――――――――――――――――――――――――――――――
      subroutine Cal_check_stiff_M5(Member,Element,MSS_work,vv,vpp,
     *                              n_sprg,cosin )
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s5    / Member
      record / element_s5   / Element
      record / MSS_work_s   /  MSS_work
     dimension vv(12),vpp(12),cosin(2,32)
     du=vv(8)-vv(2)
     dv=vv(9)-vv(3)
     uu=vpp(8)-vpp(2)
     vv=vpp(9)-vpp(3)
     ip= Element.nm_type
      call MSS_ep(ip,MSS_work.istat(1),du,dv,uu,vv,n_sprg,
     *            Member.rkb(1),MSS_work.d_smm(1,1),cosin)
      return
      end
```

3.4.2 MSS モデルの履歴

本節では、MSS モデルの初期設定と履歴を追跡する処理について解説する。最初に、初期設定するサブルーチン Cal_MSS_dat()及び関連するサブルーチン MSS_initial_set()と MSS_dt_set()の内容を以下に示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_MSS_dat
C       ―――――――――――――――――――――――――――――――――
C     ●   免震装置のデータを初期セットする(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_MSS_dat(Member,Element,Model_type,MSS_work,
     *                       Parameter_C)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s5    / Member
      record / element_s5   / Element
      record / MSS_work_s   / MSS_work
      record / parameter_s  / Parameter_C
      record / n_model_s    / Model_type
      dimension Member(*),Element(*),MSS_work(*)
C     ―――――――――――――――――――――――――――――――――
      n_sprg=16                 !MSS スプリング数のセット                  ! 1
      Model_type.n_spring=n_sprg
      call MSS_initial_set(n_sprg,Model_type.cosin(1,1))                  ! 2
      tcos=0.                                                             ! 3
      tcos2=0.
      do k=1,n_sprg                                                       ! 4
        c=Model_type.cosin(1,k)
        tcos=tcos+abs(c)
        tcos2=tcos2+c*c
      enddo
      do i=1,Parameter_C.n_member                                         ! 5
      iet  = Member(i).element_type
        if(iet.eq.5) then                                                 ! 6
        ie   = Member(i).nm_element
        iee  = Member(i).n_model_type
        a=Element(ie).A
        e=Element(ie).E
        al=Member(i).alength
        ak1=Element(ie).GK1
        ak2=Element(ie).GK2
        qy= Element(ie).QY
        ih= Element(ie).nm_type
        call  MSS_dt_set(ih,Member(i).rkb(1),Model_type.cosin(1,1),        ! 7
     *                 n_sprg,e,a,al,ak1,ak2,qy,
     *   MSS_work(iee).istat(1),MSS_work(iee).d_smm(1,1),tcos,tcos2)
        endif
      enddo
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /MSS_initial_set
C       ―――――――――――――――――――――――――――――――――――
C     ●   免震装置の角度を先にセットする(ok)
C     ―――――――――――――――――――――――――――――――――――
      subroutine MSS_initial_set(nsprg,cosin)
      implicit real*8(A-H,O-Z)
      dimension cosin(2,16)
      pi=acos(-1.d0)                                          ! 8
      do k=1,nsprg
      theta = pi*dble(k-1)/dble(nsprg)                        ! 9
      cosin(1,k)=cos(theta)                                   ! 10
      cosin(2,k)=sin(theta)
      enddo
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /MSS_dt_set
C       ―――――――――――――――――――――――――――――――――――
C     ●   免震装置 MSS モデルのデータを初期セットする(ok)
C     ―――――――――――――――――――――――――――――――――――
c     ih   :
c     nsprg:スプリングの数
c     e    :ヤング係数
c     a    :断面積
c     al   :長さ
c     ak1  :第一剛性
c     ak2  :第二剛性
c     qy   :折れ点のせん断力
c     istat:
C     ―――――――――――――――――――――――――――――――――――
      subroutine MSS_dt_set(ih,rkb0,cosin,nsprg,e,a,al,
     *                ak1,ak2,qy,istat,dsmm,tcos,tcos2)
      implicit real*8(A-H,O-Z)
      dimension rkb0(12),cosin(2,16),dsmm(7,16),istat(32)
      rkb0(1)=e*a/al                                          ! 11
      rkb0(2)=ak1/tcos2                                       ! 12
      rkb0(3)=ak2/tcos2                                       ! 13
      rkb0(4)=qy/tcos                                         ! 14
      rkb0(5)=rkb0(4)/rkb0(2)                                 ! 15
      rkb0(6)=rkb0(2)                                         ! 16
      if(ih.eq.1) then                                        ! 17
      do k=1,nsprg                                            ! 18
      dsmm(1,k)=rkb0(4)
      dsmm(2,k)=rkb0(5)
      dsmm(3,k)=-rkb0(4)
      dsmm(4,k)=-rkb0(5)
      dsmm(7,k)=rkb0(2)
      istat(k)=0
      enddo
      else                                                    ! 19
      do k=1,nsprg
```

```
      dsmm(1,k)=rkb0(5)
      dsmm(2,k)=-rkb0(5)
      dsmm(3,k)=rkb0(5)
      dsmm(4,k)=-rkb0(5)
      dsmm(5,k)=rkb0(5)
      dsmm(6,k)=-rkb0(5)
      dsmm(7,k)=rkb0(2)
      istat(k)=0
      enddo
      endif
      return
      end
```

上記のサブルーチンの説明を、プログラムの右側にあるコメント番号に従って説明する。

1. MSS モデルのスプリングの数を 16 に設定し、構造体にセットする。
2. サブルーチン MSS_initial_set()を用いて、円周上スプリング位置の角度を計算し、構造体にセットする。
3. 変数 tcos と tcos2 をゼロセットする。
4. せん断剛性を各スプリングに割り振るために、各位置における COS と COS*COS の値を全スプリングについて和をとる。
5. 全部材について以下の処理を行う。
6. この部材が MSS モデルであるかどうかチェックし、同モデルである場合は、以下の処理を行う。該当する部材の断面積、ヤング係数、部材長さ、第１せん断剛性、第２せん断剛性、折れ点のせん断力、履歴モデル番号を、構造体から変数にセットする。
7. サブルーチン MSS_dt_set()を用いて、該当する部材の各スプリング関するデータを初期設定する。
8. サブルーチン MSS_initial_set()では、まず、2πの値を正確に求める。次に、この 2πを利用して、全スプリングについて以下の処理を行う。
9. 該当するスプリングの位置を、角度（ラジアン）で求める。
10. そのスプリングの位置である COS および SIN を求め、配列にセットする。
11. サブルーチン MSS_initial_set()では、各部材の剛性などのデータ、及び各スプリングのデータをセットする。まず、軸方向剛性を計算し、配列にセットする。
12. 部材の第１せん断剛性をスプリング１個分に変更して、配列にセットする。

13. 部材の第２せん断剛性をスプリング１個分に変更して、配列にセットする。
14. 部材の第１折れ点のせん断力をスプリング１個分に変更して、配列にセットする。
15. 第１折れ点の変位を計算し、配列にセットする。
16. スプリングの第１せん断剛性を配列にセットする。
17. 履歴モデルをチェックし、バイリニア型の場合は、以下の処理を行う。
18. バイリニア型の履歴用に、全スプリングに対しデータをセットする。
19. 最大点指向型バイリニアの履歴用に、全スプリングに対しデータをセットする。

次に、MSS モデルの履歴追跡は、サブルーチン Cal_check_stiff_M5() からコールされるサブルーチン MSS_ep()で行われる。ここでも、履歴モデルに階層構造が見られ、現在、2 種類の履歴モデルが組み込まれている。一つ目は、バイリニア型であり、他の一つは最大点指向型バイリニアである。以下に、サブルーチン MSS_ep()を示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /MSS_ep
C       ―――――――――――――――――――――――――――――――
C     ●   免震装置 MSS モデルの弾塑性チェックする(ok)
C     ―――――――――――――――――――――――――――――――
      subroutine MSS_ep(ip,istmss,du,dv,uu,vv,nsprg,
     *            rkb0,dsmm,cosin)
      implicit real*8(A-H,O-Z)
      dimension istmss(16),dsmm(7,16),rkb0(12),cosin(2,16)
C     ―――――――――――――――――――――――――――――――
      if(ip.eq.1) then
        do k=1,nsprg
        cc=cosin(1,k)
        ss=cosin(2,k)
        uux=uu*cc + vv*ss
        dux=du*cc + dv*ss
        dpx=dux*dsmm(7,k)
        call MSS_epb(istmss(k),dsmm(7,k),uux,dpx,dux,
     &     rkb0(4),rkb0(5),rkb0(2),rkb0(3),dsmm(1,k),
     &     dsmm(2,k),dsmm(3,k),dsmm(4,k))
        enddo
      elseif(ip.eq.2) then
        do k=1,nsprg
        cc=cosin(1,k)
        ss=cosin(2,k)
        uux=uu*cc + vv*ss
        dux=du*cc + dv*ss
        dpx=dux*dsmm(7,k)
```

```
        call MSS_epd(istmss(k),dsmm(7,k),uux,dpx,dux,
     &     rkb0(4),rkb0(5),rkb0(2),rkb0(3),dsmm(1,k),
     &     dsmm(2,k),dsmm(3,k),dsmm(4,k),dsmm(5,k),dsmm(6,k))
        enddo
       endif
       return
       end
```

上記のように、ここでは、履歴モデルによって、2 種類のモデルが管理されており、2 種のサブルーチンがコールされている。これらのサブルーチンの内容については、次節で説明する。

3.4.3 MSS モデルのスプリング履歴（バイリニア型）

本節では、バイリニア型の履歴を追跡するサブルーチンを解説する。図 3-6 は、この履歴モデルを図示するものであり、履歴を制御する状態パラメータ istat は、3 つに分類される。

状態：0 は、弾性状態を表し、塑性状態を経験しない場合は、せん断力は、p1 と p2 の間の値となる。この耐力 p1 と p2 を超えると、骨格曲線上の第 2 勾配上を移動することになる。正側の第 2 剛性上の骨格曲線を状態：1 と云い、負側の第 2 勾配上の骨格曲線を状態：2 とする。

状態：1 で、正の荷重増分が生じると、履歴はこの第 2 勾配上をそのまま進行するが、逆に除荷となると状態は 0 となり、弾性復帰することとなる。この場合の状態：0 では、正負の最大耐力は求め直され、釣合点 (pp1,uu1) と (pp2,uu2) の点の間を移動する。

同様に、状態：2 で、負の荷重増分が生じると、履歴はこの第 2 勾配上をそのまま進行するが、逆に除荷となると、状態は 0 となり、弾性復帰することとなる。この場合の状態：0 では、正負の最大耐力は求め直され、釣合点(pp1,uu1)と(pp2,uu2)の点の間を移動する。

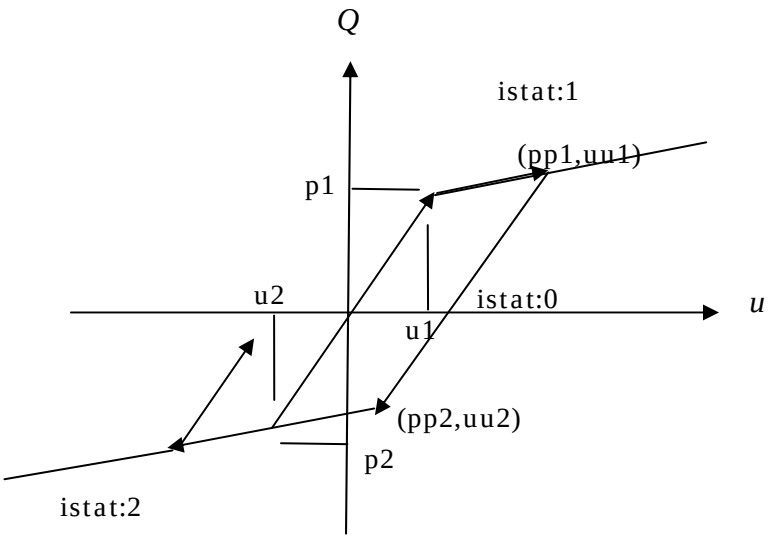

図 3-6　MSS モデルのバイリニア型履歴特性

図 3-6 を参考にして、次のサブルーチンを検討しよう。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /MSS_epb
C       ―――――――――――――――――――――――――――――――――――
C     ●   bilinear type for a sinple spring(ok)
C     ―――――――――――――――――――――――――――――――――――
      subroutine MSS_epb(istat,ep,uu,dp,du,p1,u1,
     *                   ak1,ak2,pp1,uu1,pp2,uu2)
      implicit real*8(A-H,O-Z)
c   istat    : スプリングの状態
c   ep       : スプリングの接線剛性
c   uu       : スプリングの増分後の変位
c   dp       : スプリングの増分せん断力
c   du       : スプリングの増分変位
c   p1       : スプリングの正側第１折れ点せん断力
c   u1       : スプリングの正側第１折れ点変位
c   ak1      : スプリングの第１剛性
c   ak2      : スプリングの第２剛性
c   pp1      : スプリングの現時点における正側第１折れ点せん断力
c   uu1      : スプリングの現時点における正側第１折れ点変位
c   pp2      : スプリングの現時点における負側第１折れ点せん断力
c   uu2      : スプリングの現時点における負側第１折れ点変位
C     ―――――――――――――――――――――――――――――――――――
      yy(a1,x1,y1,xx)=a1*(xx-x1)+y1                                  ! 1
      p2=-p1                                                         ! 2
      u2=-u1
      ak3=ak2
      if(du.eq.0) return                                             ! 3
c―――――――――――――――――――――――――――――――★ state = 0
      if(istat.eq.0)then                                             ! 4
           pp=yy(ak1,uu1,pp1,uu)                                      ! 5
c―――――――――――――――――――――――――★ 増分変位が正
        if(du.gt.0.0) then                                           ! 6
           if(pp.lt.pp1) then                                        ! 7
            ep=ak1
           else                                                      ! 8
            ep=ak2
            istat=1
           endif
c―――――――――――――――――――――――――★増分変位が負
        else                                                         ! 9
           if(pp.gt.pp2) then
            ep=ak1
           else                                                      ! 10
            ep=ak3
            istat=2
           endif
        endif
        return
c―――――――――――――――――――――――――――――――★ state = 1
      elseif(istat.eq.1) then                                        ! 11
c―――――――――――――――――――――――――★増分変位が正
         if(du.ge.0.0) return                                        ! 12
c―――――――――――――――――――――――――★増分変位が負で除荷
           pp=yy(ak2,u1,p1,uu)                                       ! 13
```

```
            uu1=uu-du
            pp1=pp-dp
            call pntset(pp1,uu1,ak1,p2,u2,ak3,pp2,uu2)                    ! 14
            istat=0
            ep=ak1
            return
c―――――――――――――――――――――――――――――★ state = 2
      elseif(istat.eq.2) then                                             ! 15
c―――――――――――――――――――――――――★増分変位が負
          if(du.le.0.) return                                             ! 16
c―――――――――――――――――――――――――★増分変位が正で除荷
            pp=yy(ak3,u2,p2,uu)                                           ! 17
            uu2=uu-du
            pp2=pp-dp
            call pntset(pp2,uu2,ak1,p1,u1,ak2,pp1,uu1)                    ! 18
            istat=0                                                       ! 19
            ep=ak1
      endif
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /pntset
C       ―――――――――――――――――――――――――――――――――
C     ●    ２直線の交点を求める
C     ―――――――――――――――――――――――――――――――――
      subroutine pntset(p1,u1,a1,p2,u2,a2,p,x)
      implicit real*8(A-H,O-Z)
      if(a2.eq.0.) then                                                   ! 20
        p=p2
        x=(p2-p1)/a1+u1
      else                                                                ! 21
        x=(p2-p1+a1*u1-a2*u2)/(a1-a2)
        p=a1*(x-u1)+p1
      endif
      return
      end
```

1．1 次式の値を求める内部関数を定義する。
2．負側の折れ点位置を正側の折れ点位置からコピーする。負側の第 2 剛性を正側の第 2 剛性よりコピーする。
3．増分変位がない場合は、このサブルーチンを抜ける。
4．現在の状態をチェックし、以降では、istat=0 の場合の処理を行う。
5．増分後の変位から、せん断力を求める。
6．増分変位が正方向の場合は以下の処理を行う。負の場合は、処理 9 に分岐する。
7. 現在のせん断力が正方向の状態：0 の耐力 pp1 より大きいか小さいかチェックする。小さい場合は、接線剛性 ep に第 1 剛性 ak1 をセット

する。
8. 上記でない場合は、せん断剛性を第２剛性に変更し、istat=1 として、状態：1 とする。
9. 増分変位が負の場合の処理を行う。まず、現在のせん断力が、負側の耐力 pp2 より大きいかどうかチェックする。大きい場合は、接線剛性を第１剛性とする。
10. 上記でない場合は、接線剛性を負側の第２剛性 ak3 にセットし、状態を 2 とするため、istat=2 とする。
11. 以降は、状態：１の場合の処理である。
12. 増分変位が正方向もしくは 0 の場合は、このサブルーチンから抜ける。
13. 除荷となり、剛性は弾性復帰する。現時点のせん断力を求める。増分変位と増分せん断力より、増分前の変位とせん断力を求め、この値を正方向の折れ点位置(pp1,uu1)とする。
14. サブルーチン pntset()を用いて、負側の折れ点位置 pp2,uu2 を求める。次に、状態を 0 とし、接線剛性を第１剛性とする。
15. 以降は、状態：２の場合の処理である。
16. 増分変位が負の場合は、処理を中止し、このサブルーチンより抜ける。
17. 除荷となり、剛性は弾性復帰する。現時点のせん断力を求める。増分変位と増分せん断力より、増分前の変位とせん断力を求め、この値を負方向の折れ点位置 pp2,uu2 とする。
18. サブルーチン pntset()を用いて、正側の折れ点位置 pp1,uu1 を求める。
19. 次に、状態を 0 とし、接線剛性を第１剛性とする。
20. サブルーチン pntset()では、２直線の交点位置を求める。まず、勾配 a2 がゼロかどうかをチェックする。ゼロの場合の式を用いて交点を計算する。
21. 上記以外の場合、ここに書かれたコードの式を用いて交点を計算する。

3.4.4 MSS モデルのスプリング履歴（最大点指向型バイリニア）

本節では、最大指向型の履歴モデルについて解析する。このモデルの骨格曲線は、バイリニア型であり、図 3-7a に示すように塑性域に達した後で除荷となり、弾性復帰するが、せん断力が 0 の位置で、反対側の

