

Supercomputing in Plain English

Part XI: Grab Bag:

Scientific Libraries, I/O, Visualization

Henry Neeman, Director

**OU Supercomputing Center for Education & Research
University of Oklahoma Information Technology**

Tuesday May 5 2009





This is an experiment!

It's the nature of these kinds of videoconferences that
FAILURES ARE GUARANTEED TO HAPPEN!
NO PROMISES!

So, please bear with us. Hopefully everything will work out well enough.

If you lose your connection, you can retry the same kind of connection, or try connecting another way.

Remember, if all else fails, you always have the toll free phone bridge to fall back on.





Access Grid

This week's Access Grid (AG) venue: Titan.

If you aren't sure whether you have AG, you probably don't.

Tue May 5	Titan
-----------	-------

Many thanks to
John Chapman of
U Arkansas for
setting these up
for us.





H.323 (Polycom etc)

If you want to use H.323 videoconferencing – for example, Polycom – then dial

69.77.7.203##12345

any time after 2:00pm. Please connect early, at least today.

For assistance, contact Andy Fleming of KanREN/Kan-ed (afleming@kanren.net or 785-230-2513).

KanREN/Kan-ed's H.323 system can handle up to 40 simultaneous H.323 connections. If you cannot connect, it may be that all 40 are already in use.

Many thanks to Andy and KanREN/Kan-ed for providing H.323 access.





iLinc

We have unlimited simultaneous iLinc connections available.

If you're already on the SiPE e-mail list, then you should already have an e-mail about iLinc. Your personal URL will always be the same.

If you want to use iLinc, please follow the directions in the iLinc e-mail.

For iLinc, you **MUST** use either Windows (XP strongly preferred) or MacOS X with Internet Explorer.

To use iLinc, you'll need to download a client program to your PC. It's free, and setup should take only a few minutes.

Many thanks to Katherine Kantardjieff of California State U Fullerton for providing the iLinc licenses.





QuickTime Broadcaster

If you cannot connect via the Access Grid, H.323 or iLinc, then you can connect via QuickTime:

`rtsp://129.15.254.141/test_hpc09.sdp`

We recommend using QuickTime Player for this, because we've tested it successfully.

We recommend upgrading to the latest version at:

<http://www.apple.com/quicktime/>

When you run QuickTime Player, traverse the menus

File -> Open URL

Then paste in the rstp URL into the textbox, and click OK.

Many thanks to Kevin Blake of OU for setting up QuickTime Broadcaster for us.





Phone Bridge

If all else fails, you can call into our toll free phone bridge:

1-866-285-7778, access code 6483137#

Please mute yourself and use the phone to listen.

Don't worry, we'll call out slide numbers as we go.

Please use the phone bridge **ONLY** if you cannot connect any other way: the phone bridge is charged per connection per minute, so our preference is to minimize the number of connections.

Many thanks to Amy Apon and U Arkansas for providing the toll free phone bridge.





Please Mute Yourself

No matter how you connect, please mute yourself, so that we cannot hear you.

At OU, we will turn off the sound on all conferencing technologies.

That way, we won't have problems with echo cancellation.

Of course, that means we cannot hear questions.

So for questions, you'll need to send some kind of text.

Also, if you're on iLinc: **SIT ON YOUR HANDS!**

Please DON'T touch ANYTHING!





Questions via Text: iLinc or E-mail

Ask questions via text, using one of the following:

- iLinc's text messaging facility;
- e-mail to sipe2009@gmail.com.

All questions will be read out loud and then answered out loud.





Thanks for helping!

- OSCER operations staff (Brandon George, Dave Akin, Brett Zimmerman, Josh Alexander)
- OU Research Campus staff (Patrick Calhoun, Josh Maxey, Gabe Wingfield)
- Kevin Blake, OU IT (videographer)
- Katherine Kantardjieff, CSU Fullerton
- John Chapman and Amy Apon, U Arkansas
- Andy Fleming, KanREN/Kan-ed
- This material is based upon work supported by the National Science Foundation under Grant No. OCI-0636427, “CI-TEAM Demonstration: Cyberinfrastructure Education for Bioinformatics and Beyond.”





This is an experiment!

It's the nature of these kinds of videoconferences that
FAILURES ARE GUARANTEED TO HAPPEN!
NO PROMISES!

So, please bear with us. Hopefully everything will work out well enough.

If you lose your connection, you can retry the same kind of connection, or try connecting another way.

Remember, if all else fails, you always have the toll free phone bridge to fall back on.





Supercomputing Exercises

Want to do the “Supercomputing in Plain English” exercises?

- The first several exercises are already posted at:

<http://www.oscer.ou.edu/education.php>

- If you don’t yet have a supercomputer account, you can get a temporary account, just for the “Supercomputing in Plain English” exercises, by sending e-mail to:

hneeman@ou.edu

Please note that this account is for doing the **exercises only**, and will be shut down at the end of the series.





