IDAV View

Hank Childs
Computer Science Department
University of California, Davis

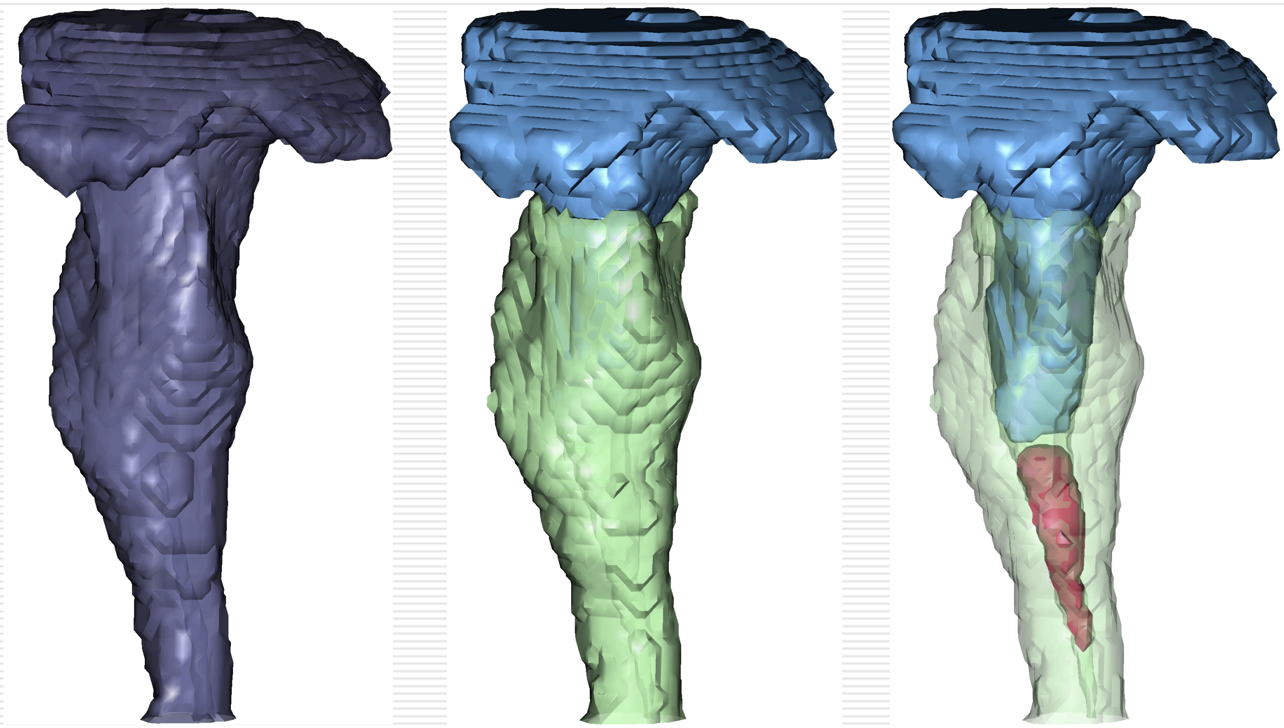
IDAV Overview

- IDAV: Institute for Data Analysis and Visualization at UC Davis
- Faculty
  - Ken Joy (director), Nina Amenta, Bernd Hamann, Nelson Max, Michael Neff, John Owens
- Researchers/Adjuncts
  - Hank Childs, Oliver Kreylos, Silvia Crevelli, Hans Hagen, Owen Carmichael
- Postdocs/Graduate Students
  - 5 postdocs, 30 graduate students.

Basic IDAV View

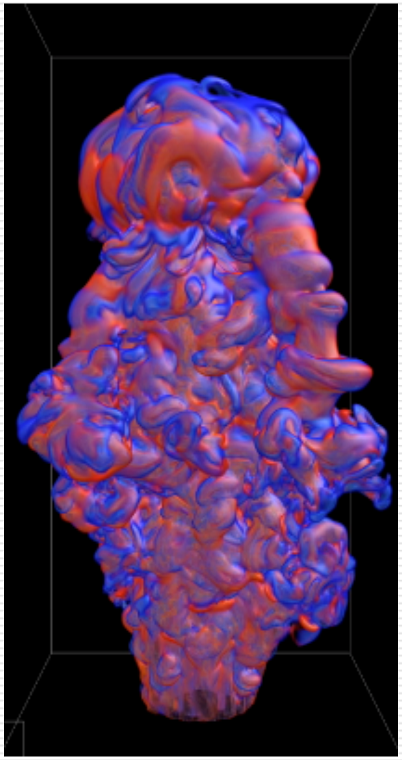
- Visualizing large scale data presents incredible challenges in both managing scale and data understanding.
- IDAV portfolio contains research in both areas:
  - Managing scale
    - Query-driven visualization
    - Visualization algorithms on the GPU
    - Particle advection
  - Data understanding
    - Function data (energy groups)
    - Embedded boundaries / material interfaces
    - Particle advection

Query-Driven Visualization

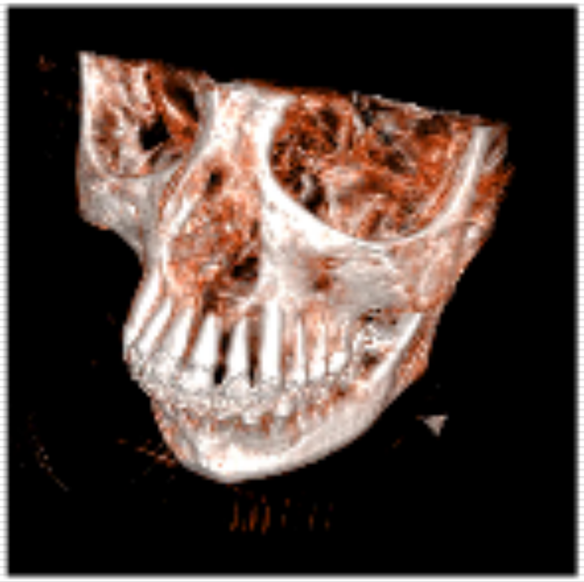


Courtesy Gossink, Joy, et al.

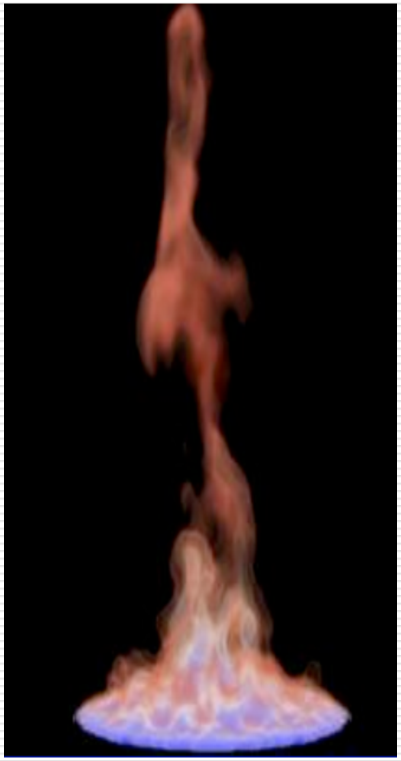
Visualization Algorithms on the GPU



FTLE computations of unstructured meshes on the GPU, courtesy Garth et al

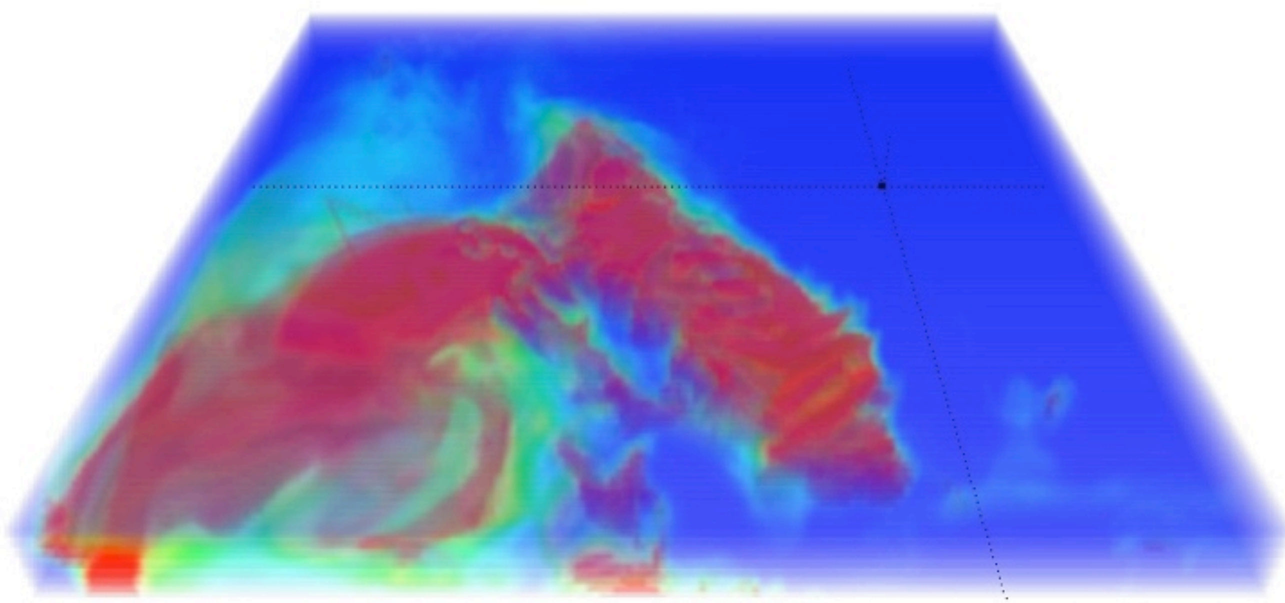


Using MapReduce to do GPU volume rendering, courtesy Stuart, Chen, Ma, and Owens.



GPU volume rendering at massive scale, courtesy Fogal (UUtah), Childs, et al.

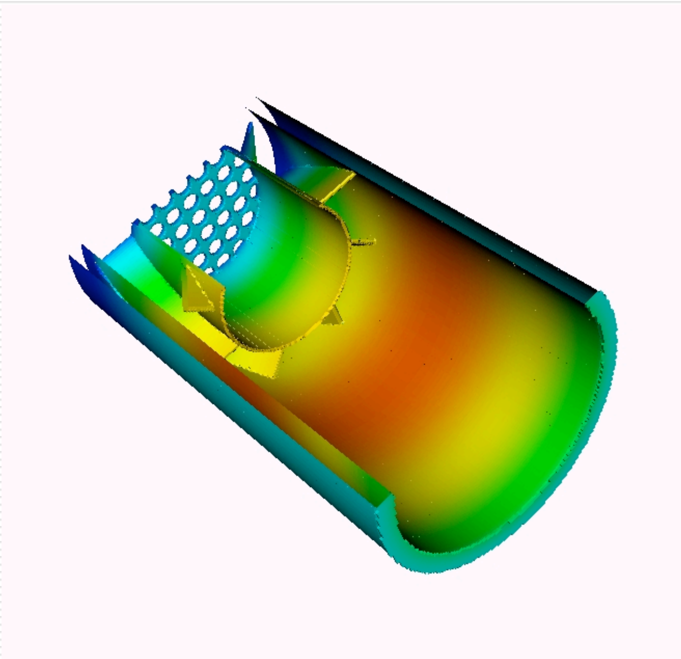
Visualization of Function Data

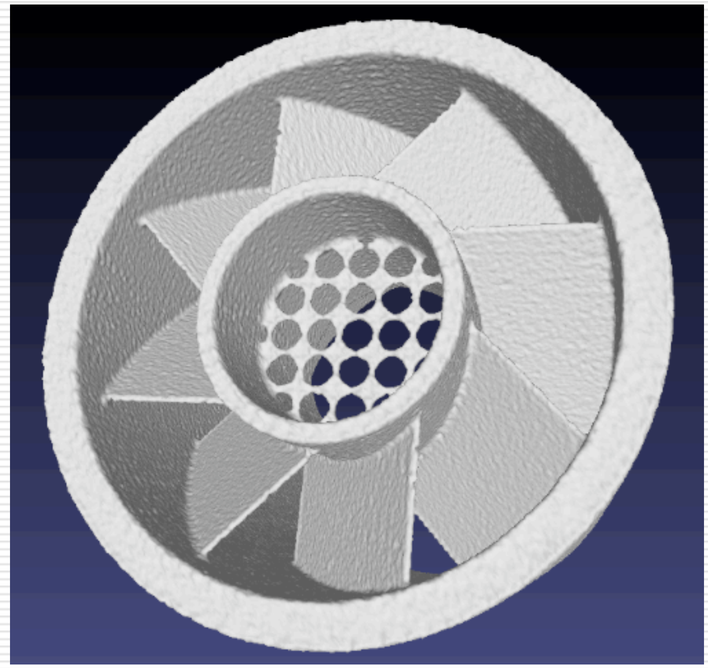


Air-pollution data from the San Joaquin Valley, CA. Each vertex has an associated function [particle size by number of particles]. This frame is from a large-scale 24-hour simulation of the air quality in the valley

Courtesy Anderson, Joy, et al.

Embedded Boundary/Material Interfaces





Cross section of a “swirler” (Courtesy of APDEC). The swirler is
boundaries are generated by active contour methods

Courtesy Anderson, Joy, et al.

