A recovery-assisted DG code for the compressible Navier-Stokes equations

January 6th, 2017
5th International Workshop on High-Order CFD Methods
Kissimmee, Florida

Philip E. Johnson & Eric Johnsen

Scientific Computing and Flow Physics Laboratory
Mechanical Engineering Department
University of Michigan, Ann Arbor

UNIVERSITY OF MICHIGAN

Code Overview

Basic Features:

- **Spatial Discretization:** Discontinuous Galerkin, nodal basis
- **Time Integration:** Explicit Runge-Kutta (4^{th} order and 8^{th} order available)
- **Riemann solver:** Roe, SLAU2[†]
- **Quadrature:** One quadrature point per basis function

Non-Standard Features:

- **Discontinuity Sensor:** Detects shock/contact discontinuities, tags "troubled" elements
- **ICB reconstruction:** compact technique, adjusts Riemann solver arguments
- **Compact Gradient Recovery (CGR):** Mixes Recovery with traditional mixed formulation for viscous terms
- **Shock Capturing:** PDE-based artificial dissipation inspired by C-method[††] of Reisner et al.

† Kitamura & Shima, JCP 2013

†† Reisner et al., JCP 2013

Shock Capturing for Euler Equations

Approach: Store and evolve artificial dissipation coefficients (C^x, C^y) as extra flow variables

$$\left.\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ E\\ C^x\\ C^y\end{bmatrix} = \frac{\partial}{\partial x}\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\\ 0\\ 0\end{bmatrix} + \frac{\partial}{\partial y}\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(E+p)\\ 0\\ 0\end{bmatrix} + \right\}\ \boxed{\text{Euler equations}}$$

$$\underbrace{\frac{\partial}{\partial x}\begin{bmatrix}\kappa^x\rho_{,x}\\ \kappa^x(\rho u)_{,x}\\ \kappa^x(\rho v)_{,x}\\ \kappa^x E_{,x}\\ \mu_S C^x_{,x}\\ 0\end{bmatrix} + \frac{\partial}{\partial y}\begin{bmatrix}\kappa^y\rho_{,y}\\ \kappa^y(\rho u)_{,y}\\ \kappa^y(\rho v)_{,y}\\ \kappa^y E_{,y}\\ 0\\ \mu_s C^y_{,y}\end{bmatrix}}_{\boxed{\text{Artificial dissipation + C-diffusion}}} + \underbrace{\begin{bmatrix}0\\ 0\\ 0\\ 0\\ G^x - \frac{C^x}{\epsilon}\\ G^y - \frac{C^y}{\epsilon}\end{bmatrix}}_{\boxed{\text{Source/Sink term for C variable}}}$$

Shock Capturing for Euler Equations

Approach: Store and evolve artificial dissipation coefficients (C^x, C^y) as extra flow variables

$\kappa^x \sim C^x, \kappa^y \sim C^y$

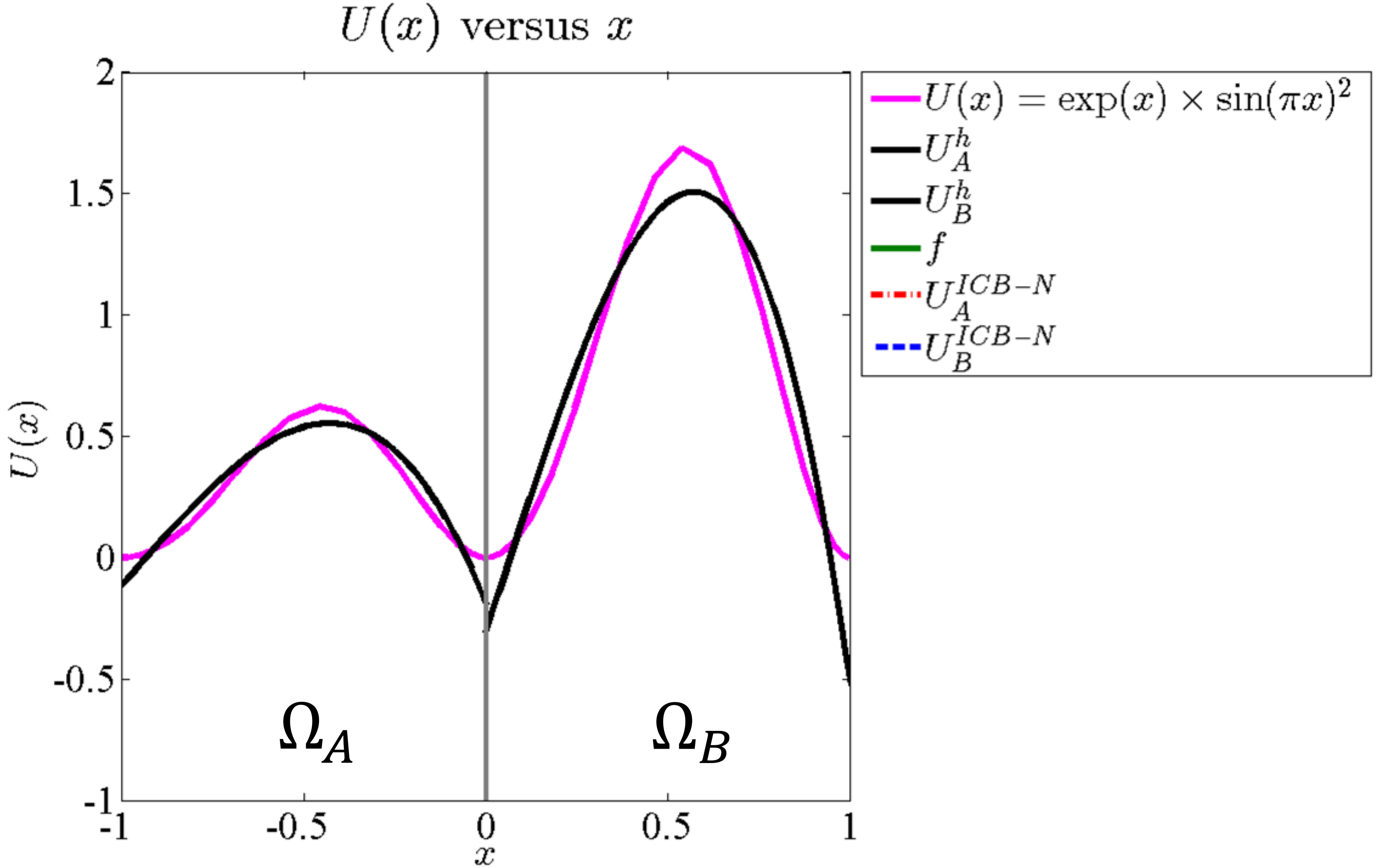
$\epsilon = \frac{h}{p} = \frac{element\ width}{poloynomial\ order}$

$G^x, G^y = 0$ in non-troubled elements (known from sensor)

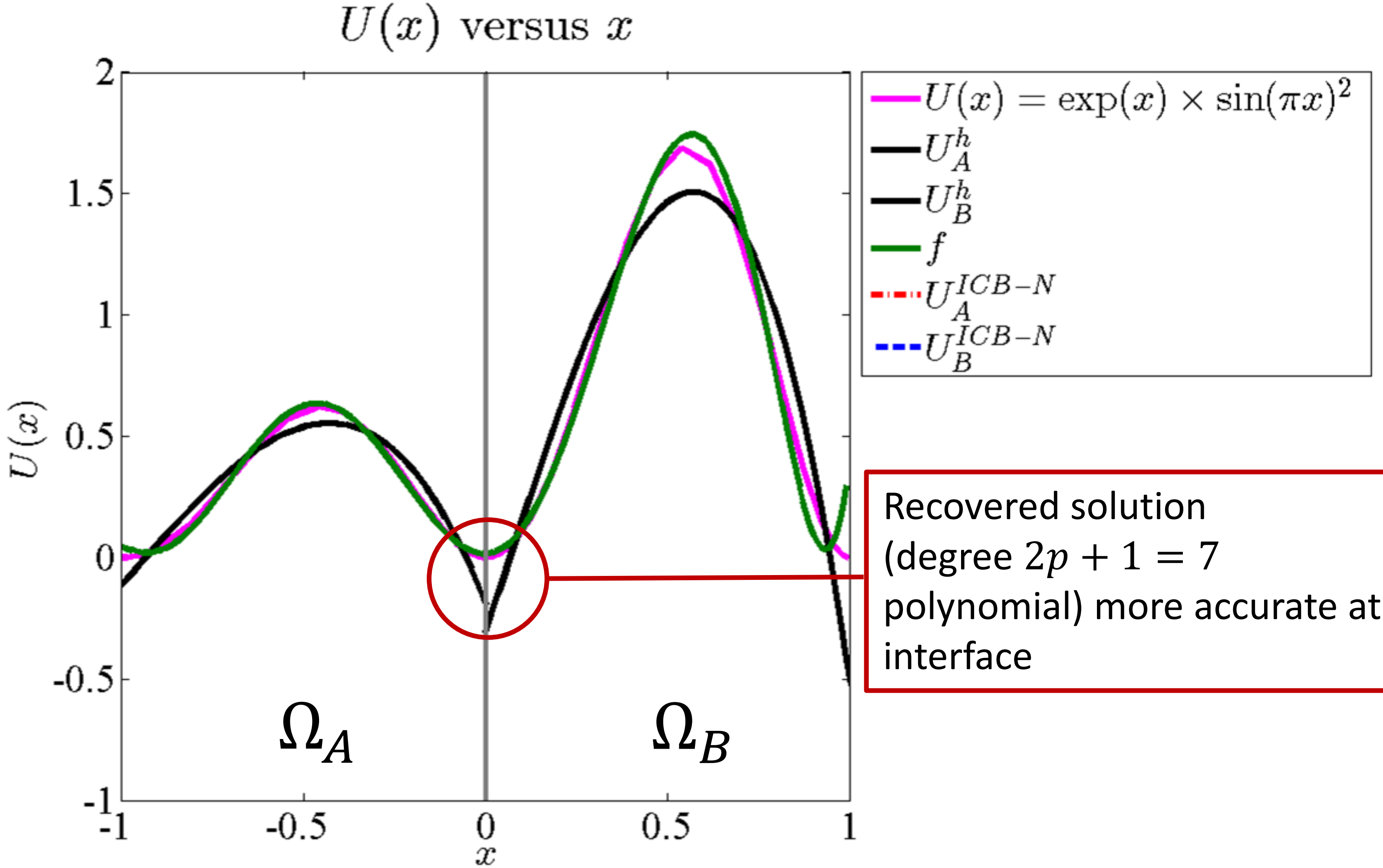
In troubled elements: $G^x \sim \left|\frac{\partial u}{\partial x}\right|, G^y \sim \left|\frac{\partial v}{\partial y}\right|$

