5.9 ファイバーモデル

5.9.1 ファイバーモデルの概要

ファイバーモデルとは、断面内を細分化して、その細分化した要素の応力とひずみの関係を一軸状態で表現する簡易なモデルである。ここでは、はりのひずみは平面保持の仮定を用いて、図芯位置での軸方向ひずみと面外の 2 方向曲げひずみによって決定する。詳しくは、理論マニュアルを見られたい。

ファイバーモデルにおける接線剛性は、次式で示すように、ファイバーの弾塑性状態を考慮した弾塑性剛性と、さらに幾何剛性と大変位剛性によって構成される。

$$[K_T]=[K_L]+[K_G]+[K_N] \qquad \cdots\cdots(5.13)$$

最初に、線形増分剛性$[K_L]$は、以下のように求められている。

$$[K_L]=\begin{bmatrix}
\frac{\overline{EA}}{\ell} & 0 & 0 & 0 & \frac{\overline{EAZ}}{\ell} & -\frac{\overline{EAY}}{\ell} & -\frac{\overline{EA}}{\ell} & 0 & 0 & 0 & -\frac{\overline{EAZ}}{\ell} & \frac{\overline{EAY}}{\ell} \\
 & \frac{12\overline{EAY^2}}{\ell^3} & 0 & 0 & 0 & \frac{6\overline{EAY^2}}{\ell^2} & 0 & -\frac{12\overline{EAY^2}}{\ell^3} & 0 & 0 & 0 & \frac{6\overline{EAY^2}}{\ell^2} \\
 & & \frac{12\overline{EAZ^2}}{\ell^3} & 0 & -\frac{6\overline{EAZ^2}}{\ell^2} & 0 & 0 & 0 & -\frac{12\overline{EAZ^2}}{\ell^3} & 0 & -\frac{6\overline{EAZ^2}}{\ell^2} & 0 \\
 & & & \frac{GJ_x}{\ell} & 0 & 0 & 0 & 0 & 0 & -\frac{GJ_x}{\ell} & 0 & 0 \\
 & & & & \frac{4\overline{EAZ^2}}{\ell} & -\frac{\overline{EAYZ}}{\ell} & -\frac{\overline{EAZ}}{\ell} & 0 & \frac{6\overline{EAZ^2}}{\ell^2} & 0 & \frac{2\overline{EAZ^2}}{\ell} & \frac{\overline{EAYZ}}{\ell} \\
 & & & & & \frac{4\overline{EAY^2}}{\ell} & \frac{\overline{EAY}}{\ell} & -\frac{6\overline{EAY^2}}{\ell^2} & 0 & 0 & \frac{\overline{EAYZ}}{\ell} & \frac{2\overline{EAY^2}}{\ell} \\
 & & & & & & \frac{\overline{EA}}{\ell} & 0 & 0 & 0 & \frac{\overline{EAZ}}{\ell} & -\frac{\overline{EAY}}{\ell} \\
 & & & & & & & \frac{12\overline{EAY^2}}{\ell^3} & 0 & 0 & 0 & -\frac{6\overline{EAY^2}}{\ell^2} \\
 & & & & & & & & \frac{12\overline{EAZ^2}}{\ell^3} & 0 & \frac{6\overline{EAZ^2}}{\ell^2} & 0 \\
 & & sym & & & & & & & \frac{GJ_x}{\ell} & 0 & 0 \\
 & & & & & & & & & & \frac{4\overline{EAZ^2}}{\ell} & -\frac{\overline{EAYZ}}{\ell} \\
 & & & & & & & & & & & \frac{4\overline{EAY^2}}{\ell}
\end{bmatrix} \qquad \cdots\cdots(5.14)$$

ここで、剛性行列内の各係数は

$$\overline{EA} = \sum E_i A_i$$

$$\left.\begin{aligned}
\overline{EAY_1} &= \int_A EydA\int_0^\ell \frac{1}{\ell}dx = \int_A EydA = \sum E_i Y_i A_i \\
\overline{EAY_2} &= \int_A EydA\int_0^\ell \frac{2x}{\ell^2}dx = \int_A EydA = \sum E_i Y_i A_i
\end{aligned}\right\} = \overline{EAY}$$

$$\left.\begin{aligned}
\overline{EAZ_1} &= \int_A EzdA\int_0^\ell \frac{1}{\ell}dx = \int_A EzdA = \sum E_i Z_i A_i \\
\overline{EAZ_2} &= \int_A EzdA\int_0^\ell \frac{2x}{\ell^2}dx = \int_A EzdA = \sum E_i Z_i A_i
\end{aligned}\right\} = \overline{EAZ}$$

$$\left.\begin{aligned}
\overline{EAY_1^2} &= \int_A Ey^2dA\int_0^\ell \frac{1}{\ell}dx = \int_A Ey^2dA = \sum E_i Y_i^2 A_i \\
\overline{EAY_2^2} &= \int_A Ey^2dA\int_0^\ell \frac{2x}{\ell^2}dx = \int_A Ey^2dA = \sum E_i Y_i^2 A_i \\
\overline{EAY_3^2} &= \int_A Ey^2dA\int_0^\ell \frac{3x^2}{\ell^3}dx = \int_A Ey^2dA = \sum E_i Y_i^2 A_i
\end{aligned}\right\} = \overline{EAY^2}$$

$$\left.\begin{aligned}
\overline{EAZ_1^2} &= \int_A Ez^2dA\int_0^\ell \frac{1}{\ell}dx = \int_A Ez^2dA = \sum E_i Z_i^2 A_i \\
\overline{EAZ_2^2} &= \int_A Ez^2dA\int_0^\ell \frac{2x}{\ell^2}dx = \int_A Ez^2dA = \sum E_i Z_i^2 A_i \\
\overline{EAZ_3^2} &= \int_A Ez^2dA\int_0^\ell \frac{3x^2}{\ell^3}dx = \int_A Ez^2dA = \sum E_i Z_i^2 A_i
\end{aligned}\right\} = \overline{EAZ^2}$$

$$\overline{EAYZ} = \int_A\int_0^\ell \frac{Eyz}{\ell}dxdA = \int_A EyzdA\int_0^\ell \frac{1}{\ell}dx = \int_A EyzdA = \sum E_i Y_i Z_i A_i$$

……(5.15)

となり、断面内の積分は各ファイバーにおける積の和となる。ただし、上の行列の中で、断面極２次モーメントについては弾性部材の剛性を用いることになる。

さらに、幾何剛性と大変位剛性は、弾性部材の幾何剛性行列$[K_G]$と同一となり、また、$[K_N]$は、断面に関する積分を除いて、弾性部材のそれと一致する。したがって、ファイバーモデルの接線剛性は、線形剛性を除いて、幾何剛性と大変位剛性は弾性の剛性を用いれば良いことになる。

5.9.2 各モデルの階層構造

本節では、剛性$[K_L]$を求めるサブルーチンについて解説する。例として用いる部材モデルは、両端ファイバーエレメントを組み込んだ部材モデルとする。剛性$[K_L]$に関連するサブルーチンは、以下のようにコールされる。

```
      call Cal_lin_stiff_M11(Model_type,Member(i),Element(ie),
     *      ak_linear(1,1,i) ,E_model11, E_model_fiber,
     *      M_model11, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work)
      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)
      call Cal_nonlin_stiff_M11(N_analysis,
     *      Model_type,Member(i),Element(ie),
     *      ak ,ier,E_model11, E_model_fiber,
     *      M_model11 , M_model_fiber)
```

上記の３つのサブルーチンの内容を以下の示す。これらは、前章で既に説明したので、内容はほぼ理解できていることと思う。本節では、このサブルーチンの中身を検証することにする。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Cal_lin_stiff_M11
C       ―――――――――――――――――――――――――――――――――
C     ●   代表的な部材モデルの剛性(両端ファイバーモデル)
C     ―――――――――――――――――――――――――――――――――
      subroutine Cal_lin_stiff_M11(Model_type,Member,Element,ak,
     *      E_model11, E_model_fiber,
     *      M_model11, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work)
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 / E_model_fiber_s    / E_model_fiber
      record / M_model11_s        / M_model11
      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 E_model11(*),M_model11(*)
      dimension ak(12,12),akk(12,12)
      real*8, ALLOCATABLE :: c(:,:),ab(:,:),ba(:),alength(:)
      integer, ALLOCATABLE :: irest_Point(:,:),n_type(:)
C     ―――――――――――――――――――――――――――――――――
      iet = Member.n_model                    ! モデルタイプ番号
      n_div = Model_type.n_div_model(iet)     ! 部材分割数
      imm= Element.n_element                  ! 要素番号
      immm= Member.n_model_type               ! モデルタイプ別番号
      n_if = 6*(n_div-1)                      ! 内部自由度
```

