High-order Solution Transfer between Curved Meshes and Ill-conditioned Bézier Curve Intersection

Danny Hermes

August 9, 2018

dhermes@berkeley.edu
UC Berkeley

Outline

1. Introduction and motivation
2. Curved Elements
3. Solution Transfer
4. Compensated Evaluation
5. Modified Newton's for Intersection

Introduction and motivation

Method of Characteristics

Method of Characteristics

Solve simple transport equation

$$u_t + cu_x = 0, \quad u(x,0) = u_0(x).$$

Method of Characteristics

Solve simple transport equation

$$u_t + cu_x = 0, \quad u(x,0) = u_0(x).$$

Divide physical domain

$$x(t) = x_0 + ct$$

Method of Characteristics

Solve simple transport equation

$$u_t + cu_x = 0, \quad u(x,0) = u_0(x).$$

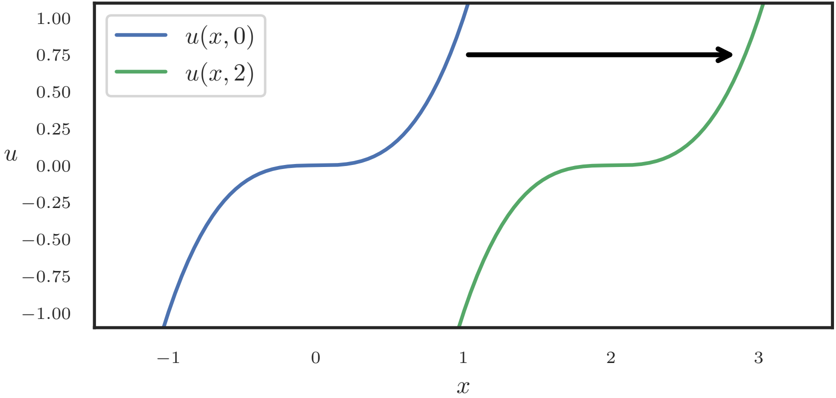
Divide physical domain

$$x(t) = x_0 + ct$$

PDE becomes a (trivial) ODE

$$\frac{d}{dt}u(x(t),t) = 0.$$

Method of Characteristics



Lagrangian Methods

Lagrangian Methods

- Each point in physical domain is a **particle**

Lagrangian Methods

- Each point in physical domain is a **particle**
- Carry value (e.g. heat, pressure, density) along characteristic curve

Lagrangian Methods

- Each point in physical domain is a **particle**
- Carry value (e.g. heat, pressure, density) along characteristic curve
- Transform PDE to family of ODEs

Add viscosity term to transport equation

$$u_t + cu_x - \varepsilon u_{xx} = 0.$$

Lagrangian Methods

Add viscosity term to transport equation

$$u_t + cu_x - \varepsilon u_{xx} = 0.$$

Same characteristics used, **but** solution no longer constant

Lagrangian Methods

Add viscosity term to transport equation

$$u_t + cu_x - \varepsilon u_{xx} = 0.$$

Same characteristics used, **but** solution no longer constant

$$\frac{d}{dt} u(x(t), t) = \varepsilon u_{xx}.$$

Remeshing and Adaptivity

- Problems caused by flow-based mesh changes

Remeshing and Adaptivity

- Problems caused by flow-based mesh changes
    - Distortion

Remeshing and Adaptivity

- Problems caused by flow-based mesh changes
    - Distortion
    - Tangling

Remeshing and Adaptivity

- Problems caused by flow-based mesh changes
  - Distortion
  - Tangling
  - Travel outside relevant physical domain

Remeshing and Adaptivity

- Problems caused by flow-based mesh changes
    - Distortion
    - Tangling
    - Travel outside relevant physical domain
- Adaptivity

Remeshing and Adaptivity

- Problems caused by flow-based mesh changes
  - Distortion
  - Tangling
  - Travel outside relevant physical domain
- Adaptivity
  - Dynamically focus computational effort

Remeshing and Adaptivity

- Problems caused by flow-based mesh changes
  - Distortion
  - Tangling
  - Travel outside relevant physical domain
- Adaptivity
  - Dynamically focus computational effort
  - Resolve sensitive features

Remeshing Example

Consider

$$u_t + \begin{bmatrix} y^2 \\ 1 \end{bmatrix} \cdot \nabla u + F(u, \nabla u) = 0$$

