Présentation de la méthode d'identification

Sommaire

8.1 Introduction

Grâce au recalage d'images déformables, nous sommes à présent en mesure de déformer une image numériquement et de la comparer à une image de type "expérimentale". Cette comparaison est faite par la définition d'une *fonction coût* qui est une fonction décrivant l'écart entre les 2 images. L'identification étant la détermination du jeu de paramètres des lois de comportement permettant de faire correspondre ces deux images, elle est le résultat d'une minimisation de la fonction coût.

Dans ce chapitre, nous nous intéressons à deux algorithmes d'optimisation qui sont l'algorithme de **Newton-Raphson** et l'algorithme de **Nelder-Mead**. Après avoir décrit le principe de ces algorithmes, nous allons les appliquer à un cas-test afin d'évaluer les difficultés qui pourront être rencontrées lors de l'identification des propriétés mécaniques des tissus biologiques mous.

8.2 Algorithmes d'optimisation envisagés

Le but de l'identification étant de trouver un jeu de paramètres minimisant la fonction coût définie précédemment, l'utilisation d'algorithmes d'optimisation est nécessaire. Nous avons choisi de ne présenter que deux algorithmes de différents types. Une méthode d'ordre 1, soit l'algorithme de **Newton-Raphson** [Jol99] et une méthode d'ordre 0, soit l'algorithme du **Simplexe** ou algorithme de **Nelder-Mead** [NM65],[Vay04].

8.2.1 Algorithme de Newton-Raphson

L'algorithme de Newton-Raphson est un algorithme de Newton multidimensionnel [Jol99]. C'est une méthode d'ordre 1 *i.e.* utilisant les gradients de la fonction coût. Cette méthode est choisie pour sa simplicité d'implémentation dans Matlab® et sa robustesse. Soit $f_c(X)$ la fonction coût à minimiser et X le vecteur constitué des paramètres à identifier, on cherche à résoudre,

$$\frac{df_c(X)}{dX} = 0 \tag{8.1}$$

Par un développement de Taylor au premier ordre au voisinage de X^n on a,

$$\left.\frac{df_c(X)}{dX}\right|_{X^{n+1}} = \left.\frac{df_c(X)}{dX}\right|_{X^n} + \left.\frac{d^2f_c(X)}{dX^2}\right|_{X^n} (X^{n+1} - X^n) \tag{8.2}$$

On peut déterminer X^{n+1} tel que l'Équation (8.2) soit nulle,

$$X^{n+1} = X^n - H^{-1} \left.\frac{df_c(X)}{dX}\right|_{X^n} \tag{8.3}$$

où H est la matrice *hessienne*. On rappelle que $H_{ij}^n = \left.\frac{d^2 f_c(X_i)}{(dX_j^n)^2}\right|_{X^n}$. Les gradients de la fonction coût sont obtenus par différences finies centrée à l'ordre 1.
À chaque itération, on determine le nouveau jeu de paramètres grâce à celui de l'itération précédente. Dans certains cas, il peut être utile d'introduire un coefficient d'amortissement d tel que,

$$X^{n+1} = X^n - d * H^{-1} \left.\frac{df_c(X)}{dX}\right|_{X^n} \qquad \text{avec} \qquad d < 1 \tag{8.4}$$

Le coefficient d'amortissement permet la diminution de la distance entre deux jeux de paramètres successifs. On s'approchera de manière moins rapide de la solution mais cela permet d'éviter l'éloignement de la solution. On peut représenter graphiquement l'algorithme de Newton-Raphson à une seule dimension [Figure 8.1]. Comme il s'agit

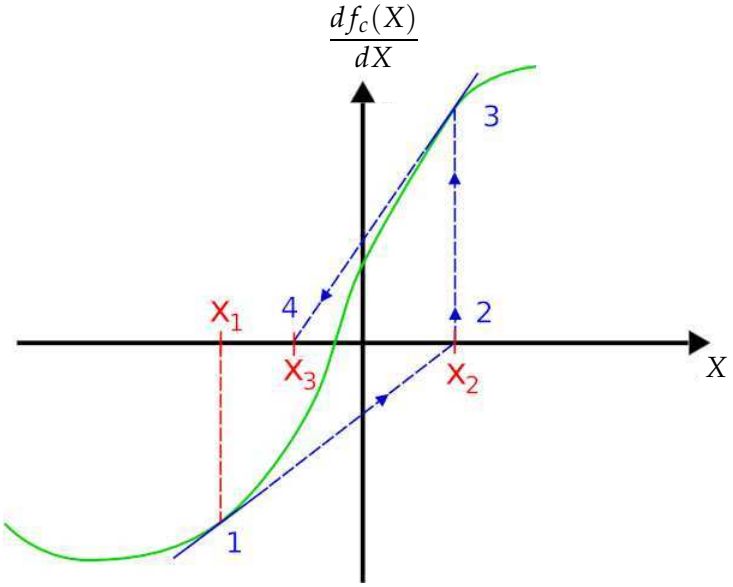


Figure 8.1 – Représention graphique de l'algorithme de Newton-Raphson

d'une procédure itérative, il est nécessaire de définir un ou plusieurs critères d'arrêt. Nous nous munissons de quatre critères.

