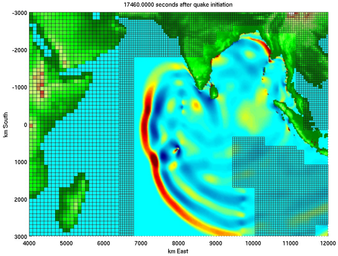
Adaptive Mesh Refinement in CLAWPACK and GeoClaw

Randall J. LeVeque
Department of Applied Mathematics
University of Washington



http://www.clawpack.org

Outline

- CLAWPACK (Conservation Laws Package) and GeoClaw
- Finite volume methods for hyperbolic equations
  - Riemann problems and Godunov's method
  - Wave limiters and high-resolution methods
- Adaptive mesh refinement strategies
- Two applications
  - Tsunami modeling, shallow water equations
  - Seismic waves in heterogeneous earth
- Quadrilateral/hexahedral grids for the sphere

Some collaborators on these projects

Marsha Berger, NYU
David George, UW
Jan Olav Langseth, Norwegian Defence Research Est., Oslo
Donna Calhoun, Commissariat à l'Energie Atomique, Paris
Christiane Helzel, Bochum

Harry Yeh, Civil Engineering, OSU
Roger Denlinger, Dick Iverson, USGS CVO

Supported in part by NSF, DOE

CLAWPACK software — `www.clawpack.org`

- Solves general nonlinear systems of hyperbolic conservation laws (fortran 77, Matlab)
- Version 1.0: 1994, Currently Version 4.3, More than 5000 downloads.
- Finite volume high-resolution Godunov methods (cell averages, solution of “Riemann problems”)
- Shock-capturing methods developed in 1970’s, 80’s originally for compressible gas dynamics (aeronautics, detonations, astrophysics)
- Wave propagation algorithm: general approach for arbitrary hyperbolic systems (also for problems not in conservation form)

CLAWPACK software — AMRCLAW

- Uses Berger-Oliger-Colella style mesh refinement
- Collaboration with Marsha Berger, based on her AMR code for gas dynamics
- "Rectangular grids" — (i, j) grid indexing, *e.g.*, lat-long
- Refinement on rectangular patches (in space and time)
- Refines automatically to follow wave and/or
  in specified regions
- Other AMR wrappers:
  - CHOMBO-CLAW (Colella et al, Calhoun), C++
  - BEARCLAW (Mitran), f90
  - AMROC (Deiterding), C++

High resolution finite volume methods

Hyperbolic conservation law:

$$1D: q_t + f(q)_x = 0 \qquad 2D: q_t + f(q)_x + g(q)_y = 0$$

$$1D: q_t + f'(q)q_x = 0 \qquad 2D: q_t + f'(q)q_x + g'(q)q_y = 0$$

Variable coefficient linear hyperbolic system:

$$1D: q_t + A(x)q_x = 0 \qquad 2D: q_t + A(x,y)q_x + B(x,y)q_y = 0$$

Def: Hyperbolic if eigenvalues of Jacobian $f'(q)$ in 1D or $\alpha f'(q) + \beta g'(q)$ in 2D are real and there exists a complete set of eigenvectors.

Eigenvalues are wave speeds, eigenvectors yield decomposition of data into waves.

Finite-difference Methods

- Pointwise values $Q_i^n \approx q(x_i, t_n)$
- Approximate derivatives by finite differences
- Assumes smoothness

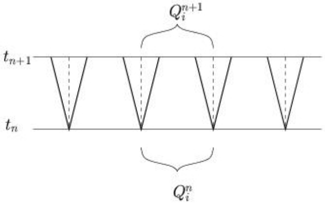
Finite-volume Methods

- Approximate cell averages: $Q_i^n \approx \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} q(x, t_n)\, dx$
- Integral form of conservation law,

$$\frac{\partial}{\partial t} \int_{x_{i-1/2}}^{x_{i+1/2}} q(x, t)\, dx \quad = \quad f(q(x_{i-1/2}, t)) - f(q(x_{i+1/2}, t))$$

leads to conservation law $q_t + f_x = 0$ but also directly to numerical method.

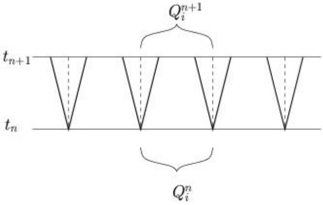
Godunov's Method for $q_t + f(q)_x = 0$



1. Solve Riemann problems at all interfaces, yielding waves $\mathcal{W}^p_{i-1/2}$ and speeds $s^p_{i-1/2}$, for $p = 1,\ 2,\ \ldots,\ m$.

Riemann problem: Original equation with piecewise constant data.

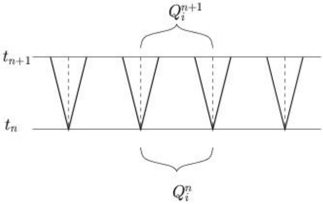
Godunov's Method for $q_t + f(q)_x = 0$



Then either:

1. Compute new cell averages by integrating over cell at t_{n+1},

Godunov's Method for $q_t + f(q)_x = 0$

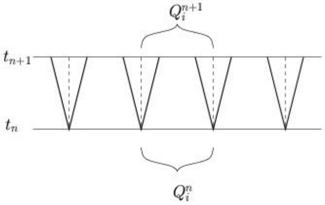


Then either:

1. Compute new cell averages by integrating over cell at t_{n+1},
2. Compute fluxes at interfaces and flux-difference:

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}[F_{i+1/2}^n - F_{i-1/2}^n]$$

Godunov's Method for $q_t + f(q)_x = 0$



Then either:

1. Compute new cell averages by integrating over cell at t_{n+1},
2. Compute fluxes at interfaces and flux-difference:

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}[F_{i+1/2}^n - F_{i-1/2}^n]$$

3. Update cell averages by contributions from all waves entering cell:

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}[\mathcal{A}^+ \Delta Q_{i-1/2} + \mathcal{A}^- \Delta Q_{i+1/2}]$$

where $\mathcal{A}^\pm \Delta Q_{i-1/2} = \sum_{i=1}^{m} (s_{i-1/2}^p)^\pm \mathcal{W}_{i-1/2}^p$.

