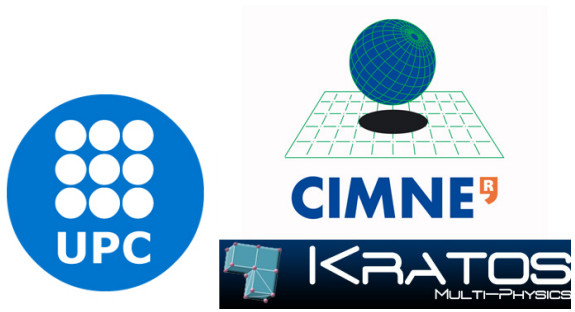
The Finite Element Method for Fluid-Structure Interaction with open source software - Time Integration

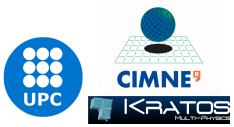
Riccardo Rossi Pooyan Dadvand

Kratos Team

Centre Internacional de Mètodes Numèrics en Enginyeria (CIMNE)



July, 2014, TU Munich



Outline

Linear Structural Dynamics

In *LINEAR* Structural dynamics, the dynamic equilibrium point is normally presented as

$$\mathbf{M}\ddot{\mathbf{d}} + \mathbf{D}\dot{\mathbf{d}} + \mathbf{K}\mathbf{d} = \mathbf{f}_{\mathbf{ext}} \tag{1}$$

where $\mathbf{M}$ is the mass of the system, $\mathbf{D}$ is the "damping", a measure of the dissipation of the system and $\mathbf{K}$ is the stiffness of the system. The vector $\mathbf{d}$ represents here the displacement of the system and $\mathbf{f}_{\mathbf{ext}}$ is the external force.
The time discretization of such system is represented on any textbook of elementary finite elements.
The system to be solved is "second order" in the sense that both first and second derivatives appear in the equations.

Non-Linear Structural Dynamics

In order to proceed further we need however to consider the nonlinear problem which stems out by modifying the linear one:

$$\mathbf{f}_{inertia}\left(\ddot{\mathbf{d}}\right) + \mathbf{f}_{diss}\left(\dot{\mathbf{d}}\right) + \mathbf{f}_{int}\left(\mathbf{d}\right) = \mathbf{f}_{\mathbf{ext}}\left(\mathbf{d}\right) \tag{2}$$

where we introduced a number of new symbols to indicate the "dissipative" contribution, "inertia" contribution as well as the "internal forces". This implies *conceptually* introducing the symbols

$$\mathbf{f}_{inertia}\left(\ddot{\mathbf{d}}\right) := \mathbf{M}\left(\ddot{\mathbf{d}}\right)\ddot{\mathbf{d}}$$
$$\mathbf{f}_{diss}\left(\dot{\mathbf{d}}\right) := \mathbf{D}_{\mathbf{m}}\left(\dot{\mathbf{d}}\right)\dot{\mathbf{d}}$$
$$\mathbf{f}_{int}\left(\ddot{\mathbf{d}}\right) := \mathbf{K}\left(\mathbf{d}\right)\mathbf{d}$$

Newton Raphson algorithm

In order to solve a nonlinear problem we need to write it as a succession of linear steps. conceptually we will write a problem of the type

$$f\left(x\right) = 0 \tag{3}$$

and instead of finding directly x which satisfies the above equation, we assume we have an estimate x_0 and that we want to find iteratively a correction dx so that

$$f\left(x_0 + dx\right) = 0 \tag{4}$$

by applying Taylor expansion

$$f\left(x_0 + dx\right) = f\left(x_0\right) + \frac{\partial f}{\partial x}|_{x=x_0} dx = 0 \tag{5}$$

hence we can find the correction dx by solving

$$-\left(\frac{\partial f}{\partial x}|_{x=x_0}\right) dx = f\left(x_0\right) \; ; \; x_i = x_0 + dx \tag{6}$$

and iterate until the solution is found (usually consider $|x_i - x_{i+1}| < \epsilon$ as a stopping criteria)

$$-\left(\frac{\partial f}{\partial x}|_{x=x_i}\right) dx = f\left(x_i\right) \; ; \; x_{i+1} = x_i + dx \tag{7}$$

