3. OVERVIEW OF NUMERICAL METHODS

3.1 Introductory remarks

This chapter summarizes those numerical techniques whose knowledge is indispensable for the understanding of the different discrete element methods:

- the ***Newton-Raphson-method***, a powerful technique to numerically solve nonlinear equations;
- the ***Gauss-Seidel-method,*** a relaxation technique which produces gradually improving approximations for the solution of linear systems of equations;
- the ***Euler-method*** or ***first-order Runge-Kutta-method***, to numerically solve initial value problems of first-order differential equations;
- the ***method of central differences***, which can be considered as an improved version of the Euler-method;
- ***Newmark's β-method***, for the time integration of second-order differential equations like the equations of motion.

We shall not deal in detail with the deeper problems like convergence analysis, matrix invertibility. The interested reader is advised to consult the wide literature of numerical mathematics; an excellent starting point is Belytschko et al (2000).

3.2 The Newton-Raphson method

The aim of this method is to determine a vector $\hat{\mathbf{u}}$ which satisfies the given nonlinear equations

$$\begin{aligned} f_1(u_1,u_2,...,u_n) &= 0 \\ f_2(u_1,u_2,...,u_n) &= 0 \\ &\vdots \\ f_n(u_1,u_2,...,u_n) &= 0 \end{aligned}$$

or shortly

$$\mathbf{f}(\mathbf{u}) = \mathbf{0} \ ,$$

where $\mathbf{f}(\mathbf{u})$ is a continuously differentiable function of u. It is assumed that we have an initial estimate $\mathbf{u}_0$ for $\hat{\mathbf{u}}$, and also assumed that for any arbitrary $\mathbf{u}$, we are able to determine $\mathbf{f}(\mathbf{u})$ as well as its Jacobian matrix, $\mathbf{K}(\mathbf{u})$:

$$\mathbf{K}(\mathbf{u}) = \frac{d\mathbf{f}(\mathbf{u})}{d\mathbf{u}}$$

or, in a more detailed form:

$$K_{pq} = \frac{df_p(u_1,u_2,...,u_n)}{du_q} \ .$$

Note that the explicit form of $\mathbf{f}(\mathbf{u})$ and $\mathbf{K}(\mathbf{u})$ is not necessary to know; but we must be able to calculate them for any arbitrary $\mathbf{u}$.

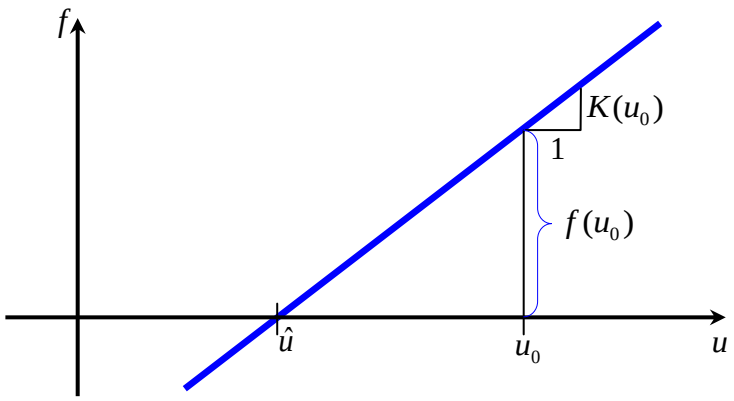

Figure 1.
The Newton-Raphson-method for a linear equation

Imagine first that **f** is a linear function of **u**. In this case **K** is constant (the same for any **u**). Assume, in addition, that **K** is not singular, i.e. it is invertable. In this case the equations are easy to solve in a single step:

$$\hat{\mathbf{u}} = \mathbf{u}_0 - \mathbf{K}(\mathbf{u}_0)^{-1} \cdot \mathbf{f}(\mathbf{u}_0)$$

(Figure 1. illustrates this for the very special case $n = 1$). However, for a non-linear **f** the above calculation leads to a next approximation:

$$\mathbf{u}_1 = \mathbf{u}_0 - \mathbf{K}(\mathbf{u}_0)^{-1} \cdot \mathbf{f}(\mathbf{u}_0)$$