Wave-propagation form of high-resolution method

$$Q_i^{n+1} = Q_i^n - \frac{\Delta t}{\Delta x}\left[\sum_{p=1}^{m}(s_{i-1/2}^p)^+ \mathcal{W}_{i-1/2}^p + \sum_{p=1}^{m}(s_{i+1/2}^p)^- \mathcal{W}_{i+1/2}^p\right]$$
$$- \frac{\Delta t}{\Delta x}(\tilde{F}_{i+1/2} - \tilde{F}_{i-1/2})$$

Correction flux:

$$\tilde{F}_{i-1/2} = \frac{1}{2}\sum_{p=1}^{M_w}|s_{i-1/2}^p|\left(1 - \frac{\Delta t}{\Delta x}|s_{i-1/2}^p|\right)\widetilde{\mathcal{W}}_{i-1/2}^p$$

where $\widetilde{\mathcal{W}}_{i-1/2}^p$ is a limited version of $\mathcal{W}_{i-1/2}^p$ to avoid oscillations.

(Unlimited waves $\widetilde{\mathcal{W}}^p = \mathcal{W}^p \implies$ Lax-Wendroff for a linear system $\implies$ nonphysical oscillations near shocks.)

Limiter methods

Differencing $\mathcal{W}^p_{i+1/2} - \mathcal{W}^p_{i-1/2}$ approximates q_{xx}.
Gives second order terms in Taylor series (Lax-Wendroff)
This improves solution only if q is sufficiently smooth.

Limiter methods

Differencing $\mathcal{W}^p_{i+1/2} - \mathcal{W}^p_{i-1/2}$ approximates q_{xx}.
Gives second order terms in Taylor series (Lax-Wendroff)
This improves solution only if q is sufficiently smooth.

Limiters: Compare magnitude of $\mathcal{W}^p_{i-1/2}$ to corresponding wave from adjacent Riemann problem in upwind direction.
Apply limiter based on ratio of wave strengths.

Limiter methods

Differencing $\mathcal{W}^p_{i+1/2} - \mathcal{W}^p_{i-1/2}$ approximates q_{xx}.
Gives second order terms in Taylor series (Lax-Wendroff)
This improves solution only if q is sufficiently smooth.

Limiters: Compare magnitude of $\mathcal{W}^p_{i-1/2}$ to corresponding wave from adjacent Riemann problem in upwind direction.
Apply limiter based on ratio of wave strengths.

Host of high-resolution methods developed since late 70's: flux corrected transport, TVD methods, flux limiters, slope limiters, PPM, ENO, WENO, ...

Developed by: Boris, Book, Harten, Zwas, van Leer, Roe, Osher, Zalesak, Sweby, Colella, Woodward, Engquist, Chakravarthy, Shu, ...

Some past applications

- Volcanic flows, dusty gas jets, pyroclastic surges
- Seismic: drumbeat tremors at Mount St. Helens
- Drumlin formation
- Geophysical flow on the sphere
- Flow in porous media, groundwater contamination
- Ultrasound, lithotripsy, shock wave therapy
- Plasticity, nonlinear elasticity
- Electromagnetic waves, photonic crystals
- Hyperbolic equations on general curved manifolds (CLAWMAN)
- Chemotaxis and pattern formation
- Semiconductor modeling
- Traffic flow
- Multi-fluid, multi-phase flows, bubbly flow
- Incompressible flow (projection methods or streamfunction vorticity)
- Combustion, detonation waves
- Astrophysics: binary stars, planetary nebulae, jets
- Magnetohydrodynamics, plasmas
- Relativistic flow, black hole accretion
- Numerical relativity — gravitational waves, cosmology

GeoClaw

TsunamiClaw: (David George) Version of AMRCLAW specifically for tsunami modeling.

- Two dimensional shallow water equations
- Small amplitude waves relative to variations in bathymetry
- Rectangular grid, with dry cells above sea level
- Wet/dry interface moves during inundation
  Need robust "dry-state Riemann solver"

GeoClaw

TsunamiClaw: (David George) Version of AMRCLAW specifically for tsunami modeling.

- Two dimensional shallow water equations
- Small amplitude waves relative to variations in bathymetry
- Rectangular grid, with dry cells above sea level
- Wet/dry interface moves during inundation
  Need robust "dry-state Riemann solver"

GeoClaw: Work in progress

Initially: generalize TsunamiClaw to other depth-averaged flows over topography with dry states, e.g.

- SWE: rivers, estuaries, storm surges
- Dam break problems, flows on steeper topography
- Debris flows: tsunami inundation, volcanos
- Landslides and avalanches
- Multi-layer SWE: internal waves, ocean models

GeoClaw

Future plan: Other geophysical problems involving topography

- Three-dimensional flows over topography
- Two-dimensional vertical slices of such flows
- Volcanic jets and plumes (work with Marica Pelanti)
- Subsurface flows
- Seismic waves
- Coupled problems, e.g. poro-elastic, seismic/tsunami, magma flow/seismic, etc.

GeoClaw

Future plan: Other geophysical problems involving topography

- Three-dimensional flows over topography
- Two-dimensional vertical slices of such flows
- Volcanic jets and plumes (work with Marica Pelanti)
- Subsurface flows
- Seismic waves
- Coupled problems, e.g. poro-elastic, seismic/tsunami, magma flow/seismic, etc.

Desired features?

