Adaptive Mesh Refinement for Geophysical Flow Modeling

Donna Calhoun (Boise State University)

Carsten Burstedde (Univ. of Bonn, Germany)
Melody Shih (Columbia/BSU); Kyle Mandli
Ram Sampath (Centroid Lab); Steve Prescott
(Idaho National Labs)

David George (USGS), Marsha Berger (NYU) Randall LeVeque (Univ. of Washington), Kyle Mandli (Columbia), Melody Shih (Columbia), David Ketcheson (KAUST, Saudi Arabia), Grady Lemoine (CD-Adaptco, Bellevue WA)

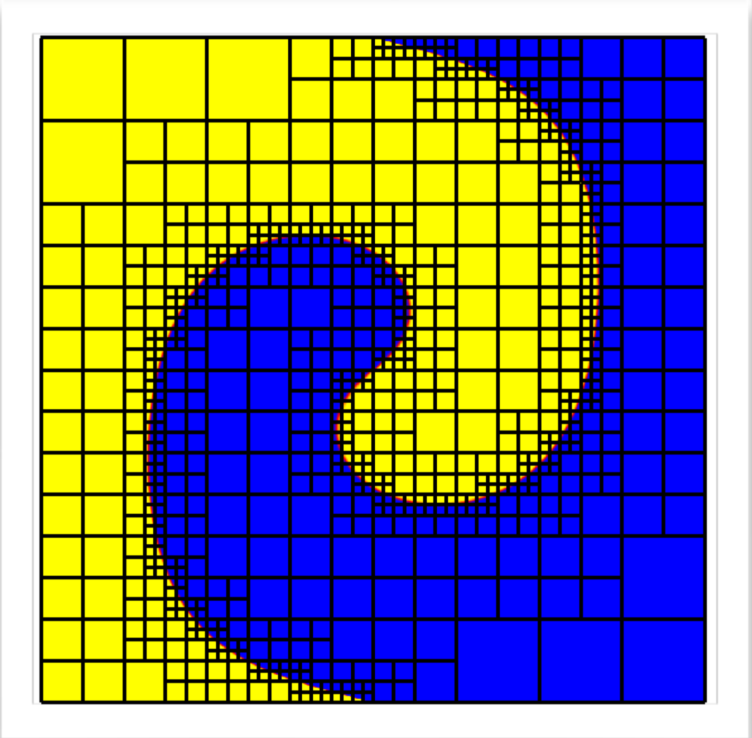
SIAM CSE
March 1 , 2017
Atlanta, GA

Multi-scale PDE software

Spatial and temporal adaptivity is necessary to make efficient use of computational resources.

GeoClaw Adaptive solver for depth averaged flows (shallow water wave equations), now widely used for simulation of tsunamis, inundation, debris flow and landslides. *Based on ClawPack* (Univ. of Washington, Columbia, NYU, USGS) **www.geoclaw.org**

ForestClaw Highly **parallel** quadtree based code for solving PDEs on mapped, multi-block domains. Has been recently extended to include GeoClaw (**www.forestclaw.org**)



ForestClaw Project

A parallel, adaptive library for logically Cartesian, mapped, multi-block domains

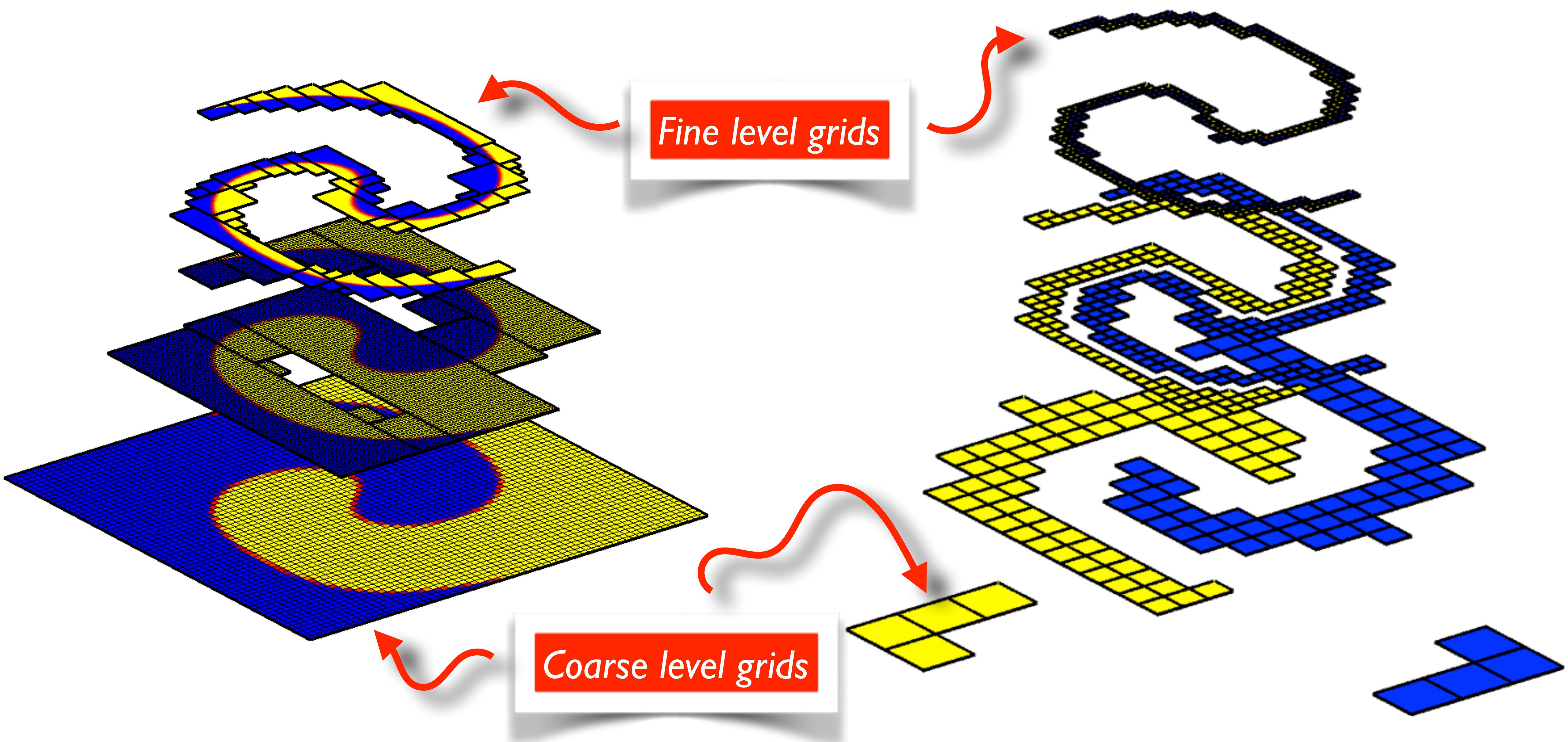
Features of ForestClaw include :

- Fully dynamic grid adaptivitiy, based on the **highly scalable p4est** dynamic grid management library (C. Burstedde, Univ. of Bonn, Germany)
- Each leaf of the quadtree contains a fixed-size uniform grid,
- Optional multi-rate time stepping strategy,
- Has **mapped, multi-block** capabilities, (cubed-sphere, for example) to allow for flexibility in physical domains,
- Modular design gives user flexibility in extending ForestClaw with Cartesian grid based solvers and packages.
- Uses essentially the same algorithmic components as patch-based AMR

ForestClaw development supported by the National Science Foundation

www.forestclaw.org

Approaches to AMR



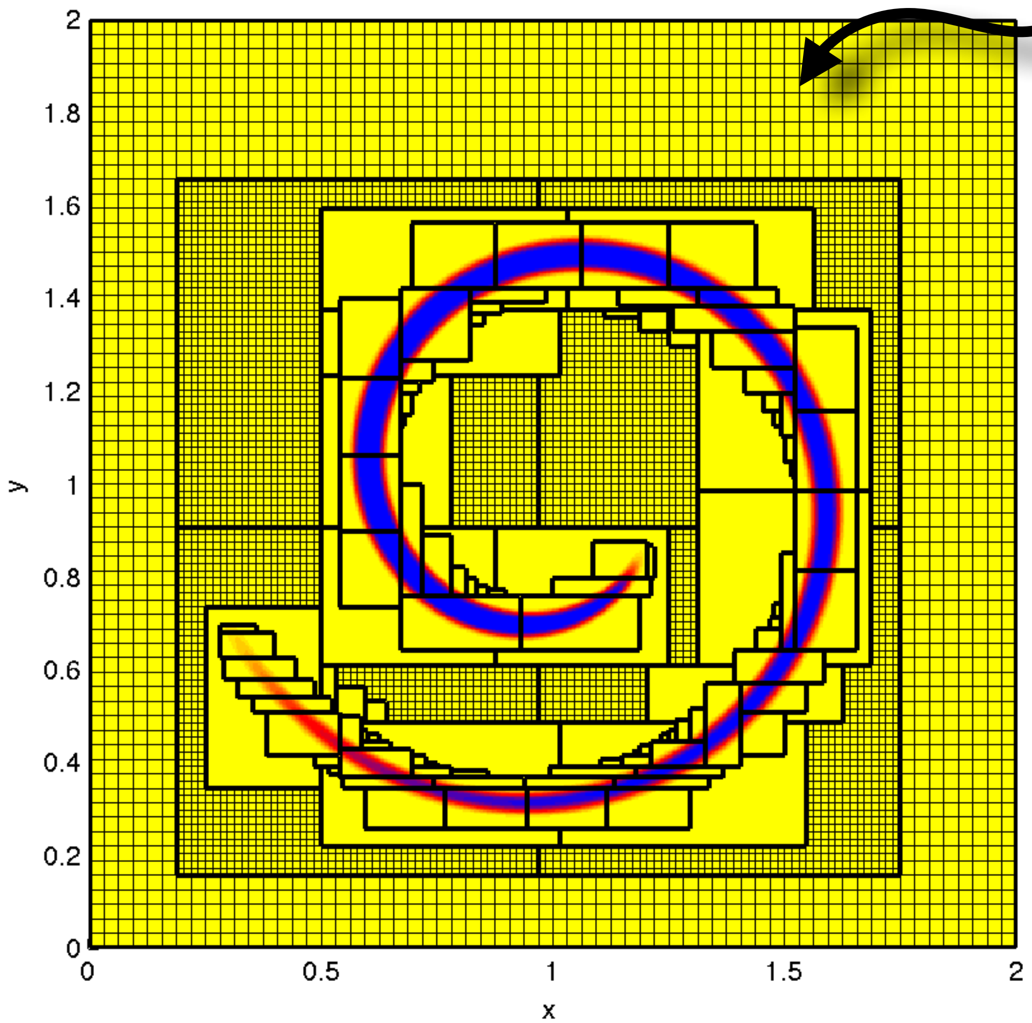
Original approach (M. Berger, J. Oliger, JCP, 1984)