Remeshing Example

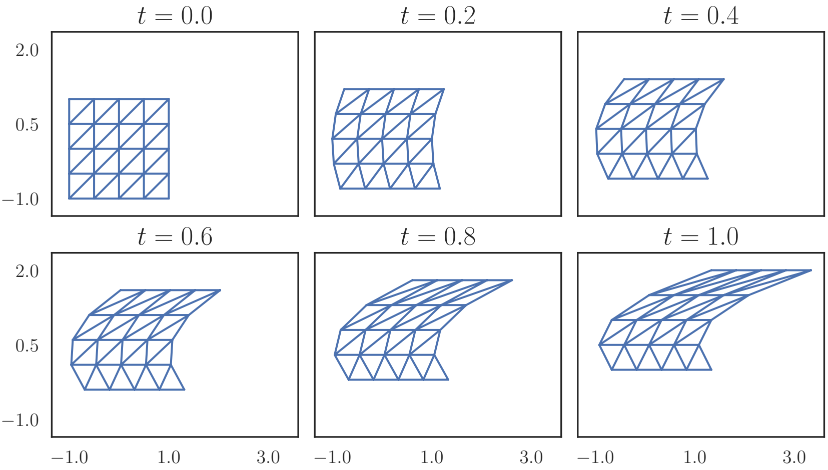
Consider

$$u_t + \begin{bmatrix} y^2 \\ 1 \end{bmatrix} \cdot \nabla u + F(u, \nabla u) = 0$$

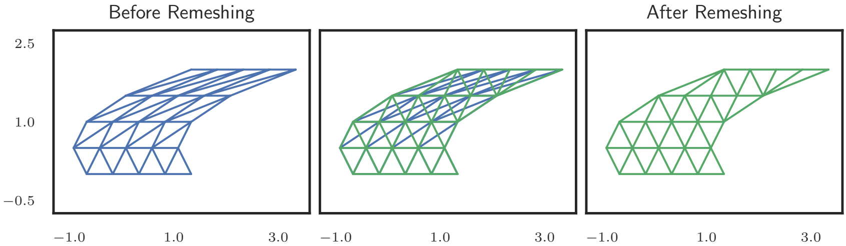
with cubic characteristics

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \frac{1}{3} \begin{bmatrix} (y_0 + t)^3 - y_0^3 \\ 3t \end{bmatrix}.$$

Remeshing Example



Remeshing Example



Curved Meshes

- Benefits

Curved Meshes

- Benefits
  - High-order shape functions, highly accurate solutions

Curved Meshes

- Benefits
  - High-order shape functions, highly accurate solutions
  - Low dissipation and dispersion error

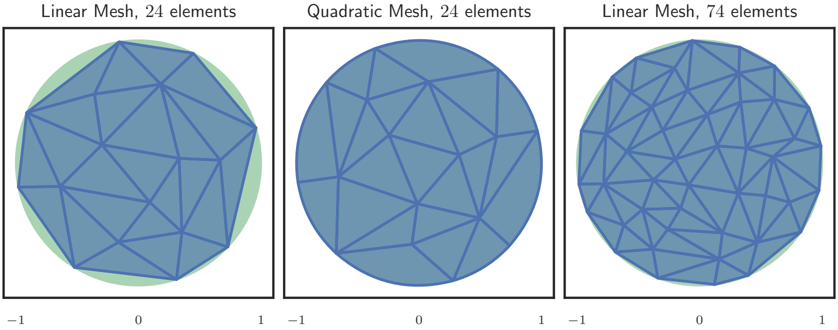
Curved Meshes

- Benefits
    - High-order shape functions, highly accurate solutions
    - Low dissipation and dispersion error
    - Greater geometric flexibility

Curved Meshes

- Benefits
  - High-order shape functions, highly accurate solutions
  - Low dissipation and dispersion error
  - Greater geometric flexibility
  - Fewer elements

Curved Meshes



Curved Meshes

- Benefits
  - High-order shape functions, highly accurate solutions
  - Low dissipation and dispersion error
  - Greater geometric flexibility
  - Fewer elements

Curved Meshes

- Benefits
  - High-order shape functions, highly accurate solutions
  - Low dissipation and dispersion error
  - Greater geometric flexibility
  - Fewer elements
- Drawbacks

Curved Meshes

- Benefits
  - High-order shape functions, highly accurate solutions
  - Low dissipation and dispersion error
  - Greater geometric flexibility
  - Fewer elements
- Drawbacks
  - Harder to implement

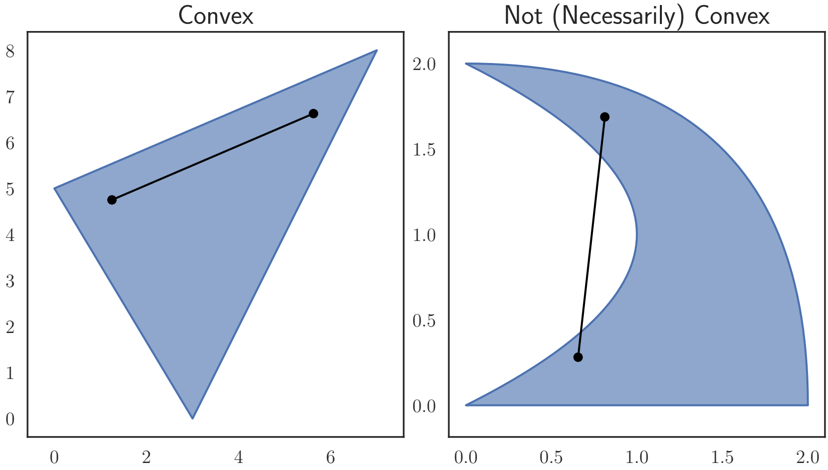
Curved Meshes

- Benefits
  - High-order shape functions, highly accurate solutions
  - Low dissipation and dispersion error
  - Greater geometric flexibility
  - Fewer elements
- Drawbacks
  - Harder to implement
  - Loss of accuracy in high degree (e.g. Runge's phenomenon)