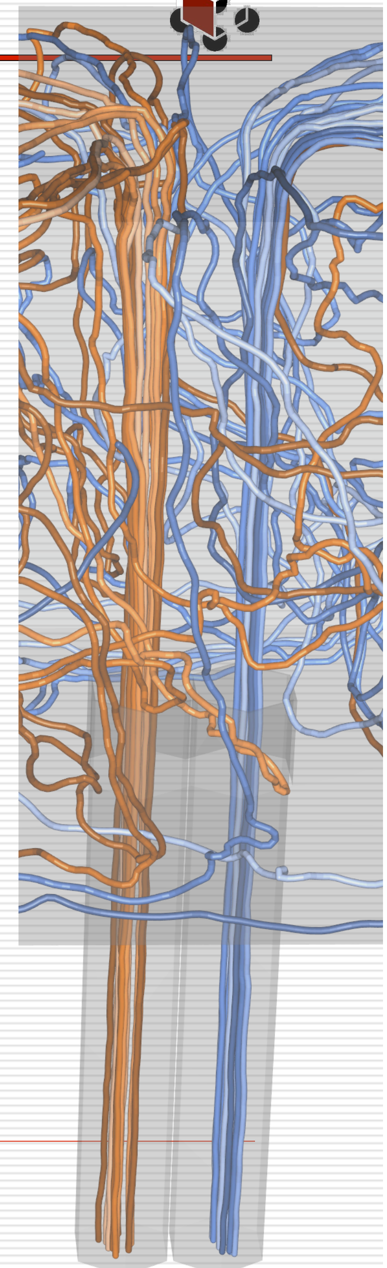
Particle advection basics

- Advecting particles create integral curves

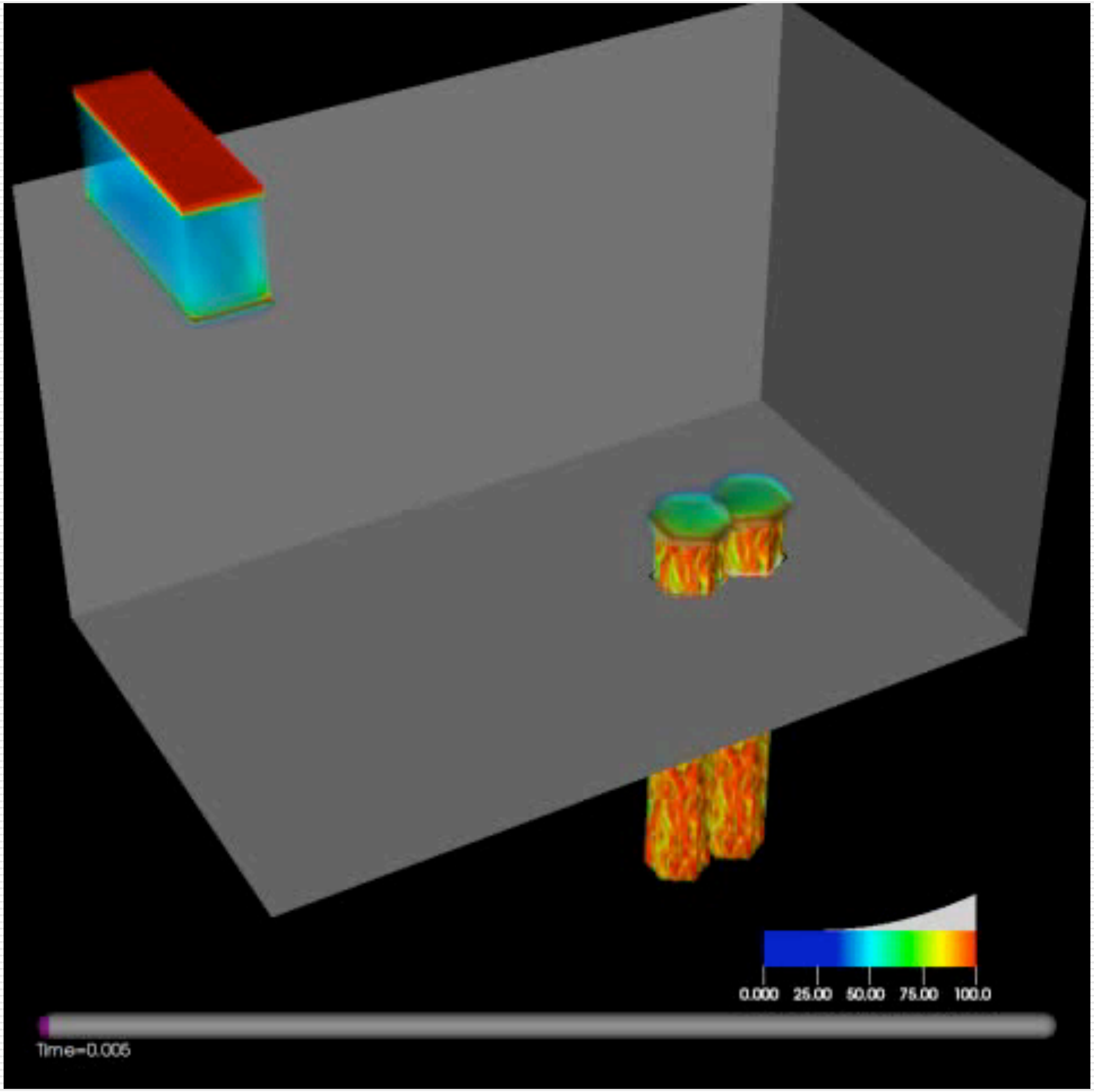
$$S'(t) = v(t, S(t)) \quad S(t_0) := x_0$$

Most of the remainder of this presentation explores what analysis we can do using particle advection as a building block.

- Streamlines: display particle path (instantaneous velocities)
- Pathlines: display particle path (velocity field evolves as particle moves)

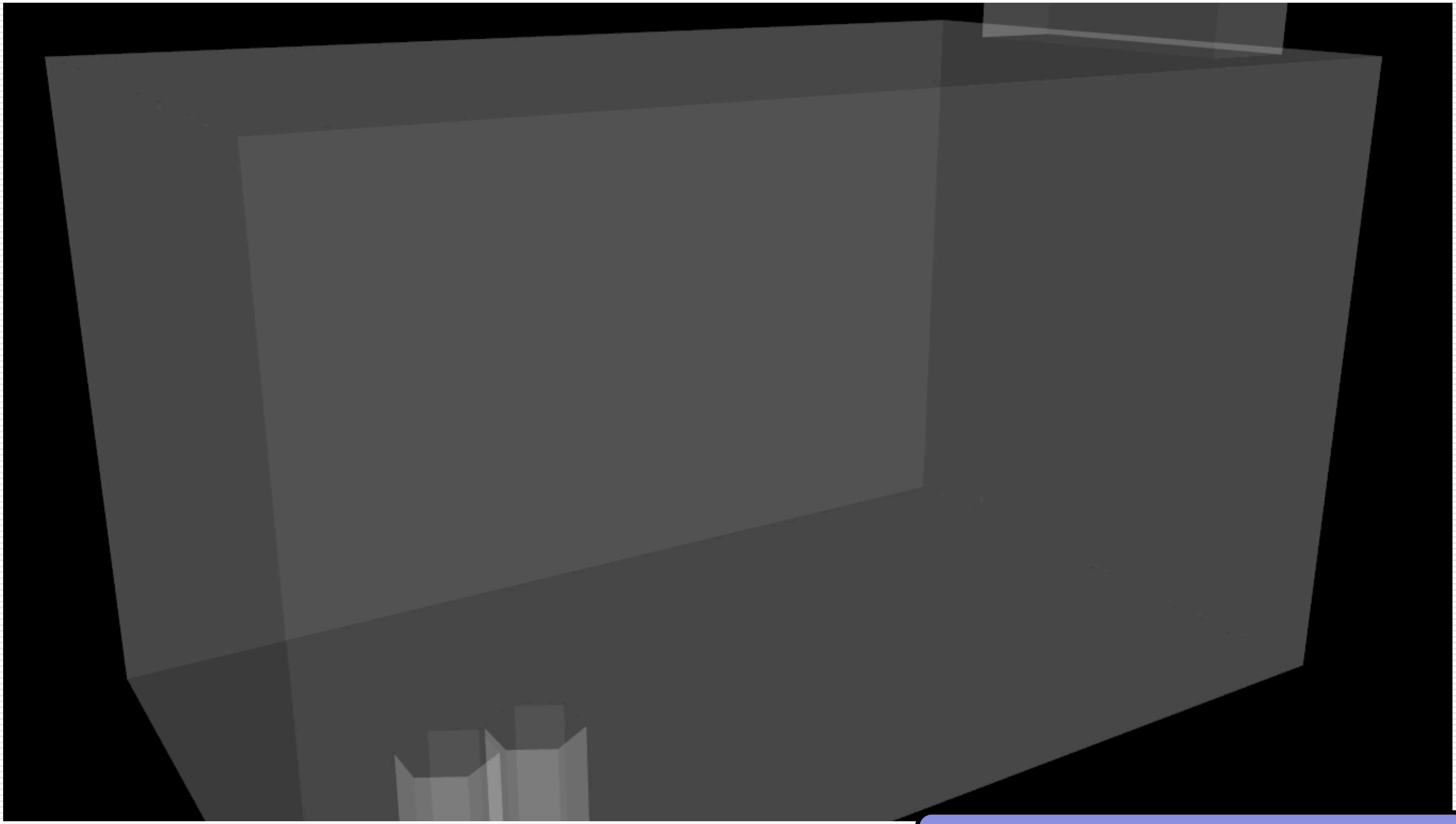


“The Fish Tank”

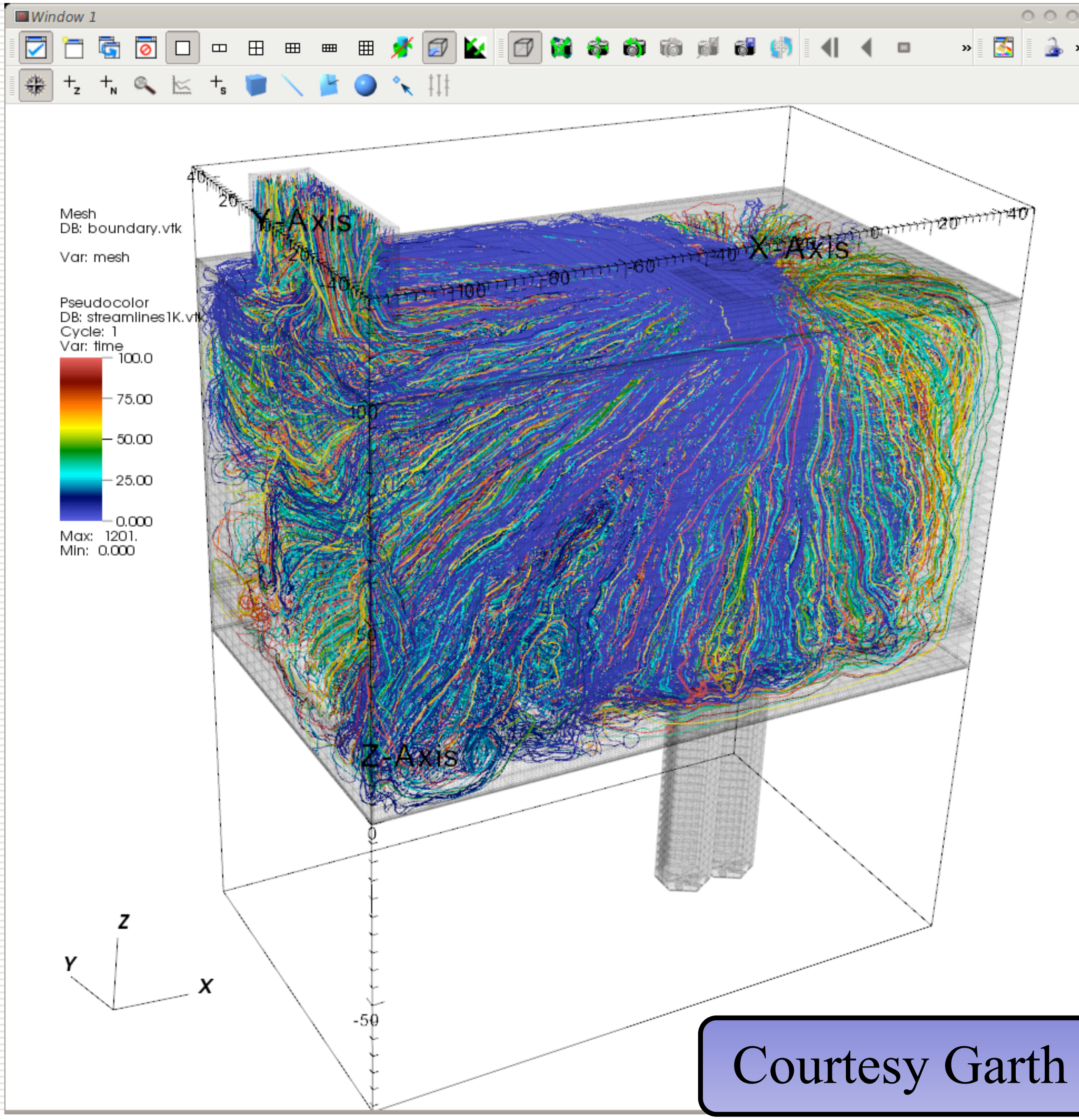


“Simulation of the Turbulent Flow of Coolant in an Advanced Recycling Nuclear Reactor.” Movie credits to Childs, Fischer, Obabko, Pointer, and Siegel

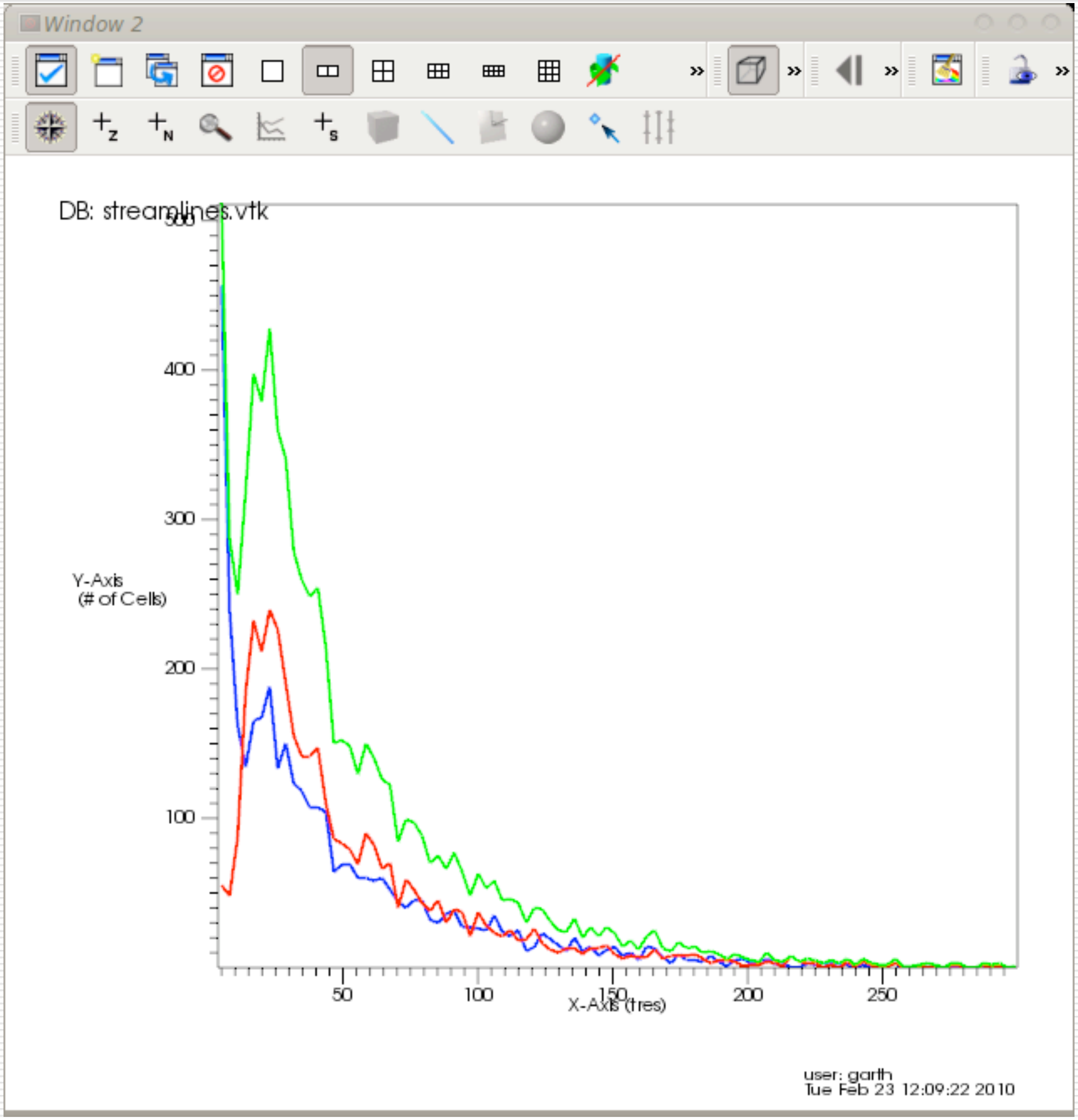
Particles Moving Through the “Fish Tank”