Approach used in GeoClaw

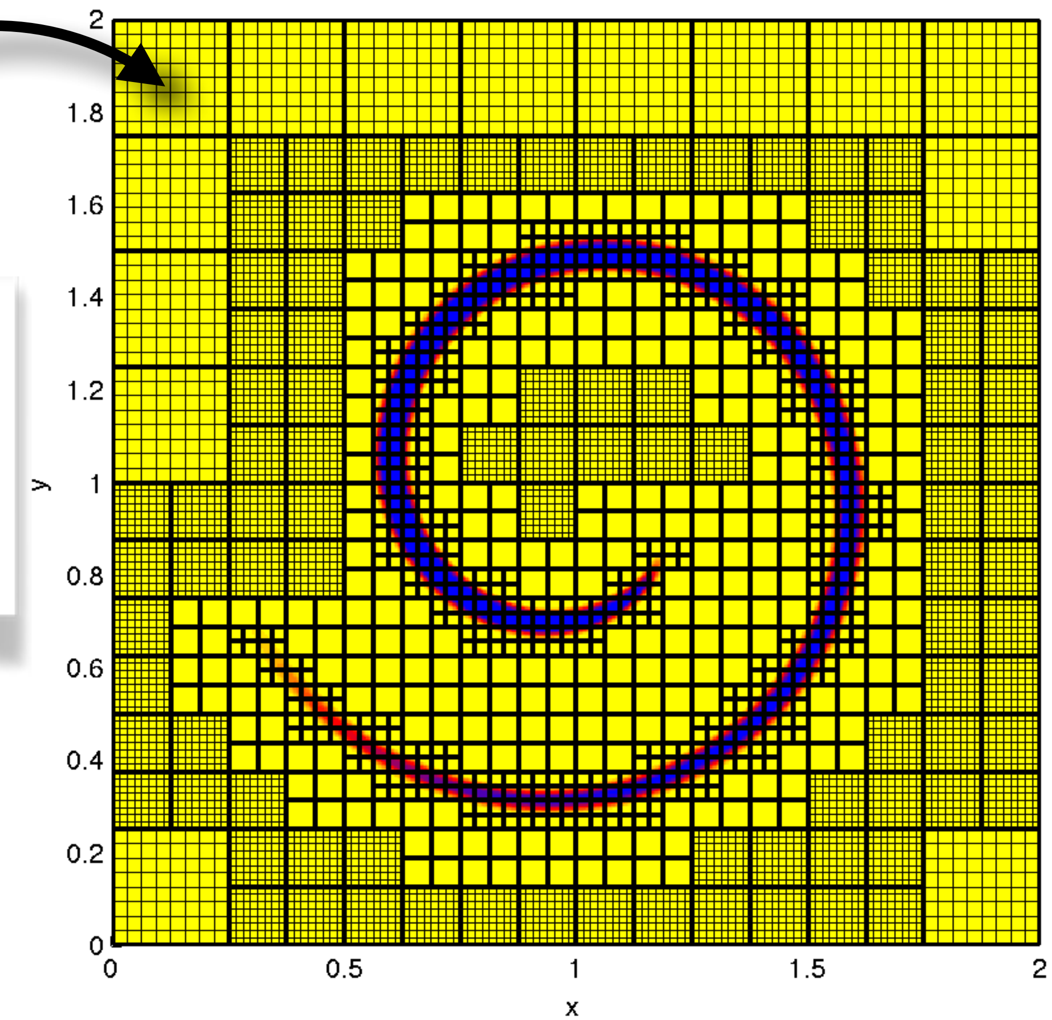
ForestClaw (D. Calhoun, C. Burstedde)

Adaptive Mesh Refinement

Original approach (Berger, 1984)



