Can Scalable Development Lead to Scalable Execution?

Damian Rouson
Scalable Computing R&D
Sandia National Laboratories

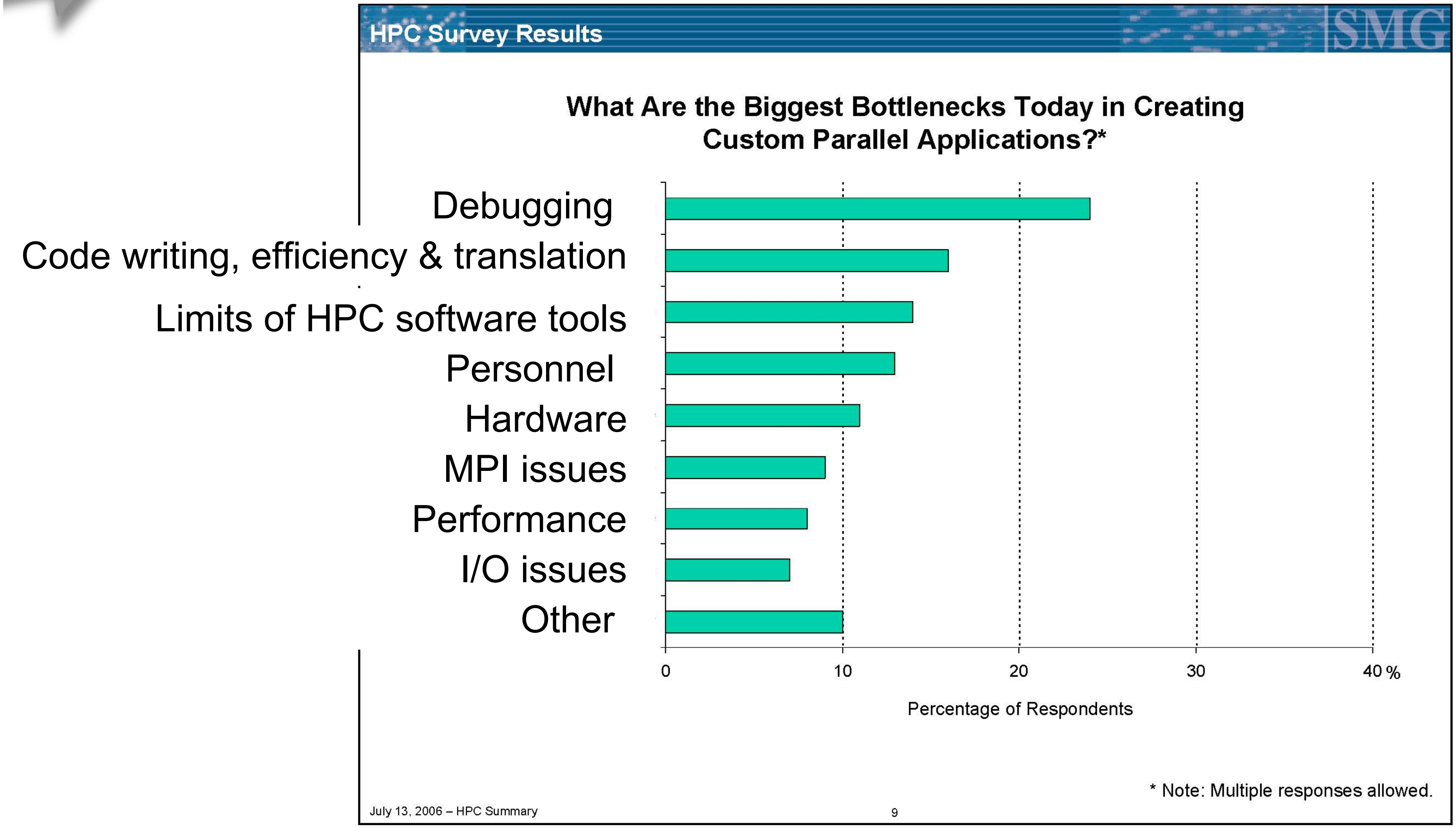
Sponsors:
Office of Naval Research, City University of New York

Sandia National Laboratories

Outline

- Motivation, Objectives & Guideposts
- Conventional development
- Scalable development
- Applications
- Toward scalable execution
- Conclusions & Acknowledgments

Motivation


HPC Survey Results
SMG
What Are the Biggest Bottlenecks Today in Creating Custom Parallel Applications?*
Debugging
Code writing, efficiency & translation
Limits of HPC software tools
Personnel
Hardware
MPI issues
Performance
I/O issues
Other
0
10
20
30
40 %
Percentage of Respondents
* Note: Multiple responses allowed.
July 13, 2006 – HPC Summary
9

Objectives

1. To develop a design methodology that scales up to large numbers of programming units, e.g. procedures, classes, data structures, etc.
2. To demonstrate that this methodology can produce new science.
3. To demonstrate that this approach also scales up to large numbers of execution units, e.g. threads, processes, cores, etc.

Guideposts

“What are the metrics?”
Oyekunle Olukotun, Stanford EE/CS, c. 1996

“Procedural programming is like an N-body problem.”
Lester Dye, Stanford Petroleum Eng., c. 1995

“Separate the data from the physcis.”
Jaideep Ray, Sandia, c. 2004

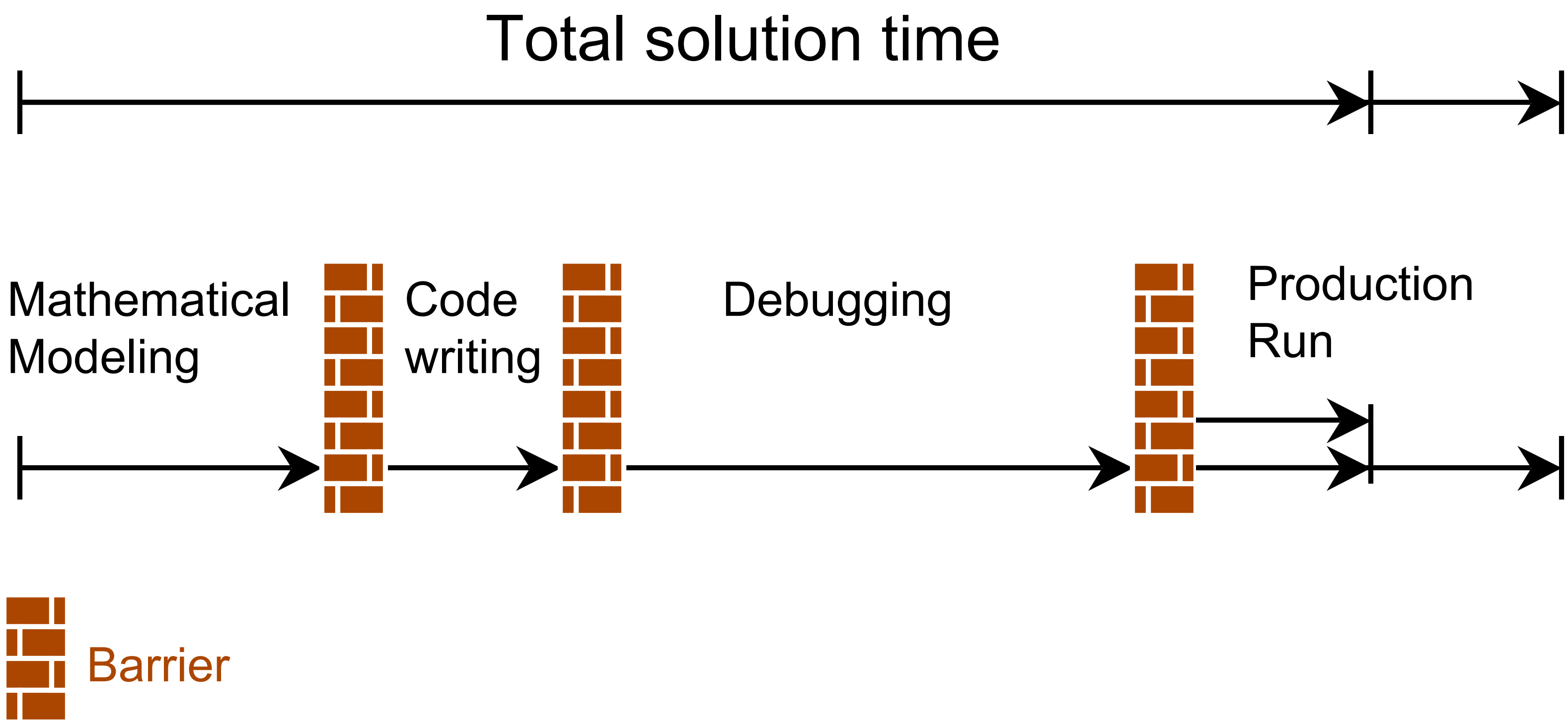
“First they ignore you. Then they laugh at you. Then they fight you. Then you win.”
Mahatma Ghandi, c. ????

