Accuracy-Preserving Boundary Quadrature for Edge-Based Finite-Volume Scheme:

Third-order accuracy without curved elements

JCP2015, v281, pp518-555

Hiroaki Nishikawa

National Institute of Aerospace

56th NIA CFD Seminar

December 16, 2014

“You Already Have It.”

You already have what you want:
happiness, jobs, money, or anything.

The issue is always just how to manifest what you already have.

The first step is to believe it.

Edge-Based Discretization

Edge-Based Finite-Volume Method

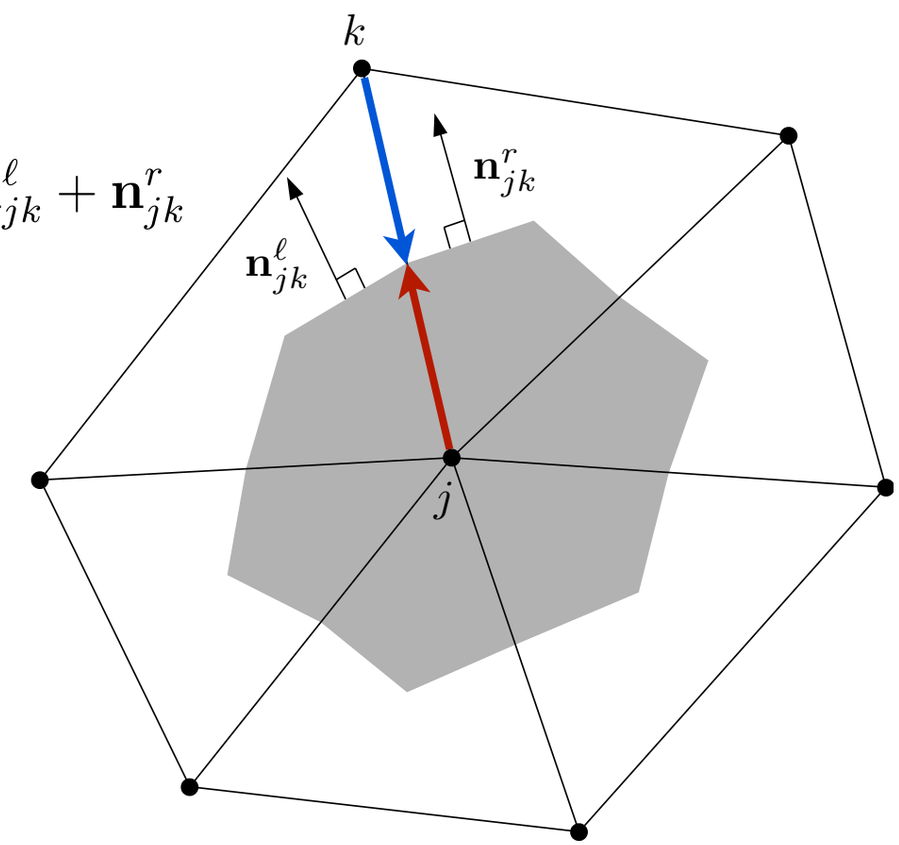
NASA's FUN3D; Software Cradle's SC/Tetra; DLR Tau code, etc.

$$\operatorname{div} \mathbf{f} = 0$$

Edge-based finite-volume scheme:

$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$

$$\mathbf{n}_{jk} = \mathbf{n}^{\ell}_{jk} + \mathbf{n}^{r}_{jk}$$



with the upwind flux at <u>edge midpoint</u>:

$$\phi_{jk}(\mathbf{n}_{jk}) = \left(\frac{1}{2}(\mathbf{f}_L + \mathbf{f}_R) \cdot \hat{\mathbf{n}}_{jk} - \frac{1}{2}|a_n|(u_R - u_L) \right) |\mathbf{n}_{jk}|$$

Accuracy with left/right states:

- 1st-order with nodal values
- 2nd-order with linear extrapolation, linear LSQ
- 3rd-order with linear extrapolation, quadratic LSQ Katz&Sankaran(JCP2011)

Efficient 3rd-order scheme: edge-loop with a flux per edge

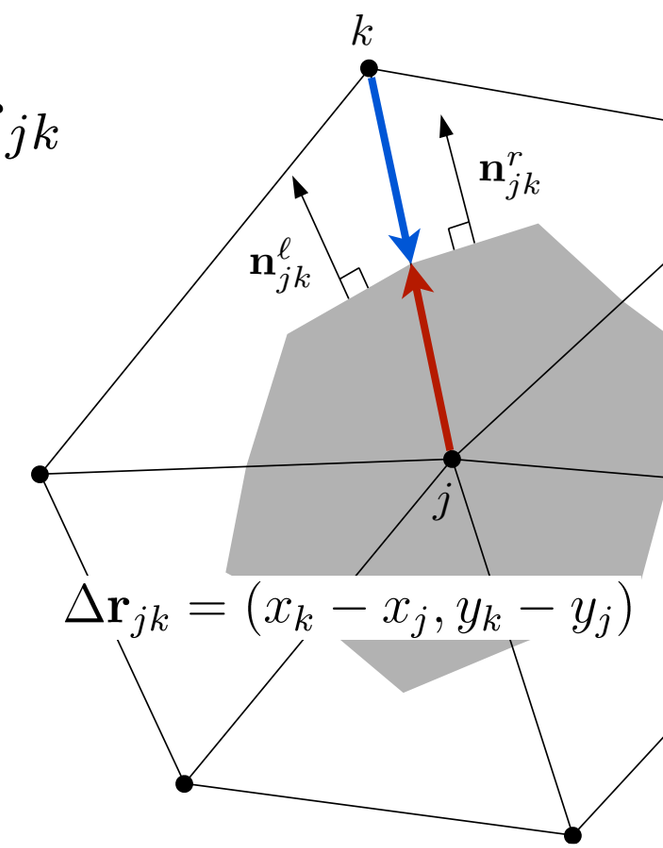
JCP2015, v281, pp518-555

Why Third-Order? Part I

Zero dissipation for quadratic solution

Linear extrapolation with quadratic LSQ gradients:

$$u_L = u_j + \frac{1}{2}\overline{\nabla} u_j \cdot \Delta \mathbf{r}_{jk} \qquad u_R = u_k - \frac{1}{2}\overline{\nabla} u_k \cdot \Delta \mathbf{r}_{jk}$$



For a quadratic solution, it gives

$$u_L = u_R = u_j + \frac{1}{2}\nabla u_j \cdot \Delta \mathbf{r}_{jk}$$

The same left and right states, but not exact.

and therefore,

$$\phi_{jk}(\mathbf{n}_{jk}) = \left(\frac{1}{2}(\mathbf{f}_L + \mathbf{f}_R) \cdot \hat{\mathbf{n}}_{jk} - \frac{1}{2}|a_n|\cancel{(u_R - u_L)}\right)|\mathbf{n}_{jk}|$$

Averaged flux is the source of error.

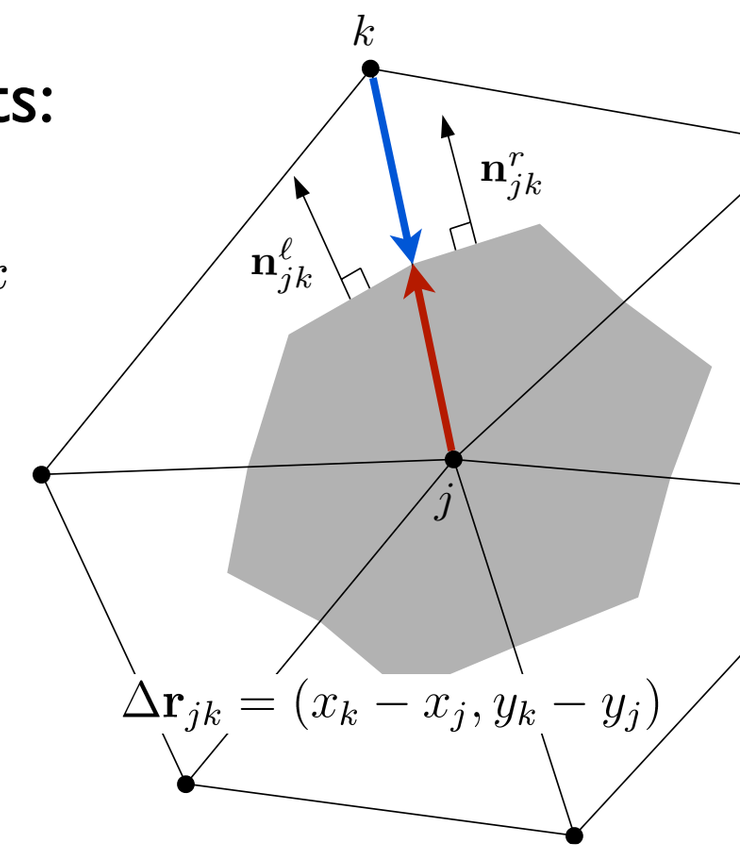
JCP2015, v281, pp518-555

Why Third-Order? Part II

Exact for quadratic fluxes

Linear flux extrapolation with quadratic LSQ gradients:

$$\mathbf{f}_L = \mathbf{f}_j + \frac{1}{2}\overline{\nabla}\mathbf{f}_j \cdot \Delta\mathbf{r}_{jk} \qquad \mathbf{f}_R = \mathbf{f}_k - \tfrac{1}{2}\overline{\nabla}\mathbf{f}_k \cdot \Delta\mathbf{r}_{jk}$$



For a quadratic flux, it gives

$$\mathbf{f}_L = \mathbf{f}_R = \mathbf{f}_j + \frac{1}{2}\nabla\mathbf{f}_j \cdot \Delta\mathbf{r}_{jk}$$

The same left and right fluxes, but not exact.

and the edge-based discretization becomes

$$\frac{1}{V_j}\sum_{k\in\{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = \frac{1}{V_j}\sum_{k\in\{k_j\}} \frac{1}{2}(\mathbf{f}_L + \mathbf{f}_R)\cdot\mathbf{n}_{jk} = \underbrace{\frac{1}{V_j}\sum_{k\in\{k_j\}} (\mathbf{f}_j + \frac{1}{2}\nabla\mathbf{f}_j \cdot \Delta\mathbf{r}_{jk})\cdot\mathbf{n}_{jk} = \operatorname{div}\mathbf{f}_j}_{\text{True for arbitrary triangles/tetrahedra}}$$

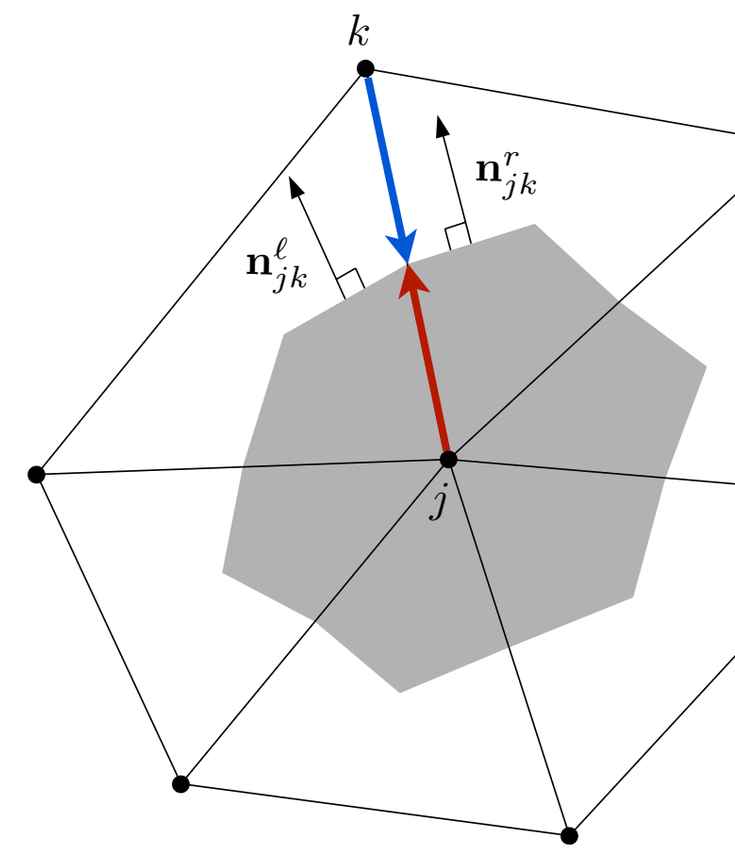
Exact for quadratic fluxes, and thus third-order accurate

JCP2015, v281, pp518-555

Lost with Exact Flux

If the flux is exact for quadratic fluxes (e.g., quadratic extrapolation or kappa=0.5 with UMUSCL), we have

$$\mathbf{f}_L = \mathbf{f}_R = \mathbf{f}_j + \frac{1}{2}\nabla\mathbf{f}_j \cdot \Delta\mathbf{r}_{jk} + \frac{1}{8}(\Delta\mathbf{r}_{jk}\cdot\nabla)^2\mathbf{f}_j$$



and the edge-based discretization becomes

$$\frac{1}{V_j}\sum_{k\in\{k_j\}}\phi_{jk}(\mathbf{n}_{jk}) = \frac{1}{V_j}\sum_{k\in\{k_j\}}\frac{1}{2}(\mathbf{f}_L+\mathbf{f}_R)\cdot\mathbf{n}_{jk}$$

$$= \frac{1}{V_j}\sum_{k\in\{k_j\}}\left(\mathbf{f}_j + \frac{1}{2}\nabla\mathbf{f}_j\cdot\Delta\mathbf{r}_{jk} + \frac{1}{8}(\Delta\mathbf{r}_{jk}\cdot\nabla)^2\mathbf{f}_j\right)\cdot\mathbf{n}_{jk} = \operatorname{div}\mathbf{f}_j + O(h)$$

TE=O(h), and so DE=O(h^2)

3rd-order is lost if the flux is exact for quadratic fluxes.

DO NOT use quadratic extrapolation nor kappa=0.5 for fluxes.

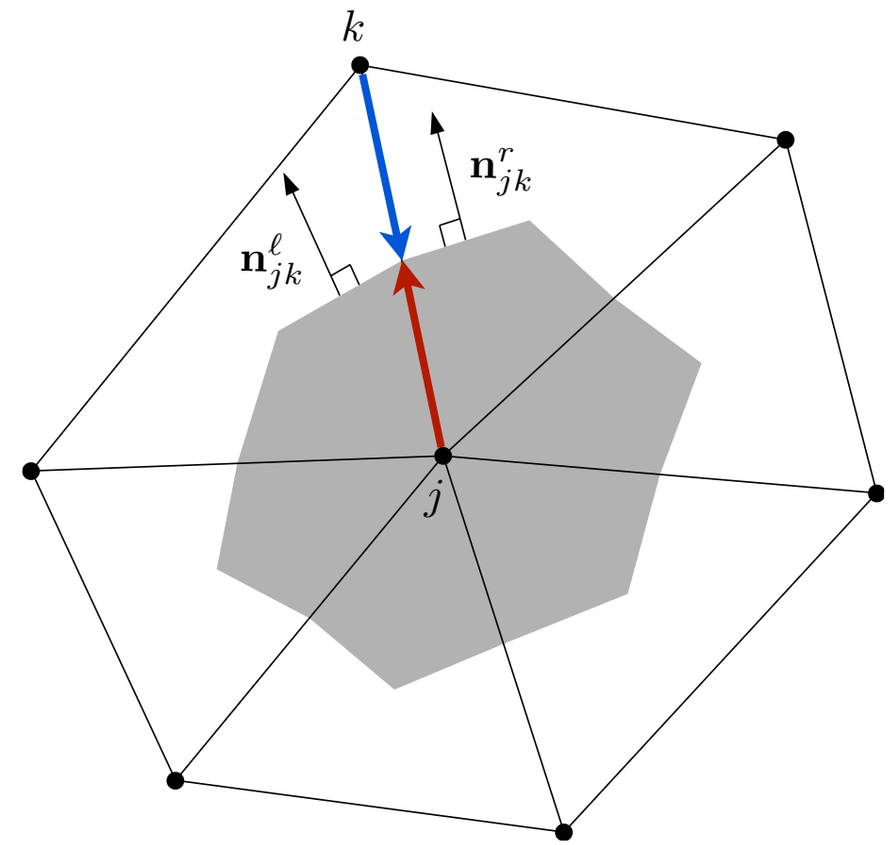
Edge-Based Discretization

1st-Order

$$u_L = u_j \qquad \mathbf{f}_L = \mathbf{f}(u_L)$$
$$u_R = u_k \qquad \mathbf{f}_R = \mathbf{f}(u_R)$$

2nd-Order (Linear LSQ gradients)

$$u_L = u_j + \frac{1}{2}\overline{\nabla} u_j \cdot \Delta \mathbf{r}_{jk} \qquad \mathbf{f}_L = \mathbf{f}(u_L)$$
$$u_R = u_k - \frac{1}{2}\overline{\nabla} u_k \cdot \Delta \mathbf{r}_{jk} \qquad \mathbf{f}_R = \mathbf{f}(u_R)$$



3rd-Order (Quadratic LSQ gradients) $\overline{\nabla}\mathbf{f}_j = \left(\frac{\partial \mathbf{f}}{\partial u}\right)_j \overline{\nabla} u_j$

$$u_L = u_j + \frac{1}{2}\overline{\nabla} u_j \cdot \Delta \mathbf{r}_{jk} \qquad \mathbf{f}_L = \mathbf{f}_j + \frac{1}{2}\overline{\nabla}\mathbf{f}_j \cdot \Delta \mathbf{r}_{jk}$$
$$u_R = u_k - \frac{1}{2}\overline{\nabla} u_k \cdot \Delta \mathbf{r}_{jk} \qquad \mathbf{f}_R = \mathbf{f}_k - \frac{1}{2}\overline{\nabla}\mathbf{f}_k \cdot \Delta \mathbf{r}_{jk}$$

EB Scheme for Diffusion, Source, Unsteady

"You Already Have It."

