第3章　粒子要素法シミュレーション

3.1　シミュレーションモデル

いま，図3.1に示す粒子群が矢印の方向に流動しているものとしよう．図3.2に示すようにこの流動粒子群の中の1個の粒子に着目すると，その着目粒子の運動は周りの接触粒子から受ける力によるものである．そこで，周りの粒子から受ける接触力 $\boldsymbol{F}$ を刻々知ることができるなら，質量 m をもつ着目粒子の運動はニュートンの運動の第2法則（$\boldsymbol{F}=m\boldsymbol{a}$）にもとづく運動方程式で表される．このとき $\boldsymbol{F}$ と m はわかっているから，運動方程式を解くことができ，粒子の加速度 $\boldsymbol{a}$ が得られる．それを時間で積分すると粒子の速度 v が，さらに積分すると変位 u が得られ，流動する粒子の運動軌跡を計算できる．この方法で流動するすべての粒子の運動軌跡を得るなら，それが粒子群の流動挙動を表すことになる[1]．

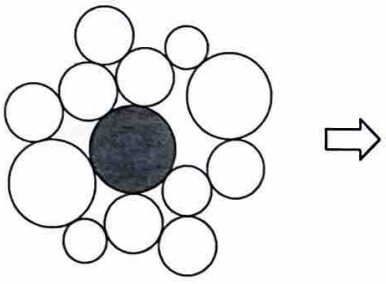


図 3.1　流動粒子群

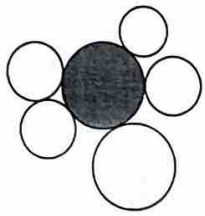


図 3.2　着目粒子に接触する隣接粒子

この方法による粒子群流動挙動のシミュレーションで，まず問題になるのは粒子間接触力を知ることである．通常の流動粒子群は近接粒子と衝突や摩擦を繰り返しながら流動している．そのときの粒子の接触は弾性的であったり，非弾性的であったりするだろう．そこで粒子要素法では，この接触力を図3.3に示すようなフォークトモデル（Voigt model）で表現する．この接触力はいろいろな方向に作用するので，計算の便宜上，接触2粒子の中心方向（法線方向）成分とそれに垂直なせん断方向成分に分け，それぞれの成分にフォークトモデルを適用する．このとき粉体粒子で大切な摩擦相互作用が考慮できるように，せん断方向成分には摩擦スライダーが挿入されている．

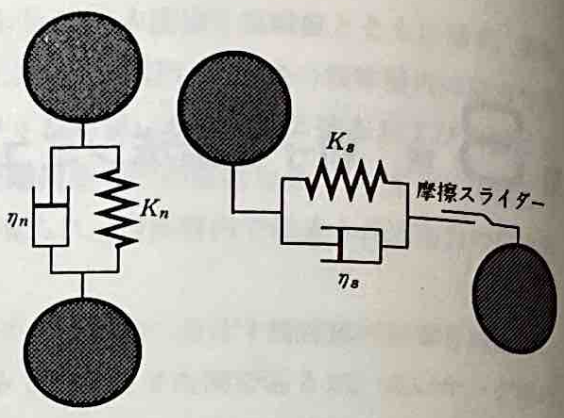
この質量 m をもつ2粒子間に働く力 $\boldsymbol{F}$ による接触粒子の並進変位 u と回転変位 ϕ は，バネ

の弾性定数を K，ダッシュポットの粘性係数を η として次式で表される．ただし，添え字 n，s はそれぞれ法線およびせん断方向を表す．

$$m\frac{du^2}{dt^2}+\eta\frac{du}{dt}+Ku=0 \qquad (3.1\,\mathrm{a})$$

$$I\frac{d\psi^2}{dt^2}+\eta r^2\frac{d\psi}{dt}+Kr^2\psi=0 \qquad (3.1\,\mathrm{b})$$

ここで，I は慣性モーメント（$=\rho_p\pi r^4/2$）を示し，ρ_p は粒子密度である．



(a)　法線方向　　(b)　せん断方向

図 3.3　粒子間相互作用力を表すフォークトモデル

一般の粉体では，1個の粒子の周りに多数の粒子が接触（図3.2）しているので，それら個々の接触点に対して式（3.1）が成立し，着目粒子の運動を知るのに接触点数と同じ数の方程式を連立して解かなければならない．この困難を避けるために，実際の計算では，例えば式（3.1 a）を次式のように時間増分 Δt で差分近似して計算する．

$$m[u]_t=-\eta[u]_{t-\Delta t}-K[u]_{t-\Delta t} \qquad (3.2)$$

したがって，新しい時刻 t における加速度 $[u]_t$ は，時刻 Δt だけ前の変位と加速度から得られる．新しい時刻 t における加速度 $[u]_t$ を数値積分すると，時刻 t における変位と加速度が得られる．この計算を時間刻み Δt で繰り返すことによって粒子の運動軌跡が計算できる．詳しい計算法を木山ら[2]らが示した方法にそって次に述べる．

3.1.1　粒子間の相対変位増分

いま，粒子 i が時刻 $t-\Delta t$ から時刻 t までの微小時間 Δt の間に図3.4に示す位置 A (x_i, y_i) から位置 B $(x_i+\Delta u_i,\ y_i+\Delta v_i)$ に変位するものとする．このとき変位増分の x，y 成分をそれ

