

UNIVERSITÉ DE TECHNOLOGIE DE BELFORT-MONTBÉLIARD

# Modélisation inverse dynamique de vêtements

Rapport de stage ST50 – P2017

**JAFFALI Hamza**

Département Informatique  
Filière Image, Interaction  
et Réalité Virtuelle

**INRIA Grenoble Rhône-Alpes**

655 Avenue de l'Europe  
38330 Montbonnot-Saint-Martin  
<http://www.inria.fr/centre/grenoble>

Tuteurs en entreprise

**BERTAILS-DESCOUBES Florence**

**FRANCO Jean-Sébastien**

**Suiveur UTBM**  
**HOLWECK Frédéric**

# Remerciements

Je tiens tout d'abord à remercier particulièrement Florence Bertails-Descoubes et Jean-Sébastien Franco, mes tuteurs en laboratoire, pour leurs qualités tant humaines que professionnelles. Ils ont su se rendre disponibles pour m'aider, m'accompagner, et ont également su me faire confiance en me laissant une large autonomie dans le travail. Leurs compétences dans leurs domaines respectifs m'ont permis de développer des compétences scientifiques de valeur et d'apprendre énormément à leur côté. Ce sont des tuteurs de qualité comme il en existe peu, et je leur en suis vraiment reconnaissant.

Je remercie également les membres du groupe Elan pour m'avoir épaulé tout au long du projet. Je remercie particulièrement Eric Madaule pour son analyse du code et son aide précieuse. Je remercie, par la même occasion, Nicolas Pautet pour avoir pris le temps de m'éclairer sur les aspects mécaniques et les détails du modèle physique et de simulation de vêtements.

Je tiens aussi à remercier Frédéric Holweck, mon enseignant suiveur à l'UTBM, pour sa visite et ses conseils pour la restitution de mon stage, et pour l'attention qu'il apportera à ce rapport et ma soutenance orale. Je remercie également Fabrice Lauri d'avoir accepté de faire partie du jury de soutenance orale.

Je tiens à remercier les collègues de l'équipe BiPoP et du laboratoire Inria, notamment Alexandre Vieira, Eric Madaule, Nestor Bohorquez, Stanislas Brossette, Kirill Vorotnikov, Nahuel Villa, Nicolas Pautet, Matteo Ciocca, Jan Michalczyk, Alejandro Blumentals, Victor Romero, Dario Mangoni, Gabriel Devilliers, ainsi que tous les autres, pour leur convivialité durant le travail.

Je me dois de remercier Bernard Brogliato et Diane Courtiol, le chef et l'assistante de l'équipe BiPoP, pour m'avoir accueilli et accompagné. Je remercie également l'Inria de Grenoble pour l'accueil, et pour les moyens mis en œuvre pour permettre à ses employés de travailler dans des conditions très agréables.

Ces remerciements ne seraient pas complets sans ceux que je dois à Franck Gechter, Mireille Jacquot, Rachid Bouyekhf, et Ghislain Montavon, qui permettent chaque semestre aux étudiants de l'UTBM de pouvoir vivre une expérience professionnelle enrichissante.

J'adresse enfin un remerciement sincère à La Lumière et à mes proches pour m'avoir respectivement guidé et soutenu tout au long de mes études, et durant ce stage en particulier.

# Introduction

Dans le cadre de la formation d'ingénieur en informatique de l'Université de Technologie de Belfort Montbéliard (UTBM), les étudiants doivent effectuer un stage de fin d'études d'une durée de 24 semaines en entreprise ou en laboratoire. L'objectif de ce stage est de permettre à l'étudiant, qui s'est initié au métier d'ingénieur lors de son premier stage, de mettre en application et de valider les connaissances acquises durant ses études, dans les conditions qui seront celles de ses activités et responsabilités à venir.

Mon travail effectué durant ce stage de recherche pendant 6 mois prit place au sein de l'Inria (voir la section [1.1](#)), plus particulièrement sur le site de Montbonnot rattaché au centre Grenoble Rhône-Alpes à Montbonnot-Saint-Martin. Le sujet conjointement proposé par Florence Bertails-Descoubes, Jean-Sébastien Franco et Stefanie Wuhrer consiste en la modélisation inverse et dynamique de vêtements. J'ai donc été intégré l'équipe de recherche BiPoP, et plus particulièrement le groupe Elan composé d'ingénieurs, de chercheurs et de doctorants, et dirigé par Florence Bertails-Descoubes.

Plus précisément, l'objectif de ce stage est, dans un premier temps, de formuler le problème d'inversion de vêtements dans le cas dynamique, notamment sous la forme d'un problème d'optimisation, puis dans un second temps de proposer et d'implémenter des solutions pour résoudre le problème précédemment posé.

Le premier chapitre est dévolu à la présentation du contexte du stage : la structure d'accueil ainsi que la génèse du projet. Le deuxième chapitre détaille dans un premier temps les objectifs du projet, ainsi que les outils et la méthodologie utilisés. Il se poursuit ensuite avec la restitution du travail réalisé. Le troisième chapitre clos ce rapport avec une synthèse des résultats et un bilan personnel de ce projet de fin d'études.

# Table des matières

<b>1 Présentation du contexte du stage</b>	<b>7</b>
1.1 Structure d'accueil - INRIA . . . . .	7
1.1.1 INRIA Grenoble Rhône-Alpes . . . . .	8
1.1.2 Equipe-Projet BiPoP . . . . .	8
1.1.3 Equipe-Projet Morphéo . . . . .	9
1.2 Présentation du stage . . . . .	9
1.2.1 Contexte du projet . . . . .	9
1.3 Travail demandé . . . . .	11
1.3.1 Direction prise pour le stage . . . . .	12
1.4 Planning du stage . . . . .	12
<b>2 Travail réalisé</b>	<b>13</b>
2.1 Environnement de travail et développement . . . . .	13
2.1.1 Langages informatiques . . . . .	13
2.1.2 Logiciels et outils informatiques . . . . .	14
2.1.3 Méthodologie de travail . . . . .	15
2.1.4 Versioning et Clusters de calcul . . . . .	16
2.2 Etat de l'art . . . . .	17
2.3 Modèles mécaniques et numériques . . . . .	19
2.3.1 Modèle de coques élastiques de Koiter . . . . .	20
2.3.2 Modèle des Discrete Shells . . . . .	20
2.3.3 Modèle de simulation – Modèle d'intégration . . . . .	21
2.4 Formulation mathématique du problème . . . . .	22
2.4.1 Notations et paramétrisation du problème . . . . .	22
2.4.2 Formulation du problème d'inversion statique . . . . .	23
2.4.3 Passage au cas dynamique . . . . .	24
2.5 Méthodes d'optimisation sans gradient . . . . .	27
2.5.1 Etat de l'art non exhaustif . . . . .	27
2.5.2 Méthodes implémentées . . . . .	28
2.5.3 Tests de performance et comparaison . . . . .	32
2.6 Optimisation avec gradient . . . . .	39
2.6.1 Calcul du gradient dans le cas statique . . . . .	40
2.6.2 Calcul du gradient dans le cas dynamique . . . . .	42
2.7 Résultats et discussions . . . . .	44
2.7.1 Inversion statique et dynamique du bitriangle . . . . .	44
2.7.2 Inversion dynamique du drap à 16 sommets . . . . .	47
2.7.3 Validation des résultats . . . . .	52
<b>3 Conclusion</b>	<b>54</b>
3.1 Synthèse des résultats . . . . .	54
3.2 Perspectives . . . . .	54
3.3 Bilan personnel . . . . .	55
<b>Bibliographie</b>	<b>57</b>

# Table des figures

1.1 Logo de l'institut [98]	7
1.2 Site de Montbonnot-Saint-Martin [99]	8
1.3 Exemple d'activités de l'équipe BiPoP [100]	9
1.4 Exemple d'activités de l'équipe Morphéo [101, 102]	10
1.5 Planning hebdomadaire réel du stage	12
2.1 Exemple de visualisation – Paraview [103]	15
2.2 Exemple de visualisation – Création du modèle de Drap – Blender	15
2.3 Outils de versioning [104, 105]	16
2.4 Architecture réseau du cluster Nef – Inria [106]	17
2.5 Exemple de coque élastique [11]	20
2.6 Représentation visuelle de l'angle dièdre [107]	21
2.7 Représentation du maillage – Bitriangle	23
2.8 Représentation du maillage – Drap 16 sommets	23
2.9 Schéma récapitulatif du problème en dynamique	24
2.10 Déplacement d'une particule [56]	29
2.11 Fonctionnement global de l'algorithme génétique [108]	30
2.12 Trois exemples de promenade aléatoire dans un espace à trois dimensions [109]	32
2.13 Pseudo-code mettant en œuvre le recuit simulé [41]	32
2.14 Fonction de Rosenbrock – $f(\mathbf{x}) = \sum_{i=1}^{n-1} [100 (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$ [62]	33
2.15 Fonction Quartique – $f(\mathbf{x}) = \sum_{i=1}^{n-1} x_i^2 + (x_{i+1}^2 + x_i^2)^2$ [63]	33
2.16 Fonction de Rastrigin – $f(\mathbf{x}) = 10n + \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i)]$ [62]	34
2.17 Fonction de Styblinski-Tang – $f(\mathbf{x}) = 0.5 \sum_{i=1}^n x_i^4 - 16x_i^2 + 5x_i$ [62]	34
2.18 Fonction de Levy – $f(\mathbf{x}) = \sin^2(\pi\omega_1) + \sum_{i=1}^{n-1} (\omega_i - 1)^2 [1 + 10 \sin^2(\pi\omega_i + 1)] + (\omega_n - 1)^2 [1 + \sin^2(2\pi\omega_n)]$ avec $\omega_i = 1 + 0.25(x_i - 1)$ [63]	34
2.19 Fonction de Griewank – $f(\mathbf{x}) = \sum_{i=1}^n \frac{x_i^2}{4000} - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ [63]	35
2.20 Performance profile – Optimisation par essaim particulaire	37
2.21 Performance profile – Algorithme génétique	38
2.22 Performance profile – Promenade aléatoire	39
2.23 Performance profile – Récuit simulé	40
2.24 Performance profile – Comparaison des méthodes	41
2.25 Forme au repos (gauche) et forme à l'équilibre (droite) – Bitriangle	44
2.26 Résultat de l'inversion statique du Bitriangle	45
2.27 Résultat de l'inversion statique du Drap 16	46
2.28 Coordonnée fixée selon l'axe $y$ – Bitriangle	46
2.29 Résultat de l'inversion dynamique du Bitriangle – $T_0 = 25, T_N = 50$	47
2.30 Forme au repos – Drap 16	47
2.31 Evolution de l'affaissement du drap – Drap 16	48
2.32 Exemple d'état initial d'une particule – En bleu l'état initial, en jaune la forme au repos théorique – Drap 16	48
2.33 Résultat de l'inversion dynamique du Drap 16, dans le cas 1 – $T_0 = 1, T_N = 20 – J_1(x_0) = 0.33$	49
2.34 Exemple d'initialisation d'une particule – Drap 16	49
2.35 Simulation d'affaissement après inversion dynamique du Drap 16, dans le cas 1 – $T_0 = 5, T_N = 25 – J_1(x_0) = 0.076$	50

2.36 Résultat de l'inversion dynamique du Drap 16, dans le cas 1 – $T_0 = 200, T_N = 280$ – $J_1(x_0) = 0.0197$	50
2.37 Simulation d'affaissement après inversion dynamique du Drap 16, dans le cas 1 – $T_0 = 200, T_N = 280 - J_1(x_0) = 0.0197$	51
2.38 Résultat de l'inversion dynamique du Drap 16, dans le cas 4 – $T_0 = 1, T_N = 20$ (gauche) – $T_0 = 5, T_N = 25$	51
2.39 Résultat de l'inversion dynamique du Drap 16, dans le cas 4 – $T_0 = 25, T_N = 50$ (gauche) – $T_0 = 200, T_N = 280$	52
2.40 Simulation d'affaissement après inversion dynamique du Drap 16, dans le cas 4 – $T_0 = 200, T_N = 280$	52
2.41 Résultat de l'inversion dynamique du Drap 16, dans le cas 5 – $T_0 = 200, T_N = 280$	53

# Chapitre 1

## Présentation du contexte du stage

Mon travail a eu lieu au sein de l'INRIA à Grenoble. Nous allons donc présenter succinctement l'institut, le centre de Grenoble ainsi que l'équipe de recherche intégrée. Nous poursuivrons par une présentation du sujet de stage, du travail demandé, afin de terminer par une description du planning prévisionnel et réel.

### 1.1 Structure d'accueil - INRIA

L'INRIA, Institut National de Recherche en Informatique et en Automatique, est un institut de recherche français en mathématiques et informatique. Initialement créé en janvier 1967, il a le statut d'établissement public à caractère scientifique et technologique. L'INRIA tente de répondre aux enjeux pluridisciplinaires et applicatifs de la transition numérique, en apportant ses résultats et ses compétences dans des domaines tels que la santé, les transports, l'énergie, les communications, la sécurité et la protection de la vie privée, la ville intelligente, l'usine du futur...



FIGURE 1.1 – Logo de l'institut [98]

En quelques chiffres, l'INRIA c'est : 178 équipes-projets dont 139 en collaboration avec des universités et d'autres établissements de recherche, 4600 publications scientifiques par an, 300 thèses soutenues, 390 brevets en portefeuilles regroupés dans 190 familles actives, 141 logiciels déposés à l'Agence pour la Protection des programmes en 2014, 34 start-up Inria créées depuis 2010, 2 600 collaborateurs de 87 nationalités différentes dont 1 750 scientifiques, 230 M Euros de Budget total et 27 % de Ressources propres.

Le premier centre créé est situé à Rocquencourt, et constitue également le siège actuel de l'institut. L'INRIA est désormais composé de huit centres de recherche autonomes répartis sur tout le territoire français :

- Centre de recherche Bordeaux - Sud-Ouest
- Centre de recherche Grenoble - Rhône-Alpes
- Centre de recherche Lille - Nord Europe
- Centre de recherche Nancy - Grand Est
- Centre de recherche Paris - Rocquencourt

- Centre de recherche Rennes - Bretagne Atlantique
- Centre de recherche Saclay - Île-de-France
- Centre de recherche Sophia Antipolis - Méditerranée

### 1.1.1 INRIA Grenoble Rhône-Alpes

Présent depuis 1992, le Centre de Recherche Inria Grenoble – Rhône-Alpes s'est constitué puis développé en partenariat avec les pôles et établissements de Grenoble et Lyon. Le centre compte aujourd'hui 34 équipes de recherche, dont 11 sont localisées à Lyon.



FIGURE 1.2 – Site de Montbonnot-Saint-Martin [99]

Les centres sont structurés en *Equipes-Projets* composés de personnels permanents et non-permanents, avec des objectifs scientifiques précis et une durée de vie limitée (évaluation tous les 4 ans). Les principales thématiques de recherche d'Inria Grenoble – Rhône-Alpes, définies dans le cadre de son plan stratégique "Objectif Inria 2020" ont permis d'identifier 6 priorités scientifiques :

- des robots partageant notre espace de vie et de travail ;
- internet des objets et internet des données : la société numérique ;
- modélisation des interactions en biologie ;
- formes, apparences et mouvements pour les mondes virtuels ;
- interface matériel logiciel ;
- apprentissage et optimisation distribuée pour systèmes à grande échelle.

J'ai intégré l'un des cinq sites d'implantations du Centre Grenoble – Rhône-Alpes, situé à l'Inovallée de Montbonnot-Saint-Martin.

### 1.1.2 Equipe-Projet BiPoP

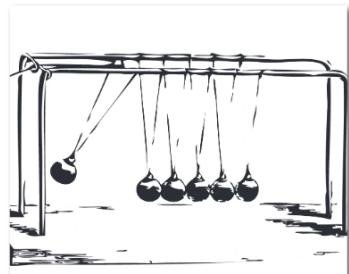
Mon stage a été accueilli par l'équipe de recherche BiPoP, dirigée par Bernard Brogliato. Les thématiques de cette équipe-projet sont centrées autour des systèmes dynamiques non réguliers (modélisation, commande, simulation numérique) et de l'optimisation non-différentiable. Les travaux de cette équipe relèvent donc de la mécanique non-régulière, de l'automatique, et de l'analyse non-lisse.



Hair dynamics



Modelling and simulation bipeds



Impact modelling

FIGURE 1.3 – Exemple d’activités de l’équipe BiPoP [100]

### Groupe Elan

Le groupe Elan a pour objectif de discuter les problématiques liées à la modélisation et la simulation de structures élancées (tiges, plaques, coques), y compris en présence de contacts frottants. Sous la direction de Florence Bertails-Descoubes, c'est un groupe de 6 personnes environ, composé de chercheurs, de doctorants, d'ingénieurs de recherche, d'expérimentateurs, de post-doctorants et d'étudiants en stage de Master.

#### 1.1.3 Equipe-Projet Morpheo

Le sujet de stage a été proposé par l'équipe BiPoP conjointement avec l'équipe Morpheo, dirigée par Edmond Boyer. L'équipe Morpheo s'intéresse à la capacité de percevoir et d'interpréter les formes en mouvement, à l'aide de systèmes multi-caméras, dans l'objectif d'analyser la déformation, de générer des animations et de développer des environnements immersifs et interactifs. La mise en place de systèmes multi-caméras fournit une information dense sur les formes et leurs mouvements. Ceci permet d'ouvrir la voie à un nouveau domaine d'investigation en recherche sur la modélisation, la compréhension et l'animation de formes dynamiques réelles, dont les vêtements.

## 1.2 Présentation du stage

### 1.2.1 Contexte du projet

La modélisation, simulation et animation de vêtements en informatique graphique n'a cessé d'intéresser de nombreux chercheurs et ingénieurs, et est actuellement en plein essor. On peut aujourd'hui obtenir des animations très réalistes, avec des formes très proches visuellement de la réalité, et des modèles de plus en plus intéressants permettant de simuler une variété de tissus et de vêtements toujours plus complexes.

Néanmoins, la phase de modélisation du vêtement et de détermination de ses paramètres physiques et géométriques, en vue d'une simulation, reste une étape très coûteuse en termes de temps et de moyens mis en oeuvre. La difficulté peut être double dans la mesure où, afin d'instancier le simulateur, il faut connaître à la fois la géométrie et les caractéristiques physiques du vêtement.

Dans le cas où l'on peut connaître la géométrie du vêtement (forme au repos, forme à l'équilibre, ...), déterminer, à partir de notre objet réel, les paramètres physiques (raideur, coefficients de frottement, masse, ...) n'est pas une étape triviale, et il en est de même pour l'opération inverse. Les méthodes classiques en mécanique consistent à effectuer des essais en prélevant un échantillon réel de l'objet à modéliser, ce qui peut constituer un procédé assez fastidieux, complexe à mettre en place, invasif, et qui ne suffit pas pour des configurations complexes impliquant du contact, du frottement, etc. Les méthodes utilisées par les studios d'animation sont basées



FIGURE 1.4 – Exemple d’activités de l’équipe Morpheo [101, 102]

sur le procédé d’essai/erreur. On sélectionne un jeu de paramètres de simulation, et on corrige manuellement ces paramètres, jusqu’à converger vers une solution qui ressemble à ce que l’on attend. Ce procédé est souvent long et pénible, et nous conduit vers une solution plus ou moins approximative.

L’idée principale de ce projet est donc de trouver une solution à ces difficultés, en partant d’une (ou plusieurs) représentation(s) visuelle(s) de notre objet. À partir de ces données, on se propose alors de résoudre un problème inverse pour retrouver les paramètres permettant d’instancier le simulateur. Il faut alors disposer d’un dispositif de vision permettant de reconstruire des modèles 3D à partir d’objets, et d’autre part d’un simulateur prédictif pour simuler la déformation de l’objet. Le but est donc d’inférer les propriétés mécaniques de ces objets uniquement à partir de données géométriques (maillages de l’objet), de manière à construire une méthode d’identification paramétrique non invasive, et de pouvoir éventuellement prédire le mouvement de ces objets.

Selon Kern [26], le problème direct consiste à déduire les effets en fonction de causes connues, tandis que le problème inverse consiste à déterminer les causes connaissant les effets (on parle “d’expliquer les effets”). En effet, beaucoup de problèmes peuvent se reformuler comme des problèmes inverses. À partir du moment où l’on possède des observations, et que l’on cherche à expliquer ces observations pour pouvoir réaliser de la prédiction ou simulation, on peut se ramener à la résolution d’un problème inverse.

On distingue usuellement deux classes de problèmes inverses : les problèmes de conception inverse et les problèmes de mesure inverse. La première classe consiste en la détermination des paramètres géométriques de l’objet en connaissant initialement ses paramètres physiques. La seconde classe consiste en l’estimation de paramètres physiques de l’objet, en connaissant les

paramètres géométriques de ce dernier.

Le but de ce projet est de permettre, à partir d'une succession de maillages 3D du vêtement représentant son mouvement, et en supposant les paramètres physiques connus, la détermination de la configuration non déformée (forme au repos), ainsi que la configuration et vitesse initiale du vêtement. Cela consiste donc en la résolution d'un problème de conception inverse.

La succession de maillages 3D représentant le mouvement du vêtement peut être déterminée à l'aide d'un dispositif de vision 3D. À l'aide d'un dispositif multi-caméras et de logiciels adéquats, comme la plateforme d'acquisition Kinovis et les travaux de l'équipe Morpheo, si l'on fait bouger le vêtement on peut reconstruire une succession de maillages le représentant. On placera alors en entrée de notre problème ces observations, ainsi qu'une connaissance en amont des paramètres physiques, et l'objectif sera de déterminer les paramètres restants en entrée du simulateur (forme au repos, forme initiale, ...) pour retrouver un mouvement simulé du vêtement similaire à l'observation. Durant ce stage, la séquence de maillage sera néanmoins construite uniquement par simulation.

Cette relation entre le domaine de la vision 3D et de la modélisation mécanique est profitable pour l'un comme pour l'autre. Dans un cas, on peut obtenir des configurations géométriques 3D à partir de poses d'objets réels ce qui constitue l'information principale du processus d'inversion. Dans l'autre cas, on se sert des modèles mécaniques relatifs aux objets et des résultats de l'inversion, par exemple, pour valider ou corriger les maillages 3D obtenus par les dispositifs de visions, ces maillages pouvant être plus ou moins pertinents en termes de topologie et d'association de sommets d'une séquence de maillages.

Mon stage de fin d'études intervient donc dans le cadre d'un projet de coopération entre le domaine de la modélisation inverse pour la mécanique et de la vision par ordinateur. Il fait suite en effet au travail de thèse mené par Romain Casati portant sur l'inversion statique de vêtements, mené au sein de l'équipe BiPoP et sous l'encadrement de Florence Bertails-Descoubes. Afin de poursuivre dans le domaine de l'inversion de vêtements, cette fois-ci dans le cas dynamique, l'association avec l'équipe de vision Morpheo a été envisagée. En effet, Jean-Sébastien Franco s'est également intéressé à la capture de vêtements durant ses travaux, notamment en proposant une estimation de la forme d'un corps en mouvement en présence de vêtements larges ??.

### 1.3 Travail demandé

Etant donné que le domaine de l'inversion dynamique de vêtements a été très peu étudié par la communauté scientifique, le but de ce stage est d'entamer la recherche dans ce domaine, de soulever les principales problématiques mises en jeu afin de proposer une première approche de résolution.

Le première étape du stage est de réaliser un état de l'art en ce qui concerne la détermination de paramètres des vêtements, l'inversion de vêtements, ainsi que ce qui a été développé par les chercheurs en vision par ordinateur.

La seconde tâche demandée est la formulation du problème d'inversion de vêtements dans le cas dynamique, ce qui consiste en la mise en place des hypothèses simplificatrices, et l'écriture du problème comme un problème d'optimisation.

Le travail suivant est la résolution de ce problème d'optimisation. Cela passe alors, dans un premier temps, par l'établissement d'un état de l'art des méthodes d'optimisation sans calcul de gradient, nécessaire pour pouvoir ensuite faire un choix d'implémentation pour résoudre le problème d'optimisation. Dans un second temps, la mise en place du calcul du gradient pour implémenter les méthodes d'optimisation "classiques" constitue le dernier travail demandé pour ce stage.

### 1.3.1 Direction prise pour le stage

Le problème de détermination ou d'estimation des paramètres de vêtements aurait pu être attaqué sous un autre angle, mais pour ce stage une direction précise a été choisie. En effet, les modèles de mécaniques et de mouvement ont été mis en équations par des physiciens, et leur expression est bien connue. La mise en équation fait intervenir des dérivées partielles de forces et d'énergies (potentielle notamment), ce qui lui procure une expression d'un type particulier. Etant donné que l'on décide de traiter la question comme un problème inverse, l'expression des équations du modèle ont guidé vers une méthode de résolution basée sur la mise en place d'une fonction objectif et sur son optimisation. Dès lors, l'étude des méthodes d'optimisation adaptées à notre problème prend tout son sens et donne la direction à prendre pour ce stage.

## 1.4 Planning du stage

Etant donné que mon stage intervient dans un projet de recherche scientifique assez vaste et nouveau, il est assez difficile de pouvoir fournir un planning daté précis et prévisionnel du travail. Néanmoins, les grandes étapes d'approche du sujet étaient connues au début du stage, puis elles ont pu être affinées au fur et à mesure. Le planning hebdomadaire réel du stage est quant à lui présenté ci-dessous.

Mois	Février				Mars				Avril				Mai				Juin				Juillet			
Semaine	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Tâche du projet																								
Etat de l'art																								
Méthodes d'optimisation sans gradient																								
Inversion statique																								
Inversion dynamique																								
Calcul du gradient																								

FIGURE 1.5 – Planning hebdomadaire réel du stage

# Chapitre 2

## Travail réalisé

### 2.1 Environnement de travail et développement

#### 2.1.1 Langages informatiques

Le principal langage que j'ai été amené à utiliser est le célèbre langage de programmation orienté objet et multi paradigme **Python**. Langage créé par Guido van Rossum, il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser. C'est un langage qui peut s'utiliser dans de nombreux contextes et s'adapter à tout type d'utilisation grâce à de nombreuses bibliothèques spécialisées. Il est particulièrement utilisé comme langage de script, comme langage de développement de prototype, et est particulièrement répandu dans le monde scientifique grâce à ses nombreuses extensions destinées au calcul numérique.



En effet, je me suis initialement tourné vers la version Python 3.5.2 afin d'implémenter mes premiers tests d'optimisation sans gradient. Par la suite, étant donné que le reste du code Python du projet est de version 2.7, j'ai dû adopter cette ancienne version du langage. En fait, tout code, documentation et spécification ajouté, depuis la sortie de Python 2.1 alpha, est détenu par la Python Software Foundation (PSF), une association sans but lucratif fondée en 2001, modelée d'après l'Apache Software Foundation. Afin alors de réparer certains défauts du langage (par exemple l'orienté objet avec deux types de classes), et pour nettoyer la bibliothèque standard de ses éléments obsolètes et redondants, Python a choisi de casser la compatibilité ascendante dans la nouvelle version majeure, Python 3.0, publié en décembre 2008. Il y a donc une travail d'adaption du code à faire pour passer d'une version à l'autre.

Afin d'implémenter des algorithmes d'optimisation et simulation numérique, un certain de nombre de package Python existent pour les calculs scientifiques, et ils sont assez variés et de qualité. Par exemple, Matplotlib [93] est une bibliothèque de d'affichage Python 2D qui produit des figures de qualité de publication dans une variété de formats imprimés et d'environnements interactifs à travers les plates-formes. Aussi, NumPy [94] est le package fondamental pour l'informatique scientifique avec Python. Il contient entre autres un puissant classe de tableau N-dimensionnel, des fonctions sophistiquées (diffusion), des outils d'intégration du code C / C ++ et Fortran et l'algèbre linéaire utile, la transformée de Fourier et des générateurs numériques aléatoires. Ces deux packages appartiennent à SciPy [95], un ensemble de logiciels Python open source pour les mathématiques, la science et l'ingénierie.

