Anisotropic Diffusion in Toroidal geometries

Ahmed Ratnani, B Nkonga, Emmanuel Franck, Alina Eksaeva, Maria Kazakova

Anisotropic Diffusion in Toroidal geometries

A. Ratnani[1,*], B. Nkonga[2], E. Franck[3], A. Eksaeva[4], M. Kazakova[5]

[1] *Max-Planck Institute für PlasmaPhysik, Garching, Munich*

[2] *Laboratoire Jean Dieudonné, Université Nice Sophia-Antipolis, Nice, France*

[3] *Inria Nancy Grand-Est & IRMA, Strasbourg, France*

[4] *Moscow Engineering & Physics Institute MEPHI, Russia*

[5] *Lavrentyev Institute of Hydrodynamics of SB RAS, Russia*

February 24, 2015

Abstract

In this work, we present a new Finite Element framework for toroidal geometries based on a tensor product description of the $3D$ basis functions. In the poloidal plan, different discretizations, including B-Splines and cubic Hermite-Bézier patchs are defined, while for the toroidal direction both Fourier discretization and cubic Hermite-Bézier elements can be used. In this work, we study the MHD equilibrium by solving the Grad-Shafranov equation, which is the basis and the starting point of any MHD simulation. Then we study the Anistropic Diffusion problem in both steady and unsteady states.

1 Introduction

The context of this work is the simulation and the modeling of nuclear fusion reaction as power source. The aim of magnetic confinement fusion is to develop a power plant that gains energy from the fusion of deuterium and tritium in a magnetically confined plasma. ITER, a tokamak type fusion experiment currently being built in the South of France, is the next step towards this goal. One of the big challenge for the numerical simulation In a tokamak is the modeling and the simulation of edge instabilities as disruptions or ELMs [11] [1]. These instabilities which occurs a the boundary of the plasma generate a loss of energy and can damage critically the wall of the Tokomak. For this reason it is necessary to understand the behavior of these instabilities and find a way to control them using experiment and simulation. A physical model well suited to describe those large scale instabilities is the set of magneto-hydrodynamic equations (MHD) with resistivity and bi-fluid effects. Some code in the world work on this problem. One of this code is the JOREK code which solve some reduced MHD models [2] based on assumption on the velocity and magnetic fields in a toroidal geometry. The equations in the poloidal plane (circular, d-shape or x-point meshes) [8] [7] [6] are discretized using bezier splines and the toroidal discretization use Fourier expansion. In this work we are interested by study a small part of the equations used in JOREK which generate lot of numerical diffilcuties: the anisotropic diffusion.

The anistropic diffusion in the tokamak and JOREK context is a diffusion process mainly in the direction of the magnetic field described by a constant toroidal part and a poloidal perturbation. At the limit (where the ratio between the diffusion in the magnetic field direction and the isotropic diffusion tend to the infinity) the problem is singular [10]. This singularity at the limit generate ill-conditioning, large error in the perpendicular direction and other numerical problems. At the end, we want study the discretization of this operator with splines on toroidal geometry to understand and identify the main numerical problems linked to this operator and in the future proposed solution for the JOREK code.

In the first part of this paper we will details the construction of the 3D B-Splines (obtained by tensor product) used in the finit element method. The second part is on the discretization of the elliptic Grad-Safranov operator.

*Corresponding author: *ahmed.ratnani@ipp.mpg.de*

This operator is important in the JOREK since we solve this operator to obtain the magnetic equilibrium and the current in the MHD. To finish we will treat the anisotropic diffusion equation. These problems are study on some geometry classic in the tokamak context : circular or d-shape poloidal section.

2 Bézier Finite Elements Method

In this section, we recall some basics from B-splines used to construct the meshes linked to the JOREK and tokamak context and to discretize with finite element the equations.

2.1 B-Splines surfaces

We start this section by recalling some basic properies about B-splines curves and surfaces. We also recall some fundamental algorithms (knot insertion and degree elevation). Later, those algorithms will be used to develop a two grids solver for the Monge-Ampère equation. For a basic introduction to the subject, we refer to the book [9].
A B-Splines family, $(N_i)_{1\leqslant i\leqslant n}$ of order k, can be generated using a non-decreasing sequence of knots $T = (t_i)_{1\leqslant i\leqslant n+k}$.

Definition 2.1 (B-Splines series) *The j-th B-Spline of order k is defined by the recurrence relation:*

$$N_j^k = w_j^k N_j^{k-1} + (1 - w_{j+1}^k) N_{j+1}^{k-1}$$

where,

$$w_j^k(x) = \frac{x - t_j}{t_{j+k-1} - t_j} \qquad\qquad N_j^1(x) = \chi_{[t_j, t_{j+1}[}(x)$$

for $k \geq 1$ and $1 \leq j \leq n$.

We note some important properties of a B-splines basis:

- B-splines are piecewise polynomial of degree $p = k - 1$,
- Compact support; the support of N_j^k is contained in $[t_j, t_{j+k}]$,
- If $x \in\ ]t_j, t_{j+1}[$, then only the *B-splines* $\{N_{j-k+1}^k, \cdots, N_j^k\}$ are non vanishing at x,
- Positivity: $\forall j \in \{1, \cdots, n\}\ \ N_j(x) > 0, \ \ \forall x \in]t_j, t_{j+k}[$,
- Partition of unity : $\sum_{i=1}^n N_i^k(x) = 1, \forall x \in \mathbb{R}$,
- Local linear independence,
- If a knot t_i has a multiplicity m_i then the B-spline is $\mathcal{C}^{(p-m_i)}$ at t_i.

The vectorial space spanned by these B-splines, which we denote $\mathcal{S}_k(T, I)$, where I denotes the interval $[t_1, t_{n+1}]$, is called the Schoenberg space.

Definition 2.2 (B-Spline curve) *The B-spline curve in $\mathbb{R}^d$ associated to knot vector $T = (t_i)_{1\leqslant i\leqslant n+k}$ and the control polygon $(\mathbf{P}_i)_{1\leqslant i\leqslant n}$ is defined by :*

$$\mathcal{C}(t) = \sum_{i=1}^{n} N_i^k(t) \boldsymbol{P}_i$$

In (Fig. 1), we give an example of a quadratic B-Spline curve, and its corresponding knot vector and control points.

We have the following properties for a *B-spline* curve:

- If $n = k$, then $\mathcal{C}$ is just a Bézier-curve,
- $\mathcal{C}$ is a piecewise polynomial curve,

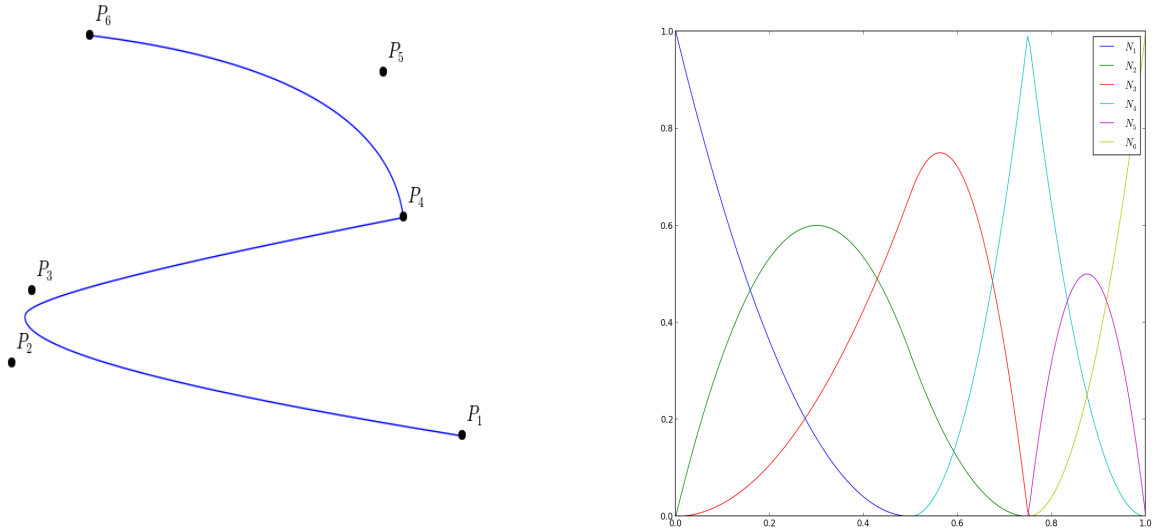


Figure 1: (left) A quadratic B-Spline curve and its control points using the knot vector $T = \{000\ \frac{1}{2}\ \frac{3}{4}\frac{3}{4}\ 111\}$, (right) the corresponding B-Splines.

- The curve interpolates its extremas if the associated multiplicity of the first and the last knot are maximum (*i.e.* equal to k), *i.e.* open knot vector,
- Invariance with respect to affine transformations,
- Strong convex-hull property:
  if $t_i \leq t \leq t_{i+1}$, then $\mathcal{C}(t)$ is inside the convex-hull associated to the control points $\mathbf{P}_{i-p}, \cdots, \mathbf{P}_i$,
- Local modification : moving the i^{th} control point $\mathbf{P}_i$ affects $\mathcal{C}(t)$, only in the interval $[t_i, t_{i+k}]$,
- The control polygon approaches the behavior of the curve.

Remark 2.3 *In order to model a singular curve, we can use multiple knots or control points, i.e.* $\mathbf{P}_i = \mathbf{P}_{i+1}$.