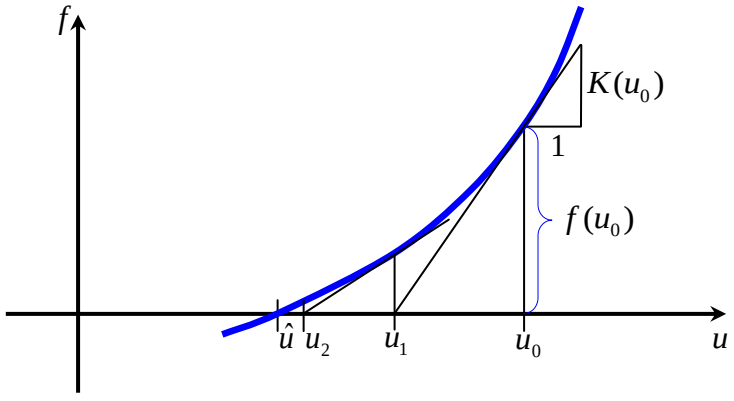

Figure 2.
The Newton-Raphson-method for a nonlinear equation

which can be used to produce one more approximation of the root:

$$\mathbf{u}_{i+1} = \mathbf{u}_i - \mathbf{K}(\mathbf{u}_i)^{-1} \cdot \mathbf{f}(\mathbf{u}_i) \ .$$

(Instead of inverting **K**, an alternative is to solve the linear equations

$$\mathbf{K}(\mathbf{u}_i)\left(\mathbf{u}_{i+1}-\mathbf{u}_i\right)=-\mathbf{f}(\mathbf{u}_i)\ .$$

for $\mathbf{u}_{i+1}$.)

This has to be repeated until the "magnitude" of $\mathbf{f}(\mathbf{u}_k)$ (e.g. the Euclidean norm, $|\mathbf{f}|=\sqrt{\mathbf{f}^T\cdot\mathbf{f}}$) becomes smaller than a pre-defined error limit. In addition to the termination criterion $|\mathbf{f}|<$ [a prescribed value], the value of $|\mathbf{u}_{i+1}-\mathbf{u}_i|$ is also often tracked, which is particularly helpful if the function $\mathbf{f}(\mathbf{u})$ is rather "flat", i.e. $\mathbf{K}$ is nearly singular around the root.

Usually the method quickly converges to a root of the equation, assuming that the initial "guess" $\mathbf{u}_0$ was close enough. However, there are many situations when the method fails or badly converges, for instance:

- Bad starting point: e.g. $\mathbf{K}$ is singular or nearly singular at the starting point; or the starting point is not in an interval from where the method converges. The starting point may also enter an infinite cycle, jumping back-and-forth between two (or perhaps more) points without a convergence.
- The derivative $\mathbf{K}$ does not exist at the root, or is discontinuous near the root.

Among the different discrete element techniques this method is applied for instance in the DDA methods and in the Bagi-Bojtar-method.

3.3 Relaxation: The Gauss-Seidel method

Quasi-static DEM models are based on the solution of a system of linear equations. As it will be explained later, they apply the well-known ***displacement method*** of structural analysis, so the equilibrium equations corresponding to each degree of freedom give the system of equations to solve, whose unknowns are the characteristic displacements of the model. In these systems the number of equations equals the number of unknowns. The problem to solve can be formulated as follows:

Determine the unknown vector **u:**

$$\mathbf{u}=\begin{bmatrix}u_1\\u_2\\\vdots\\u_n\end{bmatrix}$$

so that it would satisfy the equations

$$\mathbf{K}\cdot\mathbf{u}+\mathbf{f}=\mathbf{0}$$

where $\mathbf{K}$ and $\mathbf{f}$ consist of given constants:

$$\mathbf{K} = \begin{bmatrix} k_{11} & k_{12} & k_{1n} \\ k_{21} & k_{22} & k_{2n} \\ & & \\ k_{n1} & k_{n2} & k_{nn} \end{bmatrix} \quad \text{and} \quad \mathbf{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix} .$$

There exist several methods to solve large systems of linear equations. ***Direct solvers*** (e.g. Gauss-Jordan elimination, LU-factorization) aim to produce the exact (up to numerical rounding errors) solution, e.g. by inverting the coefficient matrix **K**, or by other techniques: they obtain the solution in a finite number of calculation steps. ***Relaxation methods***, on the other hand, produce a sequence of approximations of the solution, hopefully getting closer and closer to the exact solution.