De plus, j'ai également eu l'occasion de m'intéresser à certaines parties du code de *Cloth* implémentées en **C** par Romain Casati, ancien doctorant de Florence Bertails-Descoubes (voir 2.1.2). Le C est un langage de programmation généraliste et l'un des plus utilisés dans le monde. Il fut à la base inventé pour réécrire le système d'exploitation UNIX. Certains de ses aspects ont été repris afin de créer le C++, langage orienté objet de renomée, plus proche de la machine, parfois considéré plus puissant, que son concurrent Java.

Pour le développement en Python, l'IDE (environnement de développement) utilisé fut principalement **PyCharm**, développé par JetBrains, similaire à CLion pour le C et IntelliJ pour Java. Pour manipuler le code écrit en C, dont la majeure partie était un travail de lecture, j'ai principalement utilisé l'outil **gedit** de Linux.

### 2.1.2 Logiciels et outils informatiques

#### Cloth

Le principal logiciel que j'ai utilisé tout au long du stage est le logiciel **Cloth**, développé par Romain Casati durant sa thèse de doctorat. Développé en C afin d'exploiter la rapidité du langage et de permettre la parallélisation des calculs, l'interface quant à elle est développée en Python afin de permettre une utilisation simple du simulateur, ce qui implique la présence d'un wrapper (cython) entre la partie C et Python.

Le logiciel Cloth permet de simuler les mouvements des vêtements en prenant en compte les paramètres d'instanciation. En effet, le simulateur doit disposer d'un certain nombre d'informations afin de pouvoir initialiser la simulation, dont les principales sont :

- la forme au repos du vêtement<sup>1</sup>;
- la forme initiale (à  $t = 0$ ) du vêtement ;
- les paramètres physiques : raideur (traction, cisaillement, courbure), densité de masse ;
- la vitesse initiale du vêtement ;
- le vecteur forces extérieures sur le vêtements (généralement force de gravité) ;
- sommets du vêtement fixés (encastrés).

Afin d'instancier un vêtement dans le logiciel il faut instancier en objet de la classe *Cloth*. La méthode *step* permet d'avancer dans la simulation d'un pas de temps  $dt$  précisé en paramètre.

#### Paraview

**ParaView** est un logiciel libre de visualisation de données, utilisé pour représenter graphiquement des données statistiques, des visualisations scientifiques et des objets 3D notamment (figure 2.1). Développé principalement par le Sandia National Laboratories et la société Kitware Inc. Il est fondé sur la bibliothèque VTK et publié sous licence BSD.

Nous avons été amené à utiliser Paraview afin de visualiser les résultats des simulations et étapes du processus d'optimisation. Le logiciel permet effectivement de visualiser une succession de maillage, et d'ajouter des filtres afin de traiter les données de manière à ce qu'elles soient plus compréhensible ou pertinentes.

Nous avons alors pu utiliser ces filtres pour notifier des informations relatives au temps de simulation, ou la valeur de la fonction objectif tout au long de l'optimisation. Le logiciel permet aussi d'exporter simplement ces animations au format de vidéos *.avi* et d'effectuer des captures d'écran.

#### Blender

Blender est un logiciel libre et gratuit de modélisation, d'animation et de rendu en 3D, créé en 1995. Il est actuellement développé par la Fondation Blender. Il dispose de fonctions avancées de modélisation, de sculpture 3D, de dépliage UV, de texturage, de rigging, d'armaturage, d'animation 3D, et de rendu. Il gère aussi le montage vidéo non linéaire, la composition, la création

---

1. Forme du vêtement sans aucune vitesse, accélération ou force extérieure

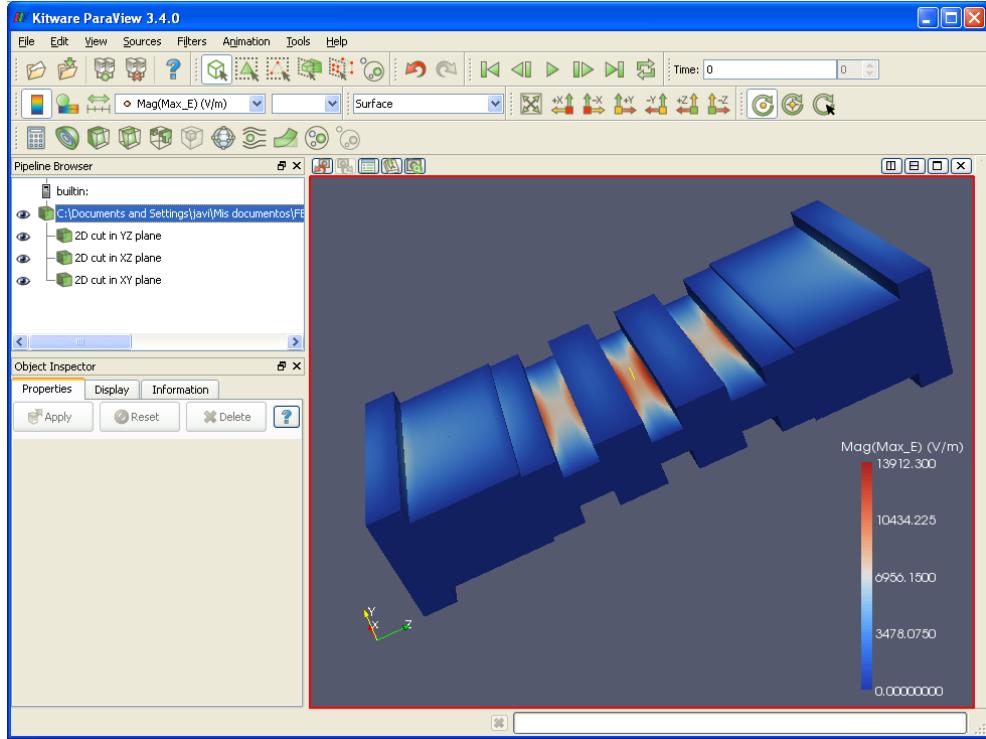


FIGURE 2.1 – Exemple de visualisation – Paraview [103]

nodale de matériaux, la création d'applications 3D interactives ou de jeux vidéo grâce à son moteur de jeu intégré (le Blender Game Engine), ainsi que diverses simulations physiques telles que les particules, les corps rigides, les corps souples et les fluides (voir figure 2.2).

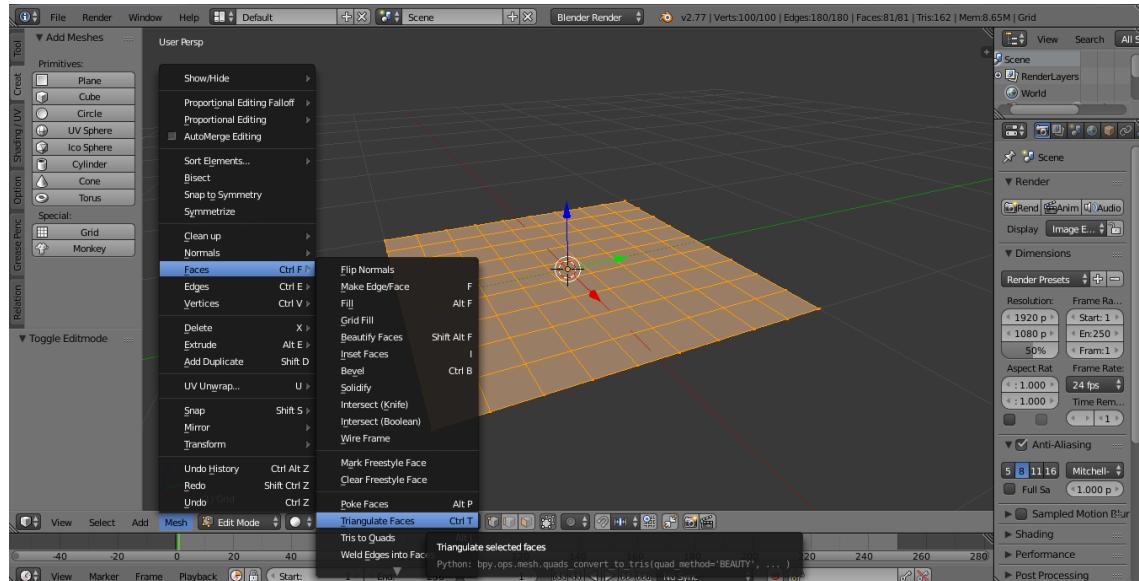


FIGURE 2.2 – Exemple de visualisation – Création du modèle de Drap – Blender

C'est un logiciel multifonction, et qui a fait ses preuves dans le domaine de l'informatique graphique et également dans le monde scientifique. Ainsi, afin de générer les modèles 3D représentant la forme au repos des exemplaires de vêtements pour simulation et inversion, l'utilisation du logiciel Blender apparue comme une solution simple et rapide.

### 2.1.3 Méthodologie de travail

Tout au long de mon stage, j'ai dû interagir régulièrement avec mes encadrants et ponctuellement avec les membres du groupe Elan.

De manière hebdomadaire, une réunion est organisée avec mes tuteurs en entreprise, Florence Bertails-Descoubes et Jean-Sebastien Franco, afin de pouvoir travailler ensemble sur le projet, et apprécier son avancement. À chaque réunion, je présente alors le travail effectué depuis la dernière réunion, les fonctions implémentées, les articles rencontrés et les résultats obtenus. C'est l'occasion pour nous de discuter de l'avancement, d'analyser les résultats, pour mes encadrants de m'informer des inexactitudes et erreurs à corriger dans mon travail, et de définir les prochaines étapes et implémentations sur le court et long terme.

Le principal support de ces réunions est une notice en LaTeX, que j'ai rédigé en continu pendant mon stage. En effet, cette notice de stage réunit de manière écrite les détails concernant les différentes tâches effectuées durant le stage. Cela va de l'explication des algorithmes d'optimisation, de la formulation mathématique du problème, à l'état de l'art des travaux scientifiques, en passant par des détails techniques de compilation ou d'utilisation des clusters de calculs.

Entre deux réunions, mes encadrants sont disponibles pour toutes sollicitations et discussions, et c'est aussi le cas pour les membres du groupe Elan. Je travaille donc conjointement avec l'ingénieur informatique chargé de maintenir et faire évoluer les logiciels développés par le groupe, afin de m'assurer que le code que je produis est conforme aux normes de développement imposées dans le groupe, afin de s'entraider dans la compréhension de certaines parties du code informatique des logiciels, et afin de gérer et utiliser convenablement les services informatiques mis à disposition par l'Inria. L'ingénieur de recherche spécialisé en mécanique poursuit quant à lui le travail de Romain concernant l'inversion statique de vêtement, et nous avons donc été amené à collaborer sur certains aspects similaires de nos travaux, notamment la compréhension et utilisation du logiciel Cloth et des problèmes possibles liés à l'inversion statique.

#### 2.1.4 Versioning et Clusters de calcul

##### Versioning

Le versioning est une pratique répandue dans le monde du développement, notamment lorsque plusieurs personnes doivent intervenir simultanément sur un même projet de développement logiciel. Mis en place à travers un logiciel spécialisé, ce processus permet de conserver une trace des modifications successives apportées à un fichier numérique (documentation, code source, base de données). Il est ainsi possible de retrouver des données effacées, mais aussi d'effectuer de nombreuses manipulations, comme la comparaison de sous-parties d'un logiciel évoluant parallèlement.



FIGURE 2.3 – Outils de versioning [104, 105]

Parmi les nombreux logiciels spécialisés existant, j'ai été amené à utiliser le logiciel de gestion de versions Git. Plus précisément j'ai pu utiliser la forge de projet mise à disposition par l'Inria [97]. Ainsi, le service en ligne GForge, développé principalement par SourceForge, fournit une interface unifiée à une série de logiciels serveur, ne limite pas le nombre de contributeurs à un projet et permet à l'Inria de privatiser les différents projets des équipes.

##### Clusters de calcul

Afin de pouvoir accélérer les calculs liés à la simulation et à l'inversion de vêtements, qui peuvent s'avérer assez longs si l'on manque de puissance de calcul, l'idée de lancer plusieurs calculs d'optimisation en parallèle sur un cluster de calcul a été proposée lors d'une réunion de travail.

Les méthodes d'optimisation implémentées nécessitent un assez important nombre d'évaluation de la fonction objectif. Cette fonction objectif, comme nous le verrons par la suite, nécessite à son tour de lancer une simulation complète d'affaissement du vêtement. Ceci peut donc nous amener rapidement à un grand temps de calcul nécessaire pour les calculs d'optimisation.

La première solution, appelée "parallélisation naïve", consiste à paralléliser les appels à la fonction objectif, car c'est ce qui constitue le cœur du problème en terme de temps de calcul. Néanmoins, les simulations de vêtements développées en C sont déjà elles même parallélisées. Un rapide essai nous a permis de constater qu'ajouter une parallélisation naïve par dessus la parallélisation présente dans Cloth ne permettait pas une meilleure performance sur mon ordinateur de travail (constitué de 8 coeurs).

La seconde solution consiste à lancer les différents calculs d'inversion et d'optimisation de manière distribuée, sans paralléliser l'intérieur de la méthode d'optimisation. Pour ce faire, il faut disposer de nombreuses ressources de calculs et de mémoire pour que cela soit efficace et pertinent. C'est alors que la possibilité de lancer ces calculs sur des machines dédiées (clusters de calcul) nous est apparue être une solution à envisager.

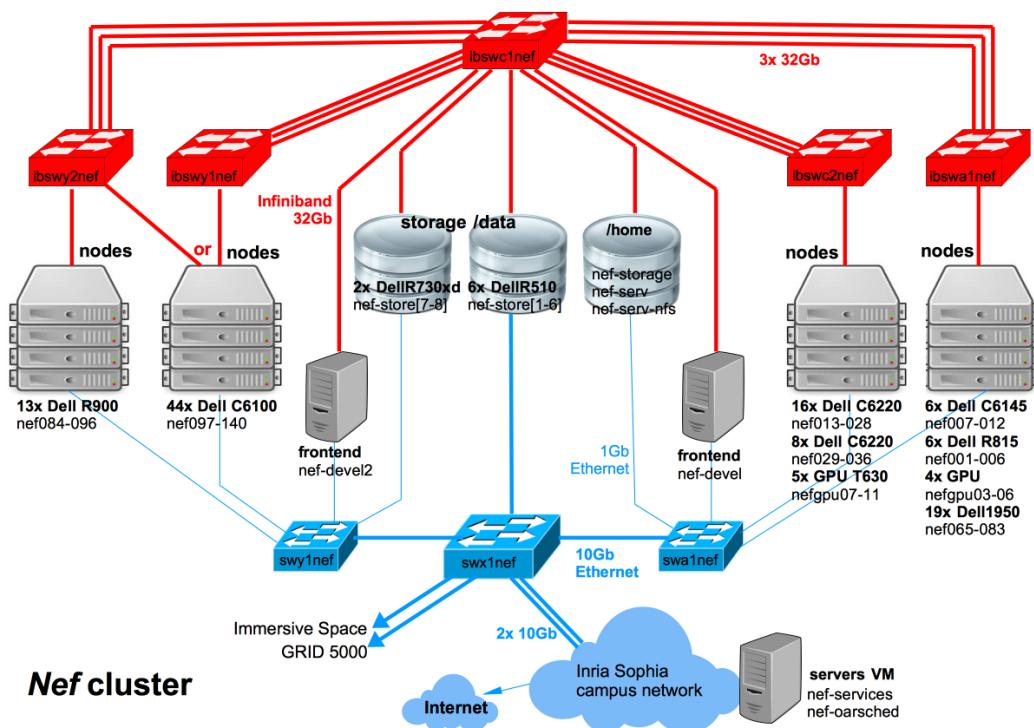


FIGURE 2.4 – Architecture réseau du cluster Nef – Inria [106]

L'Inria dispose de clusters de calculs accessibles par toute équipe, après formulation précise du besoin d'utilisation pour une période donnée. D'autre part, le cluster Ciment implanté sur le campus de Grenoble est également accessible par notre équipe de recherche BiPoP. La création d'un compte, l'importation des données et du code, et la planification de lancement des tâches ont permis d'accélérer les calculs d'inversion et d'optimisation et de pouvoir obtenir et donc analyser plus rapidement les résultats.

## 2.2 Etat de l'art

La modélisation et simulation de vêtements sont difficiles à mettre en place en raison des nombreux paramètres qui doivent être ajustés pour permettre un rendu et comportement très

proche et réaliste pour un tissu particulier. Dès lors, de nombreuses méthodes ont vu le jour afin de proposer une manière d'estimer ou d'ajuster correctement certains paramètres liés à la simulation ou à la mécanique du vêtement, et ce en utilisant différentes technologies.

Nous proposons alors de présenter les différents travaux connus de la littérature. Tout d'abord, nous présenterons chronologiquement et à chaque paragraphe les travaux qui ont pu être menés concernant l'estimation de paramètres des vêtements. Nous discuterons ensuite succinctement des principaux et récents travaux menés les chercheurs en vision par ordinateur. Nous terminerons par une synthèse des principaux manquements et inconvénients de ces méthodes, nous ayant poussé à proposer une nouvelle manière d'aborder le problème.

En 1997, Jojic *et al.* [1] mettent en place un dispositif de technologie laser pour capter la forme de la surface formée par le vêtement. Par un méthode d'analyse-synthèse, le but est de comparer le modèle de drapé du vêtement avec les données capturées et de chercher la meilleure correspondance. L'avantage de leur travail est qu'il peut être appliqué à tout modèle physique de vêtements, et qu'il constitue une alternative à l'utilisation du Kawabata Evaluation System (KES) [96].

Bhat *et al.*, proposent en 2003 [2] l'utilisation de vidéos d'habits réels afin de reconnaître le type de matériau d'un tissu. En effet, la mise en place d'une métrique basée sur la correspondance des plis est utilisée pour comparer la vidéo réelle et la simulation. L'application de leur travail permet de différencier 4 types différents de matériaux. Néanmoins, il ne serait pas possible d'estimer correctement les paramètres de vêtements possédant des couleurs ou des textures.

La même année, David Pritchard [3] s'intéresse dans le cadre de sa thèse de Master à l'influence des paramètres de vêtements pour un algorithme de simulation. Par la même occasion, il présente un système permettant de capturer le mouvement des surfaces déformables, notamment les habits, incluant à la fois une capture de la géométrie et la paramétrisation. Ces données sont alors utilisées pour retrouver les paramètres liés à la simulation. Plus en détail, il utilise la stéréovision pour capturer la géométrie et ensuite identifier un motif imprimé sur le vêtement, même en présence de mouvement rapide. Les informations de raideur en traction, cisaillement et flexion pourraient être calculées à partir des données mesurées, se proposant donc comme une amélioration des résultats présentés par Bhat *et al.* [2].

En 2007, Magnenat-Thalmann *et al.* [4] discutent du lien entre modélisation de vêtements et de la manière de les mesurer, et entre la simulation de vêtements et les paramètres nécessaires. Le travail de Kunitomo *et al.* [5], intervenant trois années plus tard, propose uniquement l'utilisation d'un système de *motion capture*, en fixant de petits marqueurs sur le vêtement, sans utiliser d'autres appareils. La capture se focalise sur la forme du vêtement, la vitesse et la courbure de ce dernier, et permet de reformuler cela comme un problème de minimisation d'erreur entre le mouvement capturé et le mouvement simulé, en considérant à la fois le cas statique et dynamique. L'optimisation se fait à l'aide d'une méthode non basée sur le gradient, le recuit simulé (voir 2.5.2 et 2.5.1), permettant de mettre en évidence cinq paramètres physiques : la raideur en étirement et en flexion, le coefficient d'amortissement, la densité et l'épaisseur du vêtement.

Dans le même esprit de formulation du problème comme un problème d'optimisation, Wang *et al.* [6] proposent un modèle élastique linéaire par morceau, approchant un modèle non linéaire, ainsi que des techniques de mesure des déformations élastiques en traction et en flexion sur des vêtements réels. En disposant d'une base de données de 10 exemples où les paramètres de traction et flexion sont mesurés, le but est de résoudre un problème d'optimisation afin de déterminer ces paramètres. Des algorithmes de type BFGS et Quasi-Newton sont utilisés, mais la présence de nombreux minimums locaux ne permet pas d'avoir une méthode robuste. De plus, la méthode nécessite d'avoir une base de données préalable pour fonctionner, et pour finalement ne déterminer qu'un nombre limité de paramètres statiques.

Un autre méthode, également invasive, mise en place par Miguel *et al.* en 2012 [7], consiste à mesurer les déformations 3D complexes d'un morceau rectangulaire de vêtement à l'aide d'un dispositif adapté. Reformulant ceci comme un problème d'optimisation afin de déterminer les inconnues qui sont un vecteur de raideur  $k$  et la position  $x_n$  des noeuds du vêtement, la résolution s'effectue par une succession de deux boucles imbriquées, affinant chaque variable séparément.

Plus récemment, en 2013, Katherine L. Bouman *et al.* [8] présentent une méthode pour analyser automatiquement des vidéos de vêtements bougeant sous l'effet de forces de vent variées et inconnues. Le but est de retrouver les propriétés matérielles de raideur et densité de masse du vêtement. Pour ce faire, un modèle de regression linéaire est utilisé, en combinaison avec une analyse en composantes principales (ACP). Une base de données est utilisée pour la phase d'apprentissage et de test de la méthode.

En 2015, Davis *et al.* [9] établissent une connexion entre les fondements de la mécanique vibratoire et les techniques de vision par ordinateur. Sous l'hypothèse de connaissance de la géométrie du vêtement, et à partir de vibrations (sonores), on extrait à l'aide d'un dispositif les petits et parfois imperceptibles mouvements du vêtements. En analysant ces mouvements, les auteurs parviennent à en déduire des résultats sur la nature des matériaux des vêtements. Le travail a été appliqué à d'autres objets que les vêtements, et avec succès. La même année, des travaux d'identification de paramètres pour des systèmes non-linéaires ont été publiés par Moradi *et al.* [10], utilisant des résultats issus du domaine du contrôle optimal, et éventuellement adaptables pour le domaine des vêtements.

Du point de vue du travail mené en vision par ordinateur, les travaux actuels sont encore imprécis et balbutiants, que cela soit dans un contexte d'acquisition contrôlé (studio, vêtement seul), ou en situation (vêtements portés par une personne en mouvement). Cependant, aucune méthode de précision utilisant un modèle réaliste de vêtements et les contraintes physiques liées permettant de guider l'estimation, n'existent encore. Parmi les récents travaux remarquables dans ce domaine, on peut citer les travaux de Jinlong *et al.* (avec Jean-Sébastien Franco) [13, 14] de 2016, ceux de Theobalt *et al.* [15] en 2010, Hilton *et al.* [16] en 2014, et Gerard Pons Moll *et al.* en 2015 et 2017 [17, 18, 19, 20].

Malgré la diversité de travaux présents dans l'estimation de paramètres physiques et/ou géométriques de vêtements, les applications des méthodes ainsi issues restent assez restreintes et parfois coûteuses. En effet, certaines méthodes proposées ne sont applicables que dans des conditions particulières, concernant le matériel, les bases de données et les informations initiales nécessaires, le nombre de paramètres physiques déduits, la forme et nature des vêtements gérés, etc. D'autre part, bon nombre des méthodes vues peuvent être qualifiées d'invasives, dans la mesure où l'on doit formater ou préparer le vêtement d'une manière précise, et parfois être amenées à le détrierer pour qu'il puisse être exploité par le système de mesure ou de capture. L'automatisation du processus de détermination des paramètres semble également compromise, compte de tenu du grand nombre de paramètres manuels nécessaires dans la mise en place de ces méthodes.

C'est pourquoi nous nous proposons de travailler sur un système automatisé, non invasif et polyvalent de déduction de paramètres physiques et géométriques de vêtements. Ce stage intervient donc comme un travail d'introduction à ce projet.

## 2.3 Modèles mécaniques et numériques

Dans cette section, nous présenterons succinctement les modèles mécanique et numérique mis en œuvre dans le logiciel Cloth, utilisé pour l'inversion. Nous commencerons par présenter le modèle de coque servant de base pour la modélisation de vêtements, puis le modèle mécanique réellement implémenté. Nous poursuivrons ensuite par l'énoncé de la manière dont la dynamique et les forces sont prises en compte dans le modèle de simulation de vêtements.

### 2.3.1 Modèle de coques élastiques de Koiter

Plusieurs solutions ont été proposées dans la littérature afin de pouvoir modéliser la mécanique d'un vêtement, et le plus fidèlement possible. Le modèle choisi par Romain Casati dans son travail est celui des coques élastiques. En effet, son travail nécessitait un modèle élastique adapté à l'inversion. Plus précisément, cela induit des contraintes par rapport aux forces élastiques, qui doivent dépendre linéairement des degrés de liberté, et par rapport à la matrice de raideur, qui ne doit pas dépendre de la forme au repos du vêtement. Un modèle continu de coques élastiques assez complet et compact est celui de Koiter [21, 22], et c'est celui qui a été retenu par Romain, et présenté succinctement ci-dessous.

Ainsi, une coque est "un solide homogène (ses propriétés sont les mêmes en tout point), isotrope (ses propriétés sont les mêmes dans toutes les directions) et élastique (le solide retrouve toujours la même forme lorsqu'il n'est plus soumis à aucune contrainte). Il est contenu entre deux surfaces d'espacement constant  $2h$ . On appelle surface moyenne la surface espacée de  $h$  des surfaces extérieures ; à elle seule elle permet, dans le modèle de Koiter, de décrire toute la géométrie du solide." On peut décrire géométriquement la surface d'une coque élastique par une application paramétrique  $(s_1, s_2) \in \mathbb{R}^2 \rightarrow r(s) \in \mathbb{R}^3$ , qui est au moins de classe  $\mathcal{C}^2$ , et réalisant une immersion.

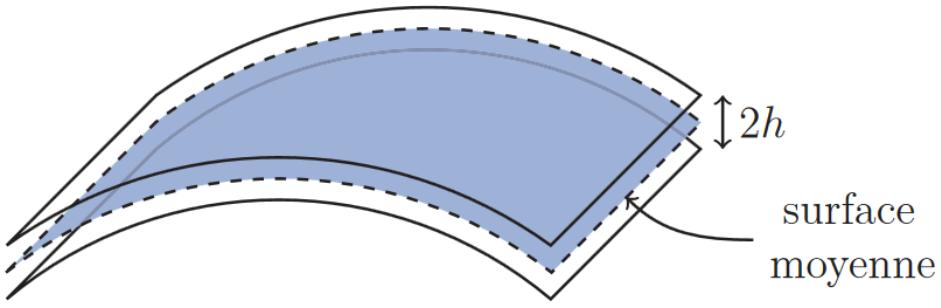


FIGURE 2.5 – Exemple de coque élastique [11]

Selon Ciarlet [21, 22], l'expression de l'énergie interne d'une coque élastique de Koiter peut être séparée en une énergie de membrane et une énergie de flexion. On note la surface moyenne déformée  $r$  et non déformée  $\bar{r}$ , l'énergie interne est alors donnée par :

$$E_{\text{int}}^K(r, \bar{r}) = \underbrace{\frac{1}{2} \int_{\Omega} \frac{h}{4} \|A_r(s) - A_{\bar{r}}(s)\|_{A_{\bar{r}}(s)}^2 dS(r, s)}_{\text{énergie de membrane}} + \underbrace{\frac{1}{2} \int_{\Omega} \frac{h^3}{3} \|B_r(s) - B_{\bar{r}}(s)\|_{A_{\bar{r}}(s)}^2 dS(r, s)}_{\text{énergie de flexion}}$$

En pratique et dans le monde de l'informatique graphique, le modèle de coques élastiques de Koiter n'est pas directement implantable, et doit alors être adapté pour permettre la modélisation effective de vêtements. Ceci est l'object de la prochaine sous-section.

### 2.3.2 Modèle des Discrete Shells

En 1998, Baraff et Witkin [24] proposent un modèle de plaques destiné à l'utilisation d'un schéma d'intégration en temps de type Euler implicite. Les énergies sont construites directement sur modèle direct, qui est en réalité un maillage triangulaire, auquel on associe une version non déformée plane. L'énergie de membrane est inspirée de celle de Koiter. L'énergie de flexion est une version discrétisée de l'énergie de flexion du modèle de Koiter sous l'hypothèse d'une déformation isométrique. Ces simplifications permettent de découpler les termes de flexion et de membrane.