- General interface to topography/bathymetry data sets,
- Better user interface — Python support
- Interface to other visualization tools — VisIt
- Parallel version

Adaptive Mesh Refinement (AMR)

- Cluster grid points where needed
- Automatically adapt to solution
- Refined region moves in time-dependent problem

Basic approaches:

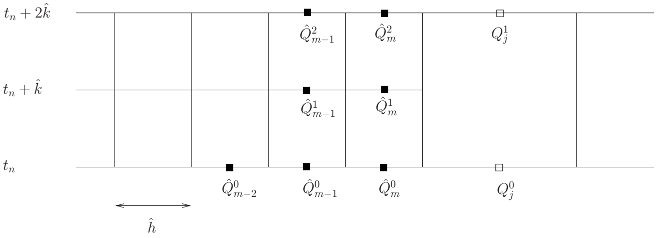
- Cell-by-cell refinement
  Quad-tree or Oct-tree data structure
  Structured or unstructured grid
- Refinement on “rectangular” patches
  Berger-Colella-Oliger style
  (AMRCLAW and CHOMBO-CLAW)

AMR Issues

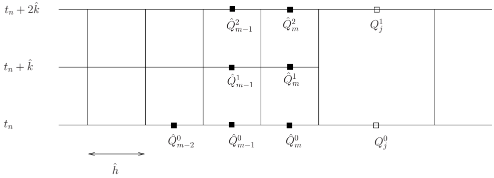
- Refinement in time as well as space
- Conservation at grid interfaces
- Accuracy at interfaces, Spurious reflections?
- Refinement strategy, error estimation
- Clustering flagged points into rectangular patches

Time stepping algorithm for AMR

- Take 1 time step of length k on coarse grid with spacing h.
- Use space-time interpolation to set ghost cell values on fine grid near interface.
- Take L time steps on fine grid.
  $L =$ refinement ratio, $\hat{h} = h/L$, $\hat{k} = k/L$.
- Replace coarse grid value by average of fine grid values on regions of overlap — better approximation and consistent representations.
- Conservative fix-up near edges.



Conservative fix-up



Coarse-grid update: $Q^1_i = Q^0_i - \frac{k}{h}(F^0_{i+1/2} - F^0_{i-1/2}).$

Fine-grid update: $\hat{Q}^{n+1}_i = \hat{Q}^n_i - \frac{\hat{k}}{\hat{h}}(\hat{F}^n_{i+1/2} - \hat{F}^n_{i-1/2}), \quad n = 0,\ 1.$

Corrections: $Q^1_{j-1} := \frac{1}{2}(\hat{Q}^2_{m-1} + \hat{Q}^2_m).$

$$Q^1_j := Q^1_j + \frac{k}{h}\left[\frac{1}{2}(\hat{F}^0_{m+1/2} + \hat{F}^1_{m+1/2}) - F^0_j\right].$$

Global conservation of the total mass:

$$\hat{h}\sum_{i\le m}\hat{Q}_i + h\sum_{i\ge j} Q_j \quad \text{conserved up to boundary fluxes.}$$

Flagging Cells for Refinement

Every `kcheck` time-steps at each level (except finest), check all grid cells and flag those needing refinement.

Use one or more of the following flagging criteria:

- Richardson estimation of truncation error.
  Compare result after last two time steps on this grid with one time step on a coarsened grid.
- Estimate spatial gradient of one or more components of solution.
- Check for regions where refinement is user-forced to some level.
- Problem-specific, e.g. near shore for tsunami simulation.
- Other user-supplied criterion set in `flag2refine.f`.

Clustering Flagged Cells for Refinement

Use Berger-Rigoutsos algorithm
[IEEE Trans. Sys. Man & Cyber.] 21(1991), p. 1278]

Clusters flagged points into a set of rectangular patches.

Tradeoff between:

- Many small patches cover flagged points with minimal refinement of unflagged points.
- But.... increases overhead associated with each patch, e.g. boundary values: ghost cell values set by copying or interpolation from other grids,

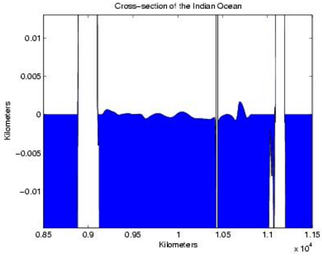
B-G algorithm has cut-off paramter: require that this fraction of refined cells be flagged (usually set to 0.7).

Tsunamis

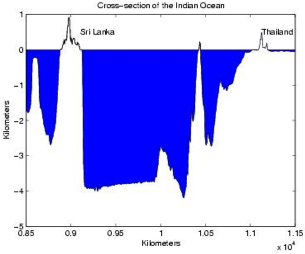
- Small amplitude in ocean (< 1 meter) but can grow to 10s of meters at shore.
- Run-up along shore can inundate 100s of meters inland
- Long wavelength (as much as 200 km)
- Propagation speed $\sqrt{gh}$ (bunching up at shore)
- Average depth of Pacific or Indian Ocean is 4km $\implies$ average speed 200 m/s $\approx$ 450 mph

Cross section of Indian Ocean & tsunami

Surface elevation
on scale of 10 meters:



Cross-section
on scale of kilometers:



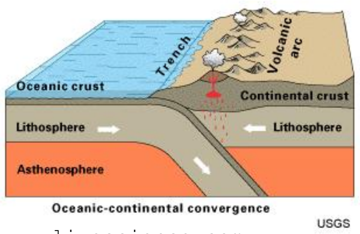
Sumatra event of December 26, 2004

Magnitude 9.1 quake near Sumatra, where Indian tectonic plate is being subducted under the Burma platelet.

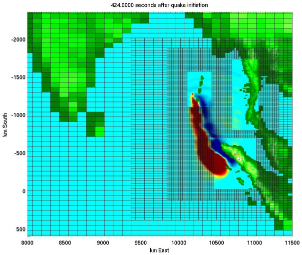
Rupture along subduction zone
$\approx$ 1200 km long, 150 km wide

Propagating at $\approx$ 2 km/sec (for $\approx$ 10 minutes)

Fault slip up to 15 m, uplift of several meters.
(Fault model from Caltech Seismolab.)

