Bend but don't break: Prospects for resilience without recovery in algorithms for hyperbolic systems

2015 Salishan Conference on High-Speed Computing

28 April 2015

J. A. F. Hittinger and J. Loffeld

Lawrence Livermore National Laboratory

LLNL-PRES-670050

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

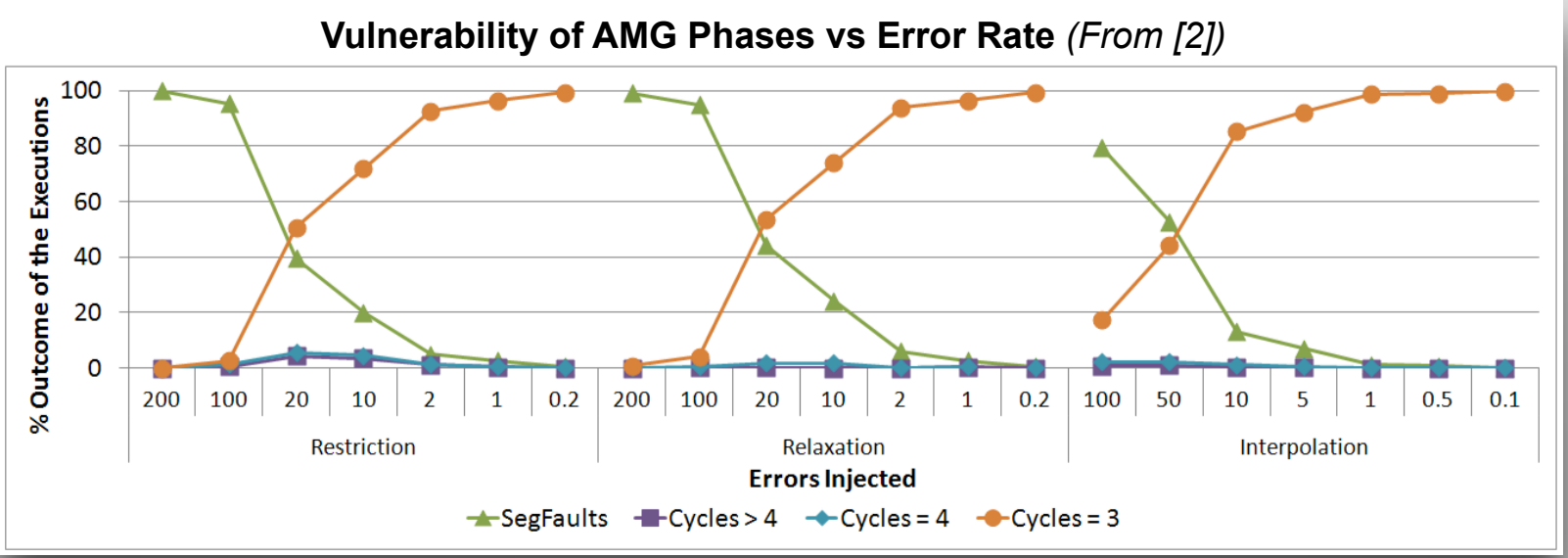
Don't perform CPR (Check-Point Restart) if you don't have too!

- **Scalable *Detect and Rollback* strategies are necessary for some types of faults**
  - Node failures
  - Soft faults leading to segfaults, kernel panics, unrecoverable corruption in data
- **Some Silent Data Corruptions (SDC) allow for more nuanced responses**
  - Local re-computation may be more efficient
  - Masked errors may need no correction