2.2 Fundamental geometric operations

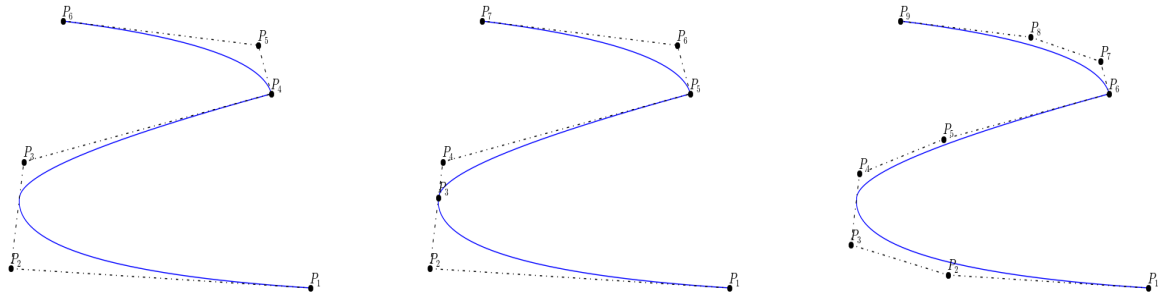
By inserting new knots into the knot vector, we add new control points without changing the shape of the B-Spline curve. This can be done using the DeBoor algorithm [3]. We can also elevate the degree of the B-Spline family and keep unchanged the curve [4]. In (Fig. 2), we apply these algorithms on a quadratic B-Spline curve and we show the position of the new control points.

2.3 Deriving a B-spline curve

The derivative of a B-spline curve is obtained as:

$$\mathcal{C}'(t) = \sum_{i=1}^{n} N_i^{k\prime}(t)\mathbf{P}_i = \sum_{i=1}^{n} \left(\frac{p}{t_{i+p} - t_i} N_i^{k-1}(t)\mathbf{P}_i - \frac{p}{t_{i+1+p} - t_{i+1}} N_{i+1}^{k-1}(t)\mathbf{P}_i \right) = \sum_{i=1}^{n-1} N_i^{k-1^*}(t)\mathbf{Q}_i \qquad (2.1)$$

where $\mathbf{Q}_i = p\frac{\mathbf{P}_{i+1} - \mathbf{P}_i}{t_{i+1+p} - t_{i+1}}$, and $\{N_i^{k-1^*},\ \ 1 \leq i \leq n-1\}$ are generated using the knot vector T^* which is obtained from T by reducing by one the multiplicity of the first and the last knot (in the case of open knot vector), *i.e.*



(a) (b) (c)

Figure 2: (a) A quadratic B-spline curve and its control points. The knot vector is $T = \{000, \frac{1}{2}, \frac{3}{4}, \frac{3}{4}, 111\}$. (b) The curve after a h-refinement by inserting the knot $\{0.5\}$ while the degree is kept equal to 2. (c) The curve after a p-refinement, the degree was raised by 1 (using cubic B-splines).

by removing the first and the last knot.
More generally, by introducing the B-splines family $\{N_i^{k-j^*}, 1 \leq i \leq n-j\}$ generated by the knot vector T^{j^*} obtained from T by removing the first and the last knot j times, we have the following result:

Proposition 2.4 *The j^{th} derivative of the curve $\mathcal{C}$ is given by*

$$\mathcal{C}^{(j)}(t) = \sum_{i=1}^{n-j} N_i^{k-j^*}(t)\mathbf{P}_i^{(j)}, \quad \textit{where} \quad \mathbf{P}_i^{(j)} = \frac{p-j+1}{t_{i+p+1}-t_{i+j}}\left(\mathbf{P}_{i+1}^{(j-1)} - \mathbf{P}_i^{(j-1)}\right) \quad \textit{and} \quad \mathbf{P}_i^{(0)} = \mathbf{P}_i$$

By denoting $\mathcal{C}'$ and $\mathcal{C}''$ the first and second derivative of the B-spline curve $\mathcal{C}$, it is easy to show that:

Proposition 2.5 *We have,*

- $\mathcal{C}'(0) = \frac{p}{t_{p+2}}(\mathbf{P}_2 - \mathbf{P}_1)$, $\mathcal{C}''(0) = \frac{p(p-1)}{t_{p+2}}\left(\frac{1}{t_{p+2}}\mathbf{P}_1 - \{\frac{1}{t_{p+2}} + \frac{1}{t_{p+3}}\}\mathbf{P}_2 + \frac{1}{t_{p+3}}\mathbf{P}_3\right)$,
- $\mathcal{C}'(1) = \frac{p}{1-t_n}(\mathbf{P}_n - \mathbf{P}_{n-1})$, $\mathcal{C}''(1) = \frac{p(p-1)}{1-t_n}\left(\frac{1}{1-t_n}\mathbf{P}_n - \{\frac{1}{1-t_n} + \frac{1}{1-t_{n-1}}\}\mathbf{P}_{n-1} + \frac{1}{1-t_{n-1}}\mathbf{P}_{n-2}\right)$.

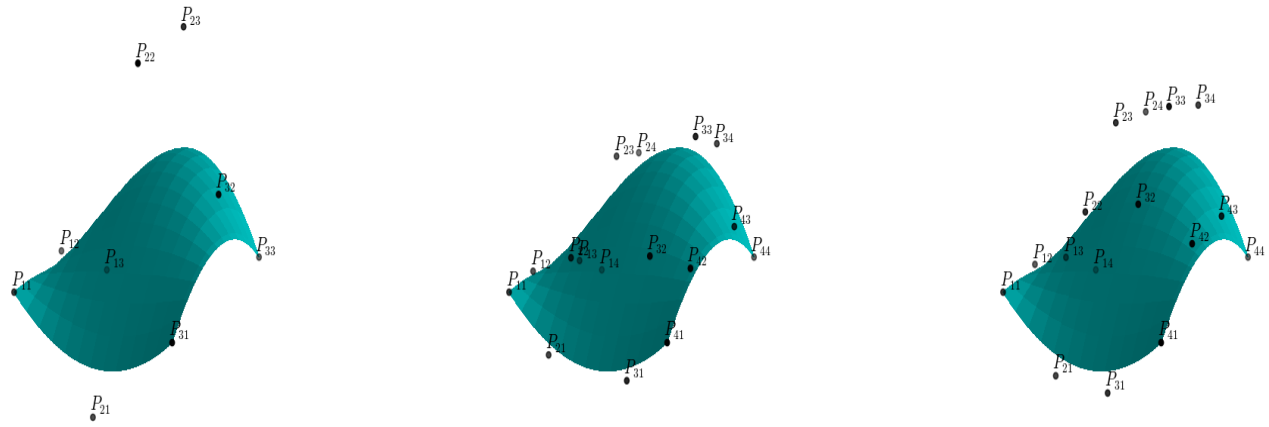
2.4 Multivariate tensor product splines

Let us consider d knot vectors $\mathcal{T} = \{T^1, T^2, \cdots, T^d\}$. For simplicity, we consider that these knot vectors are open, which means that k knots on each side are duplicated so that the spline is interpolating on the boundary, and of bounds 0 and 1. In the sequel we will use the notation $I = [0,1]$. Each knot vector T^i, will generate a basis for a Schoenberg space, $\mathcal{S}_{k_i}(T^i, I)$. The tensor product of all these spaces is also a Schoenberg space, namely $\mathcal{S}_{\mathbf{k}}(\mathcal{T})$, where $\mathbf{k} = \{k_1, \cdots, k_d\}$. The cube $\mathcal{P} = I^d = [0,1]^d$, will be referred to as a patch.
The basis for $\mathcal{S}_{\mathbf{k}}(\mathcal{T})$ is defined by a tensor product :

$$N_{\mathbf{i}}^{\mathbf{k}} := N_{i_1}^{k_1} \otimes N_{i_2}^{k_2} \otimes \cdots \otimes N_{i_d}^{k_d}$$

where, $\mathbf{i} = \{i_1, \cdots, i_d\}$.
A typical cell from $\mathcal{P}$ is a cube of the form : $Q_{\mathbf{i}} = [\xi_{i_1}, \xi_{i_1+1}] \otimes \cdots \otimes [\xi_{i_d}, \xi_{i_d+1}]$. In (Fig. 3), we apply knot insertion and elevation degree algorithms on a quadratic B-Spline surface and we show the position of the new control points.

Remark 2.6 *A B-Splines surface can be converted to a list of Bézier patchs. In order to do so, we only need to increase the multiplicity of internal knots to match the spline degree. Therefor, it is easy to extract B ézier patchs for all logical elements. Finaly, we can elevate their degrees to a desired one.*



(a) (b) (c)

Figure 3: (a) A quadratic B-spline surface and its control points. The knot vectors are $T_1 = T_2 = \{000, 111\}$. (b) The curve after a h-refinement by inserting the knot $\{0.5\}$ in both directions, while the degree is kept equal to 2. (c) The curve after a p-refinement, the degree was raised by 1 (using cubic B-splines).

