Clawpack Tutorial
Part 2

Randall J. LeVeque
Applied Mathematics
University of Washington

Slides posted at
http://www.clawpack.org/links/tutorials
http://faculty.washington.edu/rjl/tutorials

Outline

- Boundary conditions

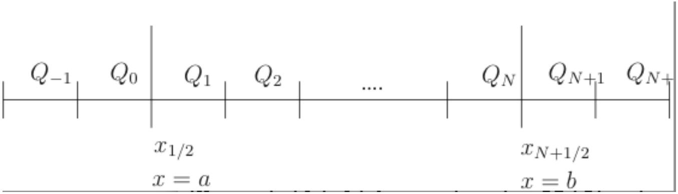
- Python plotting tools
- Specifying plotting parameters
- Options for viewing plots:
  Web pages
  Interactive

- Two space dimensions
  Normal and transverse Riemann solvers

Boundary conditions and ghost cells

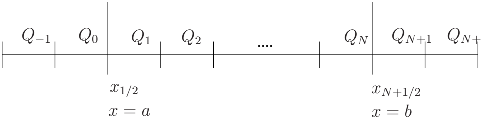
In each time step, the data in cells 1 to $N = \mathtt{mx}$ is used to define ghost cell values in cells outside the physical domain.

The wave-propagation algorithm is then applied on the expanded computational domain, solving Riemann problems at all interfaces.



The data is extended depending on the physical boundary conditons.

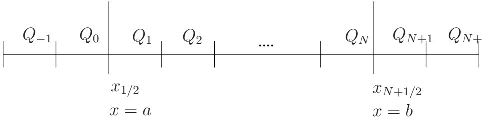
Boundary conditions



Periodic:

$$Q^n_{-1} = Q^n_{N-1}, \quad Q^n_0 = Q^n_N, \quad Q^n_{N+1} = Q^n_1, \quad Q^n_{N+2} = Q^n_2$$

Boundary conditions



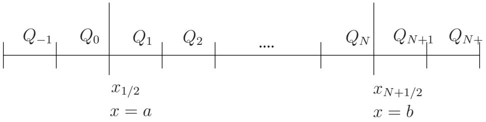
Periodic:

$$Q^n_{-1} = Q^n_{N-1}, \quad Q^n_0 = Q^n_N, \quad Q^n_{N+1} = Q^n_1, \quad Q^n_{N+2} = Q^n_2$$

Extrapolation (outflow):

$$Q^n_{-1} = Q^n_1, \quad Q^n_0 = Q^n_1, \quad Q^n_{N+1} = Q^n_N, \quad Q^n_{N+2} = Q^n_N$$

Boundary conditions



Periodic:

$$Q_{-1}^n = Q_{N-1}^n, \quad Q_0^n = Q_N^n, \quad Q_{N+1}^n = Q_1^n, \quad Q_{N+2}^n = Q_2^n$$

Extrapolation (outflow):

$$Q_{-1}^n = Q_1^n, \quad Q_0^n = Q_1^n, \quad Q_{N+1}^n = Q_N^n, \quad Q_{N+2}^n = Q_N^n$$

Solid wall:

For Q_0 : $\quad p_0 = p_1, \quad u_0 = -u_1,$

For Q_{-1} : $\quad p_{-1} = p_2, \quad u_{-1} = -u_2.$

Setting BCs in Clawpack

In `setrun.py`, at each boundary
(`xlower, xupper, ylower, yupper`)
must specify for example:

```
clawdata.mthbc_xlower = 3  # solid wall
clawdata.mthbc_xupper = 1  # extrapolation
```

Set to 2 at both boundaries for periodic.

Setting BCs in Clawpack

In `setrun.py`, at each boundary
(`xlower, xupper, ylower, yupper`)
must specify for example:

```
clawdata.mthbc_xlower = 3  # solid wall
clawdata.mthbc_xupper = 1  # extrapolation
```

Set to 2 at both boundaries for periodic.

Set to 0 if you want to impose something special.

In this case need to copy **$CLAW/clawpack/1d/lib/bc1.f** to application directory and modify (along with Makefile).

Extrapolation boundary conditions

If we set $Q_0 = Q_1$ then the Riemann problem at $x_{1/2}$ has zero strength waves:

$$Q_1 - Q_0 = \mathcal{W}^1_{1/2} + \mathcal{W}^2_{1/2}$$

So in particular the incoming wave $\mathcal{W}^2$ has strength 0.

Extrapolation boundary conditions

If we set $Q_0 = Q_1$ then the Riemann problem at $x_{1/2}$ has zero strength waves:

$$Q_1 - Q_0 = \mathcal{W}^1_{1/2} + \mathcal{W}^2_{1/2}$$

So in particular the incoming wave $\mathcal{W}^2$ has strength 0.

The outgoing wave perhaps should have nonzero magnitude, but it doesn't matter since it would only update ghost cell.

Ghost cell value is reset at the start of each time step by extrapolation.

Extrapolation boundary conditions

If we set $Q_0 = Q_1$ then the Riemann problem at $x_{1/2}$ has zero strength waves:

$$Q_1 - Q_0 = \mathcal{W}^1_{1/2} + \mathcal{W}^2_{1/2}$$

So in particular the incoming wave $\mathcal{W}^2$ has strength 0.

