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

Numerical Methods for Partial Differential Equations

Prof. R. Hiptmair, SAM, ETH Zurich

Spring Term 2020

Q & A Session

May 25, 2020

Flagging boundary entities in Lehr FEM++

(i) Treatment of Dirichlet BVPs

(ii) Assembly of boundary contributions to (bi-) linear forms.

→ HW 6.6 :

$$a(u,v) = \int_\Omega \operatorname{grad} u \cdot \operatorname{grad} v \, dx + \int_{\partial\Omega} uv \, dS$$

Galerkin discretization based on $S^0_1(\mathcal{M})$

Tool : flagEntitiesOnBoundary

②

`CodimMeshDataSet< bool > lf::mesh::utils::flagEntitiesOnBoundary ( const std::shared_ptr< const Mesh > & mesh_p, lf::base::dim_t codim )`

flag entities of a specific co-dimension located on the boundary

=1 ≙ edges

Parameters

codim co-dimension of entities to be flagged, must be > 0.

Returns

an object of a boolean-valued CodimMeshDataSet (= an array of boolean values index by entities) for the entities of the specified co-dimension

An entity of co-dimension 1 is located on the boundary, if it is adjacent to exactly 1 cell (= entity of co-dimension 0).

The boundary of a mesh is the set of all entities that are either entities of co-dimension 1 located on the boundary or sub-entities of those.

The implementation of this function relies on CountNumSuperEntities().

The following example code shows how to create a flag array for marking the boundary edges of a 2D mesh:

```
std::shared_ptr<lf::mesh::Mesh> mesh_p{
   lf::mesh::test_utils::GenerateHybrid2DTestMesh(3)};
lf::mesh::utils::CodimMeshDataSet<bool> bd_flags{
   lf::mesh::utils::flagEntitiesOnBoundary(mesh_p, 1)};
```

not needed

$$a(u,v) = \int_\Omega \operatorname{grad} u \cdot \operatorname{grad} v \, dx + \int_{\partial\Omega} u v \, dS$$

C++ code 6.6.10: Implementation of assembly for matrix $\mathbf{M}$ ↪ GITHUB

```
lf::assemble::COOMatrix<double> buildM(
    const std::shared_ptr<lf::uscalfe::FeSpaceLagrangeO1<double>> &fe_space_p) {
  // I. TOOLS AND DATA
  // Pointer to current fe_space and mesh
  std::shared_ptr<const lf::mesh::Mesh> mesh_p(fe_space_p->Mesh());
  // Obtain local->global index mapping for current finite element space
  const lf::assemble::DofHandler &dofh{fe_space_p->LocGlobMap()};
  // Dimension of finite element space
  const lf::uscalfe::size_type N_dofs(dofh.NumDofs());

  // II : ASSEMBLY
  // Matrix in triplet format holding Galerkin matrix, zero initially.
  lf::assemble::COOMatrix<double> M(N_dofs, N_dofs);
  // Obtain an array of boolean flags for the edges of the mesh, 'true'
  // indicates that the edge lies on the boundary
  auto bd_flags{lf::mesh::utils::flagEntitiesOnBoundary(mesh_p, 1)};
  // Creating a predicate that will guarantee that the computations are carried
  // only on the edges of the mesh using the boundary flags
  auto edges_predicate = [&bd_flags](const lf::mesh::Entity &edge) -> bool {
    return bd_flags(edge);
  };
  // Coefficient function used in the class template MassEdgeMatrixProvider
  auto eta =
      lf::mesh::utils::MeshFunctionGlobal([](Eigen::Vector2d x) -> double { return
          1.0; });
  lf::uscalfe::MassEdgeMatrixProvider<double, decltype(eta),
                                      decltype(edges_predicate)>
      edgemat_builder(fe_space_p, eta, edges_predicate);
  // Invoke assembly on edges by specifying co-dimension = 1
  lf::assemble::AssembleMatrixLocally(1, dofh, dofh, edgemat_builder, M);
  return M;
};
```

→ a predicat on the set of edges

also possible: bd_flags

skips entities for which predicate returns false

③

CodimMeshDataSet (std::shared_ptr< const **Mesh** > mesh, **dim_t** codim)
construct data vector indexed by entities of a particular co-dimension More...

template<class = typename std::enable_if< std::is_copy_constructible<T>::value>::type>

CodimMeshDataSet (std::shared_ptr< const **Mesh** > mesh, **dim_t** codim, T init)
construct and initialize data vector indexed by entities of a particula co-dimension More...

T & **operator()** (const **Entity** &e)
Get a (modifiable) reference to the data stored with entity e. More...

const T **operator()** (const **Entity** &e) const override
Get the data stored with entity e. More...

bool **DefinedOn** (const **Entity** &e) const override
Does the dataset store information with this entity? More...

→ If T = bool → CodimMeshDataSet object is predicate over the set of entities already!