$$\underbrace{\frac{\partial}{\partial x}\begin{bmatrix} \kappa^x \rho_{,x} \\ \kappa^x(\rho u)_{,x} \\ \kappa^x(\rho v)_{,x} \\ \kappa^x E_{,x} \\ \mu_S C^x_{,x} \\ 0 \end{bmatrix} + \frac{\partial}{\partial y}\begin{bmatrix} \kappa^y \rho_{,y} \\ \kappa^y(\rho u)_{,y} \\ \kappa^y(\rho v)_{,y} \\ \kappa^y E_{,y} \\ 0 \\ \mu_s C^y_{,y} \end{bmatrix}}_{\text{Artificial dissipation + C-diffusion}} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ G^x - \frac{C^x}{\epsilon} \\ G^y - \frac{C^y}{\epsilon} \end{bmatrix}}_{\text{Source/Sink term for C variable}}$$

Recovery Demonstration: $p = 3$

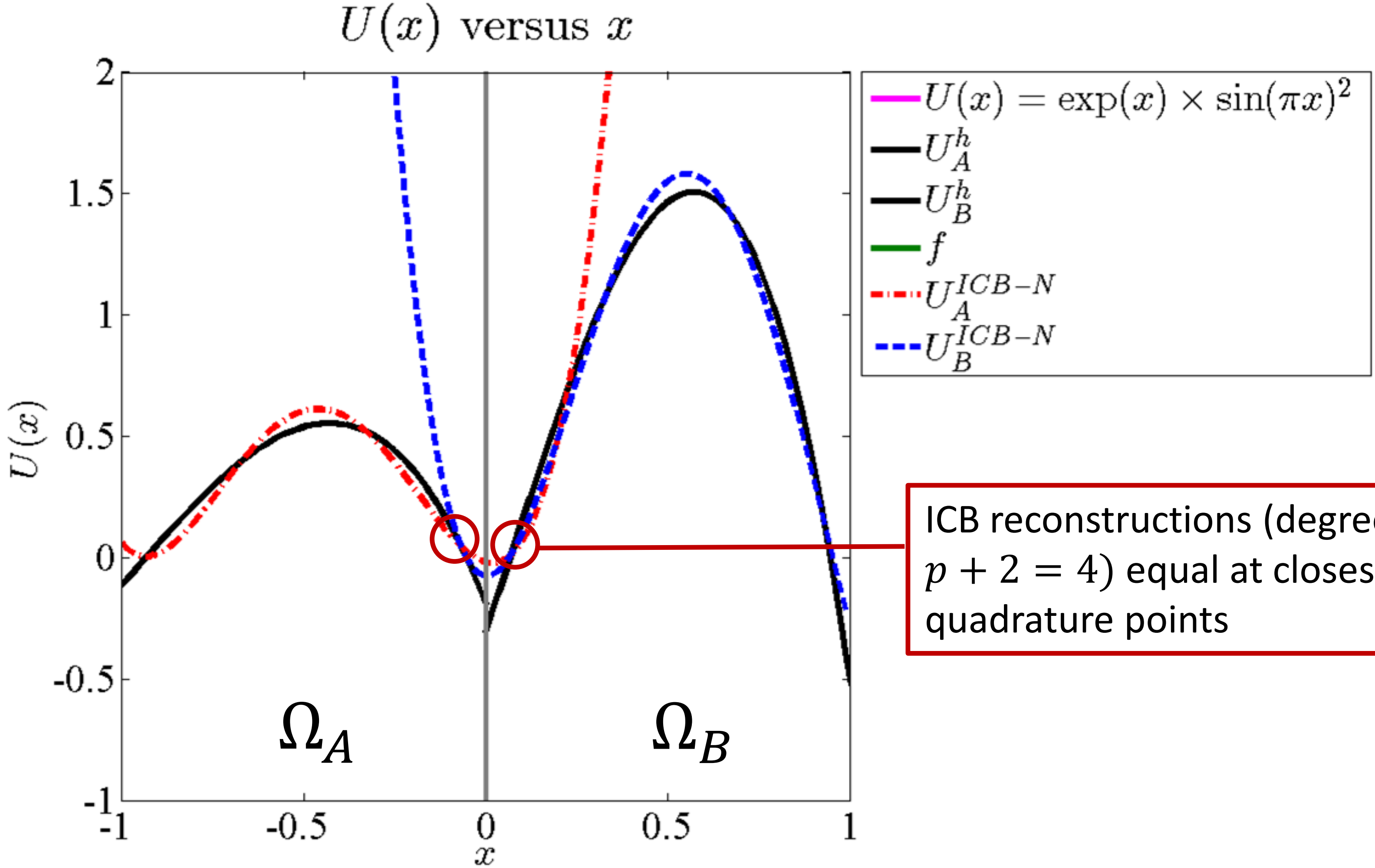


Recovery Demonstration: $p = 3$



Recovered solution (degree $2p + 1 = 7$ polynomial) more accurate at interface

Recovery Demonstration: $p = 3$



ICB reconstructions (degree $p + 2 = 4$) equal at closest quadrature points

UNIVERSITY OF MICHIGAN

Our Approach vs. Conventional DG

- For diffusive fluxes: CGR maintains compact stencil[†], offers advantages over BR2
  - Larger allowable explicit timestep size
  - Improved wavenumber resolution

- For advection problems: $\int_{\Omega_e} \phi_e^k \frac{\partial}{\partial t} U_e^h d\mathbf{x} = -\int_{\Omega_e} \phi_e^k \nabla \cdot \mathcal{F}(U^h) d\mathbf{x}$
- DG weak form: Must calculate flux along interfaces
  - Conventional approach (upwind DG): plug in left/right values of DG solution

$$\int_{\Omega_e} \phi_e^k \frac{\partial}{\partial t} U_e^h d\mathbf{x} = -\int_{\partial\Omega_e} \phi_e^k (\tilde{\mathcal{F}} \cdot n^-) ds + \int_{\Omega_e} (\nabla \phi_e^k) \cdot \mathcal{F}(U_e^h)) dx$$

- **Conventional approach:** $\tilde{\mathcal{F}} = Rie(U_L^h, U_R^h, n_L)$
- **Our approach:** ICB reconstruction scheme[††]
  - Replace left/right solution values with ICB reconstruction: $\tilde{\mathcal{F}} = Rie(U_L^{ICB}, U_R^{ICB}, n_L)$

† Johnson & Johnsen, AIAA Aviation 2017
†† Khieu & Johnsen, AIAA Aviation 2014

Vortex Transport Case (VI1)

