An Introduction to Finite Element Methods

Veronika Pillwein

RISC

JKU JOHANNES KEPLER UNIVERSITY LINZ

Der Wissenschaftsfonds.

ISSAC 2015, 40th International Symposium on Symbolic and Algebraic Computation, Bath, July 6–9, 2015

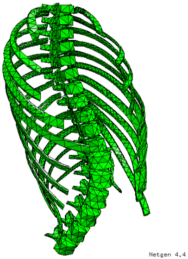
Finite element methods (FEM)

Numerical methods for finding *approximate solutions* to *partial differential equations* (PDEs) on *non-trivial domains*.

Finite element methods (FEM)

Numerical methods for finding *approximate solutions* to *partial differential equations* (PDEs) on *non-trivial domains*.

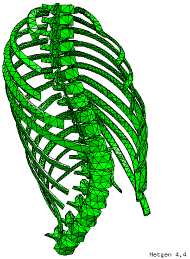
The domain is subdivided into *simple geometrical objects* (triangles, tetrahedra, . . .) and the solution is then approximated by *locally supported, piecewise polynomial basis functions*.



Finite element methods (FEM)

Numerical methods for finding *approximate solutions* to *partial differential equations* (PDEs) on *non-trivial domains*.

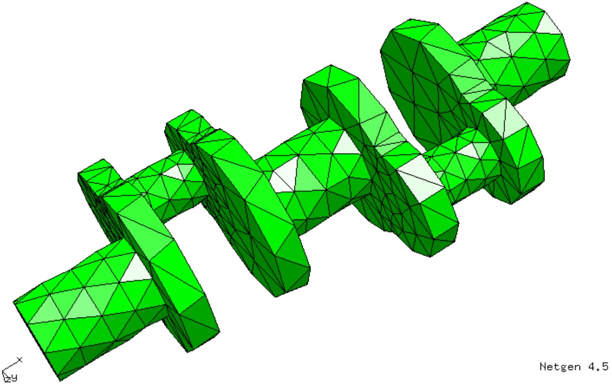
The domain is subdivided into *simple geometrical objects* (triangles, tetrahedra, ...) and the solution is then approximated by *locally supported, piecewise polynomial basis functions*.



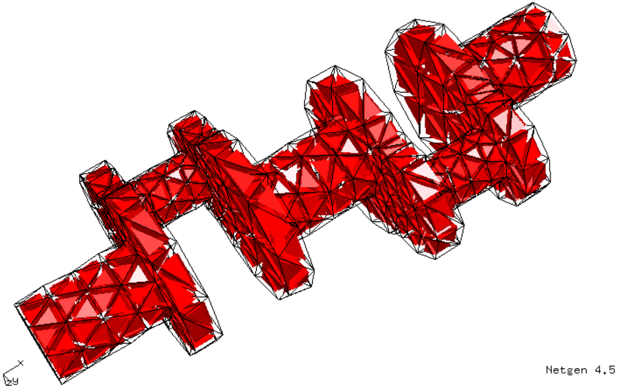
The task of solving the PDE is reduced to the solution of a (large scale) linear system.

$$(\text{Solving the PDE}) \longrightarrow (\text{Solving } A\underline{u} = \underline{f})$$

Another non-trivial domain



Another non-trivial domain



Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$
\begin{aligned}
-u''(x) &= f(x), \quad \text{in} \quad \Omega = (0,1) \\
u(0) = 0, &\quad u'(1) = 0.
\end{aligned}
$$

Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Let v be a sufficiently smooth function with $v(0) = 0$.

$$- \quad u''(x) \quad = \quad f(x)$$

Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Let v be a sufficiently smooth function with $v(0) = 0$. We multiply the equation above by $v(x)$

$$- \qquad u''(x)v(x) \quad = \quad f(x)v(x)$$

Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Let v be a sufficiently smooth function with $v(0) = 0$. We multiply the equation above by $v(x)$ and integrate over Ω.

$$-\int_0^1 u''(x)v(x)\,dx = \int_0^1 f(x)v(x)\,dx$$

Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Let v be a sufficiently smooth function with $v(0) = 0$. We multiply the equation above by $v(x)$ and integrate over Ω. Next we do integration by parts.

$$-\int_0^1 u''(x)v(x)\,dx = \int_0^1 f(x)v(x)\,dx$$
$$\int_0^1 u'(x)v'(x)\,dx - \left[u'(1)v(1) - u'(0)v(0)\right] = \int_0^1 f(x)v(x)\,dx$$

Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$
\begin{aligned}
-u''(x) &= f(x), \quad \text{in} \quad \Omega = (0,1) \\
u(0) &= 0, \quad u'(1) = 0.
\end{aligned}
$$

- Let v be a sufficiently smooth function with $v(0) = 0$. We multiply the equation above by $v(x)$ and integrate over Ω. Next we do integration by parts. Now we use the Neumann boundary condition

$$
-\int_0^1 u''(x)v(x)\,dx = \int_0^1 f(x)v(x)\,dx
$$

$$
\int_0^1 u'(x)v'(x)\,dx - \left[u'(1)v(1) - u'(0)v(0)\right] = \int_0^1 f(x)v(x)\,dx
$$

Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$\begin{aligned} -u''(x) &= f(x), \quad \text{in} \quad \Omega = (0,1) \\ u(0) = 0, &\quad u'(1) = 0. \end{aligned}$$

- Let v be a sufficiently smooth function with $v(0) = 0$. We multiply the equation above by $v(x)$ and integrate over Ω. Next we do integration by parts. Now we use the Neumann boundary condition and the essential condition on v.

$$-\int_0^1 u''(x)v(x)\,dx = \int_0^1 f(x)v(x)\,dx$$

$$\int_0^1 u'(x)v'(x)\,dx - \Big[\qquad\qquad - u'(0)v(0)\Big] = \int_0^1 f(x)v(x)\,dx$$

Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$
\begin{aligned}
-u''(x) &= f(x), \quad \text{in} \quad \Omega = (0,1) \\
u(0) &= 0, \quad u'(1) = 0.
\end{aligned}
$$

- Let v be a sufficiently smooth function with $v(0) = 0$. We multiply the equation above by $v(x)$ and integrate over Ω. Next we do integration by parts. Now we use the Neumann boundary condition and the essential condition on v.

$$
\begin{aligned}
-\int_0^1 u''(x)v(x)\,dx &= \int_0^1 f(x)v(x)\,dx \\
\int_0^1 u'(x)v'(x)\,dx &= \int_0^1 f(x)v(x)\,dx
\end{aligned}
$$

Variational/Weak formulation (1D)

- Consider the two-point boundary value problem: given f, find u such that

$$
\begin{aligned}
-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1) \\
u(0) = 0, \quad u'(1) = 0.
\end{aligned}
$$

- Let v be a sufficiently smooth function with $v(0) = 0$. We multiply the equation above by $v(x)$ and integrate over Ω. Next we do integration by parts. Now we use the Neumann boundary condition and the essential condition on v.

$$
\begin{aligned}
-\int_0^1 u''(x)v(x)\,dx &= \int_0^1 f(x)v(x)\,dx \\
\int_0^1 u'(x)v'(x)\,dx &= \int_0^1 f(x)v(x)\,dx
\end{aligned}
$$

$$
a(u,v) := \int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx =: F(v)
$$

Variational/Weak formulation (1D)

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$
\begin{aligned}
-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1) \\
u(0) = 0, \quad u'(1) = 0.
\end{aligned}
$$

Variational/Weak formulation (1D)

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Given f, find u such that

$$a(u,v) := \int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx =: F(v)$$

for all v sufficiently smooth with $v(0) = 0$.

Variational/Weak formulation (1D)

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$\begin{aligned} -u''(x) &= f(x), \quad \text{in} \quad \Omega = (0,1) \\ u(0) = 0, &\quad u'(1) = 0. \end{aligned}$$

- Given f, find u such that

$$a(u,v) := \int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx =: F(v)$$

for all v sufficiently smooth with $v(0) = 0$.

Galerkin method

- *Problem:* Given a *bilinear form* $a(\cdot,\cdot)$ and a *linear form* $F(\cdot)$,

$$\text{find } u \in V: \quad a(u,v) = F(v) \quad \forall\, v \in V,$$

where V is some infinite dimensional function space.

Galerkin method

- *Problem:* Given a *bilinear form* $a(\cdot,\cdot)$ and a *linear form* $F(\cdot)$,

$$\text{find } u \in V: \quad a(u,v) = F(v) \quad \forall\, v \in V,$$

  where V is some infinite dimensional function space.
- Approximate V by finite dimensional subspaces $V_h \subset V$.

Galerkin method

- *Problem:* Given a *bilinear form* $a(\cdot, \cdot)$ and a *linear form* $F(\cdot)$,

$$\text{find } u \in V: \quad a(u, v) = F(v) \quad \forall\, v \in V,$$

  where V is some infinite dimensional function space.
- Approximate V by finite dimensional subspaces $V_h \subset V$.
- *Discrete problem:* Given $a(\cdot, \cdot)$ and $F(\cdot)$,

$$\text{find } u_h \in V_h: \quad a(u_h, v_h) = F(v_h) \quad \forall\, v_h \in V_h.$$

Galerkin method

- *Problem:* Given a *bilinear form* $a(\cdot,\cdot)$ and a *linear form* $F(\cdot)$,

$$\text{find } u \in V: \quad a(u,v) = F(v) \quad \forall\, v \in V,$$

  where V is some infinite dimensional function space.
- Approximate V by finite dimensional subspaces $V_h \subset V$.
- *Discrete problem:* Given $a(\cdot,\cdot)$ and $F(\cdot)$,

$$\text{find } u_h \in V_h: \quad a(u_h,v_h) = F(v_h) \quad \forall\, v_h \in V_h.$$

- Let $N_h = \dim V_h$ and $\{\phi_1, \ldots, \phi_{N_h}\}$ be a basis for V_h, then we can expand

$$u_h(x) = \sum_{i=1}^{N_h} u_i \phi_i(x)$$

  and it is sufficient to consider $v_h(x) = \phi_j(x)$ for $j = 1, \ldots, N_h$.

Galerkin method

- Plug $u_h(x) = \sum_{i=1}^{N_h} u_i \phi_i(x)$ and $v_h(x) = \phi_j(x)$ into

$$\text{find } u_h \in V_h : \quad a(u_h, v_h) = F(v_h) \quad \forall\, v_h \in V_h$$

to obtain

$$\text{find } \underline{u} \in \mathbb{R}^{N_h} : \quad \sum_{i=1}^{N_h} u_i a(\phi_i, \phi_j) = F(\phi_j), \quad \forall\, j = 1, \ldots, N_h.$$

where $\underline{u} = (u_1, \ldots, u_{N_h})$.

Galerkin method

- Plug $u_h(x) = \sum_{i=1}^{N_h} u_i \phi_i(x)$ and $v_h(x) = \phi_j(x)$ into

  $$\text{find } u_h \in V_h : \quad a(u_h, v_h) = F(v_h) \quad \forall\, v_h \in V_h$$

  to obtain

  $$\text{find } \underline{u} \in \mathbb{R}^{N_h} : \quad \sum_{i=1}^{N_h} u_i a(\phi_i, \phi_j) = F(\phi_j), \quad \forall\, j = 1, \ldots, N_h.$$

  where $\underline{u} = (u_1, \ldots, u_{N_h})$.
- Let

  $$A = (a(\phi_i, \phi_j))_{i,j=1}^{N_h} \quad \text{and} \quad \underline{f} = (f_1, \ldots, f_{N_h}),$$

  then we arrive at the *linear system*

  $$\text{find } \underline{u} \in \mathbb{R}^{N_h} : \quad A\underline{u} = \underline{f}.$$

Variational/Weak formulation (1D)

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Given f, find u such that

$$a(u,v) := \int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx =: F(v)$$

for all v sufficiently smooth with $v(0) = 0$.

Variational/Weak formulation (1D)

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Given f, find u such that

$$a(u,v) := \int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx =: F(v)$$

for all v sufficiently smooth with $v(0) = 0$.

- Let

$$L^2(\Omega) = \{f \colon \int_\Omega f(x)^2\,dx < \infty\},$$

then we require that $f \in L^2(0,1)$ and define

$$V = \{v \in L^2(0,1) \colon a(v,v) < \infty \text{ and } v(0) = 0\}.$$

Variational/Weak formulation (1D)

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Given f, find u such that

$$a(u,v) := \int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx =: F(v)$$

for all v sufficiently smooth with $v(0) = 0$.

- Let

$$L^2(\Omega) = \{f \colon \int_\Omega f(x)^2\,dx < \infty\},$$

then we require that $f \in L^2(0,1)$ and define

$$V = \{v \in L^2(0,1) \colon a(v,v) < \infty \text{ and } v(0) = 0\}.$$

Weak derivative

A function $f \in L^2(a,b)$ is *weakly differentiable*, if there exists $w \in L^1_{loc}(a,b)$ satisfying

$$\int_a^b f(x)v'(x)\,dx = -\int_a^b w(x)v(x)\,dx$$

for all $v \in C_0^\infty(a,b)$. If such a w exists, then it is unique a.e. and we write $f'(x) = w(x)$.

Weak derivative

A function $f \in L^2(a,b)$ is *weakly differentiable*, if there exists $w \in L^1_{loc}(a,b)$ satisfying

$$\int_a^b f(x)v'(x)\,dx = -\int_a^b w(x)v(x)\,dx$$

for all $v \in C_0^\infty(a,b)$. If such a w exists, then it is unique a.e. and we write $f'(x) = w(x)$.

- Weak and strong derivative coincide (in the L^2-sense) if f is classically differentiable.

Weak derivative

A function $f \in L^2(a, b)$ is *weakly differentiable*, if there exists $w \in L^1_{loc}(a, b)$ satisfying

$$\int_a^b f(x)v'(x)\,dx = -\int_a^b w(x)v(x)\,dx$$

for all $v \in C_0^\infty(a, b)$. If such a w exists, then it is unique a.e. and we write $f'(x) = w(x)$.

- Weak and strong derivative coincide (in the L^2-sense) if f is classically differentiable.
- Continuous functions that are piecewise differentiable are weakly differentiable.

Weak derivative

A function $f \in L^2(a,b)$ is *weakly differentiable*, if there exists $w \in L^1_{loc}(a,b)$ satisfying

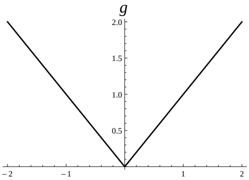
$$\int_a^b f(x)v'(x)\,dx = -\int_a^b w(x)v(x)\,dx$$

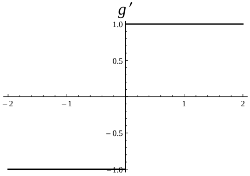
for all $v \in C_0^\infty(a,b)$. If such a w exists, then it is unique a.e. and we write $f'(x) = w(x)$.

- Weak and strong derivative coincide (in the L^2-sense) if f is classically differentiable.
- Continuous functions that are piecewise differentiable are weakly differentiable.
- Functions with jumps are *not* weakly differentiable.

Examples

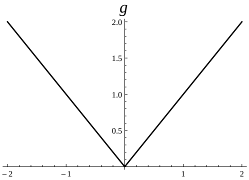
- $$g(x) = |x|$$

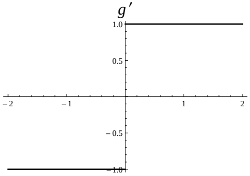




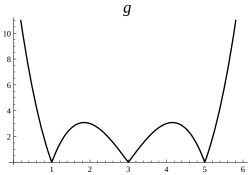
Examples

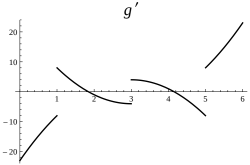
- $$g(x) = |x|$$





- $$g(x) = |(x-1)(x-3)(x-5)|$$





Model problem in 1D

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

Model problem in 1D

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Given $f \in L^2(0,1)$, find $u \in V$ such that

$$\int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx, \quad \forall\, v \in V,$$

where $V = \{v \in L^2(0,1) \colon a(v,v) < \infty \text{ and } v(0) = 0\}$.

Model problem in 1D

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Given $f \in L^2(0,1)$, find $u \in V$ such that

$$\int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx, \quad \forall\, v \in V,$$

  where $V = \{v \in L^2(0,1) \colon a(v,v) < \infty \text{ and } v(0) = 0\}$.
- Find $\underline{u} \in \mathbb{R}^{N_h}$:

$$A\underline{u} = \underline{f},$$

  where $N_h = \dim V_h \subset V$, and the approximate solution is $u_h(x) = \sum_{i=1}^{N_h} u_i \phi_i(x)$ for a basis $\{\phi_1(x), \ldots, \phi_{N_h}(x)\}$.

Model problem in 1D

- Given $f \in C(0,1)$, find $u \in C^2(0,1)$ such that

$$-u''(x) = f(x), \quad \text{in} \quad \Omega = (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

- Given $f \in L^2(0,1)$, find $u \in V$ such that

$$\int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx, \quad \forall\, v \in V,$$

where $V = \{v \in L^2(0,1) \colon a(v,v) < \infty \text{ and } v(0) = 0\}$.

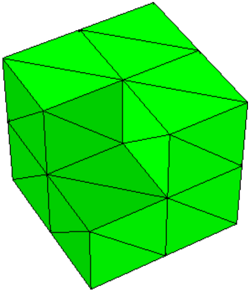
- Find $\underline{u} \in \mathbb{R}^{N_h}$:

$$A\underline{u} = \underline{f},$$

where $N_h = \dim V_h \subset V$, and the approximate solution is $u_h(x) = \sum_{i=1}^{N_h} u_i\phi_i(x)$ for a basis $\{\phi_1(x), \ldots, \phi_{N_h}(x)\}$.