The Gauss-Seidel-method will be particularly important for us. It works in the following way:

Starting from an initial $\mathbf{u}_0$ estimation, the method prepares a $\mathbf{u}_1$, $\mathbf{u}_1$, … $\mathbf{u}_i$, $\mathbf{u}_{i+1}$, … series of approximations, converging to the exact solution. Denote the *j*-th scalar component of the solution by u_j, and its *i*-th estimation by $u_{j,i}$. From the i-th estimation, $\mathbf{u}_i$, the *i*+1-th estimation, $\mathbf{u}_{i+1}$, is calculated as follows.

– Calculate the $\mathbf{r}_i$ "residual vector" belonging to the *i*-th approximation:

$$\mathbf{r}_i = \mathbf{K} \cdot \mathbf{u}_i + \mathbf{f} \ .$$

If $\mathbf{u}_i$ were equal to the exact solution, then the components of $\mathbf{r}_i$ were all zeros, and the calculations should be terminated. However, if this is not the case, then the scalar components of $\mathbf{r}_i$ are not all zeros. A general, say *j*-th component is:

$$r_{j,i} = k_{j1}u_{1,i} + k_{j2}u_{2,i} + ...k_{jn}u_{n,i} + f_j$$

– Find that scalar in $\mathbf{r}_i$ which has the largest absolute value, say this is the *p*-th component of $\mathbf{r}_i$:

$$r_{p,i} = \max_{(j)} \left| r_{j,i} \right|$$

(We can also say that the *p*-th equation is satisfied "worst" by the *i*-th approximation of the solution.)

– Calculate now what value should be stand at the *p*-th position of $\mathbf{u}_i$ so that this "worst" value in the residual would become zero. In other words, determine $u_{p,i+1}$ to replace $u_{p,i}$ so that

$$r_{p,i+1} = \sum_{h=1}^{p-1} k_{ph}u_{h,i} + k_{pp}u_{p,i+1} + \sum_{h=p+1}^{n} k_{ph}u_{h,i} + f_p = 0 \ .$$

After some rearrangement, the solution for $u_{p,i+1}$ is:

$$u_{p,i+1} := -\frac{1}{k_{pp}} \left[\sum_{h=1}^{p-1} k_{ph}u_{h,i} + \sum_{h=p+1}^{n} k_{ph}u_{h,i} + f_p \right] .$$

– The rest of the components of $\mathbf{u}_{i+1}$ remain equal to the corresponding components of $\mathbf{u}_i$:

$$u_{j,i+1} := u_{j,i} \quad \text{for all} \quad j \neq p \ .$$

Then let $i := i + 1$, and a next approximation can be prepared in the same way.

Further and further iteration steps have to be done, until the components of the residual vector **r** become sufficiently small (e.g. the Euclidean norm of **r** can be the used for a terminating criterion).

Method of Cross, widely applied in the 20ieth century in manual structural analysis, is based on this relaxation technique. Method of Cross is for frames whose nodes have rotational degrees of freedom. The moment balance equations of the nodes are a system of linear equations, and the unknowns are the rotations of the nodes which lead to such a position where the moment balance is satisfied for all nodes. In any step, the node with the largest equilibrium error is selected, and while keeping all others fixed, the considered node is allowed to rotate into the equilibrium position.

The method of Kishino, a quasi-static DEM technique, applies this type of relaxation.

3.4 Solution methods for initial value problems

3.4.1 Introductory remarks

Time-stepping DEM algorithms intend to determine (or, at least, approximate with sufficient accuracy) the time-dependent displacements $\mathbf{u}(t)$ and velocities

$$\mathbf{v}(t) = \frac{d\mathbf{u}(t)}{dt}$$

of the model at the discrete "time points" $t_1, t_2, \ldots, t_i, t_{i+1}, \ldots$, starting from a known $\mathbf{u}(t_0)$ and $\mathbf{v}(t_0)$ which belong to the initial time t_0. The time-stepping algorithms calculate a series of

$$\mathbf{u}(t_1), \mathbf{v}(t_1); \quad \mathbf{u}(t_2), \mathbf{v}(t_2); \quad \ldots \mathbf{u}(t_i), \mathbf{v}(t_i), \quad \mathbf{u}(t_{i+1}), \mathbf{v}(t_{i+1}); \quad \ldots$$

approximations for which the equations of motion, i.e. the equations

$$\mathbf{M} \cdot \frac{d^2\mathbf{u}(t)}{dt^2} = f(t, \mathbf{u}(t), \mathbf{v}(t))$$

are sufficiently accurately satisfied at the $t_1, t_2, \ldots, t_i, t_{i+1}, \ldots$ time instants.