OK Supercomputing Symposium 2009



2003 Keynote:
Peter Freeman
NSF
Computer &
Information
Science &
Engineering
Assistant Director



2004 Keynote:
Sangtae Kim
NSF Shared
Cyberinfrastructure
Division Director



2005 Keynote:
Walt Brooks
NASA Advanced
Supercomputing
Division Director



2006 Keynote:
Dan Atkins
Head of NSF's
Office of
Cyber-
infrastructure



2007 Keynote:
Jay Boisseau
Director
Texas Advanced
Computing Center
U. Texas Austin



2008 Keynote:
José Muñoz
Deputy Office
Director/ Senior
Scientific Advisor
Office of Cyber-
infrastructure
National Science
Foundation



2009 Keynote:
Ed Seidel
Director
NSF Office of
Cyber-
infrastructure

FREE! Wed Oct 7 2009 @ OU
<http://symposium2009.oscer.ou.edu/>

Parallel Programming Workshop
FREE! Tue Oct 6 2009 @ OU

Registration
is OPEN!

Sponsored by SC09 Education Program
FREE! Symposium Wed Oct 7 2009 @ OU

Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009





SC09 Summer Workshops

This coming summer, the SC09 Education Program, part of the SC09 (Supercomputing 2009) conference, is planning to hold two weeklong supercomputing-related workshops in Oklahoma, for **FREE** (except you pay your own transport):

- **At OSU Sun May 17 – the May 23:**
FREE Computational Chemistry for Chemistry Educators (2010 TENTATIVE: Computational Biology)
- **At OU Sun Aug 9 – Sat Aug 15:**
FREE Parallel Programming & Cluster Computing

We'll alert everyone when the details have been ironed out and the registration webpage opens.

Please note that you must **apply** for a seat, and acceptance **CANNOT** be guaranteed.



Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009



SC09 Summer Workshops

1. May 17-23: Oklahoma State U: Computational Chemistry
2. May 25-30: Calvin Coll (MI): Intro to Computational Thinking
3. June 7-13: U Cal Merced: Computational Biology
4. June 7-13: Kean U (NJ): Parallel Progrmg & Cluster Comp
5. June 14-20: Widener U (PA): Computational Physics
6. July 5-11: Atlanta U Ctr: Intro to Computational Thinking
7. July 5-11: Louisiana State U: Parallel Progrmg & Cluster Comp
8. July 12-18: U Florida: Computational Thinking Grades 6-12
9. July 12-18: Ohio Supercomp Ctr: Computational Engineering
10. Aug 2- 8: U Arkansas: Intro to Computational Thinking
11. Aug 9-15: U Oklahoma: Parallel Progrmg & Cluster Comp





Outline

- Scientific Computing Pipeline
- Scientific Libraries
- I/O Libraries
- Scientific Visualization



Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009



Scientific Computing Pipeline

Real World

Physics

Mathematical Representation (continuous)

Numerical Representation (discrete)

Algorithm

Implementation (program)

Port (to a specific platform)

Result (run)

Analysis

Verification

Thanks to Julia Mullen of MIT Lincoln Lab for this concept.

Supercomputing in Plain English: Grab Bag

Tuesday May 5 2009





Five Rules of Scientific Computing

1. **Know** the physics.
2. **Control** the software.
3. **Understand** the numerics.
4. **Achieve** expected behavior.
5. **Question** unexpected behavior.

Thanks to Robert E. Peterkin for these.



Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009



Scientific Libraries



Preinvented Wheels

Many simulations perform fairly common tasks; for example, solving systems of equations:

$$\mathbf{Ax} = \mathbf{b}$$

where \mathbf{A} is the matrix of coefficients, \mathbf{x} is the vector of unknowns and \mathbf{b} is the vector of knowns.

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,n} \\ a_{3,1} & a_{3,2} & a_{3,3} & \cdots & a_{3,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & a_{n,3} & \cdots & a_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{bmatrix}$$



Scientific Libraries

Because some tasks are quite common across many science and engineering applications, groups of researchers have put a lot of effort into writing *scientific libraries*: collections of routines for performing these commonly-used tasks (for example, linear algebra solvers).

The people who write these libraries know a lot more about these things than we do.

So, a good strategy is to use their libraries, rather than trying to write our own.





Solver Libraries

Probably the most common scientific computing task is solving a system of equations

$$\mathbf{Ax} = \mathbf{b}$$

where \mathbf{A} is a matrix of coefficients, \mathbf{x} is a vector of unknowns, and \mathbf{b} is a vector of knowns.

The goal is to solve for \mathbf{x} .





Solving Systems of Equations

Don'ts:

- **Don't** invert the matrix ($\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$). That's much more costly than solving directly, and much more prone to numerical error.
- **Don't** write your own solver code. There are people who devote their whole careers to writing solvers. They know a lot more about writing solvers than we do.





Solving Do's

Do's:

- Do use standard, portable solver libraries.
- Do use a version that's tuned for the platform you're running on, if available.
- Do use the information that you have about your system of equations to pick the most efficient solver.

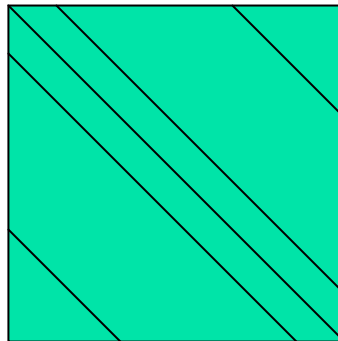


All About Your Matrix

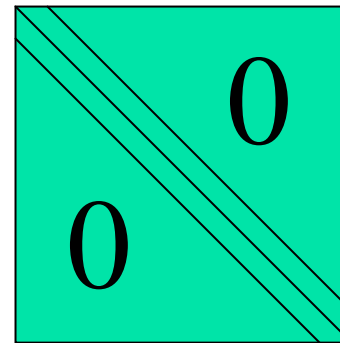
If you know things about your matrix, you maybe can use a more efficient solver.