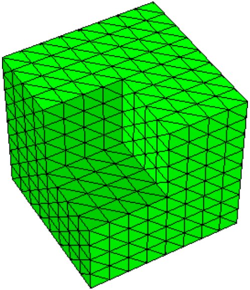
Strategies for increasing the accuracy

- *initial mesh*



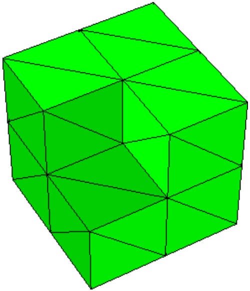
Strategies for increasing the accuracy

- *initial mesh*
- *h-version:* mesh refinement while keeping the polynomial degree fixed



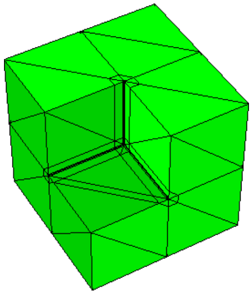
Strategies for increasing the accuracy

- *initial mesh*
- *h-version:* mesh refinement while keeping the polynomial degree fixed
- *p-version:* increasing the polynomial degree while keeping the mesh fixed



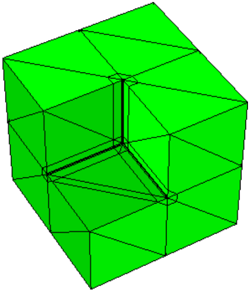
Strategies for increasing the accuracy

- *initial mesh*
- *h-version:* mesh refinement while keeping the polynomial degree fixed
- *p-version:* increasing the polynomial degree while keeping the mesh fixed
- *hp-version:* combination of both strategies



Strategies for increasing the accuracy

- *initial mesh*
- *h-version:* mesh refinement while keeping the polynomial degree fixed
- *p-version:* increasing the polynomial degree while keeping the mesh fixed
- *hp-version:* combination of both strategies



- higher order (i.e., p- and hp-) methods converge faster with respect to the number of unknowns, but require more effort in the implementation.

Conforming finite element bases

- The given domain is subdivided into *simple geometric objects* (intervals, quadrilaterals, triangles, prisms, tetrahedra, etc.) that are called *elements*.

Conforming finite element bases

- The given domain is subdivided into *simple geometric objects* (intervals, quadrilaterals, triangles, prisms, tetrahedra, etc.) that are called *elements*.
- A *finite element basis* consists of *locally supported, piecewise polynomial basis functions* $\phi_i(x)$.

Conforming finite element bases

- The given domain is subdivided into *simple geometric objects* (intervals, quadrilaterals, triangles, prisms, tetrahedra, etc.) that are called *elements*.
- A *finite element basis* consists of *locally supported, piecewise polynomial basis functions* $\phi_i(x)$.
- A finite element basis is *conforming* if *for all* i, $\phi_i \in V$, where V is the function space where the variational problem is posed.

Conforming finite element bases

- The given domain is subdivided into *simple geometric objects* (intervals, quadrilaterals, triangles, prisms, tetrahedra, etc.) that are called *elements*.
- A *finite element basis* consists of *locally supported, piecewise polynomial basis functions* $\phi_i(x)$.
- A finite element basis is *conforming* if *for all* i, $\phi_i \in V$, where V is the function space where the variational problem is posed.
- For 1D we consider next examples for two types of conforming finite element bases: a *nodal basis* and a *hierarchic basis*.

Conforming finite element bases

- The given domain is subdivided into *simple geometric objects* (intervals, quadrilaterals, triangles, prisms, tetrahedra, etc.) that are called *elements*.
- A *finite element basis* consists of *locally supported, piecewise polynomial basis functions* $\phi_i(x)$.
- A finite element basis is *conforming* if *for all* i, $\phi_i \in V$, where V is the function space where the variational problem is posed.
- For 1D we consider next examples for two types of conforming finite element bases: a *nodal basis* and a *hierarchic basis*.
- The basis functions are defined on a *reference element* and then mapped to the actual element in the *mesh*.

Setting for the 1D model problem

- Given $f \in L^2(0,1)$, find $u \in V$ such that

$$\underbrace{\int_0^1 u'(x)v'(x)\,dx}_{=a(u,v)} = \underbrace{\int_0^1 f(x)v(x)\,dx}_{=F(v)}, \quad \forall\, v \in V,$$

where $V = \{v \in L^2(0,1) \colon a(v,v) < \infty \text{ and } v(0) = 0\}$.

Setting for the 1D model problem

- Given $f \in L^2(0,1)$, find $u \in V$ such that

$$\underbrace{\int_0^1 u'(x)v'(x)\,dx}_{=a(u,v)} = \underbrace{\int_0^1 f(x)v(x)\,dx}_{=F(v)}, \quad \forall\, v \in V,$$

  where $V = \{v \in L^2(0,1) \colon a(v,v) < \infty \text{ and } v(0) = 0\}$.

- *Regularity* requirements for the basis functions: weakly differentiable and square integrable on $(0,1)$.

Setting for the 1D model problem

- Given $f \in L^2(0,1)$, find $u \in V$ such that

$$\underbrace{\int_0^1 u'(x)v'(x)\,dx}_{=a(u,v)} = \underbrace{\int_0^1 f(x)v(x)\,dx}_{=F(v)}, \quad \forall\, v \in V,$$

  where $V = \{v \in L^2(0,1)\colon a(v,v) < \infty \text{ and } v(0) = 0\}$.
- *Regularity* requirements for the basis functions: weakly differentiable and square integrable on $(0,1)$. We will define *continuous, piecewise polynomial basis functions*.

Setting for the 1D model problem

- Given $f \in L^2(0,1)$, find $u \in V$ such that

$$\underbrace{\int_0^1 u'(x)v'(x)\,dx}_{=a(u,v)} = \underbrace{\int_0^1 f(x)v(x)\,dx}_{=F(v)}, \quad \forall\, v \in V,$$

  where $V = \{v \in L^2(0,1)\colon a(v,v) < \infty \text{ and } v(0) = 0\}$.
- *Regularity* requirements for the basis functions: weakly differentiable and square integrable on $(0,1)$. We will define *continuous, piecewise polynomial basis functions*.
- For sake of simplicity we use an *equidistant* subdivision of $(0,1)$ with the *same* maximal polynomial *degree* p on each element.

Setting for the 1D model problem

- Given $f \in L^2(0,1)$, find $u \in V$ such that

$$\underbrace{\int_0^1 u'(x)v'(x)\,dx}_{=a(u,v)} = \underbrace{\int_0^1 f(x)v(x)\,dx}_{=F(v)}, \quad \forall\, v \in V,$$

  where $V = \{v \in L^2(0,1)\colon a(v,v) < \infty \text{ and } v(0) = 0\}$.
- *Regularity* requirements for the basis functions: weakly differentiable and square integrable on $(0,1)$. We will define *continuous, piecewise polynomial basis functions*.
- For sake of simplicity we use an *equidistant* subdivision of $(0,1)$ with the *same* maximal polynomial *degree* p on each element. *Note* that the construction we present allows to *vary the polynomial degree* on the elements.

High order nodal finite element basis

- reference element $\hat{\mathcal{I}} = [-1, 1]$, maximal polynomial degree p

High order nodal finite element basis

- reference element $\hat{\mathcal{I}} = [-1, 1]$, maximal polynomial degree p
- Lagrange basis polynomials:

$$\ell_j(x) = \prod_{i \neq j} \frac{x - x_i}{x_j - x_i},$$

where $\{x_1, \ldots, x_{p+1}\}$ is a subdivision of $\hat{\mathcal{I}}$.

High order nodal finite element basis

- reference element $\hat{\mathcal{I}} = [-1, 1]$, maximal polynomial degree p
- Lagrange basis polynomials:

$$\ell_j(x) = \prod_{i \neq j} \frac{x - x_i}{x_j - x_i},$$

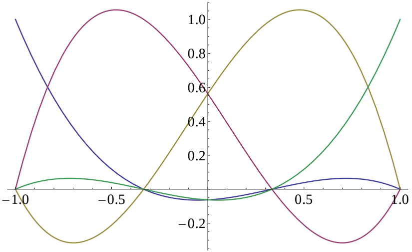
where $\{x_1, \ldots, x_{p+1}\}$ is an equidistant subdivision of $\hat{\mathcal{I}}$.

High order nodal finite element basis

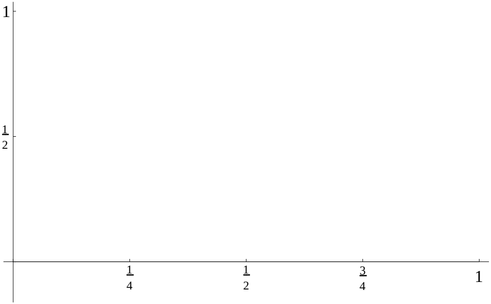
- reference element $\hat{\mathcal{I}} = [-1, 1]$, maximal polynomial degree p
- Lagrange basis polynomials:

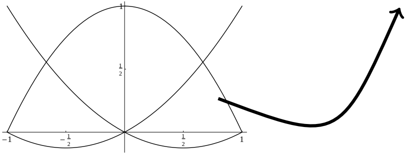
$$\ell_j(x) = \prod_{i \neq j} \frac{x - x_i}{x_j - x_i},$$

  where $\{x_1, \ldots, x_{p+1}\}$ is an equidistant subdivision of $\hat{\mathcal{I}}$.
- Example $p = 3$:



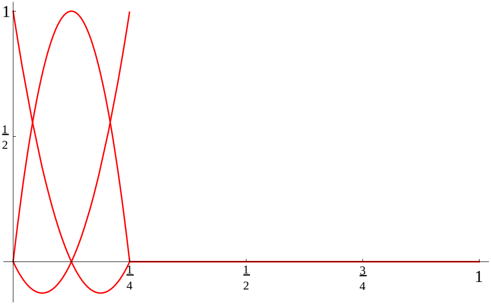
Local and global basis ($p = 2$, 4 elements)



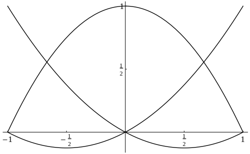


$\ell_1(x), \ell_2(x), \ell_3(x)$

Local and global basis ($p = 2$, 4 elements)

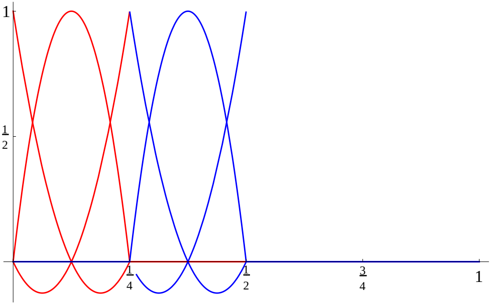


$\ell_1^{(1)}(x), \ell_2^{(1)}(x), \ell_3^{(1)}(x)$

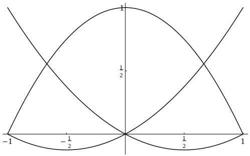


$\ell_1(x), \ell_2(x), \ell_3(x)$

Local and global basis ($p = 2$, 4 elements)

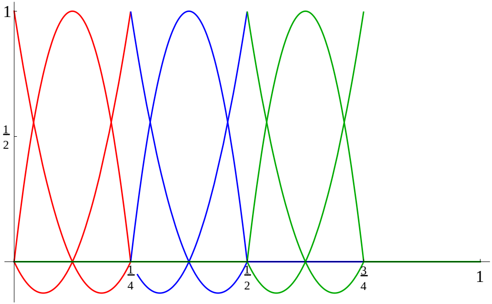


$\ell_1^{(1)}(x), \ell_2^{(1)}(x), \ell_3^{(1)}(x), \ell_1^{(2)}(x), \ell_2^{(2)}(x), \ell_3^{(2)}(x)$

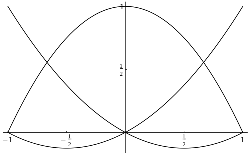


$\ell_1(x), \ell_2(x), \ell_3(x)$

Local and global basis ($p = 2$, 4 elements)

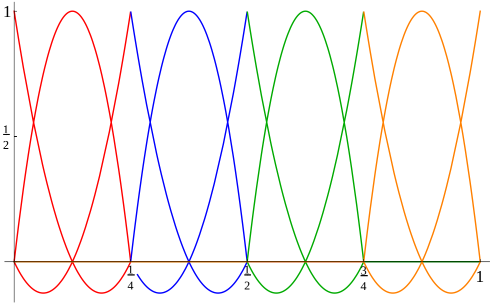


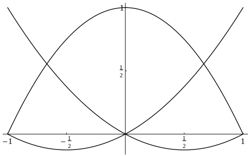
$\ell_1^{(1)}(x), \ell_2^{(1)}(x), \ell_3^{(1)}(x), \ell_1^{(2)}(x), \ell_2^{(2)}(x),$
$\ell_3^{(2)}(x), \ell_1^{(3)}(x), \ell_2^{(3)}(x), \ell_3^{(3)}(x)$



$\ell_1(x), \ell_2(x), \ell_3(x)$

Local and global basis ($p = 2$, 4 elements)

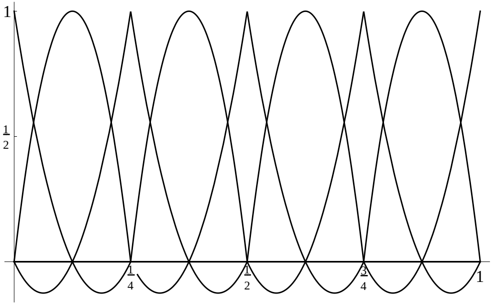


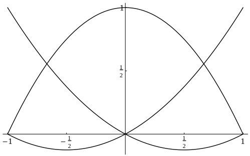


$\ell_1(x), \ell_2(x), \ell_3(x)$

$\ell_1^{(1)}(x), \ell_2^{(1)}(x), \ell_3^{(1)}(x), \ell_1^{(2)}(x), \ell_2^{(2)}(x), \ell_3^{(2)}(x), \ell_1^{(3)}(x), \ell_2^{(3)}(x), \ell_3^{(3)}(x), \ell_1^{(4)}(x), \ell_2^{(4)}(x), \ell_3^{(4)}(x)$

Local and global basis ($p = 2$, 4 elements)



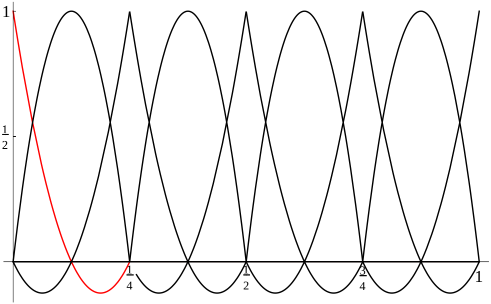


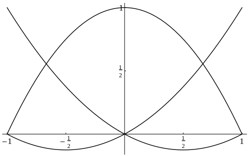
$\ell_1(x), \ell_2(x), \ell_3(x)$

$\ell_1^{(1)}(x), \ell_2^{(1)}(x), \ell_3^{(1)}(x), \ell_1^{(2)}(x), \ell_2^{(2)}(x), \ell_3^{(2)}(x), \ell_1^{(3)}(x), \ell_2^{(3)}(x), \ell_3^{(3)}(x), \ell_1^{(4)}(x), \ell_2^{(4)}(x), \ell_3^{(4)}(x)$

$\rightarrow$ *local basis functions*

Local and global basis ($p = 2$, 4 elements)



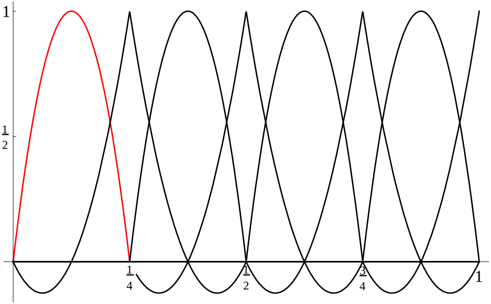


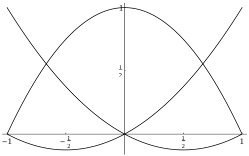
$\ell_1(x), \ell_2(x), \ell_3(x)$

$\ell_1^{(1)}(x), \ell_2^{(1)}(x), \ell_3^{(1)}(x), \ell_1^{(2)}(x), \ell_2^{(2)}(x), \ell_3^{(2)}(x), \ell_1^{(3)}(x), \ell_2^{(3)}(x), \ell_3^{(3)}(x), \ell_1^{(4)}(x), \ell_2^{(4)}(x), \ell_3^{(4)}(x)$

$\rightarrow$ *local basis functions*

Local and global basis ($p = 2$, 4 elements)





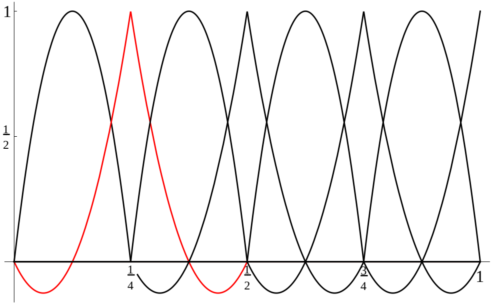
$\ell_1(x), \ell_2(x), \ell_3(x)$

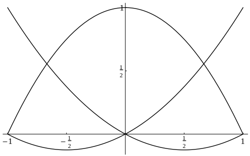
$\ell_2^{(1)}(x), \ell_3^{(1)}(x), \ell_1^{(2)}(x), \ell_2^{(2)}(x), \ell_3^{(2)}(x), \ell_1^{(3)}(x), \ell_2^{(3)}(x), \ell_3^{(3)}(x), \ell_1^{(4)}(x), \ell_2^{(4)}(x), \ell_3^{(4)}(x)$

$\rightarrow$ *local basis functions*

$\phi_1(x)$,

Local and global basis ($p = 2$, 4 elements)





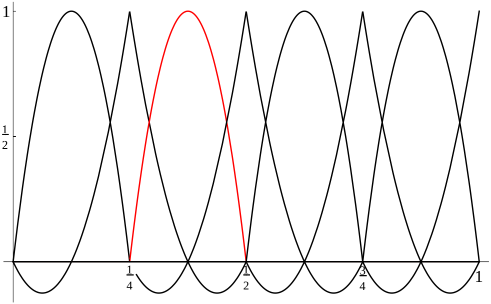
$\ell_1(x), \ell_2(x), \ell_3(x)$

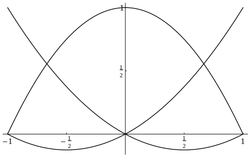
$\ell_2^{(1)}(x), \ell_3^{(1)}(x), \ell_1^{(2)}(x), \ell_2^{(2)}(x),$
$\ell_3^{(2)}(x), \ell_1^{(3)}(x), \ell_2^{(3)}(x), \ell_3^{(3)}(x), \ell_1^{(4)}(x),$
$\ell_2^{(4)}(x), \ell_3^{(4)}(x)$

$\rightarrow$ *local basis functions*

$\phi_1(x), \phi_2(x),$

Local and global basis ($p = 2$, 4 elements)





