Fusion:

The App of our Eyes

David Keyes, Reporter and Questioner

On behalf of your CS and Applied Math colleagues

ISOFS Workshop

San Diego, CA

18 September 2002

What we heard yesterday about your problem characteristics

- **Multiple time scales**
  - **Range of about fourteen orders of magnitude**
- **Multiple spatial scales**
  - **Reynolds numbers of 10**8**
- **Linear ill conditioning**
- **Complex geometry and severe anisotropy**
- **Coupled physics, with essential nonlinearities**
- **Variable fidelity models, some with remaining closure questions**
- **Ambition for predictability and design**

Our pat answers

- **Multiple time scales**
  - **Range of about fourteen orders of magnitude**
- **Multiple spatial scales**
  - **Reynolds numbers of 10**
- **Linear**
- **Complex geometry and**
- **Coupled physics, with essential**
- **Variable fidelity**
- **Ambition for predictability and**

Stiff integrators

Adaptivity

Optimal solvers

Unstructured meshing

Physics-based preconditioning

Component-based design

Sensitivity tools

In scalable, distributed memory, cache friendly implementation

More problem characteristics

- **Nonlocal operators**
- **Physical instabilities**
- **High dimensional independent variable space**
- **High dimensional dependent variable space**
- **Mixed dimension code components**
- **Mixed continuum/particle models, perhaps on different co-located meshes**

Our confidence

- **You are devilishly clever**
- **You are foolishly courageous**
- **You are heroically energetic**
- **You care about science, not computing stunts**
- **Your science is *as worthy* of our own research investments *as any we can find* …**
  - **Scientific ripeness**
  - **Methodological ripeness**
  - **Justification of our efforts**
  - **Visibility and external impact**

Our fears

- **You are wedded to what has worked up until now**
- **You regard your own specialty code as the base to which others should adapt theirs**
- **You think that there is a magic CS solution that provides generality of application and ease of use without counting the cost in performance**
- **You really do like F90 ☺**
- **You think 4,096 processors is like 64 processors, only more so**

Our issues

- **High performance**
  - **Single-node performance**
  - **Parallel scalability**
- **Algorithmic optimality**
  - **Discretization complexity**
  - **Solution complexity**
- **Verification of the numerics**
- **Extensibility (to advances in both models and methods)**
- **Reusability (in other problems)**
- **Portability (to other machines)**
- **Ultimate functionality**

Where we would enjoy helping

- **Performance engineering**
  - **Cache and distributed memory friendly implementations**
- **Complex geometry**
  - **Unstructured and multicomponent meshes**
- **Advanced discretizations**
  - **High order, solution-specific schemes**
- **Solution-based adaptivity**
  - **Automated mesh and discretization refinement**
- **Advanced solvers**
  - **Optimal schemes adapted to physics of interest, using all physics available**
- **Coupling issues**
  - **Transferring physical quantities between "natural" representations in different code components**
- **Interpretation of results**
  - **Convergence, conservation, visualization, advanced post-processing, data mining**
- **Going after the ultimate ends**
  - **Sensitivity, stability, design, control, parameter identification, data assimilation, experimental validation, computational steering, scientific discovery**

Why work with us?

- **Logically innermost kernels (e.g., linear algebra) are often the most computationally complex (in terms of memory accesses or flops or both) in the entire code — should be designed from the inside out by experts with right "handles" presented to users**
- **Many widely used methods and software libraries are quite behind the times, algorithmically**
- **Today's components do not "talk to" each other very well**
- **Mixing and matching procedures too often requires mapping data between different storage structures (taxes memory and memory bandwidth)**
- **Many fusion issues are generic to computational science and solutions can be leveraged**