Credit: Ilin Sergey/Shutterstock

Let's consider SDC resulting from transient errors in computations

Some iterative solver algorithms can compute through Silent Data Corruptions

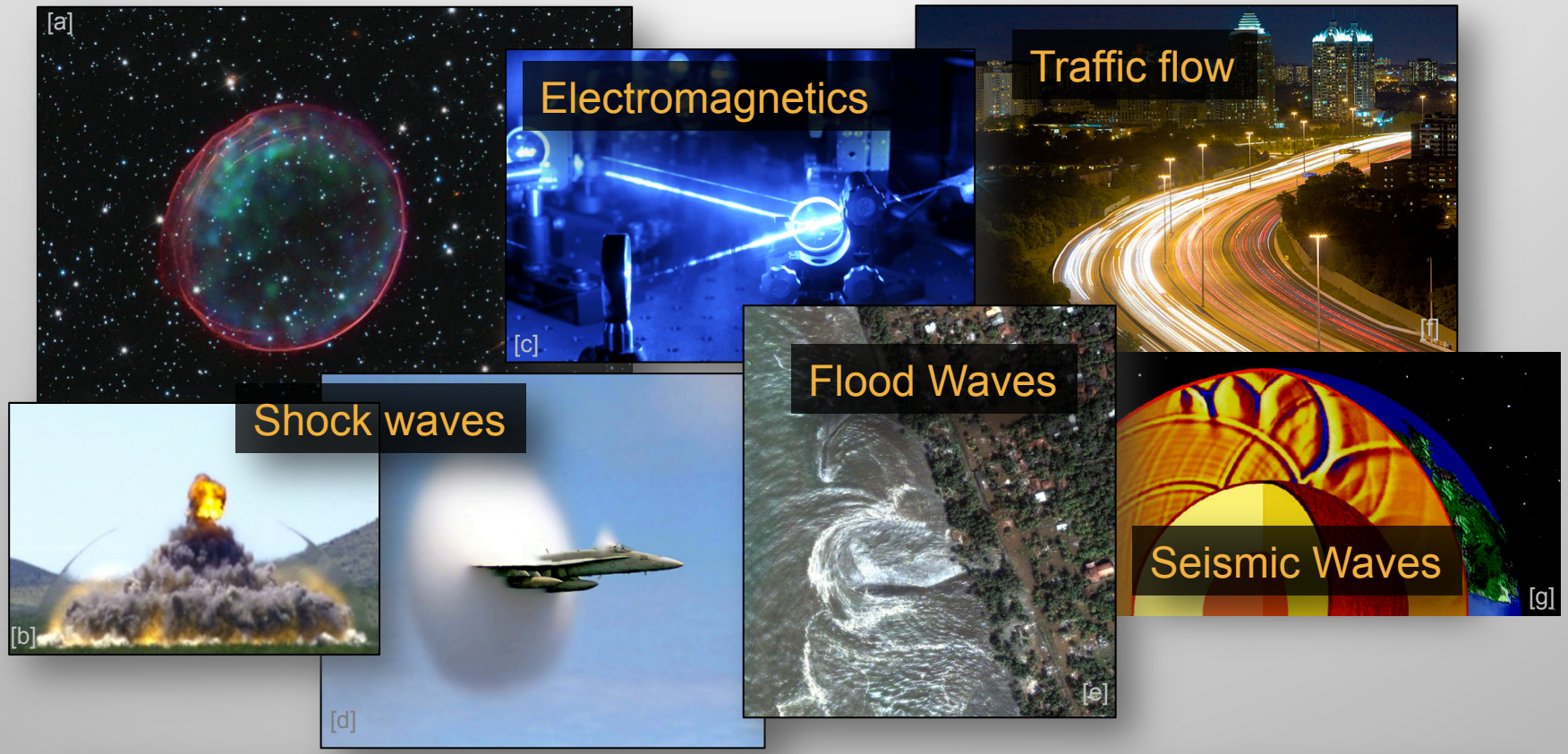


- Fault-tolerant variants of iterative solvers such as FT-GMRES [1] and Algebraic Multigrid (AMG) [2] provide "eventual" convergence
- Suitable for parabolic (diffusion) and elliptic (equilibrium) problems

Can we just "compute through" SDCs with algorithms for hyperbolic problems?