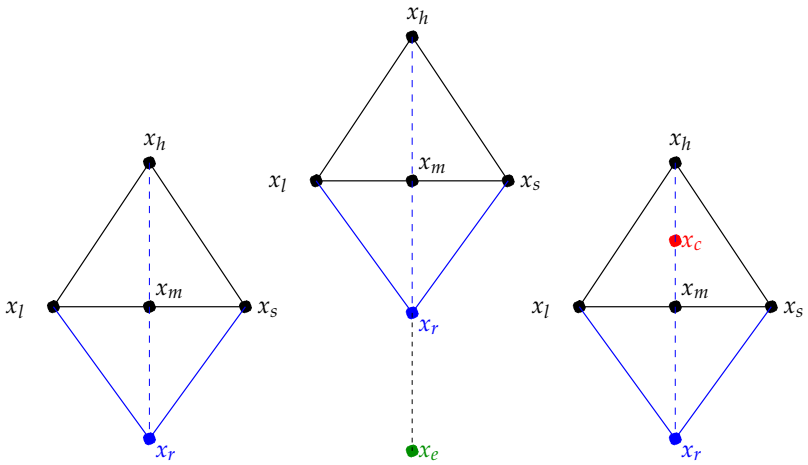
- **critère 1 :** valeur de la fonction coût en deçà de laquelle elle est considérée satisfaisante
- **critère 2 :** nombre d'itérations du calcul d'optimisation
- **critère 3 :** différence normalisée entre 2 valeurs de la fonction coût successives
- **critère 4 :** distance normalisée entre 2 jeux de paramètres testés successifs.

Dans le cas d'identification de nombreux paramètres, l'algorithme de Newton-Raphson est relativement rapide.

8.2.2 Algorithme de Nelder-Mead

La méthode de Nelder-Mead [NM65] permet une optimisation en s'affranchissant du calcul des gradients. Cette méthode est articulée autour de trois étapes qui sont la réflexion, la contraction et l'expansion. En dimension n, un simplexe est un polygone à $n+1$ sommets. Le principe est de déplacer le simplexe dans le domaine en remplaçant itérativement le plus mauvais point par un point meilleur.

On définit un simplexe à trois sommets (un triangle), on appelle x_h le sommet pour lequel la fonction coût est maximale $f_c(x_h) = f_h$, x_s le sommet où la fonction coût prend la deuxième plus grande valeur $f_c(x_s) = f_s$, x_l le sommet où la fonction coût est minimale $f_c(x_l) = f_l$. Le barycentre entre les points x_s et x_l est défini par x_m. La première étape de l'algorithme est la **réflexion**, on cherche à s'éloigner de x_s en créant x_r un nouveau sommet tel que $x_r = x_m + a(x_m - x_s)$ [Figure 8.2-*(a)*]. Soit $f_r = f_c(x_r)$, si



(a) Réflexion *(b)* Expansion *(c)* Contraction

Figure 8.2 – Les différents étapes de l'algorithme de Nelder-Mead en 2D

$f_l < f_r < f_s$, on remplace x_s par x_r. Si f_r est meilleur que le miminum actuel *i.e.* $f_r < f_l$, on essaye d'aller plus loin en créant x_e par l'**expansion** tel que $x_e = x_m + b(x_r - x_m)$ [Figure 8.2-*(b)*]. Mais si x_r n'est pas meilleur que x_s, on cherche à se rapprocher de x_l en effectuant une **contraction** *i.e.* en créant x_c tel que $x_c = x_m + c(x_h - x_m)$ [Figure 8.2-*(c)*]. Pour plus de détails concernant cet algorithme, nous invitons le lecteur à se référer à la [Figure 8.3]. D'une manière générale, les paramètres a, b et c sont 1, 2, 0,5 respectivement.

L'algorithme de Nelder-Mead est une méthode efficace et robuste mais elle est à proscrire dans le cas où de nombreux paramètres sont à déterminer (>10), on observerait alors une explosion du temps de convergence [Vay04]. On retrouve cet algorithme implémenté dans la fonction *fminsearch* de Matlab®.