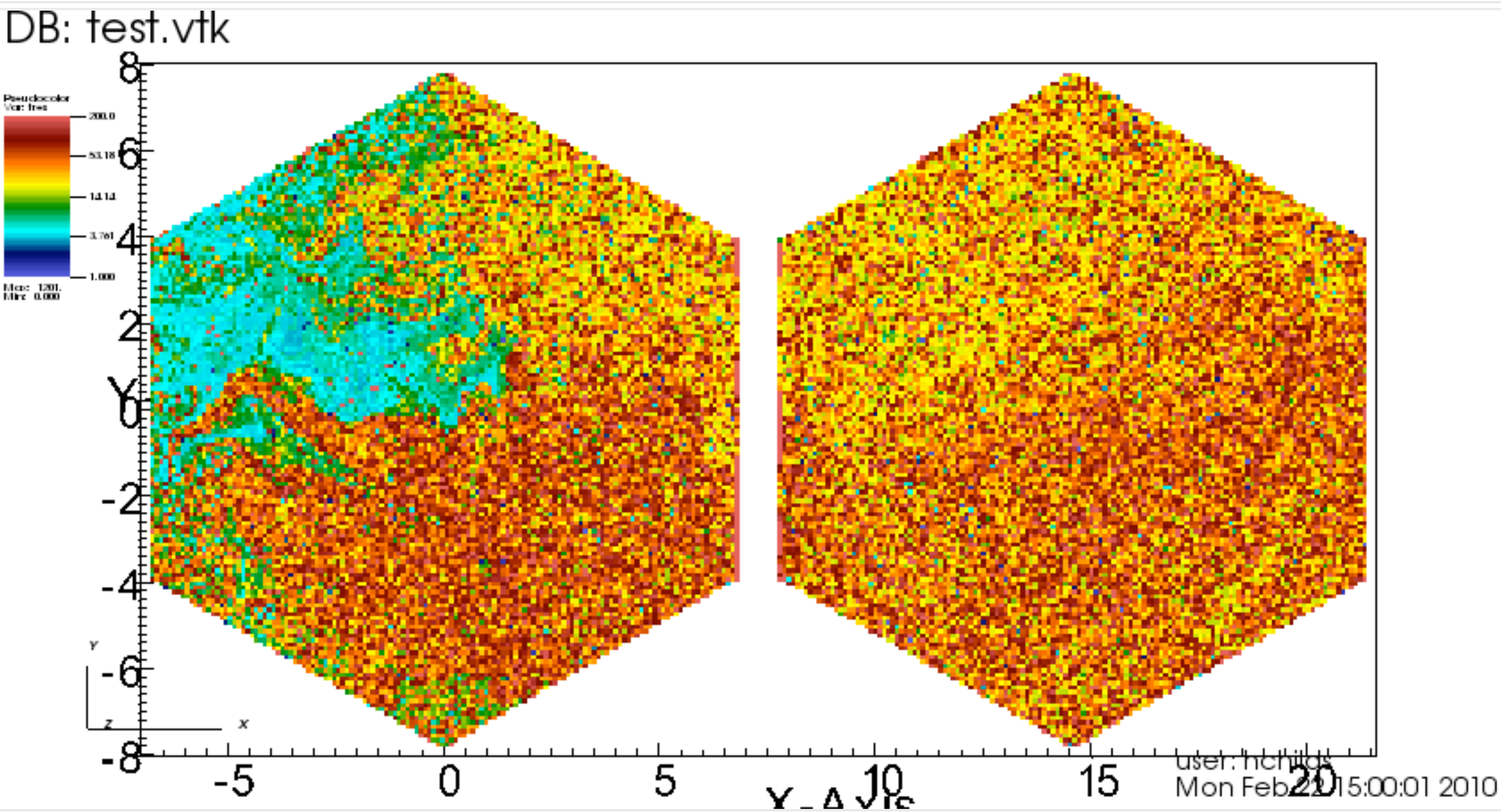
Courtesy Garth & Childs



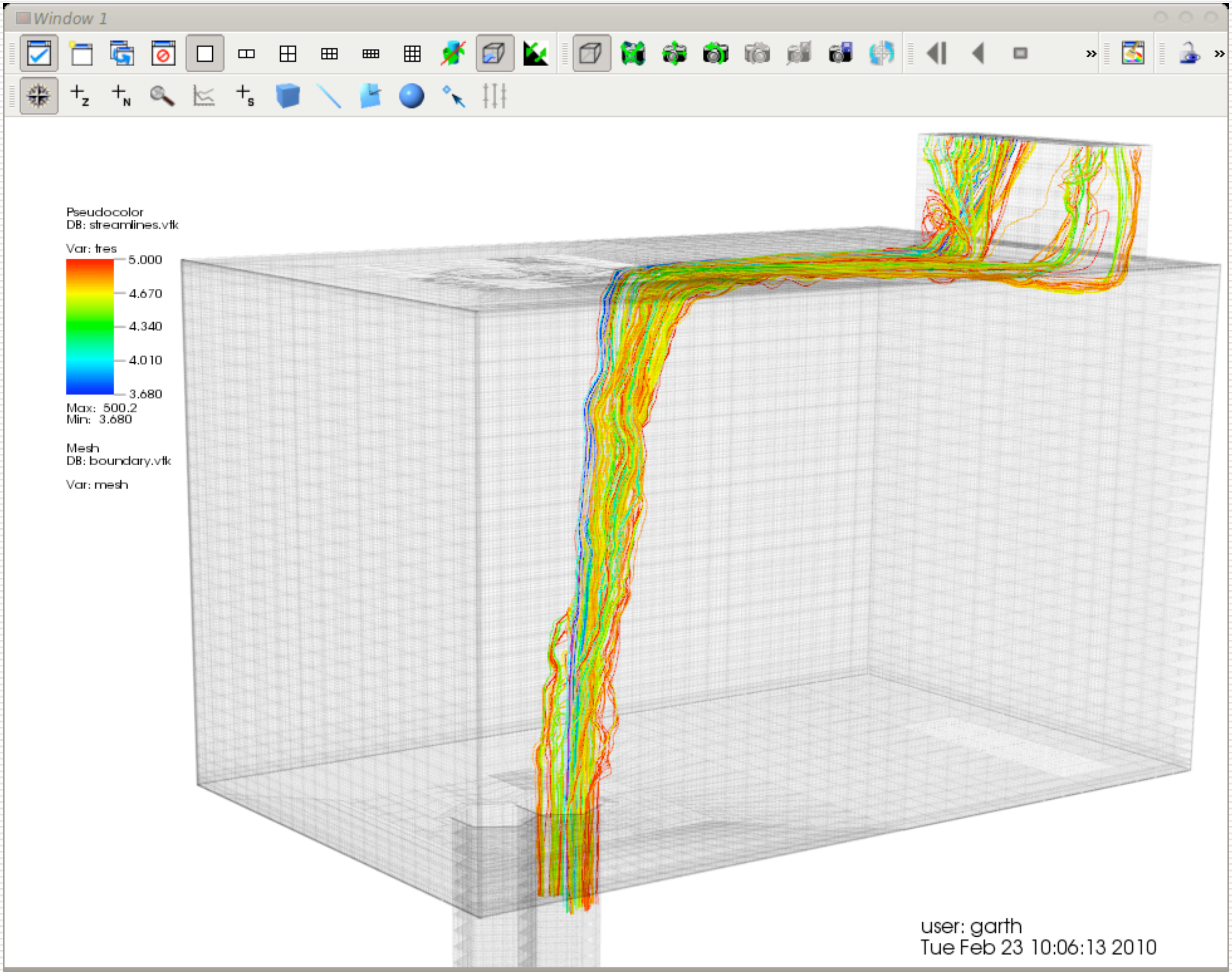
Courtesy Garth & Childs



Courtesy Garth & Childs



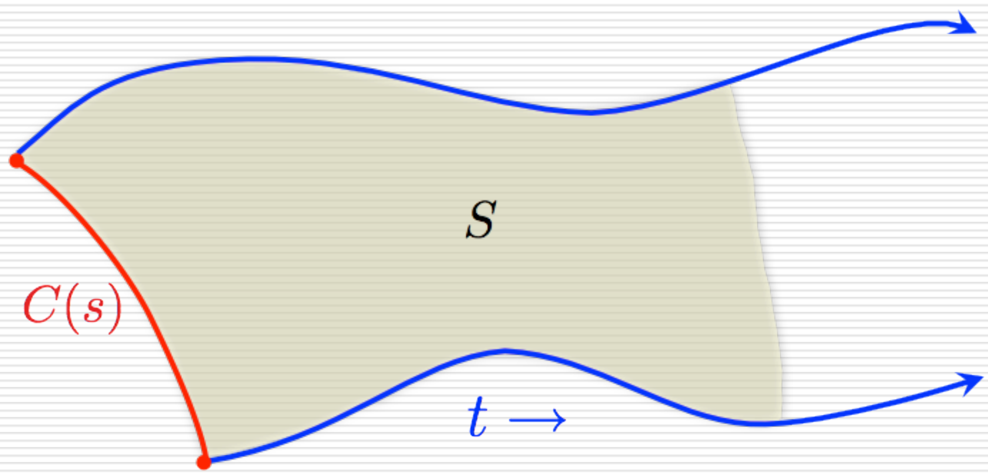
Courtesy Garth & Childs



Courtesy Garth & Childs

Sets of Streamlines

- Visualizing all integral curves…
  - … starting from a seed curve:
    Stream Surface or Path Surface

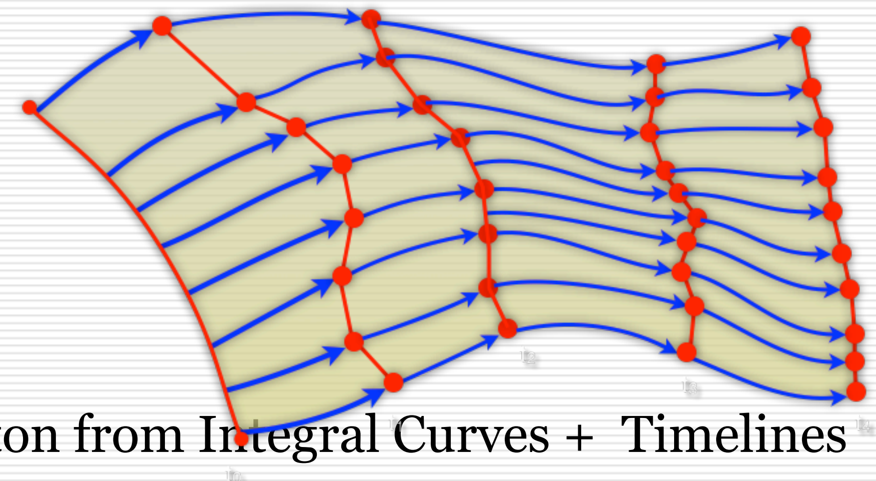


$$\frac{d}{dt}S(s,t) = \vec{v}\,(t, S(s,t)) \qquad S(s,0) := C(s)$$

Courtesy Garth

Sets of Streamlines

- Stream surface computation:

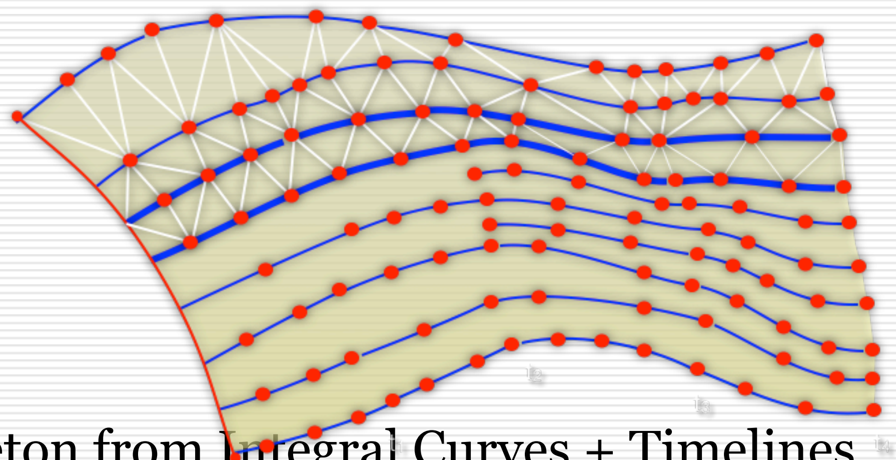


  - Skeleton from Integral Curves + Timelines

Courtesy Garth

Sets of Streamlines

- Stream surface computation:



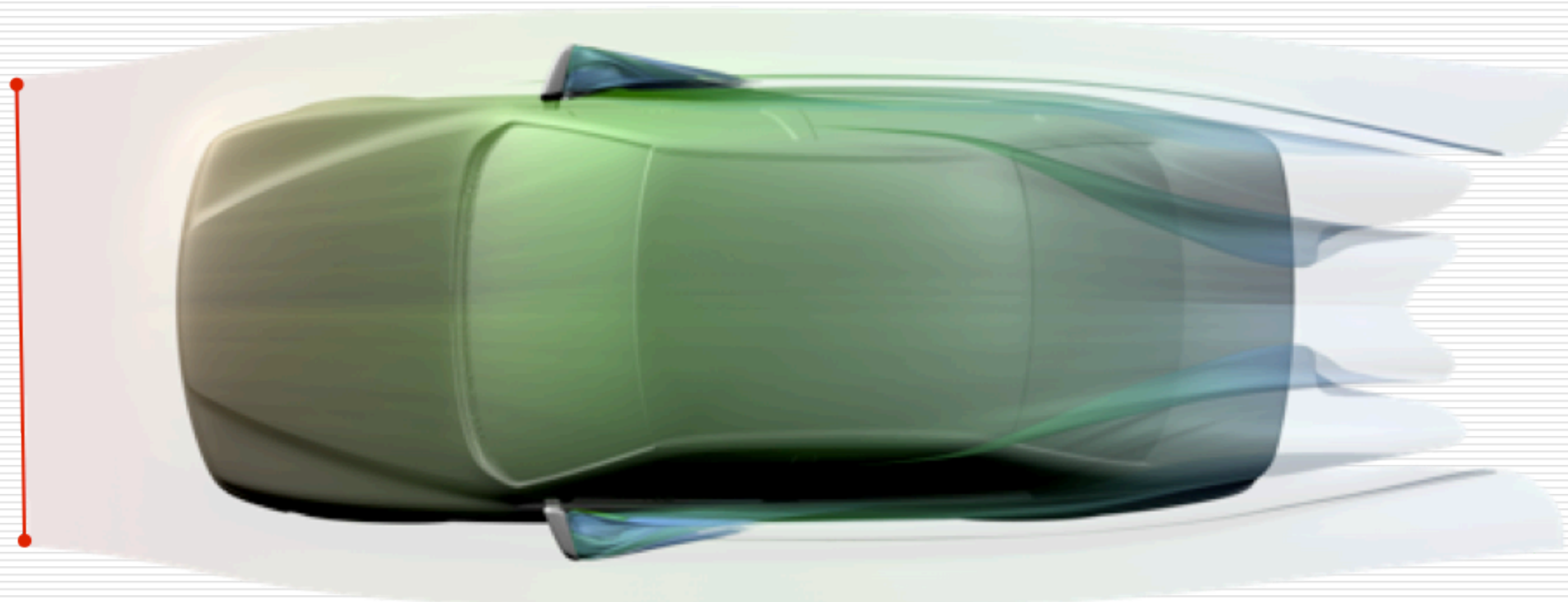
  - Skeleton from Integral Curves + Timelines
  - Triangulation

Generation of Accurate Integral Surfaces in Time-Dependent Vector Fields. C. Garth, H. Krishnan, X. Tricoche, T. Bobach, K. I. Joy. In IEEE TVCG, 14(6):1404–1411, 2007

Courtesy Garth

Sets of Streamlines

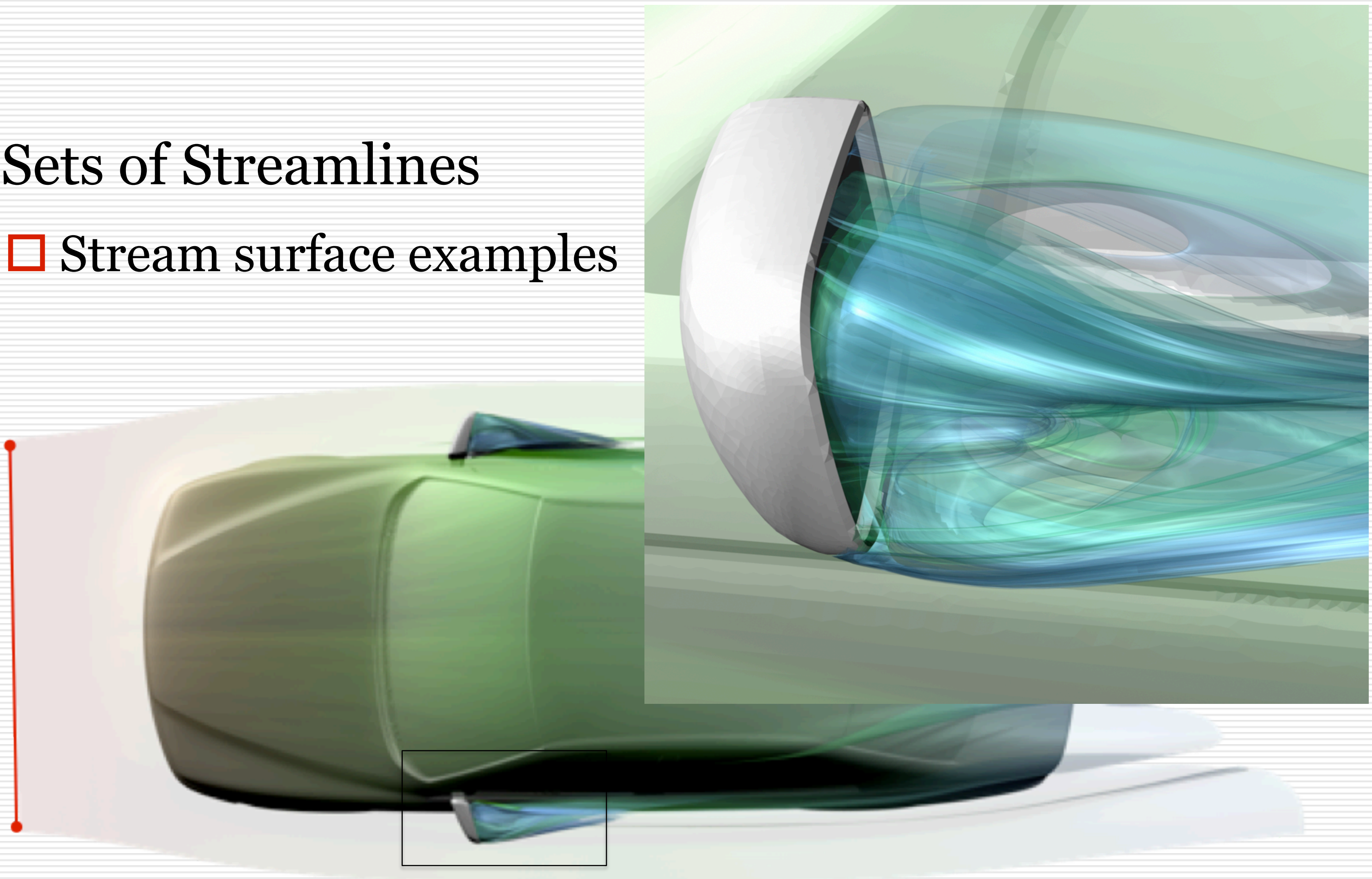
- Visualizing all integral curves…
  - … starting from a seed curve:
    Stream Surface or Path Surface



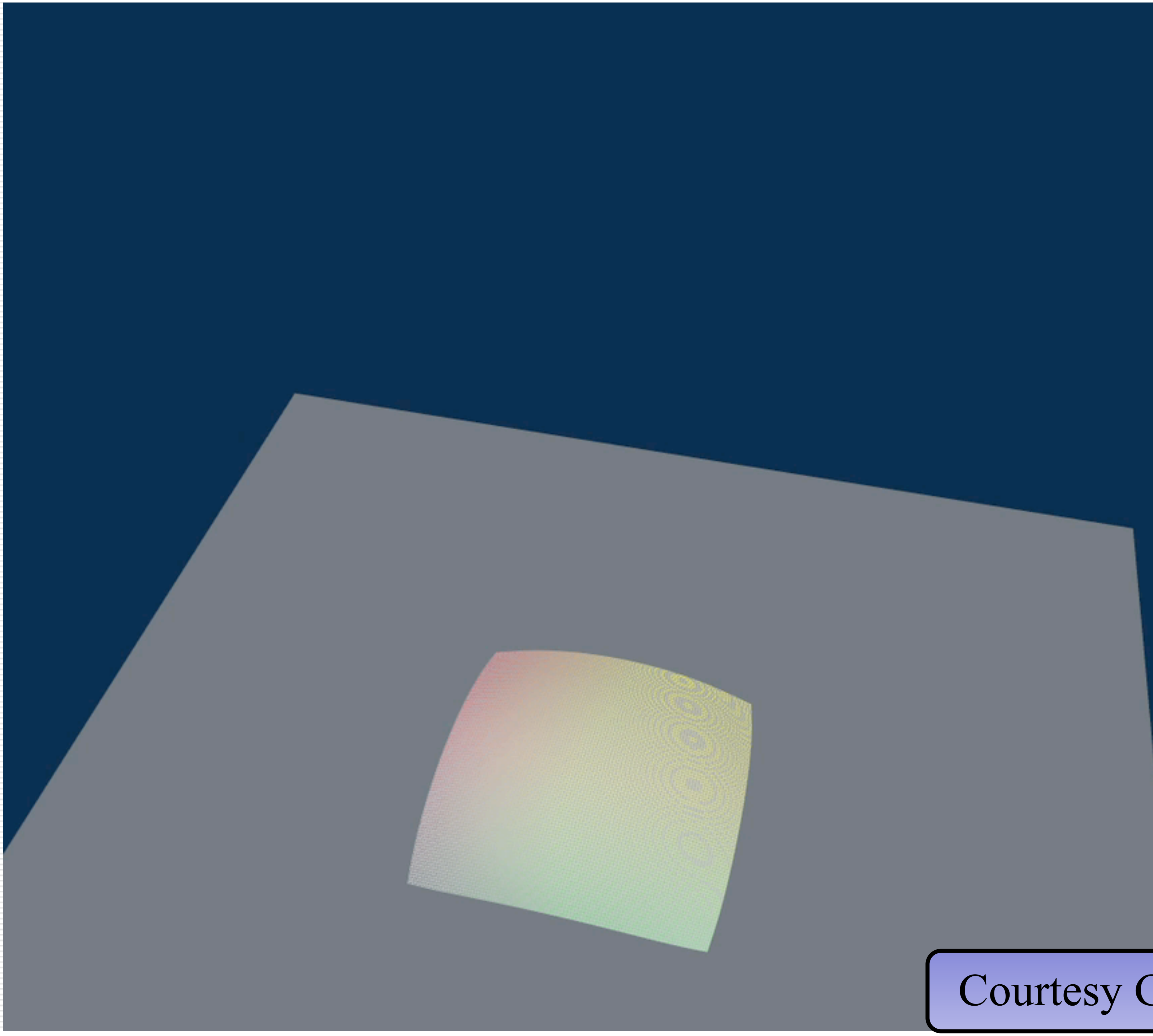
Courtesy Garth

Sets of Streamlines

- Stream surface examples



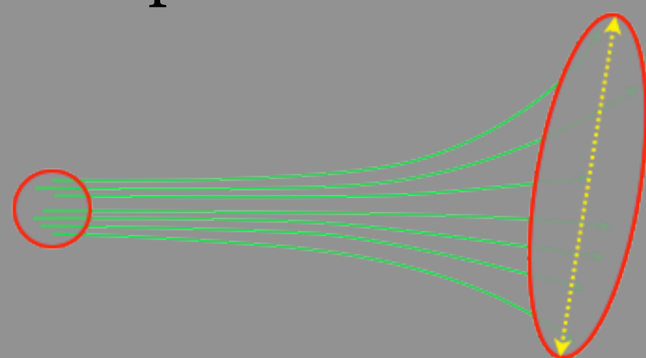
Courtesy Garth



Courtesy Garth

Lagrangian Methods

- Visualize manifolds of maximal stretching in a flow, as indicated by dense particles



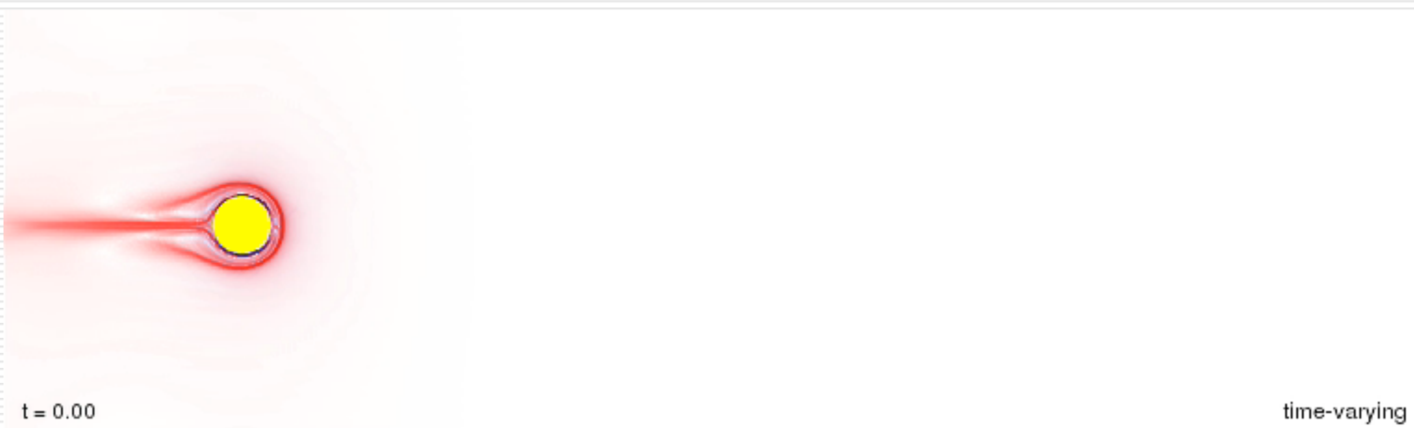
- Finite-Time Lyapunov Exponent (FTLE)

$$\sigma_{\Delta t}(t,x) := \frac{1}{\Delta t} \ln \sqrt{\lambda_{max}\left(D_x\ \phi_{\Delta t}(t,x)\right)}$$

