A New Cell-Vertex Reconstruction Method for Finite Volume Schemes

R. Costa[1], S. Clain[1,2], G.J. Machado[1]

[1] Universidade do Minho, Centro de Matemática, Campus de Azurém, Guimarães, Portugal

[2] Université Paul Sabatier, Institut de Mathématiques de Toulouse, Toulouse, France

SHARK-FV 2014 Conference

SHARING HIGHER-ORDER ADVANCED RESEARCH KNOW-HOW on FINITE VOLUME

Ofir, Portugal

April 28 - May 2, 2014

This research was financed by FEDER Funds through Programa Operacional de Fatores de Competitividade – COMPETE and by Portuguese Funds FCT – Fundação para a Ciência e a Tecnologia, within the Projects PEst-C/MAT/UI0013/2011 and PTDC/MAT/121185/2010

Outline

Motivations

- Why a second-order scheme?

Motivations

- Why a second-order scheme?
  - ☺ Quite simple and easy to code

Motivations

- Why a second-order scheme?
  - ☺ Quite simple and easy to code
  - ☺ Easy dissemination in a broaden scientific community

Motivations

- Why a second-order scheme?
  - ☺ Quite simple and easy to code
  - ☺ Easy dissemination in a broaden scientific community
  - ☺ Many practical problems do not require complex and accurate methods

Motivations

- Why a second-order scheme?
  - ☺ Quite simple and easy to code
  - ☺ Easy dissemination in a broaden scientific community
  - ☺ Many practical problems do not require complex and accurate methods
  - ☺ High-order methods can be very time and memory consuming

Motivations

- Why a second-order scheme?
  - ☺ Quite simple and easy to code
  - ☺ Easy dissemination in a broaden scientific community
  - ☺ Many practical problems do not require complex and accurate methods
  - ☺ High-order methods can be very time and memory consuming
- Why a cell-vertex method?

Motivations

- Why a second-order scheme?
  - ☺ Quite simple and easy to code
  - ☺ Easy dissemination in a broaden scientific community
  - ☺ Many practical problems do not require complex and accurate methods
  - ☺ High-order methods can be very time and memory consuming
- Why a cell-vertex method?
  - ☺ More degrees of freedom

Motivations

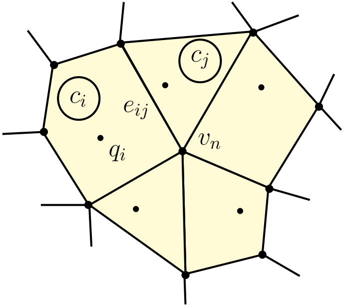
- Why a second-order scheme?
  - ☺ Quite simple and easy to code
  - ☺ Easy dissemination in a broaden scientific community
  - ☺ Many practical problems do not require complex and accurate methods
  - ☺ High-order methods can be very time and memory consuming
- Why a cell-vertex method?
  - ☺ More degrees of freedom
  - ☺ Simple reconstruction of the gradient

Motivations

- Why a second-order scheme?
  - ☺ Quite simple and easy to code
  - ☺ Easy dissemination in a broaden scientific community
  - ☺ Many practical problems do not require complex and accurate methods
  - ☺ High-order methods can be very time and memory consuming
- Why a cell-vertex method?
  - ☺ More degrees of freedom
  - ☺ Simple reconstruction of the gradient
  - ☺ Simple integration formulas on cells

Cell-Vertex Reconstruction

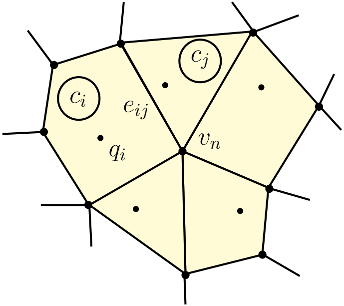
Notations



- c_i and $|c_i|$, $i = 1, \ldots, I$ – cell and its area

Cell-Vertex Reconstruction

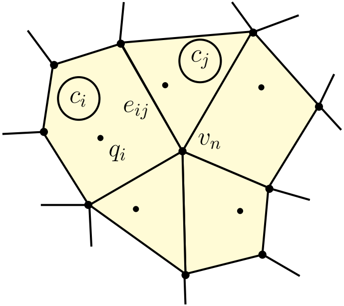
Notations



- c_i and $|c_i|$, $i = 1, \ldots, I$ – cell and its area
- e_{ij} and $|e_{ij}|$ – edge between c_i and c_i, and its length

Cell-Vertex Reconstruction

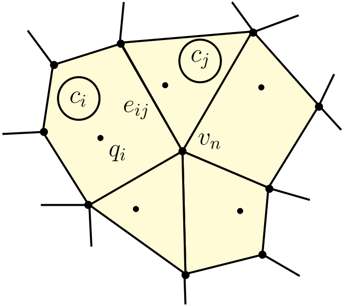
Notations



- c_i and $|c_i|$, $i = 1, \ldots, I$ – cell and its area
- e_{ij} and $|e_{ij}|$ – edge between c_i and c_i, and its length
- $q_i = (q_{ix}, q_{iy})$ – reference point of c_i

Cell-Vertex Reconstruction

Notations



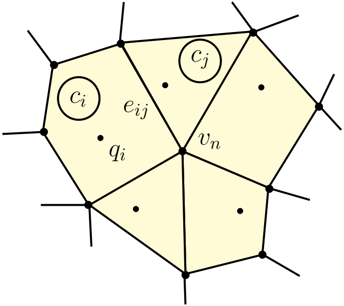
- c_i and $|c_i|$, $i = 1, \ldots, I$ – cell and its area
- e_{ij} and $|e_{ij}|$ – edge between c_i and c_i, and its length
- $q_i = (q_{ix}, q_{iy})$ – reference point of c_i

  ☞ Any point, not necessary the centroid or the mass centre!

- $v_n = (v_{nx}, v_{ny})$, $n = 1, \ldots, N$ – vertices

Cell-Vertex Reconstruction

Notations



- c_i and $|c_i|$, $i = 1, \dots, I$ – cell and its area
- e_{ij} and $|e_{ij}|$ – edge between c_i and c_i, and its length
- $q_i = (q_{ix}, q_{iy})$ – reference point of c_i

  ☞ Any point, not necessary the centroid or the mass centre!
- $v_n = (v_{nx}, v_{ny})$, $n = 1, \dots, N$ – vertices
- Stencil:

$$\mu(n) = \{\text{indicies of the cells which share } v_n\}$$

Cell-Vertex Reconstruction

Goal of the Method

- $\phi \equiv \phi(x, y)$ – regular function in Ω (domain)

Cell-Vertex Reconstruction

Goal of the Method

- $\phi \equiv \phi(x, y)$ – regular function in Ω (domain)
- Approximations on cells (given, at this stage):

$$\phi_i \approx \phi(q_i)$$

$$\Phi = (\phi_i)_{i=1,\dots,I}$$

Cell-Vertex Reconstruction

Goal of the Method

- $\phi \equiv \phi(x, y)$ – regular function in Ω (domain)
- Approximations on cells (given, at this stage):

$$\phi_i \approx \phi(q_i)$$

$$\Phi = (\phi_i)_{i=1,\dots,I}$$

- Approximations on vertices:

$$\psi_n \approx \phi(v_n)$$

$$\Psi = (\psi_n)_{n=1,\dots,N}$$

Cell-Vertex Reconstruction

Goal of the Method

- $\phi \equiv \phi(x, y)$ – regular function in Ω (domain)
- Approximations on cells (given, at this stage):

$$\phi_i \approx \phi(q_i)$$

$$\Phi = (\phi_i)_{i=1,\dots,I}$$

- Approximations on vertices:

$$\psi_n \approx \phi(v_n)$$

$$\Psi = (\psi_n)_{n=1,\dots,N}$$

☞ Goal:

$$\text{Cell } (\Phi) \rightarrow \text{Vertex } (\Psi)$$

Cell-Vertex Reconstruction

The Method

- Frink's method (1991): linear combination (☹ First-order approximation!)