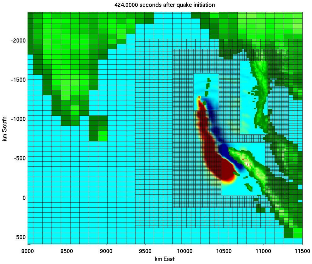


www.livescience.com

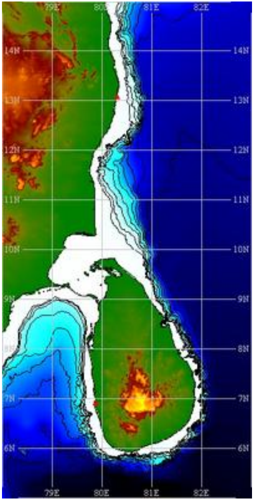


Tsunami simulations

- 2D shallow water + bathymetry
- Finite volume method
- Cartesian grid
- Cells can be dry ($h = 0$)
- Cells become wet/dry as wave moves on shore
- Mesh refinement on rectangular patches
- Adaptive — follows wave, more levels near shore



Local modeling near Madras



Tsunami simulations

Adaptive mesh refinement is essential

Zoom on Madras harbors with 4 levels of refinement:

- Level 1: 1 degree resolution ($\Delta x \approx 60$ nautical miles)
- Level 2 refined by 8.
- Level 3 refined by 8: $\Delta x \approx 1$ nautical mile (only near coast)
- Level 4 refined by 64: $\Delta x \approx 25$ meters (only near Madras)

Factor 4096 refinement in x and y.

Less refinement needed in time since $c \approx \sqrt{gh}$.

Runs in a few hours on a laptop. Movie

Tsunami simulations

Movies:

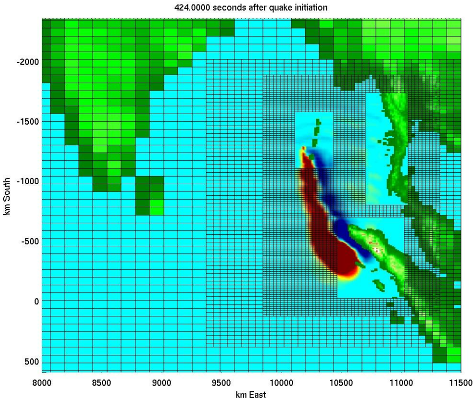
Fault area

Bay of Bengal

Sri Lanka

Indian Ocean

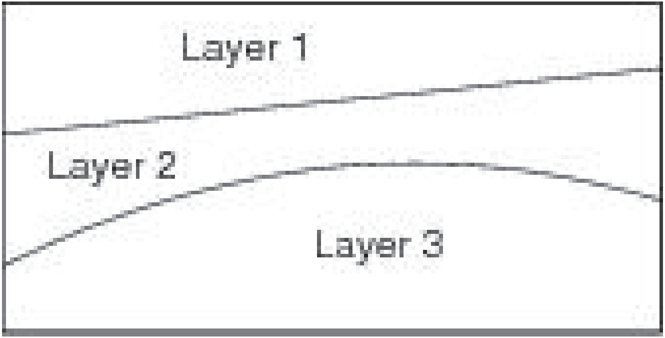
Zoom on Madras



For movies, see

`http://www.amath.washington.edu/∼dgeorge/research.html`

Seismic waves in layered earth



Layers 1 and 3: $\rho = 2, \;\; \lambda = 1, \;\; \mu = 1, \;\; c_p \approx 1.2, \;\; c_s \approx 0.7$

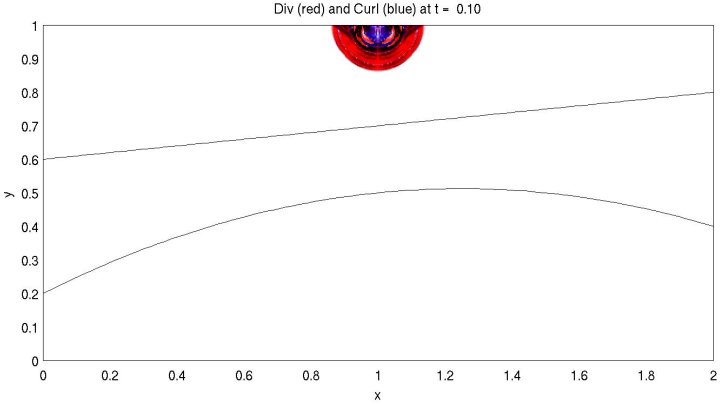
Layer 2: $\rho = 5, \;\; \lambda = 10, \;\; \mu = 5, \;\; c_p = 2.0, \;\; c_s = 1$

Impulse at top surface at $t = 0$.

Solved on uniform Cartesian grid (600×300).

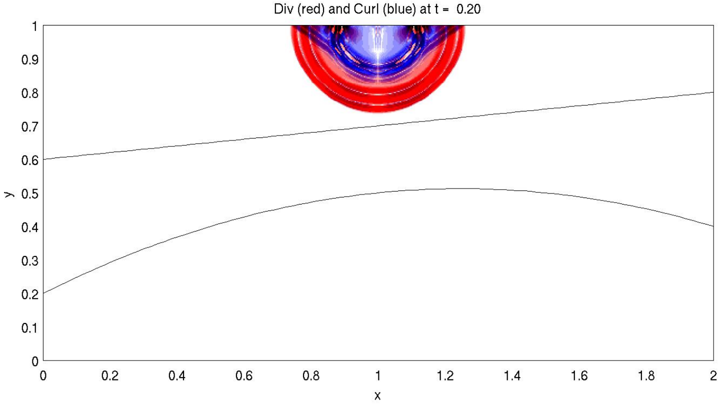
Seismic wave in layered medium

Red = div(u) [P-waves], Blue = curl(u) [S-waves]



