第１章　履歴特性の更新と新規組み込み法

1.1 はじめに

SPACE (SPace frame Analysis package for Civil Engineers, researchers and students)に組み込まれている履歴モデルには、せん断型モデルの履歴やファイバーの履歴、あるいは個材座屈を考慮したトラスモデルのように部材特有の履歴モデルもある。本章では、部材モデル階層構造の最下層に位置する履歴モデルをどのようにSPACEシステムに組み込むか、また組み込む際どのような注意が必要かについて述べる。読者は、既にSPACEの動的解析システムについては、多くの知識を得ていると思う。知識が不十分であるにもかかわらず、安易にSPACEに変更を加えると、妥当な解が得られるどころか、他のモデルも解析できなくなってしまう恐れがある。もし、動的解析システムに関する知識が不十分な場合は、マニュアル：動的解析編を良く読んで理解して頂きたい。

SPACE や動的解析に関する知識を利用して、新たな履歴モデルや部材モデルを組み込む手続きと注意点を示す。特に、SPACE では静的解析と動的解析において同じ構造用データを用いて解析を行っている。従って、動的解析に新しいモデルを組み込む場合は、静的解析にも同様に新規モデルを組み込んで頂きたい。また、他のモジュール、例えばプレゼンターなどにも影響を与えることになるので、システム全体を良く理解した上で変更されたい。

履歴モデルや部材モデルを組み込むには、適切な順序と方法で行う必要がある。一般的な履歴モデルの組み込み順序は次のようである。

１）モデルを設計し、そのモデルに登録番号を付ける。
２）データの入力仕様の設計と現在の仕様との調整を行う。
３）出力仕様の設計とプレゼンターとの関係を調査する。
４）特殊な構造体を必要とするかどうか調査する。構造体を設計する場合は、他の構造体との整合性を確認する。また、その構造体の定義と構造体配列の宣言と確保、及び、解放処理を行う場所を確認する。
５）新規に必要となるサブルーチンを設計し、作成する。設計に際し、次の点に注意する。
- **骨格曲線を形成するパラメータを設計する。**
- **履歴ルールを明確にする。特に、他の状態に移る条件を正確に設定する。**

・　誤差が生じないような処置を講じる。
６）モデルの階層構造を理解し、どのレベルに組み込むかを調査する。
７）既存のサブルーチンとのインターフェイス、並びに既存のサブルーチンの追加・変更部分の調査を行う。
８）設計した新規サブルーチンを SPACE の適切な位置に組み込む。
９）ここから実際にシステムを動作させ、組み込まれた新規モデルが設計した仕様を満足するかどうか動作確認する。最初に設計したサブルーチンの引数を出力し、適切な値となっているかどうか、チェックする。
10）次に、簡単なモデルで履歴などをチェックする。
11）複雑な解析モデルを用いて、整合性をチェックする。
12）他のモデルとの整合性をチェックする。

SPACE システムは、一般に公開にされたシステムであり、常にバージョンアップが行われる。従って、個人で新規モデルを組み込む場合は、次の点に注意を払って頂ければ幸いである。

１）個人で使用する場合は、モデル番号や履歴番号は、個人用の番号を使用する。SPACE システムがバージョンアップした場合は、再度開発したプログラムに組み込む必要が生じるが、個人用番号を使用すればデータなどに不具合を生じることはない。
２）新規モデルが完成したときは、SPACE に登録していただきたい。それには、まず関連サブルーチンとマニュアルを提出する。登録することで、正規のモデル番号を取得し、SPACE の新バージョンが発行されるとき正規モデルとなる。SPACE を育てるためにも、是非、新規開発モデルの正規登録にご協力ください。

履歴モデルは最下層にあるため、他のプログラムと強く関連することがない。そのため、複雑なバグを発生させる可能性は少ない。しかし、履歴モデルには、それ特有の難しさやエラーを生じさせる要因を含んでおり、あらゆる場合を想定して徹底的にチェックする必要があろう。

履歴モデルを作成する場合、気を付けなければいけない点が多々ある。理論を正確に表現するコードのみでは正常に動作しない。履歴モデルは多くの場合、増分型で作られるため、一旦正常な履歴ループから外れると、後は全く意味のない計算を行うことになる。このような現象が如何にして生じるのか、その原因を考えてみよう。

履歴モデルを要因とするエラーには、次第に誤差が蓄積し、解析結果に信頼が置けなくなる場合と、一瞬に大きな誤差が発生し、履歴ループを正常に追えなくなってしまう場合とがある。前者は、履歴ループを直線で近似するモデルで、ある直線から他の直線に乗り移るときに生じる。このとき応力は大なり小なり必ず飛び越しを起こし、誤差が生じる。これを避けるために、履歴モデルの中になんらかの処置をしておかなければならない。SPACE での対処法は、各モデルの解説の項で紹介する。一方、ユーザー側は増分間隔を小さくしてこれに対処することになる。

後者のエラーは、以後の数値計算を無意味とする。この種のエラーは、増分ひずみが突然大きくなり、履歴ループを正常に追えなくなるときに生じる。例えば、構造物に崩壊メカニズムが発生し、部分的に大きな変位が発生するときにこのような大きなひずみを生じる。また、接線剛性がゼロに近い場合や極端な剛性変化が生じる場合にもこのような現象が起き易い。これらの現象を踏まえて、履歴モデルをプログラムコードで表すとき、何らかの対処法が必要となろう。SPACE に組み込まれている履歴モデルがどのように処理しているかを、このマニュアルで理解されたい。

現在、SPACE に組み込まれている部材に関する履歴モデルは、

１．バイリニア型：
２．トリリニア型：
３．最大点指向型
４．武田モデル
５．修正バイリニア型
６．修正 RO モデル
７．スリップバイリニア型
８．個材座屈を考慮したトラス型

があり、さらに、ファイバー用の履歴モデルとしては、

１．対称バイリニア型：BiLinear()
２．対称トリリニア型：TriLinear()
３．直線コンクリート履歴モデル：Concrete()
４．曲線コンクリート履歴モデル：Concrete_e()
５．バイリニア型（移動＋等方硬化用）：BiLinear_h()
６．対称トリリニア型（移動＋等方硬化用）：TriLinear_h()
７．非対称バイリニア型：TriLinear_AS()
８．非対称トリリニア型：BiLinear_AS()

などがある。次章以降で、これらの履歴モデルを具体的に説明する。まずは、次節より新規に履歴モデルを組み込む方法について解説しよう。

1.2 せん断型モデルの履歴の組み込み方法

1.2.1 せん断型モデルの履歴の設計

本節では、せん断型モデルの履歴特性を安全に組み込む方法について具体的に説明する。このせん断型モデルの履歴特性の組み込みが、最も単純、かつ容易である。前節で説明した順序にしたがって、新しいせん断型モデルの履歴特性を組み込んでみよう。

最初に、せん断型モデルの履歴特性を設計する。**履歴特性は骨格曲線と履歴ルールで表す**ため、それらを図で表現し、また、履歴ルールを文章で書き下すと良い。矛盾のない履歴ルールを設計しなければならない。特に、直線で状態を表す直線区分型の場合、その連結部分で接線剛性が大きく変化するときや接線剛性が負勾配を有するときは、誤差やエラーが発生しやすくなる。この段階でなんらかの対策を立てておく必要があろう。

せん断型モデルの履歴特性が設計できたとして、ここでは、仮にその名前を New_Model としよう。この履歴特性に登録番号を付けることになるが、ここでは、仮登録であるとして、101 を設定する。現在の Ver.3.00 では、以下に示すせん断型モデルの履歴が既に登録されており、このバージョン以降も追加の予定がある。

登録番号	履歴モデル名
1	トリリニア型：
2	最大点指向型
3	武田モデル
11	修正バイリニア型
12	修正 R0 モデル

1.2.2 モデルの入力仕様

次に、このモデルの入力仕様について考えてみよう。せん断型モデルでは、標準入力仕様と特別入力仕様の両者が使用されている。登録番号 1-3 の履歴モデルが標準仕様でデータを入力しており、登録番号 11-12 の履歴モデルが特別仕様でデータ入力を行っている。この特別仕様の入力方法を用いると、これらのデータを保存する構造体を別途設計しなければならず、他にも余分な手続きが必要となる。これについては、登録番号 11 の修正バイリニア型を用いて説明しよう。

まず、標準入力仕様のモデルについて説明する。データ入力を行うサブルーチン Get_structure()のなかで、要素データを入力する部分のコードを取り出す。ここで示すデータ構造で、新規履歴モデルの特性が全て設定できるようであれば、標準入力仕様となり、後の手続きが非常に単純となる。

```
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)
       .
       .
      read(5,*,err=9913,end=9918) m_type,e,g,a,rix,riy,riz,asy,asz,              ! 1
     *         am1,am2,anp,ampy,ampz,nm_type
       .
       .
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                                                    ! 2
      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
      Element(i).element_type  =  m_type     ! 要素タイプ番号                   ! 3
      Element(i).E             =  e          ! ヤング係数
      Element(i).G             =  g          ! せん断力係数
      Element(i).A             =  a          ! 断面積
      Element(i).RIx           =  rix        ! ねじり剛性
      Element(i).RIy           =  riy        ! y 軸回りの断面二次モーメント
      Element(i).RIz           =  riz        ! z 軸回りの断面二次モーメント
      Element(i).ASy           =  asy        ! y 軸に関するせん断変形に関係する断面積
      Element(i).ASz           =  asz        ! z 軸に関するせん断変形に関係する断面積
      Element(i).nm_damp       =  0          ! 部材減衰有無 （システムが自動でセットする）
      Element(i).ANP           =  anp        ! 軸方向耐力
      Element(i).AMPY          =  ampy       ! y 軸塑性モーメント
      Element(i).AMPZ          =  ampz       ! z 軸塑性モーメント
      Element(i).nm_type       =  nm_type    ! 履歴タイプ番号                    ! 4
      Element(i).i_rigid_length   =  rd1     ! i 端剛域長さ
      Element(i).j_rigid_length   =  rd2     ! j 端剛域長さ
      Element(i).i_shear_G        =  sg1     ! i 端せん断剛性
      Element(i).j_shear_G        =  sg2     ! j 端せん断剛性
c―――――――――――――――――――――★部材要素は自重を構造体にセット
      if(m_type.eq.1.or.m_type.gt.10) then
      Element(i).AM(1)         =  am1/980.    ! 要素単位長さ当たり(cm*2)質量
      Element(i).AM(2)         =  am2/980.    ! 要素単位長さ当たり(cm*2)質量
      else
c―――――――――――――――――――――★その他はそのままの値を構造体にセット
      Element(i).AM(1)         =  am1
      Element(i).AM(2)         =  am2
```

1. ここで示す要素データの入力が標準仕様の場合に相当し、この中でモデルの特性が設定できるようであれば標準仕様で良い。データ項

目の内、最初の m_type は要素タイプ番号であり、**せん断型の場合、2** となる。また、最後の nm_type は、履歴特性番号であり、ここでは、新規の履歴特性 New_Model として、101 を設定することになる。この 2 つのデータ以外は、全てせん断型の特性値を設定して良いことになる。

2．特別仕様となる修正バイリニアモデルや修正 RO モデルの要素の数を数える。この値を用いて特別に設計した要素に関する構造体の配列数を動的に確保することになる。

3．入力した要素データを要素に関する構造体にセットする。構造体の名前は標準の構造体 Element を用いているので、新規に設定したモデルの特性データを表す名前とは一致しない。これについては後で述べる。

４．履歴タイプの番号が構造体成分 Element(i).nm_type にセットされる。この構造体の内容が 101 であるとき、新規の New_model が動作することになる。

次に、特別入力仕様について考えよう。ここでは、要素データの標準入力仕様を拡張して用いる場合と、特別に入力データ仕様を設計し、そのデータを読み込むサブルーチンを作成する場合とがある。

まず、入力仕様を拡張する場合について説明する。要素データは標準仕様として、静的縮合モデルでは第 2 レコードが追加されており、これと同様に必要とするデータを設計し、下記のようにプログラムを追加して対処する。

```
      read(5,*,err=9913,end=9918) m_type,e,g,a,rix,riy,riz,asy,asz,
     *         am1,am2,anp,ampy,ampz,nm_type
       .
       .
c――――――――――――――――――――――★新規モデル New_Model のために拡張データ入力
      if(m_type .eq. 2 .and. nm_type .eq. 101) then
      read(5,*,err=9913,end=9918) d1,d2,d3,d4,d5
      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,
     *   (Element(i).n_section(j),j=1,2)
      endif
```

無論、このデータが構造体element_sに保存できることが前提である。例えば、それらは上記サブルーチン Get_structure()の中で次の太文字のように設定される。

```
      Element(i).nm_damp      =  d1        ! せん断型モデルの拡張仕様
      Element(i).ANP          =  anp       ! 軸方向耐力
      Element(i).AMPY         =  ampy      ! y 軸塑性モーメント
      Element(i).AMPZ         =  ampz      ! z 軸塑性モーメント
      Element(i).nm_type      =  nm_type   ! 履歴タイプ番号
      Element(i).i_rigid_length  =  d2     ! せん断型モデルの拡張仕様
      Element(i).j_rigid_length  =  d3     ! せん断型モデルの拡張仕様
      Element(i).i_shear_G       =  d4     ! せん断型モデルの拡張仕様
      Element(i).j_shear_G       =  d5     ! せん断型モデルの拡張仕様
```

このように、要素構造体で設定できるデータ個数はおのずと制限されることになる。なお、このデータを使用する場合は、構造体 element_s2、もしくは新たに設計した構造体を使用することになる。

入力仕様を変更した場合、動的解析システムでは、この構造データファイルは 2 度読まれるため、次のサブルーチン INPUTE()でも、同様に変更する必要がある。ただし、こちらは構造図描画用であるため、入力データは読み捨てても良い。

```
C     ――――――――――――――――――――――――――――――――
C     ●   subroutine /INPUTE
C       ――――――――――――――――――――――――――――――――
C     ●    要素データ入力
C     ――――――――――――――――――――――――――――――――
      subroutine INPUTE()
       ・
       ・
      do 200 i=1,nelem
      read(5,*,end=1999,err=1998) mtype(i),youg(i),gsha(i),area(i),
     &         rix(i),riy(i),riz(i),asy,asz,am1,am2,
     &         anp,ampy,ampz,nm_type(i)
c――――――――――――――――――――★新規モデル New_Model のために拡張データ入力
      if(m_type .eq. 2 .and. nm_type .eq. 101) then
      read(5,*,err=9913,end=9918) d1,d2,d3,d4,d5
      endif
      if(mtype(i) .eq. 11.or.mtype(i) .eq. 15.or.
     *   mtype(i) .eq. 21.or.mtype(i) .eq. 31)then
      read(5,*,end=1999,err=1998) rd1,rd2,sgi,sg2,iix1,iix1
      endif
      if(mtype(i) .eq. 12.or.mtype(i) .eq. 22
     *   .or.mtype(i) .eq. 32) then
      read(5,*,end=1999,err=1998) rd1,rd2,sgi,sg2,iix1,iix1,iix1
      endif
       ・
       ・
      return
      end
```

この構造データファイルは、他のモジュールでも入力しており、整合性を取るためにも、他のシステムの構造データ入力部分を変更しなけれ

ばならない。それについては、他のマニュアルを参照されたい。

最後に、入力データを多数必要とする場合は、特別に入力データの仕様を設計し、そのデータを読み込むサブルーチンを作成する。ここでは、現在SPACEに組み込まれている修正バイリニア型モデルについて見てみよう。このサブルーチンでは、特別な構造体は、RO_work_s として設計されており、また、要素に関する構造体 element_s2_RO も、その内容を新たに設計したものである。この構造体 RO_work_s については後の節で説明する。構造体 element_s の替わりとなる構造体 element_s2_RO の内容は、プログラム内のコメントとして書き込まれている。この特別な入力データは、サブルーチン submain_dynamic_a()の中で、以下のようにコールされている。

```
c――――――――――――――――――――――――★修正 RO モデル領域セット
      n=Model_type.n_m_ro_model                                    ! 1
      if(n.ne.0) then
      nfix=5
      nfi=53
      call infile(nfi,nfix,ierr)                                   ! 2
      if(ierr.ne.0) then
      ierr_dat = 35
      call err_outf(ierr_dat)                                      ! 3
      return
      endif
      call RO_data_input(n,RO_work,Element,Parameter_C.n_element,ierr)  ! 4
      close(nfix)
      if(ierr.ne.0) then
      ierr_dat = 36
      call err_outf(ierr_dat)                                      ! 5
      return
      endif
      endif
       ・
       ・
```

1. 修正バイリニアと修正 RO モデルの個数をセットする。ここで、この個数がゼロでない場合は、これ以降の処理を実行し、特殊データを読み込むことになる。この個数は、サブルーチン Get_structure() の中で数えており、従って、特殊な構造体あるいはそれに付随する入力用のサブルーチンを必要とする場合は、Get_structure()の中でその個数を数えるコードを付け加える必要がある。
2. 修正バイリニアと修正 RO モデルに関する特殊データが保存されているファイルについて、サブルーチン infile()を用いて、存在のチェックとオープン処理を行う。

3．上記の処理でエラーがあった場合の処理を行う。エラー出力をサブルーチン err_outf()を用いて、エラーコード 35 と共にダンプファイルにエラーの内容を出力する。
4．サブルーチン RO_data_input()を用いて、修正バイリニアと修正 RO モデルに関する特殊データファイルを読み込む。
5．このデータファイルの内容を読み込む際、何らかのエラーがあった場合、サブルーチン err_outf()を用いて、エラーコード 36 と共にダンプファイルにエラーの内容を出力する。

例として、具体的に特殊データファイルを読み込むサブルーチン RO_data_input()を見てみよう。ここでは、新しく定義し直した構造体 elemennt_RO_s と新しく設計した構造体 RO_work_s が使用されている。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /RO_data_input
C       ―――――――――――――――――――――――――――――――――
C     ●   RO モデルデータ入力(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine RO_data_input(n,RO_work,Element,n_element,ierr)
      implicit real*8(A-H,O-Z)
      include "submainx.h"
      include "submain.h"
      record / RO_work_s     /  RO_work                              ! 1
      record / element_s2_RO / Element                               ! 2
      dimension RO_work(*),Element(*)                                ! 3
      integer RO_MODEL_NUMBER,TRI_MODEL_NUMBER
      real*8 BI_DEF
      integer n_ro
      data   BI_MODEL_NUMBER/11/
      data   RO_MODEL_NUMBER/12/
      BI_DEF=10.0**(-10.0)
C     ―――――――――――――――――――――――――――――――――
C     ●      3 次元せん断弾塑性モデル（モデル No.2）
C                修正 R-O，修正 Bi-Liner 型用
C     ―――――――――――――――――――――――――――――――――
c    要素数（モデル No.2  3 次元せん断弾塑性モデル：トリリニア型）
c   element_s 構造体と同一
c
c     structure / element_s2_RO/
c     integer  element_type   !  要素タイプ
c     integer  n_element      !  非線形要素番号
c     real*8   AK_1           !  初期剛性（内部計算）
c     real*8   dm1(4)         !  ダミー
c     real*8   AKu            !  軸方向バネ
c     real*8   dm2(2)         !  ダミー
c     real*8   Ar             !  積層ゴムの断面積
c     real*8   dm22           !  ダミー
c     integer  nm_damp        !  部材減衰の有無(1)
```