Setup 1: $p = 1$, RK4, SLAU Riemann solver

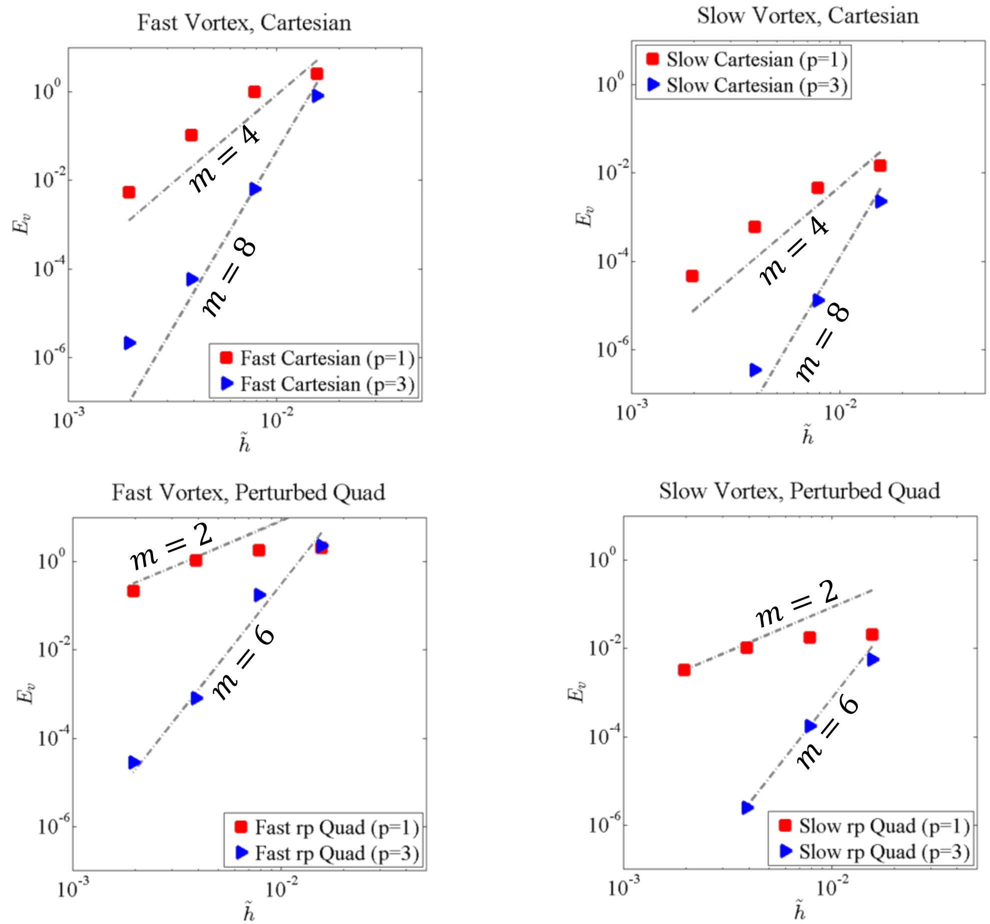
Setup 2: $p = 3$, RK8† (13 stages), SLAU Riemann solver

ICB usage: Apply ICB on Cartesian meshes, conventional DG otherwise

EQ: Global L_2 error of v:

$$E_v = \sqrt{\frac{\int_\Omega (v - v_0)^2 dV}{\int_\Omega dV}}$$

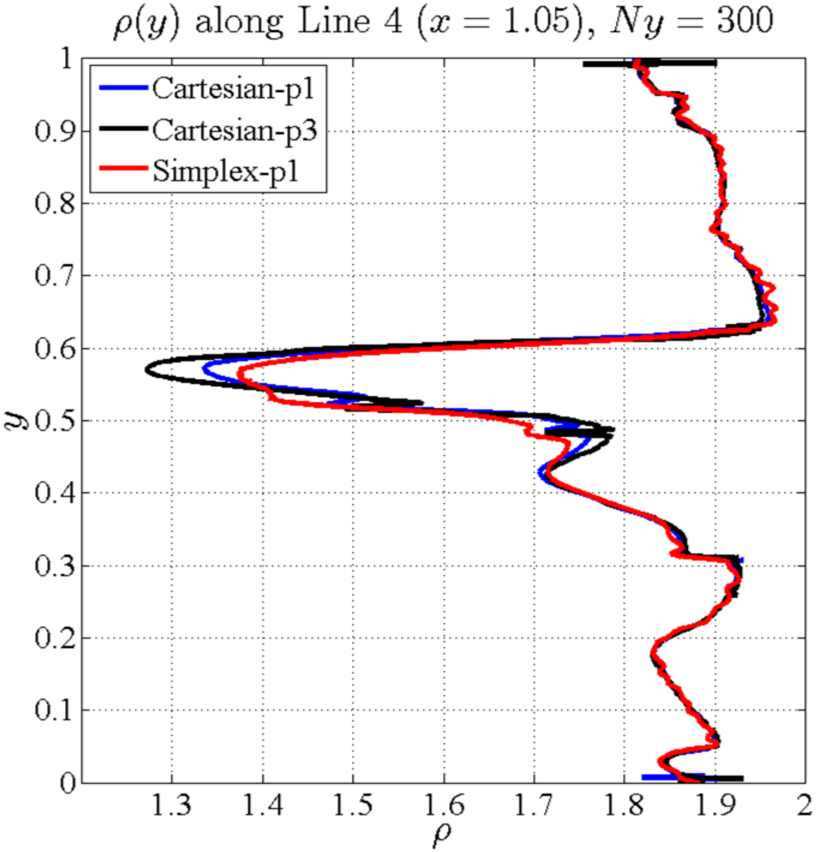
Convergence: order $2p + 2$ on Cartesian mesh, order $2p$ on perturbed quad mesh

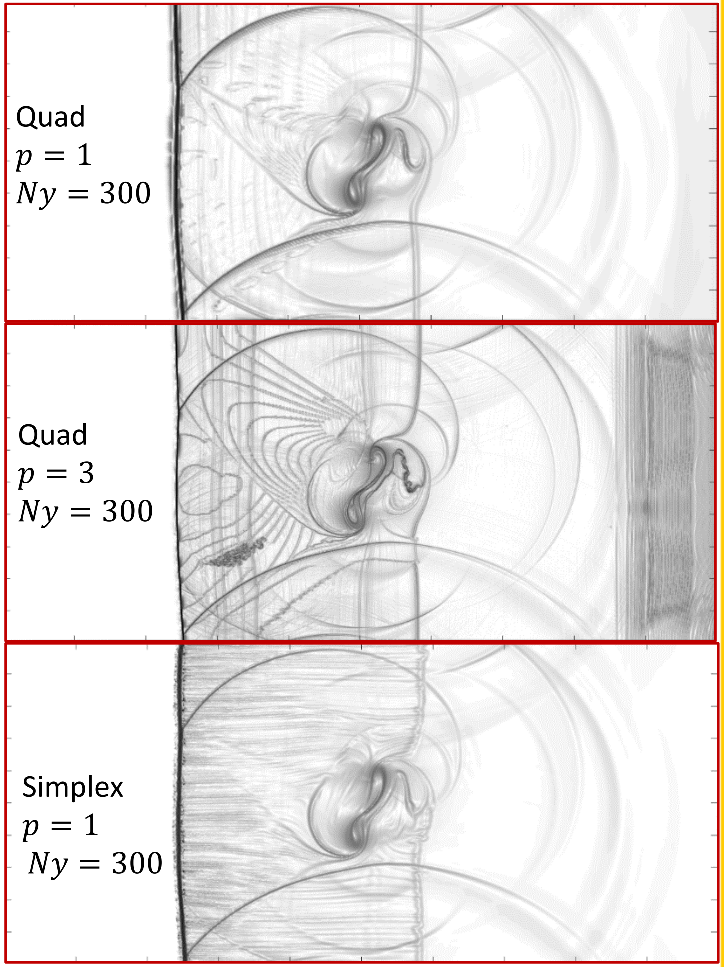


† Cash & Karp, ACMTMS 1990

Shock-Vortex Interaction (CI2)

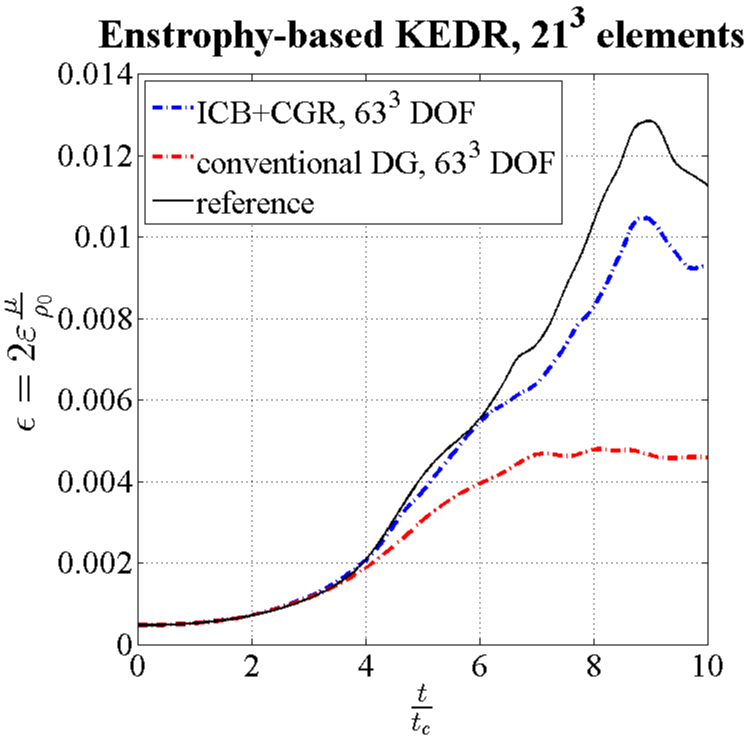
Configurations: Cartesian ($p = 1$), Cartesian ($p = 3$), Irregular Simplex ($p = 1$)
Setup: RK4 time integration, SLAU (Cartesian) and Roe (Simplex) Riemann solvers
Shock Capturing: PDE-based artificial dissipation
ICB usage: Only on Cartesian grids



