Fast Assembly of Bernstein-Bézier FEM

Oleg Davydov

University of Giessen, Germany

Splines and PDEs

CIME Summer School

Cetraro, July 3–7, 2017

Outline

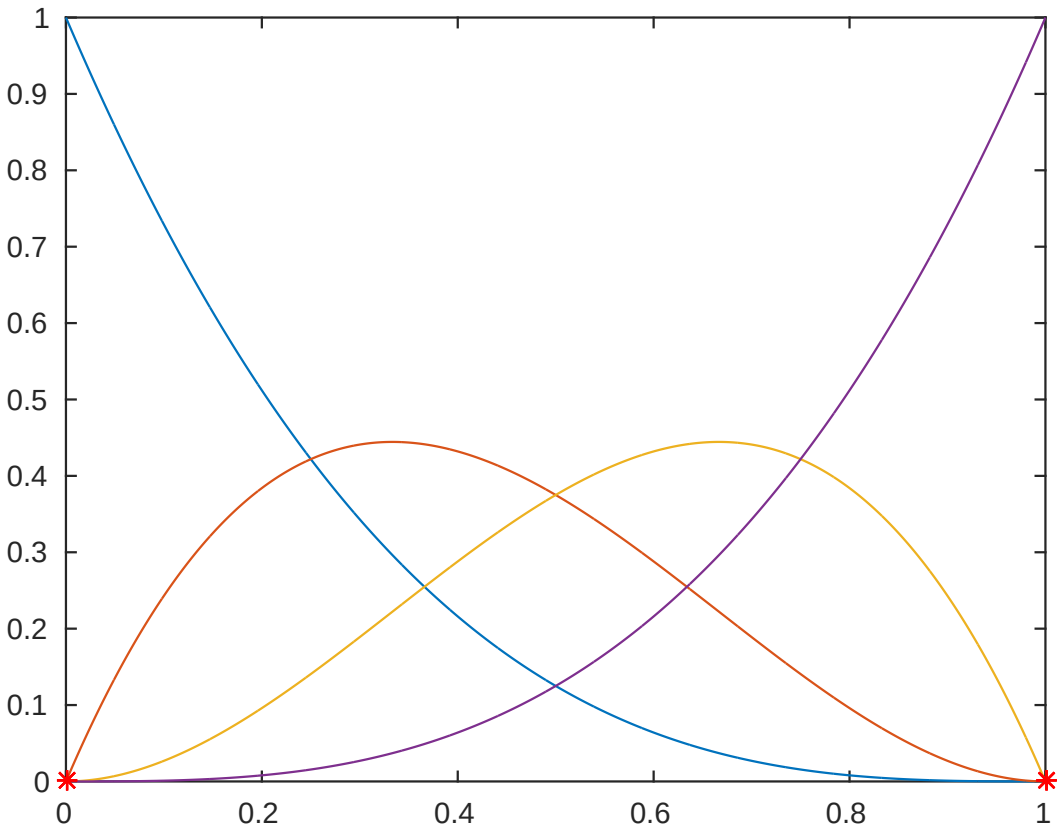
1. Bernstein Polynomials
2. FEM Assembly
3. Computation of Bernstein-Bézier Moments
4. Assembly of Element Matrices
5. Further Applications of BB Moments

Outline

1. Bernstein Polynomials
2. FEM Assembly
3. Computation of Bernstein-Bézier Moments
4. Assembly of Element Matrices
5. Further Applications of BB Moments

Bernstein Polynomials

- Definition in 1D: $B_i^n(x) = \binom{n}{i} x^i (1-x)^{n-i}, \quad i = 0, \ldots, n,$ a basis for the space of polynomials of degree n.



Bernstein Polynomials

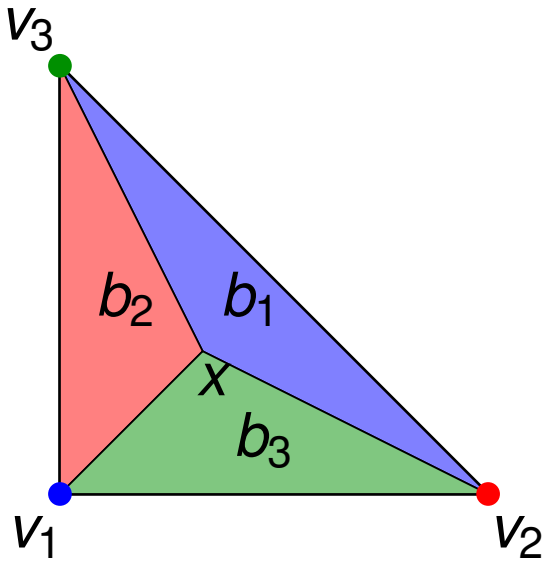
- A standard tool in curve and surface modelling:
  - Good algorithms for evaluation, differentiation, degree raising.
  - Optimal condition numbers among all non-negative bases.
  - Bernstein-Bézier techniques for smooth piecewise polynomials in 2D and 3D.
- Long ago suggested as shape functions in the finite element method (de Boor, Schumaker), alternative to Lagrange interpolation basis.
- Turn out to be optimal for the assembly of high order FEM system matrices on simplices[1]

[1] M. Ainsworth, G. Andriamaro and O. Davydov, Bernstein-Bézier finite elements of arbitrary order and optimal assembly procedures, *SIAM J. Sci. Comp.*, **33** (2011), 3087–3109.

Bernstein Polynomials

Bernstein basis polynomials on a d-simplex T

- $B_\xi^{n,d}(x) := \frac{n!}{\xi_1!\cdots\xi_{d+1}!} b_1^{\xi_1}(x)\cdots b_{d+1}^{\xi_{d+1}}(x),\ \xi \in \mathcal{I}_{n,d},\ x \in \mathbb{R}^d$
- $\mathcal{I}_{n,d} := \left\{\xi = (\xi_1,\ldots,\xi_{d+1}) \in \mathbb{Z}_+^{d+1} : |\xi| = n\right\},\ |\xi| = \sum_{i=1}^{d+1}\xi_i$
- $b_1(x),\ldots,b_{d+1}(x)$ – barycentric coordinates of x w.r.t. a simplex $T = [v_1,\ldots,v_{d+1}] \subset \mathbb{R}^d$



$b_i(x)$ are linear polynomials satisfying

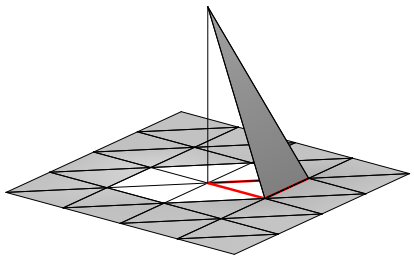
$$b_i(v_j) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise} \end{cases}$$

- $b_i(x)$ are linear $\Longrightarrow$ $B_\xi^{n,d}(x)$ is a polynomial of degree n

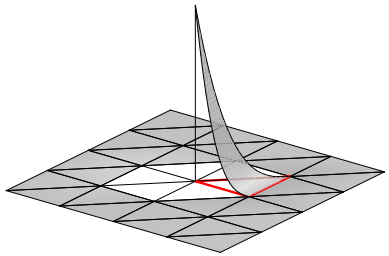
Bernstein Polynomials

Example: Bernstein polynomial on a triangle ($n = 3$, $d = 2$)

$b_1(x)$



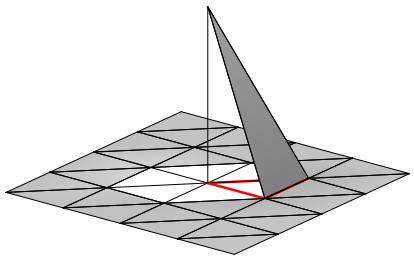
$B^{3,2}_{(3,0,0)} = b^3_1(x)$



Bernstein Polynomials

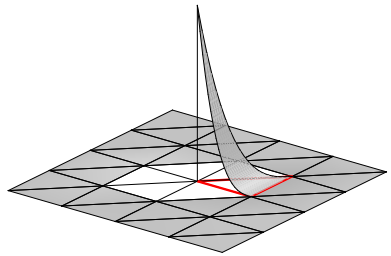
Example: Bernstein polynomial on a triangle ($n = 3$, $d = 2$)

$b_1(x)$

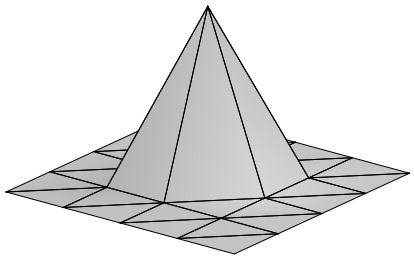


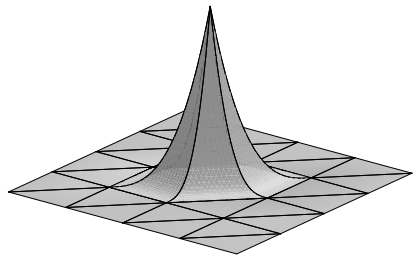
Courant hat function

$B^{3,2}_{(3,0,0)} = b^3_1(x)$



Bernstein-Bezier basis function





Bernstein Polynomials

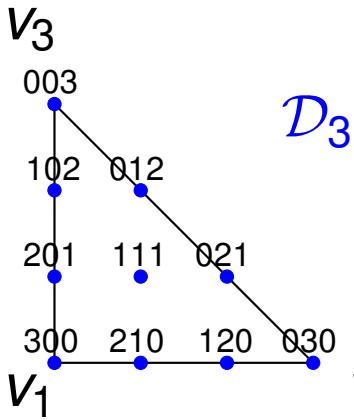
Domain points (Bernstein-Bézier techniques)

- $\mathcal{I}_{n,d} = \left\{\xi = (\xi_1, \ldots, \xi_{d+1}) \in \mathbb{Z}_+^{d+1} : \ |\xi| = n\right\}$ can be identified with the set of domain points

  $\mathcal{D}_{n,T} = \left\{p_\xi := \frac{1}{n}\sum_{i=1}^{d+1} \xi_i v_i\right\}_{\xi \in \mathcal{I}_{n,d}} \subset T = [v_1, \ldots, v_{d+1}].$

- Example: in 2D

  $\mathcal{D}_{n,T} = \left\{p_{(i,j,k)} := \frac{iv_1 + jv_2 + kv_3}{n}\right\}_{i+j+k=n} \subset T = [v_1, v_2, v_3].$



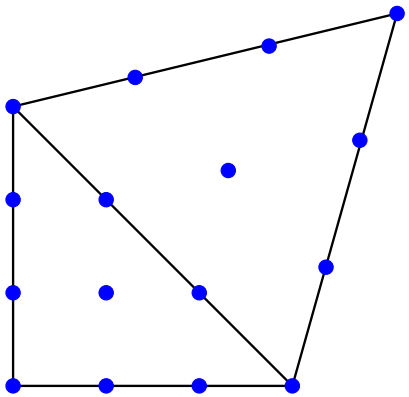
$\mathcal{D}_{3,T}$

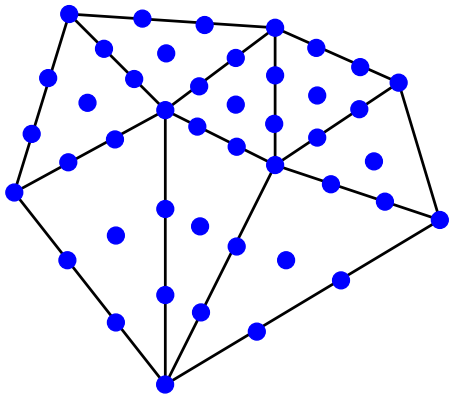
$$\mathcal{I}_{3,2} = \{(3,0,0),(0,3,0),(0,0,3),(2,1,0),(1,2,0),(0,2,1),(0,1,2),(2,0,1),(1,0,2),(1,1,1)\}$$

- Geometric splitting: “Degrees of freedom" located in p_ξ correspond to faces, edges, vertices, etc., which is important for the assembly of global system matrices.

Domain points

- We can identify domain points on faces shared by two or more simplices because the restriction of a d-dimensional Bernstein polynomial to a k-dimensional face is the k-dimensional Bernstein polynomial for the same domain point in this face

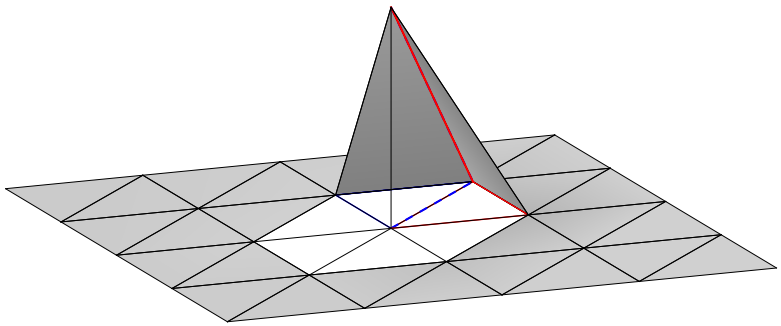




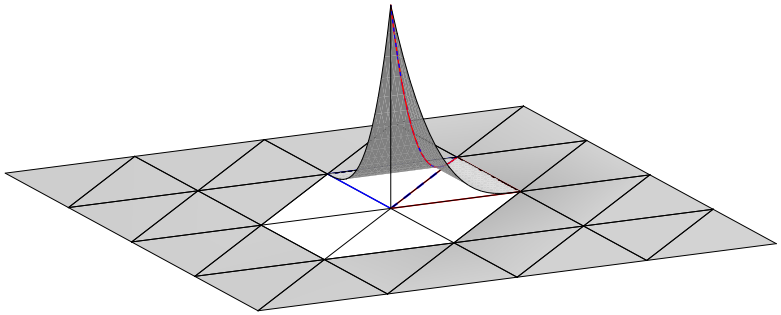
- Domain points coincide with interpolation points for Lagrange basis functions