Quadtree approach



64x64 effective resolution

Chombo, AMRClaw, GeoClaw, Boxlib, SAMRAI

ParaMesh, ForestClaw

Improving computations

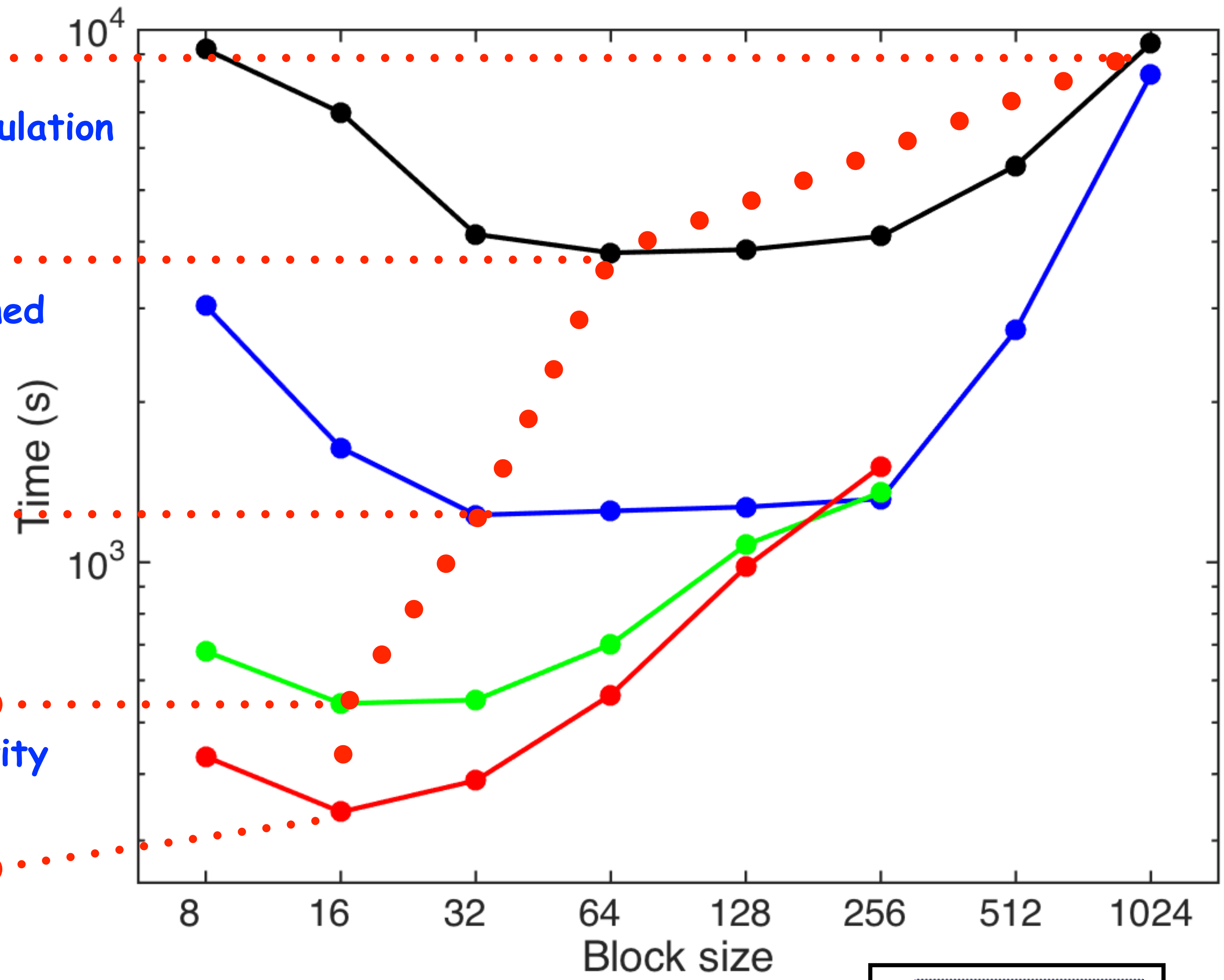
~2.5 h
Single grid calculation

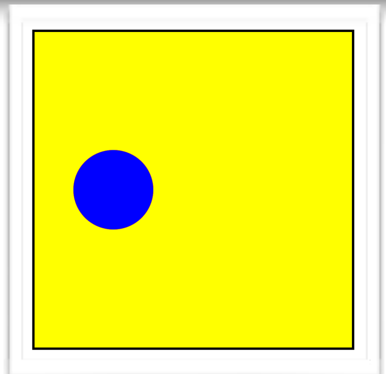
~1 h
Uniformly refined

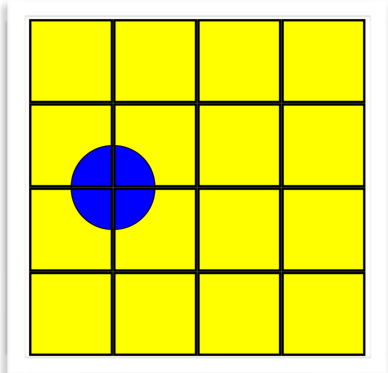
~12.5 min
Parallelism

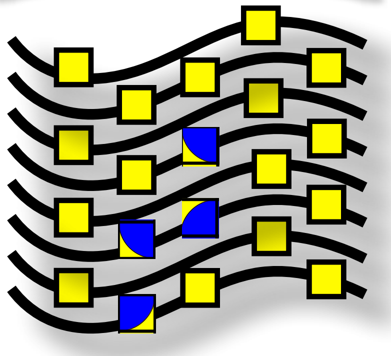
~9 min
Spatial adaptivity

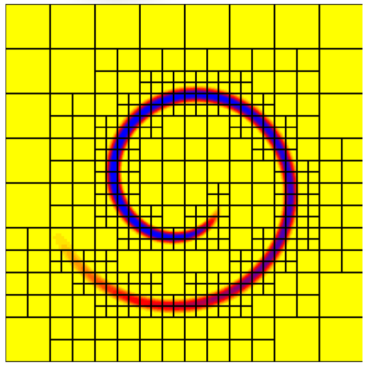
~5.5 min
Adaptive time stepping



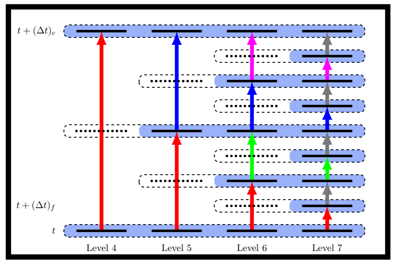






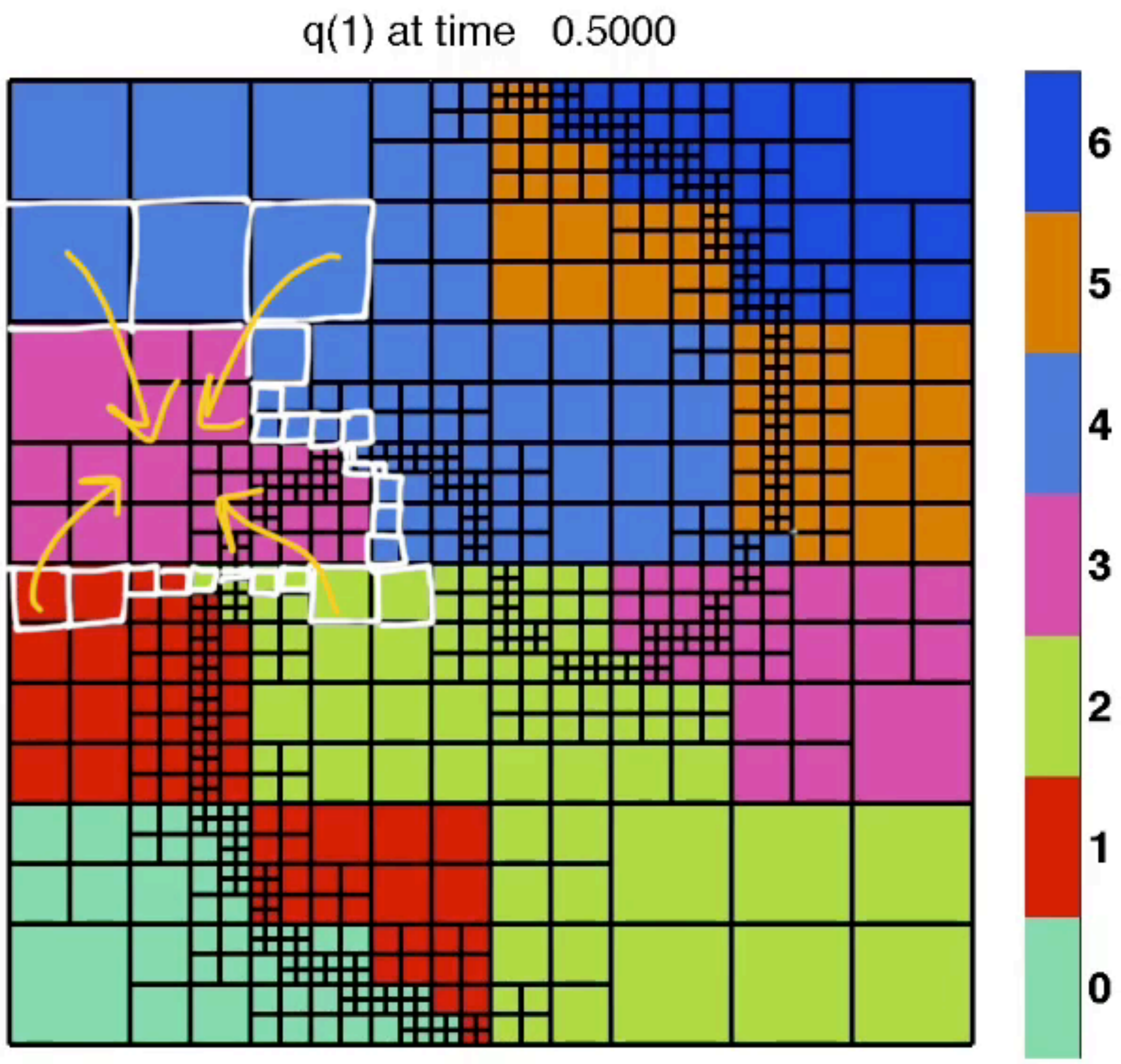


Almost 30 times improvement



Parallel implementation

- Grids are ordered and load balanced using *Morton ordering (z-ordering)*
- *Grids at processor boundaries are exchanged*



ForestClaw

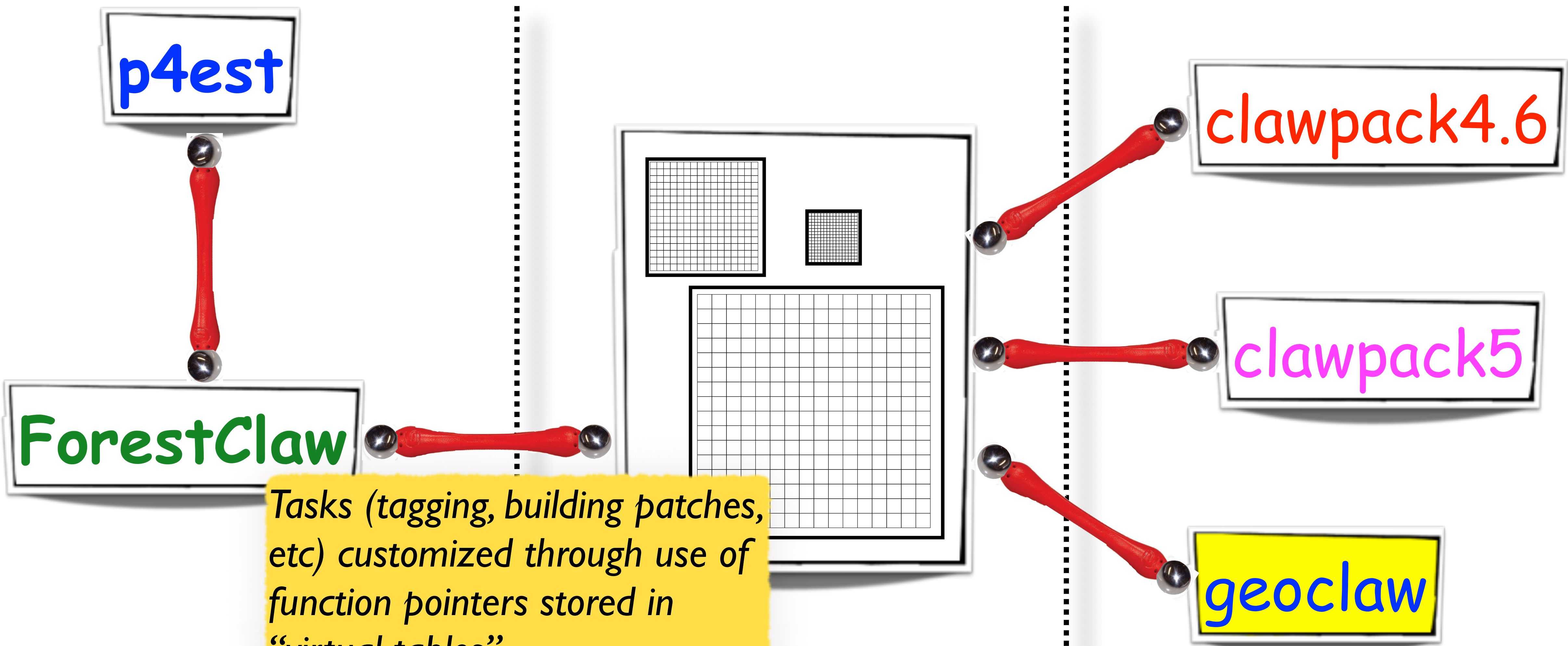
ForestClaw supports these modes for improving computations :

- Domain decomposition for better cache performance.
- Parallelism. Space-filling curves allow for load balancing and non-square parallel regions (e. g. easy to run on 17 processors)
- Dynamic spatial adaptivity for following solution features of interest.
- Adaptive time stepping to reduce communication between grids by reducing number of ghost cell communications, and reduce number of grids that need to be updated.

