Multi-patch

Alexander HOFFMANN, Pauline VIDAL

March 2024

Contents

1 Structure for multipatch

We are trying here to answer to the following questions:

- What kind of information do we need to store in each patch?
- What kind of information do the patches need to exchange?

1.1 Type of global space splitting

The splitting of the global space into patches is currently not fixed. However we can encounter three types of sticking:

- simple interface which sticks two whole edges together;
- T-joint which sticks several edges to one edge of a patch;
- X-point where the sticking is only on one point.

Each patch is defined on a logical domain and maps the mesh points to a physical domain (cf. Fig. 1). The figure Fig. 2 gives examples of the different stickings on the logical domain.

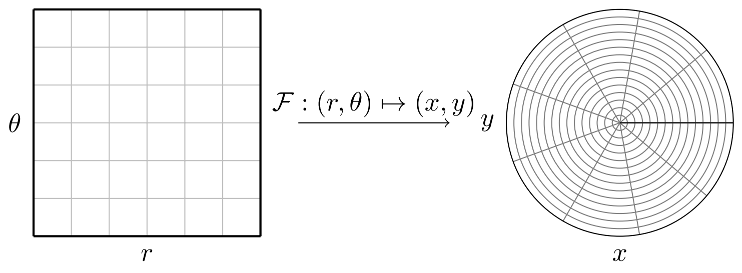
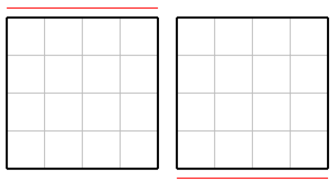
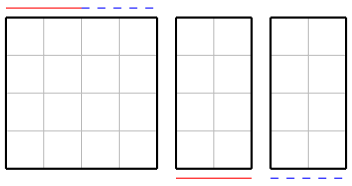
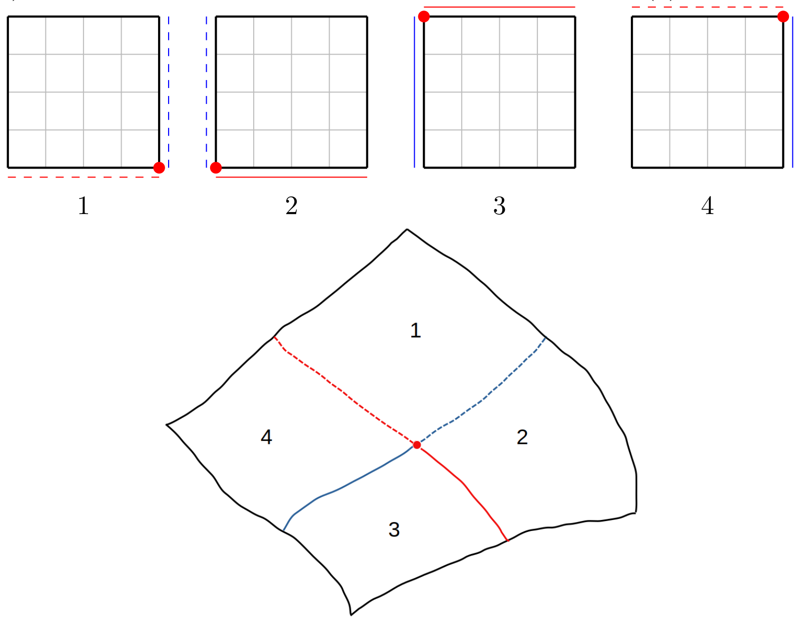


Figure 1: Mapping from the logical domain to the physical domain. Example with a circular mapping.


(a) Simple interface.


(b) T-joint.



(c) X-point in logical and physical domain.

Figure 2: Patches in the logical domain.

We want to stay general here and not to focus on one specific decomposition of the domain into patches. The figures Fig. 3 and 4 give examples of more complex spaces. These are not necessarily for the decomposition of the separatrix.