```
c     integer  nm_type        !  履歴モデルのタイプ
c     integer  No             !  R-0 ファイル履歴モデル番号
c     integer  dm3(9)         !  ダミー
c     real*8   ANP            !  軸方向耐力
c     real*8   AQPv           !  y 方向耐力
c     real*8   AQPw           !  z 方向耐力
c     real*8   dmm(3)         !  ダミー
c     real*8   dmm2(4)        !  ダミー
c     end structure
C     ―――――――――――――――――――――――――――――――――
c     RO_work    :structure
C     ―――――――――――――――――――――――――――――――――
c―――――――――――――――――――――――――――――――★データ入力(ok)
      ierr=0
      read(5,*,end=999,err=998) nn
      if(nn.le.0) goto 998
      do i=1,nn
      read(5,*,end=999,err=998) (RO_work(i).gan(j), j=1,3),
     *   (RO_work(i).arf1(j),j=1,6),
     *   (RO_work(i).arf2(j),j=1,6),
     *   (RO_work(i).beta(j),j=1,6),
     *   (RO_work(i).anyu(j),j=1,6)
      enddo
c―――――――――――――――――――――――――――――――★部材の線形剛性計算(ok)
c    Element(i).AKu          ! 軸剛性
c    Element(i).AK_1         ! 線形せん断剛性
c
      ii=0
      do i=1,n_element
      if(Element(i).element_type.eq.2) then
      if( Element(i).nm_type.eq.RO_MODEL_NUMBER.or.
     *   Element(i).nm_type.eq.BI_MODEL_NUMBER)then
      ii=ii+1
      if(ii.gt.nn) goto 999
c―――――――――――――――★初期剛性の計算挿入箇所(後で積層ゴム厚をかける必要があり)
      n_ro=Element(i).No
      if(Element(i).nm_type.eq.BI_MODEL_NUMBER)then
      Element(i).AK_1
     *   =0.7*RO_work(n_ro).arf1(1)*BI_DEF**(-0.3)
      else if(Element(i).nm_type.eq.RO_MODEL_NUMBER)then
      Element(i).AK_1
     *   =Element(i).Ar*RO_work(n_ro).arf1(1)
      endif
      endif
      endif
      enddo
      return
  998 continue
      ierr=1
      return
  999 continue
      ierr=2
      return
      end
```

1. このサブルーチンの中で使用する構造体 RO_work_s を、引数として受け渡された実際に存在する領域 RO_work に、record 文を用いて割り付ける。
2. 同じく、構造体 element_s2_RO を、引数として受け渡された実際に存在する領域 Element に、record 文を用いて割り付ける。
3. 構造体である RO_work と Element が共に配列であることを宣言する。

1.2.3 モデルの出力仕様

本節では、この新しい履歴モデルで計算した結果をファイルに出力する場合、その出力仕様について説明する。出力データは、他のモジュールで使用されており、変更する場合は、他のシステム、例えば動的プレゼンターなどをチェックし、整合性を取る必要がある。

まずは、せん断型モデルの標準出力仕様を見てみよう。解析結果を出力するサブルーチン Out_stress()は、submain_dynamic_a()の中で、コールされている。ここでは、せん断型に関連する部分のみ示す。

```
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"
      record / Member_s         / Member                              ! 1
      record / Element_s        / Element
      record / Out_section_s    / Out_section
      record / E_model6_real_s  / E_model6_real
       .
       .
      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
      istat = 0                        ! 現在ダミー　塑性状態          ! 2
```

```
      v(1)=Member(i).stress(1)          ! 軸力
      v(3)=Member(i).stress(2)          ! ｙ軸せん断力
      v(2)=Member(i).stress(3)          ! ｚ軸せん断力
      v(4)=0.
      write(iflz(5)) istat,(v(k),k=1,4)
      write(iflz(5)) istat,(v(k),k=1,4)
     elseif(Member(i).element_type.eq.3) then
c―――――――――――――――――――――――――――――★3 次元軸力弾塑性モデル
      •
      •
      else
      •
      •
      endif
      enddo
      return
      end
```

1. このサブルーチンの中で使用する構造体 Member と Element は、標準仕様で定義されている。これは、各種のモデルの出力をこのサブルーチンが受け持っているからであり、従って、せん断型として新たに設計した構造体の内容で出力することはできない。
2. せん断型モデルの出力は非常に単純で、軸力、y 軸のせん断力、z 軸のせん断力である。現在は、他のデータは全てダミーとなっており、ここに、新たなデータ出力項目を設けて他のモジュールで使用できるようにすることも可能である。当然のこと、他のモジュール、例えば、動的プレゼンターなどでそれに対応する処理を付け加える必要がある。

1.2.4 構造体の定義

本節では、新規の履歴モデルによって、新たな構造体を作成する必要がある場合について説明する。ただし、標準入力仕様で、しかも要素に関連する element_s 構造体と部材に関連する member_s 構造体が標準の構造体の内容で良い場合、あるいはせん断型モデルの標準型として設計した構造体 element_s2 や member_s2 を使用する場合は、新たに構造体を設計する必要はない。

まず、要素に関係する element_s と element_s2 を示そう。両者の要素並びで、対応する変数は同じ記憶領域を占めており、構造体配列の数は要素数分設定される。同様に、member_s と member_s2 は同じ動的記憶領域に、解析モデルの部材数分設定される。前節で示した入力プログラムの中で、構造体に入力データをセットしているが、それら

が構造体 element_s2 のどの変数に対応しているか比べてみると良い。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   element_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c    要素
      structure / element_s/
      integer  element_type   !  要素タイプ
      integer  n_element      !  非線形要素番号
      real*8   E              !  ヤング係数
      real*8   G              !  せん断係数
      real*8   A              !  断面積
      real*8   RIx            !  ねじり剛性
      real*8   RIy            !  ｙ軸断面二次モーメント
      real*8   RIz            !  ｚ軸断面二次モーメント
      real*8   ASy            !  各部材のＹ軸回りのせん断変形用等価断面積
      real*8   ASz            !  各部材のＺ軸回りのせん断変形用等価断面積
      real*8   AM(2)          !  単位長さ当たりの質量（1:第一 2:第二ステップ用）
      integer  nm_damp        !  部材減衰の有無
      integer  nm_type        !  （maxwell モデルでは、１：ｘ方向２：ｙ方向３：ｚ方向）
      integer  n_section(5)   !  断面番号
      integer  nm_section(5)  !  ファイバー数
      real*8   ANP            !  軸方向耐力
      real*8   AMPY           !  ｙ軸塑性モーメント
      real*8   AMPZ           !  ｚ軸塑性モーメント
      real*8   dmm(3)         !  ダミー
      real*8   i_rigid_length      !  i 端剛域長さ
      real*8   j_rigid_length      !  j 端剛域長さ
      real*8   i_shear_G           !  i 端せん断剛性
      real*8   j_shear_G           !  j 端せん断剛性
      end structure
C     ―――――――――――――――――――――――――――――――――
C     ●   3 次元せん断弾塑性モデル（モデル No.2）
C     ―――――――――――――――――――――――――――――――――
c    要素数（モデル No.2  3 次元せん断弾塑性モデル：トリリニア型）
c   element_s 構造体と同一
c
      structure / element_s2/
      integer  element_type   !  要素タイプ(6)
      integer  n_element      !  非線形要素番号
      real*8   AK_1           !  第一剛性
      real*8   AK_2           !  第二剛性
      real*8   AK_3           !  第三剛性
      real*8   Q_1            !  第一折れ点のせん断力
      real*8   Q_2            !  第二折れ点のせん断力
      real*8   AKu            !  軸方向バネ
      real*8   arf            !  武田モデルα(通常は 0.4)
      real*8   U_1            !  第一折れ点の変位
      real*8   U_2            !  第一折れ点の変位
      real*8   dm4            !  ダミー
      integer  nm_damp        !  部材減衰の有無(1)
      integer  nm_type        !  履歴モデルのタイプ
```

```
      integer  n_section(5)   !  断面番号
      integer  nm_section(5)  !  ファイバー数
      real*8   ANP            !  軸方向耐力
      real*8   AQPv           !  y 方向耐力
      real*8   AQPw           !  z 方向耐力
      real*8   dmm(3)         !  ダミー
      real*8   dmm2(4)        !  ダミー
      end structure
```

上の構造体 element_s2 をそのまま利用することも可能であるが、新規の履歴モデル用として、新たに設計することもできる。もし、必要な場合は、上に示した構造体 element_s2 を参照して作成されたい。この構造体は、要素データを保存するためのものであり、部材毎に計算途中で変化するデータを入れるものではない。これについては、構造体 member_s が用意されている。また、この構造体の全ての要素が使用可能となっておらず、上記の element_s2 の中で濃い文字で示した成分はシステムが使用する。無論、ここでは、ダミーとして定義している成分を別の変数として設定し、それらを使用することもできる。

次に、部材に関する構造体 member_s を示そう。上記の element_s と同様に、標準仕様の member_s とせん断型モデルの構造体 member_s2 がある。せん断型モデルの構造体 member_s2 では、ワーク領域を多く必要としないので、設定していても使用していない成分が多数存在する。ここでも、新たな構造体を設計したい場合は、この構造体 member_s2 を参照して作成されたい。

```
C     ―――――――――――――――――――――――――――――――
C     ●   member_s 構造体
C     ―――――――――――――――――――――――――――――――
c
c     部材
      structure / member_s/
      integer  nm_element         !  要素番号（入力した要素番号を示す）
      integer  element_type       !  要素タイプ番号
      integer  n_model            !  モデルの入れ物番号
      integer  n_model_type       !  モデル別の通し番号
      integer  n_element_type     !  要素タイプ別通し番号
      integer  analysis_3D        !  解析型 0:3D 1:2D(x-z) 2:2D(y-z)
      integer  nm_so              !  部材の層番号
      integer  nm_dll_element     !  DLL を用いた要素か（０；システム内要素、１：DLL 要素）
      integer  nm_point(2)        !  節点番号
      integer  irest(12)          !  部材両端の自由度番号表
      integer  ijp(2)             !  両端節点への結合状況（0:剛結合 1:ピン結合）
      integer  nm_analysis        !  部材解析種別（-1:弾性解析、その他：通常解析）
      integer  nm_group           !  部材グループ
      integer  nm_local_coord(2)  !  局所座標系の有無とその回転行列の番号
      integer  nm_damp            !  部材減衰の有無とその減衰行列の番号
      real*8   alength            !  長さ
```

```
      real*8   i_rigid_length     !  i 端剛域長さ
      real*8   j_rigid_length     !  j 端剛域長さ
      real*8   i_shear_G          !  i 端せん断剛性
      real*8   j_shear_G          !  j 端せん断剛性
      real*8   rot_x              !  部材主軸の回転角度（度）
      real*8   force(12)          !  部材両端の部材端力（釣合座標系）
      real*8   stress(18)         !  部材両端,中央の応力（部材座標系）
      real*8   an_stress(10)      !  部材軸力（部材座標系）
      integer  d_stat(3)          !  断面の弾塑性状態　(1) i 端 (2) j 端　(3) 中央
      real*8   an_vv(10)          !  部材軸力（計算用内部変位ｖ）
      real*8   an_ww(10)          !  部材軸力（計算用内部変位w）
      end structure
C     ―――――――――――――――――――――――――――――――――
C     ●　3 次元せん断弾塑性モデル用（モデル No.2）member_s 構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / member_s2/
      integer  nm_element         !  要素番号
      integer  element_type       !  要素タイプ
      integer  n_model            !  モデルの入れ物番号
      integer  n_model_type       !  モデル別の通し番号
      integer  n_element_type     !  要素タイプ別番号
      integer  analysis_3D        !  解析型 0:3D 1:2D(x-z) 2:2D(y-z)
      integer  nm_so              !  部材の層番号
      integer  nm_dll_element     !  DLL を用いた要素か（０；システム内要素、１：DLL 要素）
      integer  nm_point(2)        !  節点番号
      integer  irest(12)          !  部材両端の自由度番号表
      integer  istat_v            !  y 方向:履歴特性の状態(*)
      integer  istat_w            !  z 方向:履歴特性の状態(*)
      integer  nm_analysis        !  部材解析種別
      integer  nm_group           !  部材グループ
      integer  nm_local_coord(2)  !  局所座標系の有無とその回転行列の番号
      integer  nm_damp            !  部材減衰の有無とその減衰行列の番号
      real*8   alength            !  長さ
      real*8   i_rigid_length     !  i 端剛域長さ
      real*8   j_rigid_length     !  j 端剛域長さ
      real*8   i_shear_G          !  i 端せん断剛性
      real*8   j_shear_G          !  j 端せん断剛性
      real*8   rot_x              !  部材主軸の回転角度（度）
      real*8   force(12)          !  部材両端の部材端力（釣合座標系）
      real*8   stress(6)          !  部材中央の応力（部材座標系）
      real*8   AKv_tan            !  接線剛性(*)
      real*8   AKw_tan            !  接線剛性(*)
      real*8   dmv(10)            !  y 方向耐力:履歴特性で使用(*)
      real*8   dmw(10)            !  z 方向耐力:履歴特性で使用(*)
      integer  d_stat(3)          !  断面の弾塑性状態　(1) i 端 (2) j 端　(3) 中央
      real*8   an_vv(10)          !  部材軸力（計算用内部変位ｖ）
      real*8   an_ww(10)          !  部材軸力（計算用内部変位w）
      end structure
```

標準仕様の構造体を使用する場合は、新たな構造体を必要としないが、その新規モデルに適合した構造体を設計する場合は、その構造体

を新たなヘッダーファイルに作成する必要がある。このヘッダーファイルを New_submain.h としよう。読者はまずこの New_submain.h ファイルを作り、この中に新たな構造体を書き込むことになる。この構造体をどのように利用するかについては、後節で説明する。

それでは、入力データ数が多くて標準仕様では保存するデータ数が不足し、新たな構造体を設計する必要がある場合について考えよう。これも前節で示した修正バイリニアモデルで説明する。まず、新たに設計した構造体 RO_work_s を示す。

```
C     ―――――――――――――――――――――――――――――――
C     ●   履歴モデル    RO/修正 Bi-Liner モデル：せん断要素用
C                                          Work エリア構造体
C     ―――――――――――――――――――――――――――――――
c
c     部材
      structure / RO_work_s/
      real*8   gan(3)              !  適用歪
      real*8   arf1(6)             !  歪１の骨格曲線の定義
      real*8   arf2(6)             !  歪２の骨格曲線の定義
      real*8   beta(6)             !  等価減衰定数の定義
      real*8   anyu(6)             !  履歴剛性の定義
      end structure
```

この構造体は、動的に確保する配列であることから、サブルーチン submain_dynamic_a()の中で以下のように確保する必要がある。ここで、確保するかしないか判別するパラメータ Model_type.n_m_ro_model は、先に説明した修正バイリニアモデルと修正 RO モデルの数である。

```
c――――――――――――――――――――――――――――――★Model_No.2 3 次元せん断弾塑性モデル
      n=Model_type.n_m_ro_model
      if(n.ne.0) then
      ALLOCATE (RO_work(n))
      endif
```

手続きとしては、動的確保の他に、動的領域であることの宣言、使用後の領域解放を行うことになる。構造体を新たに設計、使用する場合は、これらの処理を必ず既存プログラムの中に書き込む必要が生じる。

以上で、前段階の調査と手続きが終了し、これから、具体的にせん断系の履歴モデルを組み込むことになる。

1.2.5 必要となるサブルーチン

本節では、この履歴特性 New_Model 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。ただし、この中で標準のせん断型モデルのサブルーチンを転用できる場合は、そのサブルーチンを使用すれば良い。まず、上記４つの標準的なせん断型モデルに関するサブルーチンを見てみよう。

最初は、線形剛性行列を求めるサブルーチン Cal_lin_stiff_M2()であり、次に示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_lin_stiff_M2
C       ―――――――――――――――――――――――――――――――――
C     ●   Model_No.2 3 次元せん断弾塑性モデル
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_lin_stiff_M2(Member,Element,ak_linear)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s2    / Member
      record / element_s2   / Element
      dimension ak_linear(12,12)
      data    BI_MODEL_NUMBER/11/
      data    RO_MODEL_NUMBER/12/
c―――――――――――――――――――――――――――――――――――――――――――――
c           Element             structure
c           Member              structure
c           ak_linear           real*8   線形剛性行列
c―――――――――――――――――――――――――――――――――――――――――――――
      do i=1,12
      do j=1,12
        ak_linear(j,i) = 0.0
      end do
      end do
      ak=Element.AKu
      ak_linear(1,1)= ak
      ak_linear(1,7)=-ak
      ak_linear(7,7)= ak
      ak_linear(7,1)=-ak
      if (Element.nm_type.eq.BI_MODEL_NUMBER) then
        ak=Element.AK_1*Member.alength
      else if(Element.nm_type.eq.RO_MODEL_NUMBER) then
        ak=Element.AK_1/Member.alength
      else
        ak=Element.AK_1
      endif
```

```
      ak_linear(2,2)= ak
      ak_linear(2,8)=-ak
      ak_linear(8,8)= ak
      ak_linear(8,2)=-ak
      ak_linear(3,3)= ak
      ak_linear(3,9)=-ak
      ak_linear(9,9)= ak
      ak_linear(9,3)=-ak
c―――――――――――――――――――――――――――――――★履歴特性の初期設定

      Member.AKv_tan=ak              ! Member.AKv_tan=Element.AK_1
      Member.AKw_tan=ak              ! Member.AKw_tan=Element.AK_1
      Member.istat_v=0
      Member.istat_w=0
      return
      end
```

このサブルーチンを見れば分かるように、要素に関する構造体の中の成分で、軸方向剛性 Element.AKu とせん断剛性 Element.AK_1 に所定の値がセットされていれば、このまま使用できることになる。

次に、非線形剛性を求めるサブルーチン Cal_nonlin_stiff_M2()について調査する。このプログラムも以下に示すように線形剛性のプログラムと同様、構造体成分 Element.AKu と Member.AKv_tan、Member.AKw_tan を他のプログラムでセットしておけば転用可能となる。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_nonlin_stiff_M2
C       ―――――――――――――――――――――――――――――――――
C     ●   Model_No.2 3 次元せん断弾塑性モデル
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_nonlin_stiff_M2(Member,Element,ak)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s2    / Member
      record / element_s2   / Element
      dimension ak(12,12)
C     ―――――――――――――――――――――――――――――――――
c           Element             structure
c           Member              structure
c           ak_linear           real*8   線形剛性行列
c―――――――――――――――――――――――――――――――――――――――――――
      do i=1,12
      do j=1,12
      ak(j,i) = 0.0
      end do
      end do
      akk=Element.AKu
      ak(1,1)= akk
      ak(1,7)=-akk
```

```
      ak(7,7)= akk
      ak(7,1)=-akk
      akk=Member.AKv_tan
      ak(2,2)= akk
      ak(2,8)=-akk
      ak(8,8)= akk
      ak(8,2)=-akk
      akk=Member.AKw_tan
      ak(3,3)= akk
      ak(3,9)=-akk
      ak(9,9)= akk
      ak(9,3)=-akk
      return
      end
```

次に、応力を求めるサブルーチンを調べよう。このプログラムも、せん断型モデルでは、特殊な応力計算を行わない限り、非線形剛性行列を求めるプログラムと同様、構造体成分 Element.AKu と Member.AKv_tan、Member.AKw_tan を他のプログラムでセットしておけば転用可能となる。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_stress_M2
C     ―――――――――――――――――――――――――――――――
C     ●   部材の応力計算(次元せん断弾塑性モデル（モデル No.2）)
C     ―――――――――――――――――――――――――――――――
      subroutine Cal_stress_M2(Member,Element,vv)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s2    / Member
      record / element_s2   / Element
      dimension vv(12)
C     ―――――――――――――――――――――――――――――――
c           Element             structure
c           Member              structure
c           vv               real*8  増分変位
C     ―――――――――――――――――――――――――――――――
      Member.stress(1)=Member.stress(1)+Element.AKu*
     *                 (vv(7)-vv(1))
      Member.stress(2)=Member.stress(2)+Member.AKv_tan*
     *                 (vv(8)-vv(2))
      Member.stress(3)=Member.stress(3)+Member.AKw_tan*
     *                 (vv(9)-vv(3))
      return
      end
```

以上のように、せん断モデルの履歴特性では、一般的に、標準仕様の3つのサブルーチンがそのまま使用することができることが分かる。後は、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が、このサブルーチンについては、次節で説明しよう。

1.2.6 モデルの階層構造とサブルーチンの組み込み

せん断型モデルでは、submain_dynamic_a()から Check_stress()がコールされ、その中で応力の弾塑性チェックのプログラムがコールされることになる。まず、Check_stress()の中で、このせん断モデルに関係する部分を抜き出してみよう。このサブルーチンの中には、部材モデルの階層構造が見られる。

