Problèmes d'Optimisation Non Linéaire avec Contraintes en Tomographie de Réflexion 3D

Frédéric Delbos

THÈSE

présentée à

L'université Pierre et Marie Curie (Paris VI)
École Doctorale de Sciences Mathématiques de Paris Centre

par

Frédéric Delbos

pour l'obtention du diplôme de

docteur de l'université de Paris VI
en mathématiques appliquées

Sujet de la thèse :

Problèmes d'Optimisation Non Linéaire avec Contraintes en Tomographie de Réflexion 3D

Soutenue le 17 novembre 2004

Devant la commission d'examen formée de :

Jacques Blum	*Examinateur*
Jean Charles Gilbert	*Directeur de thèse*
Roland Glowinski	*Directeur de thèse*
Frédéric Hecht	*Examinateur*
Gilles Lambaré	*Rapporteur*
Claude Lemaréchal	*Rapporteur*
Delphine Sinoquet	*Encadrant IFP*

à mes grands-parents Simone, Liliane, René et Gilbert,
à mes parents Michèle et Jean-Noël,
à ma soeur Valérie et à ma chérie Sophie.

Remerciements

Mes plus sincères remerciements s'adressent à

Delphine Sinoquet, ma responsable scientifique à l'Institut Français du Pétrole, qui m'a guidé tout au long de cette thèse et m'a initié à la géophysique et plus particulièrement à la tomographie de réflexion. En plus de ses nombreuses qualités professionnelles, j'ai beaucoup apprécié sa bonne humeur et son soutien dans les moments difficiles.

Jean-Charles Gilbert, mon directeur de thèse, qui m'a fait bénéficié de son incommensurable expérience et qui, grâce à ses qualités pédagogiques m'a donné l'envie et le goût d'explorer les mystères de l'optimisation numérique. Je lui suis très reconnaissant pour son investissement sans limites dans ce travail durant ces 3 années.

Roland Glowinski, mon directeur de thèse, et Roland Masson pour leurs idées et leurs critiques constructives qui ont permis d'orienter cette thèse. Les réunions passées en leur compagnie ont toujours été fructueuses et agréables.

Gilles Lambaré et Claude Lemaréchal qui m'ont fait le grand honneur d'être les rapporteurs de mon travail de thèse. Je les remercie beaucoup d'avoir consacré du temps à la lecture de mon manuscrit.

Jacques Blum et Frédéric Hecht qui ont accepté d'être membres de mon jury de thèse. C'est un grand honneur de les avoir dans mon jury.

Merci à mes collègues et amis stagiaires, thésards ou post-doctorants, Alexandre, Benoit, Carole, François, Frédéric (A et J), Hervé, Isabelle, Laure, Ludovic, Maud, Rémi, Thomas, Valérie, Vincent (A et K), Yannick et Yohan pour tous les moments agréables que nous avons passés ensemble.

Une partie de cette thèse s'est déroulée dans le cadre du consortium Kinematic Inversion Methods. Je remercie tous les membres de cette équipe pour les idées et les discussions qui ont pu enrichir ce travail : Karine Broto (ma rockeuse préférée), Florence Delprat-Jannaud, Bertrand Duquet, Andreas Ehinger, Bertrand Iooss, Jacques Jacobs, Fabrice Jurado (l'expert Jerry), Patrick Lailly et Timothée Perdrizet.

Je remercie également Vincent Richard de m'avoir accueilli dans la division géophysique.

Un grand merci à mon grizzly préféré Jean-Philippe pour les pauses syndicales café cigarettes et les longues discussions passionnées.

Merci à toutes les personnes de la division géophysique avec qui j'ai passé 3 merveilleuses années et qui contribuent à la bonne ambiance des "Primevères". Un merci tout

particulier aux secrétaires : Monique, Pierre et Sylvie, qui m'ont aidé à me sortir des labyrinthes administratifs. Je n'oublierai pas mon informaticien préféré : Jean-Marc, qui m'a aidé à rester "zen" devant ma Silicon Graphics.

Aline et Alain pour m'avoir ouvert les portes de l'IFP en me recommandant à Delphine et surtout merci beaucoup pour leur gentillesse.

Merci à Anne-Marie, Delphine et au bureau d'accueil de l'ENSPM pour m'avoir permis de trouver rapidement un logement en région parisienne.

Tous les amis parisiens et nîmois, Romu (vive l'A.S.M !) et Aurélie, Alainnng, Bamba, Mami et Valérie, la brigade, Mathieu et Karine, Bobo, Julien , Aymerit, David, Fanny, Sophie, Elvina, Viviane, Ludo et tous les autres grâce à qui ces 3 années dans la capitale furent heureuses, festives et passèrent très rapidement.

Résumé

La tomographie de réflexion de données sismiques permet la détermination d'un modèle de sous-sol à partir des temps de trajet des ondes sismiques se réfléchissant sur les interfaces géologiques. Le modèle solution de ce problème inverse est le modèle qui minimise une fonctionnelle moindres-carrés mesurant les écarts entre les temps de trajet mesurés et les temps de trajet calculés par tracé de rayons dans ce modèle. Le sujet de ma thèse est l'étude de la résolution numérique de ce problème non linéaire de minimisation. Les principales difficultés proviennent tout d'abord de la taille de l'espace des données (de l'ordre de 500000) et de l'espace des modèles (de l'ordre de 50000 inconnues), puis des problèmes de conditionnement liés à la structure du vecteur modèle (différents types d'inconnues interviennent : les vitesses dans les couches géologiques, les interfaces en transmission ou en réflexion). Enfin, la non linéarité de l'opérateur de modélisation des temps de trajet et la complexité de la propagation des ondes dans le sous-sol conduisent souvent à un problème de minimisation mal posé. Une régularisation sur la courbure du modèle permet d'avoir un problème de minimisation mathématique bien posé. De plus, l'introduction d'informations géologiques a priori par résolution sous contraintes du problème inverse permet de réduire les incertitudes sur le modèle solution.

Classiquement, une méthode de Gauss-Newton couplée avec une méthode de recherche linéaire était utilisée pour résoudre les problèmes d'optimisation sans contrainte en tomographie de réflexion sismique (Chauvier, L., Masson, R., and Sinoquet, D., 2000). A chaque itération de Gauss-Newton, une solution (approchée) d'un modèle quadratique de la fonction coût non linéaire est calculée en utilisant l'algorithme du gradient conjugué. Cette méthode permet en général de résoudre les problèmes de minimisation traités en tomographie. Cependant, nous avons remarqué pour certains cas que la méthode de recherche linéaire était non seulement incapable de trouver une solution, mais encore de faire décroître de manière significative la fonction coût. Ces difficultés sont principalement dues à une matrice Hessienne mal conditionnée. Afin de pallier à ces difficultés, la première partie de ma thèse a consisté à remplacer la recherche linéaire par une méthode de région de confiance, et à développer un algorithme de gradient conjugué tronqué pour minimiser le modèle quadratique de la fonction coût non linéaire. Les différents paramètres de la méthode des régions de confiance (par exemple le rayon de la région de confiance) sont automatiquement réglés par le solveur. Cette méthode donne des résultats plus stables et plus précis. Elle permet de mieux résoudre les exemples complexes où le Hessien est très mal conditionné.

La deuxième partie de ma thèse consiste à étudier la résolution des problèmes d'opti-

misation avec contraintes en tomographie de réflexion sismique. Les contraintes à ajouter dans la résolution de nos problèmes d'optimisation peuvent être de types très différents. Elles peuvent être non linéaires (nous pourrions par exemple contraindre les points d'impact des rayons sur une interface à être localisés dans une région particulière), mais dans une première approche nous préférons nous limiter à des contraintes linéaires. Même si la linéarité apporte des simplifications, nos problèmes d'optimisation restent encore très difficiles à résoudre notamment à cause du nombre élevé de contraintes (de l'ordre de 10000) et de leurs types différents.

Actuellement, une méthode souvent utilisée pour résoudre les problèmes d'optimisation non linéaire avec contraintes est l'approche par points intérieurs. Cependant, nous soupçonnons que cette méthode demande significativement plus de résolutions du problème direct, le tracé de rayon, ce qui est un inconvénient lorsque celui-ci coûte cher en temps CPU. Ainsi, nous avons préféré une méthode de Programmation Quadratique Successive (PQS) car nous supposions que, comme méthode (Gauss-) newtonnienne, elle ne demanderait pas plus d'itérations (donc d'évaluation de fonctions), que l'algorithme de Gauss-Newton dans le cas sans contrainte. Cette précision s'est avérée exacte. Glowinski et Tran, en 1993, ont développé et testé une méthode PQS dans laquelle le problème quadratique tangent (PQT) est résolu via une méthode de Lagrangien Augmenté. Nous nous sommes inspirés de cette première approche en améliorant la résolution des PQT. Le code QPAL que nous avons dévellopé pendant cette thèse a montré son efficacité pour résoudre des problèmes quadratiques convexes de grandes tailles : il utilise d'une part la méthode classique des multiplicateurs de Hesteness et Powell pour minimiser la fonction duale et d'autre part l'algorithme GP-AC-GC pour résoudre les problèmes de Lagrange. Nous avons montré sur de nombreux cas tests ou réels que cette méthode donne des résultats convaincants :

- le nombre d'évaluations de la fonction coût reste du même ordre de grandeur que celui obtenu par le solveur gauss-newtonnien sans contrainte,
- cette méthode est capable de résoudre des problèmes de grande taille,
- l'introduction d'informations a priori par l'inversion avec contraintes permet de réduire les incertitudes sur le modèle solution.

Abstract

Seismic reflection tomography allows the determination of a subsurface model from the traveltimes of seismic waves reflecting on geologic interfaces. The solution model of this inverse problem is a model that minimizes a least-squares functional measuring the mismatch between observed traveltimes and traveltimes calculated by raytracing in this model. My thesis subject consists in studying the resolution of this non linear minimization problem. The main difficulties come first from the size of the data space (around 50000 traveltimes) and of the model space (around 50000 unknowns), secondly from the conditioning problems due to the structure of the model vector (containing different sorts of unknowns : velocities in the geologic layers, transmission interfaces or reflecting interfaces). Thirdly, the non linearity of the modelling operator of traveltimes and the complexity of wave propagation in subsurface often imply an ill-posed minimization problem (Delprat-Jannaud, F., and Lailly, P., 1993). To ensure its well-posedness, curvature regularization is used. In addition, introduction of geological a priori information thanks to constrained optimization reduces uncertainties on the model solution.

A classical line search Gauss-Newton method was used to solve unconstrained optimization problems in seismic reflection tomography (Chauvier, L., Masson, R., and Sinoquet, D., 2000). At each Gauss-Newton step, the (approximate) solution of a quadratic model of the objective function is computed using a conjugate gradient algorithm. This method generally allows to solve the minimization problem, but in some cases we noticed that the line search method is neither able to find a solution nor to decrease significantly the objective function. This difficulty is likely to be due to a too ill-conditioned Hessian matrix. To overcome this, we have proposed and implemented an original trust-region Gauss-Newton method coupled with a truncated conjugate gradient algorithm (Delbos, F., Sinoquet, D., Gilbert, J. C., and Masson, R., 2001) where the trust-region parameters (for example the trust-region radius) are automatically tuned by the solver. This method provides more stable and accurate results. In fact, it better solves intricate examples where the Hessian matrix is strongly ill-conditioned.

The second part of my thesis consists in studying constrained optimization problems in seismic reflection tomography. The constraints we want to introduce in the optimization problem are of multiple types. They could be non linear (for example we could constrain the impact points of the rays on one interface to be located in a particular area) but in a first approach we prefer to limit ourselves with linear constraints. Even if the linearity brings simplifications, our optimization problems are still very difficult to solve because of the huge number of constraints (around 10000) and their different types.

A widely used method to solve constrained non linear optimization problem is an interior point approach. However, we suspect this method to require significantly more resolutions of the forward problem, a raytracing solver, which is a drawback when this one is expensive in CPU time. Hence, we preferred to choose a Sequential Quadratic Programming (SQP) method because we had supposed that, as (Gauss-) newtonnian method, it does not require more iterations than the Gauss-Newton algorithm in the unconstrained case. This assumption proved to be exact. In the context of seismic tomography problems, a version of the augmented Lagrangian (AL) algorithm has been proposed by Glowinski and Tran (1993) to solve the tangent quadratic problem (TQP) of the SQP method. The solver QPAL we have developped in this thesis takes inspiration from that work and goes further by improving the efficiency of its augmented Lagrangian QP solver : it uses the classical multiplier method of Hesteness (1969) and Powell (1969) to minimize the dual fonction and it uses a GP-AC-GC algorithm to solve the Lagrange problem of the AL method. We have shown on a various concrete inversion that our nonlinear optimization method gives nice results :

- the number of Gauss-Newton iterations has the same order of magnitude than the number of Gauss-Newton iterations necessary in the unconstrained case (around 10 iterations),
- our nonlinear optimization method is efficient even for a large number of constraints,
- the introduction of geological a priori information thanks to constrained optimization reduces uncertainties on the model solution.

Table des matières

Introduction Générale

Introduction

Objectif : la prospection pétrolière

Le début de la prospection pétrolière est marqué par la date du 27 août 1859 ou le premier puits de pétrole (d'une profondeur de 23 mètres) a été foré par le Colonel Drake à Titusville, aux États-Unis. A cette époque la prospection se faisait "à l'instinct". Un forage presque au hasard de quelques dizaines de mètres de profondeur suffisait souvent pour trouver un gisement. Cette méthode appelée *wildcat*, a longtemps été privilégiée pour le coût limité des forages peu profonds et les moyens financiers dont disposaient les compagnies pétrolières. De nos jours, une telle méthode n'est plus envisageable car la plupart des réserves de pétrole non découvertes sont situées dans des zones difficiles d'accès (par exemple en mer profonde - Golfe de Guinée) et/ou à une profondeur importante. Comme le coût d'un forage dans ces zones est prohibitif, il faut utiliser les techniques d'exploration moins chères de la géologie et de la géophysique. Elles permettent d'identifier les gisements de manière plus sure que par le passé : seulement 1 forage sur 7 donne des indices d'hydrocarbure dans une zone peu connue alors que 1 sur 4 en donne en zone mature. On distingue ainsi trois types différents de techniques dédiées à l'exploration pétrolière :

1. la prospection géologique (sédimentologie, stratigraphie, géologie structurale, géochimie organique)
2. la prospection géophysique (sismique)
3. le forage d'exploration (forage rotary, mud logging, diagraphies)

Les deux premières techniques assurent que toutes les conditions nécessaires à la formation d'hydrocarbure et à l'existence d'un gisement sont bien réunies sur la zone explorée. La première des deux, la prospection géologique, s'est développée avec l'essor de l'industrie pétrolière. En recherchant les caractéristiques communes d'une série de puits producteurs (par exemple des puits se trouvant dans une vallée fluviale), les prospecteurs se sont peu à peu transformés en géologue. Les anticlinaux ont ainsi rapidement constitués les premières cibles de prospection. La prospection géophysique est capable de révéler des informations à distance (en dehors du voisinage de puits). La géophysique, appelée aussi physique du globe, est la science qui consiste à étudier la Terre par les méthodes de la physique. Les méthodes géophysiques effectuent des mesures de grandeurs physiques qui varient en fonction de la nature ou de la disposition des roches contenues dans le sous-sol.

On peut compter autant de méthodes géophysiques que de grandeurs physiques à mesurer dans le sous-sol (magnétisme, résistivité, potentiel électrique des roches, intensité de la pesanteur etc...). La représentation dans l'espace des variations de ces paramètres permet de visualiser les structures du sous-sol. Les caractéristiques de ces techniques sont variables : capacité à détecter ce que l'on cherche (spécificité), capacité à lire en profondeur (pénétration) et capacité à détecter des objets de petite taille (résolution). A l'heure actuelle, la sismique de surface est la technique de prospection géophysique la plus utilisée par les prospecteurs : elle permet d'obtenir relativement simplement des informations structurales et/ou quantitatives sur les couches géologiques du sous-sol jusqu'à plusieurs kilomètres de profondeur. Une fois que les zones les plus favorables ont été localisées, la prospection par l'utilisation de forages d'exploration peut alors débuter. Cette phase est la plus coûteuse d'une campagne de prospection : elle représente plus de 60 % de l'investissement total alors que les études géologiques et géophysiques représentent respectivement seulement 5 % et 15 % de l'investissement. Dans les trois sous-sections suivantes nous présentons les 3 campagnes de prospection les plus utilisées par les pétroliers : prospection géologique, campagne de sismique réflexion et prospection par forage ou sondage.

Campagne de prospection géologique

De part son coût peu élevé, la campagne géologique est la première technique d'exploration utilisée lorsque l'on recherche de nouveaux gisements. Elle a pour objectif de reconnaître les terrains : la présence d'affleurements en surface permet de reconstituer en partie l'architecture des couches ainsi que les différents faciès lithologiques que l'on peut espérer rencontrer dans le sous-sol. L'utilisation des techniques d'aéromagnétisme, de gravimétrie et de géochimie permet au géologue de recueillir une multitude d'indices sur la nature du sous-sol. A partir de tous ces indices, le géologue pétrolier recherche en particulier les composantes nécessaires à la formation d'un gisement : la roche mère, la roche magasin et le piège. Une fois ces trois composantes réunies le géologue peut donner une estimation de la position du gisement.

Campagne de sismique

La campagne sismique constitue la principale activité de la géophysique. Son coût, qui s'échelonne entre 1 et 6 M$ selon qu'il s'agit d'une campagne à terre ou en mer et selon la durée de cette campagne, est beaucoup plus élevé que celui d'une campagne géologique.

La sismique réflexion étudie la réponse du sous-sol à une excitation de celui-ci par des sources explosives ou vibrantes placées en surface (voir figure 1). Des charges explosives placées dans le sol ou des détentes d'air comprimé ou d'eau (en sismique marine) crée une onde qui se propage dans le sol de proche en proche et subit des réflexions et des transmissions suivant les lois de Snell-Descartes. Par l'intermédiaire, en surface, d'un réseau de capteurs (ou récepteurs) on enregistre en fonction du temps l'amplitude des ondes réfléchies (déplacement du sous-sol) par les discontinuités d'impédance acoustique. Ces discontinuités correspondent souvent à des contacts entre des formations géologiques ap-

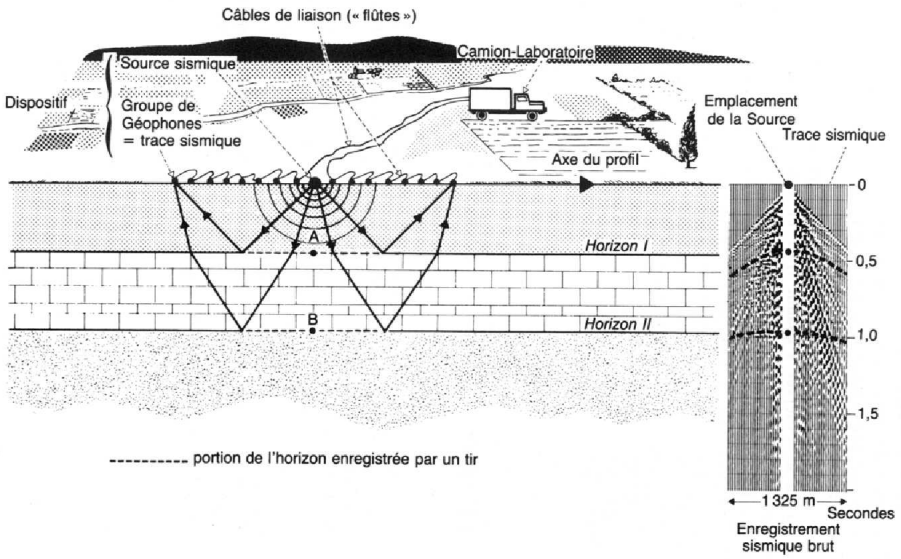

Fig. 1 Illustration d'une campagne de sismique (figure 1 de [132]) .

pelés interfaces géologiques ou horizons. Les résultats sont des courbes appelées traces sismiques. Chaque trace sismique correspond à une source et un récepteur donnés. En regroupant toutes les traces sismiques associées au même point source, on obtient une section sismique du terrain sous-jacent. On peut remarquer sur la figure 1 qu'une section sismique couvre une partie restreinte des horizons. Ainsi, il faut déplacer le dispositif d'acquisition et réaliser de nombreux tirs de manière à illuminer au mieux les interfaces géologiques de la zone étudiée. A la fin d'une campagne d'acquisition de données sismiques le prospecteur dispose de nombreuses sections sismiques associées à des points de tir. L'étape suivante consiste à traiter ces données pour obtenir finalement une image sismique du sous-sol. Cette image permet au prospecteur de mettre en évidence les éléments structuraux majeurs du sous-sol et donc de trouver les pièges d'hydrocarbure potentiels. Les informations fournies par cette image confrontées aux informations fournies par le géologue vont permettre de déterminer l'emplacement de la zone pétrolifère et des futurs puits producteurs.

Une campagne sismique se décompose donc en trois phases successives : acquisition des données, traitement des données et interprétation des données. Dans le tableau 1 nous avons retranscrit le coût détaillé de chacune de ces phases.

Campagne de prospection par forage ou sondage

A cause de son coût élevé, la campagne de prospection par forage ou sondage est en général utilisée seulement lorsque les autres méthodes de prospection (campagnes

		TERRE	*MER*
Acquisition	SISMIQUE 2D	3 000 à 9 000 $/km tracé 100 à 180 km/mois-équipe	400 $/km 1 400 à 1 800 km/mois-équipe
	SISMIQUE 3D	15 000 à 50 000 $/km² 30 à 60 km²/mois	5 000$/km² 180 à 360 km²/mois
Traitement	SISMIQUE 2D	100 $/km	
	SISMIQUE 3D	500 $/km	
Exploitation		100 000 $ à 1 000 000 $/campagne dure de quelques mois à plusieurs années	

Tableau 1 Coûts détaillés de la sismique

géologiques et sismiques) ont affirmé avec une grande probabilité la présence d'un gisement d'hydrocarbure. Ce coût varie énormément selon la zone géographique, le type de puits, la durée du forage et la disponibilité des appareils de forage et supports en mer. Par exemple, un puis à terre aux États-Unis peut coûter moins d'un million de dollars alors qu'un puits en Mer du Nord coûtera environ 12 M$. Dans des conditions extrêmes, le coût de forage dépassera les 40 M$. Malgré son coût élevé la prospection par forage reste, dans bien des cas, la seule technique capable d'affirmer ou d'infirmer avec certitude la présence d'un gisement d'hydrocarbure. Le forage de prospection (forage "rotary"), consiste à creuser un trou de faible diamètre, d'une profondeur qui peut atteindre plusieurs milliers de mètres, en faisant tourner un outil contre les parois rocheuses. Lors de ce forage, une analyse des terrains traversés est généralement effectuée. Cette analyse s'appuie sur différentes techniques :

- l'examen des débris de roche (boue du forage - mud logging),
- le prélèvement d'échantillons (carottage),
- la mesure de nombreux paramètres physiques à l'aide d'une sonde électronique descendue dans le puits (diagraphies).

Imagerie sismique

Les sections sismiques recueillies lors d'une campage sismique contiennent de nombreux événements qui ne correspondent pas seulement aux ondes réflechies sur les interfaces géologiques. On appelle signal l'ensemble des données obtenues lors d'une campagne d'acquisition sismique. Le signal est donc la somme du signal propre aux ondes réflechies (signal sismique) avec du bruit. Parmi les différents bruits polluant le signal sismique, certains bruits sont reconnaissables sur les sections sismiques car ils correspondent de part leurs caractéristiques (temps de trajet et amplitude) à un certain type d'onde : on les appelle bruits corrélés. Les bruits corrélés les plus connus correpondent à des ondes directes, des ondes réfractées, des ondes diffractées, des réflexions multiples et des ondes de surface. D'autre bruits sont complètement aléatoires : on ne peut pas les caractériser sur les sections sismiques en terme de temps de trajet ou d'amplitude. La plupart de ces bruits sont d'origine naturelle et souvent liés aux conditions météorologiques

lors de l'acquisition des données (pluie, vent, houle etc ...). La première étape du traitement des données sismiques consiste donc à traiter les sections sismiques de manière à en extraire le signal sismique. Cette étape est principalement réalisée à l'aide de méthode de filtrage (voir les travaux précurseurs de [115]).

Une fois que le signal est "nettoyé" du bruit il faut alors transformer toutes les sections sismiques d'une ligne d'acquisition en une seule coupe en profondeur du sous-sol. Cette opération qui s'appelle la convertion temps-profondeur nécessite la modélisation d'une loi physique liant le domaine du temps au domaine de la profondeur via un modèle de vitesse du sous-sol. La combinaison d'une méthode permettant de déterminer le modèle de vitesse avec une méthode de conversion temps-profondeur s'appelle l'imagerie sismique. Les techniques les plus utilisées en imagerie sismique sont la tomographie de réflexion pour déterminer le modèle de vitesse (voir [15], [54], [48] et [14]) et la migration avant ou après sommation pour réaliser l'opération de conversion temps-profondeur (voir [31] et [160]). La tomographie de réflexion s'attache à retrouver la structure des différentes couches géologiques du sous-sol et les vitesses de propagation des ondes dans ces couches. Elle n'utilise pas les informations contenues dans l'amplitude des réflexions. Or cette information peut permettre de retrouver certaines propriétés physiques du sous-sol telles que la densité des roches ou l'impédance. Une autre technique est alors utilisée pour exploiter cette information : l'inversion de formes d'onde. Elle recherche le modèle de terre qui explique au mieux les données observées, données qui sont constituées non seulement des temps de trajet mais aussi des amplitudes des ondes réfléchies. Cette technique peut aussi bien être utilisée après sommation ([9], [20]) qu'avant sommation ([102], [19], [104]).

Il existe principalement deux types de problèmes inverses en sismique :

1. la tomographie de réflexion (voir chapitre 2)
2. l'inversion de formes d'onde (voir [121])

L'objectif de ces deux problèmes inverses est le même : retrouver les paramètres du modèle de Terre dont la réponse s'ajuste au mieux sur les données sismique observées en surface (voir [150]). Dans la tomographie de réflexion on s'intéresse seulement à l'information cinématique contenu dans les données (temps de trajet des ondes réfléchies) alors que dans l'inversion de formes d'onde on exploite aussi l'information dynamique des données (amplitude des ondes). Dans l'inversion de formes d'onde tous les types d'onde (arrivées directes, arrivées multiples, ondes converties etc...) sont pris en compte. Le caractère fortement non linéaire de l'équation caractéristique par forme d'onde, ainsi que le nombre très important de données prises en compte dans l'inversion rendent cette méthode difficilement applicable. En pratique, il a été constaté que cette méthode connaît non seulement des problèmes de convergence mais qu'elle est aussi très gourmande en temps de calcul. C'est pourquoi on lui préfère souvent la tomographie de réflexion, beaucoup moins coûteuse en temps de calcul. Cependant, il est à noter que la tomographie de réflexion nécessite une étape d'interprétation des données, appellée le "pointé". Cette étape qui consiste à identifier dans les enregistements sismiques les évènements correspondant à des réflexions primaires est délicate : elle peut conduire à des données d'inver-

sion érronées et/ou incomplètes (voir [21]).

Description des parties de la thèse

Le travail de thèse que j'ai realisé pendant trois ans s'est effectué sur trois axes de recherche différents et complémentaires :

1. optimisation sans contrainte,
2. optimisation avec contraintes,
3. applications sur données réelles.

Les trois parties de ce mémoire de thèse reflètent ces trois axes de recherche.

La première partie traite de la résolution des problèmes d'optimisation sans contrainte en tomographie de réflexion. Elle donne les les éléments essentiels pour pouvoir aborder les parties suivantes. Dans le dernier chapitre de cette partie (voir le chapitre 3), l'article "Trust-region Gauss-Newton method for reflection tomography" montre l'aboutissement du travail de ma première année de thèse. Cette article présente les performances d'une méthode de Gauss-Newton globalisée par régions de confiance en tomographie de réflexion.

La deuxième partie s'interesse à la résolution du problème inverse de la tomographie de réflexion avec contraintes. Nous présentons dans cette partie une méthode de programmation quadratique successive bien adaptée aux caractéristiques de notre application en tomographie de réflexion. Nous nous attacherons particulièrement à développer en détail le fonctionnement de cette méthode et à motiver son utilisation.

Enfin, la troisième et dernière partie examine l'application sur données réelles de la méthode d'optimisation avec contraintes développée dans la deuxième partie. La lecture de l'article "Constrained optimization in seismic reflection tomography : an SQP augmented Lagrangian approach" montrera l'efficacité de notre méthode d'optimisation sur 2 jeux de données réelles et motivera l'utilisation des contraintes en tomographie de réflexion.

Par soucis de clarté, à ce niveau du document nous ne souhaitons pas décrire les chapitres de chacune des parties décrites ci-dessus. Cependant, nous noterons que chaque partie possède une introduction qui présente en détails le contenu des chapitres qu'elle va aborder.

Première partie

Résolution du problème inverse de tomographie de réflexion : optimisation sans contrainte

Dans cette partie nous étudions la résolution du problème inverse de la tomographie de réflexion sans contrainte. En optimisation, ce problème de tomographie peut être vu comme un problème de moindres-carrés non-linéaire. Une méthode de Gauss-Newton globalisé par recherche linéaire est classiquement utilisée dans le logiciel *jerry* de tomographie de réflexion. Elle décompose le problème de moindres-carrés non-linéaire en une suite de problèmes de moindres-carrés linéaires. Puis, à chaque itération de Gauss-Newton, le problème moindres-carrés linéaire est résolu par gradient-conjugué. Cette partie est divisée en trois chapitres.

Dans le premier chapitre nous donnons les résultats et concepts théoriques nécessaires pour la compréhension des chapitres et parties suivants. Nous abordons plus particulièrement les notions de :

- problèmes inverses,
- conditions d'optimalité des problèmes d'optimisation sans contrainte,
- problèmes de moindres-carrés non-linéaires.

Nous verrons que ces notions et les résultats qui y sont développés sont régulièrement utilisés dans ce mémoire.

Dans le deuxième chapitre nous présenterons les caractéristiques du problème de la tomographie de réflexion, caractéristiques liées au choix de la discrétisation du modèle par des fonctions B-spline cubiques, caractéristiques du problème direct et les caractéristiques du problème inverse. A l'issue de cette lecture, nous comprendrons aussi pourquoi le problème inverse de la tomographie de réflexion se formule comme un problème de moindres-carrés non-linéaires. Notons que ce chapitre est important pour la suite car la méthode d'optimisation avec contraintes qui sera développée dans la partie II tire parti des caractéristiques spécifiques des problèmes de tomographie de réflexion.

Enfin, dans le troisième et dernier chapitre de cette partie nous développerons une méthode de Gauss-Newton globalisée par régions de confiance. Nous observerons, sur certains exemples de tomographie de réflexion, que cette méthode est plus efficace que la méthode classique de Gauss-Newton.

Chapitre 1

Résultats généraux

1.1 Les problèmes inverses

1.1.1 Introduction aux problèmes inverses

Depuis les années 70, suite aux travaux fondateurs de [148] sur la résolution des problèmes mal posés, les problèmes inverses constituent un axe de recherche très actif du domaine des mathématiques appliquées. Les progrès spectaculaires réalisés lors des dernières décénnies dans les domaines de l'informatique et du calcul scientifique ont vu l'apparition des problèmes inverses dans de nombreux secteurs d'activité du domaine industriel :

- Les sciences de la terre, la météorologie et l'océanographie,
- L'industrie aérospatiale,
- L'imagerie médicale,
- L'industrie électronucléaire,
- Le génie civil ...

Une grande partie des problèmes inverses posés dans ces différents domaines industriels concernent la reconstruction de caractéristiques physiques d'une région inaccessible à l'observation directe. Par exemple, dans le cas particulier de la géophysique pétrolière, on recherche les propriétes physiques du sous-sol à partir de données (sismiques par exemple) le plus souvent seulement accessibles à la surface (voir l'introduction générale ainsi que [39], [114], [145] et [146]).

D'une manière générale on peut symboliser la résolution d'un problème de mécanique ou de thermique à l'aide de la notion de système : on recherche la réponse d (potentiels, déplacement, température, signal sismique etc...) à des sollicitations X (excitations, sources explosives ou vibrantes, forces etc ...) sur un système (S) (domaine de sous-sol étudié, structure mécanique analysée etc...) qui est caractérisé par un ensemble de paramètres m.

Problème direct. Le *problème direct* consiste à calculer la réponse d à partir de la donnée des sollicitations X et des paramètres m (voir figure 1.1) :

$$\text{Trouver } d \text{ tel que } G(X, d, m) = 0, \quad \text{avec } (X, m) \text{ donnés}, \tag{1.1}$$

où G représente un opérateur de modélisation qui relie (implicitement) d, m et X. Cet opérateur représente un ensemble d'équations (par exemple equations aux dérivées partielles - EDP) construites à partir des lois de la physique.

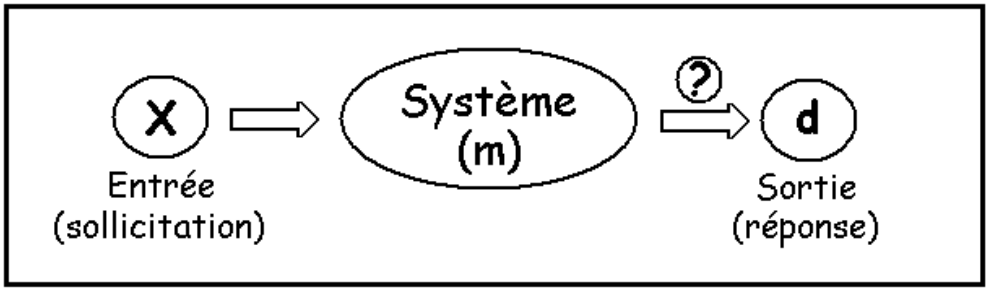

Fig. 1.1 Problème direct

Problème inverse. Dans le problème inverse, on ne connait pas tous les paramètres m caractérisant le système (S). Par contre, on connait (partiellement) la sortie d. Le but est alors de retrouver les paramètres inconnus du système en utilisant non seulement des informations sur l'entrée X mais aussi sur la sortie d (voir figure 1.2) :

$$\text{Trouver } m \text{ tel que } G(X, d, m) = 0, \quad \text{avec } (X, d) \text{ donnés}. \tag{1.2}$$

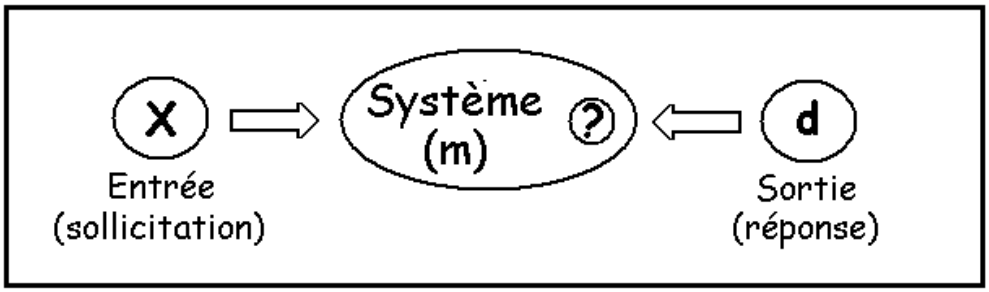

Fig. 1.2 Problème inverse

Définition 1.1.1 *On dit qu'un problème est bien posé au sens de Hadamard lorsque les trois conditions suivantes sont satisfaites :*

(i) existence d'une solution,

(ii) unicité de la solution,

(iii) stabilité de la réponse par rapport aux petites erreurs (de données, de discrétisation et etc...).

Si l'une au moins de ces trois conditions n'est pas satisfaite on dit alors que le problème est mal posé.

On appelle problème inverse les problèmes (1.2) qui sont mal posés. Plus généralement, le terme problème inverse désigne aussi les problèmes où l'on va "a contrario" des équations de la physique.

1.1.2 Méthodes d'investigation d'un problème inverse

De part leur aspect mal posé les problèmes inverses sont très souvent sensibles aux incertitudes. Les incertitudes sont nombreuses et d'origines variées :

Remarques 1.1.2

- *la nature expérimentale des données peut conduire à une solution du problème inverse très différente de la solution exacte voire même dans certains cas à la non existence d'une solution lorsque les données d sont incompatibles avec l'entrée X. On note que les données sont souvent*
  - ⋆ *imprécises ou bruitées (existence d'erreurs de mesure),*
  - ⋆ *incomplètes (caractère discret des mesures)*
  - ⋆ *redondantes (données collectées en points du modèle qui sont "proches")*
  - ⋆ *altérées par des algorithmes d'interpolation, de filtrage, etc...*
- *le modèle physique est une représentation simplifiée des lois de la physique, il repose sur des hypothèses simplificatrices et sur la mesure expérimentale de certains paramètres du modèle (constantes physiques).*

La présence de ces nombreuses incertitudes implique un changement d'approche sur la recherche des solutions d'un problème inverse. En effet, si l'on trouve une solution $\tilde{m}$ "au sens strict" du problème (1.2) alors cette solution n'est pas suffisante car elle explique à la fois le signal et les erreurs contenues dans les données. C'est à dire $\tilde{m}$ vérifie l'équation

$$G(X, d_{exact} + \delta_d, \tilde{m}) = 0,$$

où $d = d_{exact} + \delta_d$, d_{exact} représente les données exactes (non bruitées), et δ_d le bruit. En fait, tous les paramètres m expliquant la réponse d "aux incertitudes près" sont des solutions possibles au problème inverse. Selon [120], il existe trois grandes classes de méthode de résolution des problèmes inverses qui en plus du modèle physique à inverser tiennent compte des *informations a priori* sur le modèle :

1. Les approches relevant de l'analyse fonctionnelle consistent à transformer un problème mal posé en un problème bien posé en modifiant les espaces décrivant les variables et leur topologie.

2. Les méthodes de *régularisation* (voir [148]) consistent à rechercher la solution d'un problème bien posé (ou problème régularisé) qui est "proche" du problème initial. La solution du problème régularisé doit dépendre continûment des données et doit être suffisament proche de la solution exacte (les données calculées doivent être proches des données observées).
3. L'inversion *stochastique* ou *bayésienne* (voir [146] et [114]) considère que toutes les variables du problème sont aléatoires afin de pouvoir représenter les incertitudes. La solution du problème inverse est alors la fonction densité de probabilité associée à l'inconnue m et à la mesure d.

1.1.3 Régularisation d'un problème inverse

La régularisation d'un problème inverse consiste à reformuler le problème mal posé en poursuivant les objectifs suivants :

1. injecter des informations à priori qualitatives sur le modèle (informations qui proviennent de la physique du problème).
2. améliorer la stabilité de la solution par rapport aux erreurs de mesure en rajoutant ces informations supplémentaires.

On suppose disposer de l'opérateur de modélisation $G : \mathcal{M} \mapsto \mathcal{D}$, où $\mathcal{M}$ est l'ensemble des modèles et $\mathcal{D}$ l'ensemble des données. Le problème inverse (1.2) se reformule alors comme :

$$\text{Trouver } m \in \mathcal{M} \text{ tel que } G(m) = d_{obs}, \qquad \text{avec } d_{obs} \in \mathcal{D} \text{ donné,} \tag{1.3}$$

où le second membre d_{obs} correspond aux données observées qui sont entachées d'erreurs. Connaissant la valeur maximale de l'erreur δ des données observées par rapport aux données exactes (notées d_{exact}), on peut écrire l'inégalité suivante :

$$||d_{obs} - d_{exact}||_{\mathcal{D}} \leq \delta.$$

Il découle alors naturellement que l'on va chercher une solution de (1.3) parmi les modèles m vérifiant :

$$||G(m) - d_{obs}||_{\mathcal{D}} \leq \delta. \tag{1.4}$$

En raison du caractère mal posé des problèmes inverses il existe généralement un "grand" ensemble de modèles vérifiant (1.4). On ne peut donc pas se contenter de résoudre (1.4), il faut aussi pouvoir sélectionner un modèle m parmi toutes les solutions possibles et ce modèle doit dépendre continûment de δ (pour δ "suffisamment petit").

La régularisation consiste alors à introduire une *fonctionnelle stabilisatrice* $\Omega(m)$, dont la minimisation sous la condition (1.4) permet de trouver pour un δ donné une unique solution de (1.3) :

$$\begin{cases} \min\limits_{m} \Omega(m) \\ ||G(m) - d_{obs}||_{\mathcal{D}} \leq \delta. \end{cases} \tag{1.5}$$

La solution m_δ de ce problème est généralement appelée la *solution régularisée* de (1.3). Cette solution dépend du choix de la fonction Ω qui est l'outil essentiel permettant de restreindre le champ des solutions approchées de (1.4). En pratique, il existe une infinité de possibilités pour définir Ω. La plus simple et la plus intuitive consiste à rechercher le modèle m le plus proche d'un modèle de référence m_0 obtenu à partir de considérations physiques :

$$\Omega(m) = ||m - m_0||^2_{\mathcal{M}}, \qquad m_0 \in \mathcal{M} \text{ fixé.}$$

Une fois la fonction Ω définie on recherche la solution régularisée m_δ. Cependant, en général, on ne connaît pas la valeur de l'erreur δ sur les données. Ainsi, il est courant de remplacer la résolution du problème (1.5) par le problème suivant :

$$\min_m \left(H(m, \alpha) := ||G(m) - d_{obs}||_{\mathcal{D}} + \alpha\Omega(m) \right), \tag{1.6}$$

où α est le paramètre de régularisation et on note m_α l'unique solution du problème (1.6). En optimisation, le problème (1.6) peut-être vu comme une "sorte" de pénalisation de (1.5). La proposition suivante, reprise de la proposition 12.3 de [68], montre comment varie les différents termes de $H(m_\alpha, \alpha)$ lorsque α varie.

Proposition 1.1.3 *On note m_α la solution du problème (1.6). Alors, lorsque $\alpha > 0$ décroît, $||G(m_\alpha) - d_{obs}||_{\mathcal{D}}$ décroît, $\Omega(m_\alpha)$ croît et $H(m_\alpha, \alpha)$ décroît.*

Ainsi, lorsqu'on a pas de connaissance a priori sur la valeur de l'erreur δ, la resolution du problème (1.6) par la formulation (1.5) permet de contrôler l'écart $||G(m_\alpha) - d_{obs}||_{\mathcal{D}}$ en jouant sur le paramètre de régularisation α. L'une des grandes difficultés de la régularisation consiste à choisir le paramètre α. L'efficacité de l'algorithme d'inversion est en effet très sensible au choix de ce paramètre. De nombreuses techniques ont été mises au point pour aider à ce choix. Dans la section suivante nous décrivons les principales méthodes permettant de choisir une bonne valeur du paramètre de régularisation.

1.1.4 Méthodes de calcul du paramètre de régularisation

Gardons à l'esprit le double objectif de la régularisation, c'est à dire de minimiser le résidu $||G(m) - d_{obs}||_{\mathcal{D}}$ tout en minimisant aussi les effets des perturbations sur les données. A partir de cette observation on peut identifier deux types d'erreurs dans une solution régularisée :

1. *l'erreur de régularisation* qui se produit lors de la minimisation des effets des perturbations des données sur la solution. Cette erreur est généralement la source d'une perte d'informations pertinentes.
2. *l'erreur de perturbation* qui est la différence entre la solution obtenue avec les données exactes et celle obtenue avec les données bruitées.

Par le réglage du paramètre de régularisation, les techniques de régularisation tentent de trouver le bon compromis entre ces deux types d'erreurs. Il y a deux classes de critère permettant de choisir le paramètre de régularisation. La première classe se base sur la

connaissance du niveau de bruit présent dans les données. Le seul critère de cette classe, appelé critère de divergence, est due à Morozov (voir [119]). L'idée basique de ce critère est que l'on ne peut pas s'attendre à résoudre le problème inverse avec plus de précision que la précision présente dans les données. La deuxième classe tente de donner une estimation du paramètre en utilisant plusieurs solutions approchées du problème. Parmi les critères les plus connus de cette classe, on cite le *critère de validation croisée* et la *méthode de la courbe en L*. Pour plus de détails sur ce sujet on pourra consulter les travaux de [90], [149] et [53].

Critère de validation croisée

Le critère de validation croisée est présenté dans [119] puis discuté dans [79]. Ce critère consiste à choisir le paramètre de régularisation α qui minimise la fonction

$$C(\alpha) = \sum_{k=1}^{n} \left(G_k(m_\alpha^k) - d_{obs,k} \right)^2 ,$$

où k représente l'indice qui numérote les données expérimentales et m_α^k est la solution de l'inversion régularisée avec paramètre α sans tenir compte de la k-ième mesure. En fait ce critère cherche à trouver le paramètre de régularisation α de telle manière que la solution m_α respecte au mieux les mesures qui ne sont pas exploitées pour l'inversion. De nombreuses difficultés sont associées à ce critère :

1. Le minimum de la fonction $C(\alpha)$ est souvent difficile à calculer numériquement.
2. Ce critère a souvent du mal à distinguer le signal du bruit corrélé.
3. Ce critère nécessite le calcul de nombreuses solutions régularisées du problème inverse (m_α), solutions qui en général ne sont pas accessibles directement (par exemple les solutions obtenues par gradient conjugué sur l'équation normale, voir section 1.3.1)

Critère de la courbe en L

Ce critère est fondé sur une courbe d'équilibre entre deux objectifs : minimiser la norme du résidu et garder une norme de la solution pas trop grande. La courbe en L est une courbe dans laquelle on trace l'évolution de la norme de la solution régularisée en fonction de la norme du résidu pour chaque valeur du paramètre de régularisation. Son nom vient du fait que sa forme ressemble souvent à un L (pour les problèmes mal posés quand $\alpha \to 0$).

Le paramètre de régularisation optimal doit donner une solution régularisée qui se situe dans le voisinage "du coin" de la courbe en L. La figure 1.3 nous montre un exemple de courbe en L.

L'utilisation de cette courbe pour donner une estimation du paramètre de régularisation a été étudiée dans [90] et [92]. L'idée principale est d'interpoler la courbe en L de manière

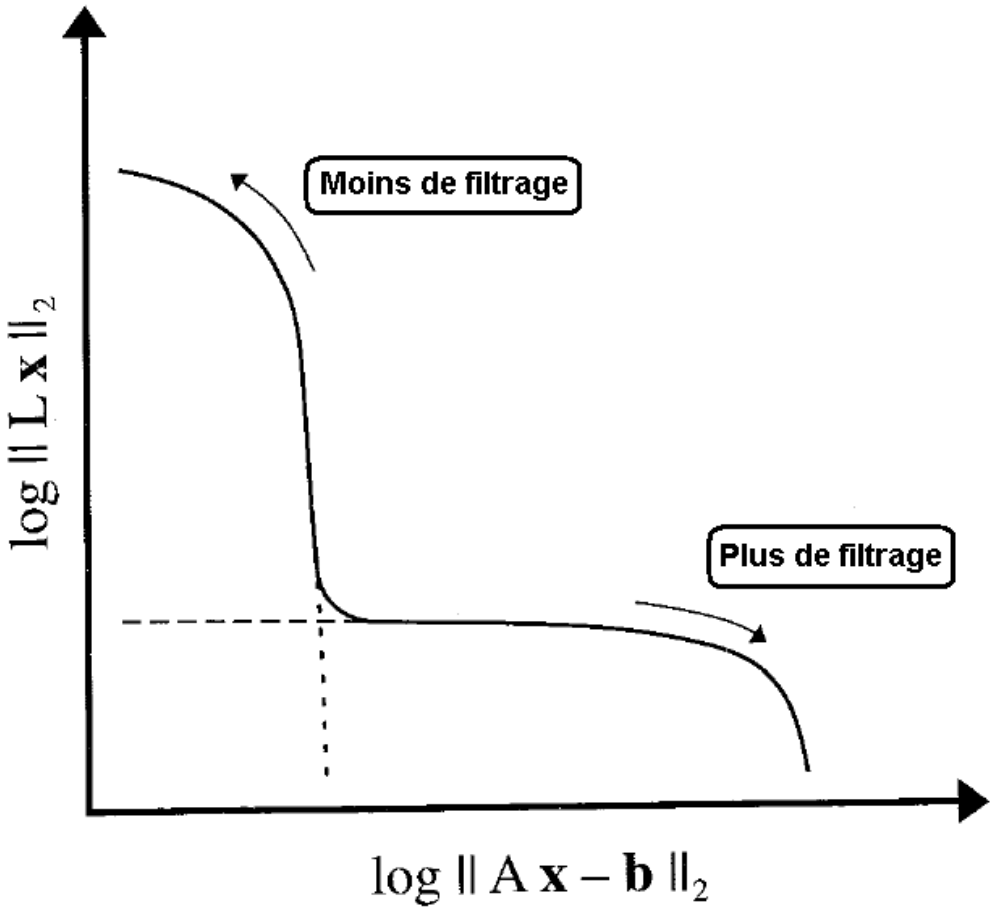

Fig. 1.3 Figure théorique de la courbe en L

à estimer précisément la position du "coin" (certains auteurs recommendent d'utiliser une échelle logarithmique pour mieux faire apparaître le coin). Le critère de la courbe en L est plus efficace que le critère de la validation croisée pour distinguer le signal du bruit corrélé. Cet avantage semble provenir du fait que la courbe en L utilise non seulement des informations sur la norme du résidu mais aussi sur la norme de la solution (voir [92]).

Les inconvénients principaux du critère de la courbe en L sont :

1. l'existence du "coin" de la courbe en L n'est pas garantie,
2. la solution dépend de l'échelle choisie car la notion de "coin", est mal définie
3. la nécessité de connaitre de nombreux points de la courbe afin de pouvoir réaliser une interpolation efficace du coin. En effet, l'obtention d'un point de la courbe est une tâche difficile : par exemple dans le cas de la regularisation de Tikhonov cela nécessite la minimisation pour un α fixé du problème (1.6). Les travaux de [88] et [154] ont mis en évidence d'autres difficultés liées à l'utilisation de ce critère pour estimer le paramètre de régularisation.

1.2 Optimisation sans contrainte : conditions d'optimalité

Nous rappelons dans cette section les résultats généraux de l'optimisation sans contrainte. Ces résultats sont expliqués plus en détails dans les livres suivants : [56], [13], [123] et [68].

Considérons le problème d'optimisation

$$(P) \quad \min_x f(x)$$

dans lequel on minimise une fonction $f : \Omega \mapsto \mathbb{R}$ (avec Ω un ouvert de $\mathbb{R}^n$) appelée indifféremment *critère*, *fonction objectif* et *fonction coût* du problème (P).

Définition 1.2.1 *On appelle minimum (global) de (P) tout point $x_* \in \Omega$ vérifiant*

$$f(x_*) \leq f(x), \quad \forall x \in \Omega.$$

Définition 1.2.2 *On dit que $x_* \in \Omega$ est un minimum local de (P) s'il existe un voisinage V de x_* tel que*

$$f(x_*) \leq f(x), \quad \forall x \in \Omega \cap V.$$

Définition 1.2.3 *On dit que $x_* \in \Omega$ est une solution de (P) si x_* est un minimum global ou local de (P)*

Les conditions nécessaires d'optimalité du problème d'optimisation (P) sont une réunion d'équations et/ou d'inéquations et/ou de propriétés que doivent vérifier les solutions de (P). On parle de conditions du premier ordre lorsque celles-ci ne font intervenir que les dérivées premières de f (idem pour les conditions du second ordre).

Le théorème suivant, repris du Corollaire 3.7 de [68], nous donne une condition nécessaire d'optimalité du premier ordre et pour les problèmes convexes une condition suffisante du premier ordre.

Théorème 1.2.4 *Supposons que $f : \Omega \mapsto \mathbb{R}$ soit dérivable en $x_* \in \Omega$. Si f a un minimum local en x_*, alors*

$$f'(x_*) = 0 \quad ou \quad \nabla f(x_*) = 0. \tag{1.7}$$

Inversement, si f est convexe et si (1.7) a lieu, alors x_* est un minimum global de f sur Ω.

Les deux résultats suivants, repris des propositions 3.8 et 3.9 de [68], nous donnent les conditions d'optimalité nécessaires puis suffisantes de (P) au second ordre.

Proposition 1.2.5 *Supposons que x_* soit un minimum local de f sur Ω et que f soit C^1 dans un voisinage de x_* et deux fois dérivables en x_*. Alors*

$$\nabla f(x_*) = 0 \quad et \quad \nabla^2 f(x_*) \text{ est semi-définie positive}.$$

Proposition 1.2.6 *Si f est dérivable dans le voisinage d'un point x_* et deux fois dérivables en x_* et si*

$$\nabla f(x_*) = 0 \quad et \quad \nabla^2 f(x_*) \text{ est définie positive},$$

alors x_* est un minimum local strict de f.

1.3 Problèmes de moindres-carrés non-linéaires

Les problèmes de moindres-carrés non-linéaires sont rencontrés dans de nombreuses applications (chimie, météo, économie et etc...). On cite notamment tous les problèmes d'identification de paramètres où les paramètres décrivant un modèle doivent être choisis pour "coller" au mieux aux données observées. Depuis plus de 30 ans, les spécialistes de l'analyse numérique et de l'optimisation ont développés des techniques efficaces et robustes pour résoudre ce type de problèmes. Dans cette section nous allons les analyser et décrire succintement les principales techniques de résolution.

Un problème de moindres-carrés s'écrit de la manière suivante :

$$\min_x \left(f(x) := \frac{1}{2}||r(x)||_2^2 \right), \tag{1.8}$$

où f est une fonction $\Omega \mapsto \mathbb{R}$ (avec Ω un ouvert de $\mathbb{R}^n$) et $r : \Omega \mapsto \mathbb{R}^n$ est appelé le vecteur résidu de x. On note $J(x) = r'(x)$ la jacobienne de r (J est de dimension $m \times n$).

Pour être efficace, les techniques de résolution de (1.8) doivent exploiter la structure particulière de f et de ces dérivées. Le gradient et le hessien de f s'expriment aisément en fonction de J :

$$\nabla f(x) = J(x)^\top r(x), \tag{1.9}$$

$$\nabla^2 f(x) = J(x)^\top J(x) + \sum_{i=1}^{p} r_i(x) \nabla^2 r_i(x). \tag{1.10}$$

L'une des questions essentielles lorsque l'on veut résoudre (1.8) est de savoir si la jacobienne $J(x)$ est directement accessible par le calcul. Cette question est importante car dans le cas positif cela veut dire que l'on peut aussi calculer le gradient de f par la formule (1.9) et, surtout, que l'on a une partie du hessien $\nabla^2 f(x)$ sans avoir à calculer de dérivées secondes (souvent coûteuse). Dans de nombreuses applications, dont en particulier celle qui nous occupera dans ce document, cette réponse est positive. La connaissance de $J(x)$ permet d'accéder gratuitement au premier terme du hessien de f ($J(x)^\top J(x)$). Or, ce premier terme de (1.10) est très souvent dominant par rapport au second terme de sommation $\sum_{i=1}^{p} r_i(x) \nabla^2 r_i(x)$ pour les deux raisons suivantes :

Remarques 1.3.1

1. *proche de la solution les résidus r_i adoptent un comportement quasi-linéaire, c'est à dire que les dérivées secondes $\nabla^2 r_i(x)$ deviennent "petites",*
2. *proche de la solution les résidus, les résidus r_i sont "petits" (cas en particulier des problèmes d'identifications).*

Les algorithmes de Gauss-Newton et de Levenberg-Marquardt (voir resp. section 1.3.3 et section 1.3.4) exploitent cette structure particulière du hessien pour résoudre (1.8).

Deux grandes classes d'algorithmes permettent de résoudre (1.8) en ne tirant partie que de l'information provenant des dérivées premières :

1. les algorithmes quasi-newtonnien,
2. les algorithmes de Gauss-Newton ou de Levenberg-Marquardt.

Ces deux grandes classes d'algorithmes se différencient par l'exploitation ou la non-exploitation de la structure particulière des problèmes de moindres-carrés. Les méthodes quasi-newtonnienne, que nous ne dévellopperons pas dans ce travail, n'utilisent pas cette structure particulière : elles ont un champ d'applications beaucoup plus vaste et peuvent être appliquées à la minimisation de fonction coût non-linéaire plus générale. Elles sont particulièrement recommandée sur les problèmes de moindres-carrés lorsque le calcul de la matrice jacobienne J prend beaucoup plus de temps CPU celui des résidus r_i (voir [41] et [42]). A l'opposée, les algorithmes de Gauss-Newton et de Levenberg-Marquardt tire partie de la structure du hessien pour résoudre (1.8). Ils sont particulièrement efficace lorsque l'une des hypothèses de la remarque 1.3.1 est satisfaite. Si aucune de ces deux hypothèses n'est satisfaite, alors ces algorithmes peuvent converger très lentement vers la solution. Afin d'accélerer cette convergence, certains auteurs (voir [49] et [159]) proposent de combiner les méthodes de Gauss-Newton et de quasi-Newton pour obtenir approximation du hessien de f meilleure que l'approximation classique par $J(x)^\top J(x)$.

1.3.1 Cas particulier des problèmes de moindres-carrés linéaires

Dans le cas de résidus linéaires l'équation (1.8) se simplifie en :

$$\min_x \left(f(x) =: \frac{1}{2}||Jx + r||_2^2 \right), \tag{1.11}$$

où $r = r(0)$. Le gradient et le hessien de f s'écrivent alors

$$\nabla f(x) = J(x)^\top (Jx + r), \quad \nabla^2 f(x) = J(x)^\top J(x). \tag{1.12}$$

On note que le second terme de (1.10) est absent dans le cas linéaire car les dérivées secondes des résidus sont nulles ($\nabla^2 r_i(x) = 0$ pour tout i). On remarque aussi que la fonction f de (1.11) est convexe. Ces problèmes sont étudiés depuis de nombreuses années et il existe de nombreux livres d'algèbre numérique traitant de leurs résolutions comme [80] et [143], plus spécifiquement on cite aussi les travaux de [106] et plus récemment ceux de [16].

En utilisant la condition d'optimalité du premier ordre des problèmes d'optimisation sans contrainte ($\nabla f(x_*) = 0$, voir théorème (1.2.4)) on obtient l'équation normale qui caractérise les solutions de (1.11) :

$$J^\top J x = -J^\top r. \tag{1.13}$$

La proposition suivante, reprise de la proposition 10.1 de [68], donne un résultat d'existence et d'unicité de (1.11).

Proposition 1.3.2 *Le problème* (1.11) *est convexe et admet toujours une solution. Celle-ci est unique si et seulement si J est injective. L'ensemble des solutions de* (1.11) *s'écrit $x_p + N(J)$, où x_p est une solution particulière de* (1.11) *et $N(J)$ est le noyau de J.*

Une technique importante permettant d'analyser les problèmes de moindres-carrés linéaires est la *décomposition en valeurs singulières* (ou décomposition SVD) de la matrice J. Cela consiste à transformer J sous la forme :

$$J = U \Sigma V,$$

tel que

$$\begin{cases} U \in \mathbb{R}^{m \times m}, V \in \mathbb{R}^{n \times n}, \Sigma \in \mathbb{R}^{m \times n} \\ U^\top U = I_n \\ V V^\top = V^\top V = I_n \\ \Sigma = \text{diag}(\sigma_1, ..., \sigma_p), \quad p = \min(m, n) \end{cases}$$

où $\sigma_1 \geq \sigma_2 \geq ...\sigma_p \geq 0$ et où les σ_i sont les valeurs singulières de J. Les colonnes de U (resp. V) sont les vecteurs singuliers à gauche (resp. droite) de J. A partir de la décomposition SVD de J on peut définir le conditionnement d'un problème de moindres-carrés.

Définition 1.3.3 *On définit le conditionnement de* (1.11) *en norme* l_2 *par :*

$$\mathit{Cond}(J) = \frac{\sigma_1}{\sigma_r},$$

où σ_r *est la plus petite valeur singulière non nulle de* J *(*$r = rg(J)$*)*

Le théorème suivant, repris du paragraphe 5.3.8 de [80], fournit un moyen de mesurer la sensibilité de la solution du problème (1.11) à des perturbations sur les données.

Théorème 1.3.4 *On suppose que* x*,* y*,* $\hat{x}$ *et* $\hat{y}$ *vérifient les équations suivantes :*

$$\begin{aligned} ||Jx + r|| &= \min, \quad y = Jx + r \\ ||(J + \delta J)\hat{x} + (r + \delta r)|| &= \min, \quad \hat{y} = (J + \delta J)\hat{x} + (r + \delta r), \end{aligned}$$

où J *et* δJ *appartiennent à* $\mathbb{R}^{m\times n}$ *(*$m \geq n$*),* $b \neq 0$ *et* $b \in \mathbb{R}^m$*. Si*

$$\epsilon = \max\left(\frac{||\delta J||}{||J||}, \frac{||\delta r||}{||r||}\right) < \frac{\sigma_r}{\sigma_1}$$

et

$$\sin(\theta) = \frac{||y||}{||r||} \neq 1$$

alors

$$\frac{||\hat{x} - x||}{||x||} \leq \epsilon\left(\frac{2\mathit{Cond}(J)}{\cos(\theta)} + \tan(\theta)\mathit{Cond}(J)^2\right) + \mathbb{O}(\epsilon^2) \tag{1.14}$$

$$\frac{||\hat{y} - y||}{||y||} \leq \epsilon(1 + 2\mathit{Cond}(J)^2)\min(1, m - n) + \mathbb{O}(\epsilon^2) \tag{1.15}$$

La partie droite de (1.14) est proportionnelle au conditionnement de la matrice J dans le cas d'un résidu nul ($y = 0$). Dans le cas d'un résidu non nul cette partie est alors proportionnelle au carré du conditionnement. Dès lors, on comprend qu'un mauvais conditionnement implique que la solution du problème de moindres-carrés est instable, i.e. elle est très sensible aux perturbations sur les données. De tels problèmes sont appelés des problèmes *mal conditionnés*. En pratique, lorsque l'on rencontre de tels problèmes, les recommandations usuelles sont de ne pas faire confiance à la solution obtenue par des méthodes classiques et de reformuler le problème pour éliminer son mauvais conditionnement. Pour résoudre de tels problèmes il faut faire appel aux techniques de régularisation numérique (voir section 1.1.3).

Pour les problèmes de moindres-carrés bien conditionnés, il existe différentes techniques classiques permettant de résoudre (1.11). Ces techniques sont classées suivant

1. la taille du problème de moindres-carrés à résoudre : on distingue les problèmes de petite et moyenne dimension aux problèmes de grande taille ; dans ce travail nous nous sommes plus particulièrement intéréssés à la résolution des problèmes de grande taille,

2. l'utilisation ou la non utilisation de l'équation normale (1.13) pour résoudre (1.11).

La solution des problèmes de taille petite et moyenne peut s'obtenir par l'utilisation de méthodes directes puisque dans ce cas il est concevable de réaliser des factorisations matricielles. Les deux principales méthodes directes utilisées pour résoudre (1.11) sont :

1. résolution directe de (1.11) par factorisation QR de J,
2. résolution de (1.13) par factorisation de Cholesky de $J^\top J$.

La première méthode repose sur la formulation (1.11) du problème original, alors que la seconde s'appuie sur l'équation normale (1.13).

La solution des problèmes de moindres-carrés de grande taille fait appel à des techniques itératives, d'une part à cause des limitations de stockage, et d'autre part à cause de la difficulté de calculer explicitement les coefficients des matrices concernées. La technique la plus utilisée consiste à résoudre (1.13) par gradient conjugué. Cette dernière approche est particulièrement efficace lorsque l'algorithme du gradient conjugué est préconditionné. La proposition suivante, reprise de la proposition 10.2 de [68], nous confirme que l'algorithme du GC peut être utilisé pour résoudre l'équation normale (1.11) (en théorie il n'est utilisable que pour des matrices définies positives).

Proposition 1.3.5 *L'algorithme du gradient conjugué pour minimiser* (1.11) *est bien défini et converge en au plus* r *itérations (avec* $r = rg(J)$*). De plus, si l'itéré initial est pris dans* $R(J^\top)$ *(par exemple* $x_1 = 0$*), les itérés convergent vers la solution de norme minimale de* (1.11)*.*

1.3.2 Une méthode de régularisation : l'algorithme gradient du conjugué sur l'équation normale

Cette approche consiste à appliquer la méthode du gradient conjugué à l'équation normale

$$J^\top J x = -J^\top r.$$

L'obtention de la matrice $J^\top J$ est souvent une source d'erreurs d'arrondi (voir exemple 5.3.2 de [80]). Cependant, l'algorithme du gradient conjugué peut être programmé sans former explicitement cette matrice. Cette méthode s'appelle la méthode du gradient conjugué sur l'équation normale (CGLS). Cette méthode a été utilisée avec succès sur de nombreux problèmes de moindres-carrés possédant des données bruitées. Son succès provient principalement de l'effet regularisant intrinsèque aux itérations de gradient conjugué.

Afin de préciser l'effet régularisant des itérations de gradient conjugué, rappelons-nous tout d'abord que l'algorithme du gradient conjugué génère des itérés dans un sous-espace de Krylov. En particulier, dans l'algorithme CGLS, l'itéré x_k appartient au sous-espace de Krylov $K_k(J^\top J, -J^\top r)$ défini comme

$$K_k(J^\top J, -J^\top r) = \text{vect}\{-J^\top r, -(J^\top J)J^\top r, ..., -(J^\top J)^{k-1}J^\top r\}.$$

Notons qu'à l'itération k, x_k est solution du problème de minimisation

$$\begin{cases} \min_x \left(f(x) = \frac{1}{2}||Jx + r||_2^2 \right) \\ x \in K_k(J^\top J, -J^\top r). \end{cases}$$

Dans certains problèmes, l'ensemble $K_k(J^\top J, -J^\top r)$ approche le sous-espace formé par les vecteurs singuliers à droite de J (les v_i) associés aux k plus grandes valeurs singulières. Ainsi x_k se décompose en une somme de vecteurs singuliers associés aux grandes valeurs singulières de J. Cependant, lorsque k augmente, des vecteurs singuliers associés à de petites valeurs singulières finissent par être pris en compte dans cette approximation. Cela implique que la contribution du bruit commence à apparaître dès le début de la divergence des itérés de gradient conjugué. Ce phénomène est plus connu sous le nom de semi-convergence des itérés de gradient conjugué.

La semi-convergence de l'algorithme du gradient conjugué implique qu'il est nécéssaire d'arrêter ses itérations avant que l'effet du bruit apparaisse. Dans ce cas c'est le nombre d'itération k de gradient conjugué qui joue le rôle du paramètre de régularisation. Cette méthode est très sensible à la valeur de l'itération k à laquelle elle est arrêtée. Le critère de la courbe en L ainsi que le critère de validation croisée de Monte Carlo sont deux techniques qui permettent d'évaluer l'itération k à laquelle le gradient conjugué doit être stoppé (voir [91] et [73]).

Voici ci dessous l'algorithme CGLS (non préconditionné).

Data : Choix un vecteur x_0.
Initialisation du résidu y_0 par : $y_0 = J^\top x_0 + r$.
$g_0 = Jy_0$, $p_0 = -g_0$ et $k = 0$.
begin
  while $y_k \neq 0$ **do**
    $q_k = J^\top p_k$
    $\alpha_k = \frac{||g_k||_2^2}{||q_k||_2^2}$
    $x_{k+1} = x_k + \alpha_k p_k$
    $y_{k+1} = y_k + \alpha_k q_k$
    $g_{k+1} = Jy_{k+1}$
    $\beta_k = \frac{||g_{k+1}||_2^2}{||q_k||_2^2}$
    $p_{k+1} = -g_{k+1} + \beta_k p_k$
    $k := k + 1$
  endw
end

Algorithme I.1 Algorithme CGLS

Quelques remarques sur cet algorithme :

Remarques 1.3.6

1. *L'algorithme CGLS est bien adapté aux problèmes de grande taille car :*
   (i) les matrices J et $J^\top$ sont seulement utilisées pour effectuer des produits matrice-vecteur. Il requiert peu de données à stocker en mémoire.
   (ii) sa convergence est théoriquement rapide (voir proposition 1.3.5).
2. *Les inconvénients principaux de cet algorithme sont*
   (i) le choix délicat de l'itération à laquelle il doit être stoppé.
   (ii) l'altération de la vitesse de convergence, voire la non convergence lorsque le problème est très mal-conditionné.

Afin d'améliorer la convergence de l'algorithme CGLS on utilise souvent en parallèle des techniques de préconditionnement. Le préconditionnement des problèmes mal-posés est souvent très délicat à effectuer. Son objectif est d'améliorer le conditionnement de la matrice $J^\top J$ en regroupant ses valeurs propres et/ou en les rapprochant de 1. Cependant, dans les cas des problèmes de moindres-carrés mal-posés, il n'est pas souhaitable de transformer tout le spectre de la matrice.

Dans le cas où l'on préconditionnerait toutes les valeurs propres de $J^\top J$, cela implique que l'on mélangerait l'information basse fréquence à l'information haute fréquence provenant de la matrice. On calculerait alors des itérés de gradient conjugué qui contiennent des contributions importantes du bruit. Ainsi, dans le cas de problèmes mal conditionnés il est important de préconditionné seulement la partie haute du spectre tout en gardant la partie basse identique. Cet argument a été souvent observé, il est analysé plus profondément dans [89] et dans le chapitre 5 de [91].

En général il n'est pas possible à priori de distinguer la partie haute de la partie basse du spectre d'une matrice. Cela explique pourquoi le préconditionnement des problèmes de moindres-carrés est difficile à effectuer. Aujourd'hui encore de nombreuses recherches sont effectuées dans ce domaine.

Notons qu'il existe une technique rapide qui permet d'obtenir une borne infèrieure du conditionnement de la matrice $J^\top J$ au cours des itérations de gradients conjugués. Cette technique est basée sur les propriétés des quotients de Rayleigh.

Définition 1.3.7 *Soit H une matrice symmétrique, et x un vecteur non nul alors le scalaire*

$$Q_{rayleigh}(H, x) = \frac{x^\top H x}{x^\top x}$$

est appelé le quotient de Rayleigh de x pour la matrice H.

Les propriétés intéréssantes des quotients de Rayleigh sont données dans la proposition suivante :

Proposition 1.3.8 *Soit H une matrice symmétrique alors les quotients de rayleigh de H sont bornés infèrieurement (resp. supérieurement) par la valeur propre minimale (resp. maximale) de H :*

$$\forall x \neq 0 \quad \lambda_{min} \leq Q_{rayleigh}(H, x) \leq \lambda_{max},$$

où λ_{min} (resp. λ_{max}) est la valeur propre la plus petite (resp. grande) de H. De plus ces bornes sont atteintes car :

$$\lambda_{min} = \frac{v_{min}^\top H v_{min}}{v_{min}^\top v_{min}} \quad \textit{et} \quad \lambda_{max} = \frac{v_{max}^\top H v_{max}}{v_{max}^\top v_{max}},$$

où v_{min} (resp. v_{max}) est un vecteur propre de H associé à la valeur propre λ_{min} (resp. λ_{max})

A chaque itération de gradient conjugué (voir algorithme I.1), le scalaire

$$\frac{p_k^\top J^\top J p_k}{p_k^\top p_k} = \frac{||q_k||_2^2}{||p_k||_2^2},$$

peut-être calculé facilement (ce calcul nécessite un seul produit scalaire supplémentaire pour obtenir $||p_k||_2^2$). Or, ce scalaire n'est autre que le quotient de Rayleigh $Q_{rayleigh}(J^\top J, p_k)$. On peut alors se servir des ces quotients pour évaluer rapidement le conditionnement de la matrice $J^\top J$ au cours des itérations de gradient conjugué. Il s'agit de calculer les scalaires suivant :

$$\tilde{\lambda}_{max} = \max_{k=1,\dots,N_{iter}} Q_{rayleigh}(J^\top J, d_k) \quad \textit{et} \quad \tilde{\lambda}_{min} = \min_{k=1,\dots,N_{iter}} Q_{rayleigh}(J^\top J, d_k),$$

où N_{iter} est la dernière itération de gradient conjugué. On obtient alors une borne inférieure du conditionnement de $J^\top J$ par le quotient des deux quantités précédentes :

$$\text{Cond}(J^\top J) \geq \frac{\tilde{\lambda}_{max}}{\tilde{\lambda}_{min}}. \tag{1.16}$$

1.3.3 Méthode de Gauss-Newton

Nous allons décrire dans cette sous-section la méthode de Gauss-Newton pour résoudre le problème de moindres-carrés non linéaires (1.8). Cette méthode consiste à minimiser f en tentant d'annuler son gradient. Pour ce faire, elle utilise l'algorithme de Newton sur l'équation $\nabla f(x) = 0$ en prenant garde de retirer les dérivées secondes de r dans le calcul de la hessienne de f (voir équation (1.12)). On approche donc $\nabla^2 f(x)$ par $J(x)^\top J(x)$, puis on calcule en x_k une direction d_k en résolvant le système linéaire

$$J_k^\top J_k d_k = -J_k^\top r_k = -g_k, \tag{1.17}$$

où on note $J_k = J(x_k)$ et $r_k = r(x_k)$.

La méthode de Gauss-Newton s'interprète aussi comme une méthode de quasi-linéarisation. En effet, par linéarisation du résidu en x_k dans l'équation (1.8) on définir le problème linéarisé de (1.8).

Définition 1.3.9 *On appelle **problème linéarisé** du problème (1.8) le problème de moindres-carrés linéaire suivant :*

$$\min_d \frac{1}{2}||r_k + J_k d||_2^2. \tag{1.18}$$

En écrivant l'équation d'optimalité de ce problème on retrouve aisément que la direction d_k définie par ce problème n'est autre que celle trouvée par le système (1.17). Cette interprétation est intéressante car comme (1.18) est convexe, on en déduit que l'équation d'optimalité est nécessaire et suffisante (voir théorème 1.2.4).

Le résultat suivant, repris du lemme 10.3 de [68], montre que la direction d_k obtenue en (1.17) est une direction de descente en x_k et qu'elle peut donc être utilisée dans le cadre d'une globalisation de l'algorithme par recherche linéaire.

Lemme 1.3.10 *Il existe toujours une direction d_k vérifiant* (1.17)*. Si x_k n'est pas un point stationnaire du problème de moindres-carrés non linéaire* (1.8)*, d_k est une direction de descente de f en x_k.*

En résumé, voici les principaux arguments conduisant à l'utilisation de la méthode de Gauss-Newton pour résoudre (1.8) :

1. L'approximation du hessien de f par $J(x)^\top J(x)$ est intéressante car elle utilise uniquement le calcul de la jacobienne $J(x)$, le coût de calcul des dérivées secondes des residus pouvant être important.
2. Dans de nombreuses applications le terme $J^\top J$ approchant le hessien de f est dominant par rapport aux termes regroupant les dérivées secondes des résidus. Cela se produit plus particulièrement dans le cas d'applications où :
   - les résidus sont petits ($r_i \approx 0$),
   - les résidus sont quasi-linéaires ($||\nabla^2 r_i|| \approx 0$).
3. Le lemme 1.3.10 assure que la direction de Gauss-Newton d_k trouvée par résolution de (1.17) est bien une direction de descente de f.
4. L'interprétation de l'algorithme de Gauss-Newton par une méthode de quasi-linéarisation fait le lien avec les méthodes utilisables dans le cas de problèmes de moindres-carrés linéaires. La méthode de Gauss-Newton peut alors se voir comme la transformation d'un problème de moindres-carrés non linéaires en la résolution d'une suite de problèmes de moindres-carrés linéaires. Ainsi, on peut appliquer les méthodes de la section 1.3.1 pour résoudre le problème (1.18).

Voici ci-dessous l'algorithme de Gauss-Newton appliqué à la résolution de (1.8).

Data : Choix d'un itéré initial x_1.
Initialisation : $k = 1$.
Constante $0 < \omega_1 < \frac{1}{2}$.

begin

while $J(x_k)^\top r(x_k) \neq 0$ **do**

((1)) Calcul de la direction de descente : prendre pour d_k une solution de (1.17).

((2)) Déterminer le pas α_k par "rebroussement", i.e. en prenant le plus grand α_k dans $\{1, \frac{1}{2}, \frac{1}{4}, ...\}$ tel que

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \omega_1 \alpha_k (g_k, d_k).$$

((3)) Mettre à jour l'itéré :

$$x_{k+1} = x_k + \alpha_k d_k.$$

((3)) Accroître k de 1 : $k := k + 1$.

endw

end

Algorithme I.2 Algorithme de Gauss-Newton globalisé par recherche linéaire

Quelques remarques sur cet algorithme :

1. L'étape la plus délicate de l'algorithme est l'étape ((1)) qui consiste à résoudre le système linéaire (1.17). Pour ce faire, suivant la taille du système à résoudre on pourra soit faire appel à des méthodes de factorisation (petit et moyen système) soit à des méthodes itératives de type gradient conjugué (grand système).
2. L'étape ((2)) de l'algorithme permet d'assurer la convergence lorsque l'itéré initial est "loin" de la solution. La technique de recherche linéaire permet d'assurer la décroissance de f à chaque itération.
3. La proposition suivante, reprise de la proposition 10.5 de [68], donne un résultat de convergence de l'algorithme de Gauss-Newton.

   Proposition 1.3.11 *Soit $\{x_k\}$ une suite générée par l'algorithme de Gauss-Newton. Si $\{J(x_k)\}$ est bornée et uniformément injective, alors $J(x_k)^\top r(x_k) \to 0$.*
4. L'algorithme de Gauss-Newton s'interprète presque comme un algorithme de Newton avec recherche linéaire. La différence entre ces 2 algorithmes réside dans l'approximation du hessien de f et dans le fait qu'on sur içi d'avoir une direction de descente.

Dans la sous-section suivante nous nous intéresserons à la méthode de Levenberg-Marquardt qui est le pendant de la méthode de Gauss-Newton mais cette fois-ci avec une globalisation par région de confiance.

1.3.4 Méthode de Levenberg-Marquardt

Dans cette sous-section nous décrivons la méthode de Levenberg-Marquardt appliquée à la résolution du problème de moindres-carrés non linéaires (1.8). Cette méthode est connue depuis déjà de nombreuses années, c'est Levenberg en 1944 qui le premier publie (voir [109]), il faut ensuite attendre 1963 pour voir ceux de Marquardt (voir [112]).

Comme pour la méthode de Gauss-Newton (voir la sous-section précédente), la méthode de Levenberg-Marquardt est une méthode itérative fondée sur un schéma newtonien qui prend en compte l'approximation du hessien $\nabla^2 f(x)$ par $J(x)^\top J(x)$. Dans certaines applications, par exemple lorsque l'approximation du hessien par $J_k^\top J_k$ n'est pas définie positive, la direction d_k trouvée l'algorithme de Gauss-Newton ne produit pas une réduction suffisante de la fonction f. Dans ce cas, la méthode de Levenberg-Marquardt permet de forcer la convergence en "rapprochant" la direction de descente d_k de la direction $-g_k$ (i.e. la direction formée par l'opposé du gradient de f en x_k). Cette méthode consiste à approcher le hessien $\nabla^2 f(x)$ par $J_k^\top J_k + \mu_k I$ ($\mu_k > 0$). Ainsi, au lieu de résoudre le système linéaire (1.17), on résout le système suivant :

$$(J_k^\top J_k + \mu_k I) d_k = -g_k, \tag{1.19}$$

où $\mu_k > 0$ est une constante choisie à chaque itération. Le choix de la constante μ_k permet de controler la direction de descente d_k. Ainsi, lorsque μ_k est grand, l'algorithme privilégie la direction du gradient pour forcer la convergence et lorsque μ_k est petit l'algorithme fait plus confiance au terme $J_k^\top J_k$. En pratique, cette constante est augmentée ou diminuée d'un facteur en fonction de la décroissance effective de f à l'itération précédente.

L'équation (1.19) est en fait l'équation normale du problème de moindres-carrés linéaire suivant :

$$\min_d \frac{1}{2} \left(||r_k + J_k d||_2^2 + \mu_k ||d_k||_2^2 \right). \tag{1.20}$$

On observe que le problème (1.20) est presque identique au problème (1.18) sauf qu'on lui a rajouté en plus le terme de régularisation $\mu_k ||d_k||_2^2$ sur la norme de la perturbation du modèle. Ainsi, la méthode de Levenberg-Marquardt peut aussi s'interpréter comme une méthode de régularisation du problème (1.20).

Il a été démontré par Moré (voir [116]) que la résolution du problème (1.19) est équivalente à la résolution du problème de région de confiance suivant :

$$\begin{cases} \min_d \frac{1}{2} ||r_k + J_k d||_2^2, \quad \text{s.t. } ||d_k|| \leq \Delta_k \\ \Delta_k = ||(J_k^\top J_k + \mu_k I)^{-1} J^\top r_k||. \end{cases} \tag{1.21}$$

Ce problème est identique au problème de moindres-carrés (1.18) sauf qu'il comporte en plus la contrainte $||d_k|| \leq \Delta_k$ correspondant à la technique des régions de confiance. L'équation de la deuxième ligne relie la valeur de Δ_k à celle de μ_k. Le choix automatique de Δ_k dans la méthode des régions de confiance permet de déterminer une valeur convenable pour le paramètre μ_k. On appelle la fonction

$$m_k(d) = \frac{1}{2} ||r_k + J_k d||_2^2$$

le modèle quadratique de f en x_k.

A partir de ces considérations, la méthode de Levenberg-Marquardt peut se voir comme la méthode de Gauss-Newton globalisée par régions de confiance.

Data : Choix d'un itéré initial x_1.
Choix d'une région de confiance initiale $\Delta_1 > 0$.
Initialisation : $k = 1$.
begin
while $J(x_k)^\top r(x_k) \neq 0$ **do**
((1)) Calculer (approximativement) la solution d_k du sous-problème :

$$\begin{cases} \min_d m_k(d) \\ ||d|| \leq \Delta_k. \end{cases}$$

((2)) Calcul du rapport de concordance

$$\rho_k = \frac{f(x_k + d_k) - f(x_k)}{m(d_k) - m(0)}.$$

((3)) Mettre à jour le rayon Δ_k (en fonction de ρ_k et d_k).
((4)) Mise à jour de x si $\rho_k > \eta$ ($0 < \eta < 1/2$ est une constante) :

$$x_{k+1} = x_k + d_k.$$

((5)) Accroître k de 1 : $k := k + 1$.
endw
end

Algorithme I.3 Algorithme de Gauss-Newton globalisé par régions de confiance

Quelques remarques sur cet algorithme

1. L'étape ((1)) de l'algorithme est l'étape la plus délicate de l'algorithme. Pour les systèmes de grande taille il est intéréssant d'utiliser un gradient conjugué tronqué (approche de Steihaug, voir [142]) pour trouver une solution approchée de I.3.
2. Les étapes ((2)) et ((3)) constituent la mécanique des méthodes de regions de confiance.
2. La proposition suivante, adaptée du théorème 10.2 de [123], donne un résultat de convergence de l'algorithme de Gauss-Newton globalisée par région de confiance.

   Proposition 1.3.12 *Soit $\{x_k\}$ une suite générée par l'algorithme I.3. Si $\{J(x_k)\}$ est bornée est si en chaque itéré x_k, la direction d_k assure une réduction suffisante de m_k (au moins autant que le pas de Cauchy), alors $J(x_k)^\top r(x_k) \to 0$.*

   Ce résultat est remarquable car contrairement à l'algorithme de Gauss-Newton glo-

balisé par recherche linéaire il n'y a pas d'hypothèse sur l'injectivité de J. De plus, on observe qu'il pas nécessaire de résoudre complètement le sous-problème (1.21) pour avoir la convergence.

Chapitre 2

La tomographie de réflexion

2.1 Introduction

La tomographie de réflexion est une méthode géophysique dont l'objectif est de déterminer un modèle de vitesse de propagation des ondes dans le sous-sol à partir des temps de trajet des ondes sismiques. A ses débuts, seuls les temps de propagation des rais en transmission (voir par exemple [17] et [1]) étaient utilisés. Puis, elle fut reprise pour les temps de propagation des ondes réfléchies (généralement les temps d'arrivée des réflexions primaires) afin de pouvoir déterminer les principaux évènements structuraux du sous-sol (voir les travaux précurseurs de [15] et [54]). A l'IFP, dans le cadre des consortiums PSI ("Prestack Structural Interpretation") et KIM ("Kinematic Inversion Methods), le logiel *jerry* (voir [98], [97]) a été développé pour traiter les problèmes de tomographie de réflexion 3D.

Il est à noter que d'autres méthodes de détermination de vitesse du sous-sol existent. Parmi les méthodes les plus classiques on trouve :

- la correction NMO ("Normal Move Out"),
- la correction DMO ("Dip Move Out").

Ces deux méthodes, classées dans les méthodes dites temporelles (voir le chapitre 2 de [21] et les références qui y sont citées) sont fondées sur l'hypothèse que la variation des temps de trajet des ondes réfléchies en fonction de l'offset (distance source-récepteur) est une hyperbole. Cette hypothèse est réalisée sous les conditions suivantes :

- la vitesse de propagation ne doit pas varier beaucoup latéralement,
- les réflecteurs ne doivent pas avoir des pentes trop importantes.

Ainsi, les méthodes classiques de correction NMO et DMO sont inadaptées dans le cas de modèle de vitesse comportant des structures complexes.

Il existe une autre classe de méthodes permettant de retrouver un modèle de vitesse du sous-sol. Cette classe, dénommée analyse de vitesse de migration, s'appuie sur des données qui ne proviennent plus du domaine temporel mais du domaine migré profondeur

(via une opération de conversertion temps-profondeur). Parmi les plus connues on cite :

- les méthodes DFA (“Depth Focusing Analysis”),
- les méthodes RCA (“Residual Curvature Analysis”),
- les tomographies MVA (“Migration Velocity Analysis”).

Pour plus d’informations sur ces méthodes nous renvoyons le lecteur au chapitre 2 de [21]. Ces méthodes sont généralement coûteuses en temps CPU car elles nécessitent plusieurs étapes de migration, opération coûteuse en 3D.

Contrairement aux méthodes temporelles classiques, le point fort de la tomographie de réflexion est qu’elle ne fait aucune hypothèse sur les variations des temps de trajet en fonction de l’offset. Elle est ainsi particulièrement recommandée lors de la recherche d’un modèle de vitesse composé de structures complexes. En tomographie de réflexion il est possible d’inverser toutes sortes de données dont par exemple :

- les réflexions multiples (voir [27]),
- les arrivées multiples (voir [33]).

Cependant, dans ce travail on ne s’intéresse qu’aux données qui correspondent à des réflexions primaires et à des arrivées premières. Nous noterons que les algorithmes d’optimisation développés dans la suite de ce travail sont indépendants du type de données à inverser. La tomographie de réflexion nécessite une phase d’interprétation des données. Cette phase est délicate, notamment pour les structures complexes. Les données, extraites des enregistrements sismiques, peuvent être non seulement entachées d’erreurs de mesure mais aussi d’erreurs d’interprétation (voir remarque 2.2.2). On comprend dès lors que la confiance à accorder aux résultats de la tomographie de réflexion est fortement dépendante de la qualité des données interprétées à inverser dépendant elle-même de la qualité des données sismiques.

2.2 Principe

L’objectif de la tomographie de réflexion est de retrouver le champ de vitesse et la géométrie des réflecteurs vérifiant les temps de trajet des réflexions primaires majeures enregistrées en surface.

Définition 2.2.1 *On appelle **données** ou **observations** de la tomographie de réflexion les temps de trajet des ondes sismiques associées à des réflexions primaires majeures. L’obtention de ces données s’effectue par une étape d’interprétation appelée le **pointé** : à partir d’informations structurales fournies par le géologue, le géophysicien identifie sur les sections sismiques les temps de trajet des ondes associées à des réflexions primaires majeures (voir figure 2.1). A chaque temps de trajet pointé est associé une **signature** de rayon (S,R,i) où S est la source de laquelle est émise l’onde acoustique, R est le récepteur enregistrant le temps de trajet et i l’interface sur laquelle l’onde s’est réfléchie (voir figure 2.2 pour une illustration de la signature d’un rayon). On peut alors regrouper l’ensemble des*

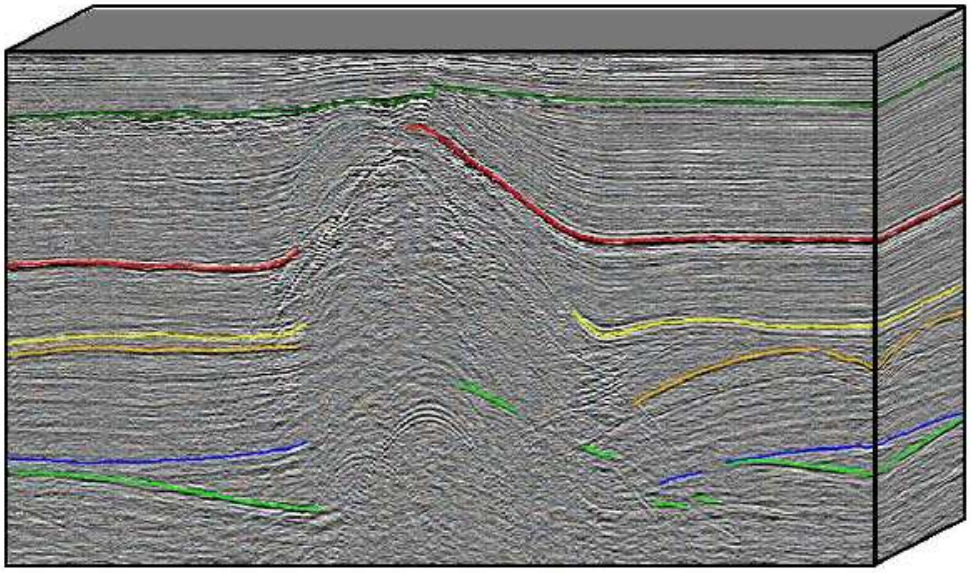

Fig. 2.1 Illustration de l'interprétation d'une coupe sismique : les évènements sismiques identifiés par des marqueurs de couleur correspondent à des contrastes de vitesse majeur (figure 6.5 de [21]).

temps de trajet pointés dans un unique vecteur T^{obs} défini par :

$$T_k^{obs}, \quad k = 1, N_{obs},$$

où N_{obs} représente le nombre de temps de trajet pointés et k est l'indice qui numérote les triplets (S,R,i).

Soit T l'opérateur (non linéaire) de modélisation physique qui permet de générer des réponses synthétiques $T(m)$ associées à un modèle m. Cette application est définie par :

$$\begin{aligned} T: \quad & \mathcal{M} \to \mathcal{D} \\ & m \mapsto T(m), \end{aligned}$$

où $\mathcal{M}$ est l'espace des modèles et $\mathcal{D}$ est l'espace des données.

On recherche donc le modèle $\hat{m} \in \mathcal{M}$ qui vérifie l'équation :

$$T(\hat{m}) = T^{obs}. \tag{2.1}$$

Les équations de la physique permettent de calculer les temps de trajet associés à un modèle de sous-sol. Ainsi, le calcul de $T(m)$ à partir de m s'appelle le ***problème direct*** : il est dans le sens des équations de la physique. A l'opposé, l'objectif de la tomographie de réflexion, qui consiste à retrouver le modèle m à partir de données observées T^{obs} est dans le sens inverse des équations de la physique. Ainsi, le problème 2.1, est identifié comme le ***problème inverse*** de la tomographie de réflexion. La figure 2.3 illustre ces deux problèmes de la tomographie de réflexion.

Il est généralement impossible de satisfaire (2.1) car les données sismimiques sont corrompues par du bruit ($T^{obs} \notin T(\mathcal{M})$).

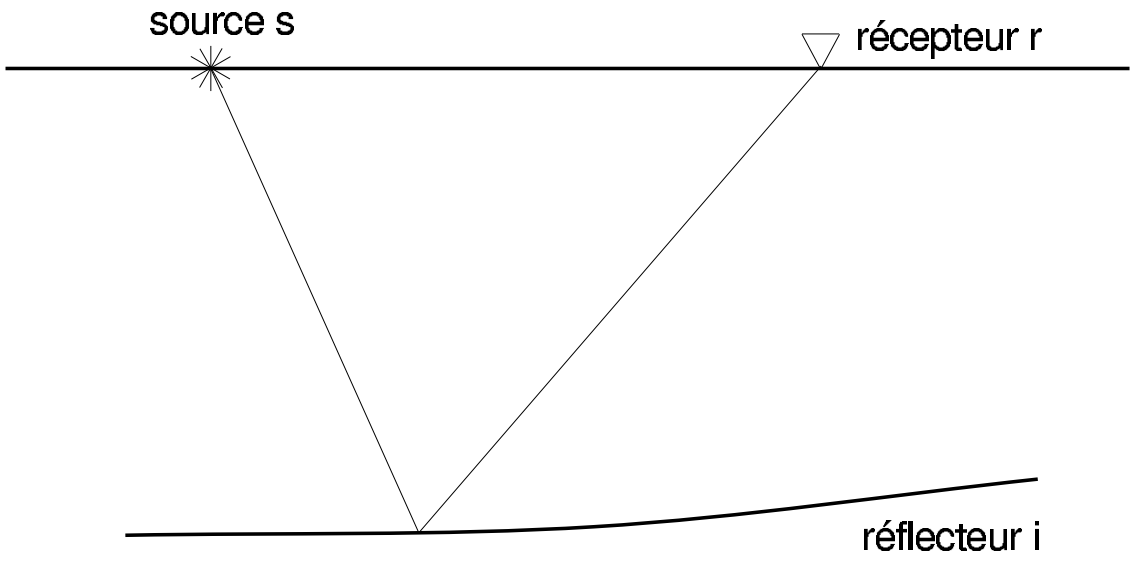

Fig. 2.2 Illustration d'un rayon de signature (S,R,i) : l'onde est émise de la source S, se propage dans le champ de vitesse V, se réfléchit sur le réflecteur i et se propage jusqu'à la surface au récepteur R où le temps de trajet est enregistré (figure reprise de [140]).

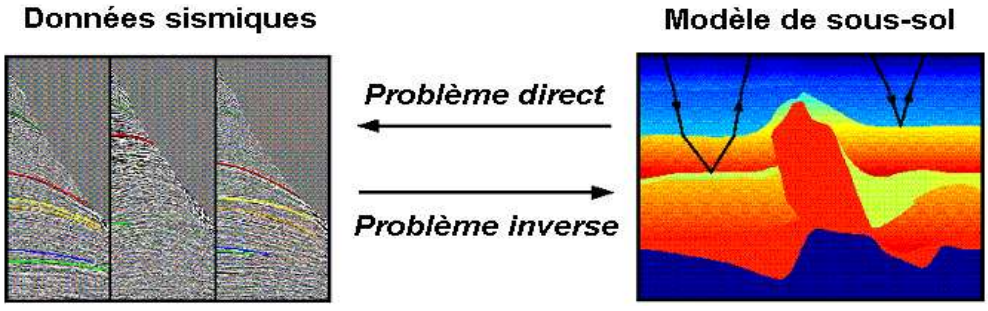

Fig. 2.3 Problème direct et problème inverse de la tomographie de réflexion (figure 2.9 de [21]) .

Remarques 2.2.2

1. *les données sismiques sont entachées d'erreurs de mesure :*
   - *des bruits non corrélés (pluie, vent, houle - en sismique marine, etc ...),*
   - *des bruits corrélés (arrivées multiples, modes convertis, ondes de surface, ondes diffractées, etc...).*
2. *les données sismiques sont incomplètes :*
   - *elles sont échantillonnées temporellement,*
   - *les récepteurs sont positionnés en un nombre fini de points de la surface,*
   - *dans le cas d'un milieu complexe, certaines zones géographiques du modèle peuvent ne pas être bien "illuminées" par le dispositif d'acquisition (les données sismiques contiennent peu d'informations interprétables sur ces zones).*
3. *les données sismiques sont entachées d'erreurs d'interprétation :*
   - *précision du pointé,*
   - *interprétation de temps de trajet érronés.*

On comprend donc que le problème (2.1) n'a en général pas de solution et que si elle existe elle n'est pas satisfaisante car elle explique à la fois le signal et le bruit contenus dans les données (voir section 1.1.2). Ainsi, au lieu de rechercher le modèle qui vérifie exactement (2.1) on va *"rechercher le modèle de vitesse du sous-sol qui satisfasse **au mieux** les données observées en surface"* (voir [51] pour une analyse des incertitudes sur le modèle solution).

Cela se traduit mathématiquement par la résolution du problème de moindres-carrés non linéaires suivant :

$$\min_{m\in\mathcal{M}} ||T(m) - T^{obs}||^2_{C_d^{-1}}, \tag{2.2}$$

où $||.||_{C_d^{-1}}$ est la norme dans l'espace des données. Si C_d est vu comme une matrice, alors on l'appelle matrice de covariance à priori des données et $||.||_{C_d^{-1}} = (.^\top C_d^{-1}.)^{1/2}$. Les éléments diagonaux de C_d sont les incertitudes sur les données et les éléments hors-diagonaux représentent les corrélations entre ces incertitudes.

Quelques remarques sur le problème (2.2) :

Remarques 2.2.3

1. *Le problème (2.2) est un problème mal posé car l'information discrète et bruitée apportée par les temps de trajet observés (voir remarque 2.2.2) est insuffisante pour déterminer les variations continues du champ de vitesse. Il faudra donc utiliser les techniques de régularisation pour rendre ce problème bien posé (voir section 1.1.3).*
2. *Comme nous avons vu dans la section 1.3, pour pouvoir résoudre le problème de moindres-carrés (2.2) par un algorithme d'optimisation efficace (par exemple Gauss-Newton ou Levenberg-Marquardt), il est important d'avoir accès à la jacobienne $J(m) = T'(m)$ de T, les dérivées premières des temps de trajet par rapport*

aux paramètres du modèle. On supposera donc par la suite que T est au moins C^1.

3. *Sachant que la fonction T est non linéaire et qu'il n'existe pas de solution analytique (sauf dans de rares situations), il faut utiliser une méthode numérique itérative pour trouver une solution du problème* (2.2).

4. *Pour des raisons de simplicité, en pratique, on suppose que les données ont toutes la même incertitude σ et qu'elles ne sont pas corrélées. Dans ce cas précis, la matrice de covariance a priori est alors égale à la matrice identité multipliée par une constante (égale à $1/\sigma^2$), et cela revient à utiliser une norme L^2 dans l'espace des données :*

$$\min ||T(m) - T^{obs}||_2^2.$$

Cependant, nous noterons qu'en tomographie de réflexion cette hypothèse n'est généralement pas valable car les résidus ($T(m) - T^{obs}$) sont spatialement dépendants (proximité des couples sources récepteurs associés aux temps de trajet observés). Dans ce cas, l'utilisation de la norme L^2 peut avoir de très mauvaises conséquences : la solution trouvée par l'inversion peut être très éloignée de la solution exacte. Pour plus de détails sur ce sujet, nous renvoyons le lecteur aux travaux de [132].

5. *La formulation par moindres-carrés de* (2.2) *est correcte si on suppose que les erreurs commises sur les données observées suivent une loi de probabilité normale. Les arguments principaux motivant l'utilisation d'une formulation par moindres-carrés sont :*

   (i) l'élimination de la valeur absolue pour comparer les données (pas de problèmes de différentiabilité si les résidus sont différentiables),

   (ii) la linéarité de sa dérivée lorsque les résidus sont linéaires.