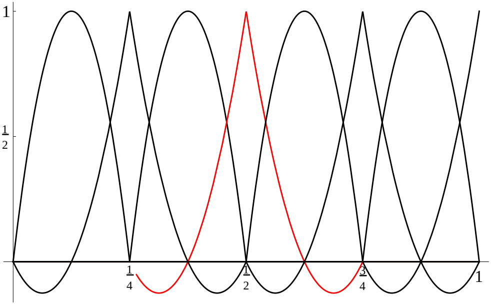
$\ell_1(x), \ell_2(x), \ell_3(x)$

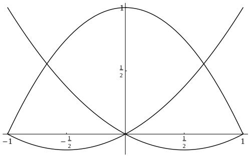
$\ell_2^{(1)}(x), \ell_3^{(1)}(x), \ell_1^{(2)}(x), \ell_2^{(2)}(x), \ell_3^{(2)}(x), \ell_1^{(3)}(x), \ell_2^{(3)}(x), \ell_3^{(3)}(x), \ell_1^{(4)}(x), \ell_2^{(4)}(x), \ell_3^{(4)}(x)$

$\rightarrow$ *local basis functions*

$\phi_1(x), \phi_2(x), \phi_3(x),$

Local and global basis ($p = 2$, 4 elements)

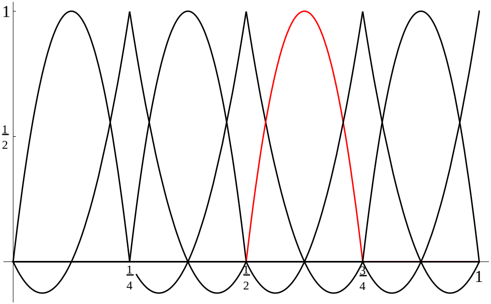


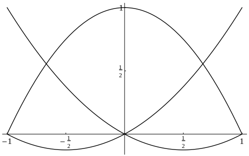


$\ell_1(x), \ell_2(x), \ell_3(x)$

$\phi_1(x), \phi_2(x), \phi_3(x), \phi_4(x),$

Local and global basis ($p = 2$, 4 elements)

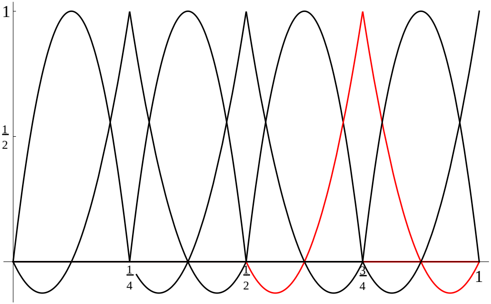


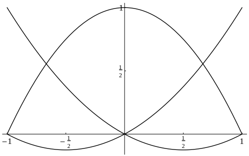


$\ell_1(x), \ell_2(x), \ell_3(x)$

$\phi_1(x), \phi_2(x), \phi_3(x), \phi_4(x), \phi_5(x),$

Local and global basis ($p = 2$, 4 elements)

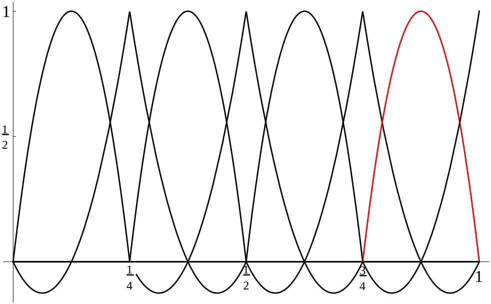


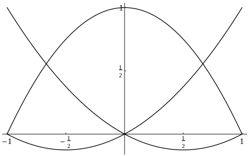


$\ell_1(x), \ell_2(x), \ell_3(x)$

$\phi_1(x), \phi_2(x), \phi_3(x), \phi_4(x), \phi_5(x),$
$\phi_6(x),$

Local and global basis ($p = 2$, 4 elements)

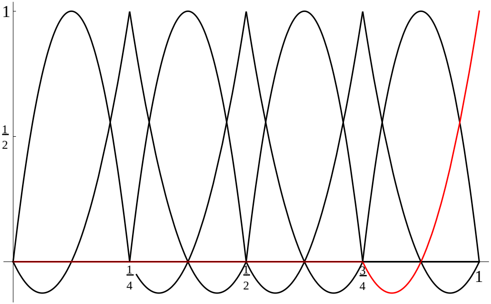


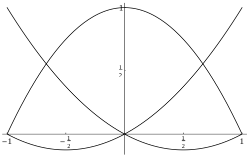


$\ell_1(x), \ell_2(x), \ell_3(x)$

$\phi_1(x), \phi_2(x), \phi_3(x), \phi_4(x), \phi_5(x),$
$\phi_6(x), \phi_7(x),$

Local and global basis ($p = 2$, 4 elements)

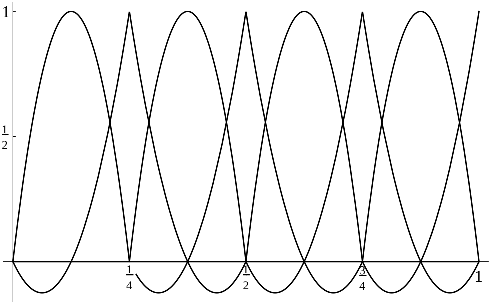


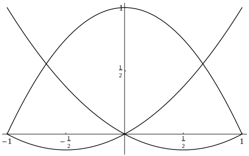


$\ell_1(x), \ell_2(x), \ell_3(x)$

$\phi_1(x), \phi_2(x), \phi_3(x), \phi_4(x), \phi_5(x),$
$\phi_6(x), \phi_7(x), \phi_8(x)$

Local and global basis ($p = 2$, 4 elements)



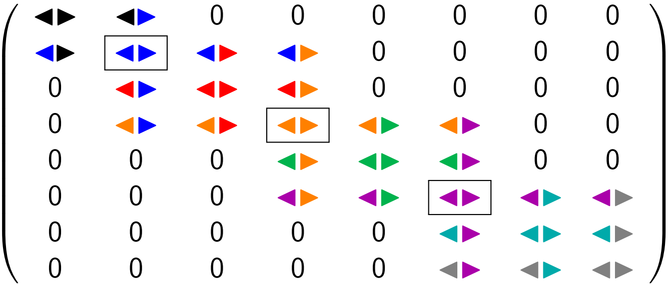


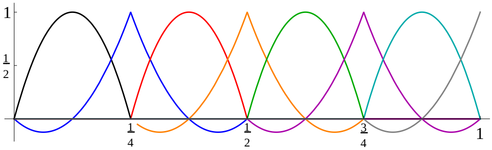
$\ell_1(x), \ell_2(x), \ell_3(x)$

$\phi_1(x), \phi_2(x), \phi_3(x), \phi_4(x), \phi_5(x),$
$\phi_6(x), \phi_7(x), \phi_8(x)$
$\rightarrow$ *global basis functions*

The system matrix

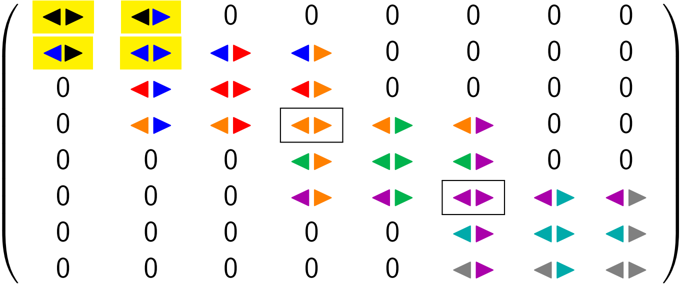
- $A = (a(\phi_i, \phi_j))_{i,j=1}^{N_h}$ with $a(u, v) = \int_0^1 u'(x)v'(x)\,dx$.

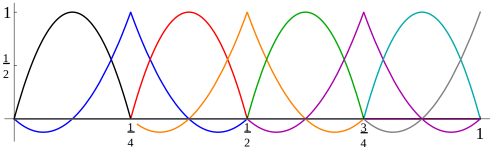
$A =$ 



The system matrix

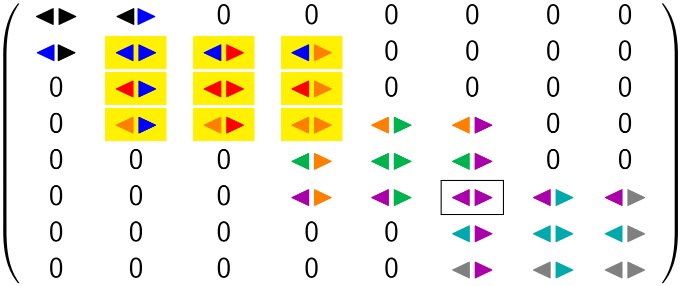
- $A = (a(\phi_i, \phi_j))_{i,j=1}^{N_h}$ with $a(u, v) = \int_0^1 u'(x) v'(x)\, dx$.

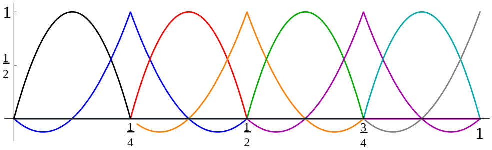
$A =$ 



The system matrix

- $A = (a(\phi_i, \phi_j)_{i,j=1}^{N_h}$ with $a(u, v) = \int_0^1 u'(x)v'(x)\,dx$.

$A =$ 



High order hierarchical finite element basis

- reference element $\hat{\mathcal{I}} = [-1, 1]$, maximal polynomial degree p

High order hierarchical finite element basis

- reference element $\hat{\mathcal{I}} = [-1, 1]$, maximal polynomial degree p
- *vertex based basis functions*: 1 in the defining vertex and 0 in all other vertices,

$$\varphi_{V,0}(x) = \frac{1-x}{2}, \quad \varphi_{V,1}(x) = \frac{1+x}{2}$$

High order hierarchical finite element basis

- reference element $\hat{\mathcal{I}} = [-1, 1]$, maximal polynomial degree p
- *vertex based basis functions*: 1 in the defining vertex and 0 in all other vertices,

$$\varphi_{V,0}(x) = \frac{1-x}{2}, \quad \varphi_{V,1}(x) = \frac{1+x}{2}$$

- *cell based basis functions*: vanish at the boundary of the element, polynomial inside:

$$\varphi_{C,i}(x) = L_i(x) := \int_{-1}^{x} P_{i-1}(x)\,dx, \quad i \geq 2,$$

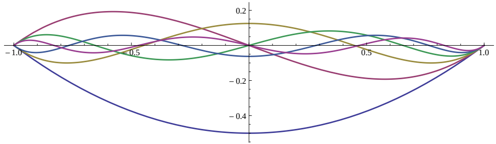
  where $P_n(x)$ is the nth Legendre polynomial. The polynomials $L_n(x)$ are called *integrated Legendre polynomials*.
- Legendre polynomials are orthogonal w.r.t. the $L^2(-1, 1)$ inner product, i.e.,

$$\int_{-1}^{1} P_i(x)P_j(x)\,dx = 0, \quad \text{if} \quad i \neq j.$$

Integrated Legendre polynomials

- For $n \geq 2$ with $L_0(x) = -1$ and $L_1(x) = x$ we have

$$L_n(x) = \frac{2n-3}{n} x L_{n-1}(x) - \frac{n-3}{n} L_{n-2}(x).$$

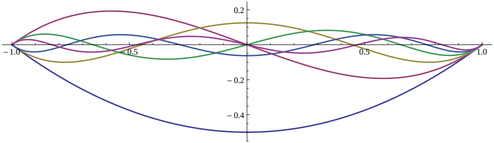


Integrated Legendre polynomials

- For $n \geq 2$ with $L_0(x) = -1$ and $L_1(x) = x$ we have

$$L_n(x) = \frac{2n-3}{n} x L_{n-1}(x) - \frac{n-3}{n} L_{n-2}(x).$$

- $L_n(\pm 1) = 0$



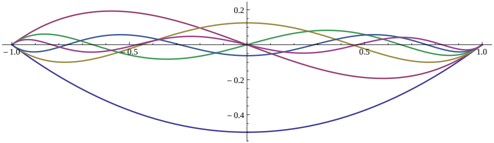
Integrated Legendre polynomials

- For $n \geq 2$ with $L_0(x) = -1$ and $L_1(x) = x$ we have

$$L_n(x) = \frac{2n-3}{n} x L_{n-1}(x) - \frac{n-3}{n} L_{n-2}(x).$$

- $L_n(\pm 1) = 0$
- Orthogonality relation

$$\int_{-1}^{1} L_i'(x) L_j'(x)\, dx = 0, \quad \text{if} \quad i \neq j$$



Integrated Legendre polynomials

- For $n \geq 2$ with $L_0(x) = -1$ and $L_1(x) = x$ we have

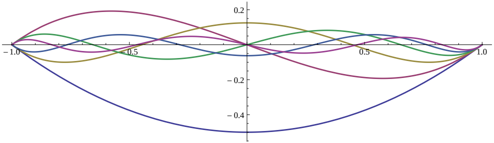
$$L_n(x) = \frac{2n-3}{n} x L_{n-1}(x) - \frac{n-3}{n} L_{n-2}(x).$$

- $L_n(\pm 1) = 0$
- Orthogonality relation

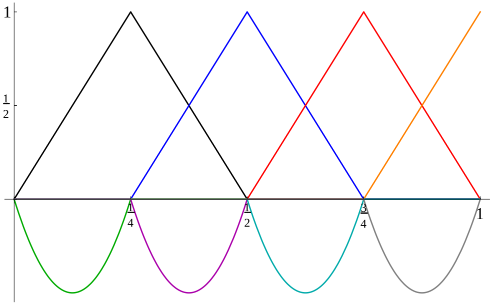
$$\int_{-1}^{1} L_i'(x) L_j'(x)\, dx = 0, \quad \text{if} \quad i \neq j$$

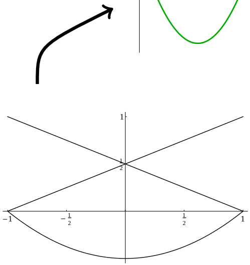
- Sparse w.r.t. the L^2 inner product

$$\int_{-1}^{1} L_i(x) L_j(x)\, dx = 0, \quad \text{if} \quad |i-j| \neq 0, 2$$



Local and global basis ($p = 2$, 4 elements)





$\varphi_{V,0}(x), \varphi_{V,1}(x), \varphi_{C,2}(x)$

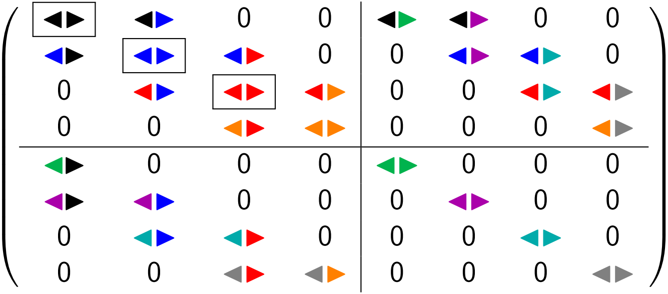
global basis:

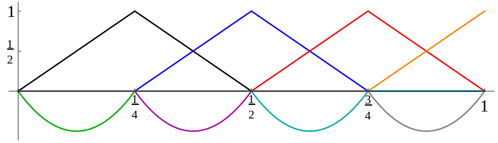
$\underbrace{\phi_{V_2}(x), \phi_{V_3}(x), \phi_{V_4}(x), \phi_{V_5}(x)}_{=\Phi_V},$

$\underbrace{\phi_{C_1,2}(x), \phi_{C_2,2}(x), \phi_{C_3,2}(x), \phi_{C_4,2}(x)}_{=\Phi_C}$

The system matrix

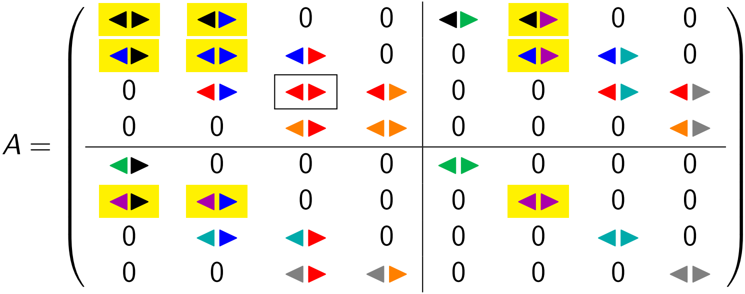
- $A = (a(\phi_i, \phi_j))_{i,j=1}^{N_h} = \left(\begin{array}{c|c} A_{VV} & A_{VC} \\ \hline A_{CV} & A_{CC} \end{array} \right)$ with $a(u,v) = \int_0^1 u'v'dx$.

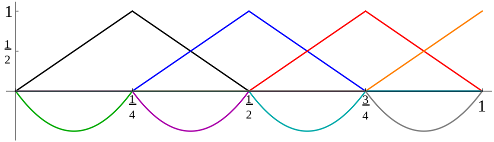
$A =$ 



The system matrix

- $A = (a(\phi_i, \phi_j)_{i,j=1}^{N_h} = \left(\begin{array}{c|c} A_{VV} & A_{VC} \\ \hline A_{CV} & A_{CC} \end{array}\right)$ with $a(u, v) = \int_0^1 u' v' dx$.