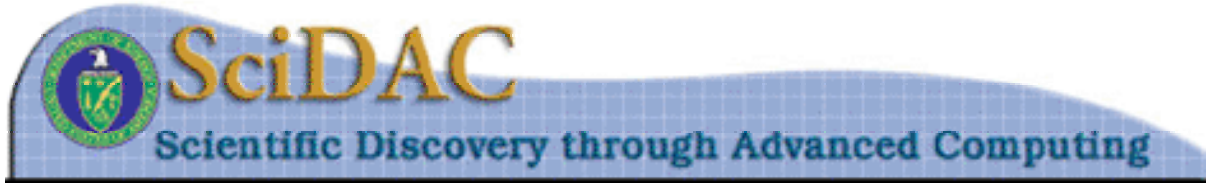
The power of algorithms

- **Since the dawn of digital computers, algorithmic advances have matched architectural advances in increasing the power available to end user scientists**
- **For instance, for a system of *N* linear equations derived from linear finite elements applied to a 3D elliptic PDE:**

Algorithm	Operation Complexity
Banded Gauss Elimination	$O(N^{7/3})$
Gauss Seidel	$O(N^{5/3}\ log(N))$
Optimal SOR	$O(N^{4/3}\ log(N))$
CG/MILU	$O(N^{7/6}\ log(N))$
F-cycle MG	$O(N)$

- **If $N=10^6$ (e.g., a 3D brick with 100 nodes on each side), then the net improvement from banded elimination to multigrid is a *factor of 10^8***
- **For this problem, *this is equivalent to replacing a 1 Mflop/s computer with a 100 Tflop/s computer***