```
c―――――――――――――――――――――――――――――――――★動的記憶領域の確保
      ALLOCATE (
     *   irest_Point(6,n_div+1),n_type(n_div),alength(n_div)
     *         )
c―――――――――――――――――――――――――――――――――★節点拘束表の作成
c                                                    未知数等をセット
      call set_model11_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,
     *         n_type,alength,
     *  Member.i_rigid_length,    ! i端剛域
     *  Member.j_rigid_length,    ! j端剛域
     *  Member.i_shear_G,         ! i 端せん断剛性
     *  Member.j_shear_G)         ! j 端せん断剛性
c―――――――――――――――――――――――――――――――――★動的記憶領域の確保
      ALLOCATE (
     *   c(0:iubw,n_if),ab(n_if,12),ba(n_if)
     *         )
c―――――――――――――――――――――――――――――――――★剛性行列のゼロクリア
      do i=1,12
      do j=1,12
      ak(j,i)=0.
      enddo
      enddo
      do i=1,n_if
      do j=0,iubw
      c(j,i)=0.
      enddo
      enddo
      do i=1,12
      do j=1,n_if
      ab(j,i)=0.
      enddo
      enddo
      do i=1,n_div
      Member.an_stress(i)=0.
      enddo
      do i=1,n_div
      Member.an_vv(i)=0.
      Member.an_ww(i)=0.
      enddo
c―――――――――――――――――――――――――――――――――★部材剛性行列の作成
      it = 0
      do i=1,n_div
      call Stiff_M11_I(i,it,n_type(i),akk,Member,alength,
     *       Model_type,Element,
     *       E_model11(imm), E_model_fiber,
     *       M_model11(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
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(Control,N_analysis,
     *      mem_x,Model_type,Member,Element,
     *      E_model11, E_model_fiber,
     *      M_model11, M_model_fiber,
     *      Bilinear_work,Trilinear_work,Concrete_work,vv,vp,f_p)
C     ------------------------------------------
      implicit real*8(A-H,O-Z)
      include "submain.h"
      include "submainx.h"
      record / member_s    / Member
      record / element_s   / Element
      record / n_model_s         / Model_type
      record / E_model11_s        / E_model11
      record / E_model_fiber_s    / E_model_fiber
      record / M_model11_s        / M_model11
      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 E_model11(*),M_model11(*)
      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(:)
      integer, ALLOCATABLE :: irest_Point(:,:),n_type(:)
C     ------------------------------------------
      iet = Member.n_model                    ! モデルタイプ番号
      n_div = Model_type.n_div_model(iet)     ! 部材分割数
      imm= Element.n_element                  ! 要素番号
      immm= Member.n_model_type               ! モデルタイプ別番号
      n_if = 6*(n_div-1)                      ! 内部自由度
c-------------------------------------------------
c
c    ★        部材の剛性行列の設定
c
c-------------------------------------------------
c-------------------------------★動的記憶領域の確保
      ALLOCATE (
     *   irest_Point(6,n_div+1),n_type(n_div),alength(n_div),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_model11_dat(irest_Point,n_if,n_div,iubw,
     *  Element,Member,
     *  n_type,alength,
     *  Member.i_rigid_length,    ! i端剛域
     *  Member.j_rigid_length,    ! j端剛域
     *  Member.i_shear_G,         ! i 端せん断剛性
     *  Member.j_shear_G)         ! j 端せん断剛性
c----------★動的記憶領域の確保
      ALLOCATE (
     *   c(0:iubw,n_if),ab(n_if,12),bav(n_if),f1(n_if)
     *         )
c----------★剛性行列のゼロクリア
      do i=1,12
      do j=1,12
      ak(j,i)=0.
      enddo
      enddo
      do i=1,n_if
      do j=0,iubw
      c(j,i)=0.
      enddo
      enddo
      do i=1,12
      do j=1,n_if
      ab(j,i)=0.
      enddo
      enddo
      do i=1,12
      f_p(i)=0.
      enddo
      do i=1,n_if
      f1(i)=0.
      enddo
      EAx=Element.A*Element.E
      it=0
      do i=1,n_div
      EA(i)=EAx
      if(n_type(i).eq.2) then
      it=it+1
      if(it.eq.1) then
      EA(i)=M_model11(immm).d_ra_1
      else
      EA(i)=M_model11(immm).d_ra_2
      endif
      endif
      enddo
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
c-----------------------------------------------------
c
c    ★        部材内部の変位と不釣合力の計算
c
c-----------------------------------------------------
c-------------------------★両端変位を剛域内部の変位に変換処理
      call Deal_Rigid_element_v(vv,Member.i_rigid_length,
     *                          Member.j_rigid_length)
c-------------------------★部材内部変位の計算(c 行列の分解計算)
      call Typical_member_v(c,ab,M_model11(immm).ff_ip(1),
     *                    n_if,n_if,iubw,iubw,ier,vv,bav)
c-------------------------★部材内部、両端節点力の計算
      it = 0
      do i=1,n_div
      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=1,n_if
      M_model11(immm).ff_ip(i)=f1(i)
      enddo
c-------------------------★部材両端節点力への縮合
      call Typical_member_f(c,ab,f1,f_p,n_if,n_if,iubw,iubw)     ! f1 はデータが変更される
c-------------------------★部材節点力の両端剛域処理
      call Deal_Rigid_element_f(f_p,Member.i_rigid_length,
     *                           Member.j_rigid_length)
c-----------------------------------------------------
c
c    ★        部材の弾塑性チェック
c
c-----------------------------------------------------
c-------------------------------★部材内部応力のチェック
      it = 0
      do i=1,n_div
c-------------------------★変位の取りだし
      ii=0
      do j=1,2
      do k=1,6
      ii=ii+1
      irest=irest_Point(k,i+j-1)
      if(irest.lt.0) then
```

```
      vvx(ii)=vv(-irest)
      elseif(irest.gt.0) then
      vvx(ii)=bav(irest)
      else
      vvx(ii)=0.
      endif
      enddo
      enddo
c-----------------------------★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
c-------------------------------★弾性部材の軸力計算（幾何剛性作成用）
      call nonlinear_stress_N(akk(1,1,i),vvx,fnn)
      Member.an_stress(i)=Member.an_stress(i)+fnn
c-------------------------★部材内部変位を足しこむ
      Member.an_vv(i)=Member.an_vv(i)+(vvx(8) - vvx(2))
      Member.an_ww(i)=Member.an_ww(i)+(vvx(9) - vvx(3))
      enddo
c---------------------------------★動的記憶領域の解放
      DEALLOCATE (
     *   c ,ab ,irest_point,n_type,alength,EA,
     *   bav,akk,f1    )
      return
      end
C     ------------------------------------------
C     ●   SUBROUTINE /Cal_nonlin_stiff_M11
C       ----------------------------------------
C     ●   代表的な部材モデルの接線剛性(両端ファイバーモデル)
C     ------------------------------------------
      subroutine Cal_nonlin_stiff_M11(N_analysis,
     *            Model_type,Member,Element, ak,ier,
     *     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 / E_model_fiber_s   / E_model_fiber
      record / M_model11_s       / M_model11
      record / M_model_fiber_s   / M_model_fiber
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension E_model11(*),M_model11(*)
      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
      n_div = Model_type.n_div_model(iet)     ! 部材分割数
      imm= Element.n_element                  ! 要素番号
      immm= Member.n_model_type               ! モデルタイプ別番号
      n_if = 6*(n_div-1)                      ! 内部自由度
c-----------------------------------★動的記憶領域の確保
      ALLOCATE (
     *   irest_Point(6,n_div+1),n_type(n_div),alength(n_div),EA(n_div)
     *         )
c-----------------------------------★節点拘束表の作成
c                                              未知数等をセット
      call set_model11_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,
     *         n_type,alength,
     *  Member.i_rigid_length,    ! i 端剛域
     *  Member.j_rigid_length,    ! j 端剛域
     *  Member.i_shear_G,         ! i 端せん断剛性
     *  Member.j_shear_G)         ! j 端せん断剛性
c-----------------------------------★動的記憶領域の確保
      ALLOCATE (
     *   c(0:iubw,n_if),ab(n_if,12),ba(n_if )
     *         )
c-----------------------------------★剛性行列のゼロクリア
      do i=1,12
      do j=1,12
      ak(j,i)=0.
      enddo
      enddo
      do i=1,n_if
      do j=0,iubw
      c(j,i)=0.
      enddo
      enddo
      do i=1,12
      do j=1,n_if
      ab(j,i)=0.
      enddo
      enddo
      EAx=Element.A*Element.E
      it=0
      do i=1,n_div
      EA(i)=EAx
      if(n_type(i).eq.2) then
      it=it+1
      if(it.eq.1) then
      EA(i)=M_model11(immm).d_ra_1
      else
      EA(i)=M_model11(immm).d_ra_2
      endif
      endif
      enddo
c-----------------------------------★部材剛性行列の作成
      it = 0
      do i=1,n_div
```

