Mathematical
Institute

Solving parabolic PDEs in half precision

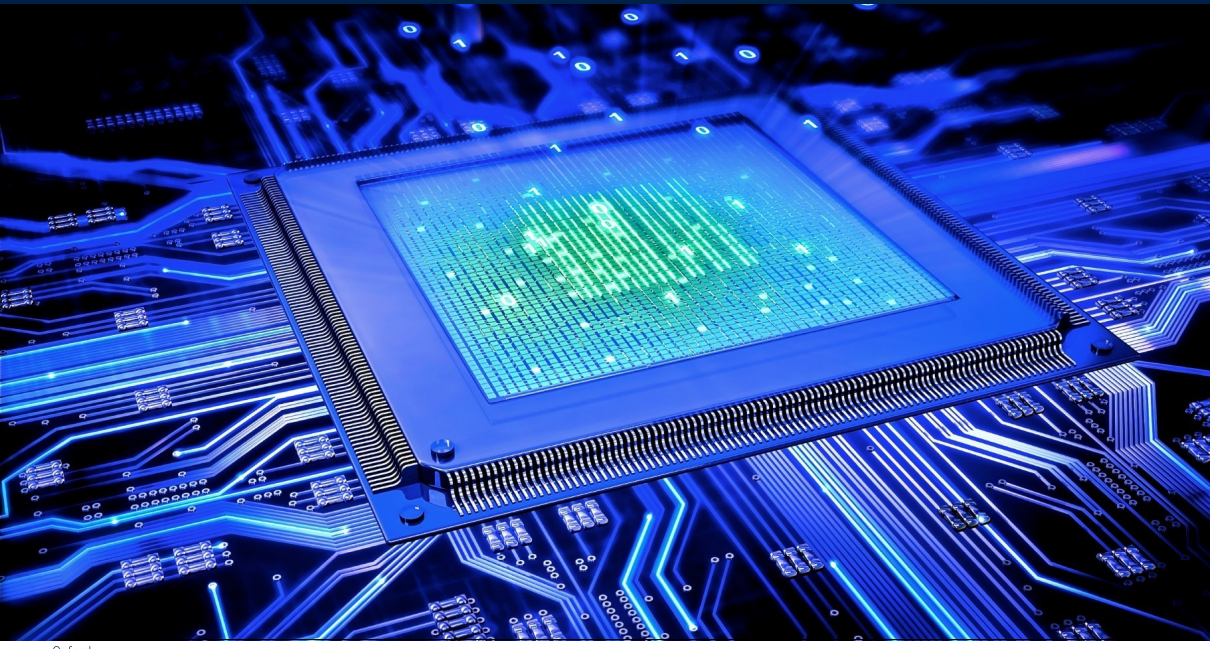
Matteo Croci & Mike Giles
Mathematical Institute
University of Oxford

NLA Group meeting, University of Manchester, 7 December 2020

Oxford
Mathematics

Objective: developing low-precision PDE solvers

UNIVERSITY OF OXFORD

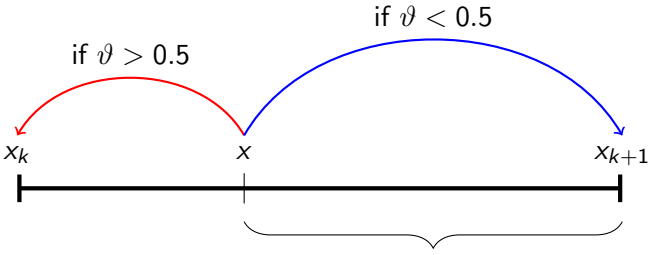


UNIVERSITY OF OXFORD

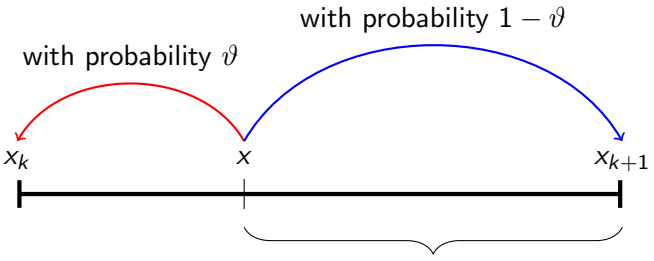
Background

A 3-step guide to solving the heat equation in low precision

Round to nearest

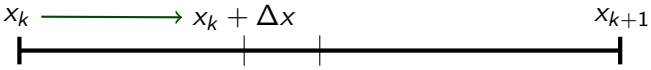


$$\mathrm{fl}(x) = x(1+\delta), \quad \text{with} \quad |\delta| \leq u.$$



$$\mathrm{sr}(x) = x(1+\delta(\omega)), \quad \text{with} \quad |\delta(\omega)| \leq 2u, \quad \text{and} \quad \mathbb{E}[sr(x)] = x.$$