```
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( )
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
   12 continue
c――――――――――――――――――――――――――――――――★Model_No.2 3 次元せん断弾塑性モデル
      if(N_analysis.ge.9) then
      call Cal_check_stiff_M2(Member(i),Element(ie),RO_work,
```

1.3 部材モデルの組み込み方法

1.3.1 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 に組み込むことする。この任意型部材モデルを通して、部材モデルの組み込み手法を説明する。この例題は静的縮合モデルであり、そのため多くの処理を必要としているが、通常の部材モデルではこれほどの手続きを必要としていない。したがって、静的縮合モデル以外の通常の部材モデルを組み込むために必要となる情報のみ得たい場合は、第 1.3.11 節以降を読まれたい。

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 SPACE Ver.3.00 では標準モデルとして既に組み込まれている。

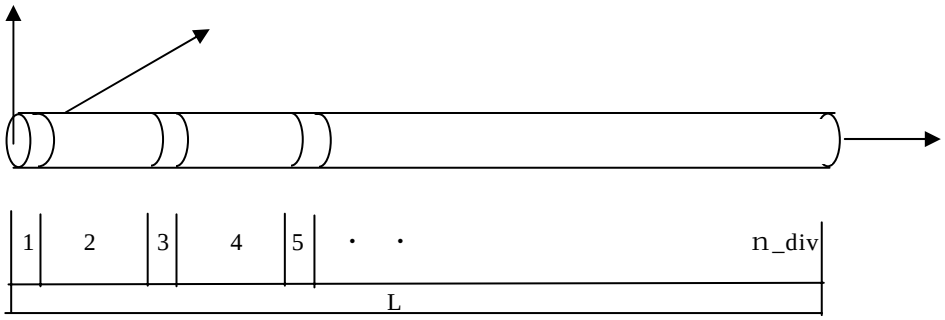


図 1-1 任意型静的縮合モデル

ここでは、任意型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、木質構造物用接合ばねの５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、プログラムからこれら構造体へのアクセス手法

ただし、通常の部材モデルでは、上記のような面倒な手続きは必要としないので、容易に組み込むことができよう。

1.3.2 静的縮合モデル設定ファイルの仕様と構造体

本節では、新規に設定した任意型部材モデルで、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   任意型静的縮合モデル
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、新規任意型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、この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任意型部材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、先に示した新規の任意型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、この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、新規任意型部材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 に新規任意型部材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 を用いて新規任意型部材モデルに必要なデータを作り出す。内容はマニュアル：動的解析編の第 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
```

1.3.3 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。

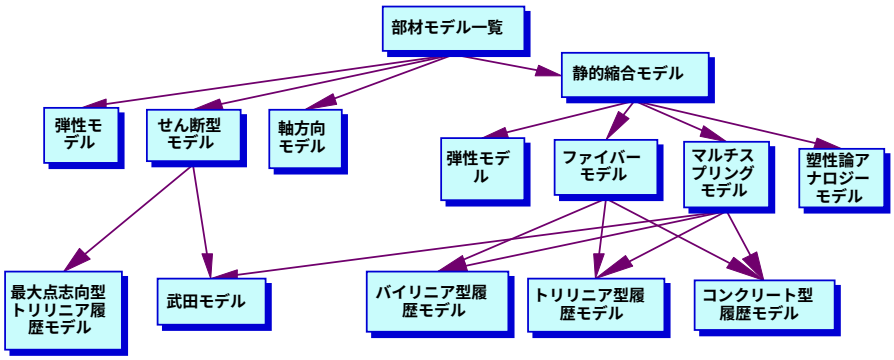


図 1-2　部材モデルの階層構造

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 1-2 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 1-1 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 1-1 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
```

新規の任意型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、新規任意型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                    !  現在ダミー
      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、この新規任意型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任意型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

含まれる新規任意型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、新規任意型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 1-2 に示す。

表 1-2 部材モデル：任意型静的縮合モデル 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 1-2 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。新規任意型モデルの構築は、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。新規任意型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 1-3 にまとめる。表中の＊印は他のモデルと共用であり、他は新規任意型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、任意型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、この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 1-3 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、新規任意型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、新規任意型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、新規任意型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、新規任意型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、新規任意型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などは新規任意型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。無論、通常の部材モデルでは、このような複雑な手続きを必要としない。

1.3.4 部材モデルの入力仕様

本節では、入力関連で新規の任意型静的縮合モデルで必要となるコードを検討する。最初に、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、新規任意型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、新規任意型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、新規任意型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――――――――――――――――――――――★新規任意型の静的縮合モデルを教える
      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、新規任意型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―――――――――――――――――――――――――★新規任意型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              ! 新規任意型モデルの要素内エレメントの最初の番地
      Element(i).n_section(2) =nxx                ! 新規任意型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――――――――――――――――――――――――★新規任意型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)                    ! 新規任意型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――――――――――――――――――――――――★新規任意型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――――――――――――――――――――――――★新規任意型モデル 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――――――――――――――――――――――――――――――――★新規任意型モデル 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         ! 数新規任意型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
```

通常の部材モデルで、入力仕様を設計するわけであるが、その仕様が上記のサブルーチン Get_structure()内の第１レコードと第２レコードで入力可能であれば、特に難しい設定を行う必要はない。ただし、ファイバーデータのような特殊断面データを必要とする部材モデルでは、以下に示すサブルーチンでデータを取り込まなければならない。

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。新規任意型モデルでは、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―――――――――――――――――――★新規任意型モデル 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―――――――――――――――――――――★新規任意型モデル 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————————————————————————★新規任意型モデル 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.or.
     *      E_model_fiber(nmm).nm_type.eq.15.or.E_model_fiber(nmm).nm_type.eq.16) 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。

1.3.5 部材モデルの出力仕様

本節では、新規任意型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。最初に、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、この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、任意型部材モデルは、番号 51-70 であり、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。同様に、通常の部材モデルでも部材モデル番号に従ってこのルーチンの中に組み込むことになる。

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―――――――――――――――――――――――★新規任意型モデル 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。新規に設計した任意型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。当然、新しく設計する部材モデル番号にしたがって、出力コードを挿入することになる。

静的縮合モデルに対して、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。新規任意型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。

1.3.6 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。通常の部材モデルも同様にこれらのサブルーチンは必要となる。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。新規任意型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、新規任意型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、任意型モデル番号 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モデルの線形剛性が受け渡される。

　新規任意型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. 先に設計した新規任意型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. この新規任意型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任意型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
c      call MS_Model_GIx(ak,alength(im),Member,Element,
c     *      E_modelx,E_model_fiber,
c     *      M_modelx,M_model_fiber,
c     *      Bilinear_work,Trilinear_work,Concrete_work)
c      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――――――――――――――――――――――――――――――――★木質構造接合部要素
      call wood_Joint_GIx(ak,Menber,Element,E_modelx,E_model_fiber,
     *                    M_modelx,M_model_fiber,Trilinear_work)
      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する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
```

1.3.7 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。新規任意型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、新規任意型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、任意型部材番号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任意型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、任意型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
c      call MS_Model_Gx(ak,alength(im),Member,Element,
c     *      E_modelx,E_model_fiber,
c     *      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――――――――――――――――――――――――――――――――――――――――★接合部ばねモデル要素
      call Wood_Joint_GK(ak,E_modelx,E_model_fiber,M_modelx,
     *                   M_model_fiber)
      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()は接線剛性行列を計算する。これについては、マニュアル：動的解析編の第 5 章を参照されたい。

```
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
```

1.3.8 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、部材内エレメントの力―変位の履歴、あるいはファイバーの応力―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。そこで、適切な階層で新規任意型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
c      call MS_checkx(N_analysis,
c     *      mem_x,alength(i),Member,Element,
c     *      E_modelx(imm),E_model_fiber,M_modelx(immm),M_model_fiber,
c     *      Bilinear_work,Trilinear_work,Concrete_work,vvx,
c     *      M_Fiber_Work(nmmx+i).an_vv, M_Fiber_Work(nmmx+i).an_ww,
c     *      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-------------------------------★接合部ばねモデル要素
      call Wood_Joint_checkx(Control,N_analysis,Member,
     *                       Element,E_modelx(imm),E_model_fiber,
     *                       M_modelx(immm),M_model_fiber,Bilinear_work,
     *                       Trilinear_Work,vvx)
      goto 100
      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、アナロジーモデルなどが登録されている。
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
```

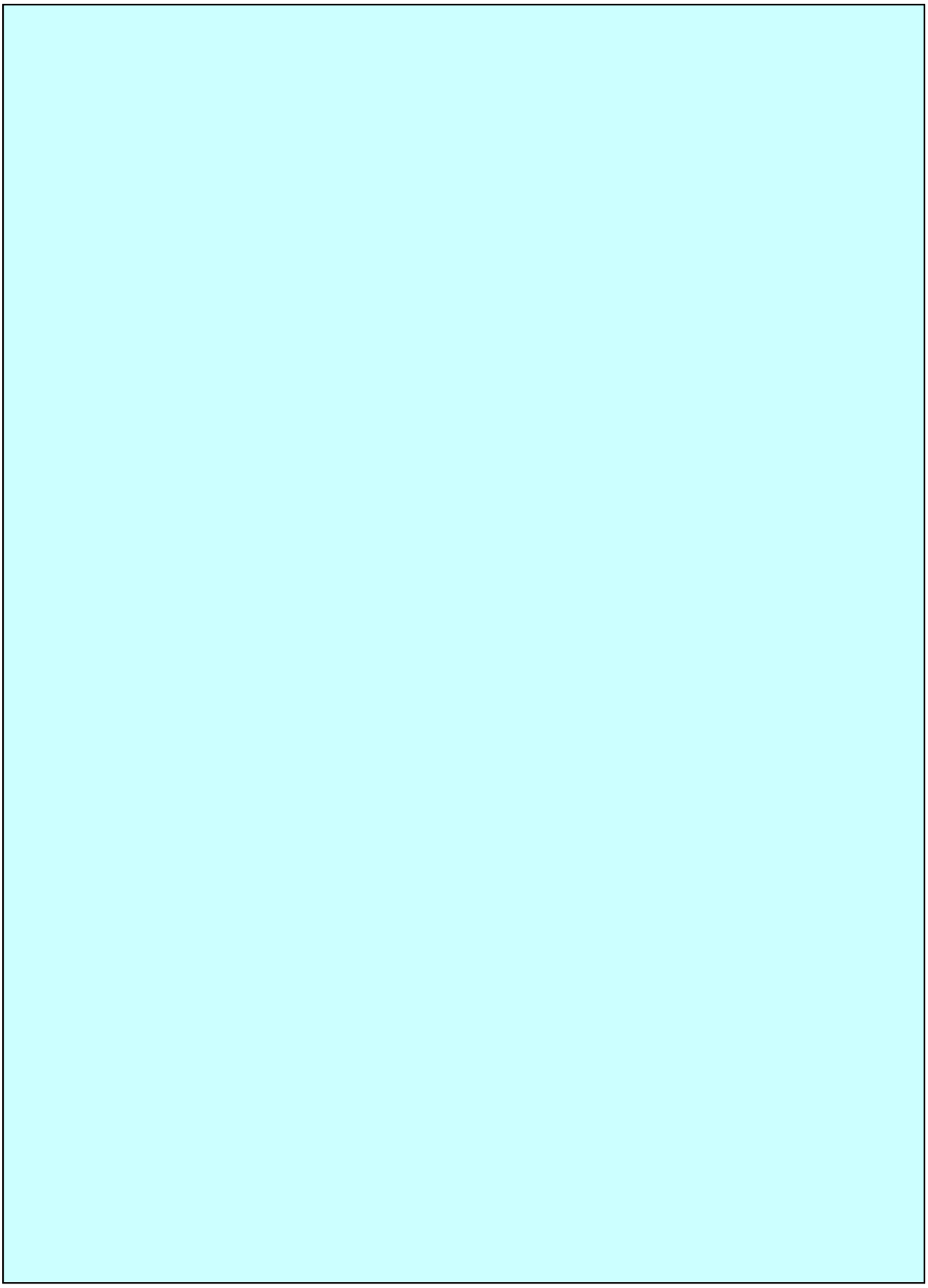
```
C     ―――――――――――――――――――――――――――――――――
      subroutine Fiber_checkx(Control,idiv,N_analysis,
     *      mem_x,Alength,Member,Element,
     *      E_modelx,E_model_fiber,M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,an_vv,an_ww,
     *       n_dstat)
      implicit 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 ( 10,20,30,40,50,60,70,80,90,99),nm_type
      elseif(nm_type.ge.10.and.nm_type.le.20)then
      goto ( 110,120,130,140,150,160),nm_type-14
      endif
   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(1))
      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),Trilinear_work(nmx).P1(2),
     *          Trilinear_work(nmx).P1(3),Trilinear_work(nmx).P1(4),
     *          Trilinear_work(nmx).P1(5),Trilinear_work(nmx).P1(6))
      if(Trilinear_work(nmx).i_stat.eq.2.or.
     *   Trilinear_work(nmx).i_stat.eq.3.or.
     *   Trilinear_work(nmx).i_stat.eq.4.or.
     *   Trilinear_work(nmx).i_stat.eq.5) 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(1),Bilinear_work(nmx).P1(2),
     *          Bilinear_work(nmx).P1(3),
     *         M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *         M_model_fiber(nnm).du_c)
      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),
     *          Trilinear_work(nmx).P1(3),Trilinear_work(nmx).P1(4),
     *          Trilinear_work(nmx).P1(5),Trilinear_work(nmx).P1(6),
     *          Trilinear_work(nmx).P1(7),
     *         M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *         M_model_fiber(nnm).du_c)
      if(Trilinear_work(nmx).i_stat.eq.2.or.
     *   Trilinear_work(nmx).i_stat.eq.3.or.
     *   Trilinear_work(nmx).i_stat.eq.4.or.
     *   Trilinear_work(nmx).i_stat.eq.5) 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).E_3,E_model_fiber(nn).Q_1,
     *          E_model_fiber(nn).Ec_1,E_model_fiber(nn).Ec_2,
     *         E_model_fiber(nn).Ec_3,E_model_fiber(nn).Qc_1,
     *          du,M_model_fiber(nnm).d_stress_x,
     *          Bilinear_work(nmx).P1(1),Bilinear_work(nmx).P1(2),
```

```
     *          Bilinear_work(nmx).P1(3),
     *          M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *         M_model_fiber(nnm).du_c,
     *          M_model_fiber(nnm).E_5,M_model_fiber(nnm).Ec_5)
      if(Bilinear_work(nmx).i_stat.eq.2.or.
     *   Bilinear_work(nmx).i_stat.eq.3.or.
     *   Bilinear_work(nmx).i_stat.eq.4.or.
     *   Bilinear_work(nmx).i_stat.eq.5) 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).E_4,
     *         E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *          E_model_fiber(nn).Ec_1,E_model_fiber(nn).Ec_2,
     *          E_model_fiber(nn).Ec_3,E_model_fiber(nn).Ec_4,
     *          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),Trilinear_work(nmx).P1(4),
     *          Trilinear_work(nmx).P1(5),Trilinear_work(nmx).P1(6),
     *          Trilinear_work(nmx).P1(7),
     *          M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *           M_model_fiber(nnm).du_c,
     *          M_model_fiber(nnm).E_5,M_model_fiber(nnm).Ec_5)
      if(Trilinear_work(nmx).i_stat.eq.2.or.
     *   Trilinear_work(nmx).i_stat.eq.3.or.
     *   Trilinear_work(nmx).i_stat.eq.4.or.
     *   Trilinear_work(nmx).i_stat.eq.5.or.
     *   Trilinear_work(nmx).i_stat.eq.6.or.
     *   Trilinear_work(nmx).i_stat.eq.7) iistat = iistat + 1
      goto 100
90       continue
c――――――――――――――――――――★降伏棚を有する対称バイリニア型(タイプ 1,2)（移動＋等方効果用）
         goto 100
99       continue
c――――――――――――――――――――★降伏棚を有する対称トリリニア型(タイプ 1,2)（移動＋等方効果用）
         goto 100
110      continue
c――――――――――――――――――――――――★鉄筋用履歴モデル(タイプ 1,2)
      call reinforcement_b(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).Q_1,
     *          du,M_model_fiber(nnm).d_stress_x,
     *          Trilinear_work(nmx).P1(1),Trilinear_work(nmx).P1(2),
     *          Trilinear_work(nmx).P1(3),Trilinear_work(nmx).P1(4),
     *          Trilinear_work(nmx).P1(5),Trilinear_work(nmx).P1(6),
     *          Trilinear_work(nmx).P1(7),Trilinear_work(nmx).P1(8),
     *          M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *           M_model_fiber(nnm).du_c,
     *          M_model_fiber(nnm).E_5,M_model_fiber(nnm).Ec_5)
      if(Trilinear_work (nmx).i_stat.eq.1.or.
```

```
     *   Trilinear_work (nmx).i_stat.eq.2.or.
     *   Trilinear_work (nmx).i_stat.eq.5.or.
     *   Trilinear_work (nmx).i_stat.eq.6) iistat = iistat + 1
         goto 100
120      continue
c―――――――――――――――――――――――――★木質構造材用履歴モデル
         call Wood_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,
     *          E_model_fiber(nn).Ec_1,E_model_fiber(nn).Ec_2,
     *          E_model_fiber(nn).Ec_3,
     *          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),
     *          M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *           M_model_fiber(nnm).du_c)
      if(Trilinear_work(nmx).i_stat.eq.1.or.
     *   Trilinear_work(nmx).i_stat.eq.2.or.
     *   Trilinear_work(nmx).i_stat.eq.5.or.
     *   Trilinear_work(nmx).i_stat.eq.6.or.
     *   Trilinear_work(nmx).i_stat.eq.8) iistat = iistat + 1
         goto 100
130      continue
c―――――――――――――――――――――――――★
         goto 100
140      continue
       •
       •
         goto 100
c―――――――――――――――――――――――――★弾塑性解析終了
      endif
      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
```



1.3.9 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、任意型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ル
      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、新規任意型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、新規任意型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。

1.3.10 部材モデルの組み込みの復習

前節では、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．ファイバーデータ入力の条件に新規任意型モデルを加える。