2.5 Bézier-Hermite patches

Let us consider a Bézier patch, defined by the following parametric surface

$$\mathbf{x}(s,t) = \sum_{i,j=0}^{p,q} \mathbf{x}_{ij} B_i(s) B_j(t), \qquad s,t \in [0,1] \tag{2.2}$$

Computing derivatives of this surface on the four (logical) points $\{(s,t) = (0,0),(0,1),(1,0),(1,1)\}$ leads to

$$\begin{cases} \mathbf{x}(0,0) = \mathbf{x}_{00}, & \mathbf{x}_s(0,0) = 3(\mathbf{x}_{10} - \mathbf{x}_{00}), & \mathbf{x}_t(0,0) = 3(\mathbf{x}_{01} - \mathbf{x}_{00}), & \mathbf{x}_{st}(0,0) = 9(\mathbf{x}_{00} + \mathbf{x}_{11} - \mathbf{x}_{01} - \mathbf{x}_{10}) \\ \mathbf{x}(0,1) = \mathbf{x}_{03}, & \mathbf{x}_s(0,1) = 3(\mathbf{x}_{13} - \mathbf{x}_{03}), & \mathbf{x}_t(0,1) = 3(\mathbf{x}_{03} - \mathbf{x}_{02}), & \mathbf{x}_{st}(0,1) = 9(\mathbf{x}_{03} + \mathbf{x}_{12} - \mathbf{x}_{02} - \mathbf{x}_{13}) \\ \mathbf{x}(1,0) = \mathbf{x}_{30}, & \mathbf{x}_s(1,0) = 3(\mathbf{x}_{30} - \mathbf{x}_{20}), & \mathbf{x}_t(1,0) = 3(\mathbf{x}_{31} - \mathbf{x}_{30}), & \mathbf{x}_{st}(1,0) = 9(\mathbf{x}_{30} + \mathbf{x}_{21} - \mathbf{x}_{20} - \mathbf{x}_{31}) \\ \mathbf{x}(1,1) = \mathbf{x}_{33}, & \mathbf{x}_s(1,1) = 3(\mathbf{x}_{33} - \mathbf{x}_{23}), & \mathbf{x}_t(1,1) = 3(\mathbf{x}_{33} - \mathbf{x}_{32}), & \mathbf{x}_{st}(1,1) = 9(\mathbf{x}_{33} + \mathbf{x}_{22} - \mathbf{x}_{23} - \mathbf{x}_{32}) \end{cases} \tag{2.3}$$

Now, let us introduce the following quantities

$$\begin{cases} a_{00} = \|\mathbf{x}_{10} - \mathbf{x}_{00}\|, & b_{00} = \|\mathbf{x}_{01} - \mathbf{x}_{00}\|, & \mathbf{u}_{00} = \frac{\mathbf{x}_{10} - \mathbf{x}_{00}}{a_{00}}, & \mathbf{v}_{00} = \frac{\mathbf{x}_{01} - \mathbf{x}_{00}}{b_{00}}, & \mathbf{w}_{00} = \frac{\mathbf{x}_{00} + \mathbf{x}_{11} - \mathbf{x}_{01} - \mathbf{x}_{10}}{a_{00} b_{00}} \\ a_{03} = \|\mathbf{x}_{13} - \mathbf{x}_{03}\|, & b_{03} = \|\mathbf{x}_{03} - \mathbf{x}_{02}\|, & \mathbf{u}_{03} = \frac{\mathbf{x}_{13} - \mathbf{x}_{03}}{a_{03}}, & \mathbf{v}_{03} = \frac{\mathbf{x}_{03} - \mathbf{x}_{02}}{b_{03}}, & \mathbf{w}_{03} = \frac{\mathbf{x}_{03} + \mathbf{x}_{12} - \mathbf{x}_{02} - \mathbf{x}_{13}}{a_{03} b_{03}} \\ a_{30} = \|\mathbf{x}_{30} - \mathbf{x}_{20}\|, & b_{30} = \|\mathbf{x}_{31} - \mathbf{x}_{30}\|, & \mathbf{u}_{30} = \frac{\mathbf{x}_{30} - \mathbf{x}_{20}}{a_{30}}, & \mathbf{v}_{30} = \frac{\mathbf{x}_{31} - \mathbf{x}_{30}}{b_{30}}, & \mathbf{w}_{30} = \frac{\mathbf{x}_{30} + \mathbf{x}_{21} - \mathbf{x}_{20} - \mathbf{x}_{31}}{a_{30} b_{30}} \\ a_{33} = \|\mathbf{x}_{33} - \mathbf{x}_{23}\|, & b_{33} = \|\mathbf{x}_{33} - \mathbf{x}_{32}\|, & \mathbf{u}_{33} = \frac{\mathbf{x}_{33} - \mathbf{x}_{23}}{a_{33}}, & \mathbf{v}_{33} = \frac{\mathbf{x}_{33} - \mathbf{x}_{32}}{b_{33}}, & \mathbf{w}_{33} = \frac{\mathbf{x}_{33} + \mathbf{x}_{22} - \mathbf{x}_{23} - \mathbf{x}_{32}}{a_{33} b_{33}} \end{cases} \tag{2.4}$$

we have,

$$\begin{cases} \mathbf{x}_{10} = a_{00}\mathbf{u}_{00} + \mathbf{x}_{00}, & \mathbf{x}_{01} = b_{00}\mathbf{v}_{00} + \mathbf{x}_{00}, & \mathbf{x}_{11} = a_{00}b_{00}\mathbf{w}_{00} + a_{00}\mathbf{u}_{00} + b_{00}\mathbf{v}_{00} + \mathbf{x}_{00} \\ \mathbf{x}_{13} = a_{03}\mathbf{u}_{03} + \mathbf{x}_{03}, & \mathbf{x}_{02} = -b_{03}\mathbf{v}_{03} + \mathbf{x}_{03}, & \mathbf{x}_{12} = a_{03}b_{03}\mathbf{w}_{03} + a_{03}\mathbf{u}_{03} - b_{03}\mathbf{v}_{03} + \mathbf{x}_{03} \\ \mathbf{x}_{20} = -a_{30}\mathbf{u}_{30} + \mathbf{x}_{30}, & \mathbf{x}_{31} = b_{30}\mathbf{v}_{30} + \mathbf{x}_{30}, & \mathbf{x}_{21} = a_{30}b_{30}\mathbf{w}_{30} - a_{30}\mathbf{u}_{30} + b_{30}\mathbf{v}_{30} + \mathbf{x}_{30} \\ \mathbf{x}_{23} = -a_{33}\mathbf{u}_{33} + \mathbf{x}_{33}, & \mathbf{x}_{32} = -b_{33}\mathbf{v}_{33} + \mathbf{x}_{33}, & \mathbf{x}_{22} = a_{33}b_{33}\mathbf{w}_{33} - a_{33}\mathbf{u}_{33} - b_{33}\mathbf{v}_{33} + \mathbf{x}_{33} \end{cases} \tag{2.5}$$

Pluging these relations into (Eq. 2.2), we get

$$\begin{aligned}\mathbf{x}(s,t) = &\mathbf{x}_{00}M_{00}(s,t) + a_{00}\mathbf{u}_{00}N_{01}(s,t) + b_{00}\mathbf{v}_{00}P_{02}(s,t) + a_{00}b_{00}\mathbf{w}_{00}Q_{03}(s,t)\\ &+\mathbf{x}_{03}M_{10}(s,t) + a_{03}\mathbf{u}_{03}N_{11}(s,t) + b_{03}\mathbf{v}_{03}P_{12}(s,t) + a_{03}b_{03}\mathbf{w}_{03}Q_{13}(s,t)\\ &+\mathbf{x}_{30}M_{30}(s,t) + a_{30}\mathbf{u}_{30}N_{30}(s,t) + b_{30}\mathbf{v}_{30}P_{30}(s,t) + a_{30}b_{30}\mathbf{w}_{30}Q_{30}(s,t)\\ &+\mathbf{x}_{33}M_{33}(s,t) + a_{33}\mathbf{u}_{33}N_{33}(s,t) + b_{33}\mathbf{v}_{33}P_{33}(s,t) + a_{33}b_{33}\mathbf{w}_{33}Q_{33}(s,t)\end{aligned} \tag{2.6}$$

where the new basis is (Eq. 2.7)

$$\begin{cases}\begin{cases}M_{00}(s,t) = B_0(s)B_0(t) + B_1(s)B_0(t) + B_0(s)B_1(t) + B_1(s)B_1(t), & N_{00}(s,t) = B_1(s)B_0(t) + B_1(s)B_1(t)\\ P_{00}(s,t) = B_0(s)B_1(t) + B_1(s)B_1(t), & Q_{00}(s,t) = B_1(s)B_1(t)\end{cases}\\ \begin{cases}M_{03}(s,t) = B_0(s)B_3(t) + B_1(s)B_3(t) + B_0(s)B_2(t) + B_1(s)B_2(t), & N_{03}(s,t) = B_1(s)B_3(t) + B_1(s)B_2(t)\\ P_{03}(s,t) = -B_0(s)B_2(t) - B_1(s)B_2(t), & Q_{03}(s,t) = B_1(s)B_2(t)\end{cases}\\ \begin{cases}M_{30}(s,t) = B_3(s)B_0(t) + B_2(s)B_0(t) + B_3(s)B_1(t) + B_2(s)B_1(t), & N_{30}(s,t) = -B_2(s)B_0(t) - B_2(s)B_1(t)\\ P_{30}(s,t) = B_3(s)B_1(t) + B_2(s)B_1(t), & Q_{30}(s,t) = B_2(s)B_1(t)\end{cases}\\ \begin{cases}M_{33}(s,t) = B_3(s)B_3(t) + B_2(s)B_3(t) + B_3(s)B_2(t) + B_2(s)B_2(t), & N_{33}(s,t) = -B_2(s)B_3(t) - B_2(s)B_2(t)\\ P_{33}(s,t) = -B_3(s)B_2(t) - B_2(s)B_2(t), & Q_{33}(s,t) = B_2(s)B_2(t)\end{cases}\end{cases} \tag{2.7}$$

2.6 Software Implementation

CAID [13] is an open source multi-platform software that has been designed for IsoGeometric Analysis Pre and Post Processing. Its design goal is to provide a fast, light and user-friendly designer and meshing tool. The later is provided by **Πgasus** [14] which is an open source Python package for solving (system of) Partial Differential Equations. Moreover, the user can write his own solvers and integrate them in **CAID**. One particular and interesting capability of **CAID** is the macro-recording, which provides the user with a Python script of all his interaction with **CAID**.

2.7 Hermite-Bézier Finite Elements Method

In classical Bézier [2] (and more generaly in IsoGeometric [5]) Finite Elements method, the isoparametric approach is used while degrees of freedom are related to the control points 2.10. In the sequel, we consider a $2D$ domain $\Omega_h \subset \Omega$ and its associated *triangulation* $\mathcal{Q}_h$ such that

$$\bigcup_{e\in\mathcal{Q}_h} e = \Omega_h \quad \text{and} \quad e_1 \bigcap e_2 = \emptyset,\ \forall e_1, e_2 \in \mathcal{Q}_h,\ \text{s.t. } e_1 \neq e_2 \tag{2.8}$$

where every element $e \in \mathcal{Q}_h$ denotes a Bézier patch.

$$\mathbf{x}(s,t) = \sum_{e\in\mathcal{Q}_h}\sum_{i,j=0}^{p,q} \mathbf{x}_{ij}B_i(s)B_j(t) \tag{2.9}$$

$$u^h(s,t) = \sum_{e\in\mathcal{Q}_h}\sum_{i,j=0}^{p,q} u^h_{ij}B_i(s)B_j(t) \tag{2.10}$$

Hermite-Bézier elements are therefor obtained by a change of basis within each element (Bézier patch), where we replace $\mathbf{x}$ in (Eq. 2.6) by $\mathbf{x} := (\mathbf{x}, u^h)$.

