Fundamentals of numerical methods

Based on “computational seismology” of Heiner Igel, 2017, Oxford University Press.

Carene Larmat, LANL, carene@lanl.gov

CIG-LLNL workshop
September 18-22, 2017
LA-UR-17-28498
Goals

- Basic understanding of numerical methods
- Understanding of which method is right for a particular problem

Computational

- Categorize “black boxes” and adapt *quality control*.
- Guidelines for Quality control
  - What problems to look for
  - Proper design of simulation
  - Ensure results are correct.

Seismology
Definition of Computational Seismology

• Computing synthetic seismograms
• Using numerical method that is a complete solution to wave propagation equation in arbitrary 3D models.
  – In contrast to ray theory vastly used in tomography
  – In contrast to modal solutions
• Limited to time-domain solutions.

From Bernhard Schuberth:
https://www.geophysik.uni-muenchen.de/~bernhard/
1. “Black boxes”; Community software.
2. Wave propagation equations.
3. Finite Difference; stability
5. Perspectives, uncertainty.
1. Black boxes; Community Software
Paradigm shift:

“Today, the shortest time to results for students involves the use of community software provided by projects like CIG, individual researchers or groups, or community platforms as developed within the VERCE program.”

1. Are you using black boxes? Give examples.
2. Should you handle/use all black boxes the same way?
3. What criteria should you use to adapt your way to use a black box?
4. What makes a software a community software?
5. What are the steps to properly use a community software?
Some vocabulary

• Benchmark
  Comparison of two codes.

• Verification
  Consistency with theory.

• Validation
  Comparison with data.

• Prediction with a numerical code
Computational seismology codes:

- open-source
- 3D -> parallelization

Software Status Description

Software in CIG's open source library are assigned one of the following support categories:

- **Developed**
  - Actively adding features to support improved science or performance by CIG [D_CIG] or by community contributors [D_CONTRIB].

- **Supported**
  - Actively supported, maintained and upgraded by CIG [S_CIG] or by community contributors [S_CONTRIB].

- **Archived**
  - No development activity; not supported. No commitment to updates. [A]

**Developed Codes** have been validated, passed benchmarks established by the appropriate community, and are leading edge codes in geodynamics. Developed codes may either be donated or developed by CIG Staff or the community. These codes are under active development or enhancements and often are actively supported by CIG through maintenance, technical assistance, training and documentation.

**Supported Codes** are mature codes that meet community standards but are no longer undergoing active development. Codes have been benchmarked and documented with examples and references such that they remain useful research tools. Supported codes include codes donated to CIG from members of our community. Minor changes such as bug fixes and binary upgrades are supported.

**Archived Codes.** Bug reports can be submitted via github but no resources are available for its development, maintenance, or support.
Necessary steps to properly work with community software

1. **Verify your implementation**
   - Run the examples and check that waveforms match

2. **Quantification of input/output**
   - What are the unit of each input/output?
   - Scaling: what is a big earthquake/small earthquake?

3. **Setting up problem**
   - What is your signal of interest?
   - What feature is important in your model?
   - What boundary conditions/external force?

4. **Substantiate your results**
   - Through benchmarking, analytical analysis, data …

**This is a take-away box**
http://seismo-live.org/
2. Wave propagation Equations
Wave equation – I

\[ \rho \partial^2_t u_i = \partial_j (\sigma_{ij} + M_{ij}) + f_i \]

where:

- \( u_i \rightarrow u_i(x, t) \) displacement field
- \( v_i \rightarrow v_i(x, t) \) velocity
- \( a_i \rightarrow a_i(x, t) \) acceleration
- \( \sigma_{ij} \rightarrow \sigma_{ij}(x, t) \) stress field
- \( \epsilon_{ij} \rightarrow \epsilon_{ij}(x, t) \) deformation
- \( \rho \rightarrow \rho(x) \) density
- \( c_{ijkl} \rightarrow c_{ijkl}(x) \) elastic “constants”
- \( M_{ij} \rightarrow M_{ij}(x, t) \) seismic moment
- \( f_i \rightarrow f_i(x, t) \) volumetric external force

\[ \mathcal{L}(\epsilon_{ij}) = c_{ijkl}: \epsilon_{kl} \]

\[ P = \rho c v \]
\[
\begin{align*}
\rho \partial_t v_i &= \partial_j (\sigma_{ij} + M_{ij}) + f_i \\
\partial_t \sigma_{ij} &= \mathcal{L} (\partial_t \varepsilon_{ij}) \\
\partial_t \varepsilon_{ij} &= 1/2(\partial_j v_i + \partial_i v_j) \\
\mathcal{L}(\varepsilon_{ij}) &= c_{ijkl} : \varepsilon_{kl}
\end{align*}
\]

where:

- \( u_i \rightarrow u_i(\mathbf{x}, t) \) displacement field
- \( v_i \rightarrow v_i(\mathbf{x}, t) \) velocity
- \( a_i \rightarrow a_i(\mathbf{x}, t) \) acceleration
- \( \sigma_{ij} \rightarrow \sigma_{ij}(\mathbf{x}, t) \) stress field
- \( \varepsilon_{ij} \rightarrow \varepsilon_{ij}(\mathbf{x}, t) \) deformation
- \( \rho \rightarrow \rho(\mathbf{x}) \) density
- \( c_{ijkl} \rightarrow c_{ijkl}(\mathbf{x}) \) elastic “constants”
- \( M_{ij} \rightarrow M_{ij}(\mathbf{x}, t) \) seismic moment
- \( f_i \rightarrow f_i(\mathbf{x}, t) \) volumetric external force
Wave equation – II – boundary conditions

**Time**

\[ u_i(x, t = 0) = u_i^0(x) \quad x \in V \]
\[ \partial_t u_i(x, t = 0) = v_i^0(x) \]

**Free surface**

**Surface wave modeling**

\[ \sigma_{ij} n_j(x, t) = t_i^0(x) \quad x \in S_{\text{free}} \]

**continuity**

Layered models; Multi-domain modeling; coupling

\[ \sigma_{ij} n_j(x, t)^{(1)} = \sigma_{ij} n_j(x, t)^{(2)} \quad x \in S \]
\[ u_i(x, t)^{(1)} = u_i(x, t)^{(2)} \]

**absorbing conditions**

Full space modeling; half-space modeling

Los Alamos National Laboratory

9/19/17 | 15
Discretization

Computers solve finite problems: numerical methods are based on approximations that reduce the real world to a finite world.
Discretization – Number of grid points per wavelength

What is the frequency content of interest?

What is the wavelengths of interest?

$$\frac{N}{\lambda} = \frac{N}{N_\lambda}$$

FROM http://ds.iris.edu/media/product/emc-pem/images/PREM.png

PREM Model

velocity (km/s)

0 2 4 6 8 10 12 14

0 100 200 300 400 500

depth (km)

500 1,000 1,500 2,000 2,500 3,000 3,500 4,000 4,500 5,000 5,500 6,000 6,500

Vertical ground velocity

Raw signal

Low-pass 40s

Low-pass 100s

Los Alamos National Laboratory
Spend some time on simple problems

- Are P- and S- arrival times correct?
- Do you recognize your input source time function in the computed seismograms?
- Is there an analytical solution you can compare your results with?
- Is the radiation pattern correct?
- Is the wavefront shape correct?
- Is the relative amplitude of wave correct?
- Is the polarization of the particle motion correct?