Sandia National Laboratories

Outline

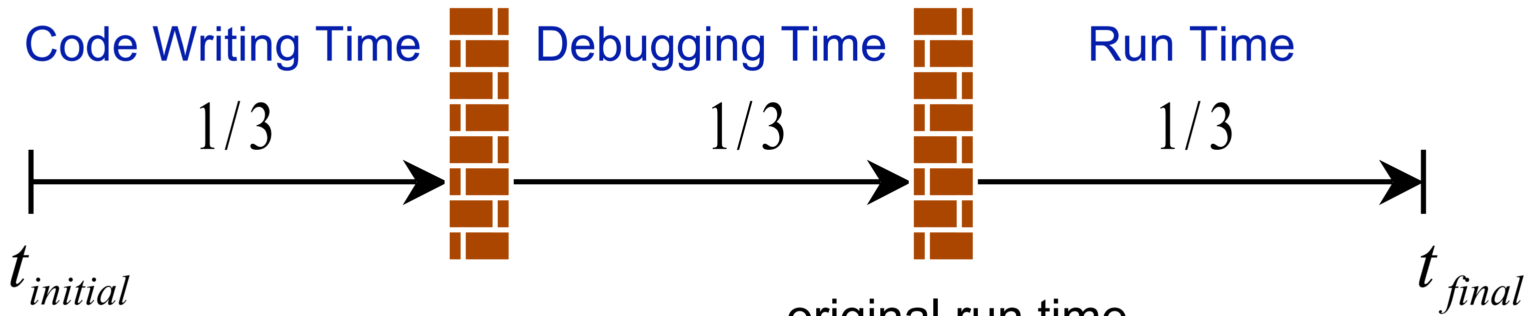
- Motivation, Objectives & Guideposts
- Conventional development
  - Amdahl's Law
  - Pareto Principle
  - Complexity
- Scalable development
- Applications
- Toward scalable execution
- Conclusions & Acknowledgments

Conventional Development



Amdahl's Law

Representative case study for a published run*:



Run-time speedup: $S_{run} \equiv \dfrac{\text{original run time}}{\text{optimized run time}}$

Total speedup: $S_{tot} = \dfrac{1}{\dfrac{2}{3} + \dfrac{1}{3}\dfrac{1}{S_{run}}} \Rightarrow \lim\limits_{S_{run}\to\infty} S_{tot} = 1.5$

The speedup achievable by focusing solely on decreasing run time is very limited.

*Rouson et al. (2007) *Proc. 2006 Summer Program*, Center for Turbulence Research, Stanford University.

Sandia National Laboratories

Pareto Principle

When participants (lines) share resources (run time), there always exists a number $k \in [50,100)$ such that (1-k)% of the participants occupy k% of the resources:

Limiting cases:

- k=50%, equal distribution
- k→100%, monopoly

Rule of thumb: 20% of the lines occupy 80% of the run time

Scalability requirements determine the percentage of the code that can be focused strictly on programmability:

$$S_{\max} = \lim_{S_{k\%} \to \infty} \frac{1}{0.2 + 0.8 / S_{k\%}} = 5$$

Sandia National Laboratories

Runtime Profile

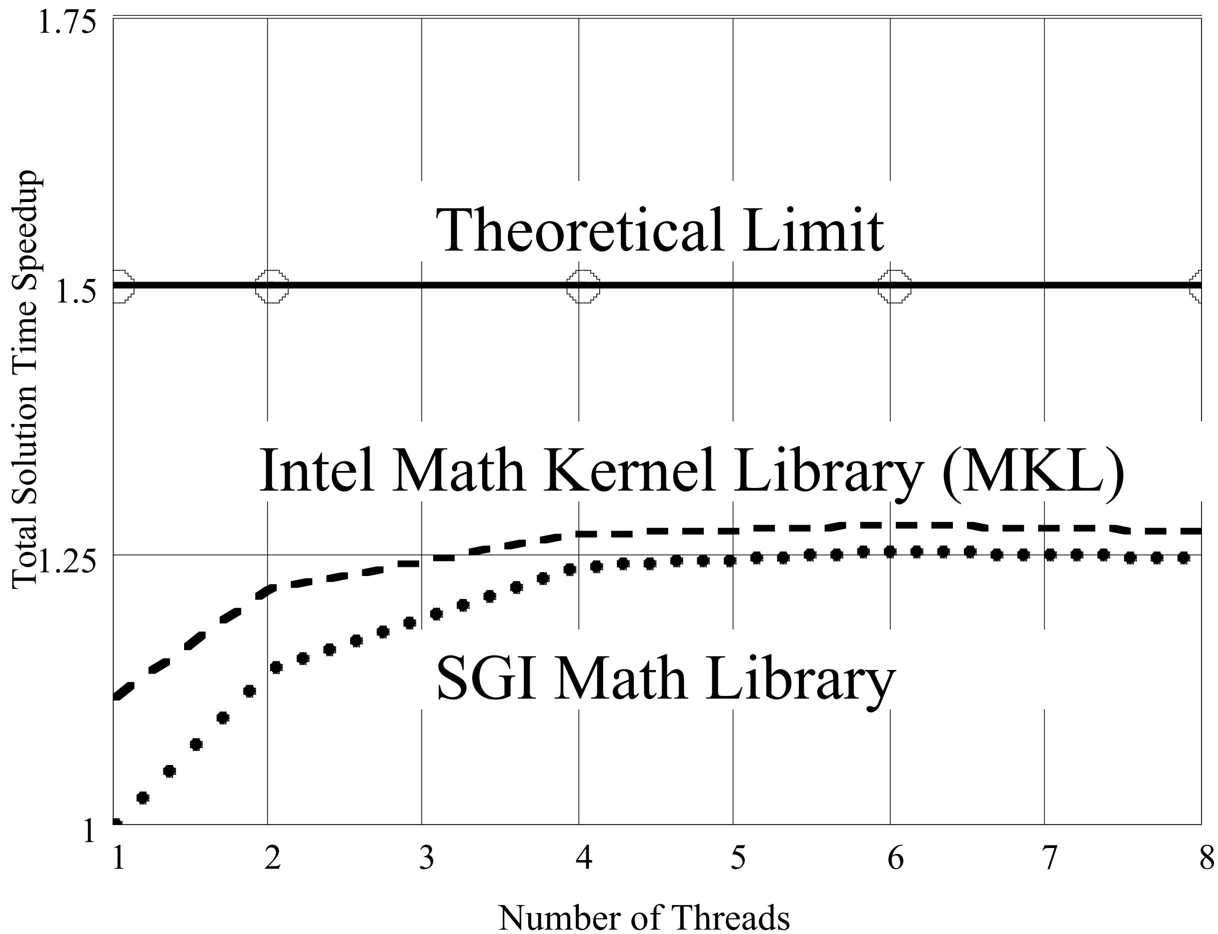
Procedure	Inclusive Run-Time Share (%)
main	100.0
operator(.x.)	79.5
RK3_Integrate	47.8
Nonlinear_Fluid	44.0
Statistics_	43.8
transform_to_fourier	38.7
transform_to_physical	23.6

Calls

- 5% procedures occupy nearly 80% of run time.
- Structure 95% of procedures to reduce <u>development time</u>.