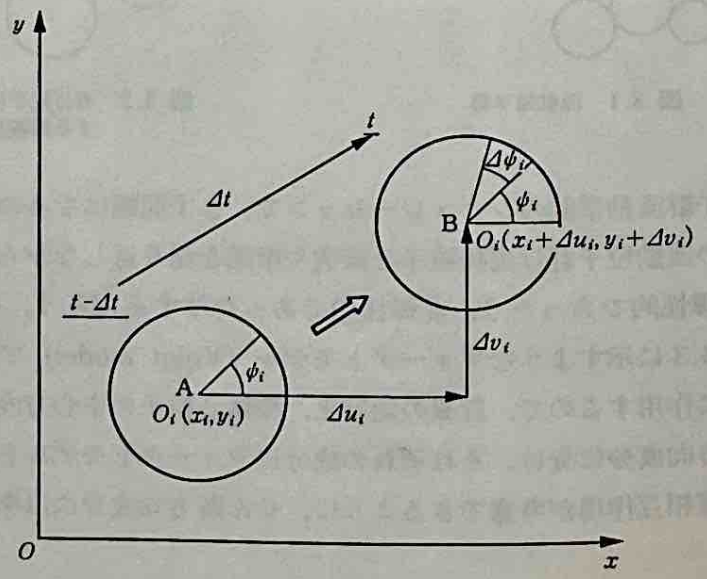


図 3.4　着目粒子の変位量

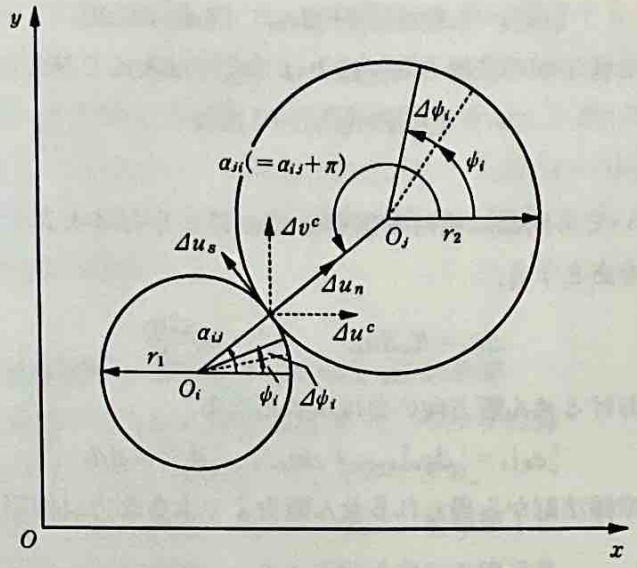


図 3.5 2粒子の接触

ぞれ Δu_i, Δv_i とし，その変位にともなう粒子の回転変位増分を $\Delta\phi_i$ で表すことにしよう．

粒子半径がそれぞれ r_i, r_j である2つの球形粒子 i, j は次の式 (3.3) で表される条件を満足するとき接触する．粒子要素法では，粒子は剛体で変形しないが，図3.5のように互いに重なり合うことができるものとする．

$$r_i+r_j \geq L_{ij} \tag{3.3}$$

L_{ij} は粒子 i と j の中心間距離であり，$L_{ij}=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2}$ で表される．また，i,j 粒子の接触点の位置を識別するために，i, j 粒子の中心を結ぶ共通法線が x 軸となす角 α_{ij} を用いる．この角度 α_{ij} と粒子の座標との関係は次式で表される．

$$\sin\alpha_{ij}=-(y_i-y_j)/L_{ij}, \quad \cos\alpha_{ij}=-(x_i-x_j)/L_{ij} \tag{3.4}$$

接触する2粒子 i, j の微小時間 Δt における相対変位量の法線方向ならびにせん断方向成分は，角度 α_{ij} を用いてそれぞれ次式で与えられる．

$$\begin{aligned}\Delta u_n&=(\Delta u_i-\Delta u_j)\cos\alpha_{ij}+(\Delta v_i-\Delta v_j)\sin\alpha_{ij}\\ \Delta u_s&=-(\Delta u_i-\Delta u_j)\sin\alpha_{ij}+(\Delta v_i-\Delta v_j)\cos\alpha_{ij}+(r_i\Delta\phi_i+r_j\Delta\phi_j)\end{aligned} \tag{3.5}$$

3.1.2 粒子間作用力

粒子間に作用する力はいろいろな方向に作用するので，法線方向成分 f_n とせん断方向成分 f_s に分けて考える．

(a) 法線方向成分

時刻 t における作用力は，時刻 $t-\Delta t$ の作用力に時間 Δt の間の粒子 i,j の相対変位増分 Δu_n による力を加えたものである．バネとダッシュポットは並列に配置されているので，相対変位増分 Δu_n による力は次式で表される．ただし圧縮力を正とする．

$$\Delta e_n=K_n\Delta u_n, \quad \Delta d_n=\eta\frac{\Delta u_n}{\Delta t} \tag{3.6}$$

したがって，時刻 t における法線方向の弾性力 $[\Delta e_n]_t$ と粘性抵抗力 $[\Delta d_n]_t$ は次式となる．

$$[e_n]_t=[\Delta e_n]_{t-\Delta t}+\Delta e_n,\quad [d_n]_t=\Delta d_n \tag{3.7}$$

また，時刻 t における2粒子間の法線方向圧縮力は $[f_n]_t$ は次式で与えられる．

$$[f_n]_t=[e_n]_t+[d_n]_t \tag{3.8}$$

(b)　せん断方向成分

せん断方向成分についても同様に相対変位増分 Δu_s による力は次式で表される．ただし，粒子 i の時計回りの方向を正とする．

$$\Delta e_s=K_s\Delta u_s,\quad \Delta d_s=\eta_s\frac{\Delta u_s}{\Delta t} \tag{3.9}$$

したがって，時刻 t におけるせん断方向の力は次式となる．

$$[e_s]_t=[\Delta e_s]_{t-\Delta t}+\Delta e_s,\quad [d_s]_t=\Delta d_s \tag{3.10}$$

このとき，クーロンの摩擦法則から得られるせん断力より大きな力は作用しないので，せん断力には次の条件を加える．

$$\begin{aligned}&[e_n]_t\le 0\ \text{のとき，}\ [e_s]_t=[d_s]_t=0\\&|[e_s]_t\ge\mu[e_n]_t|\ \text{のとき，}\ |[e_s]_t=\mu[e_n]_t\times\mathrm{SIGN}([e_s]_t)|,\quad [d_s]_t=0\end{aligned} \tag{3.11}$$