$$\psi_n = \sum_{i \in \mu(n)} \beta_{ni} \phi_i \quad \text{with} \quad \beta_{ni} = \frac{\overline{q_i v_n}}{\sum_{i \in \mu(n)} \overline{q_i v_n}}$$

Cell-Vertex Reconstruction

The Method

- Frink's method (1991): linear combination (☹ First-order approximation!)

$$\boxed{\psi_n = \sum_{i\in\mu(n)} \beta_{ni}\phi_i} \quad \text{with} \quad \beta_{ni} = \frac{\overline{q_i v_n}}{\sum\limits_{i\in\mu(n)} \overline{q_i v_n}}$$

- Rauch et al.'s method (1991): extended Frink's method

Coefficients = Minimization Functional + Affine Constraints

Cell-Vertex Reconstruction

The Method

- Frink's method (1991): linear combination (☹ First-order approximation!)

$$\boxed{\psi_n = \sum_{i\in\mu(n)} \beta_{ni}\phi_i} \quad \text{with} \quad \beta_{ni} = \frac{\overline{q_i v_n}}{\sum_{i\in\mu(n)} \overline{q_i v_n}}$$

- Rauch et al.'s method (1991): extended Frink's method

Coefficients = Minimization Functional + Affine Constraints

- Chandrashekar et al.'s method (2013): extended Rauch et al.'s method

Coefficients = Minimization Functional + Affine Constraints + Weights

Cell-Vertex Reconstruction

The Method

- Coudière et al.'s method (1999): affine reconstrution

$$\psi_n = a v_{nx} + b v_{nx} + c$$

Coefficients = Minimization Functional + Affine Constraints

Cell-Vertex Reconstruction

The Method

- Coudière et al.'s method (1999): affine reconstrution

$$\psi_n = a v_{nx} + b v_{nx} + c$$

Coefficients = Minimization Functional + Affine Constraints

☹ No control of the coefficient of each cell

Cell-Vertex Reconstruction

The Method

- Coudière et al.'s method (1999): affine reconstrution

$$\psi_n = a v_{nx} + b v_{nx} + c$$

Coefficients = Minimization Functional + Affine Constraints

☹ No control of the coefficient of each cell

☹ Does not guarantee the preservation of the positivity principle

Cell-Vertex Reconstruction

The Method

- Coudière et al.'s method (1999): affine reconstrution

$$\psi_n = a v_{nx} + b v_{nx} + c$$

Coefficients = Minimization Functional + Affine Constraints

☹ No control of the coefficient of each cell

☹ Does not guarantee the preservation of the positivity principle

- Bertolazzi et al.'s (2004): extended Coudière et al.'s method

Coefficients = Minimization Functional + Affine Constraints + Weights

Cell-Vertex Reconstruction

Costa, Clain and Machado's method

- Costa, Clain and Machado's method (2014):
  - Linear combination:

$$\psi_n = \sum_{i \in \mu(n)} \beta_{ni} \phi_i$$

$$B^n = (\beta_{ni})_{i \in \mu(n)}$$

Cell-Vertex Reconstruction

Costa, Clain and Machado's method

- Costa, Clain and Machado's method (2014):

  - Linear combination:

$$\psi_n = \sum_{i \in \mu(n)} \beta_{ni} \phi_i$$

$$B^n = (\beta_{ni})_{i \in \mu(n)}$$

  - Affine constraints:

$$\Lambda_1(B^n) = \sum_{i \in \mu(n)} \beta_{ni}, \quad \Lambda_2(B^n) = \sum_{i \in \mu(n)} \beta_{ni} x_{ni}, \quad \Lambda_3(B^n) = \sum_{i \in \mu(n)} \beta_{ni} y_{ni}$$

$$\Lambda_1(B^n) = 1 \qquad \Lambda_2(B^n) = 0 \qquad \Lambda_3(B^n) = 0$$

where $x_{ni} = q_{ix} - v_{nx}$, $y_{ni} = q_{iy} - v_{ny}$

Cell-Vertex Reconstruction

Costa, Clain and Machado's method

- Quadratic functional:

$$E(B^n) = \frac{1}{2} \sum_{i \in \mu(n)} \omega_{ni} (\beta_{ni} - \theta_{ni})^2 \qquad \sum_{i \in \mu(n)} \theta_{ni} = 1$$

ω_{ni} – positive weights, θ_{ni} – targets

Cell-Vertex Reconstruction

Costa, Clain and Machado's method

- Quadratic functional:

$$E(B^n) = \frac{1}{2} \sum_{i \in \mu(n)} \omega_{ni} (\beta_{ni} - \theta_{ni})^2 \qquad \sum_{i \in \mu(n)} \theta_{ni} = 1$$

ω_{ni} – positive weights, θ_{ni} – targets

- Lagrange multipliers: find vector $\Lambda^n = (\lambda_{n1}, \lambda_{n2}, \lambda_{n3})$ such that

$$\beta_{ni} = \theta_{ni} - \frac{1}{\omega_{ni}} (\lambda_{n1} + \lambda_{n2} x_{ni} + \lambda_{n3} y_{ni}), \ i \in \mu(n)$$

- System of linear equations $\rightarrow \Lambda^n \rightarrow B^n$

Cell-Vertex Reconstruction

Positivity principle preserving

- Positivity principle: positive cell values ϕ_i, $i \in \mu(n)$, yield a positive vertex value ψ_n

Cell-Vertex Reconstruction

Positivity principle preserving

- Positivity principle: positive cell values ϕ_i, $i \in \mu(n)$, yield a positive vertex value ψ_n
- To guarantee the positivity principle we seek positive coefficients, because

$$\text{if } \beta_{ni} \geq 0,\ \phi_i \geq 0 \ \therefore\ \psi_n = \sum_{i \in \mu(n)} \beta_{ni}\phi_i \geq 0$$

Cell-Vertex Reconstruction

Positivity principle preserving

- Positivity principle: positive cell values ϕ_i, $i \in \mu(n)$, yield a positive vertex value ψ_n

- To guarantee the positivity principle we seek positive coefficients, because

$$\text{if } \beta_{ni} \geq 0,\ \phi_i \geq 0 \ \therefore\ \psi_n = \sum_{i \in \mu(n)} \beta_{ni} \phi_i \geq 0$$

1. Positive target values

Cell-Vertex Reconstruction

Positivity principle preserving

☞ Positivity principle: positive cell values ϕ_i, $i \in \mu(n)$, yield a positive vertex value ψ_n

- To guarantee the positivity principle we seek positive coefficients, because

$$\text{if } \beta_{ni} \geq 0,\ \phi_i \geq 0 \ \therefore\ \psi_n = \sum_{i \in \mu(n)} \beta_{ni}\phi_i \geq 0$$

1. Positive target values: about 2% of negative coefficients

Cell-Vertex Reconstruction

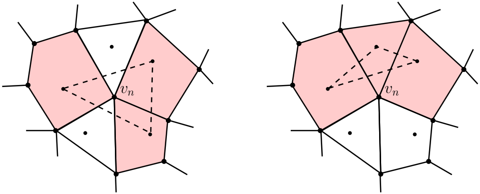
Positivity principle preserving

☞ Positivity principle: positive cell values ϕ_i, $i \in \mu(n)$, yield a positive vertex value ψ_n

- To guarantee the positivity principle we seek positive coefficients, because

$$\text{if } \beta_{ni} \geq 0, \ \phi_i \geq 0 \ \therefore \ \psi_n = \sum_{i \in \mu(n)} \beta_{ni} \phi_i \geq 0$$

