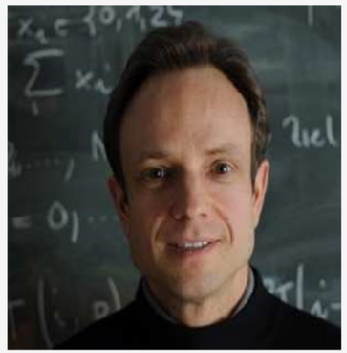
ETH Lecture 401-0674-00L Numerical Methods for Partial Differential Equations

Homework Problem Video Tutorial
Problem 2-9: Handling degrees of freedom (DOFs) in LEHRFEM++

Prof. R. Hiptmair, SAM, ETH Zurich

Date: March 2, 2019



Depends on Sections 2.6.1 (2.4), 2.7.4 (2.7.4.2)

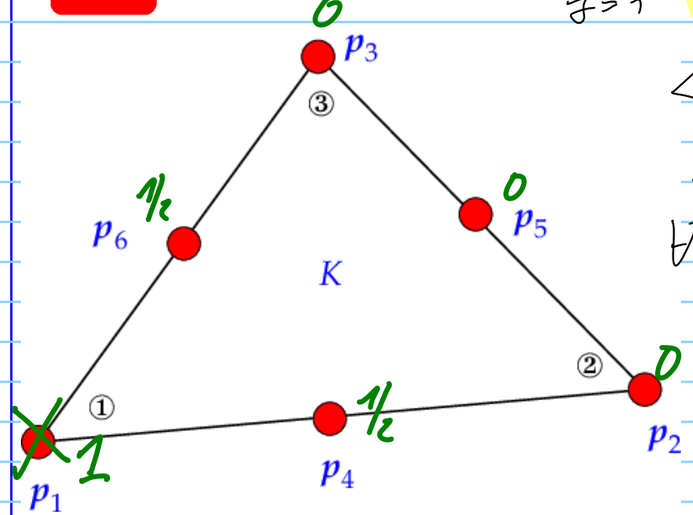
$M \hat{=}$ triangular planar mesh,

Lagrange FE spaces: $S_1^0(M)$, $S_2^0(M)$

a) LSF of $S_1^0(M)$ $\{\lambda_1, \lambda_2, \lambda_3\}$ on K expressed through LSF of $S_2^0(M)$ $\{b_K^1, \ldots, b_K^6\}$

Span $= \mathcal{P}_1(\mathbb{R}^2)$ $\subset$ Span $= \mathcal{P}_2(\mathbb{R}^2)$

$$\lambda_\ell = \sum_{j=1}^{6} ? \cdot b_K^j$$



△ interpolation nodes

By cardinal basis property

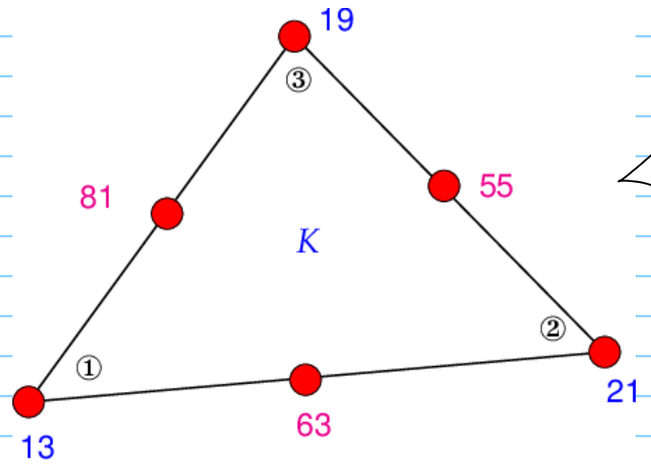
$\forall q \in \mathcal{P}_2(\mathbb{R}^2)$

$$q = \sum_{\ell=1}^{6} q(p_\ell)\, b_h^\ell$$

$$\begin{aligned}
\lambda_1 &= b_K^1 + \tfrac{1}{2} b_K^4 + \tfrac{1}{2} b_K^6 , \\
\lambda_2 &= b_K^2 + \tfrac{1}{2} b_K^5 + \tfrac{1}{2} b_K^4 , \\
\lambda_3 &= b_K^3 + \tfrac{1}{2} b_K^6 + \tfrac{1}{2} b_K^5 .
\end{aligned} \tag{2.9.1}$$

(2)

b)



For $S_2^0(M)$:

△ No of GSF associated with LSF @ interpolation nodes

The member functions of **lf::assemble::DofHandler** are explained in [Lecture → § 2.7.4.13].