Sandia National Laboratories

Total Solution Time Speedup



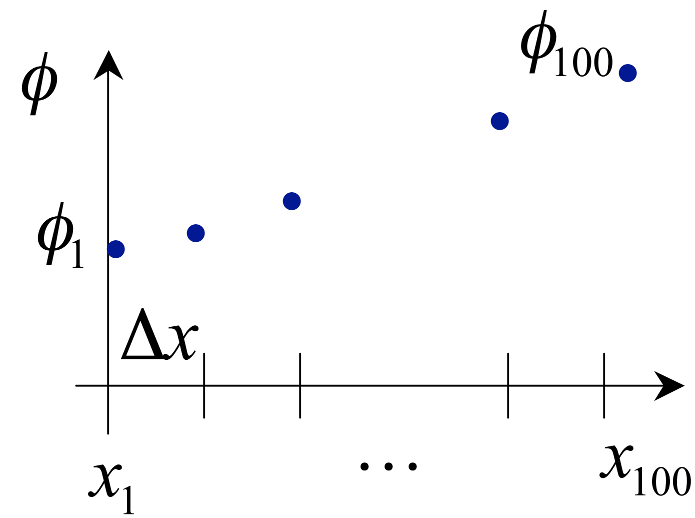
Conventional Development

Model Problem: Unsteady 1D Diffusion

$$\partial\phi / \partial t = D\nabla^2\phi$$

Semi-discrete equations:

$$\frac{d}{dt}\phi_i = D\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2}$$



Fully discrete equations:

$$\phi_i^{n+1} = \phi_i^n + \Delta t \cdot D\frac{\phi_{i+1}^n - 2\phi_i^n + \phi_{i-1}^n}{\Delta x^2}$$

Solution algorithm:

$$\vec{\phi} \leftarrow \vec{\phi} + \frac{\Delta t \cdot D}{\Delta x^2}[A]\vec{\phi}$$

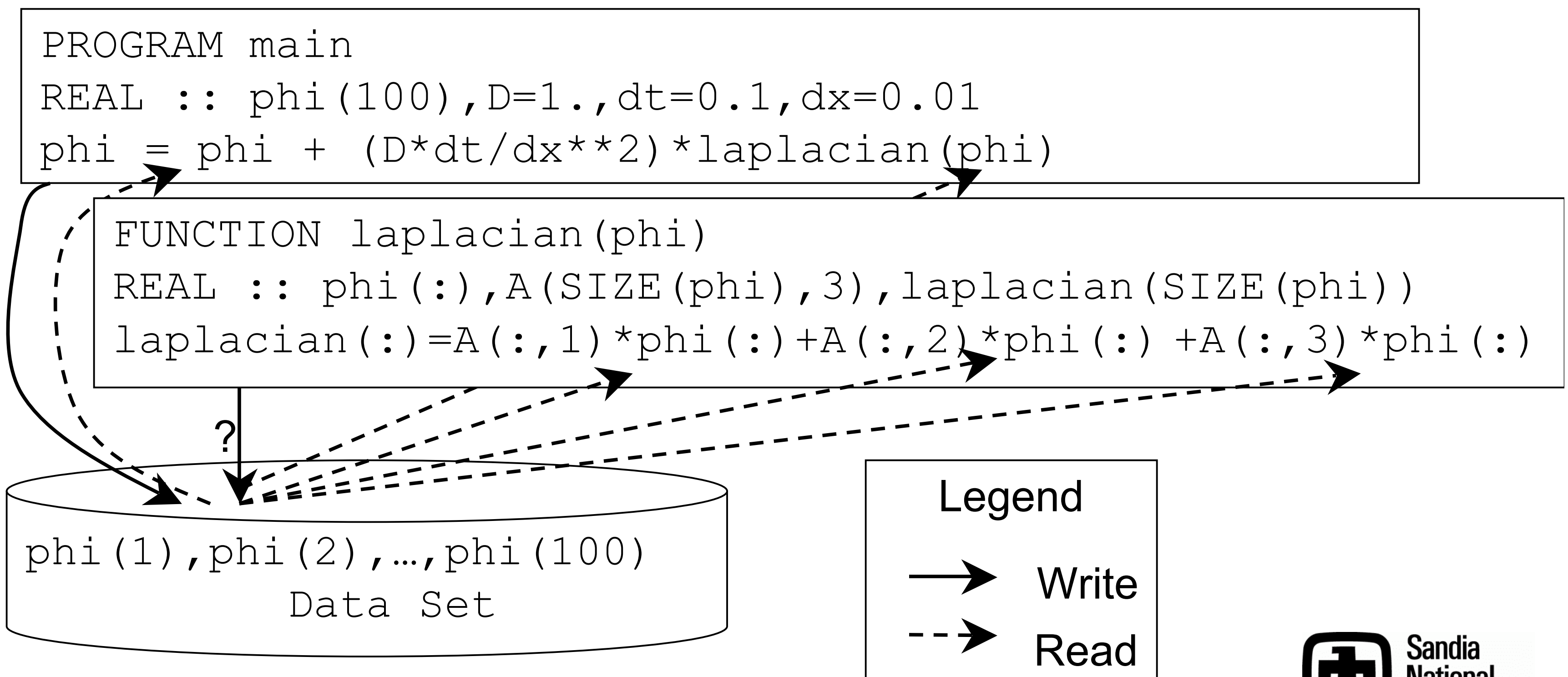
Sandia National Laboratories

Conventional Program Debugging

"Not much time is spent fixing bugs. Most of the time is spent *finding* bugs."

