Fluides à seuil : interactions modèles et données

Clément Berger

THESE

en vue de l'obtention du grade de Docteur, délivré par

l'ECOLE NORMALE SUPERIEURE DE LYON

École Doctorale N°512
École Doctorale en Informatique et Mathématiques de Lyon

Discipline : Mathématiques

Soutenue publiquement le 27/09/2024, par :

Clément BERGER

Fluides à seuil : interactions modèles et données

Devant le jury composé de :

ACARY, Vincent	DR INRIA	Université Grenoble Alpes	Rapporteur
BOUCHUT, François	DR CNRS	Université Gustave Eiffel	Examinateur
CHUPIN, Laurent	Professeur	Université Clermont Auvergne	Rapporteur
LUCAS, Carine	MCF HDR	Université d'Orléans	Examinatrice
PUSTELNIK, Nelly	DR CNRS	ENS de Lyon	Examinatrice
VIGNEAUX, Paul	Professeur	Université de Picardie Jules Verne	Directeur de thèse

Cette thèse a été réalisée au sein de l'Unité de Mathématiques Pures et Appliquées (UMPA), à l'École Normale Supérieure (ENS) de Lyon.

La majorité des simulations numériques a été effectuée grâce aux ressources du CBPsmn (Centre Blaise Pascal de Simulation et de Modélisation Numérique) de l'ENS de Lyon.

Ce travail a également bénéficié des ressources du projet ANR PRC "VPFlows" (ANR-20-CE46-0006).

Cette thèse a abouti aux productions et communications suivantes :

- C. Berger, D. Coulette, and P. Vigneaux. A metamodel for confined yield stress flows and parameters' estimation. Rheologica Acta, 2024. doi:10.1007/s00397-024-01436-0,

accompagné du code source et des données :

- C. Berger and P. Vigneaux. Supplementary code for "A metamodel for confined yield stress flows and parameters' estimation", Zenodo, 2023. doi:10.5281/zenodo.8377205,

et aux communications orales :

- CANUM 2024, Île de Ré : exposé dans un mini-symposium
- ICIAM 2023, Tokyo : exposé dans un mini-symposium.

Fluides à seuil : interactions modèles et données

Résumé : Un fluide viscoplastique se comporte comme un solide rigide lorsque la norme du tenseur des contraintes est faible, puis comme un fluide visqueux lorsqu'elle dépasse un certain seuil. Les modèles mathématiques correspondants sont non-différentiables et laissent le tenseur des contraintes indéterminé dans les zones rigides.

On dégage deux classes de stratégies de résolution numérique : modifier la loi de comportement pour régulariser les équations, ou utiliser leur formulation variationnelle afin de se ramener à un problème de minimisation non-lisse. Dans cette thèse, on utilise la seconde approche et on compare différentes méthodes d'optimisation. On confronte d'abord la méthode du Lagrangien augmenté aux méthodes proximales de type FISTA, avec une discrétisation différences finies. On décrit alors le compromis entre vitesse et stabilité, au travers d'écoulements dans une géométrie de type expansion-contraction, exhibant des zones rigides de topologie complexe. On traite ensuite d'une méthode de point intérieur pour laquelle on décrit les calculs et la formulation complète dans le cas du modèle Herschel-Bulkley, ainsi que les difficultés pratiques liées à cette méthode.

Pour finir, on s'intéresse à un modèle de lubrification viscoplastique dans une cavité fermée pour lequel les équations sont simplifiées. On introduit alors un métamodèle combinant polynômes de chaos et réduction de dimension afin de résoudre l'EDP plus rapidement. La méthode est illustrée sur l'estimation de paramètres à partir de données synthétiques et d'expériences physiques. Le code et les données permettant de reproduire et d'adapter la méthode sont mis librement à disposition des praticiens.

Mots-clefs : fluides viscoplastiques, modèle de Bingham, modèle de Herschel-Bulkley, simulations numériques, différences finies, optimisation, FISTA, ADMM, SOCP, modèle de lubrification, métamodèle, réduction de dimension, problème inverse.

Yield stress fluids : model and data interactions

Abstract : A viscoplastic fluid behaves like a rigid solid when the norm of the constraint tensor is low, then like a viscous fluid when it exceeds a given threshold. The corresponding mathematical models are non-differentiable and leave the constraint tensor undetermined in the rigid zones.

There exist two types of strategy for numerical resolution : modify the constitutive law in order to regularize the equations, or use their variational formulation to obtain an equivalent non-smooth minimization problem. In this thesis, the second approach is used and different optimization methods are compared. First, the augmented Lagrangian method is confronted to proximal FISTA-like methods, using finite differences. The compromise between speed and stability is described through flows in expansion-contraction geometries, exhibiting topologically complex rigid zones. Then an interior point method is described, with its complete derivation and formulation in the Herschel-Bulkley case, along with the practical difficulties of this framework.

Finally, we describe a viscoplastic lubrification model in a closed cavity, for which the equations are simplified. In order to accelerate the resolution of the PDE, a metamodel combining polynomial chaos and dimension reduction is introduced. The method is illustrated with parameters' estimations on synthetic data and physical experiments. The code and data necessary to reproduce and adapt the method are publicly available for practitioners.

Keywords : viscoplastic fluids, Bingham model, Herschel-Bulkley model, numerical simulations, finite differences, optimization, FISTA, ADMM, SOCP, lubrification model, metamodel, dimension reduction, inverse problem.

Remerciements

Je tiens tout d'abord à remercier Vincent Acary et Laurent Chupin qui ont accepté de donner de leur temps pour rapporter cette thèse en cette période estivale prompte au repos. Je remercie également François Bouchut, Carine Lucas et Nelly Pustelnik d'avoir accepté de faire partie du jury.

Je remercie bien entendu Paul Vigneaux pour le temps qu'il m'a consacré tout au long de ces 3 ans ainsi que pour les diverses discussions que nous avions eu auparavant alors qu'il était encore mon professeur.

Je remercie chaleureusement Timm Treskatis pour les discussions que nous avons eu au sujet de FISTA, qui m'ont permis de me débloquer et de voir l'algorithme d'un autre œil.

Je ne saurais exprimer l'importance du soutien de David Coulette durant les longues heures de debuggage qui ont rythmé ces dernières années. Son aide dans la navigation à travers les documentations lacunaires m'a sauvé des semaines d'incompréhension.

Je remercie le secrétariat de l'UMPA sur lequel j'ai toujours pu compter, notamment pour la gestion de ma mission au Japon, qui aurait été bien plus complexe dans tout autre laboratoire.

Merci Amélie pour les pauses Rampage Knight. Merci Caroline de m'avoir supporté.

Margaux, merci <3

Sommaire

Chapitre 1

Introduction

1.1 Viscoplasticité

1.1.1 Phénoménologie

Les fluides *viscoplastiques*, aussi dits *à seuil*, sont des fluides non-Newtoniens ayant la particularité de ne se déformer que lorsque les contraintes auxquelles ils sont soumis dépassent une certaine valeur seuil. Si les contraintes ne dépassent pas ce seuil alors le fluide se comporte comme un solide.

Nous pouvons trouver des exemples de tels fluides dans de nombreux contextes. Un exemple de la vie quotidienne est celui du dentifrice. En retournant un tube de dentifrice ouvert, la gravité n'est pas assez forte pour déclencher le comportement visqueux du dentifrice, aussi ce dernier ne s'écoule pas. Si le tube est pressé, alors on peut observer deux comportements sur le matériau. D'une part, sur le bord du tube, une fine couche de dentifrice s'écoule comme un fluide visqueux. D'autre part, au centre, le dentifrice conserve un comportement solide et descend comme un bloc. L'explication est la suivante : au voisinage du tube les contraintes ont suffisamment augmenté pour déclencher le comportement fluide visqueux du dentifrice, alors qu'au centre les contraintes appliquées restent inférieures au seuil, le dentifrice reste donc solide dans cette partie. Le fluide présente donc deux phases différentes, une solide et une cisaillée. Ces zones solides sont caractéristiques des fluides à seuil et dépendent des propriétés rhéologiques du fluide considéré, elles sont généralement appelées *plugs* et leur détermination constitue un point clé de cette étude. Pour finir cet exemple, une fois le dentifrice déposé sur la brosse à dents, seule la gravité exerce une contrainte sur le fluide qui redevient alors entièrement solide.

On trouve de nombreux autres exemples de tels fluides dans différents domaines, on peut citer par exemple le gel pour les cheveux, la lave [225]... Ces modèles jouent un rôle important dans l'industrie pétrolière et gazière [102] ou encore le traitement des déchets liés au forage [256]. Il est parfois possible de contrôler le paramètre de seuil (via des réactions chimiques ou encore des champs électromagnétiques) pour améliorer l'efficacité du transport de certains fluides [57]. Fernández-Nieto et al. ont travaillé sur la simulation d'avalanches à l'aide de ces modèles [90], [91]. On trouvera d'autres exemples dans la figure 1.1 empruntée à Balmforth et al. [21].

1.1.2 Difficultés expérimentales

La complexité de ces fluides donne lieu à une riche littérature sur les études expérimentales que l'on peut mener dessus. Toutefois cette thèse est avant tout dédiée aux simulations numériques portant sur ces fluides, aussi nous renvoyons au papier de Coussot [71] dédié aux aspects expérimentaux. On peut noter toutefois qu'il est encore à l'heure actuelle difficile de déterminer avec précision le paramètre de seuil, détermination qui fait toujours l'objet de nombreuses

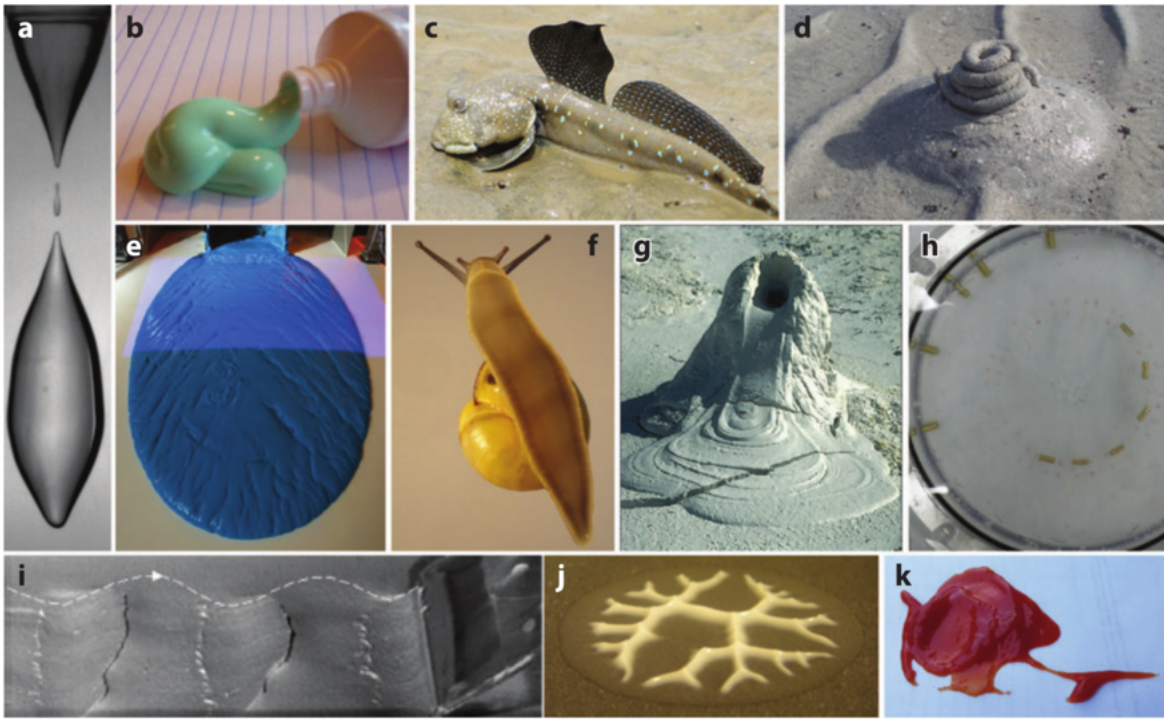


FIGURE 1.1 – Reproduction de la figure de [21] (consulter l'article original pour les références associées). Exemples de matériaux viscoplastiques. (a) Pincement et gouttes de Carbopol. (b) Une extrusion de dentifrice affaissée. (c) Un poisson (périophthalme) en mouvement sur de la boue. (d) Un monticule de boue composé d'un tube hélicoïdal issu d'une extrusion via l'action péristaltique d'un ver des sables au milieu d'ondulations sur la plage. (e) Un *dambreak* de Carbopol bleu sur une surface inclinée. (f) Le dessous d'un escargot ; la couche de mucus joue un rôle clé dans ses déplacements. (g) Un volcan de boue. (h) Fractionnement de particules dans du Carbopol. (i) Frottement d'une planche sur une surface boueuse. (j) Doigts de Saffman-Taylor viscoplastiques. (k) Éclaboussure de ketchup.

recherches [52], [80].

L'existence de tels fluides a en fait en tant que telle été l'objet de multiples débats pendant de nombreuses années. Par exemple, le papier de Barnes and Walters [26] évoque en 1985 l'hypothèse que cet effet de seuil ne soit en fait qu'une illusion due à l'imprécision des instruments de mesure. La controverse s'est poursuivie pendant de nombreuses années (par exemple [25], [226]), notamment alimentée par le paradoxe de lubrification qui sera défini plus tard, voir section 1.2.3. Cette hypothèse a par la suite été réfutée par la majorité de la communauté comme l'explique Coussot [72]. La réalité pratique de ce seuil pour ceux qui manipulent ces fluides justifie à elle seule l'étude de tels modèles. En outre, la littérature expérimentale contient à la fois de bonnes correspondances entre modèles viscoplastiques et réalité terrain [235],[99] comme de moins bonnes [207]. Nous ne débattrons donc pas ici de l'existence de ces fluides et renvoyons à Frigaard [100] et les références qu'il contient.

Quoi qu'il en soit nous considérons ici que ce seuil existe bel et bien de manière exacte, avec donc un comportement purement solide dans le régime de basses contraintes.

1.1.3 Rhéologie

Cette section a uniquement pour objectif de présenter les notions de rhéologie nécessaires à la compréhension des différents modèles régissant l'écoulement d'un fluide à seuil. Pour plus de détails sur la rhéologie ou la mécanique des fluides en général, le lecteur pourra se référer à divers ouvrages dédiés [87],[121], [189],[218].

1.1.3.1 Lois de conservation

À la base des équations qui nous intéressent se trouvent les lois de conservation, lois valables pour tous les matériaux et permettant de les décrire partiellement.

On se place dorénavant sur un domaine $\Omega \in \mathbb{R}^d$.

- La conservation de la masse implique que sur tout Ω, on a :

$$\partial_t \rho + \operatorname{div}(\rho u) = 0, \tag{1.1}$$

  avec u le champ de vitesse, ρ la densité du matériau.
- La conservation de la quantité de mouvement se lit quant à elle :

$$\rho\left(\partial_t u + (u.\nabla u)\right) = \operatorname{div}(\sigma) + f, \tag{1.2}$$

  sur Ω, avec σ le tenseur des contraintes (de dimensions $d \times d$) et f les forces extérieures s'appliquant sur le matériau.

Dans ce travail, on travaille avec des fluides dits *incompressibles*, ce qui signifie que la densité est constante dans le matériau au cours du temps. Il s'en suit que l'équation (1.1) se simplifie en :

$$\operatorname{div}(u) = 0. \tag{1.3}$$

Par ailleurs, il est coutumier de scinder le tenseur des contraintes σ en deux composantes. Pour cela on pose :

$$p = -\frac{1}{d} Tr(\sigma), \tag{1.4}$$

où Tr est l'opérateur de trace. La fonction scalaire p est appelée la *pression*. On en déduit le *déviateur du tenseur des contraintes* (de dimensions $d \times d$) :

$$\tau = \sigma - \frac{1}{d} Tr(\sigma) Id, \tag{1.5}$$

de telle sorte que :

$$\sigma = -pId + \tau. \tag{1.6}$$

L'équation (1.2) se réécrit :

$$\rho\left(\partial_t u + (u.\nabla u)\right) + \nabla p = \operatorname{div}(\tau) + f. \tag{1.7}$$

De plus, un raisonnement sur le tenseur des contraintes permet de justifier sa symétrie :

$$\forall 1 \leq i, j \leq d, \quad \sigma_{ij} = \sigma_{ji}, \quad \tau_{ij} = \tau_{ji}. \tag{1.8}$$

On a donc le système :

$$\begin{cases} \operatorname{div}(u) = 0, \\ \rho(\partial_t u + (u.\nabla u)) + \nabla p - \operatorname{div}(\tau) = f, \\ \forall 1 \leq i, j \leq d, \quad \sigma_{ij} = \sigma_{ji}, \quad \tau_{ij} = \tau_{ji}, \end{cases} \tag{1.9}$$

sur Ω. On écrit généralement (1.7) sous forme adimensionnée :

$$Re\left(\partial_t u + (u.\nabla u)\right) + \nabla p = \operatorname{div}(\tau) + f, \tag{1.10}$$

où :

$$Re = \frac{\rho U_0 L}{\eta}, \tag{1.11}$$

est le *nombre de Reynolds*, η est la viscosité du fluide, U_0 est sa vitesse caractéristique et L est la longueur caractéristique du domaine. On conserve la même notation pour les versions adimensionnées des contraintes, forces extérieures et vitesse du fluide.

Dans ce travail, on se place dans le cas des petites vitesses. En observant la formule (1.11), on en déduit que le premier terme (dit convectif) de (1.10) peut être négligé. Le système d'équations final que l'on déduit de ces lois est donc :

$$\begin{cases} \operatorname{div}(u) = 0, \\ \nabla p - \operatorname{div}(\tau) = f, \\ \forall 1 \leq i, j \leq d, \quad \sigma_{ij} = \sigma_{ji}, \quad \tau_{ij} = \tau_{ji}, \end{cases} \tag{1.12}$$

sur Ω. Les équations (1.12) sont appelées *système de Stokes*.

Remarque 1 *Il existe aussi une loi de conservation de l'énergie, cependant nous nous plaçons ici dans le cas isotherme, aussi n'est-elle pas mise en jeu ici, nous n'en parlerons donc pas plus.*

1.1.3.2 Lois de comportement

Le système (1.12) ne fournit pas assez d'équations pour décrire entièrement le comportement d'un matériau. Pour cela, il faut introduire ce qu'on appelle la *loi de comportement* du fluide. Il s'agit cette fois d'un modèle établi pour rendre compte des comportements macroscopiques observés pour le fluide considéré.

Les lois qui nous intéressent ici permettent de faire le lien entre le déviateur du tenseur des contraintes et le gradient de la vitesse. Cette relation utilise une version symétrisée de ∇u :

$$D(u) = \frac{\nabla u + (\nabla u)^T}{2}, \tag{1.13}$$

que l'on appelle *taux de déformation* du fluide.

Le premier modèle a été proposé par Newton, suivant l'hypothèse : "La résistance qui vient du défaut de lubricité des parties d'un fluide doit être, à toutes choses égales, proportionnelle à la vitesse avec laquelle les parties de ce fluide peuvent être séparées les unes des autres." La formulation tensorielle de ce principe a été donnée par Prager en 1932 [131] :

$$\tau = 2\eta D(u). \tag{1.14}$$

Le coefficient η est la *viscosité*, caractéristique d'un fluide donné. Les fluides suivant ce comportement sont appelés *fluides Newtoniens*. En replaçant (1.14) dans (1.9) on obtient un système couramment dénommé *équations de Navier-Stokes*.

Cependant (1.14) ne suffit pas à décrire l'intégralité des comportements des fluides, il a donc fallu trouver d'autres modèles traduisant le comportement de divers types de fluides. Ces modèles sont variés, certains proposant des équations dépendant du temps, pour rendre compte de phénomènes tels que la *thixotropie*, où le matériau ne répond pas de manière immédiate aux changements de contraintes (on ne considérera pas ici de tels effets). Ici n'est pas l'endroit pour discuter de l'entièreté de ces modèles, nous référons pour cela aux livres proposés plus haut et aux références qu'ils contiennent.

1.1.3.3 Modèles de fluides à seuil

Les premières discussions sur la modélisation de fluides viscoplastiques remontent à la fin du 19e siècle avec les travaux de Schwedoff [227], puis en 1911 avec Bingham et Durham, à l'époque dans le cas unidimensionnel. À partir de 1916, Bingham [45, 46] suscite l'intérêt de la communauté scientifique sur ces matériaux en publiant des expériences sur des fluides dont le comportement suit effectivement une loi de Bingham. Il faudra attendre les années 1930 pour que Prager [202] et de manière indépendante Il'iushin [136] donnent la version tensorielle qui est utilisée de nos jours :

$$\begin{cases} \tau = 2\eta D(u) + \tau_y \dfrac{D(u)}{|D(u)|}, & \text{si } D(u) \neq 0, \\ |\tau| \leq \tau_y, & \text{si } D(u) = 0, \end{cases} \tag{1.15}$$

où l'on définit sur l'espace des tenseurs réels de dimension d le produit scalaire :

$$T : \tilde{T} = \frac{1}{d} \sum_{1 \leq i,j \leq d} T_{ij} \tilde{T}_{ij}, \tag{1.16}$$

ainsi que la norme correspondante :

$$|T| = \sqrt{\frac{1}{d} \sum_{1 \leq i,j \leq d} T_{ij}^2}. \tag{1.17}$$

Le *paramètre de seuil* τ_y de (1.15) dépend du fluide considéré et permet d'encoder le caractère viscoplastique de ce dernier. En effet, dans le cas $D(u) \neq 0$, τ est somme d'un terme visqueux identique à celui d'un fluide newtonien, et d'un terme renormalisé. Cette loi est comparée à la loi d'un fluide de Newton sur la figure 1.2. Ce dernier terme n'est plus défini lorsque $D(u) = 0$, on a alors une non-définition du tenseur des contraintes. Ce cas correspond aux zones solides caractéristiques des fluides à seuil. On s'intéressera souvent à la version adimensionnée de (1.15), à savoir :

$$\begin{cases} \tau = 2D(u) + B \dfrac{D(u)}{|D(u)|}, & \text{si } D(u) \neq 0, \\ |\tau| \leq B, & \text{si } D(u) = 0, \end{cases} \tag{1.18}$$

où :

$$B = \frac{\tau_y L}{\eta U_0}, \tag{1.19}$$

est le *nombre de Bingham* qui donne le rapport entre la contrainte de seuil τ_y et $\eta U_0/L$ la pression caractéristique du système. Ainsi plus B est élevé, plus les effets de la viscoplasticité sont importants. Ce modèle est aujourd'hui encore le plus utilisé pour décrire des fluides à seuil, l'article [102] donnant une idée de l'étendue de l'impact des travaux de Bingham sur la communauté viscoplastique.

Bien que permettant d'exprimer la dualité entre les comportements solides et liquides des fluides à seuil, de nombreux fluides montrent expérimentalement une viscosité non constante. Ils sont dits *rhéo-épaississants* si la viscosité augmente avec $|D(u)|$ et *rhéo-fluidifiants* si elle diminue. Afin de capturer cette caractéristique, Herschel et Bulkley ont en 1926 [127] introduit un modèle similaire à celui de Bingham avec cette fois une viscosité évoluant avec $D(u)$ via une loi-puissance :

$$\begin{cases} \tau = 2^n K |D(u)|^{n-1} D(u) + \tau_y \dfrac{D(u)}{|D(u)|}, & \text{si } D(u) \neq 0, \\ |\tau| \leq \tau_y, & \text{si } D(u) = 0, \end{cases} \tag{1.20}$$

où K est ce qu'on appelle la *consistance* du fluide et $n > 0$. Dans (1.20), la viscosité apparente, à u donné, est donc $K|D(u)|^{n-1}$. On remarque déjà que pour $n = 1$ et $K = \eta$ on retrouve bien le modèle de Bingham, ces équations sont donc bien une généralisation de (1.15). Dans les cas non triviaux $n \neq 1$, on a une viscosité qui augmente avec $|D(u)|$ si $n > 1$, donc un fluide rhéo-épaississant, inversement le cas $n < 1$ donne un fluide rhéo-fluidifiant. Cette nouvelle loi est comparée aux précédentes dans la figure 1.2. Malgré la complexité accrue des équations, la flexibilité donnée par le paramètre n a rendu ce modèle très populaire et ce jusqu'à nos jours (par exemple [106],[107]).

D'autres modèles (tels que le modèle de Casson [60] qui connaît une certaine popularité [7]) ont vu le jour par la suite. Nous nous concentrons dans ce travail sur les plus populaires, à savoir Bingham et Herschel-Bulkley.

Remarque 2 *Étant donné que (1.15) ne fait intervenir que le gradient symétrisé de la vitesse, on pourrait se demander si la condition $D(u) = 0$ correspond bel et bien à un comportement de solide rigide. La propriété suivante permet de s'en assurer :*

Propriété 1 *Soit $v : \Omega \to \mathbb{R}^d \in \mathcal{C}^2$. Si pour tout $x \in \Omega$, $\nabla v(x)$ est antisymétrique, alors ∇f est une fonction constante.*

Preuve :

Soit $1 \leq i, j, k \leq d$. En différentiant toutes les relations $\partial_i f_k = -\partial_k f_i$, on trouve :

$$\partial^2_{ij} f_k = -\partial^2_{kj} f_i = \partial^2_{ki} f_j = -\partial^2_{ij} f_k,$$

d'où $\partial^2_{ij} f_k = 0$. □

On déduit de cette propriété que si $D(u) = 0$ alors u est une fonction affine i.e. u est de la forme $AX + b$ (où $X \in \mathbb{R}^d$), ce qui correspond à une combinaison de translation (avec b) et de rotation rigide (avec A). On a donc bien un comportement de solide rigide.

Remarque 3 *En 1946, Oldroyd [185] a publié un article reprenant le modèle de Bingham avec une modification sur le régime solide. Au lieu de laisser le tenseur τ indéterminé lorsque $D(u) = 0$, il propose un modèle de solide élastique. Bien qu'une majorité de la communauté considère plutôt les modèles dans lesquels seules les régions fluides sont soumises à des équations, ce type*

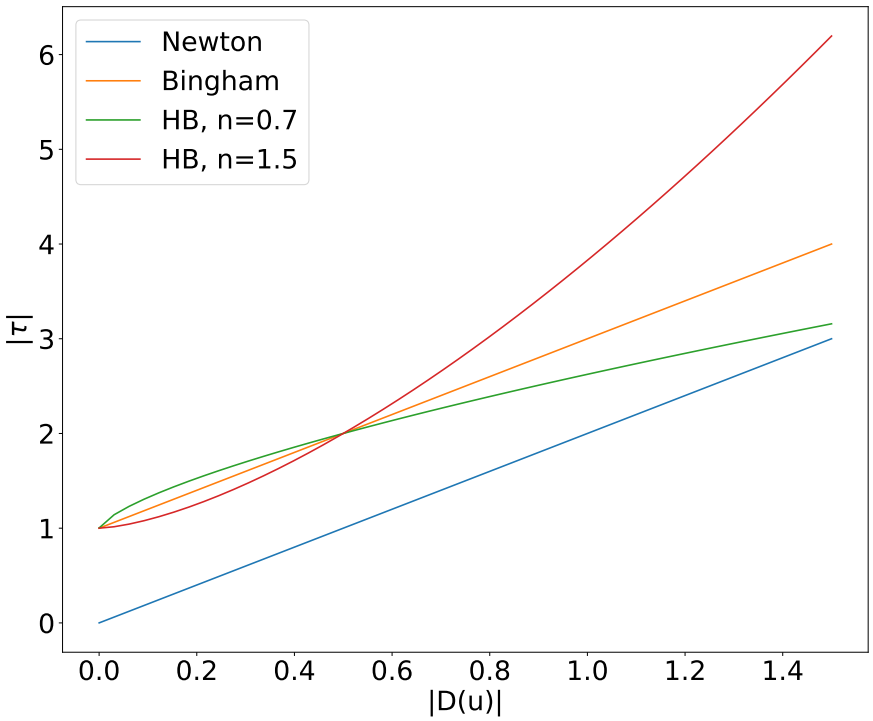


FIGURE 1.2 – Comparaison des lois de comportement de fluides de Newton, Bingham et Herschel-Bulkley (en l'occurence pour $n = 0.7$ et $n = 1.5$).

de modèle engendre une littérature conséquente. Nous ne considérerons pas ici ces modèles aussi nous renvoyons aux articles de Coussot et Rogers [75] ainsi que de Balmforth et al. [19] pour un historique détaillé de l'impact des modèles d'Oldroyd.

1.1.4 Écoulement de Poiseuille

Un exemple illustratif du comportement des fluides à seuil est celui d'un écoulement de Poiseuille, où l'on se place dans un tube de longueur infinie, de rayon $R > 0$, avec un écoulement unidimensionnel déclenché par l'imposition d'un gradient de pression, sans force extérieure.

Mathématiquement, on prend $d = 2$ et on pose $\Omega = \mathbb{R} \times [-R, R]$ et $f = 0$. On suppose que la vitesse est unidirectionnelle i.e. $u = (u^x, 0)$ et on impose le non glissement sur les bords, autrement dit $u^x = 0$ en $y \in \{-R, R\}$. On veut résoudre les équations de Stokes (1.12) complétées par la loi de comportement (1.20). Le tube étant infini, on a forcément :

$$u^x(x, y) = u^x(y), \quad \nabla p = \begin{pmatrix} -dp \\ 0 \end{pmatrix},$$

où dp est constant. On en déduit que :

$$D(u) = \frac{1}{2} \begin{pmatrix} 0 & \partial_y u^x \\ \partial_y u^x & 0 \end{pmatrix}, \tag{1.21}$$

et donc par (1.20) :

$$\tau = \begin{pmatrix} 0 & \tau_{xy} \\ \tau_{xy} & 0 \end{pmatrix}, \quad |\tau| = |\tau_{xy}|.$$

Avec (1.12) on trouve que :

$$\tau_{xy} = y\ dp.$$

Alors la condition de seuil dans (1.20) implique que l'on a un régime fluide si et seulement si :

$$|y\ dp| = |\tau_x y| = |\tau| \leq \tau_y. \tag{1.22}$$

Comme y parcourt $[-R, R]$ en incorporant la condition de non glissement on trouve que :

$$|dp|R > \tau_y, \tag{1.23}$$

est une condition nécessaire et suffisante pour qu'il y ait un écoulement (i.e. $u^x \neq 0$). On suppose donc maintenant que (1.23) est satisfaite et qu'il y a écoulement. Par (1.21), le régime $D(u) = 0$ équivaut à u constant (donc pas de mouvement de rotation rigide). De plus, par (1.22) on a $D(u) \neq 0$ uniquement dans les zones où $|y| > y_c$ où :

$$y_c = \frac{\tau_y}{|dp|}. \tag{1.24}$$

En réinjectant dans (1.20) on trouve la solution :

$$u^x_{pois}(y) = \begin{cases} u_p - \dfrac{n}{n+1}\left(\dfrac{|dp|}{K}\right)^{1/n}(|y| - y_c)^{1+1/n} & \text{si } |y| > y_c, \\ u_p & \text{si } |y| < y_c, \end{cases} \tag{1.25}$$

où :

$$u_p = \frac{n}{n+1}\left(\frac{|dp|}{K}\right)^{1/n}(R - y_c)^{1+1/n}. \tag{1.26}$$

Dans le cas Bingham, en prenant $n = 1$ et $K = \eta$ dans (1.25)-(1.26), on obtient la solution :

$$u^x_{pois}(y) = \begin{cases} u_p + \dfrac{dp}{2\eta}(|y| - y_c)^2 & \text{si } |y| > y_c, \\ u_p & \text{si } |y| < y_c, \end{cases} \tag{1.27}$$

où :

$$u_p = -\frac{dp}{2\eta}(R - y_c)^2. \tag{1.28}$$

Dans le cas Newtonien (i.e. $\tau_y = 0$), la formule (1.27) se résume à une simple parabole centrée en $y = 0$. Dans le cas Bingham $\tau_y > 0$, cette parabole est tronquée en $y = y_c$ où elle devient constante, un schéma est donné en figure 1.3. Dans cette zone, le fluide avance comme un bloc rigide, on a là un plug. Dans le cas Herschel-Bulkley, la forme de la parabole est modifiée selon la valeur de n, on retrouve cependant la même structure avec un plug central.

Cet écoulement montre une conséquence des modèles utilisant un seuil : la régularité de la solution est dégradée. Le raccord de la vitesse entre le plug et la partie fluide est seulement $\mathcal{C}^1$, engendrant des difficultés majeures pour la résolution numérique. Par ailleurs, si dans le cas de l'écoulement de Poiseuille il est possible de calculer τ sur tout le domaine, ce n'est pas possible dans le cas général.

Des résolutions explicites ont été données pour d'autres cas particuliers. On peut citer par exemple les articles de Mosolov et Miasnikov des années 1960 [170],[169],[171] qui s'intéressent aux écoulements antiplan et Poiseuille avec un obstacle. De manière générale, on trouvera d'autres exemples de ce type dans la review de Bird et al. [47] de 1983, qui est qui plus est une des premières reviews exhaustives sur le sujet des fluides viscoplastiques.

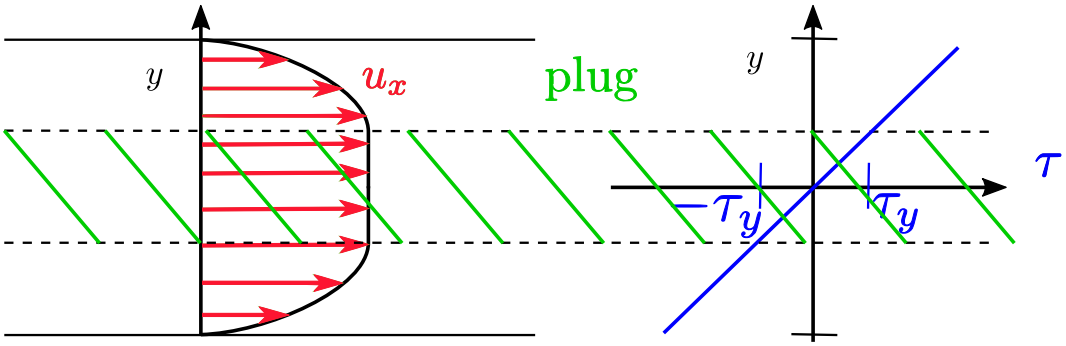


FIGURE 1.3 – Schéma d'un écoulement de Poiseuille pour un fluide de Bingham.

1.2 Éléments théoriques

On s'intéresse désormais aux modèles de Bingham (1.15) et Herschel-Bulkley (1.20) couplés aux équations de Stokes (1.12). Il faut aussi y ajouter des conditions de bord qui seront détaillées en temps utile, telles que la condition de non glissement évoquée en section 1.1.4.

Cette section a pour but de donner les différentes formulations équivalentes qui sont utilisées dans ce document ainsi que de donner quelques justifications théoriques du caractère bien-posé de ces modèles.

1.2.1 Formulations variationnelles

1.2.1.1 Problèmes de minimisation

Il a fallu attendre 1940 à l'Est avec Il'iushin [136] et indépendamment 1952 à l'Ouest avec Prager [203] pour obtenir les premières formulations variationnelles, à l'époque pour le modèle de Bingham. Commençons par introduire une description des conditions de bord. On divise la frontière de Ω en deux sous-ensembles $\partial\Omega_N$ et $\partial\Omega_D$ qui contiennent des conditions respectivement de type Neumann et Dirichlet :

$$\begin{cases} \sigma \cdot \mathbf{n} = g & \text{sur } \partial\Omega_N \\ u = u_D & \text{sur } \partial\Omega_D, \end{cases} \tag{1.29}$$

où $\mathbf{n}$ désigne la normale extérieure à $\partial\Omega_N$, g et u_D sont des fonctions prescrites respectivement sur $\partial\Omega_N$ et $\partial\Omega_N$, et l'on réutilise le tenseur total des contraintes $\sigma = -pId + \tau$. Énonçons dans le cas Herschel-Bulkley le théorème qu'ils avaient à l'époque énoncé dans le cas Bingham :

Théorème 1

Soit (u, σ) une solution continue de (1.12)-(1.20)-(1.29) admettant des dérivées continues par morceaux.

1. *Supposons que $\partial\Omega_D$ soit non vide. Alors u minimise la fonctionnelle :*

$$H(v) = \int_\Omega \left(\frac{2^n K}{n+1} |D(v)|^{n+1} + \tau_y |D(v)| \right) - \int_\Omega \langle f, v \rangle - \int_{\Omega_N} \langle g, v \rangle. \tag{1.30}$$

2. *Supposons que $\partial\Omega_N$ soit non vide. Alors σ maximise la fonctionnelle :*

$$J(\tilde{\sigma}) = 2 \int_{\Omega_D} \langle \tilde{\sigma} \cdot n, u_D \rangle - \frac{1}{2^{\frac{n+1}{n}} K^{1/n}} \int_\Omega (|\tilde{\tau}| - \tau_y)_+^{\frac{n+1}{n}}, \tag{1.31}$$

où $\tilde{\sigma} = \tilde{p} Id + \tilde{\tau}$.

1.2.1.2 Formulation faible

En 1976, Duvaut et Lions [87] donnent la formulation faible correspondant à la formulation forte (1.9)-(1.15)-(1.29), que nous transcrivons ici dans le cas Herschel-Bulkley :

Théorème 2
Posons $\mathcal{U} = \{v \in \mathcal{C}^\infty / div(v) = 0, u|_{\partial\Omega_D} = u_D\}$ *et prenons* $g = 0$ *dans* (1.29). *Soit* H *l'adhérence de* $\mathcal{U}$ *dans* $H^1(\Omega)$. *Soit* u *une solution de* (1.9)-(1.20)-(1.29). *Alors,* $\forall v \in H$ *:*

$$\rho \int_\Omega \langle \partial_t u + (u.\nabla)u, v-u\rangle + \int_\Omega 2^n K |D(u)|^{n-1} D(u) : D(v-u) \\ +\tau_y \int_\Omega |D(v)| - |D(u)| \geq \int_\Omega \langle f, v-u\rangle. \tag{1.32}$$

Preuve : Cette preuve suit celle de [87].

Soit u solution de (1.9)-(1.20)-(1.29) et $v \in H$. On multiplie la 2e équation de (1.9) par $v-u$ et on intègre :

$$\rho \int_\Omega \langle \partial_t u + (u.\nabla)u, v-u\rangle + \int_\Omega \tau : D(v-u) = \int_\Omega \langle f, v-u\rangle.$$

Or par (1.20) on a sur Ω :

$$\tau : D(v-u) = 2^n K |D(u)|^{n-1} D(u) : D(v-u) + \tau_y \frac{D(u) : D(v-u)}{|D(v)|}.$$

Enfin, en utilisant l'inégalité de Cauchy-Schwarz, on obtient bien le résultat. □

Remarque 4 *À l'heure actuelle il semble qu'il n'y ait pas de preuve rigoureuse du sens réciproque i.e. qu'une solution faible soit aussi une solution forte.*

Remarque 5 *On s'est ici replacé dans un cas où l'on considère une évolution en temps. Dans un tel cadre il convient de rajouter aux conditions de bords* (1.29) *une condition initiale du type* $u(t=0) = u_0$. *Si l'on se place à bas Reynolds, il suffit de supprimer de terme en* ρ *dans* (1.32). *On se retrouve avec :*

$$\forall v \in H, \quad \int_\Omega 2^n K |D(u)|^{n-1} D(u) : D(v-u) + \tau_y \int_\Omega |D(v)| - |D(u)| \geq \int_\Omega \langle f, v-u\rangle. \tag{1.33}$$

Remarque 6 *On peut facilement retrouver la forme du problème de minimisation* (1.30) *à partir de cette formulation faible en remarquant que :*

$$|D(u)|^{n-1} D(u) : D(v-u) \leq \frac{|D(v)|^{n+1} - |D(u)|^{n+1}}{n+1},$$

et en remplaçant le terme correspondant dans (1.33).

Remarque 7 *Ces formulations faibles ont été introduites par Duvaut et Lions [87]. En parallèle de cet ouvrage, les années 1970 voient alors l'apparition de plusieurs travaux fondamentaux liés à ces formulations, tels que [110],[112] et [113]. À l'Est, Mosolov et Miasnikov [170, 169, 171] travaillent aussi sur des formulations variationnelles similaires durant la même période, ainsi que sur différents cas de résolution analytique.*

1.2.1.3 Formulation avec des multiplicateurs

Dans le même ouvrage, Duvaut et Lions [87] donnent une autre formulation du problème en introduisant ce qu'ils appellent un multiplicateur. Nous renvoyons à [87] pour la démonstration.

Théorème 3 *À condition initiale donnée, la formulation (1.32) est équivalente à l'existence d'un tenseur symétrique de trace nulle $m \in L^{\frac{n+1}{n}}$ tel que pour tout $v \in H$:*

$$\begin{cases} \rho \int_\Omega \langle \partial_t u + (u.\nabla)u, v\rangle \int_\Omega 2^n K |D(u)|^{n-1} D(u) : D(v) + \tau_y \int_\Omega m : D(v) = \int_\Omega \langle f, v-u\rangle, \\ |m(t,x)| \leq 1, \\ m : D(u) = |D(u)|. \end{cases} \tag{1.34}$$

Remarque 8 *On peut à nouveau formuler le contexte de bas Reynolds :*

$$\begin{cases} \int_\Omega 2^n K |D(u)|^{n-1} D(u) : D(v) + \tau_y \int_\Omega m : D(v) = \int_\Omega \langle f, v-u\rangle, \\ |m(t,x)| \leq 1, \\ m : D(u) = |D(u)|. \end{cases} \tag{1.35}$$

1.2.1.4 Formulation avec un opérateur de projection

Nous mentionnons une dernière formulation introduite par Glowinski et al. [113]. À notre connaissance, elle n'a été introduite que dans le cas Bingham, toutefois rien n'empêche de l'écrire dans le cas Herschel-Bulkley général. Pour cela réécrivons la relation (1.20) sous la forme :

$$\tau = 2^n K |D(u)|^{n-1} D(u) + \tau_y m,$$

où m est un tenseur symétrique tel que $|m| \leq 1$ et $m : D(u) = |D(u)|$. On a alors $|m| = 1$ dans les zones fluides, et $|m| < 1$ dans les zones plastiques.

Remarque 9 *On peut voir cela comme une réinterprétation de (1.35) où l'on souhaite réintégrer le multiplicateur m dans la loi de τ.*

Introduisons l'ensemble $\Lambda = \{\lambda \in L^{\frac{n+1}{n}}(\Omega)_{sym} / |\lambda| \leq 1 p.p.\}$. Si on note P_Λ la projection orthogonale sur Λ (qui est un ensemble convexe), alors pour tout $r > 0$, on a :

$$|m| \leq 1, m : D(u) = |D(u)| \Leftrightarrow m = P_\Lambda(m + r\tau_y D(u)),$$

(on pourra consulter [66] proposition 1). On a la formulation suivante :

$$\begin{cases} 2^n K \int_\Omega |D(u)|^{n-1} D(u) : D(v-u) + \int_\Omega m : D(v) = \int_\Omega \langle f, v-u\rangle, \\ m = P_\Lambda(m + r\tau_y D(u)), \end{cases} \tag{1.36}$$

où l'on écrit ici le cas à bas Reynolds pour alléger les équations.

1.2.2 Existence et régularité

Duvaut et Lions [87] montrent en 1976, grâce à la formulation faible qu'ils ont introduite, le théorème suivant :

Théorème 4 *Pour tout $d \geq 1$, en prenant $n = 1$ (cas Bingham) si $f \in L^2$, alors il existe u solution de (1.32) telle que $u \in H^1$. Si $d = 2$, la solution est unique.*

Remarque 10 *Pour montrer ce théorème, Duvaut et Lions [87] modifient la loi de comportement (1.15) pour la régulariser, puis passer à la limite. Cette idée sera reprise pour le développement de méthodes numériques, voir section 1.3.1.*

Remarque 11 *De nombreux autres théorèmes de ce genre existent. Par exemple, dans le cas $d = 2$, Treskatis [237] montre aussi l'existence et l'unicité dans le cas Herschel-Bulkley pour $0.5 \leq n \leq 1$. Au début des années 2000, des résultats pour le cas compressible ont vu le jour, on peut citer Basov et Shelukhin [29],[230],[30]. Le livre de Fuchs et Seregin [104] s'intéresse à la régularité du problème stationnaire pour le cas Herschel-Bulkley. On termine avec Bulicek et al. [56] qui montrent des résultats d'existence locale dans le cas Herschel-Bulkley. Cependant il n'y a à notre connaissance toujours pas de résultat comparable au théorème 4 pour le cas Herschel-Bulkley général.*

1.2.3 Modèles de lubrification

Résoudre directement les équations complètes peut être numériquement très coûteux. Or, dans de nombreuses applications, le ratio entre l'épaisseur d'un écoulement et sa longueur est très faible, aussi il est possible de procéder à des approximations des équations via des développements asymptotiques ou des intégrations selon une dimension (voir Saramito et Wachs [223] paragraphe 3.7). D'abord introduite dans le cadre des fluides Newtoniens par Barré de Saint-Venant en 1871 [27], l'approche a été appliquée dans le cas de fluides viscoplastiques à partir des années 1980-1990 (voir Bird et Dai [76], Liu et Mei [154]). En 2006, Balmforth et al. [20] s'intéressent plus particulièrement à l'écoulement d'un fluide viscoplastique selon un plan incliné (qui peut par exemple constituer une méthode pour déterminer le seuil τ_y d'un fluide donné, voir par exemple Coussot et al. [73]). Partant d'un modèle de Herschel-Bulkley, ils adimensionnalisent les équations en mettant en valeur le ratio ε entre l'épaisseur typique de l'écoulement et sa longueur. En ne conservant que les termes d'ordre le plus bas en ε, ils se ramènent alors à une EDP sur la hauteur du fluide, qui ne dépend plus que des axes horizontaux x et y. En 2023, Vigneaux et al. [244] reprennent cette approche pour modéliser des écoulements en cavité fermée dans le cadre de la remédiation des mines, les symétries du problème leur permettant de se ramener à une EDP à une seule dimension. Des résolutions numériques du problème ont permis de montrer l'existence de deux régimes de remplissage. Les profils calculés sont également en bon accord avec des expériences physiques en laboratoire. On renvoie à [223] et les références qu'il contient pour plus d'applications de cette approche.

Paradoxe de lubrification En 1981 Bird et Dai [76] ont proposé des approximations de type lubrification pour les écoulements de Bingham. Ils seront suivis par Lipscomb et Denn [152] qui mettent en évidence le fameux paradoxe de lubrification : dans des géométries plus complexes que l'écoulement de Poiseuille, ils calculent des zones de plug, i.e. avec un tenseur des contraintes qui ne dépasse pas le seuil τ_y, mais avec $D(u) \neq 0$. Ce paradoxe sera résolu au début des années 1990 d'abord avec Bittleston et Walton [248] puis avec Szabo et Hassager [234] qui étudient des écoulements dans un anneau. L'explication vient du développement asymptotique utilisé dans l'approximation de lubrification. Le développement n'étant pas poussé assez loin, la norme du tenseur des contraintes paraissait ne pas dépasser le seuil, alors que le terme suivant montre que c'est en fait le cas. Ces zones presque rigides sont communément appelées *pseudo-plugs*, elles revêtent un comportement fluide mais avec une déformation de très faible ampleur. Plus de détails seront donnés par la suite par Balmforth et Craster [18].

1.3 Méthodes numériques

La première résolution numérique de modèles viscoplastiques remonte vraisemblablement à la thèse de Fortin [96] en 1972. Glowinski et al. [113] ont ensuite posé les fondations des méthodes numériques pour ce type de problèmes. Cette section a pour objectif de donner un panorama de l'ensemble des méthodes numériques couramment utilisées pour résoudre les équations de Bingham et Herschel-Bulkley. Nous ne nous attarderons cependant pas sur toutes les méthodes, aussi renvoyons-nous à [116],[223] ou encore à [190] pour plus de détails.

1.3.1 Méthodes de régularisation

Les modèles (1.20) et (1.15) sont non-différentiables et qui plus est non définis dans les zones de plug. Devant cette difficulté, de nombreux praticiens se sont tournés vers ce que l'on appelle de manière générale les méthodes de régularisation, c'est-à-dire des méthodes consistant à modifier les équations du modèle de sorte à lever sa singularité. Plus précisément, dans le cas Bingham on remplacera (1.15) par :

$$\tau = 2\eta_\varepsilon D(u), \tag{1.37}$$

où η_ε est une fonction à déterminer qui aura pour objectif d'approcher le modèle d'origine. Historiquement, on trouvera d'abord le modèle dit de Bercovier-Engelman [38] dont on trouve les idées dans [113] :

$$\eta_\varepsilon = \eta + \frac{\tau_y}{\sqrt{\varepsilon^2 + 2|D(u)|^2}}. \tag{1.38}$$

À la limite $\varepsilon \to 0$, on retrouve le modèle (1.15). Le principe est donc de résoudre ce nouveau modèle pour des valeurs de ε aussi faibles que possible. De nombreux autres modèles seront proposés par la suite, on peut citer notamment le modèle exponentiel de Papanastasiou [191] devenu très populaire :

$$\eta_\varepsilon = \eta + \frac{\tau_y}{2|D(u)|}\left(1 - e^{-2\frac{|D(u)|}{\varepsilon}}\right), \tag{1.39}$$

ou encore le modèle de bi-viscosité de O'Donovan et Tanner [184] :

$$\eta_\varepsilon = \begin{cases} \tau_y \dfrac{D(u)}{|D(u)|} & \text{si } |D(u)| > \varepsilon, \\ \tau_y \dfrac{D(u)}{\varepsilon} & \text{sinon.} \end{cases} \tag{1.40}$$

L'idée est de conserver le modèle d'origine dans la zone fluide, mais de changer la zone de plug en une zone visqueuse avec une viscosité presque inifinie. Ces méthodes sont très populaires encore de nos jours de par leur simplicité, autant en terme d'implémentation que conceptuellement, on peut citer par exemple [107]. Mitsoulis et Tsamopulos [165] fournissent une review détaillée de ces méthodes et de leurs applications.

Cependant cette approche possède divers inconvénients. Il y a d'abord un manque de garanties de convergence de la solution du problème régularisé vers la solution du problème d'origine lorsque $\varepsilon \to 0$. Frigaard et Nouar montrent en 2005 [101] que même si les vitesses convergent, ce n'est pas le cas des contraintes, en particulier aux limites de plug, qui sont généralement les zones cruciales à calculer pour ce type d'écoulement. En 1999, Burgos et al. [58] exhibent dans le cas Herschel-Bulkley pour un écoulement antiplan des plugs dont la concavité est inversée. En 2009, Putz et al. [206] comparent des méthodes de régularisation et des méthodes de Lagrangien augmenté (voir section 1.3.2.1) et trouvent des zones de plug pouvant drastiquement changer selon les paramètres choisis pour la régularisation. On peut aussi exhiber des comportements qualitatifs propres aux fluides à seuil, qui sont perdus dans les méthodes de régularisation. Par

exemple, un fluide visqueux s'écoulant dans une conduite s'arrête en un temps infini [31], alors qu'un fluide à seuil s'arrête en temps fini [87],[134],[133]. Par ailleurs, il devient compliqué de rigoureusement distinguer les zones rigides des zones fluides dans beaucoup de ces nouveaux modèles, aussi la détermination des zones plastiques devient plus hasardeuse et ouverte à interprétation. Pour finir, l'objectif de ces modèles est de prendre un ε très petit de sorte à se rapprocher du modèle idéal. Toutefois, lorsque $\varepsilon \to 0$, les systèmes linéaires à inverser sont de moins en moins bien conditionnés. Gagner de la précision devient donc rapidement très coûteux et rend les méthodes numériques instables, rendant bien moins avantageuses ces techniques si une haute précision est requise.

1.3.2 Méthodes exactes

1.3.2.1 Méthodes de dualité

Les méthodes de dualité ne modifient pas le modèle et s'appuient sur la reformulation en problème de minimisation (1.30). On se ramène alors à la minimisation d'une fonctionnelle dont on observe qu'elle est convexe. La difficulté majeure est celle de la non-différentiabilité de cette fonctionnelle, héritage des modèles (1.15) et (1.20). Deux grandes classes de méthodes ont été utilisées jusqu'à présent : les méthodes de Lagrangien augmenté (voir les sections 2.5.3 et 3.1.2) et les méthodes de type FISTA (voir les sections 2.5.4 et 3.1.3). Dans les deux cas, il s'agit de séparer les termes différentiables des termes non différentiables. Dans le cas du Lagrangien augmenté, on introduit alors un Lagrangien pour se ramener à un problème de point selle que l'on résoud par un algorithme d'Uzawa, comme proposé par Céa et Glowinski en 1972 [192] puis popularisé par [97] et [113]. Dans le cas de FISTA, le terme non-différentiable est traité par une méthode proximale introduite d'abord dans le domaine de l'imagerie puis appliquée au contexte viscoplastique par Treskatis en 2016 [239].

Bien que généralement plus lentes que les méthodes de régularisation, les méthodes de Lagrangien augmenté ont été beaucoup utilisées comme expliqué par exemple dans [223],[100],[190], [21],[19],[72],[116]... On les trouve notamment dans le contexte d'écoulements instationnaires de pétroles dans des conduits [246]. Huilgol et You [135] les utilisent avec différents modèles viscoplastiques dans le cadre d'écoulements stationnaires en conduites. Dimakopoulos et al. [83] étudient l'évolution de bulles sur un modèle Herschel-Bulkley en comparant Lagrangien augmenté et régularisation de type Papanastasiou [191]. On les trouve aujourd'hui encore par exemple avec Medina Lino et al. [162] qui essaient d'accélérer les calculs avec l'usage de GPU.

C'est en réaction au temps de calculs élevé de cette nouvelle approche que sont introduites les méthodes FISTA, notamment développées en détails dans le travail de thèse de Treskatis [237],[240]. Bien que la dérivation de l'algorithme soit très différente de celle du Lagrangien augmenté, l'algorithme final en est en fait très proche, mais il semble qu'il soit sensiblement plus rapide. Il a donc depuis son introduction gagné en popularité, on peut citer par exemple Pourzahedi et al. [199] qui étudient le seuil τ_y minimal à appliquer pour maintenir une bulle immobile ou encore les travaux de Muravleva [174].

1.3.2.2 Programmation en cônes du second ordre

En 2015 Bleyer et al. [50] présentent une méthode originale basée sur la formulation (1.35). À partir de cette formulation, il est possible de reformuler le problème sous le formalisme de la programmation en cônes du second ordre (SOCP). Brièvement, on définit le cône du second ordre $\mathcal{Q}$ en dimension $l \geq 2$:

$$\mathcal{Q} = \{x \in \mathbb{R}^l / x_0 \geq \|\bar{x}\|\},$$

où pour $x \in \mathbb{R}^l$, $x_0 \in \mathbb{R}$ est la première composante de x, $\bar{x} \in \mathbb{R}^{l-1}$ est sa queue et $\|.\|$ désigne la norme euclidienne sur $\mathbb{R}^{l-1}$. Il est alors possible de reformuler l'intégralité du problème comme un problème de minimisation linéaire avec l'ajout de contraintes d'appartenance de certaines quantités à des cônes du second ordre. Il est alors possible de faire appel à la riche théorie des méthodes numériques associées aux SOCP [1]. Cette première version possédait l'inconvénient majeur, de par l'introduction de multiples nouvelles variables, de nécessiter la résolution de problèmes linéaires en très grande dimension, la rendant finalement très lente. Toutefois en 2018 Bleyer [49] modifie la méthode en ne reformulant pas tous les termes, réduisant considérablement la taille des systèmes linéaires à résoudre. Le problème obtenu n'est pas linéaire, mais peut être résolu par une méthode de point intérieur (voir section 2.4 pour une description des méthodes de point intérieur puis la section 3.4 pour le cas viscoplastique). Les cas tests donnent des résultats plus rapides que les méthodes de dualité, pour une précision élevée. Il fournit de plus la possibilité de réutiliser la méthodologie via le solveur MOSEK [8].

À notre connaissance, cette méthodologie n'a été que peu reprise par la littérature depuis. Une explication possible pourrait venir de la review de Saramito et Wachs [223] qui met en évidence la possibilité d'interpéter la méthode de point intérieur comme une méthode de régularisation, ce qui peut rebuter une partie de la communauté. Il convient de noter que dans la seconde version de l'algorithme, Bleyer [49] insiste sur le fait que cette méthode ne souffre pas des inconvénients usuels liés aux méthodes de régularisation.

1.3.2.3 Méthodes de projection

Glowinski, Lions et Trémolières proposent dans [113] d'utiliser la formulation avec opérateur de projection (1.36) et d'appliquer un algorithme d'Uzawa sur cette version du problème. Ce type de méthodes a été beaucoup utilisé dans le cadre d'écoulements instationnaires. On peut citer par exemple Wachs et Yu [258] dans le cadre de particules en sédimentation. En 2016, Chupin et Dubois [66] proposent un raffinement de cette méthode sous le nom de méthode de *bi-projection*. On peut en voir des applications par exemple dans [67]. Plus de détails sur ces méthodes peuvent être trouvés dans la thèse de Chalayer [62]. Il est à noter que ces méthodes semblent avoir été principalement utilisées dans le cas Bingham.

1.3.2.4 Méthode de tracking du plug

Dans [234] Szabo et Hassager présentent une méthode dite de *tracking du plug*. Profitant d'une géométrie particulière (écoulement antiplan entre deux cylindres), ils calculent analytiquement les formes possibles de plug. Partant d'une estimation de la localisation du plug, ils résolvent l'écoulement dans la zone fluide (ce qui lève la difficulté principale du modèle). À partir de cette solution dans le domaine fluide, ils déduisent une nouvelle localisation de plug, a priori différente. Ils répètent la procédure pour la nouvelle forme de plug et ce jusqu'à convergence de la localisation du plug. Bien qu'ingénieuse, cette méthode revêt l'inconvénient de devoir être en mesure de calculer les formes possibles de plug, ce qui n'est pas le cas de manière générale.

1.3.2.5 Méthode de Newton

En 2016, Saramito [221] propose d'utiliser un algorithme de Newton en utilisant une reformulation de la loi de Herschel-Bulkley (1.20) :

$$D(u) = \begin{cases} \dfrac{1}{2K^{1/n}} \left(|\tau| - \tau_y\right)^{1/n} \dfrac{\tau}{|\tau|}, & \text{si } |\tau| > \tau_y, \\ 0, & \text{sinon.} \end{cases} \tag{1.41}$$

On voit ici $D(u)$ comme fonction de τ et on peut appliquer un algorithme de Newton amorti sur la nouvelle formulation obtenue. Notons qu'il est nécessaire de généraliser l'algorithme de Newton avec des sous-différentiels car la formulation (1.41) devient non différentiable dans le cas Bingham $n = 1$. Pour le cas d'écoulement antiplan que Saramito présente, les résultats sont très rapides.

Remarque 12 *En 2015, Treskatis et al. [238] trouvent par des méthodes de dualité la même formulation des équations. Le problème ainsi reformulé est résolu via une méthode d'optimisation différente basée sur une approche trust-region (voir remarque 34) de programmation quadratique séquentielle.*

1.3.2.6 Machine learning

Devant les difficultés rencontrées par les différentes méthodes présentées ci-dessus, Muravleva tente en 2018 [172] d'appliquer des méthodes de machine learning au problème de Bingham. L'idée est de réaliser un apprentissage *offline* à l'aide d'une méthode de Lagrangien augmenté puis de procéder à une décomposition orthogonale aux valeurs propres (POD) couplée à des réseaux de neurones. Le modèle final obtient une bonne précision avec une durée d'évaluation considérablement réduite. On note que cette méthodologie reste dépendante de l'algorithme utilisé pendant la phase d'entrainement tant en terme de vitesse que de précision.

1.3.3 Discrétisation

Nous terminons par une note sur les méthodes de discrétisation utilisées pour ces méthodes. Une grande partie de la communauté utilise des schémas de type éléments finis, notamment depuis le développement de la libraire Rheolef [220], qui permet d'utiliser des algorithmes de type Lagrangien augmenté avec aisance. Il est notamment possible de s'appuyer sur de l'adaptation de maillage comme introduit par Roquet et Saramito [222] afin de capturer avec précision les limites de plug. Il est à noter que des précautions sur le type d'élément sont à prendre comme souligné par Treskatis et al. [240], notamment pour l'usage de FISTA, afin de ne pas introduire d'erreurs d'inconsistence entre la méthode d'optimisation et la discrétisation.

En 2008, Muravleva et Olshanskii [173] introduisent deux discrétisations pour des maillages structurés dont le maillage décalé *Marker and Cells* (MAC), on revient sur cette discrétisation en section 3.1.4. Ce maillage sera utilisé notamment pour le Lagrangien augmenté, on peut citer par exemple la thèse d'Arthur Marly [157] qui compare des résultats MAC avec ceux de Roustaei et al. [215],[216],[214] obtenus en éléments finis.

1.4 Plan du manuscrit

Au vu des éléments présentés ci-dessus, on commence par présenter dans le chapitre 2 une introduction aux algorithmes d'optimisation convexe, qui donne les clés de compréhension des différents algorithmes utilisés par l'état de l'art. Le chapitre 3 présente plus en détails les algorithmes de Lagrangien augmenté, FISTA et la méthodologie SOCP dans le cadre de la simulation d'écoulements viscoplastiques. Les algorithmes sont comparés autant d'un point de vue de leur écriture théorique que de leurs performances pratiques sur une géométrie complexe. En particulier, la formulation explicite de la méthodologie SOCP est présentée de manière détaillée et numériquement testée pour la première fois dans le cas Herschel-Bulkley. Pour finir, le chapitre 4 s'intéresse au modèle de lubrification viscoplastique en cavité fermée introduit dans [244]. Après l'établissement d'un solveur numérique adapté à ce cas spécifique, un métamodèle est introduit afin de résoudre des problèmes d'estimation de paramètres.

Chapitre 2

Introduction à l'optimisation convexe

L'objectif de ce chapitre est de donner un panorama représentatif des algorithmes d'optimisation convexe, i.e. des méthodes numériques permettant d'approcher la solution de (2.13) dans le cas où la fonction objectif f et les contraintes g_i sont convexes. La publication de la méthode du simplexe par Dantzig en 1947 pour la programmation linéaire a marqué le début de recherches intensives sur l'optimisation en général. On ne prétend pas ici donner un traitement exhaustif de ces algorithmes, ni même tous les mentionner. On se concentre sur les algorithmes utilisés par la littérature viscoplastique et ceux qui leur ont donné naissance. Certains théorèmes de convergence sont donnés, mais on ne cherche pas à obtenir les théorèmes les plus généraux pour chaque algorithme présenté. Pour limiter la taille du document, on se concentre sur l'optimisation :

- convexe (voir section 2.1 pour une définition),
- continue : on utilise des fonctions d'un espace continu, l'optimisation discrète ne sera pas abordée, ici, on pourra par exemple consulter [44]
- déterministe, on pourra consulter par exemple [193], [219], [231], [253], [69] et les références qu'ils contiennent au sujet des algorithmes usant de variables aléatoires,
- générale : les problèmes à l'étude dans ce document ne rentrent pas dans des cadres restreints tels que la programmation linéaire ou quadratique (une exception est faite dans la section 3.4 où on parle de programmation en cônes de second ordre) aussi on n'aborde que ponctuellement ces sous-champs de l'optimisation convexe, en se concentrant sur les méthodes utilisables dans les cas plus généraux.

Le chapitre 3 est essentiellement basé sur des méthodes d'optimisation non lisse, en particulier les méthodes proximales et les méthodes de point intérieur, aussi les sections 2.4 et 2.5 sont plus détaillées. Après quelques rappels théoriques préliminaires d'analyse convexe, on donne les algorithmes fondamentaux pour l'optimisation sans contrainte. On tente ensuite de donner un échantillon représentatif (sans trop de détails) des algorithmes développés pour traiter la présence de contraintes, avant de présenter les méthodes de point intérieur et les algorithmes proximaux. Pour des traitements récents de chacun des algorithmes, on renvoie à Nesterov (2018) [177], Beck (2017) [33], Chambolle et Pock (2016) [65], Bertsekas (2015) [43], Boyd et Parikh (2014) [193], Allaire (2012) [2], Boyd et Vandenberghe (2004) [54], Alizadeh et Goldfarb (2003) [1]. Une grande partie des notes historiques données ici sont basées sur les textes classiques plus anciens, en particulier Nocedal et Wright (1999) [182] et Polyak (1987) [198] ont beaucoup inspiré ce chapitre.

2.1 Préliminaires

Cette section a pour but de rappeler les notions de convexité nécessaires à la bonne compréhension des algorithmes présentés ci-après ainsi que de leurs interprétations. On ne prétend pas donner ici un cours complet sur l'analyse convexe, on omet notamment la plupart des preuves des théorèmes énoncés. Pour plus de détails, on renvoie aux livres de Bertsekas (2009) [42], Beck (2017) [33], Nesterov (2018) [177], Allaire (2012) [2], Bauschke et Combettes (2011) [32], Boyd et Vandenberghe (2004) [54] ou encore Rockafellar (1997) [209]. Cette présentation est fortement inspirée de celle de Boyd et Vandenberghe (2004) [54] ainsi que de celle de Chambolle et Pock (2016) [65].

Dans la suite, $\mathcal{X}$ (et $\mathcal{Y}$) est un espace Euclidien, muni d'une norme $\|.\| = \sqrt{\langle .,. \rangle}$. Sauf mention contraire, on se limite ici au cas de la dimension finie, bien qu'une majorité de résultats soit valable en dimension infinie. Lorsque l'on parle de fonctions convexes (définition 4), il est d'usage de considérer des fonctions à valeurs dans la droite réelle complétée, on note donc $\overline{\mathbb{R}} = [-\infty, +\infty]$ et sauf mention contraire on considère une fonction $f : \mathcal{X} \to \overline{\mathbb{R}}$. On note $B(x, R)$ la boule de centre $x \in \mathcal{X}$ et de rayon $R > 0$. On utilise parfois des *inégalités généralisées*, à savoir que pour u, v deux vecteurs de $\mathbb{R}^d$, on note $u \succeq v$ pour dire que pour tout i entre 1 et d, $u_i \geq d_i$.

2.1.1 Convexité

Définition 1 *Un ensemble $C \subset \mathcal{X}$ est dit **convexe** si pour tout $t \in [0, 1]$ et tout $x, y \in \mathcal{C}$, $tx + (1 - t)y \in C$.*

Définition 2 *On appelle **domaine** de f, noté dom f, l'ensemble :*

$$dom\ f = \{x \in \mathcal{X} / f(x) < +\infty\}. \tag{2.1}$$

*Si dom $f \neq \emptyset$, et que $\{x \in \mathcal{X} / f(x) = -\infty\} = \emptyset$, alors on dit que f est **propre**.*

Définition 3 *On définit l'**épigraphe** de f par :*

$$epi(f) = \{(x, t) \in \mathcal{X} \times \mathbb{R} / t \geq f(x)\}. \tag{2.2}$$

Définition 4 *La fonction f est dite **convexe** si et seulement si son épigraphe est convexe.*

On donne ici une définition plus simplement manipulable :

Propriété 2 (Inégalité de Jensen) *Une fonction propre est convexe si et seulement si pour tout $t \in [0, 1]$ et pour tout $x, y \in \mathcal{X}$, on a :*

$$f(tx + (1 - t)y) \leq tf(x) + (1 - t)f(y). \tag{2.3}$$

*Elle est dite **strictement convexe** si pour $t \in]0, 1[$ et $x \neq y$ l'inégalité (2.3) est stricte.*

Remarque 13 *Il est nécessaire de passer par la définition 4 car pour une fonction impropre, l'expression (2.3) pourrait donner lieu à $(-\infty) + (+\infty)$ dont la définition est délicate, voir Rockafellar [209].*

Définition 5 *La fonction f est dite α-**convexe** ou **fortement convexe** ou encore α-**fortement convexe** si et seulement si $f - \alpha\|.\|^2/2$ est convexe.*

Remarque 14 *Une fonction fortement convexe est aussi strictement convexe, mais pas l'inverse, en fait on a la proposition suivante :*

Propriété 3 *Une fonction propre est α-convexe si et seulement si :*

$$\forall (x,y) \in \mathcal{X}, \forall t \in [0,1], f(tx + (1-t)y) \leq tf(x) + (1-t)f(y) - \frac{\alpha}{2}t(1-t)\|x-y\|^2. \tag{2.4}$$

Exemples 1

- *Les fonctions linéaires par morceaux sont convexes, mais pas strictement convexes.*
- *La fonction valeur absolue sur $\mathbb{R}$ ou plus généralement la norme 1 en dimension finie est convexe.*
- *La fonction exponentielle sur $\mathbb{R}$ est strictement convexe mais pas fortement convexe.*
- *La fonction de la variable réelle $x \mapsto ax^2$ est $2a$-fortement convexe.*
- *Soit $C \subset \mathcal{X}$ convexe. On définit l'**indicatrice** de C notée $\mathcal{I}_C$ comme :*

$$\mathcal{I}_C(x) = \begin{cases} 0 & \text{si } x \in C \\ +\infty & \text{sinon.} \end{cases} \tag{2.5}$$

  Cette fonction est alors convexe.
- *La fonction qui à $x \in \mathbb{R}^d$ associe la somme de ses $n \leq d$ composantes les plus grandes est une fonction convexe.*

On considère désormais que la fonction f est convexe.

Définition 6 *Une fonction $f : \mathcal{C} \to \overline{\mathbb{R}}$ est dite **semi-continue inférieurement** (on note l.s.c. pour lower semi-continuous) si pour tout $x \in \mathcal{X}$ et toute suite $(x_n)_{n\in\mathbb{N}}$ d'éléments de $\mathcal{X}$ telle que $x_n \underset{n\to\infty}{\longrightarrow} x$, on a :*

$$f(x) \leq \liminf_{n\to\infty} f(x_n). \tag{2.6}$$

Les fonctions présentées dans l'exemple 1 sont toutes l.s.c. et propres. Il s'agit là du cadre de base pour l'optimisation convexe, tous (ou presque) les théorèmes qui suivent concernent des fonctions l.s.c. et propres.

Propriété 4 *Si une fonction f est convexe, alors ses sous-niveaux, i.e. les ensembles définis comme $\{x \in \mathcal{X} / f(x) \leq t\}$ pour un certain $t \in \mathbb{R}$, sont des ensembles convexes.*

On donne en annexe 2.A quelques propriétés supplémentaires liées à la convexité, en particulier des opérations préservant la convexité.

2.1.2 Dérivées

Propriété 5 *Supposons que f (propre, pas forcément convexe) soit différentiable sur son domaine. Alors f est convexe si et seulement si dom f est convexe et :*

$$\forall (x,y) \in \mathcal{X}, f(y) \geq f(x) + \nabla f(x)^T (y-x). \tag{2.7}$$

De même, f est strictement convexe si et seulement si l'inégalité (2.7) est stricte dès que $x \neq y$.

Propriété 6 *Supposons que f (propre, pas forcément convexe) soit différentiable sur son domaine. Alors f est α-convexe si et seulement si dom f est convexe et :*

$$\forall (x,y) \in \mathcal{X}^2, f(y) \geq f(x) + \nabla f(x)^T (y-x) + \frac{\alpha}{2}\|y-x\|^2. \tag{2.8}$$

On a aussi les caractérisations de second ordre.

Propriété 7 *Supposons que f (propre, pas forcément convexe) soit deux fois différentiable sur son domaine. Alors f est convexe si et seulement si dom f est convexe et sa hessienne est semi-définie positive en tout point de dom f. Si elle est définie positive, alors f est strictement convexe.*

Remarque 15 *Il n'y a pas équivalence dans le cas strictement convexe (voir par exemple la fonction de la variable réelle $x \mapsto x^4$).*

Propriété 8 *Supposons que f (propre, pas forcément convexe) soit deux fois différentiable sur son domaine. Alors f est α-convexe si et seulement si dom f est convexe et $\nabla^2 f(x) - \alpha$ est semi-définie positive en tout point de dom f.*

On définit maintenant une notion permettant de généraliser le gradient de f.

Définition 7 *Pour f propre, l.s.c. et convexe, on définit son* ***sous-différentiel*** *au point $x \in \mathcal{X}$ par :*

$$\partial f(x) = \{p \in \mathcal{X} / \forall y \in \mathcal{X}, f(y) \geq f(x) + \langle p, y - x \rangle\}. \tag{2.9}$$

Les éléments de $\partial f(x)$ sont appelés des ***sous-gradients*** *de f.*

Cette notion a l'avantage qu'il n'est pas nécessaire de demander de la régularité sur la fonction f pour définir son sous-différentiel. Évidemment lorsque f est différentiable, son sous-différentiel se réduit à un singleton $\partial f(x) = \{\nabla f(x)\}$. Dans ce cas-là, on assimile $\partial f(x)$ et $\nabla f(x)$.

Remarque 16 *Les règles de calcul de sous-différentiels sont proches de celles des gradients, mais certaines précautions sont nécessaires. Par exemple, il n'est pas toujours vrai que le sous-différentiel d'une somme de fonctions est la somme des sous-différentiels (il y a toujours une inclusion cependant). Pour un résumé des principales règles de calcul et des hypothèses requises, on renvoie à la section 3.8 de [33].*

Remarque 17 *Le sous-différentiel peut être vu comme un cas particulier d'opérateur maximal monotone, i.e. un opérateur multi-valué $T : \mathcal{X} \to \mathcal{P}(\mathcal{X})$ tel que :*

$$\forall (x, y) \in \mathcal{X}, \forall (p, q) \in Tx \times Ty, \langle p - q, x - y \rangle \geq 0. \tag{2.10}$$

On ne fait ici qu'effleurer les notations des opérateurs monotones, afin de permettre la lecture plus aisée de la littérature sur certains algorithmes, on renvoie à Bauschke et Combettes [32] pour un traitement détaillé du sujet.

Remarque 18 *Le sous-différentiel peut aussi être défini pour une fonction non convexe. En fait, on peut caractériser la convexité (ou forte convexité) en remplaçant les gradients dans (2.7) et (2.8) par p pour tout $p \in \partial f(x)$.*

Exemples 2 *Un exemple classique est celui de la valeur absolue (ou plus généralement des fonctions linéaires par morceaux). Sur $\mathbb{R}\backslash\{0\}$, $f := |.|$ est dérivable et son sous-différentiel y est réduit au singleton $\{1\}$ ou $\{-1\}$ selon le signe de x. Cependant pour $x = 0$, la définition (2.9) donne :*

$$\partial f(0) = \{p \in \mathbb{R} / \forall y \in \mathbb{R}, |y| \geq py\}. \tag{2.11}$$

On obtient donc $\partial f(0) = [-1, 1]$. On comprend alors la perte d'information du sous-différentiel par rapport au gradient : en temps normal, suivre le gradient d'une fonction permet, par définition, de décroître sa valeur, alors que suivre une direction du sous-différentiel ne donne pas cette garantie, même localement.

2.1.3 Problème de minimisation

Les deux notions évoquées précédemment permettent de caractériser le minimum d'une fonction convexe de manière simple.

Propriété 9 (Condition d'optimalité de Fermat) *Soit f propre, l.s.c. et convexe. Alors :*

$$x \in \mathcal{X} \text{ minimise } f \text{ sur } \mathcal{X} \Leftrightarrow 0 \in \partial f(x). \tag{2.12}$$

Exemples 3 *Si l'on reprend l'exemple de la valeur absolue, le minimum de la fonction est atteint en $x = 0$. Comme montré dans l'exemple 2, $0 \in \partial|.|(x) = [-1, 1]$, cela prouve donc bien que 0 réalise le minimum global de la fonction. On note que dans ce cas, il y a aussi d'autres éléments non nuls dans le sous-différentiel.*

Si f est en plus différentiable, alors la condition (2.12) devient simplement $\nabla f(x) = 0$. Dans le cas convexe, contrairement au cas général, la condition de premier ordre est nécessaire ET suffisante, ce qui fait une des grandes forces de l'optimisation convexe (dans le cas général, elle est seulement nécessaire).

Plus généralement, considérons le problème de minimisation sous contraintes :

$$\begin{array}{rll} \min\limits_{x \in \mathcal{X}} & f(x) & \\ \text{tel que} & g(x) & \preceq 0, \\ & Ax & = b, \end{array} \tag{2.13}$$

où

$$g = \begin{pmatrix} g_1 \\ \vdots \\ g_m \end{pmatrix}, \; A = \begin{pmatrix} a_1 \\ \vdots \\ a_p \end{pmatrix}, \; b = \begin{pmatrix} b_1 \\ \vdots \\ b_p \end{pmatrix},$$

avec $g_i : \mathcal{X} \to \overline{\mathbb{R}}$, $a_j \in \mathcal{X}$ et $b_j \in \mathbb{R}$. On suppose que :

$$\left(\bigcap_{1 \leq i \leq m} dom\ g_i \right) \cap dom\ f \neq \emptyset, \tag{2.14}$$

on dit que le problème est **faisable**. On note :

$$K = \{x \in dom\ f / g(x) \preceq 0, Ax = b\}, \tag{2.15}$$

de sorte à pouvoir réécrire (2.13) sous la forme :

$$\min_{x \in K} f(x). \tag{2.16}$$

On note $f^* = f(x^*)$ où x^* réalise le minimum dans (2.13), à supposer que le problème possède une solution. Si f n'est pas minorée sur K, alors on note $f^* = -\infty$.

Propriété 10 *Supposons que f et les g_i soient propres et convexes. Alors un minimum local de* (2.13) *est global.*

Preuve : La convexité des g_i donne que K est convexe (par la propriété 4). On suppose que l'on a un minimum local $x \in K$, i.e. il existe $R > 0$ tel que x minimise f dans $K \cap B(x, R)$. On suppose que x n'est pas un minimum global, i.e. il existe $y \in K$ tel que $f(y) < f(x)$. Posons :

$$t = \frac{R}{2\|y - x\|} < 1, \; z = (1 - t)x + ty \in B(x, R).$$

Par convexité de K, $z \in K$. De plus, par convexité de f :

$$f(z) \leq (1-t)f(x) + tf(y) < f(x),$$

ce qui contredit que x soit un minimum sur $B(x,R) \cap K$. □

On donne maintenant la version générale de la condition d'optimalité (2.12).

Théorème 5
Soit f propre et convexe. Alors pour tout $x_0 \in dom\ f$, on a :

$$f(x) \leq f(x_0) \Rightarrow \forall g \in \partial f(x_0), \langle g, x_0 - x\rangle \geq 0. \tag{2.17}$$

En particulier, on a :

$$x \in K \text{ minimise } f \text{ sur } K \Leftrightarrow \exists p \in \partial f(x), \forall y \in K, \langle p, y - x\rangle \geq 0. \tag{2.18}$$

En particulier, si f est différentiable cela devient : $\forall y \in K, \langle \nabla f(x), y - x\rangle \geq 0$.

Remarque 19 *Dans le cas convexe, si l'on pose $\tilde{f}(x) = f(x) + \mathcal{I}_K(x)$ (voir la définition (2.5) de la fonction indicatrice) qui est une fonction convexe, alors on peut reformuler le problème (2.13) comme :*

$$\min_{x \in \mathcal{X}} \tilde{f}(x). \tag{2.19}$$

On obtient alors un problème d'optimisation sans contraintes. Le prix à payer est cependant de perdre une éventuelle régularité de la fonction à minimiser, aussi certains algorithmes privilégient une forme plutôt qu'une autre.

2.1.4 Dualité

Pour simplifier certaines notations, on définit $h_j(x) = a_j^T x - b_j$ et h le vecteur des h_j.

Définition 8 *On définit le **Lagrangien** $L : \mathcal{X} \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ associé au problème (2.13) par :*

$$L(x, \lambda, \nu) = f(x) + \lambda^T g(x) + \nu^T h(x). \tag{2.20}$$

*On appelle λ_i le **multiplicateur de Lagrange** associé à la contrainte g_i (de même pour les ν_j). Les vecteurs λ et ν sont appelés les **variables duales** du problème (2.13).*

*On appelle **fonction duale** $J : \mathbb{R}^m \times \mathbb{R}^p \to \overline{\mathbb{R}}$ du problème (2.13), la fonction définie par :*

$$J(\lambda, \nu) = \inf_{x \in \mathcal{X}} L(x, \lambda, \nu). \tag{2.21}$$

Remarque 20 *La fonction duale s'écrit comme l'infimum d'une famille de fonctions affines en (λ, ν), elle est donc concave, même lorsque le problème (2.13) n'est pas convexe.*

Le Lagrangien nous permet de donner une borne sur la valeur de l'optimum. En effet, si x vérifie les contraintes de (2.13) et si les λ_i sont positifs, on a immédiatement :

$$\forall x \in K, J(\lambda, \nu) \leq L(x, \lambda, \nu) \leq f(x) \leq f^*,$$

et donc :

$$J(\lambda, \nu) \leq f^*, \tag{2.22}$$

(l'équation reste vraie même si $f^* = -\infty$). La question est de savoir s'il peut y avoir égalité dans (2.22).