履歴の最大点をねらって接線勾配が決定される。ただし、反対側が未だ塑性域に達していない場合の最大点は、弾性から塑性に入る折れ曲がり点としている。ここでの正側の最大点は、(pp1,uu1)とし、負側の最大点は、(pp2,uu2)とする。

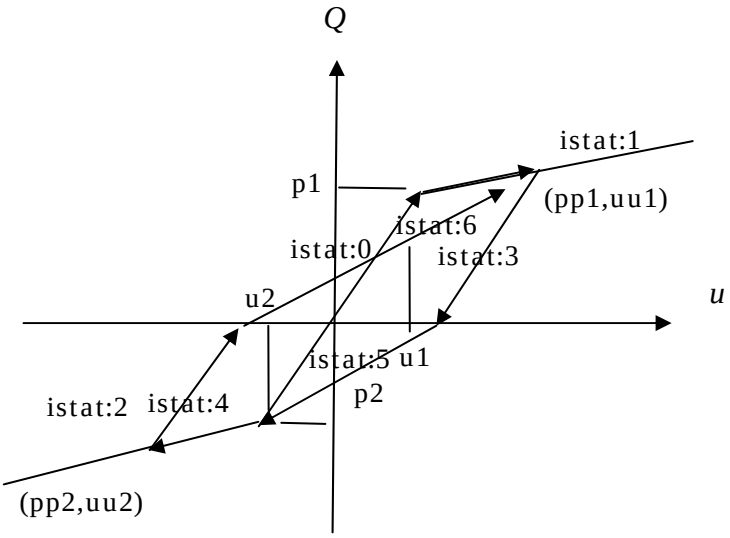

図 3-7a　MSS モデルの最大点指向型バイリニア履歴特性

この履歴を制御するパラメータは、状態:istat であり、全部で 7 領域に分類される。状態：0 は弾性状態であり、その境界は正側負側共に、骨格曲線の折れ曲がり点となる。状態：1 と状態：２は、骨格曲線の正側と負側の第 2 勾配であり、両者共に、変位が進行する場合は、そのままの状態を維持し、除荷となると反転して状態が各々3 と 4 になる。

状態：３と４では、弾性状態であり、接線剛性は第１剛性とする。この状態の境界は、両者とも、2 種類存在し、例えば、状態：３では、状態：１からの場合と状態：6 からの場合とでは、図 3-7b に示されるようにその挙動が異なる。前者の上界は、最大変位点の(pp1,uu1)であり、また、後者では、(pp5,uu5)である。なお、下界はせん断力が 0 となる(pp3,uu3)となる。

次に、状態:５と６では、接線剛性は常に最大点に向かう線分となり、この剛性は、計算して求めることになる。この 2 つの状態では、進行方向にはその最大点が境界であり、履歴が変転する場合は、除荷となり、各々状態４と３に転移する。

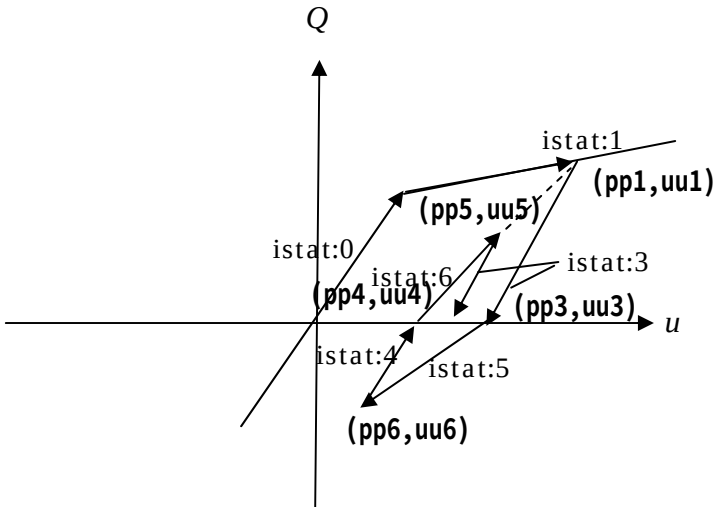

図 3-7b　MSS モデルの最大点指向型バイリニア履歴特性

上記の最大点指向型を追跡するサブルーチン MSS_epd()を以下に示す。

```
C     ------------------------------------------
C     ●   SUBROUTINE /MSS_epd
C       ------------------------------------------
C     ●   degrading bilinear type for a sinple spring
C     ------------------------------------------
      subroutine MSS_epd(istat,ep,uu,dp,du,p1,u1,
     *                   ak1,ak2,uu1,uu2,uu3,uu4,uu5,uu6)
      implicit real*8(A-H,O-Z)
       yy(a1,x1,y1,xx)=a1*(xx-x1)+y1                                  ! 1
c                      initial set
c                     dp  : initial stress
      p2=-p1                                                         ! 2
      u2=-u1
      ak3=ak2
      if(du.eq.0) return                                             ! 3
      if(istat.eq.0)then                                             ! 4
        pp=yy(ak1,0.,0.,uu)                                          ! 5
        if(du.gt.0.0) then                                           ! 6
           if(pp.lt.p1) return                                       ! 7
            ep=ak2                                                   ! 8
            istat=1
        else                                                         ! 9
           if(pp.gt.p2) return
            ep=ak3                                                   ! 10
            istat=2
        endif
      elseif(istat.eq.1) then                                        ! 11
         if(du.ge.0.0) return                                        ! 12
           uu1=uu-du                                                 ! 13
           uu5=uu1
           pp1=yy(ak2,u1,p1,uu1)
```

```
      call pntset(pp1,uu1,ak1,0.,0.,0.,pp3,uu3)                    ! 14
      istat=3
      ep=ak1
elseif(istat.eq.2) then                                            ! 15
     if(du.le.0.) return
      uu2=uu-du                                                    ! 16
      uu6=uu2
      ep=ak1
     pp2=yy(ak2,u2,p2,uu2)
      call pntset(pp2,uu2,ak1,0.,0.,0.,pp4,uu4)                    ! 17
      istat=4
elseif(istat.eq.3) then                                            ! 18
   pp=yy(ak1,uu3,0.,uu)                                            ! 19
   if(du.ge.0.0) then                                              ! 20
     pp5=yy(ak1,uu3,0.,uu5)                                        ! 21
       if(pp.lt.pp5) return                                        ! 22
       if(uu1.eq.uu5)then                                          ! 23
        istat=1
        ep=ak2
        else                                                       ! 24
        ep=yy(ak2,u1,p1,uu1)/(uu1-uu4)
        istat=6
        endif
    else                                                           ! 25
     if(pp.ge.0.) return
     istat=5
     ep=-yy(ak3,u2,p2,uu2)/(uu3-uu2)
     endif
elseif(istat.eq.4) then                                            ! 26
      pp=yy(ak1,uu4,0.,uu)                                         ! 27
     if(du.le.0.) then                                             ! 28
       pp6=yy(ak1,uu4,0.0,uu6)                                     ! 29
       if(pp.gt.pp6) return                                        ! 30
       if(uu2.eq.uu6) then                                         ! 31
        istat=2
        ep=ak3
        else                                                       ! 32
        istat=5
        ep=-yy(ak3,u2,p2,uu2)/(uu3-uu2)
        endif
      else                                                         ! 33
        if(pp.lt.0.) return                                        ! 34
         istat=6
         ep=yy(ak2,u1,p1,uu1)/(uu1-uu4)
      endif
elseif(istat.eq.5) then                                            ! 35
     pp=yy(ep,uu3,0.,uu)                                           ! 36
    if(du.gt.0.0) then                                             ! 37
     uu6=uu-du
     pp6=yy(ep,uu3,0.,uu6)
      call pntset(pp6,uu6,ak1,0.,0.,0.,pp4,uu4)                    ! 37
      istat=4
     ep=ak1
     else                                                          ! 38
```

```
        pp2=yy(ak3,u2,p2,uu2)                                    ! 39
        if(pp.gt.pp2) return                                     ! 40
        ep=ak3
        istat=2
       endif
      elseif(istat.eq.6) then                                    ! 41
       pp=yy(ep,uu4,0.,uu)                                       ! 42
       if(du.le.0.) then                                         ! 43
         uu5=uu-du
         pp5=yy(ep,uu4,0.,uu5)
        call pntset(pp5,uu5,ak1,0.,0.,0.,pp3,uu3)                ! 44
         istat=3
         ep=ak1
       else                                                      ! 45
         pp1=yy(ak2,u1,p1,uu1)                                   ! 46
         if(pp.lt.pp1) return                                    ! 47
         istat=1
          ep=ak2
       endif
      endif
      return
      end
```

1．1 次式の値を求める内部関数を定義する。
2．負側の折れ点位置を正側の折れ点位置からコピーする。負側の第 2 剛性を正側の第 2 剛性よりコピーする。
3．増分変位がない場合は、このサブルーチンを抜ける。
4．現在の状態をチェックし、以降では、istat=0 の場合の処理を行う。
5．増分後の変位から、せん断力を求める。
6．増分変位が正方向の場合は以下の処理を行う。負の場合は、処理 9 に分岐する。
7．現在のせん断力が正方向の状態：0 の耐力 p1 より大きいか否かをチェックする。小さい場合は、このサブルーチンを抜ける。
8．上記でない場合は、せん断剛性を第 2 剛性に変更し、istat=1 として、状態：1 とする。
9．増分変位が負の場合の処理を行う。まず、現在のせん断力が、負側の耐力 p2 より大きいか否かをチェックする。大きい場合は、このサブルーチンを抜ける。
10．上記でない場合は、接線剛性を負側の第 2 剛性 ak3 にセットし、状態を 2 とするため、istat=2 とする。
11．以降は、状態：１の場合の処理である。
12．増分変位が正方向もしくは 0 の場合は、このサブルーチンから抜け

る。

13. 除荷となり、剛性は弾性復帰する。現時点のせん断力を求める。増分変位と増分せん断力より、増分前の変位とせん断力を求め、この値を正方向の折れ点位置 pp1,uu1 とする。最大点変位として、ワーク領域 uu5 にこの折れ点変位をセットする。
14. サブルーチン pntset()を用いて、軸力 0 の折れ点位置 pp3,uu3 を求める。次に、状態を 3 とし、接線剛性を第 1 剛性とする。
15. 以降は、状態：２の場合の処理である。増分変位が負の場合は、処理を終了させ、このサブルーチンから抜ける。
16. 除荷となり、剛性は弾性復帰する。現時点のせん断力を求める。増分変位と増分せん断力より、増分前の変位とせん断力を求め、この値を負方向の折れ点位置 pp2,uu2 とする。負側の最大点として、この折れ点変位 uu2 を uu6 にセットする。接線剛性として第 1 剛性をセットする。
17. サブルーチン pntset()を用いて、せん断力が 0 の折れ点位置 pp4,uu4 を求める。次の状態として、istat=4 とする。
18. 以降は、状態：３の場合の処理である。
19. 現在のせん断力 pp を求める。
20. 増分変位が正かどうかチェックし、正の場合は、以下の処理を行い、負の場合は、処理 24 に分岐する。
21. 増分変位が正の場合は、最大点 uu5 を用いて、この状態の最大せん断力を求める。
22. 現在のせん断力が耐力を超えていない場合は、処理を中止し、このサブルーチンより抜ける。
23. 変位 uu1 と uu5 が等しいとき状態を 1 とし、接線剛性を第 2 剛性とする。
24. 上記が等しくない場合、状態を 6 とし、正側の最大点をねらって接線剛性を計算し直す。
25. 増分変位が負の場合、現在のせん断力が折れ点である 0 より大きいときは処理を中止し、このサブルーチンより抜ける。逆にせん断力が 0 以下のとき、状態を 5 とし、負側の最大点をねらって接線剛性を計算する。
26. 以降は、状態：４の場合の処理である。
27. 現在のせん断力を計算する。
28. 増分変位の正負をチェックし、負の場合は以下の処理を行う。また、正の場合は、処理 33 に分岐する。

29. 変位 uu6 を用いて、この状態の耐力であるせん断力 pp6 を求める。
30. 現在のせん断力が耐力を超えていない場合は処理を中止し、このサブルーチンより抜ける。
31.　変位 uu2 と uu6 が等しいとき、状態を 2 とし、接線剛性を第 2 剛性 ak3 とする。
32. 上記が等しくない場合、状態を 5 とし、負側の最大点をねらって接線剛性を計算し直す。
33. 増分変位が正の場合、現在のせん断力が折れ点である 0 より小さい場合は、処理を中止し、このサブルーチンより抜ける。せん断力が 0 以下の場合、状態を 6 とし、正側の最大点をねらって接線剛性を計算する。
34. サブルーチン pntset()を用いて、負側の折れ点位置 pp2,uu2 を求める。次に、状態を 0 とし、接線剛性を第 1 剛性とする。
35. 以降は、状態：５の場合の処理である。
36. 現在のせん断力 pp を求める。
36. 増分変位の正負をチェックし、正の場合は以下の処理を行う。また、負の場合は処理 38 に分岐する。ここでは、増分前の変位を求め、pp6 とし、また、その点のせん断力を pp6 とする。
37. サブルーチン pntset()を用いて、せん断力 0 の位置 pp4,uu4 を求める。次に、状態を 4 とし、接線剛性を第 1 剛性とする。
38. 増分変位が負の場合、以下の処理を行う。
39.　負側の最大耐力 pp2 を求める。
40. 負側の最大耐力 pp2 より、せん断力が大きいか否かをチェックし、値が大きければこの耐力を超えていないので、サブルーチンを抜ける。超えている場合は、接線剛性を第 2 剛性 ak3 とし、状態を 2 とする。
41. 以降は、状態：６の場合の処理である。
42. 現在のせん断力 pp を求める。
43. 増分変位の正負をチェックし、負の場合は以下の処理を行う。また、正の場合は、処理 45 に分岐する。ここでは、増分前の変位を求め、pp5 とし、また、その点のせん断力を pp5 とする。
44. サブルーチン pntset()を用いて、せん断力 0 の位置 pp3,uu3 を求める。次に、状態を 3 とし、接線剛性を第 1 剛性とする。
45. 増分変位が正の場合、以下の処理を行う。
46. 正側の最大耐力 pp1 を求める。
47. 正側の最大耐力 pp1 より、せん断力が大きいかどうかチェックし、

値が小さければこの耐力を超えていないので、サブルーチンを抜ける。超えている場合は、接線剛性を第２剛性 ak2 とし、状態を１とする。

以上で、MSS モデルに組み込まれているバイリニア型と最大点指向型バイリニアの履歴を追跡するサブルーチンを検討した。この２つのサブルーチンでは、せん断力は増分せん断力を足し込む方法で求められているのではなく、式を用いて求めているため、その履歴ループは骨格曲線からずれることはない。ただし、状態の変移部分で、適切な誤差修復を行っていないため、剛性が小さい状態から大きい状態に移るとき、大きな増分せん断力が生じさせてしまうことになる。例えば、骨格曲線の第２勾配上から除荷となって弾性復帰する場合である。ここでは、本来、変位が進む場合は第２剛性を用いるが、除荷では第１剛性を使用しなければならず、そのため数値計算で用いる増分ステップ毎に、収束計算を行う必要が生じる。

SPACE では、数値計算の効率を考慮して、上記のような収束計算を使用しておらず、従って、剛性が突然変化する場合には、極端に増分変位もしくは、増分応力が大きくならないような方法を組み込んでいる。ここでも、除荷部分では、適切な誤差修復処理を行う必要があろう。

3.5 Maxwell モデル

3.5.1 Maxwell モデルの概要

本節では、Maxwell モデルについて解説する。この Maxwell モデルを含むと、一般的な振動方程式を少し変形して用いなければならない。Maxwell モデルを含む変形した振動方程式は理論編を参照すると以下のようになる。

$$\begin{aligned}&[M]\{\ddot{y}\}+[\bar{C}]\{\dot{y}\}+[K]\{y\}=-[M][I]\{\ddot{u}_g\}\\&\quad+\{P_s\}-\{\bar{f}_d\}-\{Q(\bar{y})\}-[K_T(\bar{y})]\{\Delta y\}+[K]\{y\}\end{aligned}\qquad\cdots\cdots(3.13)$$

Maxwell モデルに関連する項は、上式中で次の 2 項である。

$$\left.\begin{aligned}&[\bar{C}]=[C]+[C_0]\\&\{\bar{f}_d\}=\{(\alpha_1-\alpha_0)\dot{u}_{n+1}+b\bar{\gamma}f_n+\bar{\alpha}\dot{u}_n+\bar{\alpha}'f_0+b\bar{\alpha}''f_0'\}\end{aligned}\right\}\cdots\cdots(3.14)$$

ここで、$[C_0]$は Maxwell モデルの線形減衰項であり、$[C]$はそれ以外の線形減衰項である。この方程式に *Newmark*β 法を適用すれば、振動方程式は式(3.15)として得られる。

$$\begin{aligned}&[M]\{\ddot{y}_{n+1}\}+[\bar{C}]\{\{a\}+\mu_1\{\ddot{y}_{n+1}\}\}+[K]\{\{\bar{b}\}+\mu_2\{\ddot{y}_{n+1}\}\}\\&\quad=-[M][I]\{\ddot{u}_g\}+\{P_S\}-\{Q(y_n)\}-\{\bar{f}_d\}\\&\quad-[K_T(y_n)]\{\{b\}+\mu_2\{\ddot{y}_{n+1}\}\}+[K]\{\{\bar{b}\}+\mu_2\{\ddot{y}_{n+1}\}\}\end{aligned}\qquad\cdots\cdots(3.15)$$

さらに、上式を整理すると、

$$\begin{aligned}&\left[[M]+\mu_1[\bar{C}]+\mu_2[K]\right]\{\ddot{y}_{n+1}\}\\&\quad=-[M][I]\{\ddot{u}_g\}+\{P_S\}-\{Q(y_n)\}-[\bar{C}]\{a\}-[K_T(y_n)]\{b\}\\&\quad-\{\bar{f}_d\}+\mu_2\left([K]-[K_T(y_n)]\right)\{\ddot{y}_{n+1}\}\end{aligned}\qquad\cdots\cdots(3.16)$$

となる。ここで、係数を

$$\left.\begin{aligned}&[F]=\left[[M]+\mu_1[\bar{C}]+\mu_2[K]\right]\\&\{G(\ddot{y}_{n+1})\}=\mu_2\left([K]-[K_T(y_n)]\right)\{\ddot{y}_{n+1}\}-\{\bar{f}_d\}\\&\{g\}=-[M][I]\{\ddot{u}_g\}+\{P_S\}-\{Q(y_n)\}\\&\qquad-[\bar{C}]\{a\}-[K_T(y_n)]\{b\}\end{aligned}\right\}\cdots\cdots(3.17)$$

