離散要素法について

早川尚男

平成 11 年 7 月 19 日

Ver.1 : July 3, 1999: Current version: Ver1.1

1 はじめに

このノートは研究室の学生に離散要素法 (DEM) の手ほどきをするために執筆したものである。こうしたノートを書いた動機は詳しく考え方や問題点、プログラムの実際を整理したものが殆んど存在しないためでもある。

DEM は一種の分子動力学法 (MD) である. MD そのものが他のシミュレーション法に比べ、かなり長大なプログラムリストと特殊なテクニックを要するので初学者には敷居が高い。まして DEM はわかりにくい仮定が多々含まれており、それに納得するまでに長時間を要した。

私自身ずっと DEM を避けてあやしげな現象論を論じてきたが、やはり限界があることを悟った。DEM を行なうかどうかはある種の決断ー粉体研究を続けてプロとなるかーを要した。今後、粉体の研究をする者にとって DEM は最低限の教養、道具となるだろう。

このノートは下坂氏 (同志社) による公開された計算機コード [1] を元にしている。結局、DEM 遂行の決断から完成までに 4 ヶ月を要したが、その際に彼女のコードに全面的に依存したばかりでなく、様々なアドバイスを頂いた。ここに載せたサンプルプログラムも殆んど彼女のアルゴリズムに依っている。その意味で著作権は彼女にあり、本ノートは彼女の本の中でコードの説明 (わかりやすいものとはいえない) を改善し、補完するものと位置づけられる。また梅景氏 (九工大) にも有益なアドバイスを頂いた。その他、研究室の和田、國仲の両君と磯部氏 (九大理)、酒井氏 (京大総人地学) にも様々な面でお世話になった。改めてこれらの方に謝辞を表したい。

2 概略

ここの章の記述は早川・那須野による物理学最前線の記事の該当章そのままである。

物理学が通常の意味で要素還元論を指向するのであれば粉体のミクロなモデルというものを設定しなければならない．その場合どこまで遡って考えるかということについては基本的合意がない．最も単純に考えれば，粉体の構成要素はひとつひとつの粉体粒子であり，それらの集団が粉体ということになるが，粒子の散逸性や相互作用をきちんと議論しようとすると，よりミクロなレベルにまで遡った議論が必要となる．ところが，現在まで散逸の起源について真面目に研究した例は少なく，同時に現在もその理解については混乱している．言わば，並進と回転の自由度以外の内部振動に伴う自由度の分配や気体への熱伝導などについて何もわかっていない．このことが後述する様に散逸の強度を決める際の不定性をもたらす．今後，この方面の基礎的な研究が発展していくと思われるが，ここではひとまずそれはおいておいて，粉体要素の力学的基礎とされる事実をまとめ，その事実を巧みに採り入れた離散要素法について紹介しよう [2]．

現実の粉体の相互作用は複雑であるが，最も単純には接触した際に斥力のみが働くと考えればよい．その際，どのような力が働くかについては前世紀から詳細な研究がある．最も有名なものはヘルツ（Hertz）[3] による固体接触の理論であり，大抵の弾性論の教科書には詳細に記述されてある [4]．ここで簡単にその結果を紹介しよう．まず図 2.1 に示すような 3 次元の球同士の接触を考える．両物体の圧縮力の合力を F_n とすると両物体は変形し，接触面は半径 b の円となる．その半径 b 及び接触領域の距離 δ_n は接触粉体粒子の半径 a, a' を用いて

$$b = F_n^{1/3}(D\frac{aa'}{a+a'})^{1/3} : \quad \delta_n = F_n^{2/3}[D^2(\frac{1}{a}+\frac{1}{a'})]^{1/3} \tag{1}$$

で与えられる．ここで $D = \frac{3}{4}(\frac{1-\sigma_0^2}{E}+\frac{1-\sigma_0'^2}{E'})$ であり，σ_0, E はポアソン比，ヤング率と呼ばれる基本定数となっている[1]．ここで注目すべきは変形 δ_n と力 F は $F \propto \delta_n^{3/2}$ という関係式で結ばれていることである．この関係は線形ばねによる変形と力の比例関係とは異なる．この結果は勿論次元に依存したものであり，2 次元的な物体，即ち円筒であれば線形ばねの結果に対数補正の入った形になっている．

次に接線方向に働く弾性力についてもミンドリン (Mindlin)[5] によって詳しく論じられている．ここで δ_t を粉体が接触したときからの接線方向の変位とすると先に出てきた接触円の半径 b を用いて

$$F_t = 4b\delta_t(\frac{(2-\sigma_0)(1+\sigma_0)}{E}+\frac{(2-\sigma_0')(1+\sigma_0')}{E'})^{-1} \tag{2}$$

で与えられる．$b \sim F_n^{1/3} \sim \delta_n^{1/2}$ という関係式から接線方向に働く力は $F_t \sim \delta_t\delta_n^{1/2}$ という関係を満たす．この力は接線方向のずりに対する復元力であるので静止摩擦の表現ともみなすことも可能である[2]．しかし，「粉体」全体としてのマクロな静止摩擦には，こうした素子レベルの力の他に粒子相互の配置状態（充填状態）が効いてくることを注意しておく．

上記の弾性論の枠組の中で論じることが出来た力の他に法線方向，接線方向に独立な粘性係数を持った散逸力が働く．法線方向には相対速度に比例する摩擦を用い，接線方向には低速時にはやはり相対速度に比例した摩擦，また高速時に動摩擦係数 μ に対して $F_t = \mu F_n$ に達するとスリップするというクーロン（Coulomb）摩擦を用いる．ここの動摩擦係数の導入はある意味で「粉体」のマクロな物性を素子レベルにも持たせたことになっており，動摩擦の起源といった問題はここでも不問になる．しかし後述の様にこの動摩擦がまさに摩擦現象では大きな役割を果たしているように思える．

通常，その他に粒子同士が互いに剛体回転をすることも仮定する．今までの記述では 2 体問題しか考えていなかったが，この 2 体力を多体問題に適用しようというのが離散要素法である [6]．即ち，離散要素法では並進と回転の自由度のみを残し，残りの自由度を散逸等としてモデルのパラメータに押し込めてしまうのである．離散要素法は，元来，地盤の準静的な挙動の解析に用いられ，更に流体工学，機械工学，化学工学の分野に徐々に浸透し，物理の分野では 1990 年以降に盛んに用いられるようになってきた．離散要素法では上記の弾性力，粘性力，回転をそれぞれ素子の持つばね，ダッシュポット，スライダー[3] というものに読み替え，個々の粒子をそれぞれの部品を持った機械のように扱うのである（図 2.2）．全ての粒子の粒径が d, 質量 m, 慣性モーメントが I である場合に粒子の運動方程式は

$$\begin{aligned} m\ddot{\mathbf{x}}_i &= \sum_j \theta(d-|\mathbf{x}_i-\mathbf{x}_j|)(\tilde{\mathbf{F}}_n^{ij}+\tilde{\mathbf{F}}_t^{ij}) \\ I\dot{\omega}_i &= \frac{d}{2}\sum_j \theta(d-|\mathbf{x}_i-\mathbf{x}_j|)(\mathbf{n}\times\tilde{\mathbf{F}}_t^{ij}) \end{aligned} \tag{3}$$

[1] ヤング率は単軸の引っ張りあるいは圧縮に対する応力と歪みの比例定数．ポアソン比は単軸の引っ張りを行なった場合の垂直変位（縮み）と軸方向の延びの比．

[2] 通常の離散要素法においては，静止摩擦は省略されることも多く，この項が唯一静止摩擦に相当する役割を果たしている．

[3] ダッシュポットというのは図にあるように空気鉄砲みたいなものである．これは速度に比例する摩擦を表す機械だと思って構わない．並列に結合しているということは力が加算的であることを意味する．スライダーはクーロン摩擦を意味する．クーロン摩擦が働いた状態では回路は切断されて他の力が働かずに滑べっていくことを表現している．慣れないとわかりにくい表記であるが，考えてみるとよく出来た表記である．

ここで $\mathbf{x}_i$,ω_i は粒子 i の位置と角速度であり，和は接触粒子対に対して取り，$\theta(x)$ はステップ関数，即ち $x>0$ に対して $\theta(x)=1$，$x<0$ に対しては $\theta(x)=0$ であるとする．また $\mathbf{n}, \mathbf{t}$ をそれぞれ接触粒子対の中心を結ぶ線に対して法線方向の単位ベクトル，接線方向の単位ベクトルとし，$\tilde{\mathbf{F}}_n^{ij}, \tilde{\mathbf{F}}_t^{ij}$ はそれぞれ

$$
\begin{aligned}
\tilde{\mathbf{F}}_n^{ij} &= \mathbf{F}_n^{ij} - \eta_n \mathbf{n}\mathbf{n}\cdot(\dot{\mathbf{x}}_i - \dot{\mathbf{x}}_j) \\
\tilde{\mathbf{F}}_t^{ij} &= \begin{cases} \mathbf{F}_{t,nsl}^{ij} & (|\mathbf{F}_{t,nsl}{}^{ij}| < \mu\tilde{\mathbf{F}}_n^{ij} \text{ の場合 }) \\ \mu\tilde{\mathbf{F}}_n^{ij} & (\text{それ以外}) \end{cases} \\
\mathbf{F}_{t,nsl}^{ij} &= \mathbf{F}_t^{ij} - \eta_t \mathbf{t}[\mathbf{t}\cdot(\dot{\mathbf{x}}_i - \dot{\mathbf{x}}_j) + \frac{d}{2}(\omega_i + \omega_j)]
\end{aligned}
\tag{4}
$$