Newton Raphson algorithm

remark 1

For a *linear problem* the taylor expansion is **exact**. Hence the Newton Raphson algorithm will deliver the **exact** solution of any linear problem in one iteration *starting from ANY INITIAL GUESS*

remark 2

When all the terms are correctly linearized, that is, when all the derivatives are calculated exactly, the NR "ensures" QUADRATIC CONVERGENCE to the solution. This means that provided that a sufficiently close approximation to the final solution is found the error ϵ will behave like $\epsilon_{n+1} = C\epsilon_n^2$

remark 3

The Newton Raphson technique is the fastest method to make a non linear equation to converge but also the fastest to diverge ... That is, if it works it works really good

Newton Raphson algorithm

in order to solve the original CSD problem we can now use NewtonRaphson procedure. To do so we need to start with an equation to be set to zero... the classical choice is the residual

$$\mathbf{r}(\mathbf{d}) := \mathbf{f_{ext}}(\mathbf{d}) - \mathbf{f}_{inertia}\left(\ddot{\mathbf{d}}\right) - \mathbf{f}_{diss}\left(\dot{\mathbf{d}}\right) - \mathbf{f}_{int}(\mathbf{d}) = \tag{8}$$

to apply NR we need now to compute the "derivatives" of the residual (Jacobian)

$$-\frac{\partial \mathbf{r}}{\partial d} = -\frac{\partial \mathbf{f_{ext}}}{\partial d} + \frac{\partial \mathbf{f}_{int}}{\partial d} + \frac{\partial \mathbf{f}_{inertia}}{\partial d} + \frac{\partial \mathbf{f}_{diss}}{\partial d} \tag{9}$$

by applying the chain rule

$$-\frac{\partial \mathbf{r}}{\partial d} = -\frac{\partial \mathbf{f_{ext}}}{\partial d} + \frac{\partial \mathbf{f}_{int}}{\partial d} + \frac{\partial \mathbf{f}_{inertia}}{\partial \ddot{d}}\frac{\partial \ddot{d}}{\partial d} + \frac{\partial \mathbf{f}_{diss}}{\partial \dot{d}}\frac{\partial \dot{d}}{\partial d} \tag{10}$$

Newton Raphson algorithm

The interpretation of such derivatives is now best understood by considering the original linear problem. In fact for the linear problem we have

$$\mathbf{f}_{int} = \mathbf{K}\mathbf{d} \longrightarrow \frac{\partial \mathbf{f}_{int}}{\partial d} = \mathbf{K} \tag{11}$$

$$\mathbf{f}_{inertia} = \mathbf{M}\ddot{\mathbf{d}} \longrightarrow \frac{\partial \mathbf{f}_{inertia}}{\partial \ddot{d}} = \mathbf{M} \tag{12}$$

$$\mathbf{f}_{diss} = \mathbf{D_m}\dot{\mathbf{d}} \longrightarrow \frac{\partial \mathbf{f}_{diss}}{\partial \dot{d}} = \mathbf{D_m} \tag{13}$$

External forces f_{ext} do not depend on the displacement and as a consequence their derivative is 0.
Taking all of this into account we get

$$-\frac{\partial \mathbf{r}}{\partial d} = -\mathbf{0} + \mathbf{K}\frac{\partial d}{\partial d} + \mathbf{M}\frac{\partial \ddot{d}}{\partial d} + \mathbf{D_m}\frac{\partial \dot{d}}{\partial d} \tag{14}$$

The key is now plugging in the results of $\frac{\partial \ddot{d}}{\partial d}$ and $\frac{\partial \dot{d}}{\partial d}$. **Such terms will be univocally determined once the time integration scheme is chosen**

Newton Raphson algorithm

remark

The relation between a variable and its derivatives is puerely local and it is the same everywhere in the model hence $\frac{\partial \ddot{d}}{\partial d} = \mathbf{I}\frac{\partial \ddot{d}}{\partial d}$

taking this into account each iteration of the Newton Raphson scheme will imply the following steps

1. apply an initial approximation d_0 for d
2. solve the system $-\frac{\partial \mathbf{r}}{\partial d}\Delta d = \mathbf{r}$
3. update solution as $\mathbf{d}_{i+1} = \mathbf{d}_i + \Delta d$
4. update velocity $\dot{\mathbf{d}}$ and acceleration $\ddot{\mathbf{d}}$ according to the time integration scheme
5. check convergence. If not satisfied go back to step 2

remark

The Jacobian matrix **can be approximated**. If such approximation is performed the convergence of the iterative schele gets worst, however AT CONVERGENCE, the same solution is found!!