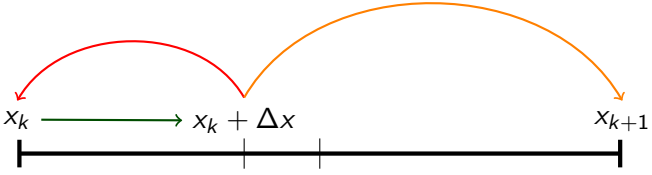
RtN might cause stagnation



RtN might cause stagnation

Example: the series $\sum_{n=1}^{\infty} \frac{1}{n}$ diverges, yet it "converges" with RtN in finite precision!

SR is resilient to stagnation



The good:

- SR roundoffs are zero mean and mean-independent: $\mathbb{E}[\delta_i|\delta_1,\ldots,\delta_{i-1}] = \mathbb{E}[\delta_i] = 0$.
- Rounding errors on n term sums accumulate like $O(\sqrt{n}u)$ rather than $O(nu)$.
- Evaluation of linear functions is exact in expectation. $O(u^2)$ bias for C^2 functions.

Properties of SR [Connolly, Higham, Mary 2020]

UNIVERSITY OF OXFORD

The good:

- SR roundoffs are zero mean and mean-independent: $\mathbb{E}[\delta_i|\delta_1,\ldots,\delta_{i-1}] = \mathbb{E}[\delta_i] = 0$.
- Rounding errors on n term sums accumulate like $O(\sqrt{n}u)$ rather than $O(nu)$.
- Evaluation of linear functions is exact in expectation. $O(u^2)$ bias for C^2 functions.

The less good:

- Worst-case error same order as RtN.
- Limited, yet growing, hardware support.

Heat equation with nonzero forcing

We consider the heat equation with non-zero forcing:

$$\begin{cases} \dot{u}(t, \boldsymbol{x}) = \nabla^2 u(t, \boldsymbol{x}) + f(t, \boldsymbol{x}), & \boldsymbol{x} \in D = [0,1]^d, \quad t > 0, \\ u(0, \boldsymbol{x}) = u_0(x), & \boldsymbol{x} \in D, \\ u(t, \boldsymbol{x}) = g(\boldsymbol{x}), & \boldsymbol{x} \in \partial D, \qquad t > 0. \end{cases}$$

We use finite differences in space and a Runge-Kutta method in time with absolute stability function $S(z)$. Discretisation parameters: Δt, h, $\lambda = \Delta t / h^2$.

We consider the heat equation with non-zero forcing:

$$\begin{cases} \dot{u}(t,\boldsymbol{x}) = \nabla^2 u(t,\boldsymbol{x}) + f(t,\boldsymbol{x}), & \boldsymbol{x} \in D = [0,1]^d, \quad t > 0, \\ u(0,\boldsymbol{x}) = u_0(x), & \boldsymbol{x} \in D, \\ u(t,\boldsymbol{x}) = g(\boldsymbol{x}), & \boldsymbol{x} \in \partial D, \qquad t > 0. \end{cases}$$

We use finite differences in space and a Runge-Kutta method in time with absolute stability function $S(z)$. Discretisation parameters: Δt, h, $\lambda = \Delta t/h^2$.

Let A be the (spd) stiffness matrix. The numerical scheme is

$$\boldsymbol{U}^{n+1} = S(-\Delta t A)\boldsymbol{U}^n + \Delta t \boldsymbol{F}^n = \boldsymbol{U}^n + \Delta \boldsymbol{U}^n,$$

e.g. $S_{\mathrm{FE}}(-\Delta t A) = (I - \Delta t A)$, $S_{\mathrm{BE}}(-\Delta t A) = (I + \Delta t A)^{-1}$.
e.g. $\Delta_{\mathrm{FE}}\boldsymbol{U}^n = -\Delta t A \boldsymbol{U}^n + \Delta t \boldsymbol{F}^n$, $\Delta_{\mathrm{BE}}\boldsymbol{U}^n = (I + \Delta t A)^{-1}(-\Delta t A \boldsymbol{U}^n + \Delta t \tilde{\boldsymbol{F}}^n)$.

