Matrix unloaded: Graphics processor-boosted solvers for diffusion physics

June 25 2024, by Holly Auten

Developed by LLNL and Portland State University researchers, innovative matrix-free solvers offer performance gains for complex multiphysics simulations. Credit: Amanda Levasseur

Underlying all complex multiphysics simulations are even more complex mathematical algorithms that solve the equations describing movement of physical phenomena—for instance, the radiation diffusion and burning plasma processes inherent in fusion reactions. Preconditioned

solvers are often used in these algorithms to transform a problem so that it quickly and accurately converges to a solution.

A [recent paper](link) in *SIAM Journal on Scientific Computing* introduces specialized solvers optimized for simulations running on graphics processing unit (GPU)–based supercomputers. The authors are computational mathematician Tzanio Kolev from LLNL's Center for Applied Scientific Computing alongside Will Pazner and Panayot Vassilevski, both former Livermore researchers now at Portland State University.

The team's algorithms solve the radiation diffusion equations underpinning MARBL, a mission-critical hydrodynamics code that is now running on GPUs like those in Livermore's forthcoming exascale supercomputer El Capitan.

"MARBL's radiation diffusion model requires a specific type of solver in order to realize GPU efficiency. Our algorithms are optimized for this scenario," said Pazner, the paper's lead author.

Breaking down the math

For faster computation, these solvers subdivide physical systems into a finite element space known as H(div). This type of discretization increases the number of finite elements while decreasing their size—think of a 3D mesh breaking down into smaller and smaller squares while retaining a constant overall volume.

H(div) is one of four finite element spaces in a set of differential equations known as the de Rham complex, all of which are incorporated into the Livermore-led MFEM (Modular Finite Element Methods) software library.

"Throughout our research, we'd developed efficient solvers for the other spaces but hadn't quite figured out the best way to approach H(div). We knew we could do better," Koley said.

Many finite elements are calculated with matrices, in which numbers are arranged into rows and columns. But when a matrix is too large to compute with efficiency—such as in large-scale hydrodynamics simulations—matrix-free algorithms provide the only feasible way forward.

"High-order problems become prohibitively expensive to run on GPUs, not just because of the size to compute but also the memory transfer. Matrix-based algorithms often cause memory bottlenecks on GPUs, so we needed to come up with algorithms that arrive at the same solution while avoiding the processing and storing of the matrix," Pazner said.

Though armed with [matrix-free de Rham solvers](), the team was still missing something to lighten H(div)'s computational burden. The turning point was their decision to reformulate the problem in the context of a saddle-point system, whose surfaces resemble a riding saddle when plotted on a graph. In essence, they turned the original problem into two coupled problems with a different mathematical structure.

The saddle-point technique may seem counterintuitive, but as the researchers tried other techniques, this one eventually rose to the top.

"Recasting the problem into a saddle-point formulation expands the original condensed, tightly packed formulation," Pazner said. "The structure becomes bigger but also much more amenable to solution."

The team's innovative H(div) solvers efficiently combine all of the above approaches. "Looking at the problem in saddle-point form enabled us to take advantage of the matrix-free structure in a favorable way," Kolev

said, crediting his co-authors with the breakthrough.

"H(div) was the last tool in the de Rham toolbox for our users. We'd been considering this problem for a long time, and now we have state-of-the-art methods in all of the de Rham spaces."

From CPUs to GPUs

As years of LLNL's high performance computing (HPC) research demonstrate, portability from central processing units (CPUs) to GPUs is the primary challenge in an application's scalability from one type of HPC architecture to another. In this context, faster math leads to faster physics.

"Next-generation HPC systems promise huge speedups, but these benefits are only realized if the solvers are optimized for modern hardware," Pazner said.

"GPUs changed everything. Problems you thought you knew how to solve suddenly became interesting again," Kolev said.

The team tested their optimized solvers on radiation diffusion benchmarking problems using Livermore's CPU-based Quartz and hybrid CPU/GPU Lassen supercomputers, resulting in significant speedups with fewer compute resources. For instance, applications that require more nodes and longer run times on Quartz can run in just a few seconds on a single Lassen node. This productivity boost is crucial as codes like MARBL are ported to El Capitan.

"These new preconditioners have enabled scalable performance on both CPU and GPU architectures for an important class of diffusion-based physics solvers, which are a critical component of our multiphysics simulation codes," MARBL project lead Rob Rieben said. "Using these

advanced methods, we can achieve performance in important simulations including inertial confinement fusion."

More information: Will Pazner et al, Matrix-Free High-Performance Saddle-Point Solvers for High-Order Problems in \(\boldsymbol{H}(\operatorname{\textbf{div}})\), *SIAM Journal on Scientific Computing* (2024). [DOI: 10.1137/23M1568806](https://doi.org/10.1137/23M1568806)

Provided by Lawrence Livermore National Laboratory

Citation: Matrix unloaded: Graphics processor-boosted solvers for diffusion physics (2024, June 25) retrieved 29 June 2024 from https://techxplore.com/news/2024-06-matrix-unloaded-graphics-processor-boosted.html