Spectral-element method

Lecture notes

The Spectral-Element Method (SEM) is a weak-form method, like the Finite-Element method (FEM). The difference lies upon the choice of basis functions used to interpolate the functions. In the case of SEM, high-order polynomials, named Lagrange polynomials, are used to expand functions. The nature of these polynomials associated with a specific integration rule leads to a key result on the mass matrix structure, which conferes to the SEM all its strength.

Weak form or variational statement

Let us consider the 1-D heat equation (the strong form): find $T(x,t)$ with $x \in [0, L]$ such that

$$\boxed{\rho\, c_p\, \partial_t T = \partial_x \left(\kappa\, \partial_x T\right)}$$

where ρ is the material density, c_p is the heat capacity, and κ is the thermal conductivity.

As previously shown with the FEM, the weak form is obtained by dotting the strong form with a test function $\mathrm{w}(x)$ and integrating over the whole domain. Thus, the weak form writes as

$$\begin{aligned}\int_0^L \mathrm{w}\, \rho c_p\, \partial_t T \mathrm{d}x &= \int_0^L \mathrm{w}\, \partial_x \left(\kappa\, \partial_x T\right) \mathrm{d}x \\ &= -\int_0^L \kappa\, \partial_x \mathrm{w}\, \partial_x T \mathrm{d}x + \int_0^L \partial_x \left(\mathrm{w}\, \kappa\, \partial_x T\right) \mathrm{d}x\end{aligned}$$

which becomes

$$\int_0^L \mathrm{w}\, \rho c_p\, \partial_t T \mathrm{d}x = -\int_0^L \kappa\, \partial_x \mathrm{w}\, \partial_x T \mathrm{d}x + \mathrm{w}\, \kappa\, \partial_x T \Big|_0^L$$

Now let us define the 1D finite-elements:

$$\Omega = \bigcup_e \Omega_e \text{ , with elements } \Omega_e = [X_e, X_{e+1}]$$

Let's also set the reference segment, $\xi \in [-1, 1]$, corresponding to a local level (see Figure 1).

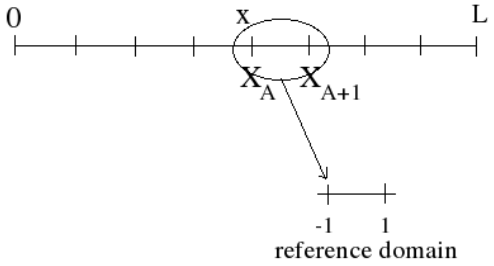

Figure 1: Global to local level.

We need a mapping between the physical 1D segment and the reference segment:

$$x(\xi) = \sum_{a=1}^{2} N_a(\xi) X_a = N_1(\xi) X_1 + N_2(\xi) X_2 \; \xi \in [-1, 1]$$

The shape functions N_a are two degree-1 Lagrange polynomials (the definition of the Lagrange poynomials is below):

$$\begin{cases} N_1(\xi) & = \frac{1}{2}(1-\xi) \\ N_2(\xi) & = \frac{1}{2}(1+\xi) \end{cases}$$

such that $x(-1) = X_1$ and $x(1) = X_2$.

In order to move from the physical domain to the reference domain we need to define the Jacobian as

$$J(\xi) = \frac{\partial x(\xi)}{\partial \xi}$$

which is, with our choice of shape function

$$J(\xi) = \frac{\partial x(\xi)}{\partial \xi} = \frac{1}{2}(X_2 - X_1)$$

Note that we could have chosen higher-order polynomials to get these also called "anchor" shape functions.