- Symmetric: $a_{i,j} = a_{j,i}$
- Positive definite: $\mathbf{x}^T \mathbf{A} \mathbf{x} > 0$ for all $\mathbf{x} \neq 0$
(for example, if all eigenvalues are positive)
- Banded:

zero
except
on the
bands



- Tridiagonal:

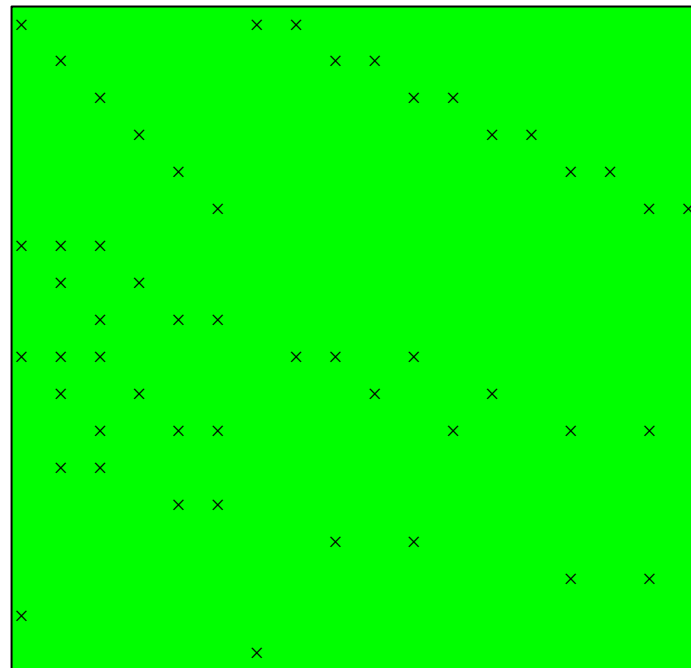


and ...



Sparse Matrices

A sparse matrix is a matrix that has mostly zeros in it. “Mostly” is vaguely defined, but a good rule of thumb is that a matrix is sparse if more than, say, 90-95% of its entries are zero. (A non-sparse matrix is dense.)





Linear Algebra Libraries

- BLAS [1],[2]
- ATLAS^[3]
- LAPACK^[4]
- ScaLAPACK^[5]
- PETSc^{[6],[7],[8]}





BLAS

The **Basic Linear Algebra Subprograms** (BLAS) are a set of low level linear algebra routines:

- Level 1: Vector-vector (for example, dot product)
- Level 2: Matrix-vector (for example, matrix-vector multiply)
- Level 3: Matrix-matrix (for example, matrix-matrix multiply)

Many linear algebra packages, including LAPACK, ScaLAPACK and PETSc, are built on top of BLAS.

Most supercomputer vendors have versions of BLAS that are highly tuned for their platforms.



ATLAS

The **Automatically Tuned Linear Algebra Software** package (ATLAS) is a self-tuned version of BLAS (it also includes a few LAPACK routines).

When it's installed, it tests and times a variety of approaches to each routine, and selects the version that runs the fastest.

ATLAS is substantially faster than the generic version of BLAS.

And, it's **FREE!**



Goto BLAS

In the past few years, a new version of BLAS has been released, developed by Kazushige Goto (currently at UT Austin).

This version is unusual, because instead of optimizing for cache, it optimizes for the **Translation Lookaside Buffer** (TLB), which is a special little cache that often is ignored by software developers.

