An Isogeometric Analysis Approach for the study of the gyrokinetic quasi-neutrality equation

Nicolas Crouseilles, Ahmed Ratnani, Eric Sonnendrücker

An Isogeometric Analysis Approach for the study of the gyrokinetic quasi-neutrality equation

Nicolas Crouseilles * Ahmed Ratnani † Eric Sonnendrücker ‡

Keywords: isogeometric analysis; NURBS; gyrokinetic quasi-neutrality equation; numerical simulations.

Abstract

In this work, a new discretization scheme of the gyrokinetic quasi-neutrality equation is proposed. It is based on Isogeometric Analysis; the IGA which relies on NURBS functions, seems to accommodate arbitrary coordinates and the use of complicated computation domains. Moreover, arbitrary high order degree of basis functions can be used. Here, this approach is successfully tested on elliptic problems like the quasi-neutrality equation.

1 Introduction

Nowadays, the modeling of magnetized plasmas is a key issue for controlled thermonuclear fusion. In practice, the study of such plasmas requires solving the Maxwell equations coupled to the computation of the plasma response. Different ways are possible to compute this response: the fluid or the kinetic description. Obviously solving the full Vlasov equation involves the discretization of the six-dimensional phase space, which is a challenging problem. On the other hand, the fluid approach seems to be insufficient when one wants to study the behavior of zonal flow, or the interaction between waves and particles for example (see [4, 10]).

In the context of strongly magnetized plasmas however, the motion of the particles is particular since it is confined around the magnetic field lines; the frequency of this cyclotron motion is faster than the frequencies of interest. Hence, averaging the Vlasov equation over the cyclotron motion reduces the dimensionality, and numerical simulations, even if they remain very costly, become possible (see [8, 11]). These simulations are performed using particles methods or a phase space grid (Eulerian methods) but the computation of the electric potential is always performed on a physical $3D$ grid. Moreover, the configuration of a tokamak is such that the physics is highly anisotropic and structures along the magnetic field lines are quite larger than across the magnetic field lines. In order to reduce the numerical effort which is huge anyway, it is quite important to use this information in order to define the grid resolution in each direction. Indeed, if the grid is aligned or almost aligned with the magnetic field lines, it is possible to use a lot fewer points in the parallel direction than in the transverse direction.

It is of great importance to develop a quasi-neutrality solver that is flexible with respect to geometry and that can provide high order accuracy. In this work, we are investigating an approach which can accommodate arbitrary coordinates and a complicated geometry of the computation domain. In [2] Czarny and Huysmans used Bézier elements for MHD simulations. Bézier surfaces are the most basic tool in (CAD) computer aided design. However, it cannot preserve the exact geometry. The Isogeometric Analysis (IGA), which has been introduced recently by Hughes et al. [14], seem to provide all these features. The IGA relies on NURBS functions, which are a generalization of Spline functions and provides an exact modeling of large classes of computational domains including conics and all spline surfaces. Moreover they rely on a cartesian grid of the parameter space and are fairly easy to use even using spline basis functions of arbitrary degree. Moreover for domains that can be represented using a periodic angular variable as is the case for the poloidal plane of the tokamak, we were able to develop a fast solver that is comparable in computation time, for any spline degree, with the spectral and Finite Difference solver used in Gysela [8].

*INRIA-Nancy Grand Est, Projet CALVI; mail: crouseil@math.unistra.fr
†INRIA-Nancy Grand Est, Projet CALVI; mail: ratnani@math.unistra.fr
‡IRMA-Université de Strasbourg et INRIA-Nancy Grand Est, Projet CALVI; mail: sonnen@math.unistra.fr

The results presented in this work use a bean shaped domain corresponding to a poloidal cut of the tokamak plasma. The next step will be to couple this field solver with a gyrokinetic solver. Moreover, being based on the projection of the approximated function into a finite dimensional space, it can not only deal with differential operators but also integral operators such as those involved in an exact computation of the double gyroaverage, which is up to now approximated by the transverse Laplacian.

Finally, let us mention a point which is particularly useful for parallel computations. Due to the adiabatic assumption for electrons, a nonlocal term intervenes in the equation which is very penalizing for massive parallelization of full gyrokinetic codes. In this work, we propose an algorithm which enables an interesting decoupling of the quasi-neutrality equation. Indeed, by decomposing the electric potential between its average on a magnetic surface and the difference between itself and this average, it is possible to solve the quasi-neutrality equation by: first, solving a $1D$ radial ordinary differential equation, and second, solving N_φ poloidal $2D$ equations (where N_φ is the number of poloidal planes). This latter equations are solved using the NURBS approach. The two equations are local and seems suitable for massively parallel computations.

The rest of the paper is organized as follows: After some recalls about the quasi-neutral approximation of Poisson's equation in gyrokinetics, the IGA approach is presented and applied to simple examples and the fast solver is described in the case of polar coordinates. Then, the last part is devoted to the numerical solution of the quasi-neutrality equation satisfied by the electric potential.

2 Quasi-neutrality equation

The Poisson equation enables to determine the electric potential ϕ as a function of the distribution function

$$\nabla^2\phi = -4\pi|e|(n_i - n_e).$$

where n_i (resp. n_e) stands for the ion density (resp. the electron density). In classical tokamak plasmas, the Debye length is one order smaller than the Larmor radius so that (see [16]), the quasi-neutrality equation is given by

$$n_i(x) = n_e(x), \tag{2.1}$$

The ion density n_i is evaluated at particles position, and can be computed from the solution of the gyrokinetic equation (posed on guiding-center position). At the first gyrokinetic order, n_i can be written

$$n_i(x) = \int \mathcal{J}(f+g)(x,v)dv, \quad \mathcal{J}(f)(x) = \frac{1}{2\pi}\int_0^{2\pi} f(x+\rho)d\varphi, \quad \rho = |\rho|(\cos\varphi, \sin\varphi), \tag{2.2}$$

where f stands for the gyrocenter distribution and g is the first order correction, which can be approximated (as in [16, 17, 9]) by

$$g = \partial_\mu F_M(\phi - \mathcal{J}(\phi)), \text{ with } F_M = \frac{n_0}{\sqrt{2\pi T}}\exp(-m\mu/(2T)),$$

where $n_0 = n_0(r)$ is an equilibrium density and $T = T(r)$ is an equilibrium ion temperature. These two profiles are generally given. Hence, injecting this last expression into (2.2) leads to

$$n_i(x) = \bar{n}_i(x) + \frac{n_0}{T}(\phi(x) - \tilde{\phi}(x)), \tag{2.3}$$

where $\bar{n}_i(x) = \int \mathcal{J}(f)(x,v)dv$ and $\tilde{\phi}(x) = \int_0^\infty \mathcal{J}^2(\phi)(x)\exp(-\mu)d\mu$ is the second gyroaverage transformation of ϕ. In Fourier variables, this term has a compact form using the function $\Gamma_0(b) = \int_0^{+\infty}\exp(-x)\mathcal{J}_0^2(bx)dx$, where $\mathcal{J}_0$ is the Bessel function. Then (2.3) becomes in Fourier variables

$$\widehat{n}_i = \widehat{\bar{n}}_i + n_0\frac{\widehat{\phi}}{T_i}(1 - \Gamma_0(b)),$$

with $b = k_\perp^2\rho_i^2$, $\rho_i^2 = T/B^2$. By expanding the Gamma function Γ_0 using a Padé approximation (see [6, 9, 17]), we obtain

$$n_i = \bar{n}_i + \frac{1}{B^2}\nabla_\perp\cdot(n_i\nabla_\perp\phi).$$

For the electron density, an adiabatic assumption is often performed so that we can write the following equality

$$n_e(x) = n_0 + \frac{n_0}{T_e}(\phi - \langle\phi\rangle),$$

where the operator $\langle\cdot\rangle$ is an integration over constant magnetic surfaces. The quasi-neutrality equation then reads,

$$-\frac{1}{B^2}\nabla_\perp \cdot (n_i \nabla_\perp \phi) + \frac{n_0}{T_e}(\phi - \langle\phi\rangle) = \bar{n}_i - n_0.$$

Linearization around the equilibrium density n_0 together with the approximation $B \approx B_0$ are usually performed (see [9, 8]); these simplifications provides only a radial dependence for the anisotropic factor

$$-\frac{1}{B_0^2}\nabla_\perp \cdot (n_0 \nabla_\perp \phi) + \frac{n_0}{T_e}(\phi - \langle\phi\rangle) = \bar{n}_i - n_0. \tag{2.4}$$

In the rest of the paper, the equation (2.4) is intended to be solved.

3 NURBS

Let us first recall some important properties on B-splines and their generalization to NURBS. These will be used to define the computational domain and also as basis functions for our Finite Element formulation.

3.1 Splines

Let $T = (t_i)_{1 \leqslant i \leqslant N+k}$ be a non-decreasing sequence of knots.

Definition 1 (B-Spline) *The i-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,

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

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

- B-splines are piecewise polynomial of degree $p = k - 1$
- Positivity
- Compact support; the support of N_j^k is contained 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 has a multiplicity m then the B-spline is $\mathcal{C}^{(p-m)}$ at t

Let $(P_i)_{1 \leqslant i \leqslant N} \in \mathbb{R}^d$ be a sequence of control points, forming a control polygon.

Definition 2 (B-Spline curve) *The B-spline curve in $\mathbb{R}^d$ associated to $T = (t_i)_{1 \leqslant i \leqslant N+k}$ and $(P_i)_{1 \leqslant i \leqslant N}$ is defined by :*

$$\boldsymbol{M}(t) = \sum_{i=1}^N N_i^k(t) \boldsymbol{P}_i$$

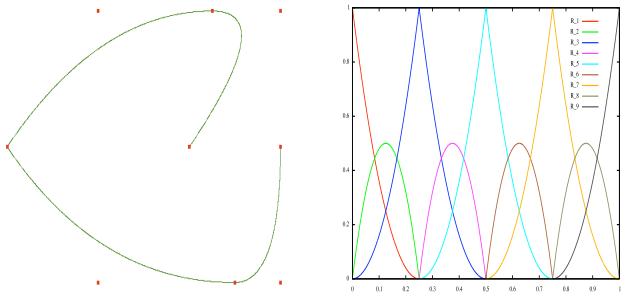

Figure 1: (left) A B-spline curve and its control points, (right) B-splines functions used to draw the curve. $N = 9$, $k = 2$, $T = \{000, \frac{1}{4}\frac{1}{4}, \frac{1}{2}\frac{1}{2}, \frac{3}{4}\frac{3}{4}, 111\}$

3.2 Fundamental geometric operations

Once a B-spline curve is defined by a number of control points N, a degree k and a set of knots T. It is possible to define the same curve using more knots or a higher degree. Mesh refinement starting from a given spline or NURBS curve for the boundary can be performed in this way. Let us briefly sketch the algorithms for performing these operations. After modification, we denote by $\widetilde{N}, \widetilde{k}, \widetilde{T}$ the new parameters. $(\mathbf{Q}_i)$ are the new control points.

