Computational Seismology

Lecture 6: SEM in SPECFEM code

March 31, 2021

University of Toronto

TABLE OF CONTENTS

SEM in 1D

1D wave equation: strong form

1D wave equation that allows shear modulus to vary spatially for $x \in \Omega = [0, L]$,

$$\rho \ddot{s} = \partial_x(\mu \, \partial_x s) \tag{1}$$

with boundary condition for fixed ends (*Dirichlet B.C.*)

$$s(0, t) = s(L, t) = 0 \tag{2}$$

or given traction on the ends (*Neumannn B.C.*)

$$\mu \partial_x s(0, t) = B_0(t), \quad \mu \partial_x s(L, t) = B_L(t) \tag{3}$$

and initial condition

$$s(x, t = 0) = f(x), \quad \dot{s}(x, t = 0) = 0, \tag{4}$$

Weak form by multiplying test function/integration by parts

$$\int_\Omega \rho w \, \ddot{s} \, dx = \int w \, \partial_x(\mu \, \partial_x s) \, dx = -\int_\Omega \mu(\partial_x w)(\partial_x s) \, dx + [\mu w \partial_x s]\Big|_{x=0}^{x=L}$$

Meshing

Discretize the simulation domain Ω into small elements $\Omega = \cup\Omega_e$, which in the case of 1D domain $[0, L]$ is composed of small line segment $\Omega_i = [x_i, x_{i+1}], i = 1, \cdots N_e$. Define mapping onto standard domain $x \to \xi \in [-1, 1]$ through the two end **anchors** ($\xi = -1$ and $\xi = 1$)

$$N_1(\xi) = \frac{1}{2}(1-\xi), \quad N_2(\xi) = \frac{1}{2}(1+\xi); \quad N_a(\xi_b) = \delta_{ab} \tag{5}$$

in order to **describe the shape of the elements**

$$x(\xi) = \sum_{a=1}^{2} N_a(\xi)x_a = N_1(\xi)x_1 + N_2(\xi)x_2, \tag{6}$$

which recover the anchor coordinates $x(-1) = x_1, x(1) = x_2$. The jacobian of mapping (for integration) requires $\dot{N}_a(\xi)$

$$\frac{\partial x}{\partial \xi} = \sum_{a=1}^{2} \dot{N}_a(\xi)\, x_a = -\frac{1}{2}x_1 + \frac{1}{2}x_2 \tag{7}$$

Shape functions Extension in 2D

For a 2D element with 9 anchors (3 in x and 3 in z direction), $N_a(\xi, \eta)$ are defined as second-order Lagrangian polynomial, and the mapping between the element and standard square is

$$\mathbf{x}(\xi, \eta) = \sum_{a=1}^{9} N_a(\xi, \eta)\mathbf{x}_a, \tag{8}$$

and computing $\frac{\partial \mathbf{x}}{\partial \xi}$ will involve the shape functions $\partial_\xi N_a$'s.

Interpolation and integration

We need to represent a function, $f(x(\xi))$, within an element, and integrate it over the standard element $[-1, 1]$.

In the classical FEM, the shape functions $N_a(\xi)$ are also used as the basis function to expand the function $f(x(\xi))$. Although it may be sufficient for some deformation problems with very small strain, it is often not enough for the wave propagation problem. Instead we use *higher-order* Lagrangian polynomials (order N) to expand the field functions

$$f(x(\xi)) = \sum_{\alpha=0}^{N} f^{\alpha} l_{\alpha}^{N}(\xi) \tag{9}$$

$$\int_{\Omega} f(x)\, dx = \bigcup_{e} \int_{\Omega_e} f(x)\, dx = \bigcup_{e} \int_{-1}^{1} f(x(\xi)) J(\xi)\, d\xi = \bigcup_{e} \sum_{\alpha} \omega_{\alpha} f^{\alpha} J^{\alpha}$$

Derivatives on GLL points

$$\frac{\partial f}{\partial x} = \partial_x \left(\sum_{\alpha=0}^{N} f^{\alpha} l_{\alpha}^{N}(\xi) \right) = \left(\sum_{\alpha=0}^{N} f^{\alpha} \dot{l}_{\alpha}^{N}(\xi) \right) \frac{\partial \xi}{\partial x}. \tag{10}$$

If we evaluate the derivative at a GLL point ξ_β

$$\frac{\partial f}{\partial x}(x(\xi_\beta)) = \left(\sum_{\alpha=0}^{N} f^{\alpha} \dot{l}_{\alpha}^{N}(\xi_\beta) \right) \frac{\partial \xi}{\partial x}(x(\xi_\beta)). \tag{11}$$

For 1D, $\frac{\partial \xi}{\partial x} = 1/|J|$, however, in 2D and 3D, the entire $\frac{\partial \xi}{\partial \mathbf{x}}$ matrix (size 3×3) needs to be stored at all GLL points.