のように整理すると、方程式(3.16)は次式となる。

$$[F]\{\ddot{y}_{n+1}\}=\{G(\ddot{y}_{n+1})\}+\{g\}\qquad\cdots\cdots(3.18)$$

式(3.18)から分かるように両辺に未知ベクトルである増分後の加速度ベクトル$\{\ddot{y}_{n+1}\}$を含んでおり、そのため、この方程式は反復法を用いて解かれることになる。

以上を整理すると、Maxwell モデルに関連する項は、線形の減衰項に付加される項$[C_0]$と右辺反復項$\{\bar{f}_d\}$である。節点力と節点変位の関係は、理論編の第 3.4 節中の式(3.26)、(3.43)より次のように得られる。

$$\begin{Bmatrix} p_i \\ p_j \end{Bmatrix} = \begin{bmatrix} \alpha & -\alpha \\ -\alpha & \alpha \end{bmatrix} \begin{Bmatrix} \dot{u}_i \\ \dot{u}_j \end{Bmatrix} + \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} \left(\alpha \dot{u}_n + b\gamma f_n + \alpha' f_0 + b\alpha' f_0' \right) \quad \cdots\cdots (3.19)$$

ここで、各係数は、

$$\alpha = \frac{ck\Delta t}{2c + k\Delta t};\ \alpha' = \frac{k\Delta t}{2c + k\Delta t};\ \gamma = \frac{2c - k\Delta t}{2c + k\Delta t} \qquad \cdots\cdots (3.20)$$

である。また、非線形性を考慮すると下に示すように各種の係数が必要となる。この係数に関する詳細は、理論マニュアルを参照されたい。

$$\left.\begin{aligned} \alpha_0 &= \frac{c_0 k\Delta t}{2c_0 + k\Delta t} & \gamma_0 &= \frac{2c_0 - k\Delta t}{2c_0 + k\Delta t} \\ \alpha_1 &= \frac{c_1 k\Delta t}{2c_1 + k\Delta t} & \gamma_1 &= \frac{2c_1 - k\Delta t}{2c_1 + k\Delta t} \\ \alpha_1' &= \frac{k\Delta t}{2c_1 + k\Delta t} & f_0 &= f_1(1 - c_1 / c_0) \end{aligned}\right\} \cdots (3.21)$$

SPACE では、パッシブ型とセミアクティブ型の制震装置に対応する Maxwell モデルが組み込まれている。しかしながら、セミアクティブ型は複雑なコードを含むため、ここでは説明を省く。これについては、他のマニュアルに譲ることとし、ここではパッシブ型のみ説明する。このパッシブ型においても限界速度があり、この速度を超えると速度に比例するダッシュポットの減衰定数は変化する。このような現象に対応するため、SPACE ではダッシュポットの減衰定数として、バイリニア型が組み込まれている。

Maxwell モデルに関連するサブルーチンは以下に示す 10 である。カッコ内のサブルーチンはこの Maxwell モデルに関連するサブルーチンをコールするサブルーチンである。

１．初期設定：Initset_maxwelldamp	:(Set_model_data)
２．線形剛性：Cal_link_maxwelldamp	:(Cal_stiff_linear)
３．線形減衰：Cal_lin_maxwelldamp	:(Cal_damp_linear)
４．非線形減衰：Cal_nonlin_maxwelldamp （陰解法用）	:(Add_damp3_ld_ex)
５．部材端力の計算：Cal_force_maxwelldamp(反復解法用)	:(Add_fdd_ld)

```
                    Cal_forcef_maxwelldamp（陰解法用）  ：(Add_fdd_ld_ex)
６．応力計算：Stress_maxwelldamp                        ：(Cal_stress)
７．履歴チェック：Check_Maxwell_stress                  ：(submain_dynamic_a)
                  Check_maxwelldamp                     ：(Check_Maxwell_stress)
                  cal_maxwelldamp                       ：(Check_maxwelldamp)
```

3.5.2 初期設定

Maxwell モデルでは、先に示したように入力したデータから多くの係数を計算する必要がある。これを初期設定プログラムで行う。初期設定プログラム Set_initial_data()は、submain_dynamic_a()でサブルーチンコールされる。

```
c――――――――――――――――――――――――――――――★モデルの初期設定
      call Set_initial_data(Element,Member,Parameter_C,Newmark_P,
     *           E_model6_real,Model_type)
```

この初期設定サブプログラムは、多くの部材モデルに対応するように、階層構造となっているが、現在では、Maxwell モデルの初期設定のみ行い、他のモデルの初期設定は行っていない。他のモデルは線形の剛性を計算するサブルーチンで行っている。

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Set_initial_data
C       ――――――――――――――――――――――――――――――――――
C     ●   各モデルの初期設定(各部材モデルで初期設定が必要な場合はここを使用)
C     ――――――――――――――――――――――――――――――――――
      subroutine Set_initial_data()
      do i=1,n_member
      iet  = Member(i).element_type
      goto(11,12,13,14,15,16,17,18,19,20),iet
        ・
        ・
   16 continue
c――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル
      call Initset_maxwelldamp(Element(ie),Newmark_P,
     *           E_model6_real(ien),Model_type.n_m_filter)
        ・
        ・
      end do
      return
      end
```

上記の Maxwell モデルに関する初期設定用サブルーチンを以下に示す。ここでは、Maxwell モデルの基本的な係数を計算し、その値を構造体にセットする。また、計算過程で必要となるワーク領域もゼロクリアする。

```
C     ―――――――――――――――――――――――――――――――――――――
C     ●   Maxwell model に関するサブルーチン群
C       ―――――――――――――――――――――――――――――――――――――
C     ●   制震用３次元モデル（パッシブ型、セミアクティブ型）
C     ―――――――――――――――――――――――――――――――――――――
c    要素数（モデル No.6  Maxwell モデルに対する構造体）
c      structure / element_s6/
c      integer  element_type    !  要素タイプ(6)
c      integer  n_element       !  非線形要素番号
c      real*8   damp_type       !  0.:パッシブ型、1.:セミアクティブ型
c      real*8   AK              !  剛性
c      real*8   C0              !  第一減衰定数
c      real*8   C1              !  第二減衰定数
c      real*8   C2              !  セミアクティブ用減衰定数
c      real*8   F0              !  リリース応力
c      real*8   Udmax           !  リリース速度
c      real*8   F0x             !  セミアクティブ型で、減衰定数変更応力
c      real*8   Dm1             !  ダミー
c      real*8   Dm2             !  ダミー
c      integer  nm_damp         !  部材減衰の有無(1)
c      integer  nm_type         !   (maxwell モデルでは、１：x 方向２：y 方向３：z 方向)
c      end structure
c      record /element_s6/ Element
c      end structure
c
C     ―――――――――――――――――――――――――――――――――――――
C     ●   制震用３次元モデル（モデル No.6  Maxwell モデルに対する構造体）
C     ―――――――――――――――――――――――――――――――――――――
c
c      部材非線形データ
c
c      structure / e_model6_real_s/
c      real*8  alph_0                ! α0
c      real*8  gumma_0               ! γ0
c      real*8  alph_d0               ! α_d0
c      real*8  alph_1                ! α1
c      real*8  gumma_1               ! γ1
c      real*8  alph_d1               ! α_d1
c      real*8  alph_2                ! α2
c      real*8  gumma_2               ! γ2
c      real*8  alph_d2               ! α_d2
c      real*8  alph_3                ! α3
c      real*8  gumma_3               ! γ3
c      real*8  alph_d3               ! α_d3
c      real*8  f00                   ! f0
c      real*8  alph                  ! α
c      real*8  gumma                 ! γ
c      real*8  f01                   ! f0'
c      real*8  f0                    ! f0
c      real*8  alfd                  ! α'
c      real*8  cx                    ! 現在の減衰定数 c
c      real*8  fn                    ! 現在の応力
c      real*8  u1                    ! ダッシュポットの変位
c      real*8  u2                    ! ばねの変位
```

```
c      real*8  u1d                  ! ダッシュポットの速度
c      real*8  uudd                 ! モデルの現在の速度
c      real*8  uud                  ! フィルター通過後のダッシュポット速度
c      real*8  uudx                 ! フィルター通過後のΔt 前のダッシュポット速度
c      integer  jact                ! 0:パッシブ型、1:セミアクティブ型
c      integer  istat               ! 現在の状態
c      real*8   cx_lag              ! Δｔの減衰定数の変化量
c      integer  max_istat_lag       ! 減衰定数の変化時の最大個数
c      integer  istat_lag           ! 減衰定数の変化時の個数
c      real*8   work(41)            ! フィルター用ワークエリア
c      end structure
c     record / e_model6_real_s / E_model6_real
c     ALLOCATABLE :: E_model1_real (:)
c     ALLOCATE (E_model1_real (n_e_model1))
c
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Initset_maxwelldamp
C       ―――――――――――――――――――――――――――――――――
C     ●   Maxwell model の線形減衰
C     ―――――――――――――――――――――――――――――――――
      subroutine Initset_maxwelldamp(Element,Newmark_P,
     *           E_model6_real,m_filter)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / newmark_s     / Newmark_P
      record / element_s6    / Element
      record / e_model6_real_s / E_model6_real
      dt=Newmark_P.dt                                          ! 1
      b = 2.*Element.C0 + Element.AK*dt
      E_model6_real.alph_0 =  Element.C0*Element.AK*dt/b       !α0
      E_model6_real.gumma_0 = (2.*Element.C0-Element.AK*dt)/b  !γ0
      E_model6_real.alph_d0 = 0.0
      b = 2.*Element.C1 + Element.AK*dt
      E_model6_real.alph_1 = Element.C1*Element.AK*dt/b        !α1
      E_model6_real.gumma_1 = (2.*Element.C1-Element.AK*dt)/b  !γ1
      E_model6_real.alph_d1 = Element.AK*dt/b                  !α'1
      b = 2.*Element.C2 + Element.AK*dt
      E_model6_real.alph_2 = Element.C2*Element.AK*dt/b        !α2
      E_model6_real.gumma_2 = (2.*Element.C2-Element.AK*dt)/b  !γ2
      E_model6_real.alph_d2 = 0.0
      E_model6_real.f00 = Element.F0*(1.-Element.C1/Element.C0)  !f0
      if(Element.Time_lag.ne.0.) then                          ! 2
        E_model6_real.max_istat_lag=Element.Time_lag/dt
        if(E_model6_real.max_istat_lag.eq.0) then              ! 3
          E_model6_real.cx_lag=0.
        else                                                   ! 4
          E_model6_real.cx_lag =
     *    (Element.C0-Element.C2)/E_model6_real.max_istat_lag
        endif
      else                                                     ! 5
        E_model6_real.max_istat_lag=0
        E_model6_real.cx_lag=0.
      endif
      E_model6_real.jact  = Element.damp_type +0.5             ! 6
```

```
      E_model6_real.istat = 0
      E_model6_real.gumma = E_model6_real.gumma_0
      E_model6_real.alph =  E_model6_real.alph_0
      E_model6_real.f01 =   0.
      E_model6_real.f0 =    0.
      E_model6_real.alfd =  0.
      E_model6_real.cx =    Element.C0
      write(76,'(a/a)') ' Maxwell member ',                                  ! 7
     *'       c         alph      gumma      alph_d      f0    f00'
      write(76,'(a,10f10.3)') ' c0',Element.C0,E_model6_real.alph_0,
     *   E_model6_real.gumma_0,E_model6_real.alph_d0
      write(76,'(a,10f10.3)') ' c1',Element.C1,E_model6_real.alph_1,
     *   E_model6_real.gumma_1,E_model6_real.alph_d1,Element.F0,
     *   E_model6_real.f00
      write(76,'(a,10f10.3)') ' c2',Element.C2,E_model6_real.alph_2,
     *   E_model6_real.gumma_2,E_model6_real.alph_d2
      write(76,'(a,i4,10e12.4)') ' lag',E_model6_real.max_istat_lag,
     *    Element.Time_lag,E_model6_real.cx_lag,dt
      E_model6_real.fn=0.                                                    ! 8
      E_model6_real.u2=0.          ! ばねの変位
      E_model6_real.u1=0.          ! ダンパーの変位
      E_model6_real.u1d=0.         ! ダンパーの速度
      E_model6_real.uud=0.         ! フィルター通過後のダッシュポット速度
      E_model6_real.uudx=0.
      if(m_filter .eq.1 .and. Element.Filter .ne. 0. ) then                  ! 9
        icon=1
        call IIRDC(ICON,uudd,E_model6_real.uud,E_model6_real.WORK(1))
      endif
      E_model6_real.uudd =0.
      return
      end
```

1．Maxwell モデルで必要となる多くの係数を計算する。計算式横のコメント及び構造体のコメントと計算式(3.21)を比較して確かめられたい。

2．セミアクティブ型を使用する場合、ダンパーのオリフィスを開放することで、減衰定数が大きく変えることができるが、ユーザーの設定で、その時間に少しの遅れを生じさせるかどうかチェックする。もし遅れを生じさせる場合は、その増分ステップ数を構造体成分 E_model6_real.max_istat_lag にセットする。生じさせない場合は、その構造体にゼロをセットする(5)。さらに、遅れを生じさせる場合で、しかも、計算した増分ステップ数がゼロとなる場合は、Δｔ間の減衰定数の変化量をゼロとし(3)、ゼロでない場合は、その値を計算し、構造体 E_model6_real.cx_lag にセットする(4)。

6. さらに、各種の係数、パラメータの初期設定を行う。

7. Maxwell モデルの係数やパラメータなど、設定した値をワークファイルに出力する。
8. ワーク用のデータ領域をゼロクリアする。
9. モデルがセミアクティブ型で、しかもフィルターを使用する場合、フィルターに関する初期設定をサブルーチン IIRDC()を用いて行う。

3.5.3 線形剛性と線形減衰

このモデルの線形剛性行列はゼロ行列である。部材モデルの線形剛性を他の部材モデルと同様に配列にセットする必要がある。そのため、次に示す線形の剛性行列を作成するサブルーチンの中で、このゼロ行列を作成するサブルーチンがコールされる。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stiff_linear
C       ―――――――――――――――――――――――――――――――――
C     ●   線形剛性行列の計算(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_stiff_linear()
      do i=1,n_member
      iet  = Member(i).element_type
      goto(11,12,13,14,15,16,17,18,19,20),iet
        ・
        ・
   16 continue
c―――――――――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル
      call Cal_link_maxwelldamp(ak_linear(1,1,i))
      goto 100
        ・
        ・
     enddo
```

線形の剛性行列は、サブルーチン Cal_link_maxwelldamp()で行われ、その内容を以下に示す。プログラムは非常に単純であり、線形減衰行列である av をゼロクリアして戻す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_link_maxwelldamp
C       ―――――――――――――――――――――――――――――――――
C     ●   Maxwell model の線形剛性
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_link_maxwelldamp(Av)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      dimension av(12,12)
```

```
      do i=1,12
      do j=1,12
      av(j,i)=0.0
      enddo
      enddo
      return
      end
```

次に、このモデルの部材型の線形減衰行列は、以下のように線形の減衰行列を作成するサブルーチンの中でコールされる。部材モデルで減衰を有している型は、現在の SPACE では、この Maxwell 型のみである。したがって、このサブルーチンでは、この Maxwell 型の部材モデルにのみサブルーチンコールが記述されている。

```
C     ――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_damp_linear
C       ――――――――――――――――――――――――――――――――
C     ●   線形減衰行列の計算(ok)
C     ――――――――――――――――――――――――――――――――
      subroutine Cal_damp_linear()
      do i=1,n_member
      iet  = Member(i).element_type
c――――――――――――――――――――――――――――――――★部材減衰を持っているか
      if( Element(ie).nm_damp .ne. 0) then
      goto(11,12,13,14,15,16,17,18,19,20),iet
        •
        •
c――――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル
      ienn=Member(i).nm_damp
      ii=Element(ie).nm_type          !  (maxwell モデルでは、１：x 方向２：y 方向３：z 方向)
       call Cal_lin_maxwelldamp(E_model6_real(ien),
     *                          ac_linear(1,1,ienn),ii)
      goto 100
        •
        •
  100 continue
      endif
      end do
      return
      end
```

上記の線形の減衰行列を計算するサブルーチンを以下に示す。

```
C     ――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_lin_maxwelldamp
C       ――――――――――――――――――――――――――――――――
C     ●   Maxwell model の線形減衰
C     ――――――――――――――――――――――――――――――――
      subroutine Cal_lin_maxwelldamp(E_model6_real,Av,ii)
      implicit real*8(A-H,O-Z)
      include "submain.h"
```

```
      record / e_model6_real_s / E_model6_real
      dimension av(12,12)
      if(ii.le.0.and.ii.gt.3) return                                    ! 1
      do i=1,12                                                         ! 2
      do j=1,12
      av(j,i)=0.0
      enddo
      enddo
      av(ii,ii)=     E_model6_real.alph_0                               ! 3
      av(ii,ii+6)=  -E_model6_real.alph_0
      av(ii+6,ii)=  -E_model6_real.alph_0
      av(ii+6,ii+6)= E_model6_real.alph_0
      return
      end
```

1. 制震ダンパーの配置方向を示す変数iiが1から３以外はデータの設定エラーであり、このサブルーチンから戻る。
2. 減衰行列をゼロクリアする。
3. 構造体 E_model6_real.alph_0 より減衰定数α_0を取得し、制震ダンパーの配置方向を示す変数にしたがって、この値を減衰行列にセットする。

3.5.4 非線形減衰

Maxwell モデルの部材型非線形減衰行列を作成するサブルーチンはCal_nonlin_maxwelldamp()であり、この非線形減衰行列は陰解法で使用される。また、このサブルーチンをコールする 2 つのサブルーチンは、Add_damp3_ld_ex ()と Build_sky_c_ex()である。陰解法の部材型減衰行列をスカイライン行列に組み込むサブルーチン Build_sky_c_ex()について以下に示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Build_sky_c_ex
C       ―――――――――――――――――――――――――――――――
C     ●    部材非線形減衰行列のスカイライン行列への組み込み(ok)
C     ―――――――――――――――――――――――――――――――
      subroutine Build_sky_c_ex(gskym,Member,Element,n_member,rot_memb,
     *                  ac_member ,Newmark_P, max_h_sky,E_model6_real)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      dimension ac_member(12,12,*),gskym(*)
      dimension max_h_sky(0:*),ac_nonlinear(12,12),bk(12,12)
      dimension rot_memb(3,3,2,*)
      record / newmark_s  / Newmark_P
      record / member_s   / Member
```

```
      record / element_s  / Element
      record /E_model6_real_s / E_model6_real
      dimension Member(*),Element(*)
      dimension E_model6_real(*)
c-----------------------------------------------------------------
c     n_istep                   :integer   第一ステップか第二ステップか
c     Member                    :structure
c     n_member                  :integer   部材数
c     Ac_member(12,12,nc_member)  :real*8   部材減衰行列（釣合系）
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-----------------------------------------------------------------
      par=Newmark_P.ddt
      do i=1,n_member
      ij=Member(i).nm_damp
      if(ij.ne.0) then                                              ! 1
      if(Member(i).element_type.eq.6) then                          ! 2
c-------------------------------★Model_No.6 3 次元制震 Maxwell モデル
      ien= Member(i).n_model_type
      ie = Member(i).nm_element
      ii=Element(ie).nm_type
      call Cal_nonlin_maxwelldamp(E_model6_real(ien),               ! 3
     *                        bk,ii)
      call Rotate_K(bk,rot_memb(1,1,1,i),                           ! 4
     *             rot_memb(1,1,2,i),ac_nonlinear)
      call Build_ak(gskym,                                          ! 5
     *     Member(i).irest(1), max_h_sky,
     *     par,ac_nonlinear )
      else                                                          ! 6
c-------------------------------★通常の線形減衰
      call Build_ak(gskym,
     *     Member(i).irest(1), max_h_sky,
     *     par,ac_member(1,1,ij) )
      endif
      end if
      end do
      return
      end
```