1. Positive target values: about 2% of negative coefficients
2. Choose a new stencil consisting of three cells of $\mu(n)$



Cell-Vertex Reconstruction

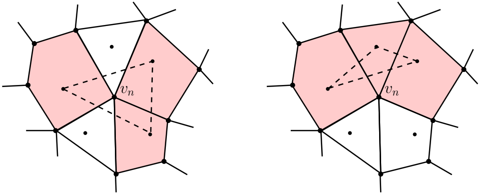
Positivity principle preserving

☞ Positivity principle: positive cell values ϕ_i, $i \in \mu(n)$, yield a positive vertex value ψ_n

- To guarantee the positivity principle we seek positive coefficients, because

$$\text{if } \beta_{ni} \geq 0,\ \phi_i \geq 0 \ \therefore\ \psi_n = \sum_{i \in \mu(n)} \beta_{ni}\phi_i \geq 0$$

1. Positive target values: about 2% of negative coefficients
2. Choose a new stencil consisting of three cells of $\mu(n)$



3. Best configuration? $B^n = (1/3, 1/3, 1/3)$

Cell-Vertex Reconstruction

3D extension

- Linear combination:

$$\psi_n = \sum_{i \in \mu(n)} \beta_{ni} \phi_i$$

- Affine constraints:

$$\Lambda_1(B^n) = \sum_{i \in \mu(n)} \beta_{ni}, \quad \Lambda_2(B^n) = \sum_{i \in \mu(n)} \beta_{ni} x_{ni}, \quad \Lambda_3(B^n) = \sum_{i \in \mu(n)} \beta_{ni} y_{ni},$$

$$\Lambda_4(B^n) = \sum_{i \in \mu(n)} \beta_{ni} z_{ni}$$

$$\Lambda_1(B^n) = 1 \quad \Lambda_2(B^n) = 0 \quad \Lambda_3(B^n) = 0 \quad \Lambda_4(B^n) = 0$$

- Quadratic functional + Lagrange Multipliers

Cell-Vertex Reconstruction

Costa, Clain and Machado's method

- The coefficients β_{ni} depend only on geometric factors

Cell-Vertex Reconstruction

Costa, Clain and Machado's method

- The coefficients β_{ni} depend only on geometric factors
- Pre-processing step

Cell-Vertex Reconstruction

Costa, Clain and Machado's method

- The coefficients β_{ni} depend only on geometric factors
- Pre-processing step
- Treatment of the boundary vertices:
  - Dirichlet vertex: $\psi_n = \phi_{\mathsf{D}}(v_n)$

Cell-Vertex Reconstruction

Costa, Clain and Machado's method

- The coefficients β_{ni} depend only on geometric factors
- Pre-processing step
- Treatment of the boundary vertices:
  - Dirichlet vertex: $\psi_n = \phi_{\text{D}}(v_n)$
  - Neumann vertex (2 different techniques):
    - interpolation technique as explained before
    - interpolation technique with ghost cells + Neumann conditions

Polynomial Reconstructions

Polynomial reconstructions on cells

☞ Let us assume vectors Φ and Ψ are given!

Polynomial Reconstructions

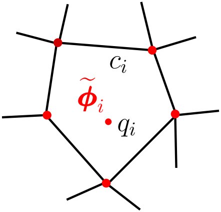
Polynomial reconstructions on cells

☞ Let us assume vectors Φ and Ψ are given!

- First-degree polynomial associated to the cell c_i:

$$\boldsymbol{\phi}_i(x,y) = \phi_i + \mathcal{R}_{i,x}\,(x - q_{ix}) + \mathcal{R}_{i,y}\,(y - q_{iy})$$

$$S_i = \{\text{indices of the vertices of } c_i\}$$



Polynomial Reconstructions

Polynomial reconstructions on cells

☞ Let us assume vectors Φ and Ψ are given!

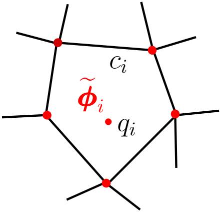
- First-degree polynomial associated to the cell c_i:

$$\boldsymbol{\phi}_i(x,y) = \phi_i + \mathcal{R}_{i,x}\left(x - q_{ix}\right) + \mathcal{R}_{i,y}\left(y - q_{iy}\right)$$

$$S_i = \{\text{indices of the vertices of } c_i\}$$

☞ Find $\widetilde{\mathcal{R}}_{i,x}$ and $\widetilde{\mathcal{R}}_{i,y}$ which minimize

$$\widetilde{E}_i(\mathcal{R}_{ix}, \mathcal{R}_{iy}) = \sum_{n \in S_i} \left(\boldsymbol{\phi}_i(v_n) - \psi_n\right)^2$$



Polynomial Reconstructions

Polynomial reconstructions on cells

☞ Let us assume vectors Φ and Ψ are given!

- First-degree polynomial associated to the cell c_i:

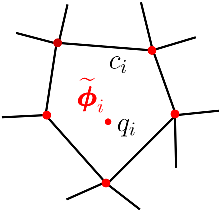
$$\boldsymbol{\phi}_i(x, y) = \phi_i + \mathcal{R}_{i,x}\,(x - q_{ix}) + \mathcal{R}_{i,y}\,(y - q_{iy})$$

$$S_i = \{\text{indices of the vertices of } c_i\}$$

☞ Find $\widetilde{\mathcal{R}}_{i,x}$ and $\widetilde{\mathcal{R}}_{i,y}$ which minimize

$$\widetilde{E}_i(\mathcal{R}_{ix}, \mathcal{R}_{iy}) = \sum_{n \in S_i} \left(\boldsymbol{\phi}_i(v_n) - \psi_n\right)^2$$

- $\widetilde{\boldsymbol{\phi}}_i(x, y)$ – associated polynomial



Polynomial Reconstructions

Polynomial reconstructions on edges

- First-degree polynomial associated to the edge e_{ij}:

$$\boldsymbol{\phi}_{ij}(x,y) = \phi_i + \mathcal{R}_{ijx}(x - q_{ix}) + \mathcal{R}_{ijy}(y - q_{iy})$$

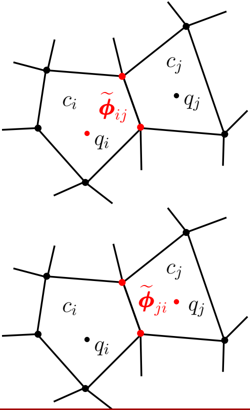
Polynomial Reconstructions

Polynomial reconstructions on edges

- First-degree polynomial associated to the edge e_{ij}:

$$\boldsymbol{\phi}_{ij}(x,y) = \phi_i + \mathcal{R}_{ijx}(x - q_{ix}) + \mathcal{R}_{ijy}(y - q_{iy})$$

$$S_{ij} = \{\text{indices of the vertices of } e_{ij}\}$$



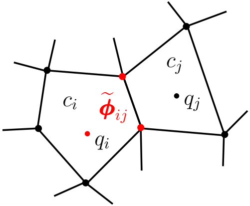
Polynomial Reconstructions

Polynomial reconstructions on edges

- First-degree polynomial associated to the edge e_{ij}:

$$\boldsymbol{\phi}_{ij}(x,y) = \phi_i + \mathcal{R}_{ijx}(x - q_{ix}) + \mathcal{R}_{ijy}(y - q_{iy})$$

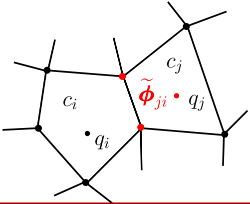
$$S_{ij} = \{\text{indices of the vertices of } e_{ij}\}$$



