4.6 予備計算

前節では、数値解析部分を解説したが、ここでは、計算に先立って行うデータ入力と予備計算について説明する。データ入力部は、計算を制御するためのデータ入力と計算に必要なデータ入力がある。前節と同様、この部分のプログラムを以下に示す。

```
c-------------------------------------------------------------
c
c    ★        サブルーチン定義
c
c-------------------------------------------------------------
c-------------------------------------------------------------
      subroutine submain_dynamic_a(i_calnum,iend_code,icontrol,ierr_dat,
     *           T,dt,n_step,ns_step,d_max_v,id_max_v,
     *           i_read_disp,F_disp,i_read_ndbalanceF,F_ndbalanceF,
     *           i_read_spring,F_fay,F_n_spring,F_my_spring,
     *           F_mz_spring,i_stat_spring,n_iterate,
     *           nm_iterate,numb_method)
c-------------------------------------------------------------
      implicit real*8(A-H,O-Z)

c-------------------------------------------------------------
c-------------------------------------------------------------
c
c    ★        システムからの制御情報を取得（以後実行文）
c
c-------------------------------------------------------------
c-------------------------------------------------------------
      if(icontrol .eq. 1  ) goto 9990    ! 解析処理へ
      if(icontrol .eq. 99 ) goto 9997    ! 終了処理へ
      do i=1,10
      M_alloc(i)=0                      ! 動的配列の確保をチェックする配列をゼロセット
      enddo
      open (damp_out,FILE='DOUTPUT')
c-----------------------------------★システムからのコントロール情報を取得(ok)
      call sysnam(FNX_file,N_analysis)
      if(N_analysis.ne.i_calnum) N_analysis=i_calnum  ! 解析番号を変更
      if(N_analysis.le.6) then
      ierr_dat=1
      call err_outf(ierr_dat)
      return
      endif
      ihan = 0
      ierr = 0
      NFILE=100
      ierr_dat =0
      write(damp_out,*) ' System file input ok. No. of analysis:',
     *                  N_analysis
c-----------------------------------★コントロールデータの内容を取得(ok)
      call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
      if(ihan.ne.0) then
      ierr_dat = 2
      call err_outf(ierr_dat)
```

```
      return
      endif
c-----------------------------------★動的解析ダイアログその１のデータを取得(ok)
      call dyctl1(ierr,NINDIT,GINDIS,F1SEC,fs_st,fl_st,ifp_st,IST,
     *           JIKUZERO,G_JIKUZERO_ALPH)
      if(ierr.ne.0) then
       ierr_dat = 3
       call err_outf(ierr_dat)
      endif
c-----------------------------------★動的解析ダイアログその２のデータを取得(ok)
      call dyctl2(ierr,NSTEP,F2SEC,DELT,IGRA,IBETA,BETA,GUMMA,XGAL,
     *           NNTIME,EPSDSP,load_memb_mass,dt_M_filter,IT_ANALYS)
      if(ierr.ne.0) then
       ierr_dat = 4
       call err_outf(ierr_dat)
      endif
c-----------------------------------★解析結果の出力パラメータを取得(ok)
      call doutcl(ierr,IWSTP,SOUTSC,DMAXCK,No_section)
      if(ierr.ne.0) then
       ierr_dat = 5
       call err_outf(ierr_dat)
      endif
c-----------------------------------★減衰ダイアログのデータを取得(ok)
      call damctl(ierr,NREAD,ITYDP,NDMP,NDMP2,NHH,HH,QHH)
      if(ierr.ne.0) then
       ierr_dat = 6
       call err_outf(ierr_dat)
      endif
c-----------------------------------★制御情報の取得に失敗した場合はここで戻る
      if(ierr_dat .ne. 0 ) return
c-------------------------------------------------------------
c
c    ★        制御情報を構造体にセット
c
c-------------------------------------------------------------
      call Set_control(Control,N_analysis,NINDIT,GINDIS,
     *                 IWSTP,SOUTSC,DMAXCK,in_disp,in_stres,
     *                 IT_ANALYS,JIKUZERO,G_JIKUZERO_ALPH)       !ok (in_disp,in_stres  未定義)
c     write(damp_out,*) ' Set_control Ok'
      call Set_model_type(Model_type)
c     write(damp_out,*) ' Set_model_type Ok'
      call Set_newmark(Newmark_P,F1SEC,F2SEC,HH(1),HH(2),
     *                 QHH(1),QHH(2),DELT,BETA,EPSDSP,NNTIME,GUMMA,
     *                 ITYDP,NDMP,NDMP2)
c     write(damp_out,*) ' Set_newmark Ok'
      call Set_dynamic_load(Dynamic_load,IST,IGRA,
     *                      XGAL,load_memb_mass)
c     write(damp_out,*) ' Set_dynamic_load Ok'
c-------------------------------------------------------------
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
c-------------------------------------------------------------
      nfix=5
```

```
      nfi=1
      call infile(nfi,nfix,ierr)
      if(ierr.ne.0) then
      ierr_dat =10
      call err_outf(ierr_dat)
      return
      endif
      call Get_parameters(Parameter_C,ierr)
      close(nfix)
      if(ierr .ne. 0) then
      ierr_dat =11
      call err_outf(ierr_dat)
      return
      endif
c―――――――――――――――――――――――――――――――――★動的解析制限条件チェック
      call Check_analysis_condition(Parameter_C,ierr,
     *    IAnalysis_number_condition,Number_of_nodes,Number_of_members)
      if(ierr .ne. 0) then
      ierr_dat =299
      call err_outf(ierr_dat)
      return
      endif
c―――――――――――――――――――――――――――――――――★地震加速度を予備入力(ok)
      call Get_earth_load(1,acc_earth,Dynamic_load,ierr)
      if(ierr.ne.0) then
      ierr_dat =21
      call err_outf(ierr_dat)
      return
      endif
c―――――――――――――――――――――――――――――――――★節点荷重履歴を予備入力(ok)
      call Get_point_load(1,fdd_point,Dynamic_load,ierr,
     *                    Newmark_P,fs_st,fl_st,ifp_st)
      if(ierr.ne.0) then
      ierr_dat =22
      call err_outf(ierr_dat)
      return
      endif
c――――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★   DLL を使用する場合、ワークエリアとして必要な大きさ及び DLL 番号を得る
c                                      DLL subroutine code No. 1
c
c――――――――――――――――――――――――――――――――――――――――――――――――
c      if(Parameter_C.n_element_dll .ne. 0 )
c     * call Get_DLL_code(dll_code,n_request1,n_request2,
c     *                   n_request3,n_request4)
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_point            ! 節点数
      ALLOCATE (
     *   fll_static_point(3,6,N),am_point(2,N),
     *   fll_force_point(3,N)
     *         )
      N=Parameter_C.n_member           ! 部材数
      ALLOCATE (
     *   am_member(12,12,N),
     *   rot_memb_t(3,3,N),rot_memb(3,3,2,N),
     *   ak_linear(12,12,N),ak_nonlinear(12,12,N)
     *         )
      N=Parameter_C.n_local_coord      ! 局所座標系を使用する場合
      if(N.ne.0)ALLOCATE (
     *   rot_local(3,3,N)
     *)
       M_alloc(1)=1
c―――――――――――――――――――――――――――――――――――――――――
c
c    ★        構造・荷重データを入力し、データの設定を行う
c
c―――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――――――――――――★基本構造データを入力(ok)
      nfix=5
       nfi=1
      call infile(nfi,nfix,ierr)
       if(ierr.ne.0) then
      ierr_dat =10
       call err_outf(ierr_dat)
       return
       endif
      call Get_structure(Point,Member,Element,Parameter_C,
     *                   Model_type,ierr)
       close(nfix)
       if(ierr .ne. 0) then
      ierr_dat =iabs(ierr)
c                         err No. 12-19 使用
       call err_outf(ierr_dat)
       return
       endif
c      write(damp_out,*) ' Get_structure ok'
c―――――――――――――――――――――――――――――――――――――――――
c
c    ★        制震セミアクティブダンパー用フィルターのパラメータセット
c
c―――――――――――――――――――――――――――――――――――――――――
      if(Model_type.n_m_model(6) .ne. 0) then
      call Set_Maxwell_filter(dt_M_filter,Newmark_P.dt,Model_type,ierr)
```

```
      if(ierr.ne.0) then
       ierr_dat = 20
       call err_outf(ierr_dat)
       return
       endif
c     write(damp_out,*) ' Set_Maxwell_filter Ok'
      endif
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-----------------------------------------------------------
      N=Model_type.n_m_damp
      if(N.ne.0)then
      ALLOCATE (ac_member(12,12,N))
      endif
c-------------------------------★Model_No.1 通常の有限要素弾塑性モデル
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
c-------------------------------★Model_No.3 3 次元軸力弾塑性モデル
c     n=Model_type.n_m_model(3)
c     if(n.ne.0)then
c     ALLOCATE(N_Buckling(n))
c     endif
c-------------------------------★Model_No.4 3 次元ケーブル弾塑性モデル
c-------------------------------★Model_No.5 3 次元免振モデル
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      ALLOCATE (MSS_work(n))
      endif
c-------------------------------★Model_No.6 3 次元制震 Maxwell モデル
      n=Model_type.n_m_model(6)
      if(n.ne.0) then
      ALLOCATE (E_model6_real(n))
      endif
c-------------------------------★Model_No.7 3 次元バネモデル
c     n=Model_type.n_m_model(7)
c     if(n.ne.0) then
c       ALLOCATE (E_model7(n))
c     endif
c-------------------------------★ファイバーモデル
c-------------------------------★Model_No.11 両端ファイバーモデル
      n= Model_type.n_e_model(11)        !要素モデルの数
      if(n.ne.0) then
      ALLOCATE (E_model11(n))
      endif
      n= Model_type.n_m_model(11)        !部材モデルの数
      if(n.ne.0) then
      ALLOCATE ( M_model11(n))
      endif
c-------------------------------★Model_No.12 両端、中央ファイバーモデル
```

```
      n= Model_type.n_e_model(12)        !要素モデルの数
       if(n.ne.0) then
      ALLOCATE (E_model12(n))
      endif
      n= Model_type.n_m_model(12)        !部材モデルの数
