

# Mathematical Issues and Opportunities

SWIM Meeting

16 October 2007

3:30-5:30pm

Moderated by David Keyes

# What is a “math issue”?

- In the computational science “chain of being”, the mathematics generally lies in between the *science* and the *computing*, but transitions on both sides are blurry
  - this bluriness is one of the reasons SciDAC is needed and resonates with us (and our managers)
- If one arbitrarily “deblurs” by drawing a sharp interface, one is probably throwing away opportunities to reformulate the overall problem in useful ways
- When the issue can be abstracted to something that can be studied apart from any particular physics, and apart from any particular software/hardware implementation, it can definitely be called a “math issue” and handed off (for a season, with a re-entry point capability well defined)
- Blurrier issues at the physics-math interface are also “math issues” for the purpose of today’s discussion

# Goals

- Identify and prioritize math issues that are now or shortly (upon further *model advances, scaling up, and coupling up* existing modules) will be impediments to progress
- Abstract the issues to see if there are “off the shelf” solutions from other applications
- For the nontrivial remainder, consider forming subteams

# Proposed format

- Collect and rank issues
  - Six issues have been suggested in advance
  - Are there others?
- Run through topics sequentially, seeding the discussion with a few-slide overview per issue
- Keyes is moderator, not primary presenter for most topics; clarification or even definition for several of the issues will have to come “from the floor”
- We will summarize and distribute current and future action items, short- and long-term

# Digression:

## A commendation from ASCR PIs

- Yesterday and today, the PIs and/or their delegates of the three math CETs, the six computer science CETs, and the four institutes met with ASCR managers to assess SciDAC-2 progress and establish “best practices” from pooled experience
  - A resonant theme was the treatment of the math and CS groups as “plumbers” by the physics groups
  - Fusion PIs, however, got good marks from everyone in ASCR who has partnered with fusion
  - Fusion PIs also got credit for their willingness and ability to abstract model problems for ASCR mathematicians
  - Fusion PIs get good marks from me on having math and CS sophistication “in house”
- Maybe there is a correlation between being sophisticated in math and CS and being good partners with math and CS
  - You know that it’s not just “plumbing”

# Topics (to be ranked)

- Data compression through the singular value decomposition (Berry, Spong)
- Conversion of particle data in physical and phase space into multivariate distribution functions (Berry, Keyes)
  - need sufficient derivatives for avoidance of instabilities that are numerical artifacts, while maintaining positivity and conservation
  - SVD and constrained splines under consideration
- Preconditioning for  $C^1$  version of M3D, based on the successful solution of its blocks in the 2D version (Jardin)
- Equation-free, projective integration version of NUBEAM (Jardin, Batchelor)
- Implicit code coupling, as will be required in SWIM, for instance, between TSC and CQL3D (Jardin, Batchelor)
- Load balancing for multiphase multiphysics computations with incommensurate granularities between phases, and incommensurate times per phase (Jardin)
- Coupling particle-based (energetic particles) and F-P schemes (low energy particles) – multiscale and multimodel (Berry)

# Data compression through SVD

- Berry, Spong proposed
- Evidently there was a talk already this morning

# Distribution functions from particles

- Berry, Keyes proposed
- Conversion of particle data in physical and phase space into multivariate distribution functions that possess sufficient derivatives for avoidance of instabilities that are numerical artifacts, while maintaining positivity and conservation.
- Possible approaches
  - constrained splines
  - singular value decomposition
- Questions
  - In what codes does this come up?
  - What is the number of independent variables in the distribution function?
  - What is done today and what improvements are needed?
  - What is a good test suite?
  - Who are the physics contacts person?

# Conversion of representations: particle data to velocity PDF

Formulation:

- B-spline basis with least squares fit to data
- conservation through penalization
- positivity through working with the log of the pdf (“density estimation” technique borrowed from financial modeling)

Leads to nonlinear rootfinding problem solved with Newton’s method

Currently demonstrated in 2D with explicit Jacobian construction; can be made Jacobian-free and parallel

Needs to be made more robust (“globalization”)

# Conversion of representations: particle data to velocity PDF

## Formulation

B-spline basis with least squares fit to data  
conservation through penalization

positivity through working with the log of the pdf

Consider the regression of particles in velocity space on a set of B-splines of the dimension of the phase space and appropriate approximation order, e.g., cubics

$$f(v) = \sum_i a_i B_i(v)$$

$$S(a) = \sum_k [y_k - \sum_i a_i B_i(v_k)]^2 - \lambda P(a)$$

$f(v)$  can be negative, and is in the tails, in practice, so set  
 $f = \exp(F)$  and fit  $f$

# Preconditioning $C^1$ finite elements

- Jardin proposed
- Preconditioning for  $C^1$  version of M3D, based on the successful solution of blocks of the 2D version by SuperLU
- Possible approach
  - block diagonal preconditioning in PETSc, where the blocks are instances of the current 2D problem with augmentation in off-diagonals
  - precondition high-order discretizations with low-order, after conversion to nodal basis
- Questions
  - What are the states of the current 2D and 3D implementations?
  - Who is the physics contact person?