で与えられる．ここで $\mathbf{F}_n^{ij}, \mathbf{F}_t^{ij}$ は (1),(2) 式に現れる $\mathbf{F}_n, \mathbf{F}_t$ の接触粒子の i,j のペアに対するものである．

次にこのモデルの問題点を指摘しよう．既に述べてきた様に (4) 式に現れる粘性係数 η_n,η_t の決定に任意性が残る．法線方向に限って事情を説明しよう．多くの離散要素法 [6] では 2 体問題として捉え (1) の 2 つの式から $F/\delta_n = b/D$ という関係式を利用し，形式的に線形方程式とみなし（実際には右辺が δ_n に依存する）臨界減衰値，即ち，振動から減衰に変移する点を選んだ．勿論これは根拠のないことである．一方，田中ら [7] は粘性係数が反発係数 e と関係がつく条件を求めてその値を採用した．これまた根拠のないことのように思える．即ち，反発係数が一意的に決定されなければならない理由は何もないからである．実際，後述する粉体気体の研究では反発係数は粒子の運動状態によっているのではないかという報告がある [8]．現状では反発係数 e と対応がつく条件を用いて決めている，という感があるが，今後この種の粘性パラメータの設定はよりミクロな物理的考察から決められるであろう．[4] 他のパラメータの決定はもっと深刻である．動摩擦係数などはマクロな実験から推定して決めているのが現状であろう．また剛体回転をモデルの中で用いているが，それも実はおかしい．現実には接触物体の変形に基づき反発力を決めているので完全な剛体の式を使うのはまずい筈である．実際，剛体回転では粒子が転がり過ぎることが経験的に知られている. 現実の転がり摩擦では弾性ヒステリシス損失と呼ばれる弾性体の変形に伴う効果が重要であり [9] , その効果を考慮した離散要素法のシミュレーションがより現実的な振舞をすることが知られている [10]. 更に湯ら [11] は離散要素法でのヤング率などのパラメータ設定で物質から決まる値を用いるよりもややずれた値 (ヤング率の場合は小さい値) を用いた方が現実的な振舞をすることを報告している．また現実の粒子をシミュレートしようとする場合には形状の問題が重要になってくる. 例えばやや細長い粒子をシミュレートするには楕円要素或はその 3 次元版を用いることはよく行なわれる [12]．しかし接触の判定が複雑になる上に楕円にしたところで表面の突起やラフネスは解決されていないことには注意しなければならない.

以上の様に考えると離散要素法の適用にはかなり問題があるのではないかという疑問が浮かぶ．その問いに答える形で離散要素法はどの程度現実的であるかを考えてみよう．この手法はある程度弾性論に基づく正当化はあるものの既に述べた様に多くの根本的問題を含んでいる．従って現象論であることは間違いない．しかしその一方でパラメータをうまく設定すれば驚くほど定量的に実験を再現できる．また本末転倒かもしれないが離散要素法に合わせた実験の設定というのも考えられる [13]．この場合の実験結果は定量的に離散要素法の結果と一致しており，離散要素法の正当性を支持した格好になっている．その意味では非常によく出来たモデルと言える．また離散要素法を越える様なモデルも現在の所存在しない．この手法に含まれる個々のプロセスは物理的にははっきりしているし，パラメータの設定もある程度は可能である．従って，現時点では粉体素子を記述する一つのモデルに過ぎないがその有効性は経験に裏打ちされたものであると言えよう．

もっとも離散要素法はある程度の信頼すべき結果を数値的に生み出すということの意義は認めるべきであろう．また多くの研究者が行なっている様にモデルの簡略化はまだ可能である．例えば (3),(4) の全ての

[4] 専門家が実際に決める際には実際にシミュレーションと実験を比較しつつ決めているようである．勿論反発係数 e は衝突速度に依存するので，こうした決め方も狭いパラメータ領域でしか有効ではない．いずれにしても反発係数 e は一意的に決定できないパラメータであることに注意．

式を用いる必要があるとは限らない．ミンドリン力は多くの場合考慮されていないし，回転も含まれていない場合が多い．[5] また接触力にしても線形ばねの他に変形 δ_n の逆べきで発散するようなモデルもよく用いられている．それらのモデルのバリエーションに対してあまり詳細な性質が変わらない現象も多い．特に充填率が高い場合には排除体積効果が重要になり，結局は隙間がある所に動くことができるという性質に帰着できることが多いからであろう．こうした考え方を更に積極的に押し進めて，できるだけパラメータを減らそうという指向もある．こうした考えが田口 [2] の言うところの普遍性指向に結実できるかどうかは甚だ疑問ではあるが，一つの理解できる考え方である．

更にパラメータを減らしたモデル，例えば非弾性散乱をするハードコア剛体球などを使うともっとすっきりするのではという疑問も湧くであろう．実際，多くの物理学者はその単純性故に非弾性剛体球（円盤）のシミュレーションを行なっているが，直ちにその計算が発散することに気がついた．即ち衝突の度に運動エネルギーを失い，遂には常に接触している状態になるのであるがハードコアモデルではそこで無限大の力が働くために発散が生じるのである．この発散を非弾性コラプス（inelastic collapse）として議論しようとしているが，むしろ接触力のモデル化が不適当であると捉えるべきである．

モデルの基礎ばかりでなく，離散要素法の適用限界についてもやはり認識しておく必要がある．特に想定対象は必ずしも離散要素法で仮定されたような理想的な状況にはなく，また全ての状況に応じたシミュレーションは不可能であることを注意しておく．即ち，実験では粒子は通常，球ではなく粒径も分散しており，空気や水などの介在物の影響や，次元性や粒子数なども異なる場合があり必ずしも実験とシミュレーションの状況が対応しているとは言い難いからである．特に粒子数の問題は実際の問題に応用するときは深刻である．数万個の粒子ではせいぜいスプーン一杯の粉体であり，この少量の粒子の挙動から実際の工場などでの粉体粒子の挙動を予想することは難しい．従って普遍性を持った更にマクロな現象論的なモデル化を指向することが応用上も重要になる．従って，現時点で粉体のミニマムモデルとして離散要素法を追求すべきかどうかは必ずしも自明ではない．何故なら離散要素法によってシミュレーションはできても結果の物理的解釈は殆んど不可能であるからである．少なくとも多くの理論解析は離散要素法とは独立に存在し，マクロな特徴づけを指向していることからこのことは裏付けられるだろう．

尚，離散要素法用の Fortran77 のプログラムは文献 [1] で公開されている．しかしながら印刷ミスなどが多く，説明も混乱しているのでわかりにくい．また粘着力などを導入して離散要素法を拡張し，当初の目的通りに地震に伴う地盤の崩壊や土石流などのシミュレーションに適用した例が文献 [14] に見られる．その他, 空間を格子に分割し, そこに離散的な値を与え, ある種の確率論的ルールに基づき, 離散時間の発展をするセルオートマトン法を粉体に適用し, 高速シミュレーションを可能にしようという試みもある. しかし現時点ではセルオートマトンで表現された量と粉体の対応が分からないので, その手の研究はここでは紹介しないことにする.

3　2 次元シミュレーションの考え方

実際のシミュレーションは表示の問題や粒子数の問題等から 2 次元で行なわれる事が多い。 2 次元を 3 次元の球が 1 列に並んだ系と考えれば前節の議論はそのまま使えるが、球の粒子を 2 次元物体、即ち円筒と考えると弾性理論等は変更を受ける。例えば 2 体接触の Hertz 理論は等しい半径 r、弾性率 E、ポアソン比 σ_0 を持つ場合に

$$F_n = K_n(\delta_n)\delta_n : \quad K_n(\delta_n) = \frac{\pi E}{\ln(\frac{4\pi a E}{F_n}) - 1 - \sigma_0} \tag{5}$$

[5] ミンドリン力はともかく回転が含まれていないことは考えると奇妙な状況である. 大きさがあるのに回転がないというのは非常に強い拘束か, 或は慣性モーメントが無限大であることを意味する. これは力学法則を破っているので要素の自由度が少なければよいという田口の意見 [2] には納得できない処もある.