       if(n.ne.0) then
      ALLOCATE ( M_model12(n))
      endif
c――――――――――――――――――――――――――――――――――★Model_No.13 両端、中央ファイバーモデル
      n= Model_type.n_e_model(18)        !要素モデルの数
       if(n.ne.0) then
       ALLOCATE (E_model13(n))
      endif
      n= Model_type.n_m_model(18)        !部材モデルの数
       if(n.ne.0) then
       ALLOCATE ( M_model13(n))
      endif
c――――――――――――――――――――――――――――――――――★Model_No.15 両端ファイバーFEM モデル
      n= Model_type.n_e_model(15)        !要素モデルの数
       if(n.ne.0) then
      ALLOCATE (E_model15(n))
      endif
      n= Model_type.n_m_model(15)        !部材モデルの数
       if(n.ne.0) then
      ALLOCATE ( M_model15(n))
      endif
c――――――――――――――――――――――――――――――――――★Model_No.21 両端 MS モデル
      n= Model_type.n_e_model(13)        !要素モデルの数
       if(n.ne.0) then
      ALLOCATE (E_model21(n))
      endif
      n= Model_type.n_m_model(13)        !部材モデルの数
       if(n.ne.0) then
      ALLOCATE ( M_model21(n))
      endif
c――――――――――――――――――――――――――――――――――★Model_No.22 両端、中央 MS モデル
      n= Model_type.n_e_model(14)        !要素モデルの数
       if(n.ne.0) then
      ALLOCATE (E_model22(n))
      endif
      n= Model_type.n_m_model(14)        !部材モデルの数
       if(n.ne.0) then
      ALLOCATE ( M_model22(n))
      endif
c――――――――――――――――――――――――――――――――――★Model_No.31 両端アナロジーモデル
      n= Model_type.n_e_model(16)        !要素モデルの数
       if(n.ne.0) then
      ALLOCATE (E_model31(n))
      endif
      n= Model_type.n_m_model(16)        !部材モデルの数
       if(n.ne.0) then
      ALLOCATE ( M_model31(n))
      endif
c――――――――――――――――――――――――――――――――――★Model_No.32 両端、中央アナロジーモデル
```

```
      n= Model_type.n_e_model(17)        !要素モデルの数
       if(n.ne.0) then
      ALLOCATE (E_model32(n))
      endif
      n= Model_type.n_m_model(17)        !部材モデルの数
       if(n.ne.0) then
      ALLOCATE ( M_model32(n))
      endif
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―――――――――――――――――――――――――――――――――★Model_No.51 3 次元プレテンション動作モデル
c      n=Model_type.n_m_model(51)
c     if(n.ne.0) then
c     ALLOCATE (M_model51(n),
c    *          E_model51(n))
c      endif
c――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★        配列の大きさを動的確保する（その２：DLL 用）
c
c――――――――――――――――――――――――――――――――――――――――――――――
c     if(Parameter_C.n_element_dll .ne. 0 ) then
c      N1=n_request1* Parameter_C.n_element_dll
c      N2=n_request2* Parameter_C.n_element_dll
c       N3=n_request3* Parameter_C.n_member_dll
c       N4=n_request4* Parameter_C.n_member_dll
c      ALLOCATE (
c     *   work1_elememt(N1),work2_elememt(N2),
c     *   work1_member(N3),work2_member(N4)
c     *         )
c     endif
c―――――――――――――――――――――――――――――――――★節点荷重、加速度データ領域を確保
      if(Dynamic_load.n_load_dynamic .ne. 0) then
      ALLOCATE (acc_earth(3,Dynamic_load.n_load_dynamic))
       endif
      if(Dynamic_load.n_load_point .ne. 0) then
      ALLOCATE (fdd_point(3,Dynamic_load.n_load_point))
       endif
       M_alloc(2)=1
c――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★        構造・荷重データを入力し、データの設定を行う（その２）
c
c――――――――――――――――――――――――――――――――――――――――――――――
c―――――――――――――――――――――――――――――――――★地震荷重を入力(ok)
      if(Dynamic_load.load_dynamic(1)  .ne. 0 .or.
     *   Dynamic_load.load_dynamic(2)  .ne. 0 .or.
```

```
     *   Dynamic_load.load_dynamic(3)  .ne. 0 ) then
      call Get_earth_load(2,acc_earth,Dynamic_load,ierr)
       if(ierr.ne.0) then
      ierr_dat =21
       call err_outf(ierr_dat)
       return
       endif
c     write(damp_out,*) ' Get_earth_load Ok'
      endif
c―――――――――――――――――――――――――――――――――★節点荷重分布を入力(ok)
      if(Dynamic_load.load_point(1)  .ne. 0 .or.
     *   Dynamic_load.load_point(2)  .ne. 0 .or.
     *   Dynamic_load.load_point(3)  .ne. 0 ) then
      call Get_point_loadf(fll_static_point,Parameter_C,
     *                     Dynamic_load,ierr)
       if(ierr.ne.0) then
      ierr_dat =22
       call err_outf(ierr_dat)
       return
       endif
c     write(damp_out,*) ' Get_point_loadf Ok'
c―――――――――――――――――――――――――★節点荷重時刻歴を入力およびセット
      call Get_point_load(2,fdd_point,Dynamic_load,ierr,
     *                    Newmark_P,fs_st,fl_st,ifp_st)
       if(ierr.ne.0) then
      ierr_dat = 23
       call err_outf(ierr_dat)
       return
       endif
c     write(damp_out,*) ' Get_point_load Ok'
      endif
c―――――――――――――――――――――――――――――――――★初期不整データを入力(ok)
      if(Control.init_imperfection .ne. 0) then
      nfix=5
       nfi=16
      call infile(nfi,nfix,ierr)
       if(ierr.ne.0) then
      ierr_dat = 24
       call err_outf(ierr_dat)
       return
       endif
      call Get_imperfection(Control.amp_imperfection,Point,Parameter_C)
       close(nfix)
       endif
c―――――――――――――――――――――――――――――――――★システム内要素データを入力
c    ★        ファイバーデータ
c―――――――――――――――――――――――――――――★ファイバーデータの予備入力(ok)
      n_element=Model_type.n_e_model(11)+Model_type.n_e_model(12)+
     *          Model_type.n_e_model(15)+
     *          Model_type.n_e_model(14)+Model_type.n_e_model(13)+
     *          Model_type.n_e_model(16)+Model_type.n_e_model(17)+
     *          Model_type.n_e_model(18)++Model_type.n_e_model(19)
      if(n_element .ne. 0) then
      nfix=5
```

```
       nfi=54
       call infile(nfi,nfix,ierr)
       if(ierr.ne.0) then
       ierr_dat = 25
       call err_outf(ierr_dat)
       return
       endif
       call Fiber_input(0,ierr,Parameter_C.n_member,
      *      Parameter_C.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)
       close(nfix)
       if(ierr.ne.0)then
       ierr_dat = 26
       call err_outf(ierr_dat)
       return
       endif
c      write(damp_out,*) ' Fiber_input ok'
c―――――――――――――――――――――――――――――★ファイバーモデル領域セット
      n= Model_type.nm_div_fmodel          !要素モデル内のサブ要素の数
       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
      M_alloc(3)=1
c―――――――――――――――――――――――――――――★ファイバーモデルデータ入力
       nfix=5
        nfi=54
       call infile(nfi,nfix,ierr)
       if(ierr.ne.0) then
       ierr_dat = 25
       call err_outf(ierr_dat)
       return
       endif
       call Fiber_input(1,ierr,Parameter_C.n_member,
      *      Parameter_C.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)
       close(nfix)
       if(ierr.ne.0) then
       ierr_dat = 26
       call err_outf(ierr_dat)
       return
       endif
c      write(damp_out,*) ' Fiber_input ok'
      endif
c―――――――――――――――――――――――――――――★修正 R0 モデル領域セット
```

```
      n=Model_type.n_m_ro_model
       if(n.ne.0) then
       nfix=5
        nfi=53
       call infile(nfi,nfix,ierr)
       if(ierr.ne.0) then
      ierr_dat = 35
       call err_outf(ierr_dat)
       return
       endif
       call RO_data_input(n,RO_work,Element,Parameter_C.n_element,ierr)
       close(nfix)
       if(ierr.ne.0) then
      ierr_dat = 36
       call err_outf(ierr_dat)
       return
       endif
c      write(damp_out,*) ' RO_data_input ok'
       endif
c------------------------------★DLL 内要素データを入力
c                                  DLL subroutine code No. 2
c       n_element=Parameter_C.n_member_dll
c       if(n_element .ne. 0) then
c       nfix=5
c      nfi=60
c       call infile(nfi,nfix,ierr)
c      if(ierr.ne.0) then
c       ierr_dat =13
c      return
c      endif
c       call Get_element_dll(work1_elememt,work2_elememt,Parameter_C)
c      close(nfix)
c      endif
c------------------------------★質量データを入力(ok)
      nfix=5
       nfi=2
      call infile(nfi,nfix,ierr)
       if(ierr.ne.0) then
      ierr_dat = 27
       call err_outf(ierr_dat)
       return
       endif
      call Get_mass(ierr,Point,Parameter_C)
       close(nfix)
       if(ierr.ne.0) then
      ierr_dat = 28
       call err_outf(ierr_dat)
       return
       endif
c      write(damp_out,*) ' Get_mass Ok'
c------------------------------★レーリー減衰を入力(ok)
      nfix=5
       nfi=43
      call infile(nfi,nfix,ierr)
```

```
      if(ierr.ne.0) then
      ierr_dat = 29
      call err_outf(ierr_dat)
      return
      endif
      call Get_damp(Newmark_P,ierr)
      close(nfix)
      if(ierr.ne.0) then
      ierr_dat =30
      call err_outf(ierr_dat)
      return
      endif
c     write(damp_out,*) ' Get_damp Ok'
c-----------------------------------------------------------
c
c    ★        動的解析のための予備計算を行う
c
c-----------------------------------------------------------
c-------------------------------★構造物の不安定性チェック
c      call check_structure(Point,Element,Parameter_C)
c-------------------------------★部材長さ計算(ok)
      call Cal_member_length(Member,Point,Parameter_C)
c     write(damp_out,*) ' Cal_member_length Ok'
c-------------------------------★モデルの初期設定
      call Set_initial_data(Element,Member,Parameter_C,Newmark_P,
     *           E_model6_real,Model_type)
c     write(damp_out,*) ' Set_initial_data Ok'