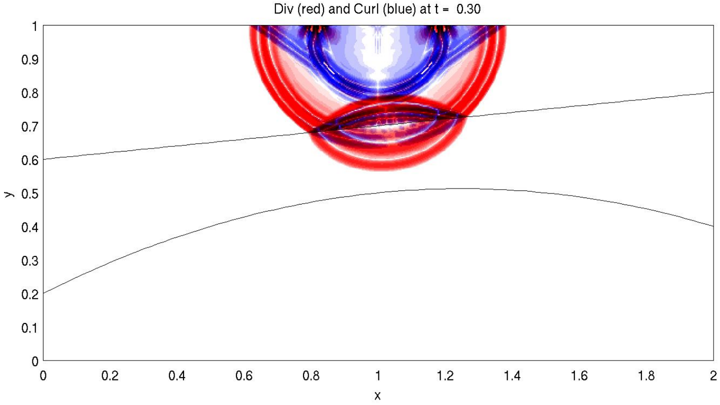
Seismic wave in layered medium

Red = div(u) [P-waves], Blue = curl(u) [S-waves]



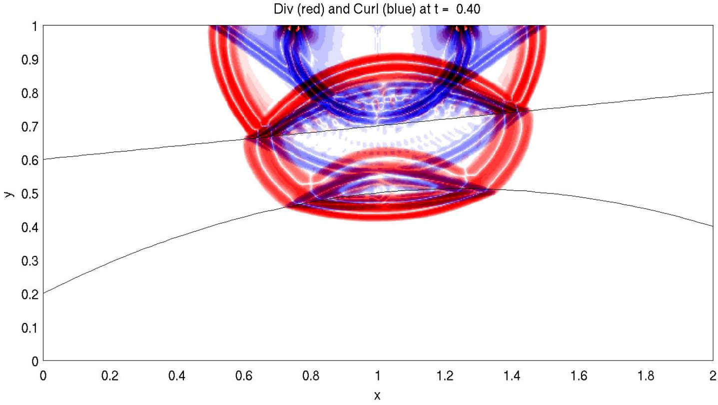
Seismic wave in layered medium

Red = div(u) [P-waves],   Blue = curl(u) [S-waves]



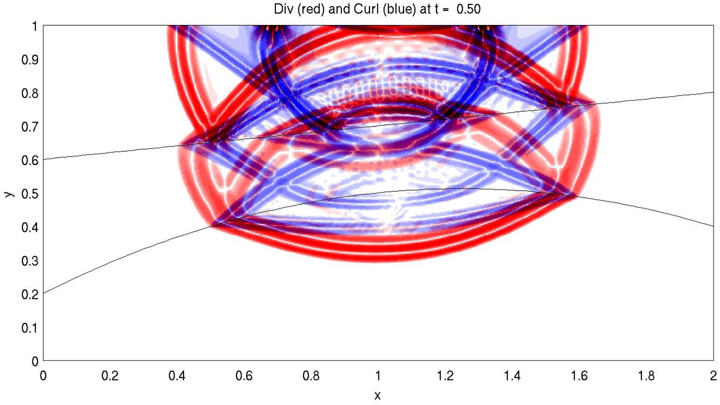
Seismic wave in layered medium

Red = div(u) [P-waves],   Blue = curl(u) [S-waves]



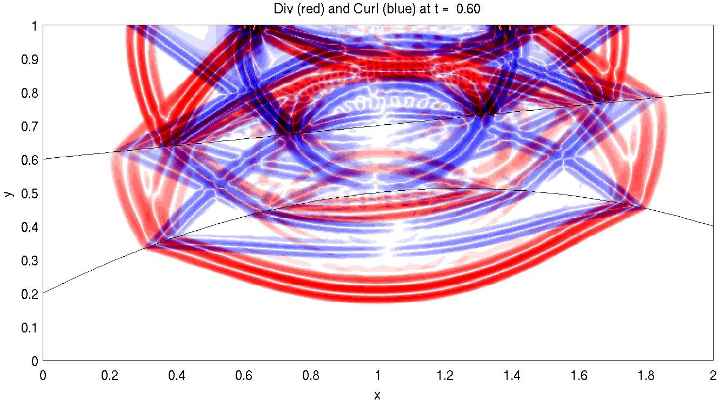
Seismic wave in layered medium

Red = div(u) [P-waves],   Blue = curl(u) [S-waves]



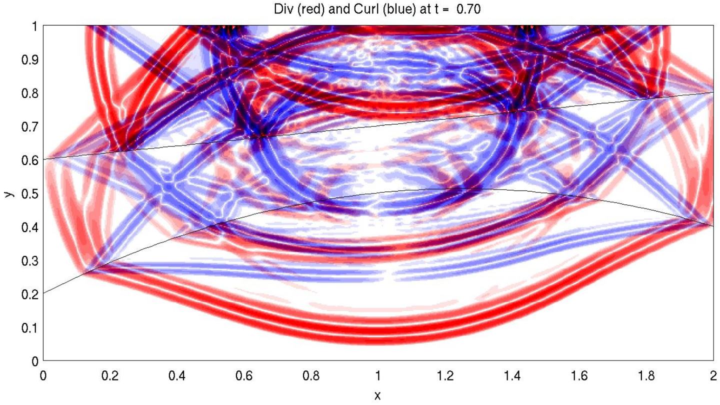
Seismic wave in layered medium

Red = div(u) [P-waves], Blue = curl(u) [S-waves]



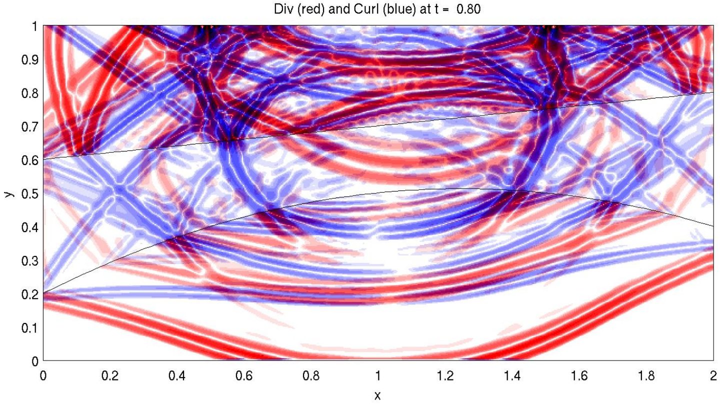
Seismic wave in layered medium

Red = div(u) [P-waves],   Blue = curl(u) [S-waves]