Continuity across common faces

Linear shape functions

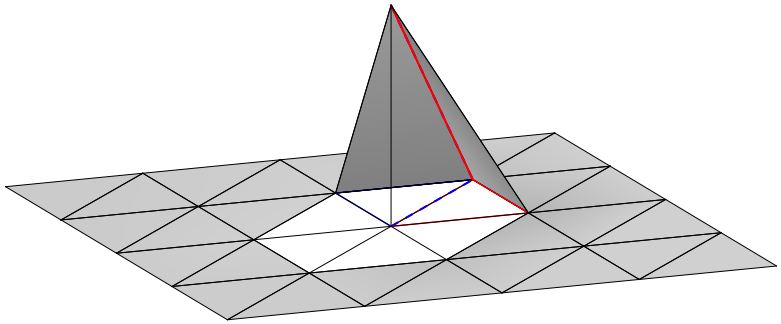


BB vertex shape functions

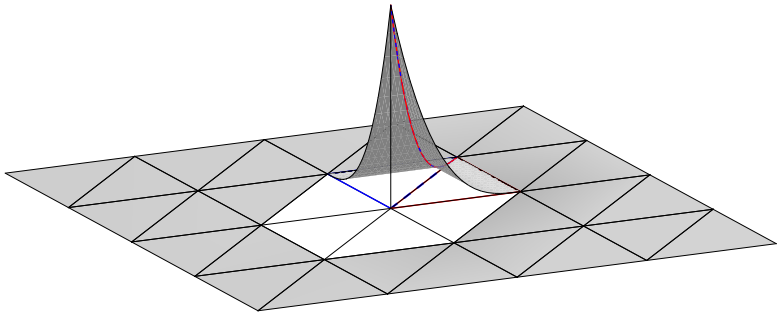


Continuity across common faces

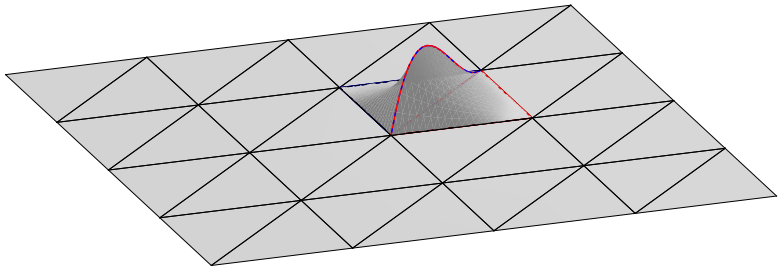
Linear shape functions



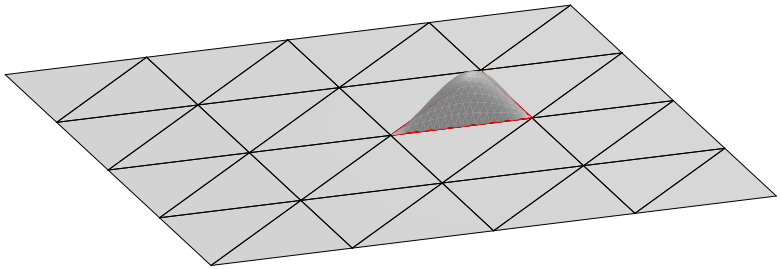
BB vertex shape functions



BB edge shape functions

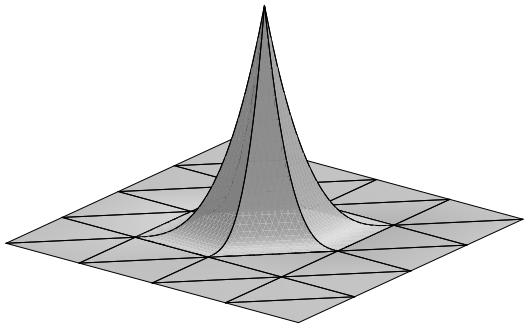


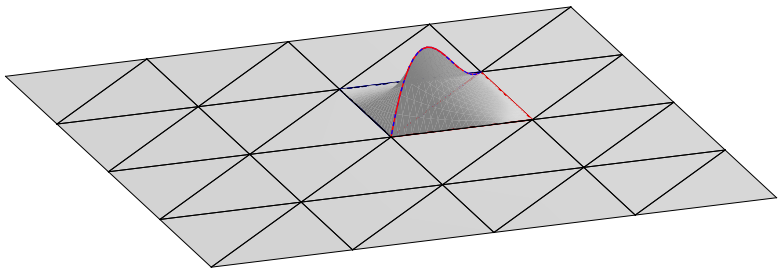
BB interior shape function

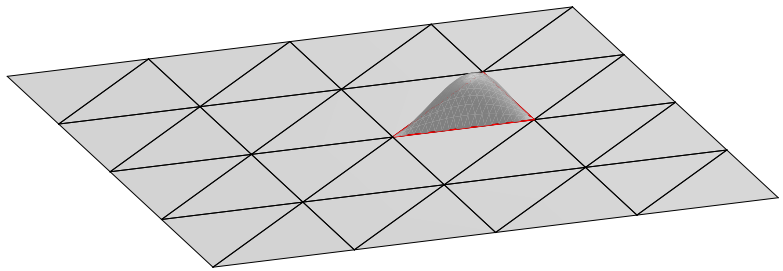


Bernstein Polynomials

Bernstein basis functions are C^0 B-splines on triangulations:

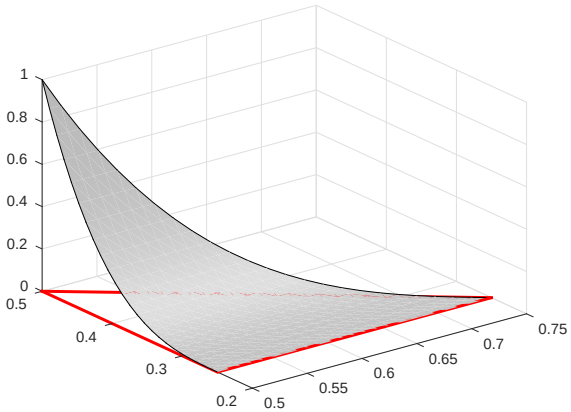




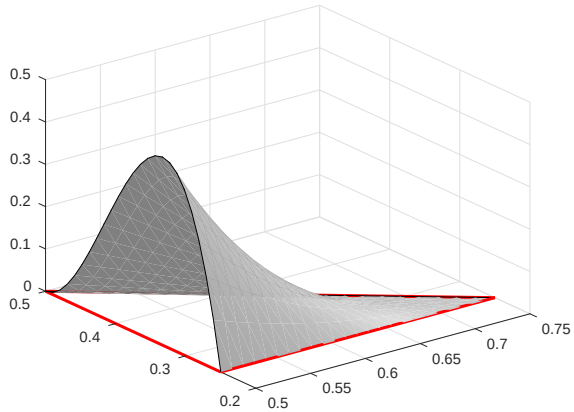


Bernstein versus Lagrange shape functions: degree 3

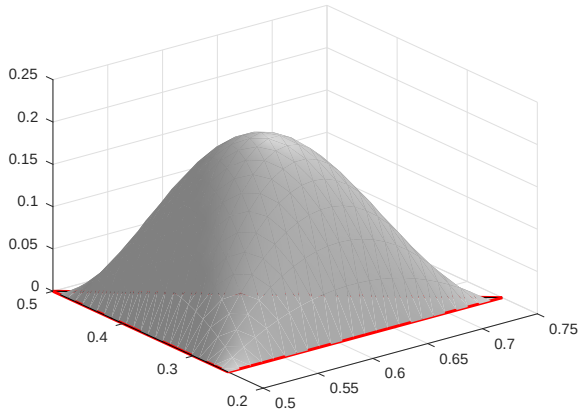
Vertex

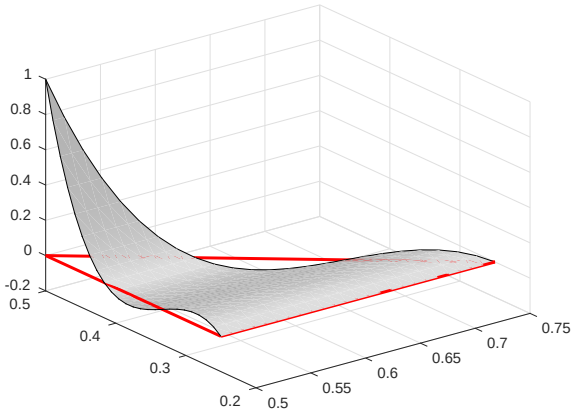


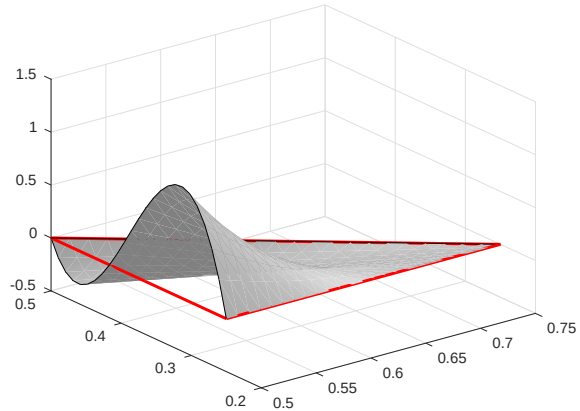
Edge

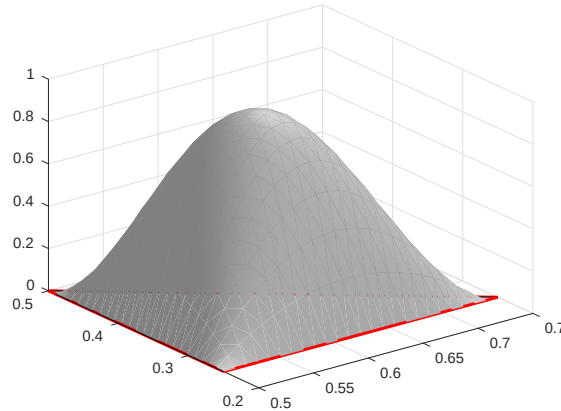


Internal



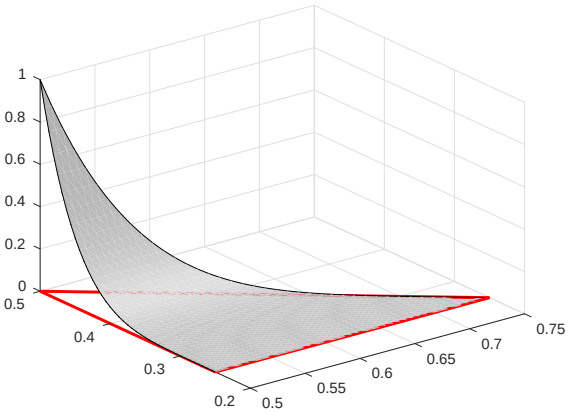


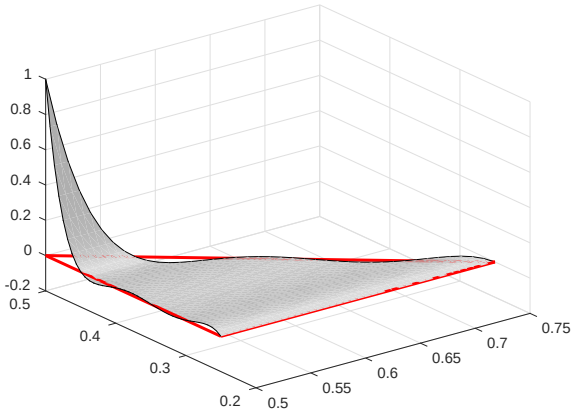




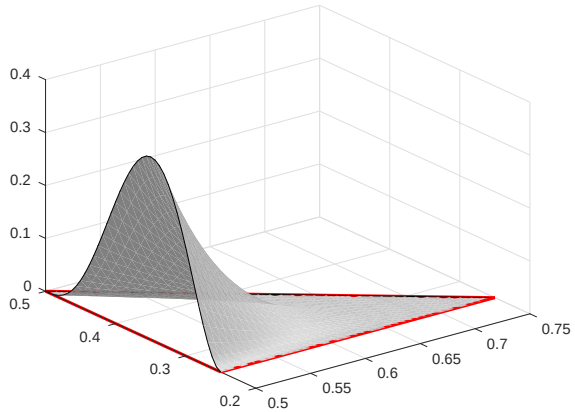
Bernstein versus Lagrange shape functions: degree 5

