Finite Element Methods

FEM for Poisson problem in 2D - Triangular element

2016-2 CSE6820

Outline

2D Poisson problem

Consider the two dimensional Poisson problem

$$\begin{aligned} -\Delta u &= f \qquad && \text{in } \Omega \\ u &= u_D && \text{on } \partial\Omega \end{aligned}$$

Weak formulation

Find $u(x, y) \in H_0^1(\Omega)$ such that

$$\int_\Omega \nabla u \cdot \nabla v \, d\boldsymbol{x} = \int_\Omega fv \, d\boldsymbol{x}, \qquad \forall v \in H_0^1(\Omega).$$

Variational formulation

Find $u_h \in V_h^k$ such that

$$\int_\Omega \nabla u_h \cdot \nabla v_h \, d\boldsymbol{x} = \int_\Omega f v_h \, d\boldsymbol{x}, \qquad \forall v_h \in V_h^k$$

where

$$V_h^k = \{v_h \in H_0^1(\Omega) \mid v_h|_T \in P_k(T), \ \forall T \in \mathcal{T}_h\},$$

and $P_k(R)$ is the polynomial function space of degrees $\leq k$.

Affine mapping

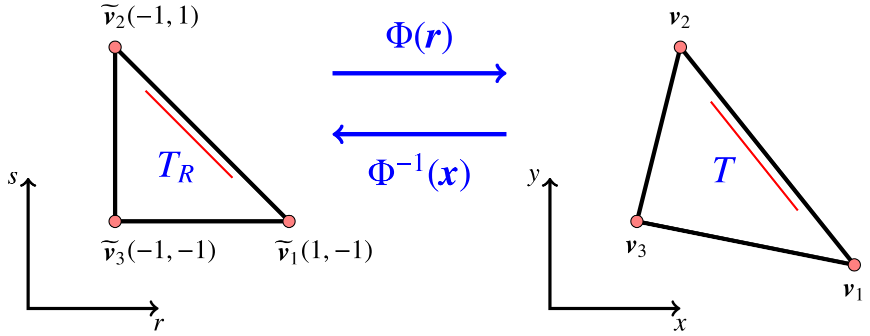


Figure: An affine mapping from the reference triangle T_R to a triangle T.

Barycentric coordinates

Barycentric coordinates $\{\lambda_i\}_{i=1}^3$ have the properties

$$\begin{cases} 0 \le \lambda_i(\boldsymbol{x}) \le 1, & i = 1,\ 2,\ 3 \\ \sum_{i=1}^3 \lambda_i(\boldsymbol{x}) = 1. \end{cases}$$

Then we have an affine mapping Φ such that

$$\Phi(\boldsymbol{r}) = \lambda_1 \boldsymbol{v}_1 + \lambda_2 \boldsymbol{v}_2 + \lambda_2 \boldsymbol{v}_3 = \boldsymbol{x},$$

where $\boldsymbol{x}$ is a point in T.

Properties

1. $\widetilde{\lambda}_1 = \dfrac{r+1}{2}, \quad \widetilde{\lambda}_2 = \dfrac{s+1}{2}, \quad \widetilde{\lambda}_3 = -\dfrac{r+s}{2}$

2. $x_r = \dfrac{v_1^{(1)} - v_3^{(1)}}{2}, \quad y_r = \dfrac{v_1^{(2)} - v_3^{(2)}}{2}, \quad x_s = \dfrac{v_2^{(1)} - v_3^{(1)}}{2}, \quad y_s = \dfrac{v_2^{(2)} - v_3^{(2)}}{2}$

3. $r_x = \dfrac{y_s}{J}, \qquad r_y = -\dfrac{x_s}{J}, \qquad s_x = -\dfrac{y_r}{J}, \qquad s_y = \dfrac{x_r}{J}$

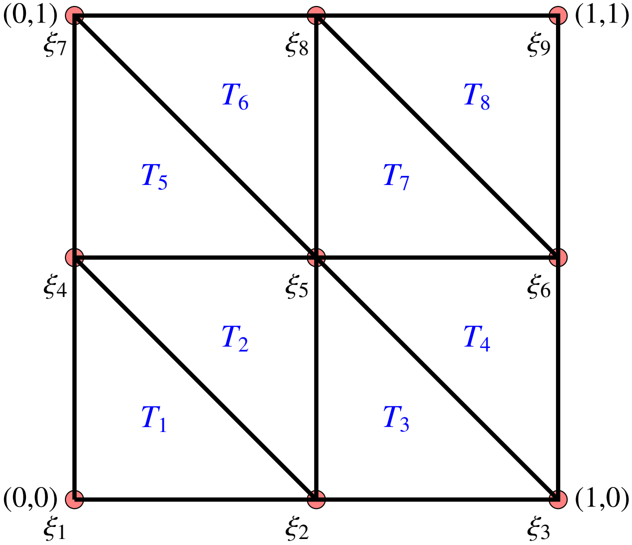
4. $\lambda_i(\boldsymbol{x}) = \widetilde{\lambda}_i(\Phi^{-1}(\boldsymbol{x})), \qquad \widetilde{\lambda}_i(\boldsymbol{r}) = \lambda_i(\Phi(\boldsymbol{r}))$