We work in **bfloat16 half precision**, $u = 2^{-8} \approx 4 \times 10^{-3}$.
Everything extends to FEM and linear parabolic equations.

Background

A 3-step guide to solving the heat equation in low precision

1) Local rounding errors and the delta form

How to best implement the Runge-Kutta scheme? Use the **delta form**!

How to best implement the Runge-Kutta scheme? Use the **delta form**!
In finite precision we compute:

$$\hat{\boldsymbol{U}}^{n+1} = S(-\Delta t A)\hat{\boldsymbol{U}}^n + \Delta t \boldsymbol{F}^n + \mathcal{E}^n, \quad \text{(non-delta form)}.$$
$$\hat{\boldsymbol{U}}^{n+1} = \hat{\boldsymbol{U}}^n + \Delta\hat{\boldsymbol{U}}^n + \varepsilon^n + \Theta^n, \quad \text{(delta form)}.$$

Here $\mathcal{E}^n$, Θ^n and ε^n are the **local rounding errors**. Θ^n is the error in the computation of $\Delta\hat{\boldsymbol{U}}^n$ and ε^n is the error in its addition to $\hat{\boldsymbol{U}}^n$.

1) Local rounding errors and the delta form

How to best implement the Runge-Kutta scheme? Use the **delta form**!
In finite precision we compute:

$$\hat{\boldsymbol{U}}^{n+1} = S(-\Delta t A)\hat{\boldsymbol{U}}^n + \Delta t \boldsymbol{F}^n + \mathcal{E}^n, \quad \text{(non-delta form)}.$$
$$\hat{\boldsymbol{U}}^{n+1} = \hat{\boldsymbol{U}}^n + \Delta\hat{\boldsymbol{U}}^n + \varepsilon^n + \Theta^n, \quad \text{(delta form)}.$$

Here $\mathcal{E}^n$, Θ^n and ε^n are the **local rounding errors**. Θ^n is the error in the computation of $\Delta\hat{\boldsymbol{U}}^n$ and ε^n is the error in its addition to $\hat{\boldsymbol{U}}^n$.

Theorem [C. and Giles 2020]

Assume that there exist constants M_f, M_u, $M_A > 0$ such that $|f(t,\boldsymbol{x})| \leq M_f$, $||\hat{\boldsymbol{U}}^n||_\infty \leq M_u$, $||\Delta t A\hat{\boldsymbol{U}}^n||_\infty \leq M_A \Delta t^p$, $||\widehat{\Delta t A\hat{\boldsymbol{U}}^n} - \Delta t A\hat{\boldsymbol{U}}^n||_\infty \leq M_A\Delta t^p$ with $p \geq 0$.

Then $\exists a, b > 0$ s.t. $\forall n$, $\quad ||\mathcal{E}^n||_\infty \leq a(M_u + \Delta t M_f)u = \mathcal{E}$,
$$||\Theta^n||_\infty \leq b(M_A\Delta t^p + \Delta t M_f)u = \Theta, \quad ||\varepsilon^n||_\infty \leq uM_u = \varepsilon.$$

Note: in theory and practice we have $\Theta \ll \varepsilon < \mathcal{E}$.

How to best implement the matrix-vector product $-A\boldsymbol{U}^n$?

$$\frac{\boldsymbol{U}_{i+1}^n - 2\boldsymbol{U}_i^n + \boldsymbol{U}_{i-1}^n}{h^2}, \qquad \frac{(\boldsymbol{U}_{i+1}^n - \boldsymbol{U}_i^n) - (\boldsymbol{U}_i^n - \boldsymbol{U}_{i-1}^n)}{h^2}.$$

2) Finite differences and exact subtraction

How to best implement the matrix-vector product $-A\boldsymbol{U}^n$?

$$\frac{\boldsymbol{U}_{i+1}^n - 2\boldsymbol{U}_i^n + \boldsymbol{U}_{i-1}^n}{h^2}, \qquad \frac{(\boldsymbol{U}_{i+1}^n - \boldsymbol{U}_i^n) - (\boldsymbol{U}_i^n - \boldsymbol{U}_{i-1}^n)}{h^2}.$$

Leads to $O(h^{-2})$ error! **Leads to near-exact matvecs.**

A similar trick works for FEM as well, but requires matrix-free matvecs.

