Role of energy transfers in the dynamo effect : application to the von Kármán flow

Melvin Creff

université PARIS-SACLAY

Role of energy transfers in the dynamo effect: application to the von Kármán flow.

Rôle des transferts d'énergie dans l'effet dynamo : application à l'écoulement de von Kármán.

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°579 : sciences mécaniques et énergétiques, matériaux et géosciences (SMEMaG).
Spécialité de doctorat: Mécanique des fluides
Graduate School : Sciences de l'ingénierie et des systèmes. Référent : Faculté des sciences d'Orsay

Thèse préparée dans les unités de recherche **LISN** (Université Paris-Saclay, CNRS), sous la direction de **Caroline NORE**, professeur des universités et la co-direction de **Bérengère DUBRULLE**, directeur de recherche CNRS.

Thèse soutenue à Paris-Saclay, le 21 Décembre 2023, par

Melvin CREFF

Composition du jury

Membres du jury avec voix délibérative

Sébastien GALTIER Professeur à l'Université Paris-Saclay, Laboratoire de Physique des Plasmas, École polytechnique	Président
Yannick PONTY Directeur de recherche, CNRS, Laboratoire Lagrange Observatoire de la Côte d'Azur	Rapporteur & Examinateur
Nathanaël SCHAEFFER Chercheur CNRS (HDR), Laboratoire ISTerre	Rapporteur & Examinateur
Francky LUDDENS Maître de conférences à l'Université de Rouen Normandie, Laboratoire de Mathématiques Raphaël Salem	Examinateur

THESE DE DOCTORAT
NNT : 2023UPAST214

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences mécaniques et énergétiques, matériaux et géosciences (SMEMAG)

Titre: Comment produire du champ magnétique dans les fluides, les étoiles et les planètes ?
Mots clés: Magnétohydrodynamique, analyse numérique, mécanique des fluides

Résumé:

Dans cette thèse, l'objectif est de mieux comprendre l'effet dynamo pouvant apparaître dans les fluides électriquement conducteurs en étudiant les transferts d'énergie de la magnétohydrodynamique (MHD). Motivés par l'observation que l'effet dynamo est un phénomène de conversion d'énergie entre champs de vitesse et magnétique, nous avons développé une nouvelle méthode se basant sur l'étude locale (en temps, espace et échelle) des bilans énergétiques décrivant ainsi les différentes dissipations ainsi que les transferts inter-échelles. Notre approche se base sur un nouveau type de filtrage des champs pouvant être utilisé sur n'importe quel maillage de simulation, incluant les maillages non-structurés.

Pour simuler la MHD, nous utilisons le code SFEMaNS qui est une méthode hybride mêlant éléments finis et décomposition spectrale. Tout d'abord, nous présentons quelques améliorations de ce code : une meilleure répartition du maillage sur les processeurs et une étude portant sur l'implémentation d'une méthode de résolution des équations de Navier-Stokes à base de méthodes matricielles plutôt que celle de prédiction-correction actuellement utilisée. Ensuite, nous expliquons les nombreux outils informatiques implémentés pour pouvoir calculer, visualiser et étudier les transferts d'énergie.

Finalement, ces outils sont utilisés pour étudier l'expérience de von Kármán Sodium, dans les régimes de croissance et de saturation, qui possèdent deux types de dynamo en fonction du matériau constituant les turbines (acier ou fer doux). Bien que ces deux dynamos diffèrent peu du point de vue de la théorie de champ moyen (les champs de vitesses étant les mêmes), la localité de notre formalisme permet d'élucider l'origine de la différence entre les deux dynamos : pour les turbines en acier, l'effet dynamo provient d'un transfert venant du fluide dans le coeur du volume, alors que, pour les turbines en fer doux, l'effet dynamo provient exclusivement des turbines. Cette approche nous permet aussi de discuter des signes précurseurs à des phénomènes de dissipations anormales et de dynamo rapide, qui peuvent devenir importants dans la limite de fluide non visqueux.

Title: How can magnetic fields be generated in fluids, stars and planets?
Keywords: Magnetohydrodynamics, numerical analysis, fluid mechanics

Abstract:

In this thesis, the aim is to better understand the dynamo effect that takes place in conducting fluids by studying the various energy transfers in magnetohydrodynamics (MHD). Motivated by the observation that dynamo is a conversion mechanism between magnetic and kinetic energy, we develop a new approach to unravel the dynamo mechanism based on local (in space, scale, and time) energy budgets describing dissipation and scale-by-scale energy transfers. Our approach is based upon a new filtering approach that can be used effectively for any type of meshes,including unstructured ones.

In order to simulate the MHD, we use the SFEMaNS code which is a hybrid method using both a finite element and spectral decomposition. We first present some developments in SFEMaNS: a better repartition of the unstructured mesh on the processors and a study on solving the Navier-Stokes equations using preconditioning matrix methods instead of the current prediction-correction method. Then, we explain how various post-processing tools have been implemented in order to compute, visualize, and study the energy transfers.

Finally, these tools are used to study the von Kármán Sodium setup, showing dynamo action for two types of impellers (steel or soft iron) in the magnetic field growth and saturation phases. Although the two types of dynamo hardly differ from the mean-field theory point of view (the velocity fields are the same in both cases), the locality of our formalism allows us to trace the origin of the differences between these two types of dynamo: for steel impellers, the dynamo is due to the transfer of kinetic energy both in the bulk and in the vicinity of the impellers, whereas for soft iron impellers, the dynamo effect comes from the impellers. We also discuss possible signatures of precursors to anomalous dissipation and fast dynamo, that could become relevant in the inviscid limit.

Remerciements

Cette thèse, ces trois années de travail, furent une belle aventure grâce à la présence, la patience et la bienveillance de nombreuses personnes.

En premier lieu, je souhaite chaleureusement exprimer ma pleine gratitude envers mes deux directrices de thèse, Caroline Nore et Bérengère Dubrulle, pour leurs conseils, leurs aides et leurs encouragements, m'ayant permis de mener à bien les travaux de cette thèse. Je leur suis sincèrement reconnaissant de m'avoir suivi durant ces trois années et d'avoir contribué à ma formation de chercheur.

Je suis très reconnaissant envers Yannick Ponty et Nathanaël Schaeffer pour leur lecture et commentaires pertinents sur mon manuscrit ainsi qu'à Sébastien Galtier et Francky Luddens pour avoir participé au jury de ma thèse et pour les discussions intéressantes qui ont suivi.

Je remercie grandement Jean-Luc Guermond pour l'opportunité qu'il m'a offerte de venir travailler à TAMU et, par la même occasion, le reste de l'équipe du département de mathématiques à TAMU, qui m'a chaleureusement accueilli en terre étrangère.

Mes remerciements reviennent aussi aux nombreuses équipes techniques et administratives des différentes structures (LISN, CEA, IDRIS, SMEMAG, IUT Orsay, Paris-Saclay, TAMU) avec qui j'ai collaboré m'ayant permis de faire cette thèse dans la sérénité.

Je remercie par ailleurs l'équipe enseignant de l'IUT d'Orsay pour ses conseils sur l'enseignement ainsi que les élèves rencontrés durant ces trois années de monitorat.

Au final, je suis heureux d'avoir passé ces trois années, entouré d'un grand nombre de collègues (chercheurs, doctorants, stagiaires, étudiants) qui ont tous apporté à mon vécu par leurs discussions.

Il me reste à remercier mes parents, mes frères et ma sœur pour leur soutien mental sans qui je n'aurai pas fait les études m'ayant permis de faire cette thèse.

Contents

1 - Introduction

1.1 . Context

The dynamo effect is a well-known process that generates a magnetic field from the movement of an electrical conductive matter. In the case of planets, it is a fluid (liquid iron in the outer core of the Earth) put in movement by different mechanisms (rotation, thermal convection, solutal convection, etc) while, in stars, it is ionized gas. In all cases, what is required is an electrical conductive matter, and an energy source maintaining the motion of the matter. The generation of a magnetic field has also been reproduced in the laboratory using liquid sodium set in motion by various mechanical devices [1, 2, 3]. The only experiment to have generated a magnetic field with an unconstrained flow is the von Kármán Sodium (VKS) experiment [4]. Previous experiments [1, 2] have generated a magnetic field using mathematically predetermined flows.

The von Kármán Sodium (VKS) experiment is an impressive device resulting from a collaboration between the ENS Paris, ENS Lyon, CEA Saclay, and CEA Cadarache. The aim of this experiment is to create and maintain a magnetic field by transferring energy from a conductive fluid, thus experimentally reproducing the dynamo effect at play inside planets and stars. The fluid used is liquid sodium, kept at 120°C, and put in turbulent motion thanks to two counter-rotating impellers which are disks fitted with curved blades.

A sustained magnetic field was first measured in 2006 in this experiment. The magnetic field was generated only when the movement was driven by soft-iron impellers [4] (and not by copper or steel impellers)] and with a flow containing limited fluctuations [3]. This flow was created by the two impellers rotating in opposite directions with an unscooping motion for the curved blades. No dynamo was observed with the scooping direction of rotation. The last point proves that this experiment cannot be interpreted as a simple magnet rotating in a conductive fluid.

However, it does indicate that the fluctuations impede the dynamo mechanisms as already seen in numerical simulations for simpler geometry [5] or asymptotics [6]. The resulting field was on average dipolar, aligned on the symmetry axis, with an azimuthal component concentrated near the impellers. However, due to experimental constraints, it was not possible to measure fluid velocity or the non-axisymmetric variation of the magnetic field. For some parameters, magnetic field inversions were also observed, as is the case for the Earth's magnetic field.

The main difficulty in finding the conditions for which the dynamo effect occurs comes from the turbulent movement of the fluids in stars and planets. A turbulent flow corresponds to a disordered and rapidly varying veloc-

ity field containing numerous vortices. This kind of flow appears when the viscous force (associated with the capacity of the flow to dissipate motion) is small compared to the inertial force of the system. This makes the flow span multiple spatial scales from the global one generated by external forces to the smallest vortices dissipating the kinetic energy. As the dynamo effect comes from this flow, it is difficult to pinpoint which events in the flow generate the magnetic energy. This can lead to a magnetic field generated by very rough velocity fields [7] through a mechanism called fast-dynamo [8, 9] (which occurs when the exponential growth of the magnetic field is as fast as the advective time scale of the field). This can occur for very turbulent flows for which the stretching of the magnetic lines grows exponentially through vortices. The turbulent fluctuations may also be detrimental to the dynamo action by desorientation of the magnetic field lines [**?**].

To deal with this difficulty, one popular approach is to build simpler dynamo models using mean-field or stochastic theories [10, 11]. These approaches revealed the importance of two aspects of the fluid's flow: the helicity (through the α-effect) and the differential rotation (through the Ω-effect). For the VKS experiment, various combinations of these effects were considered for the emergence of the magnetic field [12, 13]. However, as these models only analyze the impact of the velocity field on the magnetic field, it is not possible to understand the differences experimentally observed when using different materials for the impellers. This requires other models for the interaction between the Ω-effect and the temporal or spatial variation of electric conductivity [14, 15, 16] or magnetic permeability [17, 18, 19, 20]. It also leads to a few analyzes of increase in the α-effect by eddy collimation near the impellers [21, 22]. Mean-field theory also makes it difficult to understand the initial phase of magnetic field growth, as it necessitates distinguishing between a mean and a fluctuating part of the magnetic and velocity fields. Lastly, as all the small-scale events are parametrized in coefficients for the α and Ω effects, observing events linked to possible fast-dynamo is not possible.

In this thesis, we propose a new method for understanding the dynamo mechanisms responsible for magnetic field generation in turbulent conducting fluids. The main idea of our approach is to analyze the local (in time and space) energy budgets by describing dissipations and energy transfers between scales and fields. In the case of turbulent fluids, this approach was proposed by Duchon and Robert [23]. This method is based on a filtering process that permits the separation of the different scales of the flow. Moreover, compared to other filtering techniques [24], it is possible to detect potential singularities thanks to the introduction of anomalous dissipation corresponding to energy transfers at smaller and smaller scales. This filtering process can be extended to magnetohydrodynamics [25, 26]. However, in these references, variations in magnetic permeability or electrical conductivity are not

taken into account, even though they are essential for the VKS experiment and our simulations. In addition, as these articles are specialized in the study of astrophysics (particularly solar winds), magnetic and kinetic energies are combined. Since we want to study energy transfer between the fluid and the magnetic field, we need to revise this approach by separating the two energies. Therefore, in this thesis, we extend this approach to magnetohydrodynamics with varying magnetic permeability and conductivity, providing a complete description of where all energy transfers (from fluid to magnetic field and from one scale to the other) take place. By combining some numerical simulations carried out previously with this new description of local energy transfers, it is now possible to determine where and when the dynamo effects occur.

1.2 . Magnetohydrodynamics equations

Firstly, the equations of magnetohydrodynamics (MHD) are used to model the interactions of a magnetic field with the velocity field of a conducting fluid. They originate from Maxwell's equations taken in the non-relativistic limit and coupled to the Navier-Stokes equations through the Lorentz force and Ohm's law. In this section, we will present how they are derived and some key quantities in relation to turbulence and the dynamo effect. In particular, the introduction of Reynolds numbers and selected anti-dynamo theorems are important for understanding the choice of configuration and its limits.

1.2.1 . Maxwell's equations

Maxwell's equations are a combination of four equations that predict the behaviour of the electric and magnetic fields in the presence of moving charged particles. In the laboratory frame, they are given by:

$$\partial_t(\mu H) = -\nabla \times E, \tag{1.1}$$

$$\nabla \times H = j + \chi u + \partial_t \epsilon E, \tag{1.2}$$

$$\nabla \cdot \epsilon E = \chi, \tag{1.3}$$

$$\nabla \cdot \mu H = 0, \tag{1.4}$$

where H is the magnetic field, E the electric field, j the electric current density, χ the volumic density of charges, u the fluid velocity, ϵ the electric permittivity and μ the magnetic permeability. Usually, these equations are presented using the constitutive relation between the electric displacement field D and the electric field and the relation between the magnetic flux density (or magnetic induction) b and the magnetic field: $D = \epsilon E$ and $b = \mu H$.

When the fluid is neutral in charges ($\chi = 0$) and goes at a non-relativistic speed we can use the quasi-static approximation. This means that the time

needed for the charges to follow the variation of the electric field is negligible compared to the advective time at which the fluid flows. Hence, these equations can be simplified as:

$$\partial_t(\mu H) = -\nabla \times E, \tag{1.5}$$

$$\nabla \times H = j, \tag{1.6}$$

$$\nabla \cdot \epsilon E = 0, \tag{1.7}$$

$$\nabla \cdot \mu H = 0. \tag{1.8}$$

In order to link the electric current density and the electric field, Ohm's law is used: $j' = \sigma E'$ in the frame of the fluid. In the laboratory frame, this relation reads, using the non-relativistic transformation for an electromagnetic field:

$$j = \sigma(E + u \times b). \tag{1.9}$$

Using this relation to eliminate the electric field gives the equation for the evolution of the magnetic flux density:

$$\partial_t b = \nabla \times (u \times b) - \nabla \times \frac{j}{\sigma}. \tag{1.10}$$

This is the equation for the time evolution of the magnetic flux density. It consists of two terms on the right-hand side: the first is the source of b and the only way to maintain it; the second is the encoding of the Joule effect, which is a loss of energy by the magnetic field through the electrical resistivity of the fluid.

1.2.2 . From the Navier-Stokes equations and the Lorentz force

The Navier-Stokes equations for an incompressible fluid, of kinematic viscosity ν and density ρ, under an external volumic force f is given by:

$$\partial_t u + \partial_i(u_i u) - \nu\Delta u + \frac{\nabla p}{\rho} = \frac{f}{\rho}, \boldsymbol{\nabla} \cdot u = 0 \tag{1.11}$$

In MHD, the external force is the Lorentz force which, for a particle of charge q at velocity v under an electromagnetic field (E,H), is $q(E + v \times b)$. Hence, here the total volumic force is, respectively on the positive and negative charges:

$$f_+ = \chi_+(E + u_+ \times b), \tag{1.12}$$

$$f_- = \chi_-(E + u_- \times b), \tag{1.13}$$

where $\chi_\pm$ are the charge densities and $u_\pm$ their velocity.

Using that, for a neutral fluid $\chi_+ = -\chi_-$ and that the electric current density is $j = \chi_+ u_+ + \chi_- u_-$, we get:

$$f = f_+ + f_- = j \times b. \tag{1.14}$$

Combining these equations with the equations obtained for the magnetic field gives the magnetohydrodynamics equations for a neutral conductive incompressible fluid in the quasi-static approximation:

$$\partial_t u + \partial_i(u_i u) - \nu\Delta u + \frac{\nabla p}{\rho} = \frac{j \times b}{\rho}, \tag{1.15}$$

$$\partial_t b = \nabla \times (u \times b) - \nabla \times \frac{j}{\sigma}, \tag{1.16}$$

$$\nabla \cdot u = 0, \tag{1.17}$$

$$\nabla \cdot b = 0. \tag{1.18}$$

1.2.3 . Dynamo effect and Reynolds numbers

The dynamo effect is the amplification and maintenance of a magnetic field by the movement of an electrically neutral material. In our case, this material is a fluid, and the dynamo effect in a conductive fluid results from an instability. Indeed, $b = 0$ is always a solution for equation (1.16). Hence, for a non-trivial magnetic field to appear and be maintained, the non-linear term ($\nabla \times (u \times b)$) needs to overcome the Joule dissipation ($-\nabla \times \frac{j}{\sigma}$), meaning that, for a system of characteristic size L and fluid velocity v:

$$\left[\frac{\nabla \times (u \times \mu H)}{\nabla \times ((\nabla \times H)/\sigma)}\right] \sim \sigma\mu v L = R_\mathrm{m} > 1. \tag{1.19}$$

R_m is called the magnetic Reynolds number and gives a minimum bound to when a magnetic field can be generated. More precisely, for a fixed velocity field, equation (1.16) is linear in b, and thus R_m is the control parameter of this instability. In this framework with a prescribed-velocity dynamo (called *kinematic dynamo*), many analytical fields were derived (Ponomarenko dynamo [27], Roberts dynamo [28]) and experimentally reproduced [1, 2].

However, the most interesting dynamo effect is for the non-linear coupling between the Navier-Stokes and the induction equations. For this coupled system, there is no theorem that states in which circumstances a flow will produce a magnetic fluid and, worst, there are many anti-dynamo theorems (conditions in which the dynamo cannot appear whatever the R_m value) [10]. For example, a purely axisymmetric magnetic field can never be produced whatever the velocity field [29, 30].

The second control parameter of interest is the (kinetic) Reynolds number for a turbulent fluid. This characterizes the degree of turbulence of a flow by comparing the nonlinear inertial term ($\partial_i(u_i u)$) with the viscous dissipation ($\nu\Delta u$) in Eq. (1.15):

$$\left[\frac{\partial_i(u_i u)}{\nu\Delta u}\right] \sim \frac{vL}{\nu} = R_\mathrm{e}. \tag{1.20}$$

The higher the kinetic Reynolds number, the smaller the structures can be in relation to the size of the setup.

Both numbers are linked by the magnetic Prandtl number which only depends on the physical properties of the fluid:

$$P_m = \frac{R_\mathrm{m}}{R_\mathrm{e}} = \mu\sigma\nu. \tag{1.21}$$

Since the fluid used in the experiment is prescribed (i.e. liquid sodium), the only way to increase the chances of a magnetic field appearing is to increase the fluid's velocity. For liquid sodium at 120°C: $\nu = 7\,10^{-7}\,\mathrm{m^2\,s^{-1}}$, $\sigma = 9.6\,10^{6}\,\mathrm{S\,m^{-1}}$ and $\mu = \mu_0 = 4\pi\,10^{-7}\,\mathrm{H\,m^{-1}}$. This yields $P_m = 0.8\,10^{-5}$ and imposes that $R_\mathrm{e} \gtrsim 10^5$ at least for a magnetic field to appear. In the experiment, experimentalists needed to go as high as $R_\mathrm{e} \gtrsim 6\,10^6$ in order to generate a magnetic field. This high Reynolds number implies that something may happen on a very small scale that may contribute to the emergence of the magnetic field.

1.3 . Numerical simulations

To study the magnetohydrodynamics equations and the dynamo mechanism, various numerical codes exist. Most of the time, each geometry requires its own specific numerical framework. In addition, depending on the objectives, a kinematic dynamo equation or complete MHD equations are calculated. When the focus is on the instability threshold, a given velocity field (from experimental results, analytical models, or Navier-Stokes solvers) is used to integrate the magnetic field equation (1.16) with a variable magnetic Reynolds number. The critical threshold R_m^c is defined when the solution $b = 0$ becomes unstable. When linear and saturation phases of magnetic field generation are the objective, the complete MHD equations (1.15-1.16) must be computed.

For the Ponomarenko dynamo studied in [31] a 1D eigenvalue code for kinematic dynamo is developed. In [32] a parallelized pseudospectral code for the full MHD equations in a periodic box is used to better understand the Roberts dynamo.

Dealing with more complex flows, the authors in [33] study the dynamo driven by rotation and shear in the accretion disk of galaxies, while [34] investigate the dynamo action in rotating convective layers. The two codes used in these references employ periodic boundary conditions in at least two directions. For infinite cylinders (periodic boundary conditions in θ and z), there exist a few other codes [35, 36, 37], the last one being used to study Taylor-Couette dynamo. There are also a few codes dedicated to spheres: pseudo-spectral codes with the usual poloidal-toroidal decomposition of u and B [38, 39, 40, 41], or finite volume algorithms [42], or finite elements [43] and even commercial codes like COMSOL used in [44]. However, to simulate finite domains, only a few exist: for a spheroid [45], for a torus [46], and for cylinders [47, 48].

In order to better understand the VKS experimental results and the dynamo mechanism in place, several numerical simulations were carried out. Simple periodic codes of the complete MHD equations with Cartesian geometry were used to estimate the critical magnetic Reynolds and led to other results such as Lorentz force hysteresis cycles and the non-linear dynamo condition [49, 50, 51, 52]. A kinematic code in infinite cylinders was used to optimize the geometry of the impellers for the VKS experiment, and initial results were obtained on the structure of the magnetic field generated [53]. With a finite cylinder, the detrimental role of the lid layers behind the impeller disks was highlighted [54, 55]. However, the magnetic field structure obtained was completely different from that observed in the experiment later on: the numerical magnetic field was dominated by a component perpendicular to the cylinder axis and another parallel to this axis (i.e. an azimuthal $m = 1$ mode compatible with Cowling's anti-dynamo theorem). Then, by introducing the α and Ω effects from the mean-field theory into the kinematic dynamo simulation, the averaged structure of the magnetic fields observed in the experiment was reproduced [56, 57, 17]. However, it either necessitated very high values of α or a homogenous one which are both unrealistic. Finally, direct numerical simulations (DNS) and large eddy simulations (LES) that simulate the complete MHD equations and impeller counter-rotation have been developed [58].

In this thesis, we will use new DNS computations performed with some parameters set in [58]. These simulations use the SFEMaNS code [59] which we have chosen for several reasons: it uses cylindrical coordinates to easily represent the cylinder, the impellers are well represented using a penalty method (see [58] p.5), data from various runs for the VKS experiment [58] are available and some tools to study turbulence have already been implemented [60]. To simulate a complete turbine rotation, it takes around ten hours of simulation on 2040 processors of the Jean-Zay supercomputer supplied by IDRIS. The geometry (see Fig. 1.1) is an approximated version of the experimental setup. The inner volume for $r \in [0, 1]$ corresponds to the liquid sodium while the outer volume for $r \in [1, 1.6]$ is composed of two conductive solid layers: one with the same conductivity as the liquid for $r \in [1, 1.4]$ (modeling the sodium layer) and one with 4.5 times the conductivity $r \in [1.4, 1.6]$ (modeling the copper wall). The turbines have different magnetic permeability depending on the material used.

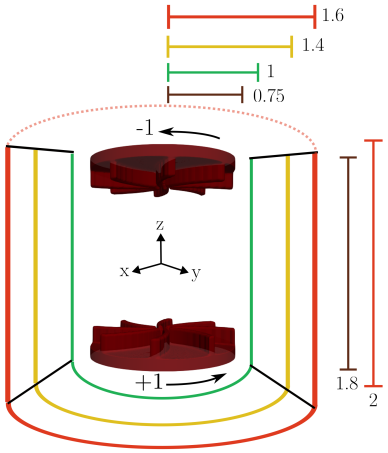


Figure 1.1: Representation of the different simulated volumes: fluid in movement (green), fluid kept at rest (yellow), other conductive solid walls (red), and turbines in movement (dark red).

In this setup, it is possible to numerically observe the generation of the magnetic field in very good agreement with experimental results [58] using DNS for small Reynolds numbers R_e or LES [61] for large R_e (the maximum R_e is 10^5, corresponding to a magnetic Prandtl number $P_m^c \simeq 7\,10^{-4}$ at threshold). In this thesis, we will use only the results obtained with $R_e = 1500$ using DNS to study energy transfer down to the smallest scale without modeling. This Reynolds number is low compared to the real one, however, it still gives a good idea of the main ingredients for which the dynamo effect occurs. In particular, it has been shown that using impellers with high magnetic permeability does lower the critical magnetic Reynolds number at which a magnetic field is produced. As observed in the experiment, the threshold at which a magnetic field was generated depends on the material used for the impellers. However, the origin of the dynamo effect is still elusive.

1.4 . Outline

In order to analyze these numerical simulations and better understand the dynamo effect in the VKS experiment, in this thesis we will implement the local energy transfer approach derived from the MHD equations in the SFEMaNS code.

First, in chapter 2, we present the program used for the simulations as well as its framework. We also present the improvements that have been made to it and a study of a potential new way of simulating our equations.

Then, in chapter 3, we describe various tools developed in order to efficiently compute, analyze, and visualize the local energy transfers. These tools make it possible to have an efficient workflow.

Finally, in chapter 4, we derive the equations for the energy transfers and present the numerical setup. This leads to the analysis of all the different transfers for two illustrative cases using steel and soft iron impellers.

2 - SFEMaNS improvements

In order to simulate the magnetohydrodynamics equations and calculate the various energy transfers, we use the SFEMaNS code. This chapter is devoted to a brief description of this code and the various improvements that have been made to it. We will first describe the numerical framework, and then present the improvements made to mesh distribution on the processors. We will next examine the possibility of modifying the simulation method used. All the work in this chapter was carried out in collaboration with J.-L. Guermond during my stay at Texas A&M University from the 5th of January to the 3rd of February 2022 and from the 15th of July to the 3rd of December 2022.

The SFEMaNS program [59] is a code written in Fortran-90 dedicated to axisymmetric systems. It uses the cylindrical coordinates and decomposes the fields on a finite element basis in the plane (r, z) and on Fourier modes along θ. This is done in order to highly parallelize the computation using the modules MPI for the Fourier modes and PETSC for a domain parallelization on the finite elements. In addition, for the sake of efficiency, Fast Fourier Transforms [62] are used to perform calculations involving any product of the fields.

For our setup, the simulated volume is presented in Fig. 1.1 where the turbines are given motion using a pseudo-penalty method (see [58] p.5). Most of the functions needed to do the basic calculations and parallelization were already implemented. Hence, the main focus is to understand how the different decompositions work in order to implement the new terms to calculate.

2.1 . Numerical approximations used in the SFEMaNS Code

In order to implement the energy transfers in SFEMaNS we need to transform the equations and other operations into the SFEMaNS framework. We will thus present the various numerical approximations done in SFEMaNS and how to use them. The goal is to be able to do various operations (integrals, derivatives, and solving equations) that are needed to calculate the energy transfers.

2.1.1 . Finite elements and Fourier decomposition

The objective of this section is to explain how to solve an equation using the finite element theory in the SFEMaNS code. The study will focus on the temperature field equation (which is already implemented) in order to present how different types of derivatives and other operations are approximated.

The finite element method approximates the space of functions by a subset called test functions. These test functions are defined on a meshgrid (see Fig. 2.1) of triangles in 2D such that, when the characteristic size of a mesh cell goes to zero, the test functions can generate all space of functions. For example, for what is called $\mathbb{P}_1$ approximation (see the left panel of Fig. 2.3), the space function is approximated by the space of continuous piece-wise linear functions and the test functions are thus equal to 1 at one summit (or node) of the triangle and 0 at the two others. There exist higher polynomial approximations using as test functions Lagrange polynomials of higher order, meaning that there are more test functions per mesh cells.

In SFEMaNS, the mesh used is a combination of other meshes: the meshgrid used to define the flow field u is, most of the time, a subdomain of the one that defines the magnetic field. This is because the magnetic field also exists outside the area where the fluid is moving. It is thus possible to create more complex meshgrids by sticking together many smaller grids. The meshgrids for magnetic and velocity fields use $\mathbb{P}_2$ polynomials while (see the central panel of Fig.2.3) the meshgrid for pressure uses $\mathbb{P}_1$ elements. This is because only some pairs of finite elements for pressure and velocity fields ensure convergence to proper solutions of the Navier-Stokes equations. $\mathbb{P}_3$ finite elements (see the right panel of Fig.2.3) are also implemented in the code but have not yet been used in simulations.

2.1.2 . Weak formulation of an equation

The time evolution of the temperature field T in a fluid is given by:

$$\partial_t T + \nabla \cdot (uT) - K\Delta T = f, \tag{2.1}$$

where u is the velocity field, f a thermal source and K the thermal diffusivity of the fluid.

The first step is to discretize the time for time-dependent functions. Using the notation $f^n = f(n\tau)$, the discretization uses the BDF2 method [63]:

$$\partial_t T \longrightarrow \frac{1}{2\tau}(3T^{n+1} - 4T^n + T^{n-1}), \tag{2.2}$$

$$\nabla \cdot (uT) \longrightarrow \nabla \cdot ((2u^n - u^{n-1})(2T^n - T^{n-1})), \tag{2.3}$$

$$\Delta T \longrightarrow \Delta T^{n+1}. \tag{2.4}$$

The next step is to multiply the equation by a test function ϕ and to integrate by parts to get rid of the highest derivatives (with Ω the domain where

T is defined):

$$\int_\Omega \left(\frac{3}{2\tau}\phi + K\vec{\nabla}\phi\cdot\vec{\nabla}\right)T^{n+1} = \tag{2.5}$$
$$\int_\Omega \left[\frac{4T^n - T^{n-1}}{2\tau} + f^{n+1} - \nabla\cdot\left((2u^n - u^{n-1})(2T^n - T^{n-1})\right)\right]\phi.$$

This is called the weak formulation of an equation (here, also discretized in time). This is done in order to project the equation on the subspace generated by the test functions. Moreover, getting rid of the highest derivative makes the calculation more precise.

For clarity purposes, let us rename the right-hand side as follows:
$\int_\Omega \left(\frac{4T^n - T^{n-1}}{2\tau} + f^{n+1} - \nabla\cdot((2u^n - u^{n-1})(2T^n - T^{n-1}))\right)\phi = \int_\Omega F\phi\,.$

2.1.3 . Test functions and Fourier transform

Now that the weak formulation of the equation is obtained, the aim is to transform this equation into a linear problem using both Fourier and finite element decompositions of T.

First, the Fourier decomposition is done as such:

$$T(r,\theta,z,t) = T_h^0(r,z,t) + \sum_{m=1}^{M} T_h^{m,\mathsf{c}}(r,z,t)\cos(m\theta) + T_h^{m,\mathsf{s}}(r,z,t)\sin(m\theta), \tag{2.6}$$

with the coefficients $T_h^{m,\mathsf{c/s}}$ living in the space generated by the test functions on the 2D meshgrid containing n_p total points. The test functions used are also in Fourier space and thus defined for every point $j \in [\![1, n_p]\!]$ as:

$$\phi_j^{m,\mathsf{c}}(r,\theta,z) = \psi_j(r,z)\cos(m\theta) \quad \text{with} \quad 0 \leqslant m \leqslant M, \tag{2.7}$$
$$\phi_j^{m,\mathsf{s}}(r,\theta,z) = \psi_j(r,z)\sin(m\theta) \quad \text{with} \quad 1 \leqslant m \leqslant M, \tag{2.8}$$

where $\psi_i(r,z)$ are the test functions defined on the triangles of the 2D meshgrid.

T can then be written as:

$$T(r,\theta,z,t) = \sum_{j=1}^{n_p} T_j^0(t)\psi_j(r,z) + \tag{2.9}$$
$$\sum_{m=1}^{M}\sum_{j=1}^{n_p} T_j^{m,\mathsf{c}}(t)\psi_j(r,z)\cos(m\theta) + T_j^{m,\mathsf{s}}(t)\psi_j(r,z)\sin(m\theta).$$

Hence, the left-hand side of equation (2.5) reads for a cosinus test function $\phi_i^{m',c}$ (after integration over θ), $m' \in [\![0, M]\!]$ and $i \in [\![1, n_p]\!]$:

$$\sum_{j=1}^{n_p} T_j^{n+1,0}\int_{r,z} r\frac{3}{2\tau}\psi_j\psi_i 2\pi\delta_{0,m'} + \tag{2.10}$$
$$\sum_{j=1}^{n_p}\sum_{m=1}^{M}\int_{r,z} r\left(T_j^{n+1,m,\mathsf{c}}\delta_{m,m'}\pi\left[\frac{3}{2\tau}\psi_j\psi_i + K(\partial_r\psi_j\partial_r\psi_i + \partial_z\psi_j\partial_z\psi_i + \frac{m^2}{r^2}\psi_j\psi_i)\right]\right),$$

and for the right-hand side:

$$\sum_{j=1}^{n_p} \int_{r,z} rF_j^0 \psi_j \psi_i 2\pi \delta_{0,m'} + \sum_{j=1}^{n_p} \sum_{m=1}^{M} \int_{r,z} rF_j^{m,\mathsf{c}} \delta_{m,m'} \pi \psi_j \psi_i. \tag{2.11}$$

Here $\int_{r,z} f(r,z)$ means $\int f(r,z)\,\mathrm{d}r\mathrm{d}z$.
Hence solving (2.5) for T^{n+1} becomes solving for each Fourier mode m and each test function i (and for the cosine part):

$$A_{ij}^{m,\mathsf{c}} T_j^{n+1,m,\mathsf{c}} = B_i^{m,\mathsf{c}}, \tag{2.12}$$

with:

$$A_{ij}^{m,\mathsf{c}} = \int_{r,z} r \left[\frac{3}{2\tau} \psi_j \psi_i + K(\partial_r \psi_j \partial_r \psi_i + \partial_z \psi_j \partial_z \psi_i + \frac{m^2}{r^2} \psi_j \psi_i) \right], \tag{2.13}$$

$$B_i^{m,\mathsf{c}} = \sum_{j=1}^{n_p} \int_{r,z} rF_j^{m,\mathsf{c}} \psi_j \psi_i. \tag{2.14}$$

It is the same linear system for the sinus part in the case of the temperature field. Now the Fourier modes m are uncoupled and thus can be solved simultaneously in parallel. Moreover, as this is a linear system requiring only a matrix inversion, powerful tools of linear algebra can be used, such as preconditioning and LU factorization. However, in order to compute the coefficients of A and B, some integrals need to be calculated.

2.1.4 . Evaluation of integrals

A 2D mesgrid $\Omega_{r,z}$ is composed of n_m cells such that $\Omega_{r,z} = \bigcup_{n=1}^{n_m} K_n$ with a total number of points n_p. For a function G defined in the (r, z) plane for the $\{\psi_i\}_{i \in [\![1,n_p]\!]}$ test functions, its integral is first cut over all n_m mesh cells :

$$\int_{\Omega_{r,z}} G(r,z)\,\mathrm{d}r\mathrm{d}z = \sum_{n=1}^{n_m} \int_{K_n} G(r,z)\,\mathrm{d}r\mathrm{d}z. \tag{2.15}$$

Then each cell is transformed into a reference cell $\hat{K}$ with J_n the associated Jacobian and $\hat{\psi}_i$ the reference local test function:

$$\int_{\Omega_{r,z}} G(r,z)\,\mathrm{d}r\mathrm{d}z = \sum_{n=1}^{n_m} \int_{\hat{K}} G(r,z) J_n(r,z)\,\mathrm{d}r\mathrm{d}z. \tag{2.16}$$

Thanks to this transformation, all different test functions ψ become generic functions. Hence, it is not needed to keep in memory all test functions but only the ones on the reference cell $\hat{K}$ and the Jacobian associated with each cell.

Then, using Gauss quadrature over n_G Gauss points (meaning approximating the integral by a sum over well-chosen and weighted points), we obtain:

$$\int_{\Omega_{r,z}} G(r,z)\,\mathrm{d}r\mathrm{d}z = \sum_{n=1}^{n_m}\sum_{l=1}^{n_G} G(l)J_n(l). \tag{2.17}$$

Using the decomposition of G over the generic test functions then gives, with $\{n_{\mathsf{glob}}\}_n \subset [\![1, n_p]\!]$ the global indices of the test functions at the nodes of the cell n:

$$\int_{\Omega_{r,z}} G(r,z)\,\mathrm{d}r\mathrm{d}z = \sum_{n=1}^{n_m}\sum_{l=1}^{n_G} J_n(l) \sum_{i\in\{n_{\mathsf{glob}}\}_n} G_i\hat{\psi}_i(l). \tag{2.18}$$

For example, in order to evaluate $A_{ij}^{m,\mathsf{c}}$, we use the sum:

$$A_{ij}^{m,\mathsf{c}} = \sum_{n=1}^{n_m}\sum_{l=1}^{n_G} J_n(l)r(l) \tag{2.19}$$
$$\left[\frac{3}{2\tau}\hat{\psi}_j(l)\hat{\psi}_i(l) + K\left(\partial_r\hat{\psi}_j(l)\partial_r\hat{\psi}_i(l) + \partial_z\hat{\psi}_j(l)\partial_z\hat{\psi}_i(l) + \frac{m^2}{r(l)^2}\hat{\psi}_j(l)\hat{\psi}_i(l)\right)\right],$$

where the radius at the Gauss point l is $r(l) = \sum_{k\in\{n_W\}_n} r_k\hat{\psi}_k(l)$ with r_k the radius at the node k. The functions $\{\hat{\psi}_i, \partial_r\hat{\psi}_i, \partial_z\hat{\psi}_i\}$ are known as well as the Jacobian $J_n(l)$ when given the meshgrid.

2.1.5 . Derivatives

The last types of operation that need to be coded in the SFEMaNS code are derivatives. For efficiency and accuracy purposes, the derivatives are calculated on the Gauss points. In addition, the Fourier mode decomposition of the fields into sine and cosine parts must be taken into account. For example, one operation needed is to calculate $j = \nabla \times H$, which decomposition in components (r, θ, z) gives:

$$j_r = \frac{1}{r}\partial_\theta H_z - \partial_z H_\theta, \tag{2.20}$$
$$j_\theta = \partial_z H_r - \partial_r H_z, \tag{2.21}$$
$$j_z = \frac{1}{r}\partial_r(rH_\theta) - \frac{1}{r}\partial_\theta H_r. \tag{2.22}$$

Using the Fourier and finite element decompositions, this gives for cosine

mode m at the Gauss point l:

$$j_r^{m,\mathsf{c}}(l) = \frac{m}{r(l)} \sum_i H_{z,i}^{m,\mathsf{s}} \psi_i(l) - \sum_i H_{\theta,i}^{m,\mathsf{c}} \partial_z \psi_i(l), \tag{2.23}$$

$$j_\theta^{m,\mathsf{c}}(l) = \sum_i H_{r,i}^{m,\mathsf{c}} \partial_z \psi_i(l) - \sum_i H_{z,i}^{m,\mathsf{c}} \partial_r \psi_i(l), \tag{2.24}$$

$$j_z^{m,\mathsf{c}}(l) = \frac{1}{r(l)} \sum_i H_{\theta,i}^{m,\mathsf{c}} \psi_i(l) + \sum_i H_{\theta,i}^{m,\mathsf{c}} \partial_r \psi_i(l) - \frac{m}{r(l)} \sum_i H_{r,i}^{m,\mathsf{s}} \psi_i(l), \tag{2.25}$$

$$j_r^{m,\mathsf{s}}(l) = -\frac{m}{r(l)} \sum_i H_{z,i}^{m,\mathsf{c}} \psi_i(l) - \sum_i H_{\theta,i}^{m,\mathsf{s}} \partial_z \psi_i(l), \tag{2.26}$$

$$j_\theta^{m,\mathsf{s}}(l) = \sum_i H_{r,i}^{m,\mathsf{s}} \partial_z \psi_i(l) - \sum_i H_{z,i}^{m,\mathsf{s}} \partial_r \psi_i(l), \tag{2.27}$$

$$j_z^{m,\mathsf{s}}(l) = \frac{1}{r(l)} \sum_i H_{\theta,i}^{m,\mathsf{s}} \psi_i(l) + \sum_i H_{\theta,i}^{m,\mathsf{s}} \partial_r \psi_i(l) + \frac{m}{r(l)} \sum_i H_{r,i}^{m,\mathsf{c}} \psi_i(l), \tag{2.28}$$

where $\{\psi_i, \partial_r \psi_i, \partial_z \psi_i\}$ are already implemented using the previous Jacobian to go to the reference cell and using the reference test functions $\hat{\psi}_i$.

That concludes a quick overview of the SFEMaNS framework and its various operations that need to be hard-coded when developing new terms in this code. Indeed, other simpler operations such as the scalar product and the cross product are already fully implemented.

2.2 . New mesh distribution on processors

This section is dedicated to improving the distribution of the meshgrid across processors. We need to do this in order to simulate fine meshes without being limited by the per-processor memory. This makes it possible to simulate higher Reynolds numbers. The problem was that each processor stored a copy of the entire grid in its memory. As each processor has only 4 to 8 GB of memory (depending on the computing cluster), it was impossible to obtain 2D meshgrids of more than 500,000 points, regardless of the number of processors used. This severely limited the highest Reynolds number achievable in 2D simulations and was very inefficient in terms of memory. It is now possible to simulate finer meshes by increasing the number of processors. This is very important for simulating magnetohydrodynamics on fine meshes, as several meshes are used (one for the velocity field, one for the pressure field, and one for the magnetic field).

To solve this problem, we have implemented a new way for each processor to save the meshgrid so that they only save in memory the cells that are allocated to them. This new implementation made it possible to have around 300,000 meshgrid points per processor (with 4Go of memory each) for 2D Navier-Stokes problems. Using multiple processors we were able to start 2D Navier-Stokes problems with hundreds of millions of points.

Two main difficulties were encountered. Firstly, the new implementation had to respect the already established convention and structure of the mesh-grid. Secondly, due to parallelization, some processors needed information about bordering cells that are allocated to other processors.

As there are three levels of indexing (locally at the cell level, at the processor level, and at the global level), all the local numbering and indexation will refer to the ones at the processor level. I will also use directly the names of the Fortran tables and functions given in the code. Also, the notations used in this section are not correlated to the ones used before.

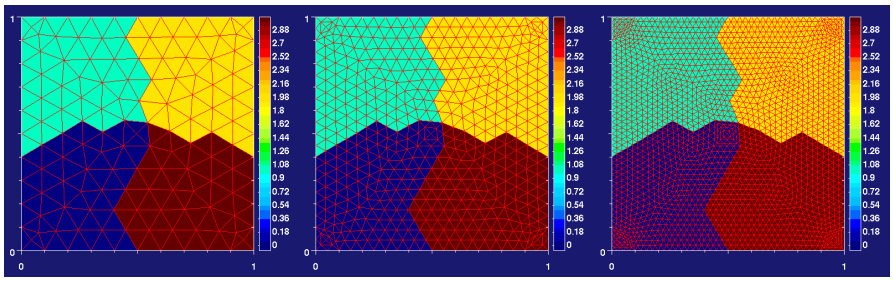


Figure 2.1: Refinement of a mesh. No refinement (left panel), one refinement (middle panel), two refinements (right panel).

2.2.1 . Cell level convention

The convention that needs to be respected for the calculation to proceed accurately (or even at all) is schematized in Fig. 2.2 and 2.3. The vertex of the triangle are number for $i \in [\![1, 3]\!]$ will the other nodes are in $[\![4, 10]\!]$.

The first convention (Fig.2.2) is that, for a cell, its vertices, edges and neighbours are cell-numbered such as, for a vertex with cell-numbering $i \in [\![1, 3]\!]$:

- the edge at the opposite of i is cell-numbered i too,
- the cell neighbouring the opposite edge is also cell-numbered i.

The easiest way to memorize it is that all objects with the same cell number are 'aligned'.

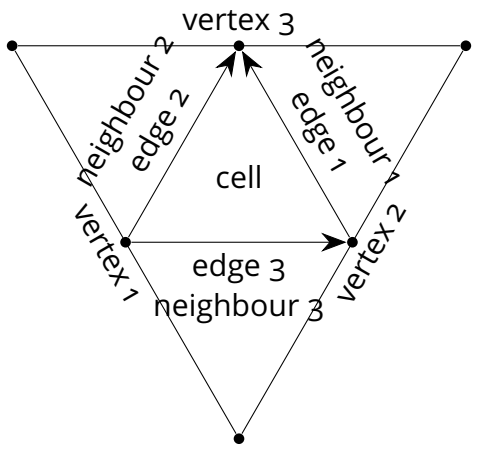


Figure 2.2: Nodes, edges and neighbours convention for a cell.

The second convention is that, for a cell, its finite element points are cell-numbered according to Fig.2.3. This means that, for all $\mathbb{P}_n$ finite elements, the points cell-numbered $i \in [\![1, 3]\!]$ correspond to the respective vertices and the edges are oriented from the lowest cell index of the vertex to the highest one. The other points follow the convention in Fig.2.3. As an example, in $\mathbb{P}_3$ finite element points, the points 4 and 5 are on the cell-numbered edge 1 opposite to the cell-numbered vertex 1 such that 4 is closer to the cell-numbered vertex 2.

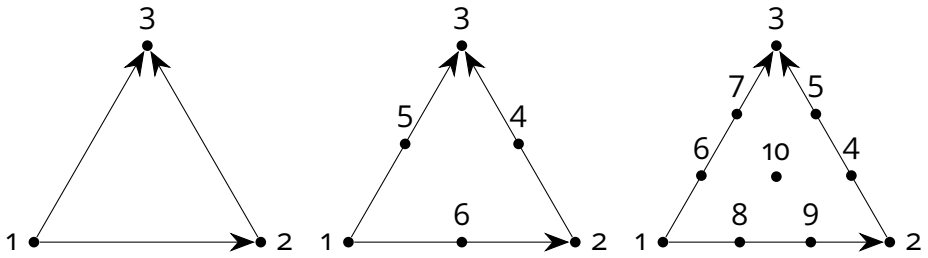


Figure 2.3: Numbering of points and edge orientations in one cell for $\mathbb{P}_1$ (left panel), $\mathbb{P}_2$ (middle panel) and $\mathbb{P}_3$ (right panel) finite elements.

These $\mathbb{P}_3$ finite elements are the reason why we want the edges to be oriented according to a specific convention. For neighbouring cells to have the points on the edges in the same order, the orientation of the edges must be the same from one neighbor to the next. This means that, for an edge between two cells m_1 and m_2 with cell-numbering of the edge's vertices (v_{1,m_1}, v_{2,m_1}) in the cell m_1 and (v_{1,m_2}, v_{2,m_2}) in m_2: if $v_{1,m_1} < v_{2,m_1}$ then $v_{1,m_2} < v_{2,m_2}$.

2.2.2 . Reading the global mesh

The first step is to read an already built $\mathbb{P}_1$ mesh. This is done by using the function load_mesh_free_format_global_p1.

This mesh is called mesh_glob. Its M cells are numbered from $[\![1, M]\!]$, its N points are numbered from $[\![1, N]\!]$ and its M_e edges are numbered from $[\![1, M_e]\!]$. Mesh structure information is stored in a number of tables and integers.

For the mesh:

- mesh_glob%me = M
- mesh_glob%np = mesh_glob%dom_np = N
- mesh_glob%medge = M_e

For each cell $m \in [\![1, M]\!]$ of that mesh:

- mesh_glob%jj(:,m) gives the points $\in [\![1, N]\!]$ of the cell m (there are 3 points per cell because we use $\mathbb{P}_1$ finite elements),
- mesh_glob%jce(:,m) gives the edges $\in [\![1, M_e]\!]$ of the cell m,
- mesh_glob%neigh(:,m) gives the neighbours $\in [\![1, M]\!]$ of the cell m (or 0 if the neighbour is on the external boundaries).

These arrays are build so that, for $i \in [\![1, 3]\!]$, mesh_glob%jj(i,m) is the point opposite to the edge mesh_glob%jce(i,m) and the cell m has mesh_glob%neigh(i,m) for neighbours along the edge mesh_glob%jce(i,m) (Fig.2.2). These arrays are also built so that each edge is oriented in the same way. This is done so that the conventions discussed in section 2.2.1 are respected. This means that, for Fig. 2.6 (left), v_1 =mesh_glob%jj(1,m), v_2 =mesh_glob%jj(1,m) and v_3 =mesh_glob%jj(3,m). Similarly, e_1 =mesh_glob%jce(1,m), e_2 =mesh_glob%jce(1,m) and e_3 =mesh_glob%jce(3,m). It should be noted that there is no reason for the value of the indices v_i and e_i (for $i \in [\![1, 3]\!]$) to be in any particular order. This property will be true for any other mesh that will be discussed in all the following sections.

For a point $v \in [\![1, N]\!]$:

- mesh_glob%rr(1,v) gives its r coordinate,
- mesh_glob%rr(2,v) gives its z coordinate.

In addition to this mesh structure, the external boundaries of the mesh must also be described in order to correctly take into account the different boundary conditions. They are thus separated in $N_{interface}$ interfaces making it possible to specify various boundary conditions (*e.g.* Dirichlet or Neumann or Robin conditions) on different borders of the mesh. The total number of edges at these interfaces is M_{es} and is saved in mesh_glob%mes. For each edge $m_s \in [\![1, M_{es}]\!]$:

- mesh%sides(m_s) gives the number $\in [\![1, N_{interface}]\!]$ of the interface,

- mesh%neighs(m_s) gives the associated cell $\in [\![1, M]\!]$,

- mesh%jjs(m_s) gives the points $\in [\![1, N]\!]$ on the edge m_s.

2.2.3 . Distribution of the $\mathbb{P}_1$ global mesh on processors

Now mesh_glob will be distributed on N_p processors, each processor having then its own mesh_p1_loc. This is done using the function reorder_mesh which does two things: first, it reorders the cells, edges, and vertices in a specific order; then, it distributes to each processor its part of the global mesh by calling the function create_local_mesh_with_extra_layer.

Attribution of cells and reordering

Using METIS_PartGraphRecursive, each cell $m \in [\![1, M]\!]$ is attributed to a processor $p_c \in [\![1, N_p]\!]$ (p_c is called the rank of the processor) (see example in Fig.2.4). The points and edges are then allocated to the lowest-rank processor (p_c) whose cells contain this point or edge. This means that some processors can end up with no vertex (see processor 3 in Fig.2.5, left panel). The same is done at the interfaces (*i.e.* on the external boundaries of the global mesh). The distributed mesh is called mesh_p1_loc.

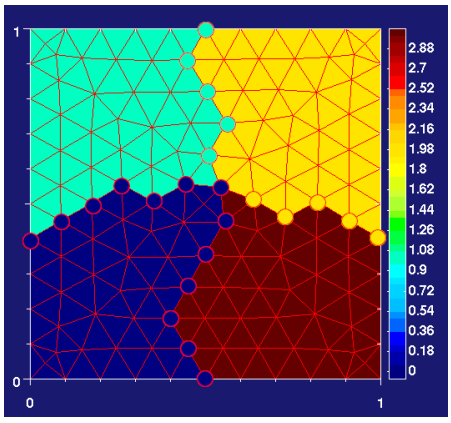


Figure 2.4: Distribution of the points at the edges between different processors: processor 0 in dark blue, processor 1 in cyan, processor 2 in yellow, processor 3 in dark red.

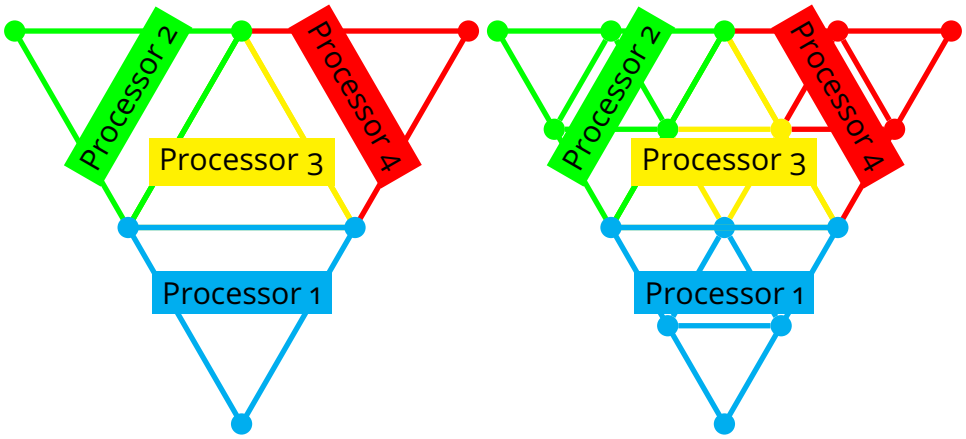


Figure 2.5: Diagram of the distribution of edges and vertices for an extreme case of 4 cells on 4 processors (left panel) and for its refined version (right panel).

Each processor $p_c \in [\![1, N_p]\!]$ will then contain:

- mesh_p1_loc%dom_me = $M(p_c)$ cells such as $\sum_{i=1}^{N_p} M(i) = M$,
- mesh_p1_loc%dom_np = $N(p_c)$ points such as $\sum_{i=1}^{N_p} N(i) = N$,
- mesh_p1_loc%medge = $M_e(p_c)$ edges such as $\sum_{i=1}^{N_p} M_e(i) = M_e$,
- mesh_p1_loc%mes = $M_{es}(p_c)$ edges at interfaces such as $\sum_{i=1}^{N_p} M_{es}(i) = M_{es}$.

As an example, the values for Fig. 2.5 are, with p the processor number :

	Unrefined (left panel)				Refined (right panel)			
p	1	2	3	4	1	2	3	4
$M(p)$	1	1	1	1	4	4	4	4
$N(p)$	3	2	0	1	6	5	1	3
$M_e(p)$	3	3	1	2	9	9	5	7
$M_{es}(p)$	2	2	0	2	4	4	0	4

These numbers are saved by all processors in, for $i \in [\![1, N_p]\!]$:

- mesh_p1_loc%domcell(i) = $M(i)$,
- mesh_p1_loc%domnp(i) = $N(i)$,
- mesh_p1_loc%domedge(i) = $M_e(i)$,

and the cumulative associated quantities are saved in:

- mesh_p1_loc%discell($i+1$) = $1 + \sum_{k=1}^{i} M(k)$,

- mesh_p1_loc%disp($i+1$) = $1+\sum_{k=1}^{i} N(k)$,
- mesh_p1_loc%disedge($i+1$) = $1+\sum_{k=1}^{i} M_e(k)$.

By convention, mesh_p1_loc%discell(1) = mesh_p1_loc%disp(1) = mesh_p1_loc%disedge(1) = 1. These quantities are designed so that, for a point, cell, or edge, it is easy to know which processor it has been assigned to.

Each processor has a local indexation for its attributed cells, vertices, and edges, denoted by a subscript l. There is also a global indexation for all processors denoted by a subscript g. To facilitate the switch between global and local numbering of vertices, edges, and cells, they are reorganized as follows:

- for a cell with global number $m_g \in [\![1, M]\!]$ attributed to processor $p_c \in [\![1, N_p]\!]$ and whose processor number in p_c is $m_l \in [\![1, M(p_c)]\!]$, the relation reads: $m_g = m_l +$ mesh_p1_loc%discell$(p_c) - 1$,
- for a point with global number $v_g \in [\![1, N]\!]$ attributed to processor $p_c \in [\![1, N_p]\!]$ and whose processor number in p_c is $v_l \in [\![1, N(p_c)]\!]$, the relation reads: $n_g = n_l +$ mesh_p1_loc%disp$(p_c) - 1$,
- for an edge with global number $e_g \in [\![1, M_e]\!]$ attributed to processor $p_c \in [\![1, N_p]\!]$ and whose processor number in p_c is $e_l \in [\![1, M_e(p_c)]\!]$, the relation reads: $e_g = e_l +$ mesh_p1_loc%disedge$(p_c) - 1$.

In other words, for cells, the $M(p_c = 1)$ first cells for the global index $\in [\![1, M]\!]$ are attributed to the first processor, the next $M(p_c = 2)$ cells in the global index are attributed to the second processor, and so on. The same goes for points and vertices. This makes it possible to define the function $p(.)$, which uses the global index of an object to return the associated processor. For example, for a point of global index $v_g \in [\![1, N]\!]$, $p(v_g)$ is such that $\sum_{k=1}^{p(v_g)-1} N(k) < v_g \leq \sum_{k=1}^{p(v_g)} N(k)$.

This reordering will change the values in mesh_glob%jj, mesh_glob%jce, mesh_glob%neigh, mesh_glob%rr, mesh_glob%jjs, mesh_glob%neighs and mesh_glob%sides as well as their ordering but it makes the actual distribution on processors much easier. As only the cell's global indices are changed and not their order at the cell numbering level, all the conventions reported in section 2.2.1 are still respected.

Distribution and dealing with external boundaries

Now that each cell, edge, and point is associated with a processor, the objects (%jj, %jce, %neigh, and so on) need to be created so that each processor knows the structure of its local mesh. In addition, other objects are created so that each p processor knows just enough about the cells that are not assigned to it, but which contain at least one point or edge assigned to it. In Fig.2.4, for

the processor 0 (dark blue), this would correspond to all the cyan, red, and yellow cells (associated with processors 1, 2, and 3) that contain one of the big blue dots. The aim is to add to each processor only the necessary information about its boundaries, in order to save memory.

Because of the borders between processors, for some processors:

- some of their cells have neighbours that are not attributed to those processors,
- some vertices of their cells are not attributed to those processors,
- some edges of their cells are not attributed to those processors.

However, to build the matrices needed to solve any problem (see section 2.1.2), each processor needs the following information: for each point v attributed to a processor, this processor needs to be able to construct the global index of all points having a cell in common with v. There are two scenarios in which this processor does not own all the points connected to its attributed point v:

- some points or edges of an attributed cell are not attributed to this processor,
- a cell of an attributed point or edge is not attributed to this processor.

The first scenario happens for a processor if one of its cells' neighbours is attributed to a lower-rank processor and the second scenario is just the other way around. As an example, in Fig.2.4, the first scenario occurs for all the cyan cells (attributed to processor 1) that border the dark blue cells (attributed to processor 0): the processor 1 needs the global index of the big dark blue dots touching any cyan cell. In the same figure, the second scenario arises in the reverse situation, the processor 0 needs to know the information of all cells containing one of the big blue dots for a total of 12 cyan, 2 yellow and 12 red cells. Similarly, the processor 2 requires the information of only 10 red cells (the ones in contact with a big yellow dot).

The first scenario is the easiest to deal with: it is only needed to extend the number of vertices saved by each processor by $N_{\text{extra}}(p_c)$ vertices. Those new vertices are in an attributed cell of p_c but not attributed directly to p_c. The same procedure is used for the additional $M_{e_{\text{extra}}}(p_c)$ edges that are in an attributed cell of p_c but not attributed directly to p_c.

In the second scenario, the processor must save the entire structure of the neighbouring cell. The structure of these $M_{\text{extra}}(p_c)$ cells will be saved in a new table. For example, in Fig. 2.5 (left), processor 1 requires 3 extra cells (2, 3, 4), processor 2 requires 2 extra cells (3, 4), processor 3 requires 1 extra cell (4) and processor 0 none.

For a processor $p_c \in [\![1, N_p]\!]$, the local numbering goes:

- for a cell $m_l \in [\![1, M(p_c)]\!]$,
- for a point $v_l \in [\![1, N(p_c)]\!]$ if it is attributed to the processor, and $v_l \in [\![N(p_c)+1, N(p_c)+N_{\text{extra}}(p_c)]\!]$ otherwise,
- for an edge $e_l \in [\![1, M_e(p_c)]\!]$ if it is attributed to the processor,
- for an edge $e_l \in [\![1, M_{e_{\text{extra}}}(p_c)]\!]$ if it is not attributed to the processor.

This defines new useful constants:

- mesh_p1_loc%np = $N(p_c) + N_{\text{extra}}(p_c)$,
- mesh_p1_loc%mextra = $M_{\text{extra}}(p_c)$,
- mesh_p1_loc%medges = $M_{e_{\text{extra}}}(p_c)$.

As an example, the values for Fig. 2.5 are, with p the processor number :

	Unrefined (left panel)				Refined (right panel)			
p	1	2	3	4	1	2	3	4
$M_{\text{extra}}(p)$	3	2	1	0	5	4	3	0
$N_{\text{extra}}(p)$	0	1	3	2	0	1	5	3
$M_{e_{\text{extra}}}(p)$	0	0	2	1	0	0	4	2

Now that the local and global numbering are fixed, here are all the necessary structures:

- for attributed cells, $m_l \in [\![1, M(p_c)]\!]$
  - mesh_p1_loc%jj(:,m_l) which are the local indices of the cell's points,
  - mesh_p1_loc%neigh(:,m_l) which are the local indices of the neighbouring cells if attributed to p_c, 0 if interface, -1 if not attributed,
  - mesh_p1_loc%jce(:,m_l) which are the global indices of the cell's edges,
  - mesh_p1_loc%i_d(m_l) which is the cell's global index,
- for points, $v_l \in [\![1, N(p_c) + N_{\text{extra}}(p_c)]\!]$
  - mesh_p1_loc%loc_to_glob(v_l) which is the point global index,
  - mesh_p1_loc%rr(:,v_l) which are the (r, z) coordinates of the point,
- for edges at the interface, $m_s \in [\![1, M_{es}(p_c)]\!]$
  - mesh_p1_loc%jjs(:,m_s) which are the global indices of the edge's points,
  - mesh_p1_loc%sides(m_s) which is the interface number,

- mesh_p1_loc%neighs(m_s) which is the local index of the cell containing that edge,

- for edges bordering a lower rank processor, edge $e_l \in [\![1, M_{e_{\text{extra}}}(p_c)]\!]$

  - mesh_p1_loc%jees(e_l) which is the edge's global index,

  - mesh_p1_loc%jecs(e_l) which is the local index of the cell containing that edge,

- for a cell at the borders with a higher rank processor and not attributed to p_c, cell $m_l \in [\![1, M_{\text{extra}}(p_c)]\!]$

  - mesh_p1_loc%extra_jj(:,m_l) which are the global indices of the cell's points,

  - mesh_p1_loc%extra_jce(:,m_l) which are the global indices of the cell's edges,

  - mesh_p1_loc%extra_jcc(m_l) which is the cell's global index.

The tables %jj, %jce, %neigh, %extra_jj, and %extra_jce respect the cell conventions (see section 2.2.1) as they are extracted from the global mesh corresponding tables. Only the output indices are modified, not the actual structure of the cell.

Now that the mesh is locally distributed, mesh_global is deallocated. The objective of ensuring that each processor retains only the necessary mesh information is achieved so that no new tables are created. All that remains is to refine the mesh and create the $\mathbb{P}_2$ and $\mathbb{P}_3$ vertices from the $\mathbb{P}_1$ points.

2.2.4 . Refinement of the distributed $\mathbb{P}_1$ mesh

Once the $\mathbb{P}_1$ mesh has been distributed, it can be refined if necessary. The refinement process is carried out by subdividing any cell into four (see Fig.2.6). The average cell size is thus halved. The refined mesh is called mesh_p1_loc_r.

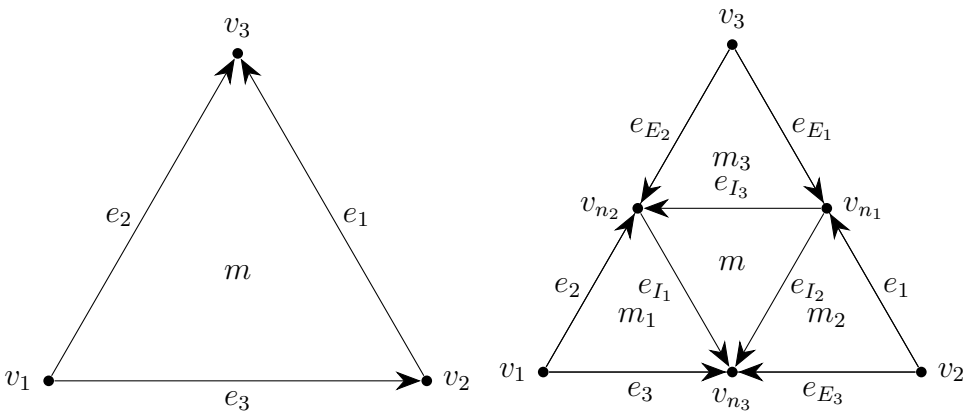


Figure 2.6: Refinement of a cell from the left panel to the right panel. The numbering of cells, points and edges is shown, along with new edge orientations.

