GeoClaw Tutorial

www.geoclaw.org

Randy LeVeque

University of Washington

Get started at:

`https://github.com/clawpack/geoclaw_tutorial_csdms2019`

`https://tinyurl.com/y3g2adcp`

html version of notebooks:

`http://depts.washington.edu/clawpack/geoclaw/tutorial_csdms2019/`

GeoClaw Software

Based on:

Clawpack (Conservation Laws Package), since 1994,
Dave George's thesis work TsunamiClaw, since 2004.

Some features:

- 2d library for depth-averaged flows over topography.
- Handles dry cells where depth $= 0$.
- Tools for dealing with multiple data sets at different resolutions.

- Shock-capturing finite volume methods
- Adaptive mesh refinement (AMR)
- OpenMP for running on shared memory machines
- Fortran with Python interface / plotting

Clawpack Development

On GitHub: `https://github.com/clawpack`

- Pull requests,
- Travis continuous integration,
- Issue tracker

Some of the core developers:

- Randy LeVeque (UW)
- Marsha Berger (NYU)
- Kyle Mandli (Columbia)
- David Ketcheson (KAUST)

Some funding sources:

NSF, NCAR, DOE, AFOSR, ONR, KAUST, UW, NTHMP, FEMA, NASA

Clawpack Development

Open source development on Github:

https://github.com/clawpack

PeerJ CompSci

PEER-REVIEWED

Clawpack: building an open source ecosystem for solving hyperbolic PDEs

Distributed and Parallel Computing | Scientific Computing and Simulation

Kyle T. Mandli[1], Aron J. Ahmadia[2], Marsha Berger[3], Donna Calhoun[4], David L. George[5], Yiannis Hadjimichael[6], David I. Ketcheson[6], Grady I. Lemoine[7], Randall J. LeVeque[8]

August 8, 2016

10.7717/peerj-cs.68

Depth-averaged models

Reduce three-dimensional free surface problem to...

Two-dimensional Shallow Water (St. Venant) Equations

$$h_t + (hu)_x + (hv)_y = 0$$

$$(hu)_t + \left(hu^2 + \frac{1}{2}gh^2\right)_x + (huv)_y = -ghB_x(x,y)$$

$$(hv)_t + (huv)_x + \left(hv^2 + \frac{1}{2}gh^2\right)_y = -ghB_y(x,y)$$

where (u,v) are velocities in the horizontal directions (x,y),

$B(x,y) =$ bathymetry (underwater topography),

$\frac{1}{2}gh^2 =$ hydrostatic pressure.

A few applications

See **www.geoclaw.org** for more references.

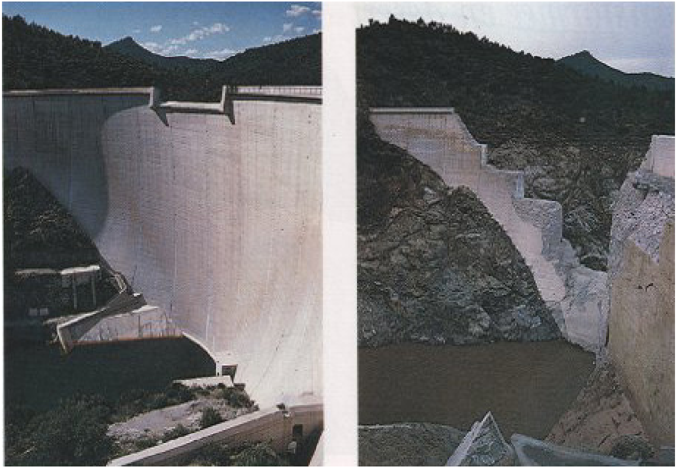
- Dam break problems, to illustrate AMR / wetting
- Tsunami modeling, see also

  `http://depts.washington.edu/ptha/projects`
- Storm surge, see e.g.

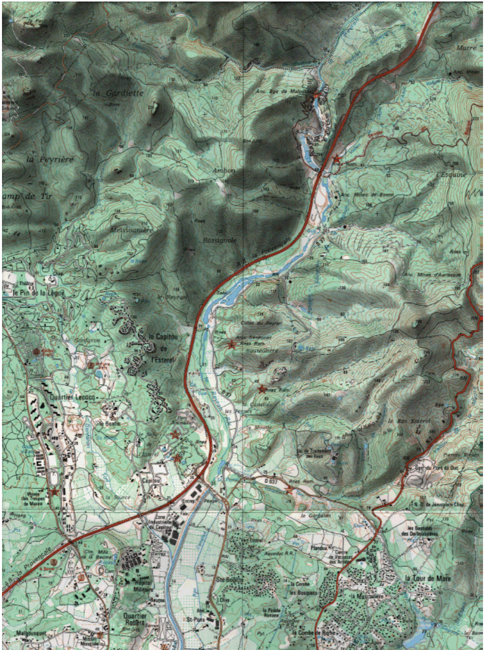
  Adaptive Mesh Refinement for Storm Surge
  by K.T. Mandli and C. Dawson, *Ocean Modeling* 2014,
  doi:10.1016/j.ocemod.2014.01.002

