HAL
open science

Optimisation de forme d'une pompe générique de fond de puits

Séverine Baillet

To cite this version:

Séverine Baillet. Optimisation de forme d'une pompe générique de fond de puits. Mathématiques [math]. Université Henri Poincaré - Nancy I, 2007. Français. NNT : 2007NAN10104 . tel-01777259v2

HAL Id: tel-01777259

https://theses.hal.science/tel-01777259v2

Submitted on 5 Dec 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

FACULTÉ DES SCIENCES ET TECHNIQUES
UNIVERSITÉ HENRI POINCARÉ NANCY I
École Doctorale IAEM Lorraine
D.F.D. Mathématiques
en collaboration avec
INSTITUT FRANÇAIS DU PÉTROLE
Direction Technologie, Informatique et Mathématiques Appliquées

Thèse de Doctorat de
Mathématiques

Présentée et soutenue par
Séverine BAILLET

OPTIMISATION DE FORME D'UNE POMPE GÉNÉRIQUE DE FOND DE PUITS

Thèse publique soutenue le 24 septembre 2007

Jury :

M. François JOUVE	Professeur Laboratoire J.L. Lions, Université Paris 7	Président
M. Jean-Antoine DÉSIDÉRI	Directeur de recherche INRIA Sophia Antipolis	Rapporteur
M. Olivier PIRONNEAU	Professeur Laboratoire J.L. Lions, Université Paris 6 et Académie des Sciences	Rapporteur
M. Jean BRAC	Ingénieur de recherche IFP	Examinateur
M. Antoine HENROT	Professeur Institut Elie Cartan, Nancy Université INPL	Examinateur
M. Jan SOKOLOWSKI	Professeur Institut Elie Cartan, Nancy Université UHP	Examinateur
M. Jean FALCIMAIGNE	Ingénieur de recherche IFP	Membre invité

Remerciements

Je souhaite remercier Antoine Henrot pour avoir dirigé ma thèse, ainsi que pour sa disponibilité, son aide et son écoute pendant ces trois années. Je remercie également Jean Brac pour avoir encadré la thèse à l'IFP et pour les enseignements que j'ai pu tirer de mon travail avec lui.

Je remercie Jean-Antoine Désidéri et Olivier Pironneau qui ont eu l'amabilité de rapporter sur ma thèse, ainsi que François Jouve et Jan Sokolowski qui ont accepté d'être membres de mon jury de thèse.

Je voudrais aussi remercier Véronique Henriot pour avoir facilité mon intégration dans son projet et pour l'attention qu'elle a accordée à mon travail. Merci aussi à Jean Falcimaigne pour son aide et ses conseils avisés, ainsi que pour sa participation à mon jury de thèse. Enfin merci à Delphine Sinoquet pour l'intérêt qu'elle a prêté à mes travaux et pour ses suggestions.

Merci à tous ceux qui ont partagé les moments agréables et les plus difficiles avec moi à l'IFP et dont je suis fière d'être devenue l'amie. Zakia, pour les longues discussions que nous avons eues, qui m'ont si souvent aidée à avancer et à ne jamais renoncer, et encore pour tant d'autres choses. Carole, pour sa gentillesse infinie, son sourire vivifiant et son efficacité redoutable. Gabriela, que je considère comme ma grande sœur de l'IFP, pour le temps qu'elle a bien voulu consacrer à me faire partager son expérience et pour celui, tout aussi important, du papotage. Elodie et Gregory pour leur patience avec moi, pour tous les bons moments et toutes nos discussions, passées et à venir, autour d'un café. Enfin merci à ceux qui n'ont été que de passage mais qui ont égayé le quotidien avec leur humour et leur bonne humeur, par ordre d'apparition : Toufik, Henri et Stéphane.

Je tiens à remercier $ylvain et Nico pour leur écoute indéfectible et leurs encouragements, ainsi que Pierre, sans qui les soirées passées à Nancy auraient été bien mornes.

Enfin un merci tout particulier à ceux qui m'ont prodigué leur amour et leur bienveillance pendant ces trois années, et sans qui je ne serais peut-être pas allée si loin : Sylvain, qui m'a apporté un soutien constant, mon frère, avec sa sagesse de docteur, et mes parents, à qui je dois tout.

Table des matières

Introduction **13**

I Modélisation du problème **29**

1 Paramétrisation et propriétés de la géométrie **31**
1.1 Quelques propriétés des B-splines cubiques 31
1.1.1 Définitions et premières propriétés des B-splines cubiques 31
1.1.2 Propriété de monotonie 32
1.1.3 Condition suffisante pour qu'une B-spline n'ait pas de point d'inflexion 34
1.2 Quelques propriétés géométriques de l'intrados de la pompe 36
1.2.1 Comportement local de l'intrados à l'attaque 36
1.2.2 Comportement local de l'intrados à la fuite 37
1.3 Recherche d'un nombre raisonnable de points de contrôle de B-spline pour approcher la géométrie de référence de la pompe de fond de puits 39
1.3.1 Objectif 39
1.3.2 Grandes lignes de la méthode retenue 40
1.3.3 Méthode des moindres carrés pour déterminer Y 40
1.3.4 Liste des contraintes à prendre en compte pour les aubages 41
1.3.5 Mise en œuvre numérique 43
1.3.6 Résultats numériques pour l'extrados 45
1.3.7 Résultats numériques pour la cambrure 46

2 Résolution numérique des équations de Navier-Stokes **47**
2.1 Le logiciel de *Computational Fluid Dynamics* (CFD) Fluent 47
2.1.1 Présentation générale 47
2.1.2 Méthode de résolution numérique 47
2.2 Modélisation d'un écoulement de gasoil dans la pompe 48
2.2.1 Création et maillage de la géométrie dans Gambit 48
2.2.2 Spécifications de la simulation numérique 48
2.2.3 Modélisation d'un écoulement basique 49
2.2.4 Modélisation d'un écoulement dans un domaine mobile 50
2.2.5 Modèles de turbulence 52

3 Résultats de convergence pour un problème semi-linéaire et un problème de Stokes dans une géométrie périodique **55**
3.1 Introduction 56
3.2 Semilinear problems in N-dimension 58
3.3 The Stokes problem with Navier slip boundary conditions 61

3.4 Conclusion 64

II Evaluation du gradient **67**

4 Calcul d'un gradient incomplet de la fonction coût **69**
4.1 Formules d'intégration sur des bords variables 69
4.2 Calcul incomplet de gradient 70
4.2.1 Dérivation de l'intégrale sur le bord du domaine : $J_1 = \int_{\partial\Omega} pn_z$ 71
4.2.2 Dérivation de l'intégrale sur les parties déformées : $J_2 = \int_D pn_z$ 73
4.2.3 Expression finale du gradient incomplet 73
4.3 Extensions du vecteur normal et du vecteur champ de déformation 74
4.4 Dérivation par rapport au domaine de l'extension de la normale 75
4.4.1 Résultat utile 75
4.4.2 Premiers éléments de calcul 75
4.4.3 Calcul du terme $\mathcal{N}' = -\nabla(V.\mathcal{N})$ 77
4.4.4 Calcul de la courbure moyenne H 80
4.5 Conclusion 80

5 Calcul du champ de déformation induit par le déplacement d'un point de contrôle **81**
5.1 Théorème des fonctions implicites 81
5.2 Déplacement d'un point de contrôle de l'extrados 82
5.2.1 Champ de déformation induit à l'extrados 82
5.2.2 Champ de déformation induit à l'intrados 83
5.3 Déplacement d'un point de contrôle de la cambrure 84
5.3.1 Champ de déformation induit à la cambrure 84
5.3.2 Champ de déformation induit à l'intrados 85
5.4 Déplacement d'un point de contrôle du moyeu 86
5.4.1 Champ de déformation induit au moyeu 86
5.5 Ecriture des vecteurs en dimension 3 87
5.5.1 Champ de déformation à l'extrados des aubes 87
5.5.2 Champ de déformation à l'intrados des aubes 88
5.5.3 Champ de déformation normal au moyeu 88
5.6 Cas particulier du champ de déformation sur les surfaces d'entrée et de sortie de l'étage de pompe 89

6 Dérivée de l'extension de la normale **91**
6.1 Calculs relatifs au moyeu 91
6.2 Calculs généraux relatifs à une aube 93
6.3 Distinction entre l'intrados et l'extrados 96
6.3.1 L'extrados 96
6.3.2 L'intrados 96

III Optimisation du moyeu **101**

7 Mise en œuvre de l'optimisation **103**
7.1 A propos de l'existence d'un minimum 103

7.2 Déroulement de l'algorithme d'optimisation 105
7.3 Evaluation numérique de la fonction coût et du gradient incomplet 106
7.4 Calcul des dérivées des variables d'état 106
7.4.1 Les macros de Fluent 107
7.4.2 Schéma numérique alternatif pour le calcul des variables d'état au centre des faces 109
7.4.3 Schéma numérique pour le calcul des dérivées au centre des faces 112
7.4.4 Conclusions 113
7.5 Algorithmes d'optimisation sans contrainte 114
7.5.1 Méthodes de gradient 114
7.5.2 Méthodes de Newton et de quasi-Newton 115
7.5.3 Conclusion 116
7.6 Prise en compte des contraintes géométriques pour le problème du moyeu 117
7.6.1 Position du problème spécifique d'optimisation du moyeu 117
7.6.2 Recensement des contraintes 118
7.6.3 Réduction du problème 119
7.6.4 Choix de la géométrie initiale 119

8 Analyse critique de la méthode de gradient incomplet 121
8.1 Résultats numériques pour la méthode de gradient incomplet 121
8.1.1 Contraintes de continuité $\mathcal{C}^0$ 121
8.1.2 Contraintes sur le caractère $\mathcal{C}^1$ 123
8.1.3 Perspectives pour un nombre de degrés de liberté supérieur 125
8.2 Comparaison avec les résultats des différences finies 127
8.2.1 Calcul du gradient par différences finies 127
8.2.2 Comparaison de l'angle entre gradient incomplet et gradient calculé par différences finies 128
8.2.3 Résultats numériques par les différences finies 129
8.2.4 Conclusion 130
8.3 Qualité du gradient incomplet et cas du moyeu cylindrique 131
8.3.1 Gradient incomplet pour un moyeu cylindrique de rayon fixé 131
8.3.2 Gradient incomplet pour un moyeu cylindrique de rayon variable 133
8.4 Bien-fondé de la méthode 135
8.4.1 Justification par l'expérience 135
8.4.2 Etude des dérivées éliminées lors du calcul du gradient incomplet 136
8.5 Perspectives 138
8.5.1 Optimisation des aubes 138
8.5.2 Solutions envisageables pour l'amélioration de l'algorithme d'optimisation . . . 138

Conclusion 141

Bibliographie 145

Glossaire

$0\mathcal{E} = (0, e_1, e_2, e_3)$	repère cartésien lié à la pompe
A_r	longueur axiale d'une aube au rotor
A_s	longueur axiale d'une aube au stator
C	bords de Ω non déformés au cours de l'optimisation
c_i	contrainte
D	bords de Ω déformés au cours de l'optimisation
$D(.)$	matrice jacobienne
$D^2(.)$	matrice hessienne
D_1	diamètre minimal du moyeu
D_2	diamètre maximal du moyeu
D_t	diamètre du carter
d_k	direction de descente
$\partial\Omega$	bord de Ω
E	face d'entrée d'un étage de pompe
e	épaisseur d'une aube
ε	pas des différences finies
Γ_a	surface d'une aube
Γ_m	surface du moyeu
H	courbure moyenne
J	fonction coût
jeu	jeu axial
L	longueur axiale d'un étage de pompe
L_r	longueur axiale du rotor
L_s	longueur axiale du stator
M_c	point de la cambrure
M_e	point de l'extrados
m	nombre de contraintes
μ	viscosité dynamique

n	nombre de points de contrôle d'une B-spline
$n = (n_1, n_2, n_3)$ $= (n_x, n_y, n_z)$	vecteur normal orienté vers l'extérieur du domaine considéré
$\overrightarrow{n_c}$	vecteur normal à la cambrure
$\overrightarrow{n_{ce}}$	vecteur normal à la couronne d'entrée
$\overrightarrow{n_{cs}}$	vecteur normal à la couronne de sortie
$\overrightarrow{n_e}$	vecteur normal à l'extrados
$\overrightarrow{n_m}$	vecteur normal au moyeu
Ω	domaine d'écoulement du fluide, ouvert de $\mathbb{R}^3$ (un étage de pompe)
Ω_{adm}	ensemble des domaines Ω admissibles
ω	vitesse de rotation des parois mobiles de la pompe
ω_D	vitesse de rotation des parois de la pompe (nulle sur les parois fixes et égale à ω sur les parois mobiles)
$(p_i)_{i=0,\ldots,3}$	fonctions de base des B-splines
r	première coordonnée cylindrique
ρ	masse volumique
S	face de sortie d'un étage de pompe
$(s_c(t), z_c(t))$	B-spline de la cambrure
$(s_e(t), z_e(t))$	B-spline de l'extrados
$(s_i(t), z_i(t))$	B-spline de l'intrados
t	paramètre de B-spline
t_k	pas de la méthode de descente
θ	seconde coordonnée cylindrique
$\overrightarrow{\tau_c}$	vecteur tangent à la cambrure
$\overrightarrow{\tau_e}$	vecteur tangent à l'extrados
$U = (u_1, u_2, u_3)$ $= (u_x, u_y, u_z)$	vecteur vitesse du fluide
V	extension du champ de déformation $\mathcal{V}$
$\mathcal{V}$	champ de déformation
$\overrightarrow{\mathcal{V}_a}$	champ de déformation sur une aube
$\overrightarrow{\mathcal{V}_{ce}}$	champ de déformation à l'entrée
$\overrightarrow{\mathcal{V}_{cs}}$	champ de déformation à la sortie
$\overrightarrow{\mathcal{V}_e}$	champ de déformation à l'extrados
$\overrightarrow{\mathcal{V}_i}$	champ de déformation à l'intrados
$\overrightarrow{\mathcal{V}_i^c}$	champ de déformation à l'intrados engendré par un déplacement à la cambrure
$\overrightarrow{\mathcal{V}_i^e}$	champ de déformation à l'intrados engendré par un déplacement à l'extrados
$\overrightarrow{\mathcal{V}_m}$	champ de déformation au moyeu

$X = (x_1, x_2, x_3)$	point du repère $0\mathcal{E}$
$= (x, y, z)$	
X_k	vecteur des inconnues de l'optimisation à la k-ième itération
x_C	ensemble des variables de contrôle de la géométrie

Introduction

L'*optimisation de forme* est un domaine de recherche des mathématiques ayant pour principe d'optimiser la forme d'un objet afin de le rendre le plus efficace possible, ou le plus résistant, ou le plus aérodynamique, ou le plus silencieux, ou le plus léger... ou le moins cher possible. Grâce au développement de la modélisation et du calcul numérique, ce sont des préoccupations auxquelles on trouve maintenant des réponses de plus en plus satisfaisantes.
En optimisation de forme, on cherche à trouver des minima (ou des maxima) de fonctions, c'est-à-dire à résoudre et analyser des problèmes du type :

$$\begin{gathered}\textit{Trouver } \Omega^* \textit{ solution de} \\ \Omega^* \in \Omega_{adm},\, J(\Omega^*) = \min_{\Omega \in \Omega_{adm}} J(\Omega)\end{gathered}$$

où Ω_{adm} est un ensemble de parties de $\mathbb{R}^N$, dites domaines ou formes *admissibles*, et J est une fonctionnelle définie sur Ω_{adm} à valeur dans $\mathbb{R}$, dite *fonctionnelle de forme*, ou *fonction coût*, ou encore *fonction objectif*.
Comme le montre la liste non exhaustive citée ci-dessus des bénéfices à tirer de l'optimisation de forme, cette dernière peut être mise en oeuvre dans de nombreux domaines (aéronautique, automobile, construction... ou même médecine). C'est dans le cadre de la production pétrolière que l'on s'y intéresse ici.

L'IFP a développé, il y a environ 15 ans, une pompe pour effluents pétroliers composés d'un mélange de pétrole et de gaz en proportions variables. Il s'agit d'une pompe rotodynamique. Son objectif est, à débit constant entre l'entrée et la sortie, d'augmenter la pression en transformant une partie de l'énergie mécanique de l'arbre de rotation en énergie hydraulique acquise par le fluide. La partie restante est une perte dégradée en chaleur.
Une particularité de cette pompe est qu'elle est introduite dans le réservoir lui-même, au cœur de la réserve fossile. Ainsi, lorsque la pression en surface est devenue trop faible, au lieu d'abandonner la production du puits la pompe immergée permet d'en prolonger l'exploitation. Cette nouvelle approche de la récupération a permis d'augmenter sensiblement le niveau des réserves prouvées.

La conception des formes géométriques de la pompe est assez ancienne maintenant, et n'avait pas fait l'objet de calculs de CFD à l'époque. De plus, elle tient compte de contraintes liées aux machines-outils d'alors, qui sont aujourd'hui dépassées. Les limitations de formes imposées alors n'ont plus lieu d'être et une paramétrisation plus souple que l'ancienne paramétrisation a été mise en place. L'IFP souhaite donc tester de nouvelles formes et mener une optimisation à l'aide de la simulation numérique, tout en respectant les règles de l'art courantes dans le métier de pompiste.

Généralités sur la physique du problème

Principe de fonctionnement de la pompe de fond de puits

La pompe est constituée de plusieurs étages identiques disposés en série. Chaque étage contient un rotor et un stator, pourvus d'aubes. Le fluide s'écoule dans un canal délimité par un moyeu et un carter.

FIG. 1: *Coupe de la partie fonctionnelle de la pompe de fond de puits avec vue partielle du carter. Sur cette coupe, un stator (en bleu) est précédé d'un rotor et est suivi d'un autre rotor. L'arbre de rotation est visible et les moyeux sont partiellement cachés par les aubages.*

Il s'agit d'une pompe *rotodynamique*, c'est-à-dire que les pièces mécaniques en rotation mettent le fluide en mouvement. Ainsi, le rotor donne par sa rotation de la quantité de mouvement au fluide. Le stator est fixe et redresse le champ de vitesse du fluide dans la direction de l'axe de rotation. La pompe est dite *hélico-axiale* en raison de la forme en hélice du rotor et parce que l'écoulement y est délivré dans l'axe de rotation de la pompe. On rappelle que l'objectif d'une telle pompe est, à débit constant entre l'entrée et la sortie, d'augmenter la pression en transformant une partie de l'énergie mécanique de l'arbre de rotation en énergie hydraulique acquise par le fluide. On peut augmenter la pression en ajoutant des étages les uns derrière les autres. Les fabricants produisent à cet effet des modules standardisés d'une quinzaine d'étages qu'il suffit d'assembler en série. Certaines pompes peuvent ainsi comporter juqu'à une centaine d'étages. Mais ceci a un coût, approximativement proportionnel à la longueur de la pompe. Etant donné qu'une pompe doit généralement fournir un gain de pression total imposé, l'objectif de l'optimisation de forme de la pompe va donc être de *maximiser le gain de pression par unité de longueur*, ce qui revient à minimiser le prix à payer pour fournir un gain de pression donné.

Il serait trop coûteux en temps de calcul de représenter une pompe complète, avec la totalité de ses étages. On se contente donc la plupart du temps de ne considérer qu'un étage, constitué d'un rotor et d'un stator. Différentes conditions aux limites en entrée et en sortie de cet étage peuvent être imposées pour représenter l'environnement de celui-ci, on y reviendra dans le chapitre 2, consacré à la modélisation de l'écoulement.
Par ailleurs, la pompe de référence considérée est une pompe dite *R3S11*, c'est-à-dire qu'elle comporte trois aubes au rotor et onze aubes au stator. L'expérience a montré qu'il vaut mieux que les nombres d'aubes au rotor et au stator soient premiers entre eux, afin d'éviter le phénomène de cliquetis. Ce phénomène est transparent numériquement, c'est pourquoi on se permet de modéliser des pompes dans lesquelles les nombres d'aubes au rotor et au stator sont proportionnels, encore une fois dans le but de simplifier le problème considéré. Ainsi il n'est plus nécessaire de mener les calculs sur toute la cir-

conférence de la pompe, mais on peut se contenter d'une fraction de celle-ci. Pour modéliser une pompe R3S11, on estime que son comportement est proche de celui d'une R3S12. Il suffit alors de représenter un tiers de pompe et d'utiliser des conditions de périodicité sur les faces latérales de celle-ci pour reconstituer l'étage entier.

Spécifications de la géométrie de la pompe

Les spécifications géométriques de la pompe de fond de puits utilisées jusqu'en 2004 étaient assez contraignantes et répondaient à des exigences de conception aujourd'hui dépassées. Un tri et une simplification des anciennes spécifications ont été réalisés et ont permis de retenir celles qui, à elles seules, permettent de définir la géométrie de la pompe.

Rotor et stator On note L_r la longueur axiale du rotor et L_s la longueur axiale du stator. On désigne par L la longueur axiale d'un étage de pompe : $L = L_r + L_s$.

Moyeu Le moyeu est défini comme une surface de révolution autour de l'axe de la pompe. On désigne par D_1 le diamètre du moyeu à l'entrée du rotor (et à la sortie du stator), et par D_2 le diamètre du moyeu à la sortie du rotor (et à l'entrée du stator).

Carter Le carter est un cylindre d'axe l'axe de rotation de la pompe. Son diamètre intérieur, dit diamètre de tête, est noté D_t.

Remarque Le fait qu'il s'agisse d'une pompe de fond de puits n'entraîne pas de contraintes fortes sur les longueurs axiales. En revanche, comme le puits a un diamètre donné, il est impossible d'augmenter le diamètre du carter de la pompe. Afin de fixer les idées, on donne les ordres de grandeur suivants, correspondant à la géométrie de référence de la pompe à optimiser : $L^{ref} \simeq 0.1$ m et $D_t^{ref} \simeq 0.1$ m.

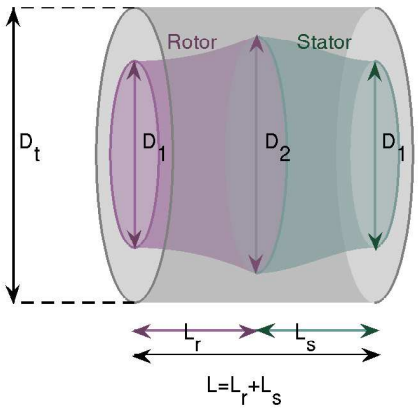

FIG. 2: *Spécifications pour le moyeu du rotor (violet), le moyeu du stator (vert) et le carter (gris).*

Aubages Les aubes du rotor et du stator sont paramétrées de la même façon. Leur longueur axiale est notée A_r au rotor et A_s au stator. Leur épaisseur, variable, est notée e. Trois courbes servent traditionnellement à caractériser le profil d'une aube : la cambrure, l'extrados et l'intrados. Si on considère l'aube comme une cuillère pénétrant le fluide, l'extrados en est le côté extérieur, et l'intrados, le côté intérieur.

La cambrure est la ligne centrale du profil. On a pris le parti de construire la géométrie d'une aube grâce à sa ligne de cambrure et à l'extrados. L'intrados est construit par symétrie de l'extrados par rapport à la cambrure. Etant donné un repère d'origine O, si X_e désigne un point de l'extrados, et X_c son projeté orthogonal sur la cambrure, son symétrique X_i sur l'intrados est donc défini par

$$\overrightarrow{OX_i} = 2\overrightarrow{OX_c} - \overrightarrow{OX_e} \text{ ou encore } X_i = X_c + \overrightarrow{X_eX_c} = X_c + e\frac{\overrightarrow{X_eX_c}}{||\overrightarrow{X_eX_c}||}.$$

On appelle *bord d'attaque* et *bord de fuite* les extrémités des aubes où le fluide attaque et se sépare de l'aube. De part et d'autre des aubes, des jeux axiaux sont imposés, afin d'éviter les frottements entre les pièces fixes et les pièces mobiles. Ces jeux ont une valeur unique que l'on note jeu et on a $L_r = A_r + 2\,jeu$ et $L_s = A_s + 2\,jeu$.

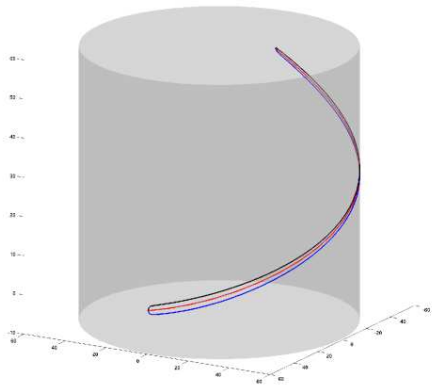

FIG. 3: *Profil d'une aube de rotor déployé sur une surface cylindrique coaxiale au carter. En rouge, la ligne de cambrure ; en bleu, l'extrados ; en noir, l'intrados.*

Méthode de paramétrisation

L'optimisation de forme de la pompe porte sur la forme des aubes et du moyeu, pour le rotor et pour le stator, c'est-à-dire sur la géométrie complète, à l'exception du carter cylindrique.
On utilise deux types de courbes en deux dimensions pour la paramétrisation :

- les courbes génératrices du moyeu, qui est une surface de révolution, pour le rotor et pour le stator ;
- le profil développé des aubes projetées sur un cylindre tel que le carter, plus précisément la courbe de la cambrure et celle de l'extrados. Les aubes ont une géométrie complexe en trois dimensions qui ne peut être ramenée en deux dimensions. On a donc procédé à une simplification : on a supposé qu'il suffit de connaître le profil des aubes sur un cylindre et de le projeter radialement entre le carter et le moyeu pour obtenir la géométrie de l'aube. L'inconvénient est qu'ainsi le mur d'aube est droit alors qu'il aurait aussi été intéressant d'optimiser sa forme, mais il ne s'agit que d'une première approche.

Ainsi, grâce à des courbes paramétrées en deux dimensions seulement on peut reconstituer une approximation de la géométrie de la pompe en trois dimensions. Pour paramétrer les courbes que l'on a choisies, on utilise des B-splines paramétriques cubiques, dont la présentation fait l'objet du chapitre 1.

Quelques défauts actuels de l'écoulement

En observant simplement les résultats de la simulation numérique de l'écoulement dans la pompe initiale, on peut d'emblée recenser quelques défauts de forme, et avoir l'intuition des améliorations que peut apporter une optimisation de forme. En voici quelques exemples.
Sur la figure FIG.4, on observe un comportement typique en turbomachinerie. La pression statique présente un pic au niveau du bord d'attaque de l'aube du rotor, situé à l'abscisse nulle du graphique. On peut penser qu'en modifiant la forme des aubes, on parviendra à lisser cette courbe, à gommer l'irrégularité, et qu'ainsi on améliorera le gain de pression dans le rotor.

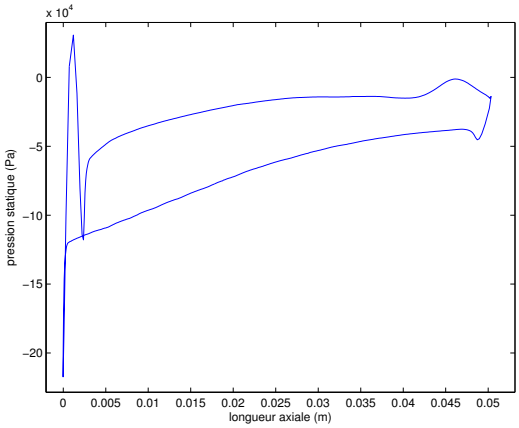

FIG. 4: *Pression statique le long d'une aube de rotor, dans une pompe mince.*

Sur la figure FIG.5, on observe la recirculation du fluide derrière les aubes du stator. Le fluide n'épouse pas correctement les aubes du stator et ces perturbations de l'écoulement contribuent à faire chuter la pression dans le stator. Une amélioration de la forme des aubes du stator est nécessaire.

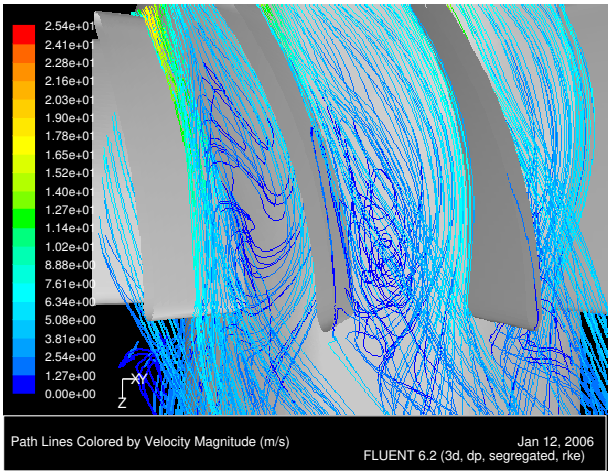

FIG. 5: *Lignes de courant colorées par amplitude de la vitesse entre les aubes du stator.*

La figure FIG.5 permet aussi d'observer que le fluide se décolle du moyeu du stator. Ceci est net sur les figures FIG.6. Le fluide, après avoir passé la partie la plus étrécie du canal hydraulique, au niveau de l'interface entre rotor et stator, est emporté et n'épouse pas la forme du moyeu du stator. Ceci se traduit

par une norme faible de la vitesse lorsque l'on trace son profil près du moyeu, norme qui augmente lorsqu'on le trace plus près du carter. Par conséquent on pressent que la forme du moyeu au stator sera amenée à changer lors de l'optimisation.

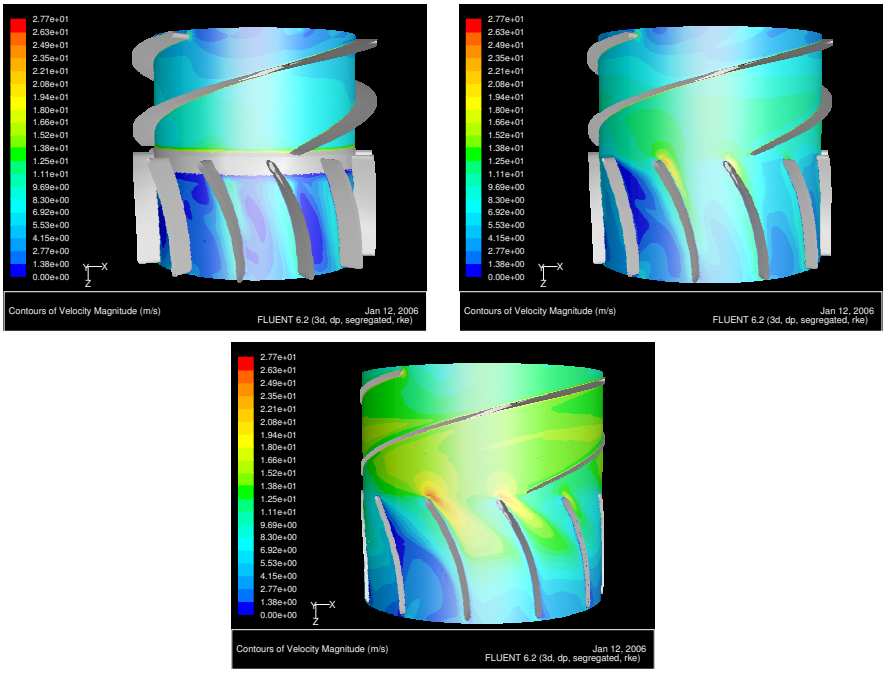

FIG. 6: *Vitesse du fluide dans l'étage de pompe complète, aux distances du carter suivantes, de gauche à droite :* 10*mm,* 5*mm et* 2*mm.*

Sur la figure FIG.7 on observe l'angle absolu formé par le fluide avec la direction tangentielle, en fonction de l'angle radial, à l'entrée du stator. Les lignes verticales signalent l'attaque des aubes de stator. On simule l'écoulement au moment où les aubes de rotor viennent de passer en vis-à-vis des première, cinquième et neuvième aubes de stator. On constate que ce passage engendre une forte diminution de l'angle absolu du fluide. L'angle se rétablit dans le reste du canal hydraulique. Ainsi, le fluide attaque mal les aubes du stator, particulièrement les première, cinquième et neuvième, et on en déduit que l'angle d'attaque des aubes du stator, signalé par un trait en pointillés noirs à 49°, est mal ajusté à l'écoulement.

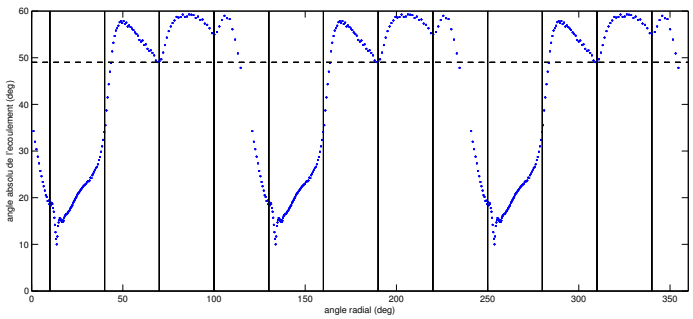

FIG. 7: *Angle absolu formé par l'écoulement du fluide et la direction tangentielle en fonction de l'angle radial, en entrée de stator. Les lignes verticales figurent la position des attaques des aubes du stator. L'angle d'attaque des aubes est représenté par la ligne horizontale en pointillés.*

Problème d'optimisation

On rappelle qu'un problème d'optimisation de forme consiste à trouver le minimum ou le maximum d'une *fonction objectif*, ou *fonction coût*, dont l'évaluation nécessite le calcul des solutions d'une équation d'état dans un domaine dépendant des paramètres du problème, tout en respectant certaines contraintes.
Une forme Ω est généralement définie grâce à un nombre fini de paramètres x_C appelés *variables de contrôle*. Ainsi on cherche dans un ensemble Ω_{adm} de formes admissibles Ω définies par les variables de contrôle x_C, satisfaisant des contraintes $g(x_C) \leq 0$, la forme optimale Ω^*, de variables de contrôle x_C^*, minimisant (ou maximisant) une fonction coût $J(x_C, W(x_C))$. La fonction coût dépend de x_C et de la solution $W(x_C)$ des équations d'état $E(x_C, W(x_C)) = 0$ posées sur $\Omega(x_C)$.

Fonction coût

L'objectif d'une pompe est d'augmenter la pression du fluide entre son entrée et sa sortie. Pour des raisons pratiques on mène l'optimisation sur un seul étage de pompe. Par conséquent le choix qui s'impose naturellement pour la fonction coût est :

$$J(x_C, W(x_C)) = \frac{\int_S p - \int_E p}{L}$$

où

- S désigne la surface de sortie d'un étage de pompe Ω de variables de contrôle x_C,
- E désigne la surface d'entrée d'un étage de pompe Ω de variables de contrôle x_C,
- L désigne la longueur axiale d'un étage de pompe Ω de variables de contrôle x_C,
- p désigne la pression statique[1], solution des équations de Navier-Stokes sur le domaine Ω de variables de contrôle x_C.

Le problème d'optimisation que l'on considère peut donc être résumé sous la forme :

Trouver la forme de pompe Ω^*, de variables de contrôle x_C^*, telle que
$$J(x_C^*) = \max_{x_C \, tq \, \Omega(x_C) \in \Omega_{adm}} J(x_C, W(x_C))$$
sous les contraintes géométriques $g(x_C) \leq 0$
et où $W(x_C)$ est solution des équations de Navier-Stokes $E(x_C, W(x_C)) = 0$.

Compte tenu du mode de représentation de la géométrie choisi, les variables de contrôle de l'optimisation sont les coordonnées des points de contrôle des B-splines. Il reste à recenser les contraintes géométriques.

[1] En fait on cherche à maximiser le gain de pression totale. Or, la pression totale s'exprime ainsi :

$$p_{totale} = p_{statique} + p_{dynamique}, \text{ avec } p_{dynamique} = \frac{1}{2}\rho ||U||^2,$$

où ρ est la masse volumique du fluide et U est sa vitesse. Etant donné que l'étude est menée pour un fluide incompressible s'écoulant à débit constant entre deux sections de même surface (l'entrée et la sortie d'un étage de pompe), la pression dynamique est la même en entrée et en sortie d'étage. Par conséquent le gain de pression totale dans un étage de pompe est égal au gain de pression statique. On peut donc n'utiliser que la pression statique dans la fonction coût.

Contraintes géométriques

Contraintes techniques

La plupart des contraintes servent à garantir la viabilité de la pompe. Si ces contraintes techniques ne sont pas respectées, il est possible que la pompe soit détériorée, ou que le fonctionnement soit sérieusement altéré. Ces contraintes sont les suivantes :

- cambrure des aubages :
  - convexité ;
- extrados des aubages :
  - convexité,
  - tangente au point d'attaque de l'extrados normale à la ligne de cambrure,
  - épaisseur minimale de $1,6$ mm (soit $0,8$ mm de part et d'autre de la cambrure) en dehors des extrémités ("extrémité" définie à 10% de la longueur de l'aube, de chaque côté) ;
- intrados des aubages :
  - deux points d'inflexion à l'intrados,
  - les points d'inflexion à l'intrados doivent se trouver dans les "80% centraux" de la longueur de l'aube ;
- moyeux du rotor et du stator :
  - $D_1 \geq 40\%.D_t$;
  - $D_2 \leq 90\%.D_t$;
  - $20\%.D_t \leq L = L_r + L_s \leq 150\%.D_t$;
- moyeu du rotor :
  - convexité,
  - tangente horizontale à l'entrée du rotor ;
- moyeu du stator :
  - un seul point d'inflexion,
  - tangente horizontale à la sortie du stator ;
- autre :
  - jeu *jeu* supérieur ou égal à $1,2$ mm au rotor et à $1,2$ mm au stator (soit $2,4$ mm de jeu minimal entre rotor et stator), aussi bien en entrée qu'en sortie ;
  - au rotor et au stator, les longueurs axiales du moyeu L_r et L_s sont liées par une contrainte d'égalité à celles des aubes A_r et A_s : $L_r = A_r + 2\,jeu$ et $L_s = A_s + 2\,jeu$.

Contraintes pratiques

Si les contraintes pratiques ne sont pas respectées, la géométrie de la pompe ne peut pas être définie. On en compte deux :

- le point d'attaque de l'extrados est confondu avec le point d'attaque de la cambrure, au rotor et au stator ;
- le point de fuite de l'extrados est confondu avec le point fuite de la cambrure, au rotor et au stator.

Synthèse bibliographique

Deux grandes classes de méthodes employées pour traiter les problèmes d'optimisation de forme peuvent être identifiées : la classe des *méthodes déterministes*, et celle des *méthodes non-déterministes*. On n'évoquera que rapidement la seconde classe, pour s'attarder davantage sur la première, dont est issue la méthode retenue dans le cadre de cette thèse.

Les méthodes non-déterministes

L'une des principales méthodes non-déterministes est la *méthode évolutionniste*, ou encore dite *génétique*. Elle consiste à imiter le processus d'évolution naturelle d'une population d'individus qui s'adapte, au fil des générations, à son environnement. Plus précisément, dans le cadre d'un problème d'optimisation de forme, une population initiale est générée, par sélection aléatoire d'individus (i.e. de formes). La performance de chaque individu est évaluée en calculant la fonction objectif correspondante. Une nouvelle génération d'individus est ensuite créée par sélection, reproduction, croisement et mutation des individus de la première génération. Les enfants remplacent les parents et au fil des générations, les individus deviennent meilleurs du point de vue des critères de l'optimisation.
Si les méthodes déterministes convergent souvent vers des minima locaux, les méthodes non-déterministes trouvent généralement le minimum global. Cependant elles demandent un très grand nombre d'évaluations de la fonction objectif et par conséquent sont très coûteuses en temps et en puissance de calcul. De plus, il faut choisir soigneusement le nombre d'évaluations à réaliser car une population trop petite peut ne mener qu'à un minimum local, tandis qu'une trop grande population ralentira considérablement le processus d'optimisation.
Pour plus d'informations et une introduction à ces méthodes, on pourra lire le chapitre 8 de [1] rédigé par M. Schoenauer.

Les méthodes déterministes

Les méthodes déterministes classiques demandent de connaître le gradient de la fonction coût afin de déterminer une direction de descente. A chaque itération de l'algorithme d'optimisation la forme est modifiée, avec une méthode d'optimisation basée sur le calcul du gradient, et ceci jusqu'à convergence vers un minimum de la fonction coût. On ne s'attarde pas ici sur les méthodes d'optimisation qui seront évoquées plus en détail au chapitre 7. Dans le présent travail, la difficulté principale d'une telle méthode est d'évaluer le gradient.

Evaluation du gradient

Méthode de l'équation adjointe

Cette méthode consiste à déterminer indirectement le gradient de la fonction coût en résolvant une équation adjointe dont les coefficients sont définis en fonction des solutions des équations d'état et de la fonction coût à minimiser. Le détail du principe, donné ci-après, est inspiré de [47].
La fonction coût J dépend des variables de contrôle x_C et de la solution $W(x_C)$ des équations d'état . La variation des x_C entraîne la variation suivante de la fonction coût :

$$\frac{dJ}{dx_C} = \frac{\partial J}{\partial x_C} + \frac{\partial J}{\partial W}\frac{\partial W}{\partial x_C}.$$

Par ailleurs, de $E(x_C, W(x_C)) = 0$ on tire :

$$\frac{dE}{dx_C} = \frac{\partial E}{\partial x_C} + \frac{\partial E}{\partial W}\frac{\partial W}{\partial x_C} = 0,$$

d'où

$$\frac{dJ}{dx_C} = \frac{\partial J}{\partial x_C} - \frac{\partial J}{\partial W}\left(\frac{\partial E}{\partial W}\right)^{-1}\frac{\partial E}{\partial x_C}.$$

On introduit une nouvelle variable ϕ, appelée variable adjointe, telle que :

$$^T\phi\left(\frac{\partial E}{\partial W}\right) = \frac{\partial J}{\partial W}, \tag{1}$$

ce qui permet d'exprimer le gradient de J sous la forme finale :

$$\frac{dJ}{dx_C} = \frac{\partial J}{\partial x_C} - {}^T\phi\frac{\partial E}{\partial x_C}. \tag{2}$$

L'équation (2) est maintenant indépendante de W. L'équation (1) est appelée *équation adjointe* et si $E(x_C, W(x_C)) = 0$ est un système d'équations aux dérivées partielles, (1) l'est aussi. Il peut être difficile de déterminer ce nouveau système, ainsi que les conditions aux limites appropriées.

Sensitivités incomplètes ou gradient incomplet

L'idée de cette approche est de simplifier l'expression du gradient pour ne garder que les termes dominants.
Si on note $x_{\mathcal{G}}$ les variables géométriques du problème (normales, courbures, dérivées des normales...), le gradient de la fonction coût peut être écrit comme suit :

$$\frac{dJ}{dx_C} = \frac{\partial J}{\partial x_C} + \frac{\partial J}{\partial x_{\mathcal{G}}}\frac{\partial x_{\mathcal{G}}}{\partial x_C} + \frac{\partial J}{\partial W}\frac{\partial W}{\partial x_{\mathcal{G}}}\frac{\partial x_{\mathcal{G}}}{\partial x_C}.$$

Dans cette expression, le calcul du terme $\dfrac{\partial W}{\partial x_{\mathcal{G}}}$ est le plus difficile à réaliser, et le plus coûteux en temps de calcul.
Après l'observation des différents termes de l'expression, et sans justification théorique, il ressort que pour une fonction coût vérifiant les assertions suivantes [49] :

- J est une intégrale sur les frontières à optimiser D du domaine $\partial\Omega$ de la forme $\int_D \varphi(x_C, x_{\mathcal{G}})\psi(W)$,
- la courbure locale de D n'est pas trop grande,
- φ et ψ sont telles que l'on puisse vérifier formellement $\frac{1}{|\varphi|}\left|\frac{\partial\varphi}{\partial n}\right| >> \frac{1}{|\psi|}\left|\frac{\partial\psi}{\partial W}\right|$, où n est la normale à la paroi et avec $\left|\frac{\partial\varphi}{\partial n}\right| = O(1)$,

la partie dominante de $\dfrac{dJ}{dx_C}$ est la variation par rapport aux variables géométriques $\dfrac{\partial J}{\partial x_{\mathcal{G}}}$. La variation par rapport à la solution W est négligeable quand x_C varie peu. On peut alors approcher le gradient par :

$$\frac{dJ}{dx_C} \simeq \frac{\partial J}{\partial x_C} + \frac{\partial J}{\partial x_{\mathcal{G}}}\frac{\partial x_{\mathcal{G}}}{\partial x_C}.$$

Ainsi il est plus important de calculer une solution W précise et d'effectuer une évaluation incomplète du gradient que de calculer un gradient exact à partir d'un état W peu précis [57].
Bien que cette méthode soit sous-optimale et présente des limitations, elle a pour avantage d'éviter le coûteux calcul de l'état adjoint. Cette approche a été largement utilisée pour divers problèmes d'écoulement ([20], [21], [56]) et c'est celle que l'on a retenue ici. Cependant les résultats obtenus sont peu satisfaisants et on essaiera d'expliquer pourquoi dans cette thèse.

Méthodes de calcul du gradient

Différences finies

C'est la méthode la plus simple pour calculer le gradient d'une fonction coût, mais pas la moins coûteuse en termes de temps de calcul. Elle nécessite en effet de nombreuses évaluations de la fonction objectif, mais aucune évaluation du gradient.
A l'aide d'un développement de Taylor, on obtient une formule décentrée (ici à droite) :

$$\frac{\partial J(x_C)}{\partial x_{C,j}} \simeq \frac{J(x_C + \varepsilon e_j) - J(x_C)}{\varepsilon},$$

ou la formule centrée :

$$\frac{\partial J(x_C)}{\partial x_{C,j}} \simeq \frac{J(x_C + \varepsilon e_j) - J(x_C - \varepsilon e_j)}{2\varepsilon},$$

où $x_{C,j}$ désigne la j-ième composante du vecteur x_C. Dans le premier cas l'erreur engendrée est en $O(\varepsilon)$ et dans le deuxième en $O(\varepsilon^2)$.
Une difficulté classique de cette méthode réside dans le choix de ε. Par ailleurs, pour un problème avec un grand nombre de variables de contrôle, elle devient très coûteuse en temps de calcul.

Différentiation automatique

De nombreux ouvrages et articles portant sur l'optimisation de forme traitent de cette méthode ou l'utilisent ([40], [46], [47]). Elle nécessite d'avoir accès au code source permettant de calculer la fonction coût.

Mode direct Lorsque la fonction à différentier est une sortie du code, son gradient peut être calculé automatiquement par différentiation ligne à ligne du code. La fonction et le gradient sont donc calculés simultanément.

Mode inverse Le mode inverse consiste à écrire un Langrangien en associant une variable adjointe à chaque ligne du programme, ainsi considérée comme une contrainte, sauf pour celle associée à la fonction objectif. Par substitution inverse on détermine les multiplicateurs, puis le gradient du Lagrangien, et enfin celui de la fonction objectif.
Un des principaux inconvénients de la différentiation automatique est qu'elle demande beaucoup de mémoire.

Applications

Dans [47], les auteurs présentent dans un chapitre introductif un certain nombre de problèmes d'optimisation de forme :

- la minimisation du poids de structures,
- la réduction de la traînée d'une aile d'avion,
- la conception d'avions moins détectables par radar,
- l'amélioration des digues coupe-vagues à l'entrée d'un port...

L'ouvrage se poursuit par la présentation des moyens théoriques et numériques pouvant ou étant mis en œuvre dans des problèmes d'optimisation de forme, en particulier la différentiation automatique de programmes. Celle-ci est utilisée ici pour éviter d'avoir à calculer le gradient exact de la fonction coût pour des équations d'état aussi complexes que Navier-Stokes avec modèle de turbulence. Enfin des résultats numériques sont donnés pour différents problèmes d'optimisation de forme en 2D ou 3D, avec

contraintes, notamment pour des ailes d'avion et des avions entiers.
Sur le même principe, [49] est une revue de quelques-uns des développements récents de la discipline.
L'objectif de [40] est de fournir aux lecteurs les méthodes et les outils numériques nécessaires à l'optimisation de forme. Une fois de plus c'est la différentiation automatique qui est utilisée. Les applications présentées sont des problèmes inverses en diffusion, la réduction de la traînée et l'optimisation de la stabilité aéroélastique d'une pale de turbine.

Dans [46], B. Mohammadi utilise la différentiation automatique en mode inverse dans le cas de problèmes 2D et 3D de minimisation de la traînée d'une aile pour des écoulements turbulents ou laminaires.
Dans [20] et [21], les auteurs présentent un algorithme d'optimisation de forme en 3D basé sur une méthode d'optimisation Broyden-Fletcher-Goldfarb-Shanno (BFGS). La différentiation automatique est menée avec un calcul de gradient incomplet. L'utilisation d'un gradient incomplet permet une économie de mémoire considérable dans la mise en œuvre de la différentiation automatique. Une application à l'optimisation d'ailes en régime transonique et supersonique est proposée. L'utilisation d'un gradient incomplet entraîne que le Hessien approché calculé dans le BFGS est inexact. Cependant la méthode converge.
Dans ses travaux de thèse [41], E. Leclerc développe un algorithme de contrôle pour des écoulements aérodynamiques instationnaires. En utilisant des gradients incomplets, l'auteur évite la détermination de l'état adjoint et réduit le temps de calcul. Par ailleurs il retient une méthode de descente à pas fixe.
Une partie des applications étudiées par M. Stanciu dans sa thèse [57] portent sur la minimisation de la traînée pour le profil d'ailes en situation d'atterrissage-décollage.

Etudes spécifiques aux turbomachines

M. Stanciu [57] étudie aussi des applications en turbomachinerie avec l'optimisation des pales d'un ventilateur pour le refroidissement d'un moteur. Une méthode de gradients incomplets est utilisée. Différentes méthodes de gradient sont testées. La méthode BFGS donne de bons résultats. Différentes aubes sont optimisées, en 2D et en 3D.
S. Pierret dans [52] et [53] utilise un réseau de neurones artificiel pour construire une fonction coût approchée. Il teste plusieurs méthodes d'optimisation : méthode de gradient, algorithme génétique et recuit simulé. Au regard du nombre d'évaluations de la fonction coût, ces deux dernières méthodes sont bien meilleures que la première. De plus, explorant un plus grand ensemble de formes, ces méthodes ont plus de chances de mener à un optimum global. Plusieurs aubes de turbines et de compresseurs, paramétrées par des courbes de Bézier, ont été optimisées, pour plusieurs régimes d'écoulements et avec différents types de contraintes. Les contraintes sont converties en termes de pénalisation incorporés à la fonction objectif. Ainsi le problème d'optimisation considéré est sans contrainte.
[25] et [37] suivent la même démarche que S. Pierret avec un réseau de neurones artificiel et un algorithme génétique, respectivement pour l'optimisation des aubes d'un rotor de turbine et pour une aube de stator de turbine.
[55] présente les résultats d'un algorithme propre, on cite : "POISK, (...) la combinaison d'une méthode de gradient et d'une méthode stochastique" pour l'optimisation des aubes d'une turbine.

Résolution du problème d'optimisation de la pompe de fond de puits

Depuis quelques années, une régulière activité de simulation numérique des pompes de fond de puits a été développée à l'IFP ([2], [6], [7], [28], [60]). Le logiciel de simulation et calcul de mécanique des fluides Fluent a généralement été employé pour ces modélisations. De façon à utiliser et à maintenir à jour ces acquis, la décision a été prise de réaliser l'optimisation de forme de la pompe avec ce même logiciel. De ce fait, un certain nombre de choix s'imposaient pour mener à bien la résolution du problème

d'optimisation.
Le premier inconvénient à prendre en compte lorsque l'on utilise un logiciel tel que Fluent est le temps de calcul : une évaluation de la fonction coût pour une forme donnée peut prendre de quelques minutes à plusieurs heures selon la complexité du cas étudié. Les méthodes non-déterministes, nécessitant un grand nombre d'évaluations, ont donc été écartées d'emblée.
Etant donné que Fluent est une "boîte noire", on ne peut pas résoudre numériquement un état adjoint avec cet outil et il a donc fallu renoncer à cette méthode.
Pour les raisons de temps de calcul évoquées précédemment, il est inenvisageable de calculer le gradient de la fonction coût par différences finies.
Enfin, la différentiation automatique nécessite d'avoir accès au code source de calcul de la fonction coût, elle ne peut donc pas être utilisée ici.
On a donc naturellement décidé de retenir une méthode de gradient incomplet. On ne procède pas exactement comme il a été présenté précédemment. En effet on commence par écrire l'expression analytique complète du gradient. Ensuite on ne conserve de cette expression que les termes que l'on sait déterminer par le calcul ou *via* le logiciel Fluent et on supprime les termes jugés inaccessibles. Bien qu'incomplet, ce gradient devrait néanmoins donner une direction de descente (ou de montée dans le cas présent) pour un algorithme de minimisation (maximisation). Tout l'enjeu est alors de savoir si cette direction est toujours significative.

Contenu du mémoire de thèse

Dans la **première partie** de ce mémoire on étudie la modélisation de la géométrie et de la physique du problème.

Le **chapitre 1** est consacré à la paramétrisation de la géométrie de la pompe, réalisée avec des B-splines paramétriques cubiques 2D. On présente les B-splines, ainsi que quelques-unes de leurs propriétés, en rapport avec certaines des contraintes que l'on souhaite imposer à la géométrie. Par ailleurs, on met en place une méthode de recherche d'un nombre raisonnable de points de contrôle de B-spline pour approcher la géométrie de référence de la pompe de fond de puits.

Dans le **chapitre 2** on s'intéresse à la modélisation de l'écoulement du gasoil dans la pompe. On présente succintement le logiciel Fluent que l'on a choisi d'utiliser, puis on s'attarde davantage sur les méthodes numériques retenues pour la résolution des équations de Navier-Stokes avec turbulence dans le cadre du problème considéré.

Enfin, dans le **chapitre 3**, on étudie l'influence sur l'écoulement du caractère périodique de la géométrie de la pompe. On montre deux résultats de convergence dans une géométrie périodique, l'un pour un problème semi-linéaire en dimension quelconque, l'autre pour le problème de Stokes en dimension 2, avec conditions aux bords de Navier.

La **deuxième partie** de ce mémoire est consacrée au calcul d'une expression incomplète du gradient de la fonction coût du problème d'optimisation.

Dans le **chapitre 4** on utilise une formule d'intégration sur des bords variables pour calculer l'expression analytique du gradient, puis on écarte les termes que l'on ne peut pas calculer avec les moyens numériques dont on dispose. On obtient ainsi une expression incomplète du gradient contenant des termes fournis par Fluent et des termes géométriques que l'on peut calculer. On donne les premiers éléments de géométrie différentielle pour ce calcul.

Dans les **chapitres 5 et 6** on développe les calculs permettant d'obtenir, pour chacune des surfaces délimitant la géométrie de la pompe, l'expression explicite du champ de déformation induit par le déplacement d'un point de contrôle, de la dérivée de l'extension de la normale et de la courbure moyenne, entre autres termes d'ordre géométrique requis dans l'expression du gradient incomplet.

Pour l'optimisation de la pompe de fond de puits on a d'abord développé un code pour l'optimisation isolée du moyeu de la pompe. Les détails de ce développement ainsi que les résultats numériques de l'optimisation font l'objet de la **troisième partie** de ce mémoire.