Solving for velocities instead of pressures

remark

For a dynamic problem one could solve for velocities instead of displacements.

that is a solution strategy of the type

1. apply an initial approximation $\dot{d}_0$ for $\dot{d}$
2. solve the system $-\frac{\partial \mathbf{r}}{\partial \dot{d}} \Delta \dot{d} = \mathbf{r}$
3. update solution as $\dot{\mathbf{d}}_{i+1} = \dot{\mathbf{d}}_i + \Delta d$
4. update displacements and acceleration
5. check convergence. If not satisfied go back to step 2

remark

The residual of the equation is exactly the same as for the first case, when we were solving for displacements. What changes is the LHS of the system

Solving for velocities instead of pressures

when we solve for velocities instead of displacements we need to expand in taylor series around a first approximation written in terms of velocities

$$-\frac{\partial \mathbf{r}}{\partial d} = -\mathbf{0} + \mathbf{K}\frac{\partial d}{\partial \dot{d}} + \mathbf{M}\frac{\partial \ddot{d}}{\partial \dot{d}} + \mathbf{D_m}\frac{\partial \dot{d}}{\partial \dot{d}} \tag{15}$$

The key is now plugging in the results of $\frac{\partial \ddot{d}}{\partial \dot{d}}$ and $\frac{\partial d}{\partial \dot{d}}$. **As before, such terms will be univocally determined once the time integration scheme is chosen**

Similiarities to the structural problem

Let's now focus on Fluids.
A fluid problem can be written (without taking into account stabilization) as

$$\mathbf{M}\dot{\mathbf{u}} + \mathbf{C}(\mathbf{u})\mathbf{u} + \mathbf{L}_\nu \mathbf{u} + \mathbf{G}\mathbf{p} = \mathbf{f} \tag{16}$$

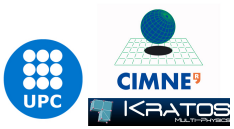
$$\mathbf{D}\mathbf{u} = \mathbf{0} \tag{17}$$

where, as we have been doing until now, we identify the velocity with the symbol $\mathbf{u} = \dot{\mathbf{d}}$
The fluid system is "first order" since only firs derivatives of the unknowns appear in the system.
such two equations could be grouped as

$$\begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \dot{\mathbf{u}} \\ \dot{\mathbf{p}} \end{pmatrix} + \begin{pmatrix} \mathbf{C}(\mathbf{u}) + \mathbf{L}_\nu & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{p} \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ \mathbf{0} \end{pmatrix} \tag{18}$$

The similarity with the structural problem is apparent. we could define an auxiliary variable $\mathbf{y}$ such that

$$\dot{\mathbf{y}} = \begin{pmatrix} \mathbf{u} \\ p \end{pmatrix} \tag{19}$$



Similiarities to the structural problem

by defining the following symbols

$$\bar{\mathbf{M}} := \begin{pmatrix} \mathbf{M} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{20}$$

$$\bar{\mathbf{D}}_m := \begin{pmatrix} \mathbf{C}(\mathbf{u}) + \mathbf{L}_\nu & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{pmatrix} \tag{21}$$

$$\bar{\mathbf{K}} := \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \tag{22}$$

$$\bar{\mathbf{F}} := \begin{pmatrix} \mathbf{f} \\ \mathbf{0} \end{pmatrix} \tag{23}$$

we can write the fluid problem as

$$\bar{\mathbf{M}}\ddot{\mathbf{y}} + \bar{\mathbf{D}}\dot{\mathbf{y}} + \bar{\mathbf{K}}\mathbf{y} = \bar{\mathbf{F}} \tag{24}$$

which looks much like a second order problem.

remark

This form of the structural problem is completely analogous to that of the CSD problem. Furthermore we can see that the structure, when solved for velocities, can be written as a subblock of the fluid system.