[1] Bridges, Ferreira, Heroux, and Hoemmen, CoRR 2012
[2] Casas, De Supinski, Bronevetsky, and Schulz, ICS 2012

Hyperbolic systems of equations describe wave propagation phenomena

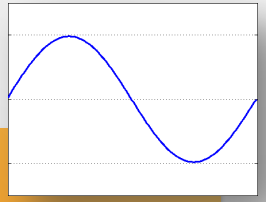


We will consider hyperbolic systems of conservation laws (HSCL) that admit shocks

Image sources listed in Acknowledgments

Hyperbolic systems have a different character – and require different algorithms

$$u_t = u_{xx}$$

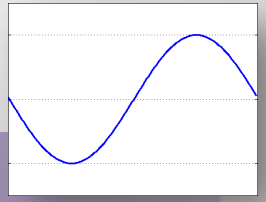


Parabolic

- Dissipative – "to slump"
- *Infinite* wave speeds
- Global coupling
- Advanced *implicitly*
- Discretization leads to *coupled system* of algebraic equations
- Requires nonlinear and linear solvers *for the system*
- Solvers are often *iterative*

$$Mu^{n+1} = Nu^n$$

$$u_t + u_x = 0$$



Hyperbolic

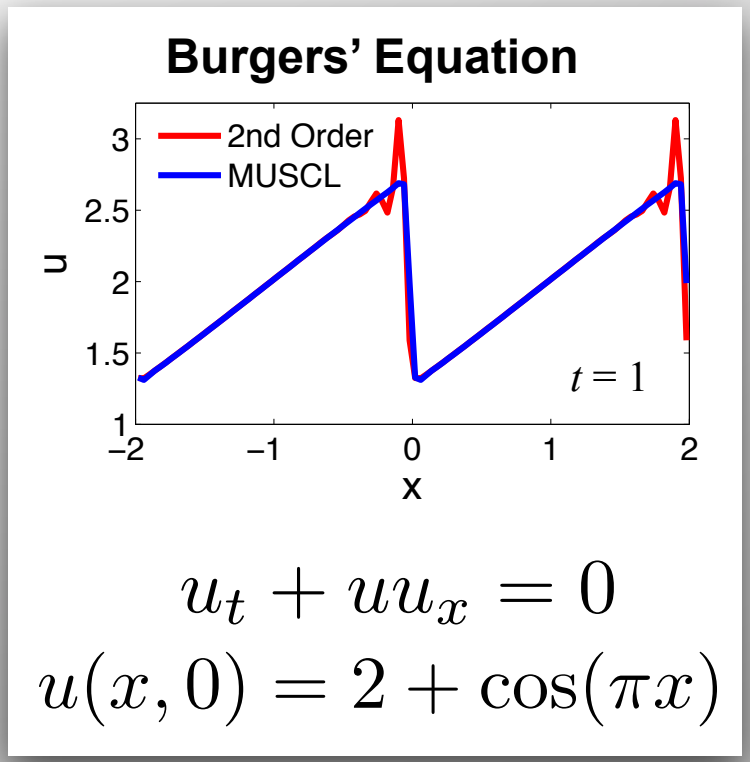
- Non-dissipative
- *Finite* wave speeds
- *Finite* domain of dependence/influence
- Advanced *explicitly*
- Discretization leads to *local* update equations
- Nonlinear or linear solver use is strictly local (if at all)
- Solvers are *direct*

