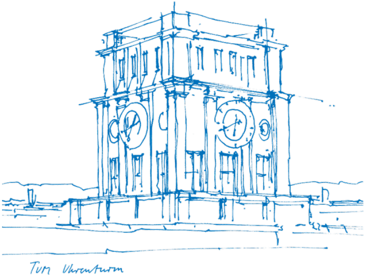
Mixed-Precision in High-Order Methods: the Impact of Numerical Precision on the ADER-DG Algorithm

Modern PDE Discretization Methods and Solvers in a Non-Smooth World

Marc Marot-Lassauzaie & Michael Bader
Technical University of Munich

June 5th, 2024



Why Mixed-Precision?

Speedup may result from:

- higher effective vectorization
- reduced bandwidth
- holding data in lower caches

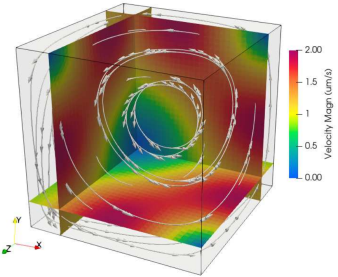
Name	Common name	Significand bits	Exponents bits	Maximal exponent
bfloat 16	bf16	7	8	127
IEEE binary 16	half precision	10	5	15
IEEE binary 32	single precision	23	8	127
IEEE binary 64	double precision	52	11	1023
IEEE binary 128	quadruple precision	112	15	16383

The respective distribution of bits in different signed floating-point formats

Why Mixed-Precision?

D. Demidov et. al:

- Simulation of Stokes problem using linear solver
- Speedup factor of 4
- Memory footprint reduced by 50%



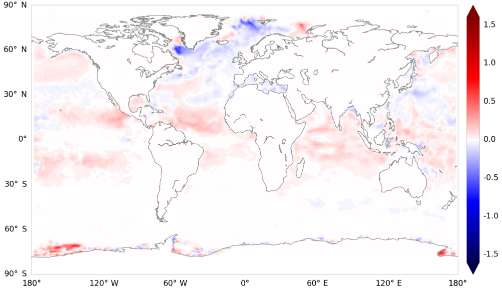
Stokes velocity field of the unit cube problem as depicted in [1]

[1] D. Demidov et. al. Accelerating linear solvers for Stokes problems with C++ metaprogramming. J. Comput. Sc. 49 (2021)

Why Mixed-Precision?

Prims, Acosta et. al:

- Simulation of ocean models using a reduced-precision emulator
- "*an improper reduction can lead to accuracy losses that may make the results unreliable*"



Monthly mean difference in sea surface temperature (single precision minus reference) as depicted in [2]

[2]Prims et. al. How to use mixed precision in ocean models: exploring a potential reduction of numerical precision in NEMO 4.0 and ROMS 3.6. Geosci. Model Dev., 12, 3135–3148, 2019. https://doi.org/10.5194/gmd-12-3135-2019

“Mixed and Variable Precision for an Exascale Hyperbolic PDE Engine”

Project funded by the German Research Foundation (project No. 462423388).

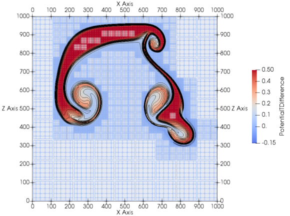
Agenda for this talk:

- Main focus will on stability and convergence, not speedup
- Overall impact of precision
- Effect of mixed-precision approaches
- Variable-precision

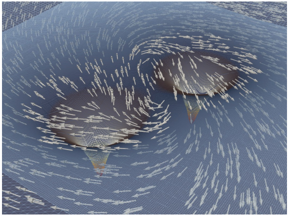
ExaHyPE – an “Exascale PDE Engine”

Goal: a PDE “engine” (as in “game engine”) $\rightsquigarrow$ Reinarz et al.[3]

