2. THE EQUATIONS OF MOTION

2.1 Introductory remarks

Every discrete element technique is based on determining the characteristic displacements of the simulated system. However, the different methods differ from each other in the chosen characteristic units of the analysis whose displacements are to be determined, in the exact meaning of the displacement-type unknowns they intend to determine, and in the numerical technique which is used in the calculations. This chapter introduces the three main types of the elements, introduces their corresponding displacement unknowns, and presents those equations which serve as the basis of the step-by-step simulation of the behaviour of the model.

In general, three basic types of elements are used in the different DEM codes:

(i) Perfectly rigid elements;
(ii) Elements made deformable with the help of their subdivision into simple finite elements (e.g. UDEC of Cundall, 1971);
(iii) Elements without subdivision, in which the stress and strain fields are approximated by simple functions (e.g. they are constant like in Shi, 1988, or they obey a linear law inside the element, e.g. MacLaughlin, 1997).

In general, there are two approaches to calculate the response of a collection of elements for some given loads. Quasi-static approaches search for the displacements which lead the system from the given state to an equilibrated state. Dynamic (time integration) methods, on the other hand, intend to simulate the motion of the elements during a series of small but finite time steps.

The usual (x, y, z) Descartes coordinate system will be applied throughout the chapter.

2.2 Perfectly rigid elements

Perfectly rigid elements have a wide variety of shapes in the different softwares: spheres, polyhedral elements, cylinders and the composition of these and similar simple shapes are widely applied in the practice. The elements do not change their shape and size, so that they are able only to translate and rotate.

In order to describe the behaviour of a collection of N rigid elements, a reference point should be chosen on each element. Theoretically the reference point can be chosen at any arbitrary location, even outside the element itself, but the calculations are significantly easier if the reference point coincides with the centre of gravity of the element. Hence in the practice the reference point is usually the same as the centre of gravity.

In the initial stage of the calculations the following data of the model (i.e. collection of N rigid bodies) have to be known:

– the exact shape, location and orientation of each element;
– the (concentrated or distributed) forces and moments transmitted in their contacts;
– the external forces which act on the elements, e.g. gravity or drag forces;

– in the case of time integration methods (which follow the motions during small time steps) the initial translational and rotational velocities of the elements also have to be known.

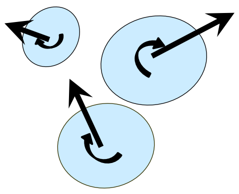

Figure 1.
Displacements of
Perfectly rigid elements

The displacement vector of the *p*-th element consists of the three components of the translation vector of the reference point, and the three rotation components about the reference point:

$$\mathbf{u}^p(t) = \begin{bmatrix} u_x^p(t) \\ u_y^p(t) \\ u_z^p(t) \\ \varphi_x^p(t) \\ \varphi_y^p(t) \\ \varphi_z^p(t) \end{bmatrix}$$

Collecting them into a single hypervector, the displacement vector of the whole system is received:

$$\mathbf{u}(t) = \begin{bmatrix} \mathbf{u}^1(t) \\ \mathbf{u}^2(t) \\ \vdots \\ \mathbf{u}^N(t) \end{bmatrix}$$

The increments of these displacements during a calculation cycle are considered to be small, and the accumulated increments during several calculation cycles give the total displacements of the system.

Let **v**(*t*) and **a**(*t*) denote the velocity and acceleration of the system:

$$\mathbf{v}(t) = \frac{d\mathbf{u}(t)}{dt}, \quad \mathbf{a}(t) = \frac{d^2\mathbf{u}(t)}{dt^2} \ .$$

The two blocks belonging to the *p* th element are:

$$\mathbf{v}^p(t)=\begin{bmatrix} v_x^p(t)\\ v_y^p(t)\\ v_z^p(t)\\ \omega_x^p(t)\\ \omega_y^p(t)\\ \omega_z^p(t)\end{bmatrix}=\begin{bmatrix}\frac{du_x^p(t)}{dt}\\ \frac{du_y^p(t)}{dt}\\ \frac{du_z^p(t)}{dt}\\ \frac{d\varphi_x^p(t)}{dt}\\ \frac{d\varphi_y^p(t)}{dt}\\ \frac{d\varphi_z^p(t)}{dt}\end{bmatrix} \quad ; \quad \mathbf{a}^p(t)=\begin{bmatrix} a_x^p(t)\\ a_y^p(t)\\ a_z^p(t)\\ \beta_x^p(t)\\ \beta_y^p(t)\\ \beta_z^p(t)\end{bmatrix}=\begin{bmatrix}\frac{d^2u_x^p(t)}{dt^2}\\ \frac{d^2u_y^p(t)}{dt^2}\\ \frac{d^2u_z^p(t)}{dt^2}\\ \frac{d^2\varphi_x^p(t)}{dt^2}\\ \frac{d^2\varphi_y^p(t)}{dt^2}\\ \frac{d^2\varphi_z^p(t)}{dt^2}\end{bmatrix}$$