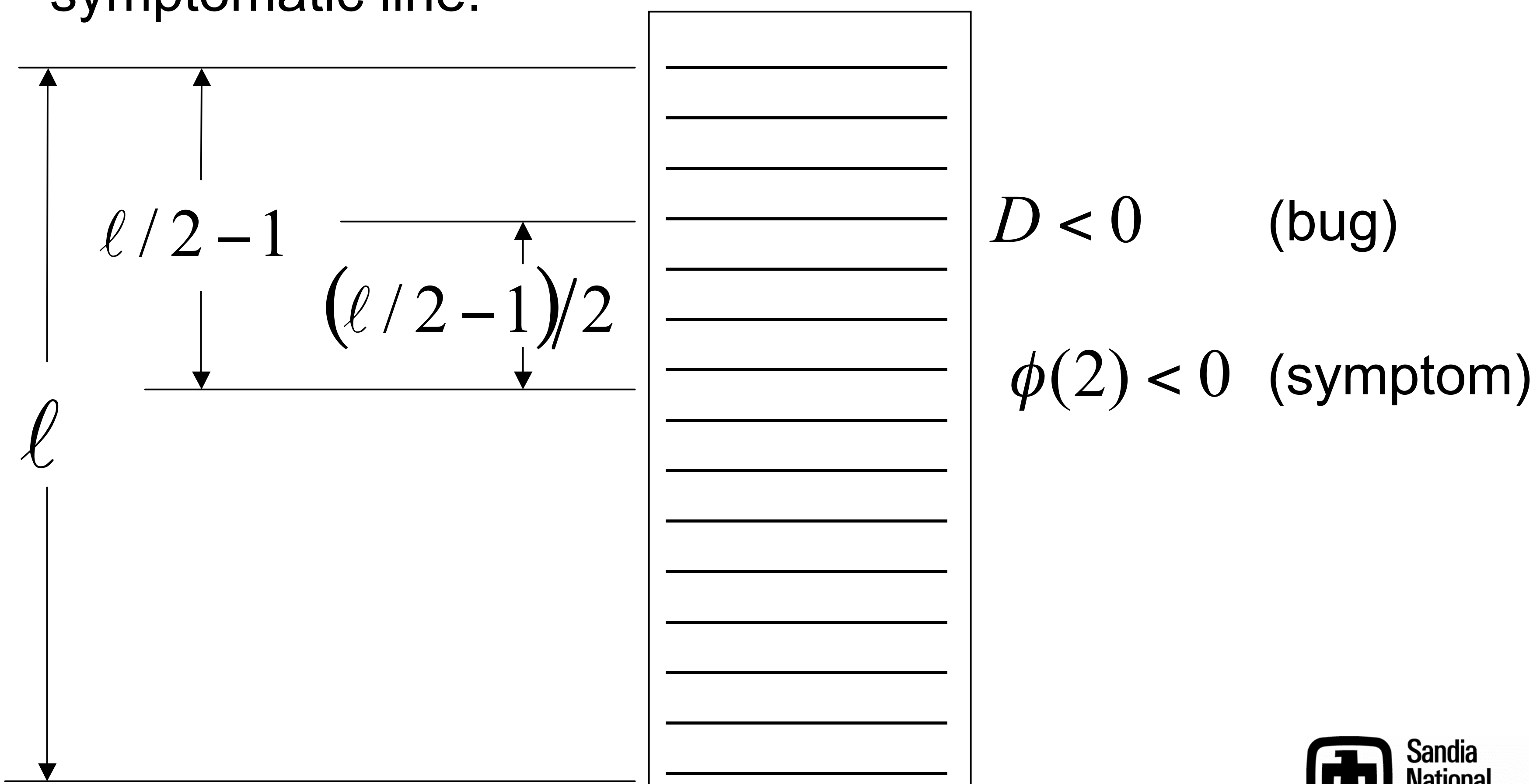
-- Shalloway & Truitt, *Design Patterns Explained*

-- Oliveira & Stewart, *Writing Scientific Software*



Bug Search Complexity

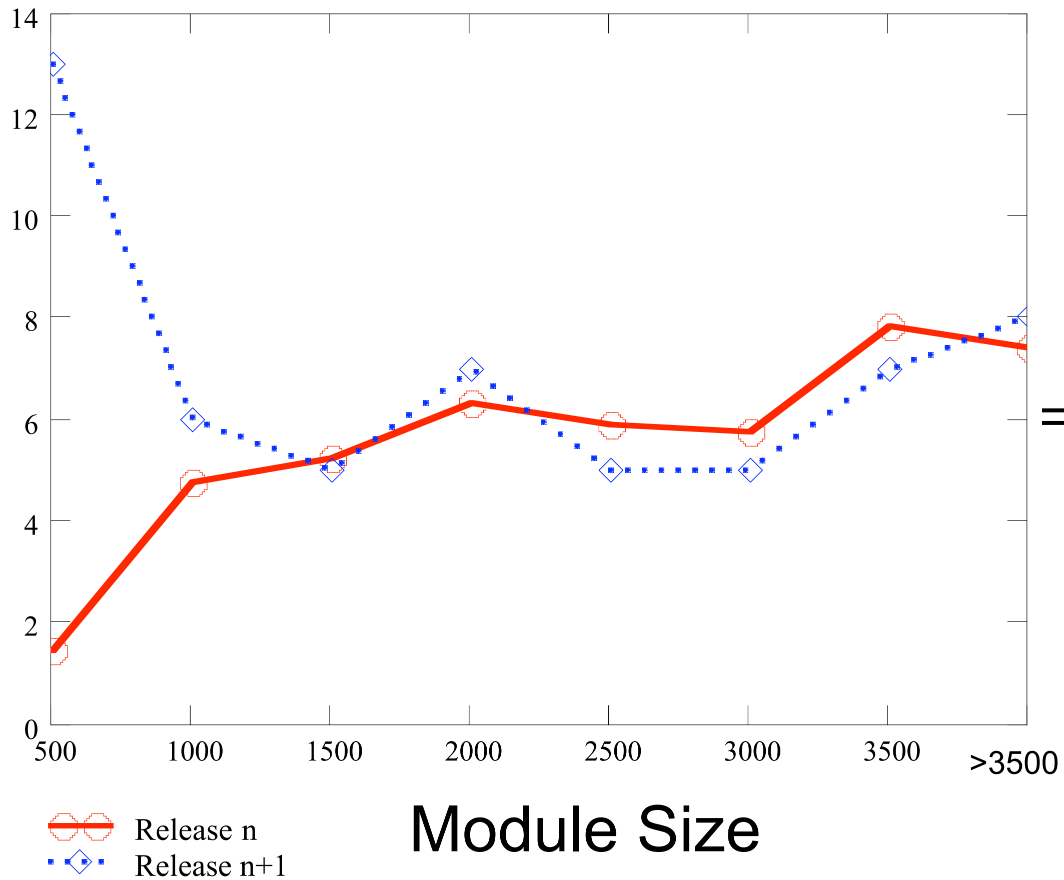
Consider a list of the unique program lines with all lines that execute before the symptom preceding the symptomatic line:



Code Fault Rates

Fenton & Ohlssen, "Quantitative analysis of faults and failures in a complex software system," *IEEE Trans. Soft. Eng.* 2000:

Fault Rate



Module Size

$\Rightarrow r \approx 6/1000$

Sandia National Laboratories

Scientific Code Fault Rates

Hatton, L. "The `T' Experiments – Errors in Scientific Software," *Comp. Sci. Eng.* 1997:

- 8 statically detectable faults/1000 lines of *commercially released* C code
- 12 statically detectable faults/1000 lines of *commercially released* Fortran 77 code
- more recent data finds 2-3 times as many faults in C++

$$\Rightarrow r \approx 0.006 - 0.036$$

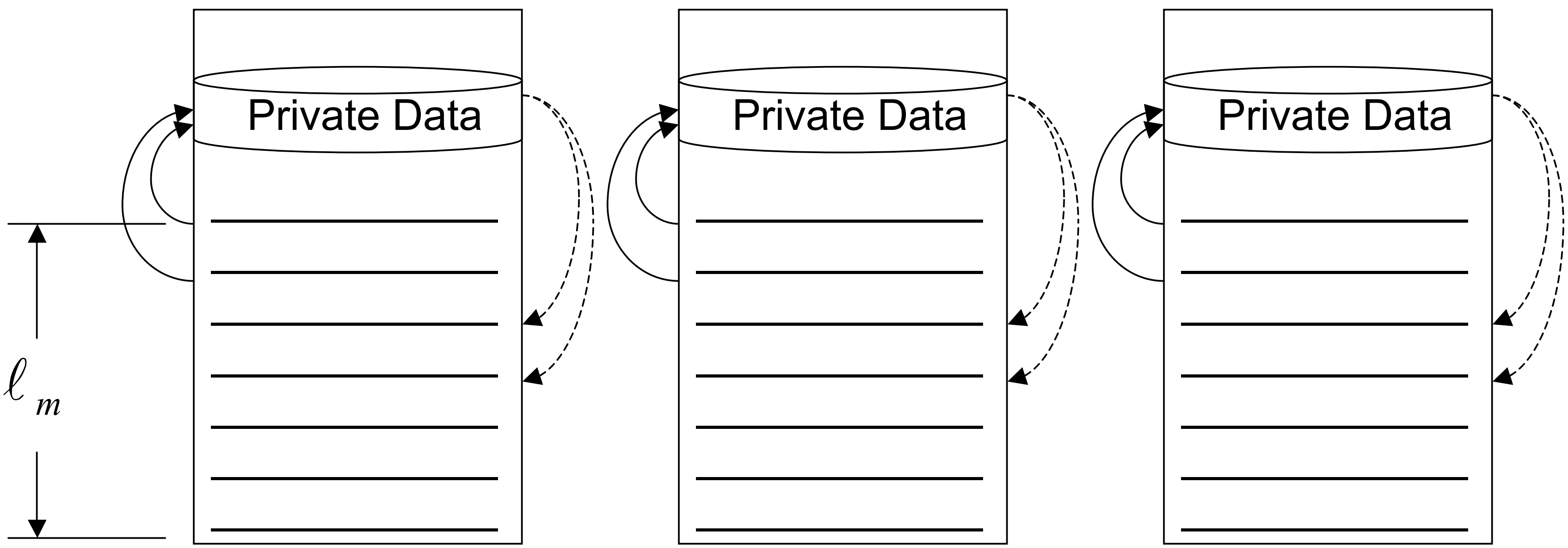
$$t_{search} = (\# bugs) \times (lines\ searched\ per\ bug)(\bar{t}_{line\ review})$$