c-------------------------------★DLL 非線形エリアの設定
c      if(Parameter_C.n_member_dll .ne. 0) then
c      call Set_nonlinear_S(Element,Parameter_C,
c     *     work1_elememt,work2_elememt,work1_member,work2_member)
c      endif
c-------------------------------★座標変換行列計算
      call Get_rotate_all(rot_memb_t,Parameter_C,Point,Member)
c                write(damp_out,*) ' Get_rotate_all Ok'
c-------------------------------★局所座標変換行列計算
      call Get_rot_local(rot_local,Parameter_C,Point)
c                write(damp_out,*) ' Get_rot_local Ok'
c-------------------------------★釣合座標系標変換行列計算
      call Get_rotate(rot_memb,rot_memb_t,rot_local,
     *               Parameter_C,Member)
c     write(damp_out,*) ' Get_rotate Ok'
c-------------------------------★節点拘束表の作成
c                                                 未知数等をセット
      call Set_restraint_point(Parameter_C,Point,Control)
c     write(damp_out,*) ' Set_restraint_point Ok'
c-----------------------------------------------------------
c
c    ★        配列の大きさを動的確保する（その３）
c
c-----------------------------------------------------------
      N=Parameter_C.n_unknown
c      ALLOCATE (test_vector(N))
      ALLOCATE (
```

```
*     disp_point(N),vel_point(N),acc_point(N),
*     est_disp_point(N),est_vel_point(N),
*     est_ddisp_point(N)
*         )
 ALLOCATE (
*   result_disp_point(N),result_vel_point(N),result_acc_point(N),
*   past_disp_point(N),past_vel_point(N),past_acc_point(N),
*   past_dacc_point(N)
*         )
 ALLOCATE (
*   ld_point(N),ld_point_repeat(N),fld_static(3,N),
*   a_vector(N),b_vector(N)
*         )
 ALLOCATE (max_h_sky(0:N+1)
*         )
  M_alloc(4)=1
c――――――――――――――――――――――――――――――――――★部材両端の拘束表作成(ok)
      call Set_restraint_member(Parameter_C,Member,Point)
c     write(damp_out,*)' Set_restraint_member Ok'
c――――――――――――――――――――――――――――――――――★スカイライン変換表作成(ok)
c――――――――――――――――――――――――――――――――――★スカイライン行列の領域数等をセット
      call Cal_table_sky(max_h_sky,Parameter_C,Member)
c     write(damp_out,*) ' Cal_table_sky Ok'
c――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★        配列の大きさ（スカイライン用）を動的確保する（その４）
c
c――――――――――――――――――――――――――――――――――――――――――――――
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の数
       if(n.ne.0) then
      ALLOCATE (Concrete_work(n))
      endif
c――――――――――――――――――――――――――――――――――★全体剛性行列など
      ALLOCATE (gskym(Parameter_C.n_skyline))
      ALLOCATE (gskym_d(Parameter_C.n_unknown))
      ALLOCATE (nwork(Parameter_C.n_unknown))
      ALLOCATE (twork(Parameter_C.n_unknown))
       M_alloc(5)=1
c――――――――――――――――――――――――――――――――――★部材両端中央応力、力のゼロセット(ok)
      call  Set_pointforce_zero(Member,Parameter_C.n_member)
c     write(damp_out,*) ' Set_pointforce_zero Ok'
c――――――――――――――――――――――――――――――――――★部材応力のゼロセット(ok)
      call  Set_stress_zero(Member,Parameter_C.n_member)
c     write(damp_out,*) ' Set_stress_zero Ok'
c――――――――――――――――――――――――――――――――――★MSS モデルの初期設定
```

```
      n=Model_type.n_m_model(5)
       if(n.ne.0) then
      call Cal_MSS_dat(Member,Element,Model_type,
     *                 MSS_work,Parameter_C)
c     write(damp_out,*) ' Cal_MSS_dat Ok'
      endif
c―――――――――――――――――――――――――――――――――★平面問題における部材の拘束方向チェック
      call Check_R_direction(Control.analysis_3D,Parameter_C,
     *                       Member,rot_memb)
c     write(damp_out,*) ' Check_R_direction ok'
c―――――――――――――――――――――――――――――――――★部材の線形剛性計算(ok)
      call Cal_stiff_linear(Model_type,Element,Member,Parameter_C,
     *      ak_linear,E_model11,E_model_fiber,M_model11,M_model_fiber,
     *      E_model12,M_model12,E_model13,M_model13,E_model15,M_model15,
     *      E_model21,M_model21,E_model22,M_model22,
     *      E_model31,M_model31,E_model32,M_model32,
     *      E_model33,M_model33,
     *      Bilinear_work,Trilinear_work,Concrete_work,
     *      work1_elememt,work2_elememt,work1_member,work2_member)
c     write(damp_out,*) ' Cal_stiff_linear Ok'
c―――――――――――――――――――――――――――――――――★剛性の釣合座標系への変換(ok)
      call Rotate_stiffness(Parameter_C,ak_linear,rot_memb)
c     write(damp_out,*) ' Rotate_stiffness Ok'
c―――――――――――――――――――――――――――――――――★部材の減衰行列計算(ok)
      if(Parameter_C.nc_member .ne. 0) then
      call Cal_damp_linear(Element,Member,Parameter_C,ac_member,
     *           E_model6_real,work1_elememt,
     *           work2_elememt,work1_member,work2_member)
c     write(damp_out,*) ' Cal_damp_linear Ok'
c―――――――――――――――――――――――――――――――――★部材減衰行列の釣合座標系への変換(ok)
      call Rotate_damp(Parameter_C.n_member,ac_member,rot_memb,Member)
c     write(damp_out,*) ' Rotate_damp Ok'
      end if
c―――――――――――――――――――――――――――――――――★節点集中質量セット(ok)
      call Set_mass(Point,Parameter_C,am_point)
c     write(damp_out,*) ' Set_mass Ok'
c―――――――――――――――――――――――――――――――――★節点荷重セット(ok)
      call Set_point_load(fll_static_point,Parameter_C,
     *              Dynamic_load,Point,fld_static,rot_local)
c     write(damp_out,*) ' Set_point_load Ok'
c―――――――――――――――――――――――――――――――――★接線剛性のコピー(ok)
      call Initset_nonlin_stiff(ak_linear,ak_nonlinear,
     *                          Parameter_C.n_member)
c     write(damp_out,*) ' Initset_nonlin_stiff Ok'
c―――――――――――――――――――――――――――――――――★ベクトルのゼロセット
c―――――――――――――――――――――――――――――――――★増分前の変位、速度、加速度(ok)
      call Set_zero_v(past_disp_point, past_vel_point,
     *       past_acc_point,past_dacc_point,Parameter_C.n_unknown)
c     write(damp_out,*) ' Set_zero_v Ok'
c―――――――――――――――――――――――――――――――――★最大変位等のゼロセット(ok)
      call Clear_max_disp(Max_disp,Parameter_C.n_point)
c     write(damp_out,*) ' Clear_max_disp Ok'
c―――――――――――――――――――――――――――――――――★最大、最小応力等のゼロセット(ok)
      call Clear_max_stress(Max_stress,Parameter_C.n_member)
```

```
c      write(damp_out,*) ' Clear_max_stress Ok'
c-----------------------------★不釣合力のゼロセット(ok)
       call Set_zero_pointforce(fll_force_point,Parameter_C.n_point)
c
c-------------------------------------------------------------
c
c    ★          静的解析結果等を取り込む
c                （初期変位、初期応力を入力）
c
c-------------------------------------------------------------
c-----------------------------★初期変位を入力
       if(Control.init_disp.ne.0) then
       nfix=5
       nfi=16
c      call infile(nfi,nfix,ierr)
       if(ierr.ne.0) then
       ierr_dat = 31
       call err_outf(ierr_dat)
       return
       endif
c      call Get_init_disp(Point,Member,Parameter_C,ierr)
       close(nfix)
       if(ierr.ne.0) then
       ierr_dat =32
       call err_outf(ierr_dat)
       return
       endif
c      write(damp_out,*) ' Get_damp Ok'
       endif
c-----------------------------★初期応力を入力
       if(Control.init_stress.ne.0) then
       nfix=5
       nfi=52
c      call infile(nfi,nfix,ierr)
       if(ierr.ne.0) then
       ierr_dat = 33
       call err_outf(ierr_dat)
       return
       endif
c      call Get_init_stress(Point,Member,Parameter_C,ierr)
       close(nfix)
       if(ierr.ne.0) then
       ierr_dat =34
       call err_outf(ierr_dat)
       return
       endif
c      write(damp_out,*) ' Get_init_stress Ok'
       endif
c-------------------------------------------------------------
c
c    ★          解析結果を出力するファイル群をオープンする(ok)
c
c-------------------------------------------------------------
       call flcheck(ifl,ifly,iflz,ierr,Control.type_analysis)
```

```
      if(ierr.ne.0) then
      ierr_dat =14
      endif
c――――――――――――――――――――――――――――――――――――★出力指定した断面がファイバー要素
c――――――――――――――――――――――――――――――――――――★かどうかチェック(ok)
      call out_section_check(Member,Element,
     *               Parameter_C.n_member,No_section,Out_section,
     *               ifl,iflz,i_print)
c――――――――――――――――――――――――――――――――――――★ファイル名一覧出力
      write(damp_out,*) ' flcheck Ok'
     *  ,Dynamic_load.load_dynamic(1)
      do i=1,16
      write(damp_out,'(i4,3i6)') i,ifl(i),iflz(i),ifly(i)
      enddo
c――――――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★        描画用データのセット
c
c――――――――――――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――――――★変位
      if(i_read_disp .ne. 0) then
      call Set_preset_disp(0,n_point,past_disp_point,F_disp,Point,
     *                           rot_local,Parameter_C)
c      write(damp_out,*) ' Set_preset_disp ok'
      endif
c――――――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★        解析ステップをセットする
c
c――――――――――――――――――――――――――――――――――――――――――――――――――
      ns_step=1
      n_step=10
      d_max_v=0.
      id_max_v=0
      stimes=secnds(0.0)                        ! 解析時間を計る
c――――――――――――――――――――――――――――――――――――★解析手法のセット
      N_implicit_method = 1  !陰解法は１回で反復法に戻る
      Iteration_method  = 1  !最初は反復法を使用
c――――――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★        予備計算はここで終了（GUI に戻る）
c
c――――――――――――――――――――――――――――――――――――――――――――――――――
      return
```