The outgoing wave perhaps should have nonzero magnitude, but it doesn't matter since it would only update ghost cell.

Ghost cell value is reset at the start of each time step by extrapolation.

In 2D or 3D, extrapolation in normal direction is not perfect but works quite well.

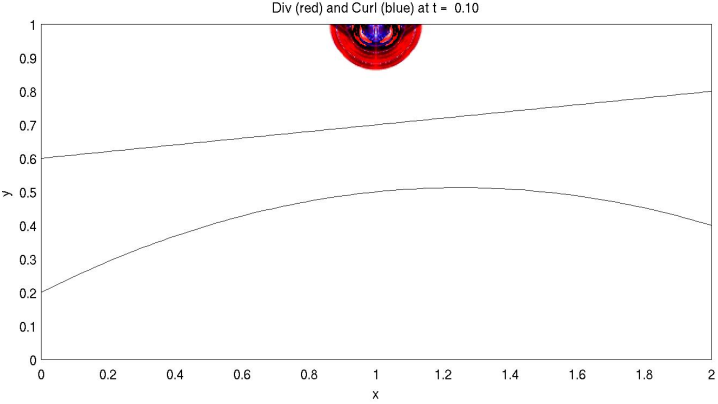
Extrapolation boundary conditions

Examples:

- One-dimensional acoustics:
  $CLAW/book/chap3/acousimple
  $CLAW/apps/acoustics/1d/example2
- Tsunami propagation:
  $CLAW/apps/tsunami/chile2010
- Seismic waves in a half-space on following slides,