How to best implement the matrix-vector product $-A\boldsymbol{U}^n$?

$$\frac{\boldsymbol{U}^n_{i+1} - 2\boldsymbol{U}^n_i + \boldsymbol{U}^n_{i-1}}{h^2}, \qquad \frac{(\boldsymbol{U}^n_{i+1} - \boldsymbol{U}^n_i) - (\boldsymbol{U}^n_i - \boldsymbol{U}^n_{i-1})}{h^2}.$$

Leads to $O(h^{-2})$ error! **Leads to near-exact matvecs.**

A similar trick works for FEM as well, but requires matrix-free matvecs.

Parts of a Theorem [C. and Giles 2020]

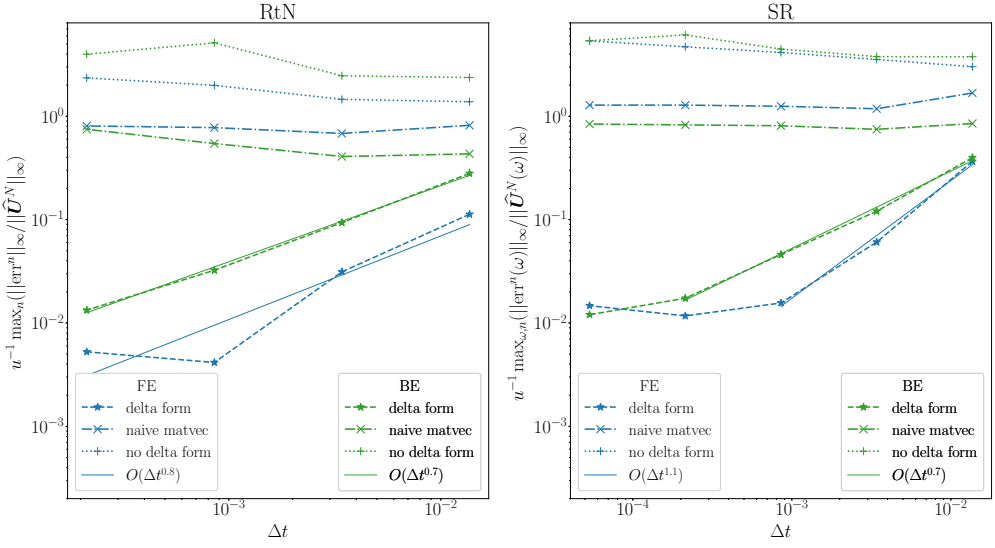
If $a, b, c \in \mathbb{R}$ are exactly represented in floating point arithmetic, and

$$\max(|a-b|, |b-c|) \leq \frac{1}{2}\min(|a|, |b|, |c|)$$

then $(a-b)-(b-c)$ is computed exactly.

See also Section 2.5 in "Accuracy and Stability of Numerical Algorithms" by Nick Higham.

Worst-case local rounding errors in 2D ($\mathcal{E}$ and Θ)



Note: from now on we use the delta form with "smart" matvecs.

Why is RtN in low precision bad for parabolic equations?

a) Stagnation:

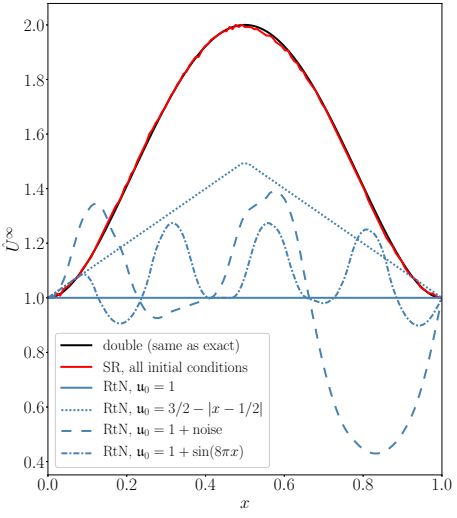
- RtN always stagnates for sufficiently small Δt.
- The RtN solution is initial condition, discretization and precision dependent.