Remark 2.7 *The assembing procedure is done on the logical domain $[0,1]^2$. Therefore, all derivatives are computed on the logical domain and must be transported to the physical domain.*

3 Grad-Shafranov solver

The first model treated is the Grad-Safranov operator. This operator is obtained using the MHD equilibrium (uniform flow) in a cylindrical geometry. The MHD equilibrium is described by the force balance, the Ampere's law in addition to the magnetic divergence constraint (Eq. 3.11)

$$\begin{cases} \mathbf{J} \times \mathbf{B} = \nabla P \\ \nabla \times \mathbf{B} = \mu_0 \mathbf{J} \\ \nabla \cdot \mathbf{B} = 0 \end{cases} \tag{3.11}$$

In tokamak geometries and cylindrical coordinate system (Fig. 4), the magnetic field $\mathbf{B}$ can be expressed as $\mathbf{B} = \nabla\varphi \times \nabla\psi + g\nabla\varphi$, where ψ is the poloidal flux function. This choice correspond to a magnetic field with a main part constant in the toroidal direction plus a poloidal perturbation which depend of ψ. The current density $\mathbf{J}$ can also be written, in the same form, as $\mathbf{J} = RJ_\varphi \nabla\varphi + \frac{1}{\mu_0}\nabla g \times \nabla\varphi$.

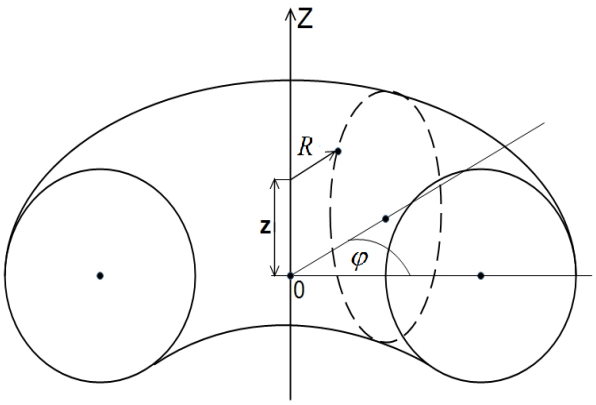


Figure 4: The toroidal plasma configuration

Using $\mathbf{B}\cdot\nabla P = 0$ it is easy to see that the pressure is a function of the flux ψ. On the other hand, $\mathbf{J}\cdot\nabla P = 0$ implies that the function g is also a flux function. Inserting these relations in the balance force equation yields to the second order nonlinear elliptic equation, knwon as Grad-Shafranov-Shlüter equation (Eq. 3.13):

$$\Delta^*\psi = R\partial_R\left(\frac{1}{R}\partial_R\psi\right) + \partial^2_{zz}\psi = -\mu_0 R^2 \frac{d}{d\psi}P - g\frac{d}{d\psi}g \tag{3.12}$$

3.1 Finite Elements solver for Grad-Shafranov equation

To simplify the weak form associated to the Grad-Safranov operator, we solve a modified version multiplying by $\frac{1}{R^2}$:

$$\frac{1}{R^2}\Delta^*\psi = \frac{1}{R}\partial_R\left(\frac{1}{R}\partial_R\psi\right) + \partial^2_{zz}\psi = -\mu_0\frac{d}{d\psi}P - \frac{1}{R^2}g\frac{d}{d\psi}g \tag{3.13}$$

In the sequel, we rewrite the right hand side term as

$$\mathcal{F}(R, Z, \psi) = -\mu_0\frac{d}{d\psi}P - \frac{1}{2R^2}\frac{d}{d\psi}g^2 \tag{3.14}$$

The Picard algorithm can be written as

- ψ^0 is given,

- knowning ψ^n, we solve:

$$\frac{1}{R^2}\Delta^*\psi_{n+1} = \mathcal{F}(R, Z, \psi_n) \tag{3.15}$$

Let $\phi \in \mathcal{V}_h \subset \mathcal{V}$ a test function. Multiplying it by the Grad-Shafranov equation 3.13 and integrating by parts, in the cylindric coordinates, leads to

$$\int_\Omega \frac{1}{R}\nabla\psi_{n+1} \cdot \nabla\phi \; d\Omega = \int_\Omega \mathcal{F}(R, Z, \psi_n)\phi \; d\Omega \tag{3.16}$$

3.2 Numerical results

A first validation is done on an analytical solution on a square domain, using Collela's mapping (Eq. 3.17) approximated by Hermite-Bézier patchs.

$$\mathbf{x} = F(\boldsymbol{\eta}) = (\eta_1 + \alpha \sin(k_1\eta_1)\sin(k_2\eta_2), \eta_2 + \alpha \sin(k_1\eta_1)\sin(k_2\eta_2)). \tag{3.17}$$

The right hand side and the nonlinear function $\mathcal{F}$ are computed so that the solution vanishes on the boundary, where $\mathcal{F}$ contains a quadratic dependence on ψ. In (Fig. 6) and (Tab. 1), we show the expected convergence order, while in (Fig. 5), the numerical solution and the numerical error are shown. (Fig. 6) shows also the evolution of the L^2 and H^1 norms depending on Picard's iterations. As we may notice, our Picard's algorithm converges after 6 iterations, but this does not seem to be the rule for other choices of the geometry and the function $\mathcal{F}$.

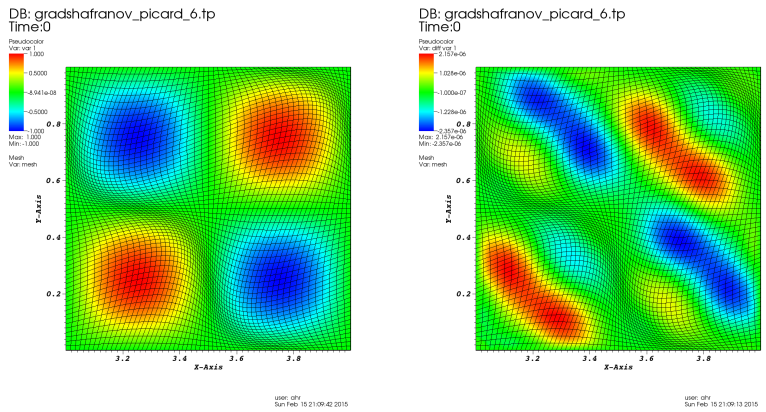


Figure 5: Numerical solution and its error for GS equation on a square domain, using Collela's mapping with a grid 64×64.

L^2	3.09669	3.2561	3.6696	3.8603
H^1	2.4120	2.5738	2.8099	2.9358

Table 1: L^2 and H^1 convergence orders, for the GS equation on a square domain, using Collela's mapping.

In the sequel, we present different simulations based on characteristic parameters describing the cross-section for *ITER*, *ASDEX-Upgrade* and *JET* tokamaks. These parameters are: the *inverse aspect-ration* ϵ, the *elongation* κ and the *triangularity* δ. They are given by the following formulae (Eq. 3.18):

$$\begin{cases} R_0 = \frac{R_{min}+R_{max}}{2} \\ \epsilon = \frac{R_{max}-R_0}{R_0} = \frac{R_0-R_{min}}{R_0} \\ \kappa\epsilon = \frac{Z_{max}}{R_0} \\ 1-\delta\epsilon = \frac{R(Z=Z_{max})}{R_0} \end{cases} \quad \text{and} \quad \begin{cases} \psi(R_{max}, 0) = 0 \\ \psi(R_{min}, 0) = 0 \\ \psi(R(Z = Z_{max}), \pm Z_{max}) = 0 \end{cases} \tag{3.18}$$

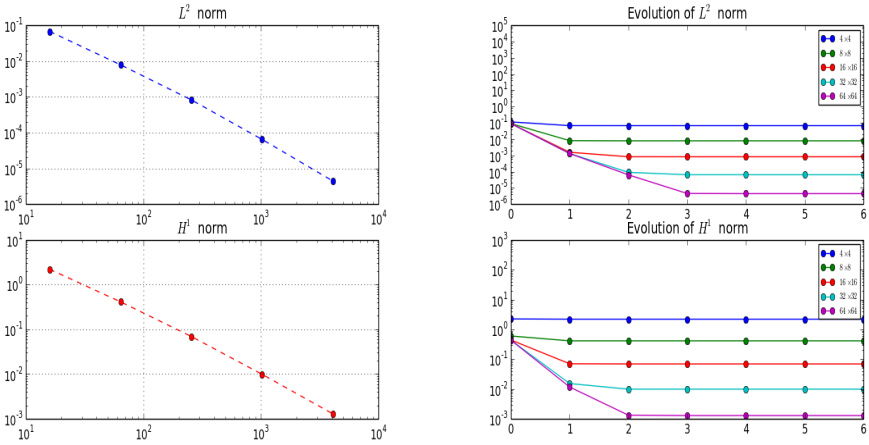


Figure 6: L^2 and H^1 errors (left) after convergence (right) and their evolutions, for the GS equation on a square domain, using Collela's mapping.