- **Diffusion** (Laplace) [JCP2014]

$$\partial_{xx}u + \partial_{yy}u = 0 \longrightarrow$$

Steady Conservation Law

$$\partial_x \mathbf{F} + \partial_y \mathbf{G} = \mathbf{0} \qquad \mathbf{U} = [u, p(=\partial_x u), q(=\partial_y u)]$$

$$\mathbf{F} = \begin{bmatrix} au - \nu p \\ -u/T_r - (y - y_j)\, q/T_r \\ (x - x_j)\, q/T_r \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} bu - \nu q \\ (y - y_j)\, p/T_r \\ -u/T_r - (x - x_j)\, p/T_r \end{bmatrix}$$

- **Source term** [JCP2012]

$$\partial_x f + \partial_y g = s(x, y) \longrightarrow$$

Steady Conservation Law

$$\partial_x (f - f^s) + \partial_y (g - g^s) = 0$$

$$f^s = (x - x_j)s - \frac{1}{2}(x - x_j)^2 \partial_x s + \frac{1}{6}(x - x_j)^3 \partial_{xx} s, \quad g^s = 0$$

- **Unsteady** [CF2014]

$$\partial_t u + \partial_x f + \partial_y g = 0 \longrightarrow$$

Steady Conservation Law (implicit time integration)

$$\partial_x (f - f^s) + \partial_y (g - g^s) = 0$$

3rd-order EB scheme for $\operatorname{div} \mathbf{f} = 0$ applies to various equations.

Two Approaches

diffusion, source terms, and unsteady equations

1. Modify the target equation - extra equations

   (so that, the same 3rd-order EB scheme can be applied.)

   1st/2nd/3rd-order Hyperbolic NS Schemes [AIAA2014-2091]

2. Modify the scheme - extra computational work

   Pincock and Katz, JSC, v61, Issue2, pp454-476

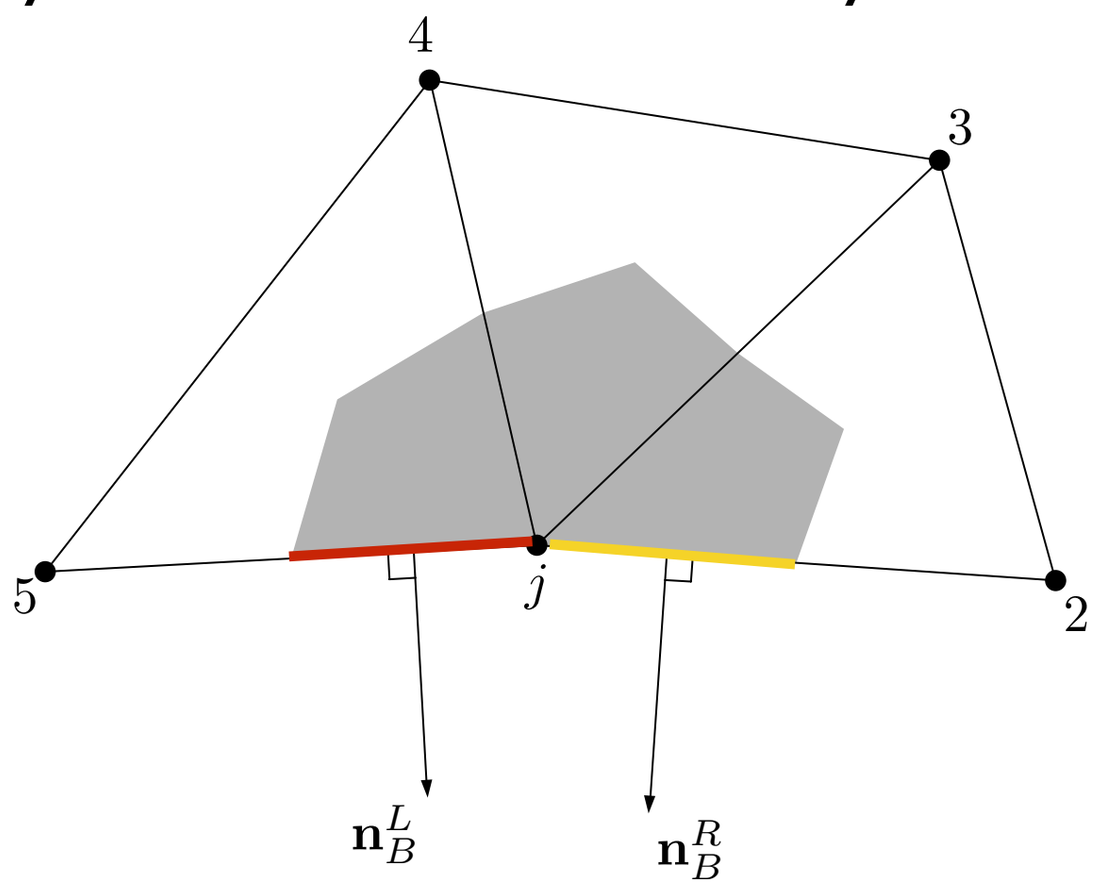
   (4th-order viscous with cubic LSQ gradients)

3rd-order EB scheme has already been demonstrated for NS.

EB Discretization at Boundary

Boundary Closure

If BCs are imposed through numerical fluxes, the residual needs to be closed by boundary contributions at boundary nodes.



$$\sum_{k=2}^{5} \phi_{jk}(\mathbf{n}_{jk}) + [\text{Boundary Fluxes}] = \phi_{j5}(\mathbf{n}_{j5}^r) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j2}(\mathbf{n}_{j2}^l) + \underline{\phi_L(\mathbf{n}_B^L)} + \underline{\phi_R(\mathbf{n}_B^R)}$$

Boundary contributions must be defined such that
the overall discretization is exact for linear/quadratic fluxes:
It depends on the element type.

Second-Order Formulas

Exact for linear fluxes

Available for triangles, quadrilaterals, tetrahedra, hexahedra, prisms, pyramids.

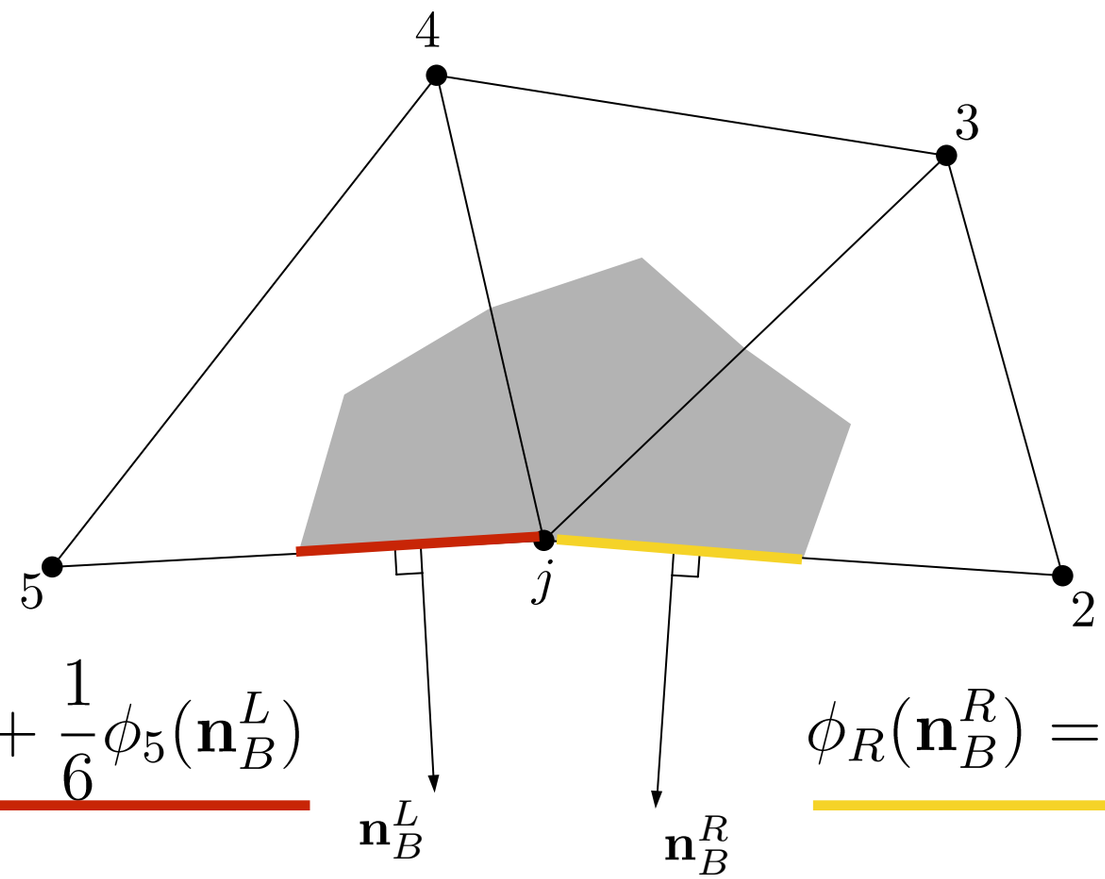
Note: Some regularity is required for quadrilaterals, hexahedra, prisms, and pyramids.

See Appendix B. in AIAA2010-5093

Appendix B. in AIAA2010-5093

Second-Order for Triangles

This formula has been known for decades (see, e.g., AIAA Paper 95-0348, 1995).



$$\phi_L(\mathbf{n}_B^L) = \frac{5}{6}\phi_j(\mathbf{n}_B^L) + \frac{1}{6}\phi_5(\mathbf{n}_B^L)$$

$$\phi_R(\mathbf{n}_B^R) = \frac{5}{6}\phi_j(\mathbf{n}_B^R) + \frac{1}{6}\phi_2(\mathbf{n}_B^R)$$

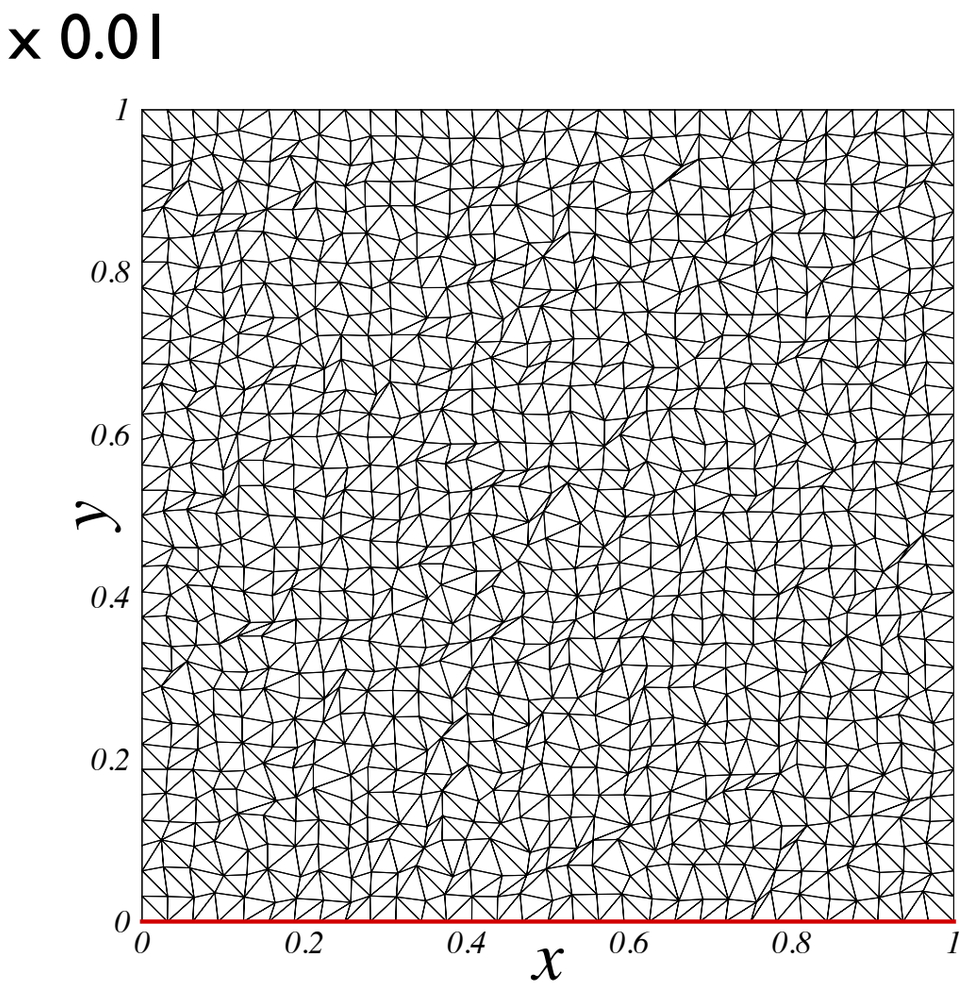
Boundary condition is set in the right (ghost) state,
and let the numerical flux determine the boundary flux.

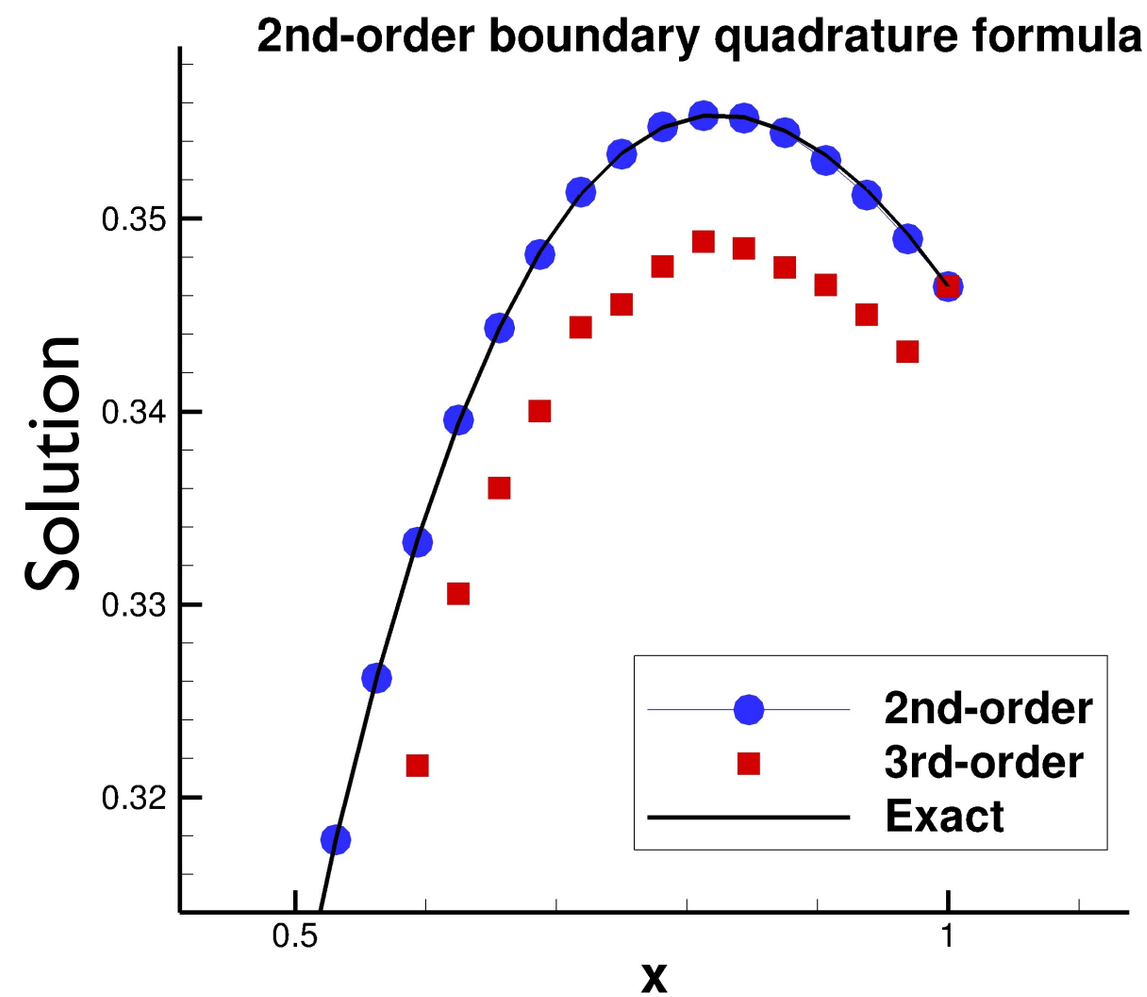
See, e.g., NASA-TM-2011-217181, AIAA2014-2923

3rd-order formula is not known at this point….
Let's see what we get with the 2nd-order boundary formula.