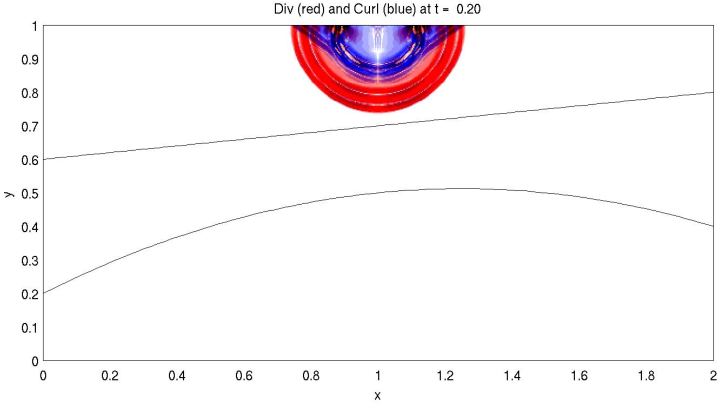
Seismic wave in layered medium

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



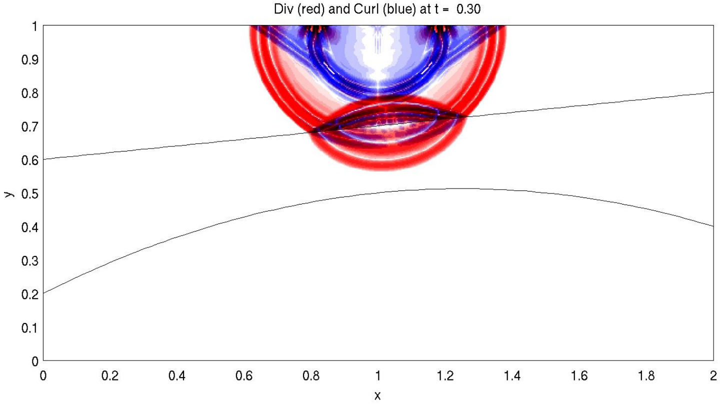
Seismic wave in layered medium

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



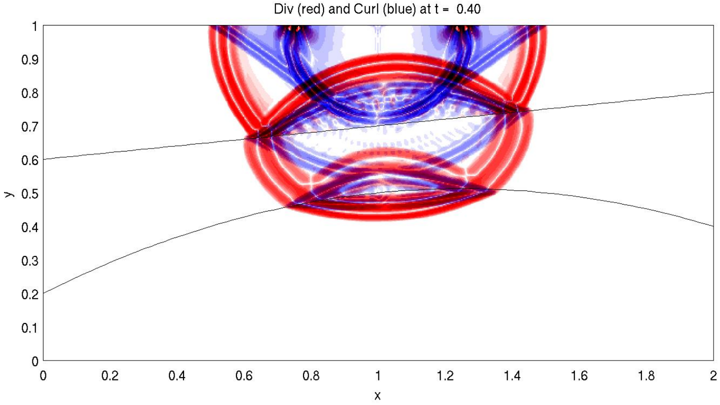
Seismic wave in layered medium

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



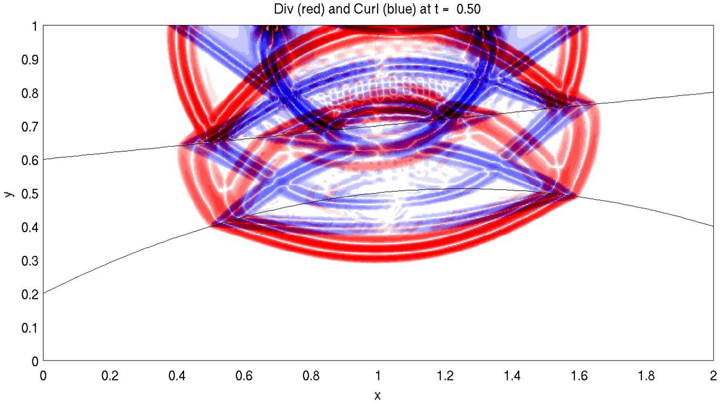
Seismic wave in layered medium

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



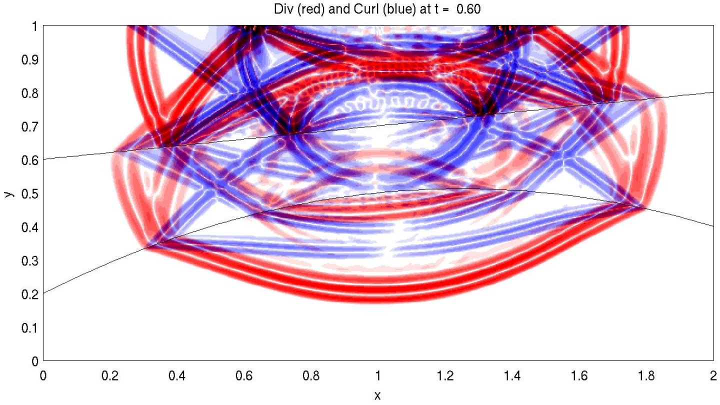
Seismic wave in layered medium

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



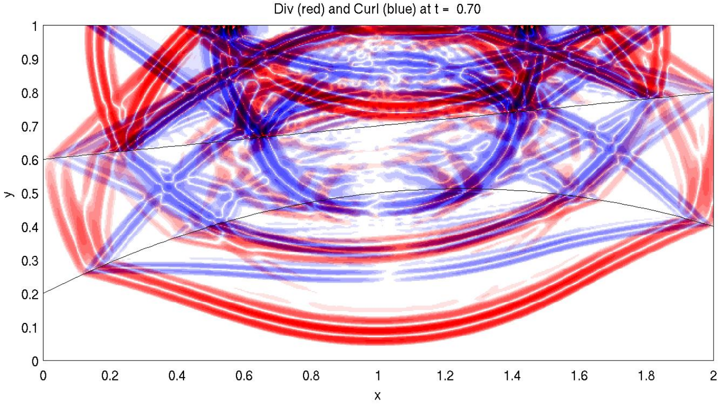
Seismic wave in layered medium

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



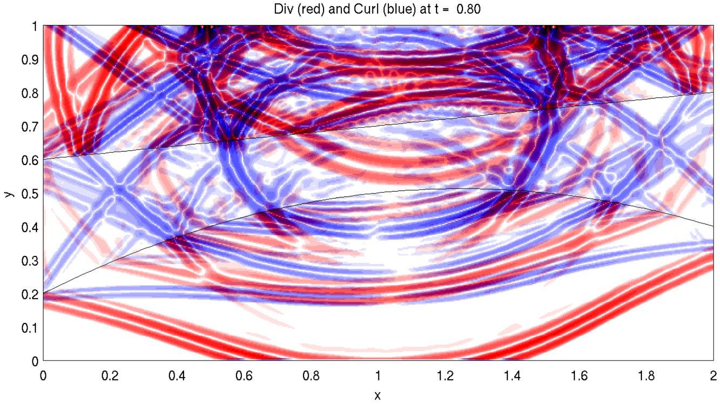
Seismic wave in layered medium

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



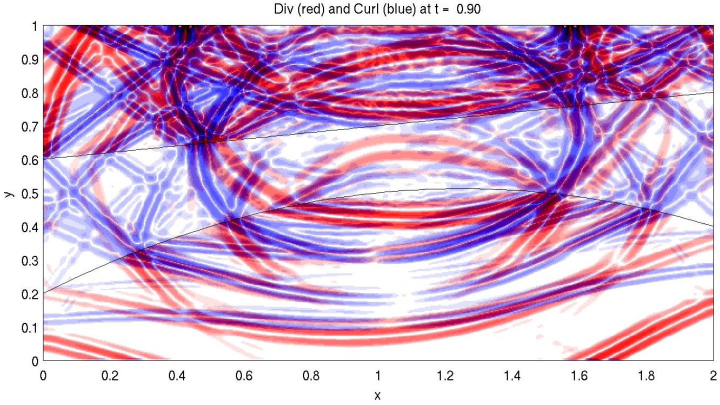
Seismic wave in layered medium

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



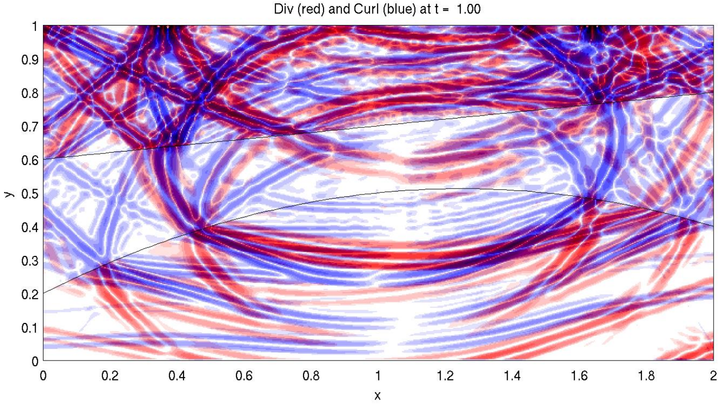
Seismic wave in layered medium

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]



Python plotting tools

Directory `_output` contains files `fort.t000N, fort.q000N` of data at frame N (N'th output time).

