HAL
open science

Mesh generation for fusion applications

Herve Guillard, Jalal Lakhlili, Alexis Loyer, Ahmed Ratnani, Eric Sonnendrücker

To cite this version:

Herve Guillard, Jalal Lakhlili, Alexis Loyer, Ahmed Ratnani, Eric Sonnendrücker. Mesh generation for fusion applications. IPP Ringberg Theory Meeting, Nov 2018, Munich, Germany. hal-01950388

HAL Id: hal-01950388

https://inria.hal.science/hal-01950388v1

Submitted on 10 Dec 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

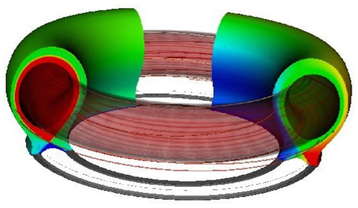
Mesh generation for fusion applications

Hervé Guillard[1], Jalal Lakhlili[2]⋆, Alexis Loyer[1], Ahmed Ratnani[2], Eric Sonnendrücker[2].

[1] Inria Sophia-Antipolis, Nice, France.
[2] Max-Planck-Institut für Plasmaphysik, Garching, Germany.

Context and motivations

- High anisotropy in magnetized fusion plasmas:
  ⇒ requires the use of **flux aligned meshes**.
- Complex and realistic geometries:
  ⇒ need other strategies (equidistribution, orthogonality).
- High order derivatives (in MHD for example):
  ⇒ require **regular representation**.

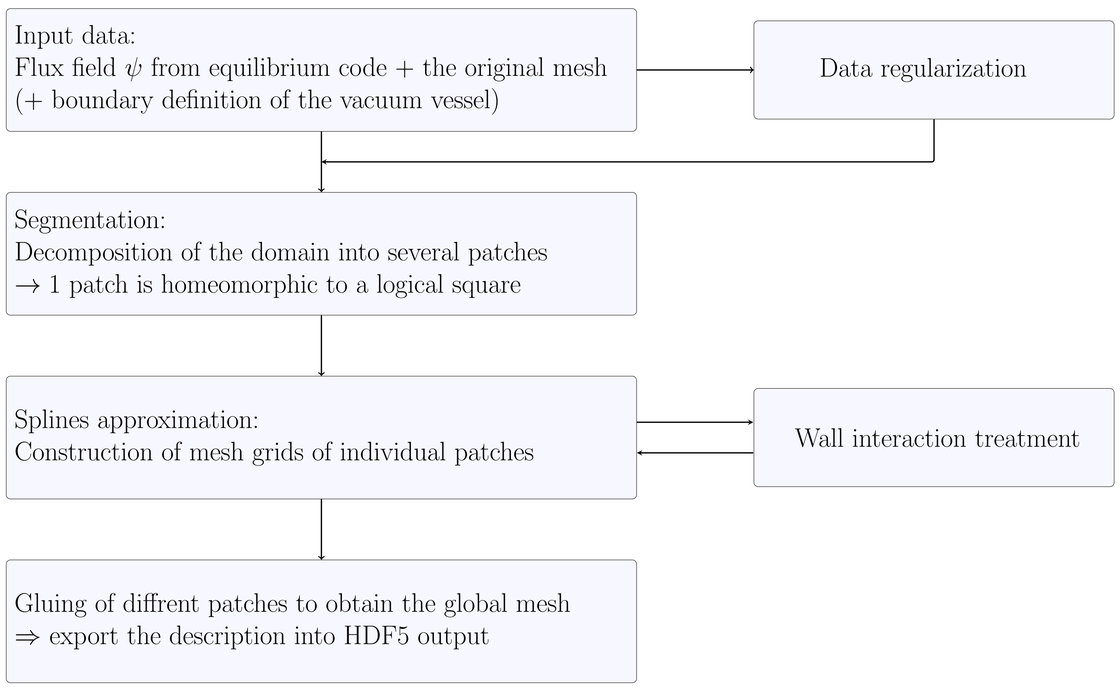


Sketch of a Tokamak reactor

Unified code for the generation of flux aligned mesh in the **poloidal plane**.

- Different codes and type of meshes.
- Different numerical methods:
  - Semi-Lagrangian approaches.
  - Finite difference, Finite volume
  - Finite Element (Spline or Hermite-Bezier on quadrangles; Powell-Sabin, Clough-Tocher on triangles, etc..)

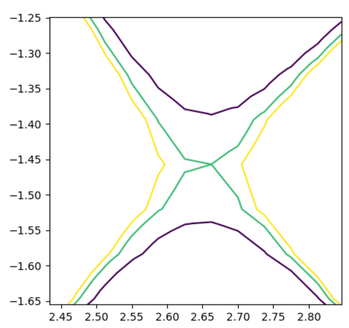
Workflow diagram

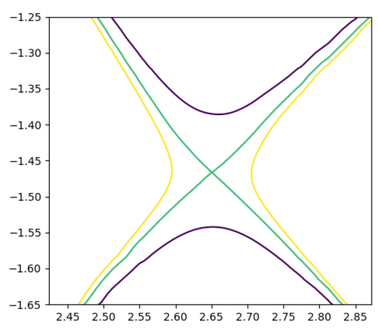


Pre-processing of the data

Isolines regularization

If the equilibrium solver results are not smooth enough : low-order discretization, coarse resolution, ..
⇒ Replace the flux function by its Clough Tocher interpolant on the refined mesh (locally).

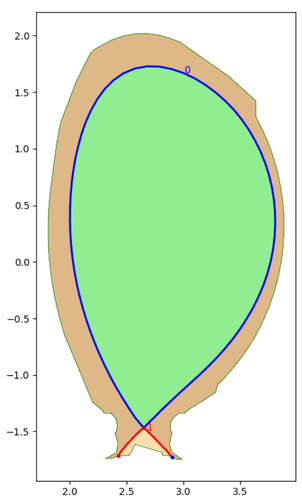


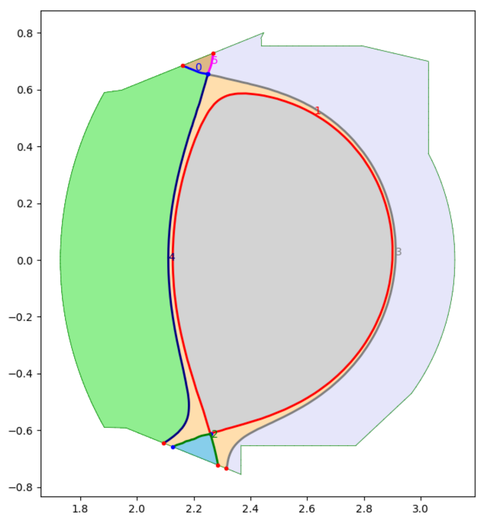


Segmentation

- Based on Morse theory and Reeb graph.
- Automatic decomposition of isolines including arbitrary number of X-Points.
- Topological set of the isolines consists of finite connected components and contains only closed orbits.

Examples of domain segementation for JET and WEST tokamaks:

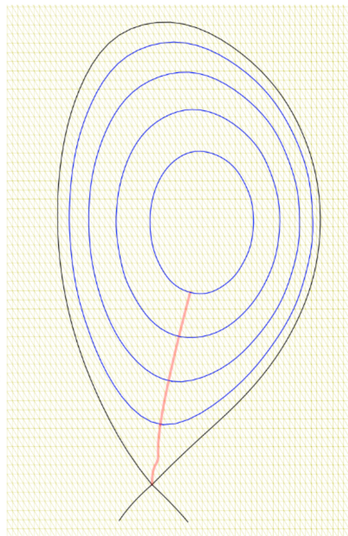




Structured grid construction

I. Isolines generation

1. Identify a mapping: $(s,t) \in [0,1]\times[0,1] \to \mathbf{x}(s,t) \in \Omega$
2. Get the radial curve $\mathscr{S}(s)$, solution of the ODE:
$$\begin{cases} \dfrac{d\mathbf{x}}{ds} = \dfrac{\psi_M - \psi_0}{\|\nabla\psi\|^2}\nabla\psi \\ \mathbf{x}(0) = \mathbf{x}_0 \end{cases}$$
3. Compute the spline interpolation of the radial curve and generate a set of internal isolines.



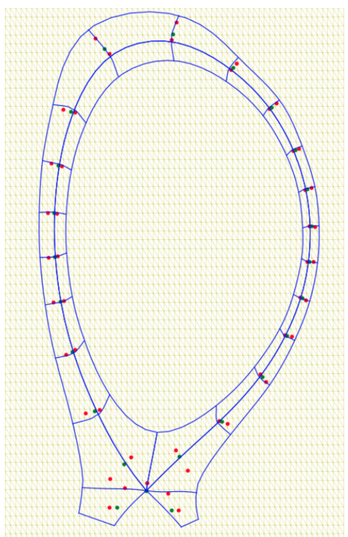
II. Meshing of individual patch Ω_k

1. Choose a finite number of isolines $f^{-1}(c_i)$.
2. Approximate each level set by a spline: $f^{-1}(c_i) \sim \mathscr{C}_i(t) = \sum_j \mathbf{C}^i_j N_j(t)$.
   N_j: Bspline Basis.
   $\mathbf{C}^i_j$: Control Points.
3. Construct a 2D tensor product mapping $[0,1]\times[0,1] \to \Omega_k$:
$$\mathcal{S}(s,t) = \sum_i\sum_j \mathbf{P}_{i,j}N_i(s)N_j(t), \text{ s.t. } \mathcal{S}(s_i,t) = \mathscr{C}_i(t), \forall i.$$

III. Gluing patches

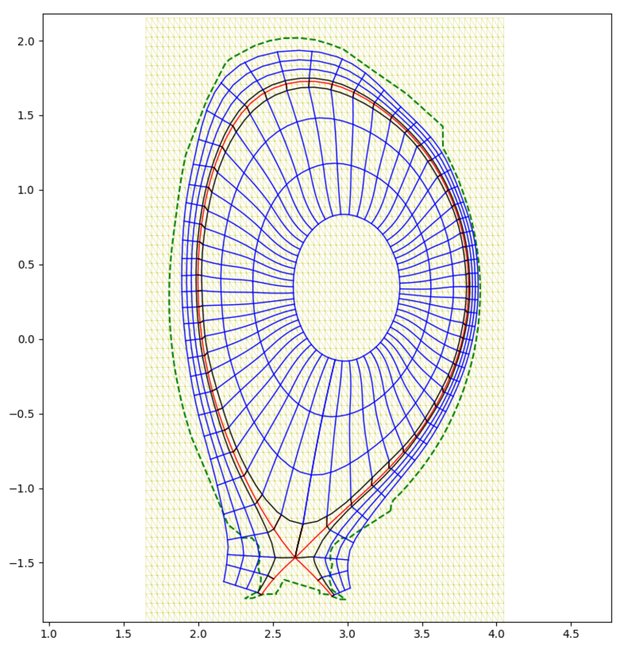
Given 2 subdomains Ω_1 and Ω_2 described by:
$$\mathcal{S}_1(s,t) = \sum_i\sum_j \mathbf{P}_{i,j}N_i(s)N_j(t)$$
$$\mathcal{S}_2(s,t) = \sum_i\sum_j \mathbf{Q}_{i,j}N_i(s)N_j(t)$$
with a common boundary for instance: $\mathcal{S}_1(0,t) = \mathcal{S}_2(0,t)$

1. $\mathcal{C}^0$ continuity: $\mathbf{P}_{0,j} = \mathbf{Q}_{0,j}, \forall j$.
2. $\mathcal{G}^1$ continuity: geometric condition on the control points, $\mathbf{P}_{1,j}, \mathbf{P}_{0,j} = \mathbf{Q}_{0,j}, \mathbf{Q}_{1,j}$ have to be aligned.



Multipatch example

Flux aligned mesh for JET (Core/Edge/SOL regions):



Summary

Development of a mesh generation software for tokamak simulations:

- Can be used by different codes and handle different types of meshes.
- Written in Python with Fortran bindings:
  → using open source libraries (except Segmentation module but free for academics).

⋆ jalal.lakhlili@ipp.mpg.de