```
      call Stiff_M11(i,it,n_type(i),akk,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,alength(i))
      call Create_Kn(akk,Member.an_vv(i),Member.an_ww(i),
     *              EA(i),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
```

5.9.3 静的縮合と剛域処理

前節で示した３つのサブルーチンについては、既に第 4.2 節で説明した。全体の計算流れはおおよそ理解できていると思う。ここでは、これら 3 つの中で同じようにコールされるサブルーチンについて解説する。まず、静的縮合を行う汎用の計算ルーチンを以下に示す。

```
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――――――――――――――――――――――――――――――――★部材内部変位の計算(c 行列の分解計算)
      call Typical_member_v(c,ab,M_model11(immm).ff_ip(1),
     *                      n_if,n_if,iubw,iubw,ier,vv,bav)
c――――――――――――――――――――――――――――――――★部材内部、両端節点力の計算
      call Typical_member_p_force(akk(1,1,i),irest_Point(1,i),
     *                            vv,bav ,f_p,f1)
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)
```

これらのサブルーチンは、第 5.3 節で説明した。内容については、その部分を参照されたい。ここで示したサブルーチンは汎用ルーチンになっているため、他のモデルでも使用している。それについては付録を参照されたい。

次は部材の両端もしくはどちらかに剛域がある場合で、剛域処理を行うサブルーチンコールについてである。

```
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――――――――――――――――――――――――――――――――――★両端の剛域処理
      call Deal_Rigid_element(ak,Member.i_rigid_length,
     *                          Member.j_rigid_length)
```

これらも既に第 5.6 節で説明した。そちらを参照されたい。次に、線形方程式の解析に関連するプログラムを以下にまとめる。