3.2.1 Knot insertion

One can insert a new knot t, where $t_j \leqslant t < t_{j+1}$. For this purpose we use the DeBoor algorithm:

$$\widetilde{N} = N + 1$$

$$\widetilde{k} = k$$

$$\widetilde{T} = \{t_1, .., t_j, t, t_{j+1}, .., t_{N+k}\}$$

$$\alpha_i = \begin{cases} 1 & 1 \leqslant i \leqslant j-k+1 \\ \frac{t-t_i}{t_{i+k-1}-t_i} & j-k+2 \leqslant i \leqslant j \\ 0 & j+1 \leqslant i \end{cases}$$

$$\mathbf{Q}_i = \alpha_i \mathbf{P}_i + (1-\alpha_i)\mathbf{P}_{i-1}$$

3.2.2 Order elevation

We can elevate the order of the basis, without changing the curve. Several algorithms exist for this purpose. We used the one by Huang et al. [13].

$$\widetilde{k} = k + m$$

$$\widetilde{m}_i = m_i + m$$

$$\widetilde{N} = N + ms$$

Differential coefficients are defined as $\widetilde{\mathbf{P}}_i^l$:

$$\widetilde{\mathbf{P}}_i^l = \begin{cases} \widetilde{\mathbf{P}}_i & l = 0 \\ \frac{1}{t_{i+p}-t_{i+l}}(\widetilde{\mathbf{P}}_{i+1}^{l-1} - \widetilde{\mathbf{P}}_i^{l-1}) & l > 0, t_{i+k-1} > t_{i+l} \\ 0 & l > 0, t_{i+k-1} = t_{i+l} \end{cases}$$

$\beta_i = \sum_{l=1}^{i} m_l, 1 \leqslant i \leqslant s-1$, et $\alpha_i = \prod_{l=1}^{i} \frac{k-1-l}{k-1+m-l}, 1 \leqslant i \leqslant k-2$
We present the algorithm by [13]:

1. Compute $\widetilde{\mathbf{P}}_0^j, 0 \leqslant j \leqslant k-1$ et $\widetilde{\mathbf{P}}_{\beta_l}^i$,$1 \leqslant l \leqslant s-1$, $k - m_l \leqslant i \leqslant k-1$
2. Compute $\widetilde{\mathbf{Q}}_0^j = \prod_{l=1}^{j}(\frac{k-l}{k+m-l})\widetilde{\mathbf{P}}_0^j, 0 \leqslant j \leqslant k-1$
3. Compute $\widetilde{\mathbf{Q}}_{\beta_l+ml}^j = \prod_{l=1}^{j}(\frac{k-l}{k+m-l})\widetilde{\mathbf{P}}_{\beta_l}^j$, $1 \leqslant l \leqslant s-1$, $k - m_l \leqslant i \leqslant k-1$
4. Compute $\widetilde{\mathbf{Q}}_{\beta_l+ml+i}^{k-1} = \widetilde{\mathbf{Q}}_{\beta_l+ml}^{(k-1)}$, $1 \leqslant l \leqslant s-1$, $1 \leqslant i \leqslant m$
5. Compute $\widetilde{\mathbf{Q}}_i^0$

Note that there exist other algorithms such those given by (see [22, 21] and others). The one given in [13] is more efficient and much more simple to implement. We can also use a more sophisticated version of this algorithm to do the insertion of new knots while elevating the degree.

3.3 Refinement strategies

Refining the grid can be done in 3 different ways. This is the most interesting aspects of B-splines basis.

- using the patch parameter h, by inserting new knots. This is the h-refinement, it is the equivalent of mesh refinement of the classical finite element method.
- using the degree p, by elevating the B-spline degree. This is the p-refinement, it is the equivalent of using higher finite element order in the classical FEM.
- using the regularity of B-splines, by increasing / decreasing the multiplicity of inserted knots. This is the k-refinement. This new strategy does not have an equivalent in the classical FEM.

J.A. Evans et al. [7] studied the k-refinement using the theory of Kolmogorov n-widths. As we will see in this article, the use of this strategy can be more efficient than the classical p-refinement, as it reduces the dimension of the basis.

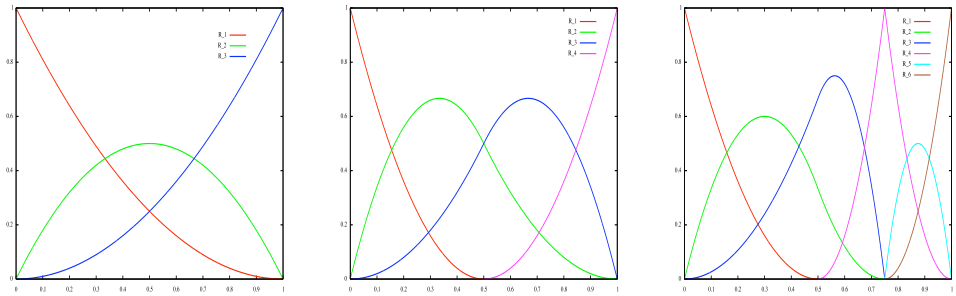

Figure 2: Illustration of h-refinement with $p = 2$, $T = \{000, 111\}$, $T = \{000, \frac{1}{2}, 111\}$ and $T = \{000, \frac{1}{2}, \frac{3}{4}\frac{3}{4}, 111\}$.

3.4 NURBS

Let $\omega = (\omega_i)_{1 \leqslant i \leqslant N}$ be a sequence of non-negative reals. The NURBS functions are defined by a projective transformation:

Definition 3 (NURBS) *The i-th NURBS of order k associated to the knot vector T and the weights ω, is defined by*

$$R_i^k = \frac{\omega_i N_i^k}{\sum_{j=1}^{N} \omega_j N_j^k}.$$

Notice that when the weights are equal to 1 the NURBS are B-splines.

Definition 4 (NURBS curve) *The NURBS curve of order k associated to the knot vector T, the control points* $(\boldsymbol{P}_i)_{1\leqslant i\leqslant N}$ *and the weights* ω*, is defined by*

$$\boldsymbol{M}(t) = \sum_{i=1}^{N} R_i^k(t)\boldsymbol{P}_i$$

Definition 5 (NURBS surface) *The NURBS surface of order k associated to the knot vectors* $\{T^{(1)}, T^{(2)}\}$*, the control points* $(\boldsymbol{P}_{i,j})_{1\leqslant i\leqslant N_1, 1\leqslant j\leqslant N_2}$ *and the weights* $\{(\omega^{(1)}, \omega^{(2)}\}$*, is defined by*

$$\boldsymbol{M}(t^{(1)}, t^{(2)}) = \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} R_{i,j}(t^{(1)}, t^{(2)})\boldsymbol{P}_{i,j}$$

with $R_{i,j}(t^{(1)}, t^{(2)}) = R_i^{(1)}(t^{(1)})R_j^{(2)}(t^{(2)})$

Remarks NURBS functions inherit all B-splines properties. Remark that in the interior of a knot span, all derivatives exist, and are rational functions with non vanishing denominator.
We present here the definition of the perspective mapping. We construct the weighted control points $\mathbf{P}_i^\omega = (\omega_i x_i, \omega_i y_i, \omega_i z_i, \omega_i)$. Then we define the B-spline curve in four-dimensional space as

$$\mathbf{M}^\omega(t) = \sum_{i=1}^{N} N_i^k(t)\mathbf{P}_i^\omega.$$

For fundamental geometric operations on NURBS curves, we use the latest transformation and algorithms on B-spline curves.
NURBS functions allow us to model, exactly, much more domains than B-splines. In fact, all conics can be exactly represented with NURBS. For more details, see [21].

3.5 Examples

3.5.1 Circle

One can draw a circle using only 9 control points, and the parameters:
$N = 9$, $p = 2$, $T = \{000, \frac{1}{4}\frac{1}{4}, \frac{1}{2}\frac{1}{2}, \frac{3}{4}\frac{3}{4}, 111\}$. Control points and weights are given in the following table:

i	P_i	ω_i
1	$(1,0)$	1
2	$(1,1)$	$\frac{1}{\sqrt{2}}$
3	$(0,1)$	1
4	$(-1,1)$	$\frac{1}{\sqrt{2}}$
5	$(-1,0)$	1
6	$(-1,-1)$	$\frac{1}{\sqrt{2}}$
7	$(0,-1)$	1
8	$(1,-1)$	$\frac{1}{\sqrt{2}}$
9	$(1,0)$	1

3.5.2 2 circles

To draw a circle inside another circle we need the following parameters:

$$N_1 = 9, p_1 = 2, T^{(1)} = \{000, \frac{1}{4}\frac{1}{4}, \frac{1}{2}\frac{1}{2}, \frac{3}{4}\frac{3}{4}, 111\}$$

and

$$N_2 = 3, p_2 = 1, T^{(2)} = \{00, \frac{1}{2}, 11\}.$$

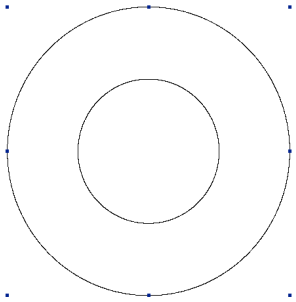

Figure 3: Domain plot and the exterior control points.

3.5.3 Ellipse

To draw an ellipse we can change the scaling of control points coordinates.

3.5.4 Modeling a Tokamak

Starting with an ellipse, one can modify the control points and the weights to have a simplified model of tokamak.