前節と同様に、プログラムの骨組みを用いて、処理の流れを理解しよう。多少長いが、プログラムの構造は単純なので理解し易い。

```
c――――――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★        サブルーチン定義
c
```

```
c-----------------------------------------------------------------
c-----------------------------------------------------------------
      subroutine submain_dynamic_a(i_calnum,iend_code,icontrol,ierr_dat,          ! 1
     *           T,dt,n_step,ns_step,d_max_v,id_max_v,
     *           i_read_disp,F_disp,i_read_ndbalanceF,F_ndbalanceF,
     *           i_read_spring,F_fay,F_n_spring,F_my_spring,
     *           F_mz_spring,i_stat_spring,n_iterate,
     *           nm_iterate,numb_method)
c-----------------------------------------------------------------
      implicit real*8(A-H,O-Z)                                                   ! 2
c-----------------------------------------------------------------
c-----------------------------------------------------------------
c
c    ★        システムからの制御情報を取得（以後実行文）
c
c-----------------------------------------------------------------
c-----------------------------------------------------------------
      if(icontrol .eq. 1  ) goto 9990    ! 解析処理へ                           ! 3
      if(icontrol .eq. 99 ) goto 9997    ! 終了処理へ                           ! 4
      do i=1,10
      M_alloc(i)=0          ! 動的配列の確保をチェックする配列をゼロセット       ! 5
      enddo
      open (damp_out,FILE='DOUTPUT')                                             ! 6
c-------------------------------★システムからのコントロール情報を取得(ok)
      call sysnam()                                                              ! 7
c-------------------------------★コントロールデータの内容を取得(ok)
      call ctlset()                                                              ! 8
c-------------------------------★動的解析ダイアログその１のデータを取得(ok)
      call dyctl1()                                                              ! 9
c-------------------------------★動的解析ダイアログその２のデータを取得(ok)
      call dyctl2()                                                              ! 10
c-------------------------------★解析結果の出力パラメータを取得(ok)
      call doutcl()                                                              ! 11
c-------------------------------★減衰ダイアログのデータを取得(ok)
      call damctl()                                                              ! 12
c-------------------------------★制御情報の取得に失敗した場合はここで戻る
      if(ierr_dat .ne. 0 ) return                                                ! 13
c-----------------------------------------------------------------
c
c    ★        制御情報を構造体にセット
c
c-----------------------------------------------------------------
      call Set_control()       !ok (in_disp,in_stres  未定義)                  ! 14
      call Set_model_type()                                                      ! 15
      call Set_newmark()                                                         ! 16
      call Set_dynamic_load()                                                    ! 17
c-----------------------------------------------------------------
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
c-----------------------------------------------------------------
      call Get_parameters()                                                      ! 18
c-------------------------------★地震加速度を予備入力(ok)
      call Get_earth_load()                                                      ! 19
```

```
c―――――――――――――――――――――――――――――――――★節点荷重履歴を予備入力(ok)
      call Get_point_load()                                                   ! 20
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))                                ! 21
      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_point            ! 節点数
      ALLOCATE (
     *   fll_static_point(3,6,N),am_point(2,N),
     *   fll_force_point(3,N)
     *         )
      N=Parameter_C.n_member           ! 部材数
      ALLOCATE (
     *   am_member(12,12,N),
     *   rot_memb_t(3,3,N),rot_memb(3,3,2,N),
     *   ak_linear(12,12,N),ak_nonlinear(12,12,N)
     *         )
      N=Parameter_C.n_local_coord      ! 局所座標系を使用する場合
      if(N.ne.0)ALLOCATE (
     *   rot_local(3,3,N)
     *)
      M_alloc(1)=1                                                           ! 22
c―――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★      構造・荷重データを入力し、データの設定を行う
c
c―――――――――――――――――――――――――――――――――――――――――――――――
c―――――――――――――――――――――――――――――――――★基本構造データを入力(ok)
      call Get_structure()                                                    ! 23
c―――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★      制震セミアクティブダンパー用フィルターのパラメータセット
c
c―――――――――――――――――――――――――――――――――――――――――――――――
      if(Model_type.n_m_model(6) .ne. 0) then                                 ! 24
      call Set_Maxwell_filter()                                               ! 25
      endif
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―――――――――――――――――――――――――――――――――――――――――――――――
      N=Model_type.n_m_damp
```

```
      if(N.ne.0)then                                                    ! 26
      ALLOCATE (ac_member(12,12,N))                                     ! 27
      endif
c――――――――――――――――――――――――――――――――――★Model_No.1 通常の有限要素弾塑性モデル
c――――――――――――――――――――――――――――――――――★Model_No.2 3 次元せん断弾塑性モデル
c          以後、処理法は同じ
      ALLOCATE (RO_work(n))
c――――――――――――――――――――――――――――――――――★Model_No.4 3 次元ケーブル弾塑性モデル
c――――――――――――――――――――――――――――――――――★Model_No.5 3 次元免振モデル
      ALLOCATE (MSS_work(n))
c――――――――――――――――――――――――――――――――――★Model_No.6 3 次元制震 Maxwell モデル
      ALLOCATE (E_model6_real(n))
c――――――――――――――――――――――――――――――――――★ファイバーモデル
c――――――――――――――――――――――――――――――――――★Model_No.11 両端ファイバーモデル
      ALLOCATE (E_model11(n))
      ALLOCATE (M_model11(n))
c――――――――――――――――――――――――――――――――――★Model_No.12 両端、中央ファイバーモデル
      ALLOCATE (E_model12(n))
      ALLOCATE (M_model12(n))
c――――――――――――――――――――――――――――――――――★Model_No.13 両端、中央ファイバーモデル
      ALLOCATE (E_model13(n))
      ALLOCATE (M_model13(n))
c――――――――――――――――――――――――――――――――――★Model_No.15 両端ファイバーFEM モデル
      ALLOCATE (E_model15(n))
      ALLOCATE (M_model15(n))
c――――――――――――――――――――――――――――――――――★Model_No.21 両端 MS モデル
      ALLOCATE (E_model21(n))
      ALLOCATE (M_model21(n))
c――――――――――――――――――――――――――――――――――★Model_No.22 両端、中央 MS モデル
      ALLOCATE (E_model22(n))
      ALLOCATE (M_model22(n))
c――――――――――――――――――――――――――――――――――★Model_No.31 両端アナロジーモデル
      ALLOCATE (E_model31(n))
      ALLOCATE (M_model31(n))
c――――――――――――――――――――――――――――――――――★Model_No.32 両端、中央アナロジーモデル
      ALLOCATE (E_model32(n))
      ALLOCATE (M_model32(n))
c――――――――――――――――――――――――――――――――――★Model_No.33 両端ピン、中央アナロジーモデル
      ALLOCATE (E_model33(n))
      ALLOCATE (M_model33(n))
c――――――――――――――――――――――――――――――――――★節点荷重、加速度データ領域を確保
      ALLOCATE (acc_earth(3,Dynamic_load.n_load_dynamic))
      ALLOCATE (fdd_point(3,Dynamic_load.n_load_point))
      M_alloc(2)=1                                                      ! 28
c――――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★         構造・荷重データを入力し、データの設定を行う（その２）
c
c――――――――――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――――★地震荷重を入力(ok)
      call Get_earth_load()                                             ! 29
c――――――――――――――――――――――――――――――――――★節点荷重分布を入力(ok)
      call Get_point_loadf()                                            ! 30
c――――――――――――――――――――――――――★節点荷重時刻歴を入力およびセット
```

```
      call Get_point_load()                                          ! 31
c-----------------------------------★初期不整データを入力(ok)
      call Get_imperfection()                                        ! 32
c-----------------------------------★システム内要素データを入力
c    ★        ファイバーデータ
c-----------------------------------★ファイバーデータの予備入力(ok)
      call Fiber_input()                                             ! 33
c-----------------------------------★ファイバーモデル領域セット
      ALLOCATE (M_model_fiber(n))                                    ! 34
      ALLOCATE (E_model_fiber(n))
      M_alloc(3)=1                                                   ! 35
c-----------------------------------★ファイバーモデルデータ入力
      call Fiber_input()                                             ! 36
c-----------------------------------★修正 RO モデル領域セット
      call RO_data_input()                                           ! 37
c-----------------------------------★質量データを入力(ok)
      call Get_mass()                                                ! 38
c-----------------------------------★レーリー減衰を入力(ok)
      call Get_damp()                                                ! 39
c-------------------------------------------------------------
c
c    ★        動的解析のための予備計算を行う
c
c-------------------------------------------------------------
c-----------------------------------★部材長さ計算(ok)
      call Cal_member_length()                                       ! 40
c-----------------------------------★モデルの初期設定
      call Set_initial_data()                                        ! 41
c-----------------------------------★座標変換行列計算
      call Get_rotate_all()                                          ! 42
c-----------------------------------★局所座標変換行列計算
      call Get_rot_local()                                           ! 43
c-----------------------------------★釣合座標系標変換行列計算
      call Get_rotate()                                              ! 44
c-----------------------------------★節点拘束表の作成
c                                     未知数等をセット
      call Set_restraint_point()                                     ! 45
c-------------------------------------------------------------
c
c    ★        配列の大きさを動的確保する（その３）
c
c-------------------------------------------------------------
      ALLOCATE (                                                     ! 46
     *     disp_point(N),vel_point(N),acc_point(N),
     *     est_disp_point(N),est_vel_point(N),
     *     est_ddisp_point(N)
     *         )
      ALLOCATE (
     *   result_disp_point(N),result_vel_point(N),result_acc_point(N),
     *   past_disp_point(N),past_vel_point(N),past_acc_point(N),
     *   past_dacc_point(N)
     *         )
      ALLOCATE (
     *   ld_point(N),ld_point_repeat(N),fld_static(3,N),
```

```
*   a_vector(N),b_vector(N)
*         )
 ALLOCATE (max_h_sky(0:N+1)
*         )
 M_alloc(4)=1                                                    ! 47
c――――――――――――――――――――――――――――――――――★部材両端の拘束表作成(ok)
      call Set_restraint_member()                                ! 48
c――――――――――――――――――――――――――――――――――★スカイライン変換表作成(ok)
c――――――――――――――――――――――――――――――――――★スカイライン行列の領域数等をセット
      call Cal_table_sky()                                       ! 49
c――――――――――――――――――――――――――――――――――――――――――――――――
c
c    ★        配列の大きさ（スカイライン用）を動的確保する（その４）
c
c――――――――――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――――★ファイバー用履歴要素
      ALLOCATE (Bilinear_work(n))                                ! 50
      ALLOCATE (Trilinear_work(n))
      ALLOCATE (Concrete_work(n))
c――――――――――――――――――――――――――――――――――★全体剛性行列など
      ALLOCATE (gskym(Parameter_C.n_skyline))
      ALLOCATE (gskym_d(Parameter_C.n_unknown))
      ALLOCATE (nwork(Parameter_C.n_unknown))
      ALLOCATE (twork(Parameter_C.n_unknown))
      M_alloc(5)=1                                               ! 51
c――――――――――――――――――――――――――――――――――★部材両端中央応力のゼロセット(ok)
      call  Set_pointforce_zero()                                ! 52
c――――――――――――――――――――――――――――――――――★部材応力のゼロセット(ok)
      call  Set_stress_zero()                                    ! 53
c――――――――――――――――――――――――――――――――――★MSS モデルの初期設定
      n=Model_type.n_m_model(5)
       if(n.ne.0) then
      call Cal_MSS_dat()                                         ! 54
c――――――――――――――――――――――――――――――――――★平面問題における部材の拘束方向チェック(ok)
      call Check_R_direction()                                   ! 55
c――――――――――――――――――――――――――――――――――★部材の線形剛性計算(ok)
      call Cal_stiff_linear()                                    ! 56
c――――――――――――――――――――――――――――――――――★剛性の釣合座標系への変換(ok)
      call Rotate_stiffness()                                    ! 57
c――――――――――――――――――――――――――――――――――★部材の減衰行列計算(ok)
      if(Parameter_C.nc_member .ne. 0) then
      call Cal_damp_linear()                                     ! 58
c――――――――――――――――――――――――――――――――――★部材減衰行列の釣合座標系への変換(ok)
      call Rotate_damp()                                         ! 59
      end if
c――――――――――――――――――――――――――――――――――★節点集中質量セット(ok)
      call Set_mass()                                            ! 60
c――――――――――――――――――――――――――――――――――★節点荷重セット(ok)
      call Set_point_load()                                      ! 61
c――――――――――――――――――――――――――――――――――★接線剛性のコピー(ok)
      call Initset_nonlin_stiff()                                ! 62
c――――――――――――――――――――――――――――――――――★ベクトルのゼロセット
c――――――――――――――――――――――――――――――――――★増分前の変位、速度、加速度(ok)
      call Set_zero_v()                                          ! 63
```

```
c――――――――――――――――――――――――――――――★最大変位等のゼロセット(ok)
      call Clear_max_disp()                                                  ! 64
c――――――――――――――――――――――――――――――★最大、最小応力等のゼロセット(ok)
      call Clear_max_stress()                                                ! 65
c――――――――――――――――――――――――――――――★不釣合力のゼロセット(ok)
      call Set_zero_pointforce()                                             ! 66
c
c――――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★         静的解析結果等を取り込む
c               （初期変位、初期応力を入力）
c
c――――――――――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――――――――――★初期変位を入力
      if(Control.init_disp.ne.0) then
c      call Get_init_disp()                                                  ! 67
       endif
c――――――――――――――――――――――――――――――――――――――★初期応力を入力
      if(Control.init_stress.ne.0) then
c      call Get_init_stress()                                                ! 68
      endif
c――――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★         解析結果を出力するファイル群をオープンする(ok)
c
c――――――――――――――――――――――――――――――――――――――――――――――――
      call flcheck()                                                         ! 69
c――――――――――――――――――――――――――――――――――――――★出力指定した断面がファイバー要素
c――――――――――――――――――――――――――――――――――――――★かどうかチェック(ok)
      call out_section_check()                                               ! 70
c――――――――――――――――――――――――――――――――――――――★ファイル名一覧出力
c――――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★         描画用データのセット
c
c――――――――――――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――――――――――★変位
      if(i_read_disp .ne. 0) then
      call Set_preset_disp()                                                 ! 71
       endif
c――――――――――――――――――――――――――――――――――――――――――――――――
c
c     ★         解析ステップをセットする
c
c――――――――――――――――――――――――――――――――――――――――――――――――
       ns_step=1                                                             ! 72
       n_step=10
       d_max_v=0.
       id_max_v=0
       stimes=secnds(0.0)                        ! 解析時間を計る
c――――――――――――――――――――――――――――――――――――――★解析手法のセット
      N_implicit_method = 1  !陰解法は１回で反復法に戻る
      Iteration_method  = 1  !最初は反復法を使用
c――――――――――――――――――――――――――――――――――――――――――――――――
```

```
c
c    ★          予備計算はここで終了（GUI に戻る）
c
c―――――――――――――――――――――――――――――――――――――――
      return                                                    ! 73
```