$$= (r\ell)\left[(\ell/2-1)/2\right](\bar{t}_{line\ review})$$

Outline

- Motivation, Objectives & Guideposts
- Conventional development
- Scalable development
  - Complexity
  - Information theory
- Applications
- Toward scalable execution
- Conclusions & Acknowledgments

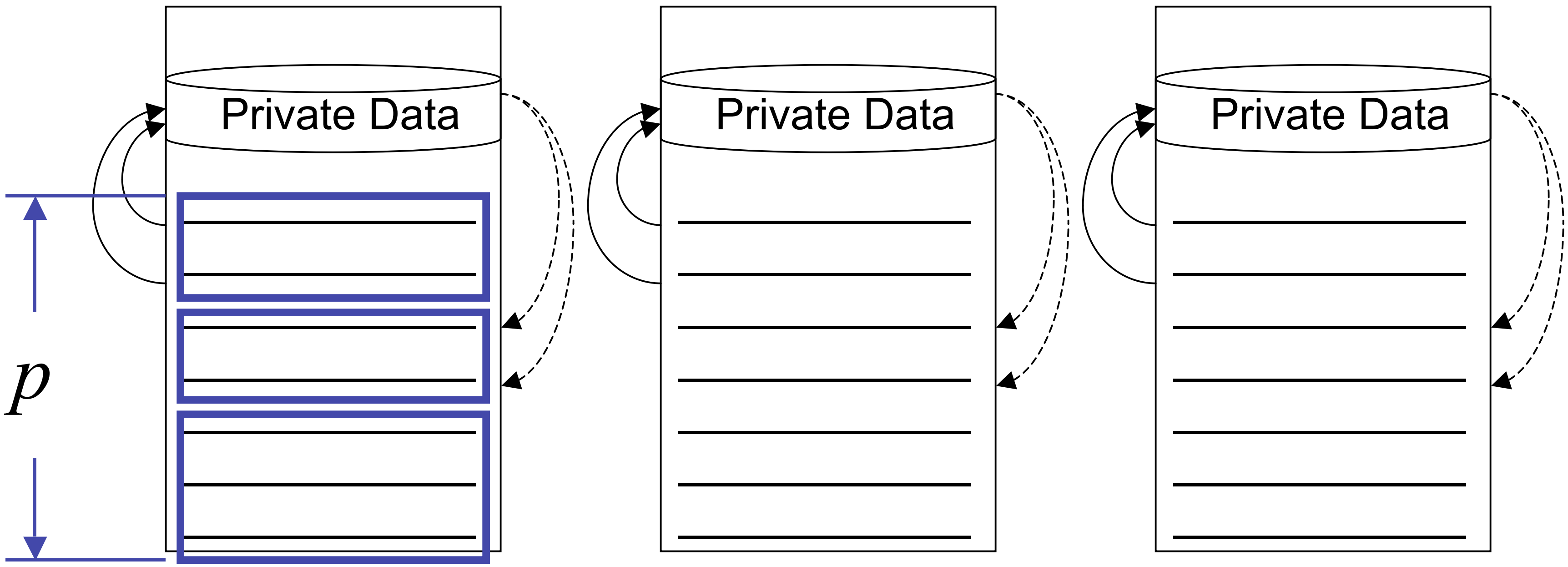
Object-Oriented Programming



$$t_{search} = (r\ell_m)\left[(\ell_m/2-1)/2\right](\bar{t}_{line\ review})$$

$$\ell_m << \ell$$

Scientific OOP



$$\rho \equiv \frac{\ell_m}{p} = \frac{\textit{lines per module}}{\textit{procedures per module}}$$

$$t_{search} = (r\rho p)\left[(\rho p/2-1)/2\right]\left(\bar{t}_{line\ review}\right)$$

Scientific OOP

Legend: **public**, private

Equation	Concept	Class	Code
$\phi(t+\Delta t) = \phi(t) + \Delta t \dfrac{\partial \phi}{\partial t}$	Time Integration Algorithm	**Integrator**	`class(Scalar) :: Smoke` `Smoke = Smoke + dt*d_dt(Smoke)`

$\partial \phi / \partial t = D \nabla^2 \phi$	Governing PDE	**Scalar**	`class(Field) :: phi`
			`REAL :: D`
			`d_dt(phi)` `= D*laplacian(phi)`

$\nabla^2 \phi = \partial^2 \phi / \partial x^2$
$\partial \phi / \partial x \approx \Delta \phi / \Delta x$
	Spatially Differentiable Fields	**Field**	`REAL, DIMENSION(n) :: p`
			`laplacian(p)` `= d2_dx2(p)`
			`class(Grid) :: x`
			`d_dx(p)=delta(p)/delta(x)`

$\Delta x = x_{i+1} - x_i$	Spatial Discretization	**Grid**	`REAL :: dx` **`delta(x)`** `= dx`

Decomposing the problem into a set of classes that admit an abstract data type calculus yields

$$\rho \approx const., \quad p \approx const.$$

Information Theory

- Interface information content sets the minimum amount of communication between developers.
- Let p_i = frequency of occurrence of the i^{th} keyword in a set of statements. Shannon entropy is

$$S = -\sum_i p_i \log p_i \geq 0$$

- Repeated implementation of same procedural interfaces generates high p_i values → low S.
- Kirk & Jensen (2004) related Shannon entropy of codes to thermodynamic entropy, enabling the study of phase transitions in code structure.

Sandia National Laboratories

Kolmogorov Complexity

- For a program *p*, the Kolmogorov complexity *K(p)* is the shortest description in some description language
- Properties:
  - Provably not computable.
  - Bounded from above by any actual description of p.
  - Lowest upper bound at any given time: compressed program length + decompression program length
- Using this measure, we have detected slightly greater complexity in C++ than Fortran 2003