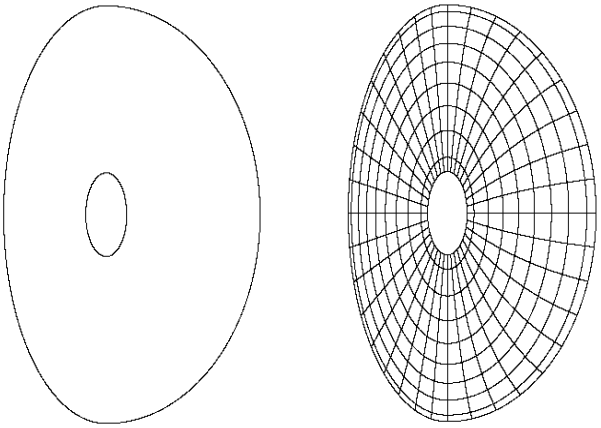

Figure 4: Poloidal plane designing of a tokamak. (left) a tokamak model, (right) tokamak after h-refinement.

We can approximate a domain, using data points. We can refer to the works of [19, 12].

4 The IGA approach for elliptic problems

The first step to use NURBS as finite element basis functions, is to define the patch, *i.e* the parametric domain, and therefore the mapping F, that maps the patch into the physical domain.

4.1 Patch

When the edge can be described as a NURBS curve, we can construct an initial patch, by adding some control points that rise from the definition of the knot vectors used. From this coarse mesh, we can then, use $h/p/k$ refinements to create the grid. We can also use multiple patches to describe more complex domains [1]. NURBS models suffers from the fact that control points must lie in a rectangular grid. Hence, we will have a lot of superfluous control points, that might exist only to satifsy this constraint. Sederberg et al. [24]

defined the notion of T-splines that allow us to reduce the number of those control points. In [5] Dörfel et al. used T-splines for local h-refinement in isogeometric analysis.

4.2 Grid generation

For this purpose, we use alternatively h and p-refinement. The minimal degree of the basis functions is imposed by the domain design. When inserting knots, we can use uniformly-spaced knots or non uniformly-spaced ones.

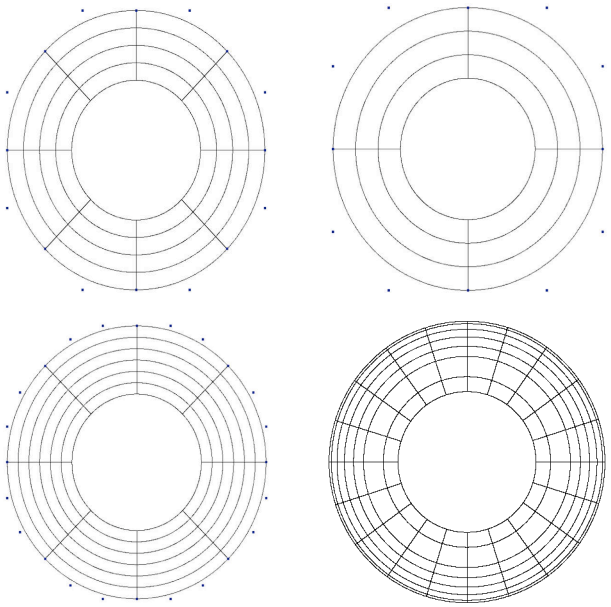

Figure 5: Grid generation. 1^{st} line: (left) after h-refinement, $N_1 = 17$, $N_2 = 5$, (right) after p-raffinement, $p_1 = p_2 = 3$. 2^{nd} line: (left) after h-refinement $p_1 = p_2 = 3$, (right) using unstructured mesh, $p_1 = p_2 = 2$.

4.3 Variational formulation of an elliptic problem

A general elliptic problem writes

$$\begin{cases} -\nabla \cdot (A\nabla u) + cu = f, & \Omega \\ u = g, & \Gamma_D \\ \nabla u \cdot \mathbf{n} = k, & \Gamma_N \end{cases} \tag{4.5}$$

where $\Gamma_D \bigcup \Gamma_N = \partial\Omega$ and $\Gamma_D \bigcap \Gamma_N = \emptyset$. A is some given tensor and c a given nonnegative function. Let define the spaces $\mathcal{X} = H^1(\Omega)$, $\mathcal{S} = \{v/v \in H^1(\Omega), v|_{\Gamma_D} = g\}$, and $\mathcal{V} = \{v/v \in H^1(\Omega), v|_{\Gamma_D} = 0\}$. The variational formulation of (4.5) is:

Find $u \in \mathcal{S}$ such that:

$$a(w,u) = l(w), \qquad \forall w \in \mathcal{V} \tag{4.6}$$

where

$$a(w,v) = \int_\Omega A\nabla v \cdot \nabla w + \int_\Omega c\, v\, w,$$

and

$$l(w) = \int_\Omega fw + \int_{\Gamma_N} kw.$$

4.4 The discrete variational formulation

Let $\mathcal{X}_h, \mathcal{V}_h$ be finite linear sub-spaces of $\mathcal{X}, \mathcal{V}$, where h is intended to tend to zero. $\mathcal{S}_h$ is a finite dimensional approximation of $\mathcal{S}$.
The discrete variational formulation is

Find $u_h = v_h + g_h \in \mathcal{S}_h$ *such that:*

$$a(w_h, u_h) = l(w_h), \qquad \forall w_h \in \mathcal{V}_h, \tag{4.7}$$

where g_h, h are given, and $g_h \in \mathcal{S}_h$.
By re-ordering the basis functions (NURBS) to have : $R_i|_{\Gamma_D} = 0, \forall i \in \{1, .., n - n_D\}$, we then obtain a construction of a basis of $\mathcal{V}_h$ *i.e.* :

$$\forall w_h \in \mathcal{V}_h, w_h = \sum_{i=1}^{n-n_D} [w_h]^i R_i$$

$g^h \in \mathcal{S}_h$ such that $[g_h]^i = 0, \forall i \in \{1, .., n - n_D\}$ and so $g^h = \sum_{i=n-n_D+1}^{n} [g_h]^i R_i$. Then, $u_h \in \mathcal{S}_h$ is of the form

$$u_h = \sum_{i=1}^{n-n_D} [u_h]^i R_i + \sum_{i=n-n_D+1}^{n} [g_h]^i R_i.$$

We obtain classically the linear system:

$$\Sigma [u_h] = L,$$

where the following notations have been used

$$\Sigma = (\Sigma_{i,j})_{1 \leqslant i,j \leqslant n-n_D},$$
$$L = (L_i)^T_{1 \leqslant i \leqslant n-n_D},$$
$$[u_h] = ([u_h]^i)^T_{1 \leqslant i \leqslant n-n_D}.$$

4.5 Computing element integrals

Let Q be a cell in the physical domain. $\widetilde{Q}$ is the parametric associated cell such that $Q = F(\widetilde{Q})$. let J_F be the jacobian of the transformation F, that maps any parametric domain point (ξ, η) into physical domain point (x, y).

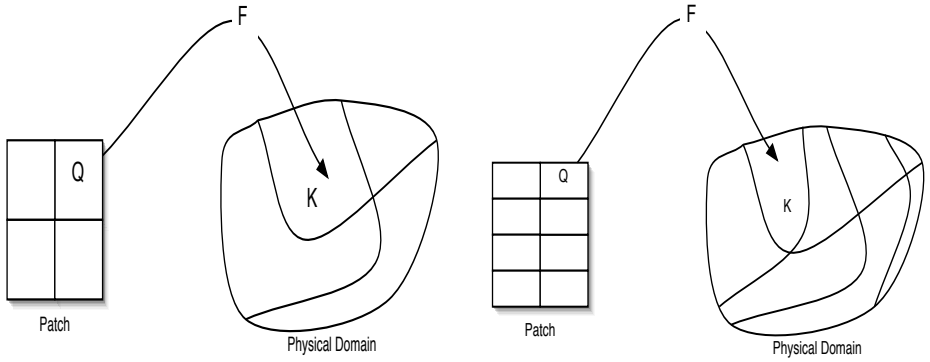

Figure 6: Mapping from the patch to the physical domain: (left) initial patch, (right) patch after h-refinement in the η direction

For any function v of (x, y) we associate its representation in the parametric domain

$$\widetilde{v}((\xi, \eta)) := v \circ F((\xi, \eta)) = v((x, y)).$$

The basis functions R_i will not be affected by these changes, the reader can always know if the we are working in the physical or parametric domain thanks to

$$(x, y) = F(\xi, \eta), x = \alpha(\xi, \eta) \text{ and } y = \beta(\xi, \eta).$$

We then have, for v, w defined on $F(\widetilde{Q}) = Q$, with $J_{F^{-1}}$ the jacobian of F^{-1},

$$\begin{aligned} \int_Q (A\nabla v) \cdot \nabla w \; dx \; dy &= \int_{\widetilde{Q}=F^{-1}(Q)} (AJ_{F^{-1}}\widetilde{\nabla}\widetilde{v}) \cdot (J_{F^{-1}}\widetilde{\nabla}\widetilde{w}) \frac{1}{\det J_{F^{-1}}} \; d\xi \; d\eta \\ &= \int_{\widetilde{Q}} \widetilde{\nabla}\widetilde{v}^T \, J_{F^{-1}}^T \, A^T \, J_{F^{-1}}\widetilde{\nabla}\widetilde{w} \; \det(J_F) \; d\xi \; d\eta. \end{aligned} \tag{4.8}$$

Introducing the notations

$$\alpha_1 = \frac{\partial \alpha}{\partial \xi}, \alpha_2 = \frac{\partial \alpha}{\partial \eta}, \beta_1 = \frac{\partial \beta}{\partial \xi}, \beta_2 = \frac{\partial \beta}{\partial \eta},$$

we have for the determinant of the Jacobian $\det(J_F) = \alpha_1\beta_2 - \alpha_2\beta_1$ and for $J_{F^{-1}}$

$$J_{F^{-1}} = \frac{1}{\det(J_F)} \begin{pmatrix} \beta_2 & -\beta_1 \\ -\alpha_2 & \alpha_1 \end{pmatrix}.$$

Now, let us define $\Theta = J_{F^{-1}}^T A^T J_{F^{-1}}$, so that (4.8) becomes

$$\int_Q (A\nabla v) \cdot \nabla w \; dx \; dy = \int_{\widetilde{Q}} \widetilde{\nabla}\widetilde{v}^T \;\; \Theta \;\; \widetilde{\nabla}\widetilde{w} \;\; \det(J_F) \; d\xi \; d\eta.$$

When A is the identity matrix the explicit form of the Θ matrix writes:

$$\Theta = \frac{1}{\det(J_F)^2} \begin{pmatrix} \alpha_2^2 + \beta_2^2 & -\alpha_1\alpha_2 - \beta_1\beta_2 \\ -\alpha_1\alpha_2 - \beta_1\beta_2 & \alpha_1^2 + \beta_1^2 \end{pmatrix}.$$

For the term of order zero, in the left hand side, we have :

$$\int_Q cvw \; dx \; dy = \int_{\widetilde{Q}} \widetilde{c}\widetilde{v}\widetilde{w} \;\; \det(J_F) \; d\xi \; d\eta.$$

For the right hand side part, the computations are easier and lead to

$$\int_Q f \; v \;\; dx \; dy = \int_{\widetilde{Q}} \widetilde{f} \; \widetilde{v} \; \det(J_F) \; d\xi \; d\eta,$$

whereas for the Neumann part, we have

$$\int_{\partial Q \cap \Gamma_N} k \; v \; d\Gamma = \int_{\partial \widetilde{Q} \cap \widetilde{\Gamma}_N} \widetilde{k} \; \widetilde{v} \; \det(J_{F_{|\Gamma}}) d\widetilde{\Gamma}.$$

In practice, we use Gauss-Legendre quadrature to compute those integrals. We give here the number of used quadrature points, per direction, with respect to the spline degree. Notice that, Hughes et al. [15], began the study of efficient numerical quadrature formulae for NURBS.

p	2	3	4	5	6	7
N_{GL}	3	4	5	6	7	8

Figure 7: The number of used Gauss-Legendre points with respect to the spline degree.