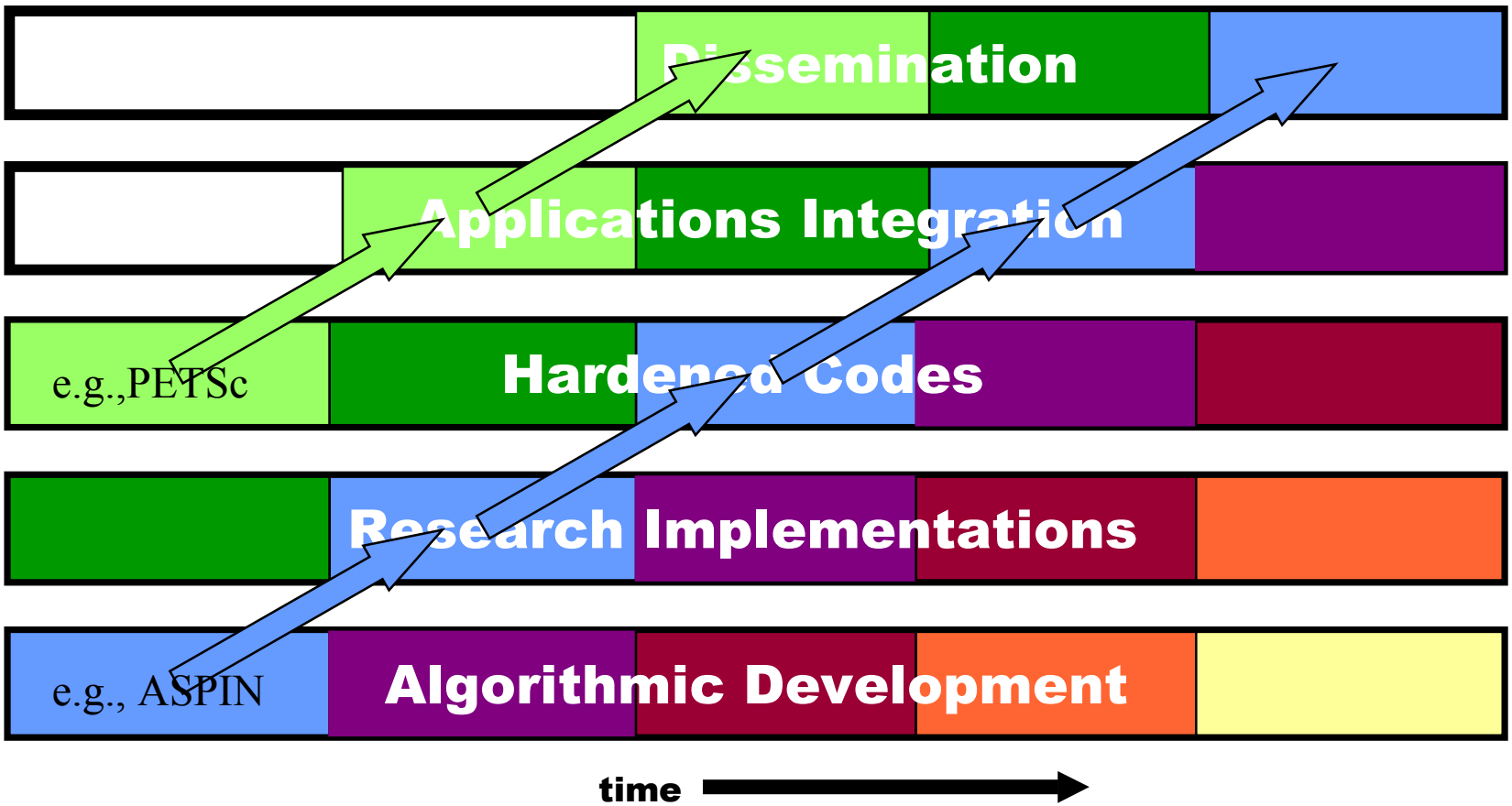
A model some of us like…



- **Lab-university collaborations to develop reusable software "solutions" and partner with application groups**
- **For FY2002, 51 new projects at $57M/year total**
  - **Approximately one-third for applications**
  - **A third for integrated software infrastructure centers (ISICs)**
  - **A third for grid infrastructure and collaboratories**
- **Multi-Tflop/s IBM SP platforms "Seaborg" at NERSC (#3 in latest "Top 500") and "Cheetah" at ORNL available for SciDAC**

Sample Gantt chart for ISICs

Each color module represents an algorithmic research idea on its way to becoming part of a supported community software tool. At any moment (vertical time slice), TOPS has work underway at multiple levels. While some codes are in applications already, they are being improved in functionality and performance as part of the TOPS research agenda.



It's 2002; do you know what your solver is up to?

Has your solver not been updated in the past five years?

Is your solver running at 1-10% of machine peak?

Do you spend more time in your solver than in your physics?

Is your discretization or model fidelity limited by the solver?

Is your time stepping limited by stability?

Are you running loops *around* your analysis code?

Do you care how sensitive to parameters your results are?

If the answer to any of these questions is "yes", you are a potential customer!

Have you thought recently about your discretization?

Have you considered updating it in the past five years?

Have you evaluated spatial and temporal convergence order experimentally for a typical case?

Do you lack solution-based error controls?

Is is difficult for you to vary scheme order and mesh type to follow up on scientific hypotheses or numerical hunches?

Is is difficult to combine methods, like Eulerian fields and Lagrangian particles or fronts?

Do you spend more time setting up your problem than running it?

If the answer to any of these questions is "yes", you are a potential customer!

CS/AppMath project goals/success metrics

We will have succeeded if (you) users —

- **Understand range of algorithmic options and their tradeoffs *(e.g., memory vs. time, memory bandwidth vs. flop/s, communication vs. flops, inner iteration work vs. outer)***
- **Can try all reasonable options from different sources easily without recoding or extensive recompilation**
- **Know how your discretizations are converging**
- **Know how your solvers are performing**
- **Spend more time in your physics than in meshing and solving**
- **Are intelligently driving algorithmic research, and publishing joint papers with CS and AppMath researchers**
- **Can simulate *truly new physics*, as limits are steadily pushed back (finer meshes, higher fidelity models, complex coupling, etc.)**