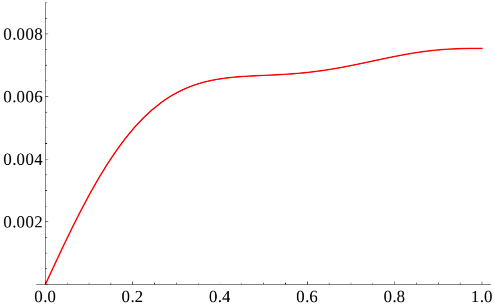


Example

$$\text{Find } u: \quad -u''(x) = \tfrac{1}{8}(2x-1)(4x-3)\sin(\pi x), \quad \text{in} \quad (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

Example

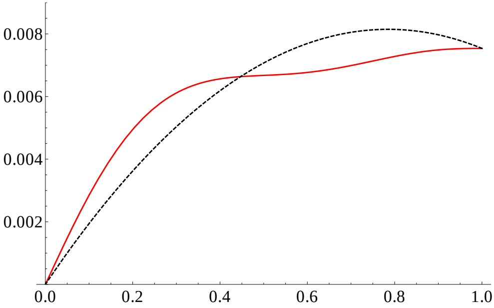
Find u: $$-u''(x) = \frac{1}{8}(2x-1)(4x-3)\sin(\pi x), \quad \text{in} \quad (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$



exact solution

Example

Find u:
$$-u''(x) = \tfrac{1}{8}(2x-1)(4x-3)\sin(\pi x), \quad \text{in} \quad (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

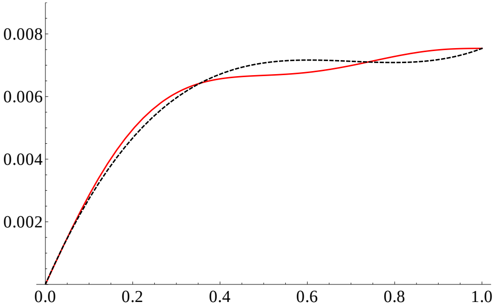


exact solution, approximation for $p = 2$

Example

Find u: $\quad -u''(x) = \frac{1}{8}(2x-1)(4x-3)\sin(\pi x), \quad \text{in} \quad (0,1)$

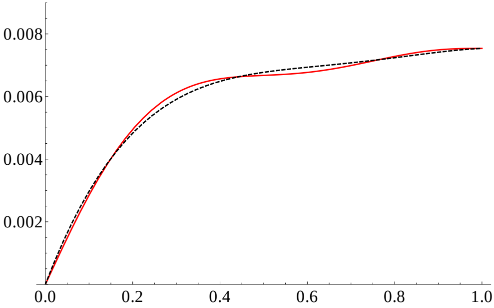
$$u(0) = 0, \quad u'(1) = 0.$$



exact solution, approximation for $p = 3$

Example

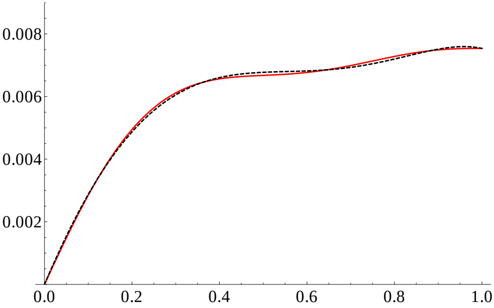
Find u: $$-u''(x) = \frac{1}{8}(2x-1)(4x-3)\sin(\pi x), \quad \text{in} \quad (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$



exact solution, approximation for $p = 4$

Example

Find u: $$-u''(x) = \frac{1}{8}(2x-1)(4x-3)\sin(\pi x), \quad \text{in} \quad (0,1)$$
$$u(0) = 0, \quad u'(1) = 0.$$

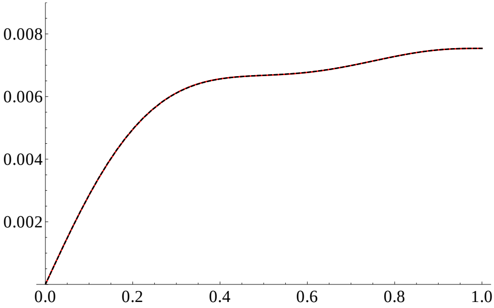


exact solution, approximation for $p = 5$

Example

Find u: $\quad -u''(x) = \frac{1}{8}(2x-1)(4x-3)\sin(\pi x), \quad \text{in} \quad (0,1)$

$$u(0) = 0, \quad u'(1) = 0.$$



exact solution, approximation for $p = 6$

Example: the system matrix for $p = 6$

in the hierarchic basis: In the nodal basis:

$$\left(\begin{array}{c|ccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & \frac{4}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{4}{5} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{4}{7} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{4}{9} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{4}{11} \end{array}\right) \left(\begin{array}{cccccc} 40. & -48. & 38. & -24. & 10. & -2.2 \\ -48. & 80. & -76. & 50. & -24. & 5.9 \\ 38. & -76. & 95. & -76. & 38. & -9.8 \\ -24. & 50. & -76. & 80. & -48. & 11. \\ 10. & -24. & 38. & -48. & 40. & -14. \\ -2.2 & 5.9 & -9.8 & 11. & -14. & 8.6 \end{array}\right)$$

Example: the system matrix for $p = 6$

in the hierarchic basis:

$$\left(\begin{array}{c|ccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & \frac{4}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{4}{5} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{4}{7} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{4}{9} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{4}{11} \end{array}\right)$$

- sparse element matrix

In the nodal basis:

$$\begin{pmatrix} 40. & -48. & 38. & -24. & 10. & -2.2 \\ -48. & 80. & -76. & 50. & -24. & 5.9 \\ 38. & -76. & 95. & -76. & 38. & -9.8 \\ -24. & 50. & -76. & 80. & -48. & 11. \\ 10. & -24. & 38. & -48. & 40. & -14. \\ -2.2 & 5.9 & -9.8 & 11. & -14. & 8.6 \end{pmatrix}$$

- full element matrix

Example: the system matrix for $p = 6$

in the hierarchic basis:

$$\left(\begin{array}{c|ccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & \frac{4}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{4}{5} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{4}{7} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{4}{9} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{4}{11} \end{array}\right)$$

- sparse element matrix
- add new polynomials when increasing the polynomial degree

In the nodal basis:

$$\begin{pmatrix} 40. & -48. & 38. & -24. & 10. & -2.2 \\ -48. & 80. & -76. & 50. & -24. & 5.9 \\ 38. & -76. & 95. & -76. & 38. & -9.8 \\ -24. & 50. & -76. & 80. & -48. & 11. \\ 10. & -24. & 38. & -48. & 40. & -14. \\ -2.2 & 5.9 & -9.8 & 11. & -14. & 8.6 \end{pmatrix}$$

- full element matrix
- recompute whole basis when increasing the polynomial degree

Example: the system matrix for $p = 6$

in the hierarchic basis:

$$\left(\begin{array}{c|ccccc} 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & \frac{4}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{4}{5} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{4}{7} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{4}{9} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{4}{11} \end{array}\right)$$

- sparse element matrix
- add new polynomials when increasing the polynomial degree
- recurrence for fast evaluation

In the nodal basis:

$$\begin{pmatrix} 40. & -48. & 38. & -24. & 10. & -2.2 \\ -48. & 80. & -76. & 50. & -24. & 5.9 \\ 38. & -76. & 95. & -76. & 38. & -9.8 \\ -24. & 50. & -76. & 80. & -48. & 11. \\ 10. & -24. & 38. & -48. & 40. & -14. \\ -2.2 & 5.9 & -9.8 & 11. & -14. & 8.6 \end{pmatrix}$$

- full element matrix
- recompute whole basis when increasing the polynomial degree

Variational/Weak formulation (2D)

- Model problem: given $f\colon \mathbb{R}^2 \to \mathbb{R}$, find $u\colon \mathbb{R}^2 \to \mathbb{R}$ s.t.

$$\begin{aligned} -\Delta u(x) &= f(x), && \text{in} \quad \Omega \\ u(x) &= 0, && \text{on} \quad \partial\Omega, \end{aligned}$$

where $\Delta g(x) = \sum_{i=1}^{d} \frac{\partial^2 g}{\partial x_i^2}(x)$ denotes the *Laplace operator*.

Variational/Weak formulation (2D)

- Model problem: given $f\colon \mathbb{R}^2 \to \mathbb{R}$, find $u\colon \mathbb{R}^2 \to \mathbb{R}$ s.t.

$$\begin{aligned} -\Delta u(x) &= f(x), && \text{in} \quad \Omega \\ u(x) &= 0, && \text{on} \quad \partial\Omega, \end{aligned}$$

  where $\Delta g(x) = \sum_{i=1}^{d} \frac{\partial^2 g}{\partial x_i^2}(x)$ denotes the *Laplace operator*.
- Let v be a sufficiently smooth function with $v\,|_{\partial\Omega} = 0$. We multiply the equation above by $v(x)$ and integrate over Ω.

$$-\int_\Omega \Delta u(x) v(x)\, dx = \int_\Omega f(x) v(x)\, dx$$

Variational/Weak formulation (2D)

- Model problem: given $f\colon \mathbb{R}^2 \to \mathbb{R}$, find $u\colon \mathbb{R}^2 \to \mathbb{R}$ s.t.

$$\begin{aligned} -\Delta u(x) &= f(x), && \text{in} \quad \Omega \\ u(x) &= 0, && \text{on} \quad \partial\Omega, \end{aligned}$$

  where $\Delta g(x) = \sum_{i=1}^{d} \frac{\partial^2 g}{\partial x_i^2}(x)$ denotes the *Laplace operator*.
- Let v be a sufficiently smooth function with $v\,|_{\partial\Omega} = 0$. We multiply the equation above by $v(x)$ and integrate over Ω. Next we apply Gauß' theorem

$$-\int_\Omega \Delta u(x) v(x)\, dx = \int_\Omega f(x) v(x)\, dx$$

$$\int_\Omega \nabla u(x) \nabla v(x)\, dx - \int_{\partial\Omega} \frac{\partial u}{\partial n}(x) v(x)\, dx = \int_\Omega f(x) v(x)\, dx$$

Variational/Weak formulation (2D)

- Model problem: given $f\colon \mathbb{R}^2 \to \mathbb{R}$, find $u\colon \mathbb{R}^2 \to \mathbb{R}$ s.t.

$$-\Delta u(x) = f(x), \quad \text{in} \quad \Omega$$
$$u(x) = 0, \quad \text{on} \quad \partial\Omega,$$

where $\Delta g(x) = \sum_{i=1}^{d} \frac{\partial^2 g}{\partial x_i^2}(x)$ denotes the *Laplace operator*.

- Let v be a sufficiently smooth function with $v \mid_{\partial\Omega} = 0$. We multiply the equation above by $v(x)$ and integrate over Ω. Next we apply Gauß' theorem and use the essential condition on v.

$$-\int_\Omega \Delta u(x) v(x)\, dx = \int_\Omega f(x) v(x)\, dx$$
$$\int_\Omega \nabla u(x) \nabla v(x)\, dx - \int_{\partial\Omega} \frac{\partial u}{\partial n}(x) v(x)\, dx = \int_\Omega f(x) v(x)\, dx$$

Variational/Weak formulation (2D)

- Model problem: given $f\colon \mathbb{R}^2 \to \mathbb{R}$, find $u\colon \mathbb{R}^2 \to \mathbb{R}$ s.t.

$$\begin{aligned} -\Delta u(x) &= f(x), && \text{in} \quad \Omega \\ u(x) &= 0, && \text{on} \quad \partial\Omega, \end{aligned}$$

  where $\Delta g(x) = \sum_{i=1}^{d} \frac{\partial^2 g}{\partial x_i^2}(x)$ denotes the *Laplace operator*.
- Let v be a sufficiently smooth function with $v\mid_{\partial\Omega} = 0$. We multiply the equation above by $v(x)$ and integrate over Ω. Next we apply Gauß' theorem and use the essential condition on v.

$$\begin{aligned} -\int_\Omega \Delta u(x) v(x)\, dx &= \int_\Omega f(x) v(x)\, dx \\ \int_\Omega \nabla u(x) \nabla v(x)\, dx &= \int_\Omega f(x) v(x)\, dx \\ a(u,v) := \int_\Omega \nabla u(x) \nabla v(x)\, dx = \int_\Omega f(x) v(x)\, dx &=: F(v) \end{aligned}$$

Model problem in 2D

- Given $f \in C(\Omega)$, find $u \in C^2(\Omega)$ such that

$$\begin{aligned} -\Delta u(x) &= f(x), \qquad && \text{in} \quad \Omega \\ u(x) &= 0, && \text{on} \quad \partial\Omega, \end{aligned}$$

- Given $f \in L^2(\Omega)$, find $u \in V$ such that

$$\int_\Omega \nabla u(x) \nabla v(x)\, dx = \int_\Omega f(x) v(x)\, dx, \quad \forall\, v \in V,$$

  where $V = \{v \in L^2(\Omega) \colon a(v,v) < \infty \text{ and } v\,|_{\partial\Omega} = 0\}$.

- Find $\underline{u} \in \mathbb{R}^{N_h}$:

$$A\underline{u} = \underline{f},$$

  where $N_h = \dim V_h \subset V$, and the approximate solution is $u_h(x) = \sum_{i=1}^{N_h} u_i \phi_i(x)$ for a basis $\{\phi_1(x), \ldots, \phi_{N_h}(x)\}$.

Hierarchic high order finite element basis in 2D

- *Vertex based basis functions*
- *Edge based basis functions*
- *Cell based basis functions*

Hierarchic high order finite element basis in 2D

- *Vertex based basis functions* 1 at the defining vertex, 0 on all other vertices, linear in between
- *Edge based basis functions*
- *Cell based basis functions*

Hierarchic high order finite element basis in 2D

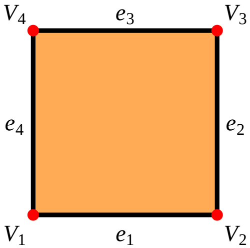
- *Vertex based basis functions* 1 at the defining vertex, 0 on all other vertices, linear in between
- *Edge based basis functions* polynomial on the defining edge, vanishing on all other edges, polynomial in between
- *Cell based basis functions*

Hierarchic high order finite element basis in 2D

- *Vertex based basis functions* 1 at the defining vertex, 0 on all other vertices, linear in between
- *Edge based basis functions* polynomial on the defining edge, vanishing on all other edges, polynomial in between
- *Cell based basis functions* supported only on the defining element, i.e., vanish on the boundary, polynomial on the interior of the element

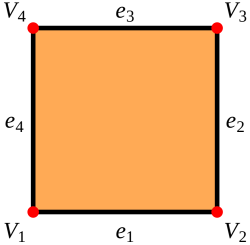
High order finite element basis on quadrilaterals

- reference element $\hat{Q} = [-1, 1]^2$



High order finite element basis on quadrilaterals

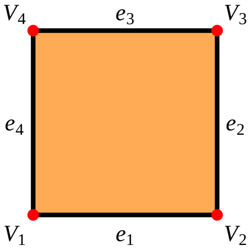
- reference element $\hat{Q} = [-1, 1]^2$
- tensor product of 1D basis functions



High order finite element basis on quadrilaterals

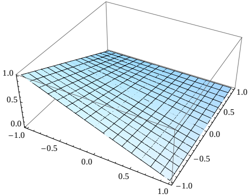
- reference element $\hat{Q} = [-1, 1]^2$
- tensor product of 1D basis functions
- *Vertex based basis functions:*

$$\varphi_{V_1}(x, y) = \varphi_{V,0}(x)\varphi_{V,0}(y) = \frac{1}{4}(1 - x)(1 - y)$$



High order finite element basis on quadrilaterals

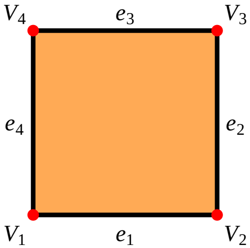
- reference element $\hat{Q} = [-1, 1]^2$
- tensor product of 1D basis functions
- *Vertex based basis functions:*



$$\begin{aligned}\varphi_{V_1}(x, y) &= \varphi_{V,0}(x)\varphi_{V,0}(y) \\ &= \tfrac{1}{4}(1-x)(1-y)\end{aligned}$$

High order finite element basis on quadrilaterals

- reference element $\hat{Q} = [-1, 1]^2$
- tensor product of 1D basis functions
- *Vertex based basis functions:*

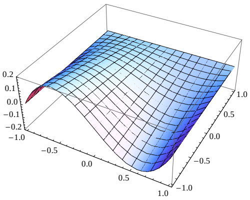


$$\varphi_{V_1}(x, y) = \varphi_{V,0}(x)\varphi_{V,0}(y) = \frac{1}{4}(1 - x)(1 - y)$$

- *Edge based basis functions:* $\varphi_{e_1,i}(x, y) = L_i(x)\varphi_{V,0}(y)$, for $i = 2, \ldots, p$

High order finite element basis on quadrilaterals

- reference element $\hat{Q} = [-1, 1]^2$
- tensor product of 1D basis functions
- *Vertex based basis functions:*



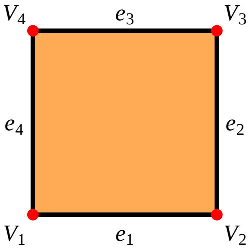
$$\varphi_{V_1}(x, y) = \varphi_{V,0}(x)\varphi_{V,0}(y) = \frac{1}{4}(1 - x)(1 - y)$$

- *Edge based basis functions:* $\varphi_{e_1,i}(x, y) = L_i(x)\varphi_{V,0}(y)$, for $i = 2, \ldots, p$

High order finite element basis on quadrilaterals

- reference element $\hat{Q} = [-1, 1]^2$
- tensor product of 1D basis functions
- *Vertex based basis functions:*

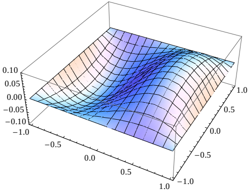
$$\begin{aligned} \varphi_{V_1}(x, y) &= \varphi_{V,0}(x)\varphi_{V,0}(y) \\ &= \tfrac{1}{4}(1 - x)(1 - y) \end{aligned}$$



- *Edge based basis functions:* $\varphi_{e_1,i}(x, y) = L_i(x)\varphi_{V,0}(y)$, for $i = 2, \ldots, p$
- *Cell based basis functions:* $\varphi_{C,i,j}(x, y) = L_i(x)L_j(y)$ for $i, j = 2, \ldots, p$

High order finite element basis on quadrilaterals

- reference element $\hat{Q} = [-1, 1]^2$
- tensor product of 1D basis functions
- *Vertex based basis functions:*

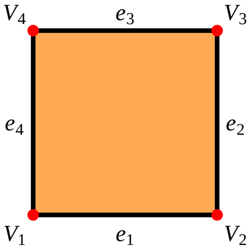


$$\varphi_{V_1}(x,y) = \varphi_{V,0}(x)\varphi_{V,0}(y)$$
$$= \frac{1}{4}(1-x)(1-y)$$

- *Edge based basis functions:* $\varphi_{e_1,i}(x,y) = L_i(x)\varphi_{V,0}(y)$, for $i = 2, \ldots, p$
- *Cell based basis functions:* $\varphi_{C,i,j}(x,y) = L_i(x)L_j(y)$ for $i, j = 2, \ldots, p$

High order finite element basis on quadrilaterals

- reference element $\hat{Q} = [-1, 1]^2$
- tensor product of 1D basis functions
- *Vertex based basis functions:*

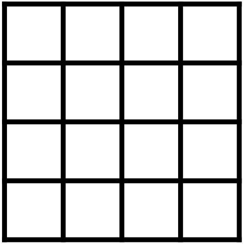


$$\begin{aligned}\varphi_{V_1}(x, y) &= \varphi_{V,0}(x)\varphi_{V,0}(y)\\ &= \tfrac{1}{4}(1-x)(1-y)\end{aligned}$$

- *Edge based basis functions:* $\varphi_{e_1,i}(x, y) = L_i(x)\varphi_{V,0}(y)$, for $i = 2, \ldots, p$
- *Cell based basis functions:* $\varphi_{C,i,j}(x, y) = L_i(x)L_j(y)$ for $i, j = 2, \ldots, p$
- *Local vector of basis functions:*

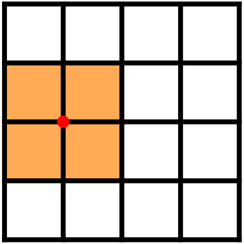
$$(\underbrace{\varphi_{V_1}, \varphi_{V_2}, \varphi_{V_3}, \varphi_{V_4}}_{\underline{\varphi}_V}, \underbrace{\varphi_{e_1,2}, \ldots, \varphi_{e_4,p}}_{\underline{\varphi}_E}, \underbrace{\varphi_{C,2,2}, \ldots, \varphi_{C,p,p}}_{\underline{\varphi}_C})$$

Global vector of basis functions



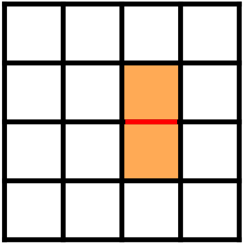
- Collect globally vertex, edge and cell based basis functions (Φ_V, Φ_E, Φ_C)

Global vector of basis functions



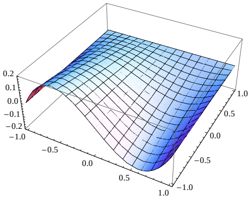
- Collect globally vertex, edge and cell based basis functions (Φ_V, Φ_E, Φ_C)
- *Vertex based basis functions* are supported on all elements sharing the defining vertex

Global vector of basis functions



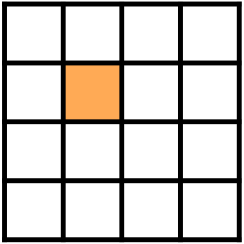
- Collect globally vertex, edge and cell based basis functions (Φ_V, Φ_E, Φ_C)
- *Vertex based basis functions* are supported on all elements sharing the defining vertex
- *Edge based basis functions* are supported on the two elements sharing the defining edge

Global vector of basis functions



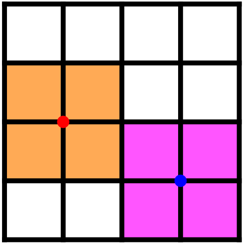
- Collect globally vertex, edge and cell based basis functions (Φ_V, Φ_E, Φ_C)
- *Vertex based basis functions* are supported on all elements sharing the defining vertex
- *Edge based basis functions* are supported on the two elements sharing the defining edge (Need to care about *orientation* of edges!)