となることが知られている。但し接触域の長さ $2b$ は圧縮力 F_n を用いて

$$b^2 = \frac{4a(1-\sigma_0^2)}{\pi E} F_n \tag{6}$$

となる [4]。[6] 導出過程については文献 [15] を参照のこと。また Mindlin 力についてはどうなるかは計算してみないとわからない。

2 次元をそのまま真面目に考えると上記の様に複雑になるが、(5) を線形弾性論に対数補正がついたと考えると話を単純化して捉えることが可能になる。つまり対数補正を無視して線形ばねで近似してもそれほど大きな違いがないという立場に立てるのである。線形弾性論を採用すると粒子の直径で長さをスケールするのが容易になり、3 次元の場合の様に粒子半径毎に異なった現象になるという危惧をする必要がなくなるのも大きなメリットである。2 次元の Mindlin 力も現時点ではパラメータとして扱ってもそれほど問題はないであろう。

3.1 粒子間の相対変位増分

ここでは下坂の説明 [1] にならって接触に伴う粒子の運動を追跡してみよう。いま下坂の図 3.4 のように粒子 i が時刻 t から $t+\Delta t$ の間に位置 A から位置 B に変位したとする。このときの変位増分の x 成分、y 成分をそれぞれ Δu_i, Δv_i とし、回転変位を $\Delta\psi_i$ であるとする。

位置 $(x_i, y_i), (x_j, y_j)$ に中心を持つ粒子半径が r_i, r_j の 2 つの円粒子は

$$r_i + r_j \geq R_{ij} : \quad R_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{7}$$

の条件で接触する。いうまでもなく R_{ij} は中心間距離である。また i,j 間の共通法線と x 軸とのなす角を α_{ij} とすると

$$\sin\alpha_{ij} = \frac{y_j - y_i}{R_{ij}} : \quad \cos\alpha_{ij} = \frac{x_j - x_i}{R_{ij}} \tag{8}$$

という関係がある。

ついで接触する 2 粒子 i,j の時間刻み Δt の間の相対変位量増分を共通法線に対する法線方向と接線（せん断）方向に分割して考える必要がある。x 軸と共通法線のなす角がわかっているのでその分回転させて

$$\begin{pmatrix} \Delta u_n \\ \Delta u_t \end{pmatrix} = \begin{pmatrix} \cos\alpha_{ij} & \sin\alpha_{ij} \\ -\sin\alpha_{ij} & \cos\alpha_{ij} \end{pmatrix} \begin{pmatrix} \Delta u_i - \Delta u_j \\ \Delta v_i - \Delta v_j \end{pmatrix} + \begin{pmatrix} 0 \\ r_i \Delta\psi_i + r_j \Delta\psi_j \end{pmatrix} \tag{9}$$

となる。右辺の第 1 項は主軸の回転によって相対変位増分を法線、接線方向に変換したことを表す。第 2 項は接線方向の相対変位が回転によってその間に増加することを表している [7]。ここで Δu_n 等は下坂の図 3.5 にあるように i から j に向かう方向を正に取っている。

3.2 運動方程式の差分近似

接触力そのものは (4) 式で与えられている。これは法線方向、接線方向なのでプログラム上では x, y 成分に直して格納する必要がある。前節のような状況で j 粒子の i 粒子から受ける力の x, y 成分 X_j, Y_j 及び i 粒子の中心の回りのモーメント M_j は α_{ij} だけ逆回転すればいいので

$$\begin{pmatrix} X_j \\ Y_j \end{pmatrix} = \begin{pmatrix} \cos\alpha_{ij} & -\sin\alpha_{ij} \\ \sin\alpha_{ij} & \cos\alpha_{ij} \end{pmatrix} \begin{pmatrix} F_n^{i\to j} \\ F_t^{i\to j} \end{pmatrix} \tag{10}$$

[6] 下坂によると $K_n(\delta_n) = \frac{\pi E}{(1-\sigma^2)(3+4\ln(4a/b))}$ となっているが導出をチェックしていないので、チェックした上式を採用。

[7] 共通法線を x 軸としてそれぞれ円盤を固定して一方だけ回転させて考えるとよい。

及び

$$M_j = -r_j F_t^{i\to j} \tag{11}$$

となる。但し接線力と粒子回転の向きの設定が反対であり、$i \to j$ という添字は i から j に働くということを明示している。ここで $\alpha_{ji} = \alpha_{ij} + \pi$ (図 3.5) を考慮して $(\cos\alpha_{ij} = -\cos\alpha_{ji}, \sin\alpha_{ij} = -\sin\alpha_{ji})$ 接触している粒子の影響を足し合わせると

$$\begin{pmatrix} X_j \\ Y_j \end{pmatrix} = \sum_i \begin{pmatrix} -\cos\alpha_{ji} & \sin\alpha_{ji} \\ -\sin\alpha_{ji} & -\cos\alpha_{ji} \end{pmatrix} \begin{pmatrix} F_n^{i\to j} \\ F_t^{i\to j} \end{pmatrix} \tag{12}$$

及び

$$M_j = -r_j \sum_i F_t^{i\to j} \tag{13}$$

となる。ここで (少なくとも私が) 混乱しがちなのは右辺を加速度と見るか力と見るかである。すなわち 2 体の和でかけるとした際に換算質量で規格化された加速度を加算したい誘惑に駆られる。しかし言うまでもないことであるが右辺は式 (12),(13) にある通り力を加算する。つまり運動方程式はあくまで

$$m_j \frac{d\mathbf{v}_j}{dt} = \mathbf{F}_j + \sum_{i\neq j} \mathbf{F}_{i\to j} \tag{14}$$

である。回転についても同様である。特に回転の場合は 2 体の問題の際も相対的に考えるよりも考えている粒子を固定させ、その固定軸に回りに第 2 粒子の接触によって接線力が働いたと考えるべきである。そうしないと個々の回転軸が固定されないために運動方程式は複雑なものになる。

変位等の時間発展は上記の力とモーメントが求まったために容易に決められる。ここではプログラムにならって Δt^2 の精度を持つスキームに基づき説明する。

まずプログラム中では速度の更新をしている。すなわち

$$\begin{aligned} \mathbf{v}_j(t+\Delta t) &= \mathbf{v}_j(t) + \frac{\Delta t}{2m_j}(3\mathbf{F}_j(t) - \mathbf{F}_j(t-\Delta t)) \\ \omega_j(t+\Delta t) &= \omega_j(t) + \frac{\Delta t}{2I_j}(3M_j(t) - M_j(t-\Delta t)) \end{aligned} \tag{15}$$

となる。ここで $\mathbf{F}_j = (X_j, Y_j)$、$m_j$,$I_j (= m_j r_j^2/2)$ は各質量と慣性モーメントである。ω_j は (回転) 角速度。このスキームが 2 次精度を保証していることは例えば $\frac{\Delta t}{2m_j}(3\mathbf{F}_j(t) - \mathbf{F}_j(t-\Delta t)) \simeq \Delta t \mathbf{F}_j(t)/m_j + \Delta t^2 \dot{\mathbf{F}}_j(t)/(2m_j) + \cdots$ と展開できることで分かる。

次に同じループの中で位置の更新をしている。このとき注意すべきは既に速度の方が更新を完了している点である。従って

$$\begin{aligned} \mathbf{r}_j(t+\Delta t) &= \mathbf{r}_j(t) + \frac{\Delta t}{2}(\mathbf{v}_j(t+\Delta t) + \mathbf{v}_j(t)) \\ \psi_j(t+\Delta t) &= \psi_j(t) + \frac{\Delta t}{2}(\omega_j(t+\Delta t) + \omega_j(t)) \end{aligned} \tag{16}$$

となる。実際 $\frac{\Delta t}{2}(\mathbf{v}_j(t+\Delta t) + \mathbf{v}_j(t)) = \Delta t \mathbf{v}_j(t) + \frac{\Delta t^2}{2}\dot{\mathbf{v}}_j(t) + \cdots$ となっていることは容易に確かめられる。

4　実際のシミュレーション

4.1　スケーリング

前章で線形ばねモデルを採用したメリットの一つとしてスケーリングを使うことが出来る点を指摘した。即ち粒子直径等を 1 としてシミュレーションを行なえばスケール変換によって実際の状況に応じたシミュ

レーションに転換することは容易である。このことで粒子のサイズに応じた時間刻みを考える必要がなくなる。

実際には長さスケールは粒子直径 d（分散がある場合は例えば最大直径とみなす）、質量 m, 重力加速度 g を用いれば次元解析でどういうスケーリングを行なえば良いのかが分かる。例えば時間の次元は $\sqrt{d/g}$, 粘性パラメータ η_n, η_t は $\sqrt{g/d}$,（単位質量当たりの）ばね定数は g/d 等である。プログラム上ではこれらの量の比例定数、即ち無次元化された量しか現れない。

4.2 アルゴリズム

ここでは例題として斜面上の粒子の流れをシミュレートしてみる。実際のプログラムに即して詳細を説明する前におおまかな流れを先に説明してみよう。

まず殆んどの変数等はヘッダーファイルに書き込んである。それを除くとほぼ下坂の図 3.9 に沿っている。主要な相違点は底にも粒子を敷き詰めて、横を周期境界としているために壁との接触判定サブルーティン wcont を使っていない点と周期境界の接触判定を 2 段階にした点である。その他、初期緩和とメインの時間発展を分割した点やデータの書き出し、グラフィックは異なる。