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、動的解析に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
```

1.3.11 通常部材モデルの組み込み法

任意型の部材モデルの組み込みは、多数のサブルーチンを用意し、多くの手続きを必要とした。部材モデル 11 のような静的縮合型の部材モデルは、任意型の部材モデルに比較すると格段に少ない手続きで組み込むことは可能であるが、やはり、かなり複雑な手続きを必要とする。今後は、任意型モデルに新規のエレメントを追加して用いるほうが、拡張性があり、しかも組み込みが容易である。この組み込み方法につては、次節で説明する。

静的縮合型でない通常の部材モデルの組み込み法は、せん断型モデルとほとんど同じである。前に説明したように、一般的な組み込み方法の順序は次のようである。

１）モデルを設計し、そのモデルに登録番号を付ける。
２）データの入力仕様の設計と現在の仕様との調整を行う。
３）出力仕様の設計とプレゼンターとの関係を調査する。
４）特殊な構造体を必要とするかどうか調査する。構造体を設計する場合は、他の構造体との整合性を確認する。また、その構造体の定義と構造体配列の宣言と確保、及び、解放処理を行う場所を確認する。
５）新規に必要となるサブルーチンを設計し、作成する。
６）モデルの階層構造を理解し、どのレベルに組み込むかを調査する。
７）既存のサブルーチンとのインターフェイス、並びに既存のサブルーチンの追加・変更部分の調査を行う。
８）設計した新規サブルーチンを SPACE の適切な位置に組み込む。
９）ここから実際にシステムを動作させ、組み込まれた新規任意型モデルが設計した仕様を満足するかどうか動作確認する。最初に設計したサブルーチンの引数を出力し、適切な値となっているかどうか、チェックする。
10）次に、簡単なモデルで履歴などをチェックする。
11）複雑な解析モデルを用いて、整合性をチェックする。
12）他のモデルとの整合性をチェックする。

必要となる手続きは、せん断型モデルとほとんど同じである。最終的に、部材モデルの履歴を管理するサブルーチンの組み込みは、次のサブルーチン Check_stgress()で行うことになる。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Check_stress
```

```
C     ―――――――――――――――――――――――――――――――――――
C     ●　部材の塑性状態をチェックする(ok)
C     ―――――――――――――――――――――――――――――――――――
      subroutine Check_stress(Control,N_analysis,Point,
     *     ak_nonlinear,Member,n_member,
     *     Model_type,Element,past_disp_point,disp_point,rot_memb,
     *     E_model6_real,E_model7_real,E_model_fiber,M_model_fiber,
     *     E_model11, M_model11,
     *     E_model12, M_model12,
     *     E_model13, M_model13,
     *     E_model15, M_model15,
     *     E_model21, M_model21,
     *     E_model22, M_model22,
     *     E_model31, M_model31,
     *     E_model32, M_model32,
     *     E_model33, M_model33,
     *     MSS_work,
     *     Bilinear_work,Trilinear_work,Concrete_work,RO_work,
     *     work1_elememt,work2_elememt, work1_member, work2_member,
     *     S_comp_model,E_modelx,M_modelx,
     *     E_fiber_work,M_fiber_work)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h"
      record /control_s    / Control
      record / point_s     / Point
      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 Point(*)
       ・
       ・
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
c――――――――――――――――――★剛床座標変換
       do k=1,2
       i1=Member(i).nm_point(k)
       if(Point(i1).n_group_gouyuka.ne.0) then
       k1=6*(k-1)+1
       call Set_gtrans_u(1,v(k1),Point(i1).coord_local(1),
     *                                 Point(i1).coord_local(2))
       call Set_gtrans_u(1,vp(k1),Point(i1).coord_local(1),
     *                                 Point(i1).coord_local(2))
       endif
      enddo
c――――――――――――――――――★座標変換
      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
      do j=1,6
      f(j)=-Member(i).stress(j)
      enddo
      do j=7,12
      f(j)=Member(i).stress(j)
      enddo
      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)=ff(j)
      enddo
      goto 100
   12 continue
c―――――――――――――――――――――――――――――――★Model_No.2 3 次元せん断弾塑性モデル
      if(N_analysis.ge.9) then
      call Cal_check_stiff_M2(Member(i),Element(ie),RO_work,
     *     vv,vpp )
      endif
      do j=1,3
      f(j)=-Member(i).stress(j)
      f(j+6)=-f(j)
```

```
      enddo
      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)=ff(j)
      enddo
      goto 100
   13 continue
c―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.3 3 次元軸力弾塑性モデル
      iee = Member(i).n_model_type
      call Cal_check_stiff_M3(Member(i),Element(ie),
     *     vv,vpp,N_analysis)
      do j=1,12
      f(j)=0.
      enddo
      do j=1,3
      f(j)=-Member(i).stress(j)
      f(j+6)=Member(i).stress(j+3)
      enddo
      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)=ff(j)
      enddo
      goto 100
   14 continue
c―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.4 3 次元ケーブル弾塑性モデル
      call Cal_check_stiff_M4(Member(i),Element(ie),
     *     vv,vpp ,N_analysis)
      f(1)=-Member(i).stress(1)
      f(7)=-f(1)
      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)=ff(j)
      enddo
      goto 100
   15 continue

c―――――――――――――――――――――――――――――――――――――――――――――――――――――――――――――★Model_No.5 3 次元免振モデル
      if(N_analysis.ge.9) then
      iee=Member(i).n_model_type
      call Cal_check_stiff_M5(Member(i),Element(ie),MSS_work(iee),
     *     vv,vpp,Model_type.n_spring,Model_type.cosin(1,1) )
      endif
      do j=1,3
      f(j)= -Member(i).stress(j)
      f(j+6)=-f(j)
      enddo
      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)=ff(j)
      enddo
      goto 100
   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
      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_stiff_M11(Control,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)
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
  112 continue
c――――――――――――――――――――――――――――――――――★Model_No.12 両端、中央ファイバーモデル
      call Cal_check_stiff_M12(Control,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)
      vv11=vv(8)-vv(2)  ! 非線形剛性計算のための変位セット
      vv22=vv(9)-vv(3)
c     write(76,'(a,10f12.4)') ' an ',Member(i).stress(7),vv11,vv22
c――――――――――――――――――――――――――――――――――★部材の両端節点力を釣合系に変換
c――――――――――――――――――――――――――――――――――★非線形軸力の計算
c       if(N_analysis.ne.9.and.N_analysis.ne.7)
c     *  call nonlinear_stress_F(Element(ie),Member(i),vv,f)
      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
  113 continue
c――――――――――――――――――――――――――――――――――★Model_No.13 両端 MS モデル
      call Cal_check_stiff_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)
      do j=1,12
     Member(i).force(j)=Member(i).force(j)+ff(j)
      enddo
      goto 100
  114 continue
c―――――――――――――――――――――――――――――――――★Model_No.14 両端、中央 MS モデル
       call Cal_check_stiff_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)
      do j=1,12
     Member(i).force(j)=Member(i).force(j)+ff(j)
      enddo
      goto 100
  115 continue
c―――――――――――――――――――――――――――――――――★Model_No.15 幾何学非線形+弾塑性型有限要素モデル
      call Cal_check_stiff_M15(Control,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
      do j=1,6
      f(j)=-Member(i).stress(j)
      enddo
      do j=7,12
      f(j)=Member(i).stress(j)
      enddo
      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)= ff(j)
      enddo
      goto 100
  116 continue
c―――――――――――――――――――――――――――――――――★Model_No.16 両端アナロジーモデル
      call Cal_check_stiff_M31(N_analysis,
     *      mem,Model_type,Member(i),Element(ie),
     *      E_model31, E_model_fiber,
     *      M_model31, M_model_fiber,
     *      vv,vpp,f)
c―――――――――――――――――――――――――★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
  117 continue
c―――――――――――――――――――――――――――――――――★Model_No.17 両端、中央アナロジーモデル
```

```
      call Cal_check_stiff_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)
      do j=1,12
     Member(i).force(j)=Member(i).force(j)+ff(j)
      enddo
      goto 100
  118 continue
c-------------------------------★Model_No.18
      call Cal_check_stiff_M13(Control,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)
      do j=1,12
     Member(i).force(j)=Member(i).force(j)+ff(j)
      enddo
      goto 100
  119 continue
c-------------------------------★Model_No.19 両端、中央アナロジーモデル
c      call Cal_check_stiff_M33(N_analysis,
c     *      mem,Model_type,Member(i),Element(ie),
c     *      E_model33, E_model_fiber,
c     *      M_model33, M_model_fiber,
c     *      vv,vpp,f,me_x)
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
  120 continue
c-------------------------------★Model_No.20

      goto 100
c-------------------------------★Model_No.11 両端ファイバーモデル
      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―――――――――――――――――――――――――――――――――★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)


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

      end do
      return
      end
```

上記のサブルーチンの中で、コメントでモデル No.8、9、10 の位置に新規に作成した部材の履歴モデルを組み込むことになる。

1.3.12 任意型静的縮合モデルのエレメント組み込み法

任意型の部材モデルでの新規エレメントの組み込みは、入力仕様と出力仕様の決定、さらに必要なサブルーチンを用意することから始まる。まず、入力仕様について検討する。サブルーチン Get_structure()では、任意型の部材モデルを用いるため、ここでは変更するコードはない。ただし、追加する任意型部材モデルのエレメントで、特殊断面データを必要としない場合は、以下の部分を変更する。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Get_structure
C       ―――――――――――――――――――――――――――――――
C     ●   構造データを入力し、データをダンプファイルに出力する。
C     ―――――――――――――――――――――――――――――――
      subroutine Get_structure(Point,Member,Element,Parameter_C,
     *           Model_type,ierr,
     *           S_comp_model,E_Fiber_work)
       .
       .
c――――――――――――――――――――――――★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――――――――――――――――――――――――★新規任意型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.and.
     *    (S_comp_model(j-50).nm_type_element(k).ne.＊) then   ！追加モデルで特殊断面データ必要としない場合、＊そのエレメント番号
       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)                    ！新規任意型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
       •
       •
```

特殊断面データを必要とする場合は、上記の変更を行ってはならない。

また、特殊断面データを必要とする場合でも標準的な入力仕様であれば、そのデータを入力するサブルーチン Fiber_input()は、任意型の部材モデル 51-70 でデータ入力を行うため、変更する必要がない。

次に、出力仕様であるが、関連する出力項目として、部材応力とファイバー応力の 2 種類がある。関連する 2 つのサブルーチン Out_stress()と Out_Fiber()は先の節で示した。ここで見られるように他のサブルーチンでデータを適切に構造体にセットすれば変更する必要はない。

最後に、必要となるサブルーチンの組み込みであるが、そのサブルーチンは、

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

であり、これらのサブルーチンで任意型の部材モデルの中にエレメントモデルを追加することであり、必要となる変更部分を以下に示す。最初に、線形剛性を求めるサブルーチンを以下に示す。任意型の部材モデルの線形剛性を求めるサブルーチンは、かなり深い階層に存在する。

```
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
c      call MS_Model_GIx(ak,alength(im),Member,Element,
c     *      E_modelx,E_model_fiber,
c     *      M_modelx,M_model_fiber,
c     *      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―――――――――――――――――――――――――――――――――★接合部ばねモデル要素
      call Wood_Joint_GK(ak,E_modelx,E_model_fiber,M_modelx,
     *                   M_model_fiber)
   16 continue
c―――――――――――――――――――――――――――――――――★新規エレメントモデル
      call New_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
   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
```

上記のサブルーチンに示されるように、新規のエレメントモデル番号にしたがって、該当する位置に線形剛性を計算するサブルーチンを組み込むことになる。

次に、非線形剛性を求めるサブルーチンと、応力を求めるサブルーチンを以下に示す。非線形剛性を計算するサブルーチンでは、先の線形剛性を計算するサブルーチンと同様に、新規エレメントモデルに対応するモデル番号の位置に、新しく作成したサブルーチンをコールするコードを挿入する。また、応力を計算するサブルーチンでは、コードから理解できるように、接線剛性を用いているため、変更する必要なない。

```
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"
      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 / 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
c      call MS_Model_Gx(ak,alength(im),Member,Element,
c     *      E_modelx,E_model_fiber,
c     *      M_modelx,M_model_fiber)
      goto 100
   14 continue
c――――――――――――――――――――――――――――――――――★アナロジー要素
       call Analogy_Model_Gx(im,ak,Member,Element,
      *      E_modelx,E_model_fiber,
      *      M_modelx,M_model_fiber)
      goto 100
   15 continue
c――――――――――――――――――――――――――――――――――★接合部ばねモデル要素
      call Wood_Joint_GK(ak,E_modelx,E_model_fiber,M_modelx,
     *                                   M_model_fiber)
      goto 100
   16 continue
c――――――――――――――――――――――――――――――――――★新規エレメントモデル
      call New_Model_GI(ak,alength(im),Member,Element,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work)

      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
```

```
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
     *                        ,Point)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
         record / point_s     / Point
         dimension Point(*)
      dimension ak(12,12) ,vv(12),r1(3,3),r2(3,3),st(12),ss(12)
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
c――――――――――――――――――――――――★剛床座標変換
       do k=1,2
        i1=Member.nm_point(k)
        if(Point(i1).n_group_gouyuka.ne.0) then
        k1=6*(k-1)+1
        call Set_gtrans_f(1,st(k1),Point(i1).coord_local(1),
     *                                Point(i1).coord_local(2))
        endif
         enddo
c――――――――――――――――――――――★全体座標から部材座標
      call RotateL_v(1,st,r1,r2,ss)
      do i=1,6
      Member.stress(i)=-ss(i)+Member.stress(i)
      enddo
      do i=7,12
      Member.stress(i)=ss(i)+Member.stress(i)
      enddo
      return
      end
```

最後に、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が、これも、任意型静的縮合モデルでは、深い階層に存在する。このサブルーチン Cal_check_stiff_Mx()を以下に示す。ここでも、先のサブルーチンと同様に、新規エレメントモデルと同様に、エレメントモデル番号に該当する位置に応力チェックサブルーチンをコールするコードを挿入する。

```
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(:)
      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      write(76,'(a,10i4)') 'check : iet,ix_model',nmmx,nmx,n_div,
c     * iet,ix_model,nmmx,nmx,n_div
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      write(76,'(a,4i4,f12.3)') 'check :',i,imm,immm,n_type(i),eaa
    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))
c      write(76,'(a,i5,2e12.4)') ' vv ww',i,M_Fiber_Work(nmmx+i).an_vv,
c     * M_Fiber_Work(nmmx+i).an_ww
       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
c      write(76,'(a,10i4)') ' n_type ',i,imm,immm,n_type(i)
      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
c      call MS_checkx(N_analysis,
c     *      mem_x,alength(i),Member,Element,
c     *      E_modelx(imm),E_model_fiber,M_modelx(immm),M_model_fiber,
c     *      Bilinear_work,Trilinear_work,Concrete_work,vvx,
c     *      M_Fiber_Work(nmmx+i).an_vv, M_Fiber_Work(nmmx+i).an_ww,
c     *      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-----------------------------★接合部ばねモデル要素
      call Wood_Joint_checkx(Control,N_analysis,Member,
     *                      Element,E_modelx(imm),E_model_fiber,
     *                      M_modelx(immm),M_model_fiber,Bilinear_work,
     *                     Trilinear_Work,vvx)
      goto 100
   16 continue
c-----------------------------★新規エレメントモデル
      call New_Model_checkx(ak,alength(im),Member,Element,
     *      E_modelx,E_model_fiber,
     *      M_modelx,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work)
```

```
      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))
c      write(76,'(a,i5,2e12.4)') ' vv ww',i,M_Fiber_Work(nmmx+i).an_vv,
c     * M_Fiber_Work(nmmx+i).an_ww
      enddo
c――――――――――――――――――――――――――――――――★動的記憶領域の解放
      DEALLOCATE (
     *   c ,ab ,irest_point,n_type,alength,
     *   bav,akk,f1,f2    )
      return
      end
```

```
     *     vv,vpp )
      endif
      do j=1,3
      f(j)=-Member(i).stress(j)
      f(j+6)=-f(j)
      enddo
      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)=ff(j)
      enddo
      goto 100
   13 continue
c-------------------------------★Model_No.3 3 次元軸力弾塑性モデル
      .
      .
  100 continue
      end do
      return
      end
```

部材モデル番号が 2 であると、せん断型弾塑性モデルのサブルーチンに制御が移る。このとき、解析種別が 9 以上で弾塑性解析を行う場合、このサブルーチンが実行される。引数として増分変位と増分前の変位を、釣合座標系から部材座標系に変換し、受け渡す。

次に、サブルーチン Cal_check_stiff_M2()を示す。

```
C     ------------------------------------------
C     ●   SUBROUTINE /Cal_check_stiff_M2
C       ----------------------------------------
C     ●   Model_No.2 3 次元せん断弾塑性モデル
C     ------------------------------------------
      subroutine Cal_check_stiff_M2(Member,Element,RO_work,vv,vpp)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      include "New_submain.h "
      record / member_s2    / Member
      record / element_s2   / Element
      record / RO_work_s    /  RO_work
      dimension vv(12),vpp(12),RO_work(*)
c-----------------------------------------------------------
      No_rireki=Element.nm_type
      if(No_rireki/10.eq.0) then
      goto(5,10,20,30,40,50,60),No_rireki+1
    5 continue
c-------------------------------★規定モデル：武田モデル
      call Takeda_TriLiner(Member,Element,vv,vpp)
      goto 999
   10 continue
c-------------------------------★トリリニア：Nomal
      call Mesing_TriLiner(Member,Element,vv,vpp)
```