Applications

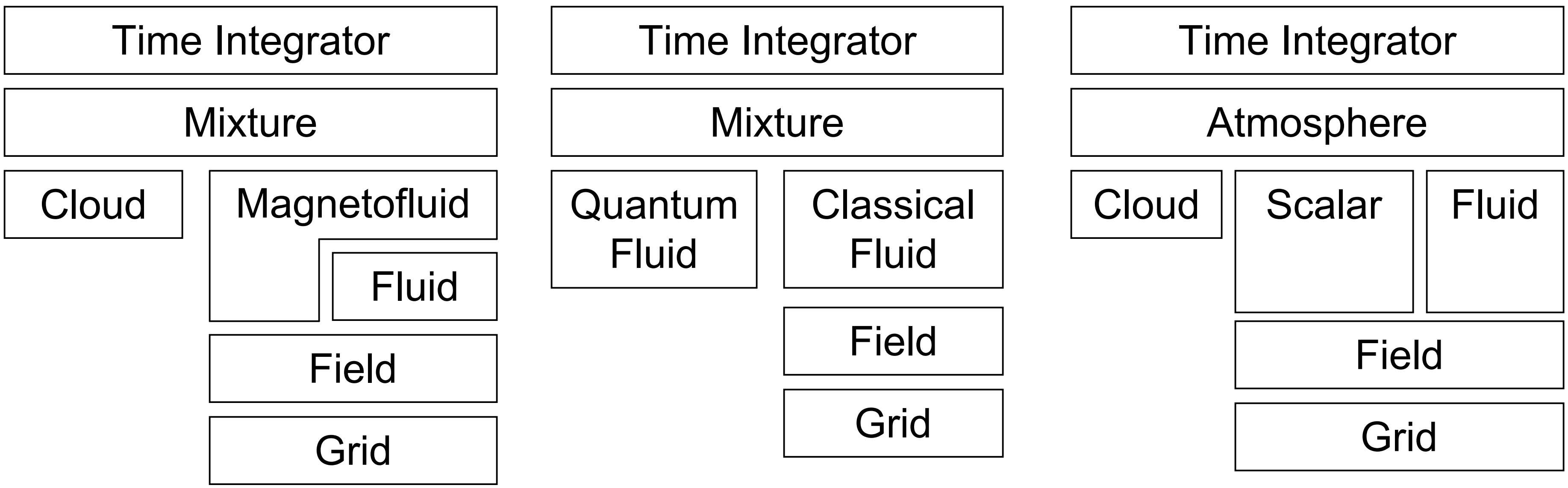
Currently Running

Under Development

Solid particle dispersion in electrically conducting fluids*:

Quantum vortex interactions with classical fluids**:

Aerosol dispersion in the atmospheric boundary layer***:



(Vertically adjacent layers communicate through interfaces.)

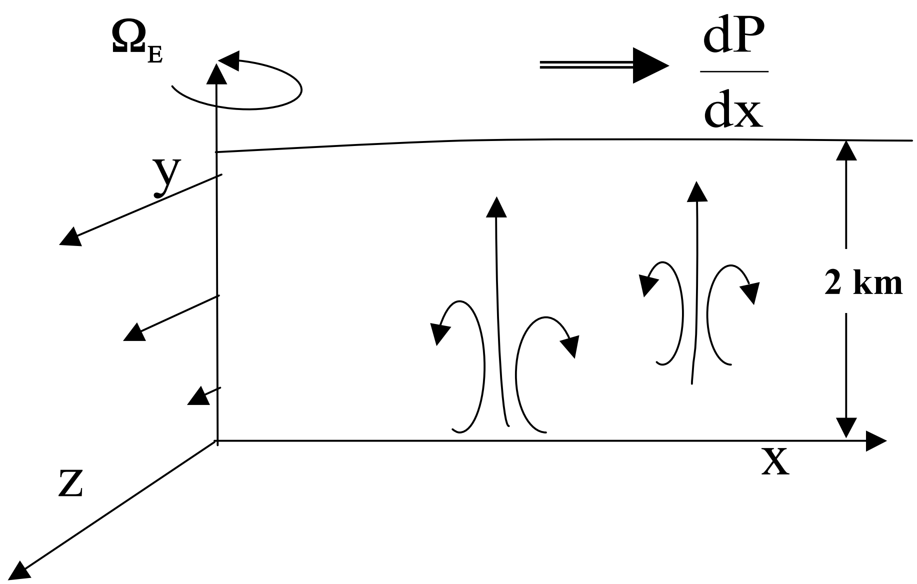
*Rouson et al. (2008) *Physics of Fluids*, February.

**Morris, Koplik & Rouson (2008) *Physical Review Letters*, in review.

***Rouson & Handler (2007) in *Environmental Sciences & Environmental Computing, Vol. III.*

Sandia National Laboratories

Large Eddy Simulation of the ABL



Physical Processes

- Shear
- Buoyancy
- Coriolis effects
- Geostrophic wind forcing
- Thermal Fluctuations
- Passive Scalar

Code Details

- Fully spectral LES: Fourier in horizontal, Chebyshev in vertical.
- Uniform grid in horizontally, cosine-stretched grid vertically.
- Compressibility is neglected (different from COAMPS).

Sandia National Laboratories

Governing Equations

Momentum:

$$\frac{\partial \vec{u}}{\partial t} = \underbrace{\vec{N}}_{\text{Advection}} + \underbrace{\vec{C}}_{\text{Coriolis}} - \underbrace{\vec{\nabla}\Pi}_{\text{Pressure}} + \underbrace{\vec{\nabla}\cdot\bar{\bar{\tau}}_{sgs}}_{\text{Subgrid Physics}} + \underbrace{\frac{\theta'}{\theta_o} g\hat{e}_3}_{\text{Buoyancy}} + \underbrace{\frac{dP}{dx}\hat{e}_1}_{\text{Geostrophic pressure gradient}}$$

Mass

$$\vec{\nabla}\cdot\vec{u} = 0$$

Heat

$$\frac{\partial \theta'}{\partial t} + \vec{u}\cdot\vec{\nabla}\theta' = \vec{\nabla}\cdot\left(\frac{\upsilon_T}{\mathrm{Pr}_T}\nabla\theta'\right)$$

$$\Pi \equiv c_p\bar{\theta}\pi' + \vec{u}\cdot\vec{u}/2$$

Exner Function: $\pi \equiv \left(p/p_0\right)^{\frac{\gamma}{\gamma-1}}$

Virtual Temperature: $\theta \equiv T/\pi$

Smagorinsky Sub-Grid Scale Turbulence Model:

$$\tau_{sgs} = 2\cdot\nu_T\cdot S \qquad \ell \sim \Delta \ \text{(grid scale)} \qquad \nu_T = C_S\ell^2\sqrt{2\cdot|S|^2}$$

Sandia National Laboratories

Simulation Parameters

- VERY SIMPLE PHYSICS
  - PRESSURE GRADIENT IN THE X-DIRECTION
  - CONSTANT TURBULENT VISCOSITY
  - ROTATING EARTH

L_x = 12.57 km L_y = 2.0 km L_z = 4.71 km

G = 2.88 m/s (geostrophic wind)

ν_T = 0.72 m^2/s (Agrees reasonably well with Sullivan et al BL Met. 1994)