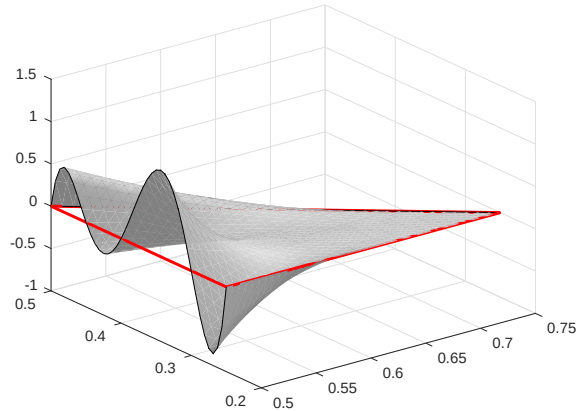
Vertex



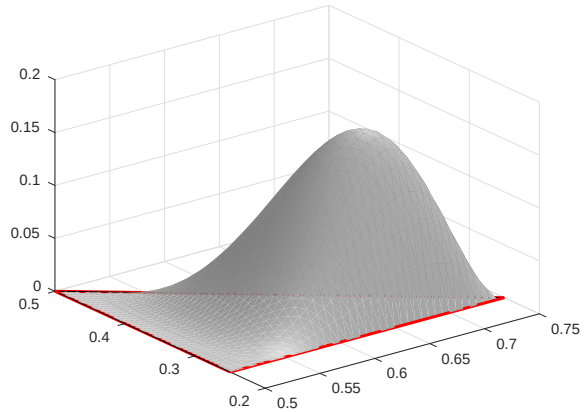


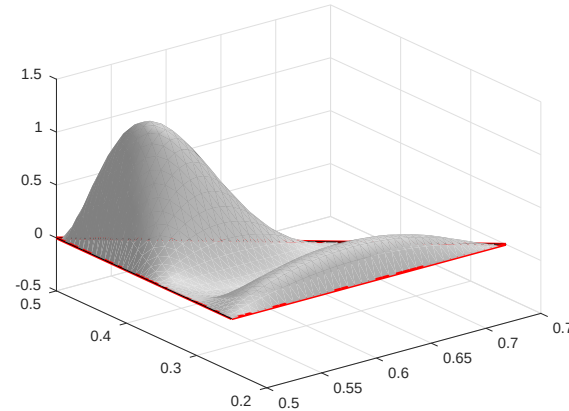
Edge





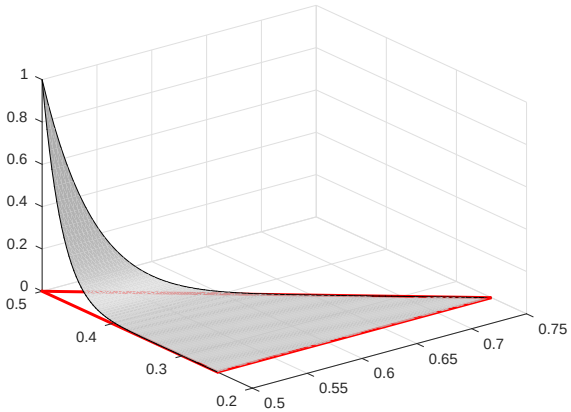
Internal

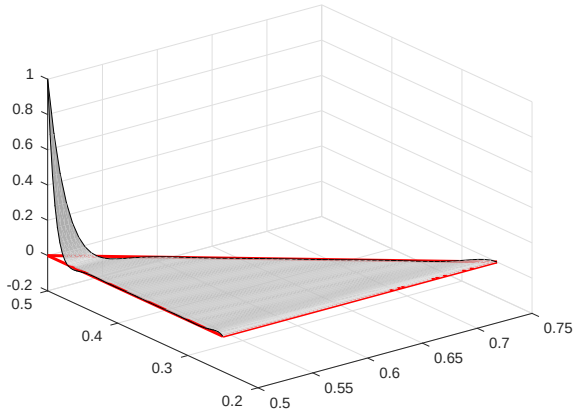




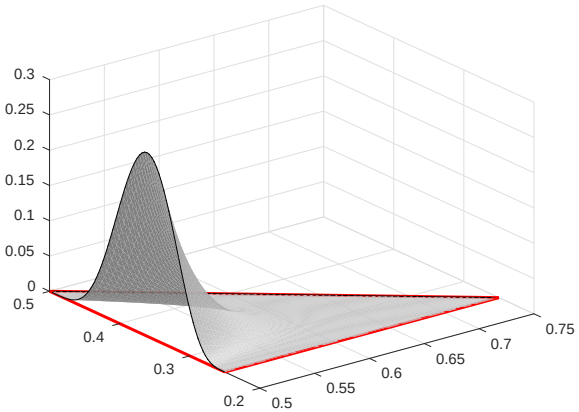
Bernstein versus Lagrange shape functions: degree 9

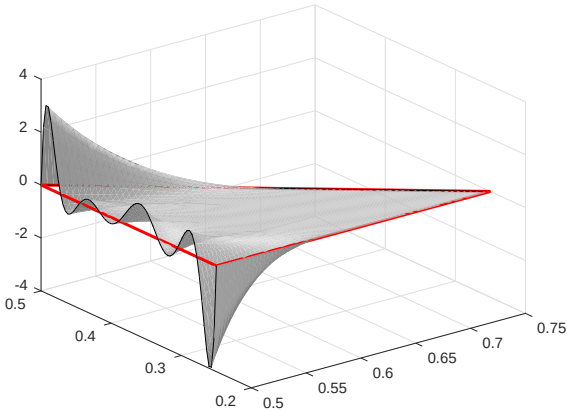
Vertex



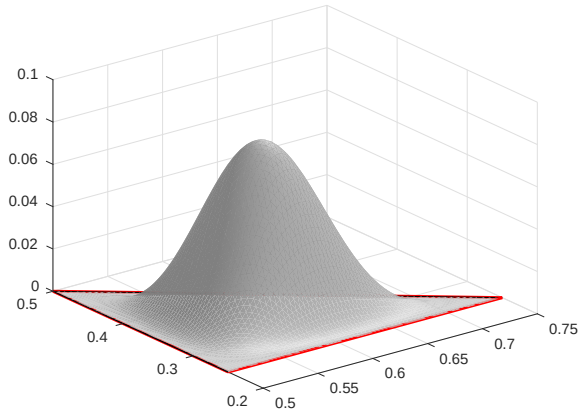


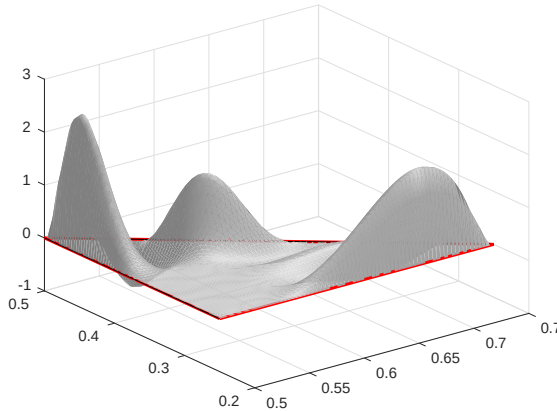
Edge





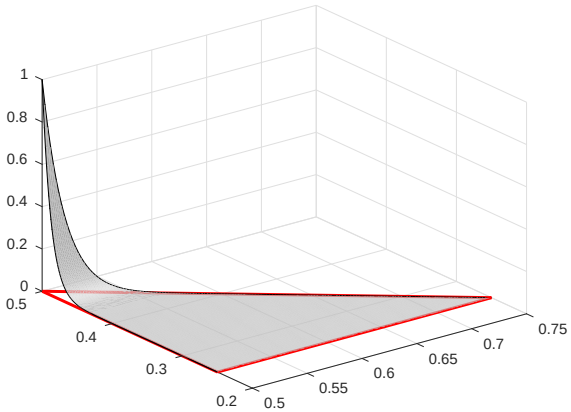
Internal

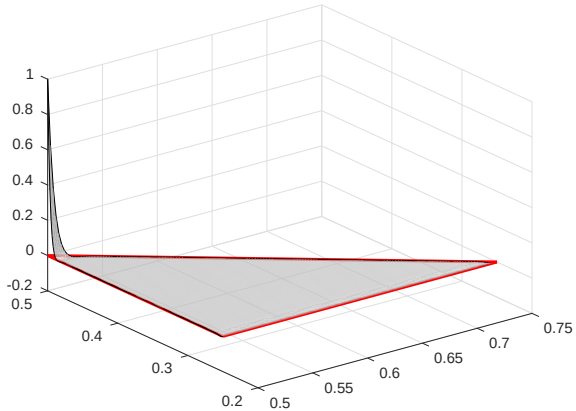




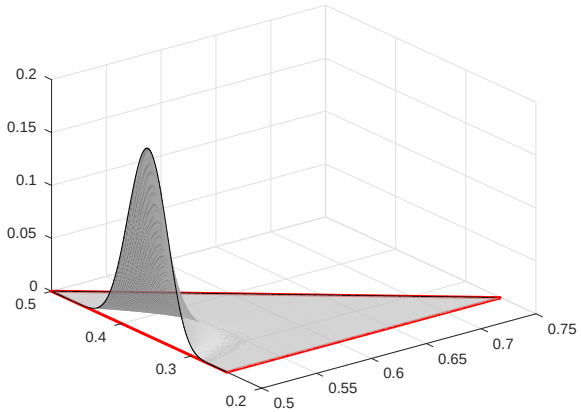
Bernstein versus Lagrange shape functions: degree 20

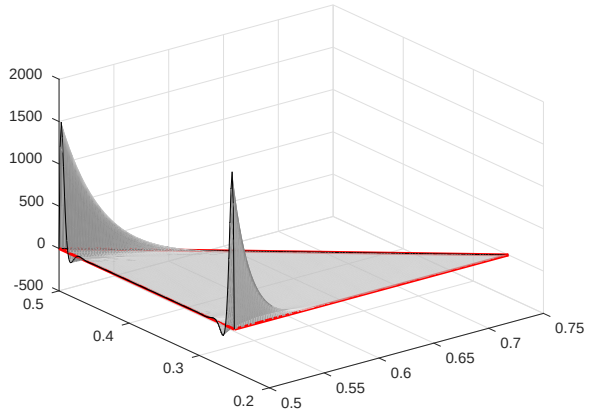
Vertex



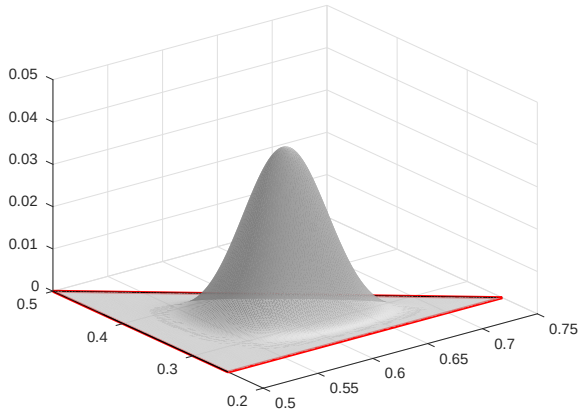


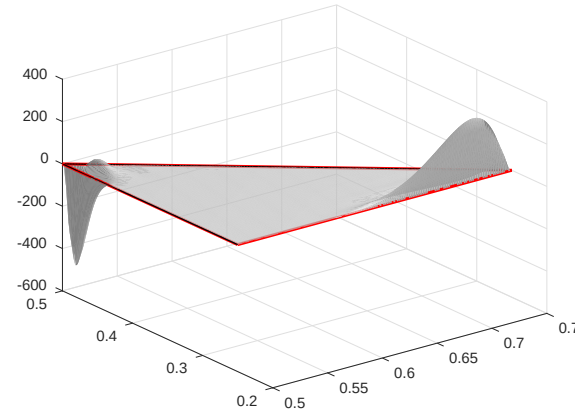
Edge





Internal





Bernstein Polynomials

Key properies for the fast computation of system matrices

- Product is a Bernstein polynomial:

$$B_{\xi}^{n,d} B_{\zeta}^{m,d} = \frac{\binom{n}{\xi}\binom{m}{\zeta}}{\binom{2n}{\xi+\zeta}} B_{\xi+\zeta}^{n+m,d}$$

Bernstein Polynomials

Key properies for the fast computation of system matrices

- Product is a Bernstein polynomial:

$$B_\xi^{n,d} B_\zeta^{m,d} = \frac{\binom{n}{\xi}\binom{m}{\zeta}}{\binom{2n}{\xi+\zeta}} B_{\xi+\zeta}^{n+m,d}$$

- Gradient is a sparse linear combination of Bernstein polynomials: $\nabla B_\xi^{n,d} = n \sum_{\alpha \in \mathcal{I}_{1,d}} B_{\xi-\alpha}^{n-1,d} \nabla b_\alpha$,

  sparse: the number of terms is independent of degree;
  ∇b_α are constant vectors depending on T;
  b_α are barycentric coordinates,
  numbered as Bernstein polynomials of degree 1:
  $b_\alpha := B_\alpha^{1,d}$, $\alpha \in \mathcal{I}_{1,d} = \{(1,0,\ldots,0),\ldots,(0,0,\ldots,1)\}$.

Bernstein Polynomials

Key properies for the fast computation of system matrices

- Product is a Bernstein polynomial:

$$B_\xi^{n,d} B_\zeta^{m,d} = \frac{\binom{n}{\xi}\binom{m}{\zeta}}{\binom{2n}{\xi+\zeta}} B_{\xi+\zeta}^{n+m,d}$$

- Gradient is a sparse linear combination of Bernstein polynomials: $\nabla B_\xi^{n,d} = n \sum_{\alpha \in \mathcal{I}_{1,d}} B_{\xi-\alpha}^{n-1,d} \nabla b_\alpha$,

  sparse: the number of terms is independent of degree;
  ∇b_α are constant vectors depending on T;
  b_α are barycentric coordinates,
  numbered as Bernstein polynomials of degree 1:
  $b_\alpha := B_\alpha^{1,d}$, $\alpha \in \mathcal{I}_{1,d} = \{(1,0,\ldots,0),\ldots,(0,0,\ldots,1)\}$.

- Hence other differential operators such as Hessian, curl, div also produce sparse linear combinations of Bernstein polynomials.

Outline

1. Bernstein Polynomials
2. FEM Assembly
3. Computation of Bernstein-Bézier Moments
4. Assembly of Element Matrices
5. Further Applications of BB Moments

Dirichlet problem for linear second order elliptic equation:

$$\begin{cases} -\text{div}\,(A\nabla u) + \mathbf{b}\cdot\nabla u + cu &= f \quad \text{in } \Omega, \\ u &= g \quad \text{on } \partial\Omega. \end{cases}$$

Weak formulation of the homogenuous problem ($g = 0$):
Find $u \in H_0^1(\Omega)$ such that for all $v \in H_0^1(\Omega)$

$$\int_\Omega \Big[\nabla u \cdot A\nabla v + (\mathbf{b}\cdot\nabla u)v + cuv\Big]\mathrm{d}x = \int_\Omega fv\mathrm{d}x. \quad (1)$$

FEM Assembly

Weak formulation of the homogenuous problem ($g = 0$):
Find $u \in H_0^1(\Omega)$ such that for all $v \in H_0^1(\Omega)$

$$\int_\Omega \Big[\nabla u \cdot A \nabla v + (\mathbf{b} \cdot \nabla u) v + cuv\Big] \mathrm{d}x = \int_\Omega fv \mathrm{d}x. \quad (1)$$

Galerkin method:

- Choose a finite dimensional subspace $\mathcal{P}$ of $H_0^1(\Omega)$ with basis $\{\phi_1, \ldots, \phi_N\}$.
- Replace u by a linear combination $u = \sum_{i=1}^N u_i \phi_i$ and require (1) to hold for each $v = \phi_j$, $j = 1, \ldots, N$.
- This results in a linear system w.r.t. $\{u_i\}_{i=1}^N$, with system matrices in LHS and load vector in RHS, with entries:

$$\int_\Omega \nabla \phi_i \cdot A \nabla \phi_j \, \mathrm{d}x, \quad \int_\Omega (\mathbf{b} \cdot \nabla \phi_i) \phi_j \mathrm{d}x, \quad \int_\Omega c \phi_i \phi_j \, \mathrm{d}x, \quad \int_\Omega f \phi_j \, \mathrm{d}x$$

FEM Assembly

Simplicial finite elements

- $\Omega \subset \mathbb{R}^d$ is triangulated into simplices ('elements').