☞ Find $\widetilde{\mathcal{R}}_{ijx}$ and $\widetilde{\mathcal{R}}_{ijy}$, such that $\widetilde{\boldsymbol{\phi}}_{ij}(x,y)$ interpolates ϕ_i and $\psi_n, n \in S_{ij}$

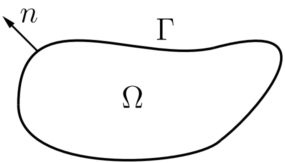
☞ $\widetilde{\boldsymbol{\phi}}_{ji}(x,y)$ is also performed

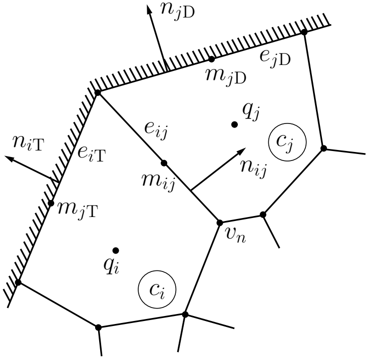
☞ The same procedure for the boundary edges



Steady-state Convection-Diffusion-Reaction Problems

Mesh





- Domain Ω, boundary $\Gamma = \Gamma_{\mathsf{D}} \cup \Gamma_{\mathsf{P}} \cup \Gamma_{\mathsf{T}}$
- $\nu(i) = \{$indices of the cells/boundaries which share an edge with $c_i\}$
- $|e_{ij}|$, $i = 1 \ldots, I$, $j \in \nu(i)$ – length of e_{ij}

Steady-state Convection-Diffusion-Reaction Problems

Formulation

$$\nabla \cdot (V\phi - \kappa \nabla \phi) + r\phi = f, \text{ in } \Omega$$

Steady-state Convection-Diffusion-Reaction Problems

Formulation

$$\nabla \cdot (V\phi - \kappa \nabla \phi) + r\phi = f, \text{ in } \Omega$$

- $\phi \equiv \phi(x, y)$ – unknown
- $V = (u, v) \equiv (u(x, y), v(x, y))$ – velocity
- $\kappa \equiv \kappa(x, y)$ – diffusion coefficient
- $r \equiv r(x, y)$ – reaction coefficient
- $f \equiv f(x, y)$ – source term

Steady-state Convection-Diffusion-Reaction Problems

Formulation

$$\nabla \cdot (V\phi - \kappa \nabla \phi) + r\phi = f, \text{ in } \Omega$$

- $\phi \equiv \phi(x, y)$ – unknown
- $V = (u, v) \equiv (u(x, y), v(x, y))$ – velocity
- $\kappa \equiv \kappa(x, y)$ – diffusion coefficient
- $r \equiv r(x, y)$ – reaction coefficient
- $f \equiv f(x, y)$ – source term
- Dirichlet: $\phi = \phi_D(x, y)$, on Γ_D

Steady-state Convection-Diffusion-Reaction Problems

Formulation

$$\nabla \cdot (V\phi - \kappa\nabla\phi) + r\phi = f, \text{ in } \Omega$$

- $\phi \equiv \phi(x, y)$ – unknown
- $V = (u, v) \equiv (u(x, y), v(x, y))$ – velocity
- $\kappa \equiv \kappa(x, y)$ – diffusion coefficient
- $r \equiv r(x, y)$ – reaction coefficient
- $f \equiv f(x, y)$ – source term
- Dirichlet: $\phi = \phi_{\mathsf{D}}(x, y)$, on Γ_{D}
- Partial Neumann: $-\kappa\nabla\phi \cdot n = g_{\mathsf{P}}(x, y)$, on Γ_{P}

Steady-state Convection-Diffusion-Reaction Problems

Formulation

$$\nabla \cdot (V\phi - \kappa \nabla \phi) + r\phi = f, \text{ in } \Omega$$

- $\phi \equiv \phi(x, y)$ – unknown
- $V = (u, v) \equiv (u(x, y), v(x, y))$ – velocity
- $\kappa \equiv \kappa(x, y)$ – diffusion coefficient
- $r \equiv r(x, y)$ – reaction coefficient
- $f \equiv f(x, y)$ – source term
- Dirichlet: $\phi = \phi_\mathsf{D}(x, y)$, on Γ_D
- Partial Neumann: $-\kappa \nabla \phi \cdot n = g_\mathsf{P}(x, y)$, on Γ_P
- Total Neumann: $V \cdot n\phi - \kappa \nabla \phi \cdot n = g_\mathsf{T}(x, y)$, on Γ_T

Steady-state Convection-Diffusion-Reaction Problems

Finite volume discretization

- Integration over cell c_i and applying the divergence theorem

$$\sum_{j\in\nu(i)} \frac{|e_{ij}|}{|c_i|}\frac{1}{|e_{ij}|}\int_{e_{ij}} (V\cdot n_{ij}\phi - \kappa\nabla\phi\cdot n_{ij})\,\mathrm{d}s + \frac{1}{|c_i|}\int_{c_i} r\phi\,\mathrm{d}X - \frac{1}{|c_i|}\int_{c_i} f\,\mathrm{d}X = 0$$

Steady-state Convection-Diffusion-Reaction Problems

Finite volume discretization

- Integration over cell c_i and applying the divergence theorem

$$\sum_{j\in\nu(i)} \frac{|e_{ij}|}{|c_i|}\frac{1}{|e_{ij}|}\int_{e_{ij}} (V\cdot n_{ij}\phi - \kappa\nabla\phi\cdot n_{ij})\,\mathrm{d}s + \frac{1}{|c_i|}\int_{c_i} r\phi\,\mathrm{d}X - \frac{1}{|c_i|}\int_{c_i} f\,\mathrm{d}X = 0$$

- Generic finite volume scheme:

$$\sum_{j\in\nu(i)} \frac{|e_{ij}|}{|c_i|}\mathcal{F}_{ij}(\Phi) + \mathcal{R}_i(\Phi) - f_i = \mathcal{O}(h^2),\ i = 1\dots,I$$

- $\mathcal{F}_{ij}$ – convective and diffusive fluxes
- $\mathcal{R}_i$ – mean reactive part
- f_i – mean source term

Steady-state Convection-Diffusion-Reaction Problems

Second-order finite volume scheme – numerical fluxes

- Inner edge e_{ij}:

$$\mathcal{F}_{ij} = [V(m_{ij}) \cdot n_{ij}]^+ \, \widetilde{\boldsymbol{\phi}}_i(m_{ij}) + [V(m_{ij}) \cdot n_{ij}]^- \, \widetilde{\boldsymbol{\phi}}_j(m_{ij}) - \kappa(m_{ij}) \nabla \check{\boldsymbol{\phi}}_{ij}(m_{ij}) \cdot n_{ij}$$

where

$$\check{\boldsymbol{\phi}}_{ij} = \check{\boldsymbol{\phi}}_{ji} = \sigma_{ij} \widetilde{\boldsymbol{\phi}}_{ij} + \sigma_{ji} \widetilde{\boldsymbol{\phi}}_{ji}, \quad \sigma_{ij} = \frac{|c_i|}{|c_i| + |c_j|}, \quad \sigma_{ji} = \frac{|c_j|}{|c_i| + |c_j|}$$

Steady-state Convection-Diffusion-Reaction Problems

Second-order finite volume scheme – numerical fluxes

- Inner edge e_{ij}:

$$\mathcal{F}_{ij} = [V(m_{ij}) \cdot n_{ij}]^+ \,\widetilde{\boldsymbol{\phi}}_i(m_{ij}) + [V(m_{ij}) \cdot n_{ij}]^- \,\widetilde{\boldsymbol{\phi}}_j(m_{ij}) - \kappa(m_{ij}) \nabla \check{\boldsymbol{\phi}}_{ij}(m_{ij}) \cdot n_{ij}$$

where

$$\check{\boldsymbol{\phi}}_{ij} = \check{\boldsymbol{\phi}}_{ji} = \sigma_{ij} \widetilde{\boldsymbol{\phi}}_{ij} + \sigma_{ji} \widetilde{\boldsymbol{\phi}}_{ji}, \quad \sigma_{ij} = \frac{|c_i|}{|c_i| + |c_j|}, \quad \sigma_{ji} = \frac{|c_j|}{|c_i| + |c_j|}$$