Choice of the basis functions

That is the place where SEM departs from FEM in that we do not choose the same shape functions to describe the geometry of the domain and to expand the functions.
In the SEM, the functions are expanded on a basis of Lagrange polynomials:

$$f(x(\xi)) = \sum_{\alpha=0}^{N} f^{\alpha}\, l_{\alpha}^{N}(\xi)$$

where l_{α}^{N} are Lagrange polynomials of degree N defined as

$$l_{\alpha}^{N}(\xi) \equiv \frac{(\xi-\xi_0)...(\xi-\xi_{\alpha-1})(\xi-\xi_{\alpha+1})...(\xi-\xi_N)}{(\xi_\alpha-\xi_0)...(\xi_\alpha-\xi_{\alpha-1})(\xi_\alpha-\xi_{\alpha+1})...(\xi_\alpha-\xi_N)}$$

The central points ξ_α with $\alpha = 0, ..., N$ are the $N+1$ Gauss-Lobatto-Legendre (GLL) points. The GLL points are the roots of the characteristic function

$$(1-\xi^2)P_N'(\xi) = 0$$

where P_N is the degree N Legendre polynomial.

If we have a closer look at the Lagrange polynomials, we see that $\boxed{l_{\alpha}^{N}(\xi_\beta) = \delta_{\alpha\beta}}$ (see Figure 2). It unfolds that the Lagrange polynomials are localized at the GLL points and are orthogonal to each other. The interpolation of the function expands on this polynomial basis, and the function coefficient $f^{\alpha} = f(\xi_\alpha)$.

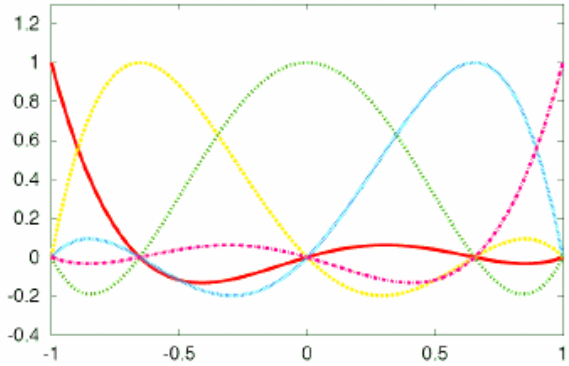

Figure 2: Five degree 4 Lagrange polynomials.

Moving from the entire domain Ω to the elemental domain Ω_e

$$\int_{\Omega} f(x)\mathrm{d}x = \sum_{e} \int_{\Omega_e} f(x)\mathrm{d}x$$

to the reference domain ξ

$$\int_{\Omega_e} f(x(\xi))\mathrm{d}x = \int_{-1}^{1} f(x(\xi))J(\xi)\mathrm{d}\xi\,,$$

we need to define an integration rule to evaluate these integrations. We use the Gauss-Lobatto-Legendre (GLL) quadrature rule which yields to

$$\int_{-1}^{1} f(x(\xi))J(\xi)\mathrm{d}\xi \approx \sum_{\alpha=0}^{N} \hat{\omega}_\alpha\, f^{\alpha}\, J^{\alpha}$$

where $\hat{\omega}_\alpha$ are weights proper to each GLL point and part of the GLL quadrature rule. Thus,

$$\begin{aligned}\int_{\Omega_e} T(x,t)\mathrm{d}x &= \int_{-1}^{1} T(x(\xi),t)\, J(\xi)\mathrm{d}\xi \\ &\approx \sum_{\alpha=0}^{N} \hat{\omega}_\alpha\, T^\alpha(t)\, J^\alpha\end{aligned}$$

2D simulations

Note that in 2D, we use two Lagrange polynomials instead of one

$$x(\xi,\eta) = \sum_{a=1}^{\text{number of anchors}} N_a(\xi,\eta) X_a$$

where N_a is the product of two Lagrange polynomials, and

$$dxdy = Jd\xi d\eta$$

Now we move from the entire domain Ω to the elemental domain Ω_e:

$$\int_\Omega f(x)\mathrm{d}^2x = \sum_e \int_{\Omega_e} f(x)\mathrm{d}^2x$$

to the reference domain (ξ,η)

$$\int_{\Omega_e} f(x)\mathrm{d}^2x = \int_{-1}^{1}\int_{-1}^{1} f(x(\xi,\eta))J(\xi,\eta)d\xi d\eta\,,$$

and use the Gauss-Lobatto-Legendre quadrature rule which yields

$$\int_{-1}^{1}\int_{-1}^{1} f(x(\xi,\eta))J(\xi,\eta)d\xi d\eta \approx \sum_{\alpha,\beta=0}^{N} \hat{\omega}_\alpha\hat{\omega}_\beta\, f^{\alpha\beta}J^{\alpha\beta}$$

where $\hat{\omega}_\alpha$ and $\hat{\omega}_\beta$ are the quadrature weights.

