Differentiating Large-Scale Finite Element Applications with MFEM

CASC
Center for Applied Scientific Computing

Julian Andrej

LLNL-PRES-2010944

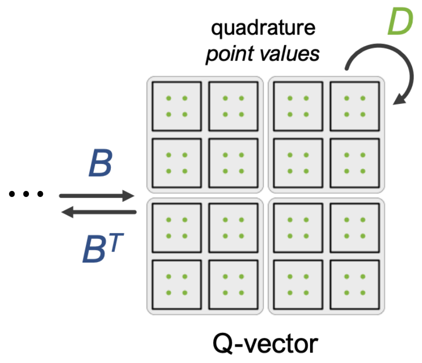
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

Usual approach to Automatic Differentiation

- AD is often perceived as a black-box tool
  - Applied at the highest possible level
  - Low implementation barrier
- AD tool has to work through
  - Complicated program structures
  - Non-trivial object types
  - Sometimes even communication layers like MPI
- Infeasible overhead in our applications even for the smallest problems
  - We must balance the implementation effort with AD convenience to ensure the least overhead possible
  - Carefully decide entry points
  - Still provide a clear and concise interface for users
- GPUs make things worse 100x (underestimated guess)

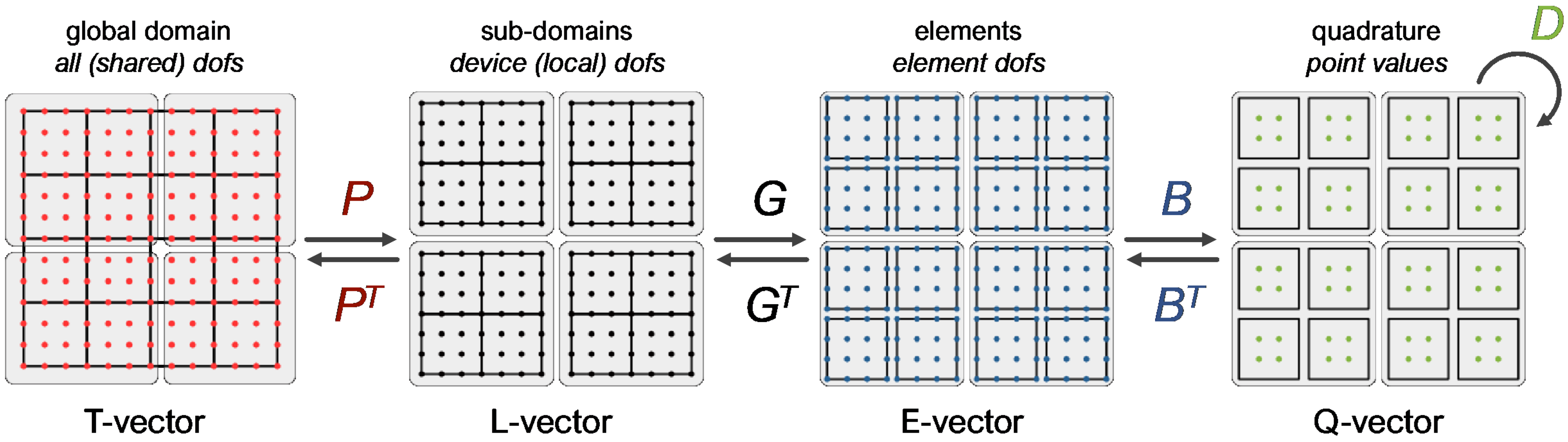
Constrain AD to the quadrature point level