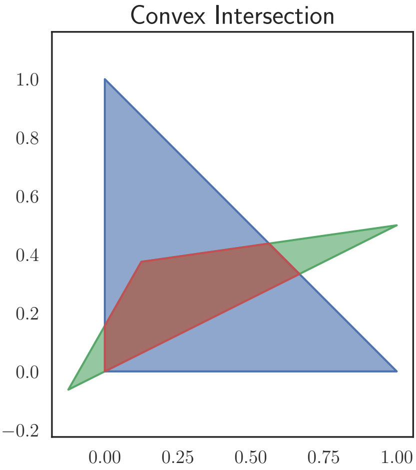
Curved Meshes

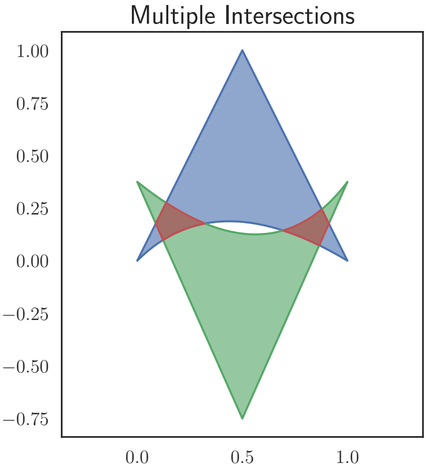
- Benefits
    - High-order shape functions, highly accurate solutions
    - Low dissipation and dispersion error
    - Greater geometric flexibility
    - Fewer elements
- Drawbacks
    - Harder to implement
    - Loss of accuracy in high degree (e.g. Runge's phenomenon)
    - More challenging geometry

Curved Meshes



Curved Meshes





Curved Elements

- Necessary for High-order

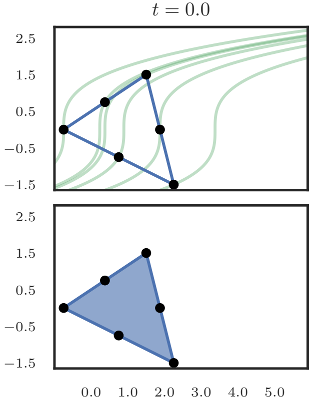
Curved Elements

- Necessary for High-order
- With non-linear shape functions (i.e. not straight sided), non-vertex nodes used

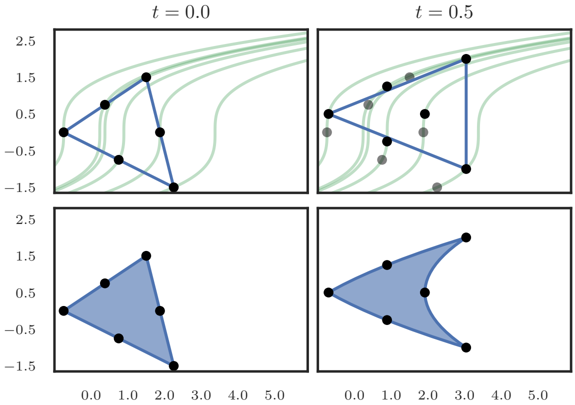
Curved Elements

- Necessary for High-order
- With non-linear shape functions (i.e. not straight sided), non-vertex nodes used
- Lagrangian method must either curve mesh or information about flow of geometry will be lost

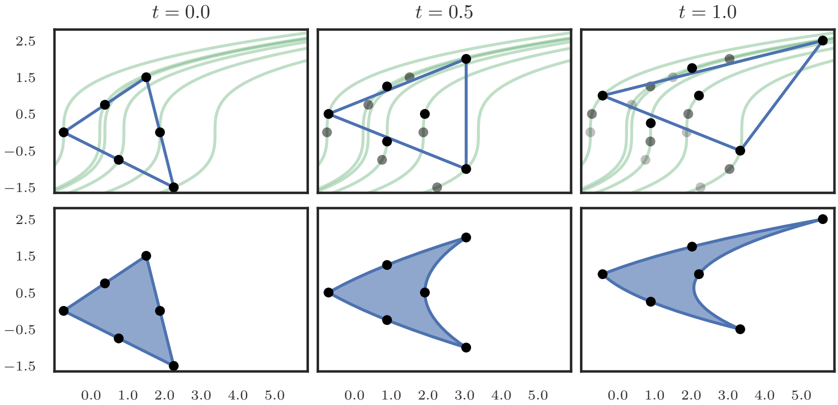
Curved Elements: Necessary for High-order