Le **chapitre 7** présente les points d'importance dans la mise en œuvre de l'optimisation. On y aborde d'abord la question de l'existence d'un minimum, puis celle de l'évaluation numérique de la fonction coût, de son gradient incomplet, et des dérivées des variables d'état. Le déroulement de l'algorithme d'optimisation est détaillé. On présente ensuite plusieurs méthodes d'optimisation sans contrainte dans le cadre général, puis on évoque, plus particulièrement pour l'optimisation du moyeu seul, la prise en compte des contraintes.

Dans le **chapitre 8** on donne les résultats numériques de l'algorithme d'optimisation avec gradient incomplet pour le problème du moyeu seul. On utilise une méthode de gradient à pas fixe sans contrainte. Des contraintes d'égalité linéaires sont cependant prises en compte par réduction du problème. On procède ensuite à une analyse critique de la méthode, notamment par comparaison aux résultats obtenus grâce à une méthode de différences finies, ou à l'aide de tests dans le cas particulier d'un moyeu cylindrique, ou enfin sur la base de l'expérience acquise dans des travaux antérieurs portant sur les sensitivités incomplètes.
On conclut sur quelques perspectives de recherche.

Première partie

Modélisation du problème

Chapitre 1

Paramétrisation et propriétés de la géométrie

1.1 Quelques propriétés des B-splines cubiques

L'objectif des B-splines est de définir des courbes en assemblant des segments de courbes polynomiales. Ce type de représentation confère à la courbe obtenue des propriétés locales : une perturbation d'un coefficient de polynôme produit une déformation locale, sans que la courbe complète soit altérée. Plus le degré des polynômes est bas et plus facile en est la prise en charge numérique. En revanche la représentation d'une courbe complexe devient plus difficile, en particulier si on ne souhaite pas multiplier les segments de courbes polynomiales. Choisir les B-splines cubiques, c'est-à-dire une représentation par des polynômes du troisième degré, est généralement un bon compromis dans les problèmes de design. C'est le choix qui a été fait dans cette thèse, compte tenu des considérations précédentes ainsi que de certaines propriétés intéressantes des B-splines qui seront énoncées dans ce paragraphe.

1.1.1 Définitions et premières propriétés des B-splines cubiques

De nombreux ouvrages sont consacrés aux B-splines et à leur emploi en modélisation. On pourra consulter, par exemple, [17] ou [26].

B-spline unidimensionnelle

Une B-spline cubique est définie comme une combinaison linéaire des *fonctions de base* suivantes, pour $t \in [0,1]$:

$$\begin{aligned}
p_0(t) &= (t^3)/6,\\
p_1(t) &= (-3t^3+3t^2+3t+1)/6,\\
p_2(t) &= (3t^3-6t^2+4)/6,\\
p_3(t) &= (-t^3+3t^2-3t+1)/6.
\end{aligned}$$

Les coefficients de ces combinaisons linéaires sont appelés *points de contrôle*.
Soit $(x_i)_{i=1\ldots n}$, où $n \geq 4$, un ensemble de réels. On définit la B-spline $x(t)$ à partir des points de contrôle $(x_i)_{i=1\ldots n}$ de la façon suivante :

$$\forall i \in \{1,\ldots,n-3\}, \forall t \in [i-1,i], x(t) = \sum_{j=0}^{3} x_{i+j} p_{3-j}(t-i+1).$$

La B-spline précédente est *uniforme*, c'est-à-dire que le paramètre t varie dans des intervalles qui sont tous de longueur égale à 1. Dans ce mémoire on n'étudie et n'utilise que de telles B-splines. La B-spline $x(t)$ est aussi une B-spline *unidimensionnelle* : il s'agit d'une application de $\mathbb{R}$ dans $\mathbb{R}$, à la différence des B-splines paramétriques présentées dans ce qui suit.

B-spline cubique paramétrique dans $\mathbb{R}^2$

Une B-spline cubique *paramétrique uniforme* est une application de $\mathbb{R}$ dans $\mathbb{R}^2$ de la forme $(x(t), y(t))$ où $x(t)$ et $y(t)$ sont des B-splines unidimensionnelles uniformes. On appelle *points de contrôle* de cette B-spline cubique dans $\mathbb{R}^2$ les points $(x_i, y_i)_{i=1\ldots n, n\geq 4}$ où les $(x_i)_{i=1\ldots n}$ et $(y_i)_{i=1\ldots n}$ sont les points de contrôle de $x(t)$ et de $y(t)$.

Soit $(x_i, y_i)_{i=1\ldots n}$ un ensemble de points de $\mathbb{R}^2$. On définit une B-spline $(x(u), y(u))$ à partir des points de contrôle $(x_i, y_i)_{i=1\ldots n}$ de la façon suivante :

$$\forall i \in \{1, \ldots, n-3\}, \forall u \in [i-1, i], \begin{cases} x(u) = \sum_{j=0}^{3} x_{i+j} p_{3-j}(u-i+1), \\ y(u) = \sum_{j=0}^{3} y_{i+j} p_{3-j}(u-i+1). \end{cases}$$

La B-spline est donc constituée de $n-3$ segments ou *intervalles* polynomiaux.

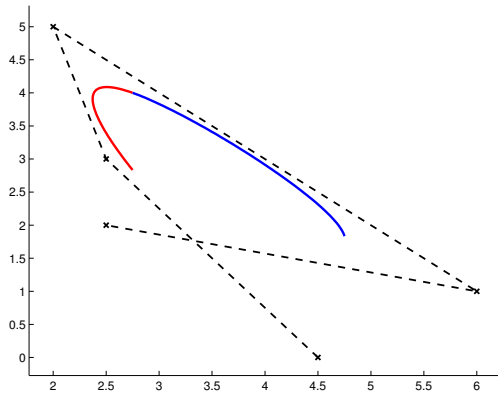

FIG. 1.1: *B-spline à $n = 5$ points de contrôle (croix noires reliées entre elles). Ses 2 intervalles sont différenciés par leur couleur.*

Il peut être plus facile numériquement de considérer que cette même B-spline est l'assemblage de $n-3$ intervalles de B-spline $(x_i(t), y_i(t))$ définis par :

$$\forall i \in \{1, \ldots, n-3\}, \begin{cases} x_i(t) = \sum_{j=0}^{3} x_{i+j} p_{3-j}(t), \\ y_i(t) = \sum_{j=0}^{3} y_{i+j} p_{3-j}(t), \end{cases} \quad \text{pour } t \in [0, 1].$$

On vérifie aisément qu'une B-spline cubique est de classe $\mathcal{C}^2$ aux jonctions entre les intervalles, ce qui est une propriété intéressante dans le cadre industriel puisque les formes considérées y sont souvent régulières.

1.1.2 Propriété de monotonie

On étudie une B-spline unidimensionnelle $x(t)$ de points de contrôle $(x_i)_{i=1\ldots n}$.

Théorème 1.1.1. *Si la suite $(x_i)_{i=1\ldots n}$ est croissante, alors la B-spline $x(t)$ admettant les x_i pour points de contrôle est croissante.*

Preuve. On considère le k-ième intervalle $x_k(t)$ de la B-spline. Pour plus de simplicité on note $x(t) = x_k(t)$, $t \in [0,1]$ et $(x_j)_{j=1\ldots 4}$, les points de contrôle permettant de définir $x(t)$. Ainsi :

$$x(t) = x_1 p_3(t) + x_2 p_2(t) + x_3 p_1(t) + x_4 p_0(t),\ t \in [0,1].$$

On étudie les variations de x sous l'hypothèse que $x_1 \leq x_2 \leq x_3 \leq x_4$. Le calcul des dérivées successives de x donne notamment :

$$x'(t) = \frac{1}{2}(-x_1 + 3x_2 - 3x_3 + x_4)t^2 + (x_1 - 2x_2 + x_3)t + \frac{1}{2}(-x_1 + x_3),$$

$$x''(t) = (-x_1 + 3x_2 - 3x_3 + x_4)t + (x_1 - 2x_2 + x_3),$$

$$x''(0) = x_1 - 2x_2 + x_3,$$

$$x''(1) = x_2 - 2x_3 + x_4,$$

$$x'(0) = (x_3 - x_1)/2 \geq 0,$$

$$x'(1) = (x_4 - x_2)/2 \geq 0.$$

1er cas (trivial) : $-x_1 + 3x_2 - 3x_3 + x_4 = 0$
Dans ce cas x' est monotone et comme $x'(0) \geq 0$ et $x'(1) \geq 0$, x' est positive et x croissante.
2ème cas : $-x_1 + 3x_2 - 3x_3 + x_4 < 0$
Dans ce cas x'' est décroissante. Si $0 \geq x''(0) \geq x''(1)$, alors x'' est négative, donc x' décroissante. Comme $x'(1) \geq 0$, x' est positive et donc x est croissante. Dans les cas où $x''(0) \geq 0 \geq x''(1)$ et $x''(0) \geq x''(1) \geq 0$, il est aussi facile de prouver que x' est positive et donc x croissante.
3ème cas : $-x_1 + 3x_2 - 3x_3 + x_4 > 0$
Dans ce cas x'' est croissante. Dans les cas $0 \geq x''(1) \geq x''(0)$ et $x''(1) \geq x''(0) \geq 0$, il est facile de prouver que x' est monotone positive et donc x croissante.
Reste le cas où $x''(1) > 0 > x''(0)$. Dans ce cas x'' s'annule en un point unique $t_0 \in [0,1]$ et x' atteint son minimum en t_0. Il faut déterminer le signe de $x'(t_0)$. On a :

$$t_0 = \frac{-(x_1 - 2x_2 + x_3)}{-x_1 + 3x_2 - 3x_3 + x_4} := \frac{N}{D},$$

avec $N = -x''(0) > 0$ et $D > 0$ par hypothèse. Avec ces notations,

$$x'(t) = \frac{1}{2}Dt^2 - Nt + \frac{1}{2}(x_3 - x_1) \text{ pour tout } t \in [0,1].$$

Par suite on calcule :

$$x'(t_0) = x'\left(\frac{N}{D}\right) = \frac{1}{2}\left((x_3 - x_1) - \frac{N^2}{D}\right).$$

Or $(x_3 - x_1) \geq (x_2 - x_1) \geq (x_2 - x_1) - (x_3 - x_2) = N > 0$. De plus, $1 \geq t_0 = \frac{N}{D} > 0$ donc $N \geq \frac{N^2}{D}$. On en déduit successivement que $x'(t_0)$ est positif, que x' est positive sur $[0,1]$ et enfin que x est croissante sur $[0,1]$. □

1.1.3 Condition suffisante pour qu'une B-spline n'ait pas de point d'inflexion

Soit une B-spline cubique paramétrique $(x(t), y(t))$. On considère le premier intervalle de la B-spline $(x_1(t), y_1(t))$ avec $t \in [0,1]$. On note $V_j = (x_j, y_j)$ pour $j = 1, ..., 4$ les points de contrôle permettant de définir $(x_1(t), y_1(t))$. Sans restreindre la généralité, on peut prendre $V_1 = (0,0)$ et $V_2 = (1,0)$. On a alors :

$$x_1(t) = p_2(t) + x_3 p_1(t) + x_4 p_0(t),$$

$$y_1(t) = y_3 p_1(t) + y_4 p_0(t),$$

avec $t \in [0,1]$. La courbe ne présente pas de point d'inflexion sur l'intervalle considéré si et seulement si $\forall t \in [0,1]$, $\phi(t) := y_1''(t)x_1'(t) - y_1'(t)x_1''(t)$ est de signe constant.

Théorème 1.1.2. *Si le quadrilatère $V_1V_2V_3V_4$ est convexe, alors l'intervalle de B-spline admettant les sommets V_1, V_2, V_3, V_4 pour points de contrôle n'a pas de point d'inflexion.*

Preuve. En développant l'expression de $\phi(t)$, on trouve :

$$2\phi(t) = (-x_4y_3 + x_3y_4 + 3y_3 - 2y_4)t^2 + (x_3y_4 - x_4y_3 - 3y_3)t + 2y_3.$$

On suppose que le quadrilatère $V_1V_2V_3V_4$ est convexe. Ceci équivaut à ce que

$$\overrightarrow{V_1V_3} = \alpha\overrightarrow{V_1V_2} + \beta\overrightarrow{V_1V_4} \text{ avec } \alpha \geq 0,\ \beta \geq 0 \text{ et } \alpha + \beta \geq 1. \tag{1.1}$$

Soit $V_2' = (0,1)$. Si $\overrightarrow{V_1V_4} = a\overrightarrow{V_1V_2} + b\overrightarrow{V_1V_2'}$, on a $\overrightarrow{V_1V_3} = (\alpha + a\beta)\overrightarrow{V_1V_2} + b\beta\overrightarrow{V_1V_2'}$. Alors les expressions

$$\phi(0) = \overrightarrow{V_1V_2} \wedge \overrightarrow{V_1V_3} = b\beta,$$

$$\phi(1) = \overrightarrow{V_2V_3} \wedge \overrightarrow{V_2V_4} = b(\alpha + \beta - 1)$$

et

$$\overrightarrow{V_1V_3} \wedge \overrightarrow{V_1V_4} = b\alpha$$

sont toutes du signe de b et il suffit d'étudier le cas $b \geq 0$.
Sachant que la dérivée de ϕ est une fonction affine, il apparaît clairement que $\phi(t) \geq 0$ dans les cas suivants : $\{\phi'(0) \geq 0 \text{ et } \phi'(1) \geq 0\}$, $\{\phi'(0) \geq 0 \text{ et } \phi'(1) \leq 0\}$ et $\{\phi'(0) \leq 0 \text{ et } \phi'(1) \leq 0\}$. Reste à étudier le cas où $\phi'(0) \leq 0$ et $\phi'(1) \geq 0$. Dans ce cas ϕ atteint son minimum en l'unique point $t_0 \in [0,1]$ tel que $\phi'(t_0) = 0$.
On cherche donc à déterminer le signe de $\phi(t_0)$. En développant cette expression, on trouve que $\phi(t_0)$ est du signe de $S = 8y_3(-x_4y_3 + x_3y_4 + 3y_3 - 2y_4) - (x_3y_4 - x_4y_3 - 3y_3)^2$.
On pose $d = x_3y_4 - x_4y_3 = \overrightarrow{V_1V_3} \wedge \overrightarrow{V_1V_4}$. Le quadrilatère $V_1V_2V_3V_4$ étant convexe, et $b \geq 0$, on a $d = b\alpha \geq 0$.
De plus $\phi'(0) \leq 0$ peut s'écrire $3y_3 - d \geq 0$ et $\phi'(1) \geq 0$ peut s'écrire $3d + 3y_3 - 4y_4 \geq 0$.
En développant l'expression de S, on trouve :

$$S = 4y_3(3d + 3y_3 - 4y_4) + (3y_3 - d)(y_3 + d).$$

D'après les inégalités précédemment énoncées et le fait que $y_3 \geq 0$ et $d \geq 0$, il vient $S \geq 0$. Par conséquent $\phi(t_0) \geq 0$ et donc ϕ est positive sur $[0,1]$. □

On considère une B-spline avec un nombre de points de contrôle $n > 4$. Le théorème 1.1.2 prouve simplement que si les quadrilatères $(V_iV_{i+1}V_{i+2}V_{i+3})_{i=1...n-3}$ sont convexes, alors sur chacun de ses intervalles la B-spline n'a pas de point d'inflexion. La question est ensuite de trouver une condition suffisante pour qu'il n'y ait pas de point d'inflexion aux jonctions entre intervalles.

Théorème 1.1.3. *Soient $V_1V_2V_3V_4$ et $V_2V_3V_4V_5$ deux quadrilatères convexes. La B-spline admettant les sommets V_1, V_2, V_3, V_4 et V_5 pour points de contrôle n'a pas de point d'inflexion si*

- *$\overrightarrow{V_2V_3}$ et $\overrightarrow{V_2V_4}$ ne sont pas colinéaires*

ou si

- *$\overrightarrow{V_2V_3}$ et $\overrightarrow{V_2V_4}$ sont colinéaires et $\overrightarrow{V_2V_4} \wedge \overrightarrow{V_2V_1}$ et $\overrightarrow{V_2V_4} \wedge \overrightarrow{V_4V_5}$ sont du même signe (c'est-à-dire que les quadrilatères $V_1V_2V_3V_4$ et $V_2V_3V_4V_5$ ont la même orientation).*

Preuve. On considère le même quadrilatère convexe $V_1V_2V_3V_4$ que précédemment, et on garde toutes les notations que l'on a introduites. Soit $(x_2(t), y_2(t))$ avec $t \in [0,1]$ l'intervalle de B-spline généré par V_2, V_3, V_4 et V_5. La B-spline constituée des intervalles $(x_1(t), y_1(t))$ et $(x_2(t), y_2(t))$ ne présente pas de point d'inflexion si et seulement si $\forall t \in [0,1]$, $\psi(t) := y_2''(t)x_2'(t) - y_2'(t)x_2''(t)$ et $\phi(t)$ sont constamment du même signe.
Pour prouver ce résultat il suffit de montrer que $\psi(0)$, $\psi(1)$ et $\overrightarrow{V_2V_4} \wedge \overrightarrow{V_2V_5}$ sont du même signe que b puis le reste de la démonstration suit comme dans la preuve du théorème 1.1.2.

- On suppose tout d'abord $\overrightarrow{V_2V_3}$ et $\overrightarrow{V_2V_4}$ colinéaires et $\overrightarrow{V_2V_4} \wedge \overrightarrow{V_2V_1}$ et $\overrightarrow{V_2V_4} \wedge \overrightarrow{V_4V_5}$ du même signe.

Le quadrilatère $V_2V_3V_4V_5$ est convexe donc il existe $A \geq 1$ tel que $\overrightarrow{V_2V_4} = A\overrightarrow{V_2V_3}$. Alors on calcule :

$$\begin{aligned}
\psi(0) &= \overrightarrow{V_2V_3} \wedge \overrightarrow{V_2V_4} = 0, \\
\psi(1) &= \overrightarrow{V_3V_4} \wedge \overrightarrow{V_3V_5} \\
&= \left(\overrightarrow{V_3V_2} + \overrightarrow{V_2V_4}\right) \wedge \overrightarrow{V_4V_5} \\
&= \left(-\frac{1}{A} + 1\right) \overrightarrow{V_2V_4} \wedge \overrightarrow{V_4V_5},
\end{aligned}$$

et

$$\overrightarrow{V_2V_4} \wedge \overrightarrow{V_2V_5} = \overrightarrow{V_2V_4} \wedge \overrightarrow{V_4V_5}.$$

Ainsi $\psi(0)$, $\psi(1)$ et $\overrightarrow{V_2V_4} \wedge \overrightarrow{V_2V_5}$ sont du même signe que $\overrightarrow{V_2V_4} \wedge \overrightarrow{V_4V_5}$ et donc que $\overrightarrow{V_2V_4} \wedge \overrightarrow{V_2V_1}$ par hypothèse.
Or

$$\overrightarrow{V_2V_4} \wedge \overrightarrow{V_2V_1} = \overrightarrow{V_1V_4} \wedge \overrightarrow{V_2V_1} = b,$$

ce qu'il fallait montrer.

- On suppose à présent que $\overrightarrow{V_2V_3}$ et $\overrightarrow{V_2V_4}$ ne sont pas colinéaires.

Le quadrilatère $V_2V_3V_4V_5$ est convexe donc :

$$\overrightarrow{V_2V_4} = A\overrightarrow{V_2V_3} + B\overrightarrow{V_2V_5} \text{ avec } A \geq 0,\ B > 0 \text{ et } A + B \geq 1. \tag{1.2}$$

Le calcul montre que :

$$\psi(0) = \overrightarrow{V_2V_3} \wedge \overrightarrow{V_2V_4} = b(\alpha + \beta - 1),$$

$$\psi(1) = \overrightarrow{V_3V_4} \wedge \overrightarrow{V_3V_5} = \frac{b}{B}(\alpha + \beta - 1)(A + B - 1)$$

et

$$\overrightarrow{V_2V_4} \wedge \overrightarrow{V_2V_5} = \frac{A}{B} b(\alpha + \beta - 1)(A + B - 1).$$

Chacune de ces trois expressions est donc du signe de b, ce qu'il fallait montrer. □

En conclusion, l'absence de point d'inflexion ou la "convexité" que l'on souhaite imposer à certaines B-splines génératrices de la géométrie de la pompe pourront donc l'être pas le biais de contraintes d'inégalité non-linéaires traduisant le théorème 1.1.3, du type de (1.1) et (1.2).

1.2 Quelques propriétés géométriques de l'intrados de la pompe

L'objet de cette section est de déterminer certaines propriétés géométriques de l'intrados d'une pompe de façon à mieux appréhender certaines des contraintes géométriques énoncées dans l'introduction.

1.2.1 Comportement local de l'intrados à l'attaque

Proposition 1.2.1. *La courbe de l'intrados est concave à l'attaque.*

Preuve. On se place dans le repère défini par la tangente et la normale à la cambrure au point d'attaque. Dans ce repère, on définit la cambrure par son équation cartésienne : $y = f_c(x)$, de même pour l'extrados et l'intrados : $y = f_e(x)$ et $y = f_i(x)$. Il existe un réel $\varepsilon > 0$ tel que f_c, f_e et f_i soient de classe $\mathcal{C}^2$ sur $]0,\varepsilon[$. On note aussi $(x_e(t), y_e(t))$ la B-spline de l'extrados.
Dans le repère choisi, $x_e(0) = 0$, $y_e(0) = 0$, $x_e'(0) = 0$ et $y_e'(0) < 0$. Par ailleurs, si la contrainte est respectée, l'extrados est convexe jusqu'à l'origine donc $y_e''(0)x_e'(0) - y_e'(0)x_e''(0) \geq 0$, d'où $x_e''(0) \geq 0$. On a par conséquent :

$$\begin{cases} x_e(t) = a_1 t^2 + a_2 t^3, \\ y_e(t) = -b_1 t + b_2 t^2 + b_3 t^3, \end{cases}$$

avec $a_1 \geq 0$, $b_1 > 0$ et $t \in [0,1]$.

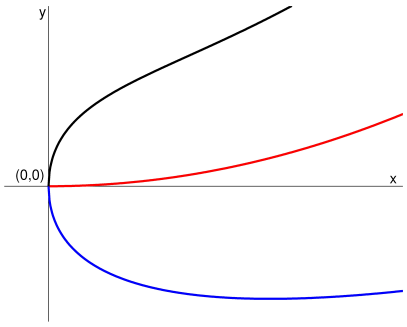

FIG. 1.2: *Cambrure (rouge), extrados (bleu) et intrados (noir) au voisinage de l'origine du repère qui vient d'être défini.*

Soit x l'abscisse d'un point de l'extrados, dans le voisinage de l'origine.

- On suppose $a_1 > 0$. Alors

$$\frac{x}{a_1} = t^2 + \frac{a_2}{a_1} t^3. \tag{1.3}$$

On définit ε tel que

$$t = \left(\frac{x}{a_1}\right)^{1/2} + x^{1/2}\varepsilon(x) + o(x).$$

Alors

$$t^2 = \frac{x}{a_1} + \frac{2x\varepsilon(x)}{\sqrt{a_1}} + x\varepsilon(x)^2 + o(x^2), \text{ et } t^3 = \left(\frac{x}{a_1}\right)^{3/2} + \frac{3x^{3/2}\varepsilon(x)}{a_1} + \frac{3x^{3/2}\varepsilon(x)^2}{\sqrt{a_1}} + x^{3/2}\varepsilon(x)^3 + o(x^3).$$

En reportant ces expressions de t^2 et t^3 dans (1.3), on détermine que

$$\varepsilon(x) = -\frac{a_2}{2a_1^2}x^{1/2} \text{ et } t = \left(\frac{x}{a_1}\right)^{1/2} - \frac{a_2 x}{2a_1^2} + O(x^2).$$

Par suite

$$f_e(x) = -\frac{b_1 x^{1/2}}{a_1^{1/2}} + \frac{b_1 a_2 x}{2a_1^2} + \frac{b_2 x}{a_1} + o(x). \tag{1.4}$$

- Si maintenant on suppose $a_1 = 0$, alors on a immédiatement $t = \left(\frac{x}{a_2}\right)^{1/3}$ et

$$f_e(x) = -\frac{b_1 x^{1/3}}{a_2^{1/3}} + \frac{b_2 x^{2/3}}{a_2^{2/3}}. \tag{1.5}$$

La cambrure est convexe à l'origine et on a $x_c(0) = 0$, $y_c(0) = 0$, $y_c'(0) = 0$, d'où le développement limité à l'origine $f_c(x) = c_1 x^2 + o(x^2)$.
Soit x_0 l'abscisse du point de la cambrure, projeté orthogonal d'un point de l'extrados $(x_1, f_e(x_1))$ situé dans le voisinage de l'origine. Par construction, le vecteur $(x_1 - x_0, f_e(x_1) - f_c(x_0))$ est colinéaire au vecteur normal à la cambrure $(-f_c'(x_0), 1)$, donc $(x_1 - x_0) + f_c'(x_0)(f_e(x_1) - f_c(x_0)) = 0$. En posant $\alpha = c_1 f_e(x_1)$, on a $x_1 = x_0 + \alpha x_0^2 + o(x_0^2)$.
Le symétrique de $(x_1, f_e(x_1))$ par rapport à $(x_0, f_c(x_0))$ est le point de l'intrados

$$(2x_0 - x_1, f_i(2x_0 - x_1)) = (2x_0 - x_1, 2f_c(x_0) - f_e(x_1)).$$

D'après ce qui précède, l'égalité $f_i(2x_0 - x_1) = 2f_c(x_0) - f_e(x_1)$ s'écrit $f_i(x_0 - \alpha x_0^2 + o(x_0^2)) = 2f_c(x_0) - f_e(x_0 + \alpha x_0^2 + o(x_0^2))$. On dérive deux fois cette égalité et on trouve :

$$(1 - 2\alpha x_0 + o(x_0)) f_i'(x_0 - \alpha x_0^2 + o(x_0^2)) = 2f_c'(x_0) - (1 + 2\alpha x_0 + o(x_0)) f_e'(x_0 + \alpha x_0^2 + o(x_0^2)) \tag{1.6}$$

et

$$\begin{aligned}(1 - 2\alpha x_0 + o(x_0))^2 f_i''(x_0 - \alpha x_0^2 + o(x_0^2)) - 2\alpha f_i'(x_0 - \alpha x_0^2 + o(x_0^2)) \\ = 2f_c''(x_0) - (1 + 2\alpha x_0 + o(x_0))^2 f_e''(x_0 + \alpha x_0^2 + o(x_0^2)) - 2\alpha f_e'(x_0 + \alpha x_0^2 + o(x_0^2)).\end{aligned} \tag{1.7}$$

Dans le cas où $a_1 > 0$, en reportant (1.4) dans (1.6), on détermine que $f_i'(x) \sim \frac{b_1 x^{-1/2}}{2a_1^{1/2}}$ au voisinage de l'origine. Puis avec (1.7) on montre que $f_i''(x) \sim -\frac{3b_1 x^{-3/2}}{2a_1^{1/2}}$ au voisinage de l'origine.
Dans le cas où $a_1 = 0$, en reportant (1.5) dans (1.6), on détermine que $f_i'(x) \sim \frac{b_1 x^{-2/3}}{3a_2^{1/3}}$ au voisinage de l'origine. Puis avec (1.7) on montre que $f_i''(x) \sim -\frac{2b_1 x^{-5/3}}{9a_2^{1/3}}$ au voisinage de l'origine. On en déduit, dans les deux cas, qu'au voisinage de l'origine $f_i'' < 0$. □

1.2.2 Comportement local de l'intrados à la fuite

L'étude du comportement de l'intrados à la fuite permet aussi d'en savoir plus sur son nombre de points d'inflexion. En effet, on a vu que l'intrados est concave à l'attaque. Si on connaît son comportement à la fuite, on aura un renseignement sur la parité du nombre de points d'inflexion. Si à la fuite l'intrados est concave alors ce nombre est pair (vraisemblablement égal à 0 ou 2), si l'intrados est convexe, le nombre est impair (vraisemblablement égal à 1).

Proposition 1.2.2. *La courbe de l'intrados est concave et sous sa tangente à la fuite si la courbure de l'extrados est suffisamment grande devant celle de la cambrure.*

Preuve. On se place dans le repère défini par la tangente et la normale à la cambrure au point de fuite. Dans ce repère, on définit la cambrure par son équation cartésienne : $y = f_c(x)$. On fait de même pour l'extrados : $y = f_e(x)$. Au voisinage de l'origine du repère on peut écrire les développements limités suivants :

$$f_c(x) = \frac{a}{2}x^2 + o(x^2) \tag{1.8}$$

$$f_e(x) = bx + \frac{c}{2}x^2 + o(x^2) \tag{1.9}$$

où $a \geq 0$ car la cambrure est convexe, $b \geq 0$ et $c \geq 0$ car l'extrados est croissant et convexe.

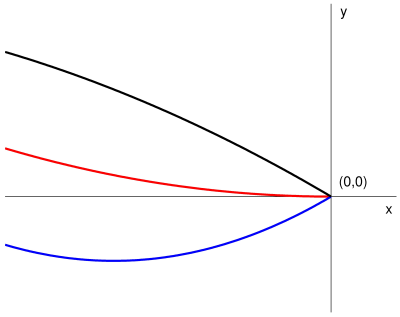

FIG. 1.3: *Cambrure (rouge), extrados (bleu) et intrados (noir) au voisinage de l'origine du repère qui vient d'être défini.*

Au voisinage de la fuite, en x_0, la normale n_c à la cambrure admet pour vecteur directeur

$$\overrightarrow{n_c} = (1 + a^2x_0^2)^{-1/2}(-ax_0, 1)$$

et pour équation

$$Y = -\frac{1}{ax_0}X + \frac{ax_0^2}{2} + \frac{1}{a}. \tag{1.10}$$

On cherche le point d'intersection $(x_1, f_e(x_1))$ de n_c avec l'extrados, de façon à déterminer l'épaisseur e de l'aube.
En exprimant $(x_1, f_e(x_1))$ grâce à (1.9) et en reportant dans (1.10), on obtient :

$$0 = -\frac{ac}{2}x_0x_1^2 - (abx_0 + 1)x_1 + \frac{a^2}{2}x_0^3 + x_0 + o(x_1^2).$$

Un développement limité mène alors à :

$$x_1 = x_0 - abx_0^2 + o(x_0^2).$$

L'épaisseur s'exprime par $e = ((x_0 - x_1)^2 + (f_c(x_0) - f_e(x_1))^2)^{1/2}$, ce qui donne, en développant :

$$e = -bx_0 + \frac{1}{2}\left(a + 2ab^2 - c\right)x_0^2 + o(x_0^2).$$

On souhaite maintenant connaître la position de l'intrados, (x_i, y_i), par rapport à sa tangente à la fuite. Du fait de la symétrie de l'intrados et de l'extrados par rapport à la cambrure, l'équation de la tangente à l'intrados au point de fuite est $y = -f_e'(0)x$, soit $y = -bx$. On étudie donc le signe de $y_i + bx_i$ au voisinage de l'origine.
Par définition de la géométrie de l'aube, on a $(x_i, y_i) = (x_0, f_c(x_0)) + e\overrightarrow{n_c}$, soit :

$$\begin{cases} x_i = x_0 + e(1 + a^2x_0^2)^{-1/2}(-ax_0), \\ y_i = \frac{a}{2}x_0^2 + e(1 + a^2x_0^2)^{-1/2}. \end{cases}$$

Un développement limité donne :

$$y_i + bx_i = \left(a + 2ab^2 - \frac{c}{2}\right)x_0^2 + o(x_0^2).$$

Par suite l'intrados est au-dessous de sa tangente si et seulement si :

$$a + 2ab^2 \leq \frac{c}{2}. \tag{1.11}$$

Par ailleurs, on étudie la concavité de l'intrados à la fuite en procédant de la même façon qu'en 1.2.1, c'est-à-dire en calculant le développement limité de $f_i''(x)$ au voisinage de l'origine. On trouve alors que la courbe de l'intrados est concave si et seulement si la condition (1.11) est vérifiée. □

Conclusion : l'intrados est concave et sous sa tangente à la fuite si la courbure de l'extrados est suffisamment grande devant celle de la cambrure. L'intrados peut alors ne pas avoir de point d'inflexion (il est concave), ou avoir deux points d'inflexion (il est concave à l'attaque, puis devient convexe et redevient concave avant la fuite).

1.3 Recherche d'un nombre raisonnable de points de contrôle de B-spline pour approcher la géométrie de référence de la pompe de fond de puits

1.3.1 Objectif

Dans l'ancien mode de représentation, la géométrie de référence de la pompe à optimiser est définie par des échantillonnages de points qui, une fois reliés, permettent de dessiner les contours des aubes et du moyeu. On a choisi de représenter la géométrie à l'aide de B-splines. Celles qui représentent la géométrie initiale (i.e. servant à initialiser l'optimisation) doivent approcher au mieux les échantillonnages de points de la géométrie de référence. Ce genre de problème d'approche de données (*data fitting*) est évoqué dans [17]. Dans [5] une technique d'approche de forme au sens des moindres carrés est utilisée pour des courbes de Bézier afin d'améliorer l'efficacité d'une optimisation par régularisation du polygone de contrôle de la forme.
On rappelle que les coordonnées des points de contrôle de B-splines représentant la géométrie de la pompe sont les variables de contrôle de l'optimisation. Par conséquent on souhaite avoir le moins de points de contrôle possible car leur nombre est moitié moindre que celui des variables de l'optimisation. Pour ce faire, on cherche un nombre de points de contrôle raisonnable permettant d'approcher les échantillonnages de points de la géométrie de référence en commettant une erreur assez petite pour être acceptable.
Les échantillonnages de points à approcher sont, pour le rotor et pour le stator, ceux de l'extrados, de la cambrure et des génératrices de la surface de révolution qu'est le moyeu.

1.3.2 Grandes lignes de la méthode retenue

On dispose, pour chacune des courbes évoquées précédemment, d'un échantillonnage de M points dont les coordonnées dans le repère cartésien sont notées $(\widehat{x_j},\widehat{y_j})_{j=1\ldots M}$, et on pose $\widehat{X}=(\widehat{x_j})_{j=1\ldots M}$ et $\widehat{Y}=(\widehat{y_j})_{j=1\ldots M}$. On cherche à tracer une B-spline à n points de contrôle $(x_i,y_i)_{i=1\ldots n}$, approchant l'échantillonnage de points. Pour plusieurs valeurs de n (par exemple comprises entre des bornes "raisonnables" comme 4 et 8) on mène une optimisation sur les coordonnées des points de contrôle. La méthode d'optimisation choisie doit prendre en compte les contraintes sur les coordonnées des points de contrôle, traduisant les contraintes géométriques imposées aux courbes à approcher. Au terme de ces optimisations, on obtient des valeurs d'erreur permettant de choisir le meilleur n, ou tout du moins un n raisonnable.
L'optimisation se déroule comme suit. Pour un entier n donné, on choisit un vecteur initial $\mathring{X}$ constitué des n abscisses initiales $\mathring{x}_i$ de points de contrôle. On détermine par une méthode des moindres carrés avec contraintes le vecteur Y des ordonnées y_i optimales des points de contrôle. On peut ainsi tracer une B-spline que l'on note $(x(t,\mathring{X}),y(t,\mathring{X}))$ avec $t\in[0,n-3]$. Puis on utilise une fonction d'optimisation avec contraintes pour déterminer le vecteur $X=(x_i)_{i=1\ldots n}$ qui minimise la fonction

$$f_n(X)=\sum_{j=1}^{M}(\widehat{y_j}-y(t_j,X))^2$$

avec t_j tel que $\widehat{x_j}=x(t_j,X)$.

1.3.3 Méthode des moindres carrés pour déterminer Y

Soit $(\widehat{x_j},\widehat{y_j})_{j=1\ldots M}$ l'échantillonnage de M points à approcher. Soit un entier $n<M$ et $\mathring{X}=(\mathring{x}_i)_{i=1\ldots n}$ le vecteur initial des abscisses des points de contrôle. On suppose que ces abscisses sont classées en ordre croissant (cf. §1.1.2). On détermine $x(t,\mathring{X})$, $t\in[0,n-3]$ la B-spline unidimensionnelle ayant pour points de contrôle les $\mathring{x}_i$.
Soit $z_k=x(k-1,X)$, $\forall k=1,\ldots,n-2$. Pour tout j on détermine i tel que $z_i<\widehat{x_j}<z_{i+1}$. On résout une équation polynomiale du troisième degré pour déterminer $t_j\in[i-1,i[$ tel que $x(t_j,X)=\widehat{x_j}$ (si $z_{n-2}=\widehat{x_M}$, on pose $t_M=n-3$).
La méthode des moindres carrés doit permettre de déterminer le vecteur $Y=(y_i)_{i=1\ldots n}$ qui minimise $||(\widehat{y_j}-y(t_j,\mathring{X}))_{j=1\ldots M}||_2^2$, où $y(t,\mathring{X})$, $t\in[0,n-3]$ est la B-spline unidimensionnelle ayant pour points de contrôle les y_i.
On peut encore écrire $||(\widehat{y_j}-y(t_j,\mathring{X}))_{j=1\ldots M}||_2^2$ sous la forme $||\widehat{Y}-PY||_2^2$ où la matrice P est de la forme :

$$P=\begin{pmatrix} p_3(t_1) & p_2(t_1) & p_1(t_1) & p_0(t_1) & 0 & 0 & \ldots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots\\ p_3(t_j) & p_2(t_j) & p_1(t_j) & p_0(t_j) & 0 & 0 & \ldots & 0\\ 0 & p_3(t_{j-1}) & p_2(t_{j-1}) & p_1(t_{j-1}) & p_0(t_{j-1}) & 0 & \ldots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & p_3(t_l) & p_2(t_l) & p_1(t_l) & p_0(t_l) & 0 & \ldots & 0\\ & & & \ddots & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & \ldots & p_0(t_M) \end{pmatrix}.$$

On cherche donc la solution, au sens des moindres carrés, de $\widehat{Y}=PY$, que l'on peut encore écrire $R=AY$ où $R={}^TP\widehat{Y}$ et $A={}^TPP$.

Une fois déterminé le vecteur Y, il ne reste qu'à calculer l'erreur commise :

$$f_n(\mathring{X}) = ||(\widehat{y_j} - y(t_j, \mathring{X}))_{j=1\dots M}||^2 = \sum_{j=1}^{M} (\widehat{y_j} - y(t_j, \mathring{X}))^2.$$

1.3.4 Liste des contraintes à prendre en compte pour les aubages

Les contraintes sont ici de différents types. Il y a celles dont on a déjà fait la liste dans l'introduction et qui sont de nature géométrique (et désignées dans la suite par l'exposant $^{(1)}$), et il y a des contraintes propres à faciliter le bon déroulement de l'algorithme de recherche d'un nombre de points raisonnable (et désignées dans la suite par l'exposant $^{(2)}$). On reprend ici ces contraintes en détaillant la façon dont elles sont prises en compte (l'exposant qui les suit fait référence à leur "type").
Notations : la plupart des contraintes sont les mêmes pour la cambrure et l'extrados, mais s'il est nécessaire de distinguer les deux cas, on utilise les exposants ^{c} ou ^{e}.

Points d'attaque et de fuite imposés$^{(2)}$

On choisit d'imposer que les B-splines de la cambrure et de l'extrados aient leur origine (*resp.* leur extrémité) au même point, qui est le point d'attaque (*resp.* de fuite) de la géométrie de référence. Ces contraintes d'égalité s'écrivent $x(0,X)\,(=z_1) = \widehat{x_1}$, $x(n-3,X)\,(=z_n) = \widehat{x_M}$, $y(0,X) = \widehat{y_1}$ et $y(n-3,X) = \widehat{y_M}$. Ce qui se traduit matriciellement par $A^x_{eq,1}X = b^x_{eq,1}$ et $A^y_{eq,1}Y = b^y_{eq,1}$ avec

$$A^x_{eq,1} = A^y_{eq,1} = \begin{pmatrix} p_3(0) & p_2(0) & p_1(0) & p_0(0) & 0 & \dots & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & p_3(1) & p_2(1) & p_1(1) & p_0(1) \end{pmatrix},$$

$$b^x_{eq,1} = \begin{pmatrix} \widehat{x_1} \\ \widehat{x_M} \end{pmatrix} \text{ et } b^y_{eq,1} = \begin{pmatrix} \widehat{y_1} \\ \widehat{y_M} \end{pmatrix}.$$

Remarque 1.3.1. *On a aussi essayé de forcer la contrainte d'égalité en menant une méthode des moindres carrés pour déterminer les $(y_i)_{i=2\dots n-1}$ et en résolvant le système $A^y_{eq,1}Y = b^y_{eq,1}$ pour déterminer y_1 et y_n. On a ensuite mené l'optimisation sur les $(x_i)_{i=2\dots n-1}$ et résolu le système $A^x_{eq,1}X = b^x_{eq,1}$ pour déterminer x_1 et x_n. Cette méthode ne donne pas un résultat satisfaisant. La recherche des $(y_i)_{i=2\dots n-1}$ et $(x_i)_{i=2\dots n-1}$ optimaux ne permet que de minimiser la distance des $n-5$ intervalles "intérieurs" de la B-spline aux points $(\widehat{x_j}, \widehat{y_j})_{j=j_2\dots j_{(n-3)}}$, avec j_2 tel que $\widehat{x_{j_2-1}} < z_2 < \widehat{x_{j_2}}$ et $j_{(n-3)}$ tel que $\widehat{x_{j_{(n-3)}}} < z_{n-3} < \widehat{x_{j_{(n-3)}+1}}$. La détermination de y_1, y_n, x_1 et x_n donne alors des formes peu convenables pour le premier et le dernier intervalle de la B-spline, principalement parce qu'ils sont très éloignés des points $(\widehat{x_j}, \widehat{y_j})_{j=1\dots j_2-1}$ et $(\widehat{x_j}, \widehat{y_j})_{j=j_{(n-3)}+1\dots M}$ à approcher.*

Contraintes pratiques $z_1 \leq \widehat{x_1}$ et $z_n \geq \widehat{x_M}$$^{(2)}$

Ces contraintes d'inégalité sont d'ordre pratique. Par exemple, il est clair que si jamais $z_1 \geq \widehat{x_1}$, il n'est plus possible de déterminer un encadrement de $\widehat{x_1}$ par des z_i. L'algorithme mis en place ne peut plus tourner et la recherche s'arrête. Comme il n'est pas garanti que les contraintes d'égalité $z_1 = \widehat{x_1}$ et $z_n = \widehat{x_M}$ soient respectées parfaitement, il est préférable d'ajouter ces deux contraintes d'inégalité. Elles se traduisent matriciellement par : $A^x_1 X \leq b^x_1$ avec

$$A^x_1 = \begin{pmatrix} p_3(0) & p_2(0) & p_1(0) & p_0(0) & 0 & \dots & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & -p_3(1) & -p_2(1) & -p_1(1) & -p_0(1) \end{pmatrix}$$

et

$$b_1^x = \begin{pmatrix} \widehat{x_1} \\ -\widehat{x_M} \end{pmatrix}.$$

Contrôle des angles entre extrados et cambrure à l'attaque et à la fuite(1)

Dans les deux cas, l'optimisation ne porte que sur l'extrados, une fois la cambrure elle-même construite. Par conséquent, les variables de l'optimisation portent l'exposant ^{e} tandis que les termes relatifs à la cambrure, portant l'exposant ^{c}, sont fixes.
A l'attaque, la tangente à l'extrados doit former un angle droit avec la tangente à la cambrure. Le produit scalaire entre les vecteurs tangents est donc nul.
On note, à l'attaque, $\vec{\tau}_e^a = (\tau_{e,x}^a, \tau_{e,y}^a)$ le vecteur tangent à l'extrados et $\vec{\tau}_c^a = (\tau_{c,x}^a, \tau_{c,y}^a)$ le vecteur tangent à la cambrure. La contrainte d'égalité non linéaire se traduit alors par :

$$\begin{aligned} c_{eq,1}(X) &= \vec{\tau}_c^a.\vec{\tau}_e^a \\ &= \tau_{c,x}^a.\left(x_1^e p_3'(0) + x_2^e p_2'(0) + x_3^e p_1'(0) + x_4^e p_0'(0)\right) \\ &\quad + \tau_{c,y}^a.\left(y_1^e p_3'(0) + y_2^e p_2'(0) + y_3^e p_1'(0) + y_4^e p_0'(0)\right) \\ &= 0. \end{aligned}$$

A la fuite, l'angle entre la tangente à l'extrados et la tangente à la cambrure n'est pas imposé. Cependant on souhaite éviter une géométrie absurde où l'angle deviendrait plus grand que $\pi/2$. On peut même décider de fixer une borne supérieure θ plus petite que $\pi/2$. Le cas où la tangente à l'extrados ne serait pas dans le plan situé au-dessus de la cambrure est aussi absurde. On note, à la fuite, $\vec{\tau}_e^f = (\tau_{e,x}^f, \tau_{e,y}^f)$ le vecteur tangent à l'extrados, $\vec{\tau}_c^f = (\tau_{c,x}^f, \tau_{c,y}^f)$ le vecteur tangent à la cambrure, et $\vec{n}_c^f = (-\tau_{c,y}^f, \tau_{c,x}^f)$ son vecteur normal.
Les contraintes énoncées ci-dessus s'écrivent sous forme vectorielle ainsi : $\vec{n}_c^f.\vec{\tau}_f^e \geq 0$ et $\vec{\tau}_c^f.\vec{\tau}_f^e \geq \cos\theta$.
On a donc une contrainte inégalité non linéaire de la forme $C_1(X^e) \leq 0_{2,1}$ avec

$$C_1(X^e) = \begin{pmatrix} \cos\theta - \tau_{c,x}^f(x_{n-3}^e p_3'(1) + x_{n-2}^e p_2'(1) + x_{n-1}^e p_1'(1) + x_n^e p_0'(1))... \\ -\tau_{c,y}^f(y_{n-3}^e p_3'(1) + y_{n-2}^e p_2'(1) + y_{n-1}^e p_1'(1) + y_n^e p_0'(1)) \\ \\ \tau_{c,y}^f(x_{n-3}^e p_3'(1) + x_{n-2}^e p_2'(1) + x_{n-1}^e p_1'(1) + x_n^e p_0'(1))... \\ -\tau_{c,x}^f(y_{n-3}^e p_3'(1) + y_{n-2}^e p_2'(1) + y_{n-1}^e p_1'(1) + y_n^e p_0'(1)) \end{pmatrix}.$$

Abscisses (x_i) croissantes(2)

La B-spline unidimensionnelle des abscisses doit être croissante. Le théorème 1.1.1 (cf. §1.1.2) montre qu'une condition **suffisante** pour cela est que la famille des abscisses (x_i) des points de contrôle soit croissante. On a donc une contrainte d'inégalité qui s'écrit matriciellement : $A_2^x X \leq b_2^x$ avec

$$A_2^x = \begin{pmatrix} 1 & -1 & 0 & \dots & 0 \\ 0 & 1 & -1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & 1 & -1 \end{pmatrix} \text{ et } b_2^x = 0_{n-1,1}.$$

Remarque 1.3.2. *Cette contrainte a pour principal avantage de réduire considérablement le nombre d'arrêts prématurés de l'algorithme. En revanche elle limite trop la qualité des résultats dans le cas*

de l'extrados. En effet, celui-ci présente à l'attaque (et aussi, dans une moindre mesure, à la fuite) une courbure importante, qui ne peut être approchée de façon correcte qu'en violant la contrainte. Pour cette raison on peut décider de relaxer la contrainte dans le cas de l'extrados, soit complètement, soit après avoir réalisé un certain nombre d'itérations. Le problème ne se pose pas à la cambrure du fait de sa grande régularité.

Condition suffisante pour la convexité des courbes(1)

Comme on l'a exposé en introduction de ce mémoire, on doit imposer une contrainte de convexité aux profils de la cambrure et de l'extrados. Dans le cadre de l'étude réalisée ici, prendre une version simplifiée des contraintes énoncées en 1.1.3 suffit à assurer la convexité. Ces contraintes simplifiées s'expriment à l'aide de produits vectoriels :

$$\begin{pmatrix} x_{i+2}-x_i \\ y_{i+2}-y_i \end{pmatrix} \wedge \begin{pmatrix} x_{i+3}-x_i \\ y_{i+3}-y_i \end{pmatrix} \geq 0 \text{ pour } i \in \{1,...,n-3\},$$

$$\begin{pmatrix} x_{i+2}-x_{i+1} \\ y_{i+2}-y_{i+1} \end{pmatrix} \wedge \begin{pmatrix} x_{i+3}-x_{i+1} \\ y_{i+3}-y_{i+1} \end{pmatrix} \geq 0 \text{ pour } i \in \{0,...,n-3\}.$$

Il va donc s'agir d'une contrainte inégalité non linéaire, s'écrivant matriciellement : $C_2(X) \leq 0_{2n-5,1}$ avec

$$\forall i \in \{1,...,n-3\},\ [C_2]_i(X) = -(x_{i+2}-x_i)(y_{i+3}-y_i)+(y_{i+2}-y_i)(x_{i+3}-x_i),$$

$$\forall i \in \{0,...,n-3\},\ [C_2]_{n-3+i}(X) = -(x_{i+2}-x_{i+1})(y_{i+3}-y_{i+1})+(y_{i+2}-y_{i+1})(x_{i+3}-x_{i+1}).$$

1.3.5 Mise en œuvre numérique

La fonction Matlab `lsqlin` pour la méthode des moindres carrés

On souhaite déterminer le vecteur Y par une méthode des moindres carrés. On a uniquement des contraintes linéaires d'égalité, ce qui conduit à choisir `lsqlin`. Cette fonction de Matlab [44] résout un problème de moindres carrés linéaire contraint de la forme :

$$\min_Y \frac{1}{2}||PY-\widehat{Y}||_2^2 \text{ avec } A_{eq}^y Y = b_{eq}^y.$$

Pour la cambrure, on pose $A_{eq}^{c,y} = A_{eq,1}^y$ et $b_{eq}^{c,y} = b_{eq,1}^y$. Pour l'extrados, on pose $A_{eq}^{e,y} = \begin{pmatrix} A_{eq,1}^y \\ A_{eq,2}^y \end{pmatrix}$ et $b_{eq}^{e,y} = \begin{pmatrix} b_{eq,1}^y \\ b_{eq,2}^y \end{pmatrix}$.

La fonction Matlab `fmincon` pour la minimisation de la fonction f_n

Déterminer X nécessite de respecter des contraintes d'égalité et d'inégalité aussi bien linéaires que non linéaires. On choisit donc d'utiliser la fonction `fmincon` [44] qui trouve (par une méthode SQP) le minimum d'une fonction réelle de plusieurs variables avec contraintes (pouvant être non linéaires) en partant d'un état initial $\mathring{X}$. Autrement dit `fmincon` traite des problèmes de la forme :

$$\min_X f_n(X) \text{ avec } A^x X \leq b^x,\ A_{eq}^x X = b_{eq}^x,\ c_{eq}(X) = 0 \text{ et } C(X) \leq 0.$$

Pour la cambrure on pose $A^{c,x} = \begin{pmatrix} A_1^x \\ A_2^x \end{pmatrix}$, $b^{c,x} = \begin{pmatrix} b_1^x \\ b_2^x \end{pmatrix}$, $A_{eq}^{c,x} = A_{eq,1}^x$, $b_{eq}^{c,x} = b_{eq,1}^x$ et $C^c(X) = C_2(X)$.
Pour l'extrados, on pose $A^{e,x} = \begin{pmatrix} A_1^x \\ A_2^x \end{pmatrix}$, $b^{e,x} = \begin{pmatrix} b_1^x \\ b_2^x \end{pmatrix}$, $A_{eq}^{e,x} = \begin{pmatrix} A_{eq,1}^x \\ A_{eq,2}^x \end{pmatrix}$, $b_{eq}^{e,x} = \begin{pmatrix} b_{eq,1}^x \\ b_{eq,2}^x \end{pmatrix}$, $c_{eq}^e(X) = c_{eq,1}(X)$ et $C^e(X) = \begin{pmatrix} C_1(X) \\ C_2(X) \end{pmatrix}$.

Choix du vecteur $\mathring{X}$ initial

On est en présence d'un problème très sensible à l'initialisation. Des optimisations successives à partir de vecteurs initiaux différents conduisent à quantité de minima locaux.
Dans un premier temps on a pensé tester un grand nombre de vecteurs initiaux $\mathring{X}$ choisis aléatoirement grâce à la fonction Matlab `rand`. Si cette méthode est efficace dans le cas de la cambrure, en revanche on constate, dans le cas de l'extrados, que l'on a rarement la chance d'aboutir à une solution convenable, les erreurs obtenues étant presque systématiquement > 1.
La méthode retenue consiste à construire des vecteurs initiaux $\mathring{X}$ "prometteurs". Pour cela on choisit n réels parmi les M abscisses $(\widehat{x_j})_{j=1\ldots M}$ des points de l'échantillonnage à approcher, de préférence équiréparties, et incluant $\widehat{x_1}$ et $\widehat{x_M}$. On note $\tilde{X}$ le vecteur ainsi constitué. On recherche une B-spline unidimensionnelle passant par chacun des $\widehat{x_j}$ retenus. Pour cela, on résout un système de la forme $\tilde{P}\mathring{X} = \tilde{X}$, où $\tilde{P}$ est écrite de manière astucieuse et adaptée au vecteur $\tilde{X}$. Par exemple, on peut prendre pour $n = 4$,

$$\tilde{P}_4 = \begin{pmatrix} p_3(0) & p_2(0) & p_1(0) & p_0(0) \\ p_3(1/3) & p_2(1/3) & p_1(1/3) & p_0(1/3) \\ p_3(2/3) & p_2(2/3) & p_1(2/3) & p_0(2/3) \\ p_3(1) & p_2(1) & p_1(1) & p_0(1) \end{pmatrix},$$

pour $n = 5$,

$$\tilde{P}_5 = \begin{pmatrix} p_3(0) & p_2(0) & p_1(0) & p_0(0) & 0 \\ p_3(1/2) & p_2(1/2) & p_1(1/2) & p_0(1/2) & 0 \\ p_3(1) & p_2(1) & p_1(1) & p_0(1) & 0 \\ 0 & p_3(1/2) & p_2(1/2) & p_1(1/2) & p_0(1/2) \\ 0 & p_3(1) & p_2(1) & p_1(1) & p_0(1) \end{pmatrix},$$

pour $n = 6$,

$$\tilde{P}_6 = \begin{pmatrix} p_3(0) & p_2(0) & p_1(0) & p_0(0) & 0 & 0 \\ p_3(1/2) & p_2(1/2) & p_1(1/2) & p_0(1/2) & 0 & 0 \\ p_3(1) & p_2(1) & p_1(1) & p_0(1) & 0 & 0 \\ 0 & p_3(1) & p_2(1) & p_1(1) & p_0(1) & 0 \\ 0 & 0 & p_3(1/2) & p_2(1/2) & p_1(1/2) & p_0(1/2) \\ 0 & 0 & p_3(1) & p_2(1) & p_1(1) & p_0(1) \end{pmatrix}, \text{ etc.}$$

Sous réserve que $\tilde{P}$ soit inversible, on obtient donc $\mathring{X} = \tilde{P}^{-1}\tilde{X}$.