For the refined mesh, the numbers of cells, edges, and pointss attributed to a processor are given by:

- mesh_p1_loc_r%dom_me = 4 * mesh_p1_loc%dom_me,
- mesh_p1_loc_r%dom_np = mesh_p1_loc%dom_np + mesh_p1_loc%medge,
- mesh_p1_loc_r%medge = 2 * mesh_p1_loc%medge + 3 * mesh_p1_loc%dom_me.

In other words, there are four times as many cells and one more point per edge. For the edges, every edge is divided into two new ones and there are three new ones to make the center cell (see Fig.2.6).

The updated cumulative quantities and per processor numbers of objects are saved in the same way as before in mesh_p1_loc_r%disxyz and mesh_p1_loc_r%domxyz (where 'xyz' should be replaced by cell, edge or np).

The notation for these numbers (M, M_e, and so on) will include an exponent r if it is the corresponding number for the refined mesh. As an example, the formulas for the new numbers of cells, edges, and points attributed to a processor $p_c \in [\![1, N_p]\!]$ will be written as:

- $M^r(p_c) = 4 * M(p_c)$,
- $M_e^r(p_c) = 2 * M_e(p_c) + 3 * M(p_c)$,
- $N^r(p_c) = N(p_c) + 2 * M_e(p_c)$.

Now the actual structure of the new cells needs to be created.

For the bulk

Let us focus on one processor $p_c \in [\![1, N_p]\!]$. For the rest of this section, we will work with a cell of processor index $m_l \in [\![1, M(p_c)]\!]$, with associated edges with local indices $\{e_{i_l} \in [\![1, M_e(p_c)]\!] | i \in \{1, 2, 3\}\}$ and with associated vertices with local indices $\{v_{i_l} \in [\![1, N(p_c)]\!] | i \in \{1, 2, 3\}\}$, all of them being attributed to the processor p_c.

Processor refinement and numbering are based on the following diagrams (see Fig. 2.6). An easy way to understand it is that, similarly to Fig.2.2, every object with the same subscript number is aligned *e.g.* v_1, e_1, m_1, e_{I_1} and e_{E_1}. For the cells, it means that the center cell m keeps the processor index of the unrefined one. For edges, the refined edge closest to the lowest cell index retains the processor index of the unrefined edge. And, for the points, the ones already existing keep their local index. For $i \in \{1, 2, 3\}$ the cell index of in the unrefined cell, the local indexation for the others follows:

- for the cells:
  - $m^r_{i_l} = M(p_c) + m_l - 1 + i$
- for the edges (we call 'outside edge' the edges (e_{E_i}, e_i) and 'inside edge' the edge e_{I_i} as displayed in Fig. 2.6):
  - for the new outside edges, $e^r_{E_{i_l}} = M_e(p_c) + e_{i_l}$
  - for the new inside edges, $e^r_{I_{i_l}} = 2 * M_e(p_c) + m_l - 1 + i$
- for the points:
  - $v^r_{n_{i_l}} = N(p_c) + e_{i_l}$

The cell-indices of the new cells are such that, for the center, the points are ordered as $(v_{n_1}, v_{n_2}, v_{n_3})$ and, for the corner, the first point is an old one and the two others are in the order of the center cell. In Fig. 2.6, this can be seen by the orientation of the new edges. This is done in order to respect the convention displayed in Fig.2.3 for $\mathbb{P}_1$ finite elements.

The global indices are then constructed by adding $\sum_{k=1}^{p_c-1} M^r(k)$ to the cells, $\sum_{k=1}^{p_c-1} M^r_e(k)$ to the edges and $\sum_{k=1}^{p_c-1} N^r(k)$ to the vertices. This means that, if the processor rank is greater than one, the ones whose local index did not change still have their global index shifted by the number of corresponding objects created in lower-rank processors. This is done in order to keep an easy conversion from local index to global index (see the end of section 2.2.3).

This new indexation allows the construction of mesh_p1_loc_r%jj, mesh_p1_loc_r%jce, mesh_p1_loc_r%i_d, mesh_p1_loc_r%rr and mesh_p1_loc_r%loc_to_glob but only for cells, vertices, and edges attributed to p_c. The position (r, z) of the new vertices v^r_i is simply the middle of the unrefined edge. For example, for v^r_1:

mesh_p1_loc_r%rr(:,$v^r_{1_l}$)= (mesh_p1_loc%rr(:,v_2)+mesh_p1_loc%rr(:,v_3))/2. These structures now only need the indexation for cells and edges bordering a processor with a lower rank as they would not be attributed to p_c (which will be presented in the following).

Before that, one last thing can be built: the neighbours of the new cells. This is done by following again Fig.2.6. The neighbours of the center cell m_l are simply $(m_{l_1}, m_{l_2}, m_{l_3})$ (this order does respect the cell level convention of neighbours, see Fig.2.2). For a corner cell $i \in \{1, 2, 3\}$, the first neighbour is always m_l and the others are determined by using mesh_p1_loc%neigh(:,m_l). The two simplest cases are: if one edge of the refined cell belongs to an external boundary, the value is 0 and, if the neighbour of the unrefined cell m_l is not attributed to the same processor, the value is -1. Lastly, if the cell is also attributed to the processor p_c, it is necessary to reconstruct the cell index of this neighbour. With m_{n_l} the local index of the unrefined neighbours bordering a new cell m_{i_l} and i_n the cell index of the vertex shared by m_{i_l} and m_{n_l}, the new neighbour of m_{i_l} would have a local index of $M(p_c) + m_{n_l} - 1 + i_n$. It is the same formula as above but from the point of view of another cell. This is used to construct all of mesh_p1_loc_r%neigh (if the neighbours cell is not attributed to the processor, it is always assigned the value -1, so there is nothing special to do in these cases).

At the border with a lower-rank processor

The only indexation that is left is for the $N_{\text{extra}}(p_c)$ points and the $M_{e_{\text{extra}}}$ edges that are in one of the cells attributed to p_c and that are on the border with a lower-rank processor. Because of the refinement, new points and edges will be created on this border.

The formulas for the new numbers of edges and points not attributed to the processor p_c but belonging to one of the cells attributed to p_c are:

- $N^r_{\text{extra}}(p_c) = N_{\text{extra}} + M_{e_{\text{extra}}}$,
- $M^r_{e_{\text{extra}}} = 2 * M_{e_{\text{extra}}}$.

Each edge not attributed to p_c will be divided in two and a new point will be created. They all will not be attributed to p_c as the edge is already attributed to another processor.

The local index in p_c of all the N_{extra} unrefined points needs to be shifted by $N^r(p_c) - N(p_c)$ so that, for the refined mesh, we still get that, for $v_l \in [\![1, N^r(p_c)]\!]$, it is attributed to the processor and, for $v_l \in [\![N^r(p_c)+1, N^r(p_c) + N^r_{\text{extra}}(p_c)]\!]$, it is not. In contrast, the $M^r_{e_{\text{extra}}}$ extra edges do not need this shift. The local index in p_c of newly created extra objects simply follows the existing ones, in no particular order, as it has no impact.

However, it is very important that the global indices of the extra points and edges are well constructed. For an already existing point whose processor numbering in the unrefined mesh is $v_l \in [\![N(p_c)+1, N(p_c)+N_{\text{extra}}(p_c)]\!]$, it is possible to get its global index using $v_g =$mesh_p1_loc%loc_to_glob(v_l). With that it is possible to find which processor that vertex is attributed to $p(v_g)$ and, as for the bulk, the new global index only needs to be shifted by the number of vertices created in lower-rank processors: $v_g^r = v_g + \sum_{k=1}^{p(v_g)-1} N^r(k)$.

For an unrefined edge with local numbering $e_l(p_c) \in [\![1, M_{e_{\text{extra}}}(p_c)]\!]$ and global numbering $e_g =$mesh_p1_loc%jees(e_l), two actions need to be done for the refinement: divide it by two and create a new vertex in the middle. Similarly to the vertex, it is possible to get the edge processor $p(e_g)$ and thus get the edge's local numbering in this processor on the unrefined mesh using $e_l(p(e_g)) = e_g - \sum_{k=1}^{p(e_g)-1} M_e^r(k)$.

Hence, constructing the new point is easy. Using the same formula as for the bulk but from the point of view of another processor, the global index of this new vertex is $v_{new_g}^r = N(p(e_g)) + e_l(p(e_g)) + \sum_{k=1}^{p(e_g)-1} N^r(k)$. The sum at the end is the shift on the global index due to the vertices on lower-rank processors.

The same is applicable for the refined edges. As for the bulk, the one closest to the lowest cell numbering vertex will have its global index shifted $e_g^r = e_l(p(e_g)) + \sum_{k=1}^{p(e_g)-1} M_e^r(k)$ while the other will use $e_{new_g}^r = M_e(p(e_g)) + e_l(p(e_g)) + \sum_{k=1}^{p(e_g)-1} M_e^r(k)$. This makes it possible to finish building mesh_p1_loc_r%jees and mesh_p1_loc_r%jecs.

All that remains is to refine the extra cells and the external boundaries.

At the border with an higher-rank processor

The refinement of the $M_{\text{extra}}(p_c)$ extra cells is done in the same way than for the bulk. However, there are two difficulties. Firstly, a recount of the extra cells is needed to keep it to a minimum to preserve memory usage. Secondly, as cells, points, and edges can belong to different processors, the local index in their attributed processor needs to be reconstructed for all objects similarly to what was done in the previous section 2.2.4. The only easy aspect is that the position of points does not need to be calculated.

The counting of the $M_{\text{extra}}^r(p_c)$ new extra cells is similar to the global mesh distribution on the processors (see section 2.2.3). We count the number of refined cells not attributed to the processor p_c but in contact with one of the refined vertices attributed to p_c. For the refined mesh in Fig. 2.5 (right panel), there are hence 5 extra cells for processor 1, 4 extra cells for processor 2, and 3 extra cells for processor 3.

The requirement for the extra cells is to get the global index of each object. It is built using the local indices in their attributed processor and then applying

the usual global shift done to get the global indices.

For an extra unrefined cell of global index v_g with edges global indices e_{i_g} and points global indices v_{i_g}, their local index in their attributed processor is given by:

- $v_l = v_g - \sum_{k=1}^{p(v_g)-1} M^r(k)$,
- for $i \in \{1,2,3\}$, $e_{i_l} = e_{i_g} - \sum_{k=1}^{p(e_{i_g})-1} M_e^r(k)$,
- for $i \in \{1,2,3\}$, $v_{i_l} = v_{i_g} - \sum_{k=1}^{p(v_{i_g})-1} N^r(k)$.

This is done by simply using the conversion from global index to local index (see the end of section 2.2.3).

Now the global index for the refined objects is built following the formulas for the bulk (see section 2.2.4) and adding the shift necessary for the global index. For $i \in \{1,2,3\}$, using the same notation as in Fig. 2.6:

- for the refined extra cells:
  - if it is a center cell, $m_g^r = m_l + \sum_{k=1}^{p(m_g)-1} M^r(k)$,
  - if it is a corner cell, $m_{i_g}^r = M(p(m_g)) + m_l - 1 + i + \sum_{k=1}^{p(m_g)-1} M^r(k)$,
- for the edges of one refined cell (we call 'outside edge' the edges (e_{E_i}, e_i) and 'inside edge' the edge e_{I_i} as displayed in Fig. 2.6):
  - if it is an outside edge closest to the lowest cell index vertex, $e_{i_g}^r = e_{i_l} + \sum_{k=1}^{p(e_{i_g})-1} M_e^r(k)$,
  - if it is an outside edge closest to the highest cell index vertex, $e_{E_{i_g}}^r = M_e(p(e_{i_g})) + e_{i_l} + \sum_{k=1}^{p(e_{i_g})-1} M_e^r(k)$,
  - if it is an inside edge, $e_{I_{i_g}}^r = 2 * M_e(p(m_g)) + m_l - 1 + i + \sum_{k=1}^{p(m_g)-1} M_e^r(k)$,
- for the points:
  - if it is an existing point, $v_{i_g}^r = v_{i_l} + \sum_{k=1}^{p(v_{i_g})-1} N^r(k)$,
  - if it is a new point on the edge e_{i_g}, $v_{n_{i_g}}^r = N(p(e_{i_g})) + e_{i_l} + \sum_{k=1}^{p(e_{i_g})-1} N^r(k)$.

For new objects, the processor $p(.)$ used is in the above formulas is the one they will be attributed to. As an example, for the outside edges, the processor of the unrefined edges is used whereas, for inside edges, the processor of the unrefined cell is used. These new global indices are what is necessary and sufficient to build mesh_p1_loc_r%extra_jj, mesh_p1_loc_r%extra_jce, and mesh_p1_loc_r%extra_jcc.

External boundaries

The only structures left to implement are mesh_p1_loc_r%jjs(:,:), mesh_p1_loc_r%sides(:), and mesh_p1_loc_r%neighs(:). They correspond to the structure of the outside edge of the total mesh and are used to impose boundary conditions. As these edges are in contact with only one cell, their refinement is simple. Hence, the number of refined edges at the external boundaries is $M^r_{es}(p_c) = M_{es}(p_c)$. However the edge's vertices (encoded by mesh_p1_loc%jjs(:,:)) can belong to another processor (for example, in Fig 2.4, for the bottom and left most cyan left, one of its edges is part of the external boundary but have one point attributed to the blue processor).

For an unrefined edge $m_s \in [\![1, M_{es}(p_c)]\!]$, the interface number $\in [\![1, N_{interface}]\!]$ of the refined edges will be the same as m_s. However, in order to get the cells and points of the refined edges, it is necessary to know the cell number of the unrefined edge. Using that the cell of m_s is given by $m =$mesh_p1_loc%neighs(m_s), the cell number i of m_s is obtained by finding which vertex of m does not belong to points of m_s (given by mesh_p1_loc%jjs(:,m_s)).

We will now consider that $i = 1$ so that we can use the unrefined edge e_1 in Fig. 2.6 as an example (meaning that we consider e_1 as being part of the external boundary of the global mesh). Using Fig. 2.6, it is now simple to find the index of the points and cells bordering the refined edges by just copying the values of the already built cells m_2 and m_3. The local indices of m_2 and m_3 are simple to get by using that they are neighbours of the cell m.

End result

Once all these steps are done, we end up with the refined mesh called mesh_p1_loc_r, which is fully structured and respects the convention discussed in section 2.2.1. The unrefined mesh_p1_loc mesh is now deallocated. The refined mesh is renamed mesh_p1_loc. This refinement process can be repeated as many times as necessary to create finer and finer meshes (see Fig.2.1).

2.2.5 . Creation of the distributed $\mathbb{P}_2/\mathbb{P}_3$ mesh

The mesh_p1_loc mesh previously created (and refined if necessary) is still only defined on the $\mathbb{P}_1$ finite elements. It is fine as is for the pressure field if we use $\mathbb{P}_1$/$\mathbb{P}_2$ Hood-Taylor elements for the Navier-Stokes equations. However, we still need at least $\mathbb{P}_2$ elements for the velocity field. Creating $\mathbb{P}_2$ and (optionally) $\mathbb{P}_3$ elements only means adding new points to all of the cells according to Fig. 2.3. This is very simple, as all that is required is to extend the following structures:

- mesh_p1_loc%jj(:,:) by adding the new points to the cells,
- mesh_p1_loc%loc_to_glob(:) by creating the processor and global index of the new vertices,

- mesh_p1_loc%rr(:,:) by creating the (r, z) coordinates of the new points,
- mesh_p1_loc%jjs(:,:) by adding the new points to the edges at the external boundaries of the global mesh,
- mesh_p1_loc%extra_jj(:,:) by adding their global index to the extra cells.

The newly created mesh will be called mesh_p2_loc and mesh_p3_loc in the following sections. In the code, the created meshes are named pp_mesh (for the pressure field) and vv_mesh (for the velocity field). For the various quantities defined on the new mesh, a superscript 1, 2, or 3 will be used depending on the finite elements considered. Again, we will be working with a processor $p_c \in [\![1, N_p]\!]$.

The number of points will now be:

- for $\mathbb{P}_2$, $N^2 = N^1 + M_e^1$,
- for $\mathbb{P}_3$, $N^3 = N^1 + 2 * M_e^1 + M^1$.

For the bulk

For the rest of this section, we will be working with a cell with local index $m_l \in [\![1, M(p_c)]\!]$, with associated edges with local indices $\{e_{i_l} \in [\![1, M_e(p_c)]\!] | i \in \{1, 2, 3\}\}$ and with associated points with local indices $\{v_{i_l} \in [\![1, N(p_c)]\!] | i \in \{1, 2, 3\}\}$ with all of them being attributed to the same processor $p_c \in [\![1, N_p]\!]$.

The local indices of the vertices do not change. However, we still need to create the other nodes and their indices. The cell numbering of the new nodes will follow the diagram shown in Fig. 2.3. For an edge e_{i_l} with cell number $i \in \{1, 2, 3\}$, the local index of the new nodes will be:

- for $\mathbb{P}_2$, $v^2_{n_{i_l}} = e_{i_l} + N^1(p_c)$,
- for $\mathbb{P}_3$,
  - for the one closest to the lowest cell number vertex (4,6,8 in Fig. 2.3), $v^3_{n_{1_{i_l}}} = e_{i_l} + N^1(p_c)$,
  - for the one closest to the highest cell number vertex (5,7,9 in Fig. 2.3), $v^3_{n_{2_{i_l}}} = e_{i_l} + M_e^1(p_c) + N^1(p_c)$,
  - for the point in the middle of the cell, $v^3_{c_l} = m_l + 2 * M_e^1(p_c) + N^1(p_c)$.

and their position is created as follows, using $x_{1(i)}$ (resp. $x_{2(i)}$) the position of the vertex of lowest (resp. highest) cell number of e_{i_l} and x_i the position of the vertex opposite of the edge e_{i_l}:

- for $\mathbb{P}_2$, mesh_p2_loc%rr($v^2_{n_{i_l}}$)$= (x_{1(i)} + x_{2(i)})/2$,
- for $\mathbb{P}_3$,

- for the one closest to the lowest cell number vertex, mesh_p3_loc%rr($v^3_{n_{1i_l}}$)= $(2 * x_{1(i)} + x_{2(i)})/3$,
- for the one closest to the highest cell number vertex, mesh_p3_loc%rr($v^3_{n_{2i_l}}$)= $(x_{1(i)} + 2 * x_{2(i)})/3$,
- for the point in the middle of the cell, mesh_p3_loc%rr($v^3_{n_{i_l}}$)= $(x_{1(i)} + x_{2(i)} + x_i)/3$.

As with refinement, the global index of all points (old and new) is then created (or updated) by adding $\sum_{k=1}^{p_c-1} N^n(k)$ to their local index. This makes it possible to almost fully create mesh_p(2/3)_loc%jj, mesh_p(2/3)_loc%loc_to_glob and mesh_p(2/3)_loc%rr. Only the points at the border with a lower-rank processor now need to be created or updated.

At the border with a lower-rank processor

Creating the $\mathbb{P}_2$ and $\mathbb{P}_3$ meshes generates new nodes on edges that are not attributed to the processor p_c but that are part of one of p_c's cells. Hence, the number of extra nodes becomes:

- for $\mathbb{P}_2$, $N^2_{\text{extra}}(p_c) = N^1_{\text{extra}} + M^1_{e_{\text{extra}}}$,
- for $\mathbb{P}_3$, $N^3_{\text{extra}}(p_c) = N^1_{\text{extra}} + 2 * M^1_{e_{\text{extra}}}$.

Firstly, the local index $v_l \in [\![N^1(p_c) + 1, N^1(p_c) + N^1_{\text{extra}}(p_c)]\!]$ of the node whose cell number is $i \in \{1, 2, 3\}$ needs to be shifted by $N^{2/3}_{\text{extra}}(p_c) - N^1_{\text{extra}}(p_c)$. This is done so that, once again, for $v_l \in [\![1, N^{2/3}(p_c)]\!]$, the point is attributed to the processor p_c and, for $v_l \in [\![N^{2/3}(p_c) + 1, N^{2/3}(p_c) + N^{2/3}_{\text{extra}}(p_c)]\!]$, it is not.

For the newly created nodes, as they are created on an edge not attributed to p_c (local index $e_l \in [\![1, M_{e_{\text{extra}}}(p_c)]\!]$ in p_c), they will be attributed to the edge's processor.

As with refinement, the edge global numbering is e_g =mesh_p1_loc%jees(e_l) and its attributed processor will be $p(e_g)$. Hence the edge's local numbering in that processor $p(e_g)$ is $e_l(p(e_g)) = e_g - \sum_{k=1}^{p(e_g)-1} M^r_e(k)$. The global indices of the new points are now easily created:

- for $\mathbb{P}_2$, $v^2_{n_g} = e_l(p(e_g)) + N^1(p(e_g)) + \sum_{k=1}^{p(e_g)-1} N^2(k)$,
- for $\mathbb{P}_3$,
  - for the one closest to the lowest cell number vertex, $v^3_{n_{1g}} = e_l(p(e_g)) + N^1(p(e_g)) + \sum_{k=1}^{p(e_g)-1} N^3(k)$,
  - for the one closest to the highest cell number vertex, $v^3_{n_{2g}} = e_l(p(e_g)) + M^1_e(p(e_g)) + N^1(p(e_g)) + \sum_{k=1}^{p(e_g)-1} N^3(k)$.

The positions of these points are created in the same way as for the bulk by using the positions of the end points of this edge. This finishes the creation of mesh_p(2/3)_loc%jj, mesh_p(2/3)_loc%loc_to_glob and mesh_p(2/3)_loc%rr.

At the border with an higher-rank processor

The extra cells are also needed to use $\mathbb{P}_2$ and $\mathbb{P}_3$ finite elements. The same process as for the bulk is used but it is not necessary to calculate the position of the points. As with refinement, only the global index needs to be calculated. We will now work with an extra cell with global index m_g.

Similarly to the refinement process, every vertex already created (points (1,2,3) in Fig. 2.3) has its global index shifted by $\sum_{k=1}^{p(v_g)-1} N^{2/3}(k) - \sum_{k=1}^{p(v_g)-1} N^1(k)$ where $p(v_g)$ is its attributed processor.

For the points created on the edges, it is the exact same process as explained in the previous section. For an edge of global index e_{i_g} with $i \in \{1,2,3\}$, attributed processor $p(e_g)$ and processor index in $p(e_g)$ being $e_l(p(e_g))$, the global indices of the points created on this edge are:

- for $\mathbb{P}_2$, $v^2_{n_g} = e_l(p(e_g)) + N^1(p(e_g)) + \sum_{k=1}^{p(e_g)-1} N^2(k)$,
- for $\mathbb{P}_3$,
  - for the one closest to the lowest cell number vertex, $v^3_{n_{1_l}} = e_l(p(e_g)) + N^1(p(e_g)) + \sum_{k=1}^{p(e_g)-1} N^3(k)$,
  - for the one closest to the highest cell number vertex, $v^3_{n_{2l}} = e_l(p(e_g)) + M^1_e(p(e_g)) + N^1(p(e_g)) + \sum_{k=1}^{p(e_g)-1} N^3(k)$.

For the $\mathbb{P}_3$ mesh, the global index of node 10 is given using the processor index of the cell: $m_l = m_g - \sum_{k=1}^{p(m_g)-1} M^r(k)$. The global index of this node will be :

- $v^3_{n_g} = m_l + 2 * M^1_e(p(m_g)) + N^1(p(m_g)) + \sum_{k=1}^{p(m_g)-1} N^3(k)$.

It is thus possible to fully create mesh_p2_loc%extra_jj and mesh_p3_loc%extra_jj.

External boundaries

All that remains is to add the newly created nodes to the external boundaries of the total mesh and update the global index of the vertices. For every edge $m_s \in [\![1, M_{es}(p_c)]\!]$ at the border, using that m_l =mesh_p1_loc%neighs(m_s) is its cell local index, we just need to copy the global index of the relevant nodes given by mesh_p(2/3)_loc%jj(:,m_l) into mesh_p(2/3)_loc%jjs(:,m_s). The relevant nodes are found using the cell index i of the edge. As an example, for $i = 2$, using Fig. 2.3, the relevant nodes are (1,6,7,3) in that order.

2.3 . Benchmark on fine meshes for solving the Schur complement for the discrete Stokes problem

This section takes the form of an article currently submitted to Computer Methods in Applied Mechanics and Engineering (October 2023).

The aim is to implement and test several preconditioning methods for solving the Stokes problem. The end goal is to see if these methods can compete with the prediction-correction method currently used in the SFEMaNS code in order to accelerate the simulation of the magnetohydrodynamics equations. This could potentially save a lot of computational time making it possible to see long-term evolution in the dynamo mechanisms. The conclusion is that preconditioning methods are still not competitive with prediction-correction methods.

The meshes used for solving the Stokes problem in the following article are successive refinements of Fig. 2.1.

PRECONDITIONING OF THE PRESSURE SCHUR COMPLEMENT OF THE AUGMENTED LAGRANGIAN FORMULATION OF THE GENERALIZED STOKES PROBLEM*

MELVIN CREFF†‡ AND JEAN-LUC GUERMOND†

Abstract. The paper investigates various preconditioning techniques for the generalized Stokes problem based on the augmented Lagrangian formulation. Numerical tests on fine unstructured meshes (100 millions degrees of freedoms) show fast convergence independent of the mesh size and convergence rate increasing with the Reynolds number. The number of GMRES iterations for the whole problem to reach the 10^{-10} relative accuracy threshold goes to single digits when the augmented Lagrangian non-dimensional penalty coefficient is equal to 10. But, although very good parallel scalability is observed, thorough tests on large problems reveal that the overall CPU time per degree of freedom and per time step is 10 to 100 time larger than traditional pressure-correction and velocity-correction methods.

Key words. Augmented Lagrangian, Stokes problem, preconditioning, Saddle point problem

AMS subject classifications. 65M60, 65M12, 65M15, 35L45, 35L65

1. Introduction.

The most popular time-stepping techniques for solving the time-dependent Navier-Stokes equations are the so-called projection methods initially proposed by Chorin Chorin [9] and Temam Temam [27]. Although significant progresses have been made since the original work of Chorin and Temam (see e.g., [19] for a brief overview of the literature on the topic), all these methods have however limited accuracy in time. Moreover, when combined with space discretization, they do not allow for a good control of the weak divergence of the velocity field. A better control on the accuracy and on the weak divergence of the velocity can be obtained by using more traditional time-stepping techniques combining implicit and explicit Runge-Kutta techniques. The downside of these techniques though is that one has to solve at each time step a generalized Stokes problem with a saddle point structure. Many iterative techniques to solve this problem have been proposed in the literature (see e.g., Elman et al. [11, Chap. 8] for a review of the literature). They are all more or less based on Uzawa iterations on the pressure Schur complement. The key problem with this type of methods is that the condition number of the Schur complement grows unboundedly as the time step and the viscosity go to zero. This in turn implies that performance of the method strongly depends on how it is preconditioned.

The objective of this paper is to revisit some preconditioning techniques for solving the generalized Stokes problem in the context of time stepping techniques. We revisit in particular the use of the augmented Lagrangian formulation introduced by Fortin and Glowinski [15] and we show that when combined with a preconditioning technique proposed by Cahouet and Chabard [8], one obtains an iterative method that has

*Draft version, October 26, 2023

Funding: This material is based upon work supported in part by the DMS-2110868 (JLG), by the Air Force Office of Scientific Research, USAF, under grant/contract number FA9550-23-1-0007 (JLG), the Army Research Office, under grant number W911NF-19-1-0431 (JLG), and the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contracts B640889, B641173 (JLG). The authors acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC resources that have contributed to the research results reported within this paper. http://www.tacc.utexas.edu

‡Université Paris-Saclay, CNRS, LISN, 91400 Orsay, France

†Department of Mathematics, Texas A&M University, 3368 TAMU, College Station, TX 77843, USA.

converging properties that are uniform with respect to the time step, the mesh size and the viscosity. The analysis is done with the assumption that the time step is proportional to the mesh size to reflect that in practical situations the nonlinearities are treated explicitly and therefore the standard CFL restriction on the time step has to be enforced to achieve stability. Thorough numerical tests are done using PETSC's parallel framework and mixed Hood–Taylor finite elements $\mathbb{P}_2/\mathbb{P}_1$ and $\mathbb{P}_3/\mathbb{P}_2$. The tests are also done with the generic Newtonian form of the viscous tensor inducing nontrivial couplings between the Cartesian components of the velocity. This allow us to test the robustness of the method and its capability to handle nonlinear expressions for the stress tensor.

The solution of the generalized Stokes problem has a long history that started with the landmark papers of Fortin and Glowinski [15] and Cahouet and Chabard [8]. In [15] the authors introduce the augmented Lagrangian method and show that the conditioning of the pressure Schur complement improves as the scalar multiplier of the augmented Lagrangian term increases (cf. e.g., Golub and Greif [17, Prop. 2.1] Benzi and Olshanskii [3, Lem. 4.1] and Proposition 2.2 below). In [8], the authors propose a preconditioner of the Schur complement that behaves well in the two limits $\mu\tau h^{-2} \to 0$ and $\mu\tau h^{-2} \to \infty$ where μ is the viscosity, τ is the time step, and h is the meshsize. The above two approaches are combined in Benzi and Olshanskii [3], Benzi et al. [5], and Benzi and Wang [4]. In these papers the authors show that combining the Cahouet&Chabard preconditioner and the augmented Lagrangian technique gives a good preconditioning method of the Generalized Stokes problem. This method has been numerically evaluated in many papers like Farrell et al. [13], Moulin et al. [23], and Shih et al. [26], where it is indeed shown that the number of MINRES or GMRES iterations used to solve the pressure Schur complement problem behaves well with respect to the mesh size and the viscosity.

The key conclusion of the paper is that although the number of outer iterations to solve the Schur complement is uniform with respect to all the parameters (meshsize, time step, and viscosity), the time efficiency of the method is nowhere near that of projection methods. The main reason is that the method takes too many inner iterations to solve the velocity problem associated with the augmented Lagrangian formulation. The loss of efficiency cannot be reasonably offset by arguing that larger time steps can be used since the nonlinear advection can be made implicit in this context. Our conclusion is that, at the time of this writing, methods based on the augmented Lagrangian and grad-div stabilization methods cannot reasonably compete with projection methods or variations thereof on very large problems requiring iterative solutions of the velocity problem. Of course, some of these problems can be offset on small size problems by using direct solution methods.

The paper is organized as follows. First we formulate the problem in §2. We describe the discrete setting and propose a preconditionner combining ideas from Cahouet and Chabard [8] and Fortin and Glowinski [15]. We discuss in §3 key practical difficulties that arise when implementing this preconditioner. We particularly focus on the parallel aspects of the question. The full preconditioner and some of its variants are tested in §5. Its robustness with respect to the viscosity, the time step and the mesh size are illustrated.

2. Formulation of the problem.

We formulate the problem and introduce the notation in this section.

2.1. Semi-discrete problem.

We consider the generalized Stokes problem arising from the approximation in time of the time-dependent Navier-Stokes equations

in a domain $\mathcal{D} \subset \mathbb{R}^d$. Here $\mathcal{D}$ is an open bounded Lipschitz polyhedron. We denote by $\ell_{\mathcal{D}}$ the diameter of $\mathcal{D}$. The exact form of the time discretization that is adopted to approximate the problem in time is irrelevant for our purpose. Let $\boldsymbol{V}$ be a closed subspace of $\boldsymbol{H}^1(\mathcal{D})$ and Q be a closed subspace of $L^2(\mathcal{D})$. We assume that we are given at each time step a linear form $\boldsymbol{f} \in \mathcal{L}(\boldsymbol{V};\mathbb{R})$, and the problem is reduced to finding a pair $(\boldsymbol{u},p) \in \boldsymbol{V}\times Q$ so that

$$a(\boldsymbol{u},p) - b(\boldsymbol{v},p) + b(\boldsymbol{u},q) = \boldsymbol{f}(\boldsymbol{v}), \qquad \forall (\boldsymbol{v},q) \in \boldsymbol{V}\times Q, \tag{2.1}$$

where the bilinear form a and b are defined by

$$a(\boldsymbol{v},\boldsymbol{w}) := \int_{\mathcal{D}} \left(\frac{1}{\tau} \boldsymbol{v}\cdot\boldsymbol{w} + 2\mu \mathrm{e}(\boldsymbol{v}) : \mathrm{e}(\boldsymbol{w}) \right) \mathrm{d}x, \quad b(\boldsymbol{v},q) := \int_{\mathcal{D}} q \nabla\cdot\boldsymbol{v} \,\mathrm{d}x. \tag{2.2}$$

Here $\mathrm{e}(\boldsymbol{v}) := \frac{1}{2}(\nabla\boldsymbol{v} + (\nabla\boldsymbol{v})^{\mathsf{T}})$ is the strain rate tensor, μ is the shear viscosity, and τ is the time step. The boundary conditions of the problem are encoded in the definition of the space $\boldsymbol{V}$ and the linear form $\boldsymbol{f}$. This problem has a unique solution (see e.g., Girault and Raviart [16], Boffi et al. [6, § 4.2]). Our objective is to construct an iterative method for the solution of a discrete version of (2.1) that performs well with respect to the product $\tau\mu$ and the meshsize.

2.2. Discrete problem. In the rest of this paper we solely focus on a discrete version of (2.1). We consider two sequences of finite-dimensional vector spaces $\{\boldsymbol{V}_h\}_{h\in\mathcal{H}}$ and $\{Q_h\}_{h\in\mathcal{H}}$ where the index set $\mathcal{H}$ is countable and has 0 as unique accumulation point. The velocity is approximated in $\boldsymbol{V}_h$ and the pressure is approximated in Q_h. To simplify some arguments we assume that the velocity approximation is conforming, i.e., $\boldsymbol{V}_h \subset \boldsymbol{V}$. We also assume that the pair $(\boldsymbol{V}_h, Q_h)$ is inf-sup stable, i.e., there is β_h so that

$$\beta_h := \inf_{q_h\in Q_h\setminus\{0\}} \sup_{\boldsymbol{v}_h\in\boldsymbol{V}_h\setminus\{\boldsymbol{0}\}} \frac{b(\boldsymbol{v}_h,q_h)}{\|\boldsymbol{v}_h\|_{\boldsymbol{H}^1(\mathcal{D})}\|q_h\|_{L^2(\mathcal{D})}} > 0. \tag{2.3}$$

The discrete version of (2.1) consists of seeking $\boldsymbol{u}_h \in \boldsymbol{V}_h$ and $p_h \in Q_h$ so that the following holds true for all $\boldsymbol{v}_h \in \boldsymbol{V}_h$ and all $q_h \in Q_h$:

$$a(\boldsymbol{u}_h,\boldsymbol{v}_h) - b(\boldsymbol{v}_h,p_h) = \boldsymbol{f}(\boldsymbol{v}_h), \qquad b(\boldsymbol{u}_h,q_h) = 0. \tag{2.4}$$

Let $\{\boldsymbol{\phi}_i\}_{i\in\mathcal{V}}$ and $\{\psi_k\}_{k\in\mathcal{Q}}$ be bases of $\boldsymbol{V}_h$ and Q_h (say, finite-element-based global shape functions). We define the velocity mass matrix $M_{\boldsymbol{V}}$, the velocity stiffness matrix $E_{\boldsymbol{V}}$, and the pressure mass matrix M_Q by setting

$$M_{\boldsymbol{V},ij} := \int_{\mathcal{D}} \boldsymbol{\phi}_i\cdot\boldsymbol{\phi}_j \,\mathrm{d}x, \qquad \forall i,j \in \mathcal{V} \tag{2.5a}$$

$$M_{Q,ij} := \int_{\mathcal{D}} \boldsymbol{\psi}_i\cdot\boldsymbol{\psi}_j \,\mathrm{d}x, \qquad \forall i,j \in \mathcal{V} \tag{2.5b}$$

$$E_{\boldsymbol{V},kl} := \int_{\mathcal{D}} 2\mathrm{e}(\boldsymbol{\phi}_k) : \mathrm{e}(\boldsymbol{\phi}_l) \,\mathrm{d}x \qquad \forall k,l \in \mathcal{Q}. \tag{2.5c}$$

To formulate the algebraic version of (2.4), we introduce the matrices associated with the bilinear forms a and b:

$$A_{ij} := \frac{1}{\tau} M_{\boldsymbol{V},ij} + \mu E_{\boldsymbol{V},ij} = a(\boldsymbol{\phi}_j,\boldsymbol{\phi}_i), \qquad \forall i,j \in \mathcal{V}, \tag{2.6}$$

$$B_{kj} := b(\boldsymbol{\phi}_j,\psi_k), \qquad \forall k \in \mathcal{Q},\ \forall j \in \mathcal{V}. \tag{2.7}$$

Denoting $\boldsymbol{u}_h := \sum_{i\in\mathcal{V}} \mathsf{U}_i\boldsymbol{\phi}_i$ and $p_h := \sum_{k\in\mathcal{Q}} \mathsf{P}_k\psi_k$ the approximations of the velocity and the pressure, the discrete problem (2.4) is then equivalent to solving the following linear system:

$$A\mathsf{U} - B^{\mathsf{T}}\mathsf{P} = \mathsf{F}, \qquad B\mathsf{U} = 0. \tag{2.8}$$

2.3. The Schur complement. As the coercivity of a implies that the matrix A is invertible, a standard way of solving (2.8) consists of constructing the Schur complement with respect to the pressure. Letting $S := BA^{-1}B^{\mathsf{T}}$, the problem (2.8) is equivalent to finding P and $\mathbf{U}$ so that

$$S\mathsf{P} = -BA^{-1}\mathsf{F}, \qquad A\mathsf{U} = \mathsf{F} + B^{\mathsf{T}}\mathsf{P}. \tag{2.9}$$

As this problem is in general solved iteratively, it is important to have some information on the ℓ^2-condition number of S. The following result can be found in Elman et al. [11, Prop. 5.24] (see also [12, Prop. 50.14] for other informations on the topic).

PROPOSITION 2.1 (Spectrum of S). *Let $\sigma(S)$ be the spectrum of S. Let $\mu_{\min}$, $\mu_{\max}$ be the smallest and largest eigenvalues of the pressure mass matrix M_Q. Let $\|a\|$ and $\|b\|$ be the norms of the bilinear forms a and b. Let α_h be the coercivity constant of a. Then $\sigma(S) \subset [\mu_{\min}\frac{\beta_h^2}{\|a\|}, \mu_{\max}\frac{\|b\|^2}{\alpha_h}]$.*

As $\|a\| \sim \frac{\mu}{\ell_{\mathcal{D}}^{2+\frac{1}{\tau}}}$ and $\alpha_h \sim \frac{\mu}{\ell_{\mathcal{D}}^2}$, the ℓ^2-condition number of S scales like $(1+\frac{\ell_{\mathcal{D}}^2}{\tau\mu})\frac{\|b\|}{\beta_h}$. This number is larger than $(1 + \frac{\ell_{\mathcal{D}}^2}{\tau\mu})$ and grows unboundedly as $\frac{\ell_{\mathcal{D}}^2}{\tau\mu} \to \infty$. Hence, if not properly preconditioned, iterative methods for solving $S\mathsf{P} = -BA^{-1}\mathsf{F}$ have convergence rates that degrade as the viscosity decreases and the time step goes to zero.

The loss of convergence rate mentioned above can be mitigated by using the augmented Lagrangian technique introduced by Fortin and Glowinski [15]. This method replaces the original system (2.8) by the following equivalent system:

$$(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)\mathsf{U} - B^{\mathsf{T}}\mathsf{P} = \mathsf{F}, \qquad B\mathsf{U} = 0, \tag{2.10}$$

where λ is a non-dimensional positive parameter. Here again, one can solve the problem by constructing the Schur complement of the pressure

$$S_{\lambda\mu} := B(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)^{-1}B^{\mathsf{T}}, \tag{2.11a}$$

$$S_{\lambda\mu}\mathsf{P} = -B(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)^{-1}\mathsf{F}, \qquad A\mathsf{U} = \mathsf{F} + B^{\mathsf{T}}\mathsf{P}. \tag{2.11b}$$

The main proporties of this method are summarized in the following standard result (cf. e.g., Golub and Greif [17, Prop. 2.1] Benzi and Olshanskii [3, Lem. 4.1])

PROPOSITION 2.2 (Augmented Lagrangian). *Let $s_\flat$ and $s_\sharp$ be the smallest and largest eigenvalues of S. Then the following holds true:*

$$S_\rho^{-1} = \rho M_Q^{-1} + S^{-1}, \tag{2.12a}$$

$$\sigma(M_Q^{-1}S_\rho) \subset [(\rho + s_\flat^{-1})^{-1}, (\rho + s_\sharp^{-1})^{-1}] \tag{2.12b}$$

The above estimate of the spectrum of $S_{\lambda\mu}$ shows that the ℓ^2-condition number of $S_{\lambda\mu}$ converges to unity as $\lambda\mu\tau/\ell_{\mathcal{D}}^2 \to \infty$. Of course the difficulty is to find a balance, because as the condition number of S converges to unity when $\lambda\mu\tau/\ell_{\mathcal{D}}^2 \to \infty$, the condition number of $(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)$ becomes unrealistically large. In the rest of the paper we discuss preconditioning techniques for $S_{\lambda\mu}$.

2.4. Preconditioning. Ideas to precondition $S_{\lambda\mu}$ can be found by investigating how $S_{\lambda\mu}$ degenerates in the limit $\mu\tau/h^2 \to \infty$ and in the limit $\mu\tau/h^2 \to 0$.

In the limit $\mu\tau/h^2 \to 0$, we have $S_{\lambda\mu} \to \tau B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}}$. The inverse of $S_{\lambda\mu}$ in this limit is then $\tau^{-1}(B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}})^{-1}$.

In the limit $\mu\tau/h^2 \to \infty$, we have $S_{\lambda\mu} \to \mu^{-1}B(E_{\boldsymbol{V}} + \lambda B^{\mathsf{T}}M_Q^{-1}B)^{-1}B^{\mathsf{T}}$. As $E_{\boldsymbol{V}}$ and $B^{\mathsf{T}}M_Q^{-1}B$ have similar spectra, it seems natural to approximate $S_{\lambda\mu}$ in the limit $\mu\tau/h^2 \to \infty$ by $\mu^{-1}(1+\lambda)^{-1}B(B^{\mathsf{T}}M_Q^{-1}B)^{-1}B^{\mathsf{T}}$. Hence we posit that the inverse of $S_{\lambda\mu}$ approximately behave like $\mu(1+\lambda)M_Q^{-1}$ when $\mu\tau/h^2 \to \infty$.

In conclusion, reasoning as in Cahouet and Chabard [8] (see also Bramble and Pasciak [7]), the preconditioner we consider for $S_{\lambda\mu}$ is

$$S_{\lambda\mu}^{-1} \sim C_{\lambda\mu} := \tau^{-1}(B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}})^{-1} + \mu(1+\lambda)M_Q^{-1}. \tag{2.13}$$

In light of the identity (2.12a), the above expression is consistent with the approximation of S_0^{-1} by $\tau^{-1}(B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}})^{-1} + \mu M_Q^{-1}$, which is the preconditionner of the Schur complement S_0 considered in Cahouet and Chabard [8, Eq. (44)]. In this paper, the authors also propose another preconditioner using the discrete pressure Laplacian instead of $B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}}$, but we are going to show that it is important to use $B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}}$ to be uniform with respect of the three parameters τ, μ and h (see preconditioner CaCh_2 defined in (5.2b) and left panel in Figure 2).

3. Solving the velocity problem. We discuss in this section implementation and performance issues associated with the solution to the velocity problem which consists of solving $(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)\mathsf{X} = \mathsf{F}$. We use Dirichlet boundary conditions over the entire boundary of the computational domain; that is, $\boldsymbol{V} := \boldsymbol{H}_0^1(\mathcal{D})$.

3.1. Numerical details. We focus on approximation settings with a continuous representation of the pressure. All the test reported in this section and in §4, §5 are done with continuous Lagrange finite elements in two space dimensions on nonuniform triangular meshes. The parallel implementation is done with PETSC [2]. All the errors that are reported are relative.

Depending on the situation we use either the direct linear solver MUMPS (see Amestoy et al. [1]) or the iterative algebraic multigrid solver BoomerAMG (see Henson and Yang [21]). For completeness and reproducibility we give in Table 1 the options of BoomerAMG that we use.

`-pc_type`	`hypre`
`-pc_hypre_type`	`boomeramg`
`-pc_hypre_boomeramg_strong_threshold`	`0.7 or 0.1`
`-pc_hypre_boomeramg_coarsen_type`	`Falgout`
`-pc_hypre_boomeramg_relax_type_all`	`Chebyshev`

Table 1: BoomerAMG options used in the paper.

In all the tests reported in the paper, the velocity and pressure mass problems involving the matrices $M_{\boldsymbol{V}}$ and M_Q are systematically solved by using BoomerAMG with the strong threshold set to 0.1.

3.2. Preconditioners for the velocity problem. It is very difficult to assemble the matrix coefficients of $A+\lambda\mu B^{\mathsf{T}}M_Q^{-1}B$ when the approximation of the pressure is based on continuous finite elements (like for the $\mathbb{P}_{k+1}/\mathbb{P}_k$ Hood Taylor elements, $k \geq 1$). The first reason is that the inverse of the pressure matrix M_Q^{-1} cannot be

computed. It can be approximated by using some lumping technique though. The second reason is that the sparsity pattern of the matrix is doubled. This drastically increases the memory footprint as well as the communications needed for parallel computing. Therefore, it is significantly easier to solve iteratively this problem by invoking only matrix vector multiplications not requiring any assembling. We now test this idea.

As we use Dirichlet boundary conditions, the bilinear form $\int_{\mathscr{D}} \mathbb{e}(\boldsymbol{u}):\mathbb{e}(\boldsymbol{v})\,\mathrm{d}x$ is also equal to $\int_{\mathscr{D}} \nabla\boldsymbol{u}:\nabla\boldsymbol{v}\,\mathrm{d}x + \int_{\mathscr{D}} \nabla\cdot\boldsymbol{u}\nabla\cdot\boldsymbol{v}\,\mathrm{d}x$ because two integrations by parts show that $\int_{\mathscr{D}} (\nabla\boldsymbol{u})^{\mathsf{T}}:\nabla(\boldsymbol{v})\,\mathrm{d}x = \int_{\mathscr{D}} \nabla\cdot\boldsymbol{u}\nabla\cdot\boldsymbol{v}\,\mathrm{d}x$ for all $\boldsymbol{u},\boldsymbol{v}\in\boldsymbol{H}^1_0(\mathscr{D})$. Hence defining $L_{\boldsymbol{V}}$ to be the stiffness matrix associated with the bilinear form $\int_{\mathscr{D}} \nabla\boldsymbol{v}_h:\nabla\boldsymbol{w}_h\,\mathrm{d}x$ and D to be the stiffness matrix associated with the grad-div bilinear form $\int_{\mathscr{D}} \nabla\cdot\boldsymbol{v}_h\nabla\cdot\boldsymbol{w}_h\,\mathrm{d}x$; i.e.,

$$L_{\boldsymbol{V},ij} := \int_{\mathscr{D}} \nabla\boldsymbol{\phi}_i:\nabla\boldsymbol{\phi}_j\,\mathrm{d}x, \qquad \forall i,j\in\mathcal{V}, \tag{3.1}$$

$$D_{ij} := \int_{\mathscr{D}} \nabla\cdot\boldsymbol{\phi}_i\nabla\cdot\boldsymbol{\phi}_j\,\mathrm{d}x, \qquad \forall i,j\in\mathcal{V}, \tag{3.2}$$

we obtain the following equivalent representation of the matrix A:

$$A = \tau^{-1}M_{\boldsymbol{V}} + \mu(L_{\boldsymbol{V}} + D), \tag{3.3}$$

where $M_{\boldsymbol{V}}$ is the velocity mass matrix. This leads use to consider the following three candidates to precondition the matrix $(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)$:

$$\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}} + \mu(1+\lambda)D = A + \lambda\mu D, \tag{3.4}$$

$$\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}} + \mu D = A, \tag{3.5}$$

$$\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}. \tag{3.6}$$

In the first preconditioner, we replace $B^{\mathsf{T}}M_Q^{-1}B$ by D. In the second preconditioner, we remove $B^{\mathsf{T}}M_Q^{-1}B$. In the third preconditioner, we remove both D and $B^{\mathsf{T}}M_Q^{-1}B$. The stencil for these three preconditioners is standard when using Lagrange finite elements. The matrix coefficients can be easily assembled by looping over the mesh cells. These coefficients can in turn be used to construct incomplete LU factorizations or any other approximations.

3.3. Choice of velocity preconditioner. We now test the three preconditioners (3.4)–(3.6) with continuous Lagrange finite elements in two space dimensions. Setting $k := 4\pi$, we consider the divergence-free vector field $\boldsymbol{u}$ and source term $\boldsymbol{f}$

$$\boldsymbol{u}(\boldsymbol{x}) = \begin{pmatrix}\sin(kx)\sin(ky)\\ \cos(kx)\cos(ky)\end{pmatrix}, \qquad \boldsymbol{f}(\boldsymbol{x}) = \frac{\boldsymbol{u}}{\tau} - 2\mu\mathbb{e}(\boldsymbol{u}). \tag{3.7}$$

Recalling that the velocity shape functions are denoted $\{\boldsymbol{\phi}_i\}_{i\in\mathcal{V}}$, we construct the vector F with entries $\mathsf{F}_i = \int_{\mathscr{D}} \boldsymbol{f}(\boldsymbol{x},t)\cdot\boldsymbol{\phi}_i(\boldsymbol{x})\,\mathrm{d}x$ and solve the linear system

$$(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)\mathsf{U} = \mathsf{F}. \tag{3.8}$$

Setting $\boldsymbol{u}_h := \sum_{i\in\mathcal{V}} \mathsf{U}_i\boldsymbol{\phi}_i$, the relative errors in the L^2-norm reported below are defined to be equal to $\|\boldsymbol{u}-\boldsymbol{u}_h\|_{\boldsymbol{L}^2(\mathscr{D})}/\|\boldsymbol{u}\|_{\boldsymbol{L}^2(\mathscr{D})}$.

We perform tests with $\mathbb{P}_2$ Lagrange elements. The tests are done on five nonuniform meshes composed of 118837, 474065, 1893697, 7569665, 30268417 velocity grid

points, respectively. In order to test the weak scalability of the preconditioning, the computations are done on those fives meshes using 2, 8, 32, 128, 512 processors, respectively. We use the parameters $\mu \in \{1, 10^{-2}, 10^{-4}\}$, with $\lambda = 1$. We also use $\tau = 1$ as our goal is just to test the preconditioners defined above.

The problem (3.8) is solved by using the GMRES solver in PETSC with threshold 10^{-10}. The linear systems induced by the action of the preconditioners are solved by using the iterative algebraic multigrid solver BoomerAMG. The options of BoomerAMG are given in Table 1. We take the strong threshold in BoomerAMG to be equal to 0.7 for the first and second preconditioners (see (3.4)-(3.5)) and equal to 0.1 for the last one (see (3.6)).

	μ	Nb. points	118837	474065	1893697	7569665	30268417
		Nb. Proc.	2	8	32	128	512
$A + \lambda\mu D$	1	GMRES iter.	4	3	2	2	2
		L2 vel. err.	0.188E-06	0.156E-07	0.136E-08	0.154E-08	0.531E-08
		Times (s)	12.3	61.5	176	529	1221
	10^{-2}	GMRES iter.	4	3	2	2	2
		L2 vel. err.	0.178E-06	0.151E-07	0.134E-08	0.118E-09	0.101E-08
		Times (s)	8.10	37.8	126	265	745
	10^{-4}	GMRES iter.	4	3	2	2	2
		L2 vel. err.	0.172E-06	0.149E-07	0.132E-08	0.117E-09	0.104E-10
		Times (s)	3.50	7.85	21.6	36.4	71.0

Table 2: Preconditioning of $A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B$ by $A + \lambda\mu D$, (with $\tau = 1$, $\lambda = 1$, and $\mathbb{P}_2$ Lagrange elements for the velocity), using BoomerAMG, strong threshold 0.7.

	μ	Nb. points	118837	474065	1893697	7569665	30268417
		Nb. Proc.	2	8	32	128	512
A	1	GMRES iter.	4	3	2	3	4
		L2 vel. err.	0.188E-06	0.156E-07	0.133E-08	0.123E-09	0.127E-10
		Times (s)	4.76	8.82	25.4	103	338
	10^{-2}	GMRES iter.	3	3	2	3	3
		L2 vel. err.	0.178E-06	0.151E-07	0.131E-08	0.115E-09	0.282E-10
		Times (s)	3.21	6.75	16.2	51.7	116
	10^{-4}	GMRES iter.	3	2	2	2	2
		L2 vel. err.	0.172E-06	0.149E-07	0.130E-08	0.115E-09	0.310E-10
		Times (s)	2.34	2.58	5.20	8.11	15.7

Table 3: Preconditioning of $A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B$ by A, (with $\tau = 1$, $\lambda = 1$, and $\mathbb{P}_2$ Lagrange elements for the velocity), using BoomerAMG, strong threshold 0.7.

	μ	Nb. points	118837	474065	1893697	7569665	30268417	121053185
		Nb. Proc.	2	8	32	128	512	700
$\tau^{-1} M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}$	1	GMRES iter.	7	5	4	2	1	2
		L2 vel. err.	0.188E-06	0.156E-07	0.133E-08	0.107E-09	0.321E-10	0.325E-10
		Times (s)	3.76	3.41	5.01	3.24	2.37	12.4
	10^{-2}	GMRES iter.	7	5	4	2	2	2
		L2 vel. err.	0.178E-06	0.151E-07	0.131E-08	0.103E-09	0.246E-10	0.335E-10
		Times (s)	3.64	3.83	4.97	3.18	3.50	11.3
	10^{-4}	GMRES iter.	7	5	4	2	2	1
		L2 vel. err.	0.172E-06	0.149E-07	0.130E-08	0.103E-09	0.350E-10	0.361E-10
		Times (s)	3.11	3.07	4.67	2.85	3.16	7.20

Table 4: Preconditioning of $A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B$ by $\tau^{-1} M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}$, (with $\tau = 1$, $\lambda = 1$, and $\mathbb{P}_2$ Lagrange elements for the velocity), using BoomerAMG, strong threshold 0.1.

The results obtained with the preconditioner (3.4) are shown in Table 2. The results with the preconditioner (3.5) are shown in Table 3. The results with the preconditioner (3.6) are shown in Table 4. We show in these tables the number of GMRES iterations that are necessary to reach the assigned (relative) threshold,

which we recall is 10^{-10}. We also show the L^2-norm of the relative error on $\boldsymbol{u}$ and the wall-clock time in seconds to perform each computation.

In all the cases, we observe that the number of GMRES iterations are small. This number clearly converges to 1 for the third preconditioner as the number of grid point goes to infinity. This is because the tests are done with a manufactured velocity field that is divergence-free. We also observe that the L^2-norm of the relative error scales like $\mathcal{O}(h^3)$ which is the expected rate of convergence for $\mathbb{P}_2$ approximation. Notice though that for the firt preconditioner, the error has a strange behavior on the last two meshes. We have verified that to get the proper behavior one has to lower the GMRES threshold to 10^{-14}. This behavior is not observed for the other two preconditioners.

Notice also that, although the number of GMRES iterations for all the preconditioners are roughly the same and very small, the wall-clock time for the first and second preconditioners do not scale well. This is due to the difficulty to coarse-grain the matrix D in the BoomerAMG algorithm. Entirely getting rid of the contribution of the grad-div operator as done in the third preconditioner gives a methods that scales perfectly well (see Table 4). Note that the accuracy and wall-clock time for the third preconditioner are robust with respect to μ, which is not the case of the other two. This series of tests shows in passing that just looking at the number of GMRES iterations, as often done in the literature, is not fully informative. The wall-clock time is also an important factor to differentiate preconditioners.

As the third method works optimally, we did an additional test on a mesh composed of approximately 120 millions grid points using 700 processors. If we had 2048 processor available, the wall-clock times would have been $\frac{700}{4\times512}\times 12.4s = 4.24s$ for $\mu = 1$, $\frac{700}{4\times512}\times 11.3s = 3.86s$ for $\mu = 10^{-2}$, and $\frac{700}{4\times512}\times 7.2s = 2.46s$ for $\mu = 10^{-4}$. We therefore conclude that the scaling is almost perfect. This is also well illustrated in Figure 1 (see the horizontal green curves). Note that for the last two meshes, the L^2-norm of the relative error on $\boldsymbol{u}$ is of the same order as the GMRES threshold. To get back the proper $\mathcal{O}(h^3)$-scaling of the error one would have to decrease the GMRES threshold to 10^{-14}.

The main conclusion of the series of tests done in this section is that although the grad-div operator has very interesting properties emphasized in the literature (Benzi and Olshanskii [3], Olshanskii et al. [24], Guermond and Minev [18], de Frutos et al. [10]), solving problems requiring the solution of linear systems involving the matrix D or variations thereof is difficult (see e.g., Heister and Rapin [20], Jenkins et al. [22], Fiordilino et al. [14]). This conclusion aligns with the recommendations made in Benzi et al. [5] that dropping the coupling blocks in D greatly improves the efficiency of the velocity preconditioner.

3.4. BoomerAMG vs. MUMPS. In this section we compare the use of iterative and direct solvers for the solution of the linear systems induced by the velocity preconditioners. We only test the third preconditioner, $\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}$, as we have shown in the previous section that it is the most efficient one. We compare the direct solution method MUMPS with BoomerAMG. We use the same parameters as in §3.3; that is, $\lambda = 1$ and $\tau = 1$.

The results for MUMPS are shown in Table 5. We recall that those for BoomerAMG are in Table 4. We observe that the number of GMRES iterations and the L^2-norm of the relative error are the same as when using BoomerAMG. A key difference with BoomerAMG is that MUMPS performs first the LU factorization and then does the backward substitution. None of these two operations scale very well. We

$\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}$	μ	Nb. points Nb. Proc.	118837 2	474065 8	1893697 32	7569665 128	30268417 512
	1	GMRES iter.	7	5	4	2	X
		L2 vel. err.	0.188E-06	0.156E-07	0.133E-08	0.106E-09	X
		Sol. times (s)	1.25	1.53	4.16	4.77	X
		LU fact. times (s)	5.42	13.1	68.8	209	X
	10^{-2}	GMRES iter.	7	5	4	2	X
		L2 vel. err.	0.178E-06	0.151E-07	0.131E-08	0.105E-09	X
		Sol. times (s)	1.23	1.31	3.55	4.36	X
		LU fact. times (s)	5.51	13.0	51.3	160	X
	10^{-4}	GMRES iter.	7	5	4	2	X
		L2 vel. err.	0.172E-06	0.149E-07	0.130E-08	0.105E-09	X
		Sol. times (s)	1.20	1.26	3.58	4.54	X
		LU fact. times (s)	5.01	11.3	51.9	206	X

Table 5: Preconditioning of $A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B$ by $\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}$, (with $\tau = 1$, $\lambda = 1$, and $\mathbb{P}_2$ Lagrange elements for the velocity), using MUMPS.

only report here the backward substitution time. We observe that MUMPS' backward substitution time is faster than BoomerAMG for meshes composed of less than 2 million points. But, the memory footprint of MUMPS grows very fast with the size of the problem, making it impossible for us to solve the problem on the mesh composed of only 30 million velocity grid points.

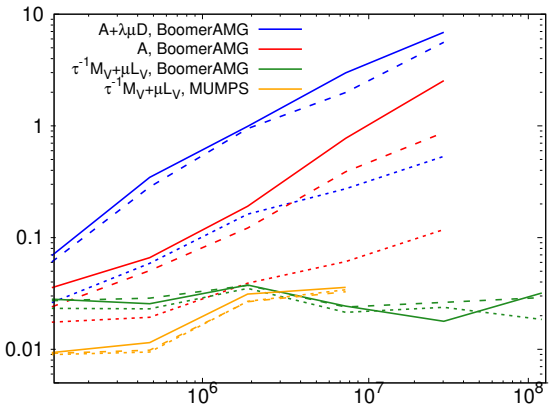


Fig. 1: Preconditioning of $A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B$ using either MUMPS or BoomerAMG for the preconditioner. Y-axis: $\frac{\text{wall-clock time}\times\text{nb. processor}}{\text{nb. vel.+press. degrees of freedom}}$ in millisecond; X-axis: number of velocity grid points. $\mu = 1$ (——); $\mu = 10^{-2}$ (- - -), $\mu = 10^{-4}$ (·····).

In Figure 1, we compare the efficiency of MUMPS using the third preconditioner with the efficiency of BoomerAMG using the three preconditioners discussed in §3.3. The X-axis shows the number of velocity grid points. The Y-axis of the figure shows the elapsed time multiplied by the number of processors and divided by the total number of degrees of freedom (two times the number of velocity grid points plus the number of pressure grid points). The result is shown in millisecond. In the ideal case, this number should be independent of the number of processor and the number of degrees of freedom. We observe that the only method for which this is indeed the case is the third preconditioner solved with BoomerAMG (see the horizontal green curves).

Finally, as already mentioned in §3.2, Figure 1 shows that the wall-clock time per degree of freedom and time step of the other two preconditioners does not behave

optimally, whereas it does for the third preconditioner.

3.5. Tests with non divergence-free solution. We now confirm the conjecture made in §3.3 that the number of GMRES iterations to solve the problem (3.8) converges to 1 as the number of grid point goes to infinity because the solution defined in (3.7) is divergence free.

In this section we test the third preconditioner with a right-hand side F constructed with a velocity field that is not divergence-free. More precisely, we take

$$\boldsymbol{u}(\boldsymbol{x}) = \begin{pmatrix} \sin(2kx)\sin(ky) \\ \cos(kx)\cos(ky) \end{pmatrix}, \qquad \boldsymbol{f}(\boldsymbol{x}) = \frac{\boldsymbol{u}}{\tau} - 2\mu\mathbb{e}(\boldsymbol{u}) - \lambda\mu\nabla\nabla\cdot\boldsymbol{u}, \tag{3.9}$$

with $k = 4\pi$, $\lambda = 1$, and $\tau = 1$. Here again we solve $(A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B)\mathsf{U} = \mathsf{F}$ and the entries of the vector F are given by $\mathsf{F}_i = \int_{\mathcal{D}} \boldsymbol{f}(\boldsymbol{x}, t)\cdot\boldsymbol{\phi}_i(\boldsymbol{x})\,\mathrm{d}x$. The tests are done with $\mathbb{P}_2$ and $\mathbb{P}_3$ finite elements for the velocity. For the $\mathbb{P}_3$ elements the tests are done with meshes composed of 266902, 1065685, 4258897, 17027905, and 68096257 velocity grid points, respectively.

$\mathbb{P}_2$, $\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}$	μ	Nb. points	118837	474065	1893697	7569665	30268417
		Nb. Proc.	2	8	32	128	512
	1	GMRES iter.	13	13	12	12	12
		L2 vel. err.	0.144E-05	0.138E-06	0.117E-07	0.104E-08	0.180E-09
		Times (s)	6.58	8.15	12.8	14.1	16.1
	10^{-2}	GMRES iter.	14	14	14	14	14
		L2 vel. err.	0.144E-05	0.137E-06	0.117E-07	0.104E-08	0.153E-09
		Times (s)	6.48	8.40	14.4	15.3	17.1
	10^{-4}	GMRES iter.	16	16	16	16	16
		L2 vel. err.	0.148E-05	0.139E-06	0.118E-07	0.103E-08	0.988E-10
		Times (s)	6.75	8.93	15.1	16.8	18.4

$\mathbb{P}_3$, $\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}$	μ	Nb. points	266902	1065685	4258897	17027905	68096257
		Nb. Proc.	2	8	32	128	512
	1	GMRES iter.	13	12	12	12	12
		L2 vel. err.	0.172E-06	0.108E-07	0.695E-09	0.158E-09	0.156E-09
		Times (s)	16.1	28.6	50.3	56.1	60.6
	10^{-2}	GMRES iter.	14	14	14	14	14
		L2 vel. err.	0.172E-06	0.108E-07	0.689E-09	0.165E-09	0.118E-09
		Times (s)	26.7	33.8	55.1	60.1	68.6
	10^{-4}	GMRES iter.	16	16	16	16	16
		L2 vel. err.	0.171E-06	0.108E-07	0.679E-09	0.568E-10	0.375E-10
		Times (s)	27.2	36.1	61.2	69.2	73.3

Table 6: Non divergence-free solution. Preconditioning of $A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B$ by $\tau^{-1}M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}}$, (with $\tau = 1$, $\lambda = 1$) using BoomerAMG, strong threshold 0.1. Top: $\mathbb{P}_2$ Lagrange elements for the velocity. Bottom: $\mathbb{P}_3$ Lagrange elements for the velocity

The results are shown in Table 6. At variance with what is observed in Tables 2–4, the number of GMRES iterations is bounded from below away from 1. Depending on the value of μ, the number of GMRES iterations needed to reach the 10^{-10} threshold varies between 12 to 16. This number is remarkably constant when μ is fixed. The L^2-norm of the relative error properly scales like $\mathcal{O}(h^3)$ for the $\mathbb{P}_2$ approximation and like $\mathcal{O}(h^4)$ for the $\mathbb{P}_3$ approximation until the 10^{-10} threshold is reached. Finally, we observe that the weak scaling is satisfactory for both the $\mathbb{P}_2$ and $\mathbb{P}_3$ approximations.