Curved Elements: Necessary for High-order



Curved Elements: Necessary for High-order



Curved Elements

Bézier Triangles

- Image $\mathcal{T} = b\,(\mathcal{U})$ of reference triangle under polynomial map $b(s, t)$

Bézier Triangles

- Image $\mathcal{T} = b\,(\mathcal{U})$ of reference triangle under polynomial map $b(s, t)$
- Barycentric coordinates $\lambda_1 = 1 - s - t, \lambda_2 = s, \lambda_3 = t$

Bézier Triangles

- Image $\mathcal{T} = b\,(\mathcal{U})$ of reference triangle under polynomial map $b(s, t)$
- Barycentric coordinates $\lambda_1 = 1 - s - t, \lambda_2 = s, \lambda_3 = t$
- Bernstein basis via trinomial expansion:

$$1 = (\lambda_1 + \lambda_2 + \lambda_3)^n$$

Bézier Triangles

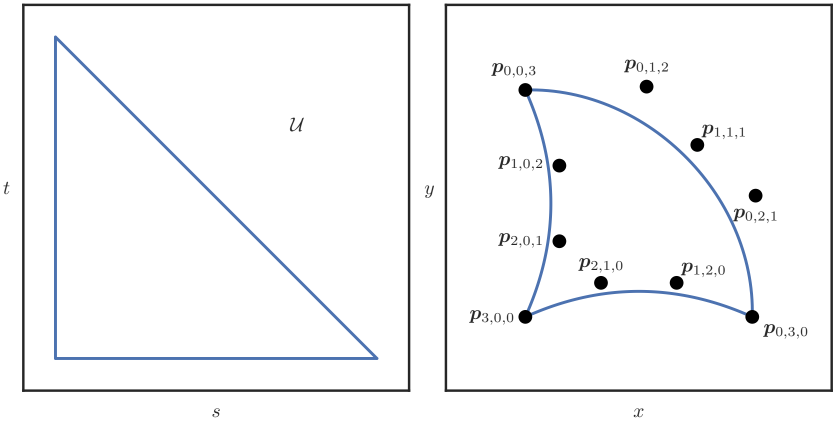
- Image $\mathcal{T} = b\,(\mathcal{U})$ of reference triangle under polynomial map $b(s,t)$
- Barycentric coordinates $\lambda_1 = 1 - s - t, \lambda_2 = s, \lambda_3 = t$
- Bernstein basis via trinomial expansion:

$$1 = (\lambda_1 + \lambda_2 + \lambda_3)^n$$

- Convex combination of control points

$$b(s,t) = \sum_{\substack{i+j+k=n\\ i,j,k\geq 0}} \binom{n}{i,\,j,\,k} \lambda_1^i \lambda_2^j \lambda_3^k \, \boldsymbol{p}_{i,j,k}$$

Bézier Triangles



Bézier Triangles

- $b(s,t)$ can be defined by data other than control net

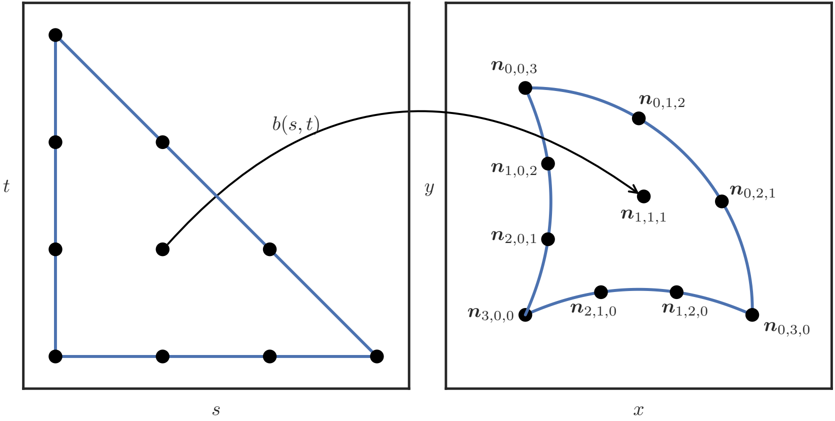
Bézier Triangles

- $b(s, t)$ can be defined by data other than control net
- Regular grid in $\mathcal{U}$, $\boldsymbol{u}_{i,j,k} = \left(\frac{j}{n}, \frac{k}{n}\right)$

Bézier Triangles

- $b(s,t)$ can be defined by data other than control net
- Regular grid in $\mathcal{U}$, $\boldsymbol{u}_{i,j,k} = \left(\frac{j}{n}, \frac{k}{n}\right)$
- $b\left(\boldsymbol{u}_{i,j,k}\right) = \boldsymbol{n}_{i,j,k}$; refer to $\boldsymbol{n}_{i,j,k}$ as **standard nodes**