Matricial form

Let us go back to the 1D heat equation. The left hand side term writes as

$$\int_{\Omega_e} \mathsf{w}\,\rho\,c_p\,\partial_t T\,\mathrm{d}x$$

Expanding the test function $\mathsf{w}(x)$ and the temperature $T(x,t)$ yields to

$$\begin{cases} T(x(\xi),t) & \approx \displaystyle\sum_{\alpha=0}^{N} T^\alpha(t)\,l_\alpha^N(\xi) \\ \mathsf{w}(x(\xi)) & \approx \displaystyle\sum_{\beta=0}^{N} \mathsf{w}^\beta\,l_\beta^N(\xi) \end{cases}$$

Notice that the test function is an arbitrary quantity, such that in practice we choose the coefficients w^β to be all equal to 0 except one, which is set to 1, such that we can treat each of the Lagrange polynomials one at a time, which means one GLL point at a time.
Using these expressions and using the quadrature rule, we find for the mass matrix

$$\begin{aligned} \int_{\Omega_e} \mathsf{w}\,\rho c_p\,\partial_t T\mathrm{d}x &= \int_{-1}^{1} \rho(x(\xi))c_p(x(\xi))\,\mathsf{w}(x(\xi))\,\partial_t T(x(\xi),t)J(\xi)\,\mathrm{d}\xi \\ &\approx \sum_{\alpha=0}^{N} \hat{\omega}_\alpha \rho^\alpha c_p^\alpha J^\alpha \sum_{\beta}^{N} \mathsf{w}^\beta\,l_\beta^N(\xi_\alpha) \sum_{\gamma}^{N} \partial_t T^\gamma l_\gamma^N(\xi_\alpha) \\ &= \sum_{\alpha=0}^{N} \hat{\omega}_\alpha \rho^\alpha c_p^\alpha J^\alpha\,\mathsf{w}^\alpha\,\partial_t T^\alpha \end{aligned}$$

and for the stiffness matrix

$$\begin{aligned} \int_{\Omega_e} \kappa\,\partial_x \mathsf{w}\,\partial_x T\mathrm{d}x &= \int_{-1}^{1} \kappa(x(\xi))\,[\partial_x \mathsf{w}(x(\xi))]\,[\partial_x T(x(\xi),t)]J(\xi)\,\mathrm{d}\xi \\ &\approx \sum_{\alpha=0}^{N} \hat{\omega}_\alpha \kappa^\alpha [\sum_{\beta}^{N} \mathsf{w}^\beta\,l_\beta'^N(\xi_\alpha)\partial_x\xi(\xi_\alpha)]\,[\sum_{\gamma}^{N} T^\gamma l_\gamma'^N(\xi_\alpha)\partial_x\xi(\xi_\alpha)]J^\alpha \end{aligned}$$

If you look closely at the expression of the mass matrix, you notice that the mass matrix is diagonal, which is the direct result of choosing the Lagrange polynomials as the basis functions together with the GLL quadrature rule. A diagonal mass matrix leads to a fully local scheme.

