9.4 静的縮合部材モデルの組み込み方法

9.4.1 部材モデルの設計

本節では、新たな部材モデルを SPACE に組み込む方法について解説する。現在、SPACE には各種の静的縮合モデルが組み込まれている。しかし、そのいずれもが部材中のエレメントの位置やその数などは設計されており、ユーザーが任意に設定することはできない。そこで例題として、静的縮合モデルにおける部材中の各種モデルのエレメントを任意に並べることができ、また履歴モデルを選択できる部材モデルを設計して、SPACE に組み込むことを考えよう。静的縮合モデルを組み込む作業はそれ自体複雑で面倒である。ここでは、さらに部材内部のエレメント数が任意であることによる難しさが重なる。しかし、この例題を理解することで SPACE に関する有用な情報が多数得られることになろう。なお、このモデルは、Ver.2.20 では既に標準モデルとして組み込まれている。

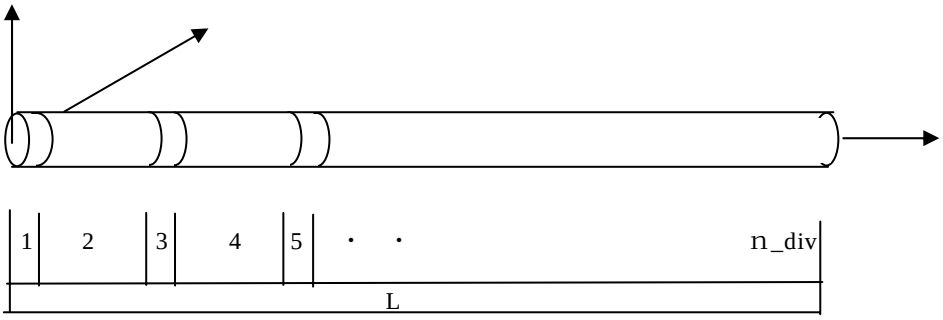

図 9-1 静的縮合モデル

ここでは、例題として新規に開発する静的縮合モデルに関する基本的な設計目標を示す。

1．部材内エレメント数 n_div は任意とし、静的縮合を行って両端の剛性や部材端力を求める。
2．部材は両端 2 節点で 1 節点 6 自由度を有する。
3．部材は、直線部材であり、内部で座標変換はないものとする。
4．幾何学的非線形性を考慮した弾塑性エレメントを含む。
5．部材内エレメントの部材モデルは、現在 SAPCE に組み込まれているモデルを使用するが、将来は新たなモデルを組み込めるように、階層構造を用いて設計する。
6．静的縮合モデルを定義する情報をファイルより読み込む。安定した静的縮合が得られる定義ファイルの設定は使用者に任せられる。したがって、モデルはそのファイルで管理することになる。
7．部材モデル番号は、51 から 70 までとし、ひとつの解析で同時に 20 モデル使用することができる。

８．プログラムの性格上、部材内のエレメント数を 100 以内に制限する。また、この部材内エレメントにおける断面特性の種類は 50 以内とする。現在は、弾性はり、ファイバー断面、MS 断面、アナロジーモデルによるばね断面の 4 種を組み込む。今後、順次増加可能となるように設計する。

このモデルを設計する上で、最も重要な点は次の２つであり、最初に、この 2 点について考える。

１．静的縮合モデルを定義する構造体の設計とその使用法
２．部材内エレメントの情報やワーク領域、例えばファイバーのワーク領域構造体の管理と、プログラムからこれら構造体へのアクセス手法

9.4.2 静的縮合モデル設定ファイルの仕様と構造体

本節では、新規に設定した部材モデルで、必要となる構造体と静的縮合モデル設定ファイルの仕様について述べる。これは前節で指摘した要点の第 1 に相当する。現在 SPACE に組み込まれている静的縮合モデルでは、サブルーチン set_model11_dat()などを用いて、部材モデルとして必要となる情報を設定している。設定しているデータとは以下のようである。

１．部材モデルのエレメント数
２．内部節点の自由度
３．内部エレメントの剛性行列半バンド幅
４．部材節点の拘束番号表あるいは自由度番号表
５．部材内エレメントの長さ
６．部材内に配置されたエレメントのタイプ

上記を考慮して静的縮合モデルを定義する構造体を設計してみよう。この構造体は、New_submain.h で定義され、必要となるサブルーチンの中にインクルードされる。この構造体の名前は S_comp_model_s とする。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   S_comp_model_s 構造体　任意静的縮合モデル
C     ―――――――――――――――――――――――――――――――――
c
c  モデルパラメータ
```

```
      structure  / S_comp_model_s /
      integer    n_div_element          ! 部材モデル中のエレメント数
      integer    i_set_ok               ! データの変換が終了か（0:変換前  1:終了）
      integer    n_out_stress           ! ファイバーデータ出力個数
      integer    nm_out_stress_x        ! 応力出力個数
      integer    n_if                   ! 内部節点自由度
      integer    n_iubw                 ! 半バンド幅
      integer    n_type_element(50)     ! エレメント型(1:弾性梁、2:ファイバー、3:MS、4:アナロジー
      integer    nm_out_stress(100)     ! 応力表示エレメント番号
      integer    nm_type_element(100)   ! エレメントモデルの番号
      integer    irest_Point(6,101)     ! 部材内の自由度
      real*8     alength(100)           ! エレメントモデルの長さ
      integer    n_out_stress_x(5)      ! 応力出力エレメント番号
      end structure
c      record / S_comp_model_s / S_comp_model
```

次に、新規静的縮合モデルの定義ファイルを設計する。この定義ファイルは以下の仕様とする。

```
1.  コメント行数
2.  上記行数分、全体コメント
3.  モデル個数、最大モデル番号
    以下のデータをモデル個数分繰り返す
4.  1 行のモデル用コメント
5.  モデル番号、部材モデル中の要素数(n_div)
6.  要素モデル番号(1 －  n_div) x 軸原点より順に設定する。
7.  要素の長さ（1 - n_div）(部材長さを 1．として、比率で設定する)
8.  次のデータを（1 －  n_div+1）繰り返す
9.  節点自由度(1 －  6 ）  x 軸原点より順に設定する。
10. 弾塑性応力出力・表示番号（1 －  n_out_stress）
11. 応力出力数、応力出力番号(1-5)
```

弾塑性応力出力・表示番号
項目数：１－n_div
番号の意味
1: ｉ端位置
2: ｊ端位置
3: 中央
4: ｉ端接合部
5: ｊ端接合部
0: 表示せず
注：弾性部材はこの値は無視される。

両端ファイバーモデルである部材モデル番号 11 で、両端にせん断変形エレメントを含まないモデルについて、上の仕様にしたがってモデル設定用ファイルを作ってみよう。

```
1
テスト用設定ファイル
1,51
両端ファイバーモデル:モデル番号 11
51,4
2,1,1,2
0.03,0.47,0.47,0.03
-1,-2,-3,-4,-5,-6
1,1,1,1,1,1
1,1,1,1,1,1
1,1,1,1,1,1
-7,-8,-9,-10,-11,-12
1, 0, 0, 2
2,1,6,0,0,0
```

静的縮合部材の節点拘束仕様として、外部節点の自由度番号は負符号を付け、拘束は 0 とする。内部節点では、1 が自由で 0 が拘束となり、10 以上の値は、他の内部節点との変位の同一視を表す。この場合、第一位のけたは、自由度番号で、第二けた以上が節点番号を表す。

上記の静的縮合モデルの定義ファイルにしたがって、構造体にデータを設定するサブルーチンを設計する。このサブルーチンでは、構造体の大きさを動的に確保するため、上記ファイルを２度読むことになる。このサブルーチン Get_S_comp_model()を以下に示す。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_S_comp_model
C       ―――――――――――――――――――――――――――――――――――
C     ●   静的縮合部材モデルの定義ファイルを読む
C     ―――――――――――――――――――――――――――――――――――
      subroutine Get_S_comp_model(nx, nx_file,S_comp_model,Model_type ,ieerx)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "New_submain.h"
      record / n_model_s / Model_type
      record / S_comp_model_s    / S_comp_model
      dimension S_comp_model(*)
      character aa*1
C     ―――――――――――――――――――――――――――――――――――
c     nx              :1: パラメータセット 2:データ入力
c     nx_file     設定モデルの最大モデル番号 -  50 ：構造体 S_comp_model の確保用
c     1. コメント行数
c     2. 上記行数分、全体コメント
c     3. モデル個数、最大モデル番号
c        以下のデータをモデル個数分繰り返す
c     4. 1行のモデル用コメント
c     5. モデル番号、部材モデル中の要素数(n_div)
c     6. 要素モデル番号(1 -  n_div) x 軸原点より順に設定する。
C     7. 要素の長さ（1 - n_div）(部材長さを 1. として、比率で設定する)
C     8. 次のデータを（1 -  n_div+1）繰り返す
C     9. 節点自由度(1 -  6 )  x 軸原点より順に設定する。
C     10. 応力出力エレメント番号（1 -  n_out_stress）
C     11. 応力出力数、応力出力番号(1-5)
C     ―――――――――――――――――――――――――――――――――――
      ierrx=0
      if(nx.eq.1) then
C     ―――――――――――――――――――――――――――――――――――
c     ★         モデルの定義ファイルを予備入力し、構造体の値を設定する
C     ―――――――――――――――――――――――――――――――――――
      nfix=5
      nfi=65
      call infile(nfi,nfix,ierr)
      if(ierr.eq.0) then
      read(nfix,*) ii
      do i=1,ii
      read(nfix,'(a)')  aa
      enddo
      read(nfix,*) nn,nx_file
      nx_file=nx_file-50
      close(nfix)
      else
      ierr=1
```

```
      endif
      return
C     ―――――――――――――――――――――――――――――――――――
c     ★        静的縮合モデルの定義ファイルを入力する
C     ―――――――――――――――――――――――――――――――――――
      else
      nfix=5
      nfi=65
      call infile(nfi,nfix,ierr)
      read(nfix,*) ii
      do i=1,ii
      read(nfix,'(a)')  aa
      enddo
      read(nfix,*) nn, nx_file
      nx_file=nx_file-50
      do ij=1, nn
      read(nfix,'(a)')  aa
      read(nfix,*)  number, n_div
      i= number -50                                    ! モデルより 50 を減ずる
      S_comp_model(i).n_div_element = n_div
      read(nfix,*) (S_comp_model(i).nm_type_element(j),j=1,n_div)
      read(nfix,*) (S_comp_model(i).alength(j),j=1,n_div)
      do j=1,n_div+1
      read(nfix,*) (S_comp_model(i).irest_Point(k,j),k=1,6)
      enddo
      read(nfix,*) (S_comp_model(i). nm_out_stress(j),j=1, n_div)
      n_out_stress=0
      do j=1,n_div
      if(n_out_stress.lt.S_comp_model(i).nm_out_stress(j))
     +      n_out_stress=S_comp_model(i).nm_out_stress(j)
      enddo
      S_comp_model(i).n_out_stress=n_out_stress
      read(nfix,*) n_out_stress,
     *     (S_comp_model(i).n_out_stress_x(j),j=1, 5)
      S_comp_model(i).n_out_stress=n_out_stress
C     ―――――――――――――――――――――――――――――――――――
      Model_type.no_e_model(number)   = number  !要素モデルの番号
      Model_type.n_div_model(number)  = n_div   !要素モデルの分割数
      Model_type.n_e_model(number)    = 0       !要素モデルの数
      Model_type.n_m_model(number)    = 0       !部材モデルの数
      Model_type.n_damp(number)       = 0       !部材減衰ありか
C     ―――――――――――――――――――――――――――――――――――
c     ★        各要素の個数を数える
C     ―――――――――――――――――――――――――――――――――――
      S_comp_model(i).i_set_ok=0
      do k=1,50
      S_comp_model(i).n_type_element(k)=0
      enddo
      do k=1,n_div
      j= S_comp_model(i).nm_type_element(k)
      S_comp_model(i).n_type_element(j)= S_comp_model(i).n_type_element(j)+1
      enddo
      enddo
      close(nfix)
```

```
      endif
      return
      end
```

このサブルーチン Get_S_comp_model()は、submain_dynamic_a()の中でコールされ、静的縮合モデルが設定される。この静的縮合モデルに関する情報が構造データを入力するサブルーチンで必要となるため、この構造データを入力する前でコールされる必要がある。まず、構造データを予備入力し、必要なパラメータを先に取得する。この時点で、静的縮合モデルに関するデータも同様に設定することにしよう。

第２の重要な点は実際のファイルとどのようにリンクするか、また、ファイルを読む、読まないをどのように判断するかである。ここでは、構造データの中にこの新規モデルを含んでいるかどうかチェックして、定義ファイルを読むかどうかを判断する。その判断パラメータとして構造体成分 Parameter_C.n_Scomp_model を用いる。なお、この構造体成分は新しく構造体 parameter_s に追加し、構造データを予備入力するサブルーチン Get_parameters()で設定されなければならない。また、ファイル番号はコントロールファイルの中の現在使用していない番号 65 を用いることにする。ファイル名の指定などは、SPACE 管理システム（他のファイルを指定する箇所）で行うことになる。

最初に構造体 parameter_s の成分に次の 3 つの成分を追加しておこう。まず、n_S_comp_model、そして、nE_New_Element、nM_New_Element である。構造体成分 n_S_comp_model は、先に示した新規の静的縮合モデルの数であり、同じく成分 nE_New_Element と nM_New_Element は、新規モデルに含まれる要素内と部材内のエレメント数である。

構造体 parameter_s で追加した成分は、新規モデルのために新たに設計した構造体の配列の大きさを以下のように決定するために使用される。

n_S_comp_model →
S_comp_model_s
nE_New_Element →
E_Fiber_work_s
nM_New_Element →
M_ Fiber_work_s

さらに、n_model_s 構造体に追加する 2 成分で構造体の配列の大きさを決定する。

n_e_New_fiber →
E_modelx_s
n_m_New_fiber →
M_ modelx_s

上記の５つの中の下 4 つの構造体に関する動的確保に関するコードは後節で説明する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   parameter_s 構造体
C     ―――――――――――――――――――――――――――――――――
c     解析パラメータ
      structure  / parameter_s/
      integer    n_unknown       !  全自由度
      integer    n_point         !  節点数
      integer    n_element       !  要素数
      integer    n_element_dll   !  DLL 用要素数
      integer    n_member        !  部材数
      integer    n_rot_axis      !  主軸回転部材数
      integer    n_local_coord   !  局所座標系を使用する節点数
      integer    n_boundary_p    !  境界節点数
      integer    nc_member       !  部材減衰機構を有する部材数
      integer    n_member_dll    !  DLL 用部材数
      integer    n_S_comp_model  !  任意型静的縮合モデル数
      integer    nE_New_Element  !  任意型静的縮合モデルに含まれる要素エレメント数
```

```
      integer    nM_New_Element  !  任意型静的縮合モデルに含まれる部材エレメント数
      integer    n_free          !  節点当たり解析自由度数
      integer    n_dim           !  解析次元数
      integer    n_skyline       !  スカイライン行列の領域数
      integer    n_sky_ave       !  平均バンド幅
      end structure
c      record /parameter_s/ Parameter_C
```

次に、構造体 n_model_s にも次に示す２つの成分を追加する。追加する成分 n_e_New_fiber と n_m_New_fiber は、新規モデルの要素内及び部材内に存在する特殊断面(ファイバーエレメントやアナロジーエレメントなど)の総数を示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   n_model_s 構造体
C     ―――――――――――――――――――――――――――――――
c
c  モデルパラメータ
      structure  / n_model_s/
      integer    n_e_models          ! 要素モデルの最大数
      integer    no_e_model(100)     ! 要素モデルの番号
      integer    n_div_model(100)    ! 要素モデルの分割数
      integer    n_e_model(100)      ! 要素モデルの数
      integer    n_m_model(100)      ! 部材モデルの数
      integer    n_damp(100)         ! 部材減衰
      integer    n_m_damp            ! 全部材減衰数
      integer    nm_div_fmodel       ! ファイバー要素の最大数
      integer    nm_div_felement     ! ファイバー要素のエレメント最大数
      integer    n_m_bilinear        ! ファイバー要素バイリニア用の最大要素数
      integer    n_m_trilinear       ! ファイバー要素トリリニア用の最大要素数
      integer    n_m_Concrete        ! ファイバー要素コンクリート用の最大要素数
      integer    n_m_analogy         ! アナロジー要素の最大要素数
      integer    nm_div_msmodel      ! ms 要素の最大数
      integer    nm_div_mselement    ! ms 要素のエレメント最大数
      integer    n_m_ro_model        ! せん断型モデルで使用する RO モデルの総数
      integer    n_m_filter          ! Maxwell 用フィルターの設計あり、なし
      integer    n_spring            ! MSS モデルのばね要素の数
      integer    n_e_New_fiber       ! 新規モデルの要素内特殊断面の総数
      integer    n_m_ New_fiber      ! 新規モデルの部材内特殊断面の総数
      real*8     cosin(2,16)         ! MSS モデルの角度係数を入れるワークエリア
      end structure
c      record / n_model_s / Model_type
```

次に、submain_dynamic_a()の中でサブルーチン Get_S_comp_model() をコールするコードを示そう。ここでは、データを２度読みして、設定領域構造体を動的確保する。無論、ここで動的確保した構造体は、宣言、保存、解放の各処理を行う必要があるが、ここでは省略する。さらに構造体 E_Fiber_work も動的領域確保を行っているが、これについては後で説明する。

```
c―――――――――――――――――――――――――――――――――――――――――
c
c    ★        構造体の大きさを動的確保する（その１）
c
c―――――――――――――――――――――――――――――――――――――――――
      ALLOCATE (Max_disp(Parameter_C.n_point))
      ALLOCATE (Member(Parameter_C.n_member))
      ALLOCATE (Max_stress(Parameter_C.n_member))
      ALLOCATE (Element(Parameter_C.n_element))
      ALLOCATE (Point(Parameter_C.n_point))
c―――――――――――――――――――――――――――――――――――――――――
c
c    ★        配列の大きさを動的確保する
c
c―――――――――――――――――――――――――――――――――――――――――
      N= Parameter_C.n_S_comp_model                    ! 任意静的縮合型モデル
      if(N.ne.0) then
      nx=1
      call Get_S_comp_model(nx,nx_file,S_comp_model, Model_type ,ieerx)
      if(ieerx.ne.0) then
      write(76,*) ' 任意静的縮合型モデル用ファイルがありません ',ieerx
      return
      else
      N = Parameter_C.nE_New_Element          !  任意型静的縮合モデルに含まれる要素エレメント数
      if(N.ne.0)then
      ALLOCATE (E_Fiber_work(N))              ! 新規縮合モデルの要素エレメント数の動的確保  ①-2
      endif
      N = nx_file                             ! 任意静的縮合型モデル
      if(N.ne.0)then
      ALLOCATE (S_comp_model(N))              ! 新規縮合モデル設定領域の動的確保
      endif
      nx=2
      call Get_S_comp_model(nx,nx_file,S_comp_model, Model_type ,ieerx)
      endif
      endif
c―――――――――――――――――――――――――――――――――――――――――
      N=Parameter_C.n_point            ! 節点数
      ALLOCATE (
     *   fll_static_point(3,6,N),am_point(2,N),
     *   fll_force_point(3,N)
     *         )
```