Hence, we get,

$$\Sigma_{i,j} = \int_\Omega A\nabla R_i \cdot \nabla R_j \, dx\, dy + \int_\Omega c\, R_i\, R_j \, dx\, dy$$
$$= \sum_{\widetilde{Q}} \left[\int_{\widetilde{Q}} \widetilde{\nabla} \widetilde{R}_i^T \;\; \Theta \;\; \widetilde{\nabla} \widetilde{R}_j \;\; \det(J_F)\, dr\, d\theta + \int_{\widetilde{Q}} \widetilde{c} \widetilde{R}_i \widetilde{R}_j \;\; \det(J_F)\, dr\, d\theta \right], \quad (4.9)$$

and

$$L_i \quad = \quad \int_\Omega f R_i \;\; + \;\; \int_{\Gamma_N} h R_i \quad = \quad \sum_{\widetilde{Q}} \left[\int_{\widetilde{Q}} \widetilde{f}\; \widetilde{R}_i \;\; \det(J_F)\; d\xi\; d\eta + \int_{\partial \widetilde{Q} \cap \widetilde{\Gamma}_N} \widetilde{k}\; \widetilde{R}_i \;\; \det(J_{F_{|\Gamma}}) d\widetilde{\Gamma} \right]. \quad (4.10)$$

As one can see, the matrix Σ is the sum of the stiffness matrix (terms $\int_\Omega A\nabla R_i \cdot \nabla R_j$), and the mass matrix (terms $\int_\Omega c\, R_i\, R_j$).

4.6 A fast solver for polar coordinates

Let us now see how the specific structure of our problem allows us to derive a specific solver which is a lot faster than using a generic sparse matrix solver. For our quasi-neutrality equation, the mapping F is the mapping defining polar coordinates $F(r,\theta) = (r\cos\theta, r\sin\theta)$. Then we have $\det(J_F) = r$. Moreover the matrix A is of the form $a(r)\mathbb{I}$ and $c \equiv c(r)$ is only a function of r. The matrix Θ becomes

$$\Theta = \begin{pmatrix} a(r) & 0 \\ 0 & \frac{a(r)}{r^2}. \end{pmatrix}$$

On the other hand the basis functions can be written as products of functions of r only and of θ only. Let us also introduce a numbering using the two indices of our logical patch $\widetilde{Q}$ in (r,θ). Thus the degree of freedom with index i will be associated with the grid index (i_r, i_θ) and the basis function R_i is such that $\widetilde{R}_i(r,\theta) = \widetilde{R}_{i_r}(r)\widetilde{R}_{i_\theta}(\theta)$. Plugging this into the element integrals defining the matrix Σ in (4.9) we get

$$\int_{\widetilde{Q}} \widetilde{\nabla} \widetilde{R}_i^T\, \Theta\, \widetilde{\nabla} \widetilde{R}_j \det(J_F)\, d\xi\; d\eta + \int_{\widetilde{Q}} \widetilde{c} \widetilde{R}_i \widetilde{R}_j \det(J_F)\, d\xi\; d\eta$$
$$= \int_{\widetilde{Q}} \left(a(r) \frac{\partial \widetilde{R}_i}{\partial r} \frac{\partial \widetilde{R}_j}{\partial r} + \frac{a(r)}{r^2} \frac{\partial \widetilde{R}_i}{\partial \theta} \frac{\partial \widetilde{R}_j}{\partial \theta} \right) r\, dr\, d\theta + \int_{\widetilde{Q}} \widetilde{c}(r) \widetilde{R}_i \widetilde{R}_j\, r\, dr\, d\theta$$
$$= \int a(r) \widetilde{R}'_{i_r}(r) \widetilde{R}'_{j_r}(r)\, r\, dr \int \widetilde{R}_{i_\theta}(\theta) \widetilde{R}_{j_\theta}(\theta)\, d\theta + \int \frac{a(r)}{r^2} \widetilde{R}_{i_r}(r) \widetilde{R}_{j_r}(r)\, r\, dr \int \widetilde{R}'_{i_\theta}(\theta) \widetilde{R}'_{j_\theta}(\theta)\, d\theta$$
$$+ \int \widetilde{c}(r) \widetilde{R}_{i_r}(r) \widetilde{R}_{j_r}(r)\, r\, dr \int \widetilde{R}_{i_\theta}(\theta) \widetilde{R}_{j_\theta}(\theta)\, d\theta. \quad (4.11)$$

These formulas lead us to two observations. First the elementary matrices decouple into products of integrals in r and integrals in θ so that the final matrices can be written in a Kronecker product structure (explanations and applications of this structure can be found in the article by Van Loan [25] and references therein). The exploitation of this structure for developing fast solvers was developed in [18]. Second the matrices in θ are simple mass and stiffness matrices with no varying parameter inside, so that if the mesh is uniform in θ the matrix will be circulant.

One way to express the Kronecker product structure is to write the unknown degrees of freedom and the right-hand-side as matrices where the terms correspond to the indices (i_r, i_θ). We denote those respectively by U and F. Then, in our case, the linear system can be written

$$K_{ar} U M_\theta + M_{ar} U K_\theta + M_{cr} U M_\theta = F, \quad (4.12)$$

where K_{ar} is the weighted stiffness matrix in r corresponding to the terms $\int a(r) \widetilde{R}'_{i_r}(r) \widetilde{R}'_{j_r}(r)\, r\, dr$, M_{ar} is the weighted mass matrix in r corresponding to the terms $\int \frac{a(r)}{r^2} \widetilde{R}_{i_r}(r) \widetilde{R}_{j_r}(r)\, r\, dr$, M_{cr} is the weighted mass

matrix in r corresponding to the terms $\int c(r)\widetilde{R}_{i_r}(r)\widetilde{R}_{j_r}(r)\,r\,dr$, K_θ is the stiffness matrix in θ corresponding to the terms $\int \widetilde{R}'_{i_\theta}(\theta)\widetilde{R}'_{j_\theta}(\theta)\,d\theta$ and M_θ is the mass matrix in θ corresponding to the terms $\int \widetilde{R}_{i_\theta}(\theta)\widetilde{R}_{j_\theta}(\theta)\,d\theta$. The latter two matrices K_θ and M_θ are circulant, which means that they can be both diagonalized in the same orthonormal basis corresponding to the Fourier modes. This can be expressed by

$$M_\theta = P\Lambda_M P^*, \qquad K_\theta = P\Lambda_K P^*,$$

where Λ_M and Λ_K are the diagonal matrices of the eigenvalues and a multiplication by P corresponds to the normalized Fast Fourier Transform and a multiplication by P^* to its inverse.

This can be exploited for the fast solution of (4.12) bringing the solution of a linear system arising from a 2D problem to sets of smaller systems corresponding to 1D problems. The procedure can be performed with the following algorithm:

1. Multiply system (4.12) on the right by P (amounts to a 1D FFT on each line of F). Then we get
$$K_{ar}UP\Lambda_M + M_{ar}UP\Lambda_K + M_{cr}UP\Lambda_M = FP. \tag{4.13}$$
2. Note that a multiplication on the right by the diagonal matrix of eigenvalues corresponds to multiplying each column of the matrix by the corresponding eigenvalue, which implies that (4.13) corresponds to uncoupled problems on each of the columns of $\hat{U} = UP$. So denoting by $\hat{U}_1, \ldots, \hat{U}_n$ the columns of the matrix $\hat{U}$ and by $\hat{F}_1, \ldots, \hat{F}_n$ the columns of the matrix $\hat{F} = FP$, (4.13) becomes for each column j,
$$(\lambda_{M_j}(K_{ar} + M_{cr}) + \lambda_{K_j}M_{ar})\hat{U}_j = \hat{F}_j \tag{4.14}$$
This is a set of banded systems of size the number of points in the r direction, that can be solved very efficiently using the LAPACK routines DPBTRF for the Cholesky factorization that is only called once at the beginning and then DPBTRS for the solution at each time step.
3. Compute $U = \hat{U}P^*$ by inverse FFT of the lines of $\hat{U}$.

Let us now compute the cost at each time step for a $N_r \times N_\theta$ mesh, disregarding the cost of the Cholesky factorization for the systems in r which needs only to be performed once for a many time steps computation. The algorithm consists of three steps: 1) N_r FFTs which need $O(N_\theta \log_2 N_\theta)$ operations, 2) N_θ up and down sweeps of a Cholesky decomposed banded system which cost $O(N_r)$ each, 3) N_r inverse FFTs which cost $O(N_\theta \log_2 N_\theta)$ operations. So all together the cost is $O(N_r N_\theta \log_2 N_\theta)$ operations, which is almost optimal. This algorithm uses the structure of the system in an optimal manner and only works on dense matrices. A generic sparse systems solvers could not do this.

Numerical results We have tested this new approach to solve an elliptic partial differential equation, using the analytic solution $u(r,\theta) = \sin(2\pi r)\sin(2\pi\theta)$, which solves :

$$-\nabla^2 u(r,\theta) + u(r,\theta) = F(r,\theta), \quad \Omega \tag{4.15}$$

where $\Omega = [0,1] \times [0,1]$, ∇^2 is the Laplacian cartesian. The boundary conditions are :

$$u(r=0,\cdot) = u(r=1,\cdot) = 0 \tag{4.16}$$

and periodic boundary condition on θ.