Malpasset Dam Failure

Catastrophic failure in 1959

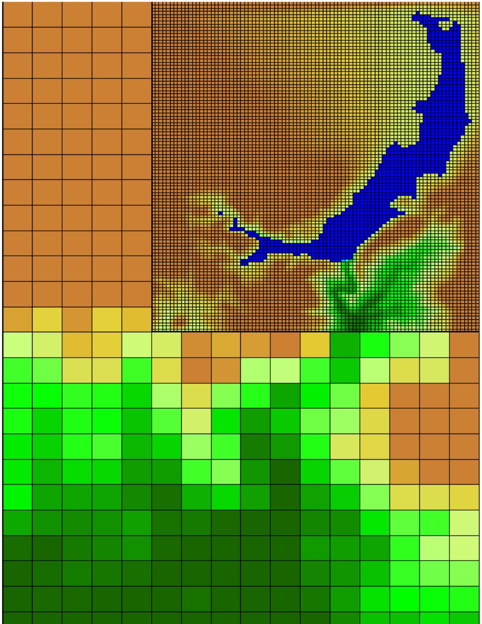


Malpasset Dam Failure



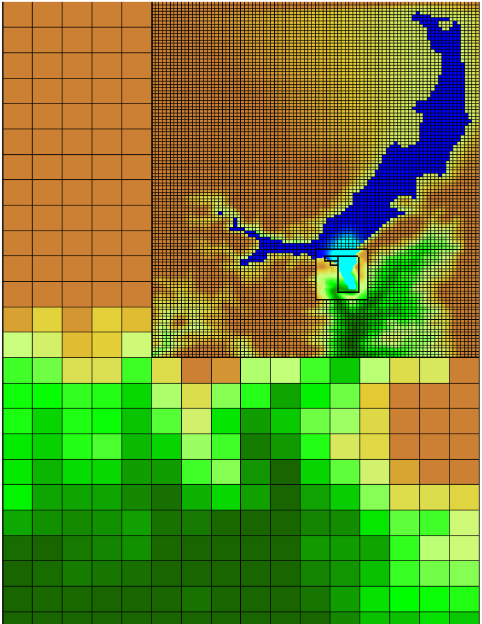
D.L. George, *Numer. Meth. Fluids* 2011, **doi:10.1002/fld.2298**

Coarse: 400m cell side,   Level 2: 50m,   Level 3: 12m,   Level 4: 3m



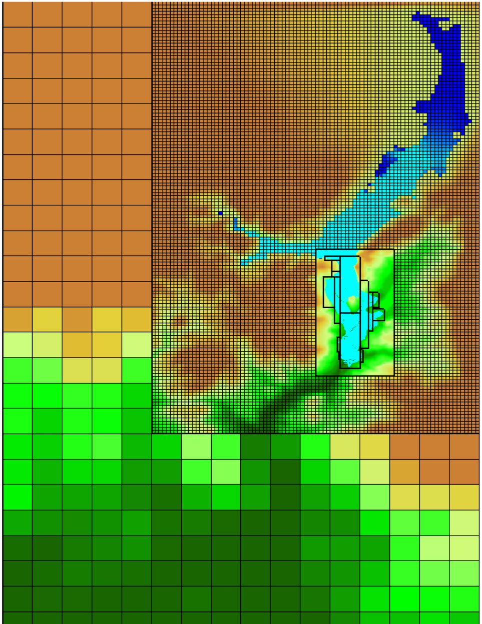
D.L. George, *Numer. Meth. Fluids* 2011, doi:10.1002/fld.2298