Définition 9 *On définit le* ***problème dual*** *de (2.13) comme :*

$$\begin{array}{ll} \max\limits_{(\lambda,\nu)\in\mathbb{R}^m\times\mathbb{R}^p} & J(\lambda,\nu) \\ \textit{tel que} & \lambda \succeq 0. \end{array} \tag{2.23}$$

On note $d^* \in \overline{\mathbb{R}}$ le maximum de (2.23), et (λ^*, ν^*) des vecteurs permettant de réaliser ce maximum, s'ils existent.

En réponse à cette nomenclature, le problème (2.13) est souvent appelé ***problème primal****.*

Remarque 21 *Au vu de la remarque 20, le problème (2.23) est un problème convexe, peu importe la convexité du problème primal.*

On a la propriété fondamentale :

Propriété 11 (Dualité faible)

$$d^* \leq f^* \tag{2.24}$$

La dualité faible (2.24) permet dans certains cas de calculer simplement une borne inférieure de la solution de (2.13). Le cas crucial pour de nombreux algorithmes de minimisation est le cas de la **dualité forte** i.e. lorsque $d^* = f^*$. Ce n'est pas toujours le cas, on a néanmoins le théorème de Slater :

Définition 10 *On dit que le problème (2.13) vérifie les* ***contraintes de qualifications de Slater*** *ou encore les* ***conditions de Slater*** *si :*

$$\exists x \in relint(K), \quad \forall i \in [1,m], g_i(x) < 0, \quad h(x) = 0, \tag{2.25}$$

*où on définit l'****intérieur relatif*** *de K :*

$$relint(K) = \{x \in K / \exists r > 0, B(x,r) \cap aff(K) \subset K\}, \tag{2.26}$$

*avec $aff(K)$ l'****enveloppe affine*** *de K, i.e. l'ensemble des combinaisons affines de points de K.*

Théorème 6 (Slater)

Si le problème (2.13) vérifie les contraintes de qualifications de Slater et est convexe, alors $d^* = f^*$.

On est maintenant en position pour énoncer le théorème fondamental :

Théorème 7 (Karush, Kuhn, Tucker (KKT))

Supposons que les conditions de Slater soient vérifiées et que le problème soit convexe. Alors $(\tilde{x}, \tilde{\lambda}, \tilde{\nu}) \in \mathcal{X} \times \mathbb{R}^m \times \mathbb{R}^p$ sont solutions des problèmes primaux et duaux si et seulement si :

$$\nabla f(\tilde{x}) + \sum_{i=1}^{m} \tilde{\lambda}_i \nabla g_i(\tilde{x}) + \sum_{j=1}^{p} \tilde{\nu}_j \nabla h_j(\tilde{x}) = 0, \tag{2.27a}$$

$$\tilde{\lambda}_i \geq 0, \qquad \forall i \in [1,m], \tag{2.27b}$$

$$g_i(\tilde{x}) \leq 0, \qquad \forall i \in [1,m], \tag{2.27c}$$

$$\tilde{\lambda}_i g_i(\tilde{x}) = 0, \qquad \forall i \in [1,m], \tag{2.27d}$$

$$h_j(\tilde{x}) = 0, \qquad \forall j \in [1,p]. \tag{2.27e}$$

On note que les conditions (2.27c) et (2.27e) correspondent simplement aux contraintes du problème. La condition (2.27b) donne la convexité du Lagrangien par rapport à x, puis (2.27a) dit que son gradient s'annule en $\tilde{x}$. Enfin la condition (2.27d), souvent appelée **conditions des écarts complémentaires** (*complementarity slackness* en anglais), peut s'exprimer :

$$\tilde{\lambda}_i > 0 \Rightarrow g_i(\tilde{x}) = 0. \tag{2.28}$$

Cela signifie que $\tilde{\lambda}_i$ permet de vérifier si la contrainte g_i est **active** i.e. $g_i(\tilde{x}) = 0$.

Remarque 22 *Le théorème de KKT dépasse le cadre de l'optimisation convexe, bien que ce dernier soit le cadre idéal puisqu'il permet une équivalence dans le théorème 7. On se limite ici à son intérêt en tant qu'outil de construction d'algorithmes d'optimisation. Pour des interprétations et plus de détails, on renvoie par exemple à Boyd et Vandenberghe [54].*

Remarque 23 *La publication du théorème de KKT remonte à 1950 [143], toutefois il semble qu'il ait été découvert dès 1939, lors du Master de Karush, sans toutefois donner lieu à une publication. Un historique détaillé a été publié par Kuhn [142]. On réfère aussi à [141] pour une analyse plus récente.*

2.1.5 Conjuguée de Fenchel

On suppose f propre, l.s.c. et convexe dans toute la section.

Définition 11 *On définit la* ***conjuguée*** *de f notée $f^* : \mathcal{X} \to \overline{\mathbb{R}}$ par :*

$$f^*(y) = \sup_{x \in \mathcal{X}} \langle y, x \rangle - f(x). \tag{2.29}$$

Cette fonction est propre, l.s.c. et convexe. On la trouve sous différents noms suivant les ouvrages, on parle notamment de **conjuguée de Fenchel**, ou encore **conjuguée de Legendre-Fenchel**. Elle possède de nombreuses propriétés que l'on ne développe pas ici pour rester concis. On en donne seulement un échantillon.

Propriété 12 *La* ***bi-conjuguée*** *de f , notée f^{**}, coïncide avec f.*

Remarque 24 *Ce n'est plus vrai si f n'est pas l.s.c. !*

Propriété 13 (Inégalité de Fenchel)

$$\forall (x, y) \in \mathcal{X}^2, f(x) + f^*(y) \geq x^T y. \tag{2.30}$$

Propriété 14 (Identité de Legendre-Fenchel)

$$y \in \partial f(x) \Leftrightarrow x \in \partial f^*(y) \Leftrightarrow f(x) + f^*(y) = \langle y, x \rangle. \tag{2.31}$$

On déduit la caractérisation suivante du sous-différentiel :

Propriété 15

$$\forall x \in \mathcal{X}, \partial f(x) = \underset{y \in \mathcal{X}}{argmax} \{\langle x, y \rangle - f^*(y)\}. \tag{2.32}$$

On introduit une nouvelle propriété au cœur de certains des algorithmes présentés après, la L-régularité.

Définition 12 *Une fonction différentiable f est dite* ***L-régulière*** *si son gradient est Lipschitzien avec une constante $L > 0$, i.e. :*

$$\forall (x, y) \in \mathcal{X}^2, \|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|. \tag{2.33}$$

Théorème 8

Supposons f différentiable et $L > 0$. Sont équivalents :

1. $$f \text{ est } L\text{-régulière}, \tag{2.34}$$

2. $$\forall (x, y) \in \mathcal{X}, f(y) \leq f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|x - y\|^2, \tag{2.35}$$

3. $$\forall (x, y) \in \mathcal{X}, f(y) \geq f(x) + \langle \nabla f(x), y - x\rangle + \frac{1}{2L}\|\nabla f(x) - \nabla f(y)\|^2, \tag{2.36}$$

4. $$\forall (x, y) \in \mathcal{X}, \langle \nabla f(x) - \nabla f(y), x - y\rangle \geq \frac{1}{L}\|\nabla f(x) - \nabla f(y)\|^2, \tag{2.37}$$

5. $$\forall (x, y) \in \mathcal{X}, \forall t \in [0, 1], f(tx + (1-t)y) \geq tf(x) + (1-t)f(y) - \frac{L}{2}t(1-t)\|x - y\|^2. \tag{2.38}$$

Remarque 25 *On note que l'inéquation (2.38) est l'inverse de l'inéquation (2.4) caractérisant la forte convexité. On a même le lien suivant :*

Théorème 9

Soit $\alpha > 0$.

1. *Si f est $1/\alpha$-régulière, alors f^* est α-fortement convexe.*
2. *Si f est α-fortement convexe, alors f^* est $1/\alpha$-régulière.*

Ceci permet dans les démonstrations de convergence de certains algorithmes de minimisation de jongler entre les propriétés du problème et celles de son dual. En effet, supposons que l'on veuille résoudre un problème de la forme :

$$\min_{x \in \mathcal{X}} f_0(Dx) + f_1(x), \tag{2.39}$$

où $D : \mathcal{X} \to \mathcal{Y}$ est linéaire bornée et f_0, f_1 sont l.s.c., propres et convexes. Alors le problème dual de (2.39) s'écrit :

$$\max_{y \in \mathcal{Y}^*} - f_0^*(y) - f_1^*(-K^*y). \tag{2.40}$$

Ainsi s'il est possible de calculer les conjuguées de f_0 et f_1, alors il devient possible de donner l'expression explicite du problème dual. On donne en annexe 2.B quelques propriétés supplémentaires sur la conjuguée de Fenchel qui peuvent notamment servir aux calculs pratiques de conjuguées.

2.1.6 Opérateur proximal

On termine cette section par l'introduction d'un opérateur crucial pour l'optimisation convexe non-lisse. On considère dans toute la section une fonction l.s.c., propre et convexe (attention, sans ces hypothèses, les propriétés énoncées plus loin sont généralement fausses).

Définition 13 *On définit l'**opérateur proximal** de f par :*

$$prox_f(x) = \underset{y\in\mathcal{X}}{argmin}\ f(y) + \frac{1}{2}\|x-y\|^2. \tag{2.41}$$

Propriété 16 *Pour tout $x \in \mathcal{X}$, $prox_f(x)$ est un singleton.*

On considérera donc prox_f comme une fonction de $\mathcal{X}$ dans $\mathcal{X}$ en assimilant $\text{prox}_f(x)$ avec l'élément qui le constitue.

Remarque 26 *On parle ici essentiellement de l'opérateur proximal, cependant l'**enveloppe de Moreau** ou encore **régularisée de Moreau-Yosida** de f (introduite à la fin des années 1960 dans les travaux de Yosida [257], voir aussi [10]) définie par :*

$$M_f(x) = \min_{y\in\mathcal{X}} f(y) + \frac{1}{2}\|x-y\|^2, \tag{2.42}$$

possède de nombreuses propriétés. Parmi elles, on observe par exemple que M_f est convexe différentiable et :

$$\nabla M_f(x) = x - prox_f(x). \tag{2.43}$$

Théorème 10
Soit $(x, y) \in \mathcal{X}^2$. Sont équivalents :

1.
$$y = prox_f(x), \tag{2.44}$$

2.
$$x - y \in \partial f(y), \tag{2.45}$$

3.
$$\forall u \in \mathcal{X}, \langle x-y, u-y\rangle \leq f(u) - f(y). \tag{2.46}$$

Corollaire 1

$$\tilde{x} = \underset{x\in\mathcal{X}}{argmin}\ f(x) \Leftrightarrow \tilde{x} = prox_f(\tilde{x}). \tag{2.47}$$

Cette caractérisation est au cœur de la construction de nombreux algorithmes de minimisation. Si prox_f est une application strictement contractante (i.e. δ-lipschitzienne avec $\delta < 1$) alors on peut minimiser f en appliquant itérativement l'opérateur proximal, on obtient un algorithme de point fixe (voir section 2.5.1). L'opérateur proximal n'est contractant en règle générale, on a cependant la propriété suivante :

Propriété 17

$$\forall (x,y) \in \mathcal{X}^2,\ \|prox_f(x) - prox_f(y)\|^2 \leq (x-y)^T\left(prox_f(x) - prox_f(y)\right). \tag{2.48}$$

On peut vérifier que cette propriété est suffisante pour assurer la convergence d'un algorithme de point fixe. Bien sûr la difficulté est transportée sur le calcul de l'opérateur proximal, qui est lui-même un problème de minimisation, aussi de manière générale cela ne semble pas être d'une grande aide. Cependant il existe des cas où il est possible de calculer analytiquement (ou d'approcher numériquement) l'opérateur proximal. Pour cela, il existe de nombreuses règles de calcul permettant de calculer des opérateurs proximaux. On en donne quelques-unes en annexe 2.C. Pour des listes plus exhaustives, on renvoie à [33] ou encore au site internet dédié proximity-operator.net. On donne ici le lien entre opérateur proximal et conjuguée de Fenchel.

Théorème 11 (Décomposition de Moreau)

$$\forall x \in \mathcal{X},\ prox_f(x) + prox_{f^*}(x) = x. \tag{2.49}$$

On termine avec une propriété concernant les fonctions **homogènes d'ordre 1** (i.e. f est à valeurs dans $\mathbb{R}$ et $\forall t \geq 0, f(tx) = tf(x)$), qui semble peu répandue dans la littérature. À la connaissance de l'auteur, elle aurait été publiée pour la première fois en annexe de l'article de Shi et al. en 2017 [231]. Elle s'appuie sur le lemme suivant caractérisant les sous-différentiels des fonctions homogènes :

Lemme 1 *Soit f convexe et homogène d'ordre 1. On a :*

1.
$$\forall t > 0, \forall x \in \mathcal{X}, \partial f(x) = \partial f(\alpha x), \tag{2.50}$$

2.
$$\forall x \in \mathcal{X}, \partial f(x) \subset \partial f(0). \tag{2.51}$$

Propriété 18 *Soit f vérifiant (2.50) et (2.51), en particulier pour f convexe et homogène d'ordre 1. Soit g convexe, propre telle que pour tout $x \in \mathcal{X}$, $prox_g(x) = \lambda_x x$ pour un certain $\lambda_x \geq 0$. Alors :*

$$prox_{f+g} = prox_g \circ prox_f. \tag{2.52}$$

Remarque 27 *On a donné ici une version un peu plus générale de la propriété énoncée par Shi et al., qui prenaient simplement $g = \beta\|.\|_2$. Toutefois comme ils le font remarquer dans la preuve du résultat, ce qui importe est l'existence de λ_x et le fait que ce coefficient soit positif. Cela permet d'utiliser d'autres fonctions possédant cette propriété, ce qui peut parfois se vérifier avec la propriété 32.*

On donne maintenant quelques exemples d'opérateurs proximaux connus.

Exemples 4

- *Soit $f(x) = \frac{1}{2}x^T Ax + b^T x + c$, avec A une matrice symétrique positive. Alors on a : $prox_f(x) = (A + Id)^{-1}(x - b)$.*
- *Regardons la valeur absolue $f : \mathbb{R} \to \mathbb{R}$: $f(t) = \lambda|t|$ pour un certain $\lambda > 0$. Alors :*

$$prox_{\lambda|.|}(t) = \mathcal{T}_\lambda(t) := (|t| - \lambda)_+ sgn(t) = \begin{cases} t - \lambda, & si\ t \geq \lambda, \\ 0, & si\ |t| < \lambda, \\ t + \lambda, & si\ t \leq -\lambda. \end{cases} \tag{2.53}$$

*L'opérateur $\mathcal{T}_\lambda$ est appelé **shrinkage-threshold operator** (traduit grossièrement en opérateur de réduction-seuillage, nomenclature qui s'explique par la formule (2.53)). Il joue*

un rôle crucial en imagerie où l'utilisation de la norme $\|.\|_1$ est prépondérante. En effet, on a :

$$prox_{\lambda\|.\|_1}(x_1, ..., x_n) = (\mathcal{T}_\lambda(x_1), ..., \mathcal{T}_\lambda(x_n)) . \tag{2.54}$$

- *On peut aussi calculer l'opérateur proximal de la norme euclidienne :*

$$prox_{\lambda\|.\|}(x) = \left(1 - \frac{\lambda}{\max(\|x\|, \lambda)}\right), \tag{2.55}$$

pour $\lambda > 0$.
- *Soit $C \subset \mathcal{X}$. Par définition, l'opérateur proximal de l'indicatrice $\mathcal{I}_C$ est l'opérateur de projection sur C. On a vu en remarque 19 qu'on pouvait voir un problème convexe comme un problème sans contrainte sous la forme $\min(f + \mathcal{I}_K)$ où K désigne l'ensemble des contraintes. Si projeter sur K est simple et si l'on peut appliquer la propriété 18, alors une étape d'algorithme proximal comme évoqué plus haut (détaillé en section 2.5.1) revient à projeter l'opérateur proximal de f sur K.*

Remarque 28 *Dans l'esprit de la remarque 17, l'opérateur proximal est associé à la théorie des opérateurs monotones. En effet, $y = prox_f(x)$ peut s'écrire (en utilisant (2.45)) $y = (I+\partial f)^{-1}x$, que l'on appelle la **résolvante** de l'opérateur ∂f en x. Cette notation est beaucoup utilisée dans les articles traitant des algorithmes proximaux car elle permet de les placer dans des cadres plus généraux, nous n'en avons toutefois ici pas la nécessité et ne présentons cette notation que pour simplifier la lecture des papiers cités ci-après. Historiquement, l'opérateur proximal remonte aux travaux de Moreau dans les années 1960 [167],[168]. Le lien avec la théorie des opérateurs monotones semble remonter à Rockafellar 1976 [211].*

Toutes les fonctions qui suivent sont a minima supposées convexes, propres et semi-continues inférieurement.

2.2 Minimisation sans contrainte

Commençons par le cas sans contraintes, i.e. :

$$\min_{x \in \mathcal{X}} f(x), \tag{2.56}$$

avec f convexe. On s'intéresse d'abord à des méthodes pouvant s'écrire sous la forme :

$$x^{(k+1)} = x^{(k)} + t^{(k)} p^{(k)}, \tag{2.57}$$

où $p^{(k)}$ est appelée la *direction* de recherche et $t^{(k)}$ est le *pas*. On parle de *méthode de descente*. L'idée est de trouver, à chaque étape, une direction $p^{(k)}$ qui permet une décroissance de la fonction objectif f. Il faut donc déterminer une procédure pour trouver la direction $p^{(k)}$, ainsi qu'une procédure pour déterminer le pas $t^{(k)}$.

2.2.1 Line search

On commence d'abord par discuter du choix du pas $t^{(k)}$. Il est courant d'utiliser une *line search*, c'est-à-dire une procédure itérative explorant la demi-ligne donnée par $x^{(k)} + tp^{(k)}$, $t > 0$ pour déterminer le choix de $t^{(k)}$.

2.2.1.1 Critère d'arrêt

En pratique, on cherche souvent à obtenir une décroissance suffisamment importante sur f. Le critère le plus communément utilisé est la **condition d'Armijo** :

$$f(x^{(k)} + tp^{(k)}) \leq f(x^{(k)}) + \alpha t \nabla f(x^{(k)})^T p^{(k)}, \tag{2.58}$$

avec $\alpha \in]0, 1[$. Présentée autrement, la condition d'Armijo dit que la décroissance de f doit au moins être proportionnelle à $\nabla f(x^{(k)})$ et t. En pratique, on choisit α petit, par exemple $\alpha \in [0.01, 0.3]$, et on requiert aussi que $x^{(k)} + tp^{(k)} \in dom\ f$. Le lecteur intéressé trouvera une discussion détaillée des critères d'arrêt dans le papier de référence d'Ortega et Rheinboldt [188] publié en 1970.

Remarque 29 *La condition (2.58) demande l'évaluation de $\nabla f(x^{(k)})$. On remarque que cette quantité est indépendante de t, aussi n'a-t-on besoin que d'une seule évaluation de ∇f pour la procédure complète.*

Remarque 30 *Il existe d'autres conditions possibles, on pourra consulter Nesterov [177] ou encore Nocedal et Wright [182].*

2.2.1.2 Backtracking

Dans un algorithme de descente, on suppose généralement que :

$$\nabla f(x^{(k)})^T p^{(k)} < 0. \tag{2.59}$$

Un développement de Taylor montre que si (2.59) est vérifiée, alors pour un $t > 0$ assez petit, $f(x^{(k)} + tp^{(k)}) < f(x^{(k)})$ (à supposer que $x^{(k)}$ ne réalise pas le minimum de f). On peut alors chercher à satisfaire la condition d'Armijo (2.58) avec le simple algorithme de **backtracking** 1.

Algorithme 1 Backtracking line search.

Requis : $\alpha, \beta \in]0, 1[$

1: $t = 1$
2: **Tant que** (2.58) n'est pas vérifiée, **faire**
3:

$$t = \beta t \tag{2.60}$$

4: **Fin Tant que**
5: **Retourner** t

Pour le résumer, on démarre avec $t = 1$ puis on réduit t itérativement selon (2.60) jusqu'à vérifier la condition d'Armijo (2.58). Le paramètre β permet de choisir le compromis entre vitesse et précision de l'algorithme. Typiquement, pour une grande rapidité on prendra $\beta \simeq 0.1$ et pour une plus grande précision on prendra $\beta \simeq 0.8$. Bien que très simple, l'algorithme 1 est en pratique très utilisé.

Remarque 31 *On présente l'algorithme en partant de $t = 1$ car la plupart des cas pratiques utilisent cette initialisation, cependant rien n'empêche de partir d'une valeur plus importante. Il est à noter toutefois que cette procédure est particulièrement adaptée dans le cas où l'on cherche un minimum sur un intervalle de type $]0, a]$ (éventuellement $[-a, 0[$), et non sur intervalle $[a, b]$ quelconque. Pour un tel intervalle, on recommande plutôt l'algorithme 2 qui suit.*

2.2.1.3 Golden section

Un autre algorithme classique pour la minimisation d'une fonction unidimensionnelle est l'algorithme appelé **golden section** 2 (section dorée). On pose ici $h(t) = f(x^{(k)} + tp^{(k)})$. On cherche à approcher le minimum de h.

L'idée est de partir d'un intervalle $[a_j, b_j]$, qui doit supposément contenir le minimum de la fonction h, puis de réduire itérativement cet intervalle sur la base d'évaluations de la fonction. Une bissection consisterait à évaluer la fonction en a_j et b_j, puis en posant $c_j = (a_j + b_j)/2$, on prend $[a_j, c_j]$ comme nouvel intervalle si $h(a_j) < h(b_j)$, $[c_j, b_j]$ sinon. Cela permet de réduire la taille de l'intervalle d'un facteur 2 à chaque itération, au prix d'une évaluation de h. À la place, l'algorithme de la section dorée évalue la fonction h non pas en a_j et b_j mais en deux points t_1^j et t_2^j strictement dans l'intervalle $[a_j, b_j]$, avec disons $a_j < t_1^j < t_2^j < b_j$. Si $h(t_1^j) > h(t_2^j)$, alors le minimum de h ne peut pas se trouver dans l'intervalle $[a_j, t_1^j]$. On peut donc déjà prendre t_1^j comme nouvelle borne inférieure de l'intervalle, soit $a_{j+1} = t_1^j$. On conserve alors la borne supérieure de l'intervalle, $b_{j+1} = b_j$. Il reste à modifier les points tests t_1^j et t_2^j. Pour cela, on impose deux règles. La première règle est de ne pas demander plus d'une évaluation de la fonction à chaque itération, on voudra donc prendre $t_1^{j+1} = t_2^j$. La seconde est une condition de symétrie, on veut :

$$t_1^j - a_j = b_j - t_2^j = \rho(b_j - a_j), \tag{2.61}$$

où ρ est pris constant entre les itérations. Alors avec la condition de symétrie on obtient nécessairement $\rho = (3 - \sqrt{5})/2$ (d'où le nom de la méthode ; voir par exemple [44] section 11.2.2), on en déduit :

$$t_2^{j+1} = b_{j+1} - \rho(b_{j+1} - a_{j+1}). \tag{2.62}$$

L'intervalle de recherche est alors réduit d'un facteur $1/\rho \simeq 2,6$, ce qui au cours des itérations permet une vitesse de convergence bien supérieure à celle de la bissection, à coût égal. Le cas $h(t_1^j) < h(t_2^j)$ est traité de manière analogue et dans le cas idéal où $h(t_1^j) = h(t_2^j)$, on peut même prendre t_1^j et t_2^j comme nouvelles bornes de l'intervalle et calculer à nouveau deux points internes (voir (2.63)), nécessitant certes deux évaluations de h mais permettant de réduire l'intervalle d'un facteur $1/(1 - 2\rho) \simeq 4,2$, ce qui reste plus rentable que la bissection (en pratique, ce cas arrive rarement, sauf si la fonction est constante sur un intervalle). Il est aussi possible de faire varier ρ au cours des itérations, de sorte à avoir une réduction encore plus importante de la taille de l'intervalle de recherche, notamment en utilisant la suite de Fibonacci (voir [198] et les références qu'il contient), cette version semble cependant peu utilisée en pratique.

Algorithme 2 Golden section line search.

Requis : précision ε, intervalle $[0, 1]$

$j = 1$, $a_1 = 0$, $b_1 = 1$, $\rho = (3 - \sqrt{5})/2$, $t_1^1 = a_1 + \rho(b_1 - a_1)$, $t_2^1 = b_1 - \rho(b_1 - a_1)$

Tant que $b_1 - a_1 > \varepsilon$ **faire**

  Si $h(t_1^j) = h(t_2^j)$ **alors**

$$a_{j+1} = t_1^j, \quad b_{j+1} = t_2^j \tag{2.63a}$$
$$t_1^{j+1} = a_{j+1} + \rho(b_{j+1} - a_{j+1}) \tag{2.63b}$$
$$t_2^{j+1} = b_{j+1} - \rho(b_{j+1} - a_{j+1}) \tag{2.63c}$$

  Sinon

    Si $h(t_1^j) > h(t_2^j)$ **alors**

$$a_{j+1} = t_1^j, \quad b_{j+1} = b_j \tag{2.64a}$$
$$t_1^{j+1} = t_2^j, \quad t_2^{j+1} = b_{j+1} - \rho(b_{j+1} - a_{j+1}) \tag{2.64b}$$

    Sinon

$$a_{j+1} = a_j, \quad b_{j+1} = t_2^j \tag{2.65a}$$
$$t_1^{j+1} = a_{j+1} + \rho(b_{j+1} - a_{j+1}), \quad t_2^{j+1} = t_1^j \tag{2.65b}$$

    Fin Si

  Fin Si

Fin Tant que

Retourner $(a_j + b_j)/2$

Remarque 32 *Les algorithmes 1 et 2 énoncés ici sont en pratique très utilisés, pas seulement pour le cas convexe. Si toutefois une plus grande efficacité est requise, il existe des variantes plus récentes avec diverses optimisations, dont certaines tiennent compte de la convexité du problème, voir par exemple [82] et [187].*

2.2.2 Méthodes de gradient

L'idée d'utiliser le gradient de la fonction objectif comme direction de recherche semble avoir été publiée pour la première fois en 1847 par Cauchy [61], où celui-ci cherche à résoudre des systèmes d'équations par le biais d'une minimisation de fonctionnelle. Précisément, la descente de gradient utilise $p^{(k)} = -\nabla f(x^{(k)})$. Cela suppose que la fonction objectif f soit différentiable. Il existe principalement 3 stratégies pour déterminer le pas $t^{(k)}$:

- un pas choisi à l'avance, par exemple $t^{(k)} = t$ ou encore $t^{(k)} = t/\sqrt{k+1}$,
- un pas permettant de minimiser la fonction objectif le long de la direction donnée par $p^{(k)}$, i.e. :
$$t_k = \underset{t \geq 0}{\operatorname{argmin}}\ f(x^{(k)} - t\nabla f(x^{(k)})), \tag{2.66}$$
- un pas donné par une procédure de line search comme présenté en section 2.2.1, typiquement pour satisfaire la condition d'Armijo (2.58) ou une de ses variantes.

Algorithme 3 Descente de gradient pour (2.56).

Requis : x_0

1: **Pour** $k \geq 0$ **faire**
2: Calculer $f(x^{(k)})$, $\nabla f(x^{(k)})$.
3: Choisir le pas $t^{(k)}$.
4:

$$x^{(k+1)} = x^{(k)} - t^{(k)} \nabla f(x^{(k)}). \tag{2.67}$$

5: **Fin Pour**
6: **Retourner** $x^{(k)}$

La première technique a l'avantage de la simplicité, elle demande cependant des hypothèses plus lourdes sur f pour converger. Une hypothèse standard est celle de L-régularité (voir définition 12), auquel cas pour le pas constant $t^{(k)} = t$, il est nécessaire de choisir $0 < tL < 2$ (voir par exemple Polyak [198]) pour assurer la convergence. Le choix optimal est en fait précisément $t = 1/L$, pour lequel on obtient (voir Nesterov [177]) :

Théorème 12
Supposons que f soit L-régulière. Alors, l'algorithme 3 muni du pas constant $t^{(k)} = 1/L$ vérifie :

$$f(x^{(k)}) - f(x^*) \leq \frac{2L\|x_0 - x^*\|^2}{k+4}. \tag{2.68}$$

La deuxième technique n'est en pratique pas utilisée, à moins d'être dans un cas particulier où le problème (2.66) possède une solution analytique.

La troisième technique est la plus utilisée en pratique, elle permet notamment de ne pas avoir à estimer L, pour un coût réduit. La borne asymptotique reste similaire à celle du théorème 12 (voir par exemple Boyd et Vandenberghe [54] ou Nesterov [177]).

2.2.2.1 Gradient conjugué

Avec les années, de nombreuses méthodes de gradient différentes ont été proposées. Parmi elles, la méthode du *gradient conjugué* a connu un grand succès. Elle est d'abord proposée en 1952 par Hestenes et Stiefel [128] pour résoudre des systèmes linéaires régis par une matrice définie positive, puis étendue au cas convexe général par Fletcher et Reeves en 1964 [94] à l'Ouest et Polyak en 1969 [197] à l'Est. La trame générale du gradient conjugué est donnée par l'Algorithme 4. Pour expliquer son origine, supposons que l'on souhaite minimiser une fonction quadratique :

$$\phi(x) = x^T A x - b^T x. \tag{2.69}$$

L'idée du gradient conjugué est de créer des directions de recherche vérifiant la propriété de conjugaison :

$$\forall k_1 \neq k_2, \left(p^{(k_1)}\right)^T A p^{(k_2)} = 0. \tag{2.70}$$

Alors si A est définie positive, on comprend vite que (2.70) implique que forcément seulement un nombre fini de $p^{(k)}$ est non nul, donc la méthode termine en temps fini. On renvoie au chapitre 5 de Nocedal et Wright [182] pour une discussion détaillée. Les différentes versions du gradient conjugué diffèrent par le choix de $\beta^{(k)}$, qui est naturellement déterminé dans le cas quadratique d'origine mais pas dans le cas général. Par exemple, Fletcher et Reeves proposent :

$$\beta^{(k)} = \frac{\|\nabla f(x^{(k+1)})\|^2}{\|\nabla f(x^{(k)})\|^2}. \tag{2.71}$$

Une variante aussi très utilisée en pratique est due à Polak et Ribière [194] :

$$\beta^{(k)} = \frac{\nabla f(x^{(k+1)})^T(\nabla f(x^{(k+1)}) - \nabla f(x^{(k)}))}{\|\nabla f(x^{(k)})\|^2}. \tag{2.72}$$

On termine en notant que bien que dans le cas convexe général, le gradient conjugué ne termine pas en un nombre fini d'étapes, il atteint la vitesse optimale atteignable par les méthodes de premier ordre (voir Polyak [198]).

Algorithme 4 Méthode de gradient conjugué pour (2.56).

Requis : $x^{(0)}$

$k = 0$, $p^{(0)} = -\nabla f(x^{(0)})$.
Tant que $\nabla f(x^{(k)}) \neq 0$ **faire**
  Calculer $f(x^{(k)})$.
  Choisir le pas $t^{(k)}$.

$$x^{(k+1)} = x^{(k)} + t^{(k)}p^{(k)}. \tag{2.73}$$

  Calculer $\nabla f(x^{(k+1)})$.
  Choisir $\beta^{(k)}$.

$$p^{(k+1)} = -\nabla f(x^{(k+1)}) + \beta^{(k)}p^{(k)}. \tag{2.74}$$

  $k = k + 1$.
Fin Tant que
Retourner $x^{(k)}$

2.2.2.2 Accélération

De nombreuses autres méthodes existent, on renvoie aux références évoquées en début de chapitre pour des listes exhaustives. On mentionne la méthode dite *Heavy Ball* proposée par Polyak [198] qui consiste à itérer :

$$x^{(k+1)} = x^{(k)} + \alpha\nabla f(x^{(k)}) + \beta(x^{(k)} - x^{(k-1)}). \tag{2.75}$$

Son nom vient de son interprétation : il s'agit de la discrétisation de l'équation différentielle régissant un corps placé dans un potentiel avec une friction :

$$\mu x'' = -\nabla f(x) - px'. \tag{2.76}$$

Cette modification permet d'obtenir, dans le cas des fonctions fortement convexes, le taux de convergence optimal (voir Polyak [198]). Il convient cependant de noter que choisir α et β est difficile [198].

Une autre forme d'accélération plus populaire est proposée par Nesterov en 1983 [179] (voir aussi Güler 1992 [120]), elle est décrite dans l'algorithme 5. Comme la méthode Heavy Ball, elle n'a quasiment pas de coût de calcul supplémentaire par rapport à la méthode de gradient 3 (elle a cependant une empreinte mémoire pour conserver x^{k-1}).

Algorithme 5 Méthode de gradient accélérée pour (2.56).

Requis : $x^{(0)}$,τ

1: $y^{(0)} = x^{(0)}$, $t^{(0)} = x^{(0)}$.

2: **Pour** $k \geq 0$ **faire**

3:
$$t^{(k+1)} = \frac{1 + \sqrt{1 + 4t^{(k)2}}}{2}. \tag{2.77}$$

4:
$$y^{(k)} = x^{(k)} + \frac{t^{(k)} - 1}{t^{(k+1)}}(x^{(k)} - x^{(k-1)}). \tag{2.78}$$

5:
$$x^{(k+1)} = y^{(k)} - \tau \nabla f(y^{(k)}). \tag{2.79}$$

6: **Fin Pour**

7: **Retourner** $x^{(k)}$

Sous les bonnes hypothèses, on obtient à nouveau le taux de convergence optimal. Par exemple, on peut donner le théorème (voir par exemple [65]) :

Théorème 13
Supposons f L-régulière et $\tau \leq 1/L$. Alors la suite $x^{(k)}$ donnée par l'algorithme 5 vérifie :

$$f(x^{(k)}) - f(x^*) \leq \frac{2\|x^{(0)} - x^*\|^2}{\tau(k+1)^2}. \tag{2.80}$$

Remarque 33 *Ces méthodes s'inscrivent dans le cadre des méthodes multi-step, qui sembleraient avoir été initiées par Polyak dans les années 1960 [195], et dont le principe est d'accumuler l'information contenue dans les étapes précédentes de l'algorithme afin d'élaborer une approximation de f de meilleure qualité (comparée à l'approximation linéaire donnée par le développement de Taylor au point courant $x^{(k)}$). Il est à noter que le gradient conjugué fait partie de ces méthodes et peut être retrouvé à partir de ces idées (voir Polyak [198]).*

2.2.2.3 Sous-gradient

Pour finir, on mentionne la possibilité d'utiliser un sous-gradient de f dans l'algorithme 3 lorsque la fonction f n'est pas différentiable. La méthode semble avoir été proposée par Shor en 1962, puis étudiée par Pshenichnyj dans les années 1970, on renvoie aux notes bibliographiques de Polyak [198]. On peut obtenir des théorèmes de convergence similaires à celui de la convergence de la méthode de gradient classique, la méthode n'est cependant que peu utilisée en pratique, on renvoie à Nesterov [177] pour plus de détails.

2.2.3 Méthodes de Newton

La méthode de Newton pour trouver un zéro d'une fonction est probablement la méthode la plus utilisée lorsque le problème est suffisamment régulier. Or, la condition d'optimalité $\nabla f(x) = 0$ permet de ramener un problème de minimisation à un problème de recherche de zéro. Cette remarque date au moins de 1820 avec Gauss [109] qui traite des problèmes de moindres carrés de cette manière. L'idée de la méthode de Newton est à nouveau celle d'un simple développement de Taylor. À supposer que l'on veuille résoudre :

$$\phi(x^*) = 0. \tag{2.81}$$

Si ϕ est différentiable et que l'on part d'un point x, alors on peut approcher (2.81) par :

$$0 = \phi(x + \Delta x) \simeq \phi(x) + \nabla\phi(x)\Delta x, \tag{2.82}$$

ce qui nous donne :

$$\Delta x = -\left(\nabla\phi(x)\right)^{-1}\phi(x), \tag{2.83}$$

à supposer que la jacobienne de ϕ soit inversible. Si $\phi = \nabla f$, avec f deux fois différentiable on obtient le schéma :

$$x^{(k+1)} = x^{(k)} - \left(\nabla^2 f(x^{(k)})\right)\nabla f(x^{(k)}), \tag{2.84}$$

où $\nabla^2 f$ désigne la hessienne de f.

Sous la forme (2.84), la convergence n'est garantie que dans un voisinage de l'optimum x^*, inconnu à l'avance. Pour pallier cela, on utilise généralement un *Newton amorti*, c'est-à-dire que l'on rajoute un pas $t^{(k)}$, calculé par line search (décrit dans la section 2.2.1), voir l'algorithme 6.

Algorithme 6 Newton amorti pour (2.56).

Requis : x_0

1: **Pour** $k \geq 0$ **faire**
2: Calculer $f(x^{(k)})$, $\nabla f(x^{(k)})$,$\nabla^2 f(x^{(k)})$.
3: Choisir le pas $t^{(k)}$.
4:

$$x^{(k+1)} = x^{(k)} - t^{(k)}\left(\nabla^2 f(x^{(k)})\right)\nabla f(x^{(k)}), \tag{2.85}$$

5: **Fin Pour**
6: **Retourner** $x^{(k)}$

Cet ajout permet d'obtenir une convergence globale tout en gardant la vitesse quadratique que l'on connaît au voisinage de la solution. On peut par exemple donner le théorème suivant (voir Boyd et Vandenberghe [54]) :

Théorème 14

Supposons que f soit deux fois différentiable, de hessienne lipschitzienne de constante L, et m-fortement convexe. Alors pour la suite $x^{(k)}$ produite par l'algorithme 6, il existe $\gamma > 0$ et η tel que $0 < \eta \leq m^2/L$ et :

- *si $\|\nabla f(x^{(k)})\| \geq \eta$, alors :*

$$f(x^{(k+1)}) - f(x^{(k)}) \leq -\gamma, \tag{2.86}$$

- *si $\|\nabla f(x^{(k)})\| < \eta$, alors la line search par backtracking (alg. 1) donne $t^{(k)} = 1$ et :*

$$\frac{L}{2m^2}\|\nabla f(x^{(k+1)})\| \leq \left(\frac{L}{2m^2}\|\nabla f(x^{(k+1)})\right)^2. \tag{2.87}$$

On a donc une convergence en deux phases, d'abord une phase linéaire lorsque l'on est éloigné de la solution, puis une phase quadratique. On peut aisément déduire de (2.86) une borne sur le nombre d'itérations nécessaires pour entrer dans la phase quadratique.

Il existe de nombreuses façons d'interpréter la méthode de Newton, qui ont servi de base à l'élaboration de nombreuses variantes. Outre l'interprétation de méthode de recherche de zéro, on peut voir l'étape de Newton (2.84) comme le résultat de la minimisation d'une approximation quadratique de f (comme dans les méthodes trust-region, voir la remarque 34). De cette interprétation sont nés des travaux sur une régularisation cubique de cette approximation quadratique, donnant une convergence globale de l'algorithme. On renvoie au chapitre 4 du livre de Nesterov [177] pour une description détaillée.

On termine cette section par les méthodes dites de *quasi-Newton*. La direction (2.84) donnée par l'algorithme de Newton peut être en pratique peu efficace pour de nombreuses raisons (empreinte mémoire et/ou calculatoire de la hessienne, résolution du système linéaire, mauvais conditionnement de la hessienne) voire impossible si par exemple la fonction n'est pas deux fois différentiable. Il y a donc de nombreux travaux destinés à trouver des alternatives à la matrice $\left(\nabla^2 f(x^{(k)})\right)^{-1}$, dans le but de retrouver les propriétés de la méthode de Newton. En une dimension, on a par exemple la méthode de la sécante, qui serait connue depuis l'époque de Gauss, et qui a été généralisée en plusieurs dimensions par Wolfe en 1959 [251]. Elle peut être vue comme les prémices des méthodes de quasi-Newton. La première est attribuée à Davidon en 1959 dans un rapport non publié [78] (la méthode ne sera publiée que bien plus tard en 1991 [79], voir [182] introduction du chapitre 8) et ces méthodes seront popularisées par Fletcher et Powell [93] dans les années 1960. Il serait inutile de faire ici une liste des méthodes de quasi-Newton existantes, on renvoie à [198], [182], [44]. On mentionne pour l'exemple une des plus populaires, la méthode BFGS (Algorithme 7) (pour Broyden-Fletcher-Goldfarb-Shanno). On conserve au cours des itérations une approximation $H^{(k)}$ de la hessienne au point courant, que l'on met à jour via (2.92). Cette formule est déterminée de sorte à ce que $H^{(k)}$ soit symétrique définie positive et de sorte à ce que, si :

$$\tilde{f}_k(x) = \langle \nabla f(x^{(k)}), x - x^{(k)} \rangle + \frac{1}{2} \langle H^{(k)}(x - x^{(k)}), x - x^{(k)} \rangle \tag{2.88}$$

est l'approximation quadratique de f que l'on utilise à l'étape k, alors le gradient de $\tilde{f}_k$ et celui de f coïncident. Pour établir de manière unique la formule (2.92), on ajoute que $H^{(k+1)}$ doit être la matrice la plus proche de $H^{(k)}$ vérifiant ces conditions (voir [182]). Les méthodes de quasi-Newton suivent toutes ce schéma et diffèrent dans le choix de la mise à jour de $H^{(k)}$, établie selon des critères plus ou moins différents de ceux énoncés pour BFGS. On note que l'algorithme nécessite une approximation initiale $H^{(0)}$ de la hessienne au point $x^{(0)}$. Il n'y a pas de manière canonique de déterminer $H^{(0)}$, on peut soit l'estimer par différences finies (ou une autre méthode appropriée au problème en question), soit simplement initialiser avec la matrice identité ou une matrice diagonale reflétant des différences d'échelle entre les variables.

Algorithme 7 BFGS pour (2.56).

Requis : $x^{(0)}$, $H^{(0)}$

1: **Pour** $k \geq 0$ **faire**

2: Calculer $f(x^{(k)})$, $\nabla f(x^{(k)})$.

3:
$$p^{(k)} = -H^{(k)} \nabla f(x^{(k)}). \tag{2.89}$$

4: Choisir le pas $t^{(k)}$.

5:
$$x^{(k+1)} = x^{(k)} + t^{(k)} p^{(k)}. \tag{2.90}$$

6: On pose :
$$s = x^{(k+1)} - x^{(k)}, \quad y = \nabla f(x^{(k+1)}) - \nabla f(x^{(k)}), \quad \rho = \frac{1}{y^T s}. \tag{2.91}$$

7:
$$H^{(k+1)} = \left(Id - \rho y s^T\right) H^{(k)} \left(Id - \rho y s^T\right) + \rho s s^T, \tag{2.92}$$

8: **Fin Pour**

9: **Retourner** $x^{(k)}$

Remarque 34 *On présente ici la méthode line search, cependant il existe aussi les méthodes dites trust-region. Là où la line search choisit une direction, puis en déduit un pas via un procédé unidimensionnel, les méthodes trust-region choisissent une zone réduite de l'espace où on considère que l'approximation d'ordre 2 de f donnée par ∇f et $\nabla^2 f$ est fiable. On obtient alors un problème quadratique, sous contrainte d'appartenance à un ensemble réduit (typiquement une boule), i.e. on cherche à résoudre à chaque étape :*

$$\min_{\|x\|\leq\rho} f(x^{(k)}) + \langle \nabla f(x^{(k)}), x - x^{(k)} \rangle + \frac{1}{2}\langle \nabla^2 f(x^{(k)})(x - x^{(k)}), x - x^{(k)} \rangle. \tag{2.93}$$

Il faut ensuite établir une règle pour le choix de la région de confiance (i.e. la valeur de ρ dans la forme (2.93)). Celui-ci est typiquement initialisé à 1 puis réduit ou agrandi selon si $f(x^{(k+1)})$ est proche de ce qui est prédit par l'approximation quadratique du modèle.

Comme pour la procédure de line search, on résout généralement de manière approchée le problème (2.93), en cherchant à satisfaire des conditions de type condition d'Armijo (2.58). Une méthode efficace pour les problèmes convexes est la méthode dogleg dont l'idée est de commencer par suivre la direction $-\nabla f(x^{(k)})$, qui est très fiable à proximité de $x^{(k)}$, puis de poursuivre avec la direction donnée par la méthode de Newton (2.84), qui est plus indiquée lorsque l'on s'éloigne de $x^{(k)}$. Soit on est interrompu par la bordure du domaine, soit on prend le point donné par la somme de ces deux directions. On pourra consulter le chapitre 12 du livre de Bierlaire [44] (notamment 12.1.1 pour la méthode dogleg) ou le chapitre 4 du livre de Nocedal et Wright [182].

2.3 Minimisation sous contraintes : un panorama

On s'intéresse désormais au problème d'optimisation (2.13) complet, sous contraintes (en supposant toujours que f et les g_i sont convexes). Il serait illusoire de présenter de manière exhaustive tous les types de méthodes utilisables pour résoudre un problème aussi général. Le but de cette section est de donner un panorama des méthodologies existantes pour ce type de problèmes. On renvoie à la section 2.4 pour les méthodes de point intérieur et à la section 2.5 pour les méthodes proximales, où ces méthodes sont plus détaillées car utilisées dans les chapitres suivants.

2.3.1 Résolution de problèmes simplifiés

Une première approche consiste à se ramener à des versions simplifiées du problème (2.13). Il y a de nombreuses manières de procéder.

2.3.1.1 Pénalisation des contraintes.

La première méthode dite de *programmation non-linéaire* remonte à 1943, où Richard Courant propose d'utiliser des *fonctions de pénalité* pour traiter les contraintes [70] (la convergence ne sera prouvée que bien plus tard en 1962 par Butler et Martin). Comme son nom l'indique, une fonction de pénalité vient pénaliser la violation d'une contrainte. Supposons par exemple que l'on veuille résoudre :

$$\begin{array}{ll} \min\limits_{x\in\mathcal{X}} & f(x) \\ \text{tel que} & g_0(x) \leq 0. \end{array} \tag{2.94}$$

On peut introduire la fonction :

$$\varphi_{g_0}(x) = [g_0(x)]_+, \tag{2.95}$$

où $[.]_+$ désigne la partie positive. On va alors remplacer le problème (2.94) par le problème sans contrainte :

$$\min_{x\in\mathcal{X}} f(x) + \alpha\varphi_{g_0}(x). \tag{2.96}$$

La fonction φ_{g_0} est non nulle seulement si x ne vérifie pas la contrainte. On peut montrer que pour α suffisamment grand, les problèmes (2.96) et (2.94) possèdent la même solution. On est donc ramenés à la résolution d'un problème sans contrainte. En sommant les fonctions de pénalité, on peut ajouter des contraintes et il est simple d'incorporer des contraintes d'égalité avec des pénalités similaires. Malheureusement, cette méthode a plusieurs inconvénients majeurs, notamment la perte de régularité mais aussi le fait qu'il est impossible de connaître à l'avance le paramètre α. Il n'y a donc aucune garantie que la solution de (2.96) vérifie les contraintes.

Une idée similaire est celle des fonctions *barrières*. Au lieu de pénaliser la sortie de la zone des contraintes, les fonctions barrières pénalisent l'approche de la frontière de la zone définie par les contraintes. On peut définir par exemple une barrière logarithmique :

$$\phi_{g_0}(x) = \begin{cases} -\log(-g_0(x)) & \text{si } g_0(x) > 0, \\ +\infty & \text{sinon.} \end{cases} \tag{2.97}$$

On cherche alors à résoudre :

$$\min_{x\in\mathcal{X}} f(x) + \varepsilon\phi_{g_0}(x). \tag{2.98}$$

Alors pour ε proche de 0, on peut à nouveau montrer que l'on récupère un problème équivalent au problème (2.94). On note que cette fois la solution vérifie nécessairement les contraintes. Ce type de méthodes remonte à 1955 avec Frisch [103]. Elles ont pris en importance dans l'ouvrage de Fiacco et McCormick [92] et ont été grandement popularisées par [138].

2.3.1.2 Approximation des contraintes.

Une autre possibilité est d'essayer de se ramener à un type de problème plus simple que les problèmes convexes généraux, par exemple un problème linéaire. Dans les années 1950-1960, Zoutendjik propose la méthode des *directions admissibles* [261]. Dans cette méthode, il utilise une approximation linéaire de f via son gradient, ainsi que des approximations linéaires de chacune des contraintes (via leur gradient aussi). Il résout alors le problème linéaire consistant à minimiser $\langle \nabla f(x^{(k)}), x\rangle$ sous les contraintes linéaires obtenues. Ces idées seront poursuivies notamment par Pschenichnyi dans les années 1970 [204], [205]. Selon le même principe, Wilson initie au cours de sa thèse, dans les années 1960 [250], les méthodes de *programmation quadratique séquentielle* qui utilisent cette fois un modèle quadratique pour la fonction objectif (et toujours des contraintes linéarisées). Ces méthodes seront développées dans les années 1970. Comme pour les méthodes de Newton et quasi-Newton, de nombreuses méthodes diffèrent selon l'approximation utilisée pour $\nabla^2 f$. Pour plus de détails, on consultera le chapitre 18 de Nocedal et Wright [182] ou encore Boggs et Tolle [51].

2.3.2 Contraintes exactes

Voyons maintenant les méthodes dans lesquelles on traite les contraintes de manière exacte.

2.3.2.1 Contraintes d'ensemble.

On peut voir les contraintes comme l'appartenance à un ensemble, i.e. on suppose qu'on veut minimiser :

$$\min_{x\in K} f(x), \tag{2.99}$$

où K est convexe.

S'il est simple de projeter sur K, alors on peut utiliser la méthode du *gradient projeté* qui consiste simplement à ajouter une étape de projection sur K après l'étape de gradient (2.66). Cette méthode a été trouvée indépendamment dans les années 1960 par Goldstein [117], Dem'yanov et Rubinov [81] et Levitin et Polyak [149]. On peut aussi appliquer la même procédure à la méthode de sous-gradient pour le cas non-lisse, comme proposé par Polyak [196]. On renvoie par exemple à [177] ou encore [198] pour une étude détaillée.

Une variante est la méthode de Frank-Wolfe (ou *gradient conditionnel*), proposée dès 1956 [98] pour le cas quadratique puis donnée dans un cadre plus général par Dem'yanov et Rubinov [81]. Plutôt que de projeter sur K, l'idée est de résoudre le sous-problème d'optimisation :

$$\bar{x} = \operatorname*{argmin}_{x \in K} \langle \nabla f(x^{(k)}), x \rangle, \tag{2.100}$$

puis de poser :

$$x^{(k+1)} = x^{(k)} + t^{(k)}(\bar{x} - x^{(k)}). \tag{2.101}$$

Le sous-problème (2.100) étant linéaire, il est potentiellement plus simple à résoudre, auquel cas la méthode est efficace. La question du choix du pas et de la vitesse de convergence sont étudiées dans les années 1960-1970, on peut citer notamment [149], [86]. On pourra consulter la section 7.2.3 de [198] pour plus de détails. On peut sur le même principe utiliser des méthodes de Newton ou quasi-Newton en cherchant à minimiser une approximation quadratique de f sur K au lieu de $\mathcal{X}$ tout entier, voir section 7.3 de [198].

2.3.2.2 Méthode des plans de coupe *(Cutting plane)*.

Un usage majeur des sous-gradients provient de la caractérisation (2.17). Supposons que l'on veuille résoudre :

$$\begin{array}{rl} \min\limits_{x \in \mathcal{X}} & f(x) \\ \text{tel que} & \forall 1 \leq i \leq m, g_i(x) \quad \leq 0. \end{array} \tag{2.102}$$

Posons $\bar{g}(x) = \max\limits_{1 \leq i \leq m} g_i(x)$. La fonction $\bar{g}$ est convexe, et le problème (2.102) peut se réécrire :

$$\begin{array}{rl} \min\limits_{x \in \mathcal{X}} & f(x) \\ \text{tel que} & \bar{g}(x) \quad \leq 0. \end{array} \tag{2.103}$$

Si on note $K = \{x \in \mathcal{X} / \bar{g}(x) \leq 0\}$ l'ensemble des contraintes, l'équation (2.17) permet de séparer $x \notin K$ de K. L'idée est alors de définir un ensemble contenant le minimum x^*, puis d'utiliser (2.17) pour réduire cet ensemble jusqu'à trouver x^*. On donne une description générale dans l'Algorithme 8. Il existe autant de versions de ces schémas que de choix possibles pour l'élément $x^{(k)} \in E^{(k)}$ et pour le nouvel ensemble $E^{(k+1)}$. Après leur introduction en 1960 par Kelley [140], de nombreux schémas ont émergé. Parmi eux, on note la méthode des *centres de gravité* initiée indépendamment par Levin [148] et par Newman [181] en 1965 (la preuve de convergence sera donnée bien plus tard par Nemirovsky [176]) dont l'idée est, comme son nom l'indique, de choisir pour $x^{(k)}$ le centre de gravité de l'ensemble courant $E^{(k)}$. Le coût de calcul du centre de gravité (une intégrale sur $E^{(k)}$) étant important, on lui préfère souvent la méthode de *l'ellipsoïde* introduite par Nemirovsky [176], dont l'idée est de choisir pour $E^{(0)}$ un ellipsoïde. Alors un choix judicieux du point $x^{(k)}$ permet de s'assurer que $E^{(k)}$ reste un ellipsoïde à chaque étape, avec des mises à jour explicites de chacune des quantités. On renvoie à la section 3.2.8 de Nesterov [177] pour des formules explicites et des théorèmes de convergence sur ces algorithmes.

Algorithme 8 Méthode générale des plans de coupe pour (2.103).

Requis : $E^{(0)} \supset K$

1: **Pour** $k \geq 0$ **faire**

2: Choisir $x^{(k)} \in E^{(k)}$.

3: Si $x^{(k)} \in K$, alors on affecte à g un élément de $\partial f(x^{(k)})$. Sinon, on lui affecte un élément séparant $x^{(k)}$ de K, i.e. un élément de $\partial \bar{g}(x^{(k)})$.

4: Choisir

$$E^{(k+1)} \supset \{x \in E^{(k)} / \langle f, x^{(k)} - x \rangle \geq 0\}. \tag{2.104}$$

5: **Fin Pour**

2.3.2.3 Méthode de gradient réduit.

Dans le cas des contraintes d'égalité, Wolfe [252] propose de diviser x en $x = (u, v)$, où $u \in \mathbb{R}^n$ et $v \in \mathbb{R}^m$, de sorte à ce que les contraintes puissent d'écrire sous la forme $g(u, v^k) = 0$ à chaque itération. On peut alors se ramener à un problème sans contrainte et utiliser des méthodes de gradient. On trouvera plus de détails dans [198] section 8.2.6.

2.3.2.4 Méthodes de Lagrangien.

On termine avec des méthodes utilisant la théorie liée au Lagrangien (2.20). Lorsque les conditions de Slater (2.25) sont vérifiées, on peut chercher à résoudre directement les conditions de KKT (2.27). Selon la forme de la fonction objectif et des contraintes, on peut espérer obtenir par exemple un problème linéaire ou quadratique simple à résoudre. Dans le cas où il n'y a que des contraintes d'égalités, on peut aussi utiliser des méthodes de Newton en contrôlant la violation des contraintes (voir Boyd et Vandenberghe chapitre 10 [54]).

On peut sinon chercher à directement travailler avec le Lagrangien, l'objectif étant de déterminer un *point selle* du Lagrangien, i.e. résoudre :

$$\min_{x} \max_{\lambda \succeq 0, \nu} L(x, \lambda, \nu). \tag{2.105}$$

Une telle approche est particulièrement utilisée dans le cas où la fonction objectif peut s'écrire comme la somme de deux fonctions plus *simples*, comme évoqué en (2.39). En effet, dans ce cas le problème dual peut s'écrire en fonction des conjuguées des deux fonctions selon (2.40), aussi il est souvent possible d'en déduire des algorithmes séparant les traitements de ces deux fonctions. La méthode la plus simple (et historiquement parmi les premières) est la méthode d'Uzawa [9], dont l'idée est de réécrire le problème (2.39) sous la forme :

$$\min_{y = Dx} f_0(y) + f_1(x), \tag{2.106}$$

afin de séparer les variables utilisées par les deux fonctions. Le Lagrangien de (2.106) s'écrit alors :

$$L(x, y, \lambda) = f_0(y) + f_1(x) - \langle \lambda, y - Dx \rangle. \tag{2.107}$$

Pour résoudre le problème (2.105), Uzawa propose alors de commencer par une étape de minimisation en (x, y), qui peut être découplée selon (2.108), puis une *montée* de gradient en la variable duale λ, voir Algorithme 9.

Cet algorithme est en pratique très lent et donc peu utilisé, il sert néanmoins de principe de base à de nombreuses méthodes, dont les méthodes de *Lagrangien augmenté*, dont l'idée est de rajouter un terme quadratique au Lagrangien afin d'améliorer ses propriétés :

$$L_r(x, y, \lambda) = f_0(y) + f_1(x) - \langle \lambda, y - Dx \rangle + \frac{r}{2} \|y - Dx\|^2, \tag{2.110}$$

Algorithme 9 Méthode d'Uzawa pour (2.106).

Requis : $\lambda^{(0)} \in \mathbb{R}^m$

1: **Pour** $k \geq 0$ **faire**

2: Résoudre :

$$x^{(k+1)} = \operatorname*{argmin}_{x \in \mathcal{X}} f_1(x) + \langle \lambda^{(k)}, Dx \rangle, \tag{2.108a}$$

$$y^{(k+1)} = \operatorname*{argmin}_{y \in \mathcal{X}} f_0(y) - \langle \lambda^{(k)}, y \rangle. \tag{2.108b}$$

3: Choisir le pas $t^{(k)}$.

4: Mise à jour de la variable duale :

$$\lambda^{(k+1)} = \lambda^{(k)} + t^{(k)}(Dx^{(k+1)} - y^{(k+1)}). \tag{2.109}$$

5: **Fin Pour**

où $r > 0$ est un coefficient à choisir à l'avance. Cette méthode a pendant longtemps été appelée *méthode des multiplicateurs*, bien que désormais ce nom semble peu utilisé. Elle a été découverte indépendamment par de nombreux acteurs qui en ont donné des versions différentes, notamment dans les années 1960-1970 avec [122], [129] et [200] dans le cas général, et plus spécifiquement dans le cas convexe par [119], [241], [41], [210], [249]. On trouvera rapidement des extensions au cas de contraintes d'inégalités avec par exemple [201] ou encore [212].

La difficulté est qu'avec ce Lagrangien, il devient impossible de découpler (2.108) de manière exacte, raison pour laquelle autant de travaux sont dédiés à l'élaboration d'un algorithme efficace de résolution du Lagrangien augmenté. Il serait illusoire de tous les présenter ici, aussi on renvoie à la section 2.5.3 qui présente un des algorithmes majeurs sur cette question, à savoir ADMM, qui s'interprète aussi comme un algorithme proximal.

2.4 Méthodes de point intérieur

Les méthodes de point intérieur ont émergé à partir des méthodes de barrière évoquées dans la section précédente, basées sur une approximation des contraintes à l'aide de fonctions barrières (2.98). Après avoir été abordées sans grand succès dans les années 1960, elles sont remises au goût du jour dans l'article de Karmarkar [138] de 1984 qui recherche alors une alternative à la méthode du simplexe pour les problèmes linéaires à grande échelle. La méthode gagnera rapidement en popularité et on s'aperçoit alors que cette méthode et ses variantes se généralisent efficacement pour des problèmes convexes plus généraux (voir par exemple Nesterov 1994 [178]). Des versions spécialisées pour différents types de problèmes sont développées avec des résultats très compétitifs, on renvoie à Todd [236] pour la programmation semi-définie et à Alizadeh et Goldfarb [1] pour la programmation en cône de second ordre. L'objectif ici n'est pas de faire une revue détaillée de toutes ces variantes, mais plutôt de présenter les concepts fondateurs de ces méthodes. On trouvera plus de détails dans les ouvrages de Ben-Tal et Nemirovski [37], Nocedal et Wright [182], ou encore Boyd et Vandenberghe [54]. Cette section est essentiellement un condensé de ce dernier ouvrage, avec un accent sur la formulation *primale-duale* de ces méthodes, qui est la plus populaire aujourd'hui et que l'on utilisera en section 3.4. On trouvera une présentation totalement différente dans le chapitre 5 du livre de Nesterov [177].

2.4.1 Principe de la méthode primale

Ici on cherche à résoudre le problème (2.13) complet, où f et les contraintes d'inégalité g_i sont convexes. On suppose que les conditions de qualification de Slater (2.25) sont satisfaites, aussi la résolution de (2.13) équivaut à la résolution des conditions de KKT (2.27). Fondamentalement, le principe de la méthode est d'éliminer les contraintes d'inégalités via une fonction barrière comme présenté en (2.98) dans la section précédente, afin de récupérer un problème avec seulement des contraintes d'égalités linéaires. Les conditions de KKT de ce nouveau problème ne contenant plus d'inéquations, on peut alors le résoudre simplement par une méthode de Newton (voir la section 2.2.3 sur les méthodes de Newton).

Plus précisément, posons :

$$\phi(x) = \sum_{i=1}^{m} \phi_{g_i}(x), \tag{2.111}$$

où ϕ_{g_i} est la barrière logarithmique définie en (2.97). On introduit alors, pour $t > 0$, une approximation du problème d'optimisation (2.13) :

$$\min_{Ax=b} f(x) + \frac{1}{t}\phi(x). \tag{2.112}$$

On note :

$$\tilde{f}_t(x) = f(x) + \frac{1}{t}\phi(x), \tag{2.113}$$

qui est une fonction convexe. Lorsque t est grand, on s'attend à ce que les problèmes (2.13) et (2.112) soient proches. En fait, on peut même montrer que si on note $x^*(t)$ une solution du problème (2.112), alors on a (voir [54] section 11.2.2) :

$$f(x^*(t)) - f(x^*) \leq \frac{m}{t}. \tag{2.114}$$

On peut donc très facilement choisir t pour atteindre une précision souhaitée sur la solution du problème d'origine (2.13). Les conditions de KKT pour (2.112) s'écrivent :

$$\nabla \tilde{f}_t(x^*(t)) + A^T \nu^*(t) = 0, \tag{2.115a}$$

$$Ax^*(t) = b. \tag{2.115b}$$

On peut exprimer le gradient de $\tilde{f}_t$:

$$\nabla \tilde{f}_t(x) = \nabla f(x) + \sum_{i=1}^{m} \frac{-1}{t f_i(x)} \nabla f_i(x). \tag{2.116}$$

On peut de manière similaire donner sa hessienne et en déduire une méthode de Newton pour résoudre (2.115). Le problème est que plus t est grand, plus la méthode de Newton est mal conditionnée ; il devient donc difficile d'obtenir une solution précise.

Remarque 35 *En fait, c'est exactement ce que proposaient Fiacco et McCormick en 1968 [92] et c'est suite aux problèmes de conditionnement que la méthode a été temporairement abandonnée jusqu'à son retour avec Karmarkar.*

Pour pallier cela, on utilise une méthode de *chemin* (*central path*), dont le principe est de faire varier t et de suivre la solution de (2.112) $x^*(t)$. À chaque étape, on va résoudre (2.112) (étape de *centrage*) via une méthode de Newton sur les conditions de KKT (2.115), puis on va incrémenter $t^{(k)}$ jusqu'à ce que l'estimation (2.114) nous assure que la précision souhaitée est atteinte. L'idée est qu'en initialisant la méthode de Newton avec le résultat de l'itération précédente, si l'on

n'augmente pas $t^{(k)}$ trop rapidement, on ne fait que peu d'itérations de la méthode de Newton (itérations *internes*).

En plus de l'analyse de l'algorithme qui en résulte, Boyd et Vandenberghe [54] donnent différentes interprétations de l'algorithme qui en résulte, on va ici s'intéresser à celle qui donne la base de la méthode primale-duale : la résolution d'un système de KKT modifié. En effet, notons :

$$\lambda_i(t) = \frac{-1}{tg_i(x)}. \tag{2.117}$$

On peut alors réécrire le système de KKT (2.115) du problème approché comme :

$$\nabla f(x^*(t)) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x^*(t)) + A^T \nu^*(t) = 0, \tag{2.118a}$$

$$\lambda_i \geq 0, \qquad \forall i \in [1, m], \tag{2.118b}$$

$$g_i(x^*(t)) \leq 0, \qquad \forall i \in [1, m], \tag{2.118c}$$

$$-\lambda_i g_i(x^*(t)) = \frac{1}{t}, \qquad \forall i \in [1, m], \tag{2.118d}$$

$$Ax^*(t) = b. \tag{2.118e}$$

On constate alors que la seule équation qui diffère entre ce nouveau système et le système de KKT (2.27) d'origine est l'équation (2.118d) qui diffère de (2.27d) du terme $-1/t$. On peut donc voir la méthode comme une relaxation de la condition de KKT (2.27d) que l'on réduit au cours des itérations pour arriver à 0.

2.4.2 Méthode primale-duale

On peut aller loin dans la direction qui vient d'être abordée. Dans la section 11.3.4, Boyd et Vandenberghe [54] écrivent une méthode de Newton directement sur le système modifié (2.118) (on ignore les inéquations pour le moment), dont les inconnues sont alors x, ν ET λ. Forcément l'équation (2.118d) redonne la formule des λ_i, aussi on peut se ramener aux équations (2.115). Une autre possibilité est de ne pas expliciter les λ_i, i.e. d'écrire la méthode de Newton directement sur le système :

$$r_{dual}(x, \lambda, \nu) = \nabla f(x) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x) + A^T \nu = 0, \tag{2.119a}$$

$$r_{cent}(x, \lambda, \nu) = -\mathbf{diag}(\lambda) g(x) - \frac{1}{t}\mathbf{1} = 0, \tag{2.119b}$$

$$r_{primal}(x, \lambda, \nu) = Ax - b = 0 \tag{2.119c}$$

où $\mathbf{diag}(\lambda)$ est la matrice diagonale dont la diagonale est λ et $\mathbf{1}$ est le vecteur de $\mathbb{R}^m$ composé uniquement de 1. Si on note J_f la jacobienne de f, on peut alors écrire la direction de recherche de la méthode de Newton appliquée à la résolution de (2.119) comme la solution de :

$$\begin{pmatrix} \nabla^2 f(x^{(k)}) + \sum_i \lambda_i \nabla^2 g_i(x^{(k)}) & J_f(x^{(k)})^T & A^T \\ -\mathbf{diag}\left(\lambda^{(k)}\right) J_f(x^{(k)}) & -\mathbf{diag}\left(f(x^{(k)})\right) & 0 \\ A & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta \lambda \\ \Delta \nu \end{pmatrix} = - \begin{pmatrix} r_{dual}\left(x^{(k)}, \lambda^{(k)}, \nu^{(k)}\right) \\ r_{cent}\left(x^{(k)}, \lambda^{(k)}, \nu^{(k)}\right) \\ r_{primal}\left(x^{(k)}, \lambda^{(k)}, \nu^{(k)}\right) \end{pmatrix}. \tag{2.120}$$

Cette direction étant calculée, il faut s'assurer que $\lambda^{(k)} \succeq 0$ et $g(x^{(k)}) \preceq 0$. On va en fait imposer que ces inégalités soient vérifiées de manière stricte à chaque itération. On suppose donc $g(x^{(0)}) \prec 0$ et $\lambda^{(0)} \succ 0$ et utilise ensuite une line search par backtracking sur la méthode de

Newton pour imposer ces conditions. Autrement dit, si on pose $z^{(k)} = (x^{(k)}, \lambda^{(k)}, \nu^{(k)})$, on utilise une line search de type Algo. 1 pour choisir $s^{(k)}$ dans laquelle au lieu des conditions d'Armijo, on cherche à satisfaire :

$$g(x^{(k+1)}) \prec 0, \tag{2.121a}$$
$$\lambda^{(k+1)} \succ 0. \tag{2.121b}$$

On pose ensuite $z^{(k+1)} = z^{(k)} + s^{(k)}\Delta z$ (où Δz est la solution de (2.120)). La dernière modification est que typiquement, les méthodes de point intérieur primales-duales n'effectuent qu'une seule étape de Newton avant d'incrémenter t. On obtient alors l'algorithme 10. Il y a plusieurs points à éclaircir pour obtenir un algorithme que l'on peut implémenter en pratique. Comme les conditions de KKT modifiées (2.118) ne sont pas résolues de manière exacte, on ne peut plus utiliser (2.114) pour estimer la qualité de la solution $x^{(k)}$. On va donc typiquement utiliser un seuil sur les résidus r_{dual}, r_{cent} et r_{primal} comme critère d'arrêt. Jusqu'à présent, aucune règle n'a été donnée pour incrémenter t. Il existe aussi différents raffinements permettant d'améliorer la stabilité numérique de la procédure. On s'en tient pour le moment à la version générale exposée et on renvoie aux références données en début de section pour des discussions détaillées sur ces points. On revient sur ces points en section 3.4 où on donne un algorithme complet utilisé dans le cadre des problèmes viscoplastiques.

Algorithme 10 Méthode de point intérieur primale-duale pour (2.13).

Requis : $x^{(0)}$ tel que $g(x^{(0)}) \prec 0$, $\lambda^{(0)} \succ 0$, $\nu^{(0)}$, t.
1: **Pour** $k \geq 0$ **faire**
2: Calculer la direction Δz en résolvant (2.120).
3: Choisir le pas $s^{(k)}$ par backtracking line search pour satisfaire (2.121).
4: Mettre à jour $z^{(k+1)} = z^{(k)} + s^{(k)}\Delta z$.
5: Incrémenter t.
6: **Fin Pour**
7: **Retourner** $z^{(k)}$

2.5 Algorithmes proximaux

On termine ce chapitre avec les algorithmes basés sur l'opérateur proximal (voir (2.41)), dont le principe de base est celui d'un algorithme de point fixe : appliquer itérativement l'opérateur proximal jusqu'à converger vers un point fixe de l'opérateur, ce qui correspond alors à un minimiseur de f (voir (2.47)). Ces méthodes trouvent leur intérêt dans l'optimisation non-lisse, i.e. dans le cas où f n'est pas différentiable. De ce fait, on pourra sans perte de généralité traiter de minimisation essentiellement sans contrainte en intégrant les contraintes dans la fonctionnelle via des fonctions indicatrices (voir équation (2.5)) selon le modèle de la remarque 19. On cherche donc dans cette section à résoudre le problème de minimisation (2.56).

On commence par brièvement discuter de l'*algorithme du point proximal* (PPA) avant de détailler des méthodes qui en découlent et qui sont celles utilisées en pratique. À nouveau l'objectif ici n'est pas d'être exhaustif mais de présenter les méthodes qui sont utilisées dans les chapitres suivants, à savoir ADMM et FISTA. Pour plus de détails, on renvoie à Boyd et Parikh (2014) [193], Chambolle et Pock (2016) [65], Beck (2017) [33] et Condat et al. (2023) [68]. Cette présentation se base sur celles de [193] et [65].

2.5.1 PPA

Les premiers travaux dans la direction du PPA (voir Algo. 11) remontent à Martinet dans les années 1970 [160],[161] puis sont concrétisés dans la forme actuelle par Rockafellar en 1976 [211]. On renvoie à Lemaire [146] pour une review couvrant les origines du PPA. En termes de convergence, choisir $t^{(k)} > 0$ tels que $\sum_k t^{(k)} = \infty$ (par exemple $t^{(k)} = t > 0$) suffit à garantir la convergence de l'algorithme 11.

Algorithme 11 Algorithme du point proximal (PPA) pour (2.56).

Requis : $x^{(0)}$, $\left(t^{(k)}\right)_{k\in\mathbb{N}}$.

1: **Pour** $k \geq 0$ **faire**

2:
$$x^{(k+1)} = \text{prox}_{t^{(k)}f}(x^{(k)}) \tag{2.122}$$

3: **Fin Pour**

4: **Retourner** $x^{(k)}$

Une première interprétation vient de la régularisée de Moreau-Yosida définie en (2.42). Par la formule (2.43), on peut réécrire l'étape (2.122) du PPA sous la forme :

$$x^{(k+1)} = x^{(k)} - t^{(k)}\nabla M_{t^{(k)}f}(x^{(k)}), \tag{2.123}$$

que l'on peut donc voir comme une descente de gradient sur $M_{t^{(k)}f}$ avec le pas $t^{(k)}$. La fonction $M_{t^{(k)}f}$ est elle-même une forme de régularisation de la fonction f, dont une propriété majeure est de posséder les mêmes minimiseurs que f.

Une autre interprétation est celle d'une descente de gradient *implicite* sur la fonction f. En effet, en utilisant (2.45) et en supposant que f est différentiable, on peut en déduire que si $x^{(k+1)}$ est donné par l'étape (2.122) du PPA, alors :

$$x^{(k)} - x^{(k+1)} = t^{(k)}\nabla f(x^{(k+1)}), \tag{2.124}$$

i.e. :

$$x^{(k+1)} = x^{(k)} - t^{(k)}\nabla f(x^{(k+1)}). \tag{2.125}$$

On peut interpréter (2.125) comme une descente de gradient implicite au sens où on prend le gradient du point à venir plutôt que du point courant $x^{(k)}$. On peut même aller plus loin en considérant le *flot de gradient* de f donné par l'équation différentielle :

$$\frac{d}{dt}x(t) = -\nabla f(x(t)). \tag{2.126}$$

Les points d'équilibre de (2.126) sont alors exactement les zéros de ∇f et donc les minimiseurs de f. Chaque trajectoire suivant (2.126) doit donc converger vers un minimiseur de f. On peut alors chercher à discrétiser (2.126) par un simple schéma d'Euler. Un schéma explicite donne la classique descente de gradient 3, alors qu'un schéma implicite donne l'équation (2.125) et donc le PPA. Cela fournit une intuition expliquant les faibles hypothèses nécessaires sur les pas $t^{(k)}$ pour garantir la convergence du PPA : utiliser un schéma d'Euler implicite plutôt qu'explicite permet d'obtenir un schéma stable, au prix de la nécessité de résoudre un sous-problème pour mettre à jour la variable. On peut voir le même compromis pour le PPA : l'évaluation de $\text{prox}_{t^{(k)}f}(x^{(k)})$ demande la résolution d'un problème d'optimisation, mais en contrepartie on obtient un schéma dont la convergence est garantie, sous des hypothèses minimales sur f (convexe, l.s.c. et propre). Pour plus d'interprétations, on renvoie à Boyd et Parikh [193].

2.5.2 Méthodes de splitting

Le PPA 11 en tant que tel n'est que peu utilisé. Il est cependant le fondement d'une grande partie des algorithmes de minimisation convexe non-lisse. La plupart d'entre eux est basée sur l'idée de *splitting*, c'est-à-dire d'écrire f comme la somme de deux fonctions afin de (partiellement) séparer le traitement de chacune des deux fonctions. Un exemple de tel algorithme est l'algorithme d'Uzawa 9 présenté précédemment. Ces idées peuvent aussi être appliquées aux méthodes proximales, ce qui a permis de donner naissance à une quantité importante d'algorithmes au cours des dernières décennies. Illustrons cette idée à travers l'algorithme de Chambolle-Pock [64] qui est devenu extrêmement populaire depuis son apparition en 2011, notamment dans le domaine de l'imagerie. Supposons que, comme pour l'algorithme d'Uzawa, on veuille minimiser la somme (2.106). Cela revient à trouver un point selle du Lagrangien (2.107). En utilisant la conjuguée de Fenchel (2.29), on peut reformuler le problème sous la forme suivante :

$$\min_x \max_y \langle Dx, y\rangle + f_1(x) - f_0^*(y), \tag{2.127}$$

où f_0^* est la conjuguée de f_0. L'idée directrice est d'appliquer une étape de PPA en y puis une autre en x afin de découpler les minimisations. Précisément, les auteurs proposent l'algorithme 12. L'étape de relaxation (2.130) n'est pas à confondre avec les méthodes d'accélération du type

Algorithme 12 Algorithme de Chambolle-Pock pour (2.127).

Requis : $x^{(0)}$, $y^{(0)}$, $\tau > 0$, $\sigma > 0$, $\theta \in [0, 1]$.

1: $\bar{x}^{(0)} = x^{(0)}$.

2: **Pour** $k \geq 0$ **faire**

3:
$$y^{(k+1)} = \mathrm{prox}_{\sigma f_0^*}(y^{(k)} + \sigma D\bar{x}^{(k)}) \tag{2.128}$$

4:
$$x^{(k+1)} = \mathrm{prox}_{\tau f_1}(x^{(k)} - \tau D^* y^{(k+1)}) \tag{2.129}$$

5:
$$\bar{x}^{(k+1)} = x^{(k+1)} + \theta\left(x^{(k+1)} - x^{(k)}\right) \tag{2.130}$$

6: **Fin Pour**

7: **Retourner** $x^{(k)}, y^{(k)}$

proposé par Nesterov pour le gradient accéléré 5. Cette possibilité est explorée dès la parution de l'algorithme, dans le même papier. Chambolle et Pock prouvent aussi la convergence de l'algorithme 12 sous différentes conditions sur τ, σ et θ. On ne va pas ici s'appesantir sur ces résultats, on retiendra seulement que l'algorithme se montre en pratique très efficace pour de nombreux problèmes, d'où son succès.

La forme de l'algorithme 12 soulève déjà de nombreuses possibilités pour créer des variantes : on pourrait remplacer la relaxation en x (2.130) par une relaxation en y. On pourrait aussi faire les deux, ou encore discuter du placement de ces relaxations par rapport aux étapes proximales. Qu'en est-il du cas où une des fonctions est régulière, peut-on alors utiliser une étape de gradient à la place d'une étape proximale ? Que peut-on faire s'il y a 3 fonctions ? Ou 2 opérateurs linéaires ? Ce sont toutes ces questions et d'autres qui font qu'il existe aujourd'hui de nombreux algorithmes proximaux, faisant varier légèrement différents paramètres. La très récente review de Condat et al. [68] tente de donner un cadre unificateur à nombre de ces algorithmes, permettant d'étendre des critères de convergence et de donner des versions plus générales de certains algorithmes. On renvoie donc à ce papier pour une présentation détaillée des algorithmes majeurs entrant dans ce cadre et on concentre les deux dernières sections de ce chapitre sur deux méthodes qui sont mises à l'étude dans ce travail dans le cadre des écoulements viscoplastiques.

2.5.3 ADMM

Le premier algorithme que l'on mentionne est l'Algorithme 13, dit ADMM (*Alternating direction method of multipliers*), que l'on trouve parfois sous le nom *splitting de Douglas-Rachford*. On cherche, comme dans l'algorithme de Chambolle-Pock, à résoudre le problème de minimisation (2.106). Historiquement, les premières versions d'ADMM remontent à Douglas et Rachford dans les années 1950 [85] puis à Gabay et Mercier [105] et Glowinski et Marrocco [114] dans les années 1970 pour sa forme actuelle. En fait, ADMM a été redécouvert plusieurs fois par différentes communautés à partir de diverses formulations, on renvoie à la section 3.5 de Boyd et al. [53] et aux références qu'il contient pour un historique plus détaillé.

Algorithme 13 Algorithme ADMM pour (2.106).

Requis : $x^{(0)}$, $y^{(0)}$, $\lambda^{(0)}$, $r > 0$, $t > 0$.