Global vector of basis functions



- Collect globally vertex, edge and cell based basis functions (Φ_V, Φ_E, Φ_C)
- *Vertex based basis functions* are supported on all elements sharing the defining vertex
- *Edge based basis functions* are supported on the two elements sharing the defining edge (Need to care about *orientation* of edges!)
- *Cell based basis functions* are only supported on the defining element

Global vector of basis functions



- Collect globally vertex, edge and cell based basis functions (Φ_V, Φ_E, Φ_C)
- *Vertex based basis functions* are supported on all elements sharing the defining vertex
- *Edge based basis functions* are supported on the two elements sharing the defining edge (Need to care about *orientation* of edges!)
- *Cell based basis functions* are only supported on the defining element
- There can only be nonzero entries in the system matrix, if the *support* of two basis functions *overlaps*.

The system matrix on the reference element

- $A = (a(\phi_i, \phi_j))_{i,j}$ with $a(u, v) = \int_\Omega \nabla u(x, y) \nabla v(x, y) \, d(x, y)$

The system matrix on the reference element

- $A = (a(\phi_i, \phi_j)_{i,j}$ with $a(u, v) = \int_\Omega \nabla u(x, y) \nabla v(x, y)\, d(x, y)$
- All basis functions are tensor products: $\phi_i(x, y) = f_i(x) g_i(y)$, hence

$$\nabla \phi_i(x, y) = (f_i'(x) g_i(y), f_i(x) g_i'(y))$$

The system matrix on the reference element

- $A = (a(\phi_i, \phi_j)_{i,j}$ with $a(u, v) = \int_\Omega \nabla u(x, y) \nabla v(x, y)\, d(x, y)$
- All basis functions are tensor products: $\phi_i(x, y) = f_i(x) g_i(y)$, hence

$$\nabla \phi_i(x, y) = (f_i'(x) g_i(y), f_i(x) g_i'(y))$$

and $a(\phi_i(x, y), \phi_j(x, y)) =$

$$= \int_{\hat{Q}} \left(f_i'(x) f_j'(x) g_i(y) g_j(y) + f_i(x) f_j(x) g_i'(y) g_j'(y) \right) d(x, y)$$

The system matrix on the reference element

- $A = (a(\phi_i, \phi_j)_{i,j}$ with $a(u, v) = \int_\Omega \nabla u(x, y) \nabla v(x, y)\, d(x, y)$
- All basis functions are tensor products: $\phi_i(x, y) = f_i(x) g_i(y)$, hence

$$\nabla \phi_i(x, y) = (f_i'(x) g_i(y), f_i(x) g_i'(y))$$

and $a(\phi_i(x, y), \phi_j(x, y)) =$

$$= \int_{\hat{Q}} \left(f_i'(x) f_j'(x) g_i(y) g_j(y) + f_i(x) f_j(x) g_i'(y) g_j'(y) \right) d(x, y)$$

$$= \int_{-1}^{1} f_i'(x) f_j'(x)\, dx \int_{-1}^{1} g_i(y) g_j(y)\, dy$$

$$+ \int_{-1}^{1} f_i(x) f_j(x)\, dx \int_{-1}^{1} g_i'(y) g_j'(y))\, dy$$

The system matrix on the reference element

- $A = (a(\phi_i, \phi_j)_{i,j}$ with $a(u,v) = \int_\Omega \nabla u(x,y) \nabla v(x,y)\, d(x,y)$
- All basis functions are tensor products: $\phi_i(x,y) = f_i(x) g_i(y)$, hence
$$\nabla \phi_i(x,y) = (f_i'(x) g_i(y), f_i(x) g_i'(y))$$
and $a(\phi_i(x,y), \phi_j(x,y)) =$
$$\begin{aligned}
&= \int_{\hat{Q}} \left(f_i'(x) f_j'(x) g_i(y) g_j(y) + f_i(x) f_j(x) g_i'(y) g_j'(y) \right) d(x,y) \\
&= \int_{-1}^{1} f_i'(x) f_j'(x)\, dx \int_{-1}^{1} g_i(y) g_j(y)\, dy \\
&\quad + \int_{-1}^{1} f_i(x) f_j(x)\, dx \int_{-1}^{1} g_i'(y) g_j'(y))\, dy
\end{aligned}$$
- combination of one-dimensional bilinear forms!

The element system matrix for $p = 3$

The matrix can again be subdivided into block according to the different types of basis functions:

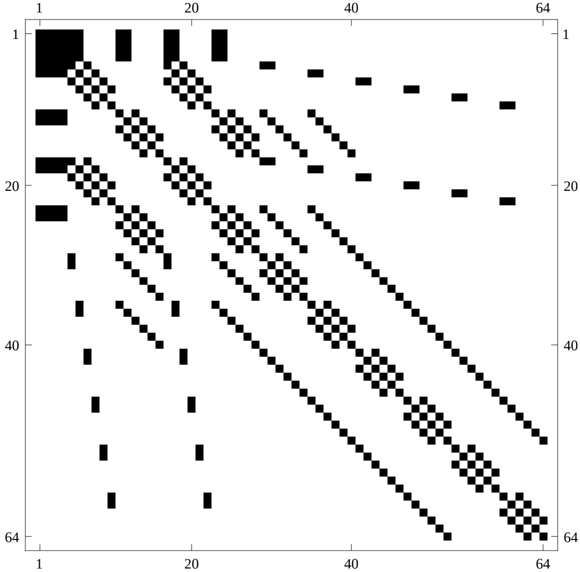
$$\begin{pmatrix} A_{VV} & A_{VE} & A_{VC} \\ A_{EV} & A_{EE} & A_{EC} \\ A_{CV} & A_{CE} & A_{CC} \end{pmatrix}$$

and if the basis using integrated Legendre polynomials is used, then the nonzero pattern is as follows...

The element system matrix for $p = 3$

$$
\left(\begin{array}{cccc|cccc|cccccccc}
* & * & * & * & * & * & * & * & * & * & * & * & & & & \\
* & * & * & * & * & * & * & * & * & * & * & * & & & & \\
* & * & * & * & * & * & * & * & * & * & * & * & & & & \\
* & * & * & * & * & * & * & * & * & * & * & * & & & & \\
\hline
* & * & * & * & * & & & & * & & & & * & * & & \\
* & * & * & * & & * & & & & * & & & & & * & * \\
* & * & * & * & & & * & & & & * & & * & & * & \\
* & * & * & * & & & & * & & & & * & & * & & * \\
\hline
* & * & * & * & * & & & & * & & & & * & * & & \\
* & * & * & * & & * & & & & * & & & & & * & * \\
* & * & * & * & & & * & & & & * & & * & & * & \\
* & * & * & * & & & & * & & & & * & & * & & * \\
 & & & & * & & * & & * & & * & & * & & & \\
 & & & & * & & & * & * & & & * & & * & & \\
 & & & & & * & * & & & * & * & & & & * & \\
 & & & & & * & & * & & * & & * & & & & *
\end{array}\right)
$$

The element system matrix for $p = 7$



Efficient computations: recurrence relations

- FE-basis on quadrilaterals defined as tensor products of 1D basis functions

Efficient computations: recurrence relations

- FE-basis on quadrilaterals defined as tensor products of 1D basis functions
- Hierarchic 1D basis functions defined using *orthogonal polynomials*

Efficient computations: recurrence relations

- FE-basis on quadrilaterals defined as tensor products of 1D basis functions
- Hierarchic 1D basis functions defined using *orthogonal polynomials*
- Orthogonal polynomials $\phi_n(x)$ satisfy *three term recurrences*

$$\phi_n(x) = (a_n x + b_n)\,\phi_{n-1}(x) + c_n \phi_{n-2}(x), \quad n \geq 1.$$

So do integrated Legendre polynomials!

Efficient computations: recurrence relations

- FE-basis on quadrilaterals defined as tensor products of 1D basis functions
- Hierarchic 1D basis functions defined using *orthogonal polynomials*
- Orthogonal polynomials $\phi_n(x)$ satisfy *three term recurrences*

$$\phi_n(x) = (a_n x + b_n)\,\phi_{n-1}(x) + c_n \phi_{n-2}(x), \quad n \geq 1.$$

  So do integrated Legendre polynomials!
- *Derivatives of orthogonal polynomials* also satisfy three term recurrences.

Efficient computations: recurrence relations

- FE-basis on quadrilaterals defined as tensor products of 1D basis functions
- Hierarchic 1D basis functions defined using *orthogonal polynomials*
- Orthogonal polynomials $\phi_n(x)$ satisfy *three term recurrences*

$$\phi_n(x) = (a_n x + b_n)\,\phi_{n-1}(x) + c_n \phi_{n-2}(x), \quad n \geq 1.$$

  So do integrated Legendre polynomials!
- *Derivatives of orthogonal polynomials* also satisfy three term recurrences.
- *Building blocks* of system matrices are integrals of the form

$$\int_{\hat{\mathcal{I}}} \phi_i(x) \psi_j(x)\, dx,$$

  where ϕ_i and ψ_j are orthogonal polynomials (satisfy three term recurrences).

Efficient computations: recurrence relations

- Let $\phi_n(x), \psi_n(x)$ be polynomials defined by the recurrences

$$\phi_n(x) = (a_n x + b_n)\,\phi_{n-1}(x) + c_n \phi_{n-2}(x),$$

and

$$\psi_n(x) = (\alpha_n x + \beta_n)\,\psi_{n-1}(x) + \gamma_n \psi_{n-2}(x).$$

Efficient computations: recurrence relations

- Let $\phi_n(x), \psi_n(x)$ be polynomials defined by the recurrences

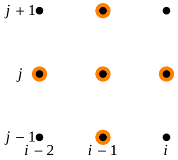
$$\phi_n(x) = (a_n x + b_n)\,\phi_{n-1}(x) + c_n \phi_{n-2}(x),$$

and

$$\psi_n(x) = (\alpha_n x + \beta_n)\,\psi_{n-1}(x) + \gamma_n \psi_{n-2}(x).$$

- Let $P_{i,j}(x) = \phi_i(x)\psi_j(x)$, then

$$P_{i,j}(x) = A_{i,j}P_{i-1,j+1}(x) + B_{i,j}P_{i-1,j}(x) + C_{i,j}P_{i-1,j-1}(x) + D_{i,j}P_{i-2,j}(x),$$



with

$$A_{i,j} = a_i/\alpha_{j+1}, \qquad B_{i,j} = (b_i\alpha_{j+1} - a_i\beta_{j+1})/\alpha_{j+1},$$
$$C_{i,j} = -a_i\gamma_{j+1}/\alpha_{j+1}, \qquad D_{i,j} = c_i.$$

Efficient computations: recurrence relations

- We have an x-free recurrence

$$P_{i,j}(x) = A_{i,j}P_{i-1,j+1}(x) + B_{i,j}P_{i-1,j}(x) + C_{i,j}P_{i-1,j-1}(x) + D_{i,j}P_{i-2,j}(x),$$

for $P_{i,j}(x) = \phi_i(x)\psi_j(x)$.

Efficient computations: recurrence relations

- We have an x-free recurrence

  $$P_{i,j}(x) = A_{i,j}P_{i-1,j+1}(x)+B_{i,j}P_{i-1,j}(x)+C_{i,j}P_{i-1,j-1}(x)+D_{i,j}P_{i-2,j}(x),$$

  for $P_{i,j}(x) = \phi_i(x)\psi_j(x)$.

- It can be multiplied by any function $f(x)$ and one may integrate over $\hat{\mathcal{I}}$ and the recurrence remains valid, i.e.,

  $$M_{i,j} = A_{i,j}M_{i-1,j+1} + B_{i,j}M_{i-1,j} + C_{i,j}M_{i-1,j-1} + D_{i,j}M_{i-2,j},$$

  for $M_{i,j} = \int_{\hat{\mathcal{I}}} f(x)\phi_i(x)\psi_j(x)\,dx$.

Efficient computations: sum factorization

- Usually the bilinear forms $a(u, v)$ building the system matrix A are evaluated using numerical integration:

$$a(\phi_{i,j}, \phi_{k,l}) = \int\int_{\hat{Q}} C(x, y)\phi_i(x)\psi_j(y)\phi_k(x)\psi_l(y)\, d(x, y)$$
$$\simeq \sum_{\alpha,\beta} w_\alpha w_\beta C(x_\alpha, y_\beta)\phi_i(x_\alpha)\psi_j(y_\beta)\phi_k(x_\alpha)\psi_l(y_\beta),$$

  where
  - $C(\cdot, \cdot)$ is a coefficient function (contains also transformation from the actual element in the mesh to the reference element),
  - w_γ are the quadrature weights and x_α, y_β are the quadrature points.

Efficient computations: sum factorization

- Usually the bilinear forms $a(u,v)$ building the system matrix A are evaluated using numerical integration:

$$a(\phi_{i,j},\phi_{k,l}) = \int\int_{\hat{Q}} C(x,y)\phi_i(x)\psi_j(y)\phi_k(x)\psi_l(y)\,d(x,y)$$
$$\simeq \sum_{\alpha,\beta} w_\alpha w_\beta C(x_\alpha,y_\beta)\phi_i(x_\alpha)\psi_j(y_\beta)\phi_k(x_\alpha)\psi_l(y_\beta),$$

  where
  - $C(\cdot,\cdot)$ is a coefficient function (contains also transformation from the actual element in the mesh to the reference element),
  - w_γ are the quadrature weights and x_α, y_β are the quadrature points.
- For the approximate solution of the linear system $A\underline{u} = \underline{f}$ an *iterative scheme* is used

$$\underline{u}^{(k+1)} = \underline{f} - A\underline{u}^{(k)},$$

  hence *not fast assemblance* of the matrix is needed, *but fast application*.

Efficient computation: sum factorization

Let's write the solution vector $\underline{u}$ with two indices, then

$$\left(A\underline{u}\right)_{k,l} \simeq \sum_{\alpha,\beta,i,j} w_\alpha w_\beta C(x_\alpha, y_\beta)\phi_i(x_\alpha)\psi_j(y_\beta)\phi_k(x_\alpha)\psi_l(y_\beta)\underline{u}_{i,j}$$

where each summation is $O(p)$ with maximal polynomial degree p.

Efficient computation: sum factorization

Let's write the solution vector $\underline{u}$ with two indices, then

$$\left(A\underline{u}\right)_{k,l} \simeq \sum_{\alpha,\beta,i,j} w_\alpha w_\beta C(x_\alpha, y_\beta)\phi_i(x_\alpha)\psi_j(y_\beta)\phi_k(x_\alpha)\psi_l(y_\beta)\underline{u}_{i,j}$$

where each summation is $O(p)$ with maximal polynomial degree p.

$$M^{(1)}_{\alpha,j} = \sum_i \phi_i(x_\alpha)\underline{u}_{i,j}$$

Efficient computation: sum factorization

Let's write the solution vector $\underline{u}$ with two indices, then

$$\left(A\underline{u}\right)_{k,l} \simeq \sum_{\alpha,\beta,i,j} w_\alpha w_\beta C(x_\alpha, y_\beta)\phi_i(x_\alpha)\psi_j(y_\beta)\phi_k(x_\alpha)\psi_l(y_\beta)\underline{u}_{i,j}$$

where each summation is $O(p)$ with maximal polynomial degree p.

$$M^{(1)}_{\alpha,j} = \sum_i \phi_i(x_\alpha)\underline{u}_{i,j}$$
$$M^{(2)}_{\alpha,\beta} = \sum_j M^{(1)}_{\alpha,j}\psi_j(y_\beta)$$

Efficient computation: sum factorization

Let's write the solution vector $\underline{u}$ with two indices, then

$$\left(A\underline{u}\right)_{k,l} \simeq \sum_{\alpha,\beta,i,j} w_\alpha w_\beta C(x_\alpha, y_\beta)\phi_i(x_\alpha)\psi_j(y_\beta)\phi_k(x_\alpha)\psi_l(y_\beta)\underline{u}_{i,j}$$

where each summation is $O(p)$ with maximal polynomial degree p.

$$M^{(1)}_{\alpha,j} = \sum_i \phi_i(x_\alpha)\underline{u}_{i,j}$$
$$M^{(2)}_{\alpha,\beta} = \sum_j M^{(1)}_{\alpha,j}\psi_j(y_\beta)$$
$$M^{(3)}_{\beta,k} = \sum_\alpha M^{(2)}_{\alpha,\beta} w_\alpha C(x_\alpha, y_\beta)\phi_k(x_\alpha)$$

Efficient computation: sum factorization

Let's write the solution vector $\underline{u}$ with two indices, then

$$\left(A\underline{u}\right)_{k,l} \simeq \sum_{\alpha,\beta,i,j} w_\alpha w_\beta C(x_\alpha, y_\beta)\phi_i(x_\alpha)\psi_j(y_\beta)\phi_k(x_\alpha)\psi_l(y_\beta)\underline{u}_{i,j}$$

where each summation is $O(p)$ with maximal polynomial degree p.