$$u^{n+1} = Nu^n$$

We can't rely on iteration to control SDCs for hyperbolic systems

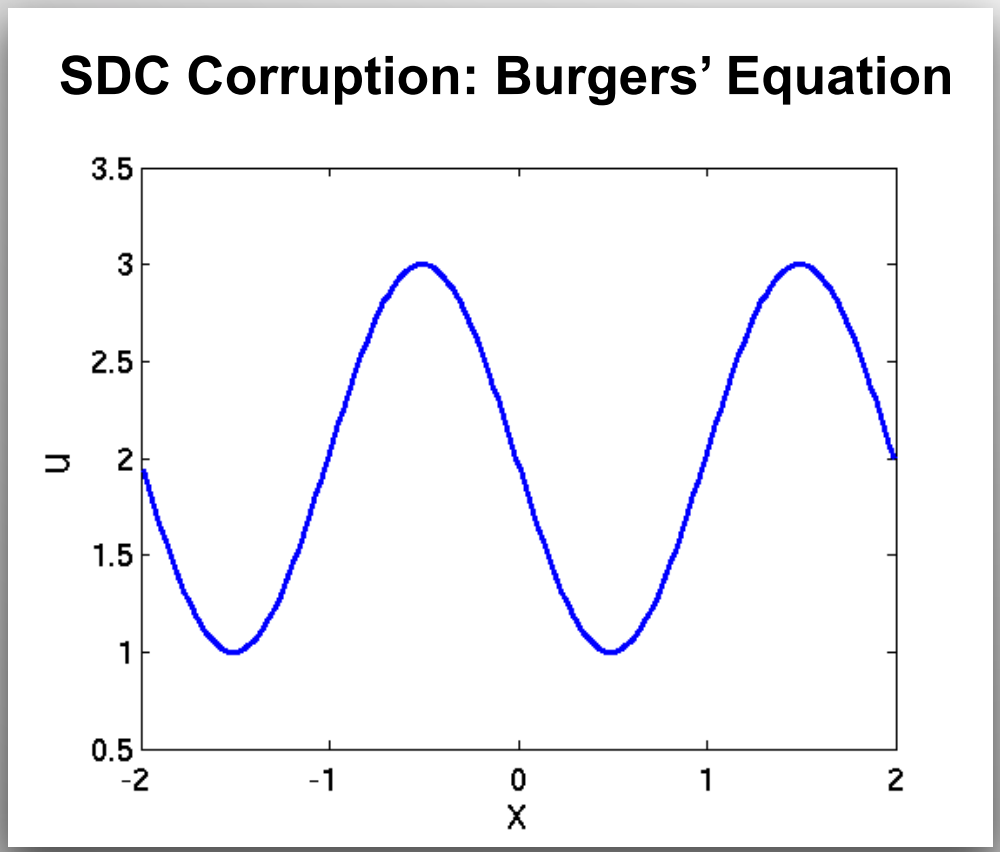
However, shock-capturing algorithms have potentially useful features

- **Artificial dissipation**
  - Nonlinearity pumps energy into higher wavenumbers
  - Eventually, these wavenumbers cannot be resolved on a fixed grid
  - Strongly damp anything not well-resolved
- **Smoothness detection**
  - Higher-order schemes produce unphysical oscillations at shocks
  - Preserve monotonicity at shocks by dropping order
  - Requires detecting the smoothness of the solution



How can we take advantage of these existing techniques?

Standard shock-capturing algorithms already provide some robustness against SDC



- Godunov method with Van Leer limited slope reconstruction (MUSCL)
- $O(1)$ spike added to single flux near $x = 0$ every 30 time steps
- Solution remains stable
- Spike is mostly damped in a dozen time steps
- Solution shows some permanent distortion

Can we fortify these methods to deal with SDC more effectively?

Physical simulations *always* have error

Emel Ataç Tunaboylu, Dreamstime.com

Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful.

- G. E. P. Box and N. R. Draper, *Empirical Model-building and Response Surfaces* (1987)

What's one more error among friends?

We can use the fact that many approximations are made to simulate physical systems

$$u(x,t) = u_i^n + \epsilon_{\mathrm{T}} + \cancel{\epsilon_{\mathrm{I}}} + \epsilon_{\mathrm{R}} + \epsilon_{\mathrm{F}}$$

Data corruption

Truncation Error

Iteration Error

Roundoff Error

Mathematical Model → Discrete Approximation → Approximate solvers → Finite-precision arithmetic

We don't care about SDCs if they are smaller than the other (controlled) errors in our approximation

NB: Stability means that the solution process does not amplify errors

Each time step for a Hyperbolic System of Conservation Laws has three main components

HSCL in flux-divergence form

$$u_t + \nabla \cdot f(u) = 0$$

Exact conservative update

$$\bar{u}_j^{n+1} = \bar{u}_j^n - \frac{\Delta t}{\Delta x}\left[\hat{f}_{j+1/2} - \hat{f}_{j-1/2}\right]$$