これで、構造体 S_comp_model に新規モデルの情報がセットされたことになる。次は、この構造体を利用して静的縮合モデルを設定するサブルーチン set_modelx_dat()を設計する。このサブルーチンは構造体 S_comp_model を用いて新規部材モデルに必要なデータを作り出す。その内容は、第 5.9.4 節で説明した set_model11_dat()とほぼ同じであり、理解することは容易であろう。後節で説明する剛性を計算するサブルーチンの中で、このサブルーチンは使用される。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /set_modelx_dat
C       ―――――――――――――――――――――――――――――――――――
C     ●   部材モデルの拘束表作成(ok)  任意部材
C     ―――――――――――――――――――――――――――――――――――
      subroutine set_modelx_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,S_comp_model,
     *         n_type,alength,gxi,gxj)
C     ―――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "New_submain.h"
      record / element_s    / Element
      record / member_s     / Member
      record / S_comp_model_s / S_comp_model
      dimension irest_Point(6,*),n_type(*),alength(*)
      real*8  gxi,gxj                 ! 剛域
c―――――――――――――――――――――――――――――――――――★
c      n_div= S_comp_model.n_div_element
      do i=1,n_div
      n_type(i)= S_comp_model.nm_type_element(i)
      enddo
      al=Member.alength - gxi - gxj      ! 剛域を削除
      do i=1,n_div
      alength(i)=al* S_comp_model.alength(i)
      enddo
      nn=n_div+1
      do i=1,nn
      do j=1,6
      irest_Point(j,i)= S_comp_model.irest_Point(j,i)
      enddo
      enddo
c―――――――――――――――――――――――――★ 内部変位の節点番号セット
      if(S_comp_model.i_set_ok.eq.0) then
      n_if =0
      do i=2,nn-1
      do j=1,6
      if(irest_Point(j,i).eq.1) then
      n_if = n_if +1
      irest_Point(j,i)= n_if
      else
      irest_Point(j,i)= -irest_Point(j,i)
      endif
      enddo
      enddo
      do i=2,nn-1
      do j=1,6
      if(irest_Point(j,i).lt.-10) then
      irest= -irest_Point(j,i)
      ip=irest/10
      ipp= Mod(irest,10)
      irest_Point(j,i)=irest_Point(ipp,ip)
      endif
      enddo
```

```
      enddo
      call set_iubw(irest_Point,n_div,iubw)
c―――――――――――――――――――――――――★ 構造体でデータ設定
      S_comp_model.i_set_ok=1
      S_comp_model.n_if=n_if
      S_comp_model.n_iubw=iubw
      do i=1,nn
      do j=1,6
      S_comp_model.irest_Point(j,i)= irest_Point(j,i)
      enddo
      enddo
      else
c―――――――――――――――――――――――――★ 2 度目以降の設定
      n_if= S_comp_model.n_if
      iubw= S_comp_model.n_iubw
      endif
      return
      end
```

9.4.3 部材モデルの階層構造と構造体

本節では、先に指摘した部材内エレメントの情報やワーク領域、例えばファイバーのワーク領域となる構造体の管理、あるいは構造体へのアクセス手法について考えよう。まず、SPACE で使用されている部材モデルの階層構造を示す。

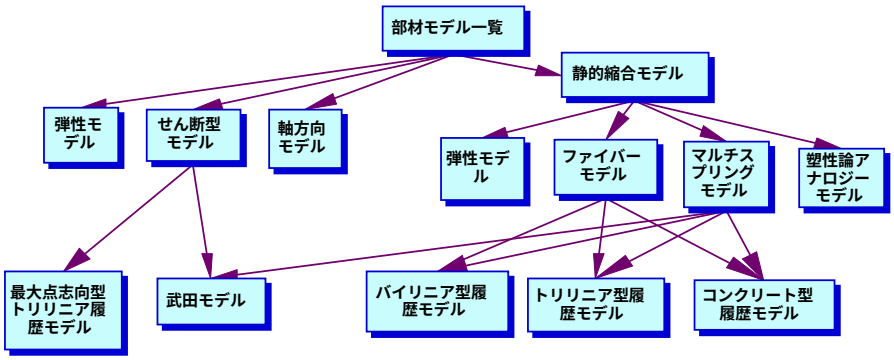

図 9-2　部材モデルの階層構造

弾塑性解析や幾何学的非線形性を考慮するために、ファイバーモデルやアナロジーモデルでは、多くのワーク領域を必要とする。このワーク領域を如何に効率よく使用するかが重要となる。ここでは、このワーク領域の使用法、特にデータへのアクセス法について考える。図 9-2 に示す部材モデル階層構造で、様々なデータ領域が構造体として定義されている。そのため、多数のモデルが混乱することなく使用でき、それに対応して設計された構造体を効率良く、しかもすばやくアクセスすること

が重要となる。

ここでは、部材モデルの階層構造に合せて、その構造体と情報の流れ、また、他の構造体へのリンク方法を検討する。例として両端ファイバーモデルとして設計された部材モデル 11 について、構造体の階層構造並びにそれらのリンク方法を以下の表にまとめた。構造体は、一般に対になっており、解析中に値が変化しない構造体と値が変化する構造体が存在する。部材に割り付けられた標準の構造体として、前者が要素構造体 Element_s であり、後者は部材に関連する構造体 Member_s である。この２つの構造体は、配列として全要素と全部材に各１つ存在することになる。表中の構造体は階層構造となっており、各部材モデルに対応する。表の最後に記述されている構造体が階層構造の最下位層となっており、各ファイバーの情報を保存する。この下位層の構造体は、効率よく配置するために要素や部材に対応せず、実際の解析モデルに合せて必要数分詰めて配置される。したがって、下位の構造体から情報を取り出すためには、上位の構造体からのリンク情報が必要となる。この表では、リンク情報を構造体から抜き出して表示し、その使用例を載せている。

表 9-1 部材モデル：両端ファイバーモデル

解析時に値が変化しない構造体　　解析時に値が変化する構造体

標準部材構造体

```
部材              Element_s                    Member_s
リンク情報     Element_s 構造体
          integer   n_element          !  非線形要素番号
リンク情報     Member_s 構造体
          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
例
        imm= Element.n_element      →    E_model11(imm)
        ie   = Member.nm_element    →    Element(ie)
        immm= Member.n_model_type   →    M_model11(immm).d_ra_1
```

静的縮合型部材モデル１１

```
                E_model11_s                    M_model11_s
リンク情報     E_model11_s 構造体
      integer   nm_section_1         !  i 端断面のファイバー開始番号
      integer   n_section_1          !  i 端断面のファイバー数
      integer   nm_section_2         !  j 端断面のファイバー開始番号
      integer   n_section_2          !  j 端断面のファイバー数
リンク情報     M_model11_s 構造体
      integer   nm_section_1         !  i 端断面のファイバー開始番号
```

```
      integer  n_section_1          !  i 端断面のファイバー数
      integer  nm_section_2         !  j 端断面のファイバー開始番号
      integer  n_section_2          !  j 端断面のファイバー数
例
       nm_div   = E_model.n_section_1  →  do i=1,nm_div
       nn       = E_model.nm_section_1 -  1 →  nm_type=E_model_fiber(nn).nm_type
       nnm      = M_model.nm_section_1 - 1  →  nmx=M_model_fiber(nnm).n_type

断面モデル

ファイバーデータ  E_model_fiber_s                  M_model_fiber_s
リンク情報    E_model_fiber_s 構造体
        integer  nm_type              !  履歴モデルの番号
リンク情報    M_model_ fiber_s 構造体
        integer  n_type               !  履歴モデルの通し番号
例
       nm_type=E_model_fiber(nn).nm_type →  goto ( 10,20,30,40,50,60,70,80),nm_type
       nmx=M_model_fiber(nnm).n_type  →    Bilinear_work(nmx).i_stat
                                            Concrete_work(nmx).P1(2)
```

```
ファイバー要素のワーク領域
ワーク領域                                  Bilinear_work_s
                                            Trilinear_work_s
                                            Concrete_work_s
```

新規の静的縮合部材モデルでは、部材内の要素数は任意であり、既存の部材モデルのように、Member_s と Element_s からリンクが張られている M_model11_s や E_model11_s のように構造体を特定することができない。そこで、他の構造体を用いてリンクを張る必要がある。まず、新規モデルで必要となる断面データやワーク領域に関する構造体を、以下のように設計する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   Model_No.51 E_modelx_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / E_modelx_s/
      integer  nm_section         !  指定断面のファイバー開始番号
      integer  n_section          !  指定断面のファイバー数
      end structure
c       record / E_modelx_s    / E_modelx
C     ―――――――――――――――――――――――――――――――――
C     ●   Model_No.51 ファイバー断面用  M_modelx_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / M_modelx_s/
```

```
      integer  nm_section        !  指定断面のファイバー開始番号
      real*8  d_aa               !  断面積の和
      real*8  d_ra               !  E*断面積の和
      real*8  d_ray              !  E*A*z
      real*8  d_raz              !  E*A*y
      real*8  d_raz2             !  E*A*y*y
      real*8  d_ray2             !  E*A*z*z
      real*8  d_rayz             !  E*A*z*y
      real*8  d_gg               !  G*A
      real*8  d_epsi_x           !  軸方向歪
      real*8  d_epsi_y           !  y 軸に関する曲げ歪
      real*8  d_epsi_z           !  z 軸に関する曲げ歪
      real*8  disp_p(12)         !  指定断面両端の変位
      end structure
c      record / M_modelx_s    / M_modelx
C     ―――――――――――――――――――――――――――――――――
C     ●   Model_No.51 Mss 断面用 M_model_mss_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / M_model_mss_s/
      integer  nm_section        !  指定断面のファイバー開始番号
      real*8  d_aa               !  断面積の和
      real*8  d_ra               !  E*断面積の和
      real*8  d_ray              !  E*A*z
      real*8  d_raz              !  E*A*y
      real*8  d_raz2             !  E*A*y*y
      real*8  d_ray2             !  E*A*z*z
      real*8  d_rayz             !  E*A*z*y
      real*8  d_gg               !  G*A
      real*8  d_epsi_x           !  軸方向歪
      real*8  d_epsi_y           !  y 軸に関する曲げ歪
      real*8  d_epsi_z           !  z 軸に関する曲げ歪
      real*8  disp_p(12)         !  指定断面両端の変位
      end structure
c      record / M_model_mss_s    / M_model_mss
C     ―――――――――――――――――――――――――――――――――
C     ●   Model_No.51 アナロジーモデル用 M_model_ang_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / M_model_ang_s/
      integer  nm_section          !  指定断面のファイバー開始番号
      real*8  fax                  !  関数φx
      real*8  fay                  !  関数φy
      real*8  faz                  !  関数φz
      real*8  c_n                  !  ｎ軸の移動
      real*8  c_my                 !  My 軸の移動
      real*8  c_mz                 !  Mz 軸の移動
      real*8  H                    !
      real*8  d_epsi_x             !  軸方向歪
      real*8  d_epsi_y             !  y 軸に関する曲げ歪
      real*8  d_epsi_z             !  z 軸に関する曲げ歪
      real*8  d_damy               !  ダミー
```

```
      real*8  disp_p(12)            !  指定断面両端の変位
      end structure
c      record / M_model_ang_s    / M_model_ang
```

上記の構造体は、この新規モデルの各ファイバー断面に対応しており、部材内エレメント数が任意であるため、Element_s と Member_s から直接アクセスすることはできない。そこで、次の 2 つの構造体を用いて間接的にアクセスする。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   Model_No.51 E_Fiber_Work  構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / E_Fiber_Work_s/
      integer  n_Fiber_section      !  指定断面の特殊断面番号
      integer  nm_section           !  特殊断面の E_modelx_s 用番号
      end structure
c       record / E_Fiber_Work_s    / E_Fiber_Work
C     ―――――――――――――――――――――――――――――――――
C     ●   Model_No.51 M_Fiber_Work  構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / M_Fiber_Work_s/
      integer  nm_section           !  特殊断面の M_modelx_s 用番号
      real*8   an_vv                !  内部エレメントのｖ方向相対変位
      real*8   an_ww                !  内部エレメントのw方向相対変位
      real*8   an_stress            !  内部エレメントの軸力
      real*8   ff_ip(6)             !  内部エレメントの不釣合力
      end structure
c       record / M_Fiber_Work_s    / M_Fiber_Work
```

構造体 E_Fiber_Work_s()は、この部材モデルの内部エレメントの特殊断面番号（含ファイバー断面、ただし弾性はりの場合は 0 をセット）と E_modelx_s へのリンク情報を保持し、その個数は、解析モデルの要素に含まれる新規モデルの全内部エレメント数となる。また、構造体 M_Fiber_Work_s()も動的配列であり、その大きさは、解析モデルの部材に含まれる新規モデルの全内部エレメント数となる。この中で、内部エレメントの変位や不釣合力は内部の釣合式を解くときに、あるいはｖ方向、ｗ方向の相対変位とエレメント軸力は幾何学的非線形性に関する剛性行列を計算するときに必要となる。

この新しく設定した動的配列と構造体を用いて、新規モデルに関する構造体の階層構造とリンク方法を以下の表にまとめる。

表 9-2 部材モデル：任意型モデル 51-70

解析時に値が変化しない構造体　　解析時に値が変化する構造体

標準部材構造体

```
部材            Element_s                    Member_s
リンク情報    Element_s 構造体
        integer  n_element          !  非線形要素番号
        integer  n_section(1)       !  部材内要素開始番号
        integer  n_section(2)       !  部材内要素数
リンク情報    Member_s 構造体
        integer  n_model            !  モデルの入れ物番号
        integer  n_model_type       !  モデル別の部材内要素開始番号
例
      imm  = Element.n_section(1)  →   E_Fiber_work(imm)
      ie   = Member.nm_element     →   Element(ie)
      immm = Member.n_model_type   →   M_Fiber_work(immm).d_ra
```

部材内要素情報

```
              E_Fiber_work_s        M_ Fiber_work_s
リンク情報    E_Fiber_work_s 構造体
     integer  n_Fiber_section   !  指定断面の特殊断面番号
     integer  nm_section        !  特殊断面の番号
リンク情報    M_Fiber_work_s 構造体
     integer  nm_section        !  特殊断面の番号
例
      nn  = E_Fiber_work.nm_section  →  nm_type= E_modelx(nn).nm_section
      nnm = M_Fiber_work.nm_section  →  nmx= M_modelx(nnm).nm_section
```

静的縮合型部材モデル 51

```
              E_modelx_s                  M_modelx_s
リンク情報    E_modelx_s 構造体
     integer  nm_section        !  断面のファイバー開始番号
     integer  n_section         !  断面のファイバー数
リンク情報    M_modelx_s 構造体
     integer  nm_section        !  断面のファイバー開始番号
例
      nm_div  = E_modelx.n_section  →  do i=1,nm_div
      nn      = E_modelx.nm_section – 1  →  nm_type=E_model_fiber(nn).nm_type
      nnm     = M_modelx.nm_section - 1  →  nmx=M_model_fiber(nnm).n_type
```

断面モデル

```
ファイバーデータ  E_model_fiber_s              M_model_fiber_s
リンク情報    E_model_fiber_s 構造体
        integer  nm_type            !  履歴モデルの番号
リンク情報    M_model_ fiber_s 構造体
        integer  n_type             !  履歴モデルの通し番号
```

例

