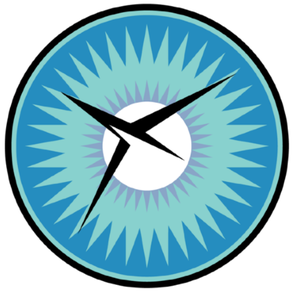
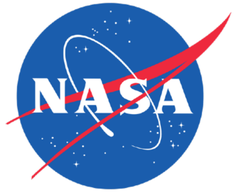


Towards Robust and Accurate Implicit Gradient Methods for Second- and Third-Order Nodal-Gradient Cell-Centered Finite-Volume Discretizations on Tetrahedral Grids

Hiroaki Nishikawa, *National Institute of Aerospace*

11:10am, in Academic Ballroom 412, AIAA Aviation,
Tuesday, July 30, 2024



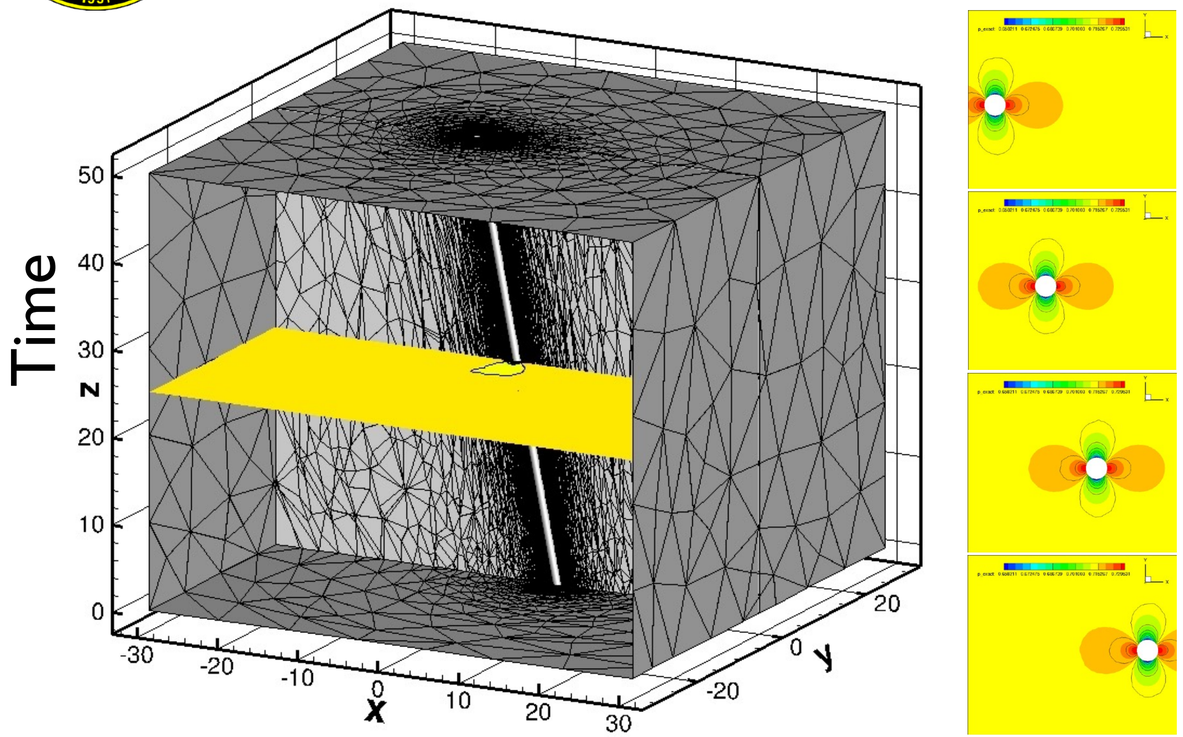
This work was supported partly by the Hypersonic Technology Project, through the Hypersonic Airbreathing Propulsion Branch of the NASA Langley Research Center, under Prime Contract No. 80LARC23DA003

Economical 3rd-Order Methods

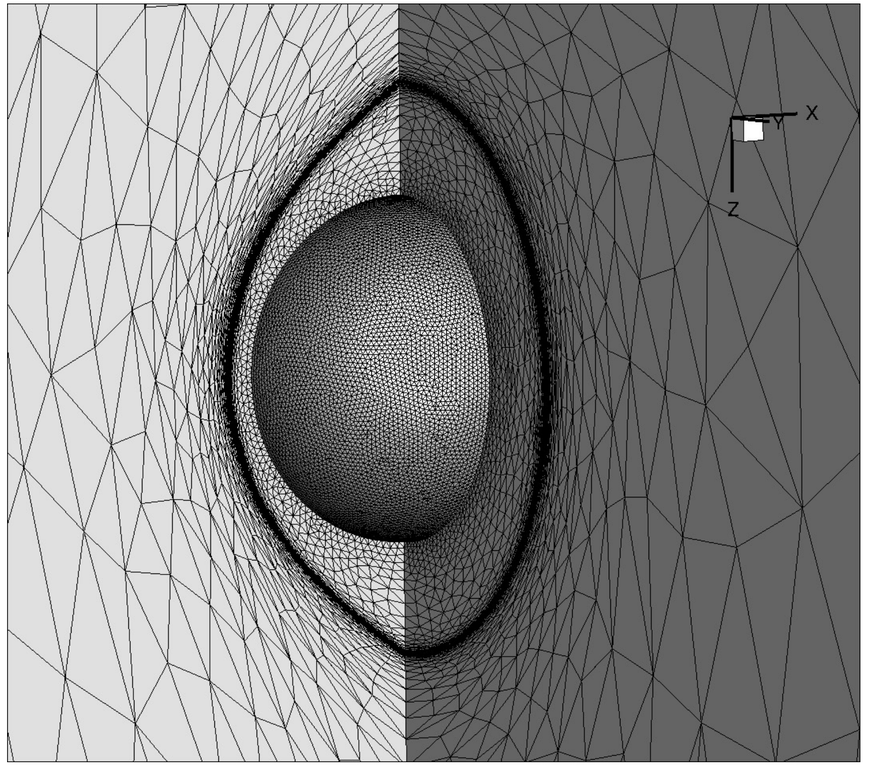
Economical 3rd-Order Methods

NATIONAL INSTITUTE OF AEROSPACE

Exploring Space-Time Hyperbolic NS



Improving Unstructured-Grid Methods



Common interest: *economical 3rd-order methods* with a single flux per face/edge, NOT requiring 2nd derivatives at all, towards automated CFD with fully adaptive tetrahedral grids.