Emploi d'une pénalisation pour imposer les points d'attaque et de fuite

Lors des premiers tests de l'algorithme, on a observé l'apparition de solutions aberrantes. Tandis que l'erreur ne semble pas très importante, on constate que les contraintes d'égalité utilisées pour imposer les points d'attaque et de fuite ne sont pas respectées, et lorsque l'on trace la B-spline, on obtient un résultat du type de celui de FIG.1.4.

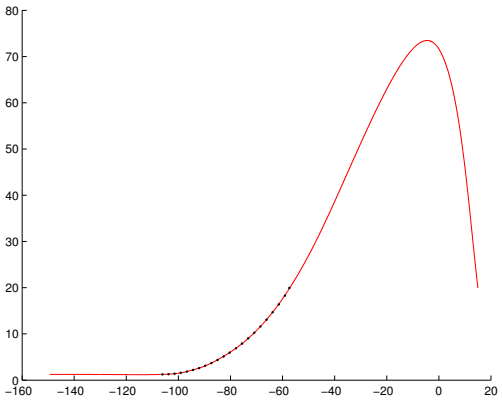

FIG. 1.4: *B-spline de la cambrure de l'extrados (rouge) et échantillonnage de points à approcher (points noirs).*

Dans de tels cas, l'erreur est petite car elle ne prend en compte que la distance verticale de la B-spline aux points de contrôle. Pour éviter que de tels cas se produisent, on a recours à une pénalisation.
Comme on vient de le dire, les contraintes d'égalité $y(0,X) = \widehat{y_1}$ et $y(n-3,X) = \widehat{y_M}$ sont respectées mais absolument pas $x(0,X)\,(= z_1)\ = \widehat{x_1}$, ni $x(n-3,X)\,(= z_n)\ = \widehat{x_M}$. On pénalise donc l'erreur f_n en considérant une nouvelle erreur :

$$f_n^p(X) = f_n(X) + \left((\widehat{x_1} - z_1)^2 + (\widehat{x_M} - z_n)^2\right).$$

On relance le processus d'optimisation avec cette nouvelle formule de l'erreur et on observe la disparition du problème.

1.3.6 Résultats numériques pour l'extrados

On présente dans le tableau TAB.1.1 les résultats de l'algorithme d'optimisation décrit ci-dessus pour l'extrados. Dans chaque cas, il a été vérifié que les contraintes sont bien respectées.

n	4	5	6	7	8
f_n	5.405	0.567	0.116	0.161	0.009

TAB. 1.1: *Erreur f_n (en mm^2) en fonction du nombre n de points de contrôle pour l'extrados.*

A une exception près, l'erreur diminue lorsque l'on augmente le nombre de points de contrôle. Pour $n = 5$ l'erreur commise est acceptable, par conséquent on retient $n = 5$ et les coordonnées $(x_i, y_i)_{i=1\ldots5}$ qui ont été obtenues à l'issue de l'optimisation.

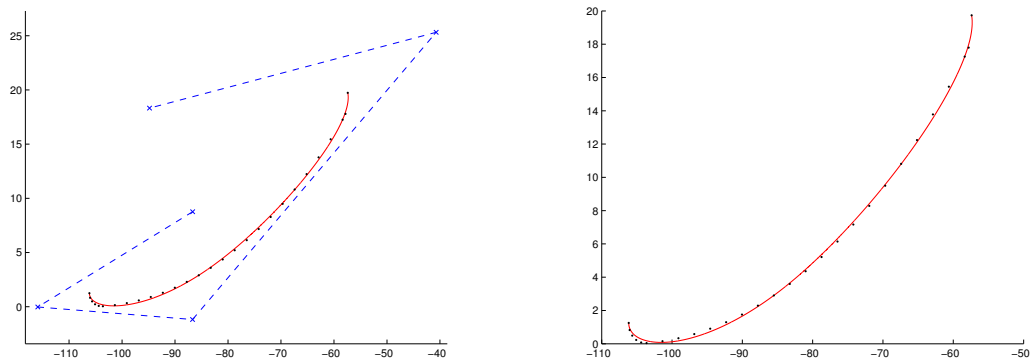

FIG. 1.5: *B-spline de l'extrados (rouge) avec $n = 5$ points de contrôle et échantillonnage de points à approcher (points noirs). A gauche, avec les points de contrôle reliés entre eux, à droite, en gros plan et sans les points de contrôle.*

1.3.7 Résultats numériques pour la cambrure

Le cas de la cambrure est différent de celui de l'extrados car il s'agissait initialement d'une courbe polynomiale du troisième degré. Par conséquent, on peut l'obtenir exactement, à l'aide de $n = 4$ points de contrôle. On se contentera donc de faire la recherche des points de contrôle pour $n = 4$.
Par ailleurs, s'il est intéressant de construire un vecteur initial $\mathring{X}$ dans le cas de l'extrados, car la géométrie est complexe, en revanche cette méthode n'apporte rien dans le cas de la cambrure. On constate qu'en faisant tourner l'algorithme à partir d'une centaine de vecteurs initiaux $\mathring{X}$ choisis aléatoirement, on obtient un bien meilleur résultat (erreur de l'ordre de 10^{-5}) qu'en essayant de construire $\mathring{X}$ (erreur de l'ordre de 10^{-2}), et ceci avec moins d'efforts.
Pour $n = 4$, on trouve un optimum local $f_4 = 6,3.10^{-5}$. On retient donc comme solution $n = 4$ et les coordonnées $(x_i, y_i)_{i=1...4}$ qui ont été obtenues à l'issue de l'optimisation.

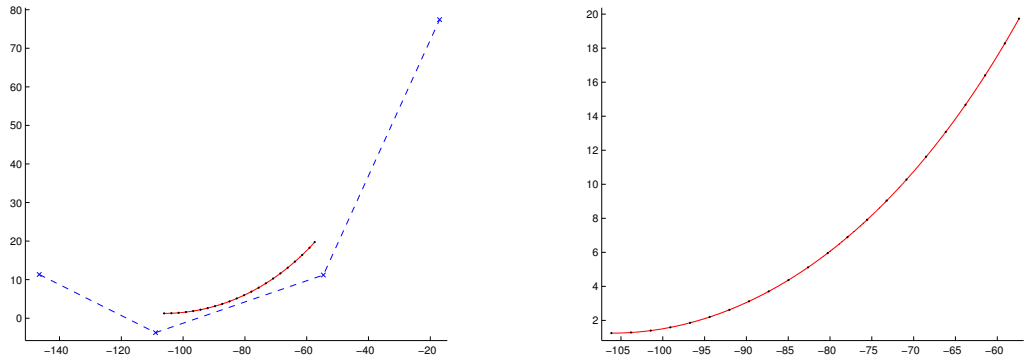

FIG. 1.6: *B-spline de la cambrure (rouge) avec $n = 4$ points de contrôle et échantillonnage de points à approcher (points noirs). A gauche, avec les points de contrôle reliés entre eux, à droite, en gros plan et sans les points de contrôle.*

Chapitre 2

Résolution numérique des équations de Navier-Stokes

2.1 Le logiciel de *Computational Fluid Dynamics* (CFD) Fluent

2.1.1 Présentation générale

La société Fluent Inc., récemment rachetée par ANSYS Inc., fournit des logiciels commerciaux et des services en simulation et calcul de mécanique des fluides, plus couramment désignés sous le terme générique de CFD (*Computational Fluid Dynamics*). Ce chapitre de présentation est en grande partie inspiré de la documentation [31], où il sera possible de trouver de nombreux compléments et détails qui ne sont pas le propos de cette thèse.
Le logiciel Fluent est un code rédigé en langage C permettant de modéliser les écoulements de fluides ainsi que les transferts de chaleur dans des géométries complexes. Fluent se compose essentiellement du solveur éponyme, et d'un préprocesseur de modélisation et de génération de maillage, nommé Gambit.
Un maillage créé dans Gambit peut être de différents types : triangles ou quadrilatères en 2D, héxaèdres, tétraèdres, prismes ou pyramides en 3D, ou encore hybride (mixte). Une fois le maillage créé dans Gambit et lu dans Fluent, il reste à réaliser dans Fluent les opérations suivantes : établir les conditions aux limites, définir les propriétés du fluide et le modèle physique, calculer l'écoulement, visualiser, sauvegarder et dépouiller les résultats.

2.1.2 Méthode de résolution numérique

Fluent utilise pour la résolution numérique en 3D la méthode des volumes finis. Le domaine d'écoulement est discrétisé en cellules volumiques appelées volumes de contrôle (c'est ce qui est fait lors de la création du maillage dans Gambit). Puis les équations d'état sont intégrées sur chaque volume de contrôle, discrétisées, puis linéarisées. Il reste alors à résoudre dans chaque volume un système linéaire.
On choisit une méthode de résolution séparée (*segregated*) : les différentes équations d'état sont résolues séparément les unes après les autres. A chaque itération Fluent procède ainsi :

- les trois équations scalaires de la conservation de la quantité de mouvement sont résolues à partir des valeurs courantes de la pression, afin de déterminer le vecteur vitesse,
- la pression et la vitesse sont "corrigées" de façon à respecter l'équation de conservation de la masse,
- les équations de la turbulence sont résolues à partir des précédentes variables mises à jour,
- un test de convergence est effectué.

Le processus est répété jusqu'à satisfaction du critère de convergence. La résolution des équations est faite par une méthode multigrille. Cette méthode permet d'accélérer la convergence du solveur en calculant les corrections de la solution courante sur des ensembles de volumes de contrôle d'abord grossiers puis de plus en plus fins au cours des itérations.
Par défaut les variables de l'écoulement sont calculées au centre des volumes de contrôle. Cependant ce sont les valeurs aux centres des faces qui sont nécessaires à la résolution des équations discrétisées. Il faut donc interpoler les valeurs aux centres des faces à partir de celles aux centres des volumes. Fluent propose plusieurs schémas de discrétisation, plus ou moins précis (premier ordre, deuxième ordre, Power-Law, QUICK, troisième ordre MUSCL...) parmi lesquels l'utilisateur doit choisir. On renvoie au paragraphe 7.4.1 pour plus de détails sur ce point.

2.2 Modélisation d'un écoulement de gasoil dans la pompe

2.2.1 Création et maillage de la géométrie dans Gambit

Dans Gambit, le module de maillage de Fluent, on lit des fichiers de données permettant de reconstituer la géométrie générée à partir des points de contrôle de B-splines. Le rotor et le stator sont maillés séparément, mais suivant la même méthode. Ils sont ensuite assemblés à l'aide d'un autre module de Fluent : *tmerge*.
Pour ce qui est du maillage, on crée une *sizefunction* appuyée sur la cambrure afin d'obtenir un maillage fin le long de l'aube et qui grossit quand on s'en éloigne radialement (dans Gambit, une *sizefunction* est une fonction de maillage qui permet de contrôler la croissance de la taille des mailles normalement à un objet qui peut être un point, une ligne ou une surface).
La première étape est le maillage surfacique du côté du carter. C'est un maillage non structuré de type pavage quoique le maillage soit localement structuré le long des aubages à l'aide d'une fonction appelée *boundary layer*, qui crée une couche de maillage structuré le long des parois.
Dans une seconde étape, on réalise le maillage volumique par extrusion du maillage surfacique du carter vers le moyeu (méthode de *Cooper*).
La qualité d'un maillage hexaédrique dépend de la régularité de la taille des mailles (il vaut mieux éviter qu'une petite maille jouxte une bien plus grande maille) et des angles aux sommets (qui doivent être le plus proche possible de 90°). La qualité des angles est contrôlée grâce au critère appelé *equiangle skew*. Ce paramètre sans dimension vaut

$$max\left(\frac{q_{max}-q_e}{180-q_e},\frac{q_e-q_{min}}{q_e}\right)$$

où q_{max} est le plus grand angle d'une cellule, q_{min} le plus petit, et $q_e = 90°$. Une valeur de 0 indique une cellule idéale (carré ou rectangle), une valeur de 1 indique une cellule complètement dégénérée (quadrangle dégénéré en triangle). On ne dépasse pas une valeur de 0.7 dans le maillage de la pompe, sous peine de ne pas voir les calculs sous Fluent converger.

2.2.2 Spécifications de la simulation numérique

Définition du matériau

On considère un écoulement monophasique d'un fluide de type gasoil, avec les constantes suivantes :
- masse volumique $\rho = 830\,kg/m^3$,
- viscosité dynamique $\mu = 0.003615\,kg/m.s$.

Conditions aux limites

- Repère du fluide au rotor : rotation à la vitesse $\omega = 377\,rad/s\,(3600\,tr/min)$
- Repère du fluide au stator : stationnaire
- Carter (du stator et du rotor) : mur stationnaire
- Moyeu du stator : mur stationnaire
- Moyeu du rotor : mur en mouvement de vitesse relative nulle par rapport au repère tournant du rotor
- Entrée du rotor :
  - sans prise en compte des étages environnants : vitesse axiale de $3\,m/s$ dans le repère absolu
  - avec prise en compte des étages environnants : condition périodique avec la sortie du stator
- Sortie du rotor et entrée du stator : interface
- Sortie du stator :
  - sans prise en compte des étages environnants : valeur arbitraire de la pression imposée en un point donné de l'espace (les valeurs numériques de la pression sont définies à une constante près, ce qui n'a pas de conséquence car le fluide étant incompressible, sa densité ne dépend pas de la pression)
  - avec prise en compte des étages environnants : condition périodique avec l'entrée du rotor
- Aubes du rotor : mur en mouvement de vitesse relative nulle par rapport au repère tournant du rotor
- Aubes du stator : mur stationnaire

2.2.3 Modélisation d'un écoulement basique

Pour tout écoulement, Fluent résout les équations de conservation de la masse et de la quantité de mouvement. Ces équations générales de la mécanique des fluides sont bien connues sous le nom d'équations de *Navier-Stokes*. On présente ici ces équations pour un écoulement laminaire dans un domaine fixe.

Notations : on écrit les équations projetées dans le repère cartésien $0\mathcal{E} = (0, e_1, e_2, e_3)$.

- Vitesse absolue du fluide : $\vec{U} = (u_1, u_2, u_3)$
- Masse volumique du fluide : ρ
- Force volumique par unité de masse : $\vec{f}$
- Tenseur global des contraintes : $\vec{\vec{\sigma}}$
- Tenseur des contraintes visqueuses : $\vec{\vec{\tau}}$

Conservation de la masse

La conservation de la masse s'énonce comme ceci :

$$\frac{\partial \rho}{\partial t} + \nabla.(\rho \vec{U}) = 0.$$

Remarque 2.2.1. *Un fluide incompressible homogène, tel que celui que l'on étudie, se caractérise par* $\rho = cste$ *et* $\nabla.\vec{U} = 0$. *L'équation de conservation de la masse devient alors*

$$\nabla.\vec{U} = 0.$$

Conservation de la quantité de mouvement

La conservation de la quantité de mouvement dans un repère fixe s'exprime ainsi :

$$\frac{\partial \rho \vec{U}}{\partial t} + \nabla.(\rho \vec{U}\vec{U}) = -\nabla p + \nabla.(\vec{\vec{\tau}}) + \rho \vec{f}. \tag{2.1}$$

Les équations projetées de la conservation de la quantité de mouvement s'écrivent alors sous la forme :

$$\frac{\partial \rho u_i}{\partial t} + \nabla.(\rho u_i \vec{U}) = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ii}}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + \frac{\partial \tau_{ik}}{\partial x_k} + \rho f_i.$$

Le tenseur global des contraintes est : $\overset{\Rightarrow}{\sigma} = -p\overset{\Rightarrow}{I} + \overset{\Rightarrow}{\tau}$.
Pour les fluides newtoniens, et d'après Stokes : $\overset{\Rightarrow}{\tau} = 2\mu\overset{\Rightarrow}{D} + \lambda(\nabla.\vec{U})\overset{\Rightarrow}{I}$ où

- $\overset{\Rightarrow}{D} = \frac{1}{2}(\nabla\vec{U} + {}^T\nabla\vec{U})$ est le tenseur des taux de déformation,
- μ est le coefficient de viscosité moléculaire (coefficient de Lamé),
- λ est le second coefficent de viscosité (second coefficient de Lamé).

Selon un postulat de Stokes, $\lambda = \frac{2}{3}\mu$, d'où la loi de comportement pour un fluide newtonien :

$$\overset{\Rightarrow}{\sigma} = -p\overset{\Rightarrow}{I} + 2\mu\overset{\Rightarrow}{D} - \frac{2}{3}\mu(\nabla.\vec{U})\overset{\Rightarrow}{I}.$$

On détaille maintenant le tenseur symétrique $\overset{\Rightarrow}{\tau}$:

$$\begin{cases} \tau_{ii} = \lambda(\nabla.\vec{U}) + 2\mu\dfrac{\partial u_i}{\partial x_i}, \\ \tau_{ij} = \tau_{ji} = \mu\left(\dfrac{\partial u_j}{\partial x_i} + \dfrac{\partial u_i}{\partial x_j}\right) \quad \text{si } j \neq i. \end{cases}$$

On en déduit l'écriture simplifiée des équations de conservation de la quantité de mouvement (sous forme conservative) pour un fluide visqueux, incompressible, homogène, en régime instationnaire :

$$\rho\frac{\partial u_i}{\partial t} + \rho\nabla.(u_i\vec{U}) = -\frac{\partial p}{\partial x_i} + \mu\triangle u_i + \rho f_i.$$

Ce système s'écrit vectoriellement :

$$\rho\left(\frac{\partial \vec{U}}{\partial t} + \frac{1}{2}\nabla(\vec{U}^2) + (\nabla\wedge\vec{U})\wedge\vec{U}\right) = -\nabla p + \mu\triangle\vec{U} + \rho\vec{f}$$

et aussi sous la forme :

$$\rho\left(\frac{\partial \vec{U}}{\partial t} + \nabla.(\vec{U}\otimes\vec{U})\right) = -\nabla p + \mu\triangle\vec{U} + \rho\vec{f}.$$

2.2.4 Modélisation d'un écoulement dans un domaine mobile

Dans un étage de pompe, le rotor est une partie mobile (en rotation) du domaine. L'objet de ce paragraphe est d'expliquer comment peut être prise en charge cette rotation dans la modélisation.
Fluent propose trois approches pour le problème d'un domaine mobile :

- le modèle MRF (*multiple reference frame*),
- le modèle *mixing plane*,
- le modèle *sliding mesh*.

Les deux premiers modèles supposent que l'écoulement est stationnaire, tandis que le modèle *sliding mesh* suppose que l'écoulement est instationnaire, modélisant ainsi avec fidélité l'interaction entre une partie mobile et une partie fixe, mais demandant bien sûr un plus grand temps de calcul.

Equations de l'écoulement dans un domaine en rotation

Quand les équations de Navier-Stokes sont résolues dans un domaine en rotation, des termes additionnels apparaissent dans l'équation de conservation de la quantité de mouvement. Si on note $\vec{U}_{rel}$ la vitesse relative, $\vec{U}$ la vitesse absolue, $\vec{\Omega}$ le vecteur vitesse angulaire et $\vec{r}$ le vecteur position dans le repère tournant, on a :

$$\vec{U}_{rel} = \vec{U} - (\vec{\Omega} \wedge \vec{r}).$$

Le terme de gauche de l'équation de mouvement (2.1) dans le repère stationnaire :

$$\frac{\partial(\rho\vec{U})}{\partial t} + \nabla.(\rho\vec{U}\vec{U})$$

devient, pour l'équation écrite dans le repère tournant, en termes de vitesse absolue :

$$\frac{\partial(\rho\vec{U})}{\partial t} + \nabla.(\rho\vec{U}_{rel}\vec{U}) + \rho(\vec{\Omega} \wedge \vec{r})$$

et en termes de vitesse relative :

$$\frac{\partial(\rho\vec{U}_{rel})}{\partial t} + \nabla.(\rho\vec{U}_{rel}\vec{U}_{rel}) + \rho(2\vec{\Omega} \wedge \vec{U}_{rel} + \vec{\Omega} \wedge \vec{\Omega} \wedge \vec{r}) + \rho\frac{\partial\vec{\Omega}}{\partial t} \wedge \vec{r}$$

où $\rho(2\vec{\Omega} \wedge \vec{U}_{rel})$ est la force de Coriolis.
Aussi bien pour la formulation en vitesse absolue que relative, l'équation de conservation de la masse s'écrit

$$\frac{\partial\rho}{\partial t} + \nabla.(\rho\vec{U}_{rel}) = 0.$$

Modèle MRF (*multiple reference frame)*

Le modèle MRF est le plus simple des trois modèles proposés. Il s'agit d'une approximation stationnaire, parfois appelée *frozen rotor*, où on considère que chaque domaine bouge à une vitesse différente dans le repère fixe du laboratoire. Les équations dans chaque domaine sont écrites dans un repère propre à celui-ci. Ainsi, dans un domaine fixe les équations de Navier-Stokes sont celles énoncées dans le paragraphe 2.2.3, et dans un domaine tournant, ce sont celles énoncées dans le paragraphe 2.2.4. A la frontière entre les domaines, Fluent force la continuité de la vitesse absolue $\vec{U}$.
Le modèle MRF a été retenu pour la modélisation de la pompe en raison de sa facilité d'utilisation, de son faible coût en temps de calcul, et compte tenu de la qualité satisfaisante des résultats obtenus.

Modèle *mixing plane*

Il consiste à résoudre un problème stationnaire dans chaque domaine. Au cours de la résolution, les données de l'écoulement dans des domaines adjacents sont régulièrement moyennées en espace dans la direction circonférentielle et utilisées comme conditions aux limites à l'interface. L'opération est répétée jusqu'à convergence.
L'intérêt de ce calcul moyenné, dans le cas de la pompe, est que l'on ne modélise qu'un canal d'écoulement (i.e. canal entre deux aubes) au rotor, et un canal au stator, ce qui réduit considérablement le temps de calcul. L'inconvénient est qu'il est impossible de différencier le comportement du fluide selon la position respective des aubes du rotor et du stator, or on a vu en introduction, lors du recensement des défauts de l'écoulement, qu'il est important de le faire.

Modèle *sliding mesh*

Il s'agit de la méthode la plus précise, mais aussi la plus coûteuse en ressources informatiques. Au cours du calcul, les domaines considérés glissent, avec un pas de temps régulier, l'un par rapport à l'autre le long d'une interface non conforme commune. Il faut donc calculer le flux à travers l'interface. Pour cela, l'intersection entre l'interface de chaque domaine est déterminée. Cette intersection, constituée des parties en vis-à-vis de l'interface de chaque domaine, est traitée comme une zone "intérieure" à travers laquelle s'écoule le fluide. Les parties qui ne sont pas en vis-à-vis sont mises en paire et traitées comme une zone périodique.

2.2.5 Modèles de turbulence

Les écoulements turbulents, comme celui de la pompe de fond de puits, se caractérisent par des fluctuations des variables d'état à petites échelles. Il serait trop coûteux de résoudre le problème d'écoulement directement, aussi les équations d'état sont-elles moyennées de façon à supprimer les petites échelles. Cependant, les équations moyennées contiennent des inconnues nouvelles, nécessitant un modèle de turbulence pour les déterminer.
Fluent propose plusieurs modèles de turbulence. Malheureusement aucun de ceux-ci n'est universellement accepté comme étant le "meilleur". On peut distinguer essentiellement les résolutions RANS (*Reynolds Averaged Navier Stokes*) et LES (*Large Eddy Simulation*).
La résolution de type RANS ne s'intéresse qu'aux grandeurs moyennées, ce qui est peu coûteux en temps de calcul. Cette résolution utilise des modèles de turbulence tels que Spalart-Allmaras, $k-\varepsilon$, ou $k-\omega$.
La simulation des grandes échelles (LES), se propose de calculer directement les grandes échelles de l'écoulement, ces échelles étant isolées par un filtrage spatial. La modélisation des effets des plus petites échelles nécessite ensuite de résoudre un problème de fermeture. Cette méthode de résolution est beaucoup plus coûteuse qu'une méthode RANS, on ne l'utilise donc que pour valider les modèles de turbulence et on n'utilise sinon qu'un modèle de résolution de type RANS, offrant un gain de temps de calcul considérable.

Moyenne de Reynolds des équations de Navier-Stokes (RANS)

Dans les équations moyennées de Reynolds, les solutions (instantanées) des équations de Navier-Stokes sont décomposées en une partie moyenne et une partie fluctuante : $\phi = \overline{\phi} + \phi'$ où ϕ est une grandeur scalaire d'état.
L'équation de conservation de la masse reste :

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho u_i) = 0.$$

Les équations moyennées de conservation de la quantité de mouvement deviennent :

$$\frac{\partial \rho \overline{u_i}}{\partial t} + \frac{\partial}{\partial x_j}(\rho \overline{u_i u_j}) = -\frac{\partial \overline{p}}{\partial x_i} + \mu \frac{\partial}{\partial x_j}\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i} - \frac{2}{3}\delta_{ij}\frac{\partial \overline{u_k}}{\partial x_k}\right) + \frac{\partial}{\partial x_j}(-\rho \overline{u_i' u_j'}).$$

On retrouve donc la forme générale des équations de Navier-Stokes, à un terme près qui représente les effets de la turbulence. Le terme $-\rho \overline{u_i' u_j'}$ est appelé tenseur de Reynolds et les modèles de turbulence servent à le calculer afin de fermer le système d'équations à résoudre.
L'hypothèse de Boussinesq consiste à introduire la viscosité turbulente μ_t et l'énergie cinétique turbulente k afin de calculer le tenseur de Reynolds selon la formule :

$$-\rho \overline{u_i' u_j'} = \mu_t \left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right) - \frac{2}{3}\left(\rho \overline{k} + \mu_t \frac{\partial \overline{u_i}}{\partial x_i}\right)\delta_{ij}.$$

L'hypothèse de Boussinesq est utilisée dans les modèles Spalart-Allmaras, $k-\varepsilon$, et $k-\omega$. Dans ces deux derniers cas deux équations supplémentaires de transport sont résolues (l'une pour l'énergie cinétique turbulente k et l'autre pour le taux de dissipation de la turbulence ε ou pour le taux de dissipation spécifique ω).

Choix du modèle de turbulence

Il reste à choisir un des trois modèles parmi Spalart-Allmaras, $k-\varepsilon$, et $k-\omega$. On élimine le modèle $k-\omega$ car des expériences sur d'autres problèmes semblaient montrer que ce modèle n'était pas stable vis-à-vis de la finesse du maillage. Le modèle Spalart-Allmaras est un modèle à une équation (surtout utilisé en aérodynamisme externe) alors que le modèle $k-\varepsilon$ est un modèle à deux équations. Les modèles à une seule équation sont souvent inaptes à s'accommoder d'écoulements variant brusquement (passage d'une zone de proche paroi à une zone d'écoulement libre). C'est pourquoi on a choisi le modèle $k-\varepsilon$.
Il existe trois modèles $k-\varepsilon$ dans Fluent. Le modèle $k-\varepsilon$ standard est très populaire pour sa robustesse, son caractère économique et sa précision raisonnable. Le modèle $k-\varepsilon$ RNG (pour *ReNormalization Group*) ressemble beaucoup au $k-\varepsilon$ standard, mais avec quelques améliorations qui le rendent plus précis pour une plus large classe d'écoulements que le $k-\varepsilon$ standard. Enfin le modèle $k-\varepsilon$ *realizable* doit son nom au fait qu'il respecte certaines contraintes mathématiques elles-mêmes cohérentes avec la physique des écoulements turbulents. C'est un développement relativement récent du modèle standard, qui en diffère cependant dans deux sens importants : il contient

- une nouvelle formulation pour la viscosité turbulente,
- une nouvelle équation de transport pour le taux de dissipation ε.

Comme le modèle RNG, le modèle realizable représente une amélioration du modèle standard pour une classe plus large de problèmes d'écoulement.
Une étude de sensibilité au modèle de turbulence a été menée pour comparer les résultats de ces différents modèles dans le cas de l'écoulement de gasoil dans la pompe. Cette étude a montré que le modèle realizable donne des résultats semblables à ceux du modèle standard, pour un temps de calcul bien moindre que le modèle standard ou le modèle RNG, ce dernier modèle donnant des résultats un peu différents, bien que cohérents. On a donc décidé de se fier au modèle $k-\varepsilon$ realizable.

Modèle $k-\varepsilon$ realizable

A titre indicatif, on donne ici les grands traits du modèle $k-\varepsilon$ retenu, on renvoie à [31] pour plus de précisions.
Les équations de transport pour le modèle $k-\varepsilon$ realizable sont :

$$\frac{\partial}{\partial t}(\rho k)+\frac{\partial}{\partial x_j}(\rho k u_j)=\frac{\partial}{\partial x_j}\left(\left(\mu+\frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right)+G_k+G_b-\rho\varepsilon-Y_M+S_k$$

et

$$\frac{\partial}{\partial t}(\rho\varepsilon)+\frac{\partial}{\partial x_j}(\rho\varepsilon u_j)=\frac{\partial}{\partial x_j}\left(\left(\mu+\frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right)+\rho C_1 S_\varepsilon-\rho C_2\frac{\varepsilon^2}{k+\sqrt{\nu\varepsilon}}+C_{1\varepsilon}\frac{\varepsilon}{k}C_{3\varepsilon}G_b+S_\varepsilon$$

où les constantes C_1, C_2, $C_{1\varepsilon}$, $C_{3\varepsilon}$ et les expressions G_k, G_b, Y_M, S_k, S_ε, ν, σ_k, σ_ε sont calculées et calibrées en fonction du problème étudié (dans Fluent elles ont des valeurs par défaut que l'utilisateur peut modifier).

Lois de paroi

Les écoulements turbulents sont considérablement affectés par la présence de parois, qui sont les principales sources de vorticité et de turbulence. C'est en effet dans la région proche des parois que

les variables de l'écoulement ont les plus forts gradients. Par conséquent la modélisation spécifique des écoulements près des parois a une grande influence sur la qualité des solutions numériques finales.
L'expérience a montré que la région près des parois peut être divisée en trois zones. Dans la zone la plus proche de la paroi, aussi appelée sous-couche visqueuse, l'écoulement est presque laminaire et la viscosité joue un rôle dominant dans l'équation de conservation de la quantité de mouvement. Dans la zone la plus éloignée de la paroi, la turbulence joue le rôle majeur. Dans la zone médiane la turbulence et la viscosité sont d'importance égale. On renvoie à la documentation Fluent [31] pour plus de détails sur les formules et modèles proposés pour représenter l'écoulement dans ces différentes zones.

Chapitre 3

Résultats de convergence pour un problème semi-linéaire et un problème de Stokes dans une géométrie périodique

Des études expérimentales [60] ont été menées par l'IFP de Solaize, en collaboration avec l'INSA de Lyon, sur une pompe hélico-axiale du type R4S15. Le carter de la pompe expérimentale était transparent dans le but de réaliser des mesures et d'avoir un accès visuel à l'écoulement. Le fluide injecté était de l'eau.
Sur une telle pompe il a pu être observé qu'après passage dans plusieurs étages, le comportement du fluide tend à devenir périodique au sens où le gain de pression dans un étage se stabilise, ainsi que les profils de vitesse.
Par ailleurs, on rappelle que représenter numériquement la pompe entière est impossible (pour des raisons de temps de calcul) et qu'il est nécessaire de simplifier le modèle, en ne représentant qu'un étage. Une question qui se pose alors naturellement est de savoir si l'écoulement dans un étage standard de pompe ressemble à celui que l'on obtiendrait en considérant des conditions aux limites périodiques à l'entrée et à la sortie d'un étage. Bien sûr on n'espère pas que ce résultat soit vrai pour les premiers et derniers étages qui sont encore influencés par les conditions aux limites aux extrémités de la pompe. Ce genre de problème a déjà été étudié dans des articles de M. Chipot and Y. Xie [15].
On livre ici l'article [3] publié sous le titre *Convergence results for a semilinear problem and for a Stokes problem in a periodic geometry* par S. Baillet, A. Henrot [1], T. Takahashi [1], dans la revue *Asymptotic Analysis*.
Dans cet article, on étudie le comportement asymptotique de la solution d'un problème semi-linéaire et d'un problème de Stokes dans un domaine périodique constitué de cellules identiques, lorsque le nombre de cellules augmente. En particulier, on prouve dans les deux cas la convergence exponentielle vers la solution du même problème, posé sur une seule cellule, avec conditions périodiques aux bords.

[1] Institut Elie Cartan de Nancy, Nancy-Université - CNRS - INRIA, BP 239, 54506 Vandœuvre-lès-Nancy Cedex.

Abstract

In this paper, we study the asymptotic behavior of the solution of a semilinear problem and of a Stokes problem, with periodic data, when the size of the domain increases. In particular, we prove exponential convergence to the solution of the corresponding problem with periodic boundary conditions.

3.1 Introduction

At the end of the exploitation of an oil well, surface pumps are no longer efficient enough to extract the oil remaining in the reservoir. Instead of closing the well, oil companies might use well-pumps, introduced deep into the ground in order to maintain the production Such pumps are composed of a succession of identical stages (typically 15-20 stages, but it could go up to 100 stages) arranged in series. The process optimization needs numerical simulation, but representing the whole pump numerically is impossible (for obvious calculation costs). Thus it is necessary to simplify the model, by representing only one stage. Therefore, a very natural question which arises is the following : does the flow in one standard stage of the pump looks like the one we would obtain by considering periodic boundary conditions at the entrance and exit of the stage. Of course, one cannot hope that this kind of result holds for the first and the last stages which are still influenced by the boundary conditions at the top and the bottom of the pump.

This kind of situation has already been considered in a series of papers by M. Chipot and Y. Xie, see [13], [14], [15], where the authors use variational techniques to prove the desired convergence. See also the book [12] by the first author. Their work is quite general and it definitely inspired us. Our own work differs to the following extent :

- we choose to work with the uniform (L^∞) norm, which might be more natural for the engineers,
- we use a simpler approach, just based on the maximum principle, which allows us to get a stronger convergence result (exponential convergence instead of polynomial convergence),
- in the elliptic case, we consider a semilinear problem instead of a linear one. Conversely, we are restricted to the Laplacian operator and to a simpler geometry (periodicity in only one dimension) than the one used by Chipot and Xie.
- motivated by the initial question, we also consider the Stokes problem in two dimensions (with particular boundary conditions : the so-called Navier slip boundary conditions).

Let us now be more precise in the statement of the problems and the results we get. We consider a domain $\Omega_k \subset \mathbb{R}^N$, composed of $2k$ copies of a bounded open set Ω translated in the last (N-th) space direction. For sake of simplicity, we assume that $\Omega \subset \mathbb{R}^{N-1} \times (0,L)$ for some positive L and that

$$\Omega_k = \bigcup_{i=-k}^{k-1} (iLe_N + \Omega).$$

In all the sequel, we assume that Ω_k is a domain of class $C^{1,1}$ for all k.

Let $f \in L^2_{loc}(\mathbb{R}^N)$ be a L-periodic function in the last space direction.

In section 3.2, we are interested in the following semilinear problem posed on Ω_k :

$$\begin{aligned} -\Delta u_k + g(x,u_k) &= f \quad \text{in } \Omega_k, \\ u_k &= 0 \quad \text{on } \partial\Omega_k, \end{aligned}$$

where the function g satisfies usual assumptions (see section 3.2 for more details) and let us denote by u_∞ the solution of the periodic problem posed on Ω :

$$\begin{aligned} -\Delta u_\infty + g(x,u_\infty) &= f \quad \text{in } \Omega, \\ u_\infty &= 0 \quad \text{on } \Gamma, \\ u_\infty & \quad Le_N\text{-periodic}. \end{aligned}$$

where Γ is the lateral boundary of Ω. Then, we prove the convergence result :

Theorem 3.1.1. *For all $k_0 \in \mathbb{N}^*$, there exist two constants $K > 0$ and $\alpha > 0$, depending only on Ω_{k_0}, such that for all $k \geq k_0$,*

$$\|u_k - u_\infty\|_{L^\infty(\Omega_{k_0})} \leq Ke^{-\alpha k}.$$

In section 3.3, we work in two dimensions ($N = 2$) and we consider the following Stokes problem :

$$\begin{aligned} -\Delta u_k + \nabla p_k &= f \quad \text{in } \Omega_k, \\ \nabla \cdot u_k &= 0 \quad \text{in } \Omega_k, \\ \text{curl } u_k &= 0 \quad \text{on } \partial\Omega_k, \\ u_k \cdot n &= 0 \quad \text{on } \partial\Omega_k. \end{aligned}$$

We are interested in proving the convergence of the solution to the previous problem, towards the solution of the following periodic Stokes problem :

$$\begin{aligned} -\Delta u_\infty + \nabla p_\infty &= f \quad \text{in } \Omega, \\ \nabla \cdot u_\infty &= 0 \quad \text{in } \Omega, \\ \text{curl}\, u_\infty &= 0 \quad \text{on } \Gamma, \\ u_\infty \cdot n &= 0 \quad \text{on } \Gamma, \\ u_\infty & \quad Le_2\text{-periodic}. \end{aligned}$$

We will prove that :

Theorem 3.1.2. *For all $k_0 \in \mathbb{N}^*$, there exist some positive constants K, K', α such that, for all $k \geq k_0$, we have*

$$\begin{aligned} \|u_k - u_\infty\|_{L^\infty(\Omega_{k_0})^2} &\leq Ke^{-\alpha k}, \\ \|p_k - p_\infty\|_{L^2(\Omega_{k_0})} &\leq K'e^{-\alpha k}. \end{aligned}$$

At last, section 3.4 is devoted to some concluding remarks and possible extensions.

Notation :

x	$x = (x_1, \ldots, x_{N-1}, x_N) \in \mathbb{R}^N$
x'	$x' = (x_1, \ldots, x_{N-1})$ where $x = (x_1, \ldots, x_{N-1}, x_N) \in \mathbb{R}^N$
Ω	bounded open set of $\mathbb{R}^N$ such that $\Omega \subset \mathbb{R}^{N-1} \times [0, L]$, $L \in \mathbb{R}^{+*}$
Ω_k	union of $2k$ open set Ω, such that $\Omega_k \subset \mathbb{R}^{N-1} \times [-kL, kL]$
$\partial\Omega$	boundary of Ω
$\partial\Omega_k$	boundary of Ω_k
Σ_+	$\Sigma_+ = \partial\Omega \cap \{x_N = L\}$
Σ_-	$\Sigma_- = \partial\Omega \cap \{x_N = 0\}$
Σ	$\Sigma = \Sigma_+ \cup \Sigma_-$
Σ_{k+}	$\Sigma_{k+} = \partial\Omega_k \cap \{x_N = kL\}$
Σ_{k-}	$\Sigma_{k-} = \partial\Omega_k \cap \{x_N = -kL\}$
Σ_k	$\Sigma_k = \Sigma_{k+} \cup \Sigma_{k-}$
Γ	$\Gamma = \partial\Omega \backslash \Sigma$
Γ_k	$\Gamma_k = \partial\Omega_k \backslash \Sigma_k$

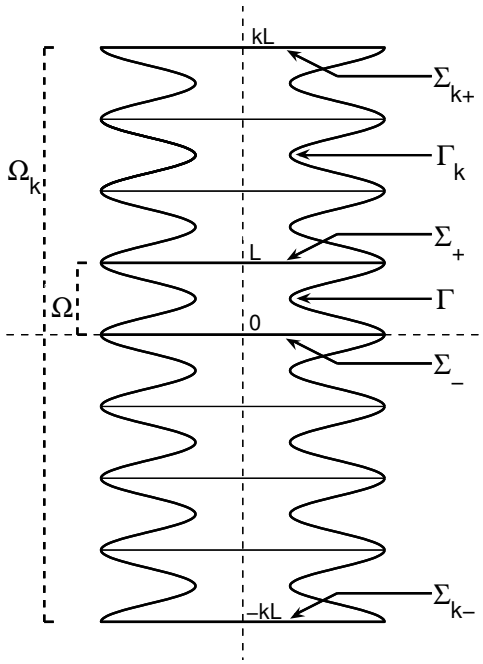

FIG. 3.1: Periodic geometry as studied in this paper.

3.2 Semilinear problems in N-dimension

In this section, we consider a semilinear problem in a periodic geometry immersed in $\mathbb{R}^N$ where $N \in \mathbb{N}$, $N \geq 2$. Let $f \in L^2_{loc}(\mathbb{R}^N)$ be a L-periodic function in the N-th space direction and let g be a function such that

$$x \mapsto g(x, \cdot)$$

is measurable and is a L-periodic function in the N-th space direction, and

$$s \mapsto g(\cdot, s) \text{ is continuous and increasing}, \quad g(\cdot, 0) = 0. \tag{3.1}$$

Let us remark that the assumption $g(\cdot, 0) = 0$ is not a restriction since, otherwise, we could consider $g_1 = g - g(\cdot, 0)$ and transfer $g(\cdot, 0)$ in the right-hand side with f.

Moreover, let us assume that g satisfies the following property :

- if $N = 2$, there exist $r \in (1, \infty)$, $a_0 \in L^{\frac{r}{r-1}}(\Omega)$ and $b_0 \geq 0$, such that

$$|g(\cdot, s)| \leq a_0(.) + b_0|s|^{r-1} \quad \text{in } \Omega; \tag{3.2}$$

- if $N \geq 3$, there exist $a_0 \in L^{\frac{2N}{N+2}}(\Omega)$ and $b_0 \geq 0$, such that

$$|g(\cdot, s)| \leq a_0(.) + b_0|s|^{\frac{N+2}{N-2}} \quad \text{in } \Omega. \tag{3.3}$$

For all $k \in \mathbb{N}^*$, we denote by u_k the solution of the problem

$$\begin{aligned} -\Delta u_k + g(x, u_k) &= f \quad \text{in } \Omega_k, \\ u_k &= 0 \quad \text{on } \partial\Omega_k, \end{aligned} \tag{3.4}$$

and by u_∞ the solution of the problem

$$\begin{aligned} -\Delta u_\infty + g(x,u_\infty) &= f \quad \text{in } \Omega, \\ u_\infty &= 0 \quad \text{on } \Gamma, \\ u_\infty & \quad Le_N\text{-periodic}. \end{aligned} \tag{3.5}$$

We recall that with the above hypotheses on g, the systems (3.4) and (3.5) are well-posed. More precisely, we have that

Proposition 3.2.1. *Assume that Ω_k is of class $C^{1,1}$ and assume that g satisfies the above hypotheses. Then for any $f \in L^2_{loc}(\mathbb{R}^N)$ and for any $k \in \mathbb{N}$, there exists a unique solution $u_k \in H^1_0(\Omega_k)$ to the problem* (3.4). *Moreover u_k belongs to the Sobolev space $W^{2,q}(\Omega_k)$ for some $q > 1$ (depending on the dimension N).*

The same result holds for problem (3.5). The proof of this proposition is very classical. The standard way to prove existence and uniqueness consists in looking for minimizers of the functional

$$I(v) = \int_{\Omega_k} \frac{1}{2}|\nabla v(x)|^2 + G(x,v(x)) - f(x)v(x)\, dx,$$

where

$$G(x,u) = \int_0^u g(x,s)\, ds.$$

We easily check that I is strictly convex, coercive and lower semicontinuous on $H^1_0(\Omega_k)$ so that I attains its minimum only at u_k. Moreover, from assumption (3.2), (3.3) together with Lemma 17.1 and Corollary 17.2 of [39, pp. 64,65] we see that I is differentiable with

$$I'(v) = -\Delta v + g(x,v) - f.$$

Consequently, we have obtained the existence and uniqueness of a function $u_k \in V$ such that

$$-\Delta u_k + g(x,u_k) = f.$$

For the regularity of the solution, in dimension 2, since u_k belongs to L^r space, we have $g(x,u_k) \in L^{\frac{r}{r-1}}(\Omega_k)$ and then $u_k \in W^{2,q}(\Omega_k)$ with $q = \min\left(\frac{r}{r-1}, 2\right)$ by using the classical L^q regularity results (see e.g. [34, Theorem 9.15]). In the case $N \geq 3$, thanks to assumption (3.3), we have that $g(x,u_k) \in L^{2N/(N+2)}(\Omega_k)$. Therefore, using again the L^q regularity results (see e.g. [34, Theorem 9.15]), u_k belongs to the Sobolev space $W^{2,q}(\Omega_k)$ with $q = 2N/(N+2)$ and then $-\Delta u_k + g(x,u_k(x)) = f(x)$ holds a.e.

The main result of this section is the following theorem.

Theorem 3.2.2. *Assume that Ω_1 is of class $C^{1,1}$. Then for all $k_0 \in \mathbb{N}^*$, there exist two constants $K > 0$ and $\alpha > 0$, depending only on Ω_{k_0}, such that for all $k \geq k_0$,*

$$\|u_k - u_\infty\|_{L^\infty(\Omega_{k_0})} \leq Ke^{-\alpha k}.$$

Proof. By periodicity assumption, the function u_∞ satisfies

$$-\Delta u_\infty + g(x,u_\infty) = f$$

in the whole domain Ω_k. Then, the function $v_k = u_k - u_\infty$ is solution of the following boundary value problem

$$\begin{aligned} -\Delta v_k + \left[\frac{g(x,u_k) - g(x,u_\infty)}{u_k - u_\infty}\right] v_k &= 0 \qquad \text{in } \Omega_k, \\ v_k &= 0 \qquad \text{on } \Gamma_k, \\ v_k &= -u_\infty \quad \text{on } \Sigma_k. \end{aligned} \tag{3.6}$$

Let B_{N-1} be a ball of $\mathbb{R}^{N-1}$ such that $\Omega_k \subset\subset B_{N-1} \times (-kL, kL)$. We denote by λ_1 the first eigenvalue of the Laplacian with Dirichlet boundary conditions in B_{N-1} and we consider an associated eigenfunction φ_1. We can assume that $\varphi_1 > 0$ in B_{N-1}. We define the function v in $B_{N-1} \times [-kL, kL]$ by

$$v(x) = M\varphi_1(x_1, \dots, x_{N-1}) \frac{\cosh(\sqrt{\lambda_1} x_N)}{\cosh(\sqrt{\lambda_1} kL)},$$

where M is a positive constant such that

$$M\varphi_1 \geq |u_\infty| \text{ on } \Sigma_k.$$

The function v satisfies the following system

$$\begin{aligned} -\Delta v &= 0 && \text{in } \Omega_k, \\ v &\geq 0 && \text{on } \Gamma_k, \\ v &\geq |u_\infty| && \text{on } \Sigma_k. \end{aligned} \tag{3.7}$$

By definition, we can notice that

$$v \geq 0 \quad \text{in } \Omega_k. \tag{3.8}$$

Using the fact that $s \mapsto g(\cdot, s)$ is an increasing function, we also have

$$\frac{g(x, u_k) - g(x, u_\infty)}{u_k - u_\infty} \geq 0 \quad \text{in } \Omega_k. \tag{3.9}$$

Therefore, from (3.6), (3.7), (3.8) and (3.9), we obtain

$$-\Delta(v - v_k) + \frac{g(x, u_k) - g(x, u_\infty)}{u_k - u_\infty}(v - v_k) = \frac{g(x, u_k) - g(x, u_\infty)}{u_k - u_\infty} v \geq 0 \text{ in } \Omega_k.$$

Consequently, the function $v - v_k$ satisfies

$$\begin{aligned} -\Delta(v - v_k) + \frac{g(x, u_k) - g(x, u_\infty)}{u_k - u_\infty}(v - v_k) &\geq 0 \quad \text{in } \Omega_k, \\ v - v_k &\geq 0 \quad \text{on } \partial\Omega_k. \end{aligned} \tag{3.10}$$

Likewise, the function $-v - v_k$ satisfies

$$\begin{aligned} -\Delta(-v - v_k) + \frac{g(x, u_k) - g(x, u_\infty)}{u_k - u_\infty}(-v - v_k) &\leq 0 \quad \text{in } \Omega_k, \\ -v - v_k &\leq 0 \quad \text{on } \partial\Omega_k. \end{aligned} \tag{3.11}$$

Then we apply the maximum principle [34, Theorem 8.1, p. 179] to the systems (3.10) and (3.11) and we get :

$$-v \leq v_k \leq v \text{ in } \Omega_k.$$

As a consequence, there exist two constants M' and K such that for all $k \geq k_0$,

$$\|u_k - u_\infty\|_{L^\infty(\Omega_{k_0})} \leq M' \frac{\cosh(\sqrt{\lambda_1} k_0 L)}{\cosh(\sqrt{\lambda_1} kL)} \leq K \exp(-\sqrt{\lambda_1} kL). \tag{3.12}$$

□

Remark 3.2.1. *According to (3.12), the rate of decay (measuring the speed of convergence of u_k to u_∞) is given by $\alpha = \sqrt{\lambda_1} L$. If we want to get the best rate of decay, we have to choose a ball B_{N-1} as small as possible. Of course, the greater λ_1, the greater the constant K appearing in (3.12). Besides, the optimal ball B_{N-1}, which would make the cylinder $B_{N-1} \times (-kL, kL)$ tangent to Ω_k, would let K go to $+\infty$.*

3.3 The Stokes problem with Navier slip boundary conditions

In this section, we consider the Stokes problem in dimension 2 in space (i.e. $N=2$). Let $f \in L^2_{loc}(\mathbb{R}^2)^2$ be a L-periodic function in the second space direction. For any $k \in \mathbb{N}$, we consider the solution (u_k, p_k) of the following Stokes problem in Ω_k (u_k is of course a vector-valued function) :

$$\begin{aligned} -\Delta u_k + \nabla p_k &= f \quad \text{in } \Omega_k, \\ \nabla \cdot u_k &= 0 \quad \text{in } \Omega_k, \\ \operatorname{curl} u_k &= 0 \quad \text{on } \partial\Omega_k, \\ u_k \cdot n &= 0 \quad \text{on } \partial\Omega_k. \end{aligned} \tag{3.13}$$

We also consider the solution (u_∞, p_∞) of the following periodic Stokes problem in Ω :

$$\begin{aligned} -\Delta u_\infty + \nabla p_\infty &= f \quad \text{in } \Omega, \\ \nabla \cdot u_\infty &= 0 \quad \text{in } \Omega, \\ \operatorname{curl} u_\infty &= 0 \quad \text{on } \Gamma, \\ u_\infty \cdot n &= 0 \quad \text{on } \Gamma, \\ u_\infty & \quad Le_2\text{-periodic}. \end{aligned} \tag{3.14}$$

We recall that for a vector function $u = (u_1, u_2)$, its curl is a scalar function defined by $\operatorname{curl} u = \frac{\partial u_2}{\partial x_1} - \frac{\partial u_1}{\partial x_2}$. Note that in the above systems, we have not considered the classical no-slip condition but instead we have chosen the Navier slip boundary condition introduced by Navier in [50]

$$u \cdot n = 0 \quad \text{and} \quad \operatorname{curl} u = 0 \quad \text{on } \Gamma.$$

The above condition can also be written under the form

$$u \cdot n = 0 \quad \text{and} \quad D(u)n \cdot \tau = 0 \quad \text{on } \Gamma,$$

where τ is the unit tangent vector field on Γ. This boundary condition corresponds to the physical hypothesis that the fluid slips at the border without friction. This slip boundary condition was derived from the Boltzmann equation in [18] and was already used by [33] and [19].

In the systems (3.13) and (3.14), the pressions p_k and p_∞ are unique up to a constant. In the sequel, we impose that

$$\int_{\Omega_k} p_k \, dx = 0, \quad \int_\Omega p_\infty \, dx = 0.$$

Let us also remark that, since $f \in L^2(\Omega)$, by classical ellipticity results the solution u_∞ belongs to $H^2(\Omega)$ and then to $L^\infty(\Omega)$. The same holds for u_k. We are now in position to state the main result of the paper.

Theorem 3.3.1. *For all $k_0 \in \mathbb{N}^*$, there exist three constants $K > 0$, $K' > 0$ and $\alpha > 0$, depending only on Ω_{k_0}, such that for all $k \geq k_0$,*

$$\|u_k - u_\infty\|_{L^\infty(\Omega_{k_0})^2} \leq K e^{-\alpha k} \text{ and } \|u_k - u_\infty\|_{H^1_0(\Omega_{k_0})^2} \leq K e^{-\alpha k},$$

$$\|p_k - p_\infty\|_{L^2(\Omega_{k_0})} \leq K' e^{-\alpha k}.$$

Proof. According to [35], there exist two functions ψ_k and ψ_∞ such that

$$\begin{aligned} \operatorname{curl} \psi_k = \begin{pmatrix} \frac{\partial \psi_k}{\partial x_2} \\ -\frac{\partial \psi_k}{\partial x_1} \end{pmatrix} &= u_k \quad \text{in } \Omega_k, \\ \psi_k &= 0 \quad \text{on } \partial\Omega_k, \end{aligned}$$

and

$$\begin{aligned} \operatorname{curl}\psi_\infty &= u_\infty \quad \text{in } \Omega, \\ \psi_\infty &= 0 \quad \text{on } \Gamma. \end{aligned}$$

Moreover, if we define

$$\omega_k = \operatorname{curl} u_k \quad \text{and} \quad \omega_\infty = \operatorname{curl} u_\infty,$$

then we have

$$\begin{aligned} -\Delta\psi_k &= \omega_k \quad \text{in } \Omega_k, \\ \psi_k &= 0 \quad \text{on } \partial\Omega_k, \end{aligned}$$

and

$$\begin{aligned} -\Delta\psi_\infty &= \omega_\infty \quad \text{in } \Omega_k, \\ \psi_\infty &= 0 \quad \text{on } \Gamma_k, \\ \psi_\infty & \; Le_2\text{-periodic.} \end{aligned}$$

Besides, the function ω_k, respectively ω_∞, satisfies the following boundary value problem :

$$\begin{aligned} -\Delta\omega_k &= \operatorname{curl} f \quad \text{in } \Omega_k, \\ \omega_k &= 0 \quad \text{on } \partial\Omega_k, \end{aligned}$$

respectively,

$$\begin{aligned} -\Delta\omega_\infty &= \operatorname{curl} f \quad \text{in } \Omega, \\ \omega_\infty &= 0 \quad \text{on } \Gamma, \\ \omega_\infty & \; Le_2\text{-periodic.} \end{aligned}$$

The results of the previous section (with $g = 0$) apply here for these two systems. Nevertheless, we need to be more precise since we want to apply it to a right-hand side which is not periodic (but almost).
Let $R_a = (-a,a) \times [-kL,kL]$ be a rectangle which strictly contains $\overline{\Omega_k}$. In this rectangle, we consider the function

$$\phi : (x_1,x_2) \mapsto K_1 \cos(\pi x_1/2a) \frac{\cosh(\pi x_2/2a)}{\cosh(\pi kL/2a)}.$$

As we did in the previous section, it is possible to choose K_1 large enough so that, using the maximum principle, we get

$$|\omega_k - \omega_\infty| \leq \phi \text{ in } \Omega_k, \tag{3.15}$$

and then

$$\forall k_0 \in \mathbb{N}^*, \forall k \geq k_0, \; \|\omega_k - \omega_\infty\|_{L^\infty(\Omega_{k_0})} \leq K_1 \frac{\cosh(\pi k_0 L/2a)}{\cosh(\pi kL/2a)}.$$

We now want the same kind of inequality for ψ_k. We define $\phi_k = \psi_k - \psi_\infty$. Then

$$\begin{aligned} -\Delta\phi_k &= (\omega_k - \omega_\infty) \quad \text{in } \Omega_k, \\ \phi_k &= 0 \quad \text{on } \Gamma_k, \\ \phi_k &= -\psi_\infty \quad \text{on } \Sigma_k. \end{aligned} \tag{3.16}$$

We consider $\varepsilon > 0$ such that

$$\overline{\Omega_k} \subset\subset (-a+\varepsilon, a-\varepsilon) \times [-kL,kL], \tag{3.17}$$

and we define the function

$$\phi_\varepsilon : (x_1,x_2) \mapsto K_2 \cos\left(\frac{\pi x_1}{2(a-\varepsilon)}\right) \frac{\cosh(\pi x_2/2a)}{\cosh(\pi kL/2a)}, \tag{3.18}$$

with a positive $K_2 \in \mathbb{R}$ to be specified further. We have

$$-\Delta\phi_\varepsilon = \left(\frac{\pi}{2(a-\varepsilon)}\right)^2 \phi_\varepsilon - \left(\frac{\pi}{2a}\right)^2 \phi_\varepsilon = \frac{\pi^2\varepsilon(2a-\varepsilon)}{4a^2(a-\varepsilon)^2}\, \phi_\varepsilon.$$

From (3.17) and (3.18), we can choose K_2 so that

$$\begin{array}{rcll} -\Delta\phi_\varepsilon & \geq & \phi & \text{in } \Omega_k, \\ \phi_\varepsilon & \geq & |\psi_\infty| & \text{on } \partial\Omega_k. \end{array}$$

Combining (3.15) and (3.16) in the above system yields that

$$\begin{array}{rll} -\Delta(\phi_\varepsilon - \phi_k) & \geq 0 & \text{in } \Omega_k, \\ \phi_\varepsilon - \phi_k & \geq 0 & \text{on } \partial\Omega_k. \end{array}$$

In the same way, we also have that

$$\begin{array}{rll} -\Delta(-\phi_\varepsilon - \phi_k) & \leq 0 & \text{in } \Omega_k, \\ -\phi_\varepsilon - \phi_k & \leq 0 & \text{on } \partial\Omega_k. \end{array}$$

Using the maximum principle, it finally comes that $|\phi_k| \leq \phi_\varepsilon$ i.e.

$$\forall k_0 \in \mathbb{N}^*, \forall k \geq k_0,\ \|\psi_k - \psi_\infty\|_{L^\infty(\Omega_{k_0})} \leq K_2' \frac{\cosh(\pi k_0 L/2a)}{\cosh(\pi k L/2a)}, \tag{3.19}$$

for some positive constant K_2'.