1. この部材が減衰項を含むかどうかチェックし、減衰項を含む場合は以下の処理を行う。
2. 部材モデルが Maxwell 型である場合は以下の処理を行い、そうでない場合は、6.に進む。
3. 非線形減衰行列 bk を、サブルーチン Cal_nonlin_maxwelldamp()を用いて求める。
4. 求めた非線形減衰行列 bk を、部材座標系から釣合座標系に変換し、配列 ac_nonlinear にセットする。
5. 非線形減衰行列 ac_nonlinear を、サブルーチン Build_ak()を用いてスカイライン行列 gskym に組み込む。

6. 部材モデルが Maxwell 型でない場合は、線形の減衰行列 ac_member をサブルーチン Build_ak()を用いて、スカイライン行列 gskym に組み込む。

次に、サブルーチン Add_damp3_ld_ex()では、部材非線形減衰行列を求め、速度と掛け算して、その結果を右辺項に足しこむ。このサブルーチンを見てみよう。

```
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)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / newmark_s    /Newmark_P
      record / member_s     /Member
      record / element_s    / Element
      record /E_model6_real_s / E_model6_real
      dimension Member(*),Element(*)
      real*8 ld_point(*),ac_nonlinear(12,12),bk(12,12)
      dimension rot_memb(3,3,2,*),ac_member(12,12,*)
      dimension u(12)
      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                                          ! 1
      do j=1,12
      irest = Member(i).irest(j)                                  ! 2
      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                          ! 3
      ien= Member(i).n_model_type
      ie = Member(i).nm_element
      ii=Element(ie).nm_type
      call Cal_nonlin_maxwelldamp(E_model6_real(ien),               ! 4
     *                        bk,ii)
      call Rotate_K(bk,rot_memb(1,1,1,i),                           ! 5
     *             rot_memb(1,1,2,i),ac_nonlinear)
c-------------------------------★
      else                                                          ! 6
      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)                                      ! 7
      if(irest.gt.0) then
      sum=0.
      do k=1,12
      sum=sum+ac_nonlinear(j,k)*u(k)                                ! 8
      enddo
      ld_point(irest)=ld_point(irest) -  sum                        ! 9
      endif
      end do
      endif
      enddo
c-----------------------------------------------------
      return
      end
```

1. この部材が減衰項を含むかどうかチェックし、減衰項を含む場合は以下の処理を行う。
2. 次に示す $\{a\}$ から、部材両端の速度ベクトル u を取り出す。

$$\{a\} = \{\dot{y}_n\} + \Delta t(1-\delta)\{\ddot{y}_n\}$$

3. 部材モデルが Maxwell 型である場合は以下の処理を行い、そうでない場合は、６に進む。
4. 非線形減衰行列 bk を、サブルーチン Cal_nonlin_maxwelldamp()を用いて求める。
5. 求めた非線形減衰行列 bk を、部材座標系から釣合座標系に変換し、配列 ac_nonlinear にセットする。
6. 部材モデルが Maxwell 型でない場合は、線形の減衰行列 ac_member を配列 ac_nonlinear にコピーする。

7. 部材の自由度を取り出し、拘束でない場合は以下の処理を行う。
8. 釣合座標系の減衰行列と、ベクトル$\{a\}$から部材両端の速度ベクトルuを取り出し、そのベクトルと掛け算を行う。
9. 掛け算の結果をベクトルld_pointに足しこむ。

次に、非線形減衰行列を計算するサブルーチンを以下に示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_nonlin_maxwelldamp
C       ―――――――――――――――――――――――――――――――――
C     ●   Maxwell model の非線形減衰
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_nonlin_maxwelldamp(E_model6_real,Av,ii)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / e_model6_real_s / E_model6_real
      dimension av(12,12)
      if(ii.le.0.and.ii.gt.3) return                              ! 1
      do i=1,12                                                   ! 2
      do j=1,12
      av(j,i)=0.0
      enddo
      enddo
      av(ii,ii)=     E_model6_real.alph                           ! 3
      av(ii,ii+6)=  -E_model6_real.alph
      av(ii+6,ii)=  -E_model6_real.alph
      av(ii+6,ii+6)= E_model6_real.alph
      return
      end
```

1. 制震ダンパーの配置方向を示す変数iiが1から３以外はデータの設定エラーであり、このサブルーチンから戻る。
2. 減衰行列をゼロクリアする。
3. 構造体E_model6_real.alphより非線形減衰定数αを取得し、制震ダンパーの配置方向を示す変数にしたがって、この値を減衰行列にセットする。この非線形減衰定数αは、Maxwell モデルの弾塑性チェックでセットされる。

3.5.5 右辺項(fd)の計算

本節では、振動方程式の右辺項で、Maxwell モデルが関連する部材減衰$\{\bar{f}_d\}$について説明する。この$\{\bar{f}_d\}$を計算するサブルーチンは、線形解析、反復解法と陰解法では異なり、両者について述べる。前者２つと後者の解析で使用する２つのサブルーチンでは、submain_dynamic_a()で

コールされる。

```
c―――――――――――――――――――――――――――――――★Maxwell 型モデルの計算(ok)
      call Add_fdd_ld(ld_point,E_model6_real,Element,
     *       Member,n_member,est_vel_point,rot_memb)
c―――――――――――――――――――――――――――――――★Maxwell 型モデルの右辺項 fd の計算(ok)
      call Add_fdd_ld_ex(ld_point,E_model6_real,Element,
     *       Member,n_member,est_vel_point,rot_memb)
```

まず、線形解析と反復解法で使用されるサブルーチン Add_fdd_ld()
を示す。

```
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()
      do i=1,n_member
      iet  = Member(i).element_type
      ie = Member(i).nm_element
      if( Element(ie).nm_damp .ne. 0) then                              ! 1
      do  j=1,12
      irest = Member(i).irest(j)
      if(irest.ne.0) then
      ud(j)=est_vel_point(irest)                                        ! 2
      else
      ud(j)=0.
      endif
      enddo
      call RotateL_v(1,ud,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vpp)      ! 3
c―――――――――――――――――――――――――――――――★部材減衰を持っているか

      goto(11,12,13,14,15,16,17,18,19,20),iet                           ! 4
        •
        •
  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)        ! 5
      goto 100
        •
        •
  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)      ! 6
      do  j=1,12
      irest = Member(i).irest(j)
      if(irest.ne.0) then
      ld_point_repeat(irest)=ld_point_repeat(irest) - vpp(j)            ! 7
      end if
      end do
```

```
      endif
      end do
      return
      end
```

1. この部材が減衰項を含むかどうかチェックし、減衰項を含む場合は以下の処理を行う。
2. 全体速度ベクトルから、部材両端の節点速度ベクトルを取り出す。
3. 部材両端の節点速度ベクトルを釣合座標系から部材座標系に変換する。
4. 部材モデル番号にしたがって処理を分類する。
5. Maxwell モデルに関連する $\{\bar{f}_d\}$ を、Cal_force_maxwelldamp()を用いて求める。
6. 求めた Maxwell モデルに関連する $\{\bar{f}_d\}$ を、部材座標系から釣合座標系に変換する。
7. 釣合式の右辺項 ld_point_repeat に足しこむ。

上記のサブルーチン Cal_force_maxwelldamp()を以下に示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_force_maxwelldamp
C       ―――――――――――――――――――――――――――――――
C     ●   Maxwell model の反復解法用節点力
C     ―――――――――――――――――――――――――――――――
      subroutine Cal_force_maxwelldamp(ud,E_model6_real,Av,ii,im)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / e_model6_real_s / E_model6_real
      dimension av(12),ud(12)
      if(ii.le.0.and.ii.gt.3) return                                    ! 1
      do j=1,12                                                         ! 2
      av(j)=0.0
      enddo
      udn1=ud(ii+6)-ud(ii)                  !増分後の速度               ! 3
      udn =E_model6_real.uudd               !増分前の速度               ! 4
      fd=(E_model6_real.alph - E_model6_real.alph_0)*udn1 +             ! 5
     *   E_model6_real.gumma*E_model6_real.fn +
     *   E_model6_real.alph*udn  +
     *   2.*E_model6_real.alfd*E_model6_real.f0
      av(ii)=     -fd                                                   ! 6
      av(ii+6)=    fd
      return
      end
```

1. 制震ダンパーの配置方向を示す変数iiが1から３以外はデータの設定エラーであり、このサブルーチンから戻る。
2. 減衰に関連する応力ベクトル$\{\bar{f}_d\}$をゼロクリアする。
3. 制震ダンパーの配置方向にしたがって、増分後の速度を計算する。
4. 増分前の速度を構造体より取得する。この値は弾塑性チェックを行うサブルーチンで設定する。
5. 応力を計算する。ここでは線形解析時と反復解法時の応力は、理論式では異なるが、線形解析における初期設定でE_model6_real.alphにE_model6_real.alph_0をセットしているため、線形解析においても正確に対応する。
6. 減衰に関連する応力ベクトル$\{\bar{f}_d\}$に求めた応力をセットする。

次に、陰解法で使用されるサブルーチンAdd_fdd_ld_ex()を示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Add_fdd_ld_ex
C       ―――――――――――――――――――――――――――――――――
C     ●   Maxwell 減衰項に関するベクトルを加える。(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Add_fdd_ld_ex(ld_point_repeat,E_model6_real,Element,
     *       Member,n_member,est_vel_point,rot_memb)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s   / Member
      record / e_model6_real_s / E_model6_real
      record /element_s / Element
      dimension Member(*),E_model6_real(*),Element(*)
      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
      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
      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
      call RotateL_v(1,ud,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vpp)
      goto(11,12,13,14,15,16,17,18,19,20),iet
        •
        •
   16 continue
c―――――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル
      ii=Element(ie).nm_type
      call Cal_forcef_maxwelldamp(E_model6_real(ien),av,ii)
      goto 100
        •
        •
  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)
      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
```

上記サブルーチンは、Add_fdd_ld()と内容はほとんど同じであり、異なるのは応力を求めるサブルーチン Cal_forcef_maxwelldamp()のみである。次に、このサブルーチンを示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_forcef_maxwelldamp
C       ―――――――――――――――――――――――――――――――――
C     ●   Maxwell model の陰解法用節点力
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_forcef_maxwelldamp(E_model6_real,Av,ii)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / e_model6_real_s / E_model6_real
      dimension av(12)
      if(ii.le.0.and.ii.gt.3) return                                    ! 1
      do j=1,12                                                         ! 2
      av(j)=0.0
      enddo
      udn =E_model6_real.uudd                !増分前の速度              ! 3
      fd=                                                               ! 4
     *   E_model6_real.gumma*E_model6_real.fn +
     *   E_model6_real.alph*udn +
     *   2.*E_model6_real.alfd*E_model6_real.f0
```

```
      av(ii)=     -fd                                                  ! 5
      av(ii+6)=    fd
      return
      end
```

1. 制震ダンパーの配置方向を示す変数iiが1から３以外はデータの設定エラーであり、このサブルーチンから戻る。
2. 速度ベクトルavをゼロクリアする。
3. 増分前の速度を構造体成分E_model6_real.uuddより取得する。この値は弾塑性チェックを行うサブルーチンで設定される。
4. 陰解法の応力 $\bar{f}_d$ を計算する。
5. 減衰に関連する応力ベクトル $\{\bar{f}_d\}$ に求めた応力 $\bar{f}_d$ をセットする。

3.5.6 Maxwellモデルの応力計算

Maxwell モデルのばねに生じる応力は、submain_dynamic_a()においてコールされるサブルーチン Cal_stress()によって計算される。その部分のコードを以下に示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stress
C       ―――――――――――――――――――――――――――――――
C     ●   部材内応力の計算(ok)
C     ―――――――――――――――――――――――――――――――
      subroutine Cal_stress()
C     ―――――――――――――――――――――――――――――――
      do i=1,n_member
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)=est_ddisp_point(ires)
      else
      v(j)=0.
      vp(j)=0.
      endif
      enddo
c――――――――――――――――――――――――――――――★変位を釣合系から部材座標系に変換
      call RotateL_v(1,v,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vv)
      call RotateL_v(1,vp,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
      goto(11,12,13,14,15,16,17,18,19,20),iet
   11 continue
```

```
      ・
      ・
   16 continue
c―――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル(ok)
      do j=1,12
      ires=Member(i).irest(j)
      if(ires.gt.0) then
      v(j)=past_vel_point(ires)
      else
      v(j)=0.
      endif
      enddo
      call RotateL_v(1,v,rot_memb(1,1,1,i),rot_memb(1,1,2,i),vv)
      ii=Element(ie).nm_type
      call Stress_maxwelldamp(vpp,vv,E_model6_real(ien),
     *             Element(ie),ii,Member(i),Model_type.n_m_filter)
      goto 100
      ・
      ・
  100 continue
      end do
      return
      end
```

上記の Maxwell モデルの応力計算を行うサブルーチンを具体的に示す。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Stress_maxwelldamp
C       ―――――――――――――――――――――――――――――――――――
C     ●   Maxwell model の応力計算
C     ―――――――――――――――――――――――――――――――――――
      subroutine Stress_maxwelldamp(u,ud,E_model6_real,
     *                         Element,ii,Member,m_filter)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / e_model6_real_s / E_model6_real
      record / element_s6    / Element
      record / Member_s      / Member
      dimension u(12),ud(12)
      if(ii.le.0.and.ii.gt.3) return                                    ! 1
      udn1=ud(ii+6)-ud(ii)     ! 速度                                   ! 2
      un  =u(ii+6) - u(ii)     ! 変位                                   ! 3
      fn = E_model6_real.fn    ! 前回の応力                             ! 4
      fn1 = E_model6_real.gumma*fn + E_model6_real.alph*udn1 +          ! 5
     *   E_model6_real.alph*E_model6_real.uudd +
     *   2.*E_model6_real.alfd*E_model6_real.f0
      E_model6_real.fn=fn1                  ! 今回の応力のセット        ! 6
      E_model6_real.u2=fn1/Element.AK       ! ばねの変位
      E_model6_real.u1=un - E_model6_real.u2   ! ダンパーの変位
      E_model6_real.u1d=(fn1-E_model6_real.f0)/E_model6_real.cx     ! ダンパーの速度
C     ―――――――――――――――――――――――――――――――――――
c                  非線形から線形へ戻るときのダンパー速度の飛び越し処理
```

```
C     ―――――――――――――――――――――――――――――――――――
      if(E_model6_real.istat.eq.1.or.E_model6_real.istat.eq.2) then          ! 7
      if(abs(fn1).lt.Element.F0)then
      E_model6_real.u1d=(fn1)/Element.C0     ! ダンパーの速度
      endif
      endif
C     ―――――――――――――――――――――――――――――――――――
c                  データを構造体にセット
C     ―――――――――――――――――――――――――――――――――――
      E_model6_real.uudd= udn1                                               ! 8
      E_model6_real.f01=E_model6_real.f0
      Member.stress(ii)=fn1                    !応力のセット
      E_model6_real.uudx=E_model6_real.uud
C     ―――――――――――――――――――――――――――――――――――
c                  速度にフィルターをかける
C     ―――――――――――――――――――――――――――――――――――
      if(m_filter .eq.1 .and. Element.Filter .ne. 0.  ) then                 ! 9
      uudx=ud(ii+6)-ud(ii)
      icon=2
      call IIRDC(ICON,uudx,E_model6_real.uud,E_model6_real.WORK(1))
      else
      E_model6_real.uud=(ud(ii+6)-ud(ii) )          !フィルター用速度
      endif
      return
      end
```

1. 制震ダンパーの配置方向を示す変数iiが1から３以外はデータの設定エラーであり、このサブルーチンから戻る。
2. Maxwell 部材内に生じる速度を計算する。
3. 部材両端に生じる相対変位を求める。
4. 前ステップの応力 fn を構造体より取得する。
5. 式(5.42)より今回の応力 fn1 を計算する。
6. 応力 fn1 を構造体にセットする。また、ばねの変位、ダンパーの変位、ダンパーの速度を計算し、該当する構造体にセットする。
7. ダンパーの状態を表す E_model6_real.istat が１か２で、非線形状態であるとき、しかも、今回の応力値が限界応力より小さくなったとき、線形状態に戻ったとしてダンパーの速度を再計算する。
8. 今回計算した速度などを構造体にセットする。
9. セミアクティブダンパー用で、速度にフィルターを掛ける場合は、サブルーチン IIRDC()をコールする。

3.5.7 Maxwell モデルの非線形チェック

SPACE では、Maxwell モデルの応力と速度の関係がバイリニア型とするパッシブ型の制震装置を組み込んでいる。ここでは、この非線形性をチェックするサブルーチン Ceck_Maxwell_stress()について説明する。このサブルーチンは、submain_dynamic_a()で以下のように呼ばれる。

```
c――――――――――――――――――――――――――――――★Maxwell 型モデルの非線形性チェック(ok)
      call Check_Maxwell_stress(Member,n_member,
     *     Element,E_model6_real,Newmark_P)
```