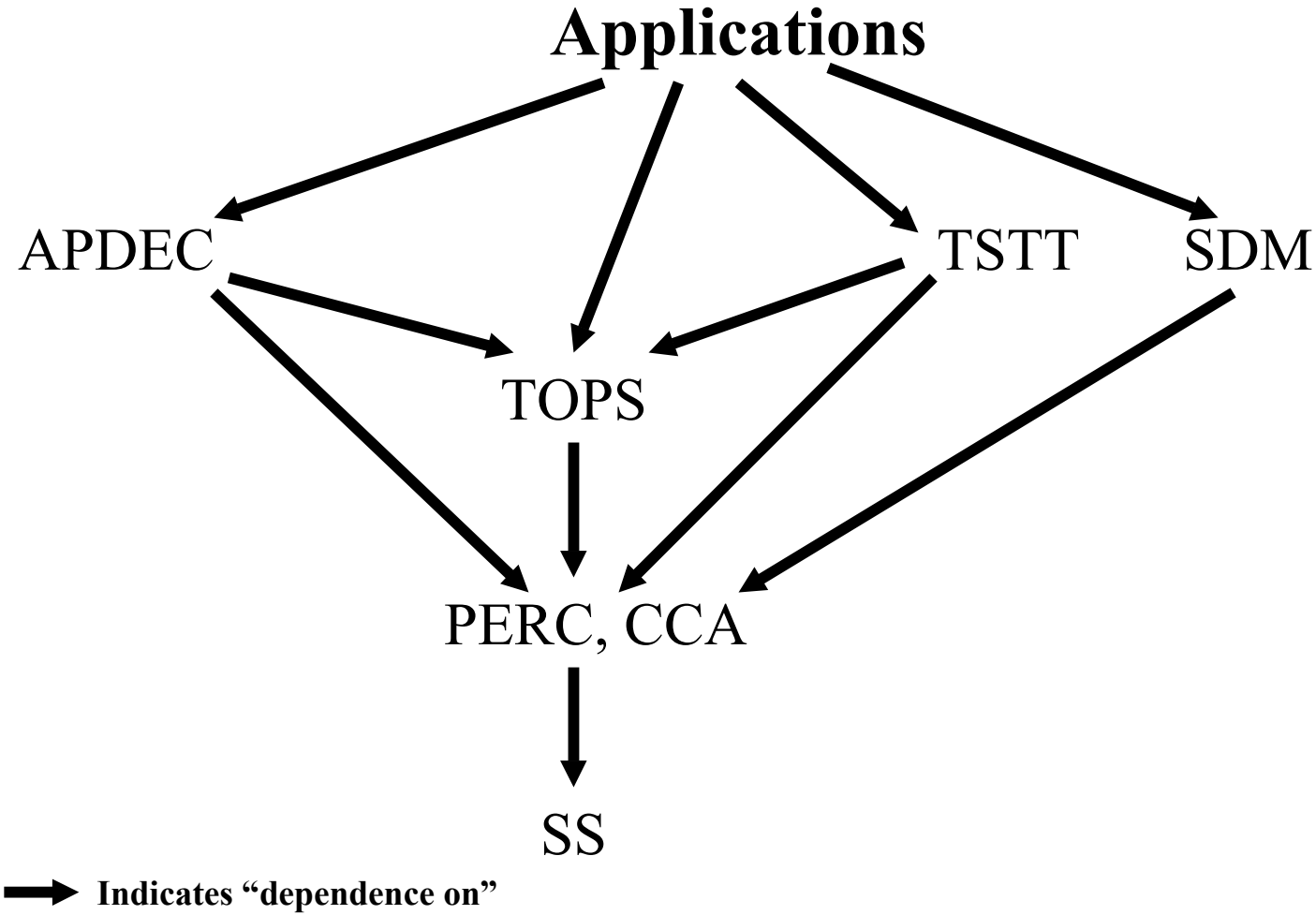
Interacting with SciDAC ISICs

Applications

APDEC TSTT TOPS SDM PERC CCA SS

Indicates "dependence on"