Coarse: 400m cell side, Level 2: 50m, Level 3: 12m, Level 4: 3m



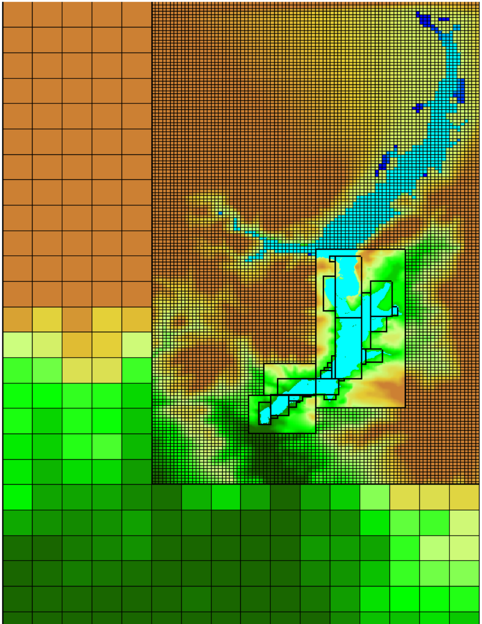
D.L. George, *Numer. Meth. Fluids* 2011, **doi:10.1002/fld.2298**

Coarse: 400m cell side, Level 2: 50m, Level 3: 12m, Level 4: 3m



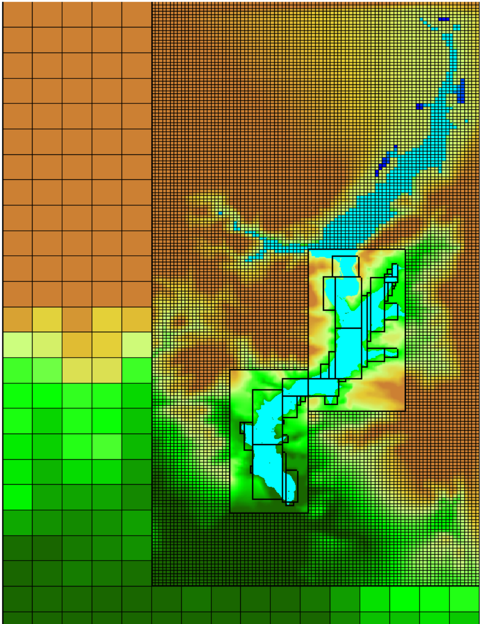
D.L. George, *Numer. Meth. Fluids* 2011, **doi:10.1002/fld.2298**

Coarse: 400m cell side, Level 2: 50m, Level 3: 12m, Level 4: 3m



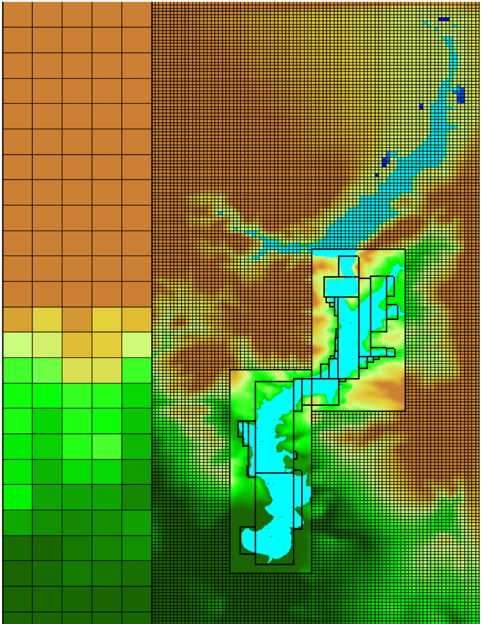
D.L. George, *Numer. Meth. Fluids* 2011, **doi:10.1002/fld.2298**

Coarse: 400m cell side, Level 2: 50m, Level 3: 12m, Level 4: 3m



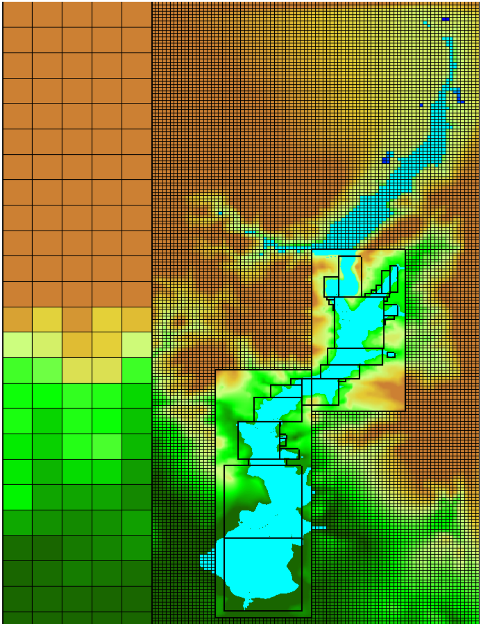
D.L. George, *Numer. Meth. Fluids* 2011, **doi:10.1002/fld.2298**