```
nm_type=E_model_fiber(nn).nm_type →  goto ( 10,20,30,40,50,60,70,80),nm_type
nmx=M_model_fiber(nnm).n_type  →     Bilinear_work(nmx).i_stat
                                     Concrete_work(nmx).P1(2)
```

ファイバー要素のワーク領域
ワーク領域　　**Bilinear_work_s**
Trilinear_work_s
Concrete_work_s

表 9-2 に示すような構造体の階層構造とリンク情報を用いて、この新規の任意型静的縮合モデルを構築することになる。新規モデルの構築は、多くの構造体を必要とするが、その全てが動的に領域確保することになる。新規モデルで必要とする構造体配列の大きさを決めるパラメータと動的確保を行う位置について、表 9-3 にまとめる。表中の＊印は他のモデルと共用であり、他は新規モデル専用の構造体である。また、構造体配列の数欄の要素内エレメント総数と部材内エレメント総数とは、新規モデルに対する数を表し、要素内特殊断面エレメント総数と部材内特殊断面エレメント総数とは、新規モデル中における特殊断面を使用しているエレメントを抽出した数を表す。

表 9-3 構造体とリンク情報

構造体	配列の大きさを決めるパラメータ	動的確保の位置	構造体配列の数
Element_s	Parameter_C.n_element	M_alloc(1)	全要素数*
E_Fiber_work_s	**Parameter_C.nE_New_Element**	M_alloc(1)	要素内エレメント総数
E_modelx_s	**Model_type.n_e_New_fiber**	M_alloc(2)	要素内特殊断面エレメント総数
E_model_fiber_s	Model_type.nm_div_felement	M_alloc(3)	要素内特殊断面エレメント総数*
Member_s	Parameter_C.n_member	M_alloc(1)	全部材数*
M_ Fiber_work_s	**Parameter_C.nM_New_Element**	M_alloc(2)	部材内エレメント総数
M_modelx_s	**Model_type.n_m_ New_fiber**	M_alloc(2)	部材内特殊断面エレメント総数
M_model_fiber_s	Model_type.nm_div_fmodel	M_alloc(3)	部材内特殊断面エレメント総数*
Bilinear_work_s	Model_type.n_m_bilinear	M_alloc(5)	バイリニアファイバーの総数*
Trilinear_work_s	Model_type.n_m_trilinear	M_alloc(5)	トリリニアファイバーの総数*
Concrete_work_s	Model_type.n_m_Concrete	M_alloc(5)	コンクリートファイバーの総数*

構造体	備考
Element_s	入力データの要素数で総数が決定、コードを追加する必要がない。

E_Fiber_work_s	新しい構造体であるため、個数を数えるコードを新たに追加する必要がある。①
E_modelx_s	新しい構造体であるため、個数を数えるコードを新たに追加する必要がある。②
E_model_fiber_s	既存の個数を数えるコードに、新規モデル用のコードを加える。③
Member_s	入力データの部材数で総数が決定、コードを追加する必要がない。
M_ Fiber_work_s	新しい構造体であるため、個数を数えるコードを新たに追加する必要がある。④
M_modelx_s	新しい構造体であるため、個数を数えるコードを新たに追加する必要がある。⑤
M_model_fiber_s	既存の個数を数えるコードに、新規モデル用のコードを加える。⑥
Bilinear_work_s	既存の個数を数えるコードに、新規モデル用のコードを加える。⑦
Trilinear_work_s	既存の個数を数えるコードに、新規モデル用のコードを加える。⑧
Concrete_work_s	既存の個数を数えるコードに、新規モデル用のコードを加える。⑨

次節では、これらの構造体配列の大きさをどのように決定しているか、またどのようにデータをセットするか、特にリンク情報をどのように設定し、利用するかにについて解説する。大切なのは、これらの構造体配列を動的確保する場合、その個数とどのタイミングで実際に確保すべきかを決定することである。表中右側に示す①、②などは新規モデルに関する構造体の設定する箇所を示す印であり、実際のプログラムコード内に付された印と対応する。また、①-1 などは構造体配列の個数の計算部分、①-2 は動的領域の確保、①-3 はリンク情報の設定位置を示す。

9.4.4 部材モデルの入力仕様

本節では、入力関連で新規の静的縮合モデルで必要となるコードを検討する。まず構造データを予備入力するサブルーチン Get_parameters() について考えよう。ここでは２つの変更点がある。ひとつは静的縮合設定ファイルを読むか否かの判断を行うために、新規静的縮合部材モデル 51-70 を使用しているかチェックするコードを追加することであり、他の一つはこのモデルに含まれるファイバー断面の数を数えるコードである。太文字で示した部分が上記２つの作業を行うコードである。なお、このサブルーチン内では、ここでは必要のない入力データは読み捨てている。

構造データの入力仕様を少し変更することとし、新規静的縮合部材モデルの要素データを入力する部分で、標準データ入力の後の第 2 レコードを以下の仕様とする。内部エレメント数は、新規静的縮合モデルの内部に含まれる全エレメントの数であり、ファイバー断面番号で、弾性部材など特殊な断面を含まない場合は 0 とする。

１．この部材に含まれる内部エレメント数
２．ファイバー断面番号（１－ 内部エレメント数）

追加したコードと共にサブルーチン Get_parameters()を見てみよう。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_parameters
C       ―――――――――――――――――――――――――――――――――
C     ●   構造データを予備入力し、構造用のパラメータを設定する(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Get_parameters(Parameter_C,ierr)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / parameter_s  / Parameter_C
      character amoji*1
      dimension n_section(100)
C     ―――――――――――――――――――――――――――――――――
C     ―――――――――――――――――――――――――――――――――
c     ★   タイトル入力
C     ―――――――――――――――――――――――――――――――――
      ierr=0
      read(5,*,err=999) n
      if(n.ne.0) then
      do J=1,n
      read(5,'(a)') amoji
      end do
      endif
C     ―――――――――――――――――――――――――――――――――
c     ★   制御データ入力とパラメータセット
C     ―――――――――――――――――――――――――――――――――
      read(5,*,err=999) node,nelem,memb,nrbound,locod,njiku
      Parameter_C.n_point      =node
      Parameter_C.n_element    =nelem
      Parameter_C.n_member     =memb
      Parameter_C.n_boundary_p =nrbound
      Parameter_C.n_local_coord=locod
      Parameter_C.n_rot_axis   =njiku
      Parameter_C.n_free       =6
c
      write(76,'(//a)') ' 構造体：Parameter_C'
      write(76,*) Parameter_C.n_point      ,'=node'
      write(76,*) Parameter_C.n_element    ,'=nelem'
      write(76,*) Parameter_C.n_member     ,'=memb'
      write(76,*) Parameter_C.n_boundary_p ,'=nrbound'
      write(76,*) Parameter_C.n_local_coord,'=locod'
      write(76,*) Parameter_C.n_rot_axis   ,'=njiku'
      write(76,*) Parameter_C.n_free       ,'=6'
C     ―――――――――――――――――――――――――――――――――
c     ★   これ以降のコードを追加する
C     ―――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――★節点データ入力
      do i=1,node
      read(5,*,err=9912,end=9918) ii,x,y,z,idm
```

```
      end do
c―――――――――――――――――――――――――――――――――★節点局所座標入力
      if(locod.ne.0) then
      do i=1,locod
      read(5,*,err=9912,end=9918) j,tlx,tly,tlz
      end do
      end if
c―――――――――――――――――――――――――――――――――★境界拘束条件入力
c――――――――――――――――――――――――★境界拘束条件入力
      if(nrbound.ne.0) then
      do i=1,nrbound
      read(5,*,err=9912,end=9918) j,ir1,ir2,ir3,ir4,ir5,ir6
      end do
      end if
c―――――――――――――――――――――――――――――――――★線形要素モデルデータ入力
      Parameter_C.n_S_comp_model=0
      Parameter_C.nE_New_Element =0
      do  i=1,nelem
      read(5,*,err=9913,end=9918) m_type,e,g,a,rix,riy,riz,asy,asz,
     *          am1,am2,anp,ampy,ampz,nm_type
c――――――――――――――――――――――★新規の静的縮合モデルを数える
      mmx=(m_type-1)/10
      if(mmx .eq. 5 .or. mmx .eq. 6) then
      Parameter_C.n_S_comp_model= Parameter_C.n_S_comp_model + 1
      read(5,*,err=9913,end=9918) nxx,(n_section(j),j=1,nxx)
      Parameter_C.nE_New_Element= Parameter_C.nE_New_Element + nxx                    ! ①-1
      endif
c――――――――――――――――――――――★要素番号 11，15，21 に対してデータ入力
      if(m_type .eq. 11.or.m_type .eq. 15
     * .or.m_type .eq. 21) then
      read(5,*,err=9913,end=9918) rd1,rd2,sg1,sg2,j1,j2
      endif
c――――――――――――――――――――――★要素番号１2，22 に対してデータ入力
      if(m_type .eq. 12.or.m_type .eq. 22) then
      read(5,*,err=9913,end=9918) rd1,rd2,sg1,sg2,j1,j2,j3
      endif
c――――――――――――――――――――――★要素番号 31 に対してデータ入力
      if(m_type .eq. 31) then
      read(5,*,err=9913,end=9918) rd1,rd2,sg1,sg2,j1,j2
      endif
c――――――――――――――――――――――★要素番号 32 に対してデータ入力
      if(m_type .eq. 32) then
      read(5,*,err=9913,end=9918) rd1,rd2,sg1,sg2,j1,j2,j3
      endif
      end do
      return
  999 continue
      write(76,'(a)')' 構造データファイルの制御データにエラーが存在する'
      ierr=1
      return
 9912 continue
      write(76,'(a)')' 構造データファイルの節点データにエラーが存在する'
      ierr=1
      return
```

```
 9913 continue
      write(76,'(a)')' 構造データファイルの要素データにエラーが存在する'
      ierr=1
 9918 continue
      end
```

ここでは、構造体 S_comp_model を動的確保すべきかどうか判定するためパラメータ Parameter_C. n_S_comp_model と、構造体 E_Fiber_work の大きさを決めるパラメータ Parameter_C.nE_New_Element を設定している（①-1)。

構造データのファイル仕様が変更になったので、これに伴い、予備入力サブルーチン Get_parameters()と同様に、Get_structure()も変更しなくてはならない。以下に、このサブルーチンの変更部分についてのみ記す。追加したコードでは、新規の静的縮合モデルにおけるファイバー断面の番号を動的配列 n_Fiber_section にセットする。このとき、構造体成分 Element(i).n_section(1)に要素断面番号の初期値がセットされ、また、Element(i).n_section(2)にはその部材のエレメント個数がセットされる。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_structure
C       ―――――――――――――――――――――――――――――――
C     ●   構造データを入力し、データをダンプファイルに出力する。
C     ―――――――――――――――――――――――――――――――――
      subroutine Get_structure(Point,Member,Element,Parameter_C,
     *           Model_type,ierr,
     *           S_comp_model,E_Fiber_work)

      include "New_submain.h"
      record / E_Fiber_work_s / E_Fiber_work
      record / S_comp_model_s / S_comp_model
       .
      dimension  n_section(100),E_Fiber_work(*),S_comp_model(*)
       .
      nnx=0
      do  i=1,nelem
       .
      read(5,*,err=9913,end=9918) m_type,e,g,a,rix,riy,riz,asy,asz,
     *          am1,am2,anp,ampy,ampz,nm_type
       .
c―――――――――――――――――――――――――★新規の静的縮合モデルの第２レコード
      mmx=(m_type-1)/10
      if(mmx .eq. 5 .or. mmx .eq. 6) then
      read(5,*,err=9913,end=9918) nxx,(n_section(j),j=1,nxx)
c―――――――――――――――――――――――――★エレメント数チェック
      if(nxx.ne. S_comp_model(m_type-50).n_div_element)
     *  write(76,*) ‘ エレメント個数エラー  ‘, m_type, nxx
      do j=1,nxx
```

```
      E_Fiber_work(j+nnx).n_Fiber_section = n_section(j)
      enddo
      Element(i).n_section(1) =nnx+1              ! 新規モデルの要素内エレメントの最初の番地
      Element(i).n_section(2) =nxx                ! 新規モデルの要素内エレメントの個数
      nnx = nnx + nxx
      endif
c―――――――――――――――――――――★せん断タイプで修正 RO モデルの数をかぞえる
      if(m_type .eq. 2) then
      if(nm_type.eq.RO_MODEL_NUMBER.or.
     *   nm_type.eq.TRI_MODEL_NUMBER) then
      Model_type.n_m_ro_model= Model_type.n_m_ro_model + 1
      Element(i).n_section(1) = Model_type.n_m_ro_model
      endif
c―――――――――――――――――――――★要素データを構造体にセット
      ・
      ・
c――――――――――――――――――――――――★モデルタイプ別の要素数の計算（ゼロセット）
      do i=1,Model_type.n_e_models
      Model_type.n_e_model(i)=0
      Model_type.n_m_model(i)=0
      end do
      Parameter_C.n_element_dll=0
c―――――――――――――――――――――★各タイプ別の要素数を数える。
      DO i=1,nelem
      m_type = Element(i).element_type
      do j=1,Model_type.n_e_models
      if(m_type .eq. Model_type.no_e_model(j)) then
      Model_type.n_e_model(j) = Model_type.n_e_model(j)+1
      Element(i).nm_damp = Model_type.n_damp(j)
      Element(i).element_type = j       ! モデル番号からモデルの記憶領域番号に変換
      goto 19
      end if
      end do
      ierr=14
      write(damp_out,'(a,i4,a)') ' 要素：',i,
     *            'のモデルタイプが存在しない。'
   19 continue
c―――――――――――――――――――――★dll を使用した要素数を数える。
      if(m_type .gt.1000)
     *   Parameter_C.n_element_dll = Parameter_C.n_element_dll+1
      end do
      if(ierr.ne.0) goto 9914
c―――――――――――――――――――――★新規モデルの要素内エレメント数を数える
      isum=0
      do j=51,70
      if(Model_type.n_e_model(j).ne.0) then
      do k=1, Model_type.n_div_model(j)
       if(S_comp_model(j-50).nm_type_element(k).ne.1) then          ! 弾性はりエレメントを除く
       isum=isum+ Model_type.n_e_model(j)
       endif
      enddo
      endif
      enddo
      Model_type.n_e_New_fiber =isum                                    ! ②-1
```

```
      •
c―――――――――――――――――――――――――――――――★部材データ入力
      Parameter_C.nM_New_ Element = 0
      do i=1,memb
      read(5,*,err=9915,end=9918) ii,i1,i2,ie,ian,ig,iso,ii1,ii2,
     *          rigid_i,rigid_j,shear_i,shear_j
      •
      •
      if(ie.le. nelem ) then
      Member(ii).element_type = Element(ie).element_type
      mmx=(Member(ii).element_type -1)/10
      if(mmx .eq. 5 .or. mmx .eq. 6) then
      nxx= Element(ie).n_section(2)                          ! 新規モデルに含まれるエレメント数
      Parameter_C.nM_New_ Element = Parameter_C.nM_New_Element + nxx          ! ④-1
      shear_i=0
      shear_j=0
      elseif(Member(ii).element_type eq. 50 ) then
      Member(ii).nm_dll_element=1                            ! DLL 部材を数える
      endif
      write(damp_out,'(6i8,i12,2i8,4f10.2)')
     *          ii,i1,i2,ie,ian,ig,iso,ii1,ii2,
     *          rigid_i,rigid_j,shear_i,shear_j
      else
      write(damp_out,'(a,6i8)') ' data err:',ii,i1,i2,ie
      endif
      •
      •
c―――――――――――――――――――――――――――――――★モデルタイプ別部材数
      Parameter_C.n_member_dll=0
      DO 30 ii=1,memb
      i = Member(ii).nm_element
      if(i .le. nelem ) then
      m_type = Element(i).element_type
      do j=1,Model_type.n_e_models
      if(m_type .eq. j) then
      Model_type.n_m_model(j) = Model_type.n_m_model(j)+1
      Member(ii).n_model= j
      goto 29
      end if
      end do
      ierr=16
      write(76,'(a,i4,a)') ' 部材：',i,
     *       '  の要素データはモデルタイプに適合しない。'
   29 continue
      if(m_type .gt.1000)
     *   Parameter_C.n_member_dll = Parameter_C.n_member_dll+1
      else
      ierr = 16
      write(76,'(a,i4,a)') ' 部材：',i,
     *       '  は要素データに適合しない。'
      endif
   30 continue
c―――――――――――――――――――――――――★新規モデルの部材内エレメント数を数える
      isum=0
```

```
      do j=51,70
      if(Model_type.n_e_model(j).ne.0) then
      do k=1, Model_type.n_div_model(j)
       if(S_comp_model(j-50).nm_type_element(k).ne.1) then        ! 弾性はりエレメントを除く
       isum=isum+ Model_type.n_m_model(j)
       endif
      enddo
      endif
      enddo
      Model_type.n_m_New_fiber =isum                                          ! ⑤-1
c-------------------------------★モデル別通し番号計算
      do 32 i=1,Model_type.n_e_models
      if(Model_type.no_e_model(i).ne.0) then
      j=0
      do ii=1,memb
      if(i.eq.Member(ii).n_model) then
      j=j+1
      Member(ii).n_model_type = j
      endif
      enddo
      endif
   32 continue
c-----------------------★新規モデル 51-70 までの通し番号設定
      j=1
      do ii=1,memb
            if(Member(ii).n_model.ge.51 .and. Member(ii).n_model.le.70) then
            Member(ii).n_model_type = j
            j=j+Model_type.n_div_model(Member(ii).n_model)
            endif
      enddo
       .
       .
```

先に示したように構造体 E_Fiber_work の動的確保は、サブルーチン Get_prarameters()でその大きさを設定した後、Get_structur()をコールする前に行わなければならない。このコードについては前節で示した。新たに設計した他の構造体の動的確保は以下に示される。ここでは省略するが、これら新規に設計した構造体に対し、構造体の宣言、保存、解放処理を忘れてはならない。

```
c-----------------------------------------------------
c
c    ★        計算用構造体の大きさを動的確保する（その２）
c
c-----------------------------------------------------
       .
       .
c-------------------------------★Model_No.33 両端ピン、中央アナロジーモデル
      n= Model_type.n_e_model(19)        !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model33(n))
      endif
```