$\Omega_E = 7.2722 \times 10^{-5}$ rad/s

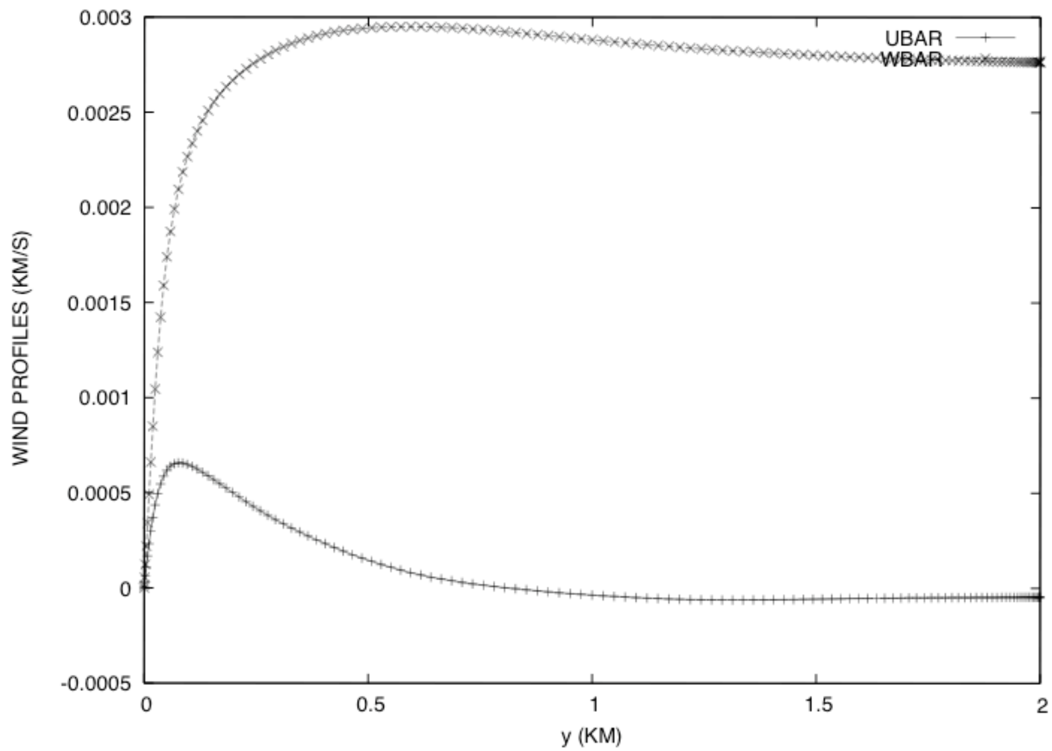
THESE PARAMETERS GIVE

$\Delta_{Ekman} = (\nu_T / \Omega_E)^{1/2} = 0.1$ km

$Re = (G * L_y)/\nu_T = 8000$

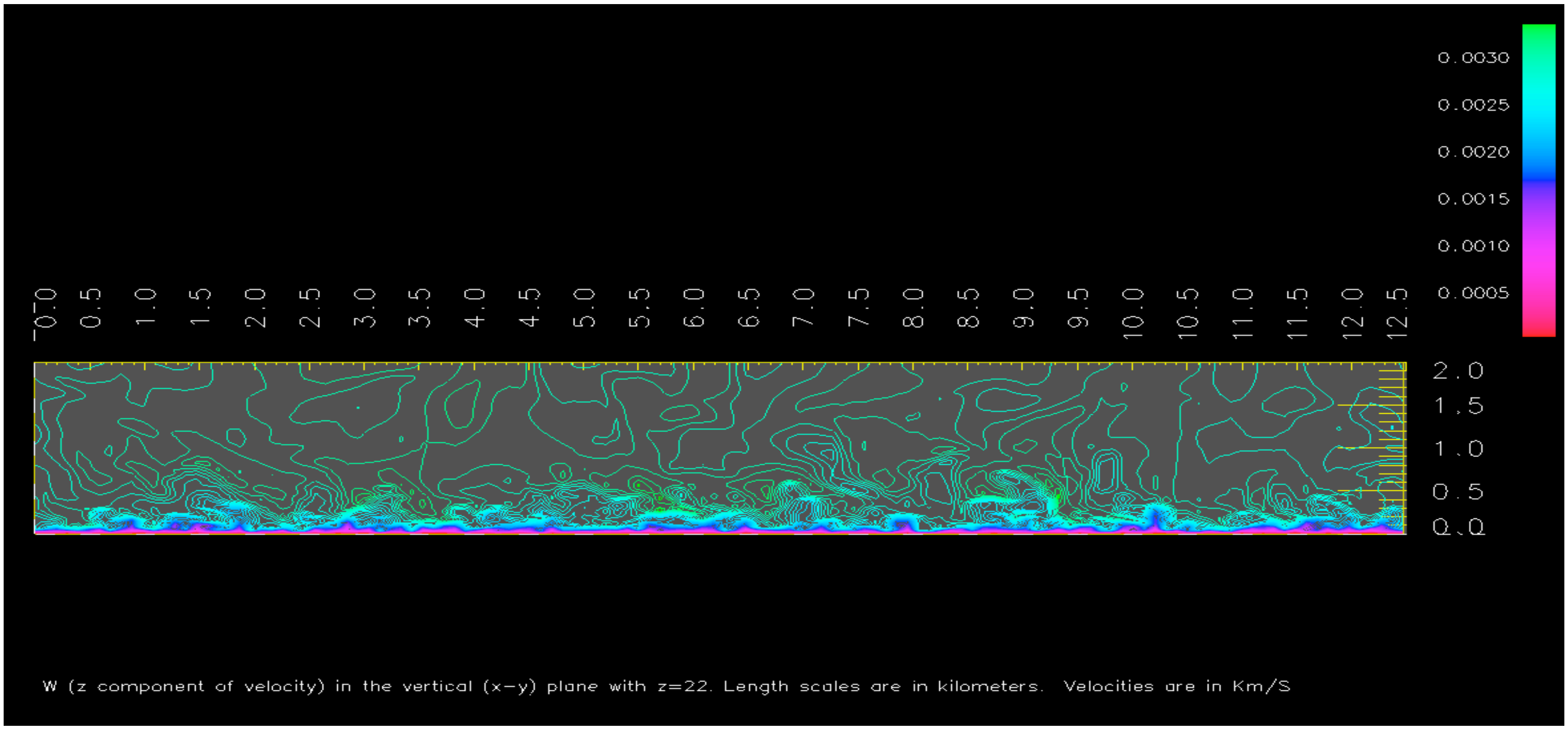
Sandia National Laboratories

Mean Wind Profiles



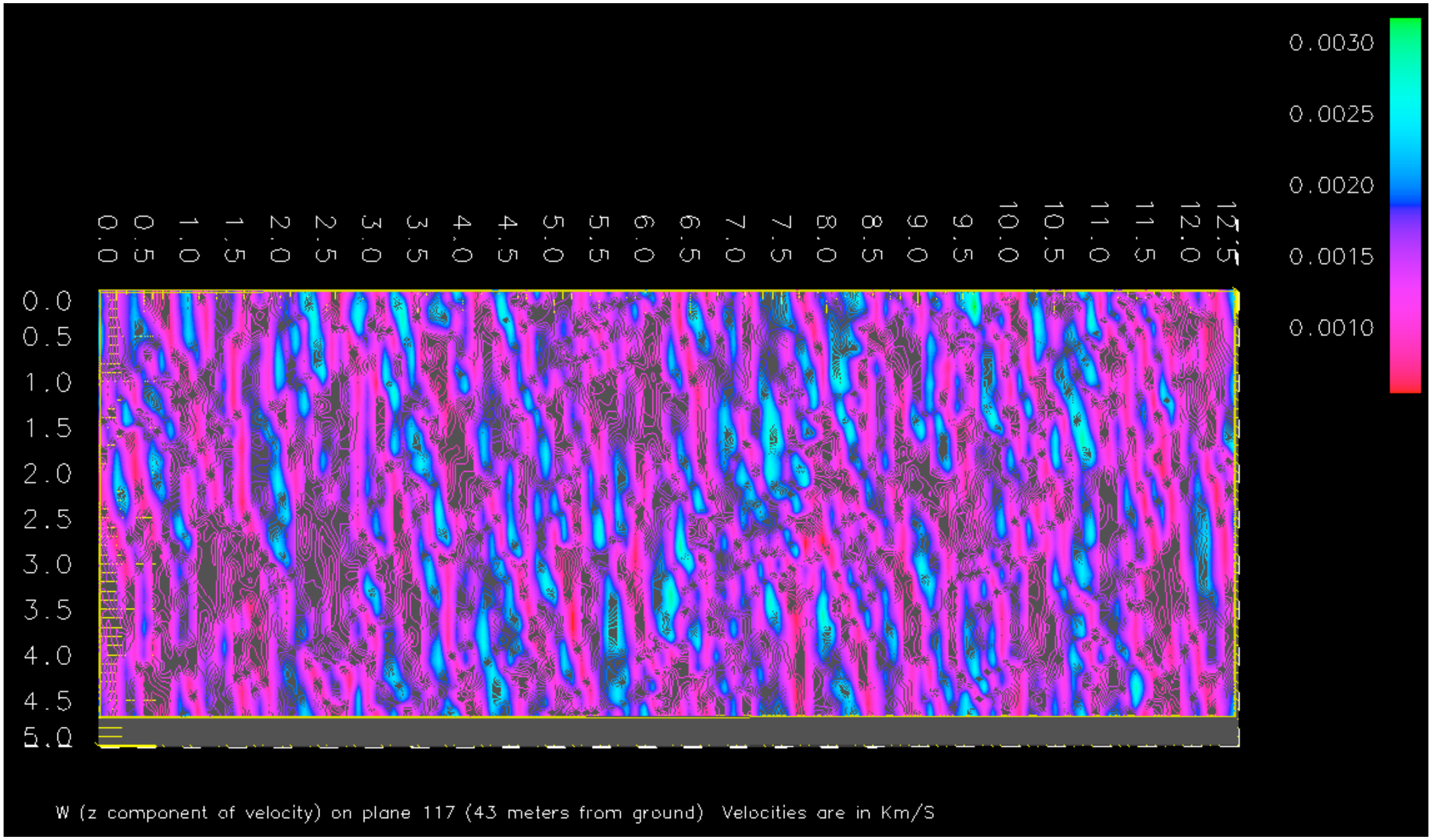
U*/G = 0.067 In good agreement with Coleman et al (JFM, 1990)