- Dirichlet boundary edge $e_{i\mathsf{D}}$:

$$\mathcal{F}_{i\mathsf{D}} = [V(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}]^+ \,\widetilde{\boldsymbol{\phi}}_i(m_{i\mathsf{D}}) + [V(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}]^- \,\phi_{\mathsf{D}}(m_{i\mathsf{D}}) - \kappa(m_{i\mathsf{D}}) \nabla \widetilde{\boldsymbol{\phi}}_{i\mathsf{D}}(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}$$

Steady-state Convection-Diffusion-Reaction Problems

Second-order finite volume scheme – numerical fluxes

- Inner edge e_{ij}:

$$\mathcal{F}_{ij} = [V(m_{ij}) \cdot n_{ij}]^{+} \widetilde{\boldsymbol{\phi}}_i(m_{ij}) + [V(m_{ij}) \cdot n_{ij}]^{-} \widetilde{\boldsymbol{\phi}}_j(m_{ij}) - \kappa(m_{ij}) \nabla \check{\boldsymbol{\phi}}_{ij}(m_{ij}) \cdot n_{ij}$$

where

$$\check{\boldsymbol{\phi}}_{ij} = \check{\boldsymbol{\phi}}_{ji} = \sigma_{ij} \widetilde{\boldsymbol{\phi}}_{ij} + \sigma_{ji} \widetilde{\boldsymbol{\phi}}_{ji}, \quad \sigma_{ij} = \frac{|c_i|}{|c_i| + |c_j|}, \quad \sigma_{ji} = \frac{|c_j|}{|c_i| + |c_j|}$$

- Dirichlet boundary edge $e_{i\mathsf{D}}$:

$$\mathcal{F}_{i\mathsf{D}} = [V(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}]^{+} \widetilde{\boldsymbol{\phi}}_i(m_{i\mathsf{D}}) + [V(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}]^{-} \phi_{\mathsf{D}}(m_{i\mathsf{D}}) - \kappa(m_{i\mathsf{D}}) \nabla \widetilde{\boldsymbol{\phi}}_{i\mathsf{D}}(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}$$

- Partial Neumann boundary edge $e_{i\mathsf{P}}$: $\mathcal{F}_{i\mathsf{P}} = V(m_{i\mathsf{P}}) \cdot n_{i\mathsf{P}} \widetilde{\boldsymbol{\phi}}_i(m_{i\mathsf{P}}) + g_{\mathsf{P}}(m_{i\mathsf{P}})$

Steady-state Convection-Diffusion-Reaction Problems

Second-order finite volume scheme – numerical fluxes

- Inner edge e_{ij}:

$$\mathcal{F}_{ij} = [V(m_{ij}) \cdot n_{ij}]^{+}\, \widetilde{\boldsymbol{\phi}}_i(m_{ij}) + [V(m_{ij}) \cdot n_{ij}]^{-}\, \widetilde{\boldsymbol{\phi}}_j(m_{ij}) - \kappa(m_{ij}) \nabla \check{\boldsymbol{\phi}}_{ij}(m_{ij}) \cdot n_{ij}$$

where

$$\check{\boldsymbol{\phi}}_{ij} = \check{\boldsymbol{\phi}}_{ji} = \sigma_{ij} \widetilde{\boldsymbol{\phi}}_{ij} + \sigma_{ji} \widetilde{\boldsymbol{\phi}}_{ji}, \quad \sigma_{ij} = \frac{|c_i|}{|c_i| + |c_j|}, \quad \sigma_{ji} = \frac{|c_j|}{|c_i| + |c_j|}$$

- Dirichlet boundary edge $e_{i\mathsf{D}}$:

$$\mathcal{F}_{i\mathsf{D}} = [V(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}]^{+}\, \widetilde{\boldsymbol{\phi}}_i(m_{i\mathsf{D}}) + [V(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}]^{-}\, \phi_{\mathsf{D}}(m_{i\mathsf{D}}) - \kappa(m_{i\mathsf{D}}) \nabla \widetilde{\boldsymbol{\phi}}_{i\mathsf{D}}(m_{i\mathsf{D}}) \cdot n_{i\mathsf{D}}$$

- Partial Neumann boundary edge $e_{i\mathsf{P}}$: $\mathcal{F}_{i\mathsf{P}} = V(m_{i\mathsf{P}}) \cdot n_{i\mathsf{P}} \widetilde{\boldsymbol{\phi}}_i(m_{i\mathsf{P}}) + g_{\mathsf{P}}(m_{i\mathsf{P}})$
- Total Neumann boundary edge $e_{i\mathsf{T}}$: $\mathcal{F}_{i\mathsf{T}} = g_{\mathsf{T}}(m_{i\mathsf{T}})$

Steady-state Convection-Diffusion-Reaction Problems

Finite volume scheme – reactive part and source term

- $\mathcal{R}_i$ for cell c_i

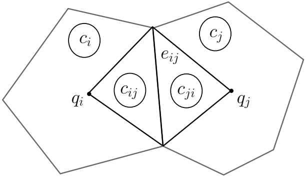
$$\mathcal{R}_i = r(q_i)\phi_i$$

or

$$\mathcal{R}_i = \frac{1}{|c_i|}\left[\sum_{j\in\nu(i)} \frac{|c_{ij}|}{3}\left(\sum_{n\in S_{ij}} r(v_n)\psi_n + r(q_i)\phi_i\right)\right]$$

- f_i for cell c_i

$$f_i = \frac{1}{|c_i|}\left[\sum_{j\in\nu(i)} \frac{|c_{ij}|}{3}\left(\sum_{n\in S_{ij}} f(v_n) + f(q_i)\right)\right]$$



Steady-state Convection-Diffusion-Reaction Problems

Residual scheme

- ☞ $\mathcal{F}_{ij}$ and $\mathcal{R}_i$ linearly depend on vector Φ

Steady-state Convection-Diffusion-Reaction Problems

Residual scheme

☞ $\mathcal{F}_{ij}$ and $\mathcal{R}_i$ linearly depend on vector Φ

- Affine operator $\Phi \to \mathcal{G}_i(\Phi)$ for each cell c_i, $i = 1, \dots, I$

$$\mathcal{G}_i(\Phi) = \sum_{j \in \nu(i)} \frac{|e_{ij}|}{|c_i|} \mathcal{F}_{ij}(\Phi) + \mathcal{R}_i(\Phi) - f_i$$

Steady-state Convection-Diffusion-Reaction Problems

Residual scheme

☞ $\mathcal{F}_{ij}$ and $\mathcal{R}_i$ linearly depend on vector Φ

- Affine operator $\Phi \to \mathcal{G}_i(\Phi)$ for each cell c_i, $i = 1, \ldots, I$

$$\mathcal{G}_i(\Phi) = \sum_{j \in \nu(i)} \frac{|e_{ij}|}{|c_i|} \mathcal{F}_{ij}(\Phi) + \mathcal{R}_i(\Phi) - f_i$$

- $\mathcal{G}(\Phi) = (\mathcal{G}_i(\Phi))_{i=1\ldots,I}$ is an affine operator from $\mathbb{R}^I$ into $\mathbb{R}^I$