予備計算部分について、処理の流れを説明しよう。予備計算は、データ入力部と予備計算部に分かれている。データ入力部は、制御データと構造解析用データの入力に分かれている。また、多くの部分で、動的領域の確保に当てられている。

1. 振動解析プログラムの入り口となっており、このサブルーチンの引数は、解析の制御とグラフィック用に使用するデータである。これらの領域確保は、C++で書かれた動的解析管理ルーチンで行われている。
2. この implicit 文が数値解析全般に用いられており、数値解析で使用する実数は全て倍精度とする。
3. パラメータ icontrol の値が１であると、実際の動的解析へ処理が移行する。
4. 同じく、パラメータ icontrol の値が 99 であると、動的に領域確保した構造体や配列を解放し、動的解析の後処理を行うコードへ処理が移行する。
5. 動的解析では、多くの動的領域を用いて解析を行っている。ただし、動的領域の確保は予備計算が進んで、大きさを決定するパラメータが求められるまで行うことができない。確保する前に、何らかのエラーがあって終了処理にプログラムが移行し、その領域を解放しようすると重大なエラーとなりプログラムがハングすることがある。これを避けるために M_alloc 配列でどこまで動的領域が確保されたかをチェックする。まずは、この配列をゼロセットし、動的領域確保された段階で１を代入する。
6. 解析の進行状況を出力するファイル DOUTPUT をオープンする。このファイルは SPACE の中から見ることができる。
7. SPACE がディスクの中の一次記憶領域である TEMP フォルダーに書き出したコントロールファイル名と動的解析種別を読み込む。そのファイル名は spacesys.xxx である。
8. コントロールファイル(***.ctl)を読み込み、そこに書かれている全ファイルをシステム内にセットする。ここで、***は任意のファイル名を表す。

9. SPACE の中の動的解析ダイアログで定義した解析制御用データを入力する。ここで、その仕様を以下に示す。最初の一行目の 7 は整数データの個数、次の 11 は実数データの個数を表す。各々のデータの持つ意味は、第 7 章のファイル管理で示す。

```
      7          11
   0        0        1        2       10        0        1
0.150000E+02    0.100000E-02    0.100000E+03    0.100000E+03    0.100000E+04
0.100000E+01    0.100000E-04    0.100000E+01    0.200000E+00    0.500000E+00
0.500000E+02
```

10. 動的解析データ用パラメータをファイルより読み込み、データを設定する。
11. 解析結果の出力を制御するファイルを読み込み、データを設定する。
12. 減衰に関するファイルを読み込み、データを設定する。
13. 制御情報の取得に失敗した場合は、ここで解析が終了する。
14. ここからの４つのサブルーチンでは、先に入力した制御情報を各構造体にセットする。このサブルーチンでは解析を制御するパラメータを設定している。構造体は Control であり、その内容は第 7 章を参照されたい。
15. ここでは、解析モデルで使用している部材モデルなどを設定する。構造体は Model_type である。
16. 数値解析で使用するニューマークβ法の係数を計算し、構造体に設定する。構造体は Newmark_P である。
17. 動的荷重に関するデータを構造体に設定する。構造体は Dynamic_load である。
18. このサブルーチン以後の３つのサブルーチンは、動的領域を確保するために、構造データなどを予備入力する。この予備入力 Get_parameters()によって、節点数、要素数、部材数などの構造用パラメータを得ることができる。
19. 地震加速度の予備入力を行い、地震データの大きさを得る。
20. 擬似的静的荷重として用いる節点荷重の大きさを、予備入力によって得る。
21. 上の予備入力で得たパラメータを利用して、ALLOCATE 文によって配列と構造体配列の動的領域を確保する。
22. ここまでが、動的領域確保その１であり、ここで M_alloc(1)に１を設定し、動的領域を確保したことを保障する。

23. 動的に領域確保した構造体配列などを用いて、ここでは、構造物に関するデータを入力する。
24. 構造体 Model_type 中の n_m_model(6)は Maxwell モデルを表しており、それをこの解析モデルが使用しているかどうかチェックしている。使用しておれば、25. を行う。
25. フィルターを使用する場合は、このサブルーチンで初期設定を行う。
26. ここから、28. まで、2 度目の動的領域を確保する。ここでは、解析モデルが各部材モデルを使用した場合、該当する動的領域を確保するように設定されている。使用されているかどうかは、構造体の成分 Model_type.n_m_damp にその部材モデル数がセットされている。以後、使用されているかどうかのチェックは省略されているが、全て同様の方法で各モデルに関する構造体配列が動的確保されている。
27. ALLOCATE 文で、減衰部材モデルに関する配列が動的に領域確保されている。
28. ここまでが、動的領域確保その２であり、ここで M_alloc(2)に１を設定し、動的領域を確保したことを保障する。
29. 地震加速度データをファイルより入力し、セットする。ここでも省略されているが、もしこの解析でこの地震加速度データを使用していればこのサブルーチンが呼ばれることになる。以後の入力用サブルーチンは全て同様である。
30. 擬似的静的荷重としての節点荷重をファイルより入力する。
31. 上記の節点荷重が時刻歴の中でどの様に変化するかを、SPACE のダイアログでユーザーが設定したデータを用いて、具体的に節点荷重の大きさの時刻暦を求める。
32. 解析モデルに初期不整があれば、ファイルより基本的な節点の初期変位を読み込み、次に SPACE のダイアログで設定した初期変位の倍率を掛け、節点の座標に加える。この操作をこのサブルーチンで行う。
33. 解析モデルがファイバー部材を使用しておれば、このサブルーチンでファイバーデータの予備入力を行い、動的領域確保のためのパラメータを取得する。
34. ファイバーに関する構造体配列の動的領域を確保する。
35. この動的確保を行ったことを保障するために、M_alloc(3)を１にセットする。

36. ファイバーデータを入力する。データを入力すると共に、部材データと突合せを行うなど多くの処理を行うため、後節で具体的にその処理を見ていくことにする。
37. 修正 RO モデルを使用する場合、ここでデータ入力を行う。
38. 質量データをこのサブルーチンによって入力する。
39. レーリー減衰を設定するパラメータを入力する。
40. データの入力が終了し、ここからは予備計算に入る。まず、ここでは部材の長さを計算する。
41. 部材モデルの初期設定を行う。
42. 部材座標系から全体座標系へ変換する変換行列を全部材について求める。
43. 節点に局所座標を使用する場合は、ここで、全体座標系から局所座標系に変換する変換行列を計算する。
44. 各部材の部材座標系から、局所座標系を考慮した釣合座標系への変換行列を全部材について求める。
45. 節点拘束データから節点拘束表を作成し、未知番号データを挿入する。
46. 未知数に関連する配列の動的領域確保を行う。
47. ここまでが、動的領域確保その４であり、ここで M_alloc(4)に１を設定し、動的領域を確保したことを保障する。
48. 部材両端の拘束表を作成する。
49. スカイライン行列の変換表を作成する。
50. ファイバー用の動的領域と剛性行列などの配列を動的確保する。
51. ここまでが、動的領域確保その５であり、ここで M_alloc(5)に１を設定し、動的領域を確保したことを保障する。
52. 部材両端と中央の応力をゼロセットする。
53. 部材の応力をゼロセットする。
54. MSS モデルを使用している場合は、ここで初期設定を行う。
55. 平面問題で解析を行う場合、ファイバー断面の y 軸と z 軸のどちらに曲げが生じるかをチェックする。
56. 全部材の線形剛性行列を計算し、配列に保存する。
57. 上で求めた線形の剛性行列を部材座標系から釣合座標系に変換し、配列に保存する。
58. 部材で減衰を使用するモデルに対して、その線形の減衰行列を作成する。
59. 上で求めた減衰行列を、部材座標系から釣合座標系に変換し、保

