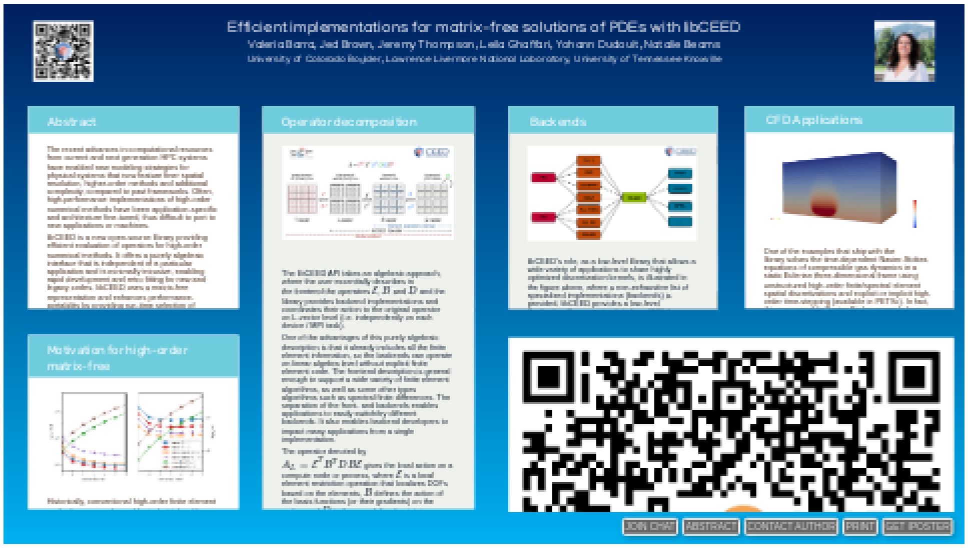
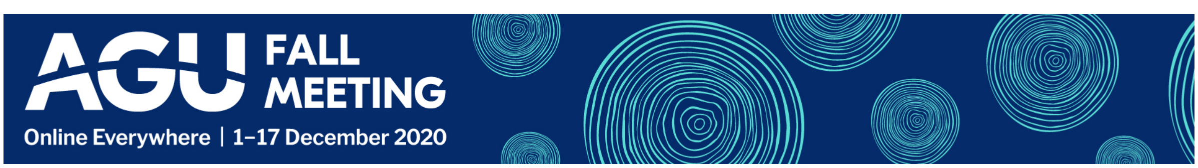
Efficient implementations for matrix-free solutions of PDEs with libCEED



Valeria Barra, Jed Brown, Jeremy Thompson, Leila Ghaffari, Yohann Dudouit, Natalie Beams

University of Colorado Boulder, Lawrence Livermore National Laboratory, University of Tennessee Knoxville

PRESENTED AT:

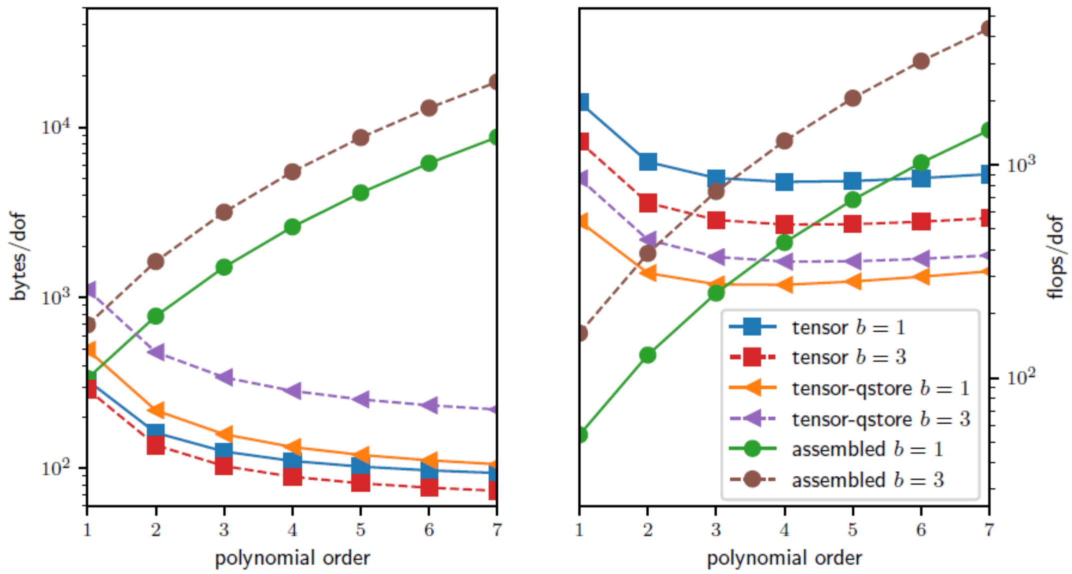


ABSTRACT

The recent advances in computational resources from current and next generation HPC systems have enabled new modeling strategies for physical systems that now feature finer spatial resolution, higher-order methods and additional complexity, compared to past frameworks. Often, high-performance implementations of high-order numerical methods have been application-specific and architecture fine-tuned, thus difficult to port to new applications or machines.

libCEED is a new open-source library providing efficient evaluation of operators for high-order numerical methods. It offers a purely algebraic interface that is independent of a particular application and is minimally intrusive, enabling rapid development and retro fitting for new and legacy codes. libCEED uses a matrix-free representation and enhances performance-portability by providing run-time selection of implementations tuned for a variety of computational devices including CPUs and GPUs (with both CUDA and HIP support). We introduce libCEED's interface and demonstrate its usage and performance through examples in computational fluid dynamics and solid elasticity that are of interest to the Earth sciences community at large.

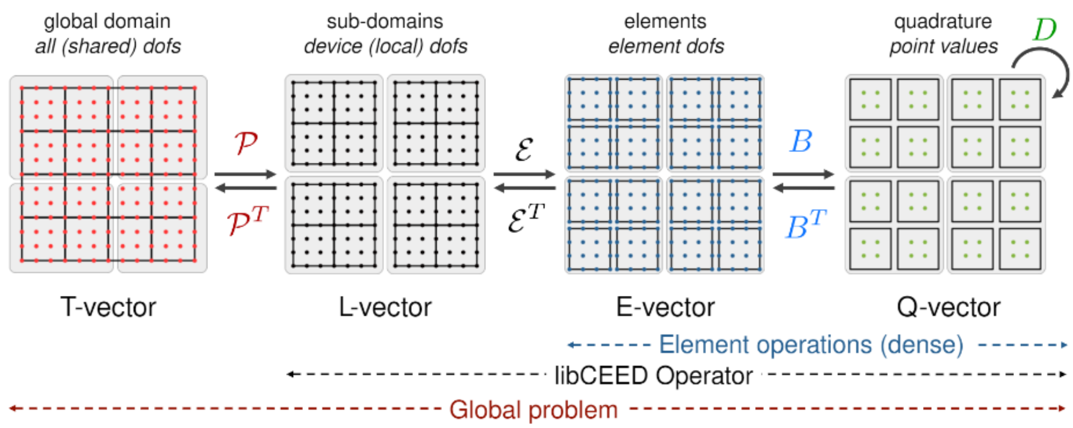
MOTIVATION FOR HIGH-ORDER MATRIX-FREE



Historically, conventional high-order finite element methods were rarely used for industrial problems because the Jacobian rapidly loses sparsity as the order is increased, leading to unaffordable solve times and memory requirements. This effect typically limited the order of accuracy to at most quadratic, especially because they are computationally advantageous in terms of floating point operations (FLOPS) per degree of freedom (DOF)—despite the fast convergence and favorable stability properties offered by higher order discretizations. Nowadays, high-order numerical methods, such as the spectral element method (SEM)—a special case of nodal p-Finite Element Method (FEM) which can reuse the interpolation nodes for quadrature—are employed, especially with (nearly) affine elements, because linear constant coefficient problems can be very efficiently solved using the fast diagonalization method combined with a multilevel coarse solve.