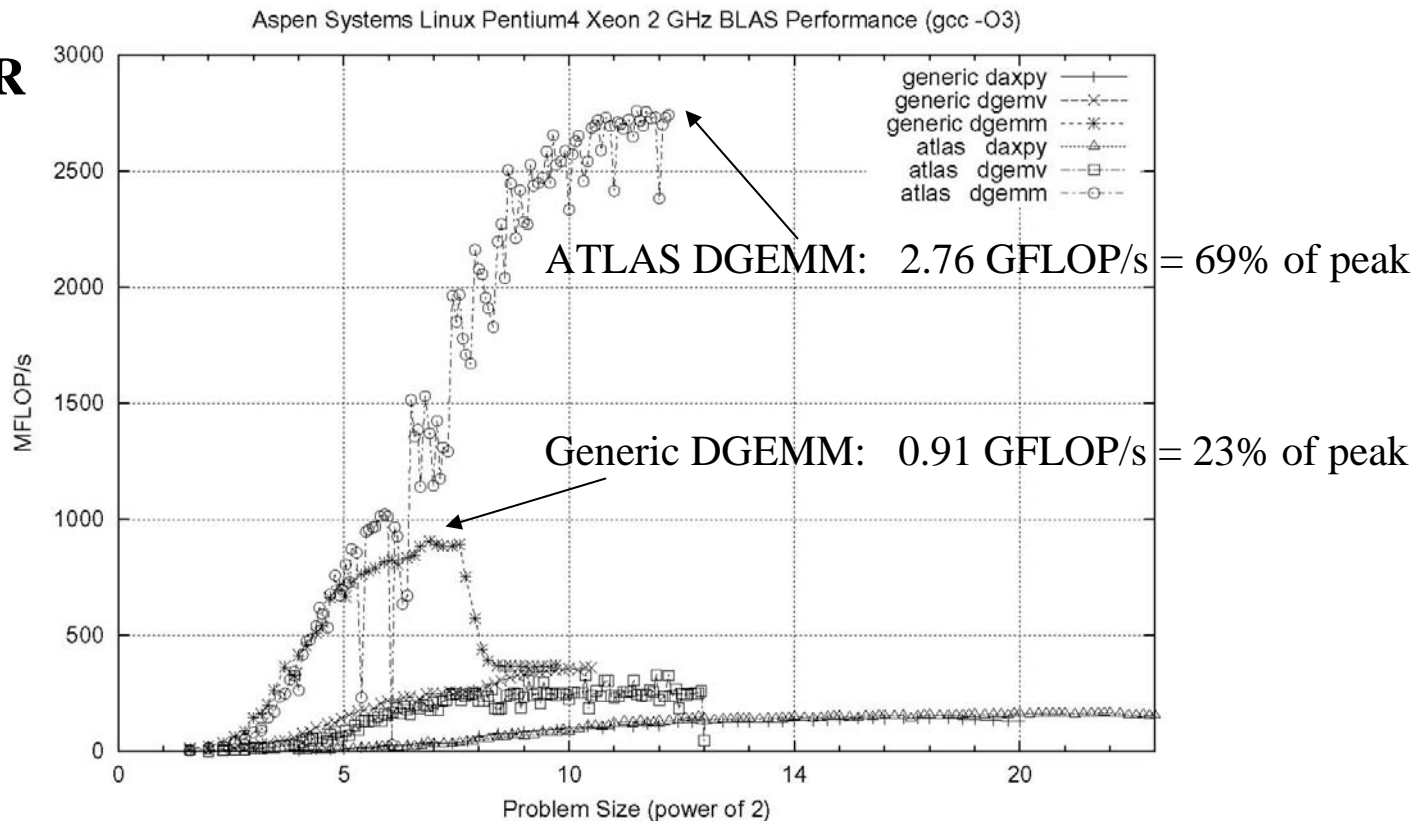
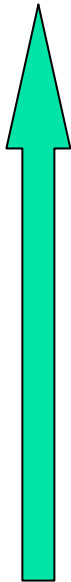
Goto realized that optimizing for the TLB would be more efficient than optimizing for cache.





ATLAS vs. Generic BLAS

BETTER



DGEMM: Double precision **GE**neral **M**atrix-**M**atrix multiply

DGEMV: Double precision **GE**neral **M**atrix-**V**ector multiply

Supercomputing in Plain English: Grab Bag

Tuesday May 5 2009



LAPACK

LAPACK (Linear Algebra PACKage) solves dense or special-case sparse systems of equations depending on matrix properties such as:

- Precision: single, double
- Data type: real, complex
- Shape: diagonal, bidiagonal, tridiagonal, banded, triangular, trapezoidal, Hessenberg, general dense
- Properties: orthogonal, positive definite, Hermetian (complex), symmetric, general

LAPACK is built on top of BLAS, which means it can benefit from ATLAS.



LAPACK Example

```
REAL,DIMENSION(numrows,numcols) :: A
REAL,DIMENSION(numrows)          :: B
REAL,DIMENSION(numcols)          :: X
INTEGER,DIMENSION(numrows)       :: pivot
INTEGER :: row, col, info, numrhs = 1
DO row = 1, numrows
  B(row) = ...
END DO
DO col = 1, numcols
  DO row = 1, numrows
    A(row,col) = ...
  END DO
END DO
CALL sgesv(numrows, numrhs, A, numrows, pivot, &
&          B, numrows, info)
DO col = 1, numcols
  X(col) = B(col)
END DO
```



LAPACK: A Library and an API

LAPACK is a library that you can download for free from the Web:

`www.netlib.org`

But, it's also an Application Programming Interface (API): a definition of a set of routines, their arguments, and their behaviors.

So, anyone can write an implementation of LAPACK.





It's Good to Be Popular

LAPACK is a good choice for non-parallelized solving, because its popularity has convinced many supercomputer vendors to write their own, highly tuned versions.

The API for the LAPACK routines is the same as the portable version from NetLib, but the performance can be much better, via either ATLAS or proprietary vendor-tuned versions.

Also, some vendors have shared memory parallel versions of LAPACK.





LAPACK Performance

Because LAPACK uses BLAS, it's about as fast as BLAS.

For example, DGESV (Double precision General SolVer) on a 2 GHz Pentium4 using ATLAS gets 65% of peak, compared to 69% of peak for Matrix-Matrix multiply.

In fact, an older version of LAPACK, called LINPACK, is used to determine the top 500 supercomputers in the world.





ScaLAPACK

ScaLAPACK is the distributed parallel (MPI) version of LAPACK. It actually contains only a subset of the LAPACK routines, and has a somewhat awkward Application Programming Interface (API).

Like LAPACK, ScaLAPACK is also available from
`www.netlib.org`.





PETSc

PETSc (Portable, Extensible Toolkit for Scientific Computation) is a solver library for sparse matrices that uses distributed parallelism (MPI).

PETSc is designed for general sparse matrices with no special properties, but it also works well for sparse matrices with simple properties like banding and symmetry.

It has a simpler, more intuitive Application Programming Interface than ScaLAPACK.