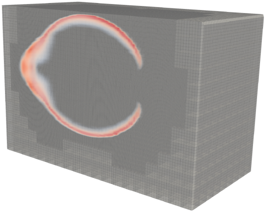
- Fixed numerics and mesh infrastructure, but *stay flexible w.r.t. PDE* (focus on hyperbolic conservation laws and high-order DG)
- Load balancing and adaptive mesh refinement via **Peano4**



atmospheric flows



relativistic astrophysics (Durham University)

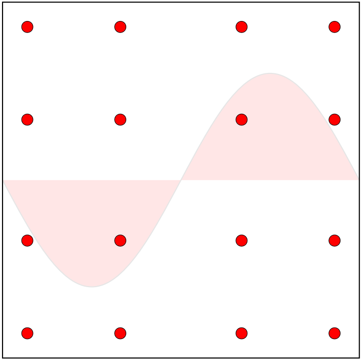


earthquake dynamic rupture

[3]Reinarz et. al., ExaHyPE: An engine for parallel dynamically adaptive simulations of wave problems. Comp. Phys. Comm. 254, 2020. https://doi.org/10.1016/j.cpc.2020.107251

ADER-DG

- High-order hyperbolic PDE solver
- Discontinuous Galerkin with ADER time stepping
- Piecewise polynomials within cells
- One data exchange per timestep
- Predictor-Corrector scheme

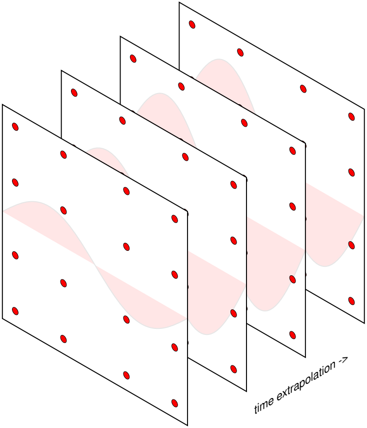


$$\int \partial_t U * \phi \, dx = \int F * \nabla \cdot \phi \, dx - \oint (F * \phi) \cdot \vec{n} \, ds \quad (1)$$

Predictor

- Expansion of cell-local polynomial in time
- Projection of the expansion to cell faces
- Integration of expansion for local update

This corresponds to a **volume integral** of the problem over the cell

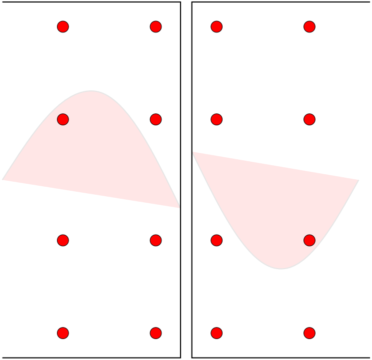


$$\int \partial_t U * \phi \, dx = \underbrace{\int F * \nabla \cdot \phi \, dx}_{\text{Predictor}} - \oint (F * \phi) \cdot \vec{n} \, ds \qquad (2)$$

Corrector

- Solving of Riemann problem at the cell faces
- Integration of Riemann fluxes over all faces

This corresponds to a **surface integral** of the problem over the cell boundary



$$\int \partial_t U * \phi \, dx = \int F * \nabla \cdot \phi \, dx - \underbrace{\oint (F * \phi) \cdot \vec{n} \, ds}_{\text{Corrector}} \tag{3}$$

Implementation

- Templating of all utilized kernels
- Persistent storage
- Precomputed matrices for DG
- User-defined functions in various precisions

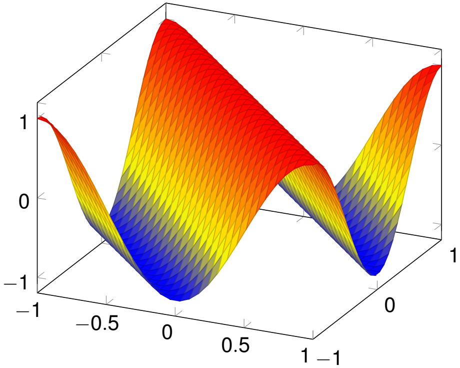
```
Template_kernelgenerator_linear_no_ps = """
//linear, no pointSources
generated::kernels::AderDG::spaceTimePredictor<
  {{SOLUTION_STORAGE_PRECISION}},
  {{PREDICTOR_COMPUTATION_PRECISIONS}},
  {{CORRECTOR_COMPUTATION_PRECISION}}>(

  repositories::{{SOLVER_INSTANCE}},
  lduh, lQhbnd, lFhbnd,
  lQi, lFi, lSi,
  lQhi, lFhi, lShi,
  gradQ, nullptr, nullptr,
  luh,
  marker.x(), marker.h(), t, dt,
  nullptr
);
"""
```