Grinspun *et al.* introduisent en 2003 [25] un modèle de coques élastiques venant étendre celui de Baraff et Witkin [24] : c'est le modèle des *Discrete Shells*. Les énergies sont également construites directement sur le modèle discret (le maillage triangulaire). L'énergie est composée de trois termes : deux termes de membrane  $E_l$  et  $E_a$  et un terme de flexion  $E_\theta$  :

$$E_l = \frac{k_l}{2} \sum_{\text{arêtes } e} \frac{1}{\bar{\ell}_e} (\ell_e - \bar{\ell}_e)^2 \quad \left\{ \begin{array}{l} \ell_e \text{ est la longueur de l'arête } e \\ \end{array} \right\}$$

$$E_a = \frac{k_a}{2} \sum_{\text{faces } f} \frac{1}{\bar{a}_f} (a_f - \bar{a}_f)^2 \quad \left\{ \begin{array}{l} a_f \text{ est l'aire de la face } f \\ \end{array} \right\}$$

$$E_\theta = \frac{k_\theta}{2} \sum_{\substack{\text{arêtes} \\ \text{intérieures}}} e \frac{3\bar{\ell}_e^2}{\bar{A}_e} (\theta_e - \bar{\theta}_e)^2 \quad \left\{ \begin{array}{l} A_e \text{ est la somme des aires} \\ \text{des faces adjacentes à } e \end{array} \right\}$$

L'angle dièdre, formé par les normales, est ici noté  $\theta_e$ , et on note  $k_l$ ,  $k_a$  et  $k_\theta$  les constantes de raideur dépendantes du matériau considéré.

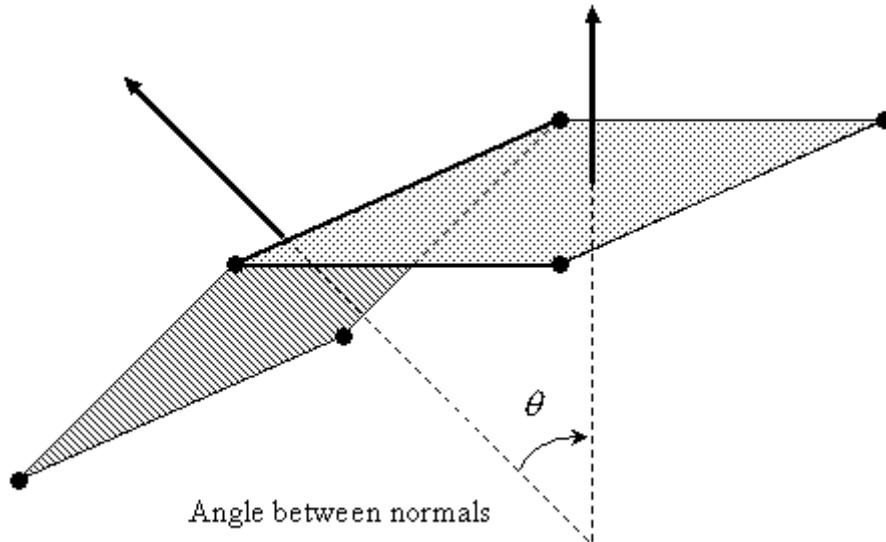


FIGURE 2.6 – Représentation visuelle de l'angle dièdre [107]

La simplicité des termes d'énergie en font un modèle assez facile à implémenter. Cependant, les dérivations doivent être faites de manière algorithmique, par calcul formel, ou à la compilation en utilisant la différentiation automatique, comme cela est fait et détaillé dans [23].

### 2.3.3 Modèle de simulation – Modèle d'intégration

Dans le logiciel Cloth, une des fonctions les plus importantes est le calcul de la nouvelle position du vêtement après un pas de temps. Cela prend en compte la position actuelle du vêtement, sa vitesse, son énergie interne, les forces dépendant de sa position/vitesse, sa masse, et le modèle de simulation. Le modèle d'intégration implémenté dans Cloth est celui proposé par Baraff et Witkin [24].

La dynamique du système est gouvernée par l'équation différentielle ordinaire (principe fondamental de la dynamique) :

$$\ddot{\mathbf{x}} = \mathbf{M}^{-1} \mathbf{f}(\mathbf{x}, \dot{\mathbf{x}}) \quad (2.1)$$

avec  $\mathbf{x}$  représentant l'état géométrique du vêtement,  $\dot{\mathbf{x}}$  sa vitesse et  $\ddot{\mathbf{x}}$  son accélération,  $\mathbf{M}$  la distribution de masse du vêtement et  $\mathbf{f}$  (dépendant de  $\mathbf{x}$  et  $\dot{\mathbf{x}}$ ) représentant les forces internes et les autres forces agissant sur le vêtement.

À partir de la position  $\mathbf{x}_0 = \mathbf{x}(t_0)$  et de la vitesse  $\mathbf{v}_0 = \mathbf{v}(t_0) = \dot{\mathbf{x}}(t_0)$  du système au temps  $t_0$  le but est de déterminer la nouvelle position  $\mathbf{x}(t_0 + h)$  et la vitesse  $\mathbf{v}(t_0 + h)$  du système au temps  $t_0 + h$ , avec  $h = dt$  le pas de temps.

En posant  $\Delta\mathbf{x} = \mathbf{x}(t_0 + h) - \mathbf{x}(t_0)$  et  $\Delta\dot{\mathbf{x}} = \Delta\mathbf{v} = \mathbf{v}(t_0 + h) - \mathbf{v}(t_0)$ , et en utilisant la méthode implicite (backward) d'Euler<sup>2</sup>, on peut approximer  $\Delta\mathbf{x}$  et  $\Delta\mathbf{v}$  par :

$$\begin{pmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{v} \end{pmatrix} = h \begin{pmatrix} \mathbf{v}_0 + \Delta\mathbf{v} \\ \mathbf{M}^{-1}\mathbf{f}(\mathbf{x}_0 + \Delta\mathbf{x}, \mathbf{v}_0 + \Delta\mathbf{v}) \end{pmatrix} \quad (2.2)$$

Un développement de Taylor au premier ordre nous permet d'obtenir l'approximation suivante :

$$\mathbf{f}(\mathbf{x}_0 + \Delta\mathbf{x}, \mathbf{v}_0 + \Delta\mathbf{v}) \approx \mathbf{f}(\mathbf{x}_0, \mathbf{v}_0) + \frac{\partial\mathbf{f}}{\partial\mathbf{x}}(\mathbf{x}_0, \mathbf{v}_0)\Delta\mathbf{x} + \frac{\partial\mathbf{f}}{\partial\mathbf{v}}(\mathbf{x}_0, \mathbf{v}_0)\Delta\mathbf{v} \quad (2.3)$$

En posant  $\mathbf{f}_0 = \mathbf{f}(\mathbf{x}_0, \mathbf{v}_0)$ , en précisant que les dérivées partielles sont évaluées en  $(\mathbf{x}_0, \mathbf{v}_0)$ , et en substituant  $\Delta\mathbf{x}$  par  $h(\mathbf{v}_0 + \Delta\mathbf{v})$  dans l'équation 2.2 on obtient :

$$\Delta\mathbf{v} = h\mathbf{M}^{-1}\left(\mathbf{f}_0 + \frac{\partial\mathbf{f}}{\partial\mathbf{x}}h(\mathbf{v}_0 + \Delta\mathbf{v}) + \frac{\partial\mathbf{f}}{\partial\mathbf{v}}\Delta\mathbf{v}\right) \quad (2.4)$$

Cela nous amène donc à résoudre pour  $\Delta\mathbf{v}$  l'équation suivante :

$$\left(\mathbf{M} - h\frac{\partial\mathbf{f}}{\partial\mathbf{v}} - h^2\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\right)\Delta\mathbf{v} = h\left(\mathbf{f}_0 + h\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\mathbf{v}_0\right) \quad (2.5)$$

Une fois  $\Delta\mathbf{v}$  trouvé, on peut en déduire  $\Delta\mathbf{x} = h(\mathbf{v}_0 + \Delta\mathbf{v})$ , et donc déterminer la nouvelle position  $\mathbf{x}(t_0 + h)$  et vitesse  $\mathbf{v}(t_0 + h)$ . La principale difficulté dans la détermination de  $\Delta\mathbf{v}$  est l'évaluation de  $\mathbf{f}_0$  et le calcul effectif des dérivées partielles  $\frac{\partial\mathbf{f}}{\partial\mathbf{v}}$  et  $\frac{\partial\mathbf{f}}{\partial\mathbf{x}}$ . Ce calcul est effectué en pratique en utilisant des outils de différentiation automatique, tout en considérant les énergies et contraintes comme des matrices et fonctions creuses.

## 2.4 Formulation mathématique du problème

### 2.4.1 Notations et paramétrisation du problème

Dans cette sous-section nous nous proposons de présenter les notations utilisées tout au long de notre étude, ainsi que la nature des objets considérés pour l'inversion.

Tout d'abord, un maillage 3D est noté par la variable  $x$  et est considéré comme un vecteur de  $\mathbb{R}^{3n_v}$  avec  $n_v$  le nombre de sommets (vertices) du maillage, chaque sommet étant un point de  $\mathbb{R}^3$ . Le vecteur  $x$  représentant le maillage est donc une concaténation de toutes les coordonnées de ses sommets : il modélise la position du maillage dans l'espace. Le maillage d'un vêtement étant amené à évoluer, on notera alors  $x(t_i)$  le maillage au temps  $t_i$  du vêtement considéré.

De la même manière, le vecteur vitesse du maillage au temps  $t_i$  sera noté  $\dot{x}(t_i)$ , et sera également une concaténation des vecteurs vitesse pour chaque sommet du maillage. On note  $\bar{x}$  le vecteur de  $\mathbb{R}^{3n_v}$  représentant la forme du vêtement au repos.

2. En mathématiques, la méthode d'Euler est une procédure numérique pour résoudre par approximation des équations différentielles du premier ordre avec une condition initiale. La méthode implicite concerne la nouvelle approximation  $y_{k+1}$  qui apparaît des deux côtés de l'équation et donc la méthode doit résoudre une équation algébrique pour inconnue  $y_{k+1}$  (le pas suivant).

Afin de pouvoir appliquer nos méthodes de résolution pour un problème d'inversion formulé comme un problème d'optimisation, nous devons disposer de maillages représentant des vêtements afin de mettre en place des exemples et cas d'applications (voir 2.4.3 pour les cas considérés pour l'inversion dynamique). Nous avons restreint notre choix sur deux maillages en particulier : l'un est très simple, l'autre un peu plus complexe.

Le premier maillage est uniquement composé de quatre sommets formant deux triangles. On fixe alors deux sommets diagonaux opposés, laissant les deux autres libres. L'avantage de ce premier maillage est qu'il permet d'obtenir une formulation explicite de la solution théorique, du fait de la simplicité du maillage. Le second maillage modélise un morceau rectangulaire de vêtement, pouvant être vu comme un drap, composé de seize sommets. Le maillage initialement modélisé comme une grille quadrillé est triangulaire pour permettre plus de flexibilité au mouvement du vêtement. Dans les figures qui suivent, les sommets en rouge correspondent aux sommets fixés (encastrés) tandis que les sommets en noir sont libres.

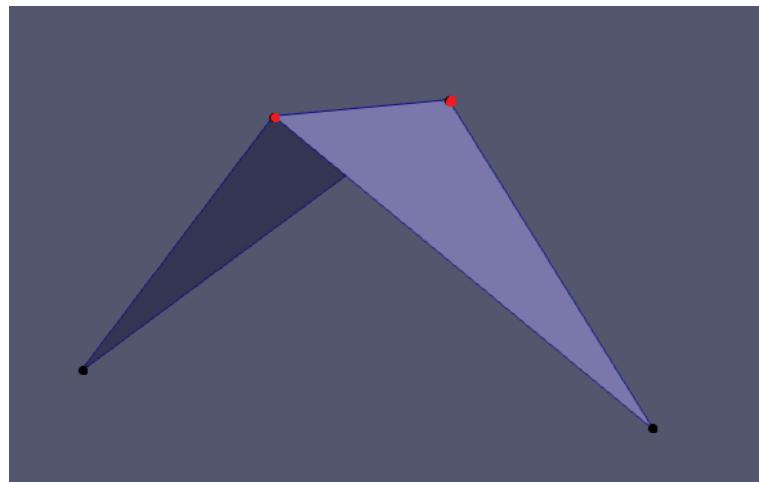


FIGURE 2.7 – Représentation du maillage – Bitriangle

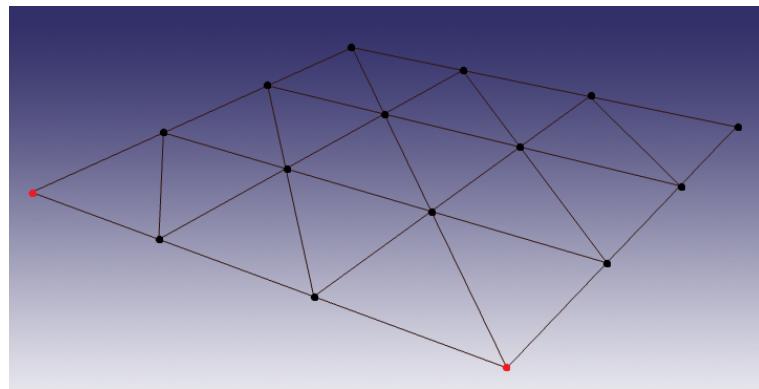


FIGURE 2.8 – Représentation du maillage – Drap 16 sommets

## 2.4.2 Formulation du problème d'inversion statique

De manière simplifiée, le problème d'inversion statique de vêtements traité par Romain Casati [11] consiste, à partir de la donnée de la forme du vêtement à l'équilibre des forces (équilibre statique), à déterminer la forme au repos du vêtement. On dispose d'un simulateur permettant, à partir de la forme au repos et de la position initiale (ici choisie comme étant également la forme au repos) de nous donner la forme du vêtement à l'équilibre, notée  $x_{eq}$ .

On cherche donc à trouver la forme au repos  $\bar{x}$  qui, après simulation d'affaissement sous gravité, donne une forme à l'équilibre  $x$  la plus proche possible de la vraie position à l'équilibre

(*target*). La seule information ici faisant office d'observation est la réelle forme à l'équilibre du vêtement, et est donc utilisée afin de mettre en place le problème d'optimisation lié à la détermination de la forme au repos la plus convenable.

La fonction objectif à minimiser s'écrit avec une formulation aux moindres carrés comme suit :

$$J(\bar{x}) = \frac{1}{2} \|\phi(\bar{x}) - x_{eq}\|^2 \quad (2.6)$$

avec  $\phi$  la fonction associant à la forme au repos  $\bar{x}$  une position à l'équilibre  $\phi(\bar{x}) = x$  possible (car la fonction  $\phi$  n'est ni injective ni subjective : plusieurs équilibres sont possibles pour une seule forme au repos, plusieurs formes au repos peuvent mener à la même position à l'équilibre).

### 2.4.3 Passage au cas dynamique

Dans le cas dynamique, on étudie une séquence de mouvements d'un vêtement, et l'on cherche à partir de cette observation à déterminer des informations sur les paramètres d'instanciation du simulateur. Ainsi, l'observation dans le cas dynamique des positions successives du maillage de vêtement correspond à l'observation dans le cas statique à la position du vêtement à l'équilibre des forces. Ces observations vont donc être l'information principale utilisée pour définir notre fonction objectif, le but étant de trouver les paramètres nécessaires à la reproduction, la plus proche fidèle possible, du mouvement observé précédemment.

Le but sur le long terme est d'utiliser les techniques de vision par ordinateur pour effectuer ces observations sur des mouvements de vêtements réels. Néanmoins, dans le cadre du stage, les observations proviendront exclusivement du simulateur Cloth (voir 2.1.2).

On réalise alors une simulation d'affaissement du vêtement choisi. On note  $x(t)$  le vecteur position du vêtement à l'instant  $t$  de la simulation. On note  $\dot{x}(t)$  le vecteur de vitesse du vêtement à l'instant  $t$  de la simulation. Ces deux données de position et de vitesse à l'instant 0 définissent les conditions initiales de la simulation. Usuellement, et dans notre cas, on pose :

$$\text{C.I. : } \begin{cases} x(0) = \bar{x} \\ \dot{x}(0) = 0 \end{cases} \quad (2.7)$$

On définit ensuite un intervalle  $[T_0, T_N]$  comme fenêtre temporelle d'observation, ce qui nous permet d'avoir une séquence de positions  $\{X_{obs}(T_0), \dots, X_{obs}(T_N)\}$  formant notre observation.

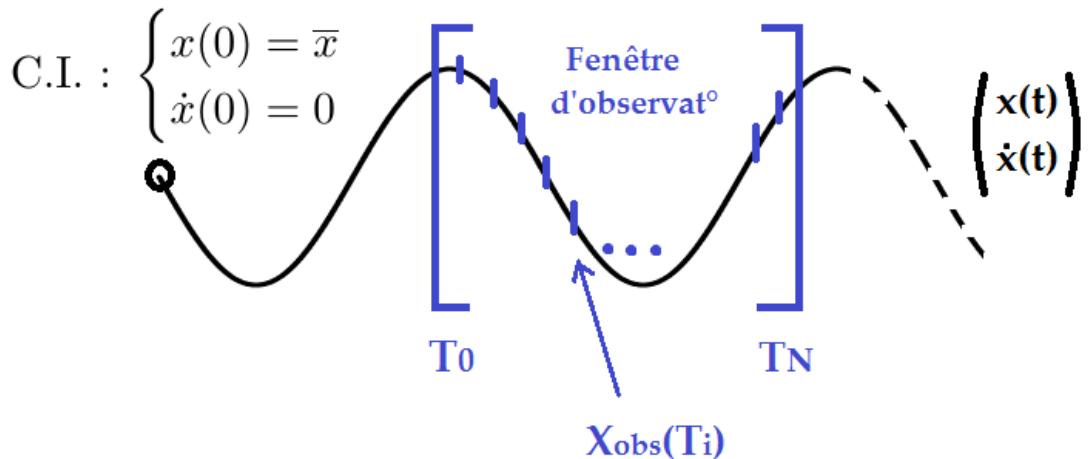


FIGURE 2.9 – Schéma récapitulatif du problème en dynamique

On pose alors  $\phi$  comme la fonction représentant le lancement d'une simulation. Une simulation nécessite la donnée de  $\bar{x}$  la forme au repos,  $x(0)$  la position initiale,  $p$  les paramètres

physiques du vêtement et  $\dot{x}(0)$  la vitesse initiale du vêtement en chaque sommet. L'application de  $\phi$  nous donne donc une séquence de positions.

Afin de résoudre ce problème, nous avons dû poser un certain nombre de simplification pour s'abstraire des difficultés réelles. On suppose dans notre configuration actuelle que la simulation (modèle) et que l'observation sont parfaits et sans erreurs. En réalité, il y a une certaine erreur de mesure notamment car l'observation n'est pas parfaite, et il en va de même pour le modèle qui est une approximation des réels phénomènes physiques. On pourrait alors considérer les matrices de covariance d'erreur pour le modèle et la mesure. On suppose également que le laps de temps d'observation, compris entre  $T_0$  et  $T_N$  est connu et peut être situé temporellement par rapport à la simulation initiale du vêtement. Or dans le cas réel, lorsque nous obtiendrons une observation, on ne saura pas forcément à quelle moment elle correspond par rapport à tout le mouvement de l'objet dans le temps. Une remarque du même type peut être faite en ce qui concerne la fréquence d'acquisition et de simulation qui seront différentes, et qu'il faudra connaître pour faire le lien entre nombre de frame observées, fréquence d'acquisition, temps total d'acquisition. Dans notre cas, on choisira un pas de temps d'observation similaire au pas de temps de simulation.

Face au grand nombre de variables nécessaires à l'instanciation du simulateur, tenter de résoudre directement le problème d'inversion en supposant tous les paramètres inconnus, et sans aucun a priori sur le vêtement ou son mouvement, peut s'avérer très difficile. C'est pourquoi nous choisissons de traiter différents cas mettant uniquement en jeu une (voire deux) inconnues à optimiser, afin d'avoir une idée claire des performances des nos algorithmes pour l'inversion dynamique des vêtements pour une inconnue donnée. De plus, pour chaque cas, on considérera plusieurs placements possibles de la fenêtre d'observation, afin d'étudier en parallèle l'influence de la nature du mouvement observé sur l'information déduite et réutilisée pour l'inversion. En pratique, on choisira 4 fenêtres d'observations assez variées, définissant l'ensemble suivant (l'unité est le pas de temps de simulation) :

$$[T_0, T_N] \in \left\{ [1, 20], [5, 25], [25, 50], [200, 280] \right\} \quad (2.8)$$

### Cas 1 - Recherche de la position initiale $x(0)$

Le premier de cas de figure concerne la recherche de la position initiale de la simulation  $x(0)$ . Dans notre cas, cela correspond en effet à  $\bar{x}$ , mais cela est dû à notre choix antérieur pour les conditions initiales.

On suppose connaître la forme au repos  $\bar{x}$ , la vitesse initiale  $\dot{x}(0) = 0$ , et on suppose également savoir positionner temporellement la fenêtre d'observation sur la simulation initiale, c'est à dire connaître la donnée  $T_0$  et  $T_N$ . Pour la recherche de solutions, on initialise  $x(0)$  à  $X_{obs}(T_0)$ .

On définit alors la fonction objectif à minimiser  $\mathcal{J}_1$  comme une formulation aux moindres carrés comme suit :

$$\mathcal{J}_1(x(0)) = \sum_{t=T_0}^{T_N} \| \phi(x(0))(t) - X_{obs}(t) \|^2 \quad (2.9)$$

### Cas 2 - Recherche de la vitesse en début d'observation $\dot{x}(T_0)$

Le second cas de figure concerne la recherche du vecteur vitesse du vêtement au temps  $T_0$ , c'est à dire au début de l'observation. En connaissant la position du vêtement au temps  $T_0$  et en connaissant la forme au repos  $\bar{x}$ , on veut retrouver la vitesse "initiale à l'observation" nous permettant de pouvoir retrouver le mouvement observé. L'inconnue recherchée est donc  $\dot{x}(T_0)$ .

On suppose donc connaître la forme au repos  $\bar{x}$ , la position au début de l'observation  $x(T_0) = X_{obs}(T_0)$ , et on suppose également savoir positionner temporellement la fenêtre d'observation sur

la simulation initiale, c'est à dire connaitre la donnée  $T_0$  et  $T_N$ . Pour la recherche de solutions, on initialise  $\dot{x}(T_0)$  à 0.

On définit alors la fonction objectif à minimiser  $\mathcal{J}_2$  comme une formulation aux moindres carrés comme suit :

$$\mathcal{J}_2(\dot{x}(T_0)) = \sum_{t=T_0}^{T_N} \| \phi(\dot{x}(T_0))(t) - X_{obs}(t) \|^2 \quad (2.10)$$

### Cas 3 - Recherche de la forme au repos $\bar{x}$

Le troisième cas de figure concerne la recherche de la forme au repos du vêtement  $\bar{x}$ .

On suppose connaître la position initiale  $x(0)$  de la simulation ainsi que la vitesse initiale  $\dot{x}(0)$ , et on suppose également savoir positionner temporellement la fenêtre d'observation sur la simulation initiale, c'est à dire connaitre la donnée  $T_0$  et  $T_N$ . Pour la recherche de solutions, on initialise  $\bar{x}$  avec  $X_{obs}(T_0)$ .

On définit alors la fonction objectif à minimiser  $\mathcal{J}_3$  comme une formulation aux moindres carrés comme suit :

$$\mathcal{J}_3(\bar{x}) = \sum_{t=T_0}^{T_N} \| \phi(\bar{x})(t) - X_{obs}(t) \|^2 \quad (2.11)$$

### Cas 4 - Recherche de la forme au repos $\bar{x} = x(0)$

Le quatrième cas de figure concerne la recherche de la forme au repos du vêtement  $\bar{x}$ , en sachant que la position initiale de la simulation est  $x(0) = \bar{x}$ .

On suppose connaître la vitesse initiale  $\dot{x}(0)$ , et on suppose également savoir positionner temporellement la fenêtre d'observation sur la simulation initiale, c'est à dire connaitre la donnée  $T_0$  et  $T_N$ . Pour la recherche de solutions, on initialise  $x(0) = \bar{x}$  avec  $X_{obs}(T_0)$ .

On définit alors la fonction objectif à minimiser  $\mathcal{J}_4$  comme une formulation aux moindres carrés comme suit :

$$\mathcal{J}_4(x(0) = \bar{x}) = \sum_{t=T_0}^{T_N} \| \phi(x(0) = \bar{x})(t) - X_{obs}(t) \|^2 \quad (2.12)$$

### Cas 5 - Recherche de la forme au repos $\bar{x} = x(0)$ sans répartition massique connue

Le cas 5 est très similaire au cas 4, dans la mesure où l'on utilise l'information que la position initiale est aussi la forme au repos qui est notre inconnue. Cependant, dans le cas 5, la répartition nodale de la masse globale du vêtement est supposée non-connue. En effet, à chaque fois que l'on attribue une forme au repos et une densité de masse à un vêtement de la classe *Cloth*, le logiciel recalcule automatiquement la répartition de la masse à chaque noeud du vêtement. Jusqu'à présent, dans tous les cas précédents, cette répartition était réalisée à partir de la vraie masse (les paramètres physiques sont tous connus) et à partir de la vraie forme au repos également, même si  $\bar{x}$  et l'inconnue, comme cela est le cas dans le cas 3 et 4. Dans le cas 5, la répartition nodale de la masse se fait à partir de la solution  $\bar{x}^*$  proposée à chaque évaluation de la fonction objectif.

## 2.5 Méthodes d'optimisation sans gradient

La première approche de résolution du problème d'optimisation, correspondant à l'inversion dynamique de vêtements, est celle des méthodes d'optimisation sans gradient. En effet, la facilité d'implémentation de certaines méthodes non basées sur le calcul du gradient permet de pouvoir rapidement disposer de résultats, et de les analyser pour mieux comprendre le problèmes et les éventuelles difficultés liés à l'inversion. La mise en place du calcul du gradient de la fonction objectif n'est pas une mince affaire, et nécessite un travail d'expression mathématique et d'implémentation avant de pouvoir envisager une première tentative d'optimisation de la fonction objectif : disposer d'algorithmes permettant d'outrepasser cette contrainte nous a incité à utiliser en priorité ces derniers.

D'autre part, ces méthodes sont assez présentes comme méthodes de résolution dans l'état de l'art, et nous voulions alors implémenter à notre tour ce type de méthodes pour notre problème, afin d'avoir une meilleure idée de l'efficacité de ces méthodes pour nos problèmes d'optimisation liés à l'inversion. Elles permettent enfin, pour la plupart, d'obtenir un minimum global de la fonction à minimiser, ce qui n'est pas toujours le cas des méthodes d'optimisation avec gradient.

Dans cette section nous vous présenterons donc un état de l'art non exhaustif des méthodes d'optimisation sans gradient connues dans la littérature. Par la suite, nous préciserons le principe de recherche de l'optimum pour 4 d'entre elles sélectionnées. Nous présenterons alors la manière dont elles ont été implémentées, et les tests relatifs à la précision et l'efficacité de ces méthodes pour des problèmes de différentes complexités.

### 2.5.1 Etat de l'art non exhaustif

Lorsque l'on cherche à minimiser une fonction objectif dans le cadre d'un problème d'optimisation, il est important de déterminer la nature de notre problème et de choisir la méthode d'optimisation qui convient le mieux pour la résolution. Ainsi, une méthode basée sur une recherche aléatoire risque de ne pas être le meilleure choix pour une fonction linéaire ou convexe. Il est donc important de sélectionner la bonne méthode, en étudiant la présence ou non de minima locaux, les taux de convergence de différentes méthodes, le nombre de variables du problèmes (grand ou petit), et d'autres aspects permettant de se diriger vers certaines méthodes. On se propose donc d'établir un état de l'art des méthodes de minimisation afin de prendre compte des différentes caractéristiques de chacunes et de porter notre attention sur les plus intéressantes au vu de notre problème.