Pick Your Solver Package

- Dense Matrix
 - Serial: LAPACK
 - Shared Memory Parallel: threaded LAPACK
 - Distributed Parallel: ScaLAPACK
- Sparse Matrix: PETSc





I/O Libraries



I/O Challenges

I/O presents two important challenges to scientific computing:

- Performance
- Portability

The performance issue arises because I/O is much more time-consuming than computation, as we saw in the “Storage Hierarchy” session.

The portability issue arises because different kinds of computers can have different ways of representing real (floating point) numbers.





Storage Formats

When you use a **PRINT** statement in Fortran or a **printf** in C or output to **cout** in C++, you are asking the program to output data in human-readable form:

```
x = 5
```

```
PRINT *, x
```

But what if the value that you want to output is a real number with lots of significant digits?

```
1.3456789E+23
```



Data Output as Text

1.3456789E+23

When you output data as text, each character takes 1 byte.
So if you output a number with lots of digits, then you're outputting lots of bytes.

For example, the above number takes 13 bytes to output as text.

Jargon: Text is sometimes called **ASCII** (American Standard Code for Information Interchange).





Output Data in Binary

Inside the computer, a single precision real number (Fortran **REAL**, C/C++ **float**) typically requires 4 bytes, and a double precision number (**DOUBLE PRECISION** or **double**) typically requires 8.

That's less than 13.

Since I/O is very expensive, it's better to output 4 or 8 bytes than 13 or more.

Happily, Fortran, C and C++ allow you to output data as **binary** (internal representation) rather than as text.



Binary Output Problems

When you output data as **binary** rather than as text, you output substantially **fewer bytes**, so you save time (since I/O is very expensive) and you save disk space.

But, you pay two prices:

- **Readability**: Humans can't read binary.
- **Portability**: Different kinds of computers have different ways of internally representing numbers.





Binary Readability: No Problem

Readability of binary data **isn't a problem** in scientific computing, because:

- You can always write a little program to read in the binary data and display its text equivalent.
- If you have lots and lots of data (that is, MBs or GBs), you wouldn't want to look at all of it anyway.





Binary Portability: Big Problem

Binary data portability is a **very big problem** in scientific computing, because data that's output on one kind of computer may not be readable on another, and so:

- You can't output the data on one kind of computer and then use them (for example, visualize, analyze) on another kind.
- Some day the kind of computer that output the data will be obsolete, so there may be no computer in the world that can input it, and thus the data are lost.





Portable Binary Data

The HPC community noticed this problem some years ago, and so a number of portable binary data formats were developed.

The two most popular are:

- **HDF** (Hierarchical Data Format) from the National Center for Supercomputing Applications:

<http://www.hdfgroup.org/>

- **NetCDF** (Network Common Data Form) from Unidata:

<http://www.unidata.ucar.edu/software/netcdf>





Advantages of Portable I/O

Portable binary I/O packages:

- give you portable binary I/O;
- have simple, clear APIs;
- are available for free;
- run on most platforms;
- allow you to **annotate** your data (for example, put into the file the variable names, units, experiment name, grid description, etc).

Also, both HDF and netCDF support distributed parallel I/O.





Scientific Visualization



Too Many Numbers

A typical scientific code outputs lots and lots of data.

For example, the ARPS weather forecasting code, running a 5 day forecast over the continental U.S. with a resolution of 1 km horizontal and 0.25 km vertical outputting data for every hour would produce about **10 terabytes** (10^{13} bytes).

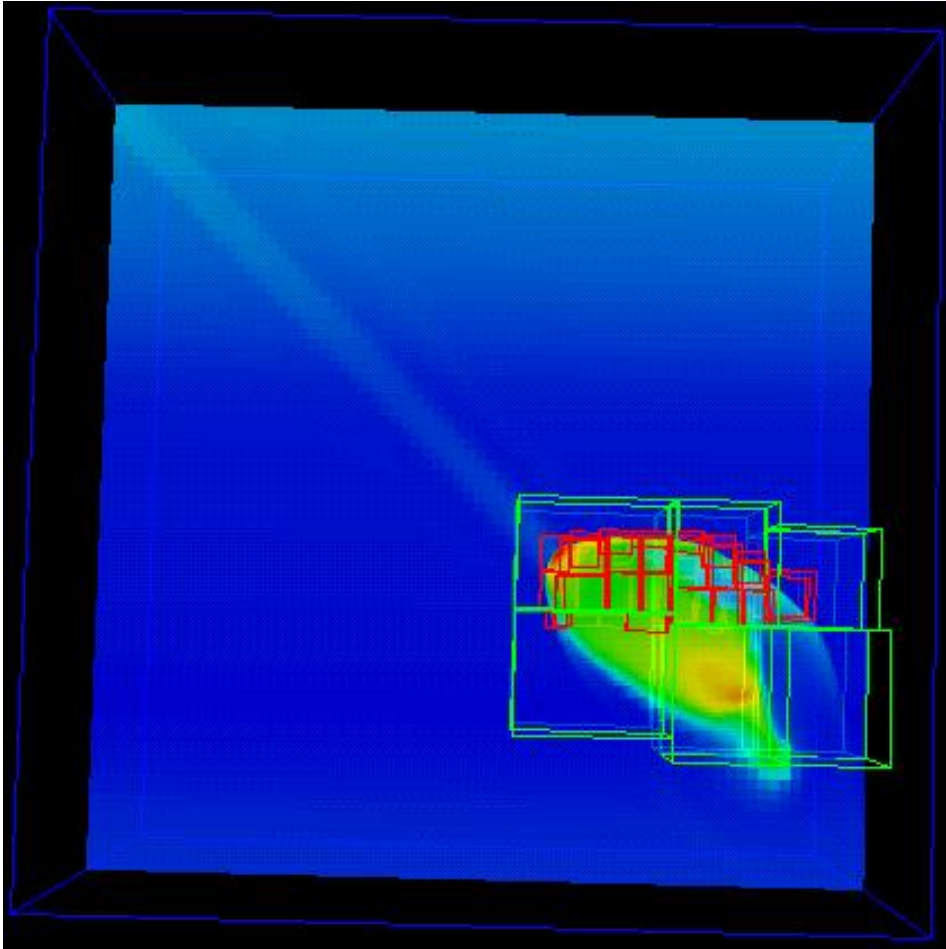
No one can look at that many numbers.