Terrible Results

Advection-diffusion problem



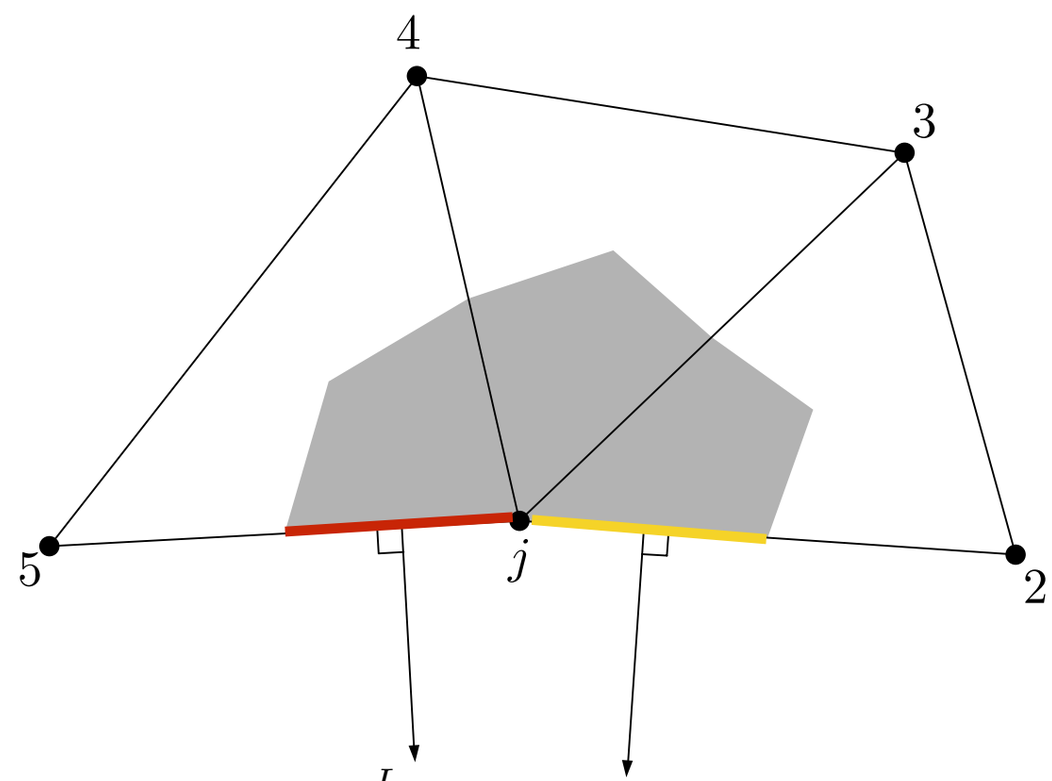


2nd-order boundary formula doesn't seem compatible with 3rd-order EB scheme.

We need a formula for third-order.

A Formula for Third-Order?

$$\sum_{k=2}^{5} \phi_{jk}(\mathbf{n}_{jk}) + [\text{Boundary Fluxes}] = \phi_{j5}(\mathbf{n}_{j5}^{r}) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j2}(\mathbf{n}_{j2}^{l}) + \underline{\phi_L(\mathbf{n}_B^L)} + \underline{\phi_R(\mathbf{n}_B^R)}$$



$\phi_R(\mathbf{n}_B^R) =$?

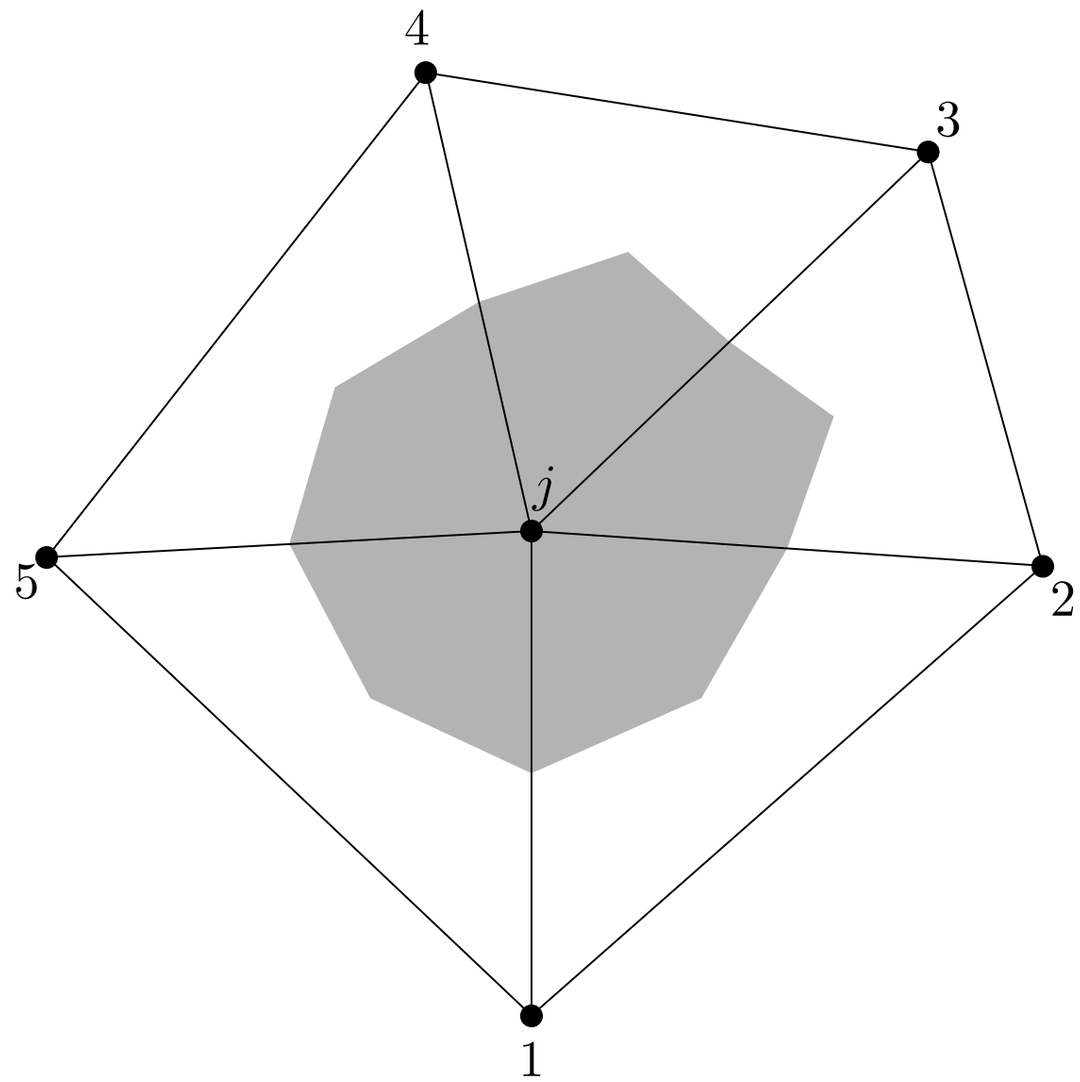
$\phi_L(\mathbf{n}_B^L) =$?

The overall discretization must be exact quadratic fluxes.

How can I derive such a formula? Very difficult…..

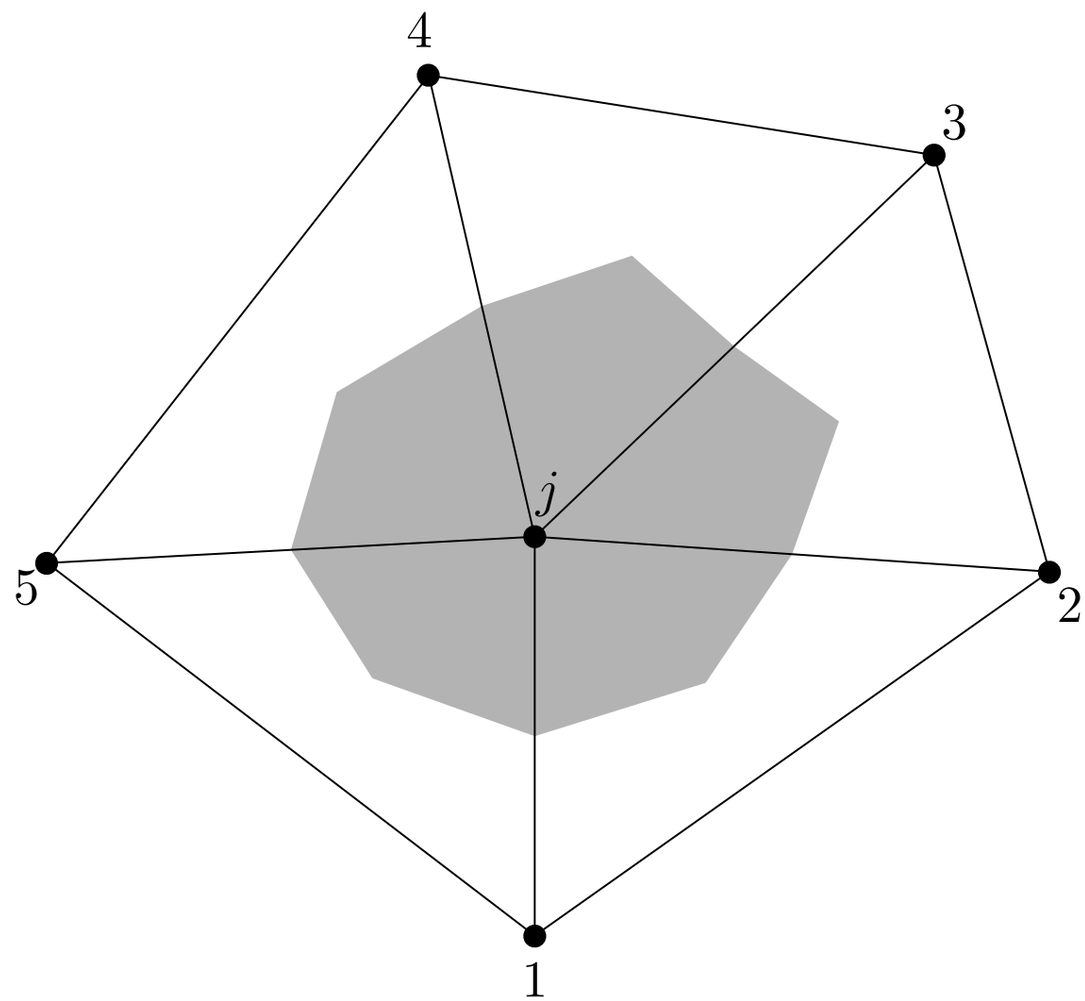
“You Already Have It.”

Free yourself and reset your mindset:
Forget about deriving the formula.
Believe it.
Behold what you have.



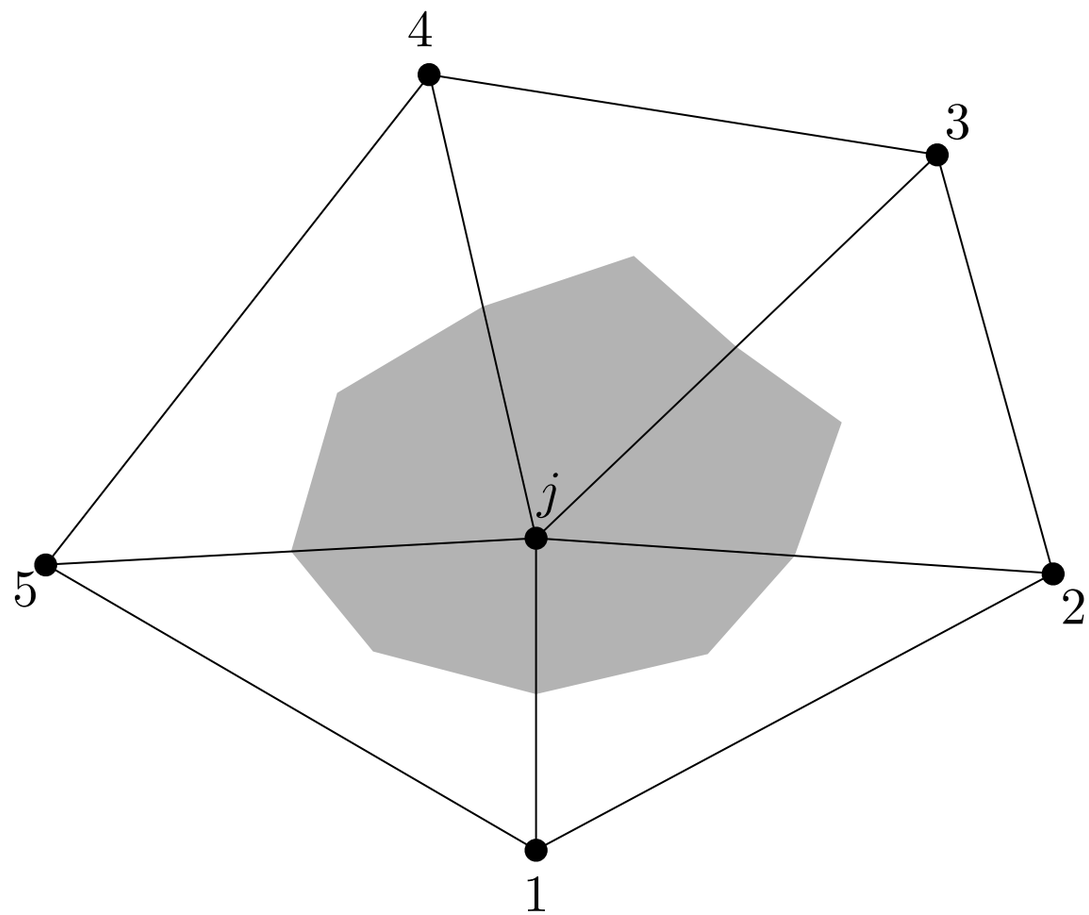
EB scheme is 3rd-order accurate.

$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$



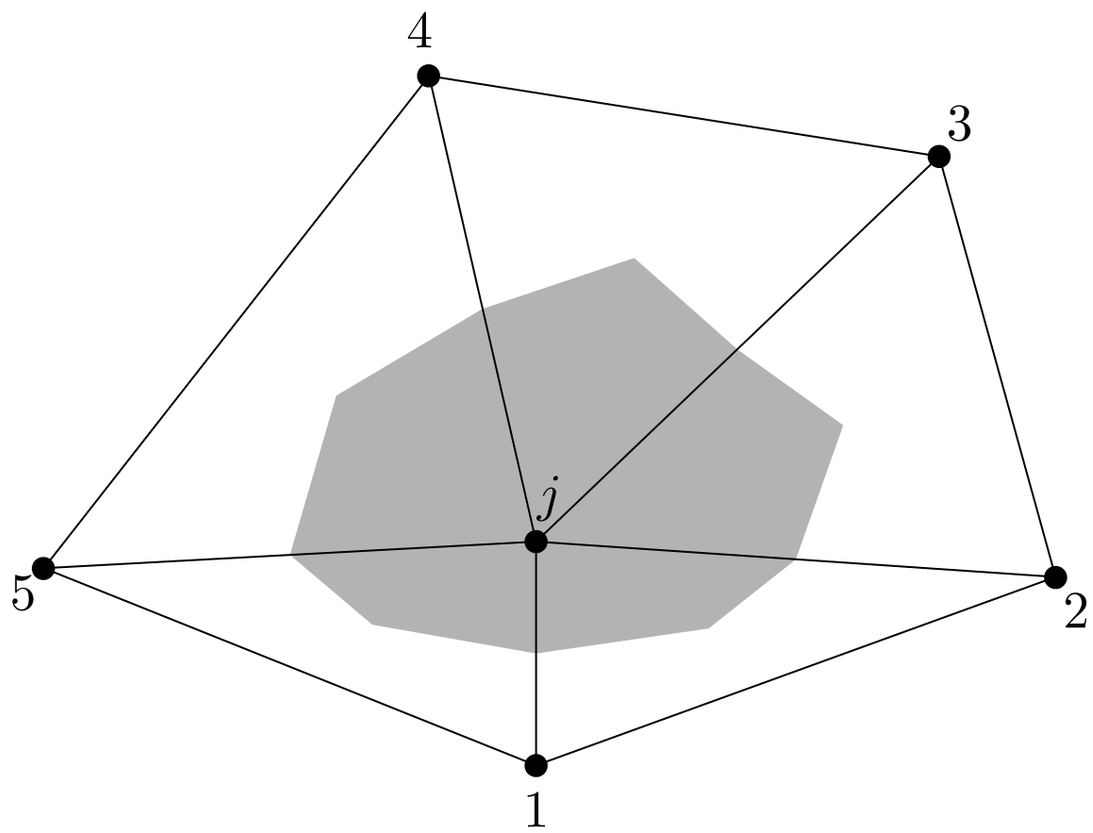
EB scheme is 3rd-order accurate.

$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$



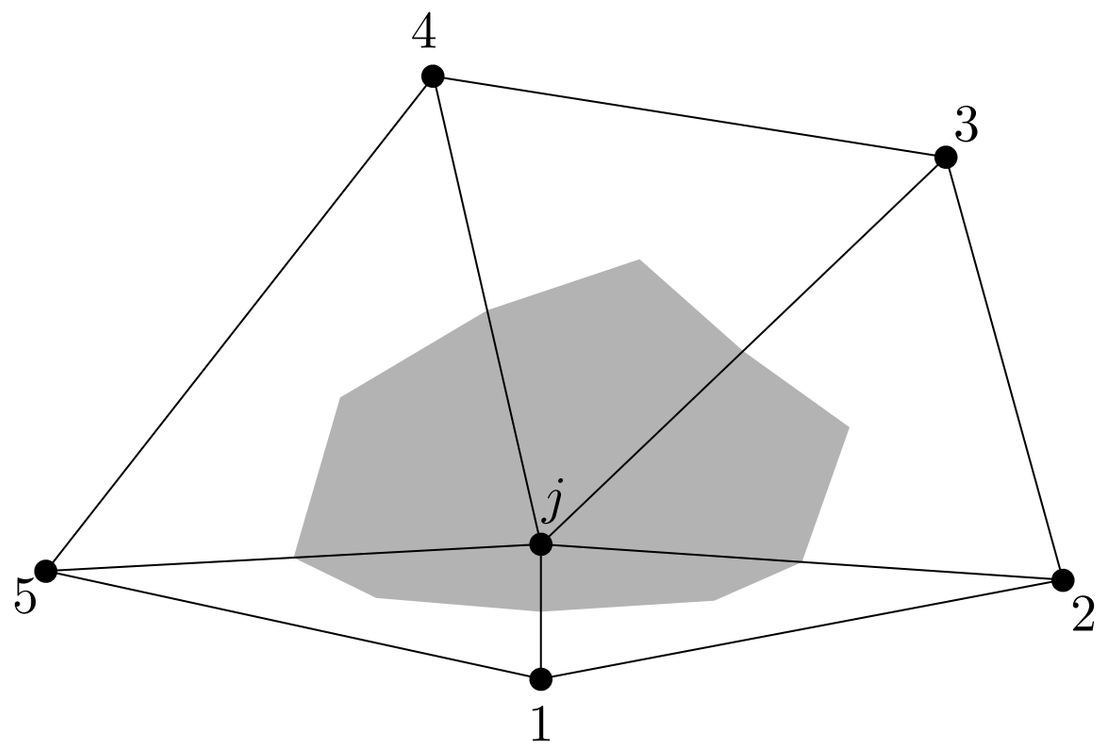
EB scheme is 3rd-order accurate.