Courtesy Garth

Lagrangian Methods

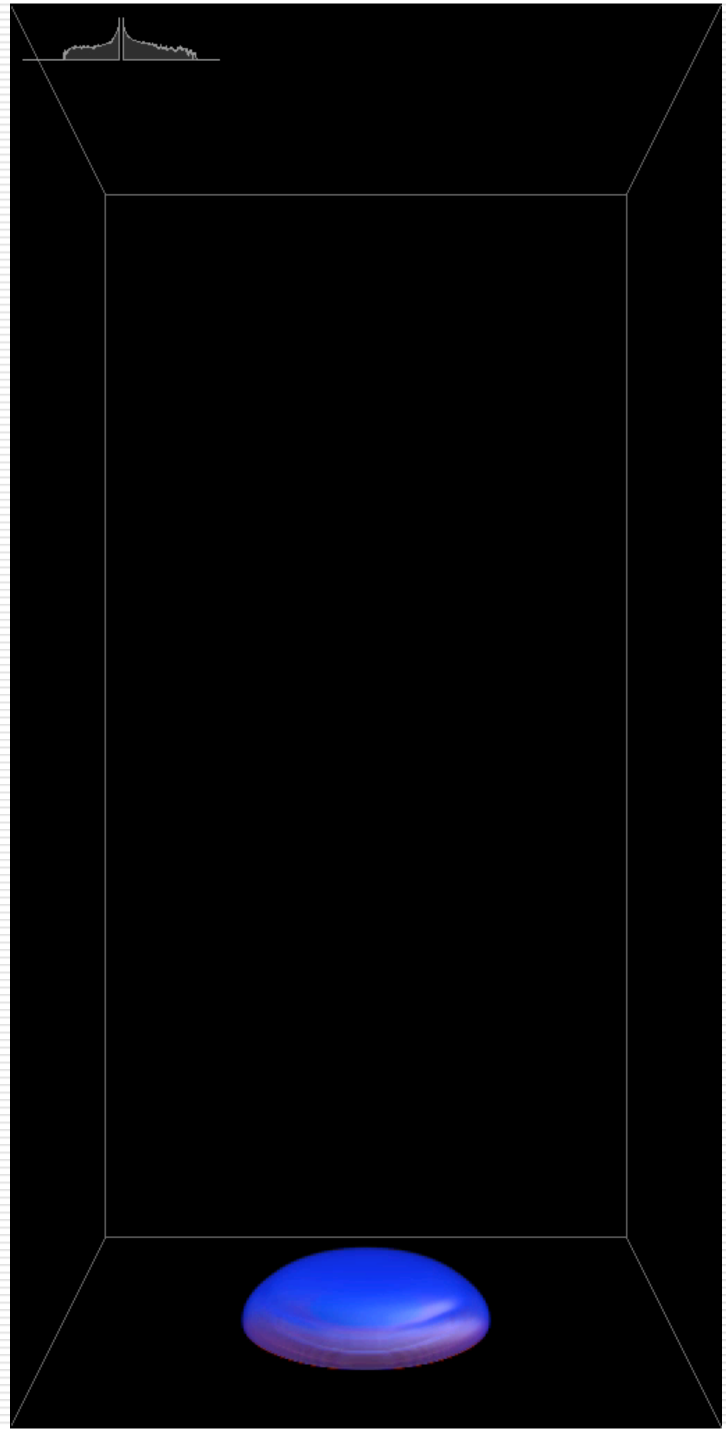
- Visualize manifolds of maximal stretching in a flow, as indicated by dense particles
  - Forward in time: FTLE+ indicates divergence
  - Backward in time: FTLE- indicates convergence



Courtesy Garth

Lagrangian Methods

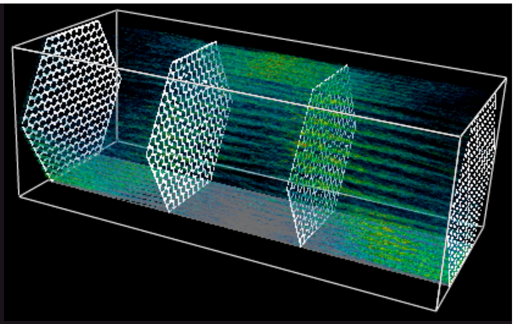
- FTLE example



Jet Flow Turbulence

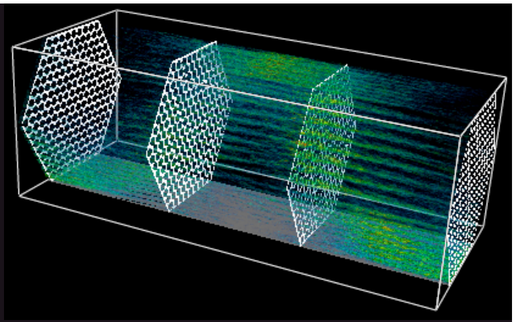
Courtesy Garth

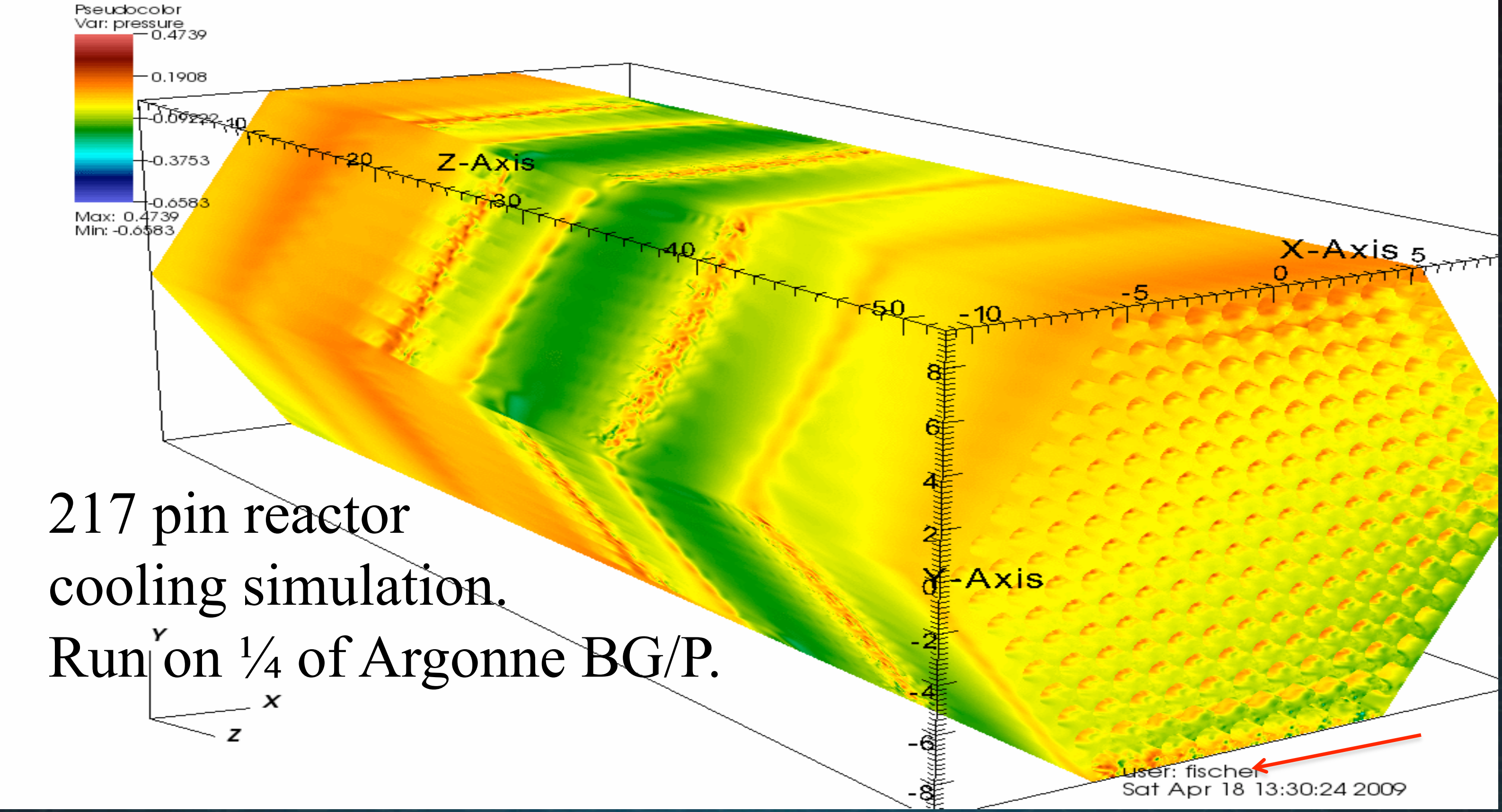
Particle Advection for Very Large Data Sets



- Do we need advanced parallelization schemes for particle advections of large data sets?
  - (Yes)
- Why is it hard?
- How to parallelize particle advections?
  - Over particles…
  - Over data…
  - Other?

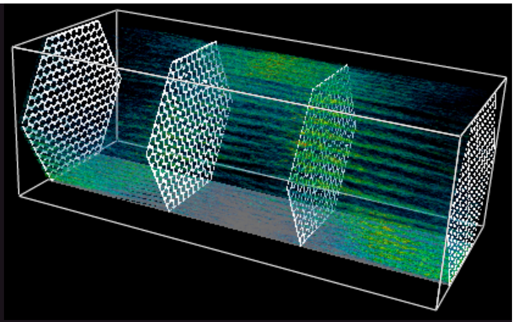
Flow analysis for 217 pin simulation / 1 billion grid points





217 pin reactor cooling simulation.
Run on ¼ of Argonne BG/P.

Flow analysis for 217 pin simulation / 1 billion grid points



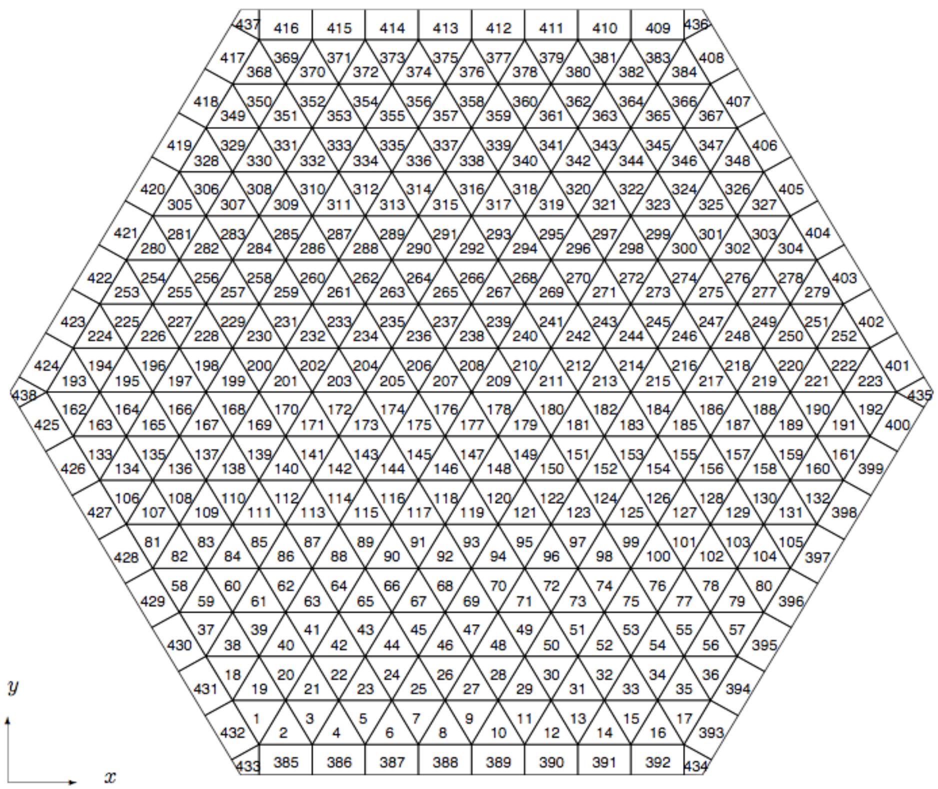
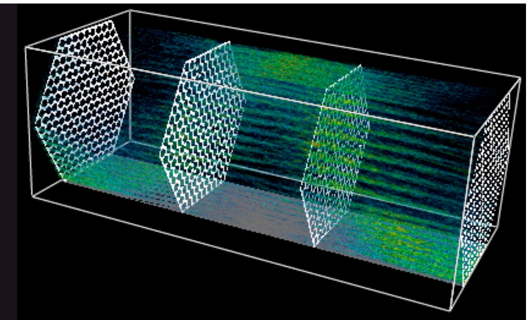
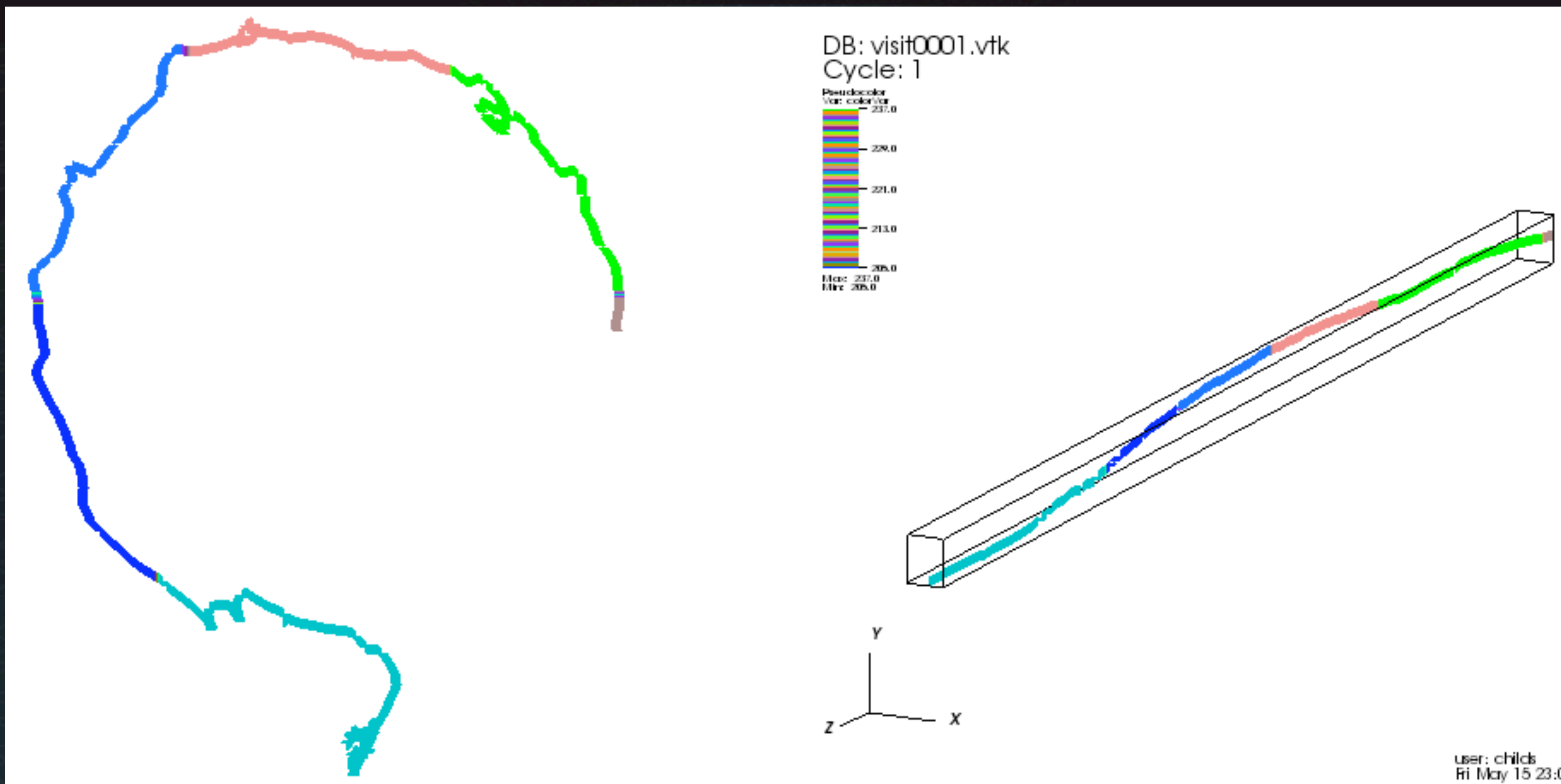