A Picture is Worth ...

... millions of numbers.

This is Comet Shoemaker-Levy 9, which hit Jupiter in 1994; the image is from 35 seconds after hitting Jupiter's inner atmosphere.^[9]





Types of Visualization

- Contour lines
 - Slice planes
 - Isosurfaces
 - Streamlines
 - Volume rendering
- ... and many others.

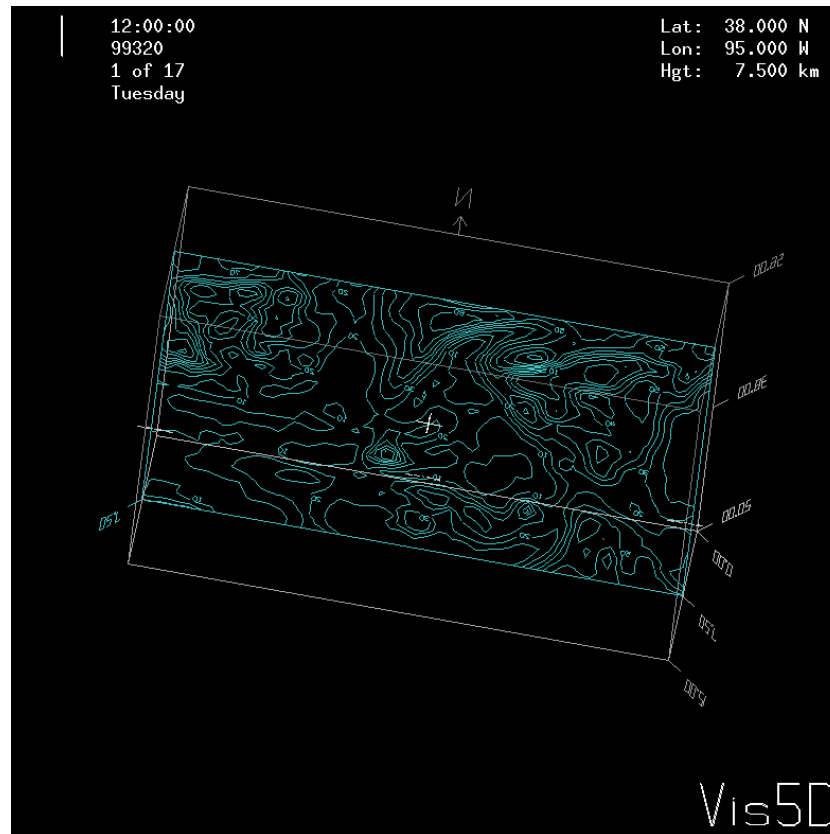
Note: except for the volume rendering, the following images were created by Vis5D,^[10] which you can download for free.





Contour Lines

This image shows *contour lines* of relative humidity. Each contour line represents a single humidity value.



Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009

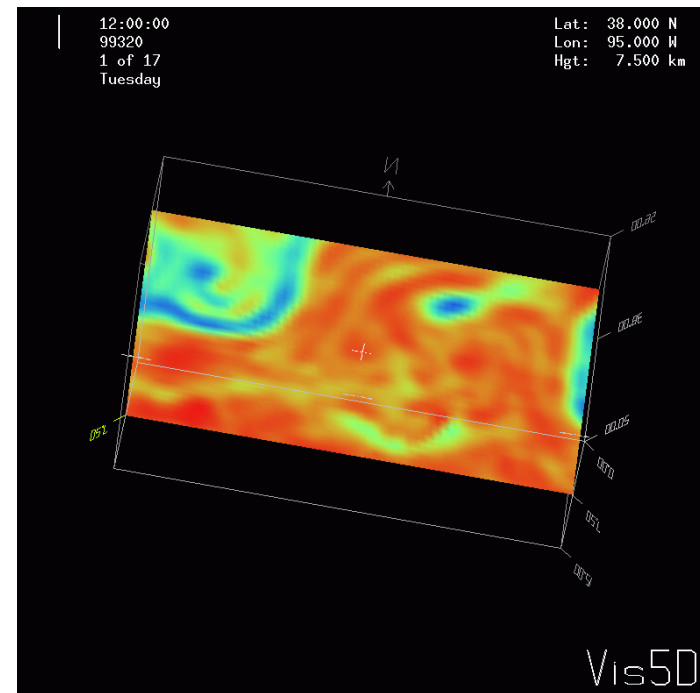




Slice Planes

A *slice plane* is a single plane passed through a 3D volume.