```
        goto 999
   20 continue
c――――――――――――――――――――――――――――――――――★トリリニア：最大点指向型
      call DirecMax_TriLiner(Member,Element,vv,vpp)
      goto 999
   30 continue
c――――――――――――――――――――――――――――――――――★トリリニア：武田モデル
      call Takeda_TriLiner(Member,Element,vv,vpp)
      goto 999
   40 continue
c――――――――――――――――――――――――――――――――――★バイリニア：Nomal
      if(Member.istat_v.eq.0.and.Member.istat_w.eq.0) then
         Element.AK_2=Element.AK_1
         Element.Q_2 =Element.Q_1
      endif
      call Mesing_TriLiner(Member,Element,vv,vpp)
      goto 999
   50 continue
c――――――――――――――――――――――――――――――――――★バイリニア：S 字スリップ型
      call S_slip_bilinear_s(Member,Element,vv,vpp)
      goto 999
   60 continue
c――――――――――――――――――――――――――――――――――★木質パネル：S 字スリップ型
      call S_slip_woodpanel(Member,Element,vv,vpp)
      goto 999
c――――――――――――――――――――――――――――――――――★
      elseif(No_rireki/10.eq.1) then
      goto(101,102),No_rireki - 10
  101 continue
c――――――――――――――――――――――――――――――――――★修正バイリニアモデル
      mro=Element.n_section(1)
      call Modify_Bi_Liner1(Member,Element,R0_work(mro),vv,vpp)
      goto 999
  102 continue
c――――――――――――――――――――――――――――――――――★修正 R0 モデル
      mro=Element.n_section(1)
      call Modify_R01(Member,Element,R0_work(mro),vv,vpp)
      goto 999
      elseif(No_rireki/100.eq.1) then
      goto(201,202),No_rireki - 100
  201 continue
c――――――――――――――――――――――――――――――――――★個人用新規モデル
      call New_model_rireki(Member,Element,vv,vpp)
      goto 999
  202 continue
c――――――――――――――――――――――――――――――――――★個人用新規モデル
      goto 999
      endif
  999 continue
      return
      end
```

このサブルーチンの中に履歴モデルに関する第２層の階層構造が見ら

れる。この中に履歴特性番号 101 として、太文字で示すコードが追加され、新規の履歴モデルをチェックするサブルーチンが加えられている。ここでは、サブルーチン名を New_model_rireki()としている。これで、既存サブルーチンに新規モデルが組み込まれたことになる。引数として受け渡すデータは、構造体の Member と Element、部材座標系の部材両端の節点増分変位と増分前の変位である。特殊な構造体を設計した場合は、ここで引数として受け渡す必要がある。この構造体は、当然、上位のサブルーチンから受け渡される必要があり、また、その利用法は修正 RO モデルにおける RO_work を参考にされたい。その場合、構造体を設定したヘッダーファイル New_submain.h をインクルードしなければならない。

この新規サブルーチンの役割は、実際にせん断型モデルの履歴を追うことであり、その中で、接線剛性行列を他のサブルーチンが作成するために、初期設定と構造体 Member 中の次の値を設定しなければならない。

１．Member.istat が -1 のとき、初期設定行う。
２．接線剛性 Member.AKw_tan を設定する。
３．せん断ばねの応力 Member.stress を求める。

以上のように、新規せん断型モデルの組み込みは、データ設定が標準入力仕様で間に合えば非常に簡単である。基本的には、上記の応力の弾塑性チェック用サブルーチンを用意すれば良い。このサブルーチンを所定の位置に設定するのみで、新しいせん断型履歴モデルが組み込むことが可能となる。

1.4 ファイバー履歴の組み込み方法

1.4.1 履歴の組み込み

本節では、新規のファイバー履歴を SPACE に組み込む方法について解説する。基本的には、せん断モデルの履歴組み込み方法と同じであるが、多少異なるので、その違いを中心に説明する。現在、SPACE に登録されている履歴モデルは、以下のようであり、履歴番号 1-13 までが使用されている。個人用としては、せん断型と同じく 101 を用いることにする。ここでは、新規のモデルを New_model_fiber とする。

１．対称バイリニア型：BiLinear()
２．対称トリリニア型：TriLinear()
３．直線コンクリート履歴モデル：Concrete()
４．曲線コンクリート履歴モデル：Concrete_e()
５．バイリニア型（移動＋等方硬化用）：BiLinear_h()
６．対称トリリニア型（移動＋等方硬化用）：TriLinear_h()
７．非対称バイリニア型：TriLinear_AS()
８．非対称トリリニア型：BiLinear_AS()
９．降伏棚を有する対称バイリニア型（移動＋等方硬化用）
11．降伏棚を有する対称トリリニア型（移動＋等方硬化用）
12．鉄筋用履歴モデル
13．木質構造材用履歴モデル

さらに、新規部材モデルで使用可能な接合部履歴モデルとして以下の履歴モデルが用意されている。

１．Ｓ字型バイリニアスリップモデル
２．Ｓ字型トリリニアスリップモデル
３．バイリニア型スリップモデル
４．トリリニア型スリップモデル
５．ボックス型スリップモデル

1.4.2 モデルの入力仕様

最初に、このファイバーモデルの入力仕様について考えてみよう。ファイバーの履歴モデルは全て特別仕様でデータ入力を行っており、データを保存する構造体が設計されている。

まず、データ入力を行うサブルーチン Fiber_input()の中で、要素データを入力する部分のコードを取り出す。新規履歴モデルに関するコードが太文字で追加されている。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Fiber_input
C     ―――――――――――――――――――――――――――――――
C     ●   ファイバー要素の入力(ok)
C     ―――――――――――――――――――――――――――――――
      subroutine Fiber_input(  )
      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
      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
       kk1 = kk1 + 1
        do k=1,2
         if(Element(i1).n_section(k).eq.n_m) then
          Element(i1).nm_section(k)     = nmm
          if(k.eq.1) then
            E_model11(kk1).n_section_1  = nmm
            E_model11(kk1).nm_section_1 = ii + 1
          endif
          if(k.eq.2) then
            E_model11(kk1).n_section_2  =nmm
            E_model11(kk1).nm_section_2 = ii+1
          endif
          endif
        enddo
      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
      enddo
c―――――――――――――――――――――★
      do j=1,nmm
      ii = ii + 1
      read(5,*,err=999) n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az                    ! 1
      if(nm_type.le.10) then
      goto(901,902,903,904,905,906,907,908,909,910),nm_type                      ! 2
 901  continue
      goto 900
 902  continue
      read(5,*,err=999) E_3,Q_2                          ! 対称トリリニア型 （スチール用）
      goto 900
 903  continue
      read(5,*,err=999) AK_3,AK_4,Q_2,Q_3,Q_4            ! コンクリート型
      goto 900
 904  continue
      read(5,*,err=999) AK_4,Q_3,STR_3,STR_7             ! 直線曲線コンクリート型
      goto 900
 905  continue
c     read(5,*,err=999) beta                             ! 等方硬化＋移動硬化バイリニア型
      goto 900
 906  continue
      read(5,*,err=999) E_3,Q_2                          ! 等方硬化＋移動硬化トリリニア型
      goto 900
 907  continue
      read(5,*,err=999) E_3,Ec_1,Ec_2,Ec_3,Qc_1          ! 非対称バイリニア型
      goto 900
 908  continue
      read(5,*,err=999) E_3,E_4,Q_2,Ec_1,Ec_2,Ec_3,Ec_4,Qc_1,Qc_2   ! 非対称トリリニア型
      goto 900
 909  continue
      goto 900
 910  continue
      goto 900
      elseif(nm_type.le.20) then
      goto(911,912,913,914,915,916,917,918,919,920),nm_type-10
 911  continue
      goto 900
 912  continue
      goto 900
 913  continue
      goto 900
 914  continue
      goto 900
 915  continue
      goto 900
 916  continue
     read(5,*,err=999) E_3,Q_2,Ec_1,Ec_2,Ec_3,Qc_1,Qc_2          ! 木質構造材用履歴モデル
      goto 900
```

```
 917  continue
      goto 900
 918  continue
      goto 900
 919  continue
      goto 900
 920  continue
      goto 900
      else
      goto(921,922,923,924,925,926,927,928,929,930),nm_type-100                    ! 3
 921  continue
      read(5,*,err=999) E_3,Q_2                              ! 個人用新規ファイバー履歴
      goto 900
 922  continue
      read(5,*,err=999) E_3,Q_2                              ! 個人用新規ファイバー履歴
      goto 900
 923  continue
      goto 900
 924  continue
      goto 900
 925  continue
      goto 900
 926  continue
      goto 900
 927  continue
      goto 900
 928  continue
      goto 900
 929  continue
      goto 900
 930  continue
      goto 900
      endif
 900  continue
      enddo
      enddo
c―――――――――――――――――――――――★要素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
      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
         k = 1
          M_model11(immm).n_section_1   = Element(i1).nm_section(k)
          M_model11(immm).nm_section_1  = jj + 1
          jj = jj + Element(i1).nm_section(k)
         k  = 2
          M_model11(immm).n_section_2   = Element(i1).nm_section(k)
          M_model11(immm).nm_section_2  = jj + 1
```

```
          jj = jj + Element(i1).nm_section(k)
         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
      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     ! 4
c――――――――――――――――――――――――★部材断面ファイバー数セット
      ii = ii + 1
      E_model_fiber(ii).nm_type = nm_type              !  履歴特性番号
      E_model_fiber(ii).E_1     = E_1                  !  ファイバーの第一剛性 E1
      E_model_fiber(ii).E_2     = E_2                  !  ファイバーの第二剛性 E2
      E_model_fiber(ii).Q_1     = Q_1                  !  ファイバーの第一折れ点
      E_model_fiber(ii).G       = G                    !  ファイバー断面積 G
```

```
      E_model_fiber(ii).A       = A                   !  ファイバー断面積
      E_model_fiber(ii).Ay      = Ay                  !  ファイバーｙ軸せん断用断面積
      E_model_fiber(ii).Az      = Az                  !  ファイバーｚ軸せん断用断面積
      E_model_fiber(ii).ry      = ry                  !  中立軸から断面中心までのｙ方向距離
      E_model_fiber(ii).rz      = rz                  !  中立軸から断面中心までのｚ方向距離
      if(nm_type.le.10) then
      goto(801,802,803,804,805,806,807,808,809,810),nm_type                    ! 5
 801  continue
c――――――――――――――――――――――――★バイリニア型
      write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
      E_model_fiber(ii).E_3     = 0.                  !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_2     = 0.                  !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = E_1                 !  ファイバーの圧縮側第一剛性 E1
      E_model_fiber(ii).Ec_2    = 0.                  !  ファイバーの圧縮側第二剛性 E2
      E_model_fiber(ii).Ec_3    = 0.                  !  ファイバーの圧縮側第三剛性 E3
      E_model_fiber(ii).Qc_1    = 0.                  !  ファイバーの圧縮側第一折れ点
      E_model_fiber(ii).Qc_2    = 0.                  !  ファイバーの圧縮側第二折れ点
      goto 800
 802  continue
c――――――――――――――――――――――――★対称トリリニア型
      read(5,*,err=999) E_3,Q_2          ! 対称トリリニア型 等方 硬化＋移動硬化トリリニア型
      write(76,'(2i4,18e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az,
     *     E_3,Q_2,beta,beta_2
      E_model_fiber(ii).E_3     = E_3                 !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_2     = Q_2                 !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = 0.                  !  ファイバーの圧縮側第一剛性 E1
      E_model_fiber(ii).Ec_2    = 0.                  !  ファイバーの圧縮側第二剛性 E2
      E_model_fiber(ii).Ec_3    = 0.                  !  ファイバーの圧縮側第三剛性 E3
      E_model_fiber(ii).Qc_1    = 0.                  !  ファイバーの圧縮側第一折れ点
      E_model_fiber(ii).Qc_2    = 0.                  !  ファイバーの圧縮側第二折れ点
      E_model_fiber(ii).Beta    = 0.                  !  移動硬化用パラメータ
      E_model_fiber(ii).Beta_2  = 0.                  !  移動硬化用パラメータその２
      goto 800
 803  continue
c――――――――――――――――――――――――★直線コンクリート型
      read(5,*,err=999) AK_3,AK_4,Q_2,Q_3,Q_4               ! 直線コンクリート型
      write(76,'(2i4,18e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az,
     *     AK_3,AK_4,Q_2,Q_3,Q_4
      E_model_fiber(ii).E_3     = AK_3                !  圧縮第三勾配
      E_model_fiber(ii).Q_2     = Q_2                 !  圧縮側第一折れ点の応力
      E_model_fiber(ii).Ec_1    = Q_3                 !  圧縮強度
      E_model_fiber(ii).Ec_2    = Q_4                 !  圧縮流れ点
      E_model_fiber(ii).Ec_3    = AK_4                !  引張第二勾配
      E_model_fiber(ii).Qc_1    = 0.                  !  ダミー
      E_model_fiber(ii).Qc_2    = 0.                  !  ダミー
      goto 800
 804  continue
c――――――――――――――――――――――――★曲線コンクリート型
      read(5,*,err=999) AK_4,Q_3,STR_3,STR_7                ! 曲線コンクリート型
      write(76,'(2i4,18e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az,
     *     AK_4,Q_3,STR_3,STR_7
      E_model_fiber(ii).E_3     = AK_4                !  引張第二勾配
      E_model_fiber(ii).Q_2     = Q_3                 !  圧縮強度
      E_model_fiber(ii).Ec_1    = STR_3               !  最大圧縮応力点におけるひずみ量
```

```
      E_model_fiber(ii).Ec_2    = STR_7                !  弾性限界ひずみ量
      E_model_fiber(ii).Ec_3    = 0.                   !  ダミー
      E_model_fiber(ii).Qc_1    = 0.                   !  ダミー
      E_model_fiber(ii).Qc_2    = 0.                   !  ダミー
      goto 800
 805  continue
c―――――――――――――――――――――――★等方硬化＋移動硬化バイリニア型
      read(5,*,err=999) beta
       write(76,'(2i4,10e12.4)') n,nm_type,A,ry,rz,E_1,E_2,
     *                          Q_1,G,Ay,Az            !等方硬化＋移動硬化バイリニア型
      E_model_fiber(ii).E_3     = 0.                   !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_2     = 0.                   !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = E_1                  !  ファイバーの圧縮側第一剛性 E1
      E_model_fiber(ii).Ec_2    = 0.                   !  ファイバーの圧縮側第二剛性 E2
      E_model_fiber(ii).Ec_3    = 0.                   !  ファイバーの圧縮側第三剛性 E3
      E_model_fiber(ii).Qc_1    = 0.                   !  ファイバーの圧縮側第一折れ点
      E_model_fiber(ii).Qc_2    = 0.                   !  ファイバーの圧縮側第二折れ点
      E_model_fiber(ii).Beta    = beta                 !  移動硬化用パラメータ
      goto 800
 806  continue
c―――――――――――――――――――――――★等方硬化＋移動硬化トリリニア型
      read(5,*,err=999) E_3,Q_2,beta,beta_2      !
      write(76,'(2i4,18e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az,
     *     E_3,Q_2
      E_model_fiber(ii).E_3     = E_3                  !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_2     = Q_2                  !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = 0                    !  ファイバーの圧縮側第一剛性 E1
      E_model_fiber(ii).Ec_2    = 0                    !  ファイバーの圧縮側第二剛性 E2
      E_model_fiber(ii).Ec_3    = 0                    !  ファイバーの圧縮側第三剛性 E3
      E_model_fiber(ii).Qc_1    = 0                    !  ファイバーの圧縮側第一折れ点
      E_model_fiber(ii).Qc_2    = 0                    !  ファイバーの圧縮側第二折れ点
c      E_model_fiber(ii).Beta    = beta                 !  移動硬化用パラメータ
c      E_model_fiber(ii).Beta_2  = beta_2               !  移動硬化用パラメータ
      goto 800
 807  continue
c―――――――――――――――――――――――★非対称バイリニア型
      read(5,*,err=999) Ec_1,Ec_2,Qc_1,beta
       write(76,'(2i4,15e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az,
     *              Ec_1,Ec_2,Qc_1                    !等方硬化＋移動硬化バイリニア型
      E_model_fiber(ii).E_3     = 0.                   !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_2     = 0.                   !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = Ec_1                 !  ファイバーの圧縮側第一剛性 E1
      E_model_fiber(ii).Ec_2    = Ec_2                 !  ファイバーの圧縮側第二剛性 E2
      E_model_fiber(ii).Ec_3    = 0.                   !  ファイバーの圧縮側第三剛性 E3
      E_model_fiber(ii).Qc_1    = Qc_1                 !  ファイバーの圧縮側第一折れ点
      E_model_fiber(ii).Qc_2    = 0.                   !  ファイバーの圧縮側第二折れ点
c      E_model_fiber(ii).Beta    = beta                 !  移動硬化用パラメータ
      goto 800
 808  continue
 c―――――――――――――――――――――――★非対称トリリニア型
      read(5,*,err=999) E_3,Q_2,Ec_1,Ec_2,Ec_3,Qc_1,Qc_2
      E_model_fiber(ii).E_3     = E_3                  !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_2     = Q_2                  !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = Ec_1                 !  ファイバーの圧縮側第一剛性 E1
```

```
      E_model_fiber(ii).Ec_2    = Ec_2                  !  ファイバーの圧縮側第二剛性 E2
      E_model_fiber(ii).Ec_3    = Ec_3                  !  ファイバーの圧縮側第三剛性 E3
      E_model_fiber(ii).Qc_1    = Qc_1                  !  ファイバーの圧縮側第一折れ点
      E_model_fiber(ii).Qc_2    = Qc_2                  !  ファイバーの圧縮側第二折れ点
c      E_model_fiber(ii).Beta    = beta                  !  移動硬化用パラメータ
c      E_model_fiber(ii).Beta_2  = beta_2                !  移動硬化用パラメータ
      goto 800
809   continue
c――――――――――――――――――――――――★降伏棚を有する等方硬化＋移動硬化バイリニア型
      goto 800
 810  continue
c――――――――――――――――――――――――★降伏棚を有する等方硬化＋移動硬化トリリニア型
      elseif(nm_type.le.20) then
      goto(811,812,813,814,815,816,817,818,819,820),nm_type-10
 811  continue
      write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
      E_model_fiber(ii).E_1     = A                     !  ファイバーの第一剛性 E1
      E_model_fiber(ii).E_2     = ry                    !  ファイバーの第二剛性 E2
      E_model_fiber(ii).E_3     = rz                    !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_1     = E_1                   !  ファイバーの第一折れ点
      E_model_fiber(ii).Q_2     = E_2                   !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = 0
      E_model_fiber(ii).Ec_2    = 0
      E_model_fiber(ii).Ec_3    = 0
      E_model_fiber(ii).Qc_1    = 0
      E_model_fiber(ii).Qc_2    = 0
      if(ry.eq.0.)  E_model_fiber(ii).E_2=A*0.000001
      goto 800
 812  continue
      write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
      E_model_fiber(ii).E_1     = A                     !  ファイバーの第一剛性 E1
      E_model_fiber(ii).E_2     = ry                    !  ファイバーの第二剛性 E2
      E_model_fiber(ii).E_3     = rz                    !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_1     = E_1                   !  ファイバーの第一折れ点
      E_model_fiber(ii).Q_2     = E_2                   !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = 0
      E_model_fiber(ii).Ec_2    = 0
      E_model_fiber(ii).Ec_3    = 0
      E_model_fiber(ii).Qc_1    = 0
      E_model_fiber(ii).Qc_2    = 0
      if(ry.eq.0.)  E_model_fiber(ii).E_2=A*0.000001
      goto 800
 813  continue
      write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
      E_model_fiber(ii).E_1     = A                     !  ファイバーの第一剛性 E1
      E_model_fiber(ii).E_2     = ry                    !  ファイバーの第二剛性 E2
      E_model_fiber(ii).E_3     = rz                    !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_1     = E_1                   !  ファイバーの第一折れ点
      E_model_fiber(ii).Q_2     = E_2                   !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = 0
      E_model_fiber(ii).Ec_2    = 0
      E_model_fiber(ii).Ec_3    = 0
      E_model_fiber(ii).Qc_1    = 0
      E_model_fiber(ii).Qc_2    = 0
```

```
       if(ry.eq.0.)  E_model_fiber(ii).E_2=A*0.000001
      goto 800
 814  continue
      goto 800
 815  continue
c――――――――――――――――――――――――★鉄筋用履歴モデル
c        read(5,*,err=999) beta                                  ! 鉄筋用履歴モデル
      write(76,'(2i4,10e12.4)') n,nm_type,A,ry,rz,E_1,E_2,
     *                          Q_1,G,Ay,Az !,beta     !等方硬化＋移動硬化バイリニア型
      E_model_fiber(ii).E_3     = 0.                    !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_2     = 0.                    !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = E_1                   !  ファイバーの圧縮側第一剛性 E1
      E_model_fiber(ii).Ec_2    = 0.                    !  ファイバーの圧縮側第二剛性 E2
      E_model_fiber(ii).Ec_3    = 0.                    !  ファイバーの圧縮側第三剛性 E3
      E_model_fiber(ii).Qc_1    = 0.                    !  ファイバーの圧縮側第一折れ点
      E_model_fiber(ii).Qc_2    = 0.                    !  ファイバーの圧縮側第二折れ点
c      E_model_fiber(ii).Beta    = beta                  !  移動硬化用パラメータ
      goto 800
 816  continue
c――――――――――――――――――――――――★木質構造材用履歴モデル
      read(5,*,err=999) E_3,Q_2,Ec_1,Ec_2,Ec_3,Qc_1,Qc_2      !,beta,beta_2
      write(76,'(2i4,16e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az,
     *              E_3,Q_2,Ec_1,Ec_2,Ec_3,Qc_1,Qc_2
      E_model_fiber(ii).E_3     = E_3                    !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Q_2     = Q_2                    !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_1    = Ec_1                   !  ファイバーの圧縮側第一剛性 E1
      E_model_fiber(ii).Ec_2    = Ec_2                   !  ファイバーの圧縮側第二剛性 E2
      E_model_fiber(ii).Ec_3    = Ec_3                   !  ファイバーの圧縮側第三剛性 E3
      E_model_fiber(ii).Qc_1    = Qc_1                   !  ファイバーの圧縮側第一折れ点
      E_model_fiber(ii).Qc_2    = Qc_2                   !  ファイバーの圧縮側第二折れ点
c      E_model_fiber(ii).Beta    = 0
c      E_model_fiber(ii).Beta_2  = 0
      goto 800
 817  continue
      goto 800
 818  continue
      goto 800
 819  continue
      goto 800
 820  continue
      goto 800
      elseif(nm_type.le.30) then
      goto(831,832,833,834,835,836,837,838,839,840),nm_type-20
c――――――――――――――――――――――――★接合部ばねモデル
c――――――――――――――――――――――――★Ｓ字型バイリニアスリップモデル
 831  continue
      write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
      E_model_fiber(ii).G       = A                     !  ばねモデルの番号
      E_model_fiber(ii).E_1     = ry                    !  ファイバーの第一剛性 E1
      E_model_fiber(ii).E_2     = rz                    !  ファイバーの第二剛性 E2
      E_model_fiber(ii).E_3     = E_1                   !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Ec_1    = E_2                   !  ファイバーの除荷剛性
      E_model_fiber(ii).Q_1     = Q_1                   !  ファイバーの第一折れ点
c      E_model_fiber(ii).Q_2     = 0                     !  ファイバーの第二折れ点
```

```
c      E_model_fiber(ii).Qc_1    = 0                                    !  ファイバーの第三折れ点
c      E_model_fiber(ii).Ec_3    = 0
c      E_model_fiber(ii).Qc_1    = 0
c      E_model_fiber(ii).Qc_2    = 0
       goto 800
c―――――――――――――――――――――――★Ｓ字型トリリニアスリップモデル
 832   continue
       write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
       E_model_fiber(ii).G       = A                    !  ばねモデルの番号
       E_model_fiber(ii).E_1     = ry                   !  ファイバーの第一剛性 E1
       E_model_fiber(ii).E_2     = rz                   !  ファイバーの第二剛性 E2
       E_model_fiber(ii).E_3     = E_1                  !  ファイバーの第三剛性 E3
       E_model_fiber(ii).Ec_1    = E_2                  !  ファイバーの第四剛性 E4
       E_model_fiber(ii).Ec_2    = Q_1                  !  ファイバーの除荷剛性 E5
       E_model_fiber(ii).Q_1     = G                    !  ファイバーの第一折れ点
       E_model_fiber(ii).Q_2     = Ay                   !  ファイバーの第二折れ点
c      E_model_fiber(ii).Ec_3    = 0
c      E_model_fiber(ii).Qc_1    = 0
c      E_model_fiber(ii).Qc_2    = 0
         goto 800
c―――――――――――――――――――――――★バイリニア型スリップモデル
 833   continue
       write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
       E_model_fiber(ii).G       = A                    !  ばねモデルの番号
       E_model_fiber(ii).E_1     = ry                   !  ファイバーの第一剛性 E1
       E_model_fiber(ii).E_2     = rz                   !  ファイバーの第二剛性 E2
       E_model_fiber(ii).Q_1     = E_1                  !  ファイバーの除荷剛性 E3
c      E_model_fiber(ii).Q_1     = E_2                   !  ファイバーの第一折れ点
c      E_model_fiber(ii).Qc_2    = 0
c      E_model_fiber(ii).Ec_3    = 0
c      E_model_fiber(ii).Qc_1    = 0
c      E_model_fiber(ii).Qc_2    = 0
         goto 800
c―――――――――――――――――――――――★トリリニア型スリップモデル
 834   continue
       write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
       E_model_fiber(ii).G       = A                    !  ばねモデルの番号
       E_model_fiber(ii).E_1     = ry                   !  ファイバーの第一剛性 E1
       E_model_fiber(ii).E_2     = rz                   !  ファイバーの第二剛性 E2
       E_model_fiber(ii).E_3     = E_1                  !  ファイバーの第三剛性 E3
       E_model_fiber(ii).Q_1    = E_2                   !  ファイバーの除荷剛性 E4
       E_model_fiber(ii).Q_2     = Q_1                  !  ファイバーの第一折れ点
c      E_model_fiber(ii).Q_2     = G                     !  ファイバーの第二折れ点
c      E_model_fiber(ii).Qc_2    = 0
c      E_model_fiber(ii).Ec_3    = 0
c      E_model_fiber(ii).Qc_1    = 0
c      E_model_fiber(ii).Qc_2    = 0
       goto 800
c―――――――――――――――――――――――★ボックス型スリップモデル
 835    continue
       write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
       E_model_fiber(ii).G       = A                    !  ばねモデルの番号
       E_model_fiber(ii).E_1     = ry                   !  ファイバーの第一剛性 E1
       E_model_fiber(ii).E_2     = rz                   !  ファイバーの第二剛性 E2
```

```
      E_model_fiber(ii).E_3     = E_1                  !  ファイバーの除荷剛性 E3
      E_model_fiber(ii).Ec_1    = E_2                  !  ファイバーの最高点指向勾配
      E_model_fiber(ii).Q_1     = Q_1                  !  ファイバーの第二折れ点
      E_model_fiber(ii).Q_2     = G                    !  ファイバーの第三折れ点
      E_model_fiber(ii).Qc_1    = Ay                   ! 中間折れ点
      E_model_fiber(ii).Qc_2   = Az                    !  スリップ剛性を求めるパラメーター
c      E_model_fiber(ii).Ec_3    = 0
c      E_model_fiber(ii).Qc_1    = 0
c      E_model_fiber(ii).Qc_2    = 0
      goto 800
 836   continue
      write(76,'(2i4,9e12.4)') n,nm_type,A,ry,rz,E_1,E_2,Q_1,G,Ay,Az
      E_model_fiber(ii).G       = A                    !  ばねモデルの番号
      E_model_fiber(ii).E_1     = ry                   !  ファイバーの第一剛性 E1
      E_model_fiber(ii).E_2     = rz                   !  ファイバーの第二剛性 E2
      E_model_fiber(ii).E_3     = E_1                  !  ファイバーの第三剛性 E3
      E_model_fiber(ii).Ec_1    = E_2                  !  ファイバーの第四剛性 E4
      E_model_fiber(ii).Q_1     = Q_1                  !  ファイバーの除荷剛性 E5
      E_model_fiber(ii).Q_2     = G                    !  ファイバーの第一折れ点
      E_model_fiber(ii).Qc_1    = Ay                   !  ファイバーの第二折れ点
      E_model_fiber(ii).Ec_3    = 0
      E_model_fiber(ii).Qc_2    = 0
         goto 800
 837     continue
         goto 800
 838     continue
         goto 800
 839     continue
         goto 800
 840     continue
         goto 800
         else
      goto(821,822,823,824,825,826,827,828,829,830),nm_type-100                    ! 6
 821  continue
      read(5,*,err=999) E_3,Q_2                          ! 個人用新規ファイバー履歴
      goto 800
 822  continue
      read(5,*,err=999) E_3,Q_2                          ! 個人用新規ファイバー履歴
      goto 800
 823  continue
      goto 800
 824  continue
      goto 800
 825  continue
      goto 800
 826  continue
      goto 800
 827  continue
      goto 800
 828  continue
      goto 800
 829  continue
      goto 800
 830  continue
```

```
      goto 800
      endif
 800  continue
      enddo
c―――――――――――――――――――――――――★ファイバーモデル剛性出力
      call Fiber_output(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
          ii  = E_model11(imm).n_section_1
          nmm = E_model11(imm).nm_section_1 - 1
          nnmm=M_model11(im).nm_section_1 - 1
          do j=1,ii
            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
             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
     *        .or.E_model_fiber(nmm).nm_type.eq.15.or.E_model_fiber(nmm).nm_type.eq.16
     *        .or.E_model_fiber(nmm).nm_type.eq.101) then  ! 新規履歴モデルのための追加コード
             n_m_trilinear = n_m_trilinear + 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
             M_model_fiber(nnmm).n_type = n_m_concrete
          endif
         enddo
         ii  = E_model11(imm).n_section_2
         nmm = E_model11(imm).nm_section_2 - 1
         nnmm= M_model11(im).nm_section_2 - 1
         do j=1,ii
          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
             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
     *          .or._model_fiber(nmm).nm_type.eq.101) then  ! 新規履歴モデルのための追加コード
               n_m_trilinear = n_m_trilinear + 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
               M_model_fiber(nnmm).n_type = n_m_concrete
             endif
           enddo
         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――――――――――――――――――――★
      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
```