- `size_type lf::assemble::DofHandler::NoLocalDofs(const lf::mesh::Entity & ) const;` = 6
- `... lf::assemble::DofHandler::GlobalDofIndices(const lf::mesh::Entity & ) const;` [13, 21, 19, 63, 55, 81]
- `size_type lf::assemble::DofHandler::NoInteriorDofs(a const lf::mesh::Entity & ) const;` = 0
- `... lf::assemble::DofHandler::InteriorGlobalDofIndices(const lf::mesh::Entity & ) const` = [ ]

c) Counting GSF associated w/ entities of different codimensions

C++11 code 2.9.2: Implementation of `CountEntityDofs()` → GITLAB

```
2  std::array<std::size_t, 3> countEntityDofs(
3      const lf::assemble::DofHandler& dofhandler) {
4    std::array<std::size_t, 3> entityDofs;
5    /* BEGIN_SOLUTION */
6    // Idea:  iterate over entities in the mesh and get interior number
       of dofs for
7    // each
8    std::shared_ptr<const lf::mesh::Mesh> mesh = dofhandler.Mesh();
9    for (std::size_t codim = 0; codim <= 2; ++codim) {
10     entityDofs[codim] = 0;
11     for (const auto& el : mesh->Entities(codim)) {
12       if (el.RefEl() == lf::base::RefEl::kQuad()) {
13         throw "Only triangular meshes are allowed!";
14       }
15       entityDofs[codim] += dofhandler.NoInteriorDofs(el);
16     }
17   }
18   /* END_SOLUTION */
19   return entityDofs;
20 }
```

d) Count GSF on $\partial\Omega$

Use flag EntitiesOnBoundary()

③

C++11 code 2.9.3: Sub-problem (2-9.d): Implementation of `CountBoundaryDofs()`
→ GITLAB

```
std::size_t countBoundaryDofs(const lf::assemble::DofHandler& dofhandler) {
  std::shared_ptr<const lf::mesh::Mesh> mesh = dofhandler.Mesh();
  // given an entity, bd_flags(entity) == true, if the entity is on the
  // boundary
  lf::mesh::utils::AllCodimMeshDataSet<bool> bd_flags(
      lf::mesh::utils::flagEntitiesOnBoundary(mesh));
  std::size_t no_dofs_on_bd = 0;
  /* BEGIN_SOLUTION */
  // Edges and nodes can be on the boundary
  for (const auto& edge : mesh->Entities(1)) {
    if (bd_flags(edge)) {
      no_dofs_on_bd += dofhandler.NoInteriorDofs(edge);
    }
  }
  for (const auto& node : mesh->Entities(2)) {
    if (bd_flags(node)) {
      no_dofs_on_bd += dofhandler.NoInteriorDofs(node);
    }
  }
  /* END_SOLUTION */
  return no_dofs_on_bd;
}
```

e_1

Write a C++ function

```
double integrateLinearFEFunction(
    const lf::assemble::DofHandler &dofhandler,
    const Eigen::VectorXd &mu);
```

← basis expansion coefficients

that computes $\int_\Omega u_h \,\mathrm{d}x$ for $u_h \in \mathcal{S}_1^0(\mathcal{M})$, whose nodal basis expansion coefficients are passed in `mu`. The `dofhandler` arguments provides the local→global index mapping. Check whether the **`lf::assemble::DofHandler`** object `dofhandler` really fits the Lagrangian finite element space $\mathcal{S}_1^0(\mathcal{M})$.

$$\mu_h \in \mathcal{S}_1^0(\mathcal{M}) \Rightarrow \mu_{h|K} = \sum_{\ell=1}^{3} \mu_h(a_K^\ell)\lambda_\ell$$

$$\int_K \lambda_\ell \,dx = \tfrac{1}{3}|K|$$

$$\int_\Omega \mu_h \,dx = \sum_{K\in\mathcal{M}} \tfrac{1}{3}|K| \sum_{\ell=1}^{3} \mu_h(a_K^\ell)$$