ここで，μ は粒子間摩擦係数，SIGN は $[e_s]_t$ の正負を表す記号関数である．

前述と同様に，時刻 t におけるせん断方向の作用力 $[f_s]_t$ は次式となる．

$$[f_s]_t=[e_s]_t+[d_s]_t \tag{3.12}$$

3.1.3　運動方程式の差分近似——加速度，速度，変位を求める——

上述の方法で，着目粒子 i に接触するすべての粒子 j からの接触力 $[f_n]_t$，$[f_s]_t$ を求めると，図3.6のように粒子 i に作用する力の x 方向分力 X_i と y 方向分力 Y_i，ならびに粒子 i の中心周りのモーメント M_i（反時計回りが正）を次式により求める．

$$\left.\begin{aligned}[X_i]_t&=\sum_j\{-[f_n]_t\cos\alpha_{ij}+[f_s]_t\sin\alpha_{ij}+m_ig\}\\[Y_i]_t&=\sum_j\{-[f_n]_t\sin\alpha_{ij}-[f_s]_t\cos\alpha_{ij}\}\\[M_i]_t&=-r_i\sum_j\{[f_s]_t\}\end{aligned}\right\} \tag{3.13}$$

ここで，$\sum$ は粒子 i に接触するすべての粒子 j についての総和を表し，m_i は粒子 i の質量である．

そこで，粒子 i に関する運動方程式 (3.1) を式 (3.2) のように時間増分 Δt で差分近似すると，作用力は式 (3.13) で与えられるから，時刻 t における粒子 i の加速度は次式で得られる．

$$\left.\begin{aligned}[u_i]_t&=[X_i]_t/m_i,\\[v_i]_t&=[Y_i]_t/m_i,\\[\phi_i]_t&=[M_i]_t/I_i\end{aligned}\right\} \tag{3.14}$$

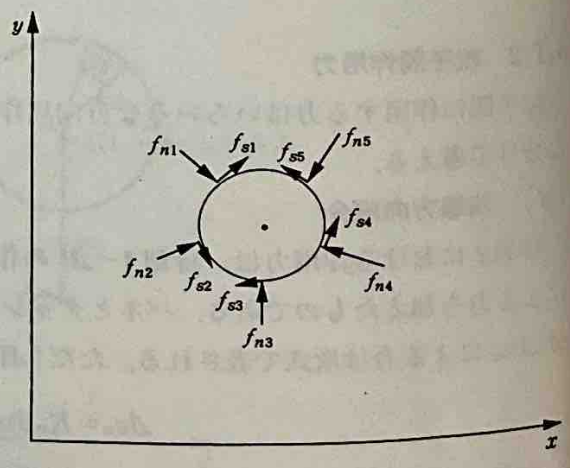


図 3.6　各接触力の合力の x，y の方向成分を求める

また，時刻 t における粒子の速度は式 (3.14) を時間増分 Δt で積分すると得られる．

$$[u_i]_t=[u_i]_{t-\Delta}+[\dot{u}_i]_t\Delta t,\quad [v_i]_t=[v_i]_{t-\Delta}+[\dot{v}_i]_t\Delta t,\quad [\psi_i]_t=[\psi_i]_{t-\Delta}+[\dot{\psi}_i]_t\Delta t \tag{3.15}$$

さらに，速度を時間増分 Δt で積分すると Δt の間の変位増分 Δu と Δv が得られる．

$$[\Delta u_i]_t=[\dot{u}_i]_t\Delta t,\quad [\Delta v_i]_t=[\dot{v}_i]_t\Delta t,\quad [\Delta\psi_i]_t=[\dot{\psi}_i]_t\Delta t \tag{3.16}$$

ここで得られた変位を時刻 t から $t+\Delta t$ までの変位増分として式 (3.6) からの計算を繰り返すと，Δt ごとの粒子の運動が計算できる．

3.1.4 弾性定数 K と粘性定数 η ならびに時間増分 Δt の決定

円形粒子間に圧縮力が作用する時の弾性係数 K は，ヘルツの弾性接触理論[3] にもとづいて次のように算出することができる．

半径がそれぞれ r_1，r_2，ヤング率 E，ポアソン比 ν，長さが 1 である 2 つの円形粒子に圧縮力 q が作用し，2 つの粒子中心の接近量が δ で接触幅が b であるとすると，δ と q の関係は次式で表される (図 3.7)．

$$\delta=\frac{2(1-\nu^2)}{\pi}\frac{q}{E}\left(\frac{2}{3}+\ln\frac{4r_1}{b}+\ln\frac{4r_2}{b}\right),$$

$$b^2=\frac{8}{\pi}\frac{r_1r_2}{r_1+r_2}\left(\frac{1-\nu^2}{E}\right)q \tag{3.17}$$

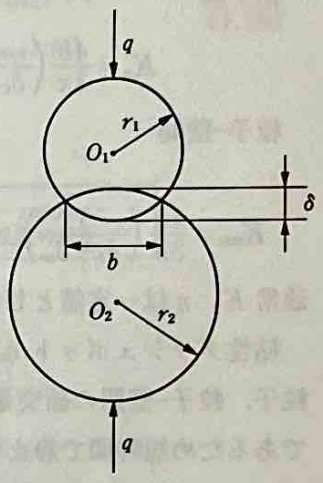


図 3.7 ヘルツの弾性接触理論にもとづくばね定数 K の算出

簡単のために，いま 2 つの円形粒子の大きさが等しいとすると ($r_1=r_2=r$)，法線方向の弾性係数 K_n は次式で与えられる．

$$K_n=\frac{\Delta e_n}{\Delta u_n}=\frac{\pi E}{2(1-\nu^2)[1.5+2\ln(4r/b)]} \tag{3.18}$$