Economical 3rd-Order NGCC Discretization

NATIONAL INSTITUTE OF AEROSPACE

This paper focuses on a 3rd-order cell-centered method:

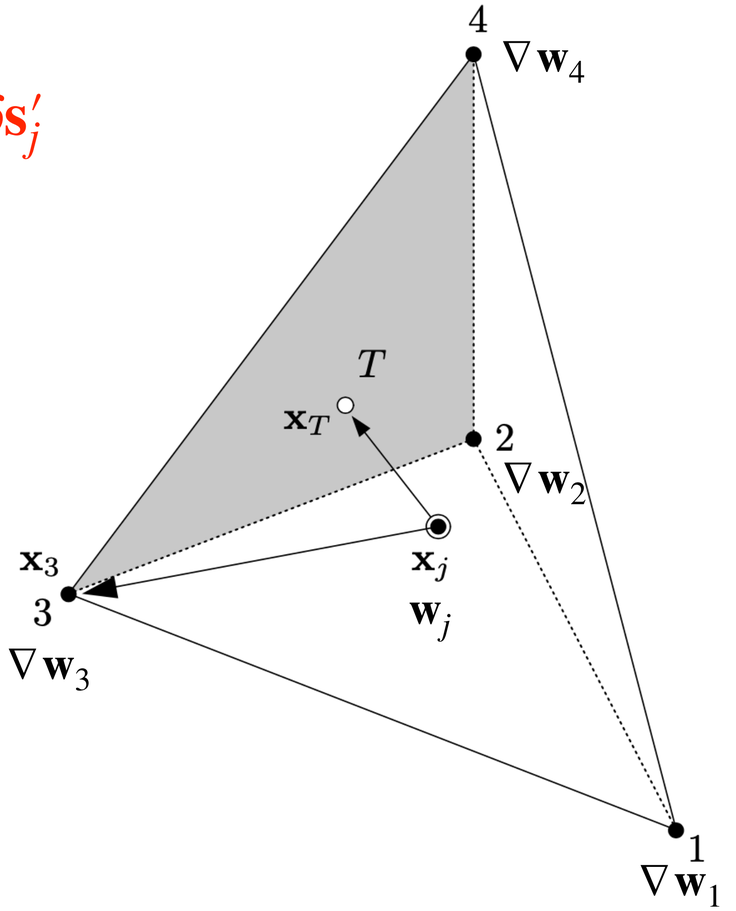
$$V_j \frac{\partial \mathbf{u}_j}{\partial t} + \sum_{k \in \{k_j\}} \Phi_{jk}(\mathbf{w}_L, \mathbf{w}_R) \, |\mathbf{n}_T| - \mathbf{s}_j V_j = \delta \mathbf{f}_j + \delta \mathbf{s}'_j$$

where $\mathbf{u}_j$ is a vector of point-valued solutions at cell center, and

Quadratic interpolation:
$$\mathbf{w}_L = \mathbf{w}_j + \frac{1}{2}\left(\overline{\nabla \mathbf{w}}_T + \overline{\nabla \mathbf{w}}_j\right)(\mathbf{x}_T - \mathbf{x}_j)$$

Flux correction:
$$\delta \mathbf{f}_j = \frac{1}{24} \sum_{k \in \{k_j\}} \sum_{i=1}^{3} \left(\frac{\partial \mathbf{f}}{\partial \mathbf{w}}\right) \nabla \mathbf{w}_i (\mathbf{x}_i - \mathbf{x}_T) \, |\mathbf{n}_T|$$

Source correction:
$$\delta \mathbf{s}'_j = \frac{1}{40} \sum_{i=1}^{4} \nabla \mathbf{s}'_i (\mathbf{x}_i - \mathbf{x}_j) V_j \qquad \mathbf{s}' = \mathbf{s}_j - \frac{\partial \mathbf{u}}{\partial t}$$



We only need gradients at nodes for 3rd-order accuracy: no second derivatives required.

Explicit and Implicit Gradient Methods $\mathbf{g} = \left[\partial_x u, \partial_y u, \partial_z u\right]$

NATIONAL INSTITUTE OF AEROSPACE

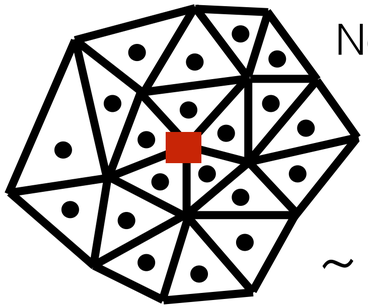
Explicit:

Linear LSQ

$$\mathbf{g}_j = \sum_{k \in nghbrs} C_{jk} u_k$$

Quadratic LSQ

$$\mathbf{g}_j = \sum_{k \in nghbrs} C_{jk} u_k$$



Node-nghbr aug

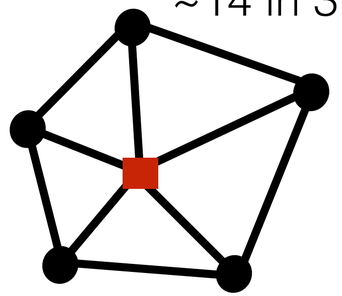
~ 200 in 3D

Large stencil: a solver gets more robust but less accurate.