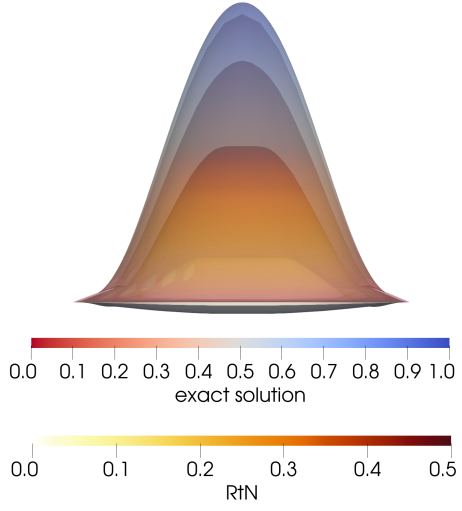
b) Global error:

- RtN rounding errors are strongly correlated and cannot be modelled as zero-mean independent random variables.
- RtN global errors grow like $O(u\Delta t^{-1})$ until stagnation.

SR fixes all these issues!

3a) Stagnation (left 1D, right 2D)





RtN computations are discretization and initial condition dependent. SR works!

3b) Global rounding errors [C. and Giles 2020]

Define the global rounding error $\boldsymbol{E}^n = \hat{\boldsymbol{U}}^n - \boldsymbol{U}^n$. Since Θ^n is negligible we have

$$\boldsymbol{E}^{n+1} = S(-\Delta t A)\boldsymbol{E}^n + \varepsilon^n.$$

Compare this with traditional convergence/rounding error results for ODE solvers where ε^n is a $O(\Delta t^2)$ term [Henrici 1962-1963, Arató 1983].

We can distinguish two cases:

RtN: we can only assume the worst-case scenario, $|\varepsilon_i^n| \leq \varepsilon$ for all n, i.

SR: the ε_i^n are zero-mean spatially independent and temporally mean-independent.

3b) Global rounding errors [C. and Giles 2020]

Define the global rounding error $\boldsymbol{E}^n = \hat{\boldsymbol{U}}^n - \boldsymbol{U}^n$. Since Θ^n is negligible we have

$$\boldsymbol{E}^{n+1} = S(-\Delta t A)\boldsymbol{E}^n + \varepsilon^n.$$

Compare this with traditional convergence/rounding error results for ODE solvers where ε^n is a $O(\Delta t^2)$ term [Henrici 1962-1963, Arató 1983].

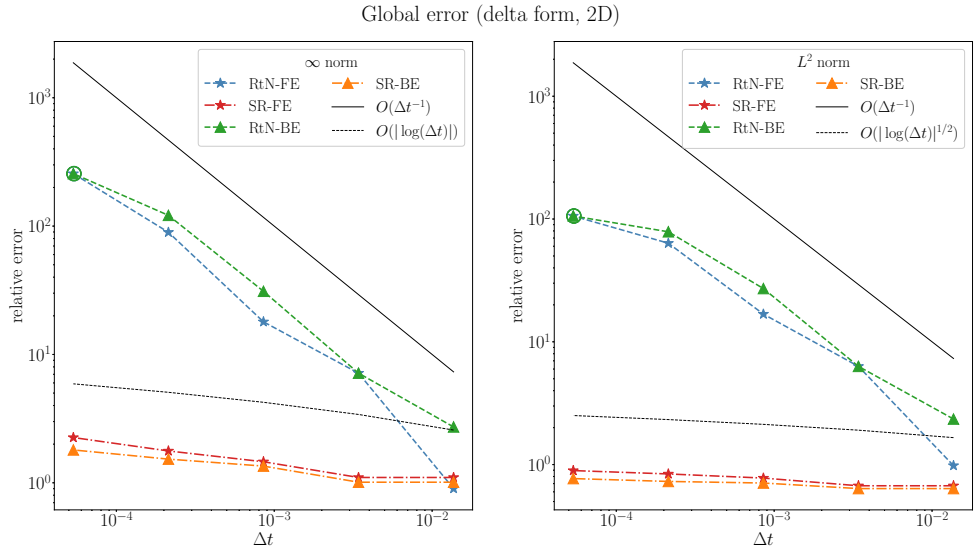
We can distinguish two cases:

RtN: we can only assume the worst-case scenario, $|\varepsilon_i^n| \leq \varepsilon$ for all n, i.

SR: the ε_i^n are zero-mean spatially independent and temporally mean-independent.