# Projective integration version of NUBEAM

- Jardin, Batchelor proposed
- Equation-free, projective integration version of NUBEAM
- Possible approach
  - standard projective integration methods for two-scale problems in which the dynamics is well modeled only at the fine scale and the observables are needed only at the coarse scales
  - Needed are: stiff coarse scale integrator, first-principles fine-scale integrator, averaging/restriction function, lifting/prolongation function
- Questions:
  - What has been done so far in this model?
  - What would make a good lifting procedure to reinitialize the fine scale distribution from the coarse observables?
  - Who is the physics contact?

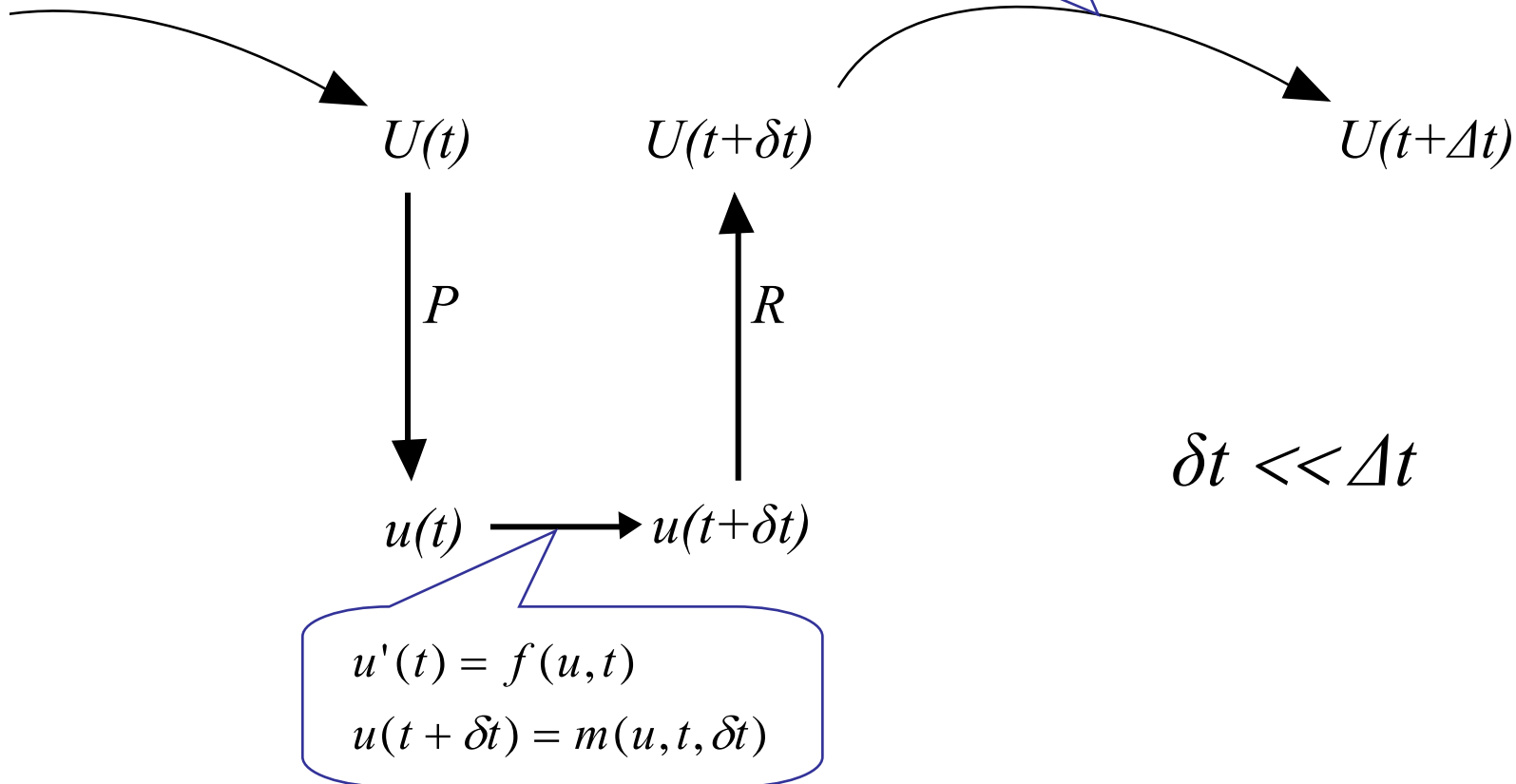
# Gear & Kevrekidis' "projective integration"

- Given: a first-principles model for microscopic quantities  $\mathbf{u}(t)$ , whose dynamics,  $\mathbf{u}' = \mathbf{f}(\mathbf{u}, t)$ , we believe but cannot afford to integrate over long time-scales of interest (e.g., timestep is Courant-stability or accuracy bounded)
- Desired: an effective model for macroscopic quantities,  $\mathbf{U}(t)$ , evolved over long times, for which we may lack the dynamics,  $\mathbf{U}' = \mathbf{F}(\mathbf{U}, t)$
- Projective integration probes the "fine" model over short intervals to advance the "coarse" model
- This can be "equation-free" in the sense that the coarse model dynamics is defined implicitly
- Key need: interpolation operators to convert between the variables of the two models
  - "restrict" from fine to coarse,  $\mathbf{R}$
  - "prolong" from coarse to fine,  $\mathbf{P}$  (sometimes called "lifting")