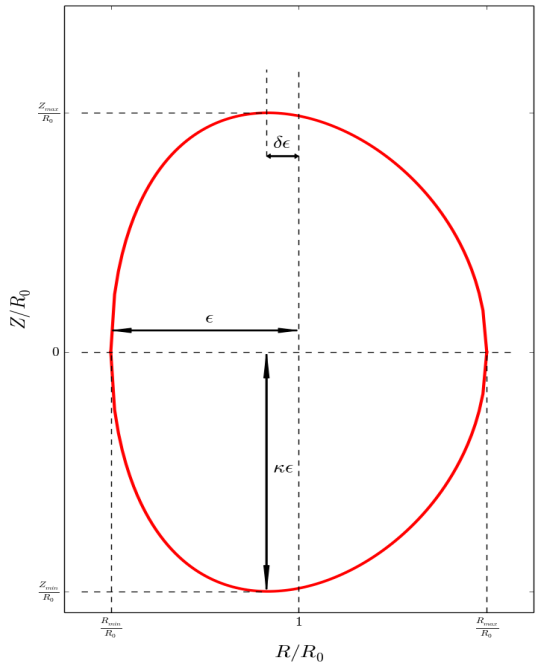


Figure 7: Geometric definition of the parameters ϵ, κ and δ.

where in the last equation, we consider a given flux surface (in our case, defined by $\psi(R, Z) = 0$) and R as a function of Z.
In the following test, we consider the right hand side function

$$\mathcal{F}(R, Z, \psi) = R^2 \tag{3.19}$$

which gives the analytical solution [12],

$$\psi(R,Z) = \frac{R^4}{8} + d_1 + d_2 R^2 + d_3(R^4 - 4(RZ)^2) \tag{3.20}$$

Pluging (Eq. 3.18) in (Eq. 3.20) leads to the following system for d_1, d_2 and d_3:

$$\begin{pmatrix} 1 & (1+\epsilon)^2 & (1+\epsilon)^4 \\ 1 & (1-\epsilon)^2 & (1-\epsilon)^4 \\ 1 & (1-\delta\epsilon)^2 & (1-\delta\epsilon)^4 - 4(1-\delta\epsilon)^2\kappa^2\epsilon^2 \end{pmatrix} \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} = -\frac{1}{8} \begin{pmatrix} (1+\epsilon)^4 \\ (1-\epsilon)^4 \\ (1-\delta\epsilon)^4 \end{pmatrix} \tag{3.21}$$

In figure (Fig. 8), we plot the computaional domain and the numerical solution for *ITER*, *ASDEX-Upgrade* and *JET* relevant parameters. In figure (Fig. 9) we show two different meshes generated by **CAID**. In the

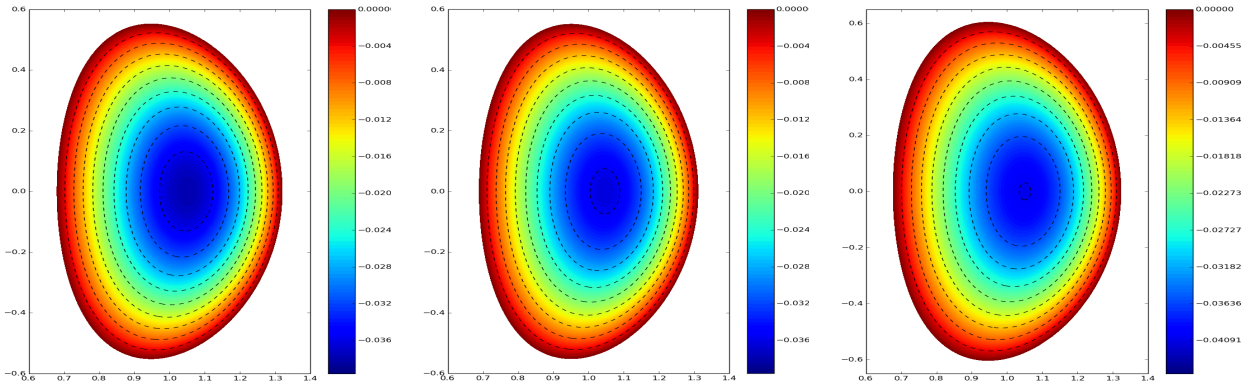


Figure 8: Coputational domain and the analytical solution for: (left) *ITER* $(R_0, \epsilon, \kappa, \delta) = (1, 0.32, 1.7, 0.33)$, (middle) *ASDEX-Upgrade* $(R_0, \epsilon, \kappa, \delta) = (1.645, 0.311, 1.77, 0.0932)$, (right) *JET* $(R_0, \epsilon, \kappa, \delta) = (2.924, 0.323, 1.87, 0.141)$

next section, we shall present the impact of these meshes on the numerical solution of the Anisotropic Diffusion problem. In (Fig. 10) we plot the L^2 and H^1 norms for MESH2. In this case, the theoretical convergence order is not achieved; the reason is the bad approximation of the boundary when $\partial_R\psi = 0$ or $\partial_Z\psi = 0$.

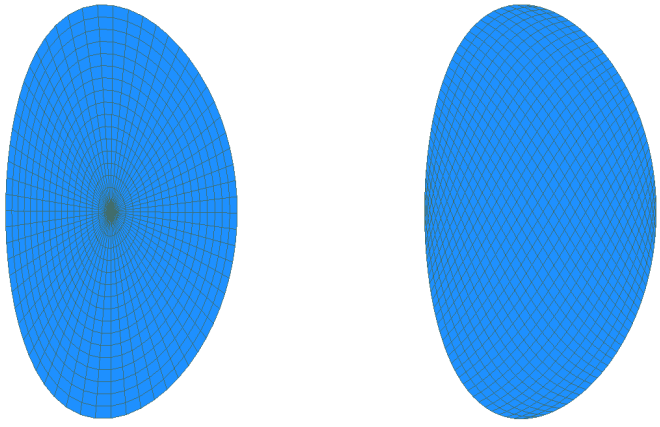


Figure 9: *ITER* like relevant parameter domain: (left) MESH1 (right) MESH2.

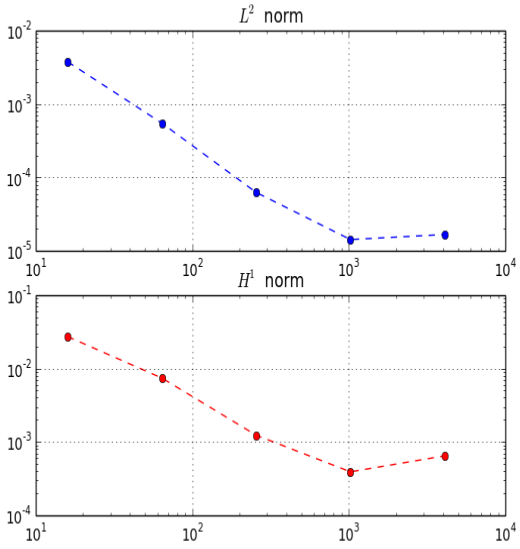


Figure 10: Convergence order for Iter like relevant parameter domain.

4 Anisotropic Diffusion

In this section, we are interested in the resolution of the anisotropic diffusion problem for both steady and unsteady cases. The anisotropic diffusion time evolution problem is

$$\partial_t u = \nabla \cdot (\mathbf{K} \nabla u) + f, \quad \mathbf{x} \in \Omega \tag{4.22}$$

where u describes the temperature inside a tokamak, the conductivity $\mathbf{K} = \kappa_{\|}\mathbf{K}_{\|} + \kappa_I \mathbf{I}$ is a 3 by 3 tensor. The later is a sum of two contributions, first parallel component is given by:

$$\kappa_{\|}\mathbf{K}_{\|} = \kappa_{\|} \frac{\mathbf{B}\mathbf{B}^{\mathbf{T}}}{\|B\|^2}$$

for prescribed magnetic field $\boldsymbol{B}$, $\kappa_{\|}$ is a parallel diffusion coefficient. The second component $\kappa_I \mathbf{I}$ is a standard isotropic diffusion. We are interested in highly anisotropic configurations with $\dfrac{\kappa_{\|}}{\kappa_I} \simeq 10^6 \gg 1$.

Circular Plasma For circular cross section plasma, the Grad-Shafranov equation associated to the nonlinear right hand side $\mathcal{F}(R, Z, \psi) = R^2$, leads to

$$\psi(R, Z) = C_\psi \ln\left[1 + \frac{r^2}{a^2}\right], \quad \text{with} \quad C_\psi = \frac{a^2}{2R_0 q_0}$$

General case In the general case, we need to solve the equilibrium (Grad-Shafranov) for the potential ψ in order to define the magnetic field.

4.1 The Elliptic Anisotropic Diffusion equation

We start by solving the following anistropic diffusion problem (Eq. 4.23)

$$-\nabla \cdot \mathbf{K} \cdot \nabla u = f, \quad \mathbf{x} \in \Omega \tag{4.23}$$

Let ϕ_i be a test function. Multiplying (Eq. 4.23) by ϕ_i and integrating by parts leads to

$$\int_\Omega \frac{\kappa_{\|}}{\|\mathbf{B}\|^2} (\boldsymbol{B} \cdot \nabla u)(\boldsymbol{B} \cdot \nabla \phi_i) + \kappa_I \nabla u \cdot \nabla \phi_i = \int_\Omega f \phi_i \tag{4.24}$$

Using the expansion $u = \sum_{j=1}^{n} u_j \phi_j$, we get

$$\sum_{j=1}^{n} u_j \left(\int_\Omega \frac{\kappa_\parallel}{\|\boldsymbol{B}\|^2} (\boldsymbol{B} \cdot \nabla \phi_j)(\boldsymbol{B} \cdot \nabla \phi_i) + \kappa_I \nabla \phi_j \cdot \nabla \phi_i \right) = \int_\Omega f \phi_i \tag{4.25}$$

which leads to the linear system $\mathcal{M}\mathbf{U} = \mathbf{F}$ where

$$\mathcal{M}_{ij} = \int_\Omega \frac{\kappa_\parallel}{\|\boldsymbol{B}\|^2} (\boldsymbol{B} \cdot \nabla \phi_j)(\boldsymbol{B} \cdot \nabla \phi_i) + \kappa_I \nabla \phi_j \cdot \nabla \phi_i, \quad \text{and} \quad F_i = \int_\Omega f \phi_i, \qquad \forall i, j \in [1, n] \tag{4.26}$$

Remark 4.1 *For validation, we take any function u that vanishes on the boundary and then compute $f = -\nabla \cdot K \cdot \nabla u$. In the case of circular cross section plasma, we take $u_{exact} = 1 - \frac{(R-R_0)^2+Z^2}{a^2}$. The solution for an annulus cross section plasma, an analytical solution is $u_{exact} = \left(1 - \frac{(R-R_0)^2+Z^2}{a^2}\right)\left(\frac{(R-R_0)^2+Z^2}{a^2_{center}} - 1\right)$*

We show in (Fig. 11), the L^2 error norm for different meshes and values of $\kappa_\parallel$. GMRES was used with a Jacobi preconditioner for a tolerance $tol_r = 10^{-11}$. Block-Jacobi preconditioner was also tested and results were quite equivalents. We may notice that for a diffusion $> 10^4$ the solver does not converge, because of the bad conditioning of the linear system. This is can be viewed in (Fig. 12), where we show the evolution of the residual error for the Krylov solver.