Solution for the displacement created by a double couple in an infinite homogeneous model (Aki and Richards, 2002)

\[
u(r, \theta) = \frac{1}{4\pi \rho} \left[ \frac{1}{r} \int r M_0(t - \tau) d\tau + \frac{1}{r^2} M_0(t - \tau) d\tau + \frac{1}{4\pi \rho^2} \left( \frac{1}{r} \int M_0(t - \tau) d\tau + \frac{1}{r^2} M_0(t - \tau) d\tau \right) \right]
\]

with the radiation patterns \( A^N \) (near-field), \( A^P \) (intermediate-field P-wave), \( A^S \) (intermediate-field S-wave), \( A^{PP} \) (far-field P-wave) and \( A^{PS} \) (far-field S-wave):

\[
\begin{align*}
A^N &= 9 \sin(2\theta) \cos(\phi) \hat{\tau} - 6 \cos(2\theta) \cos(\phi) \hat{\theta} - \cos(\theta) \sin(\phi) \hat{\phi} \\
A^P &= 4 \sin(2\theta) \cos(\phi) \hat{\tau} - 2 \cos(2\theta) \cos(\phi) \hat{\theta} - \cos(\theta) \sin(\phi) \hat{\phi} \\
A^S &= -3 \sin(2\theta) \cos(\phi) \hat{\tau} - 3 \cos(2\theta) \cos(\phi) \hat{\theta} - \cos(\theta) \sin(\phi) \hat{\phi} \\
A^{PP} &= \sin(2\theta) \cos(\phi) \hat{\tau} \\
A^{PS} &= \cos(2\theta) \cos(\phi) \hat{\theta} - \cos(\theta) \sin(\phi) \hat{\phi}
\end{align*}
\]

Elastic wavefield for an explosion in an homogeneous model. No \( \nabla \times \mathbf{u} \)

This is a take-away box
### Meshing – “an underestimated task”

<table>
<thead>
<tr>
<th>Human time</th>
<th>Simulation workflow</th>
<th>CPU time</th>
</tr>
</thead>
<tbody>
<tr>
<td>15%</td>
<td>Design</td>
<td>0%</td>
</tr>
<tr>
<td>80%</td>
<td>Geometry &amp; Meshing</td>
<td>10%</td>
</tr>
<tr>
<td>5%</td>
<td>Solver</td>
<td>90%</td>
</tr>
</tbody>
</table>

Source: E. Casarotti, pers. comm.

**SPECFEM3D & SW4: Regular meshes**

**Partitioning:** balance between computational load and communication.
Spatial discretization strategies

Grid point approximation

Function approximation

Space divided into elements

Continuity between elements

Space divided into elements

Fluxes between elements
Finite Difference Operator

time and space are discretized on regular space-time grids.

\[ \partial_x f(x) = \lim_{dx \to 0} \frac{f(x + dx) - f(x)}{dx} \]

Operator with different accuracy can be determined thanks to Taylor series:

\[ af(x + dx) = a \left[ f(x) + f'(x)dx + \frac{1}{2!} f''(x)dx^2 + \ldots \right] \]
\[ bf(x) = b[f(x)] \]
\[ cf(x - dx) = c \left[ f(x) - f'(x)dx + \frac{1}{2!} f''(x)dx^2 + \ldots \right] \]

\[ \rightarrow \quad \partial_x f \approx \frac{f(x + dx) - f(x - dx)}{2dx} \]
\[ \partial_x^2 f \approx \frac{f(x + dx) - 2f(x) + f(x - dx)}{dx^2} \]

Increasing domain of influence for differential operators for first derivatives. The weights rapidly decreases with distance from the point of evaluation.
Finite Difference Operator

Von Neumann Analysis of stability:

\[
\frac{p_j^{n+1} - 2p_j^n + p_j^{n-1}}{dt^2} = c_j^2 \left[ \frac{p_{j+1}^n - 2p_j^n + p_{j-1}^n}{dx^2} \right]
\]

with \( p_j^n = e^{i(kjdx - \omega ndt)} \)

\[
\to \sin \left(\frac{\omega dt}{2}\right) = c \frac{dt}{dx} \sin \left(\frac{k dx}{2}\right)
\]