If we choose

$$\begin{cases} \mathsf{w}^{\alpha_1} = 1 \\ \mathsf{w}^\alpha = 0,\ \forall\ \alpha \neq \alpha_1 \end{cases}$$

then the heat equation weak form looks like

$$\begin{aligned} \hat{\omega}_{\alpha_1}\rho^{\alpha_1}c_p^{\alpha_1}J^{\alpha_1}\,\partial_t T^{\alpha_1} &= -\sum_{\alpha=0}^{N} \hat{\omega}_\alpha \kappa^\alpha l_{\alpha_1}'^N(\xi_\alpha)\partial_x\xi(\xi_\alpha)\,[\sum_{\gamma}^{N} T^\gamma l_\gamma'^N(\xi_\alpha)\partial_x\xi(\xi_\alpha)]\,J^\alpha \\ &= -\sum_{\gamma=0}^{N}\sum_{\alpha=0}^{N} \hat{\omega}_\alpha \kappa^\alpha\,\left[l_{\alpha_1}'^N(\xi_\alpha)\partial_x\xi(\xi_\alpha)\right]\,\left[l_\gamma'^N(\xi_\alpha)\partial_x\xi(\xi_\alpha)\right]\,J^\alpha\,T^\gamma \\ &= -\sum_{\gamma=0}^{N}\sum_{\alpha=0}^{N} \hat{\omega}_\alpha \kappa^\alpha J^\alpha\,l_{\alpha_1}'^N(\xi_\alpha)\,l_\gamma'^N(\xi_\alpha)\,[\partial_x\xi(\xi_\alpha)]^2\,T^\gamma \end{aligned}$$

We can then write the following matricial form

$$M_{\alpha_1} \partial_t T^{\alpha_1}(t) = \sum_{\gamma=0}^{N} K_{\alpha_1 \gamma} T^{\gamma}(t)$$

where

$$\begin{cases} M_{\alpha_1} &= \hat{\omega}_{\alpha_1} \rho^{\alpha_1} c_p^{\alpha_1} J^{\alpha_1} \\ K_{\alpha_1 \gamma} &= -\sum_{\alpha=0}^{N} \hat{\omega}_{\alpha} \kappa^{\alpha} J^{\alpha} \; l'^{N}_{\alpha_1}(\xi_\alpha) \, l'^{N}_{\gamma}(\xi_\alpha) \; [\partial_x \xi(\xi_\alpha)]^2 \end{cases}$$

Dealing with every GLL point, this leads to the general matricial form

$$\mathbf{M} \, \partial_{\mathbf{t}} \mathbf{T} = \mathbf{K} \, \mathbf{T}$$

Notice that the mass & stiffness matrices are constructed at the element level, which allows for material heterogeneities within an element. The method is very well suited to account for domain discontinuities and interfaces, through the boundary integrals.

In 1D, the boundary conditions write as

$$\begin{aligned} \mathsf{w}\, \kappa \, \partial_x T &= \mathsf{w}^N \, \kappa^N \sum_{\alpha=0}^{N} T^{\alpha}(t) \, l'^{N}_{\alpha}(\xi_N) \, \partial_x \xi(\xi_N) \quad \text{at } x = L \\ \mathsf{w}\, \kappa \, \partial_x T &= \mathsf{w}^0 \, \kappa^0 \sum_{\alpha=0}^{N} T^{\alpha}(t) \, l'^{N}_{\alpha}(\xi_0) \, \partial_x \xi(\xi_0) \qquad \text{at } x = 0 \end{aligned}$$

Assembly

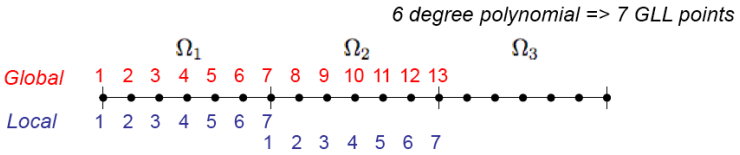

Figure 3: Mapping from the local back to the global level.

As mentioned above, the matrices are constructed at the local level. We do need a mapping to move from the local back to the global level (see Figure 3). You can use classical FE routines to define such a mapping (the SEM code SPECFEM uses a routine originally written by Paul Fischer; one could also play with space-filling Peano curves). To that end, we need an array such that

```
ibool(i,ispec) = j
```

where ispec is the element number, i is the node number locally, and j is the node number globally.