In Figures 8 and 9, we compare the time CPU spent to solve the linear system, using this approach, *spsolve*, from *scipy* based on SUPERLU solver, for the classical formulation (detailed in the sections before). The test was done on grids 128×128 and 256×256, using different spline order. As one can see, the new method does not really depend on the spline degree, the reason is that, for $1D$ problem, the bandwidth is equal the spline order, whereas it is a quadratic function of the order, in the classical approach. On a grid of 512×512, we spent 0.3 *sec* to solve the linear system, using cubic splines. Using splines of order 8, it took only 0.321 *sec*.

5 Numerical validation

In all tests we have treated, we consider an elliptic problem under Dirichlet boundary condition, *i.e* $u(\mathbf{x}) = 0, \quad \forall \mathbf{x} \in \partial\Omega$.

Spline degree	FIGA	SPLU
1	0.012	0.013
2	0.014	0.046
3	0.014	0.073
4	0.013	0.098
5	0.015	0.124
6	0.015	0.152
7	0.015	0.179

Spline degree	FIGA	SPLU
1	0.008	0.38
2	0.013	3.81
3	0.012	10.69
4	0.016	19.17
5	0.017	31.95
6	0.020	47.01
7	0.023	65.00

Figure 8: CPU-time, in seconds, spent in solving (left) and initializing (right) the linear system, using the new approach, namely Fast IGA, compared to SuperLU. Test done on a grid 128×128

Spline degree	FIGA	SPLU
1	0.074	0.067
2	0.076	0.967
3	0.075	3.505
4	0.075	16.070
5	0.077	32.852

Spline degree	FIGA	SPLU
1	0.021	3.38
2	0.043	31.40
3	0.052	197.31
4	0.060	330.28
5	0.069	415.63

Figure 9: CPU-time, in seconds, spent in solving (left) and initializing (right) the linear system, using the new approach, namely Fast IGA, compared to SuperLU. Test done on a grid 256×256

5.1 Test case 1: Order of convergence for Poisson in polar coordinates

For the validation, we solved the polar coordinate elliptic problem, which fits into our generic elliptic problem (4.5) with A the identity tensor and $c = 0$,

$$-\nabla^2_{r,\theta}\phi(r,\theta) = n(r,\theta),$$

with the following analytical solution:

$$\phi(r,\theta) = \sin\left(\frac{\pi}{r^2_{\max} - r^2_{\min}}(r^2 - r^2_{\min})\right).$$

Injecting the solution into the elliptic problem enables to compute the right hand side n which is given in input of the code. This test enables to check the L^2 norm of the error (in log scale) between the numerical and analytical solution, with respect to the parameter $h = \max\{diam(\widetilde{Q})\}$, for different orders of the basis functions (from order 2 to order 6). This is shown in Figure 10. We verify that the slopes of the different curves correspond to the order of the basis functions. In particular, for high orders, the machine precision is achieved. We also plot in Figure 10 the CPU time as a function of the parameter $h = \max\{diam(\widetilde{Q})\}$, for different orders of the basis functions (from order 2 to order 6). These two last figures gives information for choosing *a priori* the best compromise between precision (which order of the basis function should be chosen) and the CPU time.

5.2 Test case 2: Chaotic solution

Following [20], we first test our solver on a chaotic solution on a polar coordinate Laplacian $-\nabla^2\phi = n$. This analytic solution takes the form

$$\phi_{\text{math}}(r,\theta) = \left[\sin(2\pi\xi) + \epsilon\sum_M B_M \sin(2\pi M\xi)\right]\sum_l A_l \cos(l\theta + \Theta_l), \tag{5.17}$$

where $0 \leq \epsilon \leq 1$, $\xi = (r - r_{\min})/(r_{\max} - r_{\min})$, A_l and B_M are random numbers which range between 0 and 1, with $|M|, |l| (\leq 20$ for the first simulation and ≤ 40 for the second one). Finally, the phase Θ_l is also given

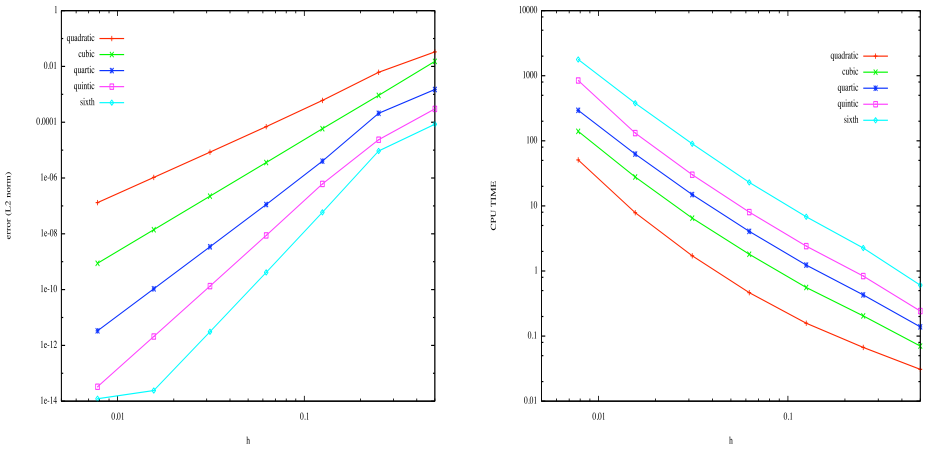

Figure 10: *Validation test* : (left) L^2 error norm, (right) CPU time.

by a random number in $0, 2\pi$. The right hand side of $-\nabla^2\phi = n$ is given by

$$\begin{aligned} n(r,\theta) &= -\frac{1}{r}\left(\partial_r(r\partial\phi_{\text{math}})\right) - \frac{1}{r^2}\partial^2_\theta\phi_{\text{math}} \\ &= -\frac{1}{r}\sigma_r(r,\theta)\sum_l A_l\cos(l\theta+\Theta_l) \\ &+ \left[\sin(2\pi\xi) + \epsilon\sum_M B_M\sin(2\pi M\xi)\right]\sum_l \frac{l^2}{r^2}A_l\cos(l\theta+\Theta_l), \end{aligned}$$

with

$$\begin{aligned} \sigma_r(r,\theta) &= \frac{2\pi}{\Delta r}\cos(2\pi\xi) - \frac{4\pi^2 r}{\Delta r^2}\sin(2\pi\xi) \\ &+ \epsilon\sum_M B_M\left[\frac{2\pi M}{\Delta r}\cos(2\pi M\xi) - \frac{4\pi^2M^2 r}{\Delta r^2}\sin(2\pi M\xi)\right] \end{aligned}$$

In our numerical experiments, ϵ is taken equal to 0.4. Note that the solution satisfies the homogeneous Dirichlet condition at $r = r_{\min} = 0.2$ and $r = r_{\max} = 0.4$ and periodic boundary conditions in the θ direction.

We show on Figure 11, the L^2 norms (in log scale) of the difference between the analytical and numerical solutions as a function of the parameter $h = \max\{diam(\widetilde{Q})\}$. The same observations as before are also available for this test; the slopes of the curves corresponds to the order of the basis, even when a very large number of modes is considered in the solution. In this kind of test high order basis functions enable to better capture the very fine scales of the solution.

Spline degree	Degrees of freedom	L^2 error norm
2	17408	$7.20\ 10^{-3}$
3	18060	$1.07\ 10^{-3}$
4	18720	$1.71\ 10^{-4}$
5	19388	$2.84\ 10^{-5}$
6	20064	$4.81\ 10^{-6}$
7	20748	$1.67\ 10^{-6}$

Table 1: *Nishimura test*: Number of degree of freedom and L^2 norm of the error for each spline degree.

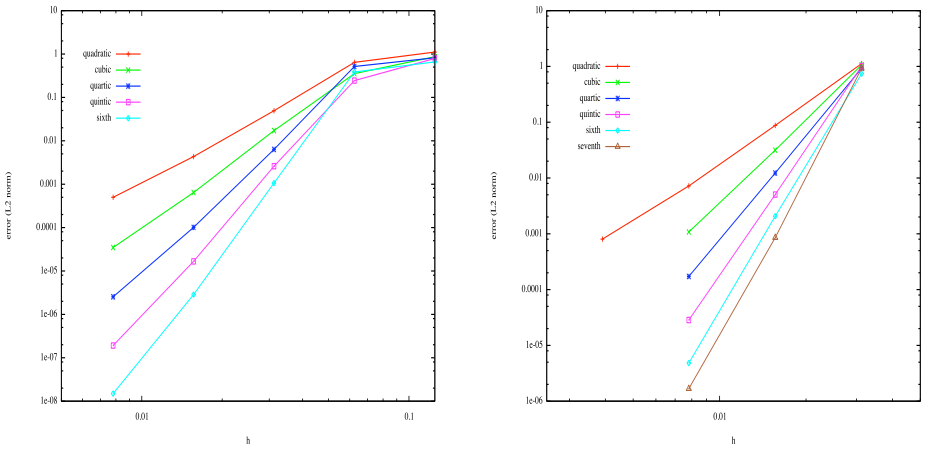

Figure 11: *Nishimura test:* L^2 error norm as a function of h; (left) for 20 modes, (right) 40 modes.

Comparing with Nishimura results [20], we see in Table 1 that the new method allows us to reach same error order, with much fewer degrees of freedom, by increasing the spline order. Let us notice, that to reach an error of 10^{-6}, Nishimura used a grid of $N = 154,560$, with our method, we only need 8 times less ($N = 20064$) degrees of freedom by using sixth or seventh order for spline basis.

6 Numerical solution of the quasi-neutrality equation

This section is devoted to the numerical approximation of the quasi-neutrality equation (2.4) by applying the method introduced above. To that purpose, we will consider a given right hand side (that is to say the distribution function of the ions is given and the coupling with the gyrokinetic equation is not considered), and we focus on the computation of the solution of (2.4) in polar coordinates:

Given $n_0(r), T_e(r)$ two radial profiles and $F(r,\theta,\varphi)$, solve for $\phi(r,\theta,\varphi)$ satisfying

$$-\nabla_\perp \cdot (n_0 \nabla_\perp \phi) + \frac{n_0}{T_e}(\phi - \langle\phi\rangle) = F(r,\theta,\varphi). \tag{6.18}$$

The $\langle\cdot\rangle$ operator refers to the magnetic flux average

$$\langle\phi\rangle(t,r) = \frac{1}{\int\int J(r,\theta)d\theta d\varphi}\int\int \phi(t,r,\theta,\varphi)J(r,\theta)d\theta d\varphi, \tag{6.19}$$

with $J(r,\theta)$ a jacobian defining the poloidal geometry, whereas the given right hand side $F(r,\theta,\varphi)$ reads

$$F(r,\theta,\varphi) = \left(\bar{n}_i(r,\theta,\varphi) - n_0(r)\right),$$

where $\bar{n}_i$ is the ion density, T_e the electronic temperature and n_0 is a correction term taking into account the presence of the electrons. The first term on the left hand side is known as the polarization term which corresponds to the difference between the guiding-center density and that of particles.