One view of interacting with ISICs



Expectations ISICs have of users

- **Be willing to experiment with novel algorithmic choices – optimality is *rarely* achieved beyond model problems without interplay between physics and algorithmics!**
- **Adopt flexible, extensible programming styles in which algorithmic and data structures are not hardwired**
- **Be willing to let us play with the real code you care about, but be willing, as well to abstract out relevant compact tests**
- **Be willing to make concrete requests, to understand that requests must be prioritized, and to work with us in addressing the high priority requests**
- **If possible, *profile, profile, profile* before seeking help**

What we believe

- **Many of us came to work on solvers through interests in applications**
- **What we believe about …**
  - **applications**
  - **users**
  - **solvers**
  - **legacy codes**
  - **software**

  … will impact how comfortable you are collaborating with us

What we believe about apps

- **Solution of a system of PDEs is rarely a goal in itself**
  - **PDEs are solved to derive various outputs from specified inputs**
  - **Actual goal is characterization of a response surface or a design or control strategy**
  - **Together with analysis, sensitivities and stability are often desired**

⇒ **Software tools for PDE solution should also support related follow-on desires**

- **No general purpose PDE solver can anticipate all needs**
  - **Why we have *national laboratories*, not *numerical libraries* for PDEs today**
  - **A PDE solver improves with user interaction**
  - **Pace of algorithmic development is very rapid**

⇒ **Extensibility is important**

What we believe about users

- **Solvers are used by people of varying numerical backgrounds**
  - **Some expect MATLAB-like defaults**
  - **Others want to control everything, e.g., even varying the type of smoother and number of smoothings on different levels of a multigrid algorithm**

⇒ **Multilayered software design is important**

- **Users' demand for resolution is virtually insatiable**
  - **Relieving resolution requirements with modeling (e.g., turbulence closures, homogenization) only defers the demand for resolution to the next level**
  - **Validating such models requires high resolution**

⇒ **Processor scalability and algorithmic scalability (optimality) are critical**

What we believe about legacy code

- **Porting to a scalable framework does not mean starting from scratch**
  - **High-value meshing and physics routines in original languages can be substantially preserved**
  - **Partitioning, reordering and mapping onto distributed data structures (that we may provide) adds code but little runtime**

⇒ **Distributions should include code samples exemplifying "separation of concerns"**

- **Legacy solvers may be limiting resolution, accuracy, and generality of modeling overall**
  - **Replacing the solver may "solve" several other issues**
  - **However, pieces of the legacy solver may have value as part of a preconditioner**

⇒ **Solver toolkits should include "shells" for callbacks to high value legacy routines**

What we believe about solvers

- **Solvers are employed as part of a larger code**
  - **Solver library is not only library to be linked**
  - **Solvers may be called in multiple, nested places**
  - **Solvers typically make callbacks**
  - **Solvers should be swappable**

⇒ **Solver threads must not interfere with other component threads, including other active instances of themselves**

- **Solvers are employed in many ways over the life cycle of an applications code**
  - **During development and upgrading, robustness (of the solver) and verbose diagnostics are important**
  - **During production, solvers are streamlined for performance**

⇒ **Tunability is important**

What we believe about numerical software

- **A continuous operator may appear in a discrete code in many different instances**
  - **Optimal algorithms tend to be hierarchical and nested iterative**
  - **Processor-scalable algorithms tend to be domain-decomposed and concurrent iterative**
  - **Majority of progress towards desired highly resolved, high fidelity result occurs through cost-effective low resolution, low fidelity parallel efficient stages**

⇒ **Operator abstractions and recurrence are important**

- **Hardware changes many times over the life cycle of a software package**
  - **Processors, memory, and networks evolve annually**
  - **Machines are replaced every 3-5 years at major DOE centers**
  - **Codes persist for decades**

⇒ **Portability is critical**

Exciting time for enabling technologies

SciDAC application groups have been chartered to build new and improved *COMMUNITY CODES*. Such codes, such as NWCHEM, consume hundreds of person-years of development, run at hundreds of installations, are given large fractions of community compute resources for decades, and acquire an "authority" that can *enable or limit* what is *done and accepted* as science in their respective communities. Except at the beginning, it is difficult to promote major algorithmic ideas in such codes, since change is expensive and sometimes resisted.