Haider&Croisille&Courbet NM2009

Implicit:

$$\mathbf{M}_{jj}\mathbf{g}_j + \sum_{k \in \{k_j\}} \mathbf{M}_{jk}\mathbf{g}_k = RHS$$



~14 in 3D

Galerkin: Löhner(1994)

Variational Reconstruction: Wang et. al., JCP2017

Implicit GG: Nishikawa, JCP2019

Implicit EB gradient: Nishikawa, AIAA2020

Advantages of Implicit Gradients

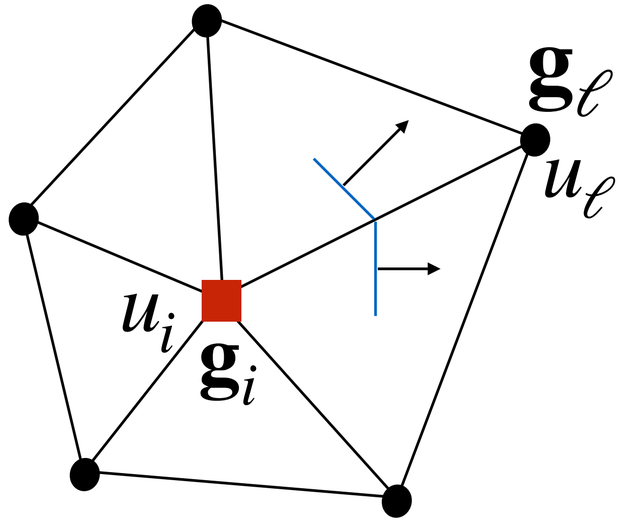
- Flow solver is more stable with a huge gradient stencil.
- Superior gradient accuracy.
- Per-relaxation cost can be lower than LSQ gradients with hundred of neighbors

Implicit Gradients

IEBG2 (AIAA2024) $\mathbf{g} = \left[\partial_x u, \partial_y u, \partial_z u\right]$

IEBG2 system: Exact for quadratic functions

$$\mathbf{M}_{ii}\mathbf{g}_i + \sum_{\ell \in \{\ell_i\}} \mathbf{M}_{i\ell}\mathbf{g}_\ell = \frac{1}{V_j} \sum_{\ell \in \{\ell_i\}} \frac{u_i + u_\ell}{2}\mathbf{n}_{i\ell} \qquad \mathbf{g}_i = \begin{bmatrix} \partial_x u_i \\ \partial_y u_i \\ \partial_z u_i \end{bmatrix}$$



We can solve the system by a relaxation scheme:

$$\mathbf{g}_i^{m+1} = \mathbf{g}_i^m - \mathbf{M}_{ii}^{-1}\left(\sum_{\ell \in \{\ell_i\}} \mathbf{M}_{i\ell}\mathbf{g}_\ell^m + \frac{1}{V_j} \sum_{\ell \in \{\ell_i\}} \frac{u_i + u_\ell}{2}\mathbf{n}_{i\ell} \right)$$

Successfully demonstrated for their 3rd-order EB discretization method.

IEBG2 is very efficient and perfect for the 3rd-order EB because it does not require second derivatives.

Can we develop something similar to our 3rd-order CCFV method?

Implicit Nodal-Gradient Cell-Solution Method

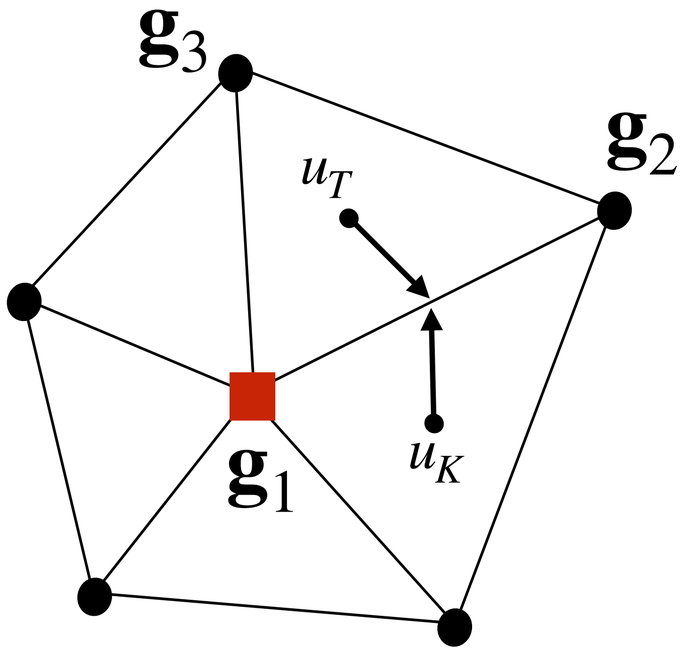
Yes, we can. First, define the cell discretization:

$$\mathbf{g} = \nabla u \longrightarrow \int_{V_T} \mathbf{g}\, dV = \oint_{\partial V_T} u\, \hat{\mathbf{n}} ds$$

3rd-order NG-CCFV

No 2nd derivatives

$$\longrightarrow \mathbf{Res}_T = \frac{\mathbf{g}_1 + \mathbf{g}_2 + \mathbf{g}_3}{3} - \sum_{K \in \{K_T\}} \left[\phi_{TK} + \delta\phi_{TK}\right] |\mathbf{n}_{TK}|$$



Then, minimizing the cell residuals in the LSQ sense:

$$\mathscr{F} = \frac{1}{2} \sum_{\text{all cells}} \mathbf{Res}_T^2 \longrightarrow \frac{\partial \mathscr{F}}{\partial \mathbf{g}_i} \equiv \mathbf{Res}_i = \sum_{T \in \{T_i\}} \left(\frac{\partial \mathbf{Res}_T}{\partial \mathbf{g}_i} \right)^t \mathbf{Res}_T = \mathbf{0}$$