さらに、このサブルーチンは、現在は、次のように Maxwell モデルの非線形性をチェックするサブルーチンのみコールする。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Check_Maxwell_stress
C       ――――――――――――――――――――――――――――――――――
C     ●   Maxwell model の応力計算
C     ――――――――――――――――――――――――――――――――――――
      subroutine Check_Maxwell_stress(Member,n_member,
     *                           Element,E_model6_real,Newmark_P)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s    / Member
      record / element_s   / Element
      record / E_model6_real_s   / E_model6_real
      record / newmark_s         / Newmark_P
      dimension Member(*),Element(*),E_model6_real(*)
c――――――――――――――――――――――――――――――――――――――――――――――
      do i=1,n_member
c――――――――――――――――――――――――――――――★要素及びモデルのセット
      iet  = Member(i).element_type
      ie   = Member(i).nm_element
      ien  = Member(i).n_model_type
      if(iet.eq.6)then
c――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル(ok)
      call  Check_maxwelldamp(E_model6_real(ien),Element(ie),Newmark_P)
      endif
      end do
      return
      end
```

サブルーチン Check_maxwelldamp()の内容を具体的に示す。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Check_maxwelldamp
C       ――――――――――――――――――――――――――――――――――
C     ●   Maxwell model の応力計算
C     ――――――――――――――――――――――――――――――――――――
      subroutine Check_maxwelldamp(E_model6_real,Element,Newmark_P)
      implicit real*8(A-H,O-Z)
      include "submain.h"
```

```
      record / e_model6_real_s / E_model6_real
      record / element_s6    / Element
      record / newmark_s     / Newmark_P
      udmax = Element.udmax
      istat = E_model6_real.istat                                    ! 1
      ud1= E_model6_real.u1d
      ichange =0                                                     ! 2
      if(E_model6_real.jact .eq. 1) goto 9900 ! セミアクティブ型へ   ! 3
C     ――――――――――――――――――――――――――――――――――――
c               パッシブ型応力チェック
C     ――――――――――――――――――――――――――――――――――――
      goto (101,102,103),istat+1                                     ! 4
 101  continue
      if(ud1 .ge.udmax ) then                                        ! 5
      ichange = 1
      istat = 1
      elseif(ud1 .le. -udmax) then                                   ! 6
      ichange = 1
      istat = 2
      endif
       goto 9800
 102  continue
      if(ud1 .ge.udmax )goto 9800                                    ! 7
      ichange = 1
      istat = 0
      goto 9800
 103  continue
      if(ud1 .le. -udmax) goto 9800                                  ! 8
      ichange = 1
      istat = 0
      goto 9800
C     ――――――――――――――――――――――――――――――――――――
c               セミアクティブ型応力チェック
C     ――――――――――――――――――――――――――――――――――――
 9900  continue                                                     ! 9
      uud=E_model6_real.uud*E_model6_real.uudx
      f0x=Element.f0x
      fn1=E_model6_real.fn
      goto(201,202,203,204,205,206,207,208,209),istat+1
  201 continue
      ・
      ・
C     ――――――――――――――――――――――――――――――――――――
c               データの設定
C     ――――――――――――――――――――――――――――――――――――
 9800 continue
      if(ichange .eq. 0) return                                      ! 10
      E_model6_real.istat= istat
      goto (10,11,12,13,14,15,16,17,18),istat+1
 10   continue
      E_model6_real.alph=E_model6_real.alph_0                        ! 11
      E_model6_real.gumma=E_model6_real.gumma_0
      E_model6_real.f01=E_model6_real.f0
      E_model6_real.f0=0.
```

```
       E_model6_real.alfd=0.
      E_model6_real.cx=Element.C0
      return
 11   continue
      E_model6_real.alph=E_model6_real.alph_1                    ! 12
      E_model6_real.gumma=E_model6_real.gumma_1
      E_model6_real.f01=0.
      E_model6_real.f0=E_model6_real.f00
      E_model6_real.alfd=E_model6_real.alph_d1
      E_model6_real.cx=Element.C1
      return
 12   continue
      E_model6_real.alph=E_model6_real.alph_1                    ! 13
      E_model6_real.gumma=E_model6_real.gumma_1
      E_model6_real.f01=0.
      E_model6_real.f0= -E_model6_real.f00
      E_model6_real.alfd=E_model6_real.alph_d1
      E_model6_real.cx=Element.C1
      return
 13   continue                                                   ! 14
      •
      •
      •
      return
      end
```

プログラムの内容をコードに従って説明する。このサブルーチンはパッシブ型とセミアクティブ型の応力チェックとデータの設定の2つに分かれている。ただし、ここではセミアクティブ型の応力チェックに関するコードは複雑であるため省略した。

1. モデルの状態を構造体 E_model6_real.istat より取得する。また、ダンパーの速度を取得する。
2. モデルの状態変化を知るパラメータ ichange をゼロクリアする。
3. 構造体 E_model6_real.jact が１で、セミアクティブ型である場合、処理を文番号 9900 に制御を移す。構造体 E_model6_real.jact が２でパッシブ型である場合、次の処理を行う。
4. モデルの状態 istat を 3 つに分け、その 3 区間で状態の変化を調査する。状態 istat が 0 の場合は線形状態であり、1 の場合は第 2 勾配区分状態であり、ダンパー速度が udmax より大きい場合に対応する。また、2 の場合も第 2 勾配区分状態であり、ダンパー速度が逆方向で udmax より大きい場合に対応する。
5. ここは線形状態であり、速度のチェックを行う。速度が udmax より大きい場合は、2 つのパラメータ ichange、istat を共に 1 にし、状

態変化があったことを示す。

6. 速度が逆方向に udmax より大きい場合は、2 つのパラメータを ichange=1、istat=2 にセットし、状態変化があったことを示す。
7. ここは状態が１で、速度が udmax より大きい場合は、状態変化なし。小さい場合は、ichange=1、istat=0 に設定する。
8. ここは状態が 2 で、速度が逆方向に udmax より大きい場合は、状態変化なし。小さい場合は、ichange=1、istat=0 に設定する。
9. ここ以降は、セミアクティブ型の制震装置の状態チェックであり、ここでは省略する。
10. 状態変化がない場合は、このサブルーチンから戻る。
11. 線形状態のパラメータを設定する。
12. 状態１のパラメータを設定する。
13. 状態２のパラメータを設定する。
14. これ以降は、セミアクティブ型のパラメータを設定する。以降の説明は省略する。

3.6 塑性論アナロジーモデル

3.6.1 アナロジーモデルの概要

本節では、塑性論アナロジーモデルについて解説する。なお、以後アナロジーモデルと呼ぶ。このアナロジーモデルの理論に関しては理論マニュアルを参照されたい。

アナロジーモデルにおける接線剛性は、ファイバーモデルと同様に求められ、以下の式で与えられる。

$$[K_T]=[K_L]+[K_G]+[K_N] \qquad \cdots\cdots(3.22)$$

最初に、増分弾塑性剛性$[K_L]$は、塑性論を応用した手法を用いて求めている。その結果、剛性行列$[K_L]$は、以下のように求められる。

$$[K_L]=\begin{bmatrix}
k_x-\frac{\alpha^2}{\omega} & 0 & 0 & 0 & -\frac{\alpha\beta}{\omega} & -\frac{\alpha\gamma}{\omega} & -\left(k_x-\frac{\alpha^2}{\omega}\right) & 0 & 0 & 0 & \frac{\alpha\beta}{\omega} & \frac{\alpha\gamma}{\omega} \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
 & & & & k_{\theta y}-\frac{\beta^2}{\omega} & -\frac{\beta\gamma}{\omega} & \frac{\alpha\beta}{\omega} & 0 & 0 & 0 & -\left(k_{\theta y}-\frac{\beta^2}{\omega}\right) & \frac{\beta\gamma}{\omega} \\
 & & & & & k_{\theta z}-\frac{\gamma^2}{\omega} & \frac{\alpha\gamma}{\omega} & 0 & 0 & 0 & \frac{\beta\gamma}{\omega} & -\left(k_{\theta z}-\frac{\gamma^2}{\omega}\right) \\
 & & & & & & k_x-\frac{\alpha^2}{\omega} & 0 & 0 & 0 & -\frac{\alpha\beta}{\omega} & -\frac{\alpha\gamma}{\omega} \\
 & & & & & & & 0 & 0 & 0 & 0 & 0 \\
 & & & & sym & & & & 0 & 0 & 0 & 0 \\
 & & & & & & & & & 0 & 0 & 0 \\
 & & & & & & & & & & k_{\theta y}-\frac{\beta^2}{\omega} & -\frac{\beta\gamma}{\omega} \\
 & & & & & & & & & & & k_{\theta z}-\frac{\gamma^2}{\omega}
\end{bmatrix} \qquad \cdots\cdots(3.23)$$

ここで、$k_x,k_{\theta y},k_{\theta z}$はｘ軸ばねの剛性、ｙ軸とｚ軸に関する曲げばねの剛性を表す。また、パラメータ$\alpha,\beta,\gamma,\omega$については以下に示す。

SPACE では、現在、降伏条件として３種類を用意しているが、ここでは、次の降伏条件について示し、プログラムについてもこの関数について説明する。他の降伏条件に関するプログラムは動的解析編の付録を参照されたい。

$$f=\left(\frac{N}{N_p}\right)^2+\sqrt{\left(\frac{M_y}{M_{yp}}\right)^2+\left(\frac{M_z}{M_{zp}}\right)^2}-1 \qquad \cdots\cdots(3.24)$$

この降伏条件に関する関数の微分関数は、応力を独立変数として、以下のように得られる。以後、降伏条件を表す式を降伏関数と呼ぶ。

$$\left.\begin{aligned} f_n &= \frac{2N}{N_p^2} \\ f_{my} &= \frac{1}{\sqrt{\left(\frac{M_y}{M_{yp}}\right)^2+\left(\frac{M_z}{M_{zp}}\right)^2}}\frac{M_y}{M_{yp}^2} \\ f_{mz} &= \frac{1}{\sqrt{\left(\frac{M_y}{M_{yp}}\right)^2+\left(\frac{M_z}{M_{zp}}\right)^2}}\frac{M_z}{M_{zp}^2} \end{aligned}\right\} \cdots\cdots(3.25)$$

なお、式(3.23)の剛性行列の中で使用しているパラメータは以下のようである。

$$\left.\begin{aligned} &\alpha = k_x f_n; \quad \beta = k_{\theta y} f_{my}; \quad \gamma = k_{\theta z} f_{mz} \\ &\omega = k_x f_n^2 + k_{\theta y} f_{my}^2 + k_{\theta z} f_{mz}^2 \end{aligned}\right\} \cdots\cdots(3.26)$$

上記の降伏関数には、曲げモーメントが 0 の点、つまり正と負の降伏軸力を示す位置の 2 箇所で、接線勾配に不連続点が存在する。この不連続性は収束計算を行うとき、反復時に解が振動して収束しない原因となる。そこで、この 2 箇所の近傍で、降伏関数を式(3.27)、(3.28)に示す関数に変更し、その不連続性を取り除くことにする。以下に示す式の誘導や内容の詳細は、参考文献４を参照されたい。

修正した 2 箇所での降伏関数として、次式のように A 点を中心とする球で表す。また、応力を独立変数とした降伏関数の微分関数も次式のように与えられる。

$$\frac{N}{N_P} < N_B$$

$$f = \frac{1}{2M_B}\left\{\left(\frac{N}{N_P}-a\right)^2+\left(\frac{M_y}{M_{yP}}\right)^2+\left(\frac{M_z}{M_{zP}}\right)^2\right\}-\frac{r_0^2}{2M_B}=0 \quad \cdots\cdots(3.27)$$

$$\left.\begin{aligned} f_n &= \frac{1}{M_B}\left(\frac{N}{N_P}-a\right)\frac{1}{N_P} \\ f_{my} &= \frac{1}{M_B}\left(\frac{M_y}{M_{yP}}\right)\frac{1}{M_{yP}} \\ f_{mz} &= \frac{1}{M_B}\left(\frac{M_z}{M_{zP}}\right)\frac{1}{M_{zP}} \end{aligned}\right\} \cdots\cdots(3.27a)$$

$$\frac{N}{N_P} < -N_B$$

$$f = \frac{1}{2M_B}\left\{(\frac{N}{N_P}+a)^2+(\frac{M_y}{M_{yP}})^2+(\frac{M_z}{M_{zP}})^2\right\}-\frac{r_0^2}{2M_B}=0 \quad \cdots\cdots(3.28)$$

$$\left.\begin{aligned} f_n &= \frac{1}{M_B}(\frac{N}{N_P}+a)\frac{1}{N_P} \\ f_{my} &= \frac{1}{M_B}(\frac{M_y}{M_{yP}})\frac{1}{M_{yP}} \\ f_{mz} &= \frac{1}{M_B}(\frac{M_z}{M_{zP}})\frac{1}{M_{zP}} \end{aligned}\right\} \cdots\cdots(3.28a)$$

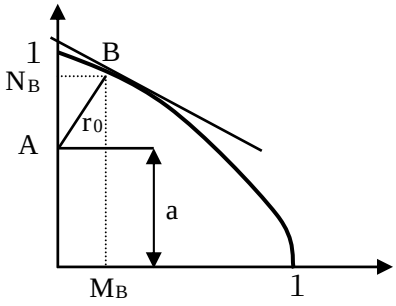

図 3-8 降伏関数の修正図

降伏関数に含まれているパラメータ N_B, M_B, r_0, a は、図 3-8 を参照されたい。同図で、B 点は２つの降伏関数の接続点であり、A 点は、B 点での接線に垂直な線分が N 軸に交差する点である。この点が新たな降伏関数である球の中心点となる。ここで、無次元軸力 N_B を独立変数とすると、後のパラメータは次のように与えられる。

$$\left.\begin{aligned} M_B &= 1-N_B^2 \\ a &= N_B(1-2M_B) \\ r_0 &= M_B\sqrt{1+4N_B^2} \end{aligned}\right\} \cdots\cdots(3.29)$$

SPACE では、現在 N_B を 0.95 としており、式(3.29)を用いると、他のパラメータは $M_B = 0.0975$，$a = 0.76475$，$r_0 = 0.209341$ となり、この値はプログラム内に Data 文として組み込まれている。

次に、アナロジーモデルにおける幾何学的非線形性を考えなければならないが、実はアナロジーモデルでは、エレメントに長さという概念がないので、ここでは幾何学的非線形性を考慮する必要はない。

3.6.2 アナロジーモデルをコールするサブルーチン

これから解説する両端アナロジーモデルが、以下の図で示されている。部材は６分割され、内部には５節点を有する。

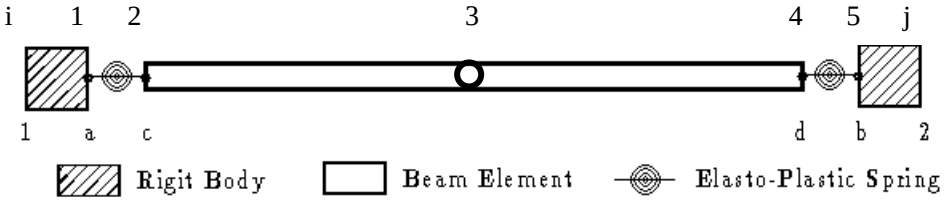

図 3-9 両端アナロジーモデルの部材解析モデル

静的縮合モデルである両端アナロジー部材に関するサブルーチンコールを以下に示す。これらのサブルーチンの中で、上記の接線剛性行列が作成されることになる。剛性を計算する上で、このアナロジーエレメントでは、長さがゼロであることに注意されたい。

```
c――――――――――――――――――――――――――――――――★Model_No.16 両端アナロジーモデル
      call Cal_lin_stiff_M31(Model_type,Member(i),Element(ie),
     *      ak_linear(1,1,i) ,E_model31, E_model_fiber,
     *      M_model31, M_model_fiber)

c――――――――――――――――――――――――――――――――★Model_No.16 両端アナロジーモデル
      call Cal_check_stiff_M31(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model31, E_model_fiber,
     *      M_model31, M_model_fiber,
     *      vv,vpp,f)
c――――――――――――――――――――――――――――――――★Model_No.16 両端、中央アナロジーモデル
      call Cal_nonlin_stiff_M31(N_analysis,
     *      Model_type,Member(i),Element(ie),
     *      ak ,ier,E_model31, E_model_fiber,
     *      M_model31 , M_model_fiber)
```

上記の３つのサブルーチンの内容を以下に示す。これらは、前節で既に説明したファイバーモデルと内容はほぼ同じであり、理解することは難しくない。

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_lin_stiff_M31
C       ――――――――――――――――――――――――――――――――――
C     ●   代表的な部材モデルの剛性(ok)
C     ――――――――――――――――――――――――――――――――――
      subroutine Cal_lin_stiff_M31(Model_type,Member,Element,ak,
     *      E_model31, E_model_fiber,
     *      M_model31, M_model_fiber)
C     ――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / E_model31_s / E_model31
      record / E_model_analogy_s  / E_model_fiber
      record / M_model31_s        / M_model31
      record / M_model_fiber_s    / M_model_fiber
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension E_model31(*),M_model31(*)
      dimension ak(12,12),akk(12,12)
      real*8, ALLOCATABLE :: c(:,:),ab(:,:),ba(:),alength(:)
```

```
      integer, ALLOCATABLE :: irest_Point(:,:),n_type(:)
C     ――――――――――――――――――――――――――――――――――――
      iet = Member.n_model                    ! モデルタイプ番号
      n_div = Model_type.n_div_model(iet)     ! 部材分割数
      imm= Element.n_element                  ! 要素番号
      immm= Member.n_model_type               ! モデルタイプ別番号
      n_if = 6*(n_div-1)                      ! 内部自由度
c       write(76,'(a,4i6)') ' model:',iet,n_div,imm,immm
c―――――――――――――――――――――――――――――――――――――――★動的記憶領域の確保
      ALLOCATE (
     *   irest_Point(6,n_div+1),n_type(n_div),alength(n_div)
     *         )
c―――――――――――――――――――――――――――――――――――――――★節点拘束表の作成
c                                                 未知数等をセット
      call set_model31_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,
     *         n_type,alength,
     *  Member.i_rigid_length,    ! i 端剛域
     *  Member.j_rigid_length,    ! j 端剛域
     *  Member.i_shear_G,         ! i 端せん断剛性
     *  Member.j_shear_G)         ! j 端せん断剛性
c―――――――――――――――――――――――――――――――――――――――★動的記憶領域の確保
      ALLOCATE (
     *   c(0:iubw,n_if),ab(n_if,12),ba(n_if)
     *         )
c―――――――――――――――――――――――――――――――――――――――★剛性行列のゼロクリア
      do i=1,12
      do j=1,12
      ak(j,i)=0.
      enddo
      enddo
      do i=1,n_if
      do j=0,iubw
      c(j,i)=0.
      enddo
      enddo
      do i=1,12
      do j=1,n_if
      ab(j,i)=0.
      enddo
      enddo
      do i=1,n_div
      Member.an_stress(i)=0.
      enddo
      do i=1,n_div
      Member.an_vv(i)=0.
      Member.an_ww(i)=0.
      enddo
c―――――――――――――――――――――――――――――――――――――――★部材剛性行列の作成
      it = 0
      do i=1,n_div
      call Stiff_M31_I(i,it,n_type(i),akk,Member,alength,
     *       Model_type,Element,
     *       E_model31(imm), E_model_fiber,
```