There exist in the literature many time integration schemes, both implicit and explicit. Their most important properties are

1. Stability
2. Order of Accuracy

and are normally assessed by considering a test equation of the type $\dot{y} = \lambda y$ where λ could be possible be a complex number.
For the case of Real(λ)¡0 this is a dissipative system whose solution will ultimately tend to zero.

Stability

An important aspect to be considered for the integration of ODE is stability. One can loosely define stability 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

Stability

An algorithm is said to be A-stable if the solution to $\dot{y} = \lambda y$ tends to zero when the Real(λ)¡0, which means that **the numerical solution decays to zero whenever the corresponding exact solution decays to zero**. The most important consequence of the A-stability property is that there is no limitation on the size of Dt for the stability of the integration process: **A-stable algorithms have also been called unconditionally stable in the linear setting**.
The use of A-stable algorithms may become strictly necessary for stiff problems. An important subclass of the A-stable methods is the L-stable ones. A method is said to be **L-stabl**e (also called stiff A-stable), if it is A-stable and when applied to the solution of $\dot{y} = \lambda y$ with Real(λ)¡0, it gives $y_{n+1} = A(\lambda \Delta t) y_n$ where $A(\lambda \Delta t)$ tends to zero as $Real(\lambda) \rightarrow \infty$. **The difference between L-stability and A-stability is that the former damps out the response of the stiff components (that is equations with high λ) very rapidly, almost in only one time step**.

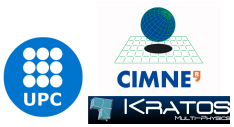
Stability

Algorithms that are stable for some restricted range of values $\lambda \Delta t$ are called **conditionally stable**
Explicit algorithms are generally "Conditionally stable".
the region of absolute stability of a method is defined as that set of values $\lambda \Delta t$ (area of the complex plane) for which a perturbation in the solution yn 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

Order of Accuracy

Time integration algorithms are means to *approximate* the time integration of the equations of interest.
In order to assess the "accuracy" of the different methods, the solution of one time step can be compared to a reference "exact" solution.
This can be done either analytically by using taylor series arguments or "experimentally" to verify the analytical predictions.
In either case, the idea is that the ERROR between the exact and approximate solution is ruled by a term of the type $O(\Delta t^p)$, which represents the first term of the taylor expansion around n which is different between the exact and approximate solution. The exponent "p" is called the "Order of accuracy of the algorithm"

Numerical Dissipation



Theta family of methods

One of the simplest and most popular family of schemes is represented by the "theta family" of methods, suitable for solving first order systems in the form

$$\dot{x} = f(x) \tag{25}$$

In their standard form such approach is not applicable to "second order" systems as the ones stemming out of structural dynamics.
the idea is that the system is discretized as

$$\dot{x}_{n+\theta} = \frac{x_{n+1} - x_n}{\Delta t} = f\left(\theta x_{n+1} + (1-\theta) x_n\right) \tag{26}$$

Theta family of methods

let's now consider a system in the form

$$\mathbf{M}\dot{\mathbf{x}} = (\mathbf{F} - \mathbf{D_m}\mathbf{x}) \tag{27}$$

now let's consider the three following scenarios:

1. $\theta = 0$
2. $\theta = \frac{1}{2}$
3. $\theta = 1$

Theta family of methods $\theta = 0$ (Linear Case)

the time discrete equation becomes

$$\mathbf{M}\frac{\mathbf{x}_{n+1} - \mathbf{x}_n}{\Delta t} = (\mathbf{F}_n - \mathbf{D_m x_n}) \tag{28}$$

in order to solve for $\mathbf{x}_{n+1}$ we put together all of the terms multiplied by such term to get

$$\mathbf{M}\mathbf{x}_{n+1} = \mathbf{M}\mathbf{x}_n + \Delta t\,(\mathbf{F}_n - \mathbf{D_m x_n}) \tag{29}$$

If we assume that the matrix $\mathbf{M}$ is diagonal, which is easily achieved by doing nodal integration, $\mathbf{x}_{n+1}$ can be obtained without solving any system. This method is *explicit* and conditionally stable. It is generally known as **Forward Euler**