存する。
60. 節点に集中する質量を、未知数番号順に配列にセットし直す。
61. 全体座標系で入力した節点荷重を釣合座標系に変換し、未知数番号順に配列に保存する。
62. 初期設定として、線形の剛性行列を非線形の剛性行列にコピーする。
63. 増分前の変位、速度、加速度の各ベクトルをゼロセットする。
64. 最大変位、最大速度、最大加速度ベクトルをゼロセットする。
65. 各部材の応力の最大・最小ベクトルをゼロセットする。
66. 不釣合力ベクトルをゼロセットする。
67. 現在では使用不可であるが、部材の初期変位を読み込む。
68. 同じく、現在では使用不可であるが、部材の初期応力を読み込む。
69. ユーザーが指定した出力用ファイルのオープン処理と管理ファイルの初期設定を行う。
70. ユーザーが指定したファイバー断面データが実際にファイバー断面であるかどうかチェックし、そうでない場合は、出力しないように設定する。
71. 画面描画用のデータを初期設定する。
72. 次に解析を実行するための解析パラメータを初期設定する。
73. これで予備計算は終了し、このサブルーチンから抜ける。

4.7 振動解析の後処理

振動解析時では、多くの動的領域を確保しており、処理終了時にこの動的領域を解放しなければならない。動的領域を解放する場合、もっとも大切なことは、領域確保していない配列などを解放しないことである。これを行うと必ずシステムがハングアップすることになる。これを避けるために、SPACE では配列 M_alloc を用いて、動的領域確保を行ったか否かをチェックしている。

ここでは、計算終了時における後処理のプログラムコードを載せる。

```
c-------------------------------------------------------------
c
c    ★          応答解析終了処理
c
c-------------------------------------------------------------
 9998 continue
c-------------------------------------------------------------
c
```

```
c    ★        描画用データのセット
c
c―――――――――――――――――――――――――――――――――――――――
c――――――――――――――――――――――★変位
      if(i_read_disp .ne. 0) then
      call Set_preset_disp(1,n_point,past_disp_point,F_disp,Point,
     *                          rot_local,Parameter_C)
c     write(damp_out,*) ' Set_preset_disp ok'
      endif
c――――――――――――――――――――――★応力
      if(i_read_spring .ne. 0) then
      call Set_preset_spring(Member,Element,E_model6_real,
     *               M_model11,M_model12,M_model13,
     *               M_model15,M_model21,M_model22,
     *               n_member,F_fay,F_n_spring,F_my_spring,
     *               F_mz_spring,i_stat_spring)
c     write(damp_out,*) ' Set_preset_spring ok'
      endif
 9997 continue
c――――――――――――――――――――――――――★最大変位、速度、加速度の出力
      call Out_max_disp(Max_disp,Parameter_C.n_point,
     *                 Point,ifl(12),iflz(12))
c     write(damp_out,*) ' Out_max_disp ok'
c――――――――――――――――――――――――――★最大応力等の出力
      call Out_max_stress(Max_stress,Parameter_C.n_member,
     *                   ifl(16),iflz(16))
c     write(damp_out,*) ' Out_max_stress ok'
c――――――――――――――――――――――――――★ファイルのクローズ
      do i=1,16
      if(ifl(i).eq.1) close(iflz(i))
      enddo
      write(damp_out,*) ' ファイルのクローズ ok'
c――――――――――――――――――――――――――★ファイルのタイムスタンプ
      ihan = 0
      NFILE=100
      call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
      call fltime(ifl,ifly,N_analysis,iflout)
c      write(damp_out,*) ' ファイルのタイムスタンプ ok' ,FNX_file
      ihan = 1
      if(iflout.eq.1) call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
c     write(damp_out,*) ' データセット ok'
c―――――――――――――――――――――――――――――――――――――――
c
c    ★        動的配列の解放
c
c―――――――――――――――――――――――――――――――――――――――
c――――――――――★        配列の動的領域を解放する（その１）
      if(M_alloc(1).eq.1)then
      DEALLOCATE (Point,Element,Member,Max_disp,Max_stress)       !ok
      DEALLOCATE (fll_static_point,am_point,fll_force_point)      !ok
      DEALLOCATE (am_member,rot_memb_t,rot_memb,ak_linear,ak_nonlinear)      !ok
      N=Parameter_C.n_local_coord
      if(N.ne.0) DEALLOCATE (rot_local)       !ok
      endif
```

```
c－－－－－－－－－－★　　配列の動的領域を解放する（その２）
      if(M_alloc(2).eq.1)then
      N=Model_type.n_m_damp
      if(N.ne.0) DEALLOCATE (ac_member)       !ok
      endif
c－－－－－－－－－－★　　配列の動的領域を解放する（その４）
      if(M_alloc(4).eq.1)then
      DEALLOCATE (disp_point,vel_point,acc_point,
     *     est_disp_point,est_vel_point,est_ddisp_point)
      DEALLOCATE (result_disp_point,result_vel_point,result_acc_point,
     *   past_disp_point,past_vel_point,past_acc_point,
     *   past_dacc_point)
      DEALLOCATE (ld_point,ld_point_repeat,fld_static)
      DEALLOCATE (a_vector,b_vector)
      DEALLOCATE (max_h_sky)
      endif
c－－－－－－－－－－★　　配列の動的領域を解放する（その５）
      if( M_alloc(5).eq.1)then
      DEALLOCATE (gskym,gskym_d,nwork,twork)        !ok
      endif
c－－－－－－－－－－★　　配列の動的領域を解放する（その２）
      if(  M_alloc(2).eq.1)then
      n=Model_type.n_m_ro_model
      if(n.ne.0) then
      DEALLOCATE (RO_work )
      endif
      n=Model_type.n_m_model(5)
      if(n.ne.0) then
      DEALLOCATE (MSS_work )            !ok
      endif
      n=Model_type.n_m_model(6)
      if(n.ne.0) then
      DEALLOCATE (E_model6_real )           !ok
      endif
      n= Model_type.n_e_model(11)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model11 )               !ok
      endif
      n= Model_type.n_m_model(11)        !部材モデルの数
      if(n.ne.0) then
      DEALLOCATE ( M_model11 )              !ok
      endif
      n= Model_type.n_e_model(12)        !要素モデルの数
      if(n.ne.0) then
      DEALLOCATE (E_model12 )              !ok
      endif
      n= Model_type.n_m_model(12)        !部材モデルの数
     if(n.ne.0) then
      DEALLOCATE ( M_model12 )              !ok
      endif
      n= Model_type.n_m_model(18)        !部材モデルの数
     if(n.ne.0) then
      DEALLOCATE ( M_model13 )              !ok
      endif
```

```
 n= Model_type.n_e_model(18)        !要素モデルの数
if(n.ne.0) then
 DEALLOCATE (E_model13 )              !ok
 endif
 n= Model_type.n_e_model(15)        !要素モデルの数
if(n.ne.0) then
 DEALLOCATE (E_model15 )              !ok
 endif
 n= Model_type.n_m_model(15)        !部材モデルの数
if(n.ne.0) then
 DEALLOCATE ( M_model15 )              !ok
 endif
 n= Model_type.n_e_model(13)        !要素モデルの数
if(n.ne.0) then
 DEALLOCATE (E_model21 )              !ok
 endif
 n= Model_type.n_m_model(13)        !部材モデルの数
if(n.ne.0) then
  DEALLOCATE ( M_model21 )              !ok
 endif
 n= Model_type.n_e_model(14)        !要素モデルの数
if(n.ne.0) then
 DEALLOCATE (E_model22 )              !ok
 endif
 n= Model_type.n_m_model(14)        !部材モデルの数
if(n.ne.0) then
 DEALLOCATE ( M_model22 )              !ok
 endif
 n= Model_type.n_e_model(16)        !要素モデルの数
if(n.ne.0) then
 DEALLOCATE (E_model31)               !ok
 endif
 n= Model_type.n_m_model(16)        !部材モデルの数
if(n.ne.0) then
 DEALLOCATE ( M_model31 )              !ok
 endif
 n= Model_type.n_e_model(17)        !要素モデルの数
if(n.ne.0) then
 DEALLOCATE (E_model32)               !ok
 endif
 n= Model_type.n_m_model(17)        !部材モデルの数
if(n.ne.0) then
 DEALLOCATE ( M_model32 )              !ok
 endif
 n= Model_type.n_e_model(19)        !要素モデルの数
if(n.ne.0) then
 DEALLOCATE (E_model33)               !ok
 endif
 n= Model_type.n_m_model(19)        !部材モデルの数
if(n.ne.0) then
 DEALLOCATE ( M_model33 )              !ok
 endif
 if(Dynamic_load.n_load_dynamic .ne. 0) then
 DEALLOCATE (acc_earth)               !ok
```

```
      endif
       if(Dynamic_load.n_load_point .ne. 0) then
       DEALLOCATE (fdd_point)              !ok
       endif
       endif
c――――――――――★        配列の動的領域を解放する（その３）
       if( M_alloc(3).eq.1)then
       n= Model_type.nm_div_fmodel         !要素モデル内のサブ要素の数
      if(n.ne.0) then
       DEALLOCATE (M_model_fiber )              !ok
       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
       DEALLOCATE (E_model_fiber )              !ok
       endif
       endif
c――――――――――★        配列の動的領域を解放する（その５）
       if( M_alloc(5).eq.1)then
       n= Model_type.n_m_bilinear          !  バイリニアの履歴要の数
      if(n.ne.0) then
       DEALLOCATE (Bilinear_work )              !ok
       endif
       n= Model_type.n_m_trilinear         !  トリリニアの履歴要の数
      if(n.ne.0) then
       DEALLOCATE (Trilinear_work )              !ok
       endif
       n= Model_type.n_m_Concrete          !  コンクリートの履歴要の数
      if(n.ne.0) then
       DEALLOCATE (Concrete_work )              !ok
       endif
       endif
c――――――――――――――――――――――――――――――★計算終了コードセット
      iend_code =1
       close (damp_out)
       return
c――――――――――――――――――――――――――――――――――――――――
c
c    ★        解析終了
c
c――――――――――――――――――――――――――――――――――――――――
     end
```