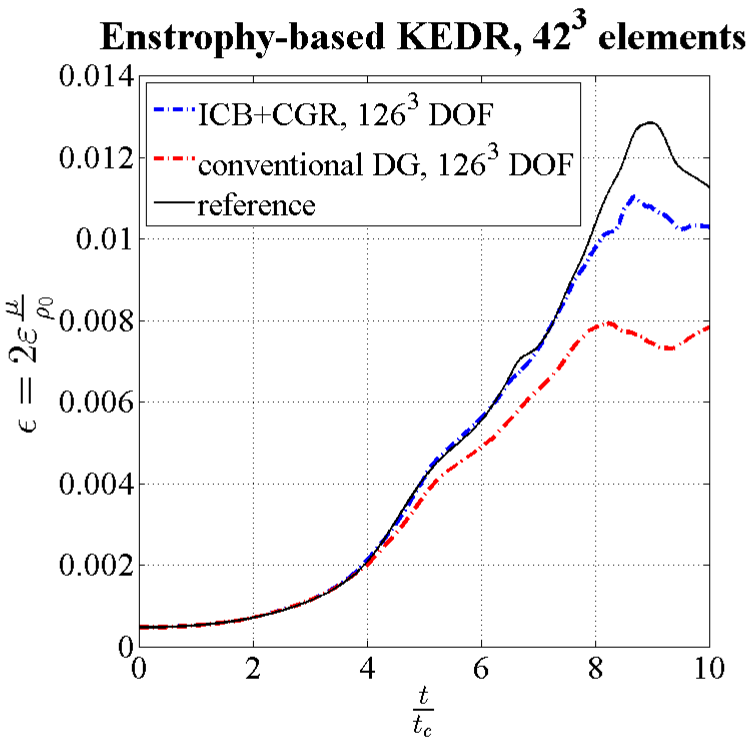


Taylor-Green Test (WS1)

- **Code setup**: p2 elements, uniform hex mesh (27 DOF/element), RK4 time integration
  - Reference result taken from HiOCFD3 workshop
  - Our approach allows larger stable time step



ICB+CGR: 2.5 CPU-hours
Conventional: 9.2 CPU-Hours



ICB+CGR: 75 CPU-hours
Conventional: 304 CPU-Hours

Conclusions

- **Were the verification cases helpful and which ones were used?**
  - Vortex transport: Shows that ICB is implemented properly
  - TGV: demonstrates value of ICB+CGR for nonlinear problem

- **What improvements are needed to the test case?**
  - Meshes for vortex transport problem are tough to work with (GMSH input file preferred)
  - Shock-Vortex interaction: No improvement, test case is perfect
  - TGV: Standardize energy spectrum calculation and make reference data more easily accessible

- **Did the test case prompt you to improve your methods/solver**
  - Yes: added shock capturing on non-Cartesian elements

Conclusions

- **What worked well with your method/solver?**
  - Feature resolution on Cartesian meshes (ICB very helpful)

- **What improvements are necessary to your method/solver?**
  - Implicit/Explicit time integration for advection-diffusion
  - Recovery troublesome on non-Cartesian elements
  - Parallel efficiency with solution-adaptive approach
  - Curved elements

SciTech Talk

Title: A Compact Discontinuous Galerkin Method for Advection-Diffusion Problems
Session: FD-33, High-Order CFD Methods 1
Setting: Sun 2, January 10, 9:30 AM

Acknowledgements

Computing resources were provided by the NSF via grant 1531752 MRI: Acquisition of Conflux, A Novel Platform for Data-Driven Computational Physics (Tech. Monitor: Ed Walker).

References

- Kitamura, K. & Shima, E., "Towards shock-stable and accurate hypersonic heating computations: A new pressure flux for AUSM-family schemes," *Journal of Computational Physics*, Vol. 245, 2013.
- Reisner, J., Serensca, J., Shkoller, S., "A space-time smooth artificial viscosity method for nonlinear conservation laws," *Journal of Computational Physics,* Vol. 235, 2013.
- Johnson, P.E. & Johnsen, E., "A New Family of Discontinuous Galerkin Schemes for Diffusion Problems," *23rd AIAA Computational Fluid Dynamics Conference*, 2017.
- Khieu, L.H. & Johnsen, E., "Analysis of Improved Advection Schemes for Discontinuous Galerkin Methods," *7th AIAA Theoretical Fluid Dynamics Conference*, 2011.
- Cash, J.R. & Karp, A.H., "A Variable Order Runge-Kutta Method for Initial Value Problems with Rapidly Varying Right-Hand Sides," *ACM Transactions on Mathematical Software,* Vol. 16, No. 3, 1990.

Spare Slides

CGR = Mixed Formulation + Recovery

Gradient approximation in Ω_e:

$$\sigma(x \in \Omega_e) = \sigma_e(x) = \sum_{k=0}^{p} \phi^k(\xi)\ \hat{\sigma}_e^k$$

Weak equivalence with ∇U:

$$\int_{\Omega_e} \phi^k\ \sigma_e dx = \int_{\Omega_e} \phi^k\ \nabla U^h dx \quad \forall k \in \{0, 1, ..., p\}$$

Integrate by parts for σ weak form:

$$\int_{\Omega_e} \phi^k\ \sigma_e dx = [\phi^k\ \tilde{U}]_L^R - \int_{\Omega_e} U_e^h\ \nabla \phi^k dx \quad \forall k \in \{0, 1, ..., p\}$$

- Must choose interface $\widetilde{U}$ approximation from available data
  - BR2: Take average of left/right solutions at the interface
  - **Compact Gradient Recovery (CGR):** $\widetilde{U}$ = recovered solution

- Interface gradient: CGR formulated to maintain compact stencil

UNIVERSITY OF MICHIGAN

The Recovery Concept

- Recovery: reconstruction technique introduced by Van Leer and Nomura† in 2005
- Recovered solution (f_{AB}) and DG solution (U^h) are equal in the weak sense
- Generalizes to 3D hex elements via tensor product basis

Recovered Solution for $\mathcal{I}_{AB}$:

$$f_{AB}(r) = \sum_{n=0}^{2p+1} \psi^n(r)\ \hat{f}^n_{AB}$$

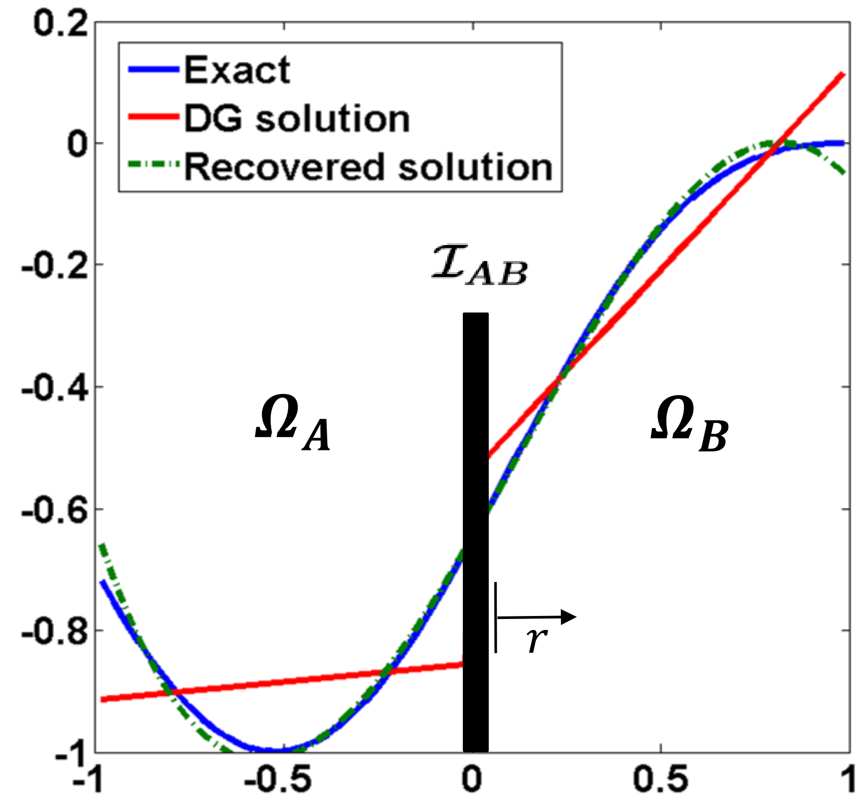
$K_R = 2p + 2$ constraints for f_{AB}:

$$\int_{\Omega_A} \phi^k_A\ f_{AB}\ dx = \int_{\Omega_A} \phi^k_A\ U^h_A\ dx \quad \forall k \in \{0, 1, ..., p\}$$

$$\int_{\Omega_B} \phi^k_B\ f_{AB}\ dx = \int_{\Omega_B} \phi^k_B\ U^h_B\ dx \quad \forall k \in \{0, 1, ..., p\}$$