Real solution only if:

\[
\epsilon = c \frac{dt}{dx} \leq 1 \quad \to \quad dt = \epsilon \frac{dx}{c}
\]

Known as the Courant-Friedrichs-Lewy CFL condition

\[
\omega = \frac{2}{dt} \sin^{-1} \left[ c \frac{dt}{dx} \sin \left(\frac{k dx}{2}\right) \right]
\]

\[
c(k) = \frac{2}{k. dt} \sin^{-1} \left[ c \frac{dt}{dx} \sin \left(\frac{k dx}{2}\right) \right]
\]

c depends on k!

**Numerical Dispersion**

Finite Difference results for 1D acoustic wave with 34 points per wavelength.
From your problem to computer – I

- Space discretization is imposed by the smallest seismic velocities in the medium and the highest frequencies to be resolved.
- CFL criterion determines the dt
- Numerical dispersion occurs and accumulates for long distances simulations. Change the number of grid points per wavelength.
- Meshing is an important effort in modeling complex 3D models.
3. Finite Difference
Discretization by grid – displacement formulation

- Finite Difference is an example of grid method. The values of the fields related to wave propagation are only known at the grid points.

\[ u(x, t) \rightarrow u(x_j, t^n) \rightarrow \begin{bmatrix} u^1_j \\ \vdots \\ u^{n+1}_j \end{bmatrix} \]

\[ u_j^{n+1} = 2u_j^n - u_j^{n-1} + \frac{dt^2}{\rho_j} f_j^n + \frac{dt^2}{4\rho_j} \begin{bmatrix} [L] & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & [L] \end{bmatrix} [u]^n \]

where \( L \) is a sparse matrix because operations are limited to neighbor points.

Discretization scheme of Finite Difference for 1D acoustic wave equation. We extrapolate the value at the open-circle point using values on the gray circle points. Such a space-time operator is called stencil.
Velocity-stress formulation

Second derivatives replaced by derivatives at 1/2 increment

No derivatives of the material parameters.

\[
\begin{align*}
\rho \partial_t v_i &= \partial_i (\sigma_{ij} + M_{ij}) + f_i \\
\partial_t \sigma_{ij} &= \mathcal{L} (\partial_t \varepsilon_{ij}) \\
\partial_t \varepsilon_{ij} &= \frac{1}{2} (\partial_j v_i + \partial_i v_j)
\end{align*}
\]

\[
\rightarrow \quad v_j^{n+1/2} = \frac{dt \sigma_j^{n+1/2} - \sigma_j^{n-1/2}}{\rho_j} + v_j^{n-1/2} + \frac{dt}{\rho_j} f_j^n
\]

\[
\sigma_{j+1/2}^{n+1} = dt \begin{bmatrix}
[L] & \cdots & 0 \\
\vdots & \ddots & \vdots \\
0 & \cdots & [L]
\end{bmatrix} \begin{bmatrix}\vdots\end{bmatrix} [v]^{n+1/2} + \sigma_{j+1/2}^n
\]

\[
Staggered discretization scheme for 1D.
\]
Diversity of Finite Difference Operators

Staggering…

Advanced methods

<table>
<thead>
<tr>
<th></th>
<th>t+dt</th>
<th>t</th>
<th>t-dt</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-dx</td>
<td>1/12</td>
<td>10/12</td>
<td>1/12</td>
</tr>
<tr>
<td>x</td>
<td>-2/12</td>
<td>-20/12</td>
<td>-2/12</td>
</tr>
<tr>
<td>x+dx</td>
<td>1/12</td>
<td>10/12</td>
<td>1/12</td>
</tr>
</tbody>
</table>

Conventional and Optimal second-order finite-difference operator.

Finite Differences on unstructured grids.