```
      n= Model_type.n_m_model(19)        !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model33(n))
      endif
c――――――――――――――――――――――――――――――――――★新規モデル Model_No.51-70
      n= Model_type.n_e_New_fiber        !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_modelx(n))                                          ! ②-2
      Endif
      n= Model_type.n_m_New_fiber        ! 部材モデルの数
      if(n.ne.0) then
      ALLOCATE (M_modelx(n))                                          ! ④-2
      endif
      n=Parameter_C.nM_New_Element         ! 数新規モデルにおける部材エレメント数
      if(n.ne.0) then
      ALLOCATE (M_Fiber_work(n))                                      ! ⑤-2
      endif
```

ファイバーデータを入力するサブルーチン Fiber_input()では、ファイバーデータの入力と部材データとのリンクのための処理が三箇所で行われる。新規モデルでは、ファイバーデータの仕様変更はないため入力部分の変更はない。ただし、部材モデルが追加されることによって、三箇所で処理に変更が加えられる。

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Fiber_input
C     ――――――――――――――――――――――――――――――――――
C     ●   ファイバー要素の入力(ok)
C     ――――――――――――――――――――――――――――――――――
      subroutine Fiber_input(it,ierr,n_member,n_element,Member,
     *      Element,Model_type,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,
     *      M_Fiber_work, E_Fiber_work,E_modelx,M_modelx,Parameter_C)
       .
       .
      include "New_submain.h"
      record / parameter_s    / Parameter_C
      record / M_Fiber_work_s / M_Fiber_work
      record / E_Fiber_work_s / E_Fiber_work
      record / M_modelx_s     / M_modelx
      record / E_modelx_s     / E_modelx
      dimension M_Fiber_work(*),E_Fiber_work(*),M_modelx(*),E_modelx(*)
       .
       .
      ierr=0
      if(it.eq.0) then
c――――――――――――――――――――――――――――――――――★ファイバーデータの予備入力
      read(5,*,err=999) nm          ! 最大断面数
```

```
      write(76,'(a,i4)') ' ファイバー断面総数：',nm
      ii=0
      nmmx=0
      do i=1,nm
      read(5,*,err=999) n_m,nmm,(ddm(j),j=1,20)     ! 断面番号、エレメント数
c-----------------------------★断面ファイバー数のセット
      do i1=1,n_element
      itype_m = Model_type.no_e_model(Element(i1).element_type)
      if(itype_m.eq.11) then
c-----------------------★モデル１１                              ! x1
       ・
      endif
c-----------------------★モデル１２
      if(itype_m.eq.12) then
       ・
      endif
c-----------------------★モデル１３
      if(itype_m.eq.13) then
       ・
      endif
c-----------------------★モデル１５
      if(itype_m.eq.15) then
       ・
      endif
c-----------------------★モデル２１
      if(itype_m.eq.21) then
       ・
      endif
c-----------------------★モデル２２
      if(itype_m.eq.22) then
       ・
      endif
c-----------------------★モデル３１
      if(itype_m.eq.31) then
       ・
      endif
c-----------------------★モデル３２
      if(itype_m.eq.32) then
       ・
      endif
c-----------------------★モデル３３
      if(itype_m.eq.33) then
       ・
      endif
c-----------------------★新規モデル 51-70
      nx_itype=(itype_m-1)/10                                  ! 1
      if(nx_itype.eq.5 .or. nx_itype.eq.6) then
        nmx = Element(i1).n_section(1)-1
        nnx = Element(i1).n_section(2)
        do k=1,nnx                                             ! 2
         if(E_Fiber_work(nmx+k).n_Fiber_section.eq.n_m) then    ! 3
           nmmx = nmmx + 1
           E_Fiber_work(nmx+k).nm_section = nmmx          ! ①-3
           E_modelx(nmmx).nm_section = ii+1               ! ④-3
```

```
            E_modelx(nmmx).n_section = nmm                    ! ④-3
          endif
        enddo
      endif
      enddo
c―――――――――――――――――――――――★
      do j=1,nmm
      ii = ii + 1                                             ! ①-1
      read(5,*,err=999) n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
      if(nm_type.le.10) then
      goto(901,902,903,904,905,906,907,908,909,910),nm_type
      ・
      enddo
      enddo
c―――――――――――――――――――――――★ファイバー数セット
      Model_type.nm_div_felement= ii
      write(76,'(a,i5)') ' ファイバー数：',ii
c―――――――――――――――――――――――――――★部材断面ファイバー数のセット
      jj = 0
      nmmx=0
      do i=1,n_member
      i1   = Member(i).nm_element
      immm = Member(i).n_model_type               ! モデルタイプ別番号
      itype_m = Model_type.no_e_model(Element(i1).element_type)
c―――――――――――――――――――――――★モデル１１                                  ! x2
      if(itype_m.eq.11) then
      ・
      endif
c―――――――――――――――――――――――★モデル１２
      if(itype_m.eq.12) then
      ・
      endif
c―――――――――――――――――――――――★モデル１３
      if(itype_m.eq.13) then
      ・
      endif
c―――――――――――――――――――――――★モデル１５
      if(itype_m.eq.15) then
      ・
      endif
c―――――――――――――――――――――――★モデル２１
      if(itype_m.eq.21) then
      ・
      endif
c―――――――――――――――――――――――★モデル２２
      if(itype_m.eq.22) then
      ・
      endif
c―――――――――――――――――――――――★モデル３１
      if(itype_m.eq.31) then
      ・
      endif
c―――――――――――――――――――――――★モデル３２
      if(itype_m.eq.32) then
```

```
      •
      endif
c―――――――――――――――――――――★モデル３３
      if(itype_m.eq.33) then
      •
      endif
c―――――――――――――――――――――★新規モデル 51-70
      nx_itype=(itype_m-1)/10
        nnx = Element(i1).n_section(2)
        nmx = Element(i1).n_section(1) -1
        nnmx = Member(i).n_model_type -1
        do k=1,nnx                                                    !4
         if(E_Fiber_work(nmx+k).n_Fiber_section.ne.0) then
              nmmx=nmmx+1
              nmmxx=E_Fiber_work(nmx+k).nm_section
              M_Fiber_work(nnmx+k).nm_section = nmmx      ! ④-3
              M_modelx(nmmx).nm_section = jj+1            ! ⑤-3
              jj = jj + E_modelx(nmmxx).n_section         ! ⑥-1
         endif
        enddo
      endif

      enddo
      Model_type.nm_div_fmodel =  jj       ! ファイバー要素の最大数
C     ―――――――――――――――――――――――――――――――――
C       ★      ファイバーデータの入力その２
C     ―――――――――――――――――――――――――――――――――
      else
      read(5,*,err=999) nm
      write(76,'(a,i4)') ' Number of sections:',nm
      ii = 0
      do i=1,nm
      read(5,*,err=999) n_m,nmm,(ddm(j),j=1,20)
      do j=1,nmm
      read(5,*,err=999) n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az      ! 標準データ
c―――――――――――――――――――――★部材断面ファイバー数セット
       (E_model_fiber(ii-nmm+1),nmm)
      enddo
c―――――――――――――――――――――★ファイバー履歴特性数セット
      n_m_bilinear    = 0
      n_m_trilinear   = 0
      n_m_concrete    = 0
      n_m_analogy     = 0
      do i=1,n_member
      ie  = Member(i).nm_element
      imm = Element(ie).n_element
      im  = Member(i).n_model_type
c―――――――――――――――――――――★モデル１１                                    ! x3
      itype_m = Model_type.no_e_model(Element(ie).element_type)
      if(itype_m.eq.11) then
       •
      endif
c―――――――――――――――――――――★モデル１２
      if(itype_m.eq.12) then
```

```
        ・
      endif
c―――――――――――――――――――――★モデル１３
      if(itype_m.eq.13) then
        ・
      endif
c―――――――――――――――――――――★モデル１５
      if(itype_m.eq.15) then
        ・
      endif
c―――――――――――――――――――――★モデル２１
      if(itype_m.eq.21) then
        ・
      endif
c―――――――――――――――――――――★モデル２２
      if(itype_m.eq.22) then
        ・
      endif
c―――――――――――――――――――――★モデル３１
       if(itype_m.eq.31) then
        ・
       endif
c―――――――――――――――――――――★モデル３２
       if(itype_m.eq.32) then
        ・
       endif
c―――――――――――――――――――――★モデル３３
       if(itype_m.eq.33) then
        ・
       endif
c―――――――――――――――――――――★新規モデル 51-70
      nx_type_m= (itype_m-1)/10
      if(nx_type_m.eq.5 .or. nx_type_m.eq.6) then                    ! 5
        nmx  = Element(ie).n_section(1)-1
        nnmx = Member(i).n_model_type -1
        nnx =  Element(ie).n_section(2)
        do k=1,nnx
         if(E_Fiber_work(nmx+k).n_Fiber_section.ne.0) then
           ii  = E_modelx(inmm).n_section
          imm = E_Fiber_work(nmx+k).nm_section
          inmm= M_Fiber_work(nnmx+k).nm_section
          nmm = E_modelx(imm).nm_section - 1
          nnmm = M_modelx(inmm).nm_section - 1
          do j=1,ii                                                  ! 6
            nmm = nmm + 1
           nnmm= nnmm + 1
          if(E_model_fiber(nmm).nm_type.eq.1.or.
     *       E_model_fiber(nmm).nm_type.eq.5.or.
     *       E_model_fiber(nmm).nm_type.eq.7) then
            n_m_bilinear  = n_m_bilinear + 1                       ! ⑦-1
            M_model_fiber(nnmm).n_type = n_m_bilinear
         elseif(E_model_fiber(nmm).nm_type.eq.2.or.
     *       E_model_fiber(nmm).nm_type.eq.6.or.
     *       E_model_fiber(nmm).nm_type.eq.8) then
```

```
                n_m_trilinear = n_m_trilinear + 1                    ! ⑧-1
               M_model_fiber(nnmm).n_type = n_m_trilinear
             elseif(E_model_fiber(nmm).nm_type.eq.3.or.
     *           E_model_fiber(nmm).nm_type.eq.4) then
                n_m_concrete  = n_m_concrete + 1                     ! ⑨-1
               M_model_fiber(nnmm).n_type = n_m_concrete
            elseif(E_model_fiber(nmm).nm_type.eq.11.or.
     *          E_model_fiber(nmm).nm_type.eq.12.or.
     *          E_model_fiber(nmm).nm_type.eq.13) then
                   n_m_analogy  = n_m_analogy + 1
               M_model_fiber(nnmm).n_type = n_m_analogy
             endif
           enddo
         endif
         enddo
         endif
c―――――――――――――――――――――★
      enddo
      Model_type.n_m_bilinear   = n_m_bilinear                              ! x4
      Model_type.n_m_trilinear  = n_m_trilinear
      Model_type.n_m_concrete   = n_m_concrete
      Model_type.n_m_analogy    = n_m_analogy
      write(76,'(a,i8)') ' 履歴 NO.1:',n_m_bilinear
      write(76,'(a,i8)') ' 履歴 NO.2:',n_m_trilinear
      write(76,'(a,i8)') ' 履歴 NO.3:',n_m_concrete
      write(76,'(a,i8)') ' アナロジーモデル:',n_m_analogy
      endif
      return
 999  continue
      ierr=1
      return
      end
```

新しく追加したコードの説明に入る前に、このプログラム右の x1 から x4 で示される４箇所の処理内容について復習しておこう。

x1. ここでは、現在設計されている部材モデルのどの位置で、このファイバー断面が使用されているのかをチェックする。もし使用している場合は、要素構造体 Element にファイバー断面番号をセットする。

x2. 上記と同じく、このファイバー断面がどの部材モデルで使用されているかチェックする。ただし、上記は、全要素についてであるが、ここでは、全部材についてチェックする。

x3. 断面内で使用しているファイバー履歴の解析で必要となるワーク領域構造体配列の数を数えている。ここでは、両端ファイバーモデルについてであり、プログラムコードを見ると、ｉ端とｊ端の 2 箇所でこの構造体の数を数えている。他の部材モデルでも同様の検索を以降のコードで行っていることは当然のことである。現在、このワ

ーク領域構造体は 3 種類用意されており、各構造体は対応する解析モデル中に存在するファイバー数分必要となる。その 3 種類とは、

１．バイリニア型構造体：Bilinear_work(:)

２．トリリニア型構造体：Trilinear_work(:)

３．コンクリート型構造体：Concrete_work(:)

である。ここで、チェックを行っているコードの順番は、上記のバイリニア型、トリリニア型、コンクリート型の順であり、その中の番号は履歴番号である。

x4. 最後に、各種の部材モデルで必要となるの３つのワーク領域構造体の数が、構造体 model_type.n_m_bilinear などにセットされる。

次に、新しく追加したコードについて説明しよう。

1. 部材モデル番号が 51-70 であるかどうかチェックする。そうであれば、以下の処理を行う。まず、その部材に含まれる要素数と構造体 E_Fiber_work の先頭番地を取得する。
2. 部材に含まれるエレメント数分、以下の処理を行う。
3. そのエレメントの断面番号が、入力した断面番号と一致するかどうかチェックし、もし、その範囲であれば断面内のファイバー数と先頭番地を構造体にセットする。
4. 部材に含まれるエレメント数分、以下の処理を行う。この断面の先頭番地を構造体にセットする。
5. ここでは、その部材が部材モデル番号 51-70 であるかどうかチェックする。もし、同じであれば、構造体 E_Fiber_work と M_Fiber_work の先頭番地を取得する。
6. 履歴のワーク領域である構造体 n_m_bilinear、n_m_trilinear、n_m_concrete、n_m_analogy の数を数える。

9.4.5 部材モデルの出力仕様

本節では、新規の静的縮合モデルに関するデータ出力を考えてみよう。関連する出力項目として、部材応力とファイバー応力との 2 種類があり、これらを順に検討する。最初は、部材に関する応力を出力するサブルーチン Out_stress()を示す。このサブルーチンの中で、太文字で示した箇所が、新規モデルのために追加・変更するコードである。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Out_stress
C       ―――――――――――――――――――――――――――――――――
C     ●   部材両端と中央の応力を出力(ok)
C     ―――――――――――――――――――――――――――――――――――
      subroutine Out_stress(Member,Element,E_model6_real,M_model11,
     *                      M_model12,M_model13,M_model15,M_model21,
     *                      M_model22,M_model31,M_model32,M_model33,
     *                      n_member,ifl,iflz,i_print,Out_section)
C     ―――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      dimension M_model(*)
      data mxtype/1,1,1,1,1,1,1,1,1,1, 1,3,1,3,1,1,3,3,3,1,
     3           1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,
     5           1,1,1,1,1,1,1,1,1,1, 3,3,3,3,3,3,3,3,3,3,
     7           3,3,3,3,3,3,3,3,3,3, 1,1,1,1,1,1,1,1,1,1,
     9           1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1/
      data myytype/2,4,3,5/
       •
       •
c     i_print         :integer 出力制御変数 0:ファイル出力あり
c――――――――――――――――――――――――――――★応力 5
      if(i_print.ne.0) return
      if(ifl(5).ne.0) then
       do i=1,n_member
       •
      if(Member(i).element_type.eq.6) then
       •
c――――――――――――――――――――――――――――★Maxwell モデル
      elseif(Member(i).element_type.eq.2) then
c――――――――――――――――――――――――――――★3 次元せん断弾塑性モデル
       •
      elseif(Member(i).element_type.eq.3) then
c――――――――――――――――――――――――――――★3 次元軸力弾塑性モデル
       •
      elseif(Member(i).element_type.eq.4 ) then
c――――――――――――――――――――――――――――★3 次元ブレースモデル
       •
      elseif(Member(i).element_type.ge.5.and.
     *   Member(i).element_type.le.10) then
c――――――――――――――――――――――――――――★Model No.5-10
        •
c――――――――――――――――――――――――――――★Model No. 18,19
      elseif(Member(i).element_type.eq.18.or.
     *       Member(i).element_type.eq.19) then
       •
      else
c――――――――――――――――――――――――――――★有限要素モデル
      i_t=Member(i).element_type
      im=mxtype(i_t)
      ns=myytype(im)
      ie = Member(i).nm_element
```

```
 immm= Member(i).n_model_type                ! モデルタイプ別番号
 do j=1,2
 istat = Member(i).d_stat(j)
 jj=6*(j-1)
  v(1)=Member(i).stress(jj+1)          ! 軸力
  v(2)=Member(i).stress(jj+5)          ! ｙ軸モーメント
  v(3)=-Member(i).stress(jj+6)         ! ｚ軸モーメント
 rrxx=0.                              ! 現在ダミー　塑性関数値
 if(Element(ie).ANP.ne.0.)
*      rrxx=(Member(i).stress(jj+1)/Element(ie).ANP)**2
 rrx=0.
 if(Element(ie).AMPY.ne.0.)
*      rrx=(Member(i).stress(jj+5)/Element(ie).AMPY)**2
 if(Element(ie).AMPZ.ne.0.)
*      rrx=rrx+(Member(i).stress(jj+6)/Element(ie).AMPZ)**2
 v(4)=rrxx+Dsqrt(rrx)
 write(iflz(5)) istat,(v(k),k=1,4)
 enddo
 if(ns.gt.2) then
 do j=3,ns
 istat = Member(i).d_stat(j)
 jj=6*(j-1)
  v(1)=Member(i).stress(jj+1)          ! 軸力
  v(2)=Member(i).stress(jj+5)          ! ｙ軸モーメント
  v(3)=-Member(i).stress(jj+6)         ! ｚ軸モーメント
 rrxx=0.                              ! 現在ダミー　塑性関数値
 if(Element(ie).ANP.ne.0.)
*      rrxx=(Member(i).stress(jj+1)/Element(ie).ANP)**2
 rrx=0.
 if(Element(ie).AMPY.ne.0.)
*      rrx=(Member(i).stress(jj+5)/Element(ie).AMPY)**2
 if(Element(ie).AMPZ.ne.0.)
*      rrx=rrx+(Member(i).stress(jj+6)/Element(ie).AMPZ)**2
 v(4)=rrxx+Dsqrt(rrx)
 write(iflz(5)) istat,(v(k),k=1,4)
 enddo
 endif
 endif
 enddo
 endif
 return
 end
```

このサブルーチンから分かるように、新規の部材モデル番号は 51-70 であることから、else 以下の処理となる。構造体成分 Member(i).stress に適切なデータがセットされていれば、このプログラムを変更しなくても良い。

次に、断面内のファイバーの応力とひずみを出力するサブルーチンについて検討しよう。サブルーチン Out_Fiber()で、関連する部分を抜き出して表示する。ここでは、部材モデルごとに出力しており、新たな部

材モデルに対しては、該当する部分を作る必要がある。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Out_Fiber
C       ―――――――――――――――――――――――――――――――――
C     ●   部材の断面応力を出力(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Out_Fiber(Member,Element,E_model11,M_model11,
     *            E_model12,M_model12,E_model13,M_model13,
     *            E_model15,M_model15,
     *            E_model21,M_model21,E_model22,M_model22,
     *            E_model31,M_model31,E_model32,M_model32,
     *            E_model33,M_model33,
     *            E_model_fiber,M_model_fiber,
     *            n_member,ifl,iflz,i_print,Out_section,
     *      S_comp_model, E_modelx, M_modelx,
     *      E_fiber_work, M_fiber_work)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
c―――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
      record / Member_s       / Member
      record / Element_s      / Element
      record / Out_section_s  / Out_section
      record / M_model11_s    / M_model11
      record / E_model11_s    / E_model11
      record / M_model12_s    / M_model12
      record / E_model12_s    / E_model12
      record / M_model13_s    / M_model13
      record / E_model13_s    / E_model13
      record / M_model15_s    / M_model15
      record / E_model15_s    / E_model15
      record / M_model21_s    / M_model21
      record / E_model21_s    / E_model21
      record / M_model22_s    / M_model22
      record / E_model22_s    / E_model22
      record / M_model31_s    / M_model31
      record / E_model31_s    / E_model31
      record / M_model32_s    / M_model32
      record / E_model32_s    / E_model32
      record / M_model33_s    / M_model33
      record / E_model33_s    / E_model33
      record / M_model_fiber_s   / M_model_fiber
      record / E_model_fiber_s   / E_model_fiber
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension ifl(16),iflz(16)
      dimension Member(*),Element(*),M_model11(*),E_model11(*),
```

```
      *          M_model12(*),E_model12(*),M_model13(*),E_model13(*),
      *          M_model15(*),E_model15(*)
       dimension M_model21(*),E_model21(*),M_model22(*),E_model22(*)
       dimension M_model31(*),E_model31(*),M_model32(*),E_model32(*)
       dimension M_model33(*),E_model33(*)
       dimension mxtype(100),myytype(4)
       dimension E_modelx(*),M_modelx(*),S_comp_model(*)
       dimension E_fiber_work(*),M_fiber_work(*)
       data mxtype/1,1,1,1,1,1,1,1,1,1, 1,3,1,3,1,1,3,3,3,1,
      3            1,3,1,1,1,1,1,1,1,1, 1,3,1,1,1,1,1,1,1,1,
      5            1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,
      7            1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1,
      9            1,1,1,1,1,1,1,1,1,1, 1,1,1,1,1,1,1,1,1,1/
       data myytype/2,4,3,5/
       real*4    v(100),vf(3)
c      i_print         :integer 出力制御変数 0:ファイル出力あり
c-------------------------------★応力 5
      if(i_print.ne.0) return
      if(ifl(6).eq.0) return
c-------------------------------★断面応力出力
      if(Out_section.n_member.eq.0) return
      do j=1,Out_section.n_member
      i=Out_section.no_member(j)
c-------------------------------★断面応力
      ie = Member(i).nm_element
      imm= Element(ie).n_element              ! 要素番号
      immm= Member(i).n_model_type            ! モデルタイプ別番号
      iet  = Member(i).element_type
      if(iet.eq.11) then
c-----------------------★モデル１１
       .
      elseif(iet.eq.12) then
c-----------------------★モデル１２
       .
      elseif(iet.eq.15) then
c-----------------------★モデル１５
       .
      elseif(iet.eq.13) then
c-----------------------★モデル２１
       .
      elseif(iet.eq.14) then
c-----------------------★モデル２２
       .
      elseif(iet.eq.16) then
c-----------------------★モデル３１
       .
      elseif(iet.eq.17) then
c-----------------------★モデル３２
       .
      elseif(iet.eq.18.or.iet.eq.19) then
c-----------------------★モデル１３
       .
      elseif((iet-1)/10.eq.5.or. (iet-1)/10.eq.6) then
c-----------------------★新規モデル 51-70
```

```
      i_comp= iet -  50                                                  ! 1
      n_out_stress  = S_comp_model(i_comp).n_out_stress
      nmmx= Member(i).n_model_type - 1                  ! M_Fiber_work の開始番号   2
      nmx = Element(ie).n_section(1) - 1                ! E_Fiber_work の開始番号
      do m=1, n_out_stress
      kk = S_comp_model(i_comp).nm_out_stress(m)         ! 出力エレメント番号       3
      if(kk.ne.0) then
      immx = E_Fiber_work(nmx+kk).nm_section             ! エレメント番号           4
      immmx= M_Fiber_work(nmmx+kk).nm_section            ! 内部エレメント番号
      nm_div=E_modelx(immx).n_section                                    ! 5
      nnm=M_modelx(immmx).nm_section -  1
      do k=1,nm_div
      v(k)=M_model_fiber(k+nnm).d_stress_x                               ! 6
      enddo
      write(iflz(6)) (v(k),k=1,nm_div)                  ! 応力           ! 7
      do k=1,nm_div
      v(k)=M_model_fiber(k+nnm).d_eps_x                                  ! 8
      enddo
      vf(1) = M_modelx(immmx).d_epsi_x                   !  軸方向歪
      vf(2) = M_modelx(immmx).d_epsi_y                   !  y 軸に関する曲げ歪
      vf(3) = M_modelx(immmx).d_epsi_z                   !  z 軸に関する曲げ歪
      write(iflz(6))(vf(k),k=1,3),(v(k),k=1,nm_div)      ! ひずみ       ! 9
      endif
      enddo
      endif
c――――――――――――――――――――――――――――――★断面応力出力終わり
      enddo
      return
      end
```