粒子が平板に接触する場合もそれに対応するヘルツの接触理論を用いて算出することができる．

一方，せん断方向の弾性係数 K_s は K_n に対する係数 s を用いて次式から決定する．

$$K_s=\frac{\Delta e_s}{\Delta u_s}=K_ns \tag{3.19}$$

また，粘性係数 η は 2 つの決定法がある．粒子の運動方程式 (3.1) はいわゆる振動方程式であり，臨界減衰の条件である次式により η が決定される．

$$\eta_n=2\sqrt{mK_n},\quad \eta_s=\eta_n\sqrt{s} \tag{3.20}$$

ダッシュポットは系のエネルギー散逸を考慮しており，粒子間の反発係数に関係している．そこで，平板上に対象とする粒子を落下させ，その運動を再現するように η を決定する方法[4] が用いられる．

時間増分 Δt は差分解の収束性と安定性[5] から次式で決定される．

$$\Delta t\le 2\sqrt{\frac{m}{K_n}} \tag{3.21}$$

3.2　シミュレーション計算

粒子要素法シミュレーションのもっとも基本的な2次元2成分系球形粒子充填プログラムを概説する（本節末尾に掲載）．本プログラムで用いたパラメータは粒子（ガラスビーズ）の物性に可能な限り一致させている．

法線方向の弾性スプリングの定数 K_n は，ヘルツの弾性接触理論にもとづいて次式で与えられる．

粒子間

$$K_n=\frac{4b}{3\pi}\left(\frac{1}{\delta_i+\delta_j}\right),\quad \delta_i=\frac{1-\nu_i^2}{E_i\pi},\quad \delta_j=\frac{1-\nu_j^2}{E_j\pi},\quad b=\sqrt[3]{\frac{3}{2}\frac{(1-\nu^2)}{E}\frac{r_ir_jP}{r_i+r_j}} \tag{3.22}$$

粒子-壁間

$$K_{nw}=\frac{4b}{3\pi}\left(\frac{1}{\delta_i+\delta_w}\right),\quad \delta_i=\frac{1-\nu_i^2}{E_i\pi},\quad \delta_w=\frac{1-\nu_w^2}{E_w\pi},\quad b=\sqrt[3]{\frac{3r_i}{4}\frac{(1-\nu_i^2)}{E_p}\frac{(1-\nu_w^2)P}{E_w}} \tag{3.23}$$

通常 K，η は一定値として扱われることが多いが，ここでは圧縮力 P に応じて決定している．

粘性ダッシュポット η_n は動的挙動のシミュレーションでは粒子の反発係数 e に対応させ，2粒子，粒子-壁間の衝突運動の PEM 解析で決定した値を用いるが，ここでは，充填プログラムであるため短時間で静止状態に達するように臨界減衰条件による値を用いている．また，接線方向についての K_s，η_s は次の縦弾性係数 E とせん断弾性係数 G との比を用いて，法線方向の定数倍で与えている．

$$\frac{G}{E}=\frac{1}{2(1+\upsilon)} \tag{3.24}$$

加速度から速度への差分近似はオイラー法，速度から変位については近似精度を上げるためオ

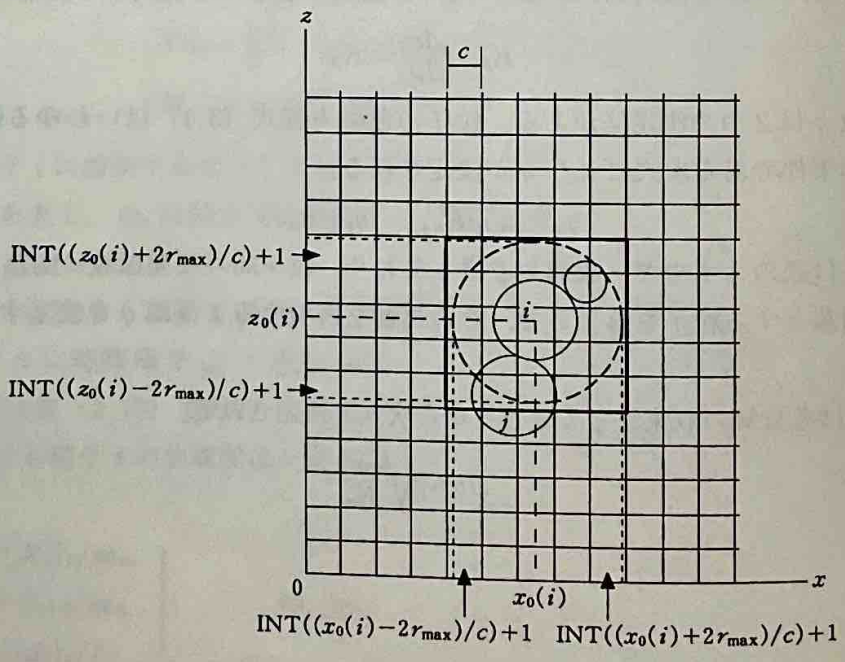


図 3.8　粒子間接触判定を行う領域

イラー法に修正子を用いている．

粒子間の接触判定の効率化のために，まず図3.8に示すように解析領域をセルで区切り各粒子を格納している．セルのサイズ C は1つのセルに小粒子の中心 $(x_0(i), z_0(i))$ が1つ入るように設定する．

$$C<\sqrt{2}\,r_{\min} \tag{3.25}$$

このとき，接触可能な粒子の存在領域 L は次式で与えられ，この領域のセル内の粒子との接触判定のみを行えばよく，計算回数が大幅に削減できる．

$$\begin{aligned} &x\text{ 方向} \quad \mathrm{INT}(x_0(i)-2r_{\max})<L_x<\mathrm{INT}(x_0(i)+2r_{\max})+1 \\ &z\text{ 方向} \quad \mathrm{INT}(z_0(i)-2r_{\max})<L_z<\mathrm{INT}(z_0(i)+2r_{\max})+1 \end{aligned} \tag{3.26}$$