Steady-state Convection-Diffusion-Reaction Problems

Residual scheme

☞ $\mathcal{F}_{ij}$ and $\mathcal{R}_i$ linearly depend on vector Φ

- Affine operator $\Phi \to \mathcal{G}_i(\Phi)$ for each cell c_i, $i = 1, \ldots, I$

$$\mathcal{G}_i(\Phi) = \sum_{j \in \nu(i)} \frac{|e_{ij}|}{|c_i|} \mathcal{F}_{ij}(\Phi) + \mathcal{R}_i(\Phi) - f_i$$

- $\mathcal{G}(\Phi) = (\mathcal{G}_i(\Phi))_{i=1\ldots,I}$ is an affine operator from $\mathbb{R}^I$ into $\mathbb{R}^I$
- $\mathcal{G}(\Phi) = 0_I$ provides the solution $\Phi^\star$

Steady-state Convection-Diffusion-Reaction Problems

Residual scheme

☞ $\mathcal{F}_{ij}$ and $\mathcal{R}_i$ linearly depend on vector Φ

- Affine operator $\Phi \to \mathcal{G}_i(\Phi)$ for each cell c_i, $i = 1, \dots, I$

$$\mathcal{G}_i(\Phi) = \sum_{j \in \nu(i)} \frac{|e_{ij}|}{|c_i|} \mathcal{F}_{ij}(\Phi) + \mathcal{R}_i(\Phi) - f_i$$

- $\mathcal{G}(\Phi) = (\mathcal{G}_i(\Phi))_{i=1\dots,I}$ is an affine operator from $\mathbb{R}^I$ into $\mathbb{R}^I$
- $\mathcal{G}(\Phi) = 0_I$ provides the solution $\Phi^\star$
- GMRES procedure to solve the affine problem (free-matrix method)

Steady-state Convection-Diffusion-Reaction Problems

Residual scheme

☞ $\mathcal{F}_{ij}$ and $\mathcal{R}_i$ linearly depend on vector Φ

- Affine operator $\Phi \to \mathcal{G}_i(\Phi)$ for each cell c_i, $i = 1, \ldots, I$

$$\mathcal{G}_i(\Phi) = \sum_{j \in \nu(i)} \frac{|e_{ij}|}{|c_i|} \mathcal{F}_{ij}(\Phi) + \mathcal{R}_i(\Phi) - f_i$$

- $\mathcal{G}(\Phi) = (\mathcal{G}_i(\Phi))_{i=1\ldots,I}$ is an affine operator from $\mathbb{R}^I$ into $\mathbb{R}^I$
- $\mathcal{G}(\Phi) = 0_I$ provides the solution $\Phi^\star$
- GMRES procedure to solve the affine problem (free-matrix method)
- Preconditioning matrix based on a Patankar-like discretization

Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Criteria

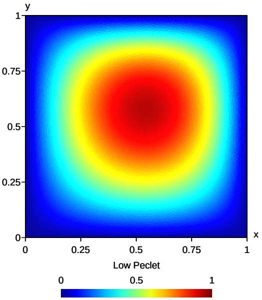
- Exact solution: $\phi_i = \phi(q_i)$, $i = 1, \ldots, I$
- Numerical solution $\Phi^\star = (\phi_i^\star)_{i=1,\ldots,I}$
- Relative discrete L^2-norm error

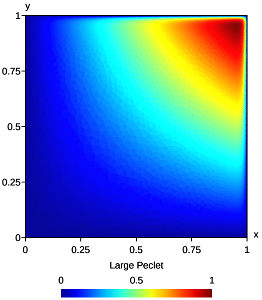
$$E_2 = \left(\frac{\displaystyle\sum_{i=1}^{I} |c_i| (\phi_i - \phi_i^\star)^2}{\displaystyle\sum_{i=1}^{I} |c_i| \phi_i^\star} \right)^{\frac{1}{2}}$$

Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Convection-diffusion problem

- $\Omega =]0,1[^2$
- $\Gamma_{\mathrm{D}} = \Gamma$
- $\phi_{\mathrm{D}}(x,y) = 0$, on Γ_{D}
- $V = (u,v)$, $\kappa = 1$
- $\phi(x,y) = C\alpha(x)\beta(y)$
- $f(x,y) = C(\alpha(x)+\beta(y))$





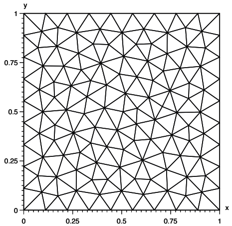
$$\alpha(x) = \frac{1}{u}\left(x - \frac{e^{ux}-1}{e^{u}-1}\right), \quad \beta(y) = \frac{1}{v}\left(y - \frac{e^{vy}-1}{e^{v}-1}\right)$$

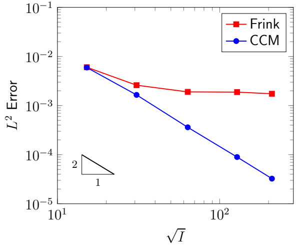
- Low Péclet number: $V = (1,2)$, $C = 65$
- Large Péclet number: $V = (100,100)$, $C = 11236$

Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Convection-diffusion problem

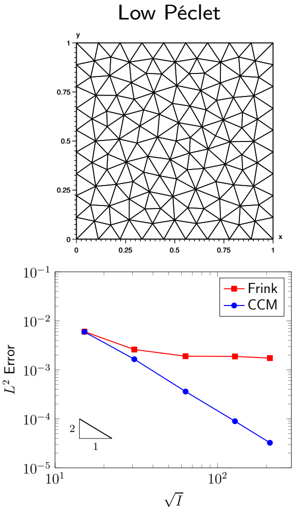
Low Péclet

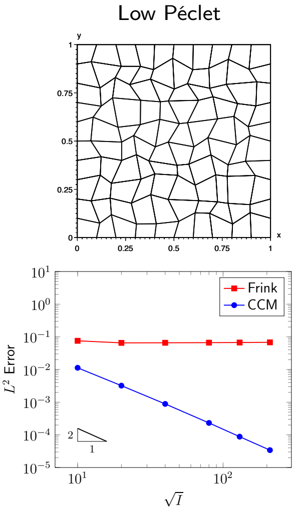




Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Convection-diffusion problem

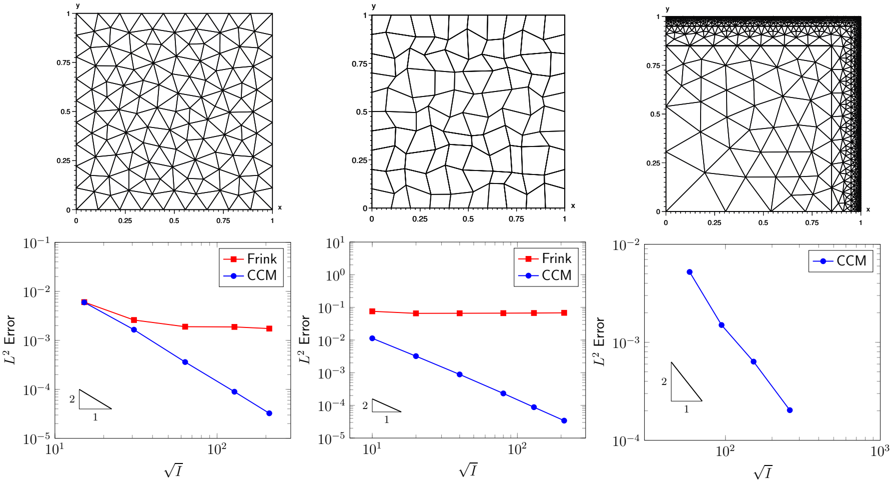




Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Convection-diffusion problem

Low Péclet | Low Péclet | Large Péclet



Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Convection-diffusion problem