このサブルーチンでは、部材モデル毎に処理が分けられている。新規に設計した静的縮合モデルに対して出力部分のコードが追加されている。右に書かれているコメント番号に従って説明する。特に階層構造となっている構造体へのアクセス方法をここで理解されたい。次節以降で説明する他の処理でも同様の方法を用いてワーク用構造体にアクセスする。

1. 部材モデル番号が 51-70 であるかどうかチェックする。そうであれば、以下の出力処理を行う。新規部材モデル番号からモデル設定用構造体の番号を求め、その番号より部材の中で、応力とひずみを出力するエレメント数を取得する。
2. 構造体 M_Fiber_work の先頭番地と、同じく構造体 E_Fiber_work の先頭番地を取得する。
3. 応力とひずみを出力するエレメント数分、以下の処理を行う。まず、そのエレメント番号 kk を取得する。そのエレメント番号が 0 の場合は出力処理を行わない。

4. 構造体 M_Fiber_work の先頭番地と、同じく構造体 E_Fiber_work より、構造体 E_model と M_model の該当番地を取得する。
5. 構造体の成分 E_model(immx).n_section より、その断面のファイバー総数を取得し、同様に構造体の成分 M_model(immmx).nm_section より、該当するファイバーの番地を取得する。さらに、以下の処理をファイバー数分行う。
6. ファイバー軸方向応力を出力用の単精度配列 v にセットする。
7. 当該のファイバー応力をファイルに出力する。
8. ファイバー軸方向ひずみを出力用の単精度配列 v にセットする。さらに、断面に関する軸方向ひずみと y 軸、z 軸に関する曲げひずみを出力用の単精度配列 vf にセットする。
9. 当該断面に関するひずみをファイルに出力する。

9.4.6 線形剛性

本節では、新しい静的縮合モデルを SPACE に組み込むために、必要となるサブルーチンについて説明する。この必要となるサブルーチンは

１）線形剛性を求めるサブルーチン
２）非線形剛性を求めるサブルーチン
３）応力を求めるサブルーチン
４）応力をチェックし、接線剛性を求めるサブルーチン

である。以降の節から部材モデルに関する上記４つのサブルーチンを検討しよう。最初は、線形剛性について考える。階層が非常に深くなっているので注意して欲しい。また、先に示した構造体のリンク手法がここで使われている。この手法は後から説明する３つのサブルーチンでも同様に使用されている。ここでは、特にこの構造体のリンク手法と部材モデルの階層構造について理解されたい。線形の剛性行列を求めるサブルーチン Cal_stiff_linear()では、太文字で示した部分がコードを追加する箇所となっている。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stiff_linear
C       ―――――――――――――――――――――――――――――――
C     ●   線形剛性行列の計算(ok)
C     ―――――――――――――――――――――――――――――――
      subroutine 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     ―――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record / parameter_s / Parameter_C
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
      dimension Member(*),Element(*)
      dimension ak_linear(12,12,*),cosin(2,32)
c―――――――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)
      dimension E_fiber_work(*),M_fiber_work (*)
       ・
c―――――――――――――――――――――――――――――――――――――――――――――――――
      n_member = Parameter_C.N_member
      do i=1,n_member
      mem = i
      iet  = Member(i).element_type
      ie = Member(i).nm_element
      iett=(iet-1)/10
      ien= Member(i).n_model_type
      if(Member(i).nm_dll_element .ne. 0) goto 9999   ! DLL 要素
      if(iett.eq.0)then
      goto(11,12,13,14,15,16,17,18,19,20),iet
   11 continue
       ・
       ・
c―――――――――――――――――――――――――――――――――――――★Model_No.1 通常の有限要素弾性モデル
      goto 100
      elseif(iett.eq.1)then
      goto(111,112,113,114,115,116,117,118,119,120),iet-10
  111 continue
       ・
       ・
c―――――――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素静的縮合モデル
```

```
      goto 100
      elseif(iett.eq.5.or. iett.eq.6)then
      call Cal_lin_stiff_Mxx(Model_type,Member(i),Element(ie),
     *      ak_linear(1,1,i),E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,
     *      S_comp_model,E_fiber_work,M_fiber_work)
      goto 100
      endif
      goto 100
 9999 continue
  100 continue
      end do
      return
      end
```

このサブルーチンでは、部材モデルで階層構造を構成している。新規部材モデルを追加したため、その対応を取る必要がある。ここでは、新規モデルで必要となる５つの構造体を仮引数とし、それらの構造体の定義文が保存されているヘッダーファイルNew_submain.hをインクルードする。部材モデルで階層となっているため、新規モデルのコード番号51-70 までを条件として、サブルーチン Cal_lin_stiff_Mxx()をコールする。このとき、サブルーチンの引数として 5 つの構造体が引き渡す。計算が終了し、このサブルーチンから戻るときこの新規モデルの線形剛性が受け渡される。

新規静的縮合モデルの線形剛性を計算するサブルーチンを示す。このサブルーチンは、新たに設計しなければならないが、その内容は他の静的縮合モデルとほとんど同じであり、作成することは容易である。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_lin_stiff_Mxx
C       ―――――――――――――――――――――――――――――――――
C     ●   代表的な部材モデルの剛性 任意要素
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_lin_stiff_Mxx(Model_type,Member,Element,ak,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,
     *      S_comp_model,E_fiber_work,M_fiber_work)                    ! 1
C     ―――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"                                          ! 2
      record / member_s          / Member
      record / element_s         / Element
      record / n_model_s         / Model_type
      record / E_model_fiber_s   / E_model_fiber
```

```
      record / M_model_fiber_s    / M_model_fiber
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
c―――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s     / S_comp_model                              ! 3
      record / E_modelx_s         / E_modelx
      record / M_modelx_s         / M_modelx
      record / E_fiber_work_s     / E_fiber_work
      record / M_fiber_work_s     / M_fiber_work
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)                       ! 4
      dimension E_fiber_work(*),M_fiber_work(*)
      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
      ix_model= iet-50                            ! 任意要素モデル(51-69)
      nmmx= Member.n_model_type - 1               ! M_Fiber_work の開始番号      ! 5
      nmx = Element.n_section(1) - 1              ! E_Fiber_work の開始番号
      n_div= Element.n_section(2)                 ! 部材分割数
      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_modelx_dat(irest_Point,n_if,n_div,iubw,                        ! 6
     *         Element,Member,S_comp_model(ix_model),
     *         n_type,alength,
     *  Member.i_rigid_length,    ! i 端剛域
     *  Member.j_rigid_length)    ! j 端剛域
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―――――――――――――――――――――――――――――――★ワークベクトルのゼロクリア
```

```
      do i=1,n_div                                                    ! 7
      M_Fiber_Work(nmmx+i).an_stress=0.
      M_Fiber_Work(nmmx+i).an_vv=0.
      M_Fiber_Work(nmmx+i).an_ww=0.
      do j=1,6
      M_Fiber_Work(nmmx+i).ff_ip(j) = 0.
      enddo
      enddo
c―――――――――――――――――――――――――――――――★部材剛性行列の作成
      do i=1,n_div
         if(E_Fiber_work(nmx+i).n_Fiber_section.eq.0) then
            E_Fiber_work(nmx+i).nm_section=1
            M_Fiber_work(nmmx+i).nm_section=1
         endif
           imm = E_Fiber_work(nmx+i).nm_section          ! エレメント番号        ! 8
           immm= M_Fiber_work(nmmx+i).nm_section         ! 内部エレメント番号    ! 9
      call Stiff_Mx_I(i,n_type(i),akk,Member,alength,
     *       Model_type,Element,
     *       E_modelx(imm), E_model_fiber,                              ! 10
     *       M_modelx(immm), M_model_fiber,
     *       Bilinear_work,Trilinear_work,Concrete_work)
      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
```

ここでは、部材モデル内の各エレメントの線形剛性を求め、さらに両端の節点変位に対応する剛性行列を、静的縮合を行うことによって求める。各部材内エレメントの線形剛性を求めるサブルーチンは、Stiff_Mx_I()である。

このサブルーチンは、他の静的縮合モデルとほとんど同じであり、太文字部分が追加・変更する箇所である。プログラムの右側のコメント番号にしたがって説明する。

1. 新しく設計した構造体５つが引数として受け渡される。
2. 新しく設計した構造体の定義文がヘッダーファイルに保持されており、そのファイル New_submain.h をインクルードしている。
3. 構造体の割付を行う。
4. 新しい構造体を整合配列として宣言する。

5. 構造体 M_Fiber_work と E_Fiber_work の開始番号をセットする。また、この部材内部のエレメント数をセットする。
6. 先に設計した新規の任意型静的縮合モデルを設定するサブルーチン set_modelx_dat()をコールし、モデル情報を取得する。
7. この新規モデルが静的縮合を行う際、必要となる内部節点とエレメントに関するワーク領域をゼロクリアする。
8. 部材内部のエレメント番号を構造体 E_Fiber_work().nm_section から取得する。ただし、この値が 0 の場合は 1 をセットする。これは、値 0 を用いると構造体の配列で設定外を指定する可能性があるためである。
9. 部材内部の部材番号を構造体 M_Fiber_work().nm_section から取得する。ただし、この値が 0 の場合は 1 をセットする。これは、前記と同様に値 0 を用いると構造体の配列で設定外を指定する可能性があるためである。
10. 線形の剛性行列を計算するサブルーチン Stiff_Mx_I()をコールする。そのとき、引数として２つの構造体を受け渡す。その配列番号は、8、9 で求められている。

次に、線形の剛性行列を計算するサブルーチンを検討しよう。このサブルーチンでも階層構造が見られる。これは、部材内のエレメントがこの階層の中から任意に選択可能であることを示す。今後、ここにエレメントの新規モデルを追加することで、さらに、この部材モデルの応用範囲が拡大する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Stiff_Mx_I
C       ―――――――――――――――――――――――――――――――――
C     ●   線形剛性行列の計算
C     ―――――――――――――――――――――――――――――――――
      subroutine Stiff_Mx_I(im,n_type,ak,Member,alength,
     *     Model_type,Element,
     *     E_modelx,E_model_fiber,M_modelx,M_model_fiber,
     *     Bilinear_work,Trilinear_work,Concrete_work)
C     ―――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record / member_s          / Member
      record / element_s         / Element
      record / n_model_s         / Model_type
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
```

```
      record / E_model_fiber_s    / E_model_fiber
      record / M_model_fiber_s    / M_model_fiber
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
      dimension ak(12,12),alength(*)
      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_lin_stiff_Mx(Member,Element,
     *      ak,alength(im) )
      goto 100
   12 continue
c―――――――――――――――――――――――――――――――――★ファイバー要素
      call Fiber_Model_GIx(ak,alength(im),Member,Element,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work)
      goto 100
   13 continue
c―――――――――――――――――――――――――――――――――★MS 要素
      call MS_Model_GIx(ak,alength(im),Member,Element,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work)
      goto 100
   14 continue
c―――――――――――――――――――――――――――――――――★アナロジー要素
      call Analogy_Model_GIx(ak,Member,Element,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber)
      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
```

ここでは、各種の断面モデルに対応して線形の剛性行列を計算サブルーチンが階層構造となって並んでいる。後は、これらのサブルーチンを設計して追加すれば良い。これらのサブルーチンは、他の部材モデルで使用されたものとほとんど同じであるため作成することは容易である。ここでは、ファイバーモデルの初期設定と線形剛性行列を計算するサブルーチン Fiber_Model_GIx()について示す。この中にも各ファイバーに対する履歴特性が下位の階層として構成されている。実際に剛性行列を作成するサブルーチン Fiber_Model_Gx()は次節で示す。

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Fiber_Model_GIx
C     ――――――――――――――――――――――――――――――――――
C     ●   線形剛性行列の計算(ok)
C     ――――――――――――――――――――――――――――――――――
C     ●   Model_No.51-70
C     ――――――――――――――――――――――――――――――――――
      subroutine Fiber_Model_GIx(ak,alength,Member,Element,
     *           E_modelx,E_model_fiber,M_modelx,M_model_fiber,
     *           Bilinear_work,Trilinear_work,Concrete_work)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record / member_s           / Member
      record / element_s          / Element
      record / E_modelx_s         / E_modelx
      record / E_model_fiber_s    / E_model_fiber
      record / M_modelx_s         / M_modelx
      record / M_model_fiber_s    / M_model_fiber
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension Bilinear_work(*),Trilinear_work(*)
      dimension Concrete_work(*)
      dimension ak(12,12)
c――――――――――――――――――――――――――――――――――★
      nm_div = E_modelx.n_section
      nn     = E_modelx.nm_section - 1
      nnm    = M_modelx.nm_section - 1
      do i=1,nm_div
      nn     =  nn  + 1
      nnm    =  nnm + 1
c――――――――――――――――――――――――――――――――――★データの初期設定
```

```
      M_model_fiber(nnm).d_E        = E_model_fiber(nn).E_1
      M_model_fiber(nnm).i_elastic  = 0          !  ファイバー要素の状態（弾性の場合は０：塑性は１）
      M_model_fiber(nnm).d_eps_x    = 0.         !  軸方向歪
      M_model_fiber(nnm).d_stress_x = 0.         !  軸方向応力
c──────────────────────★ファイバーデータセット
      E_model_fiber(nn).Ary  = E_model_fiber(nn).A*E_model_fiber(nn).ry
      E_model_fiber(nn).Arz  = E_model_fiber(nn).A*E_model_fiber(nn).rz
      E_model_fiber(nn).Ary2 =
     *                       E_model_fiber(nn).Ary*E_model_fiber(nn).ry
      E_model_fiber(nn).Arz2 =
     *                       E_model_fiber(nn).Arz*E_model_fiber(nn).rz
      E_model_fiber(nn).Aryz =
     *                       E_model_fiber(nn).Arz*E_model_fiber(nn).ry
      nmx                    = M_model_fiber(nnm).n_type
      nm_type                = E_model_fiber(nn).nm_type
      goto ( 10,20,30,30,10,20,10,20),nm_type
 10   continue
c──────────────────────★バイリニア型（スチール用）
      Bilinear_work(nmx).i_stat  = -1              !  ファイバー要素の状態
      goto 100
 20   continue
c──────────────────────★非対称トリリニア型
      Trilinear_work(nmx).i_stat = -1              !  ファイバー要素の状態
      goto 100
 30   continue
c──────────────────────★コンクリート型
      Concrete_work(nmx).i_stat  = -1              !  ファイバー要素の状態
      goto 100
 100  continue
      enddo
c────────────────────────────────★ファイバー要素のデータセット
      nm_div = E_modelx.n_section
      nn     = E_modelx.nm_section - 1
      nnm    = M_modelx.nm_section - 1
      ra     = 0.
      ray    = 0.
      raz    = 0.
      raz2   = 0.
      ray2   = 0.
      rayz   = 0.
      gg     = 0.
      aa     = 0.
      do i=1,nm_div
      nn     = nn+1
      nnm    = nnm+1
      A      = E_model_fiber(nn).A
      E      = M_model_fiber(nnm).d_E
      ra     = ra   + E*A
      ray    = ray  + E*E_model_fiber(nn).Arz
      raz    = raz  + E*E_model_fiber(nn).Ary
      ray2   = ray2 + E*E_model_fiber(nn).Arz2
      raz2   = raz2 + E*E_model_fiber(nn).Ary2
      rayz   = rayz + E*E_model_fiber(nn).Aryz
      aa     = aa   + A
```

```
      gg     = gg   + A*E_model_fiber(nn).G
      enddo
      gg     = gg / aa
      M_modelx.d_aa     = aa              !  断面積の和
      M_modelx.d_ra     = ra              !  E*断面積の和
      M_modelx.d_ray    = ray             !  E*A*z
      M_modelx.d_raz    = raz             !  E*A*y
      M_modelx.d_raz2   = raz2            !  E*A*y*y
      M_modelx.d_ray2   = ray2            !  E*A*z*z
      M_modelx.d_rayz   = rayz            !  E*A*z*y
      M_modelx.d_gg     = gg              !  G*A
      M_modelx.d_epsi_x =  0.             !  軸方向歪
      M_modelx.d_epsi_y =  0.             !  y 軸に関する曲げ歪
      M_modelx.d_epsi_z =  0.             !  z 軸に関する曲げ歪
c―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★初期剛性行列の計算
      call Fiber_Model_Gx(ak,alength,Member,Element,
     *      E_modelx,E_model_fiber,M_modelx,M_model_fiber)
      return
      end
```

9.4.7 非線形剛性

本節では、非線形剛性について考える。基本的には、線形の剛性行列を計算するサブルーチンと同じである。非線形剛性を求めるサブルーチン Get_nonlinear_stiff()を以下に示す。

```
C     ―――――――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_nonlinear_stiff
C       ―――――――――――――――――――――――――――――――――――――――――
C     ●   接線剛性行列の計算(ok)
C     ―――――――――――――――――――――――――――――――――――――――――
      subroutine Get_nonlinear_stiff(N_analysis,
     *     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)
C     ―――――――――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
```

```
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
c―――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)
      dimension E_fiber_work(*),M_fiber_work (*)
       ・
       ・
C     ―――――――――――――――――――――――――――――――――――
      if(N_analysis.eq.7) return     !   線形解析;7
      do  9999 i=1,n_member
      if(Member(i).nm_analysis .eq. -1) goto 9999 ! 各部材の解析種別のチェック(-1:線形解析)
c―――――――――――――――――――――――――――――――――★部材両端の変位取得
      do j=1,12
      ires=Member(i).irest(j)
      if(ires.gt.0) then
      v(j)=past_disp_point(ires)
      else
      v(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)
c―――――――――――――――――――――――――――――――――★要素及びモデルのセット
      mem = i
      iet  = Member(i).element_type
      iett=(iet-1)/10
      ie = Member(i).nm_element
      im = Element(ie).n_element
      imm = Member(i).n_element_type
      ien= Member(i).n_model_type
      if(Member(i).nm_dll_element .ne. 0)  goto 9998   ! DLL 要素
      if(iett.eq.0)then
      goto(11,12,13,14,15,16,17,18,19,20),iet
   11 continue
       ・
       ・
c―――――――――――――――――――――――――――――――――★Model_No.1 通常の有限要素弾塑性モデル
      goto 100
      elseif(iett.eq.1)then
      goto(111,112,113,114,115,116,117,118,119,120),iet-10
  111 continue
       ・
       ・
c―――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素静的縮合モデル
      goto 100
      elseif(iett.eq.5.or. iett.eq.6)then
      call Cal_nonlin_stiff_Mxx(N_analysis,
     *      Model_type,Member(i),Element(ie),
```

```
     *      ak,ier,E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      S_comp_model,E_fiber_work,M_fiber_work)
      goto 100
      endif
      goto 100
 9998 continue
  100 continue
c――――――――――――――――――――――――――――――――――――★部材の接線剛性を釣合系に変換
      call Rotate_K(ak,rot_memb(1,1,1,i),
     *              rot_memb(1,1,2,i),ak_nonlinear(1,1,i))
 9999 continue
      return
      end
```