↑ entries of coefficient vector $\vec{\mu}$

C++11 code 2.9.4: Sub-problem (2-9.e): Implementation of `IntegrateLinearFEFunction`
→ GITLAB

```
double integrateLinearFEFunction(
    const lf::assemble::DofHandler& dofhandler,
    const Eigen::VectorXd& mu) {
  double I = 0;
  /* BEGIN_SOLUTION */
  std::shared_ptr<const lf::mesh::Mesh> mesh = dofhandler.Mesh();
  for (const auto& cell : mesh->Entities(0)) {
    // check if we the FE space is really S_1^0
    if (dofhandler.NoLocalDofs(cell) != 3) {
      throw "Not a S_1^0 FE space!";
    }
    // iterate over dofs
    auto int_dofs = dofhandler.GlobalDofIndices(cell);
    for (auto dof_idx_p = int_dofs.begin(); dof_idx_p < int_dofs.end();
         ++dof_idx_p) {
      // local integral of the basis function associated with this dof:
      // in linear Lagrangian FE, the integral over the basis functions
      // over a triangle K is:  1/3*vol(K)
      const double I_bary = 1.0 / 3.0 *
          lf::geometry::Volume(*cell.Geometry());
      // multiply by the value at the dof to get local contribution
      I += I_bary * mu(*dof_idx_p);
    }
  }
  /* END_SOLUTION */
  return I;
}
```

(4) f)

Write a C++ function

```
double integrateQuadraticFEFunction(
  const lf::assemble::DofHandler &dofhandler,
  const Eigen::VectorXd &mu);
```

that computes $\int_\Omega u_h \,dx$ for $u_h \in \mathcal{S}_2^0(\mathcal{M})$, whose nodal basis expansion coefficients are passed in `mu`. The `dofhandler` arguments provides the local→global index mapping for the quadratic Lagrangian finite element space. Check whether the **lf::assemble::DofHandler** object `dofhandler` really fits the Lagrangian finite element space $\mathcal{S}_2^0(\mathcal{M})$.

Use the formula given in [Lecture → Lemma 2.7.5.5] to compute the integrals of the local shape functions [Lecture → Eq. (2.6.1.6)] over a single triangular cell.

Need $\int_K b_K^j \,dx = \begin{cases} 0, & j = 1,2,3 \\ \frac{1}{3}|K|, & j = 4,5,6 \end{cases}$ (Lemma 2.7.5.5)

C++11 code 2.9.5: Sub-problem (2-9.f)Implementation of `integrateQuadraticFEFunction()` → GITLAB

```
2  double integrateQuadraticFEFunction(const lf::assemble::DofHandler&
       dofhandler,
3                                      const Eigen::VectorXd& mu) {
4    double l = 0;
5    /* BEGIN_SOLUTION */
6    std::shared_ptr<const lf::mesh::Mesh> mesh = dofhandler.Mesh();
7    for (const auto& cell : mesh->Entities(0)) {
8      // check if we the FE space is really S_2^0
9      if (dofhandler.NoLocalDofs(cell) != 6) {
10       throw "Not a S_2^0 FE space!";
11     }
12     const double weight = 1.0 / 3.0 * lf::geometry::Volume(*cell.Geometry());
13     // iterate over dofs
14     auto int_dofs = dofhandler.GlobalDofIndices(cell);
15     // The integrated basis functions associated with the nodes are 0:
16     // ∫_K b_K^j dx = 0 for j = 1,2,3.  Skip!
17     for (int l = 3; l < 6; ++l) {
18       // The integrated basis functions associated with the edges are:
19       // ∫_K b_K^j dx = |K|/3 for j = 4,5,6
20       // multiply by the value at the dof to get local contribution
21       l += (weight * mu(int_dofs[l]));
22     }
23   }
24   /* END_SOLUTION */
25   return l;
26 }
```

⑤ q_1 $S_1^0(\mathcal{M}) \subset S_2^0(\mathcal{M})$

$\forall u_h \in S_1^0(\mathcal{M})$ posses an expansion in the basis of $S_2^0(\mathcal{M})$

By definition $\mathcal{S}_1^0(\mathcal{M}) \subset \mathcal{S}_2^0(\mathcal{M})$. Hence, every $u_h \in \mathcal{S}_1^0(\mathcal{M})$, which may be given through its basis expansion coefficient vector $\vec{\mu} \in \mathbb{R}^{\sharp\mathcal{V}(\mathcal{M})}$, can be written as a linear combination of nodal basis functions of $\mathcal{S}_2^0(\mathcal{M})$ with an associated coefficient vector $\vec{\zeta} \in \mathbb{R}^{\sharp\mathcal{V}(\mathcal{M})=\sharp\mathcal{E}(\mathcal{M})}$.
Devise a C++ function

```
Eigen::VectorXd convertDOFsLinearQuadratic(
    const lf::assemble::DofHandler &dofh_Linear_FE,
    const lf::assemble::DofHandler &dofh_Quadratic_FE,
    const Eigen::VectorXd &mu);
```