The boundary conditions are supposed to be 2π-periodic in the θ, φ variables, whereas the radial boundary conditions are imposed by the Dirichlet condition and writes $\phi(r = r_{\min}, \theta, \varphi) = 0$ and $\phi(r = r_{\max}, \theta, \varphi) = 0$.

The main goal of this section consists in the numerical solution of (6.18). More precisely, we want to derive an efficient method for (6.18), that is to say a method which is as local as possible. Indeed, the principal difficulty is the average term $\langle\phi\rangle$ which is totally nonlocal since it couples every values of θ and φ. The nonlocality is an important obstacle for an efficient parallelization, but also when one wants to use Fourier transforms in the θ, φ variables. We propose a decoupling approach that enables to decompose the

solution of (6.18) into two local problems, one of them reduces to a two-dimensional elliptic type problem of the form (4.5).

The rest of this section presents different ways to solve the quasi-neutrality equation, pointing out the advantages and disagrements of the solvers. Then, some numerical results are shown to compare the performance of the solvers.

6.1 The decoupling approach

In this subsection, we propose a new method to allow us the derivation of a local algorithm for (6.18) in the φ variable. An efficient algorithm can then be developed for its resolution. To this purpose, we first restrict in this section to a Jacobian such that $J(r,\theta) = r$, but the computations for arbitrary Jacobians $J(r,\theta)$ are performed in Appendix A. The main ingredient consists in the decoupling of (6.18) into two local equations: one equation on $\langle\phi\rangle$ which is given by (6.19), and one equation on $\Phi = \phi - \langle\phi\rangle$.

The first equation is simply derived by averaging the left hand side of the quasi-neutrality equation (6.18) with respect to θ and φ variables

$$\frac{1}{(2\pi)^2}\int\int \nabla_\perp\cdot[n_0(r)\nabla_\perp\phi]\,d\theta d\varphi = \left[\left(\frac{n_0(r)}{r}+n_0'(r)\right)\partial_r\langle\phi\rangle + n_0(r)\partial_r^2\langle\phi\rangle\right]. \tag{6.20}$$

Finally, introducing $\langle F\rangle(r) = 1/(2\pi)^2\int\int F(r,\theta,\varphi)d\theta d\varphi$, we have

$$\langle\nabla_\perp\cdot[n_0(r)\nabla_\perp\phi]\rangle = \nabla_\perp\cdot[n_0(r)\nabla_\perp\langle\phi\rangle] = \langle F\rangle, \tag{6.21}$$

from which we can deduce a $1D$ equation for $\langle\phi\rangle = \langle\phi\rangle(r)$

$$-\left[n_0(r)\partial_r^2\langle\phi\rangle + \left(\frac{n_0(r)}{r}+n_0'(r)\right)\partial_r\langle\phi\rangle\right] = \langle F\rangle(r), \tag{6.22}$$

Then, we want to derive an equation satisfied by $\Phi = \phi - \langle\phi\rangle$. This can be done easily by adding and removing the term $\nabla_\perp\cdot[n_0(r)\nabla_\perp\langle\phi\rangle]$ in (6.18)

$$-\nabla_\perp\cdot[n_0(r)\nabla_\perp(\phi-\langle\phi\rangle)] + \frac{n_0}{T_e}[\phi-\langle\phi\rangle] - \nabla_\perp\cdot[n_0(r)\nabla_\perp\langle\phi\rangle] = F(r,\theta,\varphi).$$

By introducing the new unknown $\Phi = \phi - \langle\phi\rangle$ in this last equation, we get

$$-\nabla_\perp\cdot[n_0(r)\nabla_\perp\Phi] + \frac{n_0}{T_e}\Phi - \nabla_\perp\cdot[n_0(r)\nabla_\perp\langle\phi\rangle] = F(r,\theta,\varphi). \tag{6.23}$$

Thanks to (6.21), we can write the equation which is satisfied by $\Phi(r,\theta,\phi) = \phi - \langle\phi\rangle$

$$-\nabla_\perp\cdot[n_0(r)\nabla_\perp\Phi] + \frac{n_0}{T_e}\Phi = F(r,\theta,\varphi) - \langle F\rangle(r). \tag{6.24}$$

More precisely, the equation to solve is

$$-n_0\partial_r^2\Phi - \left[\frac{n_0}{r}+n_0'\right]\partial_r\Phi - \frac{n_0}{r^2}\partial_\theta^2\Phi + \frac{n_0}{T_e}\Phi = F(r,\theta,\varphi) - \langle F\rangle(r). \tag{6.25}$$

Thanks to these straightforward computations, we totally decoupled (6.18) by introducing the new unknowns Φ and $\langle\phi\rangle$. Moreover, equation (6.18) is strictly equivalent to the equations (6.22)-(6.24). This new formulation provides a system of equation which does not include nonlocal term any more. The algorithm is then the following

Algorithm

- solve the $1D$ equation (6.22) to get $\langle\phi\rangle(r)$
- solve the $2D$ equation (6.24) $\forall\varphi$ to get $\Phi(r,\theta,\varphi)$
- compute $\phi = \Phi + \langle\phi\rangle$

Now, an important point consists in a good choice of the numerical approximation to solve the 2 equations (6.22) and (6.24). A two-dimensional solver will provide Φ whereas a one-dimensional solver for (6.22) leads to $\langle\phi\rangle$. By the summation of these two solutions, we can obtain the desired solution $\phi(r,\theta,\varphi)$.

6.2 First approach: spectral + finite differences

The first approach to solve (6.24) consists in the use of a Fourier transform in the θ direction (Φ is periodic in the θ variable since ϕ is). The projection of $\Phi(r,\theta,\varphi)$ onto the Fourier space writes

$$\Phi(r,\theta,\varphi) = \sum_{m=1}^{N} \Phi^m(r,\varphi) e^{im\theta}.$$

so that equation (6.25) is rewritten

$$-\left(n_0 \partial_r^2 \Phi^m + \left(\frac{n_0}{r} + n_0'(r)\right) \partial_r \Phi^m\right) + n_0 \left(\frac{1}{T_e} + \frac{m^2}{r^2}\right) \Phi^m = S^m(r,\phi), \tag{6.26}$$

where

$$S^m(r,\phi) = \begin{cases} F^m(r,\varphi) & \text{if} \quad m \neq 0 \\ F^0(r,\varphi) - \frac{1}{(2\pi)^2} \int F^0(r,\varphi) d\varphi & \text{else,} \end{cases}$$

is the Fourier coefficient of order m of the right hand side of (6.24). We are faced to an ODE for each Fourier mode m; one classical way to solve this problem is the use of a finite difference scheme in the r direction to solve (6.26); this leads to the constitution of a tridiagonal linear system, the resolution of which allows to get the Fourier modes $\Phi^m(r,\varphi)$. An inverse Fourier transform leads to $\Phi(r,\theta,\varphi)$.

The resolution of the ODE (6.22) can be performed in the same way as (6.26), using finite differences of order two.

Remark 6.1 *From the CPU time point of view, the present approach enables us to solve the total quasi-neutrality equation (i.e. to compute $\phi = \langle\phi\rangle + \Phi$) with a cost of order $O(N^2 \ln(N))$ for a $N \times N$ grid in (r,θ). Our NURBS based solver has the same complexity with only the bandwith of the systems in r that increase with the degree of the splines.*

6.3 Second approach: FEM

The second option discretizes directly equation (6.24), that considers two dimensions (r,θ) and a parameter φ. The finite element method described in section 3 is performed here.

For the ODE (6.22), a high order method is required not to penalize the high order achieved in the two-dimensional solution. Here, we propose a collocation method introduced in [3].

Recalling, that the equation 6.24 can be written in the form:

$$-\nabla_\perp \cdot (A\ \nabla_\perp \Phi) + c\ \Phi = g \tag{6.27}$$

where,

$$A = \begin{pmatrix} n_0 & 0 \\ 0 & n_0 \end{pmatrix}, \quad c = \frac{n_0}{T_e}, \quad \text{and,} \quad g = F(r,\theta,\varphi) - \langle F\rangle(r). \tag{6.28}$$

6.4 Numerical results

In the present section, we compare the different approximation of the quasi-neutrality equation (6.18) we proposed in the previous section. The different methods are studied on analytic tests, since we impose a right hand side which is consistent with a solution of (6.18).