Bézier Triangles



Bézier Triangles

- $b(s, t)$ can be defined by data other than control net
- Regular grid in $\mathcal{U}$, $\boldsymbol{u}_{i,j,k} = \left(\frac{j}{n}, \frac{k}{n}\right)$
- $b\left(\boldsymbol{u}_{i,j,k}\right) = \boldsymbol{n}_{i,j,k}$; refer to $\boldsymbol{n}_{i,j,k}$ as **standard nodes**

Bézier Triangles

- $b(s, t)$ can be defined by data other than control net
- Regular grid in $\mathcal{U}$, $\boldsymbol{u}_{i,j,k} = \left(\frac{j}{n}, \frac{k}{n}\right)$
- $b\left(\boldsymbol{u}_{i,j,k}\right) = \boldsymbol{n}_{i,j,k}$; refer to $\boldsymbol{n}_{i,j,k}$ as **standard nodes**
- For example, taking $\boldsymbol{n}_{i,j,k} = \delta_{(i,j,k)\ (i_0,j_0,k_0)}$ gives degree n shape functions on $\mathcal{U}$

Bézier Triangles

- $b(s, t)$ can be defined by data other than control net
- Regular grid in $\mathcal{U}$, $\boldsymbol{u}_{i,j,k} = \left(\frac{j}{n}, \frac{k}{n}\right)$
- $b\left(\boldsymbol{u}_{i,j,k}\right) = \boldsymbol{n}_{i,j,k}$; refer to $\boldsymbol{n}_{i,j,k}$ as **standard nodes**
- For example, taking $\boldsymbol{n}_{i,j,k} = \delta_{(i,j,k)\ (i_0,j_0,k_0)}$ gives degree n shape functions on $\mathcal{U}$
- Conversion between $\boldsymbol{n}_{i,j,k}$ and $\boldsymbol{p}_{i,j,k}$ has condition number exponential in n

Valid Element

- Element $\mathcal{T}$ is **valid** if diffeomorphic to $\mathcal{U}$

Valid Element

- Element $\mathcal{T}$ is **valid** if diffeomorphic to $\mathcal{U}$
- $b(s, t)$ bijective, i.e. Jacobian Db is everywhere invertible

Valid Element

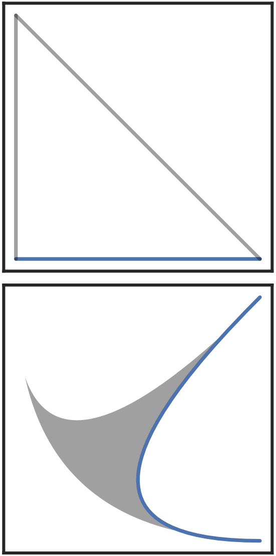
- Element $\mathcal{T}$ is **valid** if diffeomorphic to $\mathcal{U}$
- $b(s,t)$ bijective, i.e. Jacobian Db is everywhere invertible
- $\det(Db)$ positive, preserves orientation

Inverted Element

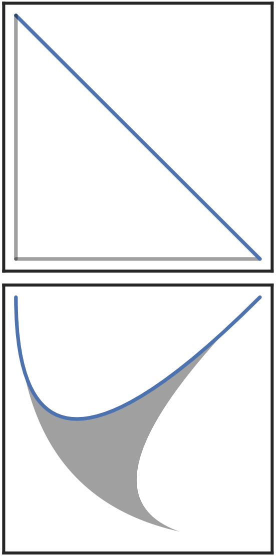
Consider element given by map

$$b(s,t) = \lambda_1^2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \lambda_2^2 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \lambda_3^2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