$\Rightarrow$ return $\vec{\eta}$

that computes $\vec{\zeta}$ from $\vec{\mu}$ passed in `mu`. The two **lf::assemble::DofHandler** objects belong to the finite element spaces $\mathcal{S}_1^0(\mathcal{M})$ and $\mathcal{S}_2^0(\mathcal{M})$, respecively (This should be checked in the code).

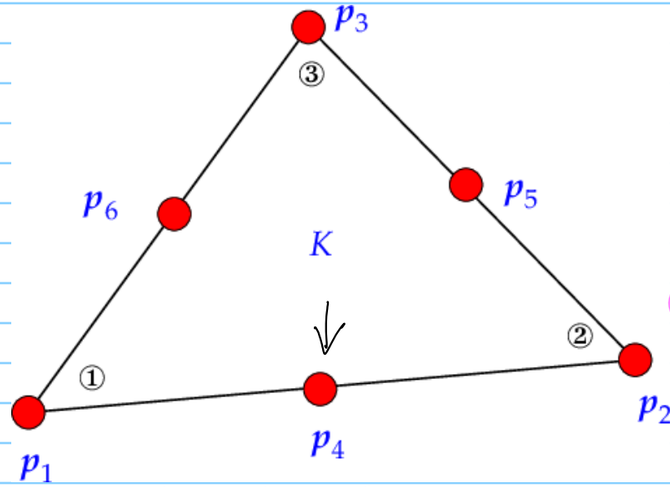
$$\forall q_h \in S_2^0(\mathcal{M}): \quad q_h = \sum_{j=1}^{\dim S_2^0(\mathcal{M})} q_h(p_j)\, b_h^j$$

$q_h(p_j)$: global interpolation node; b_h^j: GSF

$$u_h|_K = \sum_{\ell=1}^{3} u_h(a_K^\ell)\, \lambda_\ell$$

We have to find the values of u_h in the interpolation nodes for $S_2^0(\mathcal{M})$

We adopt a local perspective



$u_h(p_\ell) = u_h(a_K^\ell)$, $\ell = 1,2,3$

$u_h(p_4) = \frac{1}{2}(u_h(a_K^1) + u_h(a_K^2))$

On K $\iota_1, \iota_2, \iota_3$ are global idx's of $S_1^0(\mathcal{M})$-LSF

$j_1, \dots, j_6$ are global idx's of $S_2^0(\mathcal{M})$-LSF

$\triangleright$ $(\vec{\eta})_{j_1} = u_h(a_K^1) = (\vec{\mu})_{\iota_1}$

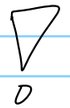
$$(\vec{\eta})_{j_\ell} = \sum_{j=1}^{3} (\vec{\mu})_{i_j} \underbrace{\lambda_j(p_K^\ell)}_{=\delta_{j,\ell}} = (\vec{\mu})_{i_\ell}, \quad \ell = 1,2,3$$

$$(\vec{\eta})_{j_4} = \sum_{j=1}^{3} (\vec{\mu})_{i_j} \underbrace{\lambda_j(p_K^4)}_{\in\{0,\frac{1}{2}\}} = \frac{1}{2}\left((\vec{\mu})_{i_1} + (\vec{\mu})_{i_2}\right),$$

$$(\vec{\eta})_{j_5} = \sum_{j=1}^{3} (\vec{\mu})_{i_j} \underbrace{\lambda_j(p_K^5)}_{\in\{0,\frac{1}{2}\}} = \frac{1}{2}\left((\vec{\mu})_{i_2} + (\vec{\mu})_{i_3}\right),$$

$$(\vec{\eta})_{j_6} = \sum_{j=1}^{3} (\vec{\mu})_{i_j} \underbrace{\lambda_j(p_K^6)}_{\in\{0,\frac{1}{2}\}} = \frac{1}{2}\left((\vec{\mu})_{i_3} + (\vec{\mu})_{i_1}\right).$$

(6)



Do not sum the values.