So, if we consider the mass matrix, the assembling looks like:

```
M_global(:) = 0
! loop over the elements
do ispec = 1,Nel
  ! loop over the GLL points
  do i=1,NGLL
    ! local mass matrix contribution
    M_local = wgll(i) * rho(i,ispec) * heat_capacity(i,ispec) * jacobian(i,ispec)
    ! get the global index
    j = ibool(i,ispec)
    M_global(j) = M_global(j) + M_local
  enddo
enddo
```

where we used the associated arrays
$\hat{\omega}_\alpha \rightarrow$ `wgll(i)`,
$\rho^\alpha \rightarrow$ `rho(i,ispec)`,
$c_p^\alpha \rightarrow$ `heat_capacity(i,ispec)` and
$J^\alpha \rightarrow$ `jacobian(i,ispec)`.

Note that since the mass matrix is defined solely by material parameters which are time independent, it is recommended to construct it at the beginning of your code once and for all, outside the main time marching loop.

The same philosophy of assembling is considered for the stiffness mass matrix. In this case however, you need two loops over the GLL points to built the gradients. Then the right-hand side (RHS) term is assembled as

```
RHS_local(:,:) = 0
! loop over all elements
do ispec = 1,Nel
  do i = 1,NGLL
    ! local stiffness matrix contribution for each GLL point
    do j = 1,NGLL
      K_local = 0
      do k = 1,NGLL
        K_local = K_local + wgll(k) * thermal_conductivity(k,ispec) &
                                    * hprime(i,k) * dxidx(k,ispec) &
                                    * hprime(j,k) * dxidx(k,ispec) &
                                    * jacobian(k,ispec)
      enddo
      iglobj = ibool(j,ispec)
      RHS_local(i,ispec) = RHS_local(i,ispec) - K_local * temperature(iglobj)
    enddo
  enddo
enddo
! global assembly loop over the elements (needed for matrix version)
RHS_global(:) = 0
do ispec = 1,Nel
  ! loop over the GLL points
  do i = 1,NGLL
    ! get the global index
    j = ibool(i,ispec)
    RHS_global(j) = RHS_global(j) + RHS_local(i,ispec)
  enddo
enddo
```

You can appreciate how cheap the mass calculation is compared to the RHS term, which also implies that you have calculated the stiffness matrix. Note that you will still need to add the boundary terms. The boundary terms only disappear for Neumann boundaries when the spatial derivative of the temperature would vanish.

Time marching

Once the assembling is done we have

$$\mathbf{M_{global}}\,\partial_{\mathbf{t}}\mathbf{T_{global}} = \mathbf{K_{global}}\,\mathbf{T_{global}}$$

To march this system in time, we use a predictor-corrector scheme, which writes as

- Predictor:

$$\begin{aligned} T_{n+1} &= T_n + \frac{1}{2}\Delta t F_n \\ F_{n+1} &= 0 \end{aligned}$$

- Solve:

$$M\Delta F = \text{RHS}$$

- Corrector:

$$\begin{aligned} F_{n+1} &= F_{n+1} + \Delta F \\ T_{n+1} &= T_{n+1} + \frac{1}{2}\Delta t F_{n+1} \end{aligned}$$

Wave equation

We have seen the heat equation. Now let us have a look at the wave equation. Let us recall that the 1D wave equation is

$$\rho\,\partial_t^2 s = \partial_x(\mu\,\partial_x s)$$

with s denoting displacement and, for example, either one of the following initial & boundary conditions

$$\left\{\begin{array}{lcl} s(x,0) &=& f(x) \\ s(L,t) &=& 0 \\ s(0,t) &=& 0 \end{array}\right. \quad \text{or} \quad \left\{\begin{array}{lcl} s(x,0) &=& f(x) \\ \mu\partial_x s(L,t) &=& B_0(t) \\ \mu\partial_x s(0,t) &=& B_L(t) \end{array}\right.$$