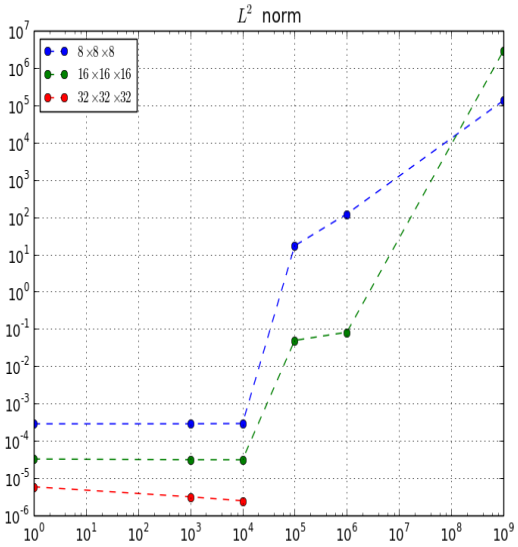


Figure 11: Anisotropic Diffusion operator: L^2 error norm depending on $\kappa_\parallel$ for different meshes.

4.2 Time evolution problem

For the time depending problem, we consider the following implicit scheme

$$\frac{u^{n+1} - u^n}{\Delta t} - \nabla \cdot \left(\mathbf{K} \nabla u^{n+1}\right) = f,$$

which leads to

$$u^{n+1} - \Delta t \ \nabla \cdot \left(\mathbf{K} \nabla u^{n+1}\right) = u^n + \Delta t f. \tag{4.27}$$

Let ϕ_i be a test function. Multiplying (Eq. 4.27) by ϕ_i, integrating by parts and using the expansion of $u = \sum_{j=1}^{n} u_j \phi_j$, leads to the following linear system problem

$$\mathcal{M}^{\mathcal{I}} \mathbf{U}^{n+1} = \mathcal{M}^{\mathcal{E}} \mathbf{U}^n + \mathbf{F}$$

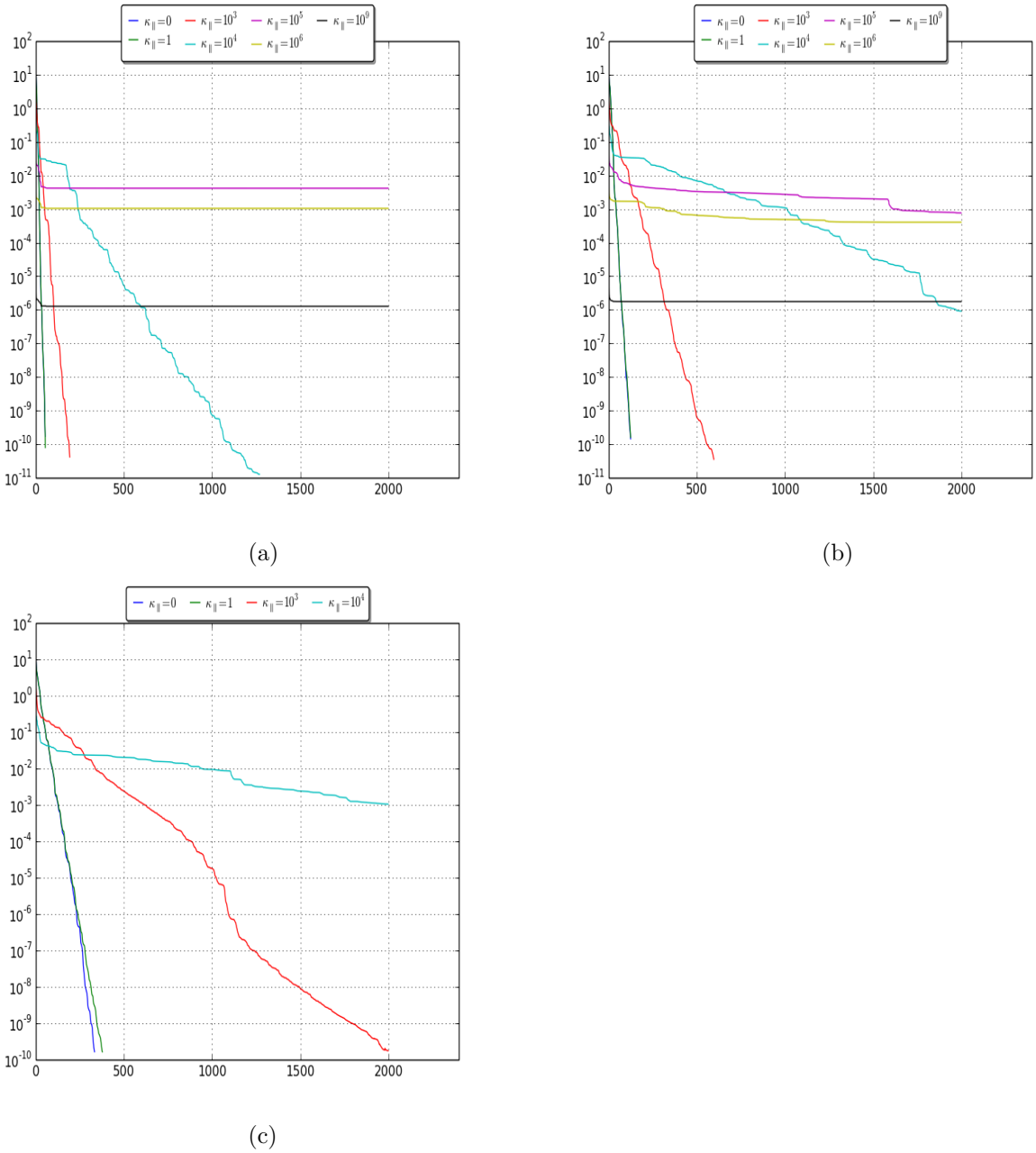


Figure 12: Anisotropic Diffusion operator: evolution of the residual error for the Krylov solver, for a mesh (a) $8 \times 8 \times 8$, (b) $16 \times 16 \times 16$ and (c) $32 \times 32 \times 32$

where for $i, j \in [1, n]$, we have

$$\mathcal{M}^{\mathcal{I}}_{ij} = \int_\Omega \phi_i \phi_j + \Delta t \left(\frac{\kappa_\parallel}{\|\mathbf{B}\|^2} (\boldsymbol{B} \cdot \nabla \phi_j)(\boldsymbol{B} \cdot \nabla \phi_i) + \kappa_I \nabla \phi_j \cdot \nabla \phi_i \right) \tag{4.28}$$

$$\mathcal{M}^{\mathcal{E}}_{ij} = \int_\Omega \phi_i \phi_j \tag{4.29}$$

$$F_i = \int_\Omega \Delta t f \phi_i \tag{4.30}$$

4.2.1 Steady case

In the sequel, we are interested in the steady state solution of the Anisotropic Diffusion problem on a circular cross section domain with a mesh type MESH1 and MESH2. In order to avoid the polar-like singularity at the center of the plasma in MESH1, we consider an annulus domain of minimum radius $r_{min} = 0.1$, while the maximum radius r_{max} is kept equal to 1 for both meshes. The source terms are computed as the solution of the Elliptic Anisotropic Diffusion equation, for $\kappa_{\|} = 0$ with the following analytical solutions

$$\begin{cases} u_{exact} = 1 - \frac{(R-R_0)^2+Z^2}{a^2}, & \text{for MESH1} \\ u_{exact} = \left(1 - \frac{(R-R_0)^2+Z^2}{a^2}\right)\left(\frac{(R-R_0)^2+Z^2}{a^2_{center}} - 1\right), & \text{for MESH2} \end{cases} \tag{4.31}$$

Numerical simulations were done with time steppings $dt = 5.10^{-3}, 1$ and a final time $T_{final} = 0.5, 20$ respectively. GMRES and Jacobi preconditioner were used. In (Fig. 13 15), we show the final L^2 and H^1 norms as functions of $\kappa_{\|}$ for different meshes while in (Fig. 14 16), we show the time evolution of the L^2 and H^1 norms. Finaly, in table (2), we show the power approximation, h-dependance, of the L^2 and H^1 norms between the grids $8 \times 4 \times 4$ and $16 \times 8 \times 8$. As we can notice, the convergence order is conserved at the final time. Dure to our observations, this convergence order should be better for finer grids.