Theta family of methods $\theta = \frac{1}{2}$ (Linear Case)

the time discrete equation becomes

$$\mathbf{M}\frac{\mathbf{x}_{n+1} - \mathbf{x}_n}{\Delta t} = \left(\mathbf{F}_{n+\frac{1}{2}} - \mathbf{D_m}\left(\frac{\mathbf{x_n} + \mathbf{x_{n+1}}}{2}\right)\right) \tag{30}$$

in order to solve for $\mathbf{x}_{n+1}$ we collect all of the terms multiplied by such term to get

$$\left(\mathbf{M} + \frac{\Delta t}{2}\mathbf{D_m}\right)\mathbf{x}_{n+1} = \mathbf{M}\mathbf{x}_n + \left(\mathbf{F}_{n+\frac{1}{2}} - \frac{\Delta t}{2}\mathbf{D_m}\mathbf{x_n}\right) \tag{31}$$

The LHS $\mathbf{M} + \frac{\Delta t}{2}\mathbf{D_m}$ is now not anymore diagonal, hence a system need to be solved. **the choice of $\theta = \frac{1}{2}$ leads to the only second-order accurate member of the theta-family of methods**. This method is known as **Crank Nicholson**.

WARNING!

For a Nonlinear problem the operator $\mathbf{D_m}$ is ... non linear. We will have a look to the linearization for such scheme later on

Theta family of methods $\theta = 1$ (Linear Case)

the time discrete equation becomes

$$\mathbf{M}\frac{\mathbf{x}_{n+1} - \mathbf{x}_n}{\Delta t} = \Delta t \left(\mathbf{F}_{n+1} - \mathbf{D_m x_{n+1}}\right) \tag{32}$$

in order to solve for $\mathbf{x}_{n+1}$ we collect all of the terms multiplied by such term to get

$$\left(\mathbf{M} + \Delta t \mathbf{D_m}\right)\mathbf{x}_{n+1} = \mathbf{M}\mathbf{x}_n + \Delta t \mathbf{F}_{n+1} \tag{33}$$

The LHS $\mathbf{M} + \frac{\Delta t}{2}\mathbf{D_m}$ is now not anymore diagonal, hence a system need to be solved. With this choice of θ the scheme is first order accurate. This scheme is known as **Backward Euler**

The BDF Family

A different time discretization approach is constituted by the "Backward Differencing Formula" approaches.
While for the theta family different members of the family were evaluating the function at different positions within the time step, with the BDF methods the RHS is ALWAYS evaluated at t_{n+1}.
the difference is in the evaluation of the time derivative:

- BDF1 (First Order Backward Differencing Scheme) –¿ coincides with Backward Euler!

$$\frac{\partial x}{\partial t}|_{n+1} = \frac{\mathbf{x}_{n+1} - \mathbf{x}_n}{\Delta t} \tag{34}$$

- BDF2 (Second Order Backward Differencing Scheme)

$$\frac{\partial x}{\partial t}|_{n+1} = \frac{1}{2\Delta t}\left(3x_{n+1} - 4x_n + x_{n-1}\right) \tag{35}$$

remark 1:

The members of the multistep family are also known as linear "multistep" methods as they require multiple past steps for the evaluation of the time derivatives

remark 2:

Other members of the family exist. Nevertheless it is proved that no A-stable method exist of order greater than 2.

The BDF Family

the application of the BDF family to a first order ODE gives

$$\frac{1}{2\Delta t}\left(3x_{n+1} - 4x_n + x_{n-1}\right) = f\left(x_{n+1}\right) \tag{36}$$

as observed before the RHS is only evaluated at t_{n+1}
this can be rewritten as

$$c_0 x_{n+1} + c_1 x_n + c_2 x_{n-1} = f\left(x_{n+1}\right) \tag{37}$$

with $c_0 := \frac{3}{2\Delta t}$, $c_1 := -\frac{4}{2\Delta t}$ and $c_2 := \frac{1}{2\Delta t}$

Non Linear Case

Non linear problems can be solved by applying the NR procedure to the residual of the dynamic equation. That is we need to find x such that the residual

$$\mathbf{r}(x) := (\mathbf{F}_{n+1} - \mathbf{D_m x_{n+1}}) - \mathbf{M}\dot{x} \tag{38}$$

becomes zero.