$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$



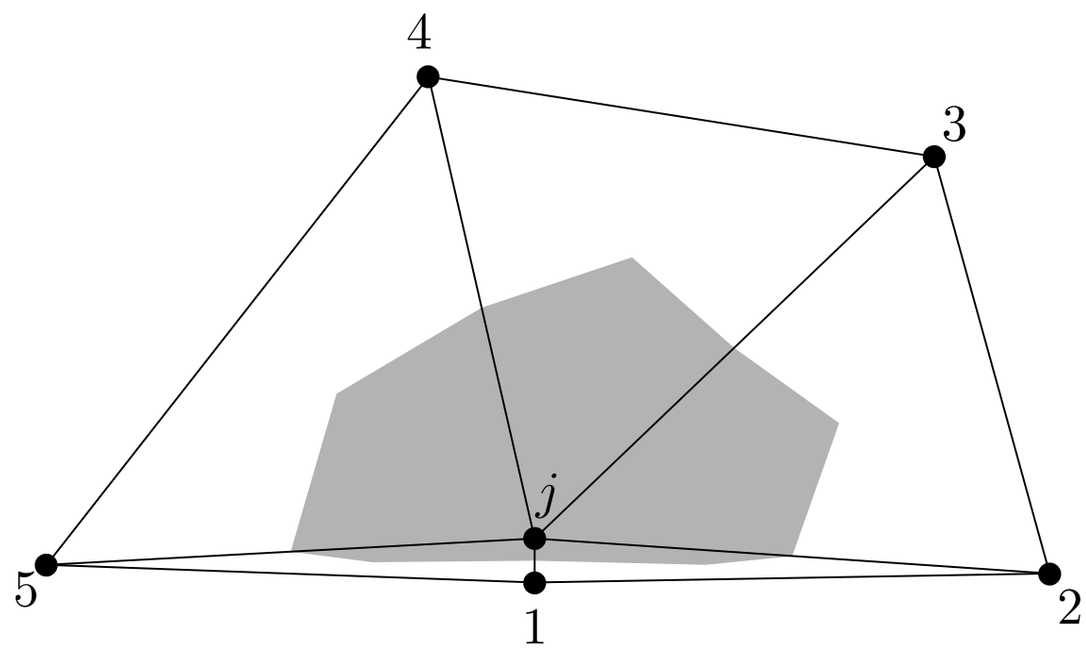
EB scheme is 3rd-order accurate.

$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$



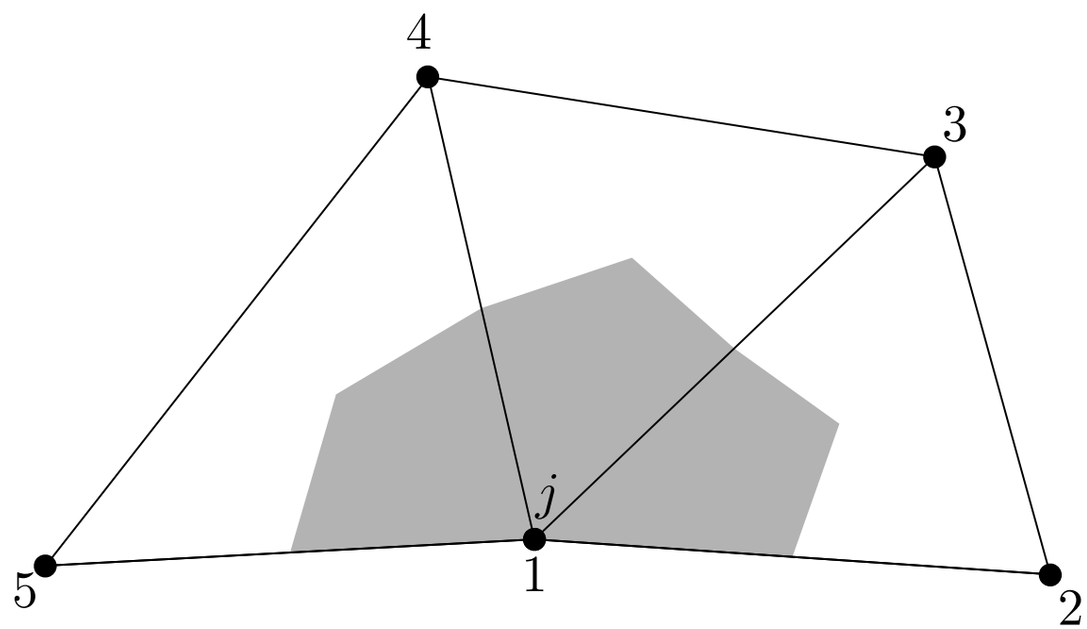
EB scheme is 3rd-order accurate.

$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$



EB scheme is 3rd-order accurate.

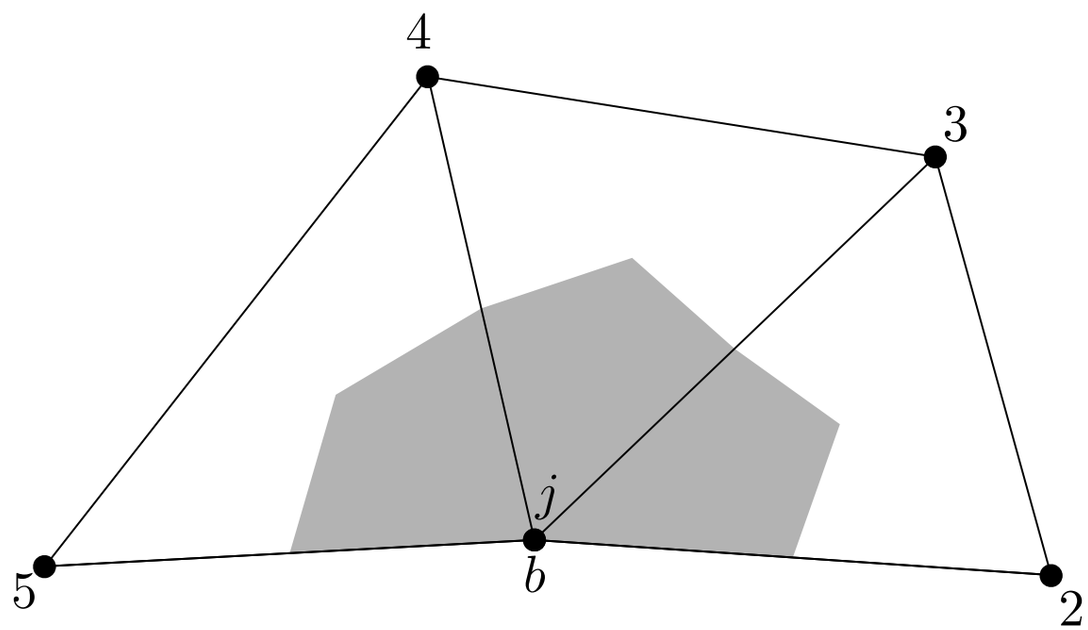
$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$



This is a boundary stencil, and we have 3rd-order!
So, we don't really need a boundary formula!

EB scheme is 3rd-order accurate.

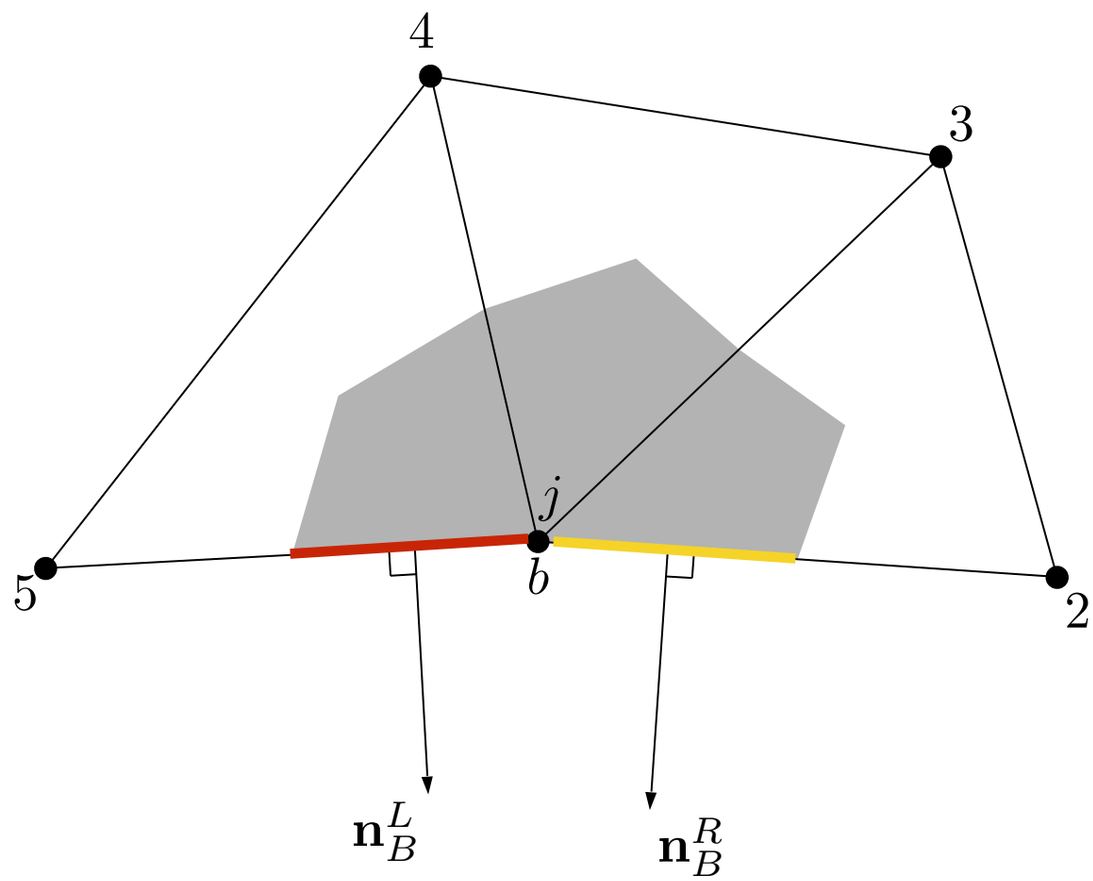
$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$



Node b can be used as a ghost node.

EB scheme is 3rd-order accurate.

$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0$$



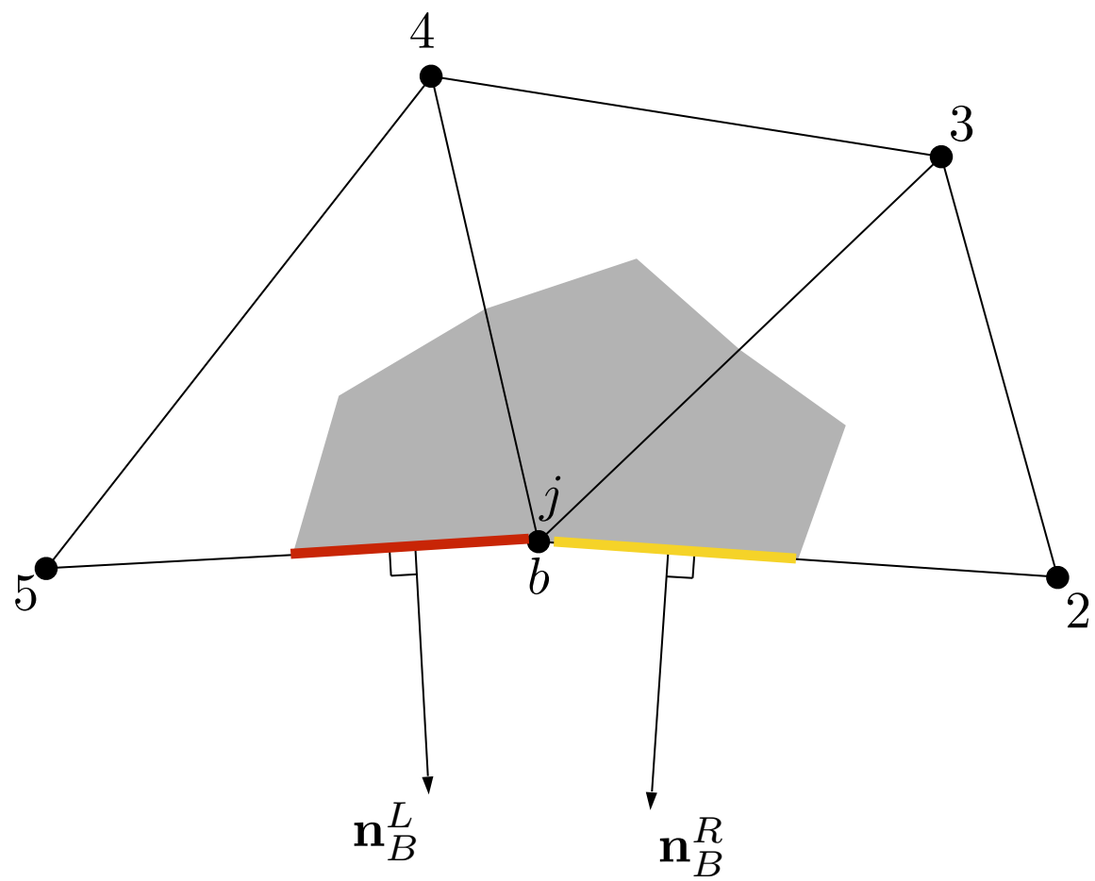
Expanding the 3rd-order EB scheme, we find

$$\sum_{k\in\{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = \phi_{j2}(\mathbf{n}_{j2}^l) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j5}(\mathbf{n}_{j5}^r)$$
$$\underline{+\frac{2}{3}\phi_{jb}\left(\mathbf{n}_B^L\right) + \frac{1}{3}\phi_{j5}\left(\mathbf{n}_B^L\right)} + \underline{\frac{2}{3}\phi_{jb}\left(\mathbf{n}_B^R\right) + \frac{1}{3}\phi_{j2}\left(\mathbf{n}_B^R\right)}$$

Oh, we have a formula!

A General Formula

It preserves 1st/2nd/3rd-order accuracy at boundary nodes.



$$\phi_L(\mathbf{n}_B^L) = \frac{2}{3}\phi_{jb}\left(\mathbf{n}_B^L\right) + \frac{1}{3}\phi_{j5}\left(\mathbf{n}_B^L\right) \qquad \phi_R(\mathbf{n}_B^R) = \frac{2}{3}\phi_{jb}\left(\mathbf{n}_B^R\right) + \frac{1}{3}\phi_{j2}\left(\mathbf{n}_B^R\right)$$

Note: the second terms (j5 and j2 terms) require linear extrapolation.

3rd-order with straight boundary edges.

Remaks

- A general formula derived for tetrahedra (3D).

  See JCP2015, v281, pp518-555

- Curved elements should not be used.

The EB discretization is 3rd-order on linear triangular elements.

Immediately applicable to existing grids (high-order grids not needed).

Other 3rd-order schemes on linear elements:

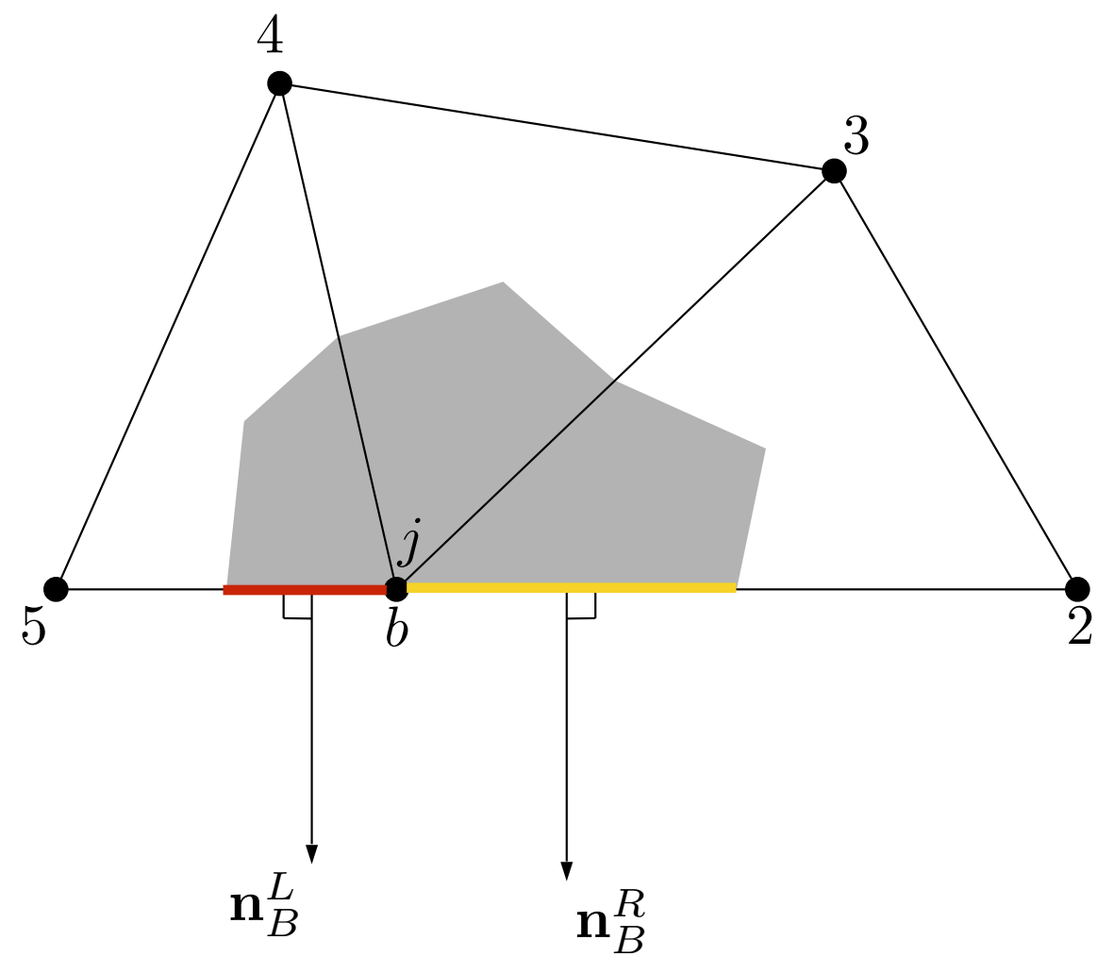
- 3rd-order fluctuation-splitting scheme [AIAA2001-2595]
- 3rd-order LSQ scheme [IJNMF2007]
- 3rd-order residual-distribution scheme [Mazaheri and Nishikawa 2015]

Attractive alternatives to high-order methods in applications for which 3rd-order accuracy is sufficient.