There are a vast number of numerical techniques to numerically solve initial value problems of differential equations. Part of these techniques are ***explicit***: as explained already in Section 2., it 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 ***implicit*** time integration techniques are more reliable. In these methods the **u** and **v** values belonging to the endpoint t_{i+1} are calculated in such a way that the equations of motion should be satisfied at the endpoint of the time interval, which 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.

There are three methods which will be particularly important in discrete element modelling:

(i) the first order Runge-Kutta-method (Euler-method);
(ii) the method of central differences;
(iii) Newmark's β-method.

The Euler-method and the method of central differences is suitable for the calculation of ***first-order differential equations*** (i.e. when the differential equations contain only the first derivative of the unknown function). The equations of motion are, on the other hand, second-order equations. Fortunately, general higher-order differential equations can easily be transformed into first-order equations in the following way:

→ Consider the *k*-th order differential equation:

$$\frac{d^k \mathbf{u}(t)}{dt} = \mathbf{f}\left(t, \mathbf{u}(t), \frac{d\mathbf{u}(t)}{dt}, \frac{d^2\mathbf{u}(t)}{dt^2}, \ldots \frac{d^{k-1}\mathbf{u}(t)}{dt^{k-1}}\right) \quad .$$

→ Introduce the following notations:

$$\mathbf{y}_1(t) = \mathbf{u}(t)$$
$$\mathbf{y}_2(t) = \frac{d\mathbf{u}(t)}{dt}$$
$$\mathbf{y}_3(t) = \frac{d^2\mathbf{u}(t)}{dt^2}$$
$$\vdots$$
$$\mathbf{y}_k(t) = \frac{d^{k-1}\mathbf{u}(t)}{dt^{k-1}}$$

and using them, the original differential equations can be written in the form:

$$\frac{d\mathbf{y}_{k+1}(t)}{dt} = \mathbf{f}(t, \mathbf{y}_1(t), \mathbf{y}_2(t), \ldots \mathbf{y}_k(t)) \quad .$$

→ The relations between the functions $\mathbf{y}_1(t)$, $\mathbf{y}_1(t)$, …, $\mathbf{y}_k(t)$ are given by the (k-1) differential equations

(1.) $\dfrac{d\mathbf{y}_1(t)}{dt} = \mathbf{y}_2(t)$

(2.) $\dfrac{d\mathbf{y}_2(t)}{dt} = \mathbf{y}_3(t)$

$\vdots$

(k-1.) $\dfrac{d\mathbf{y}_{k-1}(t)}{dt} = \mathbf{y}_k(t)$

Let the *k*-th equation be the original differential equation by using the new notations, as seen above:

(k) $\dfrac{d\mathbf{y}_k(t)}{dt} = \mathbf{f}(t, \mathbf{y}_1(t), \mathbf{y}_2(t), \ldots \mathbf{y}_{k-1}(t))$

→ Summarize these *k* equations into the following system:

$$\frac{d\mathbf{y}(t)}{dt} = \hat{\mathbf{f}}(t, \mathbf{y}(t))$$

where the notations are:

$$\mathbf{y}(t)=\begin{bmatrix}\mathbf{y}_1(t)\\ \mathbf{y}_2(t)\\ \vdots\\ \mathbf{y}_{k-1}(t)\\ \mathbf{y}_k(t)\end{bmatrix} \quad ; \qquad \hat{\mathbf{f}}(t,\mathbf{y}(t))=\begin{bmatrix}\mathbf{y}_2(t)\\ \mathbf{y}_3(t)\\ \vdots\\ \mathbf{y}_k(t)\\ \mathbf{f}(t,\mathbf{y}(t))\end{bmatrix} \quad .$$

A first-order (but of course much larger) system of differential equations is received this way.

The equations of motion are of second order, so the application of the above transformation is straightforward. Introduce the following notations:

$$\mathbf{y}(t):=\begin{bmatrix}\mathbf{u}(t)\\ \mathbf{v}(t)\end{bmatrix} \quad ; \qquad \bar{\mathbf{f}}(t,\mathbf{u}(t),\mathbf{v}(t)):=\mathbf{M}^{-1}\cdot\mathbf{f}(t,\mathbf{u}(t),\mathbf{v}(t)) \quad ; \qquad \hat{\mathbf{f}}(t,\mathbf{u}(t),\mathbf{v}(t)):=\begin{bmatrix}\mathbf{v}(t)\\ \bar{\mathbf{f}}(t,\mathbf{u}(t),\mathbf{v}(t))\end{bmatrix}$$

or shortly:

$$\frac{d\mathbf{y}(t)}{dt}=\hat{\mathbf{f}}(t,\mathbf{y}(t)) \quad .$$

The solution of such a differential equation means to find the function $\mathbf{y}(t)$ if its first derivative is known, and the initial value, i.e. the $\mathbf{y}(t_0)$ which belongs to the given $t_0$time instant is specified. (Hence the name "initial vaule problem".)