プログラムは Makefile,Header file, program の 3 つの部分から成る。Makefile ではプログラムをコンパイルし、実行する命令をするファイルである（この部分の説明は省略）。Header file ではプログラム上のパラメータの設定及び subroutine 等で使われる多くの共通変数 (common 文等で指定するもの）を外的に与えるものである。プログラムそのものも main routine,subroutines に分かれる。順番に具体的な斜面上の粒子流をシミュレートするプログラムを例にしてその構造を見ていこう。

4.3 ヘッダーファイル

まずヘッダーファイルである。日本語等で注釈がついているのでそれほど理解は難しくないと思う。

```
      implicit real*8 (a-h,k,m,o-z)
      parameter (ni=2000,nj=24,nc=50000)
c      ni: 配列の最大数、nj: 接触点の最大数、nc: セル数の上限
      parameter (rmax=.5d0,w=rmax*dble(60),rmin=0.8d0*rmax)
c      rmin : 最小粒子半径，rmax: 最大粒子半径 （分布はこの間ランダム）
c      粒子直径 1 に規格化。w: 容器横幅
      parameter (so=0.2d0)
c      so: 接線方向の比抵抗力 (Mindline force に起因。 2 次元だから適当）

      parameter (fri=5.d-1)
c      fri: クーロン動摩擦係数
      parameter (ipz=9,c=1.35d0*rmin)
c      ipz: 縦の層数、c: 接触判定のセルの大きさ
      parameter (dt=1.d-3,knn=1.d4,ktt=knn*so)
c      dt: 時間刻、knn: 法線方向ばね定数、ktt: 接線方向ばね定数
      parameter (istep=10000,imaxstep=1000,istep2=500)
c      istep: 初期緩和ステップ、imaxstep: main routine の外回りステップ数
c      istep2: 内回りステップ数
```

```
      common /wep/ r(ni),wei(ni),pmi(ni)
c      r: 粒子半径
c      wei: 粒子質量
c      pmi: 粒子の慣性モーメント

      common /cel/ n,np,idx,idz,ncl(nc)
c      n: 粒子数、np: plate ( 底 ) の粒子数
c      idx: x 方向の配列数、idz: z 方向の配列数
c      w:容器幅, ncl(nc): セルの配列

      common /pos/ x0(ni),z0(ni),qq(ni)
c      x0(i): i 粒子の位置、x 座標
c      z0(i): i 粒子の位置、z 座標
c      qq(i): i 粒子の回転変位

      common /vel/ u0(ni),v0(ni),omega(ni)
c      u0(i): i 粒子の速度、x 座標
c      v0(i): i 粒子の速度、z 座標
c      omega(i): i 粒子の回転角速度

      common /for1/ xf(ni),zf(ni),mf(ni)
c      xf(i): i 粒子の受ける力 : x 成分
c      zf(i): i 粒子の受ける力 : z 成分
c      mf(i): i 粒子の受ける力のモーメント

      common /for2/en(ni,nj),et(ni,nj),je(ni,nj)
c      en(i,j): i 粒子の法線方向の接触力
c      et(i,j): i 粒子の接線方向の接触力
c      je(i,j): 接触点番号配列

      common /dpm/ u(ni+4),v(ni+4),f(ni+4)
c       u(i): i 粒子の x 変位増分
c       v(i): i 粒子の z 変位増分
c       f(i): i 粒子の回転変位増分

       common /slope/ theta,gc,gs,pi
c      theta: 傾斜角、gc=cos(theta),gs=sin(theta)
c      pi: 円周率
       common /viscous/ etan,etat
c      etan: 法線方向の粘性率, etat: 接線方向の粘性率

c      全て物理量は無次元化されている。
```

まず 1 行目は変数の型宣言である。a-h,k,m,o-z を頭に抱く変数名は全て倍精度の変数としている。逆に言えば i,j,l,n で始まる変数を整数に割り振っている。 2 行目は粒子数の上限、接触点の上限、セル数の上限等を与えている。 1 個の粒子に隣接しうる接触数の上限は 2 次元では 20 を越えることは (まず) ない。また 24 としているのは四方の壁があった場合に接触点番号列を区別するためである。本プログラムでは必要はない。

本プログラムでは粒子半径は上限 rmax(= 0.5) と下限 rmin の間をランダムに分布しているものとした。また横幅は最大半径に比べていくつという風に設定してある。また so は接線方向と法線方向のばね定数の比である。fri はクーロンの動摩擦係数で、およそ 0.5 の値を持つ。ipz は初期充填の粒子層の数、c は粒子の接触判定に用いるセルのサイズであり、後述の様に (同一のセルに 2 つの粒子中心が入らないために) 最小粒子半径の $\sqrt{2}$ 倍以下である必要がある。

減衰振動では時間刻み dt は単位質量当たりのばね定数 $k(=\omega_0^2$: ω_0:固有角振動数) と質量の上限 m を用いて $dt < const./\sqrt{k}$ が要求されることが想像がつく。ここで $c = 2$ であることも知られている (らしい)。今 $ktt = 0.2 * knn$ といった具合に接線方向のばね定数 ktt が法線方向のそれ knn に比べて小さく (従って knn で見積もらないといけない)、最大質量が 1 に規格化されていることから、およそ $dt < 2 \times 10^{-2}$ である。しかし実際には $dt = 3 \times 10^{-3}$ で既におかしな振舞いをするので DEM の上限はもう少し厳しく見積もる必要がある。

時間発展に関しては初期緩和のステップ数 istep, 実際のシミュレーションの際の時間発展のステップ数を指定するパラメータ imaxstep,istep2 があり、シミュレーションの最終的な時間は $t = dt * (istep + imaxstep * istep2)$ となっている。また istep2 毎に図を描いたりデータを出力したりするようになっている。

後、common 文で指定された共通に現れる変数があるがそれについては注釈があるので説明を割愛する。

4.4 メインルーティン

この節ではメインルーティンの説明をする。

```
        include 'dem1.h'

        real x00,z00,r00,xm,zmm,w00

        r00=sngl(rmax)
        w00=sngl(w)

        IF (PGOPEN('?') .LT. 1) STOP
        CALL PGENV(-r00, w00+r00, -2*r00, 0.7*w00,  0,  -2)

        call fposit
        call inmat
        call init
        call plot

        t=0.d0
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c  Until it=step, the system is in an initial relaxation stage.
```

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        do  it=1,istep

          t=t+dt

         call ncel

         do i=1,n
           xf(i)=0.d0
           zf(i)=0.d0
           mf(i)=0.d0
          call pcont(i)
         enddo

         call nposit(1)

         if (mod(it,istep/50).eq.0) then
          write(6,*) it
          call plot
        endif

        enddo
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c   The initial relaxation is over at t=t0.
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

        t0=t

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c  The outer loop of the main loop (1 to imaxstep)
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
        do it=1,imaxstep

         write(6,*) it,istep+(it-1)*istep2,' time= ',t

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c The inner loop of the main loop (1 to istep2)
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
         do  ii=1,istep2

          t=t+dt

         call ncel
```

```
          do i=1,n
           xf(i)=0.d0
           zf(i)=0.d0
           mf(i)=0.d0
           call pcont(i)
          enddo

          call nposit(2)

          enddo

ccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c   The internal loop is over.
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

          call plot

         enddo

cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c   The outer loop is over.
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

         call pgclos
         end
```

容易に読みとれる様に同じ様な命令を 2 回に渡って繰り返している。これは初期緩和と実際のシミュレーションを分割したためにプログラムに不要な複雑さが現れたことになる。この辺りを見やすくするのは容易だが不要な条件分岐が現れてかえって実行速度が遅くなっては意味がないのでとりあえず現状のままにしておく。

まずヘッダーファイルを読み込むために include 'dem1.h' という 1 行が入っている。次に単精度の実数型の変数がいくつか導入されているが、これは pgplot（単精度変数しか扱えない）というグラフィックソフトを用いるための措置である。

```
         IF (PGOPEN('?') .LT. 1) STOP
         CALL PGENV(-r00, w00+r00, -2*r00, 0.7*w00,  0,  -2)
```

の 2 行も pgplot を起動させるためのものである。pgplot については例えば http://www.fluidlab.naoe.t.u-tokyo.ac.jp/softwares/pgplot-j.html を参照のこと。pgenv というのは環境の領域を定めているが、粒子半径や横幅を変えても自動的に領域が変更されるようにしてある。また pgplot を終了する際に call pgclos で締めくくる必要がある。

初期緩和に入る前に初期条件の設定その他で種々の準備が必要になる。これは最初の do loop（コメント行の次）までに当たる。fposit は初期の粒子配置を定めるサブルーティンであり、inmat はヘッダーファイルで決定していない物理量を定めている（これはパラメータ文の中で関数を読み込んだ計算が出来ないことに起因している。例えば $\pi = 4Arctan(1)$ はパラメータ文で宣言できない）。init はセルとか粒子速度等

の初期化のサブルーティン、plot はグラフィックのサブルーティンである。もちろん $t = 0$ としたのは初期時刻の設定である。

初期緩和ではまず ncel というサブルーティンを読み込んで粒子をセルに登録をしている。次いで粒子の全てに対して力等を初期化し、粒子間の接触判定を行なっている (subroutine pcont(i) : 粒子番号 i を pcont に渡していることに注意)。nposit は時間発展のサブルーティンである。