Optimizations below the patch level are not yet handled

- Loop optimizations (beyond what Clawpack/Geoclaw already do)
- Acceleration using GPUs, MICs, FPGAs and so on,
- Blocking within patches to reduce ghost cell communication

ForestClaw Solvers (so far)



Tasks (tagging, building patches, etc) customized through use of function pointers stored in "virtual tables"

ForestClaw is mostly C; original solvers libraries left largely untouched (in original Fortran)

Time stepping, dynamic grid management, input/ output.

ClawPatch

Solvers

GeoClaw Extension of ForestClaw

What stays the same

- Geo-ForestClaw uses all of existing Riemann solvers in GeoClaw, along with all bathymetry handling routines.

What had to be modified

- ForestClaw requires "coarsening" criteria, since we don't store underlying coarse grid meshes
- Gauges had to re-implemented for the quadtree mesh, but this leveraged the underlying **p4est** fast search algorithms.
- Customized averaging and interpolation routines to take into account bathymetry

Problems?

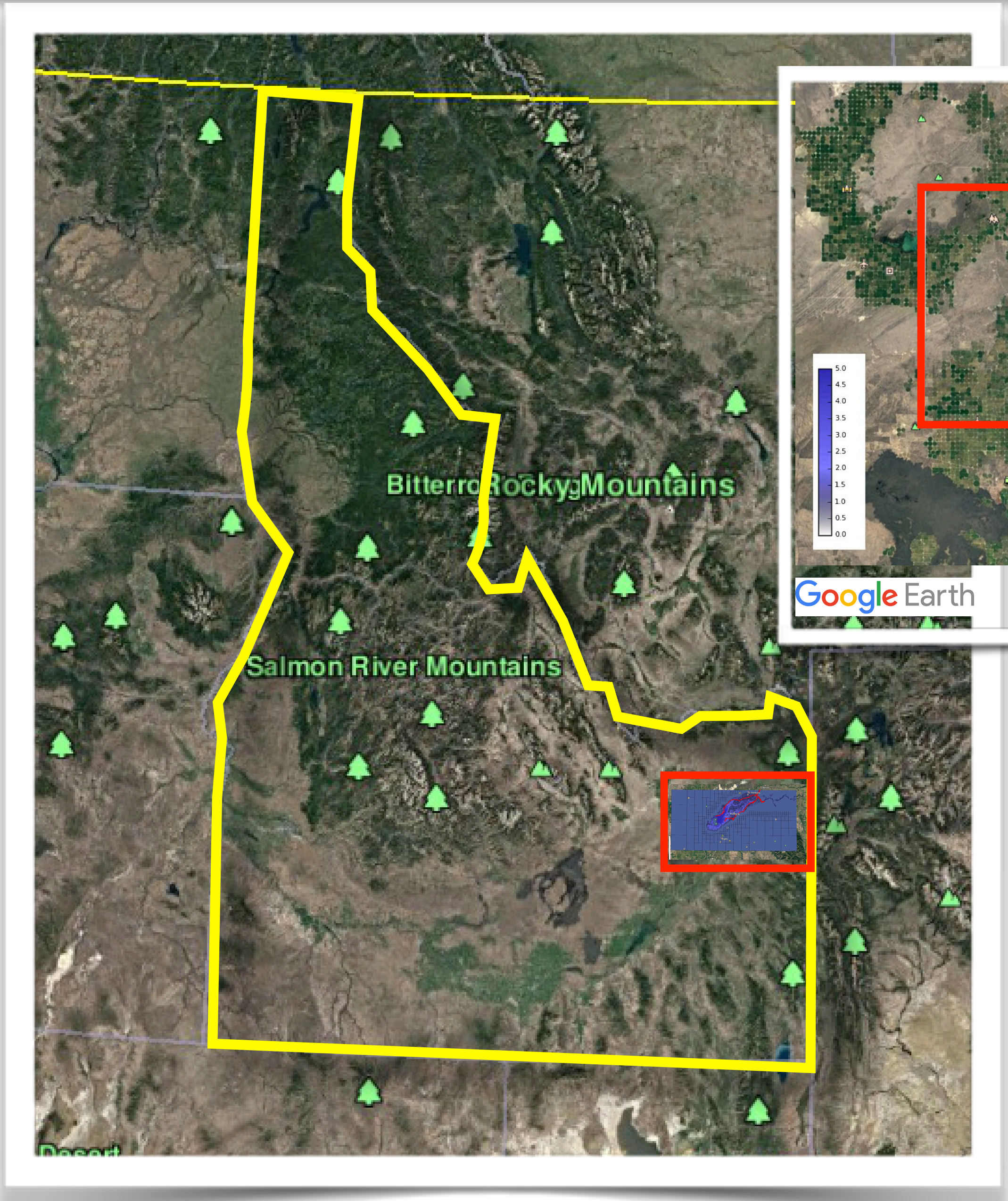
- Setting module values (done in GeoClaw driver routine)
- Argument-free subroutines (due to reliance on f90 modules).
- f90 conventions that are only implemented in later compilers

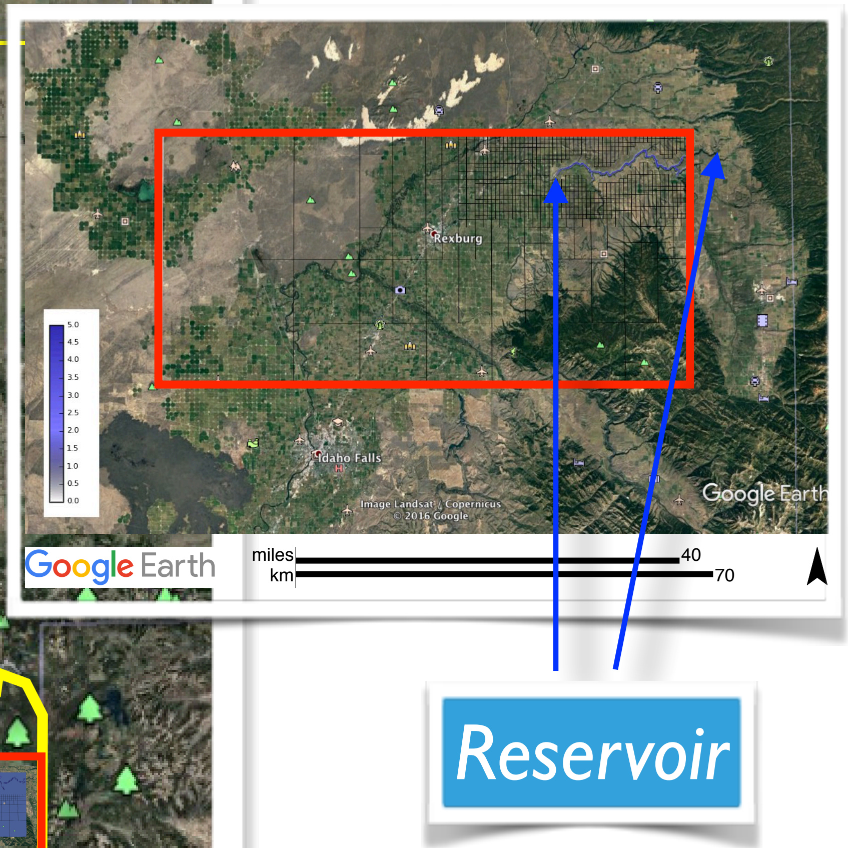
Teton Dam Failure, June 5, 1976

In collaboration with Idaho National Laboratories, we have been using GeoClaw to simulate the Teton Dam failure

- On June 5, 1976, the Teton Dam in eastern Idaho failed.
- 11 people died and $2b in damages; several cites were inundated, including Rexburg, ID.
- Historical data used as validation for using GeoClaw to study potential flooding of nuclear power plants,
- Collaborators include Steve Prescott (INL), Ram Sampath (Centroid Lab), and BSU undergraduates Cody Casteneda (Mechanical Engineering) and Stephanie Potter (Mathematics); Melody Shih (Columbia Univ.) and Kyle Mandli (Columbia)

Teton Dam Failure, June 5, 1976





Computational Area : 88km x 42km

~40m resolution : ~2048 x 1024 effective resolution

~20m resolution : ~4096 x 2048 effective resolution

Teton Dam Failure, June 5, 1976

8 minutes before dam failure

Teton Dam Failure, June 5, 1976

~11:52 AM, June 5, 1976

Teton Dam Failure, June 5, 1976

By WaterArchives.org from Sacramento, California, USA - [IDAHO-L-0010] Teton Dam Flood - Newdale, CC BY-SA 2.0,