3.4.2 First order Runge-Kutta-method (Euler-method)

Let the initial value of the function $\mathbf{y}(t)$ be $\mathbf{y}_0$:

$$\mathbf{y}(t_0)=\mathbf{y}_0 \ ,$$

and the function $\mathbf{y}(t)$ satisfies the following differential equation in which $\hat{\mathbf{f}}(t,\mathbf{y}(t))$ is given, not necessarily explicitly, but at least in the sense that it can be calculated for any t and $\mathbf{y}(t)$:

$$\frac{d\mathbf{y}(t)}{dt}=\hat{\mathbf{f}}(t,\mathbf{y}(t)) \ .$$

Our aim is to determine those $\mathbf{y}_1$, $\mathbf{y}_2$, …, $\mathbf{y}_i$, $\mathbf{y}_{i+1}$, … values which belong to the time instants t_1, t_2, …, t_i, t_{i+1}, … . (The increasing indices – unlike in the previous cases where hey indicated better and better approximations of the solution – now denote that time proceeds.)

According to the Euler-method, from value $\mathbf{y}_i$ which belongs to t_i the value $\mathbf{y}_{i+1}$ belonging to t_{i+1} is calculated in the way illustrated in Figure 3.:

→ Calculate $\hat{\mathbf{f}}$ at (t_i, $\mathbf{y}_i$) (it means to approximate the first derivative of the unknown function $\mathbf{y}(t)$ at t_i):

$$\mathbf{h}_i=\hat{\mathbf{f}}(t_i,\mathbf{y}_i) \quad .$$

→ Assuming that this first derivative remains constant on the (t_i, t_{i+1}) interval, the approximation of $\mathbf{y}_{i+1}$ at t_{i+1} (denoting the length of the interval by Δt, so $t_{i+1}=t_i+\Delta t$):

$$\mathbf{y}_{i+1}:=\mathbf{y}_i+\Delta t\cdot\mathbf{h}_i$$

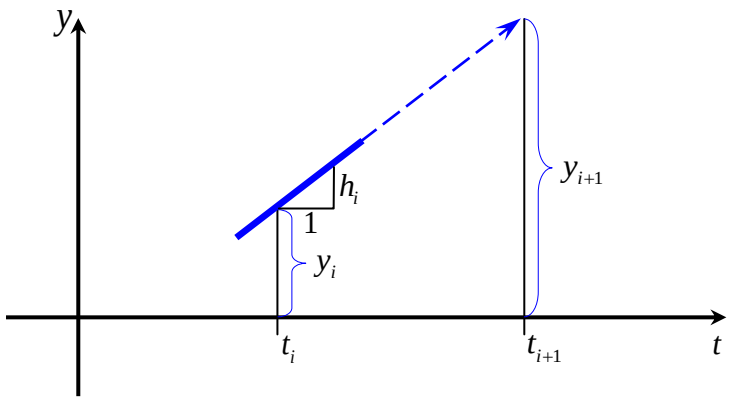

Figure 3.
The main step of the Euler-method

Then the analysis of the next interval follows.

Figure 4. illustrates that since at the ***endpoint*** of the interval it is not checked whether the equations of motion are satisfied within the necessary accuracy, the errors of the consecutive approximations accumulate, and the estimated $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_i, \mathbf{y}_{i+1}, \ldots$ values may deviate from the exact $\mathbf{y}(t)$ more and more as t increases.