`fort.t000N`: Information about this time,
`fort.q000N`: Solution on all grids at this time

There may be many grids at each output time.

Python tools provide a way to specify what plots to produce for each frame:

- One or more figures,
- Each figure has one or more axes,
- Each axes has one or more items,
  (Curve, contour, pcolor, etc.)

`setplot` function for speciying plots

The file `setplot.py` contains a function setplot
Takes an object `plotdata` of class ClawPlotData,
Sets various attributes, and returns the object.

Documentation: **www.clawpack.org/users/setplot.html**

Example: 1 figure with 1 axes showing 1 item:

```
def setplot(plotdata):
    plotfigure = plotdata.new_plotfigure(name,num)
    plotaxes = plotfigure.new_plotaxes(title)
    plotitem = plotaxes.new_plotitem(plot_type)
    # set attributes of these objects
    return plotdata
```

`setplot` function for speciying plots

Example: plot first component of q as blue curve, red circles.

```
plotfigure = plotdata.new_plotfigure('Q', 1)
plotaxes = plotfigure.new_plotaxes('axes1')

plotitem = plotaxes.new_plotitem('1d_plot')
plotitem.plotvar = 0  # Python indexing!
plotitem.plotstyle = '-'
plotitem.color = 'b'  # or [0,0,1] or '#0000ff'

plotitem = plotaxes.new_plotitem('1d_plot')
# plotitem now points to a new object!
plotitem.plotvar = 0
plotitem.plotstyle = 'ro'
```

Plotting examples and documentation

General plotting information:
www.clawpack.org/users/plotting.html

Use of setplot, possible attributes:
www.clawpack.org/users/setplot.html

Examples:

1d: **www.clawpack.org/users/plotexamples.html**

2d: **www.clawpack.org/users/plotexamples2d.html**

FAQ: **www.clawpack.org/users/plotting_faq.html**

Gallery of applications:
www.clawpack.org/users/apps.html

Plotting options

Create a set of webpages showing all plots:

```
$ make .plots
```

Disadvantages:

- May take a while to plot all frames
- Can't zoom in dynamically or explore data

View plots interactively:

```
$ ipyclaw   # alias defined in setenv.bash

In[1]: ip = Iplotclaw()
In[2]: ip.plotloop()
PLOTCLAW>> ?

PLOTCLAW>> q
In[3]: Quit
```

First order hyperbolic PDE in 2 space dimensions

Advection equation: $q_t + uq_x + vq_y = 0$

First-order system: $q_t + Aq_x + Bq_y = 0$

where $q \in \mathbb{R}^m$ and $A, B \in \mathbb{R}^{m \times m}$.

Hyperbolic if $\cos(\theta)A + \sin(\theta)B$ is diagonalizable with real eigenvalues, for all angles θ.

First order hyperbolic PDE in 2 space dimensions

Advection equation: $q_t + uq_x + vq_y = 0$

First-order system: $q_t + Aq_x + Bq_y = 0$

where $q \in \mathbb{R}^m$ and $A, B \in \mathbb{R}^{m\times m}$.

Hyperbolic if $\cos(\theta)A + \sin(\theta)B$ is diagonalizable with real eigenvalues, for all angles θ.

This is required so that plane-wave data gives a 1d hyperbolic problem:

$$q(x, y, 0) = \breve{q}(x\cos\theta + y\sin\theta)$$

`(\breve q)`

implies contours of q in x–y plane are orthogonal to θ–direction.

Acoustics in 2 dimensions

$$
\begin{aligned}
p_t + K_0(u_x + v_y) &= 0 \\
\rho_0 u_t + p_x &= 0 \\
\rho_0 v_t + p_y &= 0
\end{aligned}
$$

Note: pressure responds to compression or expansion and so p_t is proportional to divergence of velocity.

Second and third equations are $F = ma$.

Acoustics in 2 dimensions

$$\begin{aligned} p_t + K_0(u_x + v_y) &= 0 \\ \rho_0 u_t + p_x &= 0 \\ \rho_0 v_t + p_y &= 0 \end{aligned}$$

Note: pressure responds to compression or expansion and so p_t is proportional to divergence of velocity.

Second and third equations are $F = ma$.