Approximate interface flux

$$\hat{f}_{j+1/2} = g_{j+1/2}(\bar{u}^n) + \underbrace{\mathcal{O}(\Delta x^p) + \mathcal{O}(\Delta t^q)}_{\text{truncation error}}$$

while $t \le t_{\text{final}}$ **do**

Compute approximate fluxes

Compute conservative update

Compute next time step

end

Where most cost is incurred

Changes solution state

Important to make progress

Protecting each of these ensures protection of entire step

The growth of time steps is already limited

Time Step Algorithm

for each cell
  Compute Δt_j
  if $\Delta t_{\text{stable}} > \Delta t_j$
    $\Delta t_{\text{stable}} := \Delta t_j$
  end if
end for
$\Delta t^{n+1} := \min(\ \sigma\ \Delta t_{stable},\ (1+\beta)\,\Delta t^n)$

- Time steps are constrained globally by local *linear* stability:

$$\Delta t_j = C\Delta x/\lambda_j$$
$$\Delta t_{\text{stable}} = \min_j \Delta t_j$$
$$\lambda_j : \text{max wave speed in cell } j$$

- Too large a time step will cause (detectable) blow-up

Only a fraction of the stable time step is taken
$\sigma \approx 0.9$

Relative time step growth is clamped
$\beta \approx 0.1$

The real concern is to protect against overly severe time step restrictions

Time steps can be protected by using history information and piecewise smoothness

- Too small a time step limits progress
- Use a floor to detect of sudden reductions

$$\Delta t_{\text{stable}} \leq (1-\alpha)\Delta t^n$$

Detect and Correct

- Store location of minimum Δt
- On detection, recompute minimum Δt
- If value not repeated, redo full Δt computation

Detect and Defer

- Store previous time step Δt^{n-1}
- Log detection on step n
- If time step recovers, use smaller of Δt^{n-1} and Δt^{n+1}
- Cost is extra update step(s)

Box Length	Total Cost per Step (s)	Total Cost to Compute Δt (s)
16x16	0.74	0.004
256x256	123.6	0.96

Measurements from Chombo 2D PPM Gas Dynamics code

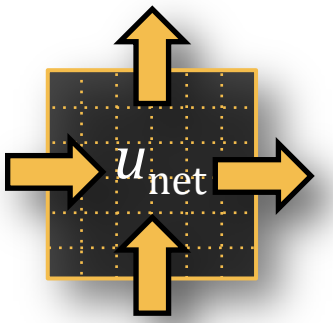
- Can afford loose criteria since cost of detection is very small
- Cost to recompute is less than 1% of a full update step

Detect and Correct is preferable

Conservation properties of the solution are local checksums that can protect the update

Discrete Conservation means (1D):

$$\underbrace{\sum_{j=1}^{N} \bar{u}_j^{n+1}}_{\text{New Net Conserved Value}} = \underbrace{\sum_{j=1}^{N} \bar{u}_j^{n}}_{\text{Old Net Conserved Value}} - \underbrace{\frac{\Delta t}{\Delta x}\left[\hat{f}_{N+1/2} - \hat{f}_{1/2}\right]}_{\text{Net fluxes into and out of perimeter}}$$



- During update, compute net fluxes
- After update, compute new net conserved value
- Compare to old net value
- On failure, redo check and/or step
- Cost of check is negligible

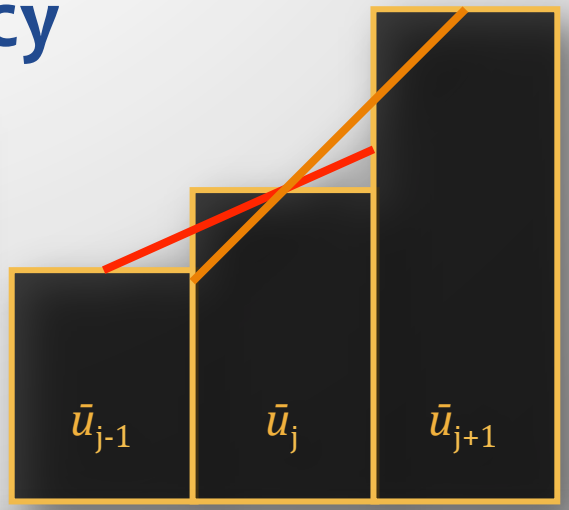
Box Length	Total Cost per Step (s)	Cost to check (s)
16x16	0.74	0.005
32x32	2.32	0.014
64x64	8.47	0.04
128x128	32.6	0.16
256x256	123.6	0.62

Measurements from Chombo 2D PPM Gas Dynamics code

NB: Conservation is insufficient to ensure convergence to the correct solution!