Coarse: 400m cell side,   Level 2: 50m,   Level 3: 12m,   Level 4: 3m



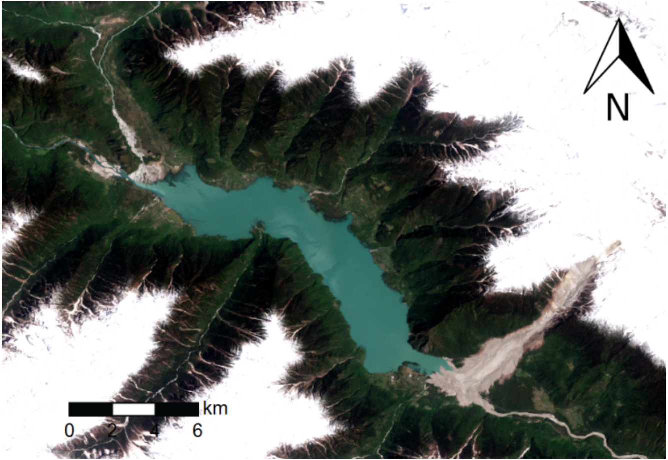
D.L. George, *Numer. Meth. Fluids* 2011, **doi:10.1002/fld.2298**

Coarse: 400m cell side, Level 2: 50m, Level 3: 12m, Level 4: 3m



Yigong River landslide-dam outburst flood:

- River impounded by rockslide (115 Mm^3) for 62 days, resulting in lake ~2.0 km^3 in volume

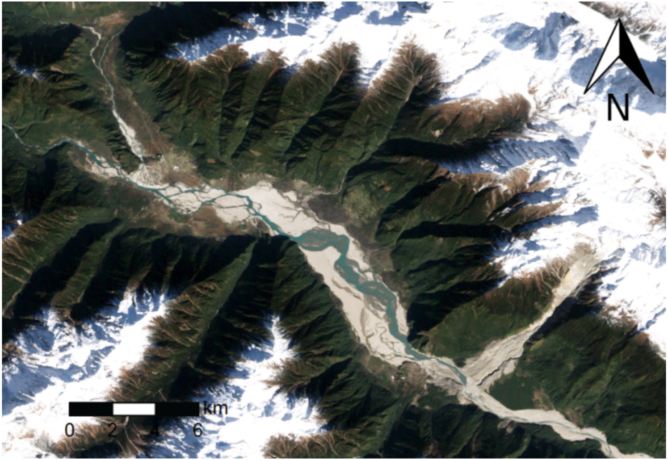


Landsat-7 imagery (May 4, 2000)

M.D. Turzewski, K.W. Huntington, RJL
JGR Earth Surface, 2019, **doi:10.1029/2018JF004778**

Yigong River landslide-dam outburst flood:

- Dam failed catastrophically causing outburst flood with discharge >1.2×10^5 m^3/s



Landsat-7 imagery (Nov. 15, 2001)

M.D. Turzewski, K.W. Huntington, RJL
JGR Earth Surface, 2019, **doi:10.1029/2018JF004778**

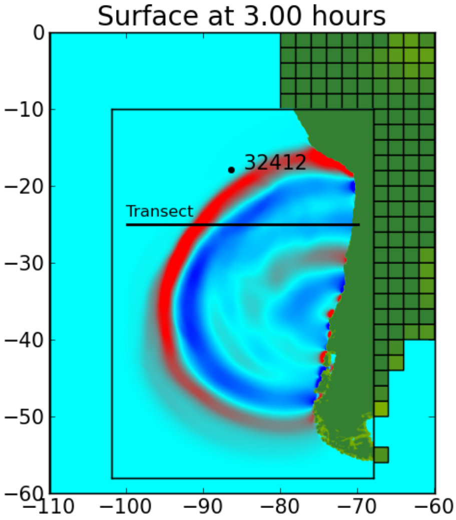
Shallow water equations with bathymetry $B(x, y)$

$$
\begin{aligned}
h_t + (hu)_x + (hv)_y &= 0 \\
(hu)_t + \left(hu^2 + \frac{1}{2}gh^2\right)_x + (huv)_y &= -ghB_x(x, y) \\
(hv)_t + (huv)_x + \left(hv^2 + \frac{1}{2}gh^2\right)_y &= -ghB_y(x, y)
\end{aligned}
$$

Some issues:

- Delicate balance between flux divergence and bathymetry:
  h varies on order of 4000m, rapid variations in ocean
  Waves have magnitude 1m or less.
- Cartesian grid used, with $h = 0$ in dry cells:
  Cells become wet/dry as wave advances on shore
  Robust Riemann solvers needed.
- Adaptive mesh refinement crucial
  Interaction of AMR with source terms, dry states