1. サブルーチン Fiber_input()は､対応するファイルを 2 度読みする。従って、2 箇所にデータを入力するコードが存在するので、同時に変更しなければならない。また、このファイバーデータの仕様を変更する場合は、静的解析と静的・動的プレゼンターの入力用プログラムも変更しなければならない。まず、ここでは、ファイバーモデルの標準入力の第 1 レコードを入力する。変数 nm_type は、ファイバーの履歴特性番号を表す。ファイバーの履歴は階層構造を構成している。
2. ファイバーの履歴特性番号毎に入力仕様が異なるため、階層構造に従って処理を分類し、入力用コードを別個に記述する。
3. **個人用新規ファイバー履歴の第 2 レコードを設計し、履歴特性番号 101 によりこの位置に入力用コードを追加する。**
4. 2 度目のファイル読み込みを行う部分であり、標準第 1 レコードを読み込む。
5. ファイバーの履歴特性番号毎に入力仕様が異なるため、階層構造に従って処理を分類し、入力用コードを別個に記述する。
6. **個人用新規ファイバー履歴の第 2 レコードを設計し、履歴特性番号 101 によりこの位置に入力用コードを記述する。つまり、新規ファイバー用履歴特性の入力データを設定する場合、標準仕様の第 1 レコードと残りのデータを第 2 レコードとして、ここと 3.の部分に入力コードを追加する。**

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かをチェックする。もし使用している場合は、要素構造体にファイバー断面番号をセットする。以下の処理では､部材モデル 12､13 などファイバー断面に関連するモデルは、全てここでファイバー断面とリンクを取らなければならない。詳細は、前節の部材モデル新規設計の項で説明した。

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. ここでは、断面内で使用しているファイバーの履歴を解析するため

に必要となるワーク領域構造体の数を数えている。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、その種類に対応して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。

新規の履歴モデルが追加される場合で、上記の３つのワーク領域を使用する場合は、コメントに示したコードを追加することになる。なお、追加する履歴特性の番号は 101 であり、使用するワーク領域はトリリニア型構造体とした。新規の履歴モデルが新しいワーク領域構造体を必要とする場合は、その構造体を新たに設計し、さらに、その数をここで数えるために、そのコードを追加する。無論、他の部材モデルでこの新規履歴モデルを組み込む場合は、該当する部分に同様の追加コードを加える必要がある。

x4. 最後に、各種の部材モデルで必要となる３つのワーク領域構造体の数が、構造体成分 model_type.n_m_bilinear などにセットされる。当然、新しいワーク領域構造体を設定した場合は、ここでその個数をセットする必要があり、そのコードを追加する。

1.4.3 モデルの出力仕様

本節では、この新しい履歴モデルに関する出力仕様について説明する。出力データは他のモジュールで使用されており、変更する場合は、他のシステム、例えば、動的プレゼンターなどをチェックし、整合性を取る必要がある。

まずは、通常の部材モデルにおける標準出力仕様を見てみよう。解析結果を出力するサブルーチン Out_stress()は、submain_dynamic_a()の中でコールされている。ここでは、両端ファイバーモデルに関連する部分のみ示す。

```
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"
      record / Member_s         / Member
      record / Element_s        / Element
      record / Out_section_s    / Out_section
      record / E_model6_real_s  / E_model6_real
       •
       •
      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
       •
       •
      else
  c―――――――――――――――――――――――――――――――★静的縮合モデル（ファイバーモデルなど）
      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)
       endif
       enddo
       return
       end
```

このサブルーチンの該当する部分を見ると、部材断面の応力が構造体成分 Member().stress()にセットされていれば、ファイバーデータに全く関係しない。したがって、ここでは、新規ファイバー履歴モデルを追加したとしても変更することはない。

次に、断面のファイバー応力とひずみの出力に関連する２つのサブルーチンを検討しよう。まず出力部材の断面をチェックし、断面ファイバーデータのヘッダー部分を出力するサブルーチン out_section_check() を見てみよう。

```
C     ―――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /out_section_check
C     ―――――――――――――――――――――――――――――――
C     ●   出力部材の断面応力をチェック
C     ―――――――――――――――――――――――――――――――
      subroutine out_section_check(Member,Element,
     *               n_member,No_section,Out_section,
     *               ifl,iflz,i_print)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / Member_s  / Member
      record / Element_s  / Element
      record / Out_section_s  / Out_section
      dimension Member(*),Element(*),No_section(*)
      dimension ifl(16),iflz(16)
C     ―――――――――――――――――――――――――――――――
C     ●   out_section_s 構造体
```

```
C     ―――――――――――――――――――――――――――――――――
c     解析パラメータ
c       structure  / out_section_s/
c       integer    n_member        !  出力部材数
c       integer    n_section       !  出力断面数
c       integer    no_member(10)   !  部材番号
c       integer    no_section(11)  !  断面数
c       integer    no_fiber(30)    !  ファイバー数
c       end structure
c       record /out_section_s/ Out_section
        if(ifl(6).eq.0) return
        nn=No_section(1)                          ! 出力断面数                    ! 1
        if(nn.gt.0) then
        iix=0
        iiy=0
        do i=1,nn                                                                 ! 2
        n=No_section(i+1)
        if(n.ge.1.and.n.le.n_member)then                                          ! 3
        im=Member(n).element_type
        ie= Member(n).nm_element
        if(im.eq.11.or.im.eq.13) then                                             ! 4
        iix=iix+1
        iiy=iiy+2
        Out_section.no_member(iix) = n
        Out_section.no_section(iix)= 2
        Out_section.no_fiber(iiy-1)= Element(ie).nm_section(1)
        Out_section.no_fiber(iiy)  = Element(ie).nm_section(2)
        Out_section.nm_fiber(iiy-1)= Element(ie).n_section(1)
        Out_section.nm_fiber(iiy)  = Element(ie).n_section(2)
        elseif(im.eq.18) then                                                     ! 5
        iix=iix+1
        iiy=iiy+1
        Out_section.no_member(iix) = n
        Out_section.no_section(iix)= 1
        Out_section.no_fiber(iiy)= Element(ie).nm_section(1)
        Out_section.nm_fiber(iiy)= Element(ie).n_section(1)
        elseif(im.eq.12.or.im.eq.14) then                                         ! 6
        iix=iix+1
        iiy=iiy+3
        Out_section.no_member(iix) = n
        Out_section.no_section(iix)= 3
        Out_section.no_fiber(iiy-2)= Element(ie).nm_section(1)
        Out_section.no_fiber(iiy-1)= Element(ie).nm_section(2)
        Out_section.no_fiber(iiy)  = Element(ie).nm_section(3)
        Out_section.nm_fiber(iiy-2)= Element(ie).n_section(1)
        Out_section.nm_fiber(iiy-1)= Element(ie).n_section(2)
        Out_section.nm_fiber(iiy)  = Element(ie).n_section(3)
        endif
        endif
        enddo
        Out_section.n_member = iix
        Out_section.n_section = iiy
        else
        Out_section.n_member = 0
```

```
      endif
c-----------------------------★断面応力ヘッダー出力
      write(iflz(6)) Out_section.n_member                                  ! 7
      if(Out_section.n_member.eq.0) return
      write(iflz(6))(Out_section.no_member(j),j=1,Out_section.n_member)
      write(iflz(6))(Out_section.no_section(j),j=1,Out_section.n_member)
      write(iflz(6)) Out_section.n_section
      write(iflz(6))(Out_section.nm_fiber(j),j=1,Out_section.n_section)
      write(iflz(6))(Out_section.no_fiber(j),j=1,Out_section.n_section)
c-----------------------------★断面応力ヘッダー出力終了
      write(76,'(a,i4)') ' No. member: ',Out_section.n_member
      write(76,'(a,i4)') ' No. section: ',Out_section.n_section
      if(Out_section.n_member.eq.0) return
      return
      end
```

1. ユーザーが出力を要求している部材数を取得する。
2. 出力部材数分、以下の処理を行う。
3. 出力要求を行っている部材番号が解析モデルの部材の範囲かどうかチェックする。範囲外の場合はその部材を無視する。
4. 部材モデルのコード番号が 11 と 13 の場合は、両端ファイバー断面もしくは両端 MS 断面であり、その管理番号をファイバー断面出力構造体 Out_section にセットする。
5. 部材モデルのコード番号が 18 の場合は、両端ピンで中央ファイバー断面を有するモデルであり、その管理番号をファイバー断面出力構造体 Out_section にセットする。
6. 部材モデルのコード番号が 12 と 14 の場合は、両端・中央ファイバー断面もしくは両端・中央 MS 断面であり、その管理番号をファイバー断面出力構造体 Out_section にセットする。
7. 断面ファイバー応力出力ファイルのヘッダー部分を出力する。

次に、実際の断面内ファイバーの応力とひずみを出力するサブルーチン Out_Fiber()を見てみよう。

```
C     ------------------------------------------
C     ●   SUBROUTINE /Out_Fiber
C       ------------------------------------------
C     ●   部材の断面応力を出力(ok)
C     ------------------------------------------
      subroutine Out_Fiber( )
      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                                    ! 1
      i=Out_section.no_member(j)                                     ! 2
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                                             ! 3
c―――――――――――――――――――――――――★モデル１１
      nm_div=E_model11(imm).n_section_1
      nnm=M_model11(immm).nm_section_1 - 1
      do k=1,nm_div
      v(k)=M_model_fiber(k+nnm).d_stress_x
      enddo
      write(iflz(6)) (v(k),k=1,nm_div)
      do k=1,nm_div
      v(k)=M_model_fiber(k+nnm).d_eps_x
      enddo
      vf(1) = M_model11(immm).d_epsi_x_1          !  軸方向歪
      vf(2) = M_model11(immm).d_epsi_y_1          !  y軸に関する曲げ歪
      vf(3) = M_model11(immm).d_epsi_z_1          !  z軸に関する曲げ歪
      write(iflz(6))(vf(k),k=1,3),(v(k),k=1,nm_div)
      nm_div=E_model11(imm).n_section_2
      nnm=M_model11(immm).nm_section_2 - 1
      do k=1,nm_div
      v(k)=M_model_fiber(k+nnm).d_stress_x
      enddo
      write(iflz(6)) (v(k),k=1,nm_div)
      do k=1,nm_div
      v(k)=M_model_fiber(k+nnm).d_eps_x
      enddo
      vf(1) = M_model11(immm).d_epsi_x_2          !  軸方向歪
      vf(2) = M_model11(immm).d_epsi_y_2          !  y軸に関する曲げ歪
      vf(3) = M_model11(immm).d_epsi_z_2          !  z軸に関する曲げ歪
      write(iflz(6))(vf(k),k=1,3),(v(k),k=1,nm_div)
      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.eq.101.or.iet.eq.102) then                    ! 4
c―――――――――――――――――――――――★個人用新規部材モデル

      endif
c―――――――――――――――――――――――――――――★断面応力出力終わり
      enddo
      return
      end
```

1. ユーザーが出力を要求している部材数で以下の出力処理を行う。
2. 出力部材番号を取得する。
3. 両端ファイバーの部材モデル 11 について出力処理を行う。ここのコードを見れば理解できるように、履歴特性に関連するコードはない。そのため、新規ファイバー履歴特性を追加しても、このサブルーチンは変更する必要はない。
4. ここでは、次節以降で説明するファイバー断面を有する部材モデルを新規に作成する場合、この部分に出力コードを追加する必要がある。

1.4.4 構造体の定義

本節では、新規のファイバー用履歴モデルによって、新たな構造体を作成する必要がある場合について説明する。現在、ファイバーを含む部材モデルは全て静的縮合モデルであり、弾性はりエレメントとファイバーによって弾塑性状態を表すエレメント部分などによって構成される。弾性はりに関する情報の保存とワーク領域として、標準型の Element_s、Member_s 構造体が用いられている。ファイバーが集まった断面に関するワーク領域としては M_model_fiber_s が使用され、さらに、ファイバーに関する情報の確保とワーク領域として、構造体 E_model_fiber_s と Bilinear_work、Trilinear_work、Concrete_work が使用されている。なお、部材の弾塑性状態を表すエレメントである MS モデルとアナロジーモデルに関する構造体も、ファイバーモデルと同じ構造体を兼用してい

る。

さて、新たにファイバーモデルの履歴特性を設計する場合であるが、標準入力仕様で、しかも解析中データが変化しないファイバー要素構造体 E_model_fiber_s（鉄骨及び鉄筋用）、及び E_model_fiberc_s（コンクリート用）と解析中データが変化するファイバー部材構造体 M_model_fiber_s の内容で十分な場合は、新たに構造体を設計する必要はない。しかし、入力データ数が多数で現在の仕様では不足する場合や、ワーク領域が不足する場合は、新たに構造体を設計する必要が発生する。これに伴って、既存のコードを変更しなければならなくなる。これについては、後で解説する。しかし、標準仕様の構造体の内容を変更することで使用可能であれば、登録手続きは単純である。