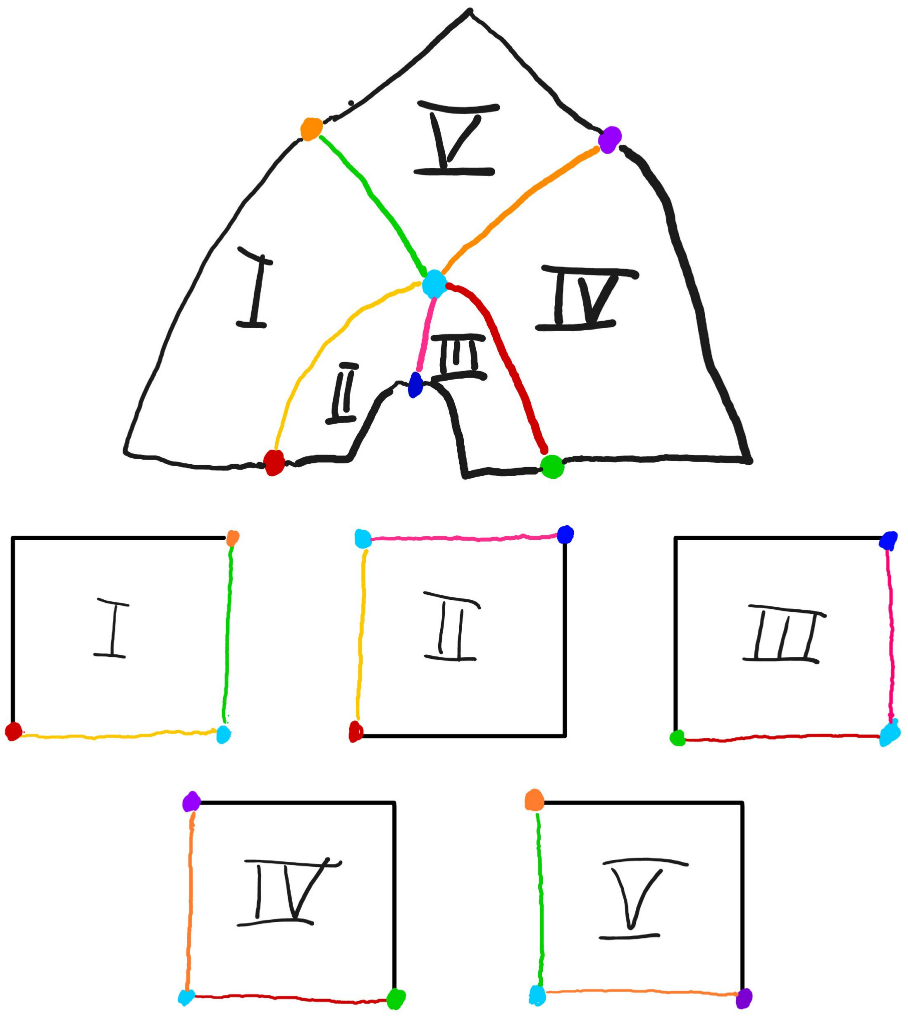


Figure 3: Simple sketch of more complicated geometry with X-point (cyan point). Physical edges and corners are identified on the logical patches. This illustrates the idea how we intend to represent the multipatch geometry using tensor-product patches.