5. $\dfrac{d}{dx}\lambda_i(\boldsymbol{x}) = r_x \dfrac{d}{dr}\widetilde{\lambda}_i(\boldsymbol{r}) + s_x \dfrac{d}{ds}\widetilde{\lambda}_i(\boldsymbol{r})$

6. $\dfrac{d}{dy}\lambda_i(\boldsymbol{x}) = r_y \dfrac{d}{dr}\widetilde{\lambda}_i(\boldsymbol{r}) + s_y \dfrac{d}{ds}\widetilde{\lambda}_i(\boldsymbol{r})$

Triangulation data (P_1)

If $\Omega = [0,\ 1]^2$, $M = 2$ and $k = 1$, data are stored as follows.



Triangulation data (P_1)

$$\text{c4n} = \begin{pmatrix} 0 & 1/2 & 1 & 0 & 1/2 & 1 & 0 & 1/2 & 1 \\ 0 & 0 & 0 & 1/2 & 1/2 & 1/2 & 1 & 1 & 1 \end{pmatrix}$$

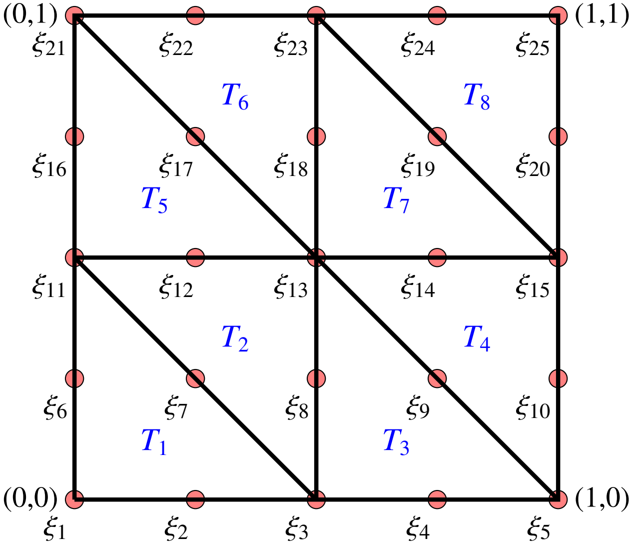
$$\text{n4e} = \begin{pmatrix} 2 & 4 & 3 & 5 & 5 & 7 & 6 & 8 \\ 4 & 2 & 5 & 3 & 7 & 5 & 8 & 6 \\ 1 & 5 & 2 & 6 & 4 & 8 & 5 & 9 \end{pmatrix}$$

$$\text{n4db} = (1,\ 2,\ 3,\ 4,\ 6,\ 7,\ 8,\ 9)$$

$$\text{ind4e} = \begin{pmatrix} 1 & 5 & 2 & 6 & 4 & 8 & 5 & 9 \\ 2 & 4 & 3 & 5 & 5 & 7 & 6 & 8 \\ 4 & 2 & 5 & 3 & 7 & 5 & 8 & 6 \end{pmatrix}$$

Triangulation data (P_2)

If $\Omega = [0,\ 1]^2$, $M = 2$ and $k = 2$, data are stored as follows.



Triangulation data (P_2)

$$\mathrm{c4n} = \begin{pmatrix} 0 & 1/4 & 1/2 & 3/4 & 1 & 0 & 1/4 & 1/2 & 3/4 & 1 & 0 & 1/4 & 1/2 & 3/4 & 1 & 0 & 1/4 & 1/2 & 3/4 & 1 & 0 & 1/4 & 1/2 & 3/4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1/4 & 1/4 & 1/4 & 1/4 & 1/4 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 3/4 & 3/4 & 3/4 & 3/4 & 3/4 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathrm{n4e} = \begin{pmatrix} 3 & 11 & 5 & 13 & 13 & 21 & 15 & 23 \\ 11 & 3 & 13 & 5 & 21 & 13 & 23 & 15 \\ 1 & 13 & 3 & 15 & 11 & 23 & 13 & 25 \end{pmatrix}$$