   L'argument (ii) est très important car il aide à la minimisation de (2.2) *par des techniques d'optimisation efficaces. Notons que, dans le cas où les erreurs ne suivent pas une loi de probabilité normale, la formulation par moindres-carrés peut s'avérer très mauvaise (voir [2]).*

2.3 Description d'un macro-modèle de vitesse du sous-sol

Afin de pouvoir calculer une réponse synthétique $T(m)$ à partir d'un modèle m il faut que ce modèle soit défini non seulement par des paramètres de vitesse qui caractérisent la nature du milieu traversé mais aussi par des paramètres géométriques qui identifient la position des réflecteurs sismiques. Ainsi le vecteur des paramètres du modèle de sous-sol s'écrit :

$$m = \begin{pmatrix} v \\ z \end{pmatrix}, \tag{2.3}$$

où v est un vecteur décrivant la vitesse du sous-sol et z est un vecteur contenant la géométrie de chaque réflecteur.

Nous considérons des milieux tri-dimensionnels (tomographie 3D) du sous-sol définis par :

$$\begin{aligned} \Omega &= I_x \times I_y \times I_z \\ I_x &= [x_m, x_M] \\ I_y &= [y_m, y_M] \\ I_z &= [z_m, z_M], \end{aligned} \tag{2.4}$$

2.3.1 Choix du type de modélisation : lisse / par blocs

Pour pouvoir déterminer un modèle de sous-sol, il faut choisir une modélisation mathématique des variations spatiales de la vitesse en profondeur. A l'heure actuelle, deux types de paramétrisation différentes sont utilisés :

1. les modèles "lisses" (voir par exemple [15] ,[47] et [14]) qui se composent d'une distribution unique de la vitesse définie globalement sur Ω et d'un ou plusieurs réflecteurs. La vitesse et les réflecteurs sont supposés être au moins C^1 sur Ω.
2. les modèles "blocky" qui reposent sur une partition du milieu en blocs. A chaque bloc est associé une distribution de vitesse et les blocs sont délimités par des interfaces. Cette modélisation autorise des discontinuités de vitesse lors de la traversée d'une interface, cependant la vitesse est supposée au moins C^1 à l'intérieur de chaque bloc. Comme pour les modèles lisses, les interfaces sont supposées C^1. Les interfaces sont généralement des réflecteurs.

La régularité C^1 qui est demandée pour les fonctions représentant les vitesses de couches et les interfaces est nécessaire pour la résolution du problème direct et du problème inverse de la tomographie de réflexion. En effet, le calcul des dérivées premières des temps de trajet par rapport aux paramètres du modèle de vitesse nécessite au moins cette régularité (voir le point numéro 2. de la remarque 2.2.3).

Le grand avantage de la modélisation lisse est qu'elle assure la continuité et la stabilité du problème direct (voir [103]). Cependant, elle ne permet pas de déterminer un modèle de vitesse précis dans le cas de milieux complexes. En effet, une unique vitesse dans tout le domaine Ω ne permet pas de représenter de fortes variations latérales de vitesses (cas des structures faillées, salifères ou encore basaltiques).

A l'opposé, les modèles blocky sont particulièrement adaptés aux structures complexes car ils peuvent modéliser des variations rapides (discontinuités) de vitesse. La possibilité de modéliser des discontinuités de vitesses à la traversée des interfaces rend ces modèles plus réalistes que ne le sont les modèles lisses. L'hypothèse de base des modèles blocky est que les variations de vitesse à l'intérieur d'un bloc sont faibles, les hétérogénéités de vitesse étant modélisées aux interfaces. L'un des grands inconvénients de ces modèles est qu'ils n'assurent pas, en général, la stabilité du problème direct (c'est à dire, la définition, la continuité et la déribabilité de l'opérateur de modélisation physique T par rapport à de petites perturbations de modèle), ce qui peut perturber la convergence de l'algorithme d'optimisation (fonction coût différente d'une itération à l'autre de l'algorithme et perte de rayons).

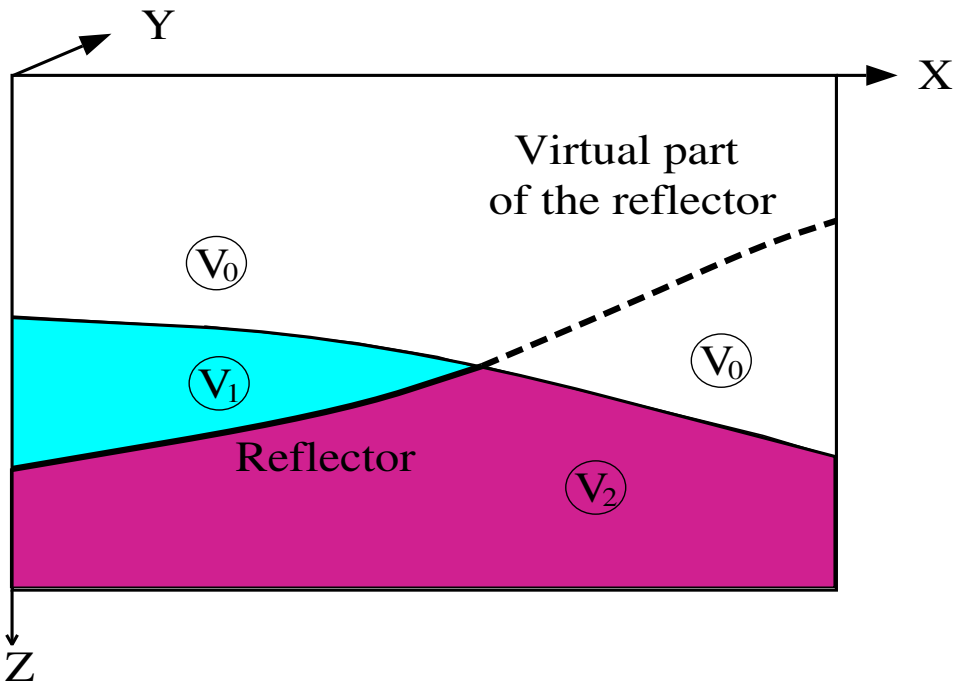

Fig. 2.4 Illustration d'un modèle blocky de sous-sol (figure 2.1 de [144]).

Dans le logiciel *jerry* développé à l'IFP une représentation blocky des modèles de sous-sol a été choisie. La figure 2.4 illustre un modèle de sous-sol modélisé par une représentation blocky. Pour plus de détails sur la mise en oeuvre de cette représentation nous renvoyons le lecteur à [32].

2.3.2 Discrétisation par des fonctions B-spline cubiques

Les variations de vitesse ainsi que les géométries des interfaces sont définies par des fonctions B-spline cubiques (voir annexe A, [10] et [44]), fonctions de régularité C^2. Cette régularité :

1. est une condition nécessaire (au moins C^1) pour assurer le bon fonctionnement de l'algorithme de tracé de rayon (voir [32] et la section 2.4). Plus précisement, afin de converger vers un rayon, la méthode de bending que nous utilisons dans notre code de tomography (voir section 2.4.2) nécessite de connaître les dérivées exactes des temps de trajet par rapport aux coordonnées spatiales des points d'impacts (voir [100]).
2. permet une grande flexibilité dans le choix de la régularisation du problème inverse (2.2) (voir section 2.5.1),
3. permet le calcul des éléments de la matrice jacobienne J de $T(m)$ nécessaire pour l'inversion (voir section 2.5.2).

On note N_Z (resp. N_V) le nombre d'interfaces (resp. le nombre de blocs de vitesse) décrivant le milieu. La représentation B-spline permet de décrire des interfaces explicites en x, y et z. Une interface 2D explicite en z s'exprime par un tenseur de fonctions B-spline cubiques :

$$Z_l(x,y) = \sum_{i=1}^{N_x^{Z_l}} \sum_{j=1}^{N_y^{Z_l}} Z_l^{ij} B_i(x) B_j(y), \tag{2.5}$$

où les Z_l^{ij} sont les $N_x^{Z_l} \times N_y^{Z_l}$ coefficients qui déterminent l'interface Z_l dans les directions x et y. B_i et B_j sont les fonctions de base des fonctions B-spline et sont fixées.

De même, on peut définir une vitesse 3D par :

$$V_l(x,y,z) = \sum_{i=1}^{N_x^{V_l}} \sum_{j=1}^{N_y^{V_l}} \sum_{k=1}^{N_z^{V_l}} V_l^{ijk} B_i(x) B_j(y) B_k(z), \tag{2.6}$$

où les V_l^{ijk} sont les $N_x^{V_l} \times N_y^{V_l} \times N_z^{V_l}$ coefficients qui déterminent la vitesse V_l dans les directions x, y et z. De même, les fonctions de base des fonctions B-splines B_i, B_j et B_k sont fixées.

2.4 Le problème direct

2.4.1 Le tracé de rayons : shooting / bending

L'objectif du problème direct de la tomographie de réflection est de calculer les temps de trajet des réflexions primaires sur les interfaces d'un modèle de vitesse du sous-sol. Rappellons que chaque temps de trajet observé (temps enregistré lors d'une campagne sismique) correspond à une position donnée de la source et du récepteur. Ainsi, calculer le temps de trajet d'une onde sismique pour aller de la source au récepteur (avec réflexion sur une interface) revient à déterminer la trajectoire qui passe par ces deux points et qui satisfasse au principe de Fermat : cela s'appelle *le tracé de rayons point à point* (ou "two point ray tracing"). Ce tracé de rayons représente une approximation haute fréquence de l'équation des ondes (pour plus de détails voir [95]). Cette approximation est acceptable si les longueurs d'onde des variations spatiales des vitesses sont "grandes" par rapport aux longueurs d'onde du signal sismique qui se propagent dans le milieu (voir [11]).

On peut identifier deux grandes classes de méthodes de tracé de rayons point à point :

1. les méthodes de "shooting" (voir les travaux [28], [105], [153], [152], [95] et [33]) : elles consistent à calculer un rai à partir d'une direction initiale à la source, puis à perturber cette direction jusqu'à ce que le rayon aboutisse au récepteur visé. Le terme "shooting" vient du fait que l'on "tire" un rayon à partir de la source.
2. les méthodes de "bending" (voir les travaux précurseurs [29] et [96], plus récemment [84], [147], [98]) reposent sur le principe de Fermat (1601-1665) que

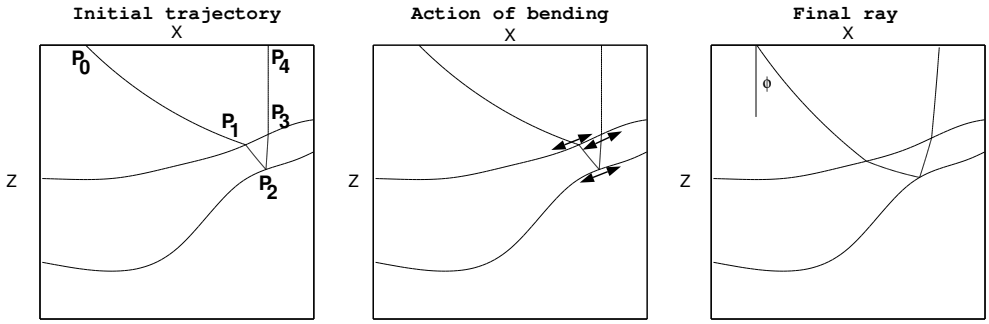

Fig. 2.5 Illustration de la méthode de bending (figure 2.2 de [144]).

voici :
Soit v la vitesse de la lumière dans un milieu transparent. L'intégrale curviligne

$$L = \int_A^B \frac{ds}{V}$$

calculée selon la trajectoire d'un rayon lumineux allant de A à B se nomme chemin optique. Le principe de Fermat énonce que le chemin optique des rayons lumineux est minimal[1].

Le principe de Fermat appliqué au tracé de rayons point à point consiste à poser que le temps de trajet d'un rayon r est stationnaire (dans la suite on appellera rayons les trajectoires qui vérifient le principe de Fermat). On recherche donc un rayon vérifiant l'équation :

$$\frac{dt}{dx}(r) = 0,$$

où t est le temps de trajet associé à la trajectoire géométrique x du rayon r concerné. Le bending est un problème d'optimisation qui consiste à déterminer parmi toutes les trajectoires partant de la source S, se réfléchissant sur l'interface i et arrivant au récepteur R celles dont le temps de trajet associé est stationnaire (voir figure 2.5). Comme dans ce travail on ne s'intéresse qu'aux arrivées premières, on recherche uniquement les trajectoires dont le temps de trajet est minimal.

Dans le logiel "jerry" de tomographie de réflexion, la méthode de bending a été préférée à une méthode de shooting car celle-ci est plus efficace en terme de temps de calcul (voir [98]). Cet argument prend encore plus de poids lorsque l'on sait que les applications en tomographie de réflexion 3D peuvent conduire au calcul d'une centaine de milliers de rayons (N_{obs} est souvent supérieur à 10^6). De plus, sachant que la méthode de résolution du problème inverse est itérative (voir le point numéro 3. de la remarque 2.2.3) le problème direct sera résolu pour plusieurs modèles (à chaque itération). On comprend

[1] Appliqué aux dioptres et aux miroirs, ce principe est équivalent aux lois de Snell-Descartes. Plus généralement, ce principe est la traduction pour l'optique géométrique des conditions aux limites de Maxwell sur un dioptre ou un miroir.

dès lors que le choix de la méthode de bending pour les inversions en tomographie de réflexion 3D est judicieux.

2.4.2 Une méthode de bending

Les points qui se trouvent à l'intersection de la trajectoire d'un rayon avec une interface s'appellent les points d'impact, noté P_i ci-dessous. La méthode de bending développée à l'IFP dans le logiciel *jerry* consiste à fragmenter la trajectoire d'un rayon en segments reliant les points d'impact. Chaque segment est défini par une courbe analytique (voir figure 2.5). Le premier (resp. dernier) segment relie la source (resp. le recepteur) à un point d'impact. Cette méthode de bending suppose que la trajectoire analytique reliant deux points d'impact est une droite[2] (voir [85] et [100]). Cette hypothèse est valable car les variations de vitesse à l'intérieur d'un bloc sont supposées faibles, les principales hétérogénéités étant modélisées par les interfaces. Le temps de trajet total le long de la trajectoire $(\mathcal{P}) = (P_0, ..., P_n)$ d'un rayon se définit par la formule :

$$t_{P_0 \to P_N}(\mathcal{P}) = \sum_{n=1}^{N} t^n(\mathcal{P}), \tag{2.7}$$

où t^n est le temps de trajet pour parcourir le segment (P_{n-1}, P_n) et N le nombre de segments définissant la trajectoire du rayon.

La méthode de bending développée à l'IFP consiste tout d'abord à initialiser une première trajectoire, voir figure 2.5. Puis, cette trajectoire initiale est ensuite optimisée en bougeant les points d'impact le long des interfaces pour satisfaire le principe de Fermat (i.e. le temps de trajet total le long de la trajectoire est stationnaire). Cela aboutit à résoudre le système non linéaire d'équations aux dérivées partielles suivant :

$$\frac{\partial t_{P_0 \to P_N}}{\partial P_n}(\mathcal{P}) = 0, \quad 1 \leq n \leq N-1. \tag{2.8}$$

Cette équation peut être simplifiée car un point d'impact P_n, $n \in [1, N-1]$, influe uniquement sur les temps de trajet partiels t_n et t_{n+1}. Le problème de bending consiste alors à rechercher la trajectoire $(\mathcal{P})$ vérifiant le système simplifié suivant :

$$\frac{\partial (t_n + t_{n+1})}{\partial P_n}(\mathcal{P}) = 0, \quad 1 \leq n \leq N-1. \tag{2.9}$$

Le choix de la représentation des interfaces en fonctions B-spline permet de calculer de façon exacte les coordonnées des points d'impact ainsi que les dérivées premières des temps de trajet par rapport aux coordonnées spatiales des points d'impact. L'accès à ces informations permet de résoudre le système non linéaire (2.9) par une méthode de quasi-Newton[3] (voir [108] et [69]).

[2] Dans le cas d'un gradient vertical de vitesse constant avec de faibles variations latérales, la trajectoire est supposée circulaire, voir [100].

[3] On peut aussi utiliser une méthode classique de Newton. Cependant les dérivées secondes des temps de trajet par rapport aux coordonnées spatiales des points d'impacts sont difficilement calculables

Comme la méthode de bending utilise une méthode d'optimisation locale pour résoudre (2.9) elle a les caractéristiques suivantes :

- elle ne peut trouver que les solutions rendant le temps de trajet total (voir équation (2.7)) localement minimal ce qui est problématique lorsqu'il existe plusieurs rayons vérifiant la même signature (cas des arrivées multiples et des points selles),
- la solution obtenue pour une signature donnée dépend de la trajectoire choisie pour initialiser l'algorithme de bending.

Rappelons-nous que dans le cadre de cette thèse nous nous limitons aux temps de trajet des arrivées premières. Il faut donc s'assurer que chaque rayon calculé par le bending correspond bien à une arrivée première. Une technique de continuation développée à l'IFP permet de bien initialiser l'algorithme de bending pour calculer uniquement les rayons associés à des arrivées premières (voir [99], [97] et [52]).

2.4.3 Caractéristiques du problème direct

Rappellons qu'à partir d'un modèle m et d'un dispositif d'acquisition (signature des données observées) le tracé de rayons consiste à calculer les temps de trajet $T(m)$. La fonction T définie par

$$\begin{array}{rcl} & & \mathcal{M} \to \mathbb{R} \\ T & : & m \mapsto T(m), \end{array}$$

est non-linéaire.

Les différentes caractéristiques du problème direct sont :

1. L'existence et l'unicité d'une solution au problème de tracé de rayons point à point ne sont pas assurées. Nous donnons ci-dessous deux cas pathologiques fréquemment rencontrés en tomographie de réflexion :

   (i) le cas pathologique de la figure 2.6 (cuvette circulaire) montre qu'il existe une infinité de rayons issus de la source S se réfléchissant sur l'interface et revenant en S. Par contre, il n'existe aucun rayon revenant en un recepteur R différent de S,

   (ii) dans le cas d'un modèle présentant des interfaces avec des pentes importantes l'algorithme de bending peut ne pas réussir à trouver une trajectoire qui respecte le principe de Fermat : de nombreux rayons peuvent être "perdus", i.e., ils sortent du modèle avant de rencontrer la surface et ne peuvent donc pas aboutir sur un récepteur.

2. Chaque calcul d'un temps de trajet $T(m)_k$ nécéssite la résolution d'un problème d'optimisation non-linéaire (voir équation (2.9)).

3. Compte tenu du nombre important de données à inverser (N_{obs} peut-être supérieur à 10^6), la résolution du problème direct coûte cher en temps CPU.

Nous supposons dans la suite que l'opérateur T est univoque et différentiable. Cepen-

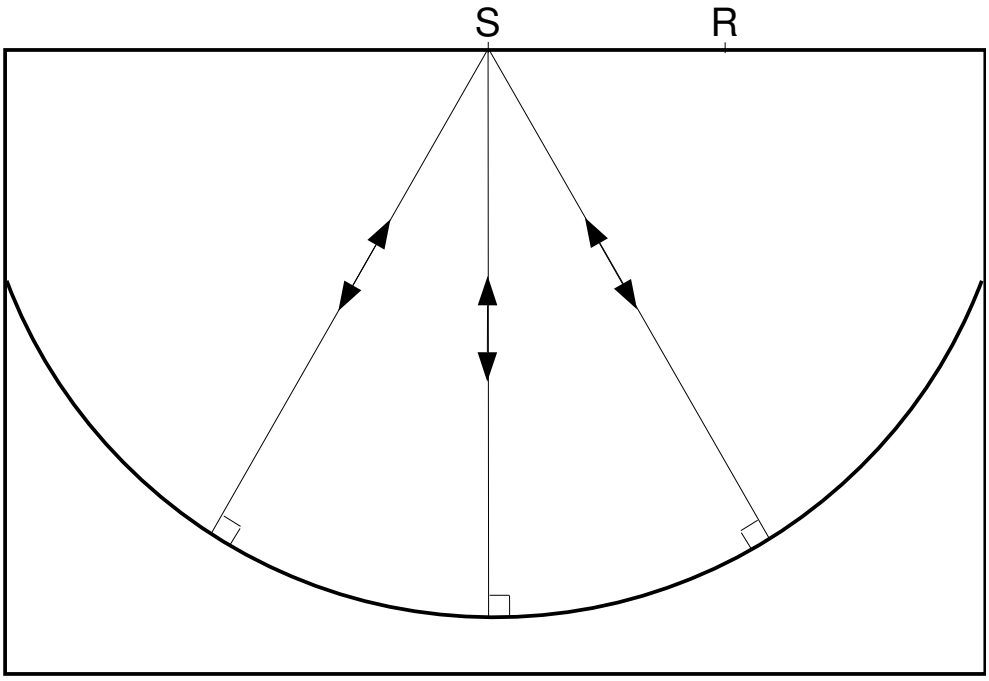

Fig. 2.6 Illustration d'un cas pathologique (cuvette circulaire) du tracé de rayons point à point où l'existence et l'unicité d'une solution ne sont pas assurée : le problème admet une infinité de solution de signature (S,S,i) mais n'admet pas de solution de signature (S,R,i) (figure reprise de [140]).

dant, nous avons vu qu'en pratique cette hypothèse peut ne pas être vérifiée (voir le point 1. de la remarque précédente).

2.5 Le problème inverse

2.5.1 Régularisation du problème inverse

Rappelons que le problème inverse de la tomographie de réflexion consiste à *"rechercher le modèle de sous-sol m tel que les temps de trajet calculés ($T(m)$) par le tracé de rayons (voir section précedente) s'ajustent au mieux avec les temps de trajet observés (T^{obs})."*

Nous avons vu à la section 2.2 (voir l'équation (2.2)) que cela revient à résoudre le problème de moindres-carrés non-linéaires suivant (pour simplifier les notations, nous avons choisi la norme L^2 dans l'espace des données, voir le point 4. de la remarque 2.2.3) :

$$\min_{m \in \mathcal{M}} ||T(m) - T^{obs}||_2^2. \tag{2.10}$$

De nombreux auteurs (voir [8], [111], [141] et le chapitre 1 de [47]) ont mis en évidence le caractère mal posé (voir définition 1.1.1) de ce problème originel de la tomographie de réflexion . Comme nous avons vu dans la section 1.1.2, l'une des techniques essentielles

permettant de résoudre les problèmes mal posés consiste en une étape de reformulation du problème : la régularisation des problèmes inverses (voir section 1.1.3). Les premiers essais de résolution du problème (2.10) (voir les travaux de [138] et [125]) ont été l'utilisation de la méthode de Levenberg-Marquardt (voir section 1.3.4 pour plus de détails sur cette méthode). L'approche de Levenberg-Marquardt s'est révélée inadaptée car d'une part la solution obtenue par cette méthode dépend fortement de la discrétisation choisie et d'autre part elle n'a pas de sens physique (voir le chapitre 3 de [47]).

Une régularisation convenable du problème de tomographie de réflexion originel, initialement proposée dans [38], consiste à rajouter dans la fonction coût de (2.10) un terme contenant la norme des dérivées secondes des vitesses de couche ainsi que des interfaces. Le problème inverse se reformule alors par :

$$\min_{m\in\mathcal{M}} \left(\frac{1}{2} \left|\left| T(m) - T^{obs} \right|\right|_2^2 + \frac{\sigma^2}{2}(m - m^{prior})^\top R(m - m^{prior}) \right), \tag{2.11}$$

où m^{prior} est un modèle à priori, $\sigma > 0$ est le poids de régularisation. Notons que, pour simplifier les notations, on ne considère dans la suite qu'un seul poids de régularisation, même si, en pratique, il est possible d'affecter un poids différent à chaque vitesse et chaque interface. L'opérateur R est un opérateur de régularisation sur les dérivées secondes défini par :

$$\begin{array}{rcl} m^\top R m & = & \sum_{l=1}^{N_V} \int_\Omega [D^2(V_l(x,y,z))]^2 dxdydz \\ & + & \sum_{l=1}^{N_I} \int_\Omega [D^2(Z_l(x,y,z))]^2 dxdy, \end{array}$$

où $m = (V_l, Z_l)$ et avec (pour une fonction g tri-dimensionnelle suffisamment régulière)

$$\begin{array}{rcl} [D^2(g(x,y,z))]^2 & = & \left[\frac{\partial^2 g}{\partial x^2}\right]^2 + \left[\frac{\partial^2 g}{\partial y^2}\right]^2 + \left[\frac{\partial^2 g}{\partial z^2}\right]^2 \\ & + & \left[\frac{\partial^2 g}{\partial x \partial y}\right]^2 + \left[\frac{\partial^2 g}{\partial x \partial z}\right]^2 + \left[\frac{\partial^2 g}{\partial y \partial z}\right]^2 . \end{array}$$

En général, en tomographie de réflexion il est souvent très difficile d'avoir accès à un modèle solution à priori. Ainsi, pour simplifier les notations, on supposera dans la suite de ce travail que le modèle m^{prior} est défini par des distributions de vitesse affines et des géométries d'interface affines. Le problème général (2.11) se simplifie alors par :

$$(P_{tomo}) : \quad \min_{m\in\mathcal{M}} \left(f(m) := \frac{1}{2} \left|\left| T(m) - T^{obs} \right|\right|_2^2 + \frac{\sigma^2}{2} m^\top R m \right), \tag{2.12}$$

où f est la fonction côut du problème de tomographie de réflexion régularisé. Afin de résoudre le problème (P_{tomo}) nous pouvons nous appuyer sur les résultats théoriques de la section 1.3 sur les problèmes de moindres-carrés non-linéaires. Dans la section suivante, nous verrons que le calcul de la matrice jacobienne des temps de trajet est facile à effectuer grâce au principe de Fermat et compte tenu du choix de la paramétrisation par des fonctions B-spline cubiques. Dans la section 2.5.3 nous motiverons le choix de la méthode de Gauss-Newton pour résoudre le problème (2.12). De plus, nous développerons cette méthode en insistant sur les particularités de son application au problème inverse de la tomographie de réflexion. Enfin, dans la section 2.5.4 nous discuterons de l'approche choisie dans le logiciel *jerry* pour fixer le poids de pénalisation σ (voir section théorique 1.1.4).

2.5.2 La matrice jacobienne des temps de trajet

La matrice jacobienne $J(m)$ de T en m s'exprime par :

$$J(m) = \frac{dT}{dm},$$

où l'élément (i, j) de cette matrice s'obtient par la dérivée partielle

$$J(m)_{ij} = \frac{\partial T_i}{\partial m_j}. \tag{2.13}$$

Comme nous avons vu dans la section 2.4, l'utilisation de la méthode de bending pour résoudre le problème direct explique qu'à chaque temps de trajet T_i est associé une trajectoire $(\mathcal{P}) = (P_0^i, ..., P_N^i)$ qui respecte le principe de Fermat. Ainsi, si on injecte la formule

$$T_i = t^i_{P_0 \to P_N}(\mathcal{P})$$

dans l'équation (2.13), on obtient :

$$J(m)_{ij} = \frac{\partial t^i_{P_0 \to P_N}}{\partial m_j}(\mathcal{P}) = \sum_{n=1}^{N-1} \left[\frac{\partial (t_n^i + t_{n+1}^i)}{\partial P_n^i} \frac{\partial P_n^i}{\partial m_j} + \frac{\partial (t_n^i + t_{n+1}^i)}{\partial m_j} \right]. \tag{2.14}$$

Le respect du principe de Fermat pour la trajectoire $(\mathcal{P})$ implique que l'équation (2.9) est satisfaite. En injectant l'équation (2.9) dans (2.14), on aboutit à la formule suivante :

$$J(m)_{ij} = \sum_{n=1}^{N-1} \frac{\partial (t_n^i + t_{n+1}^i)}{\partial m_j}. \tag{2.15}$$

Comme les interfaces et les vitesses du modèle m sont définies par des fonctions B-spline cubiques, les éléments $J(m)_{ij}$ de la matrice jacobienne peuvent être calculés explicitement (voir [97] pour plus de détails sur ces calculs).

Les conséquences importantes de l'équation (2.15) sont :

1. une fois que les temps de trajet T_i ont été calculés par la méthode de bending, le calcul explicite des éléments de la matrice jacobienne ne coûte pas cher en temps CPU (pas de nouveau rayon à calculer),
2. la matrice jacobienne J est creuse ; en effet, chaque temps de trajet $(t_n^i + t_{n+1}^i)$ s'exprime comme une somme de fonctions B-spline pondérée par des paramètres B-spline ; et, comme les fonctions B-spline sont nulles partout sauf sur quatres intervalles de noeuds (voir annexe A) les temps de trajet $(t_n^i + t_{n+1}^i)$ ne dépendent que d'un nombre restreint de paramètres B-spline.

2.5.3 Méthode de Gauss-Newton en tomographie de réflexion

Dans nos applications en tomographie de réflexion, on remarque que :

1. les résidus des temps de trajet sont relativement faibles en la solution (inférieur à 10 milli-secondes),
2. la matrice jacobienne J est facilement accessible (voir section précédente).

Ainsi, d'après les arguments développés dans la section 1.3, la méthode de Gauss-Newton est bien adaptée pour résoudre le problème (2.12). Dans la suite, nous appliquons spécifiquement au problème (2.12) la méthode de Gauss-Newton présentée dans la section 1.3.3 du chapitre théorique.

On obtient le problème inverse linéarisé (voir définition 1.3.9) en linéarisant la fonction T autour du modèle m_k courant :

$$T(m_k + \delta m) \approx T_k + J_k \delta m,$$

où $T_k := T(m_k)$, $J_k = J(m_k)$, et $\delta m = m - m_k$ représente la perturbation de modèle. En injectant cette approximation linéaire de T dans l'équation 2.12, on obtient le *problème linéarisé* en m_k suivant :

$$\begin{array}{rlll} (P_{lin}): & \min\limits_{\delta m} \Big(F_k(m) & := & \frac{1}{2} \left|\left| T_k + J_k \delta m - T^{obs} \right|\right|_2^2 \\ & & + & \frac{\sigma^2}{2} (m_k + \delta m)^\top R (m_k + \delta m) \Big). \end{array} \tag{2.16}$$

A part quelques cas pathologiques (voir section 2.4) le problème linéarisé (2.16) est bien posé (pour plus de détails sur ce sujet, nous renvoyons le lecteur aux travaux de [48] et au chapitre 3 de [47]). Une conséquence importante de ce résultat est que la solution de (2.16) ne dépend pas de la discrétisation choisie sur le modèle (à partir d'un certain seuil sur le pas de discrétisaton).

La fonction F_k est une approximation quadratique en m_k de la fonction coût non-linéaire f, on peut l'écrire sous la forme standard suivante :

$$F_k(\delta m) = \frac{1}{2} \delta m^\top H_k \delta m + g_k^\top \delta m + f_k,$$

où $f_k = f(m_k)$,

$$g_k = J_k^\top (T_k - T^{obs}) + \sigma^2 R m_k$$

est le gradient de f en m_k et

$$H_k = J_k^\top J_k + \sigma^2 R$$

est une approximation semi définie positive du hessien de f (en général définie positive, notons que le noyau de R est caractérisé par l'ensemble des modèle "plat") obtenue en négligeant les dérivées secondes des temps de trajet.

Dans la figure 2.7 nous pouvons observer le spectre de la matrice H_k associé à un modèle relativement simple de sous-sol (une seule vitesse et une seule interface). Il ressort de cette figure que l'ajout de la partie régularisation ($\sigma^2 R$) à $J_k^\top J_k$ est crucial pour rendre la matrice H_k définie positive. De plus on observe que le conditionnement de H_k est très mauvais ($\approx 10^9$) pour une matrice d'ordre seulement 500.

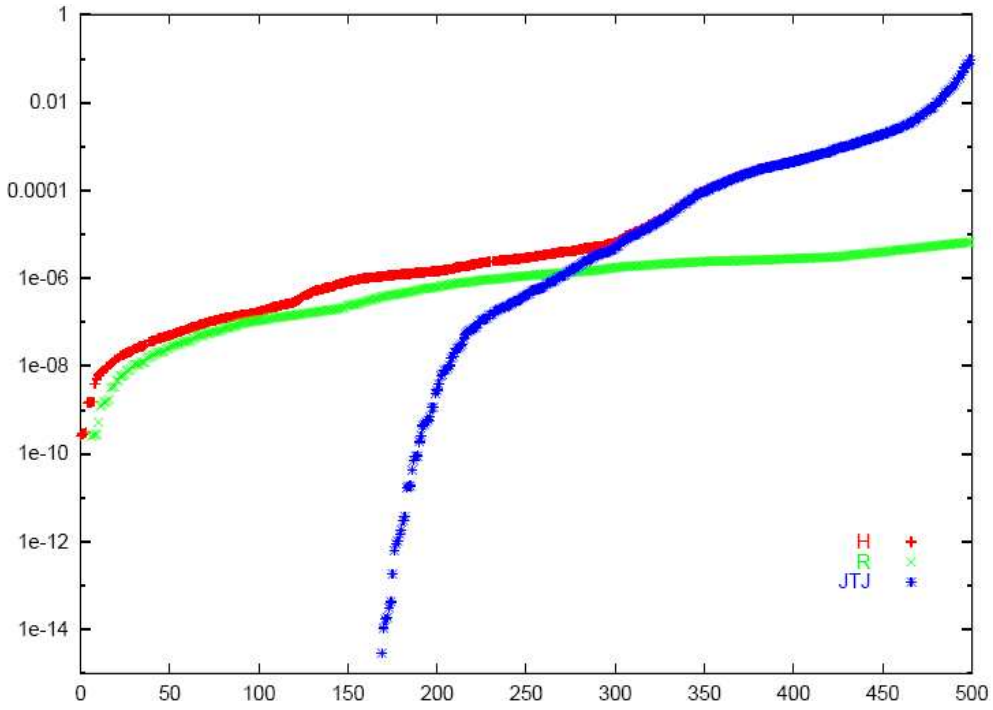

Fig. 2.7 Spectre de H (en rouge), de $J^\top J$ (en bleu) et de R (en vert) pour exemple simple : le modèle est composé d'une seule vitesse (discrétisée par 400 paramètres B-spline) et d'une seule interface (discrétisée par 100 paramètres B-spline). Cette figure est reprise de la figure 1 de [30].

L'équation d'optimalié de (2.16) (voir section 1.3.1) s'écrit :

$$H_k \delta m = -g_k. \tag{2.17}$$

Comme H_k est définie positive, le problème de moindres-carrés linéaires (2.16) peut être résolu en utilisant l'algorithme du gradient conjugué sur l'équation normale (2.17) (voir l'algorithme CGLS de la section 1.3.2). Cet algorithme est d'autant plus efficace (notamment sur des problèmes de grande taille mal conditionnés) qu'il est couplé à un préconditionneur adapté à la structure de H_k. Dans la figure 2.8, nous pouvons observer les éléments non nuls de la matrice hessienne pour un modèle complexe de sous-sol : la structure de H_k est délimitée par des blocs (vitesses / interface) et les éléments non nuls à l'intérieur d'un même bloc sont disposés en bandes. Nous pouvons aussi remarquer sur cette figure que la matrice H_k est très creuse.

L'algorithme de Gauss-Newton appliqué à la résolution de 2.12 s'écrit (voir l'algorithme théorique I.2) :

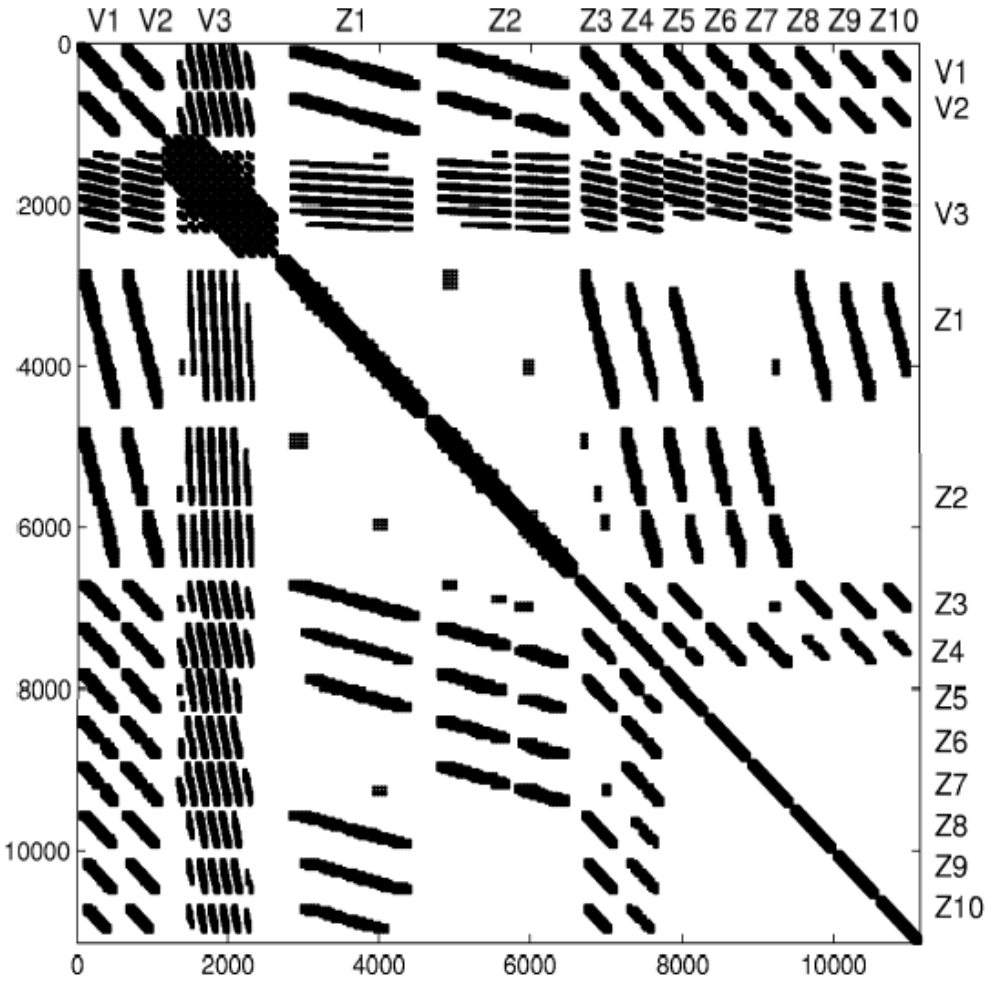

Fig. 2.8 Eléments non nuls de la matrice hessienne $H(m)$ pour un exemple complexe : le modèle est composé de 3 vitesses (discrétisées par 2634 paramètres B-spline) et de 10 interfaces (discrétisées par 8514 paramètres B-spline). Cette figure est reprise de la figure 2 de [30].

Data : Choix d'un modèle initial m_0.
Choix d'un poids de pénalisation : $\sigma > 0$.
Constante $0 < \omega_1 < \frac{1}{2}$ (typiquement $\omega_1 = 10^{-4}$).

begin

$k = 0$

Évaluer f_0, g_0, J_0 par la résolution du problème direct (algorithme de tracé de rayons).

while $||g_k||_2 \neq 0$ **do**

((1)) Calcul de la direction de descente : prendre pour δm_k une solution de (2.17).

((2)) Déterminer le pas α_k par "rebroussement", i.e., en prenant le plus grand α_k dans $\{1, \frac{1}{2}, \frac{1}{4}, ...\}$ tel que

$$f(x_k + \alpha_k \delta m_k) \leq f(x_k) + \omega_1 \alpha_k (g_k, \delta m_k).$$

((3)) Mettre à jour le modèle :

$$m_{k+1} = m_k + \alpha_k \delta m_k.$$

((4)) Évaluer f_{k+1}, g_{k+1}, J_{k+1} par la résolution du problème direct (algorithme de tracé de rayons).

((5)) Accroître k de 1 : $k := k + 1$.

endw

end

Algorithme I.4 Algorithme de Gauss-Newton en tomographie de réflexion

Quelques commentaires sur cet algorithme :

Remarques 2.5.1

1. *Les commentaires de l'algorithme général I.2 s'appliquent à cet algorithme.*
2. *L'étape ((1)) de l'algorithme est importante : il faut résoudre rapidement l'équation normale* (2.17) *pour que l'algorithme de Gauss-Newton soit efficace. Etant donné la dimension des systèmes à résoudre dans nos applications, une méthode itérative doit être préférée à une méthode directe pour résoudre le système* (2.17) *: c'est un algorithme du gradient conjugué préconditionné, adapté de l'algorithme CGLS (voir section 1.3.2), qui a été choisi pour le résoudre. Les travaux de [30] ont montré que :*

   (i) l'utilisation de préconditionneurs Jacobi ou Gauss-Seidel par blocs permet d'accélérer la convergence de l'algorithme du gradient conjugué et d'obtenir des solutions plus précises de (2.17).

   (ii) l'algorithme du gradient conjugué peut être développé de manière à ne pas avoir à stocker H_k et effectuer à chaque itération de Gauss-Newton le calcul

peu précis et coûteux des éléments de H_k (voir le point 1. de la remarque 1.3.6). Les produits matrices vecteurs sont alors réalisés grâce au stockage optimisé des matrices J_k (stockage morse) et R (stockage bande).

3. *Remarquons que la solution obtenue par l'algorithme I.4 est fortement dépendante du choix initial du poids de pénalisation σ. Dans la section suivante nous verrons comment nous choisissons ce poids en pratique.*

4. *En théorie, l'arrêt des itérations de Gauss-Newton doit s'effectuer lorsque la norme du gradient g_k est plus petite qu'un certain seuil. Cependant, ce critère n'est pas suffisant car le seuil qui lui est associé est difficile à évaluer. En pratique, l'utilisateur dispose de trois critères permettant d'arrêter l'algorithme de Gauss-Newton. Ces critères, plus proches des préoccupations de la géophysique, comparent les temps de trajet calculés aux temps de trajet observés. On distingue les critères suivants :*

   (i) le résidu maximal :

   $$||T(m) - T^{obs}||_\infty,$$

   où N_{obs} représente le nombre de temps de trajet observés,

   (ii) le résidu moyen (ou rms pour "root mean square") :

   $$rms = \sqrt{\frac{\sum_{i=1}^{NT}(T_i(m) - T_i^{obs})^2}{NT}},$$

   (iii) la distribution des résidus (histogrammes des résidus, répartition des résidus en fonction des points d'impact, etc ...).

   L'analyse comparative de ces trois critères permet de déterminer si la solution d'une inversion est acceptable. Dans le cas d'une solution non acceptable il faut soit, prolonger les itérations de Gauss-Newton, soit relancer une inversion en partant d'un poids de régularisation plus faible.

En pratique, dans nos applications en tomographie de réflexion, l'algorithme I.4 est très efficace : il permet de trouver une solution du problème (2.15) en très peu d'itérations de Gauss-Newton (de l'ordre de 5-10 itérations). Cependant, dans certains cas difficiles, la recherche linéaire (voir l'étape ((2)) de l'algorithme I.4) ne parvient pas à forcer la convergence de la suite $\{m_k\}_{k\geq 0}$ vers une solution du problème (2.15). Cette difficulté, qui apparaît notamment lorsque la matrice H_k est très mal conditionnée, peut être contournée en utilisant une globalisation par régions de confiance (voir [35]) au lieu de la recherche linéaire. Notons que cela revient en fait à remplacer l'algorithme de Gauss-Newton précédent par un algorithme proche de celui de Levenberg-Marquardt (voir l'algorithme I.3 du chapitre théorique). Dans le chapitre suivant, nous verrons que l'utilisation en tomographie de réflexion de l'algorithme de Gauss-Nexton globalisé par des régions de confiance permet de résoudre des cas où l'algorithme I.4 a échoué.

2.5.4 Réglage du poids de régularisation

En tomographie de réflexion le réglage du poids de régularisation σ est très délicat à effectuer car :

1. si le poids de régularisation est trop grand on ne va pas minimiser suffisamment les résidus des temps de trajet,
2. si le poids de régularisation est trop petit, l'inversion devient instable : (2.12) se rapproche alors de (2.10) et le système linéaire (2.17) devient très mal conditionné.

Afin de remédier à ces problèmes, une méthode de courbe en L (plus connue en tomographie sous le nom de méthode de "continuation", voir [22]) a été développée dans le logiciel *jerry* de tomographie de réflexion. Cette méthode (voir section théorique 1.1.4) consiste tout d'abord à obtenir une première solution de l'inversion en partant d'un "grand" poids de régularisation. Puis en initialisant l'inversion avec ce modèle, on obtient un nouveau modèle pour un poids de régularisation plus faible. On renouvelle cette opération. Le critère de la courbe en L permet de détecter le poids de régularisation limite au-dessous duquel aucune amélioration sensible du résidu ne sera obtenue et l'inversion devient instable. Cette méthode est très efficace en tomographie de réflexion (voir l'étude détaillée de l'utilisation du critère de la courbe en L en tomographie de réflexion dans [34]) : la diminution régulière du poids de pénalisation permet de trouver des modèles de moins en moins "lisses" dont les temps de trajet calculés se rapprochent de plus en plus des temps de trajet observés.

Chapitre 3

Une méthode de régions de confiance en tomographie de réflexion

L'article qui va suivre a été publié en 2001 dans le rapport annuel de recherche du consortium KIM (Kinematic Inversion Methods, voir [46]). Il montre que l'on peut résoudre le problème inverse de la tomographie de réflexion (voir équation (2.12)) en utilisant une méthode de Gauss-Newton globalisée par régions de confiance (voir algorithme I.3). Le modèle quadratique de la fonction coût non-linéaire est alors minimisé par un algorithme de Gradient-Conjugué tronqué (voir [142]). Les différents paramètres de la méthode des régions de confiance (par exemple le rayon de la région de confiance) sont automatiquement réglés par le solveur. Sur certains exemples, cette méthode donne des résultats plus stables et plus précis que la méthode initiale de Gauss-Newton globalisée par recherche linéaire (voir section 2.5.3 et algorithme I.4). Elle permet notamment de mieux résoudre les exemples complexes où le Hessien est très mal conditionné. Cet article propose et teste aussi de nouveaux critères d'arrêt et un nouveau préconditionnement pour l'algorithme du gradient conjugué.

Trust-region Gauss-Newton method for reflection tomography

F. Delbos*, D. Sinoquet*, J. Ch. Gilbert †, and R. Masson*

ABSTRACT

The goal of reflection tomography is to determine a subsurface velocity model that fits observed travel-times associated with reflections on interfaces. The tomographic inverse problem consists in solving a large scale non-linear minimization problem. A line search Gauss-Newton (GN) method is used in the software *jerry* to solve this optimization problem. At each GN step, an (approximate) solution of a quadratic model of the objective function is computed using a conjugate gradient (CG) algorithm. This method generally allows to solve the minimization problem. However, in some cases we have noticed that the line search method is not able to find a solution or to decrease significantly the objective function. This difficulty is likely to be caused by a too ill-conditioned Hessian matrix. To overcome these difficulties a trust-region GN method coupled with a truncated CG algorithm has been implemented. In this paper we introduce and test the performances of the trust-region GN method in unconstrained reflection tomography. Furthermore, we discuss the choice of the initial trust-region radius, the CG stopping criteria and the preconditioning.

INTRODUCTION

Let us first recall the problem of interest and introduce some notation. We consider discrete models that describe the layer velocities and the interface geometries in complex subsurface structures. The i^{th} interface is represented by a cubic B-spline function $\widehat{z}_i(x,y)$, whose coefficients define a vector z_i. Similarly, the i^{th} velocity field is represented by a cubic B-spline function $\widehat{v}_i(x,y,z)$ or $\widehat{v}_i(x,y) + k\,z$ with known k ; the vector v_i contains the velocity coefficients. For n_v layer velocities and n_z interfaces, we gather the coefficients v_1, ... , v_{n_v} in one vector v and the coefficients z_1, ... , z_{n_z} in one vector z. The model vector m is defined as $m^\top = (v^\top, z^\top)$.

Given a model m, a vector of traveltimes $T(m)$ of seismic reflections can be computed by ray-tracing ; this is referred to as the forward problem. This forward problem is known to be non-linear

* Institut Français du Pétrole, 1 & 4 avenue de Bois-Préau, 92852 Rueil-Malmaison, France.
† Institut National de la Recherche en Informatique et en Automatique, Domaine de Voluceau - Rocquencourt - B.P.105 - 78153 Le Chesnay Cedex France

but we can assume that the mapping from m to T is smooth. Reflection traveltime tomography is the corresponding inverse problem : its purpose is to adjust m so that $T(m)$ best matches a vector of traveltimes T^{obs} picked on seismic data. A natural formulation of this problem is the least squares formulation :

$$\text{minimize} \quad \tfrac{1}{2} \left|\left| T(m) - T^{obs} \right|\right|_2^2 . \tag{1}$$

where we have chosen to use the Euclidean norm $||\,.\,||_2$ (see Renard and Lailly, 2001 for an explanation of this choice).

The fact that problem (1) may be ill-posed has been pointed out by many authors. Indeed, its solution is generally not unique. To ensure well-posedness, curvature regularization is often introduced, e.g. the sum of the squared L^2-norms of all the second-order partial derivatives of every velocity $\widehat{v}_i$ and reflector $\widehat{z}_i$ (see for instance Delprat-Jannaud and Lailly, 1993). Such a regularization can be written as $m^\top R\, m$, where R is a symmetric semidefinite positive matrix that only depends on the B-spline basis functions, i.e. that is independent of m. Thus, instead of problem (1), we consider the regularized least squares problem

$$\text{minimize} \quad f(m) = \tfrac{1}{2} \left|\left| T(m) - T^{obs} \right|\right|_2^2 + \tfrac{\sigma^2}{2}\, m^\top R\, m, \tag{2}$$

where the regularization weight σ is chosen strictly positive, and $f(m)$ is called the non-linear function (or non-linear objective function).

Since $T(m)$ is a non-linear function, the resolution of the non-linear minimization problem is carried out by an iterative method. The principle of the Gauss-Newton method is to find an approximate solution of (2) by solving a sequence of linearized problems (GN steps).

The linearization of T around a current model m_k is written as

$$T(m_k + \delta m) = T_k + J_k\, \delta m + o(\delta m), \tag{3}$$

where $T_k = T(m_k)$ are the calculated travel-times in the current model m_k and $J_k = J(m_k)$ is the Jacobian of T with respect to m_k. Let us now consider the quadratic model (also called the quadratic objective function) $F_k(\delta m)$ of $f(m_k + \delta m)$ obtained from equation (3) :

$$F_k(\delta m) = \tfrac{1}{2} \left|\left| J_k\, \delta m + T_k - T^{obs} \right|\right|_2^2 + \tfrac{\sigma^2}{2}\, (m_k + \delta m)^\top R\, (m_k + \delta m), \tag{4}$$

where R is the regularization matrix. Equation (4) is equivalent to

$$F_k(\delta m) = \tfrac{1}{2}\, \delta m^\top H_k\, \delta m + g_k^\top \delta m + f_k, \tag{5}$$

where $f_k = f(m_k)$,

$$g_k = J_k^\top (T_k - T^{obs}) + \sigma^2\, R\, m_k$$

is the gradient of f in m_k and

$$H_k = J_k^\top J_k + \sigma^2\, R$$

is an approximation of its Hessian (we neglect the term involving the second-order derivatives of the travel-times).

LINE SEARCH METHOD

In the classical line search method, H_k is assumed to be positive definite thanks to the choice of the regularization matrix R (see Delprat-Jannaud and Lailly, 1992). The model perturbation δm

that minimizes the quadratic objective function $F_k(\delta m)$ is the line search direction. It is approximately obtained using a CG algorithm. At the end of the GN step, the model is updated according to

$$m_{k+1} = m_k + \lambda_k.\delta m, \tag{6}$$

where the positive scalar λ_k is called the step length. This step length is choosen in agreement with the Armijo condition, ensuring a sufficient decrease in the non-linear objective function :

$$f(m_k + \lambda_k.\delta m) \leq f(m_k) + C.\lambda_k.\nabla f_k^T \delta m \tag{7}$$

where $\nabla f_k^T \delta m$ is the directional derivates in m_k along δm, and C is a constant with $C \in (0, 1)$ (a usual choice is $C = 10^{-4}$). The Armijo condition (7) is coupled with a backtracking line search algorithm (described in Appendix C) that allows to eventually reduce the value of λ_k until condition (7) is satisfied. A step where condition (7) can not be satisfied is called a GN failure (or a GN rejected step).

We propose here an alternative method : the trust-region approach that allows us to control more accurately the model perturbation and also allows for non-positive definite Hessian approximation H_k. The motivations for implementing a trust-region inversion method (see Nocedal and Wright, 1999 and Conn et al., 2000) are twofold. First, in the line search method the model perturbation is controlled by the threshold convergence of the CG algorithm. This threshold is difficult to choose accordingly to the decrease of the non-linear objective function (see the section on CG termination criteria), and we hope that the trust-region method will better control the model perturbation. Secondly, when examples with ill-conditioned Hessian are considered (the examples 2 and 3 in the following section), the line search method is quite unstable and GN failures appear. Each failure requires the subsequent calculation of the non-linear objective function for the new model estimate and is thus expensive in CPU time. In our most unstable model (example 3), the Hessian matrix is so ill-conditioned that the line search method (with or without preconditioning) fails to retrieve the model. These troubles underline the necessity to implement another inversion method, and we expect the trust-region method to solve these difficulties.

TRUST-REGION METHOD

Contrary to line search methods, trust-region methods use the quadratic model of the objective function to choose simultaneously the direction and the size of the model perturbation δm (in (6) $\lambda_k = 1$ for trust-region methods). The main advantage of the trust-region method is its ability to control the norm of the model perturbation : the quadratic model is minimized in a region around the current model.

In other words, at each GN step, the following problem is (approximately) solved :

$$\text{minimize} \quad F_k(\delta m) \quad s.t. \mid\mid \delta m \mid\mid \leq \Delta_k \tag{8}$$

where Δ_k is the trust-region radius. For the perturbation δm found, we measure the concordance between the quadratic model and the objective function, defined as

$$\rho_k = \frac{f(m_k) - f(m_k + \delta m)}{F_k(0) - F_k(\delta m)}, \tag{9}$$

The numerator is called the actual reduction and the denominator is the predicted reduction of the non-linear objective function. If the value of ρ_k is higher than a choosen threshold (a usual choice

is 0,75), the model perturbation is accepted, the new model is calculated as $m_{k+1} = m_k + \delta m_k$ and the trust-region radius is updated (increased or decreased) for the next step depending on the value of ρ_k (see Appendix A for more details on the automatic update procedure). If the value of ρ_k is negative or close to zero, the model perturbation has to be rejected, Δ_k is decreased and another minimum in this smaller trust-region is calculated (similarly to the line search method this case is called a GN rejected step or a GN failure). Note that this case occurs rather seldom thanks to the automatic choice of the trust-region radius.

The trust-region method is intimately related to a widely used inversion method in geophysics : the Levenberg-Marquardt technique. Without regularization term, the Hessian matrix associated with the tomographic problem is not positive definite. The Levenberg-Marquardt method (see Levenberg, 1944 and Marquardt, 1963) consists in adding a diagonal matrix $\mu_c I$ to the Hessian matrix at each GN step :

$$[H_k + \mu_c I]\delta m = -g_k. \tag{10}$$

It has been shown by Sebudandi and Toint, 1993 that this method effectively overcomes the difficulties encountered in the unregularized inverse problem. The choice of a suitable parameter $\mu_c > 0$ remains an important problem of the Levenberg-Marquardt method. The resolution of problem (10) is equivalent to the resolution of the following trust-region minimization problem (see More, 1978) :

$$\begin{cases} \text{minimize } F_k(\delta m) \quad s.t \ \ || \, \delta m \, || \leq \Delta_k^c, \\ \Delta_k^c = \left|\left| \, (H_k + \mu_c I)^{-1} g_k \, \right|\right| \end{cases} \tag{11}$$

where μ_c is interpreted as a Lagrangian multiplier associated with the inequality constraint (the trust-region constraint). Then, the trust-region method allows an elegant and efficient choice of μ^c, by automatically choosing suitable values of Δ_k^c.

The trust-region problem

The major task of the trust-region method is to solve the trust-region problem (8) (see Nocedal and Wright, 1999 and Conn et al., 2000). As the matrix H_k involved in the quadratic function is sparse and large and as the problem (8) is constrained, an iterative solver is recommended. It is also sufficient for theoretical convergence. In our context, we propose to use the Steihaug-Toint approach (a truncated CG method) which does not require the exact solution of a linear system. The approach consists in approximately solving the optimality condition

$$H_k \delta m = -g_k \ , \tag{12}$$

of the strictly convex quadratic function F_k by a CG algorithm, and in ensuring that the computed CG step stays inside the trust-region : $|| \, \delta m \, || \leq \Delta_k$.

In our case, H_k is positive definite, so the only difference with the standard CG algorithm of the line search method is that the algorithm stops when the trust-region bound is violated (a negative curvature is never encountered in our case). Note that the truncation of the CG algorithm does not inhibit the global convergence of the GN algorithm. Indeed, the global convergence of the GN algorithm is ensured if at each GN step we reach at least the Cauchy point[1] δm^1 defined as

$$\delta m^1 = \tau \delta \tilde{m}^1 \tag{13}$$

[1] i.e. we minimize the linearized function at least at the Cauchy point

with

$$\delta\tilde{m}^1 = -\frac{g_k^T g_k}{g_k^T H g_k} g_k, \quad \tau = \frac{1}{||\,\delta\tilde{m}^1\,||} \min(||\,\delta\tilde{m}^1\,||, \Delta_k).$$

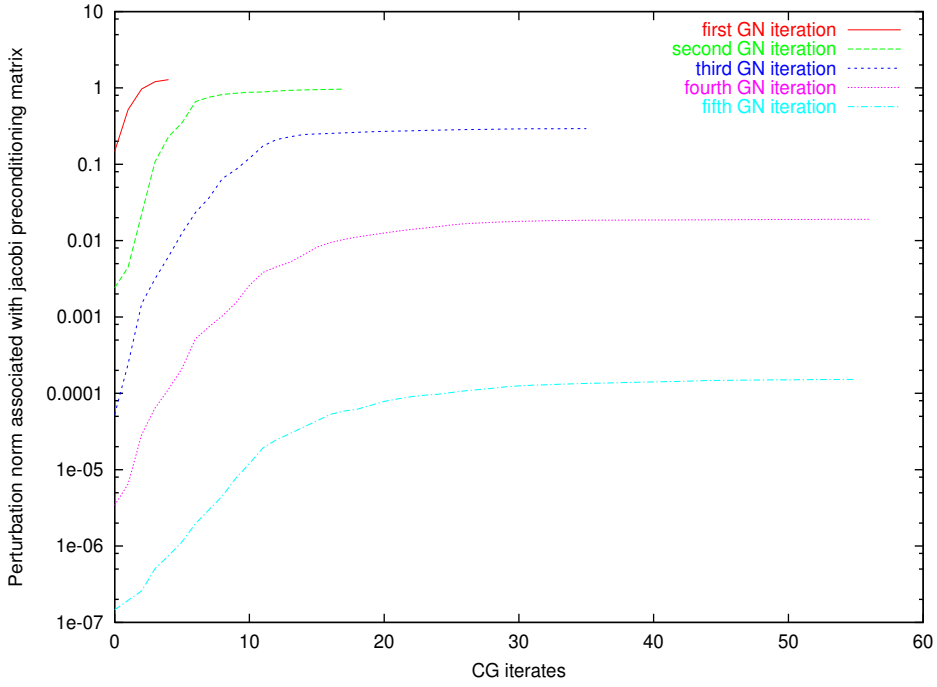

Fig. 1 Example 1 : Variations of the perturbation norm associated with the Jacobi preconditioning matrix versus CG iterates for different GN iterations. It is an application on a synthetic example presented in the following section. It shows, for the 6 GN iterations, the variations of the P_k norm of the perturbation versus CG iterates. Thus, we observe as predicted by the theory that whatever the GN iteration, the P_k norm of the perturbation increases with the CG iterates.

An interesting property of the CG algorithm is that the norm $||\,\delta m^k\,||$ strictly increases with k (see Figure 1 and Steihaug, 1983). Therefore, it makes sense to stop the CG iterates has soon as an iterate leaves the trust-region (the following iterates will not come back inside the trust-region). Since F_k decreases along the CG path, the point where this one crosses the trust-region boundary is the best one that the CG algorithm can find inside the trust-region. It can be shown that, when the model m_k becomes closer to a strong solution of the minimization problem (2), the trust-region constraint becomes inactive allowing the CG algorithm to effectively find the solution of (12).

To summarize, Steihaug's approach applied to a positive definite matrix has two possible behaviors : a solution within the trust-region can be found, i.e. the CG convergence criterion has been reached (Figure 2 right), or the solution lies outside the trust-region and the model is updated at the intersection of the trust-region boundary and the path followed by the CG iterates (Figure 2 left).

Preconditioning the CG method

It is often interesting to use a preconditioned CG algorithm to speed up the convergence, and sometimes it is even crucial to achieve convergence (see Saad, 1996). Preconditioning consists in transforming the problem (12) to the equivalent normal system

$$P_k^{-1/2}\, H_k\, P_k^{-1/2}\, P_k^{1/2} \delta m = -P_k^{-1/2}\, g_k, \tag{14}$$

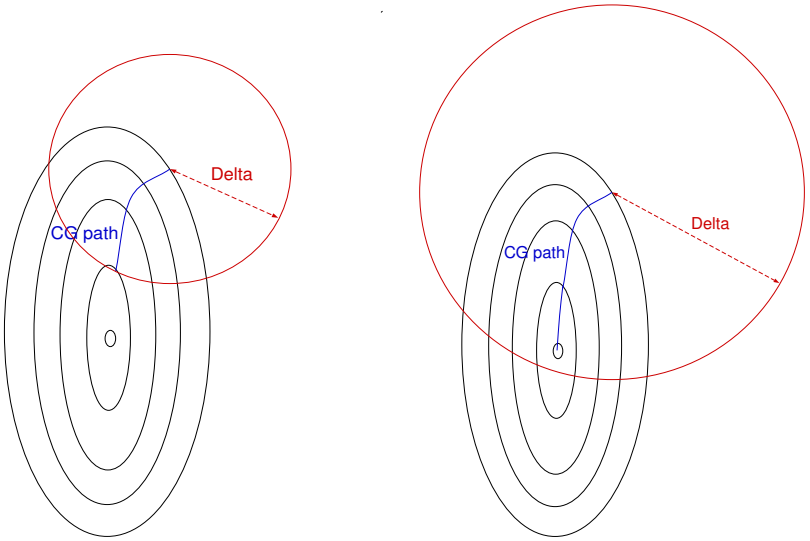

Fig. 2 Left : the solution lies outside the trust-region, the method ends when a CG iterate crosses the boundary, right : the solution is inside the trust-region, the method ends when convergence is reached.

with P_k the symmetric positive definite preconditioning matrix. The objective is to choose P_k as close as possible to H_k so that the condition number of $P_k^{-1} H_k$ is close to 1 or the eigenvalues of $P_k^{-1} H_k$ are more clustered. Preconditioning techniques have already been implemented in the tomographie software *jerry* by Chauvier et al., 2000. They have proposed two types of block preconditioners that derive directly from the Jacobi and Gauss-Seidel method. Both preconditioners are built on the decomposition of the Hessian matrix into velocity and interface blocks as written below :

$$H_k = \left(\begin{array}{cccc|ccc} H_{v_1,v_1} & H_{v_1,v_2} & \dots & H_{v_1,v_l} & H_{v_1,z_1} & \dots & H_{v_1,z_p} \\ H_{v_2,v_1} & H_{v_2,v_2} & \dots & H_{v_2,v_l} & H_{v_2,z_1} & \dots & H_{v_2,z_p} \\ \vdots & \vdots & \ddots & \dots & \dots & \dots & \dots \\ H_{v_l,v_1} & \vdots & \vdots & H_{v_l,v_l} & H_{v_l,z_1} & \dots & H_{v_l,z_p} \\ -- & -- & -- & -- & -- & -- & -- \\ H_{z_1,v_1} & \vdots & \vdots & H_{z_1,v_l} & H_{z_1,z_1} & \dots & H_{z_1,z_p} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \dots \\ H_{z_p,v_1} & \vdots & \vdots & H_{z_p,v_l} & H_{z_p,z_1} & \vdots & H_{z_p,z_p} \end{array} \right) \tag{15}$$

where l and p are respectively the number of velocities and interfaces to be inverted. This Hessian can be decomposed in a sum of block matrices

$$H_k = L_k + D_k + L_k^\top,$$

where D_k is a block diagonal matrix and L_k is the lower block triangular part of H_k :

$$D_k = \begin{pmatrix} H_{v_1,v_1} & & & & & & \\ & \ddots & & & 0 & & \\ & & H_{v_l,v_l} & & & & \\ & & & H_{z_1,z_1} & & & \\ & 0 & & & \ddots & & \\ & & & & & H_{z_p,z_p} \end{pmatrix} \tag{16}$$

$$L_k = \begin{pmatrix} H_{v_2,v_1} & & & & & \\ \vdots & \ddots & & & 0 & \\ H_{v_l,v_1} & \vdots & H_{v_l,v_{l-1}} & & & \\ H_{z_1,v_1} & \vdots & \vdots & H_{z_1,v_l} & & \\ \vdots & \vdots & \vdots & \vdots & \ddots & \\ H_{z_p,v_1} & \vdots & \vdots & \vdots & \vdots & H_{z_p,z_{p-1}} \end{pmatrix} \tag{17}$$

Since D_k is assumed to be positive definite, two classical choices for P_k are

$$P_k = D_k \quad (Jacobi),$$

or

$$P_k = \left(L_k^\top + D_k\right)\, D_k^{-1}\, \left(L_k + D_k\right) \quad (Gauss - Seidel).$$

These preconditioners give accurate solutions of the linearized problem and speed-up the convergence of the conjugate gradient method (see Chauvier et al., 2000).

When using a preconditioner to solve (12) in the framework of trust-region method, the norm that controls the trust-region boundary has to be changed. The successive iterates of the preconditioned conjugate gradient don't grow monotonously in l_2 norm, but in the norm associated with the preconditioning matrix. Thus, we have the new optimisation problem :

$$\text{minimize} \quad F_k(\delta m) = \tfrac{1}{2}\, \delta m^\top H_k\, \delta m + g_k^\top \delta m + f_k \quad s.t. \,||\, Q_k * \delta m \,||_2 \leq \Delta_k, \tag{18}$$

with $Q_k = P_k^{1/2}$, and $||\, Q_k . \,||_2 = ||\, . \,||_{P_k}$ the norm associated with the conditioning matrix. The only difference with the standard problem (8) is that the norm that controls the trust-region boundary has changed from l_2 to P_k. With the change of variables $\widetilde{\delta m} = Q_k\, \delta m$, we have

$$\widetilde{g_k} = Q_k^{-T}\, g_k, \; and \; \widetilde{H_k} = Q_k^{-T}\, H_k\, Q_k^{-1}$$

and (18) yields

$$\text{minimize} \quad \tfrac{1}{2}\, \widetilde{\delta m}^\top \widetilde{H_k}\, \widetilde{\delta m} + \widetilde{g_k}^\top \widetilde{\delta m} + f_k \quad s.t. \left|\left| \widetilde{\delta m} \right|\right|_2 \leq \Delta_k. \tag{19}$$

The above equation has exactly the same form as (8). Thus, a standard CG algorithm can be applied. In practice, we solve the problem (18) with a preconditioned CG (PCG) algorithm in which the P_k norm ($||\, . \,||_{P_k}$) of the iterate controls the trust-region radius Δ_k.

The implementations of the Jacobi and Gauss-Seidel preconditioners imply the resolution of a linear system at each iteration of the PCG loop. This linear system should be solved rapidly, and a common practice is to use a Cholesky factorization of the block diagonal matrix D_k

$$D_k = L_{D_k} L_{D_k}^T,$$

where L_{D_k} is a block diagonal and lower triangular matrix. With this new formulation of the block diagonal matrix D_k, we can now express the l_2 norm $|| Q_k * . ||_2$ associated with Jacobi or Gauss-Seidel preconditioners as

$$Q_k = L_{D_k}^T \quad (Jacobi)$$

$$Q_k = L_{D_k}^{-1} L_k + L_{D_k}^T \quad (Gauss-Seidel)$$

In our tomographic inverse problem the Hessian may become quasi-singular. In fact, a badly parameterized or a under-regularized problem often leads to a quasi-singular Hessian and a strongly ill-conditioned matrix D_k, with eigenvalues close to zero. In such a case preconditioning becomes very inefficient.

This is due, in part, to the Cholesky factorization which is numerically unstable when applied to such matrices (see Nocedal and Wright, 1999). To overcome these difficulties, a modified Cholesky factorization can be used. This factorization consists in modifying the matrix D_k during the course of the Cholesky standard factorization such that the diagonal elements of L_{D_k} are sufficiently positive and the elements of L_{D_k} are not too large. The modifications are controlled by the two positive parameters δ and β. For the computation of the jth column of L_{D_k} we impose

$$(L_{D_k})_{jj} \geq \delta$$

$$|(L_{D_k})_{ij}| \leq \beta \quad i = j+1, ..., n \quad .$$

The final factorization is given as

$$P D_k P^T + E = L_{D_k} L_{D_k}^T,$$

where P is the permutation matrix associated with the row and column interchanges, and E is a non-negative diagonal matrix that is zero if D_k has large enough eigenvalues . The computation of the matrix E is controlled by the parameters δ and β. The parameter δ is chosen close to the machine accuracy u

$$\delta = u \ \max(\gamma + \xi, 1),$$

where γ and ξ are respectively, the largest-magnitude diagonal and off-diagonal elements of the matrix D_k :

$$\gamma = \max_{1 \leq i \leq n} |(D_k)_{ii}|, \quad \xi = \max_{i \neq j} |(D_k)_{ij}| \quad .$$

The value chosen for β is the one proposed by Gill et al., 1981, which minimizes the infinite norm of the modification $||E||_\infty$

$$\beta = \max\left(\gamma, \frac{\xi}{\sqrt{n^2 - 1}}, u\right)^{1/2}$$

APPLICATIONS OF THE TRUST-REGION METHOD

The studied models

We have used error-free synthetic data to study our new algorithm. Data have been generated by ray-tracing on a given model (the same ray-tracing as the one used in the inversion). As can be seen in Table 1, this model ("the true model") is composed of one interface (40*40 coefficients) and one 2D velocity (14*8 coefficients).

Three initial models have been built to analyze the behavior of the trust-region method. These models are very similar : the interface is described by 20*20 coefficients and the velocity is described by 10*10 coefficients along x and y directions. However, the three models have a different velocity description along the z direction. As shown in Table 1, whereas the first example presents no velocity variations along z, the velocity of the second and third examples are described respectively by 5 and 10 B-spline coefficients along the z axis. This set-up has been chosen since we had noticed experimentally that vertical velocity variations are difficult to retrieve from the travel-time data (the rays are too vertical) : trying to invert for such variations leads to an ill-conditioned Hessian matrix and causes troubles to the line search method. Thus these three examples should give us a palette of more or less unstable problems and we wish to examine how our trust-region method and classical line search algorithms cope with these situations.

	Interface	velocity	Total number of parameters
true model	x :40 y :40	x :14 y :8 z :0	1712
example 1	x :20 y :20	x :10 y :10 z :0	500
example 2	x :20 y :20	x :10 y :10 z :5	900
example 3	x :20 y :20	x :10 y :10 z :10	1400

Table 1 The B-spline coefficients of the three examples and the true model

CG termination criteria

In our TR method, the iterates of the conjugate gradient can reach convergence inside the trust-region. In theory, since H is symmetric positive definite, the preconditioned conjugate gradient algorithm may find a solution within a fixed number of iterates. Furthermore, we know that it should convergence in at most n iterates, where n is the number of unknowns (see Fletcher, 1987).

So, a first criterion to stop the PCG algorithm is

$$i \leq n \tag{20}$$

where i is the current number of performed PCG iterations. Most of the time, this criterion leads to useless iterations, the convergence being achieved before. But this test is widely used as a rule of last resort.

The most popular rule for stopping the preconditioned CG algorithm is to look whether the residual norm at step i has been reduced to a small fraction of its initial value. It is called the relative residual criterion,

$$|| r_{k,i} ||_2 \leq \varepsilon_1 \ || r_{k,0} ||_2 \tag{21}$$

where ε_1 corresponds to the expected accuracy (or the fraction of reduction), and $r_{k,i}$ is the residual at step i.

Despite its popularity, this test raises the difficulty of choosing ε_1. It is not only difficult to have a physical intuition of a value for ε_1, but the series of $(|| r_{k,i} ||_2)_i$ is not monotone during CG iterations, too. Different values of ε_1 from $1E-1$ to $1E-15$ where taken. Variations of the non-linear objective function and the number of CG iterations versus the GN iteration index have been plotted in Figure 3 and Figure 4 respectively. We see that the higher the accuracy is (ε_1 small), the more the non-linear objective function is minimized. However from Figure 4 we see that the higher the accuracy required, the more CG iterations are needed to reach (21) : the decrease in the non-linear objective function is the same for an accuracy of $\varepsilon_1 = 1E-15$ or $1E-5$. But the number of CG iterations has doubled from $\varepsilon_1 = 1E-5$ to $1E-15$. As a result we can state that it is more interesting to choose $\varepsilon_1 = 1E-5$ since it yields results equivalent to $\varepsilon_1 = 1E-15$ and saves approximately half of the CPU time. However, there is no a priori method to predict the appropriate accuracy for a given problem. Renard and Lailly, 1999, have reported that, for a simple model, it was not necessary to solve accurately the quadratic problem (8) to obtain a good decrease of the non-linear objective function. Moreover, Chauvier et al., 2000, have noticed in their experiments that there were only few differences between the non-linear objective functions obtained with more or less accuracy (resp. $\varepsilon_1 = 1E-15$ or $\varepsilon_1 = 1E-5$). We have observed the same results inverting our first example with a line search method.

Another widely known CG termination criteria is based on test (21) with a non-constant accuracy. This test, called the advanced relative residual criterion is given as

$$|| r_{k,i} ||_2 \leq || r_{k,0} ||_2 \; min(\varepsilon_4, || r_{k,0} ||_2^{\theta}) \;\; or \;\; k > n \tag{22}$$

with $\varepsilon_4 < 1$ and $\theta > 0$ ($\theta = 0.5$ is a common choice). If $\theta = 0$, this rule is equivalent to the relative residual criterion (21) with $\varepsilon_4 = \varepsilon_1$. As long as $\theta > 0$, super linear convergence [2] is possible (Conn et al., 2000). We have compared this rule with the relative residual criterion for the example 1 and $\varepsilon_4 = \varepsilon_1 = 1E-2$, looking at variations of the gradient versus GN iterations. We observe similar variations of the gradient for both rules. We thus conclude that this advanced relative residual criterion is not useful for our tomography problems : the number of CG iterations increases, while better decrease is not provided.

Two other stopping criteria have been examined that depend on the decrease of the quadractic objective function between 2 PCG iterates $|F_{k,i-1} - F_{k,i}|$. This term seems more physical than the relative residual criteria, and closer to user concern. But the term $|F_{k,i-1} - F_{k,i}|$ is also non-monotone with i.

The first criterion is the Nash criterion (see Nash and Sofer, 1990)

$$F_{k,i-1} - F_{k,i} \leq \varepsilon_2 \, \frac{F_{k,0} - F_{k,i}}{i} \;\; , \; i \geq 1 \tag{23}$$

where $F_{k,i} = F_k(\delta m_i)$ ($F_{k,0} = f_k$) and δm_i is the i^{th} iterate of the PCG algorithm. This test suggests to stop the CG iterations when the decrease of the quadratic objective function during the last iteration is less than a fraction ε_2 of the average decrease. In Figure 5 we have plotted the values of the relative residual $|| r_{k,i} ||_2 / || r_{k,0} ||_2$ versus GN iterates for four different values of the Nash accuracy (ε_2 =1E-2, 1E-4, 1E-6, and 1E-8). We observe that the curves have globally a small positive slope. In the case of an exact equivalence between the relative residual criterion and

[2] superlinear convergence to m_* is defined by :

$$\frac{|| m_{k+1} - m_* ||}{|| m_k - m_* ||} \longrightarrow 0$$

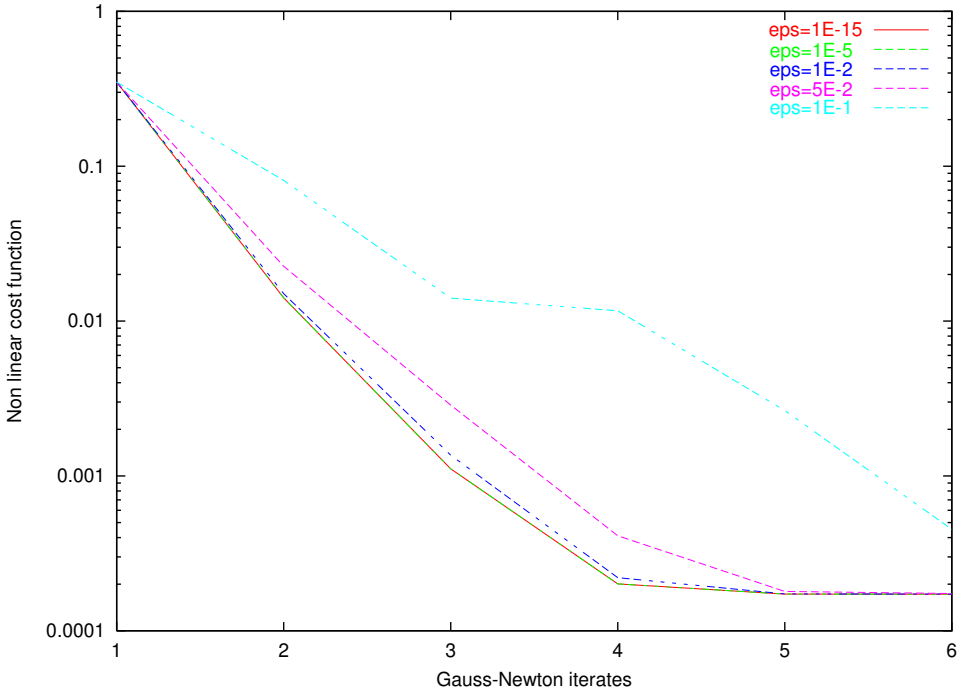

Fig. 3 Ex.1 : Variation of line search objective function versus GN iterates for different values of ε_1

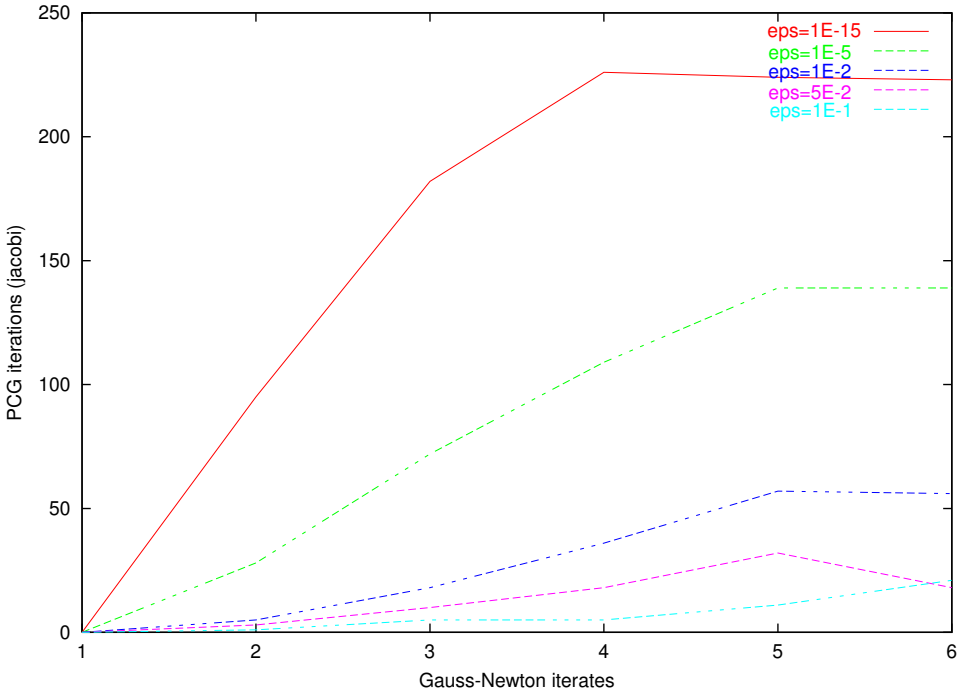

Fig. 4 Ex.1 : Variation of line search CG iterations versus GN iterates for different values of ε_1

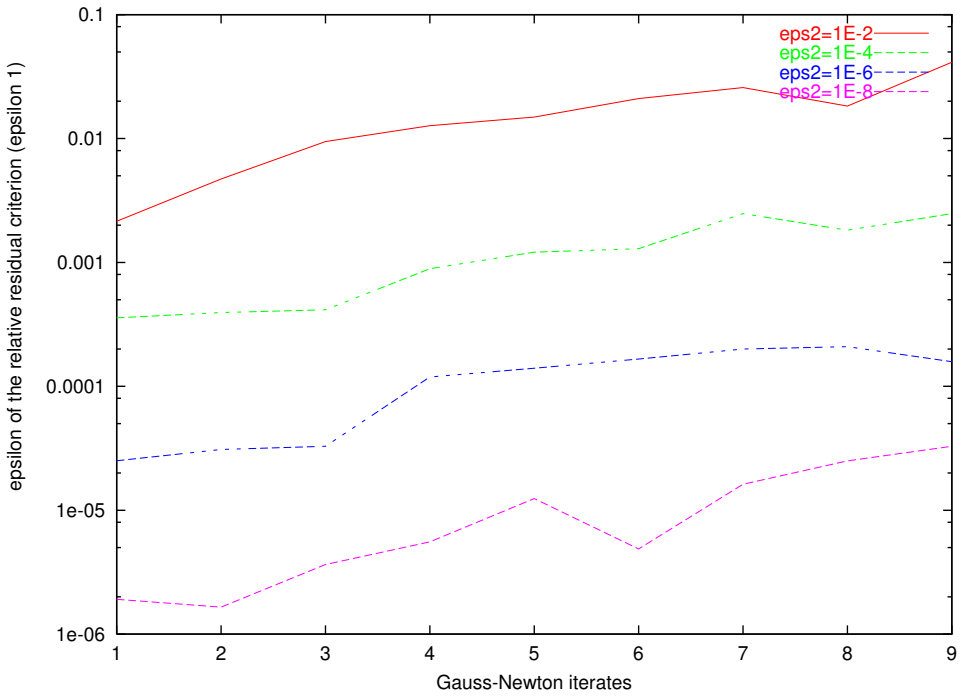

Fig. 5 Ex.1 : Variation of ϵ_1 versus GN iterates for different Nash accuracy ϵ_2 with a line search method as inversion

the Nash criterion we would observe constant curves. Since the slope of the curves in Figure 5 is very small, we can conclude that in our application both tests are almost equivalent.

The second, original, criterion, called the final cost reduction criterion (FCR), is defined as

$$F_{k,i-1} - F_{k,i} \leq \varepsilon_3 \frac{F_{k,i-1}}{n-i} \quad and \quad 0 < i < n. \tag{24}$$

This test, detailed in Figure 6, suggests to stop the CG iterations when the predicted reduction for the $n-i$ iterates ($|F_{k,i-1} - F_{k,i}| * (n-i)$) is lower than ε_3 times the previous quadratic cost $F_{k,i-1}$ (called here the potential reduction). We hope the real reduction to be overestimated by the predicted reduction.

In Figure 7 and 8 we compare the FCR criterion with respect to the relative residual criterion for example 1. The non-linear objective function and the number of CG iterations are respectively represented in Figure 7 and 8. In both graphics, two curves for the relative residual criterion (accuracies $\varepsilon_1 = 1E-15$ and $1E-5$) and four curves for the FCR criterion have been plotted (FCR accuracies of $\varepsilon_3 = 1E-1$, $1E-2$, $1E-3$ and $1E-5$). The FCR curves are identical to the reference curve if the accuracy is higher than $1E-3$ (0.1%). Between $\varepsilon_3 = 1E-3$ and $1E-5$, we notice from Figure 8 that many CG iterations have been saved compared to the reference curve. In addition, the FCR curves are even better than the relative residual curve with an accuracy of $\varepsilon_1 = 1E-5$.

We also compare the FCR criterion with the relative residual criterion for example 2 in Figure 9, 10 and 11, which represent respectively the evolution of the non-linear objective function, the CG iterates and the trust-region radius versus GN iterates for different FCR accuracies. Figure 9 shows that the instabilities (rejected GN step) at GN iterations 4 and 7 for the relative residual criterion have vanished when using a FCR accuracy of $\varepsilon_3 = 1E-4$ or $1E-3$. For these two values, the cost has been sufficiently reduced compared to the reference curve (the relative residual

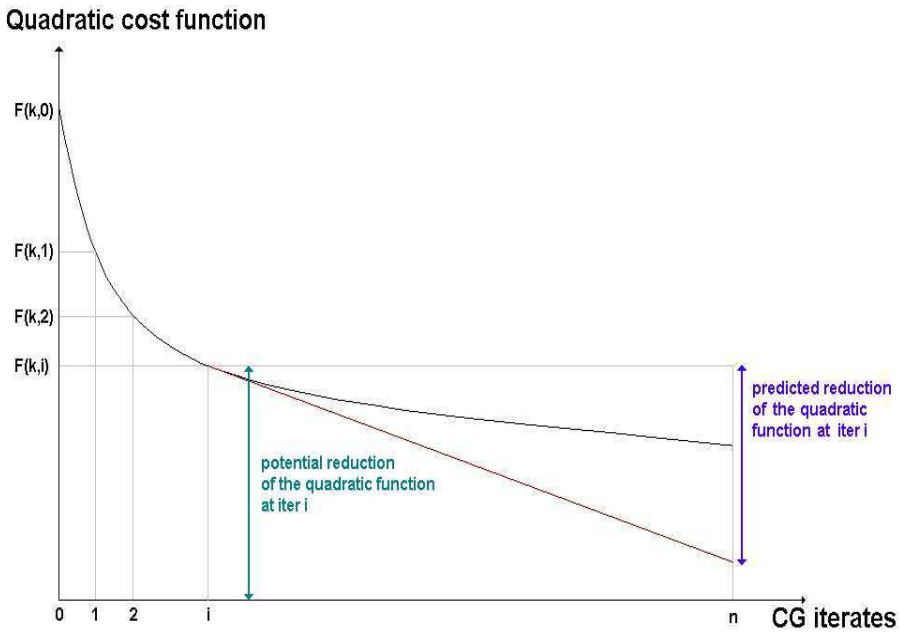

Fig. 6 FCR critetion

curve with $\varepsilon_1 = 1E-15$). Furthermore, we see in Figure 10 that the number of CG iterations needed to converge with the FCR criterion is still less than the reference curve. And as expected, the lower the accuracy is, smaller the number of CG iterations needed. Finally, Figure 11 shows that the trust-region radius is constant for FCR curves after the first GN iterates, in other word there are no GN failures.

For both examples using a FCR criterion with an accuracy of roughly $\varepsilon_3 = 1E-4$ is recommended to minimize sufficiently the cost and save the maximum number of CG iterations.

Preconditioning

We are studying here the application of the trust-region method with or without preconditioning. Tab. 2 shows the global behavior of the different inversion methods : Y means that the corresponding approach managed to reduce sufficiently the non-linear objective function, i.e. the GN algorithm converged towards a minimum ; N means that the algorithm did not succeed to invert the example. In example 3, the line search method fails to find a minimum with or without the Jacobi preconditioner. The algorithm stopped when the maximum number of unsuccessful GN iterations was reached. In the second and third examples, the preconditioned trust-region method failed also to give a solution. The symptom of failure is similar : as soon as the first CG iteration is computed the model perturbation in l_2 norm diverges. For examples 2 and 3, we observe at the first CG iteration a norm of the model perturbation greater than 1000 times the norm of the initial model.

Contrary to example 1, the Hessians computed in examples 2 and 3 are ill-conditioned, and the preconditioning matrices are also ill-conditioned. We propose two methods to overcome these difficulties.

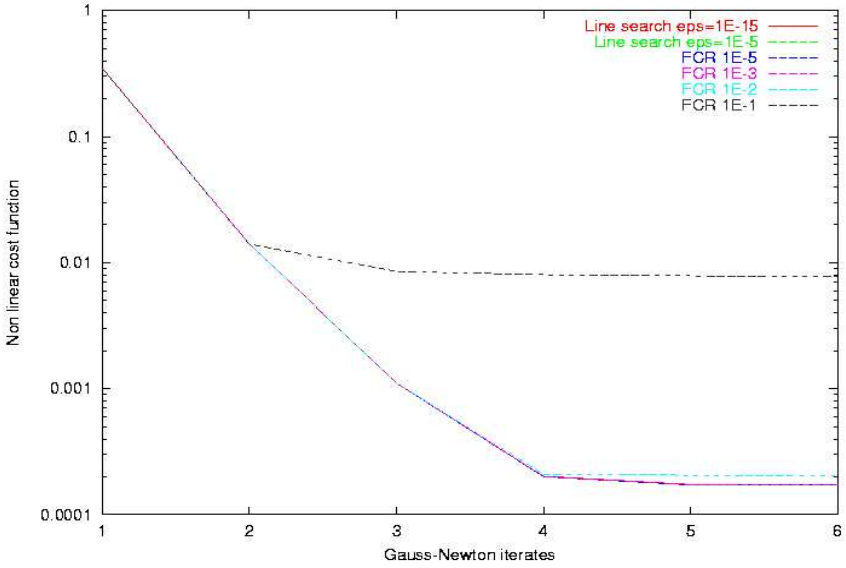

Fig. 7 Ex.1 : Comparison of FCR criterion with relative residual criterion : the non-linear objective function

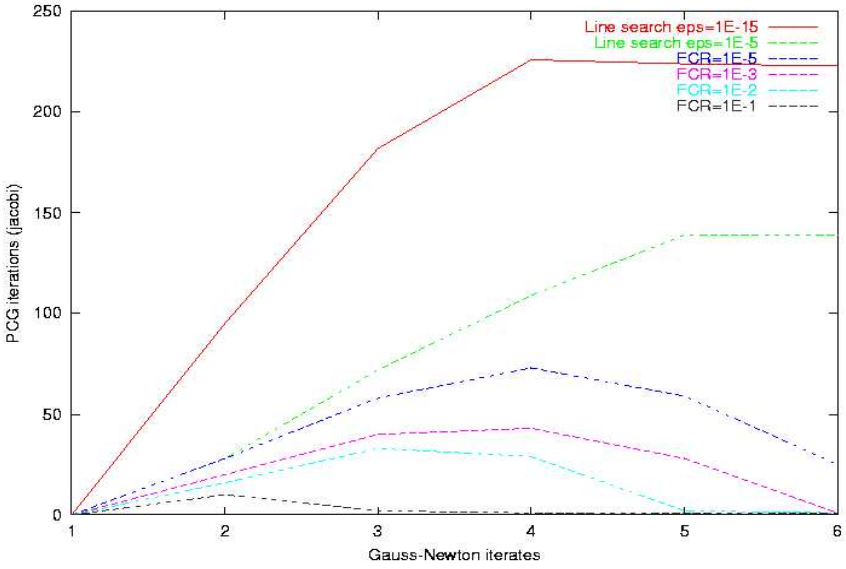

Fig. 8 Ex.1 : Comparison of FCR criterion with relative residual criterion : the number of CG iterations

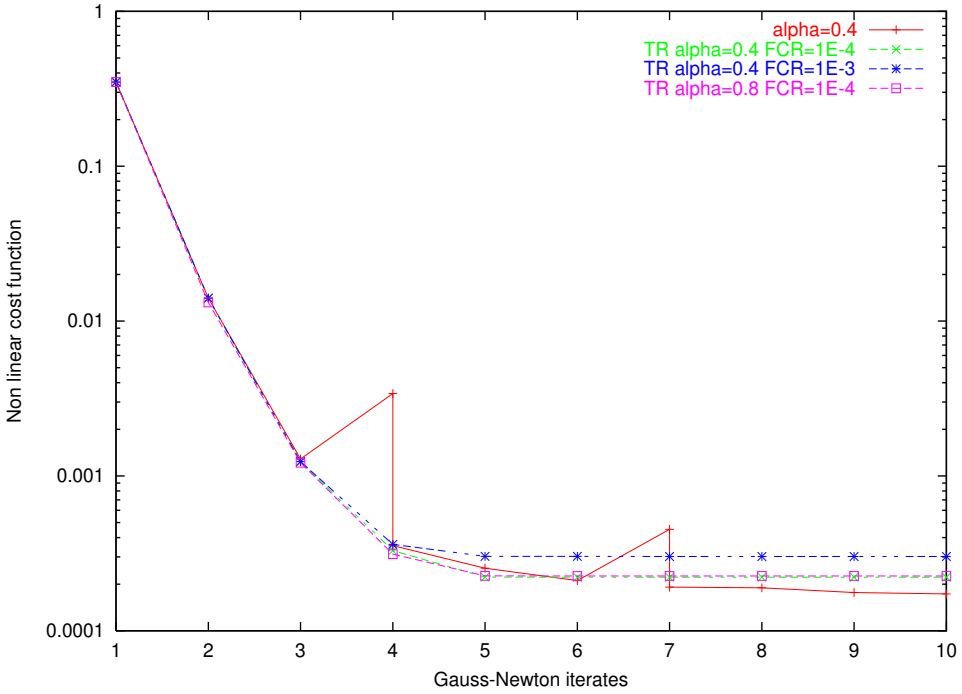

Fig. 9 Ex.2 : evolution of the non-linear objective function with GN iterates for different FCR accuracies

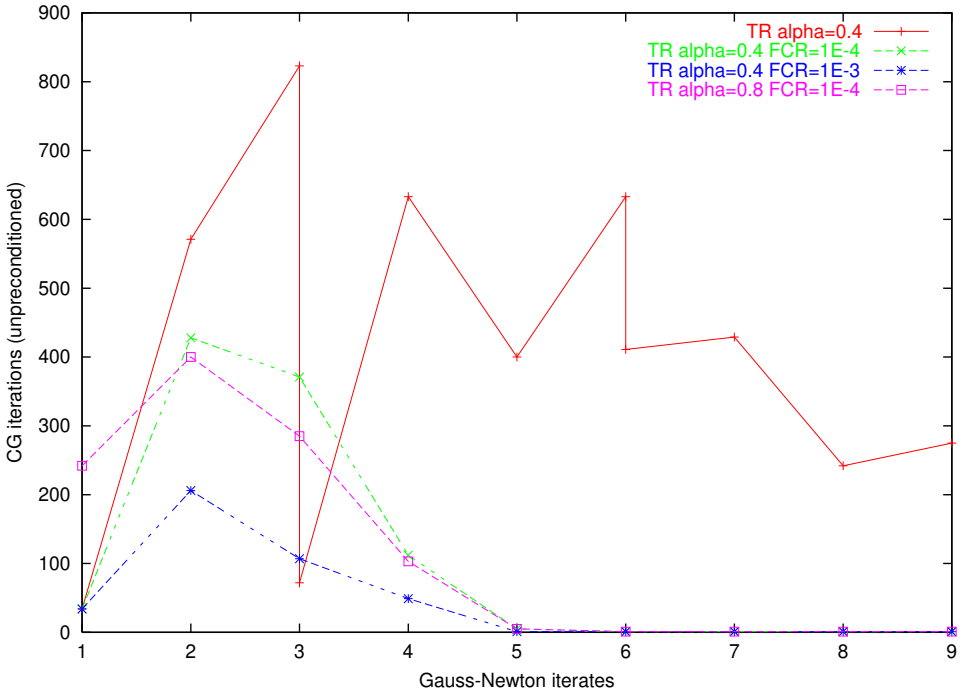

Fig. 10 Ex.2 : evolution of the CG iterates with GN iterates for different FCR accuracies

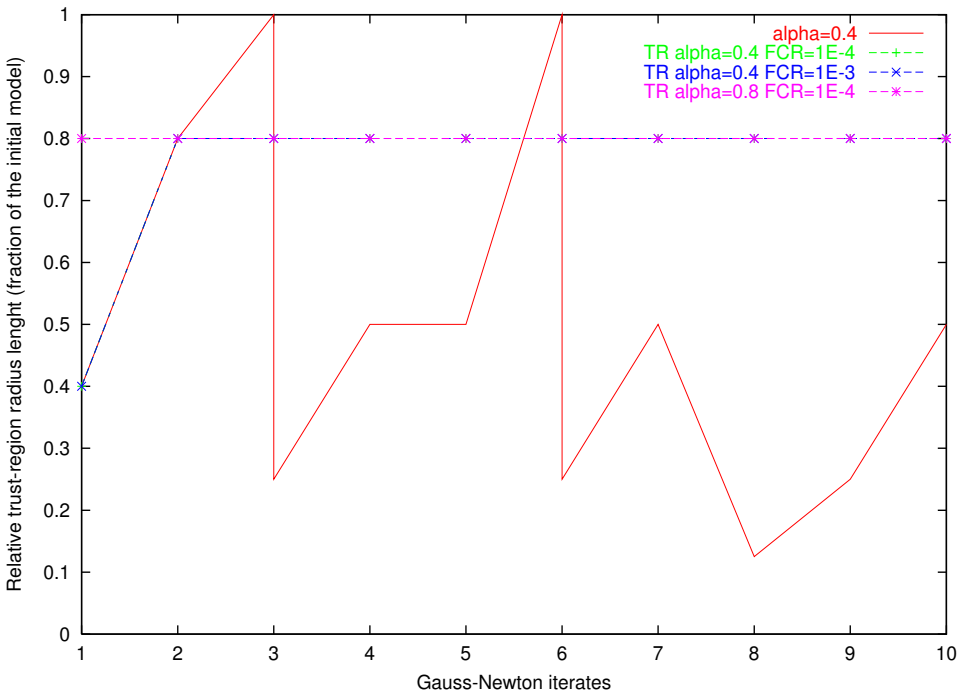

Fig. 11 Ex.2 : evolution of TR radius with GN iterates for different FCR accuracies

Succes	LS without prec.	LS with Jac. prec.	TR without prec.	TR with Jac. prec.
Example 1	Y	Y	Y	Y
Example 2	Y	N	Y	N
Example 3	N	N	Y	N

Table 2 Global behavior of the different inversion methods for our three test cases

The first and simpler one is to test in the CG loop if the l_2 norm of the model perturbation exceeds a fixed value (for instance 1000 times the initial model). If the test is positive, we restart the current GN iteration with a non-preconditioned method. This implementation enables us to use the trust-region method on our three examples (the gradient is small at convergence for our 3 examples).