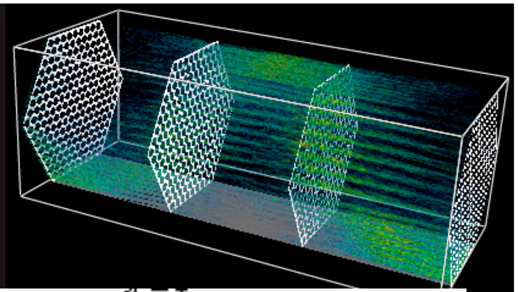
Figure 1: cells_only.ps

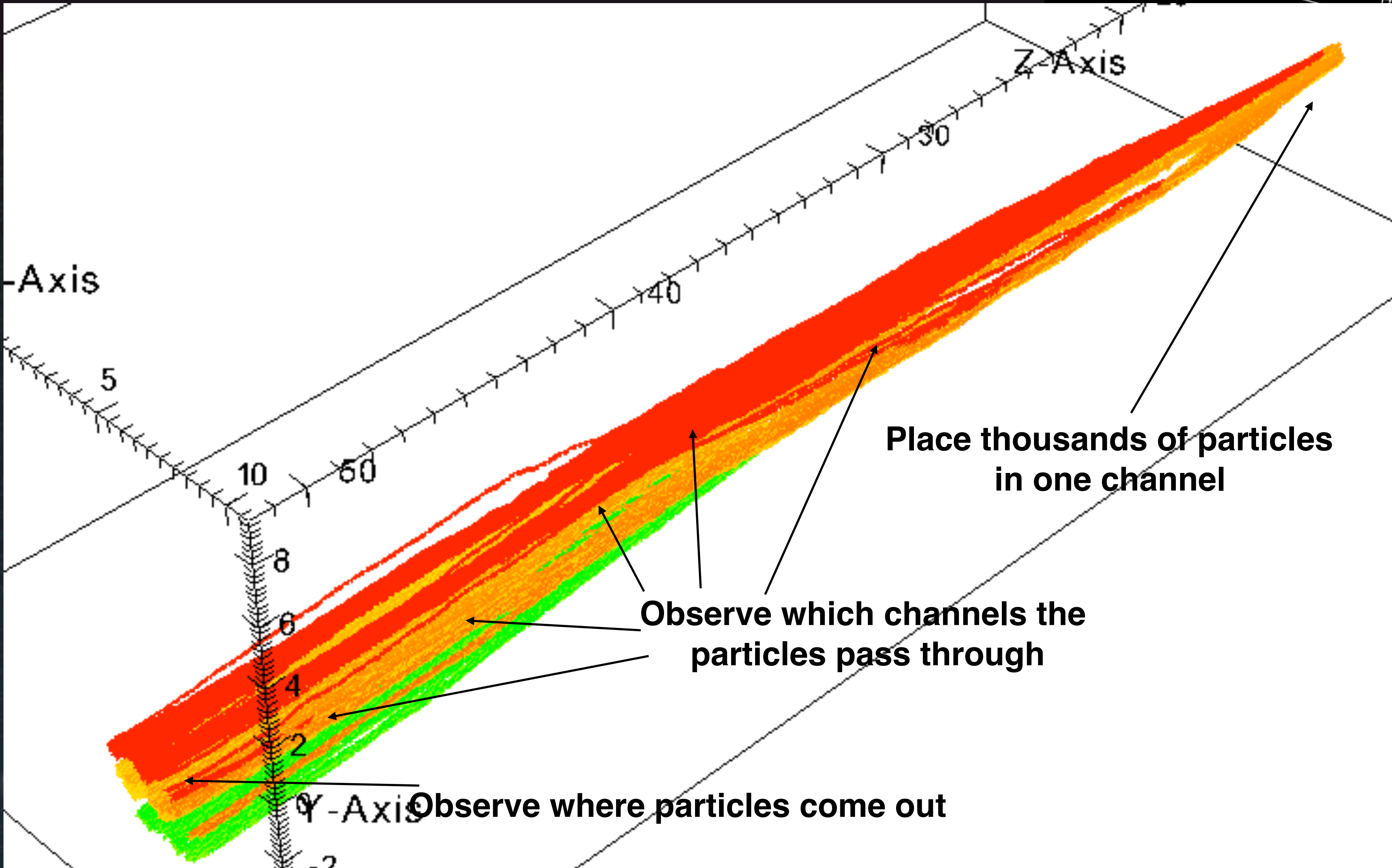
Tracing particles through channels



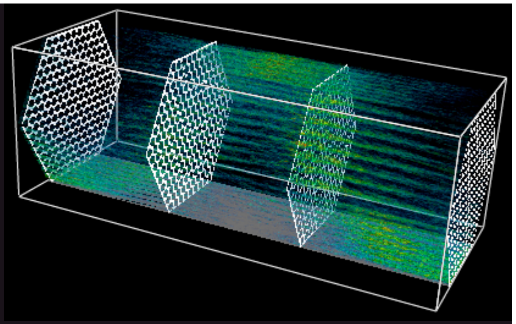


Tracing particles through channels

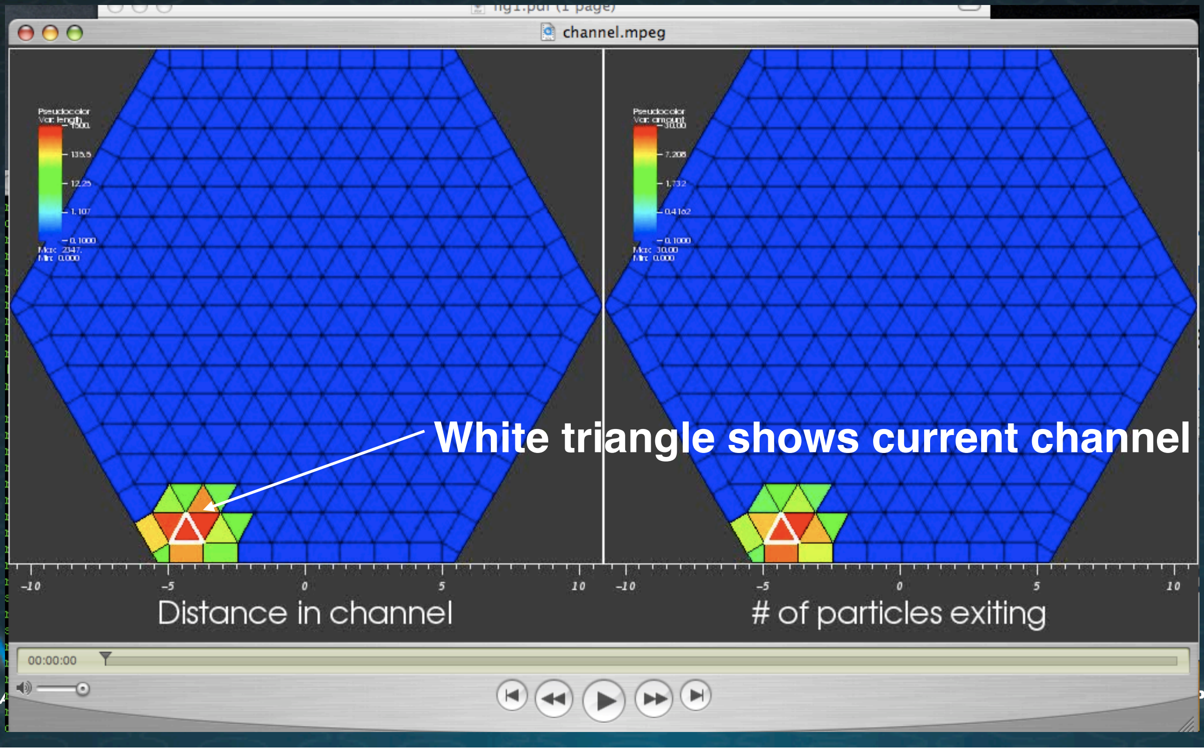




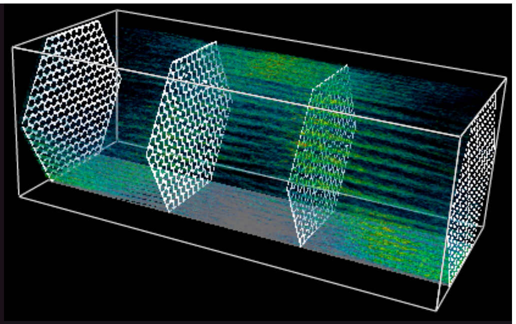
Tracing particles through the channels



- Two different "matrices" to describe flow from channel I to channel J
  - Exit location versus travel time in channel
  - Issues: pathlines vs streamlines, 12X vs A^{12}

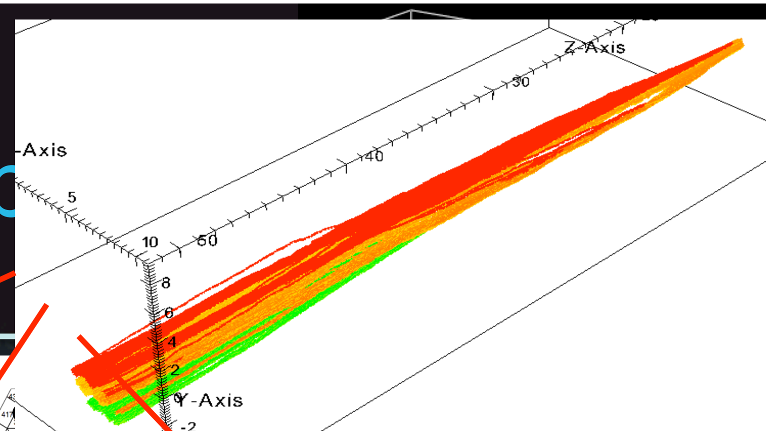


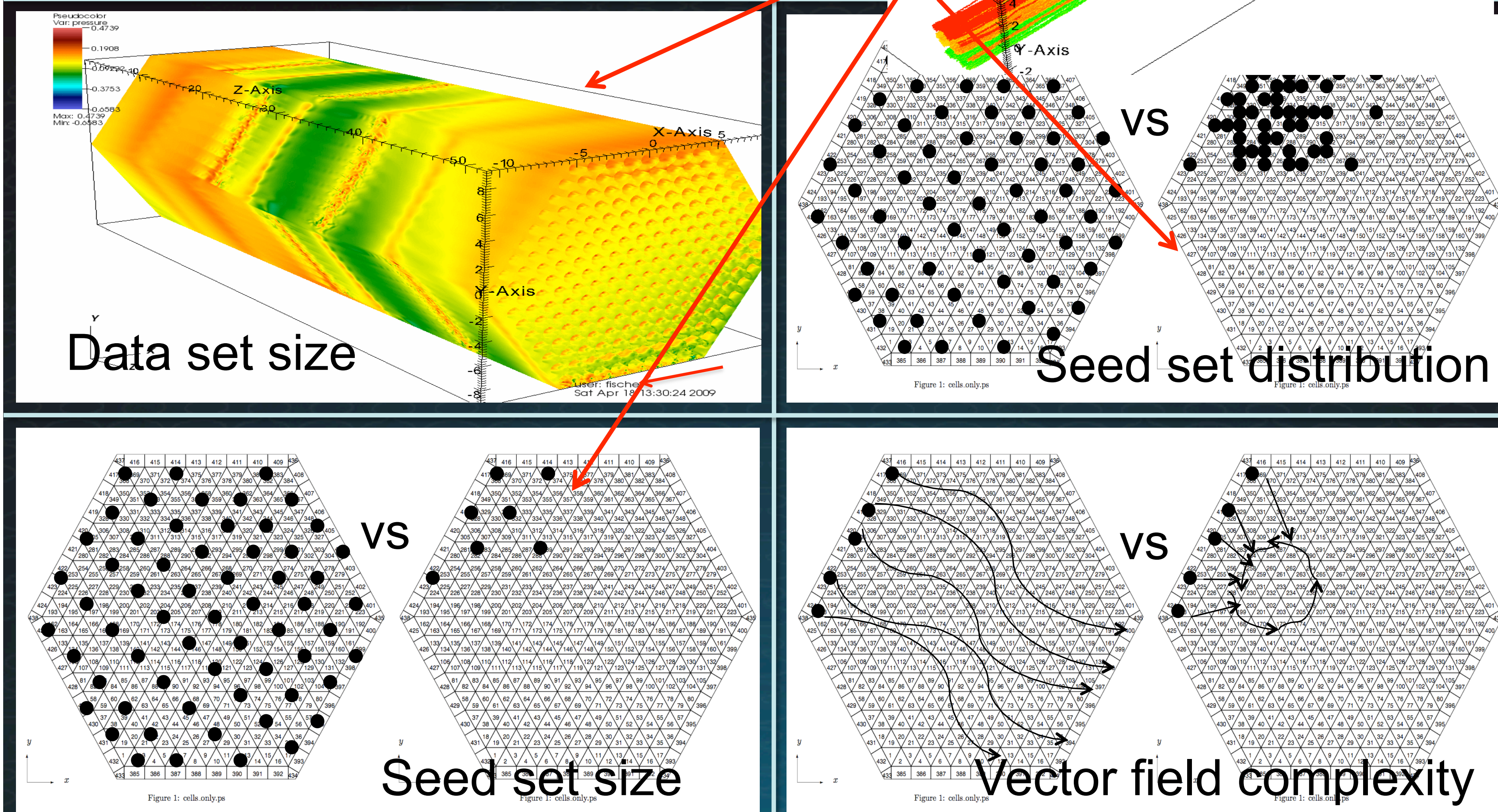
Particle Advection for Very Large Data Sets



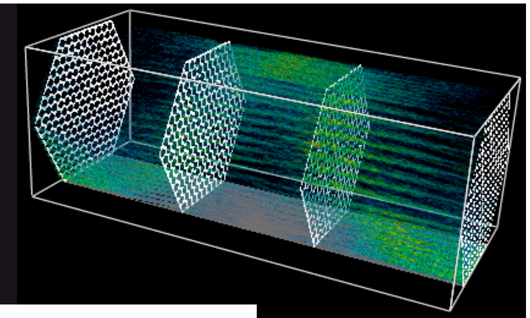
- Do we need advanced parallelization schemes for particle advections of large data sets?
  - (Yes)
- Why is it hard?
- How to parallelize particle advections?
  - Over particles…
  - Over data…
  - Other?

