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

M. Marot-Lassauzaie & M. Bader (Technical University of Munich)

Summary

We present a study on the effect of numerical precision on the high-order Discontinuous Galerkin method with ADER time stepping (ADER-DG) for solving hyperbolic partial differential equations.

The effects of numerical precision on the convergence of the algorithm, on its stability, and on key kernels of the method are evaluated.
Then, both mixed and variable precision approaches are tested to assess whether these can help restore high-order convergence and stability.

We find that while numerical precision has an effect on the computed solution and can prevent convergence in some cases, low precisions suffice to produce accurate results in stable scenarios. In addition, both mixed and variable precision can reduce the impact of lower precisions.

ExaHyPE2 and ADER-DG

ExaHyPE2 is an engine for solving systems of first-order partial differential equations, it relies on Peano4[Wei19] for the discretization and traversal of dynamically adaptive meshes. It provides several algorithms for solving PDEs, one of which is the ADER-DG method[Dum08], this combines high-order representations of the solution akin to the finite-element method with the cell-locality of finite volume methods.

The ADER-DG method uses a nodal discontinuous Galerkin representation, holding values at quadrature nodes with which cell-local interpolations of the solution are constructed.

It comprises two key steps:

- the **predictor** consists of a cell-local space-time expansion of the solution, which is then projected to cell faces. The expansion uses Picard- iterations for nonlinear equations, or the Euler method for linear equations. This corresponds to a volume integral over a cell.
- the **corrector** then uses the projected values on the faces to solve a Riemann problem and integrates the flux computed by this Riemann problem to update the cell-local solutions. This corresponds to a surface integral over the faces of a cell.

for cell C in K **do**
  $(q_t, f(q_t)) \leftarrow predictor(q_C)$
  $(\partial q_C, \partial f_C) \leftarrow expansion(q_t, f(q_t))$
  $q_C += volume\_integral(q_t, f(q_t))$
end for
$\Delta t_{next} \leftarrow 0$
for cell C in K **do**
  for face F in ∂K **do**
    $flux \leftarrow riemann(q_{F,left}, q_{F,right}, f_{F,left}, f_{F,right})$
    $q_C += face\_integral(flux)$
  end for
  $\Delta t_{next} \leftarrow max(\Delta t_{next}, compute\_timestep(q_C))$
end for

Alg. 1: The steps of the ADER-DG method

Name	Significand bits	Exponents bits	Max. exponent
bfloat 16	7	8	127
IEEE binary 16	10	5	15
IEEE binary 32	23	8	127
IEEE binary 64	52	11	1023

Table 1: Precisions defined by the IEEE 754 standard of Floating-Point Arithmetic[Ieee19]

Convergence

Comparison of the convergence behavior of three different scenarios with known analytical solutions for different mesh depths, polynomial orders and numerical precisions.

- **Acoustic equations: Planar Waves**

  An initial sinusoidal wave traverses the domain diagonally. We simulate two full domain traversals in a periodic domain and return to the initial conditions.

  fp64 and fp32 converge, though fp32 plateaus earlier.

  bf16 causes large errors and does not converge.

  fp16 fails but a mixed-precision approach can resolve the problem, though it also fails to converge.

- **Elastic equations: Planar Waves**

  Analogous to acoustic scenario, though more numerically complex due to additional terms.

  Similar results, though even more pronounced than the previous one.

- **Euler equations: Advection of a Smooth Density Bell[Mat20]**

  Initial smooth Gaussian in the density which is transported through the domain without deforming.

  We simulate again two full grid traversals.

  Here fp64 and fp32 behave similarly, fp16 produces correct results at low polynomial order but does not converge, bf16 is incapable of resolving the scenario whatsoever.

- **Results:**

  While numerical precision is important for high-order solutions, it can be used at lower orders.

  In addition, nonlinear equations require higher precision for stability but benefit less from increasing it.