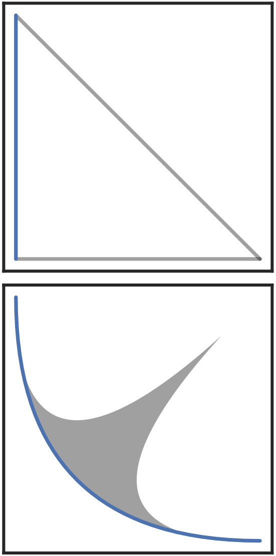
Inverted Element



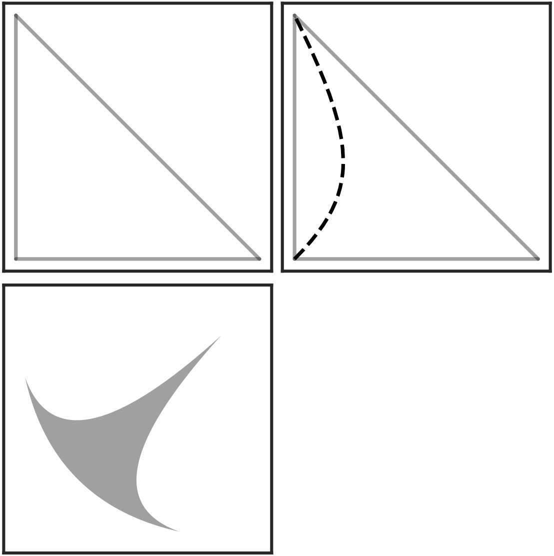
Inverted Element



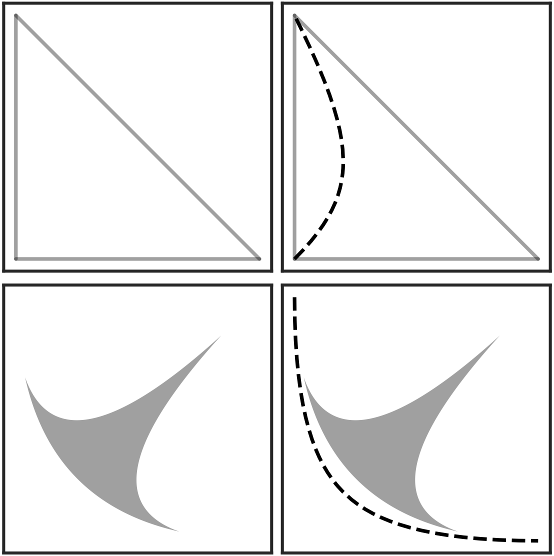
Inverted Element



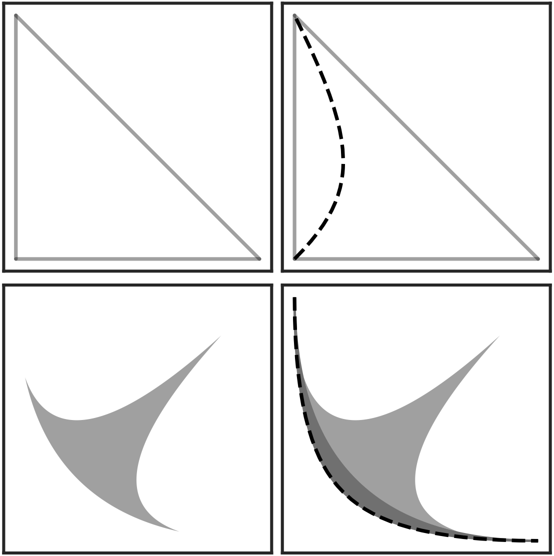
Inverted Element



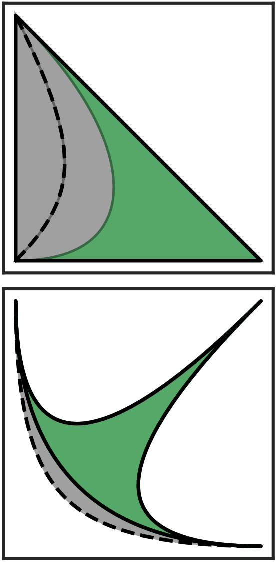
Inverted Element



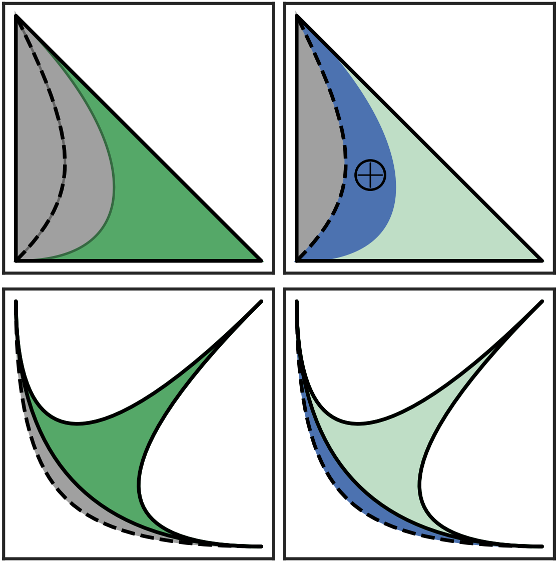
Inverted Element



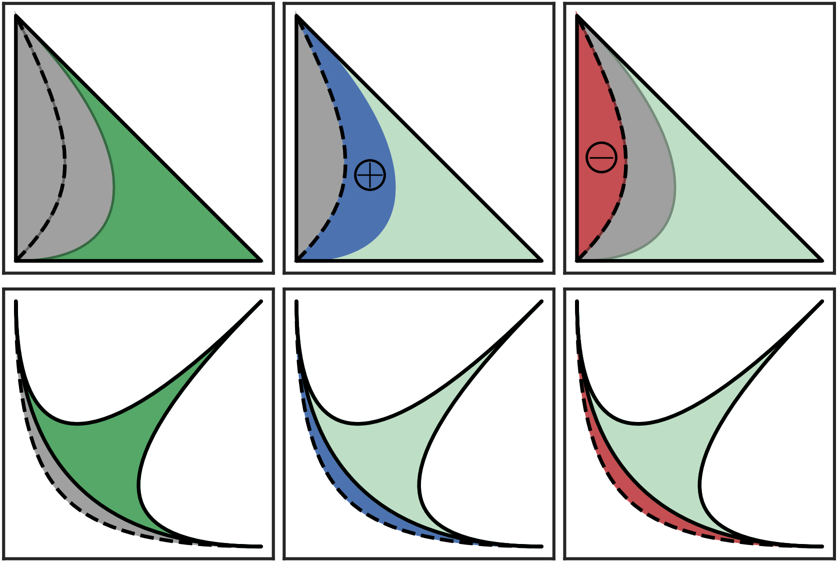
Inverted Element



Inverted Element



Inverted Element



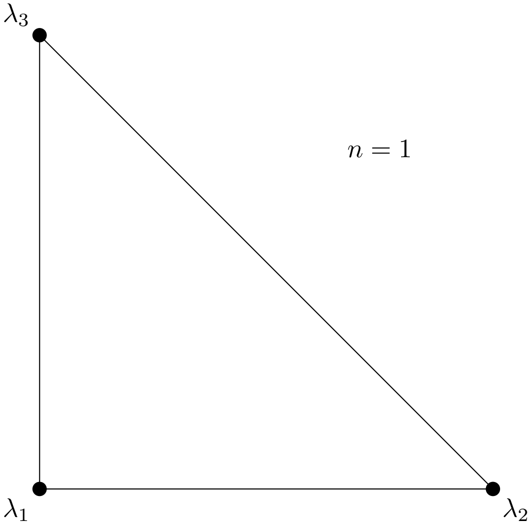
Shape Functions

- Based on $\boldsymbol{u}_{\boldsymbol{\alpha}} \in \mathcal{U}$ or $\boldsymbol{n}_{\boldsymbol{\alpha}} \in \mathbf{R}^2$ ($\boldsymbol{\alpha}$ is a multi-index)

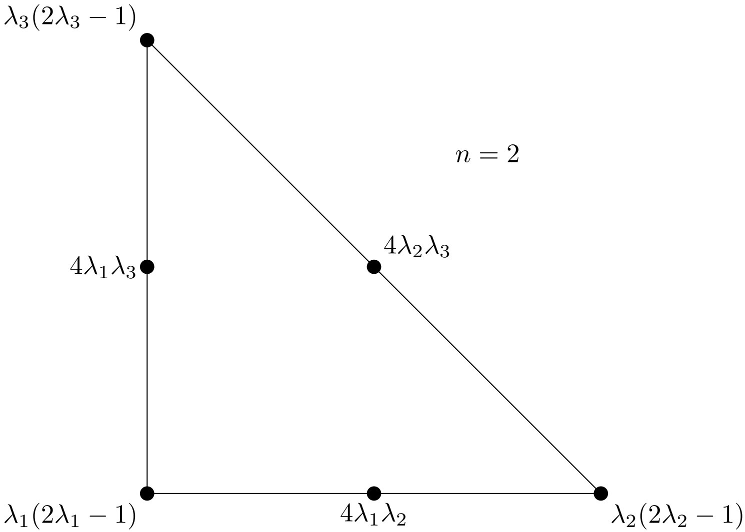
Shape Functions

- Based on $\boldsymbol{u}_{\boldsymbol{\alpha}} \in \mathcal{U}$ or $\boldsymbol{n}_{\boldsymbol{\alpha}} \in \mathbf{R}^2$ ($\boldsymbol{\alpha}$ is a multi-index)
- Pre-Image Basis: $\phi_{\boldsymbol{\alpha}}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(\boldsymbol{u}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(b^{-1}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right)\right)$

Shape Functions



Shape Functions



Shape Functions

- Based on $\boldsymbol{u}_{\boldsymbol{\alpha}} \in \mathcal{U}$ or $\boldsymbol{n}_{\boldsymbol{\alpha}} \in \mathbf{R}^2$ ($\boldsymbol{\alpha}$ is a multi-index)
- Pre-Image Basis: $\phi_{\boldsymbol{\alpha}}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(\boldsymbol{u}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(b^{-1}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right)\right)$

Shape Functions

- Based on $\boldsymbol{u}_{\boldsymbol{\alpha}} \in \mathcal{U}$ or $\boldsymbol{n}_{\boldsymbol{\alpha}} \in \mathbf{R}^2$ ($\boldsymbol{\alpha}$ is a multi-index)
- Pre-Image Basis: $\phi_{\boldsymbol{\alpha}}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(\boldsymbol{u}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(b^{-1}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right)\right)$
- Global Coordinates Basis: $\phi_{\boldsymbol{\alpha}}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right) = \delta_{\boldsymbol{\alpha}\boldsymbol{\beta}}$