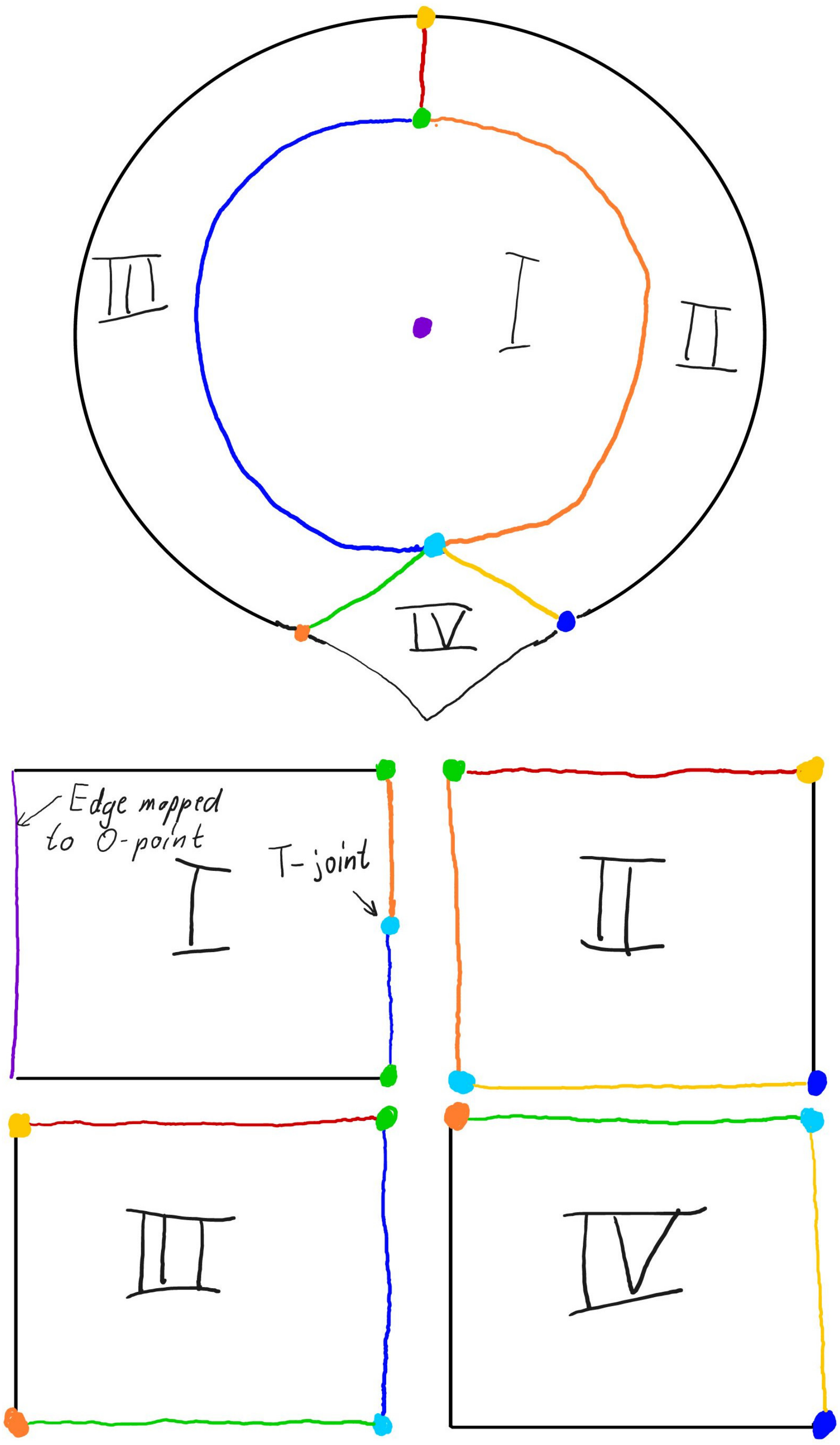


Figure 4: Quick sketch of geometry with O-point (purple point) and T-joint (cyan point). There is a T-joint at the cyan point because edges from patch II and III are connected to the eastern edge of patch I. Note that the y-dimension of patch I is periodic.

2 Advection equation

We focus on the advection and Poisson equation separately in the context of the questions raised in the introduction. We start with the advection equation.

In the guiding center equations example, the advection equation is given by

$$\partial_t \rho + A \cdot \nabla \rho = 0, \tag{1}$$

with A the advection field defined on the logical domain on the physical domain axes ($A(r,\theta) = [A_x(r,\theta), A_y(r,\theta)]$). Another example is the advection of the drift-kinetic model

$$\partial_t f + v_{GC} \cdot \nabla_\perp f + v_\parallel \partial_z f + \dot{v}_\parallel \partial_{v_\parallel} f = 0, \tag{2}$$

but it seems that the multi-patch problems are similar as in the case of the guiding center equations and so we focus on (1) only.

Computing the advection field. The advection field is computed from the solution of the Poisson equation. In the guiding center equations example,

$$A = -\nabla \phi \wedge e_z, \tag{3}$$

this computation can be done locally.

Store	• Spline representation (and values?) of ϕ. • Spline representation (and values?) of A. • Jacobian matrix.