ここでは、後処理の流れについて説明する。後処理は、変位や速度、応力などの最大値を該当するファイルに出力すること、また、動的に確保した領域を解放することである。プログラムの骨組みを取り出してみよう。

```
c――――――――――――――――――――――――――――――――――――――――
c
c    ★        応答解析終了処理
c
c――――――――――――――――――――――――――――――――――――――――
```

```
 9998 continue                                                          ! 1
c-------------------------------------------------------------
c
c    ★        描画用データのセット
c
c-------------------------------------------------------------
c-------------------------★変位
      if(i_read_disp .ne. 0) then
      call Set_preset_disp()                                            ! 2
      endif
c-------------------------★応力
      if(i_read_spring .ne. 0) then
      call Set_preset_spring()                                          ! 3
      endif
 9997 continue
c-------------------------------★最大変位、速度、加速度の出力
      call Out_max_disp()                                               ! 4
c-------------------------------★最大応力等の出力
      call Out_max_stress()                                             ! 5
c-------------------------------★ファイルのクローズ
      do i=1,16
      if(ifl(i).eq.1) close(iflz(i))                                    ! 6
      enddo
c-------------------------------★ファイルのタイムスタンプ
      ihan = 0
      NFILE=100
      call ctlset()                                                     ! 7
      call fltime()                                                     ! 8
      if(iflout.eq.1) call ctlset()                                     ! 9
c-------------------------------------------------------------
c
c    ★        動的配列の解放
c
c-------------------------------------------------------------
c----------★        配列の動的領域を解放する（その１）
      if(M_alloc(1).eq.1)then                                           ! 10
      DEALLOCATE (Point,Element,Member,Max_disp,Max_stress)       !ok
      DEALLOCATE (fll_static_point,am_point,fll_force_point)       !ok
      DEALLOCATE (am_member,rot_memb_t,rot_memb,ak_linear,ak_nonlinear)       !ok
       N=Parameter_C.n_local_coord
      if(N.ne.0) DEALLOCATE (rot_local)       !ok
      endif
c----------★        配列の動的領域を解放する（その２）
      if(M_alloc(2).eq.1)then                                           ! 11
      N=Model_type.n_m_damp
      if(N.ne.0) DEALLOCATE (ac_member)       !ok
      endif
       ・
       ・
       ・
c-------------------------★計算終了コードセット
      iend_code =1
      close (damp_out)                                                  ! 12
      return
```

```
c―――――――――――――――――――――――――――――――――――――――
c
c    ★          解析終了
c
c―――――――――――――――――――――――――――――――――――――――
      end
```

後処理の流れについて、プログラムコードの右側の番号にしたがって説明する。

１．ここが後処理の入り口となっており、icontrol が 99 の場合ここに処理が移動する。
２．画面に構造図などを描画指定している場合、ここで、変位の最終設定を行う。
３．同じく、応力の最終設定を行う。
４．最大変位、最大速度、最大加速度をファイルに出力する。
５．部材の最大応力をファイルに出力する。
６．出力ファイルの管理データに従って、ファイルをクローズする。
７．コントロールファイルを読み込む。
８．データを出力したファイルに対して、日時をコントロールデータにセットする。
９．そのコントロールデータをファイルに再度出力する。
10．ここから、解析の中で動的に確保した領域を解放する。まず、M_alloc(1)で管理された構造体や配列を解放する。
11. 同じく、M_alloc(2)で管理された構造体や配列を解放する。以後、同様の処理を行って、構造体や配列を解放する。
12. 計算終了コードを１に設定し、DOUTPUT ファイルをクローズする。これで解析の後処理が終了し、サブルーチンから戻ることになる。

4.8 固有値問題の処理

本節では、固有値問題を数値解析し、固有振動数や振動モード、また刺激係数を求めるプログラムの流れを学ぶ。計算の流れは非常に単純であり、第 4.2 節で説明されているため、おおよそ理解できていると思う。ここでは、実際の FORTRAN コードを用いて説明する。前述したように、ここでは固有値問題を解く計算手法として 2 種類用意されており、また、計算した結果が振動解析のレーリー減衰で利用されていることを頭に入れておいて頂きたい。ここでは、データ入力と予備計算部、及び、後

処理部を除いたプログラムコードを以下に示す。省いた部分のコードは付録を参照されたい。

```
c―――――――――――――――――――――――――――――――――――――――――
c
c    ★         固有値解析手法の選択
c
c―――――――――――――――――――――――――――――――――――――――――
      if(Eigen_d.n_method .eq. 1) goto 9700
c―――――――――――――――――――――――――――――――――――――――――
c
c    ★         サブスペース法の計算用配列を動的確保する
c
c―――――――――――――――――――――――――――――――――――――――――
      n_member=Parameter_C.n_member
      n_point =Parameter_C.n_point
      n_unknown=Parameter_C.n_unknown
      n_skyline=Parameter_C.n_skyline
      n_local_coord=Parameter_C.n_local_coord
      nroot = Eigen_d.n_modes
      nc = min(2*nroot,nroot+8)
      nnc = nc*(nc+1)/2
      write(76,*)   'n_unknown:',n_unknown
      write(76,*)   'n_skyline:',n_skyline
      write(76,*)   'nroot:',nroot
      write(76,*)   'nc:',nc
      write(76,*)   'nnc:',nnc
      write(76,*)   'method_type',Eigen_d.method_type
      write(76,*)   'Eigen_d.epsj',Eigen_d.epsj
      write(76,*)   'Eigen_d.n_iteration',Eigen_d.n_iteration
      if(nroot.eq.0) then
      ierr_dat=1000
      return
      endif
c―――――――――――――――――――――――――――――――――――――――――
      ALLOCATE (tw_a(n_unknown),tw_b(n_unknown),tw_c(n_unknown))
      ALLOCATE (Eigen_Value(nc),Eigen_Vector(n_unknown,nc))
      ALLOCATE (ar_a(nnc),ar_b(nnc),Omega(nnc))
      ALLOCATE (vec_w1(nc,nc),vec_w2(nc),vec_w3(nc),vec_w4(nc),
     *          vec_w5(nc),vec_w6(nc))
      ALLOCATE (ivec_w1(nc))
c―――――――――――――――――――――――――――――――――――――――――
      do n_step = 1,2
c――――――――――――――――――――――――――★スカイライン行列のゼロセット(ok)
      call Set_sky_zero(gskym,n_skyline)
      call Set_sky_zero(gskymm,n_skyline)
c      write(damp_out,*) ' Set_sky_zero ok'
c――――――――――――――――――――――――――★集中質量系の行列への足し込み
      call Build_sky_mm_E(n_step,gskymm,
     *           Point,n_point,am_point,rot_local,
     *           n_local_coord,max_h_sky,Eigen_d.load_mass)
c       write(damp_out,*) ' Build_sky_mm ok'
c     write(76,'(12e12.4)')(gskymm(j),j=1,n_unknown)
```

```
c－－－－－－－－－－－－－－－－－－－－－－★部材の整合質量系の行列への足し込み(ok)
c－－－－－－－－－－－－－－－－－－－★部材の整合質量行列計算
c        write(damp_out,*) ' Cal_mass_linear ok',Eigen_d.load_mass
      if(Eigen_d.load_mass .ne. 0) then
      call Cal_mass_linear(n_step,Element,Member,Parameter_C,am_member,
     *            work1_elememt,work2_elememt,work1_member,work2_member,
     *            Eigen_d.load_mass)
c        write(damp_out,*) ' Cal_mass_linear ok'
c－－－－－－－－－－－－－－－－－－－★整合質量の釣合座標系への変換(ok)
      call Rotate_mass(n_step,Element,Member,n_member,am_member,
     *                 rot_memb,Eigen_d.load_mass)
c        write(damp_out,*) ' Rotate_mass ok',n_member
c－－－－－－－－－－－－－－－－－－－★部材質量系の足し込み(ok)
      call Build_sky_m_E(n_step,gskymm,n_skyline,Member,n_member,
     *                  am_member,max_h_sky,Element)
      endif
c       write(damp_out,*) ' Build_sky_m ok',n_member
c－－－－－－－－－－－－－－－－－－－－－－★線形剛性の足し込み(ok)
      call Build_sky_k_E(gskym,n_skyline,Member,n_member,
     *                        Ak_linear,max_h_sky)
c      return
c       write(damp_out,*) ' Build_sky_k ok',n_member
c－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
c
c    ★        サブスペース法の計算用データをセットする
c
c－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
      nroot = Eigen_d.n_modes
      nc = min(2*nroot,nroot+8)
      nnc = nc*(nc+1)/2
      nnm=Parameter_C.n_unknown+1
      IFSS=0          !Ｓｔｕｒｍ列による検定のためのフラグ ０ ： 検定しない
      IFPR=0          !反復の間にプリントするためのフラグ ０ ： プリントしない
      if(Eigen_d.load_mass .ne. 0) then
      nm_skyline=n_skyline
      else
      nm_skyline=n_unknown
      endif
c－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
c
c    ★        固有問題の解析（サブスペース法）
c
c－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－－
c      write(damp_out,*) ' SSPACE in ok',n_member

      call SSPACE(gskym,gskymm,max_h_sky,Eigen_Value,Eigen_Vector,
     *         tw_a,tw_b,ar_a,ar_b,vec_w1,vec_w2,vec_w3,vec_w4,vec_w5,
     *         vec_w6,ivec_w1,n_unknown,nnm,n_skyline,nm_skyline,nroot,
     *         Eigen_d.epsj,nc,nnc,Eigen_d.n_iteration,
     *         IFSS,IFPR,Eigen_d.method_type,IEXIT,
     *         Eigen_d.load_mass,NSTIF,gskym_d,nwork,twork,tw_c)
      close(UNIT=NSTIF,STATUS='DELETE')
      if(iexit.ne.0) then
      ierr_dat=998
```

```
      return
      endif
c-----------------------★結果の出力
      call Out_Eigen(n_step,n_unknown,Parameter_C,Eigen_d,
     *           Point,Newmark_P,Eigen_Value,Eigen_Vector,Omega,
     *           gskymm,max_h_sky,rot_local,disp_point_m,
     *           tw_a,tw_b,ifl(7),iflz(7),ifl(8),iflz(8))
      enddo
c-----------------------★計算終了コードセット
      iend_code =1
c-----------------------★ファイルのクローズ
      do i=1,16
      if(ifl(i).eq.1) close(iflz(i))
      enddo
      write(damp_out,*) ' ファイルのクローズ ok'
c-----------------------★ファイルのタイムスタンプ
      ihan = 0
      NFILE=100
      call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
      call fltime(ifl,ifly,N_analysis,iflout)
      write(damp_out,*) ' ファイルのタイムスタンプ ok' ,FNX_file
      ihan = 1
      if(iflout.eq.1) call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
c      write(damp_out,*) ' データセット ok'
c-------------------------------------------------------------
c
c    ★        サブスペース用動的配列の解放
c
c-------------------------------------------------------------
      .
      .
      close (damp_out)
      return
c-------------------------------------------------------------
c
c    ★        せん断型の固有値解析
c
c-------------------------------------------------------------
 9700 continue
c-------------------------------------------------------------
c
c    ★        ヤコビ法の計算用配列を動的確保する
c
c-------------------------------------------------------------
      n_member=Parameter_C.n_member
      n_point =Parameter_C.n_point
      n_unknown=Parameter_C.n_unknown
      n_local_coord=Parameter_C.n_local_coord
      nc=n_unknown
      write(76,*)   'n_unknown:',n_unknown
      write(76,*)   'Eigen_d.epsj',Eigen_d.epsj
      write(76,*)   'Eigen_d.n_iteration',Eigen_d.n_iteration
c-----------------------------------------------------------
      ALLOCATE (gak(n_unknown,n_unknown))
```

```
      ALLOCATE (Eigen_Value(n_unknown),
     *          Eigen_Vector(n_unknown,n_unknown))
      ALLOCATE (Omega(nc))
      ALLOCATE (vec_w5(nc),vec_w2(nc),vec_w3(nc),vec_w4(nc))
c-------------------------------------------------------
      do n_step = 1,2
c-----------------------★剛性行列のゼロセット(ok)
      n=n_unknown*n_unknown
      call Set_sky_zero(gak,n)
      call Set_sky_zero(gskymm,n_unknown)
c      write(damp_out,*) ' Set_gak_zero ok'
c-----------------------★線形剛性の足し込み(ok)
      call Build_gak_E(gak,n_unknown,Member,n_member,
     *                       Ak_linear)
c       write(damp_out,*) ' Build_gak ok',n_member
c-----------------------★集中質量系の行列への足し込み
      call Build_mm_E(n_step,gskymm,
     *           Point,n_point,am_point)
c       write(damp_out,*) ' Build_mm ok'
c-----------------------★固有値問題の変換
      call Trans_Eigen(gak,n_unknown,gskymm,vec_w2,ier)
      if(ier.ne.0) then
      write(damp_out,*) ' err: 節点にゼロ質量があった。'
      return
      endif
c       write(damp_out,*) ' Trans_Eigen ok'
c-------------------------------------------------------
c
c    ★        ヤコビ法の計算用データをセットする
c
c-------------------------------------------------------
c-------------------------------------------------------
c
c    ★        固有問題の解析（ヤコビ法）
c
c-------------------------------------------------------
c      write(damp_out,*) ' Jacobi in ok',n_member
      call Jacobi_E(gak,n_unknown,Eigen_Value,Eigen_Vector,
     *         vec_w5,vec_w2,vec_w3,vec_w4)
c       write(damp_out,*) ' Jacobi out ok',n_member
c-----------------------★固有ベクトルの変換
      call Trans_Eigen_V(Eigen_Vector,n_unknown,gskymm,vec_w2,ier)
c-----------------------★結果の出力
      call Out_Eigen_J(n_step,n_unknown,Parameter_C,Eigen_d,
     *           Point,Newmark_P,Eigen_Value,Eigen_Vector,Omega,
     *           gskymm,rot_local,disp_point_m,
     *           vec_w3,vec_w4,ifl(7),iflz(7),ifl(8),iflz(8))
c       write(damp_out,*) ' Out_Eigen_J ok',n_member
      enddo
c-----------------------★計算終了コードセット
      iend_code =1
c-----------------------★ファイルのクローズ
      do i=1,16
      if(ifl(i).eq.1) close(iflz(i))
```

```
      enddo
      write(damp_out,*) ' ファイルのクローズ ok'
c-----------------------------★ファイルのタイムスタンプ
      ihan = 0
      NFILE=100
      call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
      call fltime(ifl,ifly,N_analysis,iflout)
      write(damp_out,*) ' ファイルのタイムスタンプ ok' ,FNX_file
      ihan = 1
      if(iflout.eq.1) call ctlset(ihan,FNX_file,TITLEX,IDFILE,NFILE)
c      write(damp_out,*) ' データセット ok'
c-------------------------------------------------------------
c
c    ★      ヤコビ法用動的配列の解放
c
c-------------------------------------------------------------
      .
      .
      close (damp_out)
      return
      end
```