```
 if (mod(it,istep/50).eq.0) then
  write(6,*) it
  call plot
endif
```

は (なくてもいいが) 初期緩和の間に５０回絵 (とステップ数) を描けという命令である。

実際のシミュレーションでも初期緩和の時間発展と殆んど同じである。違いと言えばデータの格納 (ここでは行なっていないが)、描画等のために時間発展のループを 2 重構造にしたこと、初期緩和の終った時刻 t0 を記憶させたこと等位である。後は説明の必要はないであろう。

4.5 subroutine fposit

初期配置を決めるサブルーティン fposit は以下の通りである。

```
subroutine fposit

include 'dem1.h'

delr=1.d-3
h=0.d0
rn=rmax+delr

ipx=idint(w/2.d0/rn)

do  i=1,ipz
  if (mod(i,2).eq.0) then
     dx=0.d0
     ip=ipx+1
  else
     dx=rn
     ip=ipx
  endif

 do j=1,ip

    n=n+1
    x0(n)=2.d0*rn*dble(j-1)+dx
    if (x0(n).gt.w) x0(n)=x0(n)-w
    z0(n)=dsqrt(3.d0)*rn*dble(i-1)+rn+rmax
    h=dmax1(h,z0(n))
```

```
   r(n)=rmin+(rmax-rmin)*arand(ir)
  enddo
enddo

np=dint(w/(2.d0*rn))+1

do i=1,np
  x0(n+i)=2.d0*rn*dble(i-1)
  z0(n+i)=0.d0
  r(n+i)=rmax
enddo

n=n+np

do i=1,n
 u0(i)=0.d0
 v0(i)=0.d0
 omega(i)=0.d0
enddo

write(6,*) 'number of particles:',n,ipx,np

idx=idint(w/c)+1

idz=idint(h/c)+20

if (idx*idz.gt.nc) then
  write(6,*) 'ncl is over!', idx*idz,nc
  stop
endif

end
```

fposit ではまず header file dem1.h を読み込んだ後、delr、$rn = rmax + delr$ という変数を導入する。この delr は小さな値であり rn は初期粒子配置の間隔を表すパラメータとなっている。つまり粒子が全て最大半径に等しい場合には接触半径よりわずかに大きな間隔で粒子が並ぶことを意味する。次に ipx は容器幅を rn の倍、即ち有効粒子直径で割ったものを四捨五入整数化したものである。

この fposit では基本的に粒子が 3 角格子を組んだ状態から出発させている。従って最初の do loop ($i = 1$ から ipz まで) は鉛直方向の層の数に対応して $np = \cdots$ の式の前まで i を回している。その際、まず偶数番目の粒子層にある場合に最も左にある粒子の中心の x 座標が 0 になるように $dx = 0$ と設定し、その場合には層内の粒子数も ipx より 1 大きくなるとしている。また奇数層では dx が有効粒子半径分ずれて設定され層内の粒子数も $ip = ipx$ となっている。

次の j に対する do loop では実際に粒子を x 方向にずらしながら設置している。ここでは粒子番号 n が導入され、粒子を置いていくたびに番号を一つづつ増やしている。また粒子位置の x 座標 $x0(n)$ は dx だ

けずらした状態から有効直径づつずらして設置し、鉛直座標 $z0(n)$ は底の粒子 (中心の鉛直座標は 0) 分、ずらして 3 角格子を組むように隣接する層間で有効半径の $\sqrt{3}$ 倍だけずれて粒子を置くようにしてある。また粒子半径は rmax,rmin の間を一様分布するように乱数を発生させて (arand(ir)) 決めている。

次に np は底に設定する粒子の数である。粒子半径は rmax に揃えて有効直径分ずらしながら鉛直位置 0 に粒子を置いてある。また粒子の初期の速度、角速度も 0 に設定している。idx は x 方向のセル数の上限、idz は鉛直方向のセル数の上限で最初のパックされた状態よりも 20 程度余計に配列を用意してある。

4.6 subroutine inmat

サブルーティン inmat は初期の物質パラメータを定めるためのものである。

```
include 'dem1.h'

 do i=1,n
  wei(i)=r(i)*r(i)*4.d0
  pmi(i)=wei(i)*r(i)*r(i)/2.d0
 enddo

 pi=datan(1.d0)*4.d0
 theta=pi/6.d0
 gs=dsin(theta)
 gc=dcos(theta)

 etan=2.d0*dsqrt(knn)
 etat=etan

 end
```

ここで粒子の質量、慣性モーメントを与えている。スケーリングの項を思い出して頂くと質量は $rmax(= 0.5)$ の半径を持つ円盤が 1 になるように規格化されているので上のような式になっている。また θ は斜面の傾斜角でここでは $\pi/6$ としている。言うまでもないが $gs = \sin(\theta), gc = \cos(\theta)$ はそれぞれ斜面に沿った、鉛直な重力加速度の成分 ($g = 1$ と規格化) である。

次いで法線方向の粘性パラメータ η_n=etan を臨界減衰条件から定めている。この臨界減衰条件は次の初等的考察から求まる。粒子対が接触している間は

$$\frac{d^2u_{ij}}{dt^2} = -k_n u_{ij} - \eta_n \frac{du_{ij}}{dt} : \quad F_{ij} = m_{ij}\frac{d^2u_{ij}}{dt^2} \tag{17}$$

に従うので、その特性方程式 $\lambda^2 + \eta_n\lambda + k_n = 0$ が重根を持つ場合に臨界減衰となる。従って

$$\eta_n = 2\sqrt{k_n} \tag{18}$$

が臨界減衰条件である。この臨界減衰条件はしばしば用いられるが実際には根拠はなく、目安に過ぎない。また接線方向でも同様の考察が成り立つがちょっと複雑になるので本プログラム中では $\eta_t = \eta_n$ としている。(同一の粒子であれば接線方向の運動方程式の臨界減衰条件は $\eta_t = \sqrt{2k_t} = \sqrt{s/2}\eta_n$ である。従って $\eta_t = \eta_n$ は明らかに過剰に回転運動にブレーキをかけている。しかし概説の章で述べた通り、DEM は転がり過ぎる嫌いがあるのでこのままとする。また Thompson and Grest[16] によって $\eta_t = \eta_n/2$ とすると法線方向と接線方向が同じ反発係数になる点も指摘されている。)

4.7 subroutine init

subroutine init では接触力を初期化している。

```
 subroutine init

 include 'dem1.h'

do i=1,n+4
   u(i)=0.d0
   v(i)=0.d0
   f(i)=0.d0
 enddo

do i=1,n+4
 do j=1,nj
  en(i,j)=0.d0
  et(i,j)=0.d0
  je(i,j)=0
 enddo
enddo

end
```

ここで $u(i), v(i), f(i)$ はそれぞれ i 粒子の x 変位増分、z 変位増分、回転変位増分であったことを思いだそう。また do loop で $n+4$ まで loop を回しているのは壁があった場合にそれぞれ壁を $n+1, \cdots, n+4$ と設定できるようにするためである。今の場合は n で良い。また $en(i,j)$ は i 粒子における接触点 j の法線方向の接触力 $et(i,j)$ は接線方向の接触力、$je(i,j)$ は接触点番号配列である。

4.8 subroutine ncel

サブルーティン ncel は粒子をセルに登録するためのものである。fposit で説明した通り、セルは 2 次元 $idx \times idz$ になっているが順序に少し気をつけると 1 次元化できる。ここではセルを左下から x 軸方向に順に番号づけし、横幅を越えたら 1 段上のセルに移行するという風に 1 次元化を行なっている。

```
 subroutine ncel

 include 'dem1.h'

do ib=1,idx*idz
    ncl(ib)=0
enddo

 do i=1,n
  ib=idint(z0(i)/c)*idx+idint(x0(i)/c)+1
  ncl(ib)=i
```

```
      enddo

       end
```

ここでまず全てのセルを初期化 $(ncl(ib) = 0)$ している。次に粒子の位置からセルの番号を割り出す式 $ib = idint(z0(i)/c) * idx + idint(x0(i)/c) + 1$ を用いて粒子の番号をセル ib に登録している。ここで同一セルに 2 つの粒子が入らない様に設定してあることに注意されたい。

4.9 subroutine pcont

サブルーティン pcont は粒子 i の接触判定を行なうためのものである。このプログラムでは横方向に周期境界条件を課しており、その点を考慮するために 2 段階にプログラムを分割している。即ち、横壁を通して周期境界の影響が出る場合を予め区別しておくという措置を取っている。当初、もっと凝ったプログラムを想定していたが結局エラーが取れず、下坂さんに相談した結果、最も primitive な考え方に基づいたプログラムになっている。