OPERATOR DECOMPOSITION

$$A = \mathcal{P}^T \mathcal{E}^T B^T D B \mathcal{E} \mathcal{P}$$



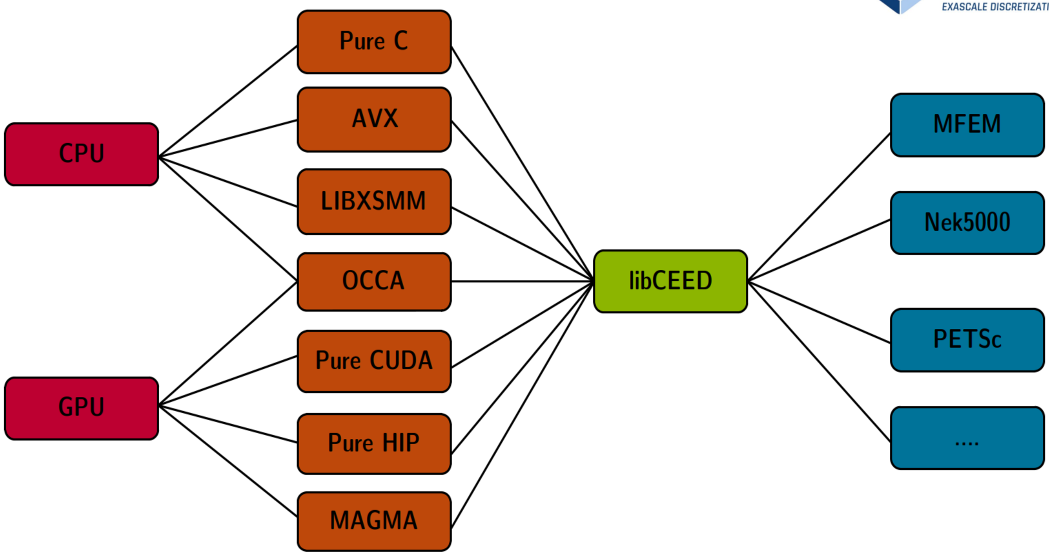
The libCEED API takes an algebraic approach, where the user essentially describes in the frontend the operators $\mathcal{E}$, B and D and the library provides backend implementations and coordinates their action to the original operator on L-vector level (i.e. independently on each device / MPI task).

One of the advantages of this purely algebraic description is that it already includes all the finite element information, so the backends can operate on linear algebra level without explicit finite element code. The frontend description is general enough to support a wide variety of finite element algorithms, as well as some other types algorithms such as spectral finite differences. The separation of the front- and backends enables applications to easily switch/try different backends. It also enables backend developers to impact many applications from a single implementation.

The operator denoted by
$A_L = \mathcal{E}^T B^T D B \mathcal{E}$ gives the local action on a compute node or process, where $\mathcal{E}$ is a local element restriction operation that localizes DOFs based on the elements, B defines the action of the basis functions (or their gradients) on the nodes, and D is the user-defined pointwise function describing the physics of the problem at the quadrature points, also called the QFunction in libCEED's API.

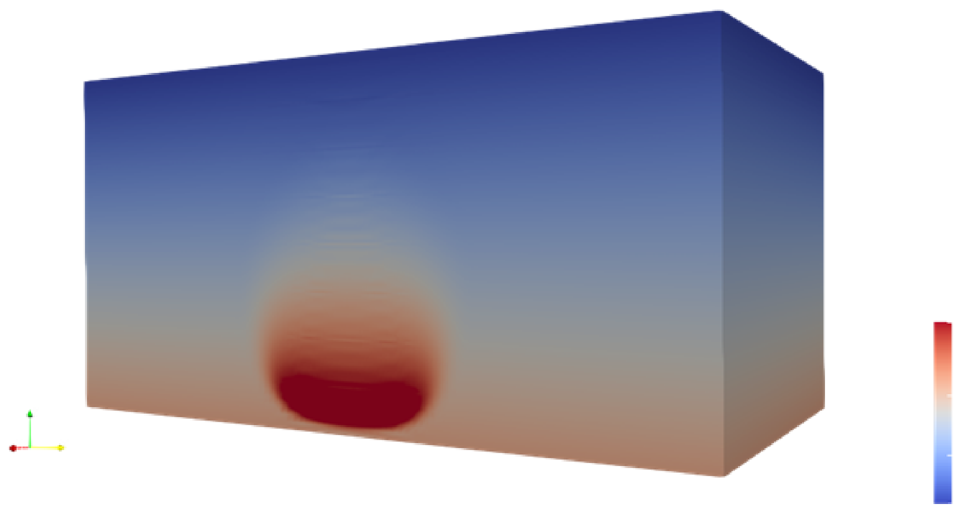
libCEED is an open-source library written in C, with currently available Fortran, Python, Rust and Julia interfaces.