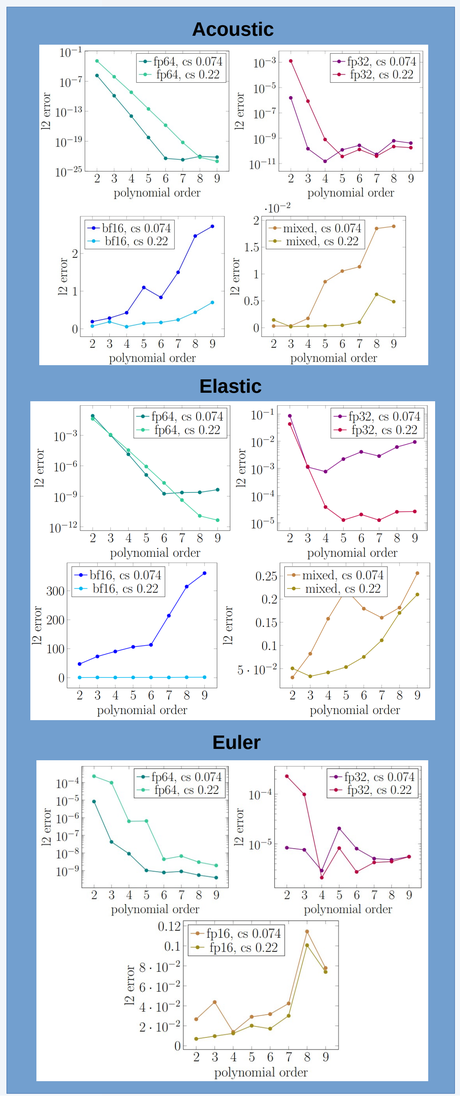

Fig. 1: Integrated L2 errors over the entire domain for three different scenarios computed with different polynomial orders, cell sizes and numerical precisions. Note that the axes differ and, in the first two scenarios, fp16-results are computed in mixed precision using a higher precision for the predictor step.

Stationary problems

Two stationary, but numerically challeging scenarios. These help measure the stability of the algorithm in different precisions, and whether these cause unphysical oscillations.

- **Shallow Water equations: Resting Lake**

  Constant water height over sinusoidal bathimetry, periodic boundaries
  Polynomial order 5, here 9x9 cells, end time at t=2.0

  Results: in all but fp64, errors form along the crest of the sinus, which indicates instabilities related to improper resolution of the geometry.

- **Euler equations: Isentropic Vortex [Shu99]**

  Stationary rotation around center of domain, periodic boundaries
  Polynomial order 5, here 9x9 cells, end time at t=10.0

  Results: fp64 and fp32 form essentially identical errors around the center of rotation, indicating slight errors in the geometry.

  fp16 and bf16 show large errors over the entire domain, indicating that the algorithm fails to resolve the equation irrespective of the local geometry.

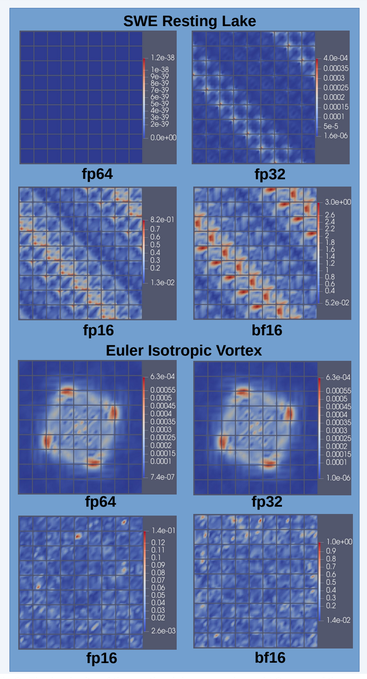

Fig 3: Final velocity of the resting lake scenario and L2-error of the isentropic vortex problem when computed in different precisions

Lagrange interpolations and rounding errors

Lagrange interpolations are susceptible to the Runge phenomenon, which causes oscillations in the interpolation of discontinuous functions.

Fig. 4 shows rounding errors from low precision triggering these phenomena.

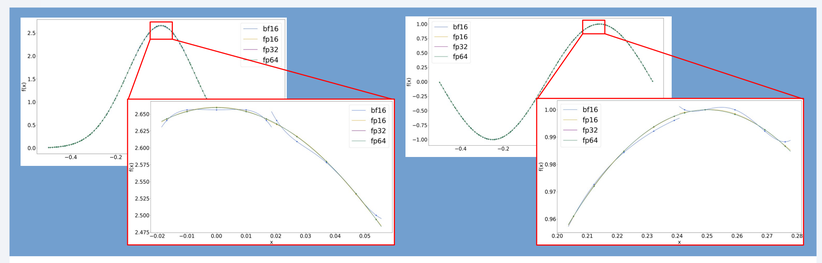

Fig. 4: 5th-Order Lagrange interpolation of Gaussian and sinusoidal functions with support points computed in different precisions. Rounding errors from lower numerical precisions lead to Gibbs-oscillation and misaligned projections at the edges of neighboring cells.