1: **Pour** $k \geq 0$ **faire**

2:
$$x^{(k+1)} = \underset{x}{\operatorname{argmin}}\ L_r(x, y^{(k)}, \lambda^{(k)}) \tag{2.131}$$
▷ L_r est le Lagrangien augmenté défini par (2.110)

3:
$$y^{(k+1)} = \underset{y}{\operatorname{argmin}}\ L_r(x^{(k+1)}, y, \lambda^{(k)}) \tag{2.132}$$

4:
$$\lambda^{(k+1)} = \lambda^{(k)} + t\left(Ax^{(k+1)} - y^{(k+1)}\right) \tag{2.133}$$

5: **Fin Pour**

6: **Retourner** $x^{(k)}, y^{(k)}, \lambda^{(k)}$

Pour donner ADMM sous sa forme 13, le principe est similaire à celui de l'algorithme d'Uzawa 9. En fait, là où dans l'algorithme d'Uzawa, on profite de la séparabilité des variables x et y dans le cas du Lagrangien classique ($r = 0$) pour séparer les minimisations en x et y dans la mise à jour primale (2.108), ADMM 13 utilise le Lagrangien augmenté ($r > 0$) et donc il faudrait effectuer une minimisation jointe en x et y. Plutôt que de faire cela, on choisit de quand même séparer les problèmes de minimisations afin d'obtenir des problèmes potentiellement plus faciles à résoudre. On trouvera dans [53] une preuve de la convergence de ADMM (convergence de $Dx^{(k)} - y^{(k)}$ vers 0, convergence de $f_0(y^{(k)}) + f_1(x^{(k)})$ vers le minimum p^* et convergence de la variable duale $\lambda^{(k)}$ vers λ^*) sous des hypothèses très générales. On note aussi qu'on se limite ici à la contrainte $y = Dx$ dans la mesure où elle suffit à comprendre l'algorithme (et c'est ce qu'on utilise dans le chapitre 3), cependant [53] donne le cas plus général d'une contrainte du type $Ax + By = c$, aussi les rôles de x et y peuvent être échangés et on peut donc inverser (2.131) et (2.132) sans problème.

Il n'y a pour l'instant aucun lien avec l'algorithme du PPA 11 présenté avant. Cependant, on peut reformuler la mise à jour de y (2.132) sous la forme :

$$y^{(k+1)} = \operatorname{prox}_{f_0/r}\left(Dx^{(k+1)} + \frac{\lambda^{(k)}}{r}\right), \tag{2.134}$$

et dans le cas où $D = Id$, on peut reformuler la mise à jour de x (2.131) sous la forme :

$$x^{(k+1)} = \operatorname{prox}_{f_1/r}\left(y^{(k)} - \frac{\lambda^{(k)}}{r}\right), \tag{2.135}$$

(voir [193] section 4.4.1). Cela permet donc de voir ADMM comme un algorithme proximal issu d'une méthode de splitting. Comme pour le PPA, on peut alors donner diverses interprétations, on renvoie pour cela à la section 4.4 de [193] et au papier [53] dédié à ADMM.

Le choix du paramètre r est un problème ouvert qui n'a pas de réponse établie. Récemment, Bartels et Milicevic [28] ont même proposé de faire varier ce paramètre au cours des itérations. Une autre manière de raffiner la méthode se trouve dans les sur-relaxations comme présenté par Nesterov [179] pour la méthode de gradient accéléré 5. On donne plus de détails dans la section qui suit dans le cadre de la méthode FISTA et on renvoie à Goldstein et al. [118] pour le cas ADMM. On note aussi la récente contribution de Sabach et Teboulle [217] qui étend cette idée de manière plus générique à de nombreuses méthodes qui peuvent s'exprimer comme des méthodes de Lagrangien.

2.5.4 Méthodes de type FISTA

Dans cette section, on s'intéresse à la minimisation de :

$$\min_{x\in\mathcal{X}} f_0(x) + f_1(x), \tag{2.136}$$

où on suppose que f_0 est L-régulière (voir définition 12). Pour traiter différents cas, on suppose que f_i $(i = 0, 1)$ est μ_i-fortement convexe, avec $\mu_i \geq 0$ (le cas $\mu_i = 0$ correspond à la convexité simple). Avec ces notations, $f = f_0 + f_1$ est $(\mu = \mu_0 + \mu_1)$-fortement convexe (comme précédemment les fonctions sont propres et l.s.c.).

2.5.4.1 FISTA

Il existe de nombreuses variantes de l'algorithme FISTA introduit par Beck et Teboulle en 2009 [35]. On en donne une formulation générale dans l'algorithme 14.

Algorithme 14 Une formulation générale des algorithmes de type (F)ISTA pour (2.136).

Requis : L, $x^{(0)}$, ψ, $\tau \leq 1/L$

1: $y^{(0)} = x_0$, $t^{(0)} = 1$

2: **Pour** $k \geq 0$ **faire**

3:
$$x^{(k+1)} = \text{prox}_{\tau f_1}\left(y^{(k)} - \tau\nabla f_0(y^{(k)})\right). \tag{2.137}$$

4:
$$t^{(k+1)} = \psi(t^{(k)}, k), \tag{2.138}$$

▷ La mise à jour la plus générale pour $t^{(k)}$

$$\theta^{(k+1)} = \frac{t^{(k)} - 1}{t^{(k+1)}}\frac{1 + \tau\mu_1 - t^{(k+1)}\tau\mu}{1 - \tau\mu_0}. \tag{2.139}$$

▷ Le paramètre de sur-relaxation

5:
$$y^{(k+1)} = x^{(k+1)} + \theta^{(k+1)}(x^{(k+1)} - x^{(k)}). \tag{2.140}$$

6: **Fin Pour**

On commence par présenter l'algorithme *Iterative Shrinkage-Threshold Algorithm* (ISTA) ou *Forward-Backward* (FB) introduit initialement par Bruck en 1975 [55]. Dans l'algorithme 14, cela correspond à prendre $\psi \equiv 0$ et $\mu = 0$, de sorte à obtenir la simple mise à jour :

$$x^{(k+1)} = \text{prox}_{\tau f_1}\left(x^{(k)} - \tau\nabla f_0(x^{(k)})\right), \tag{2.141}$$

(i.e. la variable $y^{(k)}$ n'est plus utilisée). L'idée est d'utiliser, comme dans l'algorithme de Chambolle-Pock, une étape proximale pour traiter le terme f_1 non lisse, puis cette fois d'utiliser une étape

de gradient classique pour le terme régulier f_0 (on trouvera d'autres interprétations dans les sections 4.2 et 4.3 de [193]). Bien que convergent, cet algorithme a l'inconvénient d'être lent (on pourra consulter Beck et Teboulle [35] ou encore Combettes et Pesquet [32]). C'est pour pallier ce problème qu'en 2009, Beck et Teboulle [35] proposent d'incorporer l'étape de sur-relaxation (2.140) en suivant celle proposée par Nesterov en 1983 [179] pour le gradient accéléré (Algo. 5, voir formule (2.77)). L'algorithme obtenu est significativement plus rapide et est alors baptisé FISTA, où le F est rajouté pour *Fast*. Plus précisément, Beck et Teboulle donnent le théorème :

Théorème 15 *Si $x^{(k)}$ est donné par l'algorithme 14 en utilisant le schéma d'accélération (2.77) et $\mu = 0$ dans (2.139), alors :*

$$f(x^{(k)}) - p^* \leq \frac{2}{\tau} \frac{\|x^{(0)} - x^*\|^2}{(k+2)^2}. \tag{2.142}$$

Ces résultats seront rapidement étendus pour des variations de FISTA. En 2015, Chambolle et Dossal [63] montrent qu'on peut conserver la vitesse de convergence même sans utiliser la formule de Nesterov tant que la suite $t^{(k)}$ vérifie :

$$\forall k \geq 0, (t^{(k+1)})^2 - t^{(k+1)} \leq (t^{(k)})^2. \tag{2.143}$$

Ce faisant, ils proposent un autre schéma d'accélération :

$$t^{(k)} = \frac{k+a}{a}, \tag{2.144}$$

où $a \geq 2$. De plus, ils montrent qu'utiliser ce schéma garantit la convergence de $x^{(k)}$ vers x^*. Le sujet de la vitesse exacte de convergence de FISTA selon les schémas utilisés a généré une littérature conséquente, notamment dans la prise en compte d'erreur dans l'évaluation de l'étape proximale. On peut en effet montrer que sous certaines hypothèses de décroissance de l'erreur dans l'étape proximale, on conserve le taux de convergence donné dans le théorème 15. On pourra consulter Chambolle et Dossal (2015) [63], Chambolle et Pock (2016) [65], Attouch et Peypouquet (2016) [11], Villa et al. (2013) [245] ou encore Rasch et Chambolle (2020) [208].

La formule de $\theta^{(k+1)}$ (2.139) telle qu'on la présente ici a été introduite par Chambolle et Pock [65] qui proposent des schémas d'accélération pour prendre en compte la forte convexité. Ils proposent d'utiliser la formule suivante pour ψ :

$$t^{(k+1)} = \frac{1 - q(t^{(k)})^2 + \sqrt{(1 - q(t^{(k)})^2)^2 + 4(t^{(k)})^2}}{2}, \tag{2.145}$$

où :

$$q = \frac{\tau\mu}{1 + \tau\mu_1}. \tag{2.146}$$

Ils montrent alors que l'on peut obtenir une vitesse de convergence en $\mathcal{O}((1-\sqrt{q})^k)$. On trouvera d'autres propositions pour des taux de convergence similaires dans différents cas dans le papier de Liang et al. (2022) [151].

On cite pour terminer des travaux sur la convergence de FISTA lorsque l'on se trouve à proximité de la solution. En effet, différents travaux ont pu identifier des changements de régime de convergence au voisinage de la solution (voir Liang et al. (2017) [150]), une conséquence étant que le schéma ISTA non accéléré serait, à partir d'un certain rang, plus rapide que la version accélérée FISTA. Dans une approche différente, Bareilles et al. [23], [24] montrent que dans certains cas FISTA peut permettre d'entrer dans une zone de l'espace où les fonctions sont plus régulières ce qui permet d'utiliser des méthodes de Newton pour accélérer la convergence de l'algorithme.

2.5.4.2 Variantes de FISTA

Malgré sa rapidité, un reproche qui peut être fait à FISTA est que l'introduction de l'étape d'accélération (2.140) fait perdre à ISTA sa monotonie. En pratique, on peut observer de fortes oscillations dans les valeurs de f entre plusieurs itérations de FISTA. Pour pallier cela, Beck et Teboulle [34] ont proposé, en même temps que FISTA, une version monotone de l'algorithme, baptisée MFISTA (cf. Algo. 15), dont le principe est de conserver $x^{(k)}$ si l'étape proximale ne permet pas de réduire la valeur de la fonction objectif. De manière surprenante, la vitesse

Algorithme 15 MFISTA pour (2.136).

Requis : L, $x^{(0)}$

1: $y^{(0)} = x_0$, $t^{(0)} = 1$

2: **Pour** $k \geq 0$ **faire**

3:
$$z^{(k+1)} = \text{prox}_{f_1/L}\left(y^{(k)} - \frac{1}{L}\nabla f_0(y^{(k)})\right). \tag{2.147}$$

4:
$$t^{(k+1)} = \frac{1+\sqrt{1+4(t^{(k)})^2}}{2}. \tag{2.148}$$

5:
$$x_{k+1} = \text{argmin}\,\{f(x)/x = z_{k+1} \text{ ou } x = x_k\}, \tag{2.149}$$

▷ assure la monotonie

$$y^{(k+1)} = x^{(k+1)} + \left(\frac{t^{(k)}}{t^{(k+1)}}\right)(z^{(k+1)} - x^{(k+1)}) + \left(\frac{t^{(k)}-1}{t^{(k+1)}}\right)(x^{(k+1)} - x^{(k)}). \tag{2.150}$$

6: **Fin Pour**

de convergence n'est pas affectée par cette modification, au moins théoriquement. Cependant, en plus de stocker une nouvelle variable, cet algorithme requiert deux évaluations de f par itération, ce qui peut être prohibitif dans certains cas. De fait, cet algorithme semble avoir reçu beaucoup moins de succès que FISTA, on note toutefois les travaux de Zibetti et al. [260] et Florea et Vorobyov [95] qui proposent différentes améliorations de MFISTA dans l'esprit de celles proposées pour FISTA.

Pour finir, on mentionne une variante de FISTA qui est utilisée dans ce travail. Le principe est simplement d'appliquer FISTA au problème dual. Reprenons le problème de minimisation (2.39), dont on peut écrire le dual sous la forme :

$$\min_z f_0^*(z) + f_1^*(-D^T z). \tag{2.151}$$

En utilisant les correspondances entre L-régularité et forte convexité du dual données par le théorème 9, on peut ôter l'hypothèse de L-régularité de f_0 et simplement demander sa convexité forte (i.e. $\mu_0 > 0$) pour pouvoir appliquer FISTA au problème (2.151) (pour être exact, il faut demander à D d'être bornée par en dessous sur le domaine de f_1). Sans traitement supplémentaire, on obtient cependant des calculs seulement sur la variable duale du problème. Cependant en 2016, Treskatis et al. [239, 237] montrent que l'on peut réexprimer les différentes étapes de l'algorithme FISTA qui en résulte, afin de récupérer les variables primales, ce qui permet d'obtenir l'algorithme 16, que l'on appelle donc *Dual FISTA*. L'algorithme que l'on obtient requiert la résolution de plusieurs problèmes d'optimisation, de difficulté similaire à celle de l'évaluation de l'opérateur proximal. Treskatis et al. [239] prouvent alors la convergence des itérées qui découle directement des résultats de FISTA. On peut donc utiliser les mêmes raffinements sur le schéma

Algorithme 16 Dual FISTA for (2.39).

Requis : $\tau \leq \mu_f$, $z^{(0)}$

1: $w^{(0)} = z^{(0)}$, $t^{(0)} = 1$

2: **Pour** $k \geq 0$ **faire**

3:

$$y^{(k+1)} = \underset{y}{\operatorname{argmax}} \{\langle y, w^{(k)} \rangle - f_0(y)\} = f_0^*(w^{(k)}), \tag{2.152}$$

▷ variable auxiliaire

$$x^{(k+1)} = \underset{x}{\operatorname{argmin}} \ f_1(x) + \frac{\tau}{2} \left\| Dx - \left(y^{(k+1)} - \frac{w^{(k)}}{\tau} \right) \right\|^2, \tag{2.153}$$

▷ variable auxiliaire

$$z^{(k+1)} = w^{(k)} - \tau(y^{(k+1)} - Dx^{(k+1)}) \tag{2.154}$$

4:

$$t^{(k+1)} = \frac{1 + \sqrt{1 + 4(t^{(k)})^2}}{2}. \tag{2.155}$$

5:

$$w^{(k+1)} = z^{(k+1)} + \left(\frac{t^{(k)} - 1}{t^{(k+1)}} \right) (z^{(k+1)} - z^{(k)}). \tag{2.156}$$

6: **Fin Pour**

7: **Retourner** $z^{(k)}$

d'accélération que pour FISTA, ou encore assurer la monotonie de l'algorithme. La question est maintenant de savoir comment récupérer la variable primale correspondant à la variable duale $z^{(k)}$. Sans le schéma d'accélération, on pourrait prendre simplement la variable $x^{(k)}$ définie par (2.153). Il y a principalement 3 choix possibles :

1.
$$\bar{x} = \underset{x}{\operatorname{argmin}} \ f_1(x) + \frac{\tau}{2} \left\| Dx - \left(f_0^*(z^{(k)}) - \frac{z^{(k)}}{\tau} \right) \right\|^2, \tag{2.157}$$

2.
$$\bar{x} = \underset{x \in dom \ f_1}{\operatorname{argmin}} \|Dx - f_0^*(z^{(k)})\|^2, \tag{2.158}$$

3.
$$\bar{x} = x^{(k)}. \tag{2.159}$$

Les deux premiers choix sont les plus rigoureux d'un point de vue théorique, mais aussi les plus coûteux. Le dernier choix a l'avantage de ne pas nécessiter de calcul supplémentaire. En principe, si l'algorithme a convergé, les points $z^{(k)}$ et $w^{(k)}$ devraient être presque égaux, donc le dernier choix devrait introduire une erreur faible. Il ne s'agit là néanmoins que d'un raisonnement formel. On observe que le premier choix (2.157) correspond à l'étape proximale (2.153) aussi elle n'a aucun coût supplémentaire d'implémentation et un coût en temps négligeable si l'on suppose que l'on utilise beaucoup d'itérations de FISTA.

Remarque 36 *En 2014, Beck et Teboulle [36] proposent un algorithme très similaire, en demandant la forte convexité de f_1 plutôt que celle de f_0, ce qui modifie légèrement les calculs. Le principe reste toutefois le même.*

Annexe du chapitre 2

2.A Compléments sur la convexité

On donne quelques opérations préservant la convexité, ce qui permet en pratique de vérifier la convexité d'une fonction uniquement en l'écrivant à partir d'une fonction convexe connue.

Propriété 19 (Combinaison linéaire à coefficients positifs) *Si $f_1, ..., f_n$ sont convexes et $\alpha_1, ..., \alpha_n$ sont positifs, alors $\alpha_1 f_1 + \cdots + \alpha_n f_n$ est convexe.*

Propriété 20 (Composition avec une application affine) *Si A est une application linéaire de $\mathcal{Y}$ dans $\mathcal{X}$, $b \in \mathbb{Y}$ et $f : \mathcal{X} \to \overline{\mathbb{R}}$ est convexe, alors la fonction $g : \mathcal{Y} \to \overline{\mathbb{R}}$ définie par $g(x) = f(Ax + b)$ est convexe.*

Propriété 21 (Maximum de fonctions) *Si $f_1, ..., f_n$ sont convexes, alors la fonction définie par $g(x) = \max(f_1(x), ..., f_n(x))$ est convexe.*

Propriété 22 (Minimum d'une fonction) *Soit $f : \mathcal{X} \times \mathcal{Y} \to \overline{\mathbb{R}}$ convexe, et soit $C \subset \mathcal{Y}$ un ensemble convexe non-vide. On pose $g : \mathcal{X} \to \overline{\mathbb{R}}$:*

$$g(x) = \inf_{y \in C} f(x, y). \tag{2.160}$$

On suppose que $g(x) > -\infty$ pour tout x. Alors g est convexe.

Propriété 23 (Perspective d'une fonction) *Soit f convexe. On pose $g : \mathcal{X} \times \mathbb{R}_+^* :$*

$$g(x, t) = tf\left(\frac{x}{t}\right). \tag{2.161}$$

g est convexe.

2.B Compléments sur la conjuguée de Fenchel

On donne ici quelques règles de base de calcul de conjuguées. On retrouvera ces règles dans la section 4.3 de [33].

Propriété 24 (Conjuguée de fonctions séparables)
Supposons que $f : \mathcal{X}_1 \times \cdots \mathcal{X}_n \to \overline{\mathbb{R}}$ s'écrive sous la forme :

$$f(x_1, ..., x_n) = \sum_{i=1}^{n} f(x_i), \tag{2.162}$$

où les f_i sont propres. Alors :

$$f^*(y_1, ..., y_n) = \sum_{i=1}^{n} f_i^*(y_i). \tag{2.163}$$

Propriété 25 *Soit $f : \mathcal{X} \to \overline{\mathbb{R}}$, A une application linéaire inversible de $\mathcal{Y}$ dans $\mathcal{X}$, $a \in \mathbb{Y}$, $b \in \mathbb{Y}^*$ et $c \in \mathbb{R}$. On pose $g : \mathcal{Y} \to \overline{\mathbb{R}}$ telle que $g(x) = f(A(x-a)) + \langle b, x\rangle + c$. Alors :*

$$g^*(y) = f^*((A^T)^{-1}(y-b)) + \langle a, y\rangle - c - \langle a, b\rangle. \tag{2.164}$$

Propriété 26 *Soient $t > 0$ et $g = tf$. Alors :*

$$g^*(y) = tf^*\left(\frac{y}{t}\right). \tag{2.165}$$

Propriété 27 *Soient $t > 0$ et g telle que $g(x) = tf(x/t)$. Alors :*

$$g^*(y) = tf^*(y). \tag{2.166}$$

2.C Compléments sur l'opérateur proximal

On donne ici quelques règles de calcul sur les opérateurs proximaux. En pratique, les calculs explicites d'opérateurs proximaux couplent les exemples analytiques connus tels que ceux de la section 2.1.6 (on en trouvera d'autres dans [33] ou sur le site proximity-operator.net) avec les règles de calcul présentées ci-dessous.

Propriété 28 (Prox de fonctions séparables)
Supposons que $f : \mathcal{X}_1 \times \cdots \mathcal{X}_n \to \overline{\mathbb{R}}$ s'écrive sous la forme :

$$f(x_1, ..., x_n) = \sum_{i=1}^{n} f(x_i). \tag{2.167}$$

Alors :

$$prox_f(x_1, ..., x_n) = prox_{f_1}(x_1) \times \cdots \times prox_{f_n}(x_n). \tag{2.168}$$

Propriété 29 (Composition avec une fonction affine)
Supposons f l.s.c., propre et convexe. Soient A une application linéaire de $\mathcal{Y}$ dans $\mathcal{X}$ telle que $A \circ A^T = \alpha Id$ pour un certain $\alpha > 0$ et $b \in \mathcal{X}$. On pose $g(x) = f(Ax + b)$. Alors :

$$prox_g(x) = x + \frac{1}{\alpha} A^T \left(prox_{\alpha f}(Ax + b) - Ax - b\right). \tag{2.169}$$

Propriété 30 (Perturbation quadratique)
Supposons f propre. On pose $g(x) = f(x) + a\|x\|^2 + \langle b, x\rangle + c$ où $a \geq 0$, $b \in \mathcal{X}$, $c \in \mathbb{R}$. Alors :

$$prox_g(x) = prox_{\frac{f}{2a+1}}\left(\frac{x-b}{2a+1}\right). \tag{2.170}$$

Propriété 31 *Soit f propre et $t > 0$. On pose $g(x) = tf(x/t)$. Alors :*

$$prox_g(x) = t\ prox_{f/t}\left(\frac{x}{t}\right). \tag{2.171}$$

Propriété 32 (Composition avec une norme)
Supposons $f : \mathbb{R} \to \overline{\mathbb{R}}$ propre, l.s.c. et convexe dont le domaine est inclus dans $[0, +\infty[$. On pose $g : \mathcal{X} \to \overline{\mathbb{R}}$ telle que $g(x) = f(\|x\|)$. Alors :

$$prox_g(x) = \begin{cases} prox_f(\|x\|)\frac{x}{\|x\|}, & si\ x \neq 0, \\ \{y \in \mathcal{X} / \|y\| = prox_f(0)\}, & sinon. \end{cases} \tag{2.172}$$

Chapitre 3

Comparaison de méthodes d'optimisation pour les simulations viscoplastiques

Ce chapitre a pour objectif de comparer différents algorithmes pour la simulation d'écoulements de fluides viscoplastiques où on utilise les équations naturelles complètes (1.20) du modèle de Herschel-Bulkley, ou plus exactement le problème de minimisation (1.30) qui leur est équivalent. On s'intéresse particulièrement à la vitesse effective des méthodes numériques ainsi qu'à la précision des détections d'interfaces entre zones cisaillées et zones rigides (voir chapitre 1). On sélectionne pour cette comparaison trois algorithmes : une méthode de Lagrangien augmenté résolu par ADMM (voir section 2.5.3), une méthode de type FISTA (voir section 2.5.4) puis une méthode de point intérieur (voir section 2.4) basée sur une reformulation en problème de type SOCP. La section 3.1 présente les algorithmes ADMM et Dual FISTA dans le cas Herschel-Bulkley, en commentant notamment la question de la discrétisation. La section 3.2 discute ensuite d'expériences numériques réalisées dans le cas Bingham sur des géométries en croix ainsi que sur des écoulements de Poiseuille. Cette étude est complétée dans la section 3.3 qui présente des résultats pour le cas Herschel-Bulkley. Enfin la section 3.4 présente la méthodologie SOCP et ses difficultés.

3.1 Présentation ADMM et Dual FISTA

On s'intéresse ici à la résolution du problème (1.20) muni des conditions de Dirichlet :

$$u = u_D \text{ sur } \partial\Omega, \tag{3.1}$$

on renvoie au chapitre 1 pour une description des notations. Pour cela, on utilise la formulation du problème comme la minimisation d'une fonctionnelle (voir (1.30)), que l'on réécrit ici :

$$\min_v H(v) = \int_\Omega \left(\frac{2^n K}{n+1} |D(v)|^{n+1} + \tau_y |D(v)| \right) - \int_\Omega \langle f, v \rangle, \tag{3.2}$$

puis pour le cas Bingham ($n = 1$) :

$$\min_v H(v) = \int_\Omega \left(\eta |D(v)|^2 + \tau_y |D(v)| \right) - \int_\Omega \langle f, v \rangle. \tag{3.3}$$

On renvoie à la section 1.2.1.2 pour les différentes formulations variationnelles que l'on peut obtenir. On présente dans cette section l'écriture des algorithmes d'étude dans ce cadre.

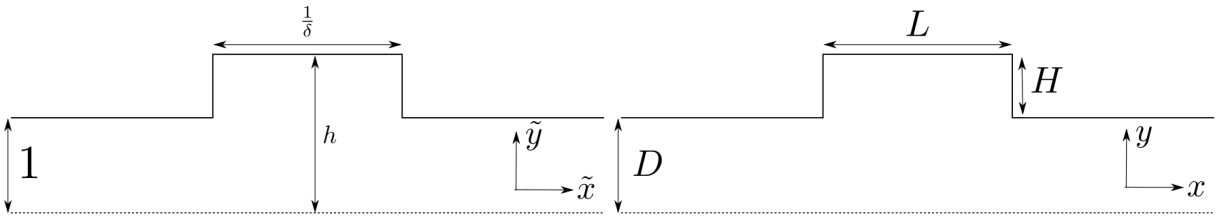


FIGURE 3.1.1 – Géométrie de l'expansion-contraction et paramètres de la géométrie sans et avec dimensions (resp. à gauche et à droite). Seule la moitié supérieure est représentée, du fait de la symétrie par rapport à l'axe des abscisses.

On pose pour la suite les espaces fonctionnels :

$$U = \{v \in H^1(\Omega)^2 / u = u_D \text{ sur } \partial\Omega\}, \tag{3.4}$$

$$U_0 = \{v \in U / \text{div}\,(v) = 0\}, \tag{3.5}$$

$$S = \{q \in L^2(\Omega)^{2\times 2}_{sym} / tr(q) = 0\}, \tag{3.6}$$

qui sont des espaces affines (espaces des vitesses et espace des tenseurs respectivement). Pour alléger les notations, on note Du au lieu de $D(u)$ (le taux de déformation du fluide, voir (1.13)). D est alors un opérateur linéaire continu de U_0 dans S.

On commence par présenter la géométrie, puis la formulation des algorithmes ADMM 13 et Dual FISTA 16 pour les écoulements viscoplastiques. On explique ensuite comment discrétiser le problème, il suffit pour le moment de préciser que l'on utilise une discrétisation de type différences finies.

3.1.1 Géométrie

Pour cette étude, on se place dans une géométrie de type *expansion-contraction*, en forme de croix, dont la moitié supérieure est exposée en figure 3.1.1. Elle est constituée d'une entrée sous la forme d'un canal rectangulaire, suivie d'une cavité (rectangulaire) plus large, qui se termine par un canal de sortie de forme identique au canal d'entrée. On utilise tantôt des variables dimensionnées, tantôt des variables sans dimension, le modèle de Bingham adimensionné a été défini dans l'équation (1.18) (on n'utilise pas de modèle adimensionné pour Herschel-Bulkley dans ce travail). Dans ce cas, l'écoulement est caractérisé par les quantités B (nombre de Bingham), h (demi-hauteur de la cavité) et δ (qui caractérise la longueur de la cavité). Avec cette géométrie, on peut se placer sur une grille cartésienne pour définir nos différences finies. On note que l'on peut voir l'écoulement de Poiseuille comme un cas particulier, en utilisant $h = 1$ (ou $H = 0$ en notation avec dimensions).

Remarque 37 *La figure 3.1.1 ne montre qu'une moitié du domaine pour simplifier, tous les calculs sont cependant effectués sur le domaine complet.*

3.1.2 ADMM

En 1976, Glowinski et al. [113] proposent d'utiliser l'algorithme ADMM 13 pour résoudre le problème de Bingham (3.3).

Remarque 38 *L'algorithme que l'on appelle ici simplement ADMM était appelé par Glowinski et al. ALG2. Ceux-ci proposaient aussi d'autres versions en changeant la décomposition* f_0, f_1

du problème. On ne présente ici que ALG2 qui est la version la plus populaire dans la littérature viscoplastique.

Le cas Bingham (3.3) et le cas Herschel-Bulkley (3.2) diffèrent légèrement aussi on présente les deux de manière distincte. On note que pour cet algorithme, on reprend intégralement le cadre étudié dans la thèse de Marly [157], on renvoie aussi à Marly et Vigneaux [158].

3.1.2.1 Cas Bingham

On pose :

$$f_0 : \begin{array}{ccc} S & \to & \mathbb{R} \\ q & \mapsto & \tau_y \int_\Omega |q|, \end{array} \tag{3.7a}$$

$$f_1 : \begin{array}{ccc} H^1(\Omega)^2 & \to & \mathbb{R} \\ u & \mapsto & \eta \int_\Omega |Du|^2 - \int_\Omega \langle f, v\rangle + \mathcal{I}_{U_0}(u), \end{array} \tag{3.7b}$$

où $\mathcal{I}_{U_0}$ est la fonction indicatrice définie par l'équation (2.5). On se retrouve avec ces notations dans le cadre du problème (2.106). Ces deux fonctions sont convexes, propres, continues et même coercives (i.e. elles tendent vers l'infini lorsque $|q|/|u|$ tend vers l'infini), on peut alors montrer que cela suffit à l'existence d'un point selle du Lagrangien associé (voir [115] p 85) ce qui suffit à la convergence de ADMM 13 (voir [53]).

L'étape 1 (2.131) de ADMM demande de résoudre :

$$\min_{v\in U_0} \int_\Omega \eta|Dv|^2 - \langle f, v\rangle + \langle \lambda, Dv\rangle + \frac{r}{2}|Dv - d|^2, \tag{3.8}$$

où λ est la notation pour la variable duale et d représente la variable y dans l'algorithme 13 (on la nomme ainsi car la contrainte qui la définit est $d = Du$). En prenant le gradient de la fonction à minimiser, on trouve que l'étape 1 de ADMM donne lieu au problème de Stokes généralisé (3.10). Ce problème ne peut pas être résolu analytiquement et nécessite d'être approché numériquement, voir section 3.1.2.2. On utilise un algorithme itératif, aussi on parle de la boucle *externe* lorsque l'on fait référence à l'algorithme 17 et on parle de boucle *interne* pour l'algorithme permettant de résoudre le système de Stokes.

L'étape 2 (2.132) de ADMM demande de résoudre :

$$\min_{q\in S} \int_\Omega \tau_y|q| - \langle \lambda, q\rangle + \frac{r}{2}|Du - q|^2. \tag{3.9}$$

On prend à nouveau le gradient de l'expression pour aboutir à l'étape 3.11. Cette fois, on obtient une expression totalement explicite que l'on peut appliquer point par point sur le tenseur $d^{(k+1)}$.

L'étape 3 (2.133) de mise à jour de la variable duale ne pose aucune difficulté, il n'y a pas de calcul supplémentaire à effectuer.

En résumé, l'algorithme ADMM 17 présente deux mises à jour explicites (3.11)-(3.12) ainsi que la résolution du problème de Stokes généralisé (3.10). C'est la résolution de ce sous-problème qui dirige la vitesse de l'algorithme en pratique.

Remarque 39 *Dans le chapitre 2, on a mentionné la possibilité d'ajouter à ADMM des schémas d'accélération dans l'idée du gradient accéléré de Nesterov 5 (voir [118],[217]). Une telle technique a été testée au cours de ce travail malheureusement sans succès : l'algorithme résultant était plus lent et donnait des résultats moins précis. On choisit de ne pas détailler cette variante mineure et de se concentrer sur d'autres algorithmes.*

Algorithme 17 Algorithme ADMM (aussi dit ALG2) pour le problème de Bingham (3.3).

1: **Pour** $k \geq 0$ **faire**

2: Résoudre le problème en $\left(u^{(k+1)}, p^{(k+1)}\right)$:

$$\begin{cases} -\text{div}\left((r+2\eta)Du^{(k+1)}\right) + \nabla p^{(k+1)} = f + \text{div}\left(\lambda^{(k)} - rd^{(k)}\right) \\ \text{div}\left(u^{(k+1)}\right) = 0 \\ u^{(k+1)} = u_D \text{ sur } \partial\Omega. \end{cases} \tag{3.10}$$

▷ utiliser l'algorithme 18

3:

$$d^{(k+1)} = \begin{cases} 0 & \text{si } |\lambda^{(k)} + rDu^{(k+1)}| \leq \tau_y, \\ \dfrac{|\lambda^{(k)} + rDu^{(k+1)}|}{r}\left(1 - \dfrac{\tau_y}{|\lambda^{(k)} + rDu^{(k+1)}|}\right) & \text{si } |\lambda^{(k)} + rDu^{(k+1)}| > \tau_y. \end{cases} \tag{3.11}$$

▷ mise à jour explicite point par point

4:

$$\lambda^{(k+1)} = \lambda^{(k)} + r\left(Du^{(k+1)} - d^{(k+1)}\right) \tag{3.12}$$

▷ mise à jour explicite point par point

5: **Fin Pour**

3.1.2.2 Sous-problème de Stokes généralisé

Le problème (3.10) est un problème linéaire en $(u^{(k+1)}, p^{(k+1)})$, dont la structure est similaire à celle des équations de Stokes, il y a donc de nombreuses méthodes de résolution possible (on en a notamment données dans le chapitre précédent). Par exemple Treskatis et al. [240],[237] proposent d'utiliser la méthode de gradient conjugué 4. On choisit ici de considérer $\text{div}\,(u) = 0$ comme une contrainte et d'utiliser à nouveau une méthode de type Lagrangien augmenté (voir section 2.3.2.4) dont on trouve le point selle par une méthode à la Uzawa (voir [111] chapitres 20-22), on obtient l'algorithme 18. Ce dernier est constitué d'un problème linéaire en u et d'une mise à jour explicite en p.

Algorithme 18 Méthode de Lagrangien augmenté pour le problème de Stokes généralisé (3.10).

1: Initialiser la pression $p^{(k+1,0)}$ à $p^{(k)}$.

2: **Pour** $i \geq 0$ **faire**

3: Résoudre le problème en $u^{(k+1,i+1)}$:

$$\begin{cases} -\text{div}\left((r+2\eta)Du^{(k+1,i+1)}\right) - s\nabla\left(\text{div}\left(u^{(k+1,i+1)}\right)\right) = f + \text{div}\left(\lambda^{(k)} - rd^{(k)}\right) - \nabla p^{(k+1,i)} \\ u^{(k+1)} = u_D \text{ sur } \partial\Omega. \end{cases} \tag{3.13}$$

4:

$$p^{(k+1,i+1)} = p^{(k+1,i)} - s\text{div}\left(u^{(k+1,i+1)}\right) \tag{3.14}$$

5: **Fin Pour**

En pratique, on résout le problème linéaire (3.13) de manière exacte via une méthode LU effectuée par la bibliothèque parallèle MUMPS [5, 4]. La parallélisation permet d'obtenir une résolution en temps raisonnable même pour des maillages fins (on renvoie à la thèse de Marly [157]). Cet algorithme nous permet de réduire de 1/3 la taille des problèmes linéaires à résoudre,

comparé à la résolution exacte de (3.10) directement. De plus, le fait d'initialiser la pression avec celle de la précédente itération de ADMM permet de rapidement réduire le nombre d'itérations (internes) nécessaires dans l'algorithme 18. Au bout d'un certain nombre d'itérations de la boucle externe, il ne faut plus qu'une ou deux itérations de la boucle interne pour atteindre la précision machine [157, 158]. Enfin, on remarque que le membre de gauche dans l'équation (3.13) ne change pas au cours des itérations, il en découle que c'est toujours la même matrice qui est impliquée dans le système linéaire. Choisir une méthode LU permet de calculer la décomposition de la matrice une seule fois en début d'algorithme puis de la stocker, ce qui permet de résoudre très rapidement les systèmes linéaires. On note tout de même que malgré ces raffinements, la résolution du système linéaire (3.13) consomme en pratique plus de 97% du temps CPU de l'algorithme ADMM complet 17.

3.1.2.3 Cas Herschel-Bulkley

On s'intéresse maintenant au cas Herschel-Bulkley, donc au problème de minimisation (3.2). Dans ce cas-là, l'exposant $n+1$ du terme visqueux nous pousse à traiter ce dernier non plus dans l'étape de Stokes mais dans la seconde étape, celle de la mise à jour de d. Cela revient donc à changer le splitting (3.7) au profit du suivant :

$$f_0 : \begin{array}{ccc} S & \to & \mathbb{R} \\ q & \mapsto & \displaystyle\int_\Omega \frac{2^n K}{n+1}|q|^{n+1} + \tau_y|q|, \end{array} \tag{3.15a}$$

$$f_1 : \begin{array}{ccc} H^1(\Omega)^2 & \to & \mathbb{R} \\ u & \mapsto & \mathcal{I}_{U_0}(u) - \displaystyle\int_\Omega \langle f, v\rangle. \end{array} \tag{3.15b}$$

L'étape de Stokes obtenue est quasiment identique, à cela près que la constante 2η disparaît, on obtient le problème (3.20). On peut donc résoudre ce problème exactement comme précédemment avec l'algorithme 18. L'étape 3 de mise à jour de λ est quant à elle totalement inchangée.

La difficulté du cas Herschel-Bulkley vis-à-vis du cas Bingham provient de la deuxième étape de ADMM, la mise à jour de d. En effet, on doit cette fois résoudre :

$$\min_{q\in S} \int_\Omega \frac{2^n K}{n+1}|q|^{n+1} + \tau_y|q| - \langle \lambda, q\rangle + \frac{r}{2}|Du - q|^2 = \min_{q\in S} g(q), \tag{3.16}$$

où on définit g comme la fonction à minimiser. Pour $q \neq 0$, g est différentiable et on a :

$$\nabla g(q) = 2^n K|q|^{n-1}q + \tau_y \frac{q}{|q|} + r(q - Du) - \lambda. \tag{3.17}$$

On peut ensuite écrire le sous-différentiel de g en 0 :

$$\partial g(0) = \{q - (\lambda + rDu)/|q| < \tau_y\}. \tag{3.18}$$

On cherche à satisfaire la condition de Fermat (2.12). Si $|\lambda + rDu| < \tau_y$, alors on obtient immédiatement que $0 \in \partial g(0)$ et donc 0 est solution du problème, sinon $0 \notin \partial g(0)$ et donc 0 n'est pas solution. Pour $q \neq 0$, on a :

$$\nabla g(q) = 0 \Leftrightarrow \underbrace{\left(\frac{\tau_y}{|q|} + \frac{2^n K}{|q|^{1-n}} + r\right)}_{>0} q = \lambda + rDu. \tag{3.19}$$

Algorithme 19 Algorithme ADMM pour le problème de Herschel-Bulkley (3.2).

1: **Pour** $k \geq 0$ **faire**

2: Résoudre le problème en $\left(u^{(k+1)}, p^{(k+1)}\right)$:

$$\begin{cases} -r\text{div}\left(Du^{(k+1)}\right) + \nabla p^{(k+1)} = f + \text{div}\left(\lambda^{(k)} - rd^{(k)}\right) \\ \text{div}\left(u^{(k+1)}\right) = 0 \\ u^{(k+1)} = u_D \text{ sur } \partial\Omega. \end{cases} \tag{3.20}$$

▷ utiliser l'algorithme 18

3: Partout où $|\lambda^{(k)} + rDu^{(k+1)}| > \tau_y$, résoudre :

$$P[\lambda^{(k)}, u^{(k+1)}](X) = 0. \tag{3.21}$$

▷ P est défini par (3.25)
▷ on utilise l'algorithme 20

4:

$$d^{(k+1)} = \begin{cases} 0 & \text{si } |\lambda^{(k)} + rDu^{(k+1)}| \leq \tau_y, \\ X\dfrac{\lambda^{(k)} + rDu^{(k+1)}}{|\lambda^{(k)} + rDu^{(k+1)}|} & \text{si } |\lambda^{(k)} + rDu^{(k+1)}| > \tau_y. \end{cases} \tag{3.22}$$

▷ mise à jour explicite point par point

5:

$$\lambda^{(k+1)} = \lambda^{(k)} + r\left(Du^{(k+1)} - d^{(k+1)}\right) \tag{3.23}$$

▷ mise à jour explicite point par point

6: **Fin Pour**

On en déduit que q et $\lambda + rDu$ sont colinéaires et que $|q|$ vérifie l'équation :

$$2^n K|q|^n + r|q| = |\lambda + rDu| - \tau_y. \tag{3.24}$$

Cette équation possède une unique solution positive si et seulement si $|\lambda + rDu| \geq \tau_y$. On pose donc :

$$P[\lambda, u](X) = 2^n K X^n + rX - |\lambda + rDu| + \tau_y, \tag{3.25}$$

et alors la mise à jour de d est donnée par les équations (3.21)-(3.22). La mise à jour de d n'est maintenant plus explicite, elle requiert la résolution de $P[\lambda, u](X) = 0$ sur une partie du domaine. Contrairement au problème de Stokes généralisé, il s'agit ici d'une équation que l'on peut résoudre point par point. Cette équation peut être résolue de manière explicite dans le cas $n = 0.5$, puisqu'on peut se ramener à une équation du second degré. On obtient alors :

$$X = \left(\frac{-\sqrt{2}K + \sqrt{2K^2 + 4r(|\lambda + rDu)| - \tau_y}}{2r}\right)^2. \tag{3.26}$$

Dans le cas général, on n'a cependant pas de solution explicite. La section suivante décrit une méthode de résolution due à Marly [157], aboutissant à l'algorithme 20. On note qu'avec les raffinements apportés à la méthode, le coût en temps de l'algorithme complet 19 reste dominé par la résolution du problème de Stokes généralisé (3.20).

3.1.2.4 Résolution de $P[\lambda, u](X) = 0$

Dans cette section, on pose pour alléger les notations :

$$z = |\lambda + rDu| - \tau_y, \tag{3.27}$$

qui dépend de l'itération considérée (on enlève les exposants (k) pour simplifier). On rappelle que l'équation $P[\lambda, u](X) = 0$ peut être résolue point par point, aussi on voit le problème comme un problème scalaire, que l'on résout indépendamment sur tout l'espace $z > 0$.

$P[\lambda, u]$ est $\mathcal{C}^1$ sur $\mathbb{R}_*^+$ de dérivée :

$$P[\lambda, u]'(X) = n2^n K X^{n-1} + r. \tag{3.28}$$

On peut alors résoudre (3.21) par une simple méthode de Newton, voir Alg. 6. On rappelle que l'étape de Newton a la forme :

$$X^{(i+1)} = X^{(i)} - \frac{P[\lambda, u](X^{(k)})}{P[\lambda, u]'(X)}. \tag{3.29}$$

Le problème est que pour $n < 1$, la dérivée de $P[\lambda, u]$ explose au voisinage de 0. Il en résulte que l'étape de Newton (3.29) devient numériquement instable si la solution est proche de 0. C'est le cas lorsque z est proche de 0 i.e. à proximité des interfaces entre les zones fluides et rigides. On en déduit donc que la méthode de Newton naïvement appliquée au problème (3.21) risque d'être imprécise justement à proximité des zones qui nous intéressent. Il faut qui plus est s'assurer que les itérées $X^{(i)}$ restent strictement positives, or un calcul direct montre que :

$$X^{(i+1)} > 0 \Leftrightarrow X^{(i)} < \left(\frac{z}{2^n K(1-n)}\right)^{1/n}. \tag{3.30}$$

Il n'y a pas a priori de méthode simple pour trouver une initialisation $X^{(0)}$ assurant que cette condition soit vérifiée par toutes les itérations. Bien qu'il existe des méthodes permettant d'incorporer des contraintes dans une méthode de Newton (voir [177],[182]), celles-ci ne permettent pas de stabiliser la méthode de Newton au voisinage de 0. À la place, on reformule le problème de façon à obtenir un problème plus stable. On suppose $n < 1$ et on pose :

$$Y = X^n, \tag{3.31}$$

$$Q(Y) = \begin{cases} rY^{1/n} + 2^n K y - z & \text{si } z > 0 \\ 2^n K y - z & \text{si } z \leq 0, \end{cases} \tag{3.32}$$

où on a enlevé la dépendance en λ, u pour alléger les notations. Trouver la racine de $P[\lambda, u]$ équivaut alors à trouver celle de Q puis récupérer $X = Y^{1/n}$. Cette fois Q est $\mathcal{C}^1$ sur $\mathbb{R}$ entier et :

$$Q'(Y) = \begin{cases} \frac{r}{n} Y^{\frac{1}{n}-1} + 2^n K & \text{si } z > 0 \\ 2^n K & \text{si } z \leq 0. \end{cases} \tag{3.33}$$

Comme $n < 1$, Q est convexe sur $\mathbb{R}$. C'est sur cette fonction que l'on applique la méthode de Newton. En suivant les recommandations de [157], on remplace la méthode de Newton par une méthode de la sécante dans le cas où $n > 1/2$, car alors la dérivée seconde de Q explose en 0.

Algorithme 20 Algorithme de résolution de (3.21).

Requis : tolérance $\varepsilon > 0$

Si $n = 1/2$ **alors**
  Utiliser (3.26).
Sinon Si $z \leq \left(\frac{24K^{3/n}n^3}{r^3(n^2-6n+8)}\right)^{\frac{n}{4-3n}}$ **alors**
  Retourner

$$X = \left(\frac{z}{2^n K}\right)^{1/n} - \frac{1}{n}\left(\frac{z}{2^n K}\right)^{2/n}\frac{r}{z} + \frac{3-n}{2n^2}\left(\frac{z}{2^n K}\right)^{3/n}\left(\frac{r}{z}\right)^2 \tag{3.34}$$

Sinon

  Si n<1/2 **alors**
    $Y^{(0)} = z/r$, $i = 0$
  Sinon
    $Y^{(0)} = 0$, $Y^{(1)} = z/r$, $i = 1$
  Fin Si
  Tant que $|Y^{(i+1)} - Y^{(i)}| < \varepsilon$ **faire**
    Si n<1/2 **alors** ▷ en pratique, ce test est fait une seule fois en amont

$$Y^{(i+1)} = Y^{(i)} - \frac{Q(Y^{(i)})}{Q'(Y^{(i)})} \tag{3.35}$$

▷ Q est donné par (3.32), Q' est donné par (3.33)

    Sinon

$$Y^{(i+1)} = Y^{(i)} - \frac{Y^{(i)} - Y^{(i-1)}}{Q(Y^{(i)}) - Q(Y^{(i-1)})}Q(Y^{(i)}) \tag{3.36}$$

    Fin Si
    $i = i + 1$
  Fin Tant que
  Retourner $X = \left(Y^{(i)}\right)^{1/n}$.
Fin Si

Avant de donner l'algorithme final, on ajoute un dernier raffinement. Bien que la méthode de Newton soit numériquement plus stable pour Q, elle requiert de nombreuses évaluations de puissances avec des exposants réels a priori non entiers, ce qui coûte un temps CPU significatif. On peut partiellement contourner ce problème lorsque z est petit. En effet, Marly [157] montre que la solution de $P[\lambda, u](X) = 0$ vérifie lorsque $z \to 0$ (section 2.5.3) :

$$X = \underbrace{\left(\frac{z}{2^n K}\right)^{1/n} - \frac{1}{n}\left(\frac{z}{2^n K}\right)^{2/n}\frac{r}{z} + \frac{3-n}{2n^2}\left(\frac{z}{2^n K}\right)^{3/n}\left(\frac{r}{z}\right)^2}_{\tilde{X}(z)} + o\left(z^{3/(n-2)}\right). \tag{3.37}$$

Il montre même que l'on peut borner quantitativement l'erreur de l'estimation (3.37) :

$$|X - \tilde{X}(z)| \leq \varepsilon \Leftrightarrow z \leq \left(\frac{24K^{3/n}n^3}{r^3(n^2-6n+8)}\right)^{\frac{n}{4-3n}}. \tag{3.38}$$

On peut donc en début d'algorithme choisir d'utiliser $\tilde{X}(z)$ si z est petit plutôt que d'utiliser la méthode de Newton (ou la sécante), avec une précision garantie par (3.38). L'algorithme complet est présenté en 20.

Remarque 40 *En réalité, Marly [157] va plus loin en donnant un développement valable à tout ordre, avec une relation de récurrence permettant de calculer les coefficients, avec une généralisation de la borne* (3.38) *aux ordres supérieurs. Cependant, ici on ne s'intéresse qu'à ce développement à 3 termes dans la mesure où si l'on prend par exemple* $K = 10$, $n = 0.4$, $r = 200$ *et* $\varepsilon = 10^{-18}$, *alors le développement à 3 termes est valable dès que* $z \leq 0.0027$. *Il n'y a donc aucune nécessité de prolonger le développement.*

3.1.3 Dual FISTA

L'approche suivante basée sur l'algorithme Dual FISTA 16 a été introduite en 2016 par Treskatis et al. [239, 237]. On consultera aussi [240] qui discute de nombreux aspects pratiques liés à cette approche, notamment la question de la mesure de la convergence, sur laquelle on revient en section 3.2.2. On commence cette fois par traiter le cas Herschel-Bulkley, puis on traite les simplifications occasionnées par le cas Bingham.

3.1.3.1 Cas Herschel-Bulkley

On réutilise le splitting (3.15) utilisé en section précédente pour le cas Herschel-Bulkley. Les calculs permettant d'expliciter l'algorithme étant un peu plus volumineux, on renvoie à la thèse de Treskatis [237] chapitre 6, en particulier le lemme 6.6 et on récapitule les résultats dans l'algorithme 21.

On obtient un algorithme très similaire à l'algorithme ADMM 19. Excepté la méthode d'accélération (3.44), la différence majeure provient de l'échange entre les étapes de mise à jour de la vitesse et de $d^{(k)}$ (et de quelques modifications de constantes). Le problème de Stokes généralisé (3.42) ne diffère des précédents que par les valeurs des constantes mises en jeu, il peut donc être résolu par l'algorithme 18 présenté plus haut. On a toujours une mise à jour explicite pour la variable duale $\lambda^{(k)}$ et cette fois la mise à jour de $d^{(k)}$ est de nouveau purement explicite comme dans le cas Bingham. À la place, on a le problème de trouver la constante $L^{(k)}$. On rappelle que l'algorithme FISTA 14 demande à la partie différentiable de la fonction objectif d'être à gradient L-lipschitziens. Dans Dual FISTA 16, cela se traduit par la nécessité d'avoir une fonction fortement convexe. Or lorsque $n < 1$, la fonction f_0 n'est fortement convexe que sur les ensembles bornés de S. Tant que la méthode numérique fournit des suites bornées, on peut la supposer fortement convexe, on ne peut cependant pas à l'avance donner de constante de forte convexité, autrement dit le paramètre L est inconnu. Dans l'article fondateur de FISTA [35], Beck et Teboulle proposent dans cette éventualité d'utiliser une procédure de backtracking afin d'estimer L au cours de itérations. Pour cela, on commence par expliciter le dual du problème (3.2), qu'on peut écrire (voir [203]) :

$$\min_{\lambda,p} J(\lambda,p) := \min_{-\mathrm{div}(\lambda)+\nabla p = f} \int_\Omega \frac{n+1}{nK^{1/n}} \left(|\lambda| - \tau_y\right)_+^{1+\frac{1}{n}} dV - \int_{\partial\Omega} \langle (\lambda - pId).N_{\partial\Omega}, u_D\rangle dS, \quad (3.39)$$

où $N_{\partial\Omega}$ est la normale sortante à la frontière $\partial\Omega$. En fait, la contrainte $-\mathrm{div}\,(\lambda) + \nabla p = f$ et les conditions de bord u_D permettent de déterminer la pression p réalisant le minimum ci-dessus (on renvoie à la thèse de Treskatis [237] pour son calcul de la fonction duale). Il en résulte qu'on peut voir J comme une fonction de λ uniquement. Le backtracking proposé par Beck et Teboulle a alors pour objectif de satisfaire la condition de descente :

$$J(\lambda) \leq J(\tilde{\lambda}^{(k)}) + \int_\Omega \langle \nabla J(\tilde{\lambda}^{(k)}), \lambda - \tilde{\lambda}^{(k)}\rangle + \frac{L^{(k+1,i)}}{2} |\lambda - \tilde{\lambda}^{(k)}|^2. \quad (3.40)$$

Algorithme 21 Algorithme Dual FISTA pour le problème de Herschel-Bulkley (3.2).

1: $\tilde{\lambda}^{(0)} = \lambda^{(0)}$, $t^{(0)} = 1$, $\alpha > 1$, $L^{(0)} > 0$
2: **Pour** $k \geq 0$ **faire**
3:

$$d^{(k+1)} = \begin{cases} 0 & \text{si } |\tilde{\lambda}^{(k)}| \leq \tau_y, \\ \dfrac{1}{2K^{1/n}} \left(|\tilde{\lambda}^{(k)}| - \tau_y \right)^{1/n} \dfrac{\tilde{\lambda}^{(k)}}{|\tilde{\lambda}^{(k)}|} & \text{si } |\tilde{\lambda}^{(k)}| > \tau_y. \end{cases} \tag{3.41}$$

▷ mise à jour explicite point par point

4: $L^{(k+1,0)} = L^{(k)}$, $i = 0$
5: **Tant que** (3.40) n'est pas satisfait **faire**
6: Résoudre le problème en (u, p) :

$$\begin{cases} -\frac{1}{L^{(k+1,i)}} \operatorname{div}(Du) + \nabla p = f + \operatorname{div}\left(\tilde{\lambda}^{(k)} - \frac{1}{L^{(k+1,i)}} d^{(k+1)} \right) \\ \operatorname{div}(u) = 0 \\ u = u_D \text{ sur } \partial\Omega. \end{cases} \tag{3.42}$$

▷ utiliser l'algorithme 18

7:

$$\lambda = \tilde{\lambda}^{(k)} + \frac{1}{L^{(k+1,i)}} \left(Du - d^{(k+1)} \right) \tag{3.43}$$

▷ mise à jour explicite point par point

8: $L^{(k+1,i+1)} = \alpha L^{(k+1,i)}$, $i = i + 1$
9: **Fin Tant que**
10: $L^{(k+1)} = L^{(k+1,i-1)}$, $u^{(k+1)} = u$, $p^{(k+1)} = p$, $\lambda^{(k+1)} = \lambda$
11: Choisir $t^{(k+1)}$ (voir section 2.5.4)
12:

$$\tilde{\lambda}^{(k+1)} = \lambda^{(k+1)} + \frac{t^{(k)} - 1}{t^{(k+1)}} \left(\lambda^{(k+1)} - \lambda^{(k)} \right) \tag{3.44}$$

13: **Fin Pour**

On en déduit la boucle interne (boucle en i) permettant de fixer $L^{(k)}$. Il y a plusieurs facteurs à prendre en compte pour évaluer le poids de cette boucle :

- chaque itération de cette boucle demande la résolution du problème de Stokes ainsi que l'évaluation de la fonction duale J (une intégrale sur tout l'espace) et est donc bien plus coûteuse qu'une itération de l'algorithme de Newton demandé par ADMM,
- sur les premières itérations de la boucle externe, l'algorithme de résolution du Stokes 18 demande de nombreuses itérations pour converger, chaque itération de la boucle de backtracking est d'autant plus longue ; on peut cependant espérer un gain en réutilisant la solution (u, p) de l'itération de backtracking précédente, mais il est difficile d'estimer si la solution est très sensible à la valeur de $L^{(k,i)}$,
- lorsque la boucle externe a effectué plusieurs itérations, on peut s'attendre à faire très peu d'itérations de la boucle de backtracking, puisque dès que la boucle externe s'approche de la solution, on devrait rapidement trouver un $L^{(k)}$ qui fonctionne pour toutes les itérées suivantes,
- l'objectif du schéma d'accélération de FISTA est qui plus est de réduire le nombre d'itérations de la boucle externe, ce qui implique moins d'appels à la boucle de backtracking.

Il est donc difficile de dire a priori si, en pratique, Dual FISTA 21 est plus rapide que ADMM 19, dans le cas Herschel-Bulkley.

3.1.3.2 Cas Bingham

Le cas Bingham $n = 1$ est bien plus favorable, puisqu'alors la fonction f_0 devient fortement convexe. On peut donc expliciter $L = 2\eta$ pour obtenir l'algorithme 22.

Algorithme 22 Algorithme Dual FISTA pour le problème de Bingham (3.3).

1: $\tilde{\lambda}^{(0)} = \lambda^{(0)}$, $t^{(0)} = 1$

2: **Pour** $k \geq 0$ **faire**

3:
$$d^{(k+1)} = \begin{cases} 0 & \text{si } |\tilde{\lambda}^{(k)}| \leq \tau_y, \\ \dfrac{1}{2K^{1/n}} \left(|\tilde{\lambda}^{(k)}| - \tau_y \right)^{1/n} \dfrac{\tilde{\lambda}^{(k)}}{|\tilde{\lambda}^{(k)}|} & \text{si } |\tilde{\lambda}^{(k)}| > \tau_y. \end{cases} \tag{3.45}$$

▷ mise à jour explicite point par point

4: Résoudre le problème en $\left(u^{(k+1)}, p^{(k+1)}\right)$:

$$\begin{cases} -2\eta \mathrm{div}\left(Du^{(k+1)}\right) + \nabla p^{(k+1)} = f + \mathrm{div}\left(\tilde{\lambda}^{(k)} - 2\eta d^{(k+1)}\right) \\ \mathrm{div}\left(u^{(k+1)}\right) = 0 \\ u^{(k+1)} = u_D \text{ sur } \partial\Omega. \end{cases} \tag{3.46}$$

▷ utiliser l'algorithme 18

5:
$$\lambda^{(k+1)} = \tilde{\lambda}^{(k)} + 2\eta \left(Du^{(k+1)} - d^{(k+1)} \right) \tag{3.47}$$

▷ mise à jour explicite point par point

6: Choisir $t^{(k+1)}$ (voir section 2.5.4)

7:
$$\tilde{\lambda}^{(k+1)} = \lambda^{(k+1)} + \frac{t^{(k)} - 1}{t^{(k+1)}} \left(\lambda^{(k+1)} - \lambda^{(k)} \right) \tag{3.48}$$

8: **Fin Pour**

Il n'y a alors plus de boucle externe, l'algorithme a donc la même complexité, par itération externe, que l'algorithme ADMM 17. On note que la question du choix du paramètre d'augmentation r est remplacée par celle du choix du schéma d'accélération (i.e. la formule de $t^{(k)}$). Ce choix est étudié dans la section 3.2 concernant les résultats expérimentaux.

3.1.4 Discrétisation

Il est désormais opportun de présenter la discrétisation qui est utilisée pour les expériences numériques. Comme évoqué précédemment, on choisit ici de se placer en différences finies. En suivant [247] et [173], les différentes inconnues sont placées sur une grille décalée, selon un maillage *Marker and Cell* (MAC) à la Harlow et Welch [124] schématisé dans les figures 3.1.2 et 3.1.3.

3.1.4.1 Maillage

On note un point de l'espace $z = (x, y)$. Les points de maillages sont notés $z_{i,j}$ pour (x_i, y_j). La composante en x (respectivement y) de la vitesse u^x (respectivement u^y) est calculée au milieu des arêtes verticales (respectivement horizontales) du maillage, sur les points $z_{i,j+1/2}$ (respectivement les $z_{i+1/2,j}$), qui correspondent aux emplacements en forme de carré (respectivement d'étoile) sur la figure de gauche de 3.1.2. La pression est quant à elle placée au milieu des mailles, en $z_{i+1/2,j+1/2}$, donc les points noirs. Pour un tenseur τ, ses composantes diagonales (τ^{xx}

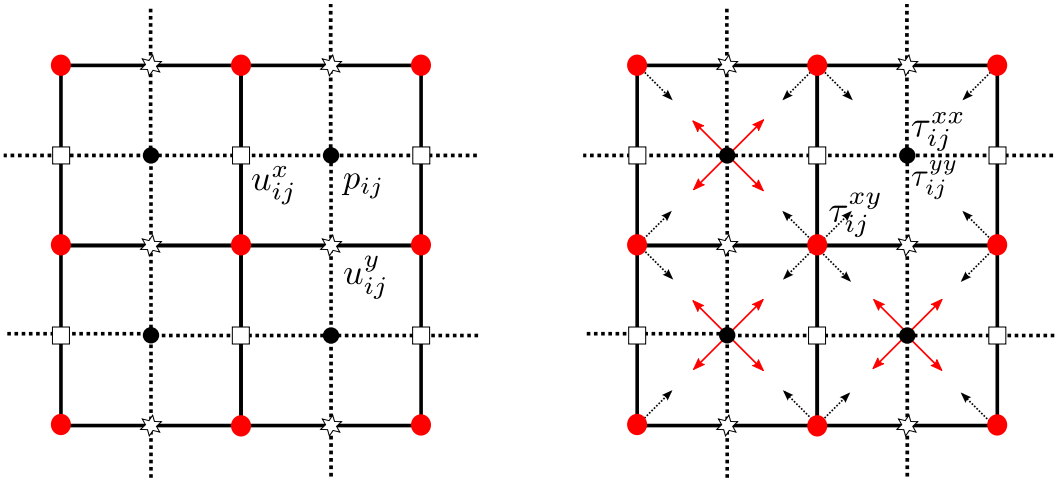


FIGURE 3.1.2 – Maillage de type MAC. À gauche : détail d'une maille intérieure pour la vitesse et la pression. À droite : détail d'une maille intérieure pour les différents tenseurs.

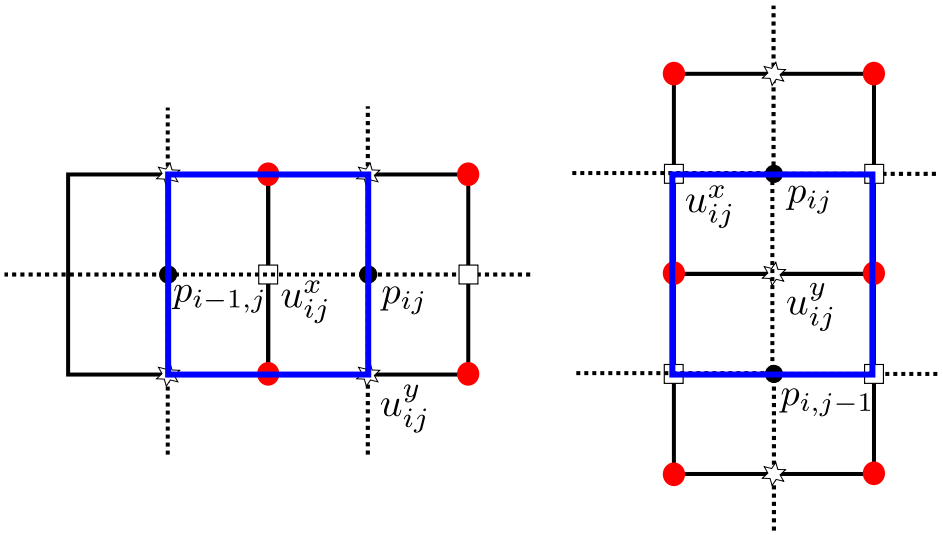


FIGURE 3.1.3 – Maillage de type MAC, les mailles de contrôles (en bleu) sont centrées en $u^x_{i,j}$ pour discrétiser (3.49a) (à gauche), en $u^y_{i,j}$ pour discrétiser (3.49b) (à droite).

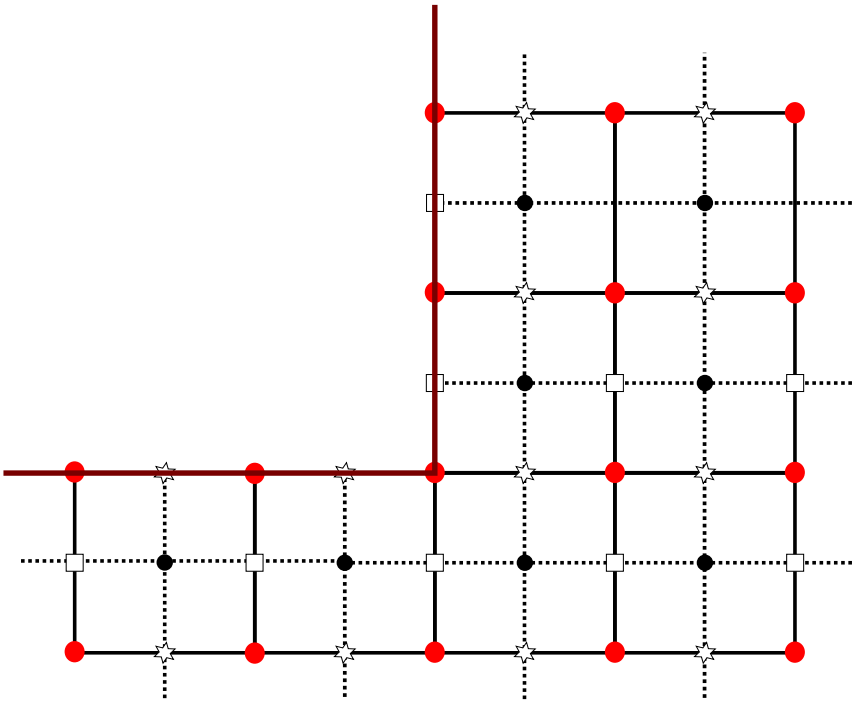


FIGURE 3.1.4 – Disposition des mailles de la frontière (en marron) au niveau des coins.

et τ^{yy}) sont toutes deux placées sur les mêmes points que la pression. Pour finir, sa composante extra-diagonale τ^{xy} est placée sur chaque sommet des mailles, donc sur les points $z_{i,j}$, ce qui correspond aux points rouges dans la figure 3.1.2. On note Δx le pas d'espace en x et Δy le pas d'espace en y. Pour finir, on adapte Δx et Δy de sorte à ce que la frontière de Ω passe par des points de type τ^{xy}, les coins devant coïncider avec un de ces points, voir figure 3.1.4.

3.1.4.2 Système de Stokes discret

Ces emplacements permettent d'utiliser des schémas centrés pour tous les calculs de dérivées qui sont nécessaires pour les problèmes de Stokes généralisés. En effet, si on développe le problème linéaire (3.13) qui intervient dans la résolution du problème de Stokes, on obtient :

$$-\alpha\partial_x\left(\partial_x u^x + \frac{\partial_x u^y + \partial_y u^x}{2}\right) - s\left(\partial_{xx}u^x + \partial_{xy}u^y\right) = f^x + \partial_x\tau^{xx} + \partial_y\tau^{xy} - \partial_x p \tag{3.49a}$$

$$-\alpha\partial_y\left(\partial_y u^y + \frac{\partial_x u^y + \partial_y u^x}{2}\right) - s\left(\partial_{yy}u^y + \partial_{xy}u^x\right) = f^y + \partial_x\tau^{xy} + \partial_y\tau^{yy} - \partial_y p, \tag{3.49b}$$

avec :

$$\tau = \lambda - \beta d, \tag{3.50}$$

où α et β dépendent de la boucle externe et où on a enlevé les exposants i et k pour alléger les notations. On peut alors utiliser des différences finies centrées pour calculer tous les termes intervenant dans (3.49). Plus exactement, les deux lignes sont discrétisées sur deux volumes de contrôle différents. L'équation en x (3.49a) est discrétisée sur les mailles centrées en les $x_{i,j+1/2}$, i.e. les $u^x_{i,j}$, et l'équation en y (3.49b) est discrétisée sur les mailles centrées en les $z_{i+1/2,j}$, voir

figure 3.1.3. L'équation (3.49a) est alors discrétisée en :

$$\begin{aligned}\left(\frac{\alpha+s}{(\Delta x)^2}+\frac{\alpha}{2(\Delta y)^2}\right)u^x_{i,j}-\frac{\alpha+s}{2(\Delta x)^2}\left(u^x_{i+1,j}+u^x_{i-1,j}\right)-\frac{\alpha}{4(\Delta y)^2}\left(u^x_{i,j+1}+u^x_{i,j-1}\right)\\ -\frac{\alpha+2s}{2\Delta x\Delta y}\left(u^y_{i,j+1}-u^y_{i-1,j+1}-u^y_{i,j}+u^y_{i-1,j}\right)=f^x_{i,j}-\frac{p_{i,j}-p_{i-1,j}}{\Delta x}\\ +\frac{\tau^{xx}_{i,j}-\tau^{xx}_{i-1,j}}{\Delta x}+\frac{\tau^{xy}_{i,j}-\tau^{xy}_{i,j-1}}{\Delta y}.\end{aligned} \tag{3.51}$$

L'équation (3.49b) est quant à elle discrétisée en :

$$\begin{aligned}\left(\frac{\alpha+s}{(\Delta y)^2}+\frac{\alpha}{2(\Delta x)^2}\right)u^y_{i,j}-\frac{\alpha+s}{2(\Delta y)^2}\left(u^y_{i+1,j}+u^y_{i-1,j}\right)-\frac{\alpha}{4(\Delta x)^2}\left(u^y_{i,j+1}+u^y_{i,j-1}\right)\\ -\frac{\alpha+2s}{2\Delta x\Delta y}\left(u^x_{i,j+1}-u^x_{i-1,j+1}-u^x_{i,j}+u^x_{i-1,j}\right)=f^y_{i,j}-\frac{p_{i,j}-p_{i,j-1}}{\Delta y}\\ +\frac{\tau^{xy}_{i,j}-\tau^{xy}_{i-1,j}}{\Delta x}+\frac{\tau^{yy}_{i,j}-\tau^{yy}_{i,j-1}}{\Delta y}.\end{aligned} \tag{3.52}$$

Cela permet d'obtenir une erreur de consistance en $\mathcal{O}(\|(\Delta x,\Delta y)\|^2)$.

3.1.4.3 Normes de tenseur

Le problème que cela cause concerne les étapes de mise à jour de la variable d ((3.11) pour l'Alg. 17 et (3.45) pour l'Alg. 22), puisqu'il faut alors calculer la norme d'un tenseur τ, à savoir $|\lambda+rDu|$ dans le cas ADMM et $|\lambda|$ dans le cas Dual FISTA. On a besoin de ce type de normes à la fois au centre $z_{i+1/2,j+1/2}$ des mailles (points homogènes aux composantes diagonales de tenseur) et aux coins $z_{i,j}$ des mailles (points homogènes aux composantes extra-diagonales). À chaque emplacement, lorsqu'une quantité requise dans le calcul de norme est inconnue, on fait une moyenne des 4 points connus l'entourant, comme montré en figure 3.1.2. Ainsi, pour calculer la norme au centre d'une maille, on commence par approcher la composante extra-diagonale de τ à cet emplacement (on note cette approximation $\tilde{\tau}$) :

$$\tilde{\tau}^{xy}_{i+1/2,j+1/2}=\frac{\tau^{xy}_{i,j}+\tau^{xy}_{i+1,j}+\tau^{xy}_{i,j+1}+\tau^{xy}_{i+1,j+1}}{4}. \tag{3.53}$$

On utilise ensuite cette approximation pour calculer la norme de τ au centre (on modifie légèrement les notations d'indices pour insister sur les différents emplacements de chaque type de variable) :

$$2|\tau|^2_{i+1/2,j+1/2}=\left(\tau^{xx}_{i+1/2,j+1/2}\right)^2+\left(\tau^{yy}_{i+1/2,j+1/2}\right)^2+2\left(\tilde{\tau}^{xy}_{i+1/2,j+1/2}\right)^2. \tag{3.54}$$

Au coin des mailles, on calcule donc :

$$\tilde{\tau}^{xx}_{i,j}=\frac{\tau^{xx}_{i-1/2,j-1/2}+\tau^{xx}_{i+1/2,j-1/2}+\tau^{xx}_{i-1/2,j+1/2}+\tau^{xx}_{i+1/2,j+1/2}}{4}, \tag{3.55a}$$

$$\tilde{\tau}^{yy}_{i,j}=\frac{\tau^{yy}_{i-1/2,j-1/2}+\tau^{yy}_{i+1/2,j-1/2}+\tau^{yy}_{i-1/2,j+1/2}+\tau^{yy}_{i+1/2,j+1/2}}{4}, \tag{3.55b}$$

suivi de :

$$2|\tau|^2_{i,j}=\left(\tilde{\tau}^{xx}_{i,j}\right)^2+\left(\tilde{\tau}^{yy}_{i,j}\right)^2+2\left(\tau^{xy}_{i,j}\right)^2. \tag{3.56}$$

Ces procédures de moyenne sont impossibles sur les bords, puisqu'il n'y a pas 4 points entourant le point considéré. Faute de mieux, on effectue une moyenne sur les seuls points disponibles, donc deux points sur les bords et un seul dans les coins.

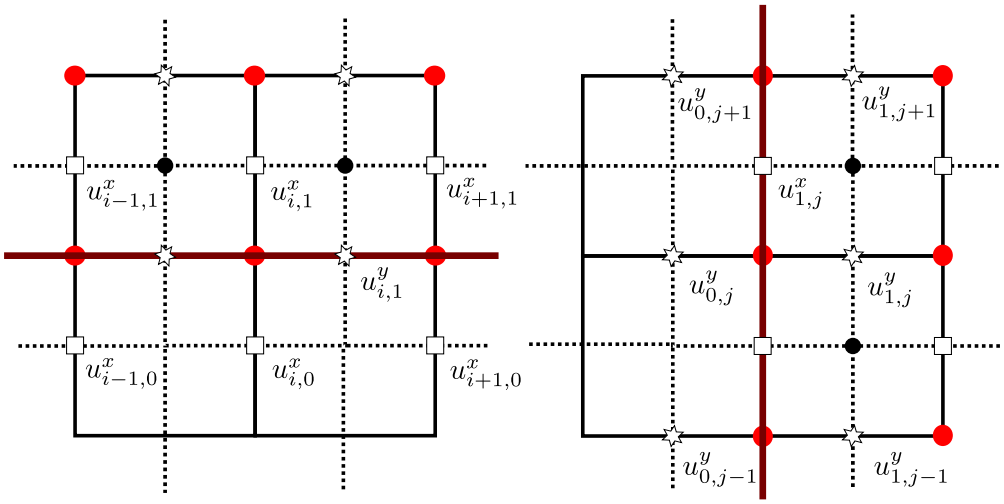


FIGURE 3.1.5 – Prolongement de u^x (à gauche) et u^y (à droite) en dehors du bord physique (en marron).

3.1.4.4 Conditions de bord

On rappelle qu'on utilise ici des conditions de bord de Dirichlet (on trouvera dans la thèse de Marly [157] une proposition de traitement de conditions de bord de type Neumann, voir notamment la section 2.3.2). On rappelle que l'on se place dans la géométrie d'expansion-contraction 3.1.1. On distingue deux zones distinctes :

- sur les murs latéraux, sur lesquels on impose une condition de non-glissement :

$$u = (0, 0), \tag{3.57}$$

- on suppose qu'à l'extérieur du domaine de calcul, les canaux d'entrée et de sortie se prolongent suffisamment de sorte à ce qu'un régime de Poiseuille (voir section 1.1.4) soit installé. On impose donc au début du canal d'entrée et à la fin du canal de sortie un régime de Poiseuille :

$$u = (u^x_{pois}, 0), \tag{3.58}$$

où u^x_{pois} est donné par (1.27) dans le cas Bingham et (1.25) dans le cas Herschel-Bulkley.

Ces conditions sont traitées de manière directe aux endroits où les vitesses sont calculées sur le bord, i.e. on impose directement une valeur sur les degrés de liberté concernés. Plus exactement :

- au début du canal d'entrée et à la fin du canal de sortie, u^x se trouve sur les bords, on impose donc u^x_{pois} de manière directe,
- sur les murs supérieurs et inférieurs des canaux d'entrée et de sortie ainsi que de la cavité, u^y se trouve sur le bord, on l'impose à 0 de manière directe,
- sur les murs latéraux de la cavité, u^x se trouve sur les bords, on impose donc ses degrés de liberté à 0.

En dehors de ces cas-là, du fait du choix du maillage décalé, les vitesses ne se trouvent pas directement sur le bord du domaine. On prend par exemple le cas de u^x sur un bord inférieur. Alors les équations (3.51) et (3.52) nécessitent sur certaines mailles l'évaluation de $u^x_{i,0}$ qui n'existe pas. On prolonge alors u^x une demi-maille en dehors du domaine. On crée ce point fantôme à l'aide d'un développement limité, c'est à travers ce dernier que l'on impose les conditions de bord restantes. Plaçons-nous dans un rectangle donné (un des deux canaux ou la cavité) et notons

$1 \leq j \leq M^y$. En rappelant que toutes les conditions de bord qu'il nous reste à établir sont nulles, on écrit :

$$\begin{aligned} u^x\left(x_i, -\frac{\Delta y}{2}\right) &= \underbrace{u^x(x_i,0)}_{=0} - \frac{\Delta y}{2}\partial_y u^x(x_i,0) + \mathcal{O}\left(k^2\right), \\ u^x\left(x_i, \frac{\Delta y}{2}\right) &= \underbrace{u^x(x_i,0)}_{=0} + \frac{\Delta y}{2}\partial_y u^x(x_i,0) + \mathcal{O}\left(k^2\right), \end{aligned} \tag{3.59}$$

ce qui nous permet de poser $u^x_{i,0} = -u^x_{i,1}$ pour obtenir une erreur d'ordre 2. On procède de même pour effectuer tous les prolongements qui s'imposent (voir figure 3.1.5) :

$$u^x_{i,0} = -u^x_{i,1}, \tag{3.60a}$$
$$u^x_{i,M^x+1} = -u^x_{i,M^x}, \tag{3.60b}$$
$$u^y_{0,j} = -u^y_{1,j}, \tag{3.60c}$$
$$u^y_{M^y+1,j} = -u^y_{M^y,j}. \tag{3.60d}$$

Il suffit alors de reporter ces valeurs dans les équations (3.51) et (3.52) et dans les calculs de norme de $\lambda + rDu$ lorsque nécessaire.

3.2 Résultats numériques pour Bingham

On présente maintenant des comparaisons des performances numériques de ces deux algorithmes. On s'intéresse ici au cas de la géométrie en croix 3.1.1, dans le cas du modèle de Bingham $n = 1$, qui entraîne la formation de plugs avec des topologies non triviales, on pourra par exemple consulter Marly et Vigneaux [158]. La section 3.2.1 présente les valeurs des paramètres qui sont restés libres dans les descriptions ci-dessus. La section 3.2.2 discute de la mesure de la convergence des algorithmes, qui est ici non triviale. La section 3.2.3 compare les résultats des différentes méthodes sur une variété de géométries en croix et de rhéologies, complétée dans la section 3.2.4 par une étude sur la convergence des méthodes sous raffinement de maillage. Pour finir, la section 3.2.5 réalise un bilan sur les performances de l'algorithme 18 de résolution du problème de Stokes.

Les algorithmes ont été implémentés en Fortran, à l'aide de la bibliothèque parallèle PETSc [15, 14, 16] qui permet la parallélisation des calculs et de la mémoire. Les problèmes linéaires requis dans l'algorithme 18 de résolution du sous-problème de Stokes sont réalisées par la bibliothèque parallèle MUMPS [5, 4], via une factorisation LU réutilisée tout le long de l'algorithme. Les calculs ont été réalisés au CBPsmn, sur 8 cœurs pour la plupart des simulations (12 cœurs pour les maillages les plus fins).

3.2.1 Réglage des paramètres libres

3.2.1.1 Paramètres des algorithmes

Pour commencer, la vitesse, la pression et les tenseurs sont tous initialisés à 0. Il a été vérifié que choisir d'autres initialisations (telles que le Poiseuille dans la continuité du canal) ne change pas les résultats.

L'algorithme 18 utilisé pour résoudre le problème de Stokes généralisé est arrêté lorsque $\|\mathrm{div}\left(u^{(k,i)}\right)\|_{L^2} \leq 5 \times 10^{-12}$, selon les recommandations de [157, 158]. Ce critère a toujours été atteint aussi il n'a pas été nécessaire d'imposer un nombre maximal d'itérations.

Du point de vue de la discrétisation, on prend le même pas en x et en y, i.e. $\Delta x = \Delta y$. Plusieurs valeurs de Δx sont testées aussi elle seront précisées au moment opportun.

Ensuite, concernant les paramètres d'augmentation des Lagrangiens augmentés r (pour ADMM 17) et s (pour le solveur 18 du Stokes), il n'y a malheureusement pas de manière générale de les choisir. On suit donc les recommandations de Marly [157, 158] qui a étudié l'impact de différentes valeurs de ces paramètres sur la performance de ADMM (voir la remarque 11 du chapitre 2 de la thèse). On prend donc $r = 200$ et $s = 2000$.