Interface Solution along $\mathcal{I}_{AB}$:

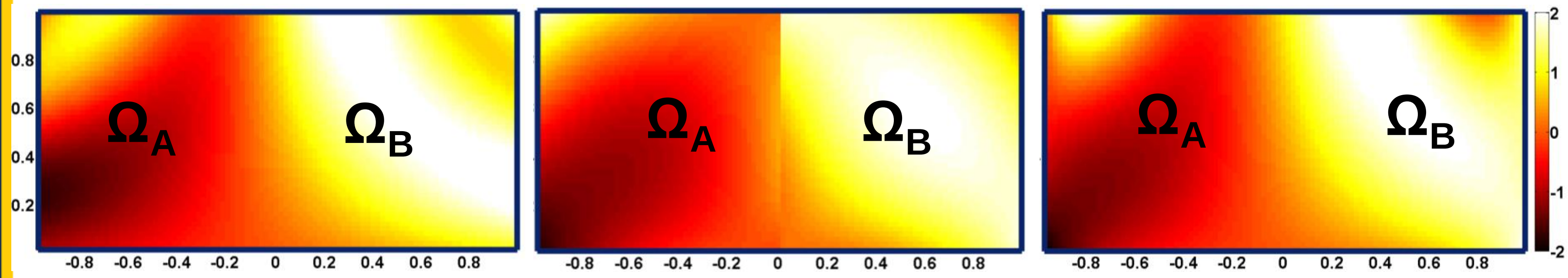
$$\mathcal{R}(U_A, U_B) = f_{AB}(0)$$



Representations of $U(x) = sin^3(x - \frac{\pi}{3})$

Recovery DG†

Exact Distribution U | DG solution: U_h^A , U_h^B | Recovered solution: f_{AB}



$$U = x + y + \sin(2\pi xy)$$

$$f_{AB} = \sum_{m=1}^{2K} \hat{f}_{AB}^{m} \psi^{m}(\vec{x})$$

$$U_A^h = \sum_{m=1}^{K} \hat{U}_A^m \phi^m$$

$$U_B^h = \sum_{m=1}^{K} \hat{U}_B^m \phi^m$$

$$\int_{\Omega_A} f_{AB}\phi^k dA = \int_{\Omega_A} U_A^h \phi^k dA \quad \forall k \in \{1..K\}$$

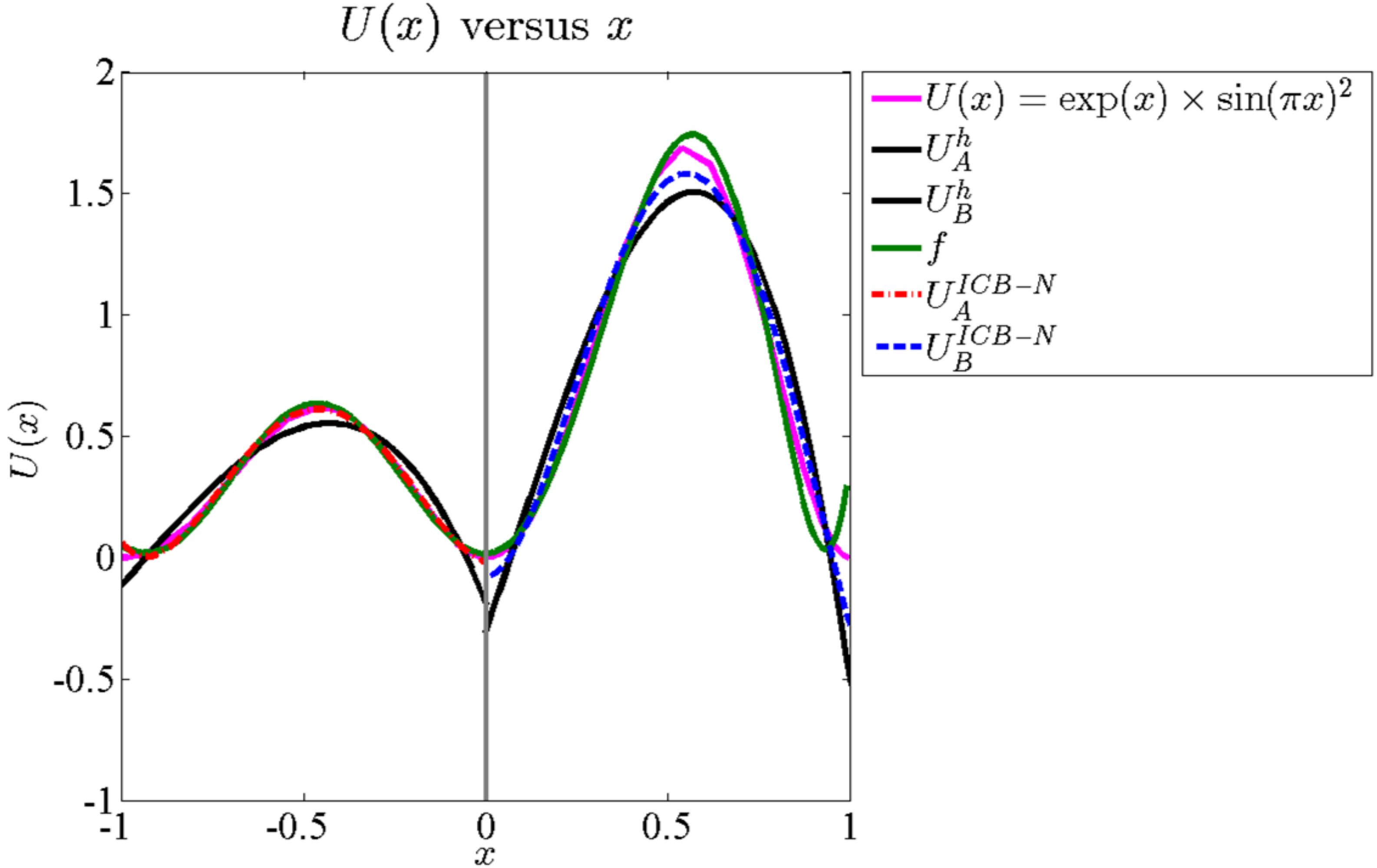
$$\int_{\Omega_B} f_{AB}\phi^k dA = \int_{\Omega_B} U_B^h \phi^k dA \quad \forall k \in \{1..K\}$$

- Recovered solution is continuous across the interface, uniquely defines $(U, \nabla U)$
  - Conventional DG approaches for Navier-Stokes lack this property

†Van Leer & Nomura, AIAA Conf. 2005
Schematic from [Johnson & Johnsen, APS DFD 2015]

UNIVERSITY OF MICHIGAN

Recovery Demonstration: All Solutions



The ICB reconstruction

- Each interface gets a pair of ICB reconstructions, one for each element:

$K_{ICB} = p + 2$ coefficients per element:

$$U_A^{ICB}(\mathbf{r}) = \sum_{n=1}^{K_{ICB}} \psi^n(\mathbf{r})\ \hat{C}_A^n$$

$$U_B^{ICB}(\mathbf{r}) = \sum_{n=1}^{K_{ICB}} \psi^n(\mathbf{r})\ \hat{C}_B^n$$

Constraints for U_A^{ICB}: (Similar for U_B^{ICB})