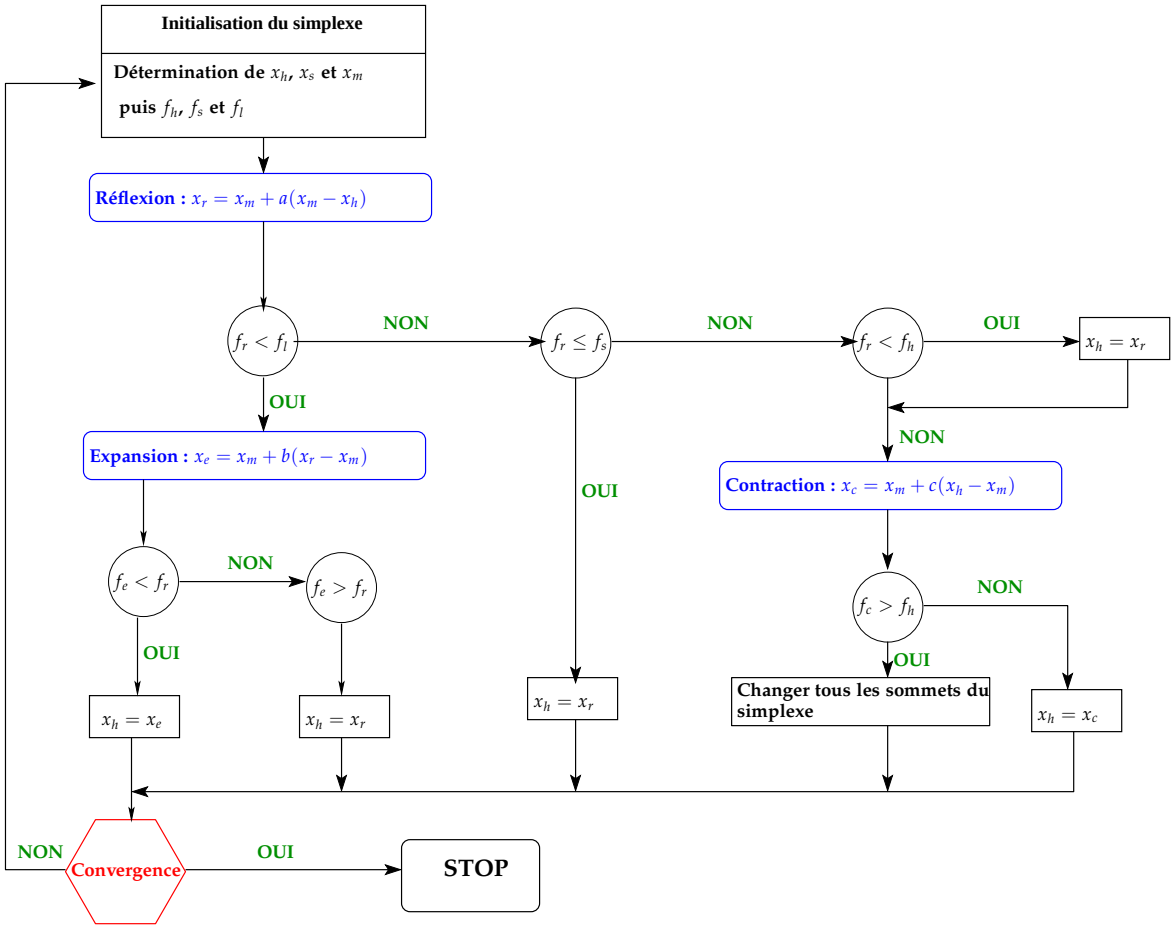


Figure 8.3 – Représention graphique de l'algorithme de Nelder-Mead

8.3 Application de la méthode d'identification à un cas-test

La méthode d'identification doit être testée avant d'être appliquée sur le cas réel. Nous choisissons pour ce faire une méthode de *blind test* [Del07] qui nous permettra de valider la méthode et de réaliser des observations utiles pour la suite du travail.

8.3.1 La méthode de *blind test*

On rappelle que l'identification proposée consiste à comparer une image non-déformée expérimentale (image cible) et une image non-déformée simulée. Afin de vérifier la validité de notre méthode d'identification, on utilise une méthode de *blind test*. Grâce à un calcul EF avec des paramètres de comportement fixés, on génère l'image déformée correspondante (utilisée alors comme image déformée expérimentale). Lors du processus d'identification, on applique à cette image les transformations inverses issues des différents calculs EF afin que la géométrie retrouve sa configuration non-déformée. On compare alors cette dernière avec l'image de la géométrie initiale (image non-déformée initiale ou image cible).

8.3.2 Le modèle numérique

Le modèle EF utilisé est celui présenté pour la validation de la condition aux limites implémentée [Section 6.6]. Une géométrie identique au *race track* est utilisé à un rapport d'homothétie près. En effet, la géométrie initiale, qui sera utilisée ici a été réalisée avec des demi-disques de rayon de 200 mm. Pour réaliser les calculs EF avec le même ordre de grandeur pour les pressions appliquées, nous faisons correspondre les périmètres de la géométrie du *race track* et celle de la jambe par l'application d'une homothétie de rapport 0,195 sur la géométrie du *race track*. La géométrie du *race track* est choisie car le contour est défini par deux droites et deux demi-cercles ce qui simplifie grandement la génération des contours et celle de l'image.
Pour ces tests, l'interface entre les deux matériaux est bloquée en déplacement. On rappelle que les deux matériaux utilisés pour la simulation EF sont de type *néo-hookéen* [Équation (6.7)] donc les paramètres matériaux de l'ensemble du modèle sont au nombre de 4. Ils sont présentés dans le [Tableau 8.1]. Nous allons à présent observer la variation

Matériau 1		Matériau 2	
c_1 (kPa)	K_1 (kPa)	c_2 (kPa)	K_2 (kPa)
9	43	13	44

Tableau 8.1 – Paramètres utilisés pour la génération du cas-test

de la fonction coût en fonction de la variation de ces paramètres *i.e.* nous allons réaliser une étude de sensibilité aux paramètres.