```
      call Bnd_FEM(i,akk,irest_Point,ak,c,ab,iubw,n_if)
      call Decom_B( c, n, nux,nu,eps, ba , ier )
      call Solv_B( c, n, nux, nu, ba  )
```

これらのサブルーチンは、部材行列を分割全体行列に分配するルーチン、その中のバンド行列 c の LU 分解を行うルーチン、最後に方程式を解くルーチンである。これらについては数値計算に関する文献[3)]を参照されたい。

5.9.4 部材モデルの設定

本節では、部材モデルを作成するサブルーチンを以下に示す。このサブルーチン set_model11_dat()は、部材モデルが静的縮合を行うために必要となる情報を作り出す。

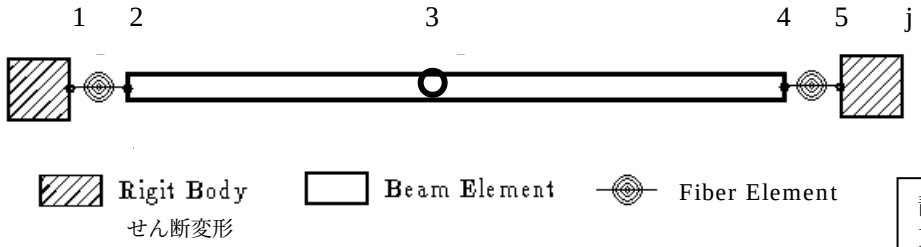

図 5-3 両端ファイバーモデルの部材解析モデル

このモデルは直線であるため、部材内部の釣合式を作成するとき、剛性行列の座標変換を行わない。このサブルーチンでは、節点における拘束表、あるいは未知数表を作成する。節点自由度に関する拘束のデータ仕様は、スペースフレームモデルの節点境界における拘束仕様と多少異なる。部材両端の節点自由度は負の値とし、ゼロは拘束とする。また、内部節点は自由は１で、拘束は 0 となる。

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。

図 5.3 は両端ファイバーの部材モデルを表しているが、このモデルでは、両端剛域の有無、さらに両端接合部にせん断変形を許すモデルに変更することができる。この処理を行うのがこのサブルーチンであり、少し長いが以下にその内容を示す。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /set_model11_dat
C       ―――――――――――――――――――――――――――――――――
C     ●   部材モデルの拘束表作成(ok)  ４部材  ５節点
C     ―――――――――――――――――――――――――――――――――
      subroutine set_model11_dat(irest_Point,n_if,n_div,iubw,
     *         Element,Member,
     *         n_type,alength,gxi,gxj,asi,asj)
C     ―――――――――――――――――――――――――――――――――
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / element_s    / Element
      record / member_s     / Member
      dimension  irest(6,5,3),nn_type(4),aleng(4),irest_s(12,3)
      dimension irest_Point(6,*),n_type(*),alength(*)
      real*8  gxi,gxj                 ! 剛域
      real*8  asi,asj                 ! 材端せん断変形用面積
      data nn_type/2,1,1,2/           ! 部材型  2:ファイバー  1:梁柱            ! 1
      data irest/-1,-2,-3,-4,-5,-6,  1,1,1,1,1,1,                               ! 2
     *           1,1,1,1,1,1,
     *           1,1,1,1,1,1,      -7,-8,-9,-10,-11,-12,
     *          -1,0,-3,0,-5,0,    1,0,1,0,1,0,
     *           1,0,1,0,1,0,
     *           1,0,1,0,1,0,      -7,0,-9,-0,-11,0,
     *           -1,-2,0,0,0,-6,   1,1,0,0,0,1,
     *           1,1,0,0,0,1,
     *           1,1,0,0,0,1,      -7,-8,0,0,0,-12/
      data irest_s/0,-2,-3,0,0,0,   0,-8,-9,0,0,0,
     *            0, 0,-3,0,0,0,   0, 0,-9,0,0,0,
     *            0,-2, 0,0,0,0,   0,-8, 0,0,0,0/
      data aleng/0.03,0.47,0.47,0.03/                                           ! 3
c―――――――――――――――――――――――――――――――――――――★モデルチェック
      itype=1           !  両端せん断変形なし
      if(asi.gt.0.) then                                                        ! 4
      if(asj.gt.0.) then
      itype=4         !  両端せん断変形考慮
      else
      itype=2         !  i 端せん断変形考慮
      endif
      elseif(asj.gt.0.) then
      itype = 3       !  j 端せん断変形考慮
      endif
      goto(1001,1002,1003,1004),itype                                           ! 5
c―――――――――――――――――――――――――――――――――――――★両端せん断変形なし
 1001 continue
      n_div=n_div-2                                                             ! 6
      do i=1,n_div                                                              ! 7
```

```
      n_type(i)=nn_type(i)
      enddo
      al=Member.alength - gxi - gxj       ! 剛域を削除                    ! 8
      do i=1,n_div                                                        ! 9
      alength(i)=al*aleng(i)
      enddo
      nn=n_div+1                                                          ! 10
      do i=1,nn                                                           ! 11
      do j=1,6
      irest_Point(j,i)=irest(j,i,1)
      enddo
      enddo
c―――――――――――――――――――――――――★ねじり剛性ゼロの場合
      if(Element.RIx .eq.0.) then                                         ! 12
      do i=2,nn-1
      irest_Point(4,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――★ｙ軸断面二次モーメントゼロの場合
      if(Element.RIy .eq.0.) then                                         ! 13
      do i=2,nn-1
      irest_Point(3,i) = 0
      irest_Point(5,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――★ｚ軸断面二次モーメント
      if(Element.RIz .eq.0.) then                                         ! 14
      do i=2,nn-1
      irest_Point(2,i) = 0
      irest_Point(6,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――★ 内部変位の節点番号セット
      n_if =0                                                             ! 15
      do i=2,nn-1                                                         ! 16
      do j=1,6
      if(irest_Point(j,i).gt.0) then
      if(irest_Point(j,i).eq.1) then
      n_if = n_if +1                                                      ! 17
      irest_Point(j,i)= n_if
      else
      ip=irest_Point(j,i)/10                                              ! 18
      ipp= Mod(irest_Point(j,i),10)
      irest_Point(j,i)=irest_Point(ipp,ip)
      endif
      endif
      enddo
      enddo
      call set_iubw(irest_Point,n_div,iubw)                               ! 19
      return
c―――――――――――――――――――――――――――――――――★i 端せん断変形考慮
 1002 continue                                                            ! 20
      n_div=n_div-1                                                       ! 21
      n_type(1)=3                         ! せん断バネ型                  ! 22
```

```
      do i=2,n_div                                                    ! 23
      n_type(i)=nn_type(i-1)
      enddo
      al=Member.alength - gxi - gxj      ! 剛域を削除
      alength(1)=gxi
      do i=2,n_div
      alength(i)=al*aleng(i-1)
      enddo
      nn=n_div+1
      do j=1,6
      irest_Point(j,1)=irest_s(j,1)
      enddo
      do i=2,nn
      do j=1,6
      irest_Point(j,i)=irest(j,i-1, 1)
      enddo
      enddo
      irest_Point(2,2)=1               ! 内部節点に変更
      if(Member.analysis_3D.eq.1)     irest_Point(2,2)=0  ! 平面問題に対応(x-z)
      irest_Point(3,2)=1               ! 内部節点に変更
      if(Member.analysis_3D.eq.2)     irest_Point(3,2)=0  ! 平面問題に対応(y-z)
c――――――――――――――――――――――――――★ねじり剛性ゼロの場合
      if(Element.RIx .eq.0.) then
      do i=3,nn-1
      irest_Point(4,i) = 0
      enddo
      endif
c――――――――――――――――――――――――――★ｙ軸断面二次モーメントゼロの場合
      if(Element.RIy .eq.0.) then
      do i=3,nn-1
      irest_Point(3,i) = 0
      irest_Point(5,i) = 0
      enddo
      endif
c――――――――――――――――――――――――――★ｚ軸断面二次モーメント
      if(Element.RIz .eq.0.) then
      do i=3,nn-1
      irest_Point(2,i) = 0
      irest_Point(6,i) = 0
      enddo
      endif
c――――――――――――――――――――――――――★内部変位の節点番号セット
      n_if =0
      do i=2,nn-1
      do j=1,6
      if(irest_Point(j,i).gt.0) then
      if(irest_Point(j,i).eq.1) then
      n_if = n_if +1
      irest_Point(j,i)= n_if
      else
      ip=irest_Point(j,i)/10
      ipp= Mod(irest_Point(j,i),10)
      irest_Point(j,i)=irest_Point(ipp,ip)
      endif
```

```
      endif
      enddo
      enddo
      call set_iubw(irest_Point,n_div,iubw)
      return
c―――――――――――――――――――――――――――――――★ j 端せん断変形考慮
 1003 continue                                                          ! 24

      n_div=n_div-1
      do i=1,n_div-1
      n_type(i)=nn_type(i)
      enddo
      n_type(n_div)=3                  ! せん断バネ型                  ! 25
      al=Member.alength - gxi - gxj    ! 剛域を削除
      do i=1,n_div-1
      alength(i)=al*aleng(i)
      enddo
      alength(n_div)=gxj
      nn=n_div+1
      do i=1,nn-1
      do j=1,6
      irest_Point(j,i)=irest(j,i, 1)
      enddo
      enddo
      do j=1,6
      irest_Point(j,nn)=irest_s(j+6,Member.analysis_3D+1)
      irest_Point(j,nn)=irest_s(j+6, 1)
      enddo
      irest_Point(2,nn-1)=1             ! 内部節点に変更
      if(Member.analysis_3D.eq.1)    irest_Point(2,nn-1)=0   ! 平面問題に対応(x-z)
      irest_Point(3,nn-1)=1             ! 内部節点に変更
      if(Member.analysis_3D.eq.2)    irest_Point(3,nn-1)=0   ! 平面問題に対応(y-z)
c―――――――――――――――――――――――――――――――★ねじり剛性ゼロの場合
      if(Element.RIx .eq.0.) then
      do i=2,nn-2
      irest_Point(4,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――――――――★ｙ軸断面二次モーメントゼロの場合
      if(Element.RIy .eq.0.) then
      do i=2,nn-2
      irest_Point(3,i) = 0
      irest_Point(5,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――――――――★ｚ軸断面二次モーメント
      if(Element.RIz .eq.0.) then
      do i=2,nn-2
      irest_Point(2,i) = 0
      irest_Point(6,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――――――――★内部変位の節点番号セット
      n_if =0
```

```
      do i=2,nn-1
      do j=1,6
      if(irest_Point(j,i).gt.0) then
      if(irest_Point(j,i).eq.1) then
      n_if = n_if +1
      irest_Point(j,i)= n_if
      else
      ip=irest_Point(j,i)/10
      ipp= Mod(irest_Point(j,i),10)
      irest_Point(j,i)=irest_Point(ipp,ip)
      endif
      endif
      enddo
      enddo
      call set_iubw(irest_Point,n_div,iubw)
      return
c―――――――――――――――――――――――――――――――★両端せん断変形考慮
 1004 continue                                                          ! 26
      do i=2,n_div-1
      n_type(i)=nn_type(i-1)
      enddo
      n_type(1)    =3                  ! せん断バネ型                  ! 27
      n_type(n_div)=3                  ! せん断バネ型
      al=Member.alength - gxi - gxj    ! 剛域を削除
      do i=2,n_div-1
      alength(i)=al*aleng(i-1)
      enddo
      alength(1)    =gxi
      alength(n_div)=gxj
      nn=n_div+1
      do i=2,nn-1
      do j=1,6
      irest_Point(j,i)=irest(j,i-1, 1)
      enddo
      enddo
      do j=1,6
      irest_Point(j,1) =irest_s(j, 1)
      irest_Point(j,nn)=irest_s(j+6, 1)
      enddo
      irest_Point(2,2)   =1            ! 内部節点に変更
      if(Member.analysis_3D.eq.1)    irest_Point(2,2)=0  ! 平面問題に対応(x-z)
      irest_Point(3,2)   =1            ! 内部節点に変更
      if(Member.analysis_3D.eq.2)    irest_Point(3,2)=0  ! 平面問題に対応(y-z)
      irest_Point(2,nn-1)=1            ! 内部節点に変更
      if(Member.analysis_3D.eq.1)    irest_Point(2,nn-1)=0   ! 平面問題に対応(x-z)
      irest_Point(3,nn-1)=1            ! 内部節点に変更
      if(Member.analysis_3D.eq.2)    irest_Point(3,nn-1)=0   ! 平面問題に対応(y-z)
c―――――――――――――――――――――――――――――――★ねじり剛性ゼロの場合
      if(Element.RIx .eq.0.) then
      do i=2,nn-1
      irest_Point(4,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――――――――★ｙ軸断面二次モーメントゼロの場合
```

```
      if(Element.RIy .eq.0.) then
      do i=3,nn-2
      irest_Point(3,i) = 0
      irest_Point(5,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――★ｚ軸断面二次モーメント
      if(Element.RIz .eq.0.) then
      do i=3,nn-2
      irest_Point(2,i) = 0
      irest_Point(6,i) = 0
      enddo
      endif
c―――――――――――――――――――――――――★内部変位の節点番号セット
      n_if =0
      do i=2,nn-1
      do j=1,6
      if(irest_Point(j,i).gt.0) then
      if(irest_Point(j,i).eq.1) then
      n_if = n_if +1
      irest_Point(j,i)= n_if
      else
      ip=irest_Point(j,i)/10
      ipp= Mod(irest_Point(j,i),10)
      irest_Point(j,i)=irest_Point(ipp,ip)
      endif
      endif
      enddo
      enddo
      call set_iubw(irest_Point,n_div,iubw)
      return
      end
C     ―――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /set_iubw
C       ―――――――――――――――――――――――――――――――――
C     ●   半バンド幅の計算
C     ―――――――――――――――――――――――――――――――――――
      subroutine set_iubw(irest,n_div,iubw)
      implicit real*8(A-H,O-Z)
      dimension irest(6,*)
      iubw=0                                                    ! 28
      do 100 i=1,n_div                                          ! 29
      do j=1,6                                                  ! 30
      if(irest(j,i).gt.0) then                                  ! 31
      do k=1,6
      if(irest(k,i+1).gt.0) then
      iu=iabs(irest(j,i)-irest(k,i+1))                          ! 32
      if(iubw.lt.iu) iubw=iu
      endif
      enddo
      endif
      enddo
  100 continue
      do 101 i=1,n_div+1                                        ! 33
```

```
      do j=1,5
      if(irest(j,i).gt.0) then
      do k=j+1,6
      if(irest(k,i).gt.0) then
      iu=iabs(irest(j,i)-irest(k,i))                                    ! 34
      if(iubw.lt.iu) iubw=iu
      endif
      enddo
      endif
      enddo
  101 continue
      return
      end
```