BACKENDS

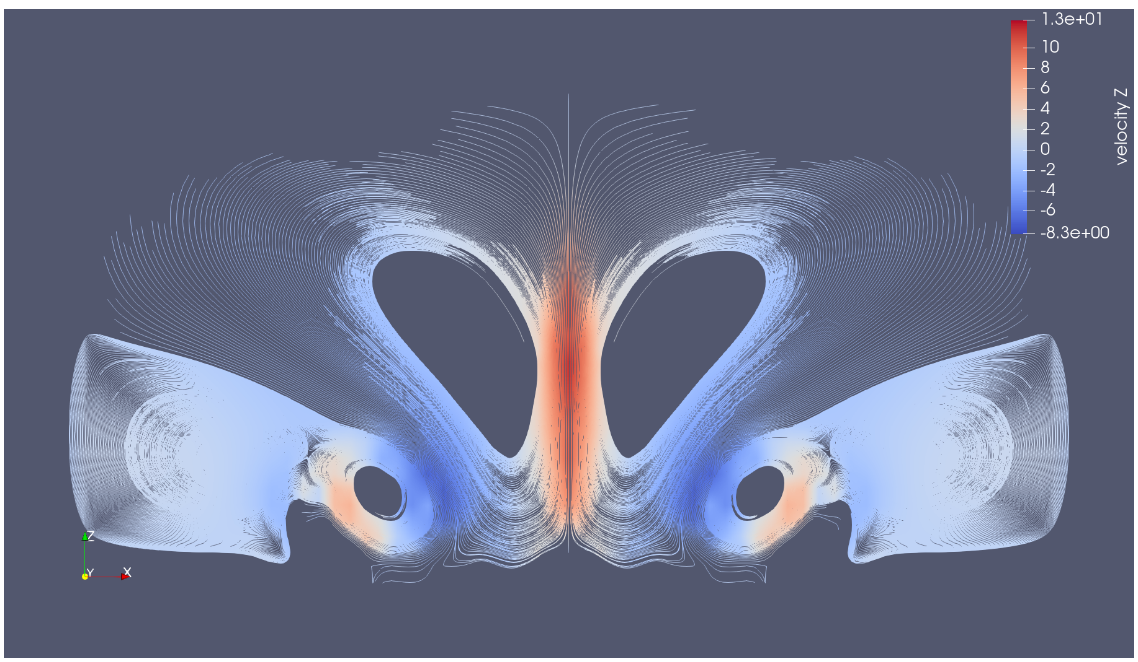


libCEED’s role, as a low-level library that allows a wide variety of applications to share highly optimized discretization kernels, is illustrated in the figure above, where a non-exhaustive list of specialized implementations (backends) is provided. libCEED provides a low-level Application Programming Interface (API) for user codes so that applications with their own discretization infrastructure (e.g., those in PETSc, MFEM and Nek5000) can evaluate and use the core operations provided by libCEED. GPU implementations are available via pure CUDA or HIP as well as the OCCA and MAGMA libraries. CPU implementations are available via pure C and AVX intrinsics as well as the LIBXSMM library. libCEED provides a unified interface, so that users only need to write a single source code and can select the desired specialized implementation at run time. Moreover, each process or thread can instantiate an arbitrary number of backends.