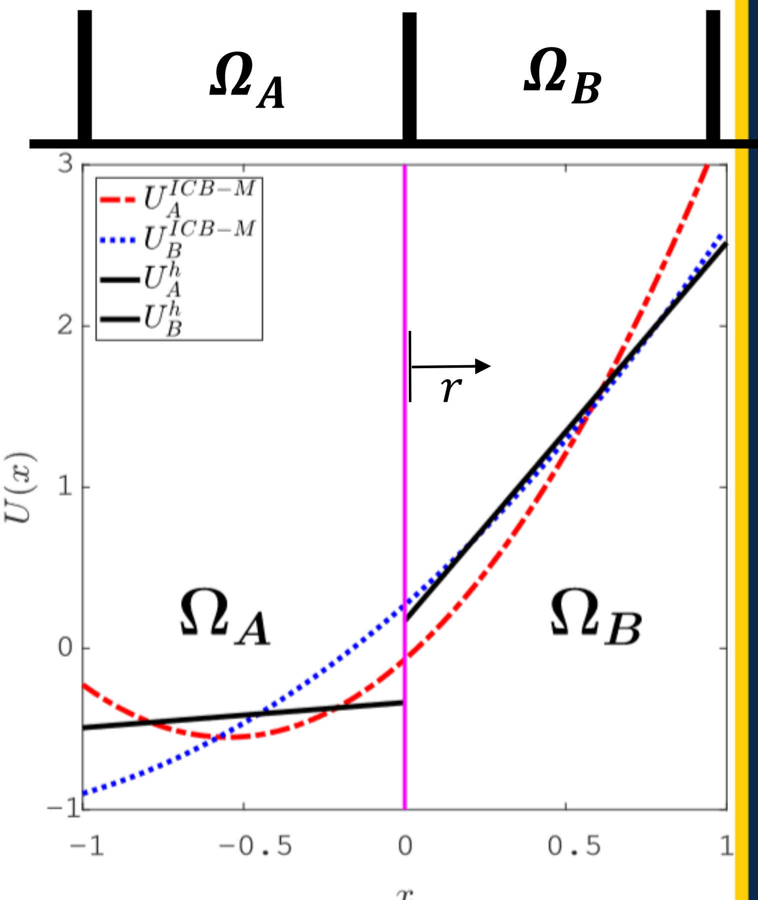
$$\int_{\Omega_A} \phi_A^k\ U_A^{ICB} dx = \int_{\Omega_A} \phi_A^k\ U_A^h dx \quad \forall k \in \{0,1,\ldots p\}$$

$$\int_{\Omega_B} \Theta_B\ U_A^{ICB} dx = \int_{\Omega_B} \Theta_B\ U_B^h dx$$

- Choice of Θ_B affects behavior of ICB scheme
  - Illustration uses $\Theta_B = 1$

Example: $p = 1$ (2 DOF/element)

$$U = e^x sin(\frac{3\pi x}{4})$$

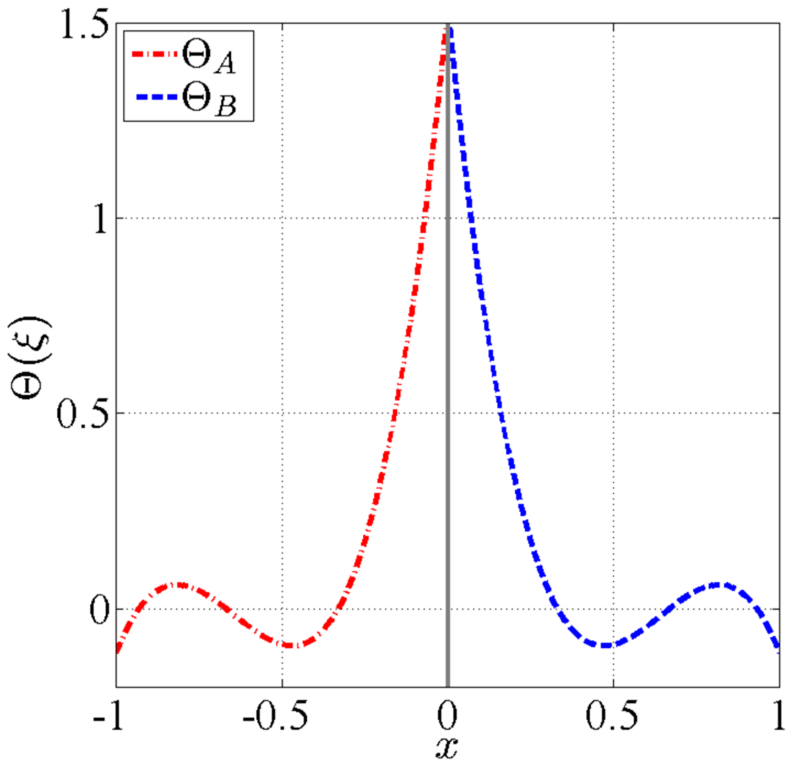


The Θ Function: ICB-Modal vs. ICB-Nodal

- **ICB-Modal (original)**: $\Theta_{\mathrm{A}} = \Theta_B = 1$ is lowest mode in each element's solution

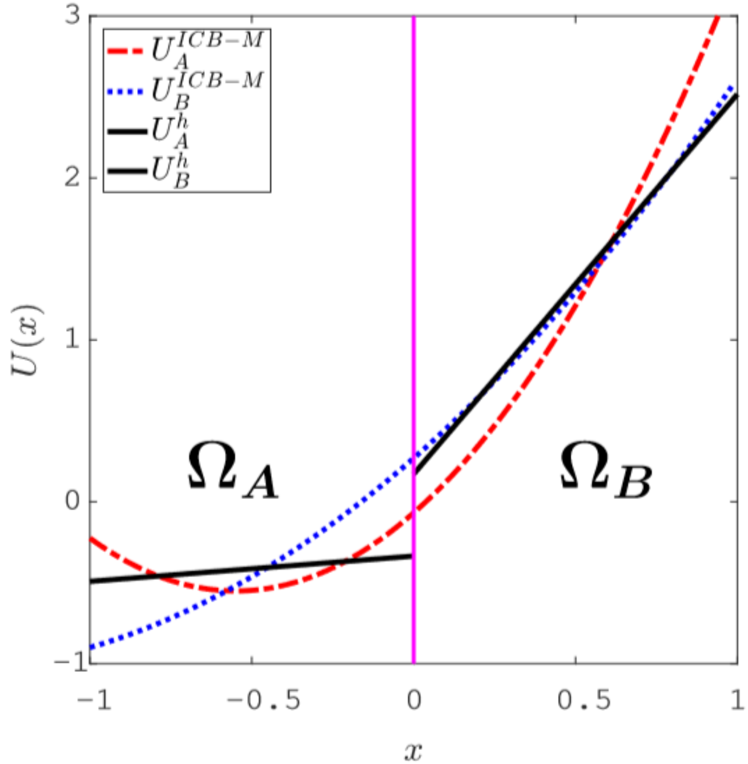
- **ICB-Nodal (new approach)**: Θ is degree p Lagrange interpolant
  - Use Gauss-Legendre quadrature nodes as interpolation points
  - Take Θ nonzero at closest quadrature point

Sample Θ choice for $p = 3$:
Each Θ is unity at quadrature point nearest interface

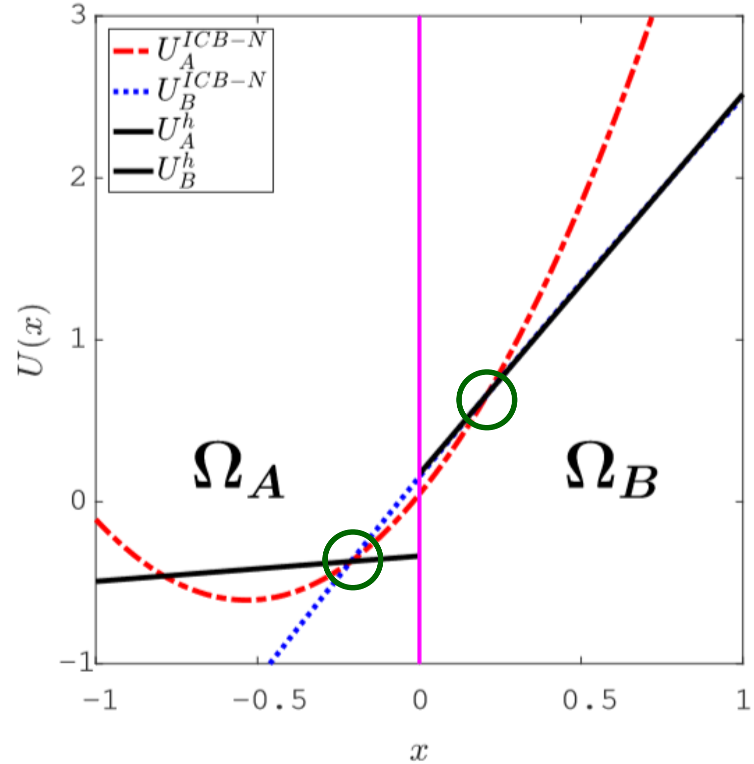


The Θ Function: ICB-Modal vs. ICB-Nodal

ICB-Modal: Each U^{ICB} matches the average of U^h in neighboring cell



ICB-Nodal: Each U^{ICB} matches U^h at near quadrature point



Fourier Analysis

- Fourier analysis performed on 2 configurations:
  - Conventional: Upwind DG + BR2
  - New: ICB-Nodal + CGR

Scheme	$\widetilde{F}$	$\widetilde{U}$
uDG + BR2	$\mathrm{Rie}(U_A^h, U_B^h, n_A^-)$	$\{\{U^h\}\}$
ICB + CGR	$\mathrm{Rie}(U_A^{ICB}, U_B^{ICB}, n_A^-)$	$\mathcal{R}(U_A^h, U_B^h)$

Analysis Procedure † :