FEM Assembly

Simplicial finite elements

- $\Omega \subset \mathbb{R}^d$ is triangulated into simplices ('elements').
- $\{\phi_i\}_{i=1}^N$ is a set of basis functions for the space of continuous piecewise polynomials of total degree n.

FEM Assembly

Simplicial finite elements

- $\Omega \subset \mathbb{R}^d$ is triangulated into simplices ('elements').
- $\{\phi_i\}_{i=1}^N$ is a set of basis functions for the space of continuous piecewise polynomials of total degree n.
- Assume that for each simplex T, nonzero restrictions $\phi_{T,\xi}$, $\xi \in I_T$, of the basis functions to T ('shape functions') form a basis for the space $\mathbb{P}_n$ of polynomials of degree n.

Assembling the linear system

- Compute for each element T:
  - (a) load vector $\int_T f\phi_{T,\xi}\,\mathrm{d}x, \quad \xi \in I_T,$
  - (b) mass matrix $\int_T c\phi_{T,\xi}\phi_{T,\zeta}\,\mathrm{d}x, \quad \xi,\zeta \in I_T,$
  - (c) stiffness matrix $\int_T \nabla\phi_{T,\xi}\cdot A\nabla\phi_{T,\zeta}\,\mathrm{d}x, \quad \xi,\zeta \in I_T,$
  - (d) convective matrix $\int_T (\mathbf{b}\cdot\nabla\phi_{T,\xi})\phi_{T,\zeta}\mathrm{d}x, \quad \xi,\zeta \in I_T.$

Assembling the linear system

- Compute for each element T:
  - (a) load vector $\int_T f\phi_{T,\xi}\,\mathrm{d}x, \quad \xi \in I_T,$
  - (b) mass matrix $\int_T c\phi_{T,\xi}\phi_{T,\zeta}\,\mathrm{d}x, \quad \xi,\zeta \in I_T,$
  - (c) stiffness matrix $\int_T \nabla\phi_{T,\xi}\cdot A\nabla\phi_{T,\zeta}\,\mathrm{d}x, \quad \xi,\zeta \in I_T,$
  - (d) convective matrix $\int_T (\mathbf{b}\cdot\nabla\phi_{T,\xi})\phi_{T,\zeta}\mathrm{d}x, \quad \xi,\zeta \in I_T.$
- Then the (i,j)-entry of e.g. the global stiffness matrix is

$$\int_\Omega \nabla\phi_i\cdot A\nabla\phi_j\,\mathrm{d}x = \sum_{T\subset \operatorname{supp}\phi_i\cap\operatorname{supp}\phi_j} \int_T \nabla\phi_{T,\xi_i}\cdot A\nabla\phi_{T,\xi_j}\,\mathrm{d}x,$$

where $\phi_{T,\xi_i} = \phi_i|_T, \quad \phi_{T,\xi_j} = \phi_j|_T.$

Assembling the linear system

- Compute for each element T:
  - (a) load vector $\int_T f\phi_{T,\xi}\,\mathrm{d}x, \quad \xi \in I_T,$
  - (b) mass matrix $\int_T c\phi_{T,\xi}\phi_{T,\zeta}\,\mathrm{d}x, \quad \xi,\zeta \in I_T,$
  - (c) stiffness matrix $\int_T \nabla\phi_{T,\xi}\cdot A\nabla\phi_{T,\zeta}\,\mathrm{d}x, \quad \xi,\zeta \in I_T,$
  - (d) convective matrix $\int_T (\mathbf{b}\cdot\nabla\phi_{T,\xi})\phi_{T,\zeta}\mathrm{d}x, \quad \xi,\zeta \in I_T.$
- Then the (i,j)-entry of e.g. the global stiffness matrix is

$$\int_\Omega \nabla\phi_i\cdot A\nabla\phi_j\,\mathrm{d}x = \sum_{T\subset \operatorname{supp}\phi_i\cap\operatorname{supp}\phi_j} \int_T \nabla\phi_{T,\xi_i}\cdot A\nabla\phi_{T,\xi_j}\,\mathrm{d}x,$$

  where $\phi_{T,\xi_i} = \phi_i|_T, \quad \phi_{T,\xi_j} = \phi_j|_T.$
- For Bernstein basis functions the load vector consists of Bernstein-Bézier moments $\int_T f(x)\,B_\xi^{n,d}(x)\,\mathrm{d}x$, $\xi \in \mathcal{I}_{n,d}$.

Outline

1. Bernstein Polynomials
2. FEM Assembly
3. Computation of Bernstein-Bézier Moments
4. Assembly of Element Matrices
5. Further Applications of BB Moments

Computation of Bernstein-Bézier Moments

Consider Bernstein-Bézier moments (of degree n):

$$\rho_{\xi}^{n,d}(f) = \int_T f(x)\, B_{\xi}^{n,d}(x)\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d}$$

Computation of Bernstein-Bézier Moments

Consider Bernstein-Bézier moments (of degree n):

$$\rho_\xi^{n,d}(f) = \int_T f(x)\, B_\xi^{n,d}(x)\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d}$$

- For constant data ($f(x) \equiv 1$),
  $$\int_T B_\xi^{n,d}\, \mathrm{d}x = \frac{|T|}{\binom{n+d}{d}}, \quad \text{independent of } \xi$$

Computation of Bernstein-Bézier Moments

Consider Bernstein-Bézier moments (of degree n):

$$\rho_\xi^{n,d}(f) = \int_T f(x)\, B_\xi^{n,d}(x)\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d}$$

- For constant data ($f(x) \equiv 1$),
  $\int_T B_\xi^{n,d}\, \mathrm{d}x = \dfrac{|T|}{\binom{n+d}{d}}$, independent of ξ
- Look for a quadrature to efficiently evaluate the moments $\{\rho_\xi^{n,d}(f) : \xi \in \mathcal{I}_{n,d}\}$, for any continuous f and large n

Computation of Bernstein-Bézier Moments

Consider Bernstein-Bézier moments (of degree n):

$$\rho_\xi^{n,d}(f) = \int_T f(x)\, B_\xi^{n,d}(x)\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d}$$

- For constant data ($f(x) \equiv 1$),
  $\int_T B_\xi^{n,d}\, \mathrm{d}x = \dfrac{|T|}{\binom{n+d}{d}}$, independent of ξ
- Look for a quadrature to efficiently evaluate the moments $\{\rho_\xi^{n,d}(f) : \xi \in \mathcal{I}_{n,d}\}$, for any continuous f and large n
- Hence, look for a quadrature $\int_T g(x)\, \mathrm{d}x \approx \sum_{i=1}^{N} w_i g(x_i)$ with $N \sim n^d$ centers, to be used with $g(x) := f(x)\, B_\xi^{n,d}(x)$.

Computation of Bernstein-Bézier Moments

Consider Bernstein-Bézier moments (of degree n):

$$\rho_\xi^{n,d}(f) = \int_T f(x)\, B_\xi^{n,d}(x)\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d}$$

- For constant data ($f(x) \equiv 1$),
  $\int_T B_\xi^{n,d}\, \mathrm{d}x = \dfrac{|T|}{\binom{n+d}{d}}$, independent of ξ
- Look for a quadrature to efficiently evaluate the moments $\{\rho_\xi^{n,d}(f) : \xi \in \mathcal{I}_{n,d}\}$, for any continuous f and large n
- Hence, look for a quadrature $\int_T g(x)\, \mathrm{d}x \approx \sum_{i=1}^{N} w_i g(x_i)$ with $N \sim n^d$ centers, to be used with $g(x) := f(x)\, B_\xi^{n,d}(x)$.
- Stroud conical product rule is especially well suited because it allows fast evaluation of the quadrature by sum factorization.

Sum factorization

- Assume we need to compute double sums

$$\Sigma_{k,\ell} = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{i,j}(k,\ell)\, f_{i,j} \quad \text{for all} \quad k,\ell = 1,\ldots,n.$$

  This requires in general $\sim n^4$ multiplications.

- If $c_{i,j}(k,\ell)$ is factorized as $c_{i,j}(k,\ell) = c_i(k)c_j(\ell)$, then

$$\Sigma_{k,\ell} = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{i,j}(k,\ell) f_{i,j} = \sum_{i=1}^{n} c_i(k) \sum_{j=1}^{n} c_j(\ell) f_{i,j}.$$

- Hence, the cost can be reduced to $2n^3$ by computing

1. $\sigma(i,\ell) = \sum_{j=1}^{n} c_j(\ell) f_{i,j}, \quad i,\ell = 1,\ldots,n.$ (n^3 multipl.)
2. $\Sigma_{k,\ell} = \sum_{i=1}^{n} c_i(k)\sigma(i,\ell), \quad k,\ell = 1,\ldots,n.$ (n^3 multipl.)

Sum factorization

- For multiple sums

$$\Sigma_{k_1,\ldots,k_d} = \sum_{i_1=1}^{n} \cdots \sum_{i_d=1}^{n} c_{i_1,\ldots,i_d}(k_1,\ldots,k_d)\, f_{i_1,\ldots,i_d}, \quad k_j = 1,\ldots,n,$$

  such that

$$c_{i_1,\ldots,i_d}(k_1,\ldots,k_d) = c_{i_1}(k_1)\cdots c_{i_d}(k_d),$$

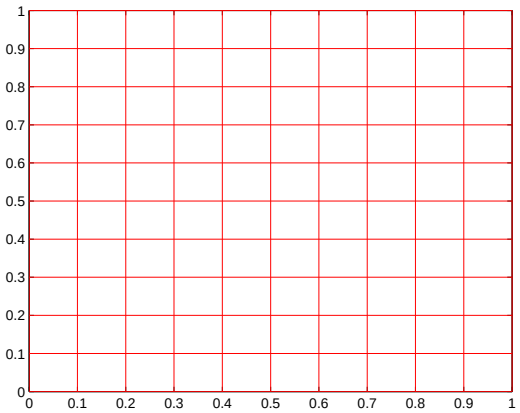
  the cost reduces from n^{2d} to dn^{d+1}.

- The method of sum factorization in FEM goes back to [Orszag, 1980] who used it with tensor-product shape functions on quadrilateral partitions.

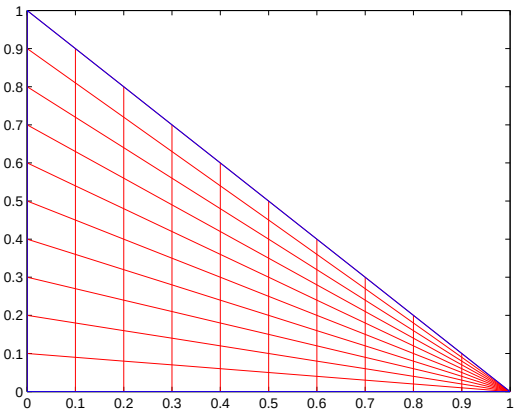
Duffy transformation

$$x(t) = \sum_{k=1}^{d+1} b_k x_k : [0,1]^d \to T = \langle x_1, \ldots, x_{d+1} \rangle$$

$$b_1 = t_1, \quad b_2 = t_2(1 - b_1), \quad b_3 = t_3(1 - b_1 - b_2), \quad \cdots,$$
$$b_d = t_d(1 - b_1 - \cdots - b_{d-1}), \quad b_{d+1} = (1 - b_1 - \cdots - b_d)$$



$\Longrightarrow$



Computation of Bernstein-Bézier Moments

Integral over simplex T reduced to box $[0,1]^d$

$$\int_T f(x)\mathrm{d}x = d!|T| \int_0^1 dt_1(1-t_1)^{d-1} \int_0^1 dt_2(1-t_2)^{d-2} \cdots \int_0^1 dt_d f(x(t))$$

Computation of Bernstein-Bézier Moments

Integral over simplex T reduced to box $[0,1]^d$

$$\int_T f(x)\mathrm{d}x = d!|T| \int_0^1 dt_1(1-t_1)^{d-1} \int_0^1 dt_2(1-t_2)^{d-2} \cdots \int_0^1 dt_d f(x(t))$$

Gauss-Jacobi quadrature (exact for poly. of degree $2q-1$)

$$\int_0^1 (1-s)^\alpha s^\beta g(s)ds \approx \sum_{i=1}^{q} w_i^{(\alpha,\beta)} g(\xi_i^{(\alpha,\beta)})$$

Computation of Bernstein-Bézier Moments

Integral over simplex T reduced to box $[0,1]^d$

$$\int_T f(x)\mathrm{d}x = d!|T| \int_0^1 dt_1 (1-t_1)^{d-1} \int_0^1 dt_2 (1-t_2)^{d-2} \cdots \int_0^1 dt_d f(x(t))$$

Gauss-Jacobi quadrature (exact for poly. of degree $2q-1$)

$$\int_0^1 (1-s)^\alpha s^\beta g(s) ds \approx \sum_{i=1}^{q} w_i^{(\alpha,\beta)} g(\xi_i^{(\alpha,\beta)})$$

Stroud conical product rule:

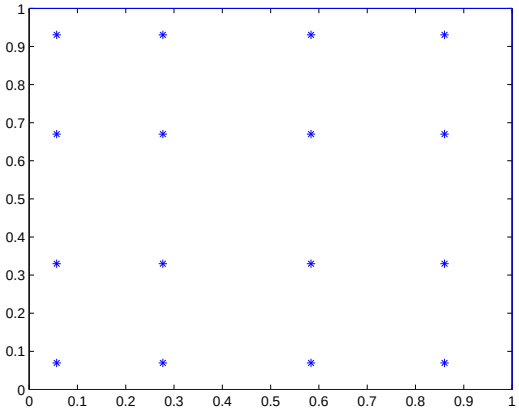
$$\int_T f(x)\mathrm{d}x \approx d!|T| \sum_{i_1=1}^{q} w_{i_1}^{(d-1,0)} \sum_{i_2=1}^{q} w_{i_2}^{(d-2,0)} \cdots \sum_{i_d=1}^{q} w_{i_d}^{(0,0)} f(x_{i_1,i_2,\ldots,i_d}),$$

$$\text{where } x_{i_1,i_2,\ldots,i_d} := x(\xi_{i_1}^{(d-1,0)}, \xi_{i_2}^{(d-2,0)}, \ldots, \xi_{i_d}^{(0,0)}).$$