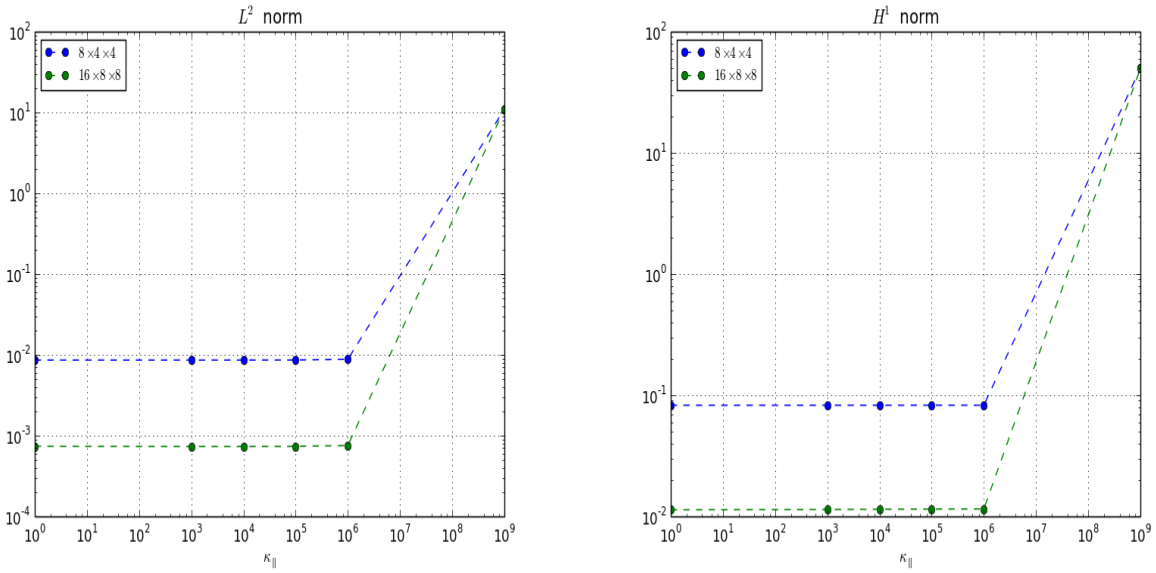


Figure 13: Short time behavior of L^2 and H^1 error norms for different values of $\kappa_{\|}$ at time $T_{final} = 0.5$ and $dt = 5.10^{-3}$ using MESH1

κ	0	1	10^3	10^4	10^5	10^6
L^2 norm						
Initial time ($t = 0$)	3.675	3.675	3.684	3.689	3.69	3.68
Final time ($t = 0.5$)	3.544	3.544	3.55	3.55	3.55	3.544
H^1 norm						
Initial time ($t = 0$)	2.925	2.925	2.918	2.911	2.907	2.906
Final time ($t = 0.5$)	2.865	2.865	2.858	2.85	2.845	2.843

Table 2: Steady state h-depdence of the L^2 and H^1 norms between the grids $8 \times 4 \times 4$ and $16 \times 8 \times 8$

4.2.2 Evolution of a Gaussian Pulse

In this last section, we are interested in the time evolution of a Gaussian pulse, without source term, for the initial condition

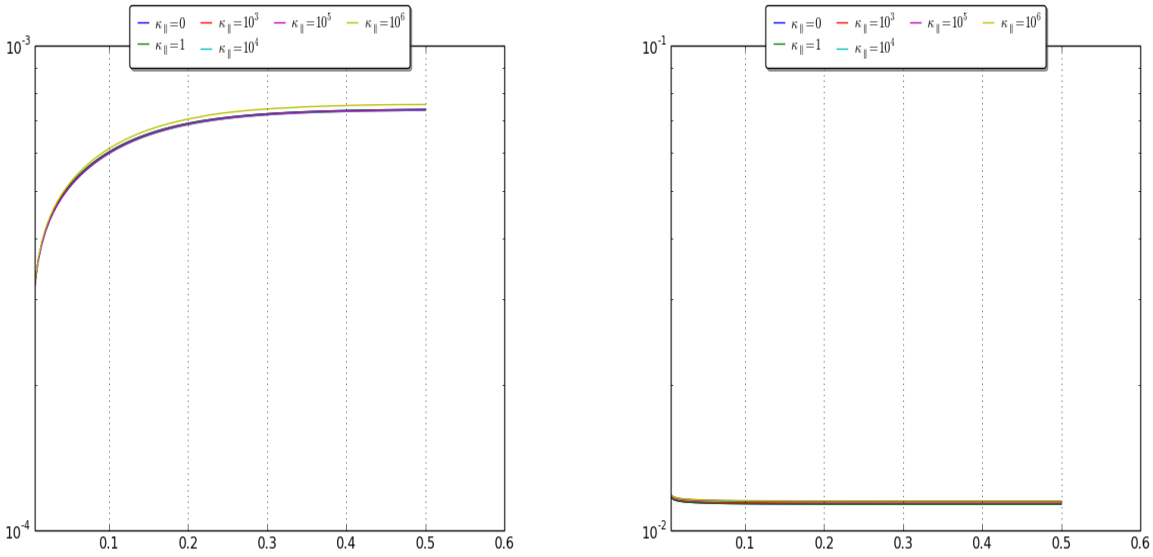


Figure 14: Short time behavior of L^2 and H^1 error norms evolution for the Steady Anisotropic Diffusion using MESH1 for $dt = 5.10^{-3}$ and a grid $16 \times 8 \times 8$.

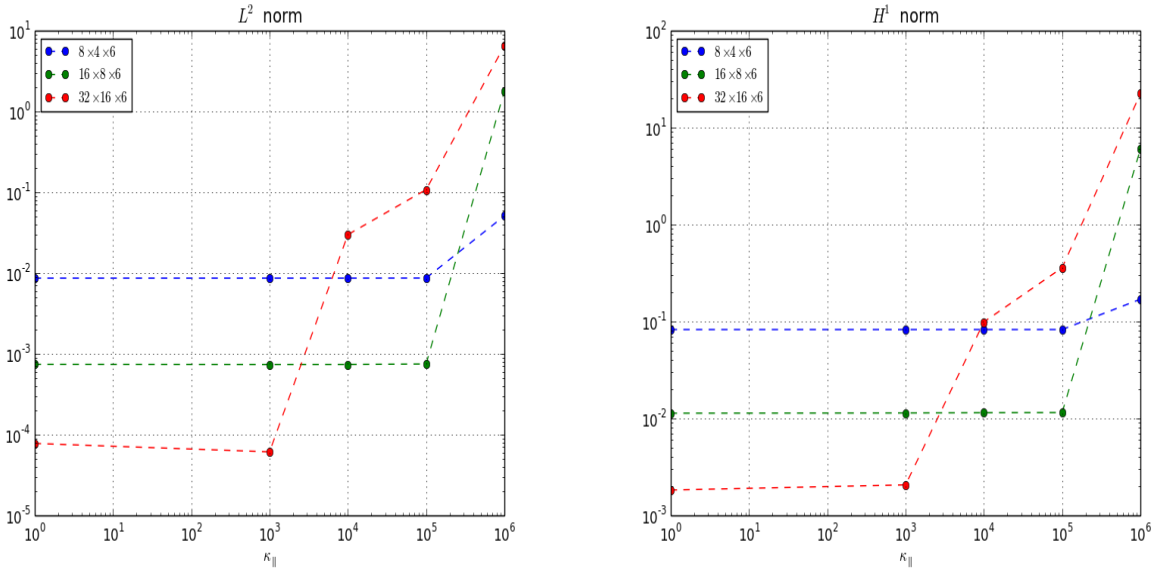


Figure 15: Long time behavior of L^2 and H^1 error norms for different values of $\kappa_\parallel$ at time $T_{final} = 20$ and $dt = 1$ using MESH1

$$u(t=0,\mathbf{x}) = e^{-\frac{1}{\sigma^2}\left((R-R_1)^2 - R_1^2\varphi^2 - Z^2\right)} \tag{4.32}$$

where $R_1 = 3.5$, $\sigma = 0.1$, $\kappa_I = 0$ and $\kappa_\parallel = 1$. In this case, energy on any given toroidal surface should be conserved. In order to validate our simlation, we compute the mean value of the numerical solution on a toroidal

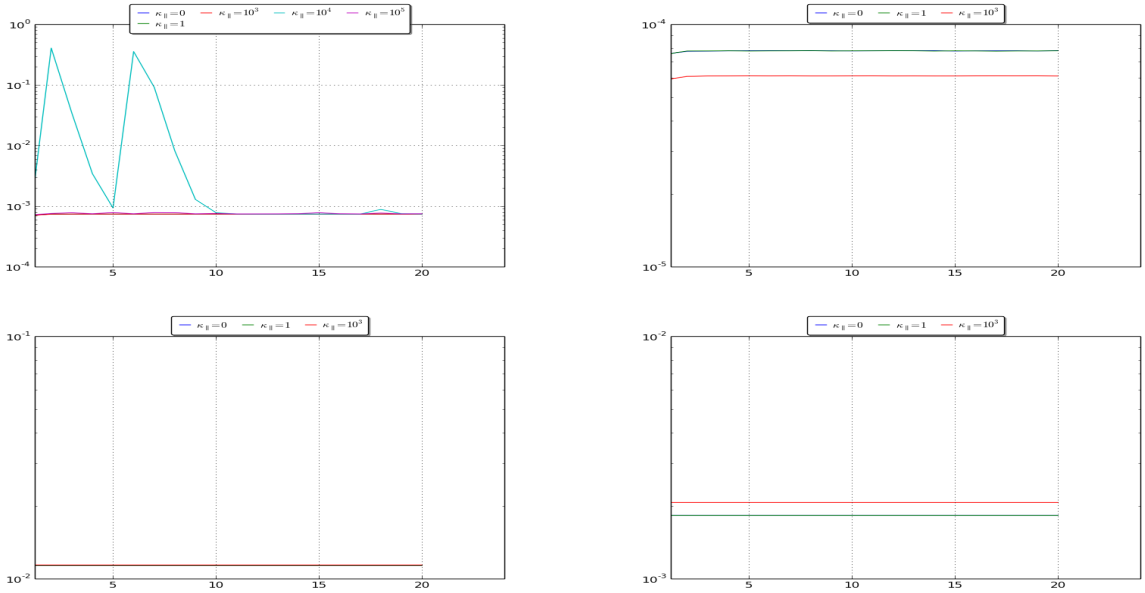


Figure 16: Long time behavior of (first line) L^2 and (second line) H^1 error norms evolution for the Steady Anisotropic Diffusion using MESH1 for and $dt = 1$ and a grid (left) $16 \times 8 \times 6$ and (right) $32 \times 16 \times 6$.

surface

$$\bar{u}(t,r) = \int_0^{2\pi} \int_0^{2\pi} Ru \; d\varphi d\theta \tag{4.33}$$

5 Conclusions

We have developed a finite element context for toroidal geometries. This finite element context includes B-splines and Bezier-Hermite formulations of the approximated space defined on quadrangular (2D) and hexahedral (3D) patches. The construction of the approximated space take advantage of the tensor product decomposition related to 3D toroidal geometries. When possible, C1 continuity is enforced at the patches interfaces and curved isoparametric formulation has been performed. This numerical strategy, applied to Grad-Shrafranov elliptic equation, gives the expected third order of convergence. For anisotropic diffusion, we have observed the usual asymptotic error for very strongly anisotropy. Future work will focus on this behavior to improve the accuracy either by meshes alignment or by asymptotic preserving preconditioning.