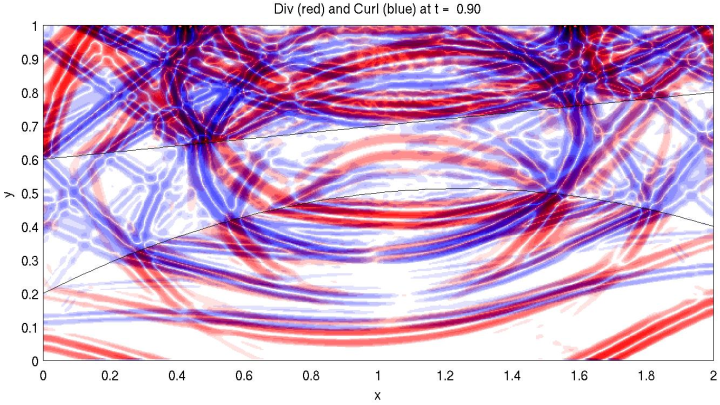
Seismic wave in layered medium

Red = div(u) [P-waves], Blue = curl(u) [S-waves]



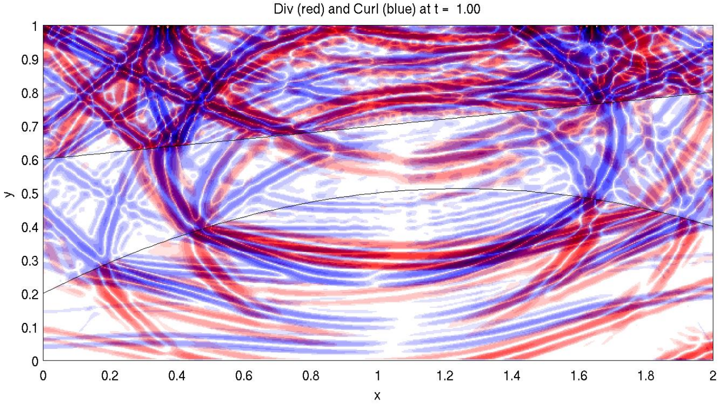
Seismic wave in layered medium

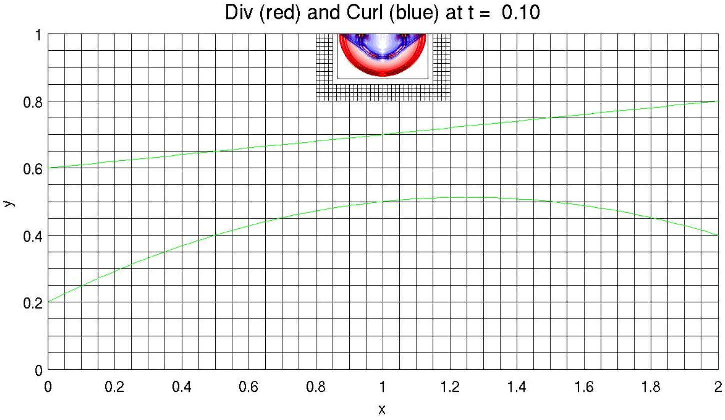
Red = div(u) [P-waves],   Blue = curl(u) [S-waves]



Seismic wave in layered medium

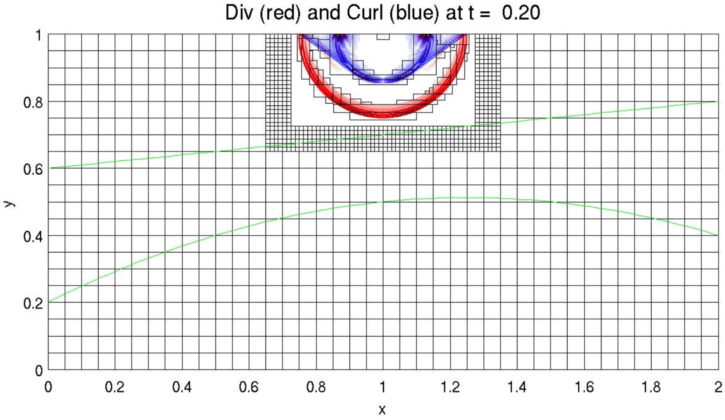
Red = div(u) [P-waves], Blue = curl(u) [S-waves]





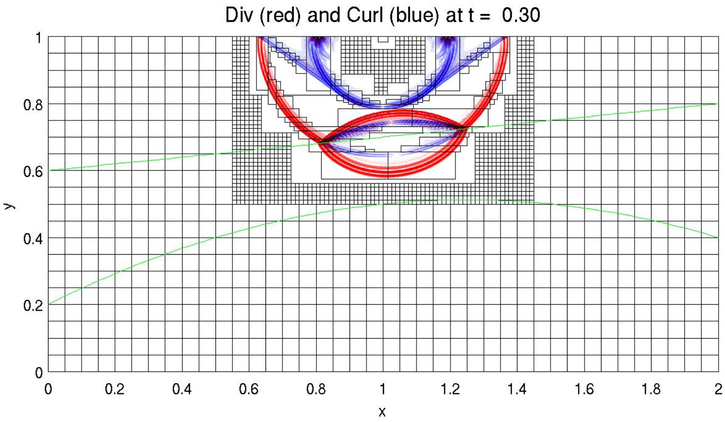
Div (red) and Curl (blue) at t = 0.10
1
0.8
0.6
0.4
0.2
0
y
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x

Sei

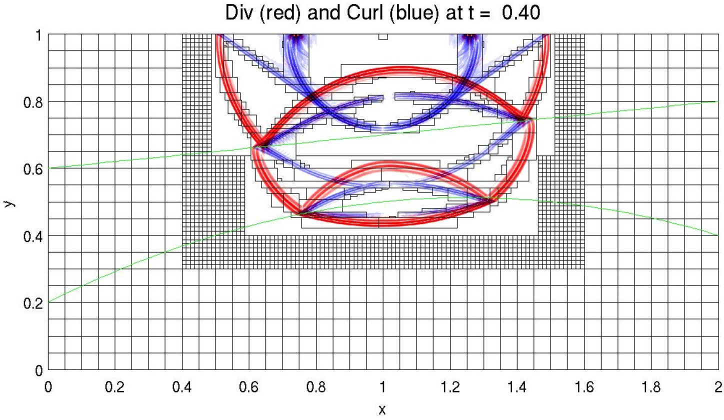


Div (red) and Curl (blue) at t = 0.20
1
0.8
0.6
y
0.4
0.2
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x

