Operator Upscaling and Adjoint State Method

Tetyana Vdovina, William Symes

The Rice Inversion Project
Rice University
vdovina@rice.edu

February 20, 2009
Motivation

- **Ultimate Goal:** Use 3D elastic upscaling algorithm as a core in the inverse solver

- **Principal Tool:** Adjoint state method: gradient = forward + adjoint problems

- **Major Question:** How do we upscale the adjoint problem? Is the upscaling algorithm the same as for the forward problem?
  - If it is, can we reuse as much of the existing code as possible?
  - If it is not, what exactly changes and how can we reuse as much of the existing code as possible?
Code Reusability

- Reasons to reuse as much of the code as possible:
  - numerical solution of the elastic wave equation is complicated
  - upscaling of the elastic wave equation is complicated
  - adjoint state method is complicated

- Solution to the code reusability problem: TSOpt

Need to provide:
- forward and adjoint update functions
- simulator/TSOpt and TSOpt/optimizer interfaces

Figure: courtesy of Marco Enriquez
Decisions

3D elastic upscaling vs. 2D acoustic upscaling

Start with acoustics, since elastic upscaling is more complicated

“Optimize then discretize” vs. “Discretize then optimize”

- get adjoint for the continuous problem, discretize
- less complicated
- additional discretization error

- get adjoint for the discrete problem directly
- more complicated
- no additional errors

Use “discretize then optimize” approach
Forward Acoustic Upscaling Algorithm

- **Pressure-acceleration formulation:**

\[
\frac{1}{c^2} \frac{\partial^2 p}{\partial t^2} + \nabla \cdot (u_x + u_y) = w(t)\delta(x, y),
\]

\[
u_x = -\frac{\partial p}{\partial x}, \quad u_y = -\frac{\partial p}{\partial y}
\]

• boundary and initial conditions

- **Upscale acceleration only:**

  - \(\delta u_x, \delta u_y\) are the subgrid components
  - \(u_x^c, u_y^c\) are the coarse components

- **After discretization mixed finite element method results in a two-scale difference scheme**
Forward Acoustic Upscaling Algorithm (cont.)

Subgrid stencil

Coarse-grid stencil

\[ p_{i+1/2,j+1/2}^{n+1} = 2p_{i+1/2,j+1/2}^n - p_{i+1/2,j+1/2}^{n-1} + c_i^2 \frac{\sqrt{c}}{D_x} \Delta t^2 \left( \delta u_x + u_x^c \right)_{i+1/2,j+1/2}^n \]

+ vertical component + source term

\[ (\delta u_x)_{i,j+1/2}^{n+1} = - \frac{p_{i+1/2,j+1/2}^{n+1} - p_{i-1/2,j+1/2}^{n+1}}{\Delta x}, \]

\[ (u_x^c)_{m,l+1/2}^{n+1} = \sum_j \frac{p_{1/2,j+1/2}^{n+1} - p_{-1/2,j+1/2}^{n+1}}{\Delta x}, \]

eqns for vertical component of subgrid and coarse accelerations
Adjoint Acoustic Upscaling Algorithm (cont.)

Subgrid stencil

- \( p_{i+\frac{1}{2},j+\frac{1}{2}}^{n-1} = 2p_{i+\frac{1}{2},j+\frac{1}{2}}^n - p_{i+\frac{1}{2},j+\frac{1}{2}}^{n+1} + \Delta t^2 D_x (\delta u_x + u_x^c)^n_{i+\frac{1}{2},j+\frac{1}{2}} \)
- \( + \text{ vertical component + source term} \)

- \( (\delta u_x)_{i,j+\frac{1}{2}}^{n-1} = - \frac{c_{i+\frac{1}{2},j+\frac{1}{2}}^2 p_{i+\frac{1}{2},j+\frac{1}{2}}^{n-1} - c_{i-\frac{1}{2},j+\frac{1}{2}}^2 p_{i-\frac{1}{2},j+\frac{1}{2}}^{n-1}}{\Delta x} \),

- \( (u_x^c)_{m,l+\frac{1}{2}}^{n-1} = \sum_j \frac{c_{\frac{1}{2},j+\frac{1}{2}}^2 p_{\frac{1}{2},j+\frac{1}{2}}^{n-1} - c_{-\frac{1}{2},j+\frac{1}{2}}^2 p_{-\frac{1}{2},j+\frac{1}{2}}^{n-1}}{\Delta x} \),

- eqns for vertical component of subgrid and coarse accelerations
Implications

- **For acoustic upscaling:**
  - Need to modify update functions for adjoint problem
  - Modifications are minor

- **For elastic upscaling:**
  - Need to modify update functions for adjoint problem
  - Modifications are likely to be more serious than for acoustics

\[
\left( \rho \frac{\partial v_1}{\partial t}, w \right) = -\left( (\lambda + 2\mu) \frac{\partial u_1}{\partial x} + \lambda \frac{\partial u_2}{\partial y} + \lambda \frac{\partial u_3}{\partial z}, \frac{\partial w}{\partial x} \right) + \text{ four other inner products } + \text{ source term }
\]

- Discretize: \((\lambda + 2\mu) \frac{\partial u_1}{\partial x} \approx (\lambda + 2\mu)_{i,j,k} \frac{(u_1)_{i+1,j,k} - (u_1)_{i,j,k}}{\Delta x}\)

- Take adjoint: \(\frac{(\lambda + 2\mu)_{i+1,j,k}(u_1)_{i+1,j,k} - (\lambda + 2\mu)_{i,j,k}(u_1)_{i,j,k}}{\Delta x}\)

- Reconsider “optimize then discretize” approach?
Current and Future Work

What have we done?
- Derived and implemented linearized and adjoint problems (parallel) for acoustic upscaling problem
- Built an interface between our simulator and TSOpt
- Verified that dot product test works up to machine precision!

What are we doing?
- Building optimizer interface

What is next?
- Do the same for 3D elastic upscaling