Durant la phase de recherche dans la littérature des méthodes d'optimisation sans gradient, nous avons pu trouver dix-sept algorithmes d'optimisation. Une grande partie d'entre eux sont des méthodes de descente, c'est à dire que le prochain état est calculé à partir du précédent, auquel on ajoute un vecteur de descente, caractérisé par un pas et une direction :

$$x_{k+1} = x_k + \alpha_k d_k \quad (2.13)$$

avec  $\alpha_k$  le pas de descente, et  $d_k$  la direction de descente.

Une grande partie des méthodes d'optimisation sans gradient appartiennent à la famille des métaheuristiques. Les métaheuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global, se comportant comme des algorithmes de recherche tentant d'apprendre les caractéristiques d'un problème afin d'en trouver une approximation de la meilleure solution. Ces méthodes utilisent cependant un haut niveau d'abstraction, leur permettant d'être adaptées à une large gamme de problèmes différents.

Certains méthodes s'inspirent du fonctionnement et l'organisation de la nature, notamment des animaux. Ainsi, l'algorithme de colonie de fourmis [44, 45] est généralement utilisé pour optimiser un trajet d'un point A à un point B, en terme de distance et de ressources nécessaires, en prenant compte l'environnement. De plus, un algorithme assez répandu dans la recherche

d'optimums globaux est celui de l'optimisation par essaim particulaire [54, 42, 55, 56, 57]. La méthode Kangourou [58, 59], quant à elle, cherche une solution par déplacements successifs minimisant la fonction J dans un voisinage de la solution courante : elle combine descente et saut.

Un autre algorithme combinant méthodes gloutonnes et aléatoires et l'algorithme GRASP, pour Greedy randomized adaptive search procedure [52, 53]. Souvent utilisé comme exemple d'implémentation dans les cours d'initiation à l'optimisation, l'algorithme du simplexe (ou Nelder-Mead pour le cas non-linéaire) [35, 36, 37, 38] est basé sur une évolution géométrique des sommets (par symétrie, expansion, etc.), qui sont les états actuel, pour converger vers la meilleure solution.

Le critère de Métropolis, considéré comme une méthode de type Monte-Carlo par chaînes de Markov, a inspiré quelques autres algorithmes d'optimisation. La méthode du récuit simulé [39, 40, 41, 42], mettant en pratique ce critère, est issue de la métallurgie et basée sur l'alternance de cycle de refroidissement et de réchauffage pour minimiser l'énergie d'un matériau. La méthode du tunnel stochastique [61] est aussi un exemple d'utilisation du critère de Métropolis.

Un certain nombre de méthodes proviennent également du domaine de l'étude sociale et de l'évolution, que l'on appelle souvent méthodes évolutionnaires. Les stratégies d'évolution [48] considèrent une population composée de parents et s'intéresse à la production d'enfant par combinaison de parents, chaque individu représentant en fait une solution possible. L'algorithme génétique [38, 43, 39] constitue une des plus célèbres implémentations de ce principe, le but étant de faire évoluer la population vers la meilleure solution par des opérations de croisement, mutation et sélection.

Certaines méthodes sont principalement basées sur une recherche aléatoire de la solution. En effet, la méthode de la promenade aléatoire [39] agit à la fois sur le pas de descente et sur sa direction, en choisissant une direction aléatoire de descente, et en réduisant le pas au fut et à mesure que l'on se rapproche de la solution. Dans le même principe, on retrouve la méthode des sauts aléatoires [39] qui, comme son nom l'indique, fait évoluer la solution actuelle en sautant aléatoirement d'une position à l'autre de l'espace de recherche. La recherche tabou [49, 42, 50, 51] suit principalement le même principe de recherche que les autres méthodes aléatoires. Cependant, il y a une interdiction de revenir sur les dernières positions explorées, d'où son nom de "tabou".

La méthode de la section dorée [32, 33] construit une succession d'intervalles qui contiennent le minimum recherché, et réduit la taille de l'intervalle d'un rapport égal au nombre d'or à chaque itération. La méthode d'interpolation parabolique [34] consiste à remplacer la fonction à optimiser par son polynôme d'interpolation d'ordre deux (parabolique) en trois points initiaux, et de mettre à jour ces points en fonction de la position du minimum théorique. Par ailleurs, la méthode univariable alternée [35] consiste à minimiser la fonction selon une seule variable en fixant les autres variables, et ainsi de suite en changeant la variable considérée. Enfin, la dernière méthode rencontrée, méthode du bruitage [60], consiste à ne pas considérer directement la fonction à optimiser, mais à la perturber en ajoutant du bruit aléatoire. Au fur et à mesure que la méthode progresse (méthode de descente), l'amplitude du bruit décroît.

### 2.5.2 Méthodes implémentées

Dans cette sous-section nous présenterons les méthodes d'optimisation sans gradient implémentées en Python, leur fonctionnement et leur particularités.

#### Optimisation par essaim particulaire

Le principe général est qu'un ensemble d'invidus faibles peuvent engendrer en communauté un force considérable. On considère alors un ensemble de particules appelé essaim, où chaque particule ou individu représente une solution possible du problème. Une particule est caractérisée

par sa position (un point dans l'espace de recherche), son vecteur vitesse actuel (déplacement dans l'espace des solutions) et d'un voisinage. Le voisinage peut être géométrique et donc évoluer au cours de l'optimisation, ou être un voisinage statique défini au début de l'optimisation. On s'intéressera également, pour chaque particule, à sa meilleure position rencontrée et à la meilleure position rencontrée par son voisinage. On entend ici par "meilleure position", celle qui minimise le plus la fonction objectif.

À l'aide de règles de déplacement très simples (dans l'espace des solutions), les particules peuvent converger progressivement vers un minimum global. Au début de l'algorithme, chaque particule est positionnée (aléatoirement ou uniformément) dans l'espace de recherche du problème.

Chaque itération fait bouger les particules en fonction de 3 paramètres : sa vitesse actuelle  $V_k$ , sa meilleure position  $P_i$ , et la meilleure solution obtenue dans son voisinage  $P_g$ . L'expression des équations de mouvement est donnée par :

$$V_{k+1} = \omega \cdot V_k + b_1(P_i - X_k) + b_2(P_g - X_k)$$

$$X_{k+1} = X_k + V_{k+1}$$

avec  $\omega$  l'intertie de la particule, ou la tendance à être individualiste (suivre sa propre route),  $b_1$  la tendance à revenir vers sa meilleure solution,  $b_2$  la tendance à se rapprocher de la solution collective. On choisit généralement la même intertie pour toutes les particules, et  $b_1$  et  $b_2$  sont choisis uniformément et de manière aléatoire dans un intervalle donné, et ce, soit à chaque itération de la méthode ou uniquement à l'initialisation.

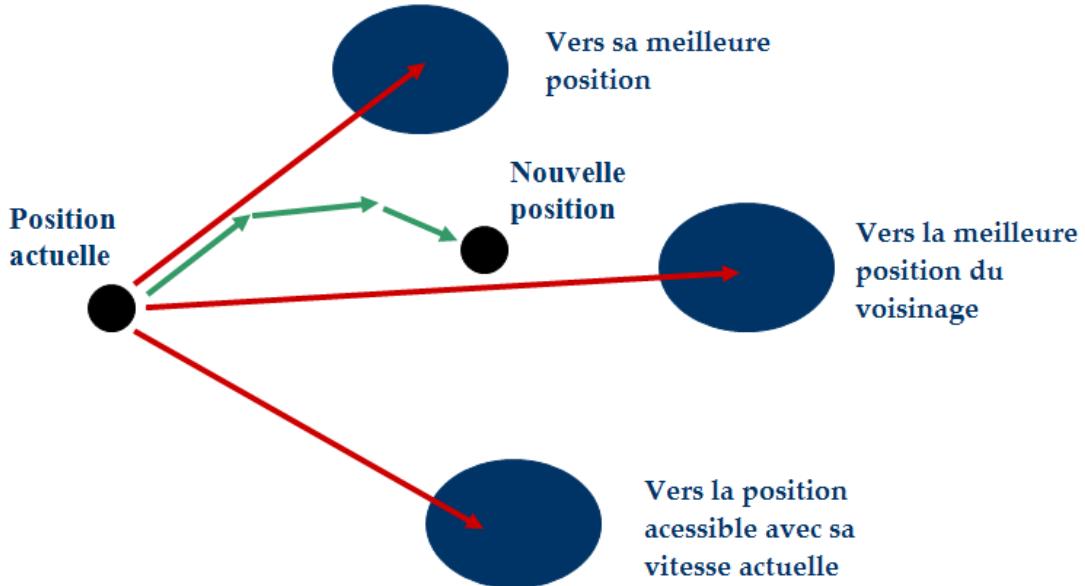


FIGURE 2.10 – Déplacement d'une particule [56]

En fonction du problème à résoudre, et de la nature de la fonction objectif à minimiser, un certain nombre de paramètres de la méthode devront être optimisés, le plus souvent manuellement, pour répondre au mieux au problème. Ainsi, la taille du voisinage peut jouer un rôle dans la convergence de la méthode. L'intertie et les tendances individualistes et collectives de la particule sont aussi des paramètres à régler. Enfin, la taille de l'essaim (nombre de particules) et l'initialisation des particules (en prenant en compte les *a priori* éventuels sur le problème) sont des facteurs prépondérants dans la convergence et la vitesse de cette dernière.

Afin d'implémenter cette méthode en Python, une classe Particule a été créée pour modéliser le comportement d'une particule. Cette classe dispose d'un certain nombre d'attributs dont : sa position actuelle, l'évaluation de la fonction objectif à cette position, sa meilleure position atteinte, l'évaluation de la fonction objectif à la meilleure position, son intertie, son vecteur

vitesse, son voisinage, la meilleure solution de son voisinage, l'indice de la particule dans la liste de particules. La classe regroupe aussi des attributs globaux partagés par toutes les particules dont : la liste de toutes les particules (essaim), le nombre de particules dans l'essaim, la meilleure solution de l'essaim et l'évaluation de la fonction objectif correspondante, le nombre de variables pour la fonction objectif, la position initiale utilisée comme centre de l'intervalle d'initialisation de chaque particule et le nombre d'évaluation de la fonction objectif depuis le début de l'optimisation.

### Algorithmé génétique

Au début de l'algorithme génétique, on dispose d'une population initiale aléatoire. Chaque individu représente une solution possible, un point dans l'espace de recherche. Le principe de l'algorithme génétique est basé sur le principe d'évolution : la population doit évoluer pour se rapprocher de la meilleure solution. Les "gènes" d'un individu vont alors étre sa position (point à  $n$  coordonées dans l'espace de recherche par exemple), et vont alors évoluer selon trois étapes pour faire émerger la nouvelle génération de population : une itération de l'algorithme correspond à produire une nouvelle génération.

La première étape du principe d'évolution est ce que l'on appelle la sélection (en référence à la "sélection naturelle"). Le principe est de garder une certaine proportion des "meilleurs" individus (qui minimisent le mieux la fonction objectif), et de laisser une chance au reste de la population d'être sélectionnée selon une probabilité aléatoire. Le fait d'avoir une sélection aléatoire de "mauvais" individus peut permettre de sortir d'un minimum local dans lequel se seraient éventuellement engagés les "meilleurs" individus.

Après l'étape de sélection vient l'étape de mutation. Une chance de muter est attribuée à tous les individus de la population, et par un tirage aléatoire uniforme, on décide quel individu sera muté ou pas. L'opération de mutation peut prendre différentes formes ; tout dépend de la nature d'encodage de la solution et des états de l'espace de recherche. Ainsi, si l'on travaille avec des mots, une mutation possible peut être un changement aléatoire de certaines lettres du mot. Si l'on travaille cette fois avec des points de l'espace, la mutation peut être une translation ou une rotation selon un axe choisi aléatoirement. C'est à celui qui implémente l'algorithme de la choisir en fonction du problème à résoudre. Dans notre cas, une solution est un point dans un espace de recherche de type  $\mathbb{R}^n$ , et l'opération de mutation choisie consiste à réaliser un écart aléatoire et différent sur chaque coordonnée du point.

Enfin, vient l'étape de croisement des individus, parfois appelée étape de reproduction. Elle consiste en la combinaison de deux individus, appelés parents, pour générer un nouvel individu, appelé enfant. Comme pour la mutation, c'est une opération qui peut prendre différentes formes, selon la nature des données et du problème. Dans notre cas, l'opération de croisement choisie consiste à générer le point milieu entre les deux points parents.

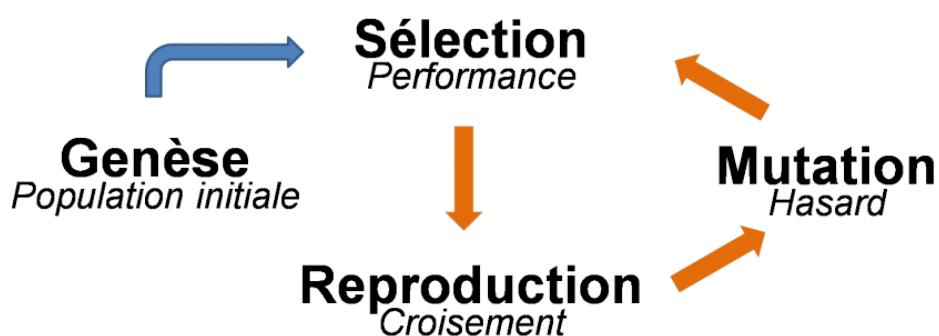


FIGURE 2.11 – Fonctionnement global de l'algorithme génétique [108]

Un certain nombre de paramètres de l'algorithme restent néanmoins à choisir en fonction du problème à résoudre. Encore une fois, la taille de population et l'initialisation des individus être un paramètre déterminant en ce qui concerne la vitesse et la précision de convergence de l'algorithme. La probabilité de muter est aussi un paramètre à choisir convenablement, tout comme la proportion de "meilleurs" sélectionnés et la probabilité de sélection d'un "mauvais" individu.

### Méthode de la promenade aléatoire

La méthode de la promenade aléatoire "joue" à la fois sur le pas de descente et sur la direction de descente. On initialise l'algorithme en choisissant une solution initiale  $x_0$ , on initialise le compteur d'itération  $k = 0$  et le compteur de recherche  $cc = 1$ , et on fixe un pas de descente initial  $\alpha_0$ .

La direction de descente est choisie de manière aléatoire. Pour ce faire, on génère  $\{r_i\}$ ,  $n$  variables aléatoires uniformément distribuées sur  $[-1, 1]$ . Soit  $R = \sqrt{r_1^2 + \dots + r_n^2}$ . Si  $R > 1$  on régénère les variables  $\{r_i\}$ , sinon on choisit la direction de descente ainsi formée :

$$d_k = \frac{1}{R} \begin{pmatrix} r_1 \\ \vdots \\ r_n \end{pmatrix} \quad (2.14)$$

On en déduit alors un nouvel état potentiel à partir de l'état précédent :

$$x_{k+1} = x_k + \alpha_k d_k \quad (2.15)$$

Si ce nouvel état est meilleur en terme de minimisation, c'est à dire  $J(x_{k+1}) < J(x_k)$ , alors on garde cet état et on passe à l'itération suivante. Les compteurs seront mis à jour, de telle sorte que  $k = k + 1$  et  $cc = 1$ , et on va générer une nouvelle direction de descente.

Si ce nouvel état n'est pas meilleur, et que le compteur  $cc$  n'est pas à sa valeur maximale, on génère une nouvelle direction de descente, et on met à jour le compteur de recherche  $cc = cc + 1$ . Si le compteur a dépassé sa valeur maximale, c'est que l'on n'a trouvé aucune solution en changeant  $cc_{max}$  fois de directions. On va alors réduire le pas de descente actuel de moitié pour affiner la recherche, d'où  $\alpha_k = \frac{\alpha_k}{2}$ , et  $cc = 1$ .

On répète alors tout ce processus jusqu'à ce que le pas de descente soit inférieur à un  $\varepsilon$  que l'on précisera.

Cette méthode, facile à implémenter, permet de traiter des problèmes d'optimisation de natures et difficultés variées. La convergence de la méthode n'est pas assurée, du fait de l'aspect très aléatoire de la méthode. Néanmoins, par un choix judicieux de la position initiale  $x_0$ , et du pas initial, la performance et l'efficacité de l'algorithme peuvent rapidement s'améliorer. Le nombre d'itération et la limite du compte maximum de recherche (avant de réduire le pas de descente) sont également des paramètres à prendre en compte pour une implémentation plus appropriée à certains problèmes.

### Méthode du recuit simulé

Le recuit simulé est une méthode issue de la métallurgie, basée sur l'alternance de cycle de refroidissement et de réchauffage pour minimiser l'énergie d'un matériau. Le principe est qu'à chaque itération, on va chercher une nouvelle solution dans le voisinage de la solution trouvée précédemment. Ceci va donc induire une variation de l'énergie  $\delta E$  du système. L'énergie du système représentera alors la fonction objectif à minimiser.

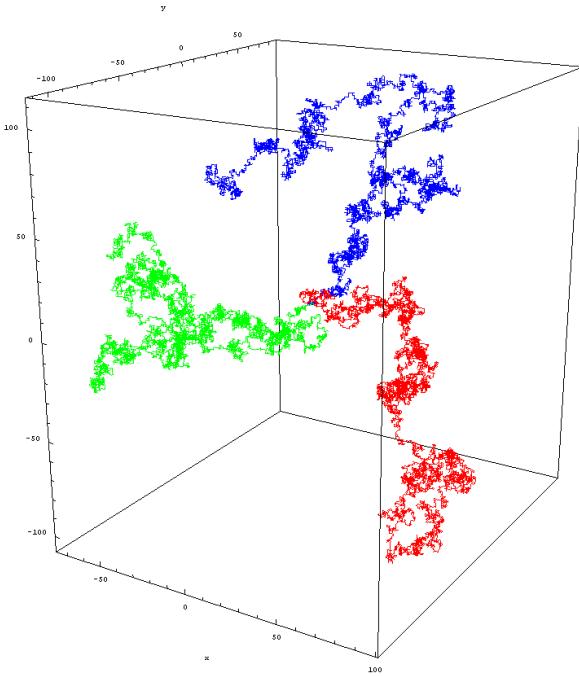


FIGURE 2.12 – Trois exemples de promenade aléatoire dans un espace à trois dimensions [109]

Si cette variation est négative, c'est que la nouvelle solution minimise mieux la fonction objectif, donc on accepte cette solution et l'on passe à la prochaine itération. Si cette variation est positive, c'est que la solution n'est pas meilleure, mais dans ce cas la solution n'est pas systématiquement refusée, elle est acceptée avec une probabilité de  $e^{\frac{\delta E}{T}}$  avec  $T$  la température actuelle : c'est la règle de Métropolis.

Tout au long de l'algorithme, la température va diminuer, soit linéairement, soit par paliers. Si on atteint la température minimale choisie, alors l'algorithme s'arrête.

```

s := s0
e := E(s)
k := 0
tant que k < kmax et e > emax
    sn := voisin(s)
    en := E(sn)
    si en < e ou aléatoire() < P(en - e, temp(k/kmax)) alors
        s := sn; e := en
    k := k + 1
retourne s

```

FIGURE 2.13 – Pseudo-code mettant en œuvre le recuit simulé [41]

Méthode simple à implémenter dans le principe, les principaux inconvénients du recuit simulé résident néanmoins dans le choix des nombreux paramètres, tels que la température initiale, la loi de décroissance de la température, les critères d'arrêt ou la longueur des paliers de température. Ces paramètres sont souvent choisis de manière empirique.

### 2.5.3 Tests de performance et comparaison

Une fois les méthodes sélectionnées implémentées, il convient de les tester sur des problèmes d'optimisation plus ou moins complexes. L'objectif est de pouvoir avoir une première idée de

l'efficacité de ces algorithmes, de valider la phase de développement, et de pouvoir sélectionner la méthode qui convient le plus à nos besoins pour pouvoir l'appliquer à notre problème d'optimisation lié à l'inversion de vêtements.

Nous présenterons alors dans cette sous-section, dans un premier temps, les fonctions à deux ou plusieurs variables utilisées pour tester les méthodes d'optimisation. Dans un second temps, nous nous intéresserons aux *Performance Profile*, représentation graphique utilisée pour mesurer et comparer la performance d'algorithmes.

### Fonctions tests à optimiser

Afin de pouvoir tester les méthodes d'optimisation sans gradient implémentées, nous avons sélectionné un certain nombre de fonctions tests habituellement choisies pour ce genre d'exercice. Dans le but de pouvoir tester les algorithmes dans différentes configurations, les fonctions tests doivent pouvoir représenter une assez large nature de problèmes à résoudre : fonctions à plusieurs minimums locaux, nombreux puits, etc.

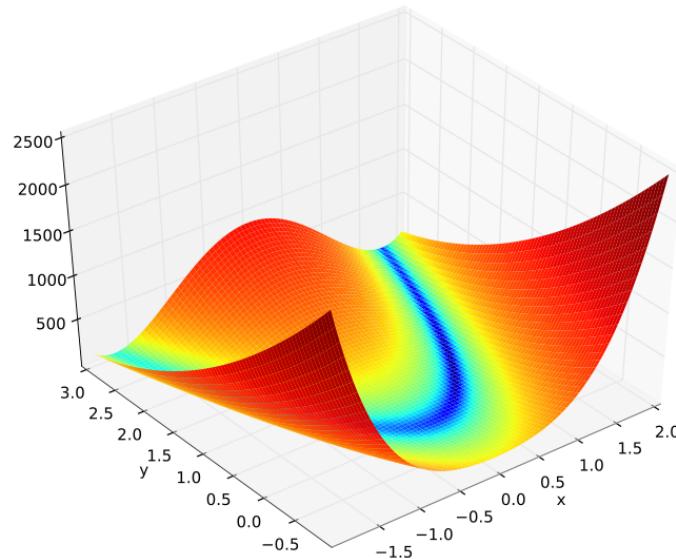


FIGURE 2.14 – Fonction de Rosenbrock –  $f(\mathbf{x}) = \sum_{i=1}^{n-1} \left[ 100 (x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$  [62]

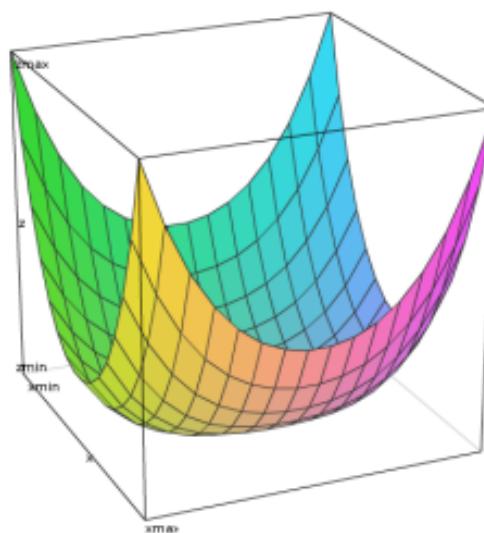


FIGURE 2.15 – Fonction Quartique –  $f(\mathbf{x}) = \sum_{i=1}^{n-1} x_i^2 + (x_i^2 + x_{i+1}^2)^2$  [63]

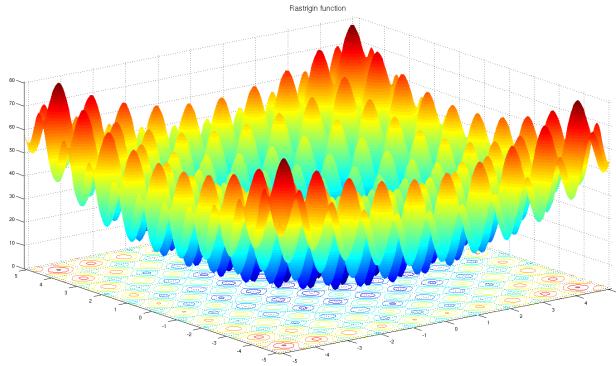


FIGURE 2.16 – Fonction de Rastrigin –  $f(\mathbf{x}) = 10n + \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i)]$  [62]

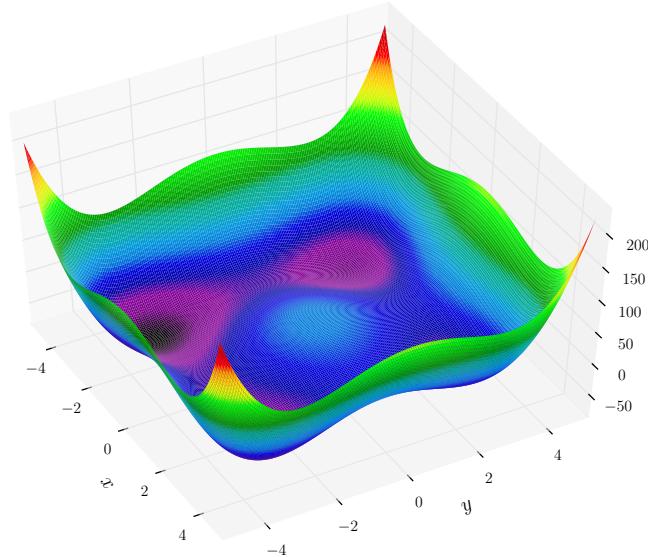


FIGURE 2.17 – Fonction de Styblinski-Tang –  $f(\mathbf{x}) = 0.5 \sum_{i=1}^n x_i^4 - 16x_i^2 + 5x_i$  [62]

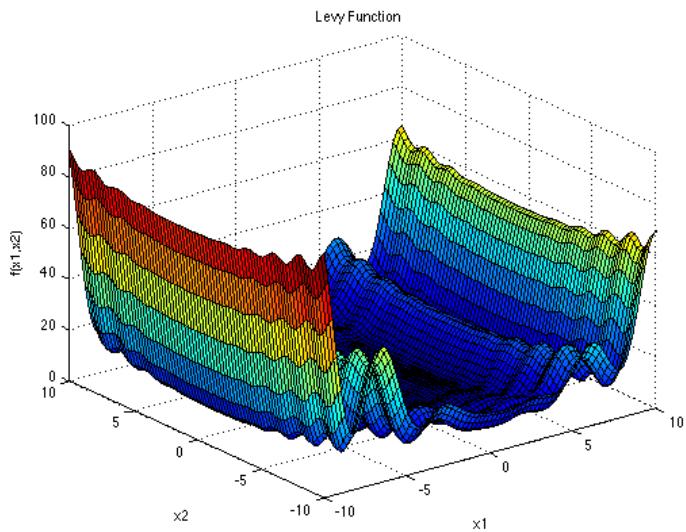


FIGURE 2.18 – Fonction de Levy –  $f(\mathbf{x}) = \sin^2(\pi\omega_1) + \sum_{i=1}^{n-1} (\omega_i - 1)^2[1 + 10 \sin^2(\pi\omega_i + 1)] + (\omega_n - 1)^2[1 + \sin^2(2\pi\omega_n)]$  avec  $\omega_i = 1 + 0.25(x_i - 1)$  [63]

### Performance profile

Les graphiques de type *Performance Profile* [65] sont un moyen assez répandus dans le monde de la recherche lorsqu'il faut comparer la performance de différents algorithmes, logiciels

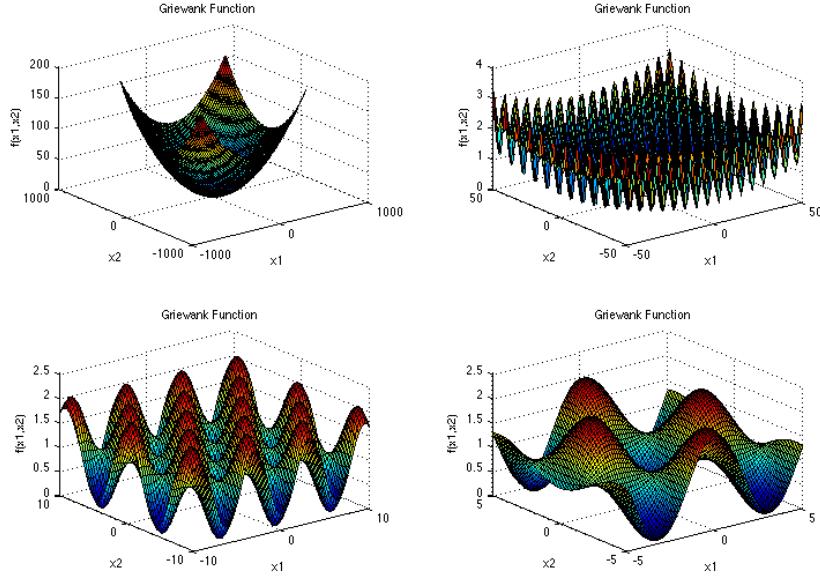


FIGURE 2.19 – Fonction de Griewank –  $f(\mathbf{x}) = \sum_{i=1}^n \frac{x_i^2}{4000} - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$  [63]

ou méthodes résolvant un problème. Ils permettent de rendre compte à la fois de la rapidité et de la précision d'un algorithme concernant la ou les solutions proposées.