Solving the characteristic equation. The equation to solve is

$$\begin{cases} \partial_t X(t;s,x) = A(t, X(t;s,x)), \\ X(s;s,x) = x. \end{cases} \tag{4}$$

we compute

$$\begin{cases} \partial_t X(t^n;t^{n+1},x) = A(t, X(t^n;t^{n+1},x)), \\ \hat{X}(t^n;t^{n+1},x) = \bar{\psi}(t^n;t^{n+1},x) \end{cases} \tag{5}$$

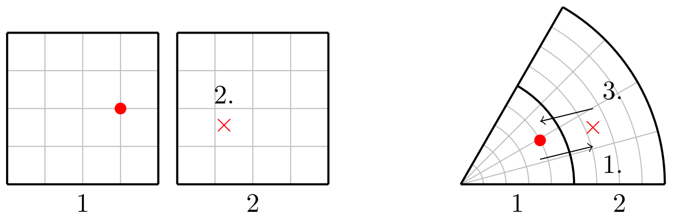
where $\bar{\psi}$ is a discrete flow computed with a time integration method. In the case of Runge-Kutta method, the advection field needs to be evaluated at intermediate feet.

Example of RK2.

$$\begin{aligned} X^1 &= X^0 - \frac{dt}{2} A(X^0), \\ X^2 &= X^0 - dt A(X^1). \end{aligned} \tag{6}$$

Outside the patch. In case of X^1 is outside the patch we are working on, we need to communicate with the other patches to evaluate the advection field. The same problem appears when we have solved the characteristic equation and want to evaluate the advected function ρ.

Exchange	• The outside feet (in the physical domain?). • The evaluated value of the advection field. • The evaluated value of the advected function.
Store	• Spline representation (and values?) of the advected function ρ. • Spline representation (and values?) of the advection field A.



1. compute feet.
2. evaluate function.
3. transfer value.

Figure 5: Example of a characteristic foot outside the patch 1 in the logical and physical domains.

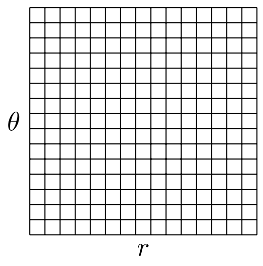
Further questions on outside feet. Problem: the mapping from the logical to the physical domain is not always easily invertible.

- For an advection in the physical domain, if the feet is outside of the domain, how to get the feet in the logical domain of another patch (inverting the mapping for the other patch might be costly)?
- For an advection in the pseudo-Cartesian domain, if the feet is outside of the domain, what are the equivalent coordinate in the physical domain?

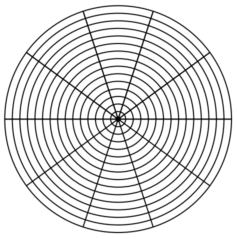
Two fundamental possibilities:

1. **Advect in physical domain**: If necessary, use control points of the spline mapping to find the patch and then invert the spline mapping to get logical coordinates (preferred by Eric),
2. **Advect in logical/pseudo-cartesian domain**: Extend the coordinates of the patch and find coordinate transformation to logical coordinates of neighboring patches (we do not know yet how to do this, it is just an idea).

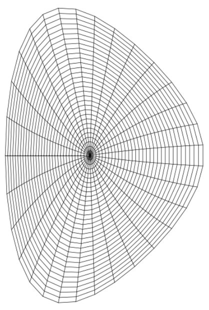
If the field lines do not cross the edge (ex. inside of the core), the characteristics do not cross the edge often and in these cases inverting a spline mapping might be feasible. Maybe a clever combination of using logical and physical coordinates in certain situations is the best approach?



(a) Logical domain.



(b) Pseudo-Cartesian domain.



(c) Physical domain.

- How to communicate the feet and values from a patch to another?

Two ideas:

1. Communicate the feet to a global space class; compute in which patch the feet are; evaluate the function at the feet in the right patch; communicate the feet to a global space class to redistribute the values to the first patch.
2. Add some ghost cells around each patch ; update the function values at each every time they change.

For the ghost cells option, we need to know in advance what will be the maximal advection length to not land outside of the ghost cells zone. (Is this length known?) To update the values in the ghost cells, we also need to evaluate the function on the mesh points of the ghost cells of other patches (the mesh points won't always coincide between two patches, especially if we want one more refined than the other).