The velocity and acceleration vectors are unnecessary for a quasi-static analysis, but indispensable for any step-by-step time integration method.

The Newtonian equations of motion for an element of general shape consist of six scalar equations:

$$\begin{gathered}
m^p a_x^p = f_x^p\\
m^p a_y^p = f_y^p\\
m^p a_z^p = f_z^p\\
I_{xx}^p\beta_x - I_{xy}^p\beta_y - I_{xz}^p\beta_z + \omega_y^p\left(\omega_z^p I_{zz}^p - \omega_x^p I_{zx}^p - \omega_y^p I_{zy}^p\right) - \omega_z^p\left(\omega_y^p I_{yy}^p - \omega_x^p I_{yx}^p - \omega_z^p I_{yz}^p\right) = m_x^p\\
I_{yy}^p\beta_y - I_{yx}^p\beta_x - I_{yz}^p\beta_z - \omega_x^p\left(\omega_z^p I_{zz}^p - \omega_x^p I_{zx}^p - \omega_y^p I_{zy}^p\right) + \omega_z^p\left(\omega_x^p I_{xx}^p - \omega_y^p I_{xy}^p - \omega_z^p I_{xz}^p\right) = m_y^p\\
I_{zz}^p\beta_z - I_{zx}^p\beta_x - I_{zy}^p\beta_y + \omega_x^p\left(\omega_y^p I_{yy}^p - \omega_x^p I_{yx}^p - \omega_z^p I_{yz}^p\right) - \omega_y^p\left(\omega_x^p I_{xx}^p - \omega_y^p I_{xy}^p - \omega_z^p I_{xz}^p\right) = m_z^p
\end{gathered}$$

On the right side, the f_x^p, f_y^p, f_z^p forces and m_x^p, m_y^p, m_z^p moments are produced by reducing all the external and contact forces acting on the element to the reference point. These forces may vary in time, and may depend on the location and velocity of the element (e.g. reaction force expressed by an elastic support, the contact force between two touching elements, or a velocity-dependent damping or drag force). Hence in general the resulting $f_x^p, f_y^p, f_z^p, m_x^p, m_y^p, m_z^p$ reduced forces are also depending on t, $\mathbf{u}(t)$ and $\mathbf{v}(t)$.

On the left side m^p denotes the mass of the p-th element, and the quantities I_{xx}, I_{xy}, I_{xz}, etc. are the rotational inertia written for the axes going through the reference point and being parallel to the global x, y and z axes. Remember that these inertia can be determined from the following data:

μ : the (not necessarily constant) density of the element
V^p : the volume of the element
(x^p, y^p, z^p): the centre of gravity of the element

in the following way:

$$I^p_{xy} = \int_{V^p} (x - x^p)\cdot(y - y^p)\cdot \mu(x,y,z)\cdot dV \quad ;$$

$$I^p_{zy} = \int_{V^p} (z - z^p)\cdot(y - y^p)\cdot \mu(x,y,z)\cdot dV \quad \text{etc.}$$

(Note that if the element is spherically symmetrical about its centre of gravity, then $I^p_{xx} = I^p_{yy} = I^p_{zz} := I^p$, and the rest of the inertia are zero. In this case the equations of motion is less complicated:

$$m^p a^p_x = f^p_x$$
$$m^p a^p_y = f^p_y$$
$$m^p a^p_z = f^p_z$$
$$I^p \beta_x = m^p_x$$
$$I^p \beta_y = m^p_y$$
$$I^p \beta_z = m^p_z \text{ .)}$$

Now rearrange the equations of motion by collecting to the right side all the terms which depend on the rotational velocity. Without showing the details, the equations are written now in the following form:

$$\mathbf{M}^p(t)\mathbf{a}^p(t) = \mathbf{f}^p(t, \mathbf{u}(t), \mathbf{v}(t))$$

in which the vector $\mathbf{f}^p$ depends not only on the location and velocity of element *p*, but it may depend also on the location and velocity of those other elements being in contact with *p*.