The wave equation weak form is

$$\int_0^L \mathsf{w}\,\rho\,\partial_t^2 s\,\mathrm{d}x = -\int_0^L \mu\,\partial_x \mathsf{w}\,\partial_x s\,\mathrm{d}x + \mathsf{w}\,\mu\,\partial_x s\Big|_0^L$$

and similarly to the heat equation problem we can show that we have

$$\boxed{M_\alpha\,\partial_t^2 S^\alpha = \sum_{\gamma=0}^{N} K_{\alpha\gamma}\,S^\gamma}$$

Following the previous section, we expand $s(x,t)$ on Lagrange basis functions

$$s(x(\xi),t) \approx \sum_{\alpha=0}^{N} s^\alpha(t)\,l_\alpha^N(\xi)$$

The Gauss-Lobatto-Legendre (GLL) quadrature integration rule then writes

$$\begin{aligned} \int_{\Omega_e} s(x,t)\mathrm{d}x &= \int_{-1}^{1} s(x(\xi),t)J(\xi)\mathrm{d}\xi \\ &\approx \sum_{\alpha=0}^{N} \hat{\omega}_\alpha\, s^\alpha(t) J^\alpha \end{aligned}$$

with integration weights $\hat{\omega}_\alpha$.

This leads to the local contributions for the mass matrix

$$\begin{aligned} \int_{\Omega_e} \mathsf{w}\,\rho\,\partial_t^2 s\,\mathrm{d}x &= \int_{-1}^{1} \rho(x(\xi))\,\mathsf{w}(x(\xi))\,\partial_t^2 s(x(\xi),t) J(\xi)\,\mathrm{d}\xi \\ &\approx \sum_{\alpha=0}^{N} \hat{\omega}_\alpha \rho^\alpha J^\alpha \sum_{\beta}^{N} \mathsf{w}^\beta\, l_\beta^N(\xi_\alpha) \sum_{\gamma}^{N} \partial_t^2 s^\gamma l_\gamma^N(\xi_\alpha) \\ &= \sum_{\alpha=0}^{N} \hat{\omega}_\alpha \rho^\alpha J^\alpha\, \mathsf{w}^\alpha\, \partial_t^2 s^\alpha \end{aligned}$$

and for the stiffness matrix

$$\begin{aligned} \int_{\Omega_e} \mu\,\partial_x \mathsf{w}\,\partial_x s\,\mathrm{d}x &= \int_{-1}^{1} \mu(x(\xi))\,[\partial_x \mathsf{w}(x(\xi))]\,[\partial_x s(x(\xi),t)] J(\xi)\,\mathrm{d}\xi \\ &\approx \sum_{\alpha=0}^{N} \hat{\omega}_\alpha \mu^\alpha [\sum_{\beta}^{N} \mathsf{w}^\beta\, l_\beta'^N(\xi_\alpha)\partial_x\xi(\xi_\alpha)][\sum_{\gamma}^{N} s^\gamma l_\gamma'^N(\xi_\alpha)\partial_x\xi(\xi_\alpha)] J^\alpha \end{aligned}$$

Choosing only one test function coefficient $\mathsf{w}^{\alpha_1} = 1$ and all others zero, we find the discretized wave equation

$$\begin{aligned}
\hat{\omega}_{\alpha_1} \rho^{\alpha_1} J^{\alpha_1} \, \partial_t^2 s^{\alpha_1} &= -\sum_{\alpha=0}^{N} \hat{\omega}_\alpha \mu^\alpha l'^N_{\alpha_1}(\xi_\alpha) \partial_x \xi(\xi_\alpha) [\sum_{\gamma}^{N} s^\gamma l'^N_\gamma(\xi_\alpha) \partial_x \xi(\xi_\alpha)] J^\alpha \\
&= -\sum_{\gamma=0}^{N} \sum_{\alpha=0}^{N} \hat{\omega}_\alpha \mu^\alpha \left[l'^N_{\alpha_1}(\xi_\alpha) \partial_x \xi(\xi_\alpha) \right] \left[l'^N_\gamma(\xi_\alpha) \partial_x \xi(\xi_\alpha) \right] J^\alpha \, s^\gamma \\
&= -\sum_{\gamma=0}^{N} \sum_{\alpha=0}^{N} \hat{\omega}_\alpha \mu^\alpha J^\alpha \, l'^N_{\alpha_1}(\xi_\alpha) \, l'^N_\gamma(\xi_\alpha) \left[\partial_x \xi(\xi_\alpha) \right]^2 \, s^\gamma
\end{aligned}$$