The second method is to use a modified Cholesky factorization (MCF) instead of the standard Cholesky factorization (see the section on preconditioning the CG method). We expect this new factorization to stabilize the preconditioning and to give smaller model perturbations. Figure 12 (exemple 2) and 14 (exemple 3) represent the variation of the non-linear objective function versus GN iterates achieved with a trust-region method for different Jacobi MCF preconditioners. On both graphics, three curves show the effect of Jacobi MCF preconditioners taking different values of the machine accuracy ($u = 1E-2$, $1E-4$ and $1E-6$), and one reference curve shows the non-linear objective function behavior without preconditioner. The different values of the chosen machine accuracy do not represent the real value ($u \simeq 1E-14$), but they allow to have varying δ and β parameters for the MCF preconditioning. We notice on both graphics that the reference curve

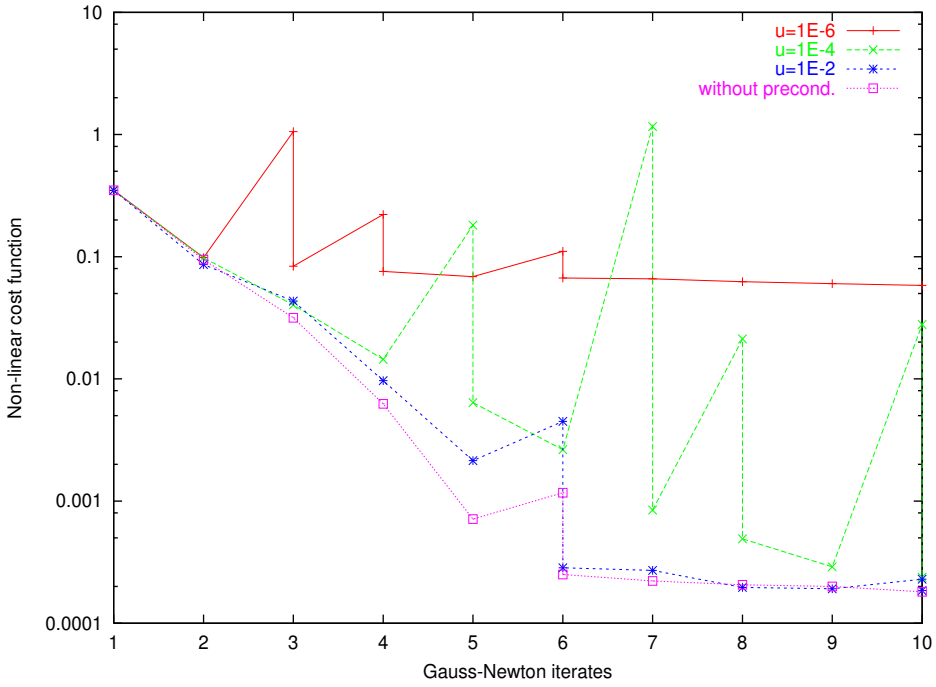

Fig. 12 Ex.2 : The variations of the non-linear objective function versus GN iterates for different Jacobi MCF preconditioners

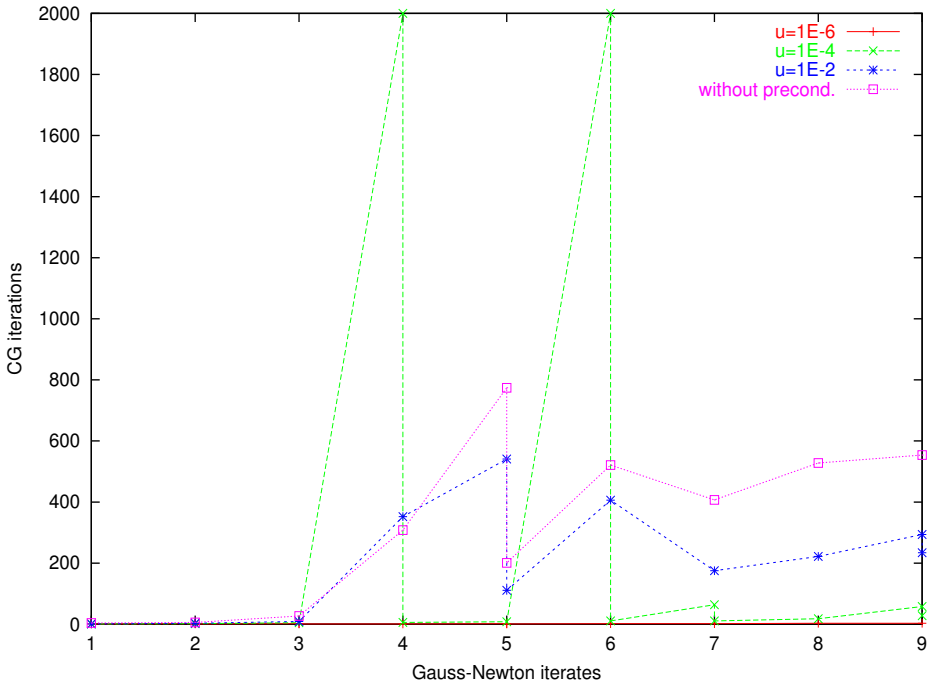

Fig. 13 Ex.2 : The variations of the CG iterations versus GN iterates for different Jacobi MCF preconditioners

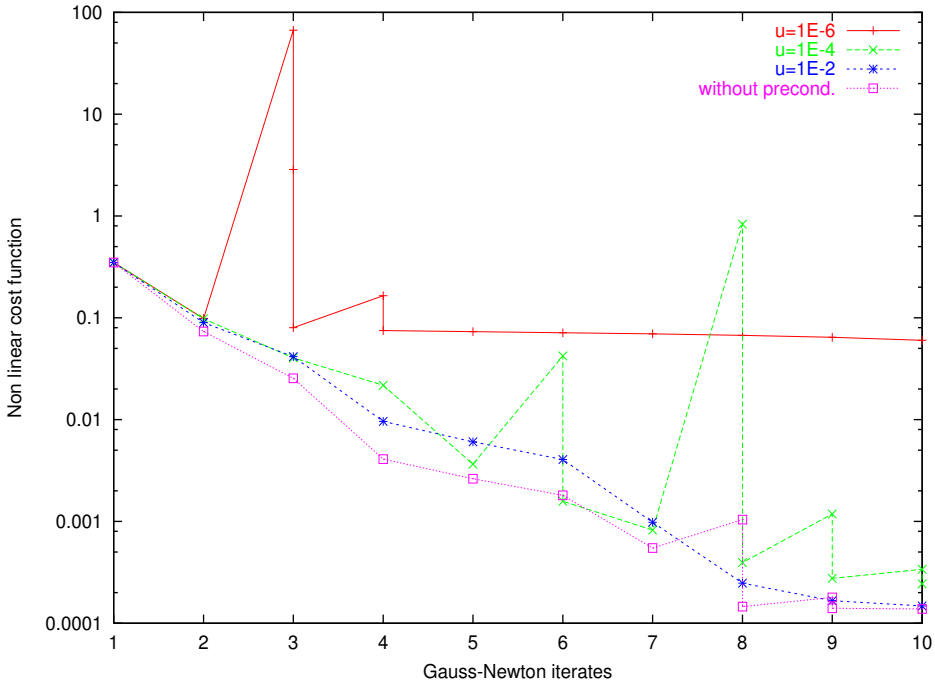

Fig. 14 Ex.3 : The variations of the non-linear objective function versus GN iterates for different Jacobi MCF preconditioners

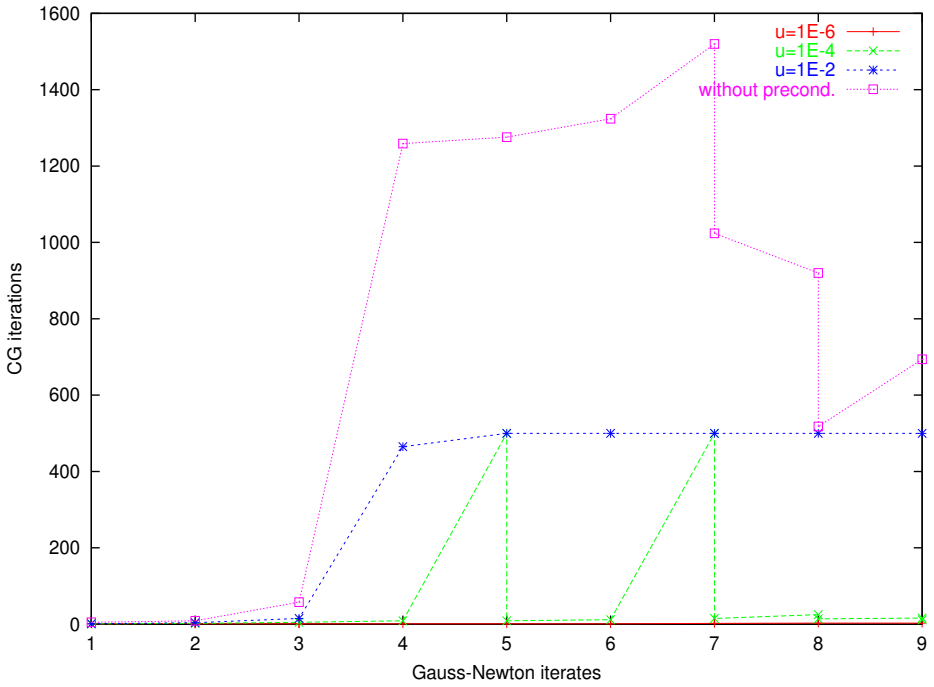

Fig. 15 Ex.2 : The variations of the CG iterations versus GN iterates for different Jacobi MCF preconditioners

(trust-region method without preconditioner) is below all the Jacobi MCF preconditioner curves. Further more, the MCF preconditioner is unable to decrease the non-linear objective function when taking a machine accuracy lower than $u = 1E-6$. Moreover, Figure 13 and 15 show the required number of CG iterations for example 3 (with a value of $u = 1E-2$). The number of CG iterations is roughly divided by 2 in the preconditioned case with respect to the unpreconditioned case. However for example 2, the reduction is almost not significant. These arguments may prove that the modified Cholesky factorization is troublesome for our applications.

As a consequence, it seems that our first proposition, solving without preconditioner examples 2 and 3, is more attractive. But, this method is not completely acceptable because solving the problem without preconditioning could leads to a bad convergence of the CG algorithm. Thus one important point of our future work will be to find suitable preconditioners for quasi-singular problems.

Trust-region versus truncated line search

In this section, we would like to emphasize the main properties of our trust-region method, compared to a truncated line search method. For a better understanding of the trust-region behavior, we have made a simple test on our first and second examples without preconditioner and with the relative residual criterion (21) as convergence test ($\varepsilon_1 = 1E-15$). This test consist in finding a model perturbation for the first GN step via trust-region or line-search method. In addition the perturbation found with the line search method is truncated to the length of the trust-region radius (the initial trust-region radius is chosen small enough to obtain a perturbation on the boundary). Then in both cases the model is updated and the non-linear objective function computed. Tab. 3 and Tab. 4 compare results achieved with trust-region or truncated line search methods for respectively the first and second example (the third example can not be inverted with a line search method). These results depend on the initial trust-region radius Δ_0 and on the concordance ratio ρ.

	cost reduction	CG iterations	perturbation angle
Trust-region	64.8%	4	42.4
line search	23.5%	5340	82.6

Table 3 example 1, trust-region versus truncated line search methods

	cost reduction	CG iterations	perturbation angle
Trust-region	72.9%	4	40.6
line search	28.9%	2000	80.8

Table 4 example 2, trust-region versus truncated line search methods

In both examples, the cost reduction is larger using the trust-region method. Moreover, it saves a lot of CPU time by computing only few CG iterations. The difference between a perturbation found with the trust-region method and the line search method is symbolized by the angle between the negative gradient and the perturbation : each perturbation has the same norm but not the same direction. Both tables show that the perturbation angle is different when using a trust-region or a truncated line search method. In theory, the perturbation angle is growing with the CG iterates and it is bounded between 0 (the first CG iterate is the gradient) and 90°. We can also observe on both tables that the line search perturbation angles are larger than the trust-region perturbation angles. This is likely explained by convergence of the CG (with the relative residual criterion) when using

the truncated line search method. The perturbations obtained via the trust-region method have smaller angles because the CG is stopped earlier by the trust-region boundary.

Initialization of the trust-region radius

In this section we focus our interest on the initialization of the trust-region radius. The trust-region radius is the value that constrains the norm of the model perturbation :

$$|| Q_0 \delta m ||_2 \leq \Delta_0 \tag{25}$$

where Δ_0 is the TR radius at the first GN step. At first sight, it seems difficult to give an appropriate value of this radius. Indeed, this radius does not only depend on the model to be inverted, but also on the preconditioner chosen in the CG algorithm. We propose to express Δ_0 as

$$\Delta_0 = \alpha \, || Q_0 * m_0 ||_2 \tag{26}$$

with $\alpha > 0$ and m_0 the initial model. In this formulation we set Δ_0 to a fraction α of the norm (associated with the preconditioner) of the initial model (it means that the model perturbation should not exceed a fraction of the norm of the initial model). The value of the first radius Δ_0 is then determined by the choice of the coefficient α. The behavior of the non-linear objective function and of the number of CG iterates with respect to the value of the coefficient α have been analyzed on our three examples . Four values have been chosen for α (0.1, 0.2, 0.4 and 0.8) and the relative residual criterion (21) together with a line search method with an accuracy of $\varepsilon_1 = 1E-15$ has been choosen as reference.

We are first looking at the behavior of trust-region method applied to the first example for a Jacobi preconditioner. Since example 1 is not ill-conditioned and strongly non-linear, we expect the trust-region method to be as efficient as the reference line search method. Figure 16, 17 compare for different values of α ($0.1, 0.2, 0.4$, and 0.8) the evolution of respectively the non-linear objective function and the number of CG iterates. Figure 16 shows that the reference curve is under most of the alpha curves. When α is increased, the trust-region curves are coming closer to the reference curve, and for $\alpha = 0.8$ cost are even identical. We can also noticed a better convergence of the alpha curves in the last iterations of GN. However, the number of CG iterations represented in Figure 17 brings important information on the trust-region behavior. In fact we can observe on this graph that a slowdown of the GN convergence occurs when α is small. Choosing a small α leads to have GN perturbations on the trust-region boundary (see Figure 2 left). In fact, for $\alpha = 0.8$ (the largest value of α) the convergence inside the trust-region occurs for the first GN iteration (see Figure 2 right), and it explains the similarity with the reference curve. Using our trust-region method to invert simple models as example 1 (the Hessian is well-conditioned and the non-linear objective function is close to a quadratic model) implies to choose high values for the initial radius. Then, in case of no GN failures, the trust-region method is equivalent to the line search method. In example 1, the best value for α is 0.8 (any value for α greater than 0.8 will give similar results).

Let us now consider the behavior of trust-region method applied to the second example. Unlike the first example, analyses could not be carried out with preconditioning. For this intricate example, the line search method is expected to encounter some failures, and we hope the trust-region method to be more efficient. Figure 18 and Figure 19 represent respectively, for different values of α, the non-linear objective function and the number of CG iterates versus GN iterations. Contrary to the first example, the non-linear cost for the line search method is unstable. Indeed, the step-size has to be reduced almost twice at each GN iteration. Despite an unstable behavior,

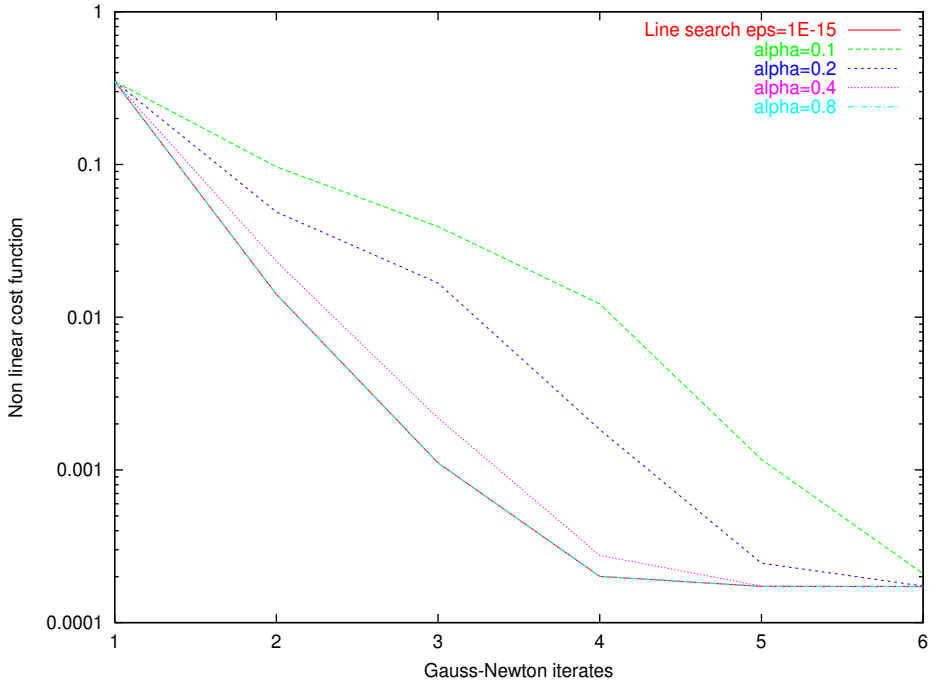

Fig. 16 Ex.1 : evolution of the non-linear objective function with GN iterates for different values of alpha

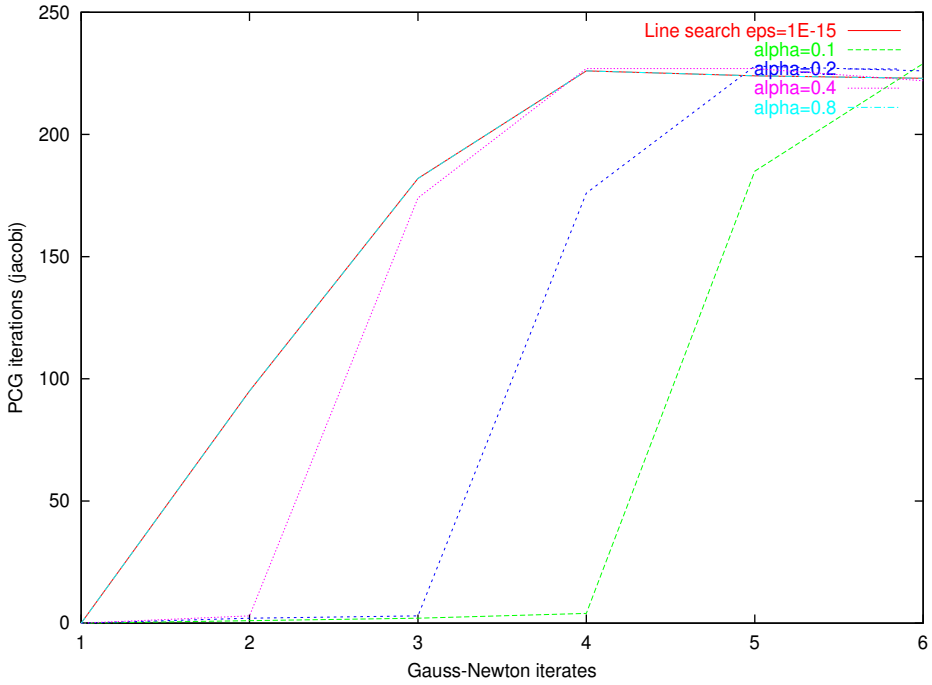

Fig. 17 Ex.1 : evolution of the CG iterates with GN iterates for different values of alpha

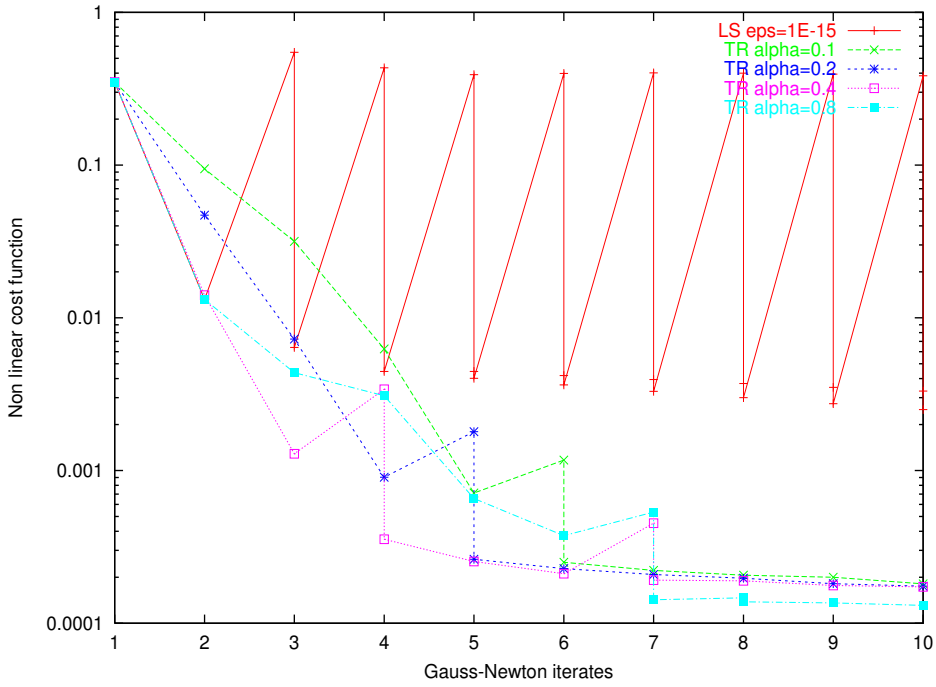

Fig. 18 Ex.2 : evolution of the non-linear objective function with GN iterates for different values of alpha

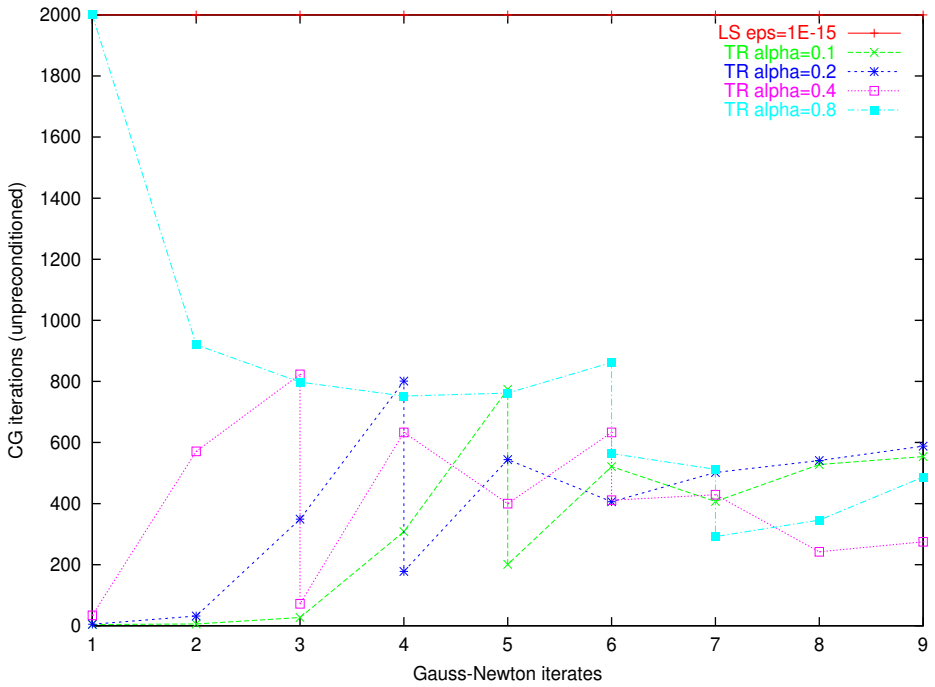

Fig. 19 Ex.2 : evolution of the CG iterates with GN iterates for different values of alpha (2000 is the maximum of CG iterates, the LS curve is stopped by this value)

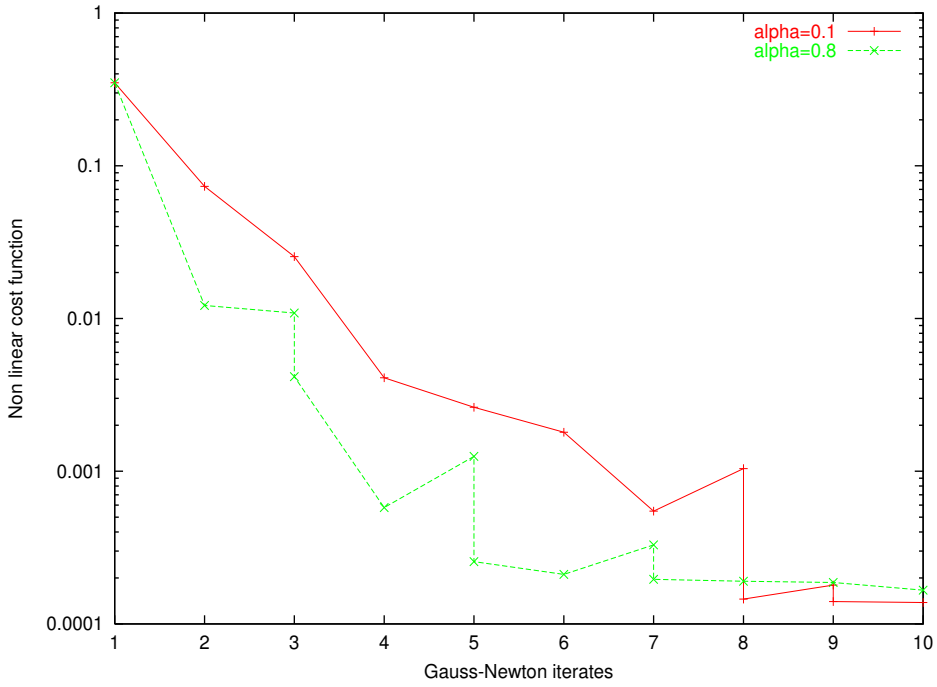

Fig. 20 Ex.3 : evolution of the non-linear objective function with GN iterates for different alpha=0.1 and 0.8

the curve is globally decreasing, minimizing the non-linear cost and finding a final cost of roughly $2E-3$. All the trust-region curves (from $\alpha = 0.1$ to 0.8) are below the reference one, and they are quite stable with at most two failures for 10 GN iterations. The final cost is below $2E-4$. Thus, whatever the value of α, the trust-region method minimizes better the non-linear objective function than the line search method. After 10 steps of GN, the final cost using trust-region method is 10 times lower than the one obtained with the line search method. We can notice slight differences between the cost of our different trust-region curves. The higher the length of the trust-region radius is, the smaller the final cost is : the best cost is then achieved with the highest values of α ($= 0.8$).

In the third example, the z direction is discretized with 10 B-spline coefficients. Contrary to the second example, the line search method fails with or without preconditioning to minimize the non-linear objective function. Only the trust-region method without preconditioning is able to invert our third example. In Figure 20 we have plotted the variations of the non-linear objective function versus GN iterations for $\alpha = 0.1$ and $\alpha = 0.8$. It shows, for both curves, that the cost is reduced to approximately $2E-4$. The plateau of convergence is reached with less GN iterations for $\alpha = 0.8$ than for $\alpha = 0.1$.

For our three examples, choosing a large value of α ($\alpha \geq 0.8$) is recommanded to better minimize the non-linear objective function.

TRUST-REGION ALGORITHM EXTENSIONS

This section aims at giving some other interesting extensions of our trust-region algorithm to be studied.

Initialization of trust-region radius

Since in our tomographic problem the GN convergence is generally reached in few iterations, the choice of the initial radius of the trust-region is crucial (see Sartenaer, 1997). The method implemented (see for instance the last section), that is based on computing the initial preconditioned norm of the model, is not commonly used. The goal of this subsection is to review the common practice techniques to initialize the trust-region radius. Usually, the user has no idea of the initial size of the region to choose. Then, the algorithms often compute the initialization of the trust-region radius thanks to heuristics. We give below two simple strategies to choose the initial trust-region radius. These rules are more general, not depending on the model to be inverted.

The first rule simply consists in choosing the initial radius equal to 1 :

$$\Delta_0 = 1$$

The second rule expresses the trust-region radius as

$$\Delta_0 = \overline{\alpha} \,||\, g_0 \,||$$

with $\overline{\alpha}$ a fraction of the initial norm of the gradient g_0. In practice, it is expensive to find a good value of $\overline{\alpha}$ (see Sartenaer, 1997).

Trust-region radius updating rule

The practical success of trust-region algorithm depends on a set of constant parameters. As described in appendix A, these parameters aim at determining the new trust-region radius Δ_{k+1} as a function of the current concordance ratio ρ_k. The update of the trust-region radius is achieved thanks to a flexible rule of the form :

$$\Delta_{k+1} = \begin{cases} \nu_1 * \Delta_k & \text{if } K_2 > \rho_k \\ \nu_2 * \Delta_k & \text{if } K_3 > \rho_k \geq K_2 \\ \nu_3 * \Delta_k & \text{if } \rho_k \geq K_3 \end{cases} \tag{27}$$

with K_i corresponds to the threshold parameters and ν_i the increasing/decreasing factors. The experimental values chosen here for the different parameters are often quoted in literature with $K_2 = 0.25$, $K_3 = 0.75$, $\nu_1 = 0.5$, $\nu_2 = 0.9$, $\nu_3 = 2.0$ (see Conn et al., 2000). This class of parameters will be referenced to as the usual parameters.

One important problem when choosing these parameters is to know whether their variation does influence or not the global convergence of the trust-region algorithm. We observe experimentally that ill-conditioned examples are more sensitive to these parameters. Then, for our third example, the choice of different values for the K_i and ν_i parameters could lead to a better convergence with less Gauss-Newton iterates. A solution to this problem is to automatically change, during the inversion, the values of these parameters. But this solution still raises 2 new questions : how and when should we modify these parameters ?

A new strategy has been implemented, answering these last questions. A first step is to write (27) as

$$\Delta_{k+1} = \Delta_k * \chi(\rho_k) \tag{28}$$

with χ is defined as a sum of Heaviside functions. In Figure 21, we can observe the variations of confident and cautious χ related to ρ_k. The confident χ function corresponds to the usual parameters. The cautious χ function is obtained by a modification of the usual parameters with the

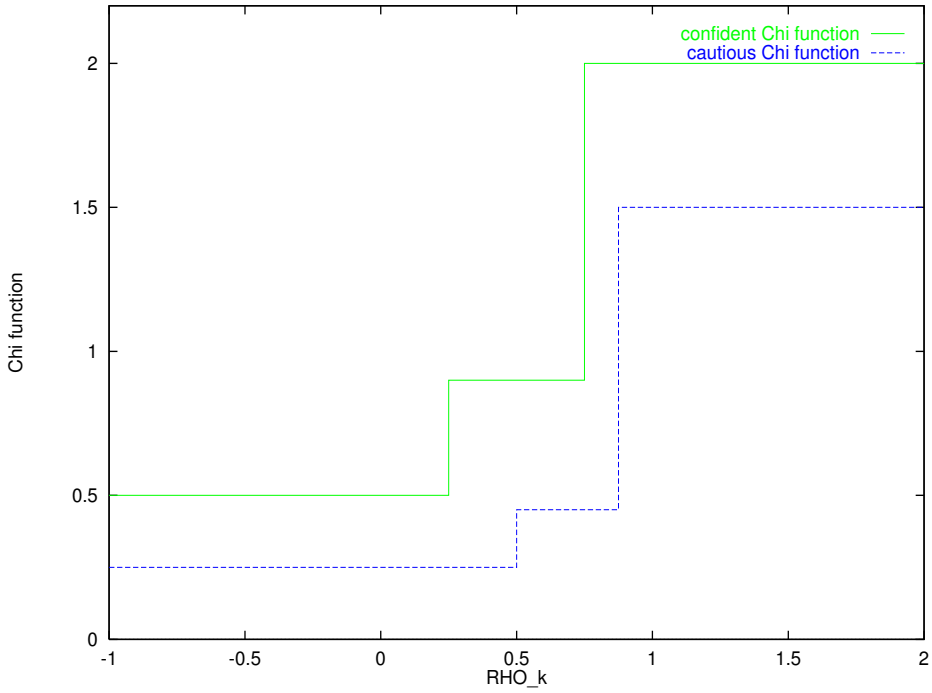

Fig. 21 Confident and cautious Chi function

following rules :

$$\begin{aligned} \nu_1 &= \tfrac{\nu_1}{2} \\ \nu_2 &= \tfrac{\nu_2}{2} \\ \nu_3 &= \tfrac{\nu_3+1}{2} \\ K_2 &= \tfrac{K_2+K_3}{2} \\ K_3 &= \tfrac{K_3+1}{2} \,. \end{aligned} \tag{29}$$

The cautious χ function is then built with the values $K_2 = 0.5$, $K_3 = 0.875$, $\nu_1 = 0.25$, $\nu_2 = 0.45$, and $\nu_3 = 1.5$. The previous rules allow to modify the trust-region parameters. We also notice that the modification (switching from confident to cautious) corresponds to a reduction and a translation of the χ function to the right. The use of χ functions in our trust-region algorithm brings more abstraction and allows modification of the parameters during the inversion (we could now easily create and imagine lots of χ functions). Another problem (the second question) is when we should use the cautious χ function rather than the confident χ function. The χ function is initialized in a confident mode. The mode is switched to cautious when a rejected Gauss-Newton step is encountered at the boundary of the trust-region. Otherwise, the mode is switched to confident if a high success Gauss-Newton step is encountered strictly inside the trust-region (see algorithm 1).

Data : real ρ_k
Result : Choose appropriate strategy to modify the trust-region radius
begin
  logical $Strategy = 1$
  repeat
    if *Gauss-Newton misfit step* ($\rho_k < 0.01$) **and** *last CG perturbation is on the trust-region boundary* $|| p_k || = \Delta_k$ **then**
      $Strategy = 0$
    end
    if *High success Gauss-Newton step* ($\rho_k > 0.75$) **and** *last CG perturbation is not on the trust-region boundary* $|| p_k || < \Delta_k$ **then**
      $Strategy = 1$
    end
    if $Strategy = 1$ **then**
      $\Delta_{k+1} = \Delta_k * \chi_{confident}(\rho_k)$
    else
      $\Delta_{k+1} = \Delta_k * \chi_{cautious}(\rho_k)$
    end
  until *Gauss-Newton convergence*
end

Algorithm 1 Update of the trust-region radius

Save of truncated CG perturbations

When the trust-region algorithm encounters a rejected GN step ($\rho_k < 0.01$), it restarts a truncated CG algorithm with a smaller radius. It is thus interesting to have saved some fallback perturbations in a file during the CG iterates. Each saved perturbation is the solution of the truncated CG algorithm for a given trust-region radius smaller than the actual radius. Hence, we only need to read in a file the corresponding updated trust-region radius. This method saves CPU time associated with CG iterations during inversion (assuming that write and read perturbations in a file is less cost expensive than computing a new CG) and is efficient for unstable examples (when lot of rejected GN steps appear).

In Figure 22, we can observe 3 different perturbations computed during the CG iterates. P is the model perturbation of the current GN step corresponding to a trust-region radius of Δ. $P1$ (resp. $P2$) is the first (resp. second) fallback perturbation corresponding to a half (resp. quater) radius size ($P1 : \Delta/2$, $P2 : \Delta/4$). The radius size of the fallback perturbations is choosen accordingly with the reduction factor ν_1 ($\nu_1 = 0.5$ implies a half reduction of the radius in case of a GN rejected step, see Appendix A for more detail about ν_1). If the first model perturbation P (resp. the first fallback perturbation $P1$) is confronted with a rejected GN step, the algorithm will automatically switch to the first fallback position $P1$ (resp. the second fallback position $P2$) stored in a file without any additional computation.

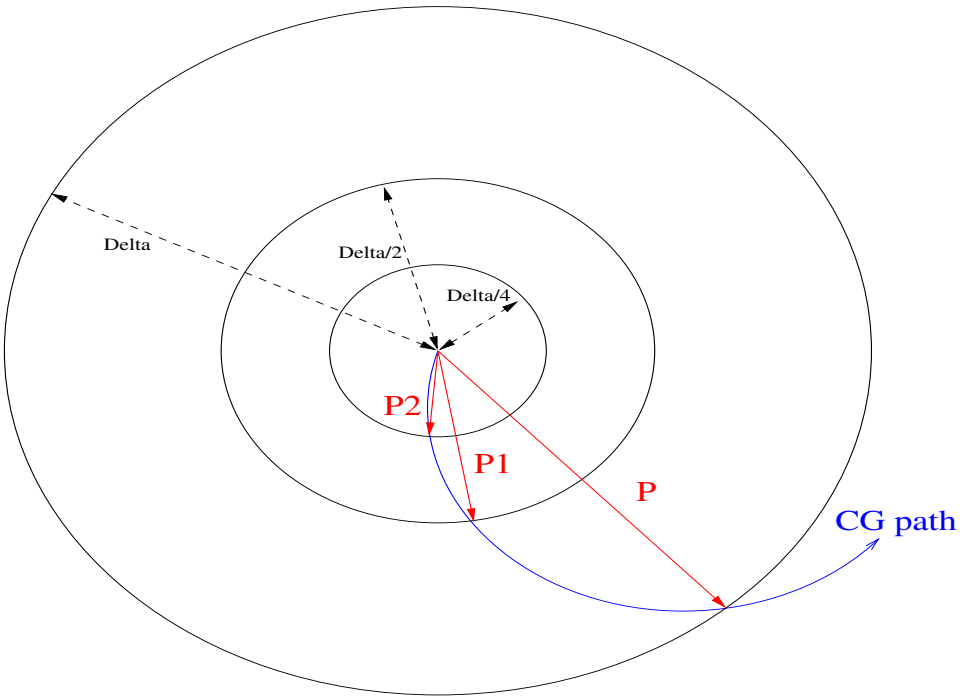

Fig. 22 2D representation of perturbations saved during CG iterations

CONCLUSIONS

In this paper a trust-region method has been introduced and tested for three different applications of 3D reflection tomography. This method provides more stable and accurate results than the line search method. In fact, it better solves intricate examples (like example 3) where the Hessian matrix is strongly ill-conditioned. The global convergence of the Gauss-Newton algorithm is dependant on the initial trust-region radius : a too small radius slows down the global convergence. A large value of the parameter α is then recommended to have a good global convergence ($\alpha \geq 0.8$ for our three examples). Further more, the trust-region problem is automatically solved thanks to a truncated CG, using a block Jacobi preconditioner. If the l_2 norm of the model perturbation diverges during the CG iterations, the algorithm switches to an unpreconditoned method. We observe experimentally the same conclusion as Chauvier, 2000, on the relative residual criterion : solving with a great accuracy the linearized problem is useless ($\varepsilon_1 \geq 10E-5$). We also observe that using the FCR criterion (with $\varepsilon_3 = 1E-4$) enables the algorithm to save CPU time and to gain stability in the inversion.

This study has raised several new problems. First, it can be interesting to better know the influence of the trust-region parameters (modify the trust-region radius updating rule) on the global convergence of the algorithm. Furthermore, new CG preconditioners should be analyzed and implemented in order to better solve the problem of intricate examples (examples with ill-conditioned Hessian matrix). And finally, an essential part of the future work will be the integration of the constraints in this new solver.

ACKNOWLEDGMENTS

This research was carried out as part of the Kinematic Inversion Methods consortium project (KIM). The authors hereby acknowledge the support provided by the sponsors of this project. We would like to particularly thank Andreas Ehinger for the fruitful discussions on optimization methods and tomography.

REFERENCES

Chauvier, L., Masson, R., and Sinoquet, D., 2000, Implementation of an efficient preconditioned conjugate gradient in jerry : KIM Annual Report, Institut Français du Pétrole, Rueil-Malmaison, France.

Chauvier, L., 2000, Commande optimale d'engins sous-marins remorqués avec contraintes : Ph.D. thesis, Université Paris I.

Conn, A. R., Gould, N. I. M., and Toint, P. L., 2000, Trust-region methods : MPS-SIAM Series on Optimization.

Delprat-Jannaud, F., and Lailly, P., 1992, What information on the earth model do reflection travel times hold ? : Journal of Geophysical Research, **97**, 19827–19844.

Delprat-Jannaud, F., and Lailly, P., 1993, Ill-posed and well-posed formulations of the reflection travel time tomography problem : Journal of Geophysical Research, **98**, 6589–6605.

Fletcher, R., 1987, practical methods in optimization : J. Wiley and sons (second edition).

Gill, P., Murray, W., and Wright, M., 1981, Practical optimization : Academic press.

Levenberg, K., 1944, A method for the solution of certain non-linear problems in least squares : Quarterly of Appl. Math., **2**, 164–168.

Marquardt, D. W., 1963, An algorithm for least squares estimation of non-linear parameters : J. Soc. Ind. Appl. Math, **11**, 431–441.

More, J. J., 1978, The Levenberg-Marquardt algorithm : Implementation and theory : Numerical analysis, **630**, 105–116.

Nash, S. G., and Sofer, A., 1990, Assessing a search direction within a truncated-Newton method : orl, **9**, 219–221.

Nocedal, J., and Wright, S. J., 1999, Numerical optimization : Springer.

Renard, F., and Lailly, P., 1999, Robust and accurate determination of complex velocity/depth models by reflection travel time tomography : KIM Annual Report, Institut Français du Pétrole, Rueil-Malmaison, France.

Renard, F., and Lailly, P., 2001, How to handle correlated noise in seismic inversion : 71^{th} Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 690–693.

Saad, Y., 1996, Iterative methods for sparse linear systems : PWS, New York.

Sartenaer, A., 1997, Automatic determination of an initial trust region in nonlinear programming : Soc. Ind. Appl. Math. optimization, **18**, 1788–1803.

Sebudandi, C., and Toint, P. L., 1993, Non linear optimization for seismic traveltime tomography : Geophysics Journal International, **115**, 929–940.

Steihaug, T., 1983, The conjugate gradient method and trust regions in large scale optimization : SIAM Journal on numerical analysis, **20**, 626–637.

APPENDIX A : THE TRUST-REGION ALGORITHM

The trust-region algorithm as implemented in *jerry* is given below. The different constant parameters of the algorithm have also to satisfy the following conditions : $0 < K_1 \leq K_2 < 1$, $0 < \nu_1 \leq \nu_2 \leq 1 \leq \nu_3$

Data : m_0 the initial model, Δ_0 the initial trust-region
begin
  $K_1 = 0.01$ the failure threshold ; $K_2 = 0.25$ the success threshold
  $K_3 = 0.75$ the high success threshold ; $\nu_1 = 0.5$ the fail factor
  $\nu_2 = 0.9$ the success factor ; $\nu_3 = 2.0$ the high success factor
  $k = 0$ Gauss-Newton indices ; Compute $f(m_0)$, the initial value of the objective function (solving the direct problem with a ray tracing method)
  repeat
    ($\longrightarrow$ Gauss-Newton loop)
    Find the solution of the trust-region problem (8), yielding the model perturbation δm_k (Truncated CG algorithm $\longrightarrow$ see Appendix B)
    Evaluate $F_k(\delta m_k)$ the final value of the quadratic model
    Compute $f(\delta m_k + m_k)$ the new value of the objective function
    Compute the concordance ratio ρ_k
    if $\rho_k < K_1$ **then**
      set a smaller trust-region radius length $\Delta_k = \nu_1 * ||\delta m_k||$
    else
      switch *the value of* ρ_k **do**
        case $\rho_k \geq K_3$
          set a larger trust-region radius length $\Delta_{k+1} = \nu_3 * \Delta_k$
        case $K_2 < \rho_k < K_3$
          set a slightly smaller trust-region radius length $\Delta_{k+1} = \nu_2 * \Delta_k$
        case $\rho_k < K_2$
          set a lower trust-region radius length $\Delta_{k+1} = \nu_1 * \Delta_k$
      endsw
      Increment the model $m_{k+1} = \delta m_k + m_k$ Increment $k = k + 1$
    end
  until *Gauss-Newton convergence*
end

Algorithm 2 Trust-region

APPENDIX B : THE TRUNCATED CG ALGORITHM

The truncated CG algorithm is used to find the solution of the trust-region problem (8). This corresponds to the minor iteration of the trust-region algorithm described in appendix A. We give below the truncated CG algorithm without preconditioning as implemented in *jerry* .

Data : $g(m)$ the gradient, H the Hessian matrix in m
begin
  $i = 0$ CG indices
  $\delta m_i = 0$
  $r_i = g(m)$
  $p_i = -r_i$
  repeat
    if $i > 0$ **then**
      $\beta_i = r_i^T r_i / r_{i-1}^T r_{i-1}$
      $p_i = r_i + \beta_i p_{i-1}$
    end
    if $p_i^T H p_i \leq 0$ *(should not appear in our convex applications)* **then**
      Find $\sigma \geq 0$ such that $\delta m = \delta m_i + \sigma * p_i$ satisfies $||\,\delta m\,|| = \Delta$
      $return\ \ \delta m$
    end
    $\alpha_i = r_i^T r_i / p_i^T H p_i$
    $\delta m_{i+1} = \delta m_i + \alpha_i p_i$
    $r_{i+1} = r_i - \alpha_i H p_i$
    if $||\,\delta m_{i+1}\,|| \geq \Delta$ **then**
      Find $\sigma \geq 0$ such that $\delta m = \delta m_i + \sigma * p_i$ satisfies $||\,\delta m\,|| = \Delta$
      $return\ \ \delta m$
    end
    $i = i + 1$
  until *CG convergence*
end

Algorithm 3 CG

APPENDIX C : THE BACKTRAKING LINE SEARCH ALGORITHM

The backtraking algorithm as implemented in *jerry* is given below. λ is updated while the Armijo condition is not satisfied.

Data : $f(m_k)$ the non-linear objective function in m_k, $\nabla f_k^T \delta m$ the directional derivate in m_k along δm
begin
  $C = 10^{-4}$ a constant parameter
  $\kappa = 0.5$ the contraction factor ($\kappa \in (0, 1)$)
  $\lambda = 1.0$ the initial step length
  repeat
    $\lambda = \kappa\lambda$
  until $f(m_k + \lambda_k.\delta m) \leq f(m_k) + C.\lambda_k.\nabla f_k^T \delta m)$
  Terminate with $\lambda_k = \lambda$
end

Algorithm 4 Backtraking line search algorithm

Deuxième partie

Résolution du problème inverse de tomographie de réflexion : optimisation avec contraintes

Dans cette partie nous étudions la résolution du problème inverse de la tomographie de réflexion avec contraintes. En optimisation, ce problème de tomographie peut être vu comme un problème de moindres-carrés non-linéaire. Une méthode de programmation quadratique successive (SQP) gauss-newtonienne qui décompose le problème de moindres-carrés non-linéaire en une suite de problèmes quadratiques convexes a été choisie. A chaque itération de Gauss-Newton, le problème quadratique est résolu par une méthode de relaxation lagrangienne augmentée. Cette méthode transforme la résolution du problème quadratique sous contraintes linéaires en une suite de problèmes quadratiques sous contraintes de borne. Les chapitres de cette partie sont classés suivant les niveaux de décomposition de la méthode d'optimisation : du niveau de détail le moins élevé au niveau de détail le plus élevé. Dans la figure 3.1, nous avons illustré les 3 grands niveaux de détail dans cette méthode :

- Résolution d'un problème de moindres-carrés non-linéaire avec contraintes linéaires
  - Résolution d'un problème quadratique avec contraintes linéaires
    - Résolution d'un problème quadratique avec contraintes de borne

Indépendamment de la méthode d'optimisation choisie pour résoudre notre problème de tomographie de réflexion avec contraintes, nous donnons dans le chapitre 4 les résultats et concepts d'optimisation nécessaires pour la compréhension des chapitres suivants.

Dans les deux chapitres 5 et 6, nous abordons le premier niveau d'analyse de la méthode d'optimisation : le problème général de la tomographie de réflexion avec contraintes y est posé. Nous y expliquons les différentes méthodes possibles pour résoudre ce problème de moindres-carrés non-linéaire avec contraintes linéaires et nous choisissons de développer plus particulièrement la méthode SQP.

Le choix en amont de la méthode SQP implique la résolution d'une suite de problèmes quadratiques avec contraintes linéaires appelés Problème Quadratique Tangent (PQT). Dans le chapitre 7, nous abordons le second niveau de décomposition de la méthode d'optimisation en faisant le point sur les méthodes de résolution possibles du PQT et en développant plus particulièrement la méthode du lagrangien augmenté (LA).

Le choix en amont de la méthode du LA pour résoudre le PQT implique la résolution d'une suite de problèmes quadratiques avec contraintes de borne appelés Problème de Lagrange (PL). Les chapitres 8 et 9 sont dédiés au troisième et dernier niveau d'analyse de la méthode d'optimisation : ils traitent respectivement de la méthode de Gradient avec Projection - Activation de Contraintes - Gradient Conjugué (GP-AC-GC) pour résoudre le PL et des différentes méthodes de minimisation de la fonction duale régularisée.

Enfin le dernier chapitre (chapitres 10) s'intéresse à l'automatisation de l'ensemble de la méthode décrite précédemment (réglage du paramètre d'augmentation r du LA).

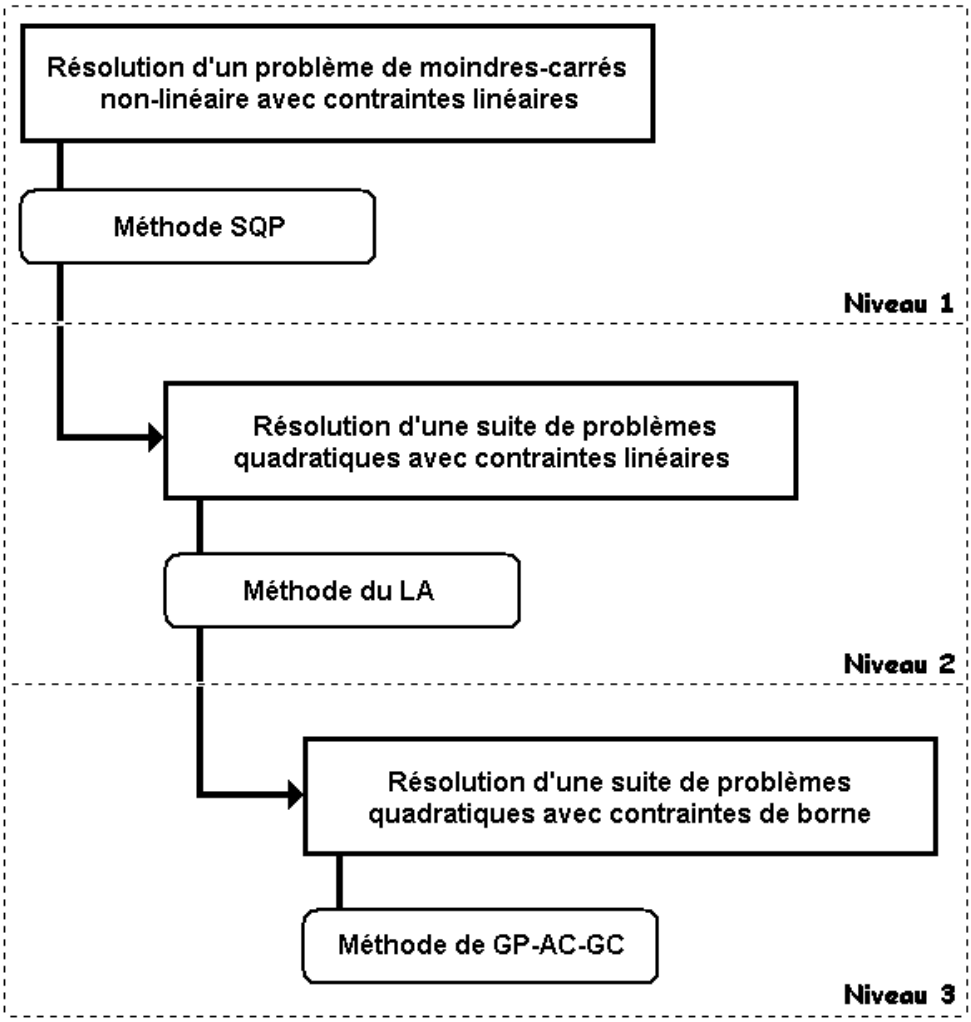

Fig. 3.1 Schéma général montrant les trois grands niveaux de détail de la méthode d'optimisation

Chapitre 4

Optimisation avec contraintes : résultats généraux

Avant de rentrer plus en détail dans le sujet, nous donnons dans ce premier chapitre les concepts nécessaires à la compréhension des chapitres suivants. Ces concepts sont expliqués plus profondément dans les livres suivants : [57], [58], [123],[68] et [18].

4.1 Problème d'optimisation avec contraintes d'égalité et d'inégalité

4.1.1 Formulation du problème

Considérons le problème d'optimisation

$$(P)\begin{cases}\min\limits_{x} f(x)\\ c_i(x)=0, \quad i\in E\\ c_i(x)\leq 0, \quad i\in I,\end{cases} \tag{4.1}$$

dans lequel on minimise une fonction $f:\Omega\mapsto\mathbb{R}$ (avec Ω un ouvert de $\mathbb{R}^n$) et où E et I forment une partition de $\{1,...,m\}$ ($E\cup I=\{1,...,m\}$ et $E\cap I=\emptyset$). Les contraintes sont définies par la fonction $c:\mathbb{R}^n\mapsto\mathbb{R}^m$. Si $v\in\mathbb{R}^m$, on note v_E (resp. v_I) le vecteur de $\mathbb{R}^{|E|}$ (resp. $\mathbb{R}^{|I|}$) formé des composantes v_i de v avec $i\in E$ (resp. $i\in I$). De même, on notera c_E (resp. c_I) la fonction définissant les contraintes d'égalité (resp. les contraintes d'inégalité).

Définition 4.1.1 *On appelle ensemble admissible du problème (P), l'ensemble $\mathcal{X}$ défini par*

$$\mathcal{X}:=\{x\in\Omega : c_E(x)=0, c_I(x)\leq 0\}.$$

Définition 4.1.2 *On appelle minimum (global) de (P) tout point $x_* \in \mathcal{X}$ vérifiant*

$$f(x_*) \leq f(x), \quad \forall x \in \mathcal{X}.$$

Définition 4.1.3 *On dit que $x_* \in \mathcal{X}$ est un minimum local de (P) s'il existe un voisinage V de x_* tel que*

$$f(x_*) \leq f(x), \quad \forall x \in \mathcal{X} \cap V.$$

Définition 4.1.4 *On dit que c_i ($i \in I$) est active en x si $c_i(x) = 0$. On note*

$$I^0(x) := \{i \in I : c_i(x) = 0\}$$

l'ensemble des indices des contraintes d'inégalité actives en x, on note également $I^0_* := I^0(x_*)$.

Définition 4.1.5 *On dit qu'un problème d'optimisation est convexe si sa fonction coût et ses contraintes d'inégalité sont convexes et si ses contraintes d'égalité sont linéaires.*

4.1.2 Conditions nécessaires d'optimalité du premier ordre de Karush-Kuhn-Tucker

Les conditions nécessaires d'optimalité du problème d'optimisation (P) sont une réunion d'équations et/ou d'inéquations et/ou de propriétés que vérifient les solutions de (P). On parle de conditions du premier ordre lorsque celles-ci ne font intervenir que les dérivées premières de f et de c.

Définition 4.1.6 *On dit que $d \in \mathbb{R}^n$ est tangent à $\mathcal{X}$ en x s'il existe une suite $\{d_k\} \subset \mathbb{R}^n$ et une suite $\{t_k\} \subset \mathbb{R}^*_+ = \{t \in \mathbb{R} : t > 0\}$ telles que*

$$d_k \to d, \;\; t_k \to 0, \;\; x + t_k d_k \in \mathcal{X}.$$

On note $T_x\mathcal{X}$ l'ensemble des vecteurs tangents à $\mathcal{X}$ en x et on l'appelle le cône tangent.

Définition 4.1.7 *On appelle cône linéarisant $T'_x\mathcal{X}$ de $\mathcal{X}$ en x l'ensemble*

$$T'_x\mathcal{X} := \{d \in \mathbb{R}^n : \;\; c'_E(x) \cdot d = 0, \;\; c'_{I^0(x)}(x) \cdot d \leq 0\}.$$

Définition 4.1.8 *On dit que les contraintes sont qualifiées en $x \in \mathcal{X}$ si*

$$(QC) \;\; T_x\mathcal{X} = T'_x\mathcal{X}$$

Définition 4.1.9 *On dit que les contraintes vérifient la condition de qualification (QC-A) en $x \in \mathbb{R}^n$ lorsque $c_{E \cup I^0(x)}$ est affine dans un voisinage de x*

Les propositions 4.1.10 et 4.1.11 sont reprises des propositions 3.25 et 3.21 de [68].

Proposition 4.1.10 *Soit x vérifiant les contraintes de (P). Supposons que $c_{E\cup I^0(x)}$ sont dérivable en x et que $c_{I\setminus I^0(x)}$ est continue en x. Si la condition de qualification des contraintes (QC-A) est satisfaite en x alors les contraintes sont qualifiées en x.*

Théorème 4.1.11 *Soit x_* un minimum local de (P), supposons que f et $c_{E\cup I^0_*}$ sont dérivables en x_* et que les contraintes soient qualifiées en x_*. Alors, il existe un multiplicateur $\lambda_* \in \mathbb{R}^m$ tel que l'on ait*

$$(KKT) \quad \begin{cases} (a)\ \nabla f(x_*) + A(x_*)^\top \lambda_* = 0 \\ (b)\ c_E(x_*) = 0,\ c_I(x_*) \leq 0 \\ (c)\ (\lambda_*)_I \geq 0 \\ (d)\ (\lambda_*)_I^\top c_I(x_*) = 0, \end{cases} \tag{4.2}$$

où ∇ représente le gradient associé au produit scalaire euclidien, et où $A(x)$ est la matrice jacobienne de c en x ($A(x) = \nabla c(x)^\top$)

Quelques remarques sur le système d'optimalité (4.1.11)

1. Le système (4.2) est connu sous le nom de conditions de Karush, Kuhn et Tucker ou (KKT).
2. L'équation (a) peut aussi s'écrire
$$\nabla_x L_{(P)}(x_*, \lambda_*) = 0,$$
avec $L_{(P)}$ le lagrangien associé au problème (P) :
$$L_{(P)}(x, \lambda) = f(x) + \lambda^\top c(x). \tag{4.3}$$
3. Le vecteur λ_* est appelé multiplicateur de Lagrange associé aux contraintes c et à la solution x_*.
4. Un couple (x_*, λ_*) vérifiant (4.2) est appelé solution primale-duale de (P) : x_* est la solution primale et λ_* la solution duale.
5. Un point x_* pour lequel il existe $\lambda_* \in \mathbb{R}^m$ tel que (4.2) est vérifiée est dit stationnaire.
6. Dans la condition (b), on reconnaît la condition d'admissibilité des contraintes en x_* ($x_* \in \mathcal{X}$).
7. Les conditions (c) et (d) concernent uniquement les contraintes d'inégalité. La condition (c) est la condition de signe constant des multiplicateurs et la condition (d) s'appelle la condition de complémentarité. La condition (d) peut aussi se reformuler composante par composante (en notant que l'on a à la fois $(\lambda_*)_I \geq 0$ et $c_I(x_*) \leq 0$) :
$$(\lambda_*)_i c_i(x_*) = 0, \quad \forall i \in I.$$
Cette dernière expression implique que le multiplicateur correspondant à une contrainte inactive est nul :
$$c_i(x_*) < 0 \quad \Longrightarrow \quad (\lambda_*)_i = 0.$$

Définition 4.1.12 *On dit que la condition de complémentarité stricte est satisfaite en une solution primale-duale (x_*, λ_*) de (P) lorsque $\forall i \in I$:*

$$c_i(x_*) < 0 \quad \Longleftrightarrow \quad (\lambda_*)_i = 0.$$

Voici ci-dessous une condition suffisante d'optimalité du premier ordre reprise de la proposition 3.22 de [68].

Proposition 4.1.13 *Considérons le problème (P) et supposons que ce problème est convexe (voir définition 4.1.5). Soit x_* un point vérifiant les contraintes de (P). Si f et c sont dérivables en x_* et s'il existe $\lambda_* \in \mathbb{R}^m$ tel que les conditions de KKT (4.2) soient vérifiées, alors x_* est un minimum global de (P).*

4.1.3 Conditions suffisantes d'optimalité du second ordre faible

Définition 4.1.14 *On appelle cône critique $\mathcal{C}_*$ du problème (P) en x_* l'ensemble*

$$\mathcal{C}_* = \{d \in \mathbb{R}^n : \quad c_E(x_*).d = 0, \quad c_{I_*^0}(x_*).d \leq 0, \quad f'(x_*).d \leq 0\}.$$

On rappelle ci-dessous la condition suffisante d'optimalité du second ordre faible reprise de la proposition 11.3 de [18] :

Théorème 4.1.15 *On suppose que f et $c_{E \cup I_*^0}$ soient différentiables dans un voisinage de $x_* \in \Omega$ et deux fois différentiables en x_*. On suppose également que l'ensemble Λ_* des multiplicateurs de Lagrange λ_* satisfaisant les conditions de (KKT) n'est pas vide et que*

$$\forall d \in \mathcal{C}_* \backslash \{0\}, \quad \exists \lambda_* \in \Lambda_* : \quad d^\top \nabla^2_{xx} L_{(P)}(x_*, \lambda_*) d > 0.$$

Alors x_* est un minimum local strict de (P).

4.1.4 Calcul effectif des solutions de (P)

On se sert des conditions nécessaires d'optimalité de (KKT) pour calculer les solutions de (P) et mettre en œuvre des méthodes numériques permettant de résoudre (P). La résolution du système d'optimalité (4.2) est compliquée du fait de la présence des contraintes d'inégalité qui génèrent les conditions de complémentarité (d).

Pour les problèmes de très petite taille, on pourra explorer l'ensemble des possibilités offertes pour satisfaire les contraintes d'inégalité. Une contrainte d'inégalité (c_i tel que $i \in I$) peut soit être active en x_* et on a $c_i(x_*) = 0$, soit être inactive et on a $c_i(x_*) < 0$. On explorera donc des ensembles d'indices $J \subset I$ dont les contraintes d'inégalité sont supposées actives en x_* : $c_{E \cup J}(x_*) = 0$ et $c_{J^c}(x_*) < 0$ ($J^c = I \backslash J$). En utilisant les équations de complémentarité $(\lambda_*)_{J^c} = 0_{J^c}$ et le système d'optimalité (4.2) se simplifie par le système de $n + |E \cup J|$ équations à $n + |E \cup J|$ inconnues :

$$\begin{cases} \nabla f(x_*) + A_{E \cup J}(x_*)^\top (\lambda_*)_{E \cup J} = 0 \\ c_{E \cup J}(x_*) = 0. \end{cases}$$

On recherche alors les solutions $(x_*, (\lambda_*)_{E\cup J})$ de ce système d'équations. Si une des solutions vérifie aussi les hypothèses du cas considéré ($c_{J^c}(x_*) < 0$ et $(\lambda_*)_J \geq 0$) alors le couple $(x_*, ((\lambda_*)_{E\cup J}, 0_{J^c}))$ est une solution primale-duale de (4.2). Cette méthodologie permet de trouver tous les points stationnaires du problème (P) en explorant tous les ensembles d'indices $J \subset I$ possibles.

Cette méthode n'est applicable que sur des problèmes de petite taille. En effet, le nombre d'ensemble J à explorer augmente exponentiellement avec le nombre de contraintes d'inégalité du problème (P). Pour m_I contraintes d'inégalité, il faudrait explorer 2^{m_I} ensembles d'indices J possibles. On notera par ailleurs, que chaque exploration d'un ensemble d'indices J nécessite la résolution d'un système non-linéaire. Ce phénomène, rencontré uniquement lors de la résolution de problème d'optimisation incluant des contraintes d'inégalité, est appelé la "combinatoire" des problèmes d'optimisation. Ainsi, les algorithmes d'optimisation qui sont mis en œuvre pour résoudre le problème (P) s'attachent à gérer de manière efficace cette combinatoire.

Dans le cas convexe, on donne la proposition ci-dessous qui est reprise de la proposition 11.7 de [68].

Proposition 4.1.16 *On suppose que les contraintes c_i de (P) sont linéaires. Soit x_* une solution du problème (P) et x_J une solution du problème*

$$(P)_J \begin{cases} \min\limits_x f(x) \\ c_i(x) = 0, \quad i \in E \cup J, \end{cases}$$

dans lequel $J = I^0(x_*)$.

(i) Si f est convexe sur $c'_{E\cup J}$ et $x_J \in \mathcal{X}$, alors x_J est solution de (P).

(ii) Si f est strictement convexe sur $c'_{E\cup J}$, alors $x_J = x_*$

4.2 Aspects algorithmiques

4.2.1 Méthodes de pénalisation

Historiquement, les méthodes de pénalisation sont les toutes premières méthodes qui sont apparues pour résoudre le problème (P). Elles consistent à transformer un problème d'optimisation avec contraintes en un problème (ou une suite de problèmes) d'optimisation sans contrainte. Les différentes méthodes de pénalisation suivent généralement le principe suivant : le problème original (P_{EI}) est remplacé par un ou des problèmes sans contrainte de la forme

$$(\widetilde{P}_\sigma) : \quad \min_x \Theta_\sigma(x),$$

où

$$\Theta_\sigma(x) := f(x) + \sigma p(x),$$

est la fonction de pénalisation, p est la fonction pénalisant les contraintes et $\sigma > 0$ est le facteur de pénalisation. On observe que les contraintes de (P) ont été remplacées par le

terme additionnel $\sigma p(x)$ à minimiser dans la fonction objectif de $(\widetilde{P}_\sigma)$. L'une des questions essentielles est de savoir si en résolvant $(\widetilde{P}_\sigma)$ on résout bien (P). La réponse à cette question dépend du choix de la fonction p et du facteur de pénalisation σ, et conduit à la notion de pénalisation exacte :

Définition 4.2.1 *On dit que Θ_σ est une fonction de pénalisation exacte en un minimum local x_* de (P) si x_* est un minimum local de Θ_σ.*

Parmi les différentes méthodes de pénalisation on distingue 2 grandes sous-classes de méthodes :

* méthodes de pénalisation extérieure,
* méthodes de pénalisation intérieure.

Dans les méthodes de pénalisation extérieure la fonction p pénalise la violation des contraintes. Elle doit respecter les propriétés suivantes :

$$\begin{cases} (i) & p \text{ est continue sur } \mathbb{R}^n \\ (ii) & p(x) \geq 0, \forall x \in \mathbb{R}^n \\ (iii) & p(x) = 0 \Longleftrightarrow x \in \mathcal{X}. \end{cases}$$

Un exemple simple et intuitif de fonction de pénalisation extérieure est la fonction de pénalisation quadratique des contraintes (fonction utilisée par [40]). Cette fonction qui est à la base des méthodes de lagrangien augmenté (voir section 4.2.2) s'écrit :

$$\Theta_\sigma(x) = f(x) + \frac{1}{2\sigma}(||c_E(x)||_2^2 + ||c_I(x)^+||_2^2), \tag{4.4}$$

où $(v^+)_i = \max(v_i, 0)$. La méthode de pénalisation consiste alors à minimiser la fonction Θ_σ précédente pour une suite décroissante de valeurs de σ jusqu'à ce que les conditions d'optimalité de (P) (équation (4.2)) soient respectées à une précision donnée. Le talon d'Achille de cette méthode est que la minimisation de la fonction Θ_σ est de plus en plus difficile à effectuer lorsque σ diminue. Le mauvais conditionnement du hessien $\nabla^2_{xx}\Theta_\sigma$ proche de la solution de (P) est la source de cette difficulté. La méthode du lagrangien augmenté, construite sur les bases de la pénalisation quadratique (voir section 4.2.2), permet en général d'obtenir de meilleurs résultats en éliminant ce mauvais conditionnement.

Dans les méthodes de pénalisation intérieure la fonction p pénalise l'abord de la frontière du domaine admissible. Elle doit respecter les propriétés suivantes : (on note $\mathcal{X}^0$ l'intérieur de $\mathcal{X}$, $\partial\mathcal{X}$ la frontière de $\mathcal{X}$ et on suppose que $\mathcal{X}^0 \neq \emptyset$)

$$\begin{cases} (i) & p \text{ est continue sur } \mathcal{X}^0 \\ (ii) & p(x) \geq 0, \forall x \in \mathcal{X}^0 \\ (iii) & p(x) \to +\infty, \text{ quand } x \in \mathcal{X}^0 \to \partial\mathcal{X}. \end{cases}$$

On parle aussi de méthodes (resp. fonction) "barrière" pour évoquer les méthodes (resp. fonction) de pénalisation intérieure : la condition (iii) crée une barrière au bord de l'ensemble admissible rendant les solutions de $(\widetilde{P}_\sigma)$ admissibles (ces solutions restent dans

$\mathcal{X}^0$). L'une des méthodes de pénalisation intérieure les plus connues est la pénalisation logarithmique (les méthodes de pénalisation logarithmique remontent à [65] et [55]) qui est à la base des algorithmes de points intérieurs (voir section 4.2.4). Pour les contraintes d'inégalité seulement, cette fonction s'écrit :

$$\Theta_\sigma(x) = f(x) - \sigma \sum_{i \in I} \log(-c_i(x)) \tag{4.5}$$

La méthode de pénalisation consiste alors à minimiser la fonction Θ_σ précédente pour une suite décroissante de valeurs de σ jusqu'à ce que les conditions d'optimalité de (P) (équation (4.2)) soient respectées à une précision donnée. Similairement au cas de la pénalisation quadratique, la minimisation de la fonction Θ_σ est de plus en plus difficile à effectuer lorsque σ diminue : le mauvais conditionnement du hessien $\nabla^2_{xx}\Theta_\sigma$ proche de la solution de (P) en est la cause. Afin de prendre aussi en compte les contraintes d'égalité de (P), on peut utiliser la fonction de pénalisation mixte suivante :

$$\Theta_\sigma(x) = f(x) + \frac{1}{2\sigma}||c_E(x)||_2^2 - \sigma \sum_{i \in I} \log(-c_i(x)). \tag{4.6}$$

Cette fonction combine les techniques de pénalisation quadratique et logarithmique : les contraintes d'égalité de (P) sont pénalisées par la fonction de pénalisation quadratique (fonction (4.4)), alors que les contraintes d'inégalité sont pénalisées par la fonction de pénalisation logarithmique (fonction (4.5)). Les propriétés de convergence de la méthode de pénalisation avec cette fonction mixte ont été mises en valeur dans [83].

4.2.2 Méthodes de lagrangien augmenté

A partir des années 70, d'intensives recherches sont menées sur les méthodes de lagrangien augmenté. Ces méthodes découlent directement de la méthode de pénalisation quadratique des contraintes (décrite dans la section précédente) : elles combinent les propriétés du lagrangien de (P) (voir équation (4.3)) et de la fonction de pénalisation quadratique des contraintes (équation (4.4)). L'approche classique par lagrangien augmenté (LA) pour les contraintes d'égalité remonte à [93] et [126] (la notion de LA avait déjà été proposée par [5]). La généralisation aux contraintes d'inégalité est attribuée à [133, 136] et [127].

Le *lagrangien augmenté* associé au problème (4.1) est la fonction $l_r : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, définie pour $(x, \lambda) \in \mathbb{R}^n \times \mathbb{R}^m$ et $r \in \mathbb{R}_{++} := \{t \in \mathbb{R} : t > 0\}$ par

$$\begin{aligned} l_r(x,\lambda) \;=\; & f(x) + \lambda_E^\top c_E(x) + \frac{r}{2}\|c_E(x)\|_2^2 \\ & + \sum_{i \in I} \left(\lambda_i \max\left(\frac{-\lambda_i}{r}, c_i(x)\right) + \frac{r}{2}\left[\max\left(\frac{-\lambda_i}{r}, c_i(x)\right)\right]^2 \right). \end{aligned} \tag{4.7}$$

On voit que, pour les contraintes d'inégalité, c'est $\max\left(\frac{-\lambda_i}{r}, c_i(x)\right)$ qui joue le rôle de $c_i(x)$.

L'approche de Powell, Hestenes et Rockafellar, appelée *méthode des multiplicateurs*, suit le paradigme de la *relaxation lagrangienne* (les spécialistes de cette "discipline" prennent soin des questions de non différentiabilité de la fonction duale associée qui sont ignorées dans la méthode des multiplicateurs, voir par exemple [94, chapitre XII] ou plus récemment [107] pour un domaine d'application plus éloigné du nôtre). Au début de l'itération k, on suppose disposer d'un multiplicateur approché $\lambda_k \in \mathbb{R}^m$ et d'un facteur de pénalisation $r_k \in \mathbb{R}_{++}$. On minimise alors $l_{r_k}(x, \lambda_k)$ par rapport à x, ce qui donne x_k ; puis on met à jour λ_k et/ou r_k par des formules appropriées. Lorsque la mise à jour de λ_k s'impose, on utilise

$$\lambda_{k+1} = \Big(\lambda_k + r_k c(x_k)\Big)^{\#}.$$

On a défini l'opérateur $(\cdot)^{\#} : v \in \mathbb{R}^m \mapsto v^{\#} \in \mathbb{R}^m$ par

$$(v^{\#})_i = \begin{cases} v_i & \text{si } i \in E \\ v_i^+ = \max(v_i, 0) & \text{si } i \in I. \end{cases}$$

Au départ cette formule était vue comme une heuristique (c'est comme cela qu'elle est présentée dans [123]), puis Rockafellar [135] lui a donné du sens : pour les problèmes convexes, c'est une *méthode proximale* sur la fonction duale. Les formules de mise à jour de r_k sont moins bien motivées.

En comparant les méthodes de LA aux méthodes de pénalisation (extérieure ou intérieure) (voir section précédente), on se rend compte que ces 2 méthodes procèdent d'une manière quasi-similaire : elles transforment la résolution du problème (P) en la résolution d'une suite de sous-problèmes non-linéaires sans contrainte. La différence entre ces 2 méthodes réside principalement dans l'introduction (dans l'algorithme du LA) d'une estimation de la valeur des multiplicateurs de Lagrange optimaux. Cette introduction des multiplicateurs de Lagrange est essentielle car elle permet en général de faire disparaître les problèmes de mauvais conditionnement rencontrés dans les méthodes de pénalisation. Contrairement aux méthodes de pénalisation où l'on doit faire tendre le paramètre σ vers 0, dans les méthodes de LA il n'est pas nécessaire de faire tendre le paramètre d'augmentation r vers $+\infty$ pour s'assurer que le minimum de $l_r(\cdot, \lambda_*)$ soit une solution de (P).

La méthode des multiplicateurs n'est qu'un schéma d'algorithme, car on ne dit pas comment on minimise $l_{r_k}(\cdot, \lambda_k)$. Or cette fonction n'est pas deux fois différentiable (on montre qu'elle est $C^{1,1}$ dans le voisinage d'une solution pour des données régulières, voir par exemple [18]), ce qui exclut l'utilisation de Newton ou Gauss-Newton. Depuis l'introduction du LA (4.7), on a proposé des LAs qui sont C^2 (ce sont souvent des classes de fonctions qui à notre connaissance n'ont pas été validées sur des applications de grandes tailles). On peut utiliser [7] et [6] comme points d'entrée sur les travaux récents dans cette direction (voir aussi les travaux de [78]).

Beaucoup de travaux ont été réalisés sur les méthodes de LA. L'application de la méthode du LA sur les problèmes d'optimisation non-linéaire s'est concrétisée par le développement du code LANCELOT (voir [37]) . L'approche utilisée dans LANCELOT ne fait intervenir le LA que pour les contraintes d'égalité et traite les contraintes de borne

explicitement par activation de contraintes et projection. Cela peut être vu comme un moyen d'éviter les problèmes de "non différentiabilité C^2" de $l_{r_k}(\cdot, \lambda_k)$. L'application de la méthode du LA à la résolution numérique de problèmes aux limites a été développée dans [62] et voir aussi les travaux plus récents de [74].

4.2.3 Méthode de programmation quadratique successive

On trouve les premières traces de la méthode SQP dans [156]. Cependant, il faut attendre le milieu des années 70 pour voir le développement de cette méthode (voir [124], [86], [87], [131], [129], [130]). Dans cette section, on décrit la méthode de programmation quadratique successive (SQP : "Sequential Quadratic Programming") appliquée à la minimisation du problème (P). S'apparentant aux algorithmes newtoniens, la méthode SQP est actuellement l'une des méthodes les plus efficaces pour résoudre les problèmes d'optimisation non-linéaires. Elle consiste à transformer la résolution d'un problème d'optimisation non-linéaire en la résolution d'une suite de problèmes d'optimisation quadratique. Ceux-ci sont obtenus par linéarisation des conditions d'optimalité (4.2) au point (x_k, λ_k) courant. Le système d'optimalité linéarisé a pour inconnues le déplacement (d, μ) à apporter à (x_k, λ_k) :

$$\begin{cases} M_k d + A_k^\top \mu = -\nabla_x L_k \\ (c_k)_E + (A_k)_E d = 0, \quad (c_k)_I + (A_k)_I d \leq 0 \\ (\lambda_k + \mu)_I \geq 0 \\ (\lambda_k + \mu)_I^\top (c_k)_I + (\lambda_k)_I^\top (A_k)_I d = 0, \end{cases} \tag{4.8}$$

avec les notations $c_k := c(x_k)$ ($(c_k)_E := c_E(x_k)$ et $(c_k)_I := c_I(x_k)$), $A_k = c'(x_k)$ la matrice jacobienne de c, $\nabla_x L_k := \nabla_x L_{(P)}(x_k, \lambda_k)$ et $M_k := \nabla^2_{xx} L_{(P)}(x_k, \lambda_k)$ le hessien du lagrangien en (x_k, λ_k). On peut obtenir à partir du système (4.8) un problème plus facilement résoluble en ajoutant le terme $(\mu)_I^\top (A_k)_I d$ dans la dernière équation (ce terme est négligeable lorsque le déplacement (d, μ) est petit, ce qui est le cas lorsque (x_k, λ_k) est proche d'une solution de (P)). Sur ce système linéarisé modifié, on réalise le changement de variable $\lambda^{QP} := \lambda_k + \mu$, ce qui nous donne :

$$\begin{cases} M_k d + A_k^\top \lambda^{QP} = -\nabla f_k \\ (c_k)_E + (A_k)_E d = 0, \quad (c_k)_I + (A_k)_I d \leq 0 \\ (\lambda^{QP})_I \geq 0 \\ (\lambda^{QP})_I^\top ((c_k)_I + (A_k)_I d) = 0, \end{cases} \tag{4.9}$$

où $\nabla f_k = \nabla f(x_k)$ est le gradient de f en x_k. On peut facilement vérifier que le système (4.9) est le système d'optimalité du problème quadratique suivant :

$$\begin{cases} \min_d \nabla f(x_k)^\top d + \frac{1}{2} d^\top M_k d \\ c_E(x_k) + (A_k)_E d = 0 \\ c_I(x_k) + (A_k)_I d \leq 0. \end{cases} \tag{4.10}$$

Le problème quadratique (4.10) s'obtient facilement à partir du problème initial (P) :

Remarques 4.2.2

1. *Les contraintes de* (4.10) *s'obtiennent en linéarisant les contraintes de* (P) *en* x_k.
2. *La fonction objectif de* (4.10) *est hybride, avec le gradient de* f *pour la partie linéaire et le hessien du lagrangien pour la partie quadratique.*
3. *Rappellons que le lagrangien de* (P) *s'écrit*

$$L_{(P)}(x, \lambda) = f(x) + \lambda^\top c(x).$$

La matrice M_k *est égale au hessien du Lagrangien c'est à dire :*

$$M_k = \nabla^2_{xx} f(x) + \lambda^\top \nabla^2_{xx} c(x).$$

On remarque dans cette dernière équation que le calcul de M_k *fait intervenir deux termes différents qui sont le hessien de* f *et la courbure des contraintes. Une conséquence directe de cette observation est que dans le cas de contraintes linéaires (*$\nabla^2_{xx} c(x) = 0$*) le hessien du lagrangien est égal au hessien de* f.

Définition 4.2.3 *Le problème d'optimisation* (4.10) *est appellé Problème Quadratique Tangent (PQT) de* (P) *en* x_k.

La méthode SQP consiste alors à produire une suite de points $\{(x_k, \lambda_k)\}$ qui converge vers une solution primale-duale (x_*, λ_*) de (P). En chaque point (x_k, λ_k), on recherche une solution primale-duale (d_k, λ_k^{QP}) du problème quadratique tangent (4.10). Une fois cette solution obtenue il ne reste plus qu'à calculer le nouveau point (x_{k+1}, λ_{k+1}) par :

$$x_{k+1} = x_k + d_k \;\text{ et }\; \lambda_{k+1} = \lambda_k^{QP}.$$

Plus généralement, on parlera de méthode SQP, lorsque la matrice M_k définie dans le PQT (4.10) est seulement une approximation du hessien du lagrangien ($M_k \approx \nabla^2_{xx} L_{(P)}(x_k, \lambda_k)$). Il est normal de choisir pour M_k une matrice symétrique (semi) définie positive. On peut utiliser l'algorithme SQP local suivant pour résoudre (P) :

Data : Un itéré initial est donné : (x_1, λ_1).
begin
  $k = 1$.
  Calculer $c(x_1)$, $\nabla f(x_1)$ et $A(x_1)$.
  while (4.2) *n'est pas satisfaite* **do**
    Calculer M_k et trouver une solution primale-duale de (4.10), i.e., une solution (d_k, λ_k^{QP}) de (4.9).
    Calculer le nouveau point :

$$x_{k+1} = x_k + d_k \quad \text{et} \quad \lambda_{k+1} = \lambda_k^{QP}.$$

    Calculer $c(x_k)$, $\nabla f(x_k)$ et $A(x_k)$.
    Accroître k de 1 : $k := k + 1$.
  endw
end

Algorithme II.1 Algorithme SQP local

Nous noterons que la résolution de (4.9) est l'étape coûteuse de l'algorithme SQP. Ainsi, il est important de pouvoir résoudre ce problème en un temps raisonnable pour valider le choix de l'utilisation de la méthode SQP par rapport aux autres méthodes envisageables (méthode de pénalisation dans la section 4.2.1, méthode de PI dans la section 4.2.4).

Le théorème suivant, repris du théorème 13.2 de [18], donne un résultat standard de convergence local de la méthode SQP.

Théorème 4.2.4 *Supposons que f et c sont de classe C^2 dans un voisinage d'un point stationnaire x_* de (P) avec λ_* un multiplicateur de Lagrange associé. Supposons aussi que la condition de complémentarité stricte est vérifiée (voir définition 4.1.12) et que $(x_*, (\lambda_*)_{E\cup I_0^*})$ est un point stationnaire régulier (voir remarque 1. de 4.2.5) du problème avec contraintes d'égalité*

$$(P_{E\cup I_0^*}) \begin{cases} \min_x f(x) \\ c_i(x) = 0, \quad i \in E \cup I_*^0, \end{cases} \tag{4.11}$$

Considérons l'algorithme SQP local (voir algorithme II.1), dans lequel d_k est un point stationnaire de norme minimale du problème quadratique tangent (4.10). *Alors, il existe un voisinage V de (x_*, λ_*) tel que, si le premier itéré $(x_1, \lambda_1) \in V$:*

(i) l'algorithme SQP local est bien défini et il génère une suite $\{(x_k, \lambda_k)\}$ qui converge superlinéairement vers (x_*, λ_*) ;

(ii) les contraintes actives du problème quadratique tangent (4.10) *sont celles du problème (P) ;*

(iii) si, en plus, f et c sont de classe $C^{2,1}$ dans un voisinage de x_*, la convergence de $\{(x_k, \lambda_k)\}$ est quadratique.

Remarques 4.2.5

1. *Soit K la matrice définie par*

$$K := \begin{pmatrix} (L_{E\cup I_0^*})_* & (A_{E\cup I_0^*})_*^\top \\ (A_{E\cup I_0^*})_* & 0 \end{pmatrix},$$

   où $(A_{E\cup I_0^*})_* = (c_{E\cup I_0^*})'(x_*)$ et $(L_{E\cup I_0^*})_* = \nabla^2_{xx} L_{(P_{E\cup I_0^*})}(x_*, \lambda_*)$. Un point stationnaire de (4.11) est dit régulier si K n'est pas singulière (voir proposition 12.1 et définition 12.2 de [18]).

2. *La propriété (ii) du théorème (4.2.4) est connue sous le nom de propriété d'**identification finie des contraintes actives** de l'algorithme SQP. Il est souhaitable que la méthode de résolution du problème quadratique tangent (4.10) utilisée dans l'algorithme SQP tire parti de cette propriété.*

De nombreux codes SQP ont été développés, voici ci-dessous les plus connus :

* NPSOL, pour les problèmes de taille moyenne (voir [71])
* SNOPT, pour les problèmes de grande taille creux (voir [70])
* NLPQL (voir [139])

Nous noterons que, par comparaison avec d'autres type de méthode (méthode de pénalisation où de LA), la méthode SQP demande en général moins d'évaluation de la fonction coût (voir [123, chapitre 15]). Plus précisement, elle demande en général moins d'évaluations de la fonction coût par rapport à une méthode de points intérieurs (voir section 3.1 de l'article de la partie III). Cependant, chaque résolution de problème quadratique tangent est plus coûteuse à effectuer que la résolution des systèmes linéaires provenant des méthodes de pénalisation où de LA.

4.2.4 Méthodes de points intérieurs

Dans les années 90, la recherche en optimisation numérique s'est fortement concentrée sur les méthodes des points intérieurs. A l'origine, ces méthodes ont été développées pour résoudre des problèmes d'optimisation linéaire (le critère et les fonctions définissant les contraintes sont linéaires[1]). Le principal objectif était alors de trouver une méthode capable de concurrencer la célèbre méthode du simplexe (voir [43]) qui n'est pas polynomiale (c'est à dire, le nombre total d'opérations nécessaires pour résoudre un problème est borné par une fonction polynomiale de la dimension de celui-ci). Or, à partir des travaux de [101], il a été prouvé que les méthodes de points intérieurs peuvent être polynomiales et donc plus efficaces en théorie que la méthode du simplexe. Similairement aux méthodes de pénalisation intérieure, les méthodes de points intérieurs génèrent des itérés dans l'intérieur relatif de l'ensemble admissible de (P). Cette propriété les rend particulièrement efficaces sur les problèmes avec beaucoup de contraintes d'inégalités car

[1] Le terme linéaire est utilisé par abus de langage, il s'agit en fait de fonctions affines.

elles ne ressentent pas l'irrégularité du bord du domaine admissible. Une simple extension de l'algorithme de points intérieurs permet de traiter le cas des problèmes d'optimisation quadratique convexe. Cette extension conserve généralement les propriétés de convergence et de complexité polynomiale démontrées en optimisation linéaire. Pour plus de détails sur ce sujet voir [158, chapitre 8]. L'extension de la méthode aux problèmes non-linéaires constitue aujourd'hui un axe de recherche très important en optimisation. Nous allons donner ci-dessous les grandes lignes des méthodes de points intérieurs primales-duales appliquées à la résolution de (P) (nous suivrons l'approche donnée dans [123, chapitre 16]).

Rappelons que les conditions de KKT de (P) s'écrivent :

$$\begin{cases} \nabla f(x) + A(x)^\top \lambda = 0 \\ c_E(x) = 0 \\ c_I(x) \leq 0 \\ \lambda_I^\top c_I(x) = 0 \\ \lambda_I \geq 0. \end{cases}$$

En introduisant la variable d'écart $s = -c_I(x)$, on obtient le système équivalent :

$$\begin{cases} (a) \quad \nabla f(x) + A(x)^\top \lambda = 0 \\ (b) \quad c_E(x) = 0 \\ (c) \quad c_I(x) + s = 0 \\ (d) \quad \lambda_i s_i = 0, \quad \forall i \in I, \\ (e) \quad (\lambda_I, s) \geq 0. \end{cases} \tag{4.12}$$

En relaxant la condition d'admissibilité des contraintes d'égalité (condition (b)) par $c_E(x) = \nu\lambda_E$ et la condition de complémentarité (condition (d)) par $\lambda_i s_i = \nu$, $i \in I$, avec $\nu \in \mathbb{R}_{++}$, et en supprimant les inégalités (condition (e)), on peut approcher les conditions d'optimalité de (P) par le système d'équation :

$$\begin{cases} \nabla f(x) + A(x)^\top \lambda = 0 \\ c_E(x) = \nu\lambda_E \\ c_I(x) + s = 0 \\ \lambda_i s_i = \nu, \quad \forall i \in I. \end{cases} \tag{4.13}$$

Quelques remarques sur ce système d'équation :

1. Lorsque $\nu \to 0$, le système d'équation ci-dessus devient de plus en plus proche du système (4.12), c'est à dire des conditions de KKT de (P). Ainsi les algorithmes de points intérieurs consistent à résoudre une suite de systèmes (4.13) en prenant des valeurs de $\nu \in \mathbb{R}_{++}$ de plus en plus petites. Les contraintes $(\lambda_I, s) > 0$ peuvent être imposées par RL où RC (on démarre l'algorithme d'un point admissible, i.e $((\lambda_I)_1, s_1) > 0$), sachant qu'elles ne peuvent pas être nulles en la solution de (4.13).

2. Le système (4.13) est un système d'équations non-linéaires. Pour une valeur de ν donnée, on note $\xi(\nu) = (x(\nu), \lambda_E(\nu), s(\nu), \lambda_I(\nu))$ une solution de (4.13) (l'application $\nu \longrightarrow \xi(\nu)$ est multivaluée). On définit alors le chemin central $\mathcal{C}_{pd}$ par :

$$\mathcal{C}_{pd} := \{\xi(\nu) \quad : \quad \nu > 0\}$$

3. En éliminant les variables s et λ du système (4.13) on obtient alors l'équation suivante en x :

$$\nabla f(x) + \frac{1}{\nu} \sum_{i \in E} c_i(x) A_i(x) - \nu \sum_{i \in I} \frac{A_i(x)}{c_i(x)} = 0. \tag{4.14}$$

Cette dernière équation est la condition d'optimalité du problème d'optimisation sans contrainte

$$\min_x \left(f(x) + \frac{1}{2\nu} ||c_E(x)||_2^2 - \nu \log(-c_I(x)) \right).$$

Or, la fonction à minimiser dans ce dernier problème est identique à la fonction de pénalisation mixte de l'équation (4.6). On retrouve ainsi le lien qui existe entre les méthodes de points intérieurs et la méthode de pénalisation logarithmique : dans ces 2 méthodes on résout de manière approchée une suite de systèmes (4.13) en prenant des valeurs de $\nu \in \mathbb{R}_{++}$ de plus en plus petites. La différence entre ces 2 méthodes réside dans la technique utilisée pour résoudre le système (4.13). Dans la méthode de pénalisation logarithmique on résout ce système en éliminant tout d'abord les variables s et λ puis en appliquant ensuite une méthode de Newton sur l'équation (4.14), alors que dans les méthodes de points intérieurs on applique directement une méthode de Newton sur (4.13).

Pour résoudre (4.13) à ν donné, on applique la méthode de Newton (ou une méthode de Newton modifiée). A cette fin, on regroupe les 4 conditions de (4.13) dans la fonction F_ν définie par :

$$F_\nu(\xi) := \begin{bmatrix} \nabla f(x) + A(x)^\top \lambda \\ c_E(x) - \nu \lambda_E \\ \Lambda S e - \nu e \\ c_I(x) + s \end{bmatrix},$$

où Λ et S sont les matrices diagonales dont les éléments sont les λ_i, $i \in I$, et les s_i, $i \in I$. Le vecteur $e \in \mathbb{R}^m$ est défini par $e_i = 1 \; \forall i \in I$.

La méthode de Newton appliquée à l'équation F_ν prend alors la forme :

$$F'_\nu d\xi = -F_\nu(\xi).$$

En substituant l'expression de F_ν et de F'_ν dans l'équation précédente on obtient alors le système linéaire :

$$\begin{bmatrix} \nabla^2_{xx} L_P(x, \lambda) & A_E(x)^\top & 0 & A_I(x)^\top \\ A_E(x) & -\nu I & 0 & 0 \\ 0 & 0 & \Lambda & S \\ A_I(x) & 0 & I & 0 \end{bmatrix} d\xi = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}, \tag{4.15}$$

où

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} -\nabla f(x) - A(x)^\top \lambda \\ -c_E(x) + \nu\lambda_E \\ -\Lambda S e + \nu e \\ -c_I(x) - s \end{bmatrix}.$$

Puis, en multipliant la troisième ligne de (4.15) par S^{-1} (on rappelle que $s_i > 0$, $i \in I$) on obtient alors le système linéaire symétrique indéfini suivant à résoudre :

$$\begin{bmatrix} \nabla^2_{xx} L_P(x,\lambda) & A_E(x)^\top & 0 & A_I(x)^\top \\ A_E(x) & -\nu I & 0 & 0 \\ 0 & 0 & S^{-1}\Lambda & I \\ A_I(x) & 0 & I & 0 \end{bmatrix} d\xi = \begin{bmatrix} f_1 \\ f_2 \\ S^{-1} f_3 \\ f_4 \end{bmatrix}. \tag{4.16}$$

On obtient le nouvel itéré par la formule

$$\xi^+ = \xi + \alpha d\xi,$$

où α est le pas choisi tel que l'inégalité $(\lambda_I, s) > 0$ soit satisfaite.

Remarques 4.2.6

1. *A chaque itération, la tache principale de l'algorithme des points intérieurs consiste à résoudre le système linéaire* (4.16)*. Il est donc indispensable d'avoir une technique efficace pour le résoudre. En procédant par élimination des variables* ds*,* $d\lambda_I$ *et* $d\lambda_E$ *on peut réduire la dimension système du système* (4.16) *et obtenir ainsi une seule équation en* dx *à résoudre. Les 3 dernières lignes de* (4.16) *nous donne successivement :*

$$\begin{aligned} ds &= f_4 - A_I(x)dx \\ d\lambda_I &= S^{-1}(f_3 - \Lambda ds) \\ &= S^{-1}(f_3 - \Lambda f_4 + \Lambda A_I(x)dx) \\ d\lambda_E &= -\tfrac{1}{\nu} f_2 + \tfrac{1}{\nu} A_E(x)dx. \end{aligned}$$

En substituant les valeurs de ds*,* $d\lambda_I$ *et* $d\lambda_E$ *obtenues dans la première ligne de* (4.16) *on obtient l'équation en* x *suivante :*

$$Q_\nu(x,\lambda)dx = f, \tag{4.17}$$

où

$$Q_\nu = \left(\nabla^2_{xx} L_P(x,\lambda) + \frac{1}{\nu} A_E(x)^\top A_E(x) + A_I(x)^\top S^{-1} \Lambda A_I(x) \right) \tag{4.18}$$

est la matrice réduite et où

$$f = f_1 + \frac{1}{\nu} A_E(x)^\top f_2 + A_I(x)^\top S^{-1}(\Lambda f_4 - f_3)$$

est le second membre. Ainsi, à chaque itération de l'algorithme des points intérieurs on peut remplacer la résolution de (4.16) *par la résolution du système linéaire réduit* (4.17)*. Il est important de remarquer que la matrice $Q_\nu(x,\lambda)$ de ce système peut être très mal conditionnée ou singulière lorsque ν devient très proche de zéro (dans les dernières itérations de l'algorithme des points intérieurs). Ce phénomène, typique des méthodes de pénalisation, provient içi de la matrice $S^{-1}\Lambda$ qui peut contenir des éléments diagonaux très petits ou très grands lorsque ν tend vers zéro : les valeurs des écarts s_i^{-1} correspondant aux contraintes actives en la solution de (P) tendent vers $+\infty$, alors que les valeurs des multiplicateurs λ_i correspondant aux contraintes inactives en la solution de (P) tendent vers 0. Une possibilité intéressante pour résoudre le système linéaire réduit* (4.17) *est d'utiliser par exemple une factorisation de Cholesky incomplète de la matrice $Q_\nu(x,\lambda)$.*

2. *Pour que l'algorithme des points intérieurs soit complet il faut ajouter une stratégie pour faire décroître le paramètre ν vers 0. Dans l'état actuel de l'art, on n'est pas capable de faire décroître ν après chaque itération de* (4.17) *sauf asymptotiquement : lorsque ν est proche de 0, l'algorithme des points intérieurs a une converge quasi-quadratique (voir [81]).*
3. *Une technique de calcul du pas α et une fonction de mérite sont aussi nécessaires pour globaliser l'algorithme. Bien que le sujet reste d'actualité, on retiendra les algorithmes proposés dans [60] (avec pour fonction de mérite* (4.6)*), [67] et [26] (un des premiers résultat de convergence globale est donné dans [25]).*

Nous donnons ci-dessous quelques exemples de codes de PI :

* KNITRO (voir [26])
* LOQO (voir [151])
* IPOPT (voir [155])

4.2.5 Pénalisation exacte et globalisation par recherche linéaire de la méthode SQP

La méthode SQP décrite dans la section 4.2.3 converge si le premier itéré (x_1,λ_1) est assez proche d'un point stationnaire régulier (voir théorème 4.2.4). Comme un tel point n'est généralement pas disponible dans la plupart des applications, il est nécessaire d'utiliser des techniques dites de globalisation d'un algorithme local qui permettent de "forcer" la convergence même si le point de départ est loin d'une solution primale-duale de (P).

A l'heure actuelle, il existe deux grandes classes de technique de globalisation :

* la recherche linéaire (RL),
* les régions de confiance (RC).

Ces deux techniques de globalisation utilisent le même principe : elles mesurent le progrès

effectué lors du passage de l'itération x_k à l'itération x_{k+1} par l'intermédiaire d'une fonction auxiliaire dite fonction de mérite. Pour le problème (P), cette fonction auxiliaire, doit non seulement prendre en compte la minimisation effective de f, mais aussi la satisfaction des contraintes. Pour ce faire, les méthodes d'optimisation utilisent souvent la fonction de pénalisation $\Theta_\sigma(x) = f(x) + \sigma p(x)$ comme fonction de mérite, où p est une fonction qui pénalise la violation des contraintes ($p(x) = 0$ si $x \in \mathcal{X}$ et $p(x) > 0$ si $x \notin \mathcal{X}$) et où $\sigma > 0$ est le facteur de pénalisation (voir section 4.2.1).

Dans cette section, nous nous intéressons plus particulièrement à la fonction de pénalisation par norme générale définie par

$$\Theta_\sigma(x) = f(x) + \sigma||c(x)^\#||_P, \tag{4.19}$$

où $||.||_P$ est une norme quelconque et $.^\# : \mathbb{R}^m \mapsto \mathbb{R}^m$ est la fonction définie par

$$(v^\#)_i = \begin{cases} v_i \text{ if } i \in E \\ v_i^+ = \max(0, v_i) \text{ if } i \in I \end{cases} .$$

Nous nous limiterons aux résultats concernant la technique de globalisation de la méthode SQP par la méthode de recherche linéaire sur la fonction de pénalisation par norme Θ_σ. Dans cette méthode, on génère une suite de points $\{(x_k, \lambda_k)\}$ qui converge vers une solution primale-duale (x_*, λ_*) de (P). Cette suite est définie par la formule de récurrence

$$(x_{k+1}, \lambda_{k+1}) = (x_k, \lambda_k) + \alpha_k(d_k, \lambda_k^{QP} - \lambda_k),$$

où (d_k, λ_k^{QP}) est une solution primale-duale du PQT (4.10) et $\alpha_k > 0$ est le pas de la RL servant à faire décroître la fonction de mérite Θ_σ dans la direction d_k (ce pas est calculé par un algorithme de rebroussement). Cette approche est originale car elle utilise la solution du PQT (4.10) pour faire décroître la fonction de mérite Θ_σ et non pas une direction fondée sur un sous-gradient de la fonction non différentiable Θ_σ (ce qui aurait conduit à un algorithme moins rapide). La proposition suivante, reprise de la proposition 15.1 de [18], est essentielle : elle assure qu'il existe un $\sigma > 0$ tel que la direction primale d_k trouvée par la méthode SQP est bien une direction de descente de la fonction de mérite Θ_σ en x_k.