$$\mathrm{n4db} = (1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 10,\ 11,\ 15,\ 16,\ 20,\ 21,\ 22,\ 23,\ 24,\ 25)$$

$$\mathrm{ind4e} = \begin{pmatrix} 1 & 13 & 3 & 15 & 11 & 23 & 13 & 25 \\ 2 & 12 & 4 & 14 & 12 & 22 & 14 & 24 \\ 3 & 11 & 5 & 13 & 13 & 21 & 15 & 23 \\ 6 & 8 & 8 & 10 & 16 & 18 & 18 & 20 \\ 7 & 7 & 9 & 9 & 17 & 17 & 19 & 19 \\ 11 & 3 & 13 & 5 & 21 & 13 & 23 & 15 \end{pmatrix}$$

mesh_FEM2D_Tri

This Matlab code generates an uniform triangular mesh on the domain $[x_\ell, x_r] \times [y_\ell, y_r]$ in 2D with $2M_x$ elements along x-direction and $2M_y$ elements along y-direction. Also this code returns an index matrix for continuous k-th order polynomial approximations.

Basis functions

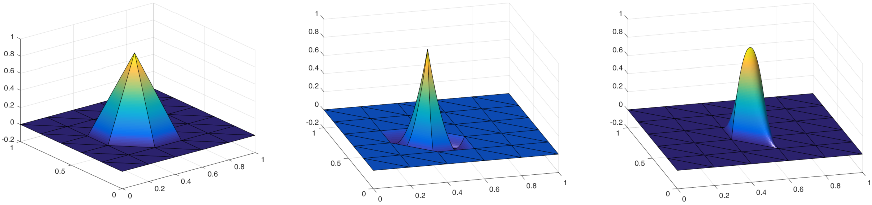
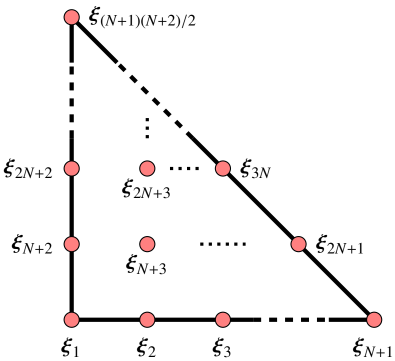


Figure: Global basis functions of V_h^1 (left) and V_h^2 (others) on an interval.

$$\psi_i(\boldsymbol{\xi}_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases}$$