Gives hyperbolic system $q_t + Aq_x + Bq_y = 0$ with

$$q = \begin{bmatrix} p \\ u \\ v \end{bmatrix}, \qquad A = \begin{bmatrix} 0 & K_0 & 0 \\ 1/\rho_0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 & 0 & K_0 \\ 0 & 0 & 0 \\ 1/\rho_0 & 0 & 0 \end{bmatrix}.$$

Acoustics in 2 dimensions

$$q = \begin{bmatrix} p \\ u \\ v \end{bmatrix}, \qquad A = \begin{bmatrix} 0 & K_0 & 0 \\ 1/\rho_0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 & 0 & K_0 \\ 0 & 0 & 0 \\ 1/\rho_0 & 0 & 0 \end{bmatrix}.$$

Plane waves:

$$A\cos\theta + B\sin\theta = \begin{bmatrix} 0 & K_0\cos\theta & K_0\sin\theta \\ \cos\theta/\rho_0 & 0 & 0 \\ \sin\theta/\rho_0 & 0 & 0 \end{bmatrix}.$$

Acoustics in 2 dimensions

$$q = \begin{bmatrix} p \\ u \\ v \end{bmatrix}, \qquad A = \begin{bmatrix} 0 & K_0 & 0 \\ 1/\rho_0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 & 0 & K_0 \\ 0 & 0 & 0 \\ 1/\rho_0 & 0 & 0 \end{bmatrix}.$$

Plane waves:

$$A\cos\theta + B\sin\theta = \begin{bmatrix} 0 & K_0\cos\theta & K_0\sin\theta \\ \cos\theta/\rho_0 & 0 & 0 \\ \sin\theta/\rho_0 & 0 & 0 \end{bmatrix}.$$

Eigenvalues: $\lambda^1 = -c_0, \;\; \lambda^2 = 0, \;\; \lambda^3 = +c_0 \;\; = \sqrt{K_0/\rho_0}$

Independent of angle θ.

Isotropic: sound propagates at same speed in any direction.

Acoustics in 2 dimensions

$$q = \begin{bmatrix} p \\ u \\ v \end{bmatrix}, \qquad A = \begin{bmatrix} 0 & K_0 & 0 \\ 1/\rho_0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad B = \begin{bmatrix} 0 & 0 & K_0 \\ 0 & 0 & 0 \\ 1/\rho_0 & 0 & 0 \end{bmatrix}.$$

Plane waves:

$$A\cos\theta + B\sin\theta = \begin{bmatrix} 0 & K_0\cos\theta & K_0\sin\theta \\ \cos\theta/\rho_0 & 0 & 0 \\ \sin\theta/\rho_0 & 0 & 0 \end{bmatrix}.$$

Eigenvalues: $\lambda^1 = -c_0, \;\; \lambda^2 = 0, \;\; \lambda^3 = +c_0 \;\; = \sqrt{K_0/\rho_0}$

Independent of angle θ.

Isotropic: sound propagates at same speed in any direction.

Note: Zero wave speed for “shear wave” with variation only in velocity in direction $(-\sin\theta,\; \cos\theta)$. (Fig 18.1)

Acoustics in 2 dimensions

$$
\begin{aligned}
p_t + K_0(u_x + v_y) &= 0 \\
\rho_0 u_t + p_x &= 0 \\
\rho_0 v_t + p_y &= 0
\end{aligned}
$$

$$
A = \begin{bmatrix} 0 & K_0 & 0 \\ 1/\rho_0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad R^x = \begin{bmatrix} -Z_0 & 0 & Z_0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}
$$

Solving $q_t + Aq_x = 0$ gives pressure waves in (p, u).
x-variations in v are stationary.

Acoustics in 2 dimensions

$$
\begin{aligned}
p_t + K_0(u_x + v_y) &= 0 \\
\rho_0 u_t + p_x &= 0 \\
\rho_0 v_t + p_y &= 0
\end{aligned}
$$

$$
A = \begin{bmatrix} 0 & K_0 & 0 \\ 1/\rho_0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad R^x = \begin{bmatrix} -Z_0 & 0 & Z_0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}
$$

Solving $q_t + Aq_x = 0$ gives pressure waves in (p, u).
x-variations in v are stationary.

$$
B = \begin{bmatrix} 0 & 0 & K_0 \\ 0 & 0 & 0 \\ 1/\rho_0 & 0 & 0 \end{bmatrix} \qquad R^y = \begin{bmatrix} -Z_0 & 0 & Z_0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}
$$

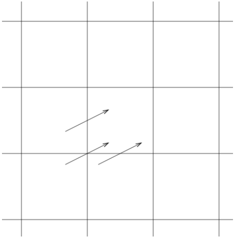
Solving $q_t + Bq_y = 0$ gives pressure waves in (p, v).
y-variations in u are stationary.

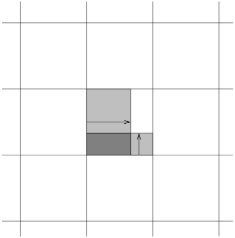
Advection: Donor Cell Upwind

With no correction fluxes, Godunov's method for advection is

Donor Cell Upwind:

$$\begin{aligned} Q_{ij}^{n+1} = Q_{ij} &- \frac{\Delta t}{\Delta x}[u^+(Q_{ij} - Q_{i-1,j}) + u^-(Q_{i+1,j} - Q_{ij})] \\ &- \frac{\Delta t}{\Delta y}[v^+(Q_{ij} - Q_{i,j-1}) + v^-(Q_{i,j+1} - Q_{ij})]. \end{aligned}$$



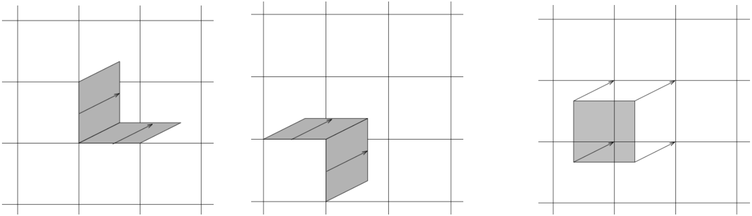


Stable only if $\left|\frac{u\Delta t}{\Delta x}\right| + \left|\frac{v\Delta t}{\Delta y}\right| \le 1.$

Advection: Corner Transport Upwind (CTU)

Correction fluxes can be added to advect waves correctly.