# Projective Integration “cartoon”

$$\frac{\delta U}{\delta t} \approx \frac{1}{\delta t} [U(t + \delta t) - U(t)]$$

$$U(t + \Delta t) = U(t) + \frac{\delta U}{\delta t} \cdot \Delta t$$



# Projective integration examples

- Fluids with particles/fields:  $u = \{ x_i, v_i \}$  ,  $U = \{ \rho(x), v(x) \}$ 
  - For  $R$  : take moments
  - For  $P$  : populate distributions
- Mechanics
  - $U$  as Eulerian fields in elasticity,  $u$  as molecules in molecular dynamics
- Biology
  - $U$  as concentration in advection-diffusion-reaction system,  $u$  as microbial agents

# Equation-free projective integration (EFPI) in MHD

- Micro and macro scales strongly linked
- Speedups possible while maintaining fidelity to microscale (see Shay, Drake & Dorland, 2006)
  - 1D model problem: kinetic code used both as the baseline “correct” model and as the inner “fine” model of an EFPI approach
  - Particles initialized with Maxwellian with velocity shift, preserving number density, velocity, and temperature
  - Explicit timestepping
- Naïve explicit formulation (previous cartoon) can be made implicit with Jacobian-free Newton-Krylov (JFNK) solver at coarse scale

# TSC-CQL3D coupling

- Jardin, Batchelor proposed
- Implicit code coupling, as will be required in SWIM, for instance, between TSC and CQL3D
- Possible approach (depending upon availability of Jacobians in individual codes)
  - nonlinear Schwarz
- Questions:
  - Is there a reduced-order, low-dimensional model of the potential coupling instabilities?
  - Who is the physics contact person?

# Multiphysics coupling: partial elimination

- Consider system  $F(u) = 0$  partitioned by physics as

$$\begin{cases} F_1(u_1, u_2) = 0 \\ F_2(u_1, u_2) = 0 \end{cases}$$

- Can formally solve for  $u_1$  in  $F_1(u_1, u_2) = 0$

$$u_1 \equiv G(u_2)$$

- Then second equation is  $F_2(G(u_2), u_2) = 0$

- Jacobian

$$\frac{dF_2}{du_2} = \frac{\partial F_2}{\partial u_1} \frac{\partial G}{\partial u_2} + \frac{\partial F_2}{\partial u_2}$$

can be applied to a vector in matrix-free manner

# Multiphysics coupling: nonlinear GS

- In previous notation, given initial iterate  $\{u_1^0, u_2^0\}$
- For  $k=1, 2, \dots$ , until convergence, do
  - Solve for  $v$  in  $F_1(v, u_2^{k-1}) = 0$
  - Solve for  $w$  in  $F_2(v, w) = 0$
- Then

$$\{u_1^k, u_2^k\} = \{v, w\}$$

# Multiphysics coupling: nonlinear Schwarz

- Given initial iterate  $\{u_1^0, u_2^0\}$
- For  $k=1, 2, \dots$ , until convergence, do
  - Define  $G_1(u_1, u_2) \equiv \delta u_1$  by  $F_1(u_1^{k-1} + \delta u_1, u_2^{k-1}) = 0$
  - Define  $G_2(u_1, u_2) \equiv \delta u_2$  by  $F_2(u_1^{k-1}, u_2^{k-1} + \delta u_2) = 0$
- Then solve  $\begin{cases} G_1(u, v) = 0 \\ G_2(u, v) = 0 \end{cases}$  in matrix-free manner
- Jacobian:
 
$$\begin{bmatrix} \frac{\partial G_1}{\partial u} & \frac{\partial G_1}{\partial v} \\ \frac{\partial G_2}{\partial u} & \frac{\partial G_2}{\partial v} \end{bmatrix} \approx \begin{bmatrix} I & \left(\frac{\partial F_1}{\partial u}\right)^{-1} \frac{\partial F_1}{\partial v} \\ \left(\frac{\partial F_2}{\partial v}\right)^{-1} \frac{\partial F_2}{\partial u} & I \end{bmatrix}$$
- Finally  $\{u_1^k, u_2^k\} = \{v, w\}$

# Load balancing for incommensurate phases

- Jardin proposed
- Load balancing for multiphase multiphysics computations with incommensurate granularities between phases, and incommensurate times per phase

# Illustration from Steve

Require each component to use the number of processors it needs in order to finish its calculation in a standard time unit. Say, 10 min. Then, every 10 min, the job will be using a different component.

A --> B --> C --> D --> A --> B --> ... (repeats each 10 min time cycle)

Then, launch 3 more jobs using these same processors, but staggered in time:

A --> B --> C --> D --> A --> ...

A --> B --> C --> D --> ...

A --> B --> C --> ...

Thus, at any given instant, we will be running 1 each of processes A-D, thus using all the processors all the time. When time completes, we will have run 4 jobs with 4 different sets of input data.