このサブルーチンでも、部材モデルで階層構造を構成している。新規部材モデルを追加したため、その対応を取る必要がある。ここでは、新規モデルで必要となる５つの構造体を仮引数とし、それらの構造体の定義文が保持されているヘッダーファイルNew_submain.hをインクルードする。部材モデルで階層となっているため、新規モデルのコード番号51-70 までを条件として、サブルーチン Cal_nonlin_stiff_Mxx()をコールする。このとき、サブルーチンの引数として5つの構造体を引き渡す。計算が終了し、このサブルーチンから戻るときこの新規モデルの接線剛性行列が受け渡される。

次に、接線剛性行列を求めるサブルーチン Cal_nonlin_stiff_Mxx()を示す。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_nonlin_stiff_Mxx
C       ――――――――――――――――――――――――――――――――――――
C     ●   代表的な部材モデルの接線剛性
C     ――――――――――――――――――――――――――――――――――――
      subroutine Cal_nonlin_stiff_Mxx(N_analysis,
     *            Model_type,Member,Element, ak,ier,
     *     E_modelx,E_model_fiber,M_modelx,M_model_fiber,
     *     S_comp_model,E_fiber_work,M_fiber_work)
C     ――――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / E_model_fiber_s   / E_model_fiber
      record / M_model_fiber_s   / M_model_fiber
      dimension E_model_fiber(*),M_model_fiber(*)
c――――――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
```

```
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)
      dimension E_fiber_work(*),M_fiber_work (*)
      dimension ak(12,12),akk(12,12)
      real*8,  ALLOCATABLE :: c(:,:),ab(:,:),ba(:),alength(:),EA(:)
      integer, ALLOCATABLE :: irest_Point(:,:),n_type(:)
C     ――――――――――――――――――――――――――――――――――――
      iet = Member.n_model                          ! モデルタイプ番号
      ix_model= iet-50                              ! 任意モデル(51-69)
      nmmx= Member.n_model_type - 1                 ! M_Fiber_work の開始番号
      nmx = Element.n_section(1) - 1                ! E_Fiber_work の開始番号
      n_div= Element.n_section(2)                   ! 部材分割数
      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),EA(n_div)
     *         )
c―――――――――――――――――――――――――――――――――★節点拘束表の作成
c                                                   未知数等をセット
      call set_modelx_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,S_comp_model(ix_model),
     *         n_type,alength,
     *  Member.i_rigid_length,    ! i 端剛域
     *  Member.j_rigid_length)    ! j 端剛域
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
      EAx=Element.A*Element.E
c―――――――――――――――――――――――――――――――――★部材剛性行列の作成
      do i=1,n_div
      imm = E_Fiber_work(nmx+i).nm_section          ! 要素番号
      immm= M_Fiber_work(nmmx+i).nm_section         ! 内部要素番号
      EA=EAx
      if(n_type(i).eq.2) then       ! ファイバー：他のモデルでも必要なときはここに加える
```

```
      EA=M_modelx(immm).d_ra
      endif
      call Stiff_Mx(i,n_type(i),akk,Member,alength,
     *       Model_type,Element,
     *       E_modelx(imm), E_model_fiber,
     *       M_modelx(immm), M_model_fiber)
      if(N_analysis.ne.9.and.N_analysis.ne.7.and.n_type(i).le.3) then        !  幾何学的非線形剛性
      call Cal_geomet_stiffx(M_Fiber_Work(nmmx+i).an_stress,Member,akk,alength(i))
      call Create_Kn(akk, M_Fiber_Work(nmmx+i).an_vv, M_Fiber_Work(nmmx+i).an_ww,
     *              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,EA
     *        )
      return
      end
```

接線剛性行列を求めるサブルーチン Cal_nonlin_stiff_Mxx()において、前節と同様に、階層構造を有する構造体にアクセスしている。下位層の構造体 E_modelx と M_modelx までのリンクは、前節で説明しており、ここでも再度確認されたい。ここでは、幾何学的非線形剛性を作成するサブルーチンで必要となる各エレメントの軸方向剛性 EA、軸力 N、y 方向と z 方向の相対変位について考えよう。まず、軸方向剛性 EA は、断面内に塑性域が生じると変化する。そこで、エレメントがファイバーモデルの場合、他のサブルーチンで計算した当該エレメントの剛性と断面積の積を表す M_modelx(immm).d_ra を軸剛性として使用する。また、そのエレメントの軸力、y 方向と z 方向の相対変位も他のサブルーチンでセットした構造体 M_Fiber_Work の各成分から取得する。これらの値の設定は、次節で説明する。最後に、この幾何学的非線形剛性は、部材長さを有するエレメントモデルに適用されて求められる。そこで、新規のエレメントモデルを組み込む場合はこの点に注意されたい。

次は、弾塑性剛性行列を求めるサブルーチン Stiff_Mx()について以下に示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Stiff_Mx
C       ―――――――――――――――――――――――――――――――――
C     ●   接線剛性行列の計算(ok)
```

```
C     ―――――――――――――――――――――――――――――――――
      subroutine Stiff_Mx(im,n_type,ak,Member,alength,
     *       Model_type,Element,
     *       E_modelx,E_model_fiber,M_modelx,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_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      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_lin_stiff_Mx(Member,Element,ak,alength(im) )
      goto 100
   12 continue
c―――――――――――――――――――――――――――――――★ファイバー要素
      it=it+1
      call Fiber_Model_Gx(ak,alength(im),Member,Element,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber)
      goto 100
   13 continue
c―――――――――――――――――――――――――――――――★MS 要素
      call MS_Model_Gx(ak,alength(im),Member,Element,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber)
      goto 100
   14 continue
c―――――――――――――――――――――――――――――――★アナロジー要素
      iep=M_model_fiber(nnm).i_elastic
      call Analogy_Model_Gx(iep,ak,Member,Element,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,nn)
      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
```

弾塑性の剛性行列を求めるサブルーチン Fiber_Model_Gx()について以下に示す。この中のサブルーチン Fiber_GX()は既に第５章で説明したのでそちらを参照されたい。

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Fiber_Model_Gx
C     ――――――――――――――――――――――――――――――――――
C     ●   接線剛性行列の計算
C     ――――――――――――――――――――――――――――――――――
C     ●   Model_No.51
C     ――――――――――――――――――――――――――――――――――
      subroutine Fiber_Model_Gx(ak,alength,Member,Element,
     *            E_modelx,E_model_fiber,M_modelx,M_model_fiber)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record / member_s          / Member
      record / element_s         / Element
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / M_model_fiber_s   / M_model_fiber
      record / E_model_fiber_s   / E_model_fiber
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension ak(12,12)
c――――――――――――――――――――――――――――――★
      rix  = Element.RIx
      asy  = Element.ASy
      asz  = Element.ASz
c――――――――――――――――――――――――――――――★
      call Fiber_GK(ak,alength,
     *     M_modelx.d_ra, M_modelx.d_ray, M_modelx.d_raz,
     *               M_modelx.d_raz2,M_modelx.d_ray2,
     *               M_modelx.d_rayz,M_modelx.d_gg,
     *                 aa,rix,asy,asz)
      return
      end
```

9.4.8 部材の応力チェック

本節では、部材の弾塑性チェックを行い、部材内エレメンの力ー変位の履歴、あるいはファイバーの応力ーひずみ履歴を追跡する。また、その時点での部材接線剛性を求める。まず、部材の弾塑性チェックを行うサブルーチン Check_stress()を検討しよう。太文字部分がコードを追加する箇所である。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Check_stress
C     ―――――――――――――――――――――――――――――――――
C     ●   部材の塑性状態をチェックする(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Check_stress(Control,N_analysis,
     *     ak_nonlinear,Member,n_member,
     *     Model_type,Element,past_disp_point,disp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
     *     E_model21, M_model21,
     *     E_model22, M_model22,
     *     E_model31, M_model31,
     *     E_model32, M_model32,
     *     E_model33, M_model33,
     *     MSS_work,
     *     Bilinear_work,Trilinear_work,Concrete_work,RO_work,
     *     work1_elememt,work2_elememt, work1_member, work2_member ,
     *     S_comp_model,E_modelx,M_modelx,
     *     E_fiber_work,M_fiber_work)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record /control_s    / Control
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
      c――――――――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)
      dimension E_fiber_work(*),M_fiber_work (*)
       .
       .
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
      v(j)=disp_point(ires)
      vp(j)=past_disp_point(ires)
      else
      v(j)=0.
      vp(j)=0.
      endif
      enddo
c―――――――――――――――――――――――――――――――――★部材両端の節点力のゼロセット
      do j=1,12
      f(j)=0.
      enddo
c―――――――――――――――――――――――――――――――――★変位を釣合系から部材座標系に変換
      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
      ie   = Member(i).nm_element
      im   = Element(ie).n_element
      ien  = Member(i).n_model_type
      if(Member(i).nm_dll_element .ne. 0)  goto 9999   ! DLL 要素
      if(iett.eq.0)then
      goto(11,12,13,14,15,16,17,18,19,20),iet
   11 continue
      ・
      ・
c―――――――――――――――――――――――――――――――――★Model_No.1 通常の有限要素弾塑性モデル
      goto 100
      elseif(iett.eq.1)then
      goto(111,112,113,114,115,116,117,118,119,120),iet-10
  111 continue
      ・
      ・
c―――――――――――――――――――――――――――――――――★Model_No.11 両端ファイバーモデル
      goto 100
      elseif(iett.eq.5.or. iett.eq.6)then
      call Cal_check_stiff_Mx(Control,N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vpp,f,
     *      S_comp_model,E_fiber_work,M_fiber_work)
c―――――――――――――――――――――――――――――――――★部材の両端節点力を釣合系に変換
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      do j=1,12
      Member(i).force(j)=Member(i).force(j)+ff(j)
      enddo
      goto 100
```

```
      endif
      goto 100
 9999 continue
c――――――――――――――――――――――――――――――――――★部材の接線剛性を釣合系に変換
  100 continue
      end do
      return
      end
```

このサブルーチンにおいても、部材モデルに関する階層構造となっている。適切な階層で、新規部材モデルに対する新しいサブルーチンをコールし、対応する新規のサブルーチン Cal_check_stiff_Mx()を作成する必要がある。このサブルーチンを以下に示す。この中で、引数として新たな構造体を受け渡す。

```
C     ――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_check_stiff_Mx
C       ――――――――――――――――――――――――――――――――――――
C     ●   代表的な部材モデルの塑性チェック
C     ――――――――――――――――――――――――――――――――――――
      subroutine Cal_check_stiff_Mx(Control,N_analysis,
     *      mem_x,Model_type,Member,Element,
     *      E_modelx, E_model_fiber,
     *      M_modelx, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vp,f_p,
     *      S_comp_model,E_fiber_work,M_fiber_work)
C     ――――――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / E_model_fiber_s    / E_model_fiber
      record / M_model_fiber_s    / M_model_fiber
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension ak(12,12),f_p(12)
      dimension vv(12),vp(12),vvx(12)
      dimension Bilinear_work(*),Trilinear_work(*)
      dimension Concrete_work(*)
      c――――――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)
      dimension E_fiber_work(*),M_fiber_work (*)
```

```
      real*8, ALLOCATABLE :: c(:,:),ab(:,:),alength(:),EA(:)
      real*8, ALLOCATABLE :: bav(:),akk(:,:,:),f1(:),f2(:)
      integer, ALLOCATABLE :: irest_Point(:,:),n_type(:)
C     ――――――――――――――――――――――――――――――――――――
      iet = Member.n_model                          ! モデルタイプ番号
      ix_model= iet-50                              ! 任意要素モデル(51-69)
      nmmx= Member.n_model_type - 1                 ! M_Fiber_work の開始番号
      nmx = Element.n_section(1) - 1                ! E_Fiber_work の開始番号
      n_div= S_comp_model(ix_model).n_div_element   ! 部材分割数
      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),EA(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_modelx_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,S_comp_model(ix_model),
     *         n_type,alength,
     *  Member.i_rigid_length,    ! i 端剛域
     *  Member.j_rigid_length)    ! j 端剛域
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),f2(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
      EAx=Element.A*Element.E
```

```
c―――――――――――――――――――――――――――――――★部材剛性行列の作成
      do i=1,n_div
      imm = E_Fiber_work(nmx+i).nm_section              ! エレメント番号
      immm= M_Fiber_work(nmmx+i).nm_section             ! 内部エレメント番号
      EA=EAx
      if(n_type(i).eq.2) then       ! ファイバー：他のモデルでも必要なときはここに加える
      EA=M_modelx(immm).d_ra
      endif
c―――――――――――――――――――――――――――――――★内部部材の剛性行列の計算
      call Stiff_Mx(i,n_type(i),akk(1,1,i),Member,alength,
     *       Model_type,Element,
     *       E_modelx(imm), E_model_fiber,
     *       M_modelx(immm), M_model_fiber)
      if(N_analysis.ne.9.and.N_analysis.ne.7.and.n_type(i).ne.3) then    !  幾何学的非線形剛性
      call Cal_geomet_stiffx(M_Fiber_Work(nmmx+i).an_stress,Member,
     *                       akk(1,1,i),alength(i))
      call Create_Kn(akk(1,1,i), M_Fiber_Work(nmmx+i).an_vv, M_Fiber_Work(nmmx+i).an_ww,
     *              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)
      do i=0,n_if-1
      k=i/6
      j = i - (k)*6
      f2(i+1) = M_Fiber_work(nmmx+k+1).ff_ip(j+1)                                    ! 1
      enddo
      call Typical_member_v(c,ab,f2,
     *                      n_if,n_if,iubw,iubw,ier,vv,bav)
c―――――――――――――――――――――――――――――――★部材内部、両端節点力の計算
      do i=1,n_div
      call Typical_member_p_force(akk(1,1,i),irest_Point(1,i),
     *                            vv,bav ,f_p,f1)
      enddo
c―――――――――――――――――――――――――――――――★部材内部、両端節点力から不釣合力へ
      do i=0,n_if-1
      k=i/6
      j = i - (k)*6
      M_Fiber_work(nmmx+k+1).ff_ip(j+1) = f1(i+1)                                    ! 2
      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
      do i=1,n_div
      imm = E_Fiber_work(nmx+i).nm_section            ! 要素番号
      immm= M_Fiber_work(nmmx+i).nm_section           ! 内部要素番号
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
      goto(11,12,13,14,15,16,17,18,19,20), n_type(i)                            ! 3
   11 continue
c―――――――――――――――――――――――――――★弾性要素
      goto 100
   12 continue
c―――――――――――――――――――――――――――★ファイバー要素
      call Fiber_checkx(Control,i,N_analysis,                                    ! 4
     *      mem_x,alength(i),Member,Element,
     *      E_modelx(imm),E_model_fiber,M_modelx(immm),M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vvx,
     *      M_Fiber_Work(nmmx+i).an_vv, M_Fiber_Work(nmmx+i).an_ww,
     *      S_comp_model (ix_model).nm_out_stress(i))
      goto 100
   13 continue
c―――――――――――――――――――――――――――★MS 要素
      call MS_checkx(N_analysis,
     *      mem_x,alength(i),Member,Element,
     *      E_modelx(imm),E_model_fiber,M_modelx(immm),M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vvx,
     *      M_Fiber_Work(nmmx+i).an_vv, M_Fiber_Work(nmmx+i).an_ww,
     *      S_comp_model (ix_model).nm_out_stress(i))

      goto 100
   14 continue
c―――――――――――――――――――――――――――★アナロジー要素
      call Analogy_checkx(N_analysis,
     *      mem_x,alength(i),Member,Element,
     *      E_modelx(imm),E_model_fiber,M_modelx(immm),M_model_fiber,vvx,
     *      M_Fiber_Work(nmmx+i).an_vv, M_Fiber_Work(nmmx+i).an_ww,
     *      S_comp_model (ix_model).nm_out_stress(i))
```

```
      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
c――――――――――――――――――――――――――――――★部材の軸力計算（幾何剛性作成用）
      call nonlinear_stress_N(akk(1,1,i),vvx,fnn)
      M_Fiber_Work(nmmx+i).an_stress = M_Fiber_Work(nmmx+i).an_stress + fnn        ! 5
c――――――――――――――――――――――――――――――★部材内部変位を足しこむ
      M_Fiber_Work(nmmx+i).an_vv = M_Fiber_Work(nmmx+i).an_vv +(vvx(8) - vvx(2))   ! 6
      M_Fiber_Work(nmmx+i).an_ww = M_Fiber_Work(nmmx+i).an_ww +(vvx(9) - vvx(3))
      enddo
c――――――――――――――――――――――――――――――★動的記憶領域の解放
      DEALLOCATE (
     *   c ,ab ,irest_point,n_type,alength,EA,
     *   bav,akk,f1,f2    )
      return
      end
```

このプログラム自身は作成しなければならないが、内容は他のプログラムとほぼ同じである。ここでは、他と異なる箇所を説明する。非線形剛性を求める部分は前節で説明した。その他の部分について解説する。

1. 前回の部材内節点に対応した不釣合力を配列 f1 にコピーする。
2. 新しく計算した不釣合力を、M_Fiber_work(nmmx+k+1).ff_ip(j+1)にコピーする。
3. ここでも、部材内のエレメントに階層構造が見られる。現在は、弾性はり、ファイバーモデル、MS モデル、アナロジーモデルが登録されている。
4. モデル番号 2 では、ファイバーモデルの履歴特性チェックを行うサブルーチン Fiber_checkx()をコールする。
5. エレメントの増分軸力を計算し、増分前の軸力に足しこむ。

6．エレメントの増分相対変位(v,w)を、増分前の相対変位に足しこむ。

ファイバーモデルの履歴を追跡するサブルーチン Fiber_checkx()を以下に示す。このサブルーチンには、各ファイバーの履歴特性に関するサブルーチンが階層構造となって構成されている。現在、SPACE に登録されているモデルは８つであり、今後履歴モデルを追加する場合は、ここに該当するサブルーチンを挿入することになる。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Fiber_checkx
C     ―――――――――――――――――――――――――――――――――
C     ●   ファイバー要素の材料非線形性チェックし、応力を計算
C     ―――――――――――――――――――――――――――――――――
      subroutine Fiber_checkx(Control,idiv,N_analysis,
     *      mem_x,Alength,Member,Element,
     *      E_modelx,E_model_fiber,M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,an_vv,an_ww,
     *       n_dstat)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record /control_s    / Control
      record / member_s          / Member
      record / element_s         / Element
      record / E_modelx_s        / E_modelx
      record / E_model_fiber_s   / E_model_fiber
      record / M_modelx_s         / M_modelx
      record / M_model_fiber_s   / M_model_fiber
      record / Bilinear_work_s   / Bilinear_work
      record / Trilinear_work_s  / Trilinear_work
      record / Concrete_work_s   / Concrete_work
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension Bilinear_work(*),Trilinear_work(*)
      dimension Concrete_work(*)
      dimension vv(12)
c―――――――――――――――――――――――――――――――★i 端部ファイバー
c―――――――――――――――――――――――――――――――★歪のセット
      d_epsi_x_1 =  (vv(7)  - vv(1)) / Alength   !  軸方向歪
      d_epsi_y_1 =  (vv(11) - vv(5)) / Alength   !  y軸に関する曲げ歪
      d_epsi_z_1 =  (vv(12) - vv(6)) / Alength   !  z軸に関する曲げ歪
      if(N_analysis.eq.10.or.N_analysis.eq.8) then                  ! 非線形軸方向歪
      d_epsi_x_1= d_epsi_x_1+strain_nonlinear(an_vv,
     *           an_ww,vv,Alength)
      endif
      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_modelx.d_epsi_x =  d_epsi_x_1 + M_modelx.d_epsi_x     !  軸方向歪
      M_modelx.d_epsi_y =  d_epsi_y_1 + M_modelx.d_epsi_y     !  y軸に関する曲げ歪
      M_modelx.d_epsi_z =  d_epsi_z_1 + M_modelx.d_epsi_z     !  z軸に関する曲げ歪
```

