A Convergence Study and Contributions to libCEED

CS 5636: Numerical Solutions to PDEs Final Project Presentation
Dec 7 2022

Leila Ghaffari, Amanda Shackelford, Joey Clement

libCEED

- Open source numerical software library for finite element methods
- Sparse matrices are expensive for high order methods
  - New format needed to represent operators
- Goals of libCEED:
  - Implement a matrix free representation
    - Note: libCEED supports matrix methods for low order solvers
  - Enable efficient computation on different device types/architectures

CEED
EXASCALE DISCRETIZATIONS

ECP
EXASCALE COMPUTING PROJECT

[1] https://github.com/CEED/libCEED

Navier Stokes Equations for Compressible Flow

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \boldsymbol{u} = 0$$ Mass

$$\frac{\partial (\rho \boldsymbol{u})}{\partial t} + \nabla \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u} + P \boldsymbol{I}_3 - \boldsymbol{\sigma}) + \rho g \hat{\boldsymbol{k}} = 0$$ Momentum

$$\frac{\partial E}{\partial t} + \nabla \cdot ((E + P)\boldsymbol{u} - \boldsymbol{u} \cdot \boldsymbol{\sigma} - k \nabla T) = 0$$ Total Energy

Conservative Variables

$$\frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{U}) - S(\boldsymbol{U}) = 0$$

$$\boldsymbol{U} = \rho, \rho \boldsymbol{u}, E$$

Primitive Variables

$$\frac{\partial}{\partial t} \boldsymbol{U}(\boldsymbol{Y}) + \nabla \cdot \boldsymbol{F}(\boldsymbol{Y}) - S(\boldsymbol{Y}) = 0$$

$$\boldsymbol{Y} = P, \boldsymbol{u}, T$$

- Primitives often easier to work with/more intuitive
- Accuracy determined by application
- Primitives are measurable

Navier Stokes Miniapp

Problems:

- Advection
  - Rotation
  - Translation
- Euler
  - Isentropic Vortex
  - Shock Tube
- Navier-Stokes
  - Density Current
  - Channel
  - Blasius Boundary Layer
- Synthetic Turbulence Generation

Features:

- Stabilization Techniques
  - SU (streamline-upwind)
  - SUPG (streamline-upwind/Petrov-Galerkin)
- Unstructured Mesh
- Conservative and Primitive variables
- Implicit and explicit time integration

Blasius Problem Refresher

- Two dimensional flow layer over a semi-infinite flat plate
- Similarity transform essentially reduces to 1-D problem in distance from plate
  - $0 < \eta < 1$ describes the entire boundary layer at any x-coordinate
    - Self similar coordinate
- Certain terms in NS equations negligible
- Problem described by:

$$2(\rho\mu f'')' + ff'' = 0$$

$$(\rho\mu h')' + \mathrm{Pr} f h' + \mathrm{Pr}(\gamma - 1)\mathrm{Ma}^2 \rho\mu (f'')^2 = 0$$