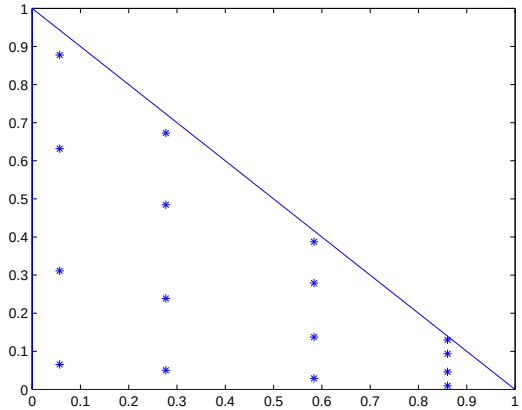
Stroud conical product rule

$$\int_T f(x)\mathrm{d}x \approx d!|T| \sum_{i_1=1}^{q} w_{i_1}^{(d-1,0)} \sum_{i_2=1}^{q} w_{i_2}^{(d-2,0)} \cdots \sum_{i_d=1}^{q} w_{i_d}^{(0,0)} f(x_{i_1,i_2,\ldots,i_d}),$$

$$\text{where } x_{i_1,i_2,\ldots,i_d} := x(\xi_{i_1}^{(d-1,0)}, \xi_{i_2}^{(d-2,0)}, \ldots, \xi_{i_d}^{(0,0)}).$$



$(\xi_{i_1}^{(1,0)}, \xi_{i_2}^{(0,0)})$



x_{i_1,i_2}

Computation of Bernstein-Bézier Moments

Stroud rule for the computation of the moments ($q \sim n$):

$$\rho_\eta^{n,d}(f) = \int_T f(x)\, B_\eta^{n,d}(x)\, \mathrm{d}x$$

$$\approx d!|T| \sum_{i_1,\ldots,i_d=1}^{q} w_{i_1}^{(d-1,0)} \cdots w_{i_d}^{(0,0)} B_\eta^{n,d}(x_{i_1,\ldots,i_d}) f(x_{i_1,\ldots,i_d})$$

Computation of Bernstein-Bézier Moments

Stroud rule for the computation of the moments ($q \sim n$):

$$\rho_\eta^{n,d}(f) = \int_T f(x)\, B_\eta^{n,d}(x)\, \mathrm{d}x$$

$$\approx d!|T| \sum_{i_1,\ldots,i_d=1}^{q} w_{i_1}^{(d-1,0)} \cdots w_{i_d}^{(0,0)} B_\eta^{n,d}(x_{i_1,\ldots,i_d}) f(x_{i_1,\ldots,i_d})$$

- Positive weights with respect to the data $f(x_{i_1,\ldots,i_d})$.

Computation of Bernstein-Bézier Moments

Stroud rule for the computation of the moments ($q \sim n$):

$$\rho_\eta^{n,d}(f) = \int_T f(x)\, B_\eta^{n,d}(x)\, \mathrm{d}x$$

$$\approx d!|T| \sum_{i_1,\ldots,i_d=1}^{q} w_{i_1}^{(d-1,0)} \cdots w_{i_d}^{(0,0)} B_\eta^{n,d}(x_{i_1,\ldots,i_d}) f(x_{i_1,\ldots,i_d})$$

- Positive weights with respect to the data $f(x_{i_1,\ldots,i_d})$.
- A naive way of computing these integrals could go through precomputing the weights $w_{i_1}^{(d-1,0)} \cdots w_{i_d}^{(0,0)} B_\eta^{n,d}(x_{i_1,\ldots,i_d})$ for all $\eta \in \mathcal{I}_{n,d}$ and all $(i_1, \ldots, i_d) \in \{1, \ldots, q\}^d$, which already results in excessive storage requirement $(nq)^d \sim n^{2d}$ (square of the size n^d of the moment vector), and then computing all integrals with $\sim n^{2d}$ operations.

Computation of Bernstein-Bézier Moments

Stroud rule for the computation of the moments ($q \sim n$):

$$\rho_\eta^{n,d}(f) = \int_T f(x)\, B_\eta^{n,d}(x)\, \mathrm{d}x$$

$$\approx d!|T| \sum_{i_1,\ldots,i_d=1}^{q} w_{i_1}^{(d-1,0)} \cdots w_{i_d}^{(0,0)} B_\eta^{n,d}(x_{i_1,\ldots,i_d}) f(x_{i_1,\ldots,i_d})$$

- Positive weights with respect to the data $f(x_{i_1,\ldots,i_d})$.
- A naive way of computing these integrals could go through precomputing the weights $w_{i_1}^{(d-1,0)} \cdots w_{i_d}^{(0,0)} B_\eta^{n,d}(x_{i_1,\ldots,i_d})$ for all $\eta \in \mathcal{I}_{n,d}$ and all $(i_1, \ldots, i_d) \in \{1, \ldots, q\}^d$, which already results in excessive storage requirement $(nq)^d \sim n^{2d}$ (square of the size n^d of the moment vector), and then computing all integrals with $\sim n^{2d}$ operations.
- We reduce this cost to $\sim n^{d+1}$ operations by using sum factorization thanks to another nice property of Bernstein polynomials: factorization w.r.t. Duffy transformation.

Computation of Bernstein-Bézier Moments

Factorization of Bernstein polynomials w.r.t. Duffy transform

$$b_1 = t_1, \quad b_2 = t_2(1-b_1), \quad b_3 = t_3(1-b_1-b_2), \quad \cdots,$$
$$b_d = t_d(1-b_1-\cdots-b_{d-1}), \quad b_{d+1} = (1-b_1-\cdots-b_d).$$

Hence

$$b_1 = t_1, \quad b_2 = (1-t_1)t_2, \quad b_3 = (1-t_1)(1-t_2)t_3, \quad \cdots,$$
$$b_d = (1-t_1)\cdots(1-t_{d-1})t_d, \quad b_{d+1} = (1-t_1)\cdots(1-t_d).$$

$$\begin{aligned} B_\eta^{n,d}(x(t)) &= \frac{n!}{\eta_1!\cdots\eta_{d+1}!} b_1^{\eta_1} b_2^{\eta_2} \cdots b_{d+1}^{\eta_{d+1}} \\ &= \frac{n!}{\eta_1!\cdots\eta_{d+1}!} t_1^{\eta_1} (1-t_1)^{\eta_2+\cdots+\eta_{d+1}} \cdots t_d^{\eta_d} (1-t_d)^{\eta_{d+1}} \\ &= B_{\eta_1}^{n}(t_1) B_{\eta_2}^{n-\eta_1}(t_2) \cdots B_{\eta_d}^{n-\eta_1-\cdots-\eta_{d-1}}(t_d), \end{aligned}$$

since $\sum_{k=1}^{d+1} \eta_k = n$.

Computation of Bernstein-Bézier Moments

Computing the moments by sum factorisation:

$$\rho_{\eta}^{n,d}(f) = \int_T f(x)\, B_{\eta}^{n,d}(x)\, \mathrm{d}x \approx d!\,|T| \sum_{i_d=1}^{q} w_{i_d}^{(0,0)}\, B_{\eta_d}^{n-\eta_1-\cdots-\eta_{d-1}}(\xi_{i_d}^{(0,0)}) \cdots$$
$$\sum_{i_2=1}^{q} w_{i_2}^{(d-2,0)}\, B_{\eta_2}^{n-\eta_1}(\xi_{i_2}^{(d-2,0)}) \sum_{i_1=1}^{q} w_{i_1}^{(d-1,0)}\, B_{\eta_1}^{n}(\xi_{i_1}^{(d-1,0)})\, f(x_{i_1,i_2,\ldots,i_d})$$

Computation of Bernstein-Bézier Moments

Computing the moments by sum factorisation:

$$\rho_\eta^{n,d}(f) = \int_T f(x)\, B_\eta^{n,d}(x)\, \mathrm{d}x \approx d!\,|T| \sum_{i_d=1}^{q} w_{i_d}^{(0,0)}\, B_{\eta_d}^{n-\eta_1-\cdots-\eta_{d-1}}(\xi_{i_d}^{(0,0)}) \cdots$$
$$\sum_{i_2=1}^{q} w_{i_2}^{(d-2,0)}\, B_{\eta_2}^{n-\eta_1}(\xi_{i_2}^{(d-2,0)}) \sum_{i_1=1}^{q} w_{i_1}^{(d-1,0)}\, B_{\eta_1}^{n}(\xi_{i_1}^{(d-1,0)})\, f(x_{i_1,i_2,\ldots,i_d})$$

- Precompute $B_j^m(\xi_i^{(k,0)})$ for all $m = 0, \ldots, n$, $j = 0, \ldots, m$, $i = 1, \ldots, q$, $k = 0, \ldots, d-1$, altogether $\mathcal{O}(dqn^2)$ values, and dq values of $w_i^{(k,0)}$, $i = 1, \ldots, q$, $k = 0, \ldots, d-1$.

Computation of Bernstein-Bézier Moments

Computing the moments by sum factorisation:

$$\rho_\eta^{n,d}(f) = \int_T f(x)\, B_\eta^{n,d}(x)\, \mathrm{d}x \approx d!\,|T| \sum_{i_d=1}^{q} w_{i_d}^{(0,0)}\, B_{\eta_d}^{n-\eta_1-\cdots-\eta_{d-1}}(\xi_{i_d}^{(0,0)}) \cdots$$
$$\sum_{i_2=1}^{q} w_{i_2}^{(d-2,0)}\, B_{\eta_2}^{n-\eta_1}(\xi_{i_2}^{(d-2,0)}) \sum_{i_1=1}^{q} w_{i_1}^{(d-1,0)}\, B_{\eta_1}^{n}(\xi_{i_1}^{(d-1,0)})\, f(x_{i_1,i_2,\ldots,i_d})$$

- Precompute $B_j^m(\xi_i^{(k,0)})$ for all $m = 0,\ldots,n$, $j = 0,\ldots,m$, $i = 1,\ldots,q$, $k = 0,\ldots,d-1$, altogether $\mathcal{O}(dqn^2)$ values, and dq values of $w_i^{(k,0)}$, $i = 1,\ldots,q$, $k = 0,\ldots,d-1$.
- Assume $\mathcal{O}(q^d)$ cost for q^d evaluations $f(x_{i_1,i_2,\ldots,i_d})$.

Computation of Bernstein-Bézier Moments

Computing the moments by sum factorisation:

$$\rho_\eta^{n,d}(f) = \int_T f(x)\, B_\eta^{n,d}(x)\, \mathrm{d}x \approx d!\,|T| \sum_{i_d=1}^{q} w_{i_d}^{(0,0)} B_{\eta_d}^{n-\eta_1-\cdots-\eta_{d-1}}(\xi_{i_d}^{(0,0)}) \cdots$$
$$\sum_{i_2=1}^{q} w_{i_2}^{(d-2,0)} B_{\eta_2}^{n-\eta_1}(\xi_{i_2}^{(d-2,0)}) \sum_{i_1=1}^{q} w_{i_1}^{(d-1,0)} B_{\eta_1}^{n}(\xi_{i_1}^{(d-1,0)}) f(x_{i_1,i_2,\ldots,i_d})$$

- Precompute $B_j^m(\xi_i^{(k,0)})$ for all $m = 0,\ldots,n$, $j = 0,\ldots,m$, $i = 1,\ldots,q$, $k = 0,\ldots,d-1$, altogether $\mathcal{O}(dqn^2)$ values, and dq values of $w_i^{(k,0)}$, $i = 1,\ldots,q$, $k = 0,\ldots,d-1$.
- Assume $\mathcal{O}(q^d)$ cost for q^d evaluations $f(x_{i_1,i_2,\ldots,i_d})$.
- k-th loop: sum over i_k with $\mathcal{O}(q)$ flops for each $(\eta_1,\ldots,\eta_k)$, $\eta_1+\cdots+\eta_k \le n$, and each $(i_{k+1}, i_2, \ldots, i_d)$, with $\mathcal{O}(qn^k q^{d-k})$ cost, for all $k = 1,\ldots,d$.

Computation of Bernstein-Bézier Moments

Computing the moments by sum factorisation:

$$\rho_\eta^{n,d}(f) = \int_T f(x)\, B_\eta^{n,d}(x)\, \mathrm{d}x \approx d!\,|T| \sum_{i_d=1}^{q} w_{i_d}^{(0,0)} B_{\eta_d}^{n-\eta_1-\cdots-\eta_{d-1}}(\xi_{i_d}^{(0,0)}) \cdots$$
$$\sum_{i_2=1}^{q} w_{i_2}^{(d-2,0)} B_{\eta_2}^{n-\eta_1}(\xi_{i_2}^{(d-2,0)}) \sum_{i_1=1}^{q} w_{i_1}^{(d-1,0)} B_{\eta_1}^{n}(\xi_{i_1}^{(d-1,0)}) f(x_{i_1,i_2,\ldots,i_d})$$

- Precompute $B_j^m(\xi_i^{(k,0)})$ for all $m = 0,\ldots,n$, $j = 0,\ldots,m$, $i = 1,\ldots,q$, $k = 0,\ldots,d-1$, altogether $\mathcal{O}(dqn^2)$ values, and dq values of $w_i^{(k,0)}$, $i = 1,\ldots,q$, $k = 0,\ldots,d-1$.
- Assume $\mathcal{O}(q^d)$ cost for q^d evaluations $f(x_{i_1,i_2,\ldots,i_d})$.
- k-th loop: sum over i_k with $\mathcal{O}(q)$ flops for each $(\eta_1,\ldots,\eta_k)$, $\eta_1+\cdots+\eta_k \le n$, and each $(i_{k+1}, i_2, \ldots, i_d)$, with $\mathcal{O}(qn^k q^{d-k})$ cost, for all $k = 1,\ldots,d$.
- Hence, overall complexity $\sim n^{d+1}$ for any $d \ge 2$.