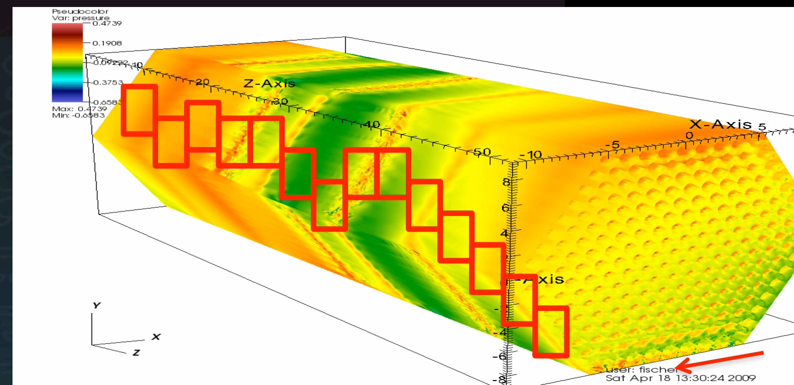
Four dimensions of comp





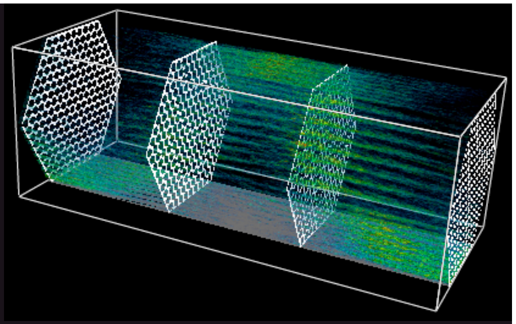
Why do we need advanced parallelization techniques?





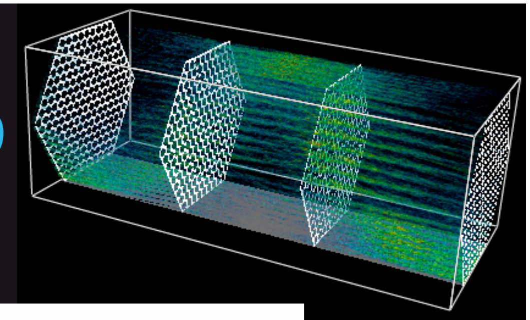
- Data set size?
  - Not enough!
- Large #'s of particles?
  - Need to parallelize, but embarrassingly parallel OK
- Large #'s of particles + large data sets sizes
  - Need to parallelize, simple schemes may be OK
- Large #'s of particles + large data set sizes + (bad distribution OR complex vector field)
  - Need smart parallelization

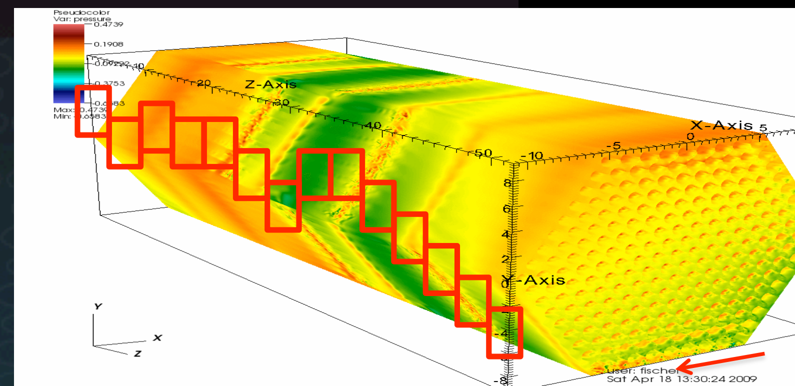
Outline



- Do we need advanced parallelization schemes for particle advections of large data sets?
  - (Yes)
- Why is it hard?
- How to parallelize particle advections?
  - Over particles…
  - Over data…
  - Other?

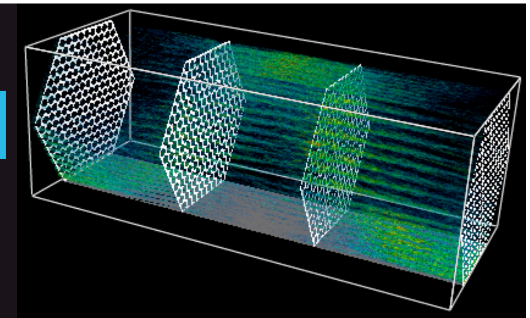
Three types of parallelization to consider.

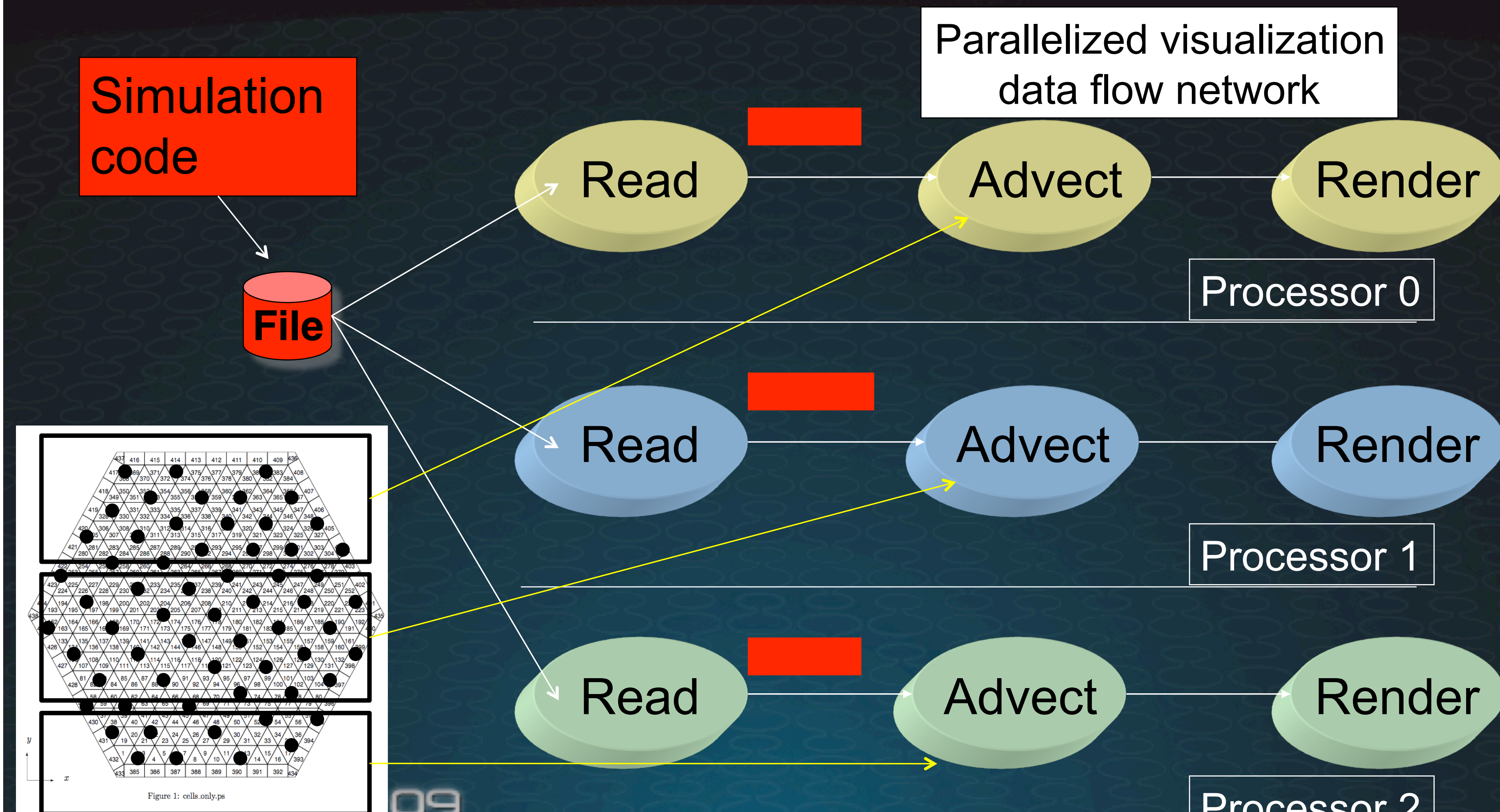




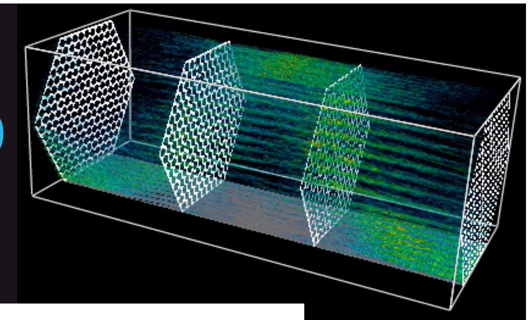
- Data set size?
  - Not enough!
- #1: Large #'s of particles?
  - Need to parallelize, but embarrassingly parallel OK
- #2: Large #'s of particles + large data sets sizes
  - Need to parallelize, simple schemes may be OK
- #3: Large #'s of particles + large data set sizes + (bad distribution OR complex vector field)
  - Need smart parallelization

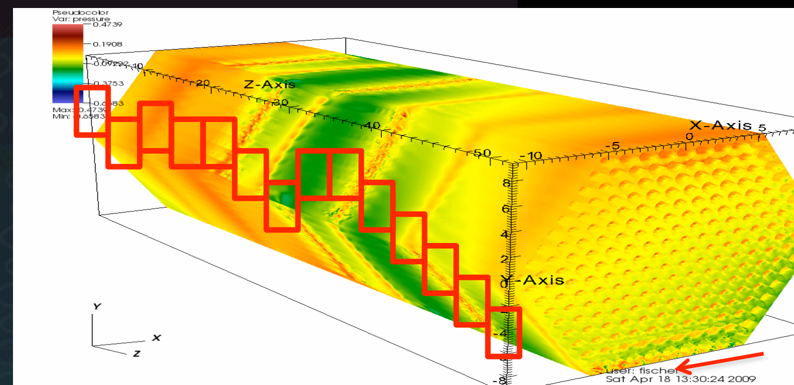
Parallelization for small data and a large number of particles.





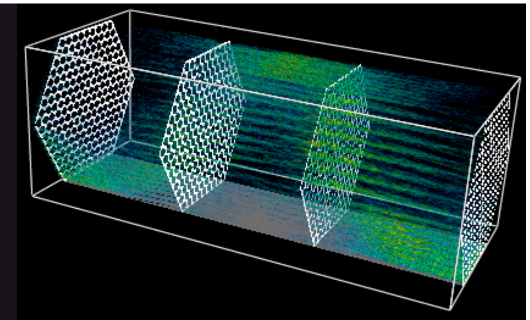
Three types of parallelization to consider.

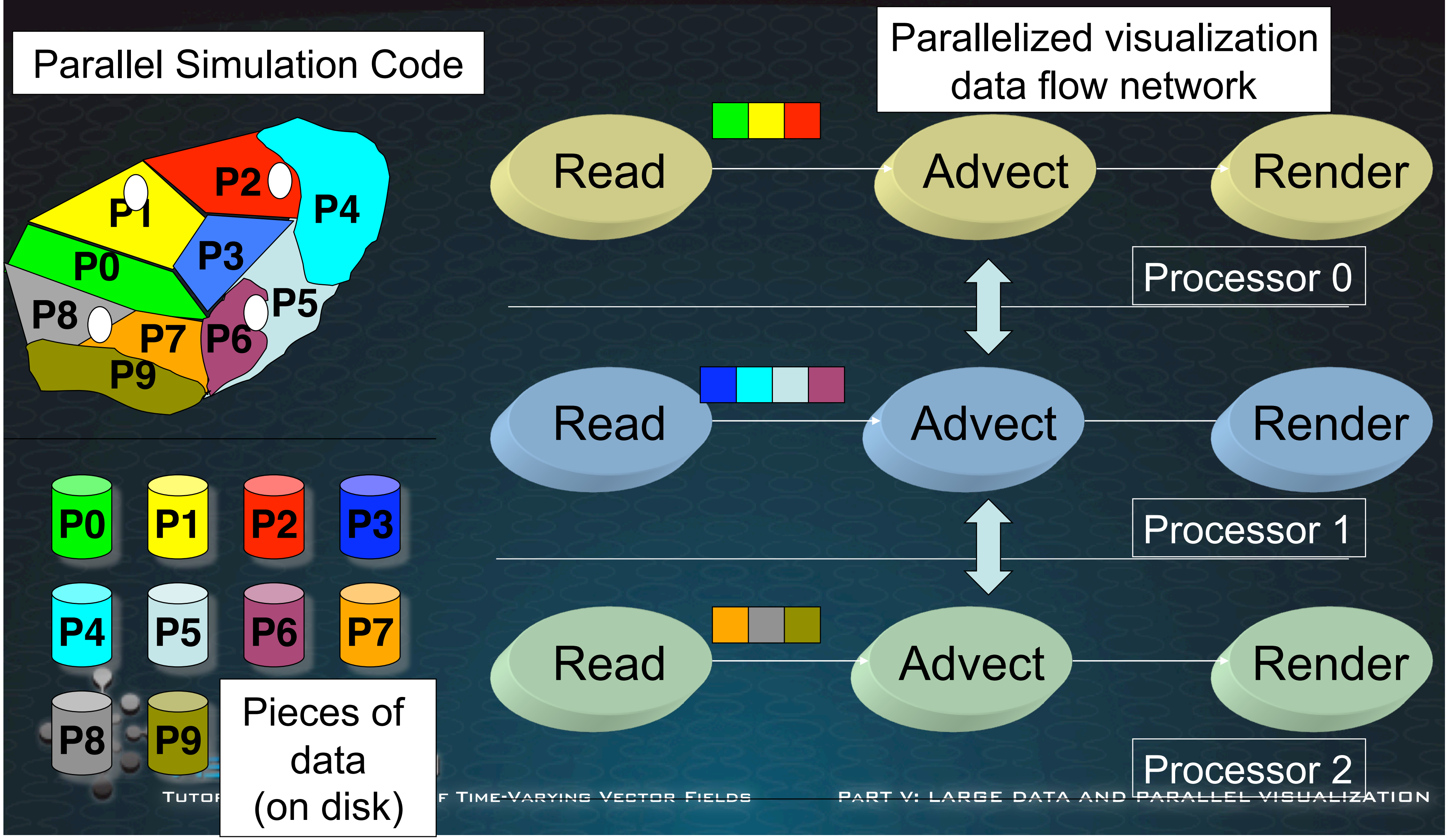




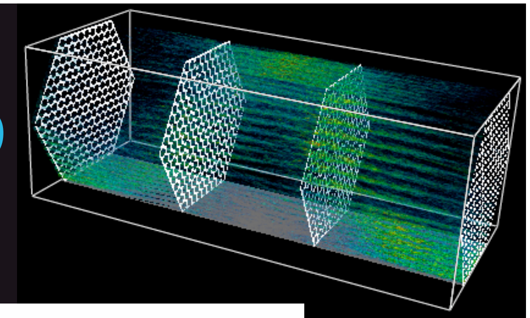
- Data set size?
  - Not enough!
- #1: Large #'s of particles?
  - Need to parallelize, but embarrassingly parallel OK
- #2: Large #'s of particles + large data sets sizes
  - Need to parallelize, simple schemes may be OK
- #3: Large #'s of particles + large data set sizes + (bad distribution OR complex vector field)
  - Need smart parallelization