Research questions

1. How does numerical precision affect the underlying polynomial representation?
   - Initial conditions
   - Static scenarios
2. How does numerical precision affect the convergence of the method?
3. Can we make effective use of mixed-precision?
4. How about variable precision?

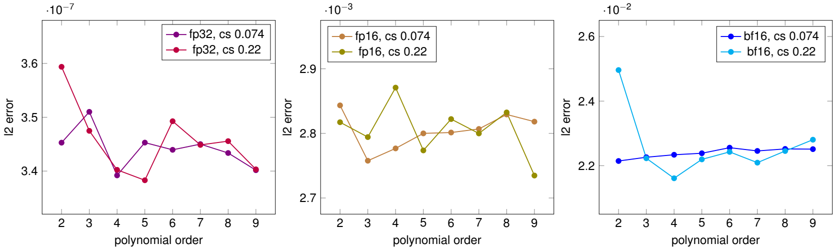
Elastic Planar Waves

- Linear elastic wave equation
- Sinusoidal starting conditions propagate through the domain without deformation
- We simulate two traversals of the entire domain
- Orders 2 through 9, cell sizes 0.074 or 0.22



The planar-wave initial condition used for verification of both the acoustic and elastic equations. $cos(-\pi * (x+y))$

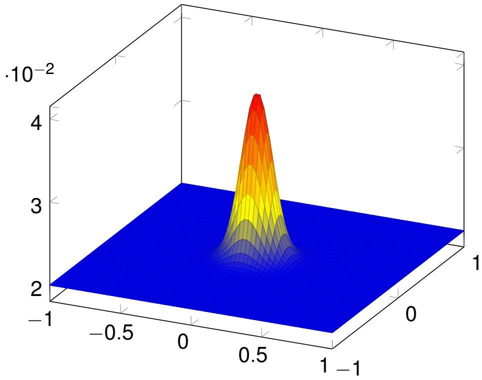
Elastic Planar Waves



- Errors stay roughly the same independently of cell size or order
- 16-bit error larger than 32-bit error

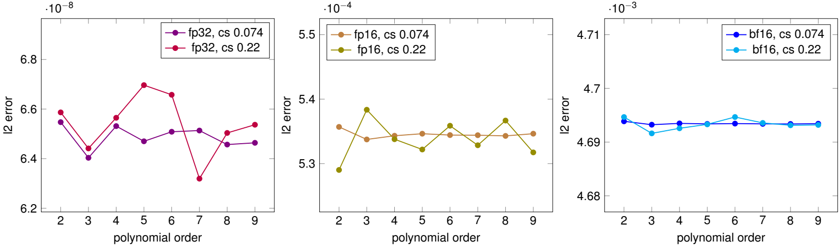
Euler Gaussian Bell

- Euler equations: fluid dynamics neglecting viscosity and heat
- Initial Gaussian in the density propagates without deformation
- We simulate two full grid traversals
- Orders 2 through 9, cell sizes 0.074 or 0.22



The gaussian bell initial condition used for verification of the Euler equations.
$0.02 * (1 + e^{-50*(x^2+y^2)})$