1) Linear advection-diffusion, 1D: $$\frac{\partial U}{\partial t} = \mu \frac{\partial^2 U}{\partial x^2} - a \frac{\partial U}{\partial x}$$

2) Define element Peclet number: $$PE_h = \frac{ah}{\mu}$$

3) Set Initial condition: $U(x, 0) = \exp(i\omega' x)$ $\quad \omega = h\omega'$ $\quad \hat{\boldsymbol{U}}_{\boldsymbol{m+J}} = \exp(iJw) \cdot \hat{\boldsymbol{U}}_{\boldsymbol{m}}$

4) Cast numerical scheme in matrix-vector form: $$\frac{\partial}{\partial t} \hat{\boldsymbol{U}}_{\boldsymbol{m}} = \frac{\mu}{h^2} \cdot \boldsymbol{\mathcal{A}}(\omega, PE_h) \hat{\boldsymbol{U}}_{\boldsymbol{m}}$$

† Watkins et al., Computers & Fluids 2016

Fourier Analysis

5) Diagonalize the update matrix: $\boldsymbol{\mathcal{A}} = \boldsymbol{V\Lambda V^{-1}}$

6) Calculate initial expansion weights, $\boldsymbol{\beta}$: $\boldsymbol{V\beta} = \boldsymbol{\hat{U}_m}(\omega, 0)$

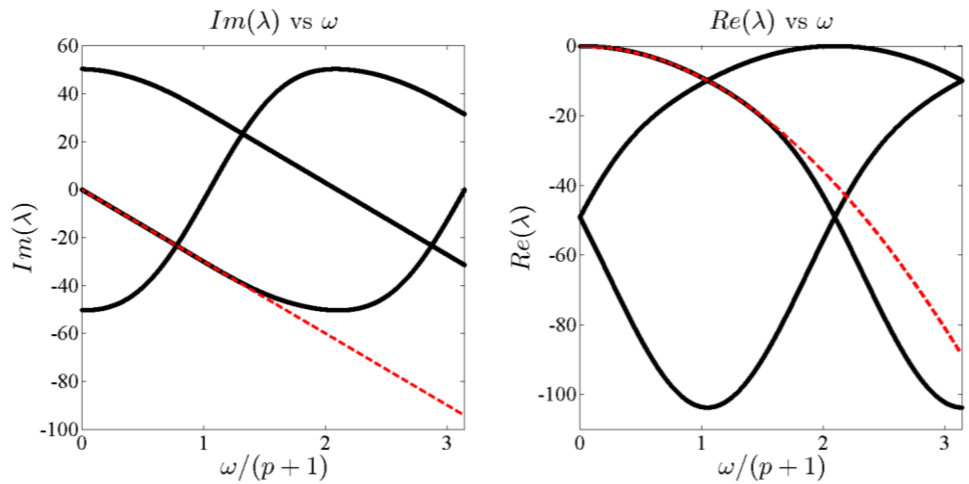
- Watkins et al. derived estimate for initial error growth:
  - — λ^n = n^{th} eigenvalue of $\boldsymbol{\mathcal{A}}$

$$\mathcal{E}(\omega, PE_h) = \frac{1}{\sqrt{p+1}} \sum_{n=1}^{p+1} |\beta_n||\lambda_n - \lambda^{ex}|$$

Eigenvalue corresponding to exact solution:

$$\lambda^{ex} = -i(PE_h\omega) - \omega^2$$

Eigenvalue Example:
ICB+CGR, $p = 2$, $PE_h = 10$,
$\lambda^{ex} = -i(10\omega) - \omega^2$



Wavenumber Resolution

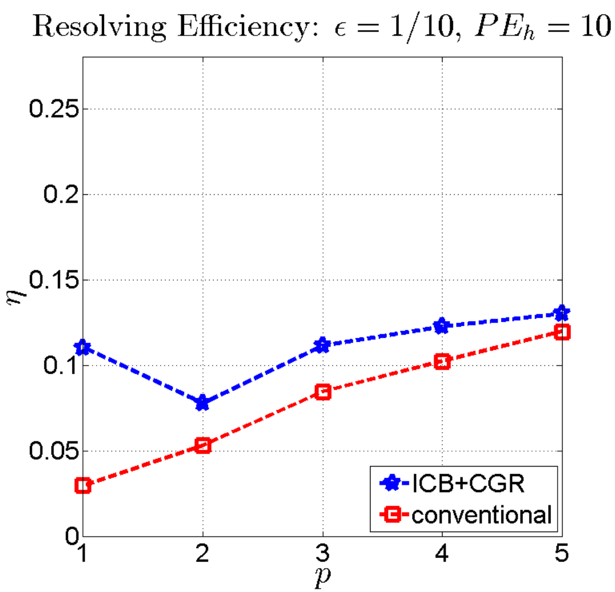
$$\mathcal{E}(\omega, PE_h) = \frac{1}{\sqrt{p+1}} \sum_{n=1}^{p+1} |\beta_n||\lambda_n - \lambda^{ex}|$$

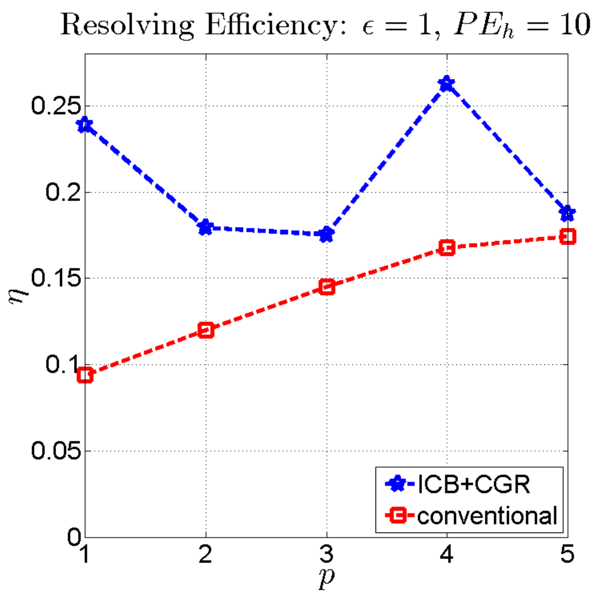
- To calculate wavenumber resolution:
  1) Define some error tolerance(ϵ) and Peclet number (PE_h)
  2) Identify cutoff wavenumber, ω_f according to: $\mathcal{E}(\omega, PE_h) \leq \epsilon$ for all $\omega \in [0, \omega_f]$.
  3) Calculate resolving efficiency: $\eta = \dfrac{\omega_f}{(p+1)\pi}$

† Watkins et al., Computers & Fluids 2016

Scheme Comparison: $PE_h = 10$

- Fourier analysis, Linear advection-diffusion
- Resolving efficiency measures effectiveness of update scheme's consistent eigenvalue





P	Conventional	ICB + CGR
1	0.0296	0.1103
2	0.0531	0.0776
3	0.0844	0.1113
4	0.1022	0.1225
5	0.1196	0.1304

P	Conventional	ICB + CGR
1	0.0940	0.2389
2	0.1200	0.1793
3	0.1451	0.1755
4	0.1677	0.2628
5	0.1743	0.1874

UNIVERSITY OF MICHIGAN

Compact Gradient Recovery (CGR) Approach

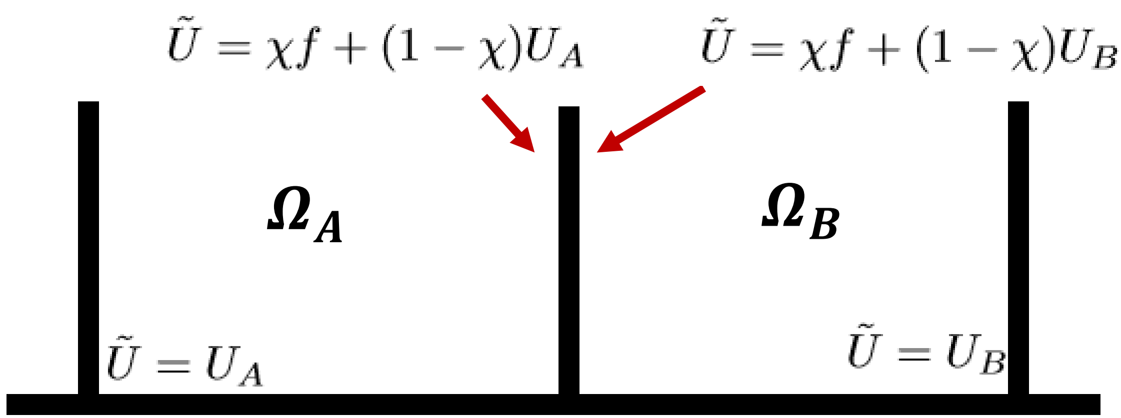
- Similar to BR2: Manage flow of information by altering gradient reconstruction
- 1D Case shown for simplicity: Let g_A, g_B be gradient reconstructions in Ω_A, Ω_B
  - Perform Recovery over g_A, g_B for $\tilde{\sigma}$ on the shared interface

$$\int_{\Omega_A} \phi^k g_A dx = \int_{\Omega_A} \phi^k \nabla U^h dx \quad \forall k \in \{1..K\}$$

$$\int_{\Omega_B} \phi^k g_B dx = \int_{\Omega_B} \phi^k \nabla U^h dx \quad \forall k \in \{1..K\}$$

$$\tilde{\sigma} = \mathcal{R}(g_A, g_B)$$

$$\int_{\Omega_e} \phi^k g_e dx = (\phi^k \tilde{U})_R - (\phi^k \tilde{U})_L - \int_{\Omega_e} (\nabla \phi^k) U^h dx \quad \forall k \in \{1..K\}$$