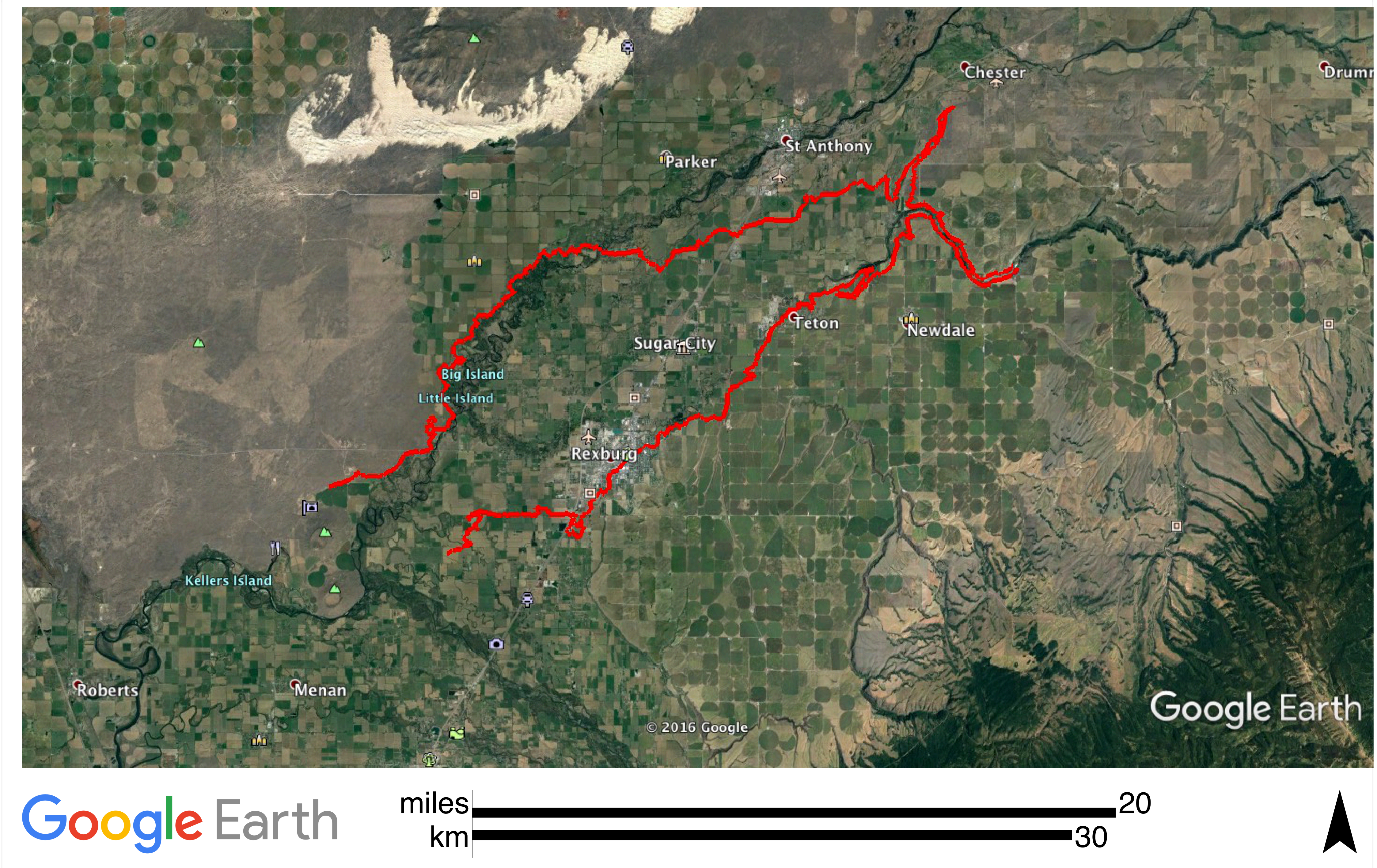
Teton Dam Failure, June 5, 1976

Historical Data

Table 2 Teton Dam Failure Summary Flood Data (Primary source: USGS Open-File Report 77-765)					
Location	Miles from Dam	Flood Arrival Time	Flood Arrival Travel Time (time from embankment breach)	Peak Flow (cubic feet per second)	Flood Description
Teton Canyon	2.5	12:05 p.m. June 5	8 minutes	2,300,000	50 to 75 ft wall-of-water
Near mouth of Teton Canyon	5.0	12:20 p.m.	23 minutes		
Wilford	8.4				120 of the 154 homes "completely swept away"
Town of Teton	8.8	12:30 p.m.	33 minutes	1,060,000	Only tiny fraction flooded
Sugar City	12.3	About 1:30 p.m.	1.5 hours		15-foot wall-of-water
Rexburg	15.3	About 2:30 p.m.	2.5 hours		6 to 8 feet in a few minutes
Roberts	43.1	9:00 p.m.	9 hours		
Idaho Falls	63.0	1 a.m. June 6	13 hours	90,500	
Shelley	71.2	2 a.m.	14 hours	67,300	Peak 21 hours after arrival. 0.5 feet per hour average rate of rise.

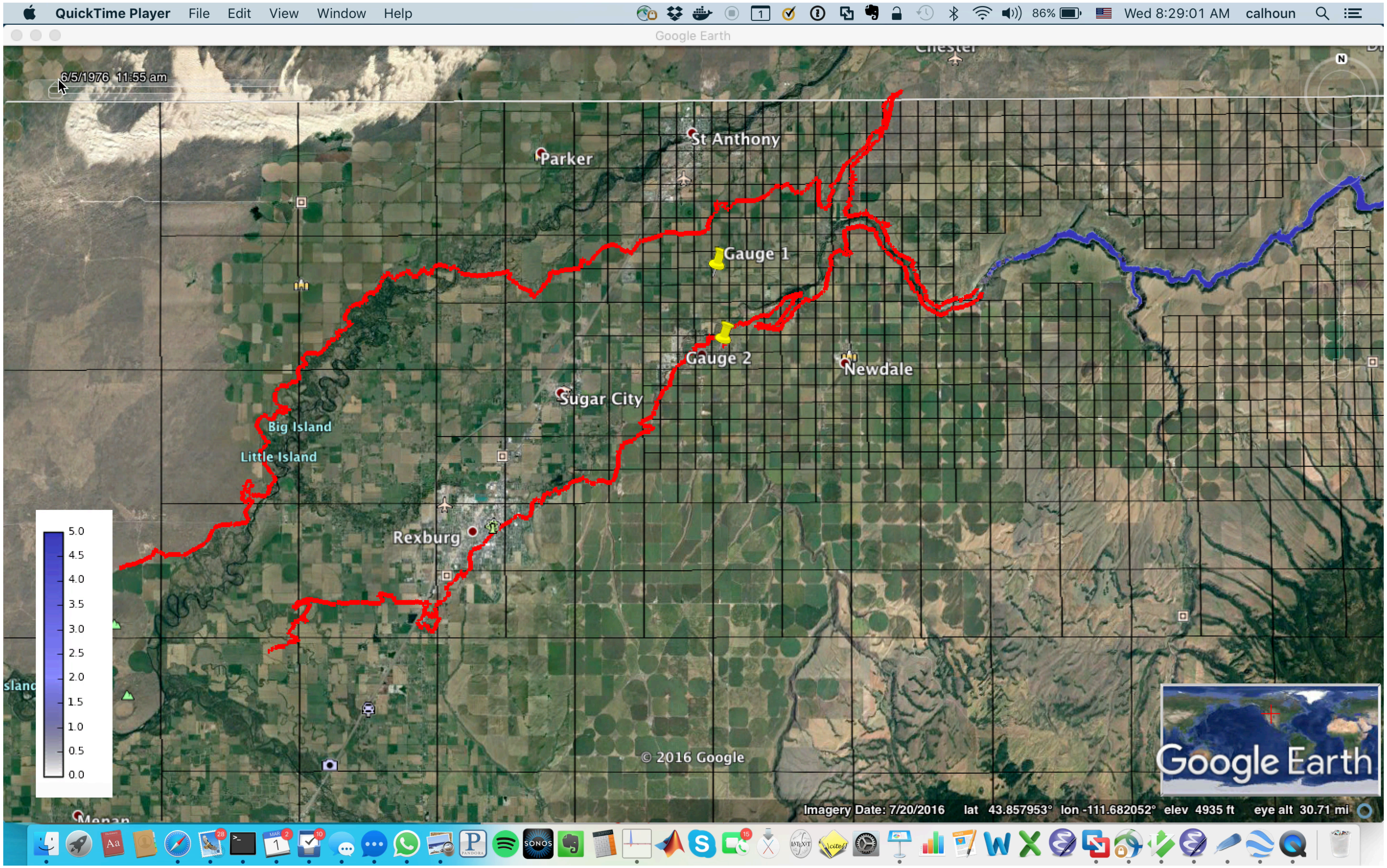
W. Graham, "Reclamation : Managing water in the west, The Teton Dam Failure - An effective warning and evacuation", U.S. Department of the Interior, Bureau of Reclamation, Denver Colorado