Le choix suivant concerne le schéma d'accélération de la méthode Dual FISTA (i.e. la formule de t_k). Différents schémas ont été discutés dans la section 2.5.4 présentant FISTA. Il n'existe pas de consensus sur le schéma exact à utiliser, aussi on choisit d'en tester plusieurs. On prend le schéma d'origine (2.77) introduit par Nesterov et utilisé par Beck et Teboulle lors de la création de FISTA. On compare les résultats avec le schéma (2.144), proposé par Chambolle et Dossal, qui semble posséder plus de propriétés théoriques assurant la stabilité de l'algorithme (voir section 2.5.4). On utilise plusieurs valeurs du paramètre a intervenant dans ce schéma. Afin d'essayer plusieurs ordres de grandeur du paramètre, on essaie 4, 50, 100 et 400.

Dans la section 2.5.4.2, la difficulté de calculer les variables primales en fin d'algorithme de Dual FISTA avait été évoqué. En effet, les variables primales qui sont calculées au cours de l'algorithme ne correspondent pas à la variable duale renvoyée à la fin (à cause du schéma d'accélération). On utilise dans toute la section la formule (2.157) qui permet de récupérer les variables primales correspondant bien à la variable duale non-accélérée. En pratique, bien que cela revête le coût, en temps de calcul, d'une itération supplémentaire, il faut généralement plusieurs milliers d'itérations pour obtenir les résultats présentés après, le coût de cette dernière étape est donc négligeable.

Enfin, une technique classique pour stabiliser FISTA consiste à utiliser des *restarts*, c'est-à-dire réinitialiser régulièrement le schéma d'accélération (i.e. poser $t^{(k)}$=1), on pourra par exemple consulter Liang et al. [151] ou O'Donoghue et Candès [183]. Ces derniers proposent de redémarrer l'algorithme dès que :

$$\langle Du^{(k+1)} - d^{(k+1)}, \lambda^{(k+1)} - \lambda^{(k)} \rangle < 0. \tag{3.61}$$

Contrairement à l'algorithme MFISTA 15, cette technique a l'avantage de ne pas nécessiter l'évaluation de la fonctionnelle, avec toutes les difficultés techniques que cela peut occasionner (on remarque que toutes les quantités dans (3.61) sont déjà calculées dans l'algorithme).

On note donc "FISTA BT" pour désigner la variante de FISTA utilisant le schéma (2.77) puis "FISTA CD a" pour le schéma (2.144), où on remplace a par sa valeur. Si on utilise la procédure de restart décrite précédemment, on note par exemple "FISTA CD a RESTART".

3.2.1.2 Spécification des paramètres de géométrie

La géométrie possède plusieurs paramètres libres, on en fixe certains. Suivant l'étude réalisée par Marly [157] (voir remarque 14 du chapitre 2), imposer des conditions de Poiseuille à l'entrée et à la sortie requiert que les canaux soient suffisamment longs pour permettre à un profil de Poiseuille de se développer. Pour ce faire, on fixe donc la longueur des canaux d'entrée et de sortie :

$$L_{inout} = 2D, \tag{3.62}$$

où on rappelle que D désigne le rayon des canaux (voir figure 3.1.1).

Ensuite, les forces extérieures f sont prises nulles, le mouvement du fluide provient uniquement d'un forçage en pression (qui se traduit par la formule de Poiseuille utilisée au bord).

Enfin on spécifie les paramètres des simulations via les paramètres adimensionnés B, δ et h. Pour que cela suffise à définir entièrement les paramètres d'entrée d'une simulation, on impose

que la vitesse moyenne V (du profil de Poiseuille) vérifie $V = 1$. Cela permet de déterminer le gradient de pression qui est imposé. Précisément, il doit vérifier :

$$dp = -\frac{B}{X} \text{ où } X^3 - 3\left(\frac{2}{B} + 1\right) X + 2 = 0, \tag{3.63}$$

ou alors en version dimensionnée :

$$dp = -\frac{\tau_y}{RX} \text{ où } X^3 - 3\left(\frac{2}{Bi} + 1\right) X + 2 = 0, \tag{3.64}$$

où Bi est un nombre imposé (à la place du B adimensionné).

3.2.2 Mesure de convergence

Pour la géométrie en croix 3.1.1, il n'existe pas de solution analytique connue au problème de Bingham. Il faut donc se poser la question de comment mesurer la convergence des algorithmes. Une mesure courante (voir par exemple [53]) consiste à regarder ce qu'on désigne ci-après comme le *résidu* :

$$res = \|d - Du\|. \tag{3.65}$$

À la convergence de l'algorithme, le résidu est nul. Cependant, Treskatis et al. [240] font remarquer que le résidu peut être très proche de 0 sans que l'algorithme ait réellement convergé (voir section 2.1 et notamment la figure 2). Pour remédier à ce problème, ils proposent d'utiliser ce qu'ils appellent le *dual gap*, c'est-à-dire la différence entre la valeur de la fonction primale et du coût dual (pour l'itérée en question). Cette quantité n'est cette fois 0 que si l'algorithme a convergé. Toutefois, le dual gap possède un inconvénient majeur : les fonctions primales et duales valent l'infini lorsque les contraintes ne sont pas vérifiées. Or, dans le cas d'une discrétisation différences finies, les contraintes, qui sont normalement vérifiées par construction de l'algorithme FISTA, ne sont plus vérifiées de manière exacte (en fait, seuls les éléments finis P_1-iso-P_2 et ceux de Brezzi-Douglas-Marini permettent de conserver cette propriété). Il en résulte que soit on doit considérer que le dual gap est infini à toutes les itérations (ce qui est donc inutile), soit on doit ignorer les contraintes, mais alors la majoration de la fonction duale par la fonction primale n'est plus nécessairement vérifiée. En pratique, l'implémentation Fortran fournit bel et bien un dual gap négatif lorsque les itérations augmentent, il est donc inutilisable. Le résidu (3.65) reste donc notre meilleure métrique à l'heure actuelle. Celui-ci doit être complété par des observations humaines des résultats pour en vérifier la cohérence physique (notamment sur l'interface fluide-rigide).

Remarque 41 *Par ailleurs, dans le cas où le dual gap est calculable, son utilisation sur ADMM est coûteuse en pratique. En effet, dans le cas de Dual FISTA, Treskatis et al. [240] montrent comment calculer le dual gap à partir des quantités qui sont déjà calculées par l'algorithme. Cependant, pour ADMM il est nécessaire de résoudre un problème auxiliaire s'apparentant au problème de Stokes* (3.46)*. Cela signifierait donc doubler le coût de calcul de l'algorithme.*

3.2.3 Comparaisons sur différentes géométries en croix

On commence par comparer les algorithmes sur différents cas tests de géométrie en croix, issus des travaux de Marly et al. [157, 158]. On choisit pour cela quatre paramétrages différents de la géométrie et pour chacun, on prend cinq valeurs différentes de B. Ces paramètres sont résumés en figure 3.2.1, où sont aussi représentés les plugs correspondant à chaque cas. Suivant [157, 158], le pas d'espace est choisi de sorte à obtenir 600 points dans une section verticale de la cavité, i.e. $\Delta x = h/300$.

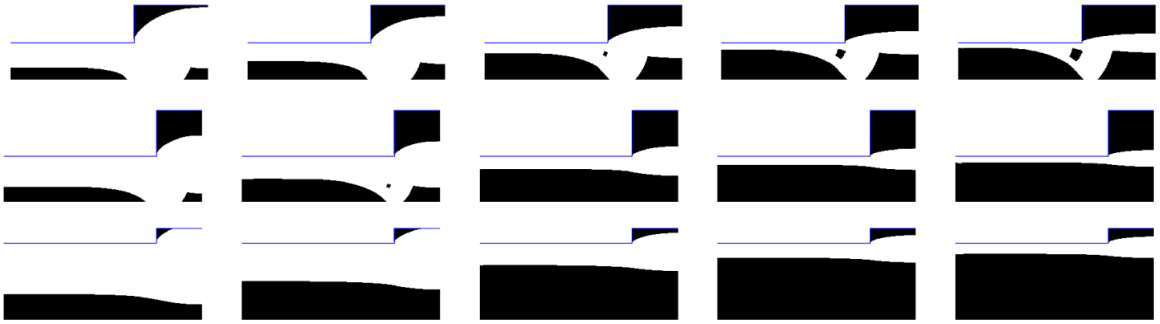


FIGURE 3.2.1 – Zones plastiques des cas tests utilisés pour comparer ADMM et FISTA. On profite des symétries de la géométrie pour ne montrer ici que la partie supérieure gauche du domaine. De haut en bas, on a $h = 2; 2; 6/5$, $\delta = 1/4; 1/2; 5/6$. De gauche à droite, on a $B = 2; 5; 20; 50; 100$. Cette figure est extraite de la thèse de Marly [157].

3.2.3.1 ADMM - FISTA

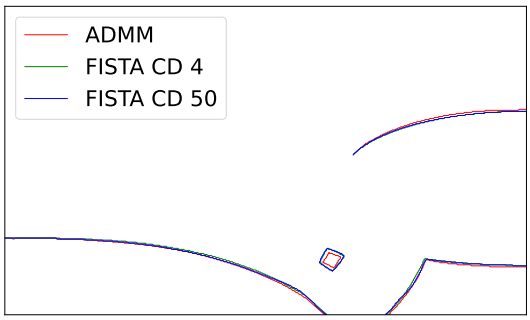
On compare dans cette section les performances d'ADMM et de FISTA muni du schéma de Chambolle et Dossal avec $a = 4$ et $a = 50$ (les autres schémas sont étudiés dans la section suivante). Les algorithmes sont lancés pour un maximum de 50 000 itérations et sont stoppés si le résidu descend sous 6×10^{-12}.

Remarque 42 *La limite de 50 000 itérations a ponctuellement été repoussée lorsque la convergence n'était pas terminée après 50 000 itérations mais que le résidu continuait à descendre à vitesse raisonnable.*

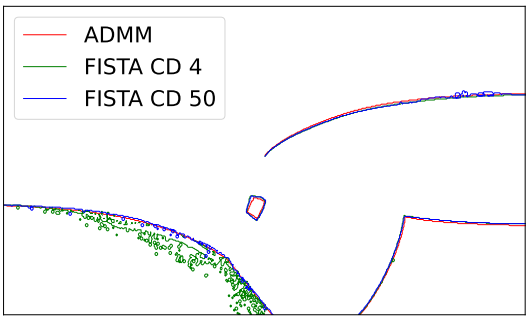
Comparaison des limites de plug On commence par comparer les délimitations de plug obtenues après convergence des différents algorithmes. Comme observé par Marly et al. [157, 158], utiliser d plutôt que Du pour déterminer les zones rigides permet d'obtenir des délimitations de meilleure qualité. Numériquement, les plugs sont caractérisés par $d \leq \varepsilon$. On choisit ici $\varepsilon = 10^{-10}$. Un échantillon représentatif des résultats obtenus est montré en figure 3.2.2.

Dans le cas $h = 6/5$, $\delta = 5/6$, $B = 5$, la figure 3.2.2d montre que les résultats fournis par les différents algorithmes sont très proches, bien que l'on puisse voir quelques légers décalages entre les limites fournies par ADMM et celles fournies par FISTA au niveau de la zone morte de la cavité (en haut à droite de la figure). Pour cette géométrie, les autres valeurs de B fournissent les mêmes observations. Dans le cas de la géométrie $h = 2$, $\delta = 1/2$, la figure 3.2.2a montre une différence plus importante entre les plugs trouvés par ADMM et les différentes versions de FISTA. Ceci se retrouve pour différentes valeurs de B. Il convient de noter qu'à chaque fois, la forme générale ainsi que la topologie des plugs sont identiques entre les résultats fournis par ADMM et FISTA, on observe cependant que les plugs obtenus via FISTA ont une tendance à être très légèrement plus gros que ceux obtenus via ADMM. Les deux résultats sont physiquement cohérents, aussi en l'absence de solution analytique il est difficile d'établir avec certitude quelle est la meilleure solution. On est cependant enclin à supposer, pour ce maillage précis, que les résultats d'ADMM sont plus proches de la solution du problème, notamment suite aux travaux de Balmforth et al. [17] sur les *sliplines*. Cependant en section 3.2.4 ont été testés des maillages plus fins sur un autre cas. Dans ce cas plus raffiné, les délimitations de plug données par ADMM et FISTA sont beaucoup plus proches et en accord avec la théorie des sliplines mentionnée ci-dessus (voir figure 3.2.12).

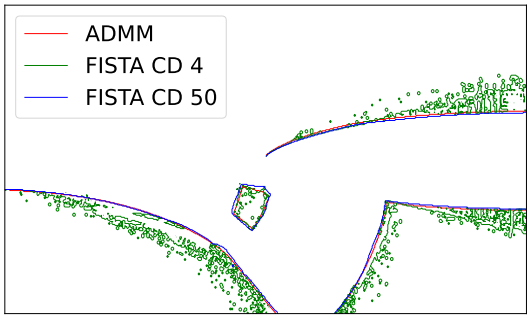
Pour la géométrie $h = 2$, $\delta = 1/4$ (ainsi que dans le cas $h = 2$, $\delta = 1/2$, $B = 100$ qui n'est pas



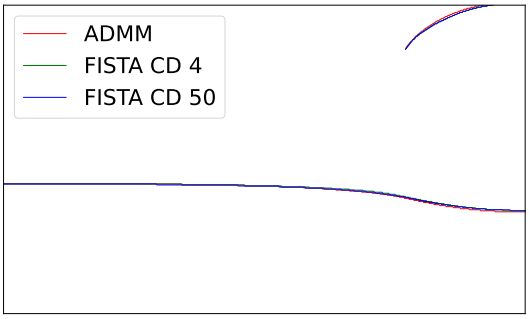
(a) $h = 2$, $\delta = 1/2$, $B = 5$



(b) $h = 2$, $\delta = 1/4$, $B = 20$



(c) $h = 2$, $\delta = 1/4$, $B = 50$

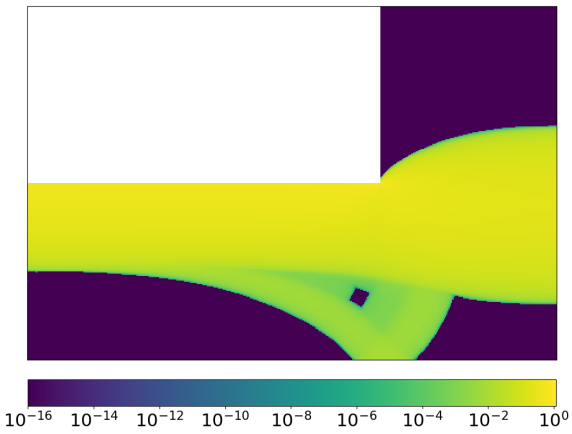


(d) $h = 6/5$, $\delta = 5/6$, $B = 5$

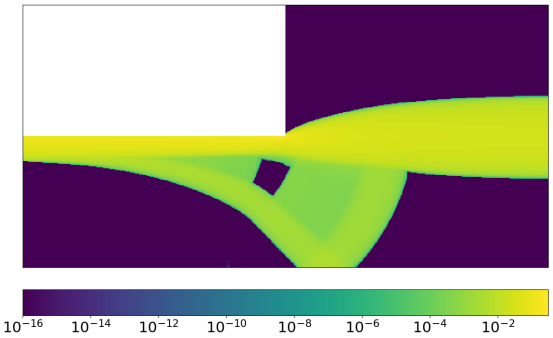
FIGURE 3.2.2 – Comparaison des délimitations de plug données par ADMM et FISTA dans différents cas. Pour la lisibilité, seul le quart en haut à gauche du domaine est représenté, mais les calculs sont réalisés sur le domaine complet.

montré ici), la situation se dégrade pour FISTA. En effet, les figures 3.2.2b et 3.2.2c montrent des délimitations de plug extrêmement bruitées pour FISTA CD 4, au point qu'il devient impossible d'identifier clairement la zone de plug. Le phénomène est drastiquement amoindri mais toujours présent dans le cas de FISTA CD 50. Pour compléter cela, on présente dans la figure 3.2.3 la norme du tenseur d pour ADMM et FISTA dans les cas $h = 2$, $\delta = 1/2$, $B = 5$ et $h = 2$, $\delta = 1/4$, $B = 50$. On observe bien que pour le premier cas test, les délimitations de plug sont parfaitement nettes, bien que différentes entre ADMM et FISTA. Pour le second cas test, la figure 3.2.3d montre que le résultat de FISTA CD 4 est extrêmement bruité à proximité des bords de plug. La figure 3.2.3f qui présente le résultat de FISTA CD 50 est déjà de bien meilleure qualité, bien qu'une observation attentive montre que certaines délimitations ne sont pas parfaitement nettes. A contrario, le champ obtenu via ADMM est parfaitement net. La même figure permet d'observer que, malgré les instabilités de FISTA, les zones de pseudo-plugs (i.e. lorsque la norme de d est très faible mais non nulle, de l'ordre de 10^{-6}) sont effectivement trouvées par FISTA et ne se confondent pas avec les véritables plugs, comme cela peut être le cas pour des méthodes dont la résolution ne serait pas assez poussée.

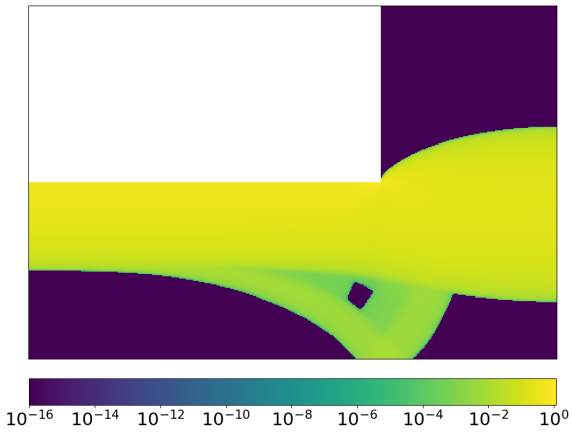
Convergence des résidus Pour essayer de comprendre ces instabilités, on observe les courbes de convergence du résidu. La figure 3.2.4 montre la convergence des résidus des différentes méthodes pour différents cas test. On constate immédiatement que les résidus de FISTA ne convergent pas tous vers 0. Le cas $h = 2$, $\delta = 1/4$, $B = 20$ qui présente des plugs bruités (figure 3.2.2b) pour FISTA donne aussi des résidus qui ne convergent pas, on observe même qu'ils



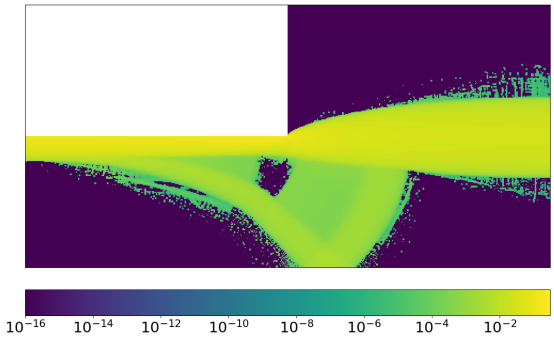
(a) $h = 2$, $\delta = 1/2$, $B = 5$, ADMM



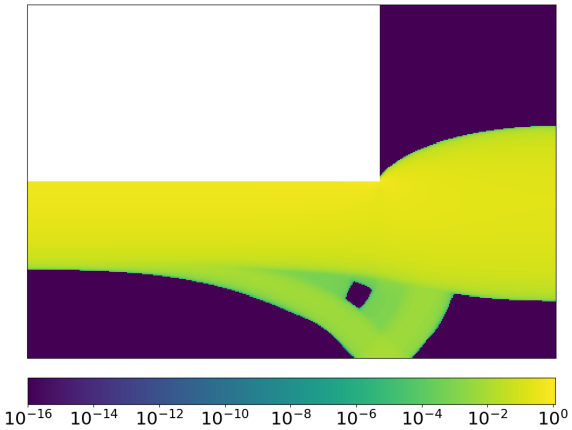
(b) $h = 2$, $\delta = 1/4$, $B = 50$, ADMM



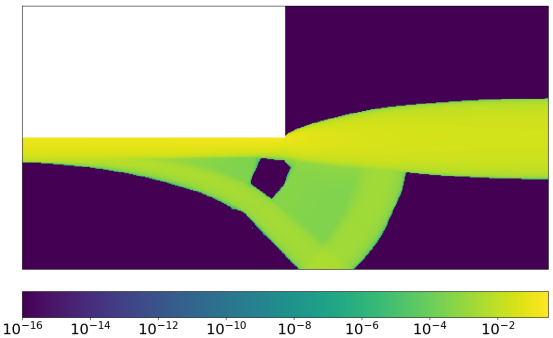
(c) $h = 2$, $\delta = 1/2$, $B = 5$, FISTA CD 4



(d) $h = 2$, $\delta = 1/4$, $B = 50$, FISTA CD 4



(e) $h = 2$, $\delta = 1/2$, $B = 5$, FISTA CD 50



(f) $h = 2$, $\delta = 1/4$, $B = 50$, FISTA CD 50

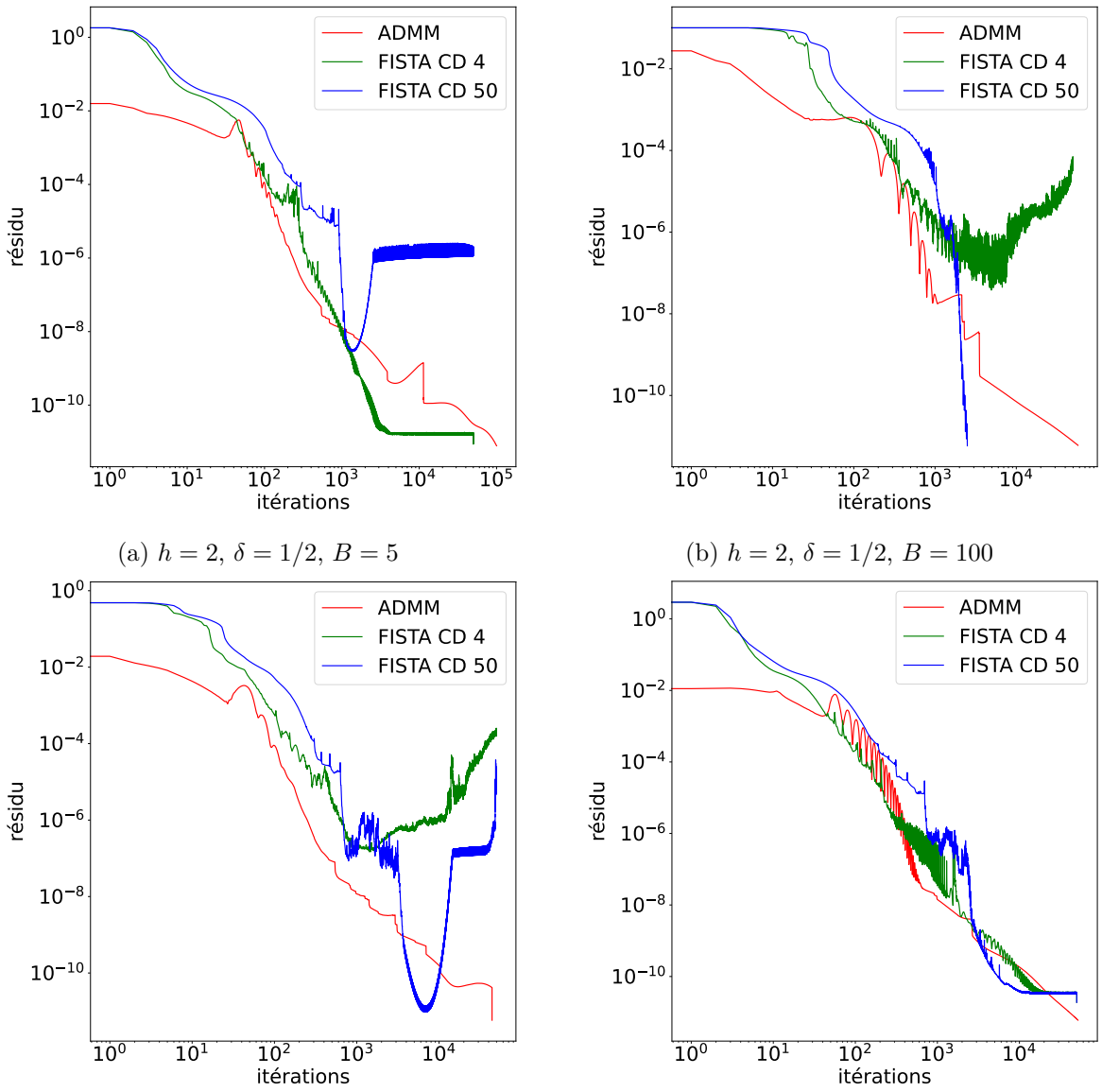
FIGURE 3.2.3 – Norme du tenseur d pour ADMM (ligne 1), FISTA CD 4 (ligne 2) et FISTA CD 50 (ligne 3). La colonne de gauche donne le cas test $h = 2$, $\delta = 1/2$, $B = 5$ tandis que celle de droite correspond à $h = 2$, $\delta = 1/4$, $B = 50$. L'échelle de couleur est logarithmique. Pour la lisibilité, seul le quart en haut à gauche du domaine est représenté, mais les calculs sont réalisés sur le domaine complet.

remontent de manière significative avant d'être coupés par le maximum de 50 000 itérations, aussi la méthode semble-t-elle diverger. Si on observe les figures (a), (b) et (c) de 3.2.4, on constate que la divergence d'une variante de FISTA n'implique pas celle de l'autre : on peut avoir FISTA CD 4 qui diverge mais FISTA CD 50 qui converge et vice-versa. On remarque aussi que, dans la plupart des cas, le résidu oscille significativement plus avec FISTA qu'avec ADMM, ce qui peut suggérer des débuts d'instabilité.

Par ailleurs, on observe que bien que toutes les courbes associées à FISTA ne soient pas divergentes, une grande majorité d'entre elles (y compris dans les courbes qui ne sont pas montrées ici) subissent une sorte de *saturation* en précision. Là où ADMM est en mesure de descendre le résidu à 6×10^{-12} (et même en dessous), une grande proportion des courbes de convergence de FISTA (parmi celles qui convergent) finissent par former un plateau à partir d'une certaine valeur du résidu. Bien que la valeur de ce plateau dépende du cas test (i.e. de la combinaison géométrie - rhéologie), elle ne semble pas dépendre de la valeur du paramètre a. En fait, la version non-accélérée ISTA a été testée sur de nombreux cas afin d'écarter l'influence de l'accélération sur le phénomène. Ce plateau est aussi observé dans ce cas-là, avec la même valeur que pour les variantes de FISTA (l'algorithme est cependant significativement plus lent). Ce phénomène est difficile à interpréter, une hypothèse est qu'il serait lié à une potentielle incompatibilité entre ISTA et la discrétisation choisie, incompatibilité qui reste à déterminer. On note que dans les cas où le plateau est atteint, on obtient généralement des délimitations de plug cohérentes (comme pour la figure 3.2.2d), aussi on peut penser que cette saturation n'entame pas trop la qualité du résultat (le plateau se trouve généralement à un résidu de l'ordre de 10^{-10} ou 10^{-11}, ce qui représente déjà une grande précision). Cependant, on rappelle qu'il a été observé plus haut que les plugs trouvés par FISTA sont différents de ceux trouvés par ADMM, cette saturation pourrait donc être le signe d'un défaut de convergence de FISTA qui expliquerait alors cette différence.

Remarque 43 *La figure 3.2.4 répertorie différents comportements qui ont été observés dans les courbes de résidu. Il convient de noter qu'en pratique, environ la moitié des courbes obtenues sont plutôt similaires à celles de la figure 3.2.4d, c'est-à-dire qu'elles convergent généralement jusqu'à atteindre la zone de saturation, autrement dit il y a de nombreux cas dans lesquels le résidu n'explose pas.*

Remarque 44 *On rappelle que le résidu est donné par $\|d - Du\|$. Au vu du défaut de convergence du résidu observé pour FISTA, il est légitime de se demander si utiliser $\varepsilon = 10^{-10}$ pour caractériser les plugs est bien pertinent, ou même s'il serait plus prudent d'utiliser directement Du. Ces deux possibilités ont été testées. Utiliser Du au lieu de d n'a que peu d'impact sur la délimitation des zones rigides. De même, monter ε jusqu'à 10^{-5} (ce qui fournirait déjà un seuil élevé) n'a produit que peu d'améliorations sur le caractère bruité de ces zones. On choisit donc de s'en tenir ici à la présentation du traitement suggéré par [157, 158].*



(a) $h = 2$, $\delta = 1/2$, $B = 5$

(b) $h = 2$, $\delta = 1/2$, $B = 100$

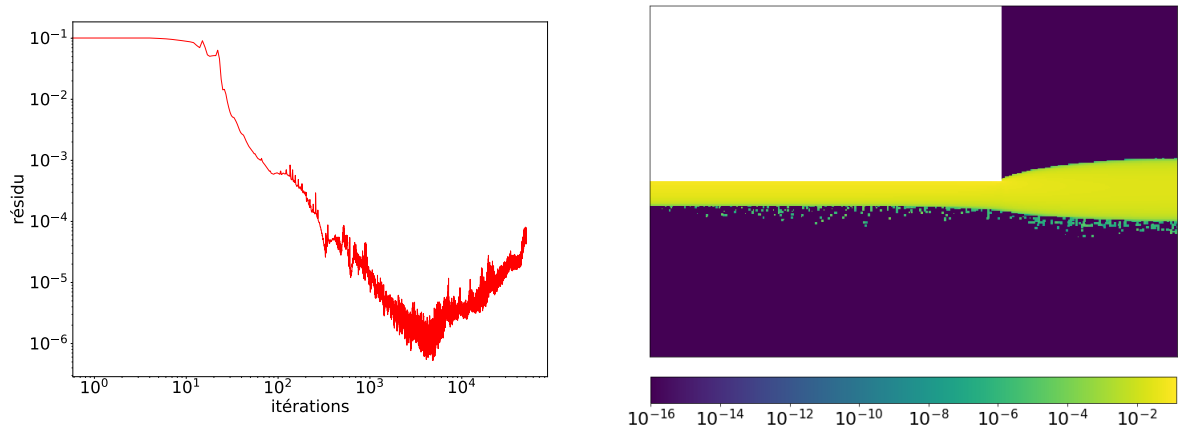
(c) $h = 2$, $\delta = 1/4$, $B = 20$

(d) $h = 6/5$, $\delta = 5/6$, $B = 5$

FIGURE 3.2.4 – Évolution du résidu au cours des itérations pour ADMM, FISTA CD 4 et FISTA CD 50 pour différents paramètres. La figure (a) montre que FISTA CD 4 peut converger alors que FISTA CD 50 ne converge pas. La figure (b) montre le phénomène opposé. La figure (c) montre un résidu qui semble converger pour finalement diverger. La figure (d) montre un cas où tous les résidus convergent. Dans les cas (a) et (d), les instances de FISTA qui convergent semblent subir une sorte de saturation en précision : le résidu minimal est bloqué à une certaine valeur. En figure (b), FISTA CD 50 est en mesure de descendre à 6×10^{-12}.

3.2.3.2 Influence du schéma d'accélération de FISTA

Schéma de Nesterov Vérifions l'influence du schéma d'accélération sur ces observations. La figure 3.2.5 montre un exemple de résultats obtenus avec FISTA BT, i.e. le schéma (2.77) originellement proposé par Nesterov. On constate que le résidu diverge assez rapidement, ne descendant pas en dessous de 10^{-6}, ce qui se traduit comme précédemment par des délimitations de plug très bruitées. L'écrasante majorité des cas tests utilisant ce schéma donnent le même résultat. Il est clair que le problème d'instabilité observé plus tôt est d'autant plus fort pour FISTA BT, le rendant inutilisable pour des calculs ayant vocation à être précis, on le retire de la suite de l'étude.

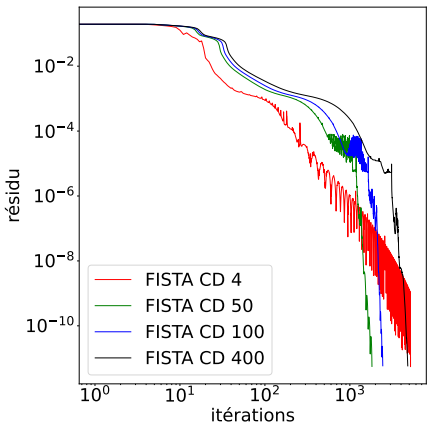


(a) Évolution du résidu (b) $\|d\|$

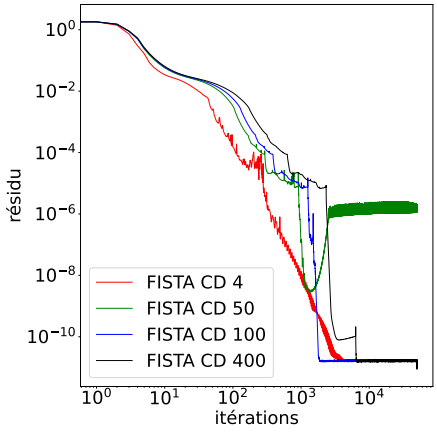
FIGURE 3.2.5 – Évolution du résidu (a) et norme de d (b) pour l'algorithme FISTA BT. Les résidus remontent à partir de 10^{-6} et les délimitations de plug sont bruitées.

Autres valeurs de a On revient donc sur le schéma de Chambolle et Dossal (2.144) et on fait varier le paramètre a. Les figures 3.2.6 et 3.2.7 montrent des courbes de convergence de résidu ainsi que les délimitations de plug sur différents cas tests pour quatre valeurs différentes de a, à savoir 4, 50, 100 et 400.

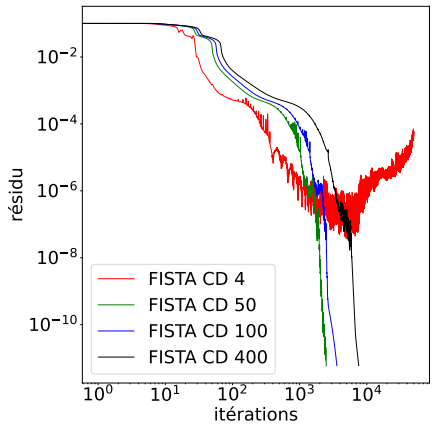
Pour commencer, la figure 3.2.6a illustre l'impact général de la variation de a : plus a est important, plus l'évolution du résidu est stable, mais plus ce dernier baisse lentement. En effet, on observe de fortes oscillations pour FISTA CD 4, mais presque aucune pour FISTA CD 400. En pratique, bien que la plupart des cas tests vérifient cette assertion, on observe des exceptions. Si on observe la figure 3.2.6b, on voit que seul le résidu associé à FISTA CD 50 remonte, là où les autres descendent jusqu'à la valeur de saturation évoquée précédemment. Sur le même principe, sur la figure 3.2.7b, seul FISTA CD 100 converge, là où FISTA CD 400 ainsi que les autres versions divergent. Il convient de noter que l'on parle ici de la convergence des résidus, qui n'est pas strictement équivalente à celle des plugs. En effet, sur la figure 3.2.7b, qui montre les limites de plug sur le même cas test que la figure précédente, on observe que seul FISTA CD 4 donne une limite de plug floue. On observe le même phénomène si l'on compare les résidus de la figure 3.2.6e et les limites de la figure 3.2.6f, où on s'attendrait à ce que FISTA CD 400 produisent des limites de basse qualité. La figure 3.2.7d montre cette fois des limites bruitées pour toutes les versions de FISTA, en accord avec les résidus de la figure 3.2.7c. Le phénomène complètement opposé peut aussi se produire, la figure 3.2.7e montre des résidus convergeant dans tous les cas alors que dans la figure 3.2.7f, FISTA CD 100 montre quelques perturbations non-négligeables dans les limites de plug.



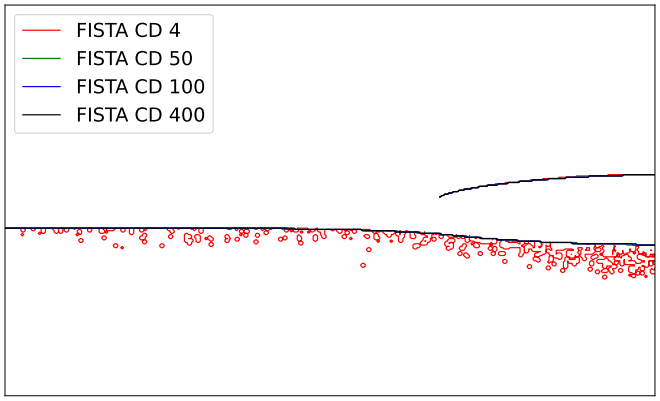
(a) Résidu pour $h = 2$, $\delta = 1/2$, $B = 50$



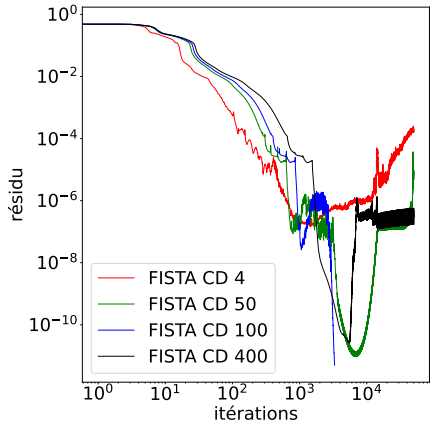
(b) Résidu pour $h = 2$, $\delta = 1/2$, $B = 5$



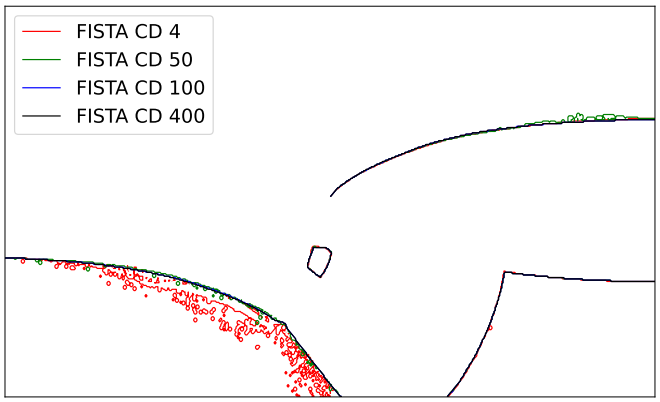
(c) Résidu pour $h = 2$, $\delta = 1/2$, $B = 100$



(d) Délimitations de plug pour $h = 2$, $\delta = 1/2$, $B = 100$

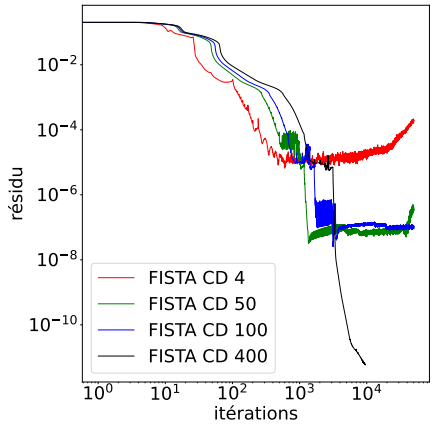


(e) Résidu pour $h = 2$, $\delta = 1/4$, $B = 20$

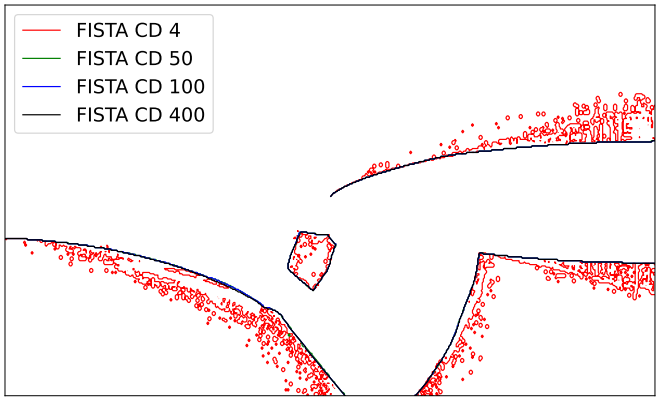


(f) Délimitations de plug pour $h = 2$, $\delta = 1/4$, $B = 20$

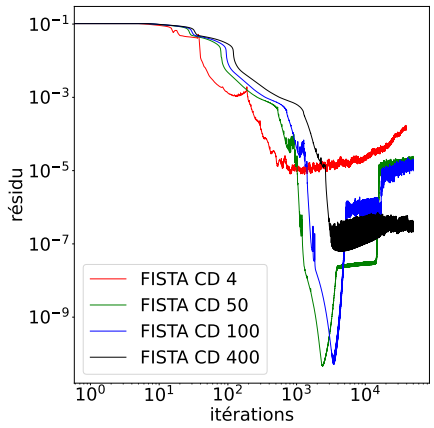
FIGURE 3.2.6 – Courbes de convergence du résidu et délimitations de plug données par FISTA CD 4, 50, 100 et 400 pour différents cas tests. Les délimitations de plug correspondant à la figure (b) pour FISTA CD 4 et 50 sont donnés par la figure 3.2.2a. Celles correspond à la figure (a) sont régulières et ne sont pas montrées ici.



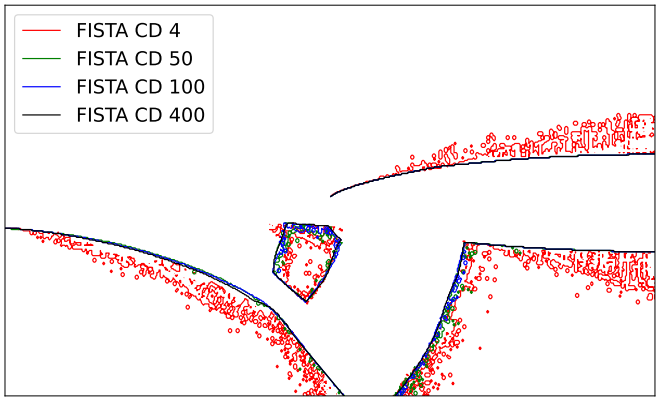
(a) Résidu pour $h = 2$, $\delta = 1/4$, $B = 50$



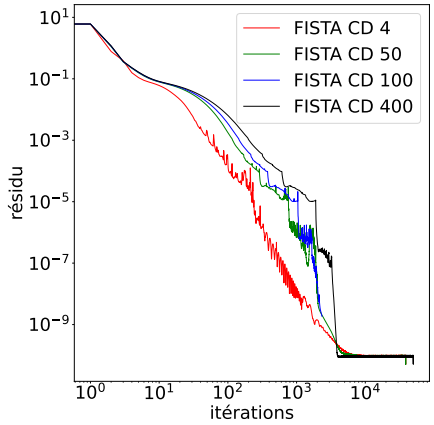
(b) Délimitations de plug pour $h = 2$, $\delta = 1/4$, $B = 50$



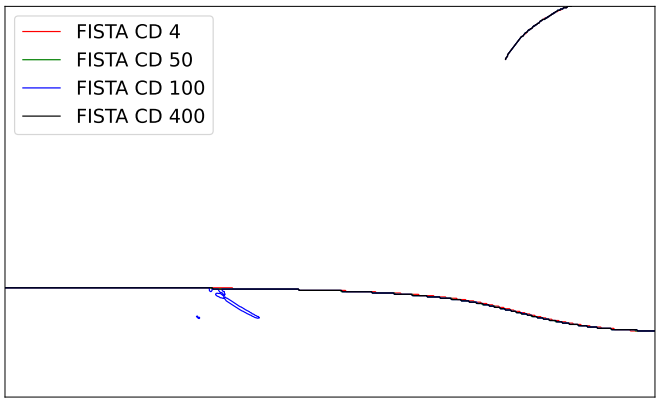
(c) Résidu pour $h = 2$, $\delta = 1/4$, $B = 100$



(d) Délimitations de plug pour $h = 2$, $\delta = 1/4$, $B = 100$



(e) Résidu pour $h = 6/5$, $\delta = 5/6$, $B = 2$



(f) Délimitations de plug pour $h = 6/5$, $\delta = 5/6$, $B = 2$

FIGURE 3.2.7 – Courbes de convergence du résidu et délimitations de plug données par FISTA CD 4, 50, 100 et 400 pour différents cas tests.

Il convient de rappeler que bien que les limites de plug semblent nettes dans certains cas, elles ne sont pas pour autant correctes. En particulier, dans la figure 3.2.7b, une observation attentive des limites de plug fournies par FISTA CD 400 permet de remarquer quelques "creux" dans le plug du canal d'entrée, ce qui semble peu réaliste d'un point de vue physique.

Remarque 45 *L'effet stabilisant que l'on observe est cohérent avec la formule* (2.144) *qui permet de calculer* $t^{(k)}$. *En effet, pour a grand,* $t^{(k)}$ *est plus proche de* 1 *et l'effet de l'accélération est donc amoindri.*

Procédure de restart Pour finir, on essaie d'incorporer la procédure de restart basée sur le critère (3.61). L'objectif étant de stabiliser le schéma sans perdre la vitesse, on applique cette procédure pour $a = 4$ et $a = 50$, autrement dit les deux schémas les plus rapides mais les moins stables. On présente dans la figure 3.2.8 différentes courbes de convergence de résidus comparant les résultats pour $a = 4$ et $a = 50$, avec et sans restart. L'effet stabilisant est bien plus marqué avec la procédure de restart que simplement en augmentant a. La figure 3.2.8b montre que le restart a permis au schéma de converger avec $a = 4$ alors qu'il divergeait auparavant. De manière plus générale, on constate comme dans la figure 3.2.8e une réduction des oscillations du résidu. Dans la grande majorité des cas, la procédure de restart a permis de faire converger les cas qui divergent sans restart, avec une amélioration conjointe de la délimitation des plugs (qui deviennent alors similaires à ceux présentés précédemment dans les cas qui convergent, on ne les montre donc pas). On note malgré tout quelques exceptions. Le cas 3.2.8a montre un exemple dans lequel la procédure avec $a = 50$ a permis d'améliorer partiellement la convergence mais au prix d'oscillations conséquentes. La figure 3.2.8f montre un cas où les restarts ont augmenté les oscillations du résidu, initialement convergent. Dans les figures 3.2.8c et 3.2.8d, on trouve deux cas où la procédure de restart a échoué à stabiliser le cas $a = 4$ ou $a = 50$. Cependant ces cas restent minoritaires parmi l'ensemble des cas tests. On observe étonnamment qu'avec la procédure de restart, le schéma obtenu avec $a = 4$ est plus stable qu'avec $a = 50$, ce qui est l'opposé du cas sans restart. Enfin, on remarque que la vitesse de convergence n'est généralement pas réduite par la procédure de restart, elle est même parfois accélérée.

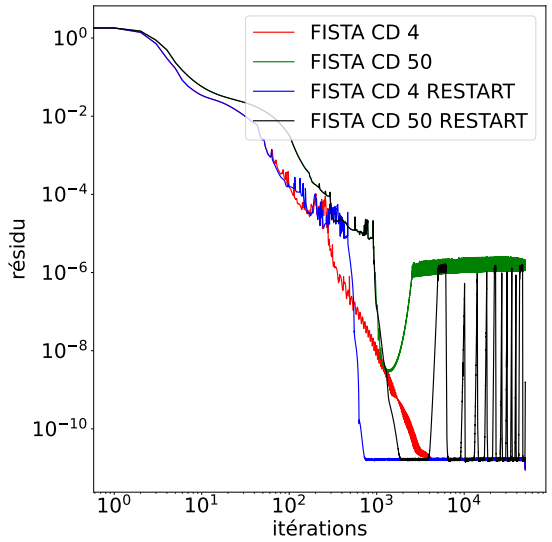
Vitesse des algorithmes Pour aller plus loin dans l'analyse de la vitesse, on présente dans le tableau 3.2.1 le nombre d'itérations requises pour atteindre une valeur donnée du résidu pour ADMM et 3 variantes de FISTA, dans plusieurs cas tests.

On constate que bien souvent, les résidus les plus bas ne sont pas atteints par FISTA CD 4, alors que FISTA CD 4 RESTART et FISTA CD 400 permettent de les atteindre presque systématiquement (et ADMM permet toujours de les atteindre). Lorsque les résidus donnés sont atteints, FISTA CD 4 est généralement le plus rapide, suivi par sa variante avec restart. FISTA CD 400 est le plus souvent plus lent que les deux précédents, en terminant avec ADMM. Cela correspond bien à ce qui a pu être observé sur les graphes de convergence des résidus présentés avant. La différence de vitesse entre ADMM et FISTA peut être conséquente. Par exemple, pour h=2, δ=1/2 et B=20, il faut plus de 35 000 itérations à ADMM pour atteindre un résidu de 6×10^{-12} contre moins de 700 itérations pour FISTA CD 4 RESTART. En résumé, il semble que lorsqu'une variante de FISTA converge, cela se produise à une vitesse significativement supérieure à ADMM, parfois de plusieurs ordres de grandeur.

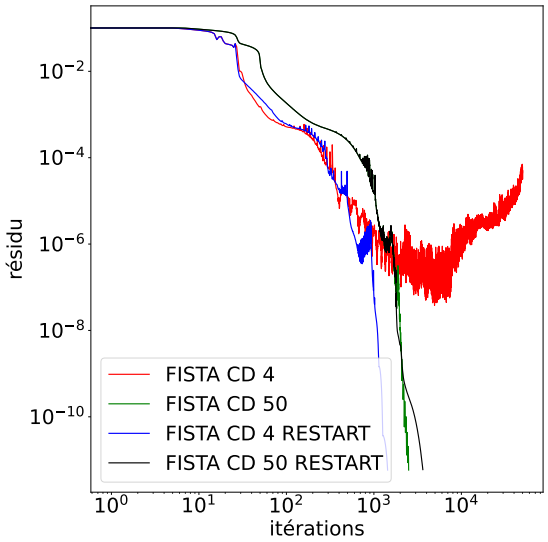
Il convient de noter que l'ordre de vitesse présenté ci-dessus n'est pas systématique. Pour h=6/5, δ=5/6 et B=100 par exemple, FISTA CD 400 est plus rapide que FISTA CD 4. On trouve aussi quelques rares cas où ADMM est plus rapide que toutes les variantes de FISTA. Ces cas restent toutefois minoritaires.

cas test	Résidu	ADMM	FISTA CD 4	FISTA CD 4 RESTART	FISTA CD 400
h=2,δ=1/2, B=2	3.9e-11	29359	44503	29477	1902
	6.0e-12	50970	-1	-1	-1
h=2,δ=1/2, B=20	3.2e-08	772	5023	461	3189
	6.0e-12	35790	-1	665	7742
h=2,δ=1/4, B=20	1.5e-07	384	1432	466	1798
	2.6e-11	44031	-1	868	5504
	6.0e-12	44045	-1	950	-1
h=2,δ=1/4, B=50	6.6e-06	218	604	612	2419
	1.9e-09	3326	-1	1060	3925
	6.0e-12	12366	-1	-1	9501
h=6/5,δ=5/6, B=2	8.8e-11	14051	10770	9835	4027
	6.0e-12	34370	-1	-1	-1
h=6/5,δ=5/6, B=100	1.2e-11	46146	23285	5083	13619

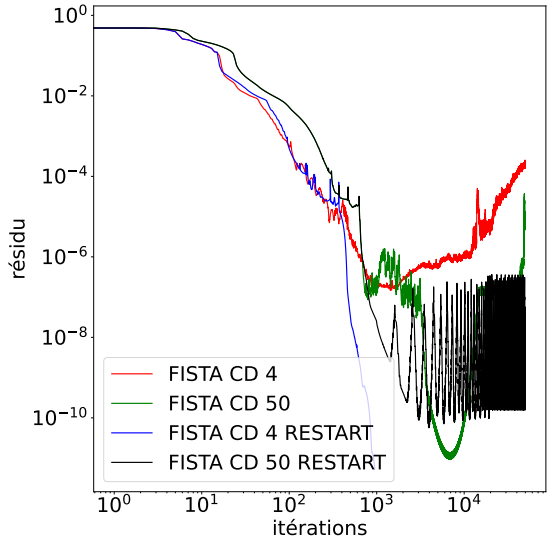
TABLE 3.2.1 – Tableau comparatif des performances de ADMM, FISTA CD 4, FISTA CD 4 RESTART et FISTA CD 400 sur différents cas tests. Pour chaque cas test, on donne une (ou plusieurs) valeur du résidu, puis, pour chaque algorithme, le nombre d'itérations qu'il a été nécessaire d'accomplir pour atteindre ce résidu. Lorsque l'algorithme n'atteint pas le résidu donné, on note -1.



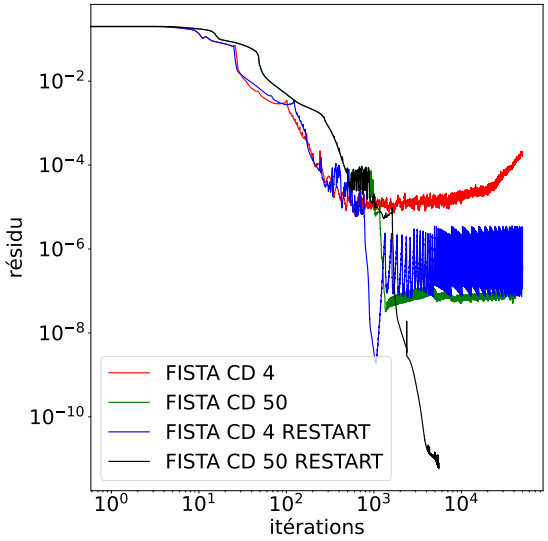
(a) $h = 2$, $\delta = 1/2$, $B = 5$



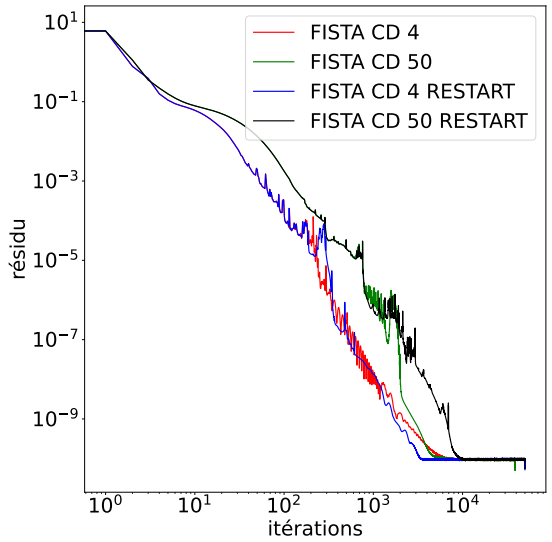
(b) $h = 2$, $\delta = 1/2$, $B = 100$



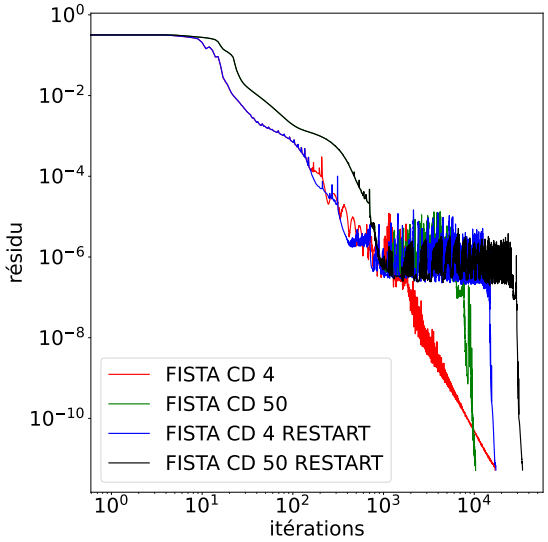
(c) $h = 2$, $\delta = 1/4$, $B = 20$



(d) $h = 2$, $\delta = 1/4$, $B = 50$



(e) $h = 6/5$, $\delta = 5/6$, $B = 2$



(f) $h = 6/5$, $\delta = 5/6$, $B = 50$

FIGURE 3.2.8 – Évolution du résidu au cours des itérations pour FISTA CD 4 et FISTA CD 50, avec et sans restart, pour différents paramètres.

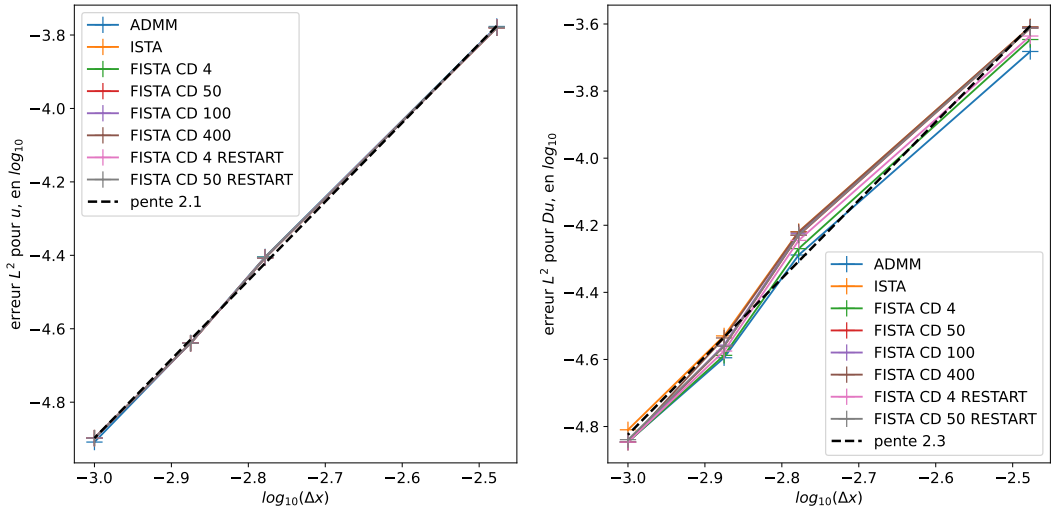
3.2.4 Convergence en maillage

Une étude de convergence en maillage est toujours une étape indispensable pour scruter le comportement des méthodes numériques. Ce type d'étude revêt ici un angle d'analyse supplémentaire, du fait des phénomènes d'instabilité mais aussi de saturation du résidu de FISTA décrits précédemment. Un raffinement de maillage change-t-il le comportement de la convergence des solutions approchées ? Devant l'ampleur des instabilités de FISTA BT (qui n'a, en pratique, presque jamais convergé), cette variante est éliminée de l'étude qui suit.

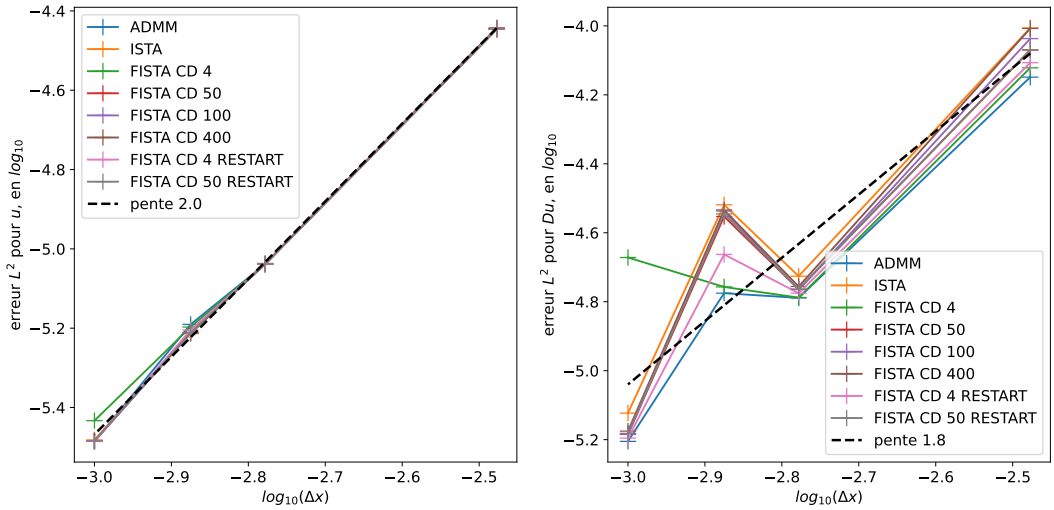
Cas Poiseuille Dans un premier temps, on s'intéresse à l'écoulement de Poiseuille qui nous permet de comparer les résultats avec la solution analytique du problème. Pour cela, le problème de Bingham est résolu par chaque algorithme pour différentes valeurs du pas Δx. On prend $\Delta x = 1/300; 1/600; 1/750; 1/1000$. La procédure est répétée pour $B = 1; 10; 100$. On présente en figure 3.2.9 l'évolution de la différence entre le résultat de chaque méthode et la formule analytique de Poiseuille, en norme H^1.

La norme L^2 de la vitesse semble converger de manière similaire entre les différents algorithmes. Concernant la convergence de Du, la situation est moins claire. En effet, pour $\Delta x = 1/750$ et $B = 10; 100$, plusieurs variantes de la méthode FISTA voient leur erreur augmenter. Le résidu redescend ensuite pour $\Delta x = 1/1000$. On remarque que pour B=10 (figure 3.2.9b), l'erreur commise par FISTA CD 4 semble remonter significativement, confirmant encore une fois qu'il s'agit du schéma le plus instable. Mise à part cette exception, les différentes variantes de FISTA se comportent de manière similaire. Par ailleurs, la pente moyenne des courbes a été calculée dans chaque cas. On obtient une pente d'ordre 2 ou presque pour l'erreur en u et en Du, pour les 3 valeurs de B utilisées, ce qui confirme la convergence en maillage d'ordre 2 fournie par la discrétisation. Il convient de noter que ces pentes sont calculées en utilisant tous les points d'un graphe donné (i.e. en agrégeant les données de toutes méthodes), ce qui n'a que peu d'influence sur les pentes. En calculant les pentes en utilisant par exemple uniquement ADMM, les pentes varient de moins de 0.1.

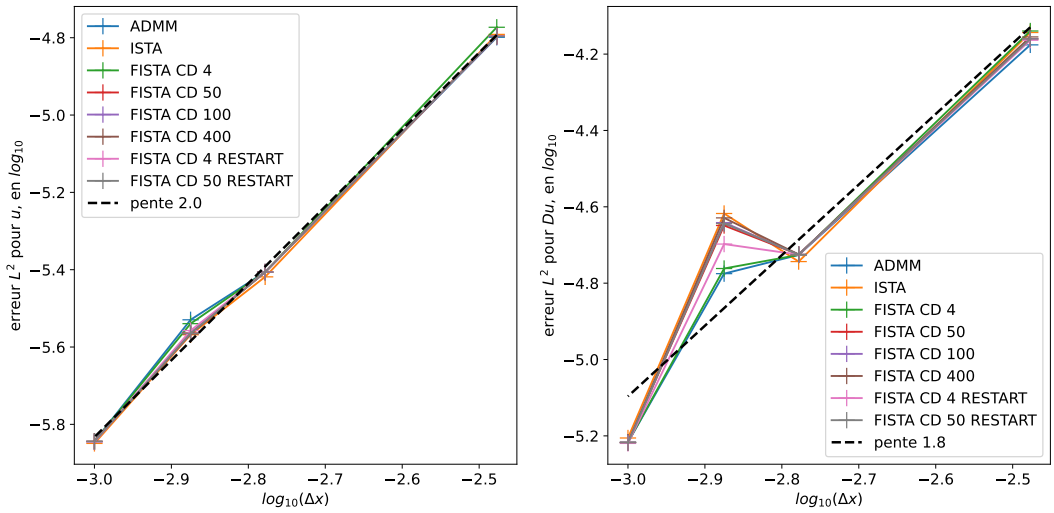
La figure 3.2.10 montre la convergence des résidus pour FISTA CD 4 RESTART en fonction de Δx. Augmenter le pas ne semble pas améliorer la convergence, que ce soit en stabilité ou en vitesse. On remarque que le phénomène de saturation en précision est toujours présent, il ne réduit pas avec Δx. Les autres cas donnent des résultats similaires et ne sont donc pas présentés.



(a) $B = 1$

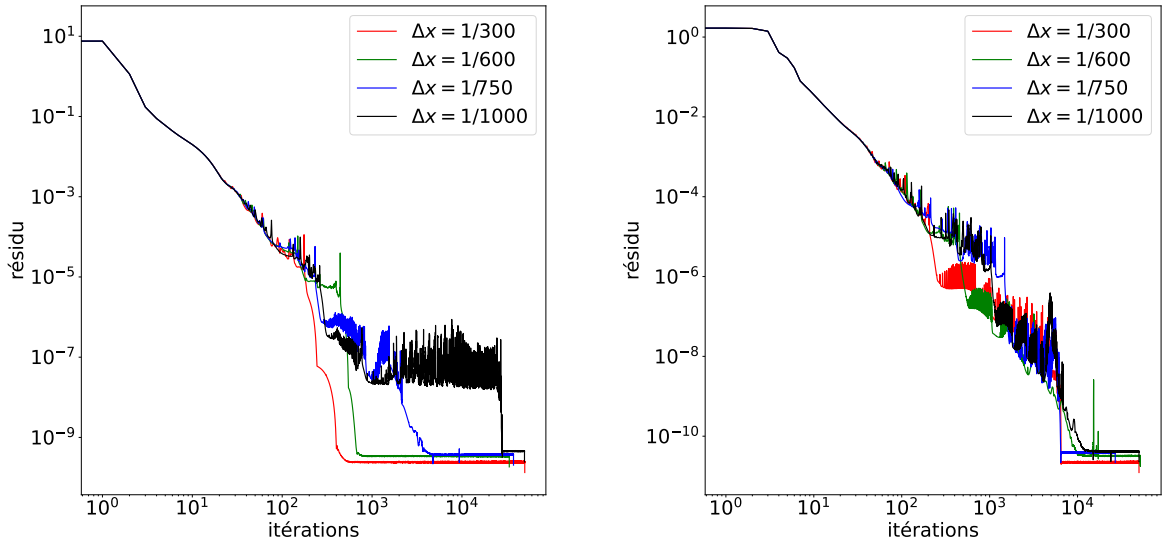


(b) $B = 10$



(c) $B = 100$

FIGURE 3.2.9 – Convergence sous raffinement de maillage des différents algorithmes (ADMM et différentes versions de FISTA) pour un écoulement de Poiseuille, pour plusieurs valeurs du nombre de Bingham B. Chaque graphe montre la pente moyenne fournie par les méthodes.



(a) $B = 1$ (b) $B = 10$

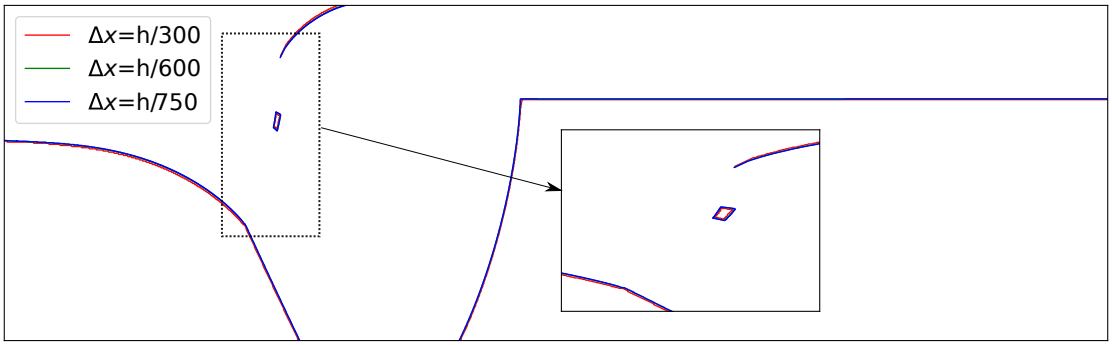
FIGURE 3.2.10 – Évolution du résidu au cours de la résolution de l'écoulement de Poiseuille par FISTA CD 4 RESTART pour différents B. Sur chaque figure sont comparés les résultats pour différentes valeurs de Δx.

Géométrie en croix La même procédure a été testée sur une géométrie en croix, en utilisant $h = 6/5$, $d = 1/12$ (il s'agit là d'une cavité longue) et $B = 20$. Le coût de calcul étant plus important que dans le cas Poiseuille (où il est possible de prendre un canal de petite longueur pour diminuer le nombre de mailles), on effectue cette étude pour $\Delta x = h/300; h/600; h/750$.

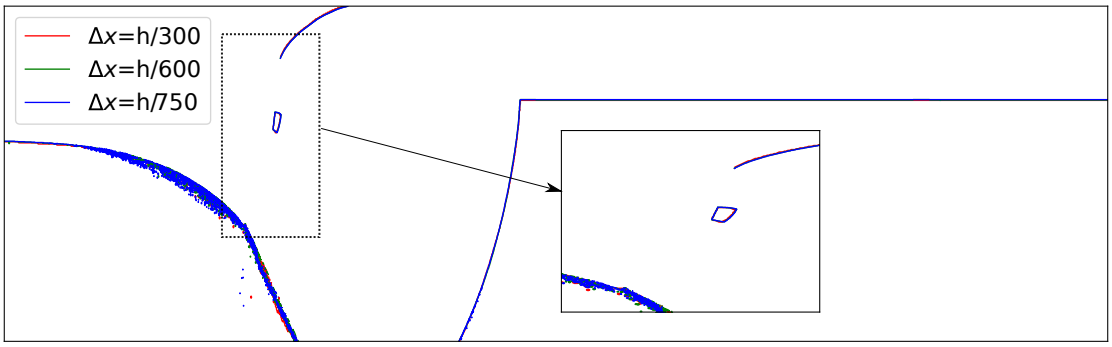
La figure 3.2.11 montre l'évolution des délimitations de plug obtenues par plusieurs algorithmes en fonction de Δx. Comme observé par Marly [157], les plugs déterminés par ADMM changent très peu entre les différents pas Δx utilisés ici (voir figure 3.2.11a). Si on observe les résultats de FISTA CD 4 (figure 3.2.11b), on constate que raffiner le pas d'espace tend à augmenter l'instabilité des délimitations de plug, ce qui est cohérent avec les observations faites dans le cas Poiseuille. Pour FISTA CD 400 et FISTA CD 4 RESTART (figures 3.2.11c, 3.2.11d) qui sont plus stables, on n'observe que peu de différences entre les résultats des différents maillages. Dans le cas FISTA CD 4 RESTART, qui ne montre ici aucune instabilité, on observe même moins de variations entre les différentes valeurs du pas que pour ADMM. On précise qu'une observation attentive des courbes de convergence de résidus a été effectuée, les conclusions sont identiques à celles que l'on peut formuler sur le cas Poiseuille.

La question de la convergence en maillage ne semble donc pas être la réponse à la question des instabilités de FISTA. Cette étude renforce néanmoins les observations précédentes quant à la stabilité significativement accrue par l'ajout de la procédure de restart.

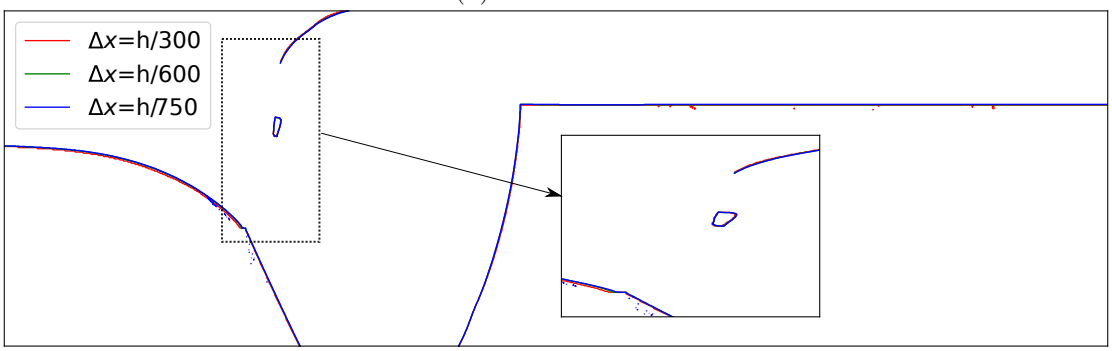
Remarque 46 *On rappelle que lors de l'introduction de la méthodologie FISTA pour les simulations d'écoulements viscoplastiques, Treskatis [237] a aussi testé la procédure de restart que l'on utilise ici, sans y voir d'influence notable. Cette différence de résultats peut s'expliquer par le fait que les écoulements utilisés par Treskatis étaient de nature plus simple que la géométrie en croix, les plugs produits par ces écoulements possédaient des topologies moins complexes. On ajoute aussi qu'ici, on tente de pousser la précision au maximum, ce qui permet de révéler certaines instabilités indétectables à plus basse précision. Enfin, Treskatis [237, 240] utilise des éléments finis $P1$-iso-$P2$ qui permettent aux algorithmes discrets et continus de coïncider sur le*



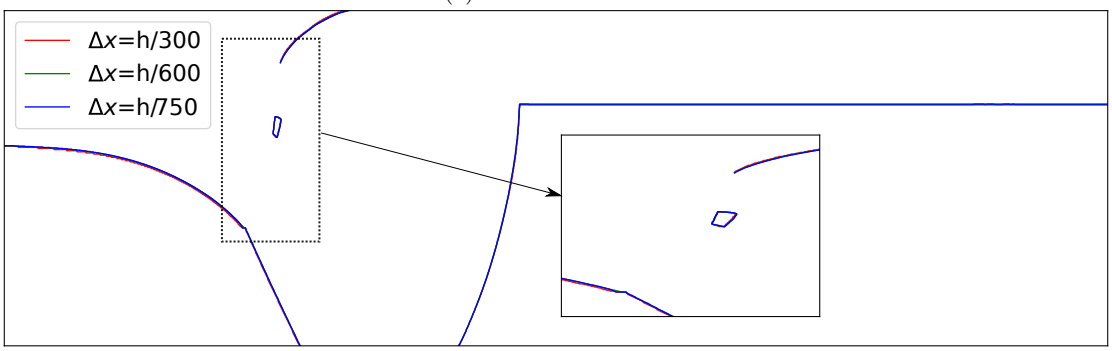
(a) ADMM



(b) FISTA CD 4



(c) FISTA CD 400



(d) FISTA CD 4 RESTART

FIGURE 3.2.11 – Comparaison des délimitations de plug du cas test $\Delta x = h/300; h/600; h/750$ pour des maillages de tailles différentes, pour plusieurs algorithmes.

problème de Bingham. Par voie de conséquence, le problème de Stokes linéaire (3.42) *peut être*

résolu de manière exacte. Ce n'est pas le cas en différences finies. Il s'ensuit que l'erreur sur la résolution du problème de Stokes ajoute aux instabilités de FISTA, ce qui peut expliquer tout ou partie des instabilités constatées dans ce travail.

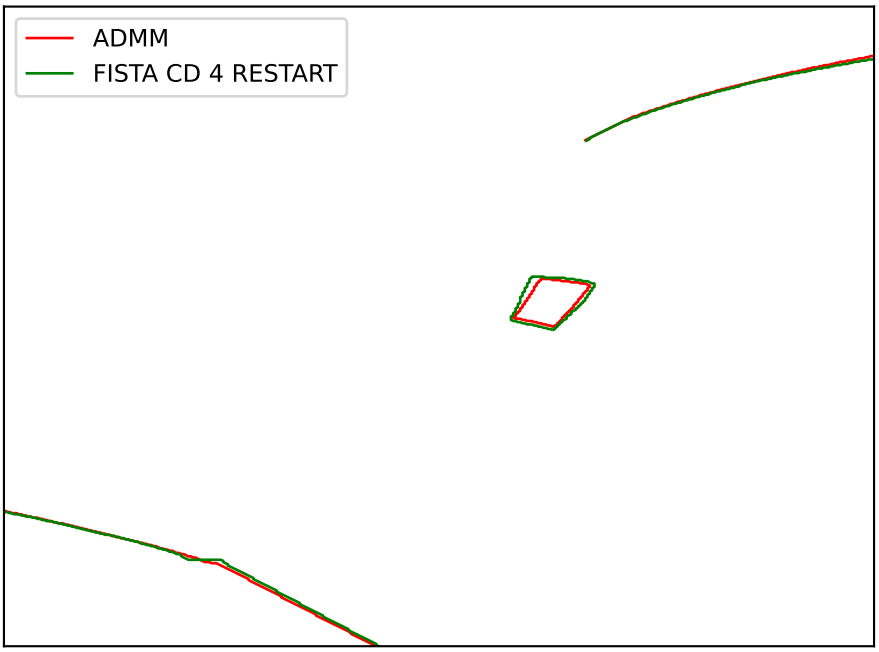
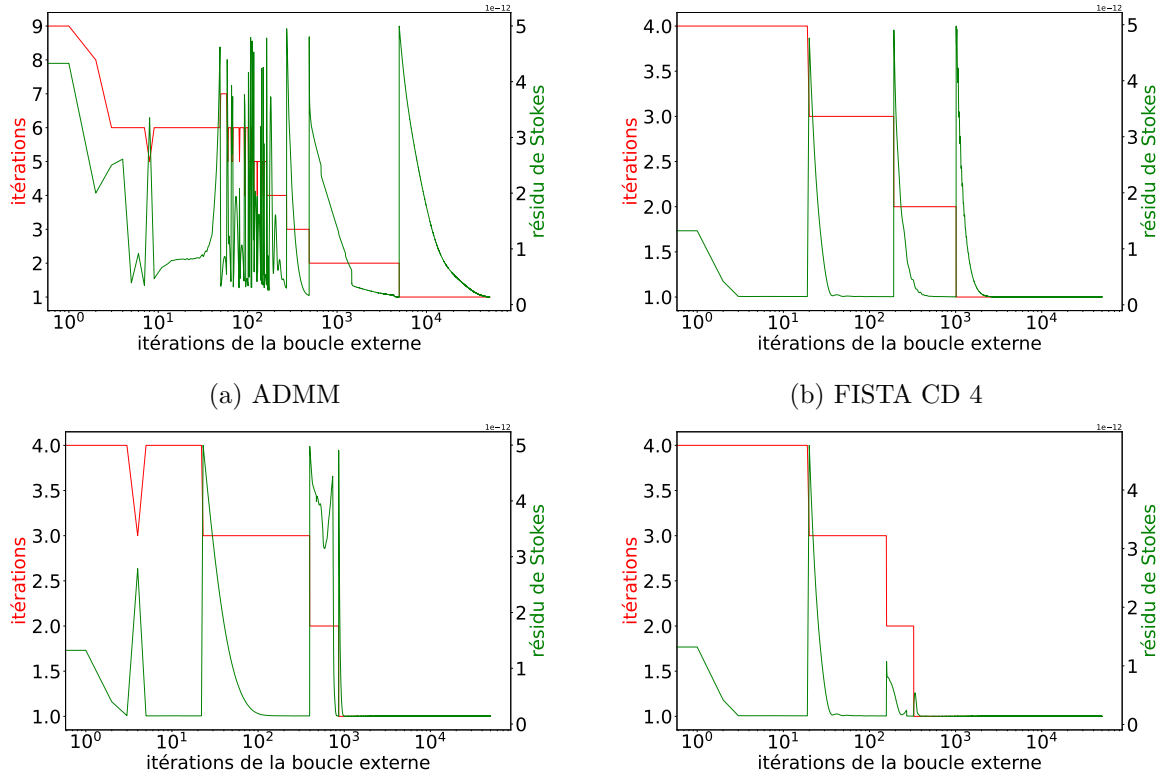


FIGURE 3.2.12 – Comparaison des limites de plug obtenues par ADMM et FISTA CD 4 RESTART sur le cas test $\Delta x = h/300; h/600; h/750$, pour $\Delta x = h/750$, avec un agrandissement sur le patch à l'entrée de la cavité.

Pour finir, on utilise le maillage le plus résolu ($\Delta x = h/750$) pour revenir sur les comparaisons de limites de plug entre ADMM et FISTA, qui avaient été évoquées plus tôt. Le résultat est illustré sur la figure 3.2.12 avec FISTA CD 4 RESTART, étant donné que c'est la version de FISTA qui semble fournir les meilleurs résultats. On peut constater deux choses. D'une part, les deux algorithmes délimitent un patch dont la forme est extrêmement similaire. Cette fois, la concavité du patch est en accord avec la théorie des sliplines de Balmforth *et al.*. D'autre part, en bas à gauche, FISTA montre une irrégularité physiquement peu cohérente. Il est donc toujours difficile de déterminer quelle méthode fournit les meilleures délimitations de plug, néanmoins les irrégularités récurrentes fournies par FISTA laissent penser que ADMM fournit les résultats les plus cohérents d'un point de vue physique.