The matrix $\mathbf{M}^p$ contains the mass and inertia of *p* , and this is also time-dependent in the general case, since the rotation of the element modifies the inertia. (The spherically symmetrical elements are exceptional in this respect: their rotations do not modify the spatial distribution of their material.)

The same can be written to all the elements in the model, and the equations of motion of the whole system is received:

$$\mathbf{M}(t)\mathbf{a}(t) = \mathbf{f}(t, \mathbf{u}(t), \mathbf{v}(t)) \text{ .}$$

where $\mathbf{M}(t)$ is the block-diagonal matrix consisting of the matrices $\mathbf{M}^p(t)$.

Rigid elements are, without doubt, the most widely used element types. The BALL-type codes, like PFC or EDEM, the Contact Dynamic models etc. are all based on rigid elements.

2.3 Elements made deformable by subdivision into finite elements

One possibility to make the discrete elements deformable is to divide them into subdomains whose simply approximated strain and stress fields are treated according to the usual methods of FEM. The simplest possibility – suggested still in the 1960ies by Goodman et al, 1968; and then applied for instance in UDEC – is to subdivide the element into uniform-strain simplexes (triangles in 2D and tetrahedral in 3D), as shown in Figure 2. More sophisticated internal finite element approaches can also be found in the literature (e.g. Ghaboussi, 1988; Barbosa and Ghaboussi, 1992; Owen et al., 1999; Munjiza, Owen and Bicanic, 1995; Munjiza, 2004), but we shall restrict the introduction here to the simplest version, i.e. to the application of uniform-strain simplexes.

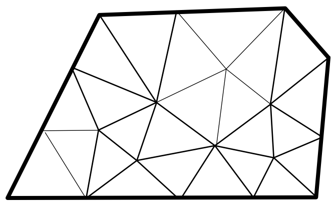

Figure 2.
Polyhedral element
subdivided into simplexes

Inside every simplex, a linear translation field can uniquely be defined as the linear interpolation of the translations of the nodes. The gradient of this translation field is, hence, also unique, and constant inside the simplex. The symmetric part of the translation gradient (i.e. the strain) characterizes the deformation of the simplex; and the skew-symmetric part describes the rigid-body-like rotation of the simplex.

Consequently, the kinematics of such a discrete element is described by the translations of the nodes of its simplexes, and through the strains, the stresses can be determined with the help of the constitutive relations of the material of the element.

The displacement vector of the whole system is:

$$\mathbf{u}(t)=\begin{bmatrix}\mathbf{u}^1(t)\\ \mathbf{u}^2(t)\\ \vdots\\ \mathbf{u}^N(t)\end{bmatrix}$$

where N denotes the total number of all nodes in all elements altogether, and the block belonging to the p-th node consists of the three scalar components of the translation vector of this node:

$$\mathbf{u}^p(t)=\begin{bmatrix}u_x^p(t)\\ u_y^p(t)\\ u_z^p(t)\end{bmatrix} \quad .$$

The equations of motion for the p-th node can be written in the

$$m^p(t)\mathbf{a}^p(t)=\mathbf{f}^p(t,\mathbf{u}(t),\mathbf{v}(t))\,.$$

Here m^p denotes the mass assigned to node p. This mass is defined as the mass of the Voronoi-cell belonging to node p. (Figure 3. illustrates that the Voronoi-cell of a node is the set of those points of the element which are closer to the given node than to any other node.) Figure 3. shows the Voronoi-cells of a node in the interior of the element, and a node on the boundary (blue area).

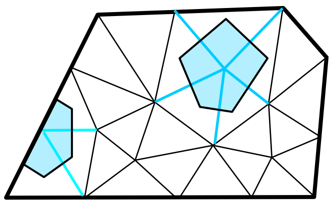

Figure 3.
Voronoi-cells of the nodes of an element subdivided into simplexes

The volume of the Voronoi-cell belonging to node p is denoted by V^p; with the help of the given material density, the m^p mass assigned to node p can easily be calculated.

The force vector $\mathbf{f}^p(t)$ on the right side of the equation denotes the resultant of all those forces acting on the cell V^p , reduced to the node p. It consists only of three force components, and the fact that a moment is also produced about the node is neglected. (Note that with increasing density of subdivision, i.e. with increasing number of simplexes within the element, the magnitude of the neglected moment decreases.)