Historical Data

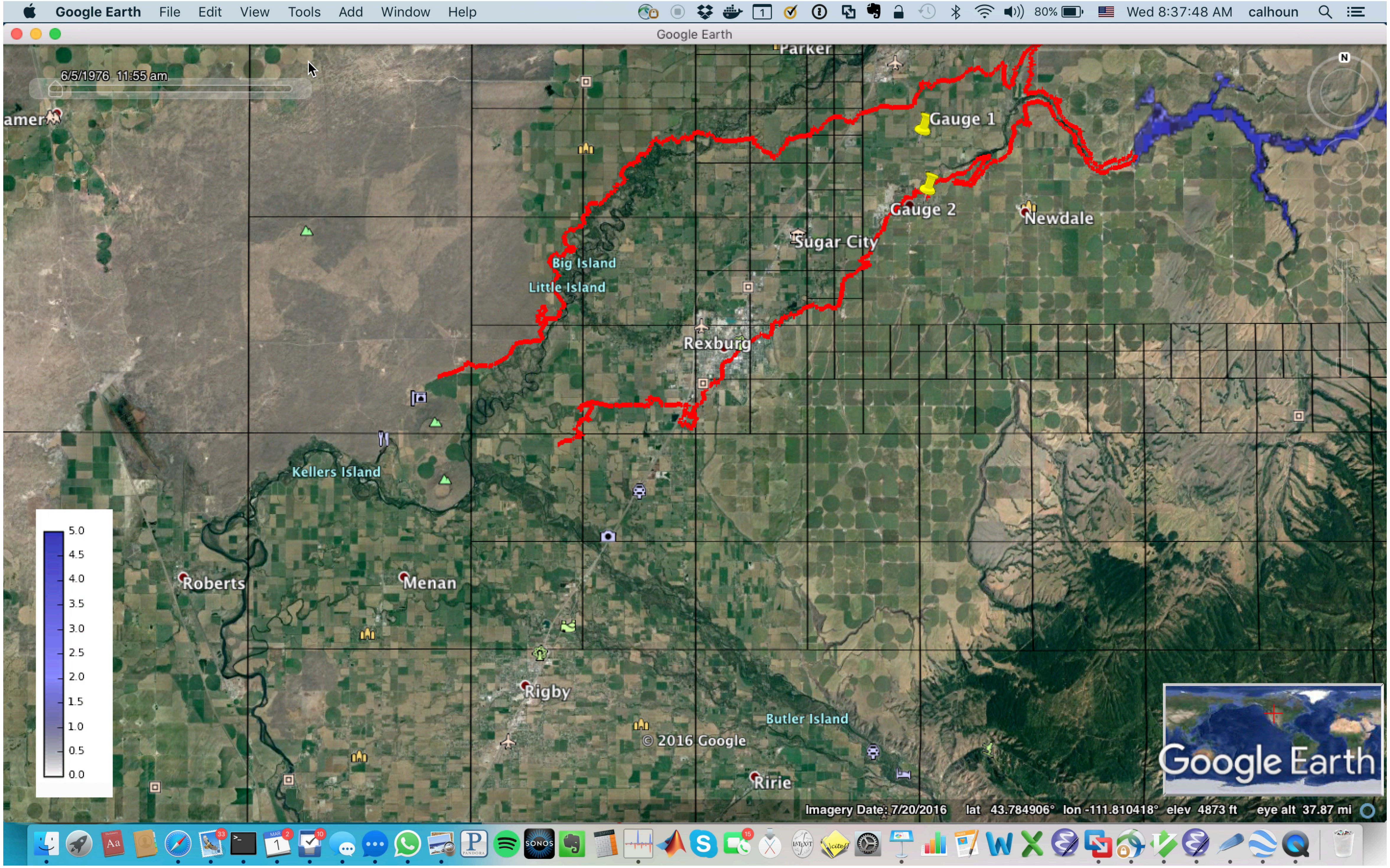


Recreated by S. Danielle Anderson, Student intern, INL

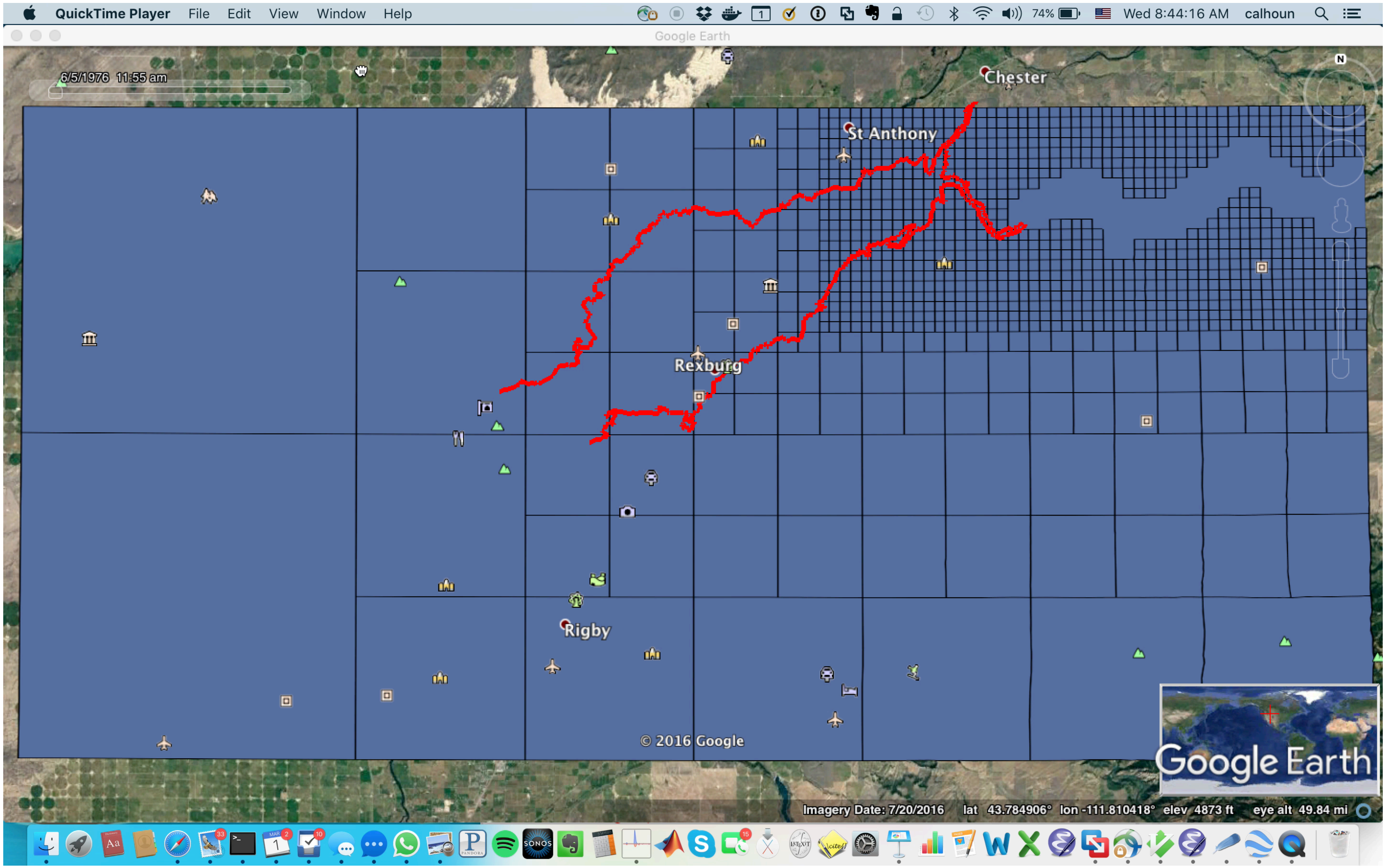
Simulation Results



Simulation Results



Simulation Results



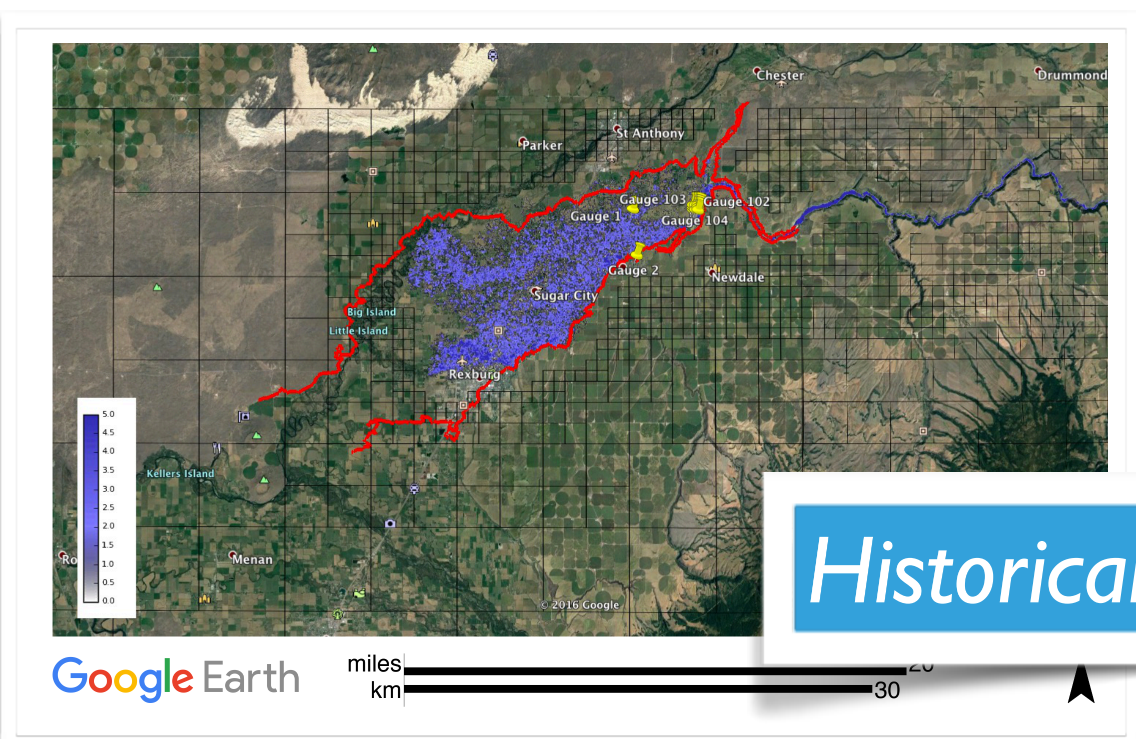
Results

Where are we getting things right?