- Random value $\xi_{ni} \in [0,1]$
- Random point on c_i:

$$p_i = \frac{\sum_{n \in S_i} \xi_{ni} v_n}{\sum_{n \in S_i} \xi_{ni}}$$

- Reference cell point of c_i:

$$q_i = m_i + \alpha(p_i - m_i)$$

- $\alpha \in [0,1[$ – deformation factor

Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Convection-diffusion problem

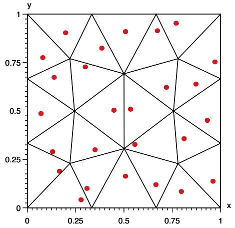
- Random value $\xi_{ni} \in [0, 1]$
- Random point on c_i:

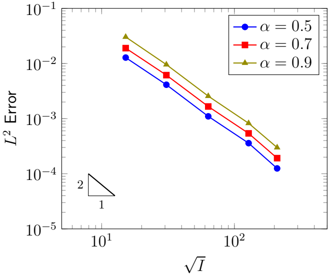
$$p_i = \frac{\sum_{n \in S_i} \xi_{ni} v_n}{\sum_{n \in S_i} \xi_{ni}}$$

- Reference cell point of c_i:

$$q_i = m_i + \alpha(p_i - m_i)$$

- $\alpha \in [0, 1[$ – deformation factor



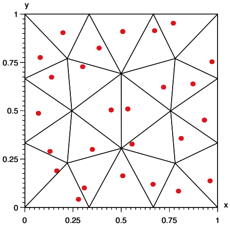


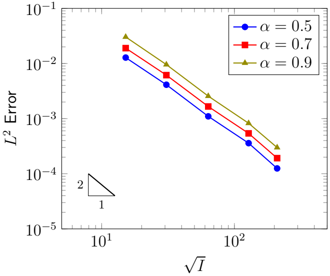
Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Convection-diffusion problem

- Random value $\xi_{ni} \in [0, 1]$
- Random point on c_i:

$$p_i = \frac{\sum_{n \in S_i} \xi_{ni} v_n}{\sum_{n \in S_i} \xi_{ni}}$$

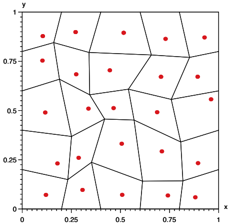


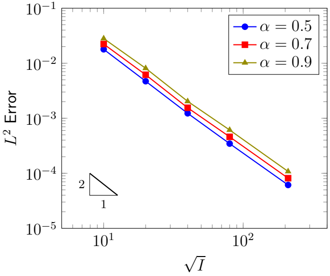


- Reference cell point of c_i:

$$q_i = m_i + \alpha(p_i - m_i)$$

- $\alpha \in [0, 1[$ – deformation factor

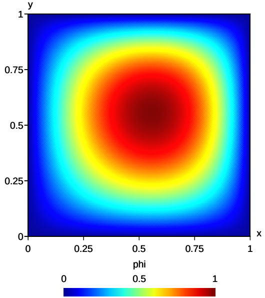




Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Diffusion-reaction problem

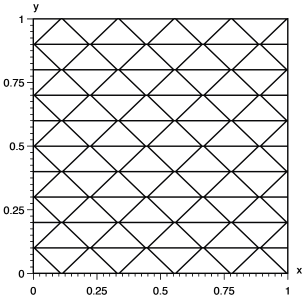
- $\Omega =]0,1[^2$
- $\Gamma_\mathsf{D} = \Gamma$
- $\phi_\mathsf{D}(x,y) = 0$, on Γ_D
- $\kappa = 1$, $r = 10^6$
- $\phi(x,y) = 3.14x\,(e^x - e)\,y\,(e^y - e)$
- $f = -\kappa\Delta\phi + r\phi$

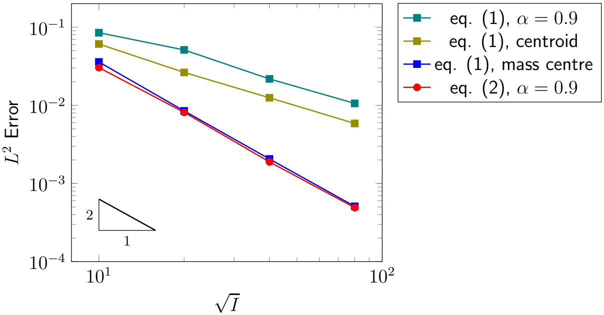


$$\mathcal{R}_i = r(q_i)\phi_i \quad (1) \quad vs \quad \mathcal{R}_i = \frac{1}{|c_i|}\left[\sum_{j\in\nu(i)} \frac{|c_{ij}|}{3}\left(\sum_{n\in S_{ij}} r(v_n)\psi_n + r(q_i)\phi_i\right)\right] \quad (2)$$

Steady-state Convection-Diffusion-Reaction Problems

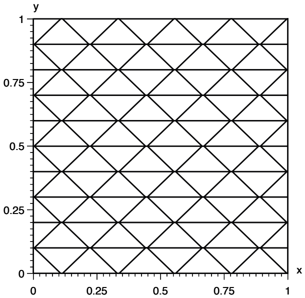
Numerical Tests – Diffusion-reaction problem

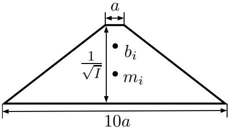


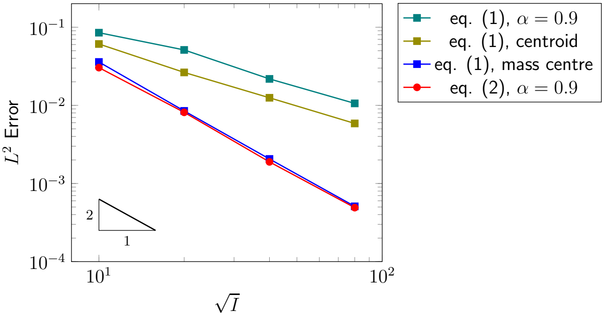


Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – Diffusion-reaction problem







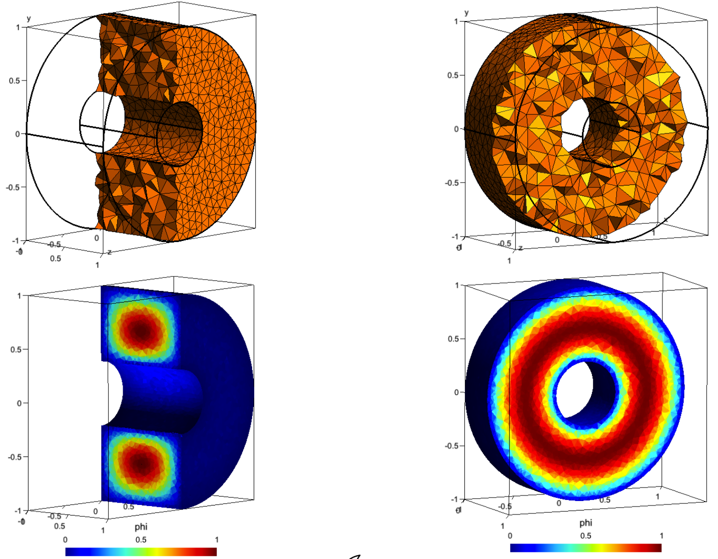
Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – 3D diffusion-reaction problem

- $\Omega = \{(x, y, x) : 0.3 \leq \sqrt{x^2 + y^2} \leq 1,\ 0 \leq z \leq 1\}$
- $\Gamma_{\mathsf{D}} = \Gamma$
- $\phi_{\mathsf{D}}(x, y, z) = 0$, on Γ_{D}
- $\kappa = 1,\ r = 10^6$
- $\phi(x, y, z) = -7.16(z - z^2)\left(\frac{\ln(x^2 + y^2)}{\ln 0.3} + \frac{200}{91}(x^2 + y^2 - 1)\right)$
- $f(x, y, z) = -\kappa\Delta\nabla\phi + r\phi$