Shape Functions

- Based on $\boldsymbol{u}_{\boldsymbol{\alpha}} \in \mathcal{U}$ or $\boldsymbol{n}_{\boldsymbol{\alpha}} \in \mathbf{R}^2$ ($\boldsymbol{\alpha}$ is a multi-index)
- Pre-Image Basis: $\phi_{\boldsymbol{\alpha}}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(\boldsymbol{u}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(b^{-1}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right)\right)$
- Global Coordinates Basis: $\phi_{\boldsymbol{\alpha}}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right) = \delta_{\boldsymbol{\alpha\beta}}$
- Element is **isoparametric** when numerical solution expressed in span of shape functions

Shape Functions

- Based on $\boldsymbol{u}_{\boldsymbol{\alpha}} \in \mathcal{U}$ or $\boldsymbol{n}_{\boldsymbol{\alpha}} \in \mathbf{R}^2$ ($\boldsymbol{\alpha}$ is a multi-index)
- **Pre-Image Basis**: $\phi_{\boldsymbol{\alpha}}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(\boldsymbol{u}_{\boldsymbol{\beta}}\right) = \widehat{\phi}_{\boldsymbol{\alpha}}\left(b^{-1}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right)\right)$
- **Global Coordinates Basis**: $\phi_{\boldsymbol{\alpha}}\left(\boldsymbol{n}_{\boldsymbol{\beta}}\right) = \delta_{\boldsymbol{\alpha}\boldsymbol{\beta}}$
- Element is **isoparametric** when numerical solution expressed in span of shape functions
- $\operatorname{supp}(\phi) = \mathcal{T}$

Solution Transfer

Galerkin Projection

- Given:

Galerkin Projection

- Given:
    - Donor mesh $\mathcal{M}_D$ and target mesh $\mathcal{M}_T$

Galerkin Projection

- Given:
  - Donor mesh $\mathcal{M}_D$ and target mesh $\mathcal{M}_T$
  - Shape function bases $\phi_D^{(j)}$ and $\phi_T^{(j)}$

Galerkin Projection

- **Given**:
  - Donor mesh $\mathcal{M}_D$ and target mesh $\mathcal{M}_T$
  - Shape function bases $\phi_D^{(j)}$ and $\phi_T^{(j)}$
  - Known discrete field $\boldsymbol{q}_D = \sum_j d_j \phi_D^{(j)}$

Galerkin Projection

- **Given**:
  - Donor mesh $\mathcal{M}_D$ and target mesh $\mathcal{M}_T$
  - Shape function bases $\phi_D^{(j)}$ and $\phi_T^{(j)}$
  - Known discrete field $\boldsymbol{q}_D = \sum_j d_j \phi_D^{(j)}$
- **Want**: L_2-optimal interpolant $\boldsymbol{q}_T = \sum_j t_j \phi_T^{(j)}$:

$$\|\boldsymbol{q}_T - \boldsymbol{q}_D\|_2 = \min_{\boldsymbol{q} \in \mathcal{V}_T} \|\boldsymbol{q} - \boldsymbol{q}_D\|_2$$

Galerkin Projection

Differentiating w.r.t. each t_j in $\boldsymbol{q}_T = \sum_j t_j \phi_T^{(j)}$ gives **weak form**

$$\int_\Omega \boldsymbol{q}_D \phi_T^{(j)}\, dV = \int_\Omega \boldsymbol{q}_T \phi_T^{(j)}\, dV, \qquad \text{for all } j.$$

Galerkin Projection

Differentiating w.r.t. each t_j in $\boldsymbol{q}_T = \sum_j t_j \phi_T^{(j)}$ gives **weak form**

$$\int_\Omega \boldsymbol{q}_D \phi_T^{(j)} \, dV = \int_\Omega \boldsymbol{q}_T \phi_T^{(j)} \, dV, \qquad \text{for all } j.$$

If $(\boldsymbol{x} \mapsto 1) \in \mathcal{V}_T$, then $\boldsymbol{q}_T$ is globally **conservative**

$$\int_\Omega \boldsymbol{q}_D \, dV = \int_\Omega \boldsymbol{q}_T \, dV.$$

Linear System

Weak form gives rise to a linear system in coefficients $\boldsymbol{d}$ and $\boldsymbol{t}$:

Linear System

Weak form gives rise to a linear system in coefficients $\boldsymbol{d}$ and $\boldsymbol{t}$:

$$M_T \boldsymbol{t} = M_{TD} \boldsymbol{d}.$$

Linear System

Weak form gives rise to a linear system in coefficients $\boldsymbol{d}$ and $\boldsymbol{t}$:

$$M_T \boldsymbol{t} = M_{TD} \boldsymbol{d}.$$

M_T is (symmetric) mass matrix for target mesh

$$(M_T)_{ij} = \int_\Omega \phi_T^{(i)} \phi_T^{(j)} \, dV.$$

Linear System

Each shape function ϕ has $\mathrm{supp}(\phi) = \mathcal{T}$ for some (curved) element, hence M_T is block diagonal in DG, sparse but globally coupled in CG.

Compensated Evaluation

Modified Newton's for Intersection