$$\sum_{i=1}^{N} \psi_i(\boldsymbol{x}) = 1 \qquad \forall x \in \Omega$$

Local basis functions

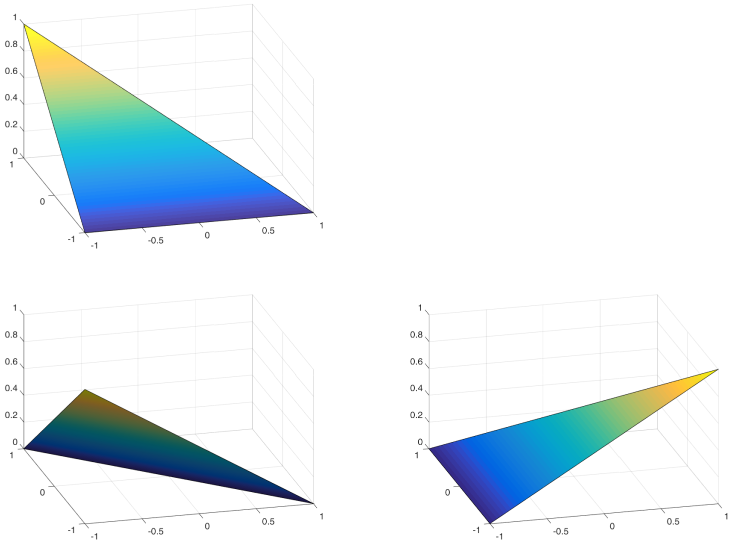


$$\xi_j^n = \xi^n_{(\ell-1)(k+2-\ell/2)+m}$$

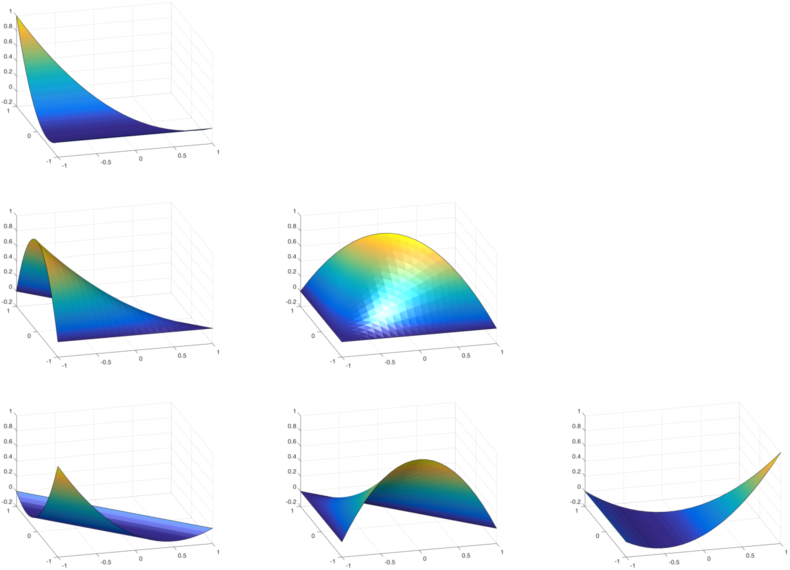
$(\ell$: row number, $\;m$: column number, $\;\ell + m = k + 1)$

Figure: Node numbering in the n-th triangular element

Local basis functions



Local basis functions



Local basis functions

$$\begin{aligned} \psi_i^n(\boldsymbol{x}) &= 0 \quad \text{if } x \in \Omega \setminus T_n \\ \sum_{i=1}^{(k+1)(k+2)/2} \psi_i^n(\boldsymbol{x}) &= 1 \quad \forall x \in T_n. \end{aligned}$$

Interpolation

Using these basis functions, the interpolate function of a function $f(\boldsymbol{x})$ can be written as follows

$$\mathcal{I}f(\boldsymbol{x}) = \sum_{i=1}^{N} f_i \psi_i(\boldsymbol{x})$$

where $f_i = f(\boldsymbol{\xi}_i)$.

Numerical solutions

- Numerical solution

$$u_h = \sum_{i=1}^{N} u_i \psi_i$$

- Local solution

$$u_h\Big|_{R_n} = \sum_{i=1}^{(k+1)^2} u_i^n \psi_i^n$$

- Derivatives

$$\nabla u_h = \sum_{i=1}^{N} u_i \nabla \psi_i,$$

$$\nabla u_h\Big|_{R_n} = \sum_{i=1}^{(k+1)^2} u_i^n \nabla \psi_i^n.$$

Variational formulation

For a test function $\psi_i(\boldsymbol{x}) \in V_h^k$, the variational formulation can be rewritten as

$$\sum_{j=1}^{N} u_j \int_\Omega \nabla\psi_i \cdot \nabla\psi_j \, d\boldsymbol{x} = \int_\Omega f\psi_i \, d\boldsymbol{x}$$

Finite element system

$$A\boldsymbol{u} = \boldsymbol{b}$$

where

$$(A)_{ij} = \int_\Omega \nabla\psi_i \cdot \nabla\psi_j \, d\boldsymbol{x}$$
$$(\boldsymbol{b})_i = \int_\Omega f\psi_i \, d\boldsymbol{x}$$
$$(\boldsymbol{u})_j = u_j.$$

Finite element system

$$A\boldsymbol{u} = M\boldsymbol{f}$$

where

$$\begin{aligned}
(A)_{ij} &= \int_\Omega \nabla\psi_i \cdot \nabla\psi_j \, d\boldsymbol{x} \\
(\boldsymbol{u})_j &= u_j \\
(M)_{ij} &= \int_\Omega \psi_i\psi_j \, d\boldsymbol{x} \\
(\boldsymbol{f})_j &= f_j.
\end{aligned}$$

Matrices

$$A = \sum_{\ell=1}^{M} A_{T_n}$$
$$M = \sum_{\ell=1}^{M} M_{T_n}$$

where $(k+1)(k+2)/2$-by-$(k+1)(k+2)/2$ matrices A_{R_n} and M_{R_n} are defined as

$$(A_{T_n})_{ij} = \int_{T_n} \nabla\psi_i^n(x) \cdot \nabla\psi_j^n \, d\boldsymbol{x}$$
$$(M_{T_n})_{ij} = \int_{T_n} \psi_i^n \psi_j^n \, d\boldsymbol{x}$$

where $1 \le i, \ j \le (k+1)(k+2)/2$.