- Good agreement with historical flood boundary\
- Arrival times depend on amount of water initially in the reservoir

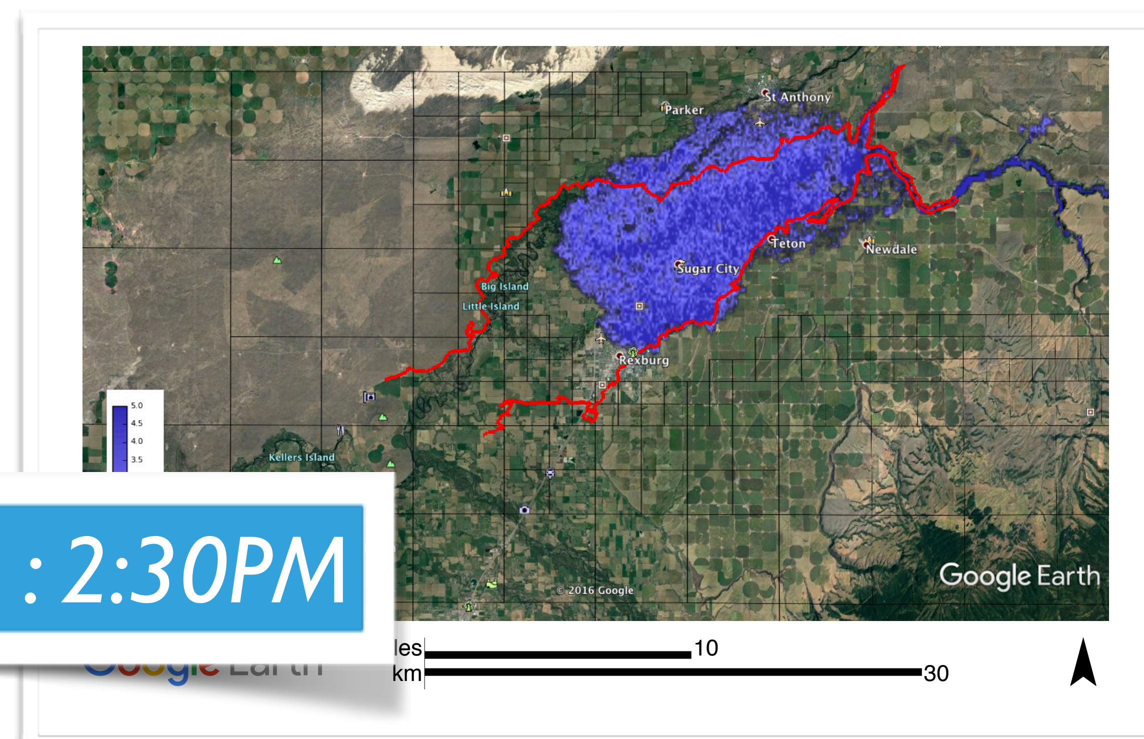
Where do we need work?

- Better model of the dam burst to modulate the initial flow of water



Arrival in Rexburg ~3:30PM

Historical Time : 2:30PM



Arrival in Rexburg ~1:30PM

Parallel Performance

~ 40m resolution

2048 x 1024
eff. res.

Procs	14	28	56	112
Wall (s)	1297.1	729.1	393.2	227.7
Speed-up	1.00	1.78	3.30	5.70
Efficiency	**100%**	**89%**	**82%**	**71%**
Grids per processor	**153**	**76**	**38**	**19**

~ 20m resolution

4096 x 2048
eff. res

Procs	14	28	56	112
Wall (s)	6416.8	3439.7	1797.9	960.4
Speed-up	1.00	1.87	3.57	6.68
Efficiency	**100%**	**93%**	**89%**	**84%**
Grids per processor	**380**	**190**	**95**	**47**

BSU R2 Cluster : 22 nodes (28 threads/node) E5 2780 v4 2.4GHz CPUs

AMR Efficiency

~ 40m resolution

Procs	Wall	Advance	(%)	Ghost Comm	(%)	Ghost fill	(%)	Regrid	(%)	Speed-up	Par. eff.
14	1297.1	818.1	**63%**	378.7	**29%**	72.7	**6%**	20.6	**2%**	1.0	**100%**
28	729.1	409.8	**56%**	254.9	**35%**	47.2	**6%**	10.6	**1%**	1.8	**89%**
56	393.2	205.1	**52%**	150.3	**38%**	26.6	**7%**	5.4	**1%**	3.3	**82%**
112	227.7	102.4	**45%**	100.4	**44%**	14.8	**6%**	3.5	**2%**	5.7	**71%**

~ 20m resolution

* Not regridding every time step

Procs	Wall	Advance	(%)	Ghost Comm	(%)	Ghost fill	(%)	Regrid	(%)	Speed-up	Par. eff.
14	6416.8	4482.7	**70%**	1515.2	**24%**	381.5	**6%**	23.9	**0%**	1.0	**100%**
28	3439.7	2242.9	**65%**	951.0	**28%**	220.5	**6%**	13.7	**0%**	1.9	**93%**
56	1797.9	1121.5	**62%**	538.3	**30%**	122.0	**7%**	6.9	**0%**	3.6	**89%**
112	960.4	560.3	**58%**	319.4	**33%**	67.4	**7%**	3.8	**0%**	6.7	**84%**

Outstanding issues

What governs good parallel performance?

- Sufficient number of grids per processor is key to good parallel efficiency
- Grid size (8x8, 16x16, 32x32 and so on) is also important,
- Ghost cell communication involves not only the solution but also bathymetry, needed to average fine grid values onto the coarse grid ghost cells

Outstanding issues

- Sub-cycling (local time stepping, multi-rate time stepping) in time for SWE?
- Handling console IO and user interface to Python routines
- Push the code to fine resolutions? (~5m, ~1m ….)
- Maximum flooding at a prescribed set of values ("fixed grid solutions").

Future work

Future work in ForestClaw for modeling dam failures

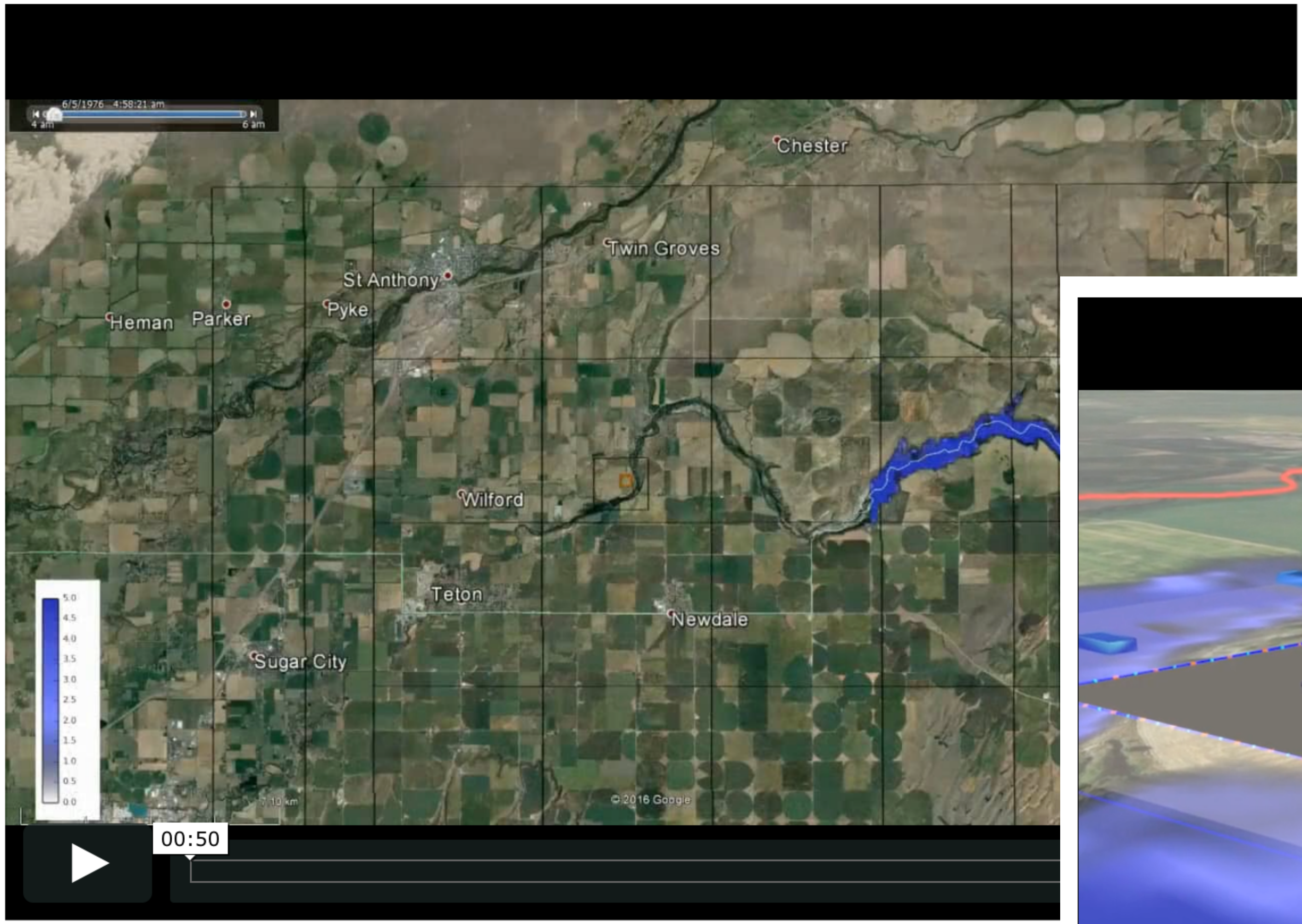
- Open MP parallelism + MPI ?
- Optimizations at the patch level (GPUs?)
- More benchmark runs
- Comparisons with AMRClaw
- Better modeling of dam failure
- Better tools for setting up problems