For the first option, we need to compute in which patch the feet are (which can be non-trivial).

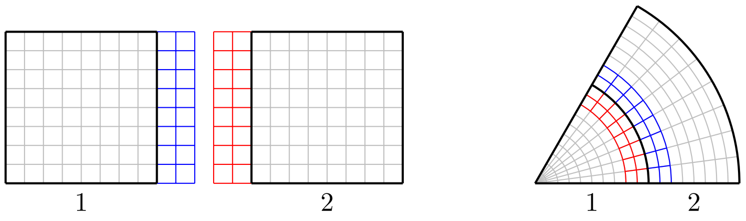


Figure 7: Ghost cells for the previous example.

Build a spline representation of the advected function. To do it, we

- Evaluate the spline representation of ρ at the characteristic feet [**local but need to communicate**];
- Compute the derivatives at the interfaces of each patch. To do it, there are different methods:
  - advect the derivatives [**local method**];
  - compute the derivatives by:
    - using Lagrange polynomials [**neighbor local (only implies 2 patches)**];
    - using global spline relation [**global method**].
- Build the new spline representation [**local**].

Exchange	• The mesh points around the interfaces for Lagrange interpolation. • The value of ρ around the interfaces for Lagrange interpolation. • The sum of values of the function $\sum_i \alpha_i s(x_i) = \sum_{p \in patches} \sum_{x_i \in p} \alpha_i s(x_i)$.
Store	• Spline representation of the advected function ρ.

2.1 Further questions

About evaluating functions.

- How do we know in which patch we are?
  - For a given point in the physical domain, how do we know in which subset it belongs to?
- Let's assume we know in which patch we are, how can we evaluate a function? How can we get $\rho(x, y)$ with (x, y) in the physical domain (especially when the mapping difficult to invert)?

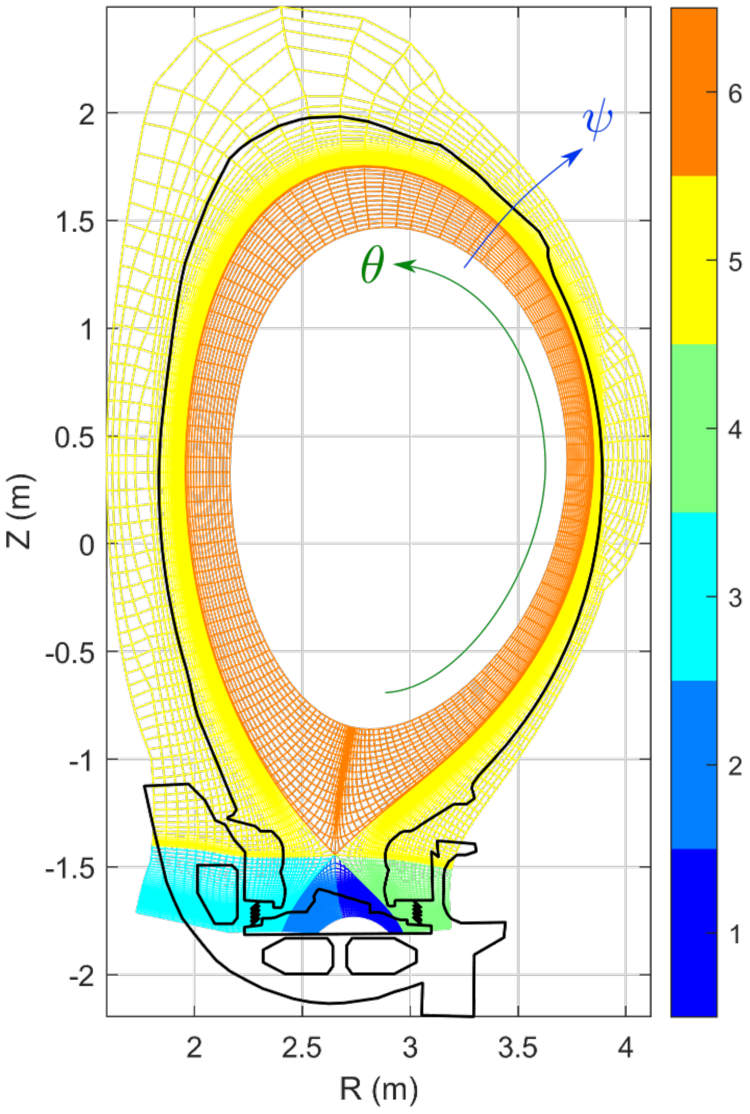


Figure 8: Multi-patch decomposition from SOLEDGE.