Corner Transport Upwind:



Stable for $\max\left(\left|\frac{u\Delta t}{\Delta x}\right|, \left|\frac{v\Delta t}{\Delta y}\right|\right) \leq 1.$

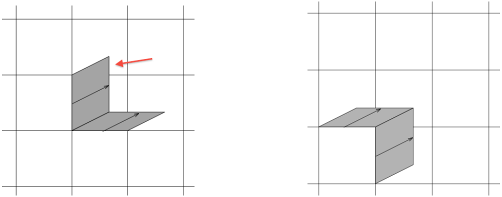
Advection: Corner Transport Upwind (CTU)

Need to transport triangular region from cell (i, j) to $(i, j+1)$:

$$\text{Area } = \frac{1}{2}(u\Delta t)(v\Delta t) \implies \left(\frac{\frac{1}{2}uv(\Delta t)^2}{\Delta x \Delta y}\right)(Q_{ij} - Q_{i-1,j}).$$

Accomplished by correction flux:

$$\tilde{G}_{i,j+1/2} = -\frac{1}{2}\frac{\Delta t}{\Delta x}\, uv(Q_{ij} - Q_{i-1,j})$$



$\frac{\Delta t}{\Delta y}(\tilde{G}_{i,j+1/2} - \tilde{G}_{i,j-1/2})$ gives approximation to $\frac{1}{2}\Delta t^2 uvq_{xy}$.

$\frac{\Delta t}{\Delta x}(\tilde{F}_{i+1/2,j} - \tilde{F}_{i-1/2,j})$ gives similar approximation.

Wave propagation algorithms in 2D

Clawpack requires:

Normal Riemann solver `rpn2.f`
Solves 1d Riemann problem $q_t + Aq_x = 0$
Decomposes $\Delta Q = Q_{ij} - Q_{i-1,j}$ into $\mathcal{A}^+\Delta Q$ and $\mathcal{A}^-\Delta Q$.
For $q_t + Aq_x + Bq_y = 0$, split using eigenvalues, vectors:

$$A = R\Lambda R^{-1} \implies A^- = R\Lambda^- R^{-1}, A^+ = R\Lambda^+ R^{-1}$$

Input parameter `ixy` determines if it's in x or y direction.
In latter case splitting is done using B instead of A.
This is all that's required for dimensional splitting.

Wave propagation algorithms in 2D

Clawpack requires:

Normal Riemann solver `rpn2.f`
Solves 1d Riemann problem $q_t + Aq_x = 0$
Decomposes $\Delta Q = Q_{ij} - Q_{i-1,j}$ into $\mathcal{A}^+\Delta Q$ and $\mathcal{A}^-\Delta Q$.
For $q_t + Aq_x + Bq_y = 0$, split using eigenvalues, vectors:

$$A = R\Lambda R^{-1} \implies A^- = R\Lambda^- R^{-1}, A^+ = R\Lambda^+ R^{-1}$$

Input parameter `ixy` determines if it's in x or y direction.
In latter case splitting is done using B instead of A.
This is all that's required for dimensional splitting.

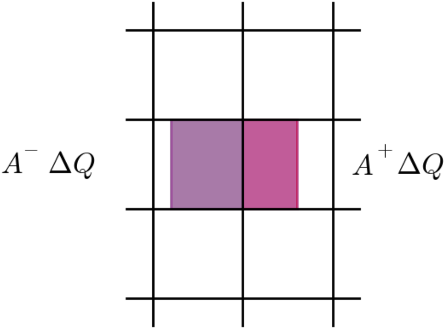
Transverse Riemann solver `rpt2.f`
Decomposes $\mathcal{A}^+\Delta Q$ into $\mathcal{B}^-\mathcal{A}^+\Delta Q$ and $\mathcal{B}^+\mathcal{A}^+\Delta Q$ by splitting this vector into eigenvectors of B.

(Or splits vector into eigenvectors of A if `ixy=2`.)

Wave propagation algorithm for $q_t + Aq_x + Bq_y = 0$

Decompose $A = A^+ + A^-$ and $B = B^+ + B^-$.

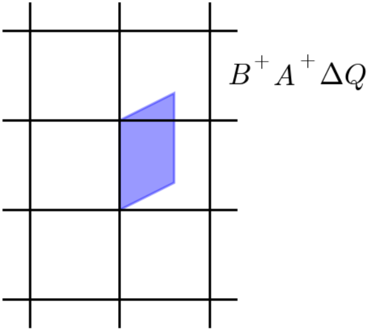
For $\Delta Q = Q_{ij} - Q_{i-1,j}$:



Wave propagation algorithm for $q_t + Aq_x + Bq_y = 0$

Decompose $A = A^+ + A^-$ and $B = B^+ + B^-$.