Further extensions of ForestClaw

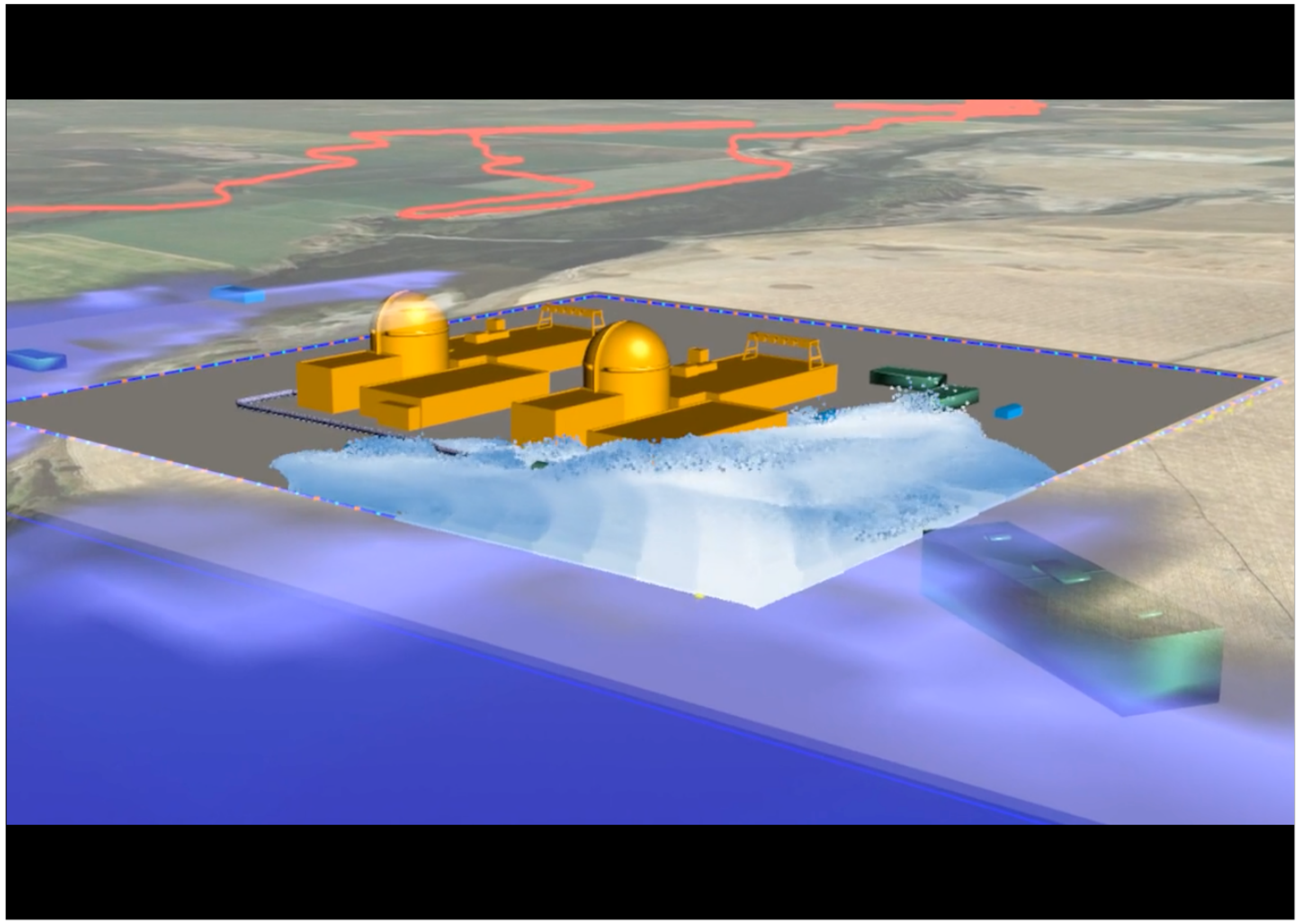
- Currently working with the USGS to port their **Ash3d** code, for modeling dispersion of volcanic ash, to ForestClaw
- Extensions to 2.5 and 3 dimensions
- More work towards latency hiding,

Thanks to NSF for supporting this work

Teton Dam Failure, June 5, 1976



GeoClaw + Neutrino

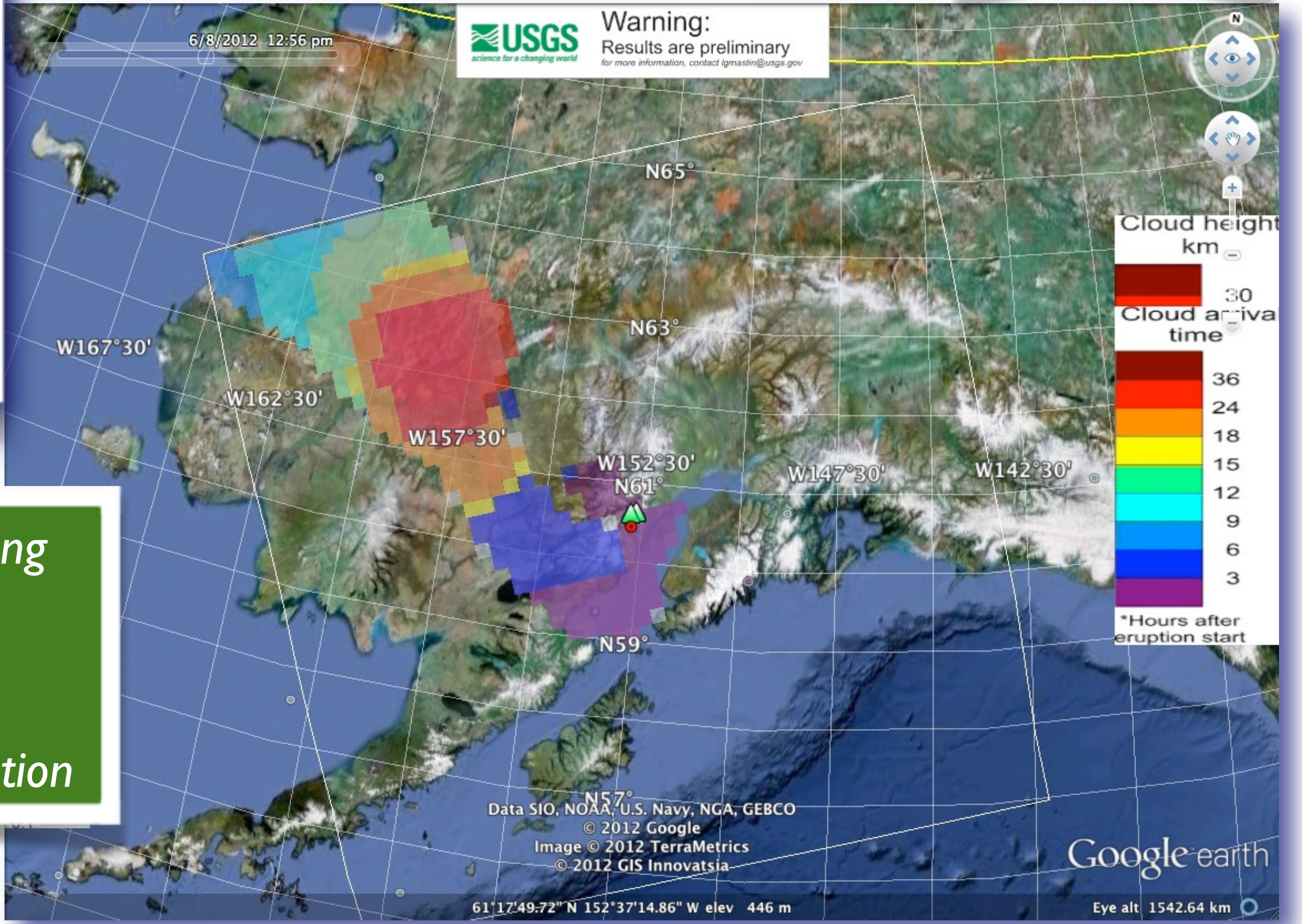


Ram Sampath, Centroid Lab, Los Angeles, CA

http://neutrinodynamics.com//portfolio-riverflood.html

Ash cloud modeling

Ash3d



- *Split horizontal, vertical time stepping*
- *Fully conservative,*
- *Eulerian, finite volume*
- *Algorithms based on wave propagation*

Ash3d :A finite-volume, conservative numerical model for ash transport and tephra deposition, Schwaiger, Denlinger, Mastin, JGR (2012)