Pour ce faire, on réalise met en place une batterie de tests d'optimisation avec les différentes fonctions tests vues précédemment, et en faisant également varier la valeur initiale des algorithmes. On suppose alors que nous avons un nombre  $n_s$  de solveurs formant l'ensemble  $\mathcal{S}$ , et un nombre  $n_p$  de problèmes à résoudre formant l'ensemble  $\mathcal{P}$ .

Dans notre cas, il semble intéressant d'utiliser le nombre d'évaluation de la fonction objectif comme mesure de la performance. Pour chaque problème  $p$  et chaque solveur  $s$ , on définit alors

$$t_{p,s} = \text{nb d'évaluat } \circ \text{ de la f } \circ \text{ objectif pour résoudre le problème } p \text{ par le solveur } s \quad (2.16)$$

Afin de pouvoir comparer les différents solveurs, il faut disposer d'une base de comparaison. Comme explicité dans [65], pour un même problème  $p$ , on compare la performance de chaque solveur  $s$  avec la performance du meilleur solveur rencontré pour ce problème. On définit alors un *performance ratio* :

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s}, s \in \mathcal{S}\}} \quad (2.17)$$

Pour un problème  $p$  donné, on définit donc un ratio pour chaque solveur  $s$  comme étant, le nombre d'évaluation de la fonction objectif nécessaire pour le solveur  $s$  pour résoudre le problème  $p$ , divisé par le plus petit nombre d'évaluation de la fonction parmis tous les solveur pour ce même problème  $p$ . Par ailleurs, si un solveur ne parvient pas à résoudre un problème, on fixera son ratio à une valeur maximale  $r_M$ , tel que  $r_M \geq r_{p,s}$  quelque soit le problème  $p$ .

On peut alors définir la performance globale d'un solveur  $s$  pour tous les problèmes de  $\mathcal{P}$ . Si l'on définit

$$\rho_s(\tau) = \frac{1}{n_p} \text{ size}\{p \in \mathcal{P} / r_{p,s} \leq \tau\} \quad (2.18)$$

alors  $\rho_s(\tau)$  est la probabilité pour le solveur  $s \in \mathcal{S}$  qu'un *performance ratio*  $r_{p,s}$  soit à un facteur  $\tau \in \mathbb{R}$  du meilleur ratio possible. La fonction  $\rho_s(\tau)$  peut aussi être vue comme la fonction de distribution (cumulative) pour le *performance ratio*. On représentera alors  $\rho_s$  en fonction de

$\tau$  pour obtenir le graphe de performance du solveur  $s$ . La fonction  $\rho_s : \mathbb{R} \rightarrow [0, 1]$  est en effet une fonction strictement croissante.

La valeur  $\rho_s(1)$  est la probabilité que le solveur  $s$  soit meilleur que les autres solveurs (en terme de nombre d'évaluation de la fonction objectif). Ainsi, si on est intéressé uniquement par la rapidité (en terme de nombre d'évaluation de la fonction objectif), on comparera uniquement les valeurs de  $\rho_s(1)$  pour tous les solveurs de  $\mathcal{S}$ .

Par ailleurs, on rappelle que les *performance ratio*  $r_{p,s} \in [1, r_M]$ , et que  $r_{p,s} = r_M$  seulement lorsque le problème  $p$  n'est pas résolu par la méthode  $s$ . Ainsi, ce qui découle de cette convention est que  $\rho_s(r_M) = 1$  et la quantité

$$\rho_s^* = \lim_{\tau \rightarrow r_M^-} \rho_s(\tau) \quad (2.19)$$

est la probabilité que la méthode  $s$  résolve un problème. Ainsi, si on est intéressé uniquement par des solveurs à la grande probabilité de succès, on comparera alors uniquement les valeurs de  $\rho_s^*$  pour tous les solveurs de  $\mathcal{S}$ , et on gardera celui à la plus grande valeur. La valeur de  $\rho_s^*$  peut être facilement observée dans un *performance profile*, car la fonction  $\rho_s$  stagne pour de grandes valeurs de  $\tau$ .

### Comparaison des méthodes implémentées

L'utilisation des *performance profile* nous permet donc de pouvoir comparer les différentes méthodes d'optimisation sans gradient implémentées, et de pouvoir observer quelle est la meilleure sur une large variété de problèmes. Cependant, avant de comparer les méthodes entre elles, il convient d'abord de comparer, pour une méthode d'optimisation donnée, les performances obtenues en fonction des paramètres manuels de la méthode en question. En effet, un des désavantages de ce type de méthode et de trouver les bons paramètres liés au problème en question, et l'utilisation de *performance profile* peut être utile dans cette optique.

On se propose donc, dans un premier temps, de choisir un paramètre prépondérant dans la vitesse de convergence de l'algorithme et de la faire varier, afin de comparer, pour une même méthode, les performances en fonction de ce paramètre. On trace alors les profils de performances pour ensemble de problèmes  $\mathcal{P}$  de cardinal 600. En effet, on sélectionne 6 fonctions tests à optimiser, et on génère 100 points initiaux pour l'optimisation et les solveurs doivent résoudre le problème avec une certaine précision pour la solution ( $10^{-2}$ ,  $10^{-1}$  ou  $10^{-4}$  selon le nombre de variables considérées).

Dans la figure 2.20, on représente le profil de performance pour la méthode de l'essaim particulaire, en faisant varier la paramètre lié à la taille de l'essaim particulaire (nombre de particules). Pour des problèmes à 2 variables (figure du haut) avec une précision de  $10^{-2}$  demandée, toutes les méthodes convergent pour tous les problèmes (les courbes jaune et noire atteignent une valeur de 1 pour un  $\tau$  suffisamment grand) sauf pour un nombre trop faible de particules (25, mais qui a la plus grande probabilité d'être la plus rapide). Pour des problèmes plus compliqués, à 4 variables (figure du bas) avec une précision de  $10^{-1}$  demandée, on remarque que pour une nombre trop faible de particules (en dessous de 1000) la méthode ne résoud pas tous les problèmes. La courbe bleu cyan (1000 particules) semble être un bon compromis entre probabilité d'être la plus rapide et de résoudre les problèmes, par rapport aux autres paramètres.

Dans la figure 2.21, on représente le profil de performance pour la méthode de l'algorithme génétique, en faisant varier la paramètre lié à la taille de la population (nombre d'individus). Pour des problèmes à 2 variables (figure du haut) avec une précision de  $10^{-2}$  demandée, peu de méthodes trouvent une solution pour tous les problèmes. Seulement les courbes jaune (1000 individus) et violette (500 individus) atteignent une valeur de 1 pour un  $\tau$  suffisamment grand. Cependant, en terme de rapidité (nombre d'évaluation de la fonction objectif), la courbe rouge (100 individus) semble avoir la plus grande probabilité de résoudre rapidement un des problèmes

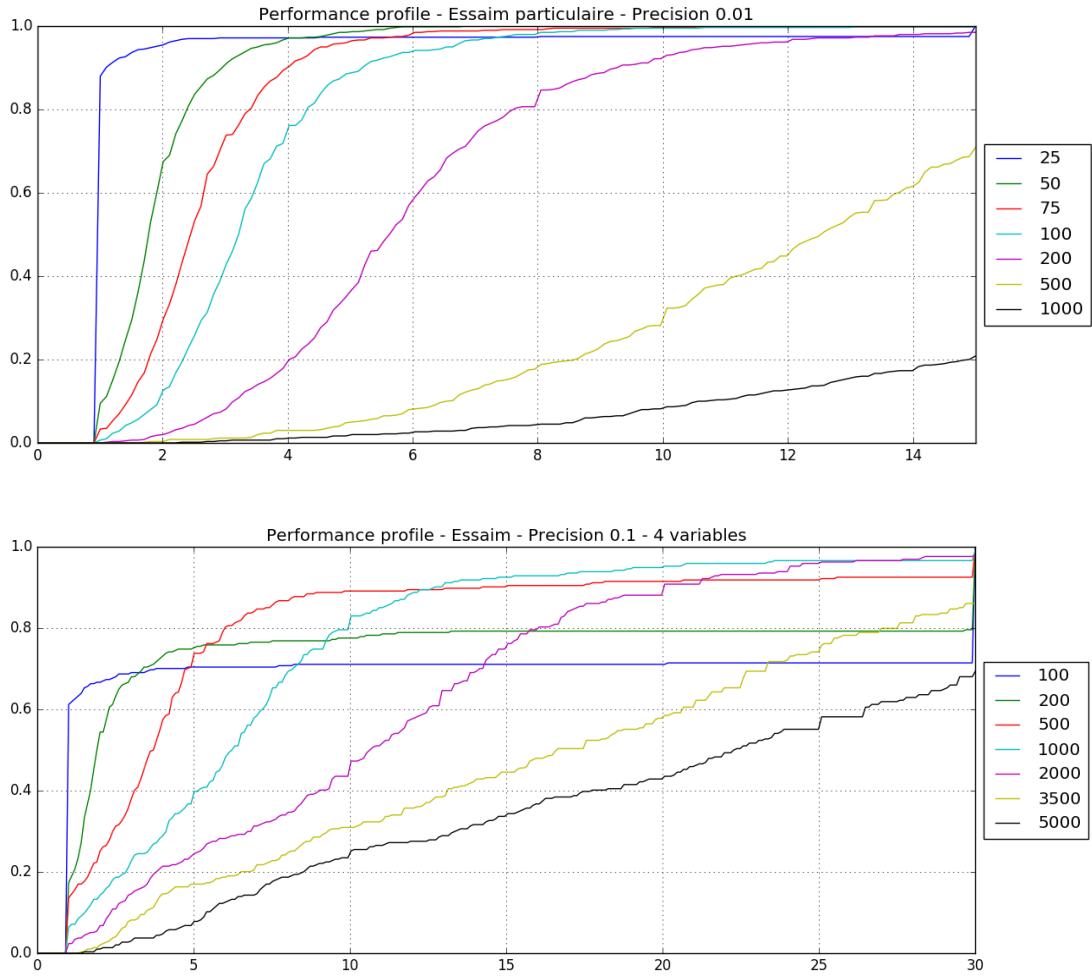


FIGURE 2.20 – Performance profile – Optimisation par essaim particulaire

posés. Pour des problèmes plus compliqués, à 4 variables (figure du bas) avec une précision de  $10^{-1}$  demandée, on remarque que pour une nombre trop faible de particules (en dessous de 1000) la méthode ne résoud pas tous les problèmes. La courbe bleu cyan (1000 particules) semble être un bon compromis entre probabilité d'être la plus rapide et de résoudre les problèmes, par rapport aux autres paramètres.

Dans la figure 2.22, on représente le profil de performance pour la méthode de la promenade aléatoire, en faisant varier la paramètre lié au pas de descente initial ( $\alpha_0$ ). Pour des problèmes à 2 variables (figure du haut) avec une précision de  $10^{-2}$  demandée, aucune méthode ne trouve de solution pour tous les problèmes. Les courbes bleu marine ( $\alpha_0 = 0.1$ ) et verte ( $\alpha_0 = 1$ ) ont une probabilité de résoudre un problème plus faible que les autres méthodes. Cependant, en terme de rapidité (nombre d'évaluation de la fonction objectif), la courbe bleu marine semble avoir la plus grande probabilité de résoudre rapidement un des problèmes posés, lorsqu'elle arrive à le résoudre.. Pour des problèmes plus compliqués, à 4 variables (figure du bas) avec une précision de  $10^{-1}$  demandée, on remarque que pour une nombre trop grand de pas de descente initial (au dessus de 20), on perd à la fois en rapidité et efficacité de la méthode. Ainsi, le paramètre  $\alpha_0 = 10$  semble être un bon compromis entre probabilité d'être la plus rapide et de résoudre les problèmes, par rapport aux autres paramètres.

Dans la figure 2.23, on représente le profil de performance pour la méthode du recuit simulé, en faisant varier la paramètre lié à la température minimale (condition d'arrêt de l'algorithme). Pour des problèmes à 2 variables (figure du haut) avec une précision de  $10^{-2}$  demandée, aucune méthode ne trouve de solution pour tous les problèmes. Les courbes rouge ( $T_{min} = 10^{-5}$ ) et bleu cyan ( $T_{min} = 10^{-8}$ ) ont une probabilité de résoudre un problème plus élevée que les autres

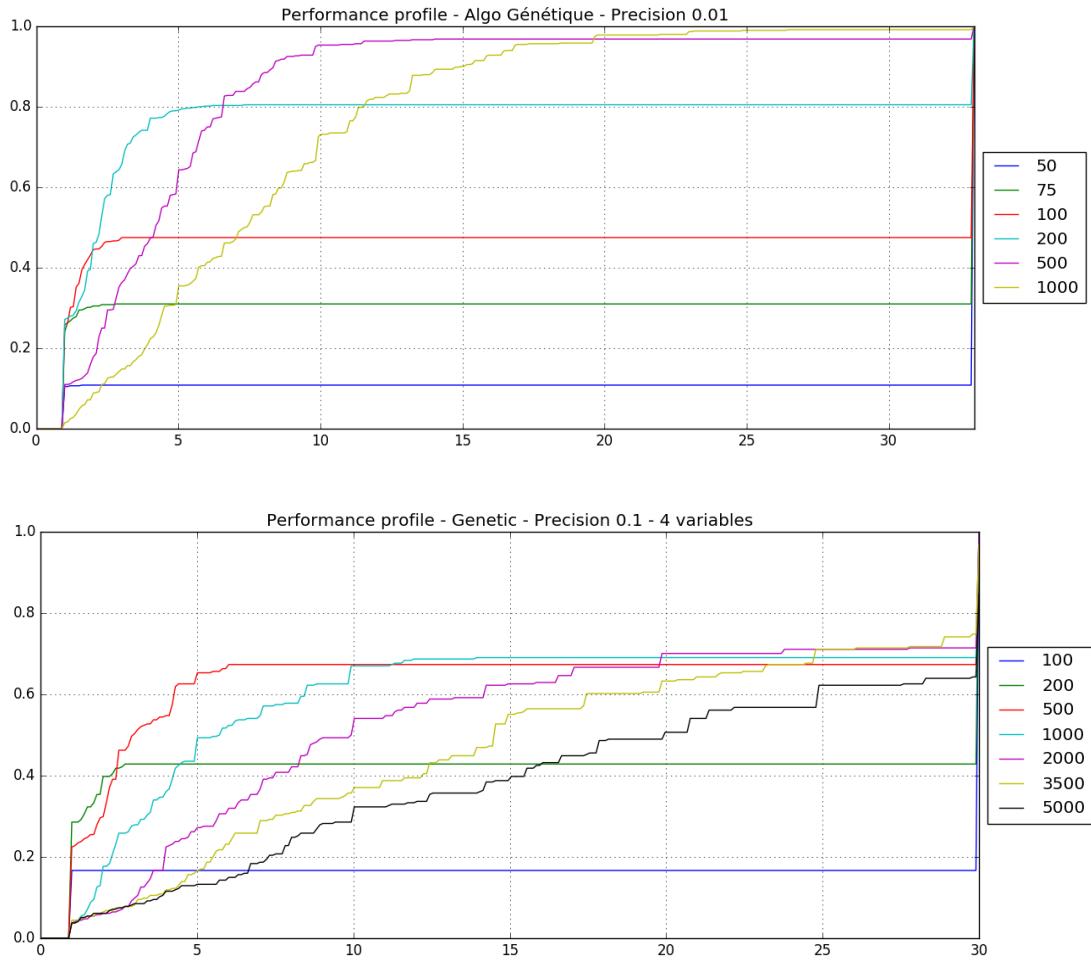


FIGURE 2.21 – Performance profile – Algorithme génétique

méthodes, mais restant tout de même assez faible. En terme de rapidité (nombre d'évaluation de la fonction objectif), toutes les courbes semblent se valoir. Etant donné la difficulté de la méthode du recuit simulé, quelque soit les paramètres, pour résoudre les problèmes à 2 variables, et après un rapide test de ses performances pour des problèmes à un plus grand nombre de variables, nous décidons de ne pas tracer de profil de performance pour des problèmes plus compliqués, à 4 variables par exemple. A la place, et pour illustrer nos propos, on s'intéresse à des problèmes à 2 variables avec une précision de  $10^{-4}$  demandée (figure du bas). On remarque qu'aucune méthode ne parvient à résoudre aucun des problèmes pour une telle précision, même les plus simples d'entre eux. Ainsi, et afin tout de même de pouvoir comparer avec les autres méthodes d'optimisation, le paramètre  $T_{min} = 10^{-8}$  est conservé comme étant le plus intéressant..

Dans la figure 2.24, on représente le profil de performance pour chaque méthode implémentée en choisissant le meilleur paramètre mis en évidence précédemment. Pour des problèmes à 2 variables (figure du haut) avec une précision de  $10^{-2}$  demandée, on remarque que l'optimisation par essaim particulier dépasse de loin toutes les autres méthodes, autant en terme de rapidité qu'en probabilité de résolution d'un problème. En second viennent l'algorithme génétique, moins rapide pour les problèmes à 2 variables mais résolvant tous les problèmes proposés aussi. Bien moins performants, les algorithmes de promenade aléatoire et du recuit n'arrivent pas à résoudre tous les problèmes posés, le recuit simulé étant à priori le plus mauvais. Pour des problèmes à 4 variables (figure du bas) avec une précision de  $10^{-1}$  demandée, on remarque que seule la méthode d'optimisation par essaim particulier réussit à résoudre tous les problèmes posés, mais que l'algorithme génétique a la plus grande probabilité d'être rapide pour résoudre un problème, sans toutefois l'être beaucoup plus que l'essaim particulier. Enfin, l'algorithme de promenade aléatoire ne résout le moins de problèmes et le moins vite par rapport aux deux autres méthodes.

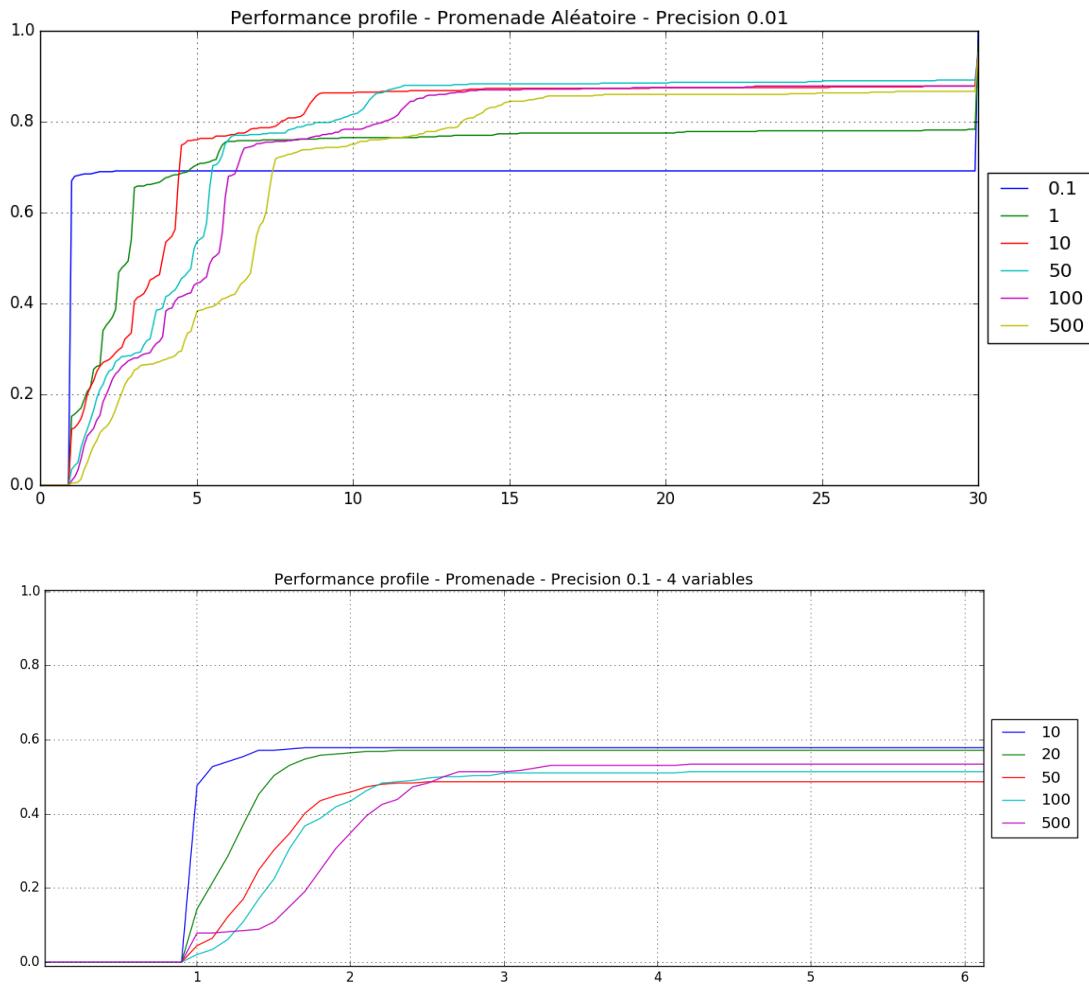


FIGURE 2.22 – Performance profile – Promenade aléatoire

A l'issue de cette étude, nous avons choisi de ne pas pousser plus loin les calculs (passer à 6, 10, 100 variables) car les problèmes posés risquent d'être différents du problème d'optimisation lié à l'inversion dynamique. Néanmoins, cela nous a amené à porter notre choix sur l'algorithme d'optimisation par essaim particulaire, qui ressortait souvent du lot lors de lancement de tests et de profils de performance.

## 2.6 Optimisation avec gradient

Les algorithmes basés gradient désignent des algorithmes d'optimisation différentiables, et par conséquent, sont destinés à minimiser une fonction réelle différentiable, généralement définie sur un espace hilbertien (et assez souvent euclidien). Ils sont pour la plupart itératifs et procèdent donc par améliorations successives. Au point courant, un déplacement est effectué dans la direction *opposée* au gradient, de manière à faire décroître la fonction objectif.

Ainsi, l'information du gradient permet de déterminer la meilleure direction de descente, et classe donc les algorithmes basés gradient parmi méthodes d'optimisation à direction de descente (on peut éventuellement jouer sur le pas de descente aussi). Cependant, un tel algorithme ne permet de trouver ou d'approcher qu'un point stationnaire (un point en lequel le gradient de la fonction à minimiser est nul) d'un problème d'optimisation sans contrainte. De tels points sont des minima globaux, si la fonction est convexe, des minima locaux sinon.

Etant donné l'expression de notre fonction objectif à minimiser, et du modèle sous-jacent

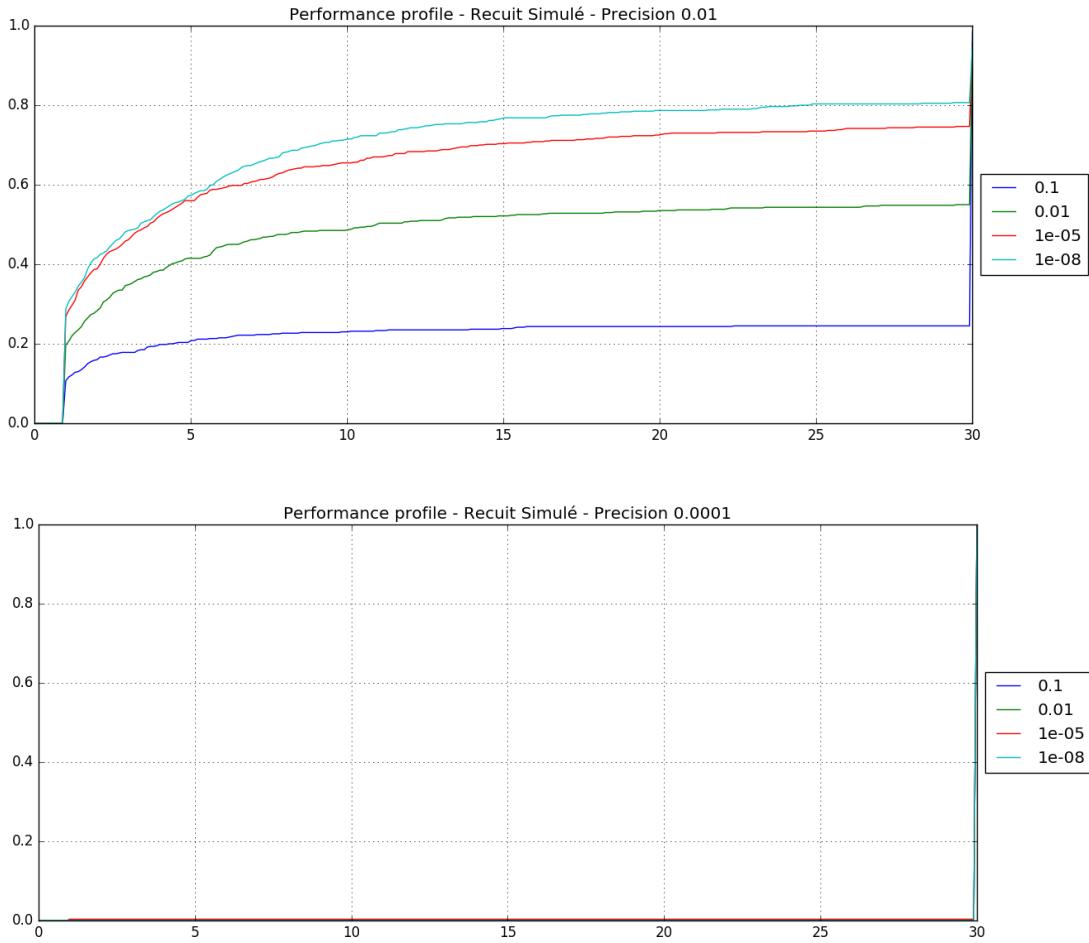


FIGURE 2.23 – Performance profile – Récuit simulé

au lancement d'une simulation d'affaissement, nous pouvons conjecturer que la fonction objectif n'est pas convexe. De plus, il est fortement possible qu'il existe une multitude de minima locaux, qui ne permettraient alors pas aux algorithmes basés gradient de toujours converger vers le minimum global attendu. Vu sous cet angle, il parrait plus intéressant de privilégier les méthodes d'optimisation sans gradient, permettant pour certaines de directement s'approcher du minimum global.

Néanmoins, si l'algorithme basé gradient est situé dans le bon puit (celui du minimum global), alors la convergence sera beaucoup plus rapide dans ce cas, car la direction de descente est optimale. On pourrait alors imaginer une combinaison des méthodes sans et avec gradient, pour profiter de la recherche globale de l'un, et de la rapidité de convergence en fin d'algorithme pour l'autre. C'est en partie ce qui nous a motivé à tenter d'implémenter des méthodes basées gradient pour notre problème d'optimisation lié à l'inversion de vêtements.

Dans cette section nous présenterons alors comment peut se mettre en place le calcul du gradient de la fonction objectif, notamment par la mise en œuvre de la méthode de l'adjoint [74, 75, 76, 77, 78]. Nous présenterons en premier lieu comment cela a été appliqué pour le problème d'inversion statique de vêtements. Dans un second temps, nous nous intéresserons à l'expression et l'implémentation du gradient pour le problème dynamique que nous rencontrons, et établirons un lien avec le problème 4D-Var connu en assimilation de données.