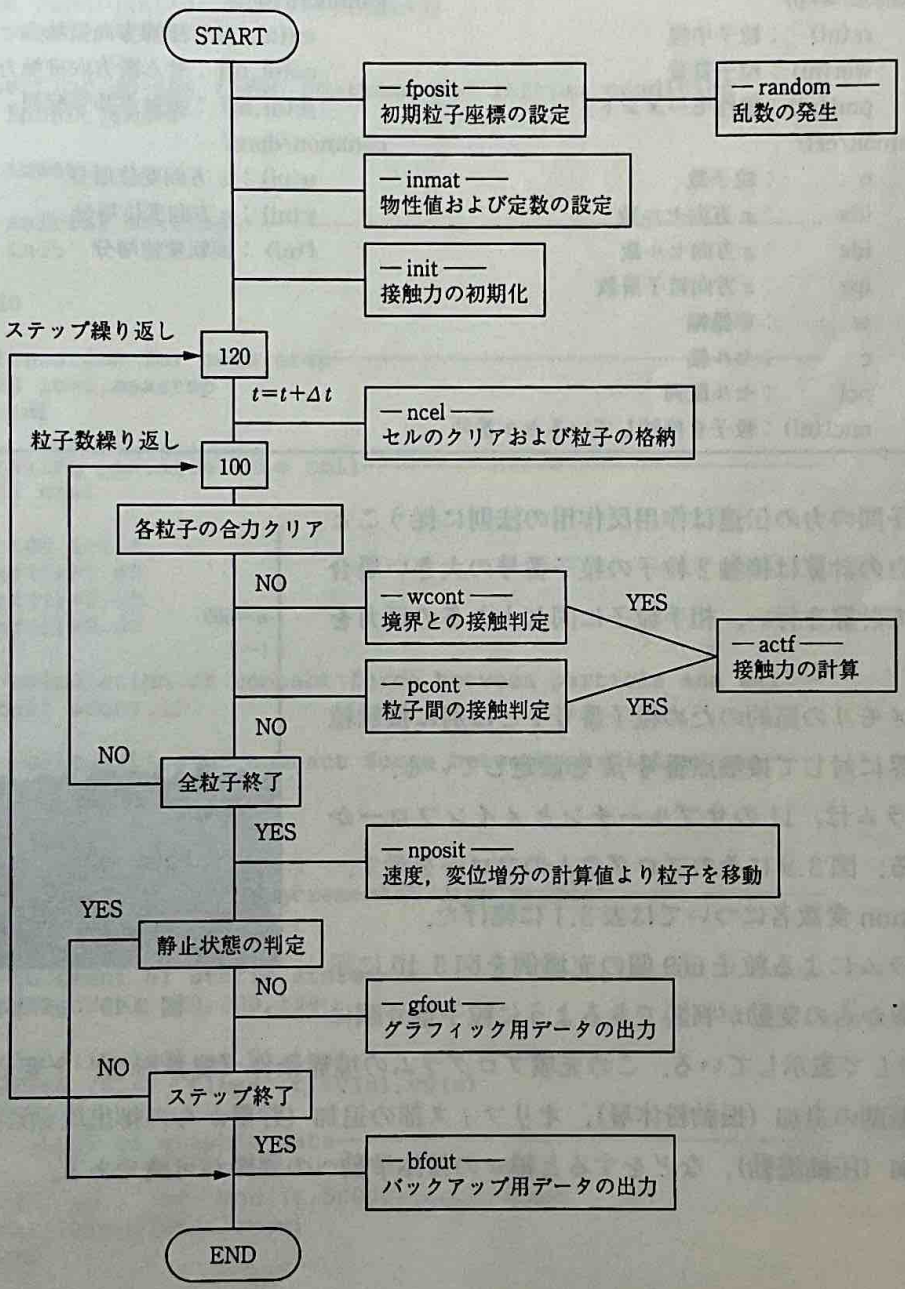


図 3.9 PEMプログラムのフローシート

表 3.1　common 文変数の説明

common/con/
- dt　：離散化時間
- fri　：粒子間摩擦係数
- frw：壁-粒子間摩擦係数
- e　：粒子の弾性係数
- ew　：壁の弾性係数
- po　：粒子のポアソン比
- pow：壁のポアソン比
- so　：縦弾性係数とせん断弾性係数の比
- g　：重力加速度
- de　：粒子の密度
- pi　：円周率

common/wep/
- rr(ni)　：粒子半径
- wei(ni)：粒子質量
- pmi(ni)：慣性モーメント

common/cel/
- n　：粒子数
- idx　：x 方向セル数
- idz　：z 方向セル数
- ipz　：z 方向粒子層数
- w　：容器幅
- c　：セル幅
- ncl　：セル配列
- nncl(ni)：粒子を格納しているセル番号

common/pos/
- x0(ni)：粒子の中心 x 座標
- z0(ni)：粒子の中心 z 応標
- qq(ni)：粒子の回転変位

common/vel/
- u0(ni)：粒子の x 方向速度
- v0(ni)：粒子の z 方向速度
- f0(ni)：粒子の回転速度

common/for1/
- xf(ni)：粒子の x 方向合力
- zf(ni)：粒子の z 方向合力
- mf(ni)：粒子のモーメント

common/for2/
- en(ni,nj)：法線方向接触力
- es(ni,nj)：せん断方向接触力
- je(ni,nj)：接触点番号配列

common/dpm/
- u(ni)：x 方向変位増分
- v(ni)：z 方向変位増分
- f(ni)：回転変位増分

また，粒子間の力の伝達は作用反作用の法則に従うことから，接触力の計算は接触2粒子の粒子番号の大きい場合のみについて計算を行い，相手粒子に同じ大きさの反力を与えている．