That the velocity preconditioner takes more time to solve a problem where the solution has a nonzero divergence will impact the performance of the preconditioner of the pressure Schur complement.

3.6. Solving the velocity problem with discontinuous pressure. In the case of an approximation of the pressure based on discontinuous finite elements (like

the Scott–Vogelius elements [25] and their generalizations) the coefficient of the matrix $B^{\mathsf{T}}M_Q^{-1}B$ can be computed. This is possible because the support of the discontinuous finite elements for the pressure are localized on each mesh cell; as a result, the pressure matrix M_Q can be locally inverted on each cell. Hence, it is possible to construct the coefficients of the matrix associated with the velocity problem. Note in passing that, contrary to $B^{\mathsf{T}}M_Q^{-1}B$, the stencil of the pressure matrix $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ is not simple, which means that computing the entries of this matrix in parallel is non trivial (see discussion in §4.3).

4. Implementation of the preconditioner $C_{\lambda\mu}$. We discuss in this section the implementation of the preconditionner (2.13). We mainly focus our attention on the matrix $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ as the linear algebra associated with the pressure mass matrix M_Q is simple.

4.1. The matrix $BM_{\mathbf{V}}^{-1}B^{\mathsf{T}}$. Here again, the matrix coefficients of $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ cannot be easily computed when using continuous elements for the pressure. Even if the velocity mass matrix is lumped, the stencil of the resulting simplified matrix is double. Hence, the only practical way to solve the linear system $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}\mathsf{X} = \mathsf{Y}$ is to use a preconditioned iterative method. A natural preconditioner for this matrix is the pressure Laplacian L_Q whose entries are given by

$$L_{Q,kl} := \int_{\mathcal{D}} \nabla\psi_k \cdot \nabla\psi_l \,\mathrm{d}x \qquad \forall k, l, \in \mathcal{Q}. \tag{4.1}$$

However, this matrix is singular when Dirichlet boundary conditions are enforced on the velocity over the entire boundary of the domain $\mathcal{D}$. As we only want to construct a preconditioner, one can consider instead the matrix with entries $\epsilon M_Q + L_Q$ with $\epsilon > 0$.

$\mathbb{P}_1$	Nb. points	29870	118837	474065	1893697	7569665
	Nb. Proc.	2	8	32	128	512
	GMRES iter.	8	8	7	7	6
	L1 press. err.	0.311E-10	0.902E-11	0.294E-10	0.111E-10	0.356E-10
	Times (s)	2.84	3.65	5.60	5.61	5.05
$\mathbb{P}_2$	Nb. points	118837	474065	1893697	7569665	30268417
	Nb. Proc.	2	8	32	128	512
	GMRES iter.	10	9	9	8	7
	L^1 press. err.	0.142E-10	0.279E-10	0.145E-10	0.334E-10	0.302E-10
	Times (s)	13.7	15.8	26.0	24.5	22.4

Table 7: Tests for the preconditioning of $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ by $\epsilon M_Q + L_Q$ for $\mathbb{P}_1$ and $\mathbb{P}_2$ Lagrange elements, using BoomerAMG, strong threshold 0.1.

As in §3.2, we illustrate this idea on continuous Lagrange finite elements in two space dimensions. We consider $\mathbb{P}_2/\mathbb{P}_1$ and $\mathbb{P}_3/\mathbb{P}_2$ elements. Recall that these finite elements are legitimate for the Navier-Stokes equations. For each mesh we build the right-hand $\mathsf{Y} := BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}\mathsf{X}$ by applying the matrix $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ to some given pressure field. Here X is the vector whose entries are the value of the following pressure field at the pressure grid points:

$$p(x, y) := \sin(4\pi(x - y)). \tag{4.2}$$

We use $\epsilon = 1$ in the preconditioner $\epsilon M_Q + L_Q$. We use BoomerAMG with strong threshold equal to 0.1 to solve the linear system induced by the preconditioner. We

show in Table 7 the L^1-norm of the relative error when the GMRES threshold is reached. As the GMRES threshold is 10^{-10}, we observe that the errors are of order 10^{-10}, (these are not approximation errors since there is noting to approximate). The tests are done on five different meshes with mixed Lagrange elements. This test shows that $\epsilon M_Q + L_Q$ is an excellent preconditioner of $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ as the number of GMRES iterations is small and decreases as the mesh is refined.

4.2. Using a lumped mass matrix instead M_V^{-1}. Even though Table 7 shows that the number of GMRES iterations for solving $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}\mathsf{P} = \mathsf{F}$ is small, the time for solving this problem may seem a bit too high for such a simple problem. The analysis of the problem reveals that most of the computation time is spent inverting the velocity mass matrix $M_{\boldsymbol{V}}$. As the matrix $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ is only used as part of the preconditioner for $S_{\lambda\mu}$, replacing the mass matrix $M_{\boldsymbol{V}}$ by a lumped one could save some computation time. To explore this idea, we then propose to replace $M_{\boldsymbol{V}}$ by a diagonal matrix $\Lambda_{\boldsymbol{V}}$ with entries defined by

$$\Lambda_{\boldsymbol{V},ij} := \delta_{ij}\int_{\mathcal{D}} |\varphi_i(\boldsymbol{x})|\,\mathrm{d}x, \qquad \forall i,j \in \mathcal{V}. \tag{4.3}$$

Taking the absolute value of the shape function is important here as $\mathbb{P}_2$ and $\mathbb{P}_3$ velocity shape functions are not uniformly positive. Then instead of solving $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}\mathsf{P} = \mathsf{F}$, we propose to solve $B\Lambda_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}\mathsf{P} = \mathsf{F}$ while again preconditioning the system with $\epsilon M_Q + L_Q$.

$\mathbb{P}_1$	Nb. points	29870	118837	474065	1893697	7569665
	Nb. Proc.	2	8	32	128	512
	GMRES iter.	14	13	13	13	12
	L1 press. err.	0.233E-10	0.506E-10	0.439E-10	0.474E-10	0.559E-09
	Times (s)	1.66	1.75	2.56	2.59	2.83
$\mathbb{P}_2$	Nb. points	118837	474065	1893697	7569665	30268417
	Nb. Proc.	2	8	32	128	512
	GMRES iter.	15	15	14	14	14
	L^1 press. err.	0.446E-10	0.446E-10	0.318E-09	0.308E-09	0.356E-09
	Times (s)	8.22	9.79	12.5	13.3	14.2

Table 8: Tests for the preconditioning of $B\Lambda_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ by $\epsilon M_Q + L_Q$ for $\mathbb{P}_1$ and $\mathbb{P}_2$ Lagrange elements, using BoomerAMG, strong threshold 0.1.

We show the results of this test in Table 8. We observe that even though using $\Lambda_{\boldsymbol{V}}$ instead of $M_{\boldsymbol{V}}$ takes more GMRES iterations, the wall-clock time is much smaller and the method scales very well. Once again the tests are purely algebraic; as a result, the errors are proportional to the GMRES threshold.

4.3. Discontinuous finite elements for pressure. The matrix coefficients of $BM_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ are given by $\sum_{i,j\in\mathcal{V}}\int_{\mathcal{D}}\psi_k\nabla\cdot\boldsymbol{\phi}_i\,\mathrm{d}x(M_{\boldsymbol{V}}^{-1})_{ij}\int_{\mathcal{D}}\psi_l\nabla\cdot\boldsymbol{\phi}_j\,\mathrm{d}x$, for all $k,l\in\mathcal{Q}$. If the velocity mass matrix is lumped or replaced by a diagonal matrix with the right scaling, the computation of the matrix's coefficients are possible. However, we observe that the velocity degrees of freedom supported at the mesh interfaces interact with all the pressure degrees of freedom on the cell touching this interface. As a result, for each cell K, the stencil of the pressure degrees of freedom located on K are composed of all the pressure degrees of freedom located on all the cells touching K (either by a face, an edge or a vertex). Hence the pressure stencil is significantly larger than that of the velocity. This type of stencil is very inconvenient for parallel processing. In conclusion, it is again more practical to use an iterative method preconditioned by a discontinuous Galerkin approximation of the Laplacian.

5. Numerical illustration of the performance of $C_{\lambda\mu}$. We now illustrate the performance of the method (2.11b) using the preconditioner $C_{\lambda\mu}$ defined in (2.13). We compare various realizations of the method. We demonstrate numerically that (2.13) is indeed a preconditioner that is robust with respect to μ, τ and the meshsize. To be representative of actual situations where the time step deceases like the mesh size due to nonlinearities being made explicit in time, we set

$$\tau = N^{-\frac{1}{2}}, \tag{5.1}$$

where N is the total number of grid points for the velocity. Here $N^{-\frac{1}{2}}$ scales like the mesh size because the tests are done in dimension two.

5.1. Tests without augmented Lagrangian. We start by illustrating the method proposed in Cahouet and Chabard [8, Eq. (44)]. This method does not use the augmented Lagrangian, which is equivalent to take $\lambda = 0$ in (2.11b) and (2.13). The authors consider two preconditioners for $S_{\lambda\mu}$ which we call CaCh_1 and CaCh_2:

$$\mathrm{CaCh}_1 := \tau^{-1}(B\ M_{\boldsymbol{V}}^{-1}B^{\mathsf{T}})^{-1} + \mu M_Q^{-1}, \tag{5.2a}$$

$$\mathrm{CaCh}_2 := \tau^{-1}(L_Q)^{-1} + \mu M_Q^{-1}. \tag{5.2b}$$

It is observed in [8] that CaCh_1 outperforms CaCh_2 when μ is large. The test reported therein also suggest that these two preconditioners are somewhat equivalent when the viscosity μ is small, (see Figure 6 in [8]). We show in this section that this is not the case on fine meshes.

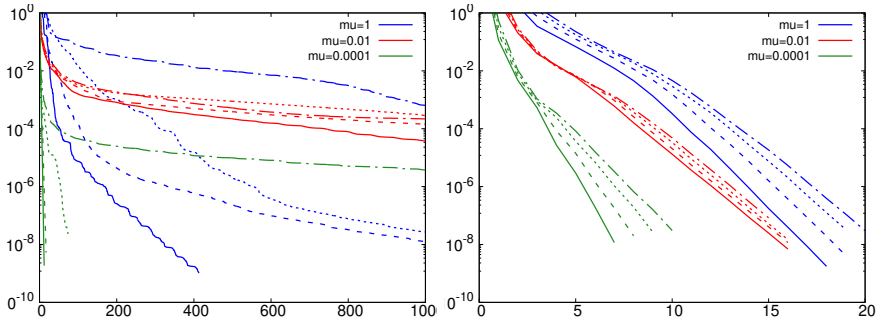


Fig. 2: GMRES residual vs. iteration count for the preconditioners CaCh_2 (left panel) and CaCh_1 (right panel) with $\mu = 1$ (blue), $\mu = 10^{-2}$ (red), and $\mu = 10^{-4}$ (green). Number of grid points: 29857 (——); 118785 (- - -), 473857 (·····) ; 1892865 (– · –).

We report in Figure 2 the GMRES residual (reported by PETSC) as a function of the iteration count for various small values of the viscosity ($\mu \in \{1, 10^{-2}, 10^{-4}\}$) and various mesh refinements. The simulations are done with mixed $\mathbb{P}_2/\mathbb{P}_1$ continuous finite elements. We test four meshes with the following velocity/pressure grid point counts: Mesh 1 (118837/29870); Mesh 2 (474065/118837); Mesh 3 (1893697/474065); Mesh 4 (7569665/1893697). As the velocity field is two dimensional, the total number of degrees of freedom is for each mesh: Mesh 1 (267544); Mesh 2 (1066967); Mesh 3 (4261459); Mesh 4 (17033027). We show in the left panel of Figure 2 the performance of the method with the preconditioner $\tau^{-1}(L_Q)^{-1} + \mu M_Q^{-1}$. The performance of the method with the preconditioner CaCh_1 is shown in the right panel of Figure 2. We observe that the preconditioner CaCh_2 does not behave properly. The convergence

rate strongly depends on the viscosity and the mesh size. The method seems to converge well on coarse meshes, but its performance becomes erratic when the meshes are refined. The method behaves well over the ten to twenty first iterations and then the convergence rate flattens. On the other hand, the method based on the preconditionner CaCh_1 works reasonably well. Notice though that the performance is not uniform with respect to μ and there seems to be a slight dependency on the mesh size.

We report in Table 9 the performance of the method using the preconditioner CaCh_1 with the stopping threshold equal to 10^{-10}. We show in this table the number of GMRES iterations that are required to reach the assigned threshold. We also show the relative L^1-norm of the error on the pressure, the relative L^2-norm of the error on the velocity, and the L^1-norm of the weak divergence. We observe the expected second-order convergence rate on the pressure and third-order convergence rate on the velocity. We also observe that the weak divergence is well below the assigned threshold.

μ	Nb. points Nb. Proc.	Mesh 1 2	Mesh 2 8	Mesh 3 32	Mesh 4 128
1	GMRES iter.	18	19	19	20
	L1 press. err.	0.773E-03	0.193E-03	0.482E-04	0.121E-04
	L2 vel. err.	0.177E-06	0.151E-07	0.131E-08	0.116E-09
	L1 weak div.	0.132E-15	0.349E-16	0.309E-16	0.321E-17
10^{-2}	GMRES iter.	16	16	16	15
	L1 press. err.	0.771E-03	0.193E-03	0.482E-04	0.121E-04
	L2 vel. err.	0.150E-05	0.134E-06	0.111E-07	0.956E-09
	L1 weak div.	0.155E-13	0.303E-14	0.431E-15	0.261E-15
10^{-4}	GMRES iter.	7	8	9	10
	L1 press. err.	0.770E-03	0.193E-03	0.482E-04	0.121E-04
	L2 vel. err.	0.705E-04	0.744E-05	0.764E-06	0.762E-07
	L1 weak div.	0.789E-13	0.195E-13	0.461E-14	0.267E-14

Table 9: Preconditioner CaCh_1 with $\lambda = 0$.

5.2. Tests with augmented Lagrangian. We now test the method with the augmented Lagrangian term, i.e., we solve (2.11b) with the preconditionner (2.13) where $M_{\boldsymbol{V}}$ is replaced by $\Lambda_{\boldsymbol{V}}$, i.e.,

$$C_{\lambda\mu} := \tau^{-1}(B\Lambda_{\boldsymbol{V}}^{-1}B^{\mathsf{T}})^{-1} + \mu(1+\lambda)M_Q^{-1}. \tag{5.3}$$

We test the method with $\mathbb{P}_2/\mathbb{P}_1$ and $\mathbb{P}_3/\mathbb{P}_2$ elements with $\lambda \in \{1, 10\}$ and compare the results with those obtained with the preconditioner CaCh_1 (which we recall corresponds to setting $\lambda = 0$). For the $\mathbb{P}_2/\mathbb{P}_1$ approximation we use the meshes already considered above with the following velocity/pressure grid point counts: Mesh 1 (118837/29870); Mesh 2 (474065/118837); Mesh 3 (1893697/474065); Mesh 4 (7569665/1893697). For the $\mathbb{P}_3/\mathbb{P}_2$ approximation we use meshes with the following velocity/pressure grid point counts: Mesh 1 (66937/29857); Mesh 2 (266785/118785); Mesh 3 (1065217/473857); Mesh 4 (4257025/1892865).

We report the results in Figure 3. Here again we show GMRES residuals versus the GMRES iteration count. We observe now that the convergence rate improves as λ increases and is almost uniform with respect to the mesh size and the viscosity for $\lambda = 10$. It takes between 7 to 15 iteration to reduce the residual by the factor 10^{-10} when $\lambda = 1$, whereas it only takes between 7 to 9 iteration for $\lambda = 10$. Hence, the method seems to be close to be optimal when one only look at the GMRES iteration count.

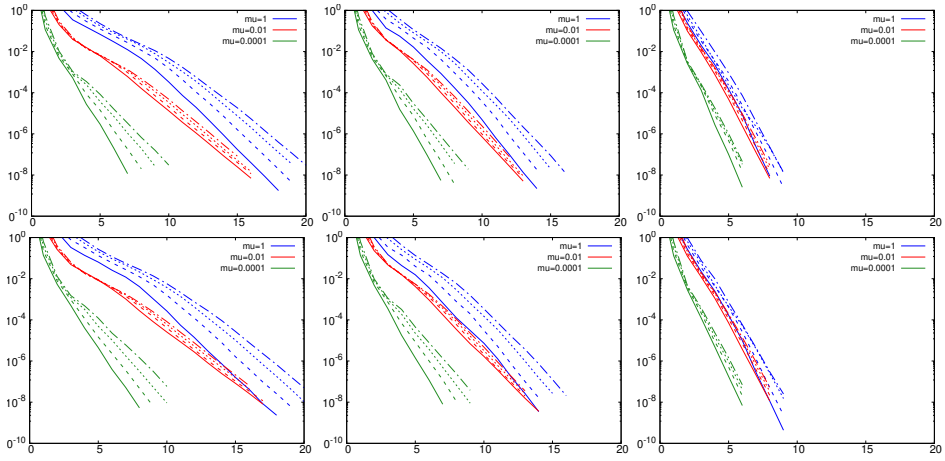


Fig. 3: GMRES residual vs. iteration count for the preconditioner $\tau^{-1}(B\, M_{\boldsymbol{V}}^{-1} B^{\mathsf{T}})^{-1} + \mu(1+\lambda) M_Q^{-1}$ with $\mu = 1$ (blue), $\mu = 10^{-2}$ (red), and $\mu = 10^{-4}$ (green). Mesh 1 (——); Mesh 2 (- - -), Mesh 3 (·····); Mesh 4 (– · –). Top: Hood-Taylor $\mathbb{P}_2/\mathbb{P}_1$ finite elements. Bottom: Hood-Taylor $\mathbb{P}_3/\mathbb{P}_2$ finite elements. Leftmost column: Cahouet&Chabard preconditioner ($\lambda = 0$). Center column: Augmented Lagrangian with $\lambda = 1$. Rightmost column: Augmented Lagrangian with $\lambda = 10$.

Finally, we compare the actual efficiency of the Cahouet&Chabard method CaCh_1 (blue lines) with that of the preconditioned augmented Lagrangian methods with $\lambda = 1$ (red lines) and $\lambda = 10$ (green lines) . For these three methods, we show in Figure 4 the wall-clock time multiplied by the number of processors divided by the total number of degrees of freedom in millisecond:

$$\frac{\text{wall-clock time}\times\text{nb. processor}}{\text{nb. vel.+press. degrees of freedom}}. \tag{5.4}$$

We note also that all the methods scale well. But, quite surprisingly, we observe that all the methods have roughly the same efficiency; that is, between 0.4 to 2 millisecond is spent per degree of freedom on average to solve the problem, irrespective of the size of the problem and the number of processors. Even though, the number of GMRES iteration count is larger for the Cahouet&Chabard method, it takes much more time to solve the velocity problem in the augmented Lagrangian methods as discussed in section 3. Hence, the Cahouet&Chabard method seems to be the most efficient on average.

The efficiency being between 0.4 to 2 millisecond can be compared to the implementations of the preconditioning method proposed by Benzi et al. [5]. In the first implementation reported in Benzi and Wang [4], tests done on a two-dimensional uniform Cartesian mesh (256×256 grid with $\mathbb{Q}_1/\mathbb{Q}_1$ finite elements) and reported in Table 3.12 therein, give an efficiency ranging between 0.2ms and 0.8ms per degree of freedom for the stopping threshold 10^{-6} (which would give 0.33 to 1.33 millisecond with the stopping threshold 10^{-10}). Similarly using the data from Table 5.5 in Farrell et al. [13], we obtain 0.2ms milliseconds for a two-dimensional mesh composed of 1.6×10^5 degrees of freedom with the stopping threshold of 10^{-6} (which would give

0.33ms with the stopping threshold 10^{-10}). For a two-dimensional test reported Figure 7 in Moulin et al. [23] using 14828 triangles with $\mathbb{P}_2/\mathbb{P}_1$ elements (roughly 37000 degrees of freedom) and threshold 10^{-3}, we get around 4ms per degree of freedoms (which would give 13ms with the threshold 10^{-10}). Using the three-dimensional test reported in Figure 9 therein with 75×10^6 degrees of freedom and threshold 10^{-6}, over 2048 processors, we obtain 20ms.

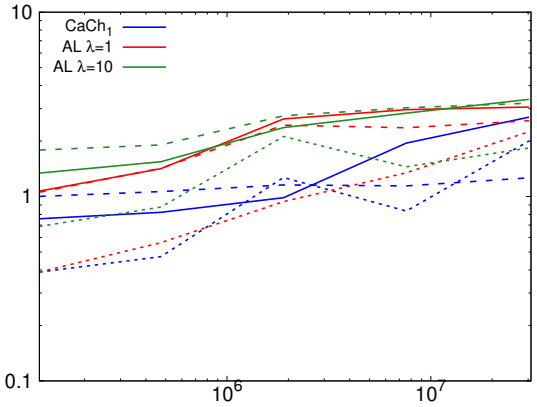


Fig. 4: . Y-axis: $\frac{\text{wall-clock time}\times\text{nb. processor}}{\text{nb. vel.+press. degrees of freedom}}$ in millisecond; X-axis: number of velocity grid points. $\mu = 1$ (——); $\mu = 10^{-2}$ (- - -), $\mu = 10^{-4}$ (·····).

6. Conclusion. We confirmed in the paper that the preconditioned augmented Lagrangian method requires less GMRES iterations than the traditional Cahouet&Chabard method. However, too much time is spent solving the velocity problem in the augmented Lagrangian method. The main difficulty consists of inverting the discrete version of the grad-div operator. Overall, the wall-clock efficiency of the augmented Lagrangian method is of the same order as that of the Cahouet&Chabard method.

For the CahouetCahouet&Chabard preconditioning, we showed that, using the discrete pressure Laplacian instead of the full matrix $B\ M_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ gives a methods that converges slowly on fine meshes and when the thresholds is stringent. Hence, we recommend using $B\ M_{\boldsymbol{V}}^{-1}B^{\mathsf{T}}$ in the preconditioner even if this means solving another linear problem using another iterative method.

From a numerical perspective, properly using an algebraic multigrid solver like BoomerAMG yields very good parallel scaling and makes it possible to solve the problem on very fine meshes with no memory problem (for instance, 120 millions velocity grip points with 700 processors). In contrast, direct linear solvers are more performant on coarser meshes (under 2 millions velocity grid points).

Finally, the results about the time efficiency reported in this paper show that solving the time-dependent Stokes and Navier-Stokes equations by means of a pressure Schur complement technique does not yet realistically compete with pressure- and velocity-correction methods, as the time efficiency for projection methods is usually of the order of 10^{-2} ms per degree of freedom per time step with the current hardware technology. It might be possible to fine tune the preconditioned augmented Lagrangian method to reach 0.1ms per degree of freedom per time step, but gaining two orders of magnitude seems unfortunately problematic at the moment without a genuine algorithmic breakthrough. Of course one can argue that Schur complement techniques are more accurate in time than projection methods as they do not induce

any time splitting error. Moreover, they can be used with larger time steps as they can be implemented with unconditionally stable time stepping technique.

References.

[1] P. R. Amestoy, A. Guermouche, J.-Y. L'Excellent, and S. Pralet. Hybrid scheduling for the parallel solution of linear systems. *Parallel Computing*, 32(2):136–156, 2006.

[2] S. Balay, W. D. Gropp, L. C. McInnes, and B. F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[3] M. Benzi and M. A. Olshanskii. An augmented Lagrangian-based approach to the Oseen problem. *SIAM J. Sci. Comput.*, 28(6):2095–2113, 2006.

[4] M. Benzi and Z. Wang. Analysis of augmented lagrangian-based preconditioners for the steady incompressible navier–stokes equations. *SIAM Journal on Scientific Computing*, 33(5):2761–2784, 2011. doi: 10.1137/100797989. URL https://doi.org/10.1137/100797989.

[5] M. Benzi, M. A. Olshanskii, and Z. Wang. Modified augmented lagrangian preconditioners for the incompressible navier–stokes equations. *International Journal for Numerical Methods in Fluids*, 66(4):486–508, 2011. doi: https://doi.org/10.1002/fld.2267. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.2267.

[6] D. Boffi, F. Brezzi, and M. Fortin. *Mixed finite element methods and applications*, volume 44 of *Springer Series in Computational Mathematics*. Springer, Heidelberg, 2013.

[7] J. H. Bramble and J. E. Pasciak. Iterative techniques for time dependent Stokes problems. volume 33, pages 13–30. 1997. Approximation theory and applications.

[8] J. Cahouet and J.-P. Chabard. Some fast 3D finite element solvers for the generalized Stokes problem. *Internat. J. Numer. Methods Fluids*, 8(8):869–895, 1988.

[9] A. J. Chorin. Numerical solution of the Navier–Stokes equations. *Math. Comp.*, 22:745–762, 1968.

[10] J. de Frutos, B. García-Archilla, V. John, and J. Novo. Grad-div stabilization for the evolutionary Oseen problem with inf-sup stable finite elements. *J. Sci. Comput.*, 66(3):991–1024, 2016.

[11] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, 2005.

[12] A. Ern and J.-L. Guermond. *Finite elements II—Galerkin approximation, elliptic and mixed PDEs*, volume 73 of *Texts in Applied Mathematics*. Springer, Cham, 2021.

[13] P. E. Farrell, L. Mitchell, and F. Wechsung. An augmented lagrangian preconditioner for the 3d stationary incompressible navier–stokes equations at high reynolds number. *SIAM Journal on Scientific Computing*, 41(5):A3073–A3096, jan 2019. doi: 10.1137/18m1219370. URL https://doi.org/10.1137%2F18m1219370.

[14] J. A. Fiordilino, W. Layton, and Y. Rong. An efficient and modular grad-div stabilization. *Comput. Methods Appl. Mech. Engrg.*, 335:327–346, 2018.

[15] M. Fortin and R. Glowinski. *Augmented Lagrangian methods*, volume 15 of *Studies in Mathematics and its Applications*. North-Holland Publishing Co., Amster-

dam, 1983. Applications to the numerical solution of boundary value problems, Translated from the French by B. Hunt and D. C. Spicer.

[16] V. Girault and P.-A. Raviart. *Finite element methods for Navier–Stokes equations. Theory and algorithms*. Springer Series in Computational Mathematics. Springer-Verlag, Berlin, Germany, 1986.

[17] G. H. Golub and C. Greif. On solving block-structured indefinite linear systems. *SIAM J. Sci. Comput.*, 24(6):2076–2092, 2003.

[18] J.-L. Guermond and P. Minev. High-order time stepping for the incompressible Navier-Stokes equations. *SIAM J. Sci. Comput.*, 37(6):A2656–A2681, 2015. ISSN 1064-8275. doi: 10.1137/140975231. URL https://doi.org/10.1137/140975231.

[19] J. L. Guermond, P. Minev, and J. Shen. An overview of projection methods for incompressible flows. *Comput. Methods Appl. Mech. Engrg.*, 195(44-47):6011–6045, 2006.

[20] T. Heister and G. Rapin. Efficient augmented Lagrangian-type preconditioning for the Oseen problem using grad-div stabilization. *Internat. J. Numer. Methods Fluids*, 71(1):118–134, 2013. ISSN 0271-2091. doi: 10.1002/fld.3654. URL https://doi.org/10.1002/fld.3654.

[21] V. E. Henson and U. M. Yang. BoomerAMG: a parallel algebraic multigrid solver and preconditioner. volume 41, pages 155–177. 2002. Developments and trends in iterative methods for large systems of equations—in memoriam Rüdiger Weiss (Lausanne, 2000).

[22] E. W. Jenkins, V. John, A. Linke, and L. G. Rebholz. On the parameter choice in grad-div stabilization for the Stokes equations. *Adv. Comput. Math.*, 40(2):491–516, 2014.

[23] J. Moulin, P. Jolivet, and O. Marquet. Augmented lagrangian preconditioner for large-scale hydrodynamic stability analysis. *Computer Methods in Applied Mechanics and Engineering*, 351:718–743, 2019. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2019.03.052. URL https://www.sciencedirect.com/science/article/pii/S0045782519301914.

[24] M. Olshanskii, G. Lube, T. Heister, and J. Löwe. Grad-div stabilization and subgrid pressure models for the incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Engrg.*, 198(49-52):3975–3988, 2009.

[25] L. R. Scott and M. Vogelius. Norm estimates for a maximal right inverse of the divergence operator in spaces of piecewise polynomials. *RAIRO Modél. Math. Anal. Numér.*, 19(1):111–143, 1985.

[26] Y.-h. Shih, G. Stadler, and F. Wechsung. Robust multigrid techniques for augmented Lagrangian preconditioning of incompressible Stokes equations with extreme viscosity variations. *SIAM J. Sci. Comput.*, 45(3):S27–S53, 2023. ISSN 1064-8275. doi: 10.1137/21M1430698. URL https://doi.org/10.1137/21M1430698.

[27] R. Temam. Sur l'approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires ii. *Arch. Rat. Mech. Anal.*, 33:377–385, 1969.

3 - Post-processing tools

The SFEManS code by itself only gives access to the velocity field u, the pressure field p, and the magnetic field H. Hence, I had to implement new tools to study the energy transfers between the velocity and magnetic fields. First, the various quantities we are interested in (see 4) need to be calculated in SFEMaNS. Afterward, they have to be transformed into a usable file format for either visualisation or statistical study. Finally, as we are dealing with a great amount of data, all visualisation and analysis have to be automatized. This automation was done using Python and was coupled with Paraview for visualisation.

In order to easily transfer and use those tools, they are available for our research group in our private GitLab as different projects. A README.md file is included in each of them with a simple tutorial. The goal of this chapter is not to make a complete documentation of these tools but to present key points of each step of the workflow.

3.1 . Post-processing calculation in SFEMaNS

While the time simulation in SFEMANS is done, only the magnetic, velocity and pressure fields are stored. However, further computation is needed to get the different energy transfers happening in our system. In order to keep a high accuracy and efficiency for the transfer terms, computations of these new terms are still done using the SFEMaNS framework. The corresponding project is called 'DR_MHD' in GitLab. For ease of use, all new options needed are written in a 'data_example' file that just needs to be copy-pasted into the 'data' file used for the temporal simulation. There is no need to modify the various options previously used for the temporal simulation.

3.1.1 . Filtering process

In order to analyze what happens at different scales, we need to differentiate between large and small structures. This is important in order to find rare events in the flow that could indicate possible singularities. It also, make it possible to study how the energy transfers from one structure to another and hence find where and when important events for dynamo and turbulence happens. To do this separation, we use a filter. It is the most important part of the computation and the most costly in resources in terms of memory and computation cost.

The SFEMaNS grid is unstructured and is build to solve linear differential

equations easily. Hence we define the filtered field $g = \overline{u}^{\ell}$ as the solution of:

$$g - \ell^2 \Delta g = u, \tag{3.1}$$

with u a vector or scalar field and $\ell \in \mathbb{R}^+$.

The filtering process is then transformed into a linear problem using the same method as for the evolution of the temperature (see section 2.1.2). Equation (3.1) becomes for the Fourier mode m of a cosine component of a scalar or vector field:

$$A_{ij}^{m,\mathsf{c}} g_j^{m,\mathsf{c}} = u_i^{m,\mathsf{c}}, \tag{3.2}$$

with:

$$A_{ij}^{m,\mathsf{c}} = \int_{r,z} r(\psi_j \psi_i + \ell^2 (\partial_r \psi_j \partial_r \psi_i + \partial_z \psi_j \partial_z \psi_i + \frac{m^2}{r^2} \psi_j \psi_i)). \tag{3.3}$$

As it takes a lot of time to build this matrix to solve the system, one matrix is saved in memory for each scale ℓ. However, it is a simple way to construct a filter that can be computed on unstructured meshes efficiently by using the powerful tools of linear algebra.

This filter was already programmed by Hugues Faller [60] but a few modifications have since been done. Firstly, the same filter has been duplicated for the magnetic mesh as it only existed for the velocity mesh. This at least doubles the usage of memory for saving the matrices. Secondly, the boundary conditions have been changed from Dirichlet's to Neumann's in order to correct a problem that had occurred for $\ell = 0$. Thirdly, the filter has been encapsulated in its own function name 'filter' for ease of use.

Currently, in order to choose at which scale to perform the calculation, a list of floats is asked from the user. Those floats are then multiplied by $h_{\max}$, the size of the biggest cell in the mesh. This makes it difficult to specify filters at characteristic lengths of the flow (Kolmogorov's scale for example). Modifications are underway to make it more practical.

3.1.2 . Computation of the various energy transfers

The biggest part of the implementation was the computation of each individual term of energy transfer. As it is only a succession of scalar and vector products and computations of a few derivatives, most of the framework was already in place. One of the difficulties was that some fields are defined on the magnetic mesh while others are on the velocity mesh. Hence, a few functions needed to be revamped in order to work with both meshes. As calculating all the energy transfers is costly, particular attention has been put toward memory usage in order to save computation time and memory space.

In order to minimize the cost of operations (filtering, deriving fields, and so on), simple optimization strategies are used, other than parallelization. When calculating different transfers, the quantities that are independent of the scale

ℓ are kept in memory. This is important when calculating the energy transfers for different scales as it reduces the number of operations done. As an example: $\nabla \times \frac{1}{\sigma}$, $\nabla \times b$, $j \times b$, and others are stored when doing the calculation for the first scale ℓ so that they can be used for all other scales. However, storing quantities puts a strain on computer memory and can reduce performance. As such, when possible, generic tables for vectors and scalars are used. The aim is to minimize the number of these storages by cleverly designing the order of all operations. To make it practical to follow, a file has been created (named 'memory_usage.ods') that tracks which table contains which quantity at each operation step. Figure 3.1 shows the way this usage was planned for the scalar tables on the velocity mesh.

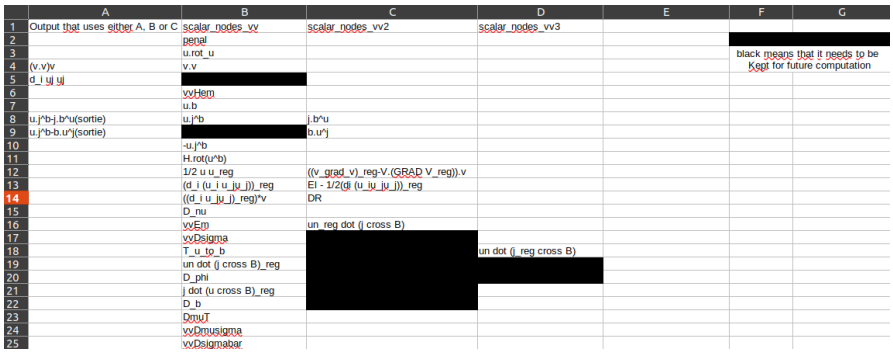


Figure 3.1: Quantities saved in the different Fortran scalar tables during computation. The order has been optimized for both computation cost and memory usage.

To illustrate the practical implementation of the calculation, here is the order of operations for the calculation of two main terms:

$$\mathcal{T}^{u\to b} = \frac{1}{4}(\overline{u}^{\ell} \cdot (b \times j) + u \cdot (\overline{b \times j}^{\ell}) + (\overline{u \times b}^{\ell}) \cdot j + (u \times b) \cdot \overline{j}^{\ell}), \tag{3.4}$$

$$\mathcal{D}^{b} = \frac{1}{4}(\overline{u}^{\ell} \cdot (b \times j) + u \cdot (\overline{b \times j}^{\ell}) - (\overline{u \times b}^{\ell}) \cdot j - (u \times b) \cdot \overline{j}^{\ell}). \tag{3.5}$$

The fields recorded from a previous simulation are u the velocity field, b the magnetic flux (or induction), and H the magnetic field on the nodes.

- Operations only done once:
  - Memory allocation for all used structures.
  - Create ghost vectors and scalar structures for PETSC parallelization.
  - Initialize MPI parallelization for the Fourier modes.
  - Assemble matrices for transfer between nodes and Gauss points on both the magnetic meshgrid and the velocity one.

  - Calculation of the radius of the Gauss points ($r(l)$) on both meshgrids.

- For the first scale ℓ:
  - Send b from the magnetic meshgrid to the velocity meshgrid and save it.
  - Send H on the Gauss points and calculate $j = \nabla \times H$ on these points.
  - Transfer j from the Gauss points to the nodes and save it.
  - Calculate $j \times b$ and $u \times b$ using fast Fourier transform and save it.
  - Send $j \times b$ from the magnetic meshgrid to the velocity meshgrid and save it.

- For all scales ℓ:
  - Prepare the matrix to solve the filtering problem for the scale ℓ and save it.
  - Calculate $\overline{u}^\ell$ in a generic vector n°1 (it will be used to compute other quantities before being used in $\mathcal{T}^{u\to b}$).
  - Calculate $\overline{j}^\ell$ on the magnetic meshgrid and send it on the velocity meshgrid in a generic vector n°2.
  - Calculate $\overline{u}^\ell \cdot (b \times j)$ and $(u \times b) \cdot \overline{j}^\ell$ and store them in generic scalars n°1 and n°2.
  - Calculate $\overline{b \times j}^\ell$ and store it in the generic vector n°1 as $\overline{u}^\ell$ will not be used anymore.
  - Calculate $u \cdot (\overline{b \times j}^\ell)$ and store them a generic scalar n°3.
  - Sum generic scalars n°1 and n°3 into n°1 to get $\overline{u}^\ell \cdot (b \times j) + u \cdot (\overline{b \times j}^\ell)$.
  - Calculate $\overline{u \times b}^\ell$ and store it in the generic vector n°1.
  - Calculate $(\overline{u \times b}^\ell) \cdot j$ and store them a generic scalar n°3.
  - Sum generic scalars n°2 and n°3 into n°2 to get $(\overline{u \times b}^\ell) \cdot j + (u \times b) \cdot \overline{j}^\ell$.
  - Sum generic scalars n°1 and n°2 and multiply by $\frac{1}{4}$ into generic scalar n°3 to get $\mathcal{T}^{u\to b}$.
  - Subtract generic scalars n°1 and n°2 and multiplying by $\frac{1}{4}$ into generic scalar n°3 to get $\mathcal{D}^b$.

For simplicity's sake, it might not match the numbers in 'memory_usage.ods' (Fig. 3.1) as the savings on the magnetic mesh are not discussed here.

3.1.3 . Output files

The computed fields are saved in the SFEMaNS format which makes it easily readable by SFEMaNS. The naming convention is as follows:

```
suite_${field_name}_nDR${filter_number}_S${domain_number}_I${iteration_number}.$${mesh_name}.FEM
```

where 'field_name' is the name of the field (for example 'Ec' for the kinetic energy), 'filter_number' is associated to the scale's number for the filter, 'domain_number' is the number associated to the processor in the meridian plane, 'iteration_number' is the temporal iteration and 'mesh_name' is the name of the SFEMaNS mesh used for the temporal simulation. Fields that do not have filtering (as for example the helicity $\mathcal{H}$) have the filtering number set to 1.

The last output file is named 'fort.75' and contains all the spatial averages of all fields for each iteration number. It also has additional information such as the simulation time at each iteration number and the scale of filtering. A description of 'fort.75' is written in 'means info' (Fig. 3.2) which tells which column corresponds to which quantity.

```
1->20 independant of filtering
1:time
2:divu
3:divb
4:Helicity
5:magnetic helicity
6:Omega effect z
7:Omega effect z * r
8:Omega effect r * r
9 -> 17: Alpha effect

1->20 + 20 * filter_number for filter dependant quantities
1:filter_number
2:hmax (such that l = filter_number * hmax)
3:Ec
4:Du
5:DR
6:Dnu

11:Em
12:Mmu
13:Tub
14:Dsigma
15:Db
16:DmuT
17:Dmusigma
18:Dsigmabar
19:lambda_H
```

Figure 3.2: 'means info' file which references which columns contain which quantities in the 'fort.75' file.

3.1.4 . Problem with chaining products

Simple tests have been put in place in order to verify that all fields have been correctly computed. For example, for $\ell = 0$, a few terms are expected to be equal to zero. This works well for all except for $\mathcal{D}^b$ (Fig. 3.3). The cause is the product of fields.

Indeed, as the fields are decomposed in Fourier space in the azimuthal direction, it is necessary to project back into the physical space for computing products. However, if not enough Fourier modes are used when projecting, aliasing happens. Indeed, the product of two vector having $N \in \mathbb{N}^+$ Fourier modes would contain Fourier modes up to $2N$ ($\cos(N\theta) * \cos(N\theta) = \frac{1}{2}(\cos(2N\theta)+1)$). Hence, if not enough Fourier modes are used in the resulting vector, the modes for $n > N$ would pollute the ones for $n < N$. This is already taken into account in SFEMaNS: anti-aliasing is done by using $N + N/2$ modes when projecting. Still, this is not enough for D_b as it contains two products. As such, it creates Fourier modes up to $3N$ and would require anti-aliasing to $2N$ when projecting. As simple as it sounds, it is not currently possible to do so because it requires the three fields used in the products to be projected at the same time in the physical space. Hence, there is currently an error of a few percent on the terms requiring double products of fields (Fig. 3.3).

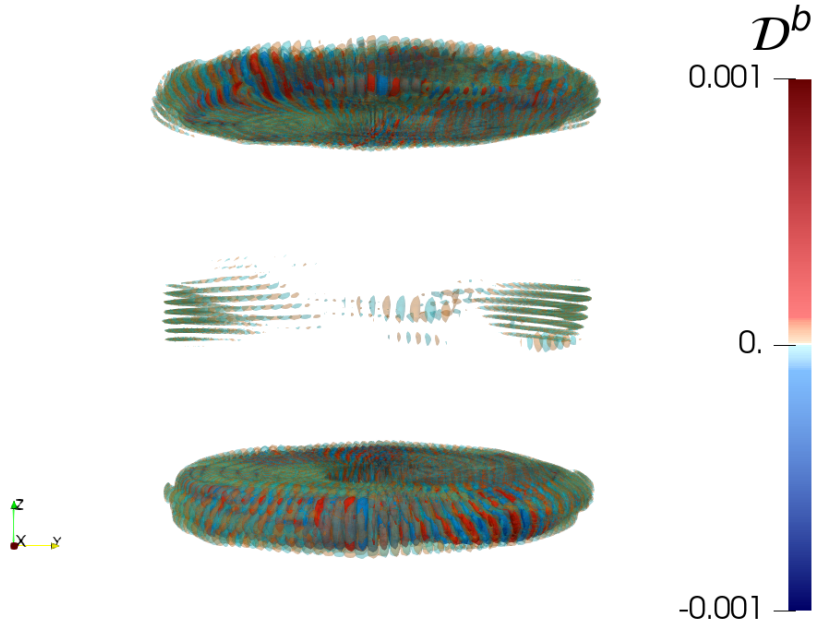


Figure 3.3: Example of the error made on $\mathcal{D}^b$. Here $\mathcal{D}^b$ should be equal to zero everywhere as $\ell = 0$.

3.2 . Transfer SFEMaNS fields to usable file formats

The files used by SFEMaNS are of a specific format that contains additional information in order to restart the temporal simulation. Moreover, all Fourier modes are saved in a complex manner depending on the mesh structure and its repartition on the processors. To make those files usable for statistical analysis and visualisation, a simple code has been developed to change the formatting of those files. Again those computations are done inside SFEMaNS and the project name is suites_to_bins in GitLab.

This project takes all the temporal iterations previously computed and outputs either **binary files for statistical analysis** using Python or **pvd/vtu files for visualisation** with Paraview. Once again, it is only needed to add the new options from 'data_example' into the data file used for the simulation and select the wanted options.

3.2.1 . Output file formats and name convention

The pvd/vtu files created for visualisation are named as follows:P2/P3 finite elements of

```
${field_name}_${iteration_number}_${filtering_number}$
```

The name for the binary files needs more specifications. Indeed, these files can be the representation of a field either in the Fourier or in the physical space. They are saved in a new directory called 'binaries_out'. For each field and each filter, a subdirectory is created with the name:

```
D${filter_number}_${field_name}
```

Depending if they are written in the physical or the Fourier space, the files will then have different names:

```
phys_${field_name}_S${domain_number}_I${iteration_number}
fourier_${field_name}_S${domain_number}_I${iteration_number}
```

For vector fields, a number is added at the end of the field's name according to which component is written: (1, 2, 3) for (r. θ, z). For Fourier modes, a c or s is specified if the modes in the files are the cosine or sine ones. The domain's number is kept for simplicity's sake: it would take a lot more time to write the files on the total domain since all processors of each domain would wait for each other to write.

The files are structured such that the mesh points are in the same order in all of them. Inside the files, the points are written such as the 2D domain associated to the processor is repeated for each angle (or Fourier mode) and

ordered in increasing order. This makes it possible to save the (r, z) positions of the points only once, the angles being deduced by knowing the number of Fourier modes used. The (r, z) positions as well as the statistical weight of each point are saved in files named:

```
${mesh}rr_S${domain_number}
${mesh}zz_S${domain_number}
${mesh}weight_S${domain_number}
```

where 'mesh' is the type of the mesh. As this code is currently only implemented for magnetohydrodynamics and we do not study the pressure field, it is either 'vv' for the velocity mesh or 'H' for the magnetic one.

Again, fields that do not have filtering have the filtering number set to 1.

3.2.2 . Temporal means and renormalization

Two other interesting features present in this code are the possibility to create a temporal average of a field or to renormalize a field in time. The renormalization is done by dividing a field by the average spatial value of the magnetic energy. It is very useful in order to study the mode responsible for the exponential growth of the magnetic field (see Chapter 4 section 5.1 of the article).

Making a temporal average will create a new field with an added 'mean' in front of its name and give it the iteration number '0'. The renormalization will just create a new temporal field with 'renorm' before the field's name. By combining both, it is possible to get the time average of a renormalized field whose new name will start with 'meanrenorm'.

3.2.3 . Selecting the fields to be processed

SFEMaNS post-processing outputs a lot of different fields and sometimes not all need to be visualised. As such, a dedicated file needs to be filled in to select the fields to study. This file is called 'data_for_suites' and, on each line, it contains:

```
${field_name} ${bool_1} ${bool_2} ${bool_3}
```

See an example in Fig. 3.4.

```
'Du' .t. .f. .t.
'Dsigma' .f. .f. .t.
'Tub' .t. .f. .t.
'He' .t. .f. .f.
```

Figure 3.4: Example of 'data_for_suites' file in order to select which fields to process for visualisation or statistical analysis.

Each boolean is either '.t.' or '.f.', the first is true if the field is on the velocity mesh, the second if the field is a vector, and the third if the field is filter dependent. The code will then only process the fields present in that file. Note that, if some renormalization or temporal average is done, 'field_name' needs to have 'renorm' or 'mean' in front.

3.3 . Visualisation

Visualisation of data is a key point for the analysis of physical phenomena by highlighting where and when they take place. All that process of transforming the data into usable pictures has been automatized because of the large quantity of fields and iterations to process. For that, we take advantage of Python's ability to run within Paraview, allowing it to operate without its graphical user interface (GUI). The fields need to be written in the pvd/vtu format in order to be read by Paraview. The project name in GitLab is paraview_python. The Paraview version used is 5.8.0. There can be problems with some other versions as Paraview does not guarantee that the names of their Python functions are kept the same.

3.3.1 . Types of visualisation

As SFEMaNS is in cylindrical coordinates, three types of visualisation have been implemented (Fig. 3.5):

- 'animation_translation': cross sections at varying height z,
- 'animation_rotation': cross sections at varying angle θ,
- 'animation_DDD': isosurfaces in 3D.

As these animations only work for scalar fields, a desirable component needs to be specified for vector fields in 'component' (see the function call below). Also, it takes a Paraview source as an entry to avoid loading in memory multiple times the same field for different animations. As an example, here is one of the function's definition:

```
def animation_rotation(source, field, p, screen_path,
    screen_name, nb_frames, component='', id_colormap=0,
    resolution=None, opt_field_tex='', opt_contour='',
    opt_view_norm=1., opt_blade_source=None)
```

The necessary inputs are the Paraview source 'source', the name of the corresponding field 'field', 'p' which sets the maxima and minima of the color scale, the directory path for the pictures 'screen_path', the name for the output pictures 'screen_name' and the number of angles (resp. heights) designated by 'nb_frames' at which visualisations are performed for rotation/3D (resp. translation). There are a lot of other options. The main ones will be explained in the following as others are here for higher-level automation.

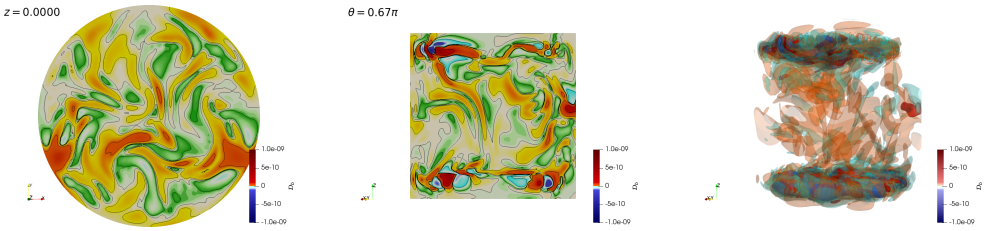


Figure 3.5: Example of the three types of visualisations with bi-logarithmic scale and isosurfaces.

The camera distance to the field is automatically rescaled in order to fit all of it into the frame by checking its maximal and minimal coordinates. However, it can still be scaled using 'opt_view_norm'. It is very useful when fields for the same simulation are on meshes of different dimensions. For example, in our case, the velocity mesh has a radial extension $r \in [0, 1]$ whereas the magnetic mesh has $r \in [0, 1.6]$. By setting 'opt_view_norm' to 1.6 for fields that are on the magnetic mesh we can easily compare all quantities (see for example Fig. 3.6).

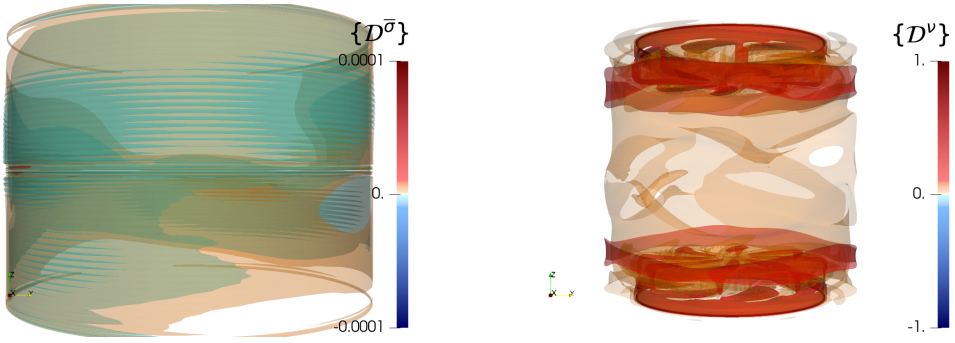


Figure 3.6: Example of how camera rescaling is used to easily compare quantities on two meshes with different dimensions.

By default the color scale is set as bi-logarithmic as the values of the studied fields greatly vary between the bulk and the turbines. Its maximal and minimal values are set by the parameter 'p' which sets them to $\pm 10^p$. Other types of color scales are available and can be set using 'id_colormap'. Creating a new one requires coding its values in 'color_and_opacity_maps.py'. For example, the bi-logarithmic is coded as:

```
    [-10 ** p, 0.0, 0.0, 0.4,
    -10 ** (p - 1), 0., .5, 1.,
    -10 ** (p - 2), 0.0, .9, 1.0,
    -10 ** (p - 3), 0.0, 1., .9,
    0.0, 1.0, 1.0, 1.0,
    10 ** (p - 3), 1.0, 1.0, 0.0,
    10 ** (p - 2), 1.0, 0.5, 0.0,
    10 ** (p - 1), 1., 0.0, 0.0,
    10 ** p, 0.4, 0.0, 0.0]
```

where each line corresponds to a new color being set at a specific value of the field. The same goes for opacity scales which are used for 3D isosurfaces.

Isosurfaces are created for all of them. By default, their values depend on the selected color scale (a logarithmic color scale will make logarithmic isosurfaces). Others can be set using 'opt_contour' and new ones created in 'contour_definitions.py'. Here are all the ones currently present:

```
contours_values = {
    'None': [],
    'positive_linear_tenth': [1., .9, .8, .7, .6, .5, .4,
                              .3, .2, .1],
    'linear_tenth': [1., .9, .8, .7, .6, .5, .4, .3, .2, .
                             1, 0, -.1, -.2, -.3, -.3,
                              -.4, -.5, -.6, -.7, -.8,
                              -.9, -1.],
    'positive_linear_ddd': [1., .9, .75, .5],
    'linear_ddd': [1., .9, .75, .5, -.5, -.75, -.9, -1.],
    'positive_log_10': [1, .1, .01, .001],
    'log_10': [1, .1, .01, .001, 0, -1, -.1, -.01, -.001],
    'positive_log_10_ddd': [1, .1, .01],
    'log_10_ddd': [1, .1, .01, -1, -.1, -.01],
    '3waygreater': [0.9, -0.9]
}
```

The ones ending with 'ddd' are used for 3D visualisations as too many 3D isosurfaces make the pictures too bloated.

One last important option, specific to the translation animation, is 'opt_z' which sets the various heights at which the visualisation will be done. It is very practical in order to study what takes place just over and under the blades and impellers.

Specific animations have been developed for vector fields ('animation_translation_compact' and 'animation_rotation_compact'). They do the same thing as the previous

ones but with the color scale being used for the perpendicular coordinate to the plane and arrows for the in-plane components. However, they did not get much use during the thesis as all energy transfers are scalar fields.

3.3.2 . Automation of visualisations

We have defined the visualisation functions. We now need to prepare a Paraview source. This is done to avoid loading the field multiple times and repeating computations on that field. For example, animations for vector fields need the computation of perpendicular components (r, θ). This is done in the 'make_animations' function in 'make_animations.py' which definition is:

```
def make_animations(file_names, data_directory,
    screen_directory, nb_frames, rot=True,
    tr=True, DDD=True, rot_cp=False, tr_cp=False,
    rot_frames=-1, tr_frames=-1, DDD_frames=-1,
    rot_cp_frames=-1, tr_cp_frames=-1, opt_vect=False,
    ids_colormap=None, calculator_name='', opt_contour='',
    opt_p=None, opt_blade_file='', opt_tr_z=None)
```

For basic usage, this function only needs the file path of the field to visualize (file name in 'file_names' and directory path in 'directory_path'). The field's characteristics need to be added beforehand to 'fields_dictionnaries.py' which requires its name in Tex and booleans to specify if it is vector, if it is on the magnetic mesh, and if it depends on the filtering. A plethora of parameters makes it possible to select different visualisations, the number of frames for each, the colormap, the number of isosurfaces, and so on.

However, further computation and combination of fields can also be done by giving multiple fields in 'file_names' and by indicating the computation to be done via 'computation_name'. Calculators are defined in 'calculators_dictionnaries.py' and require a lot of different pieces of information: their input and output type (vector (2), scalar (1) or same as input (0)), how to construct the output field's name in both Paraview and Tex as well as how the calculator is written in Paraview. Here is an example of the pieces of information for calculating the z component of a field:

```
    calculators_intype = {'X': [2]}
    calculators_outtype = {'X': 1}
    calculators_name_constructor = {'X': ['', 0, 'X']}
    calculators_calc_constructor = {'X': ['', 0, '_X']}
    calculators_tex_constructor = {'X': [r'$', 0, r'_x$']}
```

For the ones ending with 'constructor', the number 0 will be filled by the name of the field automatically at computation time. Sometimes, those calculators require fields from different meshes and thus a projection from the magnetic mesh to the velocity mesh has also been automatized in those cases.

Still, this function only does one field at a time. To visualise all fields in a directory, the function 'make_all_pvd_fields' is used. It is very similar to

'make_animations' but it only needs the directory's path containing all the fields as an input. Once again, various options make it possible to select the visualisations and their parameters. It is also possible to set the extrema of the colormap for each field via 'opt_ps' which is very useful when we do not want jumps in colorscale through multiple temporal iterations. An example of its usage is given in the project 'main.py' which is simplified below:

```
from meta_func import *

# directory where to put the .pvd and .vtu files
data_directory = 'data_para'
# directory where to save the pictures
screen_directory = 'screen_para'
# number of frames wanted for each animation
nb_frames = 100
make_all_pvd_fields(data_directory, screen_directory,
                                nb_frames, tr=False, rot=
                                False, DDD=True)
```

This code will output the 3D animations (for 100 angles of view) for all fields present in 'data_para' and write the pictures in the directory 'screen_para'.

As there can be a lot of fields to process, the computation can time out on supercomputer nodes by exceeding the time limit. In that case, an integer is written in the program's output file for each field done. The paths of those files need to be given as an argument when running this Python code to avoid doing the same field twice (see 'list' in the next section).

3.3.3 . Parallelization

The most efficient way to run this code on multiple processors is to give one field to visualize to each processor. This is done by specifying the rank and total number of processors in the first and second arguments of the Python call. An example of a complete command to run the code on multiple processors is given in 'job_para.slurm' in the GitLab project as well as simplified below:

```
nb_tasks=20 #number of paraview to run in parallel
nb_proc_per_task=2 #number of processeur given to each
                                paraview
for ((i = 0 ; i < nb_tasks ; i++)); do
srun -n1 --exclusive -c $nb_proc_per_task --unbuffered
pvbatch --mpi  --force-offscreen-rendering
"$PARAVIEW_POST_PRO/main.py" "$i" $nb_tasks
"$list" &
done$
```

where 'nb_tasks' is the number of visualisation run in parallel and 'list' is the list of the previous output files. However, when using that kind of parallelization, one processor might not have enough RAM memory for a field, hence multiple

processors need to be allocated to one Python call. This is done by using 'nb_proc_per_task'.

3.3.4 . Accessing the pictures

The quantity of pictures produced using the post-processing code can be very important. In our case, a run with around 20 different scalar fields on 120 iteration numbers with 3 different filters will output tens of thousands of frames only for 3D isosurfaces. In order to avoid transferring that huge amount of data (around 500G), a simple GUI has been implemented in Python. By just launching the code in the project 'image_viewer_sfemans' and giving it the path to the repository containing the pictures, it is possible to glimpse through all fields, iteration numbers, and filters (see Fig. 3.7).

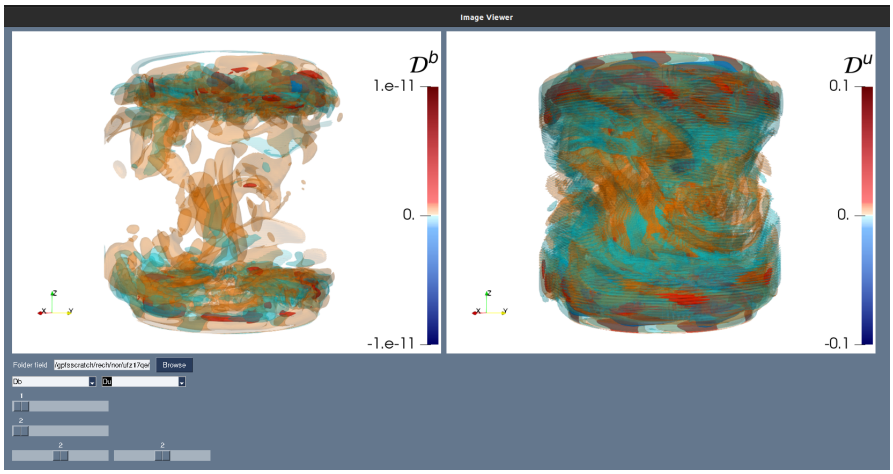


Figure 3.7: Visualisation window using the image viewer for some scalar fields.

3.4 . Outlook

It was essential to develop these different programs so as to have an ironed-out flowchart from time simulation to final visualisation. They were also important to quickly test and view any modification implemented to the filter and to the computation of our various fields. It also made it very easy and fast to produce data for various internships.

Statistical analysis tools in Python have also been implemented early in the thesis but have since been completely revamped by Nikita Allaglo (L3 intern, spring 2023) and Rémi Bousquet (M2 intern, spring 2023). They have not since been used to do analysis for the subject of this thesis.

There is still room for a lot of improvements to this workflow. The main one is being able to run an instance of Paraview on multiple processors at once. This would help when multiple processors are allocated to one field to avoid memory overflow issues.

4 - Energy transfers in the von Kármán Sodium experiment

This chapter is dedicated to the analysis of the energy transfers in SFEMaNS simulations of the von Kárman Sodium experiment. It presents itself in the form of an article currently submitted to Physics of Plasmas (August 2023). It contains the analytical demonstrations and equations for the energy transfers between fields and between scales for the magnetohydrodynamics equations for varying magnetic permeability and electrical conductivity. These energy transfers are then analyzed using two numerical simulations which generate a magnetic field but with different materials for the impellers (steel and soft iron). The conclusion is that, depending on the material of the impellers, the dynamo mechanisms have very different origins as well as spatial locations.

Tracking dynamo mechanisms from local energy transfers: application to the von Kármán sodium dynamo

M. Creff, H. Faller, B. Dubrulle, J.-L. Guermond, C. Nore.

Abstract Motivated by the observation that dynamo is a conversion mechanism between magnetic and kinetic energy, we develop a new approach to unravel dynamo mechanism based on local (in space, scale, and time) energy budget describing dissipation and scale-by-scale energy transfers. Our approach is based upon a new filtering approach that can be used effectively for any type of meshes, including unstructured ones. The corresponding formalism is very general and applies to any geometry or boundary conditions. We further discuss the interpretation of these energy transfers in the context of fast dynamo and anomalous dissipation. We apply it to results from direct numerical simulations of the Von Kármán Sodium setup using a finite element code, showing dynamo action for two types of impellers (steel or soft iron) in the magnetic field growth and saturation phases. Although the two types of dynamo hardly differ from the mean-field theory point of view (the velocity fields are the same in both cases), the locality of our formalism allows us to trace the origin of the differences between these two types of dynamo: for steel impellers, the dynamo is due to the transfer of velocity energy both in the bulk and in the vicinity of the impellers, whereas for soft iron impellers, the dynamo effect mainly comes from the rotation of the blades. We finally discuss possible signatures of precursors to anomalous dissipation and fast dynamo, that could become relevant in the inviscid limit.

Contents

1 Introduction

In many natural systems, such as the Earth or the Sun, a large-scale magnetic field can be generated by the so-called dynamo process. Such a process is described via the magnetohydrodynamics (MHD) equations coupling the equations modeling incompressible viscous fluids (Navier-Stokes equations) with the equations modeling electromagnetism (Maxwell equations in the quasi-static approximation). Given the non-linearities of the equations and the enormous range of scales involved in the process, it is very difficult to identify the main mechanisms allowing the generation of a magnetic field by the motion of an electrically conducting fluid. From a theoretical point of view, exact analytical results are generally only obtained in very simple geometries or use symmetries to produce anti-dynamo conditions. Simplified models of dynamo generation can be built using mean field or stochastic approaches [1, 2]. These approaches highlight the importance of helicity (via the so-called α-effect) and differential rotation (via the so-called Ω-effect) in the dynamo process. Such mean field approach provided several possible dynamo scenarios, depending on which combination is chosen [2]. However, confronting such simple scenarios with numerical or experimental observation turns out to be very difficult. From a numerical point of view, simulations of MHD equations are only possible for a limited range of parameters, excluding the very turbulent situations observed in the liquid core of the Earth or in the plasma making up stars like the Sun. Moreover, the large amount of data and information makes it difficult to identify the relevant mechanisms at work in the dynamo action. From an experimental point of view, the situation is the opposite: using liquid metal experiments, it is possible to reach very turbulent regimes. However, owing to the difficulty of measurements, it is only possible to produce sparse data that are not detailed enough to enable the identification of dynamo mechanisms, except in some special situations where the geometry is very constrained and the flow of liquid sodium reproduces analytical examples [3, 4].

The von Kármán flow experiment using liquid sodium (hereafter VKS) is a good example of such difficulties. The facility uses 150 liters of liquid sodium that are set in turbulent motion by two counter-rotating impellers in a cylindrical container [5]. During its 16 years of operations, the VKS facility produced dynamo onset only with (i) impellers made of soft iron [5], and not with cooper or steel, (ii) for fluid motions with limited fluctuations [6]. This last observation rules out the possibility that the VKS dynamo is a mere *rotating magnet*. It also suggests that velocity fluctuations may impede the dynamo process, as already proved by dedicated numerical simulations in a simple geometry [7] or asymptotics [6, 8]. Various attempts have been made to explain such results within the mean-field framework. For example, Fauve and Petrelis suggested a possible $\alpha - \Omega$ mechanism for the dynamo, based on the combination of eddies generated near the impellers, and differential rotation [9, 10], a scenario confirmed by numerical simulations of a simplified model of the velocity field [11] or RANS simulations of the von Kármán flow [12]. This model however does not explain the important influence of the iron impellers on the dynamo onset. Additional physical processes were then invoked such as the interplay between the Ω effect and a new conversion effect linked with spatially varying electrical conductivity [13, 14, 15] or magnetic permeability [16, 17, 18, 19]. Finally, enhancement of the α effect via eddy collimation by the impeller magnetic field was discussed in [20, 21].