```
     *       M_model31(immm), M_model_fiber)
      call Bnd_FEM(i,akk,irest_Point,ak,c,ab,iubw,n_if)
      enddo
c-------------------------------★部材剛性行列の縮合
      call Typical_member_model(c,ab,ak, n_if,n_if,iubw,iubw,ba,ier)

c-------------------------------★両端の剛域処理
      call Deal_Rigid_element(ak,Member.i_rigid_length,
     *                        Member.j_rigid_length)
c-------------------------------★動的記憶領域の解放
      DEALLOCATE (
     *   c ,ab ,ba,irest_point,n_type,alength
     *         )
      return
      end
```

```
C     ------------------------------------------
C     ●   SUBROUTINE /Cal_nonlin_stiff_M31
C       ----------------------------------------
C     ●   代表的な部材モデルの接線剛性(ok)
C     ------------------------------------------
      subroutine Cal_nonlin_stiff_M31(N_analysis,
     *            Model_type,Member,Element, ak,ier,
     *     E_model31, E_model_fiber, M_model31, M_model_fiber)
C     ------------------------------------------
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / E_model31_s       / E_model31
      record / E_model_fiber_s   / E_model_fiber
      record / M_model31_s       / M_model31
      record / M_model_fiber_s   / M_model_fiber
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension E_model31(*),M_model31(*)
      dimension ak(12,12),akk(12,12)
      real*8, ALLOCATABLE :: c(:,:),ab(:,:),ba(:),alength(:)
      integer, ALLOCATABLE :: irest_Point(:,:),n_type(:)
C     ------------------------------------------
      iet = Member.n_model                    ! モデルタイプ番号
      n_div = Model_type.n_div_model(iet)     ! 部材分割数
      imm= Element.n_element                  ! 要素番号
      immm= Member.n_model_type               ! モデルタイプ別番号
      n_if = 6*(n_div-1)                      ! 内部自由度
c-------------------------------★動的記憶領域の確保
      ALLOCATE (
     *   irest_Point(6,n_div+1),n_type(n_div),alength(n_div)
     *         )
c-------------------------------★節点拘束表の作成
c                                             未知数等をセット
      call set_model31_dat(irest_Point,n_if,n_div,iubw,
     *        Element,Member,
```

```
     *        n_type,alength,
     *  Member.i_rigid_length,    ! i端剛域
     *  Member.j_rigid_length,    ! j端剛域
     *  Member.i_shear_G,         ! i 端せん断剛性
     *  Member.j_shear_G)         ! j 端せん断剛性
c―――――――――――――――――――――――――――――――――★動的記憶領域の確保
      ALLOCATE (
     *   c(0:iubw,n_if),ab(n_if,12),ba(n_if )
     *        )
c―――――――――――――――――――――――――――――――――★剛性行列のゼロクリア
      do i=1,12
      do j=1,12
      ak(j,i)=0.
      enddo
      enddo
      do i=1,n_if
      do j=0,iubw
      c(j,i)=0.
      enddo
      enddo
      do i=1,12
      do j=1,n_if
      ab(j,i)=0.
      enddo
      enddo
c―――――――――――――――――――――――――――――――――★部材剛性行列の作成
      it = 0
      EA=Element.A*Element.E
      do i=1,n_div
      call Stiff_M31(i,it,n_type(i),akk,Member,alength,
     *       Model_type,Element,
     *       E_model31(imm), E_model_fiber,
     *       M_model31(immm), M_model_fiber)
      if(N_analysis.ne.9.and.N_analysis.ne.7.and.n_type(i).eq.1)then       !  幾何学的非線形剛性
      call Cal_geomet_stiffx(Member.an_stress(i),Member,akk,alength(i))
      call Create_Kn(akk,Member.an_vv(i),Member.an_ww(i),
     *              EA,alength(i))
      endif
      call Bnd_FEM(i,akk,irest_Point,ak,c,ab,iubw,n_if)
      enddo
c―――――――――――――――――――――――――――――――――★部材剛性行列の縮合
      call Typical_member_model(c,ab,ak, n_if,n_if,iubw,iubw,ba,ier)
c―――――――――――――――――――――――――――――――――★両端の剛域処理
      call Deal_Rigid_element(ak,Member.i_rigid_length,
     *                        Member.j_rigid_length)
c―――――――――――――――――――――――――――――――――★動的記憶領域の解放
      DEALLOCATE (
     *   c ,ab ,ba,irest_point,n_type,alength
     *        )
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_check_stiff_M31
```

```
C     ―――――――――――――――――――――――――――――――――――――
C     ●    代表的な部材モデルの塑性チェック(ok)
C     ―――――――――――――――――――――――――――――――――――――
      subroutine Cal_check_stiff_M31(N_analysis,
     *      mem_x,Model_type,Member,Element,
     *      E_model31, E_model_fiber,
     *      M_model31, M_model_fiber,
     *      vv,vp,f_p)
C     ―――――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / E_model31_s        / E_model31
      record / E_model_analogy_s  / E_model_fiber
      record / M_model31_s        / M_model31
      record / M_model_fiber_s    / M_model_fiber
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension E_model31(*),M_model31(*)
      dimension ak(12,12),f_p(12)
      dimension vv(12),vp(12),vvx(12)
      real*8, ALLOCATABLE :: c(:,:),ab(:,:),alength(:)
      real*8, ALLOCATABLE :: bav(:),f1(:),akk(:,:,:)
      integer, ALLOCATABLE :: irest_Point(:,:),n_type(:)
C     ―――――――――――――――――――――――――――――――――――――
      iet = Member.n_model                    ! モデルタイプ番号
      n_div = Model_type.n_div_model(iet)     ! 部材分割数
      imm= Element.n_element                  ! 要素番号
      immm= Member.n_model_type               ! モデルタイプ別番号
      n_if = 6*(n_div-1)                      ! 内部自由度
c―――――――――――――――――――――――――――――――――――――――――
c
c     ★        部材の剛性行列の設定
c
c―――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――★動的記憶領域の確保
      ALLOCATE (
     *   irest_Point(6,n_div+1),n_type(n_div),alength(n_div),
     *    akk(12,12,n_div)
     *        )
c――――――――――――――――――――――――――――――――★節点拘束表の作成
c                                                 未知数等をセット
      call set_model31_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,
     *         n_type,alength,
     *  Member.i_rigid_length,    ! i 端剛域
     *  Member.j_rigid_length,    ! j 端剛域
     *  Member.i_shear_G,         ! i 端せん断剛性
     *  Member.j_shear_G)         ! j 端せん断剛性
c――――――――――――――――――――――――――――――――★動的記憶領域の確保
      ALLOCATE (
     *   c(0:iubw,n_if),ab(n_if,12),
```

```
     *   bav(n_if),f1(n_if)
     *         )
    c――――――――――――――――――――――――――――――――――★剛性行列のゼロクリア
          do i=1,12
          do j=1,12
          ak(j,i)=0.
          enddo
          enddo
          do i=1,n_if
          do j=0,iubw
          c(j,i)=0.
          enddo
          enddo
          do i=1,12
          do j=1,n_if
          ab(j,i)=0.
          enddo
          enddo
          do i=1,12
          f_p(i)=0.
          enddo
          do i=1,n_if
          f1(i)=0.
          enddo
    c――――――――――――――――――――――――――――――――――★部材剛性行列の作成
          it = 0
          EA=Element.A*Element.E
          do i=1,n_div
    c――――――――――――――――――――――――――――――――――★内部部材の剛性行列の計算
          call Stiff_M31(i,it,n_type(i),akk(1,1,i),Member,alength,
         *       Model_type,Element,
         *       E_model31(imm), E_model_fiber,
         *       M_model31(immm), M_model_fiber)
          if(N_analysis.ne.9.and.N_analysis.ne.7.and.n_type(i).eq.1)then   !  幾何学的非線形剛性
          call Cal_geomet_stiffx(Member.an_stress(i),Member,
         *                       akk(1,1,i),alength(i))
          call Create_Kn(akk(1,1,i),Member.an_vv(i),Member.an_ww(i),
         *              EA,alength(i))
          endif
    c――――――――――――――――――――――――――――――――――★剛性行列の分配
          call Bnd_FEM(i,akk(1,1,i),irest_Point,ak,c,ab,iubw,n_if)
          enddo
    c――――――――――――――――――――――――――――――――――――――――――――――――
    c
    c    ★         部材内部の変位と不釣合力の計算
    c
    c――――――――――――――――――――――――――――――――――――――――――――――――
    c――――――――――――――――――――――――――――――――――★両端変位を剛域内部の変位に変換処理
          call Deal_Rigid_element_v(vv,Member.i_rigid_length,
         *                          Member.j_rigid_length)
    c――――――――――――――――――――――――――――――――――★部材内部変位の計算(c 行列の分解計算)
          call Typical_member_v(c,ab,f1,
         *                      n_if,n_if,iubw,iubw,ier,vv,bav)
    c――――――――――――――――――――――――――――――――――★部材内部、両端節点力の計算
```

```
      it = 0
      do i=1,n_div
      if(n_type(i).eq.2) then
      it=it+1
      if(it.eq.1)then
      nnm=M_model31(immm).nm_section_1
      else
      nnm=M_model31(immm).nm_section_2
      endif
      endif
      call Analogy_member_p_force(akk(1,1,i),irest_Point(1,i),
     *                            vv,bav,f_p,f1,
     *           n_type(i),M_model31(immm),M_model_fiber,nnm)
      enddo
c―――――――――――――――――――――――――★部材両端節点力への縮合
      call Typical_member_f(c,ab,f1,f_p,n_if,n_if,iubw,iubw)     ! f1 はデータが変更される
c―――――――――――――――――――――――――★部材節点力の両端剛域処理
      call Deal_Rigid_element_f(f_p,Member.i_rigid_length,
     *                           Member.j_rigid_length)
c――――――――――――――――――――――――――――――――――――――――
c
c    ★        部材の弾塑性チェック
c
c――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――★部材内部応力のチェック
      it = 0
      do i=1,n_div
c――――――――――――――――――――――――――――★変位の取りだし
      ii=0
      do j=1,2
      do k=1,6
      ii=ii+1
      irest=irest_Point(k,i+j-1)
      if(irest.lt.0) then
      vvx(ii)=vv(-irest)
      elseif(irest.gt.0) then
      vvx(ii)=bav(irest)
      else
      vvx(ii)=0.
      endif
      enddo
      enddo
c――――――――――――――――――――――――――――★アナロジー要素チェック
      if(n_type(i).eq.2) then
      it=it+1
      call Analogy_check31(N_analysis,
     *      mem_x,it,alength(i),Member,Element,
     *      E_model31(imm),E_model_fiber,M_model31(immm),M_model_fiber,
     *      vvx,Member.an_stress(i))
      endif
c――――――――――――――――――――――――――――★弾性部材の軸力計算（幾何剛性作成用）
      if(n_type(i).eq.1) then
      call nonlinear_stress_N(akk(1,1,i),vvx,fnn)
      Member.an_stress(i)=Member.an_stress(i)+fnn
```

```
c――――――――――――――――――――――――――★部材内部変位を足しこむ
      Member.an_vv(i)=Member.an_vv(i)+(vvx(8) - vvx(2))
      Member.an_ww(i)=Member.an_ww(i)+(vvx(9) - vvx(3))
      endif
      enddo
c――――――――――――――――――――――――――――――――★動的記憶領域の解放
      DEALLOCATE (
     *   c ,ab ,irest_point,n_type,alength,
     *   bav,f1,akk    )
      return
      end
```

この３つのサブルーチンの解析フローは、一部のサブルーチンを除いて、第１章で解説した両端ファイバーモデルと同じである。その中身は、同じサブルーチンを用いている場合と、名称は異なるがその内容はほとんど同じである場合とがある。例えば、解析モデルの未知数番号表を作成するサブルーチン set_model31_dat()は、set_model11_dat()とほとんど同じとなっている。

3.6.3 アナロジーモデルの剛性

本節では、接線剛性行列のひとつである$[K_L]$を求めるサブルーチンStiff_M31()について説明する。まず、このサブルーチンを以下に示す。

```
C     ――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Stiff_M31
C       ――――――――――――――――――――――――――――――――
C     ●   接線剛性行列の計算(ok)
C     ――――――――――――――――――――――――――――――――
      subroutine Stiff_M31(im,it,n_type,ak,Member,alength,
     *       Model_type,Element,
     *     E_model31, E_model_fiber,M_model31, M_model_fiber)
C     ――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s          / Member
      record / element_s         / Element
      record / n_model_s         / Model_type
      record / E_model31_s       / E_model31
      record / M_model31_s       / M_model31
      record / M_model_fiber_s   / M_model_fiber
      record / E_model_fiber_s   / E_model_fiber
      dimension ak(12,12),alength(*)
      dimension vv(12)
      dimension E_model_fiber(*),M_model_fiber(*)
c――――――――――――――――――――――――――――――――★要素及びモデルのセット
      do i=1,12
      do j=1,12
      ak(j,i)=0.
```

```
      enddo
      enddo
      goto(11,12,13,14,15,16,17,18,19,20),n_type
   11 continue
c――――――――――――――――――――――――――――――――★弾性要素
      call Cal_nonlin_stiff_Mx(Member,Element,
     *       Member.stress(7),ak,alength(im) )
      goto 100
   12 continue
c――――――――――――――――――――――――――――――――★アナロジー要素
      it=it+1
      if(it.eq.1)then
      nn=E_model31.nm_section_1
      nnm=M_model31.nm_section_1
      iep=M_model_fiber(nnm).i_elastic
      else
      nn=E_model31.nm_section_2
      nnm=M_model31.nm_section_2
      iep=M_model_fiber(nnm).i_elastic
      endif
      call Analogy_Model_G31(it,iep,ak,Member,Element,
     *      E_model31,E_model_fiber,
     *      M_model31,M_model_fiber,nn)
      goto 100
   13 continue
c――――――――――――――――――――――――――――――――★せん断バネ要素
      call Shear_G(it,ak,Member)
      goto 100
   14 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   15 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   16 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   17 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   18 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   19 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   20 continue
c――――――――――――――――――――――――――――――――★ダミー
  100  continue
      return
      end
```

このサブルーチン Stiff_M31()も、やはりサブルーチン Stiff_M11()

と同様の階層構造を持っており、内容は Stiff_M11()とほとんど同じである。そこではエレメント型番号：２のモデルがアナロジーモデルとなっており、このモデルの剛性行列は、サブルーチン Analogy_Model_G31()を使用して求めている。

サブルーチン Stiff_M31()は、剛性行列を作成するプログラムであり、その構造は非常に単純である。エレメント型番号 n_type をパラメータにして、部材内のエレメントモデルが階層構造となっている。

プログラムは、最初、剛性行列 ak をゼロクリアする。次に、エレメント型番号によって処理が分類されており、現在は１が弾性エレメント、2 がアナロジーエレメント、3 がせん断ばねエレメントとなっている。各エレメント型番号にしたがってサブルーチンがコールされ、剛性行列 ak が計算される。

次に、以下に示す弾塑性剛性行列に関連する３つのサブルーチン Analogy_Model_GI31()と Analogy_Model_G31()、及び Analogy_GK()について説明する。最初のサブルーチンは塑性関数に関する初期設定であり、各種のアナロジーモデルのためのデータを構造体に設定する。次のサブルーチンは弾塑性行列を求めるための各種のデータを、構造体から取得するルーチンである。最後のサブルーチンは実際に剛性行列を計算するプログラムである。以下に、３つのサブルーチンの内容を具体的に示すことにする。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Analogy_Model_GI
C     ―――――――――――――――――――――――――――――――――
C     ●   線形剛性行列の計算(ok)
C     ―――――――――――――――――――――――――――――――――
C     ●   Model_No.31 両端 Analogy モデル
C     ―――――――――――――――――――――――――――――――――
      subroutine Analogy_Model_GI31(it,ak,Member,Element,
     *      E_model,E_model_analogy,M_model,M_model_analogy)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
      record / E_model31_s        / E_model
      record / E_model_analogy_s  / E_model_analogy
      record / M_model31_s        / M_model
      record / M_model_fiber_s    / M_model_analogy
      dimension E_model_analogy(*),M_model_analogy(*)
      dimension ak(12,12)
c―――――――――――――――――――――――――――――――★
      if(it.eq.1) then                                          ! 1
      do i=1,30                                                 ! 2
```

```
       M_model.ff_ip(i)=0.
      enddo
      nm_div=E_model.n_section_1                                        ! 3
      nn=E_model.nm_section_1 - 1
      nnm=M_model.nm_section_1 - 1
      elseif(it.eq.2) then
      nm_div=E_model.n_section_2                                        ! 4
      nn=E_model.nm_section_2 - 1
      nnm=M_model.nm_section_2 - 1
      endif
      do i=1,nm_div                                                     ! 5
      nn=nn+1                                                           ! 6
      nnm=nnm+1
c-----------------------------★データの初期設定                        ! 7
      M_model_analogy(nnm).i_elastic  =0     !  要素の状態（弾性の場合は０：塑性は１）
      M_model_analogy(nnm).d_eps_x =0.       !  変位
      M_model_analogy(nnm).d_stress_x =0.    !  応力
      M_model_analogy(nnm).d_E = 0           ! 降伏関数の値をセット
c-----------------------★データセット
      nmx=M_model_analogy(nnm).n_type
      nm_type=E_model_analogy(nn).nm_type-10
      goto ( 10,20,30),nm_type                                          ! 8
  10  continue
c-----------------------★
      goto 100
  20  continue
c-----------------------★
      goto 100
  30  continue
c-----------------------★
      goto 100
  100 continue
      enddo
      if(it.eq.1) then                                                  ! 9
      M_model.fax_1 = 0.
      M_model.fay_1 = 0.
      M_model.faz_1 = 0.
      nnm=E_model.nm_section_1
      else
      M_model.fax_2 = 0.
      M_model.fay_2 = 0.
      M_model.faz_2 = 0.
      nnm=E_model.nm_section_2
      endif
c-----------------------------★初期剛性行列の計算
      iep=0
      call Analogy_Model_G31(it,iep,ak,Member,Element,                  ! 10
     *      E_model,E_model_analogy,M_model,M_model_analogy,nnm)
      return
      end
```

```
C     ------------------------------------
C     ●   SUBROUTINE /Analogy_Model_G31
C     ------------------------------------
```

```
C     ●   接線剛性行列の計算(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Analogy_Model_G31(it,iep,ak,Member,Element,
     *      E_model,E_model_analogy,M_model,M_model_analogy,itt)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
      record / E_model31_s         / E_model
      record / M_model31_s         / M_model
      record / M_model_fiber_s     / M_model_analogy
      record / E_model_analogy_s   / E_model_analogy
      dimension E_model_analogy(*),M_model_analogy(*)
      dimension ak(12,12)
c―――――――――――――――――――――――――――――――――★
      if(it.eq.1) then                                              ! 11
      ra  = E_model_analogy(itt).AK_1            !  ばねの軸剛性
      ray = E_model_analogy(itt+1).AK_1          !  z 軸のばねの曲げ剛性
      raz = E_model_analogy(itt+2).AK_1          !  y 軸のばねの曲げ剛性
      fax = M_model.fax_1                        ! ポテンシャル関数の x 軸微分
      fay = M_model.fay_1                        ! ポテンシャル関数の y 軸微分
      faz = M_model.faz_1                        ! ポテンシャル関数の z 軸微分
      else
      ra  = E_model_analogy(itt).AK_1            !  ばねの軸剛性
      ray = E_model_analogy(itt+1).AK_1          !  z 軸のばねの曲げ剛性
      raz = E_model_analogy(itt+2).AK_1          !  y 軸のばねの曲げ剛性
      fax = M_model.fax_2                        ! ポテンシャル関数の x 軸微分
      fay = M_model.fay_2                        ! ポテンシャル関数の y 軸微分
      faz = M_model.faz_2                        ! ポテンシャル関数の z 軸微分
      endif
      ra_x= E_model_analogy(itt).AK_2            !  ばねの最低軸剛性
      ray_x=E_model_analogy(itt+1).AK_2          !  ばねの最低曲げ剛性
      raz_x=E_model_analogy(itt+2).AK_2          !  ばねの最低曲げ剛性
      call Analogy_GK(iep,ak,ra,ray,raz,fax,fay,faz,                ! 12
     *               ra_x,ray_x,raz_x)
      return
      end
```