さらに，メモリの節約のため粒子番号 i とは別に接触粒子および境界に対して接触点番号 jk を設定している．

本プログラムは，11のサブルーチンとメインフローからなっている．図3.9にそのプログラムのフローを示す．また，common 変数名については表3.1に掲げた．

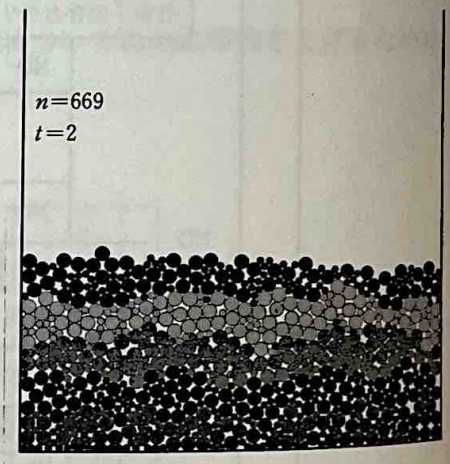


図 3.10　初期充填状態

本プログラムによる粒子669個の充填例を図3.10に示す．初期位置からの変動が判断できるように粒子番号順に5層に色分けして表示している．この充填プログラムの境界条件（境界形状）を変えたり，または境界への振動の追加（振動粉体層），オリフィス部の追加（貯層からの排出流動），上部プランジャーの追加（圧縮流動），などをすると種々の粉体挙動への展開が可能である．