The resulting system is second-derivative-free as desired, but not robust…

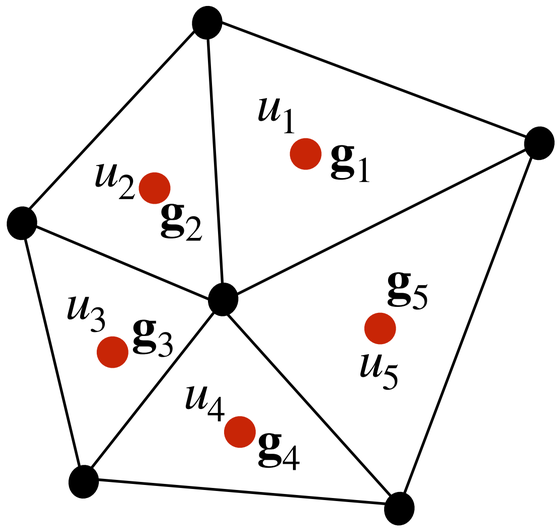
Worked very hard to make it robust, but couldn't get it done successfully…

IEBG2 for NG-CCFV Method

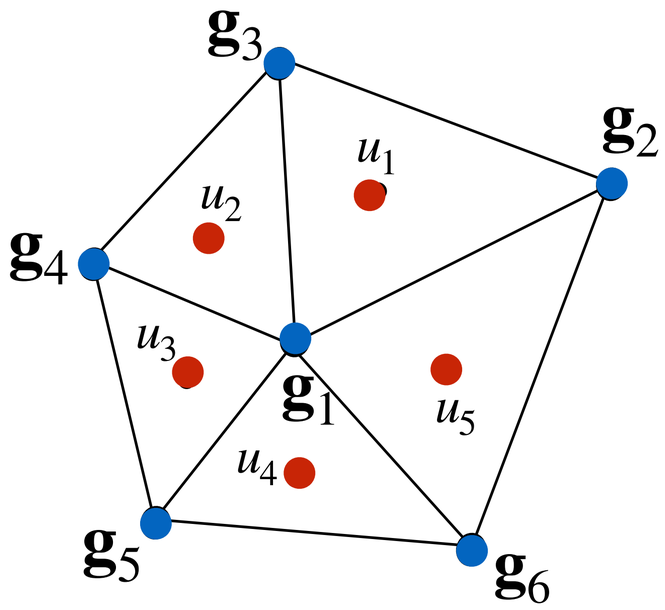
National Institute of Aerospace

Realize that NG-CCFV is somewhere between CCFV and NCEB:

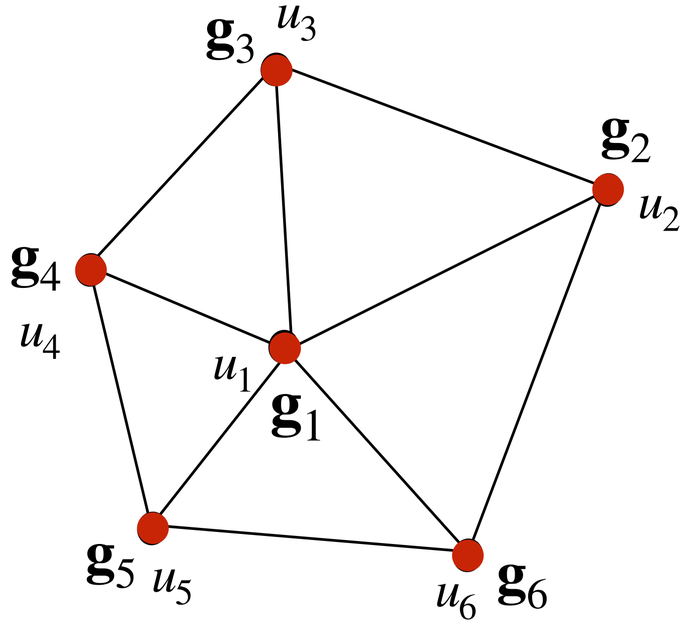
Cell-Centered



Nodal-Grad Cell-Centered



Node-Centered



$$\mathbf{M}_{ii}\mathbf{g}_i + \sum_{\ell \in \{\ell_i\}} \mathbf{M}_{i\ell}\mathbf{g}_\ell = \frac{1}{V_j} \sum_{\ell \in \{\ell_i\}} \frac{u_i + u_\ell}{2} \mathbf{n}_{i\ell}$$

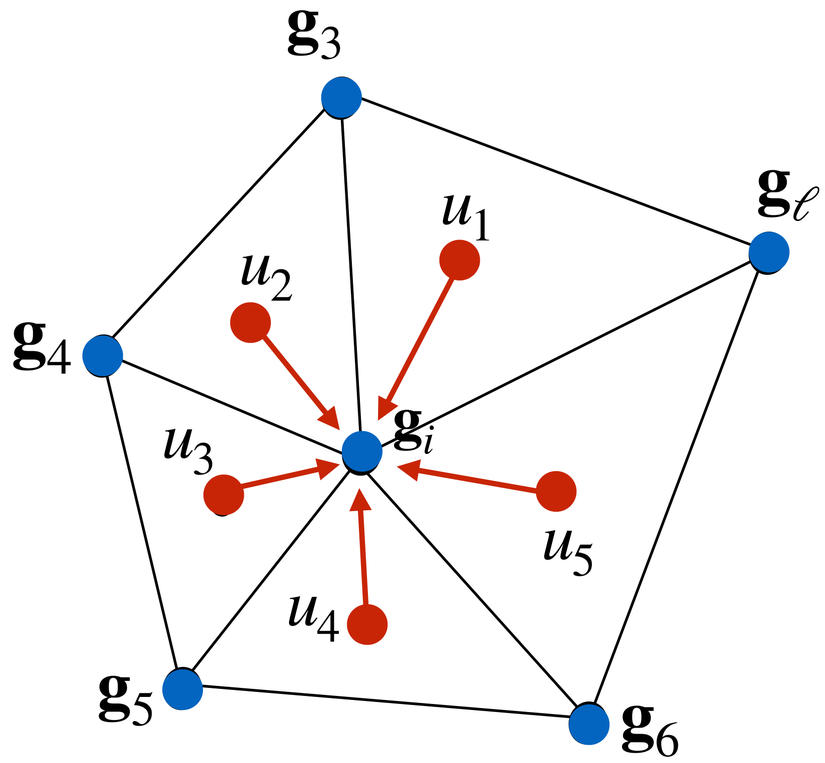
IEBG2 can be directly employed if the solution values can be transferred from cells to nodes.