Mode	Norm	1D	2D	3D
RtN	L^2, ∞	$O(\varepsilon \Delta t^{-1})$	$O(\varepsilon \Delta t^{-1})$	$O(\varepsilon \Delta t^{-1})$
SR	$\mathbb{E}[\|\| \cdot \|\|_\infty]$	$O(\varepsilon \Delta t^{-1/4} \ell(\Delta t)^{1/2})$	$O(\varepsilon \ell(\Delta t))$	$O(\varepsilon \ell(\Delta t)^{1/2})$
SR	$\mathbb{E}[\|\| \cdot \|\|_{L^2}^2]^{1/2}$	$O(\varepsilon \Delta t^{-1/4})$	$O(\varepsilon \ell(\Delta t)^{1/2})$	$O(\varepsilon)$

Asymptotic global rounding error blow-up rates; $\ell(\Delta t) = |\log(\lambda^{-1}\Delta t)|$.



Note: relative error $=$ error $\times$ $(u||\boldsymbol{U}^N||)^{-1}$

Outlook

- Working in low precision can bring large speed, memory and energy consumption improvements. New hardware supports low-precision.
- SR might be an effective way of obtaining accurate results in much lower precision when solving time-dependent parabolic PDEs.
- Custom-built C++ low-precision emulator (`bitbucket.org/croci/libchopping/`) inspired by [Higham and Pranesh 2019]. Any chance to make CPFloat object-oriented (C++/Python wrapper)?

Current/future research directions

- Hyperbolic PDEs and stabilised explicit RK methods.
- Nested multilevel Monte Carlo methods and Optimization algorithms.
- Weather forecasting and brain simulation applications.

Thank you for listening!

UNIVERSITY OF OXFORD

Preprint, slides, and more info at: `https://croci.github.io`

[1] M. Croci and M. B. Giles. Effects of round-to-nearest and stochastic rounding in the numerical solution of the heat equation in low precision, 2020. URL `http://arxiv.org/abs/2010.16225`.

[2] M. P. Connolly, N. J. Higham, and T. Mary. Stochastic Rounding and its Probabilistic Backward Error Analysis, 2020. URL `https://hal.archives-ouvertes.fr/hal-02556997/document`.

[3] M. Fasi and M. Mikaitis. Algorithms for stochastically rounded elementary arithmetic operations in IEEE 754 floating-point arithmetic, 2020. URL `http://eprints.maths.manchester.ac.uk/2758/1/fami20.pdf`.

[4] N. J. Higham and T. Mary. A new approach to probabilistic rounding error analysis. *SIAM Journal of Scientific Computing*, 41(5):2815–2835, 2019. doi: $10.1137/18\mathrm{M}1226312$.

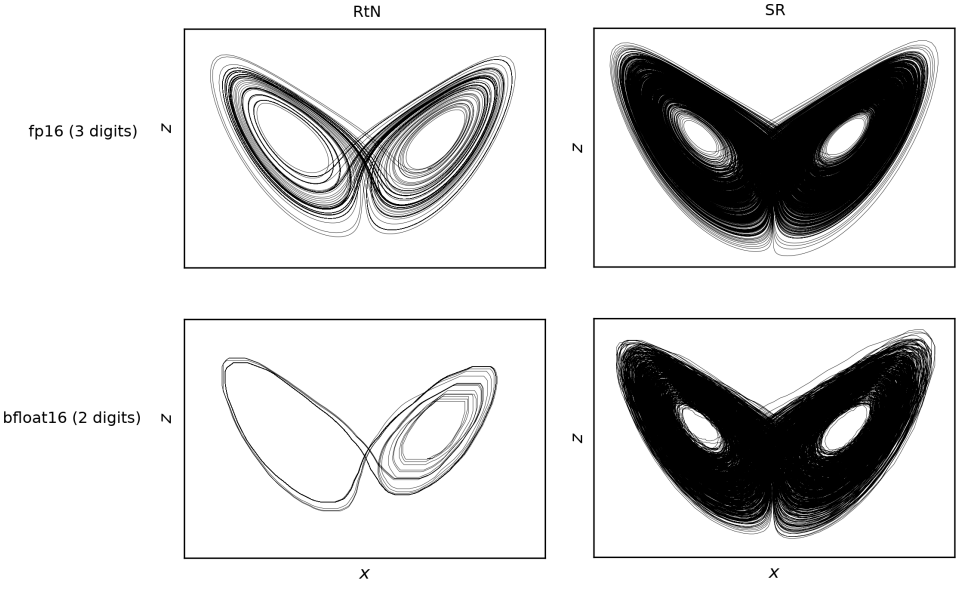
[5] N. J. Higham and S. Pranesh. Simulating low precision floating-point arithmetic. *SIAM Journal on Scientific Computing*, 41(5):C585–C602, 2019. doi: $10.1137/19\mathrm{M}1251308$.