f: Similarity form for the stream function

h: Specific enthalpy at the wall

γ = Specific heat ratio

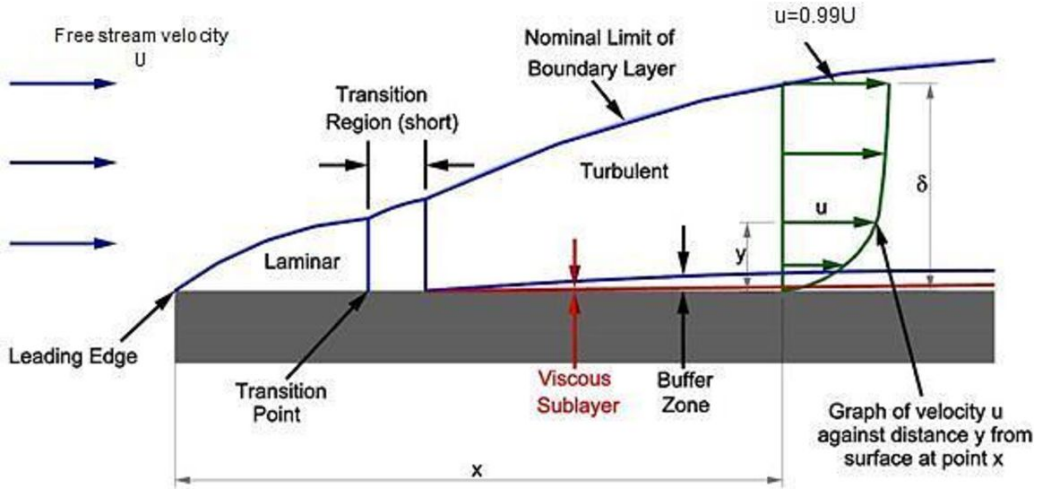
Ma = Mach number

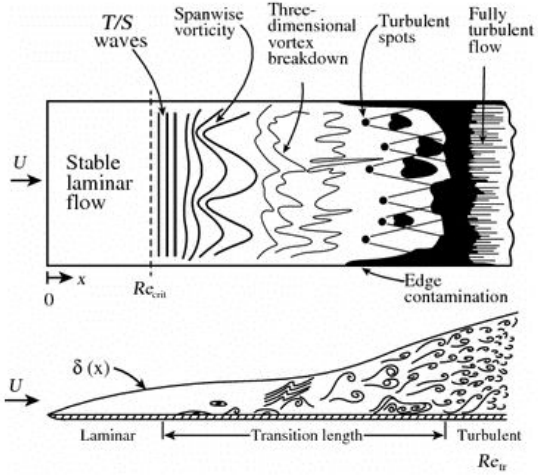
Pr = Prandtl number

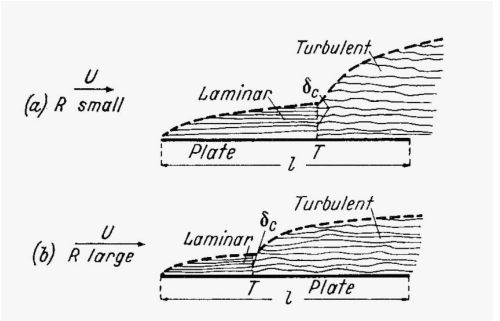
Blasius Boundary Layers

Boundary layers form as a fluid moves over a flat plate:

- At the leading edge of the plate, there is a laminar region. As the flow continues back from the leading edge, the laminar boundary layer increases in thickness.
- A Turbulent layer is eventually reached and a viscous sublayer is formed.
- Many things affect the length of these regions both along and away from the plate







Mesh Generation

Goal: Come up with a continuous transformation that has all of the information you would like to specify to generate the mesh.

What we would like to be able to specify:

- Aspect ratio at the first boundary between the viscous region and the turbulent region
- Height of the two layers and total height of the domain
- Percentage of total height to the second boundary layer (in decimal form)

Problem Formulation

$$h(y)=\begin{cases} b_1 e^{k_1 y} & 0 \leq y \leq \sigma_1 \\ b_2 e^{k_2 y} & \sigma_1 \leq y \leq \sigma_2 \\ b_3 e^{k_3 y} & \sigma_2 \leq y \leq 1 \end{cases}$$