Wind Velocity (W) in an x-y plane



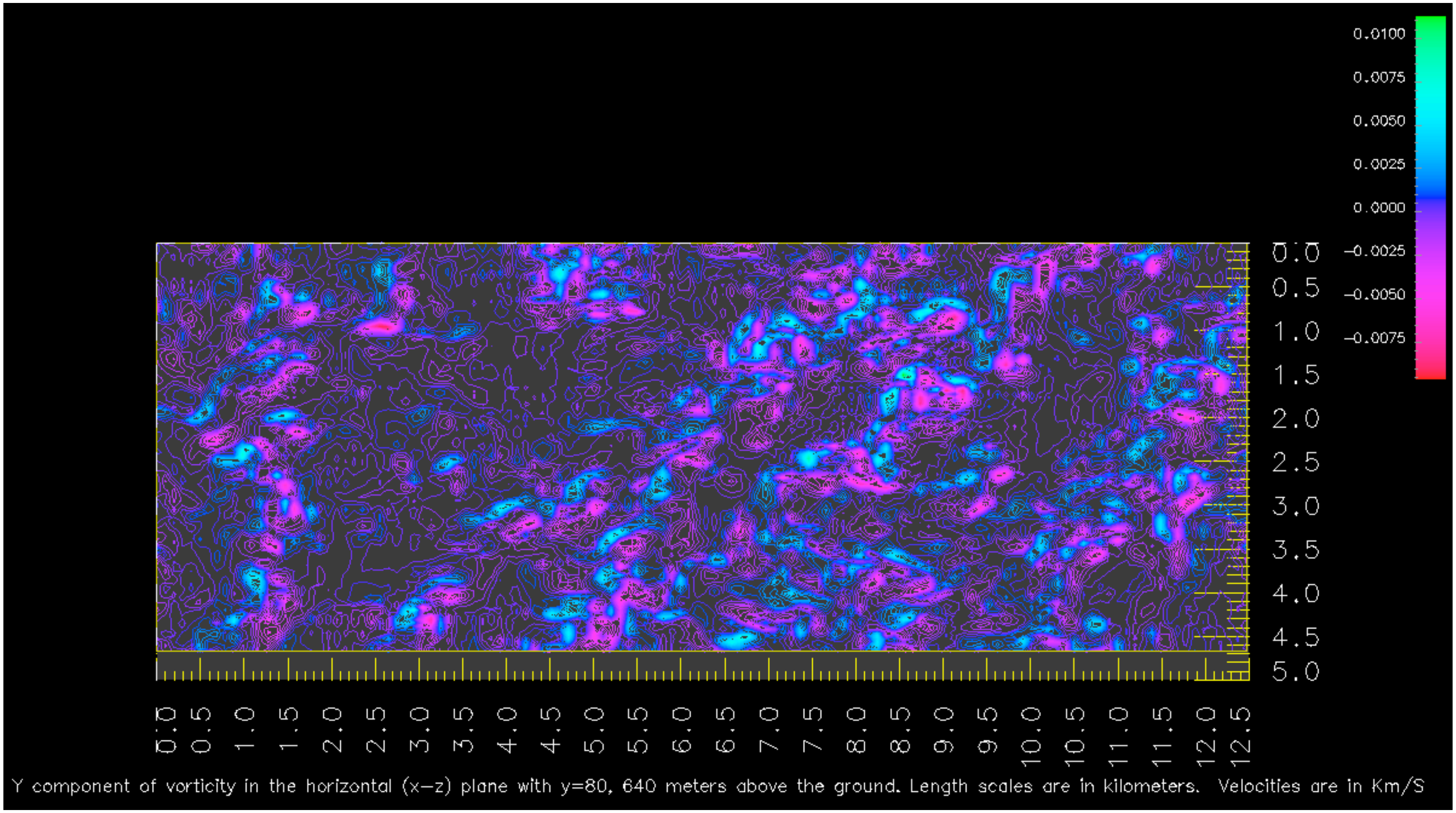
Thin Ekman Layer with turbulent "eruptions"

W in x-z plane at 43 meters



Note highly elongated low speed regions and “gusts”

Vertical vorticity 640 meters in x-z plane



Note “coherent 2D vortices” --- Air-Spikes !?

Outline

- Motivation, Objectives & Guideposts
- Conventional development
- Scalable development
- Applications
- Toward scalable execution
  - A strategy
  - Turbulence at the petascale
- Conclusions & Acknowledgments

Toward Scalable Execution

```
class(Scalar) :: Smoke

Smoke = Smoke + dt*d_dt(Smoke)
```

Strategy:

- Decompose problem into elementary operations.
- Instantiate distributed objects, e.g. via Trilinos.
- Parallelize operators across distributed objects.

Potential pitfalls:

- Cache utilization.
- Combined instructions.

Turbulence at the Petascale

- R. D. Moser* estimates 1500 Petaflop-hours required for DNS at Re_τ=5000, which will achieve asymptotic behavior in the log layer.
- The bottom plane of many ABL simulations lies in the log layer & employs a boundary condition valid at asymptotically high Reynolds number:

$$u^+ = \left(\frac{1}{\kappa} + \frac{\beta}{\mathrm{Re}_\tau} \right) \ln y^+ + \frac{\alpha y^+}{\mathrm{Re}_\tau} + B$$

$$\lim_{\mathrm{Re}_\tau \to \infty} u^+ = \frac{1}{\kappa} \ln y^+ + B$$

*NSF Workshop on Cyber-Fluid Dynamics, Arlington, VA (2006).

Sandia National Laboratories

Conclusions

- Applying Amdahl's law to the *total* solution time suggests that optimizing run time only severely limits speedup.
- The Pareto Principle determines the percentage of code that can be focused on programmability rather than efficiency.
- The global data sharing in conventional development leads to a quadratic search times.
- Enabling an abstract data type calculus
  - Renders bug search times roughly scale-invariant and
  - Limits interface content (developer communications)
- We have demonstrated scalable development on several applications and proposed a path toward scalable execution.

Acknowledgements

- NRL
  - Dr. Robert Handler
  - Dr. Robert Rosenberg
- CUNY
  - Prof. Joel Koplik
  - Ms. Karla Morris & Dr. Xiaofeng Xu
- University of Cyprus
  - Prof. Stavros Kassinos
  - Dr. Irene Moulitsas, Dr. Evaggelos Akylas, & Dr. Hari Radhakrishnan
- University of Belgium
  - Prof. Bernard Knaepen
  - Dr. Ioannis Sarris

Sandia National Laboratories