Mixed precision

Mixed precision is the utilization of different precisions for certain aspects of an algorithm. For the ADER-DG method we isolate four kernels of interest: the persistent storage, the predictor, the corrector and the Picard-iteration method used for the space-time expansion of the solution in the predictor for nonlinear equations.

Each of the previously presented scenarios are recomputed using mixed-precision to evaluate the impact of numerical precision on each of these kernels. We find that while the predictor typically has the highest impact on the results, the corrector and storage precisions are critical for the stability of certain equations.

	acoustic			elastic			Euler			
prec	predictor	corrector	storage	predictor	corrector	storage	predictor	corrector	storage	Picard
bf16	$3.06e^{-1}$	$2.09e^{-1}$	$8.45e^{-1}$	$4.50e^{-1}$	$1.84e^{-1}$	$7.58e^{-1}$	$1.42e^{-1}$	NAN	NAN	$2.32e^{-2}$
fp16	NAN	$7.34e^{-3}$	$7.28e^{-2}$	NAN	$1.35e^{-2}$	$2.12e^{-1}$	$3.20e^{-2}$	$2.08e^{-2}$	$2.12e^{-2}$	$2.97e^{-2}$
fp32	$9.84e^{-6}$	$5.21e^{-7}$	$8.07e^{-6}$	$1.58e^{-5}$	$1.61e^{-6}$	$1.54e^{-5}$	$2.65e^{-6}$	$1.62e^{-6}$	$2.48e^{-6}$	$2.25e^{-5}$
fp64	$5.75e^{-19}$			$1.66e^{-14}$			$6.56e^{-7}$			

Table 2: Final L2-error integrated over the domain for the three non-static scenarios computed with mixed-precision on a grid of 27x27 cells. One of predictor, corrector, storage or Picard-iterations was performed in the specified precision, all others were computed in fp64-precision.

	SWE resting lake				Euler isotropic vortex			
prec	predictor	corrector	storage	Picard	predictor	corrector	storage	Picard
bf16	$2.37e^{-01}$	NAN	$4.49e^{-01}$	$1.29e^{0}$	NAN	NAN	NAN	$1.77e^{-01}$
fp16	NAN	NAN	$5.53e^{-02}$	NAN	NAN	$1.84e^{-01}$	$4.55e^{-01}$	$3.41e^{-02}$
fp32	$1.23e^{-04}$	$5.78e^{-05}$	$3.56e^{-06}$	$3.26e^{-04}$	$4.30e^{-05}$	$2.46e^{-05}$	$8.19e^{-05}$	$9.40e^{-06}$
fp64	$7.44e^{-12}$				$6.86e^{-06}$			

Table 3: Final L2-error integrated over the domain for both static scenarios on a grid of 27x27 cells. One of predictor, corrector, storage or Picard-iterations was computed in the specified precision, all others were performed in fp64-precision.

Variable precision

The term variable precision is used when the numerical precision differs between areas of the simulated domain.

Here we are running the homogeneous half-space scenario (HHS1) [Kri09] for elastic-wave propagation. This consists of a singular point source in an infinite domain, and is used to asses the modeling of a planar free surface.

Using a mesh of 27x27x27 cells the solver produces accurate results in fp64, but in bf16 these contain strong oscillations.

ExaHyPE2 supports using multiple solvers concurrently, so using two coupled solvers we compute the top 4 layers of cells in fp64 and all others in bf16, totaling 2916 out of 19683 total cells computed in fp64.

As seen in Fig. 5, most oscillations disappear. Variable precision can therefore be used to exploit low-precision computation when only certain key areas are of interest.

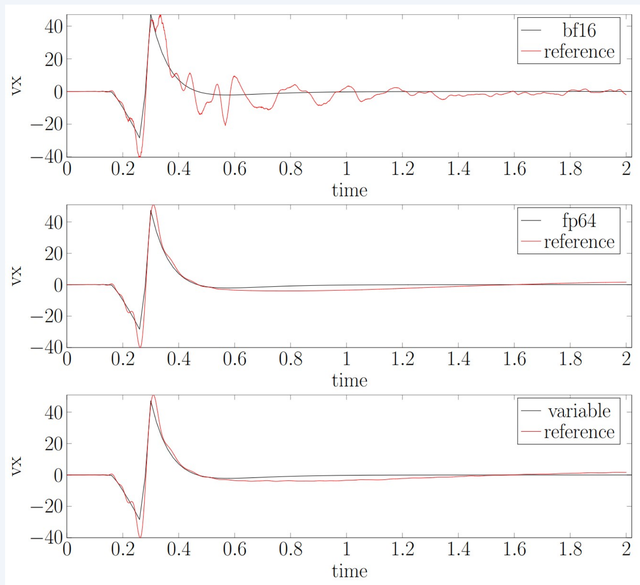

Fig. 5: Comparison of the simulated and reference velocity in x-direction for the first receiver of the HHS1 scenario as computed in fp64-precision, bf16-precision, or using both concurrently. In the latter case the uppermost 4 of the total 27 layers of cells where computed in fp64 while the rest were computed in bf16.