For $1 < \beta < \infty$, by applying a classical regularity result on elliptic equations (see [34, Theorem 9.14] for instance), we have that

$$\begin{aligned} \|\psi_k - \psi_\infty\|_{W^{2,\beta}(\Omega_{k_0})} &\leq c(\|\psi_k - \psi_\infty\|_{L^\beta(\Omega_{k_0})} + \|\omega_k - \omega_\infty\|_{L^\beta(\Omega_{k_0})}) \\ &\leq C\frac{\cosh(\pi k_0 L/2a)}{\cosh(\pi k L/2a)}. \end{aligned}$$

Since $u_k = \operatorname{curl}\psi_k$ and $u_\infty = \operatorname{curl}\psi_\infty$, the above inequality implies that

$$\|u_k - u_\infty\|_{W^{1,\beta}(\Omega_{k_0})^2} \leq C\frac{\cosh(\pi k_0 L/2a)}{\cosh(\pi k L/2a)} \leq K e^{-(\frac{\pi}{2a}L)k}. \tag{3.20}$$

For $\beta > 2$, we can use the Sobolev embedding

$$W^{1,\beta}(\Omega_{k_0}) \subset L^\infty(\Omega_{k_0}),$$

to get the L^∞ estimate for the velocity u_k.

We have now to estimate the difference $p_k - p_\infty$. To achieve this, we write the variational formulation corresponding to (3.13) :

$$\forall v \in H_0^1(\Omega_{k_0})^2, \quad \mu\int_{\Omega_{k_0}} \nabla u_k : \nabla v\, dx - \int_{\Omega_{k_0}} p_k(\nabla\cdot v)\, dx = \int_{\Omega_{k_0}} f\cdot v\, dx \tag{3.21}$$

In the above equation, for any smooth $v : \Omega_k \to \mathbb{R}^2$, we have denoted by ∇v the matrix

$$(\nabla v)_{i,j} = \frac{\partial v_i}{\partial x_j}, \quad (i,j \in \{1,2\})$$

and, for two matrices $A = (a_{i,j})_{i,j\in\{1,2\}}$ and $B = (a_{i,j})_{i,j\in\{1,2\}}$, we have denoted by $A : B$ the scalar product

$$A : B = \sum_{i,j\in\{1,2\}} a_{i,j} b_{i,j}.$$

Now, the periodic solution (u_∞, p_∞) defined on Ω can be naturally extended to Ω_k (see also [15, Lemma 2.4]) in such a way that it satisfies on Ω_k :

$$\forall v \in H_0^1(\Omega_{k_0})^2, \quad \mu \int_{\Omega_{k_0}} \nabla u_\infty : \nabla v \, dx - \int_{\Omega_{k_0}} p_\infty (\nabla \cdot v) \, dx = \int_{\Omega_{k_0}} f \cdot v \, dx. \tag{3.22}$$

Combining (3.21) and (3.22) yields that

$$\mu \int_{\Omega_{k_0}} (\nabla u_k - \nabla u_\infty) : \nabla v \, dx = \int_{\Omega_{k_0}} (p_k - p_\infty) \nabla \cdot v \, dx.$$

From (3.20) and the above equation, we get that

$$\int_{\Omega_{k_0}} (p_k - p_\infty) \nabla \cdot v \, dx \leq C e^{-(\frac{\pi}{2a} L)k} \|v\|_{H_0^1(\Omega_k)}$$

for any $v \in H_0^1(\Omega_k)^2$. This inequality and the inf-sup property (or LBB condition, see [35] or [9] for instance) imply that

$$\forall k_0 \in \mathbb{N}^*, \forall k \geq k_0, \quad \|p_k - p_\infty\|_{L^2(\Omega_{k_0})} \leq C' e^{-(\frac{\pi}{2a} L)k}$$

with C' only depending on Ω_{k_0}. □

3.4 Conclusion

This paper does not completely answer to the question raised in the introduction. Indeed, the physical situation considered in the oil extraction would necessitate to extend the present work in three directions :

- consider the three-dimensional case,
- consider more general boundary conditions,
- consider the Navier-Stokes model instead of the Stokes problem.

It seems to us that each of these extensions is really challenging.

However, numerical simulations tend to confirm, in the Navier-Stokes case, the results proved for the Stokes equations. We use Fluent to solve the Navier-Stokes equations in three dimensions for a gasoil flow in a pump. We study two cases : the first one is a six stages pump with velocity inlet and pressure outlet ; the second one is a one stage pump with periodic conditions at the entrance and exit. We plot the velocity magnitude of the fluid in the pump, at a given radius. On Figure 3.2 the velocity profile appears to be the same in the fourth and fifth stages of the six stages pump. It indicates that the velocity profile in a stage seems also to converge in the 3D Navier-Stokes model.

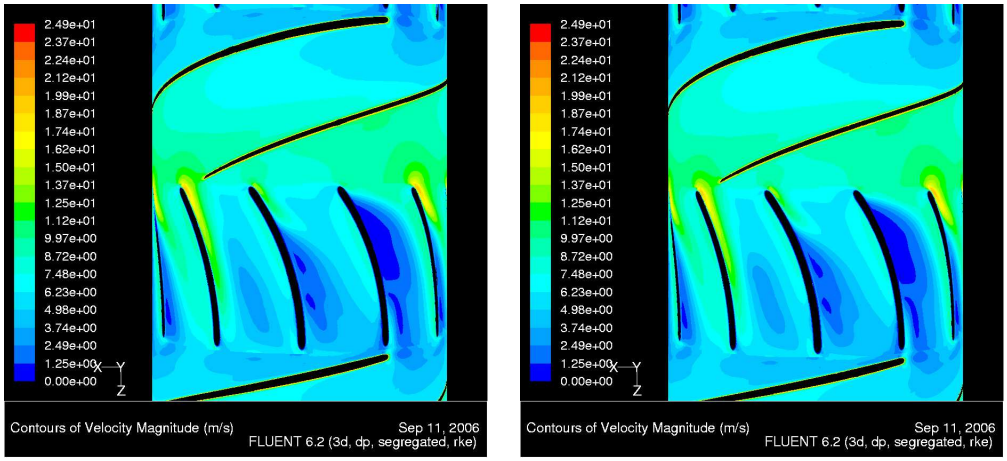

FIG. 3.2: Velocity magnitude profile in the fourth (left) and fifth (right) stage of a six stages pump.

Figure 3.3 shows that the velocity profile in the fifth stage of a six stages pump and in a single stage with periodic conditions are very alike. It is a strong indication that our convergence result should also hold for the Navier-Stokes problem (maybe with another rate of convergence). Though our simulation is limited to six stages, because of calculation costs, we could expect even better results with a bigger geometry.

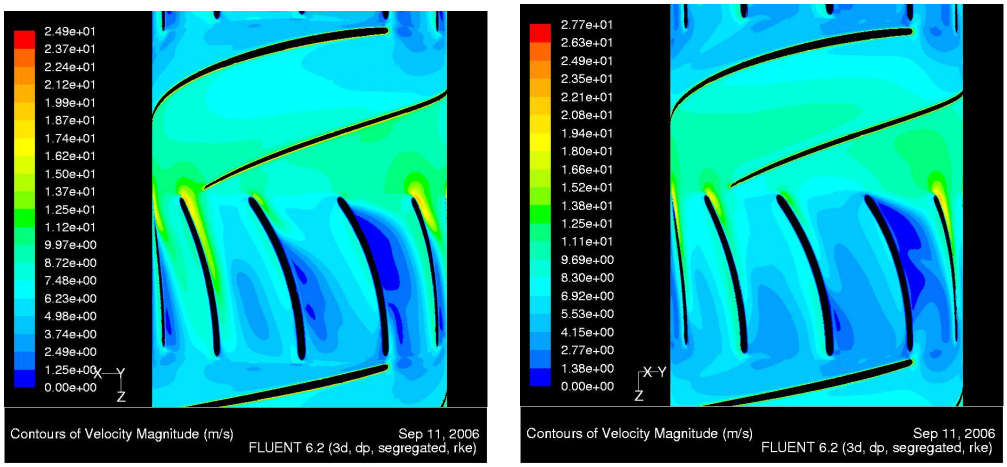

FIG. 3.3: Velocity magnitude profile in the fifth stage of a six stages pump (left) and in a single stage with periodic conditions (right, the stage is displayed periodically).

Deuxième partie

Evaluation du gradient

Chapitre 4

Calcul d'un gradient incomplet de la fonction coût

Dans ce chapitre on s'intéresse au calcul d'une valeur approchée du gradient de la fonction coût. Dans un premier temps on utilise une formule d'intégration sur des bords variables pour développer l'expression du gradient. Puis on extrait de la nouvelle formulation les termes que l'on sait calculer et on fait abstraction de ceux que l'on ne peut pas calculer. Enfin on explicite la méthode de calcul des termes retenus.

On reprend ici les notations du chapitre d'introduction. Le problème d'optimisation considéré consiste à maximiser le gain de pression par unité de longueur dans un étage de pompe. Par conséquent on cherche à maximiser la fonction coût J définie par :

$$J(x_C) = \frac{\int_S p - \int_E p}{L} \tag{4.1}$$

où S désigne la surface de sortie d'un étage de pompe, E, sa surface d'entrée, L, sa longueur axiale et p, la pression statique. x_C est l'ensemble des variables de contrôle de la géométrie de la pompe pris dans l'espace des ensembles de variables de contrôle admissibles, c'est-à-dire tels que le domaine Ω généré soit admissible ($\Omega \in \Omega_{adm}$).
Etant donné que la longueur L est une fonction explicite des variables de contrôle, il est possible de calculer de façon exacte le terme L' intervenant dans le gradient J' de la fonction coût. Pour calculer la différentielle du numérateur, on utilise une méthode de gradient incomplet.

Notations : étant donné qu'on ne va plus considérer dans le reste de ce chapitre que le numérateur de la fonction coût, on simplifie les notations pour l'ensemble de cette partie (II Evaluation du gradient), en désignant par J, non plus la fonction coût, mais son seul numérateur : $J = \int_S p - \int_E p$.

4.1 Formules d'intégration sur des bords variables

Les résultats énoncés ci-dessous sont tirés de [38].
Soit une famille $h \in [0, h_M[\to \Phi(h)$ de $\mathcal{C}^1$-difféomorphismes de $\mathbb{R}^N$ dans lui-même vérifiant :

$$\Phi : h \in [0, h_M[\to W^{1,\infty}(\mathbb{R}^N) \text{ dérivable en 0 avec } \Phi(0) = Id, \Phi'(0) = \mathcal{V}.$$

On note Γ_h le bord de $O_h = \Phi(h)(O)$ où O est un ouvert fixe de classe au moins $\mathcal{C}^1$, et de bord Γ. Compte tenu de ces premières définitions, $\mathcal{V}$ sera désigné par la suite comme *vecteur champ de déformation*. On note n_h la normale unitaire à Γ_h orientée vers l'extérieur du domaine d'écoulement. On souhaite dériver

une expression du type $h \to G(h) = \int_{\Gamma_h} g(h)$ où $g(h) : \Gamma_h \to \mathbb{R}$ est donnée.
Dans le cas où g peut s'exprimer sous la forme $g(h) = W(h).n_h$ on a un premier résultat simple :

Théorème 4.1.1. *Si $h \to W(h) \in W^{1,1}(\mathbb{R}^N, \mathbb{R}^N)$ est dérivable en 0 et si $div(W(0)) \in W^{1,1}$, alors*

$$G'(0) = \int_\Gamma \left(W'(0).n + div(W(0))\mathcal{V}.\mathcal{N}\right). \tag{4.2}$$

En supposant assez de régularité pour O, g et la famille des $\Phi(h)$, on peut montrer, pour toute intégrale de bord, le théorème suivant.

Théorème 4.1.2. *Soit $\mathcal{N}$ une extension unitaire, de classe $\mathcal{C}^1$, à $\mathbb{R}^N$ entier de la normale n à Γ. Supposons O de classe $\mathcal{C}^3$, $h \to \Phi(h) \in \mathcal{C}^2$ dérivable en 0 avec $\Phi(0) = Id$, $\Phi'(0) = \mathcal{V}$, et $h \to g(h) \in W^{1,1}(\mathbb{R}^N)$ dérivable en 0 avec $g(0) \in W^{2,1}(\mathbb{R}^N)$. Alors, $h \to G(h)$ est dérivable en 0 et, si on note $g'(0) = \frac{\partial}{\partial h}_{|h=0} g(h)$, on a :*

$$G'(0) = \int_\Gamma \left(g'(0) + \nabla g(0).\mathcal{N}\mathcal{V}.\mathcal{N} + g(0)H\mathcal{V}.\mathcal{N}\right) \tag{4.3}$$

où $H = div(\mathcal{N})$ est la courbure moyenne de Γ.

Remarque 4.1.1. *Ce sont généralement les termes de la forme g' que l'on est dans l'incapacité de calculer et que l'on sera donc amené à omettre dans l'expression du gradient incomplet.*

4.2 Calcul incomplet de gradient

Le domaine Ω considéré est un étage de pompe. Son bord $\partial\Omega$ est partitionné de la façon suivante :

$$\partial\Omega = E \cup S \cup D \cup C$$

où E est l'entrée de l'étage, S est sa sortie, C est constitué des bords non déformés au cours de l'optimisation (il ne s'agit donc que du carter), et D est constitué des bords déformés au cours de l'optimisation à l'exception de E et S (il s'agit donc des aubes et du moyeu).
Utilisant cette partition, on peut commencer par transformer la fonction coût :

$$J = \int_S p - \int_E p = \int_{E\cup S} pn_z = \int_{\partial\Omega} pn_z - \int_D pn_z - \int_C pn_z$$

et comme le carter est un cylindre d'axe $0e_3$, la composante axiale du vecteur normal au carter est nulle donc :

$$J = \int_{\partial\Omega} pn_z - \int_D pn_z. \tag{4.4}$$

On note $J_1 = \int_{\partial\Omega} pn_z$ et $J_2 = \int_D pn_z$.

4.2.1 Dérivation de l'intégrale sur le bord du domaine : $J_1 = \int_{\partial\Omega} pn_z$

Le principe du calcul d'un gradient incomplet est de ne pas tenir compte dans l'expression du gradient des termes que l'on ne sait pas calculer. En ce sens, on "supprime de l'information" et il peut sembler judicieux d'essayer d'en ré-introduire par un autre biais. La seule variable d'état considérée dans l'expression (4.4) est la pression p, d'où l'idée d'utiliser les équations de Navier-Stokes pour faire apparaître la vitesse U.
Ainsi, on introduit les équations de Navier-Stokes dans l'expression de J_1 :

$$\begin{aligned}
J_1 = \int_{\partial\Omega} pn_z = \int_{\Omega} \frac{\partial p}{\partial z} &= \int_{\Omega} \left(\mu \triangle u_z - \rho \sum_{j=1}^{3} u_j \frac{\partial u_z}{\partial x_j} \right) \\
&= \mu \left(\int_{\partial\Omega} 1.\frac{\partial u_z}{\partial n} - \int_{\Omega} \nabla u_z.\nabla 1 \right) - \rho \sum_{j=1}^{3} \left(\int_{\partial\Omega} u_z u_j n_j - \int_{\Omega} \frac{\partial u_j}{\partial x_j} u_z \right) \\
&= \mu \int_{\partial\Omega} \frac{\partial u_z}{\partial n} - \rho \left(\int_{\partial\Omega} u_z U.n - \int_{\Omega} u_z \nabla.U \right) \\
&= \mu \int_{\partial\Omega} \frac{\partial u_z}{\partial n} - \rho \int_{\partial\Omega} u_z U.n
\end{aligned}$$

car $\nabla.U = 0$. On note $J_3 = \int_{\partial\Omega} \frac{\partial u_z}{\partial n}$ et $J_4 = \int_{\partial\Omega} u_z U.n$.

Dérivation du terme $J_3 = \int_{\partial\Omega} \frac{\partial u_z}{\partial n}$

En utilisant $\partial\Omega = E \cup S \cup D \cup C$, on obtient :

$$J_3' = \left(\int_{\partial\Omega} \frac{\partial u_z}{\partial n} \right)' = \left(\int_{\partial\Omega} \nabla u_z.n \right)' = \left(\int_{E\cup S} \nabla u_z.n \right)' + \left(\int_{C} \nabla u_z.n \right)' + \left(\int_{D} \nabla u_z.n \right)'. \tag{4.5}$$

Etant donné que le carter n'est pas déformé, on a immédiatement $(\int_C \nabla u_z.n)' = \int_C (\nabla u_z)'.n$ et, ne pouvant pas calculer ce terme avec le logiciel choisi, on ne le prend pas en compte.
Le théorème 4.1.2 permet de développer les autres termes. On suppose que l'on a choisi une extension $\mathcal{N}$ unitaire et $\mathcal{C}^1$ de n, de plus telle que $D\mathcal{N}$ soit symétrique. Alors :

$$\left(\int_{E\cup S} \nabla u_z.n \right)' = \int_{E\cup S} (\nabla u_z)'.\mathcal{N} + \int_{E\cup S} \nabla u_z.\mathcal{N}' + \int_{E\cup S} \nabla(\nabla u_z.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} + \int_{E\cup S} \nabla u_z.\mathcal{N} H \mathcal{V}.\mathcal{N}.$$

Le premier terme de l'expression de droite ne peut être calculé, on l'omet. Le dernier terme est nul car la courbure moyenne H des surfaces d'entrée et de sortie est nulle. Reste donc, en utilisant la symétrie de $D\mathcal{N}$:

$$\begin{aligned}
\left(\int_{E\cup S} \nabla u_z.n \right)' &\simeq \int_{E\cup S} \nabla u_z.\mathcal{N}' + \int_{E\cup S} \nabla(\nabla u_z.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} \\
&= \int_{E\cup S} \nabla u_z.\mathcal{N}' + \int_{E\cup S} ({}^t D(\nabla u_z).\mathcal{N} + {}^t D\mathcal{N}.\nabla u_z).\mathcal{N}\mathcal{V}.\mathcal{N} \\
&= \int_{E\cup S} \nabla u_z.\mathcal{N}' + \int_{E\cup S} (D^2 u_z.\mathcal{N} + {}^t\nabla u_z.D\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N}.
\end{aligned}$$

Or, si $\mathcal{N}$ est unitaire et $D\mathcal{N}$ symétrique, alors ${}^t D\mathcal{N}.\mathcal{N} = D\mathcal{N}.\mathcal{N} = 0$ (on le vérifiera plus loin), ce qui permet de supprimer le terme en $D\mathcal{N}$. Par ailleurs Fluent ne permet pas d'avoir accès à des dérivées

secondes et, dans la mesure où on n'a pas réussi à mettre en place un schéma numérique assez stable pour les calculer (cf. §7.4), on fait abstraction du terme en D^2u_z. Enfin, dans le cas particulier des surfaces d'entrée et de sortie, la normale étant $n = \pm e_3$, on définit l'extension $\mathcal{N}$ qui, en un point quelconque, est égale à n en la projection de ce point (cf. §4.3). Il s'agit d'une extension unitaire, $\mathcal{C}^1$ et telle que $D\mathcal{N}(=0)$ est symétrique. Alors $\mathcal{N}' = 0$ sur les surfaces d'entrée et de sortie. Ainsi :

$$\left(\int_{E\cup S} \nabla u_z.n\right)' \simeq 0.$$

On procède de même pour le terme $(\int_D \nabla u_z.n)'$, à la seule différence que $\mathcal{N}'$, ainsi que la courbure moyenne H, ne sont pas nuls sur D. On obtient :

$$\left(\int_D \nabla u_z.n\right)' \simeq \int_D \nabla u_z.\mathcal{N}' + \int_D \nabla u_z.\mathcal{N}H\mathcal{V}.\mathcal{N},$$

et enfin :

$$J_3' = \left(\int_{\partial\Omega} \frac{\partial u_z}{\partial n}\right)' \simeq \int_D \nabla u_z.\mathcal{N}' + \int_D \nabla u_z.\mathcal{N}H\mathcal{V}.\mathcal{N}. \tag{4.6}$$

Dérivation du terme $J_4 = \displaystyle\int_{\partial\Omega} u_z U.n$

Comme précédemment, on a :

$$J_4' = \left(\int_{\partial\Omega} u_z U.n\right)' = \left(\int_{E\cup S} u_z U.n\right)' + \left(\int_C u_z U.n\right)' + \left(\int_D u_z U.n\right)'. \tag{4.7}$$

Le carter n'est pas déformé, donc $(\int_C u_z U.n)' = \int_C (u_z U.n)'$. La condition de non glissement implique au carter que $U = 0$ et en particulier $u_z = 0$. Un bref calcul donne alors facilement $\int_C (u_z U.n)' = 0$.
En utilisant à nouveau le théorème 4.1.2, et en supposant que l'on a choisi $\mathcal{N}$ une extension de n unitaire, $\mathcal{C}^1$, telle que $D\mathcal{N}$ soit symétrique, on a :

$$\left(\int_{E\cup S} u_z U.n\right)' = \int_{E\cup S} (u_z U.n)' + \int_{E\cup S} \nabla(u_z U.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} + \int_{E\cup S} u_z U.\mathcal{N}H\mathcal{V}.\mathcal{N}.$$

Dans le terme $\int_{E\cup S}(u_z U.n)'$ on fait abstraction des termes faisant intervenir des dérivées de U et u_z. Par ailleurs, la courbure moyenne des surfaces d'entrée et de sortie est nulle. Il reste :

$$\left(\int_{E\cup S} u_z U.n\right)' \simeq \int_{E\cup S} u_z U.\mathcal{N}' + \int_{E\cup S} \nabla(u_z U.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N}.$$

Comme précédemment, on peut simplifier l'expression puisque $\mathcal{N}' = 0$ sur les surfaces d'entrée et de sortie. On développe le second terme :

$$\begin{aligned}
\int_{E\cup S} \nabla(u_z U.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} &= \int_{E\cup S} U.\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_{E\cup S} u_z \nabla(U.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} \\
&= \int_{E\cup S} U.\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_{E\cup S} u_z({}^tDU.\mathcal{N} + {}^tU.D\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} \\
&= \int_{E\cup S} U.\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_{E\cup S} u_z({}^tDU.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N},
\end{aligned}$$

la dernière simplification venant du fait que ${}^tD\mathcal{N}.\mathcal{N} = D\mathcal{N}.\mathcal{N} = 0$. On procède de la même façon pour D, à ceci près que H n'est pas nulle sur D, il vient :

$$\left(\int_D u_z U.n\right)' \simeq \int_D u_z U.\mathcal{N}' + \int_D U.\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_D u_z({}^tDU.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} + \int_D u_z u.\mathcal{N}H\mathcal{V}.\mathcal{N}.$$

Par ailleurs, de par la condition de non glissement aux parois, on connaît la vitesse u sur D : $U = \omega_D \wedge X$, où $\omega_D = \omega$ sur les parois en rotation, c'est-à-dire le moyeu et les aubes du rotor, et $\omega_D = 0$ sur les parois fixes, c'est-à-dire le moyeu et les aubes du stator. Etant donné que la rotation se fait selon l'axe $0e_3$, on a, en particulier, $u_z = 0$ et trois des intégrales ci-dessus sont nulles. On obtient alors :

$$\left(\int_D u_z U.n\right)' \simeq \int_D (\omega_D \wedge X).\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N}.$$

D'où l'expression finale de J_4' :

$$\begin{aligned} J_4' &= \left(\int_{\partial\Omega} u_z U.n\right)' \\ &\simeq \int_{E\cup S} U.\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_{E\cup S} u_z({}^tDU.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} + \int_D (\omega_D \wedge X).\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N}. \end{aligned} \tag{4.8}$$

4.2.2 Dérivation de l'intégrale sur les parties déformées : $J_2 = \int_D pn_z$

On applique le théorème 4.1.1 :

$$J_2' = \left(\int_D pn_z\right)' = \int_D (pn_z)' + \int_D \nabla(p\mathcal{N}_z).\mathcal{N}\mathcal{V}.\mathcal{N} + \int_D p\mathcal{N}_z H\mathcal{V}.\mathcal{N}.$$

Lorsqu'on développe, on écarte l'intégrale contenant p' que l'on ne peut pas calculer :

$$J_2' = \left(\int_D pn_z\right)' \simeq \int_D p\mathcal{N}_z' + \int_D \mathcal{N}_z \nabla p.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_D p\nabla\mathcal{N}_z.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_D p\mathcal{N}_z H\mathcal{V}.\mathcal{N}.$$

Par ailleurs, on prouvera dans la suite (cf. propriété 4.3.3) que, pour une extension $\mathcal{N}$ de n unitaire, $\mathcal{C}^1$, et telle que $D\mathcal{N}$ soit symétrique, $\nabla\mathcal{N}_z.\mathcal{N} = 0$, donc il reste :

$$J_2' = \left(\int_D pn_z\right)' \simeq \int_D p\mathcal{N}_z' + \int_D \mathcal{N}_z \nabla p.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_D p\mathcal{N}_z H\mathcal{V}.\mathcal{N}. \tag{4.9}$$

4.2.3 Expression finale du gradient incomplet

En sommant les différents termes obtenus précédemment, on obtient l'expression suivante :

$$\begin{aligned} J' \simeq \mu\left(\int_D \nabla u_z.\mathcal{N}' + \int_D \nabla u_z.\mathcal{N}H\mathcal{V}.\mathcal{N}\right) - \rho\left(\int_{E\cup S} U.\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_{E\cup S} u_z({}^tDU.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N}\right) \\ - \rho\int_D (\omega_D \wedge X).\mathcal{N}\nabla u_z.\mathcal{N}\mathcal{V}.\mathcal{N} - \left(\int_D p\mathcal{N}_z' + \int_D \mathcal{N}_z\nabla p.\mathcal{N}\mathcal{V}.\mathcal{N} + \int_D p\mathcal{N}_z H\mathcal{V}.\mathcal{N}\right). \end{aligned} \tag{4.10}$$

Dans cette expression on a accès par le biais de Fluent, directement ou par application de schémas numériques, aux expressions de U, u_z, DU, ∇u_z, p et ∇p. Il reste à déterminer analytiquement les expressions de $\mathcal{N}'$, $\mathcal{N}_z'$, $\mathcal{V}.\mathcal{N}$ et H. C'est l'objet de la suite de ce chapitre.

4.3 Extensions du vecteur normal et du vecteur champ de déformation

La formule (4.3) nécessite de définir une extension du champ de déformation $\mathcal{V}$, sinon dans $\mathbb{R}^N$ entier, du moins dans un voisinage tubulaire de $\partial\Omega$ (alors qu'on ne l'a exprimé que sur $\partial\Omega$ jusqu'ici). On doit aussi définir une extension $\mathcal{N}$ du vecteur normal n qui ait de "bonnes" propriétés.
Dans [23] on trouve les définitions et propriétés permettant de définir ces extensions.

Définition 4.3.1. *Soit $\Omega \subset \mathbb{R}^n$ tel que ni Ω, ni ${}^C\Omega$, ni $\partial\Omega$ ne sont vides. On définit* **la distance signée** *b (ou* **distance orientée***) ainsi :*

$$\forall X \in \mathbb{R}^n,\ b(X) = d_\Omega(X) - d_{{}^C\Omega}(X)$$

où d_Ω et $d_{{}^C\Omega}$ sont les distances usuelles à Ω et ${}^C\Omega$.

Théorème 4.3.1. *Soit $\Omega \subset \mathbb{R}^n$ tel que ni Ω, ni ${}^C\Omega$, ni $\partial\Omega$ ne sont vides. Soit $X \in \mathbb{R}^n$. Si $\nabla b(X)$ existe, alors il existe un unique $P_{\partial\Omega}(X) \in \partial\Omega$ tel que ($|.|$ désignant la distance euclidienne) :*

$$b(X) = \begin{cases} |P_{\partial\Omega}(X) - X| & si\ X \in int({}^C\Omega) \\ 0 & si\ X \in \partial\Omega \\ -|P_{\partial\Omega}(X) - X| & si\ X \in int(\Omega) \end{cases}$$

et

$$\nabla b(X) = \frac{X - P_{\partial\Omega}(X)}{b(X)}\ si\ X \notin \partial\Omega.$$

Théorème 4.3.2. *Soit $\Omega \subset \mathbb{R}^n$ tel que ni Ω, ni ${}^C\Omega$, ni $\partial\Omega$ ne sont vides et $\partial\Omega$ est de classe $\mathcal{C}^1$. Si la normale n est localement lipschitzienne, alors, pour tout $P \in \partial\Omega$, il existe un voisinage $vois(P)$ de P tel que tout $X \in vois(P)$ admette une et une seule projection $P_{\partial\Omega}(X)$ sur $\partial\Omega$. De plus,*

$$\forall X \in vois(P),\ \nabla b(X) = n(P_{\partial\Omega}(X)),$$

∇b est lipschitzienne dans $vois(P)$, et $b \in W^{2,\infty}(vois(P))$.

Ainsi ∇b est un candidat naturel pour une extension de n. On définit donc, dans un voisinage de $\partial\Omega$, l'extension :

$$\mathcal{N}(X) = \begin{cases} n(X) & \text{si } X \in \partial\Omega, \\ \nabla b(X) = \frac{X - P_{\partial\Omega}(X)}{b(X)} & \text{si } X \notin \partial\Omega. \end{cases}$$

$\mathcal{N}$ est clairement unitaire.

Propriété 4.3.3. *1. $D\mathcal{N} = D^2 b$ est symétrique.*
2. $D\mathcal{N}.\mathcal{N} = {}^tD\mathcal{N}.\mathcal{N} = 0$.
3. $\nabla\mathcal{N}_z.\mathcal{N} = 0$.

Preuve. 1. $\forall (i,j),\ \dfrac{\partial \mathcal{N}_i}{\partial x_j} = \dfrac{\partial(\nabla b)_i}{\partial x_j} = \dfrac{\partial^2 b}{\partial x_j \partial x_i}$. Or $\dfrac{\partial^2 b}{\partial x_i \partial x_j}$ et $\dfrac{\partial^2 b}{\partial x_j \partial x_i}$ sont égales au sens des distributions donc en tant que fonctions car elles sont toutes deux dans $L^2(vois(P))$. Donc $\forall (i,j),\ \dfrac{\partial \mathcal{N}_i}{\partial x_j} = \dfrac{\partial^2 b}{\partial x_i \partial x_j} = \dfrac{\partial(\nabla b)_j}{\partial x_i} = \dfrac{\partial \mathcal{N}_j}{\partial x_i}$.
2. ${}^tD\mathcal{N}.\mathcal{N} = \left[\sum_i \mathcal{N}_i \dfrac{\partial \mathcal{N}_i}{\partial x_j}\right]_j = \left[\dfrac{1}{2}\sum_i \dfrac{\partial \mathcal{N}_i^2}{\partial x_j}\right]_j = \left[\dfrac{1}{2}\dfrac{\partial ||\mathcal{N}||^2}{\partial x_j}\right]_j = 0$ car $\mathcal{N}$ est unitaire.
3. $\nabla\mathcal{N}_z.\mathcal{N} = \nabla(\mathcal{N}.e_3).\mathcal{N} = (D\mathcal{N}.e_3).\mathcal{N} = {}^te_3.D\mathcal{N}.\mathcal{N} = 0$ d'après 1. □

Enfin on définit, dans le même voisinage de $\partial\Omega$, l'extension V du champ de déformation $\mathcal{V}$:

$$V(X) = \begin{cases} \mathcal{V}(X) & \text{si } X \in \partial\Omega, \\ \mathcal{V}(P_{\partial\Omega}(X)) & \text{si } X \notin \partial\Omega. \end{cases}$$

4.4 Dérivation par rapport au domaine de l'extension de la normale

4.4.1 Résultat utile

On énonce ici une autre proposition extraite de [38].
Soit $h \in [0, h_M[\to \Phi(h) \in \mathcal{C}^1 \cap W^{1,\infty}(\mathbb{R}^N, \mathbb{R}^N)$ dérivable en 0 avec $\Phi(0) = Id, \Phi'(0) = \mathcal{V}$. On note, comme en (4.1), Γ_h le bord de $O_h = \Phi(h)(O)$. On commence par une définition.

Définition 4.4.1. *Soit g une fonction de classe $\mathcal{C}^1$ sur le bord Γ d'un ouvert borné de classe au moins $\mathcal{C}^1$. On définit son* **gradient tangentiel** *par*

$$\nabla_\Gamma g = \nabla \tilde{g} - (\nabla \tilde{g}.n)n \text{ sur } \Gamma \tag{4.11}$$

où $\tilde{g} \in \mathcal{C}^1(\mathbb{R}^N)$ est un prolongement de g.

Remarque 4.4.1. *Cette définition est indépendante du prolongement $\tilde{g}$ choisi.*

Proposition 4.4.1. *On suppose O de classe $\mathcal{C}^2$. Soit $\mathcal{N} \in \mathcal{C}^1(\mathbb{R}^N, \mathbb{R}^N)$ une extension de la normale unitaire n à Γ. Pour toute extension $h \in [0, h_M[\to \mathcal{N}_h \in \mathcal{C}^0(\mathbb{R}^N, \mathbb{R}^N)$ de la normale à Γ_h, dérivable en 0 avec $\mathcal{N}_0 \in \mathcal{C}^1(\mathbb{R}^N, \mathbb{R}^N)$, on a*

$$\frac{\partial \mathcal{N}_h}{\partial h}_{|h=0} = -\nabla_\Gamma(\mathcal{V}.n) - (D\mathcal{N}_0.n)\mathcal{V}.n \text{ sur } \Gamma. \tag{4.12}$$

4.4.2 Premiers éléments de calcul

On considère une extension unitaire $\mathcal{N}$ de la normale n à $\partial\Omega$. Si $\mathcal{N}$ respecte les hypothèses de la proposition 4.4.1, alors on a : $\mathcal{N}' = -\nabla_\Gamma(\mathcal{V}.n) - (D\mathcal{N}_0.n)\mathcal{V}.n$ sur $\partial\Omega$. Si de plus on suppose $\mathcal{N}_0$ unitaire et $D\mathcal{N}_0$ symétrique, $D\mathcal{N}_0.n = D\mathcal{N}_0.\mathcal{N}_0 = 0$ sur Γ et il reste simplement

$$\mathcal{N}' = -\nabla_\Gamma(\mathcal{V}.n) \text{ sur } \partial\Omega.$$

Par définition $\nabla_\Gamma(\mathcal{V}.n) = \nabla(V.\mathcal{N}) - (\nabla(V.\mathcal{N}).n)n$ d'où, en utilisant la symétrie de $D\mathcal{N}$:

$$\begin{aligned} \mathcal{N}' &= -({}^tDV.\mathcal{N} + {}^tD\mathcal{N}.V) + (((^tDV.\mathcal{N} + {}^tD\mathcal{N}.V).n)\,n) \\ &= -({}^tDV.\mathcal{N} + D\mathcal{N}.V) + (((^tDV.\mathcal{N}).n + {}^tV.D\mathcal{N}.n)\,n) \\ &= -{}^tDV.\mathcal{N} - D\mathcal{N}.V + ((^tDV.\mathcal{N}).n)\,n \end{aligned}$$

car $D\mathcal{N}.n = D\mathcal{N}.\mathcal{N}_{|\partial\Omega} = 0$.

Proposition 4.4.2. $({}^tDV.\mathcal{N}).n = 0$ *sur $\partial\Omega$ (le vecteur ${}^tDV.\mathcal{N}$ appartient au plan tangent à $\partial\Omega$).*

En conséquence de cette proposition, si l'on revient à l'expression de $\mathcal{N}'$, il reste :

$$\mathcal{N}' = -{}^tDV.\mathcal{N} - D\mathcal{N}.V = -\nabla(V.\mathcal{N}). \tag{4.13}$$

On prouve maintenant la proposition 4.4.2.

Preuve. La nappe $\partial\Omega$ est suffisamment régulière pour être localement considérée comme une surface de paramétrisation

$$(t,u) \to M(t,u) = (M_1(t,u), M_2(t,u), M_3(t,u)).$$

Dans le cas des aubes les paramètres sont (t,r) et dans le cas du moyeu, ce sont (t,θ). Soit $P \in \partial\Omega$ et $vois(P)$ un voisinage de P tel que tout point de $vois(P)$ admette une et une seule projection sur $\partial\Omega$. Soit $X = (x_1, x_2, x_3) \in vois(P)$ et $P_{\partial\Omega}(X)$ sa projection sur $\partial\Omega$. On pose $P_{\partial\Omega}(X) = M(t_X, u_X)$. Ce couple (t_X, u_X) est obtenu en minimisant la fonction de deux variables $(t,u) \to ||X - M(t,u)||^2$. Donc (t_X, u_X) est solution de

$$(X - M(t,u)).\frac{\partial M(t,u)}{\partial t} = 0, \tag{4.14}$$

$$(X - M(t,u)).\frac{\partial M(t,u)}{\partial u} = 0. \tag{4.15}$$

On considère l'équation (4.14) et on la dérive par rapport à la variable x_i, $i \in \{1,2,3\}$. Etant donné que $0 = (X - M(t_X, u_X)).\dfrac{\partial M(t_X, u_X)}{\partial t} = \sum_{j=1}^{3} (x_j - M_j(t_X, u_X))\dfrac{\partial M_j(t_X, u_X)}{\partial t}$, on a :

$$\begin{aligned}
&\frac{\partial M_i}{\partial t} - \frac{\partial t_X}{\partial x_i}\sum_{j=1}^{3}\left(\frac{\partial M_j}{\partial t}\right)^2 - \frac{\partial u_X}{\partial x_i}\sum_{j=1}^{3}\left(\frac{\partial M_j}{\partial u}\frac{\partial M_j}{\partial t}\right) \\
&\qquad - \frac{\partial t_X}{\partial x_i}\sum_{j=1}^{3}\left((x_j - M_j)\frac{\partial^2 M_j}{\partial t^2}\right) - \frac{\partial u_X}{\partial x_i}\sum_{j=1}^{3}\left((x_j - M_j)\frac{\partial^2 M_j}{\partial u \partial t}\right) \\
&= \frac{\partial M_i}{\partial t} - \frac{\partial t_X}{\partial x_i}\left|\left|\frac{\partial M}{\partial t}\right|\right|^2 - \frac{\partial u_X}{\partial x_i}\frac{\partial M}{\partial u}.\frac{\partial M}{\partial t} - \frac{\partial t_X}{\partial x_i}(X - M).\frac{\partial^2 M}{\partial t^2} - \frac{\partial u_X}{\partial x_i}(X - M).\frac{\partial^2 M}{\partial u \partial t} \\
&= 0.
\end{aligned}$$

De la même façon, en dérivant l'équation (4.15) par rapport à x_i, on obtient

$$\frac{\partial M_i}{\partial u} - \frac{\partial u_X}{\partial x_i}\left|\left|\frac{\partial M}{\partial u}\right|\right|^2 - \frac{\partial t_X}{\partial x_i}\frac{\partial M}{\partial u}.\frac{\partial M}{\partial t} - \frac{\partial u_X}{\partial x_i}(X - M).\frac{\partial^2 M}{\partial u^2} - \frac{\partial t_X}{\partial x_i}(X - M).\frac{\partial^2 M}{\partial u \partial t} = 0.$$

Donc $DM_i = \begin{pmatrix} \dfrac{\partial M_i}{\partial t} \\ \dfrac{\partial M_i}{\partial u} \end{pmatrix}$ permet de calculer $\begin{pmatrix} \dfrac{\partial t_X}{\partial x_i} \\ \dfrac{\partial u_X}{\partial x_i} \end{pmatrix}$ en résolvant le système linéaire

$$A.\begin{pmatrix} \dfrac{\partial t_X}{\partial x_i} \\ \dfrac{\partial u_X}{\partial x_i} \end{pmatrix} = DM_i.$$

Lorsque l'on se place au bord $\partial\Omega$, $X - M(t_X, u_X) = 0$ donc

$$A = \begin{pmatrix} \left|\left|\dfrac{\partial M}{\partial t}\right|\right|^2 & \dfrac{\partial M}{\partial u}.\dfrac{\partial M}{\partial t} \\ \dfrac{\partial M}{\partial u}.\dfrac{\partial M}{\partial t} & \left|\left|\dfrac{\partial M}{\partial u}\right|\right|^2 \end{pmatrix}.$$

Alors, si $X = M(t_X, u_X)$ n'est pas un point stationnaire, A est inversible (cf. [42]) et il est possible d'écrire que

$$\forall i \in \{1,2,3\}, \begin{pmatrix} \dfrac{\partial t_X}{\partial x_i} \\ \dfrac{\partial u_X}{\partial x_i} \end{pmatrix} \text{ permet de calculer } DM_i \text{ en résolvant le système linéaire}$$

$$\begin{pmatrix} \dfrac{\partial t_X}{\partial x_i} \\ \dfrac{\partial u_X}{\partial x_i} \end{pmatrix} = A^{-1}.DM_i.$$

On en revient au vecteur ${}^tDV.\mathcal{N} = \left[\sum_j \left(\dfrac{\partial V_j}{\partial x_i}\mathcal{N}_j\right)\right]_i$. On rappelle que l'extension V a été choisie de sorte que $V(X) = \mathcal{V}(P_{\partial\Omega}(X)) = \mathcal{V}(M(t_X, u_X))$. Ainsi, il s'agit d'une fonction de t_X et u_X : $V(X) = V(t_X, u_X)$. Soit $i \in \{1,2,3\}$,

$$\begin{aligned} \sum_j \left(\frac{\partial V_j}{\partial x_i}\mathcal{N}_j\right) &= \sum_j \left(\frac{\partial V_j}{\partial t}\frac{\partial t_X}{\partial x_i} + \frac{\partial V_j}{\partial u}\frac{\partial u_X}{\partial x_i}\right)\mathcal{N}_j \\ &= \sum_j \left({}^tDV_j.A^{-1}DM_i\right)\mathcal{N}_j \\ &= {}^t\left(\sum_j \mathcal{N}_j DV_j\right).A^{-1}DM_i. \end{aligned}$$

A est symétrique, donc A^{-1} aussi. On introduit la notation suivante : $W := A^{-1}.\left(\sum_j \mathcal{N}_j DV_j\right)$, et on désigne par $\tilde{D}M$ la matrice de vecteurs colonnes $\dfrac{\partial M}{\partial t}$ et $\dfrac{\partial M}{\partial u}$. Alors ${}^tDV.\mathcal{N} = [{}^tDM_i.W]_i = \tilde{D}M.W$. On souhaite montrer que ce vecteur est tangent à la surface $\partial\Omega$.
Les vecteurs $\dfrac{\partial M}{\partial t}$ et $\dfrac{\partial M}{\partial u}$ sont tangents à $\partial\Omega$, donc le vecteur $\tilde{n} = \dfrac{\partial M}{\partial t} \wedge \dfrac{\partial M}{\partial u}$ est normal à $\partial\Omega$.

$$({}^tDV.\mathcal{N}).\tilde{n} = \tilde{D}M.W.\tilde{n} = {}^t\tilde{n}.\tilde{D}M.W = 0$$

car les colonnes de $\tilde{D}M$ sont des vecteurs tangents à $\partial\Omega$ et $\tilde{n}$ est normal à $\partial\Omega$.
En conclusion on a bien prouvé l'objet de la proposition 4.4.2 : ${}^tDV.\mathcal{N}$ appartient au plan tangent à $\partial\Omega$. □

4.4.3 Calcul du terme $\mathcal{N}' = -\nabla(V.\mathcal{N})$

On conserve ici les notations introduites au paragraphe 4.4.2.
D'après les définitions des extensions V et $\mathcal{N}$ introduites en 4.3, pour tout $X \in vois(P)$, V dépend uniquement du couple (t_X, u_X), tandis que $\mathcal{N}$ dépend de X et de (t_X, u_X). Ainsi $\mathcal{N}' = -\nabla(V.\mathcal{N}) = -\nabla(V(t_X,u_X).\mathcal{N}(X,t_X,u_X)) = -{}^tDV(t_X,u_X).\mathcal{N}(X,t_X,u_X) - D\mathcal{N}(X,t_X,u_X).V(t_X,u_X)$.

Le premier terme est obtenu par le calcul suivant :

$$
\begin{aligned}
{}^tDV(t_X,u_X).\mathcal{N}(X,t_X,u_X) &= \begin{pmatrix} \frac{\partial V_1}{\partial x_1} & \frac{\partial V_2}{\partial x_1} & \frac{\partial V_3}{\partial x_1} \\ \frac{\partial V_1}{\partial x_2} & \frac{\partial V_2}{\partial x_2} & \frac{\partial V_3}{\partial x_2} \\ \frac{\partial V_1}{\partial x_3} & \frac{\partial V_2}{\partial x_3} & \frac{\partial V_3}{\partial x_3} \end{pmatrix} .\mathcal{N}(X,t_X,u_X) \\
&= \begin{pmatrix} \frac{\partial V_1}{\partial t}\frac{\partial t_X}{\partial x_1}+\frac{\partial V_1}{\partial u}\frac{\partial u_X}{\partial x_1} & \frac{\partial V_2}{\partial t}\frac{\partial t_X}{\partial x_1}+\frac{\partial V_2}{\partial u}\frac{\partial u_X}{\partial x_1} & \frac{\partial V_3}{\partial t}\frac{\partial t_X}{\partial x_1}+\frac{\partial V_3}{\partial u}\frac{\partial u_X}{\partial x_1} \\ \frac{\partial V_1}{\partial t}\frac{\partial t_X}{\partial x_2}+\frac{\partial V_1}{\partial u}\frac{\partial u_X}{\partial x_2} & \frac{\partial V_2}{\partial t}\frac{\partial t_X}{\partial x_2}+\frac{\partial V_2}{\partial u}\frac{\partial u_X}{\partial x_2} & \frac{\partial V_3}{\partial t}\frac{\partial t_X}{\partial x_2}+\frac{\partial V_3}{\partial u}\frac{\partial u_X}{\partial x_2} \\ \frac{\partial V_1}{\partial t}\frac{\partial t_X}{\partial x_3}+\frac{\partial V_1}{\partial u}\frac{\partial u_X}{\partial x_3} & \frac{\partial V_2}{\partial t}\frac{\partial t_X}{\partial x_3}+\frac{\partial V_2}{\partial u}\frac{\partial u_X}{\partial x_3} & \frac{\partial V_3}{\partial t}\frac{\partial t_X}{\partial x_3}+\frac{\partial V_3}{\partial u}\frac{\partial u_X}{\partial x_3} \end{pmatrix} .\mathcal{N}(X,t_X,u_X) \\
&= \begin{pmatrix} \frac{\partial t_X}{\partial x_1} & \frac{\partial u_X}{\partial x_1} \\ \frac{\partial t_X}{\partial x_2} & \frac{\partial u_X}{\partial x_2} \\ \frac{\partial t_X}{\partial x_3} & \frac{\partial u_X}{\partial x_3} \end{pmatrix} . \begin{pmatrix} \frac{\partial V_1}{\partial t} & \frac{\partial V_2}{\partial t} & \frac{\partial V_3}{\partial t} \\ \frac{\partial V_1}{\partial u} & \frac{\partial V_2}{\partial u} & \frac{\partial V_3}{\partial u} \end{pmatrix} .\mathcal{N}(X,t_X,u_X).
\end{aligned}
$$

On peut choisir entre deux méthodes pour calculer le deuxième terme. L'une consiste à faire le calcul, un peu plus complexe que le précédent et introduit ci-dessous ; l'autre demande à employer la seconde forme fondamentale, et fait l'objet du prochain paragraphe.
Par le calcul, on obtient :

$$
\begin{aligned}
&D\mathcal{N}(X).V(t_X,u_X) \\
&= \begin{pmatrix} \frac{\partial \mathcal{N}_1}{\partial x_1}+\frac{\partial \mathcal{N}_1}{\partial t}\frac{\partial t_X}{\partial x_1}+\frac{\partial \mathcal{N}_1}{\partial u}\frac{\partial u_X}{\partial x_1} & \frac{\partial \mathcal{N}_1}{\partial x_2}+\frac{\partial \mathcal{N}_1}{\partial t}\frac{\partial t_X}{\partial x_2}+\frac{\partial \mathcal{N}_1}{\partial u}\frac{\partial u_X}{\partial x_2} & \frac{\partial \mathcal{N}_1}{\partial x_3}+\frac{\partial \mathcal{N}_1}{\partial t}\frac{\partial t_X}{\partial x_3}+\frac{\partial \mathcal{N}_1}{\partial u}\frac{\partial u_X}{\partial x_3} \\ \frac{\partial \mathcal{N}_2}{\partial x_1}+\frac{\partial \mathcal{N}_2}{\partial t}\frac{\partial t_X}{\partial x_1}+\frac{\partial \mathcal{N}_2}{\partial u}\frac{\partial u_X}{\partial x_1} & \frac{\partial \mathcal{N}_2}{\partial x_2}+\frac{\partial \mathcal{N}_2}{\partial t}\frac{\partial t_X}{\partial x_2}+\frac{\partial \mathcal{N}_2}{\partial u}\frac{\partial u_X}{\partial x_2} & \frac{\partial \mathcal{N}_2}{\partial x_3}+\frac{\partial \mathcal{N}_2}{\partial t}\frac{\partial t_X}{\partial x_3}+\frac{\partial \mathcal{N}_2}{\partial u}\frac{\partial u_X}{\partial x_3} \\ \frac{\partial \mathcal{N}_3}{\partial x_1}+\frac{\partial \mathcal{N}_3}{\partial t}\frac{\partial t_X}{\partial x_1}+\frac{\partial \mathcal{N}_3}{\partial u}\frac{\partial u_X}{\partial x_1} & \frac{\partial \mathcal{N}_3}{\partial x_2}+\frac{\partial \mathcal{N}_3}{\partial t}\frac{\partial t_X}{\partial x_2}+\frac{\partial \mathcal{N}_3}{\partial u}\frac{\partial u_X}{\partial x_2} & \frac{\partial \mathcal{N}_3}{\partial x_3}+\frac{\partial \mathcal{N}_3}{\partial t}\frac{\partial t_X}{\partial x_3}+\frac{\partial \mathcal{N}_3}{\partial u}\frac{\partial u_X}{\partial x_3} \end{pmatrix} .V(,u_X).
\end{aligned}
$$

Seconde forme fondamentale

On expose ici un bref résumé de notions extraites de [42].

Définition 4.4.2. *Soit* $T = \lambda\frac{\partial M}{\partial t} + \mu\frac{\partial M}{\partial u}$ *un vecteur du plan tangent à* $\partial\Omega$ *au point* $M(t,u)$. *La* **seconde forme fondamentale** *de* $\partial\Omega$ *au point* M *est la forme quadratique* Φ_2^M *définie par :*

$$\Phi_2^M(T) = L\lambda^2 + 2M\lambda\mu + N\mu^2$$

avec $L = \left\|\frac{\partial M}{\partial t}\wedge\frac{\partial M}{\partial u}\right\|^{-1}.\left[\frac{\partial M}{\partial t},\frac{\partial M}{\partial u},\frac{\partial^2 M}{\partial t^2}\right]$, $M = \left\|\frac{\partial M}{\partial t}\wedge\frac{\partial M}{\partial u}\right\|^{-1}.\left[\frac{\partial M}{\partial t},\frac{\partial M}{\partial u},\frac{\partial^2 M}{\partial u\partial t}\right]$ *et* $N = \left\|\frac{\partial M}{\partial t}\wedge\frac{\partial M}{\partial u}\right\|^{-1}.\left[\frac{\partial M}{\partial t},\frac{\partial M}{\partial u},\frac{\partial^2 M}{\partial u^2}\right]$.
On appelle φ_M *la forme polaire de la seconde forme fondamentale* Φ_2^M, *définie par :*

$$
\begin{aligned}
\varphi_M : \partial\Omega^2 &\to \mathbb{R} \\
(T_1,T_2) = (\lambda_1\frac{\partial M}{\partial t}+\mu_1\frac{\partial M}{\partial u},\lambda_2\frac{\partial M}{\partial t}+\mu_2\frac{\partial M}{\partial u}) &\mapsto \varphi_M(T_1).T_2 = L\lambda_1\lambda_2 + M(\lambda_1\mu_2+\lambda_2\mu_1) + N\mu_1\mu_2
\end{aligned}
$$

Définition 4.4.3. *Un vecteur non nul* $T = \lambda \frac{\partial M}{\partial t} + \mu \frac{\partial M}{\partial u}$ *tangent à* $\partial\Omega$ *en* M *est dit* **principal** *s'il existe un réel* ρ *vérifiant :* $\varphi_M(T) = \rho T$ *et* ρ *est la* **courbure principale** *relative à la direction du vecteur* T.