3.2.5 Résolution de Stokes et mesure de la vitesse

Jusqu'à présent, on a utilisé le nombre d'itérations de la boucle externe pour caractériser la vitesse des algorithmes. Cependant, on rappelle que chaque itération de la boucle externe fait appel à l'algorithme 18 de résolution du sous-problème de Stokes, qui effectue la résolution de plusieurs systèmes linéaires. Ce sont précisément ces résolutions qui prennent la majeure partie du temps de l'algorithme (dans l'implémentation utilisée ici, cela concerne plus de 97% du temps effectif de calcul). L'hypothèse implicite qui a été faite jusqu'à présent est donc que le nombre d'itérations de la boucle externe et le nombre de résolutions de systèmes linéaires sont proportionnels.



(a) ADMM

(b) FISTA CD 4

(c) FISTA CD 100

(d) FISTA CD 4 RESTART

FIGURE 3.2.13 – Évolution des performances de l'algorithme 18 de résolution du sous-problème de Stokes au cours des itérations de la boucle externe, pour différents algorithmes. En rouge, on représente le nombre d'itérations de l'algorithme 18 (donc le nombre de résolutions de systèmes linéaires). En vert, on représente l'évolution du résidu de l'algorithme 18. On considère ici le cas d'une géométrie en croix avec h=2, $\delta = 1/2$ et $B = 2$.

Pour vérifier cela, on présente en figure 3.2.13 l'évolution du nombre d'itérations de l'algorithme 18 de résolution du Stokes (donc le nombre de résolutions de systèmes linéaires) au cours des itérations de la boucle externe. On constate que pour ADMM le nombre de résolutions descend rapidement aux environs de 6 ou moins, jusqu'à ne plus faire qu'une ou deux résolutions par itération. Dans le cas FISTA, ce nombre démarre directement plus bas, avec environ 4 résolutions dès le départ, puis il descend jusqu'à 1 ou 2 itérations comme ADMM. Il convient de noter que les figures sont en échelle logarithmique, aussi les méthodes effectuent moins de 3 ou 4 résolutions pendant la majeure partie de l'algorithme. Cela se vérifie aussi pour les autres géométries et valeurs de B. Pour confirmer cela, on reproduit en annexe 3.A plusieurs figures effectuées précédemment, en utilisant le nombre de résolutions de systèmes linéaires plutôt que le nombre d'itérations de la boucle externe. Précisément, la figure 3.A.1 reproduit la figure 3.2.4 qui présente l'évolution du résidu pour ADMM, FISTA CD 4 et 50 pour différentes configurations. La figure 3.A.2 reproduit la figure 3.2.8 qui présente l'impact de l'ajout de la procédure de restart. Enfin, le tableau 3.A.1 reproduit le tableau 3.2.1 qui donne le nombre d'itérations (ici de résolutions de systèmes linéaires) pour atteindre un résidu donné. À chaque fois, les figures ne présentent pratiquement aucune variation, les rapports de vitesses sont inchangés, seules les valeurs sont modifiées. Cela confirme donc la pertinence de l'analyse précédente.

On présente aussi sur la figure 3.2.13 l'évolution du résidu $\|\text{div}\left(u^{(k,i)}\right)\|_{L^2}$ de l'algorithme

18. Dans le cas FISTA, il semble être systématiquement largement inférieur au seuil 5×10^{-12} requis (ce qui n'est pas toujours vrai pour ADMM), ce qui est cohérent avec le fait que FISTA ait une tendance à générer moins de résolutions de systèmes linéaires par itération de la boucle externe.

3.3 Résultats numériques pour Herschel-Bulkley

3.3.0.1 Paramètres des tests

On ajoute maintenant à ces résultats des tests sur le cas Herschel-Bulkley. Au vu des difficultés évoquées précédemment sur la méthode FISTA, on s'en tient à l'algorithme ADMM 19. Pour les paramètres des différents tests, on s'inspire des expériences en laboratoire réalisées par Luu et al. [156], dans une configuration très similaire au-dessus d'une cavité. Ce cadre a fait ses preuves comme benchmark utile pour des comparaisons croisées entre simulations et expériences. On choisit 5 fluides différents dont les paramètres physiques sont donnés dans le tableau 3.3.1.

Fluide	K $(Pa.s^n)$	τ_y (Pa)	n
F1	1.1	0.2	0.45
F2	1.4	0.8	0.44
F3	3	3.8	0.41
F4	7.1	15.3	0.4
F5	9.6	21.3	0.4

TABLE 3.3.1 – Paramètres physiques des différents fluides étudiés pour les expériences Herschel-Bulkley.

Pour les différents fluides, on combine plusieurs valeurs du paramètre H (voir figure 3.1.1, on fixe D via la formule $H + 2D$ à 6.5 cm) et du gradient de pression, que l'on récapitule dans le tableau 3.3.2.

Pour finir, on fait varier la longueur L de la cavité intérieure. Pour chaque cas du tableau 3.3.2, on teste $L = 1.5, 3$ et 6. On a donc 87 cas au total.

Pour la résolution, on choisit Δx de sorte à avoir 1200 mailles dans une section verticale de la cavité centrale, afin d'obtenir une haute précision. On se fixe un maximum de 150 000 itérations de ADMM, mais dans la plupart des cas cette borne n'est pas atteinte pour des contraintes de temps imposées par le cluster (qui coupe alors la simulation). Malgré cela, la plupart des simulations ont atteint une convergence tout à fait convenable.

3.3.1 Résultats sur différents cas

La figure 3.3.1 montre l'évolution du résidu au cours des itérations pour différents cas. La convergence est plus lente que dans le cas Bingham, cependant les résidus convergent tous vers 0.

La figure 3.3.2 montre à titre d'exemple le résultat du cas $R16$, $L = 6$. Le résultat est d'excellente qualité. On observe des imperfections dans les plugs en entrée et en sortie, cependant l'échelle de couleur montre qu'elles sont d'ordre 10^{-10} (comparé au reste du plug d'ordre 10^{-15}), on peut donc considérer que cela reste une délimitation de plug fiable. En pratique, la plupart des cas tests montrent ce genre d'artefacts, avec des ordres de grandeur pouvant aller jusqu'à 10^{-7} (voir notamment la figure 3.3.3b). Ceci peut se produire alors même que le résidu a atteint une valeur de 10^{-12}. On peut formuler l'hypothèse que, devant la difficulté accrue du cas Herschel-Bulkley par rapport à Bingham, il faudrait baisser le paramètre r du Lagrangien augmenté afin que l'algorithme ne privilégie par trop fortement la baisse de $|d - Du|$ au détriment de la convergence des autres variables.

Numéro du cas	fluide	H (cm)	$-\nabla p$ $(Pa.m^{-1})$
R1	F1	1.5	50,92495
R2	F1	1.5	63,665301
R3	F1	3	98,597467
R4	F2	3	126,310309
R5	F2	3	158,25871
R6	F2	3	193,191235
R7	F3	1.5	241,331623
R8	F3	1.5	280,113244
R9	F3	1.5	306,832721
R10	F3	1.5	320,390821
R11	F3	1.5	345,104274
R12	F3	3	406,47373
R13	F3	3	471,542594
R14	F3	3	493,258956
R15	F3	3	526,779011
R16	F3	3	555,022488
R17	F4	1.5	933,674603
R18	F4	1.5	994,640589
R19	F4	1.5	1043,257194
R20	F4	3	1488,902546
R21	F4	3	1573,938101
R22	F4	3	1634,575782
R23	F5	1.5	1297,280653
R24	F5	1.5	1369,478689
R25	F5	1.5	1447,706208
R26	F5	3	2038,675076
R27	F5	3	2148,476115
R28	F5	3	2228,428421

TABLE 3.3.2 – Paramètres physiques des différents fluides mis à l'épreuve pour les expériences Herschel-Bulkley.

Parmi les 87 cas tests, une vingtaine fournit des résultats de mauvaise qualité. En particulier, le fluide F1 qui possède une faible valeur de seuil semble poser difficulté. La figure 3.3.3a montre le cas R3, L=1.5 où les plugs en entrée et en sortie ont l'air d'avoir disparu de $|d|$. Il convient de noter que la courbe de résidu correspondante est présentée en figure 3.3.1 et montre pourtant un résidu descendu à 10^{-10}, ce qui renforce la discussion précédente sur la nécessite d'étudier une réduction de r.

Pour finir, la figure 3.3.4 montre le nombre d'itérations faites par la méthode de Newton (Alg. 20) nécessaire à la résolution de (3.21) (on rappelle qu'il s'agit d'une résolution à faire en tous les points de l'espace discret). Le développement limité (3.34) est utilisé dans la majeure partie des plugs, ce qui permet de réduire drastiquement la quantité de calculs nécessaire. Dans tous les cas (cela a été vérifié sur l'ensemble des 87 cas tests), la méthode de Newton converge en moins de 9 itérations, permettant une résolution de (3.21) avec une erreur sur la solution inférieure à 10^{-15}. Cette étude montre donc la grande efficacité de cette méthode présentée dans [157].

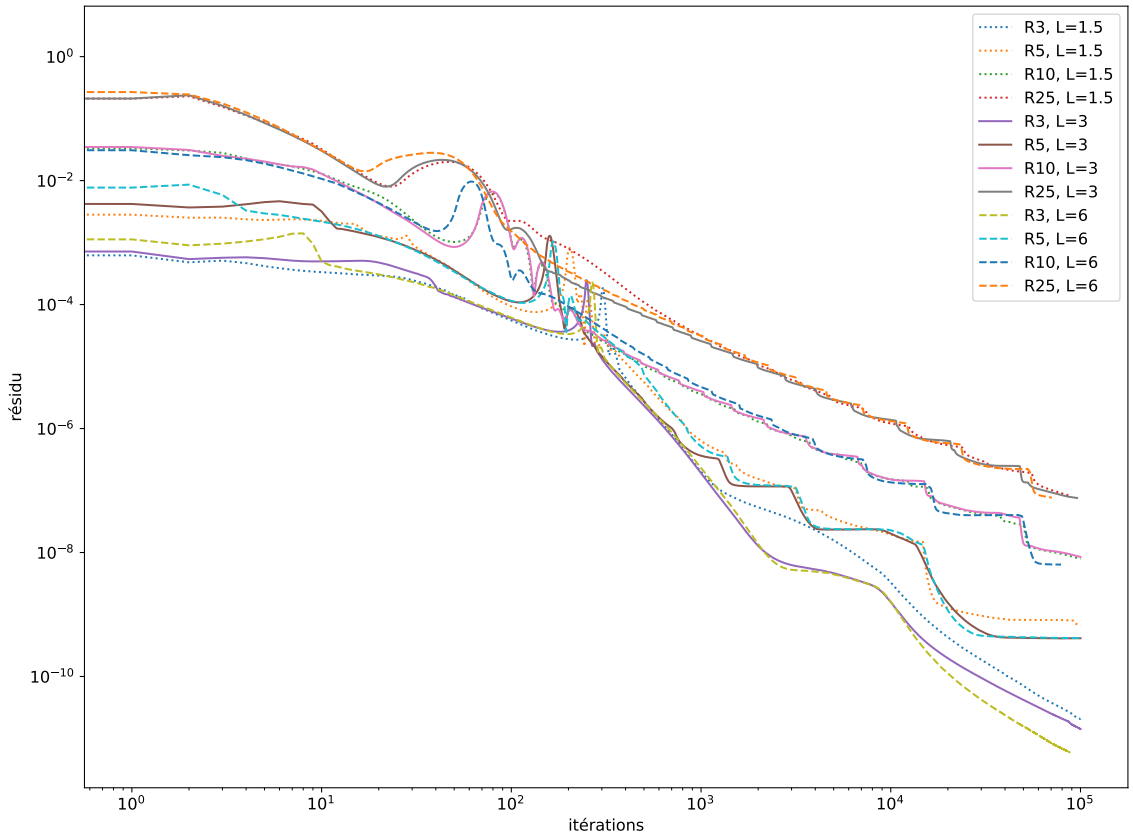


FIGURE 3.3.1 – Résidu au cours des itérations pour différents cas tests Herschel-Bulkley.

3.3.2 Étude préliminaire de couche limite

On utilise maintenant ces résultats pour procéder à des calculs de tailles de couche limite viscoplastique, dans la suite naturelle des travaux de la thèse de Marly [157]. Nous renvoyons à son manuscrit pour les définitions précises de ces couches limites et leur historique (à l'origine décrites par Oldroyd [186]) : lorsque l'on regarde une coupe transversale dans la cavité (voir figure 3.3.5), elles se situent dans les zones de matériau cisaillées et font la jonction entre la zone morte (en fond de cavité) et une zone (également cisaillée) qui vérifie un profil de Poiseuille [1], juste avant le bouchon central ([157], Fig. 3.28). Au vu des résultats présentés avant, il est ici nécessaire d'éliminer au préalable les simulations qui n'ont vraisemblablement pas convergé. On conserve les 45 simulations ayant le mieux convergé. Il s'agit donc actuellement d'une étude tout à fait préliminaire d'extraction des tailles de couches limites, notées δ_{BL}, pour étendre du cas Bingham au cas Herschel-Bulkley les résultats de [157] ([157], Figs 3.44 et 3.45).

On rappelle donc que les δ_{BL} sont obtenues de la manière suivante (voir figure 3.3.5) : on se place au milieu de la cavité en $x = x_{1/2}$ et on y considère une coupe des quantités disponibles $u(x_{1/2}, .)$ et $|Du|(x_{1/2}, .)$ dans la direction transverse, y. Tout d'abord, on localise le maximum de $|Du|(x_{1/2}, .)$ sur le profil 1D, cela fournit un $y := y_s$ qui localise la fin de la couche limite (quand on part de la zone morte). Remarquons que grâce à y_s, on récupère ensuite la quantité $u(x_{1/2}, y_s) := U_s$, dite "vitesse de glissement" qui rentre en jeu dans la définition du nombre sans dimension caractérisant la viscoplasticité. On parle du nombre de Bingham, ou plus précisément

1. en toute rigueur, il faut tenir compte de la loi puissance en $n \neq 1$, on devrait donc dire "Poiseuille-Herschel-Bulkley"

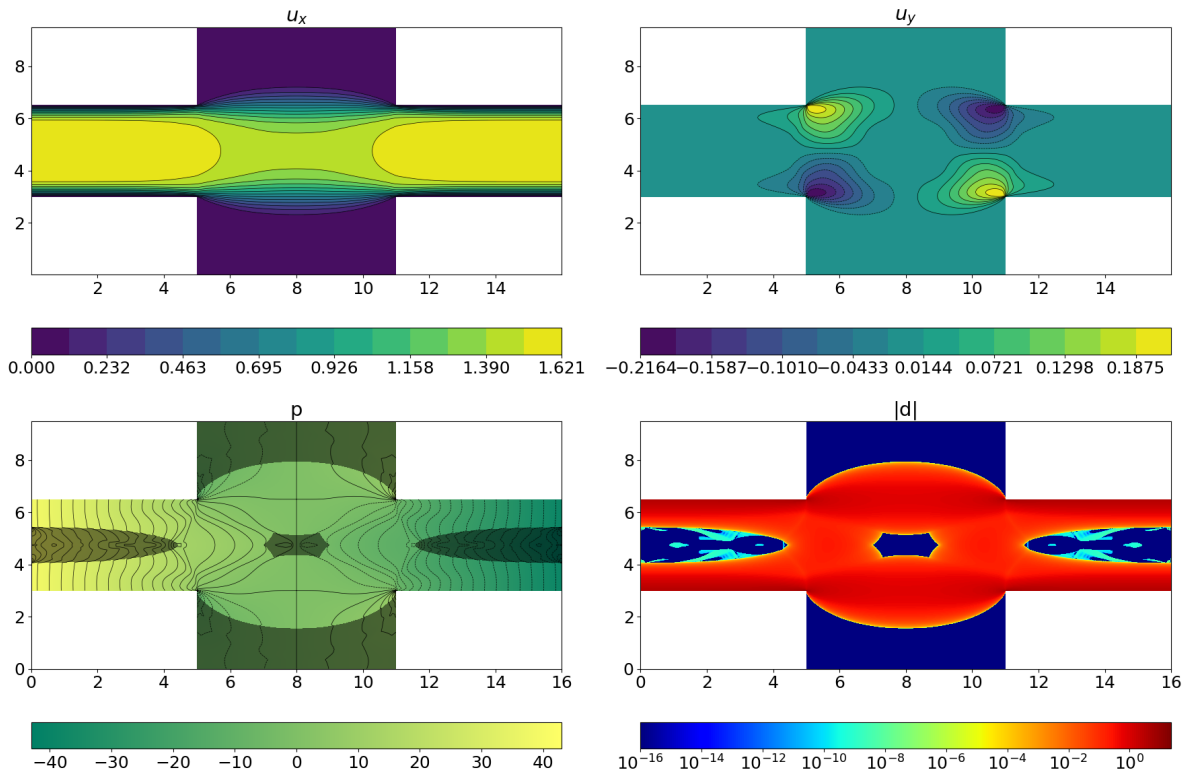
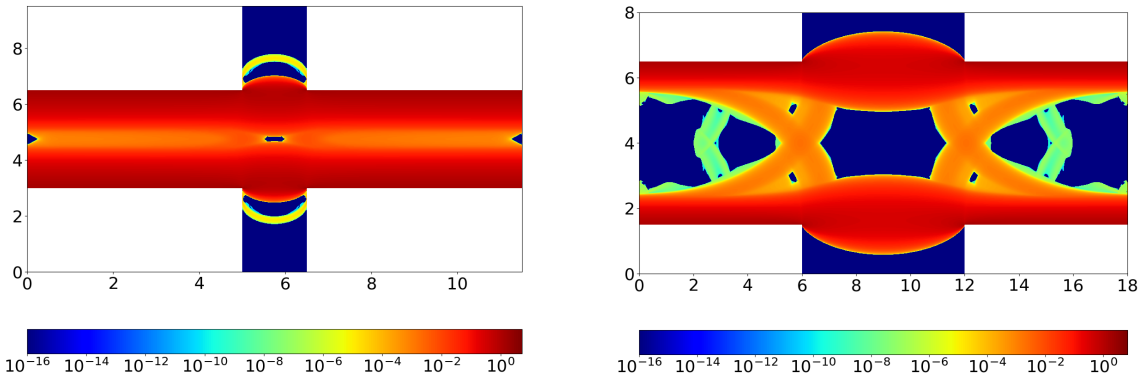


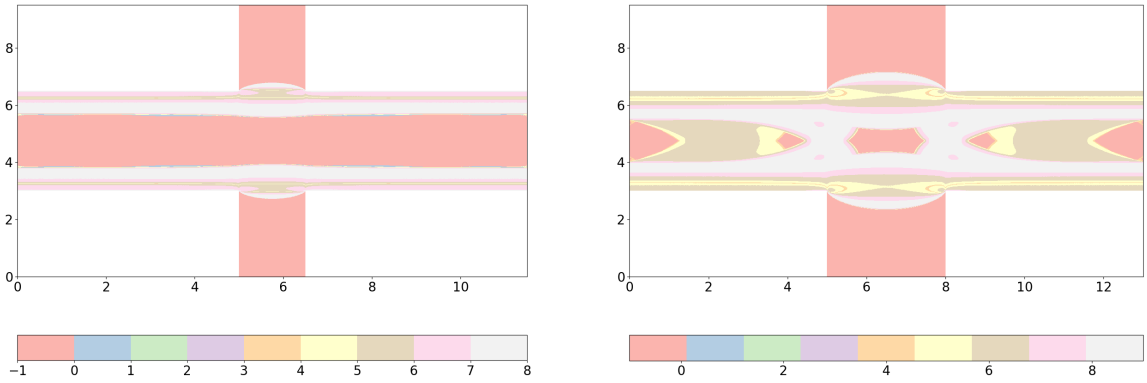
FIGURE 3.3.2 – Illustration de la vitesse, la pression et $|d|$ dans le cas $R16$, $L = 6$. La zone de plug est grisée dans l'illustration de la pression (on calcule les plugs en seuillant $|d|$ à 10^{-7}).



(a) R3, L=1.5 (b) R24, L=6

FIGURE 3.3.3 – Champ $|d|$ pour deux cas tests Herschel-Bulkley dont les plugs ont des difficultés à converger.

ici du nombre de Herschel-Bulkley, Hb_{cav}. Ce nombre est important car il rentre en jeu dans la théorie dite d'Oldroyd, que l'on peut tester au travers de graphes de δ_{BL} en fonction de Hb_{cav}, comme nous allons le faire ci-dessous.



(a) R22, L=1.5 (b) R15, L=3

FIGURE 3.3.4 – Nombre d'itérations de l'algorithme de Newton 20 pour deux cas tests Herschel-Bulkley. La valeur -1 est utilisée dans les zones où le développement limité (3.34) est utilisé.

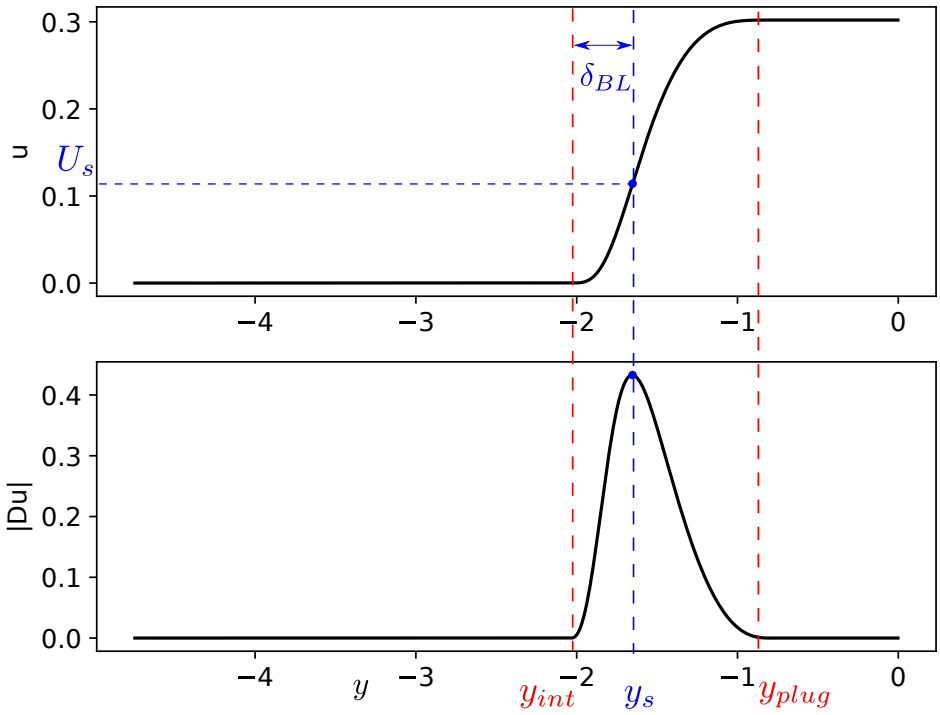


FIGURE 3.3.5 – Coupe transversale de u et $|Du|$ au centre de la cavité pour le cas R12, L=1.5. Seule la moitié inférieure est représentée.

$$Hb_{cav} = \frac{\tau_y}{2^{n-1}K}\left(\frac{L}{U_s}\right)^n \tag{3.66}$$

Pour la quarantaine de tests qui ont très bien convergé, nous avons post-traité les δ_{BL}. Sur la figure 3.3.6, on représente ces quantités à l'aide d'un changement d'échelle dérivé dans [243]. On constate un bon accord avec ce scaling d'Oldroyd étendu. Dans le détail, la pente de régression obtenue est de 0.48, ce qui semble quantitativement être légèrement plus proche du graphe

équivalent des expériences en laboratoires de [156] (où la pente obtenue est de 0.45), par rapport au cas Bingham. Comme nous l'avons dit précédemment, il s'agit ici d'une étude préliminaire et parcellaire qui demande à être poursuivie. Ces premiers résultats sont encourageants et suggèrent plusieurs pistes qui seront approfondies par le groupe du projet VPFlows.

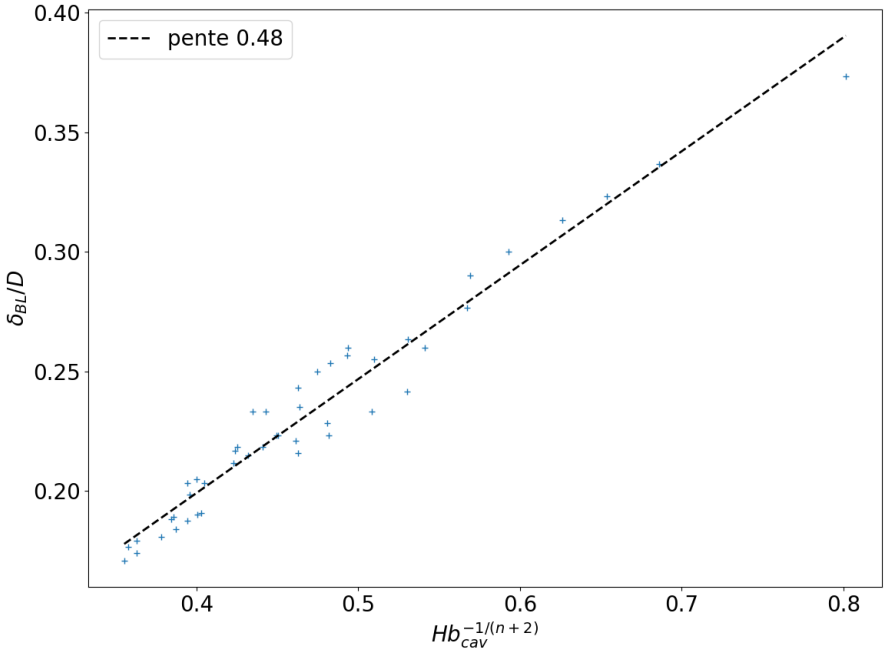


FIGURE 3.3.6 – Tracé de δ_{BL} en fonction du scaling d'Oldroyd. On obtient par régression linéaire une pente moyenne de 0.48.

3.4 Méthodologie SOCP

En 2015, Bleyer et al. [50] présentent une méthode originale basée sur la formulation (1.35). À partir de cette formulation, il est possible de reformuler le problème sous le formalisme de la programmation en cônes du second ordre (SOCP). Cependant, cette reformulation nécessite l'introduction de nombreuses variables supplémentaires, il en découle que l'algorithme obtenu demande de résoudre des systèmes linéaires de taille significativement plus grande que ceux des autres méthodes, aussi la méthode ne semble avoir été que peu réutilisée ensuite. En 2018 cependant, Bleyer [49] introduit une variation de l'algorithme qui utilise le formalisme et la méthodologie SOCP sans toutefois reformuler entièrement les équations comme un problème SOCP, de sorte à conserver des systèmes linéaires de taille raisonnable.

3.4.1 Introduction au formalisme SOCP

On commence par donner une brève présentation des concepts clés de la SOCP, pour plus de détails on renvoie à l'introduction de Alizadeh et Goldfarb [1] qui fait référence dans le domaine. On commence par introduire quelques notations.

Pour $x \in \mathbb{R}^d$, on note $x_0 \in \mathbb{R}$ sa première coordonnée et $\bar{x} \in \mathbb{R}^{d-1}$ sa *queue*, de sorte que :

$$x = \begin{pmatrix} x_0 \\ \bar{x} \end{pmatrix}. \tag{3.67}$$

La théorie SOCP étant basée sur cette distinction entre tête et queue, on écrit très souvent des éléments de $\mathbb{R}^d$ sous la forme ci-dessus, sans nécessairement préciser la taille des éléments. On introduit $\mathcal{Q}^d$ le cône de second ordre de $\mathbb{R}^d$ défini par :

$$\mathcal{Q}^d = \{x \in \mathbb{R}^d / x_0 \geq \|\bar{x}\|\}, \tag{3.68}$$

où $\|.\|$ désigne la norme euclidienne sur $\mathbb{R}^d$. On pose aussi l'intérieur de $\mathcal{Q}^d$:

$$\bar{\mathcal{Q}}^d = \{x \in \mathbb{R}^d / x_0 > \|\bar{x}\|\}. \tag{3.69}$$

Un problème SOCP est alors défini comme un problème de minimisation d'une fonction linéaire, sous contraintes d'égalités linéaires et contraintes d'appartenance à $\mathcal{Q}^d$. Les contraintes de cônes permettent de reformuler certaines contraintes quadratiques via des changements de variables, plaçant la théorie SOCP entre la programmation linéaire et la programmation quadratique.

La théorie SOCP est très bien connue et se base sur la construction d'une algèbre particulière, dont le produit $\circ$ se définit par :

$$x \circ s = \begin{pmatrix} x^T s \\ x_0 \bar{s} + s_0 \bar{x} \end{pmatrix}, \tag{3.70}$$

qui est bilinéaire symétrique et qui possède pour élément neutre :

$$e = \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \tag{3.71}$$

On trouvera dans [1] tous les éléments de construction d'une théorie de dualité spécifiquement adaptée. Le point qui nous intéresse ici particulièrement est celui des méthodes numériques utilisées pour résoudre ces problèmes. Il apparaît que les méthodes de point intérieur donnent en pratique d'excellents résultats dans ce contexte, notamment les méthodes de central path, qui relaxent les conditions de KKT pour aider l'algorithme de Newton sous-jacent à ne pas se bloquer sur la frontière du domaine défini par les contraintes (voir section 2.4). Pour fluidifier l'exposition, on présente directement sur notre problème les raffinements que l'on utilise au fur et à mesure de la présentation. On renvoie à Alizadeh et Goldfarb [1] et à Bleyer [49] pour plus de détails sur chacun de ces points.

Enfin, il est utile pour la suite de définir les notations suivantes (pour $x \in \mathcal{Q}^d$) :

$$\hat{x} = \begin{pmatrix} x_0 \\ -\bar{x} \end{pmatrix}, \tag{3.72}$$

la matrice symétrique :

$$Q = diag(1, -I_{d-1}) = \begin{pmatrix} 1 & 0 \\ 0 & -I_{d-1} \end{pmatrix}, \tag{3.73}$$

de sorte que $\hat{x} = Qx$ et enfin :

$$det(x) = x_0^2 - \|\bar{x}\|^2 > 0. \tag{3.74}$$

3.4.2 Reformulation du problème

On prend comme point de départ la formulation forte de (1.35) :

$$\begin{cases} -2^{n-1} K \mathrm{div}(\|Du\|^{n-1} Du) - \sqrt{2}\tau_y \mathrm{div}(\tau) + \nabla p = f \\ \|\tau\| \leq 1 \\ \tau : Du = \|Du\| \\ \mathrm{div}(u) = 0 \\ u = u_D \text{ sur } \partial\Omega. \end{cases} \tag{3.75}$$

Remarque 47 *On précise que le tenseur τ (noté m dans (1.35)) ne représente pas exactement la même quantité que le tenseur λ des algorithmes précédents.*

On commence par introduire une nouvelle fonction à valeurs scalaires t afin de reformuler les contraintes sur τ :

$$\begin{cases} -2^{n-1}K\mathrm{div}(\|Du\|^{n-1}Du) - \sqrt{2}\tau_y\mathrm{div}(\tau) + \nabla p = f \\ \|\tau\| \leq 1 \\ Du = t\tau \\ t \geq 0 \\ t(1 - \|\tau\|) = 0 \\ \mathrm{div}(u) = 0 \\ u = u_D \text{ sur } \partial\Omega, \end{cases} \tag{3.76}$$

La contrainte $t(1 - \|\tau\|) = 0$ permet de représenter une condition sur t : si $\|\tau\| \neq 1$, alors il faut $t = 0$. Par ailleurs, si on couple cela avec $Du = t\tau$, on trouve que dans le cas cisaillé (i.e. $Du \neq 0$), on a forcément $\|\tau\| = 1$ et $t = \|Du\|$. En revanche, on peut obtenir $\|\tau\| < 1$ dès lors que $t = 0$, ce qui implique que l'on se trouve dans une zone rigide. Cela traduit donc la contrainte $\tau : Du = \|Du\|$. On reformule enfin (3.76) en :

$$\begin{cases} -2^{n-1}K\mathrm{div}(\|Du\|^{n-1}Du) - \sqrt{2}\tau_y\mathrm{div}(\tau) + \nabla p = f \\ (t, Du) \in \mathcal{Q}^5 \\ (1, -\tau) \in \mathcal{Q}^5 \\ (t, Du) \circ (1, -\tau) = 0 \\ \mathrm{div}(u) = 0 \\ u = u_D \text{ sur } \partial\Omega. \end{cases} \tag{3.77}$$

Remarque 48 *On note abusivement $(t, Du) \in \mathcal{Q}^5$ pour dire que la fonction de l'espace (t, Du) est à valeurs dans $\mathcal{Q}^5$. En effet, étant donné que l'on utilise une discrétisation différences finies, c'est ce à quoi on s'intéresse en pratique dans la mise en place du code.*

Remarque 49 *On réécrit ici les équations avec des normes euclidiennes plutôt qu'avec la norme $|.|$ afin de faciliter les applications des formules liées à la théorie SOCP.*

La substitution $\|Du\|^{n-1}Du$ par $t^n\tau$ dans (3.77) donnant lieu a une formulation équivalente on remplace $\|Du\|^{n-1}Du$ par une fonction a priori générale $G(u, t, \tau)$, afin de garder la liberté du choix de l'expression de ce terme. Enfin, on introduit la variable annexe d telle que $d = Du$ comme pour les algorithmes précédents. On obtient donc la formulation finale du problème continu :

$$\begin{cases} -2^{n-1}K\mathrm{div}(G(u, t, \tau)) - \sqrt{2}\tau_y\mathrm{div}(\tau) + \nabla p = \nu \\ (t, d) \in \mathcal{Q}^5 \\ (1, -\tau) \in \mathcal{Q}^5 \\ (t, d) \circ (1, -\tau) = 0 \\ d = Du \\ \mathrm{div}(u) = 0 \\ u = u_D \text{ sur } \partial\Omega. \end{cases} \tag{3.78}$$

Plutôt que de reformuler entièrement le système comme un problème SOCP, Bleyer [49] propose de conserver cette formulation pour appliquer une méthode numérique.

Remarque 50 *Dans son papier [49], Bleyer ne présente explicitement que le cas Bingham, aussi les calculs suivants, qui sont faits pour le cas Herschel-Bulkley, semblent nouveaux dans la littérature.*

3.4.3 Central path

Comme annoncé précédemment, on applique une méthode de point intérieur s'appuyant sur une procédure de central path (section 2.4), on effectue donc une méthode de Newton sur le problème modifié :

$$\begin{cases} -2^{n-1}K\mathrm{div}(G(u,t,\tau)) - \sqrt{2}\tau_y \mathrm{div}(\tau) + \nabla p = \nu \\ (t,d) \in \mathcal{Q}^5 \\ (1,-\tau) \in \mathcal{Q}^5 \\ (t,d) \circ (1,-\tau) = (\mu,0)^T \\ d = Du \\ \mathrm{div}(u) = 0 \\ u = u_D \text{ sur } \partial\Omega, \end{cases} \tag{3.79}$$

où μ est le paramètre de barrière qui est déterminé à chaque itération. Ici l'ajout de μ peut se voir comme une relaxation de $t(1 - \|\tau\|) = 0$ en $t(1 - \|\tau\|) = \mu/(1 + \|\tau\|)$.

Remarque 51 *C'est précisément pour cette raison que l'algorithme peut être interprété comme une méthode de régularisation.*

Idéalement, le paramètre μ doit à terme décroître vers 0 de sorte à ce que le système perturbé (3.79) tende vers le système (3.78). On utilise alors l'*écart de complémentarité* de l'itération k pour calculer μ. On définit l'écart de complémentarité au point z par :

$$g^{(k)}(z) = t^{(k)}(z) - \langle \tau^{(k)}(z), d^{(k)}(z) \rangle, \tag{3.80}$$

et sa moyenne sur le domaine (que l'on écrit ici en version discrète en supposant une grille uniforme) :

$$\bar{g}^{(k)} = \frac{1}{N} \sum_{i,j} g_{i,j}^{(k)}, \tag{3.81}$$

où N est le nombre total de points dans la discrétisation (que l'on détaille en section 3.4.4). On pose alors :

$$\mu^{(k+1)} = \gamma \bar{g}^{(k)}, \tag{3.82}$$

où $\gamma \in]0,1]$ est le *paramètre de centrage*. Ce paramètre permet de régler le compromis entre la réduction de l'écart de complémentarité à la prochaine itération et le maintien du centrage des itérées pour permettre des pas de Newton plus importants (on revient sur le calcul du pas en section 3.4.8). On présente en section 3.4.6 une méthode permettant de choisir γ de manière adaptative.

Remarque 52 *Les calculs étant volumineux et chargés en notations, on essaie autant que possible d'éviter d'avoir à préciser les exposants k liés au numéro de l'itération. On ne les met que ponctuellement lorsque des éléments appartenant à plusieurs itérations distinctes se retrouvent dans une même équation, comme par exemple dans la formule (3.82) définissant $\mu^{(k+1)}$.*

3.4.4 Discrétisation et écriture de la méthode de Newton

On discrétise maintenant les équations (3.79) avant d'écrire la formulation de l'algorithme de Newton. On utilise comme précédemment le maillage MAC défini en section 3.1.4. Cela nous permet de discrétiser la plupart des opérations du système, mais il manque encore la discrétisation de t, ainsi que l'écriture des contraintes de cône et le calcul du produit circulaire.

Les équations faisant intervenir t (à part l'équation principale, qui ne compte qu'à travers l'opérateur G) sont liées aux composantes des tenseurs, il semble donc naturel de localiser t à un emplacement associé à une de ces composantes. Comme en plus, il partage la même nature que la pression (les deux fonctions sont à valeurs scalaires), on décide de le placer au même emplacement (ce qui, on le rappelle, correspond aussi à l'emplacement des composantes diagonales des tenseurs). Par voie de conséquence, on écrit les contraintes de cônes ainsi que le résultat du produit circulaire au même emplacement, en utilisant des moyennes à quatre points (ou deux voire un point sur les bords) pour les quantités manquantes.

Remarque 53 *Placer t à la fois aux emplacements de pression/composante diagonale de tenseur ainsi qu'aux emplacements de composantes extra-diagonales (démultipliant donc le nombre de points de discrétisation de t) a aussi été testé. Il n'y a pas eu de conséquence notable dans les tests numériques effectués, à part un coût en mémoire plus important.*

La suite de l'algorithme nécessite de procéder à différentes reformulations pour réduire la taille des systèmes linéaires, aussi il devient nécessaire d'introduire les versions discrètes des opérateurs considérés. On utilise les notations matricielles suivantes pour les opérateurs différentiels :

$$\operatorname{div}(\tau) \longrightarrow M_{dT}\tau, \tag{3.83a}$$
$$\operatorname{div}(u) \longrightarrow M_{dV}u, \tag{3.83b}$$
$$Du \longrightarrow M_D u, \tag{3.83c}$$
$$\nabla p \longrightarrow M_\nabla p. \tag{3.83d}$$

On remarque que pour les inconnues, on conserve les mêmes notations dans le cas discret que dans le cas continu. En toute rigueur, on devrait remplacer par exemple u^x par un vecteur contenant tous les u^x_{ij}. On omet cette distinction la plupart du temps.

On introduit ensuite la matrice diagonale M_{BC} agissant sur la vitesse u et permettant d'encoder les conditions de bord. Plus précisément, un terme diagonal de M_{BC} vaut 1 si et seulement si il correspond à un élément de vitesse se trouvant au bord du domaine, de telle sorte à ce que si on note u_{BC} la discrétisation des conditions de bord u_D (où on remplit u_{BC} avec des 0 à l'intérieur du domaine), on a :

$$u = u_D \text{ sur } \partial\Omega \underset{\text{discret}}{\Leftrightarrow} M_{BC}u = u_{BC}, \tag{3.84}$$

ce qui nous permet d'introduire une notation matricielle pour les conditions de bord.

Il reste enfin l'opérateur G. Celui-ci n'étant pas a priori linéaire, l'opérateur et sa différentielle demandent à être discrétisés pour l'étape de Newton. On note donc :

$$G(u,t,\tau) \longrightarrow G_{disc}(u,t,\tau) \tag{3.85a}$$
$$\nabla G(u,t,\tau) \longrightarrow M_{Gu}u + M_{Gt}t + M_{G\tau}\tau. \tag{3.85b}$$

On peut maintenant écrire le système discrétisé :

$$\begin{cases} -2^{n-1}K\mathrm{div}\left(G_{disc}(u,t,\tau)\right) - \sqrt{2}\tau_y M_{dT}\tau + M_\nabla p = f \\ (t,\tilde{d}) \in \mathcal{Q}^5 \\ (1,-\tilde{\tau}) \in \mathcal{Q}^5 \\ (t,d)\circ(1,-\tau) = (\mu,0)^T \\ d = M_D u \\ M_{dV}u = 0 \\ M_{BC}u = u_{BC}. \end{cases} \tag{3.86}$$

Enfin, on remarque que les points en lesquels l'équation principale est écrite et ceux en lesquels les conditions de bord sont écrites sont complémentaires, on peut donc ajouter l'équation sur les conditions de bord à l'équation principale de sorte à obtenir :

$$\begin{cases} -2^{n-1}KM_{dT}G_{disc}(u,t,\tau) - \sqrt{2}\tau_y M_{dT}\tau + M_\nabla p + M_{BC}u = f + u_{BC} \\ (t,\tilde{d}) \in \mathcal{Q}^5 \\ (1,-\tilde{\tau}) \in \mathcal{Q}^5 \\ (t,d)\circ(1,-\tau) = (\mu,0)^T \\ d = M_D u \\ M_{dV}u = 0. \end{cases} \tag{3.87}$$

Remarque 54 *En pratique, c'est ce qui permet de garantir que la matrice du problème n'est pas singulière.*

On peut désormais écrire le système à résoudre lors d'une étape de Newton :

$$\begin{cases} -2^{n-1}KM_{dT}\left(M_{Gu}\Delta u + M_{Gt}\Delta t + M_{G\tau}\Delta\tau\right) - \sqrt{2}\tau_y M_{dT}\Delta\tau + M_\nabla\Delta p + M_{BC}\Delta u = \\ \qquad f + u_{BC} + 2^{n-1}K\mathrm{div}\left(G_{disc}(u,t,\tau)\right) + \sqrt{2}\tau_y M_{dT}\tau - M_\nabla p - M_{BC}u \\ (\Delta t,\Delta d)\circ(0,-\Delta\tau) = (\mu,0)^T - (t,d)\circ(1,-\tau) \\ M_D\Delta u - \Delta d = d - M_D u \\ M_{dV}\Delta u = -M_{dV}u. \end{cases} \tag{3.88}$$

On calcule alors le pas de la méthode de Newton de sorte à satisfaire les contraintes de cône (voir section 3.4.8), qui sont retirées du système. On attire l'attention sur le fait que si la ligne sur le produit circulaire n'a pas été linéarisée, c'est parce que le système que l'on utilise n'est pas formulé ainsi. En effet, en l'état actuel, on a un système de taille $10N$ (si on utilise la symétrie des tenseurs) où N est le nombre de points sur le maillage, là où le sous-problème de Stokes généralisé donné par les algorithmes précédents est de taille $3N$. On commence donc par effectuer un changement de variable qui nous permet de réduire la taille du système.

3.4.5 Scaling de Todd-Nesterov et formulation matricielle

On applique le *scaling de Todd-Nesterov* [1, 6, 180, 255] qui permet de symétriser les contraintes en τ, d et t. On pose :

$$x = \begin{pmatrix} t \\ d \end{pmatrix}, s = \begin{pmatrix} 1 \\ -\tau \end{pmatrix}, \tag{3.89}$$

où d et τ sont ici aplatis. L'ordre n'a pas d'importance en pratique, disons par exemple :

$$x = \begin{pmatrix} t \\ d^{xx} \\ d^{yy} \\ d^{xy} \\ d^{yx} \end{pmatrix}, \tag{3.90}$$

(dans le code, on utilise les symétries pour supprimer d^{yx}). Avec ces nouvelles variables, on peut alors remplacer la contrainte :

$$(\Delta t, \Delta d) \circ (0, -\Delta\tau) = (\mu, 0)^T - (t, d) \circ (1, -\tau) \tag{3.91}$$

par la contrainte suivante (voir Bleyer [49] section 4.2 et annexe C) :

$$VF\Delta x + VF^{-1}\Delta s = \mu e - V^2 e, \tag{3.92}$$

où :

$$\theta = \left(\frac{det(s)}{det(x)}\right)^{1/4}, \tag{3.93}$$

$$w = \frac{\theta^{-1}s + \theta\hat{x}}{\sqrt{2}\sqrt{\left(x^T s + \sqrt{det(x)det(s)}\right)}}, \tag{3.94}$$

$$F = \theta \begin{pmatrix} w_0 & \bar{w}^T \\ \bar{w} & I_4 + \frac{\bar{w}\bar{w}^T}{1+w_0} \end{pmatrix}, \tag{3.95}$$

$$F^{-1} = \theta^{-2} QFQ, \tag{3.96}$$

$$F^{-2} = \theta^{-2}\left(-Q + 2\hat{w}\hat{w}^T\right), \tag{3.97}$$

$$v = Fx, \tag{3.98}$$

$$V = \begin{pmatrix} v_0 & \bar{w}^T \\ \bar{w} & v_0 I_4 \end{pmatrix}. \tag{3.99}$$

Les contraintes de cônes deviennent alors :

$$x \in \mathcal{Q}^5, \qquad s \in \mathcal{Q}^5. \tag{3.100}$$

Remarque 55 *Comme évoqué précédemment, les exposants k relatifs au numéro de l'itération sont omis. Il convient donc d'attirer l'attention sur le fait que les matrices F et V dépendent de l'itération courante.*

Il nous faut alors décider d'une discrétisation pour x et s. On conserve naturellement pour chaque composante sa discrétisation d'origine. Cela signifie par exemple que $x_0 = t$, $x_1 = d^{xx}$ et $x_2 = d^{yy}$ sont placés sur les $z_{i+1/2,j+1/2}$, tandis que $x_3 = d^{xy}$ et $x_4 = d^{yx}$ sont placés sur les $z_{i,j}$.

Comme évoqué précédemment, on utilise des moyennes à quatre (ou deux ou un) points lorsque les variables requises pour un calcul ne sont pas disponibles. À ce stade, il convient de donner une notation matricielle pour cette méthode. En effet, la correspondance entre le vecteur des $\tau^{xy}_{i,j}$ et celui des $\tilde{\tau}^{xy}_{i+1/2,j+1/2}$ obtenus par calculs de moyennes est linéaire. Ainsi, si on note $\tilde{\tau}$ le tenseur dont toutes les composantes sont exprimées en les $z_{i+1/2,j+1/2}$ (les $\tilde{\tau}^{xy}$ sont donc obtenus

via des moyennes et les composantes diagonales sont inchangées), on peut écrire ce processus d'interpolation sous la forme :

$$\tilde{\tau} = M_{int}\tau. \tag{3.101}$$

Pour le processus d'extrapolation des τ^{xy} à partir des $\tilde{\tau}^{xy}$, on note :

$$\tau = M_{ext}\tilde{\tau}. \tag{3.102}$$

Remarque 56 *Il est à noter que l'interpolation ne nécessite que des moyennes à quatre points, puisque les $z_{i+1/2,j+1/2}$ sont toujours entourés de quatre voisins. À l'inverse, le processus d'extrapolation pose les problèmes évoqués en section 3.1.4 sur les bords. Bien que la procédure soit toujours linéaire, M_{int} et M_{ext} ne sont pas inverses l'une de l'autre, elles ne sont en fait même pas carrées puisque le placement des mailles par rapport au bord fait qu'il y a plus de points de discrétisation pour les composantes extra-diagonales que pour les composantes diagonales.*

En conséquence, la contrainte (3.92) devient, une fois discrétisée :

$$VF\begin{pmatrix}1 & 0\\ 0 & M_{int}\end{pmatrix}\Delta x + VF^{-1}\begin{pmatrix}1 & 0\\ 0 & M_{int}\end{pmatrix}\Delta s = \mu e - V^2 e. \tag{3.103}$$

Remarque 57 *Les matrices V, F et F^{-1} sont en fait construites en chaque point du maillage, puis assemblées dans des matrices diagonales par blocs (de tailles 4×4 si l'on tient compte de la symétrie des tenseurs). Ce système est donc intrinsèquement creux, même après interpolation.*

L'étape de Newton s'exprime donc sous la forme :

$$\begin{cases} -2^{n-1}KM_{dT}\left(M_{Gu}\Delta u + (M_{Gt},0)\Delta x - (0,M_{G\tau})\Delta s\right) - \sqrt{2}\tau_y M_{dT}\Delta\tau + M_\nabla \Delta p + M_{BC}\Delta u = \\ \qquad f + u_{BC} + 2^{n-1}KM_{dT}G_{disc}(u,t,\tau) + \sqrt{2}\tau_y M_{dT}\tau - M_\nabla p - M_{BC}u \\ VF\begin{pmatrix}1 & 0\\ 0 & M_{int}\end{pmatrix}\Delta x + VF^{-1}\begin{pmatrix}1 & 0\\ 0 & M_{int}\end{pmatrix}\Delta s = \mu e - V^2 e \\ M_D\Delta u - \Delta d = d - M_D u \\ M_{dV}\Delta u = -M_{dV}u. \end{cases} \tag{3.104}$$

Enfin, concernant les contraintes de cône, on demande qu'elles soient vérifiées de manière stricte à chaque étape (il faut notamment que $x^{(0)}$ et $s^{(0)}$ soient initialisés de sorte à vérifier cette propriété). Elles deviennent donc :

$$\tilde{x} + \alpha\Delta\tilde{x} \in \bar{\mathcal{Q}}^5 \tag{3.105a}$$

$$\tilde{s} + \alpha\Delta\tilde{s} \in \bar{\mathcal{Q}}^5, \tag{3.105b}$$

où α désigne le pas du Newton.

Remarque 58 *On rappelle que dans le membre de droite de (3.104), les t, d et τ qui apparaissent sont ceux de l'itération précédente. Il sont donc connus et il n'est pas utile de les reformuler via x et s. On les laisse donc en l'état.*

On écrit maintenant le système de Newton (3.104) sous forme matricielle. On doit présenter de nombreuses formules, dont des étapes intermédiaires. Pour distinguer les équations intermédiaires des formules effectivement utilisées par l'algorithme qui est implémenté, on encadre

ces dernières. Pour implémenter la méthode, il suffit donc de suivre les formules encadrées. On commence par poser :

$$\tilde{\eta} = 2^{n-1}K, \tag{3.106}$$

$$r_f = f + u_{BC} + 2^{n-1}KM_{dT}G_{disc}(u,t,\tau) + \sqrt{2}\tau_y M_{dT}\tau - M_\nabla p - M_{BC}u, \tag{3.107}$$

$$r_p = -M_{dV}u, \tag{3.108}$$

$$r_d = d - M_D u, \tag{3.109}$$

$$r_x^\mu = \mu e - V^2 e. \tag{3.110}$$

La formulation matricielle du système de Newton (3.104) est alors :

$$\begin{pmatrix} -\tilde{\eta}M_{dT}M_{Gu} + M_{BC} & M_\nabla & \left(0, \tilde{\eta}M_{dT}M_{G\tau} + \sqrt{2}\tau_y M_{dT}\right) & \left(-\tilde{\eta}M_{dT}M_{Gt}, 0\right) \\ M_{dV} & 0 & 0 & 0 \\ M_D & 0 & 0 & \left(0, -I\right) \\ 0 & 0 & VF^{-1}\begin{pmatrix}1 & 0\\ 0 & M_{int}\end{pmatrix} & VF\begin{pmatrix}1 & 0\\ 0 & M_{int}\end{pmatrix} \end{pmatrix} \begin{pmatrix}\Delta u\\ \Delta p\\ \Delta s\\ \Delta x\end{pmatrix} = \begin{pmatrix} r_f \\ r_p \\ r_d \\ r_x^\mu \end{pmatrix}. \tag{3.111}$$

En l'état actuel, la taille du problème linéaire n'a pas été réduite, il y a cependant plusieurs blocs symétriques que l'on peut utiliser à notre avantage. Avant de présenter la façon dont on réduit le système, on présente un dernier raffinement.

3.4.6 Prédiction-correction de Mehrotra

Cette section présente une méthode permettant notamment de choisir le paramètre de centrage γ (voir section 3.4.3). En 1992, Mehrotra [163] propose une méthode permettant de choisir ce paramètre de manière adaptative, à l'aide d'un schéma de *prédiction-correction*.

On commence par résoudre une étape de Newton en utilisant $\mu = 0$ (le paramètre γ est donc supprimé), on appelle cette étape la prédiction ou l'étape *affine*. Si on note $t^+ = t + \Delta t^+$ le résultat (on note de la même façon les autres variables), alors on a :

$$\begin{aligned}(t^+, d^+) \circ (1, -\tau^+) =& (t,d)\circ(1,-\tau) \\ &+ (\Delta t^+ - \tau^T\Delta d^+ - d^T\Delta\tau^+, \Delta d^+ - t\Delta\tau^+ - \Delta t\tau^+) \\ &- ((\Delta d^+)^T\Delta\tau^+, \Delta t^+\Delta\tau^+).\end{aligned} \tag{3.112}$$

Une étape de Newton classique va normalement linéariser l'expression et donc négliger le terme quadratique $((\Delta d^+)^T\Delta\tau^+, \Delta t^+\Delta\tau^+)$. Mehrotra propose plutôt d'utiliser l'estimation de ce terme quadratique fourni par l'étape affine pour améliorer la direction de recherche du système de Newton avec centrage. Notons donc α^+_{max} le pas maximal associé à l'étape affine. On prend :

$$\boxed{\gamma = (1 - \alpha^+_{max})\min\{0.5, (1-\alpha^+_{max})^2\}}. \tag{3.113}$$

Heuristiquement, si l'étape affine permet déjà un grand pas, on choisit γ petit car alors le processus de centrage n'a pas besoin d'être important. Si au contraire l'étape affine bloque, alors on augmente γ pour recentrer la direction de recherche et agrandir le pas.

On peut ensuite écrire l'étape de Newton avec centrage en approchant le terme quadratique par celui estimé à l'aide de l'étape affine. En variables (x,s), il s'écrit sous la forme $F\begin{pmatrix}1 & 0\\ 0 & M_{int}\end{pmatrix}\Delta x^+ \circ F^{-1}\begin{pmatrix}1 & 0\\ 0 & M_{int}\end{pmatrix}\Delta s^+$. Enfin, on multiplie les résidus par $1-\gamma$ comme conseillé

par Bleyer [49]. L'étape de correction correspond donc au système (3.111) où les résidus ont été remplacés par les suivants :

$$\boxed{r_f^\gamma = (1-\gamma)\left(f + u_{BC} + 2^{n-1} K M_{dT} G_{disc}(u,t,\tau) + \sqrt{2}\tau_y M_{dT}\tau - M_\nabla p - M_{BC} u\right)}, \tag{3.114}$$

$$\boxed{r_p^\gamma = -(1-\gamma) M_{dV} u}, \tag{3.115}$$

$$\boxed{r_d^\gamma = (1-\gamma)\left(d - M_D u\right)}, \tag{3.116}$$

$$\boxed{r_x^{\mu,+} = \mu e - V^2 e - F \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \Delta x^+ \circ F^{-1} \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \Delta s^+}. \tag{3.117}$$

L'étape affine est alors vue comme un cas particulier de ce système, où l'on prend $\gamma = \mu = 0$ et $\Delta x^+ = \Delta s^+ = 0$, ce qui nous donne :

$$\begin{pmatrix} -\tilde{\eta} M_{dT} M_{Gu} + M_{BC} & M_\nabla & \left(0, \tilde{\eta} M_{dT} M_{G\tau} + \sqrt{2}\tau_y M_{dT}\right) & \left(-\tilde{\eta} M_{dT} M_{Gt}, 0\right) \\ M_{dV} & 0 & 0 & 0 \\ M_D & 0 & 0 & \left(0, -I\right) \\ 0 & 0 & V F^{-1} \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} & V F \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \end{pmatrix} \begin{pmatrix} \Delta u \\ \Delta p \\ \Delta s \\ \Delta x \end{pmatrix} = \begin{pmatrix} r_f^\gamma \\ r_p^\gamma \\ r_d^\gamma \\ r_x^{\mu,+} \end{pmatrix}. \tag{3.118}$$

Remarque 59 *La matrice du système est inchangée entre l'étape affine et l'étape de correction. Si on utilise une méthode de résolution directe de type LU, on peut alors réutiliser la factorisation pour l'étape de correction pour réduire le temps de calcul.*

3.4.7 Réduction du système

On montre maintenant comment éliminer les variables Δx et Δs du système (3.118) afin de retrouver un problème en Δu, Δp. On commence par éliminer la dernière ligne :

$$\begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \Delta x = -F^{-2} \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \Delta s + F^{-1} V^{-1} r_x^{\mu,+}. \tag{3.119}$$

Le processus d'interpolation fait que l'on ne peut pas donner de formule exacte pour Δx. On utilise le processus d'extrapolation :

$$\boxed{\Delta x = -\begin{pmatrix} 1 & 0 \\ 0 & M_{ext} \end{pmatrix} F^{-2} \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \Delta s + \begin{pmatrix} 1 & 0 \\ 0 & M_{ext} \end{pmatrix} F^{-1} V^{-1} r_x^{\mu,+}}. \tag{3.120}$$

Remarque 60 *Cette difficulté est propre aux différences finies. Une formulation éléments finis étant basée sur des intégrales, ce problème s'exprime différemment. Cela peut en partie expliquer les instabilités que l'on détaille en section 3.4.9.*

On utilise (3.120) pour reformuler la première ligne sans la variable Δx :

$$\begin{aligned} \left(-\tilde{\eta} M_{dT} M_{Gu} + M_{BC}\right) \Delta u + M_\nabla \Delta p + \left[\left(0, \tilde{\eta} M_{dT} M_{G\tau} + \sqrt{2}\tau_y M_{dT}\right) + \left(\tilde{\eta} M_{dT} M_{Gt}, 0\right) F^{-2}\right] \Delta s \\ = r_f^\gamma + \left(\tilde{\eta} M_{dT} M_{Gt}, 0\right) F^{-1} V^{-1} r_x^{\mu,+}. \end{aligned} \tag{3.121}$$

On rappelle que l'on a une formule explicite pour F^{-2}, voir section 3.4.5. On pose alors :

$$\boxed{r_f^{\gamma,\mu,+} = r_f^{\gamma} + \Big(\tilde{\eta} M_{dT} M_{Gt}, 0\Big) F^{-1} V^{-1} r_x^{\mu,+}}, \tag{3.122}$$

et on reformule aussi la troisième ligne du système :

$$M_D \Delta u + \Big(0, M_{ext}\Big) F^{-2} \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \Delta s = r_d^{\gamma} + \Big(0, M_{ext}\Big) F^{-1} V^{-1} r_x^{\mu,+}. \tag{3.123}$$

On termine en posant :

$$\boxed{r_d^{\gamma,\mu,+} = r_d^{\gamma} + \Big(0, M_{ext}\Big) F^{-1} V^{-1} \left(\mu e - V^2 e - F \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \Delta x^+ \circ F^{-1} \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \Delta s^+ \right)}, \tag{3.124}$$

pour aboutir au système :

$$\begin{pmatrix} -\tilde{\eta} M_{dT} M_{Gu} + M_{BC} & M_{\nabla} & \Big(0, \tilde{\eta} M_{dT} M_{G\tau} + \sqrt{2}\tau_y M_{dT}\Big) + \Big(\tilde{\eta} M_{dT} M_{Gt}, 0\Big) F^{-2} \\ M_{dV} & 0 & 0 \\ M_D & 0 & \Big(0, M_{ext}\Big) F^{-2} \begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix} \end{pmatrix} \begin{pmatrix} \Delta u \\ \Delta p \\ \Delta s \end{pmatrix} = \begin{pmatrix} r_f^{\gamma,\mu,+} \\ r_p^{\gamma} \\ r_d^{\gamma,\mu,+} \end{pmatrix}. \tag{3.125}$$

Réutilisons la variable τ afin de simplifier l'écriture ci-dessus. En effet, $s = (1 - \tau)$, donc on obtient que $\Delta s = (0, -\Delta\tau)$. Notons alors $\bar{F}^{-2}$ le bloc 4×4 de queue de F^{-2} (i.e. on enlève la première ligne et la première colonne de F^{-2}). La troisième ligne nous donne alors :

$$M_{ext} \bar{F}^{-2} M_{int} \Delta\tau = M_D \Delta u - r_d^{\gamma,\mu,+}, \tag{3.126}$$

et donc :

$$\boxed{\Delta\tau = M_{ext} \left(\bar{F}^{-2}\right)^{-1} M_{int} M_D \Delta u - M_{ext} \left(\bar{F}^{-2}\right)^{-1} M_{int} r_d^{\gamma,\mu,+}}, \tag{3.127}$$

où, comme pour Δx, on a rajouté une étape d'extrapolation. On note qu'on a la formule :

$$\boxed{\left(\bar{F}^{-2}\right)^{-1} = \theta^2 \left(I - 2 \frac{\bar{w}\bar{w}^T}{1 + 2\|\bar{w}\|^2} \right)}. \tag{3.128}$$

Il reste à reformuler la première ligne. Notons :

$$\boxed{L_{Gt,F} = \Big(M_{dT} M_{Gt}, 0\Big) F^{-2}}, \tag{3.129}$$

une matrice rectangulaire. On note $\overline{L_{Gt,F}}$ la même matrice dont on retire la première colonne.

Remarque 61 *On utilise donc la notation barre pour plusieurs objets différents. Pour un vecteur, cela signifie qu'on en prend la queue. Pour une matrice carrée, cela signifie qu'on lui enlève la première ligne et la première colonne. Pour une matrice rectangulaire $(d+1) \times d$, cela signifie qu'on lui enlève la première colonne. Dans tous les cas, $\bar{A}\bar{v}$ désigne une matrice carrée agissant sur la queue du vecteur v.*

Avec ces notations, on a :

$$\begin{aligned}
&\left[\left(0, \tilde{\eta} M_{dT} M_{G\tau} + \sqrt{2}\tau_y M_{dT}\right) + \left(\tilde{\eta} M_{dT} M_{Gt}, 0\right) F^{-2}\right] \Delta s \\
&= \left[-\tilde{\eta} M_{dT} M_{G\tau} - \sqrt{2}\tau_y M_{dT} - \tilde{\eta}\overline{L_{Gt,F}}\right] \Delta\tau \\
&= \left[-\tilde{\eta} M_{dT} M_{G\tau} - \sqrt{2}\tau_y M_{dT} - \tilde{\eta}\overline{L_{Gt,F}}\right] \left(M_{ext}\left(\bar{F}^{-2}\right)^{-1} M_{int} M_D \Delta u - M_{ext}\left(\bar{F}^{-2}\right)^{-1} M_{int} r_d^{\gamma,\mu,+}\right),
\end{aligned} \tag{3.130}$$

et on pose :

$$\boxed{r_u^{\gamma,\mu,+} = r_f^{\gamma,\mu,+} - \left[\tilde{\eta} M_{dT} M_{G\tau} + \sqrt{2}\tau_y M_{dT} + \tilde{\eta}\overline{L_{Gt,F}}\right] M_{ext}\left(\bar{F}^{-2}\right)^{-1} M_{int} r_d^{\gamma,\mu,+}}, \tag{3.131}$$

pour écrire le système final :

$$\boxed{\begin{pmatrix} -\tilde{\eta} M_{dT} M_{Gu} + M_{BC} - \left[\tilde{\eta} M_{dT} M_{G\tau} + \sqrt{2}\tau_y M_{dT} + \tilde{\eta}\overline{L_{Gt,F}}\right] M_{ext}\left(\bar{F}^{-2}\right)^{-1} M_{int} M_D & M_\nabla \\ M_{dV} & 0 \end{pmatrix} \begin{pmatrix} \Delta u \\ \Delta p \end{pmatrix} = \begin{pmatrix} r_u^{\gamma,\mu,+} \\ r_p^{\gamma} \end{pmatrix}.} \tag{3.132}$$

Remarque 62 *Le cas Bingham simplifie drastiquement les calculs, puisqu'on prend alors naturellement $G(u,t,\tau) = D(u)$, en particulier M_{Gt} et $M_{G\tau}$ sont nulles. On donne le système correspondant en annexe 3.B.*

3.4.8 Calcul du pas

On s'intéresse maintenant au calcul du pas α. On cherche pour cela le plus grand $\alpha \leq 1$ permettant à l'itérée suivante de vérifier les contraintes de cône (3.105). En écrivant explicitement ces contraintes, on peut donner une formule explicite pour α. On commence par fixer un point $z_{i+1/2,j+1/2}$ pour calculer le pas maximal utilisable en ce point. Le pas global admissible α_{max} est récupéré en minimisant sur tous les points du maillage. On prend enfin $\alpha = 0.99\alpha_{max}$ par sécurité.

Réécrivons (3.105) en terme des variables t, d et τ :

$$\begin{cases} 1 \geq \|\tilde{\tau} + \alpha\Delta\tilde{\tau}\| \\ t \geq \|\tilde{d} + \alpha\Delta\tilde{d}\|, \end{cases} \tag{3.133}$$

que l'on explicite en :

$$1 \geq \|\tilde{\tau} + \alpha\Delta\tilde{\tau}\|^2 \tag{3.134a}$$

$$(t + \alpha\Delta t)^2 \geq \|\tilde{d} + \alpha\Delta\tilde{d}\|^2 \tag{3.134b}$$

$$t + \alpha\Delta t \geq 0. \tag{3.134c}$$

Chacune de ces équations donne un pas maximal, respectivement α_τ, $\alpha_{t,d}$ et α_t. Le pas final s'écrit donc :

$$\boxed{\alpha = 0.99\alpha_{max} = 0.99\min\{\alpha_\tau, \alpha_{t,d}, \alpha_t\}}. \tag{3.135}$$

On rappelle que l'on suppose que les contraintes sont bien vérifiées à l'itération précédente, autrement dit :

$$1 \geq \|\tilde{\tau}\|^2 \tag{3.136a}$$
$$t^2 \geq \|\tilde{d}\|^2 \tag{3.136b}$$
$$t \geq 0. \tag{3.136c}$$

3.4.8.1 Première contrainte (3.134a)

Les traitements des contraintes (3.134a) et (3.134b) sont identiques, on trouve la première comme cas particulier de la seconde, aussi on donne ici seulement le résultat :

$$\boxed{\alpha_\tau = \begin{cases} \dfrac{-\tilde{\tau}^T\Delta\tilde{\tau} + \sqrt{(\tilde{\tau}^T\Delta\tilde{\tau})^2 - \|\Delta\tilde{\tau}\|^2\|\tilde{\tau}\|^2}}{\|\Delta\tilde{\tau}\|^2} & \text{si } \|\Delta\tilde{\tau}\| \neq 0, \\ 1 & \text{sinon.} \end{cases}} \tag{3.137}$$

3.4.8.2 Deuxième contrainte (3.134b)

On développe (3.134b) pour obtenir :

$$0 \geq \alpha^2\left(\|\Delta\tilde{d}\|^2 - (\Delta t)^2\right) + 2\alpha\left(\tilde{d}^T\Delta\tilde{d} - t\Delta t\right) + \|\tilde{d}\|^2 - t^2. \tag{3.138}$$

On définit :

$$\frac{\delta}{4} = \left(\tilde{d}^T\Delta\tilde{d} - t\Delta t\right)^2 - \left(\|\Delta\tilde{d}\|^2 - (\Delta t)^2\right)\left(\|\tilde{d}\|^2 - t^2\right), \tag{3.139}$$

puis :

$$\boxed{\tilde{\alpha} = \frac{-\tilde{d}^T\Delta\tilde{d} + t\Delta t + \sqrt{\left(\tilde{d}^T\Delta\tilde{d}\right)^2 - \|\Delta\tilde{d}\|^2\left(\|\tilde{d}\|^2 - t^2\right)}}{\|\Delta\tilde{d}\|^2 - (\Delta t)^2}}, \tag{3.140}$$

à supposer que le dénominateur soit non nul. Une étude de chaque cas donne la formule suivante :

$$\boxed{\alpha_{t,d} = \begin{cases} \tilde{\alpha} & \text{si } \|\Delta\tilde{d}\|^2 - (\Delta t)^2 > 0, \\ 1 & \text{si } \|\Delta\tilde{d}\|^2 - (\Delta t)^2 = 0 \text{ et } \Delta t \geq 0, \\ \dfrac{t^2 - \|\tilde{d}\|^2}{2\left(\tilde{d}^T\Delta\tilde{d} - t\Delta t\right)} & \text{si } \|\Delta\tilde{d}\|^2 - (\Delta t)^2 = 0 \text{ et } \Delta t < 0, \\ 1 & \text{si } \|\Delta\tilde{d}\|^2 - (\Delta t)^2 < 0 \text{ et } \Big[\delta < 0 \text{ ou } \tilde{\alpha} < 0\Big], \\ \tilde{\alpha} & \text{sinon.} \end{cases}} \tag{3.141}$$

3.4.8.3 Troisième contrainte (3.134c)

On trouve immédiatement :

$$\boxed{\alpha_t = \begin{cases} 1 & \text{si } \Delta t \geq 0, \\ -\dfrac{1}{\Delta t} & \text{sinon.} \end{cases}} \tag{3.142}$$

3.4.9 Algorithme final et résultats en différences finies

Algorithme 23 Algorithme SOCP pour le problème de Herschel-Bulkley (3.2).

1: **Pour** $k \geq 0$ **faire**
2: Calculer l'écart de complémentarité moyen $\bar{g}^{(k)}$ (3.81).
3: Calculer F et V à partir de l'itérée courante, formules (3.93) à (3.99).
4: Calculer la matrice du problème (3.132).
5: Former les résidus $r_u^{\gamma,\mu,+}$ (3.131) et r_p^{γ} (3.115) pour l'étape affine (i.e. $\mu = \gamma = 0$ et $\Delta x^+ = \Delta s^+ = 0$), puis résoudre le système (3.132).
6: En déduire Δs^+(3.127) puis Δx^+ (3.120).
7: Calculer le pas maximal α_{max}^+ (section 3.4.8).
8: En déduire γ (3.113) puis $\mu^{(k+1)}$ (3.82).
9: Former les résidus $r_u^{\gamma,\mu,+}$ (3.131) et r_p^{γ} (3.115) pour l'étape de correction, puis résoudre le système (3.132).
10: En déduire Δs(3.127) puis Δx (3.120)
11: Calculer le pas maximal α_{max} (section 3.4.8), puis poser $\alpha = 0.99\alpha_{max}$.
12: Mettre à jour les variables :

$$u^{(k+1)} = u^{(k)} + \alpha\Delta u \tag{3.143a}$$
$$p^{(k+1)} = p^{(k)} + \alpha\Delta p \tag{3.143b}$$
$$t^{(k+1)} = t^{(k)} + \alpha\Delta t \tag{3.143c}$$
$$d^{(k+1)} = d^{(k)} + \alpha\Delta d \tag{3.143d}$$
$$\tau^{(k+1)} = \tau^{(k)} + \alpha\Delta\tau. \tag{3.143e}$$

▷ Toutes le formules pour le cas Bingham sont rassemblées dans l'annexe 3.B.
13: **Fin Pour**

La méthode complète est résumée dans l'algorithme 23. On l'initialise avec toutes les variables nulles, sauf t que l'on initialise à 1. Cela permet de strictement satisfaire les contraintes de cônes à l'itération 0. Comme précédemment, l'algorithme est implémenté en Fortran à l'aide de la bibliothèque PETSc [15, 14, 16] et les systèmes linéaires sont résolus par une méthode LU via le solveur parallèle MUMPS [5, 4].

Pour le moment seul le cas Bingham (voir annexe 3.B) a été testé. En effet l'implémentation de l'algorithme s'est révélée très instable, même pour l'écoulement de Poiseuille. Au bout de seulement quelques itérations (de l'ordre de 5), le pas maximal admissible α devient proche de 0, de l'ordre de 10^{-12} voire moins, ce qui bloque la méthode bien avant que la méthode n'ait convergé, on ne peut souvent même pas observer de début de formation de plug à ce stade. Divers tests ont été effectués pour essayer de comprendre et de résoudre ce phénomène. Des géométries de Poiseuille pour des Bingham de différentes gammes (de 1 à 100) ont été testées. Différents paramétrages de géométrie en croix ont aussi été essayés. Il convient de noter qu'il n'y a pas de divergence de la méthode, les itérées restent bornées, seulement les contraintes de cône imposées par l'algorithme engendrent rapidement un pas α trop faible pour que les itérées puissent continuer à évoluer (et ce malgré la procédure de central path). La difficulté provient du fait que le pas est calculé à partir d'un minimum sur tout le domaine. Autrement dit, si une zone très localisée sature une contrainte, tout l'algorithme est gelé. Bien que α baisse souvent à proximité des bords du domaine (et des coins dans le cas de la géométrie en croix), il n'a pas été possible d'identifier de zone particulière où le α baisse de manière privilégiée. Il semble que cela puisse arriver partout sur le domaine, parfois dans plusieurs zones à la fois et surtout dans des régions différentes pour chaque cas test. Ces observations se vérifient sur chacune des

3 contraintes de cônes dont α est issu (i.e. α_t, $\alpha_{t,d}$, α_τ) : chacune crée des zones avec des pas très faibles, pour des localisations différentes les unes des autres et entre les différents cas tests. On note tout de même que α_t (i.e. la contrainte $t \geq 0$) semble poser moins souvent problème que les deux autres.

Il a été vérifié que si l'algorithme est initialisé avec seulement une très faible variation de la solution analytique du Poiseuille, alors l'algorithme ne bloque pas. Cependant, dès que la variation est trop importante, la méthode échoue. Pour la géométrie en croix (figure 3.1.1), il a été testé de partir d'un écoulement vérifiant la formule de Poiseuille tout le long des canaux d'entrée et au milieu de la cavité, en prolongeant par 0 dans le reste de la cavité (cela permet d'avoir une solution initiale régulière et donc moins sujette aux instabilités), mais l'algorithme a tout de même échoué. Une piste pourrait être d'essayer d'initialiser l'algorithme avec le résultat donné par la méthode ADMM 17. Il faudrait cependant calculer les correspondances entres les différentes variables intervenant dans les deux algorithmes, en s'assurant que les contraintes de cônes soient bien strictement vérifiées, ceci n'a cependant pas pu être essayé faute de temps. En effet, la méthode SOCP nécessite la résolution de systèmes linéaires en (u, p), qui change de plus à chaque itération, aussi les temps de calcul peuvent être longs si l'on utilise une grille raisonnablement raffinée. Pour une grille avec 600 mailles dans une section verticale, un test de la méthode SOCP peut durer plusieurs dizaines de minutes. La méthode a été testée sur des grilles allant jusqu'à 2400 mailles dans la section verticale (dont la durée se compte en heures pour quelques itérations), avec les mêmes conclusions.

La question demeure de comprendre ce qui différencie notre cas de celui étudié par Bleyer [49]. Outre la différence de discrétisation, les cas tests utilisés par Bleyer correspondent à des géométries différentes, plus simples que la géométrie en croix. Cela pourrait partiellement expliquer la différence de résultats. Un travail théorique pour mieux comprendre le comportement de la méthode couplée à la discrétisation semble indiqué (ceci dépasse cependant le cadre de cette thèse, étant donnée l'ampleur des non-linéarités à la fois du problème de Bingham et du scaling de Todd-Nesterov). On rappelle que de plus, le principe de cette nouvelle version de la méthode (comparée à celle présentée en 2015 [50]) est que la reformulation en problème SOCP n'est que partielle. Il ne s'agit ici donc pas réellement d'un problème rentrant dans le cadre SOCP aussi est-on en droit de s'attendre à un comportement plus complexe de la méthode.

3.4.10 Implémentation éléments finis

3.4.10.1 Comparaison SOCP et ADMM

Comme évoqué en remarque 60, ces instabilités pourraient être expliquées par l'usage des moyennes à quatre points, qui pourraient mal se comporter avec les non-linéarités de la méthode. Pour tester cette idée, une seconde implémentation a été réalisée cette fois en éléments finis, en Python à l'aide de la bibliothèque FEniCSx [22, 229, 228, 3] (héritière de FEniCS). Comme suggéré par Bleyer [49], des éléments de Lagrange discontinus d'ordre 1 sont utilisés pour t, d et τ et on utilise des éléments $P_2 - P_1$ pour (u, p). On discrétise x et s en accord avec t, d et τ. L'algorithme semble plus stable dans ce contexte, au sens où certains cas tests Poiseuille ont convergé. Quelques simulations sur la géométrie en croix ont montré un début de convergence, avec un profil de vitesse satisfaisant, néanmoins Du et d restent inexploitables. On présente par exemple dans la figure 3.4.1 le champ $|Du|$ pour le cas $h = 2$, $\delta = 0.5$, $B = 5$. Les plugs sont difficiles à détecter clairement, il est impossible de les différencier des pseudo-plugs. Les petits "patchs" supposés entourer le plug central (voir figure 3.2.1) sont complètement invisibles.

Tout récemment (fin avril 2024), FEniCSx a mis à disposition les éléments *quadrature*. Le principe est de ne travailler qu'aux points où sont localisés les degrés de liberté des éléments, afin d'éviter les opérations s'apparentant à des moyennes. On renvoie à la documentation de

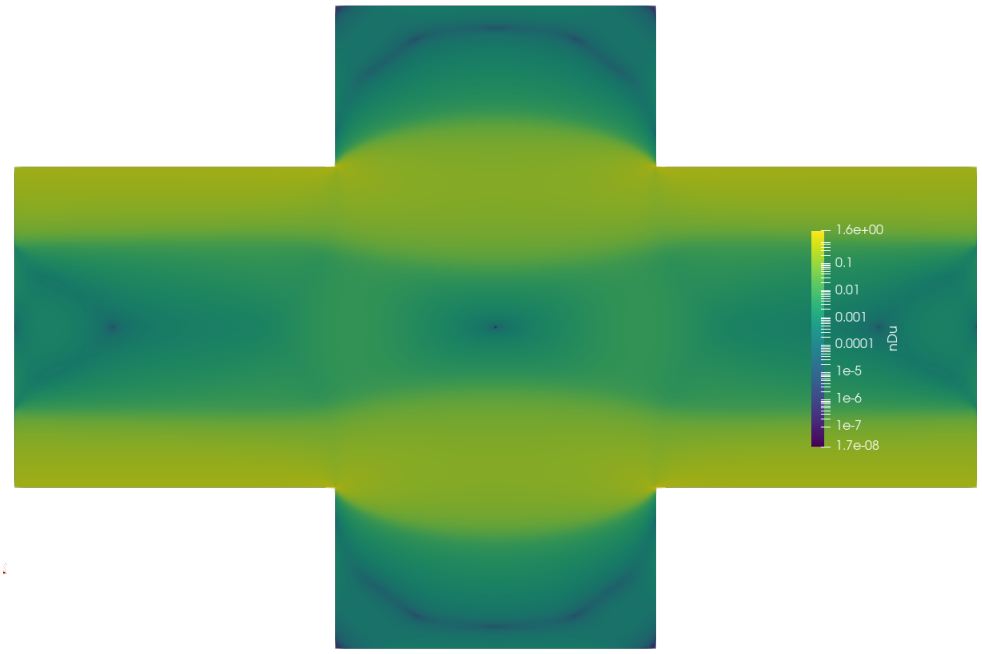


FIGURE 3.4.1 – $|Du|$ fournie par la méthodologie SOCP en utilisant des éléments de Lagrange discontinus d'ordre 1 pour x et s sur le cas test $h = 2$, $\delta = 0.5$, $B = 5$. L'approximation numérique est de mauvaise qualité, les délimitations de plugs sont floues et difficilement différentiables des pseudo-plugs.

FEniCSx pour plus de détails.