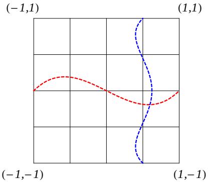
$$\begin{aligned}
M^{(1)}_{\alpha,j} &= \sum_i \phi_i(x_\alpha)\underline{u}_{i,j} \\
M^{(2)}_{\alpha,\beta} &= \sum_j M^{(1)}_{\alpha,j}\psi_j(y_\beta) \\
M^{(3)}_{\beta,k} &= \sum_\alpha M^{(2)}_{\alpha,\beta} w_\alpha C(x_\alpha, y_\beta)\phi_k(x_\alpha) \\
\left(A\underline{u}\right)_{k,l} &\simeq \sum_\beta M^{(3)}_{\beta,k} w_\beta \psi_l(y_\beta)
\end{aligned}$$

Hierarchic high order basis functions on triangles

On quadrilateral elements the basis functions were defined exploiting the tensor product structure:

$$\phi_{i,j}(x,y) = \varphi_i(x)\psi_j(y),$$

with $i,j \geq 0$ and $x,y \in [-1,1]$.

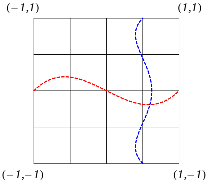


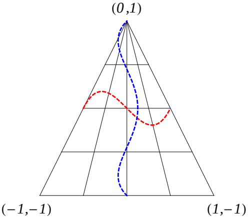
Hierarchic high order basis functions on triangles

On quadrilateral elements the basis functions were defined exploiting the tensor product structure:

$$\phi_{i,j}(x,y) = \varphi_i(x)\psi_j(y),$$

with $i, j \geq 0$ and $x, y \in [-1, 1]$.





The basis functions on the reference triangle $\hat{T}$ are defined by collapsing the quadrilateral to the triangle:

$$\phi_{i,j}(x,y) = \varphi_i\left(\frac{2x}{1-y}\right)\left(\frac{1-y}{2}\right)^i \psi_j(y).$$

Integral over reference triangle

Let $\phi_{i,j}(x,y) = \varphi_i\left(\frac{2x}{1-y}\right)\left(\frac{1-y}{2}\right)^i \psi_j(y)$, then by means of the substitution $z = \frac{2x}{1-y}$ we *decouple* the integrals:

$$\int_{\hat{T}} \phi_{i,j}(x,y)\phi_{k,l}(x,y)\,d(x,y) = \int_{-1}^{1} \varphi_i(x)\varphi_k(x)\,dx \times \int_{-1}^{1} \left(\tfrac{1-z}{2}\right)^{i+k+1} \psi_j(z)\psi_l(z)\,dz.$$

Integral over reference triangle

Let $\phi_{i,j}(x,y) = \varphi_i\left(\frac{2x}{1-y}\right)\left(\frac{1-y}{2}\right)^i \psi_j(y)$, then by means of the substitution $z = \frac{2x}{1-y}$ we *decouple* the integrals:

$$\int_{\hat{T}} \phi_{i,j}(x,y)\phi_{k,l}(x,y)\,d(x,y) = \int_{-1}^{1} P_i(x)P_k(x)\,dx \\ \times \int_{-1}^{1} \left(\tfrac{1-z}{2}\right)^{i+k+1} \psi_j(z)\psi_l(z)\,dz.$$

- Set $\varphi_i(x) = P_i(x)$ (Legendre polynomials)

Integral over reference triangle

Let $\phi_{i,j}(x,y) = \varphi_i\left(\frac{2x}{1-y}\right)\left(\frac{1-y}{2}\right)^i \psi_j(y)$, then by means of the substitution $z = \frac{2x}{1-y}$ we *decouple* the integrals:

$$\int_{\hat{T}} \phi_{i,j}(x,y)\phi_{k,l}(x,y)\,d(x,y) = g_i\delta_{i,k} \times \int_{-1}^{1} \left(\tfrac{1-z}{2}\right)^{2i+1} \psi_j(z)\psi_l(z)\,dz.$$

- Set $\varphi_i(x) = P_i(x)$ (Legendre polynomials)

Integral over reference triangle

Let $\phi_{i,j}(x,y) = \varphi_i\left(\frac{2x}{1-y}\right)\left(\frac{1-y}{2}\right)^i \psi_j(y)$, then by means of the substitution $z = \frac{2x}{1-y}$ we *decouple* the integrals:

$$\int_{\hat{T}} \phi_{i,j}(x,y)\phi_{k,l}(x,y)\,d(x,y) = g_i\delta_{i,k} \times \int_{-1}^{1} \left(\tfrac{1-z}{2}\right)^{2i+1}\psi_j(z)\psi_l(z)\,dz.$$

- Set $\varphi_i(x) = P_i(x)$ (Legendre polynomials)
- Set $\psi_j(y) = P_j^{(2i+1,0)}(y)$, the Jacobi polynomials orthogonal w.r.t. the weight function $\left(\frac{1-y}{2}\right)^{2i+1}$, i.e.,

$$\int_{-1}^{1} \left(\frac{1-y}{2}\right)^{2i+1} P_j^{(2i+1,0)}(y) P_l^{(2i+1,0)}(y)\,dy = h_{ij}\delta_{j,l}$$

Integral over reference triangle

Let $\phi_{i,j}(x,y) = \varphi_i\left(\frac{2x}{1-y}\right)\left(\frac{1-y}{2}\right)^i \psi_j(y)$, then by means of the substitution $z = \frac{2x}{1-y}$ we *decouple* the integrals:

$$\int_{\hat{T}} \phi_{i,j}(x,y)\phi_{k,l}(x,y)\, d(x,y) = g_i \delta_{i,k} \times \int_{-1}^{1} \left(\tfrac{1-z}{2}\right)^{2i+1} P_j^{(2i+1,0)}(z) P_l^{(2i+1,0)}(z)\, dz.$$

- Set $\varphi_i(x) = P_i(x)$ (Legendre polynomials)
- Set $\psi_j(y) = P_j^{(2i+1,0)}(y)$, the Jacobi polynomials orthogonal w.r.t. the weight function $\left(\frac{1-y}{2}\right)^{2i+1}$, i.e.,

$$\int_{-1}^{1} \left(\frac{1-y}{2}\right)^{2i+1} P_j^{(2i+1,0)}(y) P_l^{(2i+1,0)}(y)\, dy = h_{ij}\delta_{j,l}$$

Integral over reference triangle

Let $\phi_{i,j}(x,y) = \varphi_i\left(\frac{2x}{1-y}\right)\left(\frac{1-y}{2}\right)^i \psi_j(y)$, then by means of the substitution $z = \frac{2x}{1-y}$ we *decouple* the integrals:

$$\int_{\hat{T}} \phi_{i,j}(x,y)\phi_{k,l}(x,y)\, d(x,y) = g_i h_{ij} \delta_{i,k}\delta_{j,l}$$

Dubiner basis

- Set $\varphi_i(x) = P_i(x)$ (Legendre polynomials)
- Set $\psi_j(y) = P_j^{(2i+1,0)}(y)$, the Jacobi polynomials orthogonal w.r.t. the weight function $\left(\frac{1-y}{2}\right)^{2i+1}$, i.e.,

$$\int_{-1}^{1} \left(\frac{1-y}{2}\right)^{2i+1} P_j^{(2i+1,0)}(y) P_l^{(2i+1,0)}(y)\, dy = h_{ij}\delta_{j,l}$$

Sparsity optimized basis functions for simplices

- Dubiner: *L^2-orthogonal basis functions on triangles* (suitably chosen Jacobi polynomials)

Sparsity optimized basis functions for simplices

- Dubiner: *L^2-orthogonal basis functions on triangles* (suitably chosen Jacobi polynomials)
- Beuchler+Schöberl: *sparse system matrices for H^1 on triangles* (suitably chosen integrated Jacobi polynomials)

Sparsity optimized basis functions for simplices

- Dubiner: *L^2-orthogonal basis functions on triangles* (suitably chosen Jacobi polynomials)
- Beuchler+Schöberl: *sparse system matrices for H^1 on triangles* (suitably chosen integrated Jacobi polynomials)
- Beuchler + VP: extension of the basis construction in H^1 to *tetrahedra*

Sparsity optimized basis functions for simplices

- Dubiner: *L^2-orthogonal basis functions on triangles* (suitably chosen Jacobi polynomials)
- Beuchler+Schöberl: *sparse system matrices for H^1 on triangles* (suitably chosen integrated Jacobi polynomials)
- Beuchler + VP: extension of the basis construction in H^1 to *tetrahedra*
- Beuchler+VP+Zaglmayr: basis functions for $H(\text{div})$ *and* $H(\text{curl})$, where
  - construction principle follows Zaglmayr
  - exploit products of suitably chosen integrated Jacobi polynomials

Sparsity optimized basis functions for simplices

- Dubiner: *L^2-orthogonal basis functions on triangles* (suitably chosen Jacobi polynomials)
- Beuchler+Schöberl: *sparse system matrices for H^1 on triangles* (suitably chosen integrated Jacobi polynomials)
- Beuchler + VP: extension of the basis construction in H^1 to *tetrahedra*
- Beuchler+VP+Zaglmayr: basis functions for $H(\text{div})$ *and* $H(\text{curl})$, where
  - construction principle follows Zaglmayr
  - exploit products of suitably chosen integrated Jacobi polynomials
- Proof of sparsity in the latter three cases: exact evaluation of the integrals using symbolic computation

Jacobi and integrated Jacobi polynomials

- For $\alpha > -1$, $-1 \leq x \leq 1$ and $n \geq 0$ we denote by $P_n^{(\alpha,0)}(x)$ the nth *Jacobi polynomial* orthogonal w.r.t. the inner product

$$\langle f, g \rangle_{L^2_\alpha(-1,1)} \int_{-1}^{1} \left(\tfrac{1-x}{2}\right)^\alpha f(x) g(x)\, dx.$$

Jacobi and integrated Jacobi polynomials

- For $\alpha > -1$, $-1 \leq x \leq 1$ and $n \geq 0$ we denote by $P_n^{(\alpha,0)}(x)$ the nth *Jacobi polynomial* orthogonal w.r.t. the inner product

$$\langle f, g\rangle_{L^2_\alpha(-1,1)} \int_{-1}^{1} \left(\tfrac{1-x}{2}\right)^{\alpha} f(x)g(x)\, dx.$$

- We define the nth *integrated Jacobi polynomial* as

$$\hat{p}_n^{\alpha}(x) = \int_{-1}^{x} P_{n-1}^{(\alpha,0)}(s)\, ds.$$

Jacobi and integrated Jacobi polynomials

- For $\alpha > -1$, $-1 \leq x \leq 1$ and $n \geq 0$ we denote by $P_n^{(\alpha,0)}(x)$ the nth *Jacobi polynomial* orthogonal w.r.t. the inner product

$$\langle f, g \rangle_{L^2_\alpha(-1,1)} \int_{-1}^{1} \left(\tfrac{1-x}{2}\right)^\alpha f(x) g(x)\, dx.$$

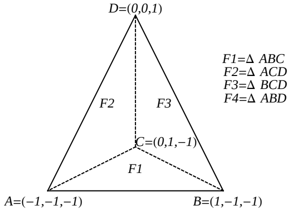
- We define the nth *integrated Jacobi polynomial* as

$$\hat{p}_n^\alpha(x) = \int_{-1}^{x} P_{n-1}^{(\alpha,0)}(s)\, ds.$$

- *Legendre polynomials* $P_n(x) = P_n^{(0,0)}(x)$

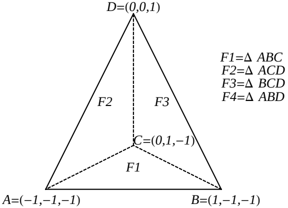
High order basis functions on tetrahedra

- vertex, edge, *face*, and cell based basis functions
- usually defined by collapsing a hexahedron to a tetrahedron



High order basis functions on tetrahedra

- vertex, edge, *face*, and cell based basis functions
- usually defined by collapsing a hexahedron to a tetrahedron



Example: Cell based basis function for H^1:

$$\phi_{ijk}(x,y,z) = \hat{p}_i^0\left(\frac{4x}{1-2y-z}\right)(1-2y-z)^i\,\hat{p}_j^{2i}\left(\frac{2y}{1-z}\right) \times (1-z)^j\,\hat{p}_k^{2i+2j}(z)$$

System matrix

- The FE-matrix that we consider is built from

$$a(u, v) = \int_{\hat{T}} uv \, d(x, y, z) + \int_{\hat{T}} (\nabla u)^T \mathcal{C} \nabla v \, d(x, y, z),$$

where $\mathcal{C}$ is a constant coefficient matrix.

System matrix

- The FE-matrix that we consider is built from

$$a(u, v) = \int_{\hat{T}} uv \, d(x, y, z) + \int_{\hat{T}} (\nabla u)^T \mathcal{C} \nabla v \, d(x, y, z),$$

  where $\mathcal{C}$ is a constant coefficient matrix.
- The results we obtain are of the form:

> Let $A = (a(\phi_{ijk}, \phi_{lmn}))_{ijklmn}$, then the entries of the system matrix are zero if $|i - l| > 2$ or $|i + j - l - m| > 3$ or $|i + j + k - l - m - n| > 2$.

System matrix

- The FE-matrix that we consider is built from

$$a(u, v) = \int_{\hat{T}} uv \, d(x, y, z) + \int_{\hat{T}} (\nabla u)^T \mathcal{C} \nabla v \, d(x, y, z),$$

  where $\mathcal{C}$ is a constant coefficient matrix.

- The results we obtain are of the form:

Let $A = (a(\phi_{ijk}, \phi_{lmn}))_{ijklmn}$, then the entries of the system matrix are zero if $\|i - l\| > 2$ or $\|i + j - l - m\| > 3$ or $\|i + j + k - l - m - n\| > 2$.

- The results are proven by *explicitly evaluating* the integrals using *rewriting*.

Example for evaluation

After decoupling the integrals for computing

$$\int_{\hat{T}} \phi_{ijk}(x,y,z)\phi_{lmn}(x,y,z)\,d(x,y,z)$$

are

Example for evaluation

After decoupling the integrals for computing

$$\int_{\hat{T}} \phi_{ijk}(x,y,z)\phi_{lmn}(x,y,z)\,d(x,y,z)$$

are

$$\int_{-1}^{1} \hat{p}_i^0(x)\hat{p}_l^0(x)\,dx \int_{-1}^{1} \hat{p}_j^{2i}(y)\hat{p}_m^{2l}(y)(1-y)^{i+l+1}\,dy \int_{-1}^{1} \hat{p}_k^{2i+2j}(z)\hat{p}_n^{2l+2m}(z)(1-z)^{i+j+l+m+2}\,dz$$

Example for evaluation

After decoupling the integrals for computing

$$\int_{\hat{T}} \phi_{ijk}(x,y,z)\phi_{lmn}(x,y,z)\, d(x,y,z)$$

are

$$\int_{-1}^{1} \hat{p}_i^0(x)\hat{p}_l^0(x)\, dx \int_{-1}^{1} \hat{p}_j^{2i}(y)\hat{p}_m^{2l}(y)(1-y)^{i+l+1}\, dy \int_{-1}^{1} \hat{p}_k^{2i+2j}(z)\hat{p}_n^{2l+2m}(z)(1-z)^{i+j+l+m+2}\, dz$$

- application of 2 rewrite rules and evaluation of 4 integrals

Example for evaluation

After decoupling the integrals for computing

$$\int_{\hat{T}} \phi_{ijk}(x,y,z)\phi_{lmn}(x,y,z)\,d(x,y,z)$$

are

$$\int_{-1}^{1} \hat{p}_i^0(x)\hat{p}_l^0(x)\,dx \int_{-1}^{1} \hat{p}_j^{2i}(y)\hat{p}_m^{2l}(y)(1-y)^{i+l+1}\,dy \int_{-1}^{1} \hat{p}_k^{2i+2j}(z)\hat{p}_n^{2l+2m}(z)(1-z)^{i+j+l+m+2}\,dz$$

- application of 2 rewrite rules and evaluation of 4 integrals
- application of 6 rewrite rules and evaluation of 40 integrals

Example for evaluation

After decoupling the integrals for computing

$$\int_{\hat{T}} \phi_{ijk}(x,y,z)\phi_{lmn}(x,y,z)\,d(x,y,z)$$

are

$$\int_{-1}^{1} \hat{p}_i^0(x)\hat{p}_l^0(x)\,dx \int_{-1}^{1} \hat{p}_j^{2i}(y)\hat{p}_m^{2l}(y)(1-y)^{i+l+1}\,dy \int_{-1}^{1} \hat{p}_k^{2i+2j}(z)\hat{p}_n^{2l+2m}(z)(1-z)^{i+j+l+m+2}\,dz$$

- application of 2 rewrite rules and evaluation of 4 integrals
- application of 6 rewrite rules and evaluation of 40 integrals
- application of 14 rewrite rules and evaluation of 147 integrals

Example for evaluation

After decoupling the integrals for computing

$$\int_{\hat{T}} \phi_{ijk}(x,y,z)\phi_{lmn}(x,y,z)\, d(x,y,z)$$

are

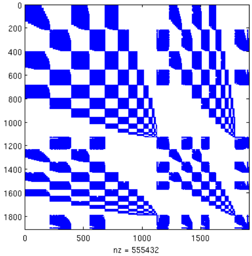
$$\int_{-1}^{1} \hat{p}_i^0(x)\hat{p}_l^0(x)\, dx \int_{-1}^{1} \hat{p}_j^{2i}(y)\hat{p}_m^{2l}(y)(1-y)^{i+l+1}\, dy \int_{-1}^{1} \hat{p}_k^{2i+2j}(z)\hat{p}_n^{2l+2m}(z)(1-z)^{i+j+l+m+2}\, dz$$

- application of 2 rewrite rules and evaluation of 4 integrals
- application of 6 rewrite rules and evaluation of 40 integrals
- application of 14 rewrite rules and evaluation of 147 integrals

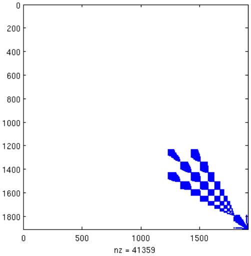
In total 189 nonzero matrix entries

Sparsity pattern for $H(\mathrm{div})$, $p = 15$

- using standard Legendre-type polynomials



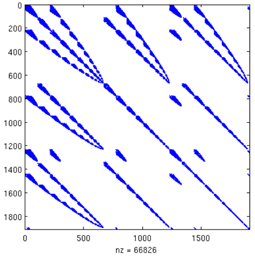
element mass matrix



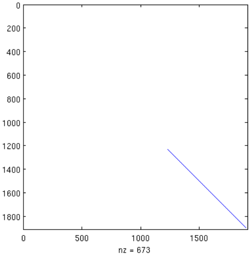
element stiffness matrix

Sparsity pattern for $H(\text{div})$, $p = 15$

- using the Jacobi-type polynomials with optimized parameters



element mass matrix



element stiffness matrix

Fast assembling of the system matrices

- Lehrenfeld+Koutschan+Schöberl: using mixed relations of the form

$$\sum_{r=0}^{R} c_r(i,j)\phi_{i+r_1,j+r_2}(x,y) = \sum_{s=0}^{S} D_x\phi_{i+s_1,j+s_2}(x,y)$$

- Combining recurrences for Jacobi polynomials $\hat{p}_n^{\alpha}(x)$, $p_n^{\alpha}(x)$ and sum factorization techniques [Beuchler et al]

- Computing recurrences for the explicit matrix entries

Local Fourier analysis (Takacs + VP)

- standard tool to analyze the *convergence* behaviour of a numerical method

Local Fourier analysis (Takacs + VP)

- standard tool to analyze the *convergence* behaviour of a numerical method
- *Model problem*: optimization problem constrained to a partial differential equation
  Find a state y and a control u that minimize

$$J(y,u) := \frac{1}{2}\|y - y_D\|^2_{L^2(\Omega)} + \frac{\alpha}{2}\|u\|^2_{L^2(\Omega)},$$

  subject to $-\Delta y = u$ in the domain Ω.