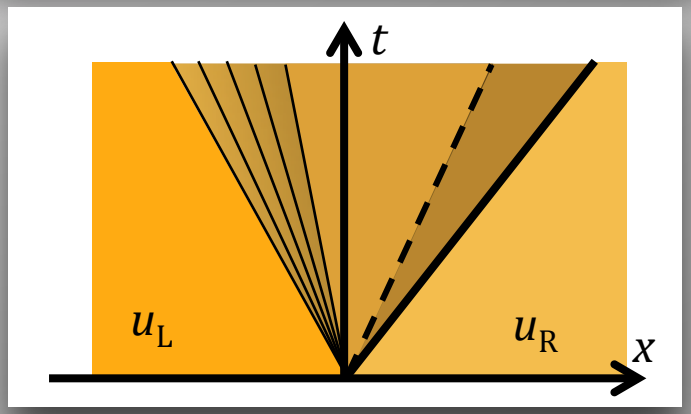
Protecting computationally intensive flux calculations ensures consistency

$$\hat{f}_{j+1/2} = g_{j+1/2}(\bar{u}^n) + \mathcal{O}(\Delta x^p) + \mathcal{O}(\Delta t^q)$$



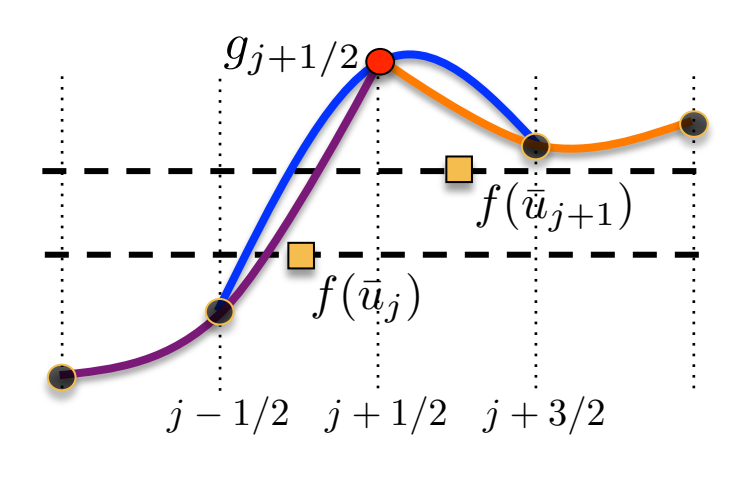
- **Limited high-order corrections using local smoothness of solution:**
  - Nonlinear solution reconstruction (MUSCL, PPM, ENO/WENO) alters stencil or falls back to first-order
  - Nonlinear blending of first- and high-order fluxes (FCT) falls back to first-order
- **Determination of the flux at each interface often involves (approximate) solution of a Riemann problem**

$$du_j = \bar{u}_j - \bar{u}_{j-1}$$
$$\widetilde{du}_j = \mathrm{minmod}\left(du_j, du_{j+1}\right)$$



We can use similar detectors to identify possibly incorrect fluxes

- Fluxes are piecewise continuous
- Use changes in local curvature to detect possible "glitches" in fluxes



"Curvature":

$$\mathrm{d2g_{i+1/2}} = \mathrm{g_{i+3/2}} - \mathrm{2g_{i+1/2}} + \mathrm{g_{i-1/2}}$$