Special Case I: Flat Boundary

$$c = \frac{|\mathbf{n}_B^R|}{|\mathbf{n}_B^L|}$$

$$c = 2$$



General formula becomes

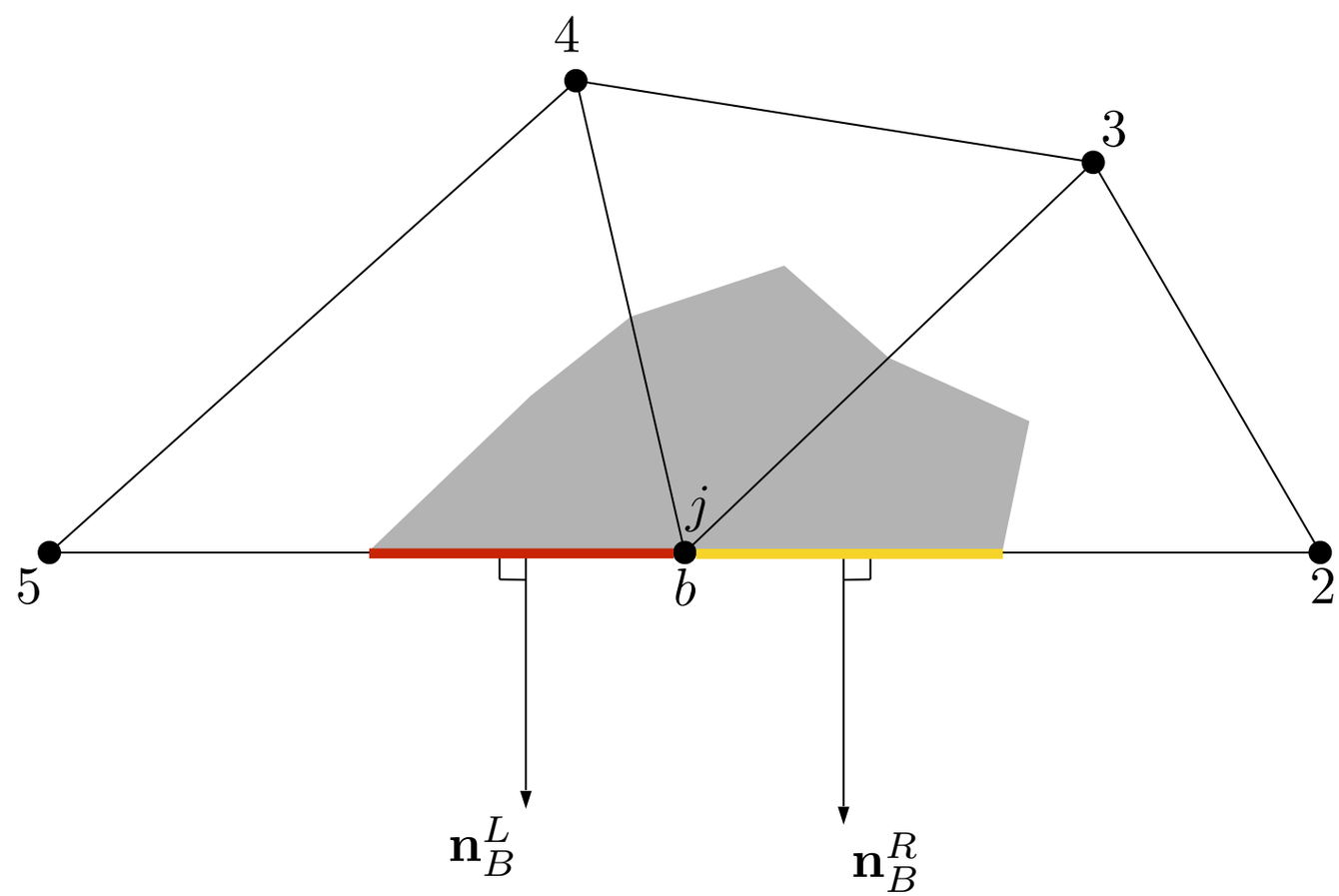
$$\phi_L(\mathbf{n}_B^L) = \tfrac{1}{6}\left[\left(5+\tfrac{1}{c}\right)\phi_j(\mathbf{n}_B^L) + c(1-c)\phi_5(\mathbf{n}_B^L)\right] \quad \phi_R(\mathbf{n}_B^R) = \tfrac{1}{6}\left[(5+c)\,\phi_j(\mathbf{n}_B^R) + \tfrac{c-1}{c^2}\phi_2(\mathbf{n}_B^R)\right]$$

This gives third-order accuracy.

Special Case II: Flat Uniform Boundary

$$c = \frac{|\mathbf{n}_B^R|}{|\mathbf{n}_B^L|}$$

$$c = 1$$



$$\phi_L(\mathbf{n}_B^L) = \phi_j(\mathbf{n}_B^L) \qquad \phi_R(\mathbf{n}_B^R) = \phi_j(\mathbf{n}_B^R)$$

This gives third-order accuracy.

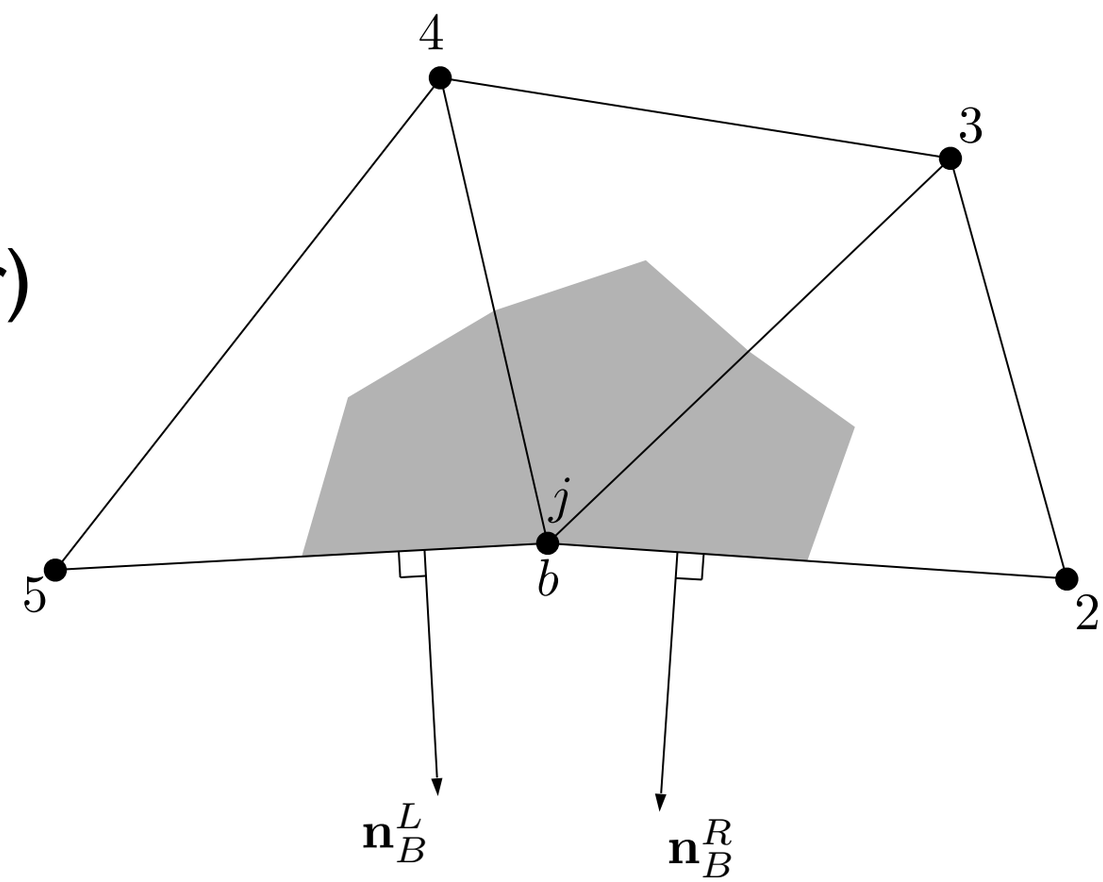
Boundary Formulas

- **One-point** formula (1st-order, 3rd-order for flat uniform boundary)

$$\phi_L(\mathbf{n}_B^L) = \phi_j(\mathbf{n}_B^L)$$

- **Two-point** formula (2nd-order)

$$\phi_L(\mathbf{n}_B^L) = \frac{5}{6}\phi_j(\mathbf{n}_B^L) + \frac{1}{6}\phi_5(\mathbf{n}_B^L)$$



- **General** formula (3rd-order)

$$\phi_L(\mathbf{n}_B^L) = \frac{2}{3}\phi_{jb}\left(\mathbf{n}_B^L\right) + \frac{1}{3}\phi_{j5}\left(\mathbf{n}_B^L\right)$$

Results

Numerical Results

- Third-order results (see JCP2015 for 2nd-order results)
- Burgers, advection-diffusion, Laplace
- All equations solved as a steady conservation law
- 10 Irregular triangular grids: 1089 to 409,600 nodes.
- Residuals reduced by 10 orders in all cases.
- Errors measured at interior and boundary nodes

Burgers' Equation

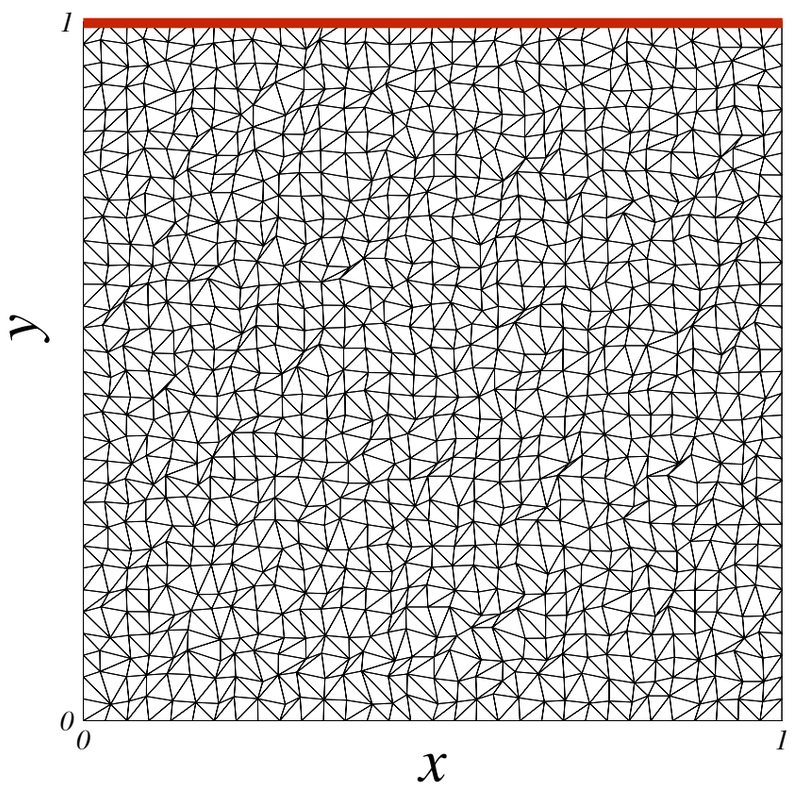
$$\partial_x f + \partial_y g = s(x, y)$$

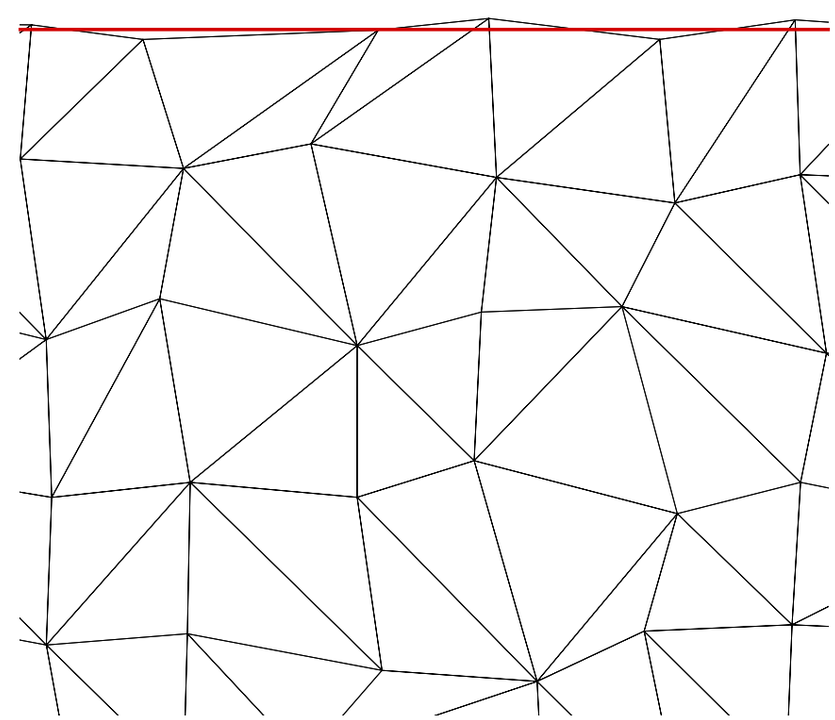
where $(f, g) = (u^2/2, u)$ and $s(x, y) = (1 + \sin(x - y)) \cos(x - y)$

Exact solution: $u(x, y) = \sin(x - y) + 2$

3rd-order EB scheme applied in the steady conservation form.

[JCP2012]