$$\tilde{y}(y') = \int_0^{y'} h(y')\, dy'$$

Boundary Conditions

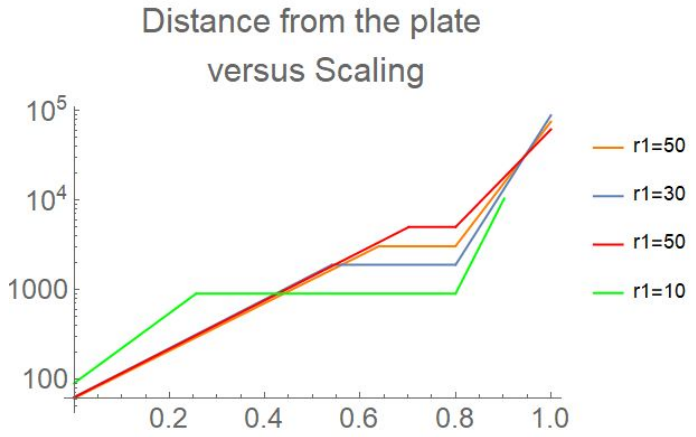
h(y) is continuous

$\tilde{y}(\sigma_1) = \delta_1$

$\tilde{y}(\sigma_2) = \delta_2$

$\tilde{y}(1) = \tilde{y}_{max}$

$\frac{h(\sigma_1)}{h(0)} = r1$



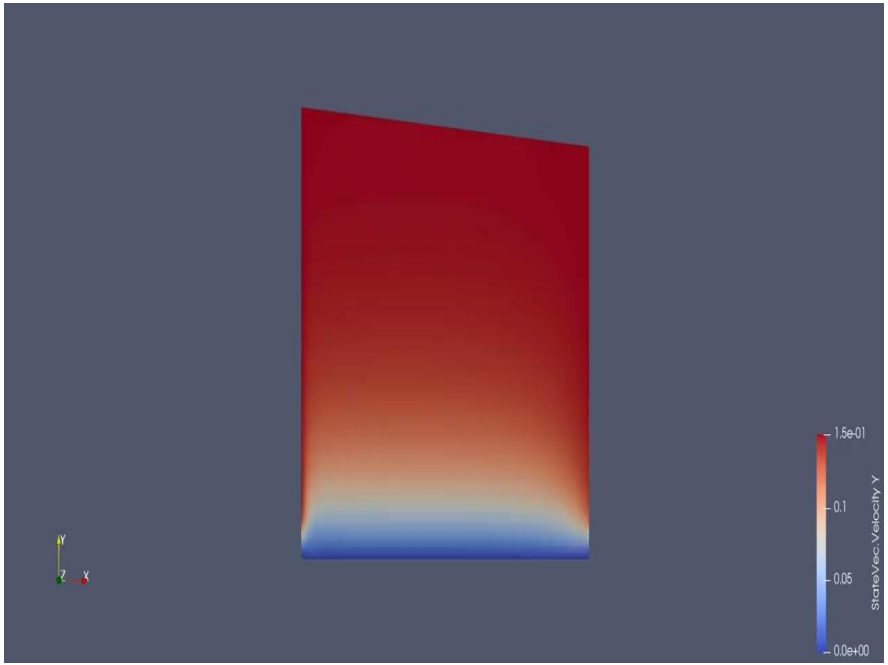
Vary the aspect ratio, set y_1=10, y_2=500, y_{max}=5000, σ_2=0.8

Contributions to libCEED

- Output errors from NS solver in both primitive and conservative variables
  - Previously only 2D Advection and Euler solvers computed errors
  - NS solver now computes errors for exactly solvable problems
    - Errors computed for one set of variables (primitive or conservative)
    - In post processing, converts errors into other set, report both
- Update mesh generation to make user input more friendly
  - Previously the mesh was generated discretely from changing the setting in the yaml file (for changing from linear to quadratic) and specifying the number of nodes
  - Now the mesh will be generated from a continuous function so can easily convert before linear and quadratic mesh
    - Functional form provides a more intuitive input for users who know what boundary layers they want to resolve
- Convergence study on Blasius boundary layer problem

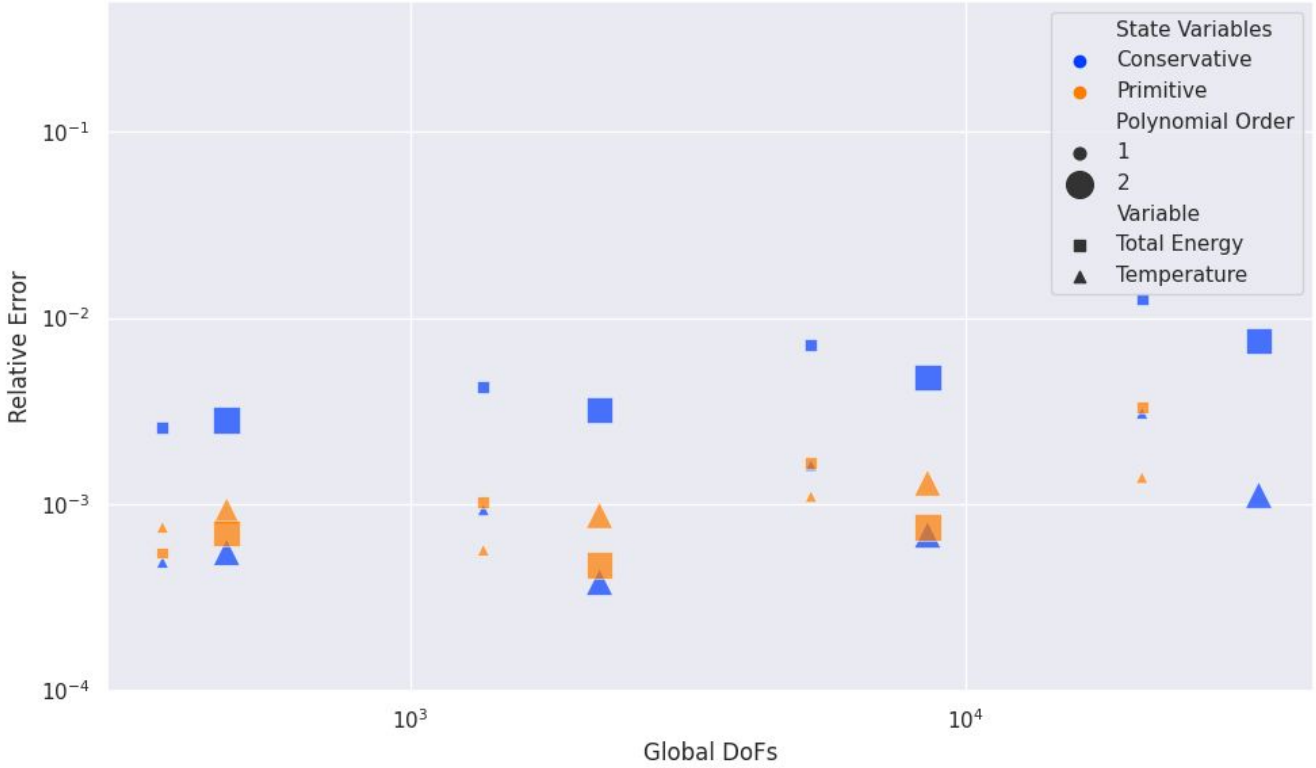
Blasius Problem With Primitive And Conservative Variables