Possible "glitch" if curvature changes sign twice over three successive points

- If flux is not between fluxes evaluated at left and right bounding states, it is an extrema or corrupted
- In this case, replace flux with low-cost first-order flux, e.g. HLLE

$$f(\bar{u}_j) \leq g_{j+1/2} \leq f(\bar{u}_{j+1})$$

This "hides" large corruptions beneath an ordered error

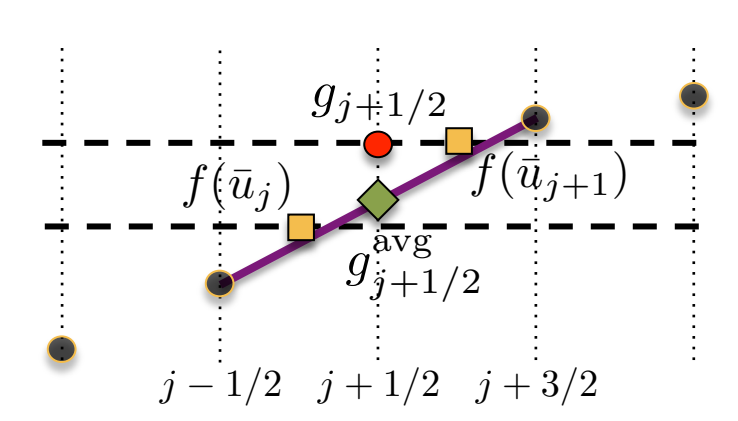
Any remaining corruptions will be bounded by at least by first-order errors

$$f(\bar{u}_j) \leq g_{j+1/2} \leq f(\bar{u}_{j+1})$$

$$\Rightarrow \quad \mathcal{O}(\Delta x) \leq g_{j+1/2} - \hat{f}_{j+1/2} \leq \mathcal{O}(\Delta x)$$

Fluxes evaluated at the bounding states are first-order approximations of the interface flux

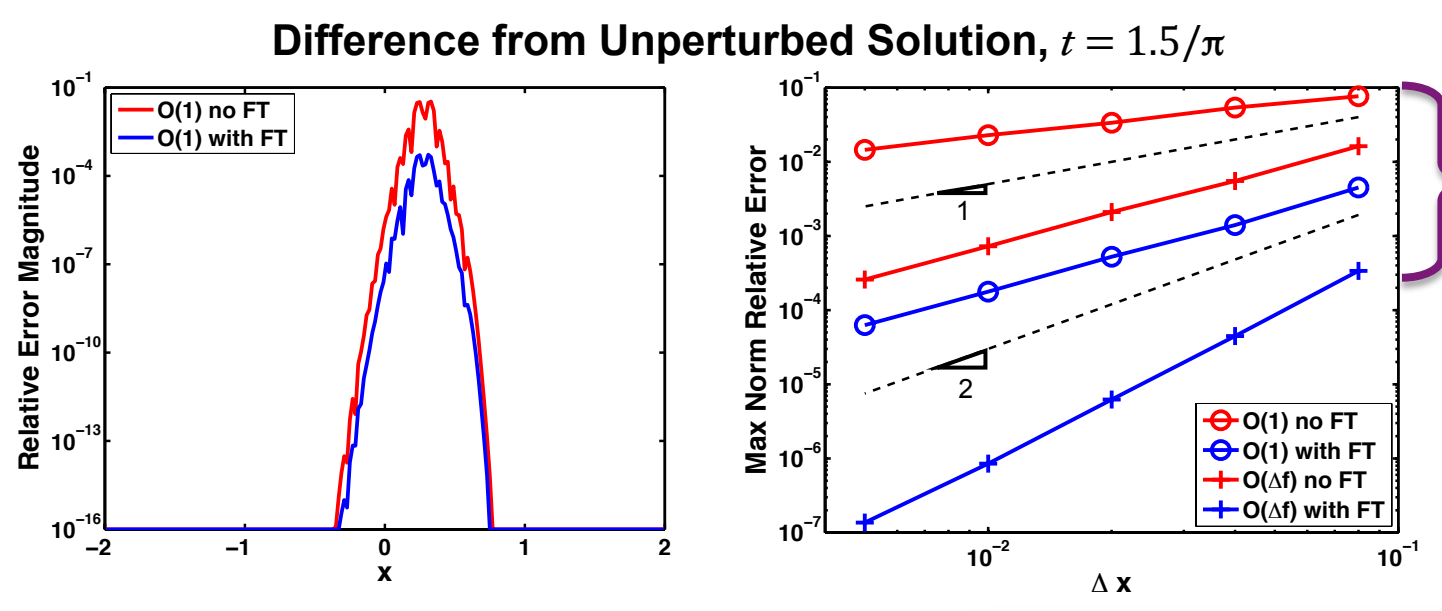
- We can do better!
- Continuity of the flux implies that the average value based on neighbors will be a second-order approximation
- Improve bounded candidate flux if it is closer to a bound than to the average
- In this case, replace with the average