2次元2成分系 PEM 充填プログラム（球モデル）

```
c*********************************************************************
c***                                                               ***
c***          two-dimensional particle element method              ***
c***             sphere model  ......   i-pem.f                    ***
c***                                                               ***
c*********************************************************************
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /con/dt,fri,frw,e,ew,po,pow,so,g,de,pi
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      common /vel/u0(ni),v0(ni),f0(ni)
      common /for1/xf(ni),zf(ni),mf(ni)
      common /for2/en(ni,nj),es(ni,nj),je(ni,nj)
      common /dpm/u(ni+3),v(ni+3),f(ni+3)
      data maxstep/2000000/
c
c---------setting up the first position and initial condition
      call fposit(rmax)
c
      call inmat
c
c--------- initial setting----------------------------------------
      call init
c
      t=0.d0
c
c---------iteration for each step---------------------------------
      do 120 it=1,maxstep
        t=t+dt
c
c---------taking particle into cell-------------------------------
        call ncel
c
        do 100 i=1,n
          xf(i)=0.d0
          zf(i)=0.d0
          mf(i)=0.d0
c
c----------calculation of contact force between particle and wall--
          call wcont(i)
c
c----------calculation of contact force between particles---------
          call pcont(i,rmax)
c
  100   continue
c
c----------superposition of incremental displacement --------------
        call nposit(judge)
c
c----------judgment of static state--------------------------------
        if (1-judge) 200,200,199
c
  199   if (mod(it,100).eq.0) then
          write (6,*) 'time=',t,z0(n),v0(n)
        endif
c----------output of graphic data----------------------------------

        if (it.eq.1 .or. mod(it,50000).eq.0) then
          call gfout(it,t,rmax)
        endif
c
  120 continue
```

```
c---------output of back up data----------------------------------
  200 call bfout
c
      close(10)
      close(11)
c
      stop
      end
c
c******* fposit ***********************************************
c
      subroutine fposit(rmax)
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /con/dt,fri,frw,e,ew,po,pow,so,g,de,pi
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      data ii/584287/
      data r1,r2/1.d-2,5.d-3/
      data w,ipz/5.d-1,30/
c
      rmax=r1
      rmin=r2
      rn=rmax+1.d-5
      ipx=idint(w/2.d0/rn)
      n=0
      do 20 i=1,ipz
        if (mod(i,2).eq.0)then
           dx=2.d0*rn
           ip=ipx-1
        else
           dx=rn
           ip=ipx
        endif
        do 10 j=1,ip
          call random(ii,ru)
          if (ru.lt.2.d-1) goto 10
          n=n+1
          if (n.gt.ni) write(6,*)
     $      'number of particles is more than ',ni
          x0(n)=2.d0*rn*(j-1)+dx
          z0(n)=2.d0*rn*(i-1)+rn
c
          call random(ii,ru)
          if (ru.lt.5.d-1) then
            rr(n)=r1
          else
            rr(n)=r2
          endif
   10   continue
   20 continue
      write (6,*) 'number of particles:',n
      c=rmin*1.35d0
      idx=idint(w/c)+1
      idz=idint(z0(n)/c)+10
      if (idx*idz.gt.nc) then
        write(6,*) 'ncl is over!!',idx*idz
        stop
      endif
      return
      end
c
c******* inmat ************************************************
c
      subroutine inmat
      implicit real*8(a-h,k,m,o-z)
```

```
      parameter (ni=1000,nj=13,nc=20000)
      common /con/dt,fri,frw,e,ew,po,pow,so,g,de,pi
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      data dt/1.d-6/,pi/3.14159d0/,g/9.80665d0/
      data de,e,ew,po,pow/2.48d3,4.9d9,3.9d9,2.3d-1,2.5d-1/
      data fri,frw/2.5d-1,1.7d-1/
c
      so=1.d0/2.d0/(1.d0+po)
      do 10 i=1,n
        wei(i)=4.d0/3.d0*pi*rr(i)*rr(i)*rr(i)*de
        pmi(i)=8.d0/15.d0*de*pi*(rr(i))**5.
   10 continue
c
      return
      end
c
c******* init ****************************************
c
      subroutine init
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /for2/en(ni,nj),es(ni,nj),je(ni,nj)
c
      do 3 i=1,n
        do 2 j=1,nj
          en(i,j)=0.d0
          es(i,j)=0.d0
    2   continue
        do 4 ij=1,nj
         je(i,ij)=0
    4   continue
    3 continue

      return
      end
c
c******* ncel ****************************************
c
      subroutine ncel
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      common /vel/u0(ni),v0(ni),f0(ni)
      common /for1/xf(ni),zf(ni),mf(ni)
      common /for2/en(ni,nj),es(ni,nj),je(ni,nj)
      common /dpm/u(ni+3),v(ni+3),f(ni+3)
c
      do 10 ib=1,(idx*idz)
        ncl(ib)=0
   10 continue
      do 20 i=1,n
         nncl(i)=0
         ib=idint(z0(i)/c)*idx+idint(x0(i)/c)+1
         ncl(ib)=i
         nncl(i)=ib
   20 continue
      return
      end
c
c******* wcont ****************************************
c
      subroutine wcont(i)
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
```

```
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      common /vel/u0(ni),v0(ni),f0(ni)
      common /for1/xf(ni),zf(ni),mf(ni)
      common /for2/en(ni,nj),es(ni,nj),je(ni,nj)
      common /dpm/u(ni+3),v(ni+3),f(ni+3)
c
      xi=x0(i)
      zi=z0(i)
      ri=rr(i)
c
c  --- left wall ---
c
      jk=11
      j=n+1
      if (xi.lt.ri) then
        as=0.d0
        ac=-1.d0
        gap=dabs(xi)
        je(i,jk)=n+1
        call actf(i,j,jk,as,ac,gap)
      else
        en(i,jk)=0.d0
        es(i,jk)=0.d0
        je(i,jk)=0
      endif
c
c  --- under wall ---
c
      jk=12
      j=n+2
      if (zi.lt.ri) then
        as=-1.d0
        ac=0.d0
        gap=dabs(zi)
        je(i,jk)=n+2
        call actf(i,j,jk,as,ac,gap)
      else
        en(i,jk)=0.d0
        es(i,jk)=0.d0
        je(i,jk)=0
      endif

c
c   --- right wall ---
c
      jk=13
      j=n+3
      if (xi+ri.gt.w) then
        as=0.d0
        ac=1.d0
        gap=dabs(xi-w)
        je(i,jk)=n+3
        call actf(i,j,jk,as,ac,gap)
      else
        en(i,jk)=0.d0
        es(i,jk)=0.d0
        je(i,jk)=0
      endif
      return
      end
c
c
c******* pcont ***********************************************
c
      subroutine pcont(i,rmax)
```

```
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      common /vel/u0(ni),v0(ni),f0(ni)
      common /for1/xf(ni),zf(ni),mf(ni)
      common /for2/en(ni,nj),es(ni,nj),je(ni,nj)
      common /dpm/u(ni+3),v(ni+3),f(ni+3)
c
      xi=x0(i)
      zi=z0(i)
      ri=rr(i)
c
      lup=idint((zi+2.d0*rmax)/c)
      lun=idint((zi-2.d0*rmax)/c)
      llf=idint((xi-2.d0*rmax)/c)
      lrg=idint((xi+2.d0*rmax)/c)
      if (lup.ge.idz) lup=idz-1