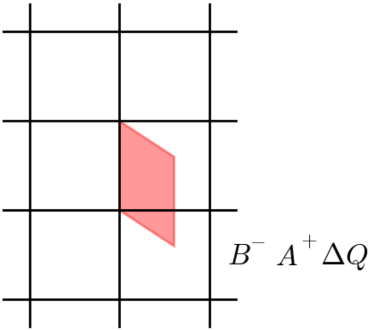
For $\Delta Q = Q_{ij} - Q_{i-1,j}$:



Wave propagation algorithm for $q_t + Aq_x + Bq_y = 0$

Decompose $A = A^+ + A^-$ and $B = B^+ + B^-$.

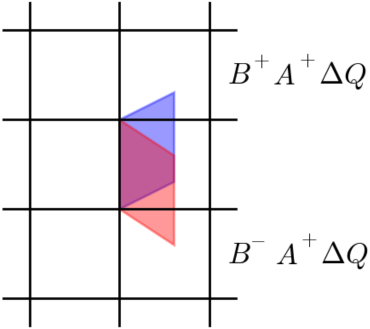
For $\Delta Q = Q_{ij} - Q_{i-1,j}$:



Wave propagation algorithm for $q_t + Aq_x + Bq_y = 0$

Decompose $A = A^+ + A^-$ and $B = B^+ + B^-$.

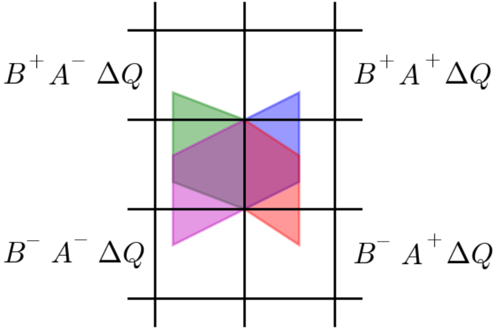
For $\Delta Q = Q_{ij} - Q_{i-1,j}$:



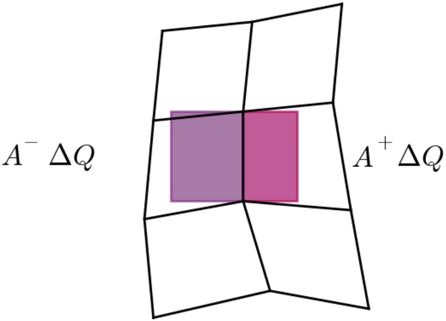
Wave propagation algorithm for $q_t + Aq_x + Bq_y = 0$

Decompose $A = A^+ + A^-$ and $B = B^+ + B^-$.

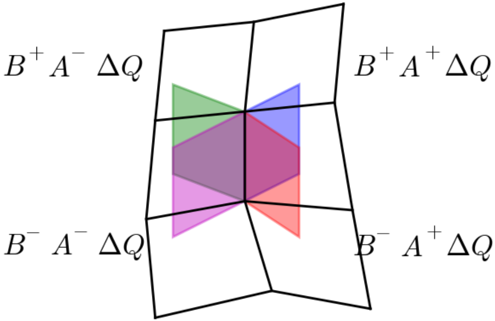
For $\Delta Q = Q_{ij} - Q_{i-1,j}$:



Wave propagation algorithm on a quadrilateral grid



Wave propagation algorithm on a quadrilateral grid



Acoustics in heterogeneous media

$$q_t + A(x,y)q_x + B(x,y)q_y = 0, \qquad q = (p,\ u,\ v)^T,$$

where

$$A = \begin{bmatrix} 0 & K(x,y) & 0 \\ 1/\rho(x,y) & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & K(x,y) \\ 0 & 0 & 0 \\ 1/\rho(x,y) & 0 & 0 \end{bmatrix}.$$

Note: Not in conservation form!

Acoustics in heterogeneous media

$$q_t + A(x,y)q_x + B(x,y)q_y = 0, \qquad q = (p,\ u,\ v)^T,$$

where

$$A = \begin{bmatrix} 0 & K(x,y) & 0 \\ 1/\rho(x,y) & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & K(x,y) \\ 0 & 0 & 0 \\ 1/\rho(x,y) & 0 & 0 \end{bmatrix}.$$

Note: Not in conservation form!

Wave propagation still makes sense. In x-direction:

$$\mathcal{W}^1 = \alpha^1 \begin{bmatrix} -Z_{i-1,j} \\ 1 \\ 0 \end{bmatrix}, \qquad \mathcal{W}^2 = \alpha^2 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad \mathcal{W}^3 = \alpha^3 \begin{bmatrix} Z_{ij} \\ 1 \\ 0 \end{bmatrix}.$$

Wave speeds: $s^1_{i-1/2,j} = -c_{i-1,j}, \;\; s^2_{i-1/2,j} = 0, \;\; s^3_{i-1/2,j} = +c_{ij}$.