### 2.6.1 Calcul du gradient dans le cas statique

Dans cette sous-section nous nous intéresserons au calcul du gradient de la fonction objectif reliée au problème d'inversion statique (voir 2.4.2). Comme vu précédemment, l'expression de

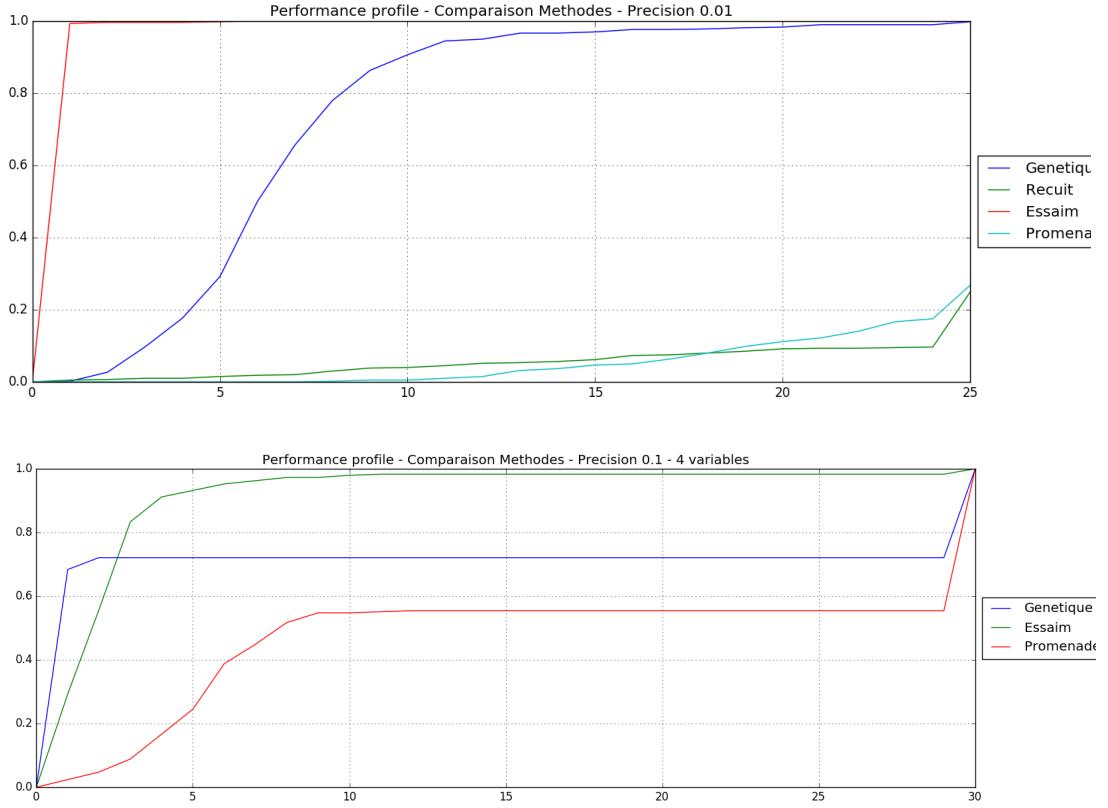


FIGURE 2.24 – Performance profile – Comparaison des méthodes

la fonction objectif est donnée par :

$$J(\bar{x}) = \frac{1}{2} \| \phi(\bar{x}) - x_t \|^2 \quad (2.20)$$

En utilisant la formule de la dérivée d'une fonction composée on arrive à l'expression suivante du gradient de  $J$  :

$$\nabla J(\bar{x}) = [D\phi(\bar{x})]^\top (\phi(\bar{x}) - x_t) \quad (2.21)$$

La principale difficulté dans cette expression de  $\nabla J$  repose sur le calcul de la différentielle de  $\phi$ , modélisant le passage de la forme au repos vers la forme à l'équilibre par le simulateur. Afin de contourner ce problème et de pouvoir obtenir une expression de  $D\phi$ , l'idée est de solliciter le théorème des fonctions implicites [90] en introduisant une fonction  $F$ , telle que :

$$x = \phi(\bar{x}) \iff F(x, \bar{x}) = 0 \quad (2.22)$$

Ainsi, en validant un certain nombre de conditions concernant la fonction  $F$ , le théorème des fonctions implicites nous permet d'obtenir l'expression de  $D\phi$  comme étant :

$$D\phi(\bar{x}) = -[D_x F(\phi(\bar{x}), \bar{x})]^{-1} \circ D_{\bar{x}} F(\phi(\bar{x}), \bar{x}) \quad (2.23)$$

Dans le cas du problème d'inversion statique considéré par Romain Casati [11], la fonction  $F$  peut être choisie comme étant le gradient  $\nabla_x E_p$  de l'énergie potentielle associée au vêtement, et dans ce cas l'expression de la différentielle de  $F$  selon  $\bar{x}$  et  $x$  est effectivement calculable.

On peut désormais expliciter l'expression de la transposée de la différentielle de la fonction  $\phi$  apparaissant dans l'équation 2.21 :

$$[D\phi(\bar{x})]^\top = -[D_{\bar{x}} F(\phi(\bar{x}), \bar{x})]^\top \circ [D_x F(\phi(\bar{x}), \bar{x})]^{-1}^\top \quad (2.24)$$

Après une ré-écriture des termes en permutant la transposition et l'inversion, et en ramenant le signe moins à droite :

$$[D\phi(\bar{x})]^\top = [D_{\bar{x}}F(\phi(\bar{x}), \bar{x})]^\top \circ \left( - \left[ [D_x F(\phi(\bar{x}), \bar{x})]^\top \right]^{-1} \right) \quad (2.25)$$

Le principe de la méthode de l'adjoint est d'introduire un *état adjoint*  $p$ , et à l'aide d'une astuce de calcul d'éviter l'inversion de la différentielle  $D_x F$ . Pour mettre en évidence cet état adjoint, on injecte l'expression de  $[D\phi(\bar{x})]^\top$  dans l'équation du gradient 2.21 :

$$\nabla J(\bar{x}) = [D_{\bar{x}}F(\phi(\bar{x}), \bar{x})]^\top \circ \underbrace{\left( - \left[ [D_x F(\phi(\bar{x}), \bar{x})]^\top \right]^{-1} \right)}_{=p} (\phi(\bar{x}) - x_t) \quad (2.26)$$

L'expression de l'état adjoint est donc donnée par :

$$p = - \left[ [D_x F(\phi(\bar{x}), \bar{x})]^\top \right]^{-1} (\phi(\bar{x}) - x_t) \quad (2.27)$$

Cependant, le calcul de l'état adjoint en utilisant directement cette formule peut s'avérer assez difficile de par la présence d'une inversion de la transposée de la différentielle de  $F$ . On multiplie alors des deux côté par cette différentielle afin de contourner le problème. On obtient alors :

$$[D_x F(\phi(\bar{x}), \bar{x})]^\top p = x_t - \phi(\bar{x}) \quad (2.28)$$

On cherchera alors à déterminer l'état adjoint  $p$  tel qu'il résoud cette équation, en ayant seulement à calculer la transposée de la différentielle, et non plus son inverse également. Une fois l'état adjoint déterminé, on en déduit la valeur de gradient de la fonction objectif, après avoir déterminé la second différentielle (selon  $\bar{x}$ ) :

$$\nabla J(\bar{x}) = [D_{\bar{x}}F(\phi(\bar{x}), \bar{x})]^\top p \quad (2.29)$$

En utilisant la méthode de l'adjoint, tout en reformulant le problème à l'aide du théorème des fonctions implicites, on peut alors arriver à calculer le gradient de la fonction objectif, dans le but d'implémenter des méthodes d'optimisation utilisant l'information du gradient (comme l'algorithme L-BFGS [91] par exemple, qui lui estime également l'inverse de la matrice hessienne de la fonction).

## 2.6.2 Calcul du gradient dans le cas dynamique

Afin de pouvoir mettre en place des méthodes basées gradient, nous ne pouvions pas passer outre le fait de calculer explicitement la valeur du gradient de la fonction objectif, afin de pouvoir s'en servir comme direction de descente. Etant donné que la méthode de l'adjoint a déjà été utilisée pour la cas classique, ceci nous a guidé afin de l'appliquer également pour le cas dynamique.

Pour rappel, l'expression de notre fonction objectif peut être simplifiée en :

$$J(x_0) = \frac{1}{2} \sum_{t=T_0}^{T_N} \| \phi(x_0, t) - X_{obs}(t) \|^2 \quad (2.30)$$

Une fois encore, l'expression de la différentielle de la fonction  $\phi(x_0, t)$  sera difficile à obtenir, d'autant plus qu'elle est à plusieurs variables et surtout prenant en compte le temps. Néanmoins, l'expression de notre fonction objectif se rapproche sensiblement de l'expression de la fonction

objectif usuellement utilisée pour modéliser le problème 4D-Var connu en assimilation de données [92].

Souvent utilisée en météorologie, l'assimilation de données est le procédé qui consiste à corriger, à l'aide d'observations, l'état de l'atmosphère d'une prévision météorologique. On distingue alors deux type de problèmes : le 3D-Var se faisant à un pas de temps fixe dans les trois dimensions cartésiennes, et le 4D-Var se faisant à plusieurs pas temps entre le temps initial et un temps futur de prévision. La fonction coût considérée dans le problème 4D-Var s'écrit :

$$J_a(x) = \underbrace{(x - x^b)^\top B^{-1} (x - x^b)}_{\text{terme d'ébauche}} + \underbrace{\sum_{i=0}^n (y_i^o - H_i(x_i))^\top R_i^{-1} (y_i^o - H_i(x_i))}_{\text{terme d'observation}} \quad (2.31)$$

Dans notre cas, nous ne considérons pas de terme d'ébauche, mais seulement un terme d'observation. De plus, la matrice  $H$  d'observation et matrice de covariance d'erreur d'observation  $R$  correspondent à l'identité dans notre cas, car on suppose l'observation des états réels parfaite (car pour l'instant issus du logiciel de simulation). On retrouve alors bien l'expression de notre fonction objectif, à un facteur  $\frac{1}{2}$  près. Pour résoudre le problème 4D-Var, les chercheurs dans ce domaine utilisent la méthode de l'adjoint pour calculer le gradient de la fonction coût. Nous nous inspirons alors des travaux de Didier Auroux [76] notamment pour exprimer et tenter de calculer le gradient de notre fonction d'objectif.

Si l'on se place alors d'un point de vue discret, on peut décrire l'évolution comme :

$$\phi(x_0, T_i) = M_{0 \rightarrow T_i}(x_0) = x(T_i) \quad (2.32)$$

On chercher alors à simplifier le problème. Pour ce fait on pose de 2 hypothèses :

– **Causalité** : On suppose que modèle d'évolution peut s'écrire comme une suite de modèles intermédiaires :

$$\phi(x_0, T_{i+1}) = M_{T_{i+1}} \phi(x_0, T_i) \quad (2.33)$$

D'où :

$$\phi(x_0, T_{i+1}) = M_{T_{i+1}} M_{T_i} \cdots M_2 M_1 x_0 \quad (2.34)$$

– **Approximation linéaire tangente** : On remplace le modèle  $M$  par son approximation linéaire tangente (ou sa dérivée). Cette hypothèse est généralement valide si la période d'assimilation n'est pas trop longue.

Ces deux hypothèses nous ramènent alors à un problème d'optimisation linéaire et sans contrainte. On peut alors reformuler la fonction objectif comme :

$$J(x_0) = \frac{1}{2} \sum_{t=T_0}^{T_N} \| M_t \cdots M_2 M_1 x_0 - X_{obs}(t) \|^2 \quad (2.35)$$

Nous pouvons à présent donner une première expression du gradient de la fonction objectif :

$$\nabla J(x_0) = \sum_{t=T_0}^{T_N} M_t^\top \cdots M_2^\top M_1^\top (M_t \cdots M_2 M_1 x_0 - X_{obs}(t)) \quad (2.36)$$

En introduisant ce que l'on appelle communément un vecteur d'innovation  $d_i = \phi(x_0, T_i) - X_{obs}(T_i)$ , on peut ré-écrire le gradient de la manière suivante :

$$\nabla J(x_0) = d_0 + M_1^\top [d_1 + M_2^\top [d_2 + \cdots [M_N^\top d_N] \cdots]] \quad (2.37)$$

Le calcul effectif de  $\nabla J(x_0)$  se fait alors d'abord par un calcul du terme d'observation. L'algorithme de résolution consiste en une intégration du modèle direct et une intégration du modèle adjoint. Il est présenté ci-dessous :

### Algorithme de résolution

**On intègre le modèle direct :** A chaque étape  $i$  (de 1 à  $T_N$ ) on calcule :

- les  $x(t_i) = M_i x(t_{i-1})$
  - les vecteurs d'innovation  $d_i = x(T_i) - X_{obs}(T_i)$ , lorsque  $i \in [T_0, T_N]$  et on les stocke
  - le terme de la fonction coût :  $J_i(x_0) = \frac{1}{2} \|x(T_i) - X_{obs}(T_i)\|^2 = \frac{1}{2} d_i^\top d_i$ , lorsque  $i \in [T_0, T_N]$
- Finalement  $J(x_0) = \sum_{i=T_0}^{T_N} J_i(x_0)$

**On calcule le gradient :**

- initialiser l'état adjoint  $p$  à 0, à l'instant final  $\Rightarrow p(T_N) = 0$
- à chaque étape  $i-1$ , l'état adjoint  $p(T_{i-1})$  est déduit en utilisant le modèle adjoint (adjoint du modèle linéaire tangent) :

$$p(T_{i-1}) = M_{T_i}^\top (p(T_i) + d_i) \quad (2.38)$$

- à l'étape 0, on obtient le gradient  $\nabla J(x_0) = p(T_0)$

Malheureusement, par manque de temps, l'algorithme calculant le gradient n'a pas pu être implémenté, et la minimisation de la fonction objectif par des algorithmes d'optimisation avec gradient n'a pas pu être effectuée.

## 2.7 Résultats et discussions

### 2.7.1 Inversion statique et dynamique du bitriangle

Le premier cas d'application considéré fut celui de l'inversion statique du bitriangle, choisi en premier pour sa simplicité et pour le nombre réduit de variables (6 variables à optimiser, correspondant à  $2 \times 3$  coordonnées de points libres). Pour rappel, le but est, à partir de la forme à l'équilibre du vêtement, de retrouver la forme au repos (qui est aussi la forme initiale de la simulation) telle qu'après simulation d'affaissement, on retrouve la même force à l'équilibre.

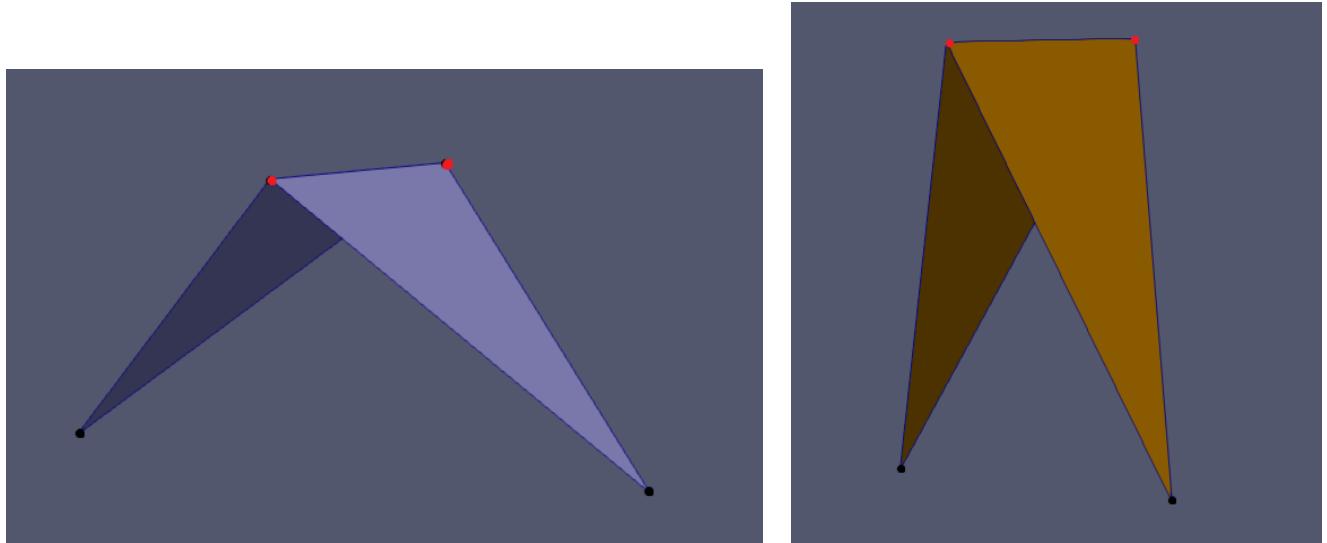


FIGURE 2.25 – Forme au repos (gauche) et forme à l'équilibre (droite) – Bitriangle

Nous partons donc de la forme à l'équilibre et le but est de retrouver la forme au repos (voir figure 2.25). On initialise donc l'algorithme d'optimisation par essaim particulaire avec la

position à l'équilibre du bitriangle, et on chercher à minimiser la fonction objectif dans le cas statique (voir équation 2.6).

Les premiers résultats sont que l'algorithme d'optimisation par essaim particulaire arrive à minimiser la fonction objectif jusqu'à l'ordre de  $10^{-9}$ , mais cependant la forme au repos n'est pas exactement la même que celle attendue en théorie. Sur la figure 2.26, en bleu est représentée la forme au repos, en doré la forme à l'équilibre et en blanc/gris la solution donnée à l'issue de l'optimisation.

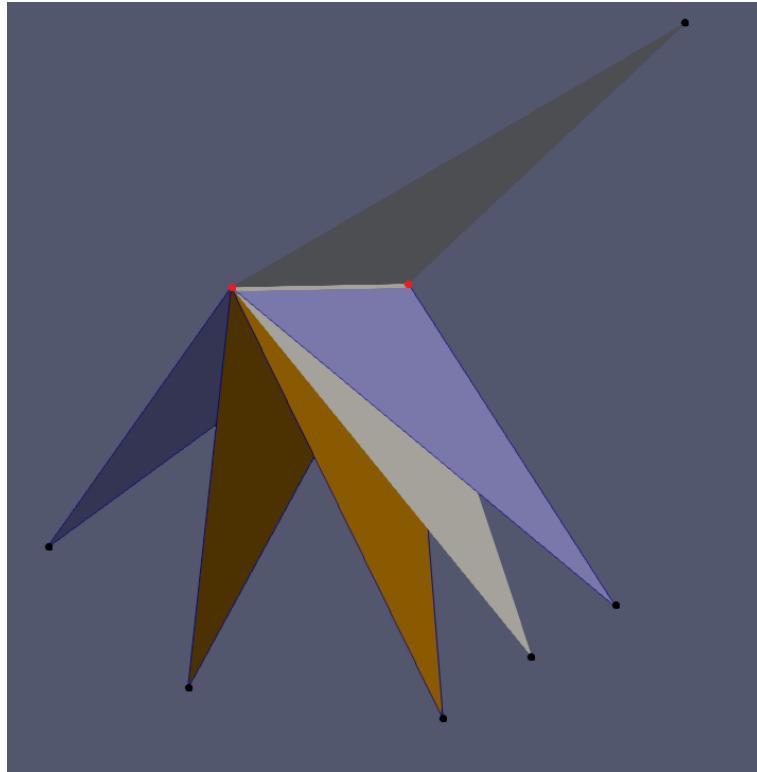


FIGURE 2.26 – Résultat de l'inversion statique du Bitriangle

On remarque que la forme au repos proposé par l'algorithme d'optimisation ressemble dans la forme globale à la forme au repos attendue, mais les sommets sont placés plus en hauteur. Néanmoins, si on lance une simulation d'affaissement à partir de cette position comme étant la position initiale et la position au repos, on retombe sur la bonne position à l'équilibre à  $10^{-9}$  près.

L'algorithme semblait donc être tombé dans le "mauvais" puit où se trouvait un autre minimum de la fonction objectif menant à cette forme au repos. En relançant l'inversion une nouvelle fois, on obtient le résultat représenté en blanc/gris sur la figure 2.27. On obtient un résultat différent, mais avec une forme similaire à la précédente, et avec une valeur de la fonction objectif très proche de l'ancienne également, montrant que l'on est proche d'un autre minimum de la fonction objectif. Après simulation d'affaissement, on retombe également très très proche de la position à l'équilibre connue.

Ceci nous a amené à réfléchir sur l'origine de ce "problème" et d'écartier les problèmes d'implémentations ou de simulation en observant que les formes au repos trouvées ont la même forme que celle attendue, mais à une rotation près autour de l'axe formé par les sommets fixés du maillage (en rouge sur les figures). En effet, l'angle dièdre (entre les deux triangles du bitriangle) est le même pour les trois formes aux repos considérées (la théoriques et celles trouvées par optimisation), et la hauteur des triangles est également la même, ce qui implique que la forme au repos est bien la même, seulement la position initiale elle va différer en fonction de l'angle de rotation autour de l'axe fixé. On peut par ailleurs montrer théoriquement que toutes les formes

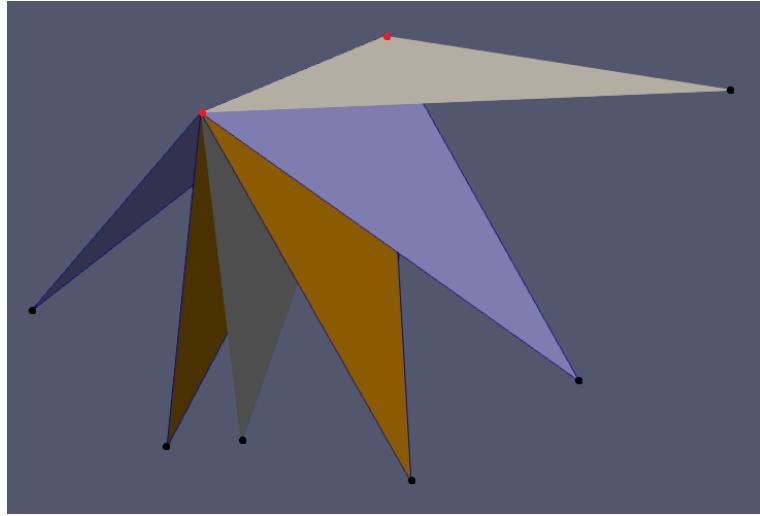


FIGURE 2.27 – Résultat de l'inversion statique du Bitriangle

au repos rotées autour de l'axe selon un angle quelconde  $\theta$  donneront la même forme à l'équilibre après simulation d'affaissement et stabilisation.

Afin d'obtenir tout de même la solution désirée, et de s'assurer qu'il n'y a pas d'autre phénomène qui intervient, l'idée de fixer une des variables du problème a été proposée. On choisit alors de fixer, pour un des sommets du bitriangle, sa coordonnée selon l'axe vertical  $y$  (l'axe  $y$  est utilisé conventionnellement par Paraview comme axe vertical). Les cinq autres variables sont libres, et toujours initialisées à partir de la position à l'équilibre, seule information connue du problème d'inversion. La figure 2.28 représente les différentes formes considérées : en jaune la forme au repos théorique, en orange la forme à l'équilibre théorique, et en verte la position initiale de l'optimisation. On remarque que sur le triangle de gauche vert, seule la coordonnée en  $y$  coincide avec la forme au repos jaune, le reste des coordonées étant égal à ceux de la forme à l'équilibre orange : le second triangle vert droit est égal au triangle orange droit.

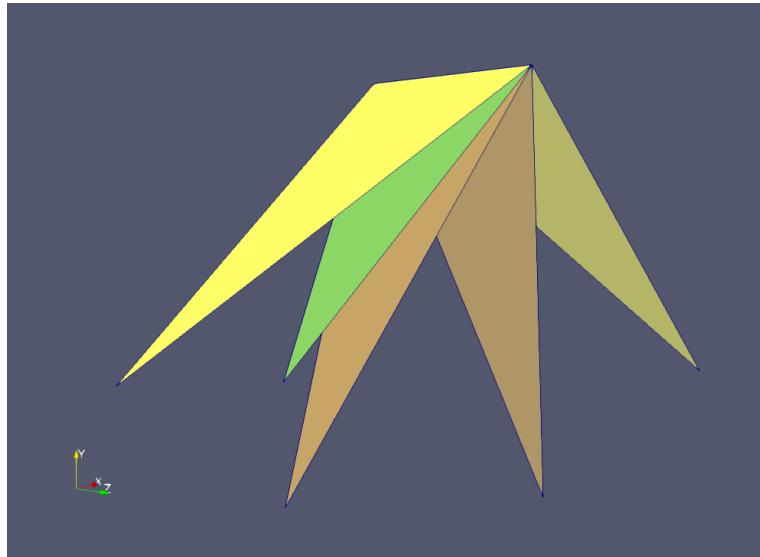


FIGURE 2.28 – Coordonnée fixée selon l'axe  $y$  – Bitriangle

En fixant cette variable afin de guider l'optimisation vers le bon minimum, on retrouve bien la forme au repos jaune avec une minimisation de la fonction objectif de l'ordre de  $10^{-8}$ . Dans le cas dynamique, ce problème apparaît moins, dans la mesure où l'on possède plus d'information sur le mouvement, et non plus seulement la position finale après stabilisation de l'affaissement.

En effet, quelque soit le placement de la fenêtre d'observation, même pour celle la plus éloignée  $[T_0, T_N] = [200, 280]$  pour laquelle on n'observe qu'une légère oscillation des triangles très proche de la position d'équilibre, on arrive à converger vers la forme au repos correcte. Le cas d'inversion dynamique du bitriangle est donc un cas très bien résolvable avec notre algorithme d'optimisation par essaim particulaire. La prochaine étape pourrait être de rajouter d'autres paramètres aux inconnues afin toujours de tester une première fois sur des exemples simples comme celui ci si notre méthode arrive à résoudre le problème et quels peuvent être les nouveaux problèmes ou interrogations soulevés.

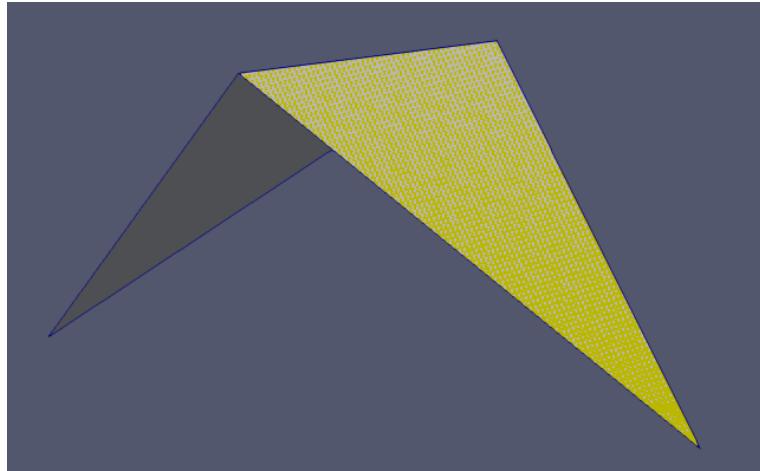


FIGURE 2.29 – Résultat de l'inversion dynamique du Bitriangle –  $T_0 = 25, T_N = 50$

### 2.7.2 Inversion dynamique du drap à 16 sommets

Une fois la méthode d'optimisation par essaim particulaire adaptée afin de résoudre le problème d'inversion dynamique de vêtement, en commençant par l'exemple du bitriangle, il paraît intéressant de choisir un second exemple de vêtement un peu plus complexe. On considère alors un drap rectangulaire composé de 16 sommets et 32 triangles.