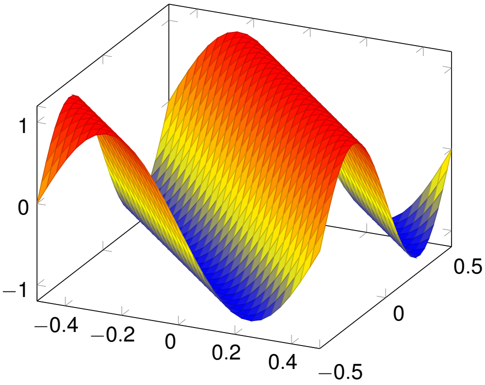
Euler Gaussian Bell



- Errors still roughly constant
- All errors about 1 magnitude lower than planar-waves

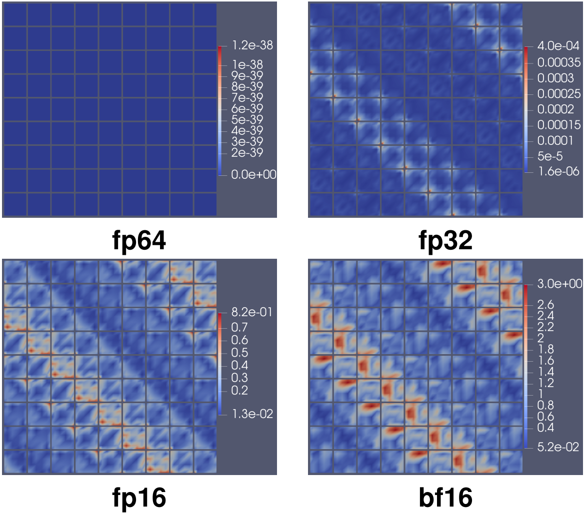
SWE Resting Lake

- Shallow water equations simulate movement of a shallow fluid
- Constant water height over sinusoidal bathymetry
- evaluate whether numerical treatment is "well balanced"



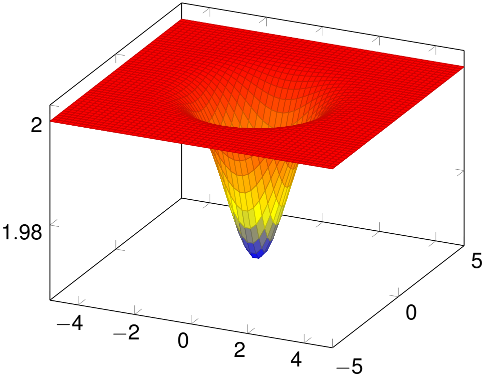
The sinusoidal initial condition used for verification of the shallow water equations.
$sin(2 * \pi * (x + y))$

SWE Resting Lake



Euler Isentropic Vortex

- Euler equations
- Rotation around center of the domain
- Each point in the vortex transmits as much fluid as it receives
- High-order methods should not contribute numerical dissipation

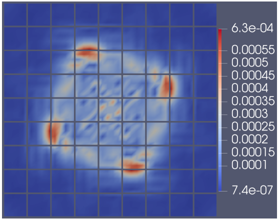


The density vortex used for verification of the Euler equations.

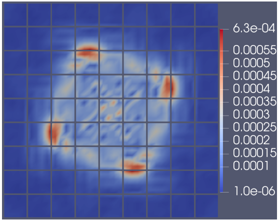
---

[3] Hu, C., Shu, C.W.: Weighted essentially non-oscillatory schemes on triangular meshes. J. Comput. Phys. 150, 97-127 (1999)