3 Poisson problem on multipatch

We want to solve the Poisson equation

$$-\operatorname{div}(\nu \operatorname{grad} \phi) = \rho$$

using a the CONGA approach on a multipatch domain.

The CONGA approach has the main advantage that it uses patch-local degrees of freedom and basis functions. The coupling of adjacent patches happens through the use of projection operators onto conforming subspaces. This approach has been studied by Martin Campos Pinto and Yaman Güçlü in the context of FEEC. The projection operators are implemented in Psydac and have been successfully applied to different problems in electromagnetics.

Very short introduction to the CONGA approach

This short section is just a very short introduction of the ideas in our context. For more details see Campos Pinto, Güçlü (2024) - Broken-FEEC discretizations and Hodge Laplace problems and Güçlü, Hadjout, Campos Pinto (2023) - A Broken FEEC Framework for Electromagnetic Problems on Mapped Multipatch Domains

We define finite element spaces V_h and V_h^c. We assume that our domain Ω is given as a disjoint union of patches Ω_k, $k = 1, ..., K$. Let $V_h(\Omega_k)$ be the finite element space on Ω_k for every k. Define the basis of V_h by taking the union of all basis functions of $V_h(\Omega_k)$ for all k and extending them by zero outside of their respective patch Ω_k. We recognize that the functions in V_h can have jumps across patch boundaries and are thus not continuous. Hence, we call V_h *broken.* Let V be a subspace of $L^2(\Omega)$ with functions fulfilling some regularity requirement e.g. $V = H^1(\Omega)$.

We further define the conforming subspace

$$V_h^c := V_h \cap V$$

i.e. the functions in V_h with appropriate regularity. We then define projection operators

$$P_h : V_h \to V_h^c.$$

and the discrete differential operator

$$\mathrm{grad}_h := \mathrm{grad}\ P_h.$$

For an operator A defined on a subspace of $L^2(\Omega)$ we denote the adjoint w.r.t. the L^2 inner product as A^*, if it exists. Then we define the CONGA Laplacian operator as $\mathrm{grad}_h^*(\nu\, \mathrm{grad}_h)$ and introduce a stabilized version via

$$\mathrm{grad}_h^*(\nu\, \mathrm{grad}_h) + \alpha(I - P_h^*)(I - P_h)$$

where $\alpha > 0$ is some constant and I is the identity. This is discretized weakly by discretizing the bilinear form $a_h : V_h \times V_h \to \mathbb{R}$

$$a_h(\phi_h, \psi_h) = \int_\Omega \nu\, \mathrm{grad}_h\, \phi_h \cdot \mathrm{grad}_h\, \psi_h\, dx + \int_\Omega (I - P_h)\phi_h\, (I - P_h)\psi_h\, dx$$

Applying the CONGA approach

- **Question:** Does the advection field **A** have to be continuous? Emily thinks so. Then we need to enforce global C^1-regularity on $\phi \to$ Use CONGA approach with C^1 conforming projection P_h
- The conforming projection is computed by taking suitable averages of spline coefficients close to the boundary of the patch so it only acts on splines close to the edge
- Need the mesh information from neighboring patches to construct the projection matrices
- Spline coefficients of ρ needed to assemble the right hand side
- Poisson solver needs to access all spline coefficients from all patches $\to$ need global managing of the splines, mesh etc to assemble everything
- **Question:** Do we make the assembly of the stiffness matrix completely global?

4 Mappings

- Mappings and corresponding patches are provided by the Tokamesh library (see Fig. 9).
- Tokamesh uses the flux function from a Grad-Shafranov solver to construct mappings and fit the patches accordingly to get flux aligned grids.
- C^1-conformity everywhere except close to the X-point.

