Performant Low-order Matrix-free Finite Element Kernels on GPU Architectures

Randolph R. Settgast
William R. Tobin
Yohann Dudouit
Nicola Castelletto
Ben C. Corbett*
settgast1@llnl.gov
Lawrence Livermore National Laboratory
Livermore, Ca, USA

Sergey Klevstov
Leland Stanford University
Palo Alto, Ca, USA

ABSTRACT

Numerical methods such as the Finite Element Method (FEM) have been successfully adapted to utilize the computational power of GPU accelerators. However, much of the effort around applying FEM to GPU's has been focused on high-order FEM due to higher arithmetic intensity and order of accuracy. For applications such as the simulation of geologic reservoirs, high levels of heterogeneity results in high-resolution grids characterized by highly discontinuous (cell-wise) material property fields. Additionally, the significant uncertainties typical of geologic reservoirs reduces the benefits of high order accuracy, and low-order methods are typically employed. In this study, we present a strategy for implementing highly performant low-order matrix-free FEM operator kernels in the context of the conjugate gradient (CG) method. Performance results of matrix-free Laplace and isotropic elasticity operator kernels are presented and are shown to compare favorably to matrix-based SpMV operators on V100, A100, and MI250X GPUs.

CCS CONCEPTS

• **Applied computing** → **Physical sciences and engineering**.

KEYWORDS

finite elements, low-order, matrix-free, GPU

ACM Reference Format:
Randolph R. Settgast, William R. Tobin, Yohann Dudouit, Nicola Castelletto, Ben C. Corbett, and Sergey Klevstov. 2023. Performant Low-order Matrix-free Finite Element Kernels on GPU Architectures. In *Proceedings of The International Conference for High Performance Computing, Networking, Storage, and Analysis (SC '23).* ACM, New York, NY, USA, 3 pages. https://doi.org/XXXXXXX.XXXXXXX

SC '23, November 12–17, 2023, Denver, CO

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Many engineering application contain highly heterogeneous material properties and in-situ state variables with high levels of uncertainty. For example, in subsurface reservoir modeling data are typically provided at a resolution representative of the heterogeneity/natural layering of the geologic formation as determined by a geophysical analysis. As such, low order methods dominate some applications such as the subsurface reservoir community.

The application of a low order FE method typically involves allocation and filling of a global sparse matrix of size ($ndof \cdot nnzr$), where $ndof$ is the global number of degrees of freedom in the problem, and $nnzr$ is the number of non-zeros in a row of the matrix. For iterative Krylov methods, these large sparse matrices are used as a vector operator through a sparse matrix vector multiply $\mathbf{Ax} = \mathbf{b}$. In contrast, a matrix-free approach eliminates the need to allocate/fill/store the sparse global stiffness matrix, which results in a significant reduction in the amount of memory required to perform a simulation.

In the case of a matrix-based operator, the speed of sparse matrix vector product has a low arithmetic intensity, i.e. is bound by the memory bandwidth (e.g. [1]). A matrix-free operator potentially allows a reduction in the bandwidth required to carry out the operator application, which can lead to faster runtime and better utilization of the GPU's capabilities in some cases.

In this work, we present a strategy for implementing highly performant low-order matrix-free FE operator kernels for the case of a Laplace problem and Linear Elasticity. These kernels are applied within an un-preconditioned conjugate gradient (CG) iteration to provide context for the potential performance in an iterative Krylov method.

2 APPROACH

In this work we show the results of a pair (Laplace, Isotropic linear elasticity) of low order matrix-free finite element kernels. The throughput of these kernels are compared to the throughput for simply transferring the memory for a SpMV over HBM a single time (i.e. no bytes are transferred more than once). We refer to this measure as NoOp-SpMV. A strong scaling study for a single MPI rank are conducted for Nvidia V100, A100, and AMD mi250x GPU's.

The algorithm for the matrix-free finite element operator for linear elasticity is given in Algorithm 1. The implementation of

this algorithm involves maximizing the information available to the compiler. For instance, the finite element type is run through a static dispatch, which provides the compiler all relevant dimensions at compile time. Additionally, the action of the constitutive response is provided to the compiler in a similar manner. All fixed size loops are unrolled using a pragma statement, or by hand in some cases to allow for the propagation of constexpr data through the kernel.

Algorithm 1 Matrix-free action of elasticity operator $A(\cdot)$

Input:
- $\mathbf{x}$: global support point coordinates
- $\mathbf{u}$: global input vector
- D: operator specifying the physics