Test case 1: In this test, we consider the full quasi-neutrality equation (6.18) in which the profile n_0 and T_e are supposed constant equal to 1, with the following solution

$$\phi(r,\theta,\varphi) = \sin\left(\frac{2\pi}{L}(r - r_{\min})\right) u(\theta,\varphi), \quad \theta,\varphi \in [0,2\pi], r \in [0.2, 0.8],$$

with $L = r_{\max} - r_{\min}$ and $u(\theta,\varphi) = 1/\pi^2(\cos^2\theta \cos^2\varphi)$. This form of solution imposes the average to be $\langle\phi\rangle(r) = \sin\left(\frac{2\pi}{L}(r - r_{\min})\right)$. Injecting this solution in (6.18) leads to an analytical right hand side which is used to recover numerically the true solution. This analytical way enables comparison with the analytical

solution. Let us detail the computations leading to the right hand side. First, we compute the three component of $\nabla^2_\perp \phi$

$$-\partial_r^2\phi = \frac{4\pi^2}{L^2}\sin\left(\frac{2\pi}{L}(r-r_{\min})\right)u(\theta,\varphi),$$
$$-\frac{1}{r}\partial_r\phi = \frac{2\pi}{L}\cos\left(\frac{2\pi}{L}(r-r_{\min})\right)u(\theta,\varphi),$$
$$-\frac{1}{r^2}\partial_\theta^2\phi = \frac{2}{r^2\pi^2}\sin\left(\frac{2\pi}{L}(r-r_{\min})\right)\cos(2\theta)\cos^2\varphi,$$

Then, we can compute $\langle F\rangle$ which will be used in the right hand side of equations (6.22) and (6.24)

$$\langle F\rangle(r) = \frac{4\pi^2}{L^2}\sin\left(\frac{2\pi}{L}(r-r_{\min})\right) - \frac{2\pi}{rL}\cos\left(\frac{2\pi}{L}(r-r_{\min})\right).$$

We also compute $F(r,\theta,\varphi)$ for the resolution of (6.24)

$$\begin{aligned}
F(r,\theta,\varphi) &= \sin\left(\frac{2\pi}{L}(r-r_{\min})\right)u(\theta,\varphi)\left[\frac{4\pi^2}{L^2}+1\right] \\
&+ \frac{2\pi}{L}\cos\left(\frac{2\pi}{L}(r-r_{\min})\right)u(\theta,\varphi) \\
&+ \frac{2}{r^2\pi^2}\sin\left(\frac{2\pi}{L}(r-r_{\min})\right)\cos(2\theta)\cos^2\varphi - \sin\left(\frac{2\pi}{L}(r-r_{\min})\right) \\
&= \sin\left(\frac{2\pi}{L}(r-r_{\min})\right)u(\theta,\varphi)\left[\frac{4\pi^2}{L^2}+1\right] \\
&+ u(r,\theta)\left[\sin\left(\frac{2\pi}{L}(r-r_{\min})\right) - \frac{1}{r}\frac{2\pi}{L}\cos\left(\frac{2\pi}{L}(r-r_{\min})\right)\right] \\
&- \langle F\rangle(r) - \sin\left(\frac{2\pi}{L}(r-r_{\min})\right)
\end{aligned}$$

We then test the decoupling method for which the spectral approach, the IGA-FEM solution is compared to the analytic solution. For the first approach, we fix the number of points in the angular directions θ, φ to 64 whereas the number of points in the radial direction is modified to recover the order of the finite difference method.

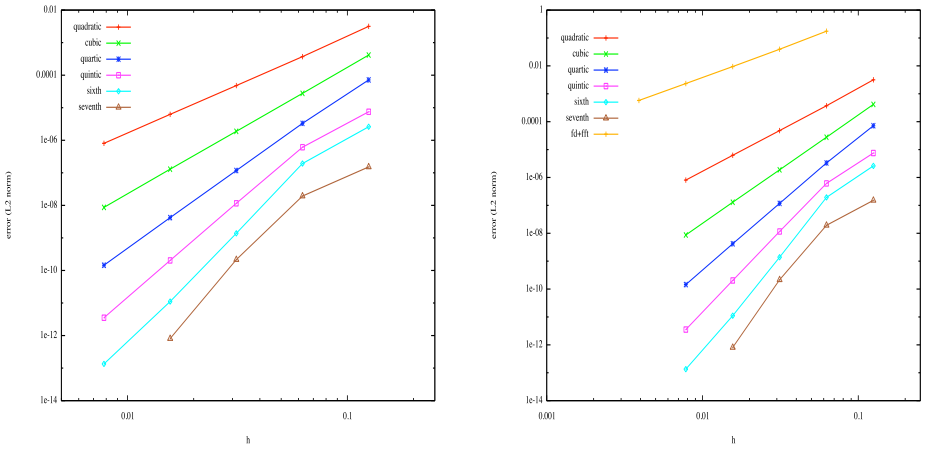

Figure 12: *Test case* 1*:* L^2 error norm, (left) for the elliptic part, (right) for the global problem ODE + FEM.

We show in the following figures comparisons between the different approaches we propose. On Figures 12, we plot the L^2 norm errors between the numerical and analytical solution. We first plot the error for the elliptic part (6.24) for which the behaviour is similar to the curves obtained in section 4. Then, we focus on the total error of the solvers for (6.18). We observe that the curves are nearly the same; this is due to the collocation method which is very precise. The error made in this part is then negligeable compared to the error made in the elliptic part. We can also remark that the error for the classical approach spectral + finite differences is of order 2 (due to the second order finite differences). This approach needs to consider a lot of point in the radial direction (2048) to reach a competitive accuracy, whereas the IGA-FEM approach presents very precise results even considering cubic or quartic order.

Test case 2: In this test, some anisotropy is introduced in the Laplacian term by initializing n_0. Moreover, a profile is imposed to T_e. These two functions are solutions of a differential equation. The radial profiles of the ion temperature $T_e(r)$ as well as the radial density profile $n_0(r)$ are deduced by numerical integration of their gradient profile given by

$$\frac{1}{T_e(r)}\frac{dT_e(r)}{dr} = -\cosh^{-2}(3(r-r_p)), \tag{6.29}$$

with $r_p = 0.5$. These parameters are employed in [8]. Hence, imposing analytical form of the solution,

$$\phi(r,\theta,\varphi) = C\sin\varphi\cos(4\theta)(r-r_{\min})^6(r_{\max}-r)^6, \tag{6.30}$$

with C a constant chosen such that $\max[\phi] = 1$, we deduce an expression of the right hand side of (6.18) which is given to the code to compute back an approximation of the solution. The right hand side F is given by

$$F(r,\theta,\varphi) = -\partial_r^2\phi - \partial_r\phi\left(\frac{1}{r} + \frac{n_0'}{n_0}\right) - \frac{1}{r^2}\partial_\theta^2\phi + \frac{\phi}{T_e},$$

where the different term are

$$\begin{aligned}
\partial_r\phi &= 6C\sin(4\theta)\sin\varphi\left[(r-r_{\min})^5(r_{\max}-r)^6 - (r-r_{\min})^6(r_{\max}-r)^5\right],\\
\partial_r^2\phi &= 30C\sin(4\theta)\sin\varphi(r-r_{\min})^4(r_{\max}-r)^4\\
&\quad\left[(r_{\max}-r)^2 - \frac{12}{5}(r-r_{\min})(r_{\max}-r) + (r-r_{\min})\right],\\
\partial_\theta^2\phi &= -16\phi.
\end{aligned}$$

As in the previous tests, we are interested in the L^2 norm of the error as a function of the size of the mesh. Results are presented in Table 2 in which the L^2 norm of the error is given as a function of the number of points and the degree of the splines. We also give the results for the standard approach where finite difference and FFT are used. In this test also, the IGA-FEM approach has a very good behaviour since with 32 points per direction; to achieve the same precision the finite difference+FFT approach needs 1024 points per direction. Note a kind of saturation of the error: this is due to the computation of the solution of the ODE 6.29. Indeed, T_e and n_0 are numerical solution of an ODE and are not analytical. We verify that when the ODE is solved precisely (by refinement), the correct orders are recovered.

On Figure 13, the solution together with the error between the analytical solution and the numerical one are plotted for the IGA-FEM approach, using degree 2 and 32 points per direction.

Test case 3: non-circular cross section

Following the Nishimura idea [20], we have generated a turbulence test over our non-circular tokamak model. We give the result of such test, with $l = m = 10$ and NURBS degree $p = 4$, in Figure 14.

For such a domain, the use of the standard finite elements method, would need a great number of degrees of freedom, moreover we couldn't be able to represent exactly the tokamak.

Spline degree	Degrees of freedom	L^2 error norm	L^2 error norm "refined"
FD+FFT	262144	$1.5\ 10^{-5}$	
FD+FFT	1048576	$3.7\ 10^{-6}$	
FD+FFT	4194304	$9.6\ 10^{-7}$	
2	256	$4.5\ 10^{-5}$	$4.5\ 10^{-5}$
2	1024	$6.4\ 10^{-6}$	$2.3\ 10^{-6}$
3	256	$9.6\ 10^{-6}$	$7.6\ 10^{-6}$
3	1024	$7.5\ 10^{-6}$	$5.7\ 10^{-7}$
4	256	$5.3\ 10^{-6}$	$1.6\ 10^{-6}$
4	1024	$6.6\ 10^{-6}$	$4.8\ 10^{-7}$

Table 2: *Test 2 QN*: Number of degree of freedom and L^2 norm of the error for some spline degrees and for the method finite-difference+FFT (FD+FFT). In the third column, the ODE (6.29) is solved using 10^3 points whereas in the last column, 10^4 are used.

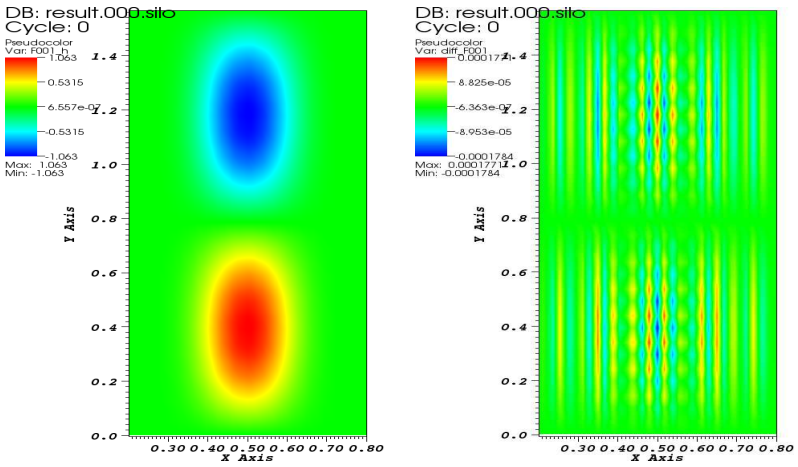

Figure 13: Test 2 QN: numerical solution with NURBS (left) and the difference with the analytical solution (right). 32 points are used per direction, order 3.

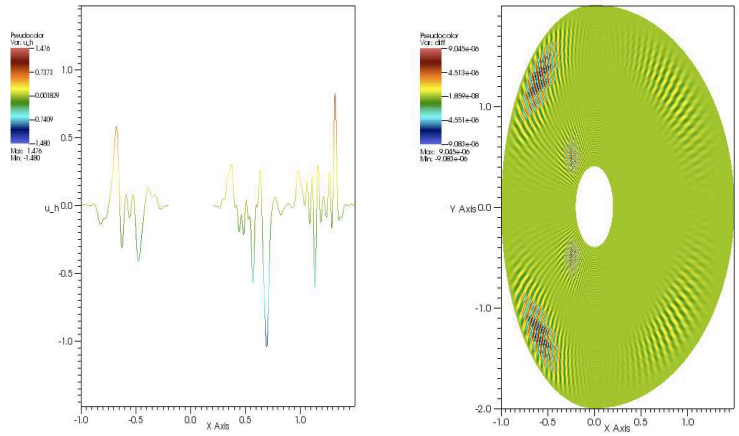

Figure 14: Turbulence test: (left) the numerical solution, with $l = m = 10$ and $p = 4$, (right) the difference $u - u_h$ for a turbulence test in the tokamak model, with $l = m = 2$ and $p = 4$.