ISIC groups have a chance, due to the interdependence *built into the SciDAC program structure*, to simultaneously influence many of these codes, by delivering software incorporating optimal algorithms that may be reused across many applications. Improvements driven by one application will be available to all. While you are building community codes, this is our chance to build a *CODE COMMUNITY*!

Related URLs

- **Personal homepage: papers, talks, etc.**

  http://www.math.odu.edu/~keyes

- **SciDAC initiative**

  http://www.science.doe.gov/scidac

Summary of Roundtables

Interpretation & Algorithms Roundtable

- **Points of greatest need**
  - **Specialized meshing and discretization capability**
  - **Specialized solver capability**
- **Both driven largely by anisotropy**
  - **General numerical PDE techniques may not apply due to failure to respect the highly anisotropic transport *along* versus *across* magnetic flux surfaces**
  - **Fundamental; basis of magnetic confinement**
- **Consequences**
  - **Small errors lead to artificially enhanced transport**
  - **Anisotropy hard for standard scalable solvers**

Interpretation & Algorithms Roundtable

- **Additional points of need**
  - **General meshing tools for toroidal and topologically toroidal geometries, fully structured and hybrid structured-unstructured (in poloidal plane)**
  - **Sophisticated interpolation tools to move fields from one discretized representation to another, co-located in same domain on different meshes**
    - ✓ **Especially including experimentally derived data, for validation**
  - **Interpolation tools to change dependent variables from one physical model to another (a fundamental part of specialized fusion/MHD framework??)**
  - **Hybrid continuum-particle solvers**

Interpretation & Algorithms Roundtable

- **Additional points of need, cont.**
  - **Fast *curl-curl* solvers**
  - **Stiff MOL solvers for integration of either compressive Alfven waves in the poloidal plane (CFL ratio of 10-100) or both compressive and shear Alfven waves (CFL ratio of 1000), to follow slower dynamically relevant timescales more efficiently**
  - **Homogenization and related upscaling techniques**
  - **Operator-adaptive multilevel methods for ill conditioned linear systems**

Interpretation & Algorithms Roundtable

- **Additional points of need, cont.**
  - **High-order hyperbolic conservation law integrators (for *v.grad*) that include field effects**
  - **Nonlinearly consistent iterative methods for coupled physics, with essential "two-way" finite amplitude nonlinearities**
    - ✓ **physics-based preconditioning for Newton-type methods, (harnessing "solvers" in current component codes as part of overall operator-split preconditioner)**
  - **Methods for design/optimization subject to satisfying the high-dimensional constraints coming from the system of governing equations**

MHD state of art for gridding/discretization

- **Not dynamic; good approximation to flux surfaces are known in advance**
- **Ill-posed to grid based only upon field lines (e.g., using some Brackbill-Saltzman variational scheme), since the field lines do not perfectly close**
- **Hybrid *h-p* schemes provide sufficient accuracy to "not sweat the small stuff"**
- **High order FE methods are "good enough" to not require exact discrete satisfaction of *div(B)=0*, etc.**
- **Useful scheme for high-order hyperbolic solvers and for (logically) Cartesian AMR would be field- and body-fitted coordinates in the poloidal planes and (possibly twisted) cylindrical extensions in the toroidal direction**

We did not hear too much about but wish to promote thoughts about…

- **Large-scale eigensolvers, stability analyses**
- **Sensitivity analyses**
- **Visualization**
- **Automated data mining for physical features**
- **Boundary and body control of plasmas through formal optimization techniques**
- **Parameter identification**
- **Assimilating experimental data into simulations**
- **Computational steering**