Problem run in primitive variables



Problem run in conservative variables

Convergence (?) study



Current State

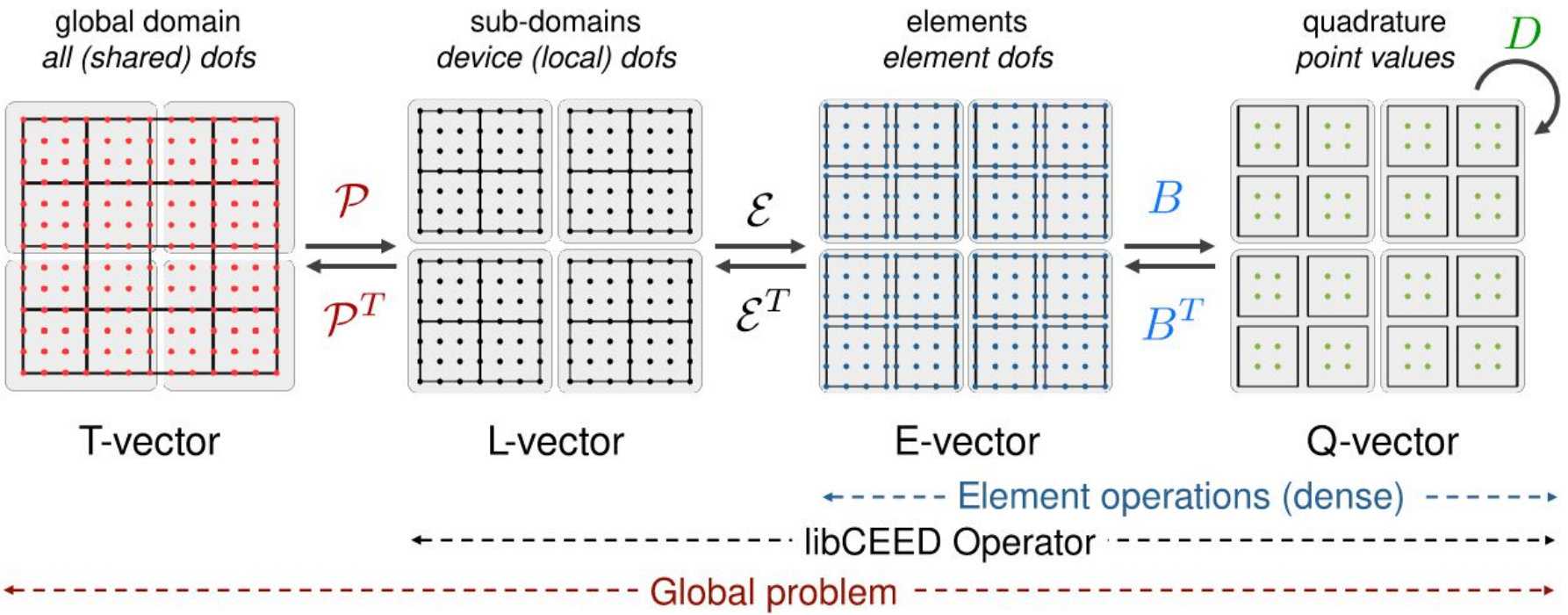
Open PRs:

- Automate stabilization coefficients
  https://github.com/CEED/libCEED/pull/1096
- Compute component-wise errors in primitive and conservative variable sets
  https://github.com/CEED/libCEED/pull/1109
- Automate mesh generation for the boundary layer
  https://github.com/CEED/libCEED/pull/1111

Outlook:

- Investigate the issue with the convergence plot
- Incorporate the entropy variables implemented by Zach and Drishan
- Perform similar study for other ICs and BCs

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



[2] https://libceed.org/en/latest/libCEEDapi/#finite-element-operator-decomposition