Computation of Bernstein-Bézier Moments

Fast evaluation of a polynomial $u(x) := \sum_{\eta \in \mathcal{I}_{n,d}} u_\eta B_\eta^{n,d}(x)$

at all Stroud points:

$$u(x_{i_1,i_2,\ldots,i_d}) = \sum_{\eta_1=0}^{n} B_{\eta_1}^{n}(\xi_{i_1}^{(d-1,0)}) \sum_{\eta_2=1}^{n-\eta_1} B_{\eta_2}^{n-\eta_1}(\xi_{i_2}^{(d-2,0)}) \cdots \sum_{\eta_d=1}^{n-\eta_1-\cdots-\eta_{d-1}} B_{\eta_d}^{n-\eta_1-\cdots-\eta_{d-1}}(\xi_{i_d}^{(0,0)}) u_\eta$$

Computation of Bernstein-Bézier Moments

Fast evaluation of a polynomial $u(x) := \sum_{\eta \in \mathcal{I}_{n,d}} u_\eta B_\eta^{n,d}(x)$ at all Stroud points:

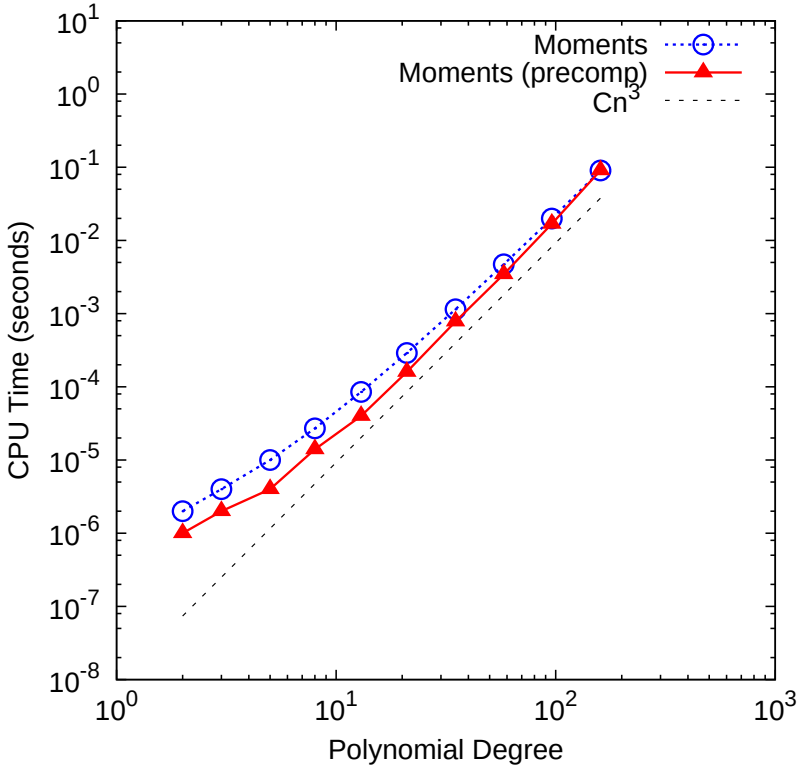
$$u(x_{i_1,i_2,\ldots,i_d}) = \sum_{\eta_1=0}^{n} B_{\eta_1}^{n}(\xi_{i_1}^{(d-1,0)}) \sum_{\eta_2=1}^{n-\eta_1} B_{\eta_2}^{n-\eta_1}(\xi_{i_2}^{(d-2,0)}) \cdots \sum_{\eta_d=1}^{n-\eta_1-\cdots-\eta_{d-1}} B_{\eta_d}^{n-\eta_1-\cdots-\eta_{d-1}}(\xi_{i_d}^{(0,0)}) u_\eta$$

- Sum factorization: start from the i_d-loop.
- k-th loop: sum over η_k with $\mathcal{O}(1)$ flops for each $(\eta_1, \ldots, \eta_k)$, $\eta_1 + \cdots + \eta_k \leq n$, and each $(i_k, i_2, \ldots, i_d)$, with $\mathcal{O}(n^k q^{d-k+1})$ cost, for $k = d, \ldots, 1$.
- Overall complexity $\sim n^{d+1}$ (the same order as a single evaluation by de Casteljau algorithm).

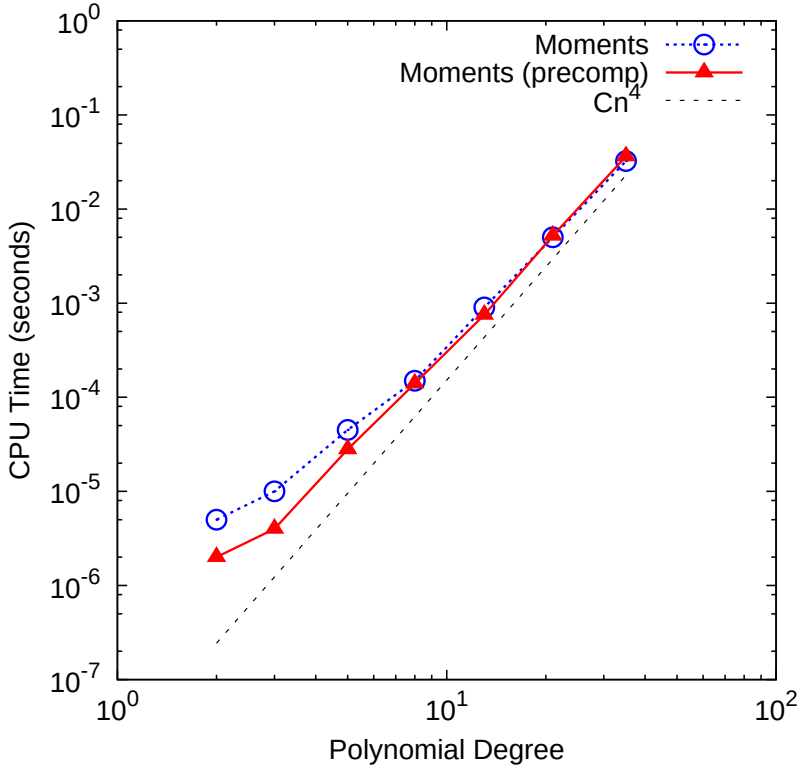
Computation of Bernstein-Bézier Moments

CPU time: computation of BB moments with $\mathcal{O}(n^{d+1})$ cost

2D



3D



C++ code available (GPL): **`www.bbfem.de`**

Outline

1. Bernstein Polynomials
2. FEM Assembly
3. Computation of Bernstein-Bézier Moments
4. Assembly of Element Matrices
5. Further Applications of BB Moments

Assembly of Element Matrices

Dirichlet problem for linear second order elliptic equation:

$$\begin{cases} -\operatorname{div}(A\nabla u) + \mathbf{b}\cdot\nabla u + cu &= f \quad \text{in } \Omega, \\ u &= g \quad \text{on } \partial\Omega. \end{cases}$$

Weak formulation of the homogenuous problem ($g = 0$):
Find $u \in H_0^1(\Omega)$ such that for all $v \in H_0^1(\Omega)$

$$\int_\Omega \Big[\nabla u \cdot A\nabla v + (\mathbf{b}\cdot\nabla u)v + cuv\Big]\mathrm{d}x = \int_\Omega fv\mathrm{d}x.$$

In Galerkin FEM with Bernstein basis shape functions on simplices, we need to compute for all $\xi, \zeta \in \mathcal{I}_{n,d}$

$$\int_T \Big[\nabla B_\xi^{n,d}\cdot A\nabla B_\zeta^{n,d} + (\mathbf{b}\cdot\nabla B_\xi^{n,d})B_\zeta^{n,d} + cB_\xi^{n,d}B_\zeta^{n,d}\Big]\mathrm{d}x, \quad \int_T f\,B_\zeta^{n,d}\mathrm{d}x.$$

Assembly of Element Matrices

Using Bernstein-Bézier moments

$$\rho_\xi^{n,d}(c) = \int_T c(x)\, B_\xi^{n,d}(x)\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d},$$

we have:

- Load vector $\int_T f(x) B_\xi^{n,d}(x) \mathrm{d}x = \rho_\xi^{n,d}(f), \quad \xi \in \mathcal{I}_{n,d},$

  This is just a moment vector, cost: $\mathcal{O}(n^{d+1})$ flops.

Assembly of Element Matrices

Using Bernstein-Bézier moments

$$\rho_\xi^{n,d}(c) = \int_T c(x)\, B_\xi^{n,d}(x)\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d},$$

we have:

- Load vector $\int_T f(x) B_\xi^{n,d}(x) \mathrm{d}x = \rho_\xi^{n,d}(f), \quad \xi \in \mathcal{I}_{n,d},$

  This is just a moment vector, cost: $\mathcal{O}(n^{d+1})$ flops.
- Mass matrix (thanks to the product formula)

$$\int_T c(x) B_\xi^{n,d}(x) B_\zeta^{n,d}(x) \mathrm{d}x = \frac{\binom{n}{\xi}\binom{n}{\zeta}}{\binom{2n}{\xi+\zeta}} \rho_{\xi+\zeta}^{2n,d}(c), \quad \xi, \zeta \in \mathcal{I}_{n,d}$$

Assembly of Element Matrices

Using Bernstein-Bézier moments

$$\rho_\xi^{n,d}(c) = \int_T c(x)\, B_\xi^{n,d}(x)\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d},$$

we have:

- Load vector $\int_T f(x) B_\xi^{n,d}(x) \mathrm{d}x = \rho_\xi^{n,d}(f), \quad \xi \in \mathcal{I}_{n,d},$

  This is just a moment vector, cost: $\mathcal{O}(n^{d+1})$ flops.
- Mass matrix (thanks to the product formula)

$$\int_T c(x) B_\xi^{n,d}(x) B_\zeta^{n,d}(x) \mathrm{d}x = \frac{\binom{n}{\xi}\binom{n}{\zeta}}{\binom{2n}{\xi+\zeta}} \rho_{\xi+\zeta}^{2n,d}(c), \quad \xi, \zeta \in \mathcal{I}_{n,d}$$

- Stiffness matrix (thanks to the gradient formula)

$$\int_T \nabla B_\xi^{n,d}(x) A(x) \nabla B_\zeta^{n,d}(x) \mathrm{d}x = n^2 \sum_{\alpha,\beta \in \mathcal{I}_{1,d}} \frac{\binom{n-1}{\xi-\alpha}\binom{n-1}{\zeta-\beta}}{\binom{2n-2}{\xi-\alpha+\zeta-\beta}} \rho_{\xi-\alpha+\zeta-\beta}^{2n-2,d}(\nabla b_\alpha^T A \nabla b_\beta)$$

Assembly of Element Matrices

Computation of the system matrices with $\mathcal{O}(1)$ cost per entry:

$$n^2 \sum_{\alpha,\beta \in \mathcal{I}_{1,d}} \frac{\binom{n-1}{\xi-\alpha}\binom{n-1}{\zeta-\beta}}{\binom{2n-2}{\xi-\alpha+\zeta-\beta}} \rho^{2n-2,d}_{\xi-\alpha+\zeta-\beta}(\nabla b_\alpha^T A \nabla b_\beta), \quad \xi, \zeta \in \mathcal{I}_{n,d}$$

Assembly of Element Matrices

Computation of the system matrices with $\mathcal{O}(1)$ cost per entry:

$$n^2 \sum_{\alpha,\beta \in \mathcal{I}_{1,d}} \frac{\binom{n-1}{\xi-\alpha}\binom{n-1}{\zeta-\beta}}{\binom{2n-2}{\xi-\alpha+\zeta-\beta}} \rho^{2n-2,d}_{\xi-\alpha+\zeta-\beta}(\nabla b_\alpha^T A \nabla b_\beta), \quad \xi, \zeta \in \mathcal{I}_{n,d}$$

- Number of entries is $|\mathcal{I}_{n,d}|^2 \sim n^{2d}$

Assembly of Element Matrices

Computation of the system matrices with $\mathcal{O}(1)$ cost per entry:

$$n^2 \sum_{\alpha,\beta \in \mathcal{I}_{1,d}} \frac{\binom{n-1}{\xi-\alpha}\binom{n-1}{\zeta-\beta}}{\binom{2n-2}{\xi-\alpha+\zeta-\beta}} \rho^{2n-2,d}_{\xi-\alpha+\zeta-\beta}(\nabla b_\alpha^T A \nabla b_\beta), \quad \xi, \zeta \in \mathcal{I}_{n,d}$$

- Number of entries is $|\mathcal{I}_{n,d}|^2 \sim n^{2d}$
- Multinomial coefficients may be computed recursively by a few divisions and multiplications per entry contributing $\mathcal{O}(n^{2d})$ flops.