The force $\mathbf{f}^p$ is due to three different effects. First, there can be external forces acting on the domain assigned to node p: e.g. gravity, or drag force if the node is on the boundary of the element. Second, contact forces are transmitted by the neighbouring elements. Third, because of the stress field inside the element, the neighbouring Voronoi-cells express distributed internal forces on the boundary of cell p. The force $\mathbf{f}^p$ is the resultant of all these forces, assigned to node p, by neglecting the fact that this resultant does not necessarily goes through the node itself. (The neglected moment decreases with increasing the density of subdivision into simplexes.)

Note that the three components of $\mathbf{f}^p$ belonging to the node correspond to the considered degrees of freedom of the node: translations are considered only, without any rotations.

The three scalar equations belonging to the individual nodes can be summarized into a single system of equations. Introduce the following notation:

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}^1 & & & \\ & \mathbf{M}^2 & & \\ & & \ddots & \\ & & & \mathbf{M}^N \end{bmatrix}$$

where

$$\mathbf{M}^p = \begin{bmatrix} m^p & & \\ & m^p & \\ & & m^p \end{bmatrix} \ .$$

Using this, the complete system of the equations of motion has the following form:

$$\mathbf{M}\cdot\mathbf{a}(t) = \mathbf{f}(t,\mathbf{u}(t),\mathbf{v}(t)) \ .$$

As an example, the code UDEC and its 3D version 3DEC works this way. More complicated internal finite element meshes are also possible, but we shall not deal with them in detail.

2.4 Deformable elements without subdivision

In the DDA models (Shi, 1988; 2001) the elements are not subdivided into subdomains; instead, their deformability is simulated with the help of a uniform (or perhaps higher-degree) strain field. For simplicity, here we shall focus on uniform-strain fields only. In this case the deformation characteristics also belong to the unknowns in addition to the displacements of the element, which means that in addition to the 6 components of the displacements of a rigid element, 6 more degrees of freedom are considered, describing the uniform deformations of the element. The generalized unknown displacement vector of the *p*-th element is:

$$\mathbf{u}^p = \begin{bmatrix} u_x^p \\ u_y^p \\ u_z^p \\ \varphi_x^p \\ \varphi_y^p \\ \varphi_z^p \\ \varepsilon_x^p \\ \varepsilon_y^p \\ \varepsilon_z^p \\ \gamma_{yz}^p \\ \gamma_{zx}^p \\ \gamma_{xy}^p \end{bmatrix}$$

and corresponding to this, the generalized force vector $\mathbf{f}^p$ also consists of 12 scalars altogether, in which the second 6 scalars contain the characteristic components of the uniform stress tensor of the element:

$$\mathbf{f}^p = \begin{bmatrix} f_x^p \\ f_y^p \\ f_z^p \\ m_x^p \\ m_y^p \\ m_z^p \\ V^p\sigma_x^p \\ V^p\sigma_y^p \\ V^p\sigma_z^p \\ V^p\tau_{yz}^p \\ V^p\tau_{zx}^p \\ V^p\tau_{xy}^p \end{bmatrix}$$

The equations of motion of the element consists now of 12 scalar equations in which the second 6 equations express relations between the stresses and the second derivatives of the strains. The equations of motion again have the form

$$\mathbf{M}(t)\cdot\mathbf{a}(t) = \mathbf{f}(t,\mathbf{u}(t),\mathbf{v}(t)),$$

but here the matrix of inertia, i.e. **M**, is much more complex than in the case of rigid elements. We shall not introduce here the very lengthy exact details.

2.5 Time stepping versus quasi-static models

It was seen above that the equations of motion have the same form for all types of elements applied in DEM modelling. The different ***time-stepping*** methods approximate the motion of the system, starting from a known initial state, through a series of small but finite time intervals, in such a way that the equations

$$\mathbf{M}(t)\cdot\mathbf{a}(t) = \mathbf{f}(t,\mathbf{u}(t),\mathbf{v}(t))$$

would (at least approximately) be satisfied at the discrete time points t_1, t_2, …, t_i , … , i.e. the endpoints of the time intervals.