Finite Element Operator Decomposition

Decompose A into parallel, mesh, finite element, and geometry/physics components

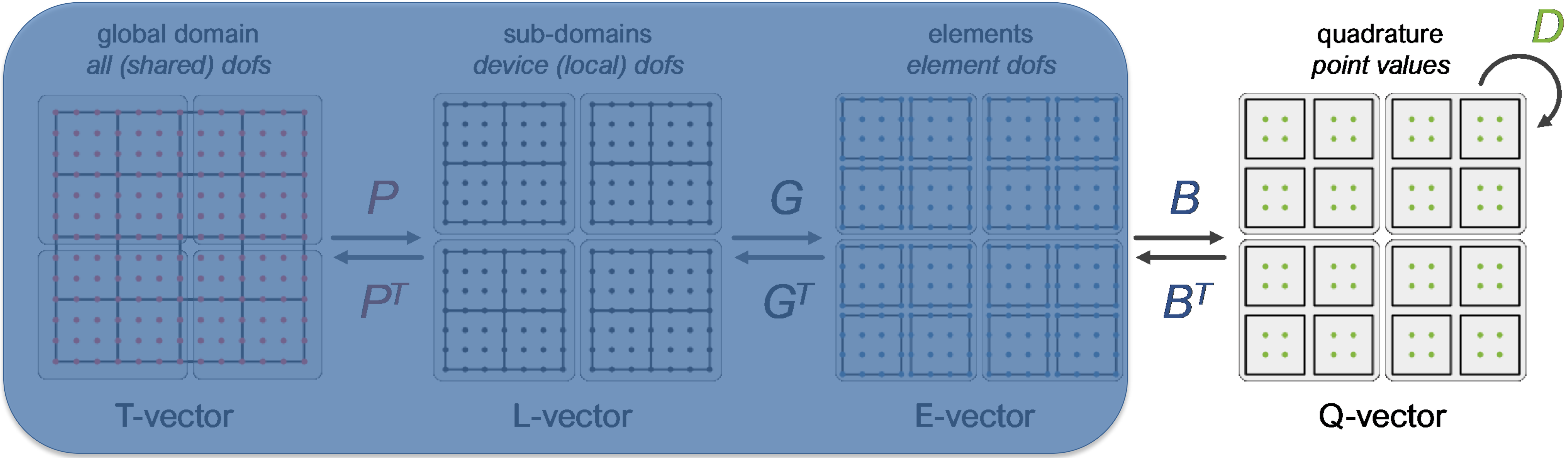
$$A = P^T G^T B^T D B G P$$



Finite Element Operator Decomposition

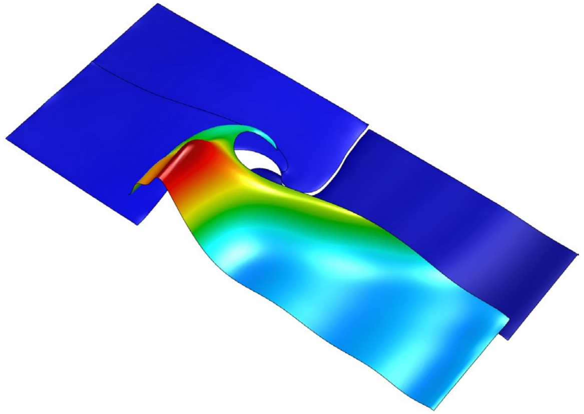
Decompose A into parallel, mesh, finite element, and geometry/physics components

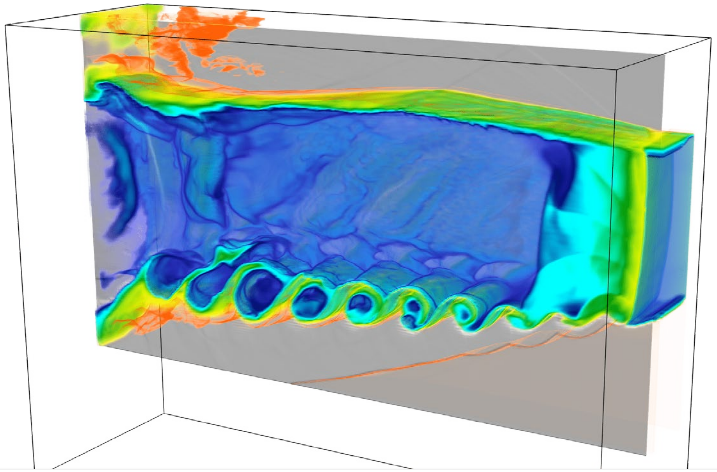
$$\nabla_u A(u;\rho) = P^T G^T B^T \nabla_{\hat{u}} D(\hat{u},\hat{\rho})$$



Why Enzyme for Automatic Differentiation?

- Production codes are compositions of complicated physics
  - Variety of domain experts work on parts of the code
  - Multiple programming languages
- Material models lead to additional experts that only see and work on specific parts of the code
- Enzyme allows us to still use the complex combination of a libraries which
  - May be written in a different programming language
  - Can’t be modified to allow different approaches of AD (e.g., type overloading)
  - User might be allowed to have a compiled binary, but no clearance to compile the code himself



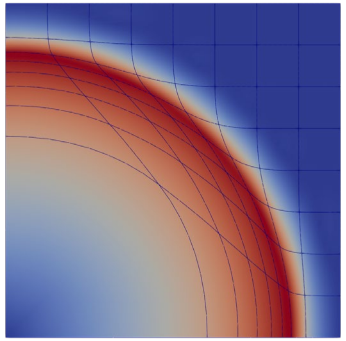


Current feature set

- Laghos (Lagrangian phase of an ALE Shock Hydrodynamics solver) replicated in dFEM with explicit time integration
- Extended implementation to support implicit time integration with automated derivatives from dFEM using Enzyme!