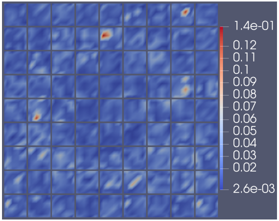
Euler Isentropic Vortex



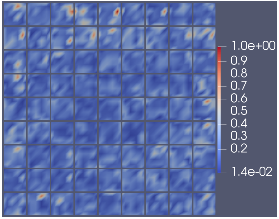
fp64



fp32



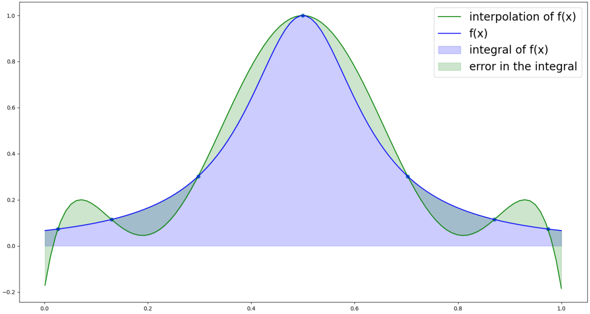
fp16



bf16

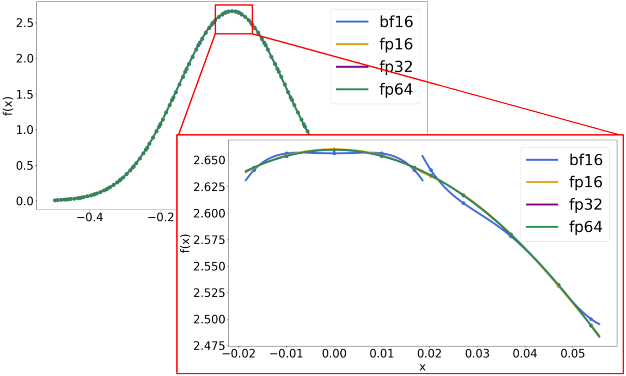
Lagrange Polynomials and Discontinuities

Runge-phenomenon: discontinuities cause oscillations



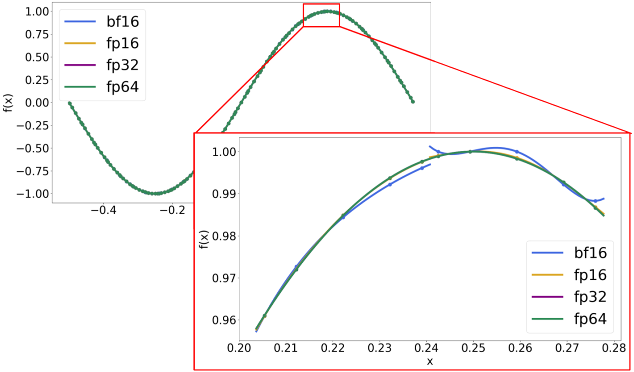
Interpolation of a non-polynomial function using Gauss-Legendre nodes. This exhibits Runge-oscillation.

Lagrange Polynomials and Numerical Precision



Detail of the Lagrange polynomial representation of a Gaussian function at Gauss-Legendre nodes with the values at the nodes computed in different precisions

Lagrange Polynomials and Numerical Precision



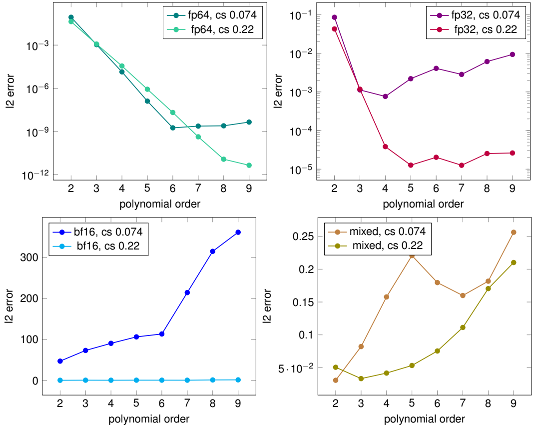
Detail of the Lagrange polynomial representation of a sinus function at Gauss-Legendre nodes with the values at the nodes computed in different precisions

How does Numerical Precision affect Convergence?