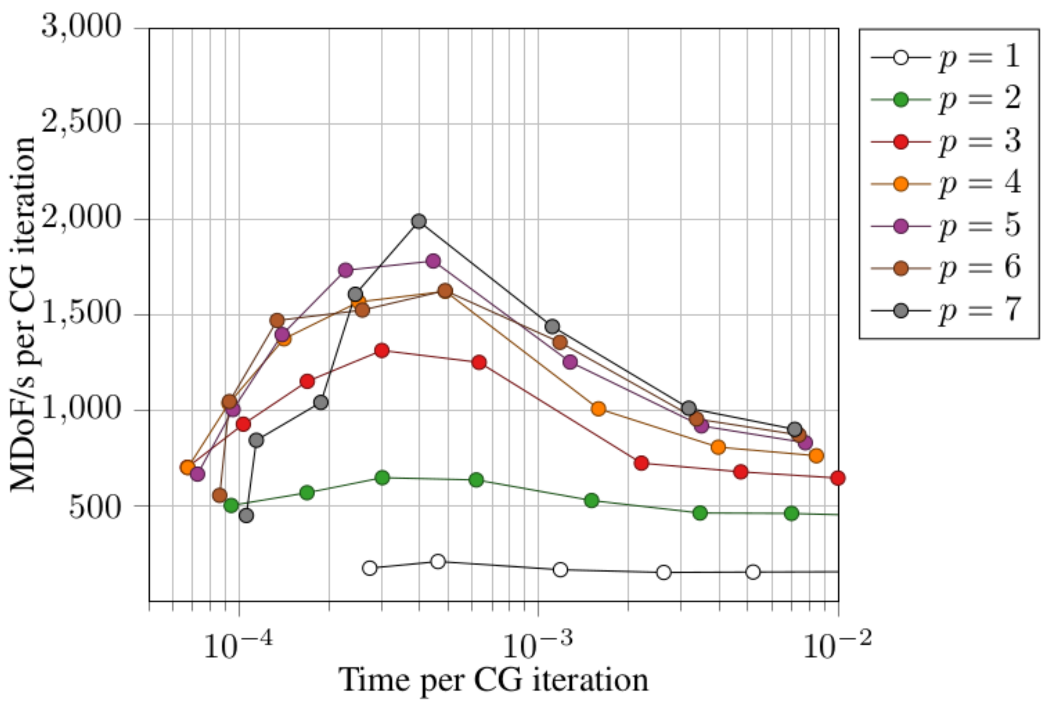
CFD APPLICATIONS

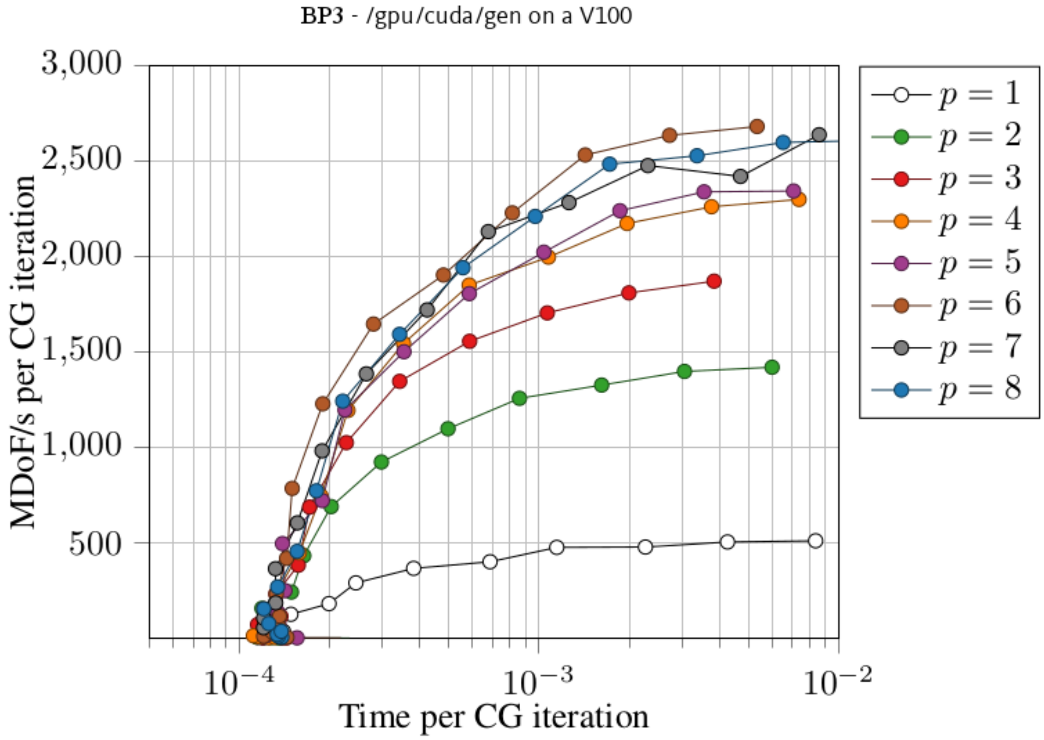


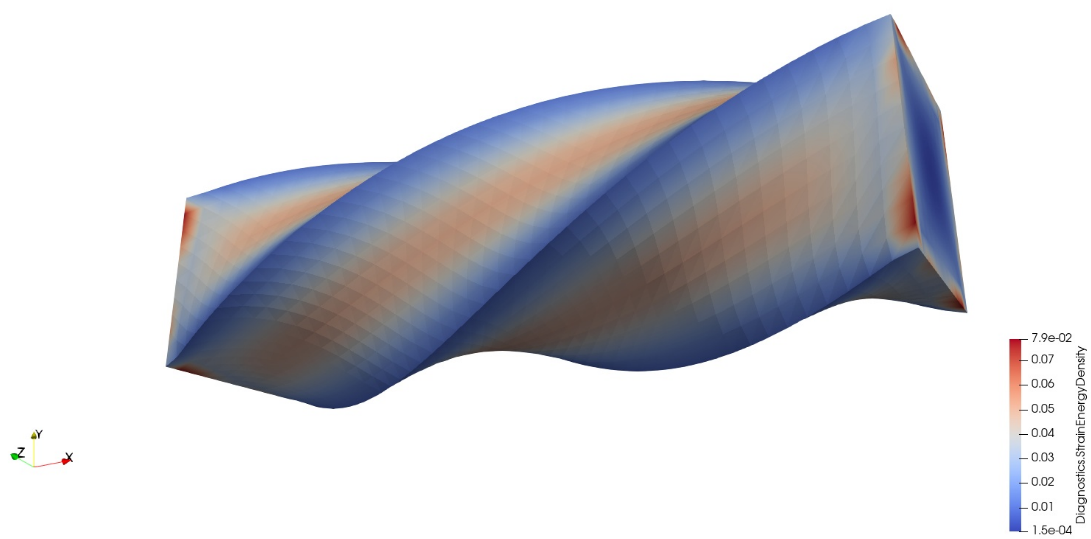
One of the examples that ship with the library solves the time-dependent Navier-Stokes equations of compressible gas dynamics in a static Eulerian three-dimensional frame using unstructured high-order finite/spectral element spatial discretizations and explicit or implicit high-order time-stepping (available in PETSc). In fact, this compressible Navier-Stokes example has been developed using PETSc, so that the pointwise physics (defined at quadrature points) is separated from the parallelization and meshing concerns. In the image above, we show the solution of the Density Current problem using an unstructured cartesian mesh, with polynomial order p=10 and have nearly 5M DOFs.



BP3 - /cpu/self/xsmm/blocked on a 2-socket AMD EPYC 7452




BP3 - /gpu/cuda/gen on a V100
3,000
2,500
2,000
1,500
1,000
500
MDoF/s per CG iteration
10^{-4}
10^{-3}
10^{-2}
Time per CG iteration
$p = 1$
$p = 2$
$p = 3$
$p = 4$
$p = 5$
$p = 6$
$p = 7$
$p = 8$


7.9e-02
0.07
0.06
0.05
0.04
0.03
0.02
0.01
1.5e-04
Diagnostics.StrainEnergyDensity

ABSTRACT

The recent advances in computational resources from current and next generation HPC systems have enabled new modeling strategies for physical systems that now feature finer spatial resolution, higher-order methods and additional complexity, compared to past frameworks. Often, high-performance implementations of high-order numerical methods have been application-specific and architecture fine-tuned, thus difficult to port to new applications or machines.

libCEED is a new open-source library providing efficient evaluation of operators for high-order numerical methods. It offers a purely algebraic interface that is independent of a particular application and is minimally intrusive, enabling rapid development and retro fitting for new and legacy codes. libCEED uses a matrix-free representation and enhances performance-portability by providing run-time selection of implementations tuned for a variety of computational devices including CPUs and GPUs (with both CUDA and HIP support). We introduce libCEED's interface and demonstrate its usage and performance through examples in computational fluid dynamics and solid elasticity that are of interest to the Earth sciences community at large.