8.3.3 La sensibilité aux paramètres

Afin de réaliser cette étude, nous allons faire varier un à un les quatre paramètres à identifier en fixant les trois autres à leurs valeurs initiales. Cette sensibilité dépend du point choisi. Il est donc attendu d'observer un minimum de la fonction coût à la valeur initiale du paramètre variant, utilisée pour générer l'image cible numérique.

Variations de c_1 et c_2

Pour faire l'étude des sensibilités des paramètres c_1 et c_2, on fait varier ces paramètres dans l'intervalle de $\pm$ 50 % autour de leur valeur initiale [Tableau 8.1]. Pour

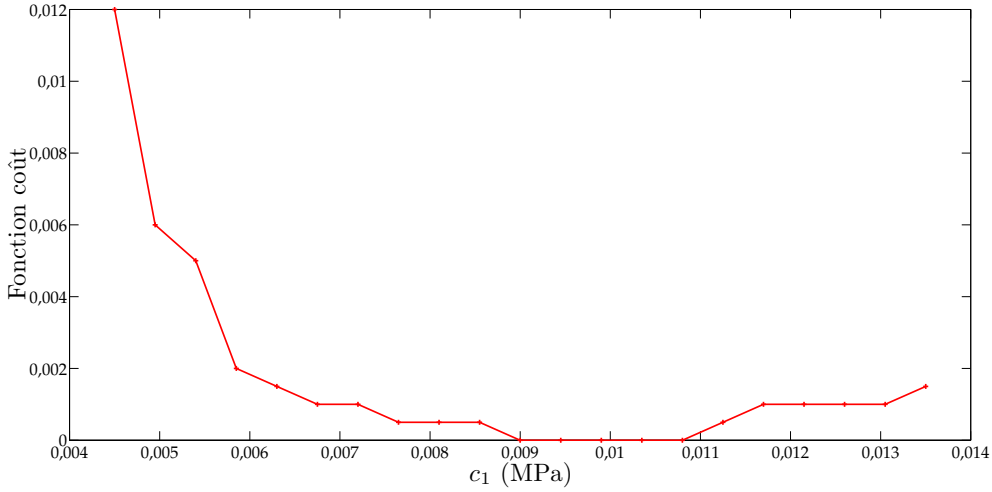


Figure 8.4 – Variation de la fonction coût en fonction du paramètre c_1

la sensibilité au paramètre c_1 [Figure 8.4], la valeur initiale n'est pas la seule valeur minimisant la fonction coût. En effet, la fonction coût est minimisée pour des valeurs comprises dans l'intervalle [9 kPa ; 10,8 kPa] soit de 0 à 20 % de la valeur attendue. La valeur maximale atteinte dans l'intervalle des valeurs testées est de 0,012.

En ce qui concerne la sensibilité au paramètre c_2 [Figure 8.5], celle-ci est beaucoup plus faible. La valeur maximale atteinte dans l'intervalle des valeurs testées est de 0,0055. Quelle que soit la valeur de c_2 supérieure à 8,5 kPa, elle minimise la fonction coût. Pour définir la solution, nous n'utilisons plus la notion d'intervalle mais le terme de borne inférieure.

Une possible explication à l'existence d'une borne inférieure et non plus d'intervalle est l'ordre de grandeur des déplacements. Les déformations sont si faibles qu'elles peuvent ne pas être détectées avec une précision au pixel près.

On fait varier simultanément les paramètres c_1 et c_2 dans ces mêmes intervalles. On représente les valeurs des fonctions coût sous la forme d'une surface [Figure 8.6]. On