Proposition 4.2.7 *Si (d_k, λ_k^{QP}) satisfait les conditions d'optimalité (4.9), alors on a*

$$\Theta'_\sigma(x_k; d_k) \leq \nabla f_k^\top d_k - \sigma||c_k^\#||_P = -d_k^\top M_k d_k + (\lambda_k^{QP})^\top c_k - \sigma||c_k^\#||_P.$$

Si, en plus, $\sigma > ||\lambda_k^{QP}||_D$, alors on a

$$\Theta'_\sigma(x_k; d_k) \leq -d_k^\top M_k d_k.$$

En conclusion : $\Theta'_\sigma(x_k; d_k) < 0$, si $\sigma > ||\lambda_k^{QP}||_D$, si M_k est définie positive, et si x_k n'est pas un point stationnaire de (P).

Ainsi, à chaque itération k de l'approche décrite précédemment, il faut adapter la valeur de σ pour que d_k soit bien une direction de descente de la fonction de mérite Θ_σ (la fonction Θ_σ peut changer à chaque itération). Le concept de fonction de pénalisation exacte (voir définition 4.2.1) joue un rôle important dans la convergence de cette approche : il permet de stabiliser la valeur de σ. Voici ci-dessous une condition suffisante d'exactitude de Θ_σ reprise de la proposition 14.7 de [18] :

Proposition 4.2.8 *Supposons que f et $c_{E\cup I^0_*}$ sont deux fois différentiables en un minimum local x_* de (P) pour lequel la condition nécessaire d'optimalité* (4.2) *est satisfaite. En ce minimum x_* on suppose également que la condition suffisante d'optimalité du second ordre faible est respectée (cf. théorème* (4.1.15)*), et que*

$$\sigma > \sup_{\lambda_* \in \Lambda_*} ||\lambda_*||_D.$$

Alors, Θ_σ a un minimum local strict en x_*.

Chapitre 5

Inversion sous contraintes en tomographie de réflexion

Dans ce chapitre, nous traduisons les contraintes géophysiques que nous souhaitons imposer sur le modèle de sous-sol en termes mathématiques. Puis, nous posons le problème inverse avec contraintes de la tomographie de réflexion. Enfin, nous décrivons les conditions d'optimalité de ce problème d'optimisation.

5.1 Les contraintes : type, nombre, localisation, etc...

Dans toute l'étude nous nous limitons à des contraintes linéaires. Cette limitation provient des difficultés supplémentaires liées à l'aspect non-linéaire des contraintes [1] . Notons que, malgré cette limitation de plus en plus d'applications en tomographie de réflexion font intervenir des contraintes non-linéaires : voir par exemple les travaux de Sinoquet [140] où la vitesse est contrainte à peu varier le long d'une interface (guidage de la vitesse le long de lignes interpolant les interfaces) et les travaux de [27] où dans le cas d'une structure faillée les points d'impact des rayons doivent être situés sur une partie restreinte d'une interface. Dans la conclusion de ce mémoire, nous identifierons ces difficultés et nous donnerons des pistes pour aborder l'inversion sous contraintes non-linéaires. Même si la linéarité des contraintes apporte des simplifications importantes, le problème inverse de la tomographie de réflexion avec contraintes reste difficile à résoudre à cause de leur nombre important (jusqu'à 10000 contraintes) et des différents types possibles de contraintes :

- Contraintes de nature physique différente :
  - sur des paramètres de vitesse et/ou d'interface,

[1] Si les contraintes sont non-linéaires et si l'on applique une approche SQP, alors il n'est pas garanti que le hessien du lagrangien soit semi défini positif. En effet, dans le cas de contraintes non-linéaires il faut prendre en compte la courbure des contraintes dans le hessien du lagrangien, voir le point 3. de la remarque 4.2.2. Dans le cas linéaire cette courbure est évidemment nulle et on a $\nabla^2_{mm} L_{(P_{EI})} = \nabla^2_{mm} f$, c'est à dire que le hessien du lagrangien est au moins semi défini positif (voir section 2.5.3).

  - sur 3 ordres de dérivation (par exemple une interface peut être contrainte sur sa profondeur, sa pente et/ou sa courbure)
- Contraintes d'égalité et/ou d'inégalité (valeur fixe du gradient de vitesse en z ; profondeur minimale d'une interface).
- Contraintes locales et/ou globales (information sur la position d'une interface dans un puits).

Au vu de la discussion ci-dessus, le problème d'optimisation que l'on veut résoudre fait partie de la classe des problèmes d'optimisation non-linéaire avec contraintes linéaires d'égalité et d'inégalité. Cette information n'est pas anodine car l'identification de la classe du problème que l'on cherche à résoudre permet le choix d'un algorithme adapté. Dans notre cas, la présence de contraintes d'inégalité est un élément clé dans le choix de la future méthode d'optimisation. En effet, la présence de contraintes d'inégalité rend le problème beaucoup plus difficile à résoudre que si seules des contraintes d'égalité étaient présentes. Ceci est dû à ce qu'il est convenu d'appeler la "combinatoire" des problèmes avec contraintes d'inégalité et lié à la détermination des contraintes actives (i.e., nulles) en la solution (voir section 4.1.4). Dans le cas, par exemple, d'un problème avec n_I contraintes d'inégalité du type $l \leq c(m) \leq u$, on dénombre 3^{n_I} choix possibles de contraintes actives (pour chaque contrainte il y a 3 possibilités : soit la contrainte est inactive et $l < c_i(m) < u$, soit elle est active avec $c_i(m) = l$, soit elle est active avec $c_i(m) = u$). Ce nombre de combinaisons possibles devient rapidement énorme avec n_I supérieur à quelques unités (par exemple pour 10 contraintes d'inégalité on a le choix entre $3^{10} = 59049$ combinaisons possibles). Dans nos problèmes de tomographie, où on doit faire face à des milliers de contraintes, il est important d'utiliser une méthode d'optimisation efficace pour gérer cette combinatoire importante.

Nous avons vu précédemment qu'il est important de différencier les contraintes d'égalité et d'inégalité par leur traitement algorithmique spécifique. Pour ce faire, nous décrivons les contraintes par les équations d'admissibilité suivantes :

$$\begin{cases} c_i(m) = e_i, \ i \in E \\ l_i \leq c_i(m) \leq u_i, \ i \in I, \end{cases}$$

avec $E := \{1, ..., n_E\}$ et $I := \{n_E + 1, ..., n_C\}$. On note $n_I = n_C - n_E$ le nombre de contraintes d'inégalité. La fonction linéaire $c_E : \mathbb{R}^n \to \mathbb{R}^{n_E}$ représente les contraintes d'égalité alors que la fonction linéaire $c_I : \mathbb{R}^n \to \mathbb{R}^{n_I}$ décrit les contraintes d'inégalité. Le vecteur $e \in \mathbb{R}^{n_E}$ est le second membre de l'équation des contraintes d'égalité. Les vecteurs $l \in \bar{\mathbb{R}}^{n_I}$ et $u \in \bar{\mathbb{R}}^{n_I}$ sont respectivement les bornes inférieures et supérieures des contraintes d'inégalité [2]. Remarquons que la formulation des contraintes d'inégalité donnée ci-dessus est différente de celle qui est donnée dans la partie théorique (voir contraintes du problème 4.1). Notons que ces deux formulations sont équivalentes, la présence supplémentaire des bornes supérieures en plus des bornes infèrieures sur c_i dans la formulation ci-dessus permet de coller aux besoins de l'application. De plus, pour les

[2] On remarquera que dans cette notation les composantes des vecteurs l et u sont numérotées de $n_E + 1$ à n_C. Ainsi, si on prend par exemple un problème avec $n_E = 1$ et $n_I = 1$, le vecteur l s'écrit $l = l_2$ (l_1 n'existe pas)

contraintes d'inégalité, on suppose pour chaque contrainte que les bornes inférieures et supérieures sont strictement compatibles, cela se traduit par l'hypothèse suivante :

Hypothèses 5.1.1 $l_i < u_i, \ \forall i \in I.$

Du fait de l'hypothèse de linéarité, les jacobiennes des fonctions c_E et c_I (fonctions linéaires) sont indépendantes de m. On peut donc écrire les équations d'admissibilité des contraintes d'égalité et d'inégalité sous la forme du système matriciel

$$\begin{cases} C_E m = e \\ l \leq C_I m \leq u \end{cases}, \tag{5.1}$$

ou C_E (resp. C_I) est une matrice rectangulaire de taille $n_E \times n$ (resp. $n_I \times n$) représentant la jacobienne de c_E (resp. c_I) et m est le vecteur contenant les paramètres de vitesse et d'interface du sous-sol (cf. section 2.3 pour la définition de m).

Définition 5.1.2 *On appelle $\mathcal{M}_C := \{m \in \mathcal{M} : C_E m = e, \ l \leq C_I m \leq u\}$ l'ensemble des modèles m admissibles (les modèles de $\mathcal{M}_C$ sont solutions des équations d'admissibilité (5.1))*

Définition 5.1.3 *On dit que les contraintes (5.1) sont compatibles si $\mathcal{M}_C \neq \emptyset$.*

Définition 5.1.4 *Pour les contraintes d'inégalité, on dit que c_i, $i \in I$ est active en m si $c_i(m) = l_i$ ou $c_i(m) = u_i$ et on note*

$$I^0(m) := \{i \in I \ : \ c_i(m) = l_i \text{ ou } c_i(m) = u_i\}$$

l'ensemble des indices des contraintes d'inégalité actives en m.

5.2 Modélisation des contraintes en tomographie de réflexion

Au cours de cette section on va tout d'abord mettre en évidence l'ensemble des contraintes que l'on souhaite appliquer au modèle de sous-sol de la tomographie. Puis, on va montrer que ces contraintes sont bien linéaires (une hypothèse requise par les algorithme mis en œuvre). Et enfin, on va s'intéresser à leur nombre et à la structure des matrices C_E et C_I qui les caractérisent.

On donne ci-dessous la liste des contraintes que l'on souhaite appliquer en un point donné de coordonnées $X = (x, y, z)$ ($X_{2D} = (\tilde{x}, \tilde{y})$ avec $(\tilde{x}, \tilde{y}) = (x, y)$ ou (x, z) ou (y, z)) du modèle de sous-sol (on rappelle que V_i représente la vitesse de la couche i et Z_i la position de l'interface i [3]) :

[3]Notons que Z_i peut être explicite en x, y ou z. Cependant, dans la plupart des cas Z_i exprime la profondeur de l'interface i (i.e. Z_i est explicite en z)

1. Sur les paramètres de vitesse :

   ☐ la vitesse de couche

$V_i(X) = e$	égalité
$V_i(X) - V_j(X) = e$	égalité couplant deux vitesses
$l \leq V_i(X) \leq u$	inégalité
$l \leq V_i(X) - V_j(X) \leq u$	inégalité couplant deux vitesses

   ☐ le gradient de vitesse dans la direction w ($w = x, y$ ou z)

$\nabla_w V_i(X) = e$	égalité
$\nabla_w V_i(X) - \nabla_w V_j(X) = e$	égalité couplant deux vitesses
$l \leq \nabla_w V_i(X) \leq u$	inégalité
$l \leq \nabla_w V_i(X) - \nabla_w V_j(X) \leq u$	inégalité couplant deux vitesses

   ☐ la dérivée seconde de la vitesse suivant les directions v et w (v et $w = x, y$ ou z)

$\nabla^2_{vw} V_i(X) = e$	égalité
$\nabla^2_{vw} V_i(X) - \nabla^2_{vw} V_j(X) = e$	égalité couplant deux vitesses
$l \leq \nabla^2_{vw} V_i(X) \leq u$	inégalité
$l \leq \nabla^2_{vw} V_i(X) - \nabla^2_{vw} V_j(X) \leq u$	inégalité couplant deux vitesses

2. Sur les paramètres d'interface :

   ☐ la profondeur d'une interface

$Z_i(X_{2D}) = e$	égalité
$Z_i(X_{2D}) - Z_j(X_{2D}) = e$	égalité couplant deux interfaces
$l \leq V_i(X_{2D}) \leq u$	inégalité
$l \leq V_i V_i(X_{2D}) - V_j(X_{2D}) \leq u$	inégalité couplant deux interfaces

   ☐ la pente d'une interface dans la direction w ($w = \tilde{x}$ ou $\tilde{y}$)

$\nabla_w Z_i(X_{2D}) = e$	égalité
$\nabla_w Z_i(X_{2D}) - \nabla_w Z_j(X_{2D}) = e$	égalité couplant deux interfaces
$l \leq \nabla_w Z_i(X_{2D}) \leq u$	inégalité
$l \leq \nabla_w Z_i(X_{2D}) - \nabla_w Z_j(X_{2D}) \leq u$	inégalité couplant deux interfaces

   ☐ la dérivée seconde d'une interface suivant les directions v et w (v et $w = \tilde{x}$ ou $\tilde{y}$)

$\nabla^2_{vw} Z_i(X_{2D}) = e$	égalité
$\nabla^2_{vw} Z_i(X_{2D}) - \nabla^2_{vw} Z_j(X_{2D}) = e$	égalité couplant deux interfaces
$l \leq \nabla^2_{vw} Z_i(X_{2D}) \leq u$	inégalité
$l \leq \nabla^2_{vw} Z_i(X_{2D}) - \nabla^2_{vw} Z_j(X_{2D}) \leq u$	inégalité couplant deux interfaces

Les fonctions Z et V, de part leur définition, ne sont linéaires que par rapport aux paramètres du modèle de sous-sol (les coefficients des fonctions de base B-spline). Ainsi, on ne peut pas vraiment imposer que ces contraintes soient satisfaites sur un ensemble de points continus du sous-sol [4]. Par exemple, on ne pourra pas exprimer la contrainte d'inégalité telle que $l \leq Z_i(X_{2D}) \leq u$ pour tout $X_{2D} \in [x_{ini}, x_{fin}] \times [y_{ini}, y_{fin}]$. En fait, cette contrainte sera "discrétisée" par $l \leq Z_i(X_{2D}) \leq u$, pour $X_{2D} \in \{x_1, x_2, ..., x_{n_x}\} \times \{y_1, y_2, ..., y_{n_y}\}$ tel que $\{x_1, x_2, ..., x_{n_x}\} \subset [x_{ini}, x_{fin}]$ et $\{y_1, y_2, ..., y_{n_y}\} \subset [y_{ini}, y_{fin}]$. Il y a alors autant de contraintes que de points (x, y) choisis par la discrétisation : dans l'exemple précédent on dénombre $n_x * n_y$ contraintes d'inégalité sur Z_i.

Le calcul des matrices jacobiennes d'égalité (C_E) et d'inégalité (C_I) se fait à partir des relations linéaires qui existent entre les grandeurs contraintes (profondeurs d'interface et leurs dérivées jusqu'à l'ordre 2, vitesses de couche et leurs dérivées jusqu'à l'ordre 2) et les paramètres du modèle (coefficients des fonctions de bases B-splines). Ces relations linéaires découlent de l'expression de la profondeur Z et de la vitesse V par la discrétisation B-spline (cf. section 2.3.2 pour les expressions de Z et V). Ainsi, les termes des matrices jacobiennes C_E et C_I contiennent uniquement des produits de fonctions de base B-spline et/ou de leurs dérivées. Comme les fonctions de base B-spline sont non nulles sur seulement 4 intervalles (cf. annexe A), on en déduit que les matrices C_E et C_I sont creuses. Un stockage de type "morse" (seuls les terme non nuls ainsi que leurs places dans la matrice, la ligne et la colonne auquelles ils appartiennent, sont stockés) de ces matrices est donc particulièrement recommandé.

Nous donnons ci-dessous, à titre d'exemple, le calcul des termes d'une matrice d'égalité C_E qui contraint une interface 2D Z_i d'être à la profondeur e sur une grille de points régulière appartenant à $[x_{ini}, x_{fin}] \times [y_{ini}, y_{fin}]$ (le calcul des termes matriciels pour d'autres grandeurs contraintes ne sera pas décrit, mais la généralisation est facile à partir des relations linéaires figurant dans l'annexe B). On note n_x (resp. n_y) le nombre de points de la grille dans la direction x (resp. y). Le nombre de points total de la grille est

[4] La proposition ci-dessous montre que dans le cas particulier d'une contrainte d'égalité discrétisée par un nombre suffisant de points d'application on peut imposer une contrainte sur un ensemble de points continus du sous-sol (la démonstration découle du fait qu'une B-spline cubique a au maximum 4 degrés de liberté).

Proposition 5.2.1 *Nous considérons le cas d'une contrainte d'égalité C_E (parmi la liste des contraintes d'égalité linéaires possibles) sur V (resp. Z) d'ordre (k_x, k_y, k_z) (resp. $(k_{\tilde{x}}, k_{\tilde{y}})$), avec k_x,k_y et $k_z \in \{0, 1, 2\}$ qui représentent respectivement l'ordre de dérivation de V dans les directions x,y et z. On suppose que la "discrétisation" de cette contrainte est telle que le nombre de points d'application à l'intérieur d'un intervalle 3D Ω (resp. 2D) de nœud B-spline d'une vitesse V_i (resp. d'une interface Z_i) soit supérieur ou égal à $(4-k_x)*(4-k_y)*(4-k_z)$ (resp. $(4-k_{\tilde{x}})*(4-k_{\tilde{y}})$). Si un modèle $m \in \mathcal{M}$ de sous-sol satisfait cette contrainte pour tous les points d'application choisis dans Ω, alors cette contrainte est également satisfaite pour tous les points de Ω.*

alors égal à $n_x * n_y$. Tous les points de cette grille sont représentés par $X_{j=0,...,n_x*n_y-1}$:

$$\begin{aligned} &\forall j \in \{0, 1, ..., n_x * n_y - 1\} \quad X_j = (x_j, y_j) \quad \text{tel que} \\ &x_j = x_{ini} + \left(\frac{x_{fin} - x_{ini}}{n_x - 1} * (j \% n_x) \right) \\ &y_j = y_{ini} + \left(\frac{y_{fin} - y_{ini}}{n_y - 1} * (j \div n_x) \right), \end{aligned} \tag{5.2}$$

avec $\div$ qui est l'opérateur de la division euclidienne entière et $\%$ l'opérateur du reste de la division euclidienne entière. On a donc, pour cet exemple, $n_x * n_y$ contraintes d'égalité qui s'écrivent

$$c_j(m) = e \quad \forall j \in E := \{0, 1, ..., n_x * n_y - 1\}.$$

La ligne j de la matrice C_E correspond à la matrice jacobienne de la contrainte $Z_i(X_j) = e$. Or l'expression de la profondeur d'une interface 2D Z_i au point X_j s'écrit (voir annexe B) :

$$Z_i(X_j) = Z_i(x_j, y_j) = \sum_{m_x=1}^{N_x^{Z_i}} \sum_{m_y=1}^{N_y^{Z_i}} m_{m_x,m_y}^{Z_i} B_{m_x}(x_j) B_{m_y}(y_j) = (C_E)_j \begin{bmatrix} m^{V_i} \\ m^{Z_i} \end{bmatrix}.$$

On en déduit que $(C_E)_j = [0 \ \ (B_{m_x}(x_j)B_{m_y}(y_j))_{m_x,m_y}]$ (les premières colonnes de $(C_E)_j$ sont nulles car cette contrainte porte uniquement sur les paramètres d'interfaces m^{Z_i}), et on remarque bien, a posteriori, que les termes de la matrice C_E ne s'expriment que par des produits de fonction de base B-spline. De plus, d'après la définition des fonctions de base B-spline (voir annexe A), les $B_*(x_j)$ (resp. $B_*(y_j)$) sont nuls sauf pour 4 indices consécutifs. Ainsi, pour x_j compris entre les noeuds de B-spline x_{l_x} et x_{l_x+1}, i.e. $x_{l_x} \leq x_j < x_{l_x+1}$, on peut simplifier l'expression ci-dessus par :

$$Z_i(x_j, y_j) = \sum_{m_x=l_x-3}^{l_x} \sum_{m_y=l_y-3}^{l_y} m_{m_x,m_y}^{Z_i} B_{m_x}(x_j) B_{m_y}(y_j).$$

Cela implique que la matrice $(C_E)_j = [0 \ \ (B_{m_x}(x_j)B_{m_y}(y_j))_{m_x=l_x-3,l_x,m_y=l_y-3,l_y}]$ possède *au maximum* 16 éléments non-nuls. Ce calcul conduit aux remarques suivantes :

1. Pour une fonction B-spline 2D (cas par exemple de l'expression des interfaces Z_i ou des vitesses V_i 2D), le nombre maximum de paramètres B-spline intervenant dans la définition d'un point est de 16. Ainsi, les matrices de contrainte d'interface ou de vitesse 2D ont au maximum 16 éléments non-nuls par ligne.
2. Pour une fonction B-spline 3D (cas de l'expression vitesse V_i 3D), le nombre maximum de paramètres B-spline intervenant dans la définition d'un point est de 64. Ainsi les matrices de contrainte de vitesse 3D ont au maximum 64 éléments non-nuls par ligne.
3. Comme les contraintes couplant deux interfaces (ou deux vitesses) 2D font intervenir deux fonctions B-spline 2D distinctes, leurs matrices ont au maximum 32 éléments non nuls par ligne. De même pour les contraintes couplant deux vitesses 3D : leurs matrices possèdent au maximum 128 éléments non-nuls par ligne.

4. Tous les éléments d'une matrice de contrainte n'agissant pas sur les paramètres associés au type de la contrainte sont nuls. Ainsi, dans le cas de l'exemple ci-dessus ($(C_E)_j = [0 \ (B_{m_x}(x_j)B_{m_y}(y_j))_{m_x,m_y}]$) tous les éléments de $(C_E)_j$ agissant sur les paramètres de vitesse m^{V_i} sont nuls. On dit aussi que les éléments des matrices de contrainte sont *compartimentés*.
5. L'expression des contraintes par des fonctions B-spline cubiques implique que les éléments non-nuls d'une ligne de matrice ont une structure particulière : ils sont dispersés par paquet successif contenant au plus 4 éléments.

Dans les trois premières remarques, nous observons que le nombre d'éléments non-nuls par ligne d'une matrice de contrainte est toujours majoré (quelque soit son type). Les observations précédentes débouchent sur les remarques suivantes :

Remarques 5.2.2 *Dans nos applications en tomographie de réflexion, nous nous attendons à rencontrer des matrices de contrainte :*

- *très creuses,*
- *compartimentées,*
- *avec une structure particulière des éléments non-nuls.*

Dans les figures 5.1, 5.2, et 5.3 nous avons représenté les éléments non-nuls de 3 matrices de contraintes d'égalité C_E pour le modèle de sous-sol (synthétique) "KARINE" (voir annexe D pour plus de détails sur ce modèle). Nous rappelons que ce modèle est constitué d'une vitesse 2D (V_1) et de 2 interfaces 2D (Z_1 et Z_2). La vitesse V_1 est représentée par 40 (10*4) paramètres B-spline (m_{V_1}) et les interfaces Z_1 et Z_2 sont respectivement représentées par 152 (38*4) et 285 (71*4) paramètres B-spline (m_{Z_1} et m_{Z_2}). Un modèle "KARINE" m de sous-sol s'exprime alors par $m = (m_{V_1}, m_{Z_1}, m_{Z_2})$.

Chaque matrice, est obtenue par la "discrétisation" d'une contrainte spécifique :

* Contrainte d'égalité ou d'inégalité sur une vitesse de couche : la matrice de la figure 5.1 est obtenue par la discrétisation de la contrainte de vitesse 2D "$V_1 = e$" (ou "$l \leq V_1 \leq u$"). Cette contrainte est appliquée en $n_x * n_y * n_z = 5 * 2 * 1 = 10$ points de discrétisation.
* Contrainte d'égalité ou d'inégalité sur la profondeur d'une interface : la matrice de la figure 5.2 est obtenue par la discrétisation de la contrainte d'interface 2D "$Z_2 = e$" (ou "$l \leq Z_2 \leq u$"). Cette contrainte est appliquée en $n_x * n_y = 4 * 35 = 140$ points de discrétisation.
* Contrainte d'égalité ou d'inégalité sur une épaisseur de couche : la matrice de la figure 5.3 est obtenue par la discrétisation de la contrainte couplant deux interfaces 2D "$Z_2 - Z_1 = e$" (ou "$l \leq Z_2 - Z_1 \leq u$"). Cette contrainte est appliquée en $n_x * n_y = 4 * 35 = 140$ points de discrétisation.

L'observation de ces trois matrices confirme la remarque 5.2.2 :

1. Ces trois matrices sont très creuses : les éléments non nuls représentés par des points noirs sont en très faible proportion par rapport aux éléments nuls (zones grises).

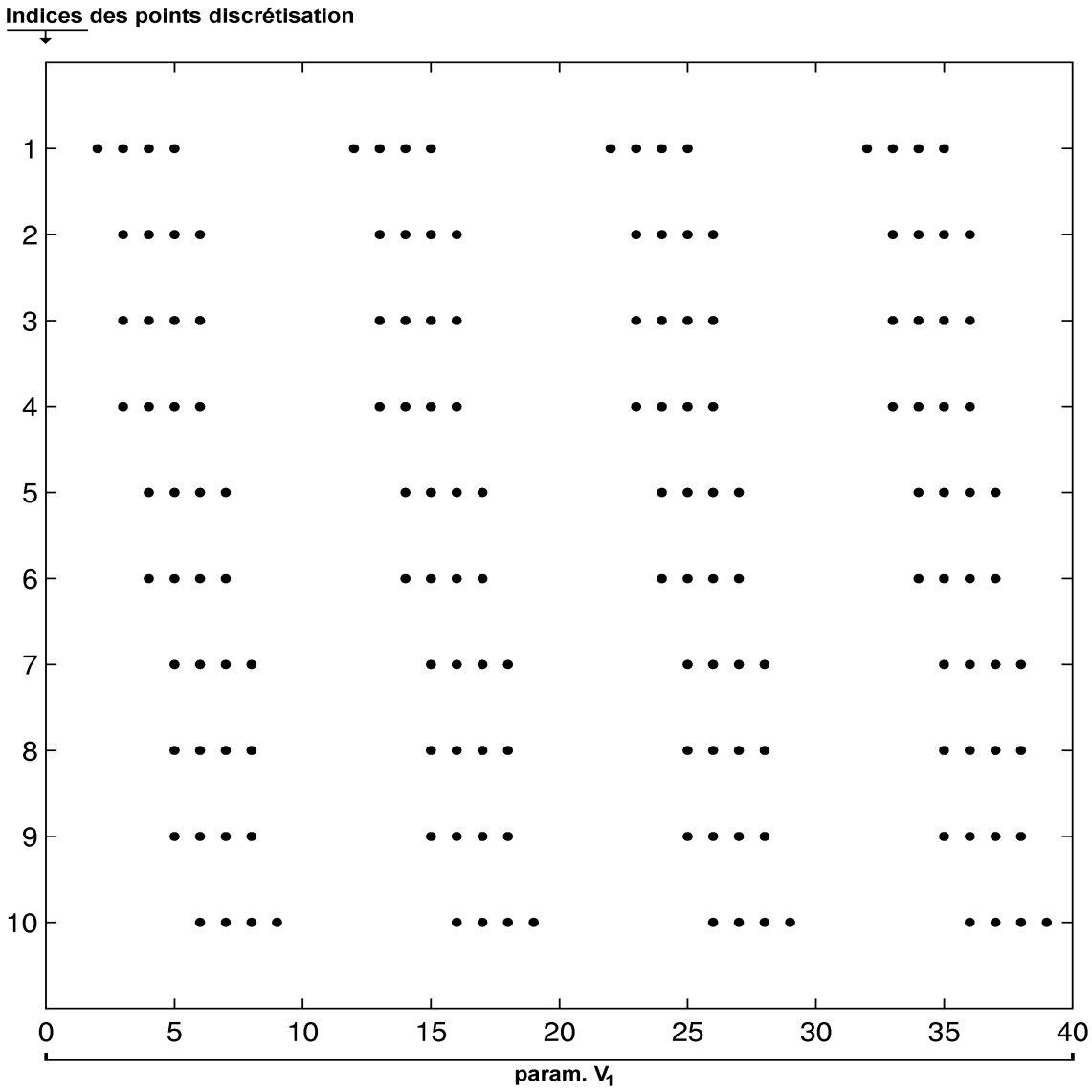

Fig. 5.1 Éléments non nuls du bloc vitesse de la matrice C_E construite à partir de $n_x{*}n_y{*}n_z = 5{*}2{*}1 = 10$ points de discrétisation : contrainte sur la vitesse V_1 2D du type "$V_1 = e$".

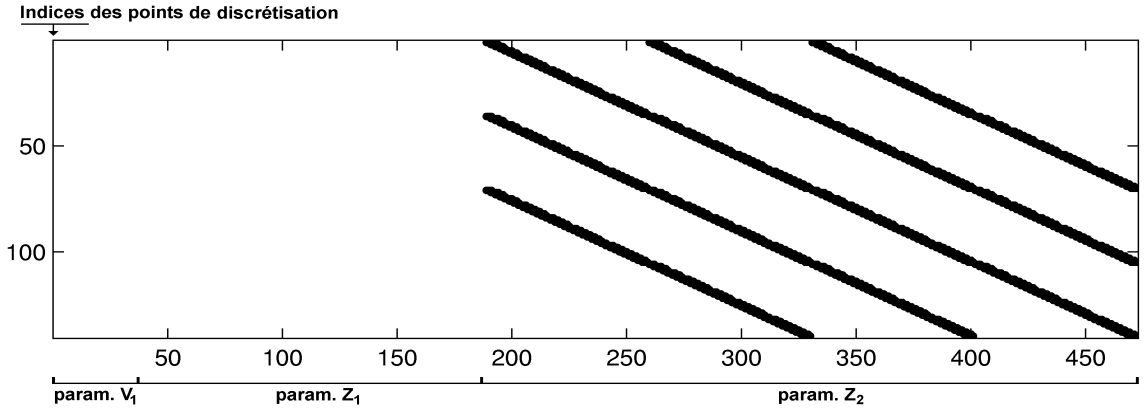

Fig. 5.2 Éléments non nuls de la matrice C_E construite à partir de $n_x * n_y = 4 * 35 = 140$ points de discrétisation : contraintes sur la profondeur Z_2 (2D) du type "$Z_2 = e$".

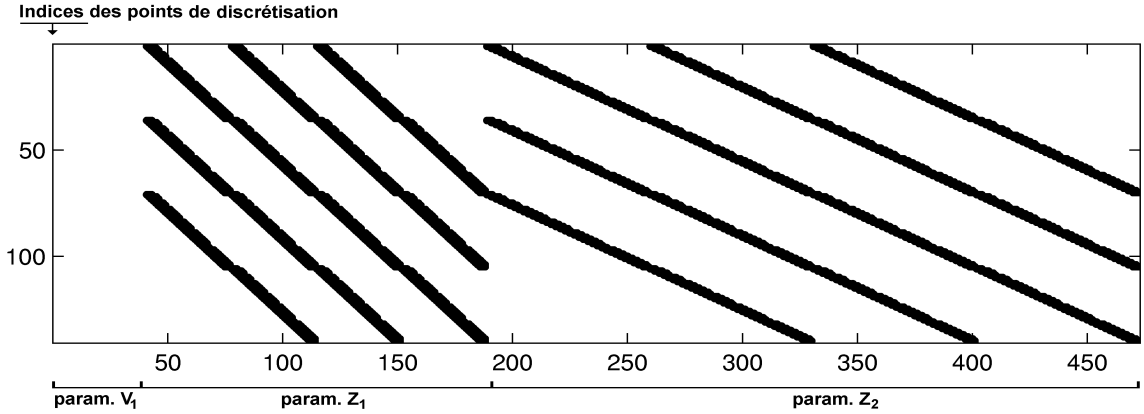

Fig. 5.3 Éléments non nuls de la matrice C_E construite à partir de $n_x * n_y = 4 * 35 = 140$ points de discrétisation : contraintes sur l'épaisseur de la couche V_1 délimitée par les interfaces Z_1 et Z_2 (2D) du type "$Z_2 - Z_1 = e$".

2. Les éléments non-nuls des matrices de contraintes sont “compartimentés” : ils appartiennent aux colonnes agissant sur les “quantités physiques” qui sont contraintes. Par exemple, dans la figure 5.1, où on contraint la vitesse de couche V_1, les éléments non-nuls appartiennent aux colonnes représentant les paramètres de vitesse m_{V_1}. En effet, seules les 40 premières colonnes contiennent des éléments non-nuls. De même, pour les matrices des figures 5.2, et 5.3 où les éléments non nuls appartiennent respectivement aux colonnes représentant les paramètres d’interface m_{Z_2} (colonnes 193 à 476) et aux colonnes représentant les paramètres d’interface m_{Z_1} et m_{Z_2} (colonnes 41 à 476). Le dernier cas est particulier car les paramètres d’interface de Z_1 et de Z_2 sont couplés : c’est le cas des contraintes couplant deux interfaces.
3. Les éléments non nuls de ces trois matrices ont des structures géométriques remarquables : ils forment une ou plusieurs stries obliques parallèles. Cette particularité s’explique tout d’abord par la paramétrisation en B-spline cubique qui limite le nombre de points non nuls par ligne. Et ensuite elle s’explique par les relations qui existent entre 2 lignes successives d’une matrice, relations qui proviennent du choix particulier de la discrétisation régulière des points d’applications des contraintes (voir l’exemple de discrétisation 2D - x,y - équation (5.2)).

Afin d’avoir une idée plus précise de la forme des matrices de contrainte en tomographie de réflexion nous avons illustré, dans les figures 5.4 et 5.5, les contraintes de l’exemple sur données réelles KIMASI (voir section 4.2 de l’article inclu dans la partie III de ce rapport) pour plus d’informations sur cet exemple). Les 3 commentaires ci-dessus sont aussi valables pour les matrices de contraintes du modèle KIMASI. Dans la figure 5.4 la matrice du haut illustre les contraintes d’inégalité sur les paramètres de vitesse, alors que celle du bas représente les contraintes d’inégalité sur les paramètres d’interface. Ces matrices sont obtenues par la “discrétisation” d’une contrainte de tomographie spécifique :

* Contraintes sur le gradient vertical de vitesse dans le tertiaire (matrice du haut - bloc en haut à gauche) : “$0.1 \leq \nabla_z$vtert ≤ 0.3”. Cette contrainte est discrétisée par $n_x * n_y * n_z = 10 * 10 * 10 = 1000$ points. Comme la vitesse “vtert” est paramétrée par des B-splines 3D, on observe que ce bloc possède plus d’éléments non-nuls par ligne (jusqu’à 64).
* Contraintes sur les variations de vitesses (matrice du haut - bloc en bas à droite) :
  ** dans le paléocène : “$2,5 \leq$ vpal ≤ 4” (contrainte discrétisée par $n_x * n_y * n_z = 15 * 10 * 1 = 150$ points),
  ** dans la craie : “$3,5 \leq$ vichalk $\leq 5,7$” (contrainte discrétisée par $n_x * n_y * n_z = 15 * 10 * 1 = 150$ points) et “$4,2 \leq$ vchalk $\leq 5,8$” (contrainte discrétisée par $n_x * n_y * n_z = 15 * 10 * 1 = 150$ points).
* Contraintes sur les épaisseurs de couches (matrice du bas) :
  ** “tpal $\leq$ tchalk” (contrainte discrétisée par $n_x * n_y = 20 * 20 = 400$ points),
  ** “tchalk $\leq$ ichalk” (contrainte discrétisée par $n_x * n_y = 20 * 20 = 400$ points).

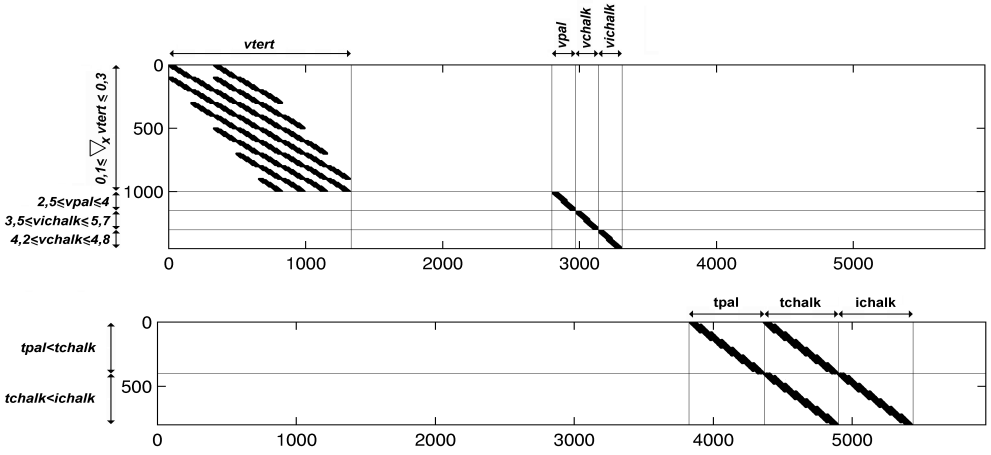

Fig. 5.4 Éléments non nuls de la matrice d'inégalité C du modèle KIMASI : contraintes sur les paramètres de vitesse (en haut), contraintes sur les paramètres d'interface (en bas).

Dans la figure 5.5 la matrice illustre les contraintes d'égalité sur les paramètres d'interface, dont les caractéristiques sont :

* Contrainte sur la profondeur d'une interface : position des interfaces "tpal", "tchalk" et "bchalk" fixées ponctuellement grâce aux mesures (logs) provenant de 6 puits différents.

5.3 Formulation du problème inverse avec contraintes

Le problème général de l'inversion avec contraintes en tomographie de réflexion, noté (P_{EI}), se formule simplement comme le problème régularisé de moindres-carrés non-linéaire (2.12) auquel on ajoute les contraintes linéaires définies en (5.1) :

$$(P_{EI}) \begin{cases} \min\limits_{m\in\mathcal{M}} \left(f(m) := \dfrac{1}{2} \left|\left| T(m) - T^{obs} \right|\right|_2^2 + \dfrac{\sigma^2}{2}\, m^\top R\, m \right) \\ C_E m = e \\ l \leq C_I m \leq u. \end{cases} \tag{5.3}$$

On rappelle que le problème (5.3) admet une solution en dimension finie si R est définie positive, si $T(.)$ est continue et si les contraintes sont compatibles.

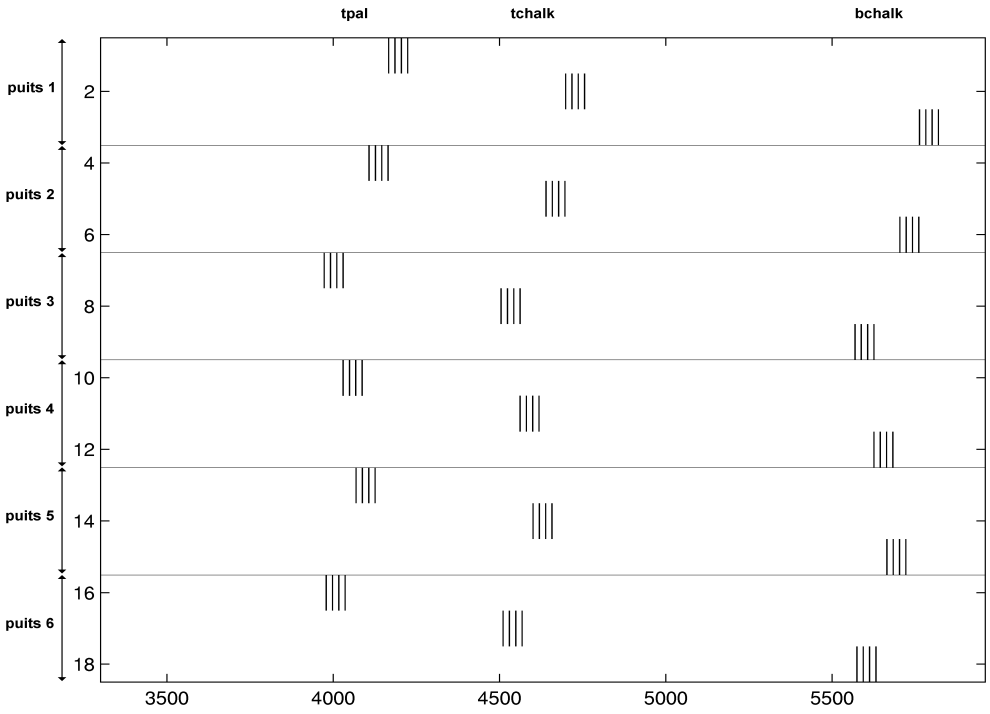

Fig. 5.5 Éléments non nuls de la matrice d'égalité E du modèle KIMASI : contraintes sur les paramètres d'interface. On peut observer sur cette figure les contraintes liées aux 6 puits, les 4 splines auxquelles appartiennent les points considérés et les 3 groupes de barres pour les 3 interfaces du modèle.

5.4 Conditions nécessaires d'optimalité du problème non-linéaire

Soit $L_{(P_{EI})} : \mathcal{M} \times \mathbb{R}^{n_E} \times \mathbb{R}^{n_I} \times \mathbb{R}^{n_I} \to \mathbb{R}$ le lagrangien associé au problème (P_{EI}) défini par

$$L_{(P_{EI})}(m, \mu_E, \mu_u, \mu_l) = f(m) + \mu_E^\top (C_E m - e) + \mu_u^\top (C_I m - u) - \mu_l^\top (C_I m - l), \quad (5.4)$$

où le vecteur μ_E (resp. μ_l et μ_u) est appelé multiplicateur de Lagrange associé aux contraintes d'égalité (resp. aux contraintes d'inégalité).

Comme f est suffisamment régulière (au moins C^1 de $\mathcal{M} \to \mathbb{R}$) et que les contraintes sont affines (donc qualifiées (QC-A) voir la définition 4.1.9 et la proposition 4.1.10), on peut appliquer le théorème de Karush-Kuhn-Tucker (cf. théorème 4.1.11) : si $\widehat{m}$ est une solution de (5.3), il existe $\widehat{\mu}_E \in \mathbb{R}^{n_E}$, $\widehat{\mu}_l \in \mathbb{R}^{n_I}$ et $\widehat{\mu}_u \in \mathbb{R}^{n_I}$ tels que les conditions nécessaires d'optimalité suivantes soient vérifiées :

$$\begin{cases} (a)\ \nabla f(\widehat{m}) + C_E^\top \widehat{\mu}_E - C_I^\top (\widehat{\mu}_l - \widehat{\mu}_u) = 0 \\ (b)\ C_E \widehat{m} = e, \quad l \le C_I \widehat{m} \le u \\ (c)\ (\widehat{\mu}_l, \widehat{\mu}_u) \ge 0 \\ (d)\ \widehat{\mu}_l^\top (C_I \widehat{m} - l) = 0,\ \widehat{\mu}_u^\top (C_I \widehat{m} - u) = 0. \end{cases} \quad (5.5)$$

Un point $(\widehat{m}, \widehat{\mu}_E, \widehat{\mu}_l, \widehat{\mu}_u)$ vérifiant (5.5) est appelé solution primale-duale ou point stationnaire de (P_{EI}) ($\widehat{m}$ est la solution primale, $(\widehat{\mu}_E, \widehat{\mu}_l, \widehat{\mu}_u)$ est la solution duale). Notons que la condition (a) de (5.5) se reformule comme :

$$\nabla_m L_{(P_{EI})}(\widehat{m}, \widehat{\mu}) = 0, \tag{5.6}$$

avec $\widehat{\mu} = (\widehat{\mu}_E, \widehat{\mu}_I)$ un multiplicateur de Lagrange optimal, et $\widehat{\mu}_I = \widehat{\mu}_l - \widehat{\mu}_u$.

Après avoir posé les bases, dans ce chapitre, de l'inversion avec contraintes en tomographie de réflexion nous allons maintenant aborder, dans le chapitre suivant, la résolution du problème (5.3) de moindres-carrés non-linéaire.

Chapitre 6

Algorithme de programmation quadratique successive en tomographie de réflexion

Du point de vue de l'optimisation, le problème (5.3) est un problème de moindres-carrés non-linéaire avec contraintes linéaires. A l'heure actuelle, il existe 2 grandes classes de méthodes pour résoudre des problèmes d'optimisation non-linéaires :

- méthodes de pénalisation (en particulier les algorithmes de points intérieurs, PI),
- méthode de programmation quadratique successive (SQP).

Le principe de ces 2 classes de méthodes a déjà été décrit dans le cadre de la résolution d'un problème d'optimisation non-linéaire général à la section 4.2. Notons en particulier que la classe des méthodes de pénalisation regroupe les méthodes classiques de pénalisation extérieure/intérieure (voir sous-section 4.2.1), les méthodes de lagrangien augmenté (voir sous-section 4.2.2) et les algorithmes de points intérieurs (voir sous-section 4.2.4). Parmi ces 2 classes de méthodes, nous avons retenu la méthode SQP (voir sous-section 4.2.3) pour résoudre le problème inverse de la tomographie de réflexion avec contraintes. Dans la section suivante, nous motiverons ce choix. Puis, dans la section 6.2, nous appliquerons cette méthode en définissant le problème quadratique tangent et en donnant les spécificités de notre algorithme local SQP. Enfin, dans la section 6.3, nous nous intéresserons à la globalisation cet algorithme.

6.1 Motivations du choix de la méthode SQP

Nous avons choisi de résoudre le problème (P_{EI}) par une méthode SQP pour les raisons suivantes :

1. Dans la classe des méthodes de pénalisation, nous avons écarté de notre choix les méthodes de pénalisation classique (intérieure/extérieure) ainsi que les méthodes de LA. Ces dernières se sont avérées en général moins efficaces que les méthodes plus récentes, que sont SQP et PI, pour résoudre les problèmes d'optimisation non-

linéaires.

2. Dans notre application en tomographie de réflexion, la résolution du problème direct (un algorithme de tracé de rayon) coûte cher en temps CPU par rapport au temps passé dans le solveur sans contrainte. La figure 6.1 illustre cet état de fait en montrant que le pourcentage de temps CPU associé à la résolution du problème direct est largement supérieur à 80% pour les modèles de notre bibliothèque (voir annexe D). Cette propriété de notre application implique que nous devons absolument choisir une méthode d'optimisation qui réduit au maximum le nombre d'évaluation de la fonction coût. Il nous a semblé que comme méthode newtonienne la méthode SQP ne devraie pas demander beaucoup plus d'itérations que l'algorithme gauss-newtonien du cas sans contrainte. Cette propriété a moins de chance d'être vérifiée par une méthode de PI car, en tant que méthode de pénalisation, elle minimise (de manière approchée) une suite de fonctions non-linéaires (voir section 3.1 de l'article figurant dans la partie III). Or, la méthode SQP est justement reconnue pour demander en général beaucoup moins d'évaluations de la fonction coût que la plupart des autres méthodes (en particulier par comparaison avec les algorithmes de points intérieurs), au prix toutefois de devoir résoudre un problème quadratique compliqué à chaque itération (voir [123, chapitre 15]). La méthode des points intérieurs et son incapacité à faire décroître le paramètre ν à chaque itération (voir le deuxième point de la remarque 4.2.6) semble demander substantiellement plus d'evaluations de la fonction coût.
3. Notons que le solveur gauss-newtonien sans contrainte (voir chapitre 2) converge en très peu d'itérations (de l'ordre de 10 itérations). Ainsi, par anticipation, nous pensons qu'une méthode de type Newton telle que la méthode SQP demandera peu d'évaluations de la fonction coût par rapport à la méthode des PI.
4. Nous avons vu que le problème principal de la méthode SQP consiste à résoudre les PQT en un temps raisonnable (voir remarque 2. de l'algorithme II.1). Nous verrons dans les chapitres suivants qu'il existe des techniques efficaces pour résoudre les problèmes quadratiques et tirer profit de la propriété d'identification finie des contraintes actives de la méthode SQP (voir théorème 4.2.4).
5. Un fait remarquable, a posteriori, est que le choix de la méthode SQP au premier niveau de décomposition de la méthode d'optimisation (voir figure 3.1) va déterminer les choix effectués dans les niveaux infèrieurs (méthode de LA au niveau 2 - chapitre 7, méthode de GP-AC-GC au niveau 3 - chapitre 8).

6.2 Mise en œuvre de la méthode de programmation quadratique successive

Dans cette section, nous allons appliquer la méthode SQP à la résolution du problème de tomographie (P_{EI}) (les résultats généraux concernant cette méthode sont décrits à la section 4.2.3). La méthode SQP consiste à décomposer la résolution d'un problème non-

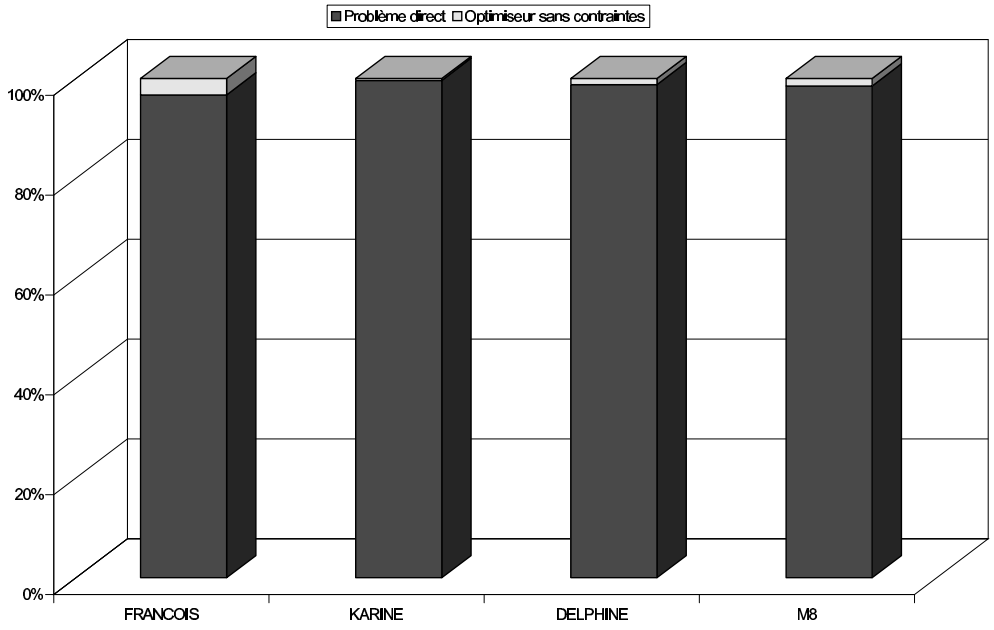

Fig. 6.1 Pourcentage du temps CPU passé dans le problème direct/solveur sans contrainte

linéaire en la résolution d'une suite de Problèmes Quadratiques Tangents (PQT). Lorsque l'on souhaite utiliser une méthode SQP, les 2 premières questions à se poser sont : comment définir les problèmes quadratiques successifs ? Peut-on les résoudre (en un temps raisonnable) ? Dans les trois sous-sections suivantes nous donnerons des réponses à ces questions. Dans la dernière sous-section nous développerons notre algorithme SQP local et présenterons ses spécificités.

6.2.1 Approximation gauss-newtonienne du Problème Quadratique Tangent

Dans la section 4.2.3 sur les rappels de la méthode SQP, nous avons vu que l'expression du PQT fait intervenir le hessien du lagrangien du problème original (voir équation (4.10)). Du fait que notre problème (P_{EI}) ne fait intervenir que des contraintes linéaires le hessien de ces dernières n'intervient pas dans le PQT :

$$\nabla^2_{mm} L_{(P_{EI})} = \nabla^2_{mm} f.$$

Comme l'approximation gauss-newtonienne H_k calculée dans la partie I est une bonne approximation de $\nabla^2_{mm} f$, elle est aussi une bonne approximation du hessien du lagrangien du problème (P_{EI}). Ainsi, pour des raisons identiques à celles développées à la section 2.5.3 on prendra l'approximation GN de $\nabla^2_{mm} f$:

$$\nabla^2_{mm} f(m_k) \approx H_k := J_k^\top J_k + \sigma^2 R,$$

où J_k est la jacobienne de T en m_k. Observons que la matrice H_k est semi-définie positive.

Les contraintes (5.1) doivent aussi être exprimées en fonction de la perturbation du modèle δm en m_k. Pour les contraintes d'égalité on a

$$c_E(m_k + \delta m) = C_E.(m_k + \delta m),$$

qui peut se récrire comme

$$C_E.\delta m = e_k, \tag{6.1}$$

où $e_k = e - C_E.m_k$. On obtient, par un calcul similaire,

$$l_k \leq C_I.\delta m \leq u_k, \tag{6.2}$$

pour les contraintes d'inégalité, avec

$$\begin{cases} l_k = l - C_I m_k, \\ u_k = u - C_I m_k. \end{cases} \tag{6.3}$$

Le Problème Quadratique Tangent (PQT) à l'itération k de Gauss-Newton se formule donc comme la minimisation de l'approximation quadratique F_k de $\delta m \mapsto f(m_k + \delta m)$ soumise aux contraintes linéaires (6.1) et (6.2) :

$$(PQT)_k \begin{cases} \min\limits_{\delta m \in \mathbb{R}^n} \left(F_k(\delta m) = \frac{1}{2}\, \delta m^\top H_k\, \delta m + g_k^\top \delta m \right) \\ C_E.\delta m = e_k \\ l_k \leq C_I.\delta m \leq u_k, \end{cases} \tag{6.4}$$

où

$$g_k = J_k^\top (T_k - T^{obs}) + \sigma^2 R m_k.$$

Définition 6.2.1 *On note* $\delta\mathcal{M}_C(m_k) := \{\delta m \in \mathcal{M} : C_E.\delta m = e_k,\ l_k \leq C_I.\delta m \leq u_k\}$ *l'ensemble des perturbations de modèle* δm *respectant les contraintes d'égalité et d'inégalité du problème* (6.4).

On vérifie aisément que $\delta m \in \delta\mathcal{M}_C(m_k)$ si et seulement si $m_k + \delta m \in \mathcal{M}_C$. Dès lors on a la relation

$$\delta\mathcal{M}_C(m_k) = \mathcal{M}_C - m_k. \tag{6.5}$$

En particulier, on a $\delta\mathcal{M}_C(m_k) \neq 0$ si et seuleument si $\mathcal{M}_C \neq 0$.

Le $(PQT)_k$ est un problème convexe (voir définition 4.1.5) car H_k est semi-définie positive. Il sera donc possible d'appliquer une méthode de lagrangien augmenté pour le résoudre (voir chapitre 7).

6.2.2 Existence et unicité de la solution du Problème Quadratique Tangent

Sans les contraintes linéaires, le PQT (6.4) est équivalent au problème régularisé linéarisé (P_{lin}) (cf. (2.16)) de la tomographie de réflexion. Nous avons vu à la section 2.5.3, l'existence et l'unicité du problème linéarisé (P_{lin}). La question ici est de savoir si ce résultat est toujours valable lorsque l'on ajoute des contraintes linéaires. On a le résultat d'existence et d'unicité suivant.

Proposition 6.2.2 *On suppose que les contraintes de (P_{EI}) sont compatibles ($\mathcal{M}_C \neq \emptyset$), alors $(PQT)_k$ a une solution. Si, en plus, H_k est définie positive alors il existe une et une seule solution au problème $(PQT)_k$.*

DÉMONSTRATION. On utilise les 2 arguments suivants.
1) Le problème $(PQT)_k$ est réalisable car $\delta\mathcal{M}_C(m_k) \neq \emptyset$ (car $\mathcal{M}_C \neq \emptyset$).
2) F_k est borné car

$$F_k(\delta m) = \frac{1}{2}||J_k\delta m + T(m_k) - T^{obs}||_2^2 + \frac{\sigma^2}{2}(m_k+\delta m)^\top R(m_k+\delta m) \geq 0.$$

On sait que F_k est une fonction quadratique convexe (H_k est semi-définie positive). Ainsi, par le théorème 17.1 de [18] il existe une solution au problème $(PQT)_k$. Si on suppose, en plus, que H_k est définie positive alors cette solution est unique car, dans ce cas, F_k est strictement convexe. □

6.2.3 Conditions nécessaires et suffisantes d'optimalité du Problème Quadratique Tangent

Soit $L_{(PQT)_k} : \mathbb{R}^n \times \mathbb{R}^{n_E} \times \mathbb{R}^{n_I} \to \mathbb{R}$ le lagrangien associé au problème quadratique tangent (6.4) défini par

$$L_{(PQT)_k}(\delta m, \lambda_E, \lambda_u, \lambda_l) = F_k(\delta m) + \lambda_E^\top(C_E\delta m - e_k) + \lambda_u^\top(C_I\delta m - u_k) - \lambda_l^\top(C_I\delta m - l_k), \tag{6.6}$$

où le vecteur μ_E (resp. μ_l et μ_u) est appelé le multiplicateur de Lagrange associé aux contraintes d'égalité (resp. aux contraintes d'inégalité).

Soit $\widehat{\delta m}_k$ un minimum de $(PQT)_k$. Comme F_k est régulière ($F_k \in C^\infty$ de $\mathbb{R}^n \to \mathbb{R}$) et que les contraintes sont affines (donc qualifiées (QC-A) voir la définition 4.1.9 et la proposition 4.1.10), on peut appliquer le théorème de Karush-Kuhn-Tucker (cf. théorème 4.1.11). Il existe $\widehat{\lambda}_{E,k} \in \mathbb{R}^{n_E}$, $\widehat{\lambda}_{l,k} \in \mathbb{R}^{n_I}$ et $\widehat{\lambda}_{u,k} \in \mathbb{R}^{n_I}$ tels que les conditions nécessaires d'optimalité suivantes soient vérifiées :

$$\begin{cases} (a)\ \nabla F_k(\widehat{\delta m}_k) + C_E^\top\widehat{\lambda}_{E,k} - C_I^\top(\widehat{\lambda}_{l,k} - \widehat{\lambda}_{u,k}) = 0 \\ (b)\ C_E\widehat{\delta m}_k = e_k,\ \ l_k \leq C_I\widehat{\delta m}_k \leq u_k \\ (c)\ (\widehat{\lambda}_{l,k}, \widehat{\lambda}_{u,k}) \geq 0 \\ (d)\ \widehat{\lambda}_{l,k}^\top(C_I\widehat{\delta m}_k - l_k) = 0,\ \widehat{\lambda}_{u,k}^\top(C_I\widehat{\delta m}_k - u_k) = 0. \end{cases} \tag{6.7}$$

On note $\widehat{\lambda}_k = (\widehat{\lambda}_{E,k}, \widehat{\lambda}_{I,k})$ avec $\widehat{\lambda}_{I,k} = \widehat{\lambda}_{l,k} - \widehat{\lambda}_{u,k}$. Ces conditions sont aussi suffisante pour avoir l'optimalité de $\widehat{\delta m}_k$ (voir proposition 4.1.13).

6.2.4 Algorithme SQP local

On peut à présent schématiser l'algorithme local de programmation quadratique successive que nous utilisons pour résoudre le problème non-linéaire (P_{EI}).

Data : Choisir un modèle initial m_0.
begin
  $k = 0$
  Évaluer f_0, g_0, J_0 par la résolution du problème direct (algorithme de tracé de rayons).
  while (5.5) *n'est pas satisfaite* **do**
    ((1)) Résoudre le $(PQT)_k$ (6.4) :
      on obtient une solution primale-duale $(\widehat{\delta m}_k, \widehat{\lambda}_k)$.
    ((2)) Mettre à jour le modèle par :

$$m_{k+1} = m_k + \widehat{\delta m}_k.$$

    ((3)) Mettre à jour les multiplicateurs de Lagrange par :

$$\mu_{k+1} = \widehat{\lambda}_k.$$

    ((4)) Évaluer f_{k+1}, g_{k+1}, J_{k+1} par la résolution du problème direct (algorithme de tracé de rayons).
    ((5)) Passer à l'itération de Gauss-Newton suivante : $k := k + 1$.
  endw
end

Algorithme II.2 Algorithme SQP local appliqué à la résolution de (P_{EI})

Quelques commentaires :

1. Les commentaires de l'algorithme général II.1 s'appliquent à cet algorithme.
2. L'algorithme II.2 est un algorithme local. Il ne converge que pour m_0 proche d'une solution locale $\widehat{m}$ de (P_{EI}) (voir proposition 4.2.4). Si l'on commence à itérer à partir d'un modèle initial m_0 "trop éloigné" de la solution $\widehat{m}$, il faut alors forcer la convergence en utilisant une méthode de globalisation de l'algorithme local . Nous développerons en détail une méthode de globalisation dans la section 6.3. Cette méthode met à jour le modèle et les multiplicateurs de Lagrange aux étapes ((2)) et ((3)) de l'algorithme.

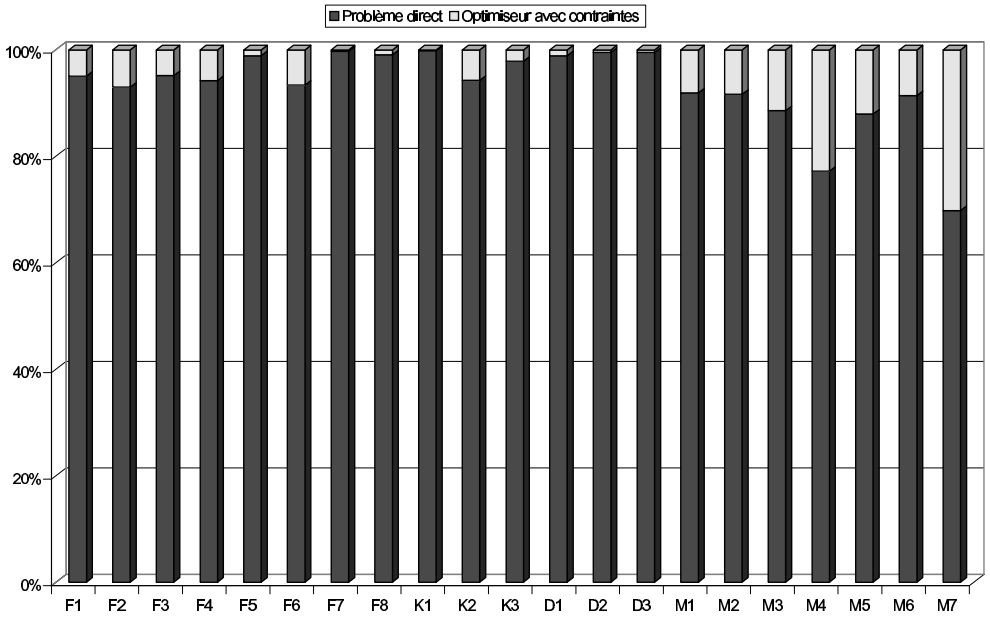

Fig. 6.2 Pourcentage du temps CPU passé dans le problème direct/solveur avec contraintes

3. Comme nous l'avons déjà remarqué dans la figure 6.2, la résolution du problème direct à l'étape ((4)) de l'algorithme représente plus de 80% du temps total de calcul (hors parallélisation et sans contrainte). La figure 6.2 confirme cette tendance lorsque l'on utilise notre optimiseur avec contraintes. En effet, on remarque que le temps CPU passé à résoudre le problème direct reste supérieur à 70% sur l'ensemble des modèles de notre bibliothèque de tomographie (voir annexe D). Il faut rester très prudent sur ces valeurs car le temps passé dans l'optimiseur avec contraintes dépend de beaucoup de paramètres : nombre de contraintes, type des contraintes, activité des contraintes en la solution, dimension du problème, etc... Ces valeurs valident notre choix pour la méthode SQP.

6.3 Globalisation de l'algorithme SQP local par recherche linéaire

L'algorithme II.2 converge vers une solution $\widehat{m}$ de (P_{EI}) à condition que le modèle initial m_0 soit dans un voisinage de $\widehat{m}$ (voir proposition 4.2.4). Afin de pallier à ce problème, il faut forcer la convergence de notre algorithme local et avoir recours aux techniques dites de "globalisation d'algorithme"[1]. Dans l'état actuel de l'art, il existe 2 grandes classes de méthodes pour globaliser un algorithme local. Ces 2 classes, déjà évoquées pour l'optimisation sans contrainte (cf. partie I) sont fondées d'une part sur les techniques de Recherche Linéaire (RL) et d'autre part sur les techniques de Régions de Confiance (RC).

[1] A ne pas confondre avec les méthodes d'optimisation globale.

Si l'on souhaite utiliser une technique de RC pour minimiser Θ, la difficulté à surmonter dans cette voie est le contrôle de la résolution approchée des $(PQT)_k$, de manière à ce que l'itéré suivant soit dans une RC prescrite. Pour qu'il en soit ainsi, il faut en effet que les contraintes du $(PQT)_k$ puissent être satisfaites de manière approchée et que le gradient du lagrangien soit annulé approximativement. Pour sa simplicité de mise en œuvre, nous avons choisi de globaliser l'algorithme SQP local par une technique de recherche linéaire (nous appliquerons donc essentiellement les résultats de la section 4.2.5). On utilise pour cela une fonction de pénalisation qui prend en compte les 2 aspects du problème à résoudre : admissibilité et optimalité. C'est une fonction que l'on fait décroître à chaque itération grâce à la recherche linéaire. Il est en général recommandé d'utiliser des fonctions de pénalisation ayant une propriété d'exactitude (voir définition 4.2.1). On comprend l'intérêt de la pénalisation exacte. Si Θ a cette propriété, ce qui ne s'obtiendra qu'avec le réglage de paramètres de pénalisation, il suffit de minimiser Θ (un problème sans contrainte) pour résoudre le problème avec contraintes. La recherche linéaire intervient pour minimiser Θ le long de directions de descente.

Notons qu'il existe des techniques de globalisation ne faisant pas intervenir de fonction de pénalisation : les méthodes de filtres (cf. [59]). Leur mise au point n'est pas terminé à ce jour. Nous ne les avons pas essayées.

Il existe de nombreuses fonctions de pénalisation exacte des problèmes non-linéaires. Parmi ces fonctions, les plus connues sont [2] :

- le lagrangien (problèmes convexes),
- le lagrangien augmenté,
- la fonction de pénalisation par norme générale.

Afin de s'assurer que la solution du $(PQT)_k$ est une direction de descente de la fonction de pénalisation choisie, on cherche une solution exacte de ce problème (voir proposition 4.2.7). Ainsi, le choix de la fonction de pénalisation est indépendant de la méthode choisie pour résoudre le $(PQT)_k$ car on cherche une solution exacte de ce problème (on verra dans le chapitre 7 suivant que nous avons choisi une méthode de lagrangien augmenté pour résoudre le $(PQT)_k$). Mais, nous avons choisi, dans la troisième classe (traitée dans le chapitre général sur l'optimisation à la section 4.2.5), le cas particulier de la fonction de pénalisation l^1 : malgré sa non-differentiabilité, c'est la fonction de pénalisation la plus facile à utiliser grâce à des conditions d'exactitude moins restrictives et la possibilité d'associer à chaque contrainte un poids de pénalisation spécifique. Cette fonction de pénalisation l_1 s'obtient en ajoutant au critère la norme l_1 pondérée des contraintes :

$$\Theta_\sigma(m) = f(m) + \Psi_E(m, \sigma_E) + \Psi_I(m, \sigma_I), \tag{6.8}$$

où Ψ_E est la fonction pénalisant la violation des contraintes d'égalité et Ψ_I celle pénalisant

[2] On notera que la fonction de pénalisation quadratique des contraintes n'est pas présente dans la liste. En effet, cette fonction n'est pratiquement jamais exacte (regarder par exemple le problème $f = \frac{x^3}{3} + x$ s.t. $x \geq 0$).

la violation des contraintes d'inégalité. Ces 2 fonctions s'écrivent

$$\begin{cases} \Psi_E(m,\sigma_E) = \sum_{i\in E} \sigma_i |e_i - C_i m| \\ \Psi_I(m,\sigma_I) = \sum_{i\in I} \sigma_i \max(C_i m - u_i, 0) + \sum_{i\in I} \sigma_i \max(l_i - C_i m, 0) \end{cases},$$

avec $\sigma = (\sigma_E, \sigma_I)$, $\sigma_E \in \mathbb{R}^{n_E}$ et $\sigma_I \in \mathbb{R}^{n_I}$ les poids de pénalisation associés respectivement aux contraintes d'égalité et d'inégalité. Sous l'hypothèse 5.1.1 de compatibilité des bornes on peut simplifier l'expression de Ψ_I par :

$$\Psi_I(m,\sigma_I) = \sum_{i\in I} \sigma_i \max(C_i m - u_i, l_i - C_i m, 0).$$

La proposition ci-dessous nous donne les conditions d'exactitude de la fonction de pénalisation Θ_σ.

Proposition 6.3.1 *Supposons que f et $c_{E\cup I^0_*}$ sont deux fois différentiables en un minimum local $\widehat{m}$ de (P_{EI}) pour lequel la condition nécessaire d'optimalité* (5.5) *est satisfaite. En ce minimum $\widehat{m}$ on suppose également que la condition suffisante d'optimalité du second ordre faible est respectée (cf. théorème* (4.1.15)*), et si $\forall(\widehat{\mu}_E, \widehat{\mu}_I) \in \widehat{\Lambda}$ [3], $\sigma_k > |\widehat{\mu}_i|$ pour $i \in E \cup I$, alors Θ_σ, définie en* (6.8)*, est une fonction de pénalisation exacte en $\widehat{m}$ et $\widehat{m}$ est un minimum local strict de Θ_σ.*

DÉMONSTRATION. Appliquer la proposition 4.2.8 en utilisant la norme $v \mapsto ||v||_P := \sum_i \sigma_i |v_i|$, dont la norme duale s'écrit

$$v \mapsto ||v||_D := \sup_{||u||_P=1} v^\top u = \max_i \frac{1}{\sigma_i}|v_i|$$

□

Rappelons que dans notre application nous cherchons à globaliser l'algorithme SQP local en faisant une RL le long de la solution $\widehat{\delta m}_k$ du problème quadratique tangent. Pour ce faire, il est impératif que $\widehat{\delta m}_k$ soit une direction de descente de la fonction de pénalisation choisie. Soit $(\widehat{\delta m}_k, \widehat{\lambda}_k)$ une solution du $(PQT)_k$ (on suppose que m_k n'est pas un point stationnaire de (P_{EI})). Si H_k est définie positive alors on peut appliquer la proposition 4.2.7 : la solution $\widehat{\delta m}_k$ du $(PQT)_k$ est une direction de descente de Θ_σ en m_k si $\sigma > ||\widehat{\lambda}_k||_D$.

Ainsi, il est important que la condition

$$\sigma_i > |(\widehat{\lambda}_k)_i|, \quad i \in E \cup I, \tag{6.9}$$

soit respectée pour toutes les itérations de Gauss-Newton, car la solution $\widehat{\delta m}_k$ du $(PQT)_k$ est alors une direction de descente de Θ_σ en m_k. Afin de satisfaire cette condition, on

[3] $\widehat{\Lambda}$ est l'ensemble non vide des multiplicateurs de Lagrange en $\widehat{m}$

modifie les poids de pénalisation σ_i de Θ_σ à certaines itérations de Gauss-Newton. On note alors, Θ_{σ_k} la fonction de pénalisation associée au poids de pénalisation σ_k de l'itération k. Nous donnons ci-dessous l'algorithme utilisé pour adapter les poids de pénalisation au cours des itérations de Gauss-Newton (algorithme adapté des règles proposées par [113] et proposé dans le chapitre 15 de [18]).

Data : Poids de pénalisation de l'itération $k-1$ de GN, $\sigma_{k-1} \in \mathbb{R}^{n_C}$.
Constante $\bar{\sigma} > 0$.
begin
  for $i = 1, ..., n_C$ **do**
    if $(\sigma_{k-1})_i \geq 1.1\,(|(\widehat{\lambda}_k)_i| + \bar{\sigma})$ **then**

$$(\sigma_k)_i = ((\sigma_{k-1})_i + |(\widehat{\lambda}_k)_i| + \bar{\sigma})/2. \tag{6.10}$$

    else
      if $(\sigma_{k-1})_i \geq |(\widehat{\lambda}_k)_i| + \bar{\sigma}$ **then**

$$(\sigma_k)_i = (\sigma_{k-1})_i. \tag{6.11}$$

      else

$$(\sigma_k)_i = \max(1.5\,(\sigma_{k-1})_i, |(\widehat{\lambda}_k)_i| + \bar{\sigma}). \tag{6.12}$$

      endif
    endif
  endfor
end

Algorithme II.3 Règles de mise à jour des poids de pénalisation σ_k

Quelques remarques sur cet algorithme :

1. Cet algorithme cherche à réaliser 3 objectifs : satisfaire (6.9), stabiliser σ_k et ne pas avoir σ_k trop grand. Le premier objectif est entièrement satisfait grâce aux équations (6.10), (6.11) et (6.12) de l'algorithme). Dans le cas où les multiplicateurs de Lagrange du $(PQT)_k$ "n'explosent pas", le deuxième objectif n'est satisfait que si l'on considère l'algorithme sans la règle de diminution de σ_k (équations (6.10)). Enfin, le troisième objectif, qui permet d'éviter d'avoir des poids de pénalisation trop forts qui pourraient en pratique nuire au bon fonctionnement de la méthode, est réalisé par le biais de la règle de diminution de σ_k (équation (6.10)).
2. Les figures 6.3, 6.4 et 6.5 montrent l'évolution du pourcentage des poids de pénalisation qui sont augmentés au cours des itérations de GN pour les trois modèles F3, K2 et M6. On observe, dans ces 3 exemples, que ce pourcentage diminue, ce qui montre bien que l'on va vers une stabilisation des poids de pénalisation.
3. La constante positive $\bar{\sigma}$ permet d'assurer avec sécurité la réalisation de la condi-

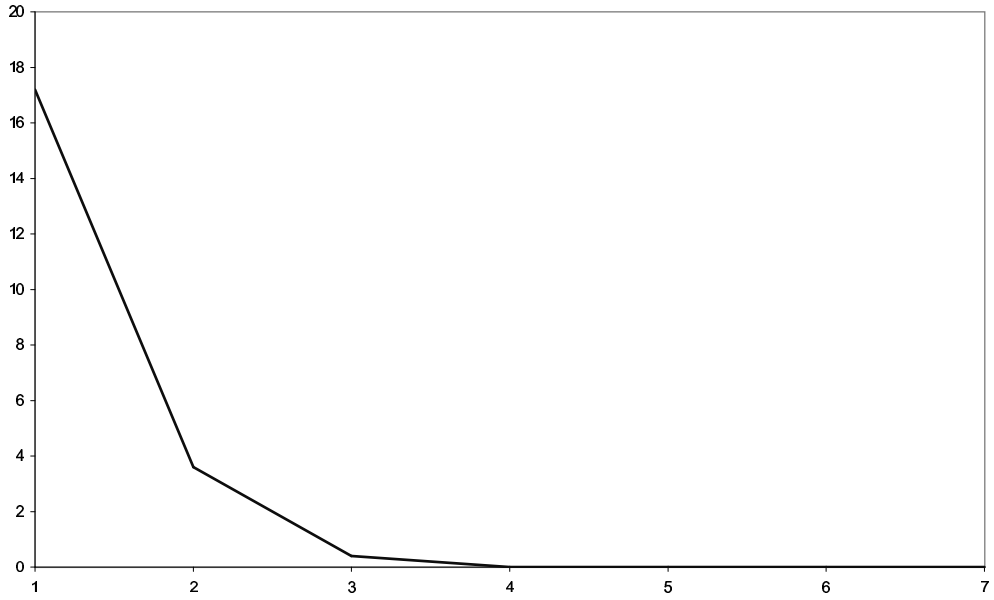

Fig. 6.3 Modèle F3 : évolution du pourcentage de poids augmentés au cours des itérations de GN

tion 6.9. Cette sécurité est essentielle pour prouver la convergence de la méthode SQP globalisée par de la recherche linéaire (voir proposition 6.3.4).

La globalisation de l'algorithme SQP local s'obtient en remplaçant les étapes ((3)) et ((4)) de l'algorithme II.2 par l'algorithme suivant :

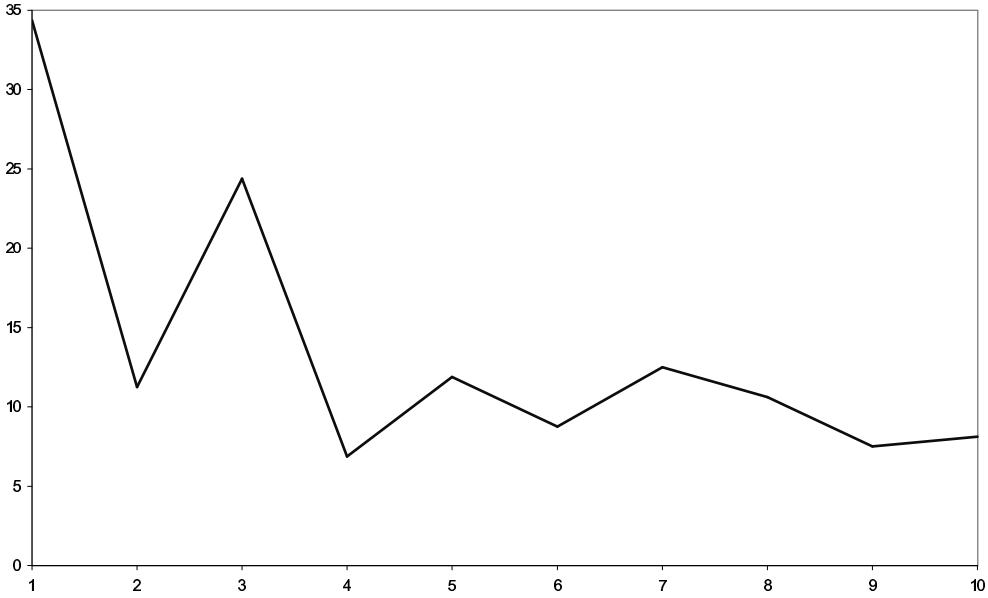

Fig. 6.4 Modèle K2 : évolution du pourcentage de poids augmentés au cours des itérations de GN

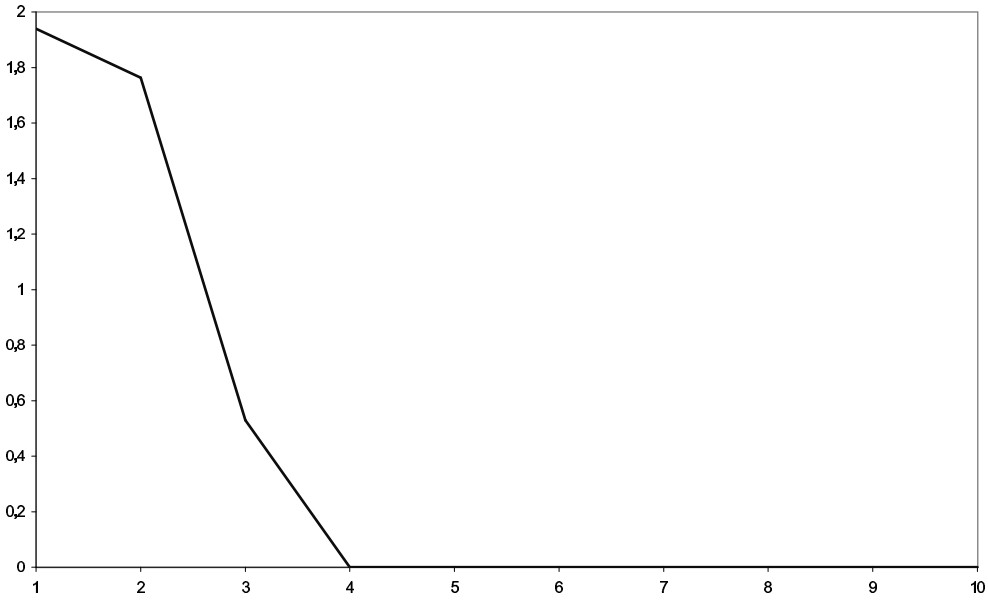

Fig. 6.5 Modèle M6 : évolution du pourcentage de poids augmentés au cours des itérations de GN

Data : Poids de pénalisation de l'itération $k-1$ de GN : σ_{k-1}.
Constante de sécurité $\bar{\sigma} > 0$.
Constante $0 < \omega < \frac{1}{2}$.

begin

((3.1)) Mise à jour des poids de pénalisation σ_k par l'algorithme II.3.
((3.2)) Déterminer le pas α_k par "rebroussement", i.e. en prenant le plus grand α_k dans $\{1, \frac{1}{2}, \frac{1}{4}, ...\}$ tel que

$$\begin{cases} m_k + \alpha_k \widehat{\delta m}_k \in \mathcal{M} \\ \Theta_{\sigma_k}(m_k + \alpha_k \widehat{\delta m}_k) \leq \Theta_{\sigma_k}(m_k) + \omega \alpha_k \Diamond_k, \end{cases} \tag{6.13}$$

avec

$$\Diamond_k = g_k^\top \widehat{\delta m}_k - \Psi_E(m_k, (\sigma_k)_E) - \Psi_I(m_k, (\sigma_k)_I). \tag{6.14}$$

((3.3)) Mettre à jour le modèle par :

$$m_{k+1} = m_k + \alpha_k \widehat{\delta m}_k.$$

((4.0)) Mettre à jour les multiplicateurs de Lagrange par :

$$\mu_{k+1} = \mu_k + \alpha_k(\widehat{\lambda}_k - \mu_k). \tag{6.15}$$

end

Algorithme II.4 Globalisation de l'algorithme SQP local : modification des étapes (3) et (4) de l'algorithme II.2

Quelques remarques sur cet algorithme :

1. La condition (6.13) est une condition proche de la condition d'Armijo appliquée à la fonction non-différentiable Θ_σ. La condition d'Armijo s'écrit

$$\Theta_\sigma(m_k + \alpha_k \widehat{\delta m}_k) \leq \Theta_\sigma(m_k) + \omega \alpha_k \Theta'_\sigma(m_k; \widehat{\delta m}_k), \tag{6.16}$$

avec $\Theta'_\sigma(m_k; \widehat{\delta m}_k)$ qui représente la dérivée directionnelle de Θ_σ dans la direction $\widehat{\delta m}_k$. Cependant cette dérivée directionnelle n'est pas facile à calculer en pratique. Or, d'après la proposition 15.1 de [18], $\Diamond_k$ est un majorant négatif de $\Theta'_\sigma(m_k; \widehat{\delta m}_k)$, facile à calculer. On montre que les résultats de convergence ne sont pas affectés par la substitution de $\Theta'_\sigma(m_k; \widehat{\delta m}_k)$ par $\Diamond_k$ dans la condition d'armijo (6.16) et on obtient alors la pseudo condition d'armijo (6.13).

2. Dans l'étape ((3.2)) de l'algorithme, on utilise un algorithme classique de rebroussement pour trouver le pas α_k qui vérifie la pseudo condition d'armijo (6.16). Il est préférable en pratique d'utiliser une technique d'interpolation (voir Chapitre 3 de [18]).

3. Le choix de la constante $0 < \omega < \frac{1}{2}$ est important en pratique pour ne pas empêcher l'admissibilité asymptotique du pas unité (c'est à dire qu'il existe un indice k_1 tel que $\forall k \geq k_1 \ \alpha_k = 1$). Cependant si on prend $0 < \omega < 1$, la convergence globale de la méthode est toujours assurée (voir proposition 6.3.4). Dans notre programme, nous avons choisi $\omega = 10^{-3}$. L'admissibilité asymptotique du pas unité ($\alpha_k = 1$ pour k grand) est très importante car, d'une part, elle permet de retrouver l'algorithme local à convergence rapide proche de la solution, et d'autre part elle permet de ne pas recalculer des problèmes directs (chaque évaluation de la fonction f en $m_k + \alpha_k \widehat{\delta m}_k$ nécessite la résolution d'un problème direct qui coûte cher en temps CPU). Cependant, s'agissant d'un algorithme de Gauss-Newton, rien n'assure que le pas unité soit accepté asymptotiquement (effet Maratos et approximation gauss-newtonienne).
4. A l'étape ((4)) de l'algorithme on met à jour les variables duales μ_k en se servant du pas α_k calculé pour la variable primale m_k. Dans notre cas où les contraintes sont linéaires, l'approximation du hessien du lagrangien ne dépend que de l'approximation du hessien de f ($\nabla^2_{mm} L_{(P_{EI})}(m_k) = \nabla^2_{mm} f(m_k) \simeq H_k$). Les multiplicateurs de Lagrange μ_k du problème non-linéaire n'interviennent donc pas dans l'approximation de $\nabla^2_{mm} L_{(P_{EI})}(m_k)$. A première vue, la mise à jour des multiplicateurs ne sert donc à rien et on pourrait éliminer l'étape ((4)) de l'algorithme précédent. Toutefois, on verra dans le chapitre 7 que nous avons choisi un algorithme dual pour résoudre le $(PQT)_k$. Ce type d'algorithme est d'autant plus efficace qu'il est "démarré à chaud". C'est à dire si l'on dispose d'une estimation des multiplicateurs de Lagrange. Or les multiplicateurs optimaux sont les multiplicateurs de Lagrange du PQT en la solution. Dès lors les μ_k serviront à initialiser le solveur du $(PQT)_{k+1}$.
5. Une propriété très intéressante de l'algorithme II.4 est que si le pas unité est accepté à une itération k de Gauss-Newton, alors tous les modèles suivants générés par l'algorithme restent à l'intérieur de l'ensemble admissible (cette propriété découle de la linéarité des contraintes du problème non-linéaire). La figure 6.6 montre que, en pratique, le pas unité est accepté très rapidement : plus de 75% des modèles de notre bibliothèque acceptent le pas unité dès la première itération de GN.

Proposition 6.3.2 *Nous considérons l'algorithme SQP globalisé (défini par les 3 algorithmes : II.2, II.3, et II.4). Soit un modèle initial $m_0 \in \mathcal{M}$, si il existe une itération $\bar{k}$ de Gauss-Newton telle que le pas unité ($\alpha_{\bar{k}} = 1$) est accepté par la RL, alors $\forall k > \bar{k}\ m_k \in \mathcal{M}_C$ (voir définition 5.1.2).*

DÉMONSTRATION. On suppose qu'à l'itération $\bar{k}$ de Gauss-Newton le pas unité ($\alpha_{\bar{k}} = 1$) est accepté par la RL. La mise à jour du modèle s'écrit $m_{\bar{k}+1} = m_{\bar{k}} + \widehat{\delta m}_{\bar{k}}$. Comme $\widehat{\delta m}_{\bar{k}}$ est solution du $(PQT)_{\bar{k}}$ il vérifie les conditions de KKT du $(PQT)_{\bar{k}}$ (voir section 6.2.3) et plus particulièrement les équations d'admissibilité suivantes :

$$C_E \widehat{\delta m}_{\bar{k}} = e_{\bar{k}}, \quad l_{\bar{k}} \leq C_I \widehat{\delta m}_{\bar{k}} \leq u_{\bar{k}},$$

Or, on sait que $e_{\bar{k}} = e - C_E m_{\bar{k}}$, $l_{\bar{k}} = l - C_I m_{\bar{k}}$ et $u_{\bar{k}} = u - C_I m_{\bar{k}}$ d'où :

$$C_E m_{\bar{k}+1} = e, \;\; l \leq C_I m_{\bar{k}+1} \leq u.$$

On a démontré que $m_{\bar{k}+1} \in \mathcal{M}_C$. Par récurrence $m_k \in \mathcal{M}_C \; \forall k \geq \bar{k}$. □

6. Une conséquence importante de la proposition 6.3.2 est que si il existe une itération $\bar{k}$ de Gauss-Newton telle que le pas unité ($\alpha_{\bar{k}} = 1$) est accepté par la RL, alors à partir de l'itération $\bar{k} + 1$, on peut substituer la pseudo-condition d'Armijo (voir équation (6.13)) par

$$\begin{cases} m_k + \alpha_k \widehat{\delta m_k} \in \mathcal{M} \\ f(m_k + \alpha_k \widehat{\delta m_k}) \leq f(m_k) + \omega \alpha_k g_k^\top \widehat{\delta m_k} \end{cases}$$

qui est la condition d'Armijo sur f. Cette substitution découle des définitions de Ψ_E et Ψ_I et de 6.3.2 : $\forall k > \bar{k}$ on a $\Theta_{\sigma_k}(m_k + \alpha_k \widehat{\delta m_k}) = f(m_k + \alpha_k \widehat{\delta m_k})$, $\Theta_{\sigma_k}(m_k) = f(m_k)$ et $\diamondsuit_k = g_k^\top \widehat{\delta m_k}$.

7. Dans les figures 6.7 et 6.8 nous avons représenté respectivement l'évolution de la fonction coût Θ_{σ_k} et le gradient du lagrangien de (P_{EI}) (voir équation 5.6) en fonction des itérations de Gauss-Newton pour tous les problèmes de notre bibliothèque de tomographie (voir annexe D). Comme ces 2 quantités sont décroissantes sur la majorité de nos problèmes on en déduit que, en pratique, l'algorithme SQP que nous avons développé fonctionne bien. Notons que, le gradient du lagrangien de (P_{EI}) n'est pas décroissant pour les exemples K2 et K3 car les problèmes quadratiques tangents ne sont pas résolus avec suffisamment de précision.

Hypothèses 6.3.3 *Les matrices $\{H_k\}$ et $\{H_k^{-1}\}$ sont bornées*

Comme f est de classe $C^{1,1}$ dans $\mathcal{M}$ et que nous utilisons la fonction de pénalisation exacte l_1, alors on a la proposition suivante qui découle directement du théorème de convergence globale de la méthode SQP avec recherche linéaire (cf. Théorème 15.2 de [18])

Proposition 6.3.4 *Sous l'hypothèse 6.3.3, lorsque l'on utilise l'algorithme SQP globalisé (défini par les 3 algorithmes : II.2, II.3, et II.4) en partant d'un modèle initial $m_0 \in \mathcal{M}$, alors l'une des situations suivantes se produit :*

(i) la suite $\{(\sigma_k)_E, (\sigma_k)_I\}$ n'est pas bornée, et dans ce cas là, la suite $\{\lambda_{E,k}, \lambda_{I,k}\}$ n'est pas bornée non plus,

(ii) il existe un indice k_1 tel que $\forall k \geq k_1$ et $\forall i \in E \cup I$ on a $(\sigma_{k+1})_i = (\sigma_k)_i$, et tel que, au moins l'une des situations suivantes se produit :

(a) $dist(m_k, \mathcal{M}^c) \longrightarrow 0$

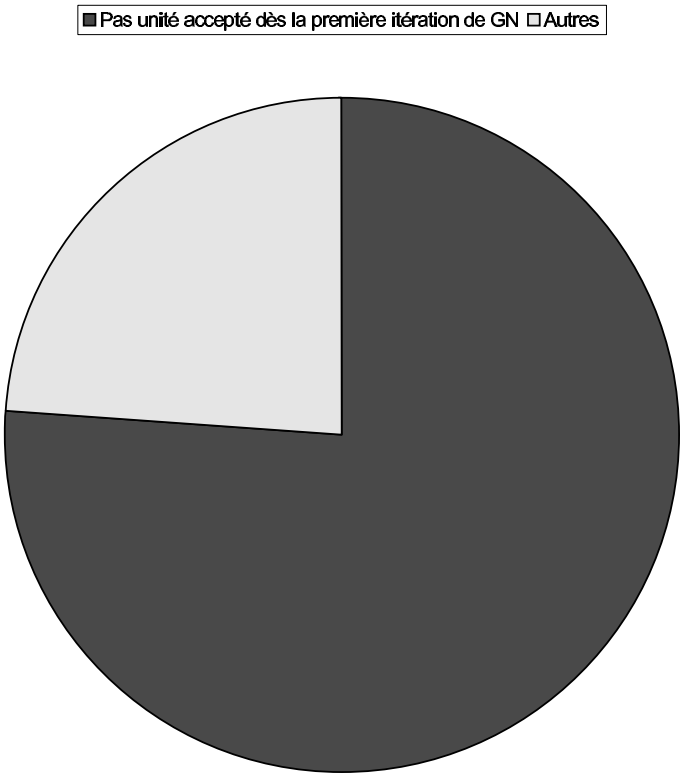

Fig. 6.6 Pourcentage de pas unité accepté à la première itération de GN

$$\textit{(b)}\quad \begin{cases} \nabla_m L_{(P_{EI})}(m_k, \lambda_k) \longrightarrow 0 \\ C_E m_k - e \longrightarrow 0 \\ \max(C_I m_k - u, l - C_I m_k, 0) \longrightarrow 0 \\ (\lambda_{l,k}, \lambda_{u,k}) \geq 0 \\ \lambda_{l,k}^\top (C_I m_k - u) \longrightarrow 0, \quad \lambda_{u,k}^\top (C_I m_k - l) \longrightarrow 0. \end{cases}$$

Parmi les situations décrites par la proposition 6.3.4 de convergence globale, seul le cas (ii-b) est satisfaisant. En effet, dans ce cas, tous les points d'accumulation de $\{(m_k, \lambda_k)\}$ sont solutions des conditions nécessaires d'optimalité du problème non-linéaire (P_{EI}) (cf. condition de KKT (5.5)). La situation (ii-a) peut se produire en tomographie de réflexion lorsque par exemple l'algorithme du tracé de rayons ne parvient pas à calculer suffisamment de temps de trajet sur le modèle solution de (P_{EI}) (ce modèle est alors à l'extérieur de $\mathcal{M}$).

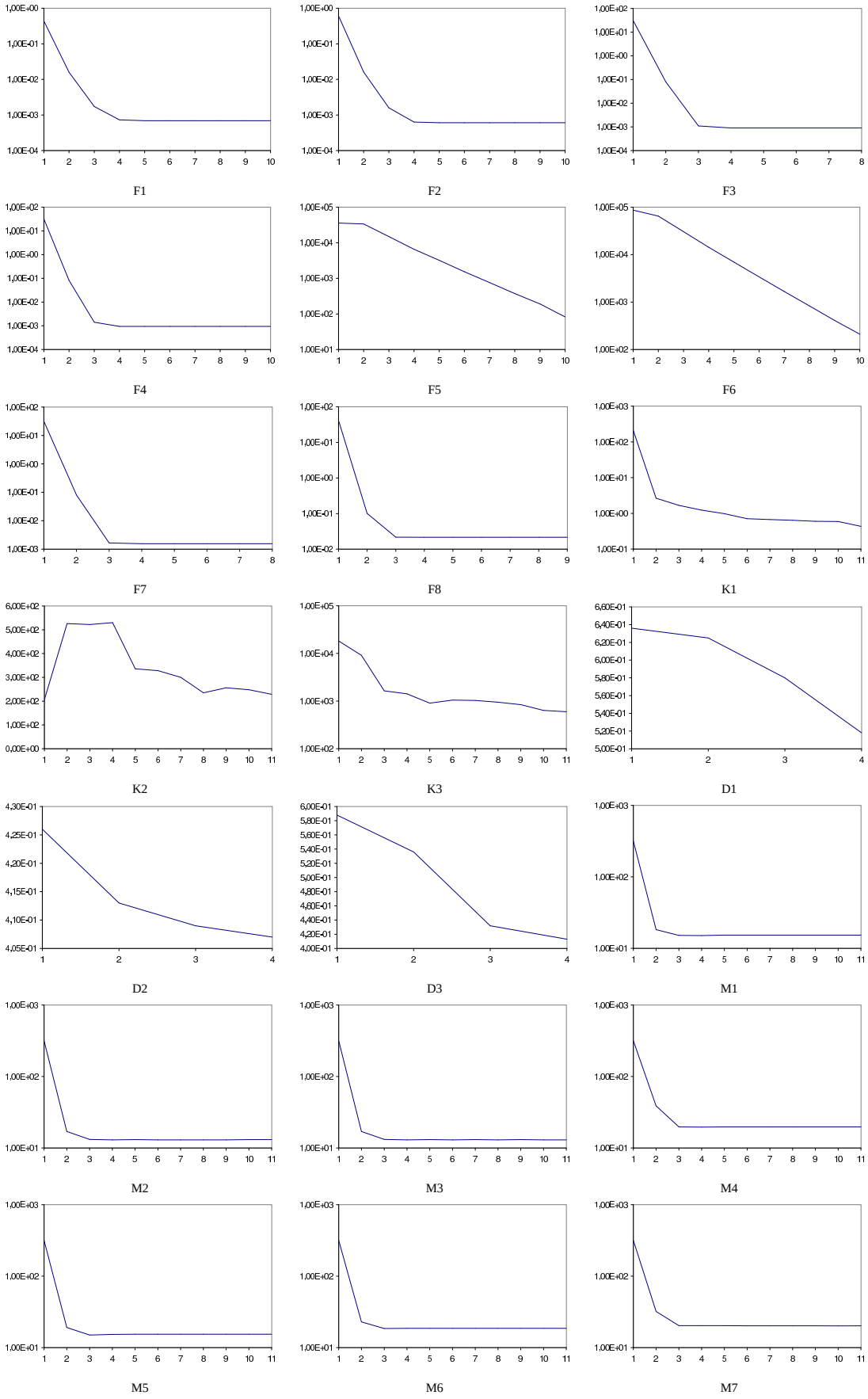

Fig. 6.7 Evolution de la fonction coût Θ_{σ_k} au cours des itérations de GN

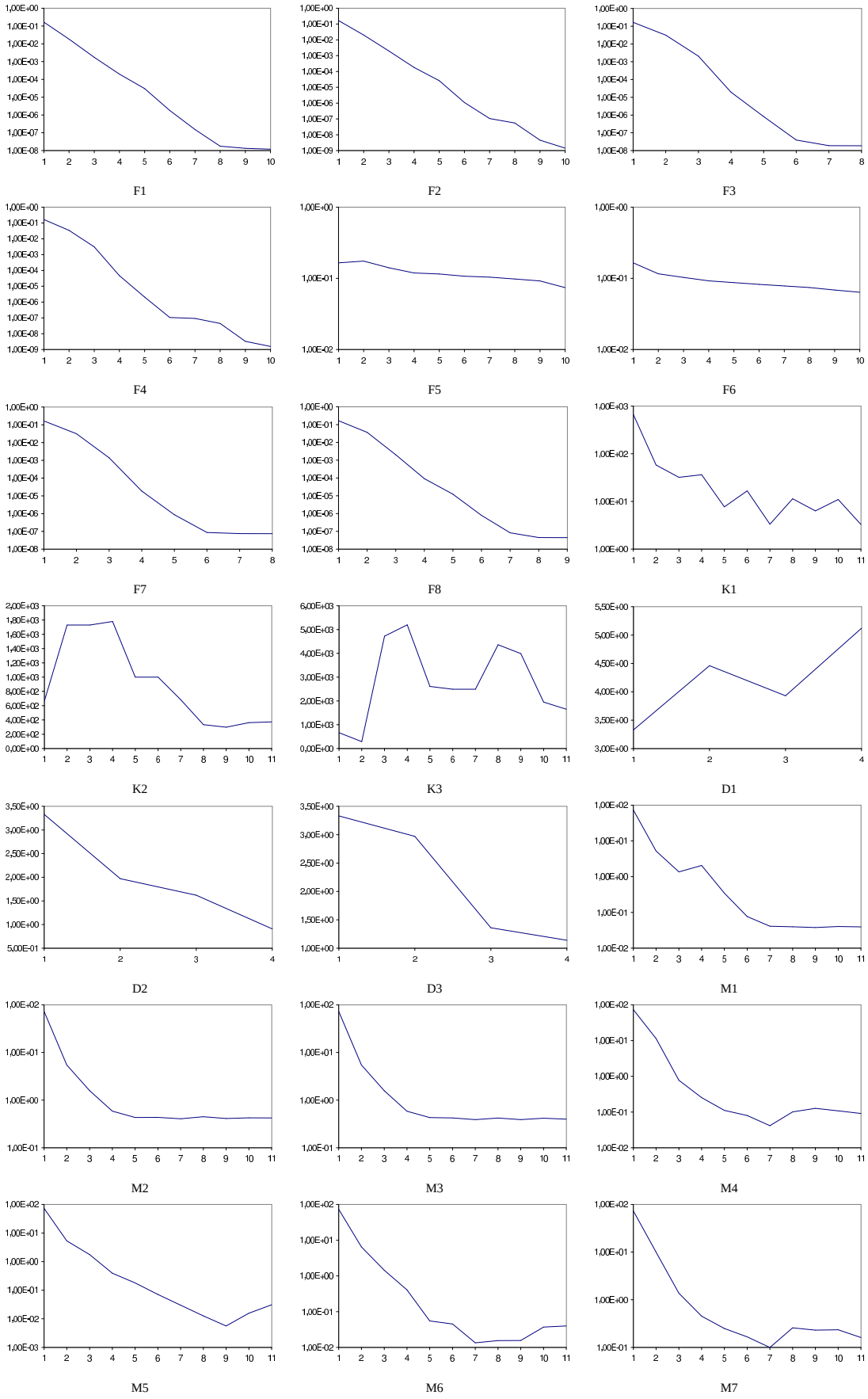

Fig. 6.8 Evolution du gradient du lagrangien de (P_{EI}) au cours des itérations de GN

Chapitre 7

Résolution du Problème Quadratique Tangent

D'une manière générale, nous appelons problèmes quadratiques convexes les problèmes d'optimisation consistant à minimiser une fonction quadratique convexe soumise à des contraintes linéaires d'égalité et/ou d'inégalité. Le (PQT) (6.4), que nous devons résoudre à chaque itération de Gauss-Newton, fait donc partie de cette classe. L'objet de ce chapitre est dans un premier temps de faire un état de l'art sur les différentes méthodes envisageables pour résoudre de tels problèmes. Puis, à partir de cet éclairage, de motiver et d'expliquer les détails de notre méthode de résolution par lagrangien augmenté et de décrire les spécificités de notre code de calcul QPAL (Quadratic Programming and Augmented Lagrangian).