with the following matricial form

$$M_{\alpha_1} \, \partial_t^2 s^{\alpha_1}(t) = \sum_{\gamma=0}^{N} K_{\alpha_1 \gamma} \, s^\gamma(t)$$

Note that the stiffness matrix is the same than in the heat equation problem. This time we have a second-order derivation in time, but the assembling procedure is the same as previously discussed.

The boundary terms again are similar to the heat equation and write as

$$\begin{aligned}
\mathsf{w} \, \mu \, \partial_x s &= \mathsf{w}^N \, \mu^N \sum_{\alpha=0}^{N} s^\alpha(t) \, l'^N_\alpha(\xi_N) \, \partial_x \xi(\xi_N) \quad \text{at } x = L \\
\mathsf{w} \, \mu \, \partial_x s &= \mathsf{w}^0 \, \mu^0 \sum_{\alpha=0}^{N} s^\alpha(t) \, l'^N_\alpha(\xi_0) \, \partial_x \xi(\xi_0) \qquad \text{at } x = 0
\end{aligned}$$

Note that implementations *without* these boundary term summations are identical to imposing a Neumann boundary with stresses put to zero, i.e., a stress-free boundary as imposed for example at the free surface. Thus, implementing a free-surface boundary condition in a spectral-element code becomes a trivial task compared to finite-difference methods.

The time scheme that we use for time stepping is the Newmark scheme, which belongs to the family of the predictor-corrector schemes. You are referred to the book by Hughes (1987) for more details. The Newmark scheme keeps track of the displacement, d_n, the velocity, v_n, and the acceleration, a_n.
The classical form of the Newmark algorithm is

$$\begin{array}{ll} Ma_{n+1} = F_{n+1}d_{n+1} & = d_n + \Delta t v_n + (\Delta t)^2 \left[(\frac{1}{2} - \beta)a_n + \beta a_{n+1}\right] \\ v_{n+1} & = v_n + \Delta t \left[(1-\gamma)a_n + \gamma a_{n+1}\right] \end{array}$$

which involves two parameters: β and γ. This scheme is unconditionally stable if $\beta \geq \gamma \geq \frac{1}{2}$ and conditionally stable if $\gamma \geq \frac{1}{2}$ & $\beta < \frac{1}{2}$. You reach a second-order accuracy if and only if $\gamma = \frac{1}{2}$. The angular momentum is conserved if and only if $\beta = 0$ & $\gamma = \frac{1}{2}$.

This latter yields to

- Predictor:

$$\begin{aligned} d_{n+1} &= d_n + \Delta t v_n + \frac{1}{2}\Delta t^2 a_n \\ v_{n+1} &= v_n + \frac{1}{2}\Delta t a_n \\ a_{n+1} &= 0 \end{aligned}$$

- Solve:

$$M\Delta a = F_{n+1}$$

- Corrector:

$$\begin{aligned} a_{n+1} &= a_{n+1} + \Delta a \\ v_{n+1} &= v_{n+1} + \frac{1}{2}\Delta t a_{n+1} \\ d_{n+1} &= d_{n+1} \end{aligned}$$

which is the preferred scheme for second-order derivation in time.

The stability criterion used for the wave equation is two-fold

- we need five grid points per wavelength to accurately resolved a given wavelength
- we use the Courant condition to guarantee numerical stability

Finally, due to its high-order accuracy and diagonality of the mass matrix, the SEM is perfectly suited for parallel computation, allowing also to address large 3D computations.