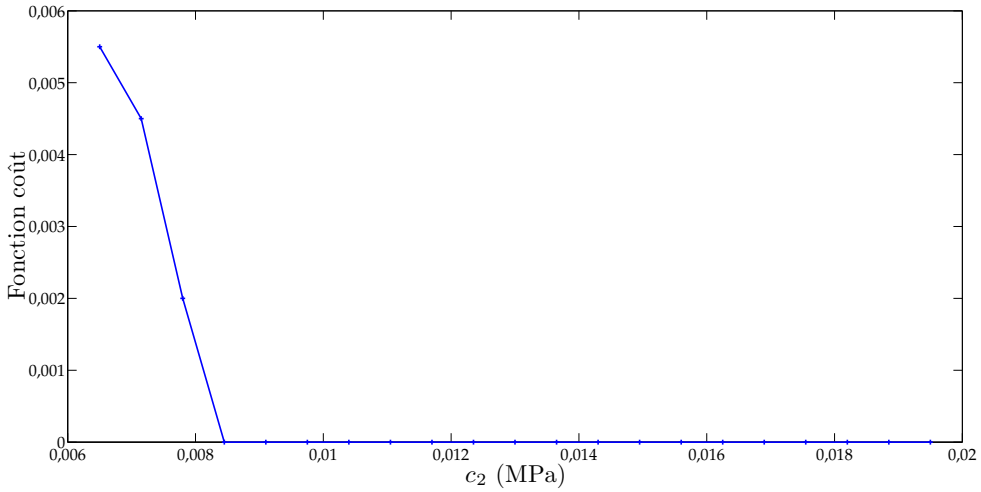


Figure 8.5 – Variation de la fonction coût en fonction du paramètre c_2

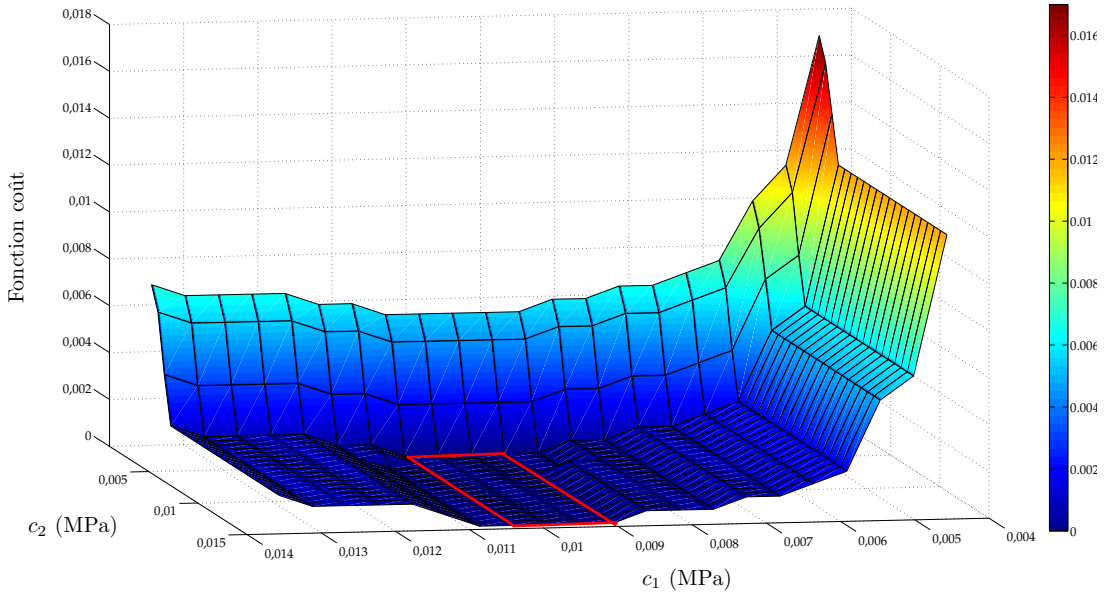


Figure 8.6 – Variation de la fonction coût en fonction des paramètres c_1 et c_2 simultanément

observe alors toute une zone où la fonction coût est minimale. La solution déterminée n'est donc pas unique, il existe plusieurs couples minimisant la fonction coût. Quel que soit c_1 situé dans l'intervalle [9 ; 10,8](kPa) et c_2 situé dans l'intervalle [8,5 ; 19,5](kPa), les couples formés dans ces intervalles minimisent la fonction coût.

Variations de K_1 et K_2

De la même manière que pour les paramètres précédents, on fait varier les paramètres K_1 et K_2 dans l'intervalle de $\pm$ 50 % autour de leur valeur initiale. Un intervalle de valeurs de K_1 comprises entre 38,7 kPa et 53,8 kPa soit une zone de 10 % en-dessus de la valeur initiale et de 25 % au-dessus de cette même valeur [Figure 8.7], minimise la fonction coût. Comme pour le paramètre c_2, le paramètre K_2 a une valeur minimale pour laquelle la fonction coût est minimisée. Cette valeur est de 39,6 kPa soit 10 % en-dessous de la valeur attendue.

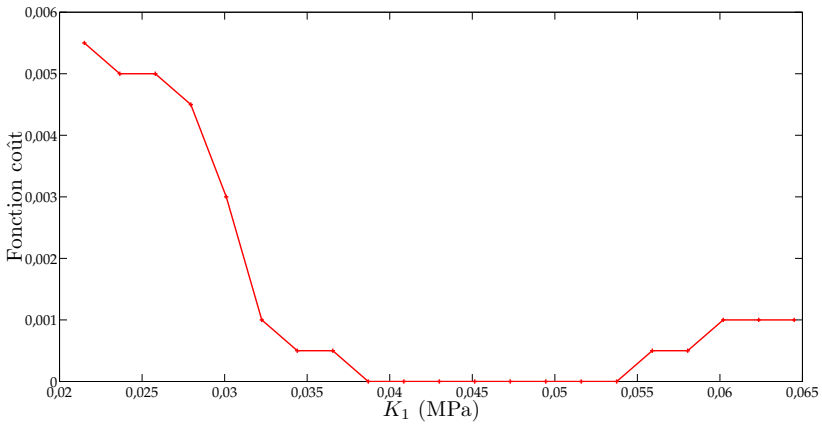


Figure 8.7 – Variation de la fonction coût en fonction du paramètre K_1

On réalise comme précédemment une surface des valeurs de la fonction coût pour différentes valeurs de K_1 et K_2 [Figure 8.9]. Comme pour la surface précédente [Figure 8.6], on observe une zone où la fonction coût est minimisée. Si K_1 est compris dans l'intervalle [38,7 ; 53,8] (kPa) et K_2 est compris dans l'intervalle [39,6 ; 66] (kPa) alors les couples [K_1, K_2] minimisent la fonction coût. La valeur de 66 kPa est la borne supérieure des valeurs testées, il est donc possible que des valeurs supérieures à 66 kPa pour le paramètre K_2 minimisent aussi la fonction coût compte tenu de l'allure de la courbe [Figure 8.8].