If we make the decision of solving for x, we will need to evaluate $\frac{\partial \dot{x}}{\partial x}|_{n+1}$

- Theta family

$$\frac{\partial \dot{x}}{\partial x} = \frac{1}{\Delta t}\mathbf{I} \tag{39}$$

- BDF2 scheme (BDF1 identical to the theta family)

$$\frac{\partial \dot{x}}{\partial x} = \frac{3}{2\Delta t}\mathbf{I} \tag{40}$$

Non Linear Case

In order to complete the definition of the problem we also need to derive the other terms, that is, to derive $\mathbf{F}_{n+1} - \mathbf{D_m x_{n+1}}$.
In fluids it is normally assumed that the external forces do not depend on the solution, hence we may assume $\frac{\partial F}{\partial x} = 0$.
On the other hand CONCEPTUALLY we would do

$$\frac{\partial \mathbf{D}_m}{\partial \mathbf{x_{n+1}}} = \mathbf{D_m} + \frac{\partial \mathbf{D_m}}{\partial \mathbf{x_{n+1}}} \mathbf{x_{n+1}} \tag{41}$$

While this is possible (although the linearization is not performed as shown) it is generally accepted to take

$$\frac{\partial \mathbf{D_m}}{\partial \mathbf{x_{n+1}}} \approx \mathbf{D_m} \tag{42}$$

Note that this approximation "spoils" the convergence (convergence will be linear and not quadratic) but DOES NOT affect the final result.

Application to second order problems

Until now we presented a number of integrators which are suitable for "first order" problems.
can those integrators be used for second order systems? The answer is yes, since we can observe that if equlibrium gives

$$\dot{\mathbf{u}} = f\left(...\right) \tag{43}$$

the definition of velocities tells that

$$\dot{d} = \mathbf{u} \tag{44}$$

hence if we define $\mathbf{y} = (\mathbf{u}, \mathbf{d})$ and $\mathbf{g} = (f, \mathbf{u})$ than we get a first order system of the type

$$\dot{\mathbf{y}} = g\left(...\right) \tag{45}$$

Newmark family

The newmark family is described by the following finite difference formulas which relate displacement, velocities and accelerations. It is applicable directly to second order systems.

$$\mathbf{d}_{n+1} = \mathbf{d}_n + \delta t \mathbf{u}_n + \delta t^2 \left(\frac{1-2\beta}{2} \dot{\mathbf{u}}_n + \beta \dot{\mathbf{u}}_{n+1} \right) \tag{46}$$

$$\mathbf{u}_{n+1} = \mathbf{u}_n + h \left((1-\gamma) \dot{\mathbf{u}}_n + \gamma \dot{\mathbf{u}}_{n+1} \right)$$

This two equations can be solved for velocities and accelerations to give

$$\mathbf{u}_{n+1} = (\mathbf{d}_{n+1} - \mathbf{d}_n) - \left(\frac{\gamma}{\beta} - 1 \right) \mathbf{u}_n - \frac{\delta t}{2} \left(\frac{\gamma}{\beta} - 2 \right) \dot{\mathbf{u}}_n \tag{47}$$

$$\dot{\mathbf{u}}_{n+1} = \frac{1}{\beta (\delta t)^2} (\mathbf{d}_{i+1} - \mathbf{d}_n) - \left(\frac{1}{\beta \delta t} \right) \mathbf{u}_n - \left(\frac{1}{2\beta} - 1 \right) \dot{\mathbf{u}}_n \tag{48}$$

where β and γ are two numeric constants which characterize the particular choice of the member of the family. The definition of the problem is completed by the equilibium equation

$$\mathbf{M} \dot{\mathbf{u}}_{n+1} + \mathbf{D}_m \mathbf{u}_{n+1} + \mathbf{K} \mathbf{d}_{n+1} = \mathbf{f}_{ext}^{n+1}$$