Par souci de clarté, nous abandonnerons l'indice k représentant les itérations de GN dans tous les chapitres traitant de la résolution du PQT. Le (PQT) s'écrit alors :

$$(PQT) \begin{cases} \min\limits_{\delta m \in \mathbb{R}^n} F(\delta m) \\ C_E \delta m = e \\ l \leq C_I \delta m \leq u. \end{cases} \tag{7.1}$$

Notons que dans notre application, nous supposons que les contraintes sont équilibrées (la norme l_2 de chaque ligne de C est égale à 1), ou qu'elles ont été équilibrées par l'application d'un processus de pré-traitement/post-traitement du problème (PQT). Dans cette direction, il existe de nombreuses recettes pour rendre le problème (PQT) plus simple à résoudre en le transformant en un problème plus aisé à résoudre (voir les travaux récents de [82]).

7.1 Quelques méthodes de résolution de problèmes quadratiques convexes

Il existe au moins 3 grandes classes de méthodes d'optimisation pour résoudre des problèmes quadratiques convexes. Ces trois classes de méthodes sont les suivantes :

- méthode d'Activation de Contraintes (AC)
- méthode de Points Intérieurs (PI)
- méthode de Lagrangien Augmenté (LA)

Dans les 2 sous-sections suivantes, nous présenterons succinctement les 2 premières classes de méthodes et expliquerons pourquoi ces méthodes ne sont pas très adaptées à la résolution du problème (PQT). Dans la section 7.2, nous motiverons le choix d'une méthode de LA pour résoudre le (PQT) et nous développerons en détail cette méthode.

7.1.1 Méthode d'activation de contraintes

Voici ci-dessous une définition utile pour la suite :

Définition 7.1.1 *On note $\mathcal{A}(\delta m)$ l'ensemble des contraintes d'inégalité actives en δm :*

$$\mathcal{A}(\delta m) = \{i \in I : C_i \delta m = l_i ou C_i \delta m = u_i\}.$$

L'efficacité d'une méthode d'AC dépend d'un certain nombre d'options. Dans le chapitre suivant, nous exposerons et discuterons des choix pour définir notre code QPAL. Nous commençons, dans cette section, par décrire le fonctionnement d'une méthode d'AC générale. La méthode d'AC procède en identifiant petit à petit l'ensemble des indices des contraintes actives en la solution de (PQT). Elle gère pour cela une suite d'ensembles de travail $\mathcal{W} \subset \mathcal{A}(\delta m)$. L'idée est de bloquer (ou d'activer) toutes les contraintes d'inégalité de (PQT) dont les indices appartiennent à l'ensemble de travail courant $\mathcal{W}$. En d'autres termes, toutes les contraintes d'inégalité de (PQT) dont les indices sont dans $\mathcal{W}$ sont transformées en contraintes d'égalité. On ne prend pas nécessairement $\mathcal{W} = \mathcal{A}(\delta m)$, car dans certains cas, on peut supputer que des contraintes actives en δm ne le resteront pas en la solution. Cette transformation donne naissance au nouveau problème :

$$(PQT)\Big|_{\mathcal{W}} \begin{cases} \min\limits_{\delta m \in \mathbb{R}^n} \ F(\delta m) \\ C_E \delta m = e \\ C_I \delta m = (l_i | u_i) \ \ i \in \mathcal{W} \\ l_i \leq C_I \delta m \leq u_i \ \ i \notin \mathcal{W}. \end{cases} \tag{7.2}$$

Ainsi, au lieu de résoudre directement le problème (PQT), on résout une suite de sous-problèmes quadratiques avec contraintes d'égalité. Les inégalités (dernière condition de (7.2)) ne sont pas prises en comptent directement, mais elles servent à déclencher le passage de la résolution d'un sous-problème à un autre en modifiant l'ensemble $(\mathcal{W})$. Si la direction de déplacement trouvée par la résolution du dernier sous-problème n'est

pas admissible par rapport à une ou plusieurs de ces inégalités, alors il faudra en rajouter au moins une dans l'ensemble $(\mathcal{W})$. Dans le cas contraire, il faudra soit enlever (ou désactiver) une ou plusieurs contraintes de l'ensemble $(\mathcal{W})$ ne satisfaisant pas à l'avant dernière condition de KKT de (PQT) (la condition (c) de signe des multiplicateurs de Lagrange, voir section (6.2.3)), soit interrompre l'algorithme et renvoyer la solution satisfaisant aux conditions de KKT de (PQT).

Dans la méthode générale AC, l'ensemble de travail ne change généralement que d'un indice entre chaque étape. Rarement plus d'une contrainte est activée (c'est à dire qu'elle entre dans $\mathcal{W}$) à chaque itération, car le plus souvent la direction de déplacement ne vient buter que sur une borne à la fois. D'autre part, le seul moyen connu pour s'assurer que la direction de déplacement, calculée après désactivation de contraintes (c'est-à-dire qu'on enlève des indices de $\mathcal{W}$), soit une direction de descente de F est de ne désactiver qu'une seule contrainte à la fois. Voir le livre de [72] qui donne plus de détails sur cet algorithme. Notons que l'on connaît aussi des algorithmes d'activation de contraintes travaillant sur les variables duales : voir [76]. Ainsi, la méthode d'AC est fortement pénalisée par l'aspect combinatoire du problème : le nombre d'étapes (ou d'ensembles de travail) nécessaires pour atteindre la convergence croît rapidement lorsque l'on augmente le nombre de contraintes d'inégalité du problème.

Nous avons donc éliminé la méthode d'AC car elle semble être peu efficace pour résoudre notre problème (PQT) : un problème quadratique convexe sujet à un grand nombre de contraintes d'inégalité linéaire[1]. Cependant, dans la section 8.1, nous verrons qu'elle est très efficace pour résoudre des problèmes quadratiques convexes sujet uniquement à des contraintes de borne, pourvu qu'elle soit associée à l'algorithme du Gradient avec Projection (GP). Lorsque les contraintes sont linéaires générales comme dans le (PQT), la projection est trop coûteuse (c'est un problème quadratique convexe sans particularité) pour être utilisée.

7.1.2 Méthode de points intérieurs

Le principe de cette méthode a déjà été évoqué dans le cadre général de la résolution de problèmes d'optimisation non-linéaire (voir section 4.2.4). Cette méthode est reconnue pour être très efficace sur une large variété de problèmes d'optimisation quadratiques convexes. Cependant, il nous a semblé que cette méthode n'est pas très appropriée pour résoudre les problèmes quadratiques tangents issus de la méthode SQP (voir chapitre précédent). En effet, rappelons que la méthode SQP possède la propriété importante d'identification finie des contraintes actives (voir théorème 4.2.4) et que par conséquent il est souhaitable d'utiliser une méthode de résolution du PQT qui tire parti de cette information (voir point numéro 2 de la remarque 4.2.5). Or, la méthode des points intérieurs peut difficilement exploiter la connaissance des contraintes actives car les itérés qu'elle génère restent à l'intérieur de l'ensemble admissible.

[1] cette technique est utilisée dans [71]

7.2 Algorithme du lagrangien augmenté en tomographie de réflexion

Nous avons déjà discuté de l'utilisation de la méthode du LA pour la résolution de problèmes d'optimisation non-linéaires à la section 4.2.2. Dans le cadre de la résolution de (PQT), l'algorithme du lagrangien augmenté nous semble bien adapté à notre application en tomographie de réflexion pour les 3 raisons suivantes :

Remarques 7.2.1

1. *Contrairement à l'algorithme des PI (voir 7.1.2), nous verrons que l'algorithme du LA peut tirer parti de la connaissance des contraintes actives. On peut ainsi espérer que la combinatoire des problèmes quadratiques tangents soit de plus en plus facile à gérer par la méthode du LA au fur et à mesure que la convergence de la méthode SQP progresse.*
2. *L'algorithme du LA peut être mis en œuvre de telle manière qu'il ne demande aucune factorisation de matrice. Or, dans nos problèmes de tomographie qui peuvent contenir des dizaines de milliers de contraintes, c'est un atout important puisque la factorisation peut s'avérer coûteuse (c'est aussi l'un des arguments principaux de [61] pour l'utilisation du LA lors de la résolution de problèmes aux limites pour des équations ou inéquations aux dérivées partielles).*
3. *Dans les méthodes de LA il n'est pas nécessaire de faire tendre le paramètre d'augmentation r_k vers $+\infty$ pour assurer la convergence (voir section 4.2.2). Ainsi, les systèmes linéaires à résoudre avec l'algorithme de LA sont susceptibles d'être mieux conditionnés que ceux issus d'une méthode de pénalisation ou de PI (pour la méthode classique des points intérieurs on a vu que ce conditionnement a toutes les chances d'exploser lorsque μ tends vers 0 - voir point numéro 1 de la remarque 4.2.6). Dans la remarque 8.2.3 nous développerons plus en détails cet argument.*

Rappelons que les premières applications d'une méthode d'optimisation avec contraintes en tomographie de réflexion, voir [75], utilisent justement une méthode de LA. La méthode que nous proposons dans la suite s'inspire de ces travaux en améliorant notamment l'efficacité de la résolution des problèmes de Lagrange (voir chapitre 8).

7.2.1 Résolution du PQT par la méthode du lagrangien augmenté : le code QPAL

Nous allons maintenant détailler la méthode de lagrangien augmenté telle que nous l'avons développée dans le code QPAL.

Relaxation des contraintes par le lagrangien augmenté

Afin d'écrire le lagrangien augmenté du problème (PQT) en ne relaxant que les contraintes d'égalité, nous utilisons la technique proposée dans [75] (voir aussi [133] ou [37]) qui consiste à introduire des variables $y \in \mathbb{R}^{n_I}$ pour les contraintes d'inégalité. Le (PQT) (équation (7.1)) se reformule alors sous la forme équivalente :

$$\begin{cases} \min\limits_{\delta m \in \mathbb{R}^n, y \in \mathbb{R}^{n_I}} & F(\delta m) \\ C_E \delta m = e \\ C_I \delta m = y \\ l \leq y \leq u. \end{cases} \tag{7.3}$$

Suivant l'approche classique de [93] et [126], le lagrangien augmenté du (PQT) s'obtient par relaxation des contraintes d'égalité de (7.3) avec un facteur (ou paramètre) d'augmentation $r > 0$. C'est la fonction $L_r(\delta m, y, \lambda) : \mathbb{R}^n \times \mathbb{R}^{n_I} \times \mathbb{R}^{n_C} \mapsto \mathbb{R}$ définie par

$$L_r(\delta m, y, \lambda) = F(\delta m) + \xi_E(\delta m, \lambda_E) + \xi_I(\delta m, y, \lambda_I), \tag{7.4}$$

avec $\lambda = (\lambda_E, \lambda_I)$,

$$\begin{cases} \xi_E(\delta m, \lambda_E) = \lambda_E^\top (C_E.\delta m - e) + \frac{r}{2} ||C_E.\delta m - e||^2 \\ \xi_I(\delta m, y, \lambda_I) = \lambda_I^\top (y - C_I.\delta m) + \frac{r}{2} ||y - C_I.\delta m||^2. \end{cases}$$

Nous prenons une même valeur de r pour chaque contrainte car nous supposons que les contraintes sont équilibrées (la norme l_2 de chaque ligne de C est égale à 1).

En prenant la convention que $\inf_\emptyset = +\infty$, le problème (PQT) est équivalent au problème primal suivant :

$$\inf_{\delta m, l \leq y \leq u} \sup_{\lambda} L_r(\delta m, y, \lambda). \tag{7.5}$$

Le problème dual : équivalence avec le problème quadratique tangent

Le problème dual de (PQT) associé au lagrangien augmenté s'obtient en inversant l'inf et le sup dans (7.5) (voir chapitre 12 de [94] et chapitre 13 de [68]).

Définition 7.2.2 *On appelle problème dual de* (7.5)*, le problème d'optimisation*

$$(PD) : \sup_{\lambda} \inf_{\delta m, l \leq y \leq u} L_r(\delta m, y, \lambda). \tag{7.6}$$

Lorsque le (PQT) a une solution, sa convexité implique que $\forall r \geq 0$, le lagrangien augmenté L_r a un point-selle sur $\mathbb{R}^{n+n_I} \times \mathbb{R}^{n_C}$. On définit alors la fonction duale régularisée $\delta_r : \mathbb{R}^{n_C} \mapsto \mathbb{R}$ associée au lagrangien augmenté (7.4) par :

$$\delta_r(\lambda) = - \inf_{\delta m, l \leq y \leq u} L_r(\delta m, y, \lambda). \tag{7.7}$$

Cette fonction est la régularisée de Moreau-Yosida de la fonction duale classique (voir [135] et chapitre 9) définie par

$$\delta_0(\lambda) = -\inf_{\delta m, l\leq y\leq u} L_0(\delta m, y, \lambda).$$

Le problème dual (7.6) consiste donc à minimiser la fonction duale régularisée :

$$(PD) : \inf_{\lambda} \delta_r(\lambda). \tag{7.8}$$

Définition 7.2.3 *On appelle problème de Lagrange en λ associé au lagrangien augmenté* (7.4)*, le problème d'optimisation :*

$$(PL) : \inf_{\delta m, l\leq y\leq u} L_r(\delta m, y, \lambda). \tag{7.9}$$

Mise en œuvre de la méthode du lagrangien augmenté

L'algorithme du LA résout le (PQT) en résolvant le problème dual (7.8). Il génère ainsi une suite de multiplicateurs $\{\lambda^j\}$ qui converge vers un multiplicateur optimal de (7.3), disons $\widehat{\lambda}$; celui-ci vérifie donc les conditions d'optimalité de KKT de (6.7). Pour un jeu de multiplicateurs de Lagrange λ^j, l'algorithme associe des valeurs primales $(\delta m^{j+1}, y^{j+1})$, solution arbitraire du problème de Lagrange, et met ensuite à jour λ^j.

De manière plus précise, l'itération j de l'algorithme du lagrangien augmenté se décompose en deux étapes :

- La première étape consiste à résoudre le problème de Lagrange (7.9) en $\lambda = \lambda^j$ et à évaluer la valeur de la fonction duale régularisée δ_r (et son gradient) en λ^j : on résoud
$$(PL)^j \begin{cases} \min\limits_{(\delta m, y)} L_r(\delta m, y, \lambda^j) \\ l \leq y \leq u. \end{cases} \tag{7.10}$$
L'algorithme de résolution de (7.10) est détaillé au chapitre 8. Soit $(\delta m^{j+1}, y^{j+1})$ une solution de (7.10) (il peut y en avoir plusieurs).
- La deuxième étape consiste à calculer les nouveaux multiplicateurs de Lagrange λ^{j+1} par la formule suivante :
$$\begin{cases} \lambda_E^{j+1} = \lambda_E^j + r_j(C_E \delta m^{j+1} - e) \\ \lambda_I^{j+1} = \lambda_I^j + r_j(y^{j+1} - C_I \delta m^{j+1}). \end{cases}$$

Conditions nécessaires et suffisantes d'optimalité du Problème de Lagrange (7.10)

Soit $(\delta m^{j+1}, y^{j+1})$ un minimum de $(PL)^j$. Comme $L_r(\cdot, \cdot, \lambda^j)$ est suffisamment régulière et que les contraintes sur y sont des contraintes de borne (donc qualifiées (QC-A) voir définition 4.1.9) les conditions d'optimalité de Karush-Kuhn-Tucker s'écrivent

(cf. théorème 4.1.11) : il existe $\iota_l^{j+1} \in \mathbb{R}^{n_I}$ et $\iota_u^{j+1} \in \mathbb{R}^{n_I}$ tels que

$$\begin{cases} (a)\ \nabla F_k(\delta m^{j+1}) + C_E^\top \lambda_E^j + rC_E^\top (C_E \delta m^{j+1} - e) \\ \qquad -C_I^\top \lambda_I^j - rC_I^\top (y^{j+1} - C_I \delta m^{j+1}) = 0 \\ (b)\ \lambda_I^j = -\iota_l^{j+1} + \iota_u^{j+1} - r(y^{j+1} - C_I \delta m^{j+1}) \\ (c)\ l \le y^{j+1} \le u \\ (d)\ \iota_l^{j+1\,\top}(y^{j+1} - l) = 0, \quad \iota_u^{j+1\,\top}(y^{j+1} - u) = 0, \quad (\iota_l^{j+1}, \iota_u^{j+1}) \ge 0, \end{cases} \tag{7.11}$$

où ι_l^{j+1} (resp. ι_u^{j+1}) est le multiplicateur de Lagrange associé aux contraintes de borne inférieures (resp. supérieures).

Algorithme du lagrangien augmenté

Définition 7.2.4 *On définit la mesure du respect des contraintes en norme* $||.||_2$ *du* (PQT) *par*

$$\kappa(\delta m, y) = \sqrt{||C_E.\delta m - e||_2^2 + ||y - C_I.\delta m||_2^2}$$

et on utilise la notation : $\kappa^j = \kappa(\delta m^j, y^j)$.

Data : $\lambda^0 \in \mathbb{R}^{n_C}$, $r_0 > 0$,
$\delta m^0 = 0$,
$j = 0$.
begin
  Initialisation de y^0 ($l \le y^0 \le u$)
  while *(κ^j n'est pas suffisament proche de* 0*)* **do**
    ((1)) Résoudre le problème de Lagrange $(PL)^j$ de (7.10) :
      on obtient la solution $(\delta m^{j+1}, y^{j+1})$.
    ((2)) Calculer les nouveaux multiplicateurs de Lagrange λ^{j+1} :

$$\begin{cases} \lambda_E^{j+1} = \lambda_E^j + r_j(C_E \delta m^{j+1} - e) \\ \lambda_I^{j+1} = \lambda_I^j + r_j(y^{j+1} - C_I \delta m^{j+1}). \end{cases}$$

    ((3)) Choisir un nouveau paramètre d'augmentation $r_{j+1} > 0$.
    ((4)) $j := j + 1$.
  endw
end

Algorithme II.5 Algorithme du lagrangien augmenté

Quelques commentaires sur cet algorithme :

1. La convergence de l'algorithme du lagrangien augmenté est assurée lorsque (7.3) a une solution. Les conditions d'optimalité de ce problème s'écrivent (conditions équivalentes à (6.7)) : il existe $\widehat{\lambda}_E$, $\widehat{\lambda}_I$, $\widehat{\lambda}_l$, et $\widehat{\lambda}_u$ tels que

$$\begin{cases} (a)\ \nabla F_k(\widehat{\delta m}) + C_E^\top \widehat{\lambda}_E - C_I^\top \widehat{\lambda}_I = 0 \\ (b)\ \widehat{\lambda}_I = \widehat{\lambda}_l - \widehat{\lambda}_u \\ (c)\ C_E \widehat{\delta m} = e, \quad C_I \widehat{\delta m} = \widehat{y}, \quad l \leq \widehat{y} \leq u \\ (d)\ \widehat{\lambda}_l^\top (\widehat{y} - l) = 0, \quad \widehat{\lambda}_u^\top (\widehat{y} - u) = 0, \quad (\widehat{\lambda}_l, \widehat{\lambda}_u) \geq 0. \end{cases} \tag{7.12}$$

   En prenant $\widehat{\lambda}_E = \lambda_E^j + r(C_E \delta m^{j+1} - e)$ et $\widehat{\lambda}_I = \lambda_I^j + r(y^{j+1} - C_I \delta m^{j+1})$, on retrouve toutes les conditions d'optimalité de $(PL)^j$ (équation (7.11)) sauf les conditions d'admissibilité des contraintes. C'est en fait le rôle des itérations de lagrangien augmenté, d'adapter λ_E^j et λ_I^j de manière à ce que les contraintes $C_E \delta m = e$ et $y = C_I \delta m$ soient vérifiées (voir étape ((2)) de l'algorithme). Comme les conditions d'optimalité de $(PL)^j$ sont satisfaites à chaque résolution de $(PL)^j$ (voir étape ((1)) de l'algorithme), la convergence de l'algorithme du lagrangien augmenté est seulement contrôlée par la mesure de l'admissibilité des contraintes : si κ^j est plus petit qu'une certaine tolérance, alors on considère que l'optimalité du (PQT) est atteinte. En pratique, sur nos exemples synthétiques de tomographie, on parvient à vérifier une précision très faible de $\kappa^{j^*} \leq 10^{-10}$, où j^* est la dernière itération de lagrangien augmenté avant d'atteindre la convergence.

2. Même si $(\delta m^{j+1}, y^{j+1})$ n'est pas déterminé de manière unique comme solution de $(PL)^j$, $C_E \delta m^{j+1} - e$ et $y^{j+1} - C_I \delta m^{j+1}$ sont indépendants de cette solution si bien que les multiplicateurs λ^j sont déterminés de manière unique.

3. Cet algorithme est encore bien défini en optimisation linéaire (critère linéaire).

4. Sous la seule hypothèse que le problème (PQT) a une solution, alors le problème de Lagrange $(PL)^j$ à l'étape ((1)) de l'algorithme a aussi une solution (voir proposition 3.3 de l'annexe C). D'autre part, $(PL)^j$ étant un problème quadratique convexe, il a une solution dès qu'il est réalisable et borné (voir théorème 17.1 de [18]). Il y aura unicité si le critère de $(PL)^j$ est strictement convexe.

5. On se donne la possibilité de modifier le paramètre d'augmentation r, à l'étape ((3)) de l'algorithme. Cette modification, ne pose pas de problème de convergence. On demande seulement que les étapes ((1)) et ((2)) soient réalisées avec le même paramètre d'augmentation et que r_j soit suffisamment grand pour assurer la convergence (voir sous-section suivante). Il est plus intéressant de prendre un paramètre d'augmentation grand pour que la convergence du LA s'améliore (voir figure 1. de l'article en annexe C). Cependant, nous verrons dans le chapitre suivant que de grandes valeurs de r_j peuvent détériorer le conditionnement de $(PL)^j$.

6. La résolution du (PQT) par une méthode de lagrangien augmenté peut se voir comme la résolution de la suite de problèmes quadratiques sous contraintes de borne $\{(PL)^j\}$. L'étape ((1)) est la plus coûteuse de l'algorithme. Il faut donc pouvoir résoudre efficacement le problème de Lagrange $(PL)^j$, pour que la méthode du

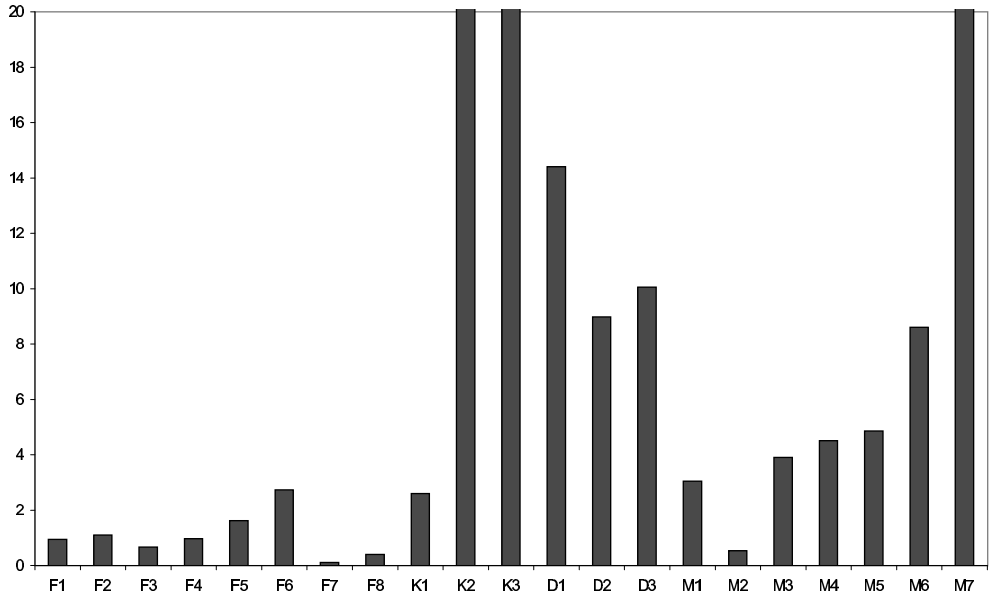

Fig. 7.1 Rapport du nombre d'itérations de GC effectuées par l'optimiseur avec contraintes sur celles effectuées par l'optimiseur sans contrainte.

lagrangien augmenté soit compétitive par rapport à d'autres algorithmes (dont par exemple une méthode de PI). Il nous a semblé intéressant de mesurer l'efficacité de la résolution du (PQT) (ou de la suite des $(PL)^j$) en prenant le rapport du nombre d'itérations de GC effectuées par notre optimiseur avec contraintes par rapport au nombre d'itérations nécessaires pour résoudre le problème sans contrainte. Dans la figure 7.1 nous remarquons que ce rapport reste bien inférieur à 20 pour la majorité des problèmes de notre bibliothèque de tomographie (voir annexe D). Seuls les problèmes dont les contraintes sont couplées (K2, K3 et M7) dépassent le rapport de 20 : dans ce cas on verra que le preconditionneur du GC est mal adapté (voir section 8.2.3).

7. Nous n'avons aucune idée de la solution a priori du (PQT). Ainsi, on initialise δm^0 à 0. Dans notre cadre, cette initialisation permet de tirer parti de la propriété d'identification finie des contraintes actives de SQP (voir section 8.5). Hors du cadre SQP (résolution par exemple d'un unique problème d'optimisation quadratique) on pourra avoir une idée de δm^0 et ainsi tirer parti de la propriété d'identification finie des contraintes actives du LA (voir propriété 7.2.6).

Propriétés de l'algorithme du lagrangien augmenté

Le LA s'interprète comme un algorithme proximal sur la fonction duale δ_0 (voir [134]). De cette interprétation découle directement la proposition suivante.

Proposition 7.2.5 *Si r_j est constant, l'algorithme II.5 du LA a les propriétés suivantes :*

1) *Les suites* $\{y^j - C_I\delta m^j\}$ *et* $\{C_E\delta m^j - e\}$ *sont décroissantes. Elles tendent vers zéro si* δ_0 *est bornée inférieurement, en particulier s'il existe un point admissible.*
2) *Si* δ_0 *a un minimiseur* $\widehat{\lambda}$*, la suite* $\{\lambda^j - \widehat{\lambda}\}$ *décroît et* $\{\lambda^j\}$ *converge vers un minimiseur de* δ_0*.*

Le théorème 4.4 de [45] (article figurant aussi dans l'annexe C) donne un résultat de convergence linéaire globale de l'algorithme II.5. Ce résultat appliqué à notre algorithme du LA s'écrit : il existe $L > 0$ (dépendant de λ^0) tel que si $r_j > \bar{r} > 0, \forall j$, alors

$$\kappa^{j+1} \leq \min(\frac{L}{r_j}, 1)\kappa^j,$$

Ce théorème montre que si le paramètre d'augmentation r_j est plus grand que la constante L (inconnue), alors la norme l_2 des contraintes tend globalement linéairement vers 0, c'est à dire : à chaque itération j de LA la norme l_2 des contraintes décroît d'un facteur uniformément plus petit que 1.

Nous donnons ci-dessous la propriété d'identification finie des contraintes actives du lagrangien augmenté :

Proposition 7.2.6 *Supposons que* H *est définie positive et que les conditions de complémentarités strictes sont satisfaites en la solution de* (PQT) *(voir définition 4.1.12), alors l'algorithme du lagrangien augmenté identifie les contraintes actives de* (PQT) *en un nombre fini d'itérations.*

DÉMONSTRATION. L'hypothèse de complémentarité stricte et la propriété de convergence de l'algorithme du LA ($\lambda^j \longrightarrow \widehat{\lambda}$) nous donne pour les contraintes d'inégalité actives en la solution ($i \in I$) :

$$(\widehat{\lambda})_i \neq 0 \Longrightarrow \exists \bar{j} \text{ tel que, } \forall j \geq \bar{j},\ (\lambda^j)_i(\widehat{\lambda})_i > 0\ .$$

Ainsi, aprés un nombre fini d'itérations de LA, $(y^j)_i$ est fixé à la bonne borne (on le déduit du fait que $\widehat{\lambda}_I = \widehat{\lambda}_l - \widehat{\lambda}_u$, voir équation (7.12)) : $(y^j)_i = (l_i|u_i)$.
D'autre part, comme H est définie positive, le critère de (PQT) est alors strictement convexe, et cela implique que

$$y^j \longrightarrow \widehat{y} = C\widehat{\delta m},$$

où la solution primale $\widehat{\delta m}$ est déterminée de manière unique. Donc, en utilisant la complémentarité stricte puis la convergence de y^j :

$$(\widehat{\lambda})_i = 0 \Longrightarrow l_i < (\bar{y})_i < u_i \Longrightarrow \exists \bar{j} \text{ tel que, } \forall j \geq \bar{j},\ l_i < (y^j)_i < u_i.$$

Ainsi, après un nombre fini d'itérations de LA, $(y^j)_i$ reste strictement à l'intérieur des bornes. □

Dans le premier point de la remarque 7.2.1 nous avons vu que l'un de nos arguments principaux pour le choix d'une méthode de LA est que celle-ci peut tirer parti d'une estimation des contraintes actives en la solution. La figure 7.2 illustre en pratique

la vérification de cette propriété : sur la majorité de nos problèmes de tomographie le nombre d'itérations de LA décroît avec les itérations de GN. Toutefois, on peut remarquer que cette propriété n'est pas vrai pour les exemples K2 et K3 où le nombre d'itération de LA fluctue entre 10 et 70. Ce mauvais comportement provient d'une mauvaise résolution des problèmes de Lagrange (le préconditionneur utilisé actuellement dans le solveur GP-AC-GC ne prend pas en compte les termes de couplage des contraintes et les exemples K2 et K3 contiennent des contraintes couplant des interfaces, voir 8.2.3). On peut aussi observer que l'on a pas de décroissance pour les exemples M2 et M3, cependant dans ces deux cas le nombre d'itérations de LA est faible (de l'ordre de 3, 4 itérations) quel que soit l'itération de GN.

Notons que, dans le cas où les contraintes du problème quadratique tangent sont incompatibles et dans le cas où l'on s'intéresse uniquement à des contraintes d'égalités alors le LA a une propriété de convergence intéressante. Nous la décrivons dans la remarque suivante (reprise de la remarque 2.13 de [74, page 65]).

Remarques 7.2.7 *Supposons que le problème $(PQT)_k$ contient uniquement des contraintes d'égalité. Supposons aussi que les contraintes de $(PQT)_k$ sont incompatibles. Dans ce cas, le problème $(PQT)_k$ n'a pas de solution. Cependant si l'on applique l'algorithme II.5 du LA pour rechercher une solution à ce problème mal posé, alors on a*

$$\lim_{j \longrightarrow +\infty} v^n = v^*,$$

où v^* est la solution du problème

$$\begin{cases} v^* \in H^* = \{w \in \mathbb{R}^n : C_E^\top (C_E w - e) = 0\}, \\ w^\top H_k v^* = -g_k^\top w, \quad \forall w \in \ker(C_E^\top C_E). \end{cases}$$

De plus, H^* ($\neq \emptyset$) est l'ensemble des solutions de l'équation

$$C_E^\top C_E z = C_E^\top e.$$

Notons que la convergence de v^j vers v^* est linéaire. Par contre, la suite $\{\lambda_E^j\}_{j \geq 0}$ des multiplicateurs de Lagrange diverge avec une progression arithmétique. Cette divergence est "beaucoup plus lente" que la convergence de $\{v^j\}_{j \geq 0}$.

7.2.2 Prise en compte explicite de contraintes d'égalité dans QPAL

Dans cette section nous décrivons une méthode permettant de traiter des contraintes d'égalité dures dans le solveur QPAL. Notons que cette méthode n'a pas été testée.

Nous entendons, par contraintes dures des contraintes d'égalité, donc de la forme

$$C_D \delta m = d, \tag{7.13}$$

que l'on souhaite réaliser exactement tout au long des itérations de LA. Dans (7.13) C_D est une matrice rectangulaire de type $n_D \times n$ et $d \in \mathbb{R}^{n_D}$. On suppose dans la suite que la matrice C_D est surjective. Les solutions de (7.13) s'écrivent alors comme :

$$\delta m = \delta m_p + Z w \tag{7.14}$$

où δm_p est une solution particulière de (7.13), Z est une matrice de type $n \times (n - n_D)$ dont les colonnes forment une base du noyau de C_D ($C_D Z = 0$ et Z injective) et $w \in \mathbb{R}^{n-n_D}$. Notons qu'il existe un nombre très important de méthodes de factorisation (souvent basé sur la structure de C_D) permettant de calculer δm_p et Z. On pourra par exemple utiliser classiquement une factorisation QR de C_D, une méthode d'élimination par pivot de Gauss ou un compromis entre ces 2 méthodes (voir [123, page 434]).

Voici ci-dessous quelques arguments qui peuvent justifier l'utilisation de contraintes dures :

1. Le LA obtient des points admissibles en minimisant $||C_D \delta m - d||_2^2$. Ce n'est pas nécessairement la méthode la plus efficace lorsqu'on doit résoudre des problèmes d'optimisation comportant un grand nombre de variables : il vaut peut-être mieux utiliser une méthode qui réalise les contraintes d'égalité tout le temps au cours des itérations de LA.
2. La prise en compte directe des contraintes d'égalité permet d'éviter de dégrader le conditionnement de $H_{\mathcal{W}}(r)$ (matrice intervenant dans la résolution du problème de lagrange, voir (8.12)) lorsque la matrice $C_D^\top C_D$ est très mal conditionnée.
3. Certaines contraintes d'égalité possèdent une structure qui permet de calculer rapidement une solution particulière δm_p et une matrice Z et ainsi d'exprimer les solutions de (7.13) par (7.14). En effet, supposons par exemple que la matrice C_D s'exprime à une permutation près par $C_D = (B|N)$, où B est une matrice inversible de type $n_D \times n_D$ et N est une matrice rectangulaire de type $n_D \times (n - n_D)$ alors on peut prendre

   $$\delta m_p = \begin{bmatrix} B^{-1} \\ 0_{n-n_D} \end{bmatrix} d$$

   comme solution particulière et

   $$Z = \begin{bmatrix} -B^{-1}N \\ I_{n-n_D} \end{bmatrix}$$

   comme matrice dont les colonnes forment une base du noyau de C_D (voir [123, page 430]).

Lorsqu'on rajoute les contraintes d'égalité dures (7.13) dans le problème quadratique tangent (7.1), celui-ci se reformule comme

$$\begin{cases} \min\limits_{\delta m \in \mathbb{R}^n} F(\delta m) \\ C_E \delta m = e \\ C_D \delta m = d \\ l \leq C_I \delta m \leq u. \end{cases} \tag{7.15}$$

En injectant l'équation (7.14) dans (7.15), on peut montrer que (7.15) est équivalent à la

résolution d'un problème quadratique avec contraintes du type :

$$\begin{cases} \min\limits_{w\in\mathbb{R}^{n-n_D}} \tilde{F}(w) \\ (C_E Z)w = \tilde{e} \\ \tilde{l} \leq (C_I Z)w \leq \tilde{w}. \end{cases} \tag{7.16}$$

Ce nouveau problème peut alors être résolu par la méthode du LA (on pourra appliquer l'algorithme II.5 du LA). Nous donnons ci-dessous quelques remarques sur cette résolution.

1. La prise en compte de contraintes dures dans le solveur QPAL induit deux produits matrice-vecteur supplémentaire à réaliser à chaque itération de GC (algorithme utilisé pour résoudre le problème de Lagrange, voir chapitre suivant) : le produit de Z et le produit de $Z^\top$ par un vecteur.
2. Notons que le nouveau problème quadratique à résoudre est de dimension $(n - n_D)$ alors que l'ancien est de dimension n. Cette réduction du nombre d'inconnues est un avantage pour la convergence l'algorithme du GC (algorithme utilisé lors de la résolution du problème de Lagrange, voir chapitre suivant).
3. Rappelons que, dans notre application en tomographie de réflexion, on utilise le solveur QPAL dans une approche SQP et que les contraintes du problème non-linéaire sont linéaires. Ainsi, une fois qu'une matrice Z est obtenue pour les contraintes d'égalité dures C_D, cette matrice peut-être utilisée pour toutes les itérations de SQP.

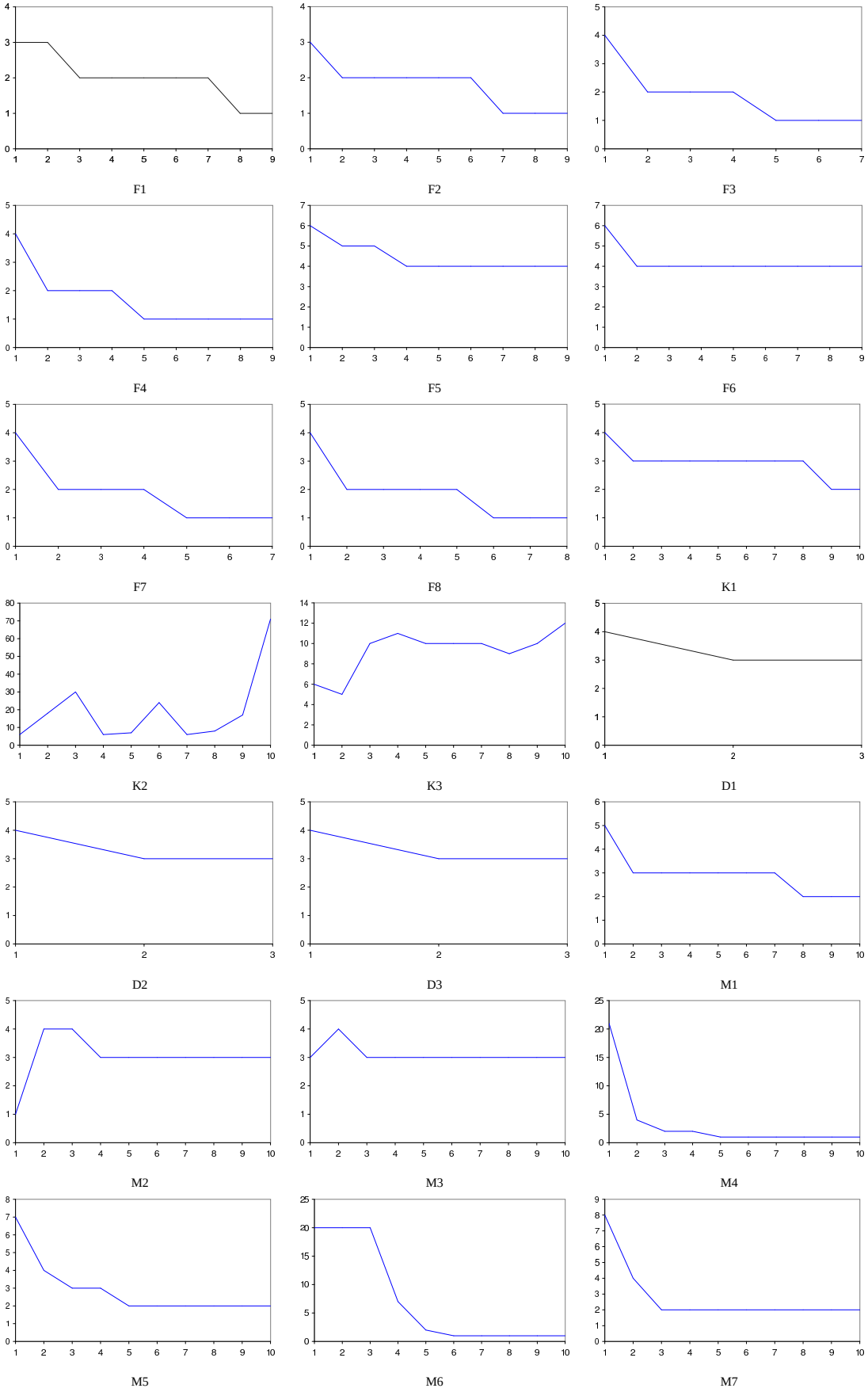

Fig. 7.2 Nombre d'itérations de LA au cours des itérations de GN

Chapitre 8

Résolution du problème de Lagrange

Dans ce chapitre nous nous intéressons à la résolution du problème de Lagrange (7.10). La première chose à remarquer sur le problème $(PL)^j$ est que la fonction $L_r(\cdot,\cdot,\lambda^j)$ à minimiser est quadratique convexe. Soit la fonction $\Upsilon^j(X) : \mathbb{R}^n \times \mathbb{R}^{n_I} \mapsto \mathbb{R}$ définie par

$$\Upsilon^j(X) = L_r(\delta m, y, \lambda^j) - L_r(0,0,\lambda^j), \tag{8.1}$$

où $X = (\delta m, y)$. En effet, Υ^j se reformule comme la fonction quadratique convexe :

$$\Upsilon^j(X) = \frac{1}{2} X^\top Q X + X^\top b^j, \tag{8.2}$$

où

$$b^j = \nabla_X \Upsilon^j(0) = \begin{bmatrix} g + C_E^\top \lambda_E^j - r C_E^\top e - C_I^\top \lambda_I^j \\ \lambda_I^j \end{bmatrix}$$

est le gradient de Υ^j en $X = 0$ et

$$Q = \nabla^2_{XX} \Upsilon^j = \begin{bmatrix} H + r C_E^\top C_E + r C_I^\top C_I & -r C_I^\top \\ -r C_I & r I_{n_I} \end{bmatrix}$$

est le hessien de Υ^j (on montre facilement que Q est semi-définie positive car Υ^j est obtenu en ajoutant un terme quadratique convexe à la fonction convexe F). A partir de l'expression quadratique (8.2) de la fonction Υ^j, on peut reformuler le $(PL)^j$ comme :

$$(PL)^j \begin{cases} \min\limits_{X=(\delta m, y)} & \Upsilon^j(X) \\ l \leq y \leq u. \end{cases} \tag{8.3}$$

Ainsi le problème $(PL)^j$ fait partie de la classe des problèmes d'optimisation quadratique convexe avec contraintes de borne.

8.1 Minimisation d'une fonction quadratique sous contraintes de borne : état de l'art

La différence avec le chapitre précédent, où nous avons traité la résolution du problème quadratique convexe (PQT), est que le problème $(PL)^j$ ne possède que des contraintes de

borne. A la section 7.1, nous avons déjà discuté des différentes classes de méthode d'optimisation pour résoudre un problème quadratique convexe. Parmi ces méthodes, nous avons choisi une méthode AC pour les trois raisons suivantes :

1. Si on décide d'utiliser une méthode de PI sur le $(PL)^j$, autant l'utiliser plutôt sur le (PQT) original : la méthode des PI ne tire aucun avantage du fait que les contraintes d'inégalité soient uniquement de bornes.
2. La méthode AC permet de tirer parti de la propriété d'activation finie des contraintes actives par le LA (voir proposition 7.2.6).
3. Pour les problèmes avec contraintes de borne, la méthode AC est reconnue pour être très efficace (voir [118], [63] et [123]).

Dans la section 7.1.1 nous avons expliqué le fonctionnement de la méthode générale AC. Nous y avons vu que cette méthode n'était pas des plus efficace pour résoudre les problèmes avec un nombre important de contraintes. Cependant si cette méthode peut être associée à l'algorithme du gradient avec projection, alors elle peut devenir très intéressante. Cette méthode, dénommée içi GP-AC, produit des changements rapides de l'ensemble de travail : au moment où l'on décide de désactiver des contraintes, un déplacement le long du chemin obtenu en projetant la direction opposée au gradient sur l'ensemble admissible est effectué. Dans ce cas, la projection est facile à mettre en oeuvre car on traite uniquement des contraintes de borne. Après cette opération, on définit $\mathcal{W}$ comme l'ensemble des contraintes actives au nouveau point. Cette idée d'interchanger des étapes d'activation de contraintes par des étapes de GP remonte au moins à [117], bien que les algorithmes utilisant la projection sur l'ensemble admissible soient plus anciens (voir [77], [110], [12] et les références dans ces articles). De part l'efficacité reconnue de la méthode GP-AC sur les problèmes avec contraintes de borne et du fait qu'elle peut tirer profit de la connaissance des contraintes actives (le LA ayant la propriété d'identification finie des contraintes actives) nous avons choisi cette méthode pour résoudre le problème $(PL)^j$.

8.2 Algorithme GP-AC-GC classique

Définition 8.2.1 *On note $\mathcal{A}(y)$ l'ensemble des contraintes de borne actives en y :*

$$\mathcal{A}(y) = \{i \in I : y_i = l_i \text{ ou } y_i = u_i\}.$$

La méthode GP-AC utilise 3 ingrédients principaux. Le premier, correspond aux techniques classiques d'activation de contraintes. Le second ingrédient est une technique de désactivation multiple de contraintes par l'utilisation d'une méthode de gradient avec projection (GP). Enfin le troisième ingrédient est une technique de minimisation de $(PL)^j$

sur l'ensemble de travail $\mathcal{W}$ courant :

$$(PL)^j_{\mathcal{W}} \begin{cases} \min\limits_{X} \ \Upsilon^j(X) \\ y_i = l_i \text{ ou } y_i = u_i \ \ i \in \mathcal{W} \\ (l_i \leq y_i \leq u_i \ \ i \notin \mathcal{W}), \end{cases} \tag{8.4}$$

où les inégalités (dernière ligne de (8.4)) ne sont pas prises en compte directement dans la résolution de $(PL)^j_{\mathcal{W}}$, mais servent à déclencher le passage à la résolution d'un nouveau problème. Nous avons choisi l'algorithme du Gradient Conjugué (GC) pour résoudre le problème $(PL)^j_{\mathcal{W}}$. Ce choix est motivé par l'efficacité déjà démontrée du GC lors de la résolution de nos problèmes de tomographie de réflexion sans contrainte[1] (cf. section 2.5, chapitre 3 et [30]). Suite à ce choix nous dénommerons donc par GP-AC-GC la méthode générale que nous avons choisie pour résoudre le Problème de Lagrange $(PL)^j$.

Voici ci-dessous l'algorithme GP-AC-GC classique appliqué à la minimisation du problème $(PL)^j$:

[1] Dans le cas de problème de petite et moyenne dimension on pourrait envisager des méthodes directes (par ex : factorisation de Choleski)

Data : Point de départ $X = (\delta m^j, y^j)$ avec $l \leq y^j \leq u$,
$GP =$ **true**,
$l = 0$.
begin
  $\mathcal{W} = \mathcal{A}(y)$.
  while *((7.11) n'est pas satisfaite)* **do**
    $\mathcal{W}_{old} = \mathcal{W}$.
    if *(GP)* **then**
      Étape GP : faire un pas de gradient avec projection.
      Mise à jour de l'ensemble de travail $\mathcal{W} = \mathcal{A}(y)$.
      if *(pas de contraintes activées ou désactivées : $\mathcal{W}_{old} = \mathcal{W}$)* **then**
        $GP =$ **false**
      endif
    else
      Étape GC : résoudre $(PL)^j_{\mathcal{W}}$ par GC jusqu'à ce que l'on rencontre une nouvelle borne sur y.
      Mise à jour de l'ensemble de travail $\mathcal{W} = \mathcal{A}(y)$.
      if *(pas de contraintes activées : $\mathcal{W}_{old} = \mathcal{W}$)* **then**
        $GP =$ **true**.
        $\mathcal{W}_l = \mathcal{W}$, $X_l = X$ (notons que $\mathcal{W}_l$ et X_l ne sont pas utiles pour le bon fonctionnement de cet algorithme, ils servent pour la démonstration du théorème 8.2.5).
        $l := l + 1$.
      endif
    endif
  endw
  $(\delta m^{j+1}, y^{j+1}) = X$.
end

Algorithme II.6 Algorithme GP-AC-GC classique : résolution de $(PL)^j$

Quelques commentaires sur cet algorithme :

1. L'algorithme GP-AC-GC se décompose en une succession d'étapes GP et GC (plusieurs étapes de GP ou de GC peuvent se succéder). La méthode AC est présente entre chacune de ces étapes par la mise à jour de l'ensemble de travail $\mathcal{W}$. L'initialisation de l'algorithme s'effectue par $X = (\delta m^j, y^j)$ permet de prendre comme point de départ la solution du problème de Lagrange précédent (c'est à dire la solution de $(PL)^{j-1}$ pour $j \geq 1$). Notons que l'initialisation de y par y^j sert d'une part à tester l'optimalité du point de départ avant de rentrer dans la boucle de l'algorithme et d'autre part à initialiser le premier ensemble de travail $\mathcal{W}$ (dans le cas de l'utilisation de l'algorithme GP avec une projection en y seuleument, voir section 8.3.1, la valeur initiale de y est inutile pour la suite de l'algorithme).

2. **Définition 8.2.2** *On appelle phase de GP (resp. phase de GC) une suite d'étape consécutive de GP (resp. de GC) avec stabilisation.*

   Lorsque l'on se trouve dans une étape (GP ou GC) on ne change de phase (passage d'une étape GP à une étape GC ou inversement) que si l'ensemble de travail $\mathcal{W}$ reste inchangé par rapport à l'étape précédente. Après une étape GC, l'absence de modification de $\mathcal{W}$ signifie que le critère a été complètement minimisé le critère sur la face activée courante. Une itération l de GP-AC-GC se réalise lorsqu'on est passé par une de phase de GP et par une de phase GC. L'ensemble de travail W_l est alors l'ensemble sur lequel le critère a été complètement minimisé à l'itération l de GP-AC-GC. On note alors X_l la solution de $(PL)^j_{\mathcal{W}_l}$.

3. Dans l'étape GP, on recherche le premier minimum local du critère quadratique (8.2) en suivant la ligne brisée définie par la projection de $-\nabla_X \Upsilon^j(X)$ sur les bornes de y (d'autres auteurs, voir par exemple [117], calculent un point d'Armijo). La difficulté principale de cette étape réside dans le calcul de la projection $-\nabla_X \Upsilon^j(X)$ sur les bornes de y. Cependant, dans notre cas la projection est facile à réaliser car les contraintes sont de borne. Ainsi, lorsqu'une borne de y est rencontrée, la direction de recherche est "amputée" pour rester admissible (la composante de $\nabla_X \Upsilon^j(X)$ correspondant à la borne rencontrée est mise à 0). Cette étape est détaillée dans la section 8.2.1 suivante.

4. Dans l'étape GC on recherche le minimum du critère quadratique (8.2) restreint à la face des contraintes de borne définie par l'ensemble de travail courant $\mathcal{W}$. Pour ce faire deux approches différentes sont envisageables. La première approche consiste à résoudre exactement le $(PL)^j_{\mathcal{W}}$ par GC, puis à projeter la perturbation obtenue sur les bornes pour modifier l'ensemble de travail $\mathcal{W}$. La deuxième approche consiste à faire du GC tronqué : on arrête le GC et on ajoute un indice dans $\mathcal{W}$ dès que l'on rencontre une borne. Nous avons choisi la seconde approche pour résoudre $(PL)^j_{\mathcal{W}}$. Dans la section 8.2.2 nous détaillerons ces deux approches et motiverons notre préférence pour la seconde.

5. Dans la section 8.2.5, nous donnerons des conditions pour que l'algorithme GP-AC-GC converge.

8.2.1 Étape de Gradient avec Projection

Les premiers travaux sur la méthode du gradient avec projection remontent au moins à [77] et [110]. Grâce à sa propriété d'identification finie des contraintes actives lorsqu'il y a complémentarité stricte, la méthode du gradient avec projection évite les phénomènes d'oscillations (ou zigzaging, i.e., activation et désactivation inintérrompue d'une même borne). Malgré cet avantage, la méthode du gradient avec projection en tant que telle est lente à converger sur des problèmes de grande taille : c'est une méthode du premier ordre, elle n'exploite que les informations provenant des dérivées premières du critère quadratique (plus précisément le gradient). Afin de résoudre le problème $(PL)^j$, nous utilisons la méthode du gradient avec projection à l'intérieur de la méthode d'activation de contrainte :

l'étape GP sert essentiellement à activer ou désactiver rapidement des contraintes et à trouver l'ensemble de travail $\mathcal{W}$ courant sur lequel on effectuera la minimisation du critère quadratique de $(PL)^j$ par GC (résolution du problème $(PL)^j_{\mathcal{W}}$).

Dans l'étape GP de l'algorithme II.6, on dispose d'un point de départ admissible X (y vérifiant les contraintes de borne de $(PL)^j$). L'objectif est de trouver le premier minimum local strict du critère quadratique de $(PL)^j$ le long de la ligne brisée obtenue par projection sur les contraintes de borne de $(PL)^j$ d'un chemin affine issu de X et pointant vers la direction opposée au gradient du critère quadratique de $(PL)^j$. On se déplace donc sur la ligne brisée

$$(\alpha > 0) \;\mapsto\; \begin{pmatrix} \delta m - \alpha g_{\delta m} \\ P_{[l,u]}(y - \alpha g_y) \end{pmatrix}, \tag{8.5}$$

avec $P_{[l,u]}$ l'opérateur de projection sur les contraintes de borne et $g_{\delta m}$ (resp. g_y) le gradient du critère quadratique Υ^j par rapport à δm (resp. y) définie comme

$$g_X = \begin{pmatrix} g_{\delta m} \\ g_y \end{pmatrix} = \nabla \Upsilon^j(X) = QX + b^j. \tag{8.6}$$

8.2.2 Étape de Gradient Conjugué

On dispose, comme dans l'étape GP, d'un point de départ X admissible (y respecte les bornes) à partir duquel on pourra démarrer les itérations de GC. Dans cette étape les variables $y_i : i \in \mathcal{W}$ sont fixées à la borne inférieure l_i ou à la borne supérieure u_i. On note $\mathcal{J} = \mathcal{W}^c = \{i \in I : i \notin \mathcal{W}\}$ l'ensemble d'indices complémentaires à $\mathcal{W}$ dans I. La variable $y_{\mathcal{J}}$ est donc formée des variables y_i non fixées à une borne, et on note aussi $C_{\mathcal{W}}$ la matrice obtenue à partir de C_I en ne retenant que les lignes d'indice dans $\mathcal{W}$ (de même pour $C_{\mathcal{J}}$). L'étape de GC de l'algorithme II.6 consiste, dans un premier temps, à résoudre le problème (8.4) sans tenir compte des contraintes de borne sur $y_{\mathcal{J}}$ (troisième ligne de (8.4)). Plus précisement, il s'agit de minimiser le problème quadratique sans contrainte suivant :

$$\min_{X_{\mathcal{J}}} \left(\frac{1}{2} X_{\mathcal{J}}^\top Q_{\mathcal{J}} X_{\mathcal{J}} + X_{\mathcal{J}}^\top b^j_{\mathcal{J}} \right), \tag{8.7}$$

avec $X_{\mathcal{J}} = (\delta m, y_{\mathcal{J}})$,

$$b^j_{\mathcal{J}} = \begin{bmatrix} g + C_E^\top \lambda_E^j - r C_E^\top e - C_I^\top \lambda_I^j - r C_{\mathcal{W}}^\top y_{\mathcal{W}} \\ \lambda^j_{\mathcal{J}} \end{bmatrix}, \tag{8.8}$$

et

$$Q_{\mathcal{J}} = \begin{bmatrix} H + r C_E^\top C_E + r C_I^\top C_I & -r C_{\mathcal{J}}^\top \\ -r C_{\mathcal{J}} & r I_{n_{\mathcal{J}}} \end{bmatrix}. \tag{8.9}$$

L'équation d'optimalité du problème (8.7) se formule comme :

$$Q_{\mathcal{J}} X_{\mathcal{J}} = -b^j_{\mathcal{J}}. \tag{8.10}$$

Nous pourrions donc appliquer l'algorithme du GC pour résoudre le système linéaire (8.10). Cependant nous allons voir ci-dessous que nous pouvons simplifier la résolution de ce système.

La matrice de ce système est définie positive si et seulement si le complément de Schur de $rI_{n_{\mathcal{J}}}$ l'est, c'est-à-dire si et seulement si

$$(H + rC_E^\top C_E + rC_I^\top C_I) - (rC_{\mathcal{J}}^\top C_{\mathcal{J}}) = H + rC_E^\top C_E + rC_{\mathcal{W}}^\top C_{\mathcal{W}} \tag{8.11}$$

est définie positive. L'élimination de $y_{\mathcal{J}}$ montre que δm est solution de

$$H_{\mathcal{W}}(r)\delta m = -g_{\mathcal{W}}^j, \tag{8.12}$$

où

$$H_{\mathcal{W}}(r) = H + r(C_E^\top C_E + C_{\mathcal{W}}^\top C_{\mathcal{W}}), \tag{8.13}$$

et

$$g_{\mathcal{W}}^j = g + C_E^\top \lambda_E^j - rC_E^\top e - C_{\mathcal{W}}^\top (ry_{\mathcal{W}} + \lambda_{\mathcal{W}}^j).$$

Ainsi, au lieu de faire du gradient conjugué en $X_{\mathcal{J}}$, on peut faire du GC en δm sur le système (8.12) pour résoudre l'équation d'optimalité de (8.7). Cette technique est préférable à la résolution directe du système (8.10) par GC car (8.12) est de dimension plus petite (le GC devant converger en un nombre d'itérations inférieur ou égal à la dimension du problème). Voici ci-dessous une remarque importante faisant le parallèle avec les systèmes linéaires issus d'une méthode de points intérieurs.

Remarques 8.2.3 *Nous avons vu dans la section 4.2.4 que la méthode des PI transforme la résolution d'un problème non-linéaire avec contraintes en la résolution d'une suite de systèmes linéaires (voir équation 4.17). Au bout du compte, notre méthode, qui regroupe dans l'ordre les méthodes SQP, LA et GP-AC-GC, peut aussi se voir comme la transformation du problème non-linéaire (P_{EI}) en une suite de systèmes lineaires du type (8.12). Ce rapprochement est d'autant plus frappant que la matrice $H_{\mathcal{W}}(r)$ est très semblable à la matrice réduite Q_ν intervenant dans les systèmes linéaires issus de la méthode des points intérieurs (voir point numéro 1. de la remarque 4.2.6). Une première différence entre ces 2 méthodes est que dans la méthode des PI, le conditionnement de la matrice Q_ν se dégrade lorsque ν devient très proche de zéro alors que dans notre méthode le conditionnement de $H_{\mathcal{W}}(r)$ est maitrisé (il est inutile de faire tendre r vers $+\infty$ pour obtenir la convergence du LA). Une seconde différence provient du fait que, contrairement à notre méthode, la méthode des points intérieurs n'a pas à gérer de problème de combinatoire. En effet, dans notre cas, nous utilisons une méthode d'AC pour gérer la combinatoire issue des contraintes de borne du problème de Lagrange.*

On va donc résoudre (8.4) par l'utilisation de l'algorithme du GC sur (8.12) et par la mise à jour des variables y. Deux approches différentes sont possibles : on peut mettre à jour les variables y soit en fin d'itération (première possibilité) soit au cours des itérations de GC (deuxième possibilité).

Première possibilité (non testée). On résout complètement (8.12) par GC. On note $\widehat{\delta m}$ sa solution. La solution en y est alors $\widehat{y}_{\mathcal{W}} = y_{\mathcal{W}}$ et $\widehat{y}_{\mathcal{J}} = C_{\mathcal{J}}\widehat{\delta m} - \frac{1}{r}\lambda^j_{\mathcal{J}}$. On détermine ensuite le plus grand pas $\alpha \in [0,1]$ tel que $y + \alpha(\widehat{y} - y) \in [l,u]$. On prend finalement comme nouvel itéré :

$$\delta m_+ := \delta m + \alpha(\widehat{\delta m} - \delta m) \quad \text{et} \quad y_+ := y + \alpha(\widehat{y} - y)$$

Seconde possibilité (possibilité retenue). L'approche précédente a l'inconvénient de demander la résolution complète du système linéaire (8.12) avant de pouvoir se rendre compte que $y_{\mathcal{J}}$ sort des bornes[2]. On propose ici une approche dans laquelle on résout partiellement par GC (tronqué) le système (8.12), tout en faisant évoluer $y_{\mathcal{J}}$ de manière à minimiser le critère quadratique de $(PL)^j_{\mathcal{W}}$. Ceci permet d'arrêter les itérations dès que $y_{\mathcal{J}}$ sort des bornes. Il faut aussi s'assurer que, si l'on résout ainsi complètement (8.12), on minimise également complètement le critère de $(PL)^j_{\mathcal{W}}$, le but final. Voici ci-dessous quelques observations sur cette seconde possibilité :

1. La seconde équation de (8.10) s'écrit

$$y_{\mathcal{J}} = C_{\mathcal{J}}\delta m - \frac{1}{r}\lambda^j_{\mathcal{J}}, \tag{8.14}$$

   si bien que si $d\delta m$ est un déplacement en δm, il est naturel de prendre pour déplacement en $y_{\mathcal{J}}$ correspondant :

$$dy_{\mathcal{J}} = C_{\mathcal{J}}d\delta m.$$

   Ceci revient à faire des déplacements dans le noyau de la jacobienne $\begin{pmatrix} -C_{\mathcal{J}} & I \end{pmatrix}$ de (8.14).

2. Un calcul simple utilisant (8.11) montre que

$$\begin{aligned}
&\begin{pmatrix} d\delta m \\ C_{\mathcal{J}}d\delta m \end{pmatrix}^\top \begin{pmatrix} H + rC_E^\top C_E + rC_I^\top C_I & -rC_{\mathcal{J}}^\top \\ -rC_{\mathcal{J}} & rI_{n_{\mathcal{J}}} \end{pmatrix} \begin{pmatrix} d\delta m' \\ C_{\mathcal{J}}d\delta m' \end{pmatrix} \\
&= d\delta m^\top (H + rC_E^\top C_E + rC_I^\top C_I) d\delta m' - rd\delta m^\top C_{\mathcal{J}}^\top C_{\mathcal{J}} d\delta m' \\
&\quad -rd\delta m^\top C_{\mathcal{J}}^\top C_{\mathcal{J}} d\delta m' + rd\delta m^\top C_{\mathcal{J}}^\top C_{\mathcal{J}} d\delta m' \qquad (8.15) \\
&= d\delta m^\top (H + rC_E^\top C_E + rC_I^\top C_I) d\delta m' - rd\delta m^\top C_{\mathcal{J}}^\top C_{\mathcal{J}} d\delta m' \\
&= d\delta m^\top H_{\mathcal{W}}(r) d\delta m'. \qquad (8.16)
\end{aligned}$$

   Dès lors si $d\delta m$ et $d\delta m'$ sont conjuguées par rapport à la matrice $H_{\mathcal{W}}(r)$ du système (8.12), il en est de même de $(d\delta m, C_{\mathcal{J}}d\delta m)$ et $(d\delta m', C_{\mathcal{J}}d\delta m')$ par rapport à la matrice du système (8.10). On pourra donc à partir des directions conjuguées en δm, générées par le GC dans la résolution de (8.12), obtenir des directions conjuguées en $(\delta m, y)$ pour résoudre (8.10) et donc minimiser le critère de $(PL)^j_{\mathcal{W}}$.

[2] L'approche précédente est plutôt utilisée dans le cadre de la résolution du système linéaire (8.12) par une méthode directe du type factorisation de Cholesky

3. Le pas optimal $\alpha_{\delta m}$ le long d'une direction $d\delta m$ pour résoudre (8.12), celui donnant le minimum de la fonction quadratique associée à (8.12), est identique au pas optimal α_z le long de la direction $dz := (d\delta m, C_{\mathcal{J}} d\delta m)$ pour minimiser (8.10). Rappelons que le pas optimal minimisant la fonction quadratique $q^\top x + \frac{1}{2} x^\top Q x$ en x le long de la direction d s'écrit $-(Qx+q)^\top d/(d^\top Q d)$. On observe alors, grâce à (8.16), que les dénominateurs des formules donnant $\alpha_{\delta m}$ et α_z sont égaux. Quant aux numérateurs, le premier s'écrit

$$-\left(H_{\mathcal{W}}(r)\delta m + g^j_{\mathcal{W}}\right)^\top d\delta m$$

et, grâce à (8.8),(8.9) et (8.14), le second s'écrit

$$\begin{aligned}
&-\Bigg(\begin{pmatrix} H + rC_E^\top C_E + rC_I^\top C_I & -rC_{\mathcal{J}}^\top \\ -rC_{\mathcal{J}} & rI_{n_{\mathcal{J}}} \end{pmatrix} \begin{pmatrix} \delta m \\ y_{\mathcal{J}} \end{pmatrix} \\
&\qquad + \begin{pmatrix} g + C_E^\top \lambda_E^j - rC_E^\top e - rC_{\mathcal{W}}^\top y_{\mathcal{W}} - C_I^\top \lambda_I^j \\ \lambda^j_{\mathcal{J}} \end{pmatrix}\Bigg)^\top \begin{pmatrix} d\delta m \\ C_{\mathcal{J}} d\delta m \end{pmatrix} \\
&= -\begin{pmatrix} (H + rC_E^\top C_E + rC_I^\top C_I)\delta m + g + C_E^\top \lambda_E^j - rC_E^\top e - C_I^\top (ry + \lambda_I^j) \\ -rC_{\mathcal{J}}\delta m + ry_{\mathcal{J}} + \lambda^j_{\mathcal{J}} \end{pmatrix}^\top dz \\
&= -\begin{pmatrix} (H + rC_E^\top C_E + rC_{\mathcal{W}}^\top C_{\mathcal{W}})\delta m + g + C_E^\top \lambda_E^j - rC_E^\top e - C_{\mathcal{W}}^\top (ry_{\mathcal{W}} + \lambda^j_{\mathcal{W}}) \\ 0 \end{pmatrix}^\top dz \\
&= -\left(H_{\mathcal{W}}(r)\delta m + g^j_{\mathcal{W}}\right)^\top d\delta m.
\end{aligned}$$

Ils sont donc aussi égaux.

4. Si $d\delta m$ est une direction de descente en δm pour le critère quadratique associé à (8.12), le numérateur de $\alpha_{\delta m}$ est strictement positif. Par le calcul précédent, celui de α_z l'est aussi et donc la direction $(d\delta m, C_{\mathcal{J}} d\delta m)$ est une direction de descente en $(\delta m, y_{\mathcal{J}})$ pour le critère de $(PL)^j_{\mathcal{W}}$.

5. Si on mène les itérations de GC jusqu'à leur terme, on trouve la solution $\widehat{\delta m}$ de (8.12). Le $y_{\mathcal{J}}$ calculé vaut alors $\widehat{y}_{\mathcal{J}} = y_{\mathcal{J}} + C_{\mathcal{J}}(\widehat{\delta m} - \delta m)$, où $(\delta m, y_{\mathcal{J}})$ est le couple de départ. En fait, on vérifie facilement que, puisque $\widehat{\delta m}$ est solution de (8.12), $(\widehat{\delta m}, \widehat{y}_{\mathcal{J}})$ sera solution de (8.10) si et seulement si la seconde équation de (8.10) est vérifiée, c'est-à-dire si et seulement si

$$\widehat{y}_{\mathcal{J}} = C_{\mathcal{J}} \widehat{\delta m} - \frac{1}{r} \lambda^j_{\mathcal{J}}.$$

Il est donc nécessaire que le couple de départ vérifie (8.14) ou que l'on apporte à $\widehat{y}_{\mathcal{J}}$ en fin d'itérations la correction suivante

$$-(y_{\mathcal{J}} - C_{\mathcal{J}}\delta m) - \frac{1}{r}\lambda^j_{\mathcal{J}}.$$

On observe que la condition (8.14) est la stationnarité du critère de $(PL)^j_{\mathcal{W}}$ par rapport à $y_{\mathcal{J}}$. D'autre part, le hessien en $y_{\mathcal{J}}$ est un multiple de l'identité, si bien que cette stationnarité est facile à obtenir par des itérations de GP.

Ce point de vue plaide en faveur de l'option qui impose de satisfaire (8.14) avant de commencer les itérations de GC tronqué et en faveur d'une phase de GP ne portant que sur y en maintenant δm fixé (voir section 8.3.1).

6. On observe aussi que, si plusieurs étapes de GC se suivent, activant chaque fois les bornes sur lesquelles le GC vient buter, l'ensemble inactif $\mathcal{J}$ diminue à chaque fois, si bien que si (8.14) est vérifiée lors de la première étape d'une phase de GC, elle le sera aussi dans les étapes de GC qui la suivent.

8.2.3 Préconditionnement et critères d'arrêt du GC

Il est important d'avoir un bon préconditionneur du GC pour que la méthode GP-AC-GC soit efficace. Dans la partie I au chapitre 3, nous avons déjà vu les techniques de Jacobi et Gauss-Seidel pour préconditionner la matrice H. Cette fois-ci, pour la résolution du problème $(PL)^j_{\mathcal{W}}$, il faut trouver un bon préconditionneur de la matrice $H_{\mathcal{W}}(r) = H + r(C_E^\top C_E + C_{\mathcal{W}}^\top C_{\mathcal{W}})$ au début de chaque étape de GC. Une première solution consiste à utiliser les techniques de préconditionnement déjà mises en œuvre dans la partie sans contrainte : on calcule les facteurs de Cholesky d'une matrice $P_{\mathcal{W}}$ extraite de $H_{\mathcal{W}}$ ($P_{\mathcal{W}}(r) \approx H_{\mathcal{W}}(r)$). Par exemple, on factorise seulement les blocs diagonaux associés à une vitesse ou à une interface. Un phénomène essentiel est à prendre en compte dans le préconditionnement : plus r est grand, plus la matrice $H_{\mathcal{W}}(r)$ est mal conditionnée.

La proposition ci-dessous est reprise de la proposition 2.3 de [62, page 13].

Proposition 8.2.4 *Le conditionnement* $\mathrm{Cond}(H_{\mathcal{W}}(r))$ *de* $H_{\mathcal{W}}(r)$ *vérifie*

$$\mathrm{Cond}(H_{\mathcal{W}}(r)) \approx r\frac{||C_E^\top C_E + C_{\mathcal{W}}^\top C_{\mathcal{W}}||^2}{\sigma} \text{ lorsque } r \to +\infty, \tag{8.17}$$

où

$$\sigma = \inf_{v \in \mathrm{Ker}(C_E^\top C_E + C_{\mathcal{W}}^\top C_{\mathcal{W}}) - \{0\}} \frac{v^\top H v}{v^\top v},$$

La proposition 8.2.4 montre que le conditionnement de $H_{\mathcal{W}}(r)$ tend vers $+\infty$ lorsque $r \to +\infty$. De plus, pour des grandes valeurs de r il faut travailler avec un excellent préconditionneur de $r(C_E^\top C_E + C_{\mathcal{W}}^\top C_{\mathcal{W}})$ ($P_{\mathcal{W}}(r) \approx r(C_E^\top C_E + C_{\mathcal{W}}^\top C_{\mathcal{W}})$), faute de quoi le problème $(PL)^j_{\mathcal{W}}$ sera très difficile à résoudre par GC .

Dans notre code de tomography, nous utilisons un préconditionneur Jacobi ou Gauss-Seidel par blocs de $H_{\mathcal{W}}(r)$. Ce préconditionneur est en fait l'extension à notre code d'optimisation avec contraintes du préconditionneur qui est utilisé dans le code sans contrainte (voir point numéro 2 de la remarque 2.5.1, chapitre 3 et [30]). Quelques remarques sur ce préconditionneur :

1. Le coût total (en temps CPU) du preconditionnement lors de la résolution d'un problème de Lagrange peut être important puisqu'il nécessite de recalculer un préconditionneur avant chaque étape de GC.
2. Les préconditionneurs Jacobi ou Gauss-Seidel par blocs ne permettent pas de prendre en compte les termes couplés des matrice de contraintes.

8.2.4 Détection de problèmes non borné

Le système linéaire (8.12) à résoudre par GC peut s'écrire

$$(H + rC_{\bar{W}}^{\top} C_{\bar{\mathcal{W}}})\delta m = C_{\bar{W}} z_{\bar{W}} - g,$$

où $\bar{W} = E \cup \mathcal{W}$ et

$$z_{\bar{W}} := \begin{pmatrix} re - \lambda_E^j \\ ry_{\mathcal{W}} + \lambda_{\mathcal{W}}^j \end{pmatrix}.$$

Rappellons que, au cours des itérations de GC, les variables $y_{\mathcal{W}}$ sont maintenues fixées à une des deux bornes et que $y_{\mathcal{J}}$ est mis à jour pour satisfaire (8.14).

Ce système linéaire n'a pas toujours une solution. Il n'en a pas, par exemple, lorsque $H = 0$ (optimisation linéaire) et $g \notin R(C_{\bar{W}}^{\top})$ (on n'est pas maître de la donnée g). Notons que dans notre application en tomographie de réflexion nous ne sommes très préoccuppé par ce problème d'existence de la solution car la matrice $(H + rC_{\bar{W}}^{\top} C_{\bar{\mathcal{W}}})$ est rarement semi définie positive (elle est en général définie positive). Afin de généraliser l'utilisation du solveur QPAL à des problèmes semi définie positif, nous allons expliquer comment contourner ce problème.

Lorsque (8.12) n'a pas de solution et que le GC n'est pas interrompu par la rencontre d'une borne sur $y_{\mathcal{J}}$, l'algorithme finit nécessairement par générer une direction conjuguée, disons $d\delta m^{\nu}$ en δm^{ν}, dans le noyau de la matrice du système linéaire et qui est de descente pour le critère quadratique associé à (8.12) :

$$(H + rC_{\bar{W}}^{\top} C_{\bar{\mathcal{W}}})d\delta m^{\nu} = 0 \quad \text{et} \quad (g - C_{\bar{W}} z_{\bar{W}})\delta m^{\nu} \tag{8.18}$$

On note $y_{\mathcal{J}}^{\nu} := \lambda_{\mathcal{J}}/r + C_{\mathcal{J}}\delta m^{\nu}$ la variable $y_{\mathcal{J}}$ associée à δm^{ν} et $dy_{\mathcal{J}}^{\nu} := C_{\mathcal{J}} d\delta m^{\nu}$ l'accroissement de $y_{\mathcal{J}}$ correspondant à $d\delta m^{\nu}$. On sait que, le long de cette direction $d\delta m^{\nu}$, le critère quadratique associé à (8.12) est affinie et tend vers $-\infty$. On peut montrer que ce critère quadratique est aussi $L_r(\delta m, y, \lambda)$ à une constante additive près, avec $y_{\mathcal{J}}$ vérifiant (8.14). Dès lors

$$L_r(\delta m^{\nu} + \alpha.d\delta m^{\nu}, (y_{\mathcal{J}}^{\nu} + \alpha.dy_{\mathcal{J}}^{\nu}, y_{\mathcal{W}}), \lambda) \longrightarrow -\infty \text{ lorsque } \alpha \longrightarrow +\infty.$$

Comme on veut minimiser L_r, on a tout intérêt à suivre cette direction $(d\delta m^{\nu}, dy_{\mathcal{J}}^{\nu})$ en $(\delta m^{\nu}, y_{\mathcal{J}}^{\nu})$. Deux cas peuvent se présenter.

Cas 1 : aucune borne n'est rencontrée par la demi-droite $\{y_{\mathcal{J}}^{\nu} + \alpha dy_{\mathcal{J}}^{\nu} : \alpha \geq 0\}$, ce qui se produit lorsque pour tout indice $i \in \mathcal{J}$, on a

$$u_i = +\infty \text{ si } C_i d\delta m^{\nu} > 0 \text{ et } l_i = -\infty \text{ si } C_i d\delta m^{\nu} < 0. \tag{8.19}$$

On observe que, commme les matrices H et $C_{\bar{W}}^{\top} C_{\bar{\mathcal{W}}}$ sont SDP, (8.18) implique que

$$H d\delta m^{\nu} = 0, \quad C_{\bar{\mathcal{W}}} d\delta m^{\nu} = 0, \quad \text{et} \quad g^{\top} d\delta m^{\nu} = 0. \tag{8.20}$$

Alors, δm^{ν} est une direction suivant laquelle le critère quadratique du problème original décroît strictement linéairement et $\delta m^{\nu} + \alpha d\delta m^{\nu}$ est admissible, quel que soit $\alpha \geq 0$ (par

les autres relations (8.19) et (8.20)). Dès lors le (PQT) (7.3) est non borné.

Cas 2 : la demi-droite $\{y^\nu_{\mathcal{J}} + \alpha dy^\nu_{\mathcal{J}} : \alpha \geq 0\}$ rencontre une borne. Alors l'algorithme se déplace jusqu'à la première borne qui s'active (de manière à minimiser L_r au mieux), modifie l'ensemble actif $\mathcal{W}$ et redémarre une étape de GC.

8.2.5 Propriété de l'algorithme GP-AC-GC

Propriété d'identification finie des contraintes actives

Si l'on suppose que les conditions de complémentarité strictes (voir définition 4.1.12) sont satisfaites, alors la méthode du gradient avec projection (voir 8.2.1) possède la propriété importante d'identification des contraintes actives en un nombre fini d'itérations (pour les contraintes de borne voir [12], pour des contraintes linéaires générales voir [66], [23], [36], [117], [3], [157], [4], [24] et [64]).

Cette propriété est particulièrement intéressante car l'algorithme GP-AC-GC utilisé pour résoudre le problème $(PL)^j$ semble alors pouvoir tirer parti de la connaissance des contraintes actives. L'algorithme du LA possédant la propriété d'identification finie des contraintes actives fournit à l'algorithme GP-AC-GC une estimation de plus en plus précise des contraintes actives en la solution de (PQT). Ainsi, les difficultés liées à la "combinatoire" (voir section 4.1.4) du problème $(PL)^j$ doivent s'atténuées au cours des itérations de LA. C'est à dire, le nombre d'ensemble de travail $\mathcal{W}$ explorés pour résoudre $(PL)^j$ doit diminuer au cours des itérations de LA.

Convergence de l'algorithme GP-AC-GC

Théorème 8.2.5 *Supposons que le critère quadratique de $(PL)^j$ est convexe, alors l'algorithme II.6 trouve une solution de $(PL)^j$ en un nombre fini d'itérations.*

DÉMONSTRATION. Remarquons que si X_l est solution de $(PL)^j_{\mathcal{W}_l}$ sans être solution de $(PL)^j$, on a

$$\Upsilon^j(X_{l+1}) < \Upsilon^j(X_l). \tag{8.21}$$

En effet, la phase de GP de l'itération $l+1$ de l'algorithme GP-AC-GC fait décroître strictement le critère de $(PL)^j$ tandis que la phase de GC fait décroître la fonction Υ^j. On obtient donc bien l'inégalité (8.21). Cette inégalité implique que $\mathcal{W}_{l+m} \neq \mathcal{W}_l$, $\forall m \geq 1$, car X_l est la solution de $(PL)^j_{\mathcal{W}_l}$. Comme le nombre d'ensemble actif est fini et que les X_l minimisent Υ^j sur les ensembles actifs, l'algorithme finira par obtenir l'ensemble actif optimal et la convergence s'en suit. □

8.3 Algorithme GP-AC-GC amélioré

8.3.1 Simplification de l'étape GP : projection en y seulement

On maintient δm constant, on ne fait varier que y le long d'une ligne brisée obtenue par projection de la droite issue de y et portée par l'opposé du gradient g_y. Il s'agit donc de minimiser le critère de $(PL)^j$ le long de

$$(\alpha > 0) \mapsto \begin{pmatrix} \delta m \\ P_{[l,u]}(y - \alpha g_y) \end{pmatrix}. \tag{8.22}$$

L'intérêt de cette approche est de pouvoir profiter de la simplicité du critère en y. La simplification est telle que, comme nous allons le montrer, le GP en y revient à projeter le minimum sans contrainte du critère de $(PL)^j$ sur l'ensemble Y défini par les contraintes de borne (Y pourrait être un convexe fermé non vide quelconque). Par conséquent, une seule étape de GP suffit entre chaque phase de GC.

Vu comme fonction de y seul, le critère de $(PL)^j$ peut se récrire comme suit

$$q(y) = \frac{r}{2}\|y - y_*\|_2^2 + K, \tag{8.23}$$

où

$$y_* = C_I \delta m - \frac{1}{r}\lambda_I^j$$

est le minimum de q sans contrainte et K est une constante. Soit

$$\bar{y} = P_{[l,u]}(y_*)$$

la projection de y_* sur Y. Alors

$$\|\bar{y} - y_*\|_2 \leq \|y - y_*\|_2, \quad \forall y \in Y,$$

ce qui implique que

$$q(\bar{y}) \leq q(y), \quad \forall y \in Y.$$

On en déduit que $\bar{y}$ est le minimum de q sur Y. D'autre part, le GP en y minimise (8.23) le long du chemin projeté $(\alpha > 0) \mapsto P_{[l,u]}(y - \alpha g_y)$. Quel que soit y, on a $y_* = y - (1/r)g_y$, si bien que le chemin projeté passe par $\bar{y} = P_{[l,u]}(y_*)$ et le minimum le long de ce chemin est nécessairement $\bar{y}$. En conclusion, pour trouver le minimum le long du chemin projeté, il suffit de projeter y_* sur Y, opération des plus aisées. De plus, comme $\bar{y}$ est le minimum de q sur Y, il ne sert à rien de réitérer le processus, tant que δm n'a pas été modifié par une étape de GC (une phase de GP avec projection en y seuleument ne comporte qu'une seule étape de GP).

8.3.2 Test d'arrêt de Rosen

Lors de la minimisation de $(PL)^j$ (à λ_I^j fixé), en étape de GC, il est inutile de minimiser complètement le critère sur la face active courante (identifiée par l'ensemble d'indices $\mathcal{W}$) si cette face n'est pas la face optimale du problème $(PL)^j$. De manière plus précise, dans l'algorithme classique, si aucune contrainte de $\mathcal{J}$ n'est activée le long du chemin suivi par les itérés de GC, le GC résoud complètement le système (8.12). Cette résolution complète est inutile si l'ensemble $\mathcal{W}$ ne correspond pas à l'ensemble des contraintes actives en la solution de $(PL)^j$ et elle peut coûter cher en temps CPU car elle peut génèrer un nombre important d'itérations de GC. Il semble ainsi raisonnable d'arrêter l'exploration d'une face, dès qu'il devient manifeste que la face active courante n'est pas optimale. Pour détecter cette situation, on utilise le test de Rosen [137]. On introduit pour cela l'opérateur $Q_{\mathcal{W}} : \mathbb{R}^{|\mathcal{W}|} \to \mathbb{R}^{|\mathcal{W}|}$ suivant :

$$(Q_{\mathcal{W}} v)_i = \begin{cases} v_i^- = \max(0, -v_i) & \text{si } i \in \mathcal{W}_{lower} \\ v_i^+ = \max(0, v_i) & \text{si } i \in \mathcal{W}_{upper}, \end{cases}$$

où $\mathcal{W}_{lower}$ (resp. $\mathcal{W}_{upper}$) est l'ensemble des indices i des y_i bloqués sur la borne inférieure (resp. supérieure) ; avec l'hypothèse $l < u$, $\mathcal{W}_{upper} \cap \mathcal{W}_{lower} = \emptyset$. Le test consiste à arrêter les itérations de GC lorsque le résidu minimisé par le GC, c'est-à-dire $(g_{\delta m}, (g_y)_{\mathcal{J}})$, est en norme plus petit qu'une constante $\gamma_{\text{R}} > 0$ fois la norme du gradient projeté $Q_{\mathcal{W}}(g_y)_{\mathcal{W}}$. Ce dernier vecteur doit en effet être nul si l'ensemble actif $\mathcal{W}$ est optimal. Le critère est donc de la forme :

$$\|(g_{\delta m}, (g_y)_{\mathcal{J}})\| \leq \gamma_{\text{R}} \|Q_{\mathcal{W}}(g_y)_{\mathcal{W}}\|,$$

où $0 < \gamma_{\text{R}} \leq 1$ est une constante (en pratique nous prenons $\gamma_{\text{R}} = 10^{-2}$). On note cependant que dans l'algorithme décrit ci-dessus, $(g_y)_{\mathcal{J}} = 0$, parce que (8.14) est toujours vérifiée au cours des itérations de GC, si bien que le test de Rosen devient

$$\|g_{\delta m}\| \leq \gamma_{\text{R}} \|Q_{\mathcal{W}}(g_y)_{\mathcal{W}}\|. \tag{8.24}$$

En pratique, le test de Rosen nécessite de connaître le vecteur $(g_y)_{\mathcal{W}}$ au cours des itérations de GC. Ainsi, chaque fois que l'on met à jour la variable δm par $\delta m_+ := \delta m + \alpha d\delta m$, on doit aussi mettre à jour la variable $(g_y)_{\mathcal{W}}$ par $((g_y)_{\mathcal{W}})_+ := (g_y)_{\mathcal{W}} - \alpha r C_{\mathcal{W}} d\delta m$. En effet, un déplacement $d\delta m$ en δm et dy en y correspond à un déplacement $dg_y = r(dy - C_I d\delta m)$ en g_y. On en déduit ainsi le déplacement en $g_{(y_{\mathcal{W}})}$ par $d(g_y)_{\mathcal{W}} = r(dy_{\mathcal{W}} - C_{\mathcal{W}} d\delta m) = -rC_{\mathcal{W}} d\delta m$ (les contraintes actives $y_{\mathcal{W}}$ étant fixées au cours des itérations de GC on a $dy_{\mathcal{W}} = 0$)

Une manière de savoir si le critère de Rosen est efficace est de le tester sur notre banc d'essaie de problèmes de tomographie. Les résultats sont présentés à la figure 8.1 au moyen de profils de performance (voir l'article [50] et l'annexe E pour la construction et l'interprétation d'un profil de performance) dans lequel on compare le nombre d'itérations de GC effectuées avec ou sans ce critère. Rappelons en effet que ce critère d'arrêt a été mis en œuvre pour diminuer le nombre d'itérations de GC générées par le code GP-AC-GC. Dans la figure 8.1, nous avons tracé un profil de performance qui compare le

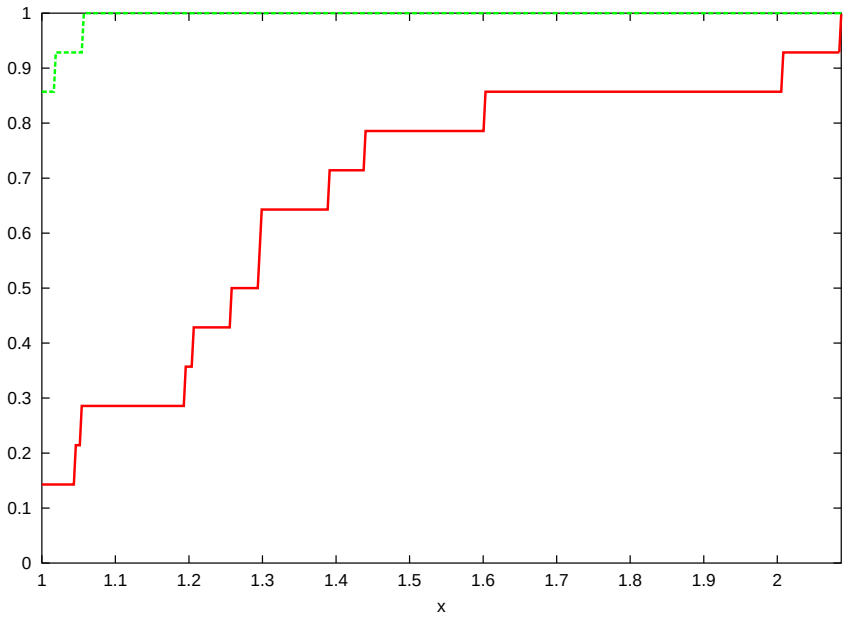

Fig. 8.1 Profil de performance sur les itérations de GC : Avec (courbe verte en pointillé) ou sans (courbe rouge) le critère d'arrêt de Rosen

nombre d'itérations de GC réalisées par le code GP-AC-GC sans le critère d'arrêt de Rosen (courbe rouge) et celui avec ce critère (courbe verte en pointillé). Ces profils de performance ont été obtenus en testant tous les exemples de notre bibliothèque de modèles en tomographie de réflexion (cf. annexe D pour la description de cette bibliothèque de modèle). Pour chaque exemple testé, nous avons noté le nombre total d'itérations de GC effectuées lors de la première itérations de GN. Cette figure montre clairement que le code GP-AC-GC avec le critère de Rosen est le plus performant : la courbe verte est située bien au-dessus de la courbe rouge. Pour environ 85% des exemples testés (voir valeur obtenue pour $\tau = 1$), le nombre d'itérations de GC effectuées par le code GP-AC-GC avec le critère d'arrêt de Rosen est inférieur à celui obtenu par le code sans ce critère. De plus, pour chaque problème le code avec le critère de Rosen n'est jamais moins bon que $1,1$ fois ce qu'a fait le code sans Rosen alors que le code sans le critère de Rosen est parfois $2,1$ fois moins bon que le code avec Rosen (i.e., le code sans le critère de Rosen demande parfois $2,1$ fois plus d'itérations de GC).

8.3.3 Activation rapide de contraintes entre deux étapes consécutives de GC

La majorité des étapes de GC ne servent principalement qu'à activer des contraintes de borne inactives (le GC s'arrêtant dès qu'une contrainte de borne non active est rencontrée), jusqu'à convergence vers un ensemble de travail $\mathcal{W}$ stable (cf. définition 8.3.1 suivante) où on minimisera complètement $(PL)^j_{\mathcal{W}}$ (ou partiellement avec le test d'arrêt de Rosen).

Définition 8.3.1 *On dit qu'un ensemble de travail $\mathcal{W}$ est stable lorsque le chemin suivi par le GC pour minimiser complètement (ou partiellement avec le test d'arrêt de Rosen) le problème $(PL)^j_{\mathcal{W}}$ est admissible pour les contraintes de borne inactives de $\mathcal{J}$.*

Comme l'algorithme II.6 de GP-AC-GC classique n'active qu'une seule contrainte de borne à la fois entre deux étapes consécutives de GC, un nombre important d'étapes de GC risquent d'être générées afin de parvenir à un ensemble de travail $\mathcal{W}$ stable. Ces étapes consécutives de GC sont chères en temps CPU pour deux raisons. La première raison est que, indépendamment du nombre d'itérations de GC effectuées, chaque étape de GC a un coût fixe relativement important dû à l'initialisation du préconditionneur de $H_{\mathcal{W}}$ (cf. section 8.2.3). La deuxième raison est que chaque itération de GC nécessite un produit matrice-vecteur relativement coûteux dans notre application. De plus, on observe souvent que peu d'itérations de GC sont nécessaires avant de rencontrer une borne et donc, que le critère quadratique du problème $(PL)^j_{\mathcal{W}}$ n'est que très légèrement minimisé. Ainsi, le coût cumulé en temps CPU engendré par l'identification d'un ensemble de travail $\mathcal{W}$ stable dans l'algorithme II.6 peut s'avérer très rapidement important (dans le pire des cas il faudra n_I étapes de GC pour converger vers un ensemble $\mathcal{W}$ stable : après un changement de phase GP $\Longrightarrow$ GC, on démarre le premier GC avec aucune contrainte de bornes actives $\mathcal{W} = \emptyset$, puis on les active toutes une par une par des étapes de GC consécutives). En pratique, on a déjà observé ce problème où beaucoup d'étapes de GC

sont généralement nécessaires pour parvenir à un ensemble de travail $\mathcal{W}$ stable. Afin de remédier d'y remedier, nous avons mis au point une technique d'activation rapide de contraintes entre chaque étape de GC. Pour ce faire, après chaque étape de GC, nous avons choisi d'utiliser une méthode de descente avec projection sur l'ensemble des contraintes inactives $\mathcal{J}$. On l'appellera par la suite méthode de direction projetée (DP). La méthode DP est similaire à la méthode GP sauf que l'on s'autorise ici à projeter une direction qui peut être différente de celle formée par l'opposé du gradient. Comme la projection d'une direction autre que l'opposée du gradient n'est pas nécessairement une direction de descente, pour s'assurer que le chemin projeté est de descente on veillera à ce que $d = (d_{\delta m}, d_y)$ satisfasse :

$$\begin{cases} (a) \text{ Pour } \alpha \text{ petit, } P_{[l,u]}(y + \alpha d_y) = y + \alpha d_y \\ (b) \text{ La direction } d \text{ est une direction de descente du critère quadratique de } (PL)^j \\ (c) \ (d_y)_i > 0 \text{ si } y_i = l_i \text{ et } (d_y)_i < 0 \text{ si } y_i = u_i \end{cases} \tag{8.25}$$

Quelques remarques sur ces propriétés :

1. Les conditions (a) et (b) assurent que l'on fait bien décroître le critère quadratique de $(PL)^j$ en se déplaçant dans la direction projetée de d.
2. La condition (c) assure que l'on se déplace dans l'ensemble de travail $\mathcal{W}$ courant et donc que l'on ne va pas désactiver des contraintes par projection de la direction d. En général cette condition n'est pas indispensable. Dans notre cas elle est utile car nous voulons utiliser la méthode DP pour activer des contraintes sans en désactiver après une étape de GC (étape dans laquelle $y_{\mathcal{J}}$ sort des bornes).

Afin d'appliquer la méthode DP, après une minimisation par GC du problème $(PL)^j_{\mathcal{W}}$, on a besoin de choisir un point de départ $X_{\mathcal{J}}$ admissible et une direction d à projeter (la direction de descente devant vérifier les propriétés de (8.25)). Pour ce faire, trois types d'approches différentes sont envisageables.

Première possibilité (non testée). On résout complètement (8.12) par GC sans tenir compte des bornes sur $y_{\mathcal{J}}$ et on note $\widehat{X}_{\mathcal{J}}$ sa solution. Si le point $\widehat{X}_{\mathcal{J}} = \begin{bmatrix} \widehat{\delta m} \\ \widehat{y} \end{bmatrix}$ n'est pas admissible, i.e. $\widehat{y} \notin [l, u]$, alors on applique la méthode DP avec pour point de départ le point $X_{\mathcal{J}_0}$ qui correspond au point $X_{\mathcal{J}}$ initialisant l'étape de GC et avec pour direction $d = \widehat{X}_{\mathcal{J}} - X_{\mathcal{J}_0}$. Cette approche revient à modifier légèrement la première possibilité du GC décrite dans la section 8.2.2. Or nous avons déjà rejeté cette approche car elle est très coûteuse en temps CPU : un grand nombre d'itérations de GC est nécessaire avant d'obtenir la solution $X_{\mathcal{J}}$ du système (8.12).

Deuxième possibilité (la possibilité retenue). On choisit d'utiliser la deuxième possibilité du GC décrite dans la section 8.2.2. Si $y_{\mathcal{J}}$ sort des bornes au cours des itérations de GC on note $X_{\mathcal{J}_{iborne}}$ la solution obtenue par GC tronqué juste avant la sortie des bornes et de même on note $X_{\mathcal{J}_{iborne+1}}$ la solution obtenue par GC tronqué juste après la sortie des bornes ($iborne$ représente l'indice d'itération de GC obtenue juste avant la sortie des bornes). Ensuite on applique la méthode DP avec pour point de départ le point $X_{\mathcal{J}_{iborne}}$ et

avec pour direction $d = X_{\mathcal{J}_{iborne+1}} - X_{\mathcal{J}_{iborne}}$ (notons qu'il existe une variante ou on prendra $d = X_{\mathcal{J}_{iborne+1}} - X_{\mathcal{J}_0}$). Cette possibilité permet de minimiser le nombre d'itérations de GC à effectuer à chaque étape de GC.

Troisième possibilité (non testée). On résout partiellement (8.12) par GC avec le critère de Rosen, tout en faisant évoluer $y_{\mathcal{J}}$ de manière à minimiser le critère quadratique de $(PL)^j_{\mathcal{W}}$. Si $y_{\mathcal{J}}$ ne sort pas des bornes alors tout se passe comme dans la deuxième possibilité du GC décrite dans la section 8.2.2. Sinon, si $y_{\mathcal{J}}$ sort des bornes, alors on désactive le critère de Rosen, et on pousse les itérations de GC jusqu'à satisfaction d'un nouveau critère :

$$\|(g_{\delta m}, g_{(y_{\mathcal{J}})})\| \leq \gamma_{\text{DP}} \|Q_{\mathcal{W}} g_{(y_{\mathcal{W}})}\|.$$

Ce nouveau critère, que l'on appellera critère de la Direction Projetée, est semblable au critère de Rosen sauf que, cette fois-ci, la constante γ_{DP} du critère doit être choisie supérieure à 1 : $\gamma_{\text{DP}} \geq 1$ (on peut voir ce nouveau critère comme le critère de Rosen "relaxé", nous prenons en pratique $\gamma_{\text{DP}} = 10^2$). Une fois le critère de la Direction Projetée satisfait on note $X_{\mathcal{J}_{DP}}$ la solution obtenue par le GC. Puis on applique la méthode DP avec pour point de départ le point $X_{\mathcal{J}_0}$ qui correspond au point $X_{\mathcal{J}}$ initialisant l'étape de GC et avec pour direction $d = X_{\mathcal{J}_{DP}} - X_{\mathcal{J}_0}$. Cette approche permet de trouver une direction de projection pour l'étape DP mieux équilibrée que la direction choisie dans la deuxième possibilité précédente. Elle devrait donc permettre d'activer des contraintes qui ont potentiellement plus de chance de figurer dans l'ensemble des contraintes actives en la solution du problème $(PL)^j$. En d'autre termes, cette approche permet de diminuer le nombre total de phases de GC générées par l'algorithme GP-AC-GC. Reste maintenant à savoir si cette diminution compense l'augmentation du nombre d'itérations de GC provoquées par l'équilibrage de la direction .