En conclusion, le paramètre le moins sensible est le paramètre K_2 car une variation de 50 % en-dessous de sa valeur de test n'entraîne qu'une variation de 0,0015 de la fonction coût. En comparaison, une variation de 50 % en dessous de la valeur de test de K_2 entraîne une variation de 0,0055 de la fonction coût. Le paramètre le plus sensible est le paramètre c_1. Une variation de 50 % en-dessous de sa valeur de test provoque

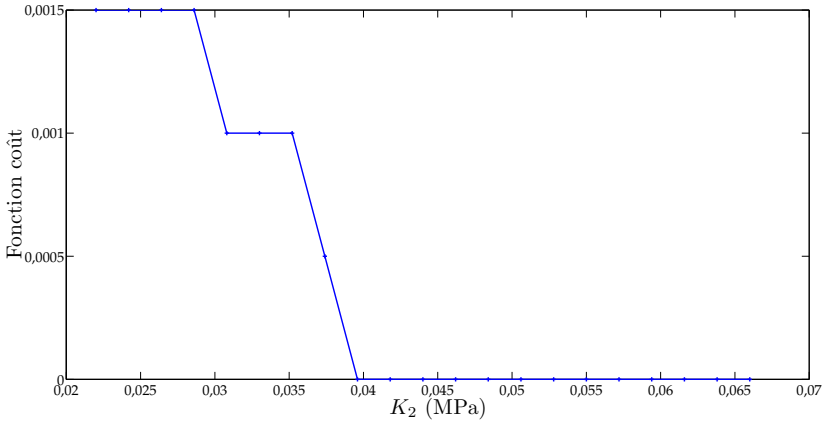


Figure 8.8 – Variation de la fonction coût en fonction du paramètre K_2

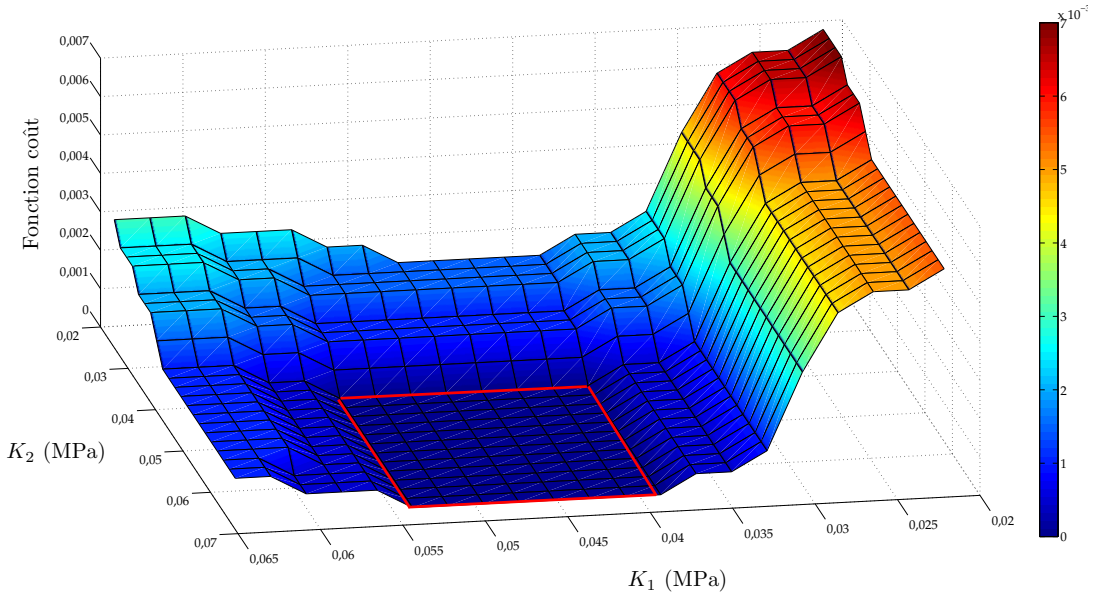


Figure 8.9 – Variation de la fonction coût en fonction des paramètres K_1 et K_2 simultanément

une variation de 0,012 de la fonction coût. Comparativement le paramètre c_2, lorsqu'il descend à 50 % de sa valeur de test, entraîne une variation de 0,006 de la fonction coût.