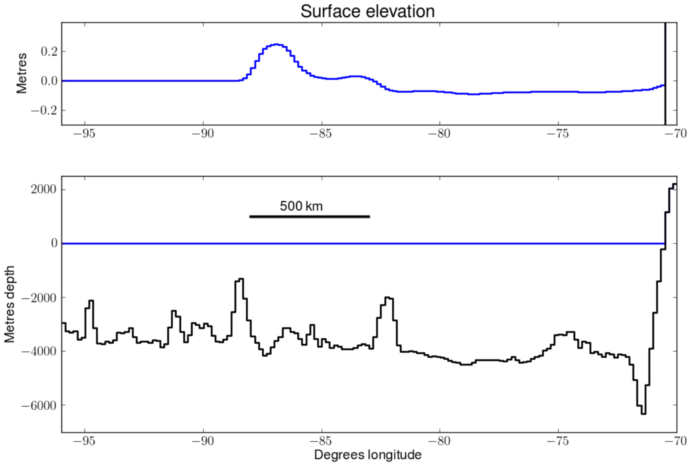
Tsunami from 27 Feb 2010 quake off Chile



Transect of 27 February 2010 tsunami

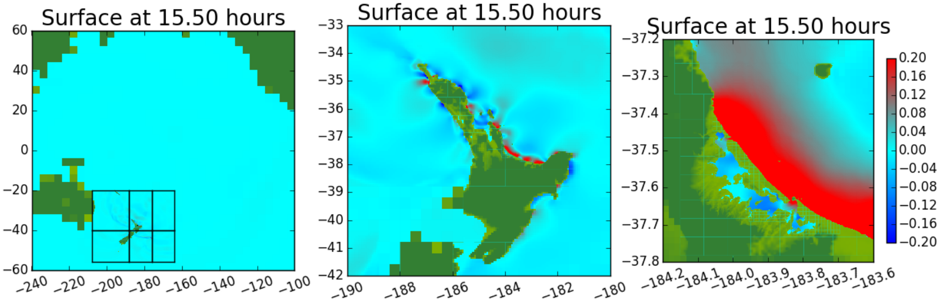
Bathymetry, depth change by > 1000 m from one cell to next,

Surface elevation changes on order of a few cm.

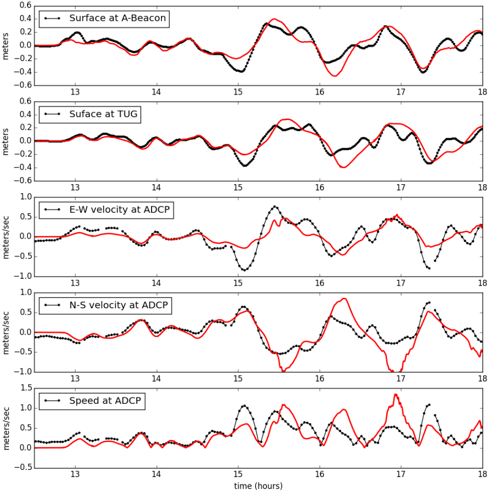


Tohoku to Tauranga Harbor, NZ with AMR

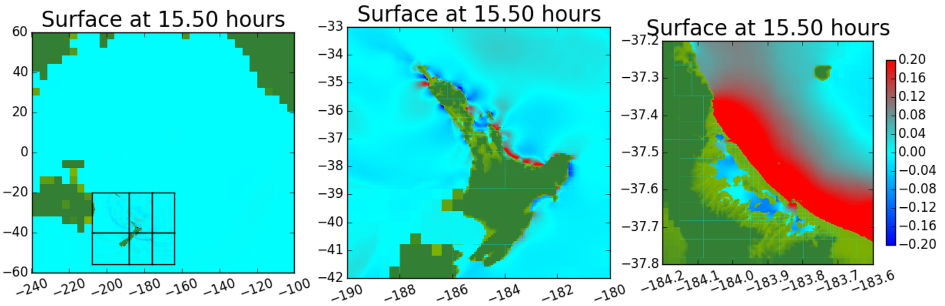
GeoClaw software uses adaptive mesh refinement (AMR) to focus finer grid resolution only where needed.