La propriété suivante est tirée de [24] ou [34] :

Propriété 4.4.3. *La seconde forme fondamentale associée à* $\partial\Omega$ *coïncide avec la matrice* $D^2 b = D\mathcal{N}$. *De plus, les valeurs propres de la matrice hessienne sont d'une part les courbures principales et d'autre part* 0, *associé au vecteur propre* $\mathcal{N} = \nabla b$.

Remarque 4.4.2. *Que* 0 *soit valeur propre associée au vecteur propre* $\mathcal{N} = \nabla b$ *résulte de la propriété 4.3.3.*

Calcul de $D\mathcal{N}.V$ grâce à la seconde forme fondamentale

On rappelle qu'au-delà du terme $\mathcal{N}'$, ce sont des termes de la forme $W.\mathcal{N}'$ que l'on souhaite calculer dans la formulation retenue pour le gradient incomplet de la fonction coût. Ainsi, pour en revenir au calcul du terme isolé $D\mathcal{N}.V$, c'est en fait un terme de la forme $W.D\mathcal{N}.V$ que l'on souhaite calculer.
On décompose V et W de la façon suivante : $V = \lambda_v \frac{\partial M}{\partial t} + \mu_v \frac{\partial M}{\partial u} + \nu_v \mathcal{N}$, $W = \lambda_w \frac{\partial M}{\partial t} + \mu_w \frac{\partial M}{\partial u} + \nu_w \mathcal{N}$.
Alors

$$\begin{aligned}
W.D\mathcal{N}.V &= W.D\mathcal{N}.\left(\lambda_v \frac{\partial M}{\partial t} + \mu_v \frac{\partial M}{\partial u}\right) && \text{car } D\mathcal{N}.\mathcal{N} = 0 \\
&= (D\mathcal{N}.W).\left(\lambda_v \frac{\partial M}{\partial t} + \mu_v \frac{\partial M}{\partial u}\right) && \text{car } D\mathcal{N} \text{ est symétrique} \\
&= \left(D\mathcal{N}.\left(\lambda_w \frac{\partial M}{\partial t} + \mu_w \frac{\partial M}{\partial u}\right)\right).\left(\lambda_v \frac{\partial M}{\partial t} + \mu_v \frac{\partial M}{\partial u}\right) && \text{car } D\mathcal{N}.\mathcal{N} = 0 \\
&= \left(\lambda_w \frac{\partial M}{\partial t} + \mu_w \frac{\partial M}{\partial u}\right).D\mathcal{N}.\left(\lambda_v \frac{\partial M}{\partial t} + \mu_v \frac{\partial M}{\partial u}\right) \\
&= \varphi_M\left(\lambda_v \frac{\partial M}{\partial t} + \mu_v \frac{\partial M}{\partial u}\right).\left(\lambda_w \frac{\partial M}{\partial t} + \mu_w \frac{\partial M}{\partial u}\right) \\
&= L\lambda_v\lambda_w + M(\lambda_v\mu_w + \lambda_w\mu_v) + N\mu_v\mu_w
\end{aligned}$$

où L, M et N sont définis ci-avant.
On peut déterminer λ_w et μ_w (et de même λ_v et μ_v) de la façon suivante. Etant donné que $W = \lambda_w \frac{\partial M}{\partial t} + \mu_w \frac{\partial M}{\partial u} + \nu_w \mathcal{N}$ avec $\frac{\partial M}{\partial t}$ et $\frac{\partial M}{\partial u}$ tangents à $\partial\Omega$ et $\mathcal{N}$ normal à $\partial\Omega$, on a :

$$\begin{cases}
W.\frac{\partial M}{\partial t} &= \lambda_w \left\|\frac{\partial M}{\partial t}\right\|^2 + \mu_w \frac{\partial M}{\partial u}.\frac{\partial M}{\partial t}, \\
W.\frac{\partial M}{\partial u} &= \lambda_w \frac{\partial M}{\partial t}.\frac{\partial M}{\partial u} + \mu_w \left\|\frac{\partial M}{\partial u}\right\|^2.
\end{cases}$$

On résout ce système pour obtenir :

$$\lambda_w = \frac{1}{\left\|\frac{\partial M}{\partial t} \wedge \frac{\partial M}{\partial u}\right\|^2}.\left(W.\frac{\partial M}{\partial t}\left\|\frac{\partial M}{\partial u}\right\|^2 - W.\frac{\partial M}{\partial u}\frac{\partial M}{\partial u}.\frac{\partial M}{\partial t}\right),$$

$$\mu_w = \frac{1}{\left\|\frac{\partial M}{\partial t} \wedge \frac{\partial M}{\partial u}\right\|^2}.\left(W.\frac{\partial M}{\partial u}\left\|\frac{\partial M}{\partial t}\right\|^2 - W.\frac{\partial M}{\partial t}\frac{\partial M}{\partial t}.\frac{\partial M}{\partial u}\right).$$

Le dénominateur $\left|\left|\frac{\partial M}{\partial t} \wedge \frac{\partial M}{\partial u}\right|\right|^2$ est non nul si M n'est pas un point stationnaire (cf. [42]).

4.4.4 Calcul de la courbure moyenne H

Certains des termes calculés précédemment peuvent être ré-employés pour déterminer la courbure moyenne H. Dans [42] on trouve la formule suivante :

$$H = \frac{1}{2}\frac{LG + EN - 2FM}{K^2}$$

où L, M, N ont été définis avant, et où $K = \left|\left|\frac{\partial M}{\partial t} \wedge \frac{\partial M}{\partial u}\right|\right|$, $E = \left|\left|\frac{\partial M}{\partial t}\right|\right|^2$, $F = \frac{\partial M}{\partial t}.\frac{\partial M}{\partial u}$ et $G = \left|\left|\frac{\partial M}{\partial u}\right|\right|^2$. Cette définition de la courbure moyenne n'est pas tout à fait la même que $H = div(\mathcal{N})$. En effet, l'expression ci-dessus correspond à une demi-somme de courbures normales relatives à deux directions orthogonales, tandis que dans le cas $H = div(\mathcal{N})$, on détermine la somme de telles courbures. Ici on utilisera, pour déterminer la courbure moyenne, la formule :

$$H = \frac{LG + EN - 2FM}{K^2}. \tag{4.16}$$

4.5 Conclusion

Dans ce chapitre on a déterminé une expression incomplète du gradient de la fonction coût. Outre les termes relatifs aux équations d'état que l'on peut obtenir numériquement, cette expression contient un certain nombre de termes d'ordre géométrique. Avec quelques considérations de géométrie différentielle on a indiqué comment calculer, de façon générale, ces quantités. Il reste à les formuler au cas par cas pour l'extrados, la cambrure et le moyeu. Dans le chapitre 5 on commence par s'intéresser au champ de déformation $\mathcal{V}$, puis dans le chapitre 6 on détermine la dérivée $\mathcal{N}'$ de l'extension de la normale et la courbure H.

Chapitre 5

Calcul du champ de déformation induit par le déplacement d'un point de contrôle

On souhaite expliciter le champ de déformation induit par le déplacement d'un point de contrôle de l'extrados, de la cambrure, et du moyeu. On rappelle que ce champ de déformation est le vecteur $\mathcal{V}$ qui apparaît dans les formules de dérivées du chapitre 4. On travaille dans un premier temps en deux dimensions, on passera par la suite en trois dimensions.

5.1 Théorème des fonctions implicites

On note $(s_c(t), z_c(t))$ et $(s_e(t), z_e(t))$ les B-splines de la cambrure et de l'extrados. On considère un point $(s_e(t_0), z_e(t_0))$, $t_0 \in]0,1[$ et $t_1 \in]0,1[$ tel que $(s_c(t_1), z_c(t_1))$ soit la projection de $(s_e(t_0), z_e(t_0))$ sur la cambrure.

Théorème 5.1.1. *Il existe deux voisinages O_0 de t_0 et O_1 de t_1 tels que $O_0 \times O_1 \subset]0,1[\times]0,1[$ et tels qu'il existe une unique application $\varphi : O_0 \to O_1$, de classe $\mathcal{C}^1$, inversible, telle que*

$$\forall (t_0, t_1) \in O_0 \times O_1,\ t_1 = \varphi(t_0) \Leftrightarrow t_0 = \varphi^{-1}(t_1).$$

Preuve. La cambrure est convexe et située au-dessus de l'extrados donc on peut montrer que la projection sur la cambrure d'un point de l'extrados est aussi le point de la cambrure le plus proche de ce point de l'extrados. Autrement dit, avec les notations choisies, t_1 est l'unique réel qui minimise la fonction $\psi : t \longmapsto (s_c(t) - s_e(t_0))^2 + (z_c(t) - z_e(t_0))^2$. Mais que $(s_c(t_1), z_c(t_1))$ soit la projection de $(s_e(t_0), z_e(t_0))$ équivaut aussi à ce que

$$\psi'(t_1) = s_c'(t_1)(s_c(t_1) - s_e(t_0)) + z_c'(t_1)(z_c(t_1) - z_e(t_0)) = 0, \tag{5.1}$$

que l'on note encore $F(t_0, t_1) = 0$. F est $\mathcal{C}^1$ sur $]0,1[\times]0,1[$ et on a $\dfrac{\partial F(t_0,t_1)}{\partial t_1} = s_c''(t_1)(s_c(t_1) - s_e) + s_c'^2(t_1) + z_c''(t_1)(z_c(t_1) - z_e) + z_c'^2(t_1)$. Il faut prouver que $\dfrac{\partial F(t_0,t_1)}{\partial t_1} \neq 0$ pour pouvoir appliquer le théorème des fonctions implicites à F.
On remarque que

$$F(t_0, t_1) = \overrightarrow{\tau_c(t_1)}.\overrightarrow{M_e(t_0)M_c(t_1)}$$

où $\overrightarrow{\tau_c(t_1)}$ est un vecteur directeur de la tangente à la cambrure en t_1, et où $M_e(t_0) = (s_e(t_0), z_e(t_0))$ et $M_c(t_1) = (s_c(t_1), z_c(t_1))$.

On note, pour la cambrure, a l'abscisse curviligne, $\overrightarrow{n_c}$ le vecteur normal orienté vers l'intérieur de la cambrure, et R le rayon de courbure. Alors

$$\begin{aligned}\frac{\partial F}{\partial t_1} &= \frac{\partial \overrightarrow{\tau_c}}{\partial a}.\overrightarrow{M_e(t_0)M_c(t_1)}\frac{\partial a}{\partial t_1} + \overrightarrow{\tau_c}^2 \\ &= \left(\frac{1}{R}\overrightarrow{n_c}.|M_e(t_0)M_c(t_1)|\overrightarrow{n_c} + \frac{\partial a}{\partial t_1}\right)\frac{\partial a}{\partial t_1} \\ &= \left(\frac{1}{R}|M_e(t_0)M_c(t_1)| + \frac{\partial a}{\partial t_1}\right)\frac{\partial a}{\partial t_1}.\end{aligned}$$

Chaque terme étant positif, et $\partial a/\partial t_1 > 0$ car il n'y a pas de point d'inflexion, on a donc $\dfrac{\partial F(t_0,t_1)}{\partial t_1} \neq 0$.
On peut dès lors appliquer le théorème des fonctions implicites : il existe deux voisinages O_0 de t_0 et O_1 de t_1 tels que $O_0 \times O_1 \subset]0,1[\times]0,1[$ et une unique application $\varphi : O_0 \mapsto O_1$, de classe $\mathcal{C}^1$, telle que $\forall (t_0,t_1) \in O_0 \times O_1$, $(F(t_0,t_1) = 0 \Leftrightarrow t_1 = \varphi(t_0))$. Et on a alors aussi :

$$\varphi'(t_0) = -\frac{\frac{\partial F(t_0,\varphi(t_0))}{\partial t_0}}{\frac{\partial F(t_0,\varphi(t_0))}{\partial t_1}}. \tag{5.2}$$

On sait déjà que $\dfrac{\partial F(t_0,\varphi(t_0))}{\partial t_1} > 0$, donc $\varphi'(t_0)$ est du signe opposé à $\dfrac{\partial F(t_0,\varphi(t_0))}{\partial t_0}$.

$$\frac{\partial F}{\partial t_0} = -\overrightarrow{\tau_c}(t_1).\overrightarrow{\tau_e}(t_0),$$

où $\overrightarrow{\tau_e}(t_0)$ est un vecteur directeur de la tangente à l'extrados en t_0. On a, par hypothèse issue d'une contrainte géométrique, $\lim_{(t_0,t_1)\to(0,0)} \overrightarrow{\tau_c}(t_1).\overrightarrow{\tau_e}(t_0) = 0^+$. On raisonne par l'absurde et pour cela on suppose que le produit scalaire s'annule et change de signe en un certain point. Alors, si on désigne par D la tangente à l'extrados en ce point, D est perpendiculaire à la tangente à la cambrure et coupe donc la cambrure en deux, laissant l'attaque et la fuite chacune d'un côté. Etant donné que l'extrados est convexe, il est situé d'un seul côté de sa tangente D donc s'il est du côté de l'attaque, il ne peut relier la fuite, et inversement. Cette situation est absurde : le produit scalaire reste donc toujours positif, $\dfrac{\partial F(t_0,\varphi(t_0))}{\partial t_0} < 0$, et enfin $\varphi'(t_0) > 0$.
Ainsi φ est $\mathcal{C}^1$, strictement croissante sur $O_0 \times O_1$, donc bijective sur cet ouvert et on peut donc définir φ^{-1} tel que $\forall (t_0,t_1) \in O_0 \times O_1$, $(F(t_0,t_1) = 0 \Leftrightarrow t_0 = \varphi^{-1}(t_1))$. □

Compte tenu des hypothèses de convexité de la cambrure et de l'extrados, la fonction φ, qui au paramètre t_0 associe le paramètre t_1 du projeté, peut être définie globalement.

5.2 Déplacement d'un point de contrôle de l'extrados

5.2.1 Champ de déformation induit à l'extrados

On se place dans un repère cartésien et les coordonnées des points de contrôle de l'extrados sont notées, de façon simplifiée, $(s_j,z_j)_j$. On note $(s(t),z(t))$ la B-spline obtenue grâce à ces points de contrôle. On considère que l'on déplace un point de contrôle (s_k,z_k) de sa position initiale (s_k^0,z_k^0) dans une direction $u_k = (u_{s,k},u_{z,k})$, de sorte que le nouveau point de contrôle obtenu est $(s_k^h,z_k^h) = (s_k^0,z_k^0) + h.u_k = (s_k^0 + hu_{s,k}, z_k^0 + hu_{z,k})$.

On cherche à déterminer l'application $V_k = (V_{s,k}, V_{z,k}) : \mathbb{R}^2 \to \mathbb{R}^2$ telle que, pour chaque intervalle Γ_j de B-spline :

$$\begin{cases} s^h(t) = s(t) + hV_{s,k}(s(t), z(t)) + o(h), \\ z^h(t) = z(t) + hV_{z,k}(s(t), z(t)) + o(h). \end{cases}$$

Le point (s_k, z_k) intervient dans le calcul de 4 intervalles de B-splines. On note Γ_0 l'intervalle qui prend pour points de contrôle les points (s_{k-3}, z_{k-3}), (s_{k-2}, z_{k-2}), (s_{k-1}, z_{k-1}) et (s_k, z_k).
Après transformation du point (s_k^0, z_k^0), l'expression de la B-spline devient, sur Γ_0, pour $t \in [0,1[$:

$$\begin{cases} s^h(t) = s_{k-3}p_3(t) + s_{k-2}p_2(t) + s_{k-1}p_1(t) + (s_k^0 + hu_{s,k})p_0(t), \\ z^h(t) = z_{n-3}p_3(t) + z_{k-2}p_2(t) + z_{k-1}p_1(t) + (z_k^0 + hu_{z,k})p_0(t). \end{cases}$$

Ainsi sur l'intervalle Γ_0 de l'extrados, on a $V_k = (u_{s,k}p_0(t), u_{z,k}p_0(t))$.
On passe à l'intervalle Γ_1 qui prend pour points de contrôle les points (s_{k-2}, z_{k-2}), (s_{k-1}, z_{k-1}), (s_k, z_k) et(s_{k+1}, z_{k+1}). De la même façon on a $V_k = (u_{s,k}p_1(t), u_{z,k}p_1(t))$ pour $t \in [0,1[$ sur Γ_1. On vérifie facilement que le champ V_k est $\mathcal{C}^2$ à la limite entre Γ_0 et Γ_1 (car $p_0(1) = p_1(0)$, $p_0'(1) = p_1'(0)$ et $p_0''(1) = p_1''(0)$).
On procède de même sur Γ_2 et Γ_3. Par ailleurs il est naturel de prolonger V_k par 0 en dehors de ces 4 intervalles de B-spline et on vérifie alors qu'aux limites V_k est encore $\mathcal{C}^2$ (car $p_0(0) = p_3(1) = p_0'(0) = p_3'(1) = p_0''(0) = p_3''(1) = 0$).
Ainsi V_k restreint à l'extrados est $\mathcal{C}^2$ et on a :

$$V_k = \begin{cases} p_0(t)u_k & \text{sur } \Gamma_0, \\ p_1(t)u_k & \text{sur } \Gamma_1, \\ p_2(t)u_k & \text{sur } \Gamma_2, \\ p_3(t)u_k & \text{sur } \Gamma_3, \\ 0 & \text{sur les autres } \Gamma_j \text{ de l'extrados.} \end{cases}$$

5.2.2 Champ de déformation induit à l'intrados

On revient à la notation $(s_e(t), z_e(t))$ pour la B-spline de l'extrados, sur laquelle on considère un point $(s_e(t_0), z_e(t_0))$, $t_0 \in [0,1[$. On simplifie les notations en le désignant par (s_e, z_e). Soit $(s_c(t), z_c(t))$ la B-spline de la cambrure (qui est invariante quand on modifie l'extrados) et $t_1 \in [0,1[$ tel que $(s_c(t_1), z_c(t_1))$ soit la projection de (s_e, z_e) sur la cambrure. On a vu dans le paragraphe 5.1 que cela équivaut à $F(t_0, t_1) = 0$. On fait le choix ici de l'écrire $G(s_e, z_e, t_1) = 0$, où G est $\mathcal{C}^1$ sur un ouvert de $\mathbb{R}^3$ et où $\dfrac{\partial G(s_e, z_e, t_1)}{\partial t_1} = \dfrac{\partial F(t_0, t_1)}{\partial t_1} > 0$. On applique donc le théorème des fonctions implicites à G : il existe deux voisinages de (s_e, z_e) et de t_1 tels qu'on peut y écrire t_1 comme fonction explicite de (s_e, z_e) : $t_1 = t_1(s_e, z_e)$. On considère maintenant que le point (s_e, z_e) subit un déplacement de $h.V_k = (hV_{s,k}, hV_{z,k})$. Alors on note t_1^h le réel tel que $(s_c(t_1^h), z_c(t_1^h))$ soit la projection de $(s_e + hV_{s,k}, z_e + hV_{z,k})$ sur la cambrure. On peut écrire que :

$$t_1^h = t_1(s_e + hV_{s,k}, z_e + hV_{z,k}) = t_1(s_e, z_e) + hV_{s,k}\frac{\partial t_1(s_e, z_e)}{\partial s_e} + hV_{z,k}\frac{\partial t_1(s_e, z_e)}{\partial z_e} + o(h),$$

$$\begin{cases} s_c(t_1^h) = s_c(t_1) + [hV_{s,k}\frac{\partial t_1(s_e,z_e)}{\partial s_e} + hV_{z,k}\frac{\partial t_1(s_e,z_e)}{\partial z_e}]s_c'(t_1) + o(h), \\ z_c(t_1^h) = z_c(t_1) + [hV_{s,k}\frac{\partial t_1(s_e,z_e)}{\partial s_e} + hV_{z,k}\frac{\partial t_1(s_e,z_e)}{\partial z_e}]z_c'(t_1) + o(h). \end{cases}$$

Le point de l'intrados symétrique de (s_e, z_e) par rapport à la cambrure est

$$(s_i, z_i) = (2s_c(t_1) - s_e, 2z_c(t_1) - z_e). \tag{5.3}$$

Donc, dans le cas d'un déplacement à l'extrados, cela devient :

$$\begin{cases} s_i^h = 2s_c(t_1^h) - (s_e + hV_{s,k}), \\ z_i^h = 2z_c(t_1^h) - (z_e + hV_{z,k}). \end{cases}$$

Alors, en reportant les expressions précédemment trouvées, on obtient :

$$\begin{cases} s_i^h = s_i + h\mathcal{V}^e_{intrados,s} + o(h), \\ z_i^h = z_i + h\mathcal{V}^e_{intrados,z} + o(h), \end{cases}$$

$$\text{où} \begin{cases} \mathcal{V}^e_{i,s} = 2\left(V_{s,k}\frac{\partial t_1(s_e,z_e)}{\partial s_e} + V_{z,k}\frac{\partial t_1(s_e,z_e)}{\partial z_e}\right)s_c'(t_1) - V_{s,k}, \\ \mathcal{V}^e_{i,z} = 2\left(V_{s,k}\frac{\partial t_1(s_e,z_e)}{\partial s_e} + V_{z,k}\frac{\partial t_1(s_e,z_e)}{\partial z_e}\right)z_c'(t_1) - V_{z,k} \end{cases}$$

sont les composantes du champ de déformation $\overrightarrow{\mathcal{V}_i^e}$, à l'intrados, engendré par un déplacement à l'extrados.
Il reste en dernier lieu à déterminer les expressions $\dfrac{\partial t_1(s_e,z_e)}{\partial s_e}$ et $\dfrac{\partial t_1(s_e,z_e)}{\partial z_e}$. On revient à l'équation (5.1) et on la dérive par rapport à s_e. On obtient :

$$s_c''(t_1)\frac{\partial t_1}{\partial s_e}(s_c(t_1) - s_e) + s_c'(t_1)(s_c'(t_1)\frac{\partial t_1}{\partial s_e} - 1) + z_c''(t_1)\frac{\partial t_1}{\partial s_e}(z_c(t_1) - z_e) + z_c'^2(t_1)\frac{\partial t_1}{\partial s_e} = 0$$

d'où

$$\frac{\partial t_1(s_e,z_e)}{\partial s_e} = \frac{s_c'(t_1)}{s_c''(t_1)(s_c(t_1) - s_e) + s_c'^2(t_1) + z_c''(t_1)(z_c(t_1) - z_e) + z_c'^2(t_1)} = \frac{s_c'(t_1)}{\frac{\partial F}{\partial t_1}}.$$

De la même façon, on a

$$\frac{\partial t_1(s_e,z_e)}{\partial z_e} = \frac{z_c'(t_1)}{s_c''(t_1)(s_c(t_1) - s_e) + s_c'^2(t_1) + z_c''(t_1)(z_c(t_1) - z_e) + z_c'^2(t_1)} = \frac{z_c'(t_1)}{\frac{\partial F}{\partial t_1}}.$$

5.3 Déplacement d'un point de contrôle de la cambrure

On procède de la même façon que pour l'extrados.

5.3.1 Champ de déformation induit à la cambrure

On se place dans un repère cartésien et les coordonnées des points de contrôle de la cambrure sont notées, de façon simplifiée, $(s_j, z_j)_j$. La B-spline obtenue grâce à ces points de contrôle est notée $(s(t), z(t))$. On considère que l'on déplace un point (s_k, z_k) de sa position initiale (s_k^0, z_k^0) dans une direction $u_k = (u_{s,k}, u_{z,k})$, de sorte que le nouveau point de contrôle est $(s_k^h, z_k^h) = (s_k^0, z_k^0) + h.u_k = (s_k^0 + hu_{s,k}, z_k^0 + hu_{z,k})$.
Les déformations induites à la cambrure par le déplacement d'un de ses points de contrôle se calculent exactement de la même façon que les déformations induites à l'extrados par le déplacement d'un point de contrôle de celui-ci.
Ainsi l'application $V_k = (V_{s,k}, V_{z,k}) : \mathbb{R}^2 \to \mathbb{R}^2$ telle que, pour chaque intervalle Γ_j de la B-spline de la cambrure :

$$\begin{cases} s^h(t) = s(t) + hV_{s,k}(s(t), z(t)) + o(h), \\ z^h(t) = z(t) + hV_{z,k}(s(t), z(t)) + o(h), \end{cases}$$

est $\mathcal{C}^2$ et s'exprime comme suit :

$$V_k = \begin{cases} p_0(t)u_k & \text{sur } \Gamma_0, \\ p_1(t)u_k & \text{sur } \Gamma_1, \\ p_2(t)u_k & \text{sur } \Gamma_2, \\ p_3(t)u_k & \text{sur } \Gamma_3, \\ 0 & \text{sur les autres } \Gamma_j \text{ de la cambrure.} \end{cases}$$

5.3.2 Champ de déformation induit à l'intrados

On reprend la notation $(s_c(t), z_c(t))$ pour la B-spline de la cambrure, sur laquelle on considère un point $(s_c(t_1), z_c(t_1))$, $t_1 \in]0,1[$. Soit $(s_e(t), z_e(t))$ la B-spline de l'extrados (qui est invariante quand on modifie la cambrure) et $t_0 \in]0,1[$ tel que $(s_c(t_1), z_c(t_1))$ soit la projection de $(s_e(t_0), z_e(t_0))$ sur la cambrure. On considère que le point $(s_c(t_1), z_c(t_1))$ subit un déplacement de $h.V_k(t_1) = (hV_{s,k}(t_1), hV_{z,k}(t_1))$. Soit t_0^h tel que $(s_e(t_0^h), z_e(t_0^h))$ ait pour projection sur la cambrure $(s_c(t_1) + hV_{s,k}(t_1), z_c(t_1) + hV_{z,k}(t_1))$. Ceci se traduit par l'équation

$$\begin{aligned}(s_c'(t_1) + hV_{s,k}'(t_1))(s_c(t_1) + hV_{s,k}(t_1) - s_e(t_0^h)) \\ + (z_c'(t_1) + hV_{z,k}'(t_1))(z_c(t_1) + hV_{z,k}(t_1) - z_e(t_0)) = 0 \end{aligned} \quad (5.4)$$

que l'on note $F_h(t_0^h, t_1) = 0$.
Soit φ_h la bijection définie, d'après le théorème des fonctions implicites, sur un voisinage de (t_0^h, t_1) et telle que $t_1 = \varphi_h(t_0^h)$, ou, aussi bien, $t_0^h = \varphi_h^{-1}(t_1) = t_0 + Ah + o(h)$. On a alors :

$$\begin{aligned} t_1 &= \varphi_h(t_0^h) \\ &= \varphi_h(t_0 + Ah + o(h)) \\ &= \varphi_h(t_0) + A\varphi_h'(t_0)h + o(h) \\ &= \varphi(t_0) + Bh + A(\varphi'(t_0) + O(h))h + o(h) \\ &= t_1 + Bh + A\varphi'(t_0)h + o(h) \end{aligned}$$

d'où $B + A\varphi'(t_0) = 0$. Il reste à déterminer l'expression de B et à en déduire celle de A.
Soit $t_1^h = \varphi_h(t_0)$. Alors $F_h(t_0, t_1^h) = 0$. Mais

$$F_h(t_0, t_1^h) = F_h(t_0, t_1) + B\frac{\partial F_h(t_0, t_1)}{\partial t_1}h + o(h). \quad (5.5)$$

Par ailleurs, si on développe $F_h(t_0, t_1)$, on obtient :

$$F_h(t_0, t_1) = F(t_0, t_1) + (\overrightarrow{V_k}(t_1).\overrightarrow{\tau_c}(t_1) + \overrightarrow{V_k'}(t_1).\overrightarrow{M_e(t_0)M_c(t_1)})h + o(h).$$

Comme, par hypothèse, $F(t_0, t_1) = 0$, on reporte l'expression précédente dans (5.5) et on obtient :

$$\begin{aligned} F_h(t_0, t_1^h) &= 0 \\ &= (\overrightarrow{V_k}(t_1).\overrightarrow{\tau_c}(t_1) + \overrightarrow{V_k'}(t_1).\overrightarrow{M_e(t_0)M_c(t_1)})h + B\frac{\partial F_h(t_0, t_1)}{\partial t_1}h + o(h) \\ &= (\overrightarrow{V_k}(t_1).\overrightarrow{\tau_c}(t_1) + \overrightarrow{V_k'}(t_1).\overrightarrow{M_eM_c})h \\ &\quad + B(s_c''(t_1)(s_c(t_1) - s_e(t_0)) + s_c'^2(t_1) + z_c''(t_1)(z_c(t_1) - z_e(t_0)) + z_c'^2(t_1))h + o(h). \end{aligned}$$

On reconnaît, en facteur de Bh le terme $\dfrac{\partial F(t_0,t_1)}{\partial t_1}$. On obtient donc

$$B = -\frac{\overrightarrow{V_k}(t_1).\overrightarrow{\tau_c}(t_1) + \overrightarrow{V_k'}(t_1).\overrightarrow{M_eM_c}}{\frac{\partial F(t_0,t_1)}{\partial t_1}},$$

$$A = -\frac{B}{\varphi'(t_0)} = \frac{B}{\frac{\partial F/\partial t_0}{\partial F/\partial t_1}} = -\frac{\overrightarrow{V_k}(t_1).\overrightarrow{\tau_c}(t_1) + \overrightarrow{V_k'}(t_1).\overrightarrow{M_eM_c}}{\frac{\partial F(t_0,t_1)}{\partial t_0}} = \frac{\overrightarrow{V_k}(t_1).\overrightarrow{\tau_c}(t_1) + \overrightarrow{V_k'}(t_1).\overrightarrow{M_eM_c}}{\overrightarrow{\tau_c}(t_1).\overrightarrow{\tau_e}(t_0)}.$$

Maintenant on sait donc expliciter l'écriture $t_0^h = t_0 + Ah + o(h)$ et l'utiliser dans

$$\begin{cases} s_e(t_0^h) = s_e(t_0) + As_e'(t_0)h + o(h), \\ z_e(t_0^h) = z_e(t_0) + Az_e'(t_0)h + o(h). \end{cases}$$

Le point de l'intrados symétrique de $(s_e(t_0), z_e(t_0))$ par rapport à la cambrure est

$$(s_i, z_i) = (2s_c(t_1) - s_e(t_0), 2z_c(t_1) - z_e(t_0)). \tag{5.6}$$

Donc, dans le cas d'un déplacement à la cambrure, ceci devient :

$$\begin{cases} s_i^h = 2(s_c + hV_{s,k}) - s_e(t_0^h) = s_i + h\mathcal{V}^c_{intrados,s} + o(h), \\ z_i^h = 2(z_c + hV_{z,k}) - z_e(t_0^h) = z_i + h\mathcal{V}^c_{intrados,z} + o(h), \end{cases}$$

$$\text{où } \begin{cases} \mathcal{V}^c_{i,s} = 2V_{s,k} - As_e'(t_0), \\ \mathcal{V}^c_{i,z} = 2V_{z,k} - Az_e'(t_0), \end{cases}$$

sont les composantes du champ de déformation $\overrightarrow{\mathcal{V}_i^c}$, à l'intrados, engendré par un déplacement à la cambrure.

5.4 Déplacement d'un point de contrôle du moyeu

On procède de la même façon que pour l'extrados et la cambrure. C'est en fait plus simple car le déplacement d'un point de contrôle du moyeu n'a de répercussions que sur le moyeu lui-même, mais pas sur les aubes.

5.4.1 Champ de déformation induit au moyeu

On se place dans un repère cartésien et les coordonnées des points de contrôle de la cambrure sont notées $(z_j, r_j)_j$. La B-spline obtenue grâce à ces points de contrôle est notée $(z(t), r(t))$. On considère que l'on déplace un point (z_k, r_k) de sa position initiale (z_k^0, r_k^0) dans une direction $u_k = (u_{z,k}, u_{r,k})$, de sorte que le nouveau point de contrôle est $(z_k^h, r_k^h) = (z_k^0, r_k^0) + h.u_k = (z_k^0 + hu_{z,k}, r_k^0 + hu_{r,k})$.
L'application $V_k = (V_{z,k}, V_{r,k}) : \mathbb{R}^2 \to \mathbb{R}^2$ telle que, pour chaque intervalle Γ_j de la B-spline du moyeu :

$$\begin{cases} r^h(t) = r(t) + hV_{r,k}(r(t), z(t)) + o(h), \\ z^h(t) = z(t) + hV_{z,k}(r(t), z(t)) + o(h), \end{cases}$$

est $\mathcal{C}^2$ et s'exprime comme suit :

$$V_k = \begin{cases} p_0(t)u_k & \text{sur } \Gamma_0, \\ p_1(t)u_k & \text{sur } \Gamma_1, \\ p_2(t)u_k & \text{sur } \Gamma_2, \\ p_3(t)u_k & \text{sur } \Gamma_3, \\ 0 & \text{sur les autres } \Gamma_j \text{ du moyeu.} \end{cases}$$

5.5 Ecriture des vecteurs en dimension 3

Dans l'expression de gradient incomplet (4.10) apparaît la projection $\mathcal{V}.\mathcal{N}$ du champ de déformation sur la normale aux surfaces considérées. On cherche donc à calculer les deux termes de ce produit scalaire.
Les repères en 2D choisis pour la modélisation des aubes et pour celle du moyeu ne sont pas les mêmes. Ils ont été choisis en fonction des "bonnes" propriétés géométriques de la pompe (aubes construites par homothétie du profil au carter, moyeu à symétrie de révolution) et ils vont donc nécessiter des calculs différents pour le passage en 3D.

5.5.1 Champ de déformation à l'extrados des aubes

Pour les besoins de la modélisation numérique de la géométrie, on connaît le profil $(s_e(t), z_e(t)) = (\theta_e(t).R, z_e(t))$ de l'extrados des aubes projeté sur une surface de rayon constant R proche de celui du carter. A un rayon quelconque r dans le repère cylindrique $(0, r, \theta, z)$, le profil 2D est de la forme $(s_e(t)r/R, z_e(t)) = (\theta_e(t)r, z_e(t))$. En passant en 3D, dans le repère cartésien $0\mathcal{E} = (0, e_1, e_2, e_3)$, on obtient le profil suivant : $(x_e(t,r), y_e(t,r), z_e(t)) = (r\cos(s_e(t)/R), r\sin(s_e(t)/R), z_e(t))$.
On détermine que le vecteur normal à la B-spline de l'extrados, orienté vers l'intérieur de l'aube (i.e. l'extérieur du domaine d'écoulement), a pour expression :

$$\overrightarrow{n_e} = \frac{1}{\sqrt{\frac{r^2 s_e'^2}{R^2} + z_e'^2}}.(z_e' \sin(s_e(t)/R), -z_e' \cos(s_e(t)/R), s_e'(t)r/R).$$

Par ailleurs, on note, pour l'extrados, $\overrightarrow{\mathcal{V}_e}$ le champ de déformation. On rappelle qu'une déformation se traduit, en 2D, par

$$(s_e(t), z_e(t)) \to (s_e(t) + h\mathcal{V}_{e,s} + o(h), z_e(t) + h\mathcal{V}_{e,z} + o(h)),$$

ce qui devient, en 3D :

$$\begin{cases} x_e(t,r) & \to r\cos((s_e(t) + h\mathcal{V}_{e,s} + o(h))/R) = x_e(t,r) - \frac{r\mathcal{V}_{e,s}}{R}\sin(s_e(t)/R)h + o(h), \\ y_e(t,r) & \to r\sin((s_e(t) + h\mathcal{V}_{e,s} + o(h))/R) = y_e(t,r) + \frac{r\mathcal{V}_{e,s}}{R}\cos(s_e(t)/R)h + o(h), \\ z_e(t) & \to z_e(t) + h\mathcal{V}_{e,z} + o(h), \end{cases}$$

c'est-à-dire $\overrightarrow{\mathcal{V}_e} = (-\mathcal{V}_{e,s} r\sin(s_e(t)/R)/R, \mathcal{V}_{e,s} r\cos(s_e(t)/R)/R, \mathcal{V}_{e,z})$. Et enfin il ne reste qu'à calculer le produit scalaire :

$$\overrightarrow{\mathcal{V}_e}.\overrightarrow{n_e} = \frac{1}{\sqrt{s_e'^2 + \frac{R^2 z_e'^2}{r^2}}}.(-\mathcal{V}_{e,s} z_e' + \mathcal{V}_{e,z} s_e').$$

5.5.2 Champ de déformation à l'intrados des aubes

Soit (s_i, z_i) un point de l'intrados. Soit $t_1 \in]0,1[$ tel que $(s_c(t_1), z_c(t_1))$ soit la projection de (s_i, z_i) sur la cambrure, et $t_0 \in]0,1[$ tel que $(s_c(t_1), z_c(t_1))$ soit la projection du point de l'extrados $(s_e(t_0), z_e(t_0))$ sur la cambrure.
On rappelle qu'en 5.1, on a énoncé le théorème des fonctions implicites 5.1.1 qui permet de définir le point (s_i, z_i) comme l'image de t_0 de la façon suivante :

$$(s_i, z_i)(t_0) = (2s_c(\varphi(t_0)) - s_e(t_0), 2z_c(\varphi(t_0)) - z_e(t_0)).$$

On définit, respectivement, $\overrightarrow{\tau_e}(t_0)$, $\overrightarrow{\tau_c}(t_0)$, $\overrightarrow{n_e}(t_0)$ et $\overrightarrow{n_c}(t_0)$ les tangentes à l'extrados en $(s_e(t_0), z_e(t_0))$ et à la cambrure en $(s_c(\varphi(t_0)), z_c(\varphi(t_0)))$, et les normales à l'extrados et à la cambrure orientées vers l'intérieur de l'aube, en ces mêmes points. Pour plus de simplicité on notera ces vecteurs $\overrightarrow{\tau_e}$, $\overrightarrow{\tau_c}$, $\overrightarrow{n_e}$ et $\overrightarrow{n_c}$. Alors, dans le plan défini par $\overrightarrow{\tau_c}$ et $\overrightarrow{n_c}$, les propriétés de symétrie de l'intrados et de l'extrados par rapport à la cambrure permettent d'écrire que les coordonnées du vecteur $\overrightarrow{n_i}$ normal à l'intrados au point (s_i, z_i) dans le plan $((s_c(t_1), z_c(t_1)), \overrightarrow{\tau_c}, \overrightarrow{n_c})$ sont :

$$\overrightarrow{n_i}.\overrightarrow{\tau_c} = \overrightarrow{n_e}.\overrightarrow{\tau_c},$$

$$\overrightarrow{n_i}.\overrightarrow{n_c} = -\overrightarrow{n_e}.\overrightarrow{n_c}.$$

On en déduit les coordonnées dans le repère cartésien $0\mathcal{E}$ lié à la pompe :

$$\overrightarrow{n_i} = \sum_{j=1}^{3} \left((\overrightarrow{n_e}.\overrightarrow{\tau_c})(\overrightarrow{\tau_c}.e_j) - (\overrightarrow{n_e}.\overrightarrow{n_c})(\overrightarrow{n_c}.e_j) \right) e_j.$$

D'autre part on note $\overrightarrow{\mathcal{V}_i}$ le champ de déformation de l'intrados. La déformation en 2D se traduit par :

$$(s_i, z_i) \to (s_i + h\mathcal{V}_{i,s} + o(h), z_i + h\mathcal{V}_{i,z} + o(h)),$$

ce qui devient, en 3D :

$$\begin{cases} x_i(r) & \to r\cos((s_i + h\mathcal{V}_{i,s} + o(h))/R) = x_i(r) - \frac{r\mathcal{V}_{i,s}}{R}\sin(s_i/R)h + o(h), \\ y_i(r) & \to r\sin((s_i + h\mathcal{V}_{i,s} + o(h))/R) = y_i(r) + \frac{r\mathcal{V}_{i,s}}{R}\sin(s_i/R)h + o(h), \\ z_i & \to z_i + h\mathcal{V}_{i,z} + o(h), \end{cases}$$

c'est-à-dire $\overrightarrow{\mathcal{V}_i} = (-\mathcal{V}_{i,s} r\sin(s_i/R)/R, \mathcal{V}_{i,s} r\cos(s_i/R)/R, \mathcal{V}_{i,z})$. Et enfin il reste à calculer le produit scalaire $\overrightarrow{\mathcal{V}_i}.\overrightarrow{n_i}$.

5.5.3 Champ de déformation normal au moyeu

Pour ce qui est du moyeu, la paramétrisation en 2D $(z(t), r(t))$ est traduite en 3D, et dans le repère cartésien, par $x(t) = r(t)\cos\theta$, $y(t) = r(t)\sin\theta$ et $z(t)$ inchangé. Le champ de déformation calculé est de la forme $\overrightarrow{\mathcal{V}_m} = (\mathcal{V}_{m,r}\cos\theta, \mathcal{V}_{m,r}\sin\theta, \mathcal{V}_{m,z})$.
Le vecteur normal au moyeu, orienté vers l'extérieur du domaine d'écoulement, a pour expression :

$$\overrightarrow{n_m} = \frac{1}{\sqrt{r'^2 + z'^2}}.(-z'\cos\theta, -z'\sin\theta, r')$$

et on en déduit que :

$$\overrightarrow{\mathcal{V}_m}.\vec{n} = \frac{1}{\sqrt{r'^2 + z'^2}}.(-\mathcal{V}_{m,r}z' + \mathcal{V}_{m,z}r').$$

5.6 Cas particulier du champ de déformation sur les surfaces d'entrée et de sortie de l'étage de pompe

Les faces d'entrée et de sortie sont des couronnes, de rayon intérieur le rayon minimal du moyeu, et de rayon extérieur le rayon du carter. La déformation de ces couronnes peut résulter d'une modification du rayon du moyeu (dans ce cas, les couronnes changent d'aire), ou d'une modification de la longueur d'un étage de pompe (et dans ce cas, les couronnes sont translatées selon l'axe de la pompe). On rappelle que l'on ne peut pas envisager de variation du rayon du carter car celui-ci est limité par le rayon du puits. On considère la couronne d'entrée. Si l'on reprend les notations du paragraphe 5.4.1, en 2D les coordonnées du point d'intersection entre la courbe représentant le moyeu et une droite représentant l'entrée sont $(z(0), r(0))$. Si on note $(z_j, r_j)_j$ les points de contrôle du moyeu, seul le déplacement des trois premiers peut entraîner un déplacement de $(z(0), r(0))$.
On considère un point de contrôle (z_k, r_k) qui subit un déplacement d'amplitude h dans la direction $u_k = (u_{z,k}, u_{r,k})$. Soit l'application $V_k = (V_{z,k}, V_{r,k}) : \mathbb{R}^2 \to \mathbb{R}^2$ telle que $(z^h(0), r^h(0)) = (z(0), r(0)) + hV_k(z(0), r(0)) + o(h)$. Cette application s'exprime comme suit au point d'intersection entre la courbe représentant le moyeu et une droite représentant l'entrée :

$$V_k(z(0), r(0)) = \begin{cases} p_3(0)u_1 = u_1/6 & \text{pour } k = 1, \\ p_2(0)u_2 = 4u_2/6 & \text{pour } k = 2, \\ p_1(0)u_3 = u_3/6 & \text{pour } k = 3, \\ 0 & \text{pour } k > 3. \end{cases}$$

On revient à des considérations en 3D et on note $\overrightarrow{\mathcal{V}_{ce}}$ le champ de déformation sur la couronne d'entrée tel que :

$$(r\cos\theta, r\sin\theta, z(0)) \to (r\cos\theta + h\mathcal{V}_{ce,r}\cos\theta + o(h), r\sin\theta + h\mathcal{V}_{ce,r}\sin\theta + o(h), z(0) + h\mathcal{V}_{ce,z} + o(h)).$$

La déformation axiale subie par la surface d'entrée est, en tous points de celle-ci : $\mathcal{V}_{ce,z} = V_{z,k}(z(0), r(0))$. Déterminer la déformation radiale $V_{ce,r}$ est inutile. En effet, la couronne d'entrée est plane et sa normale $\overrightarrow{n_{ce}}$ est dans la direction axiale. Le produit scalaire du vecteur champ de déformation par le vecteur normal se résume donc à $\overrightarrow{\mathcal{V}_{ce}}.\overrightarrow{n_{ce}} = -\mathcal{V}_{ce,z}$.
On considère maintenant la surface de sortie. Le raisonnement est le même. On note n_m le nombre total de points de contrôle du moyeu. Seul le déplacement des trois derniers points de contrôle peut entraîner un déplacement du point $(z(1), r(1))$, intersection entre la courbe représentant le moyeu et une droite représentant la sortie. On considère un point de contrôle (z_k, r_k) qui subit un déplacement d'amplitude h dans la direction $u_k = (u_{z,k}, u_{r,k})$. Soit l'application $V_k = (V_{z,k}, V_{r,k}) : \mathbb{R}^2 \to \mathbb{R}^2$ telle que $(z^h(1), r^h(1)) = (z(1), r(1)) + hV_k(z(1), r(1)) + o(h)$. Cette application s'exprime comme suit au point $(z(1), r(1))$:

$$V_k(z(1), r(1)) = \begin{cases} 0 & \text{pour } k < n_m - 2, \\ p_2(1)u_{n_m-2} = u_{n_m-2}/6 & \text{pour } k = n_m - 2, \\ p_1(1)u_{n_m-1} = 4u_{n_m-1}/6 & \text{pour } k = n_m - 1, \\ p_0(1)u_{n_m} = u_{n_m}/6 & \text{pour } k = n_m. \end{cases}$$

On note $\overrightarrow{\mathcal{V}_{cs}}$ la déformation 3D subie par la couronne de sortie :

$$(r\cos\theta, r\sin\theta, z(1)) \to (r\cos\theta + h\mathcal{V}_{ce,r}\cos\theta + o(h), r\sin\theta + h\mathcal{V}_{ce,r}\sin\theta + o(h), z(1) + h\mathcal{V}_{ce,z} + o(h)).$$

La déformation axiale subie par la surface de sortie est, en tous points de celle-ci : $\mathcal{V}_{cs,z} = V_{z,k}(z(1), r(1))$. De même que pour la couronne d'entrée, à la sortie cette dernière expression suffit à expliciter le produit scalaire $\overrightarrow{\mathcal{V}_{cs}}.\overrightarrow{n_{cs}} = \mathcal{V}_{cs,z}$.

Remarque 5.6.1. *La seule déformation des aubes qui puisse engendrer une déformation des surfaces d'entrée et/ou de sortie est axiale. Mais une contrainte relie la longueur axiale des aubes à celle du moyeu, et la déformation du moyeu engendrée est donc facile à calculer. Il ne reste plus qu'à appliquer ce qui précède pour trouver la déformation des surfaces d'entrée et/ou de sortie.*

Chapitre 6

Dérivée de l'extension de la normale

On souhaite calculer $\mathcal{N}'$, la dérivée par rapport au domaine de l'extension $\mathcal{N}$ de la normale. Ce chapitre est le développement, dans le cas de la géométrie de la pompe, des calculs généraux menés au paragraphe 4.4. Le calcul du gradient incomplet (4.10) qui a été effectué montre que l'on n'a besoin de cette dérivée que sur la surface D définie comme étant l'ensemble des bords déformés lors de l'optimisation, à l'exception de l'entrée E et de la sortie S. Ainsi, on doit mener les calculs sur le moyeu et sur les aubes, pour lesquelles on doit distinguer le cas de l'extrados et de l'intrados.
On garde ici les notations du chapitre 4.

6.1 Calculs relatifs au moyeu

Calcul de ${}^tDV.\mathcal{N}$:

Le moyeu est à symétrie de révolution, ce qui permet d'introduire une simplification appréciable. En effet, on considère la courbe 2D, à un seul paramètre, du moyeu : $M(t) = (z(t), r(t))$. Soit $X = (x_1, x_2)$ un point du voisinage du moyeu et soit $M(t_X) = (z(t_X), r(t_X))$ sa projection sur le moyeu. Alors $(X - M(t_X)).\dfrac{\partial M(t_X)}{\partial t} = 0$. Ainsi t_X est solution de l'équation polynomiale du cinquième degré :

$$z'(t)(x_1 - z(t)) + r'(t)(x_2 - r(t)) = 0. \tag{6.1}$$

Pour plus de lisibilité on note dans ce qui suit $M(t_X) = (z(t_X), r(t_X)) = (z, r)$. On dérive l'équation (6.1) par rapport à x_1 :

$$\frac{\partial t_X}{\partial x_1} z''(x_1 - z) + z' - z'^2 \frac{\partial t_X}{\partial x_1} + \frac{\partial t_X}{\partial x_1} r''(x_2 - r) - r'^2 \frac{\partial t_X}{\partial x_1} = 0.$$

On en déduit :

$$\frac{\partial t_X}{\partial x_1} = z' \left(z''(z - x_1) + z'^2 + r''(r - x_2) + r'^2 \right)^{-1}$$

et de la même façon on obtient, en dérivant par rapport à x_2 :

$$\frac{\partial t_X}{\partial x_2} = r' \left(z''(z - x_1) + z'^2 + r''(r - x_2) + r'^2 \right)^{-1}.$$

On note Γ_m la surface du moyeu, on a alors en un point $(x_1, x_2) = (z, r)$ de Γ_m :

$$\begin{cases} \dfrac{\partial t_X}{\partial x_1}_{|\Gamma_m} = z' \left(z'^2 + r'^2 \right)^{-1}, \\ \dfrac{\partial t_X}{\partial x_2}_{|\Gamma_m} = r' \left(z'^2 + r'^2 \right)^{-1}. \end{cases}$$

Le champ de déformation est, pour le moyeu représenté en 2D, un vecteur $\mathcal{V}_m = (\mathcal{V}_{m,z}, \mathcal{V}_{m,r})$ que l'on note dans ce paragraphe, $\mathcal{V}_m = (\mathcal{V}_1, \mathcal{V}_2)$. Alors, pour l'extension V_m de $\mathcal{V}_m$,

$$ {}^tDV_m(t_X) = \begin{pmatrix} \dfrac{\partial V_1}{\partial x_1} & \dfrac{\partial V_2}{\partial x_1} \\ \\ \dfrac{\partial V_1}{\partial x_2} & \dfrac{\partial V_2}{\partial x_2} \end{pmatrix} = \begin{pmatrix} \dfrac{\partial V_1}{\partial t}\dfrac{\partial t_X}{\partial x_1} & \dfrac{\partial V_2}{\partial t}\dfrac{\partial t_X}{\partial x_1} \\ \\ \dfrac{\partial V_1}{\partial t}\dfrac{\partial t_X}{\partial x_2} & \dfrac{\partial V_2}{\partial t}\dfrac{\partial t_X}{\partial x_2} \end{pmatrix} = \begin{pmatrix} \dfrac{\partial t_X}{\partial x_1} \\ \\ \dfrac{\partial t_X}{\partial x_2} \end{pmatrix} . \begin{pmatrix} \dfrac{\partial V_1}{\partial t} & \dfrac{\partial V_2}{\partial t} \end{pmatrix}, $$

et ${}^tD\mathcal{V}_m(t_X) = \begin{pmatrix} \dfrac{\partial t_X}{\partial x_1}_{|\Gamma_m} \\ \dfrac{\partial t_X}{\partial x_2}_{|\Gamma_m} \end{pmatrix} . \begin{pmatrix} \dfrac{\partial \mathcal{V}_1}{\partial t} & \dfrac{\partial \mathcal{V}_2}{\partial t} \end{pmatrix}$ sur Γ_m.

Dans le cas du moyeu, les calculs sont simples et on peut donc faire les calculs complets de produits matriciels. Ainsi, on a sur le bord du moyeu :

$$ ({}^tDV_m.\mathcal{N})_{|\Gamma_m} = {}^tD\mathcal{V}_m.n = \frac{1}{(r'^2+z'^2)^{1/2}} . \begin{pmatrix} \dfrac{\partial \mathcal{V}_1}{\partial t}\dfrac{z'}{(r'^2+z'^2)} & \dfrac{\partial \mathcal{V}_2}{\partial t}\dfrac{z'}{(r'^2+z'^2)} \\ \\ \dfrac{\partial \mathcal{V}_1}{\partial t}\dfrac{r'}{(r'^2+z'^2)} & \dfrac{\partial \mathcal{V}_2}{\partial t}\dfrac{r'}{(r'^2+z'^2)} \end{pmatrix} . \begin{pmatrix} r' \\ \\ -z' \end{pmatrix} $$

et donc :

$$ {}^tD\mathcal{V}_m.n = \frac{1}{(r'^2+z'^2)^{3/2}} \begin{pmatrix} \dfrac{\partial \mathcal{V}_1}{\partial t} r'z' - \dfrac{\partial \mathcal{V}_2}{\partial t} z'^2 \\ \\ \dfrac{\partial \mathcal{V}_1}{\partial t} r'^2 - \dfrac{\partial \mathcal{V}_2}{\partial t} r'z' \end{pmatrix} . \tag{6.2} $$

Remarque 6.1.1. *Il est facile de vérifier que* $({}^tDV_m.\mathcal{N}).n = 0$ *sur* Γ_m*, conformément à la proposition 4.4.2.*

Calcul de $D\mathcal{N}.V$ avec la seconde forme fondamentale :

On revient en 3D, le moyeu est alors une surface de révolution à deux paramètres t et θ :

$$ M(t,\theta) = (r(t)\cos\theta, r(t)\sin\theta, z(t)). $$

Alors $\dfrac{\partial M}{\partial t} = (r'(t)\cos\theta, r'(t)\sin\theta, z'(t))$ et $\dfrac{\partial M}{\partial \theta} = (-r(t)\sin\theta, r(t)\cos\theta, 0)$ et on calcule facilement :

$$ K = r(z'^2+r'^2)^{1/2},\ L = \frac{1}{(r'^2+z'^2)^{1/2}}(z''r' - z'r''),\ M = 0,\ N = \frac{1}{(r'^2+z'^2)^{1/2}} z'r, $$

$$ E = r'^2 + z'^2,\ F = 0,\ G = r^2. $$

Soit $W = \lambda_w \dfrac{\partial M}{\partial t} + \mu_w \dfrac{\partial M}{\partial \theta} + \nu_w \mathcal{N}$. On calcule alors :

$$ \lambda_w = \frac{1}{(r'^2+z'^2)} W. \begin{pmatrix} r'\cos\theta \\ r'\sin\theta \\ z' \end{pmatrix},\ \mu_w = \frac{1}{r} W. \begin{pmatrix} -\sin\theta \\ \cos\theta \\ 0 \end{pmatrix}. $$

En particulier, puisque $\mathcal{V}_m = (\mathcal{V}_{m,r}\cos\theta, \mathcal{V}_{m,r}\sin\theta, \mathcal{V}_{m,z})$, on obtient $\lambda_v = \frac{r'\mathcal{V}_{m,r}+z'\mathcal{V}_{m,z}}{(r'^2+z'^2)}$ et $\mu_v = 0$. Ainsi, on a finalement :

$$\begin{aligned}\forall W,\ W.D\mathcal{N}.\mathcal{V}_m &= L\lambda_v\lambda_w + M(\lambda_v\mu_w + \lambda_w\mu_v) + N\mu_v\mu_w \\ &= \frac{(z''r' - z'r'')}{(r'^2+z'^2)^{5/2}}(r'\mathcal{V}_{m,r} + z'\mathcal{V}_{m,z})W.\begin{pmatrix} r'\cos\theta \\ r'\sin\theta \\ z' \end{pmatrix}.\end{aligned} \tag{6.3}$$

Calcul de la courbure moyenne H :

$$H = \frac{LG+EN-2FM}{K^2} = \frac{z''r'-z'r''}{(r'^2+z'^2)^{3/2}} + \frac{z'}{r(r'^2+z'^2)^{1/2}}. \tag{6.4}$$

On observe bien que la courbure moyenne est égale à la somme des courbures normales relatives à deux directions orthogonales, dont l'une (correspondant au premier terme) est la direction axiale z.