Assembly of Element Matrices

Computation of the system matrices with $\mathcal{O}(1)$ cost per entry:

$$n^2 \sum_{\alpha,\beta \in \mathcal{I}_{1,d}} \frac{\binom{n-1}{\xi-\alpha}\binom{n-1}{\zeta-\beta}}{\binom{2n-2}{\xi-\alpha+\zeta-\beta}} \rho^{2n-2,d}_{\xi-\alpha+\zeta-\beta}(\nabla b_\alpha^T A \nabla b_\beta), \quad \xi, \zeta \in \mathcal{I}_{n,d}$$

- Number of entries is $|\mathcal{I}_{n,d}|^2 \sim n^{2d}$
- Multinomial coefficients may be computed recursively by a few divisions and multiplications per entry contributing $\mathcal{O}(n^{2d})$ flops.
- Need the moments of degree $2n-2$,

$$\rho^{2n-2,d}_\xi(a) = \int_T a(x)\, B^{2n-2,d}_\xi(x) \mathrm{d}x, \quad \xi \in \mathcal{I}_{2n-2,d}$$

Assembly of Element Matrices

Computation of the system matrices with $\mathcal{O}(1)$ cost per entry:

$$n^2 \sum_{\alpha,\beta \in \mathcal{I}_{1,d}} \frac{\binom{n-1}{\xi-\alpha}\binom{n-1}{\zeta-\beta}}{\binom{2n-2}{\xi-\alpha+\zeta-\beta}} \rho^{2n-2,d}_{\xi-\alpha+\zeta-\beta}(\nabla b_\alpha^T A \nabla b_\beta), \quad \xi, \zeta \in \mathcal{I}_{n,d}$$

- Number of entries is $|\mathcal{I}_{n,d}|^2 \sim n^{2d}$
- Multinomial coefficients may be computed recursively by a few divisions and multiplications per entry contributing $\mathcal{O}(n^{2d})$ flops.
- Need the moments of degree $2n-2$,

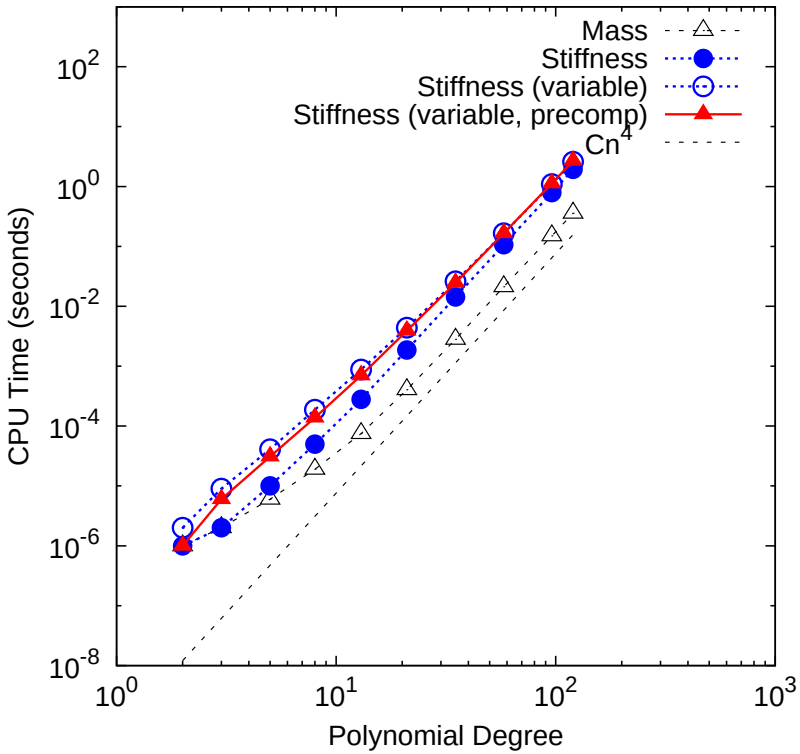
$$\rho^{2n-2,d}_\xi(a) = \int_T a(x)\, B^{2n-2,d}_\xi(x) \mathrm{d}x, \quad \xi \in \mathcal{I}_{2n-2,d}$$

- As we know they can be computed with $\mathcal{O}(n^{d+1}) \ll n^{2d}$ flops! Hence overall cost is optimal: $\mathcal{O}(n^{2d})$

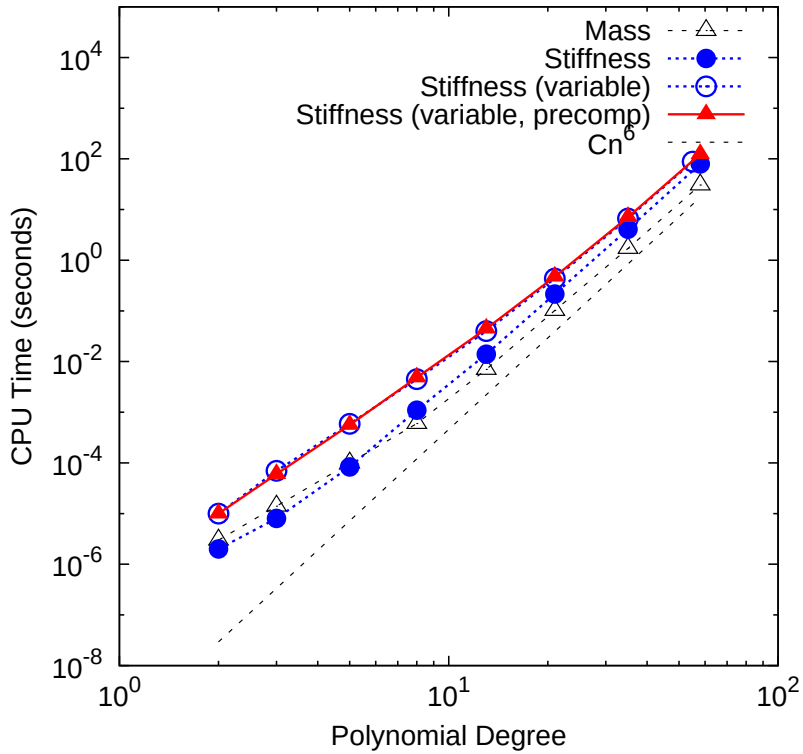
Assembly of Element Matrices

CPU time: assembly of mass and stiffness matrices

2D



3D



C++ code available (GPL): **`www.bbfem.de`**

Assembly of Element Matrices

History: Optimal assembly of element system matrices

- Lagrange basis on simplices: $\mathcal{O}(n^d)$ per entry, total $\mathcal{O}(n^{3d})$

Assembly of Element Matrices

History: Optimal assembly of element system matrices

- Lagrange basis on simplices: $\mathcal{O}(n^d)$ per entry, total $\mathcal{O}(n^{3d})$
- [Orszag, 1980]: The sum factorisation method with the optimal $\mathcal{O}(n^{2d})$ cost for tensor-product elements.

Assembly of Element Matrices

History: Optimal assembly of element system matrices

- Lagrange basis on simplices: $\mathcal{O}(n^d)$ per entry, total $\mathcal{O}(n^{3d})$
- [Orszag, 1980]: The sum factorisation method with the optimal $\mathcal{O}(n^{2d})$ cost for tensor-product elements.
- [Karniadikis & Sherwin, 1999] designed basis functions on simplices with almost optimal $\mathcal{O}(n^{2d+1})$ complexity.

Assembly of Element Matrices

History: Optimal assembly of element system matrices

- Lagrange basis on simplices: $\mathcal{O}(n^d)$ per entry, total $\mathcal{O}(n^{3d})$
- [Orszag, 1980]: The sum factorisation method with the optimal $\mathcal{O}(n^{2d})$ cost for tensor-product elements.
- [Karniadikis & Sherwin, 1999] designed basis functions on simplices with almost optimal $\mathcal{O}(n^{2d+1})$ complexity.
- [Eibner & Melenk, 2006] achieved optimal complexity $\mathcal{O}(n^{2d})$ on simplices in 2D and 3D by a extending the polynomial space by many "internal" shape functions of higher degree (e.g., $6\dim(\mathbb{P}_n)$ shape functions in 3D) and adapting Lagrange basis polynomials to the nodes of the quadrature rule used for integration.

Assembly of Element Matrices

History: Optimal assembly of element system matrices

- Lagrange basis on simplices: $\mathcal{O}(n^d)$ per entry, total $\mathcal{O}(n^{3d})$
- [Orszag, 1980]: The sum factorisation method with the optimal $\mathcal{O}(n^{2d})$ cost for tensor-product elements.
- [Karniadikis & Sherwin, 1999] designed basis functions on simplices with almost optimal $\mathcal{O}(n^{2d+1})$ complexity.
- [Eibner & Melenk, 2006] achieved optimal complexity $\mathcal{O}(n^{2d})$ on simplices in 2D and 3D by a extending the polynomial space by many "internal" shape functions of higher degree (e.g., $6\dim(\mathbb{P}_n)$ shape functions in 3D) and adapting Lagrange basis polynomials to the nodes of the quadrature rule used for integration.
- [Ainsworth, Andriamaro & D., 2011]: Optimal complexity $\mathcal{O}(n^{2d})$ with Bernstein-Bézier shape functions.

Outline

1. Bernstein Polynomials
2. FEM Assembly
3. Computation of Bernstein-Bézier Moments
4. Assembly of Element Matrices
5. Further Applications of BB Moments

Matrix-vector products for iterative methods

- For example, let $M = [M_{\xi\zeta}]_{\xi,\zeta\in\mathcal{I}_{n,d}}$ be the element mass matrix,

$$M_{\xi\zeta} = \int_T c(x) B_\xi^{n,d}(x) B_\zeta^{n,d}(x) \mathrm{d}x.$$

Matrix-vector products for iterative methods

- For example, let $M = [M_{\xi\zeta}]_{\xi,\zeta\in\mathcal{I}_{n,d}}$ be the element mass matrix,

$$M_{\xi\zeta} = \int_T c(x) B_\xi^{n,d}(x) B_\zeta^{n,d}(x) \mathrm{d}x.$$

- Then, with $u(x) := \sum_{\zeta\in\mathcal{I}_{n,d}} u_\zeta B_\zeta^{n,d}(x)$,

$$M \cdot [u_\zeta]_{\zeta\in\mathcal{I}_{n,d}} = \Big[\int_T c(x) u(x) B_\xi^{n,d}(x) \mathrm{d}x \Big]_{\xi\in\mathcal{I}_{n,d}} = [\rho_\xi^{n,d}(cu)]_{\xi\in\mathcal{I}_{n,d}}.$$

Further Applications of BB Moments

Matrix-vector products for iterative methods

- For example, let $M = [M_{\xi\zeta}]_{\xi,\zeta\in\mathcal{I}_{n,d}}$ be the element mass matrix,

$$M_{\xi\zeta} = \int_T c(x) B_\xi^{n,d}(x) B_\zeta^{n,d}(x) \mathrm{d}x.$$

- Then, with $u(x) := \sum_{\zeta\in\mathcal{I}_{n,d}} u_\zeta B_\zeta^{n,d}(x)$,

$$M \cdot [u_\zeta]_{\zeta\in\mathcal{I}_{n,d}} = \Big[\int_T c(x) u(x) B_\xi^{n,d}(x) \mathrm{d}x \Big]_{\xi\in\mathcal{I}_{n,d}} = [\rho_\xi^{n,d}(cu)]_{\xi\in\mathcal{I}_{n,d}}.$$

- Evaluation cost for $u(x_{i_1,i_2,\ldots,i_d})$ at Stroud points is $\mathcal{O}(n^{d+1})$.

Matrix-vector products for iterative methods

- For example, let $M = [M_{\xi\zeta}]_{\xi,\zeta\in\mathcal{I}_{n,d}}$ be the element mass matrix,

$$M_{\xi\zeta} = \int_T c(x) B_\xi^{n,d}(x) B_\zeta^{n,d}(x) \mathrm{d}x.$$

- Then, with $u(x) := \sum_{\zeta\in\mathcal{I}_{n,d}} u_\zeta B_\zeta^{n,d}(x)$,

$$M \cdot [u_\zeta]_{\zeta\in\mathcal{I}_{n,d}} = \Big[\int_T c(x) u(x) B_\xi^{n,d}(x) \mathrm{d}x \Big]_{\xi\in\mathcal{I}_{n,d}} = [\rho_\xi^{n,d}(cu)]_{\xi\in\mathcal{I}_{n,d}}.$$

- Evaluation cost for $u(x_{i_1,i_2,\ldots,i_d})$ at Stroud points is $\mathcal{O}(n^{d+1})$.
- The same for the moments $\rho_\xi^{n,d}(cu)$, hence the total cost is $\mathcal{O}(n^{d+1})$, much less than n^{2d} needed for matrix assembly.

Further Applications of BB Moments

Matrix-vector products for iterative methods

- For example, let $M = [M_{\xi\zeta}]_{\xi,\zeta\in\mathcal{I}_{n,d}}$ be the element mass matrix,

$$M_{\xi\zeta} = \int_T c(x) B_\xi^{n,d}(x) B_\zeta^{n,d}(x) \mathrm{d}x.$$

- Then, with $u(x) := \sum_{\zeta\in\mathcal{I}_{n,d}} u_\zeta B_\zeta^{n,d}(x)$,

$$M \cdot [u_\zeta]_{\zeta\in\mathcal{I}_{n,d}} = \Big[\int_T c(x) u(x) B_\xi^{n,d}(x) \mathrm{d}x \Big]_{\xi\in\mathcal{I}_{n,d}} = [\rho_\xi^{n,d}(cu)]_{\xi\in\mathcal{I}_{n,d}}.$$

- Evaluation cost for $u(x_{i_1,i_2,\ldots,i_d})$ at Stroud points is $\mathcal{O}(n^{d+1})$.
- The same for the moments $\rho_\xi^{n,d}(cu)$, hence the total cost is $\mathcal{O}(n^{d+1})$, much less than n^{2d} needed for matrix assembly.
- [Kirby 2011, Kirby & Thinh 2012]: A different $\mathcal{O}(n^{d+1})$ algorithm for $M \cdot [u_\zeta]_{\zeta\in\mathcal{I}_{n,d}}$ implicitly relying on the same properties of Bernstein polynomials

Nonlinear problems

- Load vectors and linearized system matrices are computed as usual, with the exception that the coefficients depend on the current iterate u of the approximate solution.
- Reduces to the computation of the BB moments of the type

$$\rho_{\xi}^{n,d}(u, f) = \int_T f(x; u(x); \nabla u(x))\, B_{\xi}^{n,d}\, \mathrm{d}x, \quad \xi \in \mathcal{I}_{n,d}$$

- With u expressed in BB form, the cost of its evaluation and evaluation of its gradient at Stroud points is $\mathcal{O}(n^{d+1})$. Then f is evaluated with $\mathcal{O}(n^d)$ cost, and the usual algorithm for the moments requires again $\mathcal{O}(n^{d+1})$ flops, and we arrive at the total $\mathcal{O}(n^{d+1})$ cost for the moment vector.

Further Applications of BB Moments

Isogeometric elements

- Quadrilateral and hexahedral elements, isoparametric and other curved elements. Involve a nonlinear mapping from parameter to physical domain $\Phi : T \to \widetilde{T}$ with Jacobi matrix J_Φ.
- Stiffness matrix can be computed with the help of a BB moment vector at optimal $\mathcal{O}(n^{2d})$ cost:

$$\int_T \nabla B_\xi^{n,d}(x) J_\Phi^{-1}(x) \cdot A \nabla B_\zeta^{n,d}(x) J_\Phi^{-1}(x) |\det J_\Phi(x)| \, \mathrm{d}x$$

- Sum factorization is also useful for the assembly of high order B-spline system matrices (isogeometric analysis). Almost optimal algorithm: $\mathcal{O}(n)$ flops per non-zero entry[2]

[2] F. Calabrò, G. Sangalli and M. Tani, Fast formation of isogeometric Galerkin matrices by weighted quadrature, Comput. Methods Appl. Mech. Engrg., 316 (2017), 606–622.