Tauranga Harbor gauges

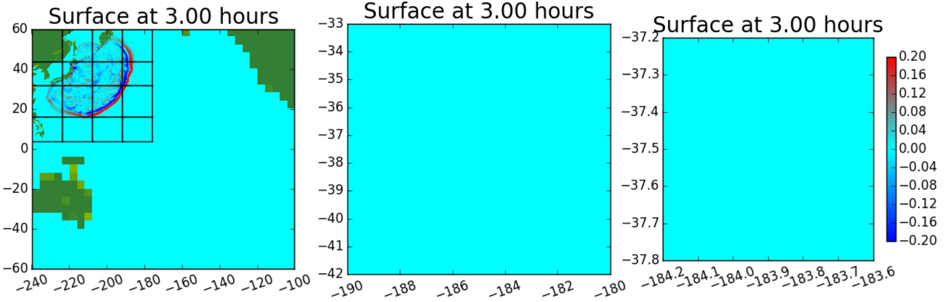


Tohoku to Tauranga Harbor, NZ with AMR



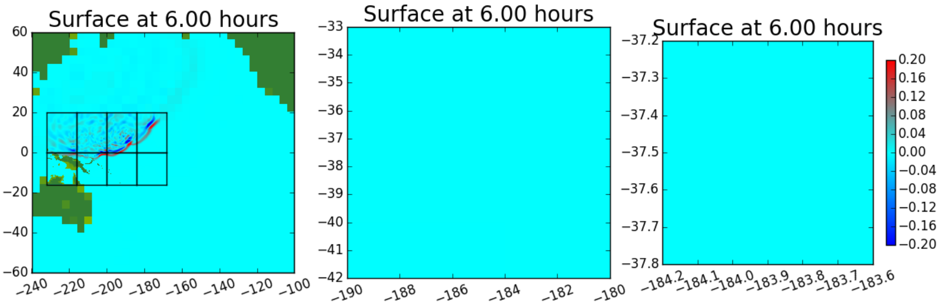
Elapsed time on quad-core MacBook: 3.5 hours

Tohoku to Tauranga Harbor, NZ with AMR



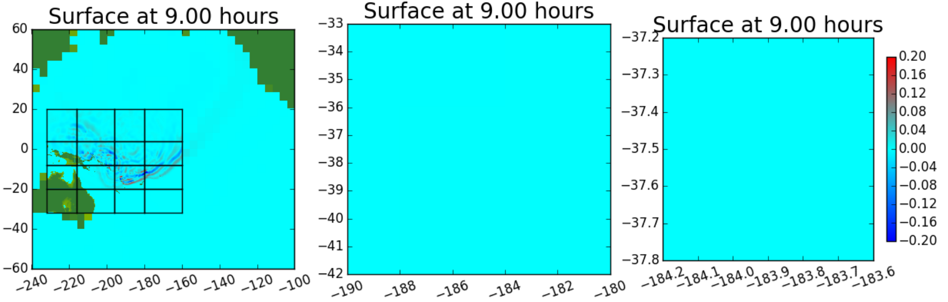
Elapsed time on quad-core MacBook: < 1 minute

Tohoku to Tauranga Harbor, NZ with AMR



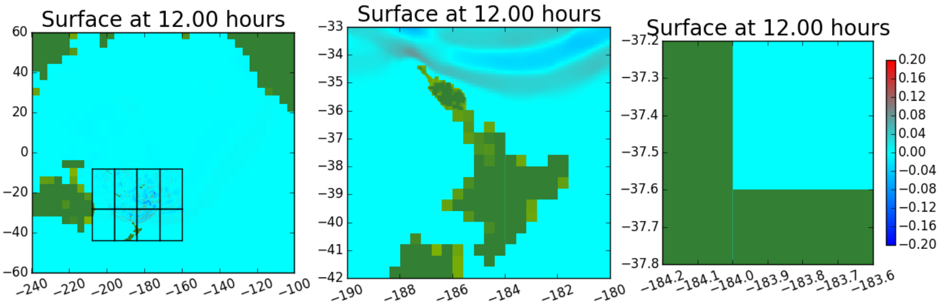
Elapsed time on quad-core MacBook: < 2 minutes

Tohoku to Tauranga Harbor, NZ with AMR



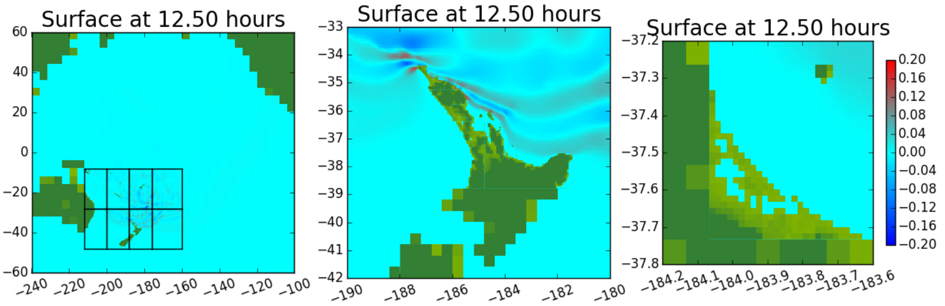
Elapsed time on quad-core MacBook: 3 minutes