Differentiation matrix

$$(Dx)_{ij} = \frac{\partial \psi_j}{\partial x}(\boldsymbol{\xi}_i), \qquad (Dy)_{ij} = \frac{\partial \psi_j}{\partial y}(\boldsymbol{\xi}_i).$$

$\Rightarrow$

$$\frac{\partial}{\partial x}\psi_i(\boldsymbol{x}) = \sum_{j=1}^{N} \frac{\partial \psi_i}{\partial x}(\boldsymbol{\xi}_j)\psi_j(\boldsymbol{x}) = (Dx^t)_i \boldsymbol{\psi}$$

$$\frac{\partial}{\partial y}\psi_i(\boldsymbol{x}) = \sum_{j=1}^{N} \frac{\partial \psi_i}{\partial y}(\boldsymbol{\xi}_j)\psi_j(\boldsymbol{x}) = (Dy^t)_i \boldsymbol{\psi}$$

where

$$(Dx^t)_i = [\frac{\partial \psi_i}{\partial x}(\xi_1) \ \cdots \ \frac{\partial \psi_i}{\partial x}(\xi_N)]$$

$$(Dy^t)_i = [\frac{\partial \psi_i}{\partial y}(\xi_1) \ \cdots \ \frac{\partial \psi_i}{\partial y}(\xi_N)]$$

$$\boldsymbol{\psi} = [\psi_1(x) \ \cdots \ \psi_N(x)]^t$$

Properties of D

- $$Dx = \sum_{n=1}^{2M^2} Dx_{T_n}, \qquad (Dx_{T_n})_{ij} = \frac{\partial \psi_j^n}{\partial x}(\boldsymbol{\xi}_i^n)$$

- $$Dy = \sum_{n=1}^{2M^2} Dy_{T_n}, \qquad (Dy_{T_n})_{ij} = \frac{\partial \psi_j^n}{\partial y}(\boldsymbol{\xi}_i^n)$$

- $$\nabla \psi_i^n(\boldsymbol{x}) = \Big((Dx_{T_n}^t)_i \boldsymbol{\psi}^n,\ (Dy_{T_n}^t)_i \boldsymbol{\psi}^n \Big)$$

- $$\nabla u_h(\boldsymbol{\xi}_m) = \Big((Dx)_m \boldsymbol{u},\ (Dy)_m \boldsymbol{u} \Big)$$

- $$\nabla u_h(\boldsymbol{\xi}_m^n) = \Big((Dx_{T_n})_m \boldsymbol{u},\ (Dy_{T_n})_m \boldsymbol{u} \Big)$$

Mass matrix

$$(M)_{ij} = \int_{R_n} \psi_i(\boldsymbol{x})\psi_j(\boldsymbol{x})\, d\boldsymbol{x} = J \int_{R_R} \widetilde{\psi}_i(\boldsymbol{r})\widetilde{\psi}_j(\boldsymbol{r})\, dr = J(M_R)_{ij}$$

$$\Rightarrow \qquad M = JM_R$$

Stiffness matrix

$$(S)_{ij} = J\Big[(r_x^2 + r_y^2)(S_R^{rr})_{ij} + (r_x s_x + r_y s_y)((S_R^{rs})_{ij} + (S_R^{sr})_{ij}) + (s_x^2 + s_y^2)(S_R^{ss})_{ij}\Big]$$

where

$$\begin{aligned}
(S_R^{rr})_{ij} &= (Dr_R^t M_R Dr_R)_{ij} \\
(S_R^{rs})_{ij} &= (Dr_R^t M_R Ds_R)_{ij} \\
(S_R^{sr})_{ij} &= (Ds_R^t M_R Dr_R)_{ij} \\
(S_R^{ss})_{ij} &= (Ds_R^t M_R Ds_R)_{ij}
\end{aligned}$$

$$\Rightarrow \qquad J\Big((r_x^2 + r_y^2)S_R^{rr} + (r_x s_x + r_y s_y)(S_R^{rs} + S_R^{sr}) + (s_x^2 + s_y^2)S_R^{ss}\Big)$$

Lemma

For a given interval $T = \text{conv}\{\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3\}$ with barycentric coordinates $\lambda_1,\ \lambda_2$ and λ_3, it holds for $a,\ b,\ c \in \mathbb{N}_0$ that

$$\int_T \lambda_1^a \lambda_2^b \lambda_3^c \, d\boldsymbol{x} = 2|T| \frac{a!\, b!\, c!}{(a+b+c+2)!}$$

where $|T|$ is area of T.