```
         subroutine pcont(i)

         include 'dem1.h'

        xi=x0(i)
        zi=z0(i)
        ri=r(i)

        call pcont2(xi,zi,i,ri)

c-------------------------
c  left boundary
c-------------------------

         if (xi-rmax-ri.lt.0.d0) then
          xi=xi+w
          call pcont2(xi,zi,i,ri)
          xi=xi-w
         endif
c------------------------------
c    right boundary
c------------------------------
         if (xi+rmax+ri.gt.w) then
          xi=xi-w
          call pcont2(xi,zi,i,ri)
          xi=xi+w
         endif

         end
```

ここでまず配列変数をただの数に置き換えている。基本的に不要な措置だが参照の際に高速化が可能になる（らしい）。一旦全部の接触判定をして壁の接触判定を別に行なっているのも無駄なようだが、見落としがあるといけないのでそのままにしておく。各自無駄を省くように工夫できるであろう。

壁の接触判定は例えば左壁であれば粒子の最左端の x 座標が xi-ri であり、それと接触し得る粒子中心の座標はそのままであれば xi-ri-rmax であることから、その値が負になった場合に周期分ずらして（右端近くに存在する粒子との）接触判定を行なっている。最後に粒子位置を元に戻していることに注意。

いずれにしても実際の接触判定は粒子位置の情報 xi,zi と粒子番号 i、その半径 ri を参照したサブルーティン pcont2 で行なっている。

4.10 subroutine pcont2

サブルーティン pcont2 は粒子の接触判定の実際を行なっている。このサブルーティンは DEM の中核の一翼を担うと同時に最もわかりにくい部分でもある。

```
        subroutine pcont2(xi,zi,i,ri)

        include 'dem1.h'

        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
        if (lrg.ge.idx) lrg=idx-1

        do  888 lz=lun,lup
         do  88 lx=llf,lrg

           ib=lz*idx+lx+1

           j=ncl(ib)

           if ((j.eq.0).or.(j.eq.i)) goto 88

ccccccccccccccccccccccccccccc
c     particle registration
cccccccccccccccccccccccccccccc
          do jj=1,nj-4
            if (je(i,jj).eq.j) then
             jk=jj
             goto 70
            endif
```

```
          enddo

         do jj=1,nj-4
           if (je(i,jj).eq.0) then
              jk=jj
              je(i,jj)=j
              goto 70
            endif
          enddo

ccccccccccccccccccccccccc
c  registration is over.
cccccccccccccccccccccccccc

 70       xj=x0(j)
          zj=z0(j)
          rj=r(j)

         gap=dsqrt((xi-xj)*(xi-xj)+(zi-zj)*(zi-zj))
         if (gap.le.(ri+rj)) then

           if (i.gt.j) then
             ac=(xj-xi)/gap
             as=(zj-zi)/gap
             j0=0
             do jj=1,nj-4
              if (je(j,jj).eq.i) then
               j0=jj
               goto 50
              endif
             enddo

 50         call actf(i,j,jk,as,ac,gap)
            en(j,j0)=en(i,jk)
            et(j,j0)=et(i,jk)
            j0=0
          endif
c        i>j の if 文の終り
        else
         en(i,jk)=0.d0
         et(i,jk)=0.d0
         je(i,jk)=0
        endif
c       gap.le.(ri+rj) の if 文の終り
```

```
 88       continue
 888    continue

        end
```

まず最初の lup,lun,llf,lrg は(それぞれ up,under,left,right の意味)粒子 i の接触する粒子が所属し得るセルを探す範囲を定めている。セルの大きさは一辺 1.35*rmax なので接触相手粒子の(最も離れ得る)中心位置、例えば鉛直上方であれば $zi + 2.d0 * rmax$ の位置をセルのサイズで割っておく必要がある(下坂の図 3.8 参照)。またセルは限られた領域でのみ定義されているのでシステムサイズをはみ出すような場合はそこにセルが存在しないという旨を明記してある。

次に 888 行及び 88 行の do loop である。これらの loop はこのサブルーチンの最後まで渡っている。言うまでもないことだが lz.lx はそれぞれ鉛直方向、水平方向のセルの index を表し、その下限から上限までを回している。

最初にセルの index を 1 次元化された ib に書き直している (subroutine ncel 参照)。次の $j = ncl(ib)$ はセル ib に格納された粒子を呼び出している。subroutine ncel を参照すれば分かる通り粒子がそのセル内に存在しない場合には 0 が格納されており、また粒子 i そのものが格納されている場合も含めて i の接触の相手たりえないので、その場合は次のセルを探すために 88 行に飛んでいる。

次の jj の loop からいよいよ粒子の登録が始まる。jj の上限が nj-4 になっているのは壁の 4 自由度を予め区別しているためである。本プログラムでは-4 は必要ないが、壁を導入した場合にそのまま援用できることを考えてそのままにしてある。まず je(i,jk) が粒子 i の接触点 jk に接触相手の粒子 j を登録すべき配列変数であることを思いだそう。既に je(i,jj) に粒子 j が登録されていたなら接触点を jk=jj と記憶して 70 行に進む。この jk は接触力を計算するサブルーチン actf 等で必要になる。次にまた各接触点に対して登録されていないかどうかを調べて登録されていない(je(i,jj)=0)であれば jk=jj、je(i,jj)=j として粒子 j を登録する。これで登録が完了する。

70 行からは粒子 j の位置 xj,zj、半径 rj を書き出してから, 粒子 i,j の中心間距離 gap=$R_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ を用いて (3.3) の判定を実行する。gap= $R_{ij} \le r_i + r_j$=ri+rj であれば実際に接触していることになる。接触は対で行なうので i>j の場合のみ考えれば良い。また ac=$\cos\alpha_{ij}$, as=$\sin\alpha_{ij}$ である。次に j の接触相手を見て、それが i であった場合に j0(default で 0 が入るようにしてある)に接触点を登録してから接触力を計算するサブルーチン actf を呼び込んでいる。因みに actf は接触粒子の番号 i,j、接触点番号 jk, 共通法線の $\cos\alpha_{ij}$, $\sin\alpha_{ij}$ と接触距離 $R_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ の関数である。今接触点番号 jk=j0(=jj) なのでその接触点における接触力は粒子 i,j で互いに等しい。これらが接触した場合の計算である。

先に触れた様に接触していない場合 $R_{ij} > r_i + r_j$ であれば接触力及び接触点番号列も 0 を代入しないといけない。これでこのサブルーチンの計算は完了している。

4.11 subroutine actf

このサブルーチンでは接触している場合に接触力を計算している。このサブルーチンも pcont2 と並んで DEM の中核をなす(わかりにくい)部分である。

```
        subroutine actf(i,j,jk,as,ac,gap)

        include 'dem1.h'

          ri=r(i)
```

```
          rj=r(j)
          dis=ri+rj-gap
          rwei=wei(i)*wei(j)/(wei(i)+wei(j))

        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)

        veln=(u0(i)-u0(j))*ac+(v0(i)-v0(j))*as
        velt=-(u0(i)-u0(j))*as+(v0(i)-v0(j))*ac+ri*omega(i)+rj*omega(j)
c--------------------------------------------------
        if (en(i,jk).eq.0.d0) then
          if (un.ne.0.d0) us=us*dis/un
          un=dis
         endif
c--------------------------------------------------
        visn=rwei*etan*veln
        vist=rwei*etat*velt

        en(i,jk)=rwei*knn*un+en(i,jk)
        et(i,jk)=rwei*ktt*us+et(i,jk)

           hn=en(i,jk)+visn
           ht=et(i,jk)+vist

       if (hn.lt.0.d0) then
        en(i,jk)=0.d0
        et(i,jk)=0.d0
        je(i,jk)=0
       else
         if (dabs(ht)-fri*hn.gt.0.d0) then
           ht=fri*dsign(hn,ht)
         endif
           xf(i)=-hn*ac+ht*as+xf(i)
           zf(i)=-hn*as-ht*ac+zf(i)
           mf(i)=mf(i)-ri*ht
              xf(j)=hn*ac-ht*as+xf(j)
              zf(j)=hn*as+ht*ac+zf(j)
              mf(j)=mf(j)-rj*ht
         endif
         end
```

ここではまず ri,rj の登録、そして i,j ペアの換算質量 m_{ij} =rwei を求めている。次に dis は粒子のオーバーラップの距離としている。

本プログラムでは 2 体運動の相対加速度、相対速度、相対位置を求めているので実際の合力を計算する際には (14) 式のように換算質量をかけた相対的な力を加算していく必要がある。そのためプログラム中

en(i,jk),et(i,jk),visn,vist に換算質量分かけてあることに注意されたい。

ここで下坂のプログラムでは if (en(i,jk).le.0.d0) en(i,jk)=1.d-6 という (類) の記述があったが、接触力は正なので、ここはコメントアウトしても構わない。むしろその記述を入れると人工的にスキームが安定化する可能性があるのでない方がいいであろう。ここの条件にかかるようであれば他の所でプログラムミスがあったと考えた方がよい。

次いで (9) によって相対変位の法線方向、接線方向を求めている。また相対速度についても同様に法線方向 veln、接線方向 velt を求める。visn,vist はそれぞれ法線方向、接線方向の粘性力である。下坂のプログラムでは un,us を用いて粘性力を計算していたが、un は $(t-\Delta t, t)$ 間の相対変位増分であるため un/dt 等は $t-\Delta t/2$ での速度と考えられる。従って速度に対して veln,velt を求めるべきであろう。

また