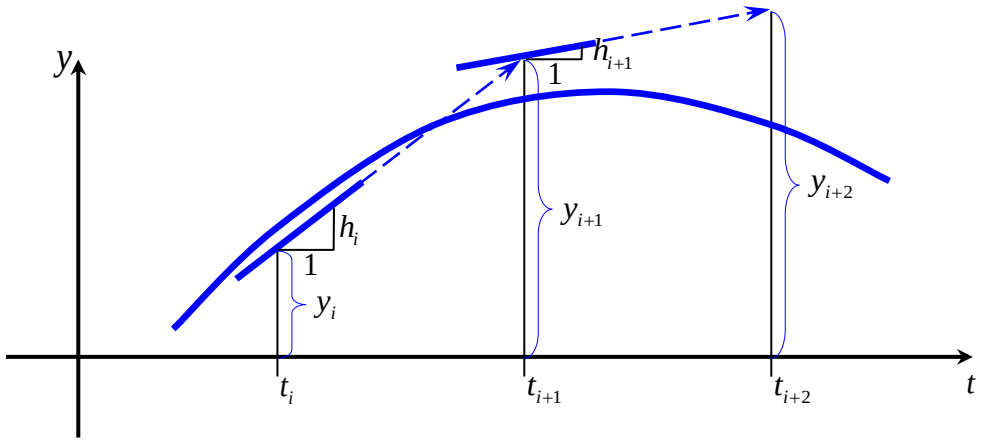

Figure 4.
Euler-method: Increasing deviations from the exact solution

The Euler-method can simply be applied to the DEM equations of motion. Introduce the notation

$$\begin{bmatrix} \dfrac{d\mathbf{u}(t)}{dt} \\ \dfrac{d\mathbf{v}(t)}{dt} \end{bmatrix} = \begin{bmatrix} \mathbf{v}(t) \\ \bar{\mathbf{f}}(t, \mathbf{u}(t), \mathbf{v}(t)) \end{bmatrix}$$

(here the $\bar{\mathbf{f}}(t, \mathbf{u}(t), \mathbf{v}(t))$ function is "known"), and the initial values belonging to t_0 are the given $\mathbf{u}_0$ and $\mathbf{v}_0$:

$$\begin{bmatrix} \mathbf{u}(t_0) \\ \mathbf{v}(t_0) \end{bmatrix} = \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{v}_0 \end{bmatrix}$$

From the already known ($\mathbf{u}_i$, $\mathbf{v}_i$) approximations which belong to t_i , those ($\mathbf{u}_{i+1}$, $\mathbf{v}_{i+1}$) belonging to a t_{i+1} can be calculated by using the derivative, $\mathbf{h}_i$, belonging to t_i:

$$\mathbf{h}_i = \begin{bmatrix} \mathbf{v}_i \\ \overline{\mathbf{f}}(t_i, \mathbf{u}_i, \mathbf{v}_i) \end{bmatrix}$$

which yields

$$\begin{bmatrix} \mathbf{u}_{i+1} \\ \mathbf{v}_{i+1} \end{bmatrix} = \begin{bmatrix} \mathbf{u}_i \\ \mathbf{v}_i \end{bmatrix} + \Delta t \cdot \mathbf{h}_i = \begin{bmatrix} \mathbf{u}_i \\ \mathbf{v}_i \end{bmatrix} + \Delta t \cdot \begin{bmatrix} \mathbf{v}_i \\ \overline{\mathbf{f}}(t_i, \mathbf{u}_i, \mathbf{v}_i) \end{bmatrix} \quad .$$

The main disadvantage of the Euler-method is its explicit nature (the accumulating errors of the approximations), which can be avoided by using the ***implicit*** version of the method. Its main idea is formulated as

$$\mathbf{y}_{i+1} := \mathbf{y}_i + \Delta t \cdot \mathbf{h}_{i+1} \quad ,$$

where

$$\mathbf{h}_{i+1} = \hat{\mathbf{f}}(t_{i+1}, \mathbf{y}_{i+1}) \quad .$$

Since $\mathbf{y}_{i+1}$ is unknown, $\hat{\mathbf{f}}(t_{i+1}, \mathbf{y}_{i+1})$ cannot be calculated. However, by using an iterative scheme, better and better approximations can be produced for $\mathbf{y}_{i+1}$, and the iteration is stopped when the considered measure of the error of the equations becomes sufficiently small. Using the notations applied in the equations of motion of DEM, the fundamental step of the implicit Euler-method is

$$\begin{bmatrix} \mathbf{u}_{i+1} \\ \mathbf{v}_{i+1} \end{bmatrix} = \begin{bmatrix} \mathbf{u}_i \\ \mathbf{v}_i \end{bmatrix} + \Delta t \cdot \begin{bmatrix} \mathbf{v}_{i+1} \\ \overline{\mathbf{f}}(t_i, \mathbf{u}_{i+1}, \mathbf{v}_{i+1}) \end{bmatrix} \quad .$$

This method is applied in the Contact Dynamics methods of discrete element modelling.