Meshing summary

Meshing summary:

1. Choose a degree number N, and use the GLL libraries to obtain the GLL points ξ_β, $\beta = 0, \cdots, N$, GLL quadrature weights ω_β, as well as the derivative evaluated at the GLL points $\dot{l}^N_\alpha(\xi_\beta)$.
2. mesh the simulation domain, and store coordinates x_i at the anchors for each element.
3. Over an element, store the material property ρ, μ, Jacobian J and the inverse of the Jacobian matrix $\frac{\partial \xi}{\partial \mathbf{x}}$ over all the GLL points ξ_β.
4. setup the mapping from local to global nodal points for assembling later.

Codes for meshing

```
call define_derivative_matrix(xigll,wgll,hprime)

! evenly spaced achors between 0 and 1
do ispec = 1,NSPEC
   x1(ispec) = LENGTH*dble(ispec-1)/dble(NSPEC)
   x2(ispec) = LENGTH*dble(ispec)/dble(NSPEC)
enddo

! set up the mesh properties
do ispec = 1,NSPEC
   do i = 1,NGLL
      rho(i,ispec) = DENSITY
      mu(i,ispec) = RIGIDITY
      dxidx(i,ispec) = 2. / (x2(ispec)-x1(ispec))
      jacobian(i,ispec) = (x2(ispec)-x1(ispec)) / 2.
   enddo
enddo

! set up local to global numbering
iglob = 1
do ispec = 1,NSPEC
   do i = 1,NGLL
      if(i > 1) iglob = iglob+1
      ibool(i,ispec) = iglob
   enddo
enddo
```

Mass matrix

LHS of EOM weak form

$$\int_{\Omega_e} \rho w \ddot{s}\, dx = \int_{-1}^{1} \rho(x(\xi)) w(x(\xi)) \ddot{s}(x(\xi)) J\, d\xi = \sum_{\alpha=0}^{N} \rho^\alpha w^\alpha \ddot{s}^\alpha J^\alpha \omega_\alpha$$

And assume the I'th $w(x)$ is at elemental level the Lagrange polynomial:

$$w^I(x)\big|_{\Omega_e} = l_\beta^N(\xi). \tag{12}$$

and the LHS over a particular e' becomes

$$\int_{\Omega_{e'}} \rho l_\alpha^N(\xi)\, \ddot{s}\, dx = \sum_{\alpha=0}^{N} \rho^\alpha l_\beta^N w^\alpha \ddot{s}^\alpha J^\alpha \omega_\alpha = \rho^\beta \ddot{s}^\beta J^\beta \omega_\beta \delta_{e'e} \tag{13}$$

For this test function $w^I(x)$ corresponding to a particular global nodal point, only those elements including this nodal point x_I, will have contribution to the LHS.

Integration at global level

$$\int_{\Omega} \rho w^I \, \ddot{s} \, dx = \left(\sum_{e;x_I \in \Omega_e} \rho^\beta J^\beta \omega_\beta \right) \ddot{s}^I \tag{14}$$

which means that when we assemble the LHS into a matrix form $M\underline{\ddot{s}}$ for all the global nodal points I, M is diagonal matrix.

```
mass_global(:) = 0.
  do ispec = 1,NSPEC
     do i = 1,NGLL
        iglob = ibool(i,ispec)
        mass_local = wgll(i) * rho(i,ispec) * jacobian(i,ispec)
        mass_global(iglob) = mass_global(iglob) + mass_local
     enddo
  enddo
```

RHS

The RHS of the weak form requires the spatial derivative of the displacement field (i.e, strain)

$$\frac{\partial s}{\partial x}(x(\xi), t)) = \sum_{\alpha=0}^{N} s^{\alpha}(t) \dot{l}_{\alpha}^{N}(\xi) \frac{\partial \xi}{\partial x} \tag{15}$$

and the RHS involving the stiffness matrix becomes

$$\begin{aligned} \int_{\Omega_e} \mu \partial_x w \partial_x s \, dx &= \int_{-1}^{1} \mu \partial_x w \partial_x s \, J \, dx \\ = \quad & \sum_{\alpha=0}^{N} \omega_{\alpha} J^{\alpha} \mu^{\alpha} \dot{l}_{\beta}^{N}(\xi_{\alpha}) \frac{\partial \xi}{\partial x}(\xi_{\alpha}) \left[\sum_{\gamma=0}^{N} s^{\gamma}(t) \dot{l}_{\gamma}^{N}(\xi_{\alpha}) \right] \frac{\partial \xi}{\partial x}(\xi_{\alpha}) \end{aligned} \tag{16}$$

Try write the corresponding code in your PS 3.