Output:
- $\mathbf{v}$: global output vector

templated Class/ templated Method:
$Class\langle FemType, MatType\rangle$::$kernelLaunch\langle LaunchPolicy\rangle$
for each element e **do**
  Gather/Load data from $\mathbf{x}$ and $\mathbf{u}$ into element local storage
    $\mathbf{x}^e = \{\boldsymbol{x}_a\}_{a=0}^{n_{supppts}} \leftarrow \mathbf{x}$,
    $\mathbf{u}^e = \{\boldsymbol{u}_a\}_{a=0}^{n_{supppts}} \leftarrow \mathbf{u}$
  for each integration point q **do**
    Calculate Jacobian transformation at integration point q:
      $J_e = J_e(\hat{\boldsymbol{x}}_q) = \sum_a \boldsymbol{x}_a \otimes \hat{\nabla}\hat{\phi}_a(\hat{\boldsymbol{x}}_q)$
    Calculate inverse Jacobian transformation J_e^{-1} and $\det(J_e)$
    Calculate gradient of input variable
      $\nabla \boldsymbol{u}_e = \nabla \boldsymbol{u}_e(\hat{\boldsymbol{x}}_q) = \left(\sum_a \boldsymbol{u}_a \otimes \hat{\nabla}\hat{\phi}_a(\hat{\boldsymbol{x}}_q)\right) J_e^{-1}$
    Calculate action of stiffness operator (i.e. stress):
      $\boldsymbol{\sigma}_e = D(\nabla \boldsymbol{u}_e)$
    Apply transform and quadrature weight to stress
      $\boldsymbol{P}_e = w_q \det(J_e)\boldsymbol{\sigma}_e J_e^{-T}$
    Apply the gradient of the test functions:
      $\boldsymbol{v}_a$ += $\boldsymbol{P}_e \cdot \hat{\nabla}\hat{\phi}_a(\hat{\boldsymbol{x}}_q)$
  end for
  add element local output vector ($\mathbf{v}^e$) to global output vector ($\mathbf{v}$)
    $\mathbf{v}^e \rightarrow \mathbf{v}$ (atomic add)
end for

3 NUMERICAL RESULTS

Figure 1 shows that the matrix-free operator kernel achieves peak performance once the problem size exceeds 10^6 degrees of freedom. Once the matrix-free operator kernel is applied to a problem size where peak performance is achieved, the throughput moderately outperforms the theoretical maximum throughput of SpMV across all architectures. (1x-1.5x) There is approximately a 1.4x jump in throughput between the V100 and the A100, and approximately a 1.15x jump in throughput between the A100 and the MI250X. The lack of a large increase in performance for the MI250X over the A100 despite a 2.5x higher dflop capacity is the higher bandwidth of the A100.

Figure 1: Laplace - A(x) Kernel Throughput

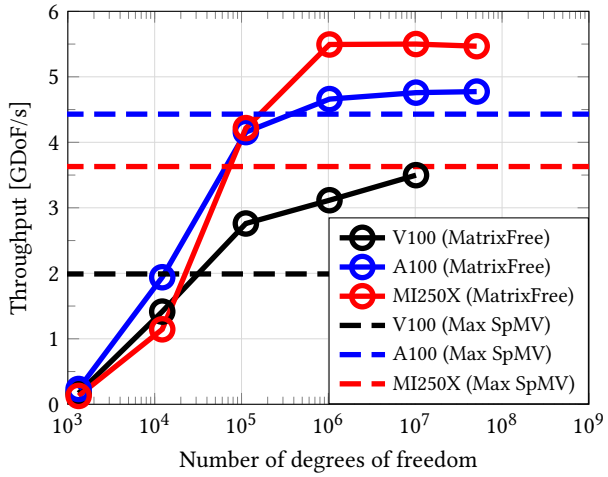

Figure 2: Isotropic Linear Elasticity - A(x) Kernel Throughput

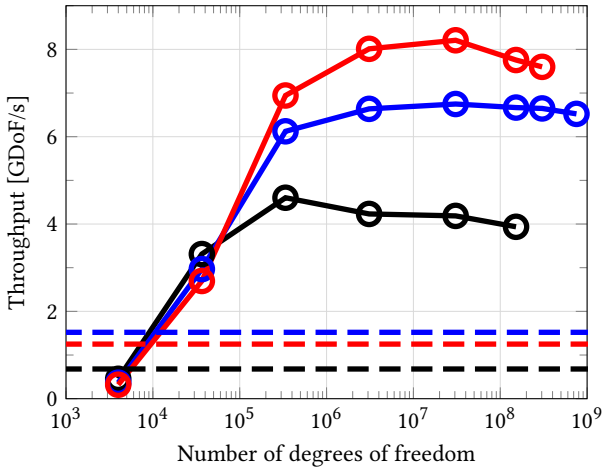

Figure 2 shows that the matrix-free operator kernel achieves peak performance once the problem size exceeds 10^6 degrees of freedom. Once the matrix-free operator kernel is applied to a problem size where peak performance is achieved, the throughput significantly outperforms the theoretical maximum throughput of SpMV across all architectures. (4x-7x)

4 CONCLUSION

In this work, we have shown that the proposed low order matrix-free fem operator kernel performs well compared to the cost of transferring the data for a sparse matrix, and input/output vectors a single time from HBM. In the case of Laplace's problem the matrix-free operator performance is similar to the NoOp-SpMV, and in the case of linear elasticity the matrix-free operator significantly outperforms NoOp-SpMV. Given the performance of the matrix-free operator compared to the NoOp-SpMV and the reduction in memory allocation requirements, the use of a matrix-free appears to have potential as a viable alternative to SpMV.

5 ACKNOWLEDGMENTS

This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

Portions of this work were performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07-NA27344.

REFERENCES

[1] P. Zardoshti, F. Khunjush, and H. Sarbazi-Azad, "Adaptive sparse matrix representation for efficient matrix-vector multiplication," in *Advances in GPU Research and Practice*. Elsevier Inc., 2017, pp. 349–369.