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Analogy_GK
C     ―――――――――――――――――――――――――――――――――
C     ●   アナロジー要素の剛性行列
C     ―――――――――――――――――――――――――――――――――
      subroutine Analogy_GK(id,ak,ra,riy,riz,fax,fay,faz,
     *               ra_x,ray_x,raz_x)
      implicit real*8(A-H,O-Z)
      dimension ak(12,12)
      if(id.eq.0) then                                              ! 13
c―――――――――――――――――――――――――――――――――★線形剛性
      ak(1,1)   = ra
      ak(1,7)   =-ra
      ak(7,1)   =-ra
      ak(7,7)   = ra
```

```
       ak(5,5)   = riy
      ak(5,11)  =-riy
      ak(11,5)  =-riy
      ak(11,11) = riy
      ak(6,6)   =  riz
      ak(6,12)  = -riz
      ak(12,6)  = -riz
      ak(12,12) =  riz
      else                                                    ! 14
      a=ra*fax
      b=riy*fay
      c=riz*faz
      h=a*fax+b*fay+c*faz
      if(h.eq.0.) then                                        ! 15
c―――――――――――――――――――――――――――――――――★線形剛性
      ak(1,1)   = ra
      ak(1,7)   =-ra
      ak(7,1)   =-ra
      ak(7,7)   = ra
      ak(5,5)   = riy
      ak(5,11)  =-riy
      ak(11,5)  =-riy
      ak(11,11) = riy
      ak(6,6)   =  riz
      ak(6,12)  = -riz
      ak(12,6)  = -riz
      ak(12,12) =  riz
      else                                                    ! 16
c―――――――――――――――――――――――――――――――――★非線形剛性
      hh=1./h
      ha=a*a*hh
      hb=b*b*hh
      hc=c*c*hh
      hab=a*b*hh
      hac=a*c*hh
      hbc=b*c*hh
c――――――――――――――――★剛性が任意閾値より小さくなるのを防ぐ処理
      raha  = ra -  ha                                        ! 17
      if(raha .lt.ra_x)   raha=ra_x
      riyhb = riy - hb
      if(riyhb .lt.ray_x) riyhb=ray_x
      rizhc = riz - hc
      if(rizhc .lt.raz_x) rizhc = raz_x
c――――――――――――――――★直線部材で 3 ヒンジ不安定になるのを避けるため
       if(hab.ne.0.) hab=hab*1.00001                          ! 18
       if(hac.ne.0.) hac=hac*1.00001
       if(hbc.ne.0.) hbc=hbc*1.00001
c――――――――――――――――★直線部材で 3 ヒンジ不安定になるのを避けるため
      ak(1,1) =  raha                                         ! 19
      ak(1,7) = -raha
      ak(7,1) = -raha
      ak(7,7) =  raha
      ak(1,5) = -hab
      ak(5,1) = -hab
```

```
      ak(1,11)=  hab
      ak(11,1)=  hab
      ak(5,7) =  hab
      ak(7,5) =  hab
      ak(7,11)= -hab
      ak(11,7)= -hab
      ak(1,12)= hac
      ak(12,1)= hac
      ak(1,6) =-hac
      ak(6,1) =-hac
      ak(6,7) = hac
      ak(7,6) = hac
      ak(7,12)=-hac
      ak(12,7)=-hac
      ak(5,6)  = -hbc
      ak(6,5)  = -hbc
      ak(5,12) =  hbc
      ak(12,5) =  hbc
      ak(6,11) =  hbc
      ak(11,6) =  hbc
      ak(11,12)= -hbc
      ak(12,11)= -hbc

      ak(5,5)   =  riyhb
      ak(5,11)  = -riyhb
      ak(11,5)  = -riyhb
      ak(11,11) =  riyhb
      ak(6,6)   =  rizhc
      ak(6,12)  = -rizhc
      ak(12,6)  = -rizhc
      ak(12,12) =  rizhc
      endif
      endif
      return
      end
```

上記の３つのサブルーチンについて、右側のコメント番号に従ってプログラムの内容を説明する。ここで使用しているアナロジーモデルのばねに関するデータ領域はファイバーモデルのそれと共用しており、そのため同じ構造体を使用している。

1. 両端アナロジーモデルでは、i 端と j 端でアナロジーばねデータの管理位置が異なる。そのため、ここでどちらかをチェックする。パラメータ it が１の場合は i 端を意味し、2 は j 端を意味する。
2. 内部節点の不釣合力の保存場所である M_model.ff_ip をゼロクリアする。
3. 構造体より i 端のアナロジーばねデータの管理用パラメータを取得し、その断面におけるアナロジーばね数を nm_div に、アナロジーの

ばねの要素に関連するデータの最初の配列番号 － 1 を nn に、同じく、部材に関連するデータの最初の配列番号 － 1 を nnm にセットする。

4. 上記と同様に、j 端に関するアナロジーばねデータの管理用パラメータを取得する。
5. 以下の処理を、その断面の全アナロジーばね数分行う。ただし、ばね数は軸方向ばね、y 軸回りの曲げばね、z 軸回りの曲げばねの３つである。
6. 構造体の配列番号に 1 を加え、次のアナロジーばねデータの配列番号をセットする。
7. 入力データ以外の各アナロジーばねデータの初期設定を行う。まず、部材に関連し、解析途中で変更するデータとして、アナロジーばねエレメントの弾塑性状態を表すパラメータ、軸方向ひずみ、軸方向応力、降伏関数の値にゼロをセットする。
８. アナロジーばねの降伏条件の型番号によって処理内容が分類されており、その番号にしたがって各プログラム位置に処理が移動する。ただし、現在は、完全弾塑性型を使用しているのみであるから、ここでは初期設定を行っていない。
９. 降伏関数の微分値を i 端と j 端に分けてゼロセットする。
10. この断面を要するエレメントの接線剛性行列のひとつである行列 $[K_L]$を、サブルーチン Analogy_Model_G31()で求める。
11. サブルーチン Analogy_Model_G31()では、まず、この部材のどちらの端部かをチェックし、該当する端部断面に関する各パラメータを構造体より取得する。
12. この取得したパラメータを用いて、実際の剛性行列 ak をサブルーチン Analogy_GK()を用いて計算する。
13. ここでは、アナロジーばねエレメントの剛性行列を計算する。塑性状態のアナロジーばねエレメントの剛性行列を求めるコードと、式(3.23)の行列とを比較して、その内容を確認されたい。また、この式の誘導などについては理論マニュアルを参照されたい。最初に、エレメントが弾性状態であるか塑性状態であるか、パラメータ id を用いてチェックする。弾性状態である場合は線形の剛性行列を作成する。
14. 塑性状態のアナロジーばねエレメントの剛性行列を計算する。ここでは、降伏関数の微分値が必要となる。
15. 式(3.23)の中で、ωの値がゼロである場合、ゼロ割り算となるので、

この場合は、以降の処理で線形の剛性行列をセットする。

16. 上記のωがゼロでない場合、式(3.26)で示す各係数を計算する。

17. エレメントの剛性行列の対角項がゼロとなるのを防ぐために、行列対角項がある閾値より小さくならないようにする。

18. 直線部材で３ヒンジの不安定部材となると、剛性が特異行列となり、計算不能状態となる。そこで、軸方向剛性と２つの曲げ剛性の関連部分の剛性に非常に小さな値をつけ加え、剛性行列が特異とならないようにする。

19. 最終的に設定した各係数を用いて剛性行列の該当する部分に係数をセットする。

3.6.4 アナロジーモデルの弾塑性チェック

最後に残った処理はアナロジーばねの弾塑性チェックである。この弾塑性チェックを行うサブルーチン Analogy_check31()をコールするコードは、サブルーチン Cal_check_stiff_M31()の中で、以下のように記述されている。

```
c――――――――――――――――――――――――――★アナロジーばねチェック
      if(n_type(i).eq.2) then
      it=it+1
      call Analogy_check31(N_analysis,
     *      mem_x,it,alength(i),Member,Element,
     *      E_model31(imm),E_model_fiber,M_model31(immm),M_model_fiber,
     *      vvx,Member.an_stress(i))
      endif
```

サブルーチン Analogy_check31()の内容を以下に示す。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Analogy_check31
C     ―――――――――――――――――――――――――――――――――――
C     ●   アナロジー要素の材料非線形性チェックし、応力を計算
C     ―――――――――――――――――――――――――――――――――――
      subroutine Analogy_check31(N_analysis,
     *      mem_x,it,Alength,Member,Element,
     *      E_model,E_model_analogy,M_model,M_model_analogy,vv,An_f)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
      record / E_model31_s        / E_model
      record / E_model_analogy_s  / E_model_analogy
      record / M_model31_s        / M_model
```

```
      record / M_model_fiber_s    / M_model_analogy
      dimension E_model_analogy(*),M_model_analogy(*)
      dimension vv(12)
c―――――――――――――――――――――――――――――――★i 端部
c―――――――――――――――――――――――――――――――★変位のセット
      if(it.eq.1) then                                                  ! 1
      d_epsi_x_1 =  (vv(7)  - vv(1))   !  増分軸方向変位                ! 2
      d_epsi_y_1 =  (vv(11) - vv(5))   !  増分ｙ軸に関する回転角
      d_epsi_z_1 =  (vv(12) - vv(6))   !  増分ｚ軸に関する回転角
      if(Member.analysis_3D .eq. 1)  d_epsi_z_1 =0.   ! 平面問題における面外方向の曲げ変形を無視する
      if(Member.analysis_3D .eq. 2)  d_epsi_y_1 =0.   ! 平面問題における面外方向の曲げ変形を無視する
      M_model.d_epsi_x_1 =  d_epsi_x_1+M_model.d_epsi_x_1          !  軸方向変位
      M_model.d_epsi_y_1 =  d_epsi_y_1+M_model.d_epsi_y_1          !  ｙ軸に関する回転角
      M_model.d_epsi_z_1 =  d_epsi_z_1+M_model.d_epsi_z_1          !  ｚ軸に関する回転角
c―――――――――――――――――――――――――――――――★アナロジー要素のチェック
      nm_div=E_model.n_section_1                                        ! 3
      nem=E_model.nm_section_1
      nnm=M_model.nm_section_1
      nmx=M_model_analogy(nnm).n_type
      nm_type=E_model_analogy(nem).nm_type
c―――――――――――――――――――――――――――――――★各要素の増分歪セット
      M_model_analogy(nnm).d_eps_x   =  d_epsi_x_1                      ! 4
      M_model_analogy(nnm+1).d_eps_x =  d_epsi_y_1
      M_model_analogy(nnm+2).d_eps_x =  d_epsi_z_1
      if(N_analysis.le.8.or.
     *   Member.nm_analysis.eq.-1) then                                 ! 5
c―――――――――――――――――――――――――――――――★弾性解析
      else
c―――――――――――――――――――――――――――――――★弾塑性解析
      goto ( 10,20,30),nm_type-10                                       ! 6
  10  continue
c―――――――――――――――――――――――――――――――★　Ｉ型
      call Check_Analogy_M1(E_model_analogy,nem,                        ! 7
     *           M_model_analogy,nnm,Member.d_stat(1),
     *           M_model.fax_1,M_model.fay_1,M_model.faz_1,i_hosei)
c―――――――――――――――――――――――――――――――★塑性ポテンシャルの微分計算
      call Cal_div_potential_M1(E_model_analogy,nem,                    ! 8
     *            M_model_analogy,nnm,
     *            M_model.fax_1,M_model.fay_1,M_model.faz_1,i_hosei)
      goto 100
  20  continue
c―――――――――――――――――――――――――――――――★　ＩＩ型                                  ! 9
      call Check_Analogy_M2(E_model_analogy,nem,
     *           M_model_analogy,nnm,Member.d_stat(1),
     *           M_model.fax_1,M_model.fay_1,M_model.faz_1)
c―――――――――――――――――――――――――――――――★塑性ポテンシャルの微分計算
      call Cal_div_potential_M2(E_model_analogy,nem,
     *            M_model_analogy,nnm,
     *            M_model.fax_1,M_model.fay_1,M_model.faz_1)
      goto 100
  30  continue
c―――――――――――――――――――――――――――――――★　ＩＩＩ型
      call Check_Analogy_M3(E_model_analogy,nem,                        ! 10
     *           M_model_analogy,nnm,Member.d_stat(1),
```

```
     *          M_model.fax_1,M_model.fay_1,M_model.faz_1)
c--------------------★塑性ポテンシャルの微分計算
      call Cal_div_potential_M3(E_model_analogy,nem,
     *           M_model_analogy,nnm,
     *           M_model.fax_1,M_model.fay_1,M_model.faz_1)
      goto 100
      endif
  100 continue
c--------------------------------------★要素応力の計算
c--------------------------------------★j 端部
c--------------------------------------★変位のセット
      elseif(it.eq.2) then                                              ! 11
      d_epsi_x_2 =  (vv(7) - vv(1))    !  軸方向変位
      d_epsi_y_2 =  (vv(11) - vv(5))   !  y軸に関する回転角
      d_epsi_z_2 =  (vv(12) - vv(6))   !  z軸に関する回転角
      if(Member.analysis_3D .eq. 1) d_epsi_z_2 =0.  ! 平面問題における面外方向の曲げ変形を無視する
      if(Member.analysis_3D .eq. 2) d_epsi_y_2 =0.  ! 平面問題における面外方向の曲げ変形を無視する
      M_model.d_epsi_x_2 =  d_epsi_x_2+M_model.d_epsi_x_2            !  軸方向変位
      M_model.d_epsi_y_2 =  d_epsi_y_2+M_model.d_epsi_y_2             !  y軸に関する回転角
      M_model.d_epsi_z_2 =  d_epsi_z_2+M_model.d_epsi_z_2             !  z軸に関する回転角
c--------------------------------------★アナロジー要素のチェック
      nm_div=E_model.n_section_2
      nem=E_model.nm_section_2
      nnm=M_model.nm_section_2
      nmx=M_model_analogy(nnm).n_type
      nm_type=E_model_analogy(nem).nm_type
c-------------------------★各要素の増分歪セット
      M_model_analogy(nnm).d_eps_x   =  d_epsi_x_2
      M_model_analogy(nnm+1).d_eps_x =  d_epsi_y_2
      M_model_analogy(nnm+2).d_eps_x =  d_epsi_z_2
      if(N_analysis.le.8.or.
     *   Member.nm_analysis.eq.-1) then
c-------------------------★弾性解析
      else
c-------------------------★弾塑性解析
      goto ( 11,21,31),nm_type-10
  11  continue
c-------------------------★ I 型
      call Check_Analogy_M1(E_model_analogy,nem,
     *           M_model_analogy,nnm,Member.d_stat(2),
     *           M_model.fax_2,M_model.fay_2,M_model.faz_2,i_hosei)
c--------------------★塑性ポテンシャルの微分計算
      call Cal_div_potential_M1(E_model_analogy,nem,
     *           M_model_analogy,nnm,
     *           M_model.fax_2,M_model.fay_2,M_model.faz_2,i_hosei)
      goto 101
  21  continue
c-------------------------★ II型
      call Check_Analogy_M2(E_model_analogy,nem,
     *           M_model_analogy,nnm,Member.d_stat(2),
     *           M_model.fax_2,M_model.fay_2,M_model.faz_2)
c--------------------★塑性ポテンシャルの微分計算
      call Cal_div_potential_M2(E_model_analogy,nem,
     *           M_model_analogy,nnm,
```

```
     *           M_model.fax_2,M_model.fay_2,M_model.faz_2)
      goto 101
   31 continue
c-----------------------★ III型
      call Check_Analogy_M3(E_model_analogy,nem,
     *          M_model_analogy,nnm,Member.d_stat(2),
     *           M_model.fax_2,M_model.fay_2,M_model.faz_2)
c-----------------★塑性ポテンシャルの微分計算
      call Cal_div_potential_M3(E_model_analogy,nem,
     *           M_model_analogy,nnm,
     *           M_model.fax_2,M_model.fay_2,M_model.faz_2)
      goto 101
      endif
  101 continue
c-------------------------------★要素応力の計算
      endif
      return
      end
```