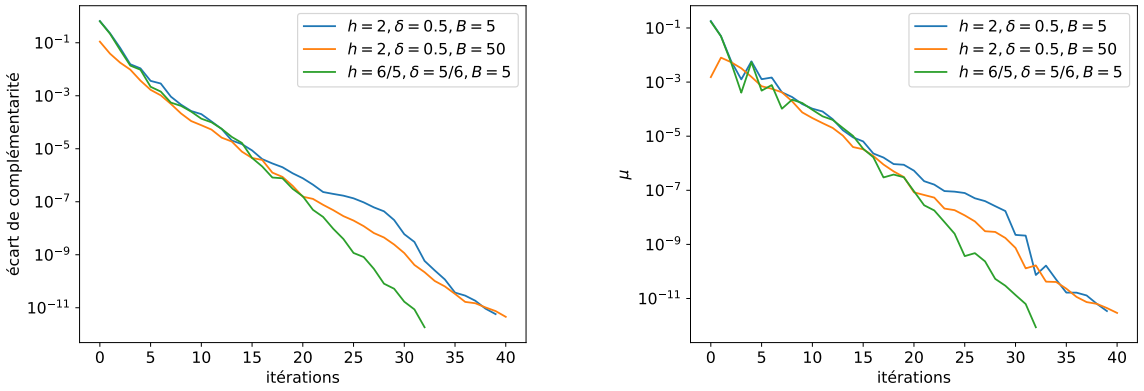
En utilisant ces éléments pour la discrétisation de x et s, on obtient une amélioration drastique des résultats. Par manque de temps, il n'a pas été possible de tester la méthode sur toute la zoologie présentée plus haut, on a cependant pu la tester pour 3 cas :

- $h = 2$, $\delta = 0.5$, $B = 5$,
- $h = 2$, $\delta = 0.5$, $B = 50$,
- $h = 6/5$, $\delta = 5/6$, $B = 5$.

Remarque 63 *On montre ici les résultats obtenus après utilisation d'une procédure de raffinement de maillage implémentée pour notre cas. Les détails sont donnés en section 3.4.10.2. Dans chaque cas, on a de l'ordre de 300 000 triangles dans chaque maillage.*

Dans les 3 cas, l'écart de complémentarité $\bar{g}^{(k)}$ descend en quelques dizaines d'itérations jusqu'à 6×10^{-12}, comme le montre la figure 3.4.2a. Il convient de noter que celui-ci peut descendre encore plus bas, cependant la méthode commence rapidement à montrer des signes d'instabilités numériques et le code renvoie assez vite des NaN. Ce comportement est à attendre : si $\bar{g}^{(k)}$ tend vers 0, cela signifie que les contraintes de cônes sont saturées et alors les matrices du scaling de Todd-Nesterov ne sont plus définies. Le temps de calcul a été comparé à une implémentation éléments finis de ADMM. Dans tous les cas, les calculs durent environ 1h20 sur un maillage donné. La méthode SOCP semble être en règle générale légèrement plus rapide, mais pas suffisamment pour tirer de conclusion significative au vu du peu de cas considérés.

La figure 3.4.2b montre que la mise à jour adaptative du paramètre de centrage μ permet bien de faire converger celui vers 0 à mesure de la convergence de l'algorithme. Ce paramètre, qui peut être vu comme une régularisation du modèle, est abaissé entre 10^{-11} et 10^{-12} en fin d'algorithme.



(a) $\bar{g}$ (b) μ

FIGURE 3.4.2 – Évolution de l'écart de complémentarité $\bar{g}$ et du paramètre de centrage μ au cours des itérations de l'algorithme 23 pour différents cas.

La figure 3.4.3 présente ensuite les champs de vitesses obtenus dans les différents cas. Ceux-ci sont d'excellente qualité, indifférentiables des résultats obtenus avec ADMM (que l'on n'affiche donc pas).

Intéressons-nous maintenant à la qualité du champ $|d|$. Là où il est connu que d fourni des délimitations de plug de meilleure qualité que $|Du|$ (voir [157, 158]), la figure 3.4.4 semble indiquer que $|Du|$ est de meilleure qualité que d. Les autres simulations tendent dans la même direction aussi on se focalise dorénavant sur $|Du|$ dans le cas SOCP et $|d|$ dans le cas ADMM.

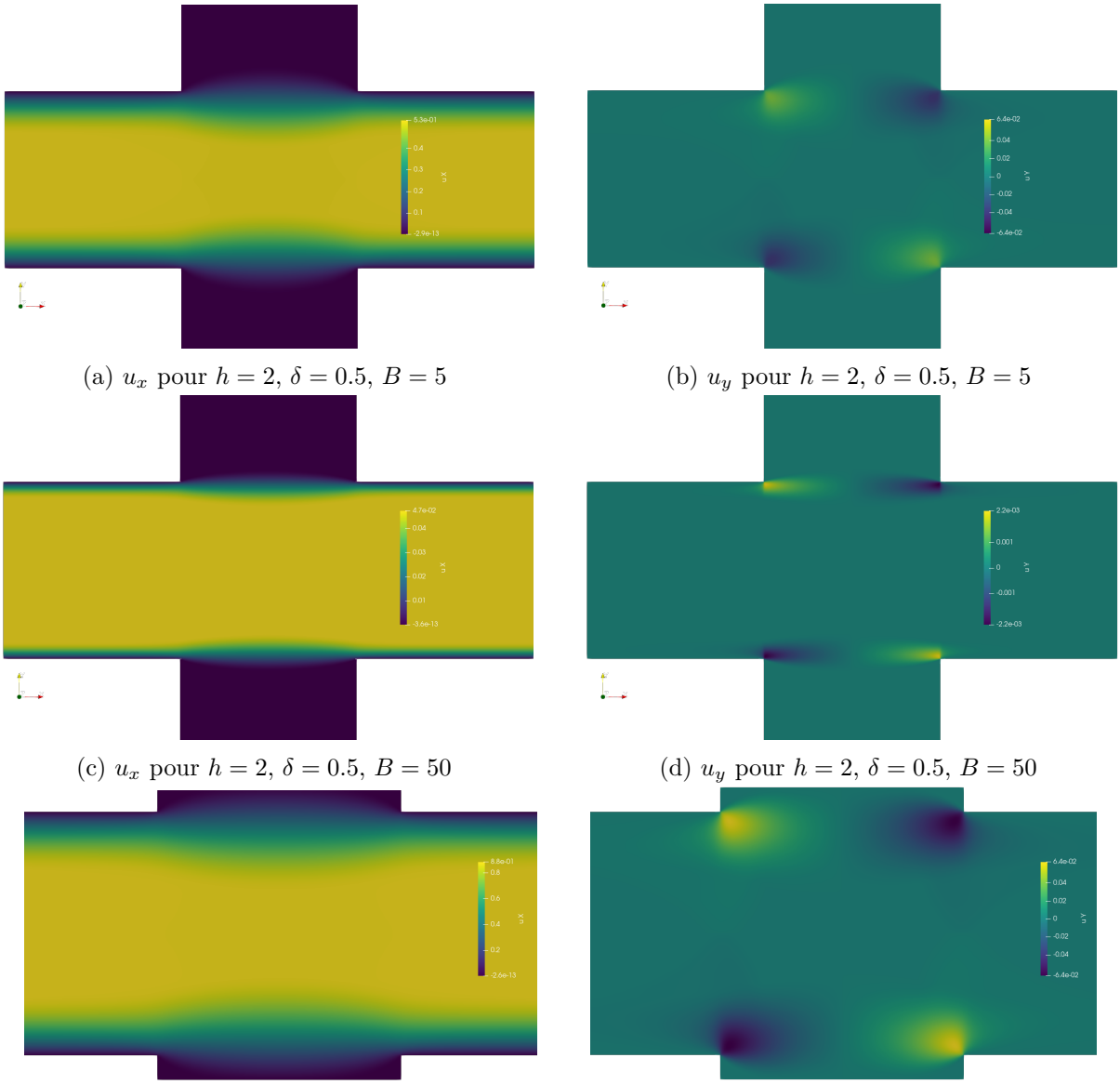
La figure 3.4.5 montre les champs de déformation que l'on obtient via la méthodologie SOCP et les compare à ceux obtenus via l'implémentation éléments finis de ADMM. Les champs obtenus sont de très bonne qualité, avec une chute drastique de la norme du tenseur au niveau des limites de plug. Il semble que la méthode SOCP fournisse un champ de meilleure qualité que ADMM, qui montre de petites imperfections en bordures de plug, notamment autour des petits "patchs" entre les deux plugs centraux.

Remarque 64 *On attire l'attention sur l'amélioration apportée par la procédure de raffinement de maillage que l'on présente en section 3.4.10.2, on renvoie notamment à la figure 3.4.9.*

En comparant finement avec les résultats obtenus en différences finies (voir par exemple la figure 3.2.3), on peut cependant noter que ces derniers sont légèrement de meilleure qualité. Il semble que l'usage des éléments finis réduise légèrement la qualité de la délimitation des plugs dans le cas ADMM. En différences finis, ADMM semble donner des plugs de meilleure qualité que la méthode SOCP, ce qui se voit particulièrement sur les petits patchs qui sont légèrement flous dans le cas SOCP (on renvoie au zoom de la figure 3.4.4b).

Remarque 65 *Il convient de noter que les mailles les plus petites obtenues après raffinement de maillage sont 2 à 3 fois plus petites que les mailles utilisées précédemment en différence finie. La différence de performance est donc d'autant plus notable, en faveur des différences finies.*

Pour tester plus précisément la qualité de ces délimitations, on montre en figure 3.4.6 l'évolution des délimitations de plug pour différentes valeurs de seuil appliqués sur le champ de déformation, pour les méthodes ADMM et SOCP en éléments finis. On ne montre ici que le cas $h = 2, \delta = 0.5, B = 5$ car les deux autres cas donnent des résultats similaires. La méthode SOCP donne des limites de plug physiquement cohérentes pour un seuil allant jusqu'à 10^{-8}. À partir de



(a) u_x pour $h = 2$, $\delta = 0.5$, $B = 5$

(b) u_y pour $h = 2$, $\delta = 0.5$, $B = 5$

(c) u_x pour $h = 2$, $\delta = 0.5$, $B = 50$

(d) u_y pour $h = 2$, $\delta = 0.5$, $B = 50$

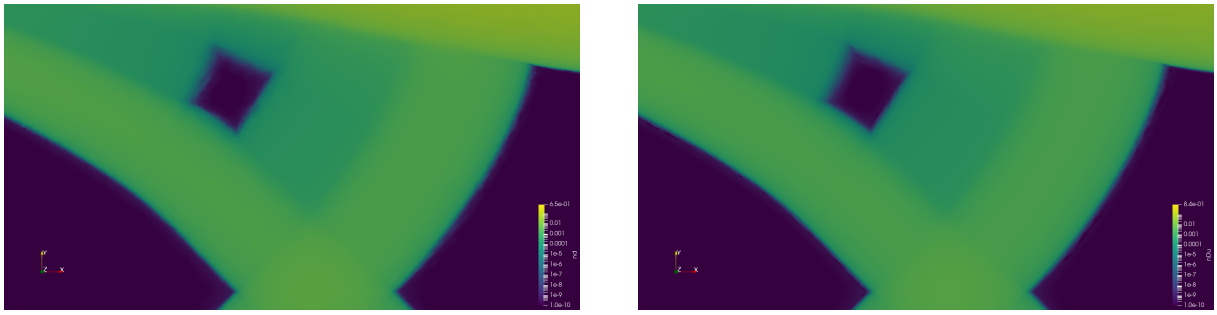
(e) u_x pour $h = 6/5$, $\delta = 5/6$, $B = 5$

(f) u_y pour $h = 6/5$, $\delta = 5/6$, $B = 5$

FIGURE 3.4.3 – Champs de vitesse obtenus par la méthodologie SOCP, pour différents cas.

10^{-9}, les contours commencent à être bruités (particulièrement dans les patchs) et pour 10^{-10} le résultat devient clairement incohérent. Pour ADMM, dès 10^{-6} on observe déjà beaucoup de bruit dans les contours, on a donc a priori deux à trois ordres de grandeur de précision supérieure de SOCP vis-à-vis de ADMM. Il convient de noter qu'en différences finies, on peut aisément seuiller à 10^{-10} sans bruit dans le résultat, un travail futur demanderait donc de rigoureusement superposer les limites obtenues par SOCP-EF et ADMM-DF pour comparer les résultats. On montre aussi dans les figures (c) à (f) de 3.4.6 l'évolution de ces contours avec le maillage. On constate que la valeur du champ de déformation chute en seulement quelques triangles, avec grossièrement une chute d'un ordre de grandeur d'un triangle à l'autre. On s'attend donc à ce que des calculs sur maillage légèrement plus fin puisse permettre d'affiner drastiquement ces contours.

Pour finir, Bleyer [49] propose d'utiliser non pas le champ de déformation mais plutôt le



(a) $|d|$ (b) $|Du|$

FIGURE 3.4.4 – Comparaison de $|Du|$ et $|d|$ obtenus par la méthode SOCP dans le cas $h = 6/5$, $\delta = 5/6$, $B = 5$.

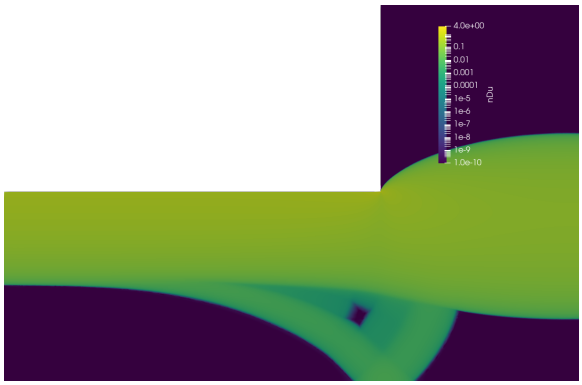
tenseur des contraintes (évalué à partir de la loi de Bingham sur la base des quantités calculées par la méthode) afin de déterminer les limites de plug. On compare dans la figure 3.4.7 les plugs obtenues via cette méthode et différents seuils sur le champ de déformation. Les délimitations obtenues se rapprochent des limites obtenues en seuillant le champ de déformation à 10^{-6}. Il a été vérifié que des variations d'ordre 10^{-8} sur τ_y ne modifient pas les délimitations obtenues, ce qui tend à indiquer que ce critère pourrait être plus stable que seuiller sur le champ de déformation. Néanmoins étant donné que cela se rapproche d'un seuil à 10^{-6} on peut craindre que la taille des plugs soit surestimée, on note par exemple dans la pointe présentée en figure 3.4.7b des irrégularités physiquement absurdes.

3.4.10.2 Raffinement de maillage

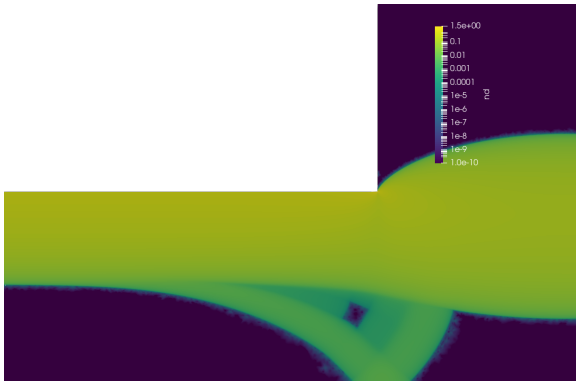
Pour les simulations éléments finis, il est commun d'utiliser une procédure de raffinement de maillage. L'objectif est double : obtenir une meilleure précision des délimitations de plug (sans avoir à ajouter un nombre trop important de mailles) et améliorer le conditionnement des systèmes linéaires impliqués dans la résolution du problème. Pour cela, on cherche à avoir un maillage fin proche des délimitations de plug et plus grossier ailleurs. On utilise une procédure de raffinement de maillage basée sur celle de Saramito et Roquet [222]. On commence avec un maillage triangulaire structuré (issu de quadrangles coupés en deux, voir figure 3.4.8a). On calcule ensuite la hessienne de la fonctionnelle :

$$\phi = \sqrt{\eta |Du|^2 + \tau_y |Du|}, \tag{3.144}$$

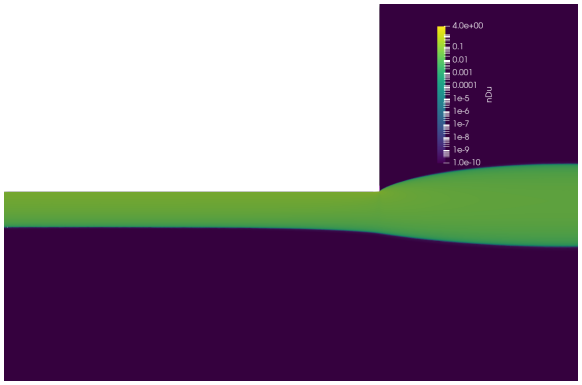
afin de construire une distance dans l'espace. Saramito et Roquet donnent directement cette distance au logiciel de raffinement de maillage anisotropique BAMG [126], qui va permettre de mailler d'autant plus les zones où les valeurs propres de la hessienne de ϕ sont importantes, permettant de mailler les délimitations des zones rigides (voir [222] pour plus de détails) et d'aligner les mailles avec les limites de plug. Cependant, on a constaté que l'usage de la fonction ϕ telle quelle donnait des résultats peu satisfaisants dans notre cas, à cause des différences importantes d'échelles des valeurs de la hessienne, notamment dû à la difficulté des coins de la géométrie en croix. Avant de raffiner le maillage, on ajoute donc à leur procédure un lissage sur la distribution spatiale des valeurs propres de cette distance. Plus précisément, on extrait les valeurs propres de H_ϕ sur tous les degrés de liberté du maillage. On prend l'inverse des valeurs propres puis on les ordonne dans l'ordre croissant. Enfin, on les remplace par leur rang dans la distribution spatiale de ces valeurs (en pratique, on peut simplement utiliser des sommes cumulées). On renormalise ces valeurs afin de les faire parcourir un intervalle $[h_{min}, h_{max}]$, qui donne les tailles minimales



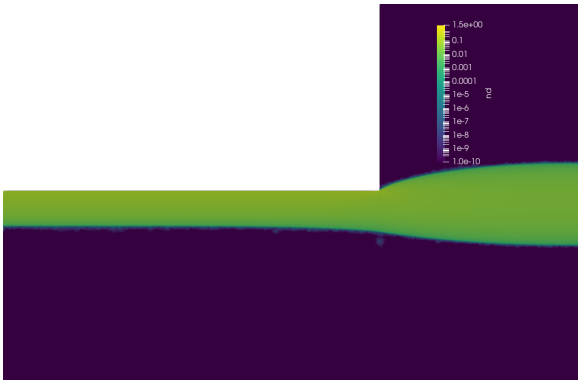
(a) SOCP, $h = 2$, $\delta = 0.5$, $B = 5$



(b) LA, $h = 2$, $\delta = 0.5$, $B = 5$



(c) SOCP, $h = 2$, $\delta = 0.5$, $B = 50$



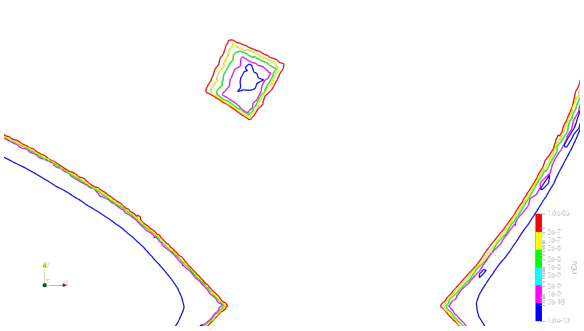
(d) LA, $h = 2$, $\delta = 0.5$, $B = 50$

(e) SOCP, $h = 6/5$, $\delta = 5/6$, $B = 5$

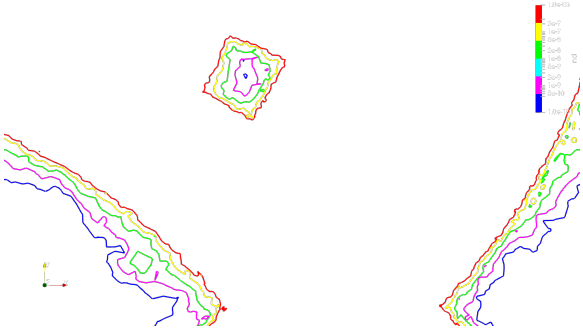
(f) LA, $h = 6/5$, $\delta = 5/6$, $B = 5$

FIGURE 3.4.5 – Comparaison des champs de déformation obtenus via SOCP et ADMM pour les 3 jeux de paramètres.

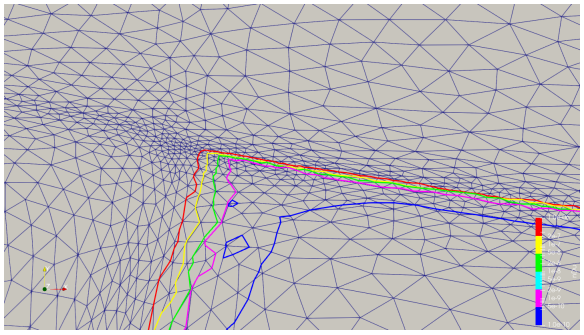
et maximales de mailles que l'on souhaite obtenir. On remplace finalement les valeurs propres initiales par l'inverse des valeurs obtenues par la procédure ci-dessus. La distance obtenue privilégie les mêmes directions que la distance d'origine, avec cependant des variations spatiales moins brutales. Pour finir, cette distance ainsi que la géométrie en croix sont données au logiciel libre MMG [13, 77, 84] qui effectue alors le raffinement de maillage. On présente en figure 3.4.8 un exemple de maillage obtenu après 3 itérations de la procédure de raffinement, où on a utilisé ADMM pour résoudre le problème. On constate que la procédure permet de particulièrement mailler les délimitations de zones rigides, on a même un maillage significatif à la limite des pseudo-plugs. Au contraire, les zones à l'intérieur des plugs ou à l'intérieur des zones fluides sont



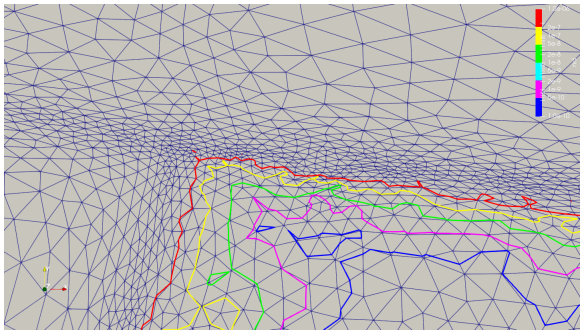
(a) SOCP



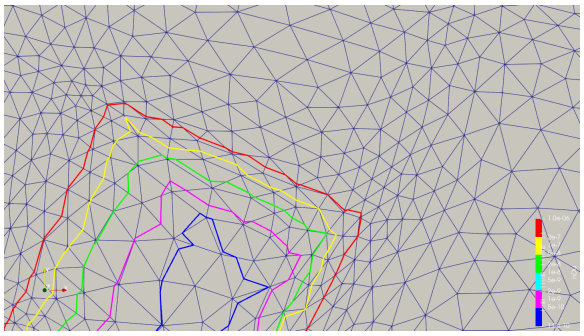
(b) LA



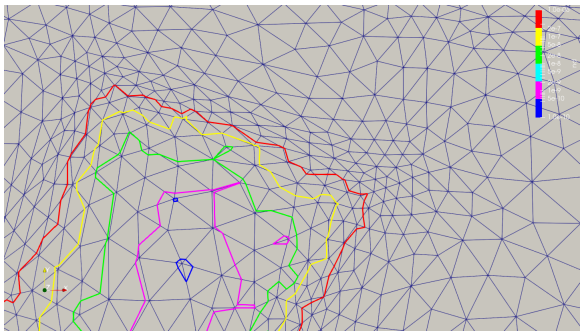
(c) SOCP avec maillage, zoom sur l'amande centrale



(d) LA avec maillage, zoom sur l'amande centrale



(e) SOCP avec maillage, zoom sur le patch

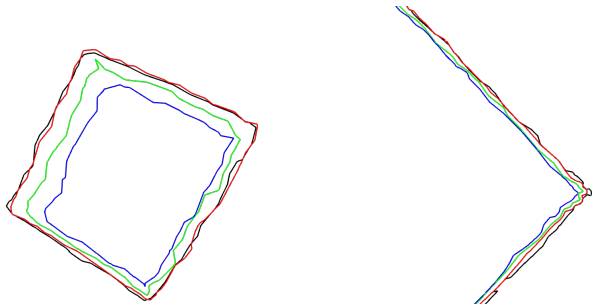


(f) LA avec maillage, zoom sur le patch

FIGURE 3.4.6 – Comparaison des délimitations de plug obtenues via différents seuillages sur le champ de déformation, dans le cas $h = 2$, $\delta = 0.5$, $B = 5$. Les seuils testés sont $10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}, 10^{-10}$, respectivement de couleurs rouge, jaune, verte, magenta et bleue. Des zooms sont faits sur différentes portions du domaine, et on montre le maillage en fond des figures (c) à (f).

plus faiblement maillées. On note que MMG propose de forcer la *gradation* du maillage, i.e. le ratio maximal de longueur entre deux arrêtes du maillage. Cette dernière est fixée par défaut à 1.3, afin d'éviter d'avoir des triangles trop aplatis, ce qui nuirait à la qualité de la discrétisation.

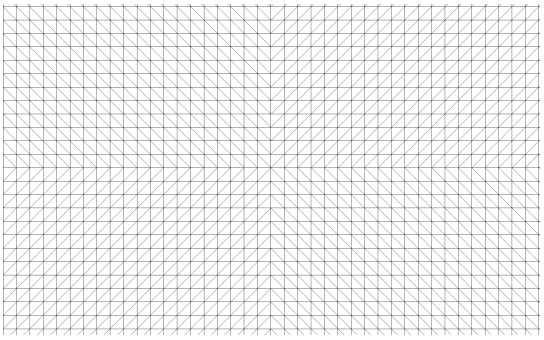
On montre en figure 3.4.9 l'impact bénéfique de cette procédure de raffinement de maillage. La procédure permet d'avoir des délimitations beaucoup plus précises des plugs, quelque soit la méthode utilisée, alors que le nombre de triangles est similaire. Toutefois, si l'on reprend la figure 3.4.6, plus particulièrement la superposition des contours du champ de déformation et du maillage, on observe qu'entre les pseudo-plugs et les plugs, le maillage est légèrement plus grossier, entraînant une légère baisse de la qualité des délimitations des zones rigides. Une



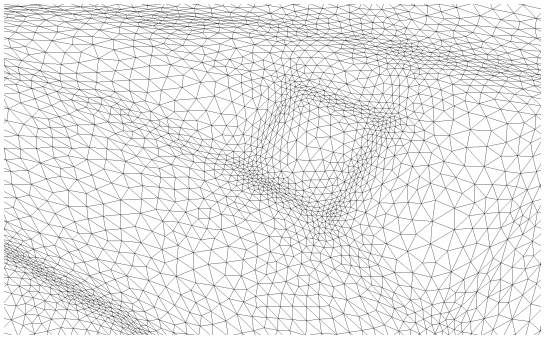
(a) (b)

FIGURE 3.4.7 – Comparaison des délimitations de plug obtenues via différents seuillages sur le champ de déformation et via l'évaluation du tenseur des contraintes, dans le cas $h = 2$, $\delta = 0.5$, $B = 5$. Les seuils testés sont $10^{-6}, 10^{-7}, 10^{-8}$, respectivement de couleurs rouge, verte, et bleue. La limite obtenue via le tenseur des contraintes est en noir. Des zooms sont faits sur différentes portions du domaine.

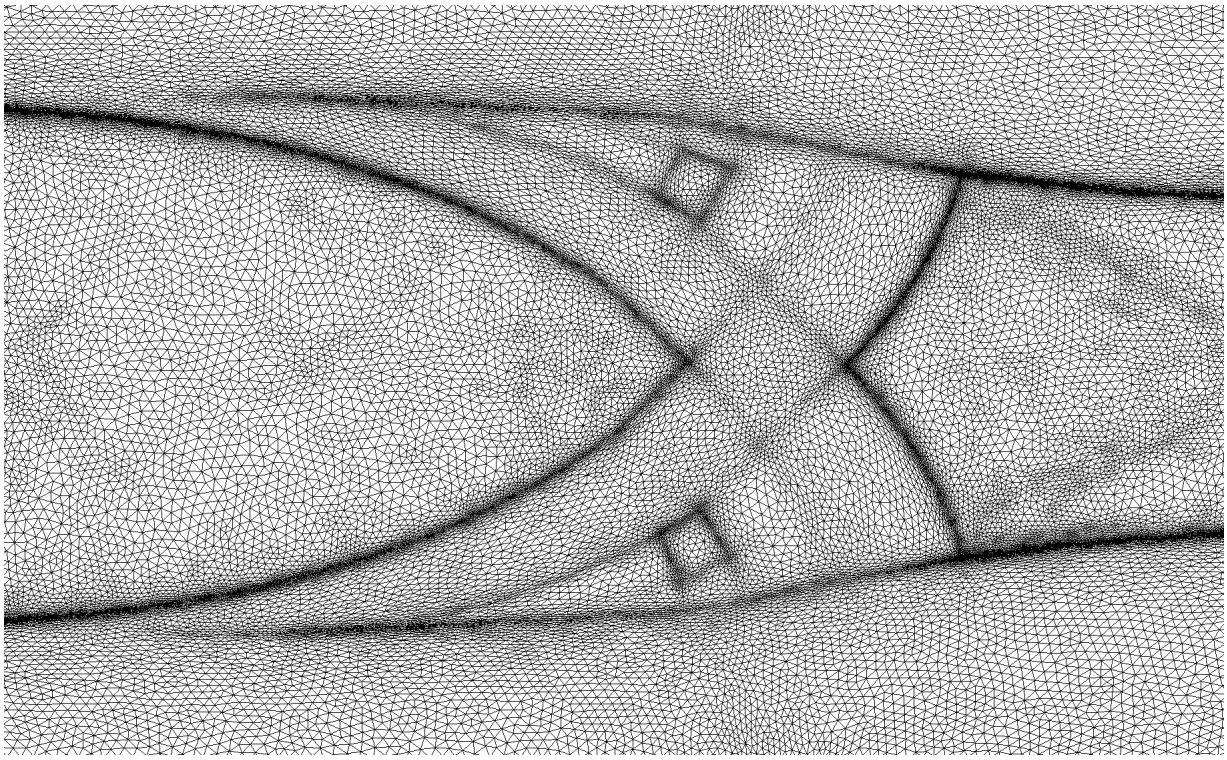
piste prometteuse consisterait soit à utiliser une lissage non-linéaire des valeurs propres de la hessienne de ϕ, afin d'accentuer la densité de maillage dans ces zones sans impacter les autres, soit à appliquer la procédure de manière locale en espace, de sorte à ce que la présence d'autres échelles de valeurs dans la géométrie n'impacte pas le maillage de ces zones.



(a) maillage structuré initial

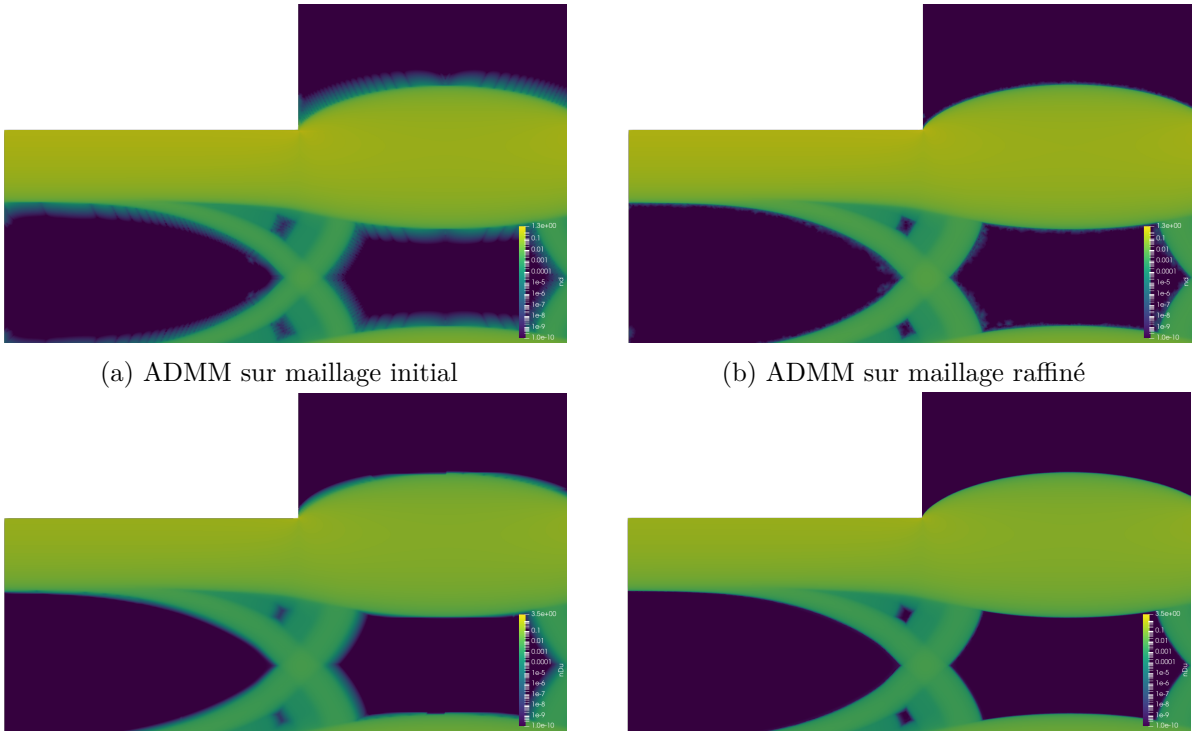


(b) après 3 raffinements de maillage (zoom)



(c) après 3 raffinements de maillage

FIGURE 3.4.8 – Comparaison entre le maillage structuré initial et le maillage après 3 itérations de la procédure de raffinement dans le cas $h = 2$, $\delta = 0.5$, $B = 5$.



(a) ADMM sur maillage initial

(b) ADMM sur maillage raffiné

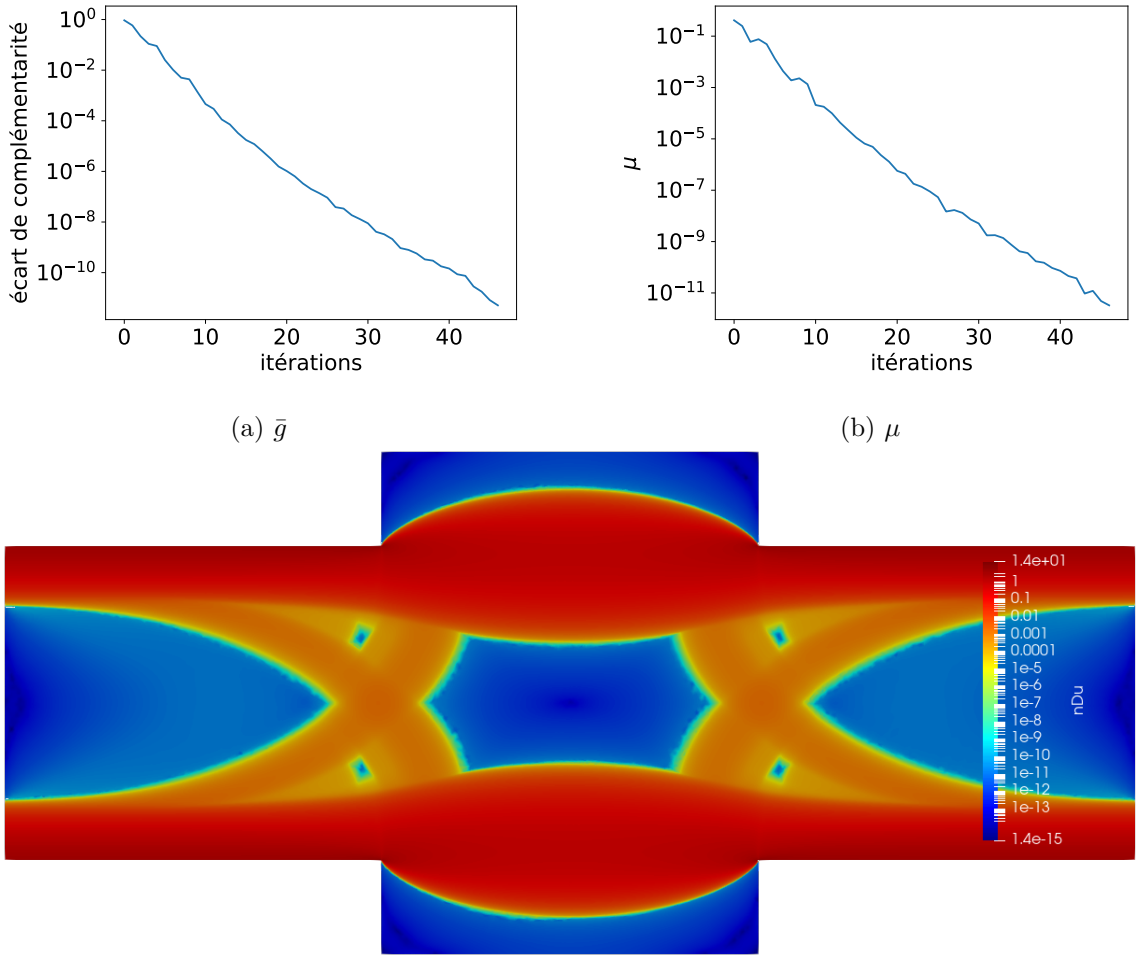
(c) SOCP sur maillage initial

(d) SOCP sur maillage raffiné

FIGURE 3.4.9 – Comparaison des résultats des algorithmes entre le maillage structuré initial et le maillage après 3 itérations de la procédure de raffinement dans le cas $h = 2$, $\delta = 0.5$, $B = 5$, pour ADMM et la méthodologie SOCP.

3.4.10.3 Un premier test sur Herschel-Bulkley

Pour terminer, on applique la méthode sur un cas Herschel-Bulkley issu de la section 3.3. Pour cela, on choisit la formulation usuelle $G(u,t,\tau) = \|Du\|^{n-1}Du$. Cette fonction n'étant pas dérivable en 0, on propose, sur les conseils de Bleyer [49], d'utiliser $\|Du^k\|^{n-1}$ obtenu à l'itération précédente pour ne garder comme inconnu à l'étape suivante que Du. Vu autrement, on modifie la viscosité apparente en utilisant l'étape précédente, pour se ramener au même type de système que dans le cas Bingham. Les résultats sont présentés en figure 3.4.10.



(a) $\bar{g}$ (b) μ

(c) $|Du|$

FIGURE 3.4.10 – Résultat du cas Herschel-Bulkley R24, L=6 résolu via la méthode SOCP (après raffinement de maillage). On présente l'évolution de l'écart de complémentarité $\bar{g}$ et du paramètre de centrage μ au cours des itérations, ainsi que le champ de déformation. Il convient de noter que, contrairement aux illustrations des figures précédentes, l'échelle de couleurs a été choisie de sorte à montrer des valeurs jusqu'à 10^{-15} afin de pouvoir comparer aux résultats produits par ADMM en section 3.3.

Comme précédemment, l'écart de complémentarité ainsi que le paramètre de centrage convergent vers 0 en quelques dizaines d'itérations. Le temps de calcul est aussi similaire à celui des calculs précédemment faits en Bingham, ici environ 1h30. Pour finir, le champ de déformation semble être de bonne qualité, néanmoins inférieure aux résultats obtenus via ADMM en différences finies (voir figure 3.3.3b). Bien que l'on observe une réduction des artefacts à l'intérieur des plugs

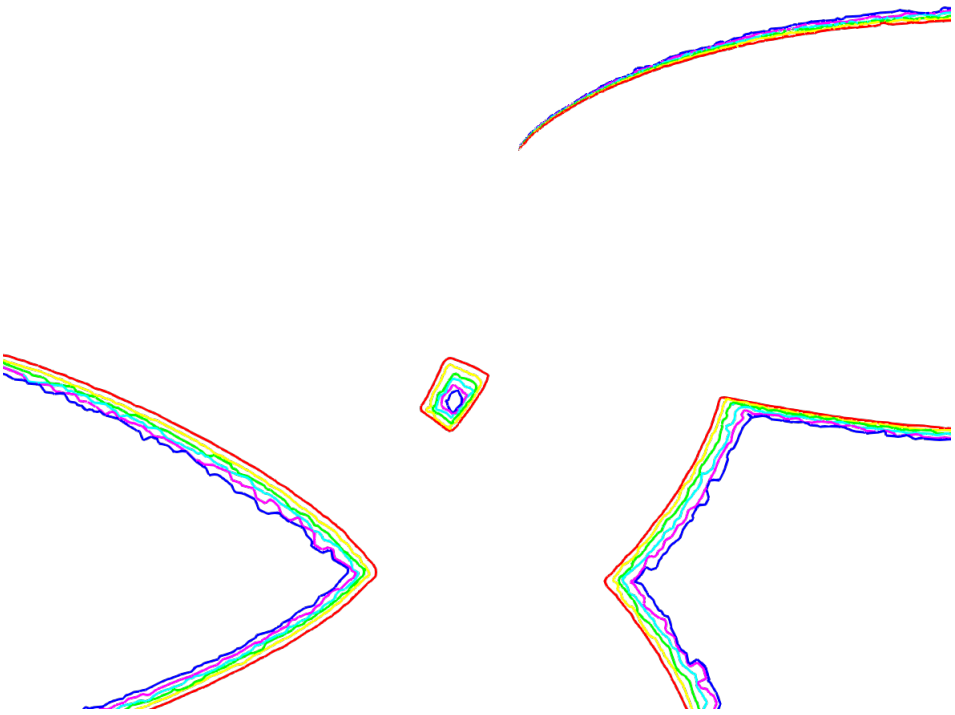


FIGURE 3.4.11 – Comparaison des délimitations de plug obtenues via différents seuillages sur le champ de déformation du cas test Herschel-Bulkley R24, L=6 résolu via la méthode SOCP. Les seuils testés sont $10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}, 10^{-9}, 10^{-10}$ respectivement de couleur rouge, jaune, vert, cyan, magenta et bleu foncé.

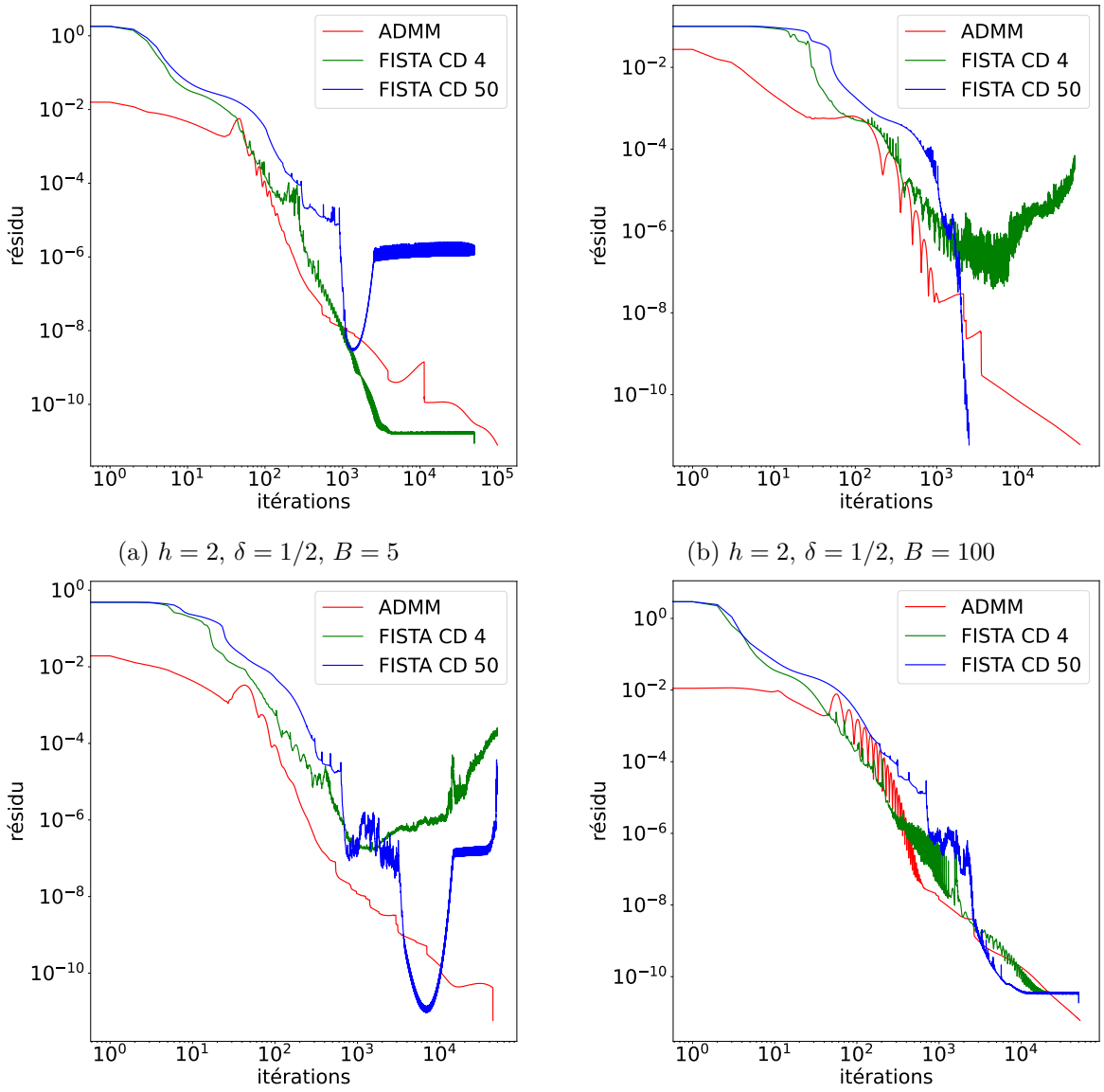
d'entrée et de sortie, les délimitations sont plus étendues. La figure 3.4.11 montre la variabilité des limites de plugs avec le seuil effectué sur le champ de déformation. Faute de temps, il n'a pas été possible de procéder à plus de tests, il ne s'agit donc ici que des conclusions issues d'un seul cas-test. Ces résultats sont néanmoins très prometteurs et encouragent à procéder à des tests plus poussés pour explorer plus précisément le potentiel de cette méthode.

Annexe du chapitre 3

3.A Figures et tableaux complémentaires sur les comparaisons ADMM/FISTA

cas test	Résidu	ADMM	FISTA CD 4	FISTA CD 4 RESTART	FISTA CD 400
h=2,δ=1/2, B=2	3.9e-11	35337	45729	29981	3954
	6.0e-12	56948	-1	-1	-1
h=2,δ=1/2, B=20	3.2e-08	3533	10529	1229	7466
	6.0e-12	42029	-1	1481	14622
h=2,δ=1/4, B=20	1.5e-07	1696	3319	1305	4436
	2.6e-11	50039	-1	1959	8621
	6.0e-12	50053	-1	2041	-1
h=2,δ=1/4, B=50	6.6e-06	1262	1809	1724	6311
	1.9e-09	7953	-1	2664	9144
	6.0e-12	17375	-1	-1	14720
h=6/5,δ=5/6, B=2	8.8e-11	19425	14331	12635	8220
	6.0e-12	39744	-1	-1	-1
h=6/5,δ=5/6, B=100	1.2e-11	53358	44911	9644	27380

TABLE 3.A.1 – Reproduction du tableau 3.2.1 où l'on remplace le nombre d'itérations nécessaires pour atteindre un résidu donné par le nombre de systèmes linéaires résolus, i.e. le nombre d'itérations cumulées de l'algorithme 18. On constate que les valeurs sont modifiées mais pas les rapports entre les algorithmes.



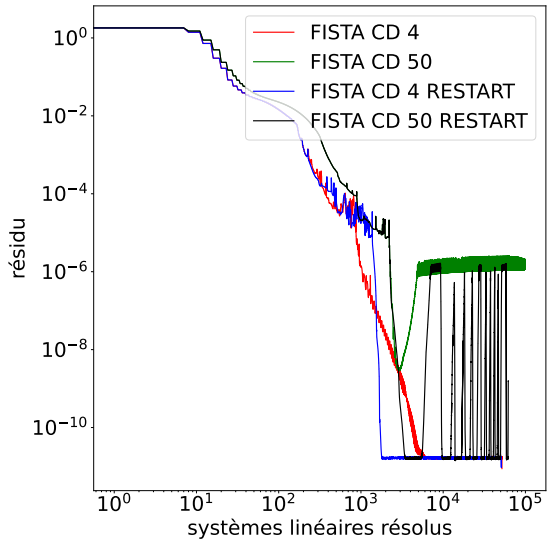
(a) $h = 2$, $\delta = 1/2$, $B = 5$

(b) $h = 2$, $\delta = 1/2$, $B = 100$

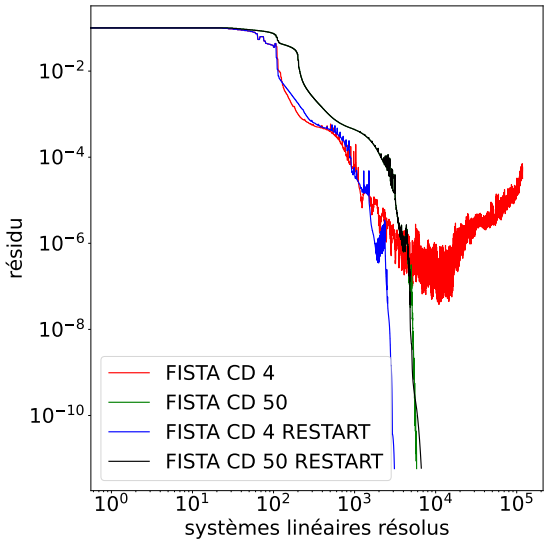
(c) $h = 2$, $\delta = 1/4$, $B = 20$

(d) $h = 6/5$, $\delta = 5/6$, $B = 5$

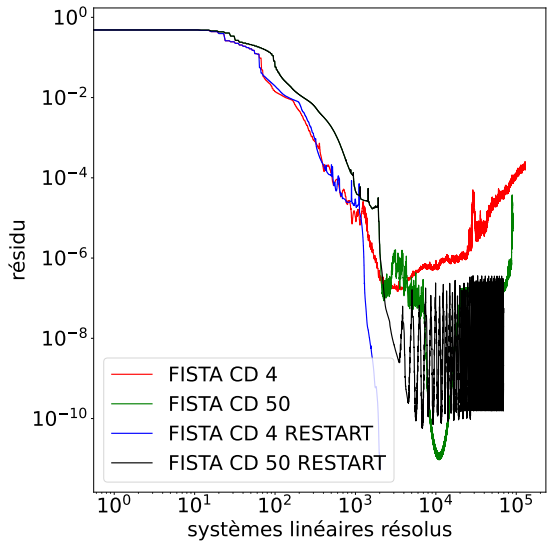
FIGURE 3.A.1 – Évolution du résidu au cours des résolutions de systèmes linéaires pour ADMM, FISTA CD 4 et FISTA CD 50 pour différents paramètres.



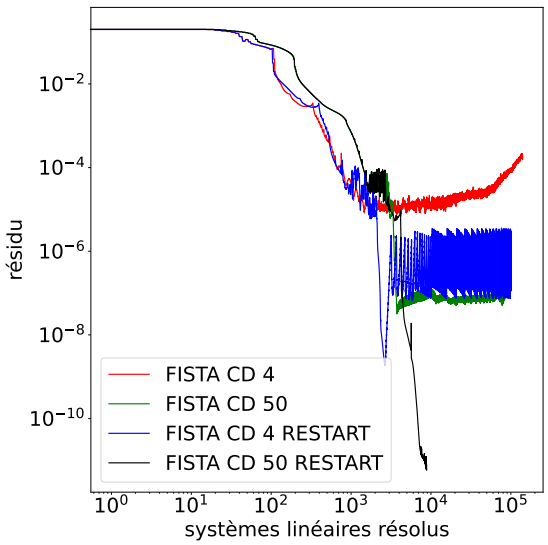
(a) $h = 2$, $\delta = 1/2$, $B = 5$



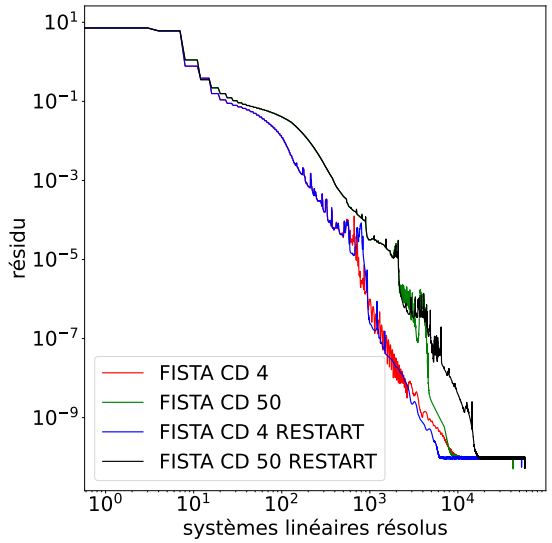
(b) $h = 2$, $\delta = 1/2$, $B = 100$



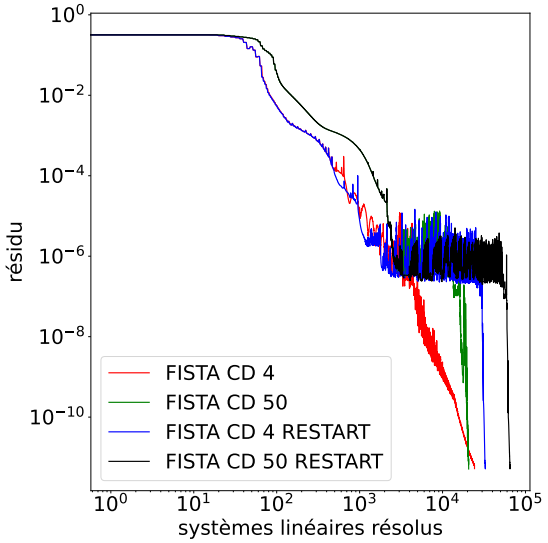
(c) $h = 2$, $\delta = 1/4$, $B = 20$



(d) $h = 2$, $\delta = 1/4$, $B = 50$



(e) $h = 6/5$, $\delta = 5/6$, $B = 2$



(f) $h = 6/5$, $\delta = 5/6$, $B = 50$

FIGURE 3.A.2 – Évolution du résidu au cours des résolutions de systèmes linéaires pour ADMM, ainsi que FISTA CD 4 et FISTA CD 50, avec et sans restart, pour différents paramètres.

3.B Système de Newton pour la méthode SOCP dans le cas Bingham

Ici $G(u,t,\tau) = D(u)$, le système (3.132) se réécrit :

$$\begin{pmatrix} -2\eta M_{divT}M_D + M_{BC} - \sqrt{2}\tau_y M_{divT}M_{ext}\left(\bar{F}^{-2}\right)^{-1} M_{int}M_D & M_\nabla \\ M_{divV} & 0 \end{pmatrix} \begin{pmatrix} \Delta u \\ \Delta p \end{pmatrix} = \begin{pmatrix} r_f^{\gamma,\mu,+} \\ r_p^\gamma \end{pmatrix}, \tag{3.145}$$

où :

$$r_f^{\gamma,\mu,+} = r_f^\gamma - \sqrt{2}\tau_y M_{divT}M_{ext}\left(\bar{F}^{-2}\right)^{-1} M_{int} r_d^{\gamma,\mu,+}, \tag{3.146}$$

$$r_d^{\gamma,\mu,+} = r_d^\gamma + \left(0, M_{ext}\right)F^{-1}V^{-1}\left(\mu e - V^2 e - F\begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix}\Delta x^+ \circ F^{-1}\begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix}\Delta s^+\right), \tag{3.147}$$

$$r_f^\gamma = (1-\gamma)\left(f + u_{BC} + 2\eta M_{dT}M_D u + \sqrt{2}\tau_y M_{dT}\tau - M_\nabla p - M_{BC}u\right), \tag{3.148}$$

$$r_p^\gamma = -(1-\gamma)M_{dV}u, \tag{3.149}$$

$$r_d^\gamma = (1-\gamma)\left(d - M_D u\right), \tag{3.150}$$

On récupère $\Delta\tau$ et Δx via :

$$\Delta\tau = M_{ext}\left(\bar{F}^{-2}\right)^{-1} M_{int}M_D\Delta u - M_{ext}\left(\bar{F}^{-2}\right)^{-1} M_{int} r_d^{\gamma,\mu,+}, \tag{3.151}$$

$$\Delta x = -\begin{pmatrix} 1 & 0 \\ 0 & M_{ext} \end{pmatrix} F^{-2}\begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix}\Delta s + \begin{pmatrix} 1 & 0 \\ 0 & M_{ext} \end{pmatrix} F^{-1}V^{-1} r_x^{\mu,+}, \tag{3.152}$$

$$r_x^{\mu,+} = \mu e - V^2 e - F\begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix}\Delta x^+ \circ F^{-1}\begin{pmatrix} 1 & 0 \\ 0 & M_{int} \end{pmatrix}\Delta s^+. \tag{3.153}$$

Chapitre 4

Problème inverse sur un modèle de lubrification

Dans ce chapitre, on remplace les équations complètes par un modèle de lubrification introduit récemment par Vigneaux et al. [244] dans le cadre d'un écoulement dans une cavité confinée. On obtient alors un problème 1D bien plus rapide à résoudre que les problèmes vus précédemment. On s'intéresse alors au problème inverse : étant donnée une observation éventuellement bruitée d'un écoulement viscoplastique, on cherche à retrouver les paramètres physiques dont ledit écoulement est issu.

Il s'agit d'une version étendue de l'article [39] publié le 27 février 2024 dans Rheologica Acta. En particulier, la discussion sur l'identifiabilité 4.2 et l'étude détaillée des résultats de l'analyse en composantes principales 4.4.1 n'ont pas été présentées dans l'article.

4.1 Problème direct

4.1.1 Motivation

La demande croissante en minerais rend crucial le traitement des déchets des sites d' extraction. Une des techniques développées par l'industrie minière consiste à remplir les vides générés par l'excavation en utilisant un mélange adjoint de ciment ; on parle de *cemented paste backfill (CPB)*. Le CPB a été documenté en Allemagne dans les années 1980 [144]. Cette technique est désormais de plus en plus utilisée à travers le monde, on peut notamment citer l'Australie, le Canada ou encore la Chine [256]. Le mélange est obtenu à partir des résidus de forage qui sont mélangés avec de l'eau et du ciment. La pâte obtenue est transportée du site de forage vers les cavités via des conduites. L'ajout du liant hydraulique qu'est le ciment, est le point clé de la stabilité du CPB. Parmi les avantages de cette technique, on compte :

- la possibilité de réutiliser les déchets de forage en les réintroduisant dans la mine. Cela permet d'éviter d'utiliser des espaces supplémentaires en surface et donc les potentiels impacts environnementaux associés [242],[12].
- Remplir les cavités souterraines issues du forage avec ces mélanges entraîne une consolidation du sol, entraînant deux avantages : (i) le risque d'affaissement [139] est diminué et (ii) il n'y a plus de nécessité de laisser des piliers inexploités, comme il était d'usage pour garantir la stabilité, puisque l'on peut reboucher l'intégralité de la cavité avec la pâte cimentée (comme pour la méthodologie *cut-and-fill*). Une meilleure exploitation du site minier peut donc être attendue.

L'implémentation pratique du CPB est un processus complexe dans lequel il est nécessaire d'optimiser différents aspects : la composition de la pâte (caractéristiques chimiques et minéra-

logiques), les propriétés physiques résultantes en lien avec la phase de transport (plus de fluidité facilite le transport) et finalement la phase de consolidation (où une importante résistance mécanique est nécessaire pour une bonne stabilité du volume comblé) [256].

On se concentre ici plus spécifiquement sur la modélisation et la simulation de la phase de remplissage où l'on considère l'écoulement d'un matériau viscoplastique dans un *domaine borné*. On renvoie à [244] pour une description plus détaillée de la modélisation, où il est montré qu'un modèle de lubrification viscoplastique permet de rendre compte avec succès d'expériences reproduisant de tels écoulements, pour des matériaux typiques de CPB. À partir de ce modèle, on présente une nouvelle méthodologie permettant d'estimer les paramètres du modèle sur la base d'un problème inverse utilisant des *observations* de la hauteur de la surface de la pâte au cours de l'écoulement. L'objectif ici est donc de proposer un algorithme rapide permettant de simuler un écoulement de pâte et de déterminer ses propriétés viscoplastiques, typiquement la viscosité (ou consistance), le paramètre de seuil et la puissance n associés au modèle rhéologique de la loi de Herschel-Bulkley. De fait, la rhéologie est une des propriétés les plus importantes d'un matériau utilisé en CPB [256],[213]. Une fois ces paramètres connus, on peut alors résoudre le problème direct pour produire des simulations à des temps plus avancés et optimiser le processus de remplissage de la cavité.

Pour résoudre le problème inverse plus rapidement, on développe ce que l'on appelle communément un *métamodèle* (on parle aussi de *surrogate model*), c'est-à-dire une approximation (cependant précise) du modèle de lubrification (sous sa forme originelle d'EDP) qui est significativement plus rapide à évaluer que la résolution de l'EDP. En effet, cette dernière ne possède pas de solution analytique et requiert donc une résolution numérique qui revêt un certain coût. Le design de l'algorithme rejoint celui de précédents travaux de Blatman et Sudret [48] et de Hawchar et al. [125], mais semble n'avoir été que rarement adapté au contexte des EDP. On développe ici une méthodologie complète et spécifiquement adaptée au travail de Vigneaux et al. [244]. On note que le code du métamodèle est disponible en accès libre sur le site Zenodo : https://zenodo.org/records/8377205.

4.1.2 Le problème EDP

Le modèle de départ est un modèle de lubrification pour des écoulements de fluides à seuil dans une géométrie fermée. Il est issu des travaux de Balmforth et al. (2006) [20], où il a été développé pour des géométries ouvertes (voir aussi les travaux de Liu et Mei (1989) [153], Coussot et al. (1996) [74], Huang et Garcia (1998) [132]). Il a par la suite été étudié dans le cas d'une cavité confinée par Vigneaux et al. (2023) [244]. On renvoie à [20] et [244] pour plus de détails.

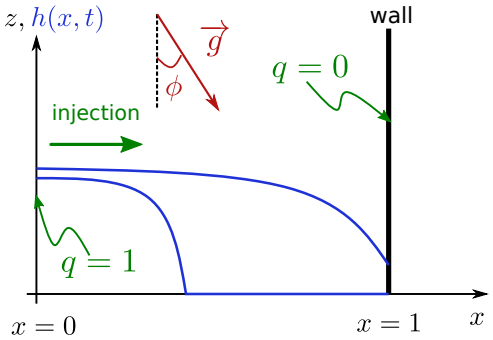


FIGURE 4.1.1 – Modèle 1D, variables adimensionnées. La courbe bleue correspond à la hauteur h du matériau à deux instants successifs (l'écoulement va de la gauche vers la droite). La gravité est inclinée d'un angle ϕ puisque l'axe x est supposé incliné par rapport à l'horizontale, afin de s'aligner avec le sol de la cavité.

On considère une cavité fermée en 1D, inclinée d'un angle ϕ par rapport à l'horizontale (voir figure 4.1.1). En variables adimensionnées, la longueur de la cavité est égale à 1. On suppose que le matériau suit une loi de Herschel-Bulkley et que le débit d'injection q est imposé sur la gauche du domaine. Le processus d'adimensionalisation donne alors $q = 1$ comme condition de bord en $x = 0$ (i.e. à gauche). En $x = 1$ (sur la droite), le mur est matérialisé par l'absence de flux $q = 0$. L'inconnue du problème à résoudre est la hauteur h du matériau au temps t et à distance x du point d'injection. On suppose qu'il n'y a pas de matériau en $t = 0$ (i.e. la cavité est initialement vide), ce qui nous donne la condition initiale $h(x, t = 0) = 0$ pour tout $x \in [0, 1]$. L'évolution de h est alors régie par l'EDP :

$$\forall x \in]0, 1[, \forall t > 0, \ \frac{\partial}{\partial t} h(x,t) + \frac{\partial}{\partial x} q(x,t) = 0, \tag{4.1}$$

où le flux $q(x, t)$ est défini par :

$$q(x,t) = sgn\left(S - \frac{\partial h(x,t)}{\partial x}\right) \left|S - \frac{\partial h(x,t)}{\partial x}\right|^{1/n} \frac{nY(x,t)^{1+1/n}}{(n+1)(2n+1)} \left((2n+1)h(x,t) - nY(x,t)\right), \tag{4.2}$$

où $sgn(.)$ est la fonction signe et n (typiquement $n \in [0.3, 1.2]$ en CPB) est l'exposant de la loi de Herschel-Bulkley suivie par le matériau. La variable $Y(x, t)$ encode la surface à laquelle le matériau transitionne entre fluide cisaillé et solide (i.e. la hauteur au-dessus de laquelle le matériau exhibe un pseudo-plug). Elle est donnée par l'expression :

$$Y(x,t) = \max\left(h(x,t) - \frac{B}{\left|S - \frac{\partial h}{\partial x}(x,t)\right|}, 0\right), \tag{4.3}$$

où B est le nombre de Bingham et S est le paramètre de pente dans notre configuration.

On observe que les paramètres indépendants de ce modèle sont alors B, S et n. Connaissant le triplet (B, S, n), en utilisant les conditions initiales sur h et les conditions de bord sur q données précédemment, il est possible de calculer l'évolution de h au sein de la cavité. On détaille maintenant ces paramètres à partir des variables dimensionnées du problème. Le paramètre de pente S est donné par :

$$S = \frac{\tan\phi}{\varepsilon}, \tag{4.4}$$

où :

$$\varepsilon = H_0/L, \tag{4.5}$$

donne le rapport d'aspect entre la hauteur typique du matériau (H_0 en mètres) et la longueur de la cavité (L en mètres, dans la direction x montrée en figure 4.1.1). Dans la théorie de lubrification (couche mince), il est supposé que $\varepsilon \ll 1$, c'est-à-dire que la hauteur de la géométrie est faible devant sa longueur.

On définit ensuite le nombre de Bingham par :

$$B = \frac{H_0 \tau_y}{\rho \nu U}, \tag{4.6}$$

où τ_y ($kg.m^{-1}.s^{-2}$ ou Pa) est le paramètre de seuil du matériau viscoplastique comme précédemment, ρ sa masse volumique ($kg.m^{-3}$) et U est la vitesse caractéristique ($m.s^{-1}$). Le paramètre $\hat{\nu}$ est ici la viscosité cinématique :

$$\hat{\nu} = \frac{K}{\rho}\left(\frac{U}{H_0}\right)^{n-1}, \tag{4.7}$$

avec K la consistance ($kg.m^{-1}.s^{n-2}$ ou $Pa.s^n$) associée au modèle de Herschel-Bulkley comme vu dans les chapitres précédents.

La hauteur typique $\hat{H}_0$ de la cavité se calcule à partir de :

$$\hat{H}_0 = \left(\frac{KL}{\rho g \cos\phi} \left(\frac{Q_0}{W} \right)^n \right)^{\frac{1}{2(1+n)}}, \tag{4.8}$$

où $g = 9.81 m^2.s^{-1}$ est la gravité ($g = \|\overrightarrow{g}\|$ dans la figure 4.1.1), W (en m) est l'épaisseur transverse de la cavité et Q_0 (en $m^3.s^{-1}$) est le débit d'injection (connu) du matériau. On définit enfin la vitesse caractéristique $\hat{U}$:

$$\hat{U} = \frac{Q_0}{W H_0}. \tag{4.9}$$

Ainsi, un praticien partant des variables dimensionnées peut calculer les quantités adimensionnées en suivant (4.8),(4.9) puis en remontant jusqu'à (4.4) et (4.6).

Étant donné l'objectif pratique de ce chapitre, il est ici utile de donner une idée des valeurs réelles de ces paramètres (B, S, n). Dans le contexte du CPB, un choix large (un peu plus étendu que les valeurs réelles par sécurité) serait $B \in [0.5, 250]$ et $S \in [\sim 0.1 \text{ ou } 1, 120]$. Concernant la puissance n, on trouve généralement $n \approx 1$. En règle générale, les mines réelles donnent lieu à des paramètres proches de $(B, S, n) \sim (150, 12, 1)$ (voir [244] pour plus de détails). On revient sur ce point dans la section 4.4.4 commentant les performances sur les données expérimentales.

4.1.3 Résolution numérique

La solution de (4.1)-(4.2) est approchée par un solveur numérique. On attire l'attention sur le fait que ce modèle a été extensivement testé dans différentes configurations par des membres majeurs de la communauté viscoplastique (Balmforth et al. [20], Li et al. [155], Hogg et Matson [130]). Ainsi bien qu'il semble difficile de trouver une preuve rigoureuse de l'existence et unicité de la solution de ce modèle au sein de la littérature, il semble être bien posé (au sens d'Hadamard) dans la configuration étudiée ici.

4.1.3.1 Discrétisation temporelle

L'évolution en temps est traitée par un schéma d'Euler explicite. Étant donnée une discrétisation spatiale Δx, on adapte le pas de temps Δt de manière dynamique à chaque itération. Une condition de stabilité a été déterminée heuristiquement de sorte à ce que le pas de temps soit déterminé en fonction du pas d'espace. Pour cela, on prend le minimum entre deux contraintes classiques de stabilité numérique sur Δt associées au problème non-linéaire d'advection-diffusion (4.1)-(4.2). Précisément, on peut développer à partir de (4.2) une expression de $\partial q/\partial x$ (voir annexe 4.A pour différentes expressions de $\partial q/\partial x$), puis l'intégrer à (4.1) afin de mettre en lumière un coefficient de transport $V(x)$ et un coefficient de diffusion $D(x)$. On obtient que (4.1) équivaut à :

$$\frac{\partial}{\partial t} h(x,t) + V(x) \frac{\partial}{\partial x} h(x,t) - D(x) \frac{\partial^2}{\partial x^2} h(x,t) = 0 \tag{4.10}$$

où l'on rappelle que $\delta = sgn(S - \partial h/\partial x)$, et on définit :

$$V(x) = \delta Y^{1/n} \left| S - \frac{\partial}{\partial x} h \right|^{1/n} h, \tag{4.11}$$

$$D(x) = \frac{-1}{(1+n)(1+2n)} Y^{1/n} \left| S - \frac{\partial}{\partial x} h \right|^{1/n-1} \left(2nh \frac{B}{\left| S - \frac{\partial}{\partial x} h \right|} + (1+n)h^2 + 2n^2 \frac{B^2}{\left(S - \frac{\partial}{\partial x} h \right)^2} \right). \tag{4.12}$$

On introduit pour le terme d'advection de (4.10) une condition CFL, puis une seconde condition pour le terme de diffusion (à cause de son traitement explicite, voir LeVeque [147]). En prenant le minimum de ces deux conditions, on obtient la formule suivante pour le pas de temps :

$$\Delta t = \min\left(\frac{\Delta x}{2\max_x(V(x))}, C_d\frac{(\Delta x)^2}{\max_x(D(x))}\right), \tag{4.13}$$

où (le classique) $C_d = 0.5$ assure la stabilité pour la plupart des simulations. Prendre une valeur plus basse, par exemple $C_d = 0.05$, pour stabiliser des simulations où le temps physique doit être long (phase de croissance sur le mur de droite), a été testé. Cependant, pour le travail à suivre, les solutions ne sont calculées que jusqu'à ce que l'écoulement atteigne le bord droit du domaine, aussi $C_d = 0.5$ permet d'obtenir des simulations stables pour tous les paramètres (B, S, n) testés.

4.1.3.2 Discrétisation spatiale

La discrétisation spatiale est quant à elle traitée via un schéma différences finies centrées pour $\partial h/\partial x$ et un schéma upwind pour $\partial q/\partial x$, i.e. :

$$\frac{\partial h}{\partial x}(t, x_i) = \frac{h(t, x_{i+1}) - h(t, x_{i-1})}{2\Delta x}, \tag{4.14}$$

$$\frac{\partial q}{\partial x}(t, x_i) = \frac{q(t, x_i) - q(t, x_{i-1})}{\Delta x}, \tag{4.15}$$

où Δx est la taille d'une maille du maillage (pris uniforme) et t un temps donné. En utilisant les conditions de bord et (4.1)-(4.2), q peut être calculé pour tous les points du maillage, ce qui permet de calculer h pour tous les points, à l'exception du premier point $x = 0$ (qui nécessiterait l'évaluation de $q(-\Delta x)$). Ce point est traité en utilisant la condition de bord $q(0) = 1$ et un schéma downwind pour $\partial h/\partial x$. Pour simplifier les notations, écrivons $h_1 = h(t, \Delta x)$ (qui est connu à ce stade) et $h_0 = h(t, 0)$ (que l'on cherche). L'équation (4.2) donne alors :

$$1 = \delta\frac{n}{(n+1)(2n+1)}Y_d(h_0)^{1+1/n} \times \left|S - \frac{h_1 - h_0}{\Delta x}\right|^{1/n}((2n+1)h_0 - nY_d(h_0)) =: q_0(h_0), \tag{4.16}$$

où :

$$\delta = sgn\left(S - \frac{h_1 - h_0}{\Delta x}\right), \tag{4.17}$$

et :

$$Y_d(h_0) = \max\left(h_0 - \frac{B}{\left|S - \frac{h_1 - h_0}{\Delta x}\right|}, 0\right). \tag{4.18}$$

On obtient alors une équation sur h_0 hautement non-linéaire, dont la solution ne peut être calculée analytiquement.

On observe que $Y_d \leq h_0$ par (4.18), donc $(2n+1)h_0 \leq nY_d(h_0)$ et ainsi le signe de q_0 est donné par celui de δ. Comme on veut $q_0 = 1 > 0$, l'équation (4.17) nous donne alors que $h_0 \geq h_1 - S\Delta x$. Étant donnée cette minoration, on peut enlever la valeur absolue dans la formule de Y_d et on peut alors vérifier que Y_d est une fonction strictement croissante de h_0 dès lors que Y_d devient strictement positive. Notons :

$$h_{min} = \inf\{h \geq 0 / h \geq h_1 - S\Delta x \text{ et } Y_d(h) > 0\}.$$

Ainsi, $\forall h_0 > h_{min}$, on trouve que $Y_d(h_0) > 0$ et $q_0(h_0) > 0$. Sur $]h_{min}, +\infty[$, q_0 est alors une fonction continue strictement croissante parcourant $]0, +\infty[$. On obtient alors l'existence et l'unicité de la solution de (4.16). Calculons maintenant h_{min} :

$$Y_d > 0 \Leftrightarrow h_0 \left(S - \frac{h_1 - h_0}{\Delta x} \right) > B,$$

$$Y_d > 0 \Leftrightarrow h_0^2 \frac{1}{d} + x \left(S - \frac{h}{\Delta x} \right) - B > 0.$$

Les racines de cette parabole sont données par :

$$h_{min} = \frac{h_1 - S\Delta x}{2} \pm \sqrt{\left(\frac{h_1 - S\Delta x}{2} \right)^2 + \Delta x B}.$$

La condition $h_{min} \geq 0$ permet alors de fixer le signe :

$$h_{min} = \frac{h_1 - S\Delta x}{2} + \sqrt{\left(\frac{h_1 - S\Delta x}{2} \right)^2 + \Delta x B}. \tag{4.19}$$

Grâce à cette borne inférieure on peut résoudre numériquement (4.16) en utilisant les fonctions built-in des bibliothèques standards (sans que celles-ci se retrouvent piégées dans la zone constante $q_0(h) = 0$). En l'occurrence, le solveur est codé en Python, on utilise la fonction `brenth` de Scipy (voir [59]) pour effectuer la résolution (en Matlab on aurait pu utiliser la fonction `fzero` par exemple).

Remarque 66 *Dans le cas particulier $n = 1$, on peut développer l'expression de $q_0(h)$ dans le cas où $h \geq h_{min}$, pour obtenir une expression polynomiale d'ordre 7. On peut alors calculer numériquement la racine souhaitée (en imposant la condition $h_0 \geq h_{min}$) en utilisant les bibliothèques standards. Cependant, cela prend en pratique plus de temps que de résoudre le problème d'optimisation défini par (4.16).*

Remarque 67 *D'aucun pourrait s'attendre à ce que les calculs de la zone de front (où h devient nul) posent problème, cependant grâce au seuil dans la formule (4.3), l'EDP (4.1)-(4.2) ne pose pas de difficulté particulière dans cette zone, le seul point problématique étant le calcul de $h(t, 0)$ développé ci-dessus.*

4.1.3.3 Performances du solveur

Il reste à déterminer le pas d'espace Δx. On étudie pour cela la convergence du solveur selon Δx. On rappelle que Δt est déterminé à partir de Δx selon la formule (4.13), aussi se pose la question de savoir selon quelle modalité étudier la convergence en espace. Il est d'usage de fixer un pas de temps Δt proche de 0 puis de faire varier Δx, en utilisant ce pas de temps de manière systématique (en ignorant donc (4.13)). Bien que d'un intérêt théorique certain, cela ne permet pas de s'assurer de la convergence de l'algorithme exposé ci-dessus qui fait usage des conditions de stabilité pour maximiser le pas de temps. Ce travail étant orienté vers un usage pratique, on choisit de présenter extensivement une seconde méthode : on va faire évoluer Δx et Δt (par (4.13)) en même temps. Toutefois, la première manière *classique* de procéder a aussi été testée et fournit des résultats similaires.

En l'absence de solution analytique du modèle, on calcule une solution de référence avec un Δx très petit (on a utilisé $n_x = 9601$ points d'espace). On compare ensuite cette solution avec les solutions fournies par des pas d'espace plus importants ($n_x = 76, 151, 301, \ldots, 4801$) afin

d'évaluer s'il y a une convergence numérique de la norme de la différence entre les deux solutions à mesure que Δx est réduit. La comparaison est effectuée à un temps bien particulier : le temps au bout duquel le fluide atteint le mur de droite de la cavité, on appelle désormais cet instant le *wall-touch*. Un exemple de cette étude est fourni en figure 4.1.2 pour $B = 100$, $S = 120$, $n = 0.8$. On observe que le schéma converge avec une loi puissance de coefficient a minima 0.6. Cet exposant n'est pas très élevé, ce qui est dû à la raideur du problème non-linéaire. Le développement d'un schéma d'ordre plus élevé n'est pas étudié ici. Des études similaires ont été produites pour d'autres valeurs des paramètres, à savoir ($B = 30$, $S = 15$, $n = 0.8$) et ($B = 70$, $S = 0.3$, $n = 0.8$), avec des convergences observées d'ordre entre 0.6 et 1. On prend ceci comme validation du solveur.

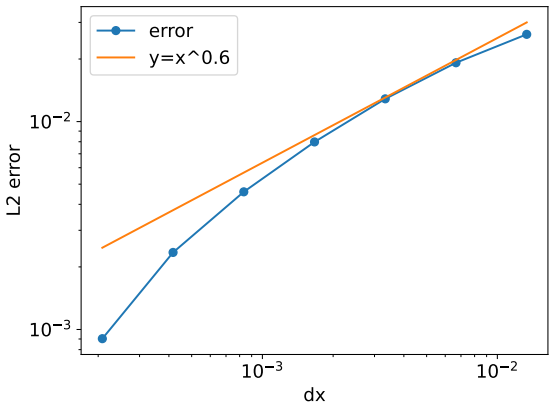


FIGURE 4.1.2 – Étude de la convergence (Δx est raffiné, et donc aussi Δt par (4.13)) du solveur numérique de l'EDP pour $B = 100$, $S = 120$, $n = 0.8$. Chaque solution a été calculée à son instant respectif de wall-touch. La solution de référence (h^{ref}) a été calculée avec $n_x = 9601$ points. L'erreur L^2 ($\|h^{ref}(x) - h^{\Delta x, \Delta t}(x)\|_2$) décroît au moins comme $y = x^{0.6}$ (en échelle log).

Certains paramètres (B, S) engendrent des résolutions très rapides, mais d'autres peuvent prendre jusqu'à deux heures pour une grille de seulement $n_x = 301$ points, voire des semaines pour des grilles plus raffinées. Cela correspond aux hautes valeurs de B et basses valeurs de S, i.e. une grande valeur du seuil τ_y et une pente faible. Une illustration de cette répartition des temps de calcul est donnée dans la figure 4.1.3 pour le cas $n = 1$. Pour de tels paramètres, le wall-touch se produit à un temps plus grand (en fait le tracé du temps de wall-touch en fonction de B et S est très similaire à la figure 4.1.3), aussi plus de pas de temps sont nécessaires pour terminer la résolution. De cette étude a été déduit que le choix de $n_x = 301$ points de maillage en espace permet à la fois une bonne précision sur le profil de la solution au wall-touch, ainsi que des temps de calcul raisonnables dans la perspective de construction d'un métamodèle, comme évoqué en début de chapitre.