- Constant precision over entire algorithm
- Numerical error as a function of cell size and polynomial order

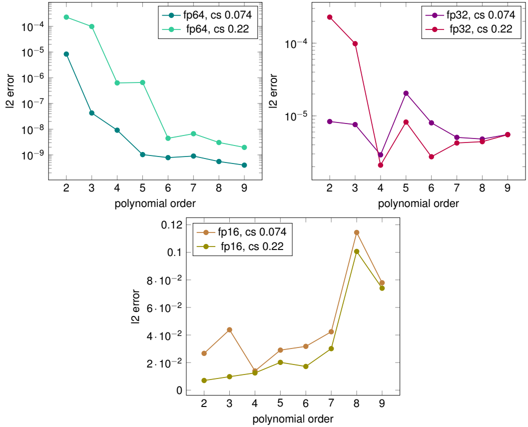
Elastic Planar Waves

- fp64 and fp32 converge
- fp16 breaks, but mixed-precision can restore stability
- Neither bf16 nor fp16 converge, both produce large errors



Euler Gaussian Bell

- fp64 still converges
- fp32 stays nearly constant
- fp16 remains stable but does not converge
- bf16 cannot resolve the equation



Mixed-Precision

- Order 5, 27x27 cells
- Default fp64 precision
- One of storage, predictor, corrector or picard iterations in lower precision

Mixed-Precision

	Elastic			Euler			
prec	predictor	corrector	storage	predictor	corrector	storage	Picard
bf16	$4.50e^{-1}$	$1.84e^{-1}$	$7.58e^{-1}$	$1.42e^{-1}$	NAN	NAN	$9.34e^{-2}$
fp16	NAN	$1.35e^{-2}$	$2.12e^{-1}$	$3.20e^{-2}$	$2.08e^{-2}$	$2.12e^{-2}$	$2.22e^{-2}$
fp32	$1.58e^{-5}$	$1.61e^{-6}$	$1.54e^{-5}$	$2.65e^{-6}$	$1.62e^{-6}$	$2.48e^{-6}$	$2.50e^{-5}$
fp64	$1.66e^{-14}$			$6.56e^{-7}$			

- bf16 least accurate, followed by fp16, then fp32
- However in the linear equations, bf16 is more stable than fp16
- Predictor and storage have largest overall impact

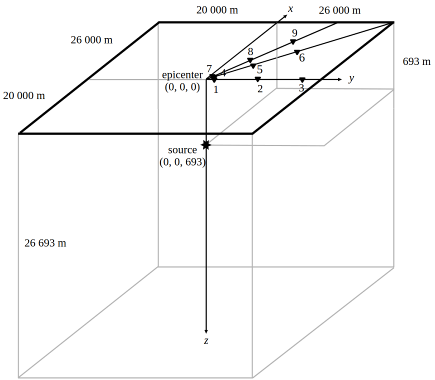
Mixed-Precision

	SWE resting lake				Euler isentropic vortex			
prec	predictor	corrector	storage	Picard	predictor	corrector	storage	Picard
bf16	$2.37e^{-01}$	NAN	$4.49e^{-01}$	NAN	NAN	NAN	NAN	$1.55e^{-01}$
fp16	NAN	NAN	$5.53e^{-02}$	NAN	NAN	$1.84e^{-01}$	$4.55e^{-01}$	$2.92e^{-02}$
fp32	$1.23e^{-04}$	$5.78e^{-05}$	$3.56e^{-06}$	$3.55e^{-05}$	$4.30e^{-05}$	$2.46e^{-05}$	$8.19e^{-05}$	$1.03e^{-05}$
fp64	$7.44e^{-12}$				$6.86e^{-06}$			

- Both bf16 and fp16 are broadly unstable
- Picard iterations have lowest impact
- Mixed-precision results are better than low-precision, but worse than high-precision

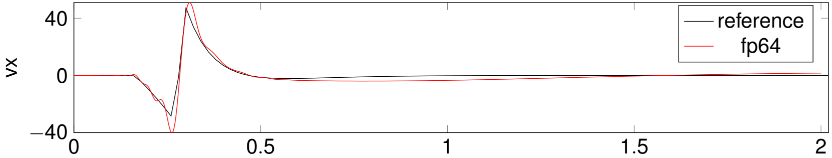
Variable Precision and the HHS1 Benchmark

- Elastic-wave propagation
- Singular point source in an infinite domain
- Free surface at the top
- Here order 5, cell size 0.11