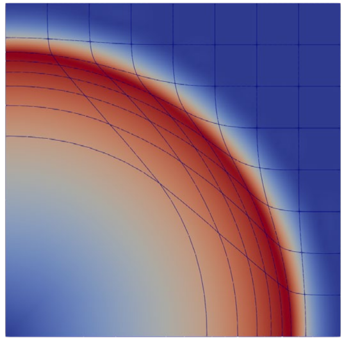
Laghos RK4 CFL 0.5

step 737, t = 0.8, dt = 0.0011935
Energy difference (initial vs final): 3.60e-05

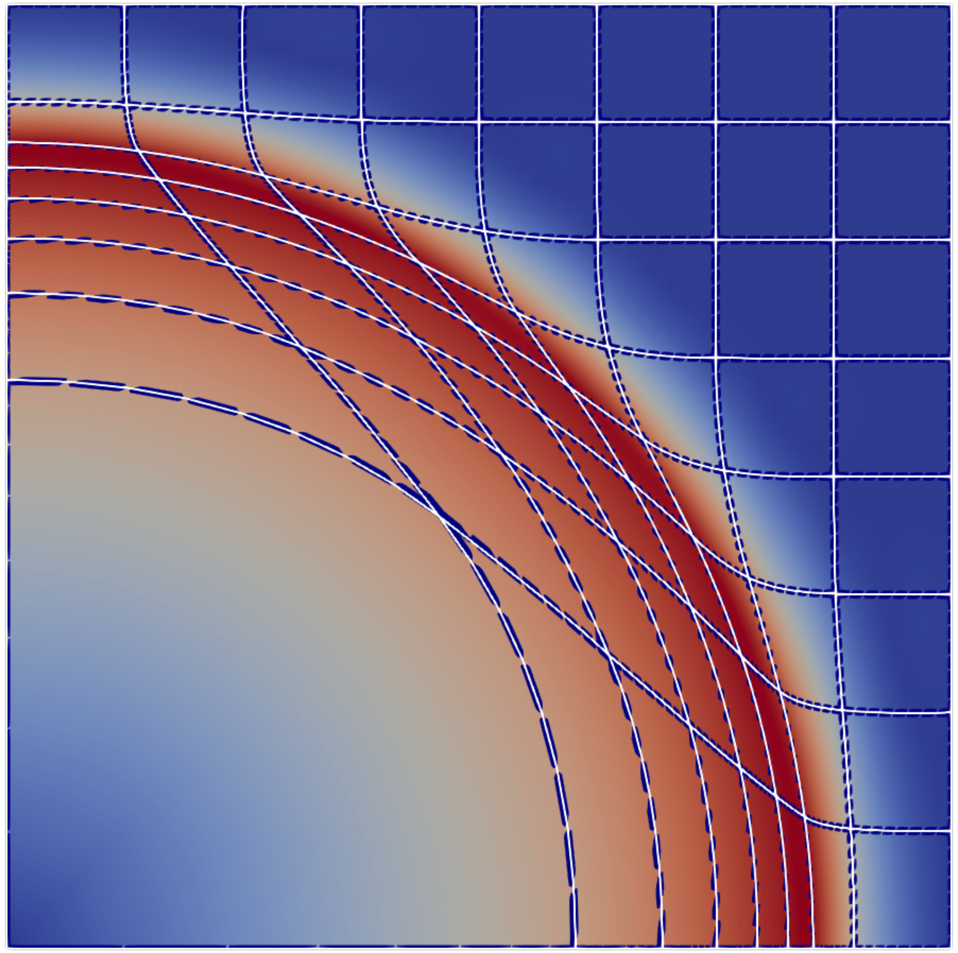


dFEM Implicit Midpoint CFL 5.0

step 101, t = 0.8, dt = 0.00231807
Energy difference (initial vs final): 5.56e-05



Comparison of ALE movement from dFEM implicit hydro vs Laghos explicit time integration