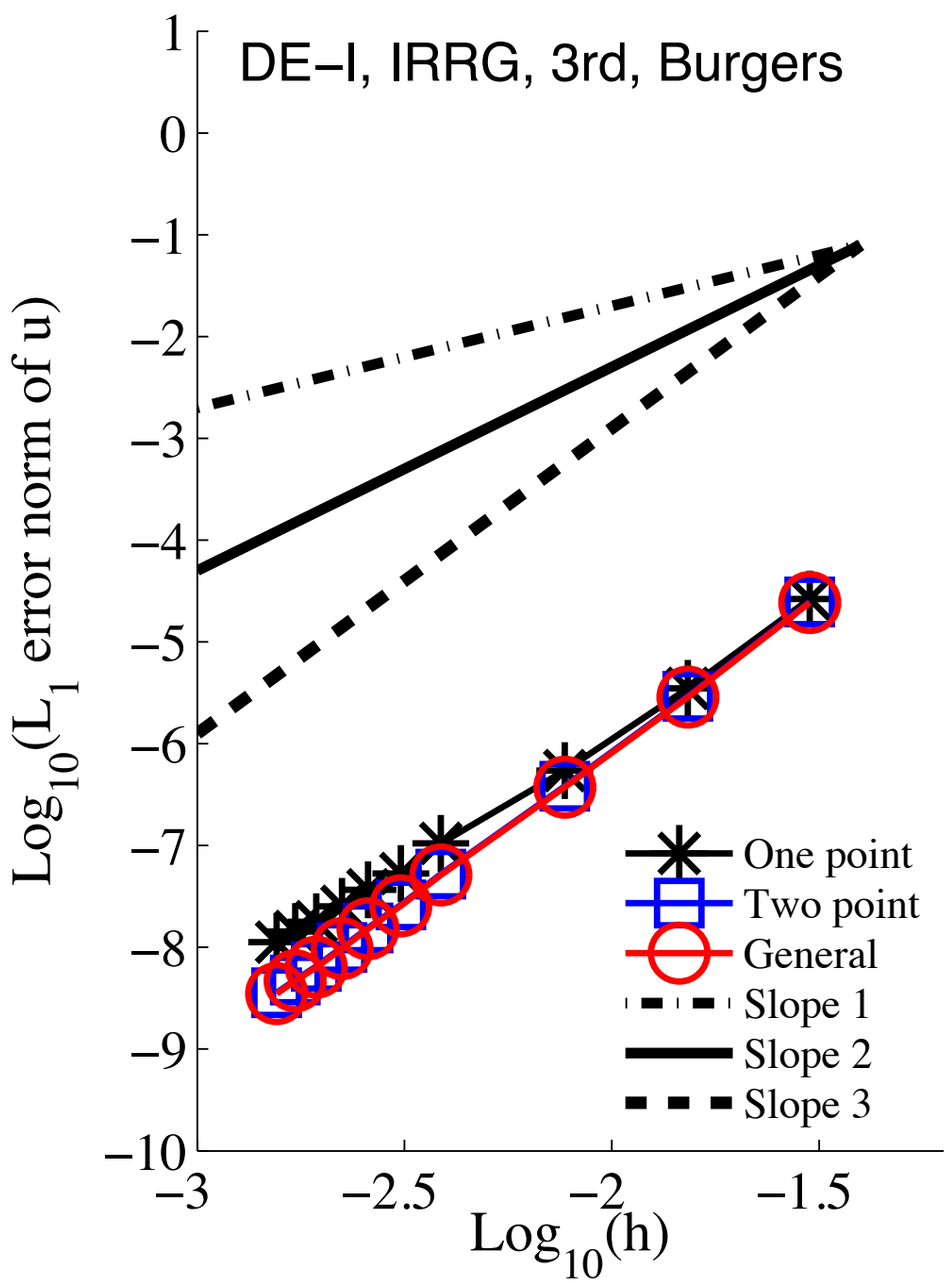




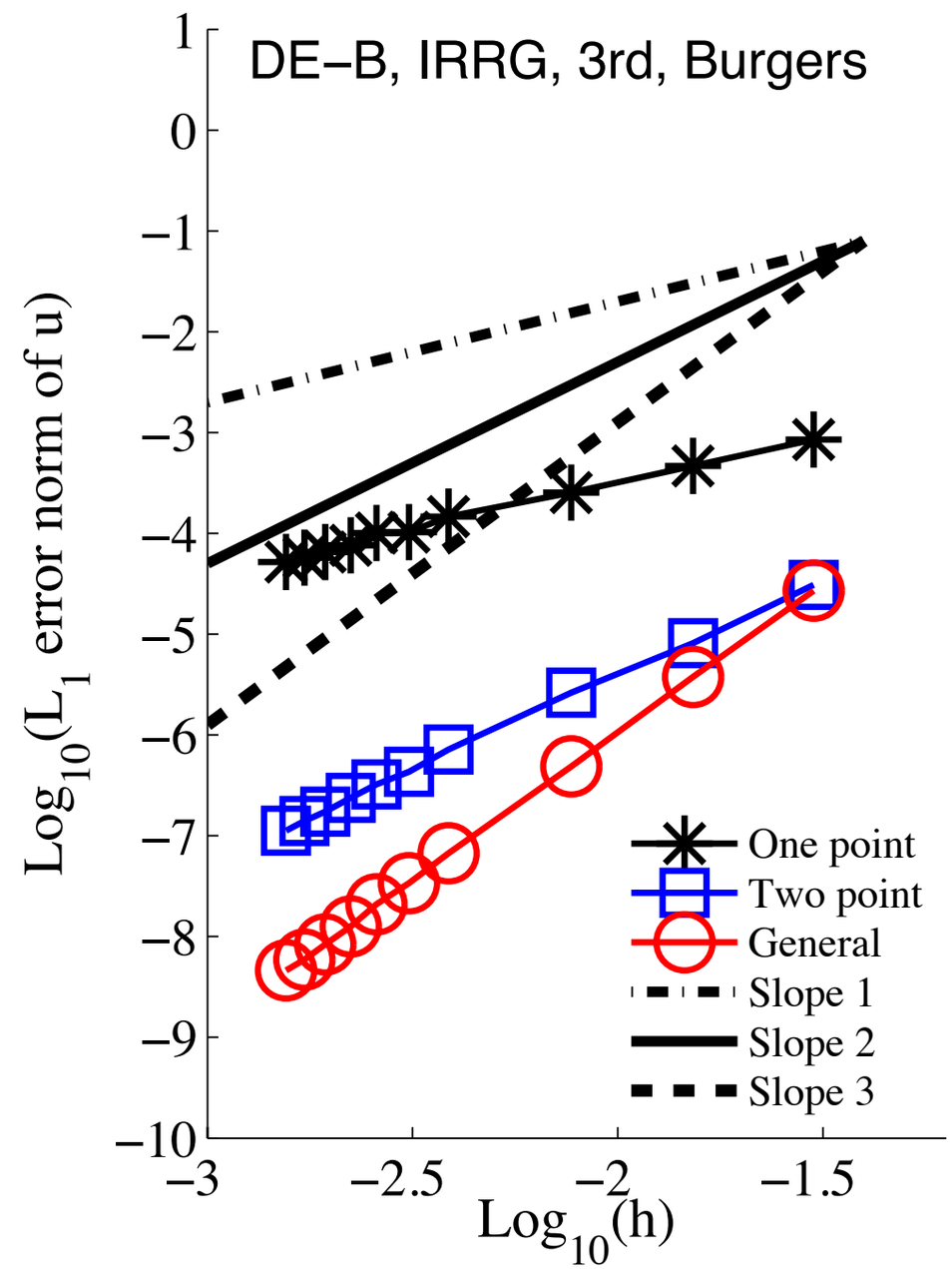
- Weak condition at top boundary (outflow); exact solution imposed elsewhere.
- Errors measured at interior nodes and boundary nodes separately.

Burgers' Equation

Interior Nodes



Boundary Nodes



Error doesn't propagate back into the domain.

Advection-Diffusion

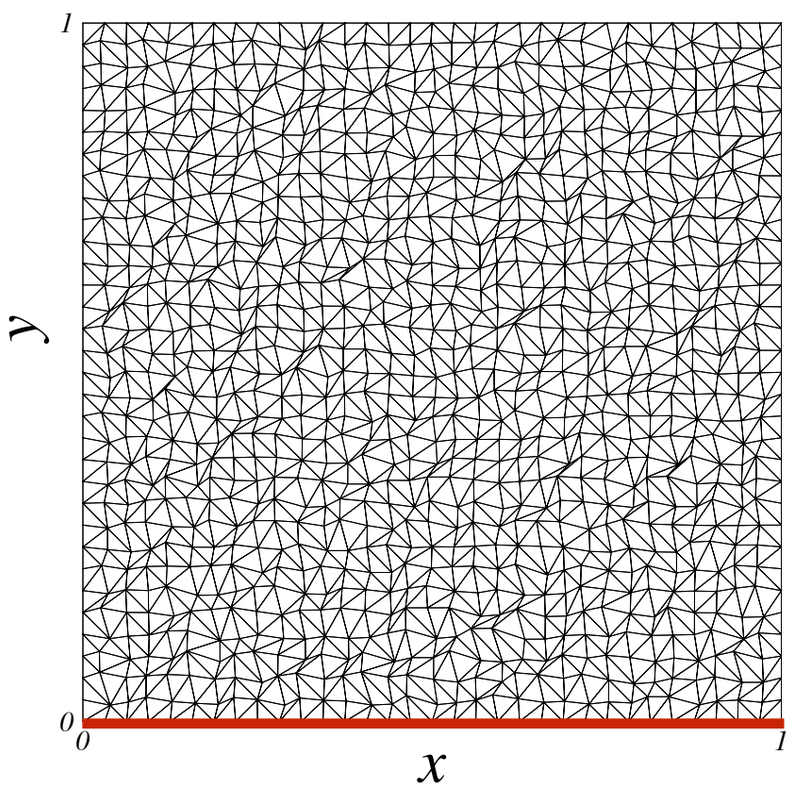
$$a\,\partial_x u + b\,\partial_y u = \nu(\partial_{xx} u + \partial_{yy} u)$$

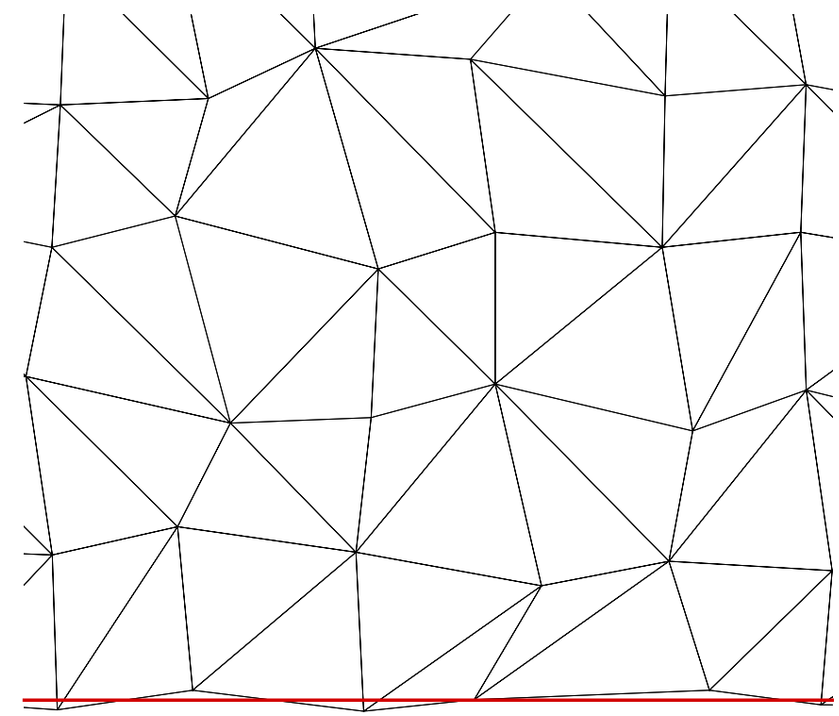
$$(a, b) = (1.23, 0.12)$$

$$\nu = \frac{\sqrt{a^2 + b^2}}{Re} \quad Re = 10$$

3rd-order EB scheme applied in the steady conservation form.

[JCP2014]



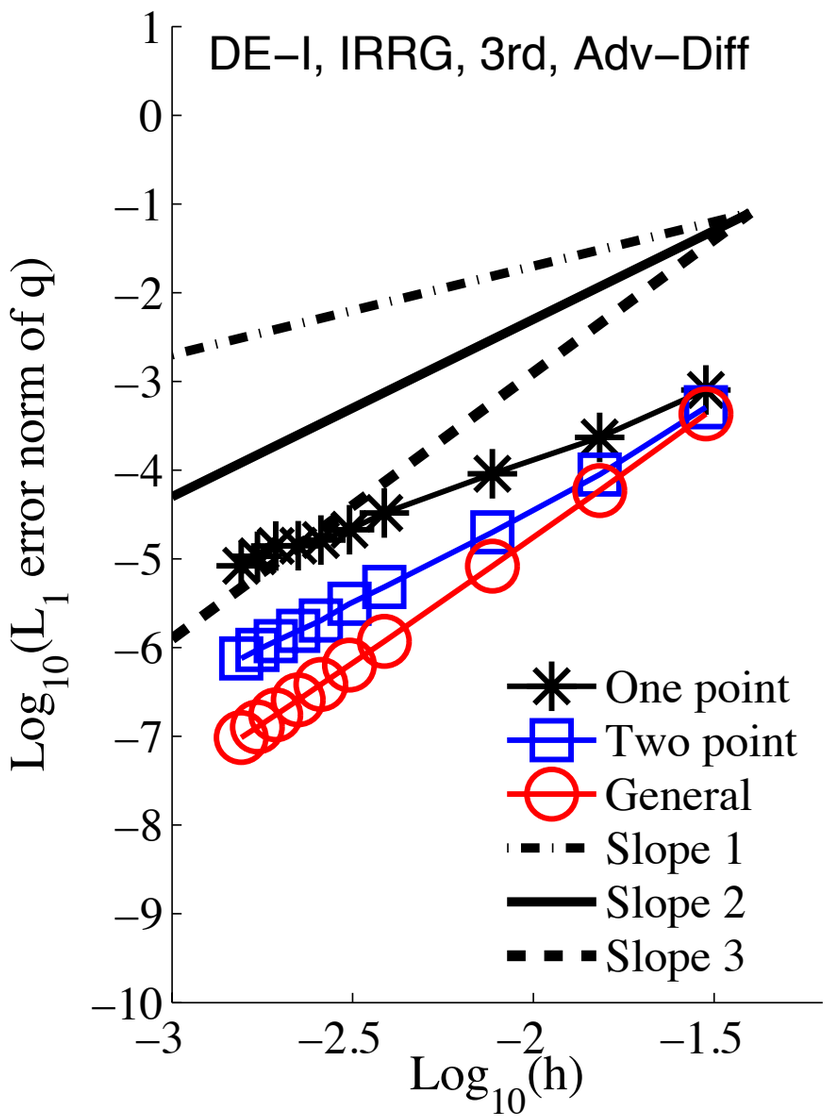


- Weak condition at bottom boundary: specify (u,p) and compute q(=uy). the exact solution imposed elsewhere.
- Errors measured in q at interior and boundary nodes separately.

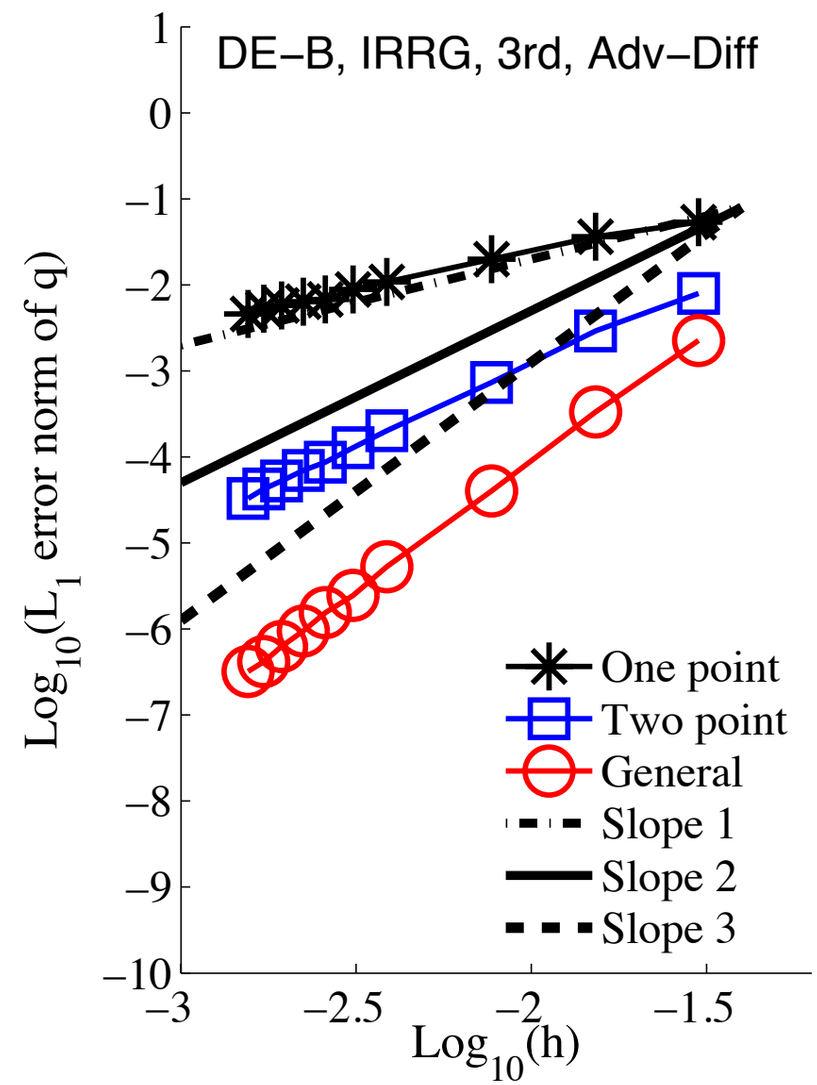
Advection-Diffusion

Isotropic Grids

Interior Nodes



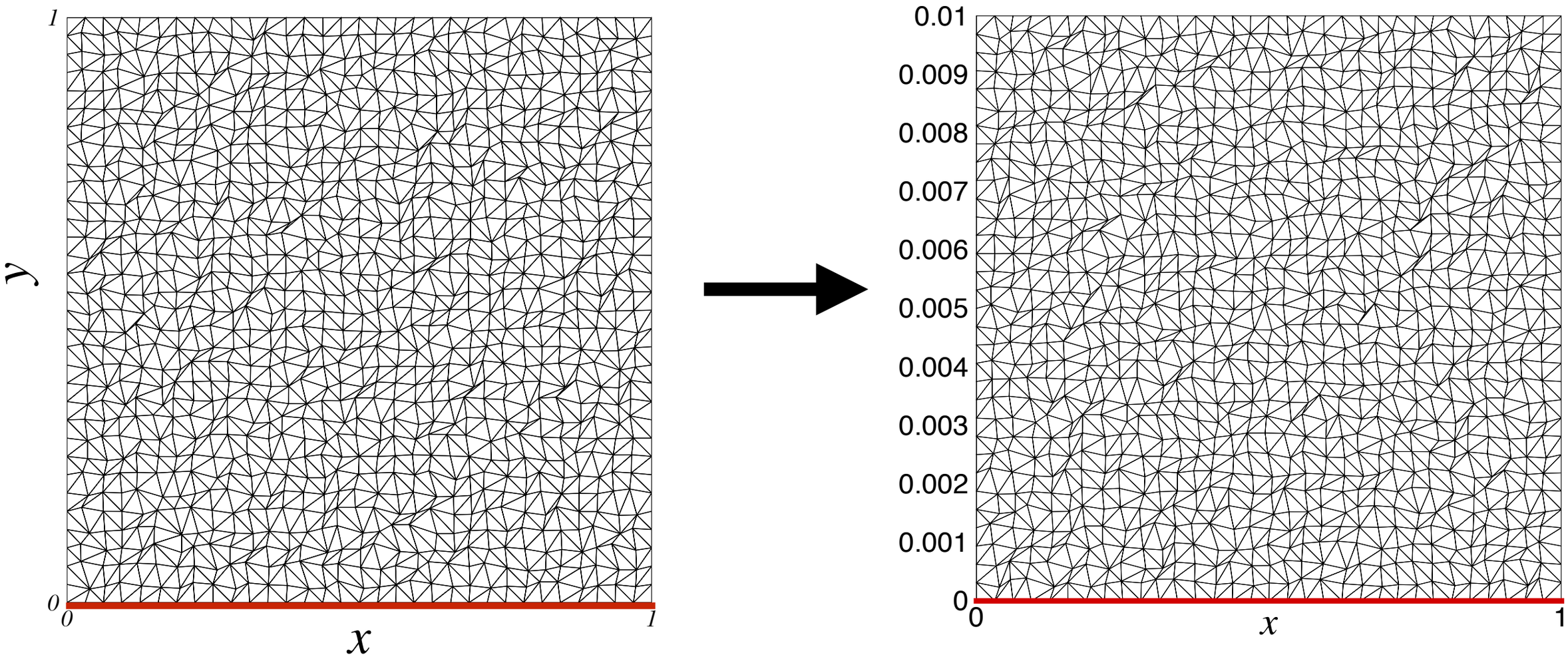
Boundary Nodes



Boundary formula affects both boundary and interior: 1st/2nd/3rd-order with one/two- and general formula.