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

1. 両端せん断変形領域なしの場合の代表的なエレメント番号を、data 文を用いて配列 nn_type にセットする。
2. 代表的な節点拘束表を、data 文を用いて配列 irest にセットする。
3. 各エレメントの長さの比率を data 文を用いて配列 aleng に設定する。
4. 両端もしくは片方にせん断変形領域が存在する場合を考慮して、次の 4 タイプに分類する。
   １．両端せん断変形領域なし
   ２．i 端せん断変形領域あり
   ３．j 端せん断変形領域あり
   ４．両端せん断変形領域あり
5. 上で分類したタイプ別に処理を分類する。これ以降はタイプ１の両端せん断変形領域なしの場合に関する処理を行う。
6. ここにでの処理では、部材の分割数（エレメント数）n_div を 2 つ減らして４とする。
7. 部材の各エレメントにエレメント型番号をセットする。
8. 剛域長さを取り除いた実際の部材長さを計算する。
9. 各エレメントの比率 aleng(i)（データ文で比率をセット）を部材長さ al に掛けて、エレメントの長さをセットする。
10. エレメント数に１を足して、節点数 nn にセットする。
11. 節点数及び節点自由度数分、data 文でセットした拘束表 irest を節点拘束表 irest_Point にコピーする。
12. この部材のねじり剛性がゼロの場合、内部節点のねじり剛性項に関連する自由度を拘束する。もしこの拘束を行わないと、ねじり剛性がないため、該当する内部節点自由度の位置で、剛性行列の LDU 分

エレメント型番号は以下のようである。
１：弾性
２：ファイバー
３：せん断

解時に、エラーが発生する。

13. ｙ軸の断面二次モーメントがゼロの場合、上記と同じ理由で該当する内部節点自由度を拘束する。
14. 同様に、ｚ軸の断面二次モーメントがゼロの場合、該当する内部節点自由度を拘束する。
15. 以降では、内部節点自由度に関連する拘束表から未知番号表を作成する。ここでは、まず未知番号 n_if に 0 をセットする。
16. 内部節点及び 6 自由度について、以下の処理を行う。
17. 拘束番号 n_if が 0 以上で、しかも 1 に等しい場合、その自由度は拘束なしであるため、未知番号を 1 増やして、その値を拘束表にセットする。
18. 拘束番号が 0 以上で、しかも 1 でない場合、その自由度は他の節点自由度と同じである。そのため、仕様にしたがって一桁目の値を自由度番号 ip、二桁目以上を節点番号とし、その節点番号と自由度番号の拘束値を拘束表にセットする。この作業が終了すると内部節点の全自由度 n_if が決定される。
19. サブルーチン set_iubw()を用いて半バンド幅 iubw を求める。これで両端せん断変形領域なしの処理が終了する。
20. ここからが、i 端せん断変形領域ありの場合の処理である。ここからの説明は、前記の説明以外の部分について行う。
21. ここでの処理は、部材の分割数（エレメント数）n_div を 1 つ減らして 5 とする。
22. 最初のエレメントのエレメント型番号をせん断ばね型：3 とする。
23. エレメント型番号をひとつずらしてセットする。後の処理は両端せん断変形領域なしと同じである。
24. ここからが j 端せん断変形領域ありの場合の処理である。
25. 最後のエレメントのエレメント型番号をせん断ばね型：3 とする。
26. ここからが両端せん断変形領域ありの場合の処理である。
27. 最初と最後のエレメントのエレメント型番号をせん断ばね型：3 とする。
28. このサブルーチン set_iubw()では、未知番号表を用いて半バンド幅を求める。まず、半バンド幅 iubw をゼロセットする。
29. エレメント数分、以下の処理を行う。ここではエレメントの両節点について調査する。
30. エレメントの左節点の自由度 6 について以下の処理を行う。
31. その自由度 i が拘束されていないことを確かめる。

32. 左の節点と右の節点の間で、未知番号の差が最大のものを半バンド幅 iubw にセットする。
33. 全節点数について、以下の処理を行う。
34. 次は、当該の節点内で、未知番号の差が最大のものを半バンド幅 iubw にセットする。

以上のように、このサブルーチンで静的縮合モデルの部材解析モデルとして、未知数番号表と全未知数、半バンド幅が決定される。このようなサブルーチンは他の静的縮合モデルにも存在し、同様な処理を行っている。詳細は付録を参照されたい。

5.9.5 ファイバーモデルの剛性

次に、両端ファイバーモデルの弾塑性剛性$[K_L]$を作成するサブルーチンを見てみよう。このサブルーチンは第２層の階層構造を有している。サブルーチン Stiff_M11()を以下に示す。

```
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
```

このサブルーチンは、剛性行列を作成するプログラムであり、その構造は非常に単純で、エレメント型番号 n_type によって、階層構造となっている。最初に、剛性行列 ak をゼロクリアする。次に、エレメント型番号によって処理が分類されており、現在は、1 は弾性エレメント、2 はファイバーエレメント、3 はせん断ばねエレメントとなっている。各エレメント型番号にしたがってサブルーチンがコールされ、剛性行列 ak が計算される。弾性エレメントの剛性を計算するサブルーチン Cal_nonlin_stiff_Mx()は、表示は非線形となっているが、実際は第 5.2 節で説明した線形の弾性剛性を求めている。

エレメント型番号 2 のファイバーにおけるサブルーチンは、ファイバー断面を有するエレメントの弾塑性剛性行列を求める処理を行う。この

弾塑性剛性行列に関連する３つのサブルーチン Fiber_Model_GI11()と Fiber_Model_G11()及び Fiber_GK()を以下に示し、説明する。最初のサブルーチンは初期設定であり、各種のフイバーデータを構造体に設定する。次のサブルーチンは、弾塑性行列を求めるため、各種のデータを構造体から取得するルーチンであり、最後のサブルーチンは実際に剛性行列を計算するプログラムである。以下に３つのサブルーチンの内容を具体的に示すことにする。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /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)
      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
      dimension E_model_fiber(*),M_model_fiber(*)
      dimension Bilinear_work(*),Trilinear_work(*)
      dimension Concrete_work(*)
      dimension ak(12,12)
c―――――――――――――――――――――――――――――――★
      if(it.eq.1) then                                          ! 1
      do i=1,30                                                 ! 2
      M_model.ff_ip(i) = 0.
      enddo
      nm_div = E_model.n_section_1                              ! 3
      nn     = E_model.nm_section_1 - 1
      nnm    = M_model.nm_section_1 - 1
      elseif(it.eq.2) then                                      ! 4
      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                                             ! 5
      nn     =  nn  + 1                                         ! 6
      nnm    =  nnm + 1
c―――――――――――――――――――――――――――――――★データの初期設定
```

```
      M_model_fiber(nnm).d_E       = E_model_fiber(nn).E_1                    ! 7
      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       ! 8
      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                               ! 9
  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    ! 10
      goto 100
  20  continue
c―――――――――――――――――――――――――★対称トリリニア型                ! 11
      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                ! 12
      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
      if(it.eq.1) then                                                      ! 13
      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.                                                           ! 14
      ray    = 0.
      raz    = 0.
      raz2   = 0.
      ray2   = 0.
      rayz   = 0.
      gg     = 0.
      aa     = 0.
      do i=1,nm_div                                                         ! 15
      nn     = nn+1
      nnm    = nnm+1
      A      = E_model_fiber(nn).A
      E      = M_model_fiber(nnm).d_E
      ra     = ra   + E*A                                                   ! 16
```

```
      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                                                  ! 17
      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,                ! 18
     *     E_model,E_model_fiber,M_model,M_model_fiber)
      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     ------------------------------------------------
C     ●   Model_No.11 両端ファイバーモデル E_model11_s 構造体
C     ------------------------------------------------
c     部材
c      structure / E_model11_s/
```