```
c--------------------------------------------------
        if (en(i,jk).eq.0) then
          if (un.ne.0.d0) us=us*dis/un
          un=dis
         endif
c--------------------------------------------------
```

は一見わかりにくいが、これは前のステップで粒子が接触していない場合にどのように相対変位を計算するかということを示している。ここでは前に接触していない場合に法線方向の相対変位増分をオーバーラップの距離、また接線方向については接線方向の変位にオーバーラップ距離分のウェートをつけている。

弾性力については力の増加分を加算しているが、粘性力には加算がない。従って (17) との比較には注意が必要である。そして弾性力と粘性力の合力の法線成分、接線成分を hn,ht とした。

```
      if (hn.lt.0.d0) then
        en(i,jk)=0.d0
        et(i,jk)=0.d0
        je(i,jk)=0
       else
       中略
       endif
```

の条件分岐はこのサブルーチンから出る際に弾性的な接触力と接触点情報をクリアしておくという命令になっている。粒子が離れようとしている時には相対的に離れる方向で運動しており、注意が必要となる。高密度の場合は接触状態から離れてもすぐに接触状態に戻る事が期待されるが、その場合は上記の if (en(i,jk).eq.0) then という条件に該当する。

トータルの接触力の法線成分 hn が負になったときはもはや接触していない。その場合に接触力を 0 にし、接触点の情報をクリアしてからこのサブルーチンの外へ出る。接線方向では (4) 式にあるようにクーロン滑べりの条件を課す必要がある。この条件が

```
        if (dabs(ht)-fri*hn.gt.0.d0) then
           ht=fri*dsign(hn,ht)
        endif
```

である。ここで Fortran では $sign(a_1, a_2) = |a_1|$ for $a_2 > 0$ and $-|a_1|$ for $a_2 < 0$ であることに注意されたい。

最後に (12),(13) (のうちの i,j ペアの) 式に従って粒子 i に働く力を計算する。また反作用によって j 粒子も動かす。特に注意すべきは粒子の回転の反作用である。接触点での反作用に従って回転するので同方向の力が働くのである。

4.12 subroutine nposit

このサブルーチンでは粒子の時間発展を実行する。初期緩和の際 (ii=1) は底に垂直な重力がかかっているのに対して実際のダイナミックス (ii=2) では斜面方向にも重力がかかる。この条件の違いを引数 ii で区別する。

```
      subroutine nposit(ii)

      include 'dem1.h'

     real*8 vdum(ni),udum(ni)
     real*8 zfdum(ni),xfdum(ni)
     real*8 mfdum(ni),omegadum(ni)

      if (ii.eq.1) then
         ggc=1.d0
         ggs=0.d0
       else
         ggc=gc
         ggs=gs
       endif

      do i=1,n-np
        v0(i)=v0(i)+dt*((zf(i)-wei(i)*ggc)/wei(i)*1.5d0-zfdum(i)/2.d0)
        u0(i)=u0(i)+dt*((xf(i)+wei(i)*ggs)/wei(i)*1.5d0-xfdum(i)/2.d0)
        omega(i)=omega(i)+dt*(mf(i)/pmi(i)*1.5d0-mfdum(i)/2.d0)

        v(i)=dt*(v0(i)+vdum(i))/2.d0
        u(i)=dt*(u0(i)+udum(i))/2.d0

        z0(i)=z0(i)+v(i)
        x0(i)=x0(i)+u(i)
        f(i)=dt*(omega(i)+omegadum(i))/2.d0
        qq(i)=qq(i)+f(i)

        if (x0(i).lt.0.d0) x0(i)=x0(i)+w
        if (x0(i).gt.w) x0(i)=x0(i)-w

        vdum(i)=v0(i)
        udum(i)=u0(i)
        zfdum(i)=(zf(i)-wei(i)*ggc)/wei(i)
```

```
  xfdum(i)=(xf(i)+wei(i)*ggs)/wei(i)
  mfdum(i)=mf(i)/pmi(i)
  omegadum(i)=omega(i)

enddo

end
```

実際には (15),(16) 式で時間発展をする。ここでは dum と名のついたダミーの配列を用意してそこに 1step 前の情報を入れてある。(15),(16) 式の説明で注意した通り、1 ステップ前と現在の力を計算して速度及び角速度を決めているのに対して、粒子の移動には直前で更新された速度を使っている点である。また x 方向の位置に対しては周期境界を考慮して基本領域に入る様にしてある。また何回も繰り返すが (14) 式から、加速度を求める際には自分の質量で割っておかないといけない。ここに換算質量の現れる余地はない。

4.13 subroutine plot

このサブルーチンはリアルタイムシミュレーションのグラフィックを描く部分である。ソフトに依存する部分なので違うソフトを使う場合はこのサブルーチンを削除すればよい。

```
subroutine plot

include 'dem1.h'

real x00,z00,r00,xm,zmm,w00,ennn,rr

 call pgbbuf
 call pgeras

  do i=1,n
    x00=sngl(x0(i))
    z00=sngl(z0(i))
    r00=sngl(r(i))
    if (mod(i,10).eq.0) then
       call pgsfs(1)
       ic=mod(i/10,8)+1
       call pgsci(ic)
       call pgsfs(1)
    else
      call pgsfs(2)
      call pgsci(1)
     endif
    call pgcirc(x00,z00,r00)
   enddo
```

```
      call pgebuf
     end
```

ここではpgplotを用いてリアルタイムでダイナミックスを追っている。pgplotについてはhttp://www.fluidlab.naoe.t.u-tokyo.ac.jp/softwares/ にいくつかの解説サイトが紹介されている。特に http://www.fluidlab.naoe.t.u-tokyo.ac.jp/softwares/pgplot-j.html が参考になるであろう。ここで用いているのは PGBBUF,PGEBUF（画像出力コマンドの内部バッファへの保存を開始する。対応する PGEBUF が呼び出される）を使ってとぎれなくアニメを作る。また pgeras は以前描いたものを消す（そうしないと重ね描きになる）コマンド。pgsfs は塗りつぶしが引数 1, 枠だけ描くのが 2 とするコマンド．pgsci は色を指定するコマンドである。また pgcirc は中心の x,y 座標と半径を指定して円を描くコマンドである。

PGPLOT は単精度の変数しか受け付けないので型変換が必要である。またそのまま描くと動きがわからなかったので１０個おきに採色した粒子を置いてみた。初期には斜めに並ぶ。

この他、粒子の速度を表示したり、（接触点における）接触力を表示したり、力の伝播を表示したりして様々なアニメが作れる。各自マニュアルを見て試してみると良い。

5 応用

ここで紹介したのは基本プログラムであるが状況を変えることで様々な応用が効く。簡単に紹介する。

5.1 壁の導入

壁の導入は容易である。基本的な考え方は下坂のプログラムを参照されたい。このときは底からスリップして、粒子層全体の流動性が高まる。このように境界条件で結果がかなり変わって来る。

以下で壁の物性（弾性率、摩擦係数等）は粒子と同じとして、必要な変更点を列挙しよう。

5.1.1 メインルーチン

ここでは壁の接触判定の subroutine wcont(i) を使うことを明示する。本プログラムでは 2 回変更の必要があることに注意。従ってメインルーチンの変更近傍を書き出すと

```
      do i=1,n
        xf(i)=0.d0
        zf(i)=0.d0
        mf(i)=0.d0
       call wcont(i)
c     壁との接触判定
       call pcont(i)
      enddo
      call nposit(1)
```

となる。

5.1.2 subroutine fposit

底の粒子がなくなるので（ここでは既にコメントアウトしてあるが）

```
c         do i=1,np
c           x0(n+i)=2.d0*ratio*rn*dble(i-1)
c           z0(n+i)=0.d0
c           r(n+i)=rmax*ratio
c         enddo
c         n=n+np
```

は削除またはコメントアウト。また粒子位置が底の粒子分下がるので $z0(n) = dsqrt(3.d0) * rn * dble(i - 1) + rmax$ と変更しておく。

5.1.3 subroutine wcont

サブルーチン wcont は粒子半径より粒子中心が下に来た場合に以下の様に設定する。

```
      subroutine wcont(i)

      include 'dem1.h'
      xi=x0(i)
      zi=z0(i)
      ri=r(i)
      jk=nj-3
      j=n+1
      if (zi.lt.ri) then
        as=-0.1d0
        ac=0.d0
        gap=dabs(zi)
        je(i,jk)=n+1
        call actf(i,j,jk,as,ac,gap)
      else
        en(i,jk)=0.d0
        et(i,jk)=0.d0
      endif
      end
```

ここで粒子の接触角が x 軸正方向に-90 傾いているから as=-1.d0,ac=0.d0 となる。また壁に粒子番号 n+1 を与え、接触点番号 jk も粒子間の最大の数 nj-4 より 1 大きくしている。

5.1.4 subroutine actf

サブルーチン actf の変更は壁と粒子を区別するために最初の部分で