B.C.

If the boundary values are not given as natural boundary (traction), we need to evaluate $w\mu\partial_x s$ values on the left boundary (in the first element)

$$w^0\mu^0 \sum_{\gamma=0}^{N} s^\gamma(t)\dot{l}_\gamma^N(\xi_0)\frac{\partial\xi}{\partial x}(\xi_0) \tag{17}$$

and the right boundary (in the last element)

$$w^N\mu^N \sum_{\gamma=0}^{N} s^\gamma(t)\dot{l}_\gamma^N(\xi_N)\frac{\partial\xi}{\partial x}(\xi_N) \tag{18}$$

where $s^0(t) = 0$ and $s^N(t) = 0$ will be supplied as the prescribed boundary condition.

ABCs

Absorbing boundary condition utilizes the one-way equation, and for the right boundary, waves can only propagate towards the right, and no reflections should be generated at this absorbing boundary.

$$\omega^2 s = \frac{\mu}{\rho} k^2 s, \quad \Rightarrow \quad -i\omega\, s = \pm\beta(-ik)\, s \tag{19}$$

which in the time domain becomes

$$\dot{s} = -\beta \partial_x s \tag{20}$$

for right travelling waves $s(x - \beta t)$. Hence

$$\mu \partial_x s = -\rho\beta\dot{s} = -\sqrt{\rho\mu}\dot{s} \tag{21}$$

Assemble and time extrapolation

With all the above, we can symbolically assemble the system into

$$M_{IJ}\ddot{s}_J = K_{IJ}S_J + \text{B.C.}(S, \dot{S}) + \text{Forcing term} \tag{22}$$

The time marching scheme used is a special Newmark scheme (more advanced time schemes have been used) Assuming we already know the $d^n = s^n, v^n = \dot{s}^n, a^n = \ddot{s}^n$ from last time step, for time step $n+1$,

- Predictor at the beginning of time loop

  $$d^{n+1} = d^n + v^n \Delta t + \frac{1}{2} a^n (\Delta t)^2, \quad v^{n+1} = v^n + \frac{1}{2} a^n \Delta t, \quad a^{n+1} = 0$$

- Corrector after solving the assembled linear system (with diagonal mass matrix)

  $$M\Delta a = F(d^{n+1}, v^{n+1}), a^{n+1} = \Delta a, v_{n+1} = v^{\frac{1}{2}}_{n+1} a^{n+1} \Delta t, d^{n+1} = d^{n+1}$$

Time marching: newmark scheme

```
do itime = 1,NSTEP

! predictor
     displ(:) = displ(:) + deltat * veloc(:) + deltatsqover2 * acce
     veloc(:) = veloc(:) + deltatover2 * accel(:)

! solve the linear system and apply BC (see above codes)
! ....

! corrector
     accel(:) = rhs_global(:) / mass_global(:)
     veloc(:) = veloc(:) + deltatover2 * accel(:)
enddo
```

Mesh design specifics

1. Based on the accuracy requirement (and/or the shortest period of the source) τ, as well as smallest wavespeed (S or surface waves), design mesh so that at least 5 points (7-8 points are recommended) per wavelength is achieved by the mesh.

   $$N = \tau(v_{min}/\Delta h) > 5, \quad \Delta h < 5\Delta h/v_{min}$$

   a rule of thumb is that if you use 5 points per wavelength (which is a bit on the low side) and $NGLL = 5$, the $5\Delta h$ is about $1.25\Delta H$, where ΔH is the element size.
2. If the grid (GLL) point spacing Δh has been determined, then the time stepping needs to satisfy that each time step the field only advances less than a fraction of the grid spacing.

   $$\Delta(v/\Delta h)_{max} < c$$

   where c is the Courant number, typically taken around 0.3.
3. Also the mapping between elements and the reference element $\frac{\partial \xi}{\partial x}$ needs to be fairly well behaved so that $J > 0$. Also the mesh quality can be controlled by requiring the skewness of the mesh elements to be below 0.75. This can be an issue for complex 3D models.