Different strategies to assign medium properties to grid points.
Summary Finite Difference

- Most intuitive, one of the oldest numerical method (Alterman and Karal, 1968)
- Communication limited to near neighbour
- Widely used in exploration geophysics which focuses on body waves

- What is the grid and formulation used? e.g. staggered scheme.
- Time stepping by extrapolation or predictor/corrector scheme?
- How are heterogeneous properties of Earth taken into account, specially in staggered grid?
- How are implemented the free surface boundary? See recent progress in Mozco et al. (2014)
- How are implemented the source terms?
Finite Element

The wavefield is replaced by a finite sum over a basis of functions $\varphi_i$. “Classical” finite element uses linear basis functions.

$$u(x) \approx \bar{u}(x) = \sum_{i=1}^{N} u_i^e(t) \varphi_i(x)$$

$$\rho \partial_t^2 u = \partial_x(\mu \partial_x u) + f \quad \text{for each } x, t$$

$$\int_D \rho \partial_t^2 u \, \varphi_j \, dx = \int_D \partial_x(\mu \partial_x u) \varphi_j \, dx + \int_D f \varphi_j \, dx$$

for each $\varphi_j$

**Galerkin method:** same set of basis functions for interpolation and test functions.
Boundary conditions facilitated thanks to the weak formulation:

\[
\int_D \partial_x (\mu \partial_x u) \varphi_j \, dx = \left[ \mu \partial_x u \varphi_j \right]_0 - \int_D \mu \partial_x u \partial_x \varphi_j \, dx \text{ for each } \varphi_j
\]

Expressed at the element level, with functions limited to element support:

\[
\sum_i \partial_t^2 u_i(t) \int_D \rho \varphi_i \varphi_j \, dx + \sum_i u_i(t) \int_D \mu \partial_x \varphi_i \partial_x \varphi_j \, dx = \int_D f \varphi_j \, dx \text{ for each } \varphi_j
\]

Solving the assembled system:

\[
u_e(t + dt) = dt^2 (M_e^T)^{-1} [f - K_e^T u_e] + 2u_e(t) - u_e(t - dt)
\]

\[u = A_e u_e \rightarrow \text{Communication}\]
Reference Cube

Mapping between "actual" element and reference element is done thanks to shape functions and Jacobian.

\[ x \approx x^{\Omega e} = \sum_a x_e^a N_a (\xi, \delta, \zeta) \]

\[ dv \approx J_a d\xi d\delta d\zeta \]

Quality control on the Jacobian of a mesh using CUBIT.
Spectral Element Method

Finite element with spectral convergence.

- Combination of Lagrange polynomials as interpolants and an integration scheme based on Gauss quadrature: Maday and Patera (1989), Komatitsch and Vilotte (1989)

(a) Gauss-Lobatto-Legendre (GLL) points used to discretize the system.

(b) Lagrange interpolants defined on order 5 GLLs.
Full waveform inversion

\[ \int_D \rho \varphi_i \varphi_j \, dx = \rho \delta_{ij} \]

\[
\begin{bmatrix}
  a & b \\
  c & d
\end{bmatrix}^{-1} = \frac{1}{ad-bc} \begin{bmatrix}
  d & -b \\
  -c & a
\end{bmatrix}
\]

Spectral convergence, geometrical flexibility, reasonable computational cost.

\[
\begin{bmatrix}
  a & 0 \\
  0 & b
\end{bmatrix}^{-1} = \begin{bmatrix}
  1/a & 0 \\
  0 & 1/b
\end{bmatrix}
\]
5. Perspectives
Scalability

Balance between computational load and communication:

\[
\text{speed-up} = \frac{1.0}{\frac{P}{N} + (1 - P)}
\]

with N number of processors and P is the fraction of the code that is parallelized.

Computational volume: CPU-hrs

A grid of 1000 pts by 1000 pts by 1000 pts takes 1hr on 100 cores.
What will be the computational cost of a grid twice finer?

Increase of computational cost for 2D modeling.

How to divide the volume into number of processors? Constraint on wall-clock and memory.