Une remarque sur la méthode DP :

1. On peut se demander pourquoi dans les possibilités précédentes on n'a pas utilisé la technique de la section 8.3.1 qui consiste à projeter d en y seulement. En fait, la direction formée par le vecteur $\begin{bmatrix} 0 \\ d_y \end{bmatrix}$ n'a aucune garantie d'être une direction de descente du critère quadratique de $(PL)^j$. Comme cela contredit la propriété (b) de (8.25), on ne peut donc pas envisager cette technique dans le cadre de notre méthode DP.

Comme pour le critère d'arrêt de Rosen, l'étape supplémentaire de DP à l'intérieur du code GP-AC-GC a été mise en œuvre pour diminuer le nombre d'itérations de GC. Parmi les trois possibilité différentes pour programmer une étape de DP (voir ci-dessus) nous avons retenue la seconde possibilité. Ainsi, en pratique, pour savoir si cette étape supplémentaire de DP est efficace, on réalise un profil de performance (voir l'article [50] et l'annexe E pour la construction et l'interprétation d'un profil de performance) dans lequel on compare le nombre d'itérations de GC effectuées avec ou sans cette étape. Dans la figure 8.2, nous avons tracé ce profil de performance : il compare le nombre d'itérations de GC réalisées par le code GP-AC-GC sans l'étape supplémentaire de DP (courbe rouge) et celui avec cette étape supplémentaire (courbe verte en pointillé). Ce profil de performance

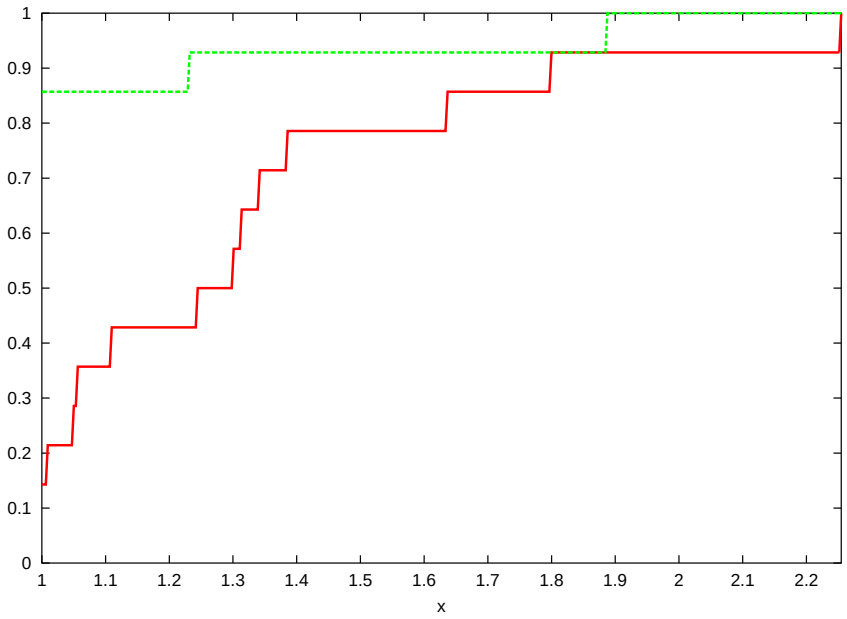

Fig. 8.2 Profil de performance sur les itérations de GC : Avec (courbe verte en pointillé) ou sans (courbe rouge) l'étape de DP

a été obtenu en testant tous les exemples de notre bibliothèque de modèles en tomographie de réflexion (cf. annexe D pour la description de cette bibliothèque de modèles). Pour chaque exemple testé, nous avons noté le nombre total d'itérations de GC effectuées lors de la première itération de GN. Cette figure montre clairement que le code GP-AC-GC avec l'étape de DP (seconde possibilité) est le plus performant : la courbe verte est située au-dessus de la courbe rouge. Pour environ 85% des exemples testés (voir valeur obtenue pour $\tau = 1$), le nombre d'itérations de GC obtenu par le code GP-AC-GC avec l'étape de DP est inférieur à celui obtenu par le code sans cette étape. De plus, pour chaque problème le code avec l'étape de DP n'est jamais moins bon que $1,9$ fois ce qu'a fait le code sans cette étape alors que le code sans l'étape de DP est parfois $2,2$ fois moins bon que le code avec cette étape (i.e., le code sans l'étape de DP demande parfois $2,2$ fois plus d'itérations de GC).

8.4 Prise en compte explicite de contraintes de borne dans QPAL

Dans cette section nous décrivons une méthode permettant de traiter explicitement des contraintes de borne dans notre solveur QPAL. Notons que cette méthode n'a pas été testée.

Le cas des contraintes de borne sur δm est un cas particulier de la formulation générale du problème quadratique tangent (7.1). On peut reformuler celui-ci comme :

$$\begin{cases} \min\limits_{\delta m \in \mathbb{R}^n} \; F(\delta m) \\ C_E \delta m = e \\ l \leq C_I \delta m \leq u \\ \delta m_{inf} \leq \delta m \leq \delta m_{sup}, \end{cases} \tag{8.26}$$

où $\delta m_{inf} \in \bar{\mathbb{R}}^n$ (resp. $\delta m_{sup} \in \bar{\mathbb{R}}^n$) est le vecteur contenant les bornes inférieures (resp. supérieures) sur δm.

On conserve la définition (7.4) du lagrangien augmenté car on ne veut pas relaxer les contraintes de borne. Ainsi, le problème de Lagrange à résoudre à chaque itération de LA s'écrit :

$$\begin{cases} \min\limits_{X=(\delta m, y)} \; \Upsilon^j(X) \\ l \leq y \leq u \\ \delta m_{inf} \leq \delta m \leq \delta m_{sup}. \end{cases} \tag{8.27}$$

Ce dernier peut se résoudre par l'algorithme GP-AC-GC (voir II.6). Notons qu'à chaque étape de GC, on doit résoudre par l'algorithme du GC le problème quadratique sans contrainte suivant :

$$\begin{cases} \min\limits_{X_{\mathcal{J}}} \left(\frac{1}{2} X_{\mathcal{J}}^\top Q_{\mathcal{J}} X_{\mathcal{J}} + X_{\mathcal{J}}^\top b_{\mathcal{J}}^j \right) \\ \delta m_i = \delta m_{inf} \text{ ou } \delta m_i = \delta m_{sup}, \quad i \in \mathcal{W}', \end{cases} \tag{8.28}$$

où $\mathcal{W}'$ est l'ensemble de travail des contraites de borne sur δm. En suivant la deuxième possibilité de l'étape GC, il faut stopper les itérations de GC dès que l'on rencontre une nouvelle borne sur δm ou sur y.

8.5 Autour de la propriété d'identification finie des contraintes actives

Pour $m \in \mathbb{R}^n$ donné, on note :

$$\tilde{l} := l - C_I m \qquad \text{et} \qquad \tilde{u} = u - C_I m.$$

Proposition 8.5.1 *Supposons que les conditions de complémentarité stricte sont satisfaites en une solution primale-duale régulière (voir remarque 1. de 4.2.5) $(\widehat{m}, \widehat{\mu})$ de (P_{EI}). On considère l'algorithme SQP avec pas unité. Supposons aussi que l'on démarre l'algorithme II.5 du LA en $\lambda^0 = \mu$. Alors, le vecteur y déterminé par l'algorithme GP-AC-GC (avec un GP en y seuleument, voir section 8.3.1) lors de la résolution du premier problème de Lagrange identifie correctement les contraintes actives en la solution.*

DÉMONSTRATION. On note (m, μ) l'itéré courant de l'algorithme SQP, (m_+, μ_+) l'itéré suivant et $(\widehat{m}, \widehat{\mu})$ la solution de (P_{EI}).

Soit $i \in I$ l'indice d'une contrainte inactive en la solution : $l_i < C_i\widehat{m} < u_i$. Pour m suffisamment proche de $\widehat{m}$, on a $l_i < C_i m < u_i$, donc

$$\tilde{l}_i < 0 < \tilde{u}_i.$$

Lorsque l'on a complémentarité stricte en $(\widehat{m}, \widehat{\mu})$ et que l'itéré (m, μ) est suffisamment proche de $(\widehat{m}, \widehat{\mu})$, l'algorithme SQP avec pas unité est tel que $\mu_i = 0$[3]. L'algorithme du LA démarrant avec $\lambda^0 = \mu$, on a $\lambda^0_i = 0$. L'algorithme du LA initialise $\delta m^0 = 0$. L'algorithme GP-AC-GC, qui résout le premier problème de Lagrange, initialise y par

$$y = P_{[\tilde{l},\tilde{u}]}\left(C_I \delta m^0 - \frac{1}{r}\lambda^0_I\right).$$

Comme $\tilde{l}_i < 0 < \tilde{u}_i$, on obtient

$$y_i = P_{[\tilde{l}_i,\tilde{u}_i]}\left(C_i \delta m^0 - \frac{\lambda^0_i}{r}\right) = P_{[\tilde{l}_i,\tilde{u}_i]}(0) = 0.$$

Ainsi $\tilde{l}_i < y_i < \tilde{u}_i$ et le y_i initial choisi pour résoudre le problème de Lagrange est inactif comme la contrainte i.

[3] En effet, si $(\delta m^{\text{PQ}}_-, \lambda^{\text{PQ}}_-)$ est une solution primale-duale du PQ osculateur en (m_-, μ_-) (l'itéré précédant (m, μ)), on a $l_i < C_i(m_- + \delta m^{\text{PQ}}_-) < u_i$ (identification finie des contraintes actives par le PQ osculateur en cas de complémentarité stricte) et donc $(\lambda^{\text{PQ}}_-)_i = 0$. Par hypothèse, $\mu = \lambda^{\text{PQ}}_-$, donc $\mu_i = 0$.

Soit à présent $i \in I$ l'indice d'une contrainte active en la solution : $C_i\widehat{m} = l_i$ (le raisonnement est identique si la borne supérieure est active). Lorsque l'on a complémentarité stricte en $(\widehat{m}, \widehat{\mu})$ et que l'itéré (m, μ) est suffisamment proche de $(\widehat{m}, \widehat{\mu})$, l'algorithme SQP avec pas unité est tel que $C_i m = l_i$[4]. Alors

$$\tilde{l}_i = 0 < \tilde{u}_i.$$

D'autre part, la solution $(\widehat{m}, \widehat{\mu})$ vérifiant la complémentarité stricte, on a $\widehat{\mu}_i > 0$ (par convention $\widehat{\mu}_I := \widehat{\mu}_l - \widehat{\mu}_u$ et ici $(\widehat{\mu}_l)_i > 0$, $(\widehat{\mu}^u)_i = 0$) et donc aussi $\mu_i > 0$ si l'itéré courant (m, μ) est proche de $(\widehat{m}, \widehat{\mu})$. On a donc

$$-\frac{\lambda_i^0}{r} < \tilde{l}_i,$$

ce qui implique que

$$y_i = P_{[\tilde{l}_i, \tilde{u}_i]}\left(C_i\delta m^0 - \frac{\lambda_i^0}{r}\right) = P_{[\tilde{l}_i, \tilde{u}_i]}\left(-\frac{\lambda_i^0}{r}\right) = \tilde{l}_i.$$

Ainsi, le y_i initial choisi pour résoudre le problème de Lagrange est actif sur la borne inférieure tout comme la contrainte i. □

Pour aller plus loin, il faudrait

- montrer que le y calculé par l'algorithme GP-AC-GC n'est pas trop différent du y de départ, ce qui impliquerait que l'on ne change pas d'ensemble actif lors de la résolution du premier problème de Lagrange et que celui-ci est résolu en une seule double phase GP-GC,
- utiliser la propriété d'identification finie de l'algorithme du LA pour pouvoir dire que les nouveaux μ^j ont les mêmes propriétés que λ^0 (on sait que c'est vrai asymptotiquement pour un problème quadratique donné, mais il faudrait dire ici que cette propriété est vraie pour tous les μ^j ; il faudrait en fait démontrer une propriété d'uniformité par rapport à de petites variations des données définissant les problèmes quadratiques, qui varient au cours des itérations de SQP).

[4] De manière plus précise, on a $m = m_- + \delta m_-^{\text{PQ}}$ et $C_i(m_- + \delta m_-^{\text{PQ}}) = l_i$.

Chapitre 9

Résolution du problème dual

On rappelle (cf. équation (7.8)) que le problème dual (PD) associé au LA consiste à minimiser la fonction duale régularisée δ_r (cf. équation (7.7)) :

$$(PD) : \inf_{\lambda} \ \delta_r(\lambda) \tag{9.1}$$

On rappelle aussi que L_0 est le lagrangien (non augmenté) associé au (PQT) :

$$L_0(\delta m, y, \lambda) = F(\delta m) + \lambda_E^\top (C_E.\delta m - e) + \lambda_I^\top (y - C_I.\delta m). \tag{9.2}$$

La fonction duale classique (non régularisée) δ_0 associée au lagrangien (7.4) est définie par :

$$\delta_0(\lambda) := \inf_{\delta m, l \leq y \leq u} L_0(\delta m, y, \lambda) \tag{9.3}$$

9.1 Méthode des multiplicateurs

Dans la méthode des multiplicateurs de [93] et [126], les multiplicateurs de Lagrange sont mis à jour par la formule suivante :

$$\begin{cases} \lambda_E^{j+1} = \lambda_E^j + r_j(C_E \delta m^{j+1} - e) \\ \lambda_I^{j+1} = \lambda_I^j + r_j(y^{j+1} - C_I \delta m^{j+1}). \end{cases} \tag{9.4}$$

Cette formule de mise à jour des multiplicateurs était vue au départ comme une heuristique, qui assure la nullité de $\nabla L_0(\delta m^{j+1}, y^{j+1}, \lambda^{j+1})$.

9.1.1 Une méthode de gradient sur δ_r

Sous un premier angle, on peut voir la méthode des multiplicateurs comme l'algorithme du gradient avec un pas constant égal à r_j pour minimiser la fonction duale régularisée δ_r. En effet, dans l'algorithme II.5, lorsqu'à l'itération j de LA on résout

le $(PL)^j$ en obtenant la solution primale $(\delta m^{j+1}, y^{j+1})$, cela équivaut à évaluer la valeur de la fonction duale régularisée en $(\lambda_E^j, \lambda_I^j)$:

$$\begin{aligned}\delta_r(\lambda^j) &= -\inf_{\delta m, l\le y\le u}\Big(L_r(\delta m, y, \lambda_E^j, \lambda_I^j)\Big)\\ &= -\Big(F(\delta m^{j+1}) + \xi_E(\delta m^{j+1}, \lambda_E^j) + \xi_I(\delta m^{j+1}, y^{j+1}, \lambda_I^j)\Big)\end{aligned}$$

La fonction δ_r est différentiable et son gradient en λ^j vaut (voir [94] et [134]) :

$$\nabla_\lambda \delta_r(\lambda^j) = -\begin{bmatrix} C_E\delta m^{j+1} - e \\ y^{j+1} - C_I\delta m^{j+1}\end{bmatrix}$$

On peut donc reformuler la méthode (9.4) par la méthode du gradient avec un pas fixe (sans recherche linéaire) égal au paramètre d'augmentation r_j :

$$\begin{bmatrix}\lambda_E^{j+1}\\ \lambda_I^{j+1}\end{bmatrix} = \begin{bmatrix}\lambda_E^{j}\\ \lambda_I^{j}\end{bmatrix} + r_j\Big(-\nabla_{\lambda^j}\delta_r(\lambda^j)\Big).$$

Dans cette première interprétation de la méthode des multiplicateurs le pas n'est pas déterminé par recherche linéaire, il est fixé à r.

9.1.2 Une méthode proximale sur δ_0

Rockafellar [134] a donné une autre interprétation de la méthode des multiplicateurs qui est aussi une méthode proximale sur la fonction duale δ_0, c'est à dire une méthode de gradient implicite sur δ_0. L'algorithme proximal génère une suite $\{(\lambda^j)\}$ par la formule

$$\lambda^{j+1} := P_{\delta_0}(\lambda^j) := \text{ point proximal de } \lambda^j.$$

De manière plus précis, λ^{j+1} est l'unique solution de :

$$\inf_\lambda \left(\delta_0(\lambda) + \frac{1}{2r}||\lambda - \lambda^j||^2\right). \tag{9.5}$$

La proposition suivante établit le lien entre la fonction duale régularisée δ_r du lagrangien augmenté et la fonction duale classique δ_0 du lagrangien (voir [134] pour une preuve de cette propriété).

Proposition 9.1.1 *δ_r est la régularisée de Moreau-Yosida de δ_0.*

On peut donc écrire

$$\begin{bmatrix}\lambda_E^{j+1}\\ \lambda_I^{j+1}\end{bmatrix} = \begin{bmatrix}\lambda_E^{j}\\ \lambda_I^{j}\end{bmatrix} - r_j g_{j+1}, \tag{9.6}$$

avec $g_{j+1} \in \partial\delta_0(\lambda^{j+1})$. D'après la formule précédente, la méthode proximale s'interprète comme une méthode de sous-gradient implicite : le calcul du sous-gradient g_{j+1} est effectué en (λ^{j+1}) plutôt qu'en (λ^j). Cette interprétation de la méthode du LA en terme d'algorithme proximal est intéressante car elle permet alors d'obtenir des propriétés de convergence (cf. article en annexe C).

9.2 Une méthode alternative au LA : méthode de BFGS appliquée à la minimisation de la fonction duale régularisée

D'après la propriété 9.1.1, δ_r et la régularisée de Moreau Yosida de δ_0. Alors δ_r est au moins de régularité $C^{1,1}$. On peut ainsi envisager une méthode quasi-newtonienne pour minimiser δ_r, celle-ci étant bien plus rapide que la méthode classique du gradient. La méthode de BFGS, qui est une méthode quasi-newtonienne, a justement besoin d'une régularité minimale $C^{1,1}$ pour être convergente (voir [128]).

9.2.1 Algorithme de BFGS appliqué à la minimisation de la fonction duale régularisée

Voici ci-dessous l'algorithme de BFGS appliqué à la minimisation de δ_r.

Data : Constantes de la recherche linéaire : $\omega_1 \simeq 10^{-4}$ et $\omega_2 \simeq 0.99$.
Initialisation du paramètre d'augmentation : $r_0 > 0$.
$n_C = n + n_I$ et $j = 0$.

begin

((1)) Choix de $M_0 \in \mathbb{S}^{n_C}_{++}$: $M_0 = r_0.I_{n_C}$.
((2)) $(\delta m^0, y^0)$: solution du problème de Lagrange $(PL)^0$.
((3)) Valeur initiale de la fonction duale régularisée : $\delta_r(\lambda^0)$.
((4)) Valeur du gradient de δ_r en λ^0 : $\widetilde{g}_0 = \begin{bmatrix} e - C_E\delta m^0 \\ C_I\delta m^0 - y^0 \end{bmatrix}$.

repeat

if *($r_j \neq r_{j-1}$ pour $j \geq 1$)* **then**

j=0.
Retour à l'étape ((1)).

endif

((5.1)) Direction de recherche : $d_j = -M_j\widetilde{g}_j$.
((5.2)) Pas initial $\alpha_j = 1$.
((5.3)) Recherche linéaire de Wolfe ; trouver le pas α_j tel que :
$\delta_r(\lambda^{j+1}) \leq \delta_r(\lambda^j) + \omega_1\alpha_j\widetilde{g}_j^\top d_j$ et $\widetilde{g}_{j+1}^\top d_j \geq \omega_2\widetilde{g}_j^\top d_j$,
avec $\lambda^{j+1} = \lambda^j + \alpha_j d_j$.
((5.4)) Mise à jour de M_{j+1} :
$s_j := \lambda^{j+1} - \lambda^j, \quad u_j := \widetilde{g}_{j+1} - \widetilde{g}_j, \quad M_{j+1} = \overline{\text{BFGS}}(M_j, u_j, s_j)$.
((5.5)) Choisir un nouveau paramètre d'augmentation $r_{j+1} > 0$.
((5.6)) $j = j + 1$.

until *($\kappa(\delta m^j, y^j) \simeq 0$)*

end

Algorithme II.7 Algorithme de BFGS appliqué à la minimisation de la fonction duale régularisée

Cet algorithme mérite quelques remarques :

1. L'opérateur de $\overline{\text{BFGS}}$ représente la formule de BFGS inverse. M_j est donc une approximation de l'inverse du "hessien" de la fonction duale régularisée. Comme le problème inverse de tomographie avec contraintes est un problème de grande taille (n_C grand), on utilisera la technique l-BFGS (voir [122]) et plus particulièrement l'algorithme implémenté dans le code *M1QN3* de la librairie *Modulopt* de l'INRIA (voir []).
2. A l'étape ((2)) de l'algorithme, on a choisi d'initialiser l'approximation de l'inverse du "hessien" de la fonction duale régularisée par $M_0 = r_0.I_{n_C}$. Cette initialisation n'est pas anodine : elle implique qu'à la première itération de BFGS ($j = 1$) on met à jour les multiplicateurs de Lagrange par

$$\lambda^1 = \lambda^0 + \alpha_0 d_0,$$

or $d_0 = -M_0\widetilde{g}_0 = -r_0\widetilde{g}_0$, donc :

$$\lambda^1 = \lambda^0 - \alpha_0 r_0 \widetilde{g}_0,$$

Dans cette dernière expression, on se rend compte que si le pas unité ($\alpha_0 = 1$) est accepté par la recherche linéaire, alors on retrouve la formule de mise a jour de l'algorithme proximal (voir (9.6)) pour λ^1 car le sous-gradient de la fonction duale classique δ_0 en λ^{j+1} est alors égal au gradient de la fonction duale régularisée en λ^j : $g_{j+1} = \widetilde{g}_j$. Ainsi, dans l'algorithme II.7, si le pas unité $\alpha_0 = 1$ est accepté par la recherche linéaire, la mise à jour du premier itéré λ^1 est équivalente à une mise a jour par la méthode des multiplicateurs :

$$\lambda^1 = \lambda^0 - \alpha_0 r_0 g_1,$$

avec g_1 le sous-gradient de la fonction duale classique δ_0 en λ^1. De plus, nous savons que l'algorithme proximal est non seulement une méthode de descente sur la fonction duale classique mais aussi sur la fonction duale régularisée (la direction $-r_j g_{j+1}$ est bien une direction de descente de δ_r en λ^j car $g_{j+1} = \widetilde{g}_j$ est le gradient de δ_r en λ^j). La propriété suivante explique pourquoi le pas unité, qui est systématiquement imposé par l'algorithme proximal le long de la direction $-r_j g_{j+1}$, fait non seulement décroître la fonction duale classique δ_0 à chaque itération mais aussi la fonction duale régularisée.

Proposition 9.2.1 *Considérons les itérés $\{\lambda^j\}$ générés par l'algorithme proximal (cf. section 9.1.2) alors on a les inégalités suivantes :*

$$\begin{cases} \delta_0(\lambda^{j+1}) \leq \delta_0(\lambda^j) \\ \delta_r(\lambda^{j+1}) \leq \delta_r(\lambda^j) \end{cases}$$

DÉMONSTRATION. On rappelle que δ_r en (λ^j) s'écrit

$$\delta_r(\lambda^j) = \inf_{\lambda} \left(\delta_0(\lambda) + \frac{1}{2r_j} ||\lambda - \lambda^j||^2 \right). \tag{9.7}$$

En prenant $\lambda_1 = \lambda^j$ à droite, on trouve :

$$\delta_r(\lambda^j) \leq \delta_0(\lambda^j). \tag{9.8}$$

D'autre part, comme λ^{j+1} est le point proximal de λ^j, on peut écrire

$$\delta_r(\lambda^j) = \delta_0(\lambda^{j+1}) + \frac{1}{2r_j} ||\lambda^{j+1} - \lambda^j||^2$$

ce qui implique l'inégalité

$$\delta_r(\lambda^j) \geq \delta_0(\lambda^{j+1}). \tag{9.9}$$

L'inégalité (9.8) est vraie pour l'indice $j+1$:

$$\delta_r(\lambda^{j+1}) \leq \delta_0(\lambda^{j+1}). \tag{9.10}$$

En rassemblant les trois inégalités (9.8), (9.9) et (9.10) on obtient le résultat attendu :

$$\begin{cases} \delta_0(\lambda^{j+1}) \leq \delta_0(\lambda^j) \\ \delta_r(\lambda^{j+1}) \leq \delta_r(\lambda^j). \end{cases}$$

□

Cette propriété implique que la première itération de l'algorithme II.7 peut être vu comme une itération de la méthode des multiplicateurs. En effet, la recherche linéaire effectuée à l'étape ((5.3)) de l'algorithme a toutes les chances d'accepter le pas unité ($\alpha_0 = 1$) puisque $\delta_r(\lambda_E^1, \lambda_I^1) \leq \delta_r(\lambda_E^0, \lambda_I^0)$ [1].

3. Dans l'étape de recherche linéaire, chaque vérification de la condition de Wolfe pour un pas α_j donné nécessite aussi le calcul de la solution $(\delta m^{j+1}, y^{j+1})$ du problème de Lagrange $(PL)^{j+1}$. Ainsi, cette étape peut paraître coûteuse *a priori* si beaucoup d'essais de pas α_j sont réalisés. Cependant, l'expérience montre que dans 90% des cas le pas unité ($\alpha_j = 1$) est accepté. Il y aurait donc un surcoût d'environ 10% à faire de la RL. Reste maintenant à savoir si cette méthode est plus efficace que la méthode des multiplicateurs (voir section 9.1).
4. Cet algorithme remplace complètement l'algorithme II.5 du LA pour résoudre le (PQT). Ainsi, on n'est plus dans le cadre classique de la méthode proximale (voir 9.1.2) pour minimiser la fonction duale classique δ_0.
5. L'une des grandes différences de cet algorithme avec l'algorithme II.5 du LA est que la valeur du paramètre d'augmentation r_j est constante au cours des itérations j de BFGS. En effet, en cas de changement de paramètre d'augmentation r ($r_{j-1} \neq r_j$) la fonction duale régularisée δ_r change aussi et tout ce qui a été engendré dans M_j devient obsolète. Ainsi, dès qu'un changement du paramètre d'augmentation r a lieu l'algorithme retourne à l'étape ((1)). D'après la remarque 2 ci-dessus, on déduit que si le paramètre d'augmentation r change tout le temps ($r_1 \neq r_0$ pour tous les redémarrages de l'algorithme), alors cet algorithme ce comporte exactement comme la méthode des multiplicateurs (i.e. l'algorithme proximal sur la fonction duale classique δ_0).
6. Notons que nous savons pas vraiment comment changer la valeur du paramètre d'augmentation r à l'étape ((5.5)) de l'algorithme ci-dessus. Cependant, on peut trouver une borne supèrieure de r en regardant le conditionnement du problème de Lagrange. Pour ce faire on procèdera comme indiqué dans la section 10.1.1.

9.2.2 Convergence de l'algorithme II.7

Voici ci-dessous le théorème de convergence de l'algorithme classique de BFGS (repris du théorème 9.5 de [68]), du à Powell [128].

[1] On peut aussi modifier légèrement l'algorithme en évitant l'étape 5.3 de RL lorsque j=1

Théorème 9.2.2 *Supposons que* $f : \mathbb{R}^n \mapsto \mathbb{R}$ *soit convexe* $C^{1,1}$ *dans un voisinage convexe de* $\{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$, *où* $x_0 \in \mathbb{R}^n$. *On considère l'algorithme de BFGS démarrant en* x_0 *avec une matrice* M_0 *symétrique définie positive et on suppose que celui-ci génère une suite* $\{x_j\}$ *telle que* $\{f(x_j)\}$ *soit bornée inférieurement. Alors,* $\liminf_{j\to\infty} ||g_j|| = 0$ *avec* $g_j = \nabla f(x_j)$.

La propriété suivante établit la convergence l'algorithme II.7 :

Proposition 9.2.3 *On considère l'algorithme II.7 de BFGS appliqué à la minimisation de la fonction duale régularisée* δ_r. *Dans cet algorithme, nous supposons que la valeur du paramètre d'augmentation* r_j *change un nombre fini de fois avec* j. *Alors,* $\liminf_{j\to\infty} ||\widetilde{g}_j|| = 0$ *avec* $\widetilde{g}_j = \nabla\delta_r(\lambda^j)$.

DÉMONSTRATION. La fonction $\delta_r : \mathbb{R}^{n_E} \times \mathbb{R}^{n_I} \mapsto \mathbb{R}$ est convexe $C^{1,1}$ et bornée inférieurement (existence d'une solution). De plus, la matrice $M_0 = r_0.I_{n_C}$ ($r_0 > 0$) utilisée dans notre algorithme II.7 est définie positive. Comme on suppose que le paramètre d'augmentation r_j change un nombre fini de fois avec j, il existe un indice k tel que $\forall j \geq k$ on as $r_{j+1} = r_j$. En d'autre termes lorsque $j \geq k$ l'algorithme II.7 est identique à l'algorithme classique de BFGS. Et alors, en appliquant le théorème 9.2.2 de convergence de la méthode de BFGS la propriété est bien démontrée. □

9.3 Résultats numériques et conclusions sur l'algorithme de BFGS appliqué à la minimisation de la fonction duale régularisée

Les figures 9.1, 9.2 et 9.3 ont été construites à partir de la résolution du problème F4 à la première itération de GN pour différentes valeurs du paramètre d'augmentation r ($r = 10^{-2}$, 1 et 10^2). Dans ces trois figures nous comparons l'évolution de l'admissibilité des contraintes obtenues en utilisant l'algorithme du LA (à paramètre d'augmentation fixé) à celle obtenues en utilisant l'algorithme de BFGS appliqué à la minimisation de la fonction duale régularisée (algorithme II.7). Nous remarquons que l'algorithme de BFGS est d'autant plus intéressant que la valeur de r est faible. Pour $r = 10^{-2}$ (figure 9.1), cet algorithme converge plus rapidement que l'algorithme classique du LA. Par contre, pour $r = 10^2$ (figure 9.3) l'algorithme classique du LA est meilleur.

Dans la figure 9.4, nous avons tracé l'évolution de l'admissibilité des contraintes au cours des itérations de LA pour le problème sur données réelles KIMASI. Nous remarquons dans cet exemple l'inefficacité de l'algorithme de BFGS comparé à l'algorithme classique du LA : après 50 itérations de LA, l'algorithme de BFGS n'arrive pas à rendre les contraintes admissibles. Ce qui n'est pas montré dans cette figure est que dès la troisième itération de Gauss-Newton cet algorithme rentre dans une phase de recherche linéaire et n'arrive plus à faire décroître suffisamment la fonction duale régularisée. Une explication de cet échec est que, proche de la solution, la fonction duale régularisée de ce

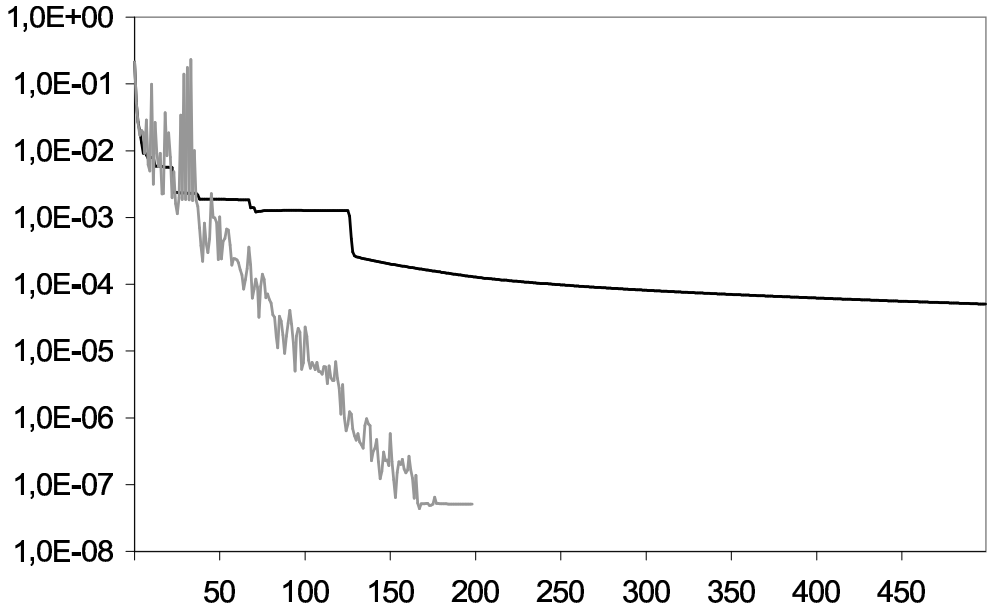

Fig. 9.1 Problème F4, admissibilité des contraintes au cours des itérations de LA pour $r = 10^{-2}$: algorithme de BFGS (courbe grise), algorithme des multiplicateurs (courbe noire)

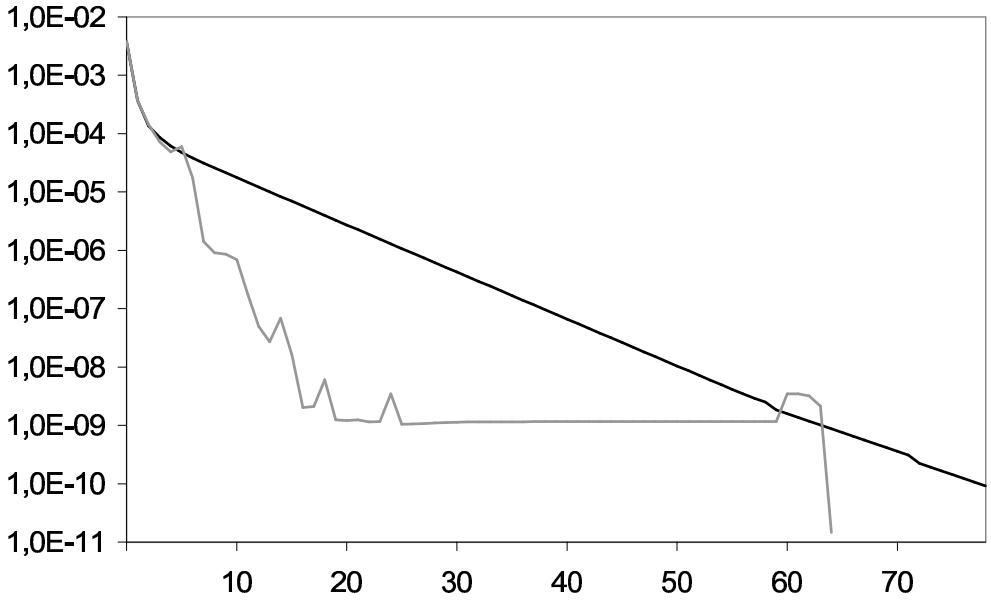

Fig. 9.2 Modèle F4, admissibilité des contraintes au cours des itérations de LA pour $r = 1$: algorithme de BFGS (courbe grise), algorithme des multiplicateurs (courbe noire)

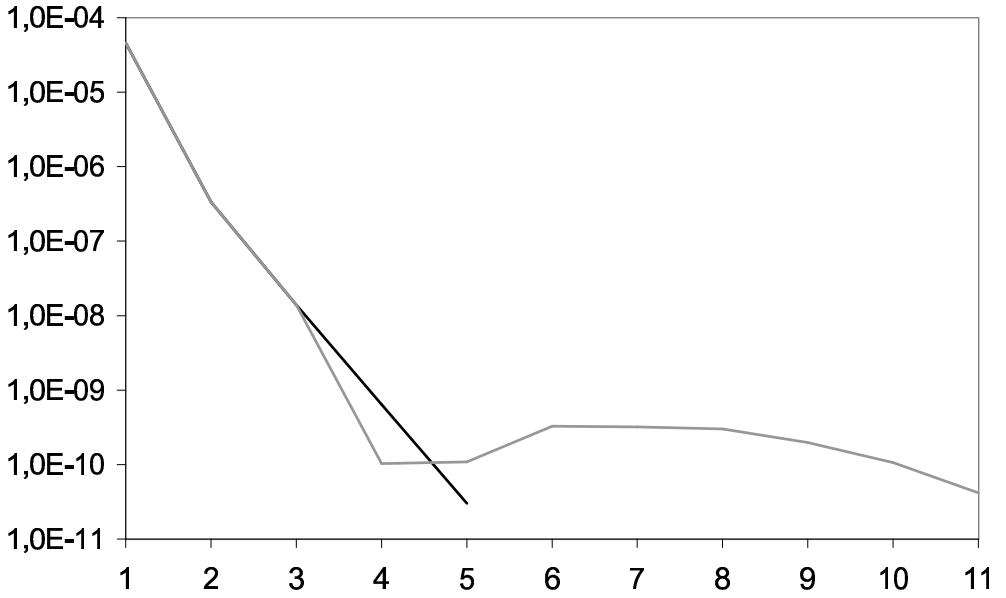

Fig. 9.3 Modèle F4, admissibilité des contraintes au cours des itérations de LA pour $r = 10^2$: algorithme de BFGS (courbe grise), algorithme des multiplicateurs (courbe noire)

problème est très “plate”. Cela rend les différences d’une itération à l’autre de la valeur de la fonction duale régularisée très faibles (plus de 10 chiffres après la virgule) et ce défaut de précision entraîne le mauvais fonctionnement de l’algorithme de BFGS.

Pour conclure sur l’algorithme de BFGS voici les arguments qui nous font préférer l’utilisation de l’algorithme du LA :

1. En général on peut résoudre les problèmes avec un paramètre d’augmentation r fort, ce qui entraîne une convergence très rapide de la méthode du LA (on converge en une dizaine d’itérations de LA). Or, compte tenu de l’initialisation dynamique de la matrice de BFGS, il faut souvent compter au moins une vingtaine d’itérations avant que l’algorithme de BFGS ne devienne efficace.
2. Lorsqu’on utilise l’algorithme de BFGS, on perd les propriétés de convergence globale et d’identification finie des contraintes actives de la méthode des multiplicateurs.
3. Si le pas unité de la RL n’est pas accepté, l’algorithme de BFGS génère la résolution supplémentaire de plusieurs problèmes de Lagrange (chaque $(PL)^j$ ayant un coût relativement élevé).

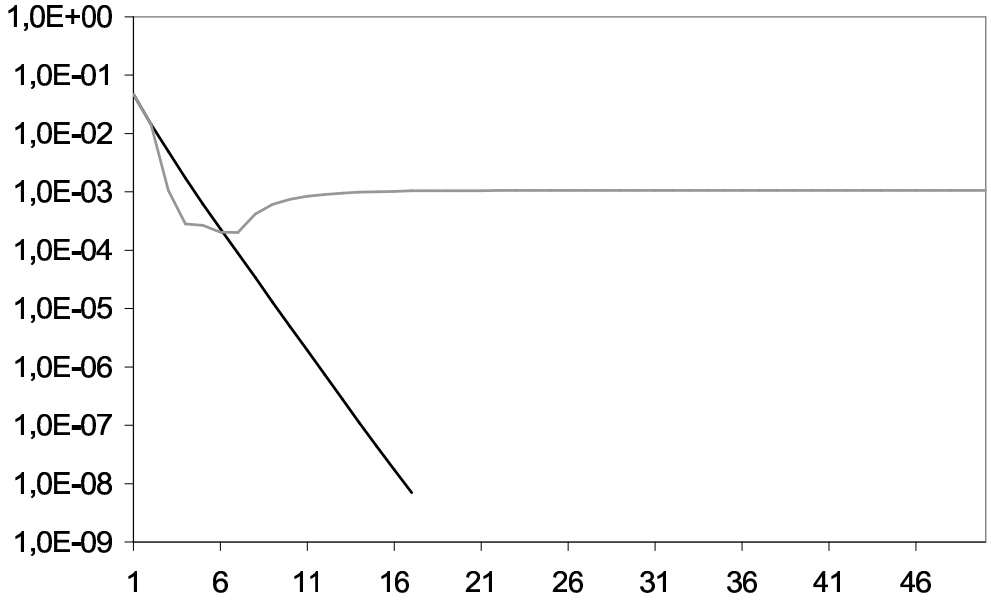

Fig. 9.4 Modèle KIMASI, admissibilité des contraintes au cours des itérations de LA pour $r = 1$: algorithme de BFGS (courbe grise), algorithme des multiplicateurs (courbe noire)

Chapitre 10

Mise en œuvre d'une stratégie automatique de l'inversion sous contraintes en tomographie de réflexion

10.1 Choix du paramètre d'augmentation : les problèmes rencontrés pour de trop petites ou grandes valeurs du paramètre d'augmentation

Nous avons déjà vu dans le chapitre 7 que le choix du paramètre d'augmentation est crucial pour l'efficacité de l'algorithme du LA (voir algorithme II.5). Deux pièges doivent être évités :

Remarques 10.1.1

1. *une trop grande valeur de r rend le problème de Lagrange (voir chapitre 8) mal conditionné et donc difficile à résoudre,*
2. *une trop petite valeur de r diminue la vitesse de convergence de l'algorithme du LA.*

Il est très difficile de déterminer une valeur a priori de ce paramètre. Ainsi, nous proposons une méthode qui met à jour la valeur de r_j au cours des itérations j de LA en fonction du comportement de l'algorithme lors des itérations précédentes. Dans les 2 sections suivantes, nous allons voir que l'on peut se donner une borne supérieure et une borne inférieure à la valeur de r_j.

10.1.1 Une borne supérieure pour r : conditionnement des problèmes de Lagrange

Comme on a vu à la section 8.2.3 que le conditionnement des problèmes de Lagrange se détériore lorsque la valeur de r est grande, ce qui induit une minimisation difficile

des problèmes de Lagrange par GP-AC-GC. Cependant, c'est le conditionnement des problèmes de Lagrange préconditionnés qui doit être regardé en priorité. En effet, un préconditionnement efficace des système linéaires résolus par GC entraîne une bonne minimisation des problèmes de Lagrange (il faut que le préconditionneur prenne bien en compte tous les termes correspondant aux contraintes : $P_{\mathcal{W}} \approx r(C_E^\top C_E + C_{\mathcal{W}}^\top C_{\mathcal{W}})$). Ainsi, le préconditionnement peut atténuer ou même faire disparaître l'effet négatif d'une grande valeur du paramètre d'augmentation r sur la minimisation des problèmes de Lagrange.

Les hessiens des problèmes quadratiques à minimiser par GC lors de la résolution d'un problème de Lagrange sont de la forme (voir section 8.2.3) :

$$H_{\mathcal{W}}(r) = H + r(C_E^\top C_E + C_{\mathcal{W}}^\top C_{\mathcal{W}})$$

pour les hessien non-préconditionnés, et

$$\widetilde{H}_{\mathcal{W}}(r) = Q^{-\top} H_{\mathcal{W}}(r) Q^{-1}$$

pour les hessiens préconditionnés, avec $Q \approx (H_{\mathcal{W}}(r))^{1/2}$ un préconditionneur de $H_{\mathcal{W}}(r)$. Un moyen simple et peu coûteux d'estimer la valeur du conditionnement consiste à utiliser les quotients de Rayleigh calculés au cours des itérations de GC préconditionnés. Ceux-ci donnent une borne inférieure du conditionnement en utilisant ces quotients (voir équation (1.16) de la section 1.3.2). Une estimation des valeurs propres minimales/maximales des matrices $H_{\mathcal{W}}(r)$ et $\widetilde{H}_{\mathcal{W}}(r)$ utilisées lors de la résolution du problème de Lagrange $(PL)^j$ peut-être calculée en prenant le min/max des quotients de Rayleigh rencontrés au cours des itérations i_{GC} de Gradient Conjugué Préconditionné :

$$\begin{aligned}
\lambda_{min_j}^{H_{\mathcal{W}}(r)} &:= \min_{i_{GC}} \frac{d_{i_{GC}}^\top H_{\mathcal{W}}(r) d_{i_{GC}}}{||d_{i_{GC}}||_2^2} \\
\lambda_{max_j}^{H_{\mathcal{W}}(r)} &:= \max_{i_{GC}} \frac{d_{i_{GC}}^\top H_{\mathcal{W}}(r) d_{i_{GC}}}{||d_{i_{GC}}||_2^2} \\
\lambda_{min_j}^{\widetilde{H}_{\mathcal{W}}(r)} &:= \min_{i_{GC}} \frac{d_{i_{GC}}^\top H_{\mathcal{W}}(r) d_{i_{GC}}}{||d_{i_{GC}}||_{P_{\mathcal{W}}(r)}^2} \\
\lambda_{max_j}^{\widetilde{H}_{\mathcal{W}}(r)} &:= \max_{i_{GC}} \frac{d_{i_{GC}}^\top H_{\mathcal{W}}(r) d_{i_{GC}}}{||d_{i_{GC}}||_{P_{\mathcal{W}}(r)}^2},
\end{aligned}$$

avec $||.||_{P_{\mathcal{W}}(r)}$ la norme associée à la matrice $P_{\mathcal{W}}$ et $d_{i_{GC}}$ les directions conjuguées. L'ensemble de travail $\mathcal{W}$ peut changer d'une itération de GC à l'autre, en fonction des bornes activées. On obtient donc finalement une estimation du conditionnement du problème de Lagrange $(PL)^j$ résolu par un gradient conjugué préconditionné ou non-préconditionné. On note

$$Cond_j := \frac{\lambda_{max_j}^{H_{\mathcal{W}}(r)}}{\lambda_{min_j}^{H_{\mathcal{W}}(r)}} \approx Cond((PL)^j)$$

l'estimation du conditionnement du problème de Lagrange $(PL)^j$ résolu par un GC non-préconditionné et

$$\widetilde{Cond}_j := \frac{\lambda_{min_j}^{\widetilde{H}_{\mathcal{W}}(r)}}{\lambda_{max_j}^{\widetilde{H}_{\mathcal{W}}(r)}} \approx Cond(\widetilde{(PL)^j})$$

l'estimation du conditionnement du problème de Lagrange $(PL)^j$ résolu par un GC préconditionné (la notation $\widetilde{(PL)^j}$ représente le problème de Lagrange $(PL)^j$ préconditionné).

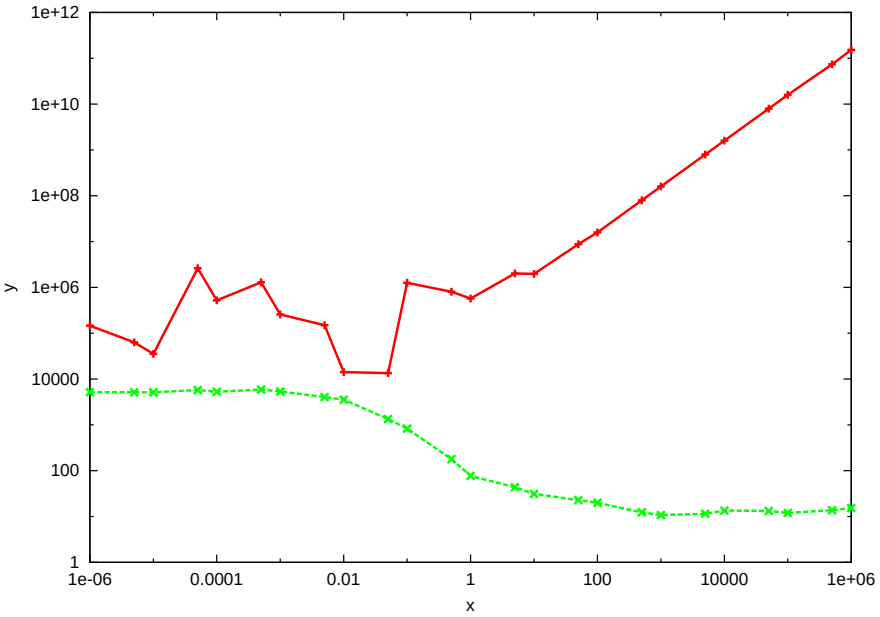

Fig. 10.1 Estimation du conditionnement d'un $(PL)^j$ en fonction de r : contraintes non couplées

Les figures 10.1 et 10.2 ont été obtenues en inversant respectivement l'exemple 3 (contraintes non couplées sur la profondeur) et l'exemple 2 (contraintes couplées sur la profondeur) du modèle de tomographie "KARINE" (cf. annexe D pour la description de ce modèle). Lors de cette inversion, nous nous intéressons plus particulièrement aux résultats concernant la résolution du premier $(PL)^j$ lors de la première itération de GN ($k = 1$ et $j = 1$). Dans ces 2 figures nous avons tracé l'évolution de l'estimation du conditionnement du premier problème de Lagrange (préconditionné ou non) en fonction du paramètre d'augmentation r du LA. On remarque que la proposition 8.2.4 est bien illustrée : dans les 2 figures, pour de grandes valeurs de r, le conditionnement estimé du $(PL)^j$ non-préconditionné augmente proportionnellement avec r (pour $r > 10$, les 2 courbes en trait plein sont croissantes, et elles augmentent proportionnellement avec r). Par contre le comportement du conditionnement estimé du $(PL)^j$ préconditionné diffère suivant la figure considérée. Cette différence de comportement s'explique par l'effet du préconditionneur sur la résolution de $(PL)^j$. Rappelons que le préconditionneur choisi pour résoudre le problème de Lagrange est un préconditionneur diagonal par blocs qui ne prend pas en compte les termes de couplage. Dans la figure 10.1, comme les contraintes d'interface ne sont pas couplées notre préconditionneur est efficace : pour $r > 10$ le conditionnement estimé du $(PL)^j$ préconditionné est faible (la courbe en pointillé est environ

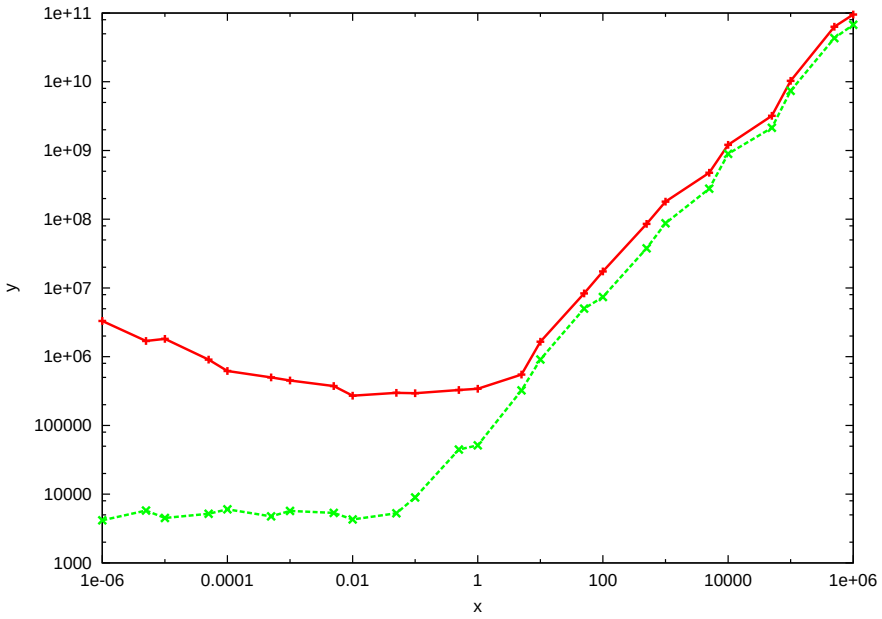

Fig. 10.2 Estimation du conditionnement d'un $(PL)^j$ en fonction de r : contraintes couplées

constamment égale à 10). A l'opposé, dans la figure 10.2, comme les contraintes d'interface sont couplées notre préconditionneur n'est pas efficace : pour $r > 10$ le conditionnement estimé du $(PL)^j$ préconditionné augmente proportionnellement avec r. A partir de ces observations et de la proposition 8.2.4 on peut tenter d'améliorer le conditionnement du problème de Lagrange en diminuant la valeur du paramètre d'augmentation.

Supposons que l'on n'arrive pas à résoudre correctement le problème de Lagrange $(PL)^j$, il est alors souhaitable de diminuer la valeur du paramètre d'augmentation afin d'améliorer le conditionnement de $(PL)^{j+1}$. Dans ces circonstance on peut utiliser une heuristique du type

$$r_{j+1} = \frac{r_j}{C},$$

où $C > 1$ est une constante. Reste à savoir ce que l'on doit faire lorsque le problème $(PL)^{j+1}$ est lui aussi impossible à résoudre correctement : doit-on en conclure que le problème de Lagrange est intrinsèquement mal conditionné et que quelque soit la valeur de r le solveur GP-AC-GC ne pourra pas trouver une solution au problème de Lagrange ? Ou doit-on en conclure que l'on peut encore diminuer la valeur de r pour améliorer le conditionnement ? C'est à ce niveau qu'intervient l'estimation $\widetilde{Cond}_j$ du conditionnement de $(PL)^j$. En effet, on peut donner une réponse à cette question en comparant les estimations du conditionnement de $(PL)^j$ et $(PL)^{j+1}$. Ainsi, si $\widetilde{Cond}_{j+1} < \widetilde{Cond}_j$ alors on en conclut que la diminution de r_j a été bénéfique et que l'on peut encore tenter d'améliorer le conditionnement en diminuant r_{j+1}. Dans le cas contraire, on conclut que le solveur

GP-AC-GC est en échec.

10.1.2 Une borne inférieure : taux de convergence de la norme des contraintes

On peut se donner une borne inférieure de r_{j+1} en regardant la valeur de

$$\rho_j := \frac{\nu_{j+1}}{\nu_j},$$

où ρ_j est le taux de convergence des contraintes à l'itération j de LA, et

$$\nu_j := ||(C_E \delta m^j - e, y^j - C_I \delta m^j)||$$

est la norme euclidienne des contraintes d'égalité de (7.3).

En effet, d'après le théorème 4.5 de l'article en annexe C, il existe une constante $L > 0$ telle que, quelque soit $j \geq 1$ on a :

$$\rho_j \leq \min\left(1, \frac{L}{r_j}\right).$$

Notons que cette formule est valable seulement si l'on résout le problème de Lagrange (voir problème (7.10)) de manière exacte. Nous allons nous servir de cette formule pour améliorer le taux de convergence de la norme des contraintes en jouant sur la valeur du paramètre d'augmentation.

Ainsi, si l'on se donne au départ de l'algorithme du LA une valeur $\rho_{des} \in (0, 1)$ de la convergence souhaitée sur la norme des contraintes on peut alors mettre à jour la valeur de r_{j+1} par la formule suivante :

$$r_{j+1} = r_j \max\left(1, \frac{\rho_j}{\rho_{des}}\right).$$

Si le taux de convergence actuel (ρ_j) est plus grand que le taux souhaité (ρ_{des}) la mise à jour précédente aura pour effet d'augmenter le paramètre d'augmentation. Dans le cas contraire on gardera la même valeur du paramètre d'augmentation ($r_{j+1} = r_j$). Cette heuristique devrait permettre de résoudre les problèmes de convergence de l'algorithme du LA lorsque le paramètre d'augmentation est trop petit (voir point 2. de la remarque 10.1.1).

10.2 Stratégie automatique du choix de r au cours des itérations de LA

Comme on l'a vu dans la section précédente, le choix du paramètre d'augmentation r est crucial pour assurer la convergence de l'algorithme II.5 du LA. Plus r est grand,

plus les multiplicateurs de Lagrange convergent rapidement vers une solution optimale duale. D'autre part, plus r est grand, plus le $(PL)^j$ est difficile à résoudre (cf. problème de conditionnement à la section 8.2.3). On cherche un algorithme capable d'adapter la valeur du paramètre d'augmentation r_j au cours des itérations j de LA.

On propose d'utiliser l'algorithme suivant pour régler automatiquement le paramètre d'augmentation r_j au cours des itérations de LA :

Data : Itération de lagrangien augmenté : : j.
Valeur du paramètre d'augmentation : : r_j.
Valeur maximale du paramètre d'augmentation : : r_{MAX}
(si $j = 0$, $r_{MAX} = 10^{15}$).
État de résolution du problème de Lagrange $(PL)^j$
(PL=true ou PL=false).
Conditionnement ressenti dans le GC pour résoudre $(PL)^j$: : $\widetilde{Cond}_j$.
Taux de convergence minimal : : ρ_{des} (si $j = 0$, $\rho_{des} = 10^{-3}$).

Pour $j \geq 1$:
Valeur antérieure du paramètre d'augmentation : : r_{j-1}.
Conditionnement ressenti dans le GC pour résoudre $(PL)^{j-1}$: : $\widetilde{Cond}_{j-1}$.
Taux de convergence de la norme des contraintes : : ρ_j.

begin
  if *(*$(r_j < r_{j-1})$ *ET (*$j \geq 1$*))* **then**
    DECREASE=true.
  else
    DECREASE=false.
  endif

  Initialisation du nouveau paramètre d'augmentation : : $r_{j+1} = r_j$.

  if *(*PL=false) **then**
    $\rho_{des} = 0.5$.
    if *(*DECREASE=false) **then**
      $r_{j+1} = r_j/10$.
    else
      if $\widetilde{Cond}_j < \widetilde{Cond}_{j-1}$ **then**
        $r_{j+1} = r_j/10$.
      else
        STOP : Impossible de résoudre $(PL)^{j+1}$ quelque soit la valeur choisie pour r_{j+1}.
      endif
    endif
  else
    if *(*DECREASE=false) **then**
      if *(*$(\rho_{des} < \rho_j)$ *ET (*$j \geq 1$*))* **then**
        $r_{j+1} = min(r_{MAX}, \frac{\rho_j}{\rho_{des}} r_j)$.
      endif
    else
      $r_{MAX} = 100 r_j$.
    endif
  endif
end

Algorithme II.8 Réglage automatique du paramètre d'augmentation r au cours des itérations de LA

Quelques remarques sur cet algorithme :

1. On dit qu'un problème de Lagrange est mal résolu (PL=false) lorsqu'il est impossible de satisfaire aux conditions d'optimalité de celui-ci en un nombre d'itérations de gradient conjugué donné. Dans le cas contraire on dit que le problème de Lagrange est bien résolu (PL=true).
2. La valeur du paramètre d'augmentation pour l'itération suivante de LA (r_{j+1}) ne peut être diminuée que si le problème de Lagrange $(PL)^j$ est mal résolu (PL=false). Dans le cas où les deux derniers problèmes de Lagrange ($(PL)^{j-1}$ et $(PL)^j$) ont été mal résolus, (PL=false et DECREASE=true), on vérifie que la diminution de r a provoqué une diminution effective du conditionnement ressentie par le GC ($\widetilde{Cond}_j < \widetilde{Cond}_{j-1}$). Si on ne constate aucune diminution effective du conditionnement ressentie par le GC ($\widetilde{Cond}_j \geq \widetilde{Cond}_{j-1}$), alors, cela implique qu'il est impossible pour le solveur GP-AC-GC de résoudre correctement le problème de Lagrange et ce quelque soit la valeur de r. Dans un tel cas, l'algorithme est en échec, et l'utilisateur peut alors tenter de relancer le solveur avec un préconditionneur du GC plus efficace. Dans tous les autres cas où le problème de Lagrange $(PL)^j$ est mal résolu (PL=false), la valeur de r est diminuée ($r_{j+1} = r_j/10$).
3. La valeur du paramètre d'augmentation pour l'itération suivante de LA (r_{j+1}) ne peut être augmentée que si les deux derniers problèmes de Lagrange ($(PL)^{j-1}$ et $(PL)^j$) ont été bien résolus (PL=true et DECREASE=false). Dans ce cas, on peut estimer la valeur de ρ_j : c'est une borne inférieure du taux de convergence courant de la norme des contraintes. Si ρ_j est plus grand que le taux de convergence minimum ρ_{des} désiré par l'utilisateur, alors on augmente la valeur du paramètre d'augmentation par la règle $r_{j+1} = \min(r_{MAX}, \frac{\rho_{des}}{\rho_j} r_j)$. On remarque que, si l'algorithme ne rencontre aucun problème de Lagrange mal résolu, cette règle est identique à la règle utilisée dans la proposition 5.1 de l'article en annexe C. En effet, si PL=true quelque soit $j \geq 0$ alors la valeur de r_{MAX} n'est jamais modifiée ($r_{MAX} = 10^{15}$ $\forall j \geq 0$) et la règle d'augmentation de r se simplifie par :

$$r_{j+1} = \min(10^{15}, \frac{\rho_{des}}{\rho_j} r_j) = \frac{\rho_{des}}{\rho_j} r_j.$$

4. La valeur de r_{MAX} est modifiée si l'algorithme rencontre au moins un problème de Lagrange mal résolu. Plus précisément, la valeur de r_{MAX} change si le problème de Lagrange courant $(PL)^j$ est bien résolu (PL=true) et si l'avant dernier problème de Lagrange a mal été résolu (DECREASE=true). Dans ce cas précis, on met à jour r_{MAX} par la règle $r_{MAX} = 100 r_j$. Ainsi, s'il advenait que, dans les itérations suivantes de LA, l'algorithme décidait d'augmenter la valeur de r (cf. remarque 2.), alors cette augmentation serait contrôlée par r_{MAX}.
5. Notons que si le problème de Lagrange est mal résolu (PL=false), il ne faut pas mettre à jour les multiplicateurs de Lagrange par la formule des multiplicateurs (voir étape ((2)) de l'algorithme II.5). En effet, rappellons que cette mise à jour des multiplicateurs n'est valable que si le problème de Lagrange est résolu de manière

exacte .

Troisième partie

Utilisation des contraintes en tomographie de réflexion

L'article qui va suivre a été soumis en 2004 dans le journal *Geophysical Journal International*. Cet article représente l'application de la méthode d'optimisation avec contraintes que nous avons longuement développée dans la partie II de ce mémoire. L'introduction de contraintes dans le processus d'inversion permet de prendre en compte des informations a priori provenant de logs de puits et de connaissances géologiques sur le milieu. Aprés un rappel des points clefs de notre méthode d'optimisation avec contraintes, cet article montre son efficacité en tomographie de réflexion sur deux jeux de données réelles (2D et 3D) : l'introduction des contraintes permet de réduire l'indétermination sur la solution des deux problèmes inverses. De plus, dans la deuxième application, nous verrons l'efficacité de l'algorithme II.8 permettant de fixer automatiquement le paramètre d'augmentation au cours des itérations de lagrangien augmenté (voir chapitre 10).

Geophys. J. Int. (2000) **142,** 000–000

Constrained optimization in seismic reflection tomography: an SQP augmented Lagrangian approach

F. Delbos[1], J. Ch. Gilbert[2], R. Glowinski[3], D. Sinoquet[1]

[1] *Institut Français du Pétrole, 1 & 4 avenue de Bois-Préau, 92852 Rueil-Malmaison, France*
[2] *Institut National de la Recherche en Informatique et en Automatique, BP 105, 78153 Le Chesnay Cedex, France*
[3] *University of Houston, 4800 Calhoun Rd, Houston, TX 77204-3476, USA*

Received 2004 July; in original form 2004 July

SUMMARY
Seismic reflection tomography is a method for determining a subsurface velocity model from the traveltimes of seismic waves reflecting on geological interfaces. From an optimization viewpoint, the problem consists in minimizing a nonlinear least-squares function measuring the mismatch between observed traveltimes and those calculated by ray tracing in this model. The introduction of a priori information on the model is crucial to reduce the under-determination. The contribution of this paper is to introduce a technique able to take into account geological a priori information in the reflection tomography problem expressed as constraints in the optimization problem. This technique is based on a Gauss-Newton sequential quadratic programming approach. At each Gauss-Newton step, a solution to a convex quadratic optimization problem subject to linear constraints is computed thanks to an augmented Lagrangian algorithm. Our choice for this optimization method is motivated and its original aspects are described. The efficiency of the method is demonstrated on a 2D OBC real data set and on a 3D real data set: the introduction of constraints coming both from well logs and from geological knowledge allows us to reduce the under-determination of both inverse problems.

Key words: seismic reflection tomography, ray tracing, a priori information, least-squares approach, constrained optimization, SQP algorithm, augmented Lagrangian.

1 INTRODUCTION

Geophysical methods for imaging a complex geological subsurface in petroleum exploration requires the determination of an accurate wave propagation velocity model. Seismic reflection tomography turns out to be an efficient method for doing this: it determines the seismic velocity distribution from the traveltimes associated with the seismic waves reflecting on geological surfaces. This inverse problem requires the solution to the specific forward problem, which consists in computing these traveltimes for a given subsurface model by a ray tracing method (based on a high frequency approximation of the wave equation, see Červený (1987) and Jurado et al. (1998)). The inverse problem is formulated as the minimization of the least-squares function that measures the mismatch between traveltimes calculated by ray tracing and the observed traveltimes.

The main interests of reflection tomography are

- its flexibility for handling various types of traveltime data simultaneously (primary reflections but also multiple reflections, traveltimes associated with converted waves -PS data-, surface seismic, well seismic), provided that the ray tracing allows the computation of such data,
- the low computational time of the forward operator, in comparison with the time needed for the calculation of the wave equation solutions,
- the reduced number of local minima of the inverse problem, in comparison with the seismic inversion based on the wave equation simulation,
- its ability to integrate a priori geological information (via a least-squares formulation).

This method has been successfully applied to numerous real data sets (Ehinger et al. (2001) and Broto et al. (2003)). Nevertheless, the under-determination of the inverse problem generally requires the introduction of additional information to reduce the number of admissible models. Penalty terms modeling this information (as mentioned previously) can be added to the seismic terms in the objective function but the tuning of the penalty weights may be tricky. The standard methodology to invert complex subsurface structures (model composed of several velocity fields and reflectors), a top-down layer-stripping approach, may be inade-

quate. This approach consists in inverting separately each velocity layers (with its associated reflectors) starting from the upper layer to the lower one. To limit bad data fitting for deeper layers, a global inversion approach, which consists in simultaneously inverting all the velocity layers and interfaces of the subsurface model, is recommended. But, this method is often discarded due to its convergence troubles: because of the underlying under-determination, a global inversion of complex subsurface structures often leads to a bad subsurface model on which the ray tracing method fails to compute the traveltimes. Additional constraints on the model (for instance, on layer thicknesses to avoid non-physical interface intersections) are necessary to avoid those non admissible models.

We believe that the possibility to introduce constraints in the optimization process can overcome part of those difficulties. Equality and inequality constraints can indeed model many different types of a priori information. An optimization approach that can face these constraints efficiently will then discharge the final user of the inversion seismic software from the cumbersome task of tuning the weights associated with the additional penalty terms in the objective function.

The goals of this paper are twofold. The first one is to present our constrained nonlinear optimization method and to motivate its appropriateness to constrained reflection tomography. The second objective is to show the contribution of constraints in reflection tomography and the numerical efficiency of the constrained optimization method thanks to its application on a 2D PP/PS real data set and then on a 3D PP real data set.

We recall the problem of interest and introduce the notation in Section 2. In Section 3, our SQP augmented Lagrangian approach for constrained reflection tomography problems is described and motivated. Numerical experiments on real data sets are detailed and analyzed in Section 4. We conclude with Section 5.

2 THE SEISMIC REFLECTION TOMOGRAPHY PROBLEM

2.1 The unconstrained problem

Let us first recall the problem of interest and introduce the notation. The choice of the model representation is crucial for the efficiency of the methods used to solve the forward and inverse problems. Lailly & Sinoquet (1996) have discussed the interest of different types of velocity models. We have chosen here a *blocky* model, where the velocity distribution is described by slowly varying layer velocities (or velocity blocks) delimited by interfaces (see Figure 1). With this representation, we introduce explicitly a strong a priori information: the number of layers. The number of parameters describing the velocity variations is limited thanks to the explicit introduction of velocity discontinuities (the velocity within a layer varies smoothly). The model is thus composed of two kinds of parameters: those describing the velocity variations within the layers and those describing the geometry of the interfaces delimiting the layers. Parameters describing the velocity anisotropy can also be included (see Jurado et al. (1998) and Stopin (2001) for more details).

The i^{th} interface is represented by a cubic B-spline function (de Boor (1978) and Inoue (1986)) $\widehat{z}_i(x,y)$, whose coefficients define a vector z_i (x and y are the horizontal coordinates). Similarly, the i^{th} velocity field is represented by a cubic B-spline function $\widehat{v}_i(x,y,z)$ or $\widehat{v}_i(x,y) + k\,z$ with known scalar k (z is the vertical coordinate); the vector v_i contains the velocity coefficients. For n_v layer velocities and n_z interfaces, we collect the coefficients $v_1, \ldots, v_{n_v}$ in one vector v and the coefficients $z_1, \ldots, z_{n_z}$ in one vector z. The *model* vector $m \in \mathbb{R}^n$ is defined here as $m = (v, z)$. The C^2 smoothness of the functions describing the model allows the exact computation of derivatives of the traveltimes with respect to the model, quantities useful for ray tracing and tomography.

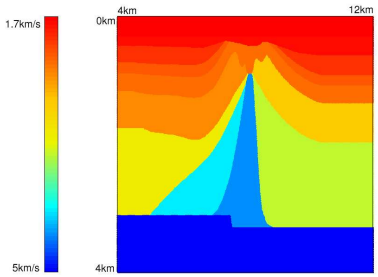

Figure 1. An example of a *blocky* subsurface velocity model.

Given a model m and an acquisition survey (locations of the sources and receivers) a vector of traveltimes $T(m)$ of seismic reflected waves can be computed by ray tracing (see Červený (1987) and Jurado et al. (1998)). The mapping $T : \mathbb{R}^n \to \mathbb{R} : m \mapsto T(m)$ is nonlinear. We assume that it is differentiable. In practice, this assumption may not be satisfied, in particular, the forward operator may even not be defined when rays escape from the region of interest or when a layer thickness vanishes (non-physical interface intersections as mentioned in the introduction).

Reflection traveltime tomography is the corresponding inverse problem: its purpose is to adjust m so that $T(m)$ best matches a vector of traveltimes $T^{obs} \in \mathbb{R}^n$ (the observed traveltimes) picked on seismic data. Since Gauss (1809), it is both classical and natural to formulate this problem as a least-squares one:

$$\min_{m \in \mathbb{R}^n} \ \frac{1}{2} \left\| T(m) - T^{obs} \right\|^2, \tag{1}$$

where $\|\cdot\|$ denotes the Euclidean norm.

The fact that problem (1) may be ill-posed has been pointed out by many authors (see for instance Delprat-Jannaud & Lailly (1993), Bube & Meadows (1999)). To ensure well-posedness, a curvature regularization is often introduced (Tikhonov & Arsenin (1977)). We use the sum of the squared L^2-norms of all the second order partial derivatives of every velocity $\widehat{v}_i$ and reflector $\widehat{z}_i$ (see for instance Delprat-Jannaud & Lailly (1993)). Such a regularization term can be written as $m^\top R\, m$, where R is a symmetric positive semidef-

inite matrix that only depends on the B-spline basis functions (it is independent of m). Thus, instead of problem (1), we consider the regularized least-squares problem

$$\min_{m\in\mathbb{R}^n} \Big(f(m) := \frac{1}{2} \left\| T(m) - T^{obs} \right\|^2 + \frac{\sigma}{2}\, m^\top R\, m \Big), \quad (2)$$

where the regularization weight σ is positive, and $f : \mathbb{R}^n \to \mathbb{R}$ is called the *cost function* (or *objective function*). The choice of the parameter σ is a difficult task. In practice, we use the L-curve method (see Hansen (1992)), also called the continuation method (Bube & Langan (1994)): starting from a large regularization weight, we decrease it regularly to retrieve more and more varying models until the data are fitted with the expected accuracy. The solution model is thus the smoothest model that fits the data up to a certain precision. This methodology allows us to do stable inversions. In the sequel, when we consider the objective function of problem (2), we assume that its regularization weight is fixed.

The unconstrained minimization problem of seismic reflection tomography, defined by (2), has the following features:

- the size of the data space and of the model space can be quite large (up to 10^6 traveltimes and 10^5 unknowns),
- the problem is ill-conditioned (Chauvier et al. (2000) have observed that the condition number of the approximated Hessian H_k given by equation (5) below can go up to 10^9 for a matrix of order 500),
- a forward simulation, i.e., a computation of $T(m)$, is CPU time consuming because of the large number of source-receiver pairs,
- the traveltime operator T is nonlinear, revealing the complexity of wave propagation in the subsurface.

To minimize efficiently a function like f in (2), it is highly desirable to have its gradient available. In the present context, thanks to Fermat's principle (Bishop et al. (1985)), it is inexpensive to compute row by row the Jacobian matrix $J(m)$ of T at m. Recall that its (i,j)th element is the partial derivative

$$J(m)_{ij} = \frac{\partial T_i}{\partial m_j}. \quad (3)$$

The gradient of the objective is then obtained by the formula $\nabla f(m) = J(m)^\top (T(m) - T^{obs}) + \sigma R m$. It is also natural to ask whether one can compute the second derivatives of f. The answer is however negative. Therefore, using a pure Newton method to solve (2) is hopeless.

There are at least two classes of methods that can take advantage of the sole first order derivatives:

- quasi-Newton (QN) methods,
- Gauss-Newton (GN) methods.

Standard QN methods do not use the structure of the least-squares problems, but have a larger scope of application. They are used for solving least-squares problems when the computation of the Jacobian matrix J is much more time consuming than the computation of the *residual* vector $T(m) - T^{obs}$ (see Courtier & Talagrand (1987) and Courtier et al. (1994) for a typical example in meteorology). We have mentioned above that this is not our case. On the other hand, the GN methods fully take benefit of the Jacobian matrix J by taking $J^\top J$ as an approximation of the Hessian of the first term in f. This algorithm can exhibit slow convergence when the residual is large at the solution and when T is strongly nonlinear at the solution. This does not seem to be the case in the problem we consider. Sometimes GN and QN methods are combined to improve the approximation of the Hessian of the first part of f by $J^\top J$ (see Dennis et al. (1981), Yabe & Yamaki (1995) and the references therein).

The above discussion motivates our choice of a classical line-search GN method to solve the unconstrained optimization problem (Chauvier et al. (2000)). The kth iteration, $k \geq 0$, proceeds as follows. Let m_k be the approximate solution known at the beginning of the iteration. Note

$$T_k := T(m_k) \quad \text{and} \quad J_k := J(m_k). \quad (4)$$

First, an approximate solution d_k to the following tangent quadratic problem is computed

$$\min_{d\in\mathbb{R}^n} \frac{1}{2} \left\| J_k d + T_k - T^{obs} \right\|^2 + \frac{\sigma}{2}\, (m_k + d)^\top R (m_k + d).$$

This is the quadratic approximation of f about m_k, in which the costly computation of the second derivatives of T has been neglected. By the choice of the positive semi-definite regularization matrix R, the Hessian of this quadratic function in d, namely

$$H_k := J_k^\top J_k + \sigma R, \quad (5)$$

is usually positive definite. Therefore, it makes sense to minimize the above quadratic function by a preconditioned conjugate gradient algorithm. The next model estimation is then obtained by the formula

$$m_{k+1} = m_k + \alpha_k d_k,$$

where $\alpha_k > 0$ is a step-size computed by a line-search technique ensuring a sufficient decrease of f at each iteration.

This method is generally able to solve the minimization problem (2). In some hard cases, however, the line-search technique fails to force convergence of the sequence $\{m_k\}_{k\geq 0}$ to a solution. This difficulty may arise when the Hessian of f is very ill-conditioned and can often be overcome by using trust regions (see Conn et al. (2000)) instead of line-searches. The former method usually provides more stable and accurate results than the latter (Delbos et al. (2001), see also Sebudandi & Toint (1993)). In any case, we observe in practice that very few iterations are needed to get convergence, typically of the order of 10.

2.2 Formulation of the constrained problem

Let us now set down the formulation of the constrained seismic tomography problem. The constraints that can be introduced in the optimization problem could be nonlinear (for example we could force the impact points of some rays on a given interface to be located in a particular area) but, in this study, we limit ourselves to *linear* constraints. Even though linearity brings algorithmic simplifications, the optimization problem is difficult to solve because of the large number (up to 10^4) and the variety of the constraints. These may be of various types:

- constraints of different physical natures: on the velocities, on the interface depths, or on the derivatives of these quantities,
- equality or inequality constraints (examples: fixed value of the velocity gradient, minimal depth of an interface),

- local or global constraints (examples: local information coming from a well, interface slope in a particular region).

The constrained reflection tomography problem we consider is formulated as the regularized least-squares problem (2) subject to linear constraints:

$$\begin{cases} \min\limits_{m\in\mathbb{R}^n} f(m) \\ C_E m = e \\ l \le C_I m \le u. \end{cases} \tag{6}$$

In this problem, C_E (resp. C_I) is an $n_E \times n$ (resp. $n_I \times n$) matrix, $e \in \mathbb{R}^{n_E}$, and l, $u \in \mathbb{R}^{n_I}$. We note

$$n_C := n_E + n_I \quad \text{and} \quad n'_C := n_E + 2n_I.$$

It is said that an inequality constraint is *active* at m if it is satisfied with equality for this m. Determining which of the inequality constraints of (6) are active at a solution among the 3^{n_I} possibilities turns out to be a major difficulty for the algorithms.

2.3 First order optimality conditions

Let $\widehat{m}$ be a local solution to (6). Since f is assumed to be differentiable and the constraints are linear (thus qualified), there exist $\widehat{\mu}_E \in \mathbb{R}^{n_E}$, $\widehat{\mu}_l \in \mathbb{R}^{n_I}$, and $\widehat{\mu}_u \in \mathbb{R}^{n_I}$ such that the following Karush, Kuhn, and Tucker conditions (KKT) hold

$$\begin{cases} (a)\ \nabla f(\widehat{m}) + C_E^\top \widehat{\mu}_E + C_I^\top(\widehat{\mu}_u - \widehat{\mu}_l) = 0 \\ (b)\ C_E\widehat{m} = e, \quad l \le C_I\widehat{m} \le u \\ (c)\ (\widehat{\mu}_l, \widehat{\mu}_u) \ge 0 \\ (d)\ \widehat{\mu}_l^\top(C_I\widehat{m} - l) = 0, \quad \widehat{\mu}_u^\top(C_I\widehat{m} - u) = 0. \end{cases} \tag{7}$$

See for example Fletcher (1987) or Bertsekas (1995) for a presentation of the optimality theory of smooth problems. The vectors $\widehat{\mu}_E$, $\widehat{\mu}_l$, and $\widehat{\mu}_u$ are called the Lagrange or KKT multipliers, and are associated with the equality and inequality constraints of (6). Condition (a) can also be written

$$\nabla_m \ell(\widehat{m}, \widehat{\mu}) = 0,$$

where $\widehat{\mu} := (\widehat{\mu}_E, \widehat{\mu}_l, \widehat{\mu}_u)$ and the function $\ell : \mathbb{R}^n \times \mathbb{R}^{n'_C} \mapsto \mathbb{R}$, called the *Lagrangian* of problem (6), is defined by

$$\begin{aligned} \ell(m, \mu) = f(m) &+ \mu_E^\top(C_E m - e) \\ &- \mu_l^\top(C_I m - l) + \mu_u^\top(C_I m - u). \end{aligned} \tag{8}$$

We note $\widehat{\mu}_I := \widehat{\mu}_l - \widehat{\mu}_u$.

Equations (d) in (7) are known as the *complementarity conditions*. They express the fact that a multiplier $\widehat{\mu}_i$ associated with an inactive inequality constraint vanishes. For some problems, the converse is also true: active inequality constraints have positive multipliers. It is then said that *strict complementarity* holds at the solution:

$$\begin{aligned} l_i < C_i\widehat{m} \quad &\Longleftrightarrow \quad (\widehat{\mu}_l)_i = 0 \\ C_i\widehat{m} < u_i \quad &\Longleftrightarrow \quad (\widehat{\mu}_u)_i = 0. \end{aligned}$$

3 SOLVING THE CONSTRAINED SEISMIC REFLECTION TOMOGRAPHY PROBLEM

In this section we motivate and describe the optimization method used to solve (6) or its optimality conditions (7).

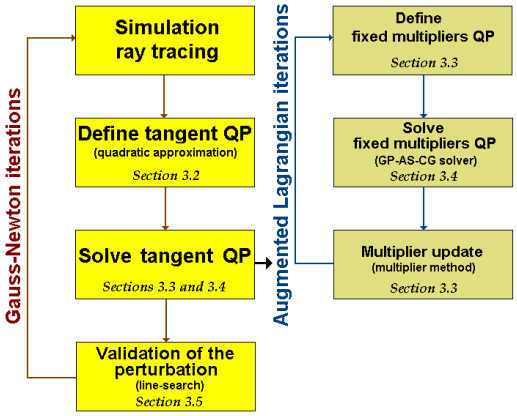

Figure 2. Operating diagram of the constrained optimization method.

A more detailed description is given by Delbos (2004). The operating diagram of the overall algorithm is presented in Figure 2 and can help the reader to follow the different levels of the approach.

3.1 Motivation for the chosen algorithmic approach

Presently, numerical methods to solve a nonlinear optimization problem like (6) can by gathered into two classes:

- the class of penalty methods, which includes the augmented Lagrangian approaches and the interior point (IP) approaches,
- the class of direct Newtonian methods, which is mainly formed of the sequential quadratic programming (SQP) approach.

Often, actual algorithms combine elements of the two classes, but their main features make them belonging to one of them.

In penalty methods, one minimizes a sequence of nonlinear functions, obtained by adding to the cost function in (6) terms penalizing more or less strongly the equality and/or inequality constraints as the iterations progress. For example, in the IP approaches, the inequality constraints are penalized in order to get rid of their combinatorial aspect, while the equality constraints are maintained, since they are easier to handle (see for instance Byrd et al. (2000) and the references therein). The iterations minimizing approximately each penalty function are usually based on the Newton iteration, which requires finding the solution to a linear system. Therefore the overall work of the optimization routine can be viewed as the one of solving a "sequence of sequences" of linear systems. This simplified presentation is mainly valid far from a solution, since close to a regular solution, a single Newton step is often enough to minimize sufficiently the penalty function (see Gould et al. (2000) for example). Now, each time a step is computed as a solution to a linear system, the nonlinear functions defining the optimization problem have to be evaluated in order to estimate the quality of the step. It is sensible to define an iteration of

the penalty approach as formed of a step computation and an evaluation of the nonlinear functions.

On the other hand, an SQP-like algorithm is a Newtonian method applied to the optimality conditions, equations (7) in our case (see part III in Bonnans et al. (2003) for example). Therefore, there is no sequence of nonlinear optimization problems to solve approximately, like in penalty methods. As a result, it is likely that such an approach will need less iterations to achieve convergence. Since, here also, the nonlinear functions defining the optimization problem need to be computed at each iteration to validate the step, it is likely that less nonlinear function evaluations are required with an SQP approach, in comparison with a penalty approach. Nevertheless, each SQP iteration is more complex, since it requires solving a *quadratic program* (QP), which is an optimization problem, with a quadratic cost function and linear equality and inequality constraints.

The discussion above shows that the choice of the class of algorithms strongly depends on the features of the optimization problem to solve. The key issue is to balance the time spent in the simulator (to evaluate the functions defining the nonlinear optimization problem) and in the optimization procedure (to solve the linear systems or the quadratic programs). In the unconstrained seismic reflection tomography problem, we have said in Section 2.1 that most of the CPU time is spent in the evaluation of the traveltimes (simulation) and that the Gauss-Newton algorithm converges in very few iterations (around 10). When choosing the algorithmic approach for the constrained version of the problem, we anticipated that the number of iterations will also be small with a Newton-like algorithm and that, in the current state of their development, IP algorithms will be unlikely to converge in so few iterations. This is our main motivation for developing an SQP-like algorithm: to keep as small as possible the number of nonlinear function evaluations. This strategy could be questioned with a ray tracing using massive parallelization; we leave this topic for future investigations.

3.2 A Gauss-Newton sequential quadratic programming approach

We have already mentioned that the sequential quadratic programming (SQP) algorithm is a Newton-like method applied to the optimality conditions of the nonlinear optimization problem under consideration, equations (7) in our case. In its standard form, it then benefits from a local quadratic convergence. Let us specify this algorithm for the constrained seismic reflection tomography problem.

The main work at the kth iteration of an SQP algorithm consists in solving the following *tangent quadratic program* (tangent QP) in d (see Chapter 13 and equation (13.4) in Bonnans et al. (2003)), in order to find the perturbation d_k to be given to m_k:

$$(\mathrm{QP}_k) \quad \left\{ \begin{array}{l} \min\limits_{d\in\mathbb{R}^n} \left(F_k(d) := g_k^\top d + \dfrac{1}{2}\, d^\top H_k d \right) \\ C_E d = \tilde{e}_k \\ \tilde{l}_k \le C_I d \le \tilde{u}_k. \end{array} \right. \tag{9}$$

The cost function of this problem has a hybrid nature. Its linear part is obtained by using the gradient of the cost function of (6) at m_k, which is

$$g_k = J_k^\top (T_k - T^{obs}) + \sigma\, R\, m_k,$$

where we used the notation in (4). Its quadratic part should use, ideally, the Hessian of the Lagrangian ℓ defined in (8), in order to get (quadratic) convergence. Since the constraints in (6) are linear, only the Hessian of f plays a role. Like for the unconstrained problem, we select the Gauss-Newton approximation H_k (see (5)) of this Hessian in order to avoid the computation of the expensive second derivatives of T. The constraints of (9) are obtained by the linearization of the constraints of (6) at m_k. Writing $C_E(m_k + d) = e$ and $l \le C_I(m_k + d) \le u$ leads to the constraints of (9) with

$$\tilde{e}_k := e - C_E m_k,\ \ \tilde{l}_k := l - C_I m_k,\ \text{ and }\ \tilde{u}_k := u - C_I m_k.$$

Let d_k be a solution to (QP_k). Near a solution, the SQP algorithm updates the primal variables m_k by

$$m_{k+1} := m_k + d_k. \tag{10}$$

The SQP algorithm is actually a primal-dual method, since it also generates a sequence of multipliers $\{\mu_k\} \in \mathbb{R}^{n'_C}$, which aims at approximating the optimal multiplier associated with the constraints of (6). These multipliers are updated by

$$\mu_{k+1} = \mu_k^{\mathrm{QP}}, \tag{11}$$

where μ_k^{QP} is the triple formed of the multipliers $(\mu_k^{\mathrm{QP}})_E$, $(\mu_k^{\mathrm{QP}})_l$, and $(\mu_k^{\mathrm{QP}})_u$, associated with the equality and inequality constraints of (QP_k). Because of the linearity of the constraints, these multipliers do not intervene in the tangent QP, but they are useful for testing optimality and for the globalization of the algorithm (Section 3.5).

The following property of the SQP algorithm deserves being quoted for the discussion in Section 3.3. When strict complementarity holds at a "regular" primal-dual solution $(\widehat{m}, \widehat{\mu})$ to (6) (see Section 2.3) and when the current iterate (m_k, μ_k) is close enough to $(\widehat{m}, \widehat{\mu})$, the active constraints of the tangent QP are those that are active at the solution $\widehat{m}$ (see Theorem 13.2 in Bonnans et al. (2003)). Therefore, the difficult task of determining which constraints are active at the solution to (QP_k) disappears once m_k is close to $\widehat{m}$, since the constraint activity is unchanged from one tangent QP to the next one.

The technique used to solve the tangent QP is essential for the efficiency of the SQP approach. In particular, because of the property mentioned in the previous paragraph, it should take benefit of an a priori knowledge of the active constraints. In the next two sections, we concentrate on this topic, which is represented by the right hand side blocks in Figure 2. We will come back to the globalization of SQP (the part of the algorithm that is depicted by the bottom block in the left hand side of Figure 2) in Section 3.5.

3.3 Solving the tangential quadratic problem by an augmented Lagrangian method

Because H_k is positive semidefinite (and usually positive definite), the tangent QP problem (9) is convex. Such a problem has been the subject of many algorithmic studies; we mention the following techniques:

- active set (AS),
- augmented Lagrangian (AL),
- interior points (IP).

Let us now motivate our choice of developing an AL algorithm to solve the QP in (9). The AS approach is often used to solve the QP's in the SQP algorithm. It has the advantage of being well defined, even when the problem is non-convex, and of being able to take advantage of an a priori knowledge of the active constraints at the solution. However, since this algorithm updates the active set one constraint at a time, it suffers from being rather slow when the active set is not correctly initialized and when there are many inequality constraints. For large problems, this can be a serious drawback and we have discarded this method for that reason. The IP algorithms are very efficient to solve convex QP's but, presently, they have some difficulty in taking benefit of a good knowledge of the active constraints as this is often the case after a few iterations of the SQP algorithm. On the other hand, the inherent ill-conditioning of the linear systems they generate and the necessity to use here iterative methods to solve them have appeared to us as deterrent factors.

The AL approach that we have implemented goes back to Hestenes (1969) and Powell (1969). This is a well established methodology, designed to solve nonlinear optimization problems, although its properties for minimizing a convex QP does not seem to have been fully explored (see Delbos & Gilbert (2005)). It is adapted to large problems, since it can be implemented in such a way that it does not need any matrix factorization (Fortin & Glowinski (1983) and Glowinski & Le Tallec (1989)). In the context of seismic tomography problems, a version of the AL algorithm has been proposed by Glowinski & Tran (1993) to solve the tangent QP of the SQP method. The present contribution takes inspiration from that paper and goes further by improving the efficiency of its augmented Lagrangian QP solver. Let us detail the approach.

The QP (9) is first written in an equivalent form, using an auxiliary variable $y \in \mathbb{R}^{n_I}$ (we drop the index k for simplicity):

$$(\mathrm{QP}') \quad \left\{ \begin{array}{l} \min\limits_{(d,y)\in\mathbb{R}^n\times\mathbb{R}^{n_I}} F(d) \\ C_E d = \tilde{e}, \quad C_I d = y \\ \tilde{l} \le y \le \tilde{u}. \end{array} \right.$$

Next, the AL associated with the equality constraints of that problem is considered. This is the function $\mathcal{L}_r : \mathbb{R}^n \times \mathbb{R}^{n_I} \times \mathbb{R}^{n_C} \mapsto \mathbb{R}$ defined at $d \in \mathbb{R}^n$, $y \in \mathbb{R}^{n_I}$, and $\lambda = (\lambda_E, \lambda_I) \in \mathbb{R}^{n_C}$ by

$$\begin{aligned} \mathcal{L}_r(d,y,\lambda) = F(d) + \lambda_E^\top (C_E d - \tilde{e}) + \frac{r}{2} \| C_E d - \tilde{e} \|^2 \\ + \lambda_I^\top (C_I d - y) + \frac{r}{2} \| C_I d - y \|^2 . \end{aligned}$$

The scalar $r > 0$ is known as the *augmentation parameter*.

The precise statement of our version of the AL algorithm to solve (9) or (QP′) can now be given: see Algorithm 1. This method, in particular the update of the multipliers by (13), has a nice interpretation in terms of the proximal point algorithm in the dual space (see Rockafellar (1973)). It is not essential to give this interpretation here, but this one is very useful for proving the properties of the method, including its convergence. In this theory, the augmentation parameter r in the definition of $\mathcal{L}_r$ is viewed as a step-size, damping the modification of the multipliers in (13). The factor of $r = r^j$ in this formula is indeed a negative subgradient of the dual function at λ^{j+1}. Note that these step-sizes r^j can now change at each iteration of Algorithm 1, while the penalty approach behind the definition of $\mathcal{L}_r$ makes the possibility of such a change less natural. On the other hand, it can be shown that the algorithm converges if (9) has a solution and if the sequence $\{r^j\}_{j\ge 0}$ is chosen bounded away from zero. If, in addition, r^j is chosen larger than some positive Lipschitz constant L (usually unknown, unfortunately), the norm of the equality constraints converges globally linearly to zero: this is inequality (14) below, on which we will come back. Actually, the larger are the augmentation parameters r^j, the faster is the convergence. The only limitation on a large value for r^j comes from the ill-conditioning that such a value induces in the AL and the resulting difficulty or impossibility to solve (12). This is why a test for updating the multipliers by (13) has been introduced. For ensuring convergence, the test prevents the multipliers from being updated when (12) is not correctly solved (a heuristic less restrictive than this test is used by Delbos (2004)).

Algorithm 1. An AL algorithm to solve (9).

data: $\lambda^0 \in \mathbb{R}^{n_C}$ and $r^0 > 0$;
begin
  for $j = 0, 1, 2, \ldots$ **do**
    if $(C_E d^j \simeq \tilde{e})$ & $(C_I d^j \simeq y^j)$ **then return**;
    by Algorithm 3, find a solution (d^{j+1}, y^{j+1}) to

$$\left\{ \begin{array}{l} \min_{(d,y)} \ \mathcal{L}_{r^j}(d, y, \lambda^j) \\ \tilde{l} \le y \le \tilde{u}; \end{array} \right. \tag{12}$$

    if (12) is solved **then**

$$\left\{ \begin{array}{l} \lambda_E^{j+1} := \lambda_E^j + r^j (C_E d^{j+1} - \tilde{e}) \\ \lambda_I^{j+1} := \lambda_I^j + r^j (C_I d^{j+1} - y^{j+1}) \end{array} \right. \tag{13}$$

    else
      $\lambda^{j+1} := \lambda^j$;
    end
    choose a new augmentation parameter $r^{j+1} > 0$ by Algorithm 2;
  end
end

Algorithm 1 is a dual method, since it essentially monitors the dual sequence $\{\lambda^j\}_{j\ge 0}$; the augmentation parameters r^j and the primal variables (d^{j+1}, y^{j+1}) are viewed as auxiliary quantities. The choice of the initial multiplier λ^0 depends on the outer SQP iteration index. When $k = 0$, Algorithm 1 simply takes $\lambda^0 = 0$, unless an estimate of the optimal multiplier is provided. When $k > 0$, Algorithm 1 takes for λ^0 the dual solution to the previous QP. A similar strategy is used for setting r^0: when $k = 0$, r^0 is set to an arbitrary value (the effect of taking $r^0 = 1$ or $r^0 = 10^4$ is tested for the 3D real data set in Section 4.2) and, when $k > 0$, r^0 is set to the value of r^j at the end of the previous QP.

Algorithm 1 can be viewed as transforming (9) into a sequence of bound constrained convex quadratic subproblems of the form (12). These subproblems have a solution, as soon as (9) has a solution (see Proposition 3.3 in Delbos & Gilbert (2005) for a weaker condition). Clearly, the major part of the CPU time required by Algorithm 1 is spent in solving the bound constrained subproblems (12). We describe an algorithm for doing this efficiently in Section 3.4: Algorithm 3.

Two facts contribute to the observed success of this method. First, a bound constrained QP is much easier to solve than (9), which has general linear constraints (see Moré & Toraldo (1991) and the references therein). Second, because of its dual and constraint convergence, the AL algorithm usually identifies the active constraints of (9) in a finite number of iterations. Since often these active constraints are stable when m_k is in some neighborhood of a solution, the combinatorial aspect of the bound constrained QP's rapidly decreases in intensity as the convergence progresses (and usually disappears after very few AL iterations).

We have already made it clear that the choice of the augmentation parameters r^j is crucial for the efficiency of Algorithm 1. Two pitfalls have to be avoided: a too small value slows down the convergence, a too large value makes it difficult to find a solution to (12). It is usually not easy to determine an a priori appropriate value for the augmentation parameter, so that updating r^j in the course of the AL iterations by observing the behavior of the algorithm looks better. In our implementation of Algorithm 1, the update of r^j at iteration $j \geq 1$ is done by a heuristic close to the one given in Algorithm 2. This one deserves some explanations.

Algorithm 2. A heuristics for updating r^j in Algorithm 1.

data: r^{j-1}, r^j, ρ^j, ρ_{des}, κ^{j-1}, and κ^j;
begin
  if (12) is solved **then**
    $r^{j+1} := r^j$;
1   **if** $\rho^j > \rho_{\text{des}}$ **then** $r^{j+1} := r^j \rho^j / \rho_{\text{des}}$;
  else
2   $r^{j+1} := r^j/10$;
    if $(r^j < r^{j-1})$ & $(\kappa^j > \kappa^{j-1})$ **then**
3   **stop** [failure of Algorithm 1]
    end
  end
end

A too small value of r^j can be deduced from the observation of $\rho^j := \nu^{j+1}/\nu^j$, where $\nu^j := \|(C_E d^j - \tilde{e},\, C_I d^j - y^j)\|$ is the Euclidean norm of the equality constraints of (QP′). It is indeed known from Theorem 4.5 of Delbos & Gilbert (2005) that there is a constant $L > 0$ such that, for all $j \geq 1$, there holds

$$\rho^j \leq \min\left(1, \frac{L}{r^j}\right). \tag{14}$$

This estimate is valid provided (12) is solved exactly. If such is the case and if $\rho_{\text{des}} \in (0,1)$ is a given desired convergence rate for the constraint norm, a value $\rho^j > \rho_{\text{des}}$ is then an incitement to increase the augmentation parameter. This update of r^j is done in statement 1 of Algorithm 2.

On the other hand, a difficulty to solve (12) can be detected by the impossibility to satisfy the optimality conditions of that problem at a given precision in a given number of preconditioned conjugate gradient (CG) iterations (the algorithm to solve (12), based on CG iterations, is explained in Section 3.4). The heuristics has also to decide whether a failure to solve (12) is due to a too large value of r^j, in which case decreasing r^j is appropriate (statement 2), or to an intrinsic excessive ill-conditioning of the problem, in which case a definitive failure is declared (statement 3). For this, it uses the sensitivity to r^j of an estimate κ^j of the condition number of the preconditioned Hessian

$$M_{r^j} := P_{r^j}^{-1/2} Q_{r^j} P_{r^j}^{-1/2},$$

where $Q_{r^j} := H + r^j(C_E^\top C_E + C_I^\top C_I)$ is the Hessian with respect to d of the criterion $\mathcal{L}_{r^j}$ of (12) and P_{r^j} ($\simeq Q_{r^j}$) is a preconditioner for Q_{r^j}. The estimate κ^j makes use of the Rayleigh quotients of M_{r^j} computed during the preconditioned CG iterations. According to Proposition 2.3 of Fortin & Glowinski (1983), the condition number of Q_{r^j} grows asymptotically linearly with r^j. The same law does not hold for M_{r^j}, since hopefully r^j intervenes in P_{r^j}. Nevertheless, it is important to have an estimate of the condition number of M_{r^j}, not of Q_{r^j}, since it is M_{r^j} that governs the performance of the CG algorithm. In view of this comments, it seems reasonable to say that, if a preceding decrease of the augmentation parameter, $r^j < r^{j-1}$, has resulted in an increase of the condition number estimate, $\kappa^j > \kappa^{j-1}$, it is likely that a new decrease of r^j will not improve the conditioning of problem (12). In that case and if it is not possible to solve (12) with r^j, a decision to stop is taken in statement 3 of Algorithm 2; the problem is declared to be too hard to solve.