まずは、標準仕様であるファイバー要素構造体 E_model_fiber_s と E_model_fiberc_s を示そう。両者の並びで対応する変数は同じ記憶領域を占めており、構造体配列の数はファイバー要素数分設定される。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   ファイバーモデル E_model_fiber_s 構造体（バイリニアー、トリリニア用）
C     ―――――――――――――――――――――――――――――――――
      structure / E_model_fiber_s/
      integer  nm_type              !  履歴モデルの番号
      real*8   E_1                  !  ファイバーの第一剛性 E1
      real*8   E_2                  !  ファイバーの第二剛性 E2
      real*8   E_3                  !  ファイバーの第三剛性 E3
      real*8   Q_1                  !  ファイバーの第一折れ点
      real*8   Q_2                  !  ファイバーの第二折れ点
      real*8   Ec_1                 !  ファイバーの圧縮側第一剛性 E1
      real*8   Ec_2                 !  ファイバーの圧縮側第二剛性 E2
      real*8   Ec_3                 !  ファイバーの圧縮側第三剛性 E3
      real*8   Qc_1                 !  ファイバーの圧縮側第一折れ点
      real*8   Qc_2                 !  ファイバーの圧縮側第二折れ点
      real*8   G                    !  ファイバーせん断剛性 G
      real*8   A                    !  ファイバー断面積
      real*8   Ay                   !  ファイバーｙ軸せん断用断面積
      real*8   Az                   !  ファイバーｚ軸せん断用断面積
      real*8   ry                   !  中立軸から断面中心までのｙ方向距離
      real*8   rz                   !  中立軸から断面中心までのｚ方向距離
      real*8   Ary                  !  断面積掛ける距離
      real*8   Arz                  !  断面積掛ける距離
      real*8   Ary2                 !  断面積掛ける距離の２乗
      real*8   Arz2                 !  断面積掛ける距離の２乗
      real*8   Aryz                 !  断面積掛ける距離の２乗
      real*8   E_4                  !  移動硬化用パラメータ
      real*8   Ec_4                 !  移動硬化用パラメータその２
      end structure
c       record / E_model_fiber_s    / E_model_fiber
c
C     ―――――――――――――――――――――――――――――――――
```

```
C     ●   ファイバーモデル E_model_fiberc_s 構造体（コンクリート用）
C     ―――――――――――――――――――――――――――――――――
      structure / E_model_fiberc_s/
      integer  nm_type              !  履歴モデルの番号
      real*8   AK_1                 !  圧縮と引張　第一勾配
      real*8   AK_2                 !  圧縮第二勾配
      real*8   AK_3                 !  圧縮第三勾配
      real*8   Q_1                  !  引張強度
      real*8   Q_2                  !  圧縮側第一折れ点の応力
      real*8   Q_3                  !  圧縮強度
      real*8   Q_4                  !  圧縮流れ点
      real*8   AK_4                 !  引張第二勾配
      real*8   dm                   !  ダミー
      real*8   dm1                  !  ダミー
      real*8   G                    !  ファイバーせん断剛性 G
      real*8   A                    !  ファイバー断面積
      real*8   Ay                   !  ファイバーｙ軸せん断用断面積
      real*8   Az                   !  ファイバーｚ軸せん断用断面積
      real*8   ry                   !  中立軸から断面中心までのｙ方向距離
      real*8   rz                   !  中立軸から断面中心までのｚ方向距離
      real*8   Ary                  !  断面積掛ける距離
      real*8   Arz                  !  断面積掛ける距離
      real*8   Ary2                 !  断面積掛ける距離の２乗
      real*8   Arz2                 !  断面積掛ける距離の２乗
      real*8   Aryz                 !  断面積掛ける距離の２乗
      real*8   Beta                 !  移動硬化用パラメータ
      real*8   Beta_2               !  移動硬化用パラメータその２
      end structure
c       record / E_model_fiberc_s    / E_model_fiber
```

上の構造体 E_model_fiber_s をそのまま利用することも可能であるが、新規の履歴モデル用として、新たに設計することもできる。上に示した構造体 E_model_fiber_s を参照して作成すれば良い。この構造体は、ファイバー要素データを保存するためのものであり、部材毎に計算途中で変化するデータを入れるものではない。これについては、ファイバー部材構造体 M_model_fiber_s が用意されている。ただし、この構造体の全ての成分が使用可能となっておらず、上記の M_model_fiber_s の中で濃い文字で示したエレメントはシステムが使用する。

次に、ファイバー部材構造体 M_model_fiber_s を示そう。ここでも、新たな構造体を設計したい場合は、この構造体 M_model_fiber_s を参照して作成されたい。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   ファイバーモデル M_model_fiber_s 構造体
C     ―――――――――――――――――――――――――――――――――・
      structure / M_model_fiber_s/
      integer  n_type               !  履歴モデルの通し番号
      integer  i_elastic            !  ファイバー要素の状態（弾性の場合は０：塑性は１）
      real*8   d_eps_x              !  軸方向歪
```

```
      real*8   d_stress_x          !  軸方向応力
      real*8   d_E                 !  接線剛性
      end structure
c       record / M_model_fiber_s    / M_model_fiber
```

さらに、ファイバーモデルでは、ワーク用として履歴特性に合わせて３つの構造体が設計され、使用されている。この３つの構造体は、現在以下の８つの履歴特性で用いられている。

１．バイリニア型ワーク領域構造体：Bilinear_work_s
- **１．対称バイリニア型：BiLinear()**
- **５．バイリニア型（移動＋等方硬化用）：BiLinear_h()**
- **７．非対称バイリニア型：TriLinear_AS()**

２．トリリニア型ワーク領域構造体：Trilinear_work_s
- **２．対称トリリニア型：TriLinear()**
- **６．対称トリリニア型（移動＋等方硬化用）：TriLinear_h()**
- **８．非対称トリリニア型：BiLinear_AS()**

３．コンクリート型ワーク領域構造体：Concrete_work_s
- **３．直線コンクリート履歴モデル：Concrete()**
- **４．曲線コンクリート履歴モデル：Concrete_e()**

上記３つのワーク領域構造体を以下に示そう。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   履歴モデル   Bilinear：ファイバー要素 用  Work エリア構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / Bilinear_work_s/
      integer  i_stat              !  ファイバー要素の状態
      real*8   P1(5)               !  ワーク領域
      real*8   Sig_z               !  ワーク領域
      end structure
c       record / Bilinear_work_s    / Bilinear_work
c
C     ―――――――――――――――――――――――――――――――――
C     ●   履歴モデル   Trilinear：ファイバー要素 用  Work エリア構造体
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / Trilinear_work_s/
      integer  i_stat              !  ファイバー要素の状態
      real*8   P1(15)               !  ワーク領域
      end structure
c       record / Trilinear_work_s    / Trilinear_work
c
C     ―――――――――――――――――――――――――――――――――
C     ●   履歴モデル   Concrete：ファイバー要素 用  Work エリア構造体
```

```
C     ―――――――――――――――――――――――――――――――――
c
c     部材
      structure / Concrete_work_s/
      integer  i_stat              !  ファイバー要素の状態
      integer  ipret               !  過去の引張履歴
      integer  ipre_c              !  過去の圧縮履歴
      real*8   P1(6)               !  ワーク領域
      end structure
c       record / Concrete_work_s    / Concrete_work
```

標準仕様の構造体を使用する場合は、新たな構造体を必要としないが、その新規モデルに適合した構造体を設計する場合は、その構造体を新たなヘッダーファイルに作成する必要がある。このヘッダーファイルを New_submain.h としよう。読者はまずこの New_submain.h ファイルを作り、この中に新たな構造体を書き込むことになる。新たなヘッダーファイルは、この構造体を使用する場合に、そのサブルーチンの中にインクルードされる。

新規にファイバー要素の履歴特性を設計する際、入力データやワーク領域が先に示した構造体では対応できない場合、新たな構造体を設計し、使用することになる。その際、多くの手続きが必要となり、既存のプログラムに手続きのためのコードを追加する必要が生じる。その手続きを知るためにも、現在の構造体がどのような手続きで利用可能となっているか調査しよう。

現在、ファイバー履歴に関連する構造体として、ファイバー断面に関する構造体２種類、ファイバーの履歴に関する構造体３種類が用いられている。これらの構造体は、submain_dynamic_a()サブルーチンの中で、構造体の手続きにしたがって、宣言、動的確保、領域解放が行われている。その手続きの幾つかを以下に示す。最初に、宣言は、

```
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―――――――――――――――――――――――――――――――★履歴モデル用エリア
      record / Bilinear_work_s    / Bilinear_work
      record / Trilinear_work_s   / Trilinear_work
      record / Concrete_work_s    / Concrete_work
c―――――――――――――――――――――――――――――――★ファイバーモデル用エリア
      ALLOCATABLE :: E_model_fiber(:)
      ALLOCATABLE :: M_model_fiber(:)
c―――――――――――――――――――――――――――――――★履歴モデル用エリア
      ALLOCATABLE :: Bilinear_work(:)
      ALLOCATABLE :: Trilinear_work(:)
      ALLOCATABLE :: Concrete_work(:)
```

これらの構造体は、次のように解析モデルで使用する数に従って、動的に領域を確保されなければならない。当然、この構造体の数をどこかで数えておく必要がある。

```
c―――――――――――――――――――――――――★ファイバーモデル領域セット
      n= Model_type.nm_div_fmodel          !部材内のファイバー要素の数
      if(n.ne.0) then
      ALLOCATE (M_model_fiber(n))
      endif
      n= Model_type.nm_div_felement        !ファイバー要素のエレメント最大数
      if(n.ne.0) then
      ALLOCATE (E_model_fiber(n))
      endif
c―――――――――――――――――――――――――――――――★ファイバー用履歴要素
      n= Model_type.n_m_bilinear          !  バイリニアの履歴要素の数
      if(n.ne.0) then
      ALLOCATE (Bilinear_work(n))
      endif
      n= Model_type.n_m_trilinear          !  トリリニアの履歴要素の数
      if(n.ne.0) then
      ALLOCATE (Trilinear_work(n))
      endif
      n= Model_type.n_m_Concrete          !  コンクリートの履歴要素の数
      if(n.ne.0) then
      ALLOCATE (Concrete_work(n))
      endif
```

これらの構造体は動的に確保する配列であることから、サブルーチンsubmain_dynamic_a()のなかで確保する必要がある。手続きとしては、動的確保の他に、動的領域であることの宣言、使用後の領域解放を行うことになる。構造体を新たに設計、使用する場合は、これらの処理を必ず既存プログラムの中に書き込まなければならない。

1.4.5 必要となるサブルーチン

次に、この履歴特性New_model_fiber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。ただし、この中で標準の静的縮合モデルに関連するサブルーチンを転用できる場合は、そのサブルーチンを使用すれば良い。まず、上

記４つの標準的な静的縮合モデルに関するサブルーチンを検討しよう。例として、両端ファイバー部材モデルを取り上げる。

最初は、線形剛性行列を求めるサブルーチン Fiber_Model_GI11()であり、次に示す。

```
C     －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
C     ●    SUBROUTINE /Fiber_Model_GI11
C     －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
C     ●    線形剛性行列の計算(ok)
C     －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
C     ●    Model_No.11 両端ファイバーモデル
C     －－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
      subroutine Fiber_Model_GI11(it,ak,alength,Member,Element,
     *           E_model,E_model_fiber,M_model,M_model_fiber,
     *           Bilinear_work,Trilinear_work,Concrete_work,New_fiber_work)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s          / Member
      record / element_s         / Element
      record / E_model11_s       / E_model
      record / E_model_fiber_s   / E_model_fiber
      record / M_model11_s       / M_model
      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
      record / New_fiber_work_s  / New_fiber_work
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension Bilinear_work(*),Trilinear_work(*)
      dimension Concrete_work(*),New_fiber_work(*)
      dimension ak(12,12)
c－－－－－－－－－－－－－－－－－－－－－－－－－－－★
      if(it.eq.1) then
      do i=1,30
      M_model.ff_ip(i) = 0.
      enddo
      nm_div = E_model.n_section_1
      nn     = E_model.nm_section_1 - 1
      nnm    = M_model.nm_section_1 - 1
      elseif(it.eq.2) then
      nm_div = E_model.n_section_2
      nn     = E_model.nm_section_2 - 1
      nnm    = M_model.nm_section_2 - 1
      endif
      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
      if(nm_type.le.10) then
      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
      else
      goto ( 110,120),nm_type-100
  110 continue
c-----------------------★個人用ファイバーモデル
      New_fiber_work(nmx).i_stat  = -1            !  新規ファイバー要素の状態
      goto 100
  120 continue
c-----------------------★
      goto 100
  100 continue
      enddo
c-------------------------------★ファイバー要素のデータセット
      if(it.eq.1) then
      nm_div = E_model.n_section_1
      nn     = E_model.nm_section_1 - 1
      nnm    = M_model.nm_section_1 - 1
      elseif(it.eq.2) then
      nm_div = E_model.n_section_2
      nn     = E_model.nm_section_2 - 1
      nnm    = M_model.nm_section_2 - 1
      endif
      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
       if(it.eq.1) then
       M_model.d_aa_1     = aa              !  断面積の和
       M_model.d_ra_1     = ra              !  E*断面積の和
       M_model.d_ray_1    = ray             !  E*A*z
       M_model.d_raz_1    = raz             !  E*A*y
       M_model.d_raz2_1   = raz2            !  E*A*y*y
       M_model.d_ray2_1   = ray2            !  E*A*z*z
       M_model.d_rayz_1   = rayz            !  E*A*z*y
       M_model.d_gg_1     = gg              !  G*A
       M_model.d_epsi_x_1 =  0.             !  軸方向歪
       M_model.d_epsi_y_1 =  0.             !  y軸に関する曲げ歪
       M_model.d_epsi_z_1 =  0.             !  z軸に関する曲げ歪
       else
       M_model.d_aa_2     = aa              !  断面積の和
       M_model.d_ra_2     = ra              !  E*断面積の和
       M_model.d_ray_2    = ray             !  E*A*z
       M_model.d_raz_2    = raz             !  E*A*y
       M_model.d_raz2_2   = raz2            !  E*A*y*y
       M_model.d_ray2_2   = ray2            !  E*A*z*z
       M_model.d_rayz_2   = rayz            !  E*A*z*y
       M_model.d_gg_2     = gg              !  G*A
       M_model.d_epsi_x_2 =  0.             !  軸方向歪
       M_model.d_epsi_y_2 =  0.             !  y軸に関する曲げ歪
       M_model.d_epsi_z_2 =  0.             !  z軸に関する曲げ歪
       endif
c――――――――――――――――――――――――――――――★初期剛性行列の計算
       call Fiber_Model_G11(it,ak,alength,Member,Element,
      *      E_model,E_model_fiber,M_model,M_model_fiber)
       return
       end
```

このサブルーチンを見れば分かるように、ファイバー要素に関する構造体の中で、Bilinear_work、Trilinear_work、Concrete_work 以外に、新たな構造体 New_fiber_work を使用する場合は、太文字で示した部分を変更し、初期設定を行うことになる。

次に、非線形剛性を求めるサブルーチン Stiff_M11()について調査す

る。このプログラムは、以下のように履歴モデルに関連するコードを含まない。そのため、新規履歴モデルを追加しても非線形剛性を求めるサブルーチンには影響しない。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Stiff_M11
C       ―――――――――――――――――――――――――――――――――
C     ●   接線剛性行列の計算(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Stiff_M11(im,it,n_type,ak,Member,alength,
     *       Model_type,Element,
     *     E_model11, E_model_fiber,M_model11, 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_model11_s       / E_model11
      record / M_model11_s       / M_model11
      record / M_model_fiber_s   / M_model_fiber
      record / E_model_fiber_s   / E_model_fiber
      dimension ak(12,12),alength(*)
      dimension vv(12)
      dimension E_model_fiber(*),M_model_fiber(*)
c―――――――――――――――――――――――――――――――★要素及びモデルのセット
      do i=1,12
      do j=1,12
      ak(j,i)=0.
      enddo
      enddo
      goto(11,12,13,14,15,16,17,18,19,20),n_type
   11 continue
c―――――――――――――――――――――――――――――――★弾性要素
      call Cal_nonlin_stiff_Mx(Member,Element,
     *       Member.stress(7),ak,alength(im) )
      goto 100
   12 continue
c―――――――――――――――――――――――――――――――★ファイバー要素
      it=it+1
      call Fiber_Model_G11(it,ak,alength(im),Member,Element,
     *      E_model11,E_model_fiber,
     *      M_model11,M_model_fiber)
      goto 100
   13 continue
c―――――――――――――――――――――――――――――――★せん断バネ要素
      call Shear_G(it,ak,Member)
      goto 100
   14 continue
c―――――――――――――――――――――――――――――――★ダミー
      goto 100
   15 continue
```

```
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   16 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   17 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   18 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   19 continue
c――――――――――――――――――――――――――――――――★ダミー
      goto 100
   20 continue
c――――――――――――――――――――――――――――――――★ダミー
  100 continue
      return
      end
```

```
C     ――――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Fiber_Model_G11
C     ――――――――――――――――――――――――――――――――――――――
C     ●   接線剛性行列の計算
C     ――――――――――――――――――――――――――――――――――――――
      subroutine Fiber_Model_G11(it,ak,alength,Member,Element,
     *      E_model,E_model_fiber,M_model,M_model_fiber)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
C     ――――――――――――――――――――――――――――――――――――――
      record / member_s          / Member
      record / element_s         / Element
      record / E_model11_s       / E_model
      record / M_model11_s       / M_model
      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――――――――――――――――――――――――――――――――★
      if(it.eq.1) then                  !  it:部材位置（１：i 端、 ２：j 端）
      aa   = M_model.d_aa_1             !  断面積の和
      ra   = M_model.d_ra_1             !  E*断面積の和
      ray  = M_model.d_ray_1            !  E*A*z
      raz  = M_model.d_raz_1            !  E*A*y
      raz2 = M_model.d_raz2_1           !  E*A*y*y
      ray2 = M_model.d_ray2_1           !  E*A*z*z
      rayz = M_model.d_rayz_1           !  E*A*z*y
      gg   = M_model.d_gg_1             !  G*A
      else
      aa   = M_model.d_aa_2             !  断面積の和
      ra   = M_model.d_ra_2             !  E*断面積の和
      ray  = M_model.d_ray_2            !  E*A*z
      raz  = M_model.d_raz_2            !  E*A*y
```

```
      raz2 = M_model.d_raz2_2          !  E*A*y*y
      ray2 = M_model.d_ray2_2          !  E*A*z*z
      rayz = M_model.d_rayz_2          !  E*A*z*y
      gg   = M_model.d_gg_2            !  G*A
      endif
      rix  = Element.RIx
      asy  = Element.ASy
      asz  = Element.ASz
c-----------------------------------★
      call Fiber_GK(ak,alength,ra,ray,raz,
     *              raz2,ray2,rayz,gg,aa,rix,asy,asz)
      return
      end
```

次に、応力を求めるサブルーチンを調べよう。以下に示すサブルーチン Cal_stress_M11()の内容から、静的縮合型部材モデルでは、特殊な応力計算を行わない限り、現在のサブルーチンが転用可能となる。

```
C     ------------------------------------------
C     ●   SUBROUTINE /Cal_stress_M11（両端ファイバーモデル）
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_M11(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
```

以上のように、静的縮合部材モデルのファイバー履歴特性では、新たな構造体を追加しなければ、標準仕様の３つのサブルーチンがそのまま

使用できることが分かる。後は応力をチェックし、接線剛性を求めるサブルーチンであるが、このサブルーチンについては、次節で説明しよう。

1.4.6 モデルの階層構造とサブルーチンの組み込み

静定縮合部材モデルでは、サブルーチン submain_dynamic_a()から Check_stress()がコールされ、その中で、両端ファイバーモデルの応力の弾塑性チェックプログラム Cal_check_stiff_M11()が呼び出される。以下にその部分のコードを示す。

```
      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_stiff_M11(Control,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)
```