Detailed verification of these hypotheses is difficult in the context of the mean-field model since it is a large-scale model where all the small-scale action is parameterized in the coefficients α and Ω. Here, we propose a new approach to unravel these mechanisms, motivated by the observation that dynamo is a conversion mechanism between magnetic and kinetic energy. The core of our approach is based on local (in space, scale and time) energy budget describing dissipation and scale-by-scale energy transfers. Identifying the locations (in scale, space, and time) where kinetic and magnetic energies are exchanged supplies information about the location of the dynamo effect. Comparison with local magnetic or energy dissipation then tells us whether the conversion is sufficient to overcome damping, and thus, sustain dynamo action. The formalism is very general and applies to any geometry or boundary conditions. Then, by applying it to the case of the VKS dynamo, we show how it can be used to detect where and when the energy conversion between kinetic and magnetic energy takes place, thereby tracing the processes at work in the dynamo. Specifically, we use the massively parallel multiphysics code called SFEMaNS (for Spectral/Finite Element code for Maxwell and Navier-Stokes equations). This code

solves the magnetohydrodynamics equations (MHD) describing the time evolution of the velocity and magnetic fields in the von Kármán sodium experiment. With this code two types of dynamo action have been explored: one using soft iron impellers and the other using steel impellers [22]. Although the two types of dynamo hardly differ from the mean-field theory point of view (the velocity fields are the same in both cases), the locality of our formalism allows us to trace the origin of the differences between these two types of dynamo.

The paper is organized as follows. The coupled equations describing the motion of a conducting fluid are presented in section §2. We first recall elements of basic mean-field theory, focusing on typical α and Ω dynamo mechanisms. In section §3, we present our new formalism, based on the derivation of local energy budgets from the MHD equations. We first build coarse-grained energy budgets including inertial dissipation, anomalous dissipation and transfer terms between velocity and magnetic fields. Taking the limit of infinite resolution, we obtain exact energy budgets with no filter that are discussed. The generalization of the famous Kolmogorov 4/3rd law is presented in this context. This formalism is applied to the numerical simulations of the dynamo in the von Kármán geometry that are presented in sections §4.1-4.2-4.3. A crucial ingredient for application of the formalism is the notion of scale filtering. Traditional definitions of scale filtering, through convolution with a filter, are not suitable for unstructured grids. We, therefore, define a suitable filtering process that applies to any type of grid and boundary conditions, including the unstructured grid used in our numerical simulations in section §4.4. We discuss the standard dynamo mechanisms based on the mean-field theory in section §4.5. In section §5, we use the energy budget to unravel the dynamo mechanisms at work in the VKS experiment for impellers with different magnetic permeabilities. The same tools are used to study the saturation of the dynamo. This is done in section §6. Using these results, we discuss in section §7 the mechanisms for dynamo action in the VKS experiment and compare them with the literature. Our conclusion follows in section §8.

2 Background material and general formalism

2.1 Magnetohydrodynamics equations

The magnetohydrodynamics equations (MHD) for a neutral conductive incompressible Newtonian fluid in the quasi-static approximation are:

$$\partial_t u + \partial_i(u_i u) - \nu\Delta u + \frac{\nabla p}{\rho} = -\frac{b\times j}{\rho} + f, \tag{1}$$

$$\partial_t b = \nabla\times(u\times b) - \nabla\times\frac{j}{\sigma}, \tag{2}$$

$$\nabla\cdot u = 0, \tag{3}$$

$$\nabla\cdot b = 0, \tag{4}$$

$$j = \nabla\times H, \tag{5}$$

$$b = \mu H, \tag{6}$$

where H is the magnetic field, b the magnetic flux density (i.e. magnetic induction[1]), j the electric current density, u the fluid velocity (with u_i one of its components with $i\in\{1,2,3\}$), p the fluid pressure, f the force per unit mass, μ the magnetic permeability, σ the electric conductivity, ν the kinematic viscosity and ρ the fluid density. The space and time dependence of all fields has been removed in the notations for convenience except when necessary.

2.2 Mean field theory: α and Ω effects

We briefly summarize classical mean-field concepts that are traditionally used to interpret numerical results and to understand the basic dynamo mechanisms. For the sake of simplicity, it is assumed here

[1] In the following, we will abusively call b the magnetic field.

that σ and μ do not vary in space nor in time. Therefore equation (2) reads:

$$\partial_t b = \nabla \times (u \times b) + \frac{1}{\mu\sigma}\nabla^2 b. \tag{7}$$

We consider b and u as superimpositions of mean and fluctuating parts such that $b = \langle b\rangle + b'$ and $u = \langle u\rangle + u'$. Injecting these decompositions into equation (7) and taking its average, we get:

$$\partial_t \langle b\rangle = \nabla \times (\langle u\rangle \times \langle b\rangle + \text{emf}) + \frac{1}{\mu\sigma}\nabla^2\langle b\rangle \;\text{ with }\; \nabla\cdot\langle b\rangle = 0 \tag{8}$$

where $\text{emf} = \langle u' \times b'\rangle$ is the mean electromotive force due to the fluctuations of motion and magnetic field. These equations together with proper initial and boundary conditions determine $\langle b\rangle$ if $\langle u\rangle$ and emf are given. The determination of the mean electromotive force as a function of $u', \langle b\rangle, \langle u\rangle$ is the crucial step in the development of mean-field theory. For the sake of simplicity, we focus only on a low magnetic Prandtl situation $\nu\mu\sigma \ll 1$, relevant to a fluid such as liquid metal. In such a case, we can consider that the dynamics of the magnetic fluctuations b' is dominated by the interaction of the velocity fluctuations u' with the mean magnetic field $\langle b\rangle$ only. This means that:

$$\partial_t b' \simeq \nabla \times (u' \times \langle b\rangle) = (\langle b\rangle \cdot \nabla)u' - (u'\cdot\nabla)\langle b\rangle. \tag{9}$$

This fluctuating magnetic field is used to compute the electromotive force as detailed in [2]:

$$\text{emf}_i = \alpha_{ij}\langle b_j\rangle + \beta_{ijk}\partial\langle b_j\rangle/\partial x_k \tag{10}$$

where the tensors α_{ij} and β_{ijk} are mean quantities determined by u'. The α-effect is the occurrence of a mean electromotive force with a component pointing in the direction of the mean magnetic field $\langle b\rangle$. Based on results from [12], we will focus on this effect and discard the β tensor. Approximating the fluctuations by a short in-time field (supposing a short turbulence correlation time), one can link the α tensor to the helicity tensor as

$$\alpha_{ij} = \tau h_{ij} = \tau\varepsilon_{ikn}u_k(\nabla u)_{nj} \tag{11}$$

where τ is the correlation time of the velocity perturbations, ε_{ikn} is the Levi-Civita tensor and ∇u is the velocity gradient expressed in cylindrical coordinates (see appendix F).

Another conversion mechanism is due to differential rotation. Indeed, using the cylindrical coordinate system, the induction equation (7) can be written as:

$$\left[\frac{\partial}{\partial t} + u_r\frac{\partial}{\partial r} + \frac{u_\theta}{r}\frac{\partial}{\partial\theta} + u_z\frac{\partial}{\partial z}\right] b_r = \left[b_r\frac{\partial}{\partial r} + \frac{b_\theta}{r}\frac{\partial}{\partial\theta} + b_z\frac{\partial}{\partial z}\right]u_r + \frac{1}{\mu\sigma}\left[\Delta_* b_r - \frac{2}{r^2}\frac{\partial b_\theta}{\partial\theta}\right] \tag{12}$$

$$\left[\frac{\partial}{\partial t} + u_r\left(\frac{\partial}{\partial r} - \frac{1}{r}\right) + \frac{u_\theta}{r}\frac{\partial}{\partial\theta} + u_z\frac{\partial}{\partial z}\right] b_\theta = \left[b_r r\frac{\partial}{\partial r} + b_\theta\frac{\partial}{\partial\theta} + b_z r\frac{\partial}{\partial z}\right]\left(\frac{u_\theta}{r}\right) + \frac{1}{\mu\sigma}\left[\Delta_* b_\theta + \frac{2}{r^2}\frac{\partial b_r}{\partial\theta}\right] \tag{13}$$

$$\left[\frac{\partial}{\partial t} + u_r\frac{\partial}{\partial r} + \frac{u_\theta}{r}\frac{\partial}{\partial\theta} + u_z\frac{\partial}{\partial z}\right] b_z = \left[b_r\frac{\partial}{\partial r} + \frac{b_\theta}{r}\frac{\partial}{\partial\theta} + b_z\frac{\partial}{\partial z}\right]u_z + \frac{1}{\mu\sigma}\nabla^2 b_z \tag{14}$$

with $\Delta_* = \nabla^2 - \frac{1}{r^2} = \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial\theta^2} + \frac{\partial^2}{\partial z^2} - \frac{1}{r^2}$. This form of the induction equation clearly shows that, if the poloidal magnetic field, *i.e.* $\{b_r, b_z\}$, is non-zero, its shearing by the differential rotation (u_θ/r) will always generate a toroidal magnetic field, *i.e.* b_θ. This mechanism is called the Ω-effect. Denoting the two main terms in the rhs of (13):

$$\Omega'_r = \frac{\partial}{\partial r}\left(\frac{u_\theta}{r}\right), \qquad \Omega'_z = \frac{\partial}{\partial z}\left(\frac{u_\theta}{r}\right),$$

one can see that the source terms for the toroidal field are $b_r r\Omega'_r$ and $b_z r\Omega'_z$.

The dynamo effect is due to a combination of poloidal and toroidal magnetic field generation mechanisms, in the form of closing loops. Therefore different dynamo mechanisms can be conceived: there is a possibility for the poloidal magnetic field $\{b_r, b_z\}$ to be generated from the toroidal field b_θ through the α-effect. To close the loop and generate further toroidal field from such poloidal field, there are two

possibilities (possibly concomitant): either through the α-effect or (and) the mean differential rotation $r\Omega'_z$. The corresponding dynamo loops are then respectively called α^2 dynamo or $\alpha-\Omega$ dynamo.

3 Local Energy budget

In order to evaluate the energy budgets describing relevant transfers of energy between the magnetic and velocity fields, we use a filtering approach and consider products of resolved and non-resolved quantities. This strategy is inspired by the mathematical derivation of anomalous dissipation due to the irregularities of (i) velocity field in the context of the Navier-Stokes equations [23] and (ii) velocity and magnetic fields in the context of the magnetohydrodynamics equations [24, 25]. Our work extends these studies by introducing possible variations of the material properties (μ, σ). This is important in order to deal with the physical quantities resulting from the simulations presented in sections § 4.1-4.2-4.3 and to evaluate the impact of jumps in electrical conductivity σ and magnetic permeability μ.

3.1 Filtered equations

To characterise the energy transfers between scales of the same vector field, we use a filtering process in order to distinguish between the energy held by the different scales of the field. Given a vector field u and $\ell \in \mathbb{R}^+$, we denote its filtered field $\overline{u}^\ell$. At the present time, we do not specify the shape of the filter that is just assumed to obey the following properties: linearity, and such that $\overline{u}^\ell \xrightarrow{\ell\to 0} u$ for a smooth field u.

Applying the filter to the MHD equations (1) and (2), we get:

$$\partial_t \overline{u}^\ell + \partial_i \overline{u_i u}^\ell - \nu\Delta\overline{u}^\ell + \frac{\nabla\overline{p}^\ell}{\rho} = -\frac{\overline{b\times j}^\ell}{\rho} + \overline{f}^\ell, \tag{15}$$

$$\partial_t \overline{b}^\ell = \nabla\times(\overline{u\times b}^\ell) - \nabla\times\overline{(\frac{j}{\sigma})}^\ell\ , \tag{16}$$

$$\nabla\cdot\overline{u}^\ell = 0, \tag{17}$$

$$\nabla\cdot\overline{b}^\ell = 0. \tag{18}$$

Because the density and the viscosity of the fluid are constant in our simulations, they do not impact the filtering and can be taken out of the convolution.

3.2 Local MHD coarse-grained energy budget at finite scale ℓ

Summing the scalar product of $\rho u/2$ with equation (15) and the scalar product of $\rho\overline{u}^\ell/2$ with equation (1) and using analogous products for b, the equations for the filtered kinetic energy $E^c = \frac{\rho u\overline{u}^\ell}{2}$ and filtered magnetic energy $E^m = \frac{b\overline{b}^\ell}{2\mu}$ can be respectively written as:

$$\partial_t E^c + \nabla\cdot J^{NS} + \mathcal{D}^\nu + \mathcal{D}^u + \mathcal{D}^b = -\mathcal{T}^{u\to b} + \mathcal{P}, \tag{19}$$

$$\partial_t E^m + \nabla\cdot J^M + \mathcal{D}^\sigma + \mathcal{D}^{\overline{\sigma}} + \mathcal{D}^b + \mathcal{D}^{\mu\sigma} + \mathcal{D}^{\mu\mathcal{T}} = \mathcal{T}^{u\to b} + \mathcal{M}^\mu, \tag{20}$$

with:

$$J^{NS} = \frac{\rho}{2}((u\overline{u}^{\ell})u - \nu\nabla(u\overline{u}^{\ell})) + \frac{1}{2}(\overline{u}^{\ell}p + u\overline{p}^{\ell}), \tag{21}$$

$$J^{M} = \frac{1}{2\mu}\{\frac{j}{\sigma} \times \overline{b}^{\ell} + \overline{(\frac{j}{\sigma})}^{\ell} \times b + b \times (\overline{u \times b}^{\ell}) + \overline{b}^{\ell} \times (u \times b)\}, \tag{22}$$

$$\mathcal{P} = \frac{\rho}{2}(u \cdot \overline{f}^{\ell} + \overline{u}^{\ell} \cdot f), \tag{23}$$

$$\mathcal{D}^{\nu} = \rho\nu\partial_i\overline{u_j}^{\ell}\partial_i u_j, \tag{24}$$

$$\mathcal{D}^{\sigma} = \frac{j \cdot \overline{j}^{\ell}}{\sigma}, \qquad \mathcal{D}^{\overline{\sigma}} = \left(\overline{(\frac{j}{\sigma})}^{\ell} - \frac{(\overline{j}^{\ell})}{\sigma}\right) \cdot \frac{j}{2}, \tag{25}$$

$$\mathcal{D}^{u} = \frac{\rho}{2}(u\overline{u_i\partial_i u}^{\ell} - uu_i\partial_i\overline{u}^{\ell}), \tag{26}$$

$$\mathcal{D}^{b} = \frac{1}{4}(\overline{u}^{\ell} \cdot (b \times j) + u \cdot (\overline{b \times j}^{\ell}) - (\overline{u \times b}^{\ell}) \cdot j - (u \times b) \cdot \overline{j}^{\ell}), \tag{27}$$

$$\mathcal{T}^{u\to b} = \frac{1}{4}(\overline{u}^{\ell} \cdot (b \times j) + u \cdot (\overline{b \times j}^{\ell}) + (\overline{u \times b}^{\ell}) \cdot j + (u \times b) \cdot \overline{j}^{\ell}), \tag{28}$$

$$\mathcal{M}^{\mu} = -\frac{b \cdot \overline{b}^{\ell}}{2\mu^2}\partial_t\mu, \tag{29}$$

$$\mathcal{D}^{\mu\sigma} = \frac{j}{2\sigma} \cdot \nabla \times \left(\frac{(\overline{b}^{\ell})}{\mu} - \overline{H}^{\ell}\right), \qquad \mathcal{D}^{\mu\mathcal{T}} = -\frac{(u \times b)}{2} \cdot \nabla \times \left(\frac{(\overline{b}^{\ell})}{\mu} - \overline{H}^{\ell}\right). \tag{30}$$

All terms involved in equations (19)-(20) measure the transferred energy from all scales larger than ℓ to the scale ℓ. By convention, we do not use a subscript ℓ for all terms in contrast with [24, 26, 27]. The demonstration for (19) (respectively (20)) is in appendix A (respectively in appendix B).

We first discuss the different terms appearing in equation (19). On the left-hand side (lhs), J_{NS} represents an energy current responsible for the spatial movement of energy and thus does not participate in any kind of dissipation or transfer of energy between the different scales. Integrated throughout space with the proper boundary conditions, it should disappear. $\mathcal{D}^{\nu}$ is the dissipation due to the kinematic viscosity: it corresponds to a loss of energy of the velocity field. $\mathcal{D}^{u}$ is the anomalous/Duchon-Robert dissipation. This term is the local energy transfer at scale ℓ and thus characterises where the irregularities, and possible singularities, appear in the fields. It is called anomalous dissipation because, in the limit $\ell \to 0$, $\mathcal{D}^{u}$ might not vanish [23]. These two different energy transfers ($\mathcal{D}^{\nu}$ and $\mathcal{D}^{u}$) have already been derived and implemented in the SFEMaNS code (see section §4.2) in order to study their effects in a pure hydrodynamic problem modeled by the Navier-Stokes equations [28]. $\mathcal{D}^{b}$ is a transfer term between different scales due to the interplay between velocity u and magnetic induction b fields. Like in the kinetic case, such a term could also produce an anomalous contribution to the magnetic energy budget in case the velocity field or the magnetic induction field are sufficiently irregular. If the contribution is positive, the net effect would then be an anomalous dissipation, like in the kinetic case. However, there is also the possibility that the contribution is negative, resulting in an *increase* of the magnetic energy due to irregularity. This would correspond to a scenario by which a dynamo is produced via a sufficiently rough velocity field [29], a mechanism known as *fast dynamo* mechanism [30].

The first right-hand side (rhs) term ($-\mathcal{T}^{u\to b}$) corresponds to the power of the Lorentz force and reflects an energy transformation between the fields u and b. It naturally appears with the opposite sign in the rhs of the magnetic energy budget (20) and is therefore responsible for the emergence and maintenance of the magnetic field. In other words, $\mathcal{T}^{u\to b}$ is a source term for the dynamo effect. The second term, $\mathcal{P}$, displays the mechanical power injected by the forcing.

Aside the $\mathcal{T}^{u\to b}$ term, new interesting energy transfer terms appear in (20). In the lhs part, as before, J_M is a purely local term that describes how magnetic energy is transported across the flow, and it should vanish after integration over space with the appropriate boundary conditions. $\mathcal{D}^{\sigma}$ is the

Joule dissipation, hence a loss of energy of the magnetic field, and can be used to trace, locally and across scales, the rate of resistive energy dissipation. $\mathcal{D}^{\overline{\sigma}}$ has contribution only where the electrical conductivity undergoes jumps, *i.e.* at $\{r = 1.4,\, 0 \leq \theta < 2\pi,\, -1 \leq z \leq 1\}$ between the liquid sodium steady layer and the copper wall of the container (see fig. 4.1a). $\mathcal{D}^{\mu\sigma}$ and $\mathcal{D}^{\mu\mathcal{T}}$ are two terms associated with the irregularity of the magnetic permeability μ which occurs only at the surface of the impellers. They are respectively denoted Joule induced in-between scales magnetic transfer and Lorentz induced in-between scales magnetic transfer. Lastly, in the rhs part, another source term for the dynamo effect appears, namely $\mathcal{M}^{\mu}$, which we will call the magnet effect. It originates from the variation in time of the magnetic permeability and is a direct energy transfer from the impellers to the magnetic field.

3.3 Exact local MHD energy budget

The exact local energy budget can be obtained by taking the limit $\ell \to 0$. A technical difficulty may arise if the field is sufficiently irregular. In that case, terms corresponding to energy transfer may not need to vanish, producing anomalous dissipation of kinetic energy [23] or magnetic energy [31, 32], or even dynamo action [29].

In the case where the field is sufficiently regular, so that these terms vanish, the time evolution for the kinetic and magnetic energies (per unit volume) simplifies to:

$$\partial_t \frac{\rho u^2}{2} + \nabla \cdot J_0^{NS} + \mathcal{D}_0^{\nu} = -\mathcal{T}_0^{u\to b} + \mathcal{P}_0, \tag{31}$$

$$\partial_t \frac{b^2}{2\mu} + \nabla \cdot J_0^{M} + \mathcal{D}_0^{\sigma} = \mathcal{T}_0^{u\to b} + \mathcal{M}_0^{\mu}, \tag{32}$$

with:

$$J_0^{NS} = \frac{\rho}{2}(u^2 u - \nu \nabla u^2) + up, \tag{33}$$

$$J_0^{M} = \frac{1}{\mu}\{\frac{j}{\sigma} \times b + b \times (u \times b)\}, \tag{34}$$

$$\mathcal{D}_0^{\nu} = \rho\nu \partial_i u_j \partial_i u_j, \qquad \mathcal{D}_0^{\sigma} = \frac{j^2}{\sigma}, \tag{35}$$

$$\mathcal{T}_0^{u\to b} = u \cdot (b \times j), \qquad \mathcal{M}_0^{\mu} = -\frac{b^2}{2\mu^2}\partial_t \mu, \qquad \mathcal{P}_0 = \rho u \cdot f. \tag{36}$$

As before, the J terms represent energy current responsible for the spatial displacement of energy and do not participate in transfers. The terms $\mathcal{D}_0^{\nu}$ and $\mathcal{D}_0^{\sigma}$ are the standard viscous and resistive dissipation terms respectively. The right-hand side of (32) shows the two dynamo source terms, namely (minus) the Lorentz force power and the magnet effect and the rhs of (31) shows the Lorentz force power and the mechanical power due to the forcing.

Note that in the case where the velocity and/or magnetic induction fields are irregular, other terms should be considered in the energy budget, due to the non-vanishing of terms like $\mathcal{D}^{u}$, $\mathcal{D}^{b}$, $\mathcal{D}^{\mu\sigma}$ and $\mathcal{D}^{\mu\mathcal{T}}$ in the limit $\ell \to 0$ in equations (19) and (20).

3.4 Definition of a local dynamo growth rate

We can use the energy budgets to define a global dynamo growth rate at several scales. Indeed the rhs of equation (20) evidences two possible source terms for the increase of the magnetic energy: the source term already discussed $\mathcal{T}^{u\to b}$ and the $\mathcal{M}^{\mu}$ term that exists only in case where permeability varies in time.[2] This last term is called in the sequel the "rotating magnet effect". Physically, the sum of these two terms must exceed the sum of all dissipative terms in the lhs of equation (20) for the dynamo to operate.

[2]This will be the case for example in the von Kármán setup if the impellers are made of permeable material such as, for example, soft iron [5].

This can be formalized with the local growth rate $\lambda^m(\vec{x},t)$ of the magnetic field at scale ℓ given by:

$$\lambda^m = \frac{\mathcal{T}^{u\to b} + \mathcal{M}^{\mu} - \mathcal{D}^{\sigma} - \mathcal{D}^{\overline{\sigma}} - \mathcal{D}^{b} - \mathcal{D}^{\mu\sigma} - \mathcal{D}^{\mu\mathcal{T}}}{\langle E^m \rangle} \tag{37}$$

where $\langle . \rangle$ denotes the spatial average. Therefore the evolution of the filtered magnetic energy (20) can be recast into:

$$\partial_t E^m + \nabla \cdot J^M = \lambda^m \langle E^m \rangle. \tag{38}$$

For $\ell \to 0$, the growth rate can have two types of behavior. For the first one, the field is sufficiently regular, in which case all transfer terms between scales vanish, and the local growth rate is simply:

$$\lambda_0^m = \frac{\mathcal{T}_0^{u\to b} + \mathcal{M}_0^{\mu} - \mathcal{D}_0^{\sigma}}{\langle E^m \rangle}. \tag{39}$$

This case will correspond to numerical simulations as no singularity can develop there because of the finite resolution. For the second type of behavior, in natural or experimental cases, there may be a contribution from the terms like $\mathcal{D}^{\overline{\sigma}}$, $\mathcal{D}^{b}$, etc, resulting in a possibility of dynamo generation by singularity if these terms are negative. This is the case discussed in [29].

3.5 Connection with previous works

Our formalism can be connected to previous works by introducing $\delta_{\boldsymbol{\ell}} u$, the longitudinal velocity increment at scale ℓ, defined as

$$\delta_{\boldsymbol{\ell}} u \ = \hat{\boldsymbol{\ell}} \cdot \ (\mathbf{u}(\mathbf{x}+\boldsymbol{\ell},t) - \mathbf{u}(\mathbf{x},t)) \tag{40}$$

where $\hat{\boldsymbol{\ell}}$ is the increment unit vector.

In [24], Galtier derives the local energy conservation in incompressible Hall MHD equations when the velocity and magnetic fields are possibly non-regular. He focuses on the time evolution of the total energy $E^c + E^m$ for a plasma with constant electrical conductivity and magnetic permeability, corresponding to the sum of equations (19) and (20). As a result, the source term $\mathcal{T}^{u\to b}$ disappears together with all terms depending on the variation of σ and μ (namely $\mathcal{M}^{\mu}$, $\mathcal{D}^{\overline{\sigma}}$, $\mathcal{D}^{\mu\sigma}$, $\mathcal{D}^{\mu\mathcal{T}}$). The energy dissipation by viscous and resistive effects is then reduced to one term $\mathcal{D}^{\nu} + \mathcal{D}^{\sigma}$. In the same way, the energy transfer term becomes lumped into mixed terms, involving both u and b, which does not allow separating the various source terms. The final energy budget can then be written using the notations of [24] (see equation (22) therein with the ion skin depth $d_I = 0$) as:

$$\begin{aligned}
\mathcal{D}_{\varepsilon} &= \frac{1}{4}\int (\nabla \varphi_{\ell}) \cdot (\delta u \delta u^2 + \delta u \delta b^2 - 2(\delta u \cdot \delta b)\delta b) d\vec{r} \\
&= S^{NS} + \tilde{S}^{Mag} \\
&= \frac{1}{\rho}\mathcal{D}^{u} + 2\mu\mathcal{D}^{b} + \frac{1}{2}\nabla \cdot (J_{\mathcal{S}^{Mag}} + J_{\mathcal{S}^{NS}})
\end{aligned} \tag{41}$$

where S^{NS} (resp. $\tilde{S}^{Mag}$) and $J_{\mathcal{S}^{NS}}$ (resp. $J_{\mathcal{S}^{Mag}}$) are defined in appendix C (resp. appendix D). Therefore our analysis has a direct link with previous works [24, 32], with the advantage that tracking E^c and E^m separately allows to get insights into the dynamo effect, as we now show.

4 Application to the VKS dynamo

4.1 Numerical setup

A standard setup for investigating turbulence in constrained geometry is the von Kármán swirling flow. The symmetries of this setup are presented in appendix F. It has been extensively studied using various fluids such as glycerol, water, air, helium gas and superfluid helium [26]. It was also used to study the dynamo action using liquid sodium in the so-called von Kármán Sodium experiment.

This experiment was successful in 2006 when a dominantly dipolar axisymmetric magnetic field with a toroidal magnetic field concentrated in the impellers was observed [5] for a magnetic Reynolds number larger than $R_m^c = 34$. We have performed MHD simulations of this setup on various supercomputers (up to about 2000 cores on the IDRIS supercomputer), allowing us to reach kinetic Reynolds number up to 10^5 and magnetic Reynolds number of the order of a few hundred, resulting in a magnetic Prandtl number around 10^{-3} [22]. This is larger than the experimental value, which is of the order $5\,10^{-6}$. Despite this, several key features of the magnetic field dynamo observed in the VKS experiment were recovered. The MHD simulations show the key role played by the ferromagnetic material (with high magnetic permeability) constituting the impellers. When soft iron impellers are modelled, the magnetic field is dominated by an axisymmetric axial dipole and azimuthal components located near the impellers in agreement with the experimental data (see figure 4.1b adapted from [22]).

4.2 SFEMaNS code

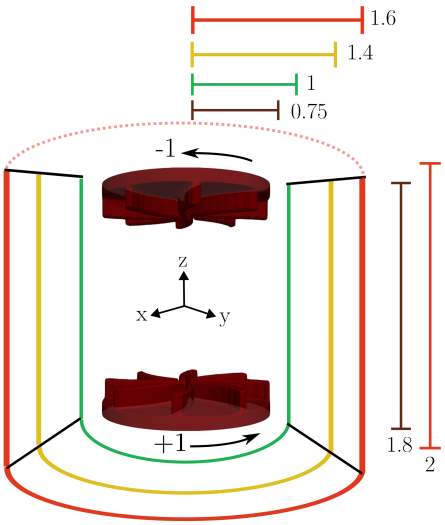
The SFEMaNS code integrates the system of coupled equations (1)-(4). It uses a hybrid spatial discretization combining finite elements and spectral decomposition. For the hydrodynamic part, the approximation in space is done by using a Fourier decomposition in the azimuthal direction and the continuous Hood-Taylor Lagrange element $\mathbb{P}_1$-$\mathbb{P}_2$ in the meridian section (quadratic approximation for the velocity field and linear approximation for pressure). All the discrete scalar functions f are written in the generic form:

$$f(r,\theta,z,t) = f_h^{c,0}(r,z,t) + \sum_{m_F=1}^{M} f_h^{c,m_F}(r,z,t)\cos(m_F\theta) + \sum_{m_F=1}^{M} f_h^{s,m_F}(r,z,t)\sin(m_F\theta), \qquad (42)$$

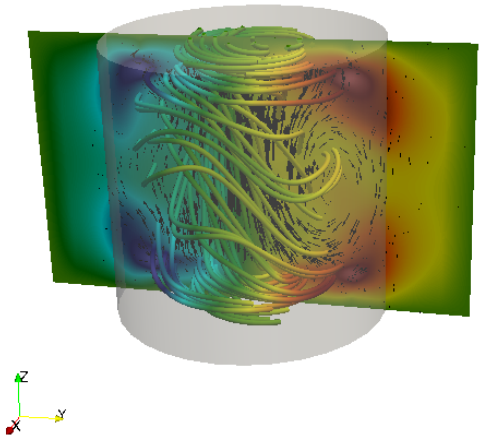
with (r,θ,z) the cylindrical coordinates, t the time and M the number of Fourier azimuthal modes considered. The functions f_h^{c,m_F} and f_h^{s,m_F} belong to a finite element space with h the typical mesh-size. The approximation in time is done by using a pressure-correction method [33]. The moving counter-rotating impellers are accounted for by using a pseudo-penalty technique described in reference [34]. Modulo the computations of nonlinear terms with the fast Fourier transform, the linear problems at each time step for each Fourier mode in the meridian section are uncoupled and are thereby parallelized by using the message passing interface. The solution of each linear problem in the meridian section is further parallelized by using graph partitioning techniques from the METIS library [35] and subroutines from the portable extensible toolkit for scientific computation library (PETSc) [36], for the linear algebra. For the magnetic part, the algorithm solves the problem using the magnetic induction b in the conducting region (after standard elimination of the electric field) and the scalar magnetic potential in the insulating exterior. The fields in each region are approximated by using H^1-conforming Lagrange elements with a penalty technique to control the divergence of b in a negative Sobolev norm that guarantees convergence under minimal regularity (see details in [37, §3.2], [38]). The coupling between conducting and insulating media is done by using an interior penalty method. SFEMaNS has been thoroughly validated on numerous manufactured solutions and against other MHD codes (see e.g. [39, 40]).

4.3 Description of the numerical runs

In this paper, we focus on two series of runs described in [22] (Table 3). We use non-dimensional units such that the reference length L_{ref} is set to the inner cylinder radius R_{cyl}. The computational domain for the hydrodynamic study is $\Omega = \{(r,\theta,z) \in [0,1]\times[0,2\pi)\times[-1,1]\}$. The computational domain for the MHD study is the larger cylinder $\Omega \cup \Omega_{\text{out}}$ with $\Omega_{\text{out}} = \{(r,\theta,z) \in [1,1.6]\times[0,2\pi)\times[-1,1]\}$ (see schematic on figure 4.1a). Denoting by σ_0 the electrical conductivity of the liquid sodium, ρ its density, and μ_0 the magnetic permeability of vacuum, the magnetic induction is made non-dimensional by using the Alfvén scaling $B = U\sqrt{\rho\mu_0}$, with $U = \omega R_{\text{cyl}}$ where ω is the angular velocity of the impellers. The two governing parameters are $R_m = \mu_0\sigma_0 R_{\text{cyl}}^2\omega$, the magnetic Reynolds number, and $R_e = R_{\text{cyl}}^2\omega/\nu$, the kinetic Reynolds number, with ν the kinematic viscosity of the fluid. We define a relative electrical conductivity σ_r and a relative magnetic permeability μ_r. These quantities are not constant since the walls and the impellers are made of different materials like copper, steel and soft iron. Specifically, we



(a) Vessel



(b) Magnetic field produced by dynamo action

Figure 4.1: (a) Schematic of the VKS experimental device of [5] in non-dimensional units. The impellers counter-rotate as indicated. (b) Magnetohydrodynamics simulation of the von Kármán Sodium dynamo at kinetic Reynolds number $R_e = 1500$, magnetic Reynolds number $R_m = 150$ and relative magnetic permeability of the impellers $\mu_r = 50$: visualization of the time-averaged magnetic field lines and section of the azimuthal magnetic field component. Adapted from [22].

define $\sigma_r = 1, \mu_r = 1$ in the region $\{(r, \theta, z) \in [1, 1.4] \times [0, 2\pi) \times [-1, 1]\}$ to model the lateral layer of stagnant liquid sodium, and $\sigma_r = 4.5, \mu_r = 1$ in $\{(r, \theta, z) \in [1.4, 1.6] \times [0, 2\pi) \times [-1, 1]\}$ to represent the lateral copper wall.

We use two series of runs described in detail in Table 3 of [22] at a moderate kinetic Reynolds number $R_e = 1500$ but with a different relative magnetic permeability for the impellers: $\mu_r = 1$ for steel impellers, $\mu_r = 50$ for soft iron impellers. As indicated, for pseudo-vacuum boundary conditions (the tangential components of H are zero on the outer cylinder), the critical magnetic Reynolds numbers for dynamo action are $R_m^c(\mu_r = 1) = 190 \pm 10$ and $R_m^c(\mu_r = 50) = 90 \pm 5$. Therefore we will use R_m values above the respective threshold, respectively $R_m(\mu_r = 1) = 300$ and $R_m(\mu_r = 50) = 150$ and in two regimes of the dynamo, i.e., in the growth and saturated phases. Each regime is sampled with 128 snapshots (every one-eighth of a turn to stroboscope the impellers' motion). We denote the average on all the snapshots as $\{.\}$.

The computations are performed with 128 Fourier modes and a non-uniform meridian mesh, with minimal spacing $h_{min} = 2.5 \times 10^{-3}$ and maximal spacing $h_{max} = 10^{-2}$ leading to an averaged meridian mesh $h_{mean} = 7.92 \times 10^{-3}$. We can estimate the Kolmogorov scale $\eta = (\nu^3/\epsilon)^{1/4}$ using the kinematic viscosity $\nu = 1/R_e$ and the energy dissipation rate $\epsilon = 2R_{\rm cyl}K_p/\pi H$ (with K_p the time-averaged torque applied by the impellers to the fluid and $H = 2$ the total height of the container). Table 2 of [22] provides $K_p = 5.08 \times 10^{-2}$ at $R_e = 1500$ resulting in $\eta = 1.16 \times 10^{-2}$.

4.4 Scale filtering

As the construction of local energy budgets relies heavily on the notion of filtering, we first present a general definition that is appropriate for any type of grid or boundary conditions. Classical definitions of scale filtering involve the convolution of a quantity with a filter of appropriate width (generally a Gaussian field). This definition is not well adapted to data on unstructured grids such as those used in SFEMaNS. We therefore consider another construction, that can be adapted to any type of grid, or boundary conditions. Given a vector field u and $\ell \in \mathbb{R}^+$, we define the filtered field $g = \overline{u}^\ell$ as the solution of:

$$g - \ell^2 \Delta g = u. \tag{43}$$

This definition ensures that it can be constructed for any type of grid, and is easy to implement inside

a numerical program. Moreover, it is endowed with all properties of a classical filter: it is linear and, for a continuous field u, we have the limit $\overline{u}^{\ell} \xrightarrow{\ell \to 0} u$. In fact, this filter corresponds to a convolution with a special filter. Indeed, in Fourier space, equation (43) reads:

$$\tilde{g} = \frac{\tilde{u}}{1 + \ell^2 k^2}, \tag{44}$$

where $\tilde{g}$ is the Fourier transform of g. Thus, $\overline{u}^{\ell}$ is the convolution of u with the inverse Fourier transform of the Lorentzian function $\frac{1}{1+\ell^2 k^2}$.

4.5 Analysis using the mean-field framework

For later discussions, we first analyze the two different dynamos within the classical mean-field framework in the saturated regime of the dynamo. For this, we compute in each case the local helicity tensor using the velocity field averaged on the 128 snapshots. They are shown on figures 4.2-4.3. Their spatial distribution is complex and exhibits fine scales. Note that the dominant patterns of each local helicity component follow the symmetry rules detailed in appendix F (see equation (84)). This means that the large structures of the velocity field are mainly R_π symmetric at $R_e = 1500$ for both μ_r values.

We can see that the maxima in absolute value are always localized near the impellers like in the RANS simulation of [12] whereas the minima are dispersed over the whole fluid domain. The maxima are associated with the swirling vortices attached to each blade and occupying part of the space between the blades. These vortices are thought to be a key ingredient of the dynamo mechanism [9, 41, 42].

The maximum absolute non-dimensional value of around 10 for both magnetic permeability values corresponds to the $\{h_{rr}\}, \{h_{zz}\}$ components. These terms may be associated with an α-effect that would transform a poloidal field (b_r, b_z) into a toroidal field b_θ (first phase of the dynamo loop). To close the dynamo loop, we need the $\{h_{\theta\theta}\}$ term that would transform the toroidal field into the poloidal field (second phase). This would lead to an α^2 dynamo mechanism, *e.g.* $(b_r, b_z) \xrightarrow{\{h_{rr}\},\{h_{zz}\}} b_\theta \xrightarrow{\{h_{\theta\theta}\}} (b_r, b_z)$.

Another possibility to transform a poloidal field into a toroidal field (first phase of the dynamo loop) is to resort to Ω terms as seen in equation (13). This would lead to an $\Omega - \alpha$ dynamo mechanism, *e.g.* $(b_r, b_z) \xrightarrow{\{\Omega'_r\},\{\Omega'_z\}} b_\theta \xrightarrow{\{h_{\theta\theta}\}} (b_r, b_z)$. These averaged Ω-terms are shown in figure 4.4. Their large structures are invariant under the R_π symmetry (as expressed in equation (82)). They are similar for the two values of μ_r and their maximum absolute non-dimensional value is around 10^2.

These two dynamo mechanisms may be present in the VKS setup as it is underlined in [12, 20, 21]. Our results are not in contradiction with these explanations. However, as we now show, the local energy budget procedure allows for a much finer analysis of the dynamo mechanisms.

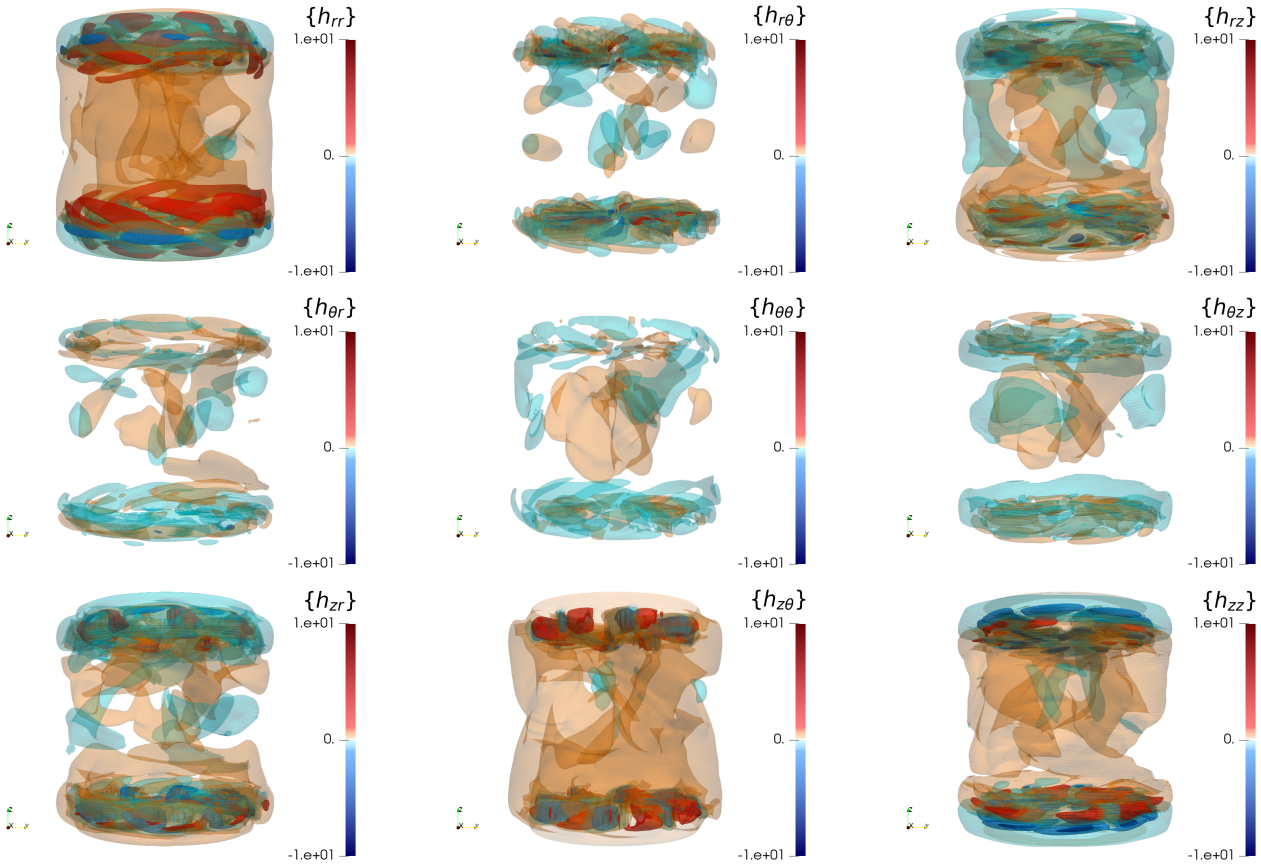


Figure 4.2: Time-averaged helicity tensor $\{h_{ij}\}$ in the saturated regime for $\mu_r = 50$.

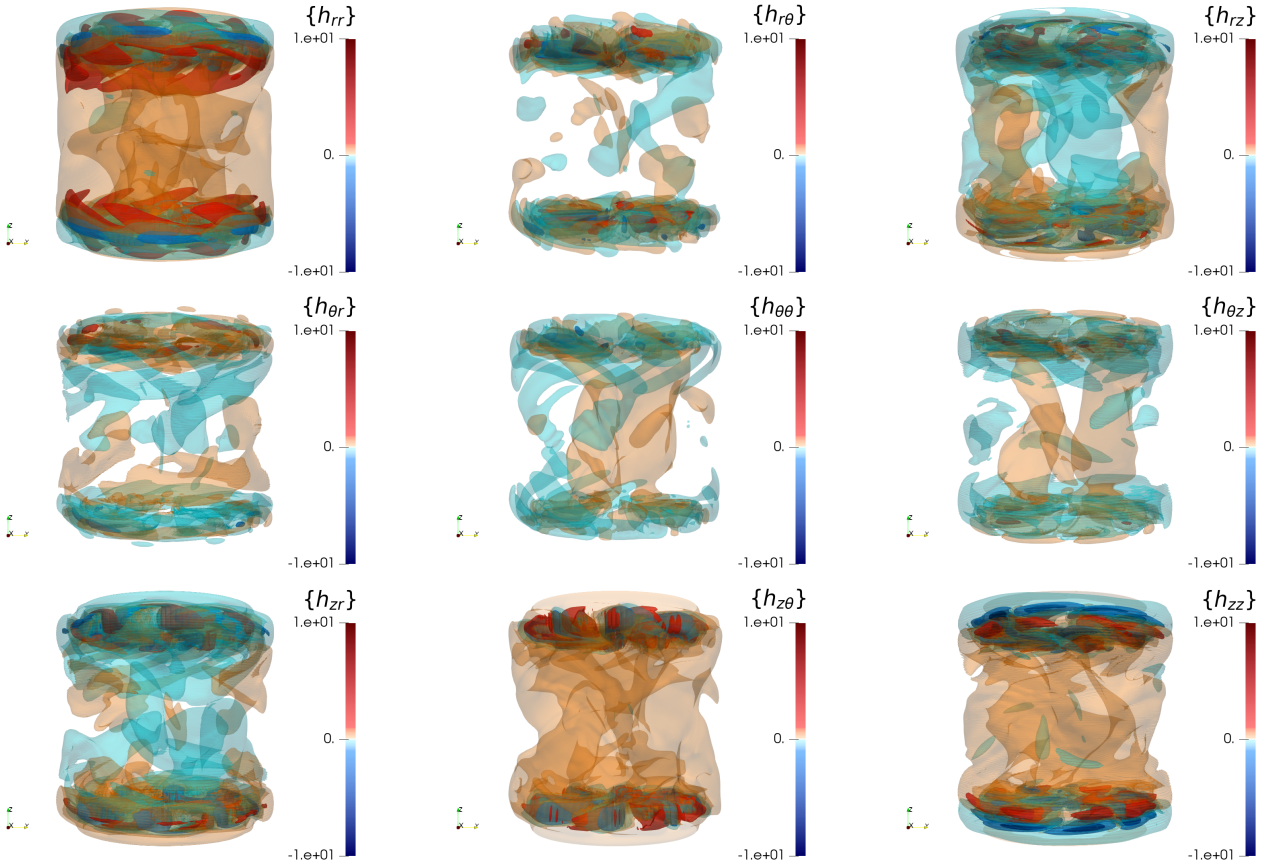


Figure 4.3: Time-averaged helicity tensor $\{h_{ij}\}$ in the saturated regime for $\mu_r = 1$.

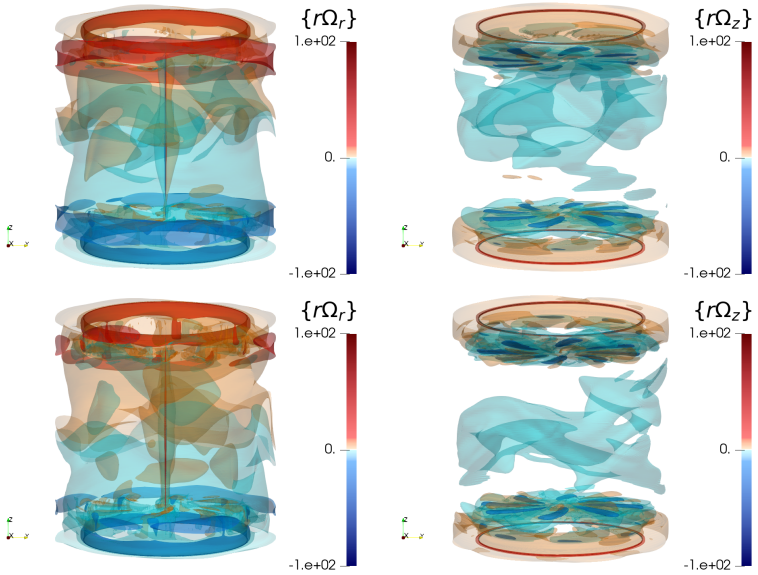


Figure 4.4: Time-averaged Ω effect in the saturated regime. Top row: $\mu_r = 50$, bottom row $\mu_r = 1$

5 Dynamo growth mechanisms from local energy budgets

We now turn to the analysis of the local energy budgets and show how they can be applied to identify important dynamo mechanisms that are absent in the mean-field formulation adapted to statistically stationary problems. The filtering is computed at three different scales $\frac{\ell}{\eta} = (0, 1, 16)$. The case $\frac{\ell}{\eta} = 0$ is used to ascertain the validity of the calculations, *i.e.* it is used to verify that $\mathcal{D}^{\overline{\sigma}} = \mathcal{D}^u = \mathcal{D}^b = \mathcal{D}^{\mu\sigma} = \mathcal{D}^{\mu\mathcal{T}} = 0$ in this case, where the fields are regular by construction. The other two filtering scales are chosen so as to identify the the mechanisms at small and large scale. Similarly, $\mathcal{M}^\mu = \mathcal{D}^{\mu\sigma} = \mathcal{D}^{\mu\mathcal{T}} = 0$ is expected for $\mu_r = 1$ (impellers with vacuum magnetic permeability such as the steel ones).

To proceed, we first analyze the spatial average of each quantity of the energy budgets (19), (20) as a function of time in the growing phase. We next describe the spatial repartition of each term of the energy budget averaged on a time range belonging to the exponential growth regime for the two types of dynamo action.

5.1 Time evolution of energy terms in the dynamo growth phase

Figure 5.1 shows that, in the growth regime of the dynamo, all transfer terms in the kinetic energy budget (19) are very similar for $\mu_r = 50$ and $\mu_r = 1$ and for each filtering scale. The reason is that the magnetic field is too weak to affect the velocity field. The only relevant terms are the viscous dissipation $\mathcal{D}^\nu$ and the anomalous dissipation $\mathcal{D}^u$ (for $\ell > 0$). $\mathcal{D}^\nu$ decreases with ℓ as expected since viscosity acts on small scales. In contrast, $\mathcal{D}^u$ increases with ℓ because the larger ℓ as a result of energy cascade from large to small scale, see [26].

Figure 5.2 shows the exponential growth in time (in absolute value) of all terms involved in the magnetic energy budget (20) for the two magnetic permeabilities. In order to better discriminate the respective role of these terms, we renormalize them on figure 5.3 by the average magnetic energy $\langle E^m \rangle(t)$.

We can note that the two dynamo regimes are different. For $\mu_r = 1$ and $\ell = 0$, the growth rate is given by:

$$\lambda_0^m = \frac{\mathcal{T}_0^{u\to b} - \mathcal{D}_0^\sigma}{\langle E^m \rangle}. \tag{45}$$

Therefore $\mathcal{T}_0^{u\to b}$ must exceed $\mathcal{D}_0^\sigma$ for the magnetic field to grow. The situation is similar for $\ell = \eta$ where the terms associated with irregularity such as $\mathcal{D}^b$ and $\mathcal{D}^{\overline{\sigma}}$ are quasi-zero due to the fact that the fields are somehow regular at this scale. Increasing ℓ enhances the irregular transfer term $\mathcal{D}^b$ (which is positive) and diminishes $\mathcal{T}^{u\to b}$ and $\mathcal{D}^\sigma$ while $\mathcal{D}^{\overline{\sigma}}$ stays negligible. This may be an indication that in

the inviscid limit, the dynamo could be sustained by irregularities of the velocity field, so that a fast dynamo could be achieved. However, in the present moderate Reynolds number regime, the dominant term is $\mathcal{T}^{u\to b}$ which compensates for all dissipative terms.

For $\mu_r = 50$, the first instants ($t < 3$) show the same behavior as that for $\mu_r = 1$. This means that $\mathcal{T}^{u\to b}$ is crucial in "launching" the dynamo process. After the launch is achieved, however, the dynamo trajectory becomes very different, as the rotating magnet effect $\mathcal{M}^{\mu}$ takes over and maintains the dynamo action, while $\mathcal{T}^{u\to b}$ is negative acting therefore like an anti-dynamo Lorentz power. Increasing ℓ enhances the modulus of the irregular terms: $\mathcal{D}^{\mu\mathcal{T}}$ relates the jump in μ and the electromotive field ($u \times b$) and is positive; $\mathcal{D}^{b}$ follows the variations of $\mathcal{T}^{u\to b}$ and is negative, *i.e.* it is an inverse cascade for the energy transfer between scales for both the velocity and magnetic fields. The terms $\mathcal{D}^{\mu\sigma}$ (correlation between the jump in μ and the current) and $\mathcal{D}^{\overline{\sigma}}$ (dissipation due to the jump in σ) are negligible and can be ignored.

In summary, the term $\mathcal{T}^{u\to b}$ is essential to launch the dynamo in both cases, highlighting the crucial role played by the velocity field structure in the lightning of the dynamo. However, the further development of the dynamo is very different in the two cases, as $\mathcal{M}^{\mu}$ becomes the driving source term for the dynamo when using soft iron impellers, while $\mathcal{T}^{u\to b}$ becomes anti-dynamo. To further characterize these different mechanisms, we now consider the spatial localization of the local growth rate of the magnetic field.

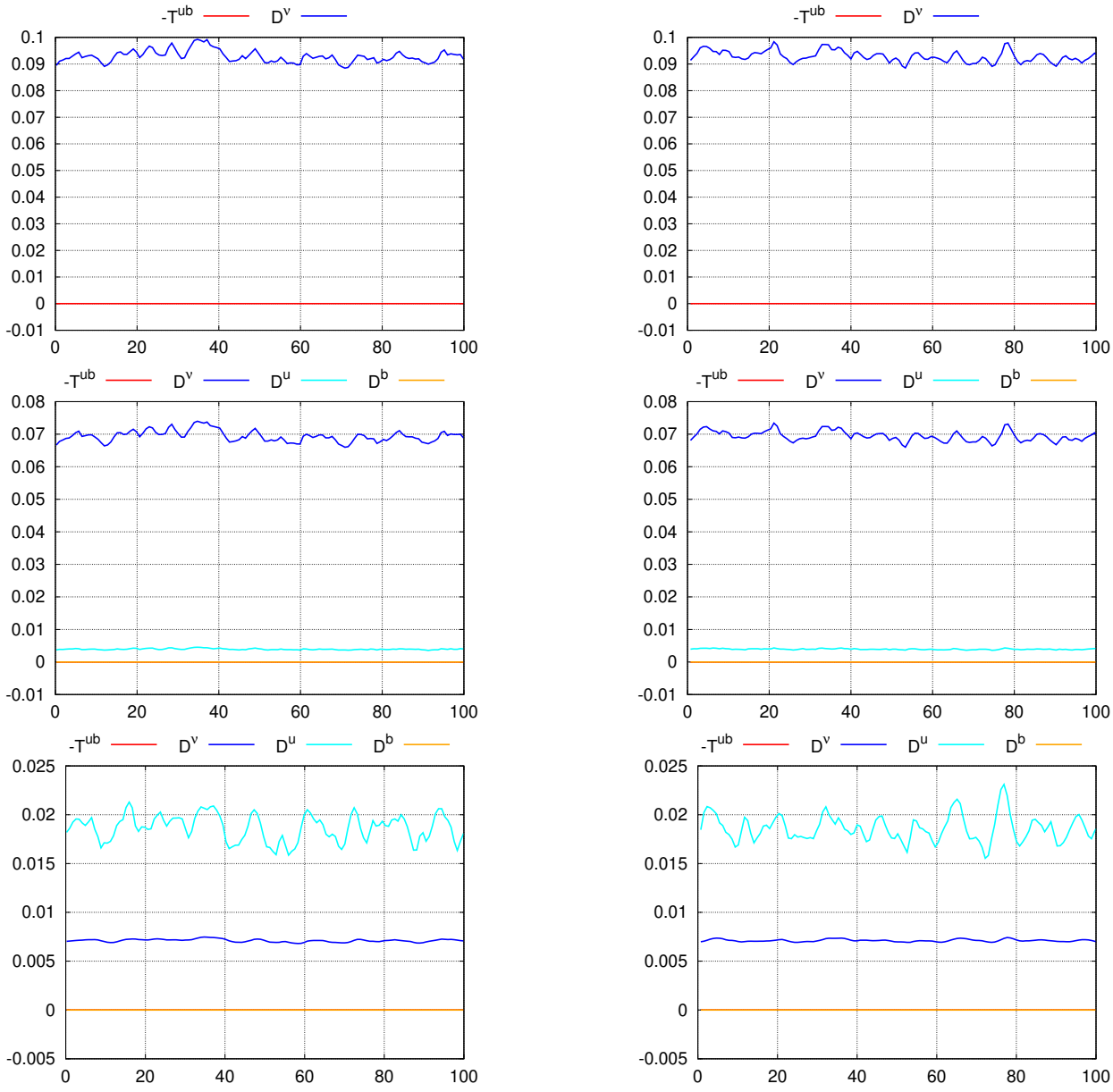


Figure 5.1: Time evolution of kinetic energy budget (19) in the growing phase. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

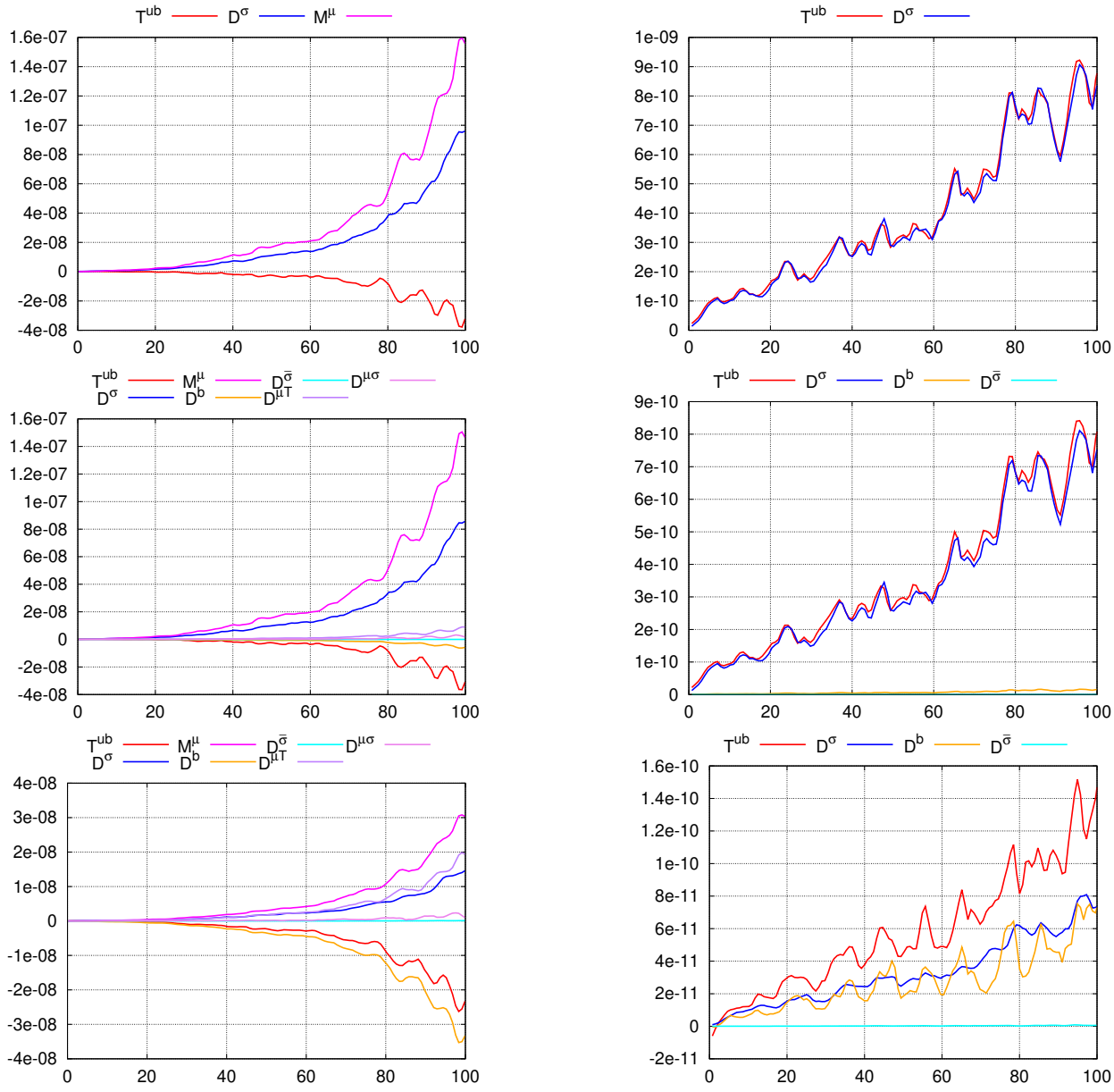


Figure 5.2: Time evolution of magnetic energy budget (20) in the growing phase. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

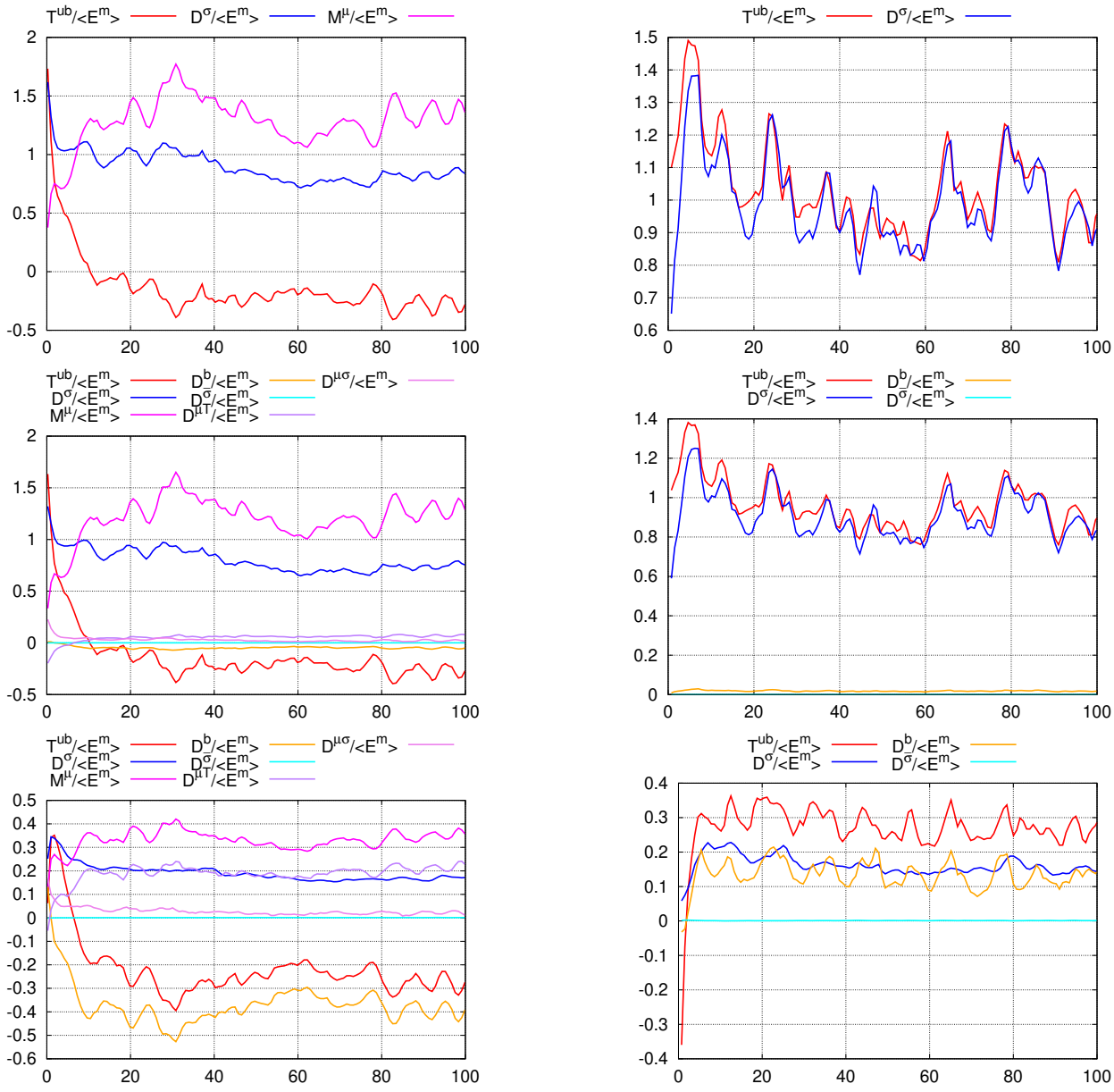


Figure 5.3: Time evolution of renormalized magnetic budget in the growing phase. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

5.2 Spatial distribution of linear growth rate

We use equations (37) and (39) to determine the local dynamo growth rate. The spatial distributions are different depending on the value of the magnetic permeability: for $\mu_r = 50$, the growth rate is localized near the impellers for each ℓ while, for $\mu_r = 1$, the main localizations are near the impellers and the shear-layer. In this region, the growth rate is positive inside the bulk and negative near the lateral wall at $r = 1$ for each ℓ. Therefore the source of the growing magnetic field is very different depending on the impellers material.