[6] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

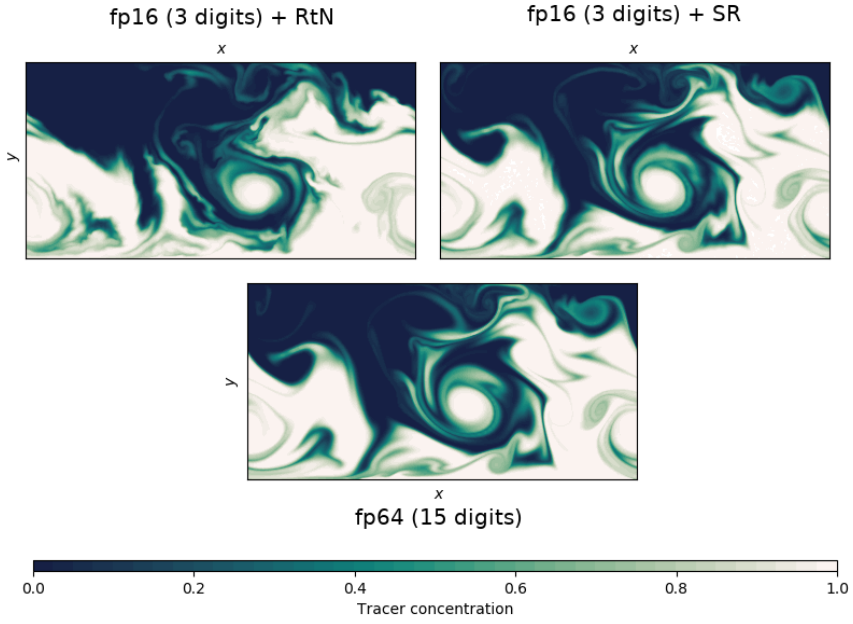
[7] M. Arató. Round-off error propagation in the integration of ordinary differential equations by one step methods. *Acta Scientiarium Mathematicarum*, 45:23–31, 1983. doi: $10.13140/2.1.3920.9608$.

[8] P. Henrici. *Discrete Variable Methods in Ordinary Differential Equations*. John Wiley & Sons, Inc., 1962.

Promising results (by Milan Kloewer in Oxford Physics)



Prominsing results (by Milan Kloewer in Oxford Physics)



Stagnation $\mathrm{fl}(x+\epsilon)=x$ occurs whenever $\frac{u}{2}|x| \geq |\epsilon|$. For the PDE:

$$\frac{u}{2}|\,\mathrm{u}(t_n, \boldsymbol{x_i})| \approx \frac{u}{2}|\hat{\boldsymbol{U}}_{\boldsymbol{i}}^n| \geq |\Delta \hat{\boldsymbol{U}}_{\boldsymbol{i}}^n| = |\hat{\boldsymbol{U}}_{\boldsymbol{i}}^{n+1} - \hat{\boldsymbol{U}}_{\boldsymbol{i}}^n| \approx \Delta t|\dot{\mathrm{u}}(t_n, \boldsymbol{x_i})|,$$

This shows that $\hat{\boldsymbol{U}}_{\boldsymbol{i}}^n$ will not be updated whenever

$$|\,\mathrm{u}(t_n, \boldsymbol{x_i})| \gtrapprox 2(\Delta t/u)|\dot{\mathrm{u}}(t_n, \boldsymbol{x_i})|.$$

More formally,

Lemma [C. and Giles 2020]

Assume that the delta form is used and that $p > 0$. If there exists $\epsilon > 0$ such that $|\hat{\boldsymbol{U}}_{\boldsymbol{i}}^{\bar{n}}| \geq \epsilon$ for some $\boldsymbol{i}$, $\bar{n}$, then there exists $\tau(\epsilon) > 0$ such that if $\Delta t < \tau$, we have $\hat{\boldsymbol{U}}_{\boldsymbol{i}}^n = \hat{\boldsymbol{U}}_{\boldsymbol{i}}^{\bar{n}}$ for all $n \geq \bar{n}$.