6.2 Calculs généraux relatifs à une aube

L'extrados et l'intrados sont des nappes paramétrées en t et r de la façon suivante : $M(t,r) = (r\cos(s(t)/R), r\sin(s(t)/R), z(t))$. Pour simplifier les notations, on omettra la plupart du temps la variable t, tout en gardant à l'esprit que s et z sont des fonctions de t. On note Γ_a la surface d'une aube.

Calcul de ${}^tDV$:

Soit $X = (x_1, x_2, x_3)$ un point du voisinage d'une aube et soit $M(t_X, r_X)$ sa projection sur celle-ci. Alors le couple (t_X, r_X) minimise la norme $||MX||^2 = (x_1 - r\cos(s/R))^2 + (x_2 - r\sin(s/R))^2 + (x_3 - z)^2$. Donc (t_X, r_X) est solution de :

$$\begin{cases} \dfrac{\partial||MX||^2}{\partial t} = 0, \\ \dfrac{\partial||MX||^2}{\partial r} = 0, \end{cases} \quad \text{soit} \begin{cases} \dfrac{rs'}{R}(x_2\cos(s/R) - x_1\sin(s/R)) + z'(x_3 - z) = 0, \\ x_1\cos(s/R) + x_2\sin(s/R) - r = 0. \end{cases}$$

On dérive chacune de ces deux équations par rapport à x_1, x_2 et x_3 et on les couple en systèmes permettant de déterminer les $\dfrac{\partial t_X}{\partial x_i}_{|\Gamma_a}$ et les $\dfrac{\partial r_X}{\partial x_i}_{|\Gamma_a}$. Par exemple :

$$\begin{cases} \dfrac{\partial}{\partial x_1}\dfrac{\partial||MX||^2}{\partial t} = 0, \\ \dfrac{\partial}{\partial x_1}\dfrac{\partial||MX||^2}{\partial r} = 0, \end{cases}$$

équivaut à :

$$\begin{cases} \dfrac{rs'}{R}\sin(s/R) & + \dfrac{\partial r}{\partial x_1}\left(\dfrac{s'}{R}(x_1\sin(s/R) - x_2\cos(s/R))\right) \\ & + \dfrac{\partial t}{\partial x_1}\left(\dfrac{rs''}{R}(x_1\sin(s/R) - x_2\cos(s/R)) + \dfrac{rs'^2}{R^2}(x_1\cos(s/R) + x_2\sin(s/R)) + z''(z - x_3) + z'^2\right) = 0, \\ \dfrac{\partial r}{\partial x_1} + \dfrac{\partial t}{\partial x_1}\dfrac{s'}{R} & (x_1\sin(s/R) - x_2\cos(s/R)) - \cos(s/R) = 0. \end{cases}$$

On cherche à déterminer $\dfrac{\partial t_X}{\partial x_1}_{|\Gamma_a}$ et $\dfrac{\partial r_X}{\partial x_1}_{|\Gamma_a}$, donc il suffit de résoudre le système pour $X \in \Gamma_a$, soit $(x_1, x_2, x_3) = (r\cos(s/R), r\sin(s/R), z)$. Cela introduit des simplifications dans le système car alors :

$$(x_1 \sin(s/R) - x_2 \cos(s/R)) = 0,\ (x_1 \cos(s/R) + x_2 \sin(s/R)) = r \text{ et } x_3 = z.$$

On en déduit :

$$\begin{cases} \dfrac{\partial t_X}{\partial x_1}_{|\Gamma_a} = -\dfrac{rs'}{R}\sin(s/R)\dfrac{1}{\frac{r^2 s'^2}{R^2} + z'^2}, \\ \dfrac{\partial r_X}{\partial x_1}_{|\Gamma_a} = \cos(s/R). \end{cases}$$

En résolvant de la même façon les deux systèmes de deux équations restants, on détermine :

$$\begin{cases} \dfrac{\partial t_X}{\partial x_2}_{|\Gamma_a} = \dfrac{rs'}{R}\cos(s/R)\dfrac{1}{\frac{r^2 s'^2}{R^2} + z'^2}, \\ \dfrac{\partial r_X}{\partial x_2}_{|\Gamma_a} = \sin(s/R), \end{cases} \quad \text{et} \quad \begin{cases} \dfrac{\partial t_X}{\partial x_3}_{|\Gamma_a} = z'\dfrac{1}{\frac{r^2 s'^2}{R^2} + z'^2}, \\ \dfrac{\partial r_X}{\partial x_3}_{|\Gamma_a} = 0. \end{cases}$$

Dans le chapitre 5 on a montré que le champ de déformation des aubes a une expression de la forme $\mathcal{V}_a = (-\mathcal{V}_{a,s}\frac{r}{R}\sin(s/R), \mathcal{V}_{a,s}\frac{r}{R}\cos(s/R), \mathcal{V}_{a,z})$ que l'on écrit aussi $\mathcal{V}_a = (\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3)$. Par ailleurs $\mathcal{V}_{a,s}$ et $\mathcal{V}_{a,z}$ ne dépendent que de t et pas de r. Alors

$$\begin{aligned} \frac{\partial \mathcal{V}_a}{\partial t} &= (\frac{\partial \mathcal{V}_1}{\partial t}, \frac{\partial \mathcal{V}_2}{\partial t}, \frac{\partial \mathcal{V}_3}{\partial t}) \\ &= \left(-\frac{r}{R}(\mathcal{V}'_{a,s}\sin(s/R) + \mathcal{V}_{a,s}\frac{s'}{R}\cos(s/R)), \frac{r}{R}(\mathcal{V}'_{a,s}\cos(s/R) - \mathcal{V}_{a,s}\frac{s'}{R}\sin(s/R)), \mathcal{V}'_{a,z}\right) \end{aligned}$$

et

$$\begin{aligned} \frac{\partial \mathcal{V}_a}{\partial r} &= (\frac{\partial \mathcal{V}_1}{\partial r}, \frac{\partial \mathcal{V}_2}{\partial r}, \frac{\partial \mathcal{V}_3}{\partial r}) \\ &= \left(-\mathcal{V}_{a,s}\frac{1}{R}\sin(s/R), \mathcal{V}_{a,s}\frac{1}{R}\cos(s/R), 0\right). \end{aligned}$$

On rappelle que, pour l'extension V_a de $\mathcal{V}_a$:

$${}^tDV_a = \begin{pmatrix} \dfrac{\partial V_1}{\partial x_1} & \dfrac{\partial V_2}{\partial x_1} & \dfrac{\partial V_3}{\partial x_1} \\ \dfrac{\partial V_1}{\partial x_2} & \dfrac{\partial V_2}{\partial x_2} & \dfrac{\partial V_3}{\partial x_2} \\ \dfrac{\partial V_1}{\partial x_3} & \dfrac{\partial V_2}{\partial x_3} & \dfrac{\partial V_3}{\partial x_3} \end{pmatrix} = \begin{pmatrix} \dfrac{\partial t_X}{\partial x_1} & \dfrac{\partial r_X}{\partial x_1} \\ \dfrac{\partial t_X}{\partial x_2} & \dfrac{\partial r_X}{\partial x_2} \\ \dfrac{\partial t_X}{\partial x_3} & \dfrac{\partial r_X}{\partial x_3} \end{pmatrix} \cdot \begin{pmatrix} \dfrac{\partial V_1}{\partial t} & \dfrac{\partial V_2}{\partial t} & \dfrac{\partial V_3}{\partial t} \\ \dfrac{\partial V_1}{\partial r} & \dfrac{\partial V_2}{\partial r} & \dfrac{\partial V_3}{\partial r} \end{pmatrix}$$

et donc, sur Γ_a :

$${}^tD\mathcal{V}_a = \begin{pmatrix} \dfrac{\partial t_X}{\partial x_1}_{|\Gamma_a} & \dfrac{\partial r_X}{\partial x_1}_{|\Gamma_a} \\ \dfrac{\partial t_X}{\partial x_2}_{|\Gamma_a} & \dfrac{\partial r_X}{\partial x_2}_{|\Gamma_a} \\ \dfrac{\partial t_X}{\partial x_3}_{|\Gamma_a} & \dfrac{\partial r_X}{\partial x_3}_{|\Gamma_a} \end{pmatrix} \cdot \begin{pmatrix} \dfrac{\partial \mathcal{V}_1}{\partial t} & \dfrac{\partial \mathcal{V}_2}{\partial t} & \dfrac{\partial \mathcal{V}_3}{\partial t} \\ \dfrac{\partial \mathcal{V}_1}{\partial r} & \dfrac{\partial \mathcal{V}_2}{\partial r} & \dfrac{\partial \mathcal{V}_3}{\partial r} \end{pmatrix}. \tag{6.5}$$

Calcul de $D\mathcal{N}.V$ avec la seconde forme fondamentale :

On calcule d'abord :

$$\frac{\partial M}{\partial t} = \left(-\frac{rs'(t)}{R}\sin(s(t)/R), \frac{rs'(t)}{R}\cos(s(t)/R), z'(t)\right)$$

et

$$\frac{\partial M}{\partial r} = (\cos(s(t)/R), \sin(s(t)/R), 0)$$

et on en déduit facilement :

$$K = \left(z'^2 + \left(\frac{rs'}{R}\right)^2\right)^{1/2},\ L = \frac{1}{\left(z'^2 + \left(\frac{rs'}{R}\right)^2\right)^{1/2}}\frac{r}{R}(s''z' - s'z''),\ M = \frac{1}{\left(z'^2 + \left(\frac{rs'}{R}\right)^2\right)^{1/2}}\frac{z's'}{R},\ N = 0,$$

$$E = \left(\frac{rs'}{R}\right)^2 + z'^2,\ F = 0,\ G = 1.$$

Soit $W = \lambda_w \frac{\partial M}{\partial t} + \mu_w \frac{\partial M}{\partial r} + \nu_w \mathcal{N}$. On calcule alors :

$$\lambda_w = \frac{1}{z'^2 + \left(\frac{rs'}{R}\right)^2} W.\begin{pmatrix} -\frac{rs'}{R}\sin(s/R) \\ \frac{rs'}{R}\cos(s/R) \\ z' \end{pmatrix},\ \mu_w = W.\begin{pmatrix} \cos(s/R) \\ \sin(s/R) \\ 0 \end{pmatrix}.$$

En particulier, puisque $\mathcal{V}_a = (-\mathcal{V}_{a,s}\frac{r}{R}\sin(s/R), \mathcal{V}_{a,s}\frac{r}{R}\cos(s/R), \mathcal{V}_{a,z})$, on obtient :

$$\lambda_v = \frac{1}{z'^2 + \left(\frac{rs'}{R}\right)^2}\left(\mathcal{V}_{a,s}\frac{r^2 s'}{R^2} + \mathcal{V}_{a,z}z'\right) \text{ et } \mu_v = 0.$$

Ainsi on a finalement :

$$\begin{aligned} \forall W,\ W.D\mathcal{N}.\mathcal{V}_a &= L\lambda_v\lambda_w + M(\lambda_v\mu_w + \lambda_w\mu_v) + N\mu_v\mu_w \\ &= \frac{1}{\left(z'^2 + \left(\frac{rs'}{R}\right)^2\right)^{5/2}}\frac{r}{R}(s''z' - s'z'')\left(\mathcal{V}_{a,s}\frac{r^2 s'}{R^2} + \mathcal{V}_{a,z}z'\right) W.\begin{pmatrix} -\frac{rs'}{R}\sin(s/R) \\ \frac{rs'}{R}\cos(s/R) \\ z' \end{pmatrix} \\ &\quad + \frac{1}{\left(z'^2 + \left(\frac{rs'}{R}\right)^2\right)^{3/2}}\frac{z's'}{R}\left(\mathcal{V}_{a,s}\frac{r^2 s'}{R^2} + \mathcal{V}_{a,z}z'\right) W.\begin{pmatrix} \cos(s/R) \\ \sin(s/R) \\ 0 \end{pmatrix}. \end{aligned} \tag{6.6}$$

Calcul de la courbure moyenne H :

$$H = \frac{LG + EN - 2FM}{K^2} = \frac{r}{R}\frac{s''z' - s'z''}{\left(z'^2 + \left(\frac{rs'}{R}\right)^2\right)^{3/2}}. \tag{6.7}$$

6.3 Distinction entre l'intrados et l'extrados

Au paragraphe 6.2 on a mené les calculs généraux relatifs à une aube. La surface d'une aube est soit l'intrados, soit l'extrados. Les expressions (6.5) à (6.7) sont valables dans les deux cas mais nécessitent de déterminer les dérivées premières s', z' et secondes s'', z'' ainsi que les dérivées du champ de déformation $\mathcal{V}_a$. Etant donné que la paramétrisation des surfaces et la valeur prise par le champ de déformation dépendent beaucoup de la nature de la surface étudiée, on distingue dans ce paragraphe les calculs des termes de (6.5) à (6.7) dans le cas de l'extrados et de l'intrados. On remplace alors la notation $\mathcal{V}_a$ par $\mathcal{V}_e$ ou $\mathcal{V}_i$ selon qu'il s'agit du champ de déformation à l'extrados ou à l'intrados.

6.3.1 L'extrados

En 2D l'extrados est une B-spline. Par conséquent s_e et z_e sont des fonctions polynomiales par morceaux parfaitement explicitées du paramètre t et il n'y a aucune difficulté à en calculer les dérivées premières et secondes. Il en va de même pour le champ de déformation $\mathcal{V}_e$.

6.3.2 L'intrados

Les choses sont différentes pour l'intrados car il ne s'agit pas d'une B-spline mais de la transformation géométrique de deux B-splines. Soient (s_i, z_i) un point de l'intrados, $t_1 \in]0,1[$ tel que $(s_c(t_1), z_c(t_1))$ soit la projection de (s_i, z_i) sur la cambrure et $t_0 \in]0,1[$ tel que $(s_c(t_1), z_c(t_1))$ soit la projection du point de l'extrados $(s_e(t_0), z_e(t_0))$ sur la cambrure.

Rappels du chapitre 5

Le théorème des fonctions implicites 5.1.1 permet de définir le point (s_i, z_i) comme l'image de t_0 de la façon suivante :

$$\begin{aligned}(s_i, z_i)(t_0) &= (2s_c(t_1) - s_e(t_0), 2z_c(t_1) - z_e(t_0)) \\ &= (2s_c(\varphi(t_0)) - s_e(t_0), 2z_c(\varphi(t_0)) - z_e(t_0)).\end{aligned}$$

Par ailleurs on a introduit et calculé les grandeurs suivantes :

$$F(t_0, t_1) = s_c'(t_1)(s_c(t_1) - s_e(t_0)) + z_c'(t_1)(z_c(t_1) - z_e(t_0)) = 0,$$

$$\frac{\partial F}{\partial t_0} = -(s_c'(t_1)s_e'(t_0) + z_c'(t_1)z_e'(t_0)),$$

$$\frac{\partial F}{\partial t_1} = s_c''(t_1)(s_c(t_1) - s_e(t_0)) + s_c'^2(t_1) + z_c''(t_1)(z_c(t_1) - z_e(t_0)) + z_c'^2(t_1),$$

$$\varphi'(t_0) = \frac{\partial t_1}{\partial t_0} = -\frac{\partial F/\partial t_0}{\partial F/\partial t_1}.$$

Dérivées premières et secondes de $(s_i(t_0), z_i(t_0))$

En dérivant une première fois, il vient :

$$\begin{cases} s_i'(t_0) &= 2\dfrac{\partial t_1}{\partial t_0} s_c'(t_1) - s_e'(t_0), \\ z_i'(t_0) &= 2\dfrac{\partial t_1}{\partial t_0} z_c'(t_1) - z_e'(t_0). \end{cases}$$

En dérivant une seconde fois, on obtient :

$$\begin{cases} s_i''(t_0) & = 2\left(\dfrac{\partial^2 t_1}{\partial t_0^2} s_c'(t_1) + \left(\dfrac{\partial t_1}{\partial t_0}\right)^2 s_c''(t_1)\right) - s_e''(t_0), \\ \\ z_i''(t_0) & = 2\left(\dfrac{\partial^2 t_1}{\partial t_0^2} z_c'(t_1) + \left(\dfrac{\partial t_1}{\partial t_0}\right)^2 z_c''(t_1)\right) - z_e''(t_0). \end{cases}$$

Reste donc à calculer le terme $\dfrac{\partial^2 t_1}{\partial t_0^2} = \varphi''(t_0)$. On définit la fonction G, $\mathcal{C}^1$ sur $[0,1]$, par :

$$G(t_0) = F(t_0, \varphi(t_0)).$$

G est identiquement nulle sur $[0,1]$ et on calcule :

$$\frac{\partial G}{\partial t_0} = \frac{\partial F}{\partial t_0} + \varphi'(t_0)\frac{\partial F}{\partial t_1} = 0,$$

$$\frac{\partial^2 G}{\partial t_0^2} = \frac{\partial^2 F}{\partial t_0^2} + \varphi'(t_0)\frac{\partial^2 F}{\partial t_1 \partial t_0} + \varphi'(t_0)\frac{\partial^2 F}{\partial t_0 \partial t_1} + \varphi'^2(t_0)\frac{\partial^2 F}{\partial t_1^2} + \varphi''(t_0)\frac{\partial F}{\partial t_1} = 0.$$

On en déduit que :

$$\varphi''(t_0) = \frac{-1}{\partial F/\partial t_1}\left(\frac{\partial^2 F}{\partial t_0^2} + \varphi'(t_0)\left(\frac{\partial^2 F}{\partial t_1 \partial t_0} + \frac{\partial^2 F}{\partial t_0 \partial t_1} + \varphi'(t_0)\frac{\partial^2 F}{\partial t_1^2}\right)\right),$$

où

$$\frac{\partial^2 F}{\partial t_0^2} = -(s_c'(t_1)s_e''(t_0) + z_c'(t_1)z_e''(t_0)),$$

$$\frac{\partial^2 F}{\partial t_1 \partial t_0} = \frac{\partial^2 F}{\partial t_0 \partial t_1} = -s_c''(t_1)s_e'(t_0) + z_c''(t_1)z_e'(t_0)),$$

$$\frac{\partial^2 F}{\partial t_1^2} = s_c^{(3)}(t_1)(s_c(t_1) - s_e(t_0)) + 3s_c''(t_1)s_c'(t_1) + z_c^{(3)}(t_1)(z_c(t_1) - z_e(t_0)) + 3z_c''(t_1)z_c'(t_1).$$

Dérivées du champ de déformation $\mathcal{V}_i$

Il y a ici encore une distinction à faire selon que la déformation de l'intrados est engendrée par une déformation de la cambrure ou de l'extrados.

Champ de déformation $\mathcal{V}_i^e$ engendré par une déformation de l'extrados On rappelle que dans ce cas, $\mathcal{V}_i^e(t_0) = (\mathcal{V}_{i,s}^e(t_0), \mathcal{V}_{i,z}^e(t_0))$ où

$$\begin{cases} \mathcal{V}_{i,s}^e(t_0) = 2\left(V_{e,s}(t_0)\dfrac{s_c'(t_1)}{\partial F/\partial t_1} + V_{e,z}(t_0)\dfrac{z_c'(t_1)}{\partial F/\partial t_1}\right)s_c'(t_1) - V_{e,s}(t_0), \\ \\ \mathcal{V}_{i,z}^e(t_0) = 2\left(V_{e,s}(t_0)\dfrac{s_c'(t_1)}{\partial F/\partial t_1} + V_{e,z}(t_0)\dfrac{z_c'(t_1)}{\partial F/\partial t_1}\right)z_c'(t_1) - V_{e,z}(t_0). \end{cases}$$

On dérive et on obtient :

$$\begin{aligned}\mathcal{V}^e_{i,s}(t_0)' = \frac{\partial}{\partial t_0}\left(\mathcal{V}^e_{i,s}(t_0)\right) &= 2\frac{s'_c(t_1)}{\partial F/\partial t_1}\left[V'_{e,s}(t_0)s'_c(t_1)+V'_{e,z}(t_0)z'_c(t_1)\right]\\ &+2\frac{s'_c(t_1)}{(\partial F/\partial t_1)^2}\left[V_{e,s}(t_0)\left(\frac{\partial t_1}{\partial t_0}s''_c(t_1)\frac{\partial F}{\partial t_1}-s'_c(t_1)\frac{\partial^2 F}{\partial t_0\partial t_1}\right)\right.\\ &\left.+V_{e,z}(t_0)\left(\frac{\partial t_1}{\partial t_0}z''_c(t_1)\frac{\partial F}{\partial t_1}-z'_c(t_1)\frac{\partial^2 F}{\partial t_0\partial t_1}\right)\right]\\ &+2\frac{\partial t_1}{\partial t_0}\frac{s''_c(t_1)}{\partial F/\partial t_1}\left(V_{e,s}(t_0)s'_c(t_1)+V_{e,z}(t_0)z'_c(t_1)\right)-V'_{e,s}(t_0),\end{aligned}$$

et de façon identique :

$$\begin{aligned}\mathcal{V}^e_{i,z}(t_0)' = \frac{\partial}{\partial t_0}\left(\mathcal{V}^e_{i,z}(t_0)\right) &= 2\frac{z'_c(t_1)}{\partial F/\partial t_1}\left[V'_{e,s}(t_0)s'_c(t_1)+V'_{e,z}(t_0)z'_c(t_1)\right]\\ &+2\frac{z'_c(t_1)}{(\partial F/\partial t_1)^2}\left[V_{e,s}(t_0)\left(\frac{\partial t_1}{\partial t_0}s''_c(t_1)\frac{\partial F}{\partial t_1}-s'_c(t_1)\frac{\partial^2 F}{\partial t_0\partial t_1}\right)\right.\\ &\left.+V_{e,z}(t_0)\left(\frac{\partial t_1}{\partial t_0}z''_c(t_1)\frac{\partial F}{\partial t_1}-z'_c(t_1)\frac{\partial^2 F}{\partial t_0\partial t_1}\right)\right]\\ &+2\frac{\partial t_1}{\partial t_0}\frac{z''_c(t_1)}{\partial F/\partial t_1}\left(V_{e,s}(t_0)s'_c(t_1)+V_{e,z}(t_0)z'_c(t_1)\right)-V'_{e,z}(t_0).\end{aligned}$$

Champ de déformation $\mathcal{V}^c_i$ engendré par une déformation de la cambrure On rappelle que dans ce cas, $\mathcal{V}^c_i(t_0) = (\mathcal{V}^c_{i,s}(t_0), \mathcal{V}^c_{i,z}(t_0))$ où

$$\begin{cases}\mathcal{V}^c_{i,s}(t_0) = 2V_{c,s}(t_1) - A(t_0)s'_e(t_0),\\ \\ \mathcal{V}^c_{i,z}(t_0) = 2V_{c,z}(t_1) - A(t_0)z'_e(t_0),\end{cases}$$

où $A(t_0) = \dfrac{-1}{\partial F/\partial t_0}\left[V_{c,s}(t_1)s'_c(t_1)+V_{c,z}(t_1)z'_c(t_1)+V'_{c,s}(t_1)(s_c(t_1)-s_e(t_0))+V'_{c,z}(t_1)(z_c(t_1)-z_e(t_0))\right]$.
On dérive et on obtient :

$$\begin{cases}\mathcal{V}^c_{i,s}(t_0)' &= \dfrac{\partial}{\partial t_0}\left(\mathcal{V}^c_{i,s}(t_0)\right)\\ &= 2\dfrac{\partial t_1}{\partial t_0}V'_{c,s}(t_1) - \left(A'(t_0)s'_e(t_0)+A(t_0)s''_e(t_0)\right),\\ \\ \mathcal{V}^c_{i,z}(t_0)' &= \dfrac{\partial}{\partial t_0}\left(\mathcal{V}^c_{i,z}(t_0)\right)\\ &= 2\dfrac{\partial t_1}{\partial t_0}V'_{c,z}(t_1) - \left(A'(t_0)z'_e(t_0)+A(t_0)z''_e(t_0)\right),\end{cases}$$

où

$$\begin{aligned}A'(t_0) = \frac{-1}{\partial F/\partial t_0}\Bigg[A(t_0)\frac{\partial^2 F}{\partial t_0^2} &+ \frac{\partial t_1}{\partial t_0}\left(V_{c,s}(t_1)s''_c(t_1)+2V'_{c,s}(t_1)s'_c(t_1)+V''_{c,s}(t_1)(s_c(t_1)-s_e(t_0))\right)\\ &+\frac{\partial t_1}{\partial t_0}\left(V_{c,z}(t_1)z''_c(t_1)+2V'_{c,z}(t_1)z'_c(t_1)+V''_{c,z}(t_1)(z_c(t_1)-z_e(t_0))\right)\\ &-V'_{c,s}(t_1)s'_e(t_0)-V'_{c,z}(t_1)z'_e(t_0)\Bigg].\end{aligned}$$

Les expressions qui ont été calculées dans ce chapitre interviennent dans la formule de gradient incomplet (4.10). Ces résultats peuvent ainsi faire l'objet de la programmation d'un code de calcul de gradient incomplet s'insérant dans un algorithme d'optimisation de forme.

Troisième partie

Optimisation du moyeu

Chapitre 7

Mise en œuvre de l'optimisation

Les formes à optimiser dans la pompe de fond de puits sont celles du moyeu et des aubes. Deux possibilités se sont présentées : réaliser un code résolvant le problème dans son intégralité ou réaliser deux modules séparés, l'un pour le moyeu et l'autre pour les aubes, modules qu'il serait ensuite possible de jumeler pour une optimisation globale.
La seconde solution est apparue plus raisonnable dans la mesure où on devait réaliser un code entier *from scratch*. Le module propre au moyeu en particulier s'annonçait relativement plus simple à coder et donc meilleur pour servir de modèle à ce qui suivrait (tant au niveau du codage, que du *debuggage*, de l'exploitation et de l'interprétation des résultats). Ce premier cas-test devait notamment permettre de juger de l'efficacité de la méthode de gradient incomplet.
On s'intéresse surtout dans cette troisième partie au module d'optimisation du moyeu seul, à sa réalisation et à ses résultats. Cependant, les conclusions que l'on en tire sont souvent d'ordre général et concernent tout aussi bien l'optimisation de forme de la pompe complète.

Dans ce chapitre on commence par dire quelques mots de l'existence d'une solution au problème d'optimisation. Puis on présente le schéma général de l'algorithme d'optimisation, c'est-à-dire applicable aussi bien à l'optimisation de la pompe complète qu'à l'optimisation ciblée du moyeu ou des aubes. On rentre ensuite dans quelques considérations plus détaillées sur les différentes étapes de l'optimisation, telles que le calcul des dérivées des variables d'état, la méthode d'optimisation ou la prise en compte de certaines contraintes. Enfin on s'intéresse plus précisement au problème du moyeu seul.

7.1 A propos de l'existence d'un minimum

Les preuves de l'existence d'un minimum dans les problèmes d'optimisation de forme (comme dans les problèmes d'optimisation plus généraux) se décomposent habituellement en deux étapes :
Etape A : preuve de la compacité de l'ensemble des formes admissibles pour une certaine topologie.
Etape B : preuve de la continuité de la fonctionnelle à minimiser pour cette même topologie.
L'étape A est ici relativement simple.

Lemme 7.1.1. *Compte-tenu des contraintes sur les coordonnées des points de contrôle, l'ensemble des jeux de coordonnées de points de contrôle admissibles est un compact.*

Preuve. On se place dans le cas du moyeu, généré par une B-spline $(z(t), r(t))$ à n points de contrôle $(z_i, r_i)_{i=1\ldots n}$. On rappelle que le moyeu ne peut avoir un diamètre minimum plus petit que $0.4D_t$, ni un diamètre maximum plus grand que $0.9D_t$ et que la longueur axiale d'un étage de pompe ne doit pas

excéder $1.5D_t$ (cf. introduction de ce mémoire). Ces conditions s'écrivent :

$$\forall t \in [0, n-3], \left\{ \begin{array}{rcl} 0.4D_t \leq & r(t) & \leq 0.9D_t, \\ 0 \leq & z(t) & \leq 1.5D_t. \end{array} \right. \tag{7.1}$$

Enfin, qu'elles aient été énumérées en introduction de ce mémoire ou qu'elles le soient pour les besoins du problème simplifié du moyeu au paragraphe 7.6.2, les autres contraintes sur les coordonnées des points de contrôle de la B-spline sont toujours des conditions fermées. Ces contraintes peuvent notamment (et souvent) se traduire par des inégalités larges et non linéaires que l'on peut écrire simplement sous la forme :

$$\forall t \in [0, n-3], F(t, z_1, r_1, ..., z_n, r_n) \geq 0,$$

ce qui équivaut à :

$$\inf_{t \in [0, n-3]} F(t, z_1, r_1, ..., z_n, r_n) \geq 0,$$

où F est une fonction continue en ses $2n+1$ variables. Lorsque l'on passe à l'infimum sur tous les $t \in [0, n-3]$, il reste une fonction semi-continue supérieurement. Pour une telle fonction, l'ensemble des points situés au-dessus d'un niveau (0 ici) est un fermé.
On déduit des considérations précédentes que l'on aura montré le résultat souhaité si on prouve que (7.1) implique que les $2n$-uplets de coordonnées des points de contrôle appartiennent à un compact de $\mathbb{R}^{2n}$.
Il suffit en fait de montrer que si $x(t)$ désigne une B-spline unidimensionnelle de points de contrôle (x_1, x_2, x_3, x_4), et qui vérifie

$$\exists M > 0, \forall t \in [0, 1], |x(t)| \leq M, \tag{7.2}$$

alors (x_1, x_2, x_3, x_4) appartient à un compact de $\mathbb{R}^4$. Pour un paramètre t fixé, l'inégalité (7.2) peut être écrite

$$|x_1 p_3(t) + x_2 p_2(t) + x_3 p_1(t) + x_4 p_0(t)| \leq M, \tag{7.3}$$

ce qui définit une bande de $\mathbb{R}^4$ (c'est-à-dire l'espace limité par les deux hyperplans $x_1 p_3(t) + x_2 p_2(t) + x_3 p_1(t) + x_4 p_0(t) = \pm M$). Chaque fois que l'on fait varier t, on définit une nouvelle bande et le quadruplet (x_1, x_2, x_3, x_4), qui vérifie (7.2), est dans l'intersection de toutes ces bandes. On veut montrer que cette partie, qui est un fermé comme intersection de fermés, est une partie bornée de $\mathbb{R}^4$ donc un compact de $\mathbb{R}^4$.
On commence par montrer que pour tous $(t, t', t'', t''') \in \mathbb{R}^4$ deux à deux distincts, les vecteurs $v(t) = (p_0(t), p_1(t), p_2(t), p_3(t))$, $v(t') = (p_0(t'), ...)$, $v(t'') = (p_0(t''), ...)$ et $v(t''') = (p_0(t'''), ...)$ sont indépendants. Pour cela, on raisonne par l'absurde. S'il existe $(t, t', t'', t''') \in \mathbb{R}^4$ deux à deux distincts tels que la famille $\{v(t), v(t'), v(t''), v(t''')\}$ soit liée, alors

$$\begin{vmatrix} p_0(t) & \dots & p_0(t''') \\ \vdots & & \vdots \\ p_3(t) & \dots & p_3(t''') \end{vmatrix} = 0, \text{ donc } \begin{vmatrix} p_0(t) & \dots & p_3(t) \\ \vdots & & \vdots \\ p_0(t''') & \dots & p_3(t''') \end{vmatrix} = 0.$$

Par suite, il existe $(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \in \mathbb{R}^4$ non tous nuls tels que

$$\left\{ \begin{array}{lll} \alpha_0 p_0(t) & +... & +\alpha_3 p_3(t) & = 0, \\ \vdots & & & \\ \alpha_0 p_0(t''') & +... & +\alpha_3 p_3(t''') & = 0. \end{array} \right.$$

Le polynôme $P = \sum_{i=0}^{3} \alpha_i p_i$ est alors un polynôme du troisième degré s'annulant en quatre points distincts, c'est donc le polynôme nul. Ainsi il existe $(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \in \mathbb{R}^4$ non tous nuls tels que $\sum_{i=0}^{3} \alpha_i p_i = 0$, c'est-à-dire que la famille de polynômes $\{p_0, p_1, p_2, p_3\}$ est liée : absurde.
Par suite on peut choisir $(t, t', t'', t''') \in \mathbb{R}^4$ tels que $\mathcal{B}_v = \{v(t), v(t'), v(t''), v(t''')\}$ soit une base de $\mathbb{R}^4$. D'après (7.2), pour tout vecteur $v \in \mathcal{B}_v$, on a :

$$|(x_4, ..., x_1).v| \leq M.$$

On orthonormalise $\mathcal{B}_v$ en $\mathcal{B} = \{v_1, v_2, v_3, v_4\}$ par le procédé de Schmidt. On voit alors que si (x_4, x_3, x_2, x_1) s'écrit $\beta_1 v_1 + \beta_2 v_2 + \beta_3 v_3 + \beta_4 v_4$, il existe une constante réelle $K > 0$ telle que $|\beta_i| \leq K$ pour $i = 1, ..., 4$. Donc (x_1, x_2, x_3, x_4) appartient à la boule fermée de centre 0 et de rayon K qui est une partie bornée de $\mathbb{R}^4$. □

Remarque 7.1.1. *Cette démonstration s'applique de la même façon au cas des aubes. En effet, le profil 2D des aubes est représenté par une B-spline $(s(t), z(t))$. Cette B-spline est bornée du fait de contraintes techniques. De la même façon que la longueur axiale du moyeu, la longueur axiale d'une aube est bornée car elle est égale, à la constante $(2\,jeu)$ près, à la longueur d'un étage de pompe. La longueur radiale des aubes est limitée par la possibilité technique de manufacturer ces aubes. On retrouve donc des inégalités de la forme (7.1) sur $s(t)$ et $z(t)$. Par ailleurs, de la même façon que pour le moyeu, les autres contraintes sont toujours des conditions fermées.*

Du fait de la définition des B-splines, la convergence d'une suite de points de contrôle $(z_i^k, r_i^k)_{i=1...n}^{k\geq 1}$ vers $(z_i, r_i)_{i=1...n}$ entraîne une convergence de type $\mathcal{C}^2$ des frontières (ce qui est une convergence forte). Néanmoins, en raison du caractère non linéaire de l'équation d'état, la continuité de la fonctionnelle à minimiser est loin d'être évidente. On n'a donc pas cherché dans cette thèse à démontrer l'étape B. On n'a pas trouvé non plus dans la bibliographie de résultats de convergence par rapport au domaine pour la solution du système de Navier-Stokes, que l'on aurait pu utiliser. Même le cas de Stokes ne semble pas complètement clair à l'heure actuelle (à part le cas bidimensionnel traité par Sverak dans [58]). Pour Navier-Stokes, on pourra consulter les travaux récents, mais qui traitent du cas compressible, de Feireisl [29] ou Plotnikov et Sokolowski [54]. Il serait sans doute intéressant de voir si leurs techniques s'adaptent au présent problème mais cela semble dépasser le cadre de cette thèse. En conclusion, on est conduit à penser que le problème d'optimisation du moyeu tel qu'on l'a traité possède bien une solution, mais la démonstration de cette affirmation nécessiterait une étude mathématique supplémentaire.

7.2 Déroulement de l'algorithme d'optimisation

On rappelle que l'on souhaite mener une optimisation de forme en utilisant une méthode de type gradient. On a donc défini une fonction coût J dont on a calculé un gradient incomplet, que l'on note $\tilde{\nabla} J$. Les inconnues de l'optimisation sont les coordonnées des points de contrôle des B-splines générant la géométrie à optimiser. Si n est le nombre de points de contrôle, il y a donc $2n$ inconnues au problème d'optimisation. On note X_k le vecteur à $2n$ composantes des inconnues à la k-ième itération de l'algorithme d'optimisation.
On dispose d'un vecteur initial X_0 à partir duquel on construit la géométrie initiale à l'aide de B-splines cubiques paramétriques. On la maille dans Gambit, le mailleur de Fluent. On modélise l'écoulement dans Fluent, on résout les équations de Navier-Stokes avec turbulence et on calcule la fonction coût $J(X_0)$. On rentre alors dans la boucle d'optimisation à proprement parler. A l'aide de fichiers écrits lors de l'éxécution de Fluent, on récupère les variables d'état et on calcule les dérivées de ces dernières par un schéma numérique. On calcule ensuite le gradient incomplet, à partir des données de Fluent et des résultats de calculs géométriques. On utilise alors une méthode d'optimisation qui doit prendre en compte

les contraintes géométriques. On obtient un nouveau vecteur de variables de contrôle. On construit et maille la géométrie correspondante et on exécute Fluent afin d'obtenir la nouvelle valeur de la fonction coût. On boucle ainsi, jusqu'à ce qu'un test d'arrêt choisi au préalable soit satisfait.

Algorithme 7.1 : Algorithme d'optimisation de forme

entrées : vecteur des variables de contrôle initiales X_0
initialisation :
construction de la géométrie (B-splines) et maillage ;
résolution des équations de Navier-Stokes ;
calcul de la fonction coût $J(X_0)$;
$k \leftarrow 0$;
tant que *le test d'arrêt n'est pas satisfait* **faire**
calcul du gradient incomplet de la fonction objectif $\tilde{\nabla} J(X_k)$;
calcul de l'itéré suivant X_{k+1} par une méthode d'optimisation classique ;
$k \leftarrow k+1$;
construction de la géométrie (B-splines) et maillage ;
résolution des équations de Navier-Stokes ;
calcul de la fonction coût $J(X_{k+1})$;
fin
sorties : vecteur des variables de contrôle X_{k+1}

Un certain nombre de points techniques de l'algorithme d'optimisation sont détaillés dans la suite de ce chapitre, notamment :

- la discrétisation des intégrales surfaciques pour l'évaluation numérique de J et $\tilde{\nabla} J$ (cf. §7.3),
- le schéma numérique utilisé pour le calcul des dérivées des variables d'état (cf. §7.4),
- la méthode d'optimisation (cf. §7.5),
- la prise en compte des contraintes géométriques (cf. §7.6),

7.3 Evaluation numérique de la fonction coût et du gradient incomplet

L'un des enjeux de la programmation du code d'optimisation est de déterminer numériquement la fonction coût J (définie par (4.1)) et le gradient incomplet $\tilde{\nabla} J$ (défini par l'expression (4.10)). Dans les deux cas, on est amené à calculer des intégrales surfaciques. De façon générale celles-ci s'écrivent sous la forme $\int_F \mathcal{G}$, où F désigne la face sur laquelle on intègre et $\mathcal{G}$ la fonction à intégrer.
Pour la mise en œuvre numérique de l'optimisation, on utilise le logiciel de CFD Fluent (cf. §2.1). Fluent donne accès à un certain nombre de valeurs au centre des faces du maillage surfacique (on y reviendra dans la section 7.4). Si on désigne par n_F le nombre de mailles de la surface F, par $(F_i)_{i=1\ldots n_F}$ les mailles en question ($F = \bigcup_{i=1}^{n_F} F_i$), et par c_{F_i} le centre de la maille F_i pour tout $i \in \{1, \ldots, n_F\}$, alors il est possible de calculer une valeur approchée des expressions de la fonction coût ou du gradient en discrétisant les intégrales surfaciques de la façon suivante :

$$\int_F \mathcal{G} \simeq \sum_{i=1}^{n_F} aire(F_i)\mathcal{G}(c_{F_i}).$$

7.4 Calcul des dérivées des variables d'état

Dans le calcul du gradient de la fonction coût, il est nécessaire de connaître les dérivées spatiales des variables d'état sur les bords de la géométrie étudiée. Fluent permet l'obtention d'un certain nombre de

données, dont on fait la liste partielle (limitée aux besoins relatifs au calcul du gradient) dans ce qui suit. Comme on a peu d'informations sur les schémas numériques employés par Fluent, en particulier pour faire les calculs de dérivées, une étude de qualité de ceux-ci doit être faite. Pour cela on met en place des schémas numériques et on compare leurs résultats à ceux de Fluent. Ces schémas numériques pourront éventuellement être conservés et employés pour calculer les dérivées si on n'est pas satisfait de ceux de Fluent.

7.4.1 Les macros de Fluent

Fluent donne accès à des types prédéfinis, tels que les variables solutions de l'écoulement ainsi que leurs dérivées. La programmation d'UDF (*User-Defined Fonction*) en langage C permet de récupérer ces données [30]. Dans le vocabulaire de Fluent, *cellule* fait référence à un volume de contrôle, et *face* fait référence à une face d'un tel volume. Dans la suite, on utilisera ces termes.

Principales variables géométriques

Il existe des macros permettant de récupérer :
- les coordonnées cartésiennes des nœuds du maillage,
- les coordonnées cartésiennes des centres de cellules volumiques,
- le volume d'une cellule volumique,
- les coordonnées cartésiennes des centres de faces,
- le vecteur normal à une face, orienté vers l'extérieur de celle-ci, et de norme égale à la surface de la face.

Variables relatives aux cellules

Des macros, commençant par le préfixe `C_`, donnent accès aux valeurs des variables au centre des cellules volumiques, dans les unités SI. Celles auxquelles on s'intéresse principalement sont :
- `C_P` la pression au centre de la cellule,
- `C_U` la composante u_x de la vitesse dans le repère cartésien au centre de la cellule,
- `C_V` la composante u_y de la vitesse dans le repère cartésien au centre de la cellule,
- `C_W` la composante u_z de la vitesse dans le repère cartésien au centre de la cellule.

Il est possible d'accéder au gradient de ces variables en ajoutant le suffixe `_G` à ces macros. Par exemple `C_V_G[1]` correspond à la dérivée $\frac{\partial u_y}{\partial y}$ (on programme en C donc les vecteurs sont indicés à partir de 0).

Variables relatives aux faces

Les macros commencent par le préfixe `F_` et sont principalement :
- `F_P` la pression au centre de la face,
- `F_U` la composante u_x de la vitesse dans le repère cartésien au centre de la face,
- `F_V` la composante u_y de la vitesse dans le repère cartésien au centre de la face,
- `F_W` la composante u_z de la vitesse dans le repère cartésien au centre de la face.

Aucune dérivée n'est accessible au centre d'une face.

Programmation d'UDF (*User-Defined Function*)

Il est possible de programmer en langage C des fonctions que l'on compile directement dans la console Fluent et qui exécutent des tâches de récupération de données. De telles fonctions sont appelées UDF (*User-Defined Function*).

Pour une surface donnée (ex : le moyeu), les UDF les plus basiques que l'on utilise permettent de récupérer les variables d'état au centre des faces adjacentes ou des cellules adjacentes à la surface considérée, ainsi que les dérivées au centre des cellules, à l'aide des macros qui viennent d'être présentées. On a aussi codé des UDF permettant de récupérer les numéros des cellules voisines à une cellule donnée, ainsi que les coordonnées du centre de ces cellules, les variables d'état au centre, etc. De telles UDF sont très utiles pour mettre en place des schémas numériques de calcul des dérivées des variables d'état, comme on va le voir en 7.4.2 et 7.4.3.

Schémas numériques dans Fluent

Comme on l'a vu au paragraphe 7.3, dans le calcul du gradient de la fonction coût on considère des intégrales surfaciques et on a donc besoin des valeurs des variables d'état et de leurs dérivées au centre des faces, sur les parois. On ignore si on peut se satisfaire de l'approximation consistant à dire que les valeurs au centre des faces sont proches de celles au centre des cellules volumiques, surtout pour les dérivées. On s'intéresse donc dans un premier temps, et avant de proposer d'autres schémas numériques, à ceux mis en œuvre par Fluent pour déterminer les valeurs des variables d'état au centre des faces et de leurs dérivées au centre des cellules volumiques.
Comme on l'a évoqué au paragraphe 2.1.2, Fluent utilise une méthode de résolution par volumes finis. Les équations d'état sont d'abord intégrées sur chaque cellule, puis discrétisées, et enfin linéarisées.
Par défaut, Fluent mémorise les valeurs d'une variable scalaire ϕ au centre des cellules. Cependant, les valeurs au centre des faces ϕ_f sont nécessaires à la résolution des équations d'état discrétisées. Il faut donc déterminer ces valeurs par interpolation de celles au centre des cellules, ce qui est fait à l'aide d'un schéma *upwind* (plusieurs sont proposés à l'utilisateur). Ces schémas portent ce nom générique car ϕ_f est calculée à partir des valeurs prises dans les cellules situées en amont de l'écoulement.
On a choisi d'utiliser un schéma au second ordre. Il s'agit d'une approximation d'un développement en série de Taylor de la forme :

$$\phi_f \simeq \phi + \nabla\phi.\delta$$

où ϕ et $\nabla\phi$ sont les valeurs de la variable d'état et de son gradient au centre de la cellule amont, et δ est le vecteur reliant le centre de la cellule amont à celui de la face considérée.
En combinant les formules du point milieu et de Green, on exprime $\nabla\phi$ en chaque centre de cellule comme étant la somme discrète sur les faces de la cellule :

$$\nabla\phi \simeq \frac{1}{V}\sum_f \widetilde{\phi_f}.a$$

où a désigne le vecteur normal à la face, de norme égale à l'aire de celle-ci, et $\widetilde{\phi_f}$ est une approximation de ϕ_f égale à la moyenne des valeurs prises par ϕ au centre de chacune des deux cellules adjacentes à la face considérée.
Dans le paragraphe qui suit, on procède à une rapide comparaison quantitative des valeurs prises par une variable donnée par Fluent au centre des faces ou au centre des cellules volumiques.

Remarques sur les données fournies par Fluent

Lorsque l'on trace la courbe de la pression statique au centre des cellules volumiques, et de la pression statique au centre des faces, en fonction de la longueur axiale z, les deux nuages de points sont quasiment superposés (la différence peut être grossièrement estimée inférieure à 1%). Il n'y a donc pas une différence significative pour la pression statique entre les valeurs au centre des cellules et au centre des faces.

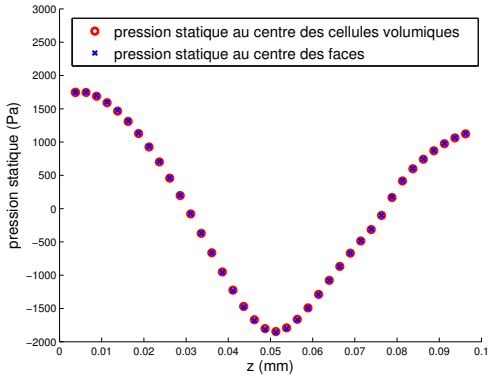

FIG. 7.1: *Profils de la pression statique au moyeu.*

Si on procède de même pour la troisième composante de la vitesse, la différence devient significative (de l'ordre de 20%).

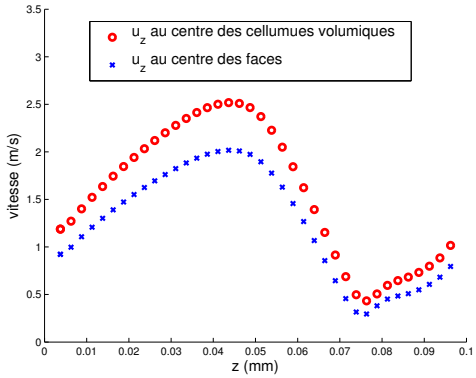

FIG. 7.2: *Profil de la composante u_z de la vitesse au moyeu.*

Dans un premier temps on va donc construire un schéma numérique pour calculer les variables d'état au centre des faces à partir des valeurs au centre des cellules volumiques et comparer ses résultats à ceux obtenus par le schéma numérique de Fluent. Puisque l'on souhaite également connaître les dérivées au centre des faces, on utilise un développement limité d'ordre 1 pour écrire le schéma numérique.

7.4.2 Schéma numérique alternatif pour le calcul des variables d'état au centre des faces

On considère une surface F et on cherche à calculer une variable d'état scalaire ϕ et ses dérivées au centre des faces du maillage de F. Comme on l'a évoqué en 7.4.1, on peut écrire des UDF récupérant un nombre au choix de voisines pour une cellule volumique donnée, adjacente à F.
Soient (x,y,z) les coordonnées du centre d'une face donnée, appartenant à F. On pose $\phi_f = \phi(x,y,z)$.
Soient $(x_i,y_i,z_i)_{i=1\ldots k}$ les coordonnées du centre de la cellule à laquelle appartient la face considérée et des centres des $k-1$ cellules voisines que l'on choisit de prendre en compte. Fluent donne accès aux valeurs $\phi_i = \phi(x_i,y_i,z_i)$ pour $i = 1,\ldots,k$.
On utilise l'approximation à l'ordre 1 :

$$\phi(x+dx,y+dy,z+dz) \simeq \quad \phi(x,y,z) + dx\frac{\partial\phi(x,y,z)}{\partial x} + dy\frac{\partial\phi(x,y,z)}{\partial y} + dz\frac{\partial\phi(x,y,z)}{\partial z}. \tag{7.4}$$

On pose $dx_i = x_i - x$, $dy_i = y_i - y$ et $dz_i = z_i - z$ pour $i = 1, ..., k$. Alors, en écrivant le développement limité pour chaque $\phi_i = \phi(x + dx_i, y + dy_i, z + dz_i)$, on obtient le système (en général surdéterminé) :

$$\begin{pmatrix} \phi_1 \\ \vdots \\ \phi_k \end{pmatrix} \simeq \begin{pmatrix} 1 & dx_1 & dy_1 & dz_1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & dx_k & dy_k & dz_k \end{pmatrix} . \begin{pmatrix} \phi_f \\ \frac{\partial \phi_f}{\partial x} \\ \frac{\partial \phi_f}{\partial y} \\ \frac{\partial \phi_f}{\partial z} \end{pmatrix} . \tag{7.5}$$

On simplifie l'écriture de (7.5) : il s'agit de résoudre un système de la forme

$$\Phi = A.\partial\phi_f. \tag{7.6}$$

Comme le conditionnement de A est très grand en général, on choisit la méthode classique de factorisation QR pour résoudre le système des équations normales ${}^tAA\partial\phi_f = {}^tA\Phi$. Ainsi, si on factorise A en $A = QR$ avec Q orthogonale et R triangulaire, on obtient ${}^tAA\partial\phi_f = {}^tRR\partial\phi_f$. Il reste à résoudre deux systèmes triangulaires : ${}^tRY = {}^tA\Phi$, et ensuite $R\partial\phi_f = Y$. Dans la mise en œuvre numérique, on surveillera le conditionnement de ${}^tAA$.
Dans les paragraphes suivants, on présente les résultats obtenus grâce au schéma numérique. Les tests ont été menés sur la troisième composante de la vitesse ($\phi = u_z$) et ses dérivées, au moyeu.

Résultats pour 5 cellules volumiques voisines

On appelle ce schéma *5C* car il prend en compte 5 centres de cellules volumiques voisines de la face considérée.

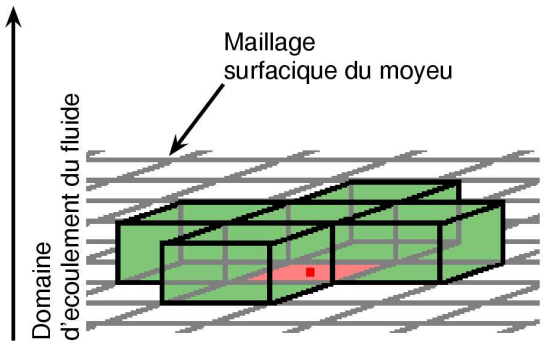

FIG. 7.3: *5 cellules dont on considère les centres pour calculer u_z au centre de face rouge.*

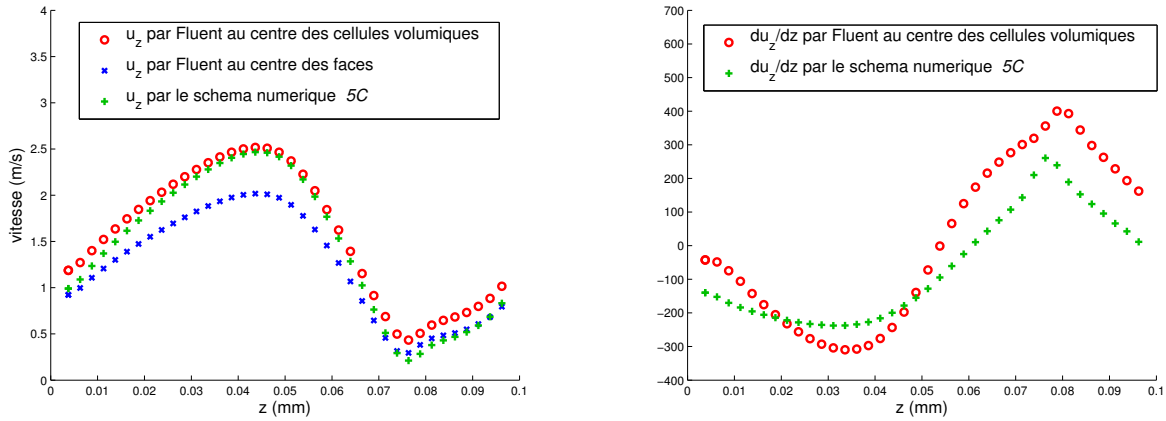

FIG. 7.4: *Valeurs de u_z et de du_z/dx obtenues par le schéma 5C, comparées aux valeurs données par Fluent.*

Le conditionnement des matrices ${}^tAA$ est de l'ordre de 10^7. Le schéma *5C* donne des résultats cohérents avec ceux de Fluent, même si on constate que pour u_z il renvoie des valeurs proches de celles au centre des cellules volumiques et non au centre des faces, comme on l'espérait.

Résultats pour 14 cellules volumiques voisines

On appelle ce schéma *14C* car il prend en compte 14 centres de cellules volumiques voisines de la face considérée.