Local Fourier analysis (Takacs + VP)

- standard tool to analyze the *convergence* behaviour of a numerical method
- *Model problem*: optimization problem constrained to a partial differential equation
  Find a state y and a control u that minimize

  $$J(y,u) := \frac{1}{2}\|y - y_D\|^2_{L^2(\Omega)} + \frac{\alpha}{2}\|u\|^2_{L^2(\Omega)},$$

  subject to $-\Delta y = u$ in the domain Ω.
- solution method: *Finite Element Method (FEM)* with a *multigrid-solver* (which ultimately means solving a large scale linear system)

Local Fourier analysis (Takacs + VP)

- standard tool to analyze the *convergence* behaviour of a numerical method
- *Model problem*: optimization problem constrained to a partial differential equation
  Find a state y and a control u that minimize

$$J(y, u) := \frac{1}{2}\|y - y_D\|^2_{L^2(\Omega)} + \frac{\alpha}{2}\|u\|^2_{L^2(\Omega)},$$

  subject to $-\Delta y = u$ in the domain Ω.
- solution method: *Finite Element Method (FEM)* with a *multigrid-solver* (which ultimately means solving a large scale linear system)
- *robust* with respect to parameters such as mesh-size and regularization parameters

Local Fourier Analysis

- Given: Iterative procedure of the form

$$x^{(k+1)} = Ax^{(k)}$$

the *convergence rate* is related to the *matrix norm* of A which can be estimated by the spectral radius, i.e., the *largest eigenvalue in absolute value*

Local Fourier Analysis

- Given: Iterative procedure of the form

$$x^{(k+1)} = Ax^{(k)}$$

  the *convergence rate* is related to the *matrix norm* of A which can be estimated by the spectral radius, i.e., the *largest eigenvalue in absolute value*
- typically rates are obtained only by *numerical interpolation*

Local Fourier Analysis

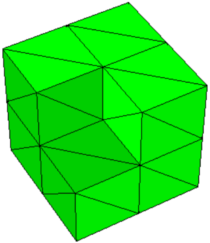
- Given: Iterative procedure of the form

$$x^{(k+1)} = Ax^{(k)}$$

  the *convergence rate* is related to the *matrix norm* of A which can be estimated by the spectral radius, i.e., the *largest eigenvalue in absolute value*
- typically rates are obtained only by *numerical interpolation*
- *symbolic* local Fourier analysis: *exact bounds*

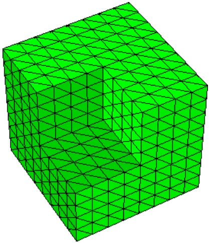
FEM and multigrid

- Multigrid methods operate on two (or more) grids



FEM and multigrid

- Multigrid methods operate on two (or more) grids



FEM and multigrid

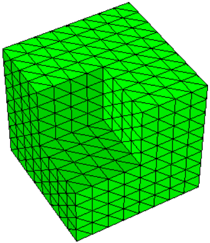
- Multigrid methods operate on two (or more) grids
- One step in a multigrid method consists of
  - (pre)smoothing steps

    $\downarrow$ restriction $\downarrow$

  - coarse grid correction

    $\downarrow$ prolongation $\downarrow$

  - (post)smoothing steps



Bound the convergence rate (2D)

Given: matrix $A(q, c_1, c_2, \eta) \in \mathbb{R}^{8\times 8}$ with $0 < q < 1$ and

$$(c_1, c_2, \eta) \in \Omega = \{(c_1, c_2, \eta) \mid 0 \leq c_1 \leq c_2 < 1 \;\wedge\; \eta > 0\},$$

where $c_i = \cos(\theta_i)$ for some frequencies θ_i and $\eta = h^4/\alpha$ with mesh-size h and regularization parameter α.

Find: bound $B(q)$ for the maximal eigenvalue $\lambda_{max}(q, c_1, c_2, \eta)$ of A over $(0, 1)^2 \times \Omega$.

The Matrix A

$$\begin{pmatrix} \mathbf{A_{1,1}} & 0 & A_{1,3} & A_{1,4} & A_{1,5} & A_{1,6} & A_{1,7} & A_{1,8} \\ 0 & A_{2,2} & A_{2,3} & A_{2,4} & A_{2,5} & A_{2,6} & A_{2,7} & A_{2,8} \\ A_{3,1} & A_{3,2} & A_{3,3} & 0 & A_{3,5} & A_{3,6} & A_{3,7} & A_{3,8} \\ A_{4,1} & A_{4,2} & 0 & A_{4,4} & A_{4,5} & A_{4,6} & A_{4,7} & A_{4,8} \\ A_{5,1} & A_{5,2} & A_{5,3} & A_{5,4} & A_{5,5} & 0 & A_{5,7} & A_{5,8} \\ A_{6,1} & A_{6,2} & A_{6,3} & A_{6,4} & 0 & A_{6,6} & A_{6,7} & A_{6,8} \\ A_{7,1} & A_{7,2} & A_{7,3} & A_{7,4} & A_{7,5} & A_{7,6} & A_{7,7} & 0 \\ A_{8,1} & A_{8,2} & A_{8,3} & A_{8,4} & A_{8,5} & A_{8,6} & 0 & A_{8,8} \end{pmatrix}$$

Common denominator of the matrix entries:

$$\begin{aligned} D =& 256 \Big(16c_2^4c_1^4\eta + 16c_2^2c_1^4\eta + 4c_1^4\eta + 16c_2^4c_1^2\eta + 16c_2^2c_1^2\eta + 4c_1^2\eta \\ &+ 4c_2^4\eta + 4c_2^2\eta + 144c_2^4c_1^4 - 72c_2^2c_1^4 + 9c_1^4 - 72c_2^4c_1^2 - 126c_2^2c_1^2 \\ &+ 36c_1^2 + 9c_2^4 + 36c_2^2 + \eta + 36 \Big) \end{aligned}$$

The Matrix A

$$\begin{pmatrix}
\mathbf{A_{1,1}} & 0 & A_{1,3} & A_{1,4} & A_{1,5} & A_{1,6} & A_{1,7} & A_{1,8} \\
0 & A_{2,2} & A_{2,3} & A_{2,4} & A_{2,5} & A_{2,6} & A_{2,7} & A_{2,8} \\
A_{3,1} & A_{3,2} & A_{3,3} & 0 & A_{3,5} & A_{3,6} & A_{3,7} & A_{3,8} \\
A_{4,1} & A_{4,2} & 0 & A_{4,4} & A_{4,5} & A_{4,6} & A_{4,7} & A_{4,8} \\
A_{5,1} & A_{5,2} & A_{5,3} & A_{5,4} & A_{5,5} & 0 & A_{5,7} & A_{5,8} \\
A_{6,1} & A_{6,2} & A_{6,3} & A_{6,4} & 0 & A_{6,6} & A_{6,7} & A_{6,8} \\
A_{7,1} & A_{7,2} & A_{7,3} & A_{7,4} & A_{7,5} & A_{7,6} & A_{7,7} & 0 \\
A_{8,1} & A_{8,2} & A_{8,3} & A_{8,4} & A_{8,5} & A_{8,6} & 0 & A_{8,8}
\end{pmatrix}$$

Common denominator of the matrix entries:

$$\begin{aligned}
D = & 256\Big(16c_2^4c_1^4\eta + 16c_2^2c_1^4\eta + 4c_1^4\eta + 16c_2^4c_1^2\eta + 16c_2^2c_1^2\eta + 4c_1^2\eta \\
& + 4c_2^4\eta + 4c_2^2\eta + 144c_2^4c_1^4 - 72c_2^2c_1^4 + 9c_1^4 - 72c_2^4c_1^2 - 126c_2^2c_1^2 \\
& + 36c_1^2 + 9c_2^4 + 36c_2^2 + \eta + 36\Big)
\end{aligned}$$

The Numerator of $A_{1,1}$

$$
\begin{gathered}
432q^2c_2^6c_1^6 + 3q^2\eta c_2^6c_1^6 + \eta c_2^6c_1^6 + 144c_2^6c_1^6 + 1008q^2c_2^5c_1^6 + 16q^2\eta c_2^5c_1^6 + 8\eta c_2^5c_1^6 + 720c_2^5c_1^6 + 972q^2c_2^4c_1^6 + \\
30q^2\eta c_2^4c_1^6 + 26\eta c_2^4c_1^6 + 1476c_2^4c_1^6 + 1008q^2c_2^3c_1^6 + 28q^2\eta c_2^3c_1^6 + 44\eta c_2^3c_1^6 + 1584c_2^3c_1^6 + 108q^2c_1^6 + 1080q^2c_2^2c_1^6 + \\
27q^2\eta c_2^2c_1^6 + 41\eta c_2^2c_1^6 + 936c_2^2c_1^6 + 12q^2\eta c_1^6 + 4\eta c_1^6 + 576q^2c_2c_1^6 + 28q^2\eta c_2c_1^6 + 20\eta c_2c_1^6 + 288c_2c_1^6 + 36c_1^6 + \\
1008q^2c_2^6c_1^5 + 16q^2\eta c_2^6c_1^5 + 8\eta c_2^6c_1^5 + 720c_2^6c_1^5 - 360q^2c_2^5c_1^5 + 80q^2\eta c_2^5c_1^5 - 64\eta c_2^5c_1^5 - 1656c_2^5c_1^5 - 3600q^2c_2^4c_1^5 + \\
128q^2\eta c_2^4c_1^5 - 304\eta c_2^4c_1^5 - 7056c_2^4c_1^5 - 2736q^2c_2^3c_1^5 + 80q^2\eta c_2^3c_1^5 - 352\eta c_2^3c_1^5 - 5328c_2^3c_1^5 - 720q^2c_1^5 - \\
1872q^2c_2^2c_1^5 + 80q^2\eta c_2^2c_1^5 - 184\eta c_2^2c_1^5 + 720c_2^2c_1^5 + 64q^2\eta c_1^5 - 96\eta c_1^5 - 2088q^2c_2c_1^5 + 128q^2\eta c_2c_1^5 - 160\eta c_2c_1^5 + \\
1800c_2c_1^5 + 432c_1^5 + 972q^2c_2^6c_1^4 + 30q^2\eta c_2^6c_1^4 + 26\eta c_2^6c_1^4 + 1476c_2^6c_1^4 - 3600q^2c_2^5c_1^4 + 128q^2\eta c_2^5c_1^4 - 304\eta c_2^5c_1^4 - \\
7056c_2^5c_1^4 - 5616q^2c_2^4c_1^4 + 108q^2\eta c_2^4c_1^4 + 2724\eta c_2^4c_1^4 + 21168c_2^4c_1^4 + 1584q^2c_2^3c_1^4 - 136q^2\eta c_2^3c_1^4 - 1672\eta c_2^3c_1^4 - \\
3600c_2^3c_1^4 + 648q^2c_1^4 + 1404q^2c_2^2c_1^4 - 114q^2\eta c_2^2c_1^4 + 3114\eta c_2^2c_1^4 - 15660c_2^2c_1^4 + 120q^2\eta c_1^4 + 616\eta c_1^4 - 576q^2c_2c_1^4 + \\
152q^2\eta c_2c_1^4 - 760\eta c_2c_1^4 - 2304c_2c_1^4 + 792c_1^4 + 1008q^2c_2^6c_1^3 + 28q^2\eta c_2^6c_1^3 + 44\eta c_2^6c_1^3 + 1584c_2^6c_1^3 - 2736q^2c_2^5c_1^3 + \\
80q^2\eta c_2^5c_1^3 - 352\eta c_2^5c_1^3 - 5328c_2^5c_1^3 + 1584q^2c_2^4c_1^3 - 136q^2\eta c_2^4c_1^3 - 1672\eta c_2^4c_1^3 - 3600c_2^4c_1^3 + 12384q^2c_2^3c_1^3 - \\
640q^2\eta c_2^3c_1^3 - 1936\eta c_2^3c_1^3 + 17568c_2^3c_1^3 + 1872q^2c_1^3 + 5904q^2c_2^2c_1^3 - 580q^2\eta c_2^2c_1^3 - 1012\eta c_2^2c_1^3 + 14544c_2^2c_1^3 + \\
112q^2\eta c_1^3 - 528\eta c_1^3 + 720q^2c_2c_1^3 - 16q^2\eta c_2c_1^3 - 880\eta c_2c_1^3 - 1872c_2c_1^3 - 2160c_1^3 + 1080q^2c_2^6c_1^2 + 27q^2\eta c_2^6c_1^2 + \\
41\eta c_2^6c_1^2 + 936c_2^6c_1^2 - 1872q^2c_2^5c_1^2 + 80q^2\eta c_2^5c_1^2 - 184\eta c_2^5c_1^2 + 720c_2^5c_1^2 + 1404q^2c_2^4c_1^2 - 114q^2\eta c_2^4c_1^2 + \\
3114\eta c_2^4c_1^2 - 15660c_2^4c_1^2 + 5904q^2c_2^3c_1^2 - 580q^2\eta c_2^3c_1^2 - 1012\eta c_2^3c_1^2 + 14544c_2^3c_1^2 + 108q^2c_1^2 - 5184q^2c_2^2c_1^2 - \\
525q^2\eta c_2^2c_1^2 + 3729\eta c_2^2c_1^2 - 15552c_2^2c_1^2 + 108q^2\eta c_1^2 + 676\eta c_1^2 - 6624q^2c_2c_1^2 - 4q^2\eta c_2c_1^2 - 460\eta c_2c_1^2 + \\
2880c_2c_1^2 + 6948c_1^2 + 576q^2c_2^6c_1 + 28q^2\eta c_2^6c_1 + 20\eta c_2^6c_1 + 288c_2^6c_1 - 2088q^2c_2^5c_1 + 128q^2\eta c_2^5c_1 - 160\eta c_2^5c_1 + \\
1800c_2^5c_1 - 576q^2c_2^4c_1 + 152q^2\eta c_2^4c_1 - 760\eta c_2^4c_1 - 2304c_2^4c_1 + 720q^2c_2^3c_1 - 16q^2\eta c_2^3c_1 - 880\eta c_2^3c_1 - \\
1872c_2^3c_1 + 1440q^2c_1 - 6624q^2c_2^2c_1 - 4q^2\eta c_2^2c_1 - 460\eta c_2^2c_1 + 2880c_2^2c_1 + 112q^2\eta c_1 - 240\eta c_1 - 3816q^2c_2c_1 + \\
176q^2\eta c_2c_1 - 400\eta c_2c_1 - 5112c_2c_1 - 6048c_1 + 108q^2c_2^6 + 12q^2\eta c_2^6 + 4\eta c_2^6 + 36c_2^6 - 720q^2c_2^5 + 64q^2\eta c_2^5 - \\
96\eta c_2^5 + 432c_2^5 + 648q^2c_2^4 + 120q^2\eta c_2^4 + 616\eta c_2^4 + 792c_2^4 + 1872q^2c_2^3 + 112q^2\eta c_2^3 - 528\eta c_2^3 - 2160c_2^3 + 1728q^2 + \\
108q^2c_2^2 + 108q^2\eta c_2^2 + 676\eta c_2^2 + 6948c_2^2 + 48q^2\eta + 144\eta + 1440q^2c_2 + 112q^2\eta c_2 - 240\eta c_2 - 6048c_2 + 5184
\end{gathered}
$$

Computational Issues

1. How to find the eigenvalues?
2. How to find the bound?
3. How to prove the bound?

Computational Issues

1. How to find the eigenvalues?
   - *symbolic* interpolation in q
2. How to find the bound?
3. How to prove the bound?

Computational Issues

1. How to find the eigenvalues?
   - *symbolic* interpolation in q
2. How to find the bound?
   - consider the boundary of the domain to obtain a plausible *guess* for the bound using *CAD*
3. How to prove the bound?

Computational Issues

1. How to find the eigenvalues?
   - *symbolic* interpolation in q
2. How to find the bound?
   - consider the boundary of the domain to obtain a plausible *guess* for the bound using *CAD*
3. How to prove the bound?
   - use *CAD*

Computational Issues

1. How to find the eigenvalues?
   - *symbolic* interpolation in q
2. How to find the bound?
   - consider the boundary of the domain to obtain a plausible *guess* for the bound using *CAD*
3. How to prove the bound?
   - use *CAD*
   - necessary to split into *subtasks* and consider *sufficient conditions*

The Eigenvalues

Using interpolation on the characteristic polynomial of the matrix we find that the eigenvalues are

$$\lambda_1 = 0, \quad \lambda_2 = q^4, \quad {}_3\lambda_4 = \frac{1}{D}\left(e(q_2) \pm \sqrt{d(q_2)}\right),$$

each of multiplicity 2 with $q_2 = q^2$ and e, d polynomials in c_1, c_2, η and q_2.