The ICB Approach (Specifically, ICBp[0])

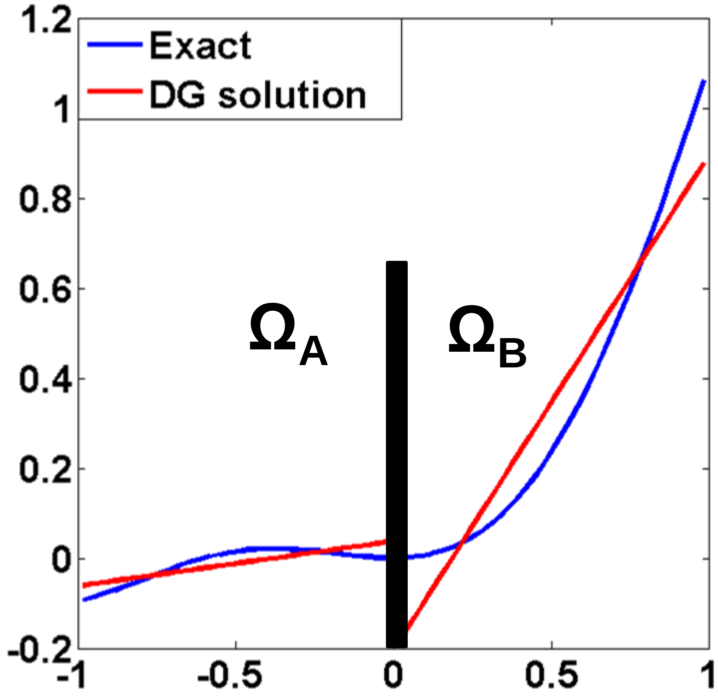
- Recovery is applicable ONLY for viscous terms; unstable for advection terms.
- Interface-Centered Binary (ICB) reconstruction scheme modifies Recovery approach for hyperbolic PDE.

Process Description:

1. Start with the DG polynomials U_A^h in Ω_A and U_b^h in Ω_B.

Example with $p1$ elements:

Representations of $U(x) = sin^3(x) + \frac{x^2}{2}$



The ICB Approach (Specifically, ICBp[0])

Process Description:

1. Start with the DG polynomials U_A^h in Ω_A and U_b^h in Ω_B.

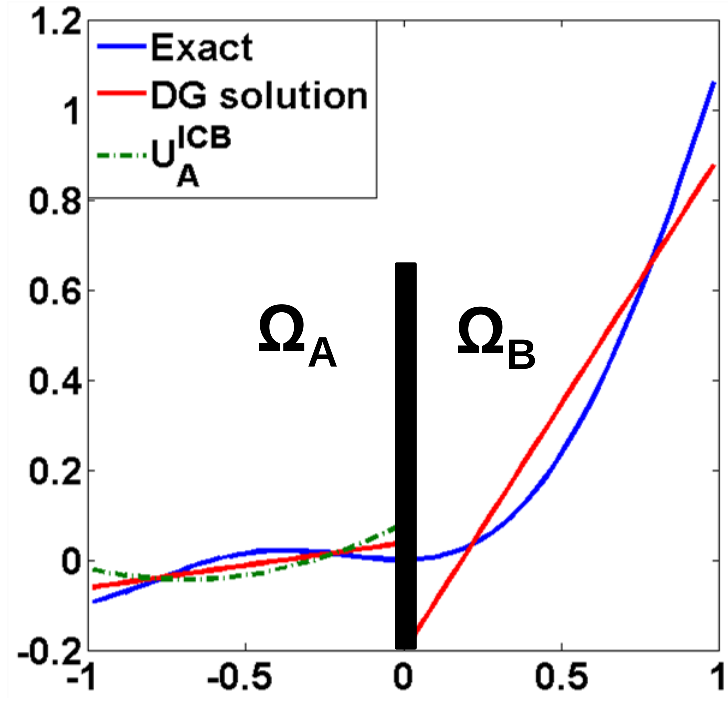
2. Obtain reconstructed solution U_A^{ICB} in Ω_A, containing $p+2$ DOF.

$$\int_{\Omega_A} U_A^{ICB} \phi^k \, dx = \int_{\Omega_A} U_A^h \phi^k \, dx \quad \forall k \in \{1..K\}$$

$$\int_{\Omega_B} U_A^{ICB} \, dx = \int_{\Omega_B} U_B^h \, dx$$

Example with $p1$ elements:

Representations of $U(x) = sin^3(x) + \frac{x^2}{2}$



The ICB Approach (Specifically, ICBp[0])

Process Description:

1. Start with the DG polynomials U_A^h in Ω_A and U_b^h in Ω_B.

2. Obtain reconstructed solution U_A^{ICB} in Ω_A, containing $p+2$ DOF.

$$\int_{\Omega_A} U_A^{ICB}\phi^k\,dx = \int_{\Omega_A} U_A^h\phi^k\,dx \quad \forall k \in \{1..K\}$$

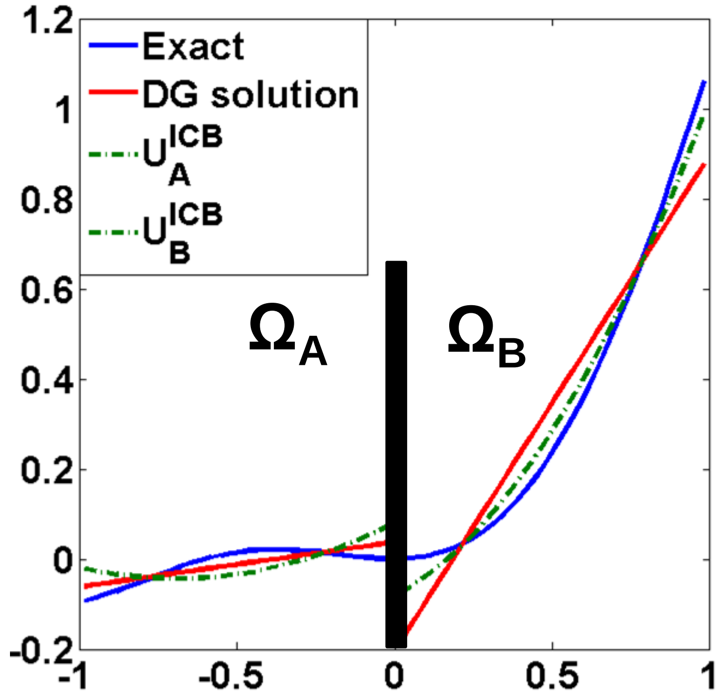
$$\int_{\Omega_B} U_A^{ICB}\,dx = \int_{\Omega_B} U_B^h\,dx$$

3. Perform similar operation for U_B^{ICB}

4. Use ICB solutions as inputs to $\widehat{\boldsymbol{H}}_{conv}(U^+,U^-)$

Example with $p1$ elements:

Representations of $U(x) = sin^3(x) + \frac{x^2}{2}$



- **ICB Method achieves $2p+2$ order of accuracy**
- **Generalizes to 2D via tensor-product basis**

Discontinuity Sensor

Approach: Check cell averages for severe density/pressure jumps across element interfaces

1) Calculate $\overline{U}$=cell average for each element
2) At each interface, use sensor of Lombardini to check for shock wave:
   i. If Lax entropy condition satisfied (hat denotes Roe average at interface):
   $$u_L - c_L > \hat{u} - \hat{c} > u_R - c_R$$
   ii. Check pressure jump:
   $$\phi = \frac{|p_R - p_L|}{p_L + p_R}, \qquad \Phi = \frac{2\phi}{(1+\phi)^2}$$
   iii. If $\Phi > 0.01$, tag both elements as "troubled"

3) At each interface, check for contact discontinuity
   i. Calculate wave strength propagating the density jump: $\Delta\widehat{\alpha_2} = \dfrac{\Delta\rho\hat{c}^2 - \Delta p}{\hat{c}^2}$
   ii. Check relative strength: $\xi = \dfrac{|\Delta\alpha_2|}{\rho_L + \rho_R}, \qquad \Xi = \dfrac{2\xi}{(1+\xi)^2}.$
   iii. If $\Xi > 0.01$, tag both elements as "troubled"