Tohoku to Tauranga Harbor, NZ with AMR



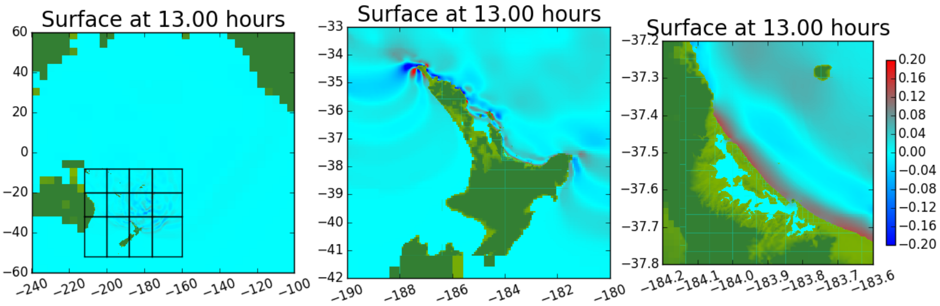
Elapsed time on quad-core MacBook: 5 minutes

Tohoku to Tauranga Harbor, NZ with AMR



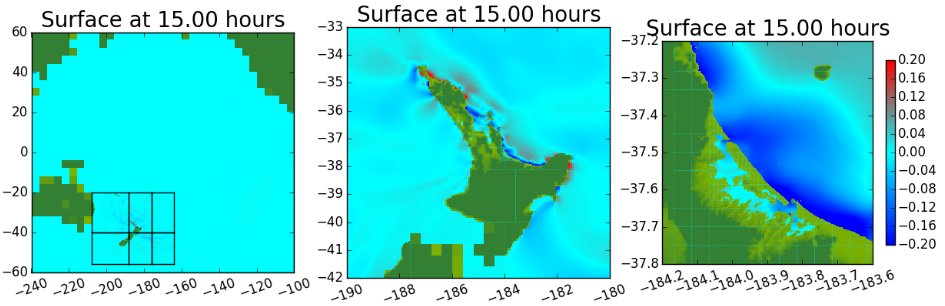
Elapsed time on quad-core MacBook: 6 minutes

Tohoku to Tauranga Harbor, NZ with AMR



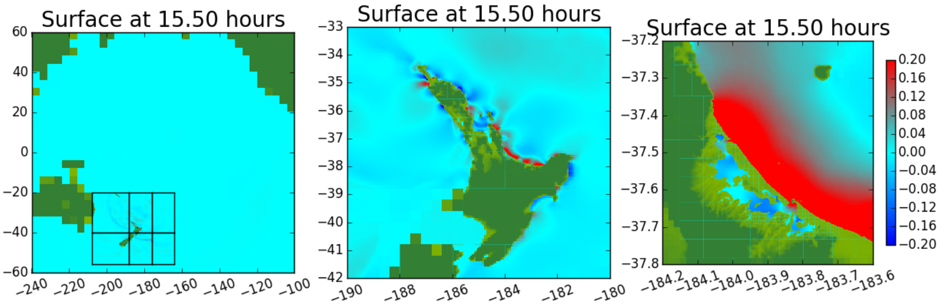
Elapsed time on quad-core MacBook: 19 minutes

Tohoku to Tauranga Harbor, NZ with AMR



Elapsed time on quad-core MacBook: 3 hours

Tohoku to Tauranga Harbor, NZ with AMR



Elapsed time on quad-core MacBook: 3.5 hours

Clawpack installation and overview

Can install with `pip`, see:

`http://www.clawpack.org/installing.html`

Downloads and sets paths,
Pre-compiles Riemann solvers for PyClaw.

Or can use `git clone`, Docker, etc.

Clawpack installation and overview

Can install with `pip`, see:

http://www.clawpack.org/installing.html

Downloads and sets paths,
Pre-compiles Riemann solvers for PyClaw.

Or can use `git clone`, Docker, etc.

Main directory contains subrepositories:

`pyclaw, classic, riemann, clawutil, visclaw`

Clawpack installation and overview

Can install with `pip`, see:

http://www.clawpack.org/installing.html

Downloads and sets paths,
Pre-compiles Riemann solvers for PyClaw.

Or can use `git clone`, Docker, etc.