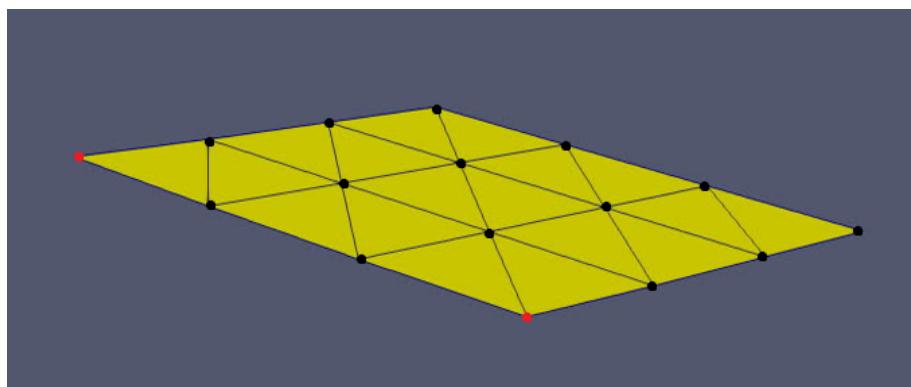


FIGURE 2.30 – Forme au repos – Drap 16

#### Etude du cas 1

Pour le cas 1 (voir [2.4.3](#)), on cherche uniquement la position initiale en connaissant la position au repos, et tous les autres paramètres. La connaissance de la bonne forme au repos implique un bonne répartition nodale de la masse lors de l'intanciation du vêtement. Cela implique aussi qu'après quelques pas de simulation, même si la position initiale est d'une forme irrégulière et très peu réaliste, le vêtement retrouvera une forme cohérente par rapport à la forme au repos (les forces et énergies internes agissent comme un rappel sur chaque sommet, ce qui redresse le vêtement).

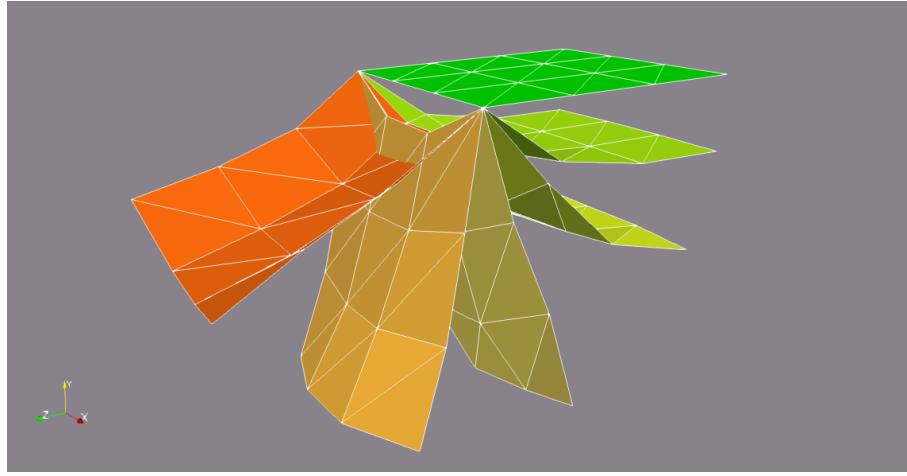


FIGURE 2.31 – Evolution de l'affaissement du drap – Drap 16

Cela montre que la convergence est difficile, car sur la majeure partie du mouvement, même si la position initiale est loin de celle attendue, le rappel des forces internes implique que la fonction objectif peut être tout de même assez "faible". Un changement sur la position initiale affectera alors très peu le reste de la simulation, et c'est pourquoi commencer au plus proche de la solution permet de gagner des itérations lors de la minimisation. On remarque pour le cas 1 par exemple, lorsque la fenêtre d'observation est la plus proche du temps 0, le résultat final après 2500 itérations de l'algorithme n'est pas exactement correspondant à la forme au repos (voir figure 2.33).

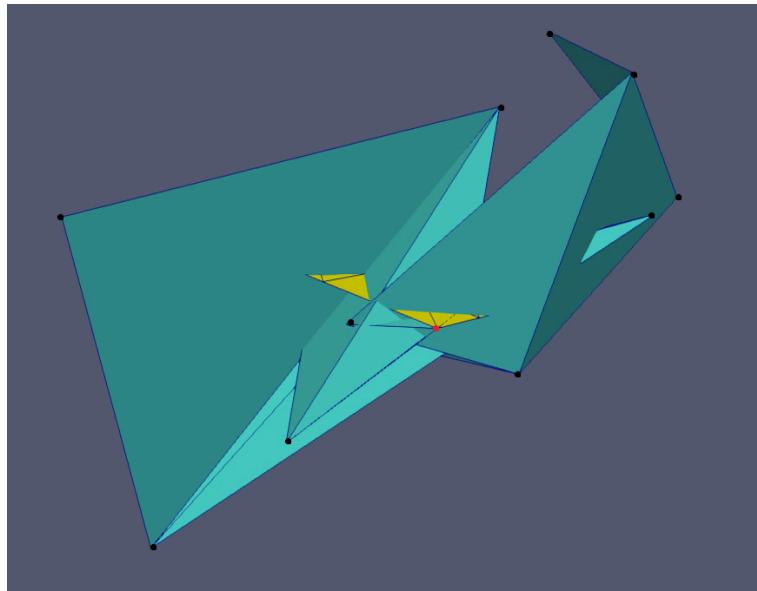


FIGURE 2.32 – Exemple d'état initial d'une particule – En bleu l'état initial, en jaune la forme au repos théorique – Drap 16

Jusqu'à présent, l'initialisation des particules se faisait de manière aléatoire. Chaque coordonnées était tirée aléatoirement et de manière uniforme dans un intervalle centré sur la même coordonnée du vecteur  $X_{obs}(T_0)$ . La taille de l'intervalle est à choisir manuellement, et va guider la forme possible de vêtement possible à l'initialisation. Cet aspect aléatoire nous amène à débuter l'optimisation avec des formes de vêtements non régulières et des surfaces discontinues.

Afin de pouvoir bénéficier d'une meilleure initialisation des particules, l'idée est d'utiliser les données à notre disposition afin de proposer une position initiale des particules donnant un vêtement plus réaliste, et plus proche de la forme au repos par ailleurs. L'idée est d'utiliser les différentes positions observées afin d'initialiser les particules. Etant donné que les positions

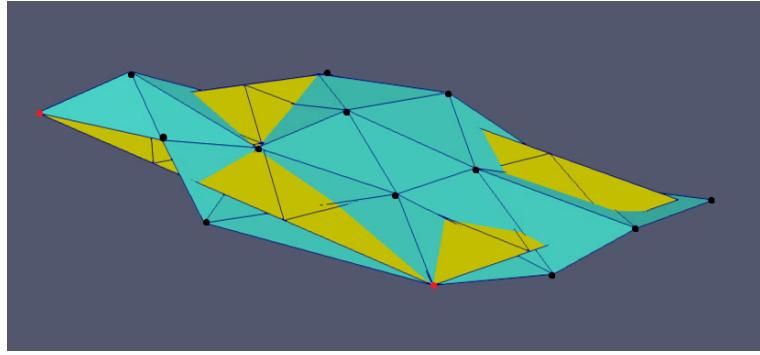


FIGURE 2.33 – Résultat de l'inversion dynamique du Drap 16, dans le cas  $1 - T_0 = 1, T_N = 20$  –  $J_1(x_0) = 0.33$

observées peuvent avoir une orientation différente de celle de la forme au repos, on ajoute la possibilité de roter le vêtement autour de l'axe fixe (formé par les deux sommets fixés) afin de pouvoir commencer plus proche de la forme au repos. Pour synthétiser, on tire au hasard une position observée, on lui applique une transformation affine de l'espace (rotation notamment) aux paramètres aléatoires (angle, rapport d agrandissement, etc.), et on obtient une position initiale possible pour une particule (voir figure 2.34).

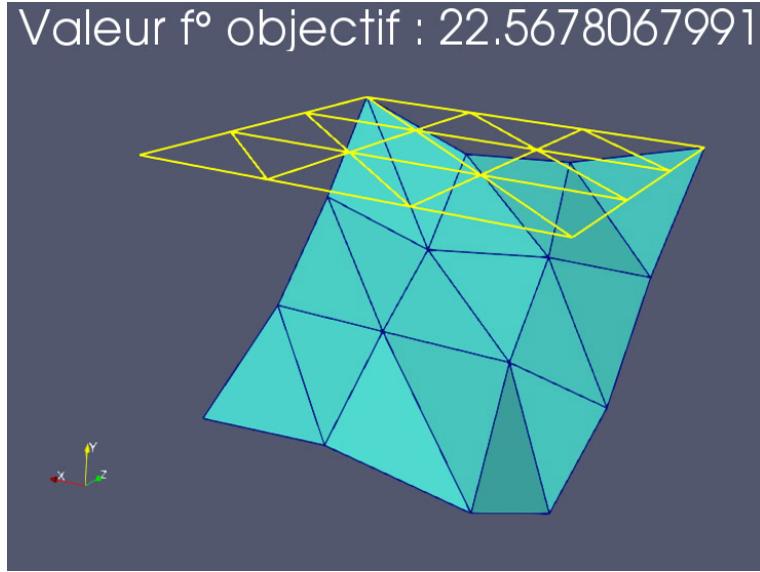


FIGURE 2.34 – Exemple d'initialisation d'une particule – Drap 16

En utilisant cette initialisation de l'optimisation, et en relançant les calculs d'optimisation pour les cas 1, on obtient une forme initiale plus proche de la forme au repos pour les deux premiers placements de la fenêtre d'observation. Comme le montre la figure 2.35, si on simule un affaissement à partir de la position initiale calculée, l'écart est très faible entre le réel (en jaune) et celui trouvé par notre optimiseur (en bleu).

Cependant, lorsqu'on décale encore la fenêtre d'observation pour avoir un mouvement de plus faible amplitude du vêtement (proche de l'équilibre), il devient assez difficile d'obtenir une forme initiale aussi rectangulaire que la forme au repos. Cela est dû au fait que le vêtement "a le temps" de reprendre une forme correcte (car nous avons renseigné la bonne forme au repos) avant que l'on entre dans la fenêtre d'observation. Par exemple, on trouve pour une fenêtre d'observation égale à [200, 280] la forme initiale représentée en figure 2.36. Si on réalise une simulation d'affaissement, et que l'on compare la simulation réelle (en jaune) et celle obtenue à partir de notre solution (en bleu) (figure 2.37), on voit qu'ils sont très proche au final, malgré la différence de forme initiale du vêtement. L'ajout d'informations concernant la forme initiale

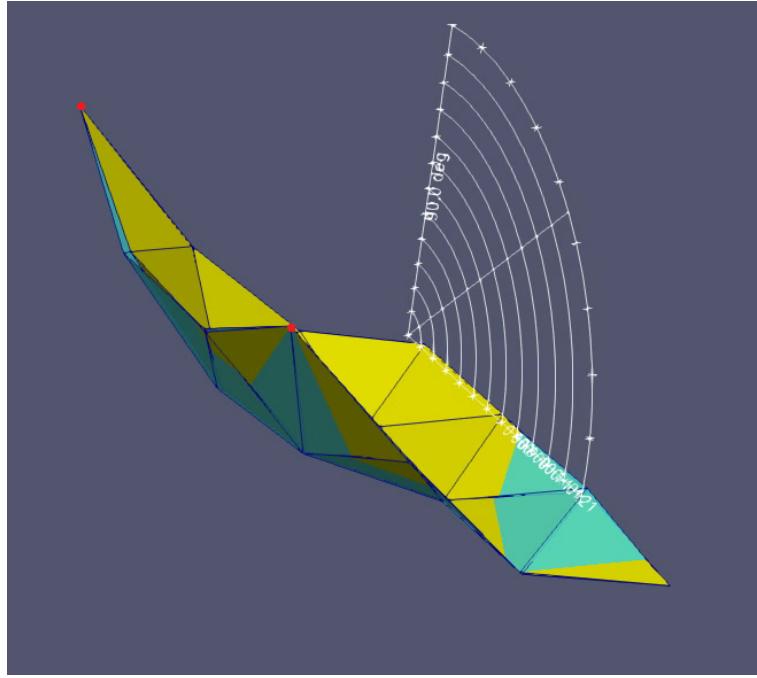


FIGURE 2.35 – Simulation d'affaissement après inversion dynamique du Drap 16, dans le cas 1 –  $T_0 = 5, T_N = 25 - J_1(x_0) = 0.076$

pourraient guider la recherche vers des formes plus proches de la forme au repos : c'est l'objet de l'étude des cas 4 et 5.

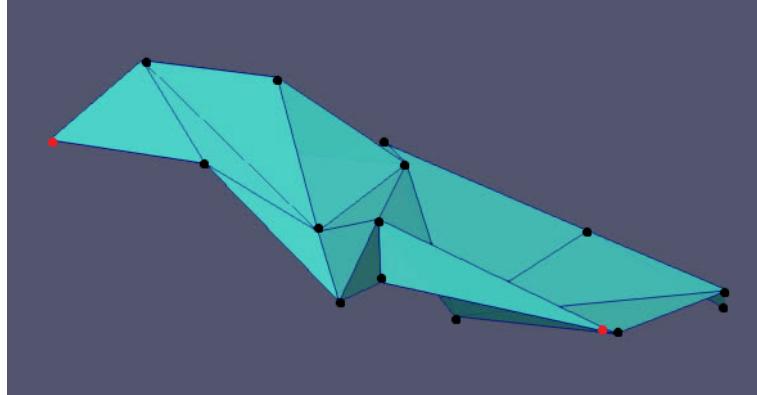


FIGURE 2.36 – Résultat de l'inversion dynamique du Drap 16, dans le cas 1 –  $T_0 = 200, T_N = 280 - J_1(x_0) = 0.0197$

#### Etude du cas 4 et 5

En ajoutant le fait que la forme initiale de la simulation est aussi la forme au repos du vêtement comme information au problème, la résolution de problème est d'autant plus guidée. En effet, si à présent la forme initiale est mauvaise, comme elle est aussi la forme au repos, le forme du vêtement ne retrouvera pas de forme cohérente (par rapport à la forme théorique du vêtement) au cours de la simulation. Cela réduit donc les solutions possibles, et la fonction objectif en est modifiée par conséquent.

Les résultats obtenus, en ajoutant cette considération pour la résolution du problème d'inversion (cas 4 voir [2.4.3](#)), sont assez concluants dans l'ensemble. Dans le cas où la fenêtre d'observation est assez proche du début de la simulation ([figure 2.38](#)), la forme au repos est très proche de celle attendue, et la fonction objectif donc très bien minimisée. Cependant, plus on éloigne la fenêtre d'observation, plus la forme au repos commence à s'éloigner de celle attendue,

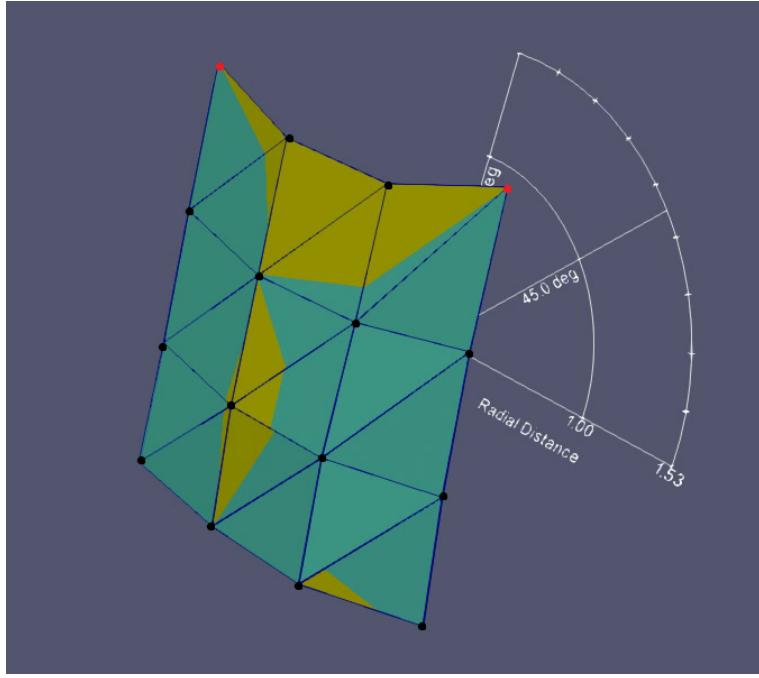
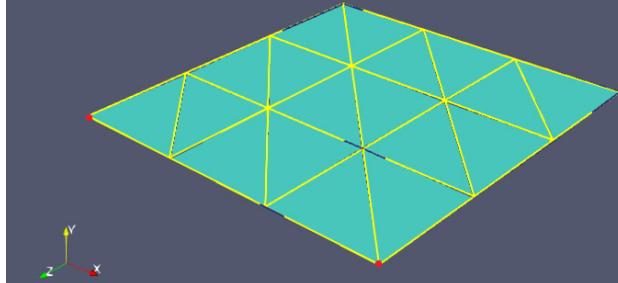


FIGURE 2.37 – Simulation d'affaissement après inversion dynamique du Drap 16, dans le cas 1  
–  $T_0 = 200, T_N = 280 - J_1(x_0) = 0.0197$

sans pour autant prendre une forme non cohérente, ou sans que cela n'affecte grandement l'écart observé après une comparaison des simulations d'affaissement.

Valeur  $f^\circ$  objectif : 0.00320801341939



Valeur  $f^\circ$  objectif : 0.0292538805876

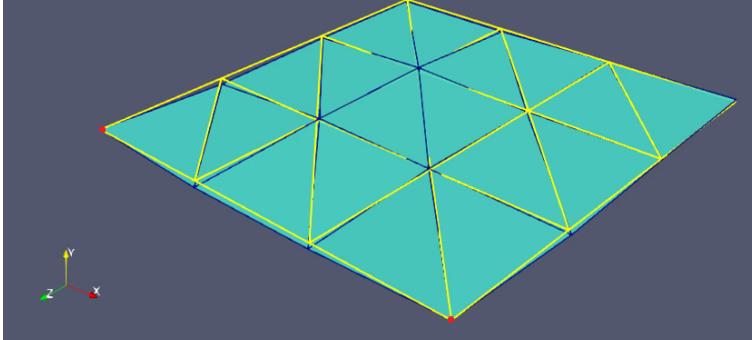


FIGURE 2.38 – Résultat de l'inversion dynamique du Drap 16, dans le cas 4 –  $T_0 = 1, T_N = 20$   
(gauche) –  $T_0 = 5, T_N = 25$

L'exemple le plus intéressant pour illustrer cela est celui correspondant au placement de la fenêtre d'observation à [200, 280]. En effet, la position au repos/initiale trouvée est plus basse que celle attendue, et légèrement plus étendue. Lorsque nous lançons la simulation d'affaissement, le vêtement a la même période de balancement que pour la réelle simulation, puis l'amplitude décroît jusqu'à ce que les deux vêtements soient très proches. L'intervalle d'observation commence justement au moment où les deux vêtements sont proches, ceci induisant une certaine minimisation de la fonction objectif tout de même (mais pas encore optimale) (voir figure 2.40).

Concernant le cas numéro 5, où la répartition de la masse sur chaque noeud du vêtement se fait à partir de la forme au repos/initiale proposée, nous pensons que cela rend le problème plus difficile, mais permet à la fois d'élaguer un certain nombre de configurations de forme au repos non consistantes. Une réflexion plus approfondie, et l'étude d'exemples précis dans ce cas

Value is: 0.344806171478

Valeur f° objectif : 4.27344377943

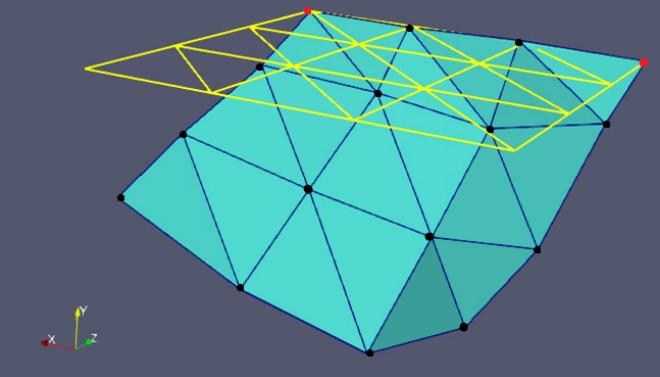
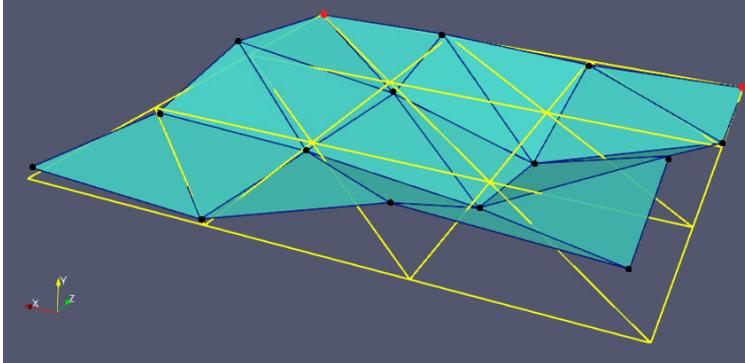


FIGURE 2.39 – Résultat de l'inversion dynamique du Drap 16, dans le cas 4 –  $T_0 = 25, T_N = 50$  (gauche) –  $T_0 = 200, T_N = 280$

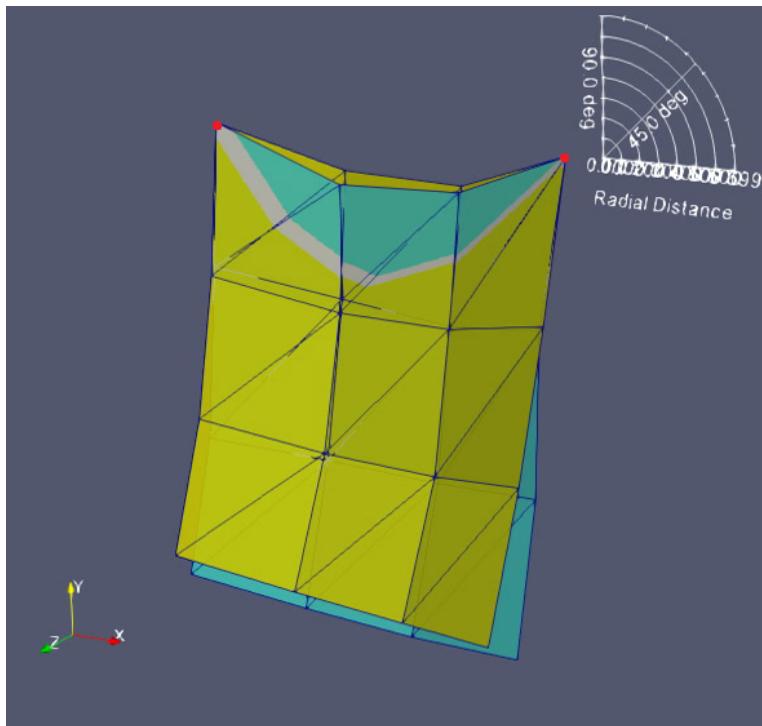


FIGURE 2.40 – Simulation d'affaissement après inversion dynamique du Drap 16, dans le cas 4 –  $T_0 = 200, T_N = 280$

permettra d'avoir une meilleure idée sur l'influence de l'ajout de cette contrainte. Les résultats obtenus avec une inversion dynamique du drap ont laissé paraître de meilleurs résultats pour les deux premières positions de fenêtres proches ([1,20] et [5,25]), et également pour la fenêtre la plus éloignée ([200,280]). Les calculs n'ont pas été lancés durant une longue durée, et donc le nombre d'itérations peut être moindre par rapport aux autres cas étudiés. On peut donc présager une meilleure minimisation de la fonction objectif dans le cas 5, mais cela reste encore à illustrer et démontrer.

### 2.7.3 Validation des résultats

Comme cela a pu apparaître dans la présentation des résultats, la validation ou la qualité des résultats obtenus après inversion statique ou dynamique a toujours été faite de manière visuelle, ou en considérant la valeur de fonction objectif sans toutefois posséder de critères discriminants et qualitatifs. Après avoir réalisé un certain nombre d'inversions dynamiques, on peut situer

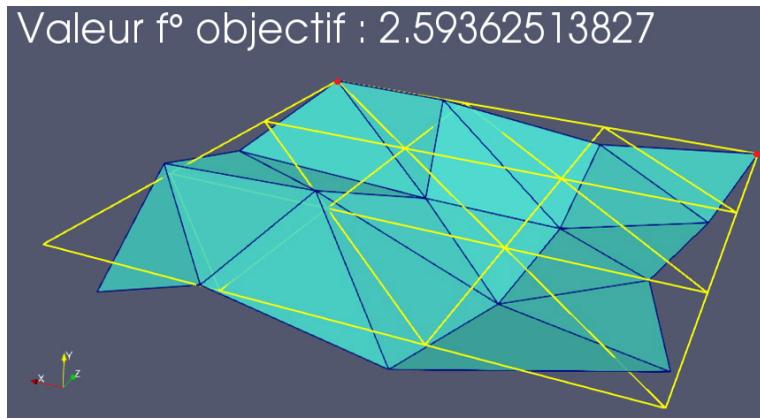


FIGURE 2.41 – Résultat de l'inversion dynamique du Drap 16, dans le cas 5 –  $T_0 = 200, T_N = 280$

environ à 0.03 la valeur de la fonction objectif à partir de laquelle on ne pourra observer de grande différence dans la simulation d'affaissement entre le réel et le calculé.

Cependant il peut être intéressant de déterminer un certain nombre de critères ou de seuils, comprenant plusieurs variables du problème (taille du vêtement,, etc.) et liés à la visualisation (position caméra, champ de vision, etc.), afin de pouvoir automatiser le processus de validation des résultats. Cela est une question générale pour toutes les analyses de résultats, et souvent l'argument visuel souvent suffisant en informatique graphique ne l'est pas forcément dans le domaine de la mécanique et de l'assimilation de données.

# Chapitre 3

# Conclusion

Ce dernier chapitre de conclusion viens clore ce rapport de projet de fin d'étude. Un synthèse des résultats et travaux menés durant le stage sera présentée, suivie d'une présentation des perspectives futures et pistes de continuité envisagées. Enfin, un bilan personnel de ce stage sera présenté.

## 3.1 Synthèse des résultats

Durant 6 mois, j'ai pu travailler sur le sujet de modélisation inverse dynamique de vêtements. Ce fut l'occasion d'appréhender pour la première fois ce genre de problématique, pour ma part, mais aussi pour mes encadrants. Le but du projet est de permettre l'estimation de paramètres géométriques et physiques de vêtements, tout en reliant différents domaines d'étude : mécanique et simulation physique, informatique graphique et vision par ordinateur, et assimilation de données.

Une expression du problème a pu être proposée, sous la forme d'un problème d'optimisation d'une fonction objectif. Cette optimisation a pu être mise en place en implémentant des méthodes d'optimisation sans gradient. Des calculs d'inversion ont donc été lancés dans différentes configurations des paramètres et des inconnues. L'expression du gradient de la fonction objectif a pu être établi, permettant de pouvoir implémenter par la suite des méthodes d'optimisation avec gradient.

De premiers résultats d'inversion on donc pu être obtenus, et des conclusions fondamentales ont pu être tirées de ces exemples traités. Ce travail d'investigation a également permis de dégager un certain nombre de pistes et de perspectives pour aborder plus en profondeur le sujet, et lier les différentes compétences des protagonistes pour proposer un premier résultat robuste pour certains cas.

## 3.2 Perspectives

À la fin de mon stage, je me suis principalement intéressé au calcul du gradient de la fonction objectif représentant notre problème d'inversion dynamique. Une des premières poursuites possible du projet serait d'effectivement implémenter le calcul de ce gradient de manière algorithmique. Des méthodes d'optimisation avec gradient pourraient ensuite être sélectionnée pour être, à leur tour, implémentées et appliquées aux mêmes cas considérés durant ce stage (voir [2.7.2](#)). De même, une hybridation des méthodes avec et sans gradient peut être envisagée pour permettre un meilleure résolution du problème, bénéficiant à la fois d'une recherche globale en début d'algorithme, et de la vitesse de convergence des méthodes basées gradient (une fois dans le bon puit).

Aussi, dans le cadre du stage, nos observations de vêtements ne provenaient exclusivement que du simulateur Cloth. Pour ne pas rester dans une manière uniquement théorique d'aborder le sujet, il paraît intéressant de rapidement entamer la réflexion du point de vue de l'acquisition

du mouvement réel des vêtements. Ce sera donc l'occasion de se rapprocher des techniques connues en vision par ordinateur et des problématiques liées, et de les prendre rapidement en considération pour la suite du projet. De nombreuses informations induites des observations et captures pourraient ainsi permettre de guider ou simplifier le problème d'inversion, peut être plus qu'en disposant uniquement de maillages du simulateur. Cependant, d'autres problèmes liés à la capture, comme la correspondance entre les sommets dans une séquence de maillages, pourraient apparaître et changer la nature des investigations.