```
c      integer  nm_section_1        !  i 端断面のファイバー開始番号
c      integer  n_section_1         !  i 端断面のファイバー数
c      integer  nm_section_2        !  j 端断面のファイバー開始番号
c      integer  n_section_2         !  j 端断面のファイバー数
c      real*8   i_rigid_length      !  i 端剛域長さ
c      real*8   j_rigid_length      !  j 端剛域長さ
c      real*8   i_shear_G           !  i 端せん断剛性
c      real*8   j_shear_G           !  j 端せん断剛性
c      end structure
c      record / E_model11_s    / E_model11
c
C     ―――――――――――――――――――――――――――――――――
C     ●　ファイバーモデル E_model_fiber_s 構造体
C     ―――――――――――――――――――――――――――――――――
c     部材
c      structure / E_model_fiber_s/
c      integer  nm_type              !  履歴モデルの番号
c      real*8   E_1                  !  ファイバーの第一剛性 E1
c      real*8   E_2                  !  ファイバーの第二剛性 E2
c      real*8   Q_1                  !  ファイバーの第一折れ点
c      real*8   G                    !  ファイバー断面積 G
c      real*8   A                    !  ファイバー断面積
c      real*8   Ay                   !  ファイバーｙ軸せん断用断面積
c      real*8   Az                   !  ファイバーｚ軸せん断用断面積
c      real*8   ry                   !  中立軸から断面中心までのｙ方向距離
c      real*8   rz                   !  中立軸から断面中心までのｚ方向距離
c      real*8   Ary                  !  断面積掛ける距離
c      real*8   Arz                  !  断面積掛ける距離
c      real*8   Ary2                 !  断面積掛ける距離の２乗
c      real*8   Arz2                 !  断面積掛ける距離の２乗
c      real*8   Aryz                 !
c      end structure
C     ―――――――――――――――――――――――――――――――――
C     ●　Model_No.11 両端ファイバーモデル M_model11_s 構造体
C     ―――――――――――――――――――――――――――――――――
c     部材
c      structure / M_model11_s/
c      integer  nm_section_1        !  i 端断面のファイバー開始番号
c      integer  n_section_1         !  i 端断面のファイバー数
c      real*8  d_state_1            !  ファイバー要素分割断面の弾塑性状態
c      real*8  d_aa_1               !  断面積の和
c      real*8  d_ra_1               !  E*断面積の和
c      real*8  d_ray_1              !  E*A*z
c      real*8  d_raz_1              !  E*A*y
c      real*8  d_raz2_1             !  E*A*y*y
c      real*8  d_ray2_1             !  E*A*z*z
c      real*8  d_rayz_1             !  E*A*z*y
c      real*8  d_gg_1               !  G*A
c      real*8  d_epsi_x_1           !  軸方向歪
c      real*8  d_epsi_y_1           !  ｙ軸に関する曲げ歪
c      real*8  d_epsi_z_1           !  ｚ軸に関する曲げ歪
c      integer  nm_section_2        !  j 端断面のファイバー開始番号
c      integer  n_section_2         !  j 端断面のファイバー数
c      real*8  d_state_2            !  ファイバー要素分割断面の弾塑性状態
```

```
c      real*8  d_aa_2              !  断面積の和
c      real*8  d_ra_2              !  E*断面積の和
c      real*8  d_ray_2             !  E*A*z
c      real*8  d_raz_2             !  E*A*y
c      real*8  d_raz2_2            !  E*A*y*y
c      real*8  d_ray2_2            !  E*A*z*z
c      real*8  d_rayz_2            !  E*A*z*y
c      real*8  d_gg_2              !  G*A
c      real*8  d_epsi_x_2          !  軸方向歪
c      real*8  d_epsi_y_2          !  y 軸に関する曲げ歪
c      real*8  d_epsi_z_2          !  z 軸に関する曲げ歪
c      real*8  disp_p(12)          !  両端の変位
c      real*8  disp_ip(18)         !  内部節点の変位
c      end structure
C     ―――――――――――――――――――――――――――――――――――
C     ●   Model_No.11 両端ファイバーモデル M_model_fiber_s 構造体
C     ―――――――――――――――――――――――――――――――――――
c     部材
c      structure / M_model_fiber_s/
c      integer  n_type             !  履歴モデルの通し番号
c      integer  i_submodel         !  ファイバー要素の状態（弾性の場合は０：塑性は１）
c      real*8   d_sigma_x          !  軸方向歪
c      real*8   d_stress_x         !  軸方向応力
c      real*8   d_E                !  接線剛性
c      end structure
c
      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）      19
      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,                          ! 20
     *              raz2,ray2,rayz,gg,aa,rix,asy,asz)
      return
      end
C     ―――――――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Fiber_GK
C     ―――――――――――――――――――――――――――――――――――――
C     ●   ファイバー要素の剛性行列
C     ―――――――――――――――――――――――――――――――――――――
      subroutine Fiber_GK(ak,RL,ra,ray,raz,riz2,riy2,riyz,
     *                    G,aa,rix,asy,asz)
      implicit real*8(A-H,O-Z)
      dimension ak(12,12)
      ak(1,1) = ra/RL                                               ! 21
      ak(1,7) =-ak(1,1)
      ak(7,7) = ak(1,1)
      ak(1,5) = ray/RL
      ak(1,11)=-ak(1,5)
      ak(5,7) =-ak(1,5)
      ak(7,11)= ak(1,5)
      ak(1,12)= raz/RL
      ak(1,6) =-ak(1,12)
      ak(6,7) = ak(1,12)
      ak(7,12)=-ak(1,12)
      ak(5,12)= riyz/RL
      ak(5,6) =-ak(5,12)
      ak(6,11)=  ak(5,12)
      ak(11,12)=-ak(5,12)
      RL2=RL*RL
      RL3=RL*RL*RL
      IF(asy.NE.0.0.OR.asz.NE.0.0) GOTO 20                          ! 22
      ZERL2   = 6.*riz2/RL2
      ZERL3   = 2.*ZERL2/RL
      ak(2,2) = ZERL3
      ak(2,6) = ZERL2
      ak(2,8) =-ZERL3
      ak(2,12)= ZERL2
      YERL2   = 6.*riy2/RL2
      YERL3   = 2.*YERL2/RL
      ak(3,3) = YERL3
      ak(3,5) =-YERL2
      ak(3,9) =-YERL3
      ak(3,11)=-YERL2
      ak(4,4) = G*RIX/RL
      ak(4,10)=-ak(4,4)
      ak(5,5) = 4.0*riy2/RL
      ak(5,9) = YERL2
      ak(5,11)= ak(5,5)*0.5
      ak(6,6) = 4.0*riz2/RL
      ak(6,8) =-ZERL2
```

```
      ak(6,12)= ak(6,6)*0.5
      ak(8,8)=  ZERL3
      ak(8,12)=-ZERL2
      ak(9,9)=  YERL3
      ak(9,11)= YERL2
      ak(10,10)= ak(4,4)
      ak(11,11)= ak(5,5)
      ak(12,12)= ak(6,6)
      do  i=1,11
      do  j=i+1,12
      ak(j,i) =ak(i,j)
      enddo
      enddo
      return
C     ----------------------------------------------------------------
C     ---  せん断変形を考慮  -----------------------------------------
C     ----------------------------------------------------------------
   20 CONTINUE                                                        ! 23
      FY = 0.0
      FZ = 0.0
      IF(ASY.NE.0.0) FY=12.*riz2/(G*ASY*RL2)
      IF(ASZ.NE.0.0) FZ=12.*riy2/(G*ASZ*RL2)
      ZERL3   = 12.*riz2/(RL3*(1.+FY))
      ZERL2   = 6.*riz2/(RL2*(1.+FY))
      ak(2,2) = ZERL3
      ak(2,6) = ZERL2
      ak(2,8) =-ZERL3
      ak(2,12)= ZERL2
      YERL3   = 12.*riy2/(RL3*(1.+FZ))
      YERL2   = 6.*riy2/(RL2*(1.+FZ))
      ak(3,3) = YERL3
      ak(3,5) =-YERL2
      ak(3,9) =-YERL3
      ak(3,11)=-YERL2
      ak(4,4) = G*RIX/RL
      ak(4,10)=-ak(4,4)
      ak(5,5) = (4.0+FZ)*riy2/(RL*(1.+FZ))
      ak(5,9) = YERL2
      ak(5,11)= (2.0-FZ)*riy2/(RL*(1.+FZ))
      ak(6,6) = (4.0+FY)*riz2/(RL*(1.+FY))
      ak(6,8) =-ZERL2
      ak(6,12)= (2.0-FY)*riz2/(RL*(1.+FY))
      ak(8,8) = ZERL3
      ak(8,12)=-ZERL2
      ak(9,9) = YERL3
      ak(9,11)= YERL2
      ak(10,10)= ak(4,4)
      ak(11,11)= ak(5,5)
      ak(12,12)= ak(6,6)
      DO  i=1,11                                                      ! 24
      DO  j=i+1,12
      ak(j,i) =ak(i,j)
      enddo
      enddo
```