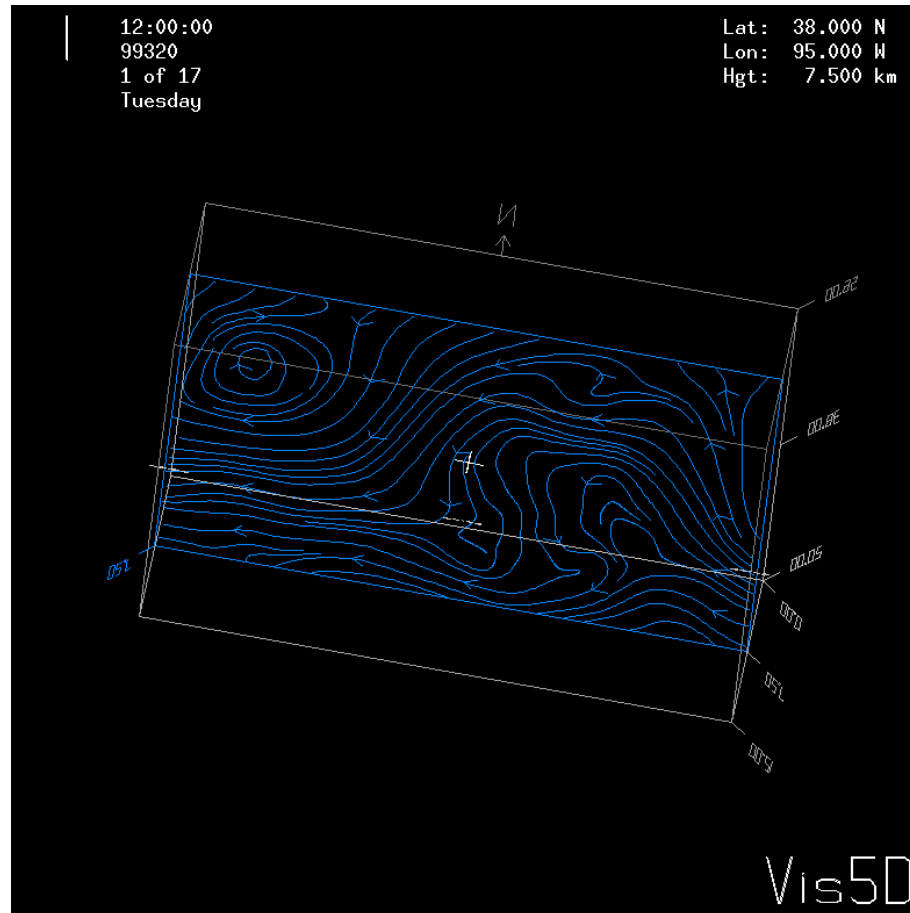
Typically, it is color coded by mapping some scalar variable to color (for example, low vorticity to blue, high vorticity to red).





Streamlines

A streamline traces a vector quantity (for example, velocity).

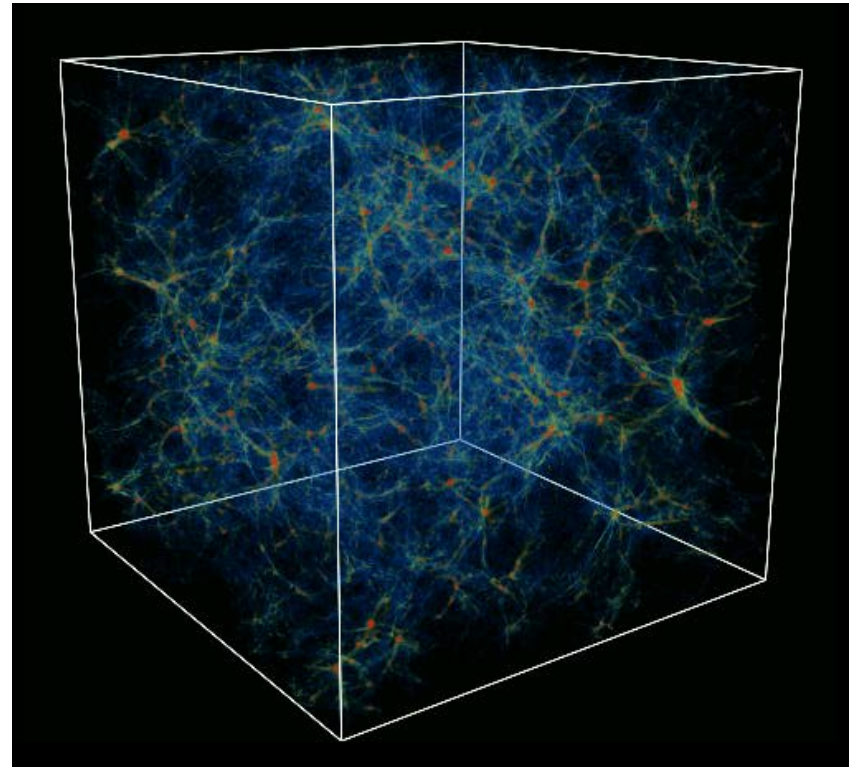


Volume Rendering

A *volume rendering* is created by mapping some variable (for example, energy) to color and another variable (for example, density) to opacity.