さらに、このサブルーチン Cal_check_stiff_M11()は、以下のような静的縮合に関する処理を行い、ファイバーの弾塑性チェックを行う。このサブルーチンから関連する部分を抜き出してみよう。

```
C     ――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_check_stiff_M11
C       ――――――――――――――――――――――――――――――――
C     ●   代表的な部材モデルの塑性チェック(両端ファイバーモデル)
C     ――――――――――――――――――――――――――――――――
      subroutine Cal_check_stiff_M11( )
C     ――――――――――――――――――――――――――――――――
      .
      .
c――――――――――――――――――――――――――――――――★部材剛性行列の作成
      it = 0
      do i=1,n_div
c――――――――――――――――――――――――――――――――★内部部材の剛性行列の計算
      call Stiff_M11(i,it,n_type(i),akk(1,1,i),Member,alength,
     *       Model_type,Element,
     *       E_model11(imm), E_model_fiber,
     *       M_model11(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(Member.an_stress(i),Member,
     *                       akk(1,1,i),alength(i))
      call Create_Kn(akk(1,1,i),Member.an_vv(i),Member.an_ww(i),
     *              EA(i),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
```

```
      enddo
       ・
       ・
c―――――――――――――――――――――――――――――――――――――――
c    ★        部材の弾塑性チェック
c――――――――――――――――――――――――――★ファイバーチェック
      if(n_type(i).eq.2) then
      it=it+1
      call Fiber_check(Control,i,N_analysis,
     *      mem_x,it,alength(i),Member,Element,
     *      E_model11(imm),E_model_fiber,M_model11(immm),M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vvx)
      endif
       ・
       ・
      return
      end
```

このサブルーチンの中で、接線剛性を求める Stiff_M11()サブルーチンがコールされているが、この中でこの静的縮合モデルで使用されるエレメントモデルが階層構造となっている。このサブルーチンは前節で示した。

さらに、このサブルーチンの中で、静定縮合部材モデルである両端ファイバーモデルの応力チェック用サブルーチン Fiber_check()がコールされている。次は、この応力チェック用サブルーチンを見てみよう。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Fiber_check11
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_check(idiv,N_analysis,
     *      mem_x,it,Alength,Member,Element,
     *      E_model,E_model_fiber,M_model,M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv)
      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_model11_s       / E_model
      record / E_model_fiber_s   / E_model_fiber
      record / M_model11_s       / M_model
      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
      if(it.eq.1) then
      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(Member.an_vv(idiv),
     *      Member.an_ww(idiv),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_model.d_epsi_x_1 =  d_epsi_x_1 + M_model.d_epsi_x_1    !  軸方向歪
      M_model.d_epsi_y_1 =  d_epsi_y_1 + M_model.d_epsi_y_1     !  y軸に関する曲げ歪
      M_model.d_epsi_z_1 =  d_epsi_z_1 + M_model.d_epsi_z_1     !  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_model.n_section_1
      nn       = E_model.nm_section_1 - 1
      nnm      = M_model.nm_section_1 - 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
      M_model_fiber(nnm).d_eps_x = M_model_fiber(nnm).d_eps_x + du !ファイバー要素の歪
      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
      if(nm_type.le.10) then
         goto ( 10,20,30,40,50,60,70,80,90,95),nm_type
         elseif(nm_type.ge.15.and.nm_type.le.20)then
         goto ( 110,120,130,140,150,160),nm_type-14

  10  continue
c――――――――――――――――――――――――★バイリニア型（スチール用）
      call BiLinear()
      goto 100
  20  continue
c――――――――――――――――――――――――★対称トリリニア型 （スチール用）
      call TriLinear()
      goto 100
  30  continue
c――――――――――――――――――――――――★コンクリート型
```

```
      call Concrete()
      goto 100
   40 continue
c――――――――――――――――――――――――★曲線コンクリート型
      call Concrete_e()
      goto 100
   50 continue
c――――――――――――――――――――――――★バイリニア型（移動＋等方効果用）
      call BiLinear_h()
      goto 100
   60 continue
c――――――――――――――――――――――――★対称トリリニア型　（移動＋等方効果用）
      call TriLinear_h()
      goto 100
   70 continue
c――――――――――――――――――――――――★非対称バイリニア型
      call BiLinear_AS()
      goto 100
   80 continue
c――――――――――――――――――――――――★非対称トリリニア型
      call TriLinear_AS()
90       continue
c――――――――――――――――――――――――★降伏棚を有する対称バイリニア型（移動＋等方効果）
         goto 100
95       continue
c――――――――――――――――――――――――★降伏棚を有する対称バトリリニア型（移動＋等方効果）

         goto 100
110      continue
c――――――――――――――――――――――――★鉄筋用履歴モデル
      call reinforcement_b()
         goto 100
120      continue
c――――――――――――――――――――――――★木質構造材用履歴モデル
         call Wood_TriLinear()
         goto 100
130      continue
c――――――――――――――――――――――――★
         goto 100
140      continue
c――――――――――――――――――――――――★
         goto 100
150      continue
c――――――――――――――――――――――――★
         goto 100
160      continue
c――――――――――――――――――――――――★
         goto 100
c――――――――――――――――――――――――★弾塑性解析
      goto (1110,1120),nm_type - 100
 1110 continue
c――――――――――――――――――――――――★個人用ファイバー履歴特性
      call New_model_fiber()
      goto 100
```

```
 1120 continue
c――――――――――――――――――――――――――★個人用ファイバー履歴特性
      goto 100
      endif
      endif
c――――――――――――――――――――――――――★弾塑性解析終了
  100 continue
      enddo
c――――――――――――――――――――――――――――――――――――――★ファイバー要素応力の計算
      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(1) = 0
      elseif(d_state .ge.0.8) then
      Member.d_stat(1) = 2
      else
      Member.d_stat(1) = 1
      endif
      nm_div  = E_model.n_section_1
      nn      = E_model.nm_section_1 - 1
      nnm     = M_model.nm_section_1 - 1
      ra      = 0.
      ray     = 0.
      raz     = 0.
      raz2    = 0.
      ray2    = 0.
      rayz    = 0.
      gg      = 0.
      aa      = 0.
      AN      = 0.
      AMy     = 0.
      AMz     = 0.
      ra_before = M_model.d_ra_1
      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
      if(iistat.ge.nm_div) then
      ra=ra_before
      endif
      M_model.d_aa_1 = aa                  !  断面積の和
```

```
      M_model.d_ra_1 = ra                !  E*断面積の和
      M_model.d_ray_1 = ray              !  E*A*z
      M_model.d_raz_1 = raz              !  E*A*y
      M_model.d_raz2_1 = raz2            !  E*A*y*y
      M_model.d_ray2_1 = ray2            !  E*A*z*z
      M_model.d_rayz_1 = rayz            !  E*A*z*y
      M_model.d_gg_1  = gg               !  G*A
c-----------------------------------★j 端部ファイバー
c-----------------------------------★歪のセット
      elseif(it.eq.2) then
       •
       •
      do i=1,nm_div
       •
       •
      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.le.10) then
         goto ( 11,21,31,41,51,61,71,81,91,99),nm_type
         elseif(nm_type.ge.15.and.nm_type.le.20)then
         goto ( 111,121,131,141,151,161),nm_type-14

c-----------------------★弾塑性解析
      goto ( 11,21,31,41,51,61,71),nm_type
  11  continue
c-----------------------★バイリニア型（スチール用）
      call BiLinear()
      goto 101
  21  continue
c-----------------------★対称トリリニア型 （スチール用）
      call TriLinear()
      goto 101
  31  continue
c-----------------------★コンクリート型
      call Concrete()
      goto 101
  41  continue
c-----------------------★曲線コンクリート型
      call Concrete_e(M_model_fiber()
      goto 101
  51  continue
c-----------------------★バイリニア型（移動＋等方効果用）
      call BiLinear_h()
      goto 101
  61  continue
c-----------------------★対称トリリニア型 （移動＋等方効果用）
      call TriLinear_h()
      goto 101
  71  continue
c-----------------------★非対称バイリニア型
```

```
      call BiLinear_AS()
      goto 101
   81 continue
c――――――――――――――――――――――――★非対称トリリニア型
      call TriLinear_AS()
   91 continue
c――――――――――――――――――――――――★降伏棚を有する対称バイリニア型（移動＋等方効果）
      goto 101
   99 continue
c――――――――――――――――――――――――★降伏棚を有する対称バトリリニア型（移動＋等方効果）
      goto 101
  111 continue
c――――――――――――――――――――――――★鉄筋用履歴モデル
      call reinforcement_b()
      goto 101
  121 continue
c――――――――――――――――――――――――★木質構造材用履歴モデル
      call Wood_TriLinear()
      goto 101
  131 continue
c――――――――――――――――――――――――★
      goto 101
  141 continue
c――――――――――――――――――――――――★
      goto 101
  151 continue
c――――――――――――――――――――――――★
      goto 101
  161 continue
c――――――――――――――――――――――――★
      goto 101
      elseif(nm_type/10.eq.10) then
c――――――――――――――――――――――――★弾塑性解析
      goto (111,121),nm_type - 100
  111 continue
c――――――――――――――――――――――――★個人用ファイバー履歴特性
      call New_model_fiber()
      goto 101
  121 continue
c――――――――――――――――――――――――★個人用ファイバー履歴特性
      goto 101
      endif
      endif
c――――――――――――――――――――――――★弾塑性解析終了
  101 continue
      enddo
c――――――――――――――――――――――――――――――――★ファイバー要素応力の計算
       .
       .
      return
      end
```

このサブルーチンの中に履歴モデルに関する第３層の階層構造が見ら

れる。この中に履歴特性番号 101 として、太文字で示すコードがあり、新規モデルが追加されている。この中で、応力を弾塑性チェックする新規履歴モデルに関するサブルーチンが両端ファイバーモデルであることから、2 箇所に変更が加えられている。ここでは、サブルーチン名を New_model_fiber()としている。これで、既存サブルーチンに新規モデルが組み込まれたことになる。引数として受け渡すデータとして、構造体の Member と Element など多くの構造体や配列が、新規に登録されたサブルーチンには見られるはずである。新規サブルーチンでもデータの受け渡しには細心の注意が必要である。また、特殊な構造体を設計した場合、その構造体は引数として受け渡すことになるが、当然、上位のサブルーチンから受け渡されることになる。

この新規サブルーチンでは、実際にファイバーの履歴を追うことが主な処理であるが、少なくとも構造体 M_model_fiber と３つのワーク領域 Bilinear_work、Trilinear_work、Concrete_work かもしくは新たに設計した New_fiber_work 中で、次の値を設定しなければならない。

1. New_fiber_work (nmx).i_stat が-1 のとき、初期設定を行う。
2. ファイバーの接線剛性(M_model_fiber(nnm).d_E を設定する。
3. ファイバーの応力 M_model_fiber(nnm).d_stress_x を求める。

上記の組み込みは、部材モデル 11 について解説した。ファイバー断面を使用している部材モデルは、部材モデル番号 11、12、15、と任意型部材モデルであり、新規のファイバー履歴モデルを組み込む場合は、常に、これらの部材モデルに同じように設定しなければならない。前 3 つの部材モデルは同じ構成をしているため、ここで説明した手続きを用いて、新規のファイバー履歴モデルを組み込むことになる。

1.4.7 任意型静的縮合モデルにおけるファイバー履歴の組み込み

ファイバー履歴は、任意型の部材モデルにも組み込まれている。したがって、ファイバー履歴を追加する場合は、この部材モデルにも追加しておかれたい。このモデルは、他の静的縮合部材モデルに比較して階層が深くなっている。まず、任意型の部材モデルで応力チェックを行うサ

ブルーチン Cal_check_stiff_Mx()を示す。

```
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     ―――――――――――――――――――――――――――――――――
       .
       .
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
c      write(76,'(a,10i4)') ' n_type ',i,imm,immm,n_type(i)
      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
c      call MS_checkx(N_analysis,
c     *      mem_x,alength(i),Member,Element,
c     *      E_modelx(imm),E_model_fiber,M_modelx(immm),M_model_fiber,
c     *      Bilinear_work,Trilinear_work,Concrete_work,vvx,
c     *      M_Fiber_Work(nmmx+i).an_vv, M_Fiber_Work(nmmx+i).an_ww,
c     *      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――――――――――――――――――――――――――――――――★接合部ばねモデル要素
         call Wood_Joint_checkx(Control,N_analysis,Member,
     *                                       Element,E_modelx(imm),E_model_fiber,
     *                                       M_modelx(immm),M_model_fiber,Bilinear_work,
     *                                       Trilinear_Work,vvx)
      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))
c      write(76,'(a,i5,2e12.4)') ' vv ww',i,M_Fiber_Work(nmmx+i).an_vv,
c     * M_Fiber_Work(nmmx+i).an_ww
```

```
      enddo
c――――――――――――――――――――――――――――――――★動的記憶領域の解放
      DEALLOCATE (
     *   c ,ab ,irest_point,n_type,alength,
     *   bav,akk,f1,f2    )
      return
      end
```

太文字で示したサブルーチンで、ファイバーの履歴をチェックしている。このサブルーチン Fiber_checkx()を以下に示す。

```
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
c      write(76,'(a,3i6)') ' check_fiber',nm_div,nn,nnm
      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
c      write(76,'(a,9i6)') ' check_nm_div',i,nn,nnm,nmx,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.le.10) then
c-----------------------------★弾塑性解析
      goto ( 10,20,30,40,50,60,70,80,90,99),nm_type
      elseif(nm_type.ge.10.and.nm_type.le.20)then
      goto ( 110,120,130,140,150,160),nm_type-14

  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(1))
      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),Trilinear_work(nmx).P1(2),
     *          Trilinear_work(nmx).P1(3),Trilinear_work(nmx).P1(4),
     *          Trilinear_work(nmx).P1(5),Trilinear_work(nmx).P1(6))
      if(Trilinear_work(nmx).i_stat.eq.2.or.
     *   Trilinear_work(nmx).i_stat.eq.3.or.
     *   Trilinear_work(nmx).i_stat.eq.4.or.
     *   Trilinear_work(nmx).i_stat.eq.5) 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(1),Bilinear_work(nmx).P1(2),
     *          Bilinear_work(nmx).P1(3),
     *         M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *         M_model_fiber(nnm).du_c)
      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),
```

```
     *          Trilinear_work(nmx).P1(3),Trilinear_work(nmx).P1(4),
     *          Trilinear_work(nmx).P1(5),Trilinear_work(nmx).P1(6),
     *          Trilinear_work(nmx).P1(7),
     *         M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *         M_model_fiber(nnm).du_c)
      if(Trilinear_work(nmx).i_stat.eq.2.or.
     *   Trilinear_work(nmx).i_stat.eq.3.or.
     *   Trilinear_work(nmx).i_stat.eq.4.or.
     *   Trilinear_work(nmx).i_stat.eq.5) 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).E_3,E_model_fiber(nn).Q_1,
     *          E_model_fiber(nn).Ec_1,E_model_fiber(nn).Ec_2,
     *         E_model_fiber(nn).Ec_3,E_model_fiber(nn).Qc_1,
     *          du,M_model_fiber(nnm).d_stress_x,
     *          Bilinear_work(nmx).P1(1),Bilinear_work(nmx).P1(2),
     *          Bilinear_work(nmx).P1(3),
     *          M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *         M_model_fiber(nnm).du_c,
     *          M_model_fiber(nnm).E_5,M_model_fiber(nnm).Ec_5)
      if(Bilinear_work(nmx).i_stat.eq.2.or.
     *   Bilinear_work(nmx).i_stat.eq.3.or.
     *   Bilinear_work(nmx).i_stat.eq.4.or.
     *   Bilinear_work(nmx).i_stat.eq.5) 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).E_4,
     *         E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *          E_model_fiber(nn).Ec_1,E_model_fiber(nn).Ec_2,
     *          E_model_fiber(nn).Ec_3,E_model_fiber(nn).Ec_4,
     *          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),Trilinear_work(nmx).P1(4),
     *          Trilinear_work(nmx).P1(5),Trilinear_work(nmx).P1(6),
     *          Trilinear_work(nmx).P1(7),
     *          M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *         M_model_fiber(nnm).du_c,
     *          M_model_fiber(nnm).E_5,M_model_fiber(nnm).Ec_5)
      if(Trilinear_work(nmx).i_stat.eq.2.or.
     *   Trilinear_work(nmx).i_stat.eq.3.or.
     *   Trilinear_work(nmx).i_stat.eq.4.or.
     *   Trilinear_work(nmx).i_stat.eq.5.or.
     *   Trilinear_work(nmx).i_stat.eq.6.or.
     *   Trilinear_work(nmx).i_stat.eq.7) iistat = iistat + 1
      goto 100
   90 continue
```

```
c――――――――――――――――――――――――★降伏棚を有する対称バイリニア型(タイプ 1,2)（移動＋等方効果用）
          goto 100
99        continue
c――――――――――――――――――――――――★降伏棚を有する対称トリリニア型(タイプ 1,2)（移動＋等方効果用）
          goto 100
110       continue
c――――――――――――――――――――――――★鉄筋用履歴モデル(タイプ 1,2)
       call reinforcement_b(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).Q_1,
     *          du,M_model_fiber(nnm).d_stress_x,
     *          Trilinear_work(nmx).P1(1),Trilinear_work(nmx).P1(2),
     *          Trilinear_work(nmx).P1(3),Trilinear_work(nmx).P1(4),
     *          Trilinear_work(nmx).P1(5),Trilinear_work(nmx).P1(6),
     *          Trilinear_work(nmx).P1(7),Trilinear_work(nmx).P1(8),
     *          M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *          M_model_fiber(nnm).du_c,
     *          M_model_fiber(nnm).E_5,M_model_fiber(nnm).Ec_5)
       if(Bilinear_work(nmx).i_stat.eq.1.or.
     *   Bilinear_work(nmx).i_stat.eq.2.or.
     *   Bilinear_work(nmx).i_stat.eq.5.or.
     *   Bilinear_work(nmx).i_stat.eq.6) iistat = iistat + 1
          goto 100
120       continue
c――――――――――――――――――――――――★木質構造材用履歴モデル
          call Wood_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,
     *          E_model_fiber(nn).Ec_1,E_model_fiber(nn).Ec_2,
     *          E_model_fiber(nn).Ec_3,
     *          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),
     *          M_model_fiber(nnm).d_eps_x,M_model_fiber(nnm).du_t,
     *          M_model_fiber(nnm).du_c)
       if(Trilinear_work(nmx).i_stat.eq.1.or.
     *   Trilinear_work(nmx).i_stat.eq.2.or.
     *   Trilinear_work(nmx).i_stat.eq.5.or.
     *   Trilinear_work(nmx).i_stat.eq.6.or.
     *   Trilinear_work(nmx).i_stat.eq.8) iistat = iistat + 1
          goto 100
130       continue
c――――――――――――――――――――――――★
          goto 100
140       continue
c――――――――――――――――――――――――★
          goto 100
150       continue
c――――――――――――――――――――――――★
          goto 100
160       continue
```

```
c――――――――――――――――――――――――★
        goto 100
c――――――――――――――――――――――――★弾塑性解析終了
      endif
      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
     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.5) 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
```

このサブルーチンは、先に示した両端ファイバーの部材モデル番号 11 に関するサブルーチン Fiber_check11()とプログラムの構造はほとんど同じであり、他の部材モデルと同一のファイバー履歴番号を用いることになる。ここでは、個人用ファイバー履歴特性として、上記のようにサブルーチン New_model_fiber()を組み込むことができる。このように新規のファイバー用履歴特性を組み込むことは容易である。