```
c―――――――――――――――――――――――――――――――★ファイバー要素のチェック
      nm_div   = E_modelx.n_section
      nn       = E_modelx.nm_section - 1
      nnm      = M_modelx.nm_section - 1
      iistat   = 0                              ! 塑性率を計算するための指標
      do i=1,nm_div
      nn       = nn  + 1
      nnm      = nnm + 1
      nmx      = M_model_fiber(nnm).n_type
      nm_type  = E_model_fiber(nn).nm_type
      du =  d_epsi_x_1 +                                        ! 軸方向歪
     *      d_epsi_y_1 * E_model_fiber(nn).rz -                 ! y 軸に関する曲げ歪
     *      d_epsi_z_1 * E_model_fiber(nn).ry                   ! z 軸に関する曲げ歪
      if(N_analysis.le.8.or.Member.nm_analysis.eq.-1) then
c―――――――――――――――――――――――――★弾性解析
c                                    ファイバー軸力計算
      M_model_fiber(nnm).d_stress_x = M_model_fiber(nnm).d_E*du +
     *                               M_model_fiber(nnm).d_stress_x
      else
      if(nm_type/10.eq.0) then
c―――――――――――――――――――――――――★弾塑性解析
      goto ( 10,20,30,40,50,60,70,80),nm_type
  10  continue
c―――――――――――――――――――――――――★バイリニア型（スチール用）
      call BiLinear(M_model_fiber(nnm).d_E,Bilinear_work(nmx).i_stat,
     *              E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *              E_model_fiber(nn).Q_1,du,
     *              M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1)
      if(Bilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      if(Bilinear_work(nmx).i_stat.ne.0) then
      endif
      goto 100
  20  continue
c―――――――――――――――――――――――――★対称トリリニア型 （スチール用）
      call TriLinear(M_model_fiber(nnm).d_E,Trilinear_work(nmx).i_stat,
     *               E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *               E_model_fiber(nn).E_3,
     *               E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *               du,M_model_fiber(nnm).d_stress_x,
     *               Trilinear_work(nmx).P1(1))
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 100
  30  continue
c―――――――――――――――――――――――――★コンクリート型
      call Concrete(M_model_fiber(nnm).d_E,Concrete_work(nmx).i_stat,
     *              E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *              E_model_fiber(nn).E_3,E_model_fiber(nn).Ec_3,
     *              E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *              E_model_fiber(nn).Ec_1,E_model_fiber(nn).Ec_2,
     *              du,  M_model_fiber(nnm).d_stress_x,
     *              Concrete_work(nmx).p1(1),Concrete_work(nmx).P1(2),
     *              Concrete_work(nmx).ipret,Concrete_work(nmx).P1(3),
     *              Concrete_work(nmx).p1(4),Concrete_work(nmx).P1(5),
     *              Concrete_work(nmx).ipre_c)
```

```
      if(Concrete_work(nmx).i_stat.ne.0.and.Concrete_work(nmx).i_stat
     *            .ne.3) iistat = iistat + 1
      goto 100
   40 continue
c-----------------------★曲線コンクリート型
      call Concrete_e(M_model_fiber(nnm).d_E,Concrete_work(nmx).i_stat,
     *             E_model_fiber(nn).E_1,E_model_fiber(nn).E_3,
     *             E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *             du,  M_model_fiber(nnm).d_stress_x,
     *             Concrete_work(nmx).P1(1),Concrete_work(nmx).P1(2),
     *             Concrete_work(nmx).ipret,Concrete_work(nmx).ipre_c,
     *             Concrete_work(nmx).P1(3),Concrete_work(nmx).P1(4),
     *             Concrete_work(nmx).P1(5),E_model_fiber(nn).Ec_1,
     *             Concrete_work(nmx).P1(6),E_model_fiber(nn).Ec_2)
      if(Concrete_work(nmx).i_stat.eq.0)then
      if(du.lt.E_model_fiber(nn).Ec_1)then
      iistat = iistat + 1
      endif
      elseif(Concrete_work(nmx).i_stat.ne.3)then
      iistat = iistat + 1
      endif
      goto 100
   50 continue
c-----------------------★バイリニア型（移動＋等方効果用）
      call BiLinear_h(M_model_fiber(nnm).d_E,Bilinear_work(nmx).i_stat,
     *             E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *             E_model_fiber(nn).Q_1,du,
     *             M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1,
     *             E_model_fiber(nn).Beta)
      if(Bilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 100
   60 continue
c-----------------------★対称トリリニア型（移動＋等方効果用）
      call TriLinear_h(M_model_fiber(nnm).d_E,
     *              Trilinear_work(nmx).i_stat,
     *              E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *              E_model_fiber(nn).E_3,
     *              E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *              du,M_model_fiber(nnm).d_stress_x,
     *             Trilinear_work(nmx).P1(1),Trilinear_work(nmx).P1(2),
     *             E_model_fiber(nn).Beta, E_model_fiber(nn).Beta_2)
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 100
   70 continue
c-----------------------★非対称バイリニア型
      call BiLinear_AS(M_model_fiber(nnm).d_E,Bilinear_work(nmx).i_stat,
     *             E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *             E_model_fiber(nn).Q_1,E_model_fiber(nn).Ec_1,
     *             E_model_fiber(nn).Ec_2,E_model_fiber(nn).Qc_1,du,
     *             M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1,
     *             Bilinear_work(nmx).P2,Bilinear_work(nmx).Sig_z,
     *             E_model_fiber(nn).Beta)
      if(Bilinear_work(nmx).i_stat.eq.2.or.
     *   Bilinear_work(nmx).i_stat.eq.3) iistat = iistat + 1
```

```
      goto 100
   80 continue
c――――――――――――――――――――――――――――★非対称トリリニア型
      call TriLinear_AS(M_model_fiber(nnm).d_E,
     *               Trilinear_work(nmx).i_stat,
     *               E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *               E_model_fiber(nn).E_3,E_model_fiber(nn).Ec_1,
     *               E_model_fiber(nn).Ec_2,E_model_fiber(nn).Ec_3,
     *               E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *               E_model_fiber(nn).Qc_1,E_model_fiber(nn).Qc_2,
     *               du,M_model_fiber(nnm).d_stress_x,
     *              Trilinear_work(nmx).P1(1),Trilinear_work(nmx).P1(2),
     *              Trilinear_work(nmx).P1(3),
     *               E_model_fiber(nn).Beta, E_model_fiber(nn).Beta_2)
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 100
      elseif(nm_type/10.eq.10) then
c――――――――――――――――――――――――――――★弾塑性解析
      goto (110,120),nm_type - 100
   110 continue
c――――――――――――――――――――――――――――★個人用ファイバー履歴特性
c     call New_model_fiber()
      goto 100
   120 continue
c――――――――――――――――――――――――――――★個人用ファイバー履歴特性
      goto 100
      endif
c――――――――――――――――――――――――――――★弾塑性解析終了
      endif
   100 continue
      M_model_fiber(nnm).d_eps_x = M_model_fiber(nnm).d_eps_x + du !ファイバー要素の歪
      enddo
c――――――――――――――――――――――――――――――――――――★ファイバー要素応力の計算
      if(n_dstat.ge.1.and.n_dstat.le.3) then      ! n_dstat:塑性状態を表す位置（0:表示しない）
      d_state = float(iistat)/float(nm_div)   ! 塑性した面積の計算
      if(d_state .eq.0) then
      Member.d_stat(n_dstat) = 0
      elseif(d_state .ge.0.8) then
      Member.d_stat(n_dstat) = 2
      else
      Member.d_stat(n_dstat) = 1
      endif
      endif
      nm_div  = E_modelx.n_section
      nn      = E_modelx.nm_section - 1
      nnm     = M_modelx.nm_section - 1
      ra      = 0.
      ray     = 0.
      raz     = 0.
      raz2    = 0.
      ray2    = 0.
      rayz    = 0.
      gg      = 0.
      aa      = 0.
```

```
      AN      = 0.
      AMy     = 0.
      AMz     = 0.
      do i=1,nm_div
      nn      = nn  + 1
      nnm     = nnm + 1
      A       = E_model_fiber(nn).A
      E       = M_model_fiber(nnm).d_E
      ra      = ra   + E*A
      ray     = ray  + E*E_model_fiber(nn).Arz
      raz     = raz  + E*E_model_fiber(nn).Ary
      ray2    = ray2 + E*E_model_fiber(nn).Arz2
      raz2    = raz2 + E*E_model_fiber(nn).Ary2
      rayz    = rayz + E*E_model_fiber(nn).Aryz
      aa      = aa + A
      gg      = gg + A*E_model_fiber(nn).G
      ANN     = M_model_fiber(nnm).d_stress_x*E_model_fiber(nn).A
      AN      = AN + ANN
      AMy     = AMy + ANN * E_model_fiber(nn).rz
      AMz     = AMz + ANN * E_model_fiber(nn).ry
      enddo
      nn=E_modelx.nm_section
      call jikuzero_control(Control,ra,E_model_fiber(nn).E_1,
     *                      E_model_fiber(nn).A)
      M_modelx.d_aa = aa              !  断面積の和
      M_modelx.d_ra = ra              !  E*断面積の和
      M_modelx.d_ray = ray            !  E*A*z
      M_modelx.d_raz = raz            !  E*A*y
      M_modelx.d_raz2 = raz2          !  E*A*y*y
      M_modelx.d_ray2 = ray2          !  E*A*z*z
      M_modelx.d_rayz = rayz          !  E*A*z*y
      M_modelx.d_gg   = gg            !  G*A
      return
      end
```

9.4.9 部材の応力

本節では、部材応力について考える。部材応力を計算するサブルーチン Cal_stress()では、部材モデル毎に計算を実行している。したがって、新規の静的縮合部材モデルに対して、部材モデル番号 51-70 で処理を行うサブルーチン Cal_stress_Mx()を挿入する。SPACE では、部材両端の応力は、接線剛性を用いて計算する簡易的な方法を使用している。そのため、サブルーチン Cal_stress_Mx()は容易に作成することができる。また、ここで、部材中央の応力を求めている。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stress
C       ―――――――――――――――――――――――――――――――――
C     ●   部材内応力の計算(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_stress(Member,n_member,Model_type,Element,
```

```
     *     past_disp_point,past_vel_point,est_ddisp_point,rot_memb,
     *     E_model6_real,ak_nonlinear)
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_model6_real_s / E_model6_real
      dimension Member(*),Element(*),E_model6_real(*)
      dimension rot_memb(3,3,2,*),ak_nonlinear(12,12,*)
      dimension past_disp_point(*),past_vel_point(*),est_ddisp_point(*),
     *          v(12),vv(12),vp(12),vpp(12),ak(12,12)
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
      ie = Member(i).nm_element
      im = Element(ie).n_element
      imm = Member(i).n_element_type
      ien= Member(i).n_model_type
      if(Member(i).nm_dll_element .ne. 0) goto 9999   ! DLL 要素
      if(iett.eq.0)then
      goto(11,12,13,14,15,16,17,18,19,20),iet
   11 continue
c――――――――――――――――――――――――――――――――★Model_No.1 通常の有限要素弾塑性モデル
      call Cal_stress_M1(Member(i),Element(ie),
     *     ak_nonlinear(1,1,i),v,rot_memb(1,1,1,i),rot_memb(1,1,2,i))
      goto 100
   12 continue
c――――――――――――――――――――――――――――――――★Model_No.2 3 次元せん断弾塑性モデル
      goto 100
   13 continue
c――――――――――――――――――――――――――――――――★Model_No.3 3 次元軸力弾塑性モデル
      goto 100
   14 continue
c――――――――――――――――――――――――――――――――★Model_No.4 3 次元ケーブル弾塑性モデル
```

```
      goto 100
   15 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.5 3 次元免振モデル
      goto 100
   16 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル(ok)
      goto 100
   17 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.7 3 次元プレテンション動作モデル
      goto 100
   18 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.8
      goto 100
   19 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.9
      goto 100
   20 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.10
      goto 100

      elseif(iett.eq.1)then
      goto(111,112,113,114,115,116,117,118,119,120),iet-10
  111 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.11 両端ファイバーモデル
      goto 100
  112 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.12 両端、中央ファイバーモデル
      goto 100
  113 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.13 両端 MS モデル
      goto 100
  114 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.14 両端、中央 MS モデル
      goto 100
  115 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.15 幾何学非線形+弾塑性型有限要素モデ
      goto 100
  116 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.16
      goto 100
  117 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.17
      goto 100
  118 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.18
      goto 100
  119 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.19
      goto 100
  120 continue
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.20
      goto 100
      elseif(iett.eq.5.or. iett.eq.6)then
c――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.51-70
```

```
      call Cal_stress_Mx( Model_type ,Member(i),Element(ie),
     *      ak_nonlinear(1,1,i) ,v,rot_memb(1,1,1,i),rot_memb(1,1,2,i))
      do j=1,6
      k=j+12
      Member(i).stress(k)=(Member(i).stress(j)+
     *                  Member(i).stress(j+6))*0.5
      enddo
      goto 100
      endif
      goto 100
 9999 continue
  100 continue
      end do
      return
      end
```

部材両端の応力を計算するサブルーチンは、接線剛性を用いて計算する簡易型で行う。そのため、以下のように容易に作成可能である。

```
C     ――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stress_Mx
C       ――――――――――――――――――――――――――――――――
C     ●   代表的な部材モデルの応力計算：非線形剛性を用いて、材端応力を計算する
C     ――――――――――――――――――――――――――――――――――
      subroutine Cal_stress_Mx(Model_type,Member,Element, ak,vv,r1,r2)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
      dimension ak(12,12) ,vv(12),r1(3,3),r2(3,3),st(12),ss(12)
C     ――――――――――――――――――――――――――――――――――
      do i=1,12
      s=0.
      do j=1,12
      s=s+ak(i,j)*vv(j)
      enddo
      st(i)=s
      enddo
c――――――――――――――――――――――――――――――――――――★全体座標から部材座標へ変換
      call RotateL_v(1,st,r1,r2,ss)
      do i=1,6
      Member.stress(i)=-ss(i)+Member.stress(i)
      enddo
      do i=7,12
      Member.stress(i)=ss(i)+Member.stress(i)
      enddo
      return
      end
```

以上で、新規静的縮合モデルの組み込みは終了である。ただし、他のモジュール、例えば動的プレゼンターや静的ソルバーなどに、新規静的

縮合モデルの仕様を追加し、動的解析システムと適合させなければならない。それについては各マニュアルで説明する。

9.4.10 部材モデルの組み込みの復習

全節では、新たな部材モデルを SPACE に組み込む方法について解説した。ここでは、復習のために前節で示した部材モデルを静的解析システムとプレゼンテーションに組み込む手順を示そう。

静的解析システムは、動的解析システムとほぼ同じシステムを用いている。そこで、新しい部材モデルを組み込む方法を確実にものにするため、この静的解析システムに部材モデルを組み込む手順を示す。部材モデルは、前節と同じとする。

1．新しいヘッダーファイルを主サブルーチン submain.f に組み込む。
2．新しいヘッダーファイルNew_submain.hで定義した新しい構造体の宣言、確保を行うコードを挿入する。
3．既構造体に項目を追加する。
4．このモデルのために新たに設計したサブルーチンをプロジェクトに挿入する。

```
Get_S_comp_model.f
Cal_check_stiff_Mx.f
Cal_lin_stiff_Mxx.f
Cal_nonlin_stiff_Mxx.f
Cal_stress_Mx.f
Fiber_checkx.f
Fiber_Model_GIx.f
Fiber_Model_Gx.f
Stiff_Mx.f
Stiff_Mx_I.f
set_modelx_dat.f
```

5．以下の主サブルーチン submain.f 内サブルーチンの引数を変更する。

```
Get_structure
Fiber_input
Cal_stiff_linear
Check_stress
Out_Fiber
Get_nonlinear_stiff
```

６．以下の既設のサブルーチン内を変更する。

inpute()
Get_parameters()
Get_structure()
Fiber_input()
Cal_stiff_linear()
Check_stress()
Out_Fiber()
Get_nonlinear_stiff()
Cal_stress()

７．ファイバー入力の条件に新規モデルを加える。

８．静的解析では､収束過程でサブルーチン Cal_unb_stress()を用いて不釣合力を計算する。この不釣合い力を計算するサブルーチンは、動的解析ではないので新たに設計する必要がある。以下に示すように太文字の部分を変更する。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_unb_stress
C     ―――――――――――――――――――――――――――――――――
C     ●   部材の不釣合力を計算する(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_unb_stress(N_analysis,
     *     ak_nonlinear,Member,n_member,
     *     Model_type,Element,past_disp_point,disp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
     *     E_model21, M_model21,
     *     E_model22, M_model22,
     *     E_model31, M_model31,
     *     E_model32, M_model32,
     *     E_model33, M_model33,
     *     MSS_work,
     *     Bilinear_work,Trilinear_work,Concrete_work,RO_work,
     *     work1_elememt,work2_elememt, work1_member,
     *     work2_member,ld_point,
     *     S_comp_model,E_modelx,M_modelx,
     *     E_fiber_work,M_fiber_work)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record / member_s    / Member
      record / element_s   / Element
```

```
      record / n_model_s   / Model_type
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
c―――――――――――――――――――――――――――――――★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)
      dimension E_fiber_work(*),M_fiber_work (*)
c―――――――――――――――――――――――――――――――★Model_No.1 通常の有限要素弾塑性モデル
c
c―――――――――――――――――――――――――――――――★Model_No.2 3 次元せん断弾塑性モデル
c      record /element2_s / Element
      record / RO_work_s          /  RO_work
c―――――――――――――――――――――――――――――――★Model_No.3 3 次元軸力弾塑性モデル
c      record /element3_s / Element
c      record /member3_s  / Member
C     record / N_Buckling_s           / N_Buckling
c―――――――――――――――――――――――――――――――★Model_No.4 3 次元ケーブル弾塑性モデル
c      record /element4_s / Element
c      record /member4_s  / Member
c―――――――――――――――――――――――――――――――★Model_No.5 3 次元免振モデル
c      record /element5_s / Element
c      record /member5_s  / Member
       record / MSS_work_s         /  MSS_work
c―――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル
c      record /element6_s         / Element
       record / E_model6_real_s   / E_model6_real
c―――――――――――――――――――――――――――――――★Model_No.7 3 次元バネモデル
c      record /element7_s / Element
c      record /member7_s  / Member
c      record / E_model7_real_s    / E_model7_real
c―――――――――――――――――――――――――――――――★Model_No.11 両端ファイバーモデル
       record / E_model11_s    / E_model11
       record / M_model11_s    / M_model11
c―――――――――――――――――――――――――――――――★Model_No.12 両端、中央ファイバーモデル
       record / E_model12_s    / E_model12
       record / M_model12_s    / M_model12
c―――――――――――――――――――――――――――――――★Model_No.13 両端ピン、中央ファイバーモデル
       record / E_model13_s    / E_model13
       record / M_model13_s    / M_model13
c―――――――――――――――――――――――――――――――★Model_No.15 両端ファイバーFEM モデル
       record / E_model15_s    / E_model15
       record / M_model15_s    / M_model15
c―――――――――――――――――――――――――――――――★Model_No.21 両端 MS モデル
       record / E_model21_s    / E_model21
       record / M_model21_s    / M_model21
c―――――――――――――――――――――――――――――――★Model_No.22 両端、中央 MS モデル
       record / E_model22_s    / E_model22
       record / M_model22_s    / M_model22
c―――――――――――――――――――――――――――――――★Model_No.31 両端アナロジーモデル
```