```
       ri=r(i)
       if ((j-n).le.0) then
         rj=r(j)
         rwei=wei(i)*wei(j)/(wei(i)+wei(j))
        else
```

```
      rj=0.d0
      rwei=wei(i)
     endif
```

となるのと最後の所で

```
       xf(i)=-hn*ac+ht*as+xf(i)
       zf(i)=-hn*as-ht*ac+zf(i)
       mf(i)=mf(i)-ri*ht
       if ((j-n).le.0) then
          xf(j)=hn*ac-ht*as+xf(j)
          zf(j)=hn*as+ht*ac+zf(j)
          mf(j)=mf(j)-rj*ht
       endif
```

と反作用に if 文が入る点が異なる。

5.1.5 subroutine nposit

時間発展での注意事項は可動粒子の番号の上限が n となっている点である。固定粒子はない。

5.1.6 subroutine plot

底を描くために宣言の最後のところから始まりにかけて

```
    real x02(2),z02(2)

      call pgbbuf
      call pgeras

       x02(1)=0.
       x02(2)=sngl(w)
       z02(1)=0.
       z02(2)=0.
       call pgline(2,x02,z02)
```

といった具合に変更する。

5.1.7 注意

この問題は例題としては適切ではない。というのは計算してみれば分かるが、底がスリップするせいで粒子速度がほぼ時間に比例して増大する。従って定常状態はなく、やがて数値計算自体は発散する。これは粒子が高校の物理で予測された通りの振舞をしていることを意味する。一方、底に固定粒子を導入すると流動はずっとおさえられ、速度の増加も線形関数よりずっと穏やかかつ不規則である。こちらが完全に定常化するかどうかはわからないし、難しい様な気もするが、固定蓋をして閉じ込めた状態にすると定常化し、空気の影響で目詰まりを起こしたパイプ中の粒子流と同様の振舞をすることが知られている。

5.2 振動層

振動層も容易にシミュレーション出来る。通常横方向を周期境界とせず、実際の壁を置く。また容器に固定したフレームで考えると(実際の時間発展の箇所で)、重力加速度が時間変化するようにすれば良い。具体的な変更については殆んど自明なので省略。

5.3 引力の効果

既に概略の章で述べた様に引力を含めることで実に多様な現象に DEM を応用出来る様になる。ここでは単純に水和力のような粒子の外側に働く微弱な引力を含めることを考えてみる。粒子間の水は架橋によって引っ張り力が生じるので接触対が相対的に離れようとする場合にのみ力が働くとしてモデル化をする。この例題では現実的な水和力をモデル化するというより最も単純な引力効果を採り入れたモデルを紹介する。

ここで考えている引力は粒子 i,j の中心間距離 R_{ij} とそれぞれの半径 r_i, r_j の間に $r_i + r_j \le R_{ij} \le range * (r_i + rj)$ を満たし、かつ i,j が相対的に離れようとする場合に

$$f_{ij} = -k' R_{ij} \exp(-R_{ij}^2/\xi^2) \tag{19}$$

という引力が働くとする。尚、プログラム中では $\xi = 1$ としている。

5.3.1 ヘッダーファイル

ヘッダーファイルには引力の相互作用レンジとその強さを陽に書く。そのために例えば

```
parameter (range=1.2d0,att=1.d-4)
```

という1行を加えれば良い。ここで range は従来の斥力コアの大きさに対する引力の及ぶ範囲であり、att は斥力に対する引力の比 (k'/k_n) である。非常に小さい様に思えるが線形ばねとしたときにばね定数1としたことになる。

5.3.2 subroutine pcont

相互作用レンジが変更したので、接触判定に影響が出る。subroutine pcont に関しては

```
      if (xi-rmax-ri.lt.0.d0) then
      if (xi+rmax+ri.gt.w) then
```

の2行がそれぞれ

```
      if (xi-range*(rmax+ri).lt.0.d0) then
      if (xi+range*(rmax+ri).gt.w) then
```

に変更される。

5.3.3 subroutine pcont2

subroutine pcont2 でも接触条件が変化したことに伴い

```
       if (gap.lt.(ri+rj)) then
```

が

```
if (gap.lt.range*(ri+rj)) then
```

に変更される。

5.3.4 subroutine actf

ここではかなりの変更がある。まず

```
en(i,jk)=rwei*knn*un+en(i,jk)
et(i,jk)=rwei*ktt*us+et(i,jk)
   hn=en(i,jk)+visn
```

を例えば

```
if (dis.ge.0.d0) then
 en(i,jk)=rwei*knn*un+en(i,jk)
 et(i,jk)=rwei*ktt*us+et(i,jk)
endif
 if ((dis.lt.0.d0).and.(gap.lt.gapdum)) then
   hn=en(i,jk)+visn-att*rwei*knn*gap*dexp(-gap*gap)
 else
   hn=en(i,jk)+visn
 endif
```

といった具合に変更する。ここで gapdum は 1 ステップ前の粒子間距離 gap である。つまりモデル設定の条件に合致するように相対距離が増加する場合に引力が働くとする。また dis が負であるという条件は斥力コアの外側でのみ引力が働くことを意味する。尚、引力の効果は直接 en(i,jk) に足さないで合力 hn に加えることにも注意が必要である。

更に end 行の前に

```
gapdum=gap
```

の 1 行を加える。

5.4 流動形態の違い

微弱に見えるが引力の効果は大きく、流動形態も全く異なる。特に傾斜面においても流動開始までに相当時間がかかる。これは水飴でくっついた粒子群がある程度傾けてから水飴が伸びるまでは殆んど動かず、その接合が切れる (あるいは伸びる) ことで動き始めることに対応している。また通常であれば表層に流れが集中するが全体がゆっくりと流れるといった記述がぴったりとする。流れは極端に遅くなることは容易に見てとれる。

6 注意事項

このノートでは DEM の基本プログラムとして斜面上の粒子運動をシミュレートしてみた。実際にはデータへの格納その他が必要であるがそれは省略した。

この程度の粒子数や設定であればリアルタイムでのシミュレーションでもそれほど問題にはならない。しかし現在の時間幅でも粒子速度が上がると粒子の重なりが見える場合があり精度に若干の問題を残す。実際にデータを取る場合には刻みをもう少し大きく取った方がよい。実際の時間刻みの上限を見積もる理論も必要になるかもしれない。

粒子の回転によって流動度が随分異なる。stick-slip 等では回転があるとどうしても一回目のスリップの後、安定に stick しない様である。回転の対症療法はあるようだが理解できない面も多く、解決すべき問題も多い。

このノートの作製で細かなバグの発見と、細部の理解が進んだ。当然、このアルゴリズムはソフトコアの MD にそのまま使える。また引力等を考慮すると更に様々な現象に応用できる。そういった応用から新たな物理学が見えてくる可能性がある。

参考文献

[1] 粉体工学会編， 粉体シミュレーション入門 第 3 章（産業図書 1998）修正等については http://www.iijnet.or.jp/SPTJ/SocPowTec/shusei.html.

[2] ここでの記述は田口善弘，物性研究 **61**, 1 (1993) とほぼ対応している．また同じ著者の，物理学会論文選集用に書かれた解説も参考にしている．

[3] H. Hertz, J. Math. (Crelle J) **92**, (1881).

[4] 例えば ランダウ＝リフシッツ，弾性理論（佐藤常三訳，東京図書 1972(原著 1965)) の第 1 章§9.

[5] R. D. Mindlin, J. Appl.Mech. Trans. ASME **16**, 259 (1949): R. D. Mindlin & H. Deresiewicz, J. Appl.Mech. Trans. ASME **20**, 327 (1953).

[6] P. A. Cundall & O. D. L. Strack, Geotechnique, **29**, 47 (1979).

[7] 田中敏嗣，石井俊哉，辻裕，日本機械学会論文集 **57**, 456 (1991).

[8] W.A. M. Morgado & I.Oppenheim, Physica A **246**, 547 (1997): see also S. Thomas and T. Pöschel,, Phys. Rev. E **57**, 650 (1998).

[9] D. G. Flom and A. M. Bueche, *J. Appl. Phys.* **30**, 1723 (1959)

[10] 松本啓和, 下坂厚子, 日高重助, 1997 年度粉体工学会春季研究発表会講演論文集, 39 (1997)

[11] S. Yuu, T. Abe, T. Saitoh & T. Umekage, Advanced Powder Technol. **6**, No.4, 259 (1995).

[12] 例えば坂口栄一郎, 川上昭太郎, 田村聡, 飛田布美子, 農業機械学会誌, **58** (4) 9 (1996)

[13] 例えば，阪口 秀 他,1993 年度秋期研究発表会講演要旨集 p110; 湯 晋一, 林昭久，脇昌裕，梅景俊彦，粉体工学会誌 **34**, 212 (1997).

[14] 泊野元彦，破壊のシミュレーションー拡張個別要素法で破壊を追うー（森北 1997）

[15] F. Gerl and A. Zippelius, Phys. Rev. E **49**, 2361 (1999).

[16] P. A. Thompson and G. S. Grest, Phys. Rev. Lett. **67**, 1751 (1991).