The actual heuristics for updating r^j in our implementation of the AL algorithm has other safeguards, detailed by Delbos (2004), but the logic is essentially the one presented in Algorithm 2. Experiments with two different initial values r^0 of the augmentation parameter are shown in Section 4.2, illustrating the behavior of the heuristics adapting r^j.

To conclude the description of Algorithm 1, we still need to say a word on its stopping criterion and to specify the value of the multiplier μ_k^{QP} used in (11). There are many ways of showing that the stopping criterion makes sense. The shortest one here is probably to observe that a solution (d^{j+1}, y^{j+1}) to (12) satisfying $C_E d^{j+1} = \tilde{e}$ and $C_I d^{j+1} = y^{j+1}$ is actually a solution to (QP′); d^{j+1} is then a solution to (9). Finally, the optimality conditions of problems (9) and (QP′) show that one can take

$$(\mu_k^{\text{QP}})_E = \lambda_E^{j+1},$$
$$(\mu_k^{\text{QP}})_l = \max(0, \lambda_I^{j+1}), \quad \text{and} \quad (\mu_k^{\text{QP}})_u = \max(0, -\lambda_I^{j+1}),$$

where λ^{j+1} is the value of the multiplier on return from Algorithm 1 at the kth iteration of the SQP algorithm.

3.4 Solving the Lagrange problem by the GP-AS-CG algorithm

Problem (12) is solved in our software by a combination of the gradient projection (GP) algorithm, the active set (AS) method, and conjugate gradient (CG) iterations. This GP-AS-CG algorithm is a classical and efficient method for minimizing a large scale bound constrained convex quadratic function, see Moré & Toraldo (1991), Friedlander & Martínez (1994), Nocedal & Wright (1999), and the references therein. We adapt it below to the special structure of problem (12), in which the variables d and y intervene differently in the objective and only y is constrained.

A brief description of the three ingredients of the GP-AS-CG algorithm is necessary for the understanding of the discussion below. The AS method solves (12) by maintaining

fixed a varying choice of variables y_i to their bounds, while minimizing the objective with respect to the other variables, which are supposed to satisfy the constraints. Each time the minimization would lead to a violation of some bounds, a displacement to the boundary of the feasible set is done and some new variables y_i are fixed to their bounds. The minimization is pursued in this way up to complete minimization with respect to the remained free variables or (in our case) up to the realization of a Rosen-like stopping criterion. The GP algorithm intervenes at this point to inactivate a bunch of erroneously fixed variables and, possibly, to activate others. This GP-AS algorithm proceeds up to finding a solution to (12). Finally, CG iterations are used to minimize the objective on the faces of the feasible set that are activated by the GP-AS algorithm.

Minimizing the objective of (12) in (d, y) jointly can be inefficient, in particular when there are many inequality constraints in the original problem (6), since then the presence of the auxiliary variable y increases significantly the number of unknowns. Our first adaptation of the GP-AS-CG algorithm consists in setting up a minimization in d only, while y is adapted to follow the change in d. Let us clarify this. Suppose that $W \subset I$ is the *working set* at a given stage of the algorithm, that is the set of indices i of the variables y_i that are fixed at one of the bounds $\tilde{l}_i$ or $\tilde{u}_i$. We note $C := (C_E^\top \; C_I^\top)^\top$, $V := I \backslash W$, $\bar{W} := E \cup W$, and denote by C_V (resp. $C_{\bar{W}}$) the matrix obtained from C_I (resp. from C) by selecting its rows with index in V (resp. in $\bar{W}$). For the current working set W, the algorithm has to solve (we drop the index j of the AL algorithm)

$$\min_{(d, y_V)} \; \mathcal{L}_r(d, y, \lambda).$$

with implicit bound constraints on $y_V \in [\tilde{l}_V, \tilde{u}_V]$ and with y_W fixed. The optimality conditions of this problem can be written

$$\begin{aligned} (H + rC^\top C)d - rC_V^\top y_V &= -g - C^\top \lambda + rC_E^\top \tilde{e} + rC_W^\top y_W, \\ -rC_V d + ry_V &= \lambda_V. \end{aligned} \tag{15}$$

Substituting the value of y_V given by (15) into the first equation gives

$$(H + rC_{\bar{W}}^\top C_{\bar{W}})d = -g - C_{\bar{W}}^\top \lambda_{\bar{W}} + rC_E^\top \tilde{e} + rC_W^\top y_W. \tag{16}$$

This is the equation that is solved by the preconditioned CG algorithm. During this process, y_V is updated so that equation (15) is continually verified. It can be shown that the directions modifying (d, y) are conjugate in $\mathbb{R}^n \times \mathbb{R}^{n_I}$ with respect to the Hessian matrix of $\mathcal{L}_r(\cdot, \cdot, \lambda)$, so that this method can be viewed as a conjugate direction algorithm that maintains the iterates in the affine subspace defined by equation (15).

In this process, as soon as y_V hits a new bound, the working set W is enlarged. Then a new CG process is started on the updated equations (16) and (15), from the (d, y) obtained so far. Note that the updated (15) is verified at the beginning of this new process, since it is obtained by deleting equations from (15) and since (15) was verified at the end of the previous CG process. We will see that (15) is also verified at the end of a GP phase of the algorithm, so that this equation makes no difficulty to be maintained all along the GP-AS-CG algorithm, provided this one starts by a GP phase.

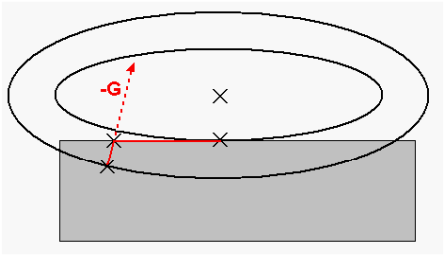

Figure 3. A 2D view of one iteration of the gradient projection (GP) algorithm: the ellipses are the level curves of the objective function, the vector $-\mathrm{G}$ is its negative gradient at the current iterate, and the shadow box represents the feasible set.

The goal of the gradient projection (GP) phase of the algorithm is to change drastically the working set if this one is felt to be very different from the set of the active constraints at the solution. As explained below, the GP phase is actually an adapted version of a single iteration of the GP algorithm (see Bertsekas (1995) for example); more iterations would be useless in our case, as we will see. Its property mentioned above, which is observed in practice, is then also supported by the fact that the GP algorithm identifies the active constraints at the solution in a finite number of iterations when strict complementarity holds.

An iteration of the standard GP algorithm forces the decrease of the objective of (12) along the piecewise linear path obtained by projecting the steepest descent path on the feasible set (see Figure 3). If $P_{[\tilde{l}, \tilde{u}]}$ stands for the projection operator on $[\tilde{l}, \tilde{u}]$ in $\mathbb{R}^{n_I}$, the projected steepest descent path emanating from the current iterate (d, y) is the mapping

$$p : (\alpha > 0) \mapsto \begin{pmatrix} d - \alpha g_d \\ P_{[\tilde{l}, \tilde{u}]}(y - \alpha g_y) \end{pmatrix},$$

where g_d (resp. g_y) is the gradient of $\mathcal{L}_r$ with respect to d (resp. y). There holds

$$g_y = -\lambda_I - r(C_I d - y).$$

In the standard GP algorithm, a local minimum or a step-size ensuring a sufficient decrease of the complex piecewise quadratic function $\alpha \mapsto \mathcal{L}_r(p(\alpha), \lambda)$ is usually taken. Because of the particularly simple structure of $\mathcal{L}_r$ in y, we prefer maintaining d fixed and minimizing completely

$$\alpha \mapsto \mathcal{L}_r(d, P_{[\tilde{l}, \tilde{u}]}(y - \alpha g_y), \lambda).$$

This is our second adaptation of the standard GP-AS-CG algorithm. It has the following interesting property. Because the Hessian of $\mathcal{L}_r$ with respect to y is a multiple of the identity matrix, the new y is the projection on $[\tilde{l}, \tilde{u}]$ of the unconstrained minimizer of $\mathcal{L}_r(d, \cdot, \lambda)$:

$$y := P_{[\tilde{l}, \tilde{u}]}\left(C_I d + \frac{\lambda_I}{r}\right). \tag{17}$$

Observe that, if W is now set to $\{i : y_i = \tilde{l}_i \text{ or } \tilde{u}_i\}$, equation (15) is satisfied, as claimed above.

Algorithm 3 summarizes our version of the GP-AS-CG algorithm. The use of Rosen's stopping test for the CG iterations and other algorithmic details are not mentioned. See

Delbos (2004) for further information on the implementation.

Algorithm 3. The GP-AS-CG algorithm to solve (12).

data: r, λ, $d := 0$;
begin
  GP := `true`;
  while optimality of (12) is not satisfied **do**
    if GP **then**
      compute y by (17);
      update $W := \{i : y_i = \tilde{l}_i \text{ or } \tilde{u}_i\}$ and $V := I \backslash W$;
      GP := `false`;
    else
      while $\tilde{l}_V < y_V < \tilde{u}_V$ **do**
        use CG iterations to solve (16) in d, while updating y_V to satisfy (15);
      end
      update the index sets W and V;
      if W has not been enlarged **then** GP := `true`;
    end
  end
end

3.5 Globalization by line-search

It is now time to remember that the constrained minimization problem (6) we want to solve is nonlinear and that, just as in unconstrained optimization, the update of the model m_k by (10), where d_k is the computed solution to the quadratic problem (QP_k), is unlikely to yield convergence if the initial estimate m_0 is not close enough to a solution. For forcing convergence from a remote starting model, we follow the standard globalization technique presented in Chapter 15 of Bonnans et al. (2003), which uses an exact penalty merit function. Using a filter method would have been an alternative, but we did not try it. We have also implemented a line-search technique, since the combination of trust regions with a QP approximately solved by an augmented Lagrangian algorithm is a technique that does not seem to have been explored. Due to its usefulness to solve the unconstrained tomography problem, we plan to investigate this possibility in a future research.

We use the exact penalty function $\Theta_\tau : \mathbb{R}^n \to \mathbb{R}$ defined by

$$\Theta_\tau(m) = f(m) + \Psi_\tau(m),$$

where

$$\Psi_\tau(m) = \sum_{i \in E} \tau_i \, | \, C_i m - e_i \, | + \sum_{i \in I} \tau_i \max(l_i - C_i m, 0, C_i m - u_i),$$

in which the τ_i's are penalty weights. Exactness of the penalization means that a solution to (6) is also a minimizer of Θ_τ. To get that important property, the τ_i's need to be large enough, although finite. It is usual to update them at some iteration, so that they always satisfy

$$\begin{cases} \tau_i \geq |(\mu_k^{\text{QP}})_i| + \bar{\tau}, \text{ for } i \in E \\ \tau_i \geq \max(|((\mu_k^{\text{QP}})_l)_i|, |((\mu_k^{\text{QP}})_u)_i|) + \bar{\tau}, \text{ for } i \in I, \end{cases} \tag{18}$$

where μ_k^{QP} is defined after (11) and $\bar{\tau} > 0$ is a small security threshold.

In our case, the convexity of the QP (9) defining d_k and the inequalities (18) ensure that Θ_τ decreases along d_k. A line-search along this direction is then possible. A step-size $\alpha_k > 0$, typically determined by backtracking, can be found such that for some small constant $\omega \in (0, 1/2)$:

$$\Theta_\tau(m_k + \alpha_k d_k) \leq \Theta_\tau(m_k) + \alpha_k \omega \Delta_k, \tag{19}$$

where $\Delta_k := \nabla f(m_k)^\top d_k - \Psi_\tau(m_k)$ can be shown to be negative when m_k is not stationary. Now the model and the multiplier are updated by

$$\begin{cases} m_{k+1} = m_k + \alpha_k d_k \\ \mu_{k+1} = \mu_k + \alpha_k(\mu_k^{\text{QP}} - \mu_k). \end{cases} \tag{20}$$

Algorithm 4 summarizes the overall algorithm to solve problem (6).

Algorithm 4. The overall algorithm to solve (6).

data: m_0, μ_0;
begin
  for $k = 0, 1, 2, \ldots$ **do**
    if optimality of (6) holds at (m_k, μ_k) **then stop**;
    compute (d_k, μ_k^{QP}) a primal-dual solution to (12) by Algorithm 1;
    update τ to satisfy (18);
    determine a step-size $\alpha_k > 0$ by backtracking, in order to satisfy (19);
    update (m_{k+1}, μ_{k+1}) by (20);
  end
end

4 APPLICATIONS OF THE CONSTRAINED REFLECTION TOMOGRAPHY ON REAL DATA SETS

4.1 2D PP/PS data set

In this section, we present an application of constrained reflection tomography to one 2D line of a 3D 4C OBC (Ocean Bottom Cable) survey with PP and PS data from bp. Broto et al. (2003) have already interpreted and studied this data set using an unconstrained inversion method. The velocity model is described by four velocity layers and five interfaces (cf Figure 4 left). The isotropic assumption was satisfying until the last layer (layer which contains the last two interfaces h4 and h5). By applying the anisotropic inversion methodology of Stopin (2001) on the last layer, they obtained a model that fits the traveltimes better than any of the previous isotropic models and that, in addition, has more reliable velocity variations. Two parameters (η, δ) describe the velocity anisotropy: η can be seen as a measure of the an-ellipticity, whereas δ controls the near vertical velocity propagation of the P-waves (Stopin (2001)).

The value of the δ anisotropy parameter has been obtained by a trial and error approach in order to match approximately the h5 depth given by well logs. Actually, the under-determination of the inverse problem does not allow the joint inversion of the velocities, interfaces and

Table 1. Inversion results of the unconstrained inversion.

Layer 4	RMS of traveltime misfits	Depth mismatch at well location	η	δ
h4-PP	3.6ms	190m	6.2%	2%
h5-PP	4.6ms	150m		
h5-PS	9.8ms			

Table 2. Inversion results of the constrained inversion.

Layer 4	RMS of traveltime misfits	Depth mismatch at well location	η	δ
h4-PP	6.3ms	0m	8.2%	15.9%
h5-PP	3.9ms	0m		
h5-PS	8.1ms			

anisotropy parameters (δ parameter is particularly undetermined, Stopin (2001)). This applied methodology is obviously not optimal. Indeed, the manual tuning of the anisotropy parameters requires a lot of time: an important numbers of anisotropic inversion with different pairs (η, δ) have to be performed before getting a satisfying result. Secondly, it is very hard to make this method accurate: we note a discrepancy of 150 meters for the reflector depth h5 compared to the depth given by the well logs data. Finally, it turned out impossible to determine the anisotropy parameter δ so that both the reflector depths of h4 and h5 given by the well logs are reasonably matched.

The solution we propose here is to compute a model using our constrained inversion method in order to fit the reflector depths given by the well, while considering (η, δ) as additional unknowns to determine. This consists in the inversion of P and S velocity variations, of the geometries of interfaces h4 and h5 and of the two anisotropy parameters, i.e., inversion of 1024 parameters from 32468 picked traveltimes (associated with reflections of PP waves and PS waves on h4 and h5).

In Tables 1 and 2, we have respectively summed up the results of the final models obtained with the unconstrained and constrained inversions. The final model (Figure 4 right) of the constrained inversion matches the traveltime data with the same accuracy than the result obtained by the unconstrained inversion (Figure 4 left), and it strictly verifies the reflector depths given by the well logs. This solution has been obtained after only 9 nonlinear SQP iterations.

We see that, the introduction of constraints at wells for the two reflectors h4 and h5 reduces the underdetermination and allows for a joint inversion of velocities, interfaces, and anisotropy parameters. The resulting model matches the traveltime and well data. The values of its anisotropy parameters are very different from those obtained with the unconstrained inversion (Tables 1 and 2).

4.2 3D PP data set

During the European KIMASI project, reflection tomography was applied on a 3D North Sea data set from bp (Ehinger et al. (2001)). A P-velocity model was obtained thanks to a top-down layer-stripping approach where lateral and vertical velocity variations within Tertiary, Paleocene, and Cretaceous units (this last layer being divided in two velocity layers) have been determined sequentially. A strong velocity under-determination in the upper Tertiary layer was detected during the inversion process due to the large layer thickness (2.5km) and to the very poor ray aperture (despite the introduction of the intermediary reflector named layer1). Several velocity parameterizations (Table 3) were inverted and led to solution models that match traveltime data with the same accuracy. These different tests are time consuming (each test is a whole inversion) and the reflector depths given by well data are not well retrieved, this information being not explicitly introduced in the inversion process. One of the resulting model is presented in Figure 5.

To obtain a model consistent with well data, we propose to apply our developed constrained tomography inversion. The interface depths are constrained at 5 well locations and we constrain the range of variations of the vertical velocity gradient in the Tertiary layer thanks to well measurements (Table 4). A global inversion of the 4 velocity layers is preferred to the layer-stripping approach: it avoids bad data fitting for deeper layers due to errors in shallow layers. The simultaneous inversion of all the layers is guided by constraints on layer thicknesses to avoid any non-physical interface intersections: Figure 6 shows an example of non-admissible model obtained by global unconstrained inversion (it presents non-physical interface intersections that make layers vanish in the pointed region and thus a large number of rays are discarded).

The experiment then consists in a global inversion of 127 569 traveltimes for 5960 unknowns describing 4 velocity layers and 5 interfaces, subject to 2300 constraints (Table 4). The constraints on the velocity variations (resp. on the layer thicknesses) are applied on a grid of 10x10x10 (resp. 20x20 points). The results are presented in Figure 7: the obtained model matches the data with the same accuracy as the models obtained by Ehinger et al (2001) and verifies all the introduced constraints (see Tables 4 and 3). The model obtained by unconstrained inversion (Figure 5) and the model obtained by constrained inversion (Figure 7) are very different: by the geometry of the interfaces and by the velocity variations within the layers. The introduction of constraints leads to local velocity variations at well locations that may perhaps be attenuated by a stronger regularization.

The total number of conjugate gradient iterations for each Gauss-Newton step (the total number of conjugate gradient iterations takes into account all the iterations of augmented Lagrangian method) is less than 10^4 (less than twice the number of unknowns), which looks like a very good result for a problem with 2300 constraints. In this experiment, only 6 nonlinear SQP iterations are necessary to reach convergence.

The automatic adaptation of the augmentation parameter r^j (see Algorithm 2 in Section 3.3) is illustrated in Figure 8. The initial value r^0 can be chosen in a large interval. Algorithm 2 modifies r^j in order to obtain a good convergence rate of the multipliers in the AL algorithm (Algorithm 1), without deterioration of the conditioning of the bound constrained problem (12).

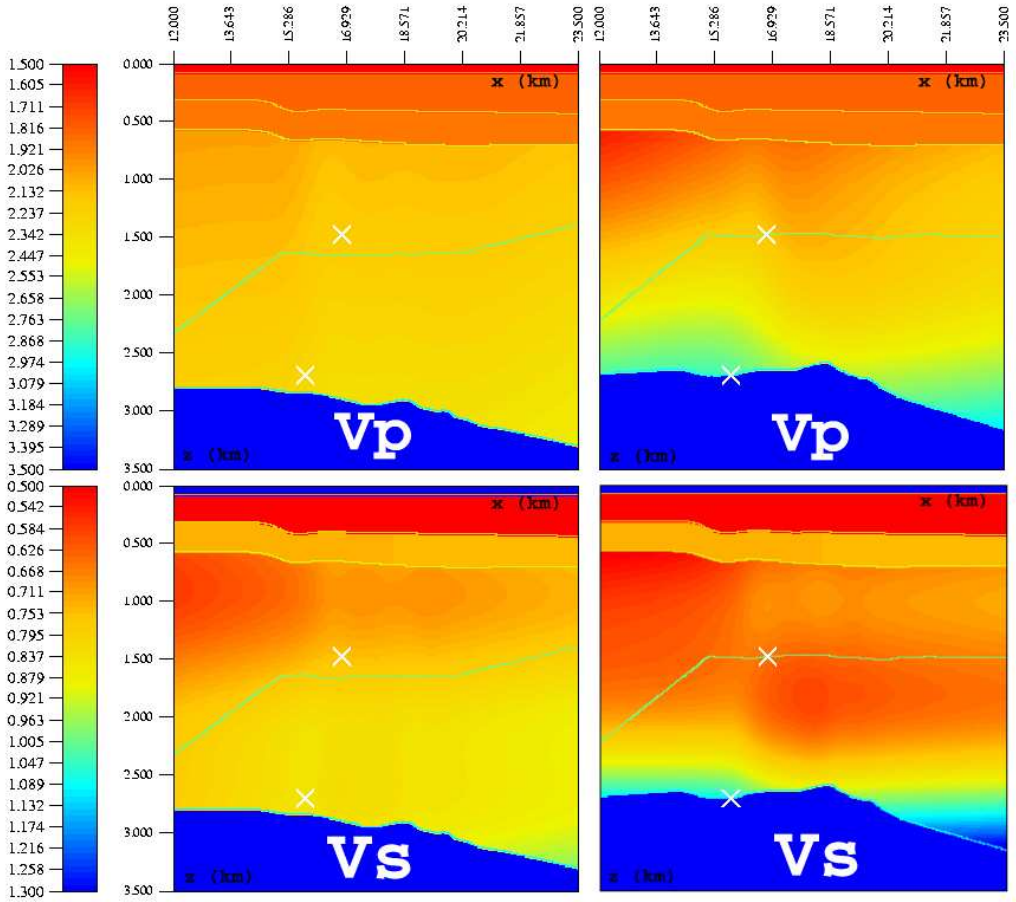

Figure 4. The velocity models (Vp and Vs) on the left hand side are computed with the unconstrained inversion method; those on the right hand side are computed with the constrained inversion method. The white crosses locate reflector depths measured from the deviated well logs, which are imposed as constraints. These additional constraints allows the software to determine the anisotropy parameters in the same run.

Tertiary velocity parameterization			RMS of traveltime misfits		Mean depth mismatch at the 5 well locations
Velocity type		Vertical velocity gradient	Layer 1	Top of Paleocene	
$V_0(x, y)$		fixed to 0/s	4.3ms	4.2ms	96m
$V_0(x, y)+0.2z$		fixed to 0.2/s	3.4ms	4.6ms	300m
$V(x, y, z)$	without constraints	inverted ~ 0.01/s	4.1ms	4.4ms	100m
	with constraints	inverted ~ 0.18/s	3.9ms	4.4ms	0m

Table 3. The different velocity parameterizations tested for the inversion of the Tertiary layer (in the layer-stripping approach applied by Ehinger et al. (2001)) for the unconstrained inversion and in a global constrained approach (last line of the table).

5 CONCLUSION

Reflection tomography often requires the introduction of additional a priori information on the model in order to reduce the under-determination of the inverse problem. A nonlinear optimization method that can deal with linear constraints on the model has been developed. This dedicated method has proved its efficiency on various concrete inversions, including those related to the two real data sets presented in this paper. The number of Gauss-Newton iterations has the same order of magnitude than the number of Gauss-Newton iterations necessary in the unconstrained case ($\sim$ 10 iterations).

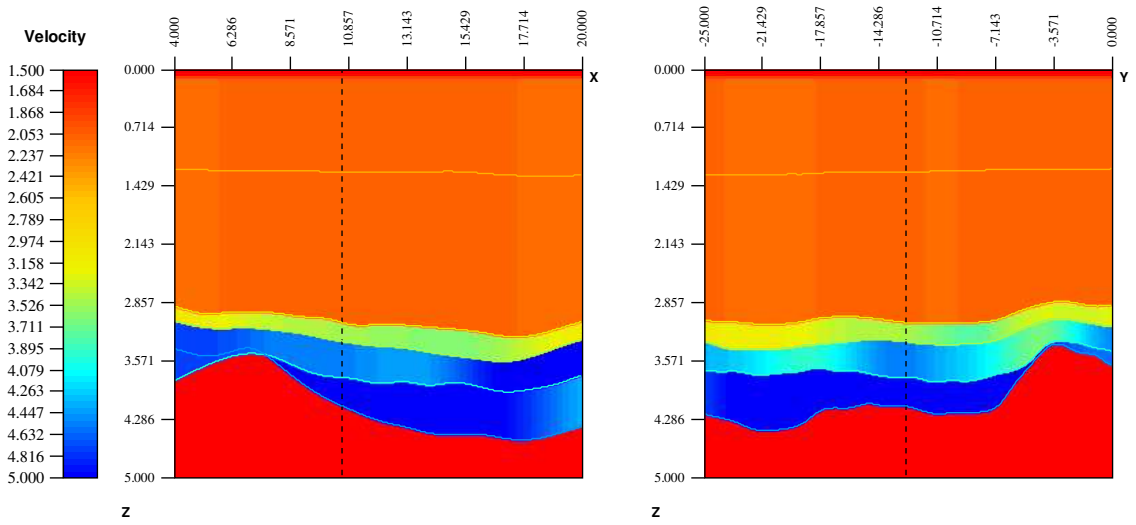

Figure 5. Velocity model (slices along x (left) and along y directions at one of the 5 well locations) obtained with a layer-stripping approach using the unconstrained reflection tomography. This model corresponds to the $v(x, y, z)$ velocity parameterization (see Table 3). The RMS value of the traveltime misfits is 6.1ms.

Table 4. Description of the equality and inequality constraints introduced in the inversion. We succeed to obtain a model that matches the data and the 2300 constraints.

Constraints		Model obtained with the UNCONSTRAINED inversion	Model obtained with the CONSTRAINED inversion
Mean depth mismatch at the 5 well locations	tpal	96m	0m
	tchalk	132m	0m
	bchalk	140m	0m
Vertical velocity gradient in Tertiary	0.1<k<0.3/s	k=0/s	k~0.18/s
Velocity range	2.5 < vpal < 4km/s	ok	ok
	3.5 < vichalk < 5.7km/s	ok	ok
	4.2 < vchalk < 5.8km/s	ok	ok

This and the fact that solving the forward problem is time consuming were preponderant factors in favor of a sequential quadratic programming approach, instead of a nonlinear interior point method. The chosen constraint activation method is efficient even for a large number of constraints. At last, the algorithm developed for updating the augmentation parameter discharges the user of the software from such a technical concern and allows an adequate choice of its value in terms of the convergence rate of the augmented Lagrangian algorithm.

ACKNOWLEDGMENTS

We would like to thank Guy Chavent and Roland Masson for helpful discussions and advises at various stages of the elaboration of this work. We acknowledge Karine Broto and Anne Jardin for their precious help for the applications on real datasets. The datasets studied in this paper were kindly provided by bp.

REFERENCES

Bertsekas, D., 1995. *Nonlinear Programming*, Athena Scientific, (second edition, 1999).

Bishop, T. N., Bube, K. P., Cutler, R. T., Langan, R. T., Love, P. L., Resnick, J. R., Shuey, R. T., Spindler, D. A., & Wyld, H. W., 1985. Tomographic determination of velocity and depth in laterally varying media, *Geophysics*, **50**(6), 903–923.

Bonnans, J., Gilbert, J. C., Lemaréchal, C., & Sagastizábal, C., 2003. *Numerical Optimization – Theoretical and Practical Aspects*, Springer Verlag, Berlin.

Broto, K., Ehinger, A., & Kommedal, J. H., 2003. Anisotropic traveltime tomography for depth consistent imaging of PP and PS data, *The Leading Edge*, pp. 114–119.

Bube, K. P. & Langan, R. T., 1994. A continuation approach to regularization for traveltime tomography, 64^{th} *Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pp. 980–983.

Bube, K. P. & Meadows, M. A., 1999. The null space of a generally anisotropic medium in linearized surface reflection tomography, *Geophys. J. int.*, **139**, 9–50.

Byrd, R., Gilbert, J. C., & Nocedal, J., 2000. A trust region method based on interior point techniques for nonlinear programming, *Mathematical Programming*, **89**, 149–185.

Červený, V., 1987. Ray tracing algorithms in three-dimensional

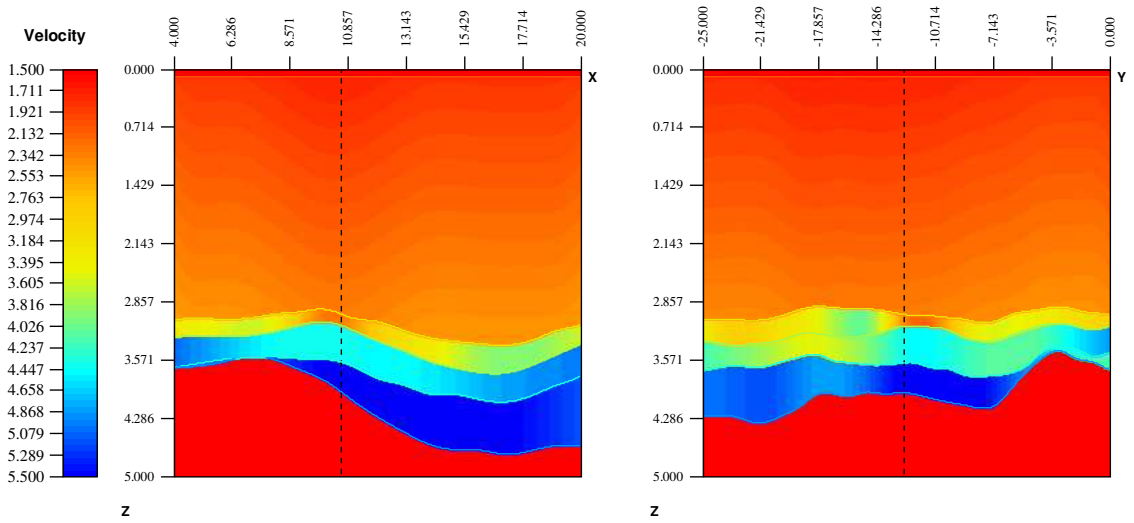

Figure 7. Velocity model (slices along x (left) and along y directions at one of the 5 well locations) obtained with a global inversion using the constrained reflection tomography (constraints described in Table 4). The RMS value of the traveltime misfits is 6.5ms.

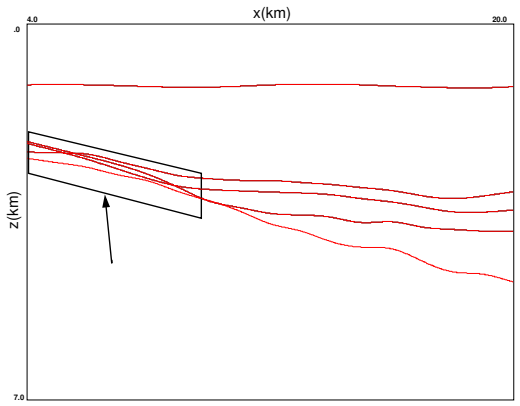

Figure 6. Interfaces (slice along x) obtained with a global inversion using the unconstrained reflection tomography. Difficulties are encountered during the Gauss-Newton iterations. Some iterates lead to non-admissible models in which the forward operator is not defined. For instance, the model found at iteration 7 presents non-physical intersections that make layers vanish in the pointed region and thus a large number of rays are discarded. The resulting discontinuity of the cost function leads to convergence troubles of the Gauss-Newton method.

laterally varying layered structures, in *Seismic Tomography*, edited by G. Nolet, pp. 99–133, D. Reidel.

Chauvier, L., Masson, R., & Sinoquet, D., 2000. Implementation of an efficient preconditioned conjugate gradient in jerry, *KIM Annual Report, Institut Français du Pétrole, Rueil-Malmaison, France, http://consortium.ifp.fr/KIM/*.

Conn, A., Gould, N., & Toint, P., 2000. *Trust-Region Methods*, MPS/SIAM Series on Optimization 1, SIAM and MPS.

Courtier, P. & Talagrand, O., 1987. Variational assimilation of meteorological observations with the adjoint vorticity equation. II: Numerical results, *Quaterly Journal of the Royal Meteorological Society*, **113**, 1329–1347.

Courtier, P., Thépaut, J., & Hollingsworth, A., 1994. A strategy for operational implementation of 4D-Var, using an incremental approach, *Quaterly Journal of the Royal Meteorological Society*, **120**, 1367–1387.

de Boor, C., 1978. *A Practical Guide to Splines*, Springer.

Delbos, F., 2004. *Problème d'optimisation de grande taille avec contraintes en tomographie de réflexion 3D*, Ph.D. thesis, Université Pierre et Marie Curie, Paris VI, France.

Delbos, F. & Gilbert, J. C., 2005. Global linear convergence of an augmented Lagrangian algorithm for solving convex quadratic optimization problems, *Journal of Convex Analysis*, to appear.

Delbos, F., Sinoquet, D., Gilbert, J. C., & Masson, R., 2001. A trust-region Gauss-Newton method for reflection tomography, *KIM Annual Report, Institut Français du Pétrole, Rueil-Malmaison, France, http://consortium.ifp.fr/KIM/*.

Delprat-Jannaud, F. & Lailly, P., 1993. Ill-posed and well-posed formulations of the reflection travel time tomography problem, *J. Geophys. Res.*, **98**, 6589–6605.

Dennis, J., Gay, D., & Welsch, R., 1981. An adaptive nonlinear least squares algorithm, *ACM Transactions on Mathematical Software*, **7**, 348–368.

Ehinger, A., Broto, K., Jardin, A., & the KIMASI team, 2001. 3D tomographic velocity model determination for two North Sea case studies, in *Expanded Abstracts*, EAGE.

Fletcher, R., 1987. *Practical Methods of Optimization*, John Wiley & Sons, Chichester, 2nd edn.

Fortin, M. & Glowinski, R., 1983. *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, North-Holland, Amsterdam.

Friedlander, A. & Martínez, J. M., 1994. On the maximization of a concave quadratic function with box constraints, *SIAM Journal on Optimization*, **4**, 177–192.

Gauss, 1809. *Theoria motus corporum coelestium*.

Glowinski, R. & Le Tallec, P., 1989. *Augmented Lagrangian and Operator Splitting Methods in Nonlinear Mechanics*, SIAM, Philadelphia.

Glowinski, R. & Tran, Q., 1993. Constrained optimization in reflection tomography: the Augmented Lagrangian method,

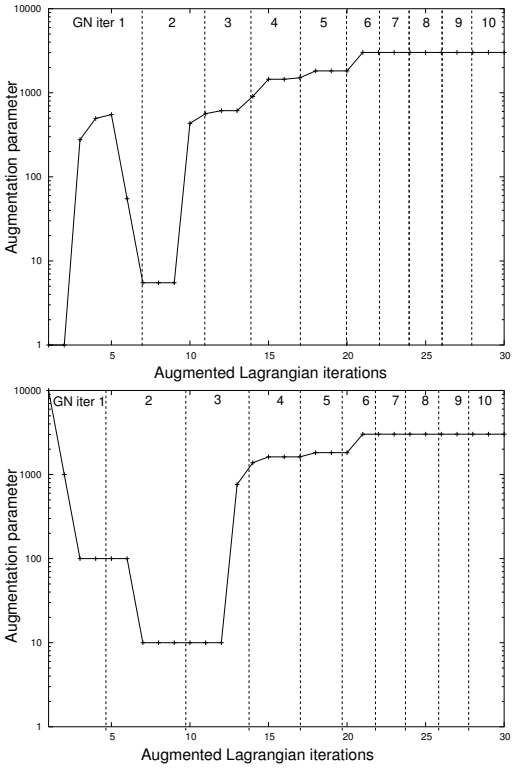

Figure 8. Automatic adaptation of augmentation parameter r^j: two experiments are performed, with an initial augmentation parameter either set to $r^0 = 1$ (top) or to $r^0 = 10^4$ (bottom). Algorithm 2 adapts r^j during the AL iterations. It may decrease its value to improve the conditioning of problem (12) or increase r^j to speed up the convergence of the constraint norm to zero (the desired convergence rate ρ_{des} is fixed to 10^{-3}).

East-West J. Numer. Math., **1**, 213–234.

Gould, N., Orban, D., Sartenaer, A., & Toint, P., 2000. Superlinear convergence of primal-dual interior point algorithms for nonlinear programming, *Mathematical Programming*, **87**, 215–249.

Hansen, P. C., 1992. Analysis of discrete ill-posed problems by mean of the L-curve, *SIAM Review*, **34**, 561–580.

Hestenes, M., 1969. Multiplier and gradient methods, *Journal of Optimization Theory and Applications*, **4**, 303–320.

Inoue, H., 1986. A least-squares smooth fitting for irregularly spaced data: finite element approach using the cubic B-splines basis, *Geophysics*, **51**, 2051–2066.

Jurado, F., Lailly, P., & Ehinger, A., 1998. Fast 3D two-point raytracing for traveltime tomography, *Proceedings of SPIE, Mathematical Methods in Geophysical Imaging V*, **3453**, 70–81.

Lailly, P. & Sinoquet, D., 1996. Smooth velocity models in reflection tomography for imaging complex geological structures, *Geophys. J. Int.*, **124**, 349–362.

Moré, J. & Toraldo, G., 1991. On the solution of large quadratic programming problems with bound constraints, *SIAM Journal on Optimization*, **1**, 93–113.

Nocedal, J. & Wright, S., 1999. *Numerical Optimization*, Springer Series in Operations Research, Springer, New York.

Powell, M., 1969. A method for nonlinear constraints in minimization problems, in *Optimization*, edited by R. Fletcher, pp. 283–298, Academic Press, London.

Rockafellar, R., 1973. A dual approach to solving nonlinear programming problems by unconstrained optimization, *Mathematical Programming*, **5**, 354–373.

Sebudandi, C. & Toint, P. L., 1993. Non linear optimization for seismic traveltime tomography, *Geophysics Journal International*, **115**, 929–940.

Stopin, A., 2001. *Détermination de modèle de vitesses anisotropes par tomographie de réflexion des modes de compression et de cisaillement (in English)*, Ph.D. thesis, Université de Strasbourg I, France, http://consortium.ifp.fr/KIM/.

Tikhonov, A. & Arsenin, V., 1977. *Solution of ill–posed problems*, Winston.

Yabe, H. & Yamaki, N., 1995. Convergence of a factorized Broyden-like family for nonlinear least squares problems, *SIAM Journal on Optimization*, **5**, 770–790.

Conclusions

La tomographie de réflexion sismique permet de déterminer la distribution de vitesse de propagation des ondes sismiques dans le sous-sol à partir de leur temps de trajet lorsqu'elles se réfléchissent sur les interfaces géologiques. Du point de vue de l'optimisation, cela consiste à résoudre un problème inverse de moindres-carrés non-linéaire avec contraintes qui a pour objectif de rechercher un modèle de sous-sol lisse tel que les temps de trajet calculés par tracé de rayons s'ajustent au mieux avec les temps de trajet observés. L'ajout d'une régularisation sur la courbure du modèle permet d'avoir un problème inverse bien posé. Le sujet de la thèse est l'étude et l'amélioration des techniques de résolution de ce problème d'optimisation non linéaire. Les principales caractéristiques des problèmes de tomographie de réflexion sont :

- la dimension de l'espace des données et de l'espace des modèles peut être très grande (jusqu'à 10^6 données et 10^5 inconnues),
- les problèmes sont très mal conditionnés (les hessiens approchés par la méthode de Gauss-Newton peuvent avoir un conditionnenment supérieur à 10^9),
- la résolution du problème direct est une opération coûteuse en temps CPU, le nombre de rayon à calculer étant grand (10^6),
- l'opérateur de modélisation T des temps de trajet est non-linéaire,
- la matrice jacobienne J des temps de trajet est aisément calculable.

Classiquement, une méthode de Gauss-Newton globalisée par recherche linéaire est utilisée pour résoudre les problèmes d'optimisation sans contraintes en tomographie de réflexion sismique. A chaque itération de Gauss-Newton, une solution (approchée) d'un modèle quadratique de la fonction coût non linéaire est calculée en utilisant l'algorithme du gradient conjugué. Cette méthode permet en général de résoudre les problèmes de minimisation traités en tomographie. Cependant, nous avons remarqué pour certains cas que la méthode de recherche linéaire était non seulement incapable de trouver une solution, mais encore de faire décroître de manière significative la fonction coût. Ces difficultés, principalement dues à une matrice Hessienne mal conditionnée sont mieux prises en compte lorsque l'on remplace la recherche linéaire par une méthode de région de confiance. La première partie de ce travail de thèse a consisté à développer et tester une méthode de Gauss-Newton globalisée par région de confiance où le modèle quadratique de la fonction coût est minimisé par un algorithme de gradient conjugué tronqué. Cette méthode donne des résultats plus stables et plus précis que la méthode de recherche linéaire sur les exemples traités.

La sous-détermination de la solution reste un problème majeur en tomographie de réflexion : l'inversion nécessite l'intégration d'information a priori sur le modèle recherché. L'introduction de contraintes sur le modèle recherché est une technique permettant de réduire le nombre de solutions admissibles du problème inverse. L'objectif principal de cette thèse a été de rechercher et de développer une méthode d'optimisation avec contraintes adaptée aux caractéristiques des problèmes de tomographie de réflexion. Dans l'état actuel de l'art, il existe deux classes de méthodes permettant de résoudre les problèmes d'optimisation non-linéaires avec contraintes : la classe des méthodes de

pénalisation (qui inclut les méthodes de lagragien augmenté et les méthodes de points intérieurs) et la classe des méthodes newtoniennes (principalement composée de l'approche SQP). Pour résoudre le problème inverse avec contraintes de la tomographie de réflexion nous avons préféré une méthode SQP à une méthode de pénalisation. Ce choix est motivé par les deux arguments suivants :

- en tant que méthode de type Newton, la méthode SQP semble demander moins d'évaluations de la fonction coût qu'une méthode de pénalisation (par exemple, une méthode des points intérieurs),
- la méthode de Gauss-Newton utilisée en tomographie de réflexion sans contrainte converge en très peu d'itérations (de l'ordre de 10) et on pouvait s'attendre à ce qu'une approche SQP ait la même efficacité.

Le principe de la méthode SQP est de décomposer le problème d'optimation non-linéaire avec contraintes en une suite de problèmes quadratiques tangents. Ainsi, à chaque itération de l'algorithme SQP, un problème quadratique convexe avec contraintes linéaires doit être résolu. Le travail principal de cette thèse a été le développement du code de calcul QPAL (Quadratic Programming and Augmented Lagrangian) qui permet de résoudre efficacement les problèmes quadratiques tangents provenant de la méthode SQP par l'utilisation d'une méthode de lagrangien augmenté. Dans l'état actuel de l'art, il existe trois grandes classes de méthodes pour résoudre les problèmes quadratiques convexes : les méthodes d'activation de contraintes, les méthodes de points intérieurs et les méthodes de lagrangien augmenté. Notre choix s'est porté sur cette dernière pour les raisons suivantes.

- Les méthodes d'activation de contraintes nous ont semblé peu efficaces sur les problèmes comportant des milliers de contraintes d'inégalité, du fait de leur manque d'efficacité pour identifier les contraintes actives en la solution.
- Malgré leur efficacité sur les problèmes quadratiques convexes, nous avons écarté les méthodes de points intérieurs car il semble que celles-ci peuvent difficilement tirer parti d'une bonne connaissance des contraintes actives en la solution, connaissance qui provient de la propriété d'identification finie des contraintes actives de la méthode SQP. De plus, le mauvais conditionnement inhérent aux systèmes linéaires générés par l'algorithme des points intérieurs et la nécessité de les résoudre par une méthode itérative nous sont apparus comme des arguments supplémentaires pour écarter cette méthode.
- La méthode du lagrangien augmenté semble être bien adaptée aux caractéristiques des problèmes de tomographie de réflexion car :
  - ⋆ elle peut résoudre des problèmes d'optimisation de grande taille (l'algorithme du lagrangien augmenté peut être développé de manière à éviter des factorisations matricielles),
  - ⋆ elle peut tirer parti d'une bonne connaissance des contraintes actives en la solution (le lagrangien augmenté a une propriété d'identification finie des contraintes actives),

- ⋆ les systèmes linéaires générés par l'algorithme du lagrangien augmenté sont susceptibles d'être mieux conditionnés que ceux issus d'une méthode de points intérieurs (dans la méthode du lagrangien augmenté, il n'est pas nécessaire de faire tendre le paramètre d'augmentation r vers $+\infty$ pour assurer la convergence).

La méthode du lagrangien augmenté consiste à transformer la résolution d'un problème quadratique tangent en une suite de sous-problèmes quadratiques convexes avec contraintes de bornes qui sont appelés les problèmes de Lagrange. L'algorithme GP-AC-GC qui combine l'algorithme du gradient avec projection, la méthode d'activation de contrainte et l'algorithme du gradient conjugué est une méthode classique très efficace pour résoudre les problèmes avec contraintes de borne. Une partie de ce travail de thèse a consisté à améliorer l'efficacité de ces algorithmes par :

- le résultat de convergence linéaire du LA qui a permis de mettre au point une heuristique efficace pour adapter le facteur d'augmentation r.
- l'adaptation de l'algorithme du gradient conjugué pour minimiser le critère quadratique sur une face de contraintes actives (au lieu de résoudre un système linéaire augmenté par les variables d'écarts, on résout un systéme linéaire dont l'ordre est égal à la dimension de l'espace des modèles),
- la simplification de l'algorithme du gradient projeté en minimisant le critère quadratique uniquement sur les variables de bornes (cette adaptation permet de désactiver plus rapidement des paquets de contraintes),
- la réduction du nombre d'itérations de gradient conjugué en utilisant le critère de Rosen,
- l'activation rapide de contraintes entre deux étapes consécutives de GC en utilisant la projection de la dernière direction conjuguée trouvée par le gradient conjugué (étape de direction projetée).

Comme nous venons de le montrer précédemment l'ensemble de la méthode SQP que nous avons choisie est adaptée aux caractéristiques des problèmes inverses de la tomographie de réflexion.

Une partie importante de cette thèse a aussi consisté à montrer qu'en pratique notre méthode SQP fonctionne bien. Les nombreuses inversions faites non seulement à partir de notre bibliothèque de "problèmes-tests" mais aussi sur deux jeux de données réelles ont démontré que :

- notre méthode SQP est efficace pour minimiser le nombre d'évaluations de la fonction coût non-linéaire. En effet, le solveur SQP génère un nombre d'itérations de Gauss-Newton du même ordre de grandeur que le nombre d'itérations réalisées par le solveur gauss-newtonien sans contrainte (de l'ordre de 10),
- notre méthode SQP est efficace même sur des problèmes comportant beaucoup de contraintes d'inégalité (la méthode GP-AC-GC ne semble pas pénalisée par la combinatoire de ces problèmes),

- l'algorithme permettant de régler la valeur du paramètre d'augmentation au cours des itérations de lagrangien augmenté rend le solveur QPAL automatique. Elle ne requiert aucune intervention de l'utilisateur au cours de l'inversion,
- l'utilisation des contraintes définies à partir d'informations a priori sur le modèle de sous-sol recherché permet de réduire les incertitudes sur le modèle solution du problème inverse.

Les travaux de cette thèse laisse un certain nombre de questions ouvertent et débouchent sur quelques perspectives :

1. La généralisation du solveur SQP à des contraintes non-linéaires permettrait de prendre en compte des contraintes sur les points d'impact des rayons ou des contraintes sur les variations de vitesse le long des interfaces. La principale difficulté de cette généralisation est que l'introduction de contraintes non-linéaires n'assure plus la definie positivité du hessien du lagrangien du problème non-linéaire (il faut prendre en compte la courbure des contraintes via le hessien du lagrangien). Une première piste facile à mettre en oeuvre serait de négliger la courbure des contraintes non-linéaires. Une deuxième piste plus compliquée consisterait à globaliser notre méthode SQP par des régions de confiance (voir point numéro 2). Cette globalisation autoriserait de pouvoir prendre en compte des problèmes où le hessien du lagrangien n'est pas défini positif.
2. La globalisation des régions de confiance pour résoudre le problème de tomographie avec contrainte permettrait, comme dans le cas sans contrainte, d'éviter de chercher une solution exacte du problème quadratique tangent (PQT) lorsque l'on se trouve encore "loin" de la solution (loin de la solution, le modèle quadratique n'est pas une bonne approximation de la fonction coût non-linéaire). La difficulté principale à surmonter dans cette voie est de concilier l'algorithme du LA avec la méthode des RC : il faut étudier la résolution approchée du problème de lagrange et la résolution approchée du PQT. Une autre difficulté, liée aux méthode de RC, est de contrôler la résolution approchée du PQT, de manière à ce que l'itéré suivant soit dans une RC prescrite.
3. L'application de notre méthode SQP sur d'autres problèmes inverses en géosciences dont les caractéristiques principales sont proches de la tomographie de réflexion, c'est à dire : la non-linéarité de l'opérateur physique, l'indétermination de la solution et les dimensions importantes de l'espace des données et de l'espace des modèles. On pourrait par exemple essayer d'appliquer notre méthode à l'inversion sismique (inversion des variations d'amplitude des ondes sismiques enregistrées pour retrouver les impédances sismiques), à l'inversion stratigraphique (pour déterminer la distribution spatiale des dépôts sédimentaires dans un bassin - échelle géologique) et à l'inversion de données de production aux puits (pour retrouver les porosités, perméabilités, etc... du réservoir).
4. Enfin, la comparaison des performances de notre méthode avec une méthode de points intérieurs permettrait de valider définitivement notre choix pour l'approche SQP.

Annexe A

Construction des B-splines

Une fonction B-spline cubique est définie comme la somme de polynômes du troisième degré. La forme générale d'une fonction B-spline cubique est :

$$\beta(x) = \sum_{m=1}^{N} \alpha_m \beta_m(x) \ , \tag{A.1}$$

où les α_m sont les paramètres de B-spline, N est le nombre de paramètres de B-spline et les β_m sont les fonctions de base de la représentation B-spline (figure A.1).

Les fonctions de base de la représentation B-spline cubique sont construites par morceaux et sont données par :

$$\beta_m(x) = \begin{cases} 0 & x \leq x_m \\ b_1(x_b) & x_m \leq x \leq x_{m+1} \\ b_2(x_b) & x_{m+1} \leq x \leq x_{m+2} \\ b_3(x_b) & x_{m+2} \leq x \leq x_{m+3} \\ b_4(x_b) & x_{m+3} \leq x \leq x_{m+4}, \\ 0 & x \geq x_{m+4} \end{cases} \tag{A.2}$$

où $x_b = \frac{x - x_m}{x_{m+1} - x_m}$ est l'abscisse relative $(0 \leq x_b \leq 1)$ d'un point dans l'intervalle $[x_m, x_{m+1}]$ et

$$\begin{aligned} b_1(x_b) &= \frac{1}{6} x_b^3 \\ b_2(x_b) &= -\frac{1}{2} x_b^3 + \frac{1}{2} x_b^2 + \frac{1}{2} x_b + \frac{1}{6} \\ b_3(x_b) &= b_2(1 - x_b) \\ b_4(x_b) &= b_1(1 - x_b) \ . \end{aligned} \tag{A.3}$$

Par construction, les dérivées premières et secondes d'une fonction B-spline cubique sont continues. De plus, les fonctions de base des fonctions B-spline cubiques ont un support fini et égal à $[x_m, x_{m+4}]$. Une fonction B-spline peut donc être évaluée localement, en utilisant seulement quatre fonctions de base et quatre paramètres.

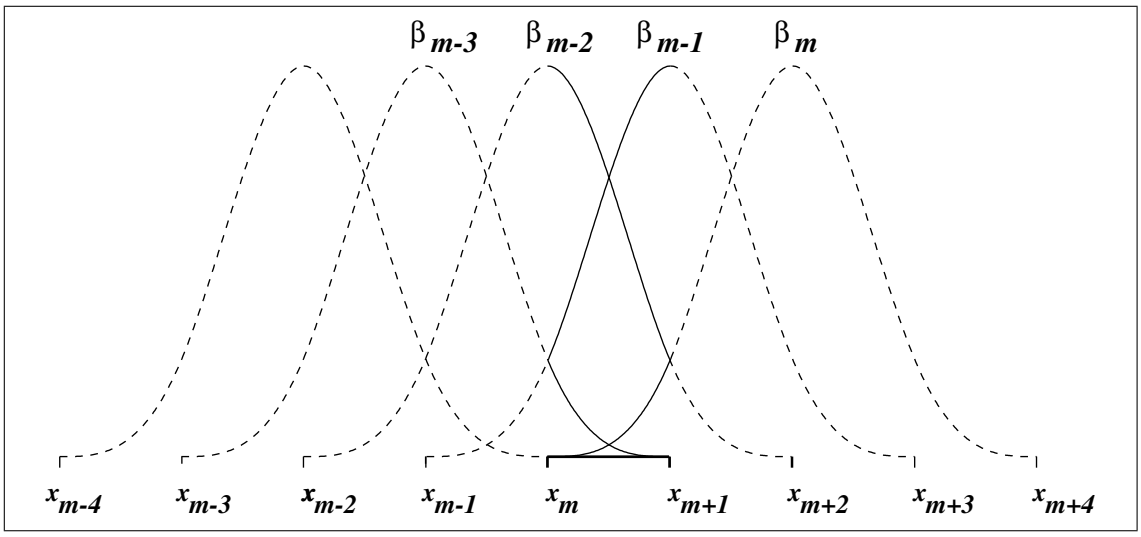

Fig. A.1 **Fonctions de base de la représentation B-spline cubique.** Seules les parties en trait plein des quatre fonctions de base voisines $\beta_{m-3},\ \beta_{m-2},\ \beta_{m-1}$, et β_m interviennent dans l'évaluation de la fonction B-spline dans l'intervalle $[x_m, x_{m+1}]$.

Annexe B

Expressions des distributions de vitesse, des géométries d'interface et de leurs dérivées par rapport aux coordonnées spatiales dans la représentation en fonctions B-spline

Nous donnons ci-dessous l'expressions d'une interface 2D (il facile obtenir les relations pour une interface 1D) ainsi que celles de ses dérivées à l'ordre 1 (on pourra facilement calculée les dérivées à l'ordre 2 en utilisant la même méthode) dans la représentation en fonction B-spline.

- Profondeur d'une interface Z_i :

$$Z_i(x,y) = \sum_{m_x=1}^{N_x^{Z_i}} \sum_{m_y=1}^{N_y^{Z_i}} m_{m_x,m_y}^{Z_i} B_{m_x}(x) B_{m_y}(y)$$

- Pente en x d'une interface Z_i :

$$\frac{\partial Z_i(x,y)}{\partial x} = \sum_{m_x=1}^{N_x^{Z_i}} \sum_{m_y=1}^{N_y^{Z_i}} m_{m_x,m_y}^{Z_i} \frac{\partial B_{m_x}(x)}{\partial x} B_{m_y}(y)$$

- Pente en y d'une interface Z_i :

$$\frac{\partial Z_i(x,y)}{\partial y} = \sum_{m_x=1}^{N_x^{Z_i}} \sum_{m_y=1}^{N_y^{Z_i}} m_{m_x,m_y}^{Z_i} B_{m_x}(x) \frac{\partial B_{m_y}(y)}{\partial y}$$

De même, nous donnons ci-dessous l'expression d'une vitesse 3D explicite en z (la généralisation à des vitesses 2D est facile à effectuer) ainsi que ses dérivées à l'ordre 1

(on obtiendra facilement les dérivées à l'ordre 2 en appliquant la même méthode) dans la représentation en fonction B-spline.

Vitesse d'une couche V_i :

$$V_i(x,y,z) = \sum_{m_x=1}^{N_x^{V_i}} \sum_{m_y=1}^{N_y^{V_i}} \sum_{m_z=1}^{N_z^{V_i}} m_{m_x,m_y,m_z}^{V_i} B_{m_x}(x) B_{m_y}(y) B_{m_z}(z)$$

Gradient de vitesse en x d'une couche V_i :

$$\frac{\partial V_i(x,y,z)}{\partial x} = \sum_{m_x=1}^{N_x^{V_i}} \sum_{m_y=1}^{N_y^{V_i}} \sum_{m_z=1}^{N_z^{V_i}} m_{m_x,m_y,m_z}^{V_i} \frac{\partial B_{m_x}(x)}{\partial x} B_{m_y}(y) B_{m_z}(z)$$

Gradient de vitesse en y d'une couche V_i :

$$\frac{\partial V_i(x,y,z)}{\partial y} = \sum_{m_x=1}^{N_x^{V_i}} \sum_{m_y=1}^{N_y^{V_i}} \sum_{m_z=1}^{N_z^{V_i}} m_{m_x,m_y,m_z}^{V_i} B_{m_x}(x) \frac{\partial B_{m_y}(y)}{\partial y} B_{m_z}(z)$$

Gradient de vitesse en z d'une couche V_i :

$$\frac{\partial V_i(x,y,z)}{\partial z} = \sum_{m_x=1}^{N_x^{V_i}} \sum_{m_y=1}^{N_y^{V_i}} \sum_{m_z=1}^{N_z^{V_i}} m_{m_x,m_y,m_z}^{V_i} B_{m_x}(x) B_{m_y}(y) \frac{\partial B_{m_z}(z)}{\partial z}$$

Annexe C

Convergence linéaire globale d'un algorithme de lagrangien augmenté pour résoudre des problèmes d'optimisation quadratiques convexes

L'article "Global linear convergence of an augmented Lagrangian algorithm for solving convex quadractic optimization problems", présenté dans la suite de cette annexe, va être publié en 2005 dans le journal "Journal of Convex Analysis". Il présente la méthode classique du lagrangien augmenté (LA) développé par Hesteness et Powell pour résoudre des problèmes quadratiques convexes avec contraintes d'inégalité linéaires. L'objectif de ce papier est de montrer que, au cours des itérations de LA, pour un paramètre d'augmentation suffisamment grand, la norme des contraintes converge linéairement globalement vers zéro. La principale difficulté réside dans l'obtention d'une propriété de continuité lipschitzienne de l'inverse du subdifférential de la fonction duale. Cette propriété est démontré en étudiant les relations entre la projection d'un point $x \in \mathbb{R}^n_+$ sur un sous-espace affine $\mathcal{A}$ avec sa projection sur le polyhèdre $\mathcal{A} \cap \mathbb{R}^n_+$. De plus, cet article illustre le résultat de convergence par de nombreuses expériences numériques. Les implications algorithmiques de ce résultat sont discutées.

Global linear convergence of an augmented Lagrangian algorithm for solving convex quadratic optimization problems

Frédéric DELBOS[†] and J. Charles GILBERT[‡]

February 25, 2004 (revised version)

Abstract. We consider an augmented Lagrangian algorithm for minimizing a convex quadratic function subject to linear inequality constraints. Linear optimization is an important particular instance of this problem. We show that, provided the augmentation parameter is large enough, the constraint value converges *globally* linearly to zero. This property is viewed as a consequence of the proximal interpretation of the algorithm and of the global radial Lipschitz continuity of the reciprocal of the dual function subdifferential. This Lipschitz property is itself obtained by means of a lemma of general interest, which compares the distances from a point in the positive orthant to an affine space, on the one hand, and to the polyhedron given by the intersection of this affine space and the positive orthant, on the other hand. No strict complementarity assumption is needed. The result is illustrated by numerical experiments and algorithmic implications, including complexity issues, are discussed.

Key words. Augmented Lagrangian, convex quadratic optimization, distance to a polyhedron, error bound, global linear convergence, iterative complexity, linear constraints, proximal algorithm.

AMS subject classification. 49M29, 65K05, 90C05, 90C06, 90C20.

1 Introduction

Convex quadratic programs (QP) arise in their own right and as subproblems in some numerical algorithms for solving optimization problems. On the one hand, since no strictly convex assumption is made, the important family of linear optimization problems, with a zero quadratic term in their objective, enters this framework. On the other hand, the SQP algorithm decomposes a regularized constrained least-squares problem into a sequence of strictly convex QP's (see [10, 5] for an example in reflection tomography, which partly motivates this study; see [19, 2] for recent books describing the SQP algorithm). Finding efficient algorithms for solving this basic multi-faceted problem in all possible situations is an objective that has been pursued for decades (see for example the already 20 year old survey on quadratic programming in [20] and the monographs on interior point methods in [15, 7, 18, 30, 14, 29, 31, 2]).

[†]Institut Français du Pétrole, 1 & 4 avenue de Bois-Préau, 92852 Rueil-Malmaison, France; e-mail: `Frederic.Delbos@ifp.fr`.

[‡]INRIA Rocquencourt, BP 105, 78153 Le Chesnay Cedex, France; e-mail: `Jean-Charles.Gilbert@inria.fr`.

The convex quadratic problem we consider is written

$$\begin{cases} \min_x \frac{1}{2} x^\top Q x + q^\top x \\ l \leq Cx \leq u, \end{cases} \tag{1.1}$$

where Q is an $n \times n$ positive semi-definite symmetric matrix, $q \in \mathbb{R}^n$, C is $m \times n$, and the m-dimensional vectors l and u satisfy $l < u$ (i.e., $l_i < u_i$ for all indices $i = 1, \ldots, m$) and may have infinite components. With lower and upper bounds, problem (1.1) is close to what is actually implemented in numerical codes (see [6] for an example). We have not included equality constraints, like $Ax = b$, in (1.1) to make the presentation simple, but such constraints can be expressed like in (1.1) by adding two inequalities $Ax \leq b$ and $-Ax \leq -b$. Therefore, our analysis covers problems with equality constraints. Note that, since Q may be zero, (1.1) also modelizes linear optimization.

The method that we further explore in this paper fits into the class of dual approaches, since it is essentially the augmented Lagrangian (AL) method of Hestenes [11] and Powell [22] that is applied to the convex QP (1.1). This algorithm can be implemented in such a way that it does not require any matrix factorization. It is therefore appropriate when the problem is so large that such a factorization is impracticable or too much time consuming. This is a motivation for using the AL algorithm when the optimization problem deals with systems governed by partial differential equations [8]. In that case, however, a good preconditioner for the unavoidable conjugate gradient iterations must be available.

The version of the AL algorithm we analyze is defined on an equivalent form of (1.1) obtained by introducing an auxiliary variable $y \in \mathbb{R}^m$ [10]:

$$\begin{cases} \min_x \frac{1}{2} x^\top Q x + q^\top x \\ y = Cx \\ l \leq y \leq u. \end{cases} \tag{1.2}$$

The algorithm generates a sequence $\{\lambda_k\} \subset \mathbb{R}^m$ converging to some optimal multiplier associated with the equality constraint of (1.2). At each iteration, an auxiliary bound constrained QP has to be solved, so that the approach can be viewed as transforming (1.1) into a sequence of bound constrained convex quadratic subproblems. Two facts contribute to the possible success of this method. First, a bound constraint QP is much easier to solve than (1.1), which has general linear constraints (see [17, 9] and the references therein). Second, because of its dual and constraint convergence, the AL algorithm usually identifies the active constraints of (1.1) in a finite number of iterations. Since often these constraints are also the active constraints of the subproblems close to the solution, the combinatorial aspect of the bound constrained QP's rapidly decreases in intensity as the convergence progresses (and usually disappears after finitely many AL iterations). This reasoning is valid, for instance, when Q is positive definite and strict complementarity holds at the solution.

The AL algorithm also generates primal iterates $(x_k, y_k) \in \mathbb{R}^n \times \mathbb{R}^m$ and is controlled by the convergence of the constraint values to zero: if $\|y_k - Cx_k\|$ is less than a given tolerance, optimality can be considered to be reached. The algorithm is also driven

by a so-called augmentation parameter r_k, whose role on the speed of convergence is major. This paper essentially shows that, provided r_k is larger than a certain positive threshold, the convergence of the constraint norm to zero is *globally* linear, meaning that *at each iteration* the constraint norm decreases by a factor uniformly less than one. This property makes predictable the number of iterations to converge to a given precision and offers a possibility to study the global iterative complexity of the algorithm.

The paper is organized as follows. In section 2, the AL algorithm under investigation is presented with the appropriate level of details. In section 3, we give the tools from convex analysis that are useful for the study of the method. We already set out some of the properties of the algorithm. This section also gives a lemma of general interest, which compares the distances from a point in the positive orthant to an affine space, on the one hand, and to the polyhedron given by the intersection of this affine space and the positive orthant, on the other hand. Section 4 deals with the global linear convergence of the AL algorithm. It starts by showing a global error bound for the dual solution set, in terms of the subgradient of the dual function. The global linear convergence is then seen as an easy consequence of this property. We conclude in section 5 by relating some numerical experiments on a seismic tomography problem and by a discussion on algorithmic consequences.

Notation

We denote the Euclidean norm by $\|\cdot\|$. The distance associated with this norm is denoted by "dist", $B := \{x : \|x\| \leq 1\}$ is the closed unit ball, and $\partial B := \{x : \|x\| = 1\}$ is the unit sphere. We note $\bar{\mathbb{R}} := \mathbb{R} \cup \{-\infty, +\infty\}$. The nonnegative orthant of $\mathbb{R}^n$ is denoted by $\mathbb{R}^n_+ := \{x \in \mathbb{R}^n : x \geq 0\}$, while $\mathbb{R}^n_{++} := \{x \in \mathbb{R}^n : x > 0\}$. The null space and range space of a matrix A are respectively denoted by $N(A)$ and $R(A)$. We write $A \succcurlyeq 0$ [resp. $A \succ 0$] to indicate that a symmetric matrix A is positive semi-definite [resp. positive definite].

Let E be a finite dimensional Euclidean space. The indicator function of a set $S \subset E$ is denoted by $\mathcal{I}_S$ (this is the function that vanishes on S and takes the value $+\infty$ outside S). The domain of a function $f : E \to \mathbb{R} \cup \{+\infty\}$ is defined by $\operatorname{dom} f := \{x \in E : f(x) < +\infty\}$ and its epigraph by $\operatorname{epi} f := \{(x, \alpha) \in E \times \mathbb{R} : f(x) \leq \alpha\}$. As in [12], $\overline{\operatorname{Conv}}(E)$ is the set of functions $f : E \to \mathbb{R} \cup \{+\infty\}$ that are convex ($\operatorname{epi} f$ is convex), proper ($\operatorname{epi} f \neq \emptyset$), and closed ($\operatorname{epi} f$ is closed). The subdifferential at $x \in E$ of a proper convex function $f : E \to \mathbb{R} \cup \{+\infty\}$ is denoted by $\partial f(x)$. We denote by $N_C(x)$ the normal cone at x to a convex set $C \subset E$. The orthogonal projection of a point x onto a nonempty closed convex set C is denoted by $P_C(x)$.

2 An AL algorithm for solving the QP

We **assume throughout that problem (1.2) has a solution** and denote by $\mathcal{S}_P$ the set of its solutions $(\bar{x}, \bar{y})$. The projections of $\mathcal{S}_P$ onto $\mathbb{R}^n$ and $\mathbb{R}^m$ are respectively denoted by $\mathcal{S}_P^x := \{\bar{x} \in \mathbb{R}^n : (\bar{x}, \bar{y}) \in \mathcal{S}_P \text{ for some } \bar{x} \in \mathbb{R}^n\}$ (this is also $\{\bar{x} \in \mathbb{R}^n : (\bar{x}, C\bar{x}) \in \mathcal{S}_P\}$, the solution set of (1.1)) and $\mathcal{S}_P^y := \{\bar{y} \in \mathbb{R}^m : (\bar{x}, \bar{y}) \in \mathcal{S}_P \text{ for some}$

$\bar{x} \in \mathbb{R}^n\}$. Since the constraints of (1.2) are qualified, there exist optimal multipliers, which certainly implies that the affine subspace

$$\Lambda := \{\lambda \in \mathbb{R}^m : C^\top \lambda \in q + R(Q)\} \tag{2.1}$$

is nonempty. Note that $\Lambda = \mathbb{R}^m$ if $Q \succ 0$.

The augmented Lagrangian is obtained by dualizing the equality constraint of (1.2). It is the function $\ell_r : (x, y, \lambda) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^m \mapsto \mathbb{R}$, defined by

$$\ell_r(x, y, \lambda) = \frac{1}{2} x^\top Q x + q^\top x + \lambda^\top (y - Cx) + \frac{r}{2} \|y - Cx\|^2, \tag{2.2}$$

where $r \geq 0$ is called the *augmentation parameter* (see [1, 11, 22]).

We can now give a precise statement of the AL algorithm we study in this paper, which is basically the method of Hestenes [11] and Powell [22] applied to (1.2).

AL ALGORITHM for solving (1.1)

Initialization: choose $\lambda_0 \in \mathbb{R}^m$ and $r_0 > 0$.

Repeat for $k = 0, 1, 2, \ldots$

1. Solve:
$$\min_{\substack{x \\ l \leq y \leq u}} \ \ell_{r_k}(x, y, \lambda_k). \tag{2.3}$$
Denote a solution by (x_{k+1}, y_{k+1}).
2. Update the multiplier
$$\lambda_{k+1} = \lambda_k + r_k(y_{k+1} - Cx_{k+1}). \tag{2.4}$$
3. Stop if $y_{k+1} \simeq Cx_{k+1}$.
4. Choose a new augmentation parameter: $r_{k+1} > 0$.

This algorithm deserves some comments.

1. Under the sole assumption that problem (1.1) has a solution, the QP in step (2.3) has also a solution. This fact is clarified in proposition 3.3. This solution is not necessary unique however.
2. Even though (x_{k+1}, y_{k+1}) is not uniquely determined as a solution to (2.3), $y_{k+1} - Cx_{k+1}$ is independent of that solution, so that the multipliers λ_k are unambiguously generated.
3. The augmentation parameter r_k can change from iteration to iteration, but the same value must be used in the AL minimized in step 1 and in the multiplier update in step 2. If the "step-size" in (2.4) is different from r_k (with the aim at minimizing better the dual function, as in [21, section 4.2] for example), several properties of the AL algorithm may no longer hold, such as the finite identification of active constraints (in the presence of strict complementarity) and the global linear convergence of section 4.

4. The larger are the augmentation parameters r_k, the faster is the convergence. The only limitation on a large value for r_k comes from the ill-conditioning that such a value induces in the AL and the resulting difficulty in solving (2.3). Actually, it is clear from the structure of the AL in (2.2) that a large r gives priority to the restoration of the equality constraint, leaving aside the minimization of the Lagrangian (whose role is to provide optimality).

In comparison with an interior point method, which faces the combinatorial aspect of (1.1) by transforming the problem into a sequence of linear systems, the AL algorithm goes around this difficulty by transforming a general QP into a sequence of QP's with simple bounds, which are easier to solve. Indeed, a number of efficient algorithms are available for dealing with the bound constraints on the AL in step 1. A possibility would be to minimize first analytically ℓ_r in y and then to minimize the resulting function in x. Unfortunately this function of x, which is the AL associated with the inequality constrained QP (1.1) [24, 26], has a combinatorial structure (it contains maxima) that is not easier to deal with than the direct numerical minimization of ℓ_r in (x, y) with bounds on y. In our code QPAL [6], used for the numerical experiments of section 5 and in [5], we have adapted to the (x, y) structure of problem (1.2) an active set strategy together with the gradient projection algorithm and conjugate gradient iterations on the activated faces (see [17, 9] and the references therein).

As opposed to standard (non shifted) interior point methods, whose elementary linear systems have an exploding condition number, the AL algorithm does not require the penalty parameter r_k to go to infinity. Actually, any sequence $\{r_k\}$ that remains bounded away from zero guarantees the convergence, even though large values speed it up [28]. Therefore, the bound constrained QP's in step 1 can be maintained reasonably well conditioned, keeping satisfactory the efficiency of a conjugate gradient based solver. This remark reinforces the viewpoint that considers the AL algorithm as a method suitable for large problems.

3 Convex analysis tools

3.1 Duality

As a dual function associated with problem (1.2), we use the one obtained by dualizing its equality constraints. It is the function $\delta : \mathbb{R}^m \to \mathbb{R} \cup \{+\infty\}$ defined by

$$\lambda \mapsto \delta(\lambda) := - \inf_{\substack{x \\ y \in [l,u]}} \left(\frac{1}{2} x^\top Q x + q^\top x + \lambda^\top (y - Cx) \right). \tag{3.1}$$

Clearly $\delta \in \overline{\mathrm{Conv}}(\mathbb{R}^m)$ (it takes a finite value, for instance, when λ is an optimal multiplier associated with the equality constraint of (1.2)).

For a given $\lambda \in \mathbb{R}^m$, (x_λ, y_λ) is a solution to the *Lagrange problem*, the minimization problem in (3.1), if and only if $x_\lambda \in X_\lambda$ and $y_\lambda \in Y_\lambda$, where X_λ is the affine space

$$X_\lambda := \{x \in \mathbb{R}^n : Qx = C^\top \lambda - q\} \tag{3.2}$$

and Y_λ is the Cartesian product of the following intervals

$$(Y_\lambda)_i = \begin{cases} [u_i, u_i] & \text{if } \lambda_i < 0 \\ [l_i, u_i] & \text{if } \lambda_i = 0 \\ [l_i, l_i] & \text{if } \lambda_i > 0. \end{cases} \tag{3.3}$$

These intervals, with their possible infinite bounds, have to be understood in a broad sense: for example, $[l_i, u_i]$ is the interval $]-\infty, u_i]$ if $l_i = -\infty$ and u_i is finite, $[l_i, l_i]$ is the empty set if $l_i = -\infty$, *etc.* Since the Lagrange problem is always feasible and since a feasible convex quadratic problem has a solution if and only if it is bounded (see [2, theorem 17.1] for example), the domain of δ is the set of λ's for which the Lagrange problem has a solution. Therefore $\operatorname{dom}\delta$ can be written as the nonempty polyhedron

$$\operatorname{dom}\delta = \{\lambda \in \mathbb{R}^m : X_\lambda \neq \emptyset,\ Y_\lambda \neq \emptyset\} = \mathbb{R}^m_{l,u} \cap \Lambda,$$

where we used the fact that $X_\lambda \neq \emptyset$ if and only if $\lambda \in \Lambda$ and the notation

$$\mathbb{R}^m_{l,u} := \{\lambda \in \mathbb{R}^m : \lambda_i \leq 0 \text{ if } l_i = -\infty,\ \lambda_i \geq 0 \text{ if } u_i = +\infty\}.$$

Observe finally that the multivalued function $\lambda \mapsto -Y_\lambda$ is monotone: for λ and $\lambda' \in \mathbb{R}^m$, and for $y_\lambda \in Y_\lambda$ and $y_{\lambda'} \in Y_{\lambda'}$, there holds

$$-(y_{\lambda'} - y_\lambda)^\top(\lambda' - \lambda) \geq 0. \tag{3.4}$$

Let $Q^\dagger$ be the pseudo-inverse of Q and take the notation

$$H := CQ^\dagger C^\top \qquad \text{and} \qquad v := CQ^\dagger q. \tag{3.5}$$

Let $\lambda \in \operatorname{dom}\delta$. Then, the minimization in x of the Lagrangian in (3.1) is achieved by any $x_\lambda \in X_\lambda$, hence satisfying $Qx_\lambda + q = C^\top\lambda$. This minimizer is not necessarily unique and we can take the minimum norm minimizer $x^\dagger_\lambda := Q^\dagger(C^\top\lambda - q)$. After substitution in the function minimized in (3.1) and the use of $Q^\dagger Q Q^\dagger = Q^\dagger$, one gets

$$\delta(\lambda) = \sup_{y \in [l,u]} \left(\frac{1}{2}\lambda^\top H\lambda - (v+y)^\top\lambda + \frac{1}{2}q^\top Q^\dagger q \right), \quad \text{for } \lambda \in \operatorname{dom}\delta. \tag{3.6}$$

On its domain, the dual function δ is therefore the maximum of a finite number of convex quadratic functions (those defined by the argument of the supremum in (3.6) with the components of y set to l_i or u_i; only the finite values of these bounds must be considered), which only differ by their slope at the origin (in particular, they have the same Hessian H).

Lemma 3.1 *The subdifferential of the dual function* (3.1) *is given at* $\lambda \in \operatorname{dom}\delta$ *by*

$$\partial\delta(\lambda) = H\lambda - v - Y_\lambda + C(N(Q)).$$

PROOF. We write δ as the sum of three convex functions. Let $\tilde{l}$ and $\tilde{u} \in \mathbb{R}^m$ be chosen such that $\tilde{l} < \tilde{u}$, $\tilde{l}_i = l_i$ if l_i is finite, and $\tilde{u}_i = u_i$ if u_i is finite. Define $\tilde{Y}_\lambda$ by formula (3.3) with l and u respectively replaced by $\tilde{l}$ and $\tilde{u}$. Then the following *finite* value function $\tilde{\delta} \in \overline{\text{Conv}}(\mathbb{R}^m)$ is identical to the right hand side of (3.6) on $\mathbb{R}^m_{l,u}$:

$$\tilde{\delta}(\lambda) = \sup_{\substack{y=(y_i)_{i=1}^m \\ y_i = \tilde{l}_i \text{ or } \tilde{u}_i}} \left(\frac{1}{2}\lambda^\top H\lambda - (v+y)^\top\lambda + \frac{1}{2}q^\top Q^\dagger q \right).$$

Clearly $\delta = \tilde{\delta} + \mathcal{I}_{\mathbb{R}^m_{l,u}} + \mathcal{I}_\Lambda$, so that theorem 23.8 in [23] implies that for $\lambda \in \operatorname{dom}\delta$:

$$\partial\delta(\lambda) = \partial\tilde{\delta}(\lambda) + \partial\mathcal{I}_{\mathbb{R}^m_{l,u}}(\lambda) + \partial\mathcal{I}_\Lambda(\lambda).$$

Equality holds above because $\mathcal{I}_{\mathbb{R}^m_{l,u}}$ and $\mathcal{I}_\Lambda$ are polyhedral and because ri $\operatorname{dom}\tilde{\delta}$ $(= \mathbb{R}^m$, ri denotes the relative interior), $\operatorname{dom}\mathcal{I}_{\mathbb{R}^m_{l,u}} = \mathbb{R}^m_{l,u}$, and $\operatorname{dom}\mathcal{I}_\Lambda = \Lambda$ have a point in common (one in $\operatorname{dom}\delta \neq \emptyset$).

To compute $\partial\tilde{\delta}(\lambda)$, we use corollary VI.4.3.2 in [12] (conv denotes the convex hull):

$$\partial\tilde{\delta}(\lambda) = \operatorname{conv}\left\{ H\lambda - v - y : y \in \tilde{Y}_\lambda \text{ and } (y_i = \tilde{l}_i \text{ or } \tilde{u}_i \text{ if } \lambda_i = 0) \right\} = H\lambda - v - \tilde{Y}_\lambda.$$

On the other hand, $\partial\mathcal{I}_{\mathbb{R}^m_{l,u}}(\lambda) = N_{\mathbb{R}^m_{l,u}}(\lambda)$, which is the set of vectors $\nu \in \mathbb{R}^m$ satisfying

$$\begin{cases} \nu_i \geq 0 & \text{when } \lambda_i = 0,\ l_i = -\infty, \text{ and } u_i \text{ is finite} \\ \nu_i \leq 0 & \text{when } \lambda_i = 0,\ l_i \text{ is finite, and } u_i = +\infty \\ \nu_i \in \mathbb{R} & \text{when } \lambda_i = 0,\ l_i = -\infty, \text{ and } u_i = +\infty \\ \nu_i = 0 & \text{when } \lambda_i \neq 0. \end{cases}$$

We deduce from this computation that for $\lambda \in \operatorname{dom}\delta$

$$\tilde{Y}_\lambda - \partial\mathcal{I}_{\mathbb{R}^m_{l,u}}(\lambda) = Y_\lambda.$$

Finally $\partial\mathcal{I}_\Lambda(\lambda) = N_\Lambda(\lambda) = \{\mu \in \mathbb{R}^m : C^\top\mu \in R(Q)\}^\perp = C(N(Q))$. Adding the last three subdifferentials provides the formula of $\partial\delta(\lambda)$ given in the statement of the lemma. □

Let us denote by $\mathcal{S}_D$ the set of dual solutions:

$$\mathcal{S}_D := \{\bar{\lambda} \in \mathbb{R}^m : 0 \in \partial\delta(\bar{\lambda})\}.$$

Not surprisingly, this is a convex polyhedron, which can be described in the standard form. It will be useful to make this form explicit and we do so in lemma 3.2 below. For this, we take a partition of $\{1, \ldots, m\}$ into the index sets

$$\begin{aligned} I_l &:= \{i : \bar{y}_i = l_i \text{ for all } (\bar{x}, \bar{y}) \in \mathcal{S}_P\}, \\ J &:= \{i : l_i < \bar{y}_i < u_i \text{ for some } (\bar{x}, \bar{y}) \in \mathcal{S}_P\}, \\ I_u &:= \{i : \bar{y}_i = u_i \text{ for all } (\bar{x}, \bar{y}) \in \mathcal{S}_P\}. \end{aligned} \tag{3.7}$$

We also introduce the orthant face $\mathcal{O}$ and the affine subspace $\mathcal{A}$

$$\begin{aligned} \mathcal{O} &:= \{\lambda \in \mathbb{R}^m : \lambda_{I_l} \geq 0,\ \lambda_J = 0,\ \lambda_{I_u} \leq 0\}, \\ \mathcal{A} &:= \{\lambda \in \mathbb{R}^m : C^\top \lambda = Q\bar{x} + q\}. \end{aligned} \tag{3.8}$$

In the definition of $\mathcal{A}$, $\bar{x}$ is an arbitrary primal solution. We have not made this dependence explicit in the symbol of the set since, as shown in the proof of the next lemma, $\mathcal{A}$ does not depend on the choice of $\bar{x} \in \mathcal{S}_P^x$.

Lemma 3.2 *The set of dual solutions $\mathcal{S}_D$ can be written as the intersection*

$$\mathcal{S}_D = \mathcal{O} \cap \mathcal{A}.$$

Furthermore, for any $\bar{\lambda} \in \mathcal{S}_D$ and any $\bar{y} \in \mathcal{S}_P^y$, we have $\bar{y} \in Y_{\bar{\lambda}}$ and $H\bar{\lambda} = v + \bar{y} + C\bar{u}$ for some $\bar{u} \in N(Q)$.

PROOF. Let $\ell(x, y, \lambda) = \frac{1}{2} x^\top Q x + q^\top x + \mathcal{I}_{[l,u]}(y) + \lambda^\top (y - Cx)$ be the Lagrangian function of the problem $\min_{(x,y)}\{\frac{1}{2} x^\top Q x + q^\top x + \mathcal{I}_{[l,u]}(y) : y = Cx\}$, which has the same dual function as problem (1.2). Since the constraint of this problem is qualified, $\bar{\lambda} \in \mathcal{S}_D$ if and only if $0 \in \partial_{(x,y)} \ell(\bar{x}, \bar{y}, \bar{\lambda})$, where $(\bar{x}, \bar{y})$ is an arbitrary primal solution. This can also be written $Q\bar{x} + q = C^\top \bar{\lambda}$ and $0 \in N_{[l,u]}(\bar{y}) + \bar{\lambda}$, which is equivalent to $\bar{\lambda} \in \mathcal{A}_{\bar{x}} \cap \mathcal{O}_{\bar{y}}$, where

$$\begin{aligned} \mathcal{A}_{\bar{x}} &:= \{\lambda \in \mathbb{R}^m : C^\top \lambda = Q\bar{x} + q\}, \\ \mathcal{O}_{\bar{y}} &:= \{\lambda \in \mathbb{R}^m : \lambda_i \geq 0 \text{ if } \bar{y}_i = l_i,\ \lambda_i = 0 \text{ if } l_i < \bar{y}_i < u_i,\ \lambda_i \leq 0 \text{ if } \bar{y}_i = u_i\}. \end{aligned}$$

By varying $(\bar{x}, \bar{y}) \in \mathcal{S}_P$, we see that $\mathcal{A}_{\bar{x}} = \mathcal{A}$ is independent of the chosen primal solution $\bar{x} \in \mathcal{S}_P^x$ and that $\bar{\lambda} \in \cap\{\mathcal{O}_{\bar{y}} : \bar{y} \in \mathcal{S}_P^y\} = \mathcal{O}$.

For proving the second part of the lemma, take $\bar{\lambda} \in \mathcal{S}_D$ and $(\bar{x}, \bar{y}) \in \mathcal{S}_P$. We have shown that $\bar{\lambda} \in \mathcal{A}_{\bar{x}} \cap \mathcal{O}_{\bar{y}}$. Actually, $\bar{\lambda} \in \mathcal{O}_{\bar{y}}$ is equivalent to $\bar{y} \in Y_{\bar{\lambda}}$. By $\bar{\lambda} \in \mathcal{A}_{\bar{x}}$, we have that $C^\top \bar{\lambda} = Q\bar{x} + q$. Multiplying to the left both sides of this equation by $CQ^\dagger$ provides $H\bar{\lambda} = v + \bar{y} + C\bar{u}$, where $\bar{u} := (Q^\dagger Q - I)\bar{x} \in N(Q)$. □

The fact observed in the proof above that the gradient of the criterion of the primal problem at a solution, here $Q\bar{x} + q$, is independent of the chosen solution is a property of general convex problems; see [16, 3]. This fact can also be deduced from the property that the subdifferential of a convex function (here the criterion of problem (1.1)) is constant on the relative interior of a set on which this function is constant (here the solution set).

3.2 Proximality

We will use the fundamental result of Rockafellar [25], according to which the AL algorithm of section 2 is the proximal algorithm on the dual function δ. More precisely, the multiplier λ_{k+1} computed in step 2 of the AL algorithm is also the unique solution to

$$\inf_{\lambda \in \mathbb{R}^m} \left(\delta(\lambda) + \frac{1}{2r_k} \|\lambda - \lambda_k\|^2 \right). \tag{3.9}$$

The same parameter $r_k > 0$ is used above and in (2.3). In addition, the optimal value of this problem is the opposite of the optimal value of problem (2.3). The optimality conditions of problem (3.9) can be written $0 \in \partial\delta(\lambda_{k+1}) + (\lambda_{k+1} - \lambda_k)/r_k$. Using (2.4), we see that:

$$Cx_k - y_k \in \partial\delta(\lambda_k), \quad \forall k \geq 1. \tag{3.10}$$

Note that, since λ_{k+1} is uniquely determined as the solution to (3.9), this is also the case for $y_{k+1} - Cx_{k+1}$, even though x_{k+1} and y_{k+1} are not uniquely determined.

Let us now clarify the conditions ensuring that the augmented Lagrange problem (2.3) has a solution.

Proposition 3.3 *The following three properties are equivalent:*

(i) $\operatorname{dom}\delta \neq \emptyset$,

(ii) problem (1.1)*, with some (or any) finite shift of its finite bounds to make it feasible, has a solution,*

(iii) for some (or any) $r_k > 0$ *and* $\lambda_k \in \mathbb{R}^m$*, problem* (2.3) *has a solution.*

PROOF. $[(i) \Rightarrow (iii)]$ Fix $r_k > 0$ and $\lambda_k \in \mathbb{R}^m$ (not necessarily the kth iterate). Since $\operatorname{dom}\delta \neq \emptyset$, the optimal value of (3.9) is finite, so that the optimal value of problem (2.3) is also finite. As a feasible bounded convex quadratic problem, (2.3) must have a solution [2, theorem 17.1].

$[(iii) \Rightarrow (ii)]$ We proceed by contradiction. Suppose that $\tilde{l}$ and $\tilde{u} \in \bar{\mathbb{R}}^m$ are such that $\tilde{l} < \tilde{u}$, $\tilde{l}_i = -\infty$ iff $l_i = -\infty$, $\tilde{u}_i = +\infty$ iff $u_i = +\infty$, and $[\tilde{l}, \tilde{u}] \cap R(C) \neq \emptyset$ (these bounds $\tilde{l}$ and $\tilde{u}$ result from a finite shift of the finite bounds of (1.1) that makes this problem feasible) and assume that the feasible problem $\min\{f(x) : \tilde{l} \leq Cx \leq \tilde{u}\}$, where $f(x) := (1/2)x^\top Qx + q^\top x$, has no solution. Then, there exists a sequence $\{x_j\}$ such that $Cx_j \in [\tilde{l}, \tilde{u}]$ and $f(x_j) \to -\infty$ when $j \to \infty$ (this is because a bounded feasible convex quadratic problem has a solution). Let $y_j := P_{[l,u]}(Cx_j)$ be the projection of Cx_j onto $[l, u]$. Then $\|y_j - Cx_j\| \leq m^{1/2}\|y_j - Cx_j\|_\infty \leq m^{1/2}\gamma$, where $\gamma := \max(\|\tilde{l} - l\|_\infty, \|\tilde{u} - u\|_\infty)$ (these norms are taken on the finite components of l and u), and (x_j, y_j) is feasible for problem (2.3). On the other hand, for an arbitrary $r_k > 0$ and $\lambda_k \in \mathbb{R}^m$, $\ell_{r_k}(x_j, y_j, \lambda_k) \leq f(x_j) + m^{1/2}\gamma\|\lambda_k\| + (r_k/2)m\gamma^2 \to -\infty$ when $j \to \infty$. Therefore problem (2.3) has no solution.

$[(ii) \Rightarrow (i)]$ Let $\tilde{l}$ and $\tilde{u}$ be some finite shifts of the finite bounds of problem (1.1), such that the problem $\min\{f(x) : \tilde{l} \leq Cx \leq \tilde{u}\}$, with f as in the previous paragraph, has a solution, $\tilde{x}$ say. Since its constraints are qualified, there exist $\tilde{\lambda}^l$ and $\tilde{\lambda}^u$ such that $Q\tilde{x} + q = C^\top(\tilde{\lambda}^l - \tilde{\lambda}^u)$, $\tilde{\lambda}^l \geq 0$, $\tilde{\lambda}^u \geq 0$, $\tilde{\lambda}^l_i = 0$ if $l_i = -\infty$, and $\tilde{\lambda}^u_i = 0$ if $u_i = +\infty$. It is easy to check that $\tilde{\lambda}^l - \tilde{\lambda}^u \in \mathbb{R}^m_{l,u} \cap \Lambda = \operatorname{dom}\delta$. □

If the original quadratic problem (1.1) has a solution, condition *(ii)* above holds (without having to shift the bounds), so that the augmented Lagrange problem (2.3) has a solution.

3.3 Projection onto a convex polyhedron

This section gives two lemmas related to the projection onto a convex polyhedron. The first lemma has a general interest. It compares the distance from a point x in the positive orthant to a convex polyhedron $\mathcal{X}$ defined in the standard form and the distance from x to the underlying affine space $\mathcal{A}$. It is claimed that the second distance is bounded below by a positive constant (independent of $x \geq 0$) times the first one. Of course, since $\mathcal{X} \subset \mathcal{A}$, $\mathrm{dist}(x, \mathcal{A}) \leq \mathrm{dist}(x, \mathcal{X})$.

Lemma 3.4 *Let A be an $m \times n$ matrix and $b \in \mathbb{R}^m$. Consider the affine subspace $\mathcal{A}$ and the convex polyhedron $\mathcal{X}$ defined by*

$$\mathcal{A} := \{x \in \mathbb{R}^n : Ax = b\} \qquad \textit{and} \qquad \mathcal{X} := \{x \in \mathbb{R}^n : Ax = b,\ x \geq 0\}.$$

These sets are supposed to be nonempty. Then, there exists a constant $\gamma > 0$ such that

$$\forall x \in \mathbb{R}^n_+, \quad \mathrm{dist}(x, \mathcal{A}) \geq \gamma\, \mathrm{dist}(x, \mathcal{X}).$$

PROOF. *First stage: reformulation of the statement of the lemma.* By using the triangle inequality, it is easy to see that the conclusion of the lemma is equivalent to claim that

$$\exists \gamma' > 0, \quad \forall x \in \mathbb{R}^n_+, \quad \|x - P_{\mathcal{A}}(x)\| \geq \gamma' \|P_{\mathcal{A}}(x) - P_{\mathcal{X}}(x)\|, \tag{3.11}$$

This inequality suggests that a certain function (whose value is the left hand side of the inequality in (3.11) divided by the factor of γ' in the right hand side) has a positive slope. This is the strategy we follow to establish (3.11).

Instead of testing the validity of the inequality in (3.11) for any $x \in \mathbb{R}^n_+$, we consider all the possible points x^0 that are projections onto $\mathcal{X}$ of points in the positive orthant and reformulate (3.11). For a point $x^0 \in \mathcal{X}$ and a direction $d \in N_{\mathcal{X}}(x^0) \cap N(A) \cap \partial B$, let us introduce the function

$$\varphi : \alpha \in \mathbb{R}_{++} \mapsto \varphi(\alpha) := \inf \left\{ \|A^\top y\| : y \in \mathbb{R}^m,\ x^0 + \alpha d + A^\top y \geq 0 \right\}. \tag{3.12}$$

This function depends on the choice of x^0 and d, but we do not mention this dependence to keep the notation light. Let us show that (3.11) holds if

$$\exists \gamma' > 0, \quad \forall x^0 \in \mathcal{X}, \quad \forall d \in N_{\mathcal{X}}(x^0) \cap N(A) \cap \partial B, \quad \forall \alpha > 0, \quad \varphi(\alpha) \geq \gamma' \alpha. \tag{3.13}$$

Let $x \in \mathbb{R}^n_+$, $x^1 := P_{\mathcal{A}}(x)$, and $x^0 := P_{\mathcal{X}}(x)$. One can assume that $x^1 \neq x^0$ (since otherwise the inequality in (3.11) is trivially satisfied). Set $d := (x^1 - x^0)/\|x^1 - x^0\|$. It is clear that $d \in N(A) \cap \partial B$ (note that both x^1 and $x^0 \in \mathcal{A}$). To show that $d \in N_{\mathcal{X}}(x^0)$, observe that $x^1 := P_{\mathcal{A}}(x)$ implies that $x = x^1 + A^\top y = x^0 + \alpha d + A^\top y$ for $\alpha := \|x^1 - x^0\| > 0$ and a certain $y \in \mathbb{R}^m$. Then, for all $z \in \mathcal{X}$, there holds

$$d^\top (z - x^0) = \frac{1}{\alpha}(x - x^0 - A^\top y)^\top (z - x^0) = \frac{1}{\alpha}(x - x^0)^\top (z - x^0) \leq 0.$$

We have used the fact that $z-x^0 \in N(A)$ and that $x^0 := P_{\mathcal{X}}(x)$ to get the last inequality. This shows that $d \in N_{\mathcal{X}}(x^0)$. Using (3.13) and the fact that $x = x^0 + \alpha d + A^\top y \geq 0$, we see that the inequality in (3.11) holds:

$$\|x - x^1\| = \|A^\top y\| \geq \varphi(\alpha) \geq \gamma'\alpha = \gamma'\|x^1 - x^0\|.$$

The claim (3.13) can be simplified. Observe that $\varphi(\alpha) \geq 0$, that $\varphi(0) = 0$, and that $\varphi(t\alpha) \leq t\varphi(\alpha)$ when $\alpha \geq 0$ and $t \in]0,1]$. To prove this last property of φ, assume that $\varphi(\alpha) < \infty$ (otherwise, there is nothing to show). Then take $\varepsilon > 0$ and $y \in \mathbb{R}^m$ such that $x^0 + \alpha d + A^\top y \geq 0$ and $\|A^\top y\| \leq \varphi(\alpha) + \varepsilon$. Since $x^0 \geq 0$, there holds $0 \leq (1-t)x^0 + t(x^0 + \alpha d + A^\top y) = x^0 + t\alpha d + A^\top(ty)$ and therefore $\varphi(t\alpha) \leq \|A^\top(ty)\| \leq t\varphi(\alpha) + \varepsilon$. Since $\varepsilon > 0$ is arbitrary, there holds $\varphi(t\alpha) \leq t\varphi(\alpha)$. Now, this property of φ implies that $\alpha \in \mathbb{R}_{++} \mapsto \varphi(\alpha)/\alpha$ is nondecreasing. Therefore, we have reduced the problem to showing that

$$\exists \gamma' > 0, \quad \forall x^0 \in \mathcal{X}, \quad \forall d \in N_{\mathcal{X}}(x^0) \cap N(A) \cap \partial B, \quad \varphi'(0;1) \geq \gamma', \tag{3.14}$$

where $\varphi'(0;1)$ denotes the right derivative of φ at zero.

Second stage: control of the decomposition of the normal directions. Consider a point $x^0 \in \mathcal{X}$ having a unitary normal direction in the null space of A, say $d \in N_{x^0}(\mathcal{X}) \cap N(A) \cap \partial B$. Define $I := I(x^0) := \{i : x^0_i = 0\}$ and $J := J(x^0) := \{i : x^0_i > 0\}$. These directions d are characterized by the conditions

$$d = A^\top z - r, \quad r_I \geq 0, \quad r_J = 0, \quad Ad = 0, \quad \text{and} \quad \|d\| = 1, \tag{3.15}$$

for some vectors $z \in \mathbb{R}^m$ and $r \in \mathbb{R}^n$. The decomposition of d in $A^\top z - r$ as above is not necessarily unique. It will be useful to identify a decomposition that provides the smallest value to $\|A^\top z\|$, which is therefore a solution to

$$\begin{cases} \min_{(z,r)} \frac{1}{2}\|A^\top z\|^2 \\ A^\top z - r = d \\ r_I \geq 0 \\ r_J = 0. \end{cases}$$

It is easy to show that this problem has a solution, which is characterized by (3.15) and

$$A(A^\top z - s) = 0, \quad s_I \geq 0, \quad \text{and} \quad s_I^\top r_I = 0, \tag{3.16}$$

for some vector $s \in \mathbb{R}^n$.

Let us show that

$$\max_{x^0 \in \mathcal{X}} \sup_{\substack{d \in N_{\mathcal{X}}(x^0) \\ Ad = 0 \\ \|d\| = 1}} \min_{\substack{(z,r) \in \mathbb{R}^m \times \mathbb{R}^n \\ A^\top z - r = d \\ r_{I(x^0)} \geq 0 \\ r_{J(x^0)} = 0}} \|A^\top z\| \; < \; +\infty. \tag{3.17}$$

We see on (3.15) that two points $x^0 \in \mathcal{X}$ having the same index set I have the same normal cone. Therefore, the point $x^0 \in \mathcal{X}$ intervenes in (3.17) only through its index

sets I and J. Since there is a finite number of such sets, one can fix x^0, hence I and J. Let us continue by contradiction, assuming that there exists a sequence $\{(d^k, z^k, r^k, s^k)\}$ such that $Ad^k = 0$, $\|d^k\| = 1$, $A^\top z^k - r^k = d^k$, $r^k_I \geq 0$, $r^k_J = 0$, $A(A^\top z^k - s^k) = 0$, $s^k_I \geq 0$, $(s^k_I)^\top r^k_I = 0$, and $\|A^\top z^k\| \to \infty$. Extracting a subsequence if necessary, it can be assumed that $A^\top z^k / \|A^\top z^k\| \to A^\top \bar{z}$, a vector of unit norm. Since $\{d^k\}$ is bounded, the identity $A^\top z^k - r^k = d^k$ shows that $r^k / \|A^\top z^k\|$ converges to $\bar{r} := A^\top \bar{z}$. Multiplying the identity $A(A^\top z^k - s^k) = 0$ by $\bar{z}$, one finds for sufficiently large k

$$0 = \bar{z}^\top A A^\top z^k - \bar{r}^\top s^k = \bar{z}^\top A A^\top z^k,$$

because, when $\bar{r}_i > 0$, then $i \in I$ and, for all sufficiently large k, $r^k_i > 0$, so that $s^k_i = 0$. Dividing the right hand side by $\|A^\top z^k\|$ and taking the limit, one would find $A^\top \bar{z} = 0$, which provides the expected contradiction.