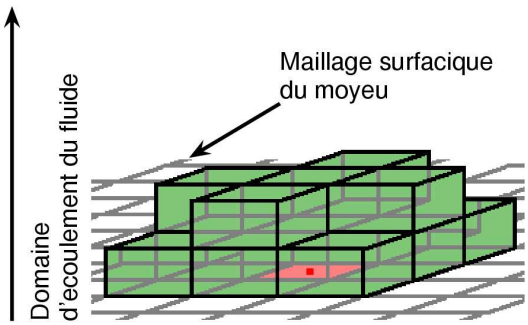

FIG. 7.5: *14 cellules dont on considère les centres pour calculer des valeurs au centre de face rouge.*

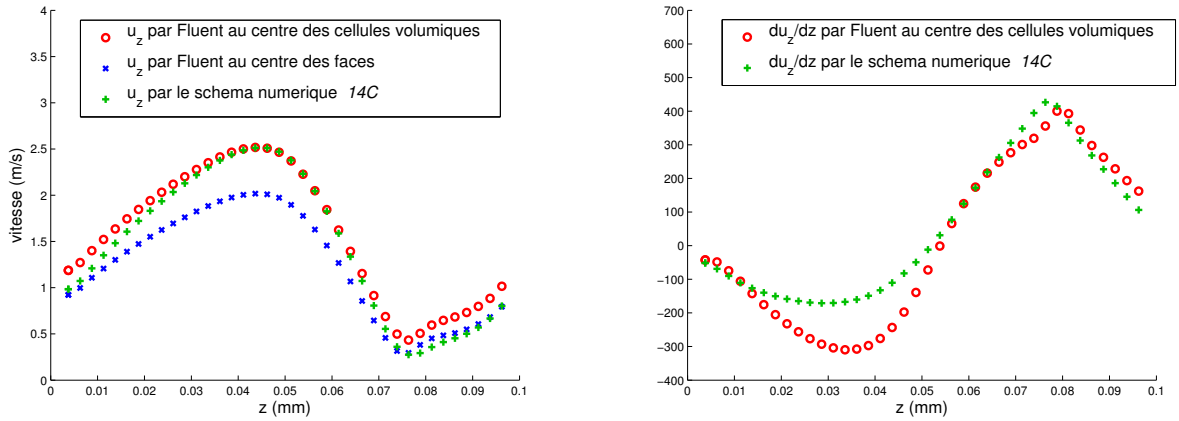

FIG. 7.6: *Valeurs de u_z et de du_z/dx obtenues par le schéma 14C, comparées aux valeurs données par Fluent.*

Le conditionnement des matrices ${}^tAA$ est de l'ordre de 10^7. Le schéma *14C* donne des résultats cohérents avec ceux de Fluent. Cependant, encore une fois, le schéma renvoie pour u_z des valeurs proches de celles au centre des cellules volumiques et non au centre des faces.
Dans le paragraphe suivant, on prend par conséquent pour acquises les valeurs des variables d'état au centre des faces fournies par Fluent. On étudie les résultats d'un schéma numérique calculant, à l'aide de ces valeurs notamment, les valeurs des dérivées des variables d'état au centre des faces.

7.4.3 Schéma numérique pour le calcul des dérivées au centre des faces

Cette fois on utilise la valeur de la variable d'état ϕ_f donnée par Fluent au centre des faces. Ainsi, si on reprend les notations de 7.4.3, $\phi_f = \phi(x,y,z)$ est une donnée et les $\phi_i = \phi(x_i,y_i,z_i)$ peuvent être prises au centre des cellules volumiques, mais aussi des faces. En utilisant le même schéma numérique (7.4) d'ordre 1 que précédemment, pour chaque $\phi_i = \phi(x+dx_i, y+dy_i, z+dz_i)$, on obtient le système :

$$\begin{pmatrix} \phi_1 - \phi_f \\ \vdots \\ \phi_k - \phi_f \end{pmatrix} \simeq \begin{pmatrix} dx_1 & dy_1 & dz_1 \\ \vdots & \vdots & \vdots \\ dx_k & dy_k & dz_k \end{pmatrix} \cdot \begin{pmatrix} \frac{\partial \phi_f}{\partial x} \\ \frac{\partial \phi_f}{\partial y} \\ \frac{\partial \phi_f}{\partial z} \end{pmatrix}.$$

Encore une fois, on écrit ce système sous forme simplifiée :

$$d\phi = B.\nabla\phi_f. \tag{7.7}$$

On utilise encore une méthode de factorisation *QR* pour résoudre le système, en surveillant le conditionnement de ${}^tBB$.
Dans les paragraphes suivants, on présente les résultats obtenus grâce au schéma numérique. Les tests ont été menés sur la troisième composante de la vitesse ($\phi = u_z$) et ses dérivées, au moyeu.

Résultats pour 5 points voisins -2-

On appelle ce schéma *5C-2* car il prend en compte 5 centres de cellules volumiques, comme représenté sur la figure FIG.7.3.

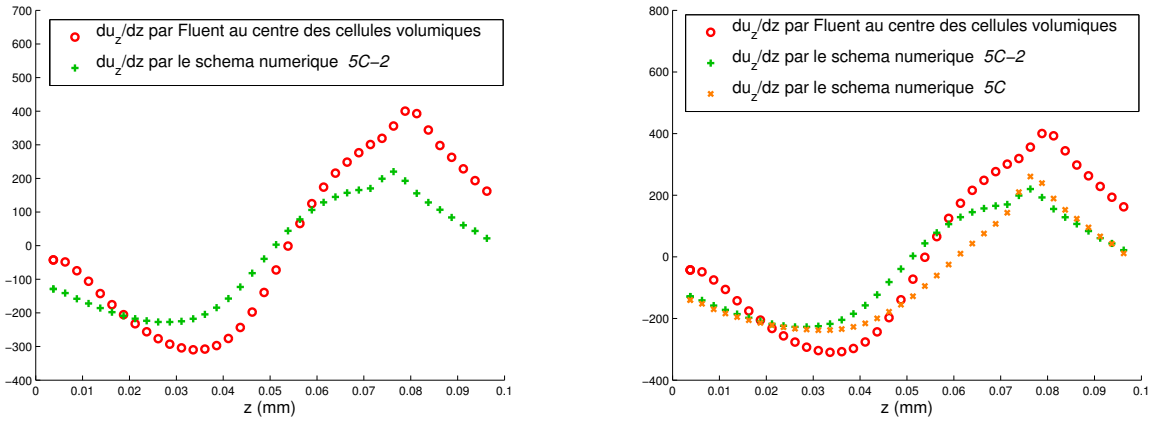

FIG. 7.7: *A gauche : valeurs de du_z/dx pour le schéma 5C-2. A droite : comparaison entre les schémas numériques 5C et 5C-2.*

Le conditionnement de ${}^tBB$ pour le schéma *5C-2* est compris entre 40 et 90. Encore une fois, le schéma *5C-2* donne des résultats cohérents avec ceux de Fluent. Il ne se distingue pas particulièrement du schéma *5C*.

Résultats pour 14 points voisins -2-

On appelle ce schéma *14C-2* car il prend en compte 14 centres de cellules volumiques voisines de la face considérée. Ces cellules sont les mêmes que représentées sur FIG.7.5.

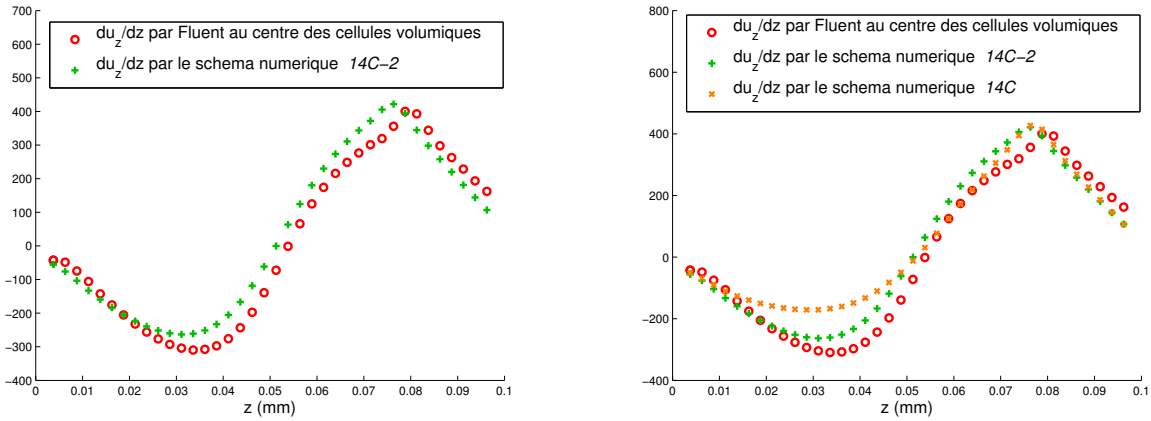

FIG. 7.8: *A gauche : valeurs de du_z/dx pour le schéma 14C-2. A droite : comparaison entre les schémas numériques 14C et 14C-2.*

Le conditionnement de ${}^tBB$ pour le schéma numérique *14C-2* est compris entre 40 et 90. Le schéma *14C-2* donne des résultats cohérents avec ceux de Fluent.

7.4.4 Conclusions

Les schémas numériques calculant les variables d'état au centre des faces présentent un grand conditionnement de la matrice ${}^tAA$. Par ailleurs, ils fournissent des valeurs des variables d'état au centre des

faces proches de celles rendues par Fluent au centre des cellules volumiques. On retient donc le schéma numérique de Fluent pour déterminer les valeurs des variables d'état au centre des faces.

En revanche on choisit de faire confiance au schéma numérique *14C-2* pour la détermination des dérivées des variables d'état au centre des faces. On préfère le schéma *14C-2* au schéma *5C-2* car les courbes des dérivées semblent plus régulières.

7.5 Algorithmes d'optimisation sans contrainte

On cherche à résoudre des problèmes du type :

$$\min_{X\in\mathbb{R}^{2n}} J(X). \tag{7.8}$$

On considère des algorithmes de la forme :

$$X_0 \text{ donné}, X_{k+1} = X_k + t_k d_k \tag{7.9}$$

où d_k est la *direction de descente* et t_k le *pas*. En effet on cherche à minimiser la fonction coût J et pour cela à avoir :

$$J(X_{k+1}) \leq J(X_k),$$

d'où le nom de **méthode de descente** donné à ce type d'algorithmes.
Dans une méthode de descente, l'algorithme choisit une direction de descente d_k et cherche dans cette direction, depuis le point courant X_k, un nouveau point où la fonction coût prendra une valeur moindre. La distance à parcourir dans la direction de descente (le pas t_k), peut être trouvée en résolvant le problème de minimisation de dimension un :

$$\min_{t>0} J(X_k + t d_k). \tag{7.10}$$

Remarque 7.5.1. *On présente dans la section 7.6.1 la géométrie étudiée pour l'optimisation du moyeu seul. On peut d'ores et déjà préciser que l'on parle bien dans ce cas d'une méthode de "descente" car il n'y a pas de gain de pression, mais seulement une perte dans cette géométrie. Par conséquent, plutôt que de considérer que l'on maximise le gain de pression, on considère, de façon équivalente, que l'on minimise la perte de pression. Ainsi, la fonction coût pour le problème de minimisation est la valeur absolue de la fonction coût à maximiser pour le problème de la pompe complète :*

$$J = \left| \frac{\int_S p - \int_E p}{L} \right| = \frac{\int_E p - \int_S p}{L}.$$

7.5.1 Méthodes de gradient

Direction de plus grande pente

La direction $-\nabla J(X_k)$ est la direction de descente la plus pertinente. En effet, si on écrit le développement limité de J pour une direction d et un pas t quelconques, on a :

$$J(X_k + td) = J(X_k) + td.\nabla J(X_k) + O(t^2) \tag{7.11}$$

et la direction unitaire d de plus rapide décroissance de J est la solution du problème :

$$\min_{\{d, \|d\|=1\}} d.\nabla J(X_k). \tag{7.12}$$

On a $d.\nabla J(X_k) = \|d\|\|\nabla J(X_k)\|\cos\beta = \|\nabla J(X_k)\|\cos\beta$ (car $\|d\| = 1$), où β est l'angle entre d et $\nabla J(X_k)$. La solution du problème (7.12) est atteinte quand $\cos\beta = -1$, c'est-à-dire pour $d = -\nabla J(X_k)/\|\nabla J(X_k)\|$. L'avantage de cette direction est qu'elle ne demande que le calcul du gradient $\nabla J(X_k)$ mais pas des dérivées secondes. En général il y a convergence vers un minimum (local), cependant la convergence peut être très lente.
Les méthodes de gradient peuvent utiliser d'autres directions. En général, toute direction faisant un angle strictement plus petit que $\pi/2$ avec $-\nabla J(X_k)$ permet de faire décroître la fonction J, à condition que le pas soit assez petit. En effet, si on reprend le développement limité (7.11), pour une direction d telle que $\cos\beta < 0$, alors $d.\nabla J(X_k) = \|d\|\|\nabla J(X_k)\|\cos\beta < 0$ et $J(X_k + td) < J(X_k)$ pour tout t positif et suffisament petit.

Choix du pas

Le choix du pas place le programmeur devant un dilemme : il aimerait choisir le pas assez grand pour réduire de façon significative la fonction coût J et pour que l'algorithme avance plus vite, mais il ne peut pas le choisir trop grand, sinon la fonction coût risque de ne pas décroître, ou d'avoir un comportement aléatoire. De plus il souhaite faire le choix le plus précis possible, sans pour autant y consacrer trop de temps. Le choix idéal serait de trouver le minimum global de la fonction d'une variable q définie par :

$$q(t) = J(X_k + td_k),\ t > 0.$$

On parle alors de *méthode de gradient à pas optimal*.
Il est souvent trop coûteux (en évaluations de f et de son gradient) de chercher cette valeur. En pratique on procède donc à une *recherche linéaire* inexacte de façon à trouver un pas permettant une réduction substantielle de J. En général les recherches linéaires testent une séquence de candidats et s'arrêtent lorsqu'une valeur satisfaisante a été trouvée.
On peut aussi utiliser simplement une *méthode de gradient à pas fixe* qui, comme son nom l'indique, utilise un pas t fixe que l'on peut modifier de temps en temps au cours de l'optimisation.

Tests d'arrêt

Les tests d'arrêt employés pour ce type d'algorithme portent classiquement sur :
- $|\nabla J(X_k)| < \upsilon$
- ou $|X_k - X_{k+1}| < \upsilon$

pour un $\upsilon > 0$ fixé à l'avance.

7.5.2 Méthodes de Newton et de quasi-Newton

Direction de Newton

La *direction de Newton* provient du développement limité au second ordre de $J(X_k + d)$:

$$J(X_k + d) \sim J(X_k) + d.\nabla J(X_k) + \frac{1}{2}{}^T dD^2J(X_k)d := m_k(d). \tag{7.13}$$

En supposant pour le moment que $D^2J(X_k)$ est définie positive, on obtient la direction de Newton en trouvant le vecteur d qui minimise $m_k(d)$. En annulant la dérivée de $m_k(d)$ on a la formule explicite :

$$d_k = -[D^2J(X_k)]^{-1}\nabla J(X_k). \tag{7.14}$$

Cette expression n'étant pas appropriée au calcul numérique (l'inversion d'une matrice est coûteuse), on écrit l'algorithme de Newton sous la forme :

Algorithme 7.2 : Algorithme de Newton

initialisation : choisir X_0 ;
tant que *le test d'arrêt n'est pas satisfait* **faire**
résoudre le système linéaire

$$D^2J(X_k)d_k = -\nabla J(X_k) \; ; \tag{7.15}$$

calcul de l'itéré suivant :
$X_{k+1} = X_k + d_k$;
$k \leftarrow k+1$;
fin

La direction de Newton peut être utilisée comme direction de descente car, lorsque $D^2J(X_k)$ est définie positive, il existe un $\sigma_k > 0$ tel que :

$$d_k.\nabla J(X_k) = -^T d_k D^2J(X_k)d_k \leq -\sigma_k \|d_k\|^2.$$

A moins que $\nabla J(X_k) = 0$ (et donc $d_k = 0$), on a $d_k.\nabla J(X_k) < 0$ donc une direction de descente.
A la différence du pas de la méthode de gradient, 1 est le pas "naturel" associé à la direction de Newton. Cependant il peut être nécessaire de l'adapter dans certains cas.
La convergence de la méthode de Newton n'est pas assurée et il vaut mieux partir d'un point initial proche de la solution. Mais quand la méthode converge, la convergence est rapide. Par ailleurs si $D^2J(X_k)$ n'est pas définie positive, la direction de Newton peut ne même pas être définie ($[D^2J(X_k)]^{-1}$ n'existe pas). Et même si elle est définie, elle peut ne pas satisfaire la propriété de descente $d_k.\nabla J(X_k) < 0$. Enfin cette méthode est très coûteuse puisqu'en plus des $2n$ composantes du gradient, il faut déterminer les $(2n)^2$ composantes de la Hessienne, et aussi résoudre le système linéaire (7.15), le tout à chaque itération.

Méthodes de quasi-Newton

Les *méthodes de quasi-Newton* ont pour avantage de ne pas nécessiter le calcul de la matrice Hessienne, tout en convergeant rapidement (quand il y a convergence...). On calcule une approximation B_k de $D^2J(X_k)$ ou, mieux, H_k de $[D^2J(X_k)]^{-1}$.
Une des formules les plus connues de mise à jour de l'approximation de l'inverse de la Hessienne est la formule *BFGS* (Broyden-Fletcher-Goldfarb-Shanno) :

$$H_{k+1} = H_k - \frac{S_k{}^T Y_k H_k + H_k Y_k{}^T S_k}{Y_k S_k} + \left(1 + \frac{^T Y_k H_k Y_k}{Y_k S_k}\right) \frac{S_k{}^T S_k}{Y_k S_k} \tag{7.16}$$

où on a posé $S_k = X_{k+1} - X_k$ et $Y_k = \nabla J(X_{k+1}) - \nabla J(X_k)$. Le choix de l'approximation H_0 est laissée au programmeur et doit être fait avec soin.

7.5.3 Conclusion

Après avoir présenté ces méthodes, on doit considérer leurs inconvénients et avantages au regard du fait qu'à chaque itération de l'algorithme d'optimisation on ne calcule qu'une valeur incomplète du gradient de la fonction coût.
La méthode de Newton est écartée car elle demande le calcul de l'inverse de la Hessienne. On émet des réserves quant aux méthodes de type quasi-Newton, car elles proposent une approximation de l'inverse de la Hessienne basée sur la valeur du gradient, qui est elle-même approchée. On peut donc craindre que

ce ne soit pas une méthode de descente (bien que la méthode BFGS ait été employée avec succès avec des sensitivités incomplètes dans [20]).
On retient donc de préférence une méthode de gradient classique. Une recherche linéaire serait trop coûteuse en évaluations de la fonction coût et de son gradient. De plus, le gradient étant approché, il est possible que la recherche linéaire ne donne pas de résultats convaincants. Par conséquent on choisit une méthode de gradient à pas fixe et on ajuste le pas lorsque cela semble nécessaire à la bonne exécution du code d'optimisation.

7.6 Prise en compte des contraintes géométriques pour le problème du moyeu

Les contraintes recensées ci-après ne sont pas exactement celles énoncées pour le moyeu dans l'introduction de ce mémoire, mais sont adaptées aux spécificités du problème simplifié, sur lequel on revient en détail pour commencer.

7.6.1 Position du problème spécifique d'optimisation du moyeu

La géométrie considérée est comparable à un étage de pompe auquel on a retiré les aubes et dont le moyeu est fixe, sans distinction entre un rotor et un stator. Comme évoqué dans le remarque 7.5.1, l'objectif est ici de maximiser le gain de pression par unité de longueur, comme dans le problème d'optimisation posé pour la pompe complète, mais dans la mesure où il n'y a pas de gain de pression dans la géométrie mais seulement une perte, le problème se ramène, de façon équivalente, à la minimisation de la fonction coût suivante :

$$J = \frac{\int_E p - \int_S p}{L}.$$

Pour la simplicité et la rapidité des calculs, on ne considère pas un écoulement périodique mais un écoulement avec conditions sur la vitesse en entrée et sur la pression en sortie. La géométrie contenant le moyeu à optimiser est encadrée de deux extensions constituées d'un carter et d'un moyeu cylindriques. Les extensions sont fixes du point de vue de l'optimisation, c'est-à-dire que la longueur axiale du moyeu ne peut pas varier. Ceci a deux conséquences : des contraintes géométriques devront être définies (cf. §7.6.2), et la dérivée du dénominateur L de la fonction coût est nulle.

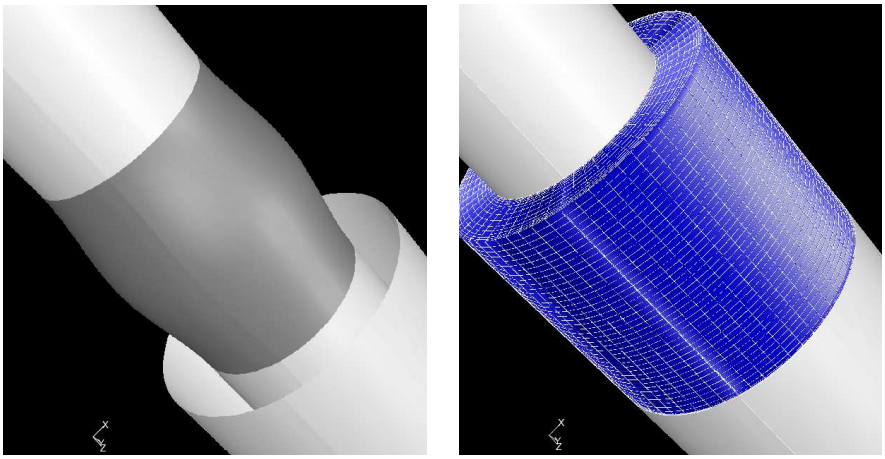

FIG. 7.9: *A gauche : le moyeu à optimiser encadré des moyeux des extensions et du carter de l'extension située en aval ; à droite : maillage structuré du canal d'écoulement du fluide entre moyeu et carter.*

On rappelle que le moyeu est une surface de révolution. Dans le cas présent, où l'on ne distingue pas un rotor et un stator, la surface du moyeu est générée par une B-spline 2D paramétrée du troisième degré. Dans ce qui va suivre, on notera $(z_1, r_1), ..., (z_n, r_n)$ les coordonnées des n points de contrôle de la B-spline. Ainsi le problème d'optimisation considéré possède $2n$ inconnues.

7.6.2 Recensement des contraintes

Contraintes d'égalité

Selon le nombre n de points de contrôle, on va pouvoir imposer un certain nombre de contraintes.
La première, indispensable au problème, est la continuité $\mathcal{C}^0$ aux raccords du moyeu avec les extrémités. Cela se traduit par les égalités recensées ci-après et numérotées de (7.17) à (7.20).
Les extensions sont cylindriques, donc générées par des droites. Par conséquent on peut imposer le caractère $\mathcal{C}^1$ en contraignant les valeurs des dérivées premières dans la direction r à être nulles : ce sont les égalités (7.21) et (7.22).
Enfin on peut imposer une contrainte d'égalité pour le caractère $\mathcal{C}^2$ aux raccords, dans la direction r : ce sont les égalités (7.23) à (7.24).
On établit ainsi la liste des contraintes d'égalité pouvant (ou devant) être imposées :

$$z_1 p_3(0) + z_2 p_2(0) + z_3 p_1(0) + z_4 p_0(0) = 0 \tag{7.17}$$
$$r_1 p_3(0) + r_2 p_2(0) + r_3 p_1(0) + r_4 p_0(0) = D_1/2 \tag{7.18}$$
$$z_{n-3} p_3(1) + z_{n-2} p_2(1) + z_{n-1} p_1(1) + z_n p_0(1) = L \tag{7.19}$$
$$r_{n-3} p_3(1) + r_{n-2} p_2(1) + r_{n-1} p_1(1) + r_n p_0(1) = D_1/2 \tag{7.20}$$
$$r_1 p_3'(0) + r_2 p_2'(0) + r_3 p_1'(0) + r_4 p_0'(0) = 0 \tag{7.21}$$
$$r_{n-3} p_3'(1) + r_{n-2} p_2'(1) + r_{n-1} p_1'(1) + r_n p_0'(1) = 0 \tag{7.22}$$
$$r_1 p_3''(0) + r_2 p_2''(0) + r_3 p_1''(0) + r_4 p_0''(0) = 0 \tag{7.23}$$
$$r_{n-3} p_3''(1) + r_{n-2} p_2''(1) + r_{n-1} p_1''(1) + r_n p_0''(1) = 0 \tag{7.24}$$

Lorsque l'on présentera les résultats de l'algorithme d'optimisation, on précisera au cas par cas les contraintes qui ont été prises en compte.

Contraintes d'inégalité

Signe des dérivées Lorsque le nombre d'inconnues devient grand et qu'il y a donc un grand nombre de degrés de liberté, alors les contraintes d'égalité deviennent insuffisantes pour contrôler le comportement de la B-spline à ses deux extrémités. En effet, si la première composante de la tangente au moyeu (dans la direction z) à l'un des raccords avec les extensions est négative, on obtient une géométrie aberrante. Par conséquent on aimerait pouvoir imposer les deux contraintes d'inégalité suivantes :

$$z_1 p_3'(0) + z_2 p_2'(0) + z_3 p_1'(0) + z_4 p_0'(0) \geq 0 \tag{7.25}$$
$$z_{n-3} p_3'(1) + z_{n-2} p_2'(1) + z_n p_1'(1) + z_{n-1} p_0'(1) \geq 0 \tag{7.26}$$

Courbure de la B-spline génératrice du moyeu Lorsque l'on utilise un grand nombre de points de contrôle, la B-spline générée par ces points devient très "souple" et sa courbure peut alors prendre de grandes valeurs. Or, dans le cadre d'une optimisation de forme, on souhaite éviter que la forme soit trop perturbée. En effet, les fortes courbures posent divers problèmes, comme la difficulté de fabrication ou la création de zones de recirculation du fluide, sans parler des cas extrêmes qui sont des aberrations numériques provoquant simplement l'arrêt prématuré du code. Pour ces raisons, on souhaite contraindre

la courbure de la B-spline génératrice du moyeu.
A cet effet, on a fait des essais peu concluants pour mettre en place une méthode de pénalisation. En reprenant les notations du paragraphe 7.3, la fonction pénalisante retenue est la somme sur les faces appartenant au moyeu :

$$\sum_{i=1}^{n_F} aire(F_i)H(c_{F_i})$$

où $H(c_{F_i})$ est la courbure moyenne au centre c_{F_i} de la face F_i, courbure calculée grâce à la formule (6.4). On donne quelques éléments de réflexion supplémentaires sur la méthode de pénalisation au paragraphe 8.1.3.

7.6.3 Réduction du problème

On considère le problème suivant :

$$\begin{cases} \min\limits_{X\in\mathbb{R}^{2n}} J(X) \\ c_i(X) = 0, \quad i \in \{1,...,m\}, \end{cases} \tag{7.27}$$

où les contraintes c_i sont m contraintes d'égalité linéaires prises parmi (7.17) à (7.24). En utilisant les m contraintes d'égalité, on ramène le problème à $2n$ inconnues (7.27) à un problème sans contrainte à $2n-m$ inconnues que l'on résout avec l'une des méthodes de descente (sans contrainte) proposées précédemment. Pour cela on considère donc un gradient restreint de la fonction coût, n'ayant que $2n-m$ composantes. A chaque itération de l'algorithme d'optimisation, on retrouve les $2n$ coordonnées des points de contrôle en calculant les m coordonnées manquantes grâce aux contraintes d'égalité.
Il est clair qu'une telle méthode ne permet pas de prendre en compte des contraintes d'inégalité. Dans cette mesure, et sachant que l'on n'a présenté ici, et testé numériquement, que des méthodes d'optimisation sans contrainte, on renvoie aux perspectives du paragraphe 8.1.3 pour des pistes traitant de la prise en compte des contraintes d'inégalité (telles que celles énoncées ci-avant).

7.6.4 Choix de la géométrie initiale

Sauf recherche particulière, la géométrie initiale est toujours construite pour ressembler à la géométrie initiale du moyeu de la pompe à optimiser.
Au minimum la B-spline du moyeu initial est constituée d'un seul intervalle, généré par quatre points de contrôle. Il faut donc écrire et résoudre un système de 8 équations afin de calculer les 8 coordonnées des points de contrôle. Quatre égalités du système sont données par les contraintes indispensables (7.17) à (7.20).
Pour que le moyeu initial soit comparable à celui de la pompe complète, on impose aussi qu'en leur milieu (en $L/2$) ils soient de rayons égaux (à $D_2/2$). On impose aussi que la tangente au milieu du moyeu soit horizontale et on peut fixer sa norme arbitrairement de sorte que le moyeu ait une forme ni trop écrasée, ni trop pointue.
Dans le cas où l'on a davantage de points de contrôle, on peut par exemple compléter le système avec des égalités choisies de (7.21) à (7.24).

Chapitre 8

Analyse critique de la méthode de gradient incomplet

On présente dans ce chapitre les résultats numériques de l'optimisation de forme du moyeu de la pompe avec la méthode de gradient incomplet ainsi que des études réalisées pour la validation de cette méthode. Le problème considéré est celui qui a été détaillé dans la section 7.6.
Pour la plupart, les tests sont menés sur des cas simples, avec un faible nombre d'inconnues et de contraintes pour le problème d'optimisation considéré. Ceci est dû au temps de calcul considérable que demandent les cas plus complexes ainsi qu'aux difficultés numériques que l'on a rencontrées et que l'on met en évidence ici. Cependant cela permet une comparaison avec les résultats d'une méthode de différences finies, méthode qu'il n'est pas envisageable de mettre en place pour des cas plus complexes.

8.1 Résultats numériques pour la méthode de gradient incomplet

8.1.1 Contraintes de continuité $\mathcal{C}^0$

On commence par étudier le cas le plus simple qui soit, où seules les contraintes indispensables sont imposées et où l'on utilise le minimum de points de contrôle pour générer la B-spline du moyeu. Les contraintes retenues sont (7.17) à (7.20), qui garantissent le caractère $\mathcal{C}^0$ à l'interface entre le moyeu et les extensions. Pour tracer une B-spline, il faut un minimum de quatre points de contrôle. Ainsi, avec un total de quatre contraintes et de huit coordonnées de points de contrôle, le problème considéré comporte quatre degrés de liberté.
Pour l'optimisation, on commence par réduire le problème puis on utilise une méthode classique de descente à pas fixe, telle que présentée en 7.5.1. Après plusieurs essais, le pas a été choisi le plus grand possible afin que l'algorithme progresse assez vite, mais suffisamment petit pour éviter de trop fortes déformations et, par suite, la génération de géométries aberrantes.
Sur la figure FIG.8.1, on a tracé l'évolution de la fonction coût J (i.e. la perte de pression par unité de longueur) au cours des itérations. On constate une décroissance satisfaisante de J dans les premières itérations seulement. Après avoir atteint une valeur minimale à la 16ème itération, J converge vers une valeur supérieure.

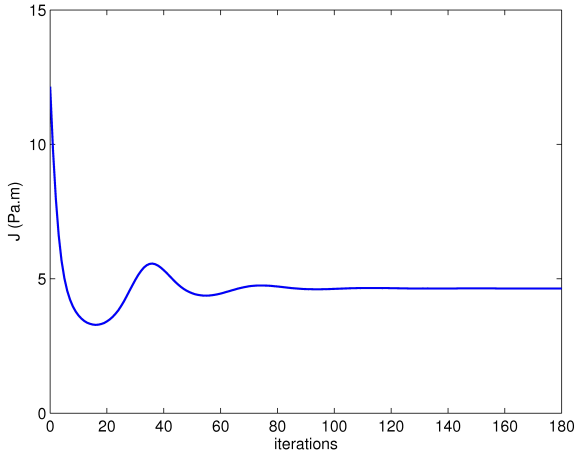

FIG. 8.1: *Fonction coût au cours des itérations, pour l'optimisation avec la méthode de gradient incomplet avec contraintes $\mathcal{C}^0$.*

La figure FIG.8.2 présente l'évolution des formes de la B-spline du moyeu au cours des itérations. La forme initiale est la B-spline située au-dessus de toutes les autres. Les courbes changent ensuite du noir au bleu et enfin jusqu'au vert, à la convergence. La B-spline rouge correspond à la 16ème itération de l'algorithme et donc à la géométrie pour laquelle la fonction coût atteint la plus faible valeur que l'on ait pu obtenir.

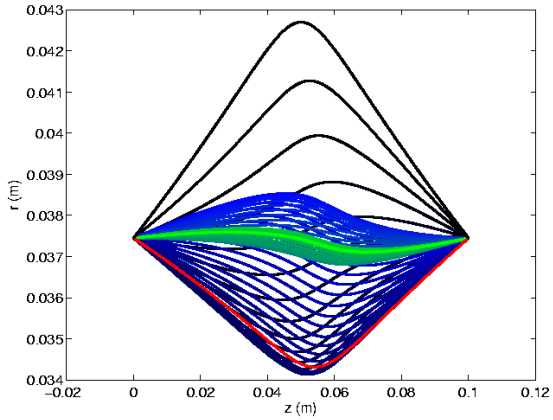

FIG. 8.2: *Evolution de la forme de la B-spline du moyeu au cours des itérations, pour l'optimisation avec la méthode de gradient incomplet avec contraintes $\mathcal{C}^0$.*

Le fluide s'écoulant entre le moyeu et le carter cylindrique, on aurait pu s'attendre à trouver pour forme optimale du moyeu un cylindre droit et donc une droite pour la B-spline. Or la perte de pression par unité de longueur dans une telle géométrie est de 4.98 Pa.m. On voit donc que l'on peut parvenir à une perte de pression encore plus faible (ici 3.29 Pa.m) pour un cylindre présentant un rétrécissement en son milieu.

On conclut de ces premières observations que le gradient incomplet n'est pas une bonne approximation du gradient de la fonction coût J dans ce cas. Les paragraphes 8.2 et au-delà vont avoir pour but d'essayer de déterminer pour quelle(s) raison(s).

Remarque 8.1.1. *En observant le comportement de la fonction coût au fil des itérations de la méthode de gradient incomplet, on a malgré tout une impression de convergence. Ceci est confirmé lorsque l'on trace sur la figure* FIG.*8.3 la courbe représentant la norme du gradient incomplet en fonction des itérations : la norme tend clairement vers* 0.

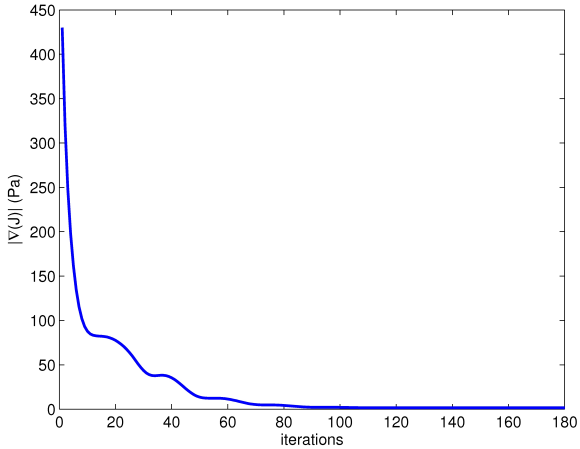

FIG. *8.3: Norme du gradient incomplet de la fonction coût au cours des itérations, pour l'optimisation avec la méthode de gradient incomplet avec contraintes* $\mathcal{C}^0$.

Il semble qu'il existe des points critiques pour le gradient incomplet. Une étude approfondie pourrait peut-être donner un sens physique à cette observation. Il est possible qu'avec le gradient incomplet tel qu'on l'a calculé, on résolve un autre problème d'optimisation que celui auquel on s'intéresse ici.

8.1.2 Contraintes sur le caractère $\mathcal{C}^1$

On étudie un problème similaire au précédent mais on introduit deux contraintes d'égalité linéaires supplémentaires : (7.21) et (7.22) qui assurent le caractère $\mathcal{C}^1$ à l'interface moyeu-extensions dans la direction r. On a donc quatre contraintes sur les coordonnées r_i des points de contrôle. Il faut donc un minimum de cinq points de contrôle, afin d'avoir au moins un degré de liberté dans la direction radiale. On retient donc un modèle avec une B-spline à cinq points de contrôle, le problème comportant ainsi, à nouveau, quatre degrés de liberté.
L'optimisation est menée de la même façon que précédemment. L'introduction de nouvelles contraintes permet de mieux maîtriser la géométrie et par conséquent on peut travailler avec un pas plus grand, ce qui explique la décroissance plus rapide de la fonction coût dans ce cas.
Malheureusement les contraintes que l'on a imposées ne suffisent pas à éliminer complètement les cas de génération de géométries aberrantes. Il est nécessaire qu'aux interfaces moyeu-extensions, la dérivée de la composante $z(t)$ de la B-spline soit positive (cf. §7.6.2), sinon on génère une géométrie avec des cavités, ce qui n'est évidemment pas bon pour l'écoulement et ce qui provoque l'arrêt prématuré de l'algorithme du point de vue numérique. Or cette nouvelle contrainte se traduit pas les deux inégalités (7.25) et (7.26) que l'on ne peut pas prendre en compte par réduction du problème d'optimisation.
Pour palier à ce problème, on choisit d'utiliser une *méthode de projection*. Soient $(z_i^k, r_i^k)_{i=1\ldots5}$ les points de contrôle de la B-spline à la k-ième itération de l'algorithme. Si à la $(k+1)$-ième itération les $(z_i^{k+1})_{i=1\ldots5}$ ne vérifient pas (7.25) et (7.26), alors on effectue la projection suivante :

$$\forall i \in \{1,\ldots,5\},\ z_i^{k+1} = z_i^k.$$

On préfère projeter sur les valeurs de l'itération précédente plutôt que sur la frontière du domaine d'admissibilité pour des raisons numériques. En effet, une géométrie trop à la limite risque de provoquer une mauvaise modélisation de l'écoulement dans le logiciel de CFD (notamment avec des angles trop marqués aux interfaces moyeu-extensions).

Sur la figure FIG.8.4, on observe la décroissance de la fonction coût au cours des itérations (sa valeur initiale, tronquée sur la figure, est de 12.14 Pa.m).

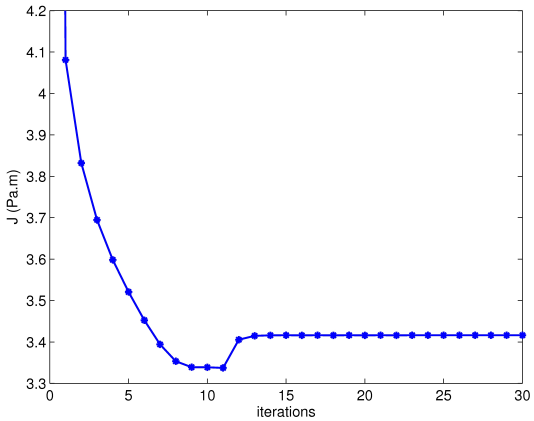

FIG. 8.4: *Fonction coût au cours des itérations, pour l'optimisation avec la méthode de gradient incomplet avec contraintes $\mathcal{C}^1$.*

On constate qu'à partir de la 12ème itération, la fonction coût croît légèrement. En observant l'évolution du vecteur des points de contrôle, on constate qu'à partir de la 12ème itération les coordonnées des points de contrôle dans la direction z sont systématiquement projetées sur celles de la 11ème itération. La croissance est sans doute liée au fait que les coordonnées des points de contrôle dans la direction r sont les seules à varier. On atteint ensuite la convergence.
Si la valeur minimale de J atteinte dans ce cas est de 3.34 Pa.m contre 3.29 Pa.m dans le cas 8.1.1, en revanche la convergence se fait autour d'une valeur plus faible (3.42 Pa.m contre 4.64 Pa.m).
Sur la figure FIG.8.5, on observe l'évolution de la forme des B-splines au cours des itérations (du noir vers le bleu, puis vers le vert, avec la géométrie initiale au-dessus des autres).

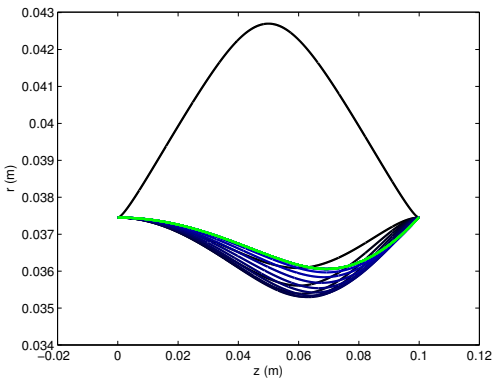

FIG. 8.5: *Evolution de la forme de la B-spline du moyeu au cours des itérations, pour l'optimisation avec la méthode de gradient incomplet avec contraintes $\mathcal{C}^1$.*

On obtient une allure comparable à l'allure finale de la B-spline dans le cas 8.1.1. Il y a moins de torsion dans la direction radiale car on n'a plus qu'un seul degré de liberté dans cette direction, contre deux dans le cas précédent.

8.1.3 Perspectives pour un nombre de degrés de liberté supérieur

On a testé sans grand succès des géométries avec un plus grand nombre de degrés de liberté. Lors des premières itérations on observe le même comportement que ce que l'on a déjà constaté : évolution des B-splines vers des formes situées sous la droite correspondant au cylindre, avec une tendance du "creux" à se trouver plus près de la sortie que de l'entrée. Cependant l'algorithme s'arrête systématiquement de façon prématurée car une géométrie inacceptable est générée tôt ou tard. Cela peut venir de différentes causes, comme le signe des dérivées de la composante $z(t)$ (déjà remarqué en 8.1.2) ou de trop grandes courbures par endroits. On a déjà évoqué en 7.6.2 les contraintes d'inégalité qu'il serait bon de mettre en place pour résoudre ces problèmes. Comme on l'a aussi dit, la réduction du problème ne peut prendre en compte que des contraintes d'égalité linéaires. Il faut donc avoir recours à d'autres méthodes, comme la projection (mise en œuvre ci-avant), la pénalisation (intérieure ou extérieure) ou le lagrangien augmenté. On considère un problème d'optimisation soumis à m contraintes d'égalité ou d'inégalité :

$$\begin{cases} \min\limits_{X\in\mathbb{R}^{2n}} J(X) \\ c_i(X) = 0, \quad i \in E, \\ c_i(X) \leq 0, \quad i \in I, \end{cases} \tag{8.1}$$

où E et I forment une partition de $\{1,...,m\}$.

Pénalisation

La pénalisation permet de transformer un problème d'optimisation avec contraintes en un problème, ou une suite de problèmes d'optimisation sans contrainte ([16], [51]). Le principe général est le suivant : on remplace le problème (8.1) par un (ou des) problème(s) sans contrainte de la forme

$$\min_{X\in\mathbb{R}^{2n}} J_\Pi(X) \tag{8.2}$$

où

$$J_\Pi(X) := J(X) + \alpha\Pi(X) \tag{8.3}$$

est la *fonction de pénalisation*, Π est la *fonction pénalisante* et α est le *facteur de pénalisation*. Le terme additionnel $\alpha\Pi(X)$ permet soit de pénaliser l'approche de la frontière du domaine admissible (c'est la *pénalisation intérieure*), soit de pénaliser la violation des contraintes (c'est la *pénalisation extérieure*). Une méthode de pénalisation intérieure classique est la pénalisation logarithmique : pour les contraintes d'inégalité seulement, la fonction de pénalisation s'écrit

$$J_\Pi(X) = J(X) - \alpha\sum_{i\in I}\log(-c_i(X)). \tag{8.4}$$

Dans une méthode de pénalisation extérieure, le terme additionnel $\alpha\Pi(X)$ est une expression, ou une somme d'expressions, qui, pour chaque contrainte, sont positives quand le point courant X viole la contrainte en question et nulles sinon. Ainsi $J_\Pi(X)$ ne diffère de $J(X)$ que quand X est *extérieur* au domaine admissible. Par ailleurs, plus le coefficient positif α sera grand, et plus les minima de $J_\Pi(X)$

auront tendance à respecter les contraintes.
Une fonction de pénalisation extérieure classique pour le problème (8.1) est :

$$J_{\Pi}(X) = J(X) + \alpha \left(\sum_{i \in E} (c_i(X))^2 + \sum_{i \in I} (\max(c_i(X), 0))^2 \right). \tag{8.5}$$

Remarque 8.1.2. *Il peut arriver que certaines contraintes soient d'un ordre plus intuitif et que l'on ne souhaite pas les exprimer sous la forme figée d'une égalité ou d'une inégalité. C'est le cas lorsque l'on souhaite contrôler la courbure de la B-spline génératrice.*
Dans ce genre de situation, l'objectif de la pénalisation n'est pas de tendre vers $\Pi(X) = 0$*, mais tout au moins de se servir du terme* $\alpha\Pi(X)$ *pour contrôler une grandeur que l'on ne souhaite pas voir croître démesurément, sans pour autant faire le choix de lui fixer une limite donnée. Cette nuance doit être traitée en procédant à un choix raisonné de* α*.*

A l'usage on a constate que le choix du coefficient α est problématique et c'est d'ailleurs une observation générale des praticiens de l'optimisation. Par exemple, dans le cas de la pénalisation de la courbure, il faut suffisamment pénaliser pour éviter toute géométrie aberrante, mais pas trop non plus, sous peine de converger vers une B-spline droite (c'est-à-dire un moyeu cylindrique).
Une solution envisagée pour déterminer α est de s'inspirer de la méthode des *L-curves*.

L-curves On introduit rapidement ici la méthode et on renvoie à [36] pour plus de détails. L'idée est de tracer une courbe paramétrée par α avec en abscisse la fonction coût J et en ordonnée la fonction pénalisante Π. Chaque point de la courbe correspond à une solution de l'optimisation obtenue pour le paramétre α courant. On obtient une courbe en L comme sur la figure FIG.8.6. La représentation en échelle logarithmique permet d'accentuer la forme en L.

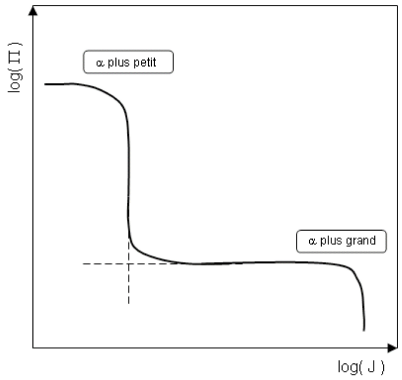

FIG. 8.6: *Forme générique de L-curve.*

Intuitivement le meilleur α que l'on puisse choisir se situe au coin du L. En effet, pour de faibles valeurs de α, on ne pénalise pas assez la géométrie, mais pour de grandes valeurs, on pénalise trop, au détriment de l'objectif. Le coin de la L-curve correspondrait donc au bon compromis.
Il faut déterminer une méthode pour ne pas avoir à calculer trop de points de la L-curve. On commence par un α grand (cela permet d'avoir une forte pénalisation et donc pas de risque d'arrêt prématuré de l'algorithme). On laisse converger, au moins partiellement, l'optimisation pour ce α. Puis on réitère l'opération pour des valeurs de α de plus en plus petites (par exemple en divisant par 2 d'un cas à l'autre). On observe en parallèle l'allure de la L-curve qui se dessine. Pour détecter le coin, on peut soit calculer

la pente (lorsqu'elle devient forte, on a atteint le coin), soit calculer la courbure (le coin est le lieu de plus forte courbure).
Les inconvénients de la méthode des L-curves sont que l'existence d'un coin, notion par ailleurs mal définie, n'est pas garantie et qu'il faut connaître de nombreux points de la courbe pour déterminer efficacement le coin.

Lagrangien augmenté

L'algorithme de lagrangien augmenté procède de façon similaire à la pénalisation car il permet de transformer un problème d'optimisation avec contraintes en un problème, ou une suite de problèmes d'optimisation sans contrainte.
Le lagrangien augmenté du problème (8.1) est la fonction $l_r : \mathbb{R}^{2n} \times \mathbb{R}^m \to \mathbb{R}$, définie pour $(X,\lambda) \in \mathbb{R}^{2n} \times \mathbb{R}^m$ et $r > 0$ par :

$$l_r(X,\lambda) = J(X) + \sum_{i\in E} \lambda_i c_i(X) + \frac{r}{2}\sum_{i\in E}(c_i(X))^2 + \sum_{i\in I}\lambda_i \max\left(-\frac{\lambda_i}{r}, c_i(X)\right) + \frac{r}{2}\sum_{i\in I}\left(\max\left(-\frac{\lambda_i}{r}, c_i(X)\right)\right)^2 \quad (8.6)$$

Le lagrangien augmenté réduit les risques de mauvais conditionnement des sous-problèmes que l'on peut rencontrer dans les méthodes de pénalisation, en introduisant au début de chaque itération k une estimation explicite λ^k du vecteur des multiplicateurs de Lagrange dans la fonction à minimiser. Pour un facteur de pénalisation r^k fixé, on minimise alors $l_{r^k}(X,\lambda^k)$ par rapport à X, obtenant ainsi X_k, et on met à jour λ^k et r^k. On renvoie à [51] pour plus de détails.

8.2 Comparaison avec les résultats des différences finies

D'après les résultats du paragraphe précédent, la méthode de gradient incomplet peut fournir alternativement une direction de montée et de descente. Pour s'assurer que cela n'est pas dû à un mauvais réglage du pas de la méthode de descente, on calcule, à chaque itération de l'algorithme avec gradient incomplet, la direction que donnerait une méthode de différences finies et l'angle entre cette direction et celle du gradient incomplet.
Par ailleurs, on mène une optimisation par méthode de gradient calculé par les différences finies de façon à comparer les résultats respectifs des deux méthodes.

8.2.1 Calcul du gradient par différences finies

Bien que la méthode des différences finies ne donne encore qu'une valeur approchée du gradient, cette approximation est généralement de bonne qualité, en particulier dans des situations simples comme l'optimisation du moyeu seul. On considère un problème à $2n$ inconnues, avec m contraintes d'égalité linéaires, que l'on réduit à un problème à $q = 2n - m$ inconnues, que l'on note $(\xi_i)_{i=1\ldots q}$. Alors les différences décentrées à droite s'écrivent :

$$\frac{\partial J}{\partial \xi_i}(\xi_1,\ldots,\xi_i,\ldots,\xi_q) \simeq \frac{J(\xi_1,\ldots,\xi_i+\varepsilon,\ldots,\xi_q) - J(\xi_1,\ldots,\xi_i,\ldots,\xi_q)}{\varepsilon}. \quad (8.7)$$

Le gradient de J est obtenu en appliquant q fois la formule. Cette méthode demande donc l'évaluation de J au point $(\xi_1,\ldots,\xi_q)$ et en chacun des q points perturbés $(\xi_1,\ldots,\xi_i+\varepsilon,\ldots,\xi_q)$, pour $i = 1,\ldots,q$, soit $(q+1)$ évaluations.

L'inconvénient majeur de cette méthode dans le cadre d'une optimisation est bien sûr qu'elle est très coûteuse en temps de calcul puisqu'elle nécessite $(q+1)$ résolutions de Navier-Stokes par itération de l'algorithme d'optimisation.

Remarque 8.2.1. *Les différences finies décentrées à droite (ou à gauche) génèrent une erreur en $O(\varepsilon)$. Les différences centrées :*

$$\frac{\partial J}{\partial \xi_i}(\xi_1,...,\xi_i,...,\xi_q) \simeq \frac{J(\xi_1,...,\xi_i+\varepsilon,...,\xi_q)-J(\xi_1,...,\xi_i-\varepsilon,...,\xi_q)}{2\varepsilon}$$

engendrent une erreur en $O(\varepsilon^2)$ et sont en ceci meilleures. Bien sûr leur inconvénient est de nécessiter $(2q+1)$ résolutions de Navier-Stokes par itération de l'algorithme d'optimisation.

Choix du paramètre ε et erreurs numériques

L'estimation de l'erreur pour les différences décentrées incite à choisir ε aussi petit que possible mais il faut aussi tenir compte des erreurs numériques. Sans même parler des erreurs d'arrondi, il se pose le problème de la stabilité du logiciel de CFD Fluent.
Le cas de l'optimisation de la pompe complète, par exemple, est problématique. En effet, en menant une même résolution de Navier-Stokes dans une même géométrie, maillée de plusieurs façons différentes, on évalue une erreur numérique. Celle-ci est du même ordre de grandeur que la variation de la fonction coût lors du calcul des différences finies (pour un ε raisonnablement petit). Malheureusement à chaque fois que l'on génère une nouvelle géométrie, et même si les instructions de maillage sont les mêmes, les maillages sont suffisamment différents pour considérer que la méthode est instable. Ceci est dû au fait que le maillage de la pompe complète est non structuré. Une solution serait de recourir à un module de déformation de maillage.
Dans le cas de l'optimisation du moyeu seul, la géométrie est suffisamment simple pour être maillée de façon structurée (figure FIG.7.9). L'erreur numérique est alors bien moindre et il est possible de choisir une valeur de ε assez petite et de ce fait satisfaisante.

8.2.2 Comparaison de l'angle entre gradient incomplet et gradient calculé par différences finies

On mène la comparaison sur le cas 8.1.1. A chaque itération de l'algorithme de gradient incomplet on évalue la fonction coût pour la géométrie courante et on exécute les $q=4$ évaluations de (8.7) dans le code de différences finies jumelé pour l'occasion. On calcule ainsi le gradient de la fonction coût par différences finies et on peut le comparer au gradient incomplet.
On trace sur la figure FIG.8.7 l'arccosinus β^* du produit scalaire des vecteurs unitaires donnant la direction du gradient incomplet et du gradient obtenu par différences finies. Ainsi $\beta^* \in [0,\pi]$ n'est pas à proprement parler l'angle géométrique entre les deux gradients mais il donne la même indication, à savoir si l'on a une direction de montée ou de descente. Il apparaît clairement qu'au-delà de la 16ème itération, la direction du gradient incomplet est une direction de montée ($\beta^* \geq \pi/2$). Puis, au-delà de la 36ème itération, on a à nouveau, pour un temps, une direction de descente. Cela correspond, dans l'évolution de la forme du moyeu, au moment où les courbes sont construites les unes en dessous des autres au fil des itérations. Toujours à ce même moment, la fonction coût recommence à décroître. Puis l'angle redevient supérieur à $\pi/2$, on a à nouveau une direction de montée, et ainsi de suite...

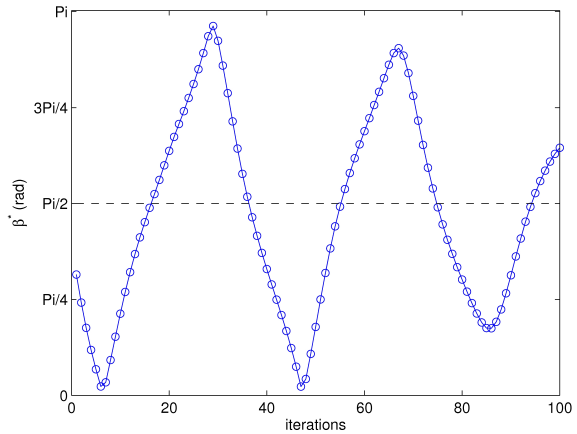

FIG. 8.7: *"Angle"* β^* *entre la direction du gradient incomplet et la direction du gradient calculé par différences finies, au cours des itérations.*

8.2.3 Résultats numériques par les différences finies

On mène une optimisation de forme du moyeu pour le problème simple présenté en 8.1.1 en calculant cette fois le gradient par différences finies. Le pas de la méthode de descente est le même que celui utilisé pour la méthode de gradient incomplet.
On confronte, sur la figure FIG.8.8, la convergence de la fonction coût J pour les deux méthodes. On constate que la minisation de J est plus efficace et plus rapide par les différences finies.