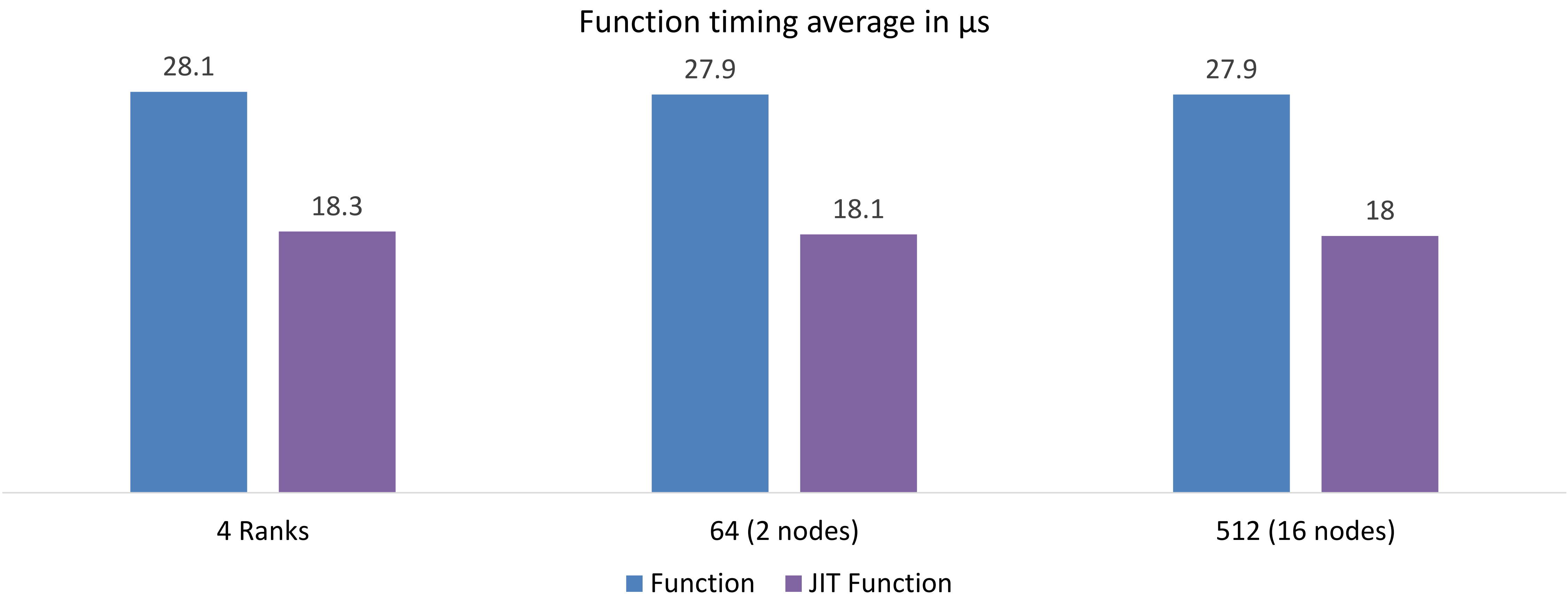


WIP

- Automatic Element Matrix, Processor local Sparse Matrix and global HypreParMatrix assemble
- Automatic Partial Assembly algorithms
  - Takes advantage of derivative knowledge and caches the quadrature point Jacobians as PA data
  - Automatically applies PA data to function
  - Enables no-overhead linearized operator transpose action (👋 Adjoint method)
- JIT (experimental)