Sei

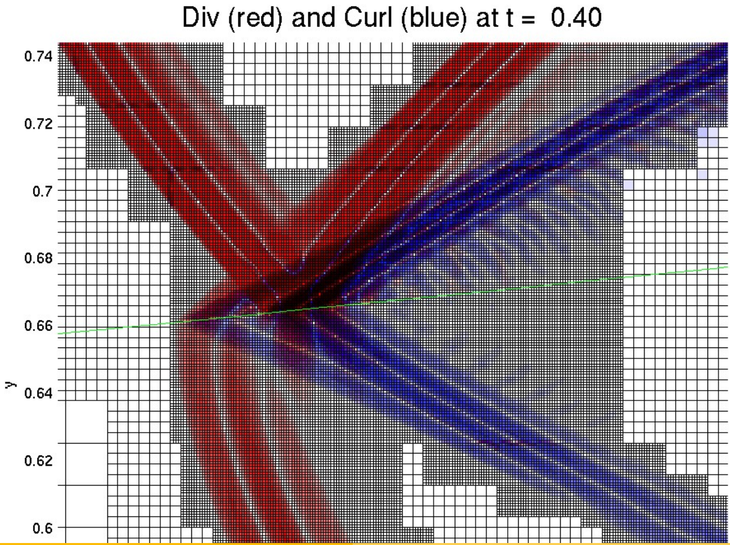


Div (red) and Curl (blue) at t = 0.30
1
0.8
0.6
0.4
0.2
0
y
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x

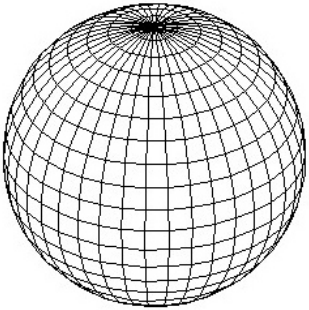

Div (red) and Curl (blue) at t = 0.40
1
0.8
0.6
0.4
0.2
0
y
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
x

Seismic wave in layered medium

Four levels with refinement factors 4, 4, 4



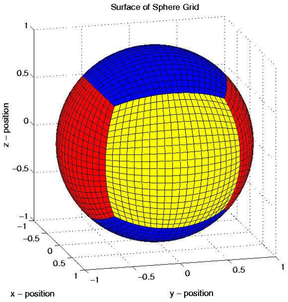
Latitude-Longitude grid on sphere

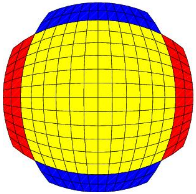


Logically rectangular, but suffers from “pole problem”

- Grid lines coalesce at poles, tiny cells
- Small time steps needed for explicit methods

Cubed Sphere Grid: another popular approach





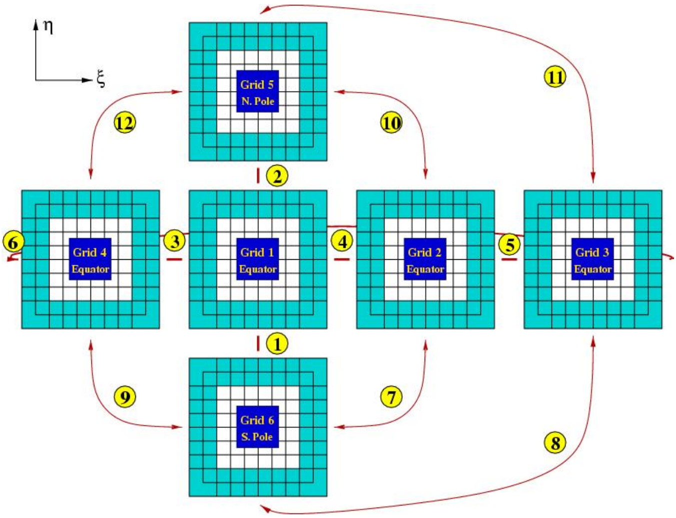
Six logically rectangular grids are patched together.

Data is transferred between patches using ghost cells

Refs: Sadourny (1972), Ronchi, Iacono, Paolucci, Rancic, Purser, Messinger,...

Rossmanith implemented with CLAWPACK

Boundary conditions for cubed sphere

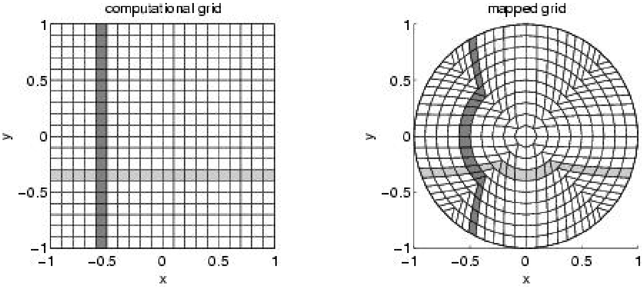


Our approach for circles

Radial projection grid:

Computational domain is square $[-1, 1] \times [-1, 1]$.

Map each point on concentric square of "radius" $d \le 0$ radially inward to circle of radius d.



Movie of radial projection

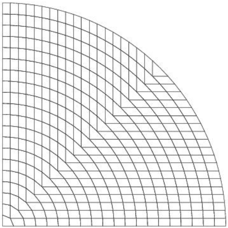
Our approach for circles

Smoother grid:

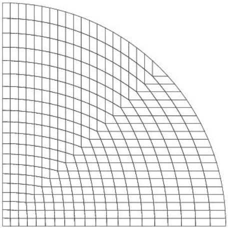
Map line segment $(-d, d)$ to (d, d) to circular arc of radius $R(d)$ passing through the points $(-D(d), D(d))$ and $(D(d), D(d))$.

