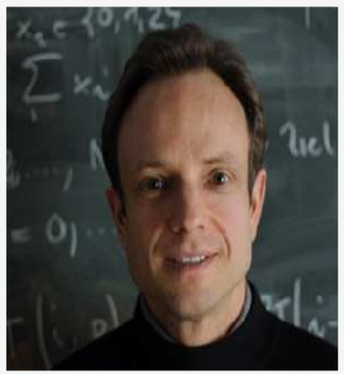
ETH Lecture 401-0674-00L Numerical Methods for Partial Differential Equations

Homework Problem Video Tutorial

Problem 2-12: Testing built-in quadrature rules of LEHRFEM++

Prof. R. Hiptmair, SAM, ETH Zurich

Date: March 16, 2019



a)

Definition 2.7.5.29. Order of a local quadrature rule

A local quadrature rule according to Def. 2.7.5.9 is said to be order $q \in \mathbb{N}$, if

- for a simplex K (triangle tetrahedron) it is exact for all *polynomials* $f \in \mathcal{P}_{q-1}(\mathbb{R}^d)$,
- for a tensor product element K (rectangle, brick) it is exact for all *tensor product polynomials* $f \in \mathcal{Q}_{q-1}(\mathbb{R}^d)$.

b) Write a function

```
bool testQuadOrderTria(const lf::quad::QuadRule &qr, unsigned int
  order);
```

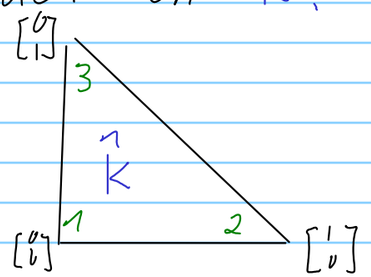
that returns true, if the passed quadrature rule for a *triangular* reference element has order `order`. Do not forget to check whether the quadrature rule is really meant to the reference triangle.

Quadrature rule on unit triangle { MatrixXd Points(), VectorXd Weights(), RefEl()

K = order ⇒ Verify exactness of QR for a basis of $\mathcal{P}_{k-1}(\mathbb{R}^2)$

Our choice: monomial basis $\{x_1^i x_2^j\}_{i+j \le k-1}$

Note: On $\hat{K}$:



$\lambda_2(x) = x_1$

$\lambda_3(x) = x_2$

$$\int_{\hat{K}} x_1^i x_2^j \, dx = \int_{\hat{K}} \lambda_2(x)^i \lambda_3(x)^j \, dx$$

(2)

Lemma 2.7.5.5. Integration of powers of barycentric coordinate functions

For any non-degenerate d-simplex K with barycentric coordinate functions $\lambda_1,\ldots,\lambda_{d+1}$ and exponents $\alpha_j \in \mathbb{N}$, $j = 1,\ldots,d+1$,

$$\int_K \lambda_1^{\alpha_1} \cdot \cdots \cdot \lambda_{d+1}^{\alpha_{d+1}} \,\mathrm{d}\boldsymbol{x} = d!|K| \,\frac{\alpha_1!\alpha_2! \cdot \cdots \cdot \alpha_{d+1}!}{(\alpha_1+\alpha_2+\cdots+\alpha_{d+1}+d)!} \quad \forall \boldsymbol{\alpha} \in \mathbb{N}_0^{d+1}\,. \tag{2.7.5.6}$$

$d=2$: $\int_{\hat{K}} x_1^i x_2^j \,dx = \frac{i!\,j!}{(i+j+2)!}$

▽ No exact comparison of floating point numbers arising from computations: Check for small ($< 10^{-9}$) relative difference!

c) Now implement a C++ function

```
bool testQuadOrderQuad(const lf::quad::QuadRule &qr,unsigned int
  order);
```

that confirms order `order` for a quadrature rule on the unit square. Make sure that the quadrature rule is really meant for the unit square.

▷ • Check exact quadrature for $p \in \mathcal{Q}_{k-1}(\mathbb{R}^2)$ [for elements $\hat{K} \to x_1^i x_2^j$, $0 \le i,j \le k-1$ of monomial bases]
• Reference element $\hat{=}$ unit square.

$$\int_0^1 \int_0^1 x_1^i x_2^j \,dx = \frac{1}{(i+1)(j+1)}\,, \quad 0 \le i,j \le k-1$$

d) Now create a C++ function

```
unsigned int calcQuadOrder(const lf::quad::QuadRule &qr);
```

that computes the *maximal order* of the quadrature rule in `qr`.

▷ Test orders until test fails