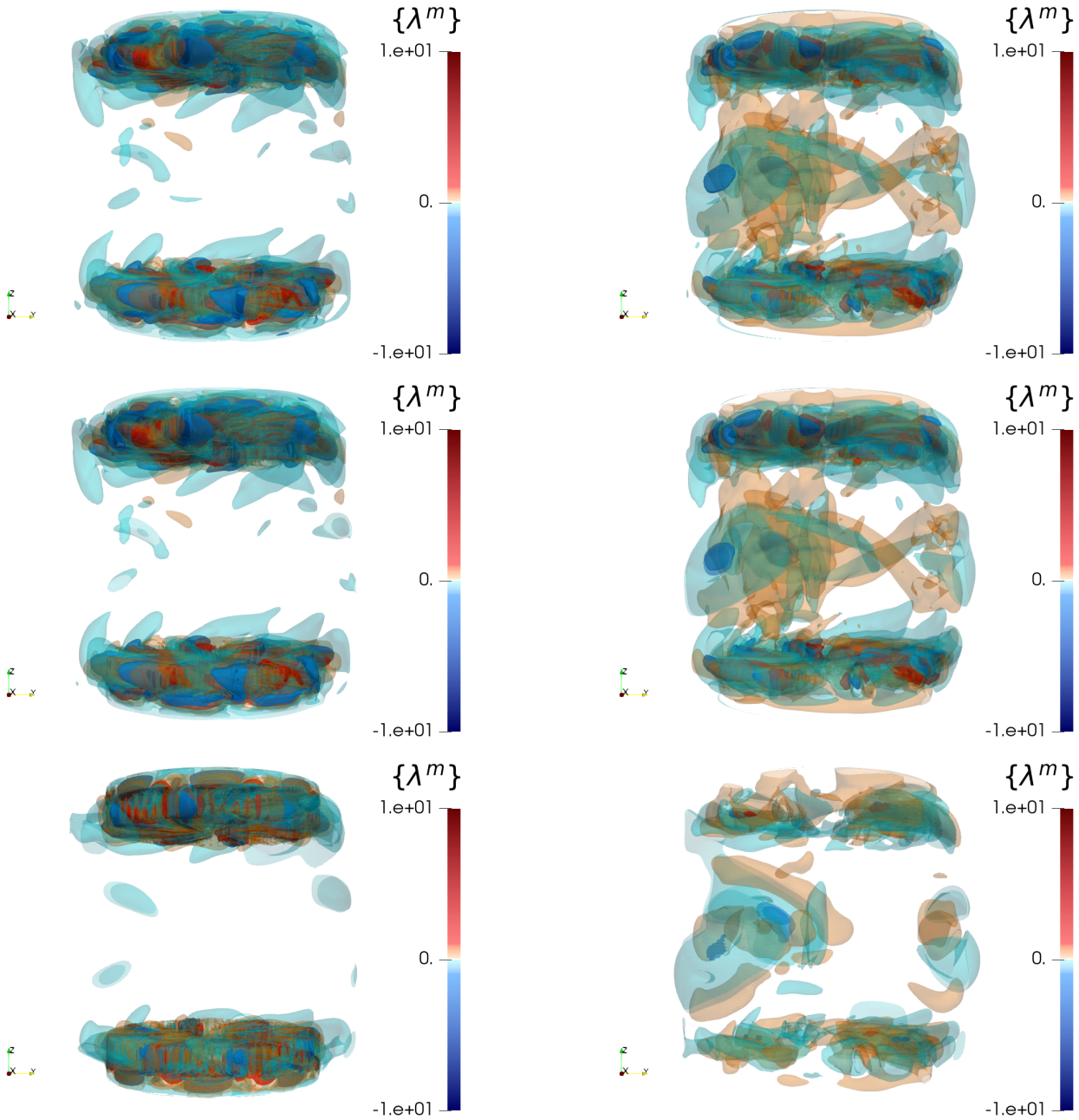


Figure 5.4: Dynamo growth rate averaged on the 128 snapshots in the dynamo growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

5.3 Kinetic and magnetic energy distributions in the growth regime

The spatial distribution of the kinetic energy (fig. 5.5) is the same for both permeabilities for each scale because the magnetic field is too weak to impact the velocity field in the growth phase.

The linear mode of the magnetic energy exponential growth (fig. 5.6) is mostly axisymmetric and localized in the impellers' region for $\mu_r = 50$ while it is distributed in the bulk for $\mu_r = 1$. The structures are smoother with the increase of ℓ.

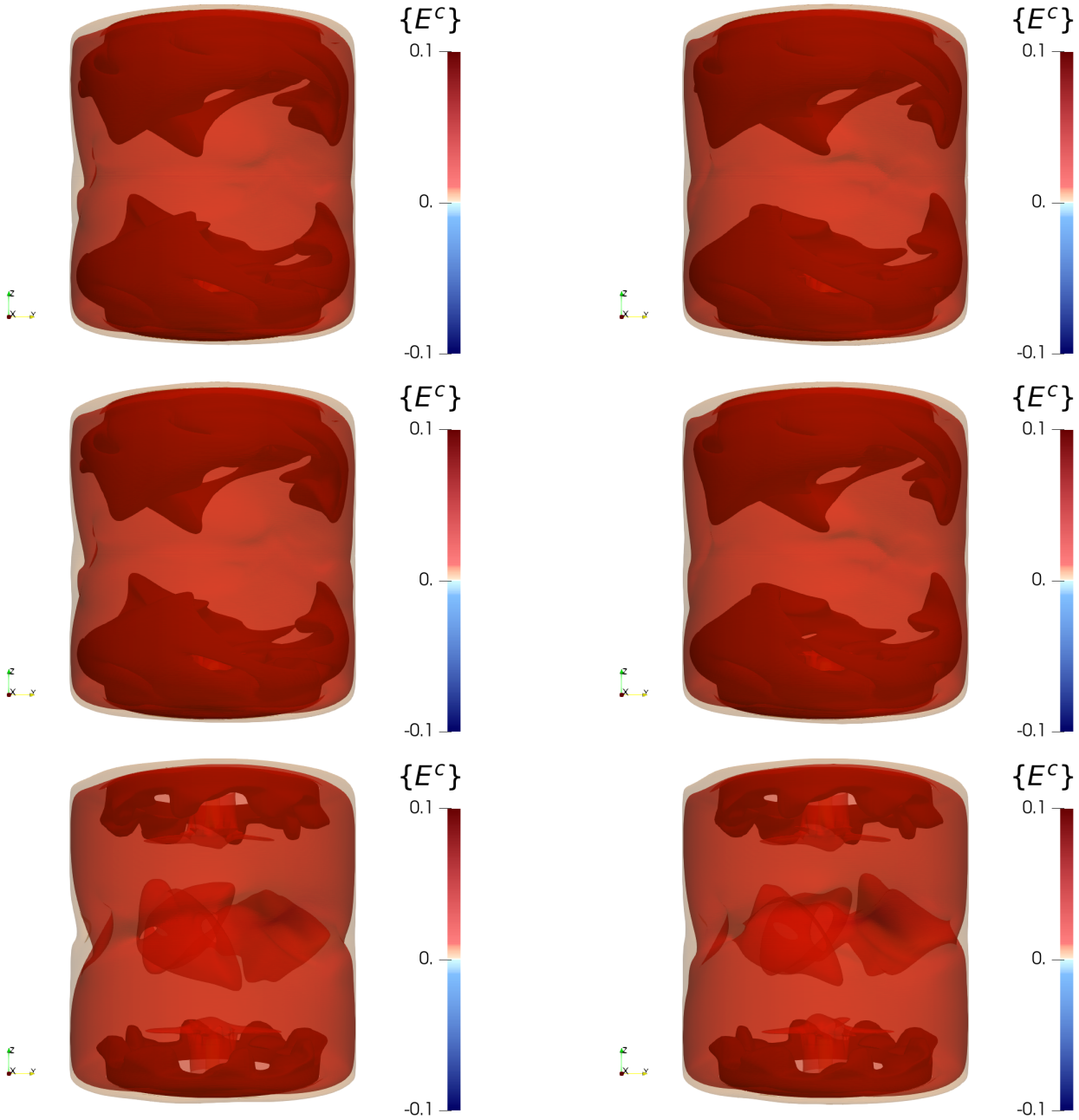


Figure 5.5: Kinetic energy averaged on the 128 snapshots in the dynamo growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

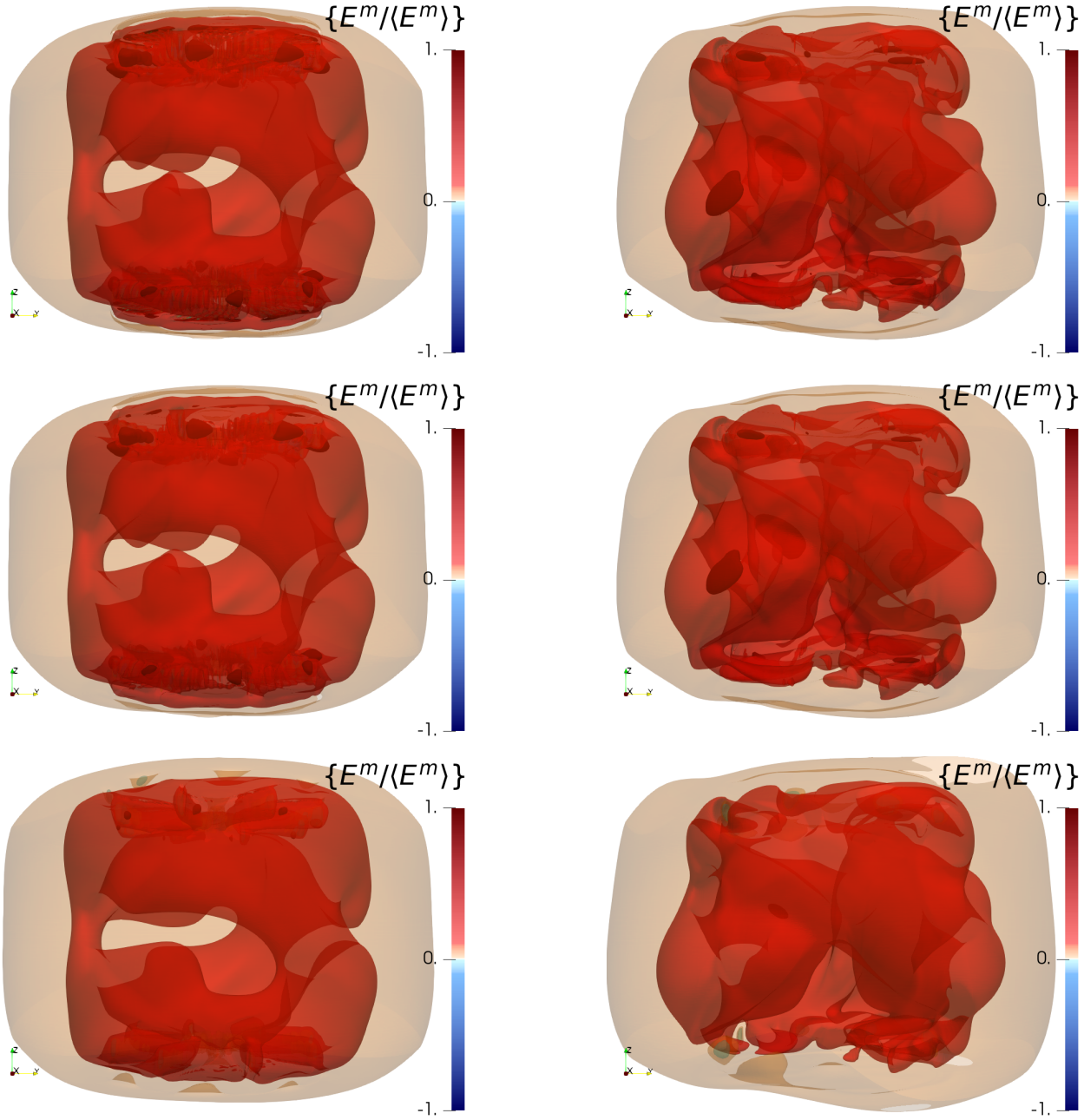


Figure 5.6: Renormalized and time-averaged magnetic energy in the growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

5.4 Energy dissipation in the growth regime

The viscous dissipation (fig. 5.7) is the same for both permeabilities and for each scale ℓ because the magnetic field is too weak to impact the velocity field in the growth phase. A small amount of dissipation occurs in the bulk at small scales but it is mainly concentrated near the impellers for all filtering scales.

The localization of the Joule dissipation (fig. 5.8) is different: for $\mu_r = 50$, $\{\mathcal{D}^\sigma/\langle E^m\rangle\}$ is dominant in the impellers for all filtering scales ℓ. For $\ell = 16\eta$, some negative regions appear near the lateral wall but the spatial average stays positive as indicated by figure 5.3; for $\mu_r = 1$, $\{\mathcal{D}^\sigma/\langle E^m\rangle\}$ shows complex structures for all ℓ. The Joule dissipation in the bulk is mainly positive for small ℓ while it becomes negative for large ℓ. However, an interesting feature appears in the shear layer and stays positive for all ℓ.

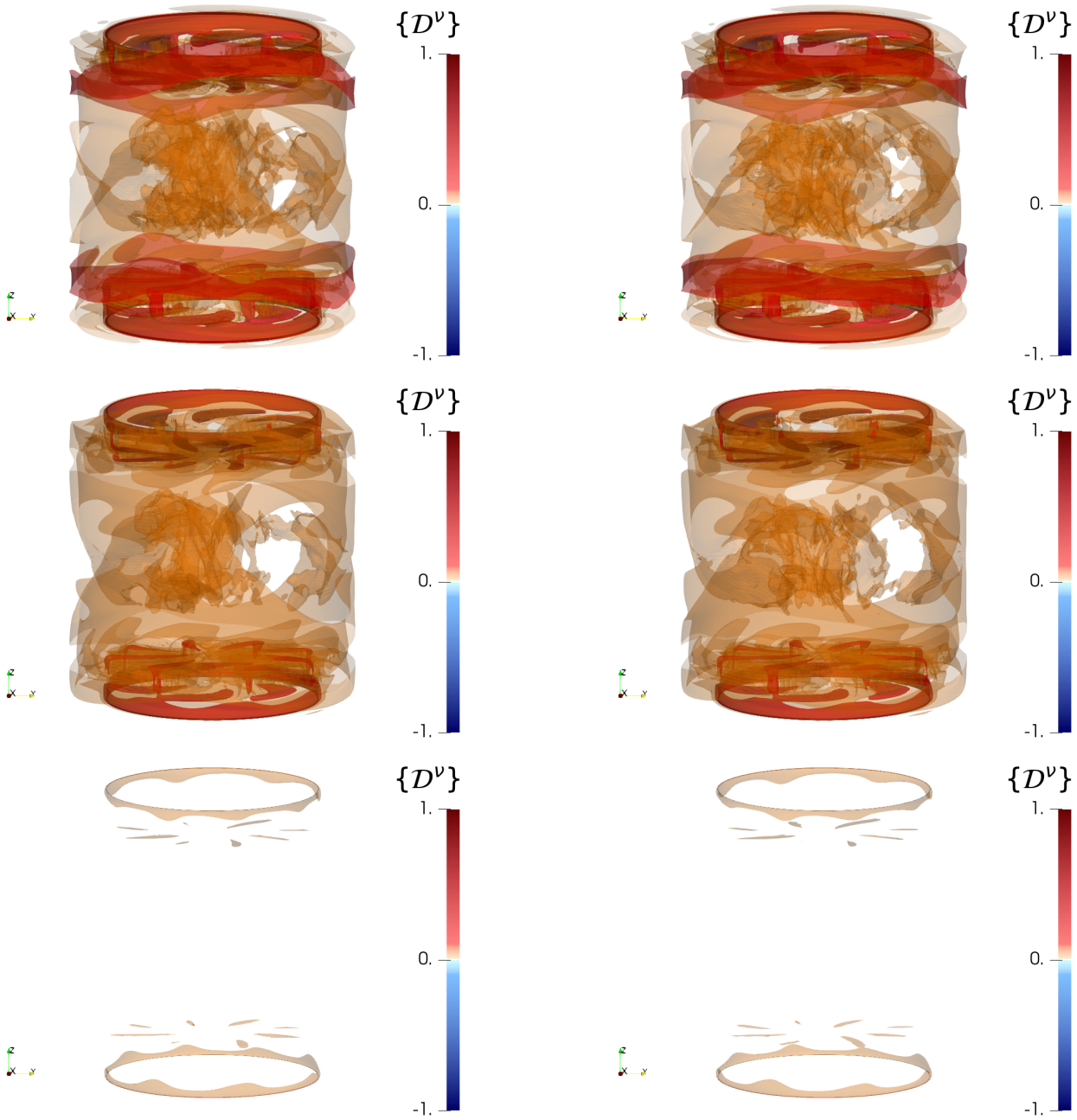


Figure 5.7: Time-averaged viscous dissipation in the growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

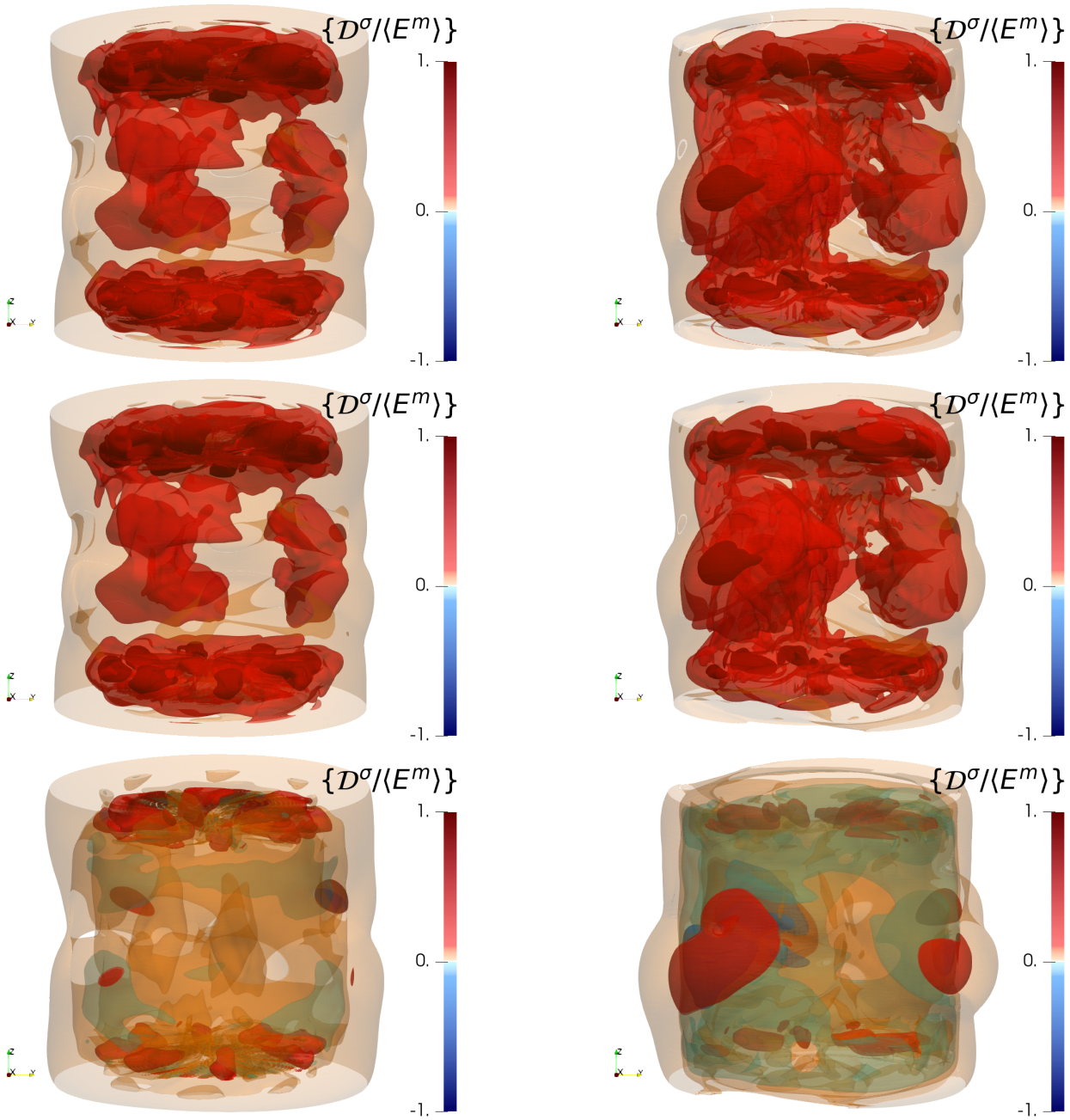


Figure 5.8: Renormalized and time-averaged Joule dissipation in the growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

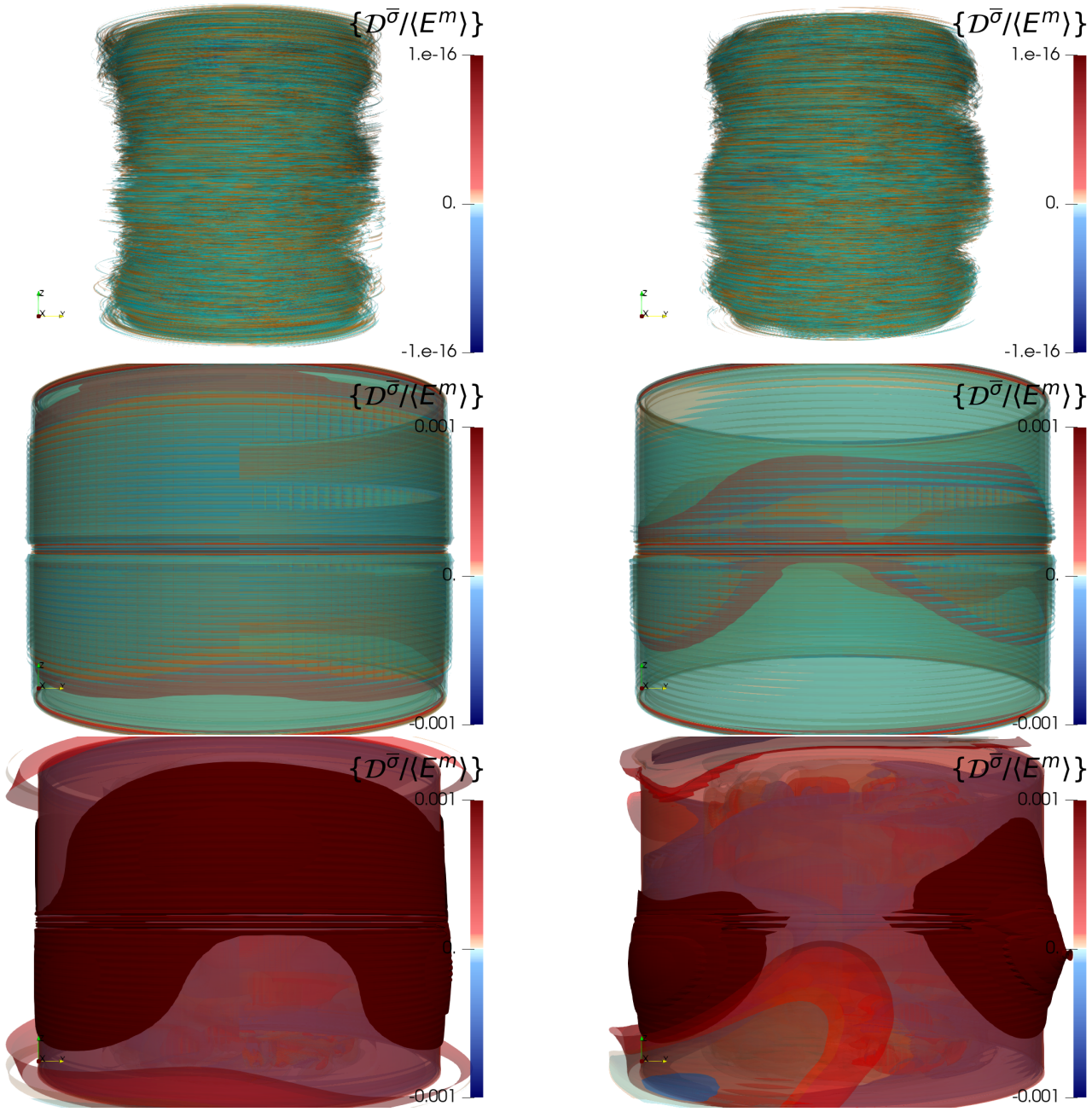


Figure 5.9: Renormalized and time-averaged irregular Joule dissipation in the growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

As shown in figure 5.9, the dissipation due to the irregularity of the current density $\{\mathcal{D}^{\overline{\sigma}}/\langle E^m\rangle\}$ is localized on the lateral wall where the conductivity is discontinuous, *i.e.*, $r = 1.4$, $0 \leq \theta < 2\pi$, $-1 \leq z \leq 1$. It is similar for both permeabilities for each scale and negligible.

5.5 Local energy transfers between scales in the growth regime

The anomalous dissipation (fig. 5.10) $\{\mathcal{D}^u/\langle E^m\rangle\}$ is similar for both permeabilities at each scale ℓ in the growth stage. At $\ell = 0$, it reaches the zero machine as expected. For $\ell = \eta$ and $\ell = 16\eta$, it is mainly positive in the bulk where large scales provide energy to small scales and it is negative near the impellers where small scales feed large scales through the motion of the impellers blades. This behavior highlights the energy cascade at work in the von Kármán flow, where energy is injected at the blades, and transferred towards smaller scales in the bulk of the flow.

In contrast, the anomalous magnetic dissipation $\{\mathcal{D}^b/\langle E^m\rangle\}$ (fig. 5.11) is very different for both per-

meabilities. For $\mu_r = 50$, for all ℓ, it is localized near the impellers and it becomes more negative when ℓ increases. Its spatial average is negative in this case as shown in figure 5.3 (left) which makes it an inverse energy cascade.

For $\mu_r = 1$, blobs of alternate signs are localized near the impellers while larger areas of positive $\{\mathcal{D}^b/\langle E^m\rangle\}$ are localized around the shear layer. The fact that $\{\mathcal{D}^b/\langle E^m\rangle\}$ is negative (pro-dynamo sign) near the impellers and not in the shear-layer is in agreement with the mean-field arguments based on helicity [9].

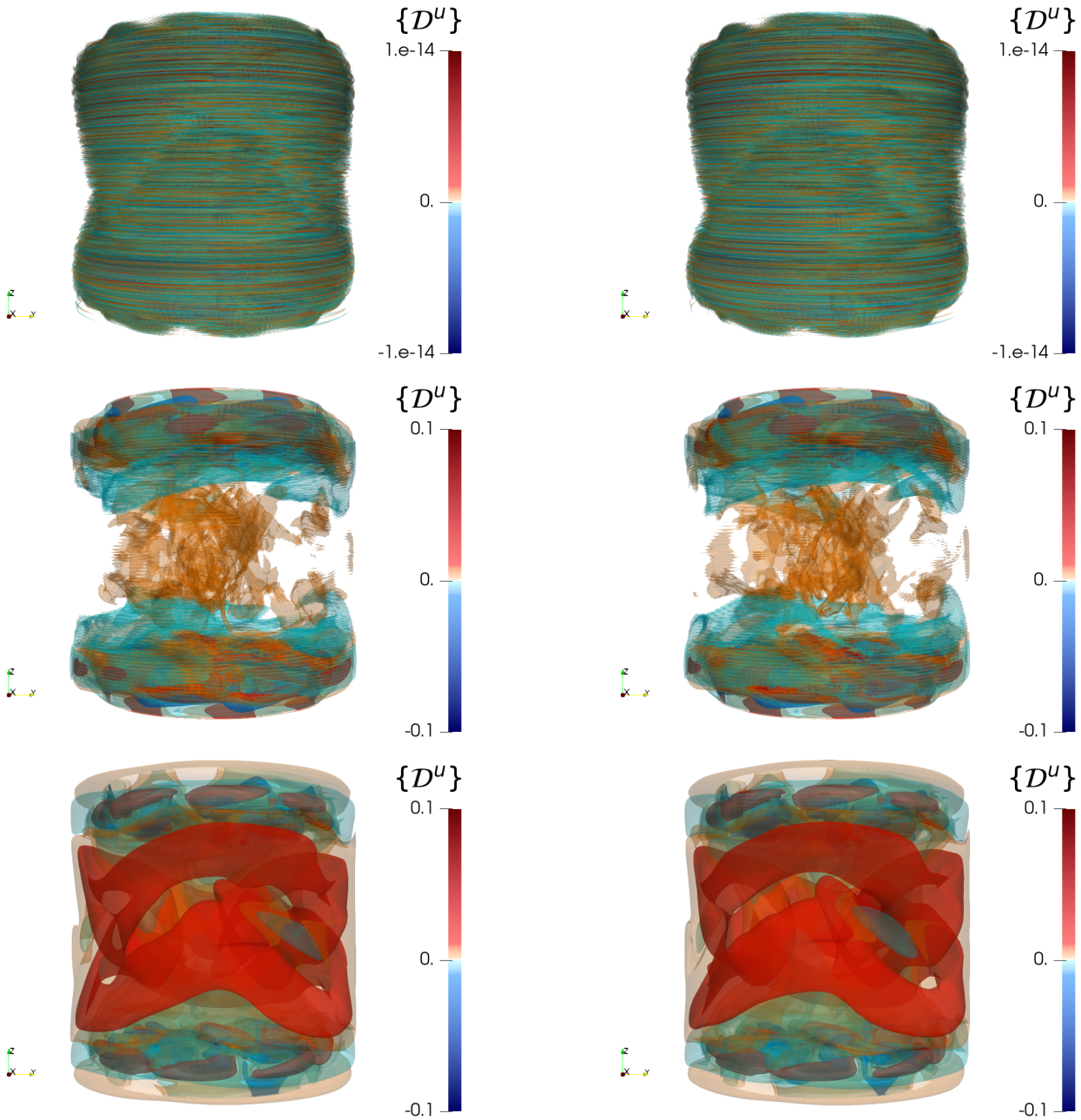


Figure 5.10: Time-averaged anomalous velocity dissipation in the growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

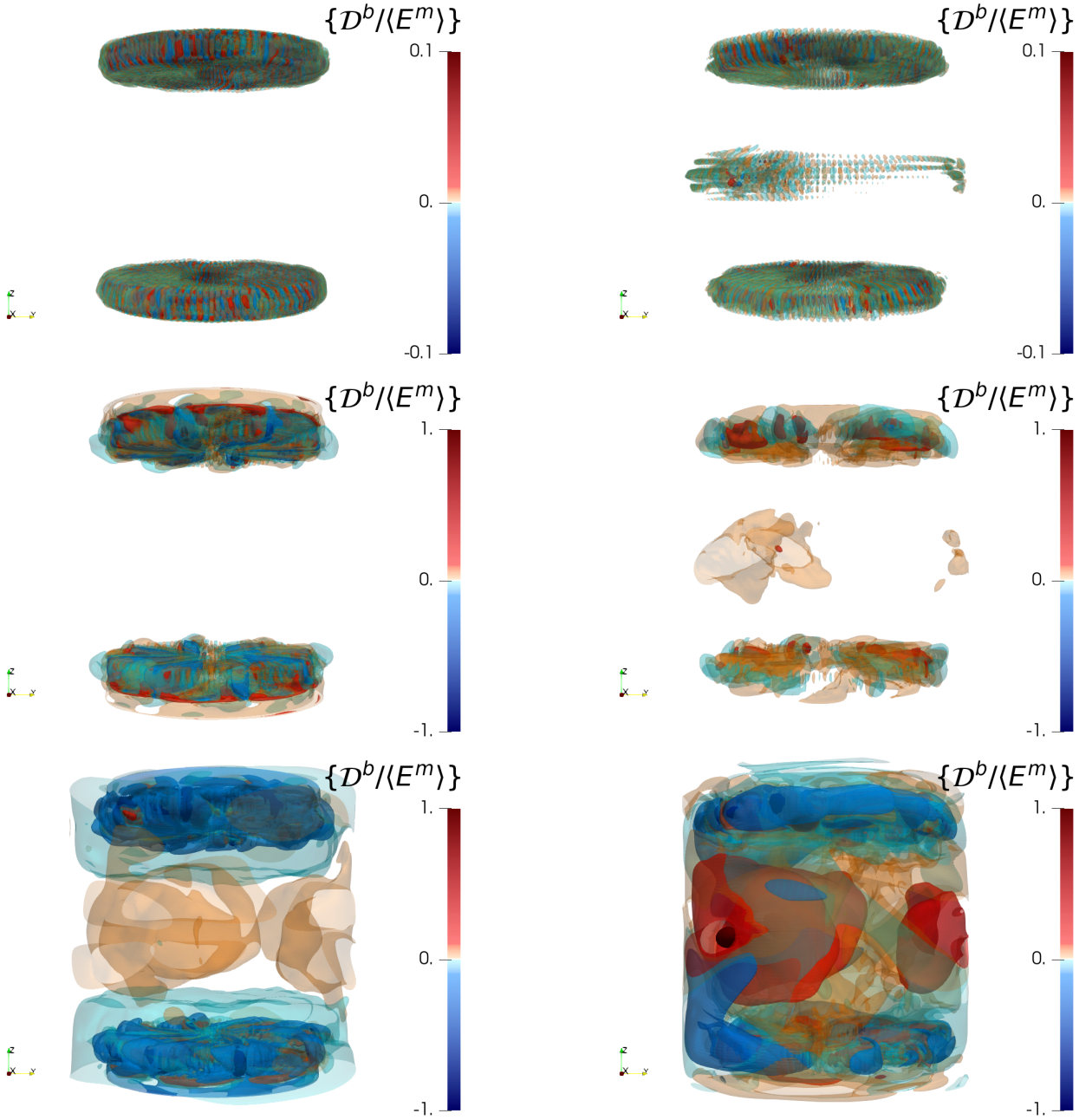


Figure 5.11: Renormalized and time-averaged anomalous magnetic dissipation in the growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

5.6 Conversion of energy in the growth regime

We now examine the two source terms already identified. At the early stage $t < 3$, the (minus) power of the Lorentz force (fig. 5.12) $\{\mathcal{T}^{u\to b}/\langle E^m\rangle\}$ is positive for both permeabilities, but beyond, its sign and spatial repartition are very different depending on the value of μ_r.

For $\mu_r = 1$, the shapes of $\{\mathcal{T}^{u\to b}/\langle E^m\rangle\}$ are similar for all ℓ: alternate signed zones near the impellers and positive areas in the bulk are visible. The latter contributes to a positive average which is compatible with the standard dynamo mechanisms linked with the presence of differential rotation and vortices in the shear layer and within the disks.

For $\mu_r = 50$ and all ℓ, we can observe two behaviors: at sufficiently small scales, it is positive within the shear layer, showing that the kinetic dynamo mechanism is still present. However, this mechanism

is countered by an anti-dynamo mechanism occurring near the impellers, where the term alternates sign such that its average is negative. Another term is required for the dynamo to be sustained: the rotating magnet effect $\{\mathcal{M}^{\mu}/\langle E^m\rangle\}$. As shown in figure 5.13, this magnet term is also only localized in the blades and disks with alternate signs, its averaged contribution being positive.

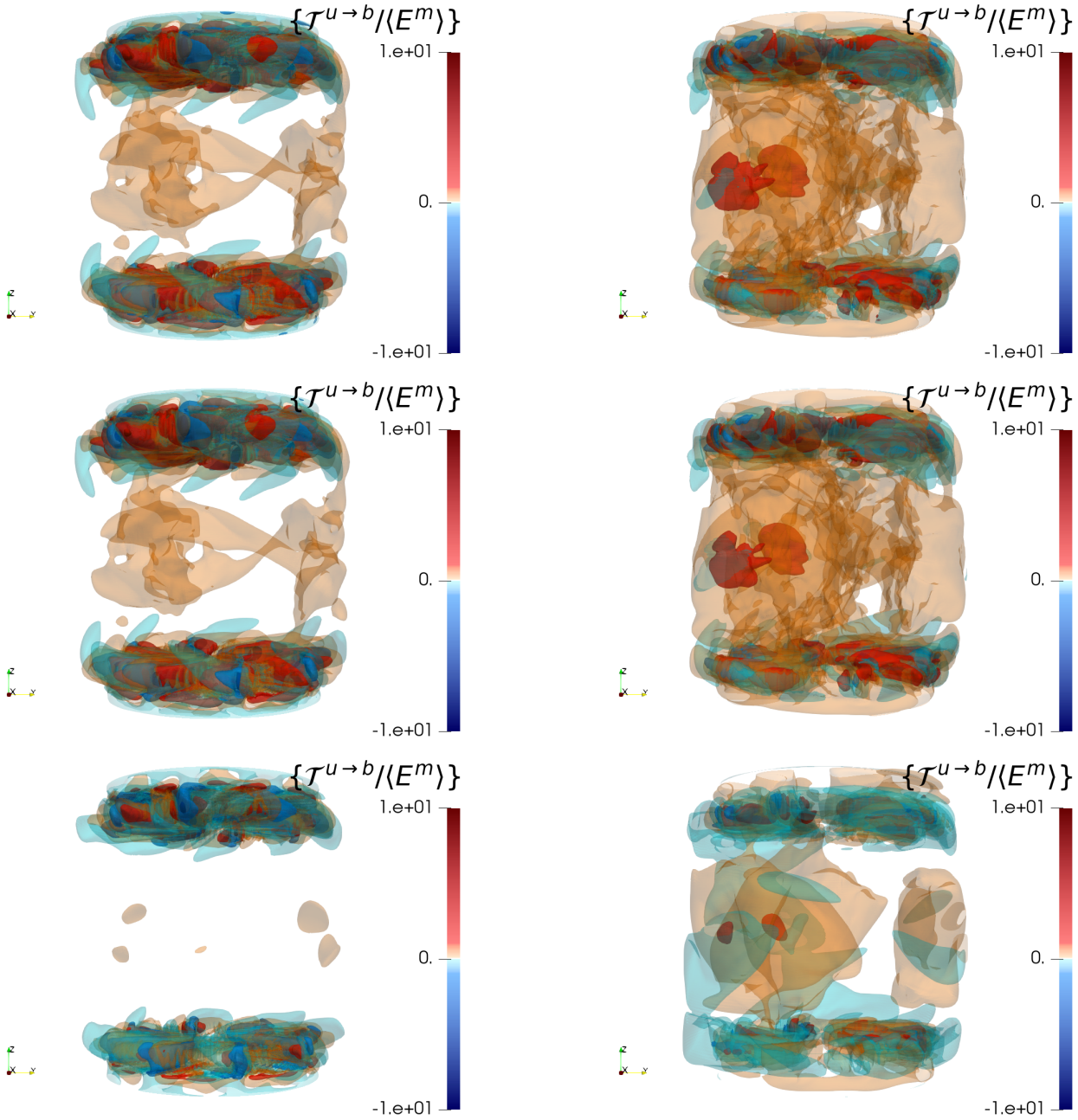


Figure 5.12: Renormalized and time-averaged energy transfer between fields in the growth regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

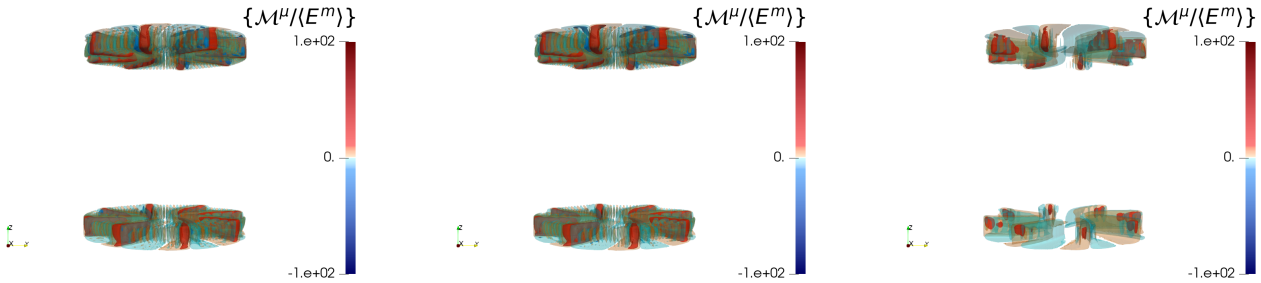


Figure 5.13: Renormalized and time-averaged rotating magnet effect for the case $R_e = 1500, R_m = 150, \mu_r = 50$ in the growth regime at different filtering scales: (left) $\frac{\ell}{\eta} = 0$, (middle) $\frac{\ell}{\eta} = 1$, (right) $\frac{\ell}{\eta} = 16$.

5.7 Dissipation and energy transfer through variation of magnetic permeability

Two dissipative terms are associated with the irregularity of the magnetic permeability μ_r and exist only for $\mu_r > 1$: $\{\mathcal{D}^{\mu\sigma}/\langle E^m \rangle\}$ (fig. 5.14) and $\{\mathcal{D}^{\mu\mathcal{T}}/\langle E^m \rangle\}$ (fig. 5.15). These two terms are therefore localized on the blades and disks on the impellers. They reach the zero machine as expected at $\ell = 0$. For $\ell > 0$, their shapes are similar for all ℓ. However, their magnitudes are different with $\{\mathcal{D}^{\mu\mathcal{T}}/\langle E^m \rangle\}$ being one order of magnitude greater than $\{\mathcal{D}^{\mu\sigma}/\langle E^m \rangle\}$. m

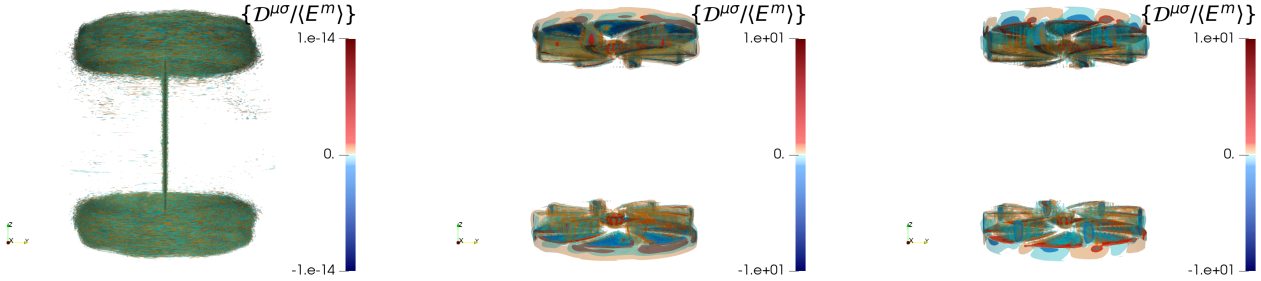


Figure 5.14: Renormalized and time-averaged Joule induced in-between scales magnetic transfers for the case $R_e = 1500, R_m = 150, \mu_r = 50$ in the growth regime.

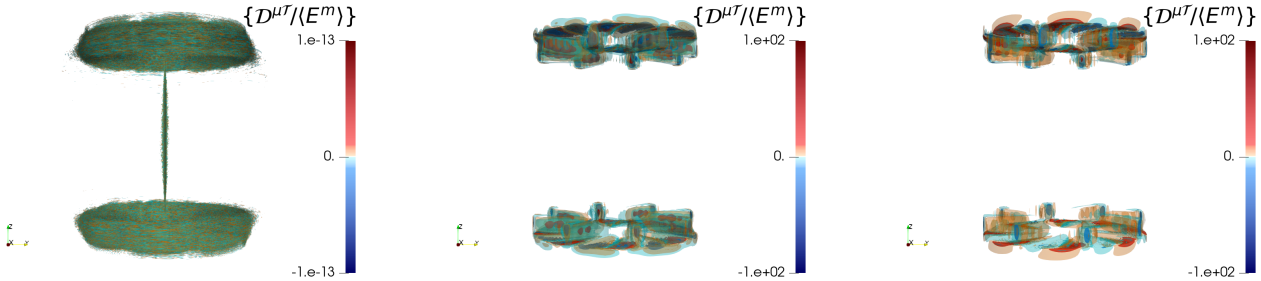


Figure 5.15: Renormalized and time-averaged Lorentz induced in-between scales magnetic transfers for the case $R_e = 1500, R_m = 150, \mu_r = 50$ in the growth regime.

6 Saturation of the dynamo from local energy budgets

To determine the main mechanisms for dynamo saturation (when $\partial_t \langle E^c \rangle = 0$ and $\partial_t \langle E^m \rangle = 0$ on average), we now consider the local energy budget given by (19)-(20) and evaluate the spatial average of all its components as a function of time. We want to determine what is the dominant balance. We next describe the spatial distribution of each term of the energy budget averaged on a time range belonging to the saturated regime for the two types of dynamo action.

6.1 Time evolution of energy terms in the dynamo saturated phase

Concerning the kinetic energy budget displayed in figure 6.1, for $\mu_r = 1$, $\mathcal{D}^\nu$ and $\mathcal{D}^u$ follow the same behavior as in the growth regime displayed in figure 5.1: $\mathcal{D}^\nu$ is positive and decreases with ℓ, $\mathcal{D}^u$ is positive and increases with ℓ. The transfer from the velocity field to the magnetic field is strong enough to be shown by the term $-\mathcal{T}^{u\to b}$ which is negative for all filtering scales. The irregular term $\mathcal{D}^b$ is positive and acts as a direct energy cascade. Consequently, the velocity field maintains the magnetic field via the $-\mathcal{T}^{u\to b}$ term and against all dissipative terms.

In contrast, for $\mu_r = 50$ in the saturated regime, the term $-\mathcal{T}^{u\to b}$ is positive and $\mathcal{D}^b$ is negative. Hence, $-\mathcal{T}^{u\to b}$ is a source term for the velocity field which extracts energy from the magnetic field and $\mathcal{D}^b$ is an inverse cascade. The other terms, $\mathcal{D}^\nu, \mathcal{D}^u$, are positive and behave like in the case $\mu_r = 1$.

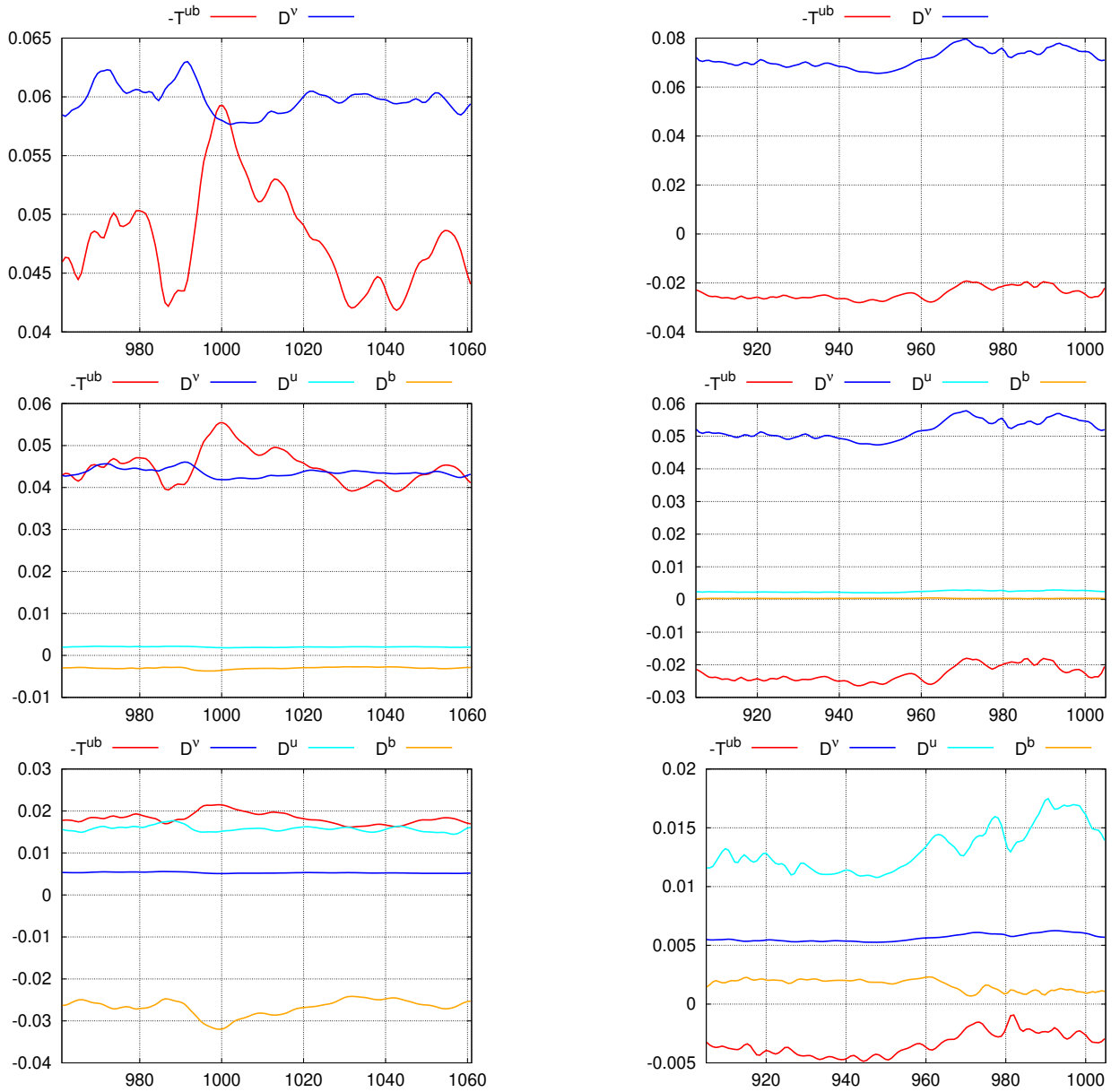


Figure 6.1: Time evolution of kinetic energy budget (19) in the saturated phase. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

Concerning the magnetic balance displayed in figure 6.2, the time evolution of $\mathcal{T}^{u\to b}, \mathcal{D}^\sigma, \mathcal{D}^b$ and $\mathcal{D}^{\overline{\sigma}}$ for $\mu_r = 1$ is similar to the one of the renormalized terms $\mathcal{T}^{u\to b}/\langle E^m\rangle, \mathcal{D}^\sigma/\langle E^m\rangle, \mathcal{D}^b/\langle E^m\rangle$ and $\mathcal{D}^{\overline{\sigma}}/\langle E^m\rangle$ shown in figure 5.3 (right column) except for the time interval $[970 : 990]$. In this interval,

$\mathcal{T}^{u\to b} - \mathcal{D}^b - \mathcal{D}^\sigma$ is negative and the magnetic energy decreases locally in time. Anyway, $\partial_t E^m$ is statistically around 0.

For $\mu_r = 50$, the time evolution of all terms of equation (20) follows that of the renormalized terms $\mathcal{M}^\mu/\langle E^m\rangle, \mathcal{T}^{u\to b}/\langle E^m\rangle, \mathcal{D}^{\mu\sigma}/\langle E^m\rangle, \mathcal{D}^b/\langle E^m\rangle, \mathcal{D}^{\mu\mathcal{T}}/\langle E^m\rangle$ of figure 5.3 (left column) with $\mathcal{D}^{\mu\mathcal{T}}$ larger than $\mathcal{D}^\sigma$. The terms $\mathcal{D}^{\bar{\sigma}}$ and $\mathcal{D}^{\mu\sigma}$ are negligible. The positive term $\mathcal{M}^\mu - \mathcal{D}^b$ compensates the negative term $\mathcal{T}^{u\to b} - \mathcal{D}^\sigma - \mathcal{D}^{\mu\mathcal{T}}$ for the magnetic field to be maintained.

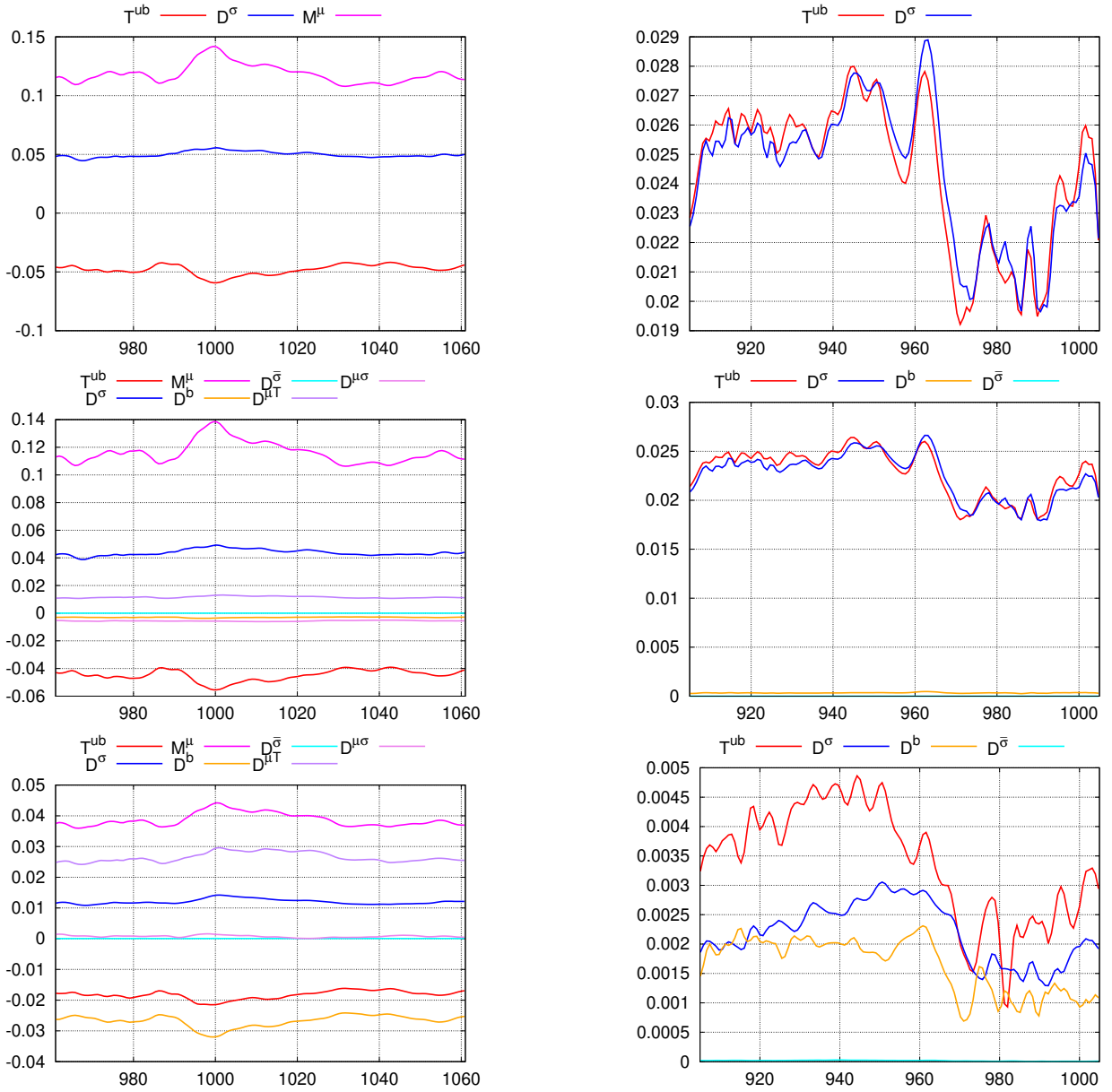


Figure 6.2: Time evolution of magnetic energy budget (20) in the saturated phase. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

6.2 Spatial distribution of the local growth rate in the saturated regime

Note that, in the saturated regime, the space-average of the local growth rate in the saturated regime $\langle\{\lambda^m\}\rangle$ is zero. However, we can look at its spatial distribution. For $\mu_r = 50$, the local growth rate is dominant in the neighborhood of the impellers. For $\mu_r = 1$, the localization is in three regions: the two impellers and the shear layer. Note that there is an asymmetry between the top and the bottom halves due to the event in the time range $[970 : 990]$ when the magnetic energy decreases. This event induces a bias in the time average of $\{\lambda^m\}$.

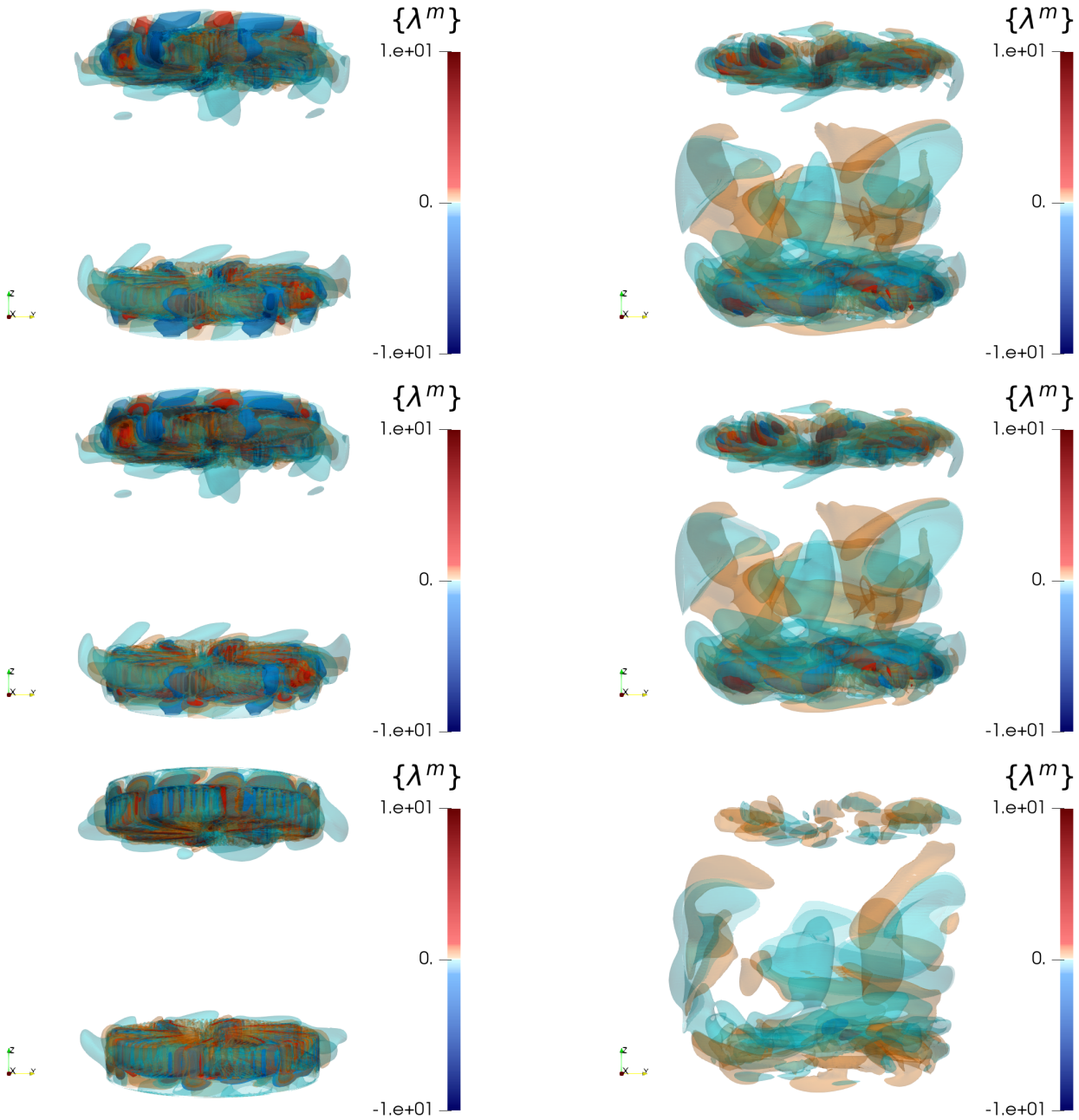


Figure 6.3: Dynamo growth rate averaged on the 128 snapshots in the dynamo saturated regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

6.3 Kinetic and magnetic energy repartition

We first examine the kinetic energy (figure 6.4). One can note that, for $\ell = 0$ and $\ell = \eta$, the kinetic energy is distributed almost identically in space. That means that the kinetic energy has typical scales larger than the Kolmogorov scale. However, it presents more fine structures in the bulk for the $\mu_r = 1$ case than for the $\mu_r = 50$ case.

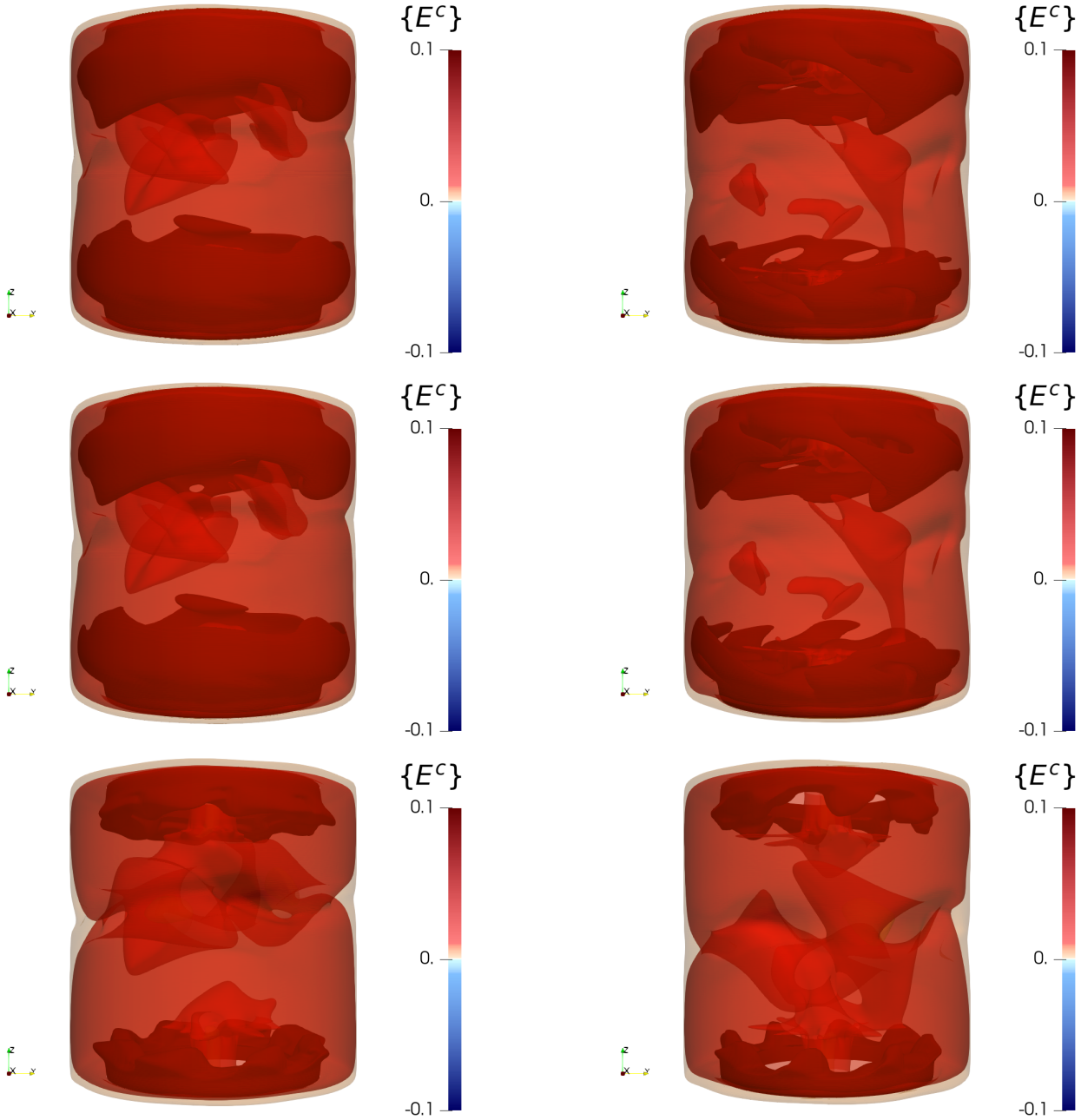


Figure 6.4: Kinetic energy averaged on the 128 snapshots in the dynamo saturated regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

The magnetic energy (figure 6.5) shows very different shapes depending on the relative magnetic permeability. For high μ_r, it is concentrated near the impellers for all filtering scales and is mainly axisymmetric. In contrast, at $\mu_r = 1$, the magnetic energy spreads across the entire volume and presents non axisymmetric features. These results are coherent with the dominant $m_F = 0$ magnetic mode for $\mu_r = 50$ and the mixing of $m_F = 0, 1$ magnetic modes for $\mu_r = 1$ as indicated in Table 3 of reference [22].