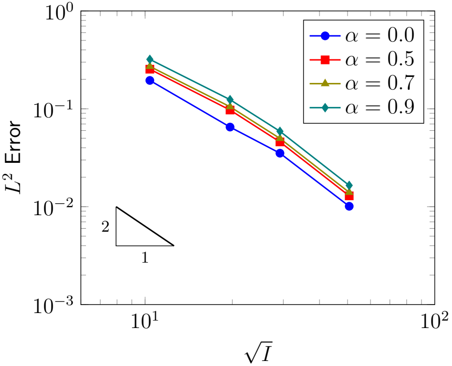
Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – 3D diffusion-reaction problem



Steady-state Convection-Diffusion-Reaction Problems

Numerical Tests – 3D diffusion-reaction problem



Anisotropic Diffusion Problems

Formulation

$$\nabla \cdot (-K\nabla\phi) = f, \text{ in } \Omega$$

- Domain Ω with boundary $\Gamma = \Gamma_\mathsf{D} \cup \Gamma_\mathsf{P}$
- $\phi \equiv \phi(x, y)$ – unknown
- $K \equiv K(x, y)$ – diffusion tensor, strictly positive definite 2×2 matrix

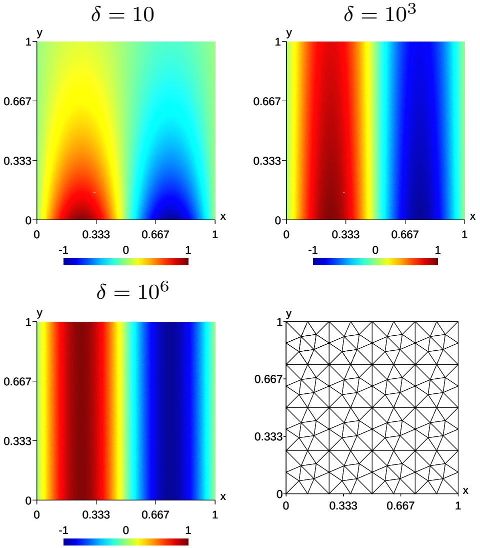
$$K = \begin{bmatrix} \kappa_{xx} & \kappa_{xy} \\ \kappa_{yx} & \kappa_{yy} \end{bmatrix}$$

- $f \equiv f(x, y)$ – source term
- Dirichlet: $\phi = \phi_\mathsf{D}(x, y)$, on Γ_D
- Partial Neumann: $-K\nabla\phi \cdot n = g_\mathsf{P}(x, y)$, on Γ_P

Anisotropic Diffusion Problems

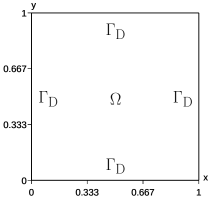
Numerical Tests – Numerical locking problem

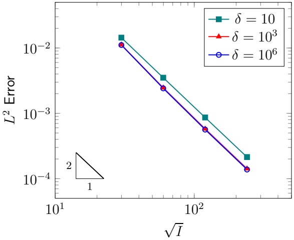
- $\Omega =]0,1[^2$
- $K = \begin{bmatrix} 1 & 0 \\ 0 & \delta \end{bmatrix}$
- $\delta = 10, 10^3, 10^6$
- $\phi(x,y) = \sin(2\pi x)\exp\left(\frac{-2\pi}{\sqrt{\delta}}y\right)$
- $f(x,y) = 0$



Anisotropic Diffusion Problems

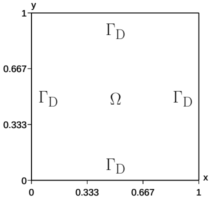
Numerical Tests – Numerical locking problem

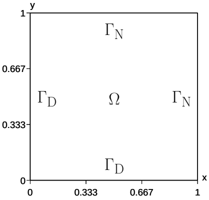


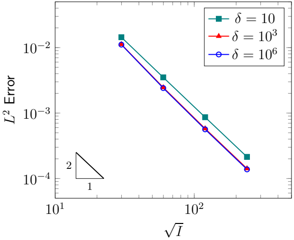


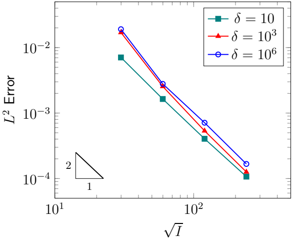
Anisotropic Diffusion Problems

Numerical Tests – Numerical locking problem



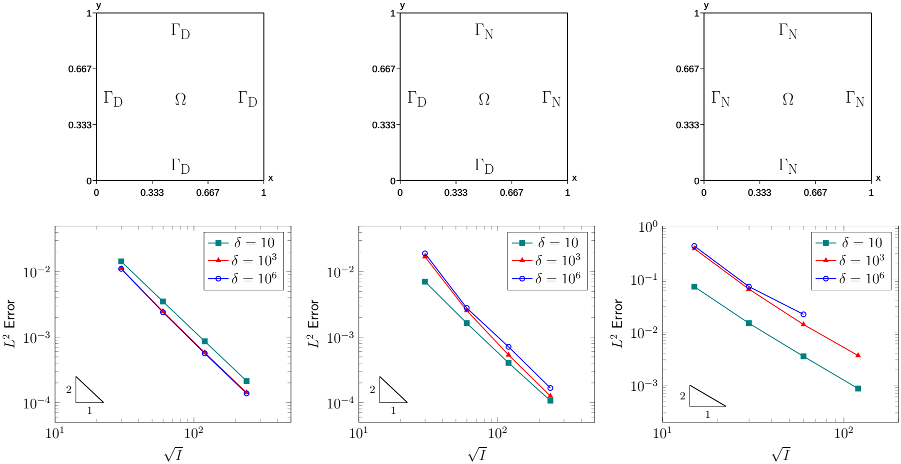






Anisotropic Diffusion Problems

Numerical Tests – Numerical locking problem



Conclusions and Perspectives

- ☑ Simple, robust, second-order accuracy

Conclusions and Perspectives

- ☑ Simple, robust, second-order accuracy
- ☑ Unstructured and deformed meshes still preserving the second-order accuracy

Conclusions and Perspectives

- ☑ Simple, robust, second-order accuracy
- ☑ Unstructured and deformed meshes still preserving the second-order accuracy
- ☑ Any reference cell points still preserving the second-order accuracy

Conclusions and Perspectives

- ☑ Simple, robust, second-order accuracy
- ☑ Unstructured and deformed meshes still preserving the second-order accuracy
- ☑ Any reference cell points still preserving the second-order accuracy
- ☑ Positivity principle preserving technique

Conclusions and Perspectives

- ☑ Simple, robust, second-order accuracy
- ☑ Unstructured and deformed meshes still preserving the second-order accuracy
- ☑ Any reference cell points still preserving the second-order accuracy
- ☑ Positivity principle preserving technique
- ☑ Low and Large Pléclet number problems, high-reactive and high-anisotropic diffusion problems, time-dependent problems

Conclusions and Perspectives

- ☑ Simple, robust, second-order accuracy
- ☑ Unstructured and deformed meshes still preserving the second-order accuracy
- ☑ Any reference cell points still preserving the second-order accuracy
- ☑ Positivity principle preserving technique
- ☑ Low and Large Pléclet number problems, high-reactive and high-anisotropic diffusion problems, time-dependent problems

⚠ Fluid dynamic problems

Thank you for your attention!

The authors grateful acknowledge funding by FEDER through Programa Operacional de Fatores de Competitividade – COMPETE and by Portuguese Funds FCT – Fundação para a Ciência e a Tecnologia, within the Projects PEst-C/MAT/UI0013/2011 and PTDC/MAT/121185/2010