FESpace classes in LifeV

Luca Bertagna, Alberto Ferroni, Iori Guido, Simone Stangalino

Politecnico di Milano - LifeV Meeting 2014

January 7th, 2014

Outline

FESpace

FESpace contains:

- map of d.o.f (class DOF),
- quadrature rule for the elements and the facets (class QuadratureRule)
- reference element (class ReferenceFE),
- a finite element for computations of local matrices (class CurrentFE)
- interpolation methods.
- methods for computation of L^2 and H^1 norms of solution and error (if the analytical solution is given).

Declaration of reference elements and quadrature rules in FEDefinitions.cpp.

The DOF class

The main purpose of this class it to provide data structures containing information about DOFs (Degrees Of Freedom):

- a map to identify the DOFs given the local ID of an element,
- a map to identify the boundary DOFs given the local ID of a boundary facet.

These data structures are instances of `ArraySimple` class, a sort of matrix (m, n) where columns correspond to an element (or facet) and rows to the DOFs (sorted by geometrical entity related to the DOF):

$$\begin{pmatrix} & \text{element } 1 & \cdots & \text{element } n \\ \text{DOF } 1 & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ \text{DOF } m & \cdots & \cdots & \cdots \end{pmatrix}$$

ETFESpace

```
template<typename MeshType, typename MapType, UInt SpaceDim, UInt
      FieldDim>
class ETFESpace
{
public:
const static UInt S_spaceDim;
const static UInt S_fieldDim;

    // constructors, get methods...

private:
void createMap (const commPtr_Type& commptr);

    meshPtr_Type M_mesh;
    const ReferenceFE* M_referenceFE;
    const GeometricMap* M_geometricMap;
    DOF* M_dof;
    MapType* M_map;
};
```

FESpace vs ETFESpace

Main differences

- ETFESpace templetized on space and field dimensions
- no CurrentFE
- no interpolation routines
- no norm and error routines
- any other?

Main question

Is it possible to substitute FESpace with ETFESpace?

FESpacemanifold

FESpaceManifold is a **derived** class of FESpace that defines a finite element space for PDEs defined on a manifold Γ embedded in $\mathbb{R}^3$.

At the moment in LifeV (not in master!) it is possible to compute the following bilinear forms on a manifold

- stiffness
$$\int_{\Gamma_h} \mu \nabla_{\Gamma_h} u \cdot \nabla_{\Gamma_h} v \, d\sigma_h$$
- advection
$$\int_{\Gamma_h} \boldsymbol{\beta} \cdot \nabla_{\Gamma_h} u \, v \, d\sigma_h$$
- mass
$$\int_{\Gamma_h} u \, v \, d\sigma_h$$
- gradient
$$-\int_{\Gamma_h} p \, \nabla_{\Gamma_h} \cdot \mathbf{v} \, d\sigma_h$$
- divergence
$$\int_{\Gamma_h} \nabla_{\Gamma_h} \cdot \mathbf{u} \, q \, d\sigma_h$$

FESpace vs FESpaceManifold

- FESpace class

```
boost::shared_ptr<CurrentFE> M_fe;
boost::shared_ptr<CurrentFEManifold> M_feBd;
```

- FESpaceManifold class

```
boost::shared_ptr<CurrentFEManifold> M_feManifold;
```

then in the constructor

```
FESpaceManifold<MeshType,MapType>::FESpaceManifold (...) :
    FESpace<MeshType,MapType>(mesh,space,fDim,comm)
{
    this->M_fe.reset();
    M_feManifold.reset(new CurrentFEManifold(*this->M_refFE,
        getGeometricMap(*mesh),*this->M_Qr));
    this->M_fe = M_feManifold;
}
```

Issues

- What if i want a different DOF map?
  - templetize `FESpace` on `DOF` type class
    Pro: (somewhat) faster. Cons: compilation time.
  - inheritance for `DOF` classes and pointer in `FESpace`
    Pro: well defined interface. Cons: dynamic dispatching.
- How to have different `CurrentFE` in the same `FESpace`?
- Should the interpolation methods be moved outside of the class (maybe in a namespace)?
- How about the error methods? After all they're just integration routines.
- Can we have a unique FESpace class? Note: this might require to template FESpace on the field dimension

How use a different type of ReferenceFE

Before

```
setSpace ( space, mesh_Type::S_geoDimensions );

M_dof.reset ( new DOF ( *M_mesh, *M_refFE ) );
M_dim = M_dof->numTotalDof();
M_fe.reset ( new CurrentFE ( *M_refFE, getGeometricMap ( *M_mesh ), *
      M_Qr ) );
```

After

```
M_Qr=...; M_bdQr =...;
const ReferenceDGElement* feDGElement= new ReferenceDGElement("
     feDGElement" , shape, NbDof, geoDim, 1, phi, dphi, refCoor,
     NbQuadNodesVolume, NbQuadNodesFacet);
M_refFEDG = feDGElement;

  M_dof.reset( new DOF( *M_mesh, *M_refFEDG ) );
  M_dim = M_dof->numTotalDof();
  M_fe.reset( new CurrentFE( *M_refFEDG, getGeometricMap( *M_mesh ), *
       M_Qr, getGeometricMap( *M_mesh ).boundaryMap() , *M_bdQr ) );
```