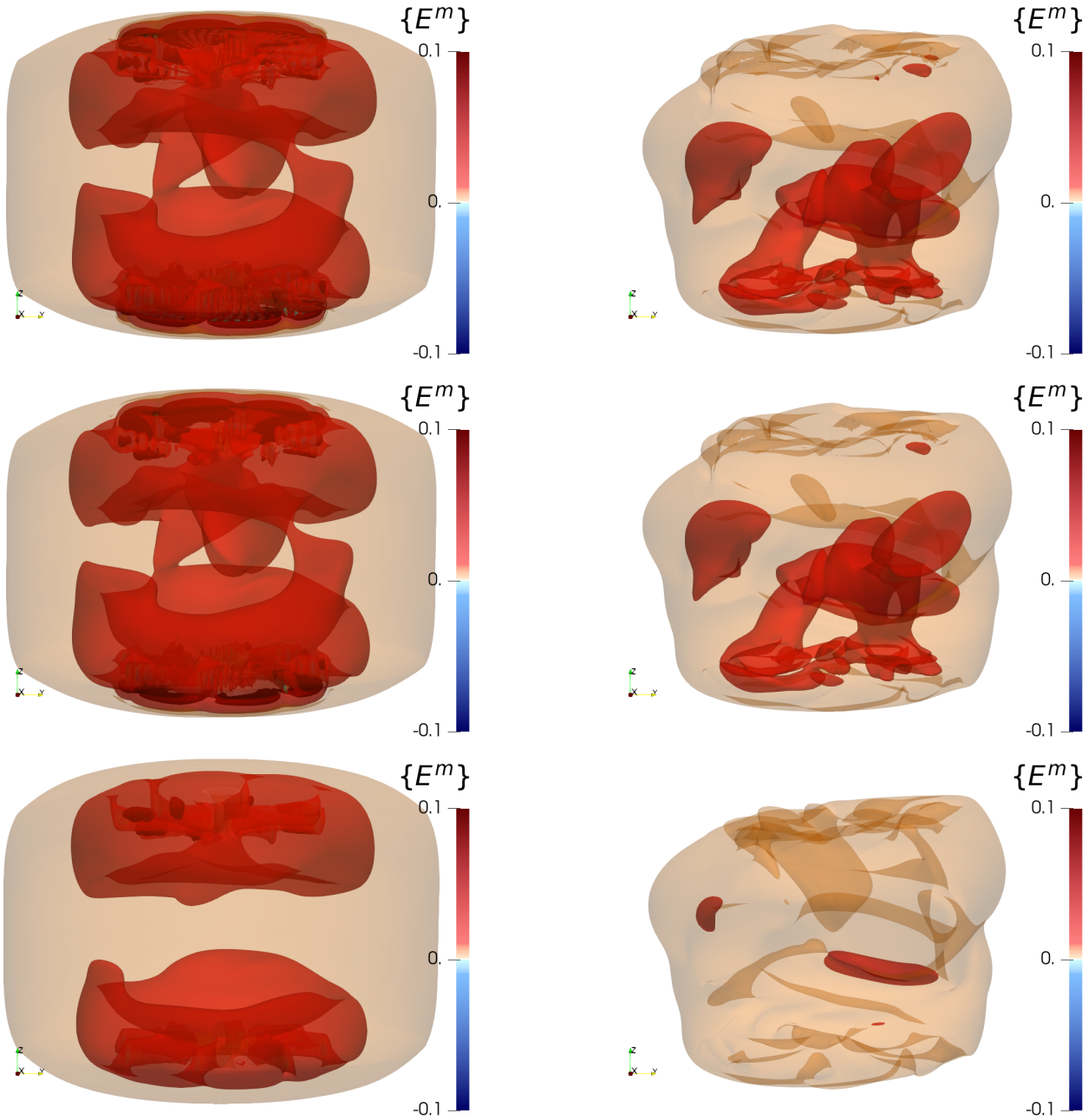


Figure 6.5: Time-averaged magnetic energy in the saturated regime. Cases are defined in figure 6.4. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

6.4 Energy dissipations

These terms refer to the dissipation of energy from usual phenomena such as viscous dissipation and Joule heating.

Most of the viscous dissipation happens near the blades and the rim of the disks which correspond to the areas with the greatest gradient of velocity (see figure 6.6). This dissipation is strictly positive everywhere for all ℓ, it does not really change with μ_r since the Reynolds number is identical.

In contrast, Joule dissipation depends strongly on μ_r: when $\mu_r = 50$, for all ℓ, it occurs mostly in the impellers and along the cylinder axis, indicating that electric current is concentrated there. This corresponds to the spatial distribution of the current lines shown in figure 22 of [22]. For $\ell = 16\eta$, a few negative spots appear between the blades.

When $\mu_r = 1$, Joule dissipation is distributed in three zones, the two impellers and the equatorial shear layer. For large scales ($\ell \gg \eta$), the filter reveals regions of negative values corresponding to areas where the resistivity smoothes the field: it transfers energy from small scales to larger ones.

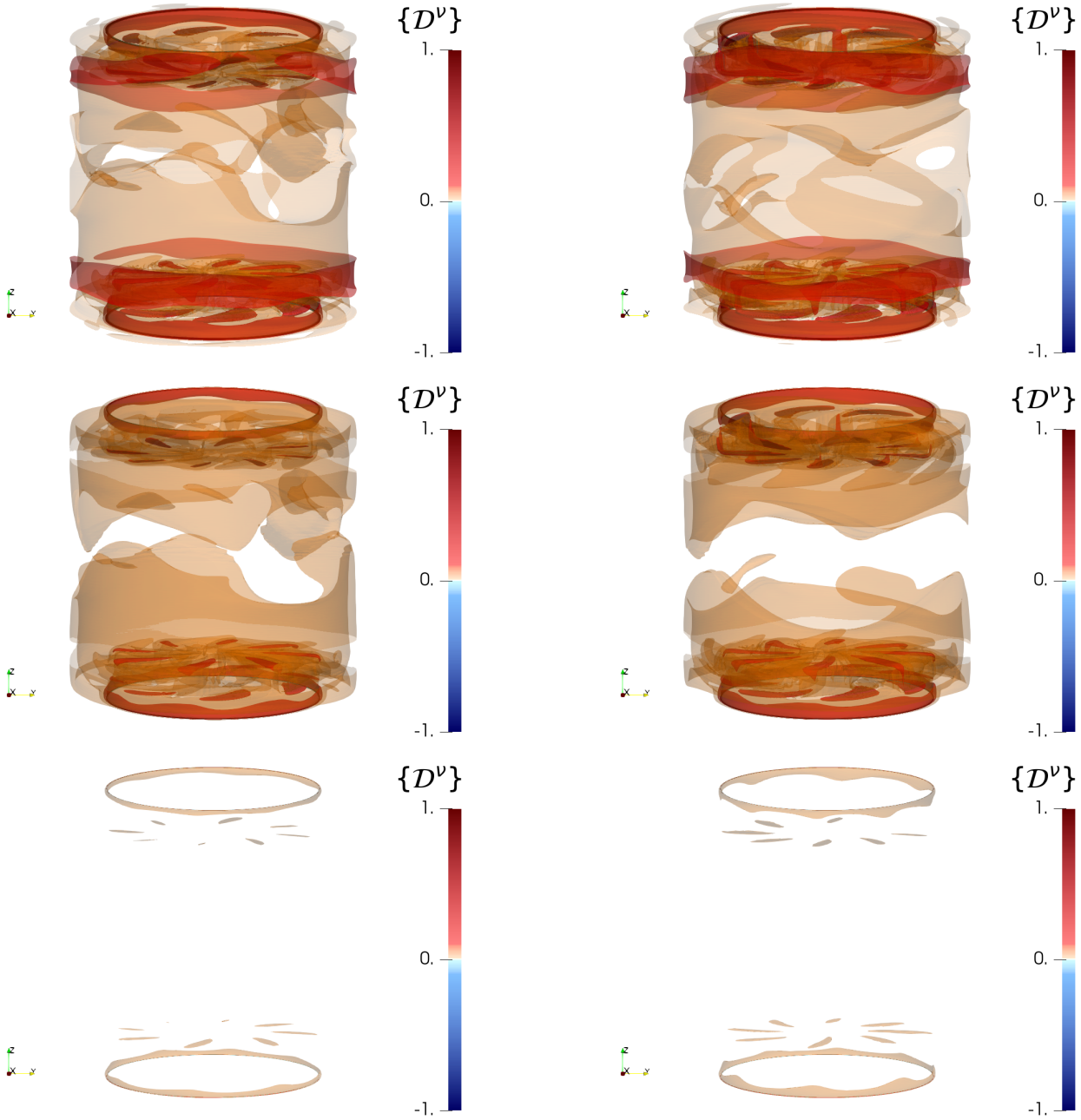


Figure 6.6: Time-averaged viscous dissipation in the saturated regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

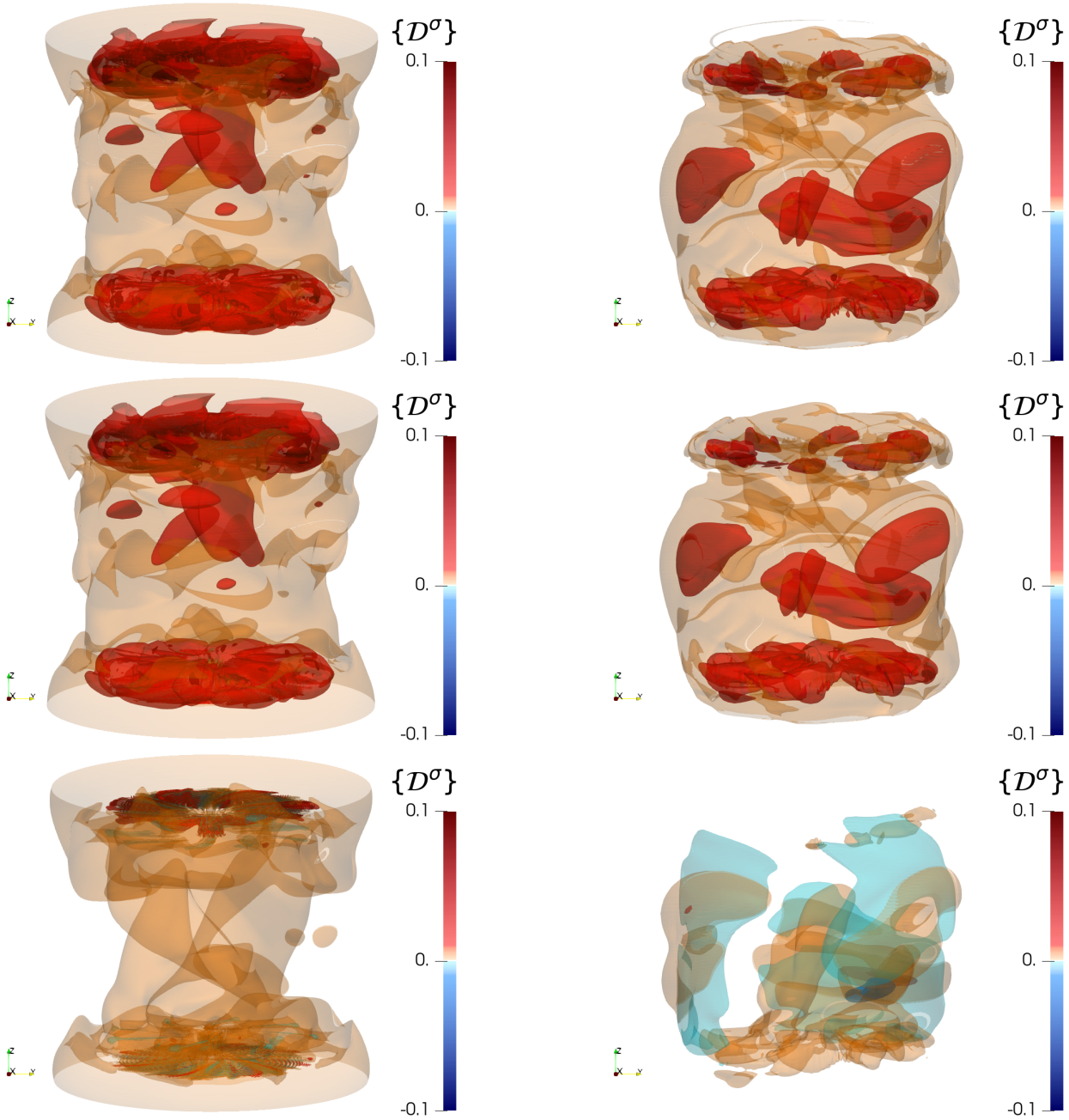


Figure 6.7: Time-averaged Joule dissipation in the saturated regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

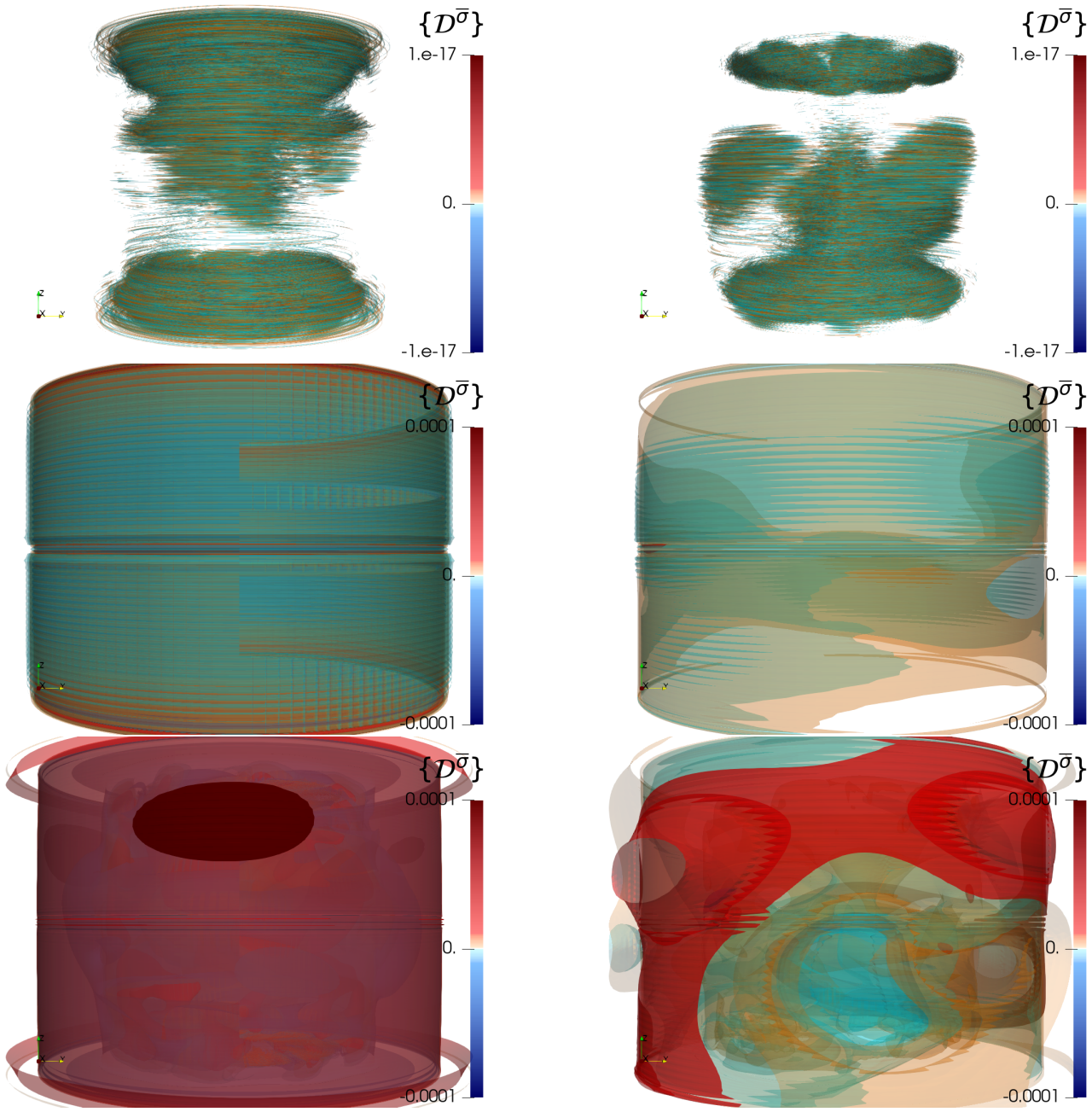


Figure 6.8: Time-averaged irregular Joule dissipation in the saturated regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

Figure 6.8 shows the Joule dissipation due to the irregularity of the electrical conductivity. As expected, it is located on the lateral border $\{r = 1.4,\ 0 \leq \theta < 2\pi,\ -1 \leq z \leq 1\}$ where the conductivity has a jump from the stagnant liquid sodium layer to the copper vessel. For all filter scales, it remains low (maximum absolute values around 10^{-4}) and is therefore not really relevant for the dynamo action.

6.5 Local energy transfers between scales

The most interesting feature of the filtering process is to highlight the energy transfer that happens in-between different scales via the $\mathcal{D}^u$ and $\mathcal{D}^b$ terms. These terms vanish in the limit $\ell \to 0$ in our simulation, but they may provide finite contributions for sufficiently rough velocity fields, resulting in anomalous dissipation and fast dynamo mechanism [23, 24].

Figure 6.9 displays $\{\mathcal{D}^u\}$. This term has only a kinetic origin linked to the Navier-Stokes equations [28]. In the inviscid limit, it may contribute to the non-vanishing dissipation observed in the von

Kármán flow [43, 44]. In our simulation, this term is zero at $\ell = 0$, as expected. For $\ell = \eta$, it is mainly localized near the impellers. For $\ell = 16\eta$, more negative regions appear. They correspond to areas where the scales smaller than 16η inject energy into the system. They are found near the impellers but also near the lateral wall, highligthing the non-homogeneity of the cascade and the importance of the boundaries in the transfer mechanism.

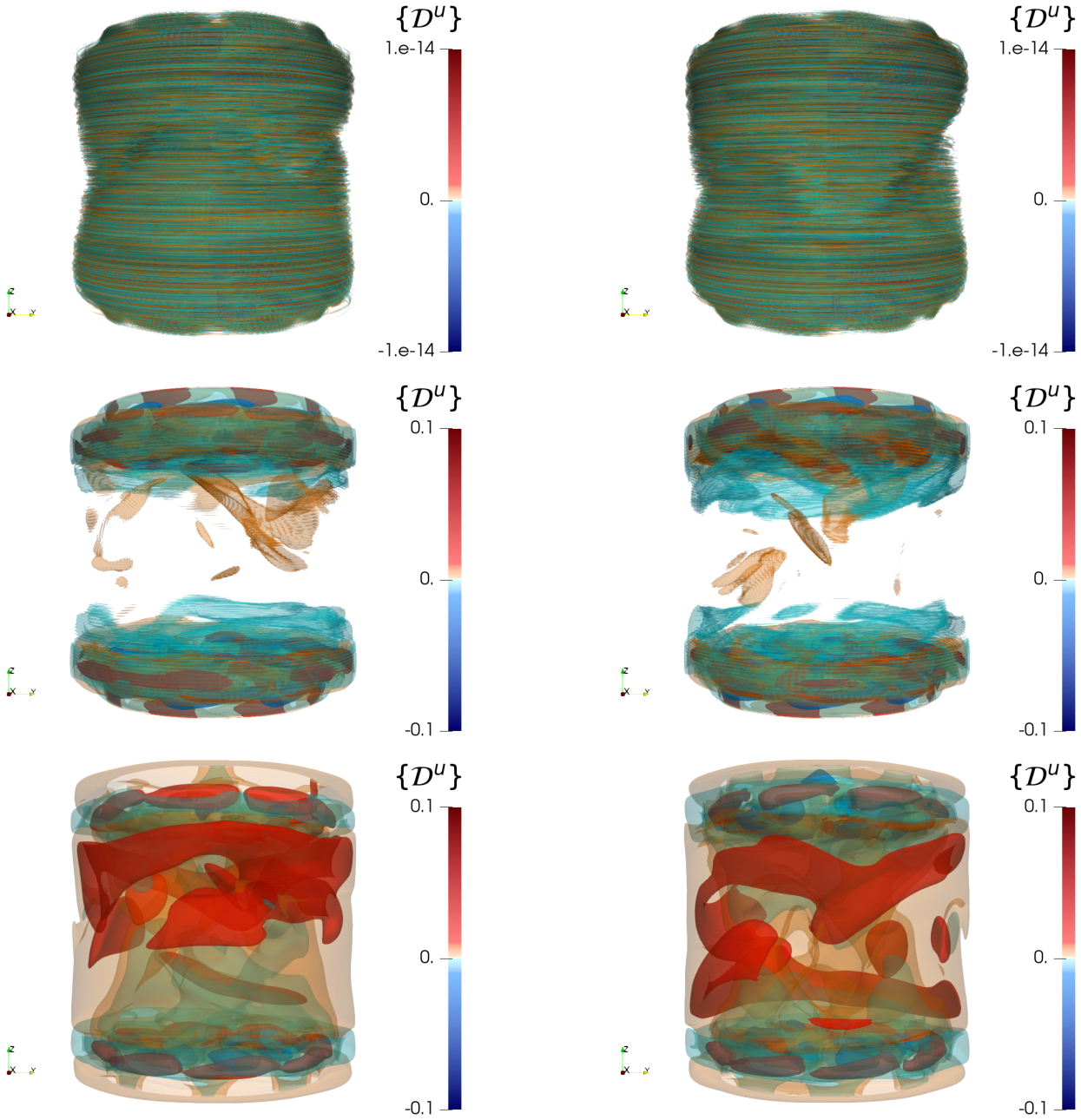


Figure 6.9: Time-averaged anomalous velocity dissipation in the saturated regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

Due to the presence of the magnetic induction, a new transfer channel is expected allowing transfers from kinetic to magnetic energy through the term $\{\mathcal{D}^b\}$, that can provide a fast dynamo mechanism in the inviscid limit [30], for rough enough velocity fields. As seen in figure 6.10, this term is dominantly localized near the impellers for $\mu_r = 50$. Patterns with alternate signs are stuck to the blades but their spatial average is negative as shown in figure 6.2 (bottom left column panel). That means that an inverse cascade involving the velocity and magnetic fields exists from small scales to large scales near the impellers.

Similarly, for $\mu_r = 1$, spots of alternate signs are close to the impellers, but complex positive structures also form close to the shear layer as ℓ increases. These positive regions are dominant and contribute to an overall positive spatial average, as shown in figure 6.2 (bottom right column panel). In this case, the cascade involving the velocity and magnetic fields is direct from large scales to small scales.

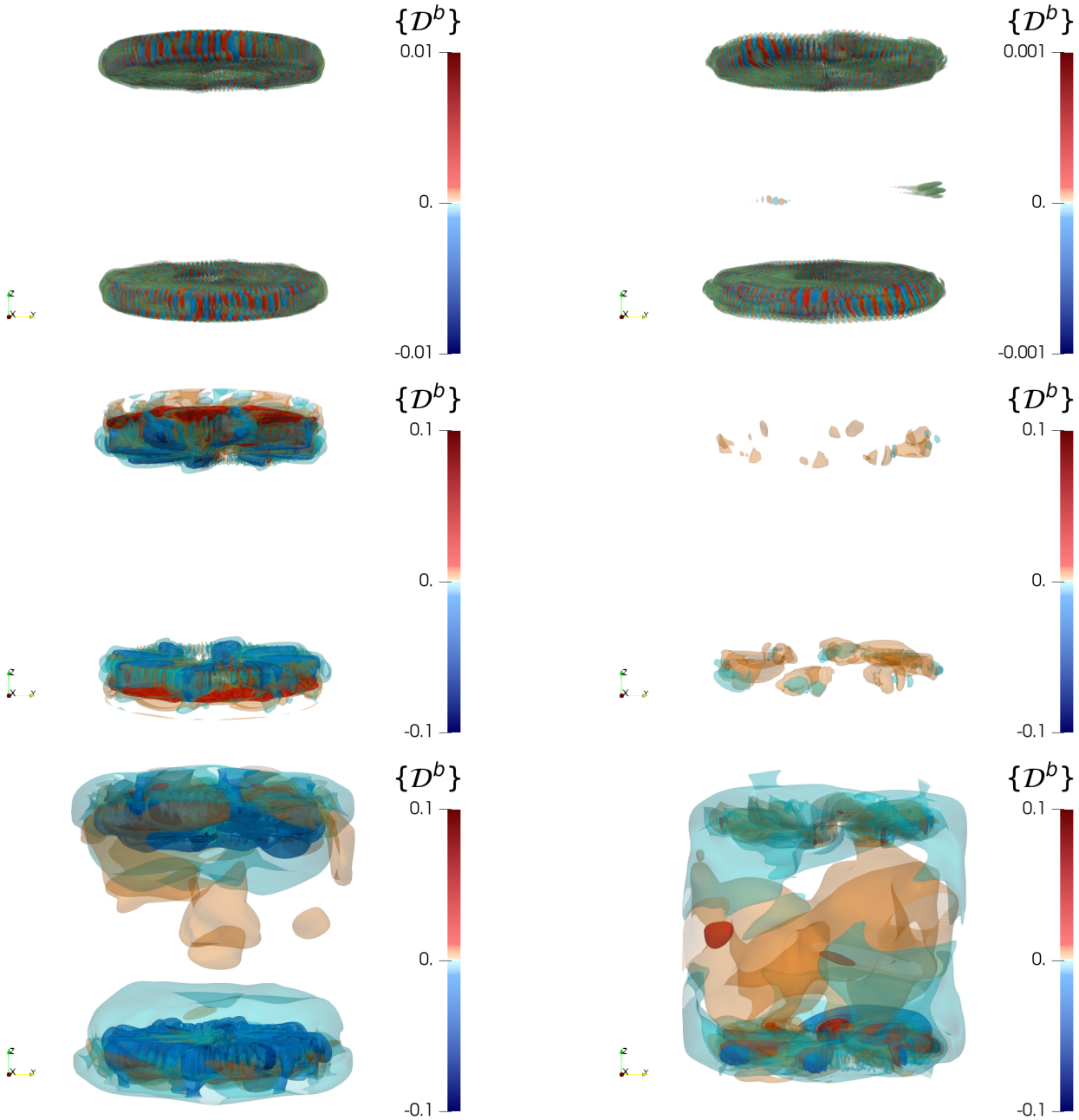


Figure 6.10: Time-averaged anomalous magnetic dissipation in the saturated regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

6.6 Conversion of energy

The conversion of kinetic energy to magnetic energy relies on the $\mathcal{T}^{u\to b}$ and $\mathcal{M}^{\mu}$ terms (as seen in equation (20)). The former is the standard source term: when it is on average positive, the velocity field provides energy to the magnetic field; the opposite occurs when it is negative.

Figure 6.11 shows that the spatial variation of $\{\mathcal{T}^{u\to b}\}$ depends on the value of μ_r. For $\mu_r = 50$, it is localized near the impellers with alternate signs. Its spatial average is negative in the saturated regime as shown in figure 6.2 (left column). It corresponds to a loss of magnetic energy. For $\mu_r = 1$, there

are more transfers in the bulk and smaller values in the impellers. Note that it is the only source term for the maintenance of a magnetic field when $\mu_r = 1$. Therefore it has to be positive on average for a magnetic field to saturate as it is seen in figure 6.2 (right column).

In contrast, for $\mu_r = 50$, $\{\mathcal{M}^\mu\}$ is another source of magnetic field increase. This term only exists where the blades rotate as expected, and it clearly follows the azimuthal variation of the blades with $m_F = 8$. It is on average positive and greater than the negative average of $\{\mathcal{T}^{u\to b}\}$ (see figure 6.2 left column). This allows qualifying this dynamo as a magnet effect in the saturated regime. However, at the beginning of the linear regime, it is dominated by the $\{\mathcal{T}^{u\to b}\}$ term as shown in figure 5.3 for $t < 3$.

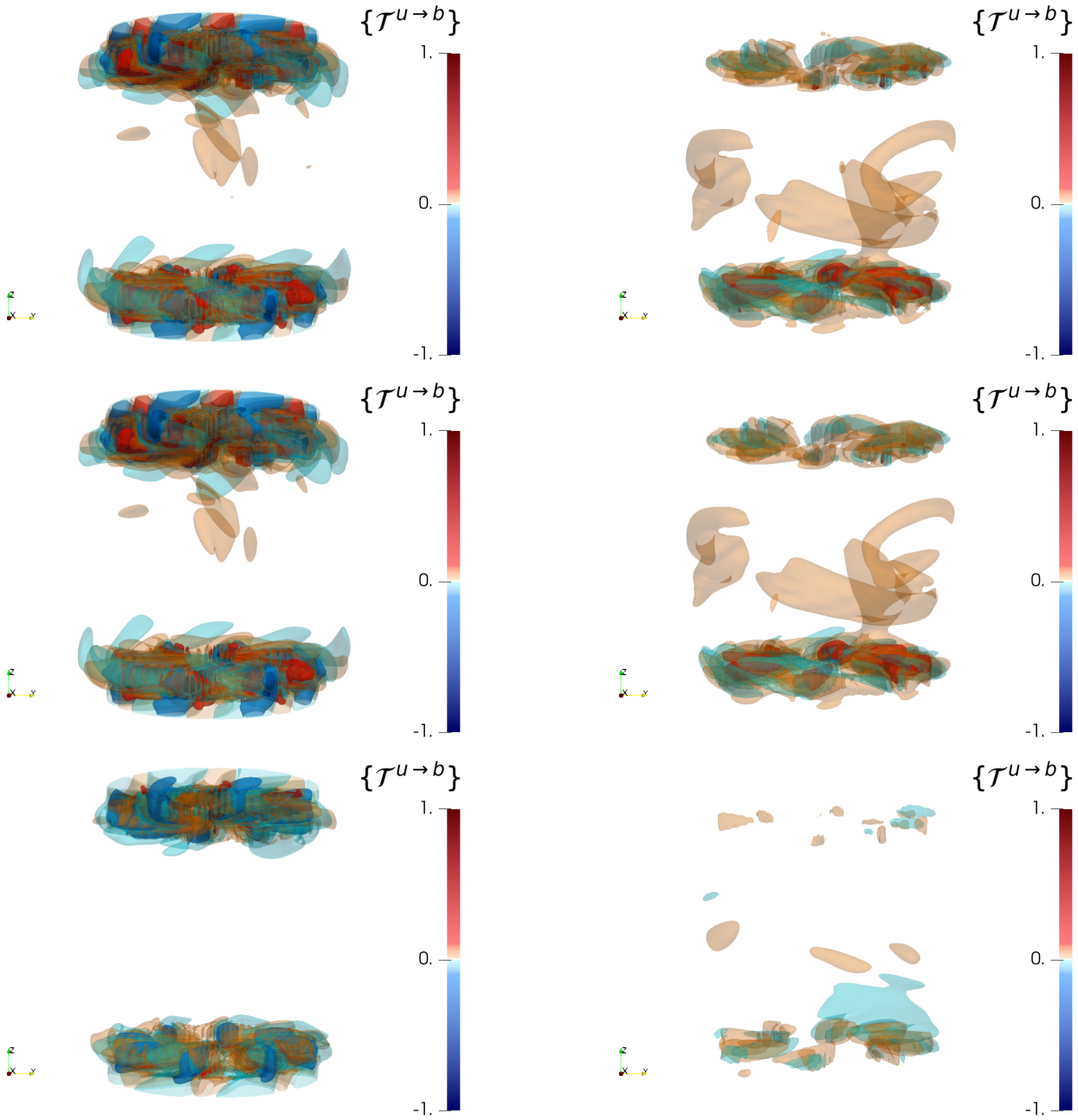


Figure 6.11: Time-averaged energy transfer between fields in the saturated regime. Panels correspond to $\frac{\ell}{\eta} = 0$ for the first row, $\frac{\ell}{\eta} = 1$ for the second row, $\frac{\ell}{\eta} = 16$ for the third row, and the case $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ on the first column and the case $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ on the second column.

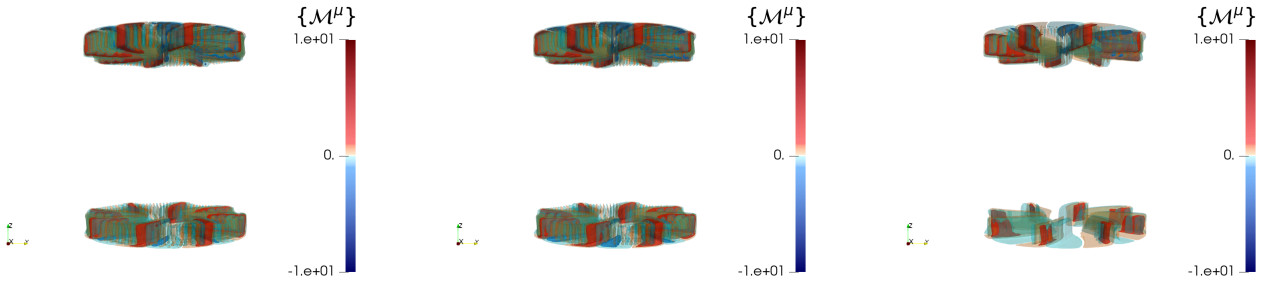


Figure 6.12: Time-averaged rotating magnet effect for the case $R_e = 1500, R_m = 150, \mu_r = 50$ in the saturated regime at different filtering scales: (left) $\frac{\ell}{\eta} = 0$, (middle) $\frac{\ell}{\eta} = 1$, (right) $\frac{\ell}{\eta} = 16$.

6.7 Dissipation and energy transfer through variation of magnetic permeability

The variation of μ_r due to the impellers is responsible for the terms $\{\mathcal{D}^{\mu\sigma}\}$ and $\{\mathcal{D}^{\mu\mathcal{T}}\}$. These are localized at the surface of the blades and disks. They exist only for $\ell > 0$ and show alternate signs. They are specific energy transfers between scales linked to jumps in magnetic permeability. The first one is induced by the Joule dissipation while the second one is induced by the Lorentz force power.

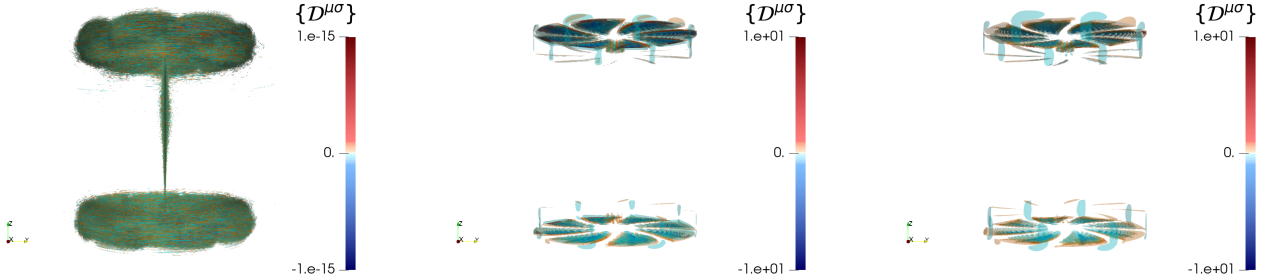


Figure 6.13: Time-averaged Joule induced in-between scales magnetic transfers for the case $R_e = 1500, R_m = 150, \mu_r = 50$ in the saturated regime.

Figure 6.14: Time-averaged Lorentz induced in-between scales magnetic transfers for the case $R_e = 1500, R_m = 150, \mu_r = 50$ in the saturated regime.

7 Mechanisms for dynamo action

Inspection of the linear regime has proved that the standard mechanism for the growth of the dynamo acts for both values of the impellers permeability at early instants: as soon as the threshold in magnetic Reynolds number is exceeded, the (minus) Lorentz force power overrides the dissipative terms. For steel impellers ($\mu_r = 1$), at larger times, a balance between $\mathcal{T}^{u\to b}$ (which is in space-average positive) and the dissipative terms (whose space-average values are positive) is reached and allows a saturated regime to be achieved. In contrast, for soft iron impellers ($\mu_r = 50$), the rotating magnet effect due to the motion of μ_r and the anomalous magnetic dissipation (which is negative on space-average) take over, and become the key players for the dynamo mechanism in the saturation phase. In light of our findings, we can now discuss the different scenarios proposed in the literature to explain the VKS dynamo.

The mechanism based upon spatial variations in electrical conductivity has first been proposed by [13]. The authors have considered the magnetic field generation by an electrically conducting fluid flowing parallel to a rigid plate presenting spatial variations in the streamwise direction. It was shown that the magnetic Reynolds number for dynamo process decreases with increasing conductivity of the plate and with increasing amplitude of its modulation. The growth mechanism is the coupling between poloidal and toroidal components of the magnetic field through the high conductivity zones. This mechanism was also recovered in [14, 15].

Spatial variations in magnetic permeability have been already put forward as a source of dynamo action [16, 17, 18, 19]. In [16], by combining the effects of the soft-iron impellers and a small uniform homogeneous $\alpha_{\theta\theta}$-effect modeling the induction effects of unresolved small-scale flow fluctuations, an axisymmetric magnetic field mode was obtained in a kinematic dynamo setup. However, a uniform homogeneous α-effect is not realistic as shown in figure 4.2. In [17], the growth of the magnetic field is due to the shear of the velocity localized on the frontier between fluid and the top of the blades. The jump conditions of section 2.2 (iii) and (iv) in [17] can be interpreted as a $\mathcal{M}^\mu$ effect localized only on the blades' top. In our case, the helical flow in between the blades distorts the magnetic field and bends it along the surface of the blades and disks making the $\mathcal{M}^\mu$ term large there (as shown in figure 6.12). Therefore it is not the dynamo mechanism described in [17] as it would occur only on the top of the blades. In [18], the solid body rotation of an anisotropic conductive or an anisotropic permeable cylindrical rotor (the anisotropy of material properties follows a logarithmic spiral shape) immersed in a static cylinder can generate an axisymmetric magnetic field. The kinematic dynamo mechanism there is due to differential rotation coupled with anisotropic diffusion. However, the generated magnetic field has a horizontal structure following spirals and a Fourier vertical structure as $\exp(ikz)$ with k the vertical wavenumber of the corresponding eigenmode, far from the VKS dynamo mode shown in figure 4.1 (right). In [19], the effect of the rotating disk is studied: by adding opposite $\alpha_{\theta\theta}$ and Ω effect above and under the disk with an higher magnetic permeability than the liquid, a decrease of the magnetic threshold is observed. However, an $\alpha_{\theta\theta}$-effect with opposite sign above and under the disk is not what is observed in figure 4.2. Moreover, any α-or-Ω-driven dynamo mechanism would only participate through $\mathcal{T}^{u\to b}$ which is globally negative in the current case $\mu_r = 50$.

Here, the main dynamo mechanism at large times for $\mu_r = 50$ is simply a self-amplifying phenomenon due to the variation of the magnetic permeability in time. Re-writing (20) by keeping only $\mathcal{M}^\mu$ gives:

$$\partial_t E^m \sim E^m \partial_t \frac{1}{\mu}. \tag{46}$$

This implies that the magnetic field grows (resp. decreases) behind (resp. in front of) the blades as can be seen in figure 6.12, which amplifies the phenomenon: stronger $\mathcal{M}^\mu$ happens where the magnetic field is amplified. Hence, for soft-iron impellers ($\mu_r = 50$), the dynamo effect is due to rotating magnets. In the saturated regime, this is counterbalanced by both the Joule dissipation, $\mathcal{D}^\sigma$, and the energy transfer toward the velocity field, $-\mathcal{T}^{u\to b}$, (see figure 6.2).

The dynamo mechanism for steel impellers ($\mu_r = 1$) is very different as it is only due to the energy transfer from the velocity field. At the same time, it couples dynamo processes in the shear layer and the in-between blades zones (see figure 5.12).

8 Conclusion

The main result of this article is the expression on the different local transfer terms appearing in equations (19)-(20). They express all possible exchanges between velocity and magnetic fields. Obtaining these terms was made possible by the development of a new filtering approach that can be used effectively for unstructured grids such as the finite element mesh used in SFEMaNS. We believe that this filtering method offers a general framework for the analysis of data obtained on unstructured meshes.

The viscous $\mathcal{D}^\nu$ and resistive $\mathcal{D}^\sigma$ dissipations correspond to the classical terms. The lack of smoothness

of the solutions leads to the appearance of $\mathcal{D}^b$ that generalizes the anomalous dissipation $\mathcal{D}^u$ [23] (which quantifies the kinetic energy dissipated by non-viscous means) to the anomalous magnetic energy dissipation. It is rather remarkable that the combination of these anomalous dissipative terms can be written in the form of structure functions already derived in the literature [24, 26] (as shown in appendices C-D).

The novelty is that the separation of the kinetic and magnetic energy equations allows highlighting the main terms responsible for the dynamo effect, namely $\mathcal{T}^{u\to b}$ and $\mathcal{M}^{\mu}$. Using two illustrative numerical dynamo simulations with different impellers materials, we were able to show that the dynamo mechanisms are totally different when using steel or soft iron impellers. In the case of steel impellers, the only source term is the (minus) Lorentz power $\mathcal{T}^{u\to b}$ that needs to compensate for all dissipative terms. In the case of soft iron impellers, the ferromagnetic nature of the impellers plays a crucial role in maintaining the magnetic field through $\mathcal{M}^{\mu}$ (and $\mathcal{D}^b$ for a filtering scale $\ell > 0$). This second mechanism cannot be described by a mean-field approach, which does not include variations in magnetic permeability nor electrical conductivity.

MHD and Navier-Stokes equations share a lot in common. However, from a mathematical perspective, it is unknown whether the viscous and resistive mechanisms that tend to smooth out potential irregularities in the velocity and magnetic field are effective enough to constrain the fields to stay smooth at all times. The analytical tools we have developed will allow us to dynamically study the rare events associated with extreme values of the anomalous dissipative terms in future work.

9 Acknowledgement

The HPC resources for SFEMaNS were provided by GENCI-IDRIS (grant 2022-0254) in France.

References

[1] H. Moffatt, *Magnetic Field Generation in Electrically Conducting Fluids*. Cambridge University Press, 1978.

[2] F. Krause and K.-H. Rädler, *Topological Aspects of the Dynamics of Fluids and Plasmas*. Oxford, Pergamon Press, 1980.

[3] R. Stieglitz and U. Müller, "Experimental demonstration of a homogeneous two-scale dynamo," *Physics of Fluids*, vol. 13, no. 3, pp. 561–564, 2001.

[4] A. Gailitis, O. Lielausis, E. Platacis, S. Dement'ev, A. Cifersons, G. Gerbeth, T. Gundrum, F. Stefani, M. Christen, and G. Will, "Magnetic field saturation in the Riga dynamo experiment," *Phys. Rev. Lett.*, vol. 86, pp. 3024–3027, Apr 2001.

[5] R. Monchaux, M. Berhanu, M. Bourgoin, M. Moulin, P. Odier, J.-F. Pinton, R. Volk, S. Fauve, N. Mordant, F. Pétrélis, A. Chiffaudel, F. Daviaud, B. Dubrulle, C. Gasquet, L. Marié, and F. Ravelet, "Generation of a magnetic field by dynamo action in a turbulent flow of liquid sodium," *Phys. Rev. Lett.*, vol. 98, p. 044502, Jan 2007.

[6] S. Miralles, N. Bonnefoy, M. Bourgoin, P. Odier, J.-F. m. c. Pinton, N. Plihon, G. Verhille, J. Boisson, F. m. c. Daviaud, and B. Dubrulle, "Dynamo threshold detection in the von kármán sodium experiment," *Phys. Rev. E*, vol. 88, p. 013002, Jul 2013.

[7] J.-P. Laval, P. Blaineau, N. Leprovost, B. Dubrulle, and F. Daviaud, "Influence of turbulence on the dynamo threshold," *Phys. Rev. Lett.*, vol. 96, p. 204503, May 2006.

[8] A. Alexakis, S. Fauve, C. Gissinger, and F. Pétrélis, "Effect of fluctuations on mean-field dynamos," *Journal of Plasma Physics*, vol. 84, no. 4, p. 735840401, 2018.

[9] F. Pétrélis, N. Mordant, and S. Fauve, "On the magnetic fields generated by experimental dynamos," *Geophysical & Astrophysical Fluid Dynamics*, vol. 101, no. 3-4, pp. 289–323, 2007.

[10] F. Pétrélis and S. Fauve, "Mechanisms for magnetic field reversals," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1916, pp. 1595–1605, 2010.

[11] C. J. P. Gissinger, "A numerical model of the vks experiment," *Europhysics Letters*, vol. 87, p. 39002, aug 2009.

[12] F. Ravelet, B. Dubrulle, F. Daviaud, and P.-A. Ratié, "Kinematic alpha tensors and dynamo mechanisms in a von Kármán swirling flow," *Phys. Rev. Lett.*, vol. 109, p. 024503, Jul 2012.

[13] F. H. Busse and J. Wicht, "A simple dynamo caused by conductivity variations," *Geophysical & Astrophysical Fluid Dynamics*, vol. 64, no. 1-4, pp. 135–144, 1992.

[14] T. M. Rogers and J. N. McElwaine, "The hottest hot jupiters may host atmospheric dynamos," *The Astrophysical Journal Letters*, vol. 841, p. L26, may 2017.

[15] F. Marcotte, B. Gallet, F. m. c. Pétrélis, and C. Gissinger, "Enhanced dynamo growth in nonhomogeneous conducting fluids," *Phys. Rev. E*, vol. 104, p. 015110, Jul 2021.

[16] A. Giesecke, F. Stefani, and G. Gerbeth, "Role of soft-iron impellers on the mode selection in the von kármán–sodium dynamo experiment," *Phys. Rev. Lett.*, vol. 104, p. 044503, Jan 2010.

[17] B. Gallet, F. Pétrélis, and S. Fauve, "Spatial variations of magnetic permeability as a source of dynamo action," *Journal of Fluid Mechanics*, vol. 727, p. 161–190, 2013.

[18] F. Plunian and T. Alboussière, "Axisymmetric dynamo action produced by differential rotation, with anisotropic electrical conductivity and anisotropic magnetic permeability," *Journal of Plasma Physics*, vol. 87, no. 1, p. 905870110, 2021.

[19] J. Herault and F. Pétrélis, "Optimum reduction of the dynamo threshold by a ferromagnetic layer located in the flow," *Phys. Rev. E*, vol. 90, p. 033015, Sep 2014.

[20] J. Varela, S. Brun, B. Dubrulle, and C. Nore, "Role of boundary conditions in helicoidal flow collimation: Consequences for the von kármán sodium dynamo experiment," *Phys. Rev. E*, vol. 92, p. 063015, Dec 2015.

[21] J. Varela, S. Brun, B. Dubrulle, and C. Nore, "Effects of turbulence, resistivity and boundary conditions on helicoidal flow collimation: Consequences for the Von-Kármán-Sodium dynamo experiment," *Physics of Plasmas*, vol. 24, 05 2017. 053518.

[22] C. Nore, D. Castanon, L. Cappanera, and J.-L. Guermond, "Numerical simulation of the von Kármán sodium dynamo experiment," *Journal of Fluid Mechanics*, vol. 854, pp. 164–195, 11 2018.

[23] J. Duchon and R. Robert, "Inertial energy dissipation for weak solutions of incompressible Euler and Navier-Stokes equations," *Nonlinearity*, vol. 13, pp. 249–255, dec 1999.

[24] S. Galtier, "On the origin of the energy dissipation anomaly in (Hall) magnetohydrodynamics," *Journal of Physics A: Mathematical and Theoretical*, vol. 51, p. 205501, apr 2018.

[25] V. David, S. Galtier, F. Sahraoui, and L. Z. Hadid, "Energy transfer, discontinuities, and heating in the inner heliosphere measured with a weak and local formulation of the Politano–Pouquet law," *The Astrophysical Journal*, vol. 927, p. 200, mar 2022.

[26] B. Dubrulle, "Beyond Kolmogorov cascades," *J. Fluid Mech.*, vol. 867, p. P1, 2019.

[27] H. Faller, D. Geneste, T. Chaabo, A. Cheminet, V. Valori, Y. Ostovan, L. Cappanera, C. Cuvier, F. Daviaud, J.-M. Foucaut, and et al., "On the nature of intermittency in a turbulent von Kármán flow," *J. Fluid Mech.*, vol. 914, p. A2, 2021.

[28] H. Faller, *Dissipation, cascades et singularités en turbulence.* PhD thesis, ED-PIF, 2018-2021.

[29] G. L. Eyink, "Stochastic flux freezing and magnetic dynamo," *Phys. Rev. E*, vol. 83, p. 056405, May 2011.

[30] S. Childress, *Topological Aspects of the Dynamics of Fluids and Plasmas*, ch. Fast dynamo theory. Ed. Editors, H. K. Moffatt, G. M. Zaslavsky, P. Comte, M. Tabor, Springer Dordrecht, 1992.

[31] G. L. Eyink, "Cascades and dissipative anomalies in nearly collisionless plasma turbulence," *Phys. Rev. X*, vol. 8, p. 041020, Nov 2018.

[32] V. David and S. Galtier, "Proof of the zeroth law of turbulence in one-dimensional compressible magnetohydrodynamics and shock heating," *Phys. Rev. E*, vol. 103, p. 063217, Jun 2021.

[33] J.-L. Guermond, P. Minev, and J. Shen, "An overview of projection methods for incompressible flows," *Computer methods in applied mechanics and engineering*, vol. 195, no. 44-47, pp. 6011–6045, 2006.

[34] R. Pasquetti, R. Bwemba, and L. Cousin, "A pseudo-penalization method for high Reynolds number unsteady flows," *Appl. Numer. Math.*, vol. 58, pp. 946–954, July 2008.

[35] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998.

[36] S. Balay, W. Gropp, L. C. McInnes, and B. Smith, "Petsc, the portable, extensible toolkit for scientific computation," *Argonne National Laboratory*, vol. 2, no. 17, 1998.

[37] A. Giesecke, C. Nore, F. Stefani, G. Gerbeth, J. Léorat, F. Luddens, and J.-L. Guermond, "Electromagnetic induction in non-uniform domains," *Geophys. Astrophys. Fluid Dyn.*, vol. 104, no. 5, pp. 505–529, 2010.

[38] A. Bonito, J.-L. Guermond, and F. Luddens, "Regularity of the Maxwell equations in heterogeneous media and Lipschitz domains.," *Journal of Mathematical Analysis and applications*, vol. 408, pp. 498–512, Dec. 2013.

[39] A. Giesecke, C. Nore, F. Stefani, G. Gerbeth, J. Léorat, W. Herreman, F. Luddens, and J.-L. Guermond, "Influence of high-permeability discs in an axisymmetric model of the Cadarache dynamo experiment," *New Journal of Physics*, vol. 14, no. 5, p. 053005, 2012.

[40] P. Marti, N. Schaeffer, R. Hollerbach, D. Cébron, C. Nore, F. Luddens, J.-L. Guermond, J. Aubert, S. Takehiro, Y. Sasaki, Y.-Y. Hayashi, R. Simitev, F. Busse, S. Vantieghem, and A. Jackson, "Full sphere hydrodynamic and dynamo benchmarks," *Geophysical Journal International*, vol. 197, pp. 119–134, Apr. 2014.

[41] R. Laguerre, C. Nore, A. Ribeiro, J. Léorat, J.-L. Guermond, and F. Plunian, "Impact of impellers on the axisymmetric magnetic mode in the VKS2 dynamo experiment," *Phys. Rev. Lett.*, vol. 101, no. 10, p. 104501, 2008.

[42] R. Laguerre, C. Nore, A. Ribeiro, J. Léorat, J.-L. Guermond, and F. Plunian, "Erratum: Impact of impellers on the axisymmetric magnetic mode in the VKS2 dynamo experiment [phys. rev. lett. 101, 104501 (2008)]," *Phys. Rev. Lett.*, vol. 101, p. 219902, Nov 2008.

[43] E. W. Saw, D. Kuzzay, D. Faranda, A. Guittoneau, F. Daviaud, W.-G. C., P. V., and B. Dubrulle, "Experimental characterization of extreme events of inertial dissipation in a turbulent swirling flow," *Nature Comm.*, vol. 7, p. 12466, 2016.

[44] P. Debue, V. Valori, C. Cuvier, F. Daviaud, J.-M. Foucaut, J.-P. Laval, C. Wiertel-Gasquet, V. Padilla, and B. Dubrulle, "Three-dimensional analysis of precursors to non-viscous dissipation in an experimental turbulent flow," *J. Fluid Mech.*, vol. A9, p. 914, 2021.

[45] C. Nore, L. S. Tuckerman, O. Daube, and S. Xin, "The 1:2 mode interaction in exactly counter-rotating von Kármán swirling flow," *Journal of Fluid Mechanics*, vol. 477, pp. 51–88, 2 2003.

A Navier-Stokes calculations

Using the scalar product of $\rho u/2$ with equation (15) and the scalar product of $\rho\overline{u}^\ell/2$ with equation (**??**) leads to the equation for the transferred kinetic energy from all scales larger than ℓ to the scale ℓ:

$$\partial_t\frac{\rho u\overline{u}^\ell}{2}+\frac{\rho}{2}\left(u\partial_i\overline{u_i u}^\ell+\overline{u}^\ell\partial_i u_i u-u\nu\Delta\overline{u}^\ell-\overline{u}^\ell\nu\Delta u+u\cdot\nabla\frac{\overline{p}^\ell}{\rho}+\overline{u}^\ell\cdot\nabla\frac{p}{\rho}\right)= \frac{1}{2}(\overline{u}^\ell\cdot(j\times b)+u\cdot(\overline{j\times b}^\ell)+\rho u\cdot\overline{f}^\ell+\rho\overline{u}^\ell\cdot f). \tag{47}$$

After some calculations, the left hand side yields a simpler form (using multiple times $\nabla\cdot u=0$ and $\nabla\cdot\overline{u}^\ell=0$):

$$\frac{1}{2}\nabla\cdot(\overline{u}^\ell p+u\overline{p}^\ell)+\frac{\rho}{2}\nabla\cdot[(u\overline{u}^\ell)u]-\frac{\rho}{2}\nu\Delta(u\overline{u}^\ell)+\rho\nu\partial_i\overline{u_j}^\ell\partial_i u_j+\frac{\rho}{2}(u\overline{u_i\partial_i u}^\ell-u_i u\partial_i\overline{u}^\ell)=\nabla\cdot J^{NS}+\mathcal{D}^\nu+\mathcal{D}^u. \tag{48}$$

And the first part of the right hand side can then be written as:

$$\begin{aligned}\overline{u}^\ell\cdot(j\times b)+u\cdot(\overline{j\times b}^\ell)&=-\big(\overline{u}^\ell\cdot(b\times j)+u\cdot(\overline{b\times j}^\ell)\big),\\&=-2\left(\mathcal{T}^{u\to b}+\mathcal{D}^b\right),\end{aligned} \tag{49}$$

while $\frac{\rho}{2}(u\cdot\overline{f}^\ell+\overline{u}^\ell\cdot f)=\mathcal{P}$.

B MHD calculations

Using the scalar product of $b/2\mu$ with equation (16) and the scalar product of $\overline{b}^\ell/2\mu$ with equation (2) leads to the equation for the transferred magnetic energy from all scales larger than ℓ to the scale ℓ:

$$\partial_t\frac{b\overline{b}^\ell}{2\mu}=\frac{1}{2\mu}\left(\overline{b}^\ell\cdot\nabla\times(u\times b)-\overline{b}^\ell\cdot\nabla\times(\frac{j}{\sigma})+b\cdot\nabla\times(\overline{u\times b}^\ell)-b\cdot\nabla\times\overline{(\frac{j}{\sigma})}^\ell\right)-\frac{b\cdot\overline{b}^\ell}{2\mu^2}\partial_t\mu. \tag{50}$$

The variation of the physical properties σ and μ leads to extra terms that we develop in the following.

B.1 Using various conductive materials

The sharp variation in the electrical conductivity at $\{r=1.4R_{\rm cyl},\,-H/2\le z\le H/2\}$ between the liquid sodium lateral layer ($\sigma=\sigma_0$) and the copper wall of the container ($\sigma=4.5\sigma_0$) needs to be taken into account:

$$\frac{(\overline{b}^\ell)}{\mu}\cdot\nabla\times(\frac{j}{\sigma})+\frac{b}{\mu}\cdot\nabla\times\overline{(\frac{j}{\sigma})}^\ell=\nabla\cdot\left(\frac{j}{\sigma}\times\frac{(\overline{b}^\ell)}{\mu}+\overline{(\frac{j}{\sigma})}^\ell\times\frac{b}{\mu}\right)+\frac{j}{\sigma}\cdot\nabla\times\frac{(\overline{b}^\ell)}{\mu}+\overline{(\frac{j}{\sigma})}^\ell\cdot\nabla\times\frac{b}{\mu}, \tag{51}$$

Using:

$$\overline{(\frac{j}{\sigma})}^\ell=\frac{(\overline{j}^\ell)}{\sigma}+\overline{(\frac{j}{\sigma})}^\ell-\frac{(\overline{j}^\ell)}{\sigma} \tag{52}$$

$$\nabla\times\frac{(\overline{b}^\ell)}{\mu}=\nabla\times\overline{H}^\ell+\nabla\times\frac{(\overline{b}^\ell)}{\mu}-\nabla\times\overline{H}^\ell \tag{53}$$

Then:

$$\frac{j}{\sigma}\cdot\nabla\times\frac{(\overline{b}^\ell)}{\mu}+\overline{(\frac{j}{\sigma})}^\ell\cdot\nabla\times\frac{b}{\mu}=\frac{j}{\sigma}\cdot\overline{j}^\ell+\frac{j}{\sigma}\cdot\left(\nabla\times\frac{(\overline{b}^\ell)}{\mu}-\nabla\times\overline{H}^\ell\right)+\frac{(\overline{j}^\ell)}{\sigma}\cdot j+\left(\overline{(\frac{j}{\sigma})}^\ell-\frac{(\overline{j}^\ell)}{\sigma}\right)\cdot j \tag{54}$$

Hence:

$$\frac{(\overline{b}^{\ell})}{\mu}\cdot\nabla\times(\frac{j}{\sigma})+\frac{b}{\mu}\cdot\nabla\times\overline{(\frac{j}{\sigma})}^{\ell}=\nabla\cdot\left(\frac{j}{\sigma}\times\frac{(\overline{b}^{\ell})}{\mu}+\overline{(\frac{j}{\sigma})}^{\ell}\times\frac{b}{\mu}\right)+2\mathcal{D}^{\sigma}+2\mathcal{D}^{\mu\sigma}+2\mathcal{D}^{\overline{\sigma}},\tag{55}$$

B.2 Using various permeable materials

The sharp variation in the magnetic permeability between the liquid sodium and the moving impellers (when $\mu\neq\mu_0$) needs to be taken into account:

$$\begin{aligned}\frac{b}{\mu}\cdot\nabla\times(\overline{u\times b}^{\ell})&=-\nabla\cdot\left(\frac{b}{\mu}\times(\overline{u\times b}^{\ell})\right)+(\overline{u\times b}^{\ell})\cdot(\nabla\times h)\\&=-\nabla\cdot\left(\frac{b}{\mu}\times(\overline{u\times b}^{\ell})\right)+\overline{u\times b}^{\ell}\cdot j\end{aligned}\tag{56}$$

$$\begin{aligned}\frac{(\overline{b}^{\ell})}{\mu}\cdot\nabla\times(u\times b)&=-\nabla\cdot\left(\frac{(\overline{b}^{\ell})}{\mu}\times(u\times b)\right)+(u\times b)\cdot\nabla\times\frac{(\overline{b}^{\ell})}{\mu}\\&=-\nabla\cdot\left(\frac{(\overline{b}^{\ell})}{\mu}\times(u\times b)\right)+(u\times b)\cdot\nabla\times\overline{H}^{\ell}+(u\times b)\cdot\left(\nabla\times\frac{(\overline{b}^{\ell})}{\mu}-\nabla\times\overline{H}^{\ell}\right).\end{aligned}\tag{57}$$

Hence :

$$\frac{b}{\mu}\cdot\nabla\times(\overline{u\times b}^{\ell})+\frac{(\overline{b}^{\ell})}{\mu}\cdot\nabla\times(u\times b)=-\nabla\cdot\left(\frac{b}{\mu}\times(\overline{u\times b}^{\ell})+\frac{(\overline{b}^{\ell})}{\mu}\times(u\times b)\right)+2\left(\mathcal{T}^{u\to b}-\mathcal{D}^{b}\right)-2\mathcal{D}^{\mu\mathcal{T}}.\tag{58}$$

C Linking kinetic energy transfers and velocity increments

Following [26], the filtering process discussed in section § 4.4 can be interpreted as a smoothing operation at a resolution ℓ achieved through the function φ_ℓ (smooth non-negative function with unit integral) such that the filtered velocity field reads:

$$\overline{u_i}^{\ell}(\mathbf{x},t)=\int\varphi_\ell(\mathbf{r})u_i(\mathbf{x}+\mathbf{r},t)d\vec{r}.\tag{59}$$

Using increments of the velocity fields, the third-order structure function can be written as (see equation (2.7) in [27]):

$$\begin{aligned}(\frac{\mathcal{D}^I}{\rho}=)S^{NS}&=\frac{1}{4}\int(\nabla\varphi_\ell)\cdot(\delta u\delta u^2)d\vec{r}\\&=-\frac{1}{4}\int\varphi_\ell\nabla\cdot(\delta u\delta u^2)d\vec{r}\\&=-\frac{1}{4}\left[\partial_i\overline{u_iu_ju_j}^{\ell}-u_i\partial_i\overline{u_ju_j}^{\ell}-2u_j\partial_i\overline{u_iu_j}^{\ell}+u_ju_j\partial_i\overline{u_i}^{\ell}+2u_iu_j\partial_i\overline{u_j}^{\ell}\right]\\&=-\frac{1}{4}\left[\nabla\cdot(\overline{uu^2}^{\ell}-u\overline{u^2}^{\ell})-2u_j\overline{u_i\partial_iu_j}^{\ell}+2u_iu_j\partial_i\overline{u_j}^{\ell}\right]\\&=-\frac{1}{4}\nabla\cdot(\overline{uu^2}^{\ell}-u\overline{u^2}^{\ell})+\frac{1}{\rho}\mathcal{D}^u\\&=\frac{1}{2}\nabla\cdot J_{\mathcal{S}^{NS}}+\frac{1}{\rho}\mathcal{D}^u\end{aligned}\tag{60}$$

with $J_{\mathcal{S}^{NS}}=-\overline{uu^2}^{\ell}+u\overline{u^2}^{\ell}$. The term S^{NS} was denoted $\mathcal{D}^I_\ell$ in [26]. Note that it differs from the anomalous dissipation term $\mathcal{D}^u$ by an additional flux term (in divergence). Integrated throughout

space with the proper boundary conditions, the latter should disappear but it can be responsible for localized large values on the boundaries. If we are interested in the energy injected in the small scales, the important term is $\mathcal{D}^u$.

D Linking magnetic energy transfers and velocity increments

In the MHD case with $\mu \neq const.$, we introduce:

$$\begin{aligned} S^{Mag} &= -\frac{\mu}{2}\int \varphi_\ell(\delta u\cdot\delta(b\times j)+\delta j\cdot\delta(b\times u))d\vec{r} \\ &= \frac{\mu}{2}\big[-\overline{u\cdot(b\times j)}^\ell-\overline{j\cdot(b\times u)}^\ell+u\cdot(\overline{b\times j}^\ell)+\overline{j}^\ell\cdot b\times u+j\cdot\overline{b\times u}^\ell+\overline{u}^\ell\cdot(b\times j)\big] \\ &= \frac{\mu}{2}\big[-u\cdot(\overline{j\times b}^\ell-\overline{j}^\ell\times b)-j\cdot(\overline{u\times b}^\ell-\overline{u}^\ell\times b)\big] \\ &= 2\mu\mathcal{D}^b \end{aligned} \tag{61}$$

In the special case $\mu = const.$, this formula simplifies like in [24]:

$$\begin{aligned} \tilde{S}^{Mag} &= \frac{1}{4}\int(\nabla\varphi_\ell)\cdot(\delta u\delta b^2-2(\delta u\cdot\delta b)\delta b)d\vec{r} \\ &= -\frac{1}{4}\int\varphi_\ell\nabla\cdot(\delta u\delta b^2-2(\delta u\cdot\delta b)\delta b)d\vec{r} \end{aligned} \tag{62}$$

Treating both parts of the integral independently:

$$\begin{aligned} -\int\varphi_\ell\nabla\cdot(\delta u\delta b^2)d\vec{r} &= -\nabla\cdot(\overline{ub^2}^\ell)+2b_j\partial_i(\overline{u_ib_j}^\ell)+u_i\partial_i(\overline{b_jb_j}^\ell)-b_jb_j\partial_i(\overline{u_i}^\ell)-2b_ju_i\partial_i(\overline{b_j}^\ell) \\ &= -\nabla\cdot(\overline{ub^2}^\ell+\overline{u}^\ell b^2+u(\overline{b}^\ell\cdot b)) \\ &+2b_j\overline{u_i\partial_i(b_j)}^\ell)+2u_i\overline{b_j\partial_i(b_j)}^\ell)+2b_j\overline{u_i}^\ell\partial_i(b_j)+2\overline{b_j}^\ell u_i\partial_ib_j \end{aligned} \tag{63}$$

$$\begin{aligned} \int\varphi_\ell\nabla\cdot((\delta u\cdot\delta b)\delta b)d\vec{r} &= \nabla\cdot(\overline{(u\cdot b)b}^\ell)+b_i\partial_i(\overline{u_jb_j}^\ell)+u_j\partial_i(\overline{b_ib_j}^\ell)+b_j\partial_i(\overline{u_jb_i}^\ell) \\ &+b_ib_j\partial_i(\overline{u_j}^\ell)+b_iu_j\partial_i(\overline{b_j}^\ell)+b_ju_j\partial_i(\overline{b_i}^\ell) \\ &= \nabla\cdot(\overline{(u\cdot b)b}^\ell+b_i\overline{u_jb_j}^\ell+b_ib_j\overline{u_j}^\ell+b_iu_j\overline{b_j}^\ell) \\ &-u_j\overline{b_i\partial_i(b_j}^\ell)-b_j\overline{b_i\partial_i(u_j}^\ell)-\overline{u_j}^\ell b_i\partial_i(b_j)-\overline{b_j}^\ell b_i\partial_i(u_j) \end{aligned} \tag{64}$$

Grouping two by two similar terms:

$$b_j\overline{u_i\partial_i(b_j}^\ell)-b_j\overline{b_i\partial_i(u_j}^\ell)=b\cdot\nabla\times(\overline{b\times u}^\ell)=\nabla\times b\cdot\overline{b\times u}^\ell-\nabla\cdot(b\times(\overline{b\times u}^\ell)) \tag{65}$$

$$\overline{b_j}^\ell u_i\partial_ib_j-\overline{b_j}^\ell b_i\partial_i(u_j)=\overline{b}^\ell\cdot\nabla\times(b\times u)=\overline{\nabla\times b}^\ell\cdot b\times u-\nabla\cdot(\overline{b}^\ell\times(b\times u)) \tag{66}$$

$$u_i\overline{b_j\partial_i(b_j}^\ell)-u_j\overline{b_i\partial_i(b_j}^\ell)=u\cdot(\overline{b\times(\nabla\times b)}^\ell) \tag{67}$$

$$b_j\overline{u_i}^\ell\partial_i(b_j)-\overline{u_j}^\ell b_i\partial_i(b_j)=\overline{u}^\ell\cdot(b\times(\nabla\times b)) \tag{68}$$

using that for fields of null divergence:

$$\nabla\times(A\times B)=(B\cdot\nabla)A-(A\cdot\nabla)B=B_i\partial_iA_j-A_i\partial_iB_j \tag{69}$$

$$B\times(\nabla\times B)=\frac{1}{2}\nabla B^2-(B\cdot\nabla)B=B_i\partial_jB_i-B_i\partial_iB_j \tag{70}$$

On the other hand:

$$4\mathcal{D}^b = -u\cdot(\overline{j\times b}^\ell - \overline{j}^\ell\times b) - j\cdot(\overline{u\times b}^\ell - \overline{u}^\ell\times b) = u\cdot(\overline{b\times j}^\ell) + \overline{j}^\ell\cdot b\times u + j\cdot\overline{b\times u}^\ell + \overline{u}^\ell\cdot(b\times j) \quad (71)$$

Hence, by regrouping all terms with a divergence in a current, and using $\nabla\times b = \mu j$ and $\overline{\nabla\times b}^\ell = \mu\overline{j}^\ell$ (because $\mu = const.$):

$$2\mu\mathcal{D}^b = \tilde{S}^{Mag} - \frac{1}{2}\nabla\cdot J_{S^{Mag}} \quad (72)$$

with:

$$J_{S^{Mag}} = \overline{b}^\ell\times(b\times u) + b\times(\overline{b\times u}^\ell) - \overline{(u\cdot b)b}^\ell - b(\overline{u\cdot b}^\ell) - b(b\cdot\overline{u}^\ell) - b(u\cdot\overline{b}^\ell) + \frac{1}{2}\overline{ub^2}^\ell + \frac{1}{2}\overline{u}^\ell b^2 + \frac{1}{2}u(\overline{b}^\ell\cdot b) \quad (73)$$

And so, in this special case :

$$S^{Mag} = \tilde{S}^{Mag} - \frac{1}{2}\nabla\cdot J_{S^{Mag}} \quad (74)$$

E Filtering issues

We examine the issues related to the filtering process concerning the boundaries and its non-commutation with derivatives. We start with equation (**??**) in a domain $\mathcal{D}\subset\mathbb{R}^3$ with boundary $\partial\mathcal{D} = \partial\mathcal{D}$:

$$(1-\ell^2\Delta)\overline{f_i}^\ell = f_i \quad (75)$$

The weak formulation is obtained using test functions ϕ_i:

$$\begin{aligned}\int_{\mathcal{D}}[(1-\ell^2\Delta)\overline{f_i}^\ell]\phi_i &= \int_{\mathcal{D}} f_i\phi_i \\ \Leftrightarrow \int_{\mathcal{D}}\overline{f_i}^\ell\phi_i + \int_{\mathcal{D}}\ell^2(\nabla_k\overline{f_i}^\ell)(\nabla_k\phi_i) - \int_{\partial\mathcal{D}}\ell^2 n_k(\nabla_k\overline{f_i}^\ell)\phi_i &= \int_{\mathcal{D}} f_i\phi_i\end{aligned}$$

This is the formulation used in SFEMANS to solve these equations.