Cela étant dit, certains paramètres peuvent donner lieu à des temps de calculs de l'ordre de l'heure. En supposant que l'on cherche à estimer les paramètres (B, S, n) à partir d'un profil $h(t, x)$, il est nécessaire de passer par une méthode de problème inverse qui peut requérir de nombreuses évaluations du problème direct $(B, S, n) \mapsto h(t, x)$. Utiliser le solveur direct dans un tel cadre pourrait entraîner des temps de calculs trop importants pour un usage pratique réaliste. C'est pour cette raison que l'on propose de construire un métamodèle, i.e. une approximation de notre solveur EDP qui soit plus rapide à évaluer.

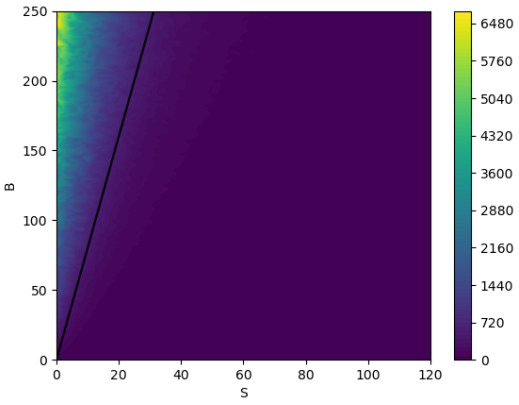


FIGURE 4.1.3 – Temps de calcul des solutions de (4.1)-(4.2) en utilisant le solveur numérique (en secondes) pour différents couples (B, S) et $n = 1$. La ligne noir d'équation $B = 8S$ donne une délimitation approximative de la zone de paramètres donnant lieu à des temps calculs longs.

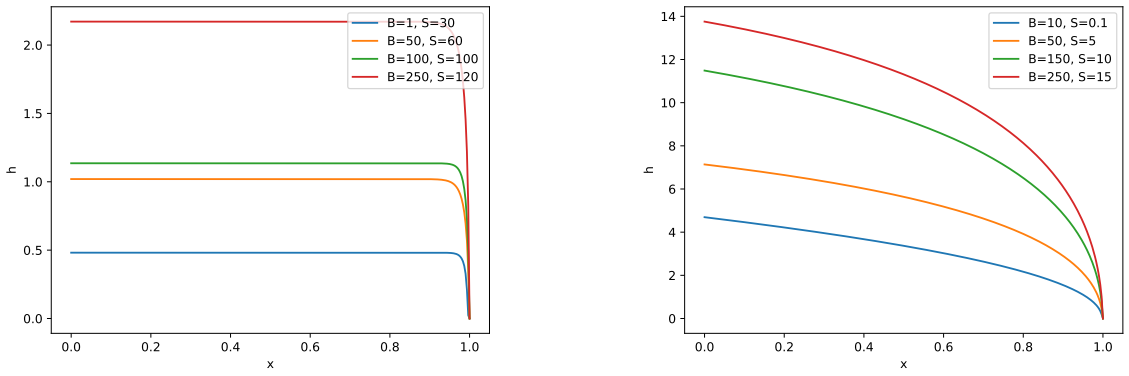
4.1.3.4 Problème direct final

On résume ici le problème direct considéré. Étant donné un triplet (B, S, n), le modèle direct nous donne un vecteur $h \in \mathbb{R}^{301}$ contenant le résultat du solveur numérique au wall-touch en utilisant $n_x = 301$ points de discrétisation spatiale. Cela correspond à l'approximation du profil de hauteur du fluide considéré au moment où il atteint le bord droit de la cavité. Le métamodèle et la procédure d'inversion seront effectués à n fixé, car on suppose que les physiciens sont en mesure de fournir une valeur précise de n via des méthodes de rhéologie, mais la procédure peut être réalisée pour une valeur quelconque de n (de fait, elle est réalisée ici pour $n = 1$ et $n = 0.8$). On remarque que le modèle (4.1)-(4.2) varie de manière régulière avec n pour les valeurs qui apparaissent dans les applications considérées ($n \sim 1$). Ainsi, l'objectif final est d'estimer B et S à partir d'un profil de hauteur dans la cavité. On suppose dorénavant que $B \in [0.5, 250]$ et $S \in [0.05, 120]$.

4.2 Identifiabilité du modèle

Avant de chercher à résoudre le problème d'estimation de paramètres, il est nécessaire de s'interroger sur l'identifiabilité du modèle, i.e. étant donné un profil de hauteur $\tilde{h}$ observé, existe-t-il un unique couple (B, S) tel que $h(B, S) = \tilde{h}$? La figure 4.2.1 donne des exemples de profils de hauteur pour différents couples (B, S). On dégage deux types de profils distincts. On a d'une part des profils de type *train qui avance* (voir figure 4.2.1a), pour lesquels la hauteur du fluide est constante sur la majorité de la cavité, puis chute brutalement à proximité du front. On note qu'une observation de l'évolution en temps des solutions de l'EDP associée à de tels couples montre les mêmes caractéristiques tout au long de l'écoulement (la hauteur est rapidement fixée puis on observe seulement le front qui avance, d'où la terminologie *train*). Les solutions associées à ces couples diffèrent essentiellement les unes des autres par la hauteur de la partie traine. D'autre part, on observe des profils de type *croissance baudruche*, similaires à une branche de parabole (voir figure 4.2.1b). De même que pour les profils train, les profils baudruche conservent globalement leur forme au cours du temps, sauf que ces derniers voient leur hauteur en $x = 0$ augmenter en même temps que le front avance.

Il semble que les profils trains correspondent à des cas où S est plutôt élevé, et B plutôt faible. Intuitivement, la pente est importante et donc le fluide "fuit vers le bas" avec une hauteur



(a) Profils *train* (b) Profils *baudruche*

FIGURE 4.2.1 – Quelques exemples de solutions du problème direct pour différents couples (B, S) $(n = 1)$. On note deux types de profils, des profils plutôt *train* avec une forte pente près du front, puis des profils plutôt *croissance baudruche*.

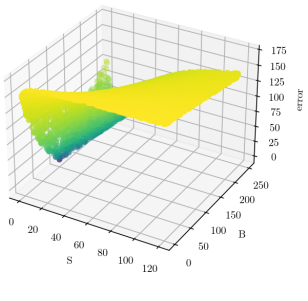
presque constante (effet d'enduction, ou *coating* en anglais). Les profils baudruche proviennent de couples où B est plus élevé et S plus faible, i.e. les effets viscoplastiques sont plus prépondérants et bloquent l'avancée du front. Il est à noter que ce sont les profils baudruche qui sont les plus longs à calculer (leur temps de wall-touch est plus important, étant donné que plus de fluide doit pénétrer dans la cavité pour faire avancer le front).

Intuitivement, on voit donc deux caractéristiques apparaître dans les profils : la caractère plutôt plat ou arrondi du profil, puis la hauteur en $x = 0$. Étant donné que l'on a deux paramètres d'entrée, on s'attend donc à ce que le problème soit bien posé et que l'on puisse bien résoudre le problème inverse. L'étude qui suit (effectuée pour $n = 1$) tend à confirmer cette intuition.

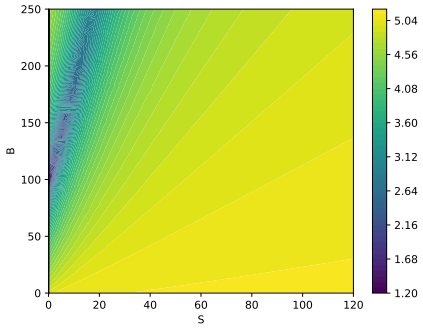
4.2.1 Étude graphique

Un data set de plus de 6000 couples (B, S) choisis aléatoirement dans $[0.5, 250] \times [0.05, 120]$ a été précalculé. Étant donné un couple (B_0, S_0), on calcule la norme L^2 de la différence entre $h(B_0, S_0)$ et $h(B, S)$ pour tous les autres couples (B, S) du data set. On montre un échantillon représentatif des résultats en figure 4.2.2.

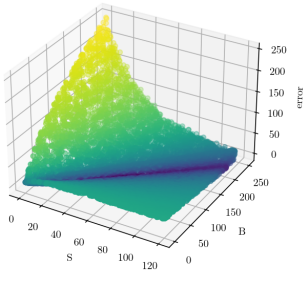
En visualisant les résultats en 3D (l'axe z représente la norme de l'erreur), on observe que les cartes d'erreur ont une forme proche d'un "V". Cette observation est moins claire pour les S petits (voir figure 4.2.2a), mais étant donné que l'on est au bord du domaine (B, S), il est raisonnable de supposer que la seconde branche du "V" n'est simplement pas calculée dans le data set. L'erreur est globalement élevée lorsque l'on considère un couple hors de la pointe du "V", cependant cette localisation crée une sorte de vallée, ou une ligne en 2D, où l'erreur est très faible, avec très peu de variation vers le couple (B_0, S_0). Sur les figures 4.2.2b, 4.2.2d, 4.2.2f et 4.2.2h, on a ajouté en noir la ligne d'équation $y = (B_0/S_0)x$. Il est intéressant de remarquer que pour la majorité des profils, cette ligne coïncide avec cette vallée de faible erreur (cette équation semble toutefois ne pas fonctionner pour les S très petits comme le montre la figure 4.2.2b). Cela suggère une sorte d'auto-similarité dans les solutions, dont on peut attendre qu'elle complique la résolution du problème inverse. Toutefois, la figure 4.2.3 montre un zoom sur cette vallée, et il semble que cette ligne pointe bel et bien vers la solution, bien qu'à vitesse réduite. La forme de la fonctionnelle semble donc propice à l'usage des méthodes d'optimisation usuelles, en particulier elle ne contient pas de minimum local et pas de zones traines.



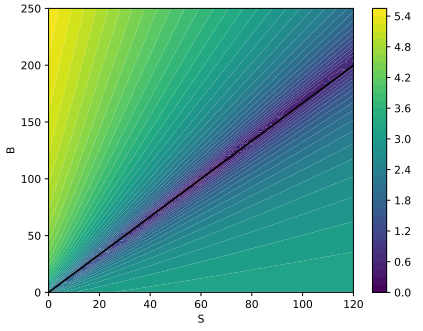
(a) $B_0 = 100$, $S_0 = 0.1$



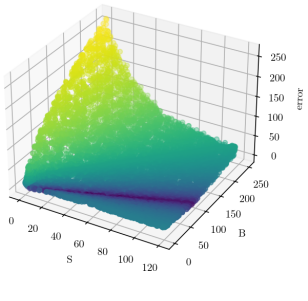
(b) $B_0 = 100$, $S_0 = 0.1$



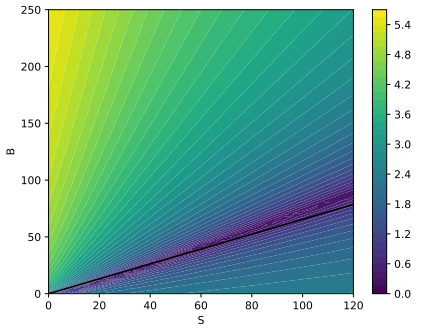
(c) $B_0 = 100$, $S_0 = 60$



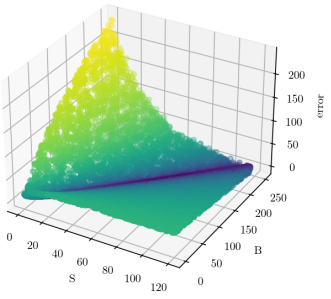
(d) $B_0 = 100$, $S_0 = 60$



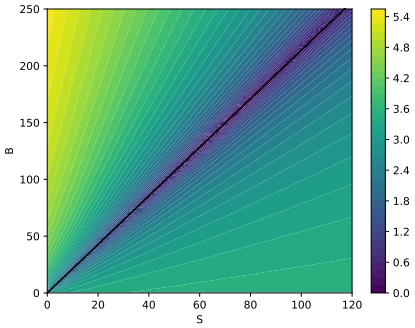
(e) $B_0 = 31.87$, $S_0 = 48.67$



(f) $B_0 = 31.87$, $S_0 = 48.67$

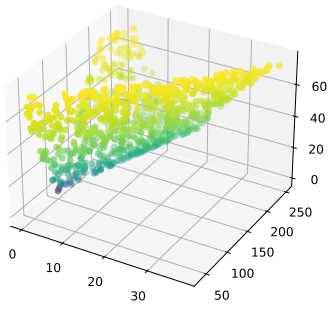


(g) $B_0 = 152.06$, $S_0 = 71.14$

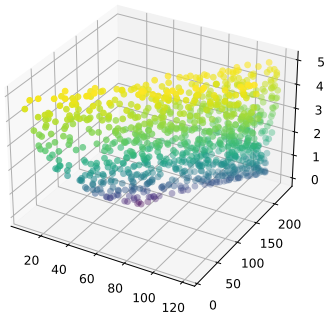


(h) $B_0 = 152.06$, $S_0 = 71.14$

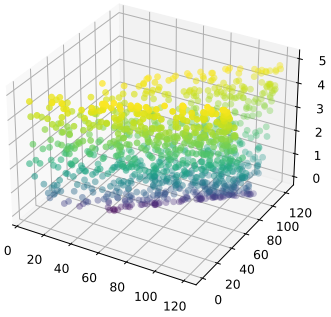
FIGURE 4.2.2 – Cartes d'erreurs pour différentes valeurs de (B_0, S_0) et $n = 1$. Pour chaque couple, une carte 3D et une carte 2D sont montrées. La carte 3D met en valeur la forme de "V" de l'erreur. Sur la carte 2D, on a ajouté la ligne d'équation $y = (B_0/S_0)x$ (en noir). Elle ne coïncide pas toujours avec le minimum de l'erreur.



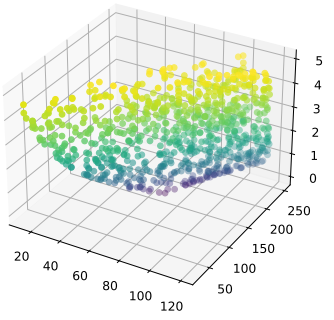
(a) $B_0 = 100$, $S_0 = 0.1$



(b) $B_0 = 100$, $S_0 = 60$



(c) $B_0 = 31.87$, $S_0 = 48.67$



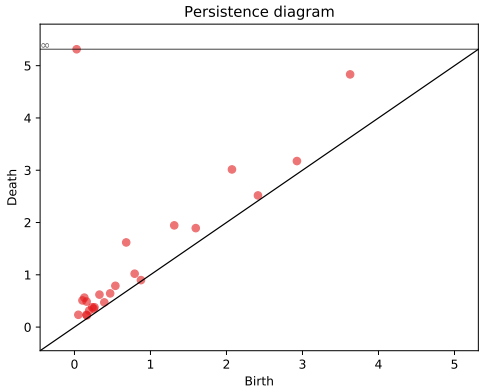
(d) $B_0 = 152.06$, $S_0 = 71.14$

FIGURE 4.2.3 – Zoom sur les cartes d'erreur 3D pour les différents couples de la figure 4.2.2.

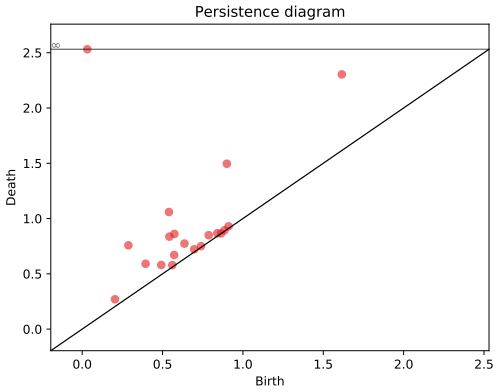
4.2.2 Analyse topologique de données

On essaie ici, de manière heuristique et avec une étude préliminaire qui a permis de s'initier au sujet, de renforcer les conclusions de la section précédente en utilisant un outil issu de l'analyse topologique de données (TDA), à savoir les diagrammes de persistance.

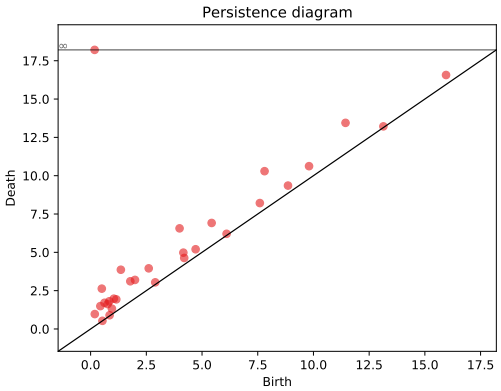
Des détails de TDA sont donnés en annexe (voir section 4.B). Pour résumer en quelques mots, l'idée est d'observer, étant donnée une fonction f (ici l'erreur entre $h(B_0, S_0)$ et $h(B, S)$), l'évolution du nombre de composantes connexes des sous-niveaux de f, i.e. des ensembles $\mathcal{L}_f(y) := \{(B, S)/f(B, S) \leq y\}$ à mesure que l'on fait varier y de $\min(f)$ à $\max(f)$ sur l'ensemble échantillonné. Lorsqu'une nouvelle composante apparaît, on dit qu'elle *naît*, et lorsqu'elle disparaît on dit qu'elle *meurt*. On traque les naissances et morts de chaque composante. Cela permet de détecter la présence de minima locaux et de quantitativement évaluer la profondeur des puits les entourant. Ainsi, sur les diagrammes de la figure 4.2.4, chaque point rouge correspond à une composante connexe, sa durée de vie étant donnée par sa hauteur par rapport à l'axe $y = x$ (voir section 4.B pour une description précise).



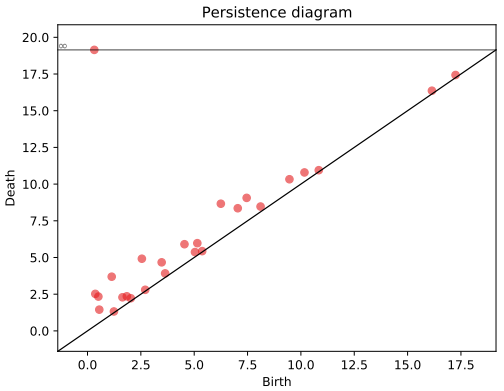
(a) $B_0 = 110.1$, $S_0 = 100.74$



(b) $B_0 = 16.03$, $S_0 = 22.37$



(c) $B_0 = 201.7$, $S_0 = 49.08$

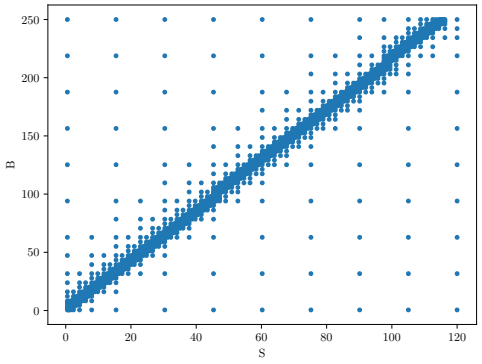


(d) $B_0 = 228.08$, $S_0 = 40.11$

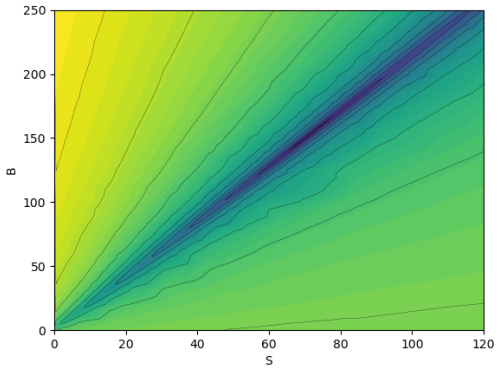
FIGURE 4.2.4 – Différents diagrammes de persistance de la fonction d'erreur pour différents couples (B_0, S_0) $(n = 1)$.

Dans un diagramme idéal, pour une fonction ne possédant qu'un seul minimum local, on observerait une seule composante, i.e. un seul point rouge. Malheureusement les diagrammes de la figure 4.2.4 semblent très bruités, ce qui peut s'expliquer par l'échantillonnage trop grossier en (B, S). Il est commun en TDA d'appliquer une procédure de seuillage (sur les durées de vie) aux diagrammes afin d'éliminer le bruit. Cependant, les graphes montrent des durées de vie d'ordres

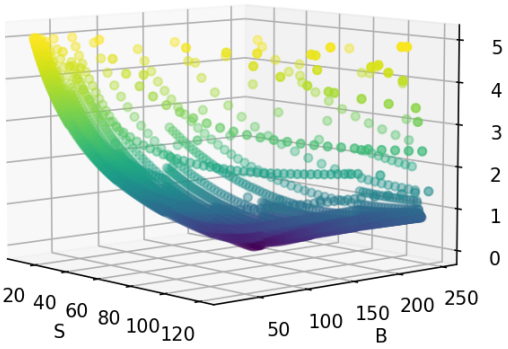
de grandeur variés aussi, serait-il arbitraire de seuiller dans notre cas. Il faudrait donc pouvoir échantillonner de manière plus raffinée, mais cela prendrait un temps trop important. À la place, on propose de prendre quelques couples et de raffiner uniquement selon la vallée d'auto-similarité mise en lumière dans la section 4.2.1. On commence par échantillonner une grille très grossière 9×9 sur le domaine (B, S), puis, étant donné le couple (B_0, S_0), on raffine le long de sa vallée. Plus précisément, on itère la procédure de raffinement suivante : étant donné un quadrillage du domaine (B, S), pour chaque cellule (définie comme un carré dont les quatre sommets font partie du quadrillage), si la vallée selon laquelle on souhaite raffiner passe à l'intérieur de ladite cellule, on ajoute le point central de sorte à diviser la cellule en quatre nouvelles. Une fois toutes les cellules d'un quadrillage parcourues, on réitère sur le nouveau quadrillage. On choisit ici de s'arrêter lorsque l'on atteint 3000 points dans le quadrillage.



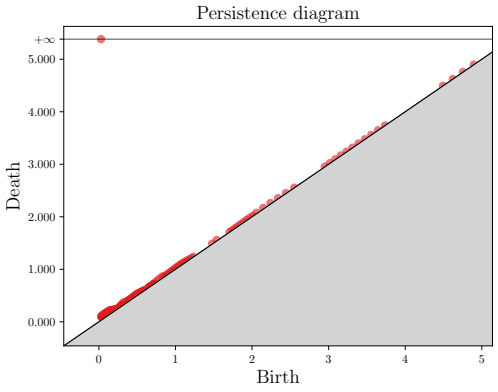
(a) Cartes des couples utilisés



(b) Carte d'erreur 2D



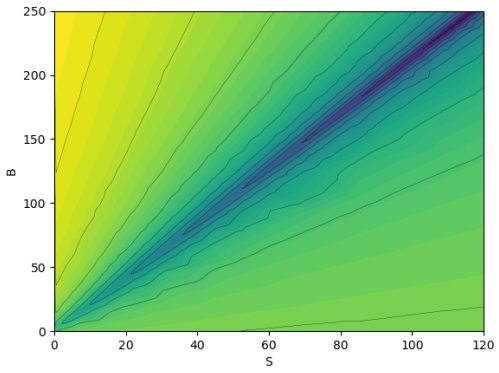
(c) Carte d'erreur 3D, zoom sur les erreurs basses



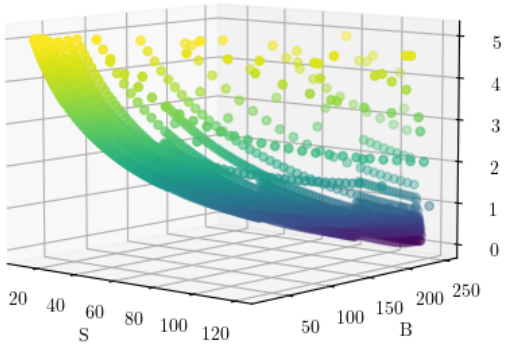
(d) Diagramme de persistance

FIGURE 4.2.5 – Résultats de l'étude selon la vallée d'auto-similarité pour le couple $B = 152.06$, $S = 71.14$. La figure (a) présente une carte des échantillons utilisés. La vallée d'auto-similarité a été déterminée en accord avec la figure 4.2.2g. La figure (b) donne la carte 2D de l'erreur. La figure (c) donne la carte d'erreur 3D correspondante. La figure (d) présente le diagramme de persistance associé.

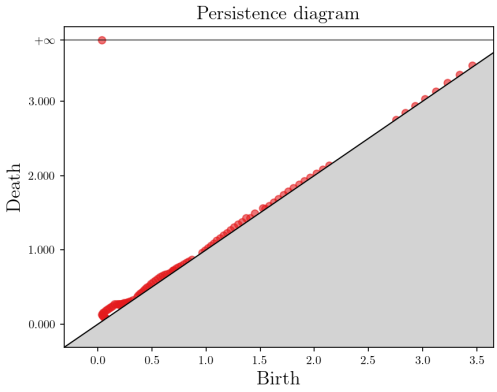
On donne les résultats pour deux couples différents en figures 4.2.5 et 4.2.6. On constate que raffiner selon la vallée de stabilité efface presque totalement le bruit dans les diagrammes de persistance. On retrouve les observations de la section précédente, à savoir qu'il n'y a pas de minima locaux. De plus, on a choisi le second couple de sorte à ce que sa vallée d'auto-similarité soit proche de celle du premier, aussi n'a-t-on pas recalculé de data set. On observe que cela



(a) Carte d'erreur 2D



(b) Carte d'erreur 3D, zoom sur les erreurs basses



(c) Diagramme de persistance

FIGURE 4.2.6 – Résultats de l'étude selon la vallée d'auto-similarité pour le couple $B = 242.46$, $S = 113.43$. Le data set utilisé pour le diagramme est le même que celui de la figure 4.2.5 car les deux couples ont une vallée d'auto-similarité de localisation proche. La figure (a) donne la carte 2D de l'erreur. La figure (b) donne la carte d'erreur 3D correspondante. La figure (c) présente le diagramme de persistance associé.

suffit quand même à effacer le bruit du diagramme.

Ces résultats ne sont pas totalement satisfaisants car comme pour la section 4.2.1, ils reposent sur des études faites seulement sur quelques couples du data set, la procédure ci-dessus étant trop coûteuse pour être reproduite sur tous les couples. Cependant, ils tendent à montrer que le modèle est identifiable. On note par ailleurs qu'une étude théorique sur une potentielle forme d'auto-similarité des solutions du problème est fortement suggérée pour des travaux futurs.

4.3 Métamodèle

Une procédure standard pour construire un métamodèle est le *développement en polynômes de chaos* (en anglais *Polynomial Chaos Expansion*, PCE) qui est une approximation polynomiale. Cette méthode est souvent utilisée pour la quantification d'incertitude (voir Le Maître et Knio [145]) ou encore l'analyse de sensibilité (voir Sudret [232]). Afin de réduire la dimension du problème (qui est actuellement $n_x = 301$), on combine cela avec une *analyse en composantes*

principales (PCA).

4.3.1 Développement en polynômes de chaos

On commence par décrire la procédure de PCE pour un vecteur aléatoire borné $X \in \mathbb{R}^d$ (pour nous $X = (B, S) \in \mathbb{R}^2$) et un modèle de calcul $\mathcal{M} : \mathbb{R}^d \to \mathbb{R}^m$.

On s'intéresse alors au vecteur aléatoire $Y = \mathcal{M}(X)$ (dans notre cas, $m = 301$ et $Y \in \mathbb{R}^{301}$ est le profil de hauteur au wall-touch). On débute cette description par le cas $m = 1$ pour simplifier la présentation, puis on décrit le cas général $m \geq 1$. L'objectif est d'approcher Y par une formule du type :

$$Y = \sum_{\alpha \in \Lambda} \psi_\alpha(X), \tag{4.20}$$

où $(\psi_\alpha)_{\alpha \in \Lambda}$ sont des polynômes. Y est supposé avoir une variance finie, ce qui est raisonnable pour nous dans la mesure où (B, S) vit dans un domaine borné et que l'on suppose que la fonction $(B, S) \mapsto h$ est régulière. On suppose aussi que les coordonnées de X sont indépendantes.

4.3.1.1 Construction générale

On suppose $m = 1$ pour le moment. On définit f_{X_i} la loi marginale de la i^e composante de X. On l'utilise pour définir un produit scalaire sur les fonctions u, v définies sur le support S_i de X_i (on rappelle que X vit sur un domaine borné) :

$$\langle u, v \rangle_i = \int_{S_i} u(z)v(z)f_{X_i}(z)dz. \tag{4.21}$$

À partir de ce produit scalaire, on peut définir une notion d'orthogonalité. Alors la procédure standard de Gram-Schmidt nous permet de construire une famille de polynômes orthogonaux $(P_k^i)_{k \in \mathbb{N}}$, qui dépend de f_{X_i}, et qui est en toute généralité non triviale. Dans certains cas particuliers, on peut retrouver une famille de polynômes classiques. Dans notre cas (distribution uniforme), on obtient les polynômes de Legendre. Pour plus d'exemples donnant lieu à des polynômes connus, voir Sudret [232], Xiu et Karniadakis [254].

Remarque 68 *À l'origine, la méthode a été introduite avec des polynômes de Hermite. En toute rigueur, il faudrait parler de PCE généralisé pour les autres cas. On omet ici cette distinction.*

Pour le moment, on a construit des familles de polynômes à une variable (une famille pour chaque composante de X). Pour construire des polynômes du vecteur X complet, on utilise un produit de polynômes univariés (ce qui est cohérent avec l'hypothèse d'indépendance entre les composantes de X). Formellement, on introduit des multi-indices $\alpha = (\alpha_1, .., \alpha_d) \in \mathbb{N}^d$ et on définit :

$$\forall x \in \mathbb{R}^d,\ \psi_\alpha(x) = \prod_{i=1}^{d} P_{\alpha_i}^i(x_i). \tag{4.22}$$

Dans ce cadre, il est possible de prouver (voir par exemple Le Maître et Knio [145]) que l'on peut écrire :

$$Y = \sum_{\alpha \in \mathbb{N}^d} c_\alpha \psi_\alpha(X), \tag{4.23}$$

où les c_α sont des réels à déterminer.

4.3.1.2 Considérations pratiques d'implémentation

Entrées standardisées. Les polynômes de Legendre (tout comme les autres familles de polynômes connues qui pourraient intervenir dans cette procédure) correspondent à une loi uniforme avec des paramètres précis ([254]). Lorsque l'entrée X ne correspond pas à une de ces familles connues, il existe des méthodes pour trouver la famille correspondante (voir [232] et les références qu'il contient), mais la procédure standard (lorsque c'est possible) est de transformer la variable X en une nouvelle variable $\hat{X}$ qui correspond à un cas connu. Dans le cas des distributions uniformes, la loi standard est la loi uniforme sur $[-1, 1]$. Or, nous avons ici $B \in [0.5, 250]$ et $S \in [0.05, 120]$. On procède donc à un simple changement affine pour récupérer l'intervalle standard :

$$\tilde{B} = \frac{B - 125.25}{125.25}, \tag{4.24}$$

$$\tilde{S} = \frac{S - 60.025}{60.025}. \tag{4.25}$$

Ordre de troncature. Il va de soi qu'on ne peut pas en pratique déterminer une infinité de coefficients c_α, il est donc nécessaire de choisir quels polynômes sélectionner. Pour cela, définissons le degré d'un polynôme multivarié par la somme des degrés des polynômes univariés qui le composent, i.e. :

$$deg\left(\prod_{i=1}^{d} P_i(X_i)\right) = \sum_{i=1}^{d} deg(P_i). \tag{4.26}$$

De manière générale, les polynômes de degrés élevés sont associés à des interactions d'ordres élevés entre les entrées, qui sont généralement limitées ([145],[232]). Pour cette raison, le choix des polynômes est généralement fait par une troncature sur le degré des polynômes, i.e. on prend tous les polynômes d'un degré inférieur ou égal à un certain β, généralement pris entre 3 et 5.

Ainsi dès lors qu'un ordre de troncature β a été déterminé, on peut réécrire le problème (4.23) comme :

$$Y \simeq \sum_{q=1}^{l} c_q \psi_q(X), \tag{4.27}$$

où les polynômes ont été réordonnés pour clarifier les notations. Dans ce cas, l (le nombre de polynômes que l'on garde dans la représentation) est fini, il dépend de l'ordre de troncature β, mais l et β sont différents (l est significativement plus grand que β).

Détermination des coefficients. Il existe de nombreuses techniques pour déterminer les coefficients de (4.27). Une première distinction à effectuer est le caractère *intrusif* ou non d'une méthode. Les méthodes intrusives telles que la projection de Galerkin sont basées sur la résolution de problèmes modifiés (i.e. non basées sur des évaluations de $\mathcal{M}$) ce qui nécessite de créer des solveurs spécifiques (voir [145]). Plus récemment, des méthodes non-intrusives basées uniquement sur l'évaluation de $\mathcal{M}$ et des outils statistiques se sont multipliées. Parmi elles, on retient pour ce travail la méthode consistant à traiter le problème comme un problème de minimisation (voir Migliorati et Nobile (2015) [164], Hadigol et Doostan (2018) [123], Sudret (2015) [232]). Étant donnés un échantillonnage $(X^{(1)}, ..., X^{(r)})$ et les évaluations correspondantes $(Y^{(1)}, ..., Y^{(r)})$, on réduit le problème (4.27) au problème de moindres-carrés suivant :

$$\min_{c \in \mathbb{R}^l} \sum_{j=1}^{r} \left(Y^{(j)} - \sum_{q=1}^{l} c_q \psi_q(X^{(j)}) \right)^2. \tag{4.28}$$

Pour plus de détails, voir [164], [123], [232].

Sortie multivariée. (Cas $m \geq 1$). Il reste à traiter le fait que dans notre cas, $\mathcal{M}$ est à valeurs vectorielles et non scalaires. Il semble que dans la plupart des articles publiés dans la littérature, les composantes de la sortie sont toutes traitées séparément. Cela signifie qu'il faut décomposer $\mathcal{M}(X) = (\mathcal{M}(X)_1, ..., \mathcal{M}(X)_m)$ et appliquer une procédure de PCE indépendamment pour chaque $\mathcal{M}(X)_i$, ce qui résulte en m résolutions différentes du problème (4.28) (voir Sudret (2015) [232], Garcia-Cabrejo et Valocchi (2014) [108], Sun et al. (2020) [233]).

4.3.2 Analyse en composantes principales

On donne ici une brève description de l'analyse en composantes principales (PCA). Pour plus de détails, voir le livre de Jolliffe [137]. Dans cette section, on étudie le vecteur aléatoire $Y \in \mathbb{R}^m$ directement, la fonction $\mathcal{M}$ est complètement oubliée. On commence par définir $\varphi^0 \in \mathbb{R}^m$ comme :

$$\varphi^0 = \underset{\varphi \in \mathbb{R}^m}{\arg\max}\ Var(\langle \varphi, Y \rangle). \tag{4.29}$$

On définit ensuite :

$$\alpha^0 = \langle \varphi^0, Y \rangle. \tag{4.30}$$

Alors α^0 est appelée la première *composante principale* (PC) de Y et φ^0 est sa direction. Pour $1 \leq k \leq m-1$, on définit récursivement la k^e PC de manière similaire :

$$\varphi^k = \underset{\alpha^0, ..., \alpha^{k-1}, \langle \varphi, Y \rangle \text{ non-corrélés}}{\arg\max} Var(\langle \varphi, Y \rangle), \tag{4.31}$$

et :

$$\alpha^k = \langle \varphi^k, Y \rangle. \tag{4.32}$$

Pour alléger les notations, on introduit la notation matricielle :

$$\alpha = \varphi Y, \tag{4.33}$$

où :

$$\alpha = \begin{pmatrix} \alpha^0 \\ \vdots \\ \alpha^{m-1} \end{pmatrix}, \quad \varphi = \begin{pmatrix} \varphi_0^0 & \cdots & \varphi_{m-1}^0 \\ \vdots & \ddots & \vdots \\ \varphi_0^{m-1} & \cdots & \varphi_{m-1}^{m-1} \end{pmatrix}. \tag{4.34}$$

On peut montrer que la matrice φ obtenue est inversible. On obtient alors immédiatement que $Y = \varphi^{-1}\alpha$, de telle sorte que l'étude de α est équivalente à celle de Y. Cependant, α est défini de sorte à ce que l'essentiel de ses variations soit contenu dans ses premières composantes. On peut donc réduire la dimension du problème en ne conservant que les premières PCs :

$$Y \approx \varphi^{-1} \begin{pmatrix} \alpha^0 \\ \vdots \\ \alpha^p \\ \bar{\alpha}^{p+1} \\ \vdots \\ \bar{\alpha}^{m-1} \end{pmatrix}, \tag{4.35}$$

où $\bar{\alpha}^k = \mathbb{E}[\alpha^k]$ et $p \in [\![1, m-1]\!]$.

Dans notre cas, nous n'avons pas accès à la distribution théorique de Y. En pratique ([137]), on utilise un data set qui est un échantillonnage supposément représentatif de Y et on utilise les variances et espérances discrètes pour calculer les PCs.

4.3.3 Métamodèle complet par couplage PCE-PCA

On revient maintenant au PCE. Jusqu'à présent, le métamodèle est donné par :

$$\mathcal{S}(X) = (\Phi^0(X), ..., \Phi^{m-1}(X)) \approx \mathcal{M}(X), \tag{4.36}$$

où Φ^k est le résultat du PCE appliqué au mapping $X \mapsto \mathcal{M}(X)_k = Y_k$. On propose ici de remplacer ces m différents PCEs appliqués aux composantes de Y par des PCEs appliqués aux premières PCs de Y. Plus précisément, on suppose que les $p+1$ premières PCs de Y rendent compte de l'essentiel de la variance de Y. Alors pour $0 \leq k \leq p$, on appelle θ^k le résultat du PCE appliqué au mapping $X \mapsto \alpha^k(X)$ (en vertu de (4.33), si Y est vu comme une fonction de X, alors α^k peut aussi l'être). On l'utilise pour le métamodèle final :

$$\hat{\mathcal{S}}(X) = \varphi^{-1} \begin{pmatrix} \theta^0(X) \\ \vdots \\ \theta^p(X) \\ \bar{\alpha}^{p+1} \\ \vdots \\ \bar{\alpha}^{m-1} \end{pmatrix} \approx \mathcal{M}(X). \tag{4.37}$$

On rappelle que φ et les $\bar{\alpha}^k$ sont des quantités déterministes qui sont précalculées pendant l'étape de PCA. On a donc seulement $p+1$ PCEs (donc $p+1$ résolutions du problème (4.28)) à effectuer au lieu de m. Si $p \ll m$, on obtient alors une drastique réduction du temps de calcul.

Récapitulons donc la procédure :

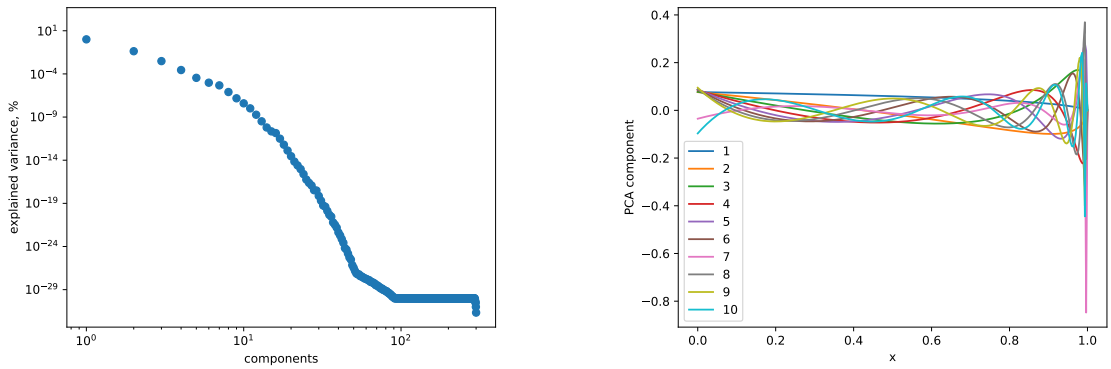
- on commence par utiliser le solveur présenté en section 4.1.3 pour obtenir un échantillonnage de l'ensemble des profils de hauteur pouvant découler des valeurs des paramètres B et S.
- On procède ensuite à une PCA sur ce data set. Cela nous fournit alors un ensemble de vecteurs α correspondant à notre data set, ainsi que la matrice φ^{-1}. Ceci se fait en temps très rapide par les bibliothèques de calcul classiques, on utilise ici la bibliothèque SciPy de Python. On choisit ensuite le nombre $p+1$ de PCs que l'on décide de conserver, puis on calcule les espérances $\bar{\alpha}^k$ pour les autres PCs.
- On procède à $p+1$ PCEs distincts sur chacune des PCs que l'on conserve, ce qui revient à résoudre $p+1$ fois le problème (4.28).
- Une fois ces étapes offlines terminées, on peut évaluer le métamodèle en utilisant (4.37), soit un produit matriciel et quelques évaluations polynomiales.

Remarque 69 *Une telle combinaison entre PCA et PCE ne semble pas courante dans la littérature. À la connaissance de l'auteur, cela a été utilisé pour la première fois par Blatman et Sudret en 2013 [48], puis en 2017 par Hawchar et al. [125].*

4.4 Résultats numériques

Dans cette section, on travaille avec $n = 1$. Toutefois, la méthodologie et les codes peuvent être utilisés pour d'autres valeurs de n (typiquement entre 0.2 et 1.2). De plus, comme mentionné précédemment, le modèle de lubrification utilisé est régulier vis-à-vis des valeurs de n considérées ici ($n \sim 1$). Enfin il s'avère que les pâtes expérimentales utilisées en section 4.4.3 sont telles que $n = 1$ (voir Vigneaux et al. [244]).

Tous les calculs sont effectués en Python. L'entraînement du PCE (i.e. la résolution de (4.28)) est effectué par la bibliothèque Chaospy [89] couplée à Numpoly. On utilise Scipy pour la PCA ainsi que pour l'algorithme d'inversion par Nelder-Mead (voir section 4.4.4).



(a) Pourcentage de variance expliquée pour chaque PC

(b) Illustration des PCs dans la cavité

FIGURE 4.4.1 – Visualisation de la PCA appliquée aux profils de hauteur échantillonnés. Les composantes sont ordonnées en fonction de leur variance expliquée, dans l'ordre décroissant. La figure (a) donne les pourcentages de variance expliquée pour chaque composante en échelle log-log. La figure (b) montre les 10 premières composantes principales.

4.4.1 Analyse en composantes principales

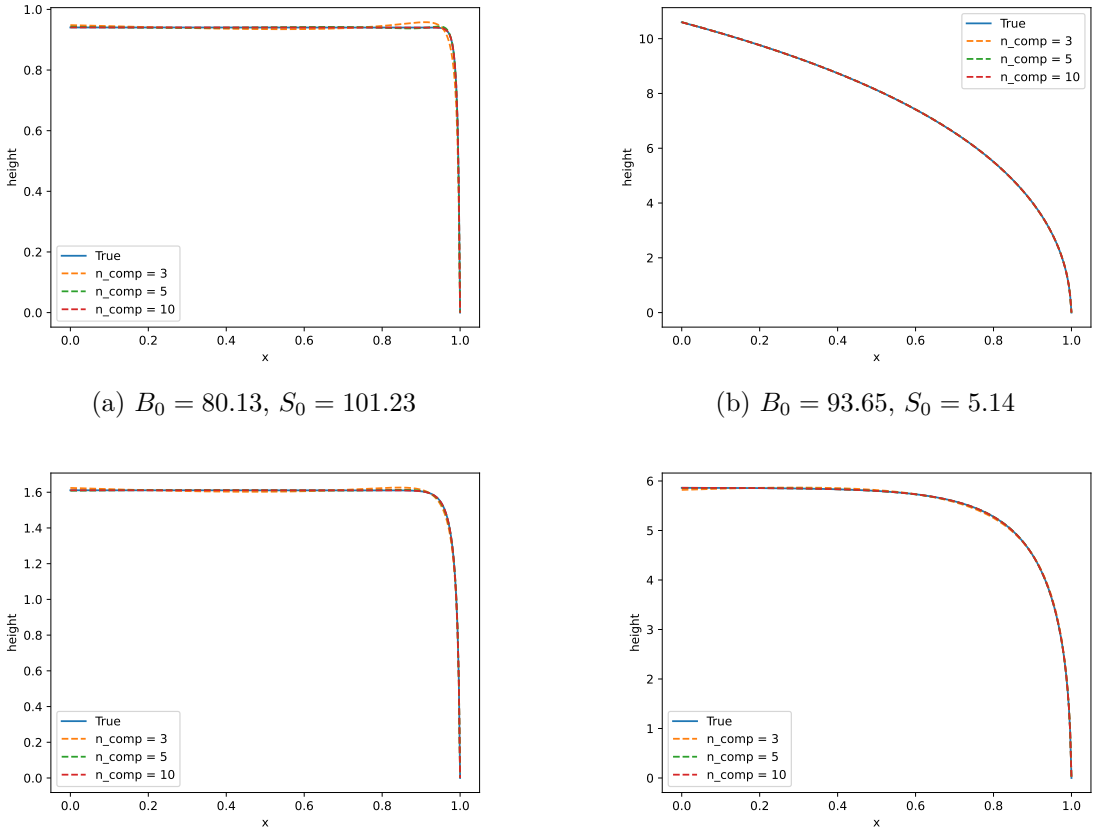
On commence par une étude préliminaire afin de vérifier la pertinence de l'utilisation de la PCA décrite en section 4.3.2. On applique pour cela la PCA à la grille aléatoire de 6000 couples évoquée en section 4.2.1. Une façon d'évaluer la pertinence d'une PCA est de calculer, pour chaque composante principale, le pourcentage de la variance du data set qui est expliquée par ladite composante. Le graphe résultant est présenté en figure 4.4.1a. On fournit aussi à titre indicatif la forme des premières PCs dans la figure 4.4.1b.

La première PC explique 98.94% de la variance totale, la deuxième en explique 1.01%, les variances suivantes sont inférieures à 0.01%. La majeure partie de l'information est donc bien contenue dans seulement quelques composantes, ce qui justifie la pertinence de la procédure. Ceci étant dit, il est difficile d'estimer combien de PCs il sera nécessaire d'utiliser seulement à partir de la figure 4.4.1. On montre donc sur la figure 4.4.2 des comparaisons entre les profils de hauteur du data set et leur reconstruction à partir des PCs tronquées selon la procédure décrite en section 4.3.2.

On voit que pour les nombres de PCs présentés (3, 5 et 10), la reconstruction est de bonne qualité dans tous les cas. Cependant, pour 3 PCs, on observe des sortes de "remontées" indésirables (voir figures 4.4.2a et 4.4.2c). Pour 5 et 10 PCs, ces artefacts semblent drastiquement réduits et il devient difficile de distinguer les véritables profils de leur reconstruction. On complète cette observation visuelle par un calcul de la norme de la différence entre les profils réels et leur reconstruction pour l'ensemble du data set, dont on représente les résultats sous la forme d'un boxplot en figure 4.4.3. On confirme bien une drastique réduction de l'erreur au fur et à mesure que l'on ajoute des PCs, avec une erreur quasiment négligeable pour 10 PCs. Pour la suite, on essaie donc d'utiliser une PCA avec moins de 10 PCs.

4.4.2 Performances du métamodèle

Afin de calculer les coefficients du métamodèle, la question de l'échantillonnage du domaine (B, S) se pose à nouveau (voir Hadigol et Doostan (2018) [123] pour des détails sur les stratégies



(a) $B_0 = 80.13$, $S_0 = 101.23$

(b) $B_0 = 93.65$, $S_0 = 5.14$

(c) $B_0 = 109.53$, $S_0 = 74.02$

(d) $B_0 = 237.61$, $S_0 = 41.16$

FIGURE 4.4.2 – Profils de hauteur réels comparés avec leur reconstruction en utilisant différents nombres de composantes principales, pour différents couples.

d'échantillonnage pour les PCEs). Plutôt que de travailler avec la grille aléatoire décrite en section 4.2.1, on choisit de travailler avec une petite grille régulière de 400 couples (20 valeurs différentes de B et 20 valeurs de S, équiréparties sur leur domaine respectif) puis une grille de taille plus conséquente avec 6084 couples (78 valeurs de B, 78 valeurs S, toujours équiréparties). Le solveur EDP présenté en 4.1.3 est ensuite appliqué à chacun des couples de ces grilles. Il est à noter que les couples pour lesquels le solveur est le plus lent semblent être les mêmes pour toutes les valeurs de n, ce qui signifie que pour des calculs futurs, il sera possible d'utiliser les données fournies avec ce travail pour estimer la vitesse de calcul associée à de nouvelles valeurs du triplet (B, S, n), ce qui implique qu'une parallélisation efficace de l'évaluation d'un nouveau data set peut être aisément implémentée. En pratique, cinq heures sur sept processeurs (sur un ordinateur domestique) suffisent pour calculer la petite grille.

Il faut maintenant définir la précision d'un métamodèle. En plus des deux grilles régulières mentionnées ci-dessus, on rappelle qu'on dispose d'un data set d'environ 6000 couples tirés au hasard (uniformément sur le domaine de (B, S)). Alors étant donné un métamodèle $\bar{\mathcal{M}}$, on l'évalue sur l'ensemble des couples de ce data set puis on le compare aux sorties réelles $\mathcal{M}(B, S)$ du modèle exact. Pour chaque couple (B, S) du data set de validation, on définit l'*erreur de reconstruction* $\|\mathcal{M}(B, S) - \bar{\mathcal{M}}(B, S)\|_2$. Plusieurs statistiques calculées sur le data set de validation pour différents paramètres du métamodèle sont présentées dans la table 4.4.1.

De multiples ordres de troncature pour le PCE ont été testés, de 4 à 30. Comme attendu,

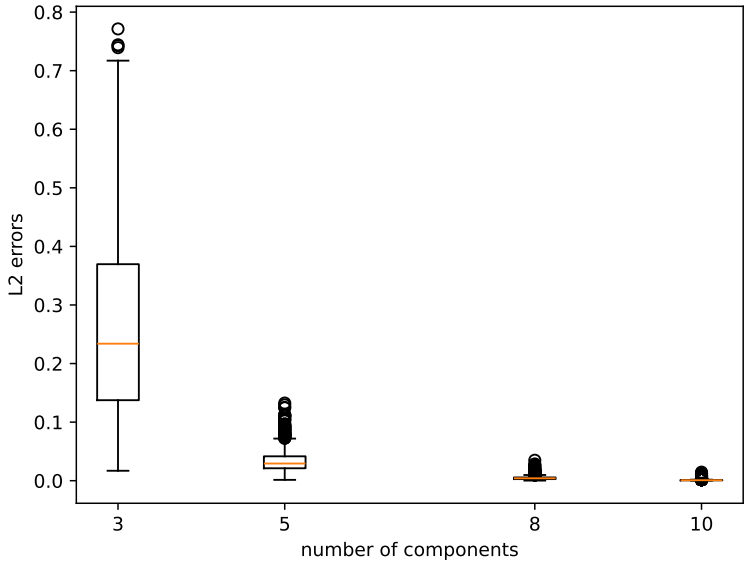


FIGURE 4.4.3 – Boxplot de la norme L^2 de l'erreur de reconstruction en utilisant seulement un nombre restreint de PCs, calculé sur le data set complet.

plus l'ordre est élevé, plus la précision du métamodèle augmente. Si l'on augmente l'ordre de 4 à 6, l'erreur est divisée par 3. De 10 à 12, elle est à nouveau divisée par 2. À l'ordre 30, pour la grille la plus fine, il y a environ 3% d'erreur pour la plupart des couples du data set de validation. Il y a cependant une limite à cette précision due au nombre fini de couples dans le data set d'entraînement, étant donné que le système devient indéterminé lorsque le degré de troncature devient trop important (le nombre de polynômes l, et donc le nombre de coefficients dans le système (4.28), augmentent quadratiquement avec le degré de troncature !). On l'observe pour la plus petite grille, qui ne voit plus d'amélioration de précision lorsque l'on dépasse l'ordre 15. Une autre limite pratique provient du temps de calcul de la résolution de (4.28) : pour un ordre petit tel que 4 ou 5, la résolution ne prend que quelques minutes, mais pour l'ordre 30, ce calcul prend plusieurs jours.

Revenons sur la discussion de la section précédente, à savoir le choix du nombre de composantes principales à conserver. Ce choix a un impact sur la qualité du métamodèle. Avec 3 PCs (i.e. $p = 2$ si l'on suit les notations de la section 4.3.2), la qualité du métamodèle atteint un plateau de précision pour des troncatures d'ordre supérieur à 10. En utilisant 10 PCs (i.e. $p = 9$), la précision n'est presque pas affectée avant l'ordre 30, ce qui est cohérent avec l'analyse de la section 4.4.1. On utilise donc ce nombre dorénavant. Ainsi, en utilisant seulement 10 PCs, on ne doit résoudre que 10 fois (4.28). Sans cette modification, il est nécessaire d'effectuer $n_x = 301$ PCEs distincts. Le temps de calcul de cette phase est donc mécaniquement divisé par 30, pratiquement sans perte de précision.

Remarque 70 *Dans le tableau 4.4.1, la dernière colonne fournit l'erreur maximale observée sur l'ensemble du data set de validation. Il semble que quels que soient les paramètres de métamodèle utilisés, il y a toujours quelques couples dont l'erreur de reconstruction est significativement élevée. Même le plus précis des métamodèles testés dans ce travail a une erreur maximale de 10%, ce qui peut être potentiellement problématique pour une procédure d'estimation de paramètres. Cependant, ces cas extrêmes ne se produisent que très rarement et une recherche approfondie*

TABLE 4.4.1 – Différentes statistiques de l'erreur de reconstruction pour différents métamodèles, où β est l'ordre de troncature du PCE. Lorsque la PCA est utilisée, on retient seulement 10 PCs.

Data set	β	PCA	médiane	3^e quartile	max
78x78	4	non	1	1.8	16.8
		oui	1	1.8	16.8
	15	non	0.018	0.027	0.59
		oui	0.018	0.027	0.59
	30	non	0.0011	0.002	0.082
		oui	0.0018	0.0024	0.082
20x20	4	non	1.1	1.9	15.8
		oui	1.1	1.9	15.8
	15	non	0.021	0.032	1.13
		oui	0.021	0.032	1.13

de ces cas a montré qu'ils étaient presque systématiquement situés proches de la frontière du domaine (B, S), pour des faibles valeurs de B et le plus souvent des faibles valeurs de S (un domaine proche de celui décrit par la figure 4.1.3). Rappelons que le domaine (B, S) a été choisi de telle sorte à ce que pour les applications considérées, les valeurs des paramètres se trouveront relativement éloignées de la bordure du domaine (aussi il est conseillé de conserver cette stratégie si l'on souhaite calculer un nouveau métamodèle pour une autre application). Il en résulte que l'on peut supposer que dans les cas concrets, les erreurs seront généralement bornées par le 3e quartile, que l'on arrive à contrôler plus raisonnablement.

Remarque 71 *D'autres alternatives se présentent à l'erreur absolue $\|\mathcal{M}(B,S) - \bar{\mathcal{M}}(B,S)\|_2$. De fait, cette étude a aussi été effectuée pour l'erreur relative $\dfrac{\|\mathcal{M}(B,S) - \bar{\mathcal{M}}(B,S)\|_2}{\|\mathcal{M}(B,S)\|_2}$. Les conclusions sont essentiellement les mêmes, aussi on ne présente ici que l'erreur absolue.*

Jusqu'à présent, nous nous sommes focalisés sur la précision de la prédiction du métamodèle. Cependant, on rappelle que la raison d'existence du métamodèle est qu'une procédure d'inversion requiert typiquement des centaines d'évaluations du modèle, aussi est-il primordial de pouvoir l'évaluer rapidement. Les métamodèles présentés ci-dessus sont tous drastiquement plus rapides que le modèle réel, mais ils ne sont pas de rapidités égales. Bien que légèrement plus précis, le PCE d'ordre 30 sans PCA nécessite l'évaluation de 301 polynômes bivariés d'ordre 30. En comparaison, le PCE d'ordre 15 avec PCA ne demande que 10 polynômes d'ordre 15 et un produit matrice-vecteur. En pratique, le second modèle est 3 fois plus rapide que le premier.

En figure 4.4.4, on présente des exemples typiques de comparaisons entre le solveur EDP et les reconstructions fournies par le métamodèle entraîné avec la grille 20×20, utilisant un PCE d'ordre 15 avec PCA. Trois couples (B, S) représentatifs sont choisis et les profils de hauteur associés donnés par le solveur EDP sont superposés avec les résultats du métamodèle. On observe une excellente adéquation entre le solveur et le métamodèle.

En considérant ces résultats, on utilise pour la suite la procédure PCE-PCA avec $\beta = 15$ (en suivant les notations de la section 4.3.1) et $p = 9$, entraînée sur la petite grille 20×20. Ce métamodèle est à la fois précis et rapide à évaluer, avec une durée d'entraînement de moins de deux jours (en comptant l'évaluation du data set d'entraînement) sur un ordinateur portable domestique.

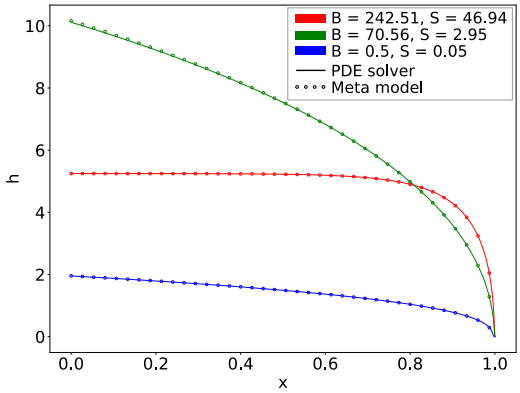


FIGURE 4.4.4 – Comparaison des résultats du solveur EDP (trait plein) et du métamodèle (cercles), pour trois couples de paramètres (B, S) (différentes couleurs).

TABLE 4.4.2 – Statistiques des erreurs (eq. (4.38)) de la procédure d'estimation de paramètres.

Bruit	médiane	3^{e} quartile	max	variance
0%	0.00045	0.00088	0.0299	7.46e-06
2%	0.016	0.043	1.21	0.0276
5%	0.035	0.097	3.07	0.064
10%	0.067	0.18	5.03	0.202

4.4.3 Problème inverse sur données synthétiques

Le but du métamodèle est d'être utilisé pour procéder à de l'estimation de paramètres. En première étape, on le teste sur des données synthétiques, i.e. on utilise des profils du data set de validation (qu'on appelle profils *réels* dans la suite) et on tente de retrouver les valeurs de B et S grâce au métamodèle. Sur le terrain, on s'attend à obtenir des données avec une précision d'environ 5% (voir section 4.4.4), aussi les données du data set sont bruitées avec un bruit d'une intensité similaire. Plus précisément, on échantillonne $h_{noised}(B, S) = h(B, S) + \varepsilon$ où $\varepsilon \sim \mathcal{N}(0, \alpha h(B, S))$. On teste 2%, 5% et 10% de bruit, autrement dit $\alpha = 0.02$, 0.05 ou 0.1. Au bout du compte, pour chaque couple (B, S), on a à notre disposition $h(B, S)$ (le vrai) et 3 profils bruités avec respectivement 2%, 5% et 10% d'intensité. On procède de la sorte pour 500 couples du data set de validation.

L'estimation de paramètres est effectuée par l'algorithme de Nelder-Mead , avec un maximum de 400 itérations (voir Nelder et Mead (1965) [175], Bierlaire (2018) [44] section 15.1, et la documentation de Scipy pour une description détaillée de l'algorithme). L'erreur de l'estimation est définie comme la distance euclidienne entre le couple (B, S) réel et le couple estimé. Les résultats sont résumés dans le tableau 4.4.2. L'erreur présentée est l'erreur relative des couples estimés, i.e. :

$$err = \left\| \left(\frac{B_{real} - B_{estim}}{B_{real}}, \frac{S_{real} - S_{estim}}{S_{real}} \right) \right\|_2 . \tag{4.38}$$

Remarque 72 *De manière analogue à la remarque 71 sur les différentes métriques possibles d'évaluation des métamodèles, il serait ici aussi possible d'utiliser d'autres formules. À nouveau, plusieurs notions d'erreur ont été testées et donnent des résultats similaires.*

Pour les couples non-bruités, l'estimation est pratiquement exacte et pour 2% de bruit, il y a généralement moins de 5% d'erreur. Pour 5% de bruit, la qualité de l'inversion diminue légère-

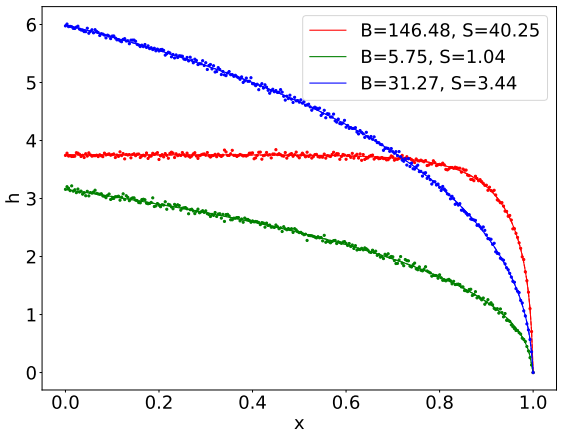


FIGURE 4.4.5 – Comparaison entre un profil de hauteur réel (trait plein), sa version bruitée (cercles) et le profil trouvé par l'algorithme de Nelder-Mead (traits pointillés). Les profils bruités sont créés à partir de 2% de bruit. Les trois couleurs correspondent à trois couples (B, S) différents choisis aléatoirement. À ce niveau de zoom, le trait plein et le trait pointillés sont superposés, aussi un seul des deux est visible.

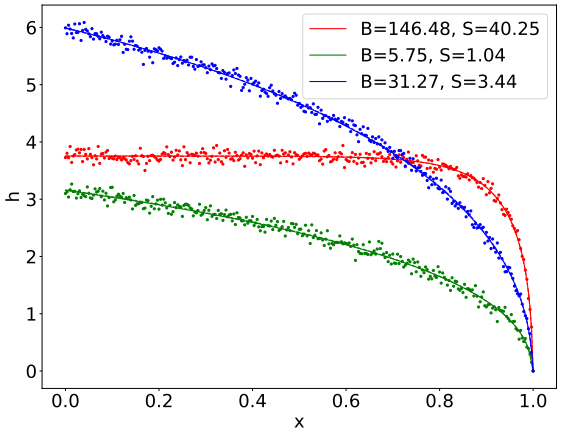


FIGURE 4.4.6 – Comparaison entre un profil de hauteur réel (trait plein), sa version bruitée (cercles) et le profil trouvé par l'algorithme de Nelder-Mead (traits pointillés). Les profils bruités sont créés à partir de 5% de bruit. Les trois couleurs correspondent aux trois couples (B, S) utilisés en figure 4.4.5. À ce niveau de zoom, le trait plein et le trait pointillés sont superposés, aussi un seul des deux est visible.

ment avec environ 10% d'erreur dans l'estimation. Il est à noter que ces profils sont extrêmement chaotiques par rapport à ce que l'on peut attendre de données expérimentales. Malgré cela, les profils correspondant aux couples déterminés par l'algorithme sont très proches des profils réels, comme le montrent les figures 4.4.5, 4.4.6, 4.4.7. Pour des profils de hauteur avec 10% de bruit, une grande prudence devient nécessaire car l'estimation de (B, S) peut diverger de manière plus significative, bien que l'on attend de données réelles qu'elles soient bien moins chaotiques. Les résultats pour 10% de bruit sont à voir comme un cas "au pire".

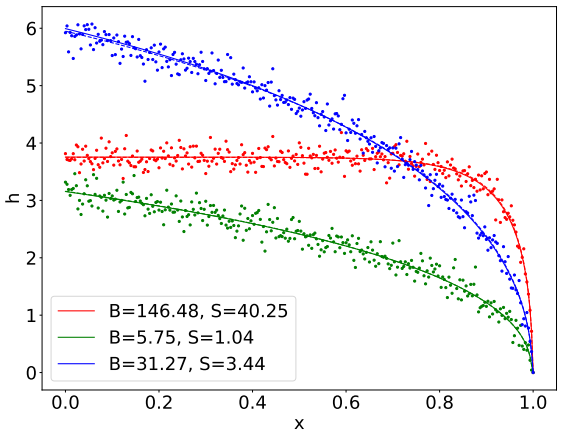


FIGURE 4.4.7 – Comparaison entre un profil de hauteur réel (trait plein), sa version bruitée (cercles) et le profil trouvé par l'algorithme de Nelder-Mead (traits pointillés). Les profils bruités sont créés à partir de 10% de bruit. Les trois couleurs correspondent aux trois couples (B, S) utilisés en figures 4.4.5 et 4.4.6. À ce niveau de zoom, le trait plein et le trait pointillés sont superposés sur presque toute la longueur du profil (et c'est sur le profil bleu que l'on commence à voir plus clairement une légère différence).

TABLE 4.4.3 – Estimation de paramètres pour des profils h en provenance des expériences de laboratoire de [244]. La colonne "#" donne le numéro de la pâte considérée. La colonne des valeurs expérimentales de B contient deux valeurs qui ont été déterminées par deux techniques rhéologiques distinctes. Les colonnes "var" donnent le pourcentage de variation entre les valeurs fournies par [244] et l'estimation de paramètres effectuée ici.

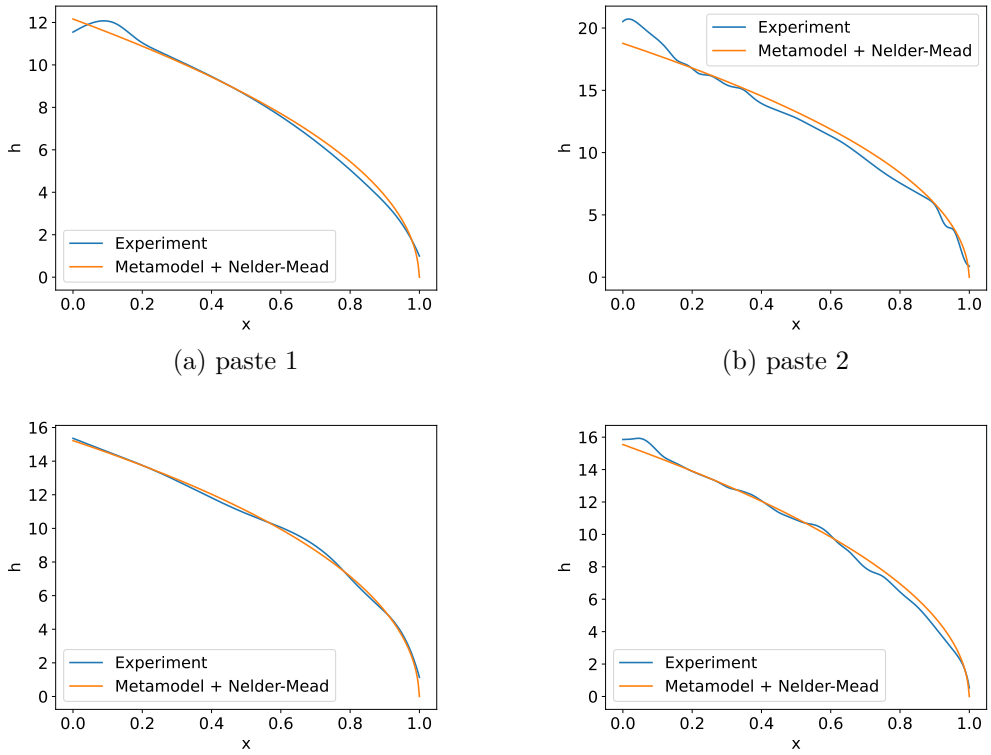
#	B : expérimental	Nelder-Mead	% var	S : expérimental	Nelder-Mead	% var
1	[43 ;58]	73	69 ; 26	0.22	0.05	-77
2	[102 ;122]	176	73 ; 44	0.28	0.05	-82
3	[74 ;98]	137	85 ; 40	0.28	2.2	686
4	[68 ;93]	120	76 ; 29	0.26	0.05	-81

4.4.4 Comparaisons aux données expérimentales

Dans cette section, on applique l'algorithme d'inversion aux données expérimentales mesurées en laboratoire présentées par Vigneaux et al. en 2023 [244]. Dans cet article, quatre pâtes sont utilisées pour effectuer quatre remplissages distincts d'une boîte vide, correspondant à un modèle réduit de laboratoire pour des expériences CPB, avec des pâtes provenant du terrain. Pour chaque expérience, une image du profil au wall-touch est capturée (voir par exemple la figure 16 de [244]). Cela nous fournit alors quatre courbes $x \mapsto h(x)$ que l'on peut utiliser pour estimer les paramètres (B, S) de chacune des pâtes.

Les résultats sont présentés en figure 4.4.8. On commence par faire remarquer que les données expérimentales montrent un bruit d'ordre similaire au bruit virtuel simulé en section précédente pour les intensités 2%-5%. Le bruit des données expérimentales n'est pas tant dû au traitement de l'image qu'à la nature très "épaisse" de la rhéologie des pâtes (ceci explique notamment le monticule que l'on observe au début des courbes). Les paramètres estimés par notre procédure sont présentés dans le tableau 4.4.3.

Il est importante de noter que les valeurs de (B, S) expérimentales ont été estimées par [244]



(a) paste 1 (b) paste 2

(c) paste 3 (d) paste 4

FIGURE 4.4.8 – Fitting via la procédure d'estimation de paramètres pour quatre jeux de données réels issus de [244]. Pour chaque pâte sont affichés en bleu le profil de hauteur expérimental $h(x)$ (variables adimensionnées, voir 4.1.2) et en orange la courbe associée au couple (B, S) estimé par l'algorithme.

selon deux techniques distinctes. La première technique consistait à fitter la loi de comportement de Bingham, la seconde était basée sur une expérience de réponse au cisaillement constant (la réponse étant alors l'évolution de la contrainte en fonction du temps et le seuil viscoplastique étant déterminé comme le maximum de cette réponse). On attire l'attention sur le fait que l'estimation de B en laboratoire est une tâche non triviale, ce à cause de l'utilisation d'un matériau réel typique du CPB, pour lequel des mesures rhéologiques précises sont difficiles à obtenir. Cela se retrouve dans la variabilité significative que l'on observe entre les deux valeurs de B fournies par les deux techniques, comme le montre la seconde colonne du tableau 4.4.3. Une autre difficulté provient du fait que les expériences peuvent former des monticules dus au caractère 3D de l'écoulement réel (au niveau de l'injection ; voir les pâtes 1, 2 et 4 de la figure 4.4.8, à proximité de $x = 0.1$), ces monticules ne pouvant pas être capturés par le modèle 2D (et non 3D) utilisé ici. Dans cette étude, ce monticule se traduit par une perturbation en forme de bosse pour x proche de 0.1, ce qui peut aussi être vu comme du bruit sur la courbe de $h(x)$. Ceci risque de mettre en difficulté la procédure d'inversion qui chercher à fitter le modèle EDP 2D avec les courbes expérimentales.

On mentionne aussi que les expériences de laboratoire ont été réalisées avec des valeurs de S très faibles à cause de la faible pente du dispositif expérimental (contrairement aux cas CPB opérationnels qui mettent en jeu des pentes bien plus importantes, voir section 4.1.2). Cela implique que les tests numériques se trouvent sur une des parties les plus délicates du domaine

(B, S) pour la procédure, en effet :

1. la grille la plus petite a été utilisée pour l'entraînement du métamodèle, il y a donc très peu de valeurs proches de la bordure du domaine qui interviennent dans l'entraînement (les deux plus petites valeurs de S utilisées dans le data set d'entraînement sont 0.05 et 6, aucune entre les deux), on peut donc attendre du métamodèle qu'il n'ait qu'une précision limitée dans ces valeurs extrêmes de S, on renvoie à la section 4.4.2 pour une discussion détaillée ;
2. une inspection minutieuse des profils de hauteur associés à différents (B, S) montrent que pour les valeurs de S très faibles (en particulier dès que $S < 1$), une faible perturbation de S engendre une forte modification de la solution du solveur EDP, aussi à la fois l'entraînement du métamodèle et la résolution du problème inverse sont naturellement beaucoup plus complexes dans cette zone.

Toutes ces raisons expliquent la difficulté de ce test particulier, et de ce fait la variation significative entre nos résultats et ceux de [244], présentée en tableau 4.4.3.

Ceci étant dit, il est important de noter que les courbes obtenues par notre procédure métamodèle+inversion correspondent très bien aux courbes expérimentales (figure 4.4.8). On fait aussi remarquer que la procédure d'inversion est extrêmement rapide comparée à une inversion qui aurait utilisé la résolution de l'EDP. Ainsi, étant donnés le bruit rencontré dans les matériaux du CPB et le fait que leurs valeurs typiques de (B, S) se trouvent plus proches du cœur du domaine (B, S) utilisé pour l'entraînement du métamodèle (et non à la frontière comme les expériences de [244]), il semble que cette procédure métamodèle+inversion puisse être utilisée en labo en tant que première estimation de la rhéologie d'expériences CPB, basée sur la seule courbe $x \mapsto h(x)$ au wall-touch.

À cet effet, on dépose en accès libre sur le site de Zenodo [259] les codes ayant produit ce travail, ainsi que l'ensemble des données du métamodèle [40], de sorte que d'autres équipes puissent tester notre méthodologie sur d'autres données et/ou entraîner un nouveau métamodèle pour d'autres valeurs de n ou un autre domaine de (B, S). Le code est écrit en Python et des routines permettant de facilement reproduire et étendre les calculs présentés ont été implémentées, de sorte à ce que l'investissement nécessaire soit minimal.

4.5 Perspectives

De nombreuses directions de recherche sont envisageables suite à ce travail. Une extension naturelle consisterait à envisager d'autres méthodes que le PCE pour construire le métamodèle. On pourrait par exemple s'intéresser au krigeage (régression par des processus gaussiens, voir par exemple la récente review de Marrel et Iooss [159] et les références qu'elle contient), ou aux méthodes de machine learning plus généralement (voir par exemple [224]).

Un autre projet pourrait inclure une étude quantitative de l'influence de l'angle ϕ (impliqué dans le calcul de S), voire une étude plus avancée sur des variations de la pente (i.e. de la topographie du sol de la cavité) dues aux couches précédemment déposées. Il est en effet commun en CPB d'effectuer des injections successives de différentes couches de pâte (voir [166]).

Annexe du chapitre 4

4.A Dérivée de q

On donne ici différentes expressions de $\partial_x q$ qui pourraient se révéler utiles pour des travaux futurs. On rappelle la notation : $\delta = sign\,(S - h_x)$ où $h_x = \partial h/\partial x$.

$$q(x) = \frac{nY^{1+1/n}|S-h_x|^{1/n}\delta}{(1+n)(1+2n)}((2n+1)h - nY) \tag{4.39}$$

$$\begin{aligned}\partial_x q(x) =& \frac{\delta}{1+2n}Y_x Y^{1/n}|S-h_x|^{1/n}\Big((2n+1)h - nY\Big)\\ &- \frac{1}{(1+n)(1+2n)}Y^{1+1/n}h_{xx}|S-h_x|^{1/n-1}\Big((2n+1)h - nY\Big)\\ &+ \frac{n\delta}{(1+n)(1+2n)}Y^{1+1/n}|S-h_x|^{1/n}\Big((2n+1)h_x - nY_x\Big)\end{aligned} \tag{4.40}$$

$$\begin{aligned}\partial_x q(x) =& \frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big((1+n)(2n+1)Y_x h - (1+n)nY_x Y\Big)\\ &+ \frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big(\frac{\delta n Y^2 h_{xx}}{|S-h_x|} - \frac{\delta(2n+1)Y h_{xx}h}{|S-h_x|}\Big)\\ &+ \frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big(n(2n+1)Y h_x - n^2 Y_x Y\Big)\end{aligned} \tag{4.41}$$

On obtient la première formule utilisable :

$$\begin{aligned}\partial_x q(x) =& \frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big[(1+n)(2n+1)Y_x h - (1+2n)nY_x Y\\ &+ \frac{\delta n Y^2 h_{xx}}{|S-h_x|} - \frac{\delta(2n+1)Y h_{xx}h}{|S-h_x|} + n(2n+1)Y h_x\Big]\end{aligned} \tag{4.42}$$

Quand $Y \neq 0$:

$$Y_x = h_x - \delta\frac{h_{xx}B}{(S-h_x)^2} \tag{4.43}$$

D'où :

$$\begin{aligned}\partial_x q(x) =& \frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big[(1+n)(2n+1)h_x h - (1+n)(2n+1)h\delta\frac{h_{xx}B}{(S-h_x)^2}\\ &- (1+2n)nh_x Y + (1+2n)nY\delta\frac{h_{xx}B}{(S-h_x)^2} + \frac{\delta n Y^2 h_{xx}}{|S-h_x|} - \frac{\delta(2n+1)Y h_{xx}h}{|S-h_x|} + n(2n+1)Y h_x\Big]\end{aligned} \tag{4.44}$$

On obtient une formulation plus naturelle ne dépendant que de Y, h, h_x et h_{xx} :

$$\begin{aligned}\partial_x q(x) =&\frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big[(1+n)(2n+1)h_x h-(1+n)(2n+1)h\delta\frac{h_{xx}B}{(S-h_x)^2}\\&+\frac{\delta Y h_{xx}}{|S-h_x|}\left(nY+n(1+2n)\frac{B}{|S-h_x|}-(1+2n)h\right)\Big]\end{aligned} \tag{4.45}$$

On explicite progressivement Y :

$$\begin{aligned}\partial_x q(x) =&\frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big[(1+n)(2n+1)h_x h-(1+n)(2n+1)h\delta\frac{h_{xx}B}{(S-h_x)^2}\\&+\frac{\delta Y h_{xx}}{|S-h_x|}\left(2n^2\frac{B}{|S-h_x|}-(1+n)h\right)\Big]\end{aligned} \tag{4.46}$$

$$\begin{aligned}\partial_x q(x) =&\frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big[(1+n)(2n+1)h_x h-(3n+1)h\delta\frac{h_{xx}B}{(S-h_x)^2}\\&-\delta(1+n)\frac{h_{xx}h^2}{|S-h_x|}-2n^2\frac{\delta B^2 h_{xx}}{|S-h_x|^3}+(1+n)\frac{\delta h_{xx}hB}{(S-h_x)^2}\Big]\end{aligned} \tag{4.47}$$

Une autre formule intéressante :

$$\begin{aligned}\partial_x q(x) =&\frac{\delta}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n}\Big[(1+n)(2n+1)h_x h-2nh\delta\frac{h_{xx}B}{(S-h_x)^2}\\&-\delta(1+n)\frac{h_{xx}h^2}{|S-h_x|}-2n^2\frac{\delta B^2 h_{xx}}{|S-h_x|^3}\Big]\end{aligned} \tag{4.48}$$

La dernière formule qui explicite les coefficients de transports et diffusion :

$$\begin{aligned}\partial_x q(x) =&h_x\left[\delta Y^{1/n}|S-h_x|^{1/n}h\right]\\&+h_{xx}\left[\frac{-1}{(1+n)(1+2n)}Y^{1/n}|S-h_x|^{1/n-1}\left(2nh\frac{B}{|S-h_x|}+(1+n)h^2+2n^2\frac{B^2}{(S-h_x)^2}\right)\right].\end{aligned} \tag{4.49}$$

4.B TDA

On donne ici une brève présentation de l'analyse topologique de données (TDA). On se limite ici aux concepts nécessaires à la compréhension des diagrammes de persistance utilisés en section 4.2.2. Pour une introduction plus complète à ce domaine, on recommande le livre de Edelsbrunner et Harer [88].

4.B.1 Complexe simplicial

Un *d-simplexe* σ est l'enveloppe convexe d'un ensemble S de $d+1$ points affinement indépendants d'un espace euclidien de dimension n, avec $0 \leq d \leq n$. On appelle d la *dimension* du simplexe. Chaque d'-simplexe τ dont les points appartiennent à S est appelé *face* de σ (pour $0 \leq d' \leq d$). On écrit alors $\tau \leq \sigma$. Des exemples de simplexes sont donnés en figure 4.B.1. Par exemple, les faces d'un tétraèdre sont tous ses sommets, ses arrêtes, ses triangles et lui-même.

Un *complexe simplicial* K est une collection finie non-vide de simplexes qui contient toutes les faces de tous ses simplexes et tel que l'intersection de deux simplexes de K est une face commune de ces simplexes. On donne des exemples en figure 4.B.2. Dans l'idée, c'est ce qui est utilisé pour définir un maillage sur le domaine étudié. Dans notre cas, les sommets sont les couples (B, S).