続いて、固有値問題に関する計算の流れを、プログラムの骨組みを取り出して詳しく見ていこう。

```
c-------------------------------------------------------------
c
c    ★         固有値解析手法の選択
c
c-------------------------------------------------------------
      if(Eigen_d.n_method .eq. 1) goto 9700                    ! 1
c-------------------------------------------------------------
c
c    ★         サブスペース法の計算用配列を動的確保する
c
c-------------------------------------------------------------
c              サブスペース法で使用するワークエリアを動的確保する
c-------------------------------------------------------------
      ALLOCATE (tw_a(n_unknown),tw_b(n_unknown),tw_c(n_unknown))  ! 2
      ALLOCATE (Eigen_Value(nc),Eigen_Vector(n_unknown,nc))
      ALLOCATE (ar_a(nnc),ar_b(nnc),Omega(nnc))
      ALLOCATE (vec_w1(nc,nc),vec_w2(nc),vec_w3(nc),vec_w4(nc),
     *          vec_w5(nc),vec_w6(nc))
      ALLOCATE (ivec_w1(nc))
c-------------------------------------------------------------
      do n_step = 1,2                                          ! 3
c-----------------------------★スカイライン行列のゼロセット(ok)
      call Set_sky_zero(gskym)                                 ! 4
      call Set_sky_zero(gskymm)
c-----------------------------★集中質量系の行列への足し込み
      call Build_sky_mm_E()                                    ! 5
c-----------------------------★部材の整合質量系の行列への足し込み(ok)
c-----------------------------★部材の整合質量行列計算
```

```
      if(Eigen_d.load_mass .ne. 0) then
      call Cal_mass_linear()                                          ! 6
c-----------------------★整合質量の釣合座標系への変換(ok)
      call Rotate_mass()                                              ! 7
c-----------------------★部材質量系の足し込み(ok)
      call Build_sky_m_E()                                            ! 8
      endif
c---------------------------★線形剛性の足し込み(ok)
      call Build_sky_k_E()                                            ! 9
c-------------------------------------------------------------------
c
c    ★        固有問題の解析（サブスペース法）
c
c-------------------------------------------------------------------
      call SSPACE()                                                   ! 10
c-------------------------★結果の出力
      call Out_Eigen()                                                ! 11
      enddo
c-------------------------------------------------------------------
c
c    ★        サブスペース用動的配列の解放
c
c-------------------------------------------------------------------
      DEALLOCATE (Point,Element,Member)                               ! 12
      DEALLOCATE (gskym,gskym_d,nwork,twork,gskymm)        !ok
      DEALLOCATE (max_h_sky)                               !ok
      DEALLOCATE (tw_a,tw_b,tw_c)                          !ok
       .
       .
      return
c-------------------------------------------------------------------
c
c    ★        せん断型の固有値解析
c
c-------------------------------------------------------------------
 9700 continue
c-------------------------------------------------------------------
c
c    ★        ヤコビ法の計算用配列を動的確保する
c
c-------------------------------------------------------------------
      ALLOCATE (gak(n_unknown,n_unknown))                             ! 13
      ALLOCATE (Eigen_Value(n_unknown),
     *          Eigen_Vector(n_unknown,n_unknown))
      ALLOCATE (Omega(nc))
      ALLOCATE (vec_w5(nc),vec_w2(nc),vec_w3(nc),vec_w4(nc))
c-----------------------------------------------------------------
      do n_step = 1,2                                                 ! 14
c-----------------------★剛性行列のゼロセット(ok)
      call Set_sky_zero(gak)                                          ! 15
      call Set_sky_zero(gskymm)
c---------------------------★線形剛性の足し込み(ok)
      call Build_gak_E()                                              ! 16
c-----------------------★集中質量系の行列への足し込み
```

```
      call Build_mm_E()                                          ! 17
c―――――――――――――――――――――――――★固有値問題の変換
      call Trans_Eigen()                                         ! 18
c――――――――――――――――――――――――――――――――――――――――
c
c    ★          固有問題の解析（ヤコビ法）
c
c――――――――――――――――――――――――――――――――――――――――
      call Jacobi_E()                                            ! 19
c―――――――――――――――――――――――――★固有ベクトルの変換
      call Trans_Eigen_V()                                       ! 20
c―――――――――――――――――――――――――★結果の出力
      call Out_Eigen_J()                                         ! 21
      enddo
c――――――――――――――――――――――――――――――――――――――――
c
c    ★          ヤコビ法用動的配列の解放
c
c――――――――――――――――――――――――――――――――――――――――

      DEALLOCATE (Point,Element,Member)                  !ok     ! 22
      DEALLOCATE (gskym,gskym_d,nwork,twork,gskymm)      !ok
      DEALLOCATE (gak)          !ok
        •
        •
     return
      end
```

プログラムの右側には番号が振られており、その番号にしたがって処理の内容を概説する。

1. データ入力した後、計算手法によって分岐する。ここでは、ヤコビ法を選択した場合、文番号 9700 に飛ぶ。
2. これ以後は、サブスペース法で数値解析を行う。ここでは、サブスペース法で必要になるワーク領域を動的確保している。
3. 解析が 2 段階で行われるため、固有値解析も 2 度行われる。
4. スカイライン行列である質量行列 gskymm と剛性行列 gskym をゼロクリアする。
5. 以下の処理で質量行列を作成する。まず、節点集中質量を行列に組み込む。
6. 部材分布質量がある場合は、次の 7 と 8 も処理する。ここでは、部材座標系で質量行列を計算する。
7. 質量行列を部材座標系から釣合座標系に変換する。
8. スカイライン行列である質量行列に、部材の整合質量を組み込む。
9. 部材の線形剛性をスカイライン行列に組み込む。

10. 固有値問題をサブスペース法で解析する。
11. 振動モード、振動数、固有周期などを求め、仕様に合わせてファイルに出力する。
12. 解析の中で動的確保した領域を解放する。
13. ここからヤコビ法を用いた解析を行う。まず、ワーク領域を動的確保する。
14. 解析が 2 段階で行われるため、固有値解析も 2 度行われることになる。
15. 正方行列である質量行列 gskymm と剛性行列 gskym をゼロクリアする。
16. 部材の線形剛性を正方行列に組み込む。
17. 質量行列を作成する。ここでは節点集中型のみ扱う。節点集中質量を正方行列に組み込む。
18. 一般固有値問題から標準固有値問題に変換する。
19. ヤコビ法を用いて固有値問題を解く。
20. 固有ベクトルを変換する。
21. 振動モード、振動数、固有周期などを求め、仕様に合わせてファイルに出力する。
22. 解析の中で動的確保した領域を解放する。

以上で、固有値問題を解析する流れを説明した。固有値問題を数値解析するサブスペース法 SSPACE()とヤコビ法 Jacobi_E()については、参考文献を参照されたい[3)]。また、他のサブルーチンは、他の節で説明されている。ここでは、省略するが、具体的なコードは付録を見られたい。