IEBG2 for NG-CCFV Method

Just interpolate the solution from cells to nodes:

$$u_i = \frac{1}{N_{c_i}} \sum_{j \in \{c_i\}} \left[u_j + \left\{ C\mathbf{g}_i + (1-C)\overline{\mathbf{g}}_j \right\} \cdot (\mathbf{x}_i - \mathbf{x}_j) \right]$$

where C=1/2 for quadratic interpolation, otherwise linear.



Then, IEBG2 can be directly employed:

$$\mathbf{M}_{ii}\mathbf{g}_i + \sum_{\ell \in \{\ell_i\}} \mathbf{M}_{i\ell}\mathbf{g}_\ell = \frac{1}{V_j} \sum_{\ell \in \{\ell_i\}} \frac{u_i + u_\ell}{2} \mathbf{n}_{i\ell}$$

Right hand side depends on the nodal gradients —> Defect correction iteration.

Defect-Correction Iteration

National Institute of Aerospace

Define the residual:

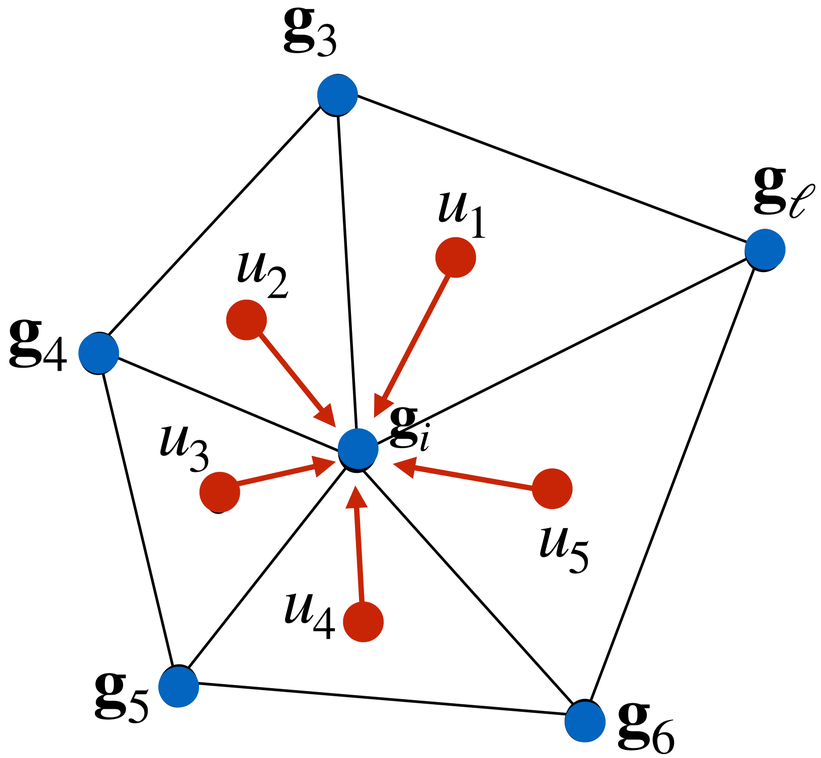
$$\mathbf{r}_i = \mathbf{M}_{ii}\mathbf{g}_i + \sum_{\ell\in\{\ell_i\}} \mathbf{M}_{i\ell}\mathbf{g}_\ell - \frac{1}{V_j}\sum_{\ell\in\{\ell_i\}} \frac{u_i + u_\ell}{2}\mathbf{n}_{i\ell} \qquad u_i = \frac{1}{N_{c_i}}\sum_{j\in\{c_i\}}\left[u_j + \left\{C\mathbf{g}_i + (1-C)\overline{\mathbf{g}}_j\right\}\cdot(\mathbf{x}_i - \mathbf{x}_j)\right]$$

Then, we can solve it by the defect-correction iteration:

$$\mathbf{g}_i^{n+1} = \mathbf{g}_i^n + \Delta\mathbf{g}_i$$

$$\Delta\mathbf{g}_i^{m+1} = \Delta\mathbf{g}_i^m - \mathbf{M}_{ii}^{-1}\left(\sum_{\ell\in\{\ell_i\}} \mathbf{M}_{i\ell}\Delta\mathbf{g}_\ell^m + \mathbf{r}_i\right) \qquad m = 0,1,2,3,4$$

5 preconditioner relaxations (Gauss-Seidel)



This algorithm can be easily implemented if IEBG is already available.

Implicit Solver for Euler

National Institute of Aerospace

Starting from the initial values:

$$\mathbf{g}_i^0 = (0,0,0), \quad i = 1,2,3,\cdots,N_{nodes} \qquad \mathbf{w}_j^0 = (\rho_\infty, u_\infty, v_\infty, w_\infty, p_\infty,), \quad j = 1,2,3,\cdots,N_{cells}$$

Then, iterate for $n = 0,1,2,3,\cdots$, until convergence,

Update gradients at all nodes i=1,2,3,…, for all variables: $\mathbf{g} = (\nabla\rho, \nabla u, \nabla v, \nabla w, \nabla p)$

$$\mathbf{g}_i^{n+1} = \mathbf{g}_i^n + \Delta\mathbf{g}_i \qquad \Delta\mathbf{g}_i^{m+1} = \Delta\mathbf{g}_i^m - \mathbf{M}_{ii}^{-1}\left(\sum_{\ell\in\{\ell_i\}} \mathbf{M}_{i\ell}\Delta\mathbf{g}_\ell^m + \mathbf{r}_i\right) \qquad m = 0,1,2,3,4$$

Update solutions:

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \Delta\mathbf{U} \qquad \left(\mathbf{D} + \frac{\partial\overline{\mathbf{Res}}}{\partial\mathbf{U}}\right)\Delta\mathbf{U} = -\mathbf{Res}(\mathbf{U}^n)$$ *<- Relax by GS*

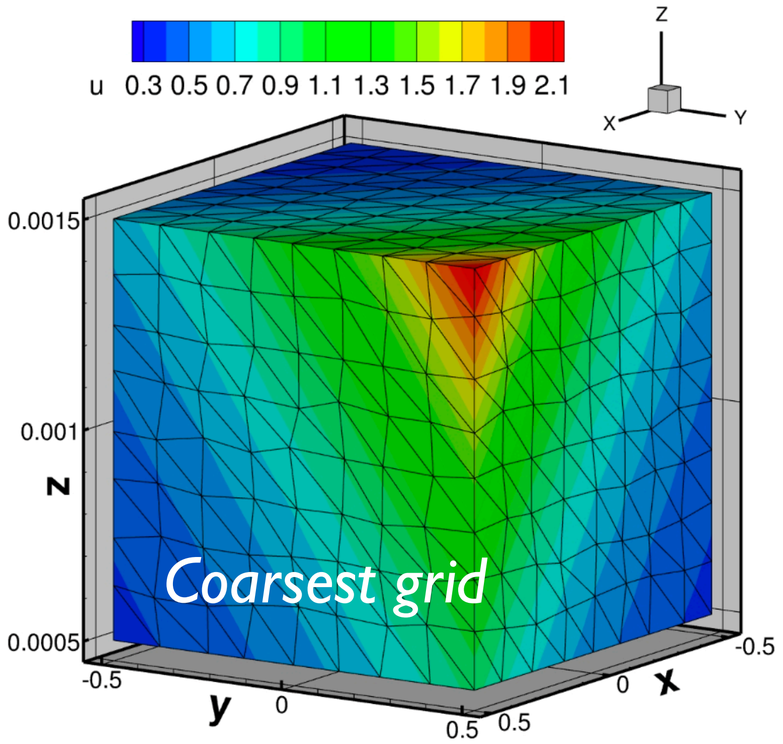
This is a loosely-coupled iteration between gradients and solutions.

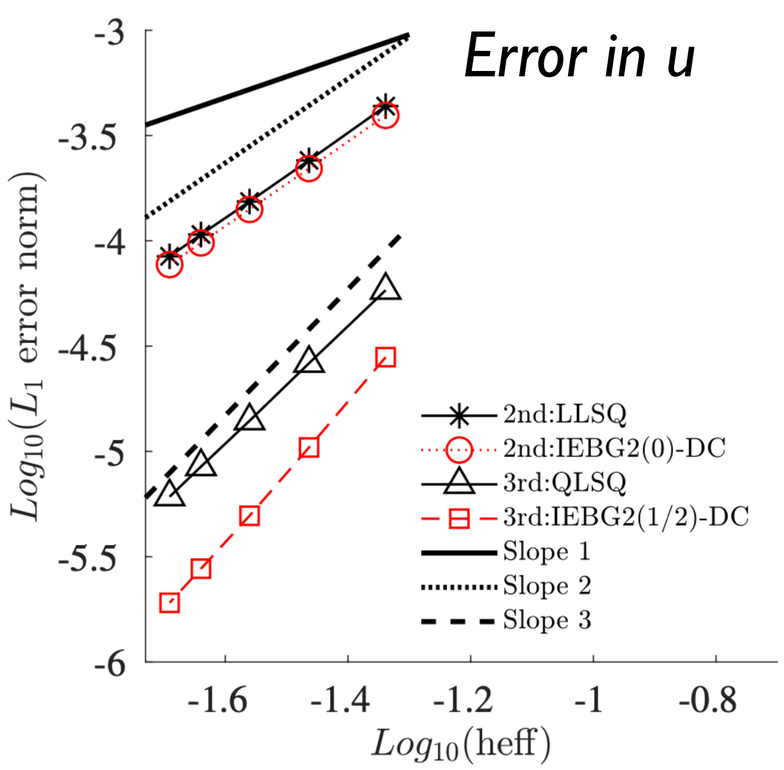
Results

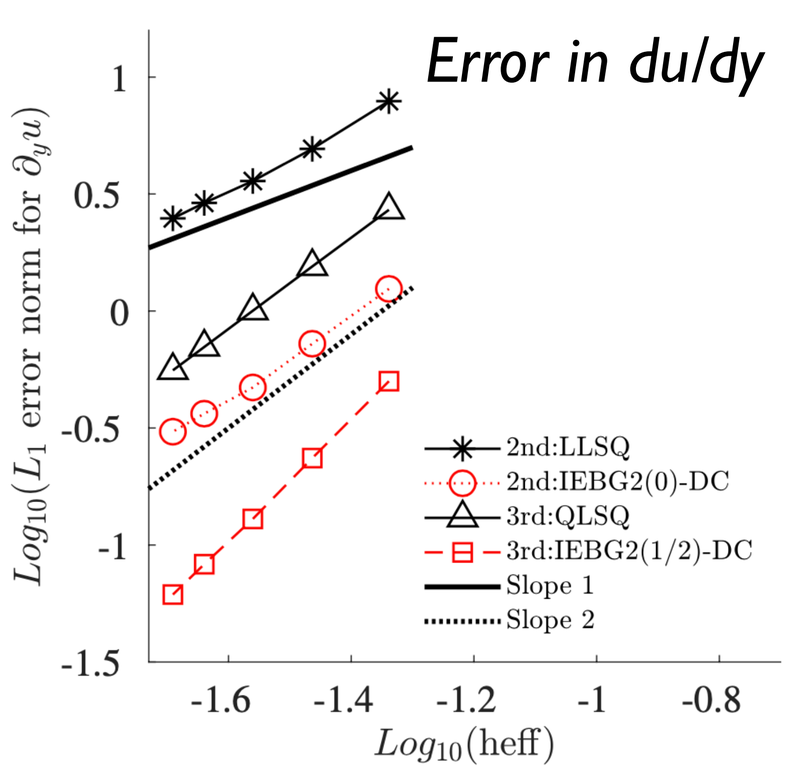
Accuracy Verification

NATIONAL INSTITUTE OF AEROSPACE

Solve Euler by 2nd- and 3rd-order NG-CCFV methods with LSQ and IEBG2 gradients over 5 levels of grids and measure the solution error, using manufactured solutions.