$$g^{\text{avg}}_{j+1/2} = \frac{1}{2}\left[g_{j-1/2} + g_{j+3/2}\right]$$

This "hides" bounded corruptions beneath an ordered error

Preliminary results indicate the flux correction process is effective for Burgers' equation



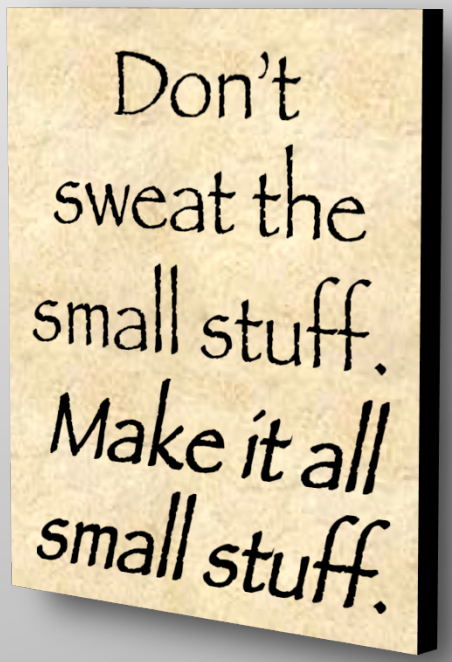
Roughly 100x decrease in max error with verification

- One error injected at $t \approx 1.1/\pi$
- Both $O(1)$ and $O(\Delta f)$ size errors
- Total verification cost will be lower for more complex flux functions
- Number of critical points is fixed, so % cost decreases with increasing N

N	Avg Flux Calc Cost (s)	Cost to verify (s)	% of Flux cost
50	1.6e-04	2.4e-04	150
100	3.0e-04	1.6e-04	53
200	5.9e-04	7.9e-05	13
400	1.1e-03	8.3e-05	7.5
800	2.2e-03	1.1e-04	5

The prospect for making algorithms for hyperbolic systems tolerant to SDCs is good

- **Complete elimination of SDCs is not necessary!**
  - Don't restart, even locally, if you don't have to
  - Sacrifice some accuracy for robustness: mask faults with controllable numerical errors
  - Let stabilizing aspects of schemes control masked SDCs



Acknowledgments

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program as part of the Resilient Extreme-Scale Solvers initiative, Karen Pao, program manager.

Image sources on slide 4:

[a] NASA, ESA, CXC, SAO, the Hubble Heritage Team (STScI/AURA), and J. Hughes (Rutgers University), http://hubblesite.org/gallery/album/entire/pr2010027c/

[b] BBC, http://newsimg.bbc.co.uk/media/images/47480000/jpg/_47480955_explosion1_copy.jpg

[c] Jeff Miller, http://www.news.wisc.edu/story_images/4351/original/Schwartz_David_lab09_1887.jpg

[d] John Gay, United States Navy ID 990707-N-6483G-001, http://upload.wikimedia.org/wikipedia/commons/d/d0/FA-18_Hornet_breaking_sound_barrier_%287_July_1999%29.jpg

[e] NASA Earth Observatory, http://eoimages.gsfc.nasa.gov/images/imagerecords/5000/5125/srilanka_kalutara_flood_dec26_2004_dg.jpg

[f] Kenny Louie, CC BY 2.0, http://www.flickr.com/photos/kwl/2743788633/

[g] M. E. Wysession and G. Caras, http://epsc.wustl.edu/seismology/michael/web/xsearth.jpg

Lawrence Livermore
National Laboratory