Some of the DEM codes apply ***explicit*** time integration for this, which means that when considering a time interval, those **u** and **v** values (generalized displacements and velocities) belonging to the endpoint t_{i+1} are determined in such a way that the equations of motion are compiled at time point t_i and the values at t_{i+1} are predicted from the approximated **u** and **v** values belonging to t_i. The explicit techniques do not check whether the equations of motion are satisfied at the endpoint of the actual interval, i.e. at t_{i+1}; the predicted values are accepted without checking the equations of motion, and then they serve as the starting values for the forthcoming time interval. Figure 4a. illustrates the explicit methods.

Other DEM codes use ***implicit*** time integration techniques. In this case the **u** and **v** values belonging to the endpoint t_{i+1} are calculated in such a way that the equations of motion would be satisfied at the endpoint of the time interval. This is done with the help of a gradually improving iteration scheme: the approximated values of **u** and **v** at t_{i+1} are checked and modified again and again, until a sufficiently exact match is reached; and these values are then used as the starting data for the next timestep. Figure 4b. shows this scheme.

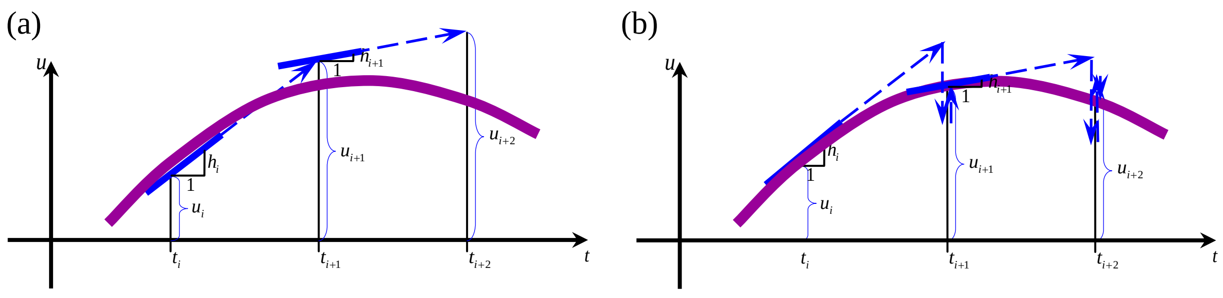

Figure 4.
The explicit (a) and the implicit (b) time integration techniques

The explicit methods are, in general, much easier to implement, and require significantly less computer time for the analysis of a timestep. On the other hand, the predicted results can quickly deviate from the exact solution, and numerical instabilities can occur. To avoid this, the applied length of the timesteps should be limited, and several numerical "tricks" are used to ensure the stability of the calculations. The implicit methods are significantly more reliable from numerical point of view, and longer timesteps can be applied; on the other hand, they are computationally more expensive. Most commercial codes use explicit techniques due to their simplicity.

Quasi-static methods are based on a completely different way of thinking. They assume that the initial state of the system is equilibrated, and as a response to the changing of the external loads, the system reaches a new equilibrated state, corresponding already to the modified external forces, through "zero velocities" (i.e. inertial effects are excluded from the analysis). These methods search for the $\Delta\mathbf{u}$ displacements taking the system from the initial to the new equilibrium state where

$$\mathbf{0} = \mathbf{f}(\Delta\mathbf{u}) \ .$$

The existing quasi-static codes all apply perfectly rigid elements, so **f** represents the reduced forces acting on the reference points of the rigid elements, and the displacements are the same as those in Section 2.2.

The calculation of $\Delta\mathbf{u}$ provides an approximation which should be checked: the reduced force vector has to be compiled and if its components are sufficiently small, the calculated new position can be accepted as an equilibrium state. On the other hand, if the equilibrium equations are not satisfied sufficiently exactly, the equilibrium error is expressed with a new reduced force vector being compiled in the new position, and the calculation is repeated again and again, until the equilibrium error becomes small enough. The accumulated displacements $\Delta\mathbf{u}_1+\Delta\mathbf{u}_2+\ldots+\Delta\mathbf{u}_i + \ldots$ take the system from the original state to the final, sufficiently equilibrated state.

Quasi-static methods are not really widespread: they can be found only in different research codes. However, from theoretical point of view they are very important.

There are several numerical techniques applied for the calculations of the above problems. The forthcoming lecture will summarize the most important techniques.

Questions

2.1. Explain the exact meaning of the quantities in the equations of motion in the case of perfectly rigid elements!
2.2. Explain the exact meaning of the quantities in the equations of motion in the case of elements subdivided into uniform-strain simplexes!
2.3. What is the difference between time-stepping and quasi-static methods?
2.4. What is the difference between explicit and implicit methods?