上記のサブルーチンについて、右側のコメント番号に従ってプログラムの内容を説明する。

1. 両端アナロジーモデルでは、i 端と j 端でアナロジーデータの管理位置が異なる。そのため、ここでどちらであるかを調査する。パラメータ it が 1 の場合は i 端であり、2 は j 端である。
2. この断面の増分ひずみ、増分軸方向ひずみ、ｙ軸に関する曲げひずみ、ｚ軸に関する曲げひずみを求める。次に、解析が平面問題の場合は、面外方向の曲げひずみをゼロとする。最後に、各増分ひずみを増分前のひずみに足しこむ。
3. 断面内のアナロジーばねの数（アナロジーばねの数は３に固定されている)、要素アナロジーばね用データの配列番号を nem に、部材アナロジーばね用データの配列番号を nnm にセットする。
4. アナロジーばねの増分ひずみをセットする。
5. 解析種別と部材の解析種別をチェックし、弾性である場合は何もせずにこのサブルーチンから戻る。
6. 降伏条件の型番号 nm_type によって処理を分類する。ただし、降伏条件の型番号は 11 番以降を用いている。
7. 降伏条件の型番号 11 のＩ型であり、降伏条件についてアナロジーばねの弾塑性チェックをサブルーチン Check_Analogy_M1()を用いて行う。
8. 降伏関数の微分を、サブルーチン Cal_div_potentioal_M1 を用いて、求める。

9. ここでは降伏条件の型番号 12 のⅡ型であり、この降伏条件についてアナロジーばねの弾塑性チェックと降伏関数の微分値を求める。
10. ここでは降伏条件の型番号 13 のⅢ型であり、この降伏条件についてアナロジーばねの弾塑性チェックと降伏関数の微分値を求める。
11. これ以降は j 端における処理を行うが、ここでの処理は上記の i 端の処理と全く同一である。プログラムを一度検証されたい。

3.6.5 降伏関数の微分

最後に、このサブルーチンの中で使用されている降伏関数の微分を求めるサブルーチン Cal_div_potential_M1()と弾塑性チェックを行うサブルーチン Check_Analogy_M1()について示す。このサブルーチンは、降伏条件型番号に従ってその関数の微分を求める。以下にその内容を示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_div_potential_M1
C     ―――――――――――――――――――――――――――――――――
C     ●   アナロジー要素の塑性ポテンシャルの微分計算
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_div_potential_M1(E_model_analogy,nem,
     *           M_model_analogy,nmm,ffx,ffy,ffz,i_hosei)
C     ―――――――――――――――――――――――――――――――――
C     ●   ファイバーモデル E_model_analogy_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
c      structure / E_model_analogy_s/
c      integer  nm_type              !  履歴モデルの番号
c      real*8   AK_1                 !  ばねの第一勾配
c      real*8   AK_2                 !  ばねの第二勾配
c      real*8   AK_3                 !  ばねの第三勾配
c      real*8   Q_1                  !  第一折れ点の応力
c      real*8   Q_2                  !  第二折れ点の応力
c      real*8   dm(16)               !  ダミー
c      end structure
c
C     ―――――――――――――――――――――――――――――――――
C     ●   ファイバーモデル M_model_analogy_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
c      structure / M_model_analogy_s/
c      integer  n_type              !  履歴モデルの通し番号
c      integer  i_elastic           !  ファイバー要素の状態（弾性の場合は０：塑性は１）
c      real*8   d_eps_x             !  軸方向歪
c      real*8   d_stress_x          !  軸方向応力
c      real*8   d_E                 !  接線剛性
c      end structure
```

```
c
      implicit real*8(A-H,O-Z)
      include "submainx.h"
      record / E_model_analogy_s    / E_model_analogy
      record / M_model_fiber_s      / M_model_analogy
      dimension E_model_analogy(*),M_model_analogy(*)
      data x_Nb,x_Mb,x_a,x_r0/0.95,0.0975,0.76475,0.209341/

      if(M_model_analogy(nmm).i_elastic  .eq.0 )then     !（弾性の場合は０：塑性は１）1
      ffx=0.
      ffy=0.
      ffz=0.
      else
      if(i_hosei.eq.1)then                                              ! 2
c――――――――――――――――――――――――★f_N > x_Nb
      ffx=(M_model_analogy(nmm).d_stress_x/E_model_analogy(nem).Q_1-
     *     x_a)/(E_model_analogy(nem).Q_1*x_Mb)
      ffy=M_model_analogy(nmm+1).d_stress_x/
     *       (E_model_analogy(nem+1).Q_1**2*x_Mb)
      ffz=M_model_analogy(nmm+2).d_stress_x/
     *       (E_model_analogy(nem+2).Q_1**2*x_Mb)
      elseif(i_hosei.eq.2)then                                          ! 3
c――――――――――――――――――――――――★f_N < -x_Nb
      ffx=(M_model_analogy(nmm).d_stress_x/E_model_analogy(nem).Q_1+
     *     x_a)/(E_model_analogy(nem).Q_1*x_Mb)
      ffy=M_model_analogy(nmm+1).d_stress_x/
     *       (E_model_analogy(nem+1).Q_1**2*x_Mb)
      ffz=M_model_analogy(nmm+2).d_stress_x/
     *       (E_model_analogy(nem+2).Q_1**2*x_Mb)
      else                                                              ! 4
c――――――――――――――――――――――――★x_Nb > f_N > -x_Nb
      ffx=2.*M_model_analogy(nmm).d_stress_x/E_model_analogy(nem).Q_1**2
      ff=dsqrt(
     *(M_model_analogy(nmm+1).d_stress_x/E_model_analogy(nem+1).Q_1)**2+
     *(M_model_analogy(nmm+2).d_stress_x/E_model_analogy(nem+2).Q_1)**2
     *        )
      if(ff.eq.0.) then                                                 ! 5
      ffy=0.
      ffz=0.
      Else                                                              ! 6
      ffy=M_model_analogy(nmm+1).d_stress_x/
     *       (E_model_analogy(nem+1).Q_1**2*ff)
      ffz=M_model_analogy(nmm+2).d_stress_x/
     *       (E_model_analogy(nem+2).Q_1**2*ff)
      endif
      endif
      endif
      return
      end
```

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Check_Analogy_M1
C     ――――――――――――――――――――――――――――――――――――
C     ●   アナロジー要素の塑性チェック
```

```
C     ―――――――――――――――――――――――――――――――――――
      subroutine Check_Analogy_M1(E_model_analogy,nem,
     *          M_model_analogy,nmm,iet,fax,fay,faz,i_hosei)
      implicit real*8(A-H,O-Z)
      include "submainx.h"
      record / E_model_analogy_s    / E_model_analogy
      record / M_model_fiber_s      / M_model_analogy

      dimension E_model_analogy(*),M_model_analogy(*)
      data x_Nb,x_Mb,x_a,x_r0/0.95,0.0975,0.76475,0.209341/
c――――――――――――――――――――――――――――――――★軸力チェック
      iett=M_model_analogy(nmm).i_elastic                              ! 7
      f_N=M_model_analogy(nmm).d_stress_x/E_model_analogy(nem).Q_1
      ff_past = M_model_analogy(nmm).d_E     ! 降伏関数の値をセット
c――――――――――――――――――――――――――★f_N > x_Nb  引張
      if(f_N.gt.x_Nb) then                                             ! 8
      i_hosei=1
c――――――――――――――――――――――――――★降伏条件式
      ff=((f_N - x_a)**2 +                                             ! 9
     *(M_model_analogy(nmm+1).d_stress_x/E_model_analogy(nem+1).Q_1)**2+
     *(M_model_analogy(nmm+2).d_stress_x/E_model_analogy(nem+2).Q_1)**2-
     * x_r0**2)
      M_model_analogy(nmm).d_E = ff     ! 降伏関数の値をセット         ! 10
      if(iett.eq.0) then                                               ! 11
c――――――――――――――――――――――――――★弾性状態の場合
      if(ff.ge.0.) then                                                ! 12
      M_model_analogy(nmm).i_elastic=2
      write(76,'(a,3i4,8e12.4)')' 塑性発生',
     *M_model_analogy(nmm).i_elastic,nem,nmm,ff,
     *M_model_analogy(nmm).d_stress_x,M_model_analogy(nmm+1).d_stress_x,
     *M_model_analogy(nmm+2).d_stress_x,E_model_analogy(nem).Q_1,
     *E_model_analogy(nem+1).Q_1,E_model_analogy(nem+2).Q_1
      endif
      else
c――――――――――――――――――――――――――★部材が既に塑性の場合
      a=fax*E_model_analogy(nem).AK_1
      b=fay*E_model_analogy(nem+1).AK_1
      c=faz*E_model_analogy(nem+2).AK_1
      df=a*M_model_analogy(nmm).d_eps_x +                               ! 13
     *   b*M_model_analogy(nmm+1).d_eps_x +
     *   c*M_model_analogy(nmm+2).d_eps_x
      ddd=a*fax+b*fay+c*faz
      if(ddd.ne.0.) df=df/ddd
      if(ff.lt.0..or.df.lt.0.)M_model_analogy(nmm).i_elastic = 0
      if(abs(df).lt.0.0001) M_model_analogy(nmm).i_elastic = 2          ! 14
      iet=M_model_analogy(nmm).i_elastic
      if(iet.eq.0) then                                                ! 15
       write(76,'(a,i4,9e12.4)')' 弾性復活',iet,
     * ff,df,fax,fay,faz,M_model_analogy(nmm).d_eps_x,
     *   M_model_analogy(nmm+1).d_eps_x,
     *   M_model_analogy(nmm+2).d_eps_x
      else
       write(76,'(a,i4,9e12.4)')' 塑性継続',iet,
     * ff,df,fax,fay,faz,M_model_analogy(nmm).d_eps_x,
```

```
     *   M_model_analogy(nmm+1).d_eps_x,
     *   M_model_analogy(nmm+2).d_eps_x
      endif
      endif
c―――――――――――――――――――――――――★f_N < -x_Nb 圧縮
      elseif(f_N.lt. -x_Nb ) then                                    ! 16
      i_hosei=2
c―――――――――――――――――――――――――★降伏関数
      ff=((f_N + x_a)**2 +                                           ! 17
     *(M_model_analogy(nmm+1).d_stress_x/E_model_analogy(nem+1).Q_1)**2+
     *(M_model_analogy(nmm+2).d_stress_x/E_model_analogy(nem+2).Q_1)**2-
     * x_r0**2)
      M_model_analogy(nmm).d_E = ff     ! 降伏関数の値をセット
      if(iett.eq.0) then
c―――――――――――――――――――――――――★弾性状態の場合
      if(ff.ge.0.) then
       M_model_analogy(nmm).i_elastic=2
      write(76,'(a,3i4,8e12.4)')' 塑性発生',
     *M_model_analogy(nmm).i_elastic,nem,nmm,ff,
     *M_model_analogy(nmm).d_stress_x,M_model_analogy(nmm+1).d_stress_x,
     *M_model_analogy(nmm+2).d_stress_x,E_model_analogy(nem).Q_1,
     *E_model_analogy(nem+1).Q_1,E_model_analogy(nem+2).Q_1
      endif
      else
c―――――――――――――――――――――――――★部材が既に塑性の場合
      a=fax*E_model_analogy(nem).AK_1
      b=fay*E_model_analogy(nem+1).AK_1
      c=faz*E_model_analogy(nem+2).AK_1
      df=a*M_model_analogy(nmm).d_eps_x +
     *   b*M_model_analogy(nmm+1).d_eps_x +
     *   c*M_model_analogy(nmm+2).d_eps_x
      ddd=a*fax+b*fay+c*faz
      if(ddd.ne.0.) df=df/ddd
      if(ff.lt.0..or.df.lt.0.)M_model_analogy(nmm).i_elastic = 0
      if(abs(df).lt.0.0001) M_model_analogy(nmm).i_elastic = 2
      iet=M_model_analogy(nmm).i_elastic
      if(iet.eq.0) then
       write(76,'(a,i4,9e12.4)')' 弾性復活',iet,
     * ff,df,fax,fay,faz,M_model_analogy(nmm).d_eps_x,
     *   M_model_analogy(nmm+1).d_eps_x,
     *   M_model_analogy(nmm+2).d_eps_x
      else
      write(76,'(a,i4,9e12.4)')' 塑性継続',iet,
     * ff,df,fax,fay,faz,M_model_analogy(nmm).d_eps_x,
     *   M_model_analogy(nmm+1).d_eps_x,
     *   M_model_analogy(nmm+2).d_eps_x
      endif
      endif
c―――――――――――――――――――――――――★x_Nb > f_N > -x_Nb
      else                                                           ! 18
      i_hosei=0
c―――――――――――――――――――――――――★降伏関数
      ff=(M_model_analogy(nmm).d_stress_x/E_model_analogy(nem).Q_1)**2
     * + dsqrt(
```

```
     *(M_model_analogy(nmm+1).d_stress_x/E_model_analogy(nem+1).Q_1)**2+
     *(M_model_analogy(nmm+2).d_stress_x/E_model_analogy(nem+2).Q_1)**2)
      M_model_analogy(nmm).d_E = ff      ! 降伏関数の値をセット
c――――――――――――――――――――――――★弾性状態の場合
      if(iett.eq.0) then
      if(ff.ge.1.) then
      M_model_analogy(nmm).i_elastic=2
      write(76,'(a,3i4,8e12.4)')' 塑性発生',
     *M_model_analogy(nmm).i_elastic,nem,nmm,ff,
     *M_model_analogy(nmm).d_stress_x,M_model_analogy(nmm+1).d_stress_x,
     *M_model_analogy(nmm+2).d_stress_x,E_model_analogy(nem).Q_1,
     *E_model_analogy(nem+1).Q_1,E_model_analogy(nem+2).Q_1
      endif
      else
c――――――――――――――――――――――――★部材が既に塑性の場合
      a=fax*E_model_analogy(nem).AK_1
      b=fay*E_model_analogy(nem+1).AK_1
      c=faz*E_model_analogy(nem+2).AK_1
      df=a*M_model_analogy(nmm).d_eps_x +                              ! 13
     *   b*M_model_analogy(nmm+1).d_eps_x +
     *   c*M_model_analogy(nmm+2).d_eps_x
      ddd=a*fax+b*fay+c*faz
      if(ddd.ne.0.) df=df/ddd
      if(ff.lt.1..or.df.lt.0.) M_model_analogy(nmm).i_elastic = 0
      if(abs(df).lt.0.00001) M_model_analogy(nmm).i_elastic = 2
      iet=M_model_analogy(nmm).i_elastic if(iet.eq.0) then
       write(76,'(a,i4,9e12.4)')' 弾性復活',iet,
     * ff,df,fax,fay,faz,M_model_analogy(nmm).d_eps_x,
     *   M_model_analogy(nmm+1).d_eps_x,
     *   M_model_analogy(nmm+2).d_eps_x
      else
       write(76,'(a,i4,9e12.4)')' 塑性継続',iet,
     * ff,df,fax,fay,faz,M_model_analogy(nmm).d_eps_x,
     *   M_model_analogy(nmm+1).d_eps_x,
     *   M_model_analogy(nmm+2).d_eps_x
      endif
      endif
      endif
      return
      end
```

上記のサブルーチンについて、右側のコメント番号に従ってプログラムの内容を説明する。

1. 降伏関数の微分を求めるサブルーチン Cal_div_potential_M1()では、式(3.25)にしたがって微係数を求める。ただし、部材が軸力のみの場合は、降伏関数に尖った点が生じる。これを避けるために、M=0 で、$\bar{N}$=+ Nb、$\bar{N}$=- Nb の 2 点近傍において降伏関数を球で近似する。従って、軸力の大きさによって降伏関数の状態が 3 つに分類

され、パラメータ i_hosei（0：Nb ＞ $\bar{N}$ ＞ -Nb、1：$\bar{N}$ ＞ Nb、2：-Nb ＞ $\bar{N}$）で見分ける。ここで、Nb は、その断面の降伏軸力の95%の値としている。

まず、ここでは、そのアナロジーばねが弾性状態であるかどうかチェックする。弾性の場合は、微係数をゼロとする。

2. 以降の処理は塑性状態に関する処理である。ここでは、$\bar{N}$ ＞ Nb の場合で、各微係数を計算する。この領域の降伏関数と微係数は式(3.27)と(3.27a)を用いて求める。
3. 次は、-Nb ＞ $\bar{N}$ の場合であり、同様に微係数を計算する。この領域の降伏関数と微係数は、式(3.28)と(3.28a)を用いて求める。
4. 最後に、Nb ＞ $\bar{N}$ ＞ -Nb の場合で、式(3.25)で表される微係数が計算される。
5. 係数 ff がゼロの場合、ゼロ割り算を生じるため、微係数 ffy と ffz をゼロとする。
6. 係数がゼロでない場合は、通常の微係数を計算する。これで微係数が求まったことになるのでこのサブルーチンから戻る。
7. このサブルーチン Check_Analogy_M1()では、アナロジーばねの弾塑性チェックを行う。最初に、このアナロジーばねの弾塑性状態を表すパラメータを構造体より iett に取得する。次に、次式で表す無次元軸力 $\bar{N}$（コード内では f_N である）を求める。

$$\bar{N} = \frac{N}{N_p}$$

8. この無次元軸力が、$\bar{N}$ ＞ Nb であるとき、領域を表すパラメータ i_hosei に 1 をセットする。
9. 降伏関数(3.24)の値を求める。
10. その値を構造体成分 M_model_analogy(nmm).d_E にセットする。
11. このアナロジーばねの現在の弾塑性状態をチェックし、弾性の場合以下の処理を行う。
12. 弾性の状態で降伏関数の値が正より大きいか等しい場合、塑性状態になったことを示すので、塑性状態：２を、弾塑性状態を表す構造体 M_model_analogy(nmm).i_elastic にセットする。部材に塑性が発生したことを出力する。
13. ばねが既に塑性状態で、さらに降伏関数の値が弾性復活したかチェックし、復活した場合は M_model_analogy(nmm).i_elastic に弾性状態を示す 0 をセットする。
14. ただし、その値の変化量が非常に小さい場合は、弾性復活を認めず、

塑性状態とする。

15. 以上の処理を終了した後、部材が塑性状態のままか、弾性復活したかを出力する。
16. これ以降の処理は、-Nb > $\bar{N}$ の場合について処理を行う。処理の流れは、上記とほとんど同じである。領域を表すパラメータ i_hosei=2 にセットする。
17. 降伏関数の値を計算し、構造体にセットする。後の処理はほとんど同じである。
18. これ以降の処理は、Nb > $\bar{N}$ > -Nb の場合について処理を行う。処理の流れは上記とほとんど同じである。領域を表すパラメータ i_hosei に 0 をセットする。