E.1 Discontinuous Galerkin method

This section explains how to solve the filtering equation at the interface between the velocity mesh and the magnetic field mesh (that includes the velocity mesh). This is needed in order for both meshes to communicate, otherwise the filter would create a discontinuity in the filtered field at the interface between the meshes.

Let us consider a domain $\mathcal{D}\subset\mathbb{R}^3$ with boundary $\partial\mathcal{D}$. The domain is assumed to be partitioned in two subdomains $\mathcal{D}_1$ and $\mathcal{D}_2$ such that $\overline{\mathcal{D}} = \overline{\mathcal{D}_1}\cup\overline{\mathcal{D}_2}$, $\mathcal{D}_1\cap\mathcal{D}_2 = \emptyset$. The interface between both domains is denoted by $\Sigma = \overline{\mathcal{D}_1}\cap\overline{\mathcal{D}_2}$.

$$\begin{aligned}\int_{\mathcal{D}_1\cup\mathcal{D}_2}[(1-\ell^2\Delta)\overline{f_i}^\ell]\phi_i &= \int_{\mathcal{D}_1\cup\mathcal{D}_2} f_i\phi_i \\ \Leftrightarrow \int_{\mathcal{D}_1\cup\mathcal{D}_2}\overline{f_i}^\ell\phi_i + \int_{\mathcal{D}_1\cup\mathcal{D}_2}\ell^2(\nabla_k\overline{f_i}^\ell)(\nabla_k\phi_i) - \int_{\partial\mathcal{D}}\ell^2 n_k(\nabla_k\overline{f_i}^\ell)\phi_i & \\ -\int_\Sigma\left(\ell^2 n_{1k}(\nabla_k\overline{f_{1i}}^\ell)\phi_{1i} + \ell^2 n_{2k}(\nabla_k\overline{f_{2i}}^\ell)\phi_{2i}\right) &= \int_{\mathcal{D}_1\cup\mathcal{D}_2} f_i\phi_i\end{aligned}$$

The vector n_i is the outward normal vector to the interface Σ with respect to $\mathcal{D}_i$.

Using $(a_1b_1 + a_2b_2) = \frac{1}{2}\left[(a_1 + a_2)(b_1 + b_2) + (a_1 - a_2)(b_1 - b_2)\right]$:

$$\begin{aligned}
&\int_\Sigma \left(n_{1k}(\nabla_k \overline{f_{1i}}^\ell)\phi_{1i} + n_{2k}(\nabla_k \overline{f_{2i}}^\ell)\phi_{2i}\right) \\
&= \int_\Sigma \frac{1}{2}\left((n_{1k}\nabla_k \overline{f_{1i}}^\ell + n_{2k}\nabla_k \overline{f_{2i}}^\ell)(\phi_{1i} + \phi_{2i}) + (n_{1k}\nabla_k \overline{f_{1i}}^\ell - n_{2k}\nabla_k \overline{f_{2i}}^\ell)(\phi_{1i} - \phi_{2i})\right) \\
&= \int_\Sigma \frac{1}{2}\left(n_{1k}\|\nabla_k \overline{f_i}^\ell\|\{\phi_i\} + n_{1k}\{\nabla_k \overline{f_i}^\ell\}\|\phi_i\|\right)
\end{aligned}$$

using $n_{2k} = -n_{1k}$ and defining $\|f_i\| = (f_{1i} - f_{2i})$ and $\{f_i\} = (f_{1i} + f_{2i})$.

In cylindrical coordinates, we have:

$$n_k(\nabla_k \overline{f_i}^\ell)\phi_i = \begin{vmatrix} (n_r\partial_r + n_z\partial_z)\overline{f_r}^\ell \\ (n_r\partial_r + n_z\partial_z)\overline{f_\theta}^\ell \\ (n_r\partial_r + n_z\partial_z)\overline{f_z}^\ell \end{vmatrix} \cdot \begin{vmatrix} \phi_r \\ \phi_\theta \\ \phi_z \end{vmatrix}$$

For regularization purposes, the following term is added:

$$\int_\Sigma \frac{\beta}{h}(\overline{f_{1i}}^\ell - \overline{f_{2i}}^\ell)(\phi_{1i} - \phi_{2i}) = \int_\Sigma \frac{\beta}{h}\|\overline{f_i}^\ell\|\|\phi_i\|$$

where h represents the mesh size and β a tunable penalty parameter that is always set to one in our numerical investigations.

The final implemented equation is:

$$\begin{aligned}
&\int_{\mathcal{D}_1\cup\mathcal{D}_2} \overline{f_i}^\ell\phi_i + \int_{\mathcal{D}_1\cup\mathcal{D}_2} \ell^2(\nabla_k \overline{f_i}^\ell)(\nabla_k\phi_i) - \int_{\partial\mathcal{D}} \ell^2 n_k(\nabla_k \overline{f_i}^\ell)\phi_i \\
&- \int_\Sigma \frac{\ell^2}{2}\left(n_{1k}\|\nabla_k \overline{f_i}^\ell\|\{\phi_i\} + n_{1k}\{\nabla_k \overline{f_i}^\ell\}\|\phi_i\|\right) + \int_\Sigma \frac{\beta}{h}\|\overline{f_i}^\ell\|\|\phi_i\| = \int_{\mathcal{D}_1\cup\mathcal{D}_2} f_i\phi_i
\end{aligned} \tag{76}$$

Note that, when $\ell = 0$, we get $\overline{f_i}^0 = f_i$ as needed. When $\ell \neq 0$, the impact of the boundaries ($\partial\mathcal{D}$, Σ) scales like ℓ^2.

E.2 Non-commutation of filter and derivatives

In this section we study commutation between filtering and derivatives. As an example, we study the case where we derive according to one direction X. The equation to filter such a derivative would be:

$$(1 - \ell^2\Delta)\overline{\partial_X f_i}^\ell = \partial_X f_i, \tag{77}$$

with its weak formulation being:

$$\int_{\mathcal{D}} [(1 - \ell^2\Delta)\overline{\partial_X f_i}^\ell]\phi_i = \int_{\mathcal{D}} (\partial_X f_i)\phi_i. \tag{78}$$

Working on the rhs gives:

$$\begin{aligned}
\int_{\mathcal{D}} (\partial_X f_i)\phi_i &= -\int_{\mathcal{D}} f_i(\partial_X\phi_i) + \int_{\partial\mathcal{D}} f_i\phi_i \\
&= -\int_{\mathcal{D}} [(1 - \ell^2\Delta)\overline{f_i}^\ell]\partial_X\phi_i + \int_{\partial\mathcal{D}} f_i\phi_i \\
&= \int_{\mathcal{D}} [\partial_X(1 - \ell^2\Delta)\overline{f_i}^\ell]\phi_i - \int_{\partial\mathcal{D}} [(1 - \ell^2\Delta)\overline{f_i}^\ell]\phi_i + \int_{\partial\mathcal{D}} f_i\phi_i \\
&= \int_{\mathcal{D}} [(1 - \ell^2\Delta)\partial_X\overline{f_i}^\ell]\phi_i + \int_{\partial\mathcal{D}} \ell^2(\Delta\overline{f_i}^\ell)\phi_i + \int_{\partial\mathcal{D}} (f_i - \overline{f_i}^\ell)\phi_i
\end{aligned}$$

Hence:

$$\int_{\mathcal{D}}[(1-\ell^2\Delta)(\overline{\partial_X f_i}^{\ell}-\partial_X\overline{f_i}^{\ell})]\phi_i=\int_{\partial\mathcal{D}}(f_i-\overline{f_i}^{\ell})\phi_i+\int_{\partial\mathcal{D}}\ell^2(\Delta\overline{f_i}^{\ell})\phi_i. \tag{79}$$

When $\ell=0$, the filter and derivatives commute if Dirichlet boundary conditions are used (vanishing of the first rhs term). In contrast, when $\ell\neq 0$, there are errors which scale like ℓ^2 on the boundaries $\partial\mathcal{D}$ as indicated by the second rhs term in ℓ^2. This means that, as long as we are interested in derivatives away from the edges $\geq\ell^2$, the error introduced by commutating derivatives and filtering is negligible.

F Symmetries of the von Kármán flow and implications for the Ω and α terms

The von Kármán flow with exactly counter-rotating impellers has one particular symmetry that can provide information about the spatial distribution of instantaneous and averaged quantities based on the velocity field. It is called the R_π symmetry [45] and corresponds to a symmetry of rotation of π about any horizontal axis:

$$R_\pi\begin{pmatrix}u_r\\u_\theta\\u_z\end{pmatrix}(r,\theta,z)=\begin{pmatrix}u_r\\-u_\theta\\-u_z\end{pmatrix}(r,-\theta,-z) \tag{80}$$

where $u=(u_r,u_\theta,u_z)$ is the velocity vector in cylindrical coordinates with r the radial distance, θ the azimuthal angle and z the height. Another symmetry, apart from the impellers, is axisymmetry about the z-axis defined as:

$$S_{\theta_0}\begin{pmatrix}u_r\\u_\theta\\u_z\end{pmatrix}(r,\theta,z)=\begin{pmatrix}u_r\\u_\theta\\u_z\end{pmatrix}(r,\theta+\theta_0,z). \tag{81}$$

Based on (80), one can deduce that, for a velocity field symmetric under the R_π symmetry, the Ω-terms satisfy:

$$\Omega'_r(r,\theta,z)=-\Omega'_r(r,-\theta,-z)\quad\text{and}\quad\Omega'_z(r,\theta,z)=+\Omega'_z(r,-\theta,-z) \tag{82}$$

For the α-tensor (11), one needs the expression for the velocity gradient tensor in cylindrical coordinates:

$$\nabla u=\begin{pmatrix}\frac{\partial u_r}{\partial r} & \frac{1}{r}\frac{\partial u_r}{\partial\theta}-\frac{u_\theta}{r} & \frac{\partial u_r}{\partial z}\\ \frac{\partial u_\theta}{\partial r} & \frac{1}{r}\frac{\partial u_\theta}{\partial\theta}+\frac{u_r}{r} & \frac{\partial u_\theta}{\partial z}\\ \frac{\partial u_z}{\partial r} & \frac{1}{r}\frac{\partial u_z}{\partial\theta} & \frac{\partial u_z}{\partial z}\end{pmatrix} \tag{83}$$

Using (83) in the helicity tensor, one can infer that, for a velocity field symmetric under the R_π symmetry, the helicity tensor satisfies:

$$\begin{pmatrix}h_{rr} & h_{r\theta} & h_{rz}\\ h_{\theta r} & h_{\theta\theta} & h_{\theta z}\\ h_{zr} & h_{z\theta} & h_{zz}\end{pmatrix}(r,\theta,z)=\begin{pmatrix}+h_{rr} & -h_{r\theta} & -h_{rz}\\ -h_{\theta r} & +h_{\theta\theta} & +h_{\theta z}\\ -h_{zr} & +h_{z\theta} & +h_{zz}\end{pmatrix}(r,-\theta,-z). \tag{84}$$

The α-tensor will follow the same rules.

5 - Conclusion

In this thesis, the goal was to better understand the dynamo effect in turbulent conducting flows. For this purpose, we simulated the von Kármán Sodium configuration using the SFEMaNS code. The choice of this configuration was motivated by the possibility of comparing the numerical results with the ones of a real experiment carried out at CEA Cadarache. To shed more light on the dynamo mechanism, we decided to study energy transfers between magnetic and velocity fields, as well as between scales. This is an extension to the magnetohydrodynamics equations of the approach used in a previous thesis by H.Faller [60]. These choices led to the need to implement various processing tools as well as updates to the SFEMaNS code.

A major improvement we implemented to the SFEMaNS code was the distribution of the simulation meshgrid across the processors. This has saved an enormous amount of memory, reaching around 300,000 grid points per processor in Navier-Stokes simulations which makes it possible to simulate higher kinetic Reynolds numbers. Thanks to this, we plan in the near future to compare DNS and LES computations performed at high Reynolds numbers to see the impact of the turbulence model.

In parallel, we developed a new method based on preconditioning the Stokes problem in SFEMaNS for potential future use in a time-stepping technique and we compared it against previously published preconditioning methods. The various preconditioning techniques have achieved similar performance and good scaling for parallel computing. However, they are not powerful enough to compete with the prediction-correction method currently in use in the SFEMaNS code. The limiting factor in these preconditioning methods is the time needed to invert the velocity problem. Hence, we see no reason to use these methods until further theoretical work has been carried out to make them more efficient.

In order to efficiently calculate and process the quantities of interest, we implemented numerous post-processing tools as there was no automation for the analysis and visualisation of SFEMaNS' outputs. Particular attention has been paid to saving memory space and processing time when calculating these quantities. A general tool to transfer any output of SFEMaNS into simple file formats for statistical treatment or visualization has been created. An automatized Paraview workflow with multiple functionalities has been implemented to easily visualize structures in 3D or slices. All these tools were necessary to implement, given the quantity of fields to be analyzed. They are also independent of each other and can therefore be used for any future simulations using SFEMaNS. The only tools that remain to be developed are those of statistical analysis, such as high-order moments and structure functions.

The derivation of energy balances in magnetohydrodynamics has led to the discovery of a variety of energy transfers. Above all, the influence of magnetic permeability variations on energy transfer at different scales is now well described. In our two case studies, the observed dynamo mechanism is highly dependent on the magnetic permeability of the turbines. In the case of steel impellers, the Lorentz force power of the fluid manages to compensate for all dissipation terms, whereas in the case of soft-iron turbines, it becomes anti-dynamo. For soft-iron impellers, in the saturated regime, rotation of the high-permeability impellers becomes the sole source of dynamo. However, in both cases, the ignition of the dynamo is purely due to the nature of the flow as the Lorentz force power is the only contributor.

A more complete analysis of the various energy transfers can be pursued. An analysis at a higher kinetic Reynolds number of the energy transfers responsible for anomalous dissipations may lead to traces of quasi-singularities or fast dynamo mechanisms. Simulations for other magnetic permeabilities of the impellers should provide a better understanding of the balance between the Lorentz force power and the rotating magnet effect. For all these simulations, it should be possible to track rare events and interesting coherent structures thanks to the implementation of all visualization tools. Finally, a statistical treatment of these quantities should be carried out to highlight correlations enabling a better understanding of their interactions.

6 - Résumé en français

6.1 . Introduction

6.1.1 . Contexte

L'effet dynamo est un processus permettant de générer un champ magnétique grâce au mouvement de matière conductrice de l'électricité. Pour les planètes, il s'agit d'un liquide (du fer dans le coeur liquide de la Terre) mis en mouvement par de nombreux mécanismes (effet Coriolis, convection thermique et solutale, etc) et, pour les étoiles, de gaz ionisé. Dans tous les cas, l'effet dynamo requiert la présence de matière conductrice et une source d'énergie la maintenant en mouvement. La génération d'un champ magnétique a, par ailleurs, été reproduite en laboratoire en utilisant du sodium liquide mis en mouvement grâce à différents dispositifs [1, 2, 3]. Cependant la seule expérience ayant reproduit une telle génération sans contraintes fortes sur le mouvement du fluide est celle appelée von Kármán Sodium (VKS) [4], les expériences précédentes ayant utilisé un champ de vitesse mathématiquement prédéterminé pour générer de la dynamo.

L'expérience de von Kármán Sodium est une installation impressionnante mise en place grâce à la collaboration de l'ENS Paris, l'ENS Lyon, du CEA Saclay et du CEA Cadarache. L'objectif de cette expérience est de créer et maintenir un champ magnétique grâce au transfert d'énergie provenant du fluide conducteur, et ainsi reproduire l'effet dynamo présent à l'intérieur des planètes et des étoiles. Le fluide utilisé est du sodium, maintenu à 120°C, et mis en mouvement turbulent grâce à deux turbines en contra-rotation.

Le maintien d'un champ magnétique a été mesuré pour la première fois en 2006. Le champ a été généré seulement lors de l'utilisation de turbines en fer doux [4] (et non en acier ou en cuivre) et avec un écoulement contenant peu de fluctuations [3]. Ce dernier point prouve alors que le champ magnétique généré ne provient pas uniquement de la rotation d'un aimant dans un milieu conducteur. Cependant, cela semble indiquer que les fluctuations nuisent aux mécanismes dynamos comme déjà remarqué dans des simulations numériques pour des géométries plus simples [5] ou en régime asymptotique [6]. Le champ obtenu est alors en moyenne dipolaire, aligné avec l'axe de symétrie, avec une composante azimutale concentrée au niveau des turbines. Expérimentalement il n'est pas possible la vitesse du fluide ou les variations non axi-symétriques du champ magnétique. Mais, pour certains paramètres, l'inversion du champ magnétique (Nord-Sud vers Sud-Nord et inversement) a été observée comme c'est le cas pour le champ magnétique terrestre.

La principale difficulté pour trouver les conditions permettant d'obtenir l'effet dynamo provient de la nature turbulente des écoulements des fluides dans les étoiles et les planètes. Un écoulement turbulent correspond à un champ de vitesse désordonné variant rapidement et contenant de nombreux vortex. Ce type d'écoulement apparaît lorsque les forces visqueuses (correspondant à la capacité pour le fluide de dissiper son mouvement) sont faibles, en comparaison à la force inertielle du système. Cela génère alors un champ de vitesse caractérisé par de nombreuses échelles spatiales : du mouvement global généré par les forces externes aux plus petits tourbillons dissipant l'énergie cinétique. L'effet dynamo provenant de cet écoulement, il devient alors difficile de trouver quels évènements provoquent la génération d'un champ magnétique. En particulier, cela peut entraîner la génération d'un champ magnétique par un champ de vitesse très irrégulier [7] via des mécanismes de dynamos rapides [8, 9]. Cela arrive quand le champ magnétique croit exponentiellement en un temps caractéristique de l'ordre de grandeur du temps advectif de l'écoulement. Ce phénomène est provoqué par des écoulements très turbulents pour lesquels les lignes de champ magnétique sont étirées de manière exponentielle par les vortex.

Pour pallier ces difficultés, une approche possible est de construire des modèles de dynamo simples en utilisant les théories de champs moyens et la théorie stochastique [10, 11]. Ces approches révèlent l'importance de deux éléments de l'écoulement du fluide : son hélicité (à travers l'effet α) et sa rotation différentielle (à travers l'effet Ω). Pour l'expérience VKS, de nombreuses combinaisons de ces effets ont été considérées pour expliquer l'émergence du champ magnétique [12, 13]. Cependant, ces modèles ne peuvent qu'analyser l'impact du champ de vitesse sur le champ magnétique. Il leur est donc impossible d'expliquer les différences de comportement observées dans l'expérience pour les différents matériaux utilisés pour les turbines. Cela requiert alors l'introduction de modélisations supplémentaires pour les interactions entre l'effet Ω et les variations spatiales ou temporelles de la conductivité électrique [14, 15, 16] ou de la perméabilité magnétique [17, 18, 19, 20]. D'autres effets d'amplification de l'effet α par collimation proche des turbines ont aussi été analysés [21, 22]. De plus, la théorie des champs moyens rend difficile la compréhension de la phase initiale de croissance du champ magnétique car elle nécessite la distinction entre un champ moyen et un champ rapidement variable pour la vitesse et le champ magnétique. Finalement, comme tous les évènements de petites échelles se retrouvent paramétrés dans les coefficients des effets α et Ω, il n'est pas possible d'observer les structures potentiellement responsables pour le mécanisme de dynamo rapide.

Dans cette thèse, nous proposons une nouvelle méthode pour comprendre les mécanismes dynamo permettant de générer un champ magnétique dans les fluides conducteurs et turbulents. L'idée principale est d'étudier les

bilans d'énergie locaux (temporellement et spatialement) en décrivant toutes les dissipations et transferts d'énergie entre champs et échelles. Cette approche a été proposée par Duchon et Robert [23] pour l'étude des écoulements turbulents décrits par les équations de Navier-Stokes. Cette méthode se base sur un principe de filtrage des champs permettant de séparer les différentes échelles de l'écoulement. Comparée à d'autres techniques de filtrage [24], celle-ci permet de détecter de potentielles singularités grâce à l'introduction de dissipations anormales correspondant à des transferts d'énergie vers des échelles de plus en plus petites. Cette technique a ensuite été étendue à la magnétohydrodynamique [25, 26]. Cependant, dans ces références, les variations de perméabilité magnétique et de conductivité électrique ne sont pas prises en compte alors qu'elles sont cruciales pour notre étude de l'expérience VKS et dans nos simulations. De plus, comme ces articles sont spécialisés dans l'étude de phénomènes astrophysiques (en particulier les vents solaires), les énergies magnétique et cinétique sont combinées. Comme nous souhaitons étudier les transferts d'énergie entre le champ de vitesse et le champ magnétique, nous devons revisiter cette approche pour séparer les deux. Ainsi, dans cette thèse, nous étendons cette méthode pour la magnétohydrodynamique dans le cas de conductivité électrique et de perméabilité magnétique variables (spatialement et temporellement) en décrivant tous les transferts d'énergie (d'un champ à l'autre et d'une échelle à l'autre). En combinant des simulations numériques déjà produites pour VKS avec cette approche de transferts locaux en énergie, il devient possible de déterminer où et quand les effets dynamos se produisent.

6.1.2 . Magnétohydrodynamique et effet dynamo

Pour commencer, les équations de la magnétohydrodynamique (MHD) sont utilisées pour modéliser les interactions entre le fluide conducteur et le champ magnétique. Ces équations proviennent de la combinaison des équations de Maxwell prises dans la limite non-relativiste, couplées aux équations de Navier-Stokes à travers la force de Lorentz et la loi d'Ohm. Dans cette section, nous proposons une courte description de ces équations ainsi que des quantités importantes pour l'étude de la turbulence et de l'effet dynamo. En particulier, nous introduisons les nombres de Reynolds et quelques théorèmes anti-dynamo importants pour comprendre les choix et limites des configurations expérimentales et numériques. Pour des raisons de simplicité, nous utilisons les conventions d'Einstein pour les sommes, c'est-à-dire pour des vecteurs f, g comprenant trois composantes $\sum_{i=1}^{3} f_i g_i$ sera écrit $f_i g_i$.

Les équations de la MHD sont les suivantes pour un fluide incompressible, conducteur et électriquement neutre dans une approximation quasi-statique

(donc non-relativiste) :

$$\partial_t u + \partial_i(u_i u) - \nu\Delta u + \frac{\nabla p}{\rho} = \frac{j \times b}{\rho}, \tag{6.1}$$

$$\partial_t b = \nabla \times (u \times b) - \nabla \times \frac{j}{\sigma}, \tag{6.2}$$

$$\nabla \cdot u = 0, \tag{6.3}$$

$$\nabla \cdot b = 0. \tag{6.4}$$

Les différentes quantités présentes dans ces équations sont : u le champ de vitesse, p le champ de pression, b la densité de flux magnétique (ou induction magnétique) , j le courant électrique, ρ la densité du fluide, ν la viscosité du fluide, σ la conductivité électrique du milieu et μ la perméabilité magnétique du milieu. La première équation (6.1) correspond à l'équation de Navier-Stokes pour les fluides incompressibles avec l'ajout de la force de Lorentz $(j \times b)/\rho$. La seconde (6.2) est obtenue en utilisant les équations de Maxwell dans la limite quasi-statique avec la loi d'Ohm écrite dans le référentiel du laboratoire $j = \sigma(E + u \times b)$ permettant alors d'éliminer le champ électrique E des équations. La troisième (6.3) traduit l'incompressibilité du fluide : l'écoulement est de divergence nulle. De même, la quatrième (6.4) provient de la conservation des lignes de champ magnétique et l'inexistence de monopoles magnétiques.

Les autres relations importantes pour le champ magnétique sont :

$$\nabla \times H = j, \tag{6.5}$$

$$\mu H = b, \tag{6.6}$$

où H est le champ magnétique. Ces équations proviennent des équations de Maxwell et permettent de lier le champ magnétique au courant électrique. Pour des raisons de simplicité, nous appellerons par la suite b le champ magnétique car nous ne ferons plus référence à H.

L'effet dynamo est alors l'amplification et le maintien du champ magnétique b par le mouvement du fluide conducteur u. Dans notre cas, cela provient d'une instabilité. En effet, $b = 0$ est toujours solution de l'équation (6.2). Ainsi, pour qu'un champ magnétique soit généré et maintenu, il est nécessaire que le terme non-linéaire $(\nabla \times (u \times b))$ dépasse la dissipation par effet Joule $(-\nabla \times \frac{j}{\sigma})$. Alors, pour un système de taille caractéristique L et vitesse caractéristique V, il est à minima nécessaire d'avoir :

$$\left[\frac{\nabla \times (u \times \mu H)}{\nabla \times ((\nabla \times H)/\sigma)}\right] \sim \sigma\mu v L = R_{\mathrm{m}} > 1. \tag{6.7}$$

R_{m} est connu sous le nom de nombre de Reynolds magnétique et donne une borne inférieure pour la génération d'un champ magnétique. En effet, pour un champ de vitesse fixe, l'équation (6.2) devient linéaire en b et donc

R_m devient le seul paramètre de contrôle. Dans ce cadre de travail avec une dynamo provenant d'un champ de vitesse déterminé (appelé dynamo cinématique), plusieurs champs ont pu être analytiquement dérivés (dynamo de Ponomarenko [27], dynamo de Roberts [28]) et reproduits expérimentalement [1, 2].

Cependant, les effets dynamo les plus intéressants proviennent du couplage non-linéaire entre l'équation du mouvement et celle d'induction. Pour ce système couplé, il n'existe pas de théorème donnant les circonstances permettant d'obtenir un champ magnétique. Pire encore, il existe de nombreux théorèmes anti-dynamo (des conditions pour lesquelles une dynamo ne peut pas apparaître peu importe la valeur de R_m) [10]. Par exemple, le théorème de Cowling [29, 30] stipule qu'un champ magnétique purement axi-symétrique ne peut jamais être produit quel que soit le champ de vitesse.

Le second paramètre de contrôle intéressant pour l'expérience VKS est le nombre de Reynolds (cinétique) pour un fluide turbulent. Il caractérise le degré de turbulence de l'écoulement en comparant le terme inertiel ($\partial_i(u_i u)$) avec celui de dissipation visqueuse ($\nu\Delta u$) dans l'équation (6.1):

$$\left[\frac{\partial_i(u_i u)}{\nu\Delta u}\right] \sim \frac{vL}{\nu} = R_\mathrm{e}. \tag{6.8}$$

Plus le nombre de Reynolds est grand, plus il est possible d'observer des structures petites en taille comparées à la taille du système.

Ces deux nombres de Reynolds sont liés par le nombre de Prandtl magnétique qui ne dépend alors que des propriétés physiques du fluide :

$$P_m = \frac{R_\mathrm{m}}{R_\mathrm{e}} = \mu\sigma\nu. \tag{6.9}$$

Comme le fluide dans l'expérience VKS est fixé (sodium liquide), la seule façon de maximiser les chances d'obtenir un champ magnétique est d'augmenter la vitesse du fluide. Pour le sodium liquide à 120°C : $\nu = 7\,10^{-7}\,\mathrm{m^2\,s^{-1}}$, $\sigma = 9.6\,10^{6}\,\mathrm{S\,m^{-1}}$ et $\mu = \mu_0 = 4\pi\,10^{-7}\,\mathrm{H\,m^{-1}}$. Cela donne $P_m = 0.8\,10^{-5}$ et impose donc $R_\mathrm{e} \gtrsim 10^5$ pour avoir une chance d'observer un champ magnétique. Cependant, dans l'expérience, il a été nécessaire de monter à $R_\mathrm{e} \gtrsim 6\,10^6$ pour générer un champ magnétique. Ce grand nombre de Reynolds implique alors que quelque chose se passe au niveau des petites échelles pour l'obtention d'un champ magnétique.

6.1.3 . Simulations numériques

Pour étudier les équations de la magnétohydrodynamique et les mécanismes dynamo, de nombreux codes numériques existent car chaque configuration nécessite son propre cadre numérique. Ainsi, en fonction de l'objet d'étude, un système de dynamo cinématique ou les équations complètes de la MHD sont calculés. Quand l'objectif est d'étudier le seuil de l'instabilité, le

champ de vitesse peut être donné (issu de résultats expérimentaux, de modèles analytiques ou de solveurs de Navier-Stokes) et seule l'équation (6.1) est résolue en faisant varier le nombre de Reynolds magnétique. Le nombre de Reynolds critique R_{m}^{c} est alors obtenu quand la solution $b = 0$ devient instable. Quand l'objectif est d'étudier les phases de croissance et de saturation du champ magnétique généré, il est nécessaire de simuler les équations complètes de la MHD (1.15-1.16).

Pour l'étude de la dynamo de Ponomarenko, un code spectral 1D pour la dynamo cinématique a été développé [31]. Dans [32], un code pseudo-spectral parallélisé pour les équations de la MHD dans une boite périodique est utilisé pour mieux comprendre la dynamo de Roberts.

Pour des écoulements plus complexes, des codes possédant au moins deux directions périodiques ont été développés : dans [33] pour l'étude de la dynamo provoquée par un écoulement cisaillé au sein des disques d'accrétion des galaxies et dans [34] pour la compréhension de l'action dynamo lors de la rotation de deux couches convectives. Pour la simulation de cylindres infinis (conditions périodiques en θ et z), différents codes ont été développés [35, 36, 37] dont le dernier a permis l'étude de la dynamo de Taylor-Couette. Il existe aussi de nombreux codes dédiés à la géométrie sphérique : codes pseudo-spectraux utilisant en général une décomposition poloidale-toroidale de u et b [38, 39, 40, 41], ou des algorithmes utilisant des volumes finis [42], ou des éléments finis [43] et même des codes commerciaux comme COMSOL utilisés dans [44]. Cependant, pour simuler des domaines finis, seulement quelques-uns existent : pour des sphéroïdes [45], pour un tore [46], et pour des cylindres [47, 48].

Pour mieux comprendre l'expérience VKS et les mécanismes dynamo y prenant place, de nombreuses simulations numériques ont été faites. Des codes périodiques simples de MHD complets dans des géométries Cartésiennes ont permis d'estimer le nombre de Reynolds magnétique critique et d'amener à l'observation de cycles d'hystérésis liés à la force de Lorentz ainsi que les conditions pour de la dynamo non-linéaire [49, 50, 51, 52]. Un code cinématique pour des cylindres infinis a permis d'optimiser le choix des turbines pour l'expérience VKS et d'obtenir les premiers résultats sur la structure du champ magnétique généré [53]. Des codes similaires mais pour des cylindres finis ont alors montré l'impact négatif sur la dynamo des couches de fluide derrière les turbines [54, 55]. Cependant, la structure du champ magnétique obtenue était complètement différente de celle qui a été observée dans l'expérience un peu plus tard : le champ magnétique numérique était dominé par une composante perpendiculaire à l'axe du cylindre et une autre parallèle à cet axe (soit par un mode azimutal $m = 1$ compatible avec le théorème anti-dynamo de Cowling).

Ensuite, en introduisant les effets α et Ω dans des simulations de dy-

namo cinématique, la structure moyenne du champ magnétique observée dans l'expérience a été reproduite [56, 57, 17]. Cependant, elle nécessite soit des valeurs très grandes de α ou un effet α homogène qui représentent des hypothèses irréalistes. Finalement, des simulations numériques directes (DNS) et avec modèle de turbulence (LES) intégrant l'ensemble des équations de la MHD et imposant des turbines en contra-rotation ont été développées [58].

Dans cette thèse nous effectuons de nouvelles simulations avec des paramètres déjà utilisés dans [58]. Elles sont réalisées avec le code SFEMaNS [59] pour les raisons suivantes : il utilise les coordonnés cylindriques permettant de représenter simplement le cylindre, les turbines sont correctement modélisées grâce à une méthode de pénalisation (voir [58] p.5), des données de simulations précédentes [58] sont accessibles et certains outils pour l'étude de la turbulence y sont déjà implémentés [60]. La simulation d'un tour complet des turbines nécessite en moyenne 10 heures de simulation sur 2040 processeurs du supercalculateur Jean-Zay fournis par l'IDRIS. La géométrie de la simulation (fig. 6.1) est une approximation de l'expérience. Le volume interne pour $r \in [0, 1]$ contient le sodium liquide alors que la partie externe $r \in [1, 1.6]$ est composée de deux couches de solides conducteurs : une couche avec la même conductivité que le liquide pour $r \in [1, 1.4]$ (modélisant une couche de sodium immobile) et une autre avec une conductivité 4.5 fois supérieure pour $r \in [1.4, 1.6]$ (modélisant la paroi en cuivre). Les turbines ont une perméabilité différente en fonction du matériau utilisé.

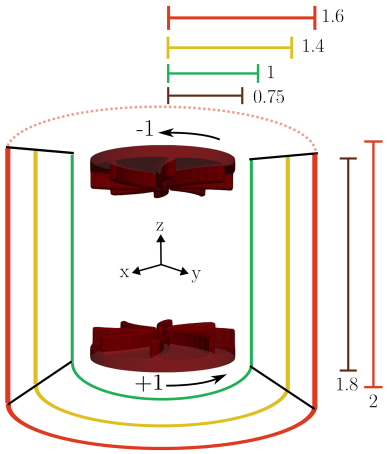


Figure 6.1: Représentation des différents volumes simulés : fluide en mouvement (vert), fluide au repos (jaune), autres murs conducteurs (rouge) et turbines en mouvement (rouge foncé).

Dans ce cadre de simulation il est alors possible d'observer la génération d'un champ magnétique très proche des résultats expérimentaux [58] en utilisant des DNS pour les faibles nombres de Reynolds ou des LES pour les

grands nombres de Reynolds [61] (le plus grand R_e simulé étant de 10^5 et correspondant alors à un nombre de Prandtl magnétique de $P_m^c \simeq 7\,10^{-4}$ au seuil dynamo). Dans cette thèse, nous utilisons les résultats obtenus pour $R_e = 1500$ dans le cadre d'une DNS pour pouvoir étudier les transferts d'énergie jusqu'aux plus petites échelles sans modélisation. Ce nombre de Reynolds est assez faible comparé au cas expérimental mais donne tout de même une idée concrète des ingrédients nécessaires à l'apparition de l'effet dynamo. En particulier, il a été observé que l'utilisation de turbines avec une haute perméabilité magnétique diminue le nombre de Reynolds magnétique critique pour lequel un champ magnétique est produit. Ainsi, comme observé dans l'expérience, le seuil pour lequel un champ magnétique est généré dépend du matériau utilisé pour les turbines. Cependant, l'origine des mécanismes dynamo reste un mystère.

6.1.4 . Plan de la thèse

Pour mieux comprendre ces résultats numériques et donc d'où provient l'effet dynamo dans l'expérience VKS, nous implémentons les bilans d'énergie locaux dérivés des équations de la MHD dans le code SFEMaNS.

Avant de proposer un résumé de chacun des chapitres de cette thèse, nous décrivons sa structure.

Dans le premier chapitre-2 résumé dans la section 6.2, nous décrivons le programme utilisé ainsi que son contexte numérique. Nous présentons aussi quelques améliorations faites pour la répartition de la grille de simulations sur les processeurs ainsi qu'une étude sur une méthode potentiellement plus rapide pour simuler les équations.

Dans le second chapitre-3 résumé dans la section 6.3, nous décrivons les différents outils de post-traitement permettant de calculer, analyser et visualiser les transferts d'énergie locaux. Ces outils sont essentiels pour que l'étude des ces termes soit automatisée et efficace.

Finalement, dans le chapitre-4 résumé dans la section 6.4, nous dérivons les équations pour les transferts d'énergie ainsi que les simulations utilisées. Nous analyserons alors les différents transferts pour deux cas différents : un pour des turbines en acier et l'autre pour des turbines en fer doux.

L'objectif de cette thèse est ainsi de découvrir quels sont les mécanismes dynamo responsables de la génération du champ magnétique dans les fluides conducteurs turbulents.

6.2 . Améliorations du code SFEMaNS

Pour simuler les équations de la magnétohydrodynamique et calculer les différents transferts d'énergie, nous utilisons le code SFEMaNS.

Cette section est un résumé du chapitre 2 de ce manuscrit donnant une

brève description du code SFEMaNS ainsi que quelques améliorations que nous y avons apportées. Après une description des approximations numériques utilisées, nous présentons une nouvelle méthode pour répartir la grille de simulation sur les processeurs. Finalement, nous étudions une autre méthode pour simuler les équations. Tout ce travail a été fait dans le cadre d'une collaboration avec J.-L. Guermond qui m'a invité à Texas A&M University du 5 Janvier au 3 Février 2022 et du 15 Juillet au 3 Décembre 2022.

La code SFEMaNS [59] est écrit en Fortran-90 et est dédié aux systèmes axi-symétriques. Il utilise les coordonnées cylindriques et décompose alors les champs sur une base d'éléments finis dans le plan (r, z) et en mode de Fourier suivant θ. Cela permet une grande parallélisation grâce à l'utilisation du module MPI pour les modes de Fourier et de PETSC pour la parallélisation en domaines sur les éléments finis.

Le volume de simulation que nous utilisons est montré en Fig. 1.1 où la rotation des turbines est implémentée grâce à une méthode de pénalisation (voir [58] p.5). La plupart des fonctions pour des calculs simples et de parallélisation sont déjà implémentées. Cependant, il reste à construire les outils pour calculer les nouveaux termes de transferts d'énergie.

6.2.1 . Approximations numériques dans SFEMaNS

Le code utilise une décomposition hybride pour les champs. L'utilisation des coordonnées cylindriques permet une décomposition en M modes de Fourier suivant θ permettant alors de réduire le problème 3D à des résolutions de problèmes 2D dans le plan (r, z). Ces problèmes sont résolus en utilisant le cadre des éléments finis et de la formulation faible des équations.

Les méthodes des éléments finis permettent d'approximer l'espace des fonctions par un sous-ensemble de fonctions appelées fonctions tests ϕ_i. Ces fonctions sont définies sur une grille (voir Fig. 6.4) composée de triangles 2D telle que, quand la taille caractéristique des cellules tend vers zéro, les fonctions tests génèrent l'espace entier des fonctions. Les grilles de simulations pour les champs magnétique et de vitesse sont en éléments $\mathbb{P}_2$ et pour la pression $\mathbb{P}_1$. Cela est dû au fait que seules certaines paires d'éléments permettent d'assurer la convergence vers des solutions propres de Navier-Stokes.

Ces combinaisons permettent alors d'écrire toute équation sous la forme d'un problème simple d'algèbre linéaire. Par exemple, pour l'évolution temporelle du champ de température pour un fluide :

$$\partial_t T + \nabla \cdot (uT) - K\Delta T = f, \tag{6.10}$$

où u est le champ de vitesse, f une source thermique et K la diffusivité thermique du fluide, cela revient à résoudre pour chaque mode de Fourier m et fonction test ϕ_i:

$$A_{ij}^{m,\mathsf{c}} T_j^{n+1,m,\mathsf{c}} = B_i^{m,\mathsf{c}}, \tag{6.11}$$

pour la partie cosinus et un pas de temps temporel n, avec:

$$A_{ij}^{m,\mathsf{c}} = \int_{r,z} r\left[\frac{3}{2\tau}\psi_j\psi_i + K(\partial_r\psi_j\partial_r\psi_i + \partial_z\psi_j\partial_z\psi_i + \frac{m^2}{r^2}\psi_j\psi_i)\right], \tag{6.12}$$

$$B_i^{m,\mathsf{c}} = \sum_{j=1}^{n_p}\int_{r,z} rF_j^{m,\mathsf{c}}\psi_j\psi_i, \tag{6.13}$$

où $F = \left(\frac{4T^n - T^{n-1}}{2\tau} + f^{n+1} - \nabla\cdot((2u^n - u^{n-1})(2T^n - T^{n-1}))\right)$.

Ce système linéaire est alors découplé en modes de Fourier permettant une parallélisation efficace. De plus, comme il s'agit d'une simple inversion de matrice, des outils efficaces d'algèbre linéaire peuvent être utilisés. Pour évaluer les intégrales dans A et B, une quadrature de Gauss est utilisée qui permet d'approximer l'intégrale en utilisant une somme pondérée de points bien choisis.

6.2.2 . Amélioration de la distribution du maillage de simulation

Une des principales limites des grilles de simulation dans SFEMaNS est leur utilisation de la mémoire. Le problème provient du fait que chaque processeur garde en mémoire l'intégralité de la grille de simulation et pas seulement la partie lui étant attribuée. Comme chaque processeur ne posséde que de 4GB à 8GB de mémoire RAM, il est impossible de simuler des grilles 2D avec plus de 500.000 points. Ceci limite sévèrement le nombre de Reynolds cinétique maximal pouvant être simulé et est très inefficace pour la mémoire rendant difficile le calcul de nouvelles quantités. Cette limite est par ailleurs d'autant plus facile à atteindre pour les équations de la MHD car elles nécessitent trois grilles (pour le champ de vitesses, de pression et magnétique).

Pour résoudre ce problème, nous avons implémenté une nouvelle méthode pour distribuer la grille de telle manière que chaque processeur ne conserve en mémoire que les cellules et points lui étant strictement nécessaires pour construire l'algèbre linéaire. Cette nouvelle implémentation permet d'obtenir des simulations contenant 300.000 points de simulation par processeur pour des problèmes de Navier-Stokes 2D. Ainsi, en utilisant de nombreux processeurs, il nous est possible de faire des simulations avec des centaines de millions de points sur nos maillages 2D.

Deux principales difficultés ont été rencontrées. Premièrement, il est nécessaire de respecter les conventions (figure 6.2) sur la numérotations des points, voisins et bords des cellules pour que les algèbres linéaires soient construites correctement. Deuxièmement, à cause de la parallélisation, certains processeurs ont besoin de l'information de cellules ne leur étant pas attribuées. Par exemple, dans la figure 6.3 , le processeur 0 (bleu foncé) a besoin de l'information des cellules en bleu clair, rouge et jaune contenant un des gros points bleus. Ce problème est résolu en sauvegardant pour chaque processeur

ces cellules voisines de manière à pouvoir reconstruire les interactions entre ces cellules. De plus, pour des raisons techniques, il est nécessaire de construire trois numérotations pour chacun des éléments de la grille (points, segments et cellules) : une numérotation globale, une au niveau du processeur et une au niveau de la cellule.

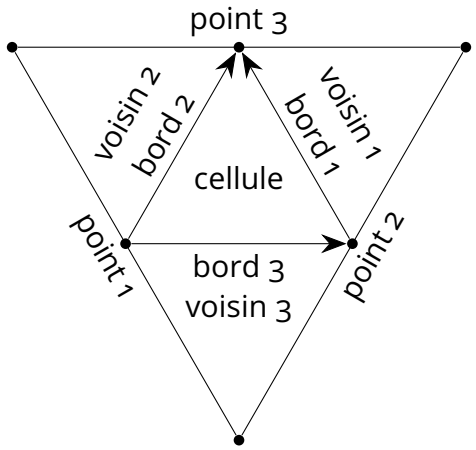


Figure 6.2: Convention pour les points, bord et voisins d'une cellule.

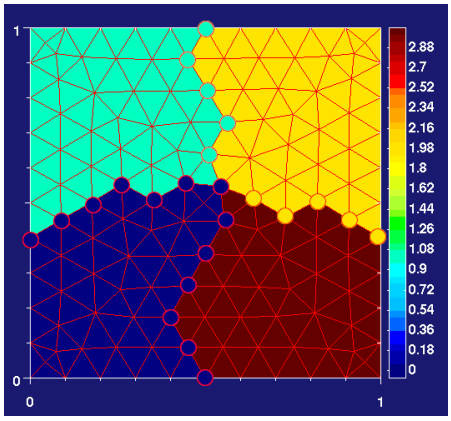


Figure 6.3: Distribution des points sur différents processeurs : processeur 0 en bleu foncé, processeur 1 en bleu clair, processeur 2 en jaune, processeur 3 en rouge.

Une fois cette distribution astucieuse faite sur les processeurs, il reste à raffiner le maillage (fig. 6.4) pour pouvoir simuler des nombres de Reynolds plus élevés. Ceci est fait en redécoupant chaque triangle en quatre. Bien sûr il faut aussi le faire pour les cellules supplémentaires sauvegardées par chaque processeur à cause de la parallélisation tout en respectant toutes les conventions sur l'indexation des éléments des cellules. Ceci est fait en suivant le diagramme 6.5 et permet alors d'obtenir des raffinements successifs (fig. 6.4)

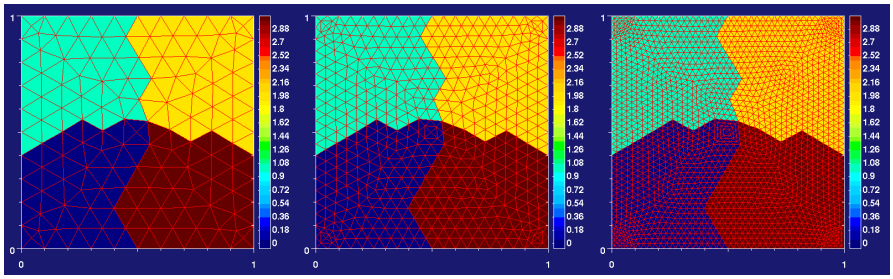


Figure 6.4: Raffinements d'un maillage. Pas de raffinement (panneau gauche), un raffinement (panneau central), deux raffinements (panneau de droite).

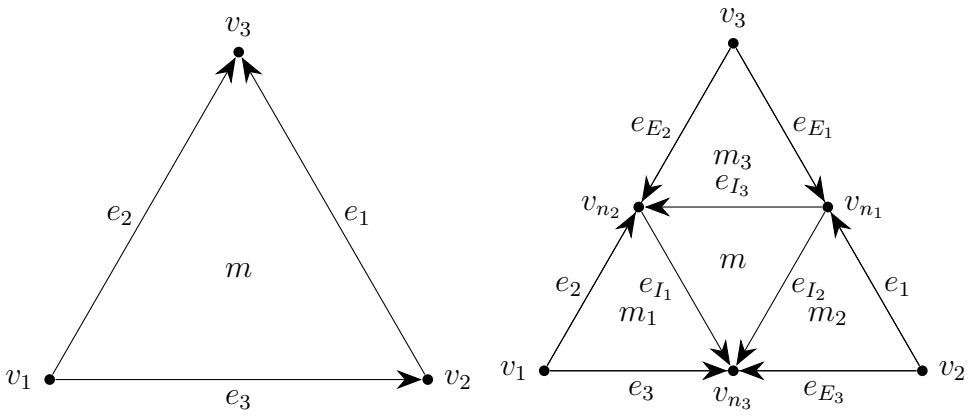


Figure 6.5: Raffinement d'une cellule (panneau gauche) en quatre cellules (panneau droit). La numérotation des cellules, points et bords ainsi que l'orientation des bords sont montrées pour chaque cellule.

Il ne reste plus qu'à construire les éléments finis $\mathbb{P}_n$, c'est-à-dire rajouter des points dans toutes les cellules en respectant leur numérotation locale (fig. 6.6).

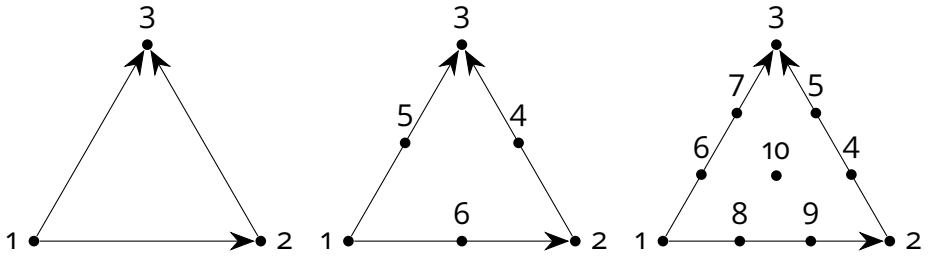


Figure 6.6: Numérotations des points et orientation des bords pour une cellule pour les éléments finis $\mathbb{P}_1$ (panneau gauche), $\mathbb{P}_2$ (panneau central) et $\mathbb{P}_3$ (panneau droit).

6.2.3 . Étude de méthodes avec préconditionnement pour résoudre Navier-Stokes

Cette section est un bref résumé de la section 2.3 composée d'un article soumis à Computer Methods in Applied Mechanics and Engineering (Octobre 2023).

Le but est d'implémenter et tester des méthodes de préconditionnements pour résoudre le problème de Stokes dans le cadre de SFEMaNS. L'objectif final est de voir si ces méthodes sont plus rapides que celle de prédiction-correction pour l'instant utilisée dans SFEMaNS. Cela pourrait économiser beaucoup de temps de calcul mais la conclusion est que, pour l'instant, cela n'est pas le cas.

Pour des raisons de simplicité, l'aspect mathématique est grossièrement vulgarisé.

Contexte

Résoudre le problème de Stokes généralisé a une longue histoire commençant par Fortin et Glowinski [64] et par Cahouet et Chabart [65]. Dans [64], les auteurs ont introduit la méthode du Lagrangien augmenté permettant d'améliorer le conditionnement du complément de Schur en pression. Dans [65], les auteurs ont proposé un préconditionnement du problème de Stokes se comportant bien dans les deux limites $\mu\tau h^{-2} \to 0$ et $\mu\tau h^{-2} \to \infty$ avec μ la viscosité, τ le pas de temps et h la taille caractéristique des éléments du maillage de simulation. Ces méthodes sont par la suite combinées [66, 67, 68] donnant alors un bon préconditionnement pour le problème de Schur généralisé. Ces différentes approches sont testées numériquement dans [69, 70, 71] et montrent en effet que le nombre de MINRES ou GMRES itérations pour obtenir une solution se comporte correctement en fonction de la viscosité et de la taille caractéristique de la grille de simulation.

Le problème de Stokes correspond à résoudre pour la pression p et la vitesse u :

$$\partial_t u - \frac{\mu}{4}[(\nabla + \nabla^{\mathsf{T}}) : (\nabla + \nabla^{\mathsf{T}})]u + \nabla p = f, \nabla \cdot u = 0 \tag{6.14}$$

Le terme $\frac{\mu}{4}(\nabla + \nabla^{\mathsf{T}}) : (\nabla + \nabla^{\mathsf{T}})$ correspond au Laplacien plus les termes extra-diagonaux en $\partial_x \partial_y$ pour le cas 2D et f une force externe.

Dans sa forme discrétisée, le problème de Stokes s'écrit (U, P et F étant respectivement la vitesse, la pression et la force externe, discrétisées) :

$$A\mathsf{U} - B^{\mathsf{T}}\mathsf{P} = \mathsf{F}, \qquad B\mathsf{U} = 0, \tag{6.15}$$

avec B correspondant à la matrice en éléments finis définissant le gradient ∇ et $A := \frac{1}{\tau} M_{\boldsymbol{V}} + \mu E_{\boldsymbol{V}}$ où $M_{\boldsymbol{V}}$ correspond à la matrice identité du maillage en vitesse en éléments finis (appelée matrice de masse) et $E_{\boldsymbol{V}}$ la matrice associée

à l'opérateur $\frac{1}{4}(\nabla + \nabla^{\mathsf{T}}) : (\nabla + \nabla^{\mathsf{T}})$. Ainsi il est possible de résoudre ce système par inversion de matrices.

Le complément de Schur de ce problème est alors défini par $S := BA^{-1}B^{\mathsf{T}}$ et rend le système (6.15) équivalent à :

$$S\mathsf{P} = -BA^{-1}\mathsf{F}, \qquad A\mathsf{U} = \mathsf{F} + B^{\mathsf{T}}\mathsf{P}. \tag{6.16}$$

Il est alors possible de résoudre le problème en pression indépendamment de la vitesse. La difficulté pour inverser S (et donc résoudre le problème) provient du fait que cette matrice est très loin d'être diagonale et prend donc beaucoup d'itérations pour des solveurs itératifs comme GMRES et MINRES. De plus, comme il n'est pas possible de construire les éléments de la matrice S pour des raisons de mémoire, on ne peut pas utiliser des méthodes de résolution plus directes. Mathématiquement, la quantité d'intérêt est le nombre de conditionnement ℓ^2 de S étudié dans [72, Prop. 5.24] et dans [73, Prop. 50.14].

Pour diminuer ce nombre de conditionnement, la méthode du Lagrangien augmenté a été développée dans [64] et transforme le système (6.15) en un système équivalent (donc possédant les mêmes solutions) :

$$(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)\mathsf{U} - B^{\mathsf{T}}\mathsf{P} = \mathsf{F}, \qquad B\mathsf{U} = 0, \tag{6.17}$$

où λ est un paramètre adimensionné positif et M_Q la matrice de masse pour la pression. Il est de nouveau possible de construire un complément de Schur pour ce système et ainsi de simplement résoudre :

$$S_{\lambda\mu} := B(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)^{-1}B^{\mathsf{T}}, \tag{6.18a}$$

$$S_{\lambda\mu}\mathsf{P} = -B(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)^{-1}\mathsf{F}, \qquad A\mathsf{U} = \mathsf{F} + B^{\mathsf{T}}\mathsf{P}. \tag{6.18b}$$

Cette méthode permet de réduire considérablement le nombre de conditionnement quand λ augmente (voir [74, Prop. 2.1] et [66, Lem. 4.1]). Cependant, cela rend bien plus compliqué le calcul de $(A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B)^{-1}$.

Préconditionneur

Une autre méthode pour diminuer le nombre d'itérations nécessaires pour inverser le problème Schur est de tout d'abord le multiplier par une matrice permettant de le rapprocher de l'identité. Pour le problème (6.16), dans [65], les auteurs proposent de multiplier l'équation en pression par une de ces deux matrices :

$$\mathsf{CaCh}_1 := \tau^{-1}(B\, M_{\boldsymbol{V}}^{-1}B^{\mathsf{T}})^{-1} + \mu M_Q^{-1}, \tag{6.19a}$$

$$\mathsf{CaCh}_2 := \tau^{-1}(L_Q)^{-1} + \mu M_Q^{-1}. \tag{6.19b}$$

De notre côté, pour le problème avec le Lagrangien augmenté (6.18b), nous proposons :

$$C_{\lambda\mu} := \tau^{-1}(B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}})^{-1} + \mu(1+\lambda)M_Q^{-1}. \tag{6.20}$$

Dans la limite $\mu\tau/h^2 \to 0$, on obtient $S_{\lambda\mu} \to \tau B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}}$ et donc $S_{\lambda\mu}$ devient $\tau^{-1}(B(M_{\boldsymbol{V}})^{-1}B^{\mathsf{T}})^{-1}$. Ainsi, en multipliant $C_{\lambda\mu}S_{\lambda\mu} \sim I_d$ dans cette limite.

De même, dans la limite $\mu\tau/h^2 \to \infty$, on obtient $S_{\lambda\mu} \to \mu^{-1}B(E_{\boldsymbol{V}} + \lambda B^{\mathsf{T}}M_Q^{-1}B)^{-1}B^{\mathsf{T}}$. Comme $E_{\boldsymbol{V}}$ et $B^{\mathsf{T}}M_Q^{-1}B$ possèdent des valeurs similaires, on peut approximer $S_{\lambda\mu}$ dans la limite $\mu\tau/h^2 \to \infty$ par $\mu^{-1}(1+\lambda)^{-1}B(B^{\mathsf{T}}M_Q^{-1}B)^{-1}B^{\mathsf{T}}$. Alors l'inverse de $S_{\lambda\mu}$ se comporte approximativement comme $\mu(1+\lambda)M_Q^{-1}$ ce qui correspond à l'inverse de $C_{\lambda\mu}$ dans cette même limite.

Ainsi, pour ces deux limites, nous nous attendons à un nombre très faible d'itérations nécessaires pour des méthodes de résolution itérative comme GMRES. Des arguments similaires marchent aussi pour les préconditionneurs 6.19a et 6.19b pour le problème sans Lagrangien augmenté 6.16.

Cadre numérique

Pour tester ces méthodes, nous utilisons des éléments finis continus de Lagrange en 2D sur des grilles non-structurées qui sont des raffinements successifs de la figure 6.4. Les éléments finis utilisés dans ce résumé sont les éléments de Taylor-Hood $\mathbb{P}_2/\mathbb{P}_1$. Pour chaque raffinement, le nombre de processeurs est multiplié par 4 pour conserver un nombre de points par processeur équivalent. Le maillage testé le plus fin posséde environ 120 millions de points pour le maillage en vitesse. Le tout est codé dans SFEMaNS et utilise donc PETSC pour la parallélisation en domaines sur la grille 2D.

En fonction des situations, nous utilisons soit un solveur linéaire direct MUMPS [75] ou algébrique multigrilles BoomerAMG [76] pour inverser les matrices dont il est possible de construire les coefficients. Pour des raisons de reproductibilité voici les paramètres utilisés pour BoomerAMG :

`-pc_type`	`hypre`
`-pc_hypre_type`	`boomeramg`
`-pc_hypre_boomeramg_strong_threshold`	`0.7 or 0.1`
`-pc_hypre_boomeramg_coarsen_type`	`Falgout`
`-pc_hypre_boomeramg_relax_type_all`	`Chebyshev`

Table 6.1: Options BoomerAMG utilisées pour tester les préconditionneurs.

Choix du préconditionneur pour le problème en vitesse

Pour résoudre le problème en pression, il est nécessaire d'inverser la matrice $A + \lambda\mu B^{\mathsf{T}}M_Q^{-1}B$. Cependant, il n'est pas possible de construire les coefficients de cette matrice et il est donc nécessaire d'utiliser une méthode itérative. Il s'agit alors de résoudre un problème en vitesse pour X : $(A +$

$\lambda\mu B^{\mathsf{T}} M_Q^{-1} B)X = Y$. Pour accélérer cette inversion, un préconditionnement est nécessaire et nous étudions les préconditionneurs suivants :

$$\tau^{-1} M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}} + \mu(1+\lambda)D = A + \lambda\mu D, \tag{6.21}$$
$$\tau^{-1} M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}} + \mu D = A, , \tag{6.22}$$
$$\tau^{-1} M_{\boldsymbol{V}} + \mu L_{\boldsymbol{V}} ., \tag{6.23}$$

avec D les termes non diagonaux de la matrice Laplacien $L_{\boldsymbol{V}}$. Ces trois expressions semblent a priori de bonnes approximations de $A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B$.

Pour comparer leur efficacité, nous mesurons combien de temps par degré de liberté est nécessaire pour résoudre le problème. Les résultats pour ces différents préconditionneurs sont reportés dans la figure 6.7. Les deux types de solveur MUMPS et BommerAMG sont testés pour inverser le préconditionneur (6.23). Clairement seul le préconditionneur (6.23) permet d'obtenir une bonne parallélisation pour des maillages de plus en plus fins (le temps par degré de liberté restant constant). Aussi la méthode directe MUMPS ne marche plus au-delà de 2 millions de points par manque de mémoire vive.

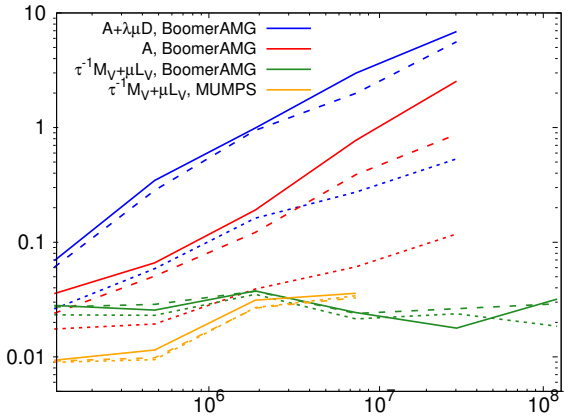
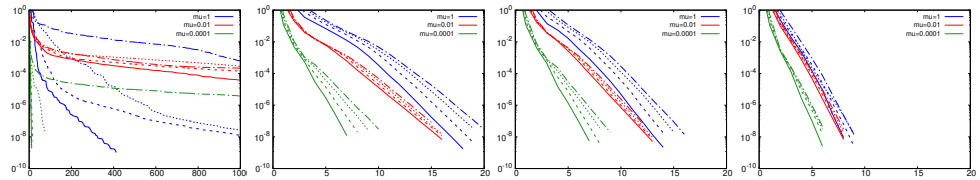


Figure 6.7: Preconditionement de $A + \lambda\mu B^{\mathsf{T}} M_Q^{-1} B$ pour différents préconditionneurs et solveurs. Axe des Y : $\frac{\text{temps de simulation} \times \text{nb. processeurs}}{\text{nb. vel.+press. degrés de libertés}}$ en millisecondes ; axe des X : nombre de points pour le maillage de vitesse. $\mu = 1$ (——); $\mu = 10^{-2}$ (- - -), $\mu = 10^{-4}$ (·····).

Résultats et conclusion pour le problème de Stokes

Nous étudions maintenant les performances des préconditionneurs pour le problème de Stokes dans sa totalité. Pour les préconditionneurs CaCh_1 (6.19a) et CaCh_2 (6.19b), le problème sans Lagrangien augmenté (6.16) est résolu. Et, pour le préconditionneur $C_{\lambda\mu}$ (6.20), le problème avec Lagrangien augmenté (6.18b) est résolu en utilisant le préconditionneur (6.23) pour le problème en vitesse.

Dans la figure 6.8, nous montrons le résidu de la méthode GMRES (c'est-à-dire la précision obtenue sur la résolution du problème) en fonction de l'itération GMRES atteinte. Pour atteindre la précision relative demandée (de 10^{-10}) lors de la résolution itérative du problème, nous observons bien une diminution du nombre d'itérations GMRES pour le préconditionneur $C_{\lambda\mu}$ (6.23). Il est par ailleurs plus faible en prenant $\lambda = 10$ plutôt que $\lambda = 1$. Par contre, le nombre d'itérations pour CaCh$_1$ (6.19a) diverge sur les maillages fins.