8.4 Identification des paramètres matériaux du cas-test

Nous avons présenté dans la première partie de ce chapitre deux algorithmes d'optimisation fréquemment utilisés pour la résolution de problèmes inverses : Newton-Raphson et Nelder-Mead. Pour ces deux algorithmes, nous avons testé plusieurs valeurs initiales. On s'attend à voir une nette différence dans les temps de calcul car l'algorithme de Newton-Raphson nécessite 9 calculs EF (calcul des gradients par différences finies centrées en dimension 4) pour chaque itération et pour l'identification de 4 paramètres alors que Nelder-Mead n'utilise qu'un seul calcul à chaque itération après l'étape d'initialisation du simplexe.

8.4.1 Identification par l'algorithme de Newton-Raphson

La méthode de Newton-Raphson est réputée pour converger de manière précise vers la solution à condition que les valeurs initiales soient suffisamment proches de la solution. Pour ces optimisations, un coefficient d'amortissement d de 0,1 est introduit. Nous avons souhaité étudier l'influence des valeurs initiales. On rappelle que les valeurs à identifier sont présentées dans le [Tableau 8.1]. Nous avons envisagé 6 jeux de valeurs initiales. On utilise les trois premiers critères d'arrêt présentés précédemment. Si un seul de ces critères est atteint, cela suffit à arrêter l'optimisation. Nous fixons le nombre maximal d'itérations (critère 1) à 200 et la fonction coût minimale (critère 2) 0,001. L'arrêt des optimisations réalisées est provoqué par la vérification du critère 3. Ce critère (C_{a3}) est défini par,

$$C_{a3} = \left| \frac{f_c^{n+1} - f_c^n}{f_c^n} \right| \tag{8.5}$$

La condition d'arrêt est C_{a3}<0,001. L'ensemble des résultats obtenus pour les différentes valeurs initiales sont récapitulés dans le [Tableau 8.2]. Quelles que soient les valeurs initiales utilisées, la fonction coût atteinte est inférieure à 0,019. La fonction coût minimale est obtenue pour les valeurs initiales de [5 ; 10 ; 40 ; 40]. Les valeurs alors identifiées sont [5 ; 10 ; 41,4 ; 40,3]. Seule la valeur initiale de c_1 n'est pas comprise dans l'intervalle minimisant la fonction coût [Figure 8.4]. Malgré des valeurs initiales proches des valeurs cibles [9 ; 13 ; 43 ; 44], l'algorithme de Newton-Raphson ne permet pas une minimisation satisfaisante.

8.4.2 Identification par l'algorithme de Nelder-Mead

Afin de comparer l'algorithme de Newton-Raphson et l'algorithme de Nelder-Mead, nous effectuons les optimisations avec les mêmes jeux de paramètres initiaux que pré-

cédemment. Les résultats sont présentés dans le [Tableau 8.2]. Le critère d'arrêt de l'algorithme de Nelder-Mead est un critère géométrique basé sur la distance entre les sommets du simplexe. Plus les sommets sont proches, plus on est près de la solution. La condition d'arrêt est que la distance entre les sommets est inférieure à 10^{-4}.

La fonction coût minimale obtenue avec l'algorithme de Nelder-Mead est 0, les paramètres alors identifiés sont [9,6 ; 10,1 ; 43,2 ; 69,3]. Ces valeurs sont comprises dans des intervalles définis dans les études de sensibilité comme étant des intervalles où la fonction coût est minimale.

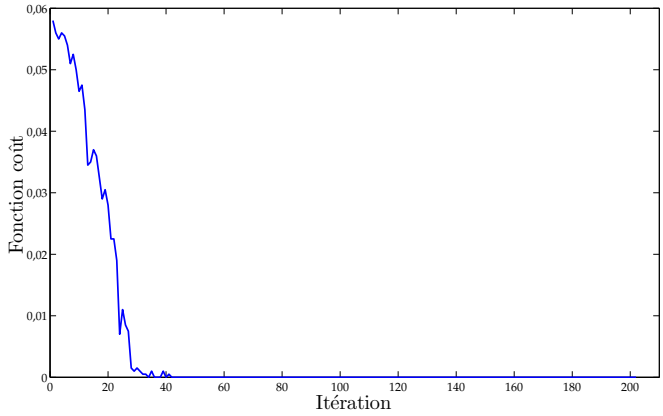


Figure 8.10 – Optimisation de la fonction coût par l'algorithme de Nelder-Mead
c_1=5kPa ; c_2=5kPa ; K_1=25kPa ; K_2=30kPa

Pour le même jeu de valeurs initiales, on compare la conververgence de la fonction coût en fonction du nombre d'itérations [Figure 8.11].

8.4.3 Discussion

L'étude des optimisations par les deux algorithmes présentés pour des valeurs initiales différentes, nous permet de pouvoir faire quelques remarques. On rappelle que la fonction coût représente la distance (en pixel) minimale moyenne entre chacun des pixels du contour cible et du contour numériquement obtenu. L'étude des sensibilités nous a permis de nous assurer que la fonction coût était minimisée avec les valeurs de test des paramètres.

Malgré la variabilité des jeux de paramètres, l'algorithme de Newton-Raphson permet d'obtenir une fonction coût dont la valeur est inférieure à 0,019. Cependant avec les valeurs initiales testées, il ne permet pas d'atteindre une valeur de fonction coût inférieure à 0,0065. Le temps moyen de calcul pour l'optimisation par l'algorithme de Newton-Raphson est de 7 minutes et 40 secondes.