Vector-valued finite elements

- Needed e.g. to solve Maxwell's equations (time harmonic):

$$\begin{cases} \operatorname{curl} \mathbf{E} + i\omega\mu\mathbf{H} &= \mathbf{0}, \\ \operatorname{curl} \mathbf{H} - i\omega\epsilon\mathbf{E} &= \mathbf{J}. \end{cases}$$

  $\mathbf{E}, \mathbf{H} \in H(\operatorname{curl}) := \{\mathbf{f} \in [L^2(\Omega)]^3 : \operatorname{curl}(\mathbf{f}) \in [L^2(\Omega)]^3\}$

- Nédélec's space for a triangulation $\triangle$ in 2D

$$\mathcal{N}_n(\triangle) = \Big\{\mathbf{s} \in [S_{n+1}(\triangle)]^2 : s \text{ is tangent continuous,}$$
$$\mathbf{s}|_T \in \mathbb{N}_n := [\mathbb{P}_n]^2 + \mathbf{x}^\perp \mathbb{H}_n \text{ for all } T \in \triangle\Big\}$$

  $\mathbf{x}^\perp = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^\perp := \begin{bmatrix} -x_2 \\ x_1 \end{bmatrix}$, $\mathbb{P}_n$: polynomials of total degree n,
  $\mathbb{H}_n$: homogeneous polynomials of degree n,
  $S_{n+1}(\triangle)$: (discont.) piecewise polynomials of degree $n+1$

- Shape functions are basis functions for $\mathbb{N}_n$ on triangles $T \in \triangle$. Combined together into basis functions for $\mathcal{N}_n(\triangle)$

Bernstein-Bézier shape functions for $H(\mathrm{curl})$ in 2D

- Shape functions for $\mathbb{N}_n$ are given by[3]

$$E_n^{(i)} = \{\nabla B_{\boldsymbol{\alpha}}^{n+1} : \boldsymbol{\alpha} \in \mathring{\gamma}_i^{n+1}\} \cup \{\boldsymbol{\omega}_i\}, \quad i = 1,2,3,$$
$$I_n^{\nabla} = \{\nabla B_{\boldsymbol{\alpha}}^{n+1} : \boldsymbol{\alpha} \in \mathring{\mathcal{I}}_{n+1,2}\},$$
$$I_n^{\sigma} = \left\{(n+1) B_{\boldsymbol{\alpha}}^{n} \sum_{i=1}^{3} \alpha_i \boldsymbol{\omega}_i : \boldsymbol{\alpha} \in \mathcal{I}_{n,2} \setminus \{\tilde{\boldsymbol{\alpha}}\}\right\},$$

  $\boldsymbol{\omega}_i := b_{i+1}\nabla b_{i-1} - b_{i-1}\nabla b_{i+1}$ are Whitney functions, $\mathring{\gamma}_i^{n+1}$, $\mathring{\gamma}_i^{n+1}$, $\mathring{\mathcal{I}}_{n+1,2}$, $\tilde{\boldsymbol{\alpha}}$: certain sets of indices

- High order gradient shape functions are separated
- Different shape functions based on barycentric coordinates have been suggested by [Arnold, Falk & Winther, 2009].

[3] M. Ainsworth and G. Andriamaro and O. Davydov, A Bernstein-Bézier basis for arbitrary order Raviart-Thomas finite elements, Constr. Approx., 41 (2015), 1-22.

Numerical example: Standing wave solutions of Maxwell's equations in 2D

- With $\Omega = [0,1]^2$, we want to solve the eigenvalue problem ($\lambda > 0$):

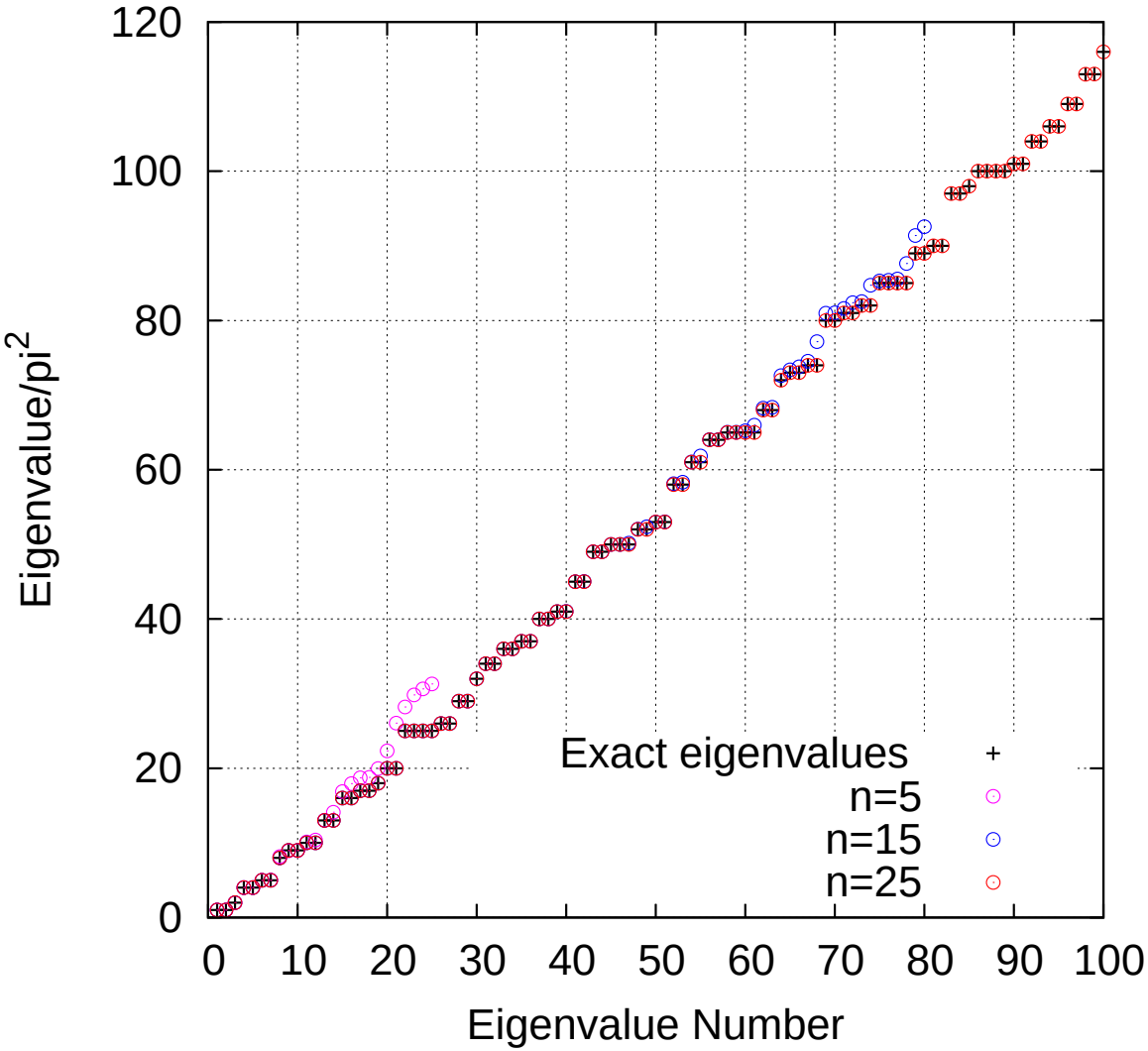
$$\begin{aligned} -\operatorname{curl}(\operatorname{curl}\mathbf{u}) &= \lambda^2 \mathbf{u} \quad \text{in } \Omega, \\ \mathbf{t}\cdot\mathbf{u} &= 0 \quad \text{on } \partial\Omega. \end{aligned}$$

- The eigenpairs are given by $(\lambda_{k,\ell}^2, u_{k,\ell})$, $k, \ell = 0, 1, \ldots$, with

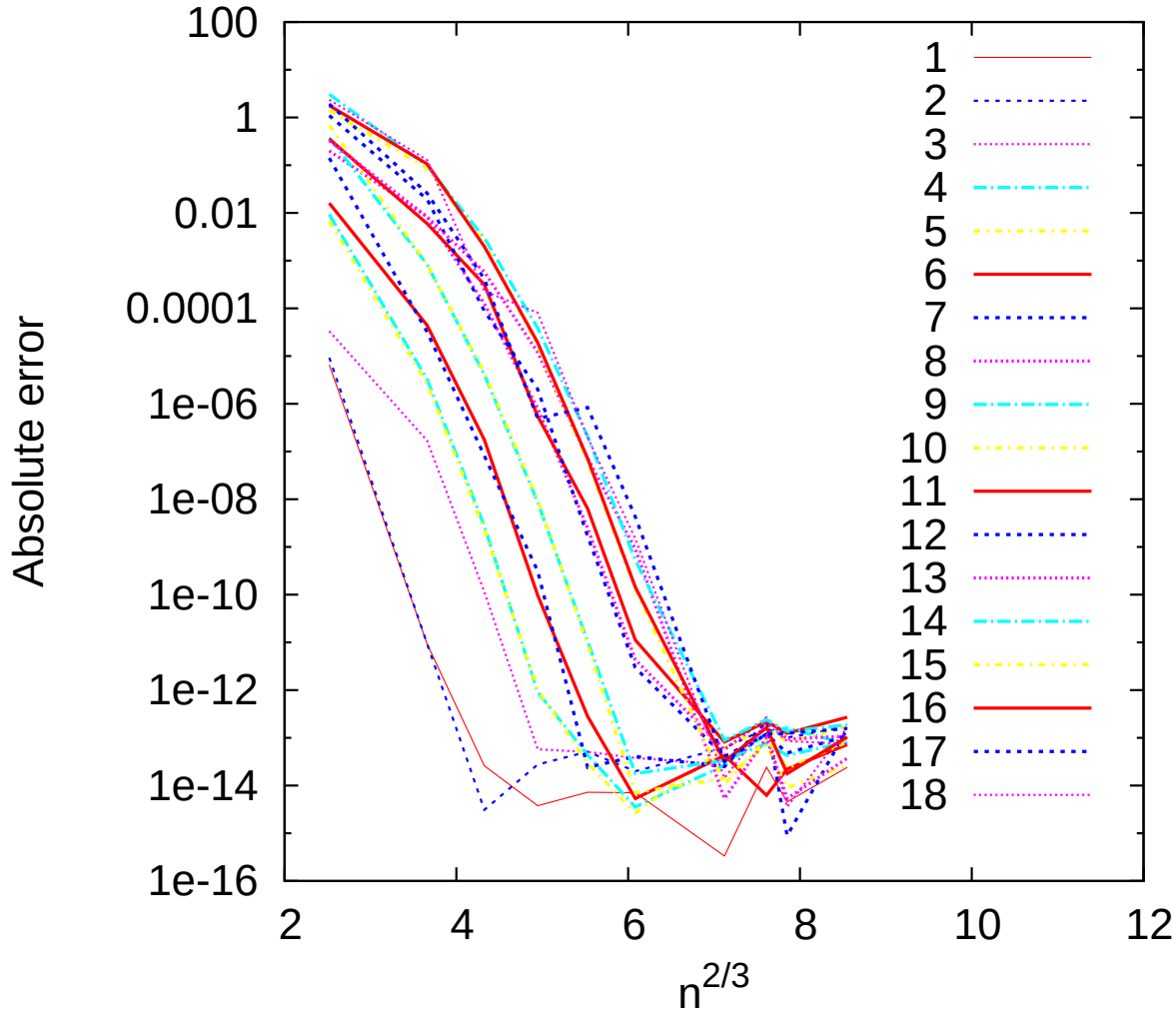
$$\lambda_{k,\ell}^2 := \pi^2(k^2+\ell^2), \quad \mathbf{u}_{k,\ell}(x,y) := \begin{pmatrix} k\sin(k\pi x)\cos(\ell\pi x) \\ \ell\sin(\ell\pi y)\cos(k\pi x) \end{pmatrix}^{\perp}.$$

- Divide the square into two triangles by a diagonal, and use BB shape functions of degree n up to 25.

Further Applications of BB Moments



Further Applications of BB Moments



Convergence with rate $q^{n^{2/3}}$ for some $0 < q < 1$, in agreement with theoretical error bounds [Ainsworth & Coyle, 2003].

Further Applications of BB Moments

Condition numbers of simple eigenvalues

$\lambda_{1,1}$	$\lambda_{2,2}$	$\lambda_{3,3}$	$\lambda_{4,4}$	$\lambda_{6,6}$	$\lambda_{7,7}$
8.97e+00	3.69e+00	3.38e+01	8.00e+02	1.16e+06	5.25e+07

- Shown: the largest condition numbers of the simple eigenvalues that occured in our experiments with $n \leq 25$.
- The condition number ν of a simple eigenvalue λ of the generalized eigenvalue problem $\det(A - \lambda B) = 0$ is defined as

$$\nu = \frac{\|x\|_2 \|y\|_2}{\sqrt{|\alpha|^2 + |\beta|^2}}, \qquad \text{with} \quad \alpha = y^H A x, \quad \beta = y^H B x,$$

  where x and y are the right and left eigenvectors of λ [Stewart, Matrix Algorithms Vol.II, 2001].
- The numbers in the table suggest reasonable conditioning even for the 85th eigenvalue $\lambda_{7,7}$.

Further Applications of BB Moments

Some further papers on BBFEM

- **Mixed tetrahedral-hexahedral-pyramidal partitions:** M. Ainsworth, O. Davydov and L. L. Schumaker, Bernstein-Bézier finite elements on tetrahedral-hexahedral-pyramidal partitions, Comput. Methods Appl. Mech. Eng., 304 (2016), 140-170.
- **Inversion of BB mass matrix:** R. Kirby, Fast inversion of the simplicial Bernstein mass matrix. Numer. Math., 135 (2017), 73–95.
- **Comparison of GPU implementation of BB and Lagrange basis functions:** Jesse Chan and T. Warburton, GPU-accelerated Bernstein-Bezier discontinuous Galerkin methods for wave problems, SIAM J. Sci. Comp., 39 (2017), A628-A654.

Library BBFEM

C++ code for the BB-moments, system matrices and all experiments is available under GPL from

www.bbfem.de