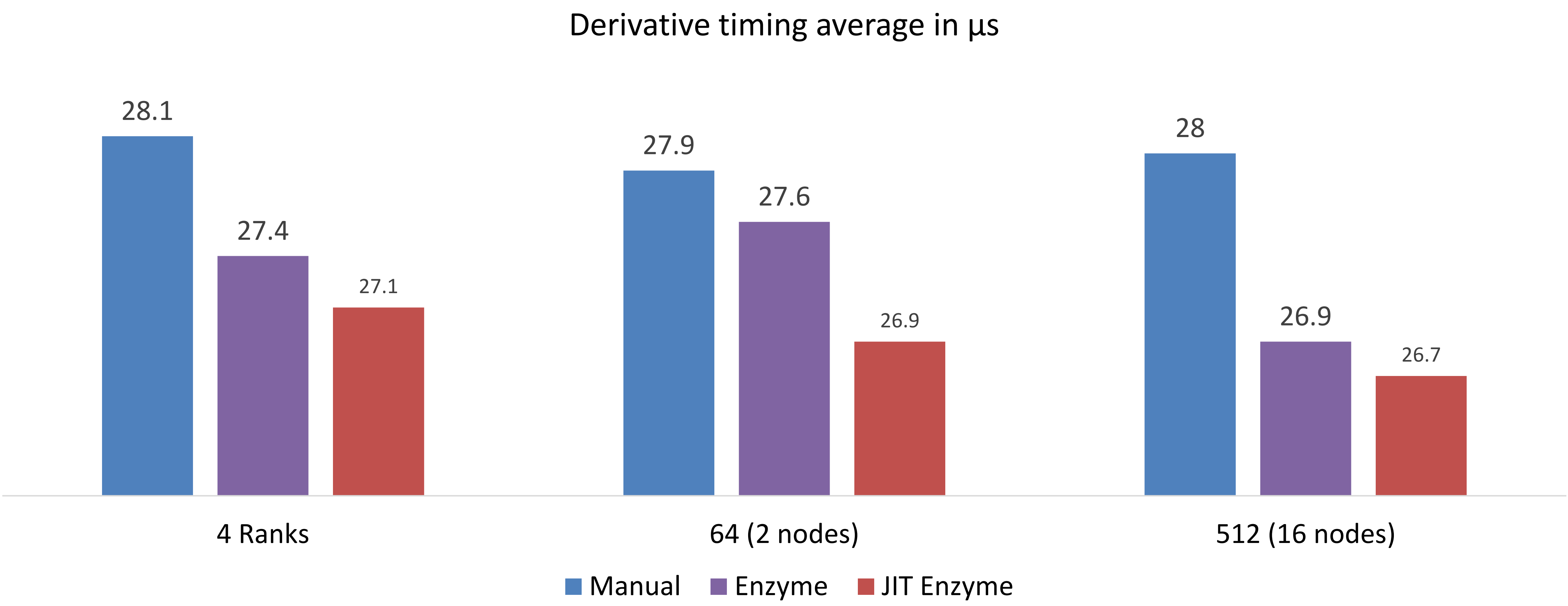
Performance. Collab with Giorgis Georgakoudis

- GPU kernel launch latency is 5μs



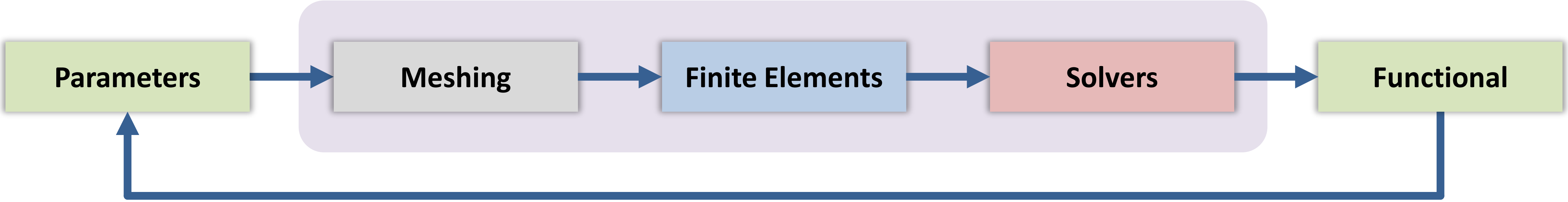
Performance. Collab with Giorgis Georgakoudis

- GPU kernel launch latency is 5µs



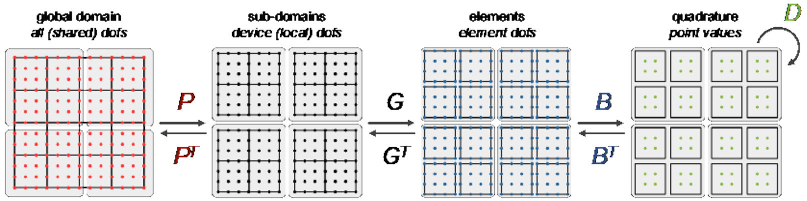
Automatic Differentiation overview in MFEM

Jacobians and derivatives of FEM operators in a user-friendly way



- FEM decomposition

$$A = P^T G^T B^T D B G P$$



- Parameters $\hat{\rho} = B_\rho G_\rho P_\rho \, \rho$
- Parametric nonlinear operator

$$A(u;\rho) = P^T G^T B^T \, D(\hat{u},\hat{\rho})$$

- Need to differentiate at Q-points only!

$$\nabla_u A(u;\rho) = P^T G^T B^T \, \nabla_{\hat{u}} D(\hat{u},\hat{\rho})$$

(Jacobian is FEM decomposed linear operator)

- Differentiate the Q-function *D* with Enzyme!
  - Can mix code from different languages
  - Differentiate across function calls (e.g., EOS)
  - Many parallel small ADs instead of 1 big one
  - Differentiate only what is necessary

MFEM + Enzyme

CASC

Center for Applied Scientific Computing

Lawrence Livermore National Laboratory