The Eigenvalues

Using interpolation on the characteristic polynomial of the matrix we find that the eigenvalues are

$$\lambda_1 = 0, \quad \lambda_2 = q^4, \quad {}_3\lambda_4 = \frac{1}{D}\left(e(q_2) \pm \sqrt{d(q_2)}\right),$$

each of multiplicity 2 with $q_2 = q^2$ and e, d polynomials in c_1, c_2, η and q_2.

The largest eigenvalue is

$$\lambda_{max} = \frac{1}{D}\left(e(q_2) + \sqrt{d(q_2)}\right).$$

The Eigenvalues

Using interpolation on the characteristic polynomial of the matrix we find that the eigenvalues are

$$\lambda_1 = 0, \quad \lambda_2 = q^4, \quad {}_3\lambda_4 = \frac{1}{D}\left(e(q_2) \pm \sqrt{d(q_2)}\right),$$

each of multiplicity 2 with $q_2 = q^2$ and e, d polynomials in c_1, c_2, η and q_2.

The largest eigenvalue is

$$\lambda_{max} = \frac{1}{D}\left(e(q_2) + \sqrt{d(q_2)}\right).$$

The Polynomial $e(q_2)$

$$\eta c_2^6 c_1^6 + 144c_2^6 c_1^6 + 8\eta c_2^5 c_1^6 + 720c_2^5 c_1^6 + 26\eta c_2^4 c_1^6 + 1476c_2^4 c_1^6 + 44\eta c_2^3 c_1^6 + 1584c_2^3 c_1^6 + 41\eta c_2^2 c_1^6 + 936c_2^2 c_1^6 +$$
$$9\eta c_2^6 q_2^2 c_1^6 + 1296c_2^6 q_2^2 c_1^6 - 24\eta c_2^5 q_2^2 c_1^6 - 2160c_2^5 q_2^2 c_1^6 + 138\eta c_2^4 q_2^2 c_1^6 + 6372c_2^4 q_2^2 c_1^6 - 132\eta c_2^3 q_2^2 c_1^6 - 4752c_2^3 q_2^2 c_1^6 +$$
$$177\eta c_2^2 q_2^2 c_1^6 + 4968c_2^2 q_2^2 c_1^6 + 36\eta q_2^2 c_1^6 - 60\eta c_2 q_2^2 c_1^6 - 864c_2 q_2^2 c_1^6 + 324q_2^2 c_1^6 + 4\eta c_1^6 + 20\eta c_2 c_1^6 + 288c_2 c_1^6 +$$
$$6\eta c_2^6 q_2 c_1^6 + 864c_2^6 q_2 c_1^6 + 16\eta c_2^5 q_2 c_1^6 + 1440c_2^5 q_2 c_1^6 + 60\eta c_2^4 q_2 c_1^6 + 1944c_2^4 q_2 c_1^6 + 88\eta c_2^3 q_2 c_1^6 + 3168c_2^3 q_2 c_1^6 +$$
$$54\eta c_2^2 q_2 c_1^6 + 2160c_2^2 q_2 c_1^6 + 24\eta q_2 c_1^6 + 40\eta c_2 q_2 c_1^6 + 576c_2 q_2 c_1^6 + 216q_2 c_1^6 + 36c_1^6 + 8\eta c_2^6 c_1^5 + 720c_2^6 c_1^5 - 64\eta c_2^5 c_1^5 -$$
$$1656c_2^5 c_1^5 - 304\eta c_2^4 c_1^5 - 7056c_2^4 c_1^5 - 352\eta c_2^3 c_1^5 - 5328c_2^3 c_1^5 - 184\eta c_2^2 c_1^5 + 720c_2^2 c_1^5 - 24\eta c_2^6 q_2^2 c_1^5 - 2160c_2^6 q_2^2 c_1^5 +$$
$$32\eta c_2^5 q_2^2 c_1^5 + 2664c_2^5 q_2^2 c_1^5 + 272\eta c_2^4 q_2^2 c_1^5 + 10224c_2^4 q_2^2 c_1^5 + 416\eta c_2^3 q_2^2 c_1^5 + 3312c_2^3 q_2^2 c_1^5 + 296\eta c_2^2 q_2^2 c_1^5 - 2736c_2^2 q_2^2 c_1^5 +$$
$$32\eta q_2^2 c_1^5 + 128\eta c_2 q_2^2 c_1^5 - 792c_2 q_2^2 c_1^5 - 144q_2^2 c_1^5 - 96\eta c_1^5 - 160\eta c_2 c_1^5 + 1800c_2 c_1^5 + 16\eta c_2^6 q_2 c_1^5 + 1440c_2^6 q_2 c_1^5 +$$
$$32\eta c_2^5 q_2 c_1^5 - 1008c_2^5 q_2 c_1^5 + 32\eta c_2^4 q_2 c_1^5 - 3168c_2^4 q_2 c_1^5 - 64\eta c_2^3 q_2 c_1^5 + 2016c_2^3 q_2 c_1^5 - 112\eta c_2^2 q_2 c_1^5 + 2016c_2^2 q_2 c_1^5 +$$
$$64\eta q_2 c_1^5 + 32\eta c_2 q_2 c_1^5 - 1008c_2 q_2 c_1^5 - 288q_2 c_1^5 + 432c_1^5 + 26\eta c_2^6 c_1^4 + 1476c_2^6 c_1^4 - 304\eta c_2^5 c_1^4 - 7056c_2^5 c_1^4 +$$
$$2724\eta c_2^4 c_1^4 + 21168c_2^4 c_1^4 - 1672\eta c_2^3 c_1^4 - 3600c_2^3 c_1^4 + 3114\eta c_2^2 c_1^4 - 15660c_2^2 c_1^4 + 138\eta c_2^6 q_2^2 c_1^4 + 6372c_2^6 q_2^2 c_1^4 +$$
$$272\eta c_2^5 q_2^2 c_1^4 + 10224c_2^5 q_2^2 c_1^4 + 4292\eta c_2^4 q_2^2 c_1^4 + 15408c_2^4 q_2^2 c_1^4 + 1496\eta c_2^3 q_2^2 c_1^4 + 8496c_2^3 q_2^2 c_1^4 + 5018\eta c_2^2 q_2^2 c_1^4 + \dots$$

The Polynomial $e(q_2)$

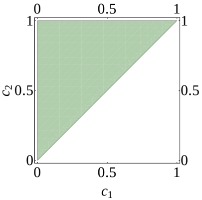
$$\cdots - 25740c_2^2q_2^2c_1^4 + 1064\eta q_2^2c_1^4 + 680\eta c_2q_2^2c_1^4 - 576c_2q_2^2c_1^4 + 1368q_2^2c_1^4 + 616\eta c_1^4 - 760\eta c_2c_1^4 - 2304c_2c_1^4 +$$
$$60\eta c_2^6q_2c_1^4 + 1944c_2^6q_2c_1^4 + 32\eta c_2^5q_2c_1^4 - 3168c_2^5q_2c_1^4 + 216\eta c_2^4q_2c_1^4 - 11232c_2^4q_2c_1^4 + 176\eta c_2^3q_2c_1^4 -$$
$$4896c_2^3q_2c_1^4 - 228\eta c_2^2q_2c_1^4 + 2808c_2^2q_2c_1^4 + 240\eta q_2c_1^4 + 80\eta c_2q_2c_1^4 + 2880c_2q_2c_1^4 + 1296q_2c_1^4 + 792c_1^4 +$$
$$44\eta c_2^6c_1^3 + 1584c_2^6c_1^3 - 352\eta c_2^5c_1^3 - 5328c_2^5c_1^3 - 1672\eta c_2^4c_1^3 - 3600c_2^4c_1^3 - 1936\eta c_2^3c_1^3 + 17568c_2^3c_1^3 -$$
$$1012\eta c_2^2c_1^3 + 14544c_2^2c_1^3 - 132\eta c_2^6q_2^2c_1^3 - 4752c_2^6q_2^2c_1^3 + 416\eta c_2^5q_2^2c_1^3 + 3312c_2^5q_2^2c_1^3 + 1496\eta c_2^4q_2^2c_1^3 +$$
$$8496c_2^4q_2^2c_1^3 + 1808\eta c_2^3q_2^2c_1^3 - 13536c_2^3q_2^2c_1^3 + 1628\eta c_2^2q_2^2c_1^3 - 14832c_2^2q_2^2c_1^3 + 176\eta q_2^2c_1^3 + 944\eta c_2q_2^2c_1^3 -$$
$$144c_2q_2^2c_1^3 + 720q_2^2c_1^3 - 528\eta c_1^3 - 880\eta c_2c_1^3 - 1872c_2c_1^3 + 88\eta c_2^6q_2c_1^3 + 3168c_2^6q_2c_1^3 - 64\eta c_2^5q_2c_1^3 + 2016c_2^5q_2c_1^3 +$$
$$176\eta c_2^4q_2c_1^3 - 4896c_2^4q_2c_1^3 + 128\eta c_2^3q_2c_1^3 - 4032c_2^3q_2c_1^3 - 616\eta c_2^2q_2c_1^3 + 288c_2^2q_2c_1^3 + 352\eta q_2c_1^3 - 64\eta c_2q_2c_1^3 +$$
$$2016c_2q_2c_1^3 + 1440q_2c_1^3 - 2160c_1^3 + 41\eta c_2^6c_1^2 + 936c_2^6c_1^2 - 184\eta c_2^5c_1^2 + 720c_2^5c_1^2 + 3114\eta c_2^4c_1^2 - 15660c_2^4c_1^2 -$$
$$1012\eta c_2^3c_1^2 + 14544c_2^3c_1^2 + 3729\eta c_2^2c_1^2 - 15552c_2^2c_1^2 + 177\eta c_2^6q_2^2c_1^2 + 4968c_2^6q_2^2c_1^2 + 296\eta c_2^5q_2^2c_1^2 - 2736c_2^5q_2^2c_1^2 +$$
$$5018\eta c_2^4q_2^2c_1^2 - 25740c_2^4q_2^2c_1^2 + 1628\eta c_2^3q_2^2c_1^2 - 14832c_2^3q_2^2c_1^2 + 6041\eta c_2^2q_2^2c_1^2 - 24768c_2^2q_2^2c_1^2 + 1220\eta q_2^2c_1^2 + \ldots$$

The Polynomial $e(q_2)$

$$\cdots + 740\eta c_2 q_2^2 c_1^2 + 4608 c_2 q_2^2 c_1^2 + 11844 q_2^2 c_1^2 + 676\eta c_1^2 - 460\eta c_2 c_1^2 + 2880 c_2 c_1^2 + 54\eta c_2^6 q_2 c_1^2 + 2160 c_2^6 q_2 c_1^2 -$$
$$112\eta c_2^5 q_2 c_1^2 + 2016 c_2^5 q_2 c_1^2 - 228\eta c_2^4 q_2 c_1^2 + 2808 c_2^4 q_2 c_1^2 - 616\eta c_2^3 q_2 c_1^2 + 288 c_2^3 q_2 c_1^2 - 1050\eta c_2^2 q_2 c_1^2 -$$
$$10368 c_2^2 q_2 c_1^2 + 216\eta q_2 c_1^2 - 280\eta c_2 q_2 c_1^2 - 7488 c_2 q_2 c_1^2 + 216 q_2 c_1^2 + 6948 c_1^2 + 20\eta c_2^6 c_1 + 288 c_2^6 c_1 -$$
$$160\eta c_2^5 c_1 + 1800 c_2^5 c_1 - 760\eta c_2^4 c_1 - 2304 c_2^4 c_1 - 880\eta c_2^3 c_1 - 1872 c_2^3 c_1 - 460\eta c_2^2 c_1 + 2880 c_2^2 c_1 - 60\eta c_2^6 q_2^2 c_1 -$$
$$864 c_2^6 q_2^2 c_1 + 128\eta c_2^5 q_2^2 c_1 - 792 c_2^5 q_2^2 c_1 + 680\eta c_2^4 q_2^2 c_1 - 576 c_2^4 q_2^2 c_1 + 944\eta c_2^3 q_2^2 c_1 - 144 c_2^3 q_2^2 c_1 + 740\eta c_2^2 q_2^2 c_1 +$$
$$4608 c_2^2 q_2^2 c_1 + 80\eta q_2^2 c_1 + 368\eta c_2 q_2^2 c_1 + 6120 c_2 q_2^2 c_1 + 2016 q_2^2 c_1 - 240\eta c_1 - 400\eta c_2 c_1 - 5112 c_2 c_1 +$$
$$40\eta c_2^6 q_2 c_1 + 576 c_2^6 q_2 c_1 + 32\eta c_2^5 q_2 c_1 - 1008 c_2^5 q_2 c_1 + 80\eta c_2^4 q_2 c_1 + 2880 c_2^4 q_2 c_1 - 64\eta c_2^3 q_2 c_1 + 2016 c_2^3 q_2 c_1 -$$
$$280\eta c_2^2 q_2 c_1 - 7488 c_2^2 q_2 c_1 + 160\eta q_2 c_1 + 32\eta c_2 q_2 c_1 - 1008 c_2 q_2 c_1 + 4032 q_2 c_1 - 6048 c_1 + 4\eta c_2^6 + 36 c_2^6 -$$
$$96\eta c_2^5 + 432 c_2^5 + 616\eta c_2^4 + 792 c_2^4 - 528\eta c_2^3 - 2160 c_2^3 + 676\eta c_2^2 + 6948 c_2^2 + 36\eta c_2^6 q_2^2 + 324 c_2^6 q_2^2 + 32\eta c_2^5 q_2^2 -$$
$$144 c_2^5 q_2^2 + 1064\eta c_2^4 q_2^2 + 1368 c_2^4 q_2^2 + 176\eta c_2^3 q_2^2 + 720 c_2^3 q_2^2 + 1220\eta c_2^2 q_2^2 + 11844 c_2^2 q_2^2 + 272\eta q_2^2 + 80\eta c_2 q_2^2 +$$
$$2016 c_2 q_2^2 + 9792 q_2^2 + 144\eta - 240\eta c_2 - 6048 c_2 + 24\eta c_2^6 q_2 + 216 c_2^6 q_2 + 64\eta c_2^5 q_2 - 288 c_2^5 q_2 + 240\eta c_2^4 q_2 +$$
$$1296 c_2^4 q_2 + 352\eta c_2^3 q_2 + 1440 c_2^3 q_2 + 216\eta c_2^2 q_2 + 216 c_2^2 q_2 + 96\eta q_2 + 160\eta c_2 q_2 + 4032 c_2 q_2 + 3456 q_2 + 5184$$

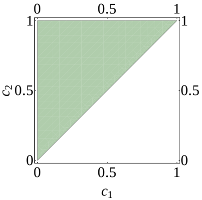
The Bound

- Consider extreme cases for (c_1, c_2)
- Consider the limits $\eta \to 0$ and $\eta \to \infty$



The Bound

- Consider extreme cases for (c_1, c_2)
- Consider the limits $\eta \to 0$ and $\eta \to \infty$

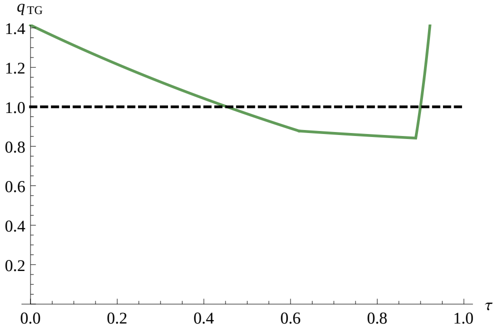


This yields the following *guess*:

$$B(q_2) = \begin{cases} \left(\frac{q_2+3}{4}\right)^2, & 0 < q_2 < Q_2, \\ q_2(q_2+1), & Q_2 \leq q_2 < 1. \end{cases},$$

with $Q_2 = \frac{1}{15}(4\sqrt{10} - 5)$.

Bound for the Convergence Rate



Two-grid convergence factor depending on τ for
$\nu = \nu_{pre} + \nu_{post} = 2 + 2$ smoothing steps

Experimental all-at-once analysis for 2D

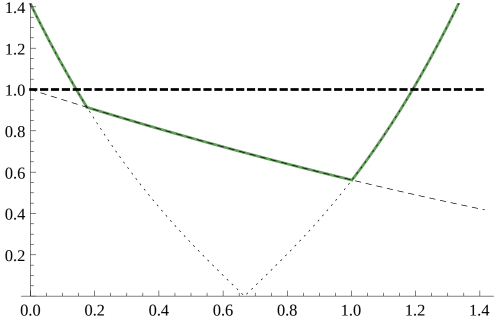
- Set up the full matrix for the all-at-once approach

Experimental all-at-once analysis for 2D

- Set up the full matrix for the all-at-once approach
- Consider *only* the limiting cases for c_1, c_2, η

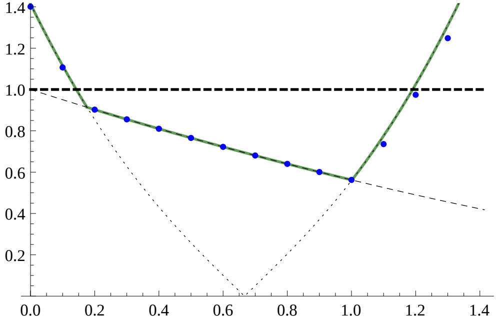
Experimental all-at-once analysis for 2D

- Set up the full matrix for the all-at-once approach
- Consider *only* the limiting cases for c_1, c_2, η
- $q_{TG}(\tau) = \max\left\{\frac{1}{16}(\tau-4)^2, \frac{1}{4}|3\tau-2|\sqrt{9\tau^2-12\tau+8}\right\}$



Experimental all-at-once analysis for 2D

- Set up the full matrix for the all-at-once approach
- Consider *only* the limiting cases for c_1, c_2, η
- $q_{TG}(\tau) = \max\left\{\frac{1}{16}(\tau-4)^2, \frac{1}{4}|3\tau-2|\sqrt{9\tau^2-12\tau+8}\right\}$



http://www.risc.jku.at/people/vpillwei/sLFA/

Chapter 0, Basic Concepts, Brenner+Scott, 2002

The finite element method provides a formalism for generating discrete (finite) algorithms for approximating the solutions of partial differential equations. It should be thought of as a black box into which one puts the differential equation (boundary value problem) and out of which pops an algorithm for approximating corresponding solutions. *Such a task could conceivably be done automatically by a computer*, but it necessitates an amount of mathematical skill that today still requires human involvement. The purpose of this book is to help people become adept at working the magic of this black box. The book does *not* focus on how to *turn the resulting algorithms into computer codes*, although this is at present also a complicated task. The latter is, however, a more well-defined task than the former and thus *potentially more amenable to automation*.