Third stage: lower bound for $\varphi'(0;1)$ and conclusion. Let us introduce $v : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, the value function of the problem

$$\left\{ \begin{array}{l} \inf_y \ \|A^\top y\| \\ x^0 + A^\top y \geq 0, \end{array} \right. \tag{3.18}$$

which is the proper convex function defined by $v(p) := \inf\{\|A^\top y\| : x^0 + A^\top y \geq p\}$. Then, for fixed $x^0 \in \mathcal{X}$ and $d \in N_{\mathcal{X}}(x^0) \cap N(A) \cap \partial B$, $\varphi(\alpha)$ defined by (3.12) can be written $\varphi(\alpha) = v(-\alpha d)$. Therefore

$$\varphi'(0;1) = v'(0;-d) \geq -g^\top d, \quad \forall g \in \partial v(0). \tag{3.19}$$

As for the subdifferential $\partial v(0)$, it is formed of the optimal multipliers associated with the constraint of (3.18), which are the g-parts of the pairs (g,u) satisfying

$$g \in u + N(A), \quad \|u\| \leq 1, \quad g \geq 0, \quad \text{and} \quad (x^0)^\top g = 0. \tag{3.20}$$

Let $d = A^\top z - r$ be a decomposition of d satisfying (3.15)-(3.16). If $A^\top z = 0$, then $g := -\alpha d = \alpha r$ is a subgradient of v at zero for any $\alpha \geq 0$ (the conditions (3.20) are satisfied with $u = 0$; recall that $d \in N(A)$), so that (3.19) shows that $\varphi'(0;1) = +\infty$. If $A^\top z \neq 0$, then $g := (A^\top z - d)/\|A^\top z\| = r/\|A^\top z\|$ is a subgradient of v at zero (the conditions (3.20) are satisfied with $u = A^\top z / \|A^\top z\|$). Then (3.19) shows that $\varphi'(0;1) \geq 1/\|A^\top z\|$. Since for these decompositions, stage 2 of the proof has shown that $A^\top z$ is bounded, (3.14) holds and, consequently, the result is proven. □

As shown by the following example, lemma 3.4 no longer holds with all its generality when $\mathcal{X}$ is the intersection of an affine space $\mathcal{A}$ and an arbitrary closed convex cone.

Example 3.5 Let us introduce the following closed convex cone $K := \{x \in \mathbb{R}^3 : x_2 x_3 \geq x_1^2,\ x_2 \geq 0,\ x_3 \geq 0\}$, the 1×3 matrix $A := (0\ 1\ 0)$, and $b = 0 \in \mathbb{R}$. Define the affine space $\mathcal{A}$ and its intersection with K by

$$\begin{aligned} \mathcal{A} &:= \{x \in \mathbb{R}^3 : Ax = b\} = \{x \in \mathbb{R}^3 : x_2 = 0\}, \\ \mathcal{X} &:= K \cap \mathcal{A} = \{x \in \mathbb{R}^3 : x_1 = x_2 = 0,\ x_3 \geq 0\}. \end{aligned}$$

Then the conclusion of lemma 3.4 does not hold for these sets $\mathcal{A}$ and $\mathcal{X}$. To see this, fix $x_1 > 0$ and consider the points $x^t := (x_1, x_1^2/t, t)$ for $t \uparrow +\infty$. Clearly $x^t \in K$, $P_{\mathcal{A}}(x^t) = (x_1, 0, t)$, and $P_{\mathcal{X}}(x^t) = (0, 0, t)$. Therefore $\|x^t - P_{\mathcal{A}}(x^t)\|/\|P_{\mathcal{A}}(x^t) - P_{\mathcal{X}}(x^t)\| = x_1/t$, which is not bounded away from zero. □

Actually, it will be useful below to have the following relaxed version of lemma 3.4. This one allows the projected point x not to belong to $\mathbb{R}^n_+$. This point must however be sufficiently close to the positive orthant with respect to its distance to $\mathcal{X}$.

Corollary 3.6 *Assume the framework defined in the statement of lemma* 3.4. *Then, there exist two constants $\tau > 0$ and $\gamma > 0$ such that for all $x \in \mathbb{R}^n$,*

$$\operatorname{dist}(x, \mathbb{R}^n_+) \leq \tau \operatorname{dist}(x, \mathcal{X}) \qquad \Longrightarrow \qquad \operatorname{dist}(x, \mathcal{A}) \geq \gamma \operatorname{dist}(x, \mathcal{X}).$$

PROOF. Let γ be the constant given by lemma 3.4 and set

$$\tau := \frac{\gamma}{4(1+\gamma)}.$$

Let x be such that $\operatorname{dist}(x, \mathbb{R}^n_+) \leq \tau \operatorname{dist}(x, \mathcal{X})$. To simplify the notation, let us define

$$x^0 := P_{\mathcal{X}}(x), \qquad x^1 := P_{\mathcal{A}}(x), \qquad \text{and} \qquad \bar{x} := P_{\mathbb{R}^n_+}(x),$$

Using several times the triangle inequality, lemma 3.4, the non-expansiveness of the projectors $P_{\mathcal{A}}$ and $P_{\mathcal{X}}$, and the definition of τ, one can write

$$\begin{aligned}
\|x - x^1\| &\geq \|\bar{x} - P_{\mathcal{A}}(\bar{x})\| - \|P_{\mathcal{A}}(\bar{x}) - P_{\mathcal{A}}(x)\| - \|x - \bar{x}\| \\
&\geq \gamma\|\bar{x} - P_{\mathcal{X}}(\bar{x})\| - 2\|x - \bar{x}\| \\
&\geq \gamma\|\bar{x} - x^0\| - (2+\gamma)\|x - \bar{x}\| \\
&\geq \gamma\|x - x^0\| - 2(1+\gamma)\|x - \bar{x}\| \\
&\geq \gamma\|x - x^0\| - 2\tau(1+\gamma)\|x - x^0\| \\
&= \frac{\gamma}{2}\|x - x^0\|.
\end{aligned}$$

This is the expected inequality. □

The following lemma will be also useful. If $I \subset \{1, \ldots, n\}$, we denote by I^c the complementary set of I in $\{1, \ldots, n\}$.

Lemma 3.7 *Let A be an $m \times n$ matrix, $b \in \mathbb{R}^m$, $I \subset \{1, \ldots, n\}$, and $\varphi : \mathbb{R}^n \to \mathbb{R}$ be a convex differentiable function. If $\bar{x}$ is a solution to the problem*

$$\min \left\{ \varphi(x) : Ax = b,\ x_I \geq 0,\ x_{I^c} = 0 \right\},$$

then there is a subset of indices $J \subset \{1, \ldots, n\}$, containing I, such that $\bar{x}$ is also a solution to the problem

$$\min \left\{ \varphi(x) : Ax = b,\ x_J \geq 0,\ x_{J^c} \leq 0 \right\}.$$

PROOF. The constraints of the first problem are affine, hence qualified. Therefore, there exist vectors $y \in \mathbb{R}^m$ and $s \in \mathbb{R}^n$ such that

$$\nabla\varphi(\bar{x}) + A^\top y + s = 0, \quad \bar{x}_I \geq 0, \quad s_I \leq 0, \quad s_I^\top \bar{x}_I = 0, \quad \bar{x}_{I^c} = 0.$$

Define

$$J := I \cup \{i \in I^c : s_i \leq 0\}.$$

Then

$$\begin{aligned} \nabla\varphi(\bar{x}) + A^\top y + s = 0, \quad & \bar{x}_J \geq 0, \quad s_J \leq 0, \quad s_J^\top \bar{x}_J = 0, \\ & \bar{x}_{J^c} \leq 0, \quad s_{J^c} \geq 0, \quad s_{J^c}^\top \bar{x}_{J^c} = 0. \end{aligned}$$

By convexity, these conditions suffice to show that $\bar{x}$ is also a solution to the second problem. □

4 Global linear convergence of the algorithm

The global linear convergence of the AL algorithm will be shown in section 4.2 to be a consequence of the radial Lipschitz continuity of the multifunction $\partial\delta^{-1}$, the reciprocal of the subdifferential of the dual function (this argument is taken from [28]). The latter property is the subject of section 4.1.

4.1 Global dual error bound

Two normed spaces E and F being given, a multifunction $T : E \twoheadrightarrow F$ is said to be *radially Lipschitz continuous at* $x_0 \in E$ *with constant* $L \geq 0$ if for all $x \in E$ and all $y \in T(x)$, there holds $\text{dist}(y, T(x_0)) \leq L\|x - x_0\|$ ("dist" denotes here the distance associated with the norm of F). Consider the multifunction

$$\partial\delta^{-1} : g \in \mathbb{R}^m \mapsto \{\lambda \in \mathbb{R}^m : g \in \partial\delta(\lambda)\} \subset \mathbb{R}^m,$$

where δ is the dual function defined in (3.1). Clearly $\partial\delta^{-1}(0) = \mathcal{S}_D$, the set of dual solutions. Then $\partial\delta^{-1}$ is radially Lipschitz continuous at 0 with constant $L \geq 0$, if

$$\forall \lambda \in \mathbb{R}^m, \ \forall g \in \partial\delta(\lambda) : \ \text{dist}(\lambda, \mathcal{S}_D) \leq L\|g\|. \tag{4.1}$$

Such a property is sometimes called a *global error bound* for the dual solution set $\mathcal{S}_D$ in terms of the dual function subgradient (see the review paper by Pang [21] and the contribution of Izmailov and Solodov [13]). In this section, we show that this property holds in a weaker form: λ has to stay at a bounded distance from $\mathcal{S}_D$ (the Lipschitz constant L depends on this distance). Nevertheless, this property still has a global nature, since λ is not required to be close to $\mathcal{S}_D$ and g is not required to be close to 0.

To show that this property is natural, consider first a quadratic problem with only equality constraints:

$$\begin{cases} \inf_x \ \frac{1}{2} x^\top Q x + q^\top x \\ Cx = b. \end{cases} \tag{4.2}$$

It is assumed that this problem is convex ($Q \succcurlyeq 0$) and has a solution. It is therefore feasible: $b \in R(C)$. Since the constraint is qualified, there exist optimal multipliers, which implies that the affine subspace Λ defined in (2.1) is nonempty.

Using the pseudo-inverse $Q^\dagger$ of Q, the symmetric matrix $H \succcurlyeq 0$, and the vector v defined in (3.5), the dual function δ associated with problem (4.2) can be written

$$\delta(\lambda) = \begin{cases} \frac{1}{2}\lambda^\top H\lambda - (v+b)^\top\lambda + \frac{1}{2}q^\top Q^\dagger q & \text{for } \lambda \in \Lambda \\ +\infty & \text{otherwise.} \end{cases} \tag{4.3}$$

A computation like in the proof of lemma 3.1 shows that

$$\partial\delta(\lambda) = H\lambda - v - b + C(N(Q)), \quad \text{for } \lambda \in \Lambda.$$

Since $\mathcal{S}_D$ is defined as the set of minimizers of δ, one finds

$$\mathcal{S}_D = \{\bar{\lambda} \in \Lambda : H\bar{\lambda} \in v + b + C(N(Q))\}.$$

It is useful to introduce

$$\sigma := \inf_{\substack{\mu \in \partial B \cap R(C) \\ C^\top\mu \in R(Q^\dagger)}} \mu^\top H\mu, \tag{4.4}$$

which, by definition, takes the value $+\infty$ when $\{\mu \in R(C) : C^\top\mu \in R(Q^\dagger)\} = \{0\}$. Below, the smallest *nonzero* eigenvalue of a *zero* matrix is defined to be $+\infty$.

Lemma 4.1 *The value in $\bar{\mathbb{R}}$ defined by (4.4) satisfies $\sigma > 0$. It is the smallest nonzero eigenvalue of H when $R(C^\top) \subset R(Q^\dagger)$.*

Proof. We only have to consider the case when $\{\mu \in R(C) : C^\top\mu \in R(Q^\dagger)\} \neq \{0\}$. Then, $Q^\dagger \neq 0$ (because $C^\top\mu = 0$ and $\mu \in R(C)$ imply that $\mu = 0$) and $C \neq 0$. Now, when $C^\top\mu \in R(Q^\dagger) = N(Q^\dagger)^\perp$, $\mu^\top H\mu = \mu^\top CQ^\dagger C^\top\mu \geq \zeta_{\min}(Q^\dagger)\|C^\top\mu\|^2$, where $\zeta_{\min}(Q^\dagger)$ is the smallest nonzero eigenvalue of $Q^\dagger$. On the other hand, when $\mu \in R(C)$, $\|C^\top\mu\| \geq \sigma_{\min}(C)\|\mu\|$, where $\sigma_{\min}(C)$ is the smallest nonzero singular value of C. We have shown that

$$\sigma \geq \zeta_{\min}(Q^\dagger)\,\sigma_{\min}(C)^2 > 0.$$

Suppose now that $R(C^\top) \subset R(Q^\dagger)$. Then $\sigma = \inf\{\mu^\top H\mu : \mu \in \partial B \cap R(C)\}$, so that σ will be the smallest nonzero eigenvalue of H if we show that $R(C) = N(H)^\perp$ or that $N(H) = N(C^\top)$. The inclusion $N(C^\top) \subset N(H)$ is clear. Conversely, let $\nu \in N(H)$, which reads $CQ^\dagger C^\top\nu = 0$. This implies that $C^\top\nu \in N(Q^\dagger) = R(Q^\dagger)^\perp \subset N(C)$, by assumption. Then $C^\top\nu = 0$. □

Note that when $R(C^\top) \not\subset R(Q^\dagger)$, σ is not the smallest nonzero eigenvalue of H. Here is an example

$$Q^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \qquad \text{and} \qquad C = \begin{pmatrix} 1 & 1 \end{pmatrix}.$$

Then $H = 1$, while $\sigma = +\infty$ since there is no nonzero μ such that $C^\top\mu \in R(Q^\dagger)$.

Proposition 4.2 *Consider problem* (4.2) *with* $Q \succcurlyeq 0$ *and suppose that it has a solution. Then property* (4.1) *is satisfied by the dual function* (4.3)*, with the Euclidean norm and a constant* $L = 1/\sigma$*, where* σ *is defined by* (4.4).

PROOF. Since problem (4.2) has a solution and its constraint is qualified, $\mathcal{S}_D$ is nonempty (it is identical to the set of optimal multipliers). To prove (4.1), we only have to consider the dual variables $\lambda \in \Lambda$, since otherwise $\partial\delta(\lambda)$ is empty. Note also that we only have to consider the case when $H \neq 0$ since otherwise $\mathcal{S}_D = \Lambda$ and (4.1) is trivially satisfied with $L = 0$.

Let $\lambda \in \operatorname{dom}\delta = \Lambda$, $g \in \partial\delta(\lambda)$, and $\bar{\lambda}$ be the projection of λ onto $\mathcal{S}_D$. We have for some u and $\bar{u} \in N(Q)$

$$g = H\lambda - (v + b) + Cu \qquad \text{and} \qquad 0 = H\bar{\lambda} - (v + b) + C\bar{u}.$$

Subtracting these two identities and taking the scalar product with $(\lambda - \bar{\lambda})$ yield

$$g^\top(\lambda - \bar{\lambda}) = (\lambda - \bar{\lambda})^\top H(\lambda - \bar{\lambda}) + (u - \bar{u})^\top C^\top(\lambda - \bar{\lambda}).$$

The last term vanishes, since $C^\top(\lambda - \bar{\lambda}) \in R(Q)$ and $Q(u - \bar{u}) = 0$. Now observe that $\lambda - \bar{\lambda} \in R(C)$, since $\bar{\lambda} + N(C^\top) \subset \mathcal{S}_D$ (equality holds actually), and that $C^\top(\lambda - \bar{\lambda}) \in R(Q) = R(Q^\dagger)$, since both λ and $\bar{\lambda} \in \Lambda$. Therefore

$$g^\top(\lambda - \bar{\lambda}) \geq \left(\inf_{\substack{\mu \in \partial B \cap R(C) \\ C^\top\mu \in R(Q^\dagger)}} \mu^\top H \mu \right) \|\lambda - \bar{\lambda}\|^2 = \sigma \|\lambda - \bar{\lambda}\|^2.$$

Now (4.1) with $L = 1/\sigma$ follows by using the Cauchy-Schwarz inequality on the left hand side. □

When C is surjective and $Q \succ 0$, extending this result to the dual function associated with the strictly convex quadratic problem (1.2) is an easy exercise (then H is positive definite and L is the inverse of the smallest eigenvalue of H). On the other hand, when C is not surjective, property (4.1) cannot hold without being lightly weakened, as shown by the following example.

Example 4.3 Consider the special QP with a single inactive constraint ($m = 1$, $C = 0$, and $l < 0 < u$) and a zero optimal value ($q = 0$). Then δ is the function

$$\delta(\lambda) = \begin{cases} -u\lambda & \text{if } \lambda \leq 0 \\ -l\lambda & \text{if } \lambda > 0. \end{cases}$$

Clearly, (4.1) can hold only if λ is not too far from the dual solution set: $|\lambda| \leq L \min(-l, u)$. □

The analysis of the inequality constrained QP is more difficult than the one of problem (4.2), since it has to cover simultaneously two different cases: the quadratic dual function of the equality constrained QP (4.2) and the sharp dual function of the previous example.

In the case of an inequality constrained QP, it will be shown that the Lipschitz constant L may also depend on the largest gap Δ between the $\bar{y} \in \mathcal{S}_P^y$ and the inactive bounds. More precisely, Δ is defined by

$$\Delta := \sup_{\bar{y} \in \mathcal{S}_P^y} \min \left(\min_{i \in I_l \cup J} (u_i - \bar{y}_i), \min_{i \in J \cup I_u} (\bar{y}_i - l_i) \right), \tag{4.5}$$

where the index sets I_l, J, and I_u are introduced in (3.7). If $J = \emptyset$, either I_l or $I_u \neq \emptyset$, and the fact that $l < u$ implies that $\Delta > 0$. If $J \neq \emptyset$, the convexity of $\mathcal{S}_P$ implies that there is a $\bar{y} \in \mathcal{S}_P^y$ such that $l_J < \bar{y}_J < u_J$, in which case also $\Delta > 0$. The dependence of L on Δ is clearly visible in example 4.3: for λ at a unit distance from the solution, we must have $L \geq 1/\min(-l, u) = 1/\Delta$. This lower bound on L goes to infinity when l or u tends to zero, and it goes to zero when $l \to -\infty$ and $u \to +\infty$.

Proposition 4.4 *Consider problem* (1.1) *with* $Q \succcurlyeq 0$ *and suppose that it has a solution. Then, for any bounded set* $\mathcal{B} \subset \mathbb{R}^m$*, there exists a constant* L*, such that*

$$\forall \lambda \in \mathcal{S}_D + \mathcal{B},\ \forall g \in \partial\delta(\lambda) : \quad \operatorname{dist}(\lambda, \mathcal{S}_D) \leq L\|g\|. \tag{4.6}$$

PROOF. *First stage: definition of* L. We know that $\mathcal{S}_D \neq \emptyset$. Let $\mathcal{B}$ be a bounded set in $\mathbb{R}^m$, i.e., $\mathcal{B} \subset \beta B$ for some $\beta > 0$. To make the proof rigorous, we now define $L > 0$, even though the motivation for its definition will not look quite clear at this point.

Let $\mathcal{K}$ be the collection of index sets $K \subset \{1, \ldots, m\}$ such that $I_l \subset K$ and $I_u \subset K^c := \{1, \ldots, m\} \backslash K$ (the index sets I_l and I_u are defined in (3.7)). With any index set $K \subset \{1, \ldots, m\}$, we associate the orthant

$$\mathcal{O}_K := \{\lambda \in \mathbb{R}^m : \lambda_K \geq 0, \lambda_{K^c} \leq 0\}.$$

Define $\mathcal{O}$ and $\mathcal{A}$ by (3.8). For any index set $K \in \mathcal{K}$, $\mathcal{O}_K \cap \mathcal{A}$ is nonempty (since it contains $\mathcal{S}_D = \mathcal{O} \cap \mathcal{A}$, see lemma 3.2). Therefore, with an index set $K \in \mathcal{K}$, corollary 3.6 associates two constants $\tau_K > 0$ and $\gamma_K > 0$ such that for any $\lambda \in \mathbb{R}^m$:

$$\operatorname{dist}(\lambda, \mathcal{O}_K) \leq \tau_K \operatorname{dist}(\lambda, \mathcal{O}_K \cap \mathcal{A}) \quad \Longrightarrow \quad \operatorname{dist}(\lambda, \mathcal{A}) \geq \gamma_K \operatorname{dist}(\lambda, \mathcal{O}_K \cap \mathcal{A}).$$

Since $\mathcal{K}$ is finite, the constants

$$\tau := \min_{K \in \mathcal{K}} \tau_K \quad \text{and} \quad \gamma := \min_{K \in \mathcal{K}} \gamma_K$$

are positive. Therefore, we have found two constants $\tau > 0$ and $\gamma > 0$ such that, for any $K \in \mathcal{K}$, there holds

$$\begin{aligned} \lambda \in \mathcal{O}_K^\tau := \{\lambda' \in \mathbb{R}^m : \operatorname{dist}(\lambda', \mathcal{O}_K) \leq \tau \operatorname{dist}(\lambda', \mathcal{O}_K \cap \mathcal{A})\} \\ \Longrightarrow \quad \operatorname{dist}(\lambda, \mathcal{A}) \geq \gamma \operatorname{dist}(\lambda, \mathcal{O}_K \cap \mathcal{A}). \end{aligned} \tag{4.7}$$

Recall the definitions (4.4) of $\sigma > 0$ and (4.5) of $\Delta > 0$. Then, the constant $L \geq 0$ is defined by

$$L := \max \left(\frac{1}{\sigma\gamma^2}, \frac{\beta}{\tau\Delta} \right). \tag{4.8}$$

In this formula, the constants $\sigma > 0$ and $\Delta > 0$ may take the value $+\infty$. Therefore, L is finite, but can vanish.

Second stage: proof of (4.6). Fix $\lambda \in \mathcal{S}_D + \mathcal{B}$ and $g \in \partial\delta(\lambda)$ (necessarily, $\lambda \in \operatorname{dom}\delta$). Denote the projection of λ onto $\mathcal{S}_D$ by $\bar{\lambda} := P_{\mathcal{S}_D}(\lambda)$. Observe that,

$$\|\lambda - \bar{\lambda}\| \leq \beta. \tag{4.9}$$

Let $\varepsilon \in]0, \Delta[$ and define L_ε by formula (4.8), but with $\Delta - \varepsilon$ in place of Δ. Observe now that showing

$$g^\top(\lambda - \bar{\lambda}) \geq \frac{1}{L_\varepsilon}\|\lambda - \bar{\lambda}\|^2 \tag{4.10}$$

suffices to conclude the proof since then the inequality in (4.6) follows from the Cauchy-Schwarz inequality applied to the left hand side of (4.10) and the fact that ε can be chosen arbitrarily close to zero.

From the form of the subdifferential $\partial\delta(\lambda)$ given by lemma 3.1, we have for some $y_\lambda \in Y_\lambda$, some $y_{\bar{\lambda}} \in Y_{\bar{\lambda}}$, and some u, $\bar{u} \in N(Q)$:

$$g = H\lambda - v - y_\lambda + Cu \qquad \text{and} \qquad 0 = H\bar{\lambda} - v - y_{\bar{\lambda}} + C\bar{u}. \tag{4.11}$$

According to lemma 3.2, $y_{\bar{\lambda}}$ can be chosen arbitrarily in $\mathcal{S}_P^y$ and we take it such that

$$\min\left(\min_{i \in I_l \cup J}(u_i - \bar{y}_i), \min_{i \in J \cup I_u}(\bar{y}_i - l_i)\right) \geq \Delta - \varepsilon. \tag{4.12}$$

As in the proof of proposition 4.2, $(u - \bar{u})^\top C^\top(\lambda - \bar{\lambda}) = 0$, because $C^\top(\lambda - \bar{\lambda}) \in R(Q)$ (both λ and $\bar{\lambda} \in \operatorname{dom}\delta \subset \Lambda$) and $Q(u - \bar{u}) = 0$. Therefore, subtracting the identities in (4.11) and taking the scalar product with $(\lambda - \bar{\lambda})$ yield

$$g^\top(\lambda - \bar{\lambda}) = (\lambda - \bar{\lambda})^\top H(\lambda - \bar{\lambda}) - (y_\lambda - y_{\bar{\lambda}})^\top(\lambda - \bar{\lambda}). \tag{4.13}$$

We will get (4.10) by finding a lower bound of the right hand side of (4.13). Note that the two terms are nonnegative (this is clear for the first one, since H is positive semi-definite; for the second one, use the monotonicity property (3.4)).

Since $\bar{\lambda} = P_{\mathcal{A}\cap\mathcal{O}}(\lambda)$, by lemma 3.7, one can find an index set $K \subset \mathcal{K}$ such that $\bar{\lambda} = P_{\mathcal{A}\cap\mathcal{O}_K}(\lambda)$. We analyze successively two complementary cases, using the set $\mathcal{O}_K^\tau$ defined in (4.7) and $\lambda^t := (1-t)\bar{\lambda} + t\lambda$ for $t \in \mathbb{R}$.

Case A: there exists a $t \in]0, 1]$ *such that* $\lambda^t \in \mathcal{O}_K^\tau$. In this case, we work on the first term in the right hand side of (4.13), discarding the second one. Because $P_{\mathcal{A}\cap\mathcal{O}_K}(\lambda^t) = \bar{\lambda}$, (4.7) gives

$$\gamma\|\lambda^t - \bar{\lambda}\| \leq \|\lambda^t - P_{\mathcal{A}}(\lambda^t)\|.$$

Decompose $\lambda^t - \bar{\lambda} = \mu_0 + \mu_1$, where $\mu_0 \in N(C^\top)$ and $\mu_1 \in R(C)$, and observe that $C^\top\mu_1 = C^\top(\lambda^t - \bar{\lambda}) \in R(Q) = R(Q^\dagger)$ (since both λ^t and $\bar{\lambda} \in \operatorname{dom}\delta \subset \Lambda$). Then, using the definition (4.4) of σ, one finds

$$(\lambda^t - \bar{\lambda})^\top H(\lambda^t - \bar{\lambda}) = \mu_1^\top H\mu_1 \geq \sigma\|\mu_1\|^2.$$

From the fact that $\mu_1 \in R(C)$ and that $C^\top(\lambda^t - \mu_1) = C^\top(\bar\lambda + \mu_0) = C^\top\bar\lambda = Q\bar x + q$, we deduce that $\mu_1 = \lambda^t - P_{\mathcal{A}}(\lambda^t)$. Therefore

$$(\lambda^t - \bar\lambda)^\top H(\lambda^t - \bar\lambda) \geq \sigma\gamma^2\|\lambda^t - \bar\lambda\|^2.$$

Since $\lambda - \bar\lambda = (\lambda^t - \bar\lambda)/t$, we also have

$$(\lambda - \bar\lambda)^\top H(\lambda - \bar\lambda) \geq \sigma\gamma^2\|\lambda - \bar\lambda\|^2.$$

Discarding the second term in the right hand side of (4.13) (it is nonnegative) and using the definition of L in (4.8), it follows that

$$g^\top(\lambda - \bar\lambda) \geq (\lambda - \bar\lambda)^\top H(\lambda - \bar\lambda) \geq \sigma\gamma^2\|\lambda - \bar\lambda\|^2 \geq \frac{1}{L}\|\lambda - \bar\lambda\|^2 \geq \frac{1}{L_\varepsilon}\|\lambda - \bar\lambda\|^2,$$

which is the expected inequality (4.10).

Case B: for any $t \in]0,1]$, $\lambda^t \notin \mathcal{O}_K^\tau$. In this case, we work on the second term in the right hand side of (4.13), discarding the first one. Let us start by choosing $t \in]0,1]$ sufficiently small such that $\lambda_i^t\bar\lambda_i > 0$ when $\bar\lambda_i \neq 0$; by assumption, this $\lambda^t \notin \mathcal{O}_K^\tau$. Let $g^t \in \partial\delta(\lambda^t)$, so that $g^t = H\lambda^t - v - y_{\lambda^t} + Cu^t$ for some $y_{\lambda^t} \in Y_{\lambda^t}$ and some $u^t \in N(Q)$ (compare with (4.11)). Proceeding as before, we get an identity like (4.13):

$$(g^t)^\top(\lambda^t - \bar\lambda) = (\lambda^t - \bar\lambda)^\top H(\lambda^t - \bar\lambda) - (y_{\lambda^t} - y_{\bar\lambda})^\top(\lambda^t - \bar\lambda). \tag{4.14}$$

Denote by $\lambda_K^t := P_{\mathcal{O}_K}(\lambda^t)$ the projection of λ^t onto $\mathcal{O}_K$ and decompose

$$-(y_{\lambda^t} - y_{\bar\lambda})^\top(\lambda^t - \bar\lambda) = -(y_{\lambda^t} - y_{\bar\lambda})^\top(\lambda^t - \lambda_K^t) - (y_{\lambda^t} - y_{\bar\lambda})^\top(\lambda_K^t - \bar\lambda).$$

The last term in the right hand side is nonnegative. Indeed, by the choice of t, if $\bar\lambda_i \neq 0$, one has $\lambda_i^t\bar\lambda_i > 0$ and therefore $(y_{\lambda^t} - y_{\bar\lambda})_i = 0$ (see the definition (3.3) of Y_λ). The only nonzero terms of the last scalar product are therefore of the form $(y_{\bar\lambda} - y_{\lambda^t})_i(\lambda_K^t)_i$. If $(\lambda_K^t)_i > 0$, one has $\lambda_i^t > 0$ (since λ_K^t is the projection of λ^t onto $\mathcal{O}_K$) and therefore $(y_{\lambda^t})_i = l_i$, so that the term can be written $((y_{\bar\lambda})_i - l_i)(\lambda_K^t)_i \geq 0$ (since $l_i \leq (y_{\bar\lambda})_i \leq u_i$). Similarly, the term is nonnegative when $(\lambda_K^t)_i < 0$. Therefore

$$-(y_{\lambda^t} - y_{\bar\lambda})^\top(\lambda^t - \bar\lambda) \geq -(y_{\lambda^t} - y_{\bar\lambda})^\top(\lambda^t - \lambda_K^t) = \sum_{i \in I_{K,\lambda^t}} (y_{\bar\lambda} - y_{\lambda^t})_i(\lambda^t - \lambda_K^t)_i,$$

where we have introduced the index set

$$I_{K,\lambda^t} := \{i : \lambda_i^t \neq (\lambda_K^t)_i\}.$$

Let us show that all the terms of the sum on the indices $i \in I_{K,\lambda^t}$ above are positive. Observe first that $(\lambda_K^t)_i = 0$ (since $\lambda_i^t \neq (\lambda_K^t)_i$ and λ_K^t is the projection of λ^t onto the orthant $\mathcal{O}_K$). On the other hand, if $\lambda_i^t > 0$, then $(y_{\lambda^t})_i = l_i$ and $l_i < (y_{\bar\lambda})_i \leq u_i$ (this is because $i \in K^c$ when $\lambda_i^t > 0$ and $(\lambda_K^t)_i = 0$, and because $I_l \subset K$); therefore $(y_{\bar\lambda} - y_{\lambda^t})_i = (y_{\bar\lambda})_i - l_i \geq \Delta - \varepsilon > 0$ (see (4.12)). Similarly, if $\lambda_i^t < 0$, then $(y_{\lambda^t})_i = u_i$,

$i \in K$, and $(y_{\bar\lambda} - y_{\lambda^t})_i = (y_{\bar\lambda})_i - u_i \leq -(\Delta - \varepsilon) < 0$. In particular, all the terms of the sum are positive. Therefore, we get ($\|\cdot\|_1$ denotes the ℓ_1-norm)

$$-(y_{\lambda^t} - y_{\bar\lambda})^\top (\lambda^t - \bar\lambda) \geq (\Delta - \varepsilon)\|\lambda^t - \lambda^t_K\|_1 \geq (\Delta - \varepsilon)\|\lambda^t - \lambda^t_K\| \geq \tau(\Delta - \varepsilon)\|\lambda^t - \bar\lambda\|, \quad (4.15)$$

where the last inequality comes from the fact that $\lambda^t \notin \mathcal{O}^\tau_K$ and that $P_{\mathcal{A}\cap\mathcal{O}_K}(\lambda^t) = \bar\lambda$. We can now conclude:

$$\begin{aligned}
g^\top(\lambda - \bar\lambda) &\geq (g^t)^\top(\lambda - \bar\lambda) && \text{[monotonicity of the subdifferential]} \\
&= \frac{1}{t}(g^t)^\top(\lambda^t - \bar\lambda) && \text{[definition of } \lambda^t\text{]} \\
&\geq -\frac{1}{t}(y_{\lambda^t} - y_{\bar\lambda})^\top(\lambda^t - \bar\lambda) && \text{[(4.14)]} \\
&\geq \frac{\tau(\Delta - \varepsilon)}{t}\|\lambda^t - \bar\lambda\| && \text{[(4.15)]} \\
&\geq \tau(\Delta - \varepsilon)\|\lambda - \bar\lambda\| && \text{[definition of } \lambda^t\text{]} \\
&\geq \frac{\tau(\Delta - \varepsilon)}{\beta}\|\lambda - \bar\lambda\|^2 && \text{[(4.9)]} \\
&\geq \frac{1}{L_\varepsilon}\|\lambda - \bar\lambda\|^2 && \text{[definition of } L_\varepsilon\text{]}.
\end{aligned}$$

This is the expected inequality (4.10). □

4.2 Global linear convergence

We can now state the global linear convergence of the constraint norm towards zero in the AL algorithm of section 2. Note that the rate of convergence $\min(L/r_k, 1)$ may depend through L on the distance from the initial iterate λ_0 to the dual solution set $\mathcal{S}_D$.

Theorem 4.5 *Suppose that problem* (1.1) *with* $Q \succcurlyeq 0$ *has a solution. Consider the AL algorithm of section* 2. *For any* $\beta > 0$, *there exists an* $L > 0$, *such that* $\operatorname{dist}(\lambda_0, \mathcal{S}_D) \leq \beta$ *implies that*

$$\|y_{k+1} - Cx_{k+1}\| \leq \min\left(\frac{L}{r_k}, 1\right)\|y_k - Cx_k\|, \qquad \text{for all } k \geq 1. \quad (4.16)$$

In particular, if $r_k \geq \bar r$ *for all* $k \geq 1$ *and some* $\bar r > L$, *the constraint norm tends to zero globally linearly.*

PROOF. The proof gathers known techniques (see for example [27, 28]) with the result of proposition 4.4. We give the details for completeness.

Let us note $g_{k+1} := Cx_{k+1} - y_{k+1}$. Recall from (3.10) that $g_{k+1} \in \partial\delta(\lambda_{k+1})$. Subtracting two consecutive iteration identities (2.4) provides

$$\frac{1}{r_{k+1}}(\lambda_{k+2} - \lambda_{k+1}) + (g_{k+2} - g_{k+1}) = \frac{1}{r_k}(\lambda_{k+1} - \lambda_k).$$

Taking norms, using the monotonicity of the subdifferential (which implies that $(g_{k+2} - g_{k+1})^\top(\lambda_{k+2} - \lambda_{k+1}) \geq 0$), and discarding $\|g_{k+2} - g_{k+1}\|^2 \geq 0$, we get $\|\lambda_{k+2} - \lambda_{k+1}\|^2 / r_{k+1}^2 \leq \|\lambda_{k+1} - \lambda_k\|^2 / r_k^2$ or $\|g_{k+2}\| \leq \|g_{k+1}\|$. This yields the second part of the min in (4.16).

Subtracting an arbitrary dual solution $\bar{\lambda} \in \mathcal{S}_D$ from both sides of the iteration identity (2.4) gives

$$\lambda_{k+1} - \bar{\lambda} + r_k g_{k+1} = \lambda_k - \bar{\lambda}, \quad \text{for } k \geq 0.$$

Taking norms and using the monotonicity of the subdifferential lead to

$$\|\lambda_{k+1} - \bar{\lambda}\|^2 + r_k^2 \|g_{k+1}\|^2 \leq \|\lambda_k - \bar{\lambda}\|^2, \quad \text{for } k \geq 0. \tag{4.17}$$

This shows in particular that the sequence $\{\|\lambda_k - \bar{\lambda}\|\}_{k \geq 0}$ is nonincreasing. Since $\bar{\lambda}$ is arbitrary in $\mathcal{S}_D$, there holds

$$\operatorname{dist}(\lambda_{k+1}, \mathcal{S}_D) \leq \|\lambda_{k+1} - P_{\mathcal{S}_D}(\lambda_k)\| \leq \|\lambda_k - P_{\mathcal{S}_D}(\lambda_k)\| = \operatorname{dist}(\lambda_k, \mathcal{S}_D).$$

Therefore, $\{\operatorname{dist}(\lambda_k, \mathcal{S}_D)\}_{k \geq 0}$ is also nonincreasing, so that $\lambda_k \in \mathcal{S}_D + \beta B$ for all $k \geq 0$. Now, let $L > 0$ be the constant that proposition 4.4 associates with $\mathcal{B} := \beta B$. By this proposition, $\|\lambda_k - P_{\mathcal{S}_D}(\lambda_k)\| \leq L\|g_k\|$. Discarding the first term in the left hand side of (4.17) and using $P_{\mathcal{S}_D}(\lambda_k)$ for $\bar{\lambda}$, we get $\|g_{k+1}\| \leq (L/r_k)\|g_k\|$. This yields the first part of the min in (4.16). □

5 Numerical experiments and discussion

The aim of this section is to illustrate by numerical experiments the global linear convergence property of the AL algorithm studied in this paper and to assess the quality of the bound given by theorem 4.5. The numerical experiments are taken from seismic reflection tomography applications. We conclude with a discussion on algorithmic implications.

5.1 A seismic reflection tomography problem

Seismic reflection tomography is a technique used to recover the geological structure of the subsoil from the measurements of the travel-times of seismic waves (see [10] for a description of the approach). From an optimization viewpoint, the problem consists in minimizing a nonlinear least-squares function subject to nonlinear constraints. In [5], a Gauss-Newton SQP method globalized by line-search is proposed and analyzed. At each iteration, a solution to a strictly convex quadratic model of the objective function subject to linear constraints is computed using our code QPAL [4, 6].

We have chosen here to present the results obtained with the problem KARINE, which is representative of those observed with our collection of 2D and 3D seismic reflection problems. These have a number of variables up to 15 10^3 and a number of constraints up to 10^4. The features of the selected problem are summarized in table 1.

n	m	m^*_{act}	κ_2
442	320	108	$8.4\ 10^5$

Table 1: Description of the tomography problem KARINE

It is a 2D model depending on $n = 442$ parameters and having $m = 320$ linear inequality constraints. The matrix Q of the selected quadratic subproblem (1.1) is positive definite and has its ℓ_2 condition number equal to $\kappa_2 = 8.4\ 10^5$. Its constraint matrix C has been balanced (the Euclidean norm of each of its rows is equal to 1). The number of active constraints at the solution is $m^*_{act} = 108$, which represents 34 % of the number of constraints.

The results presented in section 5.2 have been obtained using the AL algorithm described in section 2, with a fixed augmentation parameter r. In order to study the dependence of the results on r, we have run the QP solver for 21 different values of r, ranging from 1 to 10^5. In each case, the AL algorithm is initialized with a null Lagrangian multiplier ($\lambda_0 = 0$) and is stopped when the constraint norm is sufficiently small ($\|y_k - Cx_k\| \leq 10^{-10}$).

5.2 Assessing the global linear convergence result

In this section, we illustrate the global linear convergence property of the AL algorithm established in theorem 4.5. The actual global rate of linear convergence is given by $\rho := \sup\{\|y_{k+1} - Cx_{k+1}\| / \|y_k - Cx_k\| : k \geq 1\}$ and can be estimated during a particular run by

$$\rho_{\text{est}} := \max_{1 \leq k \leq n_{AL}} \frac{\|y_{k+1} - Cx_{k+1}\|}{\|y_k - Cx_k\|} \leq \rho, \tag{5.1}$$

where n_{AL} is the number of AL iterations actually performed to reach the required accuracy of 10^{-10} on the constraint norm.

Theorem 4.5 has shown that ρ is bounded above by a function of r:

$$\rho \leq \min\left(\frac{L}{r}, 1\right) \qquad \text{or} \qquad \log \rho \leq \min(\log L - \log r, 0). \tag{5.2}$$

A natural question is to know whether this bound is tight in practice. This is difficult to say, since the value of L is generally unknown, but the appearance of ρ_{est} as a function of r in the considered problem may give a clue on this question.

The plain curve in figure 1 gives $\log \rho_{\text{est}}$ as a function of $\log r$ (double logarithmic scale). As predicted by the theory, we see that $\rho_{\text{est}} \leq 1$ for all positive r. Furthermore, the larger is the augmentation parameter r, the faster is the convergence: $\rho \simeq 1$ for $r \leq 10$ (convergence is hardly detectable) and $\rho \simeq 3.10^{-3}$ for $r = 10^5$ (convergence is obtained in very few AL iterations). We have represented by a dotted line the tangent to the ρ_{est} curve with a slope -1. This line crosses the top horizontal line of the graph at the horizontal coordinate $L_{\text{inf}} \simeq 304$. According to (5.2), L_{inf} provides a lower estimate of the value of L. Since both curves (the plain and dotted ones) are quite

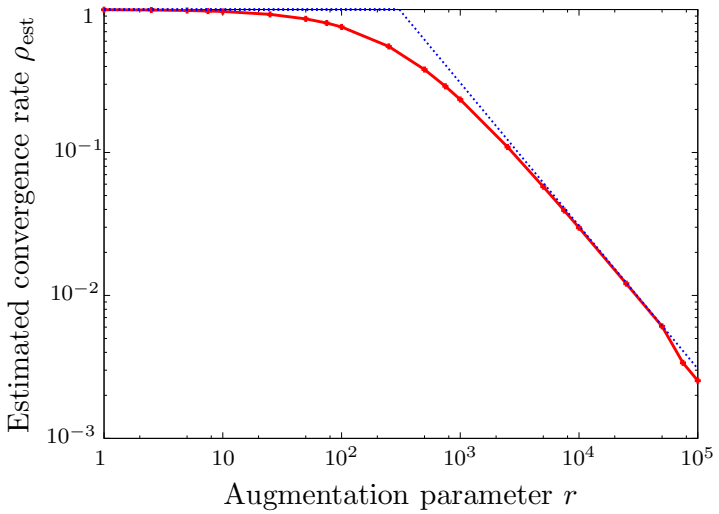

Figure 1: Global linear convergence rate of the constraint norm as a function of r

close, it is likely that the dotted curve is close to the upper bound on ρ given by (5.2). As a result, it is likely that the upper bound given by theorem 4.5 is tight. Note that the small discrepancy between both curves for large values of r ($r \geq 7.5\ 10^4$) is due here to the fact that the AL algorithm reaches the required constraint norm accuracy in very few AL iterations ($n_{AL} \leq 3$), so that the maximum in (5.1) is taken on that few number. In other cases, such a discrepancy can come from an inexact solve of the bound constraint problem (2.3), due to a large value of r.

5.3 Discussion

As shown in this paper, the global linear rate of convergence of the AL algorithm depends on the Lipschitz constant L given by (4.8) and on the value of $\bar{r} := \inf r_k$, where r_k is the value of the augmentation parameter at iteration k. More precisely, the decrease of the constraint norm at iteration k is bounded above by L/r_k. It is usually impossible to compute L in practice, since it depends on the constants γ, σ, and Δ (see lemma 3.4, (4.4), (4.5), and finally (4.8)), which are either unknown or too expensive to compute. As a Lipschitz constant, however, L has easily computable *lower* estimates.

The estimate L_{inf} of L given in section 5.2 is not available at run time, since it requires to run the AL algorithm on a particular problem for various values of r. Nevertheless, the quantities

$$L_{\text{inf},k} := \max_{1 \leq i \leq k} \left(r_i \frac{\|y_{i+1} - Cx_{i+1}\|}{\|y_i - Cx_i\|} \right)$$

satisfy $L_{\text{inf},k} \leq L$ and can therefore be used as a lower estimate of L, after iteration k is completed. A given desired rate of convergence $\rho_{\text{des}} \in]0, 1[$ is then likely to be obtained at iteration $k+1$ by taking

$$r_{k+1} \geq \frac{L_{\text{inf},k}}{\rho_{\text{des}}}. \tag{5.3}$$

It is the fact that the estimate (4.16) has a *global* validity that gives sense to an update of the value of r_k in this way at *each* iteration. It should be clear at this point that the AL algorithm gains in efficiency by taking r_k as large as possible, the only limitation being that problem (2.3) needs to be numerically solvable. Since it is sometimes difficult to tell what is a large value for a particular problem, the lower bound on r_{k+1} in (5.3) may also be useful as a reference.

We conclude with a result providing an estimate of the number of iterations needed to reach a given tolerance on the constraint norm. Assume that a number $\rho_{\text{des}} \in]0,1[$ is given as a desired rate of convergence. Of course, since the Lipschitz constant L is unknown, this rate of convergence cannot be ensured, but the algorithm can try to approach it by updating r_k when it feels it is necessary. The next result gives an estimate of the iterative complexity of the AL algorithm with an update rule based on (5.3). More precisely, defining

$$\rho_k := \frac{\|y_{k+1} - Cx_{k+1}\|}{\|y_k - Cx_k\|},$$

the AL algorithm is supposed to update the value of r_k, for $k \geq 1$, according to:

$$\text{if } \rho_k \leq \rho_{\text{des}}, \quad \text{then } r_{k+1} = r_k, \quad \text{else } r_{k+1} = \tfrac{\rho_k}{\rho_{\text{des}}}\, r_k. \tag{5.4}$$

There is nothing magic in the update rule of r_k above. It could equally use $r_{k+1} = 10\,\rho_k r_k/\rho_{\text{des}}$ or simply $r_{k+1} = 10\,r_k$ when r_k needs to be increased.

Proposition 5.1 *Suppose that the AL algorithm of section 2 uses the rule (5.4) to update the augmentation parameter r_k, for $k \geq 1$. Let $\varepsilon \in]0,1]$ and let L be the positive constant given by theorem 4.5. Fix any $t \in]\rho_{\text{des}}, 1[$. Then*

$$\|y_{k+1} - Cx_{k+1}\| \leq \varepsilon \|y_1 - Cx_1\|, \tag{5.5}$$

as soon as

$$k \geq \frac{\log \varepsilon}{\log t} + \max\left(1 + \frac{\log(L/(tr_1))}{\log(t/\rho_{\text{des}})}, 0\right). \tag{5.6}$$

PROOF. Let $t \in]\rho_{\text{des}}, 1[$. Clearly, since $\rho_i \leq 1$,

$$\frac{\|y_{k+1} - Cx_{k+1}\|}{\|y_1 - Cx_1\|} = \prod_{1 \leq i \leq k} \rho_i \leq \prod_{\substack{1 \leq i \leq k \\ \rho_i \leq t}} \rho_i \leq t^{k_t},$$

where $k_t := |K_t|$ is the number of elements in $K_t := \{i \in \mathbb{N} : 1 \leq i \leq k,\ \rho_i \leq t\}$. Taking logarithms, we see that (5.5) holds as soon as $k_t \geq (\log \varepsilon)/(\log t)$.

If $K_t^c := \{1, \ldots, k\} \backslash K_t$ is empty, then $k = k_t$ and the result is proven.

Suppose now that $K_t^c \neq \emptyset$. Since $\rho_i \leq L/r_i$ (by theorem 4.5), $i \in K_t$ as soon as $r_i \geq L/t$. Let j be the last index in K_t^c, namely the $(k-k_t)$th one, if any. Then r_j is the result of $k - k_t - 1$ updates from r_1, using factors ρ_i/ρ_{des} that are $\geq t/\rho_{\text{des}}$ (see the

update rule (5.4)). Hence we must have $(t/\rho_{\text{des}})^{k-k_t-1} r_1 \leq r_j \leq L/t$. This gives an upper bound on the number of elements of K_t^c, namely

$$k - k_t \leq 1 + \frac{\log(L/(tr_1))}{\log(t/\rho_{\text{des}})}.$$

The total number of iterations to satisfy (5.5) is therefore at most this upper bound on the number of elements in K_t^c, plus the lower bound on k_t obtained above. □

Roughly expressed, the number of iterations needed to reach precision $\varepsilon > 0$ on the relative constraint norm is of order $O(\log \varepsilon) + O(\log L)$. As shown in the proof of proposition 5.1, the first term of order $O(\log \varepsilon)$ is due to the linear convergence of the constraint norm towards zero, which is triggered when the augmentation parameter is large enough (a consequence of theorem 4.5). The second term of order $O(\log L)$, which is the only place where the dimension of the problem can intervene, is due to a possible too small value of r_1 and to the number of iterations that the rule (5.4) needs to make r_k large enough. This term can be made as small as desired by choosing a large value for r_1 or by adopting an update rule of r_k that increases these values more rapidly than in (5.4). As a result, the *computational* complexity of the AL algorithm of section 2 essentially rests on the one of the AL subproblems (2.3). When strict complementarity holds, the finite identification of the active constraints in (2.3) occurs and the computational complexity is then basically induced by the very first AL subproblems. Our experience with the AL algorithm, limited to the seismic reflection tomography problems described in section 5.1, supports that conclusion.

References

[1] K.J. Arrow, R.M. Solow (1958). Gradient methods for constrained maxima with weakened assumptions. In K.J. Arrow, L. Hurwicz, H. Uzawa (editors), *Studies in Linear and Nonlinear Programming*, pages 166–176. Stanford University Press, Standford, Calif.

[2] J.F. Bonnans, J.Ch. Gilbert, C. Lemaréchal, C. Sagastizábal (2003). *Numerical Optimization – Theoretical and Practical Aspects*. Springer Verlag, Berlin

[3] J.V. Burke, M.C. Ferris (1991). Characterization of solution sets of convex programs. *Operations Research Letters*, 10, 57–60

[4] F. Delbos (2004). *Problèmes d'Optimisation de Grande Taille avec Contraintes en Tomographie de Réflexion 3D*. PhD Thesis, University Pierre et Marie Curie (Paris VI), Paris. (to appear)

[5] F. Delbos, J.Ch. Gilbert, R. Glowinski, D. Sinoquet (2003). Constrained optimization in reflection tomography: an augmented Lagrangian SQP approach. Technical report, Institut Français du Pétrole. (to appear)

[6] F. Delbos, J.Ch. Gilbert, D. Sinoquet (2003). QPAL: a solver of large-scale convex quadratic optimization problems using an augmented Lagrangian approach. Technical report, INRIA, BP 105, 78153 Le Chesnay, France. (to appear)

[7] D. den Hertog (1992). *Interior Point Approach to Linear, Quadratic and Convex Programming*. Mathematics and its Applications 277. Kluwer Academic Publishers, Dordrecht

[8] M. Fortin, R. Glowinski (1982). *Méthodes de Lagrangien Augmenté – Applications à la Résolution Numérique de Problèmes aux Limites.* Méthodes Mathématiques de l'Informatique 9. Dunod, Paris

[9] A. Friedlander, J.M. Martínez (1994). On the maximization of a concave quadratic function with box constraints. *SIAM Journal on Optimization*, 4, 177–192

[10] R. Glowinski, Q.-H. Tran (1993). Constrained optimization in reflexion tomography: the augmented Lagrangian method. *East-West J. Numer. Math.*, 1(3), 213–234

[11] M.R. Hestenes (1969). Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4, 303–320

[12] J.-B. Hiriart-Urruty, C. Lemaréchal (1993). *Convex Analysis and Minimization Algorithms.* Grundlehren der mathematischen Wissenschaften 305-306. Springer-Verlag

[13] A.F. Izmailov, M.V. Solodov (2001). Error bounds for 2-regular mappings with Lipschitzian derivatives and their applications. *Mathematical Programming*, 89, 413–435

[14] B. Jansen (1997). *Interior Point Techniques in Optimization – Complementarity, Sensitivity and Algorithms.* Applied Optimization 6. Kluwer Academic Publishers, Dordrecht

[15] M. Kojima, N. Megiddo, T. Noma, A. Yoshise (1991). *A Unified Approach to Interior Point Algorithms for Linear Complementarity Problems.* Lecture Notes in Computer Science 538. Springer-Verlag, Berlin

[16] O.L. Mangasarian (1988). A simple characterization of solution sets of convex programs. *Operations Research Letters*, 7, 21–26

[17] J.J. Moré, G. Toraldo (1991). On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1, 93–113

[18] Y.E. Nesterov, A.S. Nemirovskii (1994). *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM Studies in Applied Mathematics 13. SIAM, Philadelphia

[19] J. Nocedal, S.J. Wright (1999). *Numerical Optimization.* Springer Series in Operations Research. Springer, New York

[20] J.S. Pang (1983). Methods for quadratic programming: a survey. *Computers and Chemical Engineering*, 7, 583–594

[21] J.S. Pang (1997). Error bounds in mathematical programming. *Mathematical Programming*, 79, 299–332

[22] M.J.D. Powell (1969). A method for nonlinear constraints in minimization problems. In R. Fletcher (editor), *Optimization*, pages 283–298. Academic Press, London

[23] R.T. Rockafellar (1970). *Convex Analysis.* Princeton Mathematics Ser. 28. Princeton University Press, Princeton, New Jersey

[24] R.T. Rockafellar (1971). New applications of duality in convex programming. In *Proceedings of the 4th Conference of Probability, Brasov, Romania*, pages 73–81

[25] R.T. Rockafellar (1973). A dual approach to solving nonlinear programming problems by unconstrained optimization. *Mathematical Programming*, 5, 354–373

[26] R.T. Rockafellar (1974). Augmented Lagrange multiplier functions and duality in nonconvex programming. *SIAM Journal on Control*, 12, 268–285

[27] R.T. Rockafellar (1976). Monotone operators and the proximal point algorithm. *SIAM Journal on Control and Optimization*, 14, 877–898

[28] R.T. Rockafellar (1976). Augmented Lagrangians and applications of the proximal point algorithm in convex programming. *Mathematics of Operations Research*, 1, 97–116

[29] C. Roos, T. Terlaky, J.-Ph. Vial (1997). *Theory and Algorithms for Linear Optimization – An Interior Point Approach.* John Wiley & Sons, Chichester

[30] T. Terlaky (editor) (1996). *Interior Point Methods of Mathematical Programming.* Kluwer Academic Press, Dordrecht

[31] S.J. Wright (1997). *Primal-Dual Interior-Point Methods.* SIAM Publication, Philadelphia

Annexe D

Description de notre bibliothèque de modèles de tomographie de réflexion

Notre bibliothèque de modèle comprend quatre modèles de tomographie différents :

1. le modèle *FRANCOIS* est un modèle synthétique très simple comprenant une interface et une vitesse 2D. Il est discrétisé par 325 paramètres de vitesse et d'interface. Le nombre de données observées de ce modèle est de 6007.
2. le modèle *KARINE* est composé de 2 interfaces et d'une vitesse 2D. Ce modèle est un peu plus compliqué que le modèle précedent car les données sont réelles. Par contre, le nombre de paramètre B-spline le discrétisant reste faible. Ce modèle possède 23277 données observées
3. le modèle *DELPHINE* est identique au modèle *KARINE* sauf que le modèle initial est pris plus proche de la solution avec un poids de pénalisation différent.
4. le modèle *M8CAROLE* est composé de 6 interfaces et d'une vitesse 3D. Il est discrétisé par 4902 paramètres. C'est le modèle le plus compliqué de notre bibliothèque de cas tests. Il dispose de 23375 données observées.

Le tableau D.1 résume les différentes caractéristiques des quatres modèles différents de notre bibliothèque.

Sur chacun des modèles de tomographie décrit ci-dessus on peut appliquer des contraintes d'égalité où d'inégalité différentes. Le tableau D.2 résume les caractéristique de tous les exemples de contraintes de notre bibliothèque. Notons que, au total, nous avons à notre disposition 21 exemples de cas tests différents. Dans ce tableau, les colonnes E_v, E_Z, C_V, et C_Z indiquent respectivement pour un exemple donné le nombre de contraintes d'égalité sur les paramètres de vitesse, le nombre de contraintes d'égalité sur les paramètres d'interface, le nombre de contrainte d'inégalité sur les paramètres de vitesse et le nombre de contraintes d'inégalité sur les paramètres de vitesse. La dernière colonne de ce tableau donne le nombre total de contraintes pour chaque exemple de notre bibliothèque. Nous décrivons ci-dessous les différents exemples de contraintes de notre bibliothèque.

Bibliotheque de modele en tomographie de reflexion					
Nom du modele	Interfaces		Vitesses		Nombre de parametres
FRANCOIS	1	x=10 y=10	1	x=15 y=15 (2D)	325
KARINE	2	x=4 y=37 x=4 y=71	1	x=10 y=4 (2D)	472
DELPHINE	2	x=4 y=37 x=4 y=71	1	x=10 y=4 (2D)	472
M8CAROLE	6	6*(x=27 y=21)	1	x=15 y=10 z=10	4902

Tableau D.1 Description des modèles de tomographie de réflexion de notre bibliothèque de cas tests.

1. Le modèle *FRANCOIS* peut être inversé en utilisant 8 exemples différents de contraintes :

   F1• contraintes d'inégalité sur la pente de l'interface Z_1 : "$\nabla_y Z_1 \leq -0,05$"

   F2• contraintes d'inégalité sur la pente de l'interface Z_1 : "$\nabla_y Z_1 \geq -0,05$"

   F3• contraintes d'inégalité sur la vitesse V_1 : "$V_1 \geq 2$"

   F4• contraintes de l'exemple F2 plus celles de l'exemple F3

   F5• contraintes d'égalité sur la pente de l'interface Z_1 : "$\nabla_y Z_1 = -0,05$"

   F6• contraintes de l'exemple F3 plus celles de l'exemple F5

   F7• contraintes d'égalité sur la vitesse V_1 : "$V_1 = 2$"

   F8• contraintes de l'exemple F1 plus celles de l'exemple F7

2. Le modèle *KARINE* peut être inversé en utilisant 3 exemples différents de contraintes :

   K1• contraintes d'égalité sur la pente de l'interface Z_1 (ces contraintes sont localisée sur le bord du modèle) : "$\nabla_x Z_1 = 0$"

   K2• contraintes d'inégalité couplant les interfaces Z_1 et Z_2 :
   "$0,5 \leq Z_2 - Z_1 \leq 0,6$"

   K3• contraintes d'égalité couplant les interfaces Z_1 et Z_2 : "$Z_2 - Z_1 = 0,2$"

3. Le modèle *DELPHINE* peut être inversé en utilisant 3 exemples différent de contraintes :

   D1• contraintes d'inégalité sur la pente des interfaces Z_1 et Z_2 (ces contraintes sont localisée sur le bord du modèle) :
   "$-0,1 \leq \nabla_x Z_1 \leq 0,1$" et "$-0,1 \leq \nabla_x Z_2 \leq 0,1$"

   D2• contraintes d'inégalité sur la pente de l'interface Z_2 : "$-0,1 \leq \nabla_x Z_2 \leq 0,1$"

   D3• contraintes d'inégalité sur la pente de l'interfaces Z_2 (ces contraintes sont localisée sur le bord droit du modèle) : "$-0,1 \leq \nabla_x Z_2 \leq 0,1$"

3. Le modèle *M8CAROLE* peut être inversé en utilisant 7 exemples différent de contraintes :

   M1• contraintes d'inégalité sur l'interface Z_{lias} (l'interface Z_{craie} étant fixée) : "$Z_{lias} - Z_{craie} \geq 0,001$"

   M2• contraintes d'inégalité couplant les interfaces Z_{ouest1} et Z_{ouest2} : "$Z_{ouest1} - Z_{ouest2} \geq 0,001$"

   M3• contraintes d'inégalité couplant les interfaces $Z_{ouest1b}$ et Z_{ouest2} puis les interfaces Z_{ouest1} et $Z_{ouest1b}$:
   "$Z_{ouest2} - Z_{ouest1b} \geq 0,001$" et "$Z_{ouest1b} - Z_{ouest1} \geq 0,001$"

   M4• contraintes d'inégalité sur la vitesse $V_{underbc}$: "$V_{underbc} \geq 5$"

   M5• contraintes d'inégalité sur la dérivée seconde de l'interface Z_{lias} :
   "$-0,001 \leq \nabla^2_{xx} Z_{lias} \leq 0,001$"

   M6• contraintes d'égalité sur la dérivée seconde de l'interface Z_{lias} :
   "$\nabla^2_{xx} Z_{lias} = 0,001$"

   M7• contraintes d'inégalité couplant les interfaces $Z_{ouest1b}$ et Z_{ouest2} puis les interfaces Z_{ouest1} et $Z_{ouest1b}$:
   "$1 \leq Z_{ouest2} - Z_{ouest1b} \leq 1,2$" et "$1 \leq Z_{ouest1b} - Z_{ouest1} \leq 1,2$"

contraintes	E_V	E_Z	C_V	C_Z	total
FRANCOIS					
exemple 1 (**F1**)	0	0	0	100	100
exemple 2 (**F2**)	0	0	0	100	100
exemple 3 (**F3**)	0	0	225	0	225
exemple 4 (**F4**)	0	0	225	100	325
exemple 5 (**F5**)	0	100	0	0	100
exemple 6 (**F6**)	0	100	255		325
exemple 7 (**F7**)	225	0	0	0	225
exemple 8 (**F8**)	225	0	0	100	325
KARINE					
exemple 1 (**K1**)	0	80	0	0	80
exemple 2 (**K2**)	0	0	0	320	320
exemple 3 (**K3**)	0	320	0	0	320
DELPHINE					
exemple 1 (**D1**)	0	0	0	320	320
exemple 2 (**D2**)	0	0	0	80	80
exemple 3 (**D3**)	0	320	0	0	320
M8					
exemple 1 (**M1**)	0	0	0	400	400
exemple 2 (**M2**)	0	0	0	100	100
exemple 3 (**M3**)	0	0	0	200	200
exemple 4 (**M4**)	0	0	1000	0	1000
exemple 5 (**M5**)	0	0	0	100	100
exemple 6 (**M6**)	0	0	0	100	100
exemple 7 (**M7**)	0	0	0	200	200

Tableau D.2 Description des différents exemples de contraintes pour chaque modèle de notre bibliothèque

Annexe E

Description et analyse d'un profil de performances

Dans cette annexe nous décrivons comment construire un profil de performances et comment l'analyser. Pour plus d'informations sur ce sujet, nous renvoyons le lecteur à [50].

Afin de comparer les performances de plusieurs codes d'optimisation (ou solveur) il faut les tester sur un banc de problèmes-tests. Pour chaque problème-test traité par un solveur, diverses sortes de critères de performances peuvent être mesurés dont par exemple :

- le temps CPU,
- le nombre d'évaluations de la fonction coût,
- le nombre d'itérations d'un algorithme (dans le cas où chaque itération représente la même quantité de travail pour chaque code).

On obtient alors des tableaux de résultats dans lequels on peut comparer pour un problème-test donné le critère de performances mesuré. Lorsque l'on compare plusieurs solveurs sur un petit nombre de problèmes-tests (moins de 10) il est facile d'interpréter les différents tableaux de résultats. Cependant, lorsque les comparaisons se font sur de nombreux problèmes-tests l'interprétation des tableaux devient difficile. Dans ce cas, il est intéressant de tracer un profil de performances car celui-ci révèle des informations pertinantes permettant de comparer rapidement les performances de différents codes d'optimisation.

Supposons que l'on doit comparer n_s solveurs sur un banc de n_p problèmes-tests. Nous allons donner ci-dessous un exemple de tracé de profils de performances en prenant le temps de calcul comme critère de performances (on procèdera de façon identique pour d'autres critères de performances). Pour chaque problème p et chaque solveur s, on définit

$$t_{p,s} = \text{ temps de calcul effectué pour résoudre le problème } p \text{ par le solveur } s.$$

Le tableau E.1 montre un exemple synthétique de valeurs prises par $t_{p,s}$ pour $n_s = 3$ solveurs et $n_p = 10$ problèmes-tests. Notons que lorsque la cellule (s, p) du tableau

Temps CPU (s)	Pblme 1	Pblme 2	Pblme 3	Pblme 4	Pblme 5	Pblme 6	Pblme 7	Pblme 8	Pblme 9	Pblme 10
Solveur 1	2,0	4,0	7,0	2,0	7,0	x	4,0	5,0	3,0	1,0
Solveur 2	4,0	9,0	6,0	9,0	8,0	1,0	2,0	17,0	4,0	2,0
Solveur 3	6,0	8,0	2,0	8,0	2,0	3,0	x	12,0	2,0	x
Temps Min.	2,0	4,0	2,0	2,0	2,0	1,0	2,0	5,0	2,0	1,0

Tableau E.1 Exemple synthétique de temps de calcul obtenus sur 3 solveurs différents pour résoudre 10 problèmes-tests.

Ratios de Perf.	Pblme 1	Pblme 2	Pblme 3	Pblme 4	Pblme 5	Pblme 6	Pblme 7	Pblme 8	Pblme 9	Pblme 10
Solveur 1	1,0	1,0	3,5	1,0	3,5	100,0	2,0	1,0	1,5	1,0
Solveur 2	2,0	2,3	3,0	4,5	4,0	1,0	1,0	3,4	2,0	2,0
Solveur 3	3,0	2,0	1,0	4,0	1,0	3,0	100,0	2,4	1,0	100,0

Tableau E.2 Ratios de performances obtenus à partir des données du tableau E.1

contient une croix cela signifie que le solveur s n'est pas parvenu à résoudre le problème p. Dans la dernière ligne de ce tableau, nous avons indiqué le temps minimum sur tous les solveurs pour résoudre un problème-test. Les résultats de ce tableau sont dépendants du problème-test considéré : on ne peut pas comparer les colonnes entre elles. Afin de remédier à ce problème, on calcule les ratios de performances pour chaque problèmes-tests p :

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : 1 \leq s \leq n_s\}}.$$

Le ratio de performances $r_{p,s}$ correspondant à la résolution d'un problème p par un solveur s est égal au rapport de $t_{p,s}$ par le temps minimum effectué par tous les solveurs pour résoudre le problème p. Dans le cas particulier où le solveur s ne parvient pas à résoudre le problème p, on prendra $r_{p,s} = r_M$, où r_M est une constante vérifiant :

$$r_M > r_{p,s}, \quad \forall (p,s) \text{ tel que } t_{p,s} \text{ existe .}$$

Dans le tableau E.2 nous avons calculé les valeurs des ratios de performances à partir des données de l'exemple synthétique du tableau E.1. Notons que nous avons pris $r_M = 100$, ce qui implique que $r_{6,1} = r_{7,3} = r_{10,3} = 100$ (cas des ratios de performances pour les problèmes non résolus).

Le profil de performances du solveur s est alors défini par la fonction de distribution du ratio de performances :

$$f_s(\tau) = \frac{1}{n_p} \text{Size}\{p \in \{1, ..., n_p\} : r_{p,s} \leq \tau\}.$$

Si on trace la fonction $f_s(\tau)$ pour $\tau \in [1, \bar{r}]$, où

$$\bar{r} = \max\{r_{p,s} \neq r_M : 1 \leq s \leq n_s \text{ et } 1 \leq p \leq n_p\},$$

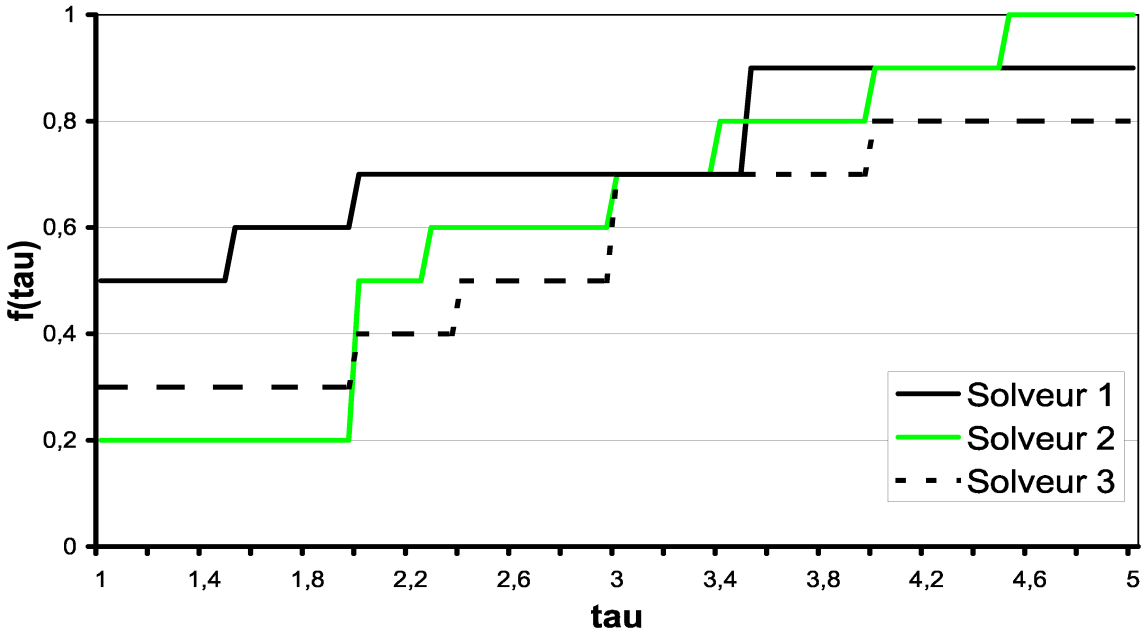

Fig. E.1 Profils de performances des solveurs 1, 2 et 3 obtenus à partir des données du tableau E.2

on obtient alors le profil de performances du solveur s. Dans la figure E.1, nous avons tracé les trois profils de performances correspondant aux trois solveurs différents de notre exemple synthétique (voir tableau E.1).

Voici ci-dessous les principales caractéristiques des profils de performances :

1. Le profil de performances $f_s(\tau) : [1, \bar{r}] \mapsto [0, 1]$ du solveur s est une fonction non décroissante continue par morceaux (elle est continue à droite aux points de discontinuité). Notons que dans notre exemple synthétique $\bar{r}$ vaut $4,5$.
2. La valeur $f_s(1)$ donne le pourcentage de problèmes-tests pour lequel le solveur s a été le plus rapide par rapport aux autres solveurs. En comparant les valeurs de $f_s(1)$ pour les différents solveurs étudiés on peut savoir quel solveur a la probabilité la plus importante d'être le plus rapide. Pour notre exemple synthétique (voir figure E.1), le solveur 1 a la probabilité la plus importante d'être le plus rapide puisqu'il est en tête sur 50% des problèmes-tests (les solveurs 2 et 3 ne sont les plus rapide que sur respectivement 20% et 30% des problèmes-tests).
3. La valeur $f_s(\bar{r})$ donne le pourcentage de problèmes-tests résolus par le solveur s. En comparant les valeurs de $f_s(\bar{r})$ pour les différents solveurs étudiés, on peut savoir lequel est le plus robuste (c'est à dire celui qui parvient à résoudre le plus de problèmes-tests). Dans notre exemple synthétique (voir figure E.1) c'est le solveur 2 qui est le plus robuste car il résout 100% des problèmes-tests alors que le solveur 1 et le solveur 3 ne resolvent respectivement que 90% et 80% des problèmes-tests.
4. La comparaison des valeurs de τ prisent par les différents profils de performances à une même hauteur permet de savoir quel est le solveur le plus efficace pour

résoudre $X\%$ des problèmes-tests (correspondant au τ le plus faible). Ainsi dans notre exemple synthétique, on peut par exemple remarquer que le solveur 2 (resp. 1) est le plus efficace pour résoudre 80% (resp. 90%) des problèmes-tests. On peut aussi noter que le solveur 3 n'est jamais plus efficace que les deux autres et ce quelque soit le pourcentage de problèmes-tests considéré.

Annexe F

Publications et communications

Publications

2005 : F. Delbos et J. Ch. Gilbert,
"Global linear convergence of an augmented Lagrangian algortihm for solving convex quadratic optimization problems", accepté pour publication dans Journal of convex analysis.
2004 : F. Delbos, J. Ch. Gilbert, R. Glowinski, D. Sinoquet,
"Constrained optimization in reflection tomography : an augmented Lagrangian SQP approach", soumis pour Geophysical Journal International.
2004 : D. Sinoquet, F. Delbos, J. Ch. Gilbert,
"A dedicated constrained optimization method for 3D reflection tomography", 66^{th} Conference and Technical Exhibition, EAGE, expanded abstracts.

Communications

2003 : F. Delbos et al., *"Application of an SQP augmented Lagrangian method to a large-scale problem in 3D reflection tomography"*, ISMP meeting, Copenhague.
2002 : F. Delbos et al., *"Trust-region Gauss-Newton method for 3D reflection tomography"*, SIAM meeting on optimization, Toronto.

Références

[1] K. Aki, A. Christofferson, E. S. Husebye (1977). Determination of the three-dimensional seismic structures of the lithosphere. *J. Geophys. Res.*, pages 277–296.

[2] M. Al-Chalabi (1992). When least-squares squares least. *Geophysical Prospecting*, 40, 359–378.

[3] F. Al-Kkayyal, J. Kiparisis (1990). Finite convergence of algorithms for nonlinear programs and variational inequalitites. *Journal of Optimization Theory and Applications*, 70, 319–332.

[4] P.L. De Angelis, G. Toraldo (1993). On the identification property of a projected gradient method. *SIAM Journal on Numerical Analysis*, 30, 1483–1497.

[5] K.J. Arrow, R.M. Solow (1958). Gradient methods for constrained maxima with weakened assumptions. In K.J. Arrow, L. Hurwicz, H. Uzawa, éditeurs, *Studies in Linear and Nonlinear Programming*, pages 166–176. Stanford University Press, Standford, Calif.

[6] A. Auslender, M. Teboulle (2000). Lagrangian duality and related multiplier methods for variational inequality problems. *SIAM Journal on Optimization*, 10, 1097–1115.

[7] A. Auslender, M. Teboulle, S. Ben-Tiba (1999). Interior proximal and multiplier methods based on second order homogeneous functionals. *Mathematics of Operations Research*, 24, 645–668.

[8] G. E. Backus, F. Gilbert (1970). Uniqueness in the inversion of inaccurate gross data. *Phil. Trans. Roy. Soc. London*, 266, 123–192.

[9] A. Bamberger, G. Chavent, C. Hemon, P. Lailly (1982). Inversion of normal incidence seismograms. *Geophysics*, 47(5), 757–770.

[10] B. A. Bartels, J. C. Beatty, R. H. Barsky (1987). *An introduction to splines for use in computer graphics and geometry modeling*. Morgan Kaufmann Publishers.

[11] A. Ben-Menahem, W. B. Beydoun (1985). Range of validity of seismic ray and beam methods in general inhomogeneous media - 1.General theory. *Geophys. J. Roy. Ast. Soc.*, 82, 207–234.

[12] D.P. Bertsekas (1976). On the Goldstein-Levitin-Polyak gradient projection method. *IEEE Transactions on Automatic Control*, 21, 174–184.

[13] D.P. Bertsekas (1995). *Nonlinear Programming*. Athena Scientific.

[14] F. Billette, G. Lambaré (1998). velocity macro model estimation by stereotomography. *Geophysical Journal International*, 135(2), 671–680.

[15] T. N. Bishop, K. P. Bube, R. T. Cutler, R. T. Langan, P. L. Love, J. R. Resnick, R. T. Shuey, D. A. Spindler, H. W. Wyld (1985). Tomographic determination of velocity and depth in laterally varying media. *Geophysics*, 50(6), 903–923.

[16] Å. Björck (1996). *Numerical Methods for Least Squares Problems*. SIAM Publication, Philadelphia.

[17] P. Bois, M. La-Porte, M. Lavergne, G. Thomas (1971). Essai de détermination automatique des vitesses sismiques par mesures entre puits. *Geophys. Prospect.*, 19, 42–83.

[18] J.F. Bonnans, J. Ch. Gilbert, C. Lemaréchal, C. Sagastizábal (2003). *Numerical Optimization*. Springer.

[19] A. Bourgeois, P. Lailly, F. Collino, F. Dumont, D. Macé (1989). The linearized seismic inverse problem : an attractive method for a sharp description of strategic traps. *59^{th} Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstract*, pages 973–976.

[20] J. Brac, P.Y. Dequirez, H. Hervé, C. Jacques, P. Lailly, V. Richard, D. Tran-Van-Nhieu (1992). Inversion with a priori information : an approach to integrated stratigraphic interpretation. In R.E. Sheriff, éditeur, *Reservoir Geophysics, Investigations in Geophysics,SEG publication*, Tome 7, pages 251–258.

[21] K. Broto (1999). *Accès à l'information cinématique pour la détermination du modèle de vitesse par tomographie de réflexion 3D (in French)*. Thèse de doctorat, Université de Pau et des Pays de l'Adour, France.

[22] K. P. Bube, R. T. Langan (1994). A continuation approach to regularization for traveltime tomography. *64^{th} Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 980–983.

[23] J.V. Burke, J.J. Moré (1988). On the identification of active constraints. *SIAM Journal on Numerical Analysis*, 25, 1197–1211.

[24] J.V. Burke, J.J. Moré (1994). Exposing constraints. *SIAM Journal on Optimization*, 4, 573–595.

[25] R.H. Byrd, J.Ch. Gilbert, J. Nocedal (2000). A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89, 149–185.

[26] R.H. Byrd, M.E. Hribar, J. Nocedal (1999). An interior point algorithm for large scale nonlinear programming. *SIAM Journal on Optimization*, 9, 877–900.

[27] M. Cavalca, P. Lailly (2004). A traveltime tomography dedicated to the inversion of prismatic reflections. In *Expanded Abstracts*. EAGE.

[28] V. Červený, I. A. Molotkov, I. Pšenčík (1977). *Ray method in seismology*. Univerzita Karlova, Praha.

[29] R. Chander (1977). On tracing seismic rays with specified end points in layers of constant velocity and plane interfaces. *Geophysical Prospecting*, 25, 120–124.

[30] L. Chauvier, R. Masson, D. Sinoquet (2000). Implementation of an efficient preconditioned conjugate gradient in jerry. *KIM Annual Report, Institut Français du Pétrole, Rueil-Malmaison, France*.

[31] J. F. Claerbout (1976). *Fundamentals of geophysical data processing*. Mc Graw–Hill Book Co.

[32] R. A. Clarke (1996). Raytracing and tomography friendly geological models. *KIM Annual Report, Institut Français du Pétrole, Pau, France*.

[33] R. A. Clarke (1997). *Modélisation et inversion de données cinématiques complexes en 3D (in English)*. Thèse de doctorat, Université de Pau et des Pays de l'Adour, France.

[34] I. Comby (2004). Etude de méthodes d'optimisation pour la tomographie de réflexion (rapport de stage). *IFP Report, Institut Français du Pétrole, Rueil-Malmaison, France*, (58072).

[35] A.R. Conn, N. Gould, P.L. Toint (2000). *Trust-Region Methods*. MPS/SIAM Series on Optimization. SIAM and MPS.

[36] A.R. Conn, N.I.M. Gould, Ph.L. Toint (1988). Testing a class of methods for solving minimization problems with simple bounds on the variables. *Mathematics of Computation*, 50, 399–430.

[37] A.R. Conn, N.I.M. Gould, Ph.L. Toint (1992). *LANCELOT : A Fortran Package for Large-Scale Nonlinear Optimization (Release A)*. Computational Mathematics 17. Springer Verlag, Berlin.

[38] S.C. Constable, R. L. Parker, C. G. Constable (1987). Occam's inversion : A practical algorithm for generating smooth models from electromagnetic sounding data. *Geophysics*, 52, 289–300.

[39] Ph. Cote, R. Lagabrielle (1986). La tomographie sismique comme méthode de reconnaissance détaillée du sous-sol. *Rev. Franç. Géotech.*, 36, 47–53.

[40] R. Courant (1943). Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the American Mathematical Society*, 49, 1–23.

[41] P. Courtier, O. Talagrand (1987). Variational assimilation of meteorological observations with the adjoint vorticity equation. II : Numerical results. *Quaterly Journal of the Royal Meteorological Society*, 113, 1329–1347.

[42] P. Courtier, J.N. Thépaut, A. Hollingsworth (1994). A strategy for operational implementation of 4D-Var, using an incremental approach. *Quaterly Journal of the Royal Meteorological Society*, 120, 1367–1387.

[43] G.B. Dantzig (1951). Maximization of a linear function of variables subject to linear inequalities. In Tj.C. Koopmans, éditeur, *Activity Analysis of Production and Allocation*, pages 339–347. Wiley, New York.

[44] C. de Boor (1978). *A Practical Guide to Splines*. Springer.

[45] F. Delbos, J. Ch. Gilbert (2003). Global linear convergence of an augmented lagrangian algorithm for solving convex quadratic optimization problems. *Journal of Convex Analysis*.

[46] F. Delbos, D. Sinoquet, J. Ch. Gilbert, R. Masson (2001). Trust-region gauss-newton method for reflection tomography. *KIM Annual Report, Institut Français du Pétrole, Rueil-Malmaison, France*.

[47] F. Delprat-Jannaud (1991). *Tomographie de reflection : quelle est l'information contenue dans les temps d'arrivée ? (in English)*. Thèse de doctorat, Université de Paris-Sud Orsay.

[48] F. Delprat-Jannaud, P. Lailly (1993). Ill-posed and well-posed formulations of the reflection travel time tomography problem. *J. Geophys. Res.*, 98, 6589–6605.

[49] J.E. Dennis, D.M. Gay, R.E. Welsch (1981). An adaptive nonlinear least squares algorithm. *ACM Transactions on Mathematical Software*, 7, 348–368.

[50] E.D. Dolan, J.J. Moré (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91, 201–213.

[51] C. Duffet, D. Sinoquet (2002). Quantifying geological uncertainties in velocity model building. In *Expanded Abstracts*, pages 926–929. Soc. Expl. Geophys.

[52] A. Ehinger, F. Jurado, Ch. Monestié (1998). Calculation of multi-valued traveltimes with a ray bending method. In *Expanded Abstracts*, pages 1893–1896. Soc. Expl. Geophys.

[53] H. W. Engl, M. Hanke, A. Neubauer (1996). *A Regularization of inverse problems*. Kluwer, Dordrecht.

[54] V. Farra, R. Madariaga (1988). Non-linear reflection tomography. *Geophys. J. Roy. Ast. Soc.*, 95, 135–147.

[55] A.V. Fiacco, G.P. McCormick (1968). *Nonlinear Programming : Sequential Unconstrained Minimization Techniques*. John Wiley, New York.

[56] R. Fletcher (1980). *practical methods of optimization, I : Unconstrained optimization.* J. Wiley and sons.

[57] R. Fletcher (1980). *Practical Methods of Optimization. Volume 1 : Unconstrained Optimization*. John Wiley & Sons, Chichester.

[58] R. Fletcher (1981). *Practical Methods of Optimization. Volume 2 : Constrained Optimization*. John Wiley & Sons, Chichester.

[59] R. Fletcher, S. Leyffer (2002). Nonlinear programming without a penalty function. *Mathematical Programming*, 91, 239–269.

[60] A. Forsgren, P.E. Gill (1998). Primal-dual interior methods for nonconvex nonlinear programming. *SIAM Journal on Optimization*, 8, 1132–1152.

[61] M. Fortin, R. Glowinski (1982). *Méthodes de Lagrangien augmenté*. Bordas, Paris, Dunod edition.

[62] M. Fortin, R. Glowinski (1982). *Méthodes de Lagrangien Augmenté – Applications à la Résolution Numérique de Problèmes aux Limites*. Méthodes Mathématiques de l'Informatique 9. Dunod.

[63] A. Friedlander, J.M. Martínez (1994). On the maximization of a concave quadratic function with box constraints. *SIAM Journal on Optimization*, 4, 177–192.

[64] A. Friedlander, J.M. Martínez, S.A. Santos (1994). On the resolution of linearly constrained convex minimization problems. *SIAM Journal on Optimization*, 4, 331–339.

[65] K.R. Frisch (1955). The logarithmic potential method for convex programming. Rapport de recherche, Institute of Economics, University of Oslo, Oslo, Norway.

[66] E.M. Gafni, D.P. Bertsekas (1984). Two-metric projection methods for constrained optimization. *SIAM Journal on Control and Optimization*, 22, 936–964.

[67] D.M. Gay, M.L. Overton, M.H. Wright (1998). A primal-dual interior method for nonconvex nonlinear programming. In Y.-X. Yuan, éditeur, *Advances in Nonlinear Programming*. Kluwer Academic Publishers.

[68] J.Ch. Gilbert (2003). *Optimisation Différentiable – Théorie et Algorithmes, Syllabus de cours à l'ENSTA*. http ://www-rocq.inria.fr/∼gilbert/ensta/optim.html.

[69] J.Ch. Gilbert, C. Lemaréchal (1989). Some numerical experiments with variable-storage quasi-Newton algorithms. *Mathematical Programming*, 45, 407–435.

[70] P.E. Gill, W. Murray, M.A. Saunders (1996). SNOPT : an SQP algorithm for large-scale constrained optimization. Numerical Analysis Report 96-2, Department of Mathematics, University of California, San Diego, La Jolla, CA.

[71] P.E. Gill, W. Murray, M.A. Saunders, M.H. Wright (1986). User's guide for NPSOL (version 4.0) : a Fortran package for nonlinear programming. Technical Report SOL-86-2, Department of Operations Research, Stanford University, Stanford, CA 94305.

[72] P.E. Gill, W. Murray, M.H. Wright (1981). *Practical Optimization*. Academic Press, New York.

[73] D.A. Girard (1986). A fast 'monte-carlo cross-validation' procedure for large least squares problems with noisy data. *Numer. Math.*, 56, 1–23.

[74] R. Glowinski, P. Le Tallec (1989). *Augmented Lagrangian and Operator Splitting Methods in Nonlinear Mechanics*. Studies in Applied Mathematics 9. SIAM, Philadelphia.

[75] R. Glowinski, Q.H. Tran (1993). Constrained optimization in reflection tomography : the Augmented Lagrangian method. *East-West J. Numer. Math.*, 1, 213–234.

[76] D. Goldfarb, A. Idnani (1983). A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming*, 27, 1–33.

[77] A.A. Goldstein (1964). Convex programming in Hilbert space. *Bulletin of the American Mathematical Society*, 70, 709–710.

[78] E.G. Gol'shteĭn, N.V. Tretyakov (1996). *Modified Lagrangians and Monotone Maps in Optimization*. Discrete Mathematics and Optimization. John Wiley & Sons, New York.

[79] G. H. Golub, M. T. Heath, G. Wahba (1979). Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2), 215–223.

[80] G. H. Golub, C. F. Van Loan (1989). *Matrix Computations*, Tome 2. The John Hopkins University Press, Baltimore.

[81] N. Gould, D. Orban, A. Sartenaer, Ph.L. Toint (2000). Superlinear convergence of primal-dual interior point algorithms for nonlinear programming. *Mathematical Programming*, 87, 215–249.

[82] N. Gould, Ph.L. Toint (2004). Preprocessing for quadratic programming. *Mathematical Programming*.

[83] N.I.M. Gould (1989). On the convergence of a sequential penalty function method for constrained minimization. *SIAM Journal on Numerical Analysis*, 26, 107–126.

[84] J. L. Guiziou (1993). *Estimation de modèles de vitesse en 3D par tomographie de sismique réflexion : développements d'outils sur la base du modeleur géométrique Gocad*. Thèse de doctorat, Université de Paris VII.

[85] J. L. Guiziou, J. L. Mallet, P. Nobili, R. Anandappane, P. Thisse (1991). 3–D ray tracing through complex triangulated surfaces. *61^{st} Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts*, pages 1497–1500.

[86] S.-P. Han (1976). Superlinearly convergent variable metric algorithms for general nonlinear programming problems. *Mathematical Programming*, 11, 263–282.

[87] S.-P. Han (1977). A globally convergent method for nonlinear programming. *Journal of Optimization Theory and Applications*, 22, 297–309.

[88] M. Hanke (1996). Limitations of the l-cirve method in ill-posed problems. *BIT*, 36(2), 287–301.

[89] M. Hanke, P.C. Hansen (1993). Regularization methods for large scale problems. *Surv. Math. Ind.*, 3, 253–315.

[90] P. C. Hansen (1992). Analysis of discrete ill-posed problems by mean of the L-curve. *SIAM Review*, 34, 561–580.

[91] P. C. Hansen (1996). Rank-deficient and discrete ill-posed problems. *Doctoral Dissertation, UNI•C, Technical University of Denmark.*

[92] P. C. Hansen, D.P. O'Leary (1993). The use of the l-curve in the regularization of discrete ill-posed problems. *SIAM J. Sci. Comput.*, 14(6), 1487–1503.

[93] M.R. Hestenes (1969). Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4, 303–320.

[94] J.-B. Hiriart-Urruty, C. Lemaréchal (1993). *Convex Analysis and Minimization Algorithms*. Grundlehren der mathematischen Wissenschaften 305-306. Springer-Verlag.

[95] L. R. Jannaud (1992). Two-point raytracing in complex media. *PSI Annual Report, Institut Français du Pétrole, Rueil-Malmaison, France*.

[96] B. R. Julian, D. Gubbins (1977). Three-dimensional seismic raytracing. *Journal of Geophysics*, 43, 95–113.

[97] F. Jurado, P. Lailly, A. Ehinger (1998). Fast 3D two-point raytracing for traveltime tomography. *Proceedings of SPIE, Mathematical Methods in Geophysical Imaging V*, 3453, 70–81.

[98] F. Jurado, D. Sinoquet, A. Ehinger (1996). 3D reflection tomography designed for complex structures. In *Expanded Abstracts*, pages 711–714. Soc. Expl. Geophys.

[99] F. Jurado, D. Sinoquet, A. Ehinger, P. Lailly, Ch. Monestié (1997). Improvements in the reflection tomography software jerry. *KIM Annual Report, Institut Français du Pétrole, Pau, France*.

[100] F. Jurado, D. Sinoquet, P. Lailly (1996). Jerry : a 3D reflection tomography designed for complex structures. *KIM Annual Report, Institut Français du Pétrole, Pau, France.*

[101] N. Karmarkar (1984). A new polynomial-time algorithm for linear programming. *Combinatorica*, 4, 373–395.

[102] P. Kolb, F. Collino, P. Lailly (1986). Pre–stack inversion of a 1d medium. *Proc. IEEE*, 74, 498–508.

[103] P. Lailly, D. Sinoquet (1996). Smooth velocity models in reflection tomography for imaging complex geological structures. *Geophys. J. Int.*, 124, 349–362.

[104] G. Lambaré, J. Virieux, R. Madariaga, S. Jin (1992). Iterative asymptotic inversion in the acoustic approximation. *Geophysics*, 57(9), 1138–1154.

[105] R. T. Langan, I. Lerche, R. T. Cutler (1985). Tracing of rays through heterogeneous media : an accurate and efficient procedure. *Geophysics*, 50, 1456–1465.

[106] C. L. Lawson, R.J. Hanson (1974). *Solving least-squares problems*. Prentice-hall.

[107] C. Lemaréchal (2001). Lagrangian relaxation. Rapport de recherche, INRIA. `http ://www.optimization-online.org/DB_HTML/2001/03/298.html`.

[108] C. Lemarechal, E. Panier (1983). Modulopt. Rapport de recherche, INRIA, Rocquencourt, France.

[109] K. Levenberg (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of Appl. Math.*, 2, 164–168.

[110] E.S. Levitin, B.T. Polyak (1966). Constrained minimization problems. *USSR Comput. Math. Math. Phys.*, 6, 1–50.

[111] L. R. Lines, S. Treitel (1984). Tutorial : A review of least–squares inversion and its application to geophysical problems. *Geophysical prospecting*, 32, 159–186.

[112] D.W. Marquardt (1963). An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Indust. Appl. Math.*, 11, 431–441.

[113] D.Q. Mayne, E. Polak (1982). A superlinearly convergent algorithm for constrained optimization problems. *Mathematical Programming Study*, 16, 45–61.

[114] W. Menke (1984). *Geophysical data analysis : Discrete Inverse Theory*. Academic press.

[115] L. Mintrop (1931). *On the history of the seismic method for the investigation of the underground formations and mineral deposits*. Seismos, Hanovre.

[116] J. J. More (1978). The Levenberg-Marquardt algorithm : Implementation and theory. *Numerical analysis*, 630, 105–116.

[117] J.J. Moré, G. Toraldo (1989). Algorithms for bound constrained quadratic programming problems. *Numerische Mathematik*, 55, 377–400.

[118] J.J. Moré, G. Toraldo (1991). On the solution of large quadratic programming problems with bound constraints. *SIAM Journal on Optimization*, 1, 93–113.

[119] V. A. Morozov (1966). On the solution of functional equations by the method of regularization. *Sov. Math. Doklady*, 167(3), 414–417.

[120] Z. Nashed (1974). Approximate regularized solutions to improperly posed linear integral and operator eqution. *Constructive and computational methods for differential equations*.

[121] M. Noble, A. Tarantola (1991). Nonlinear waveform inversion of seismic reflection data : the marmousi model. pages 113–125.

[122] J. Nocedal (1980). Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35, 773–782.

[123] J. Nocedal, S. J. Wright (1999). *Numerical Optimization*. Springer.

[124] U.M. Garcia Palomares, O.L. Mangasarian (1976). Superlinear convergent quasi-Newton algorithms for nonlinearly constrained optimization problems. *Mathematical Programming*, 11, 1–13.

[125] W.S. Phillips, M.C. Fehler (1991). Traveltime tomography : A comparison of popular methods. *Geophysics*, 56, 1639–1649.

[126] M.J.D. Powell (1969). A method for nonlinear constraints in minimization problems. In R. Fletcher, éditeur, *Optimization*, pages 283–298. Academic Press, London, New York.

[127] M.J.D. Powell (1973). On search directions for minimization algorithms. *Mathematical Programming*, 9, 193–201.

[128] M.J.D. Powell (1976). Some global convergence properties of a variable metric algorithm for minimization without exact line searches. In R.W. Cottle, C.E. Lemke, éditeurs, *Nonlinear Programming*, SIAM-AMS Proceedings 9. American Mathematical Society, Providence, RI.

[129] M.J.D. Powell (1978). Algorithms for nonlinear constraints that use Lagrangian functions. *Mathematical Programming*, 14, 224–248.

[130] M.J.D. Powell (1978). The convergence of variable metric methods for nonlinearly constrained optimization calculations. In O.L. Mangasarian, R.R. Meyer, S.M. Robinson, éditeurs, *Nonlinear Programming 3*, pages 27–63. Academic Press, New York.

[131] M.J.D. Powell (1978). A fast algorithm for nonlinearly constrained optimization calculations. In G.A. Watson, éditeur, *Numerical Analysis Dundee 1977*, Lecture Notes in Mathematics 630, pages 144–157. Springer-Verlag, Berlin.

[132] F. Renard (2002). *Prise en compte des corrélations dans l'inversion de données sismiques.* Thèse de doctorat, Université de Monpellier.

[133] R.T. Rockafellar (1971). New applications of duality in convex programming. In *Proceedings of the 4th Conference of Probability, Brasov, Romania*, pages 73–81. (version écrite d'un exposé donné à différentes conférences, en particulier au "7th International Symposium on Mathematical Programming", La Haye, 1970).

[134] R.T. Rockafellar (1973). A dual approach to solving nonlinear programming problems by constrained optimization. *Mathematical Programming*, 5, 354–373.

[135] R.T. Rockafellar (1973). The multiplier method of Hestenes and Powell applied to convex programming. *Journal of Optimization Theory and Applications*, 6, 555–562.

[136] R.T. Rockafellar (1974). Augmented Lagrange multiplier functions and duality in nonconvex programming. *SIAM Journal on Control*, 12, 268–285.

[137] J.B. Rosen (1960). The gradient projection method for nonlinear programming ; part I : linear constraints. *J. Soc. Indust. Appl. Math.*, 8, 181–217.

[138] J. A. Scales, P. Docherty, A. Gersztenkorn (1990). Regularisation of nonlinear inverse problems : Imaging the near-surface weathering layer. *Inverse Problems*, 6, 115–131.

[139] K. Schittkowski (1985). NLPQL : a FORTRAN subroutine solving constrained nonlinear programming problems. *Annals of Operations Research*, 5, 485–500.

[140] D. Sinoquet (1995). *Utilisation de modèles lisses pour l'inversion tomographique de données sismiques (in English)*. Thèse de doctorat, Université Paris XIII, France.

[141] H. Stark (1987). *Image and Recovery : Theory and Application*. Academic Press, San Diego.

[142] T. Steihaug (1983). The conjugate gradient method and trust regions in large scale optimization. *SIAM Journal on numerical analysis*, 20, 626–637.

[143] G.W. Stewart (1973). *Introduction to matrix Computations*. Academic Press, New-York.

[144] A. Stopin (2001). *Détermination de modèle de vitesses anisotropes par tomographie de réflexion des modes de compression et de cisaillement (in English)*. Thèse de doctorat, Université de Strasbourg I, France.

[145] A. Tarantola (1984). Linearized inversion of seismic reflection data. *Geophysical Prospecting*, 32, 998–1015.

[146] A. Tarantola (1987). *Inverse problem theory*. Elsevier.

[147] P. Thisse (1994). *Tracé de rayons et migration 3D pour des interfaces triangulées : conception orientée objets et application à la sismique réflexion profonde*. Thèse de doctorat, Université Louis Pasteur de Strasbourg.

[148] A. Tikhonov, V. Arsenin (1977). *Solution of ill–posed problems*. Winston.

[149] A. Tikhonov, A. V. Goncharski, V. V. Stepanov, A. G. Yagoda (1995). *Numerical methods for the solution of ill-posed problems*. Kluwer Academic Publishers.

[150] T.L. Tonellot (2000). *Introduction d'informations a priori dans l'inversion linéarisée élastique de données sismiques de surface avant sommation*. Thèse de doctorat, Université René Descartes.

[151] R. J. Vanderbei (1999). LOQO : An interior point code for quadratic programming. *Optimization Methods and Software*, 11, 451–484.

[152] J. Virieux, V. Farra (1991). Ray tracing in 3–D complex isotropic media : an analysis of the problem. *Geophysics*, 56(12), 2057–2069.

[153] J. Virieux, V. Farra, R. Madariaga (1988). Ray tracing in laterally heterogeneous media for earthquake location. *J. Geophys. Res.*, 93, 6585–6599.

[154] C. R. Vogel (1996). Non-convergence of the l-curve regularization method for nonlinear ill-posed problems. *Inverse Problems*, 12, 535–547.

[155] A. Wächter (2002). *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering*. Thèse de doctorat, Carnegie Mellon University.

[156] R.B. Wilson (1963). *A simplicial algorithm for concave programming*. Thèse de doctorat, Graduate School of Business Administration, Harvard University, Cambridge, MA, USA.

[157] S.J. Wright (1993). Identifiable surfaces in constrained optimization. *SIAM Journal on Control and Optimization*, 31, 1063–1079.

[158] S.J. Wright (1997). *Primal-Dual Interior-Point Methods*. SIAM Publication, Philadelphia.

[159] H. Yabe, N. Yamaki (1995). Convergence of a factorized Broyden-like family for nonlinear least squares problems. *SIAM Journal on Optimization*, 5, 770–790.

[160] O. Yilmaz (1987). *Seismic data processing*. Society of Exploration Geophysicists.