Similarly in other three quadrants.

$D(d) = d, \;\; R(d) = 1:$



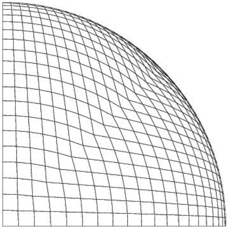
$D(d) = d, \;\; R(d) = 1:$



Our approach for circles and sphere

Redistribute points near boundary:

$D(d) = d(2-d), \quad R(d) = 1:$



Gives good mapping to upper hemisphere
(think of looking down on sphere)

Our approach for sphere

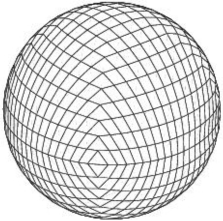
Map $[-1,1]\times[-1,1]$ to unit circle by this approach.

At each point set $z=\sqrt{1-(x^2+y^2)}$.

This defines mapping of $[-1,1]\times[-1,1]$ to upper hemisphere.

Map points in $[-3,-1]\times[-1,1]$ to lower hemisphere by similar mapping.

This defines mapping of rectangle $[-3,1]\times[-1,1]$ to sphere.



Ratio of largest to smallest cell is < 2.

Grid is highly non-orthogonal at a few points near equator.

Movie of mapping

Numerical results on the sphere

Direct application of CLAWPACK — wave-propagation finite volume method

AMRCLAW can also be used.

Movie — advection on the sphere

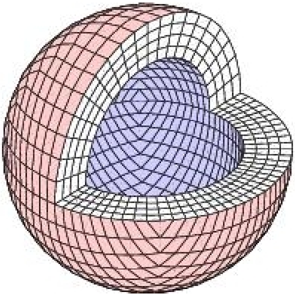
Movie — in computational rectangle

Movie — shallow water on the sphere

Movie — depth vs. "latitude" compared to 1d solution

Our approach for shells

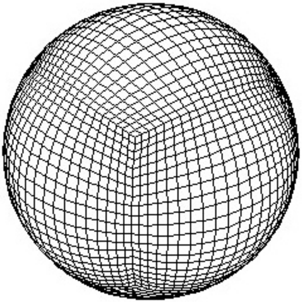
Above approach can be used on sphere and then extended radially:

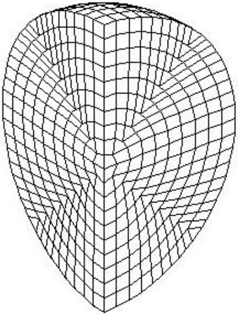


For most applications wouldn't want to extend into origin for full ball — radial lines meet at center and give small cells.

Instead can use 3d version of circle mapping

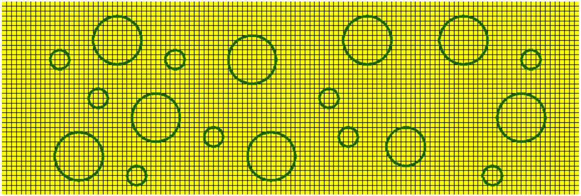
3D hexahedral grid in the ball



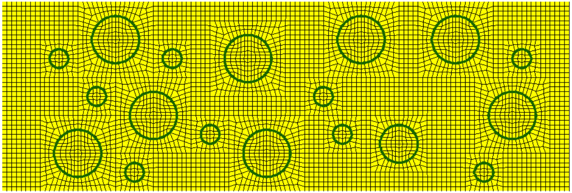


Acoustics with inclusions

120×40 Cartesian grid:

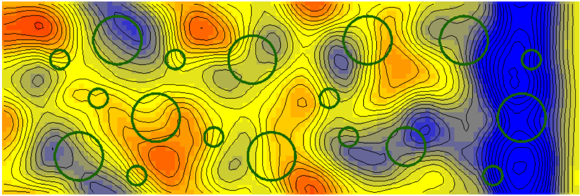


120×40 mapped grid:

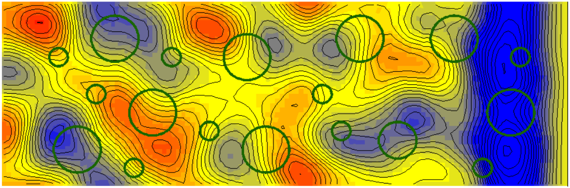


Acoustics with inclusions

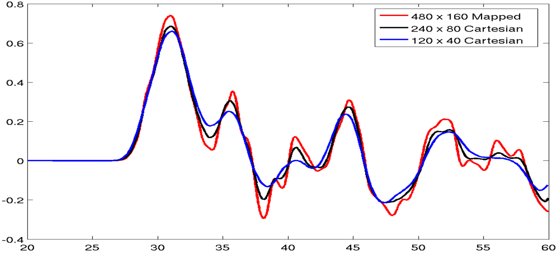
120×40 Cartesian grid:



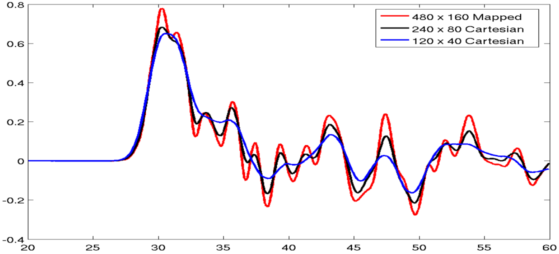
120×40 mapped grid:



Acoustics with inclusions: pressure gauges



$x = 0.5,$



$x = 0.25:$

Conclusions

- High-resolution (shock capturing) methods good also for wave propagation in heterogeneous media.
- Low dispersion, accurately captures reflection and transmission.
- Refinement on rectangular patches is efficient and effective.
- Wave propagation algorithms work well even on highly deformed grids.
- Challenging geophysical flow/wave problems often have special needs.
- Flexible open-source software is useful.