Newmark family

Accuracy is optimal when $\gamma = \frac{1}{2}$, no numerical dissipation is included when $\beta = \frac{1}{4}$.
Unconditional stability is guaranteed as long as $\beta \geq \frac{1}{4}\left(\gamma + \frac{1}{2}\right)^2$.
The Newmark method is very often used in structural mechanics due to its optimal (2nd order) accuracy and to its ease of implementation and robustness. Unfortunately it is **NOT suitable for non-linear problems**, since it appears to conserve poorly the energy.
The problem in the nonlinear range is **related to the lack of dissipation for high frequency modes**

Generalized-alpha method

A number of methods were developed in the past to solve the stability problems of the Newmark scheme in the nonlinear regime.
The idea is to replace the "equilibrium" equation by a modified equation, which happens to coincide with the original one as the time step tends to zero.
The most successfull example of such modification is the so called "Generalized-alpha" family of methods which comes from considering

$$\mathbf{M}\dot{\mathbf{u}}_{n+\alpha_m} + \mathbf{D}_m\mathbf{u}_{n+\alpha_f} + \mathbf{K}\mathbf{d}_{n+\alpha_f} = \mathbf{f}^{ext}_{n+\alpha_f} \tag{49}$$

where we define

$$\dot{\mathbf{u}}_{n+\alpha_m} := \alpha_m\dot{\mathbf{u}}_n + (1-\alpha_m)\,\dot{\mathbf{u}}_{n+1} \tag{50}$$

$$\mathbf{u}_{n+\alpha_f} := \alpha_f\mathbf{u}_n + (1-\alpha_f)\,\dot{\mathbf{u}}_{n+1} \tag{51}$$

$$\mathbf{d}_{n+\alpha_f} := \alpha_f\mathbf{d}_n + (1-\alpha_f)\,\mathbf{d}_{n+1} \tag{52}$$

$$\mathbf{f}^{ext}_{n+\alpha_f} := \alpha_f\mathbf{u}_n + (1-\alpha_f)\,\dot{\mathbf{u}}_{n+1} \tag{53}$$

with this definition one may control the amplification factor ρ_∞ (which should be in the range 0-1) by the following choice of parameters. By taking $\rho_\infty \approx 0$ we get an L-stable method.

$$\alpha_m := \frac{2\rho_\infty - 1}{\rho_\infty + 1} \tag{54}$$

$$\alpha_f := \frac{\rho_\infty}{\rho_\infty + 1} \tag{55}$$

the method is then completed by the definition of the constant γ and β to be used in the Newmark Finite Difference Formulas

$$\gamma = 1/2 - \alpha_m + \alpha_f \tag{56}$$

$$\beta = 1/4\,(1 - \alpha_m + \alpha_f)^2 \tag{57}$$

leading to a second-order accurate method with controllable dissipation.

Bossak member of the family

The Bossak time integrator can be considered as a member of the generalized alpha family where

$$\alpha_f = 0 \tag{58}$$

and

$$\alpha_m = [-1/3, 0] \tag{59}$$

since only the mass term is present, the implementation is slightly simpler. Maximal dissipation is found for $\alpha_m = -\frac{1}{3}$

Runge Kutta scheme

The Runga Kutta scheme is member of a family of schemes which are known for their accuacy and robustness in dealing with stiff equations. Even though there exist implicit versions, we only consider here the explicit ones
It is designed for application to a first order system in the form $\dot{x} = f(t, x)$.
The most known member of the family is the 4 stage member which is a fourth-order method, meaning that the error per step is on the order of Δt^5, with a total accumulated error of order Δt^4
The method can be written as

$$\frac{x_{n+1} - x_n}{\Delta t} = \frac{1}{6}(r_1 + 2r_2 + 2r_3 + r_4) \tag{60}$$

where

$$r_1 := f(t_n, x_n) \tag{61}$$

$$r_2 := f\left(t_n + \frac{\Delta t}{2}, x_n + \frac{\Delta t}{2} r_1\right) \tag{62}$$

$$r_3 := f\left(t_n + \frac{\Delta t}{2}, x_n + \frac{\Delta t}{2} r_2\right) \tag{63}$$

$$r_4 := f(t_n + \Delta t, x_n + \Delta t r_3) \tag{64}$$

comparison of some examples of lagrangian convection

–¿ see attached pdf