3.4.3 The method of central differences

This method can be considered as an improved version of the explicit Euler-method. For the sake of simplicity, without dealing with the general notations with ($\mathbf{y}(t)$, $\mathbf{h}(t)$), the application for the equations of motion will be introduced here only.

Based on the $\mathbf{u}_i$ approximation of the function $\mathbf{u}(t)$ at t_i , and on the approximation of the function $\mathbf{v}(t)$ in the middle point of the interval (t_{i-1}, t_i) , denoted by $\mathbf{v}_{i-1/2}$, the approximations belonging to the time instants later by Δt are calculated as

$$\mathbf{v}_{i+1/2} := \mathbf{v}_{i-1/2} + \Delta t \cdot \overline{\mathbf{f}}(t_i, \mathbf{u}_i, \mathbf{v}_{i-1/2})$$

and

$$\mathbf{u}_{i+1} := \mathbf{u}_i + \Delta t \cdot \mathbf{v}_{i+1/2}$$

and then the next time interval can follow. (At the beginning of the whole analysis, i.e. at the first interval when $i = 0$, $\mathbf{v}_{i-1/2}$ can be approximated by the given initial value $\mathbf{v}_0$.)

Due to its simplicity and effectiveness, this method is very widely applied: it can be found in several commercial softwares like PFC, UDEC, EDEM etc.

3.4.4 Newmark's β-method

As every ***implicit*** technique, Newmark's β-mehod analyses the (t_i, t_{i+1}) interval to find the unknowns belonging to t_{i+1} so that the equations of motion would be satisfied at the endpoint, i.e. at t_{i+1}. The method applies for second-order differential equations like the equations of motion.

Let us search for the solution of the equation

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

in which

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

and given the $\mathbf{u}(t_0) = \mathbf{u}_0$, $\mathbf{v}(t_0) = \mathbf{v}_0$ initial values. For those $\mathbf{u}(t)$, $\mathbf{v}(t)$, $\mathbf{a}(t)$ functions which exactly satisfy the equations of motion, the residual function

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

would be constantly zero for every t ; these $\mathbf{u}(t)$, $\mathbf{v}(t)$, $\mathbf{a}(t)$ functions would indeed describe the real history of the analysed system. Assume now that the $\mathbf{u}_i$, $\mathbf{v}_i$ and $\mathbf{a}_i$ numerical solutions which belong to t_i indeed satisfy the equations; and our aim is to find the $\mathbf{u}_{i+1}$, $\mathbf{v}_{i+1}$ and $\mathbf{a}_{i+1}$ belonging to $t_{i+1} = t_i + \Delta t$ so that

$$\mathbf{r}(t_{i+1}, \mathbf{u}_{i+1}, \mathbf{v}_{i+1}, \mathbf{a}_{i+1}) = 0 \ .$$

According to Newmark's β-method the velocities and accelerations at t_{i+1} are approximated in terms of the unknown $\mathbf{u}_{i+1}$ with the help of the parameters β and γ which control the behaviour of the time integration:

$$\mathbf{a}_{i+1} := \frac{1}{\beta \cdot \Delta t^2}\left[\mathbf{u}_{i+1} - \left(\mathbf{u}_i + \Delta t \cdot \mathbf{v}_i + \frac{\Delta t^2}{2}(1-2\beta)\mathbf{a}_i\right)\right]$$
$$\mathbf{v}_{i+1} := \mathbf{v}_i + (1-\gamma) \cdot \Delta t \cdot \mathbf{a}_i + \gamma \cdot \Delta t \cdot \mathbf{a}_{i+1}$$

These formulas are based on the following basis:

- The velocity at t_{i+1} is approximated with the help of an "average" acceleration, $\mathbf{a}_\gamma$:

$$\mathbf{v}_{i+1} := \mathbf{v}_i + \Delta t \cdot \mathbf{a}_\gamma$$

  where

$$\mathbf{a}_{\gamma} := (1-\gamma)\mathbf{a}_i + \gamma \cdot \mathbf{a}_{i+1} \ ,$$