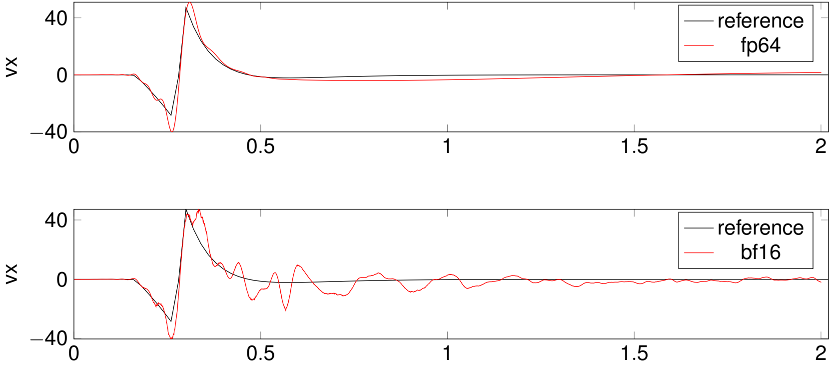


Geometry of the HHS1 problem as defined in the SISMOWINE collection (http://www.sismowine.org/)

In fp64

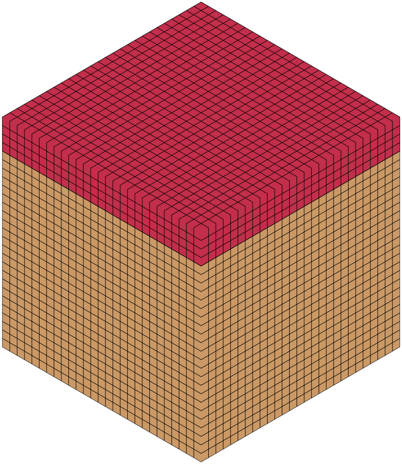


In bf16

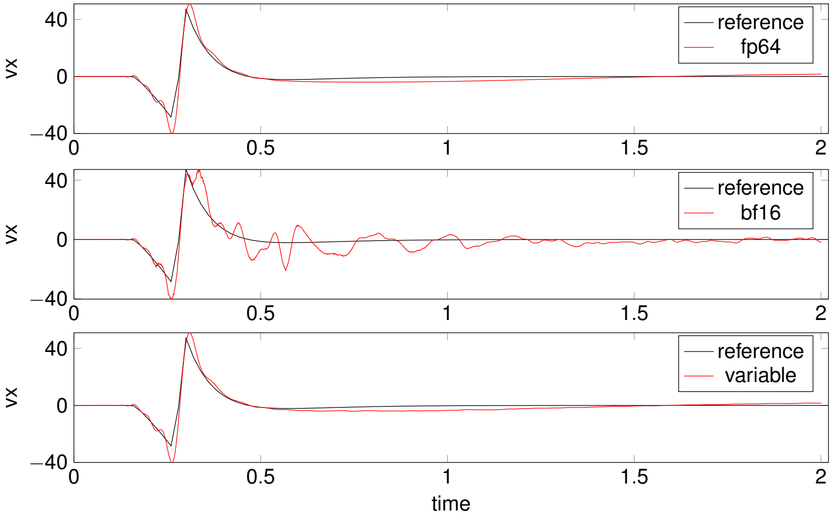

40
0
−40
vx
0
0.5
1
1.5
2
reference
fp64
40
0
−40
vx
0
0.5
1
1.5
2
reference
bf16

HHS1

- 4 layers of fp64 cells on top
- 23 layers of bf16 cells below
- about 15% of domain in fp64



In Variable Precision



Profiling

- Running code in fp32 reduces runtime by about 25% as compared to fp64
- Main improvement from fewer memory-bound pipelines, about 50% reduction
- Required memory for cell-data for HHS1 shrinks from about 408MB to 204MB

In Conclusion

- **Numerical precision matters**
- Unstable scenarios require higher precisions, but benefit less from increases to precision
- In stable scenarios, particularly for lower polynomial orders, lower precisions are sufficient
- Mixed precision approaches can't always replace high-precision, but help with stability and improve results as compared to purely low-precision

Thank you for your attention