```
return
end
```

上記の３つのサブルーチンについて、右側のコメント番号に従ってプログラムの内容を説明する。

1. 両端ファイバーモデルでは、i 端と j 端でファイバーデータの管理位置が異なる。そのため、ここでどちらかをチェックする。パラメータ it が１の場合は i 端であり、2 は j 端である。
2. 内部節点の不釣合力の保存場所である構造体内の成分である M_model.ff_ip()をゼロクリアする。
3. 構造体より i 端のファイバーデータの管理用パラメータを取得する。それらは、その断面におけるファイバー数 nm_div、ファイバーに関連するデータの最初の配列番号 - 1 を nn に、同じく部材に関連するデータの最初の配列番号 - 1 を nnm にセットする。
4. 上記と同様に、j 端に関するファイバーデータの管理用パラメータを取得する。
5. 以下の処理を、その断面の全ファイバー数分行う。
6. 上記の nn 及び nnm に１を加え、構造体への配列番号をセットする。
7. 入力データ以外の各ファイバーデータの初期設定を行う。まず、部材に関連し、解析途中で変更するデータとして、ファイバーのヤング係数、ファイバーの弾塑性状態を表すパラメータ、軸方向ひずみ、軸方向応力をゼロ、もしくは値をセットする。
8. 次に、ファイバーデータの断面図芯位置における断面特性を計算し、要素に関連する構造体 E_model にセットする。この値は、解析途中で変更しない。
9. ファイバーの履歴特性番号によって処理内容が分類されており、その番号にしたがって各プログラム位置に処理が移動する。ただし、ここでは初期設定であるので、ファイバー履歴番号が 4 以上では、ワーク領域が同じであることより、１から３までの処理で行われる。
10. バイリニアデータの初期状態を-1 にセットする。
11. トリリニアデータの初期状態を-1 にセットする。
12. コンクリートデータの初期状態を-1 にセットする。
13. 次の処理を行うため、構造体より、i 端及び j 端のファイバーデータの管理用パラメータを取得する。
14. ファイバー断面の断面積や断面一次モーメント、断面二次モーメントを求めるために、ここでは各パラメータをゼロセットしている。

15. その断面における全ファイバーについて次の処理を行う。
16. 各ファイバーについて断面積や断面一次モーメントなどを計算し、和を取る処理を行う。
17. 断面全体で得られた断面積や断面一次モーメントなどを構造体にセットする。
18. ここで、この構造体でセットした値を用いて、この断面を要するエレメントの剛性行列をサブルーチン Fiber_Model_G11()で計算する。
19. サブルーチン Fiber_Model_G11()では、まず、この部材のどちらの端部かをチェックし、該当する端部断面に関する各パラメータを構造体より取得する。
20. この取得したパラメータとこのエレメントの長さを用いて、実際の剛性行列をサブルーチン Fiber_GK()を用いて計算する。
21. ここでは、ファイバーエレメントの剛性行列を計算する。ファイバー断面の剛性行列は、式(5.14)の行列と比較して確認されたい。また、この式の誘導などについては理論マニュアルを参照されたい。
22. せん断変形用の断面積がセットされている場合は、せん断変形を考慮するとして文番号 20 へ制御が移る。せん断変形を考慮しない場合は以降の処理を行う。
23. せん断変形を考慮する場合は、以降の処理を行う。
24. 対称行列であるため、下三角行列部分へ値をコピーする。

5.9.6 ファイバーモデルの弾塑性チェック

最後に残ったのがファイバーの弾塑性チェックである。この弾塑性チェックを行うサブルーチン Fiber_check()をコールするコードは、サブルーチン Cal_check_stiff_M11()の中で､以下のように記述されている。

```
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
```

サブルーチン Fiber_check()の内容を以下の示す。

```
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(Control,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"
      record /control_s    / Control
      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                                                  ! 1
      d_epsi_x_1 =  (vv(7)  - vv(1)) / Alength   !  軸方向歪             2
      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  3
      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                                    ! 4
      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  5
      do i=1,nm_div                                                     ! 6
      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     7
     *      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                ! 8
```

```
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
c―――――――――――――――――――――――――★弾塑性解析
      goto ( 10,20,30,40,50,60,70,80),nm_type                          ! 9
  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,   ! 10
     *              E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *              E_model_fiber(nn).Q_1,du,
     *              M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1)
      if(Bilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      if(Bilinear_work(nmx).i_stat.ne.0) then
      endif
      goto 100
  20  continue
c―――――――――――――――――――――――――★対称トリリニア型 （スチール用）
      call TriLinear(M_model_fiber(nnm).d_E,Trilinear_work(nmx).i_stat,
     *               E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *               E_model_fiber(nn).E_3,
     *               E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *               du,M_model_fiber(nnm).d_stress_x,
     *               Trilinear_work(nmx).P1(1))
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 100
  30  continue
c―――――――――――――――――――――――――★コンクリート型
      call Concrete(M_model_fiber(nnm).d_E,Concrete_work(nmx).i_stat,   ! 11
     *              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＋等方硬化用）
      call BiLinear_h(M_model_fiber(nnm).d_E,Bilinear_work(nmx).i_stat,
     *              E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *              E_model_fiber(nn).Q_1,du,
     *              M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1,
     *              E_model_fiber(nn).Beta)
      if(Bilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 100
   60 continue
c――――――――――――――――――――――――――★対称トリリニア型（移動＋等方硬化用）
      call TriLinear_h(M_model_fiber(nnm).d_E,
     *               Trilinear_work(nmx).i_stat,
     *               E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *               E_model_fiber(nn).E_3,
     *               E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *               du,M_model_fiber(nnm).d_stress_x,
     *              Trilinear_work(nmx).P1(1),Trilinear_work(nmx).P1(2),
     *              E_model_fiber(nn).Beta, E_model_fiber(nn).Beta_2)
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 100
   70 continue
c――――――――――――――――――――――――――★非対称バイリニア型
      call BiLinear_AS(M_model_fiber(nnm).d_E,Bilinear_work(nmx).i_stat,
     *              E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *              E_model_fiber(nn).Q_1,E_model_fiber(nn).Ec_1,
     *              E_model_fiber(nn).Ec_2,E_model_fiber(nn).Qc_1,du,
     *              M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1,
     *              Bilinear_work(nmx).P2,Bilinear_work(nmx).Sig_z,
     *              E_model_fiber(nn).Beta)
      if(Bilinear_work(nmx).i_stat.eq.2.or.
     *   Bilinear_work(nmx).i_stat.eq.3) iistat = iistat + 1
      goto 100
   80 continue
c――――――――――――――――――――――――――★非対称トリリニア型
      call TriLinear_AS(M_model_fiber(nnm).d_E,
     *               Trilinear_work(nmx).i_stat,
     *               E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *               E_model_fiber(nn).E_3,E_model_fiber(nn).Ec_1,
     *               E_model_fiber(nn).Ec_2,E_model_fiber(nn).Ec_3,
     *               E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *               E_model_fiber(nn).Qc_1,E_model_fiber(nn).Qc_2,
     *               du,M_model_fiber(nnm).d_stress_x,
     *              Trilinear_work(nmx).P1(1),Trilinear_work(nmx).P1(2),
     *              Trilinear_work(nmx).P1(3),
     *               E_model_fiber(nn).Beta, E_model_fiber(nn).Beta_2)
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 100
```

```
c――――――――――――――――――――――――――★弾塑性解析終了
      endif
  100 continue                                                          ! 12
      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
      d_state = float(iistat)/float(nm_div)   ! 塑性した面積の計算          ! 13
      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                                     ! 14
      nn      = E_model.nm_section_1 - 1
      nnm     = M_model.nm_section_1 - 1
      ra      = 0.                                                      ! 15
      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                                                     ! 16
      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_model.nm_section_1
      call jikuzero_control(Control,ra,E_model_fiber(nn).E_1,           ! 17
     *                      E_model_fiber(nn).A)
      M_model.d_aa_1 = aa               !  断面積の和                   ! 18
      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                                                  ! 19
      d_epsi_x_2 =  (vv(7)  - vv(1)) / Alength   !  軸方向歪
      d_epsi_y_2 =  (vv(11) - vv(5)) / Alength   !  y 軸に関する曲げ歪
      d_epsi_z_2 =  (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_2= d_epsi_x_2+strain_nonlinear(Member.an_vv(idiv),
     *      Member.an_ww(idiv),vv,Alength)
      endif
      if(Member.analysis_3D .eq. 1) d_epsi_z_2 =0.  ! 平面問題における面外方向の曲げ変形を無視する
      if(Member.analysis_3D .eq. 2) d_epsi_y_2 =0.  ! 平面問題における面外方向の曲げ変形を無視する
      M_model.d_epsi_x_2 = d_epsi_x_2 + M_model.d_epsi_x_2          !  軸方向歪
      M_model.d_epsi_y_2 = d_epsi_y_2 + M_model.d_epsi_y_2           !  y 軸に関する曲げ歪
      M_model.d_epsi_z_2 = d_epsi_z_2 + M_model.d_epsi_z_2          !  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_2
      nn          = E_model.nm_section_2 - 1
      nnm         = M_model.nm_section_2 - 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_2 +                           ! 軸方向歪
     *      d_epsi_y_2 * E_model_fiber(nn).rz -     ! y 軸に関する曲げ歪
     *      d_epsi_z_2 * 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
c―――――――――――――――――――――――★弾塑性解析
      goto ( 11,21,31,41,51,61,71,81),nm_type
  11  continue
c―――――――――――――――――――――――★バイリニア型（スチール用）
      call BiLinear(M_model_fiber(nnm).d_E,Bilinear_work(nmx).i_stat,
     *              E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *              E_model_fiber(nn).Q_1,du,
     *              M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1)
      if(Bilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      if(Bilinear_work(nmx).i_stat.ne.0) then
      endif
      goto 101
  21  continue
c―――――――――――――――――――――――★対称トリリニア型 （スチール用）
      call TriLinear(M_model_fiber(nnm).d_E,Trilinear_work(nmx).i_stat,
     *               E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *               E_model_fiber(nn).E_3,
```