– the displacement at t_{i+1} is calculated from another "average" acceleration, $\mathbf{a}_{\beta}$:

$$\mathbf{u}_{i+1} := \mathbf{u}_i + \Delta t \cdot \mathbf{v}_i + \frac{\Delta t^2}{2}\mathbf{a}_{\beta}$$

where

$$\mathbf{a}_{\beta} := (1-2\beta)\mathbf{a}_i + 2\beta \cdot \mathbf{a}_{i+1}$$

After some rearrangement, the above two formulas are received. (Note that for special values of β and γ the method becomes equivalent to other, previously shown methods. For $\beta = \gamma = 0$, for example, the explicit Euler-method is given: the velocities and accelerations are assumed to be the same along the whole interval as at its beginning point, and no iteration is needed for the calculation. For $\beta = 0$ and $\gamma = 1/2$, the method of central differences, also explicit, is given. For other values of β and γ other time integration methods are received, being mostly implicit.)

If choosing an approximation for $\mathbf{u}_{i+1}$, the two formulas provide an approximation to $\mathbf{v}_{i+1}$ and $\mathbf{a}_{i+1}$. These can be inserted into the equation $\mathbf{r}(t_{i+1}, \mathbf{u}_{i+1}, \mathbf{v}_{i+1}, \mathbf{a}_{i+1}) = 0$, where the unknowns are now only the components of $\mathbf{u}_{i+1}$. This (usually nonlinear) equation can then be numerically solved, e.g. with the Newton-Raphson-method, and after finding $\mathbf{u}_{i+1}$ sufficiently exactly, from the two formulas $\mathbf{a}_{i+1}$ and $\mathbf{v}_{i+1}$ are also received.

This method is the fundament of the DDA models.

3.4.5 Remark: Stability of the solution

An important aspect of the time integration of the equations of motion (and of differential equations in general) is *stability*. Stability can loosely be defined as the property of an integration method to keep the errors resulting in the integration process of a given equation bounded at subsequent time steps. An unstable method will make the integration errors increase exponentially, and an arithmetic overflow can be expected even after just a few time steps. Since stability depends not only on the given method but also on the type of problem, for a one-dimensional case the test equation $y' = \lambda y$, where λ is a complex valued constant, is used to characterize the stability properties of a given method. This characterization is performed by defining the set of values of λ and Δt for which the corresponding method is stable.

Algorithms that are stable for some restricted range of values $(\lambda, \Delta t)$ are called *conditionally stable*. When using such methods, the time step should be chosen depending on the characteristics of the problem as defined by λ (or a set of λ). In the case of a nonlinear problem for which the value of λ changes with time, the algorithm may be stable for some part of the integration and unstable for another. Consequently, it is very important when using conditionally stable algorithms to know in advance the range of values $(\lambda, \Delta t)$ for which the method is stable and to compare it with the possible range of λ values of the given problem. For this purpose the *region of absolute stability* of a method is defined as that set of values $(\lambda, \Delta t)$ for which a perturbation in the solution y_i will produce a change in subsequent values which does not increase from step to step. The region of absolute stability is an intrinsic

characteristic of the method which should be considered prior to the use of conditionally stable algorithms. As an example, Euler's method described above is conditionally stable and Δt must be less than $|\lambda|/2$ to assure stability.

An algorithm is said to be *A-stable* or *unconditionally stable* if the solution to $y'=\lambda y$ tends to zero as $i\rightarrow\infty$ when the $\mathrm{Re}(\lambda)<0$, which means that the numerical solution decays to zero whenever the corresponding exact solution decays to zero. An *A*-stable algorithm may be also defined as an algorithm whose region of absolute stability is the complete left half complex plane including the imaginary axis. The most important consequence of the *A*-stability property is that there is no limitation on the size of Δt for the stability of the integration process: this is why A-stable algorithms are also called *unconditionally stable*. Obviously, this property is very important and generally desired in the integration of multibody and other engineering systems, since the analyst would only have to be concerned with the step size for accuracy purposes and not for stability.

The unconditional stability of Newmark's β–method is guaranteed for $2\beta\geq\gamma\geq 1/2$. The method of central differences is, on the other hand, only conditionally stable: the maximal allowed length of the timestep depends on the largest eigenfrequency of the system. Since the method of central differences is most widely applied in DEM, the reader will meet the problem of allowable timestep length e.g. at the BALL-type models and at UDEC.

Questions

3.1. Introduce the Newton-Raphson-method!
3.2. Introduce the Gauss-Seidel-method!
3.3. Introduce the Euler-method!
3.4. Introduce the method of central differences!
3.5. Introduce Newmark's β-method!