7 Conclusion

This work presented a new approach for the resolution of elliptic type operator, with an application to the quasi-neutrality equation. The IGA-FEM approach seems to be very efficient to deal with this kind of problem, but generalization to a large class of operators can also be dealt with following the FEM approach.

A Appendix: Decoupling approach for $J(r,\theta)$

The starting equation writes

$$-\nabla_\perp \cdot (n_0 \nabla_\perp \phi) + [\phi - \langle\phi\rangle] = F, \tag{A.31}$$

with $F = (n_i - n_e)/n_0$ and

$$\langle\phi\rangle(r) = \frac{\int \phi(r,\theta,\varphi) J(r,\theta) d\theta d\varphi}{\int J(r,\theta) d\theta d\varphi}.$$

The appendix is devoted to the extension of the decoupling approach to (r,θ) depending jacobians J. As in the case where $J(r,\theta) = r$, we integrate (A.31) with respect to θ, φ to get a one-dimensional equation on $\bar{\phi}(r) = 1/(4\pi^2) \int \phi d\theta d\varphi$

$$-\nabla_\perp \cdot (n_0 \nabla_\perp \bar{\phi}) + \bar{\phi} - \langle\phi\rangle = \bar{F}. \tag{A.32}$$

Let us remark that this operation is transparent for the operator $\langle\cdot\rangle$. Moreover, we are looking for an equation satisfied by $\Phi = \phi - \bar{\phi}$ from (A.31):

$$\begin{aligned}
-\nabla_\perp \cdot (n_0 \nabla_\perp \Phi) - \nabla_\perp \cdot (n_0 \nabla_\perp \bar{\phi}) + \phi - \bar{\phi} + \bar{\phi} - \langle\phi\rangle &= F \\
-\nabla_\perp \cdot (n_0 \nabla_\perp \Phi) - \nabla_\perp \cdot (n_0 \nabla_\perp \bar{\phi}) + \Phi + \bar{\phi} - \langle\phi\rangle &= F \\
-\nabla_\perp \cdot (n_0 \nabla_\perp \Phi) + \bar{F} - \bar{\phi} + \langle\phi\rangle + \Phi + \bar{\phi} - \langle\phi\rangle &= F \text{ thanks to (A.32)} \\
-\nabla_\perp \cdot (n_0 \nabla_\perp \Phi) + \bar{F} - \bar{\phi} + \langle\phi\rangle + \Phi + \bar{\phi} - \langle\phi\rangle &= F \\
-\nabla_\perp \cdot (n_0 \nabla_\perp \Phi) + \Phi &= F - \bar{F}.
\end{aligned} \tag{A.33}$$

Let us come back to (A.32) in order to derive an equation satisfied by $\langle\phi\rangle$. From (A.32), we have

$$-\nabla_\perp \cdot (n_0 \nabla_\perp (\bar{\phi} - \langle\phi\rangle)) - \nabla_\perp \cdot (n_0 \nabla_\perp \langle\phi\rangle) + \bar{\phi} - \langle\phi\rangle = \bar{F}.$$

By introducing the notation $h(r) = \bar{\phi} - \langle\phi\rangle$, we can derive an equation for $\langle\phi\rangle(r)$

$$-\nabla_\perp \cdot (n_0 \nabla_\perp \langle\phi\rangle) = \bar{F} + \nabla_\perp \cdot (n_0 \nabla_\perp h) - h, \;\; h(r) = \bar{\phi} - \langle\phi\rangle. \tag{A.34}$$

To conclude, we present the decoupling algorithm which enables to solve (A.31)

- (a) solve (A.33) $\rightarrow \Phi = \phi - \bar{\phi}$
- (b) compute $\langle\Phi\rangle$
- (c) compute (and store) tmp $= (\Phi - \langle\Phi\rangle)$
- (d) compute $h(r) = 1/(4\pi^2) \int (\Phi - \langle\Phi\rangle) d\theta d\varphi = \bar{\phi} - \langle\phi\rangle$
- (e) solve (A.34) $\rightarrow \langle\phi\rangle$
- (f) $\phi = \text{tmp} + \langle\phi\rangle = (\Phi - \langle\Phi\rangle) + \langle\phi\rangle = \phi - \bar{\phi} - \langle\phi\rangle + \bar{\phi} + \langle\phi\rangle$ using (c) and (e).

Remark A.1 *One of the key argument comes from the fact that operators $\langle\cdot\rangle$ and $\bar{\cdot}$ are transparent from each other*

$$\langle\bar{\phi}\rangle = \int \bar{\phi} J d\theta d\varphi / (\int J d\theta d\varphi) = \bar{\phi} \int J d\theta d\varphi / (\int J d\theta d\varphi) = \bar{\phi},$$

and

$$\bar{\langle\phi\rangle} = 1/(4\pi^2) \int \langle\phi\rangle d\theta d\varphi = \langle\phi\rangle / (4\pi^2) \int d\theta d\varphi = \langle\phi\rangle.$$

References

[1] J.A Cottrell, T. Hughes, Y. Bazilevs, *Isogeometric Analysis, toward Integration of CAD and FEA*, first ed., John Wiley & Sons, Ltd, (2009).

[2] O. Czarny, G. Huysmans, *Bézier surfaces and finite elements for MHD simulations*, J. of Comput. Phys. **227**, pp. 7423-7445, (2008).

[3] C. de Boor, B. Swartz, *Collocation at Gaussian points*, SIAM J. Numer. Anal. **19**, (1973).

[4] A.M. Dimits, M.A. Beer, G.W. Hammett, C. Kim, S.E. Parker, D.E. Shumaker, R. Sydora, A.J. Redd, J. Weiland, M.T. Kotschenreuther, W.M. Nevins, G. Bateman, C. Bolton, B.I. Cohen, W.D. Dorland, A.H. Kritz, J.E. Kinsey, L.L. Lao, J. Mandrekas., *Comparisons and physics basis of tokamak transport models and turbulence simulations*, Phys. Plasmas **7**, pp. 969-983, (2000).

[5] M.R. Dörfel, B. Jüttler, B. Simeon, *Adaptive isogeometric analysis by local h-refinement with T-splines*, Comput. Methods Appl. Mech. Engrg. **199**, pp. 264-275, (2010).

[6] D.H.E. Dubin, J. A. Krommes, C. Oberman, W. W. Lee, *Nonlinear gyrokinetic equations*, Phys. Fluids **26** (12), 1983.

[7] J.A. Evans, Y. Bazilevs, I. Babuska, T. Hughes, *n-Widths, supinfs, and optimality ratios for the k-version of the isogeometric finite element method*, Comput. Methods Appl. Mech. Engrg. **198**, pp. 17261741, (2009).

[8] V. Grandgirard, M. Brunetti, P. Bertrand, N. Besse, X. Garbet, Ph. Gendrih, G. Manfredi, Y. Sarazin, O. Sauter, E. Sonnendrücker, J. Vaclavik, L. Villard, *A drift-kinetic Semi-Lagrangian 4D code for ion turbulence simulation*, J. Comput. Phys. **60**, pp. 163-194, (1991).

[9] T.S. Hahm, *Nonlinear gyrokinetic equations for tokamak microturbulence*, Phys. Fluids **31**, (9), 1988.

[10] G.W. Hammett, F.W. Perkins, *Fluid models for Landau damping with application to the ion-temperature-gradient instability*, Phys. Rev. Lett. **64**, pp. 3019-3022, (1990).

[11] R. Hatzky, T.M. Tran, A. Koenis, R. Kleiber, S.J. Allfrey, *Energy conservation in a nonlinear gyrokinetic particle-in-cell code for ion-temperature-gradient-driven modes in θ-pinch geometry* Phys. Plasmas bf 9, (2002).

[12] W. Heidrich, R. Bartels, G. Labahn, *Fitting Uncertain Data with NURBS*, in Proc. 3rd Int. Conf. on Curves and Surfaces in Geometric Design (1996) 1-8, Vanderbilt University Press.

[13] Q.X. Huang, S.M. Hua, R.R. Martin, *Fast degree elevation and knot insertion for B-spline curves*, Comput. Aided Geom. Design **22**, pp. 183197, (2005).

[14] T. Hughes,J. A. Cottrell, Y. Bazilevs, *Analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement*, Comput. Methods Appl. Mech. Engrg. **194**, pp. 4135-4195, (2005).

[15] T. Hughes, A. Reali, G. Sangalli, *Efficient quadrature for NURBS-based isogeometric analysis*, Comput. Methods Appl. Mech. Engrg. **199**, pp. 301-313, (2010).

[16] W.W. Lee, *Gyrokinetic approach in particle simulation*, Phys. Fluids **26**, p. 556, (1983).

[17] W.W. Lee, R. A. Kolesnikov, *On high-order corrections to gyrokinetic Vlasov-Poisson equations in the long wavelength limit*, PPPL- 4382 report, (2009).

[18] R. Lynch, J. Rice, and D. Thomas, *Direct solution of partial difference equations by tensor product methods*, Numer Math **6**, pp. 185-199, (1964).

[19] W. Ma, J.P. Kruth, *NURBS Curve and Surface Fitting for Reverse Engineering*, Int. J. Adv. Manuf. Technol. **14**, pp. 918-927, (1998).

[20] Y. Nishimura, Z. Lin, J.L.V. Lewandowski, S. Either, *A finite element Poisson solver for gyrokinetic particle simulations ina global field aligned mesh*, J. Comput. Phys., **214**, pp. 657-671, (2006).

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

[22] H. Prautzsch, B. Piper, *A fast algorithm to raise the degree of B-spline curves*, Computer Aided Geometric Design **4**, pp. 253-266, (1991).

[23] A. Ratnani, E. Sonnendrücker, *Arbitrary High-Order Spline Finite Element Solver for the Time Domain Maxwell equations*, submitted, http://hal.inria.fr/hal-00507758/en.

[24] T.W. Sederberg, D.L. Cardon, J. Zheng, T. Lyche, *T-spline Simplification and Local Refinement*, ACM Trans, Graphics 23 (3) (2004) 276-283.

[25] C.F. Van Loan, *The uniquitous Kronecker product*, J. Comput. Appl. Math. 123 (2000) 85-100.

Contents