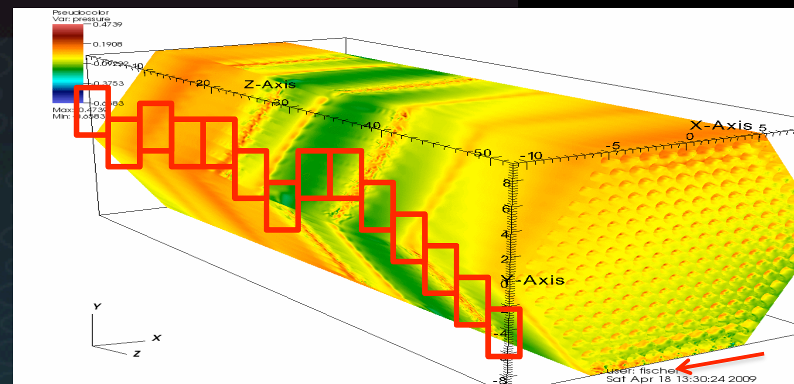
Parallelization for large data with good "distribution"





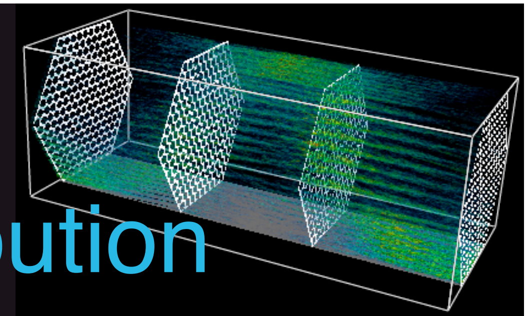
Three types of parallelization to consider.



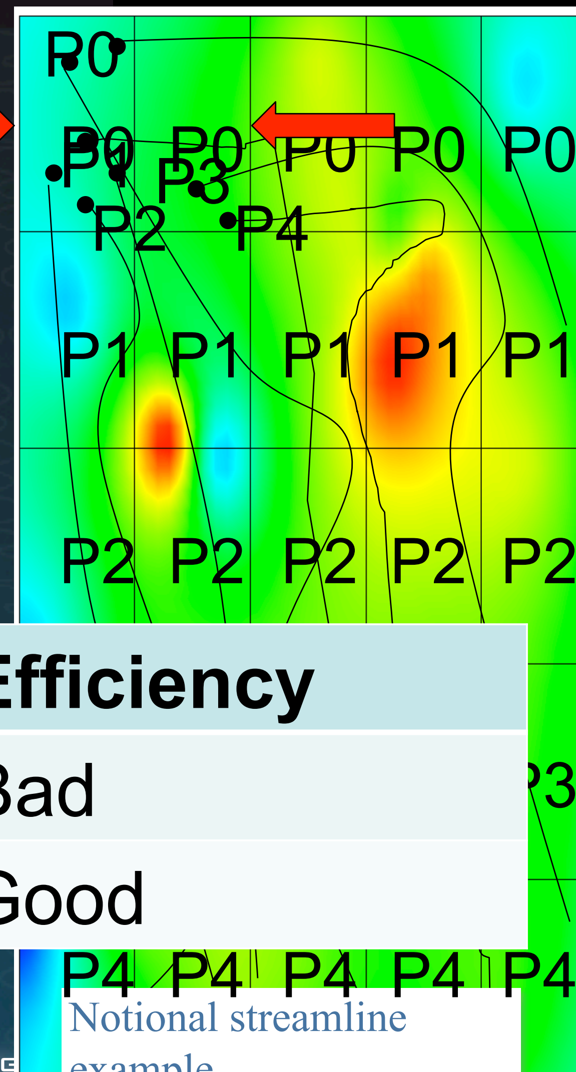


- Data set size?
  - Not enough!
- #1: Large #'s of particles?
  - Need to parallelize, but embarrassingly parallel OK
- #2: Large #'s of particles + large data sets sizes
  - Need to parallelize, simple schemes may be OK
- #3: Large #'s of particles + large data set sizes + (bad distribution OR complex vector field)
  - Need smart parallelization

Parallelization with big data & lots of seed points & bad distribution



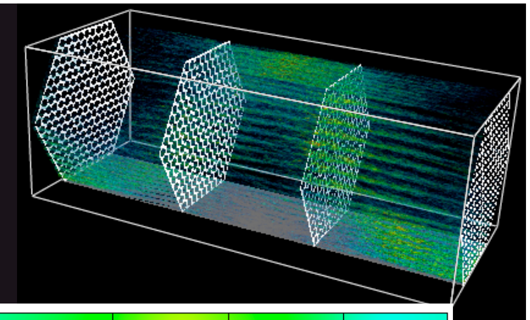
- Two extremes:
  - Partition data over processors and pass particles amongst processors
    - Parallel inefficiency!
  - Partition seed points over processors and process necessary data for advection



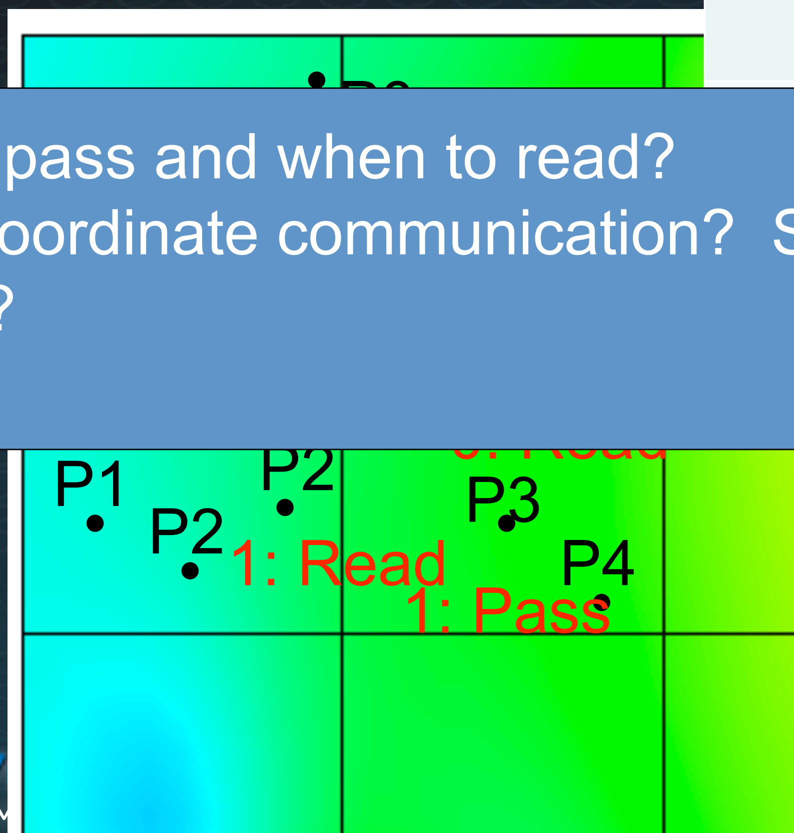
Notional streamline example

Parallelizing Over	I/O	Efficiency
Data	Good	Bad
Particles	Bad	Good

Hybrid solution: balance between tensions on I/O and parallel efficiency



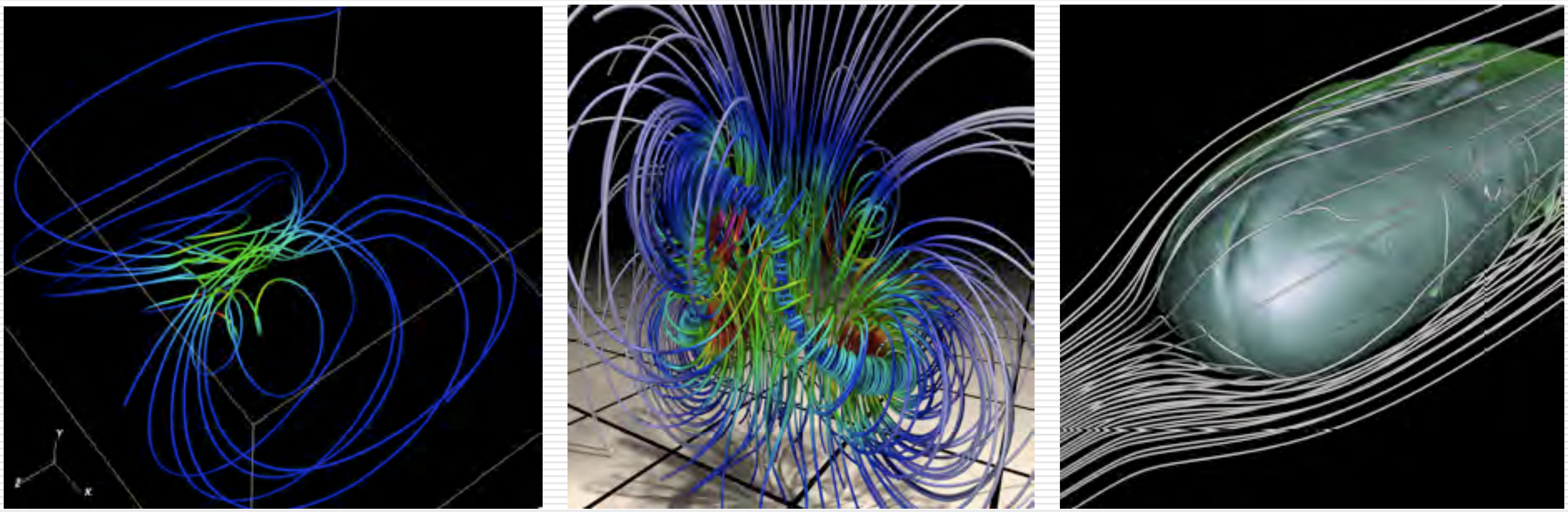
Iteration	Action
0	P0 reads B0, P3 reads B1



- When to pass and when to read?
- How to coordinate communication? Status? Efficiently?

Notional streamline example

AMR Particle Advection



Courtesy Deines, Garth, Weber & Childs

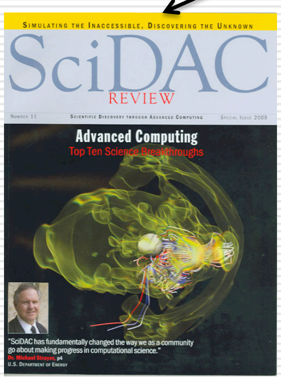
This work is the effort of many people from VACET, both inside and outside IDAV.

Flexible serial streamline library

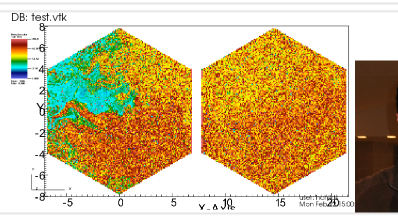
Research effort: Efficient parallel streamline generation

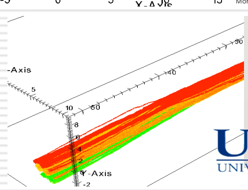
Deployment effort in VisIt

Hybrid Parallelism

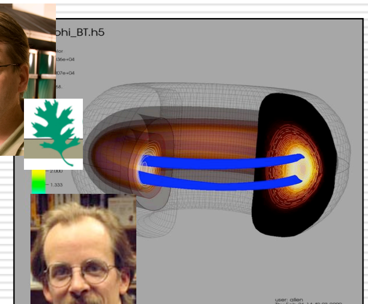


Usage by VisIt community

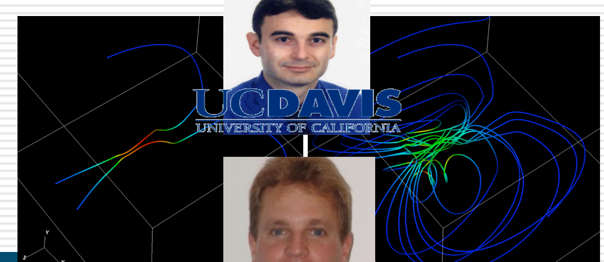




Nuclear Energy



Fusion



Astrophysics: radiation patterns

What's publicly available in VisIt now?

- Only about 20% of what was discussed
  - Streamlines
  - All 3 parallel algorithms
- What's not ready yet?
  - Pathlines
  - Stream surfaces
  - FTLE
- Analysis
  - Currently the analysis described is easy to do, but requires a "VisIt buddy"
  - Long term, we'd like to open up arbitrary analysis of integral curves (likely via Python)

What controls are available for particle advection?

- How to evaluate / interpolate?
- How to advect? (e.g. Dormand-Prince / Adams-Bashforth)
- How to parallelize? (e.g. three algorithms)
- Where to place seed points?
- How to analyze the curve
  - Residence time
  - FTLE
  - Poincare analysis
  - Streamlines / pathlines
  - …

Summary

- Visualizing large scale data presents incredible challenges in both managing scale and data understanding.
- IDAV portfolio contains research query-driven vis, GPU algorithms, function data, embedded boundaries, and particle advection
- Particle advection is:
  - A powerful tool for understanding vector data and flow
  - Difficult to parallelize efficiently for large data

- Hank Childs, hrchilds@ucdavis.edu / hchilds@lbl.gov