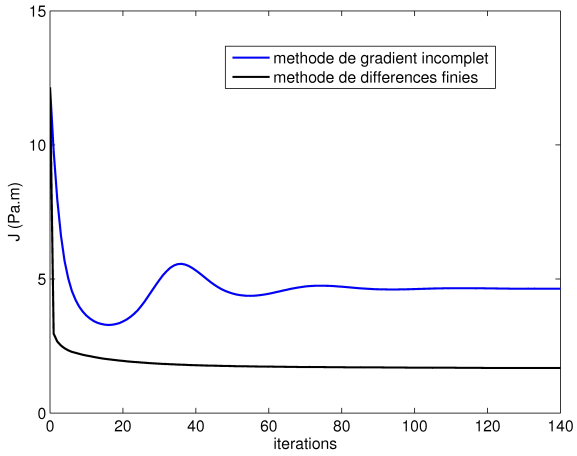

FIG. 8.8: *Fonction coût au cours des itérations.*

Sur la figure FIG.8.9, on représente l'évolution des formes de la B-spline du moyeu dans le cadre de la méthode des différences finies. Une fois encore, l'évolution se fait de la forme initiale noire, au-dessus de toutes les autres, vers les courbes bleues, puis vertes.

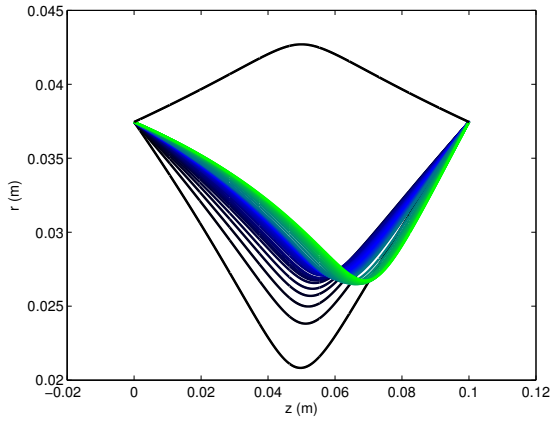

FIG. 8.9: *Evolution de la forme de la B-spline du moyeu au cours des itérations, pour l'optimisation avec la méthode de différences finies.*

Ainsi, que ce soit par le biais de la méthode de gradient incomplet ou par celui de la méthode des différences finies, on voit que le moyeu a tendance à devenir plus resserré que le cylindre. De plus la déformation est située plus près de la face de sortie que de la face d'entrée dans les deux cas. Cependant, sur la figure FIG.8.10 il apparaît que le moyeu obtenu grâce aux différences finies est nettement plus resserré que celui obtenu par la méthode de gradient incomplet.

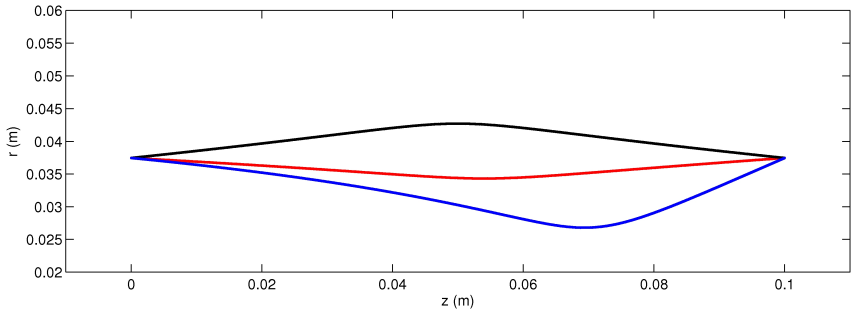

FIG. 8.10: *Comparaison de la meilleure forme obtenue avec la méthode de gradient incomplet (en rouge) et de la forme optimale selon la méthode de différences finies (en bleu). (La géométrie initiale est rappelée en noire.)*

8.2.4 Conclusion

Cette étude a permis de confirmer que la méthode de gradient incomplet ne donne pas toujours une direction de descente pour la minimisation de la perte de charge. Les différences finies donnent un meilleur résultat. Pourtant, dans la mesure où cette dernière méthode est particulièrement coûteuse, elle ne peut être utilisée que ponctuellement, comme ici, dans un but de comparaison puis de validation (ou d'invalidation) d'une autre méthode. Par ailleurs, les cas où on choisit de l'appliquer sont des cas où les inconnues sont peu nombreuses et, par conséquent, où le nombre d'évaluations de J par itération de l'algorithme est petit. De plus, ce sont des cas d'optimisation du moyeu seul puisque la géométrie est alors maillée avec un maillage structuré.

8.3 Qualité du gradient incomplet et cas du moyeu cylindrique

8.3.1 Gradient incomplet pour un moyeu cylindrique de rayon fixé

Présentation du problème

On s'intéresse au gradient incomplet de la fonction coût J lorsque la géométrie initiale est un cylindre de rayon fixé. Un cylindre est une surface de révolution générée par une droite. Encore une fois on considère le cas simple où la B-spline $(z(t), r(t))$ a quatre points de contrôle $(z_i, r_i)_{i=1\ldots4}$. Pour obtenir une droite, toutes les coordonnées $(r_i)_{i=1\ldots4}$ doivent être égales au rayon, que l'on note R, fixé pour le cylindre. On étudie encore une géométrie avec extensions, donc les coordonnées z_1 et z_4 sont imposées par les contraintes (7.17) et (7.19). En revanche, plusieurs couples (z_2, z_3) peuvent servir à générer la même droite. En effet, pour obtenir le cylindre, il suffit que $\forall t \in [0,1]$, $z(t) \in [0, L]$. Cependant, afin d'éviter que la droite ne soit dédoublée par endroits, on cherche une condition nécessaire et suffisante pour que $\forall t \in [0,1]$, $z'(t) \geq 0$. (On a alors nécessairement $\forall t \in [0,1]$, $z(t) \in [0, L]$ car (7.17) et (7.19) ne sont autres que $z(0) = 0$ et $z(1) = L$.)
On calcule :

$$z(t) = \frac{1}{6}\left(t^3(-z_1 + 3z_2 - 3z_3 + z_4) + t^2(3z_1 - 6z_2 + 3z_3) + t(-3z_1 + 3z_3) + (z_1 + 4z_2 + z_3)\right)$$

et

$$z'(t) = \frac{1}{2}\left(t^2(-z_1 + 3z_2 - 3z_3 + z_4) + t(2z_1 - 4z_2 + 2z_3) + (-z_1 + z_3)\right).$$

En prenant en compte les égalités (7.17) et (7.19), on obtient :

$$z'(t) = at^2 + bt + c \text{ avec } \begin{cases} a &= 3L + 3z_2 - 3z_3, \\ b &= -6z_2, \\ c &= 2z_2 + z_3. \end{cases}$$

Finalement $\forall t \in [0,1]$, $z'(t) \geq 0$ si et seulement si :

$$\begin{cases} & z'(0) \geq 0, \\ \text{et} & z'(1) \geq 0, \\ \text{et} & \left(\text{si } \dfrac{-b}{2a} \in]0,1[\text{ et } a > 0, \text{ alors } b^2 - 4ac \leq 0\right), \end{cases}$$

si et seulement si :

$$\begin{cases} & 2z_2 + z_3 \geq 0, \\ \text{et} & 3L \geq z_2 + 2z_3, \\ \text{et} & \left(\text{si } \dfrac{z_2}{L + z_2 - z_3} \in]0,1[\text{ et } L + z_2 - z_3 > 0, \text{ alors } z_2^2 + z_3^2 + z_2 z_3 - 2Lz_2 - Lz_3 \leq 0\right). \end{cases}$$

On désigne par E l'ensemble des couples (z_2, z_3) vérifiant ce dernier système d'inégalités.

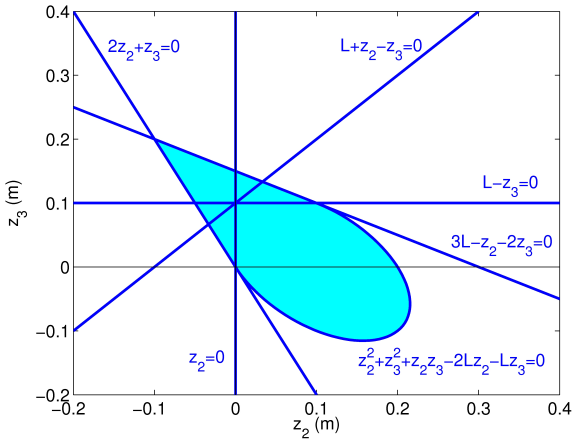

FIG. 8.11: *Domaine admissible E (zone colorée) pour* $L = 0.1$ *m.*

Ainsi, lorsque le couple (z_2, z_3) varie dans E et que les $(r_i)_{i=1...4}$ ne varient pas, la géométrie reste identique. Par suite $\dfrac{\partial J}{\partial z_2} = \dfrac{\partial J}{\partial z_3} = 0$ pour tout $(z_2, z_3) \in E$ et $\dfrac{\partial J}{\partial r_2}$ et $\dfrac{\partial J}{\partial r_3}$ donnent la direction du gradient incomplet pour le problème réduit, donc la direction de l'algorithme d'optimisation.

Vérification numérique

On teste trois couples (z_2, z_3) choisis dans le domaine E. Pour le cylindre, la valeur initiale de la fonction coût est $J_0 = 4.98$ Pa.m. On calcule dans chacun des trois cas le gradient incomplet réduit $\tilde{\nabla} J_0 = \left(\dfrac{\tilde{\partial} J_0}{\partial z_2}, \dfrac{\tilde{\partial} J_0}{\partial r_2}, \dfrac{\tilde{\partial} J_0}{\partial z_3}, \dfrac{\tilde{\partial} J_0}{\partial r_3} \right)$. Pour le même pas de descente que précédemment, on détermine la nouvelle géométrie à partir de $\tilde{\nabla} J_0$ et on calcule la nouvelle valeur de la fonction coût J_1, pour s'assurer que l'on a bien une direction de descente.

z_2 (m)	z_3 (m)	$\tilde{\nabla} J_0$	J_1 (Pa.m)
0.05	0.05	$\begin{pmatrix} 0.0 \\ 48.237 \\ 0.0 \\ 48{,}191 \end{pmatrix}$	4.69
0.15	-0.05	$\begin{pmatrix} 0.0 \\ 37.577 \\ 0.0 \\ 37.373 \end{pmatrix}$	4.82
-0.05	0.15	$\begin{pmatrix} 0.0 \\ 57.658 \\ 0.0 \\ 59.797 \end{pmatrix}$	4.47

TAB. 8.1: *Valeurs numériques pour une itération de l'algorithme de gradient incomplet avec pour forme initiale le cylindre.*

Les résultats sont conformes à ce que l'on attendait. Pour une forme cylindrique du moyeu, les composantes du gradient incomplet prennent des valeurs cohérentes et on a bien une direction de descente.

Plus précisément, on remarque que les dérivées incomplètes $\frac{\tilde{\partial} J_0}{\partial z_2}$ et $\frac{\tilde{\partial} J_0}{\partial z_3}$ sont exactement nulles, ce qui peut s'expliquer.
Le moyeu est cylindrique, par conséquent la troisième composante n_z de la normale est nulle. D'autre part, lorsque l'on ne fait varier que z_2 ou z_3, seule la troisième composante du champ de déformation $\mathcal{V}$ est non nulle. Par suite $\mathcal{V}.n = 0$ en tout point du moyeu.
Par ailleurs, on a dit que lorsque seul z_2 ou z_3 varie dans E, la géométrie est inchangée, donc $n' = 0$.
Or, si on consulte l'expression (4.10) du gradient incomplet, on constate que n' ou $\mathcal{V}.n$ est en facteur dans chacune des intégrales. Ceci explique que numériquement on trouve exactement la valeur 0.0 pour $\frac{\tilde{\partial} J_0}{\partial z_2}$ et $\frac{\tilde{\partial} J_0}{\partial z_3}$.

8.3.2 Gradient incomplet pour un moyeu cylindrique de rayon variable

Ici on étudie un problème différent des précédents. Le moyeu est cylindrique et son rayon R est variable, de même que celui des extensions qui sont considérées comme infinies. On considère une B-spline à quatre points de contrôle $(z_i, r_i)_{i=1\ldots4}$. Les contraintes appliquées à ce problème sont :

$$\begin{cases} \forall i \in \{1,\ldots,4\},\ r_i = R, \\ (z_2, z_3) \in E \text{ est fixé}, \\ (7.17), \\ (7.19). \end{cases}$$

On s'intéresse à l'écoulement d'un fluide incompressible entre deux cylindres infinis de rayons R et $R' > R$ et on néglige la turbulence. Si on note u_r, u_θ, u_z les coordonnées du vecteur vitesse dans le repère cylindrique, alors les équations de Navier-Stokes s'écrivent dans ce repère :

$$-\mu\left(\Delta u_r - \frac{u_r}{r^2}\right) + \rho\left(u_r\frac{\partial u_r}{\partial r} + u_z\frac{\partial u_r}{\partial z} - \frac{1}{r}u_\theta^2\right) + \frac{\partial p}{\partial r} = 0, \tag{8.8}$$

$$-\mu\left(\Delta u_\theta - \frac{u_\theta}{r^2}\right) + \rho\left(u_r\frac{\partial u_\theta}{\partial r} + u_z\frac{\partial u_\theta}{\partial z} + \frac{1}{r}u_r u_\theta\right) = 0, \tag{8.9}$$

$$-\mu\Delta u_z + \rho\left(u_r\frac{\partial u_z}{\partial z} + u_z\frac{\partial u_z}{\partial z}\right) + \frac{\partial p}{\partial z} = 0, \tag{8.10}$$

$$\frac{\partial}{\partial r}(ru_r) + \frac{\partial}{\partial z}(ru_z) = 0. \tag{8.11}$$

On ajoute la condition de non-glissement aux parois :

$$u_r(R) = u_\theta(R) = u_z(R) = 0, \tag{8.12}$$

$$u_r(R') = u_\theta(R') = u_z(R') = 0. \tag{8.13}$$

On cherche une solution particulière telle que $u_r = u_\theta = 0$. Le système (8.8)-(8.11) se réduit alors à :

$$\frac{\partial p}{\partial r} = 0, \tag{8.14}$$

$$-\mu\Delta u_z + \frac{\partial p}{\partial z}\left(= -\mu\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u_z}{\partial r}\right) + \frac{\partial p}{\partial z}\right) = 0, \tag{8.15}$$

$$\frac{\partial u_z}{\partial z} = 0. \tag{8.16}$$

Une solution explicite d'un tel système est :

$$\begin{cases} u_r = u_\theta = 0, \\ u_z = \dfrac{c_0 r^2}{4\mu} + c_1 log(r) + c_2, \\ p = c_0 z + c_3, \end{cases}$$

pour toutes constantes réelles c_0 et c_3, et avec c_1 et c_2 déterminées par (8.12) et (8.13).
Si on isole dans la géométrie (constituée de deux cylindres infinis) une section finie de longueur L entre une surface d'entrée E située à la cote 0 et une surface de sortie S située à la cote L, on peut calculer explicitement la fonction coût J dans la section :

$$J = \frac{\int_E p - \int_S p}{L} = \frac{1}{L}\left(c_3\pi(R'^2 - R^2) - (c_0 L + c_3)\pi(R'^2 - R^2)\right) = -c_0\pi(R'^2 - R^2)$$

d'où

$$\frac{\partial J}{\partial R} = 2c_0\pi R. \tag{8.17}$$

Si on considère l'écoulement dans la section de longueur L, pour la solution particulière où $u_r = u_\theta = 0$, on peut imposer les conditions aux limites suivantes :

$$u_z = \frac{c_0 r^2}{4\mu} + c_1 log(r) + c_2 \quad \text{sur} \quad E, \tag{8.18}$$

$$u_z = 0 \quad \text{sur} \quad \text{les bords latéraux}, \tag{8.19}$$

$$p = c_0 L + c_3 \quad \text{sur} \quad S. \tag{8.20}$$

On peut alors retrouver le résultat (8.17) en considérant le problème dérivé de (8.14) à (8.16) :

$$\frac{\partial p'}{\partial r} = 0, \tag{8.21}$$

$$-\mu\Delta u'_z + \frac{\partial p'}{\partial z} = 0, \tag{8.22}$$

$$\frac{\partial u'_z}{\partial z} = 0, \tag{8.23}$$

avec les conditions aux limites dérivées de (8.18) à (8.20) :

$$u'_z = u'_{z,0} - \frac{\partial}{\partial n}(u_z - u_{z,0})\mathcal{V}.n \quad \text{sur} \quad E, \tag{8.24}$$

$$u'_z = -\frac{\partial u_z}{\partial n}\mathcal{V}.n \quad \text{sur} \quad \text{les bords latéraux}, \tag{8.25}$$

$$p' = -\frac{\partial p}{\partial n}\mathcal{V}.n \quad \text{sur} \quad S, \tag{8.26}$$

où $u_{z,0}$ est la solution explicite obtenue sur E dans la géométrie initiale (cylindres de rayon R et R').
Si on note $u_{z,\varepsilon}$ la solution explicite obtenue par résolution du système (8.14) à (8.16) sur la géométrie déformée où $R_\varepsilon = R + \varepsilon$, alors :

$$u'_z = \lim_{\varepsilon \to 0} \frac{u_{z,\varepsilon} - u_{z,0}}{\varepsilon}.$$

Le calcul et un développement limité permettent de montrer que u'_z est de la forme $k_0 log(r) + k_1$ où k_0 et k_1 sont des constantes dépendantes de R. Il s'agit donc d'une fonction harmonique donc $\dfrac{\partial p'}{\partial z} = 0$. Par

suite p' est constante et comme sur S, $\mathcal{V}.n = 0$, la condition sur S conduit à $p' = 0$.
On reprend l'expression analytique (4.4) de la fonction coût J, tout en considérant que J est la perte de pression :

$$J = -\int_{\partial\Omega} pn_z + \int_D pn_z. \tag{8.27}$$

Le calcul de l'expression analytique exacte du gradient du second terme a déjà été effectué (cf. §4.2.2) et on a :

$$\left(\int_D pn_z\right)' = \int_D \left(p'n_z + pn_z' + n_z \nabla p.n\mathcal{V}.n + pn_z H\mathcal{V}.n\right). \tag{8.28}$$

Pour calculer le gradient du premier terme, on utilise le résultat du théorème 4.1.1. On définit le vecteur $P = (0,0,p)$. On peut alors écrire :

$$\int_{\partial\Omega} pn_z = \int_{\partial\Omega} P.n$$

et la formule (4.2) s'applique :

$$\left(\int_{\partial\Omega} pn_z\right)' = \left(\int_{\partial\Omega} P.n\right)' = \int_{\partial\Omega} \left(P'.n + div(P)\mathcal{V}.n\right) = \int_{\partial\Omega} \left(p'n_z + \frac{\partial p}{\partial z}\mathcal{V}.n\right). \tag{8.29}$$

On a donc finalement :

$$\frac{\partial J}{\partial R} = \frac{1}{L}\left(-\int_{\partial\Omega}\left(p'n_z + \frac{\partial p}{\partial z}\mathcal{V}.n\right) + \int_D \left(p'n_z + pn_z' + n_z\nabla p.n\mathcal{V}.n + pn_z H\mathcal{V}.n\right)\right). \tag{8.30}$$

Etant donné que $p' = 0$, que $n_z = 0$ et $n_z' = 0$ sur le moyeu et que $\mathcal{V}.n = 0$ et $n_z' = 0$ sur E et S, il vient :

$$\frac{\partial J}{\partial R} = -\frac{1}{L}\int_{\partial\Omega}\left(\frac{\partial p}{\partial z}\mathcal{V}.n\right) = -\frac{1}{L}\int_{\partial\Omega} c_0\mathcal{V}.n.$$

Or, dans le repère 2D, en tout point de D de paramètre $t \in [0,1]$, on a $\mathcal{V} = (V_z(t), V_r(t))$ avec, pour ce problème particulier, $V_z(t) = 0$ et $V_r(t) = \sum_{i=0}^{3} p_i(t) = 1$, et $n = (0,-1)$, d'où :

$$\frac{\partial J}{\partial R} = -\frac{c_0}{L}\int_{\partial\Omega}(-1) = 2c_0\pi R. \tag{8.31}$$

On retrouve bien ainsi le résultat précédent.

8.4 Bien-fondé de la méthode

8.4.1 Justification par l'expérience

On a vu en introduction de ce mémoire que l'approche de gradient incomplet a déjà été largement étudiée, employée, et validée dans certains cas d'après [49].
On rappelle les notations : x_C désigne les variables de contrôle de l'optimisation, $W(x_C)$ la solution des équations d'état $E(x_C, W(x_C)) = 0$ et x_G les variables géométriques. Le gradient de la fonction coût peut être écrit ainsi :

$$\frac{dJ}{dx_C} = \frac{\partial J}{\partial x_C} + \frac{\partial J}{\partial x_G}\frac{\partial x_G}{\partial x_C} + \frac{\partial J}{\partial W}\frac{\partial W}{\partial x_G}\frac{\partial x_G}{\partial x_C}.$$

Après étude des différents termes de l'expression, et sans justification théorique, il a été observé que pour une fonction coût vérifiant certaines assertions on pouvait approcher le gradient par :

$$\frac{dJ}{dx_C} \simeq \frac{\partial J}{\partial x_C} + \frac{\partial J}{\partial x_G}\frac{\partial x_G}{\partial x_C},$$

la variation de J par rapport à la solution W étant négligeable quand x_C varie peu. Les assertions que doit vérifier la fonction coût sont :

a. J est une intégrale sur les frontières à optimiser D du domaine Ω de la forme $\int_D \varphi(x_C, x_G)\psi(W)$,
b. la courbure locale de D n'est "pas trop grande",
c. φ et ψ sont telles que l'on puisse vérifier formellement $\frac{1}{|\varphi|}\left|\frac{\partial\varphi}{\partial n}\right| >> \frac{1}{|\psi|}\left|\frac{\partial\psi}{\partial W}\right|$, où n est la normale à la paroi et avec $\left|\frac{\partial\varphi}{\partial n}\right| = O(1)$.

Assertion a

Dans le problème que l'on considère ici, cette assertion n'est pas tout à fait vérifiée puisque l'on a transformé la fonction coût en la différence d'une intégrale sur le bord du domaine *entier* et d'une intégrale sur les bords déformés D (cf. §4.2.1) :

$$J = \int_{\partial\Omega} p\frac{n_z}{L} - \int_D p\frac{n_z}{L}.$$

Assertion b

La courbure locale de D n'est pas trop grande, tout du moins dans le cas de l'étude du moyeu, et lorsque la géométrie n'est pas encore trop déformée par le processus d'optimisation.

Assertion c

Cette assertion est la plus difficile à interpréter ici. Etant donné que $\varphi(x_C, x_G) = \frac{n_z}{L}$, on a $\frac{1}{|\varphi|}\left|\frac{\partial\varphi}{\partial n}\right|$ qui est de l'ordre de l'unité, et comme $\psi(W) = p$, on a $\frac{1}{|\psi|}\left|\frac{\partial\psi}{\partial W}\right|$ qui est de l'ordre de $\frac{1}{|p|}$, c'est-à-dire de l'ordre de 10^{-3}. Par suite $\frac{1}{|\varphi|}\left|\frac{\partial\varphi}{\partial n}\right| >> \frac{1}{|\psi|}\left|\frac{\partial\psi}{\partial W}\right|$ est vérifiée. En revanche $\left|\frac{\partial\varphi}{\partial n}\right|$ est de l'ordre de 10^4, donc pas vraiment en $O(1)$.

8.4.2 Etude des dérivées éliminées lors du calcul du gradient incomplet

On revient ici sur le calcul du gradient incomplet de la fonction coût J, effectué dans le chapitre 4, et plus spécifiquement sur les termes que l'on a éliminés et qui contiennent les dérivées par rapport à la déformation du domaine. On peut répartir ces termes en trois catégories.

Termes effectivement nuls

On considère l'intégrale sur le bord du domaine $J_1 = \int_{\partial\Omega} pn_z$. On rappelle que l'on a introduit en 4.2.1 les équations de Navier-Stokes de sorte que :

$$J_1 = \mu\int_{\partial\Omega}\frac{\partial u_z}{\partial n} - \rho\int_{\partial\Omega} u_z U.n.$$

On s'intéresse dans un premier temps au terme $J_4 = \int_{\partial\Omega} u_z U.n$ dont on a calculé la dérivée (4.7) :

$$J_4' = \left(\int_{\partial\Omega} u_z U.n\right)' = \left(\int_{E\cup S} u_z U.n\right)' + \left(\int_C u_z U.n\right)' + \left(\int_D u_z U.n\right)'.$$

Chacun des trois termes de J_4' peut être transformé en une somme d'intégrales de la forme :

$$\left(\int u_z U.n\right)' = \int (u_z U)'.\mathcal{N} + \int (u_z U).\mathcal{N}' + \int \nabla(u_z U.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} + \int u_z U.\mathcal{N}H\mathcal{V}.\mathcal{N},$$

et c'est au terme $\int (u_z U)'.n$, le seul que l'on ait dû omettre, que l'on s'intéresse.
On étudie ici le cas du problème simplifié de l'optimisation du moyeu seul. On a imposé une condition de non glissement sur le carter et sur le moyeu : $U = 0$ sur C et D. Etant donné que $(u_z U)' = u_z' U + u_z U'$, la condition de non-glissement implique que $\int_C (u_z U)'.\mathcal{N} = \int_D (u_z U)'.\mathcal{N} = 0$.

Termes pour lesquels on ne se prononce pas

Dans l'expression que l'on vient d'étudier, on ne peut rien dire du terme $\int_{E\cup S} (u_z U)'.\mathcal{N}$ car on n'a pas de conditions aux bords favorables.

On considère à présent l'intégrale $J_2 = \int_D p n_z$. Lorsqu'en 4.2.2 on a développé cette expression, il a fallu éliminer le terme $\int_D p' \mathcal{N}_z$. On n'est pas capable d'apprécier l'importance de ce terme non plus.

Termes pour lesquels on est limité numériquement

On revient à :

$$J_1 = \mu \int_{\partial\Omega} \frac{\partial u_z}{\partial n} - \rho \int_{\partial\Omega} u_z U.n$$

et plus particulièrement à $J_3 = \int_{\partial\Omega} \frac{\partial u_z}{\partial n}$ dont on a calculé la dérivée (4.5) :

$$J_3' = \left(\int_{\partial\Omega} \frac{\partial u_z}{\partial n}\right)' = \left(\int_{\partial\Omega} \nabla u_z.n\right)' = \left(\int \nabla u_z.n\right)' + \left(\int_C \nabla u_z.n\right)' + \left(\int_D \nabla u_z.n\right)'.$$

Chacun des trois termes de J_3' peut être transformé en une somme d'intégrales de la forme :

$$\left(\int \nabla u_z.n\right)' = \int (\nabla u_z)'.\mathcal{N} + \int \nabla u_z.\mathcal{N}' + \int \nabla(\nabla u_z.\mathcal{N}).\mathcal{N}\mathcal{V}.\mathcal{N} + \int \nabla u_z.\mathcal{N}H\mathcal{V}.\mathcal{N}.$$

On s'intéresse uniquement aux termes de la forme $\int (\nabla u_z)'.\mathcal{N}$, c'est-à-dire les termes que l'on a omis pour pouvoir poursuivre le calcul du gradient incomplet.
Encore une fois on se place ici dans le cas du problème simplifié de l'optimisation du moyeu seul. La condition de non glissement est $U = 0$ sur C et D. On a alors la formule exacte de la dérivée des composantes u_i pour $i = 1, ..., 3$ de la vitesse U :

$$u_i' = -\frac{\partial u_i}{\partial n} \mathcal{V}.n \text{ sur } C \text{ et } D. \tag{8.32}$$

En principe on peut aussi calculer les termes de la forme $(\nabla u_i)'$ mais cela ferait apparaître des dérivées secondes de la vitesse, et on a dit que l'on ne prenait pas en compte de telles dérivées car on n'est pas parvenu à mettre en place un schéma numérique suffisamment stable pour les calculer. On ne peut donc pas se prononcer sur la légitimité de la suppression de ces termes.

8.5 Perspectives

8.5.1 Optimisation des aubes

On a essayé de réaliser une optimisation des aubes de la pompe de fond de puits. On a alors rencontré diverses difficultés numériques, que l'on recense ici, et à cause desquelles on n'a pas pu mener à terme l'optimisation. De possibles solutions sont proposées en 8.5.2.

Instabilité de Fluent vis-à-vis du maillage

Le maillage de la géométrie avec aubes n'est pas structuré. Comme on l'a déjà dit lorque l'on a étudié les différences finies (cf. §8.2), Fluent n'est pas stable vis-à-vis du remaillage. Ainsi, lorsqu'à chaque itération on remaille la géométrie, on introduit des erreurs numériques non négligeables.
Pour l'optimisation du moyeu seul on a contourné la difficulté puisqu'il est possible de réaliser un maillage structuré de la géométrie et alors un remaillage pour une faible variation de la géométrie équivaut à une déformation du maillage.
Par ailleurs l'automatisation du remaillage des aubes est impossible, empêchée par un certain nombre de *bugs* dans le mailleur Gambit. Une intervention humaine est donc régulièrement nécessaire pour rectifier un maillage raté et relancer l'optimisation.
Enfin, dans le cas de l'optimisation couplée du moyeu et des aubes, le temps de calcul pour une seule évaluation de la fonction coût et de son gradient devient très important (jusqu'à plusieurs heures) du fait de la grande taille du maillage (des centaines de milliers de mailles, soit dix fois plus de mailles que pour le cas simple du moyeu).

Implémentation des contraintes

On rencontre le même type de difficultés concernant l'implémentation des contraintes non-linéaires que dans le cas du moyeu seul. On essaie pour commencer de mener une optimisation sans contrainte des aubes. Malheureusement les géométries générées deviennent rapidement aberrantes et l'algorithme s'arrête prématurément après deux ou trois itérations seulement. Cependant, avant que cela ne se produise on constate une chose intéressante : l'algorithme a affiné les aubes. On peut interpréter ce comportement de la façon suivante : les aubes représentent un obstacle pour l'écoulement du fluide, bien qu'elles soient indispensables pour le canaliser. Cependant, un trop grand amincissement n'est acceptable ni techniquement (des aubes trop fines cassent), ni numériquement. C'est déjà avec cette idée que l'on avait proposé d'imposer une contrainte sur l'épaisseur des aubes en introduction de ce mémoire. On se trouve devant une double difficulté : traduire mathématiquement cette contrainte et ensuite l'implémenter. Il est probable que l'on obtiendra une (ou des) contrainte(s) d'inégalité non-linéaire(s). D'ailleurs les autres contraintes que l'on souhaite imposer aux aubes sont de ce même type. On a donc le même problème que pour l'optimisation du moyeu seul : il faut utiliser une méthode d'optimisation avec contraintes.

8.5.2 Solutions envisageables pour l'amélioration de l'algorithme d'optimisation

Déformation du maillage

Pour éviter les difficultés liées au remaillage on peut développer un module de déformation de maillage. Un premier travail en 2D a été réalisé en ce sens par F. Bayoro au cours de son stage à l'IFP [4]. Le recours à un module de déformation du maillage limite les problèmes d'instabilité de Fluent et les risques d'arrêt du code. Cependant, pour éviter d'avoir un maillage trop déformé et par conséquent de mauvaise qualité au bout d'un certain nombre d'itérations, il faut procéder à un remaillage à intervalle régulier pendant l'optimisation.

Optimisation avec contraintes et méthode SQP

Il est nécessaire dans un premier temps d'implémenter toutes les contraintes sur les aubes et le moyeu. On a présenté les méthodes de pénalisation ou de Lagrangien augmenté pour prendre en compte ces contraintes non-linéaires.
On a aussi pensé utiliser un code propre à l'IFP, *SQPAL*, développé pour des problèmes d'optimisation soumis à des contraintes non-linéaires [22]. Ce code repose sur une méthode de *programmation quadratique successive* (SQP : Sequential Quadratic Programming) qui consiste à transformer la résolution d'un problème d'optimisation non-linéaire (P) en la résolution d'une suite de problèmes d'optimisation quadratiques. Il suffit de fournir au code la fonction coût de (P), le gradient de celle-ci et les contraintes. La méthode SQP s'apparente à une méthode de Newton : on cherche à générer une suite $(x_k)_{k\geq 0}$ qui converge vers la solution x^* du problème d'optimisation (P). Cette suite est obtenue récursivement par $x_{k+1} = x_k + d_k$ où la direction d_k est solution d'un problème quadratique tangent (PQT). On ne rentre pas dans le détail de l'obtention du PQT (on consultera pour cela [51]) mais elle nécessite notamment la linéarisation des contraintes de (P). La fonction objectif du PQT est hybride, faisant intervenir le gradient de la fonction coût de (P) pour la partie linéaire et la Hessienne du Lagrangien associé à (P) pour la partie quadratique. La Hessienne du Lagrangien peut être déterminée grâce à la formule BFGS.

Réduction du temps de calcul

Dans une itération de l'algorithme d'optimisation pour le problème de la pompe complète, l'exécution de Fluent pour déterminer la fonction coût et son gradient incomplet prend 99% du temps, c'est donc sur cette étape que l'on doit travailler.
On peut envisager d'employer une méthode d'*optimisation emboîtée*. Pour cela, dans les premières itérations de l'optimisation on peut ne résoudre que partiellement les équations d'état puis de plus en plus précisément. On peut aussi utiliser des maillages emboîtés. Dans les premières itérations de l'algorithme d'optimisation on utilise un maillage grossier puis deux ou trois maillages de plus en plus fins au fur et à mesure que l'on s'approche de la convergence de l'optimisation. Dans les deux cas la question est de trouver la limite du point de vue numérique entre l'acceptable et l'inacceptable dans la grossièreté du modèle.

Conclusion

Lorsqu'un puits de pétrole arrive en fin d'exploitation et que la pression en surface est devenue trop faible pour que les pompes classiques permettent d'extraire ce qu'il reste dans le réservoir, il est possible d'utiliser des pompes dites "de fond de puits". De telles pompes sont introduites à même la réserve fossile et leur but est d'augmenter, entre leur entrée et leur sortie, la pression du fluide aspiré. L'objectif de l'optimisation de forme d'une pompe de fond de puits est donc naturellement d'y maximiser le gain de pression par unité de longueur, en redessinant les aubes et le moyeu, tout en respectant un certain nombre de contraintes géométriques.

Après avoir étudié les méthodes généralement mises en œuvre dans le domaine de l'optimisation de forme, on a retenu la méthode qui semblait la mieux adaptée aux contraintes numériques et aux caractéristiques du problème considéré. Dans la mesure où l'on souhaitait notamment utiliser le logiciel de CFD Fluent, certains choix étaient imposés. Ainsi il a été décidé que l'on emploierait une méthode déterministe, moins coûteuse en évaluations de la fonction coût du problème qu'une méthode non-déterministe. Une méthode déterministe nécessite de connaître le gradient de la fonction coût, ne serait-ce que de façon approchée. Compte tenu des limitations numériques qu'impose l'emploi d'un code "boîte noire", on a procédé à un calcul de gradient incomplet. Dans ce mémoire on a présenté ce calcul, ainsi que les résultats de l'optimisation obtenus dans une configuration géométrique simple, permettant une analyse critique de la méthode.

Dans un premier temps on a totalement repensé le mode de représentation, un peu daté, de la géométrie de la pompe. On a choisi d'utiliser des B-splines paramétriques cubiques dont les points de contrôle ont dès lors été considérés comme les inconnues du problème d'optimisation. En simplifiant légèrement la géométrie à modéliser en trois dimensions, on a pu en reconstituer une approximation grâce à des courbes paramétrées en deux dimensions seulement. Par ailleurs, les contraintes géométriques à prendre en compte dans le processus d'optimisation peuvent être exprimées sous forme d'égalités ou d'inégalités, linéaires ou non-linéaires portant sur les points de contrôle. Enfin on a cherché à déterminer un nombre raisonnable de points de contrôle de B-spline pour approcher la géométrie initiale de la pompe de fond de puits. On disposait, pour chaque courbe à paramétrer, d'un échantillonnage de points. Pour plusieurs nombres de points de contrôle on a mené une optimisation sur les coordonnées des points. Sur le plan numérique, l'utilisation de fonctions d'optimisation avec contraintes dans Matlab a permis de prendre en compte les contraintes géométriques imposées aux courbes. Au terme des différentes optimisations on a pu choisir le nombre de points minimal permettant de ne pas dépasser une erreur d'approximation donnée.

On a en outre préparé avec soin la modélisation de l'écoulement dans le logiciel de CFD Fluent, respectant en ceci la règle habituellement reconnue dans le cadre des méthodes de gradient incomplet et selon laquelle il vaut mieux calculer une solution précise aux équations d'état et effectuer une évaluation incomplète du gradient, plutôt que de calculer un gradient exact basé sur un état peu précis. Ici on étudie l'écoulement monophasique d'un fluide de type gasoil, on résout les équations de Navier-Stokes avec

modèle de turbulence $k-\varepsilon$ realizable et on modélise la rotation des parties mobiles de la pompe grâce à la technique dite de *frozen rotor*.

Toujours dans le cadre de la modélisation du problème physique, on s'est intéressé à une particularité de la pompe : le caractère périodique de sa géométrie, conféré par la disposition en série de ses étages identiques. Il a été observé expérimentalement qu'après passage dans plusieurs étages, le comportement du fluide tend à devenir périodique. On a donc étudié le comportement asymptotique de la solution d'un problème semi-linéaire et d'un problème de Stokes dans un domaine périodique constitué de cellules identiques, lorsque le nombre de ces cellules augmente. On a alors prouvé dans les deux cas la convergence exponentielle vers la solution du même problème, posé sur une seule cellule, avec conditions périodiques aux bords. Le résultat pour Stokes n'a été montré qu'en deux dimensions et avec des conditions aux bords de Navier. Il serait intéressant de parvenir à généraliser ce résultat, notamment en trois dimensions, puis pour d'autres conditions aux bords et enfin pour les équations de Navier-Stokes. La simulation numérique, menée sur un modèle de pompe à six étages, laisse penser que le même type de résultat sera valable dans ce dernier cas.

Une fois le cadre de la modélisation de la géométrie et de l'écoulement bien défini, on a mené le calcul d'un gradient incomplet de la fonction coût. Pour cela on a d'abord écrit l'expression analytique exacte du gradient, ceci à l'aide d'une formule d'intégration sur les bords variables. Le principe de la méthode de gradient incomplet telle qu'on l'a appliquée est de ne pas prendre en compte les termes que l'on ne sait pas calculer ou déterminer numériquement. Ces termes sont généralement les dérivées des solutions de l'équation d'état par rapport au déplacement d'un point de contrôle. Avant de procéder à ces simplifications, on a voulu ré-introduire de nouvelles données en utilisant les équations de Navier-Stokes. Ainsi, dans une expression de gradient ne contenant que la pression comme solution des équations d'état, on a fait apparaître la vitesse. Au terme des calculs on a obtenu une formule incomplète dans laquelle un certain nombre de termes peuvent être obtenus par le biais de Fluent, le reste des termes, de nature géométrique, pouvant être calculé exactement.
Parmi les termes d'ordre géométrique, on a calculé les expressions générales du champ de déformation induit par le déplacement d'un point de contrôle, de la courbure moyenne, ainsi que de la dérivée par rapport au domaine de la normale. Par la suite on a explicité ces expressions au cas par cas pour les différentes surfaces déformées au cours de l'optimisation.
Pour ce qui est des termes accessibles grâce à Fluent, on a distingué deux cas. La pression et la vitesse ont été extraites de Fluent à l'issu de la simulation de l'écoulement. En revanche, suite à une étude comparative menée entre les résultats rendus par des schémas numériques propres et ceux rendus par les schémas de Fluent, on a préféré utiliser un schéma numérique développé dans le cadre de ce travail pour calculer les dérivées des variables d'état.
Dans la mesure où tous les termes de la formule incomplète étaient alors déterminés, on a pu programmer un code de calcul du gradient incomplet interfacé avec Fluent, qu'il ne restait alors qu'à intégrer à un code d'optimisation.

Pour l'optimisation de la pompe de fond de puits on a choisi de développer d'abord un code d'optimisation pour le moyeu de la pompe, puis un autre pour les aubes, avec l'objectif final de jumeler les deux codes.
Plusieurs méthodes d'optimisation sans contrainte ont été étudiées. Les méthodes de Newton et de quasi-Newton ont été écartées car elles demandent de calculer exactement ou de façon approchée l'inverse de la Hessienne, ce qui est impossible dans le premier cas et délicat dans le second lorsque le gradient n'est lui-même pas exact. On a donc retenu une méthode de gradient classique, à pas fixe.

Par la suite on s'est intéressé plus particulièrement au cas de l'optimisation du moyeu seul. On a considéré le problème équivalent à maximiser le gain de pression par unité de longueur consistant à minimiser la perte de pression par unité de longueur. Cette étude devait servir de modèle et permettre de donner une première appréciation sur la méthode d'optimisation avec gradient incomplet. On a traduit une partie des contraintes géométriques de ce problème simple par des égalités linéaires qu'il a été possible de prendre en compte dans le cadre d'une optimisation sans contrainte en réduisant le problème. Les autres contraintes géométriques pouvaient être exprimées sous forme d'inégalités linéaires ou non-linéaires et on a choisi de ne pas les prendre en compte pour commencer.
En étudiant un problème simple, avec un petit nombre d'inconnues, on a constaté que la fonction coût ne convergeait pas vers un minimum, bien que l'on ait réussi à la faire décroître dans les premières itérations de l'algorithme d'optimisation. Pour des problèmes avec davantage d'inconnues, on a rapidement été confronté à des difficultés numériques. Du fait de l'absence de formulation de certaines contraintes d'inégalité indispensables, les formes générées peuvent vite devenir aberrantes et provoquer l'arrêt du code d'optimisation.

Enfin on a mis en place quelques tests de façon à analyser les mauvais résultats de l'algorithme d'optimisation avec gradient incomplet et leur(s) cause(s) éventuelle(s).
Dans un cas simple, avec peu d'inconnues, il était facile et relativement peu coûteux de calculer le gradient par différences finies. Ce premier test a confirmé que le gradient incomplet n'est pas systématiquement une direction de descente au cours des itérations. Evidemment, si les différences finies donnent de bons résultats, elles ne peuvent être utilisées dans le cadre d'une optimisation telle que celle de la pompe car elles sont bien trop coûteuses. Elles n'ont donc valeur que d'outil de validation.
On a aussi mené deux tests pour le cas particulier du moyeu cylindrique et leur conclusion a été favorable au gradient incomplet.
On a fini avec quelques réflexions sur la légitimité de l'emploi d'un gradient incomplet, notamment au vu de l'expérience acquise dans d'autres travaux et en essayant de juger de l'importance des termes qui n'ont pas été pris en compte dans l'expression analytique du gradient. On n'a pas vraiment su conclure sur ce point, manquant de matériel théorique pour ce faire.

Pour prolonger ce travail, on peut proposer quelques perspectives.

- Il serait intéressant de comprendre le comportement du gradient incomplet. En effet, si on constate que l'on ne minimise pas la fonction coût qui a été définie et que l'on n'a pas une direction de descente, on voit pourtant que la norme du gradient tend vers zéro. Quel en est le sens physique ? Résout-on un autre problème d'optimisation que celui que l'on souhaite considérer ?

- Il faudrait traduire mathématiquement l'ensemble des contraintes portant sur la géométrie de la pompe. Certaines ne sont pas faciles à exprimer, comme la nécessité d'une borne inférieure à l'épaisseur des aubes. Une fois ces contraintes clairement établies et programmées, on pourra envisager d'employer un code d'optimisation avec contraintes. Si la programmation de méthodes telles que la pénalisation se révèle très délicate, on peut choisir d'utiliser des codes existants et ayant déjà fait leurs preuves.

- Du point de vue numérique il est important de pouvoir automatiser complètement le code d'optimisation, en particulier l'étape problématique du maillage. Cet objectif est atteint pour l'optimisation du moyeu seul car on considère un maillage structuré, mais il ne sera pas facilement atteint pour des maillages non structurés. Un module de déformation du maillage est indispensable mais ne règlera pas nécessairement tous les problèmes puisqu'un remaillage ponctuel restera nécessaire.

- Enfin il faut parvenir à réduire les temps de calcul. Pour cela on peut essayer d'utiliser des méthodes consistant à ne pas systématiquement mener à convergence les simulations dans Fluent ou à employer un maillage grossier que l'on affine au fil des itérations. On peut aussi tenter de définir un problème grossier de substitution (en anglais : *surrogate model*) et de l'utiliser dans le cadre de méthodes de type *space mapping*.

Bibliographie

[1] G. Allaire, *Conception optimale de structures*, Mathématiques et Applications, volume 58, Springer, Berlin/Heidelberg/New York, 2007.

[2] S. Baillet, J. Brac, A. Henrot, *Première optimisation d'un étage de pompe mince de fond de puits*, Rapport interne 57 846, Institut Français du Pétrole, 2004.

[3] S. Baillet, A. Henrot, T. Takahashi, *Convergence results for a semilinear problem and for a Stokes problem in a periodic geometry*, Asymptotic Analysis, 50(3-4) : 325-337, 2006.

[4] F. Bayoro, J. Brac, J. Devauchelle, *Optimisation de formes et aspects de remaillage, exemple 2D d'un col de tuyère*, Rapport interne 58 951, Institut Français du Pétrole, 2005.

[5] F. Bélahcène, J.-A. Désidéri, *Paramétrisation de Bézier adaptative pour l'optimisation de forme en aérodynamique*, Rapport INRIA, n°4943, 2003.

[6] J. Brac, X. Gaillard, *Etude d'une pompe multi-étagée hélico-axiale miroir*, Rapport interne 57 179, Institut Français du Pétrole, 2003.

[7] J. Brac, P. Tardif d'Hammonville, *Etude d'une pompe multi-étagée hélico-axiale*, Rapport interne 57 339, Institut Français du Pétrole, 2003.

[8] H. Brézis, *Analyse fonctionnelle. Théorie et applications*, Dunod, Paris, 1999.

[9] F. Brezzi, M. Fortin, *Mixed and hybrid finite element methods*, volume 15 de Springer Series in Computational Mathematics, Springer-Verlag, New York, 1991.

[10] B. Cassagne, *Introduction au langage C*, Grenoble, 1998.

[11] I. Charpentier, Y. Maday, *Deux méthodes de décomposition de domaine pour la résolution d'équations aux dérivées partielles avec conditions de périodicité : application à la théorie de l'homogénéisation*, C. R. Acad. Sci. Paris, t. 321, série I, 359-366, 1995.

[12] M. Chipot, *ℓ goes to plus infinity*, Birkhäuser Advanced Texts, Birkhäuser, Basel/Boston, Massachusetts/Berlin, 2002.

[13] M. Chipot, Y. Xie, *On the asymptotic behaviour of elliptic problems with periodic data*, C. R. Math. Acad. Sci. Paris, 339 : 477-482, 2004.

[14] M. Chipot, Y. Xie, *Asymptotic behavior of nonlinear parabolic problems with periodic data*, Elliptic and parabolic problems, volume 63 de Progr. Nonlinear Differential Equations Appl., 147-156, Birkhäuser, Basel, 2005.

[15] M. Chipot, Y. Xie, *Elliptic problems with periodic data : an asymptotic analysis*, J. Math. Pures Appl. (9), 85(3) : 345-370, 2006.

[16] P.G. Ciarlet, *Introduction à l'analyse numérique matricielle et à l'optimisation*, Dunod, Paris, 1998.

[17] E. Cohen, R.F. Riesenfeld, G. Elber, *Geometric Modeling with Splines : an introduction*, A K Peters Ltd., Natick, Massachusetts, 2001.

[18] F. Coron, *Derivation of slip boundary conditions for the Navier-Stokes system from the Boltzmann equation*, J. Statist. Phys., 54 : 829-857, 1989.

[19] J.-M. Coron, *On the controllability of the* 2*-D incompressible Navier-Stokes equations with the Navier slip boundary conditions*, ESAIM Contrôle Optim. Calc. Var., 1 : 35-75 (electronic), 1995/96.

[20] A. Dadone, B. Mohammadi, N. Petruzzelli, *Incomplete sensitivities and BFGS methods for 3D aerodynamic shape design*, Rapport INRIA, n°3633, 1999.

[21] A. Dadone, B. Mohammadi, N. Petruzzelli, *3-D Aerodynamic Shape Design Using Hessians Based on Incomplete Sensitivities*, International Journal of Computational Fluid Dynamics, 14 : 103-116, 2001.

[22] F. Delbos, *Problèmes d'optimisation non linéaire avec contraintes en tomographie de réflexion 3D*, PhD thesis, Université Pierre et Marie Curie (Paris VI), 2004.

[23] M.C. Delfour, J.-P. Zolésio, *Shape Analysis via Oriented Distance Functions*, Journal of Functional Analysis, 123 : 129-201, 1994.

[24] M.C. Delfour, J.-P. Zolésio, *Shape analysis via distance functions : local theory*, Report CRM-2299, Centre de Recherches Mathématiques, Université de Montréal, 1995.

[25] A. Demeulenaere, C. Hirsch, *Application of multipoint and multistage optimization to the design of turbomachinery blades*, 6th European Conference on Turbomachinery : Fluid Dynamics and Thermodynamics, Proceedings, 2005.

[26] P. Dierckx, *Curve and Surface Fitting with Splines*, Clarendon Press, Oxford, 1993.

[27] L. C. Evans, *Partial Differential Equations*, Graduate Studies in Mathematics, volume 19 de American Mathematical Society, Providence, RI, 1998.

[28] J. Falcimaigne, J. Brac, Y. Charron, P. Pagnier, R. Vilagines, *Multiphase Pumping : achievements and perspectives*, Oil and Gas Science and Technology, Revue de l'IFP, 57(1) : 99-107, 2002.

[29] E. Feireisl, *Shape optimization in viscous compressible fluids*, Appl. Math. Opt., 47 : 59-78, 2003.

[30] *Fluent 6.2 UDF Manual*, Fluent Inc., 2005.

[31] *Fluent 6.2 User's Guide*, Fluent Inc., 2005.

[32] C. Foias, O. Manley, R. Rosa, R. Temam, *Navier-Stokes Equations and Turbulence*, Cambridge University Press, Cambridge, 2001.

[33] G. Geymonat, E. Sánchez-Palencia, *On the vanishing viscosity limit for acoustic phenomena in a bounded region*, Arch. Rational Mech. Anal., 75(3) : 257-268, 1980/81.

[34] D. Gilbarg, N.S. Trudinger, *Elliptic partial differential equations of second order*, volume 224 de Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Springer-Verlag, Berlin, second edition, 1983.

[35] V. Girault, P.-A. Raviart, *Finite element methods for Navier-Stokes equations*, volume 5 de Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 1986.

[36] P. C. Hansen, *Rank-Deficient and Discrete Ill-Posed Problems, Numerical Aspects of Linear Inversion*, SIAM, Philadelphia, 1998.

[37] J. Harinck, Z. Alsalihi, J. P. Van Buijtenen, R. A. Van den Braembussche, *Optimization of a 3D radial turbine by means of an improved genetic algorithm*, 6th European Conference on Turbomachinery : Fluid Dynamics and Thermodynamics, Proceedings, 2005.

[38] A. Henrot, M. Pierre, *Variation et Optimisation de Formes*, Springer, Berlin/Heidelberg/New York, 2005.

[39] O. Kavian, *Introduction à la théorie des points critiques et applications aux problèmes elliptiques*, volume 13 de Mathématiques & Applications (Berlin) [Mathematics & Applications], Springer-Verlag, Paris, 1993.

[40] E. Laporte, P. Le Tallec, *Numerical Methods in Sensitivity Analysis and Shape Optimization*, Birkhäuser, Boston, 2003.

[41] E. Leclerc, *Contrôle sub-optimal pour les écoulements instationnaires turbulents*, PhD thesis, Université des Sciences et Techniques du Languedoc, 2003.

[42] J. Lelong-Ferrand, J. M. Arnaudiès, *Cours de mathématiques tome 3 : géométrie et cinématique*, Dunod, Paris, 1975.

[43] *MATLAB*, The MathWorks Inc., 1994-2005.

[44] *Optimization Toolbox*, The MathWorks Inc., 1994-2005.

[45] B. Mohammadi, J.-M. Malé, N. Rostaing-Schmidt, *Automatic differentiation in direct and reverse modes : Application to optimum shapes design in fluid mechanics*, Proceedings of SIAM workshop on AD, Santa Fe, US, 1995.

[46] B. Mohammadi, *A new optimal shape design procedure for inviscid and viscous turbulent flows*, IJNMF, 25 : 183-203, 1997.

[47] B. Mohammadi, O. Pironneau, *Applied Shape Optimization for Fluids*, Clarendon Press, Oxford, 2001.

[48] B. Mohammadi, O. Pironneau, *Applied optimal shape design*, Journal of Computational and Applied Mathematics, 149 : 193-205, 2002.

[49] B. Mohammadi, O. Pironneau, *Shape Optimization in Fluid Mechanics*, Annu. Rev. Fluid Mech., 255-279, 2004.

[50] C. L. M. H. Navier, *Sur les lois du mouvement des fluides*, Mem. Acad. R. Sci. Inst. France, 6 : 389-440, 1823.

[51] J. Nocedal, S.J. Wright, *Numerical Optimization*, Springer, New York, 1999.

[52] S. Pierret, *Turbomachinery blade design systems*, Von Karman Institute for Fluid Dynamics, Lecture Series, 1999.

[53] S. Pierret, *Designing turbomachinery blades by means of the function approximation concept based on artificial neural network, genetic algorithm, and the Navier-Stokes equations*, PhD thesis, Faculté Polytechnique de Mons, Von Karman Institute for Fluid Dynamics, 1999.

[54] P. I. Plotnikov, J. Sokolowski, *Domain dependence of solutions to compressible Navier-Stokes equations*, SIAM J. Control Optim., 45(4) : 1165-1197, 2006.

[55] V. A. Skorospelov, D. V. Chirkov, S. G. Cherny, I. M. Pylev, A. A. Sotnikov, *Using 3D CFD and optimization methods for the design of Francis hydraulic turbine runner*, 6th European Conference on Turbomachinery : Fluid Dynamics and Thermodynamics, Proceedings, 2005.

[56] O. Soto, R. Löhner, *CFD shape optimization using an incomplete-gradient adjoint formulation*, Int. J. Numer. Eng., 51 : 735-753, 2001.

[57] M. Stanciu, *Gradient incomplet pour l'optimisation de formes en aérodynamique. Applications aux turbomachines*, PhD thesis, Université de Montpellier II, Sciences et Techniques du Languedoc, 2001.

[58] V. Sverak, *On optimal shape design*, J. Math. Pures Appl., 72(6) : 537-551, 1993.

[59] R. Temam, *Navier-Stokes equations, theory and numerical analysis*, North-Holland, Amsterdam/New York/Oxford, 1977.

[60] R. Vilagines, P. Pagnier, A. Akhras, M. El Hajem, J. Y. Champagne, R. Morel, *A study of the flow in a helico-axial pump rotor*, Conference on Modeling Fluid Flow (CMFF'03), Budapest, 2003.