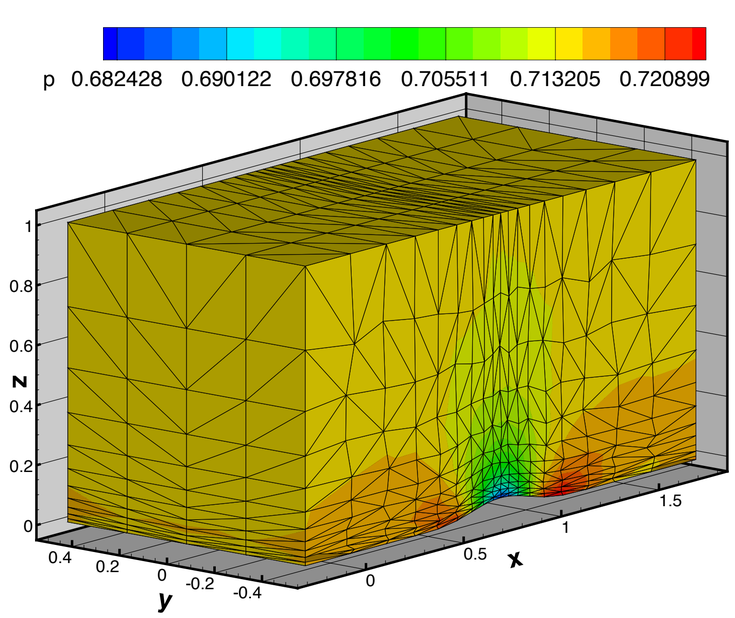
The solver took twice as many iteration with IEBG2. Not expected but…

Mach 0.3 Subsonic Flow over a smooth bump

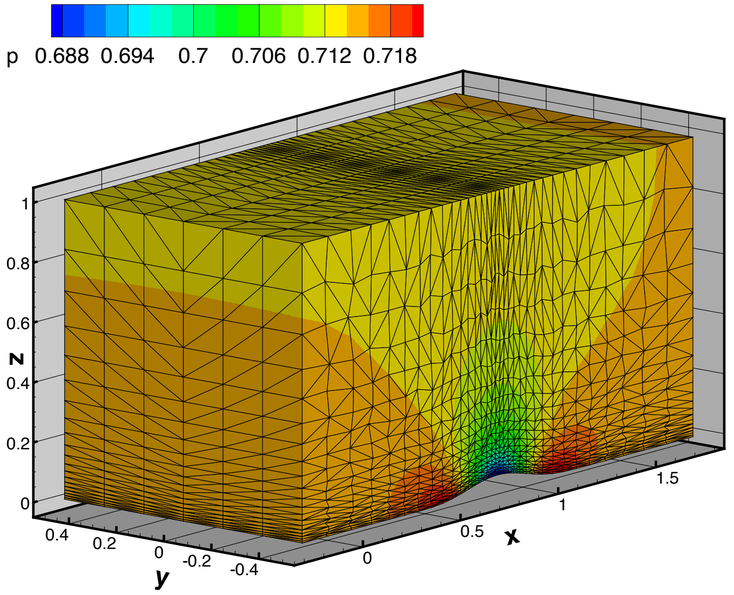
NATIONAL INSTITUTE OF AEROSPACE

Solve Euler by 2nd- and 3rd-order NG-CCFV methods with LSQ and IEBG2 gradients over 5 levels of grids and measure the spurious entropy error.

