Non Matching Grids in a Distributed Setting

Giovanni Alzetta
prof. Luca Heltai

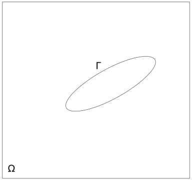
6^{th} deal.II Users and Developers Workshop
- Trieste -

Outline

Topics

- Motivation: Model problem (step-60)
- Coupling Matrix
- Distributed Approach (on-going effort)

Example: Multiphysics Coupling



Domains

Consider the two domains with appropriate regular hypothesis:

- Ω in R^2
- Γ in R^1

such that $\Gamma \subset \Omega$.

The considered PDE

Consider $V := H^1_0(\Omega)$ and $Q := H^{1/2}(\Gamma)$ and the trace operator $\gamma : H^1_0(\Omega) \mapsto H^{1/2}(\Gamma)$.

Strong form

For $g \in Q$, find $(u, \lambda) \in V \times Q^*$ such that:

$$\begin{aligned} -\Delta u + \gamma^T \lambda &= 0 \text{ in } \Omega \\ \gamma u &= g \text{ in } \Gamma \\ u &= 0 \text{ on } \partial\Omega. \end{aligned}$$

where λ is a lagrange multiplier.

Weak form

For $g \in Q$, find $(u, \lambda) \in V \times Q^*$ such that:

$$\begin{aligned} (\nabla u, \nabla v)_\Omega + \langle \lambda, \gamma v \rangle_\Gamma &= 0 && \forall v \in V \\ \langle q, \gamma u \rangle_\Gamma &= \langle q, g \rangle_\Gamma && \forall q \in Q^* \end{aligned}$$

FE Discretization

Consider $V_h = \text{span}\{v_i\}_{i=1}^{n}$ discretization of V and $Q_h = \text{span}\{q_\alpha\}_{\alpha=1}^{m}$ discretization of Q.

Discretized form

For $g_h \in Q_h$, find $(u_h, \lambda_h) \in V_h \times Q_h^*$ such that:

$$\begin{aligned} (\nabla u_h, \nabla v_h)_{V_h} + \langle \lambda_h, \gamma v_h \rangle_{Q_h} &= \quad 0 & \forall v_h \in V_h \\ \langle q_h, \gamma u_h \rangle_{Q_h} &= \quad \langle q_h, g_h \rangle_{Q_h} & \forall q_h \in Q_h^* \end{aligned}$$

$$\begin{pmatrix} K & C^T \\ C & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ G \end{pmatrix}$$

$$\begin{aligned} K_{ij} &:= (\nabla v_j, \nabla v_i)_\Omega & i, j = 1, \dots, n \\ C_{\alpha j} &:= \langle q_\alpha^*, \gamma v_j \rangle_{Q_h} & j = 1, \dots, n, \alpha = 1, \dots, m \\ G_\alpha &:= \langle q_\alpha^*, g_h \rangle_{Q_h} & \alpha = 1, \dots, m. \end{aligned}$$

Numerical Coupling

$$
\begin{aligned}
C(q_\alpha^*, \gamma v_j) &:= \langle q_\alpha^*, \gamma v_j \rangle_{Q_h} \\
&= \int_\Gamma \int_\Gamma q_\alpha^* \gamma v_i \mathrm{d}\Gamma \mathrm{d}\Gamma \\
&= \sum_{K \in \mathcal{T}(\Gamma)} \int_{K \cap \mathrm{supp}(v_j)} q_\alpha(x) v_j(x) \mathrm{d}x.
\end{aligned}
$$

where $\mathcal{T}(\Gamma)$ is the triangulation of *Gamma*.

General Problem

C has an intrinsic problem: it involves both triangulations V_h and Q_h.

- Find the cells of $K \in \mathcal{T}(\Gamma)$ such that $K \cap \mathrm{supp}(v_j)$
- Let $v_j := F \circ \hat{v}_j$; then compute F^{-1} and restrict it to Γ.

Serial and Shared case in deal.II

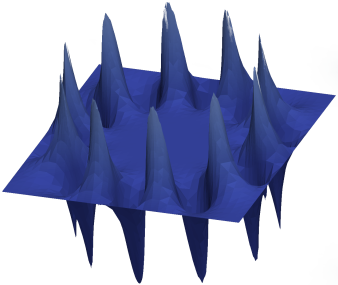
In deal.II 9.0 we have the following functions:

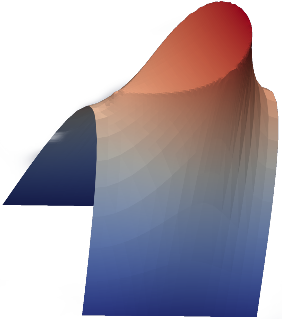
$$NonMatching :: create\_coupling\_sparsity\_pattern$$
$$NonMatching :: create\_coupling\_mass\_matrix$$

It currently computes the coupling matrix for:

- serial case
- with a shared immersed triangulation

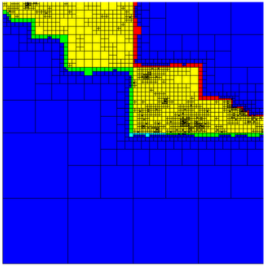
Step 60





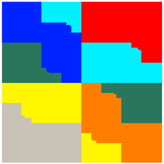
Extra: the tutorial explains how to use the **Parameter Acceptor**.

Parallel Distributed Meshes



- Owned cells (we know everything)
- Ghost cells (we know everything...at least about the triangulation)
- Artificial cells

Non-matching grids



$$C(q_\alpha^*, \gamma v_j) = \sum_{K \in \mathcal{T}(\Gamma)} \int_{K \cap \mathrm{supp}(v_j)} q_\alpha^*(x) v_j(x) \mathrm{d}x.$$

Computation of C

Consider $v_j(x)$ on a process, if it lies:

1. in a locally owned cell or a ghost cell: integration can be performed
2. in an artificial cell: data is missing

Approaches

Existing Approaches

- Use constraints to build the partitions
- Use another mesh decomposition e.g. rendezvous decomposition (as in DTK, Data Toolkit)

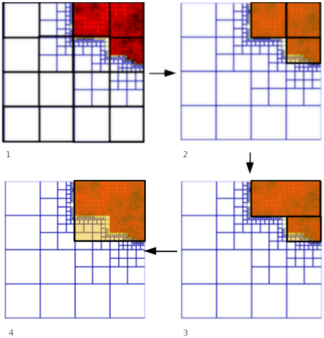
We want to use distributed Meshes with independent decomposition $\Rightarrow$ non-spatially conforming domain.

Proposed Approach

Use Bounding Boxes to identify the owner:

- Lightweight
- Flexible

Current Approach



Limitations

- Speed (both at build and search time)
- Results need manual tinkering and are not always good

Exchanging Objects

Sending Objects with MPI

- Manual de-construction and reconstruction
- Custom MPI Types
- Use char buffers through boost library (needs a serialize function)

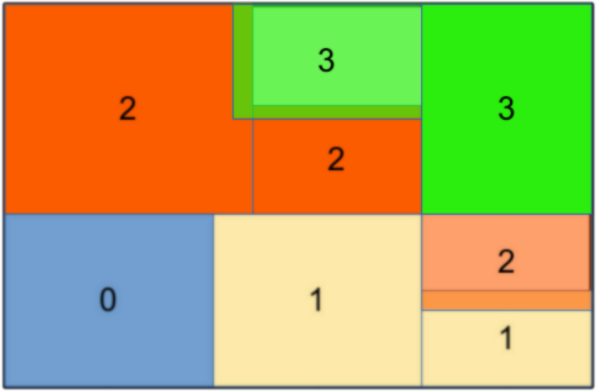
Implemented functions in deal.II *Utilities MPI*:

Send and Receive

Implemented an **all gather** and a **some to some** function:

- Work with arbitrary objects (as long as serialize is implemented)
- Code is cleaner, shorter and more readable
- Avoid many bugs' sources

Bounding Box Description



Usage Example

```
// Computing bounding boxes describing the locally owned part of the mesh
IteratorFilters::LocallyOwnedCell locally_owned_cell_predicate;
std::vector<BoundingBox<dim>>     local_bbox =
  GridTools::compute_mesh_predicate_bounding_box(
    cube_d, locally_owned_cell_predicate);

// Obtaining the global mesh description through an all to all communication
std::vector<std::vector<BoundingBox<dim>>> global_bboxes;
global_bboxes = Utilities::MPI::all_gather(mpi_communicator, local_bbox);
//LinearAlgebra::distributed::Vector<double> data_d(dof_handler_d.n_dofs());

IndexSet ghost;
DoFTools::extract_locally_relevant_dofs   (dof_handler_d,ghost);

LinearAlgebra::distributed::Vector<double> data_d_ghosted(dof_handler_d.locally_owned_dofs(), ghost, mpi_communicator);
data_d_ghosted.zero_out_ghosts();

VectorTools::interpolate(dof_handler_d, f_cosine, data_d_ghosted);
data_d_ghosted.update_ghost_values();

//  Functions::FEFieldFunction<dim>
Functions::FEFieldFunction<dim,
                           DoFHandler<dim>,
                           LinearAlgebra::distributed::Vector<double>>
  fe_function_d(dof_handler_d,
                data_d_ghosted,
                StaticMappingQ1<dim, dim>::mapping,
                true);


fe_function_d.set_up_bounding_boxes(global_bboxes);
```

Summary

What we have

- Implemented a serial algorithm
- Implemented the basic framework needed for the parallel case

What's in the future

- Suggestions on the implementation?
- Complete FEFieldFunction $\Rightarrow$ Coupling Matrix
- Improve the search using Axis Aligned Bounding Boxes (AABB). **(Aim: re-using work of others i.e. using libraries)**
- Use the implementation and and look for bottlenecks in the final simulations...