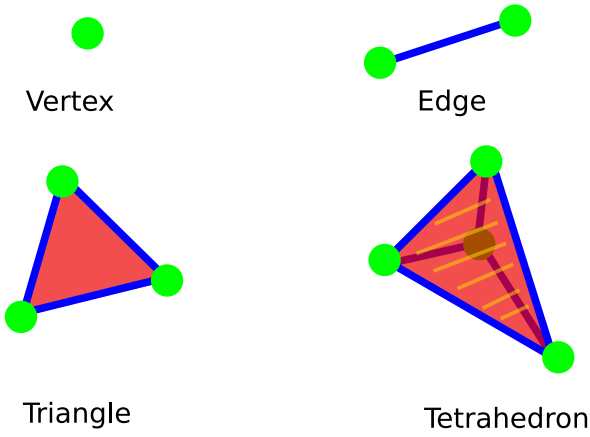


FIGURE 4.B.1 – Exemples de d-simplexes pour $0 \leq d \leq 3$ avec leurs faces. Les différentes couleurs sont utilisées pour distinguer les différentes faces de chaque simplexe et sont cohérentes avec les figure suivantes dans cette section.

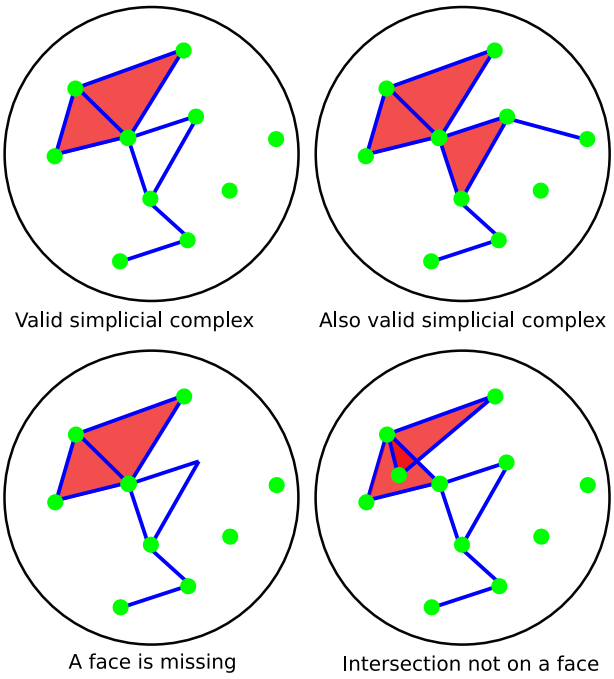


FIGURE 4.B.2 – Exemples de collection de simplexes. Certains remplissent les conditions pour être qualifiés de complexe simplicial, d'autres non.

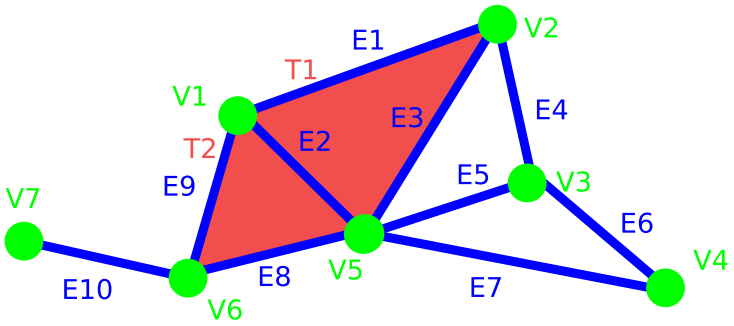


FIGURE 4.B.3 – Un complexe simplicial, les noms de sommets commencent par un V, les noms d'arêtes par un E, les noms de triangles par un T. Les couleurs sont utilisées pour faciliter la reconnaissance des simplexes.

4.B.2 Homologie

On définit maintenant le concept qui va encapsuler les caractéristiques topologiques que l'on souhaite détecter. Soit K un complexe simplicial. Une *p-chaîne* de K est une somme formelle de p-simplexes de K dont les coefficients appartiennent à $\mathbb{Z}/2\mathbb{Z}$. Pour $p = 1$ (on considère donc des arrêtes), cela peut être vu comme un chemin dans le complexe simplicial. On définit la *bordure* de cette p-chaîne comme la somme formelle des $(p-1)$-faces des simplexes de la chaîne. Une p-chaîne est appelée un *p-cycle* si sa bordure est égale à 0 (on parle aussi de bordure vide, étant donné que l'on interprète souvent une chaîne comme la collection de simplexes avec des coefficients non-nuls dans sa somme).

Concrétisons ces définitions en prenant le complexe de la figure 4.B.3. La 0-bordure de la 1-chaîne E1 est V1+V2. La bordure de E1+E4+E7+E3 est V1+V3+V4. La chaîne E5+E6+E7 est un cycle. La chaîne E1+E3+E8+E9 est la bordure de T1+T2.

On remarque que la bordure d'une bordure est toujours 0. Cela signifie que l'ensemble des p-bordures est inclus dans l'ensemble des p-cycles, mais il n'y a pas égalité (par exemple E5+E6+E7). On note l'ensemble des p-bordures B_p et l'ensemble des p-cycles Z_p. On peut alors définir le quotient de Z_p par B_p, appelé le p^e *groupe d'homologie* : $H_p = Z_p/B_p$. Cela signifie que deux p-cycles sont équivalents si et seulement si leur somme est une p-bordure.

Les *nombres de Betti* sont définis comme $\beta_p = card(H_p)$. Dans ce travail, on ne s'intéresse qu'à β_0, qui correspond au nombre de composantes connexes du complexe simplicial. En effet, comme un sommet n'a pas de face, chaque 0-chaîne est un 0-cycle. Alors deux 0-chaînes sont équivalentes dans H_0 si et seulement si leur somme est une 0-bordure i.e. la bordure d'une 1-chaîne i.e. la bordure d'une somme d'arêtes. Donc deux 0-chaînes sont équivalentes si et seulement si elles sont liées par des arrêtes, donc si et seulement si elles sont dans la même composante connexe.

Remarque 73 *En réalité, il faut des hypothèses supplémentaires sur le complexe simplicial pour définir tout cela, cependant cela n'est pas pertinent dans notre cas (simple). On réfère à [88].*

4.B.3 Filtration

Pour un complexe simplicial K, une *filtration* de K est une suite de sous-complexes $\emptyset \subset K_0 \subset K_1 \subset \cdots \subset K_m = K$ telle que $K_{i+1} \backslash K_i$ est un singleton, contenant seulement un simplexe. Intuitivement, c'est une construction simplexe par simplexe de K, de telle sorte qu'à chaque étape on ait toujours un complexe simplicial.

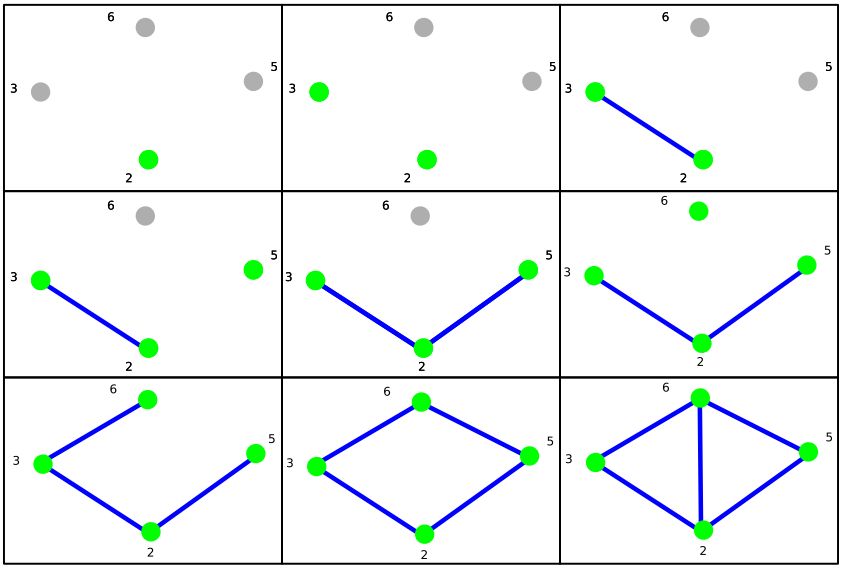


FIGURE 4.B.4 – Exemple de filtration lower star d'un complexe simplicial. Les sommets gris sont des points de maillage qui n'ont pas encore été inclus dans la filtration. Les valeurs affichées sont les valeurs d'une fonction hypothétique f échantillonnée sur les sommets.

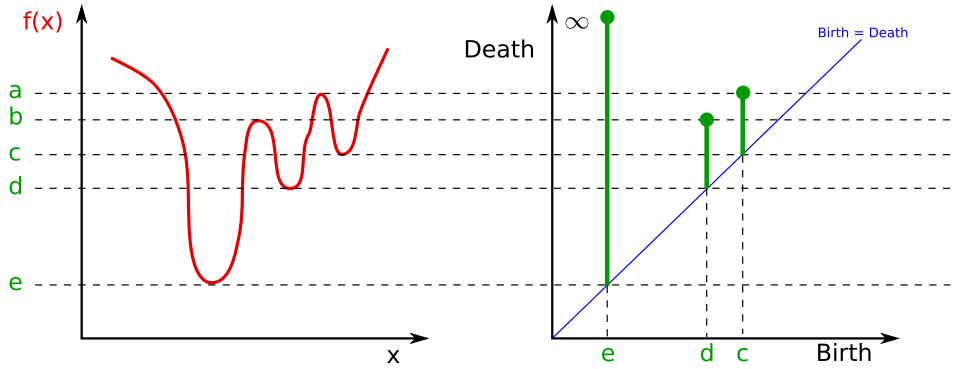


FIGURE 4.B.5 – Exemple d'un diagramme de persistance pour une fonction 1D f.

Dans ce travail, K est un maillage sur lequel une fonction f (la différence entre deux profils de hauteur) a été évaluée. Une filtration particulière peut être calculée dans ce contexte, la *filtration lower star*. On commence avec le point réalisant le minimum de f. Ensuite, on ajoute le point réalisant le minimum des valeurs restantes. S'il y a une arrête entre eux dans K, on l'ajoute à l'étape suivante. On continue jusqu'à ce que tous les points et arrêtes aient été ajoutés. On ajoute bien les arrêtes dès que cela est possible (i.e. dès que ses sommets ont été ajoutés). Si deux choix sont possibles, on choisit un ordre au hasard. Un exemple est donné en figure 4.B.4. Ce processus peut être généralisé en dimension plus élevée.

4.B.4 Homologie persistante

On peut désormais calculer le *diagramme de persistance* d'une fonction. Pour simplifier, on l'explique dans le cas d'une fonction 1D. Pour commencer, retournons dans le cas continu (i.e. on oublie K). L'idée est d'observer les changements de la topologie des sous-niveaux de f à mesure que l'on augmente leur seuil. Plus précisément, on s'intéresse à $\mathcal{L}_f(y) = \{x/f(x) \leq y\}$ pour y

allant de $\min(f)$ à $\max(f)$ (on suppose qu'ils existent). À mesure que l'on fait augmenter y, on traque le nombre de composantes connexes de $\mathcal{L}_f(y)$, comme montré en figure 4.B.5. Chaque fois qu'une nouvelle composante connexe apparaît, on crée une ligne verticale partant du point (y, y) sur le graphe (naissance, mort). On arrête cette ligne lorsque la composante disparaît (on dit qu'elle meurt). La convention est que lorsque deux composantes deviennent une seule, la composante la plus jeune meurt. Il reste toujours au moins une composante à la fin, donnant une ligne de hauteur infinie.

Le processus est exactement le même avec le complexe simplicial. La filtration nous donne les différents sous-niveaux et on peut traquer le nombre de composantes connexes. En réalité, c'est le nombre de groupes d'homologie que l'on traque. En dimension 1, les deux notions coïncident comme expliqué plus tôt. En dimension 2, des groupes d'homologie d'ordre 1 apparaissent. Cependant, ils n'apparaissent pas dans notre cas donc on ne passe pas plus de temps dessus.

En dimension 1, on comprend aisément que cela permet de détecter la présence de minima locaux, ainsi que la profondeur du "puits" qu'ils forment. En dimension 2, cela permet aussi de détecter les points-selles. Si l'on revient à nos cartes d'erreur de la section 4.2.2, cela permet de vérifier que les cartes ont globalement une forme de puits centré sur le minimum.

Conclusion

Cette thèse s'est consacrée à la simulation de fluides viscoplastiques régis par une loi de Herschel-Bulkley ou Bingham. Le chapitre 3 s'est concentré sur la comparaison de différents algorithmes d'optimisation ayant pour objectif de résoudre les équations complètes dans le cas d'un écoulement stationnaire. Une première partie a été dédiée à la comparaison entre ADMM (souvent appelé simplement Lagrangien augmenté dans la communauté viscoplastique) et Dual FISTA, qui a été plus récemment introduit dans le cadre viscoplastique. Bien que les formulations mathématiques de ces algorithmes soient presque identiques dans le cas Bingham, le cas Herschel-Bulkley crée des différences notables. ADMM nécessite la résolution d'un sous-problème non-linéaire mais local, alors que Dual FISTA nécessite l'introduction d'une sur-boucle de contrôle des paramètres de la méthode.

Des tests numériques comparatifs de ces deux méthodes ont par la suite été effectués dans le cadre d'une discrétisation différences finies sur une grille décalée de type MAC, pour le cas Bingham. Une géométrie de type expansion-contraction a été choisie, dans la mesure où elle crée des topologies de plugs complexes et permet donc de tester les limites des algorithmes. Ce cadre plus complexe a permis de révéler d'importantes instabilités de la méthodologie Dual FISTA, au point que dans de nombreux cas, il est impossible d'identifier des limites de plug à partir des résultats de simulation. L'ajout d'une procédure de *restart* a cependant permis de drastiquement réduire ce phénomène. Bien que l'on observe toujours quelques cas de non-convergence de la méthode, ceux-ci sont minoritaires. L'algorithme qui en résulte est alors significativement plus rapide que l'algorithme ADMM, les simulations étant bien souvent dix fois plus rapides avec Dual FISTA. Un travail théorique supplémentaire se doit d'être effectué pour comprendre l'origine de ces instabilités et les empêcher de manière plus systématique. Une étude de la convergence des algorithmes sous raffinement de maillage a aussi été effectuée. On retrouve la convergence à l'ordre 2 attendue pour les maillages MAC, mais on a aussi pu constater que la stabilité de Dual FISTA n'était pas améliorée par le raffinement. Il convient de noter que ADMM n'est pas totalement exempt des phénomènes d'instabilité. En effet, sans la discrétisation, le résidu devrait observer une convergence monotone au cours des itérations, ce qui n'est pas le cas (ce phénomène a déjà pu être mis en évidence dans la littérature). Cependant, dans le cas d'ADMM, ces oscillations du résidu n'ont jamais, au cours de toutes les simulations, empêché la convergence de l'algorithme, qui montre des plugs nettement définis et une convergence du résidu au zéro machine. Cette analyse est complexifiée par le fait que le résidu ne permet pas de rendre compte de manière rigoureuse de la qualité de la convergence d'une simulation (voir [240]). Malgré l'existence du *dual gap*, introduit dans [240] au sein de la littérature viscoplastique, cet outil très intéressant nécessite d'utiliser une discrétisation compatible, à savoir certains éléments finis spécifiques. Il n'est donc pas trivialement applicable au cadre différences finies. Le résidu reste donc le meilleur indicateur quantitatif de convergence à l'heure actuelle. Pour la même raison, la mise en œuvre pratique de Dual FISTA dans le cas Herschel-Bulkley en différences finies pose question puisque l'algorithme nécessite l'évaluation de la fonctionnelle duale, qui devrait à nouveau nécessiter une discrétisation compatible pour ne pas renvoyer $-\infty$. Il est donc primordial

que de futurs travaux identifient les causes exactes des instabilités de FISTA pour s'assurer que ces erreurs sur l'évaluation de la fonction duale n'entacheront pas davantage la convergence de la méthode.

Par la suite, la méthodologie introduite par Bleyer [49] basée sur les cônes de second ordre a été étudiée. La dérivation explicite de la méthode primale-duale complète proposée par Bleyer a été effectuée et étendue pour la première fois dans le cadre Herschel-Bulkley, avec des ouvertures sur des variantes de la formulation. Il en ressort que la méthode est significativement plus complexe à implémenter que les méthodes de dualité, notamment du fait de l'introduction de plusieurs variables intermédiaires, qui de plus nécessitent une réflexion supplémentaire pour leur discrétisation. Les tests numériques en différences finies ont été infructueux. La méthode échoue complètement à ne serait-ce que commencer à approcher la solution du problème, se retrouvant bloquée par le pas imposé par les contraintes, qui se retrouve proche de 0 dans certaines régions de l'espace. L'introduction d'une version locale du pas pourrait être envisagée pour relaxer la méthode et lui permettre de commencer à approcher la solution, quitte à reprendre un pas global en fin d'algorithme. Toutefois, une telle procédure demande de grandes précautions pour ne pas risquer de violer les contraintes de cône à cause des opérations non-locales qui suivent. Il reste cependant difficile d'identifier précisément ce qui cause ce problème, tant les blocages semblent aléatoires. L'hypothèse la plus probable est celle de la discrétisation, qui demande d'utiliser des procédés de moyennes qui introduisent des erreurs dans les étapes intermédiaires. Pour vérifier cela, une implémentation en éléments finis a été effectuée, avec de bons résultats, notamment grâce à l'introduction d'une procédure de raffinement de maillage permettant d'améliorer significativement la qualité des délimitations de plugs. Toutefois, les résultats en différences finies ainsi que la nécessité d'utiliser les éléments *quadrature* proposés par FEniCSx engagent à rester prudent sur l'utilisation de cette méthode. Des tests plus poussés, notamment des comparaisons minutieuses entre les limites de plug obtenues par cette méthode et par la méthode ADMM, demandent à être effectués, notamment dans le cas Herschel-Bulkley. Il est aussi nécessaire de procéder à une analyse théorique de l'écriture différences finies de la méthode pour comprendre les instabilités observées.

Plus globalement, le chapitre 2 fournit des pistes pour réduire le décalage entre les méthodes utilisées à l'heure actuelle par la communauté viscoplastique et les méthodes les plus pointues développées pour optimisation convexe non-lisse. En effet, il semble que la littérature viscoplastique soit à l'heure actuelle dominée par l'utilisation du Lagrangien augmenté, à laquelle commence à s'ajouter l'algorithme FISTA. Cependant, la littérature sur l'optimisation convexe non-lisse offre aujourd'hui de nombreuses autres possibilités. En particulier, la review de Condat et al. [68] propose de nombreux algorithmes proximaux. Un travail important constituerait à vérifier si un de ces algorithmes pourrait permettre de s'affranchir de la nécessité de résoudre un sous-problème de Stokes, par exemple au profit d'un problème local voire idéalement un algorithme ne présentant que des mises à jour explicites (du moins dans le cas Bingham). D'autres pistes se trouvent du côté des méthodes de Newton, avec par exemple les dernières méthodes de régularisation cubique expliquées notamment par Nesterov [177], qui pourraient venir compléter la méthodologie proposée par Saramito [221] basée sur une méthode de Newton. Une large direction de recherche est aussi donnée par l'étude d'algorithmes randomisés, qui pourraient permettre d'effectuer des résolutions sur des maillages très raffinés pour un coût qui reste réduit, on renvoie aux références au début du chapitre 2 pour plus de détails sur ce type de méthodes.

Pour finir, le chapitre 4 s'est concentré sur un modèle de lubrification viscoplastique dans une cavité fermée. Les simplifications dues aux approximations de lubrification ont permis de se ramener à une EDP instationnaire avec une seule dimension d'espace. Après l'introduction d'un solveur numérique adapté, on s'est intéressé à la construction d'un métamodèle afin d'approcher la solution de l'EDP plus rapidement. L'association d'un développement en polynômes de chaos

et d'une analyse en composantes principales a permis d'accélérer significativement la résolution tout en permettant de conserver une grande précision sur la solution dans la majeure partie de l'espace des paramètres. Ce métamodèle a ensuite été utilisé pour réaliser l'estimation de paramètres physiques d'écoulements viscoplastiques, d'abord à partir de données synthétiques, avec un grand succès, puis à partir d'expériences physiques. Ces dernières ont fourni des résultats satisfaisants, bien que plus mitigés sur certaines expériences. Il convient cependant de rappeler que ces expériences ont été réalisées sur une gamme de paramètres à la limite du domaine sur lequel a été construit le métamodèle. Aussi, on s'attend à ce que dans le cadre du CPB, pour lequel la méthodologie a été élaborée, le couple métamodèle+problème inverse fournisse des résultats de meilleure qualité. Un effort a été réalisé pour produire un code clair et documenté afin qu'il puisse être réutilisé par des praticiens et éventuellement adapté, par exemple pour d'autres gammes de paramètres. De plus, le code ainsi que les bases de données calculées pendant ce travail ont été librement mis à disposition sur la plateforme Zenodo. De futurs travaux pourraient essayer d'utiliser d'autres techniques que le développement en polynômes de chaos pour créer le métamodèle, telles que des méthodes de machine learning. Il y a aussi de nombreuses directions de recherche sur le problème direct, telles que l'étude de l'influence de l'angle d'inclinaison de la cavité, ou encore la modification de la topographie de la cavité, par exemple pour prendre en compte des dépôts préexistants (typiques d'un remplissage multicouches).

Bibliographie

[1] F. Alizadeh and D. Goldfarb. Second-order cone programming. *Mathematical Programming*, 95(1) :3–51, Jan. 2003. doi:10.1007/s10107-002-0339-5. 15, 17, 41, 95, 96, 100

[2] G. Allaire. *Analyse numérique et optimisation*. Les Éditions de l'École Polytechnique, 2 edition, 2012. 17, 18

[3] M. S. Alnaes, A. Logg, K. B. Ølgaard, M. E. Rognes, and G. N. Wells. Unified form language : A domain-specific language for weak formulations of partial differential equations. *ACM Transactions on Mathematical Software*, 40, 2014. doi:10.1145/2566630. 109

[4] P. Amestoy, A. Buttari, J.-Y. L'Excellent, and T. Mary. Performance and Scalability of the Block Low-Rank Multifrontal Factorization on Multicore Architectures. *ACM Transactions on Mathematical Software*, 45 :2 :1–2 :26, 2019. 58, 70, 108

[5] P. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1) :15–41, 2001. 58, 70, 108

[6] E. Andersen, C. Roos, and T. Terlaky. On implementing a primal-dual interior-point method for conic quadratic optimization. *Mathematical Programming*, 95(2) :249–277, Feb. 2003. doi:10.1007/s10107-002-0349-3. 100

[7] A. J. Apostolidis and A. N. Beris. Modeling of the blood rheology in steady-state shear flows. *Journal of Rheology*, 58(3) :607–633, May 2014. doi:10.1122/1.4866296. 6

[8] M. ApS. *The Mosek optimization software*, 2022. 15

[9] K. J. Arrow, L. Hurwicz, and H. Uzawa. *Studies in linear and non-linear programming*. Stanford University Press, 1958. 40

[10] H. Attouch. Convergence de fonctions convexes, de sous-différentiels et semi-groupes. *Comptes rendus hebdomadaires des séances de l'Académie des sciences de Paris*, 284 :539–542, 1977. 26

[11] H. Attouch and J. Peypouquet. The Rate of Convergence of Nesterov's Accelerated Forward-Backward Method is Actually Faster Than $1/k^2$. *SIAM Journal on Optimization*, 26(3) :1824–1834, Jan. 2016. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/15M1046095. 49

[12] S. Azam and Q. Li. Tailings Dam Failures : A Review of the Last One Hundred Years. *Geotechnical News (Canada USA Mexico)*, 28(4) :50–53, Dec. 2010. Section : Waste Geotechnics. URL : http://ksmproject.com/wp-content/uploads/2017/08/Tailings-Dam-Failures-Last-100-years-Azam2010.pdf. 125

[13] G. Balarac, F. Basile, P. Bénard, F. Bordeu, J.-B. Chapelier, L. Cirrottola, G. Caumon, C. Dapogny, P. Frey, A. Froehly, G. Ghigliotti, R. Laraufie, G. Lartigue, C. Legentil, R. Mercier, V. Moureau, C. Nardoni, S. Pertant, and M. Zakari. Tetrahedral remeshing in the context of large-scale numerical simulation and high performance computing. *MathematicS In Action*, 11(1) :129–164, 2022. doi:10.5802/msia.22. 114

[14] S. Balay, S. Abhyankar, M. F. Adams, S. Benson, J. Brown, P. Brune, K. Buschelman, E. Constantinescu, L. Dalcin, A. Dener, V. Eijkhout, J. Faibussowitsch, W. D. Gropp, V. Hapla, T. Isaac, P. Jolivet, D. Karpeev, D. Kaushik, M. G. Knepley, F. Kong, S. Kruger, D. A. May, L. C. McInnes, R. T. Mills, L. Mitchell, T. Munson, J. E. Roman, K. Rupp, P. Sanan, J. Sarich, B. F. Smith, S. Zampini, H. Zhang, H. Zhang, and J. Zhang. PETSc/TAO users manual. Technical Report ANL-21/39 - Revision 3.21, Argonne National Laboratory, 2024. doi:10.2172/2205494. 70, 108

[15] S. Balay, S. Abhyankar, M. F. Adams, S. Benson, J. Brown, P. Brune, K. Buschelman, E. M. Constantinescu, L. Dalcin, A. Dener, V. Eijkhout, J. Faibussowitsch, W. D. Gropp, V. Hapla, T. Isaac, P. Jolivet, D. Karpeev, D. Kaushik, M. G. Knepley, F. Kong, S. Kruger, D. A. May, L. C. McInnes, R. T. Mills, L. Mitchell, T. Munson, J. E. Roman, K. Rupp, P. Sanan, J. Sarich, B. F. Smith, S. Zampini, H. Zhang, H. Zhang, and J. Zhang. PETSc Web page. https://petsc.org/, 2024. URL : https://petsc.org/. 70, 108

[16] 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. 70, 108

[17] N. Balmforth, R. Craster, D. Hewitt, S. Hormozi, and A. Maleki. Viscoplastic boundary layers. *Journal of Fluid Mechanics*, 813 :929–954, 2017. doi:10.1017/jfm.2016.878. 73

[18] N. J. Balmforth and R. V. Craster. A consistent thin-layer theory for Bingham plastics. *Journal of Non-Newtonian Fluid Mechanics*, 84(1) :65–81, July 1999. doi:10.1016/S0377-0257(98)00133-5. 12

[19] N. J. Balmforth, R. V. Craster, and D. R. Hewitt. Building on Oldroyd's viscoplastic legacy : Perspectives and new developments. *Journal of Non-Newtonian Fluid Mechanics*, 294 :104580, Aug. 2021. doi:10.1016/j.jnnfm.2021.104580. 7, 14

[20] N. J. Balmforth, R. V. Craster, A. C. Rust, and R. Sassi. Viscoplastic flow over an inclined surface. *Journal of Non-Newtonian Fluid Mechanics*, 139(1-2) :103–127, 2006. 12, 126, 128

[21] N. J. Balmforth, I. A. Frigaard, and G. Ovarlez. Yielding to Stress : Recent Developments in Viscoplastic Fluid Mechanics. *Annual Review of Fluid Mechanics*, 46(1) :121–146, Jan. 2014. Publisher : Annual Reviews. doi:10.1146/annurev-fluid-010313-141424. 1, 2, 14

[22] I. A. Baratta, J. P. Dean, J. S. Dokken, M. Habera, J. S. Hale, C. N. Richardson, M. E. Rognes, M. W. Scroggs, N. Sime, and G. N. Wells. DOLFINx : the next generation FEniCS problem solving environment. preprint, 2023. doi:10.5281/zenodo.10447666. 109

[23] G. Bareilles. *Optimisation non-lisse structurée : identification proximale, convergence locale rapide et applications*. PhD thesis, Université Grenoble Alpes, 2022. Thèse de doctorat dirigée par Malick, Jérôme et Iutzeler, Franck Mathématiques appliquées. 49

[24] G. Bareilles, F. Iutzeler, and J. Malick. Newton acceleration on manifolds identified by proximal gradient methods. *Mathematical Programming*, 200(1) :37–70, June 2023. doi:10.1007/s10107-022-01873-w. 49

[25] H. A. Barnes. The yield stress—a review or '$\pi\alpha\nu\tau\alpha\ \rho\varepsilon\iota$'—everything flows ? *Journal of Non-Newtonian Fluid Mechanics*, 81(1) :133–178, Feb. 1999. doi:10.1016/S0377-0257(98)00094-9. 3

[26] H. A. Barnes and K. Walters. The yield stress myth ? *Rheologica Acta*, 24(4) :323–326, July 1985. doi:10.1007/BF01333960. 3

[27] A. Barré de Saint-Venant. Théorie du mouvement non permanent des eaux, avec application aux crues des rivières et à l'introduction des marées dans leur lit. *C. R. Acad. Sci., Paris*, 73 :147–154, 1871. 12

[28] S. Bartels and M. Milicevic. Alternating direction method of multipliers with variable step sizes. *arXiv :1704.06069 [math]*, Apr. 2017. 48

[29] I. Basov and V. Shelukhin. Generalized solutions to the equations of compressible bingham flows. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 79(3) :185–192, 1999. doi:https://doi.org/10.1002/(SICI)1521-4001(199903)79:3<185::AID-ZAMM185>3.0.CO;2-N. 12

[30] I. V. Basov. Existence of a Rigid Core in the Flow of a Compressible Bingham Fluid under the Action of a Homogeneous Force. *Journal of Mathematical Fluid Mechanics*, 7(4) :515–528, Nov. 2005. doi:10.1007/s00021-005-0179-1. 12

[31] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press, 2000. doi:10.1017/CBO9780511800955. 14

[32] H. H. Bauschke and P. L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics. Springer, New York, NY, 2011. doi:10.1007/978-1-4419-9467-7. 18, 20, 49

[33] A. Beck. *First-Order Methods in Optimization*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Oct. 2017. doi:10.1137/1.9781611974997. 17, 18, 20, 27, 44, 53, 54

[34] A. Beck and M. Teboulle. Fast Gradient-Based Algorithms for Constrained Total Variation Image Denoising and Deblurring Problems. *IEEE Transactions on Image Processing*, 18(11) :2419–2434, Nov. 2009. doi:10.1109/TIP.2009.2028250. 50

[35] A. Beck and M. Teboulle. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*, 2(1) :183–202, Jan. 2009. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/080716542. 48, 49, 63

[36] A. Beck and M. Teboulle. A fast dual proximal gradient algorithm for convex minimization and applications. *Operations Research Letters*, 42(1) :1–6, Jan. 2014. doi:10.1016/j.orl.2013.10.007. 51

[37] A. Ben-Tal and A. Nemirovski. *Lectures on Modern Convex Optimization*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Jan. 2001. doi:10.1137/1.9780898718829. 41

[38] M. Bercovier and M. Engelman. A finite-element method for incompressible non-Newtonian flows. *Journal of Computational Physics*, 36(3) :313–326, July 1980. doi:10.1016/0021-9991(80)90163-1. 13

[39] C. Berger, D. Coulette, and P. Vigneaux. A metamodel for confined yield stress flows and parameters' estimation. *Rheologica Acta*, Feb. 2024. doi:10.1007/s00397-024-01436-0. 125

[40] C. Berger and P. Vigneaux. Supplementary code for "A metamodel for confined yield stress flows and parameters' estimation" https://doi.org/10.5281/zenodo.8377205. *Zenodo*, page 8377205, Oct. 2023. doi:10.5281/zenodo.8377205. 151

[41] D. Bertsekas. On the method of multipliers for convex programming. *IEEE Transactions on Automatic Control*, 20(3) :385–388, June 1975. doi:10.1109/TAC.1975.1100976. 41

[42] D. P. Bertsekas, editor. *Convex optimization theory*. Athena Scientific, Belmont, Mass, 2009. 18

[43] D. P. Bertsekas. *Convex optimization algorithms*. Athena scientific, Nashua, 2015. 17

[44] M. Bierlaire. *Optimization : Principles and Algorithms*. EPFL Press, Lausanne, 2nd edition, 2018. 17, 30, 36, 37, 147

[45] E. C. Bingham. An investigation of the laws of plastic flow. *Bulletin of the Bureau of Standards*, 13(2) :309–353, 1917. doi:https://hdl.handle.net/2027/mdp.39015086559054. 5

[46] E. C. Bingham. *Fluidity and Plasticity*. McGraw-Hill Book Company, Incorporated, 1922. 5

[47] R. B. Bird, G. C. Dai, and B. J. Yarusso. The Rheology and Flow of Viscoplastic Materials. *Reviews in Chemical Engineering*, 1(1) :1–70, Mar. 1983. Publisher : De Gruyter. doi:10.1515/revce-1983-0102. 8

[48] G. Blatman and B. Sudret. Sparse polynomial chaos expansions of vector-valued response quantities. In *Safety, Reliability, Risk and Life-cycle Performance of Structures and Infrastructures*, pages 3245–3252. CRC Press/Balkema, 2013. doi:10.3929/ethz-a-010057918. 126, 142

[49] J. Bleyer. Advances in the simulation of viscoplastic fluid flows using interior-point methods. *Computer Methods in Applied Mechanics and Engineering*, 330 :368–394, 2018. doi:https://doi.org/10.1016/j.cma.2017.11.006. 15, 95, 96, 97, 98, 101, 104, 109, 112, 119, 160

[50] J. Bleyer, M. Maillard, P. de Buhan, and P. Coussot. Efficient numerical computations of yield stress fluid flows using second-order cone programming. *Computer Methods in Applied Mechanics and Engineering*, 283 :599–614, Jan. 2015. doi:10.1016/j.cma.2014.10.008. 14, 95, 109

[51] P. T. Boggs and J. W. Tolle. Sequential quadratic programming. *Acta Numerica*, 4 :1–51, 1995. doi:10.1017/S0962492900002518. 38

[52] D. Bonn, M. M. Denn, L. Berthier, T. Divoux, and S. Manneville. Yield stress materials in soft condensed matter. *Reviews of Modern Physics*, 89(3) :035005, Aug. 2017. Publisher : American Physical Society. doi:10.1103/RevModPhys.89.035005. 3

[53] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine Learning*, 3(1) :1–122, Jan. 2011. doi:10.1561/2200000016. 47, 57, 72

[54] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. 17, 18, 24, 32, 35, 40, 41, 42, 43

[55] R. E. J. Bruck. An iterative solution of a variational inequality for certain monotone operators in Hilbert space. *Bulletin of the American Mathematical Society*, 81(5) :890–892, Sept. 1975. Publisher : American Mathematical Society. 48

[56] M. Bulícek, P. Gwiazda, J. Málek, K. R. Rajagopal, and A. Swierczewska-Gwiazda. On flows of fluid described by an implicit constitutive equation characterized by a maximal monotone graph. *London Math. Soc. Lecture Note, Cambridge Univ. Press, Cambridge*, 2012. 12

[57] T. Burghelea, K. Wielage-Burchard, I. A. Frigaard, D. M. Martinez, and J. J. Feng. A novel shear flow instability triggered by a chemical reaction in the absence of inertia. *Physics of Fluids*, 19(8) :083102, 2007. doi:10.1063/1.2759190. 1

[58] G. R. Burgos, A. N. Alexandrou, and V. Entov. On the determination of yield surfaces in Herschel–Bulkley fluids. *Journal of Rheology*, 43(3) :463–483, May 1999. doi:10.1122/1.550992. 13

[59] J. C. P. Bus and T. J. Dekker. Two efficient algorithms with guaranteed convergence for finding a zero of a function. *ACM Trans. Math. Softw.*, 1(4) :330–345, dec 1975. doi:10.1145/355656.355659. 130

[60] N. Casson. A flow equation for pigment-oil suspensions of the printing ink type. *Rheology of disperse systems*, 84, 1959. 6

[61] A. L. Cauchy. Méthode générale pour la résolution des systèmes d'équations simultanées. *Comptes rendus hebdomadaires des séances de l'Académie des sciences de Paris*, 25 :536–538, 1847. 31

[62] R. Chalayer. *Méthodes de projection pour des écoulements à seuil, incompressibles et à densité variable*. These de doctorat, Université Clermont Auvergne, Sept. 2019. 15

[63] A. Chambolle and C. Dossal. On the Convergence of the Iterates of the "Fast Iterative Shrinkage Thresholding Algorithm". *Journal of Optimization Theory and Applications*, 166(3) :968–982, Sept. 2015. doi:10.1007/s10957-015-0746-4. 49

[64] A. Chambolle and T. Pock. A First-Order Primal-Dual Algorithm for Convex Problems with Applications to Imaging. *Journal of Mathematical Imaging and Vision*, 40(1) :120–145, May 2011. doi:10.1007/s10851-010-0251-1. 46

[65] A. Chambolle and T. Pock. An introduction to continuous optimization for imaging. *Acta Numerica*, 25 :161–319, May 2016. Publisher : Cambridge University Press. doi:10.1017/S096249291600009X. 17, 18, 34, 44, 49

[66] L. Chupin and T. Dubois. A bi-projection method for Bingham type flows. *Computers & Mathematics with Applications*, 72(5) :1263–1286, Sept. 2016. doi:10.1016/j.camwa.2016.06.026. 11, 15

[67] L. Chupin, T. Dubois, M. Phan, and O. Roche. Pressure-dependent threshold in a granular flow : Numerical modeling and experimental validation. *Journal of Non-Newtonian Fluid Mechanics*, 291 :104529, May 2021. doi:10.1016/j.jnnfm.2021.104529. 15

[68] L. Condat, D. Kitahara, A. Contreras, and A. Hirabayashi. Proximal Splitting Algorithms for Convex Optimization : A Tour of Recent Advances, with New Twists. *SIAM Review*, 65(2) :375–435, May 2023. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/20M1379344. 44, 46, 160

[69] L. Condat and P. Richtárik. Randprox : Primal-dual optimization algorithms with randomized proximal updates. In *The Eleventh International Conference on Learning Representations*, 2023. 17

[70] R. Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the American Mathematical Society*, 49 :1–23, 1943. 37

[71] P. Coussot. Yield stress fluid flows : A review of experimental data. *Journal of Non-Newtonian Fluid Mechanics*, 211 :31–49, Sept. 2014. doi:10.1016/j.jnnfm.2014.05.006. 1

[72] P. Coussot. Bingham's heritage. *Rheologica Acta*, 56(3) :163–176, Mar. 2017. doi:10.1007/s00397-016-0983-y. 3, 14

[73] P. Coussot, S. Proust, and C. Ancey. Rheological interpretation of deposits of yield stress fluids. *Journal of Non-Newtonian Fluid Mechanics*, 66(1) :55–70, Sept. 1996. doi:10.1016/0377-0257(96)01474-7. 12

[74] P. Coussot, S. Proust, and C. Ancey. Rheological interpretation of deposits of yield stress fluids. *Journal of Non-Newtonian Fluid Mechanics*, 66(1) :55–70, Sept. 1996. doi:10.1016/0377-0257(96)01474-7. 126

[75] P. Coussot and S. A. Rogers. Oldroyd's model and the foundation of modern rheology of yield stress fluids. *Journal of Non-Newtonian Fluid Mechanics*, 295 :104604, Sept. 2021. doi:10.1016/j.jnnfm.2021.104604. 7

[76] G. Dai and R. Byron Bird. Radial flow of a Bingham fluid between two fixed circular disks. *Journal of Non-Newtonian Fluid Mechanics*, 8(3) :349–355, Jan. 1981. doi:10.1016/0377-0257(81)80031-6. 12

[77] C. Dapogny, C. Dobrzynski, and P. Frey. Three-dimensional adaptive domain remeshing, implicit domain meshing, and applications to free and moving boundary problems. *Journal of Computational Physics*, 262 :358–378, 2014. doi:https://doi.org/10.1016/j.jcp.2014.01.005. 114

[78] W. C. Davidon. Variable metric method for minimization. Technical report ANL-5990, Argonne National Laboratory, 5 1959. doi:10.2172/4252678. 36

[79] W. C. Davidon. Variable Metric Method for Minimization. *SIAM Journal on Optimization*, 1(1) :1–17, Feb. 1991. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/0801001. 36

[80] D. De Kee. Yield stress measurement techniques : A review. *Physics of Fluids*, 33(11) :111301, Nov. 2021. doi:10.1063/5.0070209. 3

[81] V. F. Dem'yanov and A. M. Rubinov. Approximate methods of solving extremal problems. *New York : American Elsevier*, 1970. 39

[82] E. den Boef and D. den Hertog. Efficient Line Search Methods for Convex Functions. *SIAM Journal on Optimization*, 18(1) :338–363, Jan. 2007. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/04061115X. 31

[83] Y. Dimakopoulos, M. Pavlidis, and J. Tsamopoulos. Steady bubble rise in Herschel–Bulkley fluids and comparison of predictions via the Augmented Lagrangian Method with those via the Papanastasiou model. *Journal of Non-Newtonian Fluid Mechanics*, 200 :34–51, Oct. 2013. doi:10.1016/j.jnnfm.2012.10.012. 14

[84] C. Dobrzynski and P. Frey. Anisotropic delaunay mesh adaptation for unsteady simulations. In R. V. Garimella, editor, *Proceedings of the 17th International Meshing Roundtable*, pages 177–194, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 114

[85] J. Douglas and H. H. J. Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Transactions of the American Mathematical Society*, 82 :421–439, 1956. 47

[86] J. C. Dunn. Rates of Convergence for Conditional Gradient Algorithms Near Singular and Nonsingular Extremals. *SIAM Journal on Control and Optimization*, 17(2) :187–211, Mar. 1979. doi:10.1137/0317015. 39

[87] G. Duvaut and J.-L. Lions. *Les inéquations en mécanique et en physique*. Dunod, Paris, France, 1972. ISSN : 0564-156X. 3, 10, 11, 12, 14

[88] H. Edelsbrunner and J. Harer. *Computational Topology : An Introduction*, volume 69 of *Miscellaneous Books*. American Mathematical Society, 2010. doi:10.1090/mbk/069. 154, 156

[89] J. Feinberg and H. P. Langtangen. Chaospy : An open source tool for designing methods of uncertainty quantification. *Journal of Computational Science*, 11 :46–57, Nov. 2015. doi:10.1016/j.jocs.2015.08.008. 142

[90] E. D. Fernández-Nieto, J. M. Gallardo, and P. Vigneaux. Efficient numerical schemes for viscoplastic avalanches. Part 1 : the 1D case. *Journal of Computational Physics*, 264 :55, Jan. 2014. URL : https://hal.science/hal-01011373, doi:10.1016/j.jcp.2014.01.026. 1

[91] E. D. Fernández-Nieto, J. M. Gallardo, and P. Vigneaux. Efficient numerical schemes for viscoplastic avalanches. Part 2 : The 2D case. *Journal of Computational Physics*, 353 :460–490, 2018. URL : https://hal.science/hal-01593148, doi:https://doi.org/10.1016/j.jcp.2017.09.054. 1

[92] A. V. Fiacco and G. McCormick. *Sequential Unconstrained Minimization Techniques*. John Wiley & Sons, 1968. 38, 42

[93] R. Fletcher and M. J. D. Powell. A Rapidly Convergent Descent Method for Minimization. *The Computer Journal*, 6(2) :163–168, Aug. 1963. doi:10.1093/comjnl/6.2.163. 36

[94] R. Fletcher and C. M. Reeves. Function minimization by conjugate gradients. *The Computer Journal*, 7(2) :149–154, 01 1964. doi:10.1093/comjnl/7.2.149. 32

[95] M. I. Florea and S. A. Vorobyov. A Generalized Accelerated Composite Gradient Method : Uniting Nesterov's Fast Gradient Method and FISTA. *IEEE Transactions on Signal Processing*, 68 :3033–3048, 2020. Conference Name : IEEE Transactions on Signal Processing. doi:10.1109/TSP.2020.2988614. 50

[96] M. Fortin. *Calcul numérique des écoulements des fluides de Bingham et des fluides newtoniens incompressibles par la méthode des éléments finis*. These de doctorat, Université Paris VI, 1972. 13

[97] M. Fortin and R. Glowinski. *Méthodes de Lagrangien augmenté : Application à la résolution numérique de problèmes aux limites*. Dunod, 1982. 14

[98] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2) :95–110, 1956. doi:https://doi.org/10.1002/nav.3800030109. 39

[99] P. Freydier, G. Chambon, and M. Naaim. Experimental characterization of velocity fields within the front of viscoplastic surges down an incline. *Journal of Non-Newtonian Fluid Mechanics*, 240 :56–69, Feb. 2017. doi:10.1016/j.jnnfm.2017.01.002. 3

[100] I. Frigaard. Simple yield stress fluids. *Current Opinion in Colloid & Interface Science*, 43 :80–93, Oct. 2019. doi:10.1016/j.cocis.2019.03.002. 3, 14

[101] I. A. Frigaard and C. Nouar. On the usage of viscosity regularisation methods for viscoplastic fluid flow computation. *Journal of Non-Newtonian Fluid Mechanics*, 127(1) :1–26, Apr. 2005. doi:10.1016/j.jnnfm.2005.01.003. 13

[102] I. A. Frigaard, K. G. Paso, and P. R. de Souza Mendes. Bingham's model in the oil and gas industry. *Rheologica Acta*, 56(3) :259–282, Mar. 2017. doi:10.1007/s00397-017-0999-y. 1, 6

[103] K. R. Frisch. The logarithmic potential method of convex programming. Technical report, University Institute of Economics, 1955. doi:10.2172/4252678. 38

[104] M. Fuchs and G. Seregin. *Variational Methods for Problems from Plasticity Theory and for Generalized Newtonian Fluids*, volume 1749 of *Lecture Notes in Mathematics*. Springer, Berlin, Heidelberg, 2000. doi:10.1007/BFb0103751. 12

[105] D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite element approximation. *Computers & Mathematics With Applications*, 2 :17–40, 1976. 47

[106] C. Gallery, S. Bourge, and G. Agoda-Tandjawa. Flow behaviors of multiple molten chocolate matrices : Appropriate curve fitting models and impact of different types of surfactants. *Journal of Food Engineering*, 363 :111780, Feb. 2024. doi:10.1016/j.jfoodeng.2023.111780. 6

[107] H. Gao, L. Qing, G. Ma, D. Zhang, and C. Wei. Numerical investigation into effects of rheological properties on grout flow in rock fracture using Herschel-Bulkley model. *Engineering Geology*, 329 :107402, Feb. 2024. doi:10.1016/j.enggeo.2023.107402. 6, 13

[108] O. Garcia-Cabrejo and A. Valocchi. Global sensitivity analysis for multivariate output using polynomial chaos expansion. *Reliability Engineering and System Safety*, 126 :25–36, 2014. doi:10.1016/j.ress.2014.01.005. 141

[109] C. Gauss. *Theory of the Combination of Observations Least Subject to Errors : Part One, Part Two, Supplement. Translated from original 1820 manuscript by G.W. Stewart.* Classics in Applied Mathematics. Society for Industrial and Applied Mathematics, 1995. 34

[110] R. Glowinski. Sur l'ecoulement d'un fluide de bingham dans une conduite cylindrique. *Journal de mécanique*, 13(4) :601–621, 1974. Copyright : Copyright 2017 Elsevier B.V., All rights reserved. 10

[111] R. Glowinski. Finite element methods for incompressible viscous flow. In *Numerical Methods for Fluids (Part 3)*, volume 9 of *Handbook of Numerical Analysis*. Elsevier, 2003. doi:https://doi.org/10.1016/S1570-8659(03)09003-3. 58

[112] R. Glowinski and M. O. Bristeau. Finite element analysis of the unsteady flow of a visco-plastic fluide in a cylindrical pipe. *Finite Element Methods in Flow Problems*, pages 471–488, 1974. 10

[113] R. Glowinski, J.-L. Lions, and R. Trémolières. *Analyse numérique des inéquations variationnelles (Tomes 1 et 2)*. Dunod, Paris, France, 1976. 10, 11, 13, 14, 15, 56

[114] R. Glowinski and A. Marroco. Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de Dirichlet non linéaires. *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, 9(R2) :41–76, 1975. 47

[115] R. Glowinski and P. L. Tallec. *Augmented Lagrangian and Operator Splitting Methods in Nonlinear Mechanics*. Society for Industrial and Applied Mathematics, 1989. 57

[116] R. Glowinski and A. Wachs. On the Numerical Simulation of Viscoplastic Fluid Flow. In R. Glowinski and J. Xu, editors, *Handbook of Numerical Analysis*, volume 16 of *Numerical Methods for Non-Newtonian Fluids*, pages 483–717. Elsevier, Jan. 2011. doi:10.1016/B978-0-444-53047-9.00006-X. 13, 14

[117] A. A. Goldstein. Convex programming in hilbert space. *Bulletin of the American Mathematical Society*, 70 :709–710, 1964. doi:10.1090/S0002-9904-1964-11178-2. 39

[118] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk. Fast Alternating Direction Optimization Methods. *SIAM Journal on Imaging Sciences*, Aug. 2014. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/120896219. 48, 57

[119] E. G. Gol'stejn and N. V. Tret'yakov. Augmented lagrangians. *Ekonom. i Matemat. Metody*, 10(3) :568–91, 1974. 41

[120] O. Güler. New proximal point algorithms for convex minimization. *SIAM Journal on Optimization*, 2(4) :649–664, 1992. doi:10.1137/0802032. 33

[121] E. Guyon, J.-P. Hulin, and L. Petit. *Hydrodynamique Physique*. EDP Sciences, Jan. 2001. ISBN : 2868835023, 9782868835024. 3

[122] P. C. Haarhoff and J. D. Buys. A new method for the optimization of a nonlinear function subject to nonlinear constraints. *The Computer Journal*, 13(2) :178–184, 01 1970. doi:10.1093/comjnl/13.2.178. 41

[123] M. Hadigol and A. Doostan. Least squares polynomial chaos expansion : A review of sampling strategies. *Computer Methods in Applied Mechanics and Engineering*, 332 :382–407, 2018. doi:10.1016/j.cma.2017.12.019. 140, 143

[124] F. H. Harlow and J. E. Welch. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *The Physics of Fluids*, 8(12) :2182–2189, 12 1965. doi:10.1063/1.1761178. 65

[125] L. Hawchar, C.-P. E. Soueidy, and F. Schoefs. Principal component analysis and polynomial chaos expansion for time-variant reliability problems. *Reliability Engineering and System Safety*, 167 :406–416, 2017. doi:10.1016/j.ress.2017.06.024. 126, 142

[126] F. Hecht. Bamg : bidimensional anisotropic mesh generator. Technical report, INRIA Rocquencourt, 1997. 113

[127] W. H. Herschel and R. Bulkley. Konsistenzmessungen von Gummi-Benzollösungen. *Kolloid-Zeitschrift*, 39(4) :291–300, Aug. 1926. doi:10.1007/BF01432034. 6

[128] M. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal of Research of the National Bureau of Standards*, 49(6) :409–436, Dec. 1952. doi:10.6028/jres.049.044. 32

[129] M. R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5) :303–320, Nov. 1969. doi:10.1007/BF00927673. 41

[130] A. J. Hogg and G. P. Matson. Slumps of viscoplastic fluids on slopes. *Journal of Non-Newtonian Fluid Mechanics*, 158(1) :101–112, May 2009. doi:10.1016/j.jnnfm.2008.07.003. 128

[131] K. Hohenemser and W. Prager. Über die Ansätze der Mechanik isotroper Kontinua. *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, 12(4) :216–226, 1932. doi:10.1002/zamm.19320120403. 5

[132] X. Huang and M. H. Garcia. A Herschel-Bulkley model for mud flow down a slope. *Journal of Fluid Mechanics*, 374 :305–333, nov 1998. doi:10.1017/s0022112098002845. 126

[133] R. R. Huilgol. Variational inequalities in the flows of yield stress fluids including inertia : Theory and applications. *Physics of Fluids*, 14(3) :1269–1283, 03 2002. doi:10.1063/1.1448347. 14

[134] R. R. Huilgol, B. Mena, and J. M. Piau. Finite stopping time problems and rheometry of Bingham fluids. *Journal of Non-Newtonian Fluid Mechanics*, 102(1) :97–107, Jan. 2002. doi:10.1016/S0377-0257(01)00166-5. 14

[135] R. R. Huilgol and Z. You. Application of the augmented Lagrangian method to steady pipe flows of Bingham, Casson and Herschel–Bulkley fluids. *Journal of Non-Newtonian Fluid Mechanics*, 128(2) :126–143, July 2005. doi:10.1016/j.jnnfm.2005.04.004. 14

[136] A. A. Il'iushin. Deformation of a viscous-plastic plastic body (in Russian). *Uch. zap. MGU, Mekhanika*, 1940. 5, 9

[137] I. T. Jolliffe. *Principal Component Analysis*. Springer Series in Statistics. Springer, 2 edition, 2002. doi:10.1007/b98835. 141

[138] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4) :373–395, Dec. 1984. doi:10.1007/BF02579150. 38, 41

[139] E. A. Keller. *Introduction to environmental geology*. Pearson Prentice Hall, 4th edition, 2008. 125

[140] J. E. Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4) :703–712, 1960. doi:10.1137/0108053. 39

[141] T. H. Kjeldsen. A Contextualized Historical Analysis of the Kuhn–Tucker Theorem in Nonlinear Programming : The Impact of World War II. *Historia Mathematica*, 27(4) :331–361, Nov. 2000. doi:10.1006/hmat.2000.2289. 24

[142] H. W. Kuhn. Nonlinear programming : a historical view. *Non Linear Programming. SIAM-AMS Proc.*, pages 1–26, jun 1976. 24

[143] H. W. Kuhn and A. W. Tucker. Nonlinear programming. *Proceedings of the Second Berkeley Symposium on mathematical Statistics and Probability*, pages 481–492, 1950. 24

[144] D. Landriault. They Said "It Will Never Work" - 25 Years of Paste Backfill 1981 - 2006. In *Paste 2006 : Ninth International Seminar on Paste and Thickened Tailings, 2006 3-7 April. R.J. Jewell, S. Lawson, P. Newman (eds)*, pages 277–292, Limerick, Ireland, Apr. 2006. Australian Centre for Geomechanics. URL : https://papers.acg.uwa.edu.au/p/663_24_Landriault/, doi:10.36487/ACG_repo/663_24. 125

[145] O. P. Le Maître and O. M. Knio. *Spectral Methods for Uncertainty Quantification, With Applications to Computational Fluid Dynamics*. Scientific Computation. Springer Netherlands, 2010. doi:10.1007/978-90-481-3520-2. 138, 139, 140

[146] B. Lemaire. The proximal algorithm. *International Series of Numerical Mathematics*, pages 73–87, 1989. 45

[147] R. J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2004. 129

[148] A. Levin. On an algorithm for minimization of convex functions. *Dokl. Akad. Nauk SSSR*, 6(1) :286–90, 1965. 39

[149] E. S. Levitin and B. T. Polyak. Constrained minimization methods. *USSR Computational Mathematics and Mathematical Physics*, 6(5) :1–50, Jan. 1966. doi:10.1016/0041-5553(66)90114-5. 39

[150] J. Liang, J. Fadili, and G. Peyré. Activity Identification and Local Linear Convergence of Forward–Backward-type Methods. *SIAM Journal on Optimization*, 27(1) :408–437, Jan. 2017. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/16M106340X. 49

[151] J. Liang, T. Luo, and C.-B. Schönlieb. Improving "Fast Iterative Shrinkage-Thresholding Algorithm" : Faster, Smarter, and Greedier. *SIAM Journal on Scientific Computing*, 44(3) :A1069–A1091, June 2022. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/21M1395685. 49, 71

[152] G. G. Lipscomb and M. M. Denn. Flow of bingham fluids in complex geometries. *Journal of Non-Newtonian Fluid Mechanics*, 14 :337–346, Jan. 1984. doi:10.1016/0377-0257(84)80052-X. 12

[153] K. F. Liu and C. C. Mei. Slow spreading of a sheet of Bingham fluid on an inclined plane. *Journal of Fluid Mechanics*, 207 :505–529, Oct. 1989. doi:https://doi.org/10.1017/S0022112089002685. 126

[154] K. F. Liu and C. C. Mei. Approximate equations for the slow spreading of a thin sheet of Bingham plastic fluid. *Physics of Fluids A : Fluid Dynamics*, 2(1) :30–36, 01 1990. doi:10.1063/1.857821. 12

[155] Y. Liu, N. J. Balmforth, S. Hormozi, and D. R. Hewitt. Two–dimensional viscoplastic dambreaks. *Journal of Non-Newtonian Fluid Mechanics*, 238 :65–79, Dec. 2016. doi:10.1016/j.jnnfm.2016.05.008. 128

[156] L.-H. Luu, P. Philippe, and G. Chambon. Flow of a yield-stress fluid over a cavity : Experimental study of the solid–fluid interface. *Journal of Non-Newtonian Fluid Mechanics*, 245 :25–37, 2017. doi:https://doi.org/10.1016/j.jnnfm.2017.04.011. 90, 95

[157] A. Marly. *Analyse mathématique et numérique d'écoulements de fluides à seuil*. Thèse de doctorat, Université de Lyon, Sept. 2018. URL : https://theses.hal.science/tel-02145210. 16, 57, 58, 59, 60, 61, 62, 63, 69, 70, 71, 72, 73, 76, 86, 91, 92, 111

[158] A. Marly and P. Vigneaux. Augmented Lagrangian simulations study of yield-stress fluid flows in expansion-contraction and comparisons with physical experiments. *Journal of Non-Newtonian Fluid Mechanics*, 239 :35–52, Jan. 2017. URL : https://hal.science/hal-01432028, doi:10.1016/j.jnnfm.2016.12.004. 57, 59, 70, 71, 72, 73, 76, 111

[159] A. Marrel and B. Iooss. Probabilistic surrogate modeling by Gaussian process : A review on recent insights in estimation and validation. *Reliability Engineering & System Safety*, page 110094, Mar. 2024. doi:10.1016/j.ress.2024.110094. 151

[160] B. Martinet. Brève communication. Régularisation d'inéquations variationnelles par approximations successives. *Revue française d'informatique et de recherche opérationnelle. Série rouge*, 4(R3) :154–158, 1970. 45

[161] B. Martinet. Détermination approchée d'un point fixe d'une application pseudo-contractante. *Comptes rendus hebdomadaires des séances de l'Académie des sciences de Paris*, 274A :163–165, 1972. 45

[162] I. L. Medina Lino, M. Carrasco-Teja, and I. Frigaard. GPU computing of yield stress fluid flows in narrow gaps. *Theoretical and Computational Fluid Dynamics*, 37(5) :661–680, Oct. 2023. doi:10.1007/s00162-023-00674-x. 14

[163] S. Mehrotra. On the Implementation of a Primal-Dual Interior Point Method. *SIAM Journal on Optimization*, 2(4) :575–601, Nov. 1992. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/0802028. 103

[164] G. Migliorati and F. Nobile. Analysis of discrete least squares on multivariate polynomial spaces with evaluations at low-discrepancy point sets. *Journal of Complexity*, 31(4) :517–542, 2015. doi:10.1016/j.jco.2015.02.001. 140

[165] E. Mitsoulis and J. Tsamopoulos. Numerical simulations of complex yield-stress fluid flows. *Rheologica Acta*, 56(3) :231–258, Mar. 2017. doi:10.1007/s00397-016-0981-0. 13

[166] S. Mizani, X. He, and P. Simms. Application of lubrication theory to modeling stack geometry of high density mine tailings. *Journal of Non-Newtonian Fluid Mechanics*, 198 :59–70, 2013. doi:https://doi.org/10.1016/j.jnnfm.2013.03.002. 151

[167] J. J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes rendus hebdomadaires des séances de l'Académie des sciences de Paris*, 255 :2897–2899, 1962. 28

[168] J. J. Moreau. Proximité et dualité dans un espace hilbertien. *Bulletin de la Société Mathématique de France*, 93 :273–299, 1965. doi:10.24033/bsmf.1625. 28

[169] P. P. Mosolov and V. P. Miashikov. On stagnant flow regions of a viscous-plastic medium in pipes. *Journal of Applied Mathematics and Mechanics*, 30(4) :841–854, Jan. 1966. doi:10.1016/0021-8928(66)90035-9. 8, 10

[170] P. P. Mosolov and V. P. Miasnikov. Variational methods in the theory of the fluidity of a viscous-plastic medium. *Journal of Applied Mathematics and Mechanics*, 29(3) :545–577, Jan. 1965. doi:10.1016/0021-8928(65)90063-8. 8, 10

[171] P. P. Mosolov and V. P. Miasnikov. On qualitative singularities of the flow of a viscoplastic medium in pipes : PMM vol. 31, no. 3, 1967, pp. 581–585. *Journal of Applied Mathematics and Mechanics*, 31(3) :609–613, Jan. 1967. doi:10.1016/0021-8928(67)90055-X. 8, 10

[172] E. Muravleva, I. Oseledets, and D. Koroteev. Application of machine learning to viscoplastic flow modeling. *Physics of Fluids*, 30(10) :103102, Oct. 2018. doi:10.1063/1.5058127. 16

[173] E. A. Muravleva and M. A. Olshanskii. Two finite-difference schemes for calculation of bingham fluid flows in a cavity. *Russian Journal of Numerical Analysis and Mathematical Modelling*, 23(6) :615–634, 2008. doi:doi:10.1515/RJNAMM.2008.035. 16, 65

[174] L. Muravleva. Squeeze flow of Bingham, Casson and Herschel-Bulkley fluids with yield slip at the wall by accelerated augmented Lagrangian method. *Journal of Non-Newtonian Fluid Mechanics*, 282 :104320, Aug. 2020. doi:10.1016/j.jnnfm.2020.104320. 14

[175] J. A. Nelder and R. Mead. A Simplex Method for Function Minimization. *The Computer Journal*, 7(4) :308–313, 01 1965. doi:10.1093/comjnl/7.4.308. 147

[176] A. S. Nemirovsky and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley-Interscience Series in discrete Mathematics. John Wiley, Chichester, 1983. OCLC : 456408871. 39

[177] Y. Nesterov. *Lectures on Convex Optimization*, volume 137 of *Springer Optimization and Its Applications*. Springer International Publishing, Cham, 2018. doi:10.1007/978-3-319-91578-4. 17, 18, 29, 32, 34, 35, 39, 41, 61, 160

[178] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. Studies in Applied and Numerical Mathematics. Society for Industrial and Applied Mathematics, Jan. 1994. doi:10.1137/1.9781611970791. 41

[179] Y. E. Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. *Dokl. Akad. Nauk SSSR*, 269 :543–547, 1983. 33, 48, 49

[180] Y. E. Nesterov and M. J. Todd. Self-Scaled Barriers and Interior-Point Methods for Convex Programming. *Mathematics of Operations Research*, 22(1) :1–42, 1997. 100

[181] D. J. Newman. Location of the maximum on unimodal surfaces. *J. ACM*, 12(3) :395–398, jul 1965. doi:10.1145/321281.321291. 39

[182] J. Nocedal and S. J. Wright, editors. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer-Verlag, New York, 1999. doi:10.1007/b98874. 17, 29, 32, 36, 37, 38, 41, 61

[183] B. O'Donoghue and E. Candès. Adaptive restart for accelerated gradient schemes. *Found. Comput. Math.*, 15(3) :715–732, jun 2015. doi:10.1007/s10208-013-9150-3. 71

[184] E. J. O'Donovan and R. I. Tanner. Numerical study of the Bingham squeeze film problem. *Journal of Non-Newtonian Fluid Mechanics*, 15(1) :75–83, Jan. 1984. doi:10.1016/0377-0257(84)80029-4. 13

[185] J. G. Oldroyd. A rational formulation of the equations of plastic flow for a bingham solid. *Mathematical Proceedings of the Cambridge Philosophical Society*, 43(1) :100–105, 1947. doi:10.1017/S0305004100023239. 6

[186] J. G. Oldroyd. Two-dimensional plastic flow of a bingham solid : A plastic boundary-layer theory for slow motion. *Mathematical Proceedings of the Cambridge Philosophical Society*, 43(3) :383–395, 1947. doi:10.1017/S0305004100023616. 92

[187] L. Orseau and M. Hutter. Line Search for Convex Minimization, July 2023. arXiv :2307.16560 [cs, math]. doi:10.48550/arXiv.2307.16560. 31

[188] J. M. Ortega and W. C. Rheinboldt. Iterative solution of nonlinear equations in several variables. *Academic Press*, 1970. 29

[189] P. Oswald. *Rheophysics The Deformation and Flow of Matter*. Cambridge University Press, Jan. 2009. 3

[190] G. Ovarlez and S. Hormozi, editors. *Lectures on Visco-Plastic Fluid Mechanics*, volume 583 of *CISM International Centre for Mechanical Sciences*. Springer International Publishing, Cham, 2019. doi:10.1007/978-3-319-89438-6. 13, 14

[191] T. C. Papanastasiou. Flows of Materials with Yield. *Journal of Rheology*, 31(5) :385–404, 07 1987. doi:10.1122/1.549926. 13, 14

[192] par J. Cea and et R. Glowinski. Methodes numeriques pour i'ecoulement laminaire d'un fluide rigide viscoplastique incompressible. *International Journal of Computer Mathematics*, 3(1-4) :225–255, 1972. doi:10.1080/00207167208803065. 14

[193] N. Parikh and S. Boyd. Proximal Algorithms. *Foundations and Trends in Optimization*, 1(3) :127–239, Jan. 2014. doi:10.1561/2400000003. 17, 44, 45, 47, 49

[194] E. Polak and G. Ribiere. Note sur la convergence de méthodes de directions conjuguées. *Revue française d'informatique et de recherche opérationnelle. Série rouge*, 3(R1) :35–43, 1969. 33

[195] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5) :1–17, Jan. 1964. doi:10.1016/0041-5553(64)90137-5. 34

[196] B. T. Polyak. A general method for solving extremum problems. *Soviet Mathematics. Doklady*, 8(3) :593–97, 01 1967. 39

[197] B. T. Polyak. The conjugate gradient method in extremal problems. *USSR Computational Mathematics and Mathematical Physics*, 9(4) :94–112, Jan. 1969. doi:10.1016/0041-5553(69)90035-4. 32

[198] B. T. Polyak. *Introduction to optimization*. Translations series in mathematics and engineering. Optimization Software, Inc., New York, 1987. 17, 30, 32, 33, 34, 36, 39, 40

[199] A. Pourzahedi, E. Chaparian, A. Roustaei, and I. A. Frigaard. Flow onset for a single bubble in a yield-stress fluid. *Journal of Fluid Mechanics*, 933 :A21, 2022. doi:10.1017/jfm.2021.1055. 14

[200] M. J. D. Powell. A method for nonlinear constraints in minimization problems. *Optimization, ed. R. Fletcher, Academic Press*, pages 283–98, 1969. 41

[201] M. J. D. Powell. On search directions for minimization algorithms. *Mathematical Programming*, 4(1) :193–201, Dec. 1973. doi:10.1007/BF01584660. 41

[202] W. Prager. *Mécanique des solides isotropes au delà du domaine élastique*. Number 87 in Mémorial des sciences mathématiques. Gauthier-Villars, 1937. 5

[203] W. Prager. On slow visco-plastic flow. Technical report 84, Graduate Division of Applied Mathematics. Brown University, Dec. 1952. Section : Technical Reports. 9, 63

[204] B. N. Pshenichnyi. Algorithms for general mathematical programming problems. *Cybernetics*, 6(5) :677–684, Sept. 1970. doi:10.1007/BF01072819. 38

[205] B. N. Pshenichnyi. The Linearization Method. *Moscow, Nauka*, 1983. 38

[206] A. Putz, I. A. Frigaard, and D. M. Martinez. On the lubrication paradox and the use of regularisation methods for lubrication flows. *Journal of Non-Newtonian Fluid Mechanics*, 163(1) :62–77, Nov. 2009. doi:10.1016/j.jnnfm.2009.06.006. 13

[207] A. M. V. Putz, T. I. Burghelea, I. A. Frigaard, and D. M. Martinez. Settling of an isolated spherical particle in a yield stress shear thinning fluid. *Physics of Fluids*, 20(3) :033102, Mar. 2008. doi:10.1063/1.2883937. 3

[208] J. Rasch and A. Chambolle. Inexact first-order primal–dual algorithms. *Computational Optimization and Applications*, 76(2) :381–430, June 2020. doi:10.1007/s10589-020-00186-y. 49

[209] R. Rockafellar. *Convex Analysis*. Princeton Landmarks in Mathematics and Physics. Princeton University Press, 1997. 18

[210] R. T. Rockafellar. The multiplier method of Hestenes and Powell applied to convex programming. *Journal of Optimization Theory and Applications*, 12(6) :555–562, Dec. 1973. doi:10.1007/BF00934777. 41

[211] R. T. Rockafellar. Monotone Operators and the Proximal Point Algorithm. *SIAM Journal on Control and Optimization*, 14(5) :877–898, Aug. 1976. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/0314056. 28, 45

[212] R. T. Rockafellar. Lagrange multipliers and optimality. *SIAM Review*, 35(2) :183–238, 1993. doi:10.1137/1035044. 41

[213] N. Roussel and P. Coussot. "Fifty-cent rheometer" for yield stress measurements : From slump to spreading flow. *Journal of Rheology*, 49(3) :705–718, May 2005. doi:10.1122/1.1879041. 126

[214] A. Roustaei. *Yield stress fluid flows in uneven geometries : applications to the oil & gas industry*. PhD thesis, University of British Columbia, 2016. doi:http://dx.doi.org/10.14288/1.0300446. 16

[215] A. Roustaei and I. A. Frigaard. The occurrence of fouling layers in the flow of a yield stress fluid along a wavy-walled channel. *Journal of Non-Newtonian Fluid Mechanics*, 198 :109–124, Aug. 2013. doi:10.1016/j.jnnfm.2013.03.005. 16

[216] A. Roustaei, A. Gosselin, and I. A. Frigaard. Residual drilling mud during conditioning of uneven boreholes in primary cementing. Part 1 : Rheology and geometry effects in non-inertial flows. *Journal of Non-Newtonian Fluid Mechanics*, 220 :87–98, June 2015. doi:10.1016/j.jnnfm.2014.09.019. 16

[217] S. Sabach and M. Teboulle. Faster lagrangian-based methods in convex optimization. *SIAM Journal on Optimization*, 32(1) :204–227, 2022. doi:10.1137/20M1375358. 48, 57

[218] J. Salençon. *Mécanique des milieux continus - Tome 1 - Concepts généraux*. Ellipses, Jan. 2002. 3

[219] A. Salim, L. Condat, K. Mishchenko, and P. Richtárik. Dualize, Split, Randomize : Toward Fast Nonsmooth Optimization Algorithms. *Journal of Optimization Theory and Applications*, 195(1) :102–130, Oct. 2022. doi:10.1007/s10957-022-02061-8. 17

[220] P. Saramito. *Efficient C++ finite element computing with Rheolef*, 2015. 16

[221] P. Saramito. A damped Newton algorithm for computing viscoplastic fluid flows. *Journal of Non-Newtonian Fluid Mechanics*, 238 :6–15, Dec. 2016. doi:10.1016/j.jnnfm.2016.05.007. 15, 160

[222] P. Saramito and N. Roquet. An adaptive finite element method for viscoplastic fluid flows in pipes. *Computer Methods in Applied Mechanics and Engineering*, 190(40) :5391–5412, July 2001. doi:10.1016/S0045-7825(01)00175-X. 16, 113

[223] P. Saramito and A. Wachs. Progress in numerical simulation of yield stress fluid flows. *Rheologica Acta*, 56(3) :211–230, Mar. 2017. doi:10.1007/s00397-016-0985-9. 12, 13, 14, 15

[224] S. Saraygord Afshari, F. Enayatollahi, X. Xu, and X. Liang. Machine learning-based methods in structural reliability analysis : A review. *Reliability Engineering & System Safety*, 219 :108223, Mar. 2022. doi:10.1016/j.ress.2021.108223. 151

[225] T. Schiano, B. Bigot, J.-F. Haquet, P. Saramito, and C. Smutek. A viscoplastic approach to corium spreading during a severe nuclear accident. *Nuclear Engineering and Design*, 401 :112045, 2023. doi:https://doi.org/10.1016/j.nucengdes.2022.112045. 1

[226] J. Schurz. The yield stress — An empirical reality. *Rheologica Acta*, 29(2) :170–171, Mar. 1990. doi:10.1007/BF01332384. 3

[227] Schwedoff, Théodore. Recherches expérimentales sur la cohésion des liquides. *J. Phys. Theor. Appl.*, 8(1) :341–359, 1889. doi:10.1051/jphystap:018890080034100. 5

[228] M. W. Scroggs, I. A. Baratta, C. N. Richardson, and G. N. Wells. Basix : a runtime finite element basis evaluation library. *Journal of Open Source Software*, 7(73) :3982, 2022. doi:10.21105/joss.03982. 109

[229] M. W. Scroggs, J. S. Dokken, C. N. Richardson, and G. N. Wells. Construction of arbitrary order finite element degree-of-freedom maps on polygonal and polyhedral cell meshes. *ACM Transactions on Mathematical Software*, 48(2) :18 :1–18 :23, 2022. doi:10.1145/3524456. 109

[230] V. V. Shelukhin. Bingham Viscoplastic as a Limit of Non-Newtonian Fluids. *Journal of Mathematical Fluid Mechanics*, 4(2) :109–127, Apr. 2002. doi:10.1007/s00021-002-8538-7. 12

[231] H.-J. M. Shi, S. Tu, Y. Xu, and W. Yin. A Primer on Coordinate Descent Algorithms. *arXiv :1610.00040*, Jan. 2017. 17, 27

[232] B. Sudret. Polynomial chaos expansions and stochastic finite element methods. In *Risk and Reliability in Geotechnical Engineering*, pages 265–300. CRC Press, 2015. 138, 139, 140, 141

[233] X. Sun, Y. Y. Choi, and J.-I. Choi. Global sensitivity analysis for multivariate outputs using polynomial chaos-based surrogate models. *Applied Mathematical Modelling*, 82 :867–887, 2020. doi:10.1016/j.apm.2020.02.005. 141

[234] P. Szabo and O. Hassager. Flow of viscoplastic fluids in eccentric annular geometries. *Journal of Non-Newtonian Fluid Mechanics*, 45(2) :149–169, Nov. 1992. doi:10.1016/0377-0257(92)85001-D. 12, 15

[235] H. Tabuteau, P. Coussot, and J. R. de Bruyn. Drag force on a sphere in steady motion through a yield-stress fluid. *Journal of Rheology*, 51(1) :125–137, Jan. 2007. doi:10.1122/1.2401614. 3

[236] M. J. Todd. Semidefinite optimization. *Acta Numerica*, 10 :515–560, May 2001. doi:10.1017/S0962492901000071. 41

[237] T. Treskatis. *Fast proximal algorithms for applications in viscoplasticity.* PhD thesis, University of Canterbury, 2016. doi:10.26021/5801. 12, 14, 50, 58, 63, 86

[238] T. Treskatis, M. A. Moyers-Gonzalez, and C. J. Price. A trust-region SQP method for the numerical approximation of viscoplastic fluid flow, Oct. 2015. arXiv :1504.08057 [math]. doi:10.48550/arXiv.1504.08057. 16

[239] T. Treskatis, M. A. Moyers-González, and C. J. Price. An accelerated dual proximal gradient method for applications in viscoplasticity. *Journal of Non-Newtonian Fluid Mechanics*, 238 :115–130, Dec. 2016. doi:10.1016/j.jnnfm.2016.09.004. 14, 50, 63

[240] T. Treskatis, A. Roustaei, I. Frigaard, and A. Wachs. Practical guidelines for fast, efficient and robust simulations of yield-stress flows without regularisation : A study of accelerated proximal gradient and augmented Lagrangian methods. *Journal of Non-Newtonian Fluid Mechanics*, 262 :149–164, Dec. 2018. doi:10.1016/j.jnnfm.2018.05.002. 14, 16, 58, 63, 72, 86, 159

[241] N. V. Tret'yakov. The penalty function method for convex programming problems. *Ekonom. i Matemat. Metody*, 9(3) :526–40, 1973. 41

[242] UNCED. Agenda 21 : program of action from the United Nations organization. Technical report, United Nations Conference on Environment and Development (UNCED), Rio de Janeiro, Brazil, June 1992. §11.13, §13.15, §17.28. URL : https://www.un.org/en/conferences/environment/rio1992. 125

[243] P. Vigneaux, G. Chambon, A. Marly, L.-H. Luu, and P. Philippe. Flow of a yield-stress fluid over a cavity : Experimental and numerical investigation of a viscoplastic boundary layer. *Journal of Non-Newtonian Fluid Mechanics*, 261 :38–49, Nov. 2018. URL : https://hal.science/hal-01857303, doi:10.1016/j.jnnfm.2018.08.005. 94

[244] P. Vigneaux, Y. Shao, and I. A. Frigaard. Confined yield stress lubrication flows for cement paste backfill in underground stopes. *Cement and Concrete Research*, 164 :107038, Feb. 2023. URL : https://hal.science/hal-03892951, doi:10.1016/j.cemconres.2022.107038. 12, 16, 125, 126, 128, 142, 149, 150, 151

[245] S. Villa, S. Salzo, L. Baldassarre, and A. Verri. Accelerated and Inexact Forward-Backward Algorithms. *SIAM Journal on Optimization*, 23(3) :1607–1633, Jan. 2013. Publisher : Society for Industrial and Applied Mathematics. doi:10.1137/110844805. 49

[246] G. Vinay, A. Wachs, and J.-F. Agassant. Numerical simulation of non-isothermal viscoplastic waxy crude oil flows. *Journal of Non-Newtonian Fluid Mechanics*, 128(2) :144–162, July 2005. doi:10.1016/j.jnnfm.2005.04.005. 14

[247] G. Vinay, A. Wachs, and J.-F. Agassant. Numerical simulation of weakly compressible bingham flows : The restart of pipeline flows of waxy crude oils. *Journal of Non-Newtonian Fluid Mechanics*, 136(2) :93–105, 2006. doi:https://doi.org/10.1016/j.jnnfm.2006.03.003. 65

[248] I. C. Walton and S. H. Bittleston. The axial flow of a bingham plastic in a narrow eccentric annulus. *Journal of Fluid Mechanics*, 222 :39–60, 1991. doi:10.1017/S002211209100099X. 12

[249] A. P. Wierzbicki. A penalty function shifting method in constrained static optimization and its convergence properties. *Archiw. Atumo. i Telemech.*, 16(4) :395–416, 1971. 41

[250] R. B. Wilson. *A simplicial algorithm for concave programming*. PhD thesis, Graduate School of Business Administration, Harvard University, 1963. 38

[251] P. Wolfe. The Secant method for simultaneous nonlinear equations. *Communications of the ACM*, 2(12) :12–13, Dec. 1959. doi:10.1145/368518.368542. 36

[252] P. Wolfe. Methods for nonlinear constraints. *Nonlinear Programming*, pages 120–31, 1967. 40

[253] S. J. Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1) :3–34, June 2015. doi:10.1007/s10107-015-0892-3. 17

[254] D. Xiu and G. E. Karniadakis. The Wiener-Askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, 24(2) :619–644, jan 2002. doi:10.1137/S1064827501387826. 139, 140

[255] H. Yamashita and H. Yabe. A primal–dual interior point method for nonlinear optimization over second-order cones. *Optimization Methods and Software*, 24(3) :407–426, 2009. doi:10.1080/10556780902752447. 100

[256] E. Yilmaz and M. Fall, editors. *Paste Tailings Management*. Springer International Publishing, 2017. doi:10.1007/978-3-319-39682-8. 1, 125, 126

[257] K. Yosida. *Functional Analysis*. Springer, Berlin, Heidelberg, 1968. doi:10.1007/978-3-662-11791-0. 26

[258] Z. Yu and A. Wachs. A fictitious domain method for dynamic simulation of particle sedimentation in Bingham fluids. *Journal of Non-Newtonian Fluid Mechanics*, 145(2) :78–91, Sept. 2007. doi:10.1016/j.jnnfm.2007.02.007. 15

[259] Zenodo. European Organization For Nuclear Research and OpenAIRE, 2013. URL : https://www.zenodo.org/, doi:10.25495/7GXK-RD71. 151

[260] M. V. Zibetti, E. S. Helou, R. R. Regatte, and G. T. Herman. Monotone FISTA With Variable Acceleration for Compressed Sensing Magnetic Resonance Imaging. *IEEE Transactions on Computational Imaging*, 5(1) :109–119, Mar. 2019. Conference Name : IEEE Transactions on Computational Imaging. doi:10.1109/TCI.2018.2882681. 50

[261] G. Zoutendijk. *Methods of feasible directions : a study in linear and nonlinear programming*. Elsevier Pub. Co, Amsterdam, 1960. 38