Acoustics in heterogeneous media

$$\mathcal{W}^1 = \alpha^1 \begin{bmatrix} -Z_{i-1,j} \\ 1 \\ 0 \end{bmatrix}, \qquad \mathcal{W}^2 = \alpha^2 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad \mathcal{W}^3 = \alpha^3 \begin{bmatrix} Z_{ij} \\ 1 \\ 0 \end{bmatrix}.$$

Decompose $\Delta Q = (\Delta p,\ \ \Delta u,\ \ \Delta v)^T$:

$$\begin{aligned} \alpha^1_{i-1/2,j} &= (-\Delta Q^1 + Z\Delta Q^2)/(Z_{i-1,j} + Z_{ij}), \\ \alpha^2_{i-1/2,j} &= \Delta Q^3, \\ \alpha^3_{i-1/2,j} &= (\Delta Q^1 + Z_{i-1,j}\Delta Q^2)/(Z_{i-1,j} + Z_{ij}). \end{aligned}$$

Acoustics in heterogeneous media

$$\mathcal{W}^1 = \alpha^1 \begin{bmatrix} -Z_{i-1,j} \\ 1 \\ 0 \end{bmatrix}, \qquad \mathcal{W}^2 = \alpha^2 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad \mathcal{W}^3 = \alpha^3 \begin{bmatrix} Z_{ij} \\ 1 \\ 0 \end{bmatrix}.$$

Decompose $\Delta Q = (\Delta p, \ \Delta u, \ \Delta v)^T$:

$$\begin{aligned} \alpha^1_{i-1/2,j} &= (-\Delta Q^1 + Z\Delta Q^2)/(Z_{i-1,j} + Z_{ij}), \\ \alpha^2_{i-1/2,j} &= \Delta Q^3, \\ \alpha^3_{i-1/2,j} &= (\Delta Q^1 + Z_{i-1,j}\Delta Q^2)/(Z_{i-1,j} + Z_{ij}). \end{aligned}$$

Fluctuations: (Note: $s^1 < 0, \ s^2 = 0, \ s^3 > 0$)

$$\mathcal{A}^-\Delta Q_{i-1/2,j} = s^1_{i-1/2,j}\mathcal{W}^1_{i-1/2,j},$$

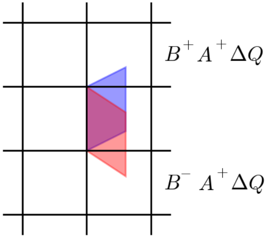
$$\mathcal{A}^+\Delta Q_{i-1/2,j} = s^3_{i-1/2,j}\mathcal{W}^3_{i-1/2,j}.$$

Acoustics in heterogeneous media

Transverse solver: Split right-going fluctuation

$$\mathcal{A}^+\Delta Q_{i-1/2,j} = s^3_{i-1/2,j}\mathcal{W}^3_{i-1/2,j}$$

into up-going and down-going pieces:



Decompose $\mathcal{A}^+\Delta Q_{i-1/2,j}$ into eigenvectors of B. Down-going:

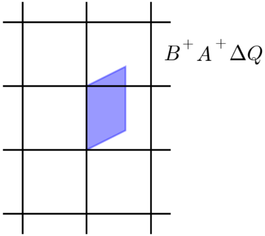
$$\mathcal{A}^+\Delta Q_{i-1/2,j} = \beta^1 \begin{bmatrix} -Z_{i,j-1} \\ 0 \\ 1 \end{bmatrix} + \beta^2 \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} + \beta^3 \begin{bmatrix} Z_{ij} \\ 0 \\ 1 \end{bmatrix},$$

Transverse solver for acoustics

Up-going part: $\mathcal{B}^+\mathcal{A}^+\Delta Q_{i-1/2,j} = c_{i,j+1}\beta^3 r^3$ from

$$\mathcal{A}^+\Delta Q_{i-1/2,j} = \beta^1 \begin{bmatrix} -Z_{ij} \\ 0 \\ 1 \end{bmatrix} + \beta^2 \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} + \beta^3 \begin{bmatrix} Z_{i,j+1} \\ 0 \\ 1 \end{bmatrix},$$

$$\beta^3 = \left((\mathcal{A}^+\Delta Q_{i-1/2,j})^1 + (\mathcal{A}^+\Delta Q_{i-1/2,j})^3 Z_{i,j+1}\right) / (Z_{ij} + Z_{i,j+1}).$$



Transverse Riemann solver in Clawpack

`rpt2` takes vector `asdq` and returns `bmasdq` and `bpasdq` where

`asdq` $= \mathcal{A}^*\Delta Q$ represents either
  $\mathcal{A}^-\Delta Q$ if `imp = 1`, or
  $\mathcal{A}^+\Delta Q$ if `imp = 2`.

Returns $\mathcal{B}^-\mathcal{A}^*\Delta Q$ and $\mathcal{B}^+\mathcal{A}^*\Delta Q$.

Transverse Riemann solver in Clawpack

`rpt2` takes vector `asdq` and returns `bmasdq` and `bpasdq` where

`asdq` $= \mathcal{A}^*\Delta Q$ represents either
$\mathcal{A}^-\Delta Q$ if `imp = 1`, or
$\mathcal{A}^+\Delta Q$ if `imp = 2`.

Returns $\mathcal{B}^-\mathcal{A}^*\Delta Q$ and $\mathcal{B}^+\mathcal{A}^*\Delta Q$.

Note: there is also a parameter `ixy`:

`ixy = 1` means normal solve was in x-direction,

`ixy = 2` means normal solve was in y-direction,
In this case `asdq` represents $\mathcal{B}^-\Delta Q$ or $\mathcal{B}^+\Delta Q$ and the routine must return $\mathcal{A}^-\mathcal{B}^*\Delta Q$ and $\mathcal{A}^+\mathcal{B}^*\Delta Q$.