```
      record / E_model31_s    / E_model31
      record / M_model31_s    / M_model31
c-----------------------------★Model_No.32 両端、中央アナロジーモデル
      record / E_model32_s    / E_model32
      record / M_model32_s    / M_model32
c-----------------------------★Model_No.33 両端、中央アナロジーモデル
      record / E_model33_s    / E_model33
      record / M_model33_s    / M_model33
c-----------------------------★Model_No.51 3 次元プレテンション動作モデル
c     record / E_model51_s    / E_model51
c     record / M_model51_s    / M_model51

c-----------------------------★ファイバーモデル用エリア
      record / E_model_fiber_s   / E_model_fiber
      record / M_model_fiber_s   / M_model_fiber
c-------------------------------------------------------------
     dimension E_model_fiber(*),M_model_fiber(*)
     dimension Member(*),Element(*),E_model6_real(*)
     dimension MSS_work(*),RO_work(*)
     dimension E_model11(*),M_model11(*)
     dimension E_model12(*),M_model12(*)
     dimension E_model13(*),M_model13(*)
     dimension E_model15(*),M_model15(*)
     dimension E_model21(*),M_model21(*)
     dimension E_model22(*),M_model22(*)
     dimension E_model31(*),M_model31(*)
     dimension E_model32(*),M_model32(*)
     dimension E_model33(*),M_model33(*)
     dimension ak_nonlinear(12,12,*),rot_memb(3,3,2,*)
     dimension work1_elememt(*),work2_elememt(*),
    *          work1_member(*), work2_member(*)
     dimension past_disp_point(*),disp_point(*)
     dimension vv(12),vp(12),vpp(12),v(12),ak(12,12)
     dimension f(12),ff(12)
     real*8 ld_point(*)
c-------------------------------------------------------------
c       Model_No.1 = 1             !  通常の有限要素弾塑性モデル
c       Model_No.2 = 2             !  3 次元せん断弾塑性モデル
c       Model_No.3 = 3             !  3 次元軸力弾塑性モデル
c       Model_No.4 = 4             !  3 次元ケーブル弾塑性モデル
c       Model_No.5 = 5             !  3 次元免振モデル
c       Model_No.6 = 6             !  3 次元制震 Maxwell モデル
c       Model_No.7 = 7             !  3 次元プレテンション動作モデル
c
c    要素数
c      structure / element_s/
c      integer  element_type   !  要素タイプ
c      integer  n_element      !  非線形要素番号
c      real*8   E              !  ヤング係数
c      real*8   G              !  せん断係数
c      real*8   A              !  断面積
c      real*8   RIx            !  ねじり剛性
c      real*8   RIy            !  ｙ軸断面二次モーメント
c      real*8   RIz            !  ｚ軸断面二次モーメント
```

```
c      real*8   AM              !  単位長さ当たりの質量
c      integer  nm_damp         !  部材減衰の有無
c      end structure
c     record /element_s/ Element
c     ALLOCATABLE ::Element(:)
c     ALLOCATE (Element(n_element))
c
c
c     部材数
c      structure / member_s/
c      integer  nm_element         !  要素番号
c      integer  element_type       !  要素タイプ
c      integer  n_element_type     !  要素タイプ別番号
c      integer  nm_dll_element     !  DLL を用いた要素か（０；システム内要素、１：DLL 要素）
c      integer  nm_point(2)        !  節点番号
c      integer  irest(12)          !  部材両端の自由度番号表
c      integer  nm_analysis        !  部材解析種別
c      integer  nm_group           !  部材グループ
c      integer  nm_local_coord(2)  !  局所座標系の有無とその回転行列の番号
c      integer  nm_damp            !  部材減衰の有無とその減衰行列の番号
c      real*4   alength            !  長さ
c      real*4   rot_x              !  部材主軸の回転角度（度）
c      real*8   force(12)          !  部材両端の部材端力
c      end structure
c     record / member_s / Member
c     ALLOCATABLE :: Member (:)
c     ALLOCATE (Member (n_member))
c
c  モデルパラメータ
c      structure  / n_model_s/
c     integer    n_e_models         ! 要素モデルの最大数
c      integer    no_e_model(20)     ! 要素モデルの番号
c      integer    n_div_model(20)    ! 要素モデルの分割数
c      integer    nm_div_model(20)   ! 要素モデル内のサブ要素の分割数
c      integer    n_e_model(20)      ! 要素モデルの数
c      integer    n_m_model(20)      ! 部材モデルの数
c      integer    n_damp             ! 部材減衰ありか
c       end structure
c      record / n_model_s / Model_type
c
c
c      ak_nonlinear          real*8    接線剛性行列
c      Member                structure
c      n_member              integer   部材数
c      Model_type            structure
c      Element               structure
c      past_disp_point(*)    real*8    増分前の変位
c      rot_memb              real*8    回転行列
c      E_model1_int          integer   要素モデル１
c      E_model1_real         real*8    要素モデル１
c      M_model1_int          integer   部材モデル１
c      M_model1_real         real*8    部材モデル１
c      E_model2_int          integer   要素モデル２
c      E_model2_real         real*8    要素モデル２
```

```
c      M_model2_int          integer   部材モデル２
c      M_model2_real         real*8    部材モデル２
c      E_model3_int          integer   要素モデル３
c      E_model3_real         real*8    要素モデル３
c      M_model3_int          integer   部材モデル３
c      M_model3_real         real*8    部材モデル３
c      E_model4_int          integer   要素モデル４
c      E_model4_real         real*8    要素モデル４
c      M_model4_int          integer   部材モデル４
c      M_model4_real         real*8    部材モデル４
c      E_model5_int          integer   要素モデル５
c      E_model5_real         real*8    要素モデル５
c      M_model5_int          integer   部材モデル５
c      M_model5_real         real*8    部材モデル５
c      E_model6_int          integer   要素モデル６
c      E_model6_real         real*8    要素モデル６
c      M_model6_int          integer   部材モデル６
c      M_model6_real         real*8    部材モデル６
c      work1_elememt         real*8    DLL 用ワークエリア
c      work2_elememt         real*8    DLL 用ワークエリア
c      work1_member          real*8    DLL 用ワークエリア
c      work2_member          real*8    DLL 用ワークエリア
c-------------------------------------------------------------
       do i=1,n_member
c      write(76,'(a,2i4)') ' memb : ',i,Member(i).element_type
c      kk=10
c      write(76,'(a,i4,3f12.3)') 'non c',kk,Member(kk).stress(7),
c     *  Member(kk).stress(8)
c-------------------------------------★部材両端の変位取得
       do j=1,12
       ires=Member(i).irest(j)
       if(ires.gt.0) then
c      write(76,*) j,'  =  ',ires
       v(j)=disp_point(ires)
       vp(j)=past_disp_point(ires)
       else
       v(j)=0.
       vp(j)=0.
       endif
       enddo
c      write(76,'(a,i3,12e12.5)') 'vv',i,(v(j),j=1,12)
c      write(76,'(a,i3,12e12.5)') 'vp',i,(vp(j),j=1,12)
c-------------------------------------★部材両端の節点力のゼロセット
       do j=1,12
       f(j)=0.
       enddo
c-------------------------------------★変位を釣合系から部材座標系に変換

       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        write(76,'(a,i3,12e12.5)') 'vv',i,(vv(j),j=1,12)
c        write(76,'(a,i3,12e12.5)') 'vp',i,(vpp(j),j=1,12)
c-------------------------------------★要素及びモデルのセット
       mem = i
```

```
      iet  = Member(i).element_type
      iett=(iet-1)/10
      ie   = Member(i).nm_element
      im   = Element(ie).n_element
      ien  = Member(i).n_model_type
      if(Member(i).nm_dll_element .ne. 0)  goto 9999   ! DLL 要素
c      write(76,'(a,i4)') ' check member:',i
      if(iett.eq.0)then
      goto(11,12,13,14,15,16,17,18,19,20),iet
   11 continue
c-------------------------------★Model_No.1 通常の有限要素弾塑性モデル
      call Cal_check_unb_M1(ak_nonlinear(1,1,i),v,ff)
      goto 101
   12 continue
c-------------------------------★Model_No.2 3 次元せん断弾塑性モデル
      call Cal_check_unb_Mx(ak_nonlinear(1,1,i),v,ff)
c     do j=1,3
c     f(j)=-Member(i).stress(j)
c     f(j+6)=-f(j)
c     enddo
c     call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
   13 continue
c-------------------------------★Model_No.3 3 次元軸力弾塑性モデル
      call Cal_check_unb_M3(ak_nonlinear(1,1,i),v,ff)
      goto 101
   14 continue
c-------------------------------★Model_No.4 3 次元ケーブル弾塑性モデル
      call Cal_check_unb_Mx(ak_nonlinear(1,1,i),v,ff)
c     f(1)=-Member(i).stress(1)
c     f(7)=-f(1)
c      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
   15 continue
c-------------------------------★Model_No.5 3 次元免振モデル
      call Cal_check_unb_Mx(ak_nonlinear(1,1,i),v,ff)
c     do j=1,3
c     f(j)= -Member(i).stress(j)
c     f(j+6)=-f(j)
c     enddo
c      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
   16 continue
c-------------------------------★Model_No.6 3 次元制震 Maxwell モデル(ok)
c      call  Check_maxwelldamp(E_model6_real(ien),Element(ie))

      goto 100
   17 continue
c-------------------------------★Model_No.7 3 次元プレテンション動作モデル

c      call Cal_check_stiff_M7(Member(i),Element(ie),
c     *     E_model1_int(im),E_model1_real(im),
c     *     M_model1_int(imm),M_model1_int(imm),
c     *     vv,ak )
```

```
c      do j=1,12
c      f(j)=0.
c      enddo
c      do j=1,3
c      f(j)=-Member(i).stress(j)
c      f(j+6)=-f(j)
c      enddo
c       call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
c       do j=1,12
c      Member(i).force(j)=ff(j)
c       enddo
       goto 100
   18 continue
c-------------------------------★Model_No.8

       goto 100
   19 continue
c-------------------------------★Model_No.9

       goto 100
   20 continue
c-------------------------------★Model_No.10

       goto 100


       elseif(iett.eq.1)then
c        write(76,'(a,i3,12e12.5)') 'iet',iet
       goto(111,112,113,114,115,116,117,118,119,120),iet-10
  111 continue
c-------------------------------★Model_No.11 両端ファイバーモデル
       call Cal_check_unb_M11(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model11, E_model_fiber,
     *      M_model11, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vpp,f)
       call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
       goto 101
  112 continue
c-------------------------------★Model_No.12 両端、中央ファイバーモデル
       call Cal_check_unb_M12(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model12, E_model_fiber,
     *      M_model12, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vpp,f)
       call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
       goto 101
  113 continue
c-------------------------------★Model_No.13 両端 MS モデル
       call Cal_check_unb_M21(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model21, E_model_fiber,
     *      M_model21, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vpp,f)
```

```
c――――――――――――――――――――――――――★部材の両端節点力を釣合系に変換
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
  114 continue
c――――――――――――――――――――――――――――――――――――★Model_No.14 両端、中央 MS モデル
       call Cal_check_unb_M22(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model22, E_model_fiber,
     *      M_model22, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vpp,f)
c――――――――――――――――――――――――――★部材の両端節点力を釣合系に変換
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
  115 continue
c――――――――――――――――――――――――――――――――――――★Model_No.15 幾何学非線形+弾塑性型有限要素モデ
      call Cal_check_unb_M15(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model15, E_model_fiber,
     *      M_model15, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vpp,f)
c――――――――――――――――――――――――――★部材の両端節点力を釣合系に変換
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
  116 continue
c――――――――――――――――――――――――――――――――――――★Model_No.16 両端アナロジーモデル
      call Cal_check_unb_M31(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model31, E_model_fiber,
     *      M_model31, M_model_fiber,
     *      vv,vpp,f)
c――――――――――――――――――――――――――★部材の両端節点力を釣合系に変換
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
  117 continue
c――――――――――――――――――――――――――――――――――――★Model_No.17 両端、中央アナロジーモデル
      call Cal_check_unb_M32(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model32, E_model_fiber,
     *      M_model32, M_model_fiber,
     *      vv,vpp,f)
c――――――――――――――――――――――――――★部材の両端節点力を釣合系に変換
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
  118 continue
c――――――――――――――――――――――――――――――――――――★Model_No.18 両端ピン、中央ファイバーモデル
      call Cal_check_unb_M13(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model13, E_model_fiber,
     *      M_model13, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vpp,f)
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
  119 continue
c――――――――――――――――――――――――――――――――――――★Model_No.19 両端ピン、中央アナロジーモデル
```

```
      call Cal_check_unb_M33(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model33, E_model_fiber,
     *      M_model33, M_model_fiber,
     *      vv,vpp,f)
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 101
  120 continue
c―――――――――――――――――――――――――――――――★Model_No.20

      goto 100

      elseif(iett.eq.5.or. iett.eq.6)then
      call Cal_check_unb_Mxx(Control,N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vpp,f,
     *      S_comp_model,E_fiber_work,M_fiber_work)
c―――――――――――――――――――――――――★部材の両端節点力を釣合系に変換
      call RotateL_v(2,f,rot_memb(1,1,1,i),rot_memb(1,1,2,i),ff)
      goto 100

      endif
      goto 100
 9999 continue

c―――――――――――――――――――――――――――――――★Model_No.DLL
c      call Cal_check_stiff_dll(mem,Member(i),Element(ie),
c     *   work1_elememt,work2_elememt,work1_member,work2_member,
c     *     vv,ak)
      goto 100
  101 continue
c―――――――――――――――――――――――――――――――★不釣合力の足しこみ
c     write(76,'(a,i4,12f12.4)') ' ff',i,(ff(j),j=1,12)
      call get_pointforce_ldx(ff,ld_point,Member(i))

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

      end do
      return
      end
```

上のサブルーチンで使用する新たなサブルーチンを以下のように設計する。

```
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_check_unb_Mxx
C     ―――――――――――――――――――――――――――――――――――
C     ●    代表的な部材モデルの不釣合い力チェック(ok)
C     ―――――――――――――――――――――――――――――――――――
```

```
      subroutine Cal_check_unb_Mxx(N_analysis,
     *      mem_x,Model_type,Member,Element,
     *      E_modelx, E_model_fiber,
     *      M_modelx, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vp,f_p,
     *      S_comp_model,E_fiber_work,M_fiber_work)
c-----------------------------------------------------------------
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s   / Model_type
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
c-------------------------------★Model_No.51-70 任意要素型縮合モデル
      record / S_comp_model_s    / S_comp_model
      record / E_modelx_s        / E_modelx
      record / M_modelx_s        / M_modelx
      record / E_fiber_work_s    / E_fiber_work
      record / M_fiber_work_s    / M_fiber_work
      dimension E_modelx(*),M_modelx(*),S_comp_model(*)
      dimension E_fiber_work(*),M_fiber_work(*)
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension ak(12,12),f_p(12)
      dimension vv(12),vp(12),vvx(12)
      dimension Bilinear_work(*),Trilinear_work(*)
      dimension Concrete_work(*)

      real*8, ALLOCATABLE :: c(:,:),ab(:,:),alength(:),EA(:)
      real*8, ALLOCATABLE :: bav(:),akk(:,:,:),f1(:),f2(:)
      integer, ALLOCATABLE :: irest_Point(:,:),n_type(:)
c-----------------------------------------------------------------
C     ---------------------------------------
      iet = Member.n_model                        ! モデルタイプ番号
      ix_model= iet-50                            ! 任意要素モデル(51-69)
      nmmx= Member.n_model_type -1               ! M_Fiber_work の開始番号
      nmx = Element.n_section(1) -1              ! E_Fiber_work の開始番号
      n_div= S_comp_model(ix_model).n_div_element  ! 部材分割数
      n_if = 6*(n_div-1)                           ! 内部自由度
      write(76,'(a,4i6)') ' model:',iet,n_div,imm,immm
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_modelx_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,S_comp_model(ix_model),
     *         n_type,alength,
     *  Member.i_rigid_length,    ! i端剛域
     *  Member.j_rigid_length)    ! j端剛域
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),f2(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
      EAx=Element.A*Element.E
c-------------------------------------★部材剛性行列の作成
      do i=1,n_div
      imm = E_Fiber_work(nmx+i).nm_section            ! エレメント番号
      immm= M_Fiber_work(nmmx+i).nm_section           ! 内部エレメント番号
      EAA=EAx
      if(n_type(i).eq.2) then       ! ファイバー：他のモデルでも必要なときはここに加える
      EAA=M_modelx(immm).d_ra
      endif
c-------------------------------------★内部部材の剛性行列の計算
      call Stiff_Mx(i,n_type(i),akk(1,1,i),Member,alength,
     *       Model_type,Element,
     *       E_modelx(imm), E_model_fiber,
     *       M_modelx(immm), M_model_fiber)
      if(N_analysis.ne.9.and.N_analysis.ne.7.and.n_type(i).ne.3) then    !  幾何学的非線形剛性
      call Cal_geomet_stiffx(M_Fiber_Work(nmmx+i).an_stress,Member,
     *                       akk(1,1,i),alength(i))
      call Create_Kn(akk(1,1,i), M_Fiber_Work(nmmx+i).an_vv,
     *                                M_Fiber_Work(nmmx+i).an_ww,
     *                                EAA,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)
      do i=0,n_if-1
      k=i/6
      j = i -(k)*6
      f2(i+1) = M_Fiber_work(nmmx+k+1).ff_ip(j+1)                                  ! 1
      enddo
      call Typical_member_v(c,ab,f2,
     *                      n_if,n_if,iubw,iubw,ier,vv,bav)
c――――――――――――――――――――――――★部材内部、両端節点力の計算
      do i=1,n_div
      call Typical_member_p_force(akk(1,1,i),irest_Point(1,i),
     *                            vv,bav ,f_p,f1)
      enddo
c――――――――――――――――――――――――★部材内部、両端節点力から不釣合力へ
      do i=0,n_if-1
      k=i/6
      j = i -(k)*6
      M_Fiber_work(nmmx+k+1).ff_ip(j+1) = f1(i+1)                                  ! 2
      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――――――――――――――――――――――――――――――――★動的記憶領域の解放
      DEALLOCATE (
     *   c ,ab ,irest_point,n_type,alength,
     *   bav,akk,f1,f2   )
      return
      end
```