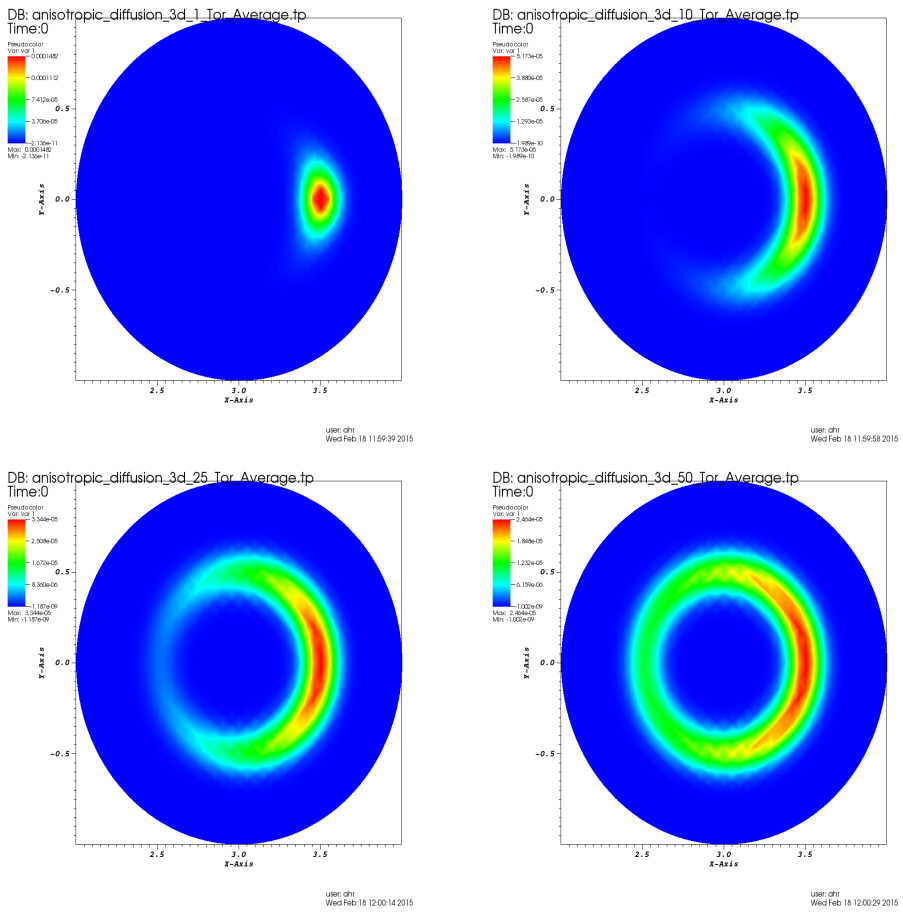


Figure 17: Gaussian pulse test: evolution of $\int_0^{2\pi} u \, d\varphi$ for MESH1 and a grid $32 \times 32 \times 6$ at time $t = 0, 10, 25, 50$.

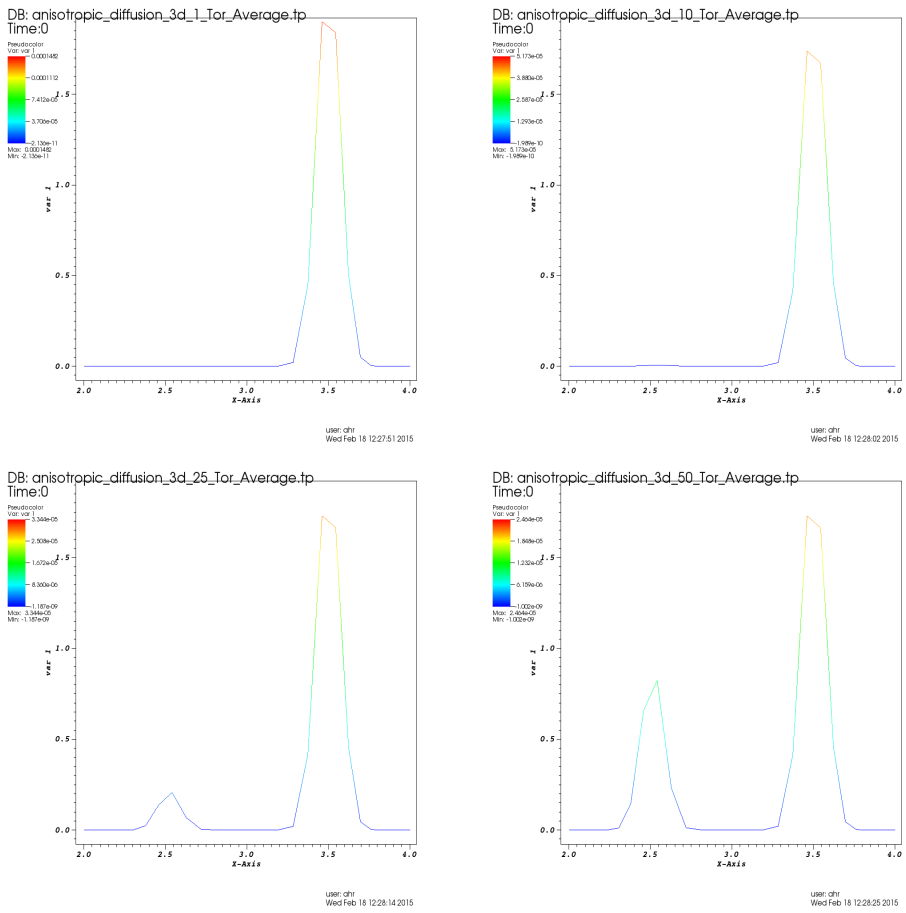


Figure 18: Gaussian pulse test: evolution of $\int_0^{2\pi} u \, d\varphi$ for MESH1 and a grid $32 \times 32 \times 6$ at time $t = 0, 10, 25, 50$ for $Z = 0$.

References

[1] M. Bécoulet, Orain, and al. Mechanism of edge localized mode mitigation by resonant magnetic perturbations. *Phys. Rev. Lett.*, 113:115001, Sep 2014.

[2] Olivier Czarny and Guido Huysmans. Bézier surfaces and finite elements for mhd simulations. *J. Comput. Phys.*, 227:7423–7445, August 2008.

[3] C. DeBoor. *A practical guide to splines*. Springer-Verlag, New York, applied mathematical sciences 27 edition, 2001.

[4] Qi-Xing Huang, Shi-Min Hu, and Ralph R. Martin. Fast degree elevation and knot insertion for b-spline curves. *Computer Aided Geometric Design*, 22(2):183 – 197, 2005.

[5] T.J.R. Hughes, Y. Bazilevs, L. Beirào de Veiga, J.A. Cottrell, and G. Sangalli. Isogeometic analysis: approximation, stability and error estimates for h-refined meshes. *Mathematical Models and Methods in Applied Sciences (M3AS)*, 16:1031–1090, 2006.

[6] G.T.A. Huijsmans and A. Loarte. Non-linear mhd simulation of elm energy deposition. *Nuclear Fusion*, 53(12):123023, 2013.

[7] G T A Huysmans, S Pamela, E van der Plas, and P Ramet. Non-linear mhd simulations of edge localized modes (elms). *Plasma Physics and Controlled Fusion*, 51(12):124012, 2009.

[8] G.T.A. Huysmans and O. Czarny. Mhd stability in x-point geometry: simulation of elms. *Nuclear Fusion*, 47(7):659, 2007.

[9] W. Tiller L. Piegl. *The NURBS Book*. Springer-Verlag, Berlin, Heidelberg, 1995. second ed.

[10] Andrea Mentrelli and Claudia Negulescu. Asymptotic-preserving scheme for highly anisotropic non-linear diffusion equations. *J. Comput. Physics*, 231(24):8229–8245, 2012.

[11] F. Orain, M. Bécoulet, G. Dif-Pradalier, G. Huijsmans, S. Pamela, E. Nardon, C. Passeron, G. Latu, V. Grandgirard, A. Fil, A. Ratnani, I. Chapman, A. Kirk, A. Thornton, M. Hoelzl, and Cahyna P. Non-linear magnetohydrodynamic modeling of plasma response to resonant magnetic perturbations. *Physics of Plasmas (1994-present)*, 20(10):–, 2013.

[12] Andras Pataki, Antoine J. Cerfon, Jeffrey P. Freidberg, Leslie Greengard, and Michael OøNeil. A fast, high-order solver for the gradøshafranov equation. *Journal of Computational Physics*, 243(0):28 – 45, 2013.

[13] A. Ratnani. Caid : Computer aided isogeometric design tool using python. https://github.com/ratnania/caid/tree/devel.

[14] A. Ratnani. Python for isogeometric analysis. https://github.com/ratnania/pigasus/tree/devel.