Advection-Diffusion

Anisotropic Grids

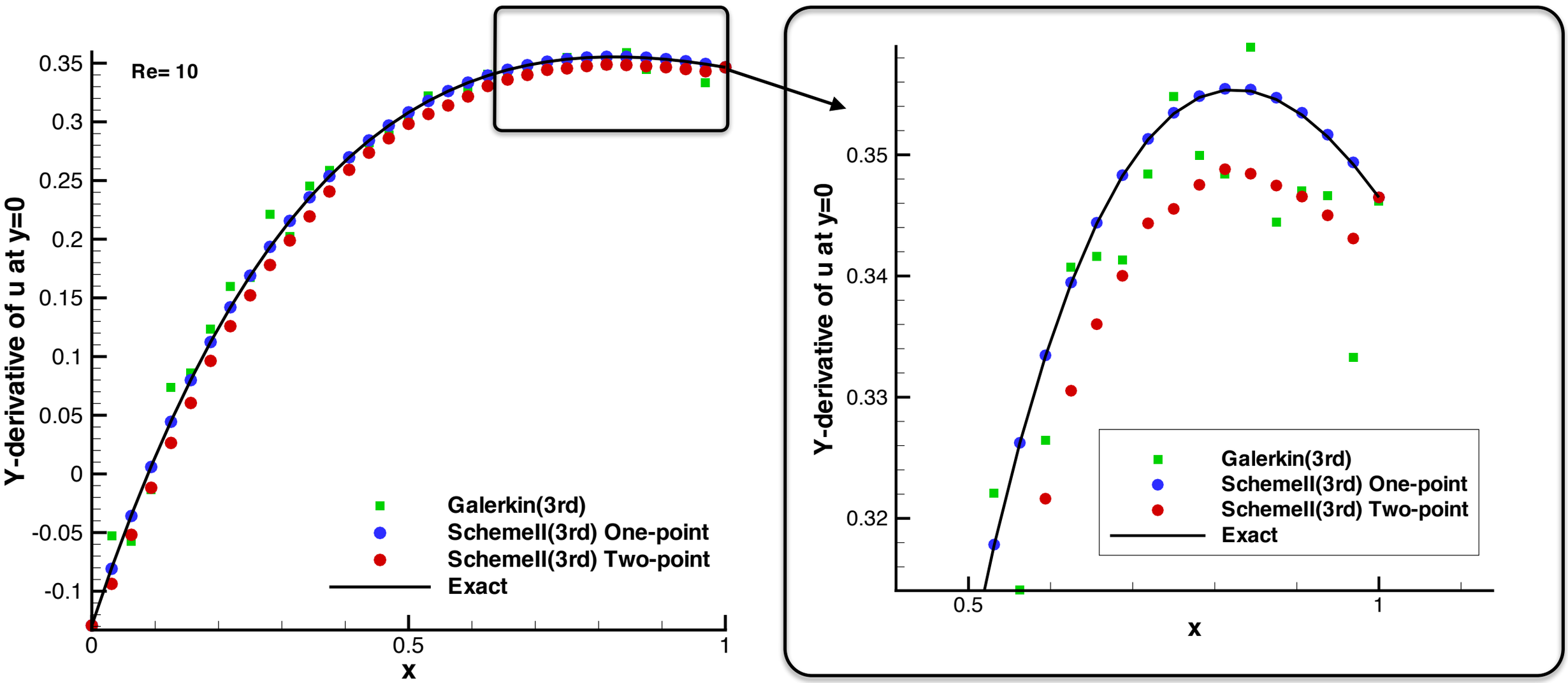


Flat uniform boundary grid:

- **One-point** formula: 3rd-order for flat uniform boundary
- **Two-point** formula: 2nd-order

Advection-Diffusion

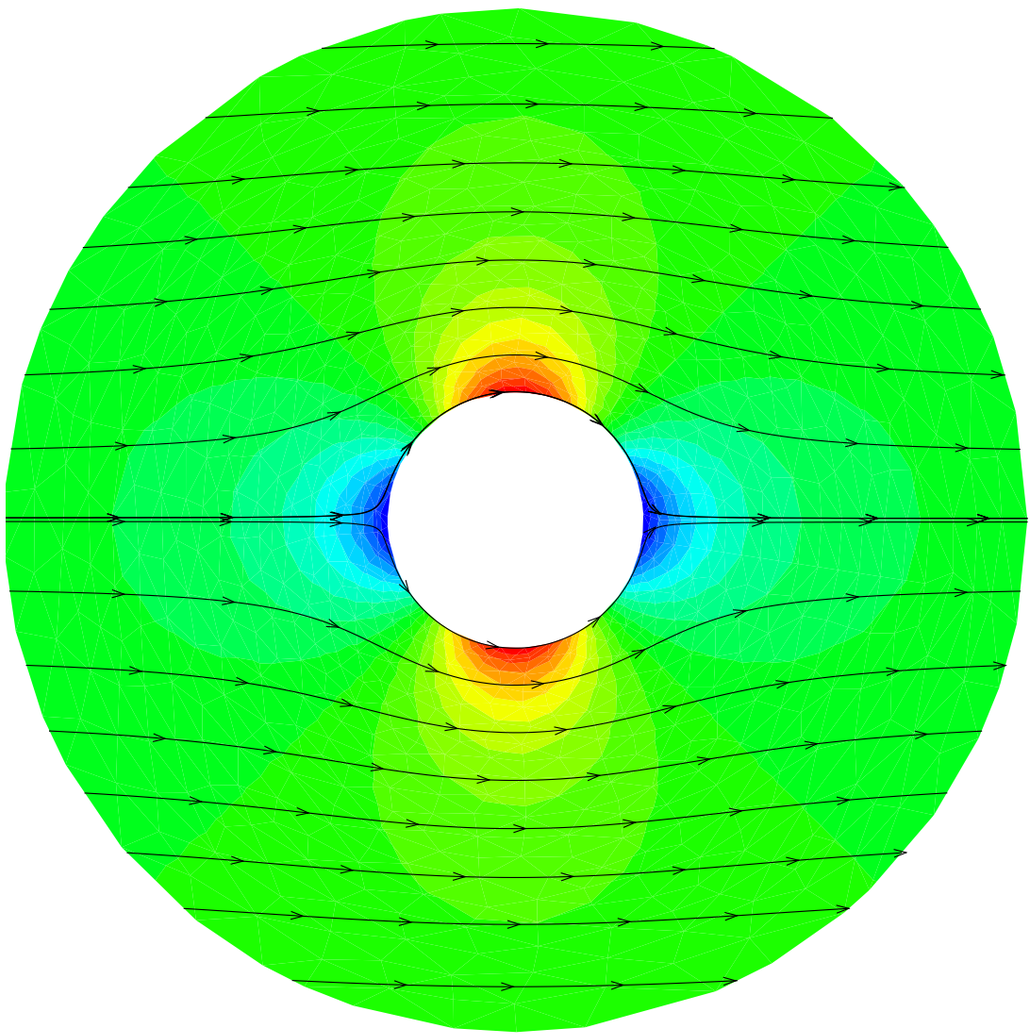
Anisotropic Grids



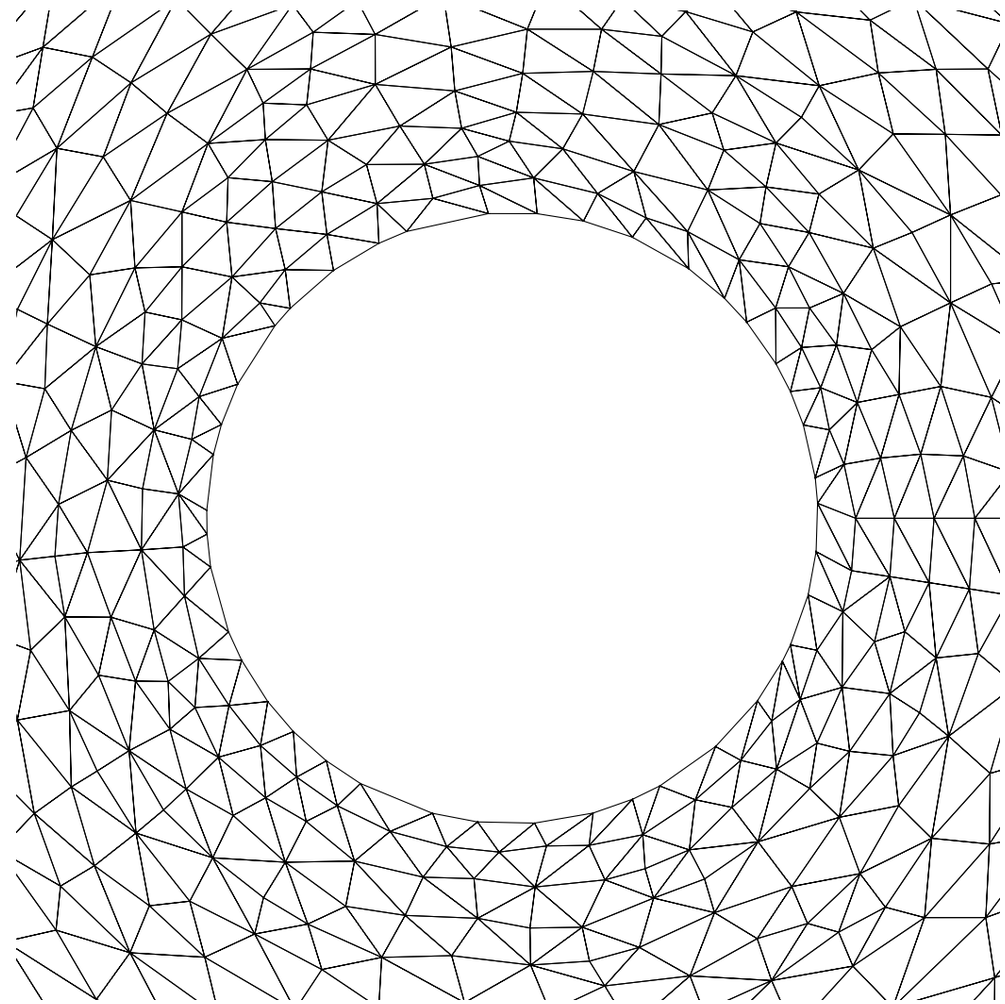
Third-order boundary formula is essential.

Curved Boundary Problem

Potential flow over a cylinder



Exact solution



Fully irregular grid

Governing Equation

Laplace equation for the stream function

$$\partial_{xx}\psi + \partial_{yy}\psi = 0$$

Third-order EB discretization is applied in the steady conservation form. [JCP2014]

$$\partial_x \mathbf{F} + \partial_y \mathbf{G} = \mathbf{0} \qquad \mathbf{U} = [u, p(= \partial_x u), q(= \partial_y u)]$$

Extra variables, p and q, correspond to the velocity components.

$$u = \partial_y \psi = q \qquad v = -\partial_x \psi = -p$$

Third-order EB scheme produces 3rd-order accurate (u,v).

Boundary Condition

Strong and weak conditions

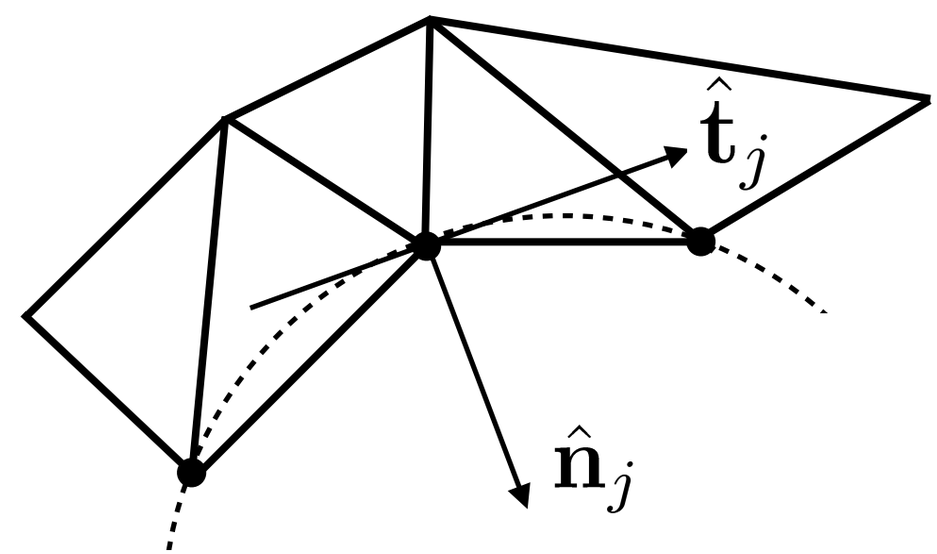
1. Outer boundary: Specify the exact solution.

2. Cylinder (slip wall)

$$\psi_j = 0$$

$$(q_j, -p_j) \cdot \hat{\mathbf{n}}_j = 0$$

$$(Res_j(3), -Res_j(2)) \cdot \hat{\mathbf{t}}_j = 0$$



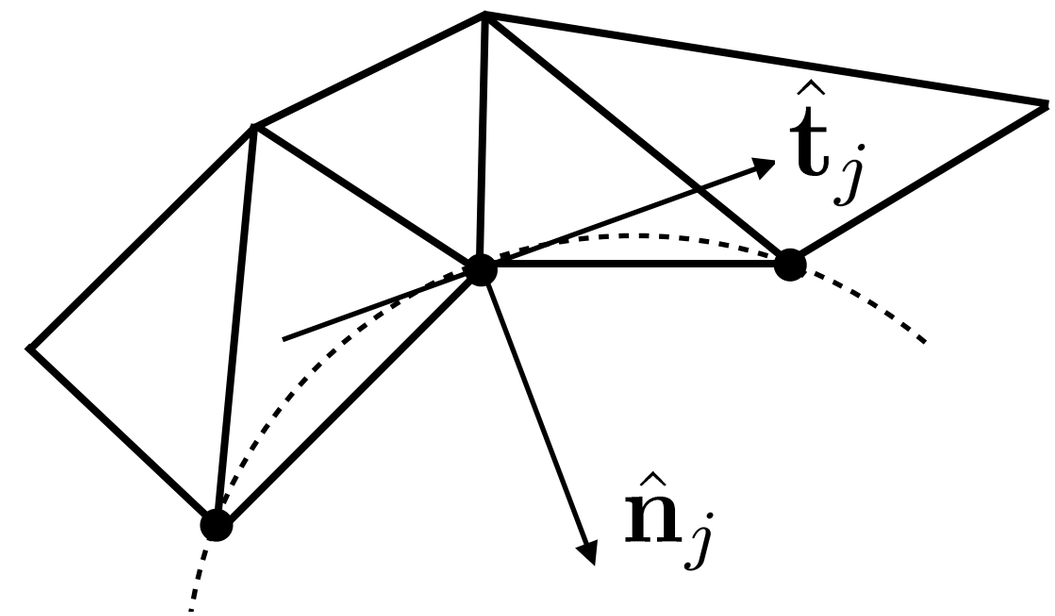
The last equation approximates $(\partial_y \psi, -\partial_x \psi) \cdot \hat{\mathbf{t}}_j - (q, -p) \cdot \hat{\mathbf{t}}_j = 0$

This is where the boundary flux is required.

Normal and Tangent Vectors

1. Linear Approximation

$$\hat{\mathbf{n}}_j = \frac{(y_{j+1} - y_{j-1}, x_{j-1} - x_{j+1})}{\sqrt{(x_{j+1} - x_{j-1})^2 + (y_{j+1} - y_{j-1})^2}}$$



2. Quadratic Approximation

Quadratic interpolation over 3 nodes in the parameter space of edge-length, s.

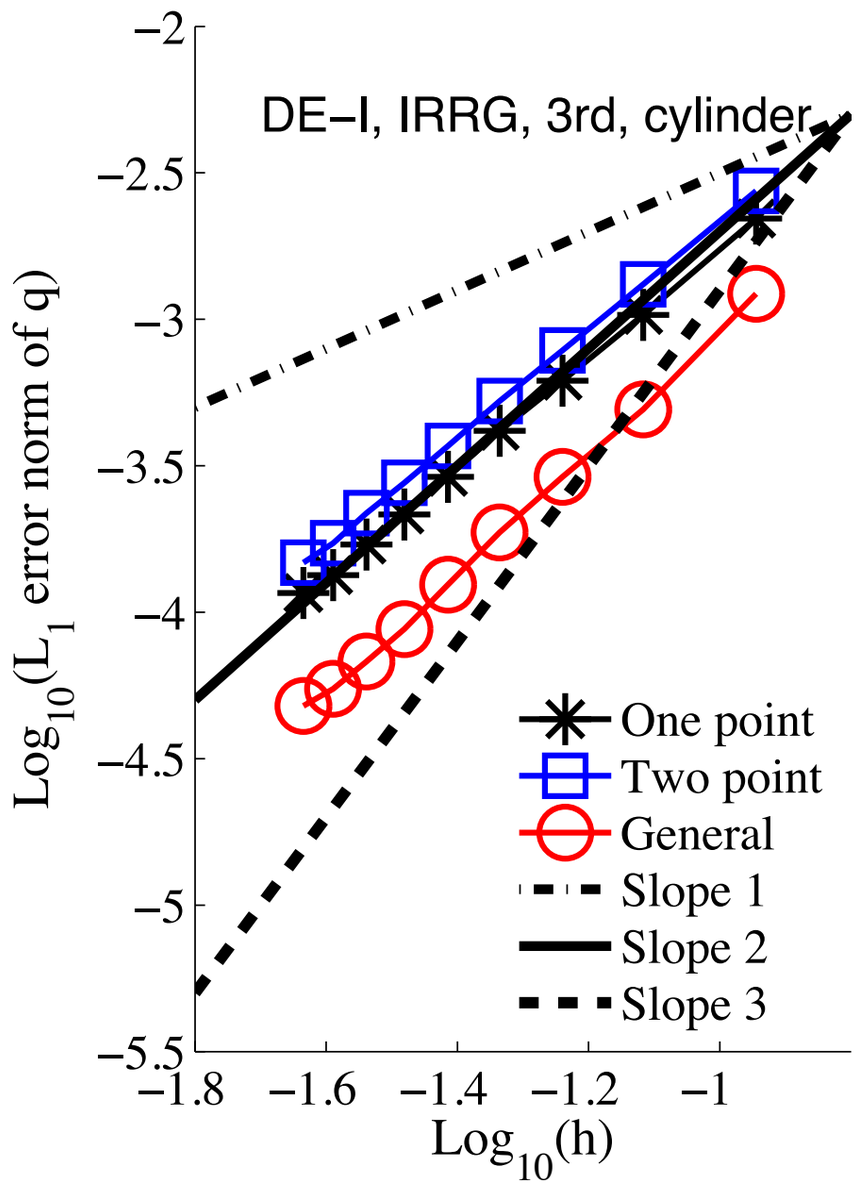
$$\hat{\mathbf{n}}_j = \frac{(dy/ds, -dx/ds)_j}{\sqrt{(dx/ds)_j^2 + (dy/ds)_j^2}}$$

NOTE:
- This is reconstruction. No analytical geometry is required.
- Geometrical singularities (e.g., corner) would require a special treatment.