Profiling

While the focus here is on the influence of precision on the results of a simulation, we can briefly look at how it impacts the performance as well.

- Reducing the precision improves the runtime through higher effective vectorization, reduced bandwidth and improved caching

- In ExaHYPE2, reducing the precision of an HHS1 simulation from fp64 to fp32 reduces the average bandwidth by 59%

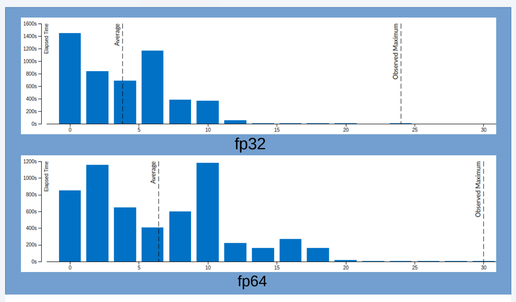

Fig. 6: DRAM Bandwidth utilization in GB/s of the HHS1 scenario simulated in fp64 and fp32-precision in ExaHyPE2, measured using the Intel VTune profiler [Vtu24]

References

[Rei20] Anne Reinarz, Dominic E. Charrier, Michael Bader, Luke Bovard, Michael Dumbser, Kenneth Duru, Francesco Fambri, Alice-Agnes Gabriel, Jean-Matthieu Gallard, Sven Köppel, Lukas Krenz, Leonhard Rannabauer, Luciano Rezzolla, Philipp Samfass, Maurizio Tavelli, and Tobias Weinzierl. 2020. ExaHyPE: An engine for parallel dynamically adaptive simulations of wave problems. Computer Physics Communications 254 (2020), 107251. doi: 10.1016/j.cpc.2020.10725

[Dum08] Michael Dumbser, Dinshaw S. Balsara, Eleuterio F. Toro, and Claus-Dieter Munz. 2008. A unified framework for the construction of one-step finite volume and discontinuous Galerkin schemes on unstructured meshes. J. Comput. Phys. 227, 18 (2008), 8209–8253. https://doi.org/10.1016/j.jcp.2008.05.025

[Wei19] Tobias Weinzierl. The peano software-parallel, automaton-based, dynamically adaptive grid traversals. ACM Trans. Math. Softw., 45(2), April 2019.

[Kri09] Kristeková, M., Kristek, J., Moczo, P.: Time-frequency misfit and goodness-of-fit criteria for quantitative comparison of time signals. Geophys. J. Int. 178, 813–825 (2009)

[Vtu24] intel Vtune Profiler User Guide V. 2024.1, accessed 18/04/24, https://intel.com/content/www/us/en/docs/vtune-profiler/user-guide/2024-1/overview.html

[Shu99] Hu, C., Shu, C.W.: Weighted essentially non-oscillatory schemes on triangular meshes. J. Comput. Phys. 150, 97–127 (1999)

[Mat20] Ioriatti Matteo, Dumbser Michael, and Loubère Raphaël. A staggered semi-implicit discontinuous galerkin scheme with a posteriori subcell finite volume limiter for the euler equations of gas dynamics. Journal of Scientific Computing, 83, 04 2020.

[Ieee19] "IEEE Standard for Floating-Point Arithmetic," in IEEE Std 754-2019 (Revision of IEEE 754-2008) , vol., no., pp.1-84, 22 July 2019, doi: 10.1109/IEEESTD.2019.8766229.

Acknowledgements

The presented work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) as part of the project no. 462423388, *ExaHyPE MVP -- mixed and variable precision for an Exascale Hyperbolic PDE Engine*.

We thank Tobias Weinzierl and his group, at Durham University, for collaboration, hints and support on the ExaHyPE engine and the Peano framework.
We also thank Michael Dumbser (University of Trento) for comments and discussions regarding the ADER-DG algorithm.