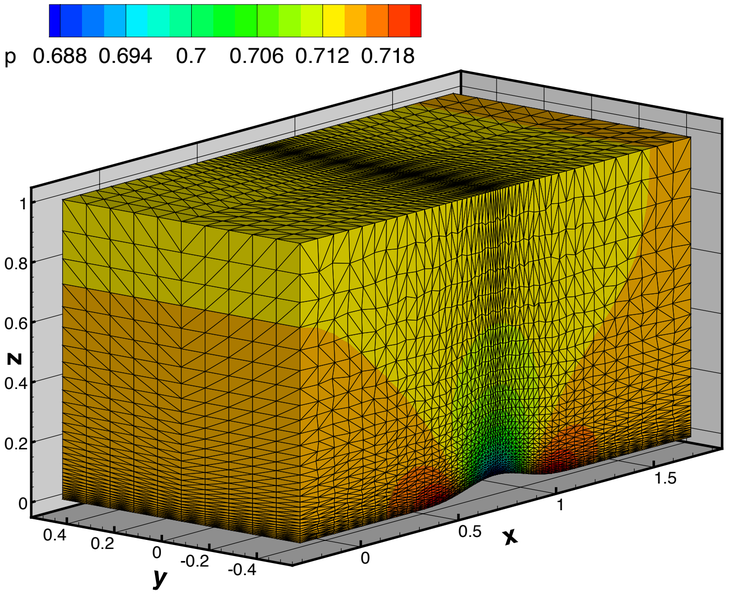
Grid 1



Grid 2



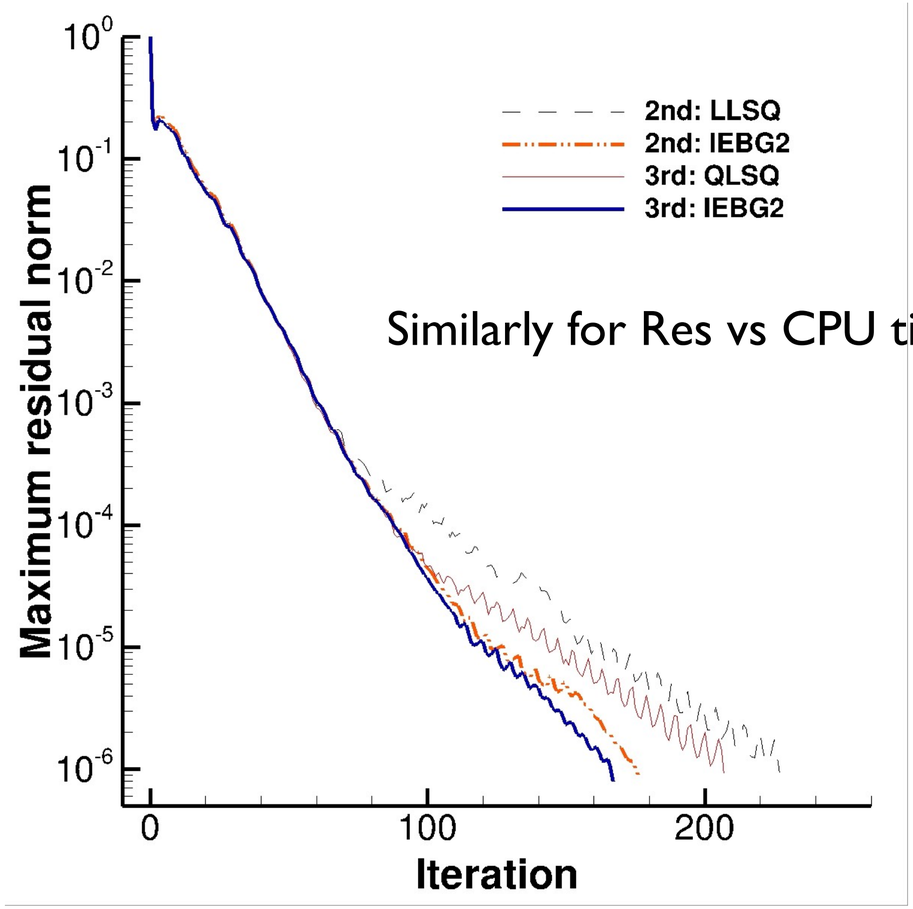
Grid 3



Subsonic Flow: $\kappa = 0$ and $c = 5$

NATIONAL INSTITUTE OF AEROSPACE

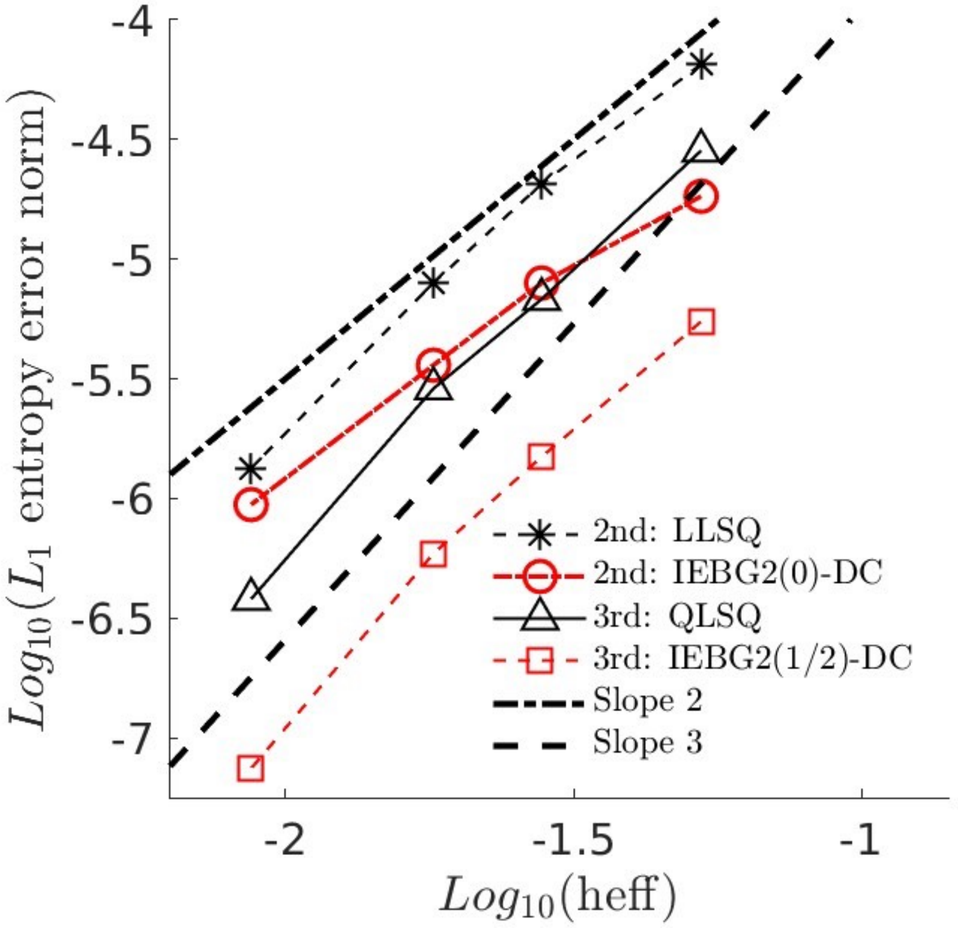
(1) Residual norm vs iteration



Fewer iterations and faster time-to-solution

IEBG methods are not expensive.

(2) Entropy error convergence



2nd- and 3rd-order convergence

Lower 3rd-order error by QIEBG.

Conclusions and Future Work

NATIONAL INSTITUTE OF AEROSPACE

Conclusions:

- Developed simple implicit gradient methods for 2nd/3rd-order NG-CCFV methods.
- 3rd-order NG-CCFV method remains 2nd-derivative-free.
- Significantly lower gradient errors by implicit gradient methods.

Future work:

- Compact-stencil version depending only on solutions at cells around a node.
- JFNK solver with gradients added as additional unknowns.
- Further tests needed for more realistic problems with shocks.