L'algorithme de Nelder-Mead permet d'atteindre des paramètres annulant la fonc-

	Valeurs initiales (kPa)				Convergence		Fonction coût	Valeurs identifiées (kPa)			
	c_1	c_2	K_1	K_2	Nb d'itérations	Temps de calcul (s)	(10^{-3})	c_1	c_2	K_1	K_2
Newton-Raphson	9	13	20	20	6	161	8	8,9	13,2	20,4	22
	5	10	40	40	4	108	6,5	5	10	41,4	40,3
	5	5	25	30	29	783	13	7	6,8	25,8	28,1
	7	7	35	35	6	159	8	9,5	9	37,7	34
	20	20	43	44	6	158	9,5	14,4	10,7	39,2	19,7
	2	4	40	30	51	1390	19	4,5	6,2	47,3	34,5
Nelder-Mead	9	13	20	20	154	486	7	8,7	13,7	21,1	19,9
	5	10	40	40	177	573	1,5	6	9	44,2	39,9
	5	5	25	30	202	642	0	9,6	10,1	43,2	69,3
	7	7	35	35	162	593	8,5	7	7,3	35	35
	20	20	43	44	167	698	4	14,4	11	37,4	22
	2	4	40	30	221	510	7,8	4,4	4,2	33,6	20,5

Tableau 8.2 – Récapitulatif des optimisations réalisées sur le cas-test

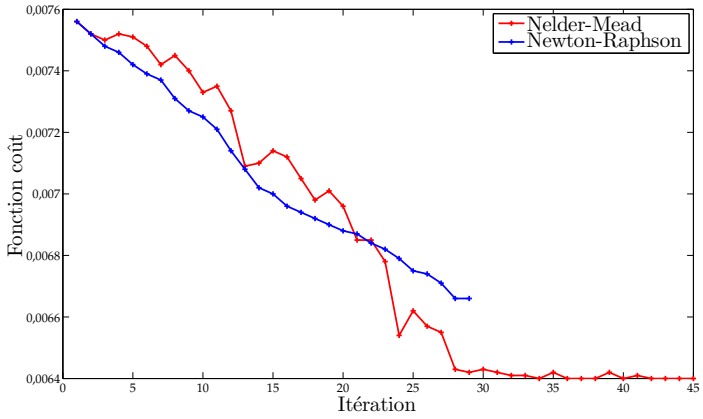


Figure 8.11 – Optimisation de la fonction coût par l'algorithme de Nelder-Mead
c_1=5kPa ; c_2=5kPa ; K_1=25kPa ; K_2=30kPa

tion coût, malgré des valeurs initiales assez éloignées. Pour des valeurs initiales de [5 ; 5 ; 25 ; 30] la fonction coût est annulée alors que pour le jeu [7 ; 7 ; 35 ; 35] qui sont des valeurs moins éloignées des valeurs de test, on obtient une valeur de la fonction coût de 0,0085. Cette valeur est la plus grande obtenue avec l'algorithme de Nelder-Mead. Le temps moyen d'une optimisation par l'algorithme de Nelder-Mead est de 9 minutes et 43 secondes.

Les différences attendues sur le temps de calcul ne sont pas très marquées. L'algorithme de Newton-Raphson nécessite 9 calculs EF par itération mais le nombre d'itération est inférieur à 51 alors que pour l'algorithme de Nelder-Mead, un seul calcul EF est nécessaire par itération mais le nombre d'itération est supérieur à 154. Cependant, l'algorithme de Nelder-Mead est plus robuste, on s'assure donc de s'approcher de la solution malgré un nombre important (>154) d'itérations. L'utilisation de l'algorithme de Newton-Raphson peut être plus rapide dans certains cas mais peut aussi s'éloigner de la solution. Sur le cas-test le temps de calcul moyen pour l'algorithme de Newton-Raphson est de 7 minutes et 40 secondes et il est de 9 minutes et 44 secondes pour l'algorithme de Nelder-Mead. Compte tenu de la faible différence entre les temps de calculs moyens et la robustesse des algorithmes, on préfèrera utiliser l'algorithme de Nelder-Mead.

Comme nous l'avions observé dans l'étude des sensibilités, les résultats sont plus sensibles aux paramètres c qu'aux paramètres K. Pour le premier jeu utilisé [9 ; 13 ; 20 ; 20], les valeurs initiales de c sont les valeurs de test. Les deux algorithmes présentent alors des difficultés à identifier les paramètres K. Cependant, lorsqu'on utilise comme valeurs initiales pour K, leurs valeurs de test, les valeurs identifiées pour ces mêmes paramètres s'éloignent de la solution.

8.5 Conclusion

Nous avons présenté dans ce chapitre deux algorithmes permettant de minimiser la fonction coût. Nous avons choisi de présenter une méthode d'ordre 0, soit l'algorithme de Nelder-Mead et une méthode d'ordre 1, soit l'algorithme