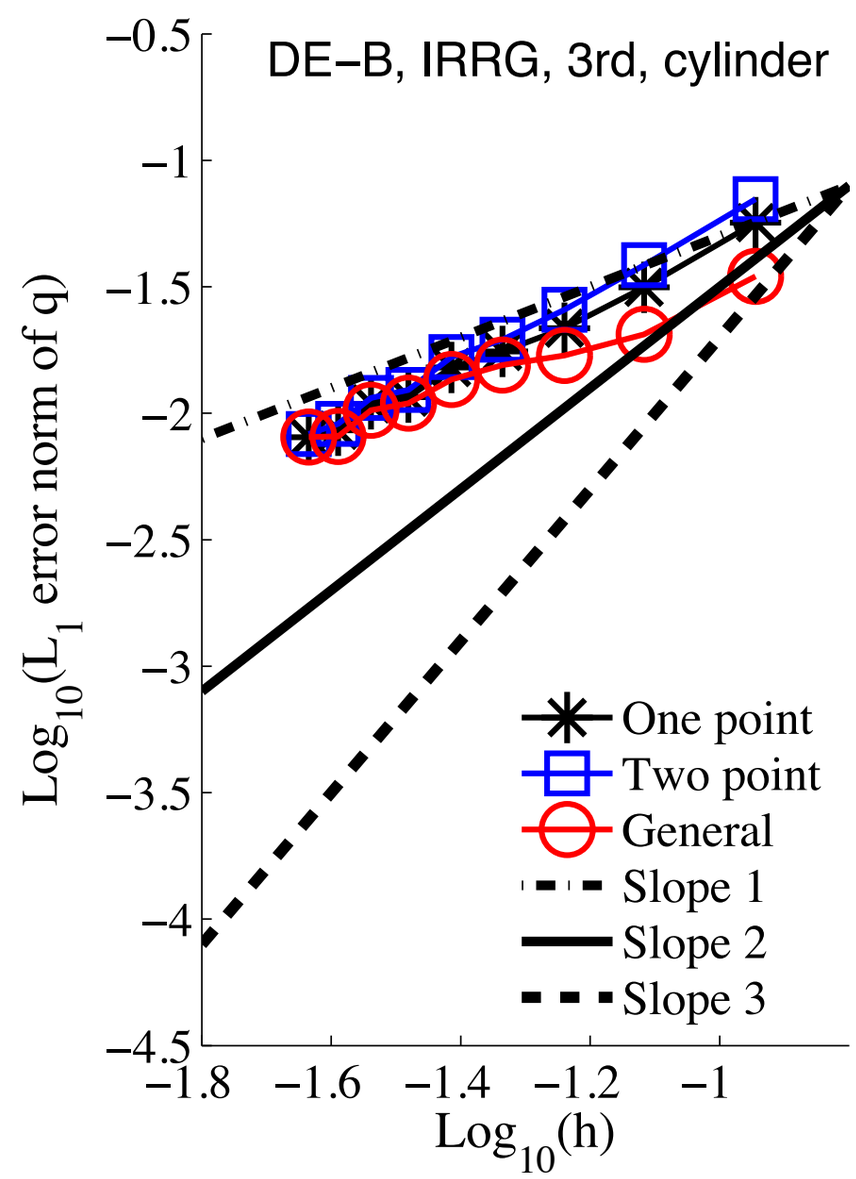
Linear Approximation

Discretization error in the x-velocity (u)

Interior Nodes



Boundary Nodes

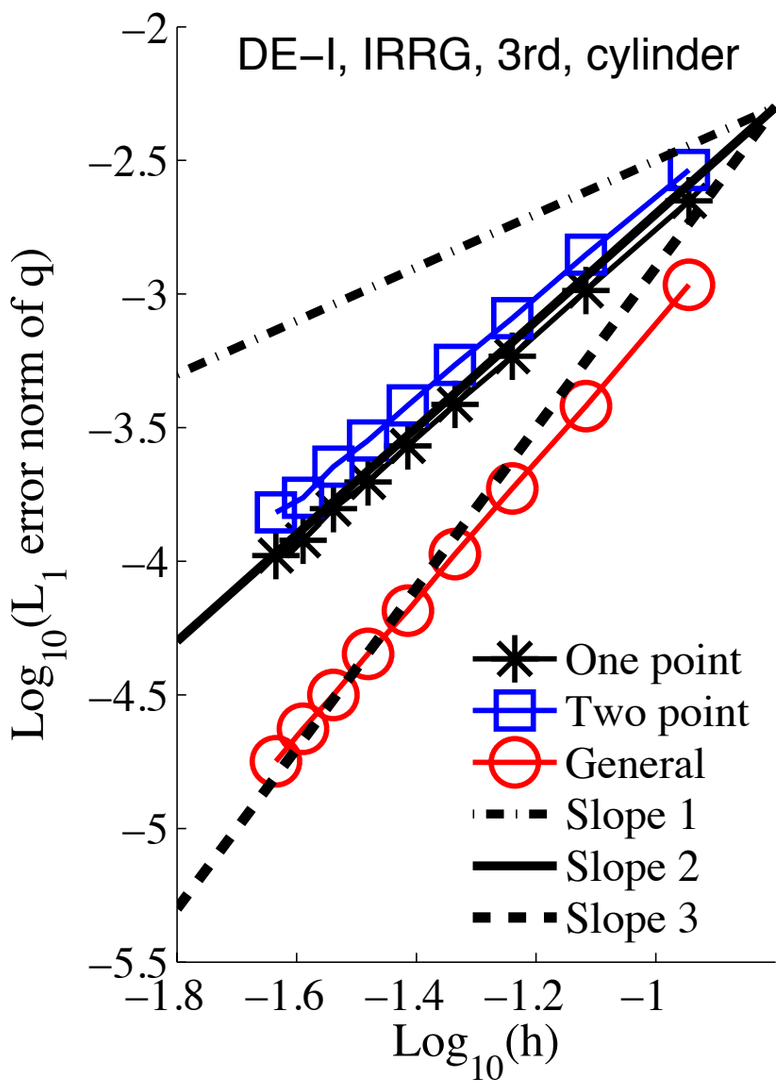


No boundary formula gives 3rd-order accuracy

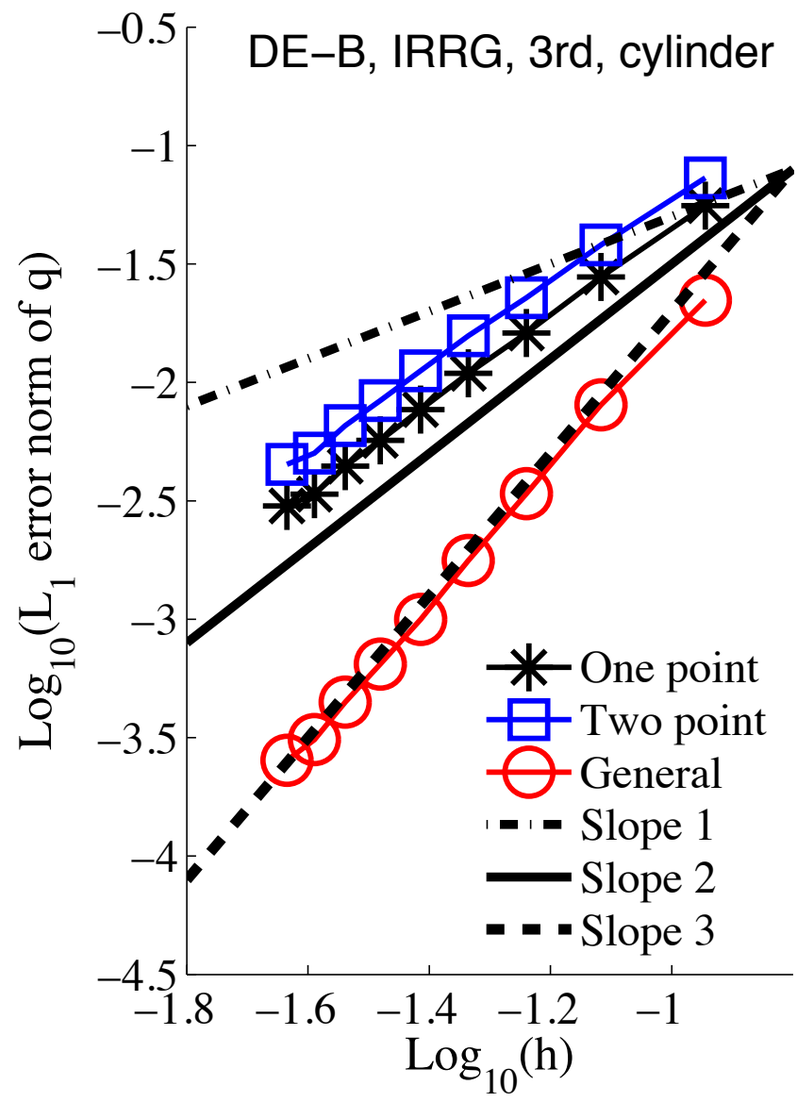
Quadratic Approximation

Discretization error in the x-velocity (u)

Interior Nodes



Boundary Nodes

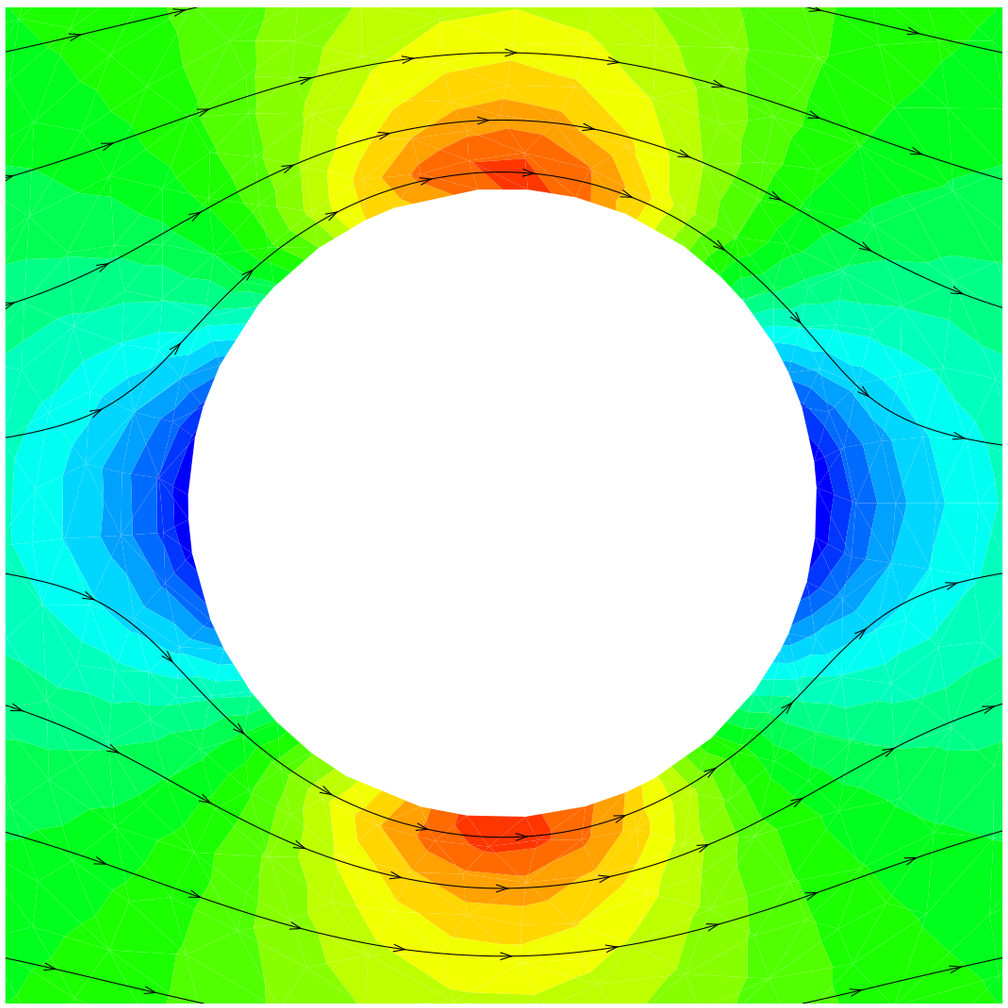


Only the general formula gives 3rd-order accuracy.

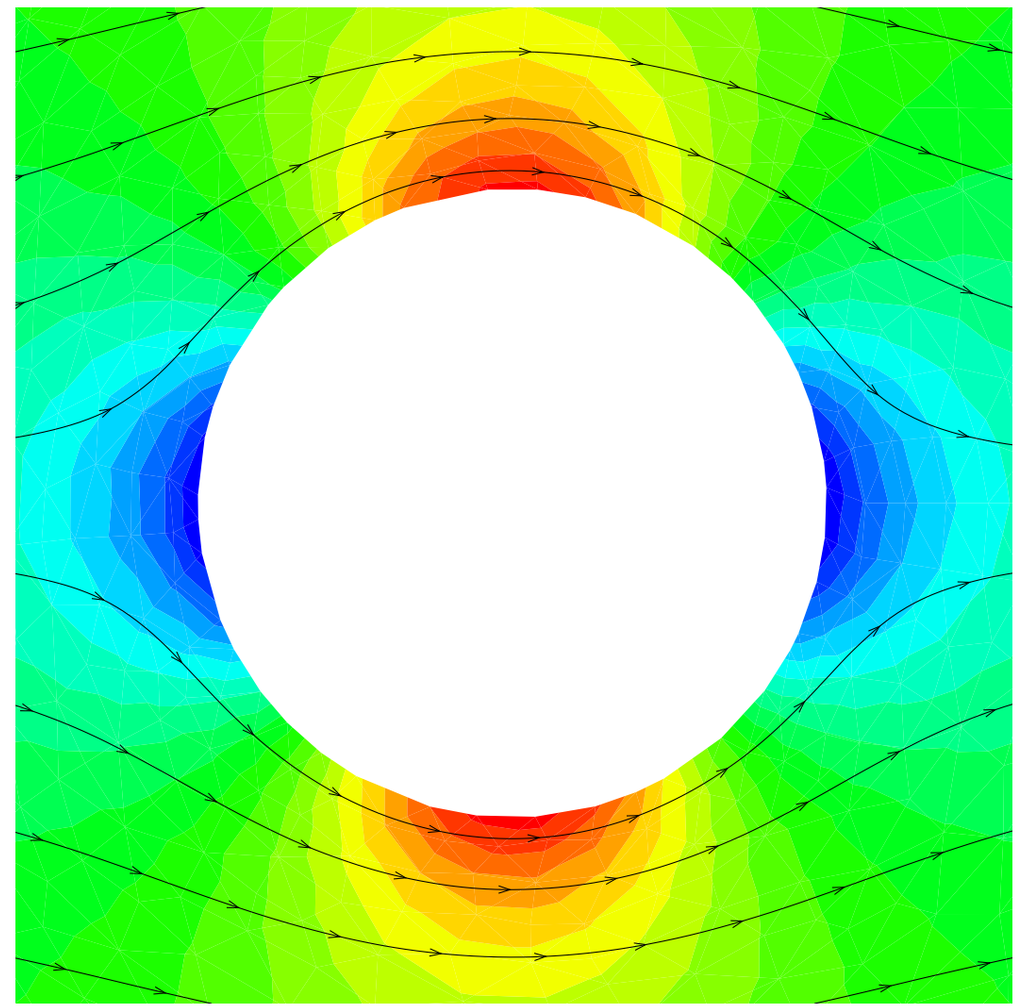
Quadratic Approximation

Contours of the x-velocity (u)

Two-point formula



General Formula



The quadrature formula has a significant impact on the solution.

Conclusions

- Edge-based scheme is 1st/2nd/3rd-order through boundary nodes.

- We already had a general boundary quadrature formula.

- Numerical flux should NOT be exact for quadratic fluxes

- 3rd-order accurate without curved elements.

- Accurate normal vectors required for 3rd-order (quadratic interpolation)

See JCP2015, v281, pp518-555 for details.

See AIAA2014-2091 for NS results.

Don’t worry,
if Santa Claus doesn’t bring you a present.

Happy Holidays!