      if (lun.lt.0) lun=0
      if(llf.lt.0) llf=0
c
      if(lup.le.lun) then
         write(6,*) i,'lup=',lup,'lun=',lun,'c=',c,
     &              'rmax=',rmax,xi,zi,idz
      endif
c
      do 90 lz=lun,lup
        do 80 lx=llf,lrg
         ib=lz*idx+lx+1
         j=ncl(ib)
         if (j.le.0) goto 80
         if (j.eq.i) goto 80
         do 11 jj=1,10
           if (je(i,jj).eq.j) then
             jk=jj
             goto 70
           end if
   11    continue
         do 12 jj=1,10
           if (je(i,jj).eq.0) then
             jk=jj
             je(i,jj)=j
             goto 70
           end if
   12    continue
   70    xj=x0(j)
         zj=z0(j)
         rj=rr(j)
         gap=dsqrt((xi-xj)*(xi-xj)+(zi-zj)*(zi-zj))
         if (gap.lt.dsqrt((ri+rj)*(ri+rj))) then
           if (i.gt.j) then
             ac=(xj-xi)/(gap)
             as=(zj-zi)/(gap)
             j0=0
             do 555 jj=1,10
             if (je(j,jj).eq.i) then
                 j0=jj
                 goto 554
             endif
  555        continue
  554        call actf(i,j,jk,as,ac,gap)
             en(j,j0)=en(i,jk)
             es(j,j0)=es(i,jk)
             j0=0
           endif
         else
```

```
   85      en(i,jk)=0.d0
           es(i,jk)=0.d0
           je(i,jk)=0
         endif
   80    continue
   90 continue
      return
      end
c
c******* nposit ***********************************************
c
      subroutine nposit(judge)
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /con/dt,fri,frw,e,ew,po,pow,so,g,de,pi
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      common /vel/u0(ni),v0(ni),f0(ni)
      common /for1/xf(ni),zf(ni),mf(ni)
      common /dpm/u(ni+3),v(ni+3),f(ni+3)
c
      sum=0.d0
      do 110 i=1,n
        v0(i)=v0(i)+(zf(i)-wei(i)*g)/wei(i)*dt
        u0(i)=u0(i)+xf(i)/wei(i)*dt
        f0(i)=f0(i)+mf(i)/pmi(i)*dt
        v(i)=(v0(i)*dt+v(i))/2.d0
        u(i)=(u0(i)*dt+u(i))/2.d0
        f(i)=(f0(i)*dt+f(i))/2.d0
        z0(i)=z0(i)+v(i)
        x0(i)=x0(i)+u(i)
        qq(i)=qq(i)+f(i)
        sum=dabs(u(i))+dabs(v(i))+sum
  110 continue
      av=sum/real(n)/2.d0
      if (av.lt.(dt*dt*g*1.0d-1)) then
        judge=1
      else
        judge=0
      endif
      return
      end
c
c
c******* actf **************************************************
c
      subroutine actf(i,j,jk,as,ac,gap)
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /con/dt,fri,frw,e,ew,po,pow,so,g,de,pi
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      common /vel/u0(ni),v0(ni),f0(ni)
      common /for1/xf(ni),zf(ni),mf(ni)
      common /for2/en(ni,nj),es(ni,nj),je(ni,nj)
      common /dpm/u(ni+3),v(ni+3),f(ni+3)
c
      ri=rr(i)
      if ((j-n).lt.0) then
        rj=rr(j)
        dis=ri+rj-gap
        wei3=2.d0*wei(i)*wei(j)/(wei(i)+wei(j))
      else
        rj=0.d0
        dis=ri-gap
```

```
          wei3=wei(i)
        endif
        enn=en(i,jk)
        if (enn.le.0.d0) enn=1.d0
        if ((j-n).le.0.d0) then
          b1=(3.d0/2.d0/e*ri*rj/(ri+rj)*(1.d0-po*po)
     &   *enn)**(1./3.)
          knn=2.d0/3.d0*b1*e/(1.d0-po*po)
          kss=knn*so
          vnn=2.d0*dsqrt(wei3*knn)
          vss=vnn*dsqrt(so)
        else
          b1=((3.d0/4.d0*ri*((1.d0-po*po)/e+(1.d0-pow*pow)/ew))
     &   *enn)**(1./3.)
          knn=4.d0/3.d0*b1*e*ew/((1.d0-po*po)*ew+(1.d0-pow*pow)*e)
          kss=knn*so
          vnn=2.d0*dsqrt(wei3*knn)
          vss=vnn*dsqrt(so)
        endif
        ddt=1.d-1*dsqrt(wei3/knn)
        if (ddt.lt.1.d-6) write (6,*) 'over!! ddt=',ddt,i,j,jk,knn,wei(i)
        un=(u(i)-u(j))*ac+(v(i)-v(j))*as
        us=-(u(i)-u(j))*as+(v(i)-v(j))*ac+(ri*f(i)+rj*f(j))
c-----------------------------------------------------------------------
        if(en(i,jk).eq.0.d0) then
          if (un.ne.0.d0) us=us*dis/un
          un=dis
        endif
c-----------------------------------------------------------------------
        en(i,jk)=knn*un+en(i,jk)
        es(i,jk)=es(i,jk)+kss*un
        if (i.eq.4) then
        endif
        dn=vnn*un/dt
        ds=vss*us/dt
        if (en(i,jk).lt.0.d0) then
          en(i,jk)=0.d0
          es(i,jk)=0.d0
          dn=0.d0
          ds=0.d0
          je(i,jk)=0
         return
        elseif ((j-n).le.0) then
          frc=fri
        else
          frc=frw
        endif
        if ((dabs(es(i,jk))-frc*en(i,jk)).gt.0) then
          es(i,jk)=frc*dsign(en(i,jk),es(i,jk))
          ds=0.d0
        endif
          hn=en(i,jk)+dn
          hs=es(i,jk)+ds
          xf(i)=-hn*ac+hs*as+xf(i)
          zf(i)=-hn*as-hs*ac+zf(i)
          mf(i)=mf(i)-ri*hs
          if (jk.lt.10) then
            xf(j)=hn*ac-hs*as+xf(j)
            zf(j)=hn*as+hs*ac+zf(j)
            mf(j)=mf(j)-rj*hs
          endif

        return
        end
c
c******* gfout ****************************************
c
```

```
      subroutine gfout(it,t,rmax)
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      common /vel/u0(ni),v0(ni),f0(ni)
      common /for2/en(ni,nj),es(ni,nj),je(ni,nj)
c
      if (it.eq.1) then
        open(10,file='graph11.d')
        open(11,file='graph21.d')
        write(10,*) n,t,w,rmax
      else
        write(10,*) (sngl(x0(i)),sngl(z0(i)),sngl(rr(i)),i=1,n)
        write(10,*) (sngl(u0(i)),sngl(v0(i)),sngl(f0(i)),i=1,n)

        write(11,*) ((sngl(es(i,j)),sngl(en(i,j)),j=1,nj),i=1,n)
        write(11,*) ((je(i,j),j=1,nj),i=1,n)
      endif

      return
      end
c
c******* bfout ***********************************************
c
      subroutine bfout
      implicit real*8(a-h,k,m,o-z)
      parameter (ni=1000,nj=13,nc=20000)
      common /con/dt,fri,frw,e,ew,po,pow,so,g,de,pi
      common /wep/rr(ni),wei(ni),pmi(ni)
      common /cel/n,idx,idz,ipz,w,c,ncl(nc),nncl(ni)
      common /pos/x0(ni),z0(ni),qq(ni)
      common /vel/u0(ni),v0(ni),f0(ni)
      common /for1/xf(ni),zf(ni),mf(ni)
      common /for2/en(ni,nj),es(ni,nj),je(ni,nj)
      common /dpm/u(ni+3),v(ni+3),f(ni+3)
c
      open(13,file='back1.d')
      write(13,*) n,idx,idz,ipz
      write(13,*) rmax,t,w,c,dt
      write(13,*) de,fri,frw,g,pi
      write(13,*) e,ew,po,pow,so
      write(13,*) (wei(i),pmi(i),i=1,n)
      write(13,*) (x0(i),z0(i),rr(i),i=1,n)
      write(13,*) (u(i),v(i),f(i),i=1,n)
      write(13,*) (u0(i),v0(i),f0(i),i=1,n)
      write(13,*) ((es(i,j),en(i,j),j=1,nj),i=1,n)
      write(13,*) ((je(i,j),j=1,nj),i=1,n)
      close(13)

      return
      end
c
c******* random **********************************************
c
      subroutine random(ii,ru)
      implicit real*8(a-h,k,m,o-z)
c
      ii=ii*65539
      if (ii.lt.0) ii=(ii+2147483647)+1
      ru=ii*0.4656613d-9
      return
      end
```