Afin également d'avoir une première idée de l'influence de chaque paramètre physique lié au vêtement (masse, raideur en torsion, flexion et cisaillement), on peut considérer le cas où l'on connaît les paramètres géométriques du vêtements, et on cherche à déterminer un paramètre à la fois en minimisant la même fonction objectif.

D'un point de vue plus théorique, une perspective possible serait de s'intéresser plus en détail à la nature mathématique de la fonction objectif et de l'analyser plus en profondeur. L'ajout de termes, ou la considération d'une nouvelle fonction objectif différente peuvent être envisagés afin de permettre une réduction des minimas locaux ou de se rapprocher d'une forme plus convexe notamment.

Une réflexion plus profonde sur les résultats actuels peut également être menée. Les résultats lors de phénomènes d'oscillation observés durant les simulation d'affaissement pourraient être exploités pour établir certaines règles qui guideraient la recherche une fois ce genre de phénomènes détectés. Par ailleurs, une réflexion peut être menée quant au placement de la fenêtre d'observation, et comment le lieu de son placement par rapport au temps de total de simulation peut influencer la difficulté ou guider la résolution du problème d'inversion.

Ainsi, de nombreuses perspectives, et dans des directions et domaines variés, sont envisageables à l'issue de ce stage. Dans une certaine mesure, ce stage a pu permettre de faire le premier pas dans ce projet, et de mettre en évidence les considérations et difficultés liées à ce problème, et pourra je l'espère permettre à mes encadrants et successeurs de poursuivre encore plus loin les recherches dans ce domaine et de parvenir à des résultats concluants.

### 3.3 Bilan personnel

Tout au long de ce stage, j'ai pu évoluer et développer de nouvelles compétences techniques et humaines. Ce fut pour moi une expérience exclusivement positive, et j'ai pu apprendre beaucoup tant sur le plan scientifique et technique, que sur le plan humain.

Tout d'abord, j'ai pu consolider mes connaissances dans le domaine de l'informatique graphique et de la modélisation 3D, domaine constituant ma spécialité dans le département informatique. J'ai pu également renforcer mes compétences en mathématiques, physique et mécanique, tout en apprenant à adapter mes connaissances en programmation pour appréhender de nouveaux langages ou environnements de développement informatique.

Ce stage m'a aussi permis de découvrir de nouveaux domaines scientifiques. Ainsi, j'ai pu être initié à l'optimisation numérique et parvenir à implémenter et tester des méthodes d'optimisation. Ce fut aussi l'occasion pour moi d'avoir un premier contact avec la modélisation mécanique, notamment celle rapportée aux formes élancées. Découvrir comment on peut modéliser un vêtement à l'aide d'un modèle de coque par exemple, ou un cheveux avec le modèle de tiges de Kirchhoff, a été très enrichissant. Ce stage fut également une première expérience professionnelle pour moi dans le domaine de la simulation et de l'informatique graphique. Enfin, j'ai pu découvrir de nouveaux concepts et méthodes mathématiques d'analyse, d'analyse numérique et de géométrie.

Au cours de ce stage j'ai pu développer un certain nombre de compétences professionnelles également. Restituer oralement et de manière écrite les informations préalablement synthétisées, communiquer les informations importantes, organiser des réunions et le temps de travail, développer une automie et anticipation dans le travail, rechercher et demander l'aide efficacement, et coopérer avec les autres membres du groupe sont des capacités professionnelles que j'ai pu perfectionner pour certaines ou acquérir pour d'autres grâce à ce stage.

Le sujet et l'environnement du stage étant étroitement liés au monde de la recherche scientifique, j'ai ainsi eu la chance d'y évoluer durant 6 mois et de pouvoir me faire une idée plus précise sur le métier de chercheur et ses aspects périphériques. J'ai pu me familiariser plus amplement avec la méthodologie employée pour la recherche, cotoyer divers profils d'équipes, chercheurs et doctorants. Evoluer au sein de l'institut de l'Inria m'a permis d'assister à de nombreuses conférences, des séminaires, et des soutenances de thèse et de HDR (habilitation à diriger des recherches). De plus, j'ai pu mieux comprendre les étapes de restitutions des recherches sous la forme d'articles scientifiques notamment. Enfin, l'institut de recherche étant proche de l'université de Grenoble, j'ai également eu l'occasion d'en savoir plus sur les aspects liés à l'enseignement et le monde académique. J'ai vraiment apprécié découvrir comment le site de Montbonnot-Saint-Martin, où s'est déroulé mon stage, était engagé à différentes échelles dans des problématiques de premier ordre, et comme le site était aménagé afin de faciliter le cadre de travail des chercheurs et ingénieurs.

Ce stage est pour moi la plus grande expérience professionnelle que j'ai vécue jusqu'à présent, et cela m'a permis de préciser mon projet professionnel futur entre autres. Ayant en effet évolué au sein d'une équipe de recherche, et cotoyant le monde de la recherche au laboratoire, j'ai pu me faire une idée précise du travail que j'aimerais exercer plus tard. C'est un projet de fin d'étude de qualité, sur tous les points, et j'en ressors grandit techniquement et humainement, prêt à poursuivre mes études dans le cadre d'une thèse, et sous un tout nouvel angle d'attaque.

# Références - Bibliographie

## Etat de l'art

- [1] Jovic, N. and Huang, T. S. Estimating Cloth Draping Parameters from Range Data. In International Workshop on Synthetic-Natural Hybrid Coding and 3-D Imaging, 1997, 73-76.
- [2] Bhat, K. S. ; Twigg, C. D. ; Hodgins, J. K. ; Khosla, P. K. ; Popovic, Z. and Seitz, S. M. Breen, D. and Lin, M. (Eds.). Estimating Cloth Simulation Parameters from Video. Symposium on Computer Animation, The Eurographics Association, 2003.
- [3] David Pritchard. M. Cloth Parameters and Motion Capture. Master Thesis. UBC, UBC, 2003.
- [4] Magnenat-Thalmann, N. ; Luible, C. ; Volino, P. and Lyard, E. From Measured Fabric to the Simulation of Cloth. 2007 10th IEEE International Conference on Computer-Aided Design and Computer Graphics, 2007, 7-18.
- [5] Kunitomo, S. ; Nakamura, S. and Morishima, S. Optimization of cloth simulation parameters by considering static and dynamic features. ACM SIGGRAPH 2010 Posters, SIGGRAPH '10, 2010.
- [6] Wang, H. ; Ramamoorthi, R. and O'Brien, J. F. Data-Driven Elastic Models for Cloth : Modeling and Measurement. ACM Transactions on Graphics, 2011, 30, 71 :1-11.
- [7] Miguel, E. ; Bradley, D. ; Thomaszewski, B. ; Bickel, B. ; Matusik, W. ; Otaduy, M. A. and Marschner, S. Data-Driven Estimation of Cloth Simulation Models. Computer Graphics Forum (Proc. of Eurographics), 2012, 31, 10.
- [8] Katherine L. Bouman ; Bei Xiao ; Peter Battaglia and William T. Freeman. Estimating the Material Properties of Fabric from Video. 2013 IEEE International Conference on Computer Vision (ICCV), IEEE Computer Society, 2013, 00, 1984-1991
- [9] Davis, A. ; Bouman, K. L. ; Chen, J. G. ; Durand, M. R. F. and Freeman, W. T. Visual Vibrometry : Estimating Material Properties from Small Motions in Video. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2015, 5335-5343
- [10] Moradi, M. ; Naraghi, M. and Nikoobin, A. Parameter identification of nonlinear systems using indirect solution of optimal control problem. Robotics and Mechatronics (ICROM), 2015 3rd RSI International Conference on Robotics and Mechatronics, IEEE, 2015
- [11] Casati, R. Quelques contributions à la modélisation numérique de structures élancées pour l'informatique graphique. Inria Grenoble Alpes. Thesis. 2015
- [12] Casati, R. ; Daviet, G. and Bertails-Descoubes, F. Inverse Elastic Cloth Design with Contact and Friction. Inria Grenoble Rhône-Alpes, Université de Grenoble, Inria Grenoble Rhône-Alpes, Université de Grenoble, 2016.
- [13] Jinlong Yang, Jean-Sébastien Franco, Franck Hétroy-Wheeler, Stefanie Wuhrer. Estimation of Human Body Shape in Motion with Wide Clothing. Bastian Leibe ; Jiri Matas ; Nicu Sebe ; Max Welling. ECCV 2016 - European Conference on Computer Vision 2016, Oct

2016, Amsterdam, Netherlands. Springer, 9908 (Part IV), pp.439-454, Lecture Notes in Computer Science.

- [14] Aurela Shehu, Jinlong Yang, Jean-Sébastien Franco, Franck Hétroy-Wheeler, Stefanie Wuhrer. Computing temporal alignments of human motion sequences in wide clothing using geodesic patches. 3DV 2016 - International Conference on 3D Vision 2016, Oct 2016, Stanford, United States. IEEE, pp.185-193, 2016.
- [15] Stoll, C., Gall, J., de Aguiar, E., Thrun, S., Theobalt, C. : Video-based reconstruction of animatable human characters. ACM Transactions on Graphics 29 (6) (2010). 139 :1–10 Proceedings of SIGGRAPH Asia.
- [16] Neophytou, A., Hilton, A. : A layered model of human body and garment deformation. In : International Conference on 3D Vision. (2014) 171–178.
- [17] Pons-Moll, G., Romero, J., Mahmood, N., Black, M. : DYNA : a model of dynamic human shape in motion. Transactions on Graphics 34 (4) (2015) 120 :1–14 Proceedings of SIGGRAPH.
- [18] Meekyoung Kim, Gerard Pons-Moll, Sergi Pujades, Seungbae Bang, Jinwook Kim, Michael Black, and Sung-Hee Lee. Data-Driven Physics for Human Soft Tissue Animation. ACM Transaction on Graphics (Proceedings of SIGGRAPH 2017). Volume 36, Number 4, 2017.
- [19] Bogo, F. ; Romero, J. ; Pons-Moll, G. and Black, M. J. Dynamic FAUST : Registering Human Bodies in Motion. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2017.
- [20] Gerard Pons-Moll, Sergi Pujades, Sonny Hu, and Michael J. Black. 2017. ClothCap : seamless 4D clothing capture and retargeting. ACM Trans. Graph. 36, 4, Article 73 (July 2017), 15 pages. DOI : <https://doi.org/10.1145/3072959.3073711> .
- [21] Philippe G. Ciarlet : Mathematical elasticity. vol. 3 : theory of shells. Studies in mathematics and its applications. Elsevier Science, Amsterdam, New York, Oxford, 2000. ISBN 0-444-82891-5.
- [22] Philippe G. Ciarlet : An Introduction to Differential Geometry with Applications to Elasticity. Available online. Springer, 2006. ISBN 9781402042485.
- [23] Rasmus Tamstorf et Eitan Grinspun : Discrete bending forces and their jacobians. Graphical Models, 75(6) :362 – 370, 2013. ISSN 1524-0703. URL <http://www.sciencedirect.com/science/article/pii/S1524070313000209>.
- [24] Baraff, D. and Witkin, A. Large Steps in Cloth Simulation. Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, ACM, 1998, 43-54.
- [25] Grinspun, E. ; Hirani, A. N. ; Desbrun, M. and Schröder, P. Discrete Shells. Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, Eurographics Association, 2003, 62-67.

## Problèmes inverses

- [26] Michel Kern. Problèmes inverses : aspects numériques. Engineering school. 1999 à 2002, Ecole supérieure d'ingénieurs Léonard de Vinci, 2002, pp.138. <https://hal.archives-ouvertes.fr/cel-00168393v2/document>
- [27] G.Allaire. Conception optimale de structures. Chapitre 3, Rappels d'optimisation. 13 Janvier 2010. Département de Mathématiques Appliquées. <http://www.cmap.polytechnique.fr/~allaire/map562/amphi2.pdf>

- [28] Marc Bonnet. Problèmes Inverses. Master recherche à l'Ecole Centrale de Paris. Mention Matière, Structures, Fluides, Rayonnement. Spécialité Dynamique des Structures et Systèmes Couplés. Octobre 2008. <http://perso.ensta-paristech.fr/~mbonnet/invdssc.pdf>

## Optimisation

- [29] T.Dumont, C.Leonard, X.Mary, H.Mohamed. Cours d'optimisation. L1 Eco - Analyse 2. Université Paris Ouest - Nanterre - La Défense. <http://leonard.perso.math.cnrs.fr/teaching/L1%20sceco-analyse%202-notes%20de%20cours.pdf>
- [30] Ionel Sorin CIUPERCA. Cours Optimisation. Cours à l'ISFA, en M1SAF. <http://math.univ-lyon1.fr/~ciuperca/cours-optim-M1SAF.pdf>
- [31] Xavier Antoine, Pierre Dreyfuss, Yannick Privat. Introduction à l'optimisation : aspects théoriques, numériques et algorithmes. ENSMN-ENSEM 2A (2006-2007). <https://www.1j11.math.upmc.fr/privat/documents/mainOptimisation.pdf>

## Optimisation sans gradient

- [32] [http://cours.mido.dauphine.fr/opti\\_num/cours>Note%20Golden%20Search.pdf](http://cours.mido.dauphine.fr/opti_num/cours>Note%20Golden%20Search.pdf)
- [33] [https://fr.wikipedia.org/wiki/M%C3%A9thode\\_du\\_nombre\\_d%27or](https://fr.wikipedia.org/wiki/M%C3%A9thode_du_nombre_d%27or)
- [34] <https://www.1j11.math.upmc.fr/privat/documents/optimENSEM.pdf>
- [35] <http://icube-avr.unistra.fr/fr/images/5/5a/Chapitre2.pdf>
- [36] [https://en.wikipedia.org/wiki/Simplex\\_algorithm](https://en.wikipedia.org/wiki/Simplex_algorithm)
- [37] [https://en.wikipedia.org/wiki/Nelder%E2%80%93Mead\\_method](https://en.wikipedia.org/wiki/Nelder%E2%80%93Mead_method)
- [38] <https://team.inria.fr/opale/files/2011/11/0cet.pdf>
- [39] <http://icube-avr.unistra.fr/fr/images/7/7d/Chapitre3.pdf>
- [40] <https://deptmedia.cnam.fr/new/spip.php?pdoc5709>
- [41] [https://en.wikipedia.org/wiki/Simulated\\_annealing](https://en.wikipedia.org/wiki/Simulated_annealing)
- [42] <http://evelyne.lutton.free.fr/SlidesCours/SlidesCoursHeuristiques-Agro.pdf>
- [43] [https://fr.wikipedia.org/wiki/Algorithme\\_g%C3%A9n%C3%A9tique](https://fr.wikipedia.org/wiki/Algorithme_g%C3%A9n%C3%A9tique)
- [44] [https://fr.wikipedia.org/wiki/Algorithme\\_de\\_colonies\\_de\\_fourmis](https://fr.wikipedia.org/wiki/Algorithme_de_colonies_de_fourmis)
- [45] [http://www.eyrolles.com/Chapitres/9782212113686/chap4\\_Dreo.pdf](http://www.eyrolles.com/Chapitres/9782212113686/chap4_Dreo.pdf)
- [46] [https://fr.wikipedia.org/wiki/Algorithme\\_%C3%A0\\_estimation\\_de\\_distribution](https://fr.wikipedia.org/wiki/Algorithme_%C3%A0_estimation_de_distribution)
- [47] <http://slideplayer.fr/slide/1212272/>
- [48] [https://fr.wikipedia.org/wiki/Strat%C3%A9gie\\_d%C3%A9volution](https://fr.wikipedia.org/wiki/Strat%C3%A9gie_d%C3%A9volution)
- [49] [https://fr.wikipedia.org/wiki/Recherche\\_tabou](https://fr.wikipedia.org/wiki/Recherche_tabou)
- [50] [https://www.cmi.univ-mrs.fr/~preaux/PDF/pdf\\_proteges/OptimisationCombinatoire/Metaheuristiques1.pdf](https://www.cmi.univ-mrs.fr/~preaux/PDF/pdf_proteges/OptimisationCombinatoire/Metaheuristiques1.pdf)
- [51] [http://ireboot.u-bourgogne.fr/master1/mi1-tc5/CM2009/Tabou/Tabou\\_1.pdf](http://ireboot.u-bourgogne.fr/master1/mi1-tc5/CM2009/Tabou/Tabou_1.pdf)
- [52] [https://fr.wikipedia.org/wiki/Greedy\\_randomized\\_adaptive\\_search\\_procedure](https://fr.wikipedia.org/wiki/Greedy_randomized_adaptive_search_procedure)

- [53] <https://pdfs.semanticscholar.org/0b1b/6c7d0ed5cdf66b8cba1199e473ddf0eb0cc6.pdf>
- [54] <http://www.isima.fr/f4/projets2010/bombrun.pdf>
- [55] [http://guillaume.calas.free.fr/data/Publications/PSO-Overview\\_v2.pdf](http://guillaume.calas.free.fr/data/Publications/PSO-Overview_v2.pdf)
- [56] Yann Cooren. Perfectionnement d'un algorithme adaptatif d'Optimisation par Essaim Particulaire. Applications en génie médical et en électronique. Thèse de Doctorat de l'Université de Paris 12 Val de Marne. 27 novembre 2008.
- [57] Maurice Clerc, Patrick Siarry. Une nouvelle métaheuristique pour l'optimisation difficile : la méthode des essaims particulaires. J3eA, Volume 3, 2004. DOI : 10.1051/bib-j3ea:2004007.
- [58] <https://rfia2012.files.wordpress.com/2011/12/la-mc3a9thode-kangourou.pdf>
- [59] <https://reussirleme1info.files.wordpress.com/2013/02/exposc3a9-kangourou.pptx>
- [60] <http://www.info.univ-angers.fr/pub/hao/papers/RIA.pdf>
- [61] [https://en.wikipedia.org/wiki/Stochastic\\_tunneling](https://en.wikipedia.org/wiki/Stochastic_tunneling)

## Fonctions test et Performance Profile

- [62] <https://www.sfu.ca/~ssurjano/optimization.html>
- [63] [https://en.wikipedia.org/wiki/Test\\_functions\\_for\\_optimization](https://en.wikipedia.org/wiki/Test_functions_for_optimization)
- [64] Mohammad Sofi, Azfizaidi and Mamat, Mustafa and Zaharah Mohid, Siti and Asrul, Mohd and Ibrahim, Mohd Asrul Hery and Khalid, Nurkaliza. (2015). Performance Profile Comparison Using Matlab. [https://www.researchgate.net/publication/301625932\\_Performance\\_Profile\\_Comparison\\_Using\\_Matlab](https://www.researchgate.net/publication/301625932_Performance_Profile_Comparison_Using_Matlab)
- [65] Dolan, E. and Moré, J. Math. Program. (2002) Benchmarking optimization software with performance profiles. 91 : 201. <https://doi.org/10.1007/s101070100263>.

## Parallélisation naïve en Python

- [66] <https://docs.python.org/2/library/multiprocessing.html>
- [67] <https://blog.dominodatalab.com/simple-parallelization/>
- [68] <https://pythonhosted.org/joblib/parallel.html>
- [69] <http://matthewrocklin.com/blog/work/2013/12/05/Parallelism-and-Serialization>
- [70] <http://qingkaikong.blogspot.fr/2016/12/python-parallel-method-in-class.html>

## Assimilation de données

- [71] Eric Blayo, Maëlle Nodet. Introduction à l'assimilation de données variationnelle. 27 mars 2012. <https://team.inria.fr/moise/files/2012/03/Methodes-Inverses-Var-M2-math-2009.pdf>
- [72] B. Sportisse, D. Quélo. Assimilation de données. 1ère Partie : Eléments théoriques. Cerea. [http://cerea.enpc.fr/fich/doc\\_ENSTA\\_da1.pdf](http://cerea.enpc.fr/fich/doc_ENSTA_da1.pdf)
- [73] Nelson Feyeux. Transport optimal pour l'assimilation de données images. Mathématiques. Université de Grenoble Alpes, 2016. Français. jtel-01480695;

## Méthode de l'adjoint

- [74] Eric Blayo, Maëlle Nodet, Arthur Vidard. Introduction to data assimilation. [http://lgge.osug.fr/meom/pages-perso/cosme/doc\\_cours/ASSIM/Lecture\\_notesAV.pdf](http://lgge.osug.fr/meom/pages-perso/cosme/doc_cours/ASSIM/Lecture_notesAV.pdf)
- [75] Eric Blayo, Maëlle Nodet, Arthur Vidard. Variational data assimilation algorithms and practical implementation. [http://lgge.osug.fr/meom/pages-perso/cosme/doc\\_cours/ASSIM/NodetLectureNotes2016.pdf](http://lgge.osug.fr/meom/pages-perso/cosme/doc_cours/ASSIM/NodetLectureNotes2016.pdf)
- [76] Didier Auroux. Problèmes inverses pour l'environnement. Printemps 2015. <http://math.unice.fr/~auroux/EPU/PI5.pdf>
- [77] Eric Blayo. Méthode adjointe pour l'analyse de sensibilité. Université de Grenoble et INRIA. Ecole ASPEN, 7-12 avril 2013. <http://aspen.forge.imag.fr/2013/Support/Slides/sensibilite-adjoint.pdf>
- [78] Mehdi Laroussi. Analyse de sensibilité 3D par la méthode de l'état adjoint : application au forgeage. Mécanique [physics.med-ph]. Ecole Nationale Supérieure des Mines de Paris, 2003. Français. jtel-00005755; <https://pastel.archives-ouvertes.fr/tel-00005755>
- [79] Isabelle Charpentier, Mohammed Ghémirès. Génération automatique de codes adjoints : Stratégies d'utilisation pour le logiciel Odyssée, Application au code météorologique Meso-NH. [Rapport de recherche] RR-3251, INRIA. 1997. jinria-00073438; <https://hal.inria.fr/inria-00073438>

## Rapport de stage

- [80] [https://fr.wikipedia.org/wiki/Institut\\_national\\_de\\_recherche\\_en\\_informatique\\_et\\_en\\_automatique](https://fr.wikipedia.org/wiki/Institut_national_de_recherche_en_informatique_et_en_automatique)
- [81] <https://www.inria.fr/institut/inria-en-bref>
- [82] <https://www.inria.fr/centre/grenoble/presentation/acteur-des-sciences-numeriques/notre-centre>
- [83] <https://www.inria.fr/equipes/bipop>
- [84] <http://morpheo.inrialpes.fr/>
- [85] [https://fr.wikipedia.org/wiki/Python\\_\(langage\)](https://fr.wikipedia.org/wiki/Python_(langage))
- [86] <https://fr.wikipedia.org/wiki/ParaView>
- [87] <https://fr.wikipedia.org/wiki/Blender>
- [88] <https://fr.wikipedia.org/wiki/M%C3%A9taheuristique>
- [89] [https://fr.wikipedia.org/wiki/Algorithme\\_du\\_gradient](https://fr.wikipedia.org/wiki/Algorithme_du_gradient)
- [90] [https://fr.wikipedia.org/wiki/Th%C3%A9orie\\_des\\_fonctions\\_implicitlyes](https://fr.wikipedia.org/wiki/Th%C3%A9orie_des_fonctions_implicitlyes)
- [91] [https://en.wikipedia.org/wiki/Limited-memory\\_BFGS](https://en.wikipedia.org/wiki/Limited-memory_BFGS)
- [92] [https://fr.wikipedia.org/wiki/Assimilation\\_de\\_donn%C3%A9es](https://fr.wikipedia.org/wiki/Assimilation_de_donn%C3%A9es)
- [93] <https://matplotlib.org/>
- [94] <http://www.numpy.org/>
- [95] <https://www.scipy.org/>
- [96] [https://en.wikipedia.org/wiki/Kawabata\\_evaluation\\_system](https://en.wikipedia.org/wiki/Kawabata_evaluation_system)
- [97] <https://gforge.inria.fr/>

## Figures

- [98] <https://www.inria.fr/var/inria/storage/images/medias/inria/images-corps/logo-inria-institutionnel-couleur/410230-1-fre-FR/logo-inria-institutionnel-couleur.jpg>
- [99] [https://www.echosciences-grenoble.fr/uploads/article/image/attachment/1005162784/xl\\_inria\\_grenoble.jpg](https://www.echosciences-grenoble.fr/uploads/article/image/attachment/1005162784/xl_inria_grenoble.jpg)
- [100] <http://www.inrialpes.fr/bipop/Poster/posterBipopsmall.jpeg>
- [101] <http://kinovis.inrialpes.fr/wp-content/uploads/2013/08/Kinovis-room.jpg>
- [102] [https://raweb.inria.fr/rapportsactivite/RA2014/morpheo/IMG/sequence\\_goalkeeper.png](https://raweb.inria.fr/rapportsactivite/RA2014/morpheo/IMG/sequence_goalkeeper.png)
- [103] [http://www.fest3d.com/Online\\_Help/FEST3D/Project\\_imgs/FEST3D/Tutorial/Tutorial\\_EM\\_analysis/tfield4.PNG](http://www.fest3d.com/Online_Help/FEST3D/Project_imgs/FEST3D/Tutorial/Tutorial_EM_analysis/tfield4.PNG)
- [104] [https://pbs.twimg.com/profile\\_images/2366701444/fj4k3e5tac2q24c2droq\\_400x400.jpeg](https://pbs.twimg.com/profile_images/2366701444/fj4k3e5tac2q24c2droq_400x400.jpeg)
- [105] <https://gforgegroup.com/img/gforge-logo.png>
- [106] [https://wiki.inria.fr/wikis/ClustersSophia/images/thumb/4/4b/Nef\\_schema.png/700px-Nef\\_schema.png](https://wiki.inria.fr/wikis/ClustersSophia/images/thumb/4/4b/Nef_schema.png/700px-Nef_schema.png)
- [107] [https://www.sharcnet.ca/Software/Gambit/html/modeling\\_guide/mgimage/fig\\_m\\_face\\_modify01.gif](https://www.sharcnet.ca/Software/Gambit/html/modeling_guide/mgimage/fig_m_face_modify01.gif)
- [108] <http://ashaku.free.fr/images/evolution.png>
- [109] [https://fr.wikipedia.org/wiki/Marche\\_a1%C3%A9atoire](https://fr.wikipedia.org/wiki/Marche_a1%C3%A9atoire)

## Mots clefs

Vêtements et Textiles – Centre ou Laboratoire de recherche –  
Informatique – Recherche – Algorithmes – Mécanique –  
Multimédia – Modélisation, Simulation, Calcul

**JAFFALI Hamza**

**Rapport de stage ST50 – P2017**

## Résumé

Dans le cadre de la formation d'ingénieur en informatique de l'Université de Technologie de Belfort Montbéliard (UTBM), les étudiants doivent effectuer un stage de fin d'études d'une durée de 24 semaines en entreprise ou en laboratoire. L'objectif de ce stage est de permettre à l'étudiant, qui s'est initié au métier d'ingénieur lors de son premier stage, de mettre en application et de valider les connaissances acquises durant ses études, dans les conditions qui seront celles de ses activités et responsabilités à venir.

Mon travail effectué durant ce stage de recherche pendant 6 mois prit place au sein de l'Inria, plus particulièrement sur le site de Montbonnot rattaché au centre Grenoble Rhône-Alpes à Montbonnot-Saint-Martin. Le sujet conjointement proposé par Florence Bertails-Descoubes, Jean-Sébastien Franco et Stefanie Wuhrer consiste en la modélisation inverse et dynamique de vêtements. J'ai donc été intégré l'équipe de recherche BiPoP, et plus particulièrement le groupe Elan composé d'ingénieurs, de chercheurs et de doctorants, et dirigé par Florence Bertails-Descoubes.

Plus précisément, l'objectif de ce stage est, dans un premier temps, de formuler le problème d'inversion de vêtements dans le cas dynamique, notamment sous la forme d'un problème d'optimisation, puis dans un second temps de proposer et d'implémenter des solutions pour résoudre le problème précédemment posé.

## **INRIA Grenoble Rhône-Alpes**

**655 Avenue de l'Europe  
38330 Montbonnot-Saint-Martin  
<http://www.inria.fr/centre/grenoble>**