$\lambda = 0$ $\lambda = 1$ $\lambda = 10$

Figure 6.8: Résidu GMRES vs. nombre d'itérations pour le préconditionneur $\tau^{-1}(B\ M_{\boldsymbol{V}}^{-1}B^{\mathsf{T}})^{-1} + \mu(1+\lambda)M_Q^{-1}$ avec $\mu = 1$ (bleu), $\mu = 10^{-2}$ (rouge), et $\mu = 10^{-4}$ (vert). Maillage 0.1M pts. vit. (——); Maillage 0.5M pts. vit. (- - -), Maillage 1.9M pts. vit. (·····); Maillage 7.6M pts. vit. (– · –). Préconditionneur de gauche à droite: Cahouet&Chabard CaCh$_1$ ($\lambda = 0$), Cahouet&Chabard CaCh$_2$ ($\lambda = 0$), Lagrangien augmenté $C_{\lambda\mu}$ avec $\lambda = 1$, Lagrangien augmenté $C_{\lambda\mu}$ avec $\lambda = 10$.

Cependant, en regardant l'efficacité en temps de calcul (fig. 6.9), les méthodes avec le moins d'itérations se retrouvent avec des performances similaires de l'ordre de la milliseconde par degré de liberté. Ce même ordre de grandeur est obtenu dans la littérature [68, 69, 70]. Les performances en parallèle sont globalement bonnes (peu de variation du temps de calcul par degré de liberté en raffinant le maillage).

Pour conclure, l'efficacité en temps pour résoudre les problèmes de Stokes et de Navier-Stokes en utilisant les techniques de complément de Schur n'est pas, pour l'instant, meilleure que des méthodes de prédiction-correction (en vitesse ou pression). Ces méthodes de prédiction-correction [77, 78, 79] ayant une efficacité d'environ 10^{-2}ms par degré de liberté, il semble difficile de gagner deux ordres de grandeur même en optimisant les paramètres des préconditionneurs.

6.3 . Outils de post-traitement

Le code SFEMaNS permet de base de simuler les équations de la MHD et donne donc l'accès aux champs de vitesse u, de pression p et au champ magnétique. Ainsi, il est nécessaire d'implémenter des outils pour calculer, étudier et visualiser les transferts d'énergie. Dans un premier temps, il est nécessaire de calculer ces quantités (voir équations 6.31 et 6.32) dans SFEMaNS. Ensuite,

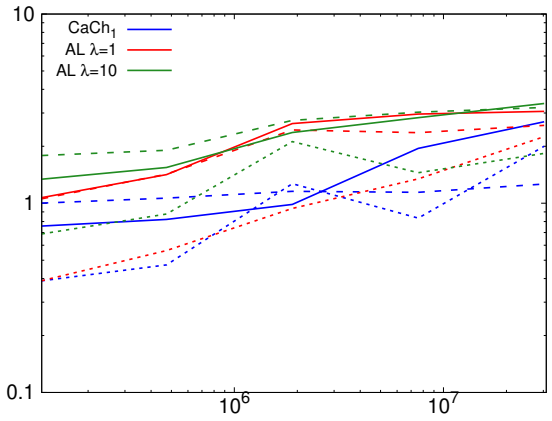


Figure 6.9: Préconditionnement de $\tau^{-1}(B\, M_{\boldsymbol{V}}^{-1} B^{\mathsf{T}})^{-1} + \mu(1+\lambda) M_Q^{-1}$ par différents préconditionneurs. Axe des Y : $\frac{\text{temps de simulation} \times \text{nb. processeurs}}{\text{nb. vel.+press. degrés de liberté}}$ en millisecondes ; axe des X : nombre de points pour le maillage de vitesse. $\mu = 1$ (——); $\mu = 10^{-2}$ (- - -), $\mu = 10^{-4}$ (·····).

il faut transformer les données en sortie de SFEMaNS dans des formats de fichiers permettant l'analyse statistique ou la visualisation. Finalement, au vu des grandes quantités de données, il est important d'automatiser la visualisation et l'analyse des champs obtenus. Cette automatisation est faite grâce au couplage entre Python et Paraview.

Pour facilement transférer ces outils d'une plateforme à l'autre et les utiliser, ils sont tous disponibles pour notre groupe de recherche dans un GitLab privé en tant que projets différents. Un fichier README.md est inclus dans chacun de ces projets contenant un tutoriel simple. L'objectif de cette section est de résumer les principaux paramètres de ces outils développés en plus de détails dans le chapitre 3.

6.3.1 . Méthodes pour le calcul des termes de transferts d'énergie

Dans un but de conserver une haute précision et une grande efficacité, les différents termes de transferts énergétiques sont calculés en utilisant le code SFEMaNS. Le projet correspondant se nomme 'DR_ MHD' dans GitLab. Toutes les options utiles pour faire fonctionner le code sont dans le fichier 'data_example' et il suffit de le copier-coller dans le fichier 'data' utilisé pour les simulations temporelles.

Pour pouvoir analyser ce qu'il se passe à différentes échelles, il est nécessaire de les séparer. Ceci est en particulier important pour trouver les évènements rares et intenses de la turbulence pouvant être des indices de quasi-singularités ou des processus de dynamo rapide. Pour faire cette séparation, nous utilisons un filtrage se présentant sous la forme d'une résolution d'équation différentielle. Ce choix de filtre permet d'avoir une méthode qui est rapide et efficace sur les grilles non-structurées comme c'est le cas pour SFEMaNS.

Dans le cadre de SFEMaNS, résoudre des équations différentielles linéaires est facile. Ainsi, nous définissons un champ filtré $g = \overline{u}^{\ell}$ comme la solution de :

$$g - \ell^2 \Delta g = u, \tag{6.24}$$

avec u un vecteur ou scalaire et $\ell \in \mathbb{R}^+$ la taille du filtre.

Le filtrage est alors transformé en un problème d'algèbre linéaire similaire à ce qui est fait dans la section 6.2.1. L'équation (6.24) devient alors, pour un mode de Fourier m et la partie cosinus d'un scalaire ou d'un vecteur :

$$A_{ij}^{m,\mathsf{c}} g_j^{m,\mathsf{c}} = u_i^{m,\mathsf{c}}, \tag{6.25}$$

avec

$$A_{ij}^{m,\mathsf{c}} = \int_{r,z} r(\psi_j \psi_i + \ell^2 (\partial_r \psi_j \partial_r \psi_i + \partial_z \psi_j \partial_z \psi_i + \frac{m^2}{r^2} \psi_j \psi_i)). \tag{6.26}$$

Comme la construction de la matrice A prend beaucoup de temps, une matrice est sauvegardée pour chaque échelle ℓ.

Une fois le filtre construit, il est alors possible de calculer chacun des termes de transferts d'énergie. Comme il s'agit simplement de successions de produits scalaires et vectoriels ainsi que de dérivées, la seule difficulté provient du fait que les champs soient définis sur des maillages de simulation différents. Cela nécessite de parfois projeter des champs existants sur le maillage du champ magnétique vers le maillage du champ de vitesse. Comme le coût en mémoire vive et en temps de calcul est important, une attention particulière a été prêtée à l'optimisation de cette partie du code. Ainsi, par exemple, les quantités indépendantes de l'échelle du filtre ℓ (comme le courant électrique j) ne sont calculées qu'une seule fois et gardées en mémoire vive pour d'autres échelles. Du côté optimisation mémoire, les tableaux de données utilisés pour stocker des champs scalaires et vectoriels dans le code sont réutilisés plusieurs fois pour éviter plusieurs allocations et désallocations de mémoire. Ceci est fait tout en minimisant le nombre de tableaux utilisés. Ainsi un fichier nommé 'memory_usage.ods' (traduit par utilisation de la mémoire) a été créé permettant de suivre quels termes sont présents dans quels tableaux à chaque instant du code.

En plus du calcul de tous les termes de transferts, d'autres quantités utiles sont calculées comme les différentes hélicités et courants. Finalement, les moyennes spatiales sur l'ensemble du volume de simulation de toutes ces quantités sont sauvegardées pour chaque pas de temps temporel dans un fichier nommé 'fort.75'.

De nombreux tests ont permis de vérifier que les termes sont calculés correctement. Par exemple, pour $\ell = 0$, certains termes de transfert doivent être nuls. Cependant cela n'est pas le cas pour tous à cause du repliement de spectre de Fourier (aussi connu sous le nom d'aliasing) lors des produits

entre champs. Pour éviter cela, SFEMaNS a déjà implémenté les stratégies d'anti-aliasing pour de simples produits à deux termes $f \times g$ mais cela n'est pas suffisant pour des produits à trois termes ou plus $f \times (g \times h)$. Ainsi, il y a une erreur de quelques pourcents sur le calcul des énergies de transfert contenant des produits multiples.

6.3.2 . Transferts des données SFEMaNS vers des formats de fichiers utilisables

Les sorties SFEMaNS se trouvent être dans un format spécifique contenant de nombreuses informations superflues à l'étude des termes mais nécessaires pour relancer les simulations temporelles. De plus, les différents modes de Fourier sont sauvegardés de manière complexe en fonction de la structure du maillage et de sa répartition sur les processeurs. Pour pouvoir utiliser et visualiser ces données de manière efficace, un code simple permettant de changer le format de ces fichiers a été développé. Cela est fait nécessairement dans le code SFEMaNS et a pour nom de projet sur GitLab 'suites_to_bins'.

Ce code prend toutes les itérations temporelles des différents termes déjà calculés et donne en sortie soit des fichiers binaires pour le traitement statistique via Python ou des fichier pvd/vtu pour la visualisation dans Paraview. Encore un fois, toutes les options sont incluses dans un fichier 'data_example' qui ont simplement besoin d'être copiées dans le fichier 'data' de la simulation temporelle. Les différentes options permettent de sélectionner les champs à transférer mais aussi de faire des moyennes temporelles de ceux-ci. Il est aussi possible de les normaliser par la valeur moyenne de l'énergie magnétique au cours du temps pour pouvoir ainsi mieux mettre en évidence le mode propre de la croissance exponentielle du champ magnétique dans la phase de croissance de la dynamo.

6.3.3 . Automatisation de la visualisation avec Paraview

La visualisation est un point important pour l'analyse des phénomènes physiques car elle permet de mettre en évidence où et quand ils arrivent. Cependant, au vu des grandes quantités de données, il est important d'automatiser ce processus. Une fois les champs transférés dans les fichiers pvd/vtu, il est possible de les visualiser grâce à Paraview et d'automatiser le tout via Python. Le projet GitLab se nomme 'paraview_python' et la version Paraview utilisée est 5.8.0. Il peut y avoir quelques problèmes si des versions Paraview différentes sont utilisées car il n'y a pas de garantie que les noms des fonctions Python soient les mêmes.

Comme SFEMaNS est en coordonnées cylindriques, les différentes visualisations implémentées pour des quantités scalaires sont (voir figure 6.10) : des sections planes (θ, r) pour différentes hauteurs z, des sections planes (r, z) pour différents angles θ et des isosurfaces 3D.

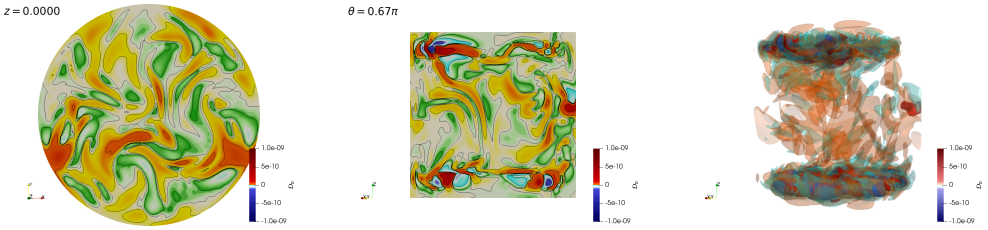


Figure 6.10: Exemple des trois types de visualisation avec échelles et contours bi-logarithmiques.

De nombreuses options permettent de choisir différentes échelles de couleurs et de stipuler les iso-surfaces à représenter. Par défaut, elles sont en échelles bi-logarithmiques car les quantités étudiées varient grandement en intensité à cause des turbines.

Ces visualisations sont codées de manière à éviter de charger en mémoire plusieurs fois un même champ pour des visualisations différentes. Ainsi, pour parcourir tous les champs à visualiser, d'autres fonctions sont utilisées pour créer les tableaux Paraview à donner aux fonctions de visualisation. Cela permet par ailleurs de faire des calculs supplémentaires sur les champs avant de les visualiser grâce aux modules de calculs dans Paraview. Pour rendre les visualisations plus rapides, une parallélisation de celles-ci a été mise en place.

Comme la quantité d'images produites est importante, il est nécessaire d'avoir un outil permettant de les visualiser directement sur un super-calculateur sans les transférer en local. Par exemple, dans notre cas avec environ 20 champs scalaires sur 120 itérations temporelles et trois échelles de filtrage, cela donne des dizaines de milliers d'images (200 Go de mémoire) et ceci seulement pour les isosurfaces 3D. Ainsi un GUI simple a été codé en Python dans le projet GitLab 'image_viewer_sfemans'. Il suffit alors simplement de donner le répertoire contenant les images pour y avoir accès (voir figure 6.11).

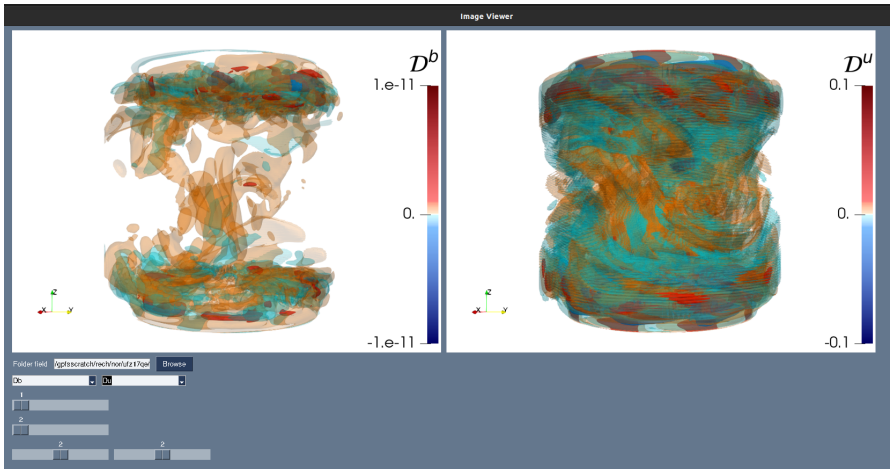


Figure 6.11: Fenêtre de visualisation pour accès à distance des différents champs scalaires.

6.3.4 . Perspectives

Il était essentiel de programmer ces différents programmes permettant d'avoir une stratégie efficace depuis les sorties des simulations numériques jusqu'à la visualisation des transferts d'énergie. De plus, cela a été important pour rapidement identifier et tester toutes modifications faites au filtrage et aux calculs des différentes quantités. Cela a permis par ailleurs de produire rapidement des données pour de nombreux stages.

Par exemple, quelques outils d'analyse statistique en Python qui ont été implémentés en début de thèse ont ainsi pu être restructurés par Nikita Allaglo (stagiaire de L3, printemps 2023) et Rémi Bousquet (stagiaire de M2, printemps 2023). Malheureusement, ils n'ont pas pu être utilisés dans le cadre de cette thèse.

6.4 . Transferts énergétiques dans l'expérience de von Kármán Sodium

Cette section propose un résumé du chapitre 4 qui se présente sous la forme d'un article soumis à Physics of Plasmas (Août 2023). Elle contient les démonstrations analytiques des différents transferts d'énergie entre champs et entre échelles pour les équations de la magnétohydrodynamique avec variations de perméabilité magnétique et de conductivité électrique. Ces transferts sont ensuite appliqués à l'étude de deux simulations numériques ayant généré un champ magnétique mais avec des matériaux différents pour les turbines (acier et fer doux).

6.4.1 . Bilans énergétiques locaux

Pour évaluer les bilans d'énergie locaux permettant de décrire les transferts d'énergie entre champ magnétique et champ de vitesse, nous utilisons une méthode de filtrage. Cette stratégie s'inspire de travaux mathématiques pour dériver des dissipations anormales dues aux singularités dans le contexte des équations de Navier-Stokes et dans le contexte des équations MHD. Notre travail étend cette méthode en introduisant des variations possibles des propriétés des matériaux (μ, σ). Ceci est important pour bien prendre en compte ce qui se passe dans nos simulations ainsi que pour évaluer l'impact des sauts de conductivité électrique et de perméabilité magnétique.

Ainsi, en appliquant la méthode de filtrage (décrite en section 6.24) aux équations de la MHD (6.1) et (6.2), nous obtenons :

$$\partial_t \overline{u}^\ell + \partial_i \overline{u_i u}^\ell - \nu \Delta \overline{u}^\ell + \frac{\nabla \overline{p}^\ell}{\rho} = -\frac{\overline{b \times j}^\ell}{\rho} + \overline{f}^\ell, \tag{6.27}$$

$$\partial_t \overline{b}^\ell = \nabla \times (\overline{u \times b}^\ell) - \nabla \times \overline{(\frac{j}{\sigma})}^\ell, \tag{6.28}$$

$$\nabla \cdot \overline{u}^\ell = 0, \tag{6.29}$$

$$\nabla \cdot \overline{b}^\ell = 0. \tag{6.30}$$

Comme la densité ρ et la viscosité du fluide ν sont constantes dans nos simulations, elles n'impactent pas le filtrage et peuvent donc en être extraites.

En sommant le produit scalaire de $\rho u/2$ avec l'équation (6.27) et le produit scalaire de $\rho \overline{u}^\ell/2$ avec l'équation (6.1) et en faisant des produits similaires pour b, les équations pour l'énergie cinétique filtrée $E^c = \frac{\rho u \overline{u}^\ell}{2}$ et magnétique filtrée $E^m = \frac{b \overline{b}^\ell}{2\mu}$ sont :

$$\partial_t E^c + \nabla \cdot J^{NS} + \mathcal{D}^\nu + \mathcal{D}^u + \mathcal{D}^b = -\mathcal{T}^{u \to b} + \mathcal{P}, \tag{6.31}$$

$$\partial_t E^m + \nabla \cdot J^M + \mathcal{D}^\sigma + \mathcal{D}^{\overline{\sigma}} + \mathcal{D}^b + \mathcal{D}^{\mu\sigma} + \mathcal{D}^{\mu\mathcal{T}} = \mathcal{T}^{u \to b} + \mathcal{M}^\mu, \tag{6.32}$$

avec

$$J^{NS} = \frac{\rho}{2}((u\overline{u}^{\ell})u - \nu\nabla(u\overline{u}^{\ell})) + \frac{1}{2}(\overline{u}^{\ell}p + u\overline{p}^{\ell}), \tag{6.33}$$

$$J^{M} = \frac{1}{2\mu}\{\frac{j}{\sigma} \times \overline{b}^{\ell} + \overline{(\frac{j}{\sigma})}^{\ell} \times b + b \times (\overline{u \times b}^{\ell}) + \overline{b}^{\ell} \times (u \times b)\}, \tag{6.34}$$

$$\mathcal{P} = \frac{\rho}{2}(u \cdot \overline{f}^{\ell} + \overline{u}^{\ell} \cdot f), \tag{6.35}$$

$$\mathcal{D}^{\nu} = \rho\nu\partial_i\overline{u_j}^{\ell}\partial_i u_j, \tag{6.36}$$

$$\mathcal{D}^{\sigma} = \frac{j \cdot \overline{j}^{\ell}}{\sigma}, \qquad \mathcal{D}^{\overline{\sigma}} = \left(\overline{(\frac{j}{\sigma})}^{\ell} - \frac{(\overline{j}^{\ell})}{\sigma}\right) \cdot \frac{j}{2}, \tag{6.37}$$

$$\mathcal{D}^{u} = \frac{\rho}{2}(u\overline{u_i\partial_i u}^{\ell} - uu_i\partial_i\overline{u}^{\ell}), \tag{6.38}$$

$$\mathcal{D}^{b} = \frac{1}{4}(\overline{u}^{\ell} \cdot (b \times j) + u \cdot (\overline{b \times j}^{\ell}) - (\overline{u \times b}^{\ell}) \cdot j - (u \times b) \cdot \overline{j}^{\ell}), \tag{6.39}$$

$$\mathcal{T}^{u\to b} = \frac{1}{4}(\overline{u}^{\ell} \cdot (b \times j) + u \cdot (\overline{b \times j}^{\ell}) + (\overline{u \times b}^{\ell}) \cdot j + (u \times b) \cdot \overline{j}^{\ell}), \tag{6.40}$$

$$\mathcal{M}^{\mu} = -\frac{b \cdot \overline{b}^{\ell}}{2\mu^2}\partial_t\mu, \tag{6.41}$$

$$\mathcal{D}^{\mu\sigma} = \frac{j}{2\sigma} \cdot \nabla \times \left(\frac{(\overline{b}^{\ell})}{\mu} - \overline{H}^{\ell}\right), \qquad \mathcal{D}^{\mu\mathcal{T}} = -\frac{(u \times b)}{2} \cdot \nabla \times \left(\frac{(\overline{b}^{\ell})}{\mu} - \overline{H}^{\ell}\right). \tag{6.42}$$

Tous les termes mesurent les transferts d'énergie pour toutes les échelles plus grandes que ℓ.

Pour commencer, nous discutons les termes de l'équation (6.31). Du côté gauche de l'équation, J_{NS} représente le courant d'énergie responsable du mouvement spatial de l'énergie et ainsi ne participe ni aux transferts d'énergie entre échelles et champs ni aux dissipations. $\mathcal{D}^{\nu}$ est la dissipation due à la viscosité du fluide : c'est globalement une perte d'énergie pour le champ de vitesse. $\mathcal{D}^{u}$ est la dissipation anormale dite de Duchon-Robert. Ce terme correspond au transfert local d'énergie vers les échelles plus petites que ℓ et ainsi permet de caractériser où se forment les irrégularités du champ de vitesse. Elle est appelée dissipation anormale car, dans la limite $\ell \to 0$, elle peut ne pas disparaître [23]. De même, $\mathcal{D}^{b}$ est un transfert d'énergie entre échelles mais cette fois dû aux interactions entre le champ de vitesse et le champ magnétique. Comme dans le cas cinétique, ce terme peut produire une contribution anormale si les champs sont suffisamment irréguliers. Si la contribution est positive, il s'agit alors d'une dissipation anormale. Cependant, il est aussi possible que sa contribution soit négative résultant alors en une amplification du champ magnétique. Cela correspondrait à une situation pour laquelle l'effet dynamo est produit par un champ de vitesse irrégulier [7], un mécanisme aussi connu sous le nom de dynamo rapide [8].

Du côté droit, le terme $(-\mathcal{T}^{u\to b})$ correspond à la puissance de la force de Lorentz et reflète le transfert direct d'énergie entre les champs u et b. Il apparaît naturellement avec un signe opposé du côté droit du bilan d'énergie magnétique (6.32). Il est ainsi le candidat principal pour l'apparition et le maintien du champ magnétique et est donc un terme source pour l'effet dynamo. Le dernier terme, $\mathcal{P}$, est simplement l'injection d'énergie mécanique par le forçage (dans notre cas les turbines).

En plus des termes $\mathcal{T}^{u\to b}$ et $\mathcal{D}^b$, de nouveaux termes intéressants apparaissent dans l'équation (6.32). Du côté droit, J_M est simplement le courant d'énergie décrivant comment l'énergie magnétique est transportée par le fluide. $\mathcal{D}^\sigma$ est la dissipation Joule et donc une perte d'énergie pour le champ magnétique. $\mathcal{D}^{\overline{\sigma}}$ est la contribution aux transferts d'énergie entre échelles due aux sauts de conductivité électrique. De même, $\mathcal{D}^{\mu\sigma}$ et $\mathcal{D}^{\mu\mathcal{T}}$ sont des termes de transferts d'énergie entre échelles dus aux irrégularités de la perméabilité magnétique. Finalement, du côté gauche, un autre terme source pour l'effet dynamo apparaît, $\mathcal{M}^\mu$. Il est produit par des variations temporelles de perméabilité magnétique et est un transfert direct des turbines vers le champ magnétique.

6.4.2 . Simulations numériques

Nous étudions alors ces transferts d'énergie pour la géométrie de von Kármán Sodium représentée en figure 6.1. Nos simulations ont été faites en utilisant le code SFEMaNS présenté en section 6.2 en utilisant le supercalculateur Jean-Zay de l'IDRIS. Les simulations utilisées correspondent à deux séries décrites dans [58] (Table 3). Nous utilisons des unités adimensionnées de telle sorte que la longueur caractéristique du système L corresponde au rayon du cylindre interne R_{cyl}. En dénotant σ_0 la conductivité électrique du sodium liquide, ρ sa densité et μ_0 la perméabilité magnétique du vide, le champ magnétique est adimensionné en utilisant l'échelle de Alfvén $B = U\sqrt{\rho\mu_0}$, avec $U = \omega R_{\text{cyl}}$ où ω est la vitesse angulaire des turbines. Les deux paramètres de contrôle sont alors $R_m = \mu_0\sigma_0 R_{\text{cyl}}^2\omega$, le nombre de Reynolds magnétique et $R_e = R_{\text{cyl}}^2\omega/\nu$, le nombre de Reynolds cinétique. Nous définissons aussi la conductivité électrique relative σ_r et la perméabilité magnétique relative μ_r. Ces quantités peuvent ne pas être uniformes pour des murs et turbines de matériaux différents. Dans notre cas, $\sigma_r = 1, \mu_r = 1$ dans la région $\{(r,\theta,z) \in [1,1.4]\times[0,2\pi)\times[-1,1]\}$ pour modéliser le sodium , et $\sigma_r = 4.5, \mu_r = 1$ in $\{(r,\theta,z) \in [1.4,1.6]\times[0,2\pi)\times[-1,1]\}$ pour représenter les murs en cuivre.

Les deux simulations que nous utilisons sont réalisées à un nombre de Reynolds modéré $R_e = 1500$ mais avec différentes perméabilités magnétiques pour les turbines : $\mu_r = 1$ pour celles en acier et $\mu_r = 50$ pour celles en fer doux. Pour des conditions aux limites de pseudo-vide, les nombres de

Reynolds magnétique pour avoir une dynamo sont $R_m^c(\mu_r = 1) = 190 \pm 10$ et $R_m^c(\mu_r = 50) = 90 \pm 5$. Ainsi, nous utilisons des R_m au-dessus de ces seuils, respectivement $R_m(\mu_r = 1) = 300$ et $R_m(\mu_r = 50) = 150$, pour pouvoir observer les deux phases de croissance et saturation de l'effet dynamo. Chaque régime est étudié pour 128 pas de temps (1 tous les huitièmes de tour pour stroboscoper le mouvement des turbines). Nous utiliserons le symbole $\{.\}$ pour la moyenne faite sur tous ces pas de temps. La moyenne sur tout le volume aura le symbole $langle.\rangle$. L'échelle de Kolmogorov pour ces simulations est $\eta = 1.16 \times 10^{-2}$.

Nous allons maintenant succinctement résumer les principaux résultats observés pour les deux phases (en croissance et en saturation). Pour cela, nous nous concentrerons seulement sur le bilan d'énergie pour le champ magnétique et ne discuterons que l'effet dynamo et non les diverses contributions anormales.

6.4.3 . Mécanismes dynamo

En phase de croissance, nous obtenons les bilans énergétiques pour le champ magnétique montrés en figure 6.12. Pour mieux souligner l'importance respective des différents termes, ils sont renormalisés par l'énergie magnétique moyenne $\langle E^m \rangle(t)$. Cela permet d'éviter de seulement voir la croissance exponentielle de tous ces termes due à la croissance exponentielle du champ magnétique.

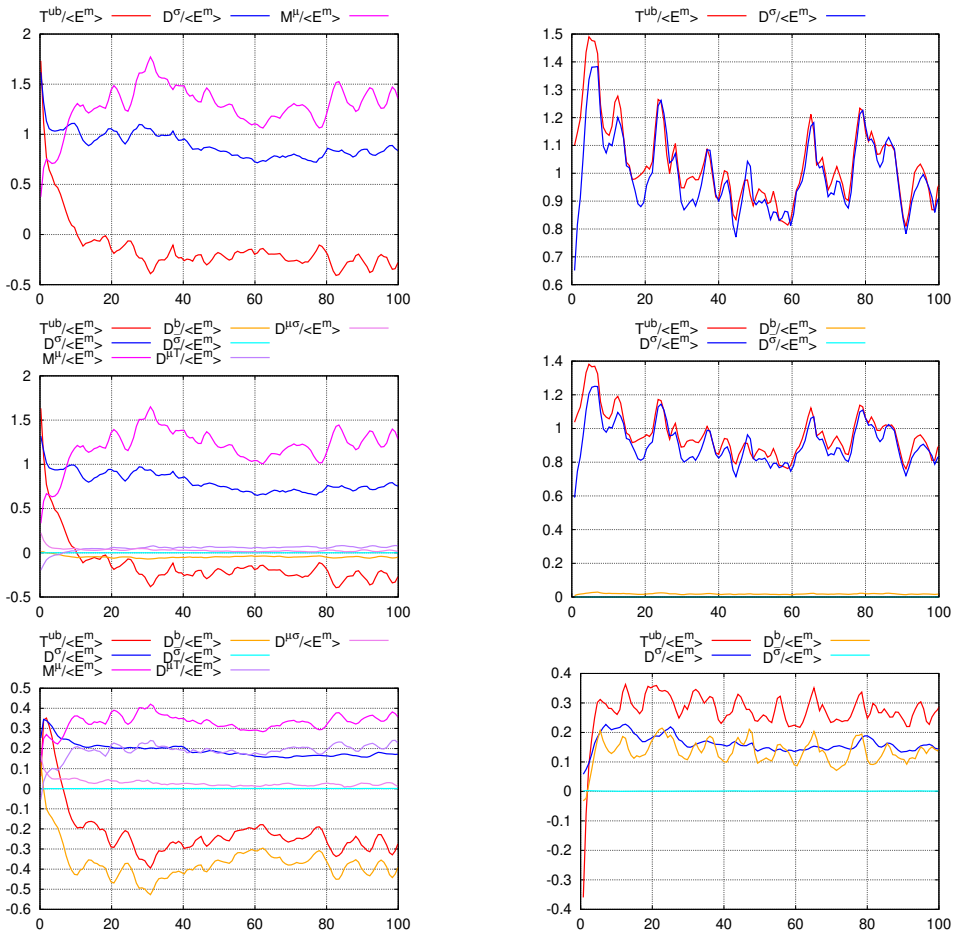


Figure 6.12: Évolution temporelle du bilan d'énergie magnétique (6.32) sur l'ensemble du volume renormalisée en phase de croissance. Les panneaux correspondent à $\frac{\ell}{\eta} = 0$ pour la première ligne, $\frac{\ell}{\eta} = 1$ pour la deuxième ligne $\frac{\ell}{\eta} = 16$ pour la troisième ligne, et pour le cas $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ dans la première colonne et pour le cas $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ dans la deuxième colonne.

Nous observons alors deux régimes dynamo très différents. Pour $\mu_r = 1$ et $\ell = 0$, le transfert d'énergie venant du champ de vitesse $\mathcal{T}^{u\to b}$ surpasse les pertes par dissipation Joule $\mathcal{D}^{\sigma}$. La situation est similaire pour $\ell = \eta$ car les termes associés aux irrégularités $\mathcal{D}^{b}$ et $\mathcal{D}^{\overline{\sigma}}$ sont presque nuls. Un ℓ plus grand augmente considérablement l'impact des transferts d'énergie entre échelles $\mathcal{D}^{b}$ (positif donc transférant de l'énergie vers les petites échelles) et diminue $\mathcal{T}^{u\to b}$. La participation des sauts de conductivité $\mathcal{D}^{\overline{\sigma}}$ reste cependant négligeable. Ainsi, pour ce nombre de Reynolds modéré, c'est le terme $\mathcal{T}^{u\to b}$ qui compense toutes les pertes dissipatives. En observant localement ce transfert d'énergie (figure 6.13, colonne de droite), on remarque qu'il prend place à la fois au niveau des turbines et dans le centre du volume et ce pour toutes

les échelles. Cette dernière contribution permet à $\mathcal{T}^{u\to b}$ d'être globalement positif, ce qui en accord avec les mécanismes standards de la dynamo liés à la présence de rotation différentielle et de vortex dans la couche de cisaillement dans le plan équatorial et au niveau des turbines.

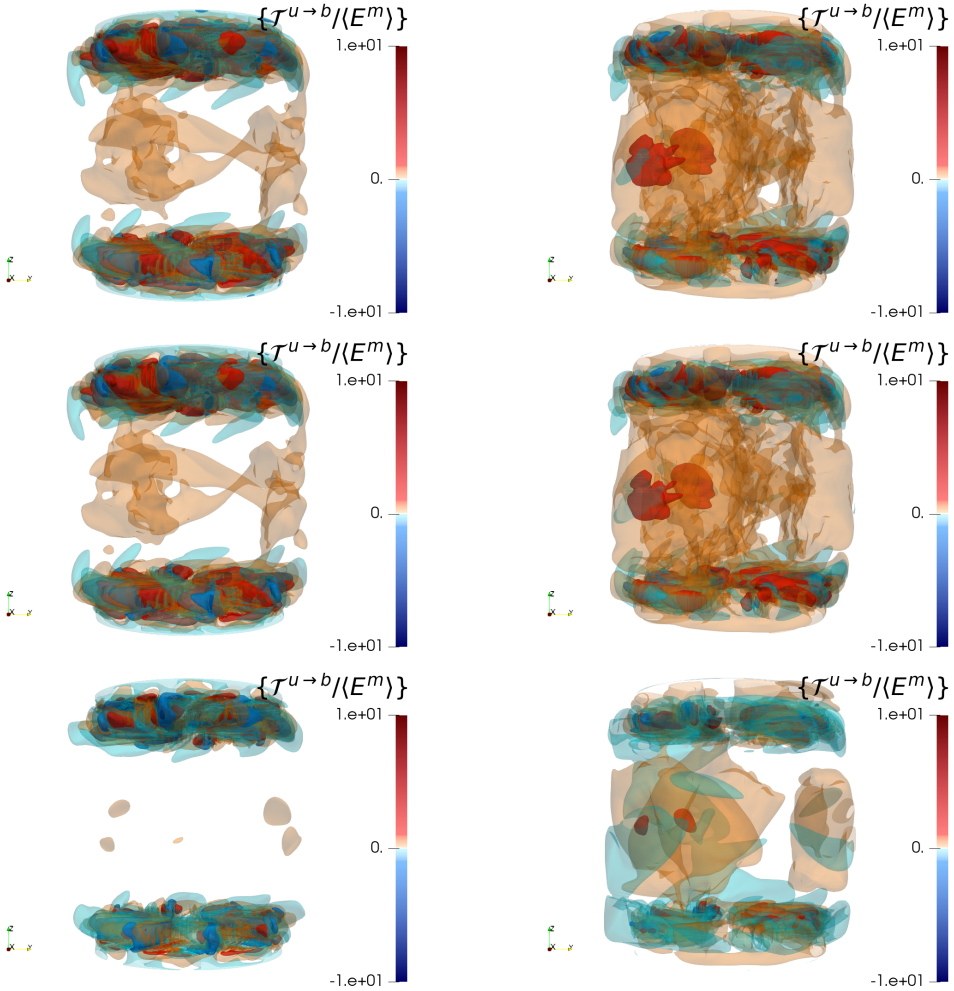


Figure 6.13: Transferts d'énergie entre les champs de vitesse et magnétique renormalisés et moyennés en temps en phase de croissance. Les panneaux correspondent à $\frac{\ell}{\eta} = 0$ pour la première ligne, $\frac{\ell}{\eta} = 1$ pour la deuxième ligne $\frac{\ell}{\eta} = 16$ pour la troisième ligne, et pour le cas $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ dans la première colonne et pour le cas $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ dans la deuxième colonne.

Pour $\mu_r = 50$, les premiers instants ($t < 3$) se comportent de la même manière que pour $\mu_r = 1$ (figure 6.12, colonne de gauche). Cela indique que $\mathcal{T}^{u\to b}$ est crucial pour démarrer les mécanismes dynamo. Après ce démarrage, le comportement dynamo devient totalement différent. L'injection

d'énergie par les turbines $\mathcal{M}^{\mu}$ prend place et devient la source de génération et d'amplification du champ magnétique pendant que $\mathcal{T}^{u\to b}$ devient négatif et donc la puissance de la force de Lorentz devient anti-dynamo. En volume, nous observons que $\mathcal{T}^{u\to b}$ (figure 6.13, colonne de gauche) reste toujours positif pour des échelles suffisamment petites dans la couche de mélange. L'effet anti-dynamo de $\mathcal{T}^{u\to b}$ provient alors de ce qui se passe au niveau des turbines où sa contribution moyenne est négative. La génération dynamo provient alors de $\mathcal{M}^{\mu}$ qui prend place essentiellement au niveau des surfaces des pales (voir figure 6.14). Augmenter l'échelle de filtrage ℓ rend les termes liés aux irrégularités plus intenses : $\mathcal{D}^{\mu\mathcal{T}}$ qui est lié aux sauts de perméabilité magnétique et au champ électromoteur devient positif ; $\mathcal{D}^{b}$ suit les variations de $\mathcal{T}^{u\to b}$ et devient négatif provoquant alors une cascade inverse d'énergie entre les échelles. Les deux autres termes $\mathcal{D}^{\mu\sigma}$ et $\mathcal{D}^{\overline{\sigma}}$, tous deux liés aux sauts de conductivité, sont négligeables.

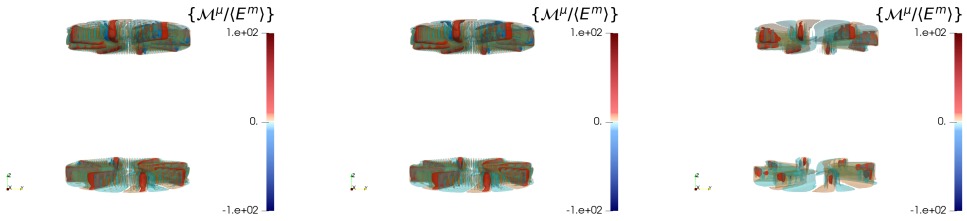


Figure 6.14: Injections d'énergie par les turbines dans le champ magnétique, renormalisées et moyennées en temps en phase de croissance pour le cas $R_e = 1500, R_m = 150, \mu_r = 50$ aux échelles: (gauche) $\frac{\ell}{\eta} = 0$, (centre) $\frac{\ell}{\eta} = 1$, (droit) $\frac{\ell}{\eta} = 16$.

Pour les bilans d'énergie magnétique représentés en figure 6.15, on observe deux régimes dynamo très différents en fonction de μ_r. On observe un comportement globalement similaire à la phase de croissance pour les deux cas. Cependant, on a bien $\partial_t E^m$ autour de zéro ; il s'agit donc bien de la phase saturée.

On remarque cependant un évènement intéressant dans l'intervalle de temps $[970 : 990]$ où $\mathcal{T}^{u\to b} - \mathcal{D}^{b} - \mathcal{D}^{\sigma}$ est négatif correspondant à une décroissance momentanée du champ magnétique.

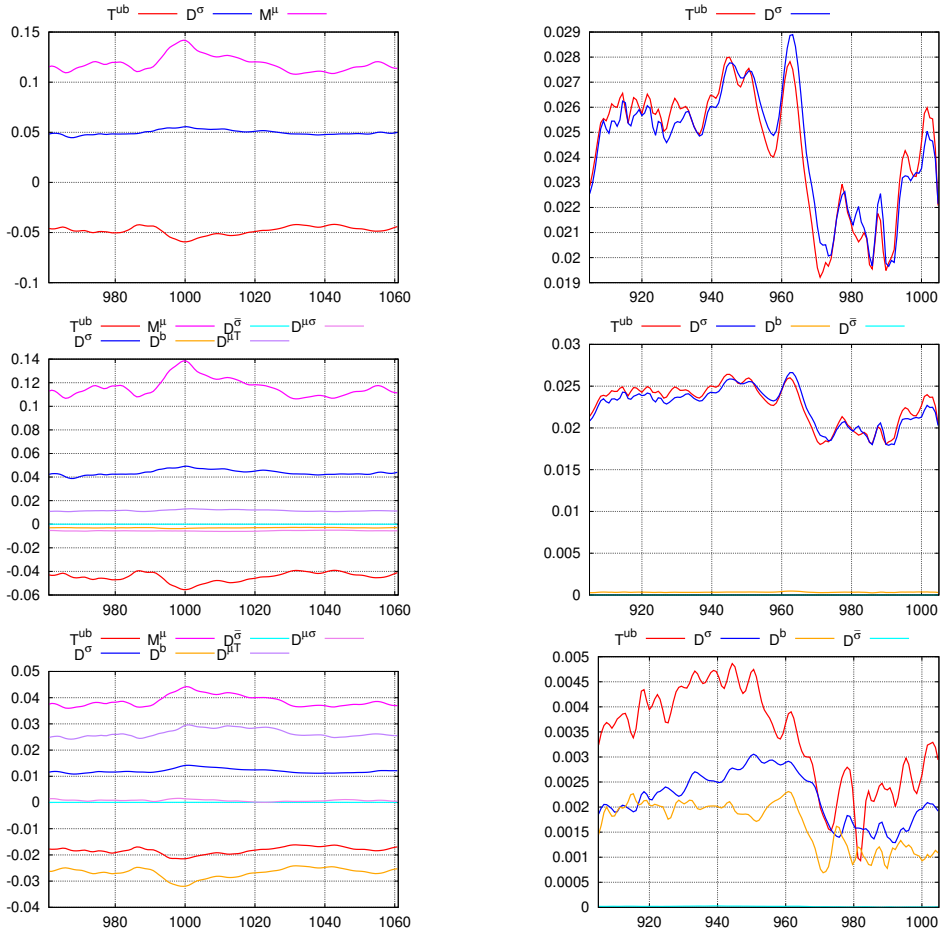


Figure 6.15: Évolution temporelle du bilan d'énergie magnétique (6.32) sur l'ensemble du volume en phase saturée. Les panneaux correspondent à $\frac{\ell}{\eta} = 0$ pour la première ligne, $\frac{\ell}{\eta} = 1$ pour la deuxième ligne $\frac{\ell}{\eta} = 16$ pour la troisième ligne, et pour le cas $R_e = 1500$, $R_m = 150$, $\mu_r = 50$ dans la première colonne et pour le cas $R_e = 1500$, $R_m = 300$, $\mu_r = 1$ dans la deuxième colonne.

Pour résumer, la puissance de la force de Lorentz $\mathcal{T}^{u\to b}$ est nécessaire au démarrage de la dynamo dans les deux cas, soulignant le rôle crucial du champ de vitesse pour initialiser les mécanismes dynamo. Cependant, le développement de la dynamo et son maintien diffèrent fortement pour les deux cas. Pour les turbines en fer doux, la rotation des turbines via $\mathcal{M}^{\mu}$ devient le seul acteur dynamo pendant que l'action de la force de Lorentz devient anti-dynamo. Pour les turbines en acier, le phénomène dynamo provient uniquement du champ de vitesse grâce à plusieurs mécanismes se passant au niveau des turbines et de la couche de mélange.

6.5 . Conclusion

Dans cette thèse, l'objectif principal a été de comprendre l'effet dynamo dans les fluides conducteurs et turbulents. Pour cela, la configuration de von Kármán Sodium a été simulée en utilisant le code SFEMaNS. Le choix de cette configuration est motivé par la possibilité de comparer les résultats numériques à des expériences faites au CEA de Cadarache. Pour mieux comprendre les mécanismes dynamo, nous avons décidé d'étudier les transferts d'énergie entre les champs de vitesse et magnétique ainsi qu'entre les différentes échelles de l'écoulement. Cette approche est une extension directe à la magnétohydrodynamique de ce qui a été fait pour les équations de Navier-Stokes dans le cadre de la thèse de H.Faller [60]. Ces choix ont mené à l'implémentation de nombreux outils de post-traitement ainsi qu'à quelques modifications dans le code SFEMaNS.

Une grande partie des améliorations que nous avons apportées à SFEMaNS vient de la nouvelle distribution du maillage de simulation sur les processeurs. Cela a permis d'économiser de la place mémoire et d'atteindre 300.000 points de simulations par processeur pour des simulations 2D de Navier-Stokes. Ainsi, il est maintenant possible de lancer des simulations numériques pour de plus grands nombres de Reynolds cinétique à la fois pour l'hydrodynamique et la MHD. En plus de cela, nous avons testé une méthode de préconditionnement pour résoudre le problème de Stokes dans SFEMaNS pour une éventuelle utilisation dans des méthodes de simulations temporelles implicites et/ou explicites. Les différentes techniques de préconditionnement testées obtiennent une performance similaire d'environ 10^{-3} secondes de temps de calcul dépensé par degré de liberté avec une très bonne efficacité de la parallélisation. Cependant, elles ne sont pas suffisamment performantes comparées à la méthode de prédiction-correction utilisée actuellement dans SFEMaNS (ayant un temps de calcul par degré de liberté d'environ 10^{-5} secondes).

De plus, pour calculer et analyser de manière efficace les différents transferts d'énergie, nous avons mis en place de nombreux outils de post-traitement. Une attention particulière a été portée pour économiser la mémoire et le temps de calcul durant l'évaluation de ces termes. Un outil général permettant de transférer les sorties de SFEMaNS en des fichiers simples d'utilisation pour un traitement statistique ou de la visualisation a été développé. Une automatisation de Paraview pour la visualisation des structures des champs avec de nombreuses fonctionnalités a été développée. Tous ces outils ont été nécessaires pour rendre l'analyse et la visualisation des transferts d'énergie efficaces au vu de la quantité de données à analyser. Ces outils sont par ailleurs indépendants les uns des autres et peuvent être donc utilisés pour toute simulation future utilisant SFEMaNS.

Enfin, la dérivation des bilans d'énergie pour la magnétohydrodynamique

a amené à la découverte de nombreux termes de transferts d'énergie. En particulier, l'influence des variations temporelles et spatiales de la perméabilité magnétique sur les transferts d'énergie entre échelles est maintenant correctement décrite. Ainsi, dans nos deux cas d'étude, nous observons des mécanismes dynamo très dépendants de la perméabilité magnétique des turbines. Dans le cas des turbines en acier, la puissance de la force de Lorentz permet de compenser tous les termes de dissipation alors que, pour des turbines en fer doux, elle devient anti-dynamo. Pour des turbines en fer, dans le régime saturé, la seule contribution à l'effet dynamo provient alors de la rotation des turbines ayant une grande perméabilité magnétique.

Dans le futur, une analyse plus complète des différents transferts d'énergie serait utile. En particulier, des analyses à plus grands nombres de Reynolds cinétique des transferts d'énergie responsables des dissipations anormales pourraient mener à des indices sur l'apparition de quasi-singularités et de phénomènes de dynamo rapide dans la MHD. De plus, des simulations pour des perméabilités magnétiques intermédiaires devraient apporter une meilleure compréhension de la bascule se passant entre la puissance de la force de Lorentz et l'effet de la rotation de la perméabilité magnétique. Dans toutes ces simulations, il serait par ailleurs intéressant de suivre les évènements rares et extrêmes et les autres structures cohérentes grâce aux outils de visualisation. Finalement, un traitement statistique de ces données serait nécessaire pour mieux souligner les corrélations entre les différents phénomènes et ainsi construire de nouveaux modèles permettant de mieux comprendre l'effet dynamo.

Bibliography

[1] A. Gailitis, O. Lielausis, E. Platacis, S. Dement'ev, A. Cifersons, G. Gerbeth, T. Gundrum, F. Stefani, M. Christen, and G. Will, "Magnetic field saturation in the Riga dynamo experiment," *Phys. Rev. Lett.*, vol. 86, pp. 3024–3027, Apr 2001.

[2] R. Stieglitz and U. Müller, "Experimental demonstration of a homogeneous two-scale dynamo," *Physics of Fluids*, vol. 13, no. 3, pp. 561–564, 2001.

[3] S. Miralles, N. Bonnefoy, M. Bourgoin, P. Odier, J.-F. m. c. Pinton, N. Plihon, G. Verhille, J. Boisson, F. m. c. Daviaud, and B. Dubrulle, "Dynamo threshold detection in the von kármán sodium experiment," *Phys. Rev. E*, vol. 88, p. 013002, Jul 2013.

[4] R. Monchaux, M. Berhanu, M. Bourgoin, M. Moulin, P. Odier, J.-F. Pinton, R. Volk, S. Fauve, N. Mordant, F. Pétrélis, A. Chiffaudel, F. Daviaud, B. Dubrulle, C. Gasquet, L. Marié, and F. Ravelet, "Generation of a magnetic field by dynamo action in a turbulent flow of liquid sodium," *Phys. Rev. Lett.*, vol. 98, p. 044502, Jan 2007.

[5] J.-P. Laval, P. Blaineau, N. Leprovost, B. Dubrulle, and F. Daviaud, "Influence of turbulence on the dynamo threshold," *Phys. Rev. Lett.*, vol. 96, p. 204503, May 2006.

[6] A. Alexakis, S. Fauve, C. Gissinger, and F. Pétrélis, "Effect of fluctuations on mean-field dynamos," *Journal of Plasma Physics*, vol. 84, no. 4, p. 735840401, 2018.

[7] G. L. Eyink, "Stochastic flux freezing and magnetic dynamo," *Phys. Rev. E*, vol. 83, p. 056405, May 2011.

[8] S. Childress, *Topological Aspects of the Dynamics of Fluids and Plasmas*, ch. Fast dynamo theory. Ed. Editors, H. K. Moffatt, G. M. Zaslavsky, P. Comte, M. Tabor, Springer Dordrecht, 1992.

[9] P. H. Roberts and A. M. Soward, "Dynamo theory," *Annual Review of Fluid Mechanics*, vol. 24, no. 1, pp. 459–512, 1992.

[10] N. Leprovost, *Influence des petites échelles sur la dynamique à grande échelle en turbulence hydro et magnétohydrodynamique*. Theses, Université Pierre et Marie Curie - Paris VI, Nov. 2004.

[11] H. Moffatt, *Magnetic Field Generation in Electrically Conducting Fluids*. Cambridge University Press, 1978.

[12] F. Krause and K.-H. Rädler, *Mean-field magnetohydrodynamics and dynamo theory*. Oxford, Pergamon Press, 1980.

[13] F. Pétrélis, N. Mordant, and S. Fauve, "On the magnetic fields generated by experimental dynamos," *Geophysical & Astrophysical Fluid Dynamics*, vol. 101, no. 3-4, pp. 289–323, 2007.

[14] F. Pétrélis and S. Fauve, "Mechanisms for magnetic field reversals," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1916, pp. 1595–1605, 2010.

[15] F. H. Busse and J. Wicht, "A simple dynamo caused by conductivity variations," *Geophysical & Astrophysical Fluid Dynamics*, vol. 64, no. 1-4, pp. 135–144, 1992.

[16] T. M. Rogers and J. N. McElwaine, "The hottest hot jupiters may host atmospheric dynamos," *The Astrophysical Journal Letters*, vol. 841, p. L26, may 2017.

[17] F. Marcotte, B. Gallet, F. c. Pétrélis, and C. Gissinger, "Enhanced dynamo growth in nonhomogeneous conducting fluids," *Phys. Rev. E*, vol. 104, p. 015110, Jul 2021.

[18] A. Giesecke, F. Stefani, and G. Gerbeth, "Role of soft-iron impellers on the mode selection in the von kármán–sodium dynamo experiment," *Phys. Rev. Lett.*, vol. 104, p. 044503, Jan 2010.

[19] B. Gallet, F. Pétrélis, and S. Fauve, "Spatial variations of magnetic permeability as a source of dynamo action," *Journal of Fluid Mechanics*, vol. 727, p. 161–190, 2013.

[20] F. Plunian and T. Alboussière, "Axisymmetric dynamo action produced by differential rotation, with anisotropic electrical conductivity and anisotropic magnetic permeability," *Journal of Plasma Physics*, vol. 87, no. 1, p. 905870110, 2021.

[21] J. Herault and F. Pétrélis, "Optimum reduction of the dynamo threshold by a ferromagnetic layer located in the flow," *Phys. Rev. E*, vol. 90, p. 033015, Sep 2014.

[22] J. Varela, S. Brun, B. Dubrulle, and C. Nore, "Role of boundary conditions in helicoidal flow collimation: Consequences for the von kármán sodium dynamo experiment," *Phys. Rev. E*, vol. 92, p. 063015, Dec 2015.

[23] J. Varela, S. Brun, B. Dubrulle, and C. Nore, "Effects of turbulence, resistivity and boundary conditions on helicoidal flow collimation: Consequences for the Von-Kármán-Sodium dynamo experiment," *Physics of Plasmas*, vol. 24, 05 2017. 053518.

[24] J. Duchon and R. Robert, "Inertial energy dissipation for weak solutions of incompressible Euler and Navier-Stokes equations," *Nonlinearity*, vol. 13, pp. 249–255, dec 1999.

[25] G. L. Eyink, "Cascades and dissipative anomalies in nearly collisionless plasma turbulence," *Phys. Rev. X*, vol. 8, p. 041020, Nov 2018.

[26] S. Galtier, "On the origin of the energy dissipation anomaly in (Hall) magnetohydrodynamics," *Journal of Physics A: Mathematical and Theoretical*, vol. 51, p. 205501, apr 2018.

[27] V. David, S. Galtier, F. Sahraoui, and L. Z. Hadid, "Energy transfer, discontinuities, and heating in the inner heliosphere measured with a weak and local formulation of the Politano–Pouquet law," *The Astrophysical Journal*, vol. 927, p. 200, mar 2022.

[28] Y. Ponomarenko, "Theory of the hydromagnetic generator," *J Appl Mech Tech Phys*, vol. 14, pp. 775–778, 1973.

[29] G. Roberts, "Spatially periodic dynamos," *Philosophical Transactions of the Royal Society of London A*, vol. 266, no. 1179, pp. 553–558, 1970.

[30] T. G. Cowling, "The magnetic field of sunspots," *Monthly Notices of the Royal Astronomical Society*, vol. 94, pp. 39–48, 1933.

[31] T. G. COWLING, "THE DYNAMO MAINTENANCE OF STEADY MAGNETIC FIELDS," *The Quarterly Journal of Mechanics and Applied Mathematics*, vol. 10, pp. 129–136, 01 1957.

[32] W. Dobler, A. Shukurov, and A. Brandenburg, "Nonlinear states of the screw dynamo.," *Physical review. E, Statistical, nonlinear, and soft matter physics*, vol. 65 3 Pt 2B, p. 036311, 2001.

[33] Y. Ponty and F. Plunian, "Transition from large-scale to small-scale dynamo.," *Physical review letters*, vol. 106 15, p. 154502, 2011.

[34] S. Fromang, J. C. B. Papaloizou, G. R. J. Lesur, and T. Heinemann, "Numerical simulations of mhd turbulence in accretion disks," 2009.

[35] F. Cattaneo and D. W. Hughes, "Dynamo action in a rotating convective layer," *Journal of Fluid Mechanics*, vol. 553, pp. 401 – 418, 2006.

[36] J. Léorat, "Numerical study of the dynamo effect in a cylinder and the role of small scales," 1995.

[37] L. Marié, J. Burguete, F. Daviaud, and J. Léorat, "Numerical study of homogeneous dynamo based on experimental von kármán type flows," *The European Physical Journal B - Condensed Matter and Complex Systems*, vol. 33, pp. 469–485, 2003.

[38] A. P. Willis and C. F. Barenghi, "Hydromagnetic taylor–couette flow: numerical formulation and comparison with experiment," *Journal of Fluid Mechanics*, vol. 463, pp. 361 – 375, 2002.

[39] A. Tilgner and F. H. Busse, "Finite-amplitude convection in rotating spherical fluid shells," *Journal of Fluid Mechanics*, vol. 332, pp. 359 – 376, 1997.

[40] F. H. Busse and R. D. Simitev, "Quasi-geostrophic approximation of anelastic convection," *Journal of Fluid Mechanics*, vol. 751, pp. 216 – 227, 2014.

[41] A. Jackson, A. Sheyko, P. Marti, A. Tilgner, D. Cébron, S. Vantieghem, R. D. Simitev, F. H. Busse, X.-H. Zhan, G. Schubert, S. ichi Takehiro, Y. Sasaki, Y.-Y. Hayashi, A. Ribeiro, C. Nore, and J.-L. Guermond, "A spherical shell numerical dynamo benchmark with pseudo-vacuum magnetic boundary conditions," *Geophysical Journal International*, vol. 196, pp. 712–723, 2014.

[42] N. Schaeffer, "Efficient spherical harmonic transforms aimed at pseudospectral numerical simulations," *Geochemistry*, vol. 14, pp. 751 – 758, 2012.

[43] S. Vantieghem, A. Sheyko, and A. Jackson, "Applications of a finite-volume algorithm for incompressible mhd problems," *Geophysical Journal International*, vol. 204, pp. 1376–1395, 2016.

[44] K. H. Chan, K. Zhang, L. Li, and X. Liao, "A new generation of convection-driven spherical dynamos using ebe finite element method," *Physics of the Earth and Planetary Interiors*, vol. 163, pp. 251–265, 2007.

[45] D. Cébron and R. Hollerbach, "Tidally driven dynamos in a rotating sphere," *The Astrophysical Journal Letters*, vol. 789, 2014.

[46] C.-C. Wu and P. H. Roberts, "On a dynamo driven by topographic precession," *Geophysical & Astrophysical Fluid Dynamics*, vol. 103, pp. 467 – 501, 2009.

[47] J. A. Morales, W. J. T. Bos, K. Schneider, and D. C. Montgomery, "Magnetohydrodynamically generated velocities in confined plasma," *Physics of Plasmas*, vol. 22, p. 042515, 04 2015.

[48] e. a. A. Giesecke, "Generation of axisymmetric modes in cylindrical kinematic mean-field dynamos of vks type," *Geophysical & Astrophysical Fluid Dynamics*, vol. 104, no. 2-3, pp. 249–271, 2010.

[49] A. B. Iskakov, S. Descombes, and E. Dormy, "An integro-differential formulation for magnetic induction in bounded domains: boundary element-finite volume method," *Journal of Computational Physics*, vol. 197, pp. 540–554, 2004.

[50] Y. Ponty, P. D. Mininni, J.-F. Pinton, H. Politano, and A. Pouquet, "Dynamo action at low magnetic prandtl numbers: mean flow versus fully turbulent motions," *New Journal of Physics*, vol. 9, p. 296, aug 2007.

[51] B. Dubrulle, P. Blaineau, O. M. Lopes, F. Daviaud, J.-P. Laval, and R. Dolganov, "Bifurcations and dynamo action in a taylor–green flow," *New Journal of Physics*, vol. 9, p. 308, aug 2007.

[52] Y. Ponty, J.-P. Laval, B. Dubrulle, F. Daviaud, and J.-F. Pinton, "Subcritical dynamo bifurcation in the taylor-green flow.," *Physical review letters*, vol. 99 22, p. 224501, 2007.

[53] G. Krstulovic, G. Thorner, J.-P. Vest, S. Fauve, and M. E. Brachet, "Axial dipolar dynamo action in the taylor-green vortex.," *Physical review. E, Statistical, nonlinear, and soft matter physics*, vol. 84 6 Pt 2, p. 066318, 2011.

[54] F. Ravelet, "Bifurcations globales hydrodynamiques et magnetohydrodynamiques dans un ecoulement de von karman turbulent," 2005.

[55] F. Stefani, M. Xu, G. Gerbeth, F. Ravelet, A. Chiffaudel, F. Daviaud, and J. Léorat, "Ambivalent effects of added layers on steady kinematic dynamos in cylindrical geometry: application to the vks experiment," *European Journal of Mechanics - B/Fluids*, vol. 25, no. 6, pp. 894–908, 2006.

[56] R. Laguerre, C. Nore, J. Léorat, and J.-L. Guermond, "Effects of conductivity jumps in the envelope of a kinematic dynamo flow," *Comptes Rendus. Mécanique*, vol. 334, no. 10, pp. 593–598, 2006.

[57] R. Laguerre, C. Nore, A. Ribeiro, J. Léorat, J.-L. Guermond, and F. Plunian, "Impact of impellers on the axisymmetric magnetic mode in the VKS2 dynamo experiment," *Phys. Rev. Lett.*, vol. 101, no. 10, p. 104501, 2008.

[58] A. Giesecke, C. Nore, F. Stefani, G. Gerbeth, J. Léorat, F. Luddens, and J.-L. Guermond, "Electromagnetic induction in non-uniform domains," *Geophys. Astrophys. Fluid Dyn.*, vol. 104, no. 5, pp. 505–529, 2010.

[59] C. Nore, D. Castanon, L. Cappanera, and J.-L. Guermond, "Numerical simulation of the von Kármán sodium dynamo experiment," *Journal of Fluid Mechanics*, vol. 854, pp. 164–195, 11 2018.

[60] J.-L. Guermond *et al.*, "SFEMaNS documentation." https://www.math.tamu.edu/~guermond/SFEMaNS/html/index.html.

[61] H. Faller, *Dissipation, cascades et singularités en turbulence*. PhD thesis, ED-PIF, 2018-2021.

[62] L. Cappanera, P. Debue, H. Faller, D. Kuzzay, E.-W. Saw, C. Nore, J.-L. Guermond, F. Daviaud, C. Wiertel-Gasquet, and B. Dubrulle, "Turbulence in realistic geometries with moving boundaries: when simulations meet experiments," *Comput. Fluids*, vol. 241, p. 104750, 2021.

[63] C. V. Loan, *Computational Frameworks for the Fast Fourier Transform*. SIAM, 1992.

[64] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, 1998.

[65] M. Fortin and R. Glowinski, *Augmented Lagrangian methods*, vol. 15 of *Studies in Mathematics and its Applications*. North-Holland Publishing Co., Amsterdam, 1983. Applications to the numerical solution of boundary value problems, Translated from the French by B. Hunt and D. C. Spicer.

[66] J. Cahouet and J.-P. Chabard, "Some fast 3D finite element solvers for the generalized Stokes problem," *Internat. J. Numer. Methods Fluids*, vol. 8, no. 8, pp. 869–895, 1988.

[67] M. Benzi and M. A. Olshanskii, "An augmented Lagrangian-based approach to the Oseen problem," *SIAM J. Sci. Comput.*, vol. 28, no. 6, pp. 2095–2113, 2006.

[68] M. Benzi, M. A. Olshanskii, and Z. Wang, "Modified augmented lagrangian preconditioners for the incompressible navier–stokes equations," *International Journal for Numerical Methods in Fluids*, vol. 66, no. 4, pp. 486–508, 2011.

[69] M. Benzi and Z. Wang, "Analysis of augmented lagrangian-based preconditioners for the steady incompressible navier–stokes equations," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2761–2784, 2011.

[70] P. E. Farrell, L. Mitchell, and F. Wechsung, "An augmented lagrangian preconditioner for the 3d stationary incompressible navier–stokes equations at high reynolds number," *SIAM Journal on Scientific Computing*, vol. 41, pp. A3073–A3096, jan 2019.

[71] J. Moulin, P. Jolivet, and O. Marquet, "Augmented lagrangian preconditioner for large-scale hydrodynamic stability analysis," *Computer Methods in Applied Mechanics and Engineering*, vol. 351, pp. 718–743, 2019.

[72] Y.-h. Shih, G. Stadler, and F. Wechsung, "Robust multigrid techniques for augmented Lagrangian preconditioning of incompressible Stokes equations with extreme viscosity variations," *SIAM J. Sci. Comput.*, vol. 45, no. 3, pp. S27–S53, 2023.

[73] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Numerical Mathematics and Scientific Computation, Oxford University Press, New York, 2005.

[74] A. Ern and J.-L. Guermond, *Finite elements II—Galerkin approximation, elliptic and mixed PDEs*, vol. 73 of *Texts in Applied Mathematics*. Springer, Cham, 2021.

[75] G. H. Golub and C. Greif, "On solving block-structured indefinite linear systems," *SIAM J. Sci. Comput.*, vol. 24, no. 6, pp. 2076–2092, 2003.

[76] P. R. Amestoy, A. Guermouche, J.-Y. L'Excellent, and S. Pralet, "Hybrid scheduling for the parallel solution of linear systems," *Parallel Computing*, vol. 32, no. 2, pp. 136–156, 2006.

[77] V. E. Henson and U. M. Yang, "BoomerAMG: a parallel algebraic multigrid solver and preconditioner," vol. 41, pp. 155–177, 2002. Developments and trends in iterative methods for large systems of equations—in memoriam Rüdiger Weiss (Lausanne, 2000).

[78] A. J. Chorin, "Numerical solution of the Navier–Stokes equations," *Math. Comp.*, vol. 22, pp. 745–762, 1968.

[79] R. Temam, "Sur l'approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires ii," *Arch. Rat. Mech. Anal.*, vol. 33, pp. 377–385, 1969.

[80] J.-L. Guermond, P. Minev, and J. Shen, "An overview of projection methods for incompressible flows," *Computer methods in applied mechanics and engineering*, vol. 195, no. 44-47, pp. 6011–6045, 2006.