This image shows the overall structure of the universe.^[11]

Notice that the image looks like thick colored smoke.





OK Supercomputing Symposium 2009



2003 Keynote:
Peter Freeman
NSF
Computer &
Information
Science &
Engineering
Assistant Director



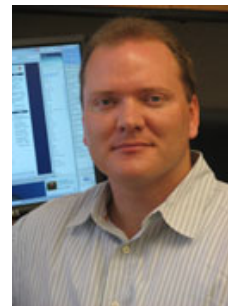
2004 Keynote:
Sangtae Kim
NSF Shared
Cyberinfrastructure
Division Director



2005 Keynote:
Walt Brooks
NASA Advanced
Supercomputing
Division Director



2006 Keynote:
Dan Atkins
Head of NSF's
Office of
Cyber-
infrastructure



2007 Keynote:
Jay Boisseau
Director
Texas Advanced
Computing Center
U. Texas Austin



2008 Keynote:
José Muñoz
Deputy Office
Director/ Senior
Scientific Advisor
Office of Cyber-
infrastructure
National Science
Foundation



2009 Keynote:
Ed Seidel
Director
NSF Office of
Cyber-
infrastructure

FREE! Wed Oct 7 2009 @ OU
<http://symposium2009.oscer.ou.edu/>

Parallel Programming Workshop
FREE! Tue Oct 6 2009 @ OU

Sponsored by SC09 Education Program
FREE! Symposium Wed Oct 7 2009 @ OU

Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009

**Registration
is OPEN!**





SC09 Summer Workshops

This coming summer, the SC09 Education Program, part of the SC09 (Supercomputing 2009) conference, is planning to hold two weeklong supercomputing-related workshops in Oklahoma, for **FREE** (except you pay your own transport):

- **At OSU Sun May 17 – the May 23:**
FREE Computational Chemistry for Chemistry Educators (2010 TENTATIVE: Computational Biology)
- **At OU Sun Aug 9 – Sat Aug 15:**
FREE Parallel Programming & Cluster Computing

We'll alert everyone when the details have been ironed out and the registration webpage opens.

Please note that you must **apply** for a seat, and acceptance **CANNOT** be guaranteed.



Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009



SC09 Summer Workshops

1. May 17-23: Oklahoma State U: Computational Chemistry
2. May 25-30: Calvin Coll (MI): Intro to Computational Thinking
3. June 7-13: U Cal Merced: Computational Biology
4. June 7-13: Kean U (NJ): Parallel, Distributed & Grid
5. June 14-20: Widener U (PA): Computational Physics
6. July 5-11: Atlanta U Ctr: Intro to Computational Thinking
7. July 5-11: Louisiana State U: Parallel, Distributed & Grid
8. July 12-18: U Florida: Computational Thinking Pre-college
9. July 12-18: Ohio Supercomp Ctr: Computational Engineering
10. Aug 2- 8: U Arkansas: Intro to Computational Thinking
11. Aug 9-15: U Oklahoma: Parallel, Distributed & Grid



Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009



To Learn More Supercomputing

<http://www.oscer.ou.edu/education.php>



Supercomputing in Plain English: Grab Bag
Tuesday May 5 2009



Thanks for helping!

- OSCER operations staff (Brandon George, Dave Akin, Brett Zimmerman, Josh Alexander)
- OU Research Campus staff (Patrick Calhoun, Josh Maxey, Gabe Wingfield)
- Kevin Blake, OU IT (videographer)
- Katherine Kantardjieff, CSU Fullerton
- John Chapman and Amy Apon, U Arkansas
- Andy Fleming, KanREN/Kan-ed
- This material is based upon work supported by the National Science Foundation under Grant No. OCI-0636427, “CI-TEAM Demonstration: Cyberinfrastructure Education for Bioinformatics and Beyond.”



**Thanks for your
attention this whole
semester!**



Questions?



References

- [1] C. L. Lawson, R. J. Hanson, D. Kincaid, and F. T. Krogh, *Basic Linear Algebra Subprograms for FORTRAN Usage*, [ACM Trans. Math. Soft., 5 \(1979\)](#), pp. 308--323.
- [2] <http://www.netlib.org/blas/>
- [3] <http://math-atlas.sourceforge.net/>
- [4] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, D. Sorensen, *LAPACK Users' Guide*, 3rd ed, 1999.
<http://www.netlib.org/lapack/>
- [5] L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, R. C. Whaley, *ScaLAPACK Users' Guide*, 1997.
<http://www.netlib.org/scalapack/>
- [6] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, L. Curfman McInnes and B. F. Smith, PETSc home page, 2001. <http://www.mcs.anl.gov/petsc>
- [7] S. Balay, W. D. Gropp, L. Curfman McInnes and B. Smith, *PETSc Users Manual*, ANL-95/11 - Revision 2.1.0, Argonne National Laboratory, 2001.
- [8] S. Balay, W. D. Gropp, L. Curfman McInnes and B. F. Smith, "Efficient Management of Parallelism in Object Oriented Numerical Software Libraries", in *Modern Software Tools in Scientific Computing*, E. Arge, A. M. Bruaset and H. P. Langtangen, editors, Birkhauser Press, 1997, 163-202.
- [9] <http://hneeman.oscer.ou.edu/hamr.html>
- [10] <http://www.ssec.wisc.edu/~billh/vis5d.html>
- [11] Image by Greg Bryan, MIT.