```
     *              E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *              du,M_model_fiber(nnm).d_stress_x,
     *              Trilinear_work(nmx).P1(1))
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 101
   31 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 101
   41 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 101
   51 continue
c-----------------------★バイリニア型移動＋等方硬化用）
      call BiLinear_h(M_model_fiber(nnm).d_E,Bilinear_work(nmx).i_stat,
     *             E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *             E_model_fiber(nn).Q_1,du,
     *             M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1,
     *             E_model_fiber(nn).Beta)
      if(Bilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      if(Bilinear_work(nmx).i_stat.ne.0) then
      endif
      goto 101
   61 continue
c-----------------------★対称トリリニア型（移動＋等方硬化用）
      call TriLinear_h(M_model_fiber(nnm).d_E,
     *              Trilinear_work(nmx).i_stat,
```

```
     *               E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *               E_model_fiber(nn).E_3,
     *               E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *               du,M_model_fiber(nnm).d_stress_x,
     *              Trilinear_work(nmx).P1(1),Trilinear_work(nmx).P1(2),
     *              E_model_fiber(nn).Beta, E_model_fiber(nn).Beta_2)
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 101
  71  continue
c―――――――――――――――――――――――――★非対称バイリニア型
      call BiLinear_AS(M_model_fiber(nnm).d_E,Bilinear_work(nmx).i_stat,
     *              E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *              E_model_fiber(nn).Q_1,E_model_fiber(nn).Ec_1,
     *              E_model_fiber(nn).Ec_2,E_model_fiber(nn).Qc_1,du,
     *              M_model_fiber(nnm).d_stress_x,Bilinear_work(nmx).P1,
     *              Bilinear_work(nmx).P2,Bilinear_work(nmx).Sig_z,
     *              E_model_fiber(nn).Beta)
      if(Bilinear_work(nmx).i_stat.eq.2.or.
     *   Bilinear_work(nmx).i_stat.eq.3) iistat = iistat + 1
      goto 101
  81  continue
c―――――――――――――――――――――――――★非対称トリリニア型
      call TriLinear_AS(M_model_fiber(nnm).d_E,
     *               Trilinear_work(nmx).i_stat,
     *               E_model_fiber(nn).E_1,E_model_fiber(nn).E_2,
     *               E_model_fiber(nn).E_3,E_model_fiber(nn).Ec_1,
     *               E_model_fiber(nn).Ec_2,E_model_fiber(nn).Ec_3,
     *               E_model_fiber(nn).Q_1,E_model_fiber(nn).Q_2,
     *               E_model_fiber(nn).Qc_1,E_model_fiber(nn).Qc_2,
     *               du,M_model_fiber(nnm).d_stress_x,
     *              Trilinear_work(nmx).P1(1),Trilinear_work(nmx).P1(2),
     *              Trilinear_work(nmx).P1(3),
     *               E_model_fiber(nn).Beta, E_model_fiber(nn).Beta_2)
      if(Trilinear_work(nmx).i_stat.ne.0) iistat = iistat + 1
      goto 101
      endif
c―――――――――――――――――――――――――★弾塑性解析終了
  101 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
      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(2) = 0
      elseif(d_state .ge.0.8) then
      Member.d_stat(2) = 2
      else
      Member.d_stat(2) = 1
      endif
      nm_div           = E_model.n_section_2
      nn               = E_model.nm_section_2 - 1
      nnm              = M_model.nm_section_2 - 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_model.nm_section_2
      call jikuzero_control(Control,ra,E_model_fiber(nn).E_1
     *                     ,E_model_fiber(nn).A)
      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
      endif
      return
      end
```

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

1. 両端ファイバーモデルでは、i 端と j 端でファイバーデータの管理位置が異なる。そのため、ここでどちらかをチェックする。パラメータ it が１の場合は i 端であり、2 は j 端である。
2. この断面の増分ひずみである増分軸方向ひずみ、ｙ軸に関する曲げひずみ、ｚ軸に関する曲げひずみを求める。

3. 幾何学的非線形解析を行う場合は、非線形増分軸方向ひずみを足し込む。次に、解析が平面問題の場合、面外方向の曲げひずみをゼロとする。最後に各増分ひずみを増分前のひずみに足しこむ。
4. 断面内のファイバーの数、要素に関連するファイバー用データの配列番号 -1 を nn に、部材に関連するファイバー用データの配列番号 -1 を nnm に設定する。
5. 断面内塑性率を計算するパラメータをゼロセットする。
6. 断面内の全ファイバーについて以下の処理を行う。まず、配列番号 nn と nnm に 1 を足し、次の配列番号をセットする。ファイバー履歴特性番号 nm_type とそのタイプのワーク用配列のエレメント番号を構造体より取得する。
7. 増分軸方向ひずみと各増分曲げひずみより、該当するファイバーの軸方向増分ひずみを求める。
8. 解析種別が弾性である場合、もしくはその部材が弾性部材として設定されている場合、弾性材であるとして軸力を計算する。
9. ファイバー履歴特性番号 nm_type にしたがって処理を分類する。ここで第 3 層の階層構造が見られる。現在、ファイバー履歴特性種類は８つであるが、現在使用許可になっているのは 1 と 3 の 2 つである。
10. ここではバイリニア型の履歴特性となっており、サブルーチン BiLinear()を用いて処理している。後は仕様にしたがって各サブルーチンがコールされることになる。
11. ここでは直線コンクリート型の履歴特性となっており、サブルーチン Concrete()を用いて処理している。
12. 履歴特性を処理する各サブルーチンの処理が終了すると、ここに処理が集結する。ここではファイバーの増分ひずみが増分前のひずみに足しこまれる。
13. 塑性化したファイバーの数を断面内のファイバーの総数で割った値を d_state にセットする。この値がゼロのとき断面は弾性であるとする。また、d_state が 0.8 以上の場合は塑性ヒンジが発生したとする。それ以外は弾塑性状態であるとする。
14. 断面全体の断面積や断面二次モーメントを求めるために、まずファイバーデータが保存されている配列の番号を取得する。
15. 各断面特性パラメータをゼロセットする。
16. 全ファイバーについて断面積などの和を取っていく。
17. 全てのファイバーについて上記の断面特性を計算した後、軸方向剛

性の調整をサブルーチン jikuzero_control()を用いて行う。ここでは、軸方向剛性が全断面塑性化してゼロとなるのを防ぐ処理を行っている。

18. 求めた断面特性を構造体にセットし、次の剛性行列を計算するときに使用する。
19. これ以降は j 端における処理を行うが、ここでの処理は上記の i 端の処理と全く同一である。プログラムを一度検証されたい。

5.9.7 ファイバーモデルのせん断剛性

最後に、せん断変形を考慮する部材の剛性行列を作成するサブルーチンを以下に示す。プログラムは非常に単純なので容易に理解できよう。

```
C     ―――――――――――――――――――――――――――――――――
C     ●   SUBROUTINE /Shear_G
C       ―――――――――――――――――――――――――――――――
C     ●   せん断バネ剛性行列の計算(ok)
C     ―――――――――――――――――――――――――――――――――
      subroutine Shear_G(it,ak,Member)
      implicit real*8(A-H,O-Z)
      include "submain.h"
      record / member_s    / Member
      dimension ak(12,12)
C     ―――――――――――――――――――――――――――――――――
      if(it.lt.1) then
      akk=Member.i_shear_G
      else
      akk=Member.j_shear_G
      endif
      ak(2,2)=akk
      ak(2,8)=-akk
      ak(8,2)=-akk
      ak(8,8)=akk
      ak(3,3)=akk
      ak(3,9)=-akk
      ak(9,3)=-akk
      ak(9,9)=akk
      return
      end
```