Main directory contains subrepositories:

`pyclaw, classic, riemann, clawutil, visclaw`

`amrclaw, geoclaw`

Clawpack installation and overview

Can install with `pip`, see:

http://www.clawpack.org/installing.html

Downloads and sets paths,
Pre-compiles Riemann solvers for PyClaw.

Or can use `git clone`, Docker, etc.

Main directory contains subrepositories:

`pyclaw, classic, riemann, clawutil, visclaw`

`amrclaw, geoclaw`

`doc, clawpack.github.com`

Clawpack installation and overview

Can install with `pip`, see:

http://www.clawpack.org/installing.html

Downloads and sets paths,
Pre-compiles Riemann solvers for PyClaw.

Or can use `git clone`, Docker, etc.

Main directory contains subrepositories:

`pyclaw, classic, riemann, clawutil, visclaw`

`amrclaw, geoclaw`

`doc, clawpack.github.com`

`apps` — `notebooks, tsunami, surge-examples,`

GeoClaw organization

http://www.clawpack.org/contents.html#geoclaw-geophysical-flows

`clawpack/geoclaw` directory contains:

- `src/2d/shallow` — Fortran source code for 2D SWE

GeoClaw organization

http://www.clawpack.org/contents.html#geoclaw-geophysical-flows

`clawpack/geoclaw` directory contains:

- `src/2d/shallow` — Fortran source code for 2D SWE
- `src/python/geoclaw` — Python tools, e.g.
  `topotools.py, dtopotools.py`

GeoClaw organization

http://www.clawpack.org/contents.html#geoclaw-geophysical-flows

`clawpack/geoclaw` directory contains:

- `src/2d/shallow` — Fortran source code for 2D SWE
- `src/python/geoclaw` — Python tools, e.g.
  `topotools.py, dtopotools.py`
- `tests` — unit tests run by `travis-ci`.

GeoClaw organization

http://www.clawpack.org/contents.html#geoclaw-geophysical-flows

`clawpack/geoclaw` directory contains:

- `src/2d/shallow` — Fortran source code for 2D SWE
- `src/python/geoclaw` — Python tools, e.g.
  `topotools.py, dtopotools.py`
- `tests` — unit tests run by `travis-ci`.
- `examples` — some examples, e.g.
  `examples/tsunami/chile2010` — basis for tutorial

GeoClaw organization

`http://www.clawpack.org/contents.html#geoclaw-geophysical-flows`

`clawpack/geoclaw` directory contains:

- `src/2d/shallow` — Fortran source code for 2D SWE
- `src/python/geoclaw` — Python tools, e.g.
  `topotools.py, dtopotools.py`
- `tests` — unit tests run by `travis-ci`.
- `examples` — some examples, e.g.
  `examples/tsunami/chile2010` — basis for tutorial

  Note that `maketopo.py` *currently broken in v5.5.0!*

Compiling, running, plotting

Applications directories contain a Makefile.

`$ make help` For list of options

`$ make .data` Uses `setrun.py` to make Fortran data

`$ make .exe` Compiles Fortran codes

`$ make .output` Runs code, produces `_output`

`$ make .plots` Plots results, produces `_plots`

`$ make .htmls` Produces html versions of source files

Setting runtime parameters

The file `setrun.py` contains a function setrun
that returns an object rundata of class ClawRunData.

`make .data` rewrites data in `*.data` files read in by Fortran.

Never need to write from scratch...
Modify an existing example!

Don’t need to know much if anything about Python!

Lots of comments in the sample versions.

Documentation: **www.clawpack.org/setrun_geoclaw.html**

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 (ASCII)
> `fort.b000N`: Solution on all grids at this time (binary)

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.

For details see the **VisClaw Documentation**

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
```

Plotting options

Create a set of webpages showing all plots:

```
$ make .plots
```

Disadvantages:

- May take a while to plot all frames
  (can make frames in parallel with OpenMP)
- Can't zoom in dynamically or explore data

View plots interactively:

```
$ ipython --pylab

In[1]: from clawpack.visclaw.Iplotclaw import Iplotclaw

In[2]: ip = Iplotclaw()

In[3]: ip.plotloop()
PLOTCLAW>> ?

PLOTCLAW>> q
In[4]: Quit
```

GeoClaw Tutorial

www.geoclaw.org

Switch to live tutorial...

Get started at:

`https://github.com/clawpack/geoclaw_tutorial_csdms2019`

`https://tinyurl.com/y3g2adcp`

html version of notebooks:

`http://depts.washington.edu/clawpack/geoclaw/tutorial_csdms2019/`