P_1 matrices

$$
\begin{aligned}
\widetilde{\psi}_1(\boldsymbol{r}) &= \widetilde{\lambda}_3(\boldsymbol{r}), \qquad & \nabla\widetilde{\psi}_1(\boldsymbol{r}) &= \Big(-\frac{1}{2},\ -\frac{1}{2}\Big),\\
\widetilde{\psi}_2(\boldsymbol{r}) &= \widetilde{\lambda}_1(\boldsymbol{r}), \qquad & \nabla\widetilde{\psi}_2(\boldsymbol{r}) &= \Big(\frac{1}{2},\ 0\Big),\\
\widetilde{\psi}_3(\boldsymbol{r}) &= \widetilde{\lambda}_2(\boldsymbol{r}), \qquad & \nabla\widetilde{\psi}_3(\boldsymbol{r}) &= \Big(0,\ \frac{1}{2}\Big),
\end{aligned}
$$

P_1 matrices

$$M_R = \frac{1}{6}\begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}.$$

P_1 matrices

$$Dr_R = \frac{1}{2}\begin{pmatrix} -1 & 1 & 0 \\ -1 & 1 & 0 \\ -1 & 1 & 0 \end{pmatrix}$$

$$Ds_R = \frac{1}{2}\begin{pmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{pmatrix}.$$

P_1 matrices

$$S_R^{rr} = Dr_R^t M_R Dr_R = \frac{1}{2}\begin{pmatrix} 1 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$S_R^{rs} = Dr_R^t M_R Ds_R = \frac{1}{2}\begin{pmatrix} 1 & 0 & -1 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

$$S_R^{sr} = Ds_R^t M_R Dr_R = \frac{1}{2}\begin{pmatrix} 1 & -1 & 0 \\ 0 & 0 & 0 \\ -1 & 1 & 0 \end{pmatrix}$$

$$S_R^{ss} = Ds_R^t M_R Ds_R = \frac{1}{2}\begin{pmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{pmatrix}$$

P_2 matrices

$$
\begin{aligned}
\widetilde{\psi}_1(\boldsymbol{r}) &= \widetilde{\lambda}_3(\boldsymbol{r})(2\widetilde{\lambda}_3(\boldsymbol{r})-1), & \nabla\widetilde{\psi}_1(\boldsymbol{r}) &= \Big(-2\widetilde{\lambda}_3(\boldsymbol{r})+\frac{1}{2},\ -2\widetilde{\lambda}_3(\boldsymbol{r})+\frac{1}{2}\Big),\\
\widetilde{\psi}_2(\boldsymbol{r}) &= 4\widetilde{\lambda}_1(\boldsymbol{r})\widetilde{\lambda}_3(\boldsymbol{r}), & \nabla\widetilde{\psi}_2(\boldsymbol{r}) &= \Big(2\widetilde{\lambda}_3(\boldsymbol{r})-2\widetilde{\lambda}_1(\boldsymbol{r}),\ -2\widetilde{\lambda}_1(\boldsymbol{r})\Big),\\
\widetilde{\psi}_3(\boldsymbol{r}) &= \widetilde{\lambda}_1(\boldsymbol{r})(2\widetilde{\lambda}_1(\boldsymbol{r})-1), & \nabla\widetilde{\psi}_3(\boldsymbol{r}) &= \Big(2\widetilde{\lambda}_1(\boldsymbol{r})-\frac{1}{2},\ 0\Big),\\
\widetilde{\psi}_4(\boldsymbol{r}) &= 4\widetilde{\lambda}_2(\boldsymbol{r})\widetilde{\lambda}_3(\boldsymbol{r}), & \nabla\widetilde{\psi}_4(\boldsymbol{r}) &= \Big(-2\widetilde{\lambda}_2(\boldsymbol{r}),\ 2\widetilde{\lambda}_2(\boldsymbol{r})-2\widetilde{\lambda}_3(\boldsymbol{r})\Big),\\
\widetilde{\psi}_5(\boldsymbol{r}) &= 4\widetilde{\lambda}_1(\boldsymbol{r})\widetilde{\lambda}_2(\boldsymbol{r}), & \nabla\widetilde{\psi}_5(\boldsymbol{r}) &= \Big(2\widetilde{\lambda}_2(\boldsymbol{r}),\ 2\widetilde{\lambda}_1(\boldsymbol{r})\Big),\\
\widetilde{\psi}_6(\boldsymbol{r}) &= \widetilde{\lambda}_2(\boldsymbol{r})(2\widetilde{\lambda}_2(\boldsymbol{r})-1), & \nabla\widetilde{\psi}_6(\boldsymbol{r}) &= \Big(0,\ 2\widetilde{\lambda}_2(\boldsymbol{r})-\frac{1}{2}\Big),
\end{aligned}
$$

P_2 matrices

$$M_R = \frac{1}{90}\begin{pmatrix} 6 & 0 & -1 & 0 & -4 & -1 \\ 0 & 32 & 0 & 16 & 16 & -4 \\ -1 & 0 & 6 & -4 & 0 & -1 \\ 0 & 16 & -4 & 32 & 16 & 0 \\ -4 & 16 & 0 & 16 & 32 & 0 \\ -1 & -4 & -1 & 0 & 0 & 6 \end{pmatrix},$$

P_2 matrices

$$Dr_R = \frac{1}{2}\begin{pmatrix} -3 & 4 & -1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 \\ 1 & -4 & 3 & 0 & 0 & 0 \\ -1 & 2 & -1 & -2 & 2 & 0 \\ 1 & -2 & 1 & -2 & 2 & 0 \\ 1 & 0 & -1 & -4 & 4 & 0 \end{pmatrix}$$

$$Ds_R = \frac{1}{2}\begin{pmatrix} -3 & 0 & 0 & 4 & 0 & -1 \\ -1 & -2 & 0 & 2 & 2 & -1 \\ 1 & -4 & 0 & 0 & 4 & -1 \\ -1 & 0 & 0 & 0 & 0 & 1 \\ 1 & -2 & 0 & -2 & 2 & 1 \\ 1 & 0 & 0 & -4 & 0 & 3 \end{pmatrix}$$

P_2 matrices

$$S_R^{rr} = \frac{1}{6}\begin{pmatrix} 3 & -4 & 1 & 0 & 0 & 0 \\ -4 & 8 & -4 & 0 & 0 & 0 \\ 1 & -4 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & -8 & 0 \\ 0 & 0 & 0 & -8 & 8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$S_R^{rs} = \frac{1}{6}\begin{pmatrix} 3 & 0 & 0 & -4 & 0 & 1 \\ -4 & 4 & 0 & 4 & -4 & 0 \\ 1 & -4 & 0 & 0 & 4 & -1 \\ 0 & 4 & 0 & 4 & -4 & -4 \\ 0 & -4 & 0 & -4 & 4 & 4 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

P_2 matrices

$$S_R^{sr} = \frac{1}{6}\begin{pmatrix} 3 & -4 & 1 & 0 & 0 & 0 \\ 0 & 4 & -4 & 4 & -4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -4 & 4 & 0 & 4 & -4 & 0 \\ 0 & -4 & 4 & -4 & 4 & 0 \\ 1 & 0 & -1 & -4 & 4 & 0 \end{pmatrix}$$

$$S_R^{ss} = \frac{1}{6}\begin{pmatrix} 3 & 0 & 0 & -4 & 0 & 1 \\ 0 & 8 & 0 & 0 & -8 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -4 & 0 & 0 & 8 & 0 & -4 \\ 0 & -8 & 0 & 0 & 8 & 0 \\ 1 & 0 & 0 & -4 & 0 & 3 \end{pmatrix}$$

MatrixforPoisson_2D_Tri

This Matlab code generates the mass matrix M_R, the stiffness matrices $Srr_R,\ Srs_R,\ Ssr_R,\ Sss_R$ and the differentiation matrices $Dr_R,\ Ds_R$ for continuous k-th order polynomial approximations on the reference triangle T_R.

FEMforPoisson_2D_Tri

The following Matlab code solves the Poisson problem. In order to use this code, mesh information (c4n, n4e, n4db, ind4e), matrices ($M_R, S_R^{rr}, S_R^{rs}, S_R^{sr}, S_R^{ss}$), the source f, and the boundary condition u_D. Then the results of this code are the numerical solution u, the global stiffness matrix A, the global load vector b and the freenodes.

```
function [u, A, b, fns] = ...
    FEMforPoisson_2D_Tri(c4n,n4e,n4db,ind4e,M_R,Srr_R,Srs_R,Ssr_R,Sss_R,f,u_D)
A = sparse(length(c4n),length(c4n));
b = zeros(length(c4n),1);
u = b;
for j=1:length(n4e)
    xr = (c4n(1,n4e(1,j))-c4n(1,n4e(3,j)))/2;
    yr = (c4n(2,n4e(1,j))-c4n(2,n4e(3,j)))/2;
    xs = (c4n(1,n4e(2,j))-c4n(1,n4e(3,j)))/2;
    ys = (c4n(2,n4e(2,j))-c4n(2,n4e(3,j)))/2;
    J = xr*ys-xs*yr;
    rx=ys/J; ry=-xs/J; sx=-yr/J; sy=xr/J;
    A(ind4e(:,j),ind4e(:,j)) = A(ind4e(:,j),ind4e(:,j)) + J*((rx^2+sx^2)*Srr_R ...
         + (rx*sx+ry*sy)*(Srs_R+Ssr_R) + (ry^2+sy^2)*Sss_R);
    b(ind4e(:,j)) = b(ind4e(:,j)) + J*M_R*f(c4n(:,ind4e(:,j))');
end
fns = setdiff(1:length(c4n), n4db);
u(fns) = A(fns,fns)\b(fns);
end
```

ComputeErrorFEM_2D_Tri

The following Matlab code computes the semi H1 error between the exact solution and the numerical solution.

```
function error = ...
    ComputeErrorFEM_2D_Tri(c4n,n4e,ind4e,M_R,Dr_R,Ds_R,u,ux,uy)
error = 0;
for j=1:size(ind4e,2)
    xr = (c4n(1,n4e(1,j))-c4n(1,n4e(3,j)))/2;
    yr = (c4n(2,n4e(1,j))-c4n(2,n4e(3,j)))/2;
    xs = (c4n(1,n4e(2,j))-c4n(1,n4e(3,j)))/2;
    ys = (c4n(2,n4e(2,j))-c4n(2,n4e(3,j)))/2;
    J = xr*ys-xs*yr;
    rx=ys/J; ry=-xs/J; sx=-yr/J; sy=xr/J;

    Dex=ux(c4n(:,ind4e(:,j))') - (rx+sx)*Dr_R*u(ind4e(:,j));
    Dey=uy(c4n(:,ind4e(:,j))') - (ry+sy)*Ds_R*u(ind4e(:,j));
    error=error+J*(Dex'*M_R*Dex+Dey'*M_R*Dey);
end
error=sqrt(error);
end
```

main_FEMforPoisson_2D_Tri

The following Matlab code solves the Poisson problem by using several matlab codes such as mesh_FEM2D_Tri_rectangle, MatrixforPoisson_2D_Tri, FEMforPoisson_2D_Tri and ComputeErrorFEM_2D_Tri.

```
iter = 10;
xl = 0; xr = 1; yl = 0; yr = 1; k = 2; M = 2.^(1:iter);
f=@(x) 2*pi^2*sin(pi*x(:,1)).*sin(pi*x(:,2));
u_D=@(x) x(:,1)*0;
ux=@(x) pi*cos(pi*x(:,1)).*sin(pi*x(:,2));
uy=@(x) pi*sin(pi*x(:,1)).*cos(pi*x(:,2));

error=zeros(1,iter);
h=1./M;
for j=1:iter
    [c4n, n4e, ind4e, n4db] = ...
        mesh_FEM2D_Tri_rectangle(xl, xr, yl, yr, M(j), M(j), k);
    [M_R, Srr_R, Srs_R, Ssr_R, Sss_R, Dr_R, Ds_R] = ...
        MatrixforPoisson_2D_Tri(k)
    u = FEMforPoisson_2D_Rec(c4n,n4e,n4db,ind4e, ...
        M_R,Srr_R,Sss_R,f,u_D);
    error(j) = ComputeErrorFEM_2D_Tri(c4n,n4e,ind4e,M_R,Dr_R,Ds_R,u,ux,uy);
end
```

Example

Consider the domain $\Omega = [0, 1]^2$. The source term f is chosen such that

$$u = \sin(\pi x) \sin(\pi y)$$

is the analytical solution to the Poisson problem.

Convergence history

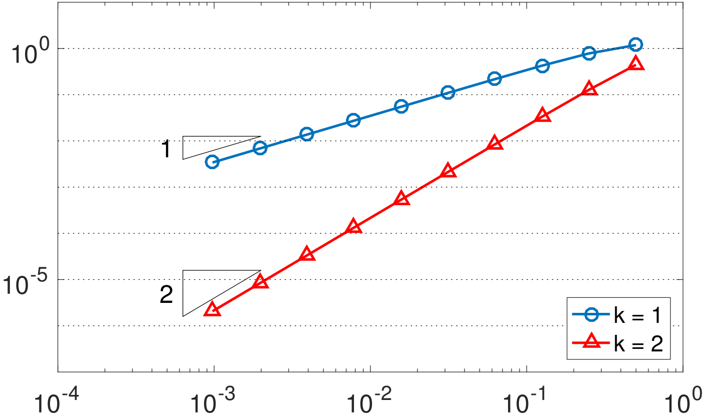


Figure: Convergence history for Example