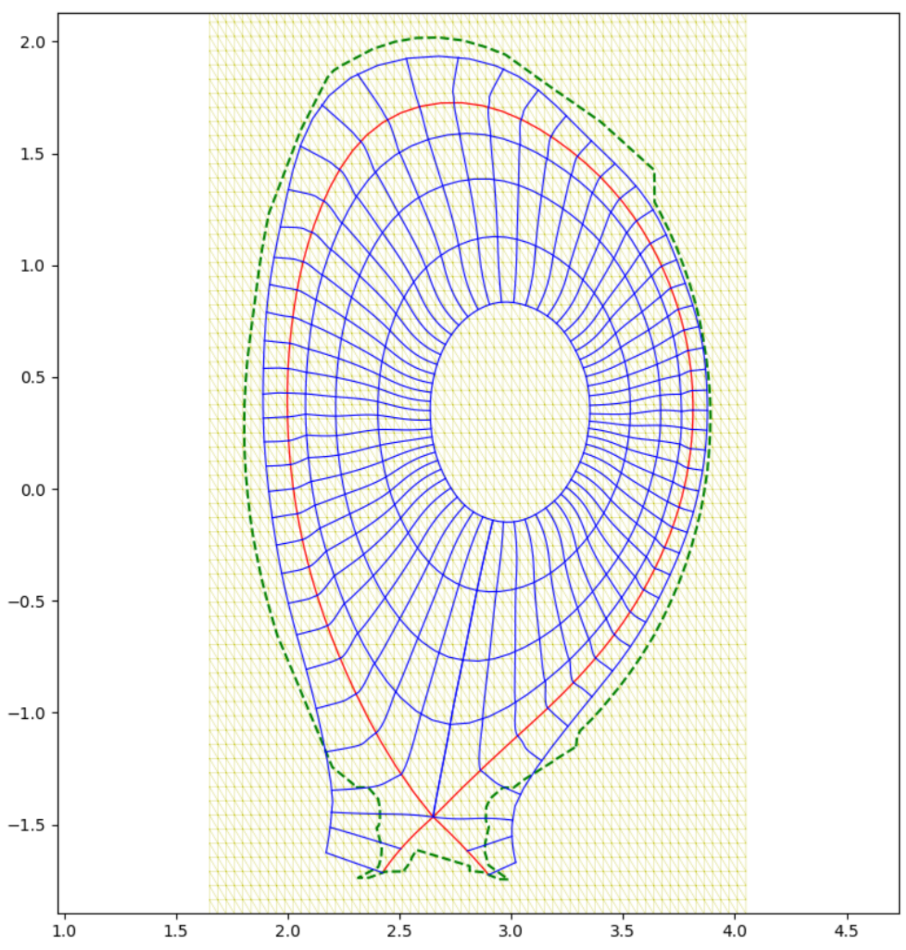


Figure 9: Flux-aligned multipatch grid (blue lines) from Tokamesh

5 Summary

5.1 What information do patches exchange?

- Mesh-points (Lagrange polynomials at interfaces, conforming projection),
- Local sums to compute the derivatives at the interfaces thanks to this type of sum of spline values (Advection)

$$\sum_{x_i \in \text{global space}} \alpha_i s(x_i) = \sum_{p \in \text{Patches}} \sum_{x_i \in p} \alpha_i s(x_i),$$

- Characteristic feet outside of the patch (Advection),
- Interpolated values for $\mathbf{A}$ and ρ (Advection).

5.2 What information do patches store?

- Mesh points, dimension (`DimXi`, `DimYi`), mapping, `SplineBuilder`, metadata,
- Boundary condition of global domain if an edge of the patch is on the global boundary,
- Values of functions ρ, ϕ, $\mathbf{A}$ on mesh points,
- Spline coefficients of functions (ρ, ϕ, $\mathbf{A}$),
- Reference to global domain class.

6 Global domain

- Global domain class
  - References to patches,
  - 'Connectivity' class which encodes the geometrical information.
- Connectivity class
  - Identify edges and corners of different patches (do we need to identify, corners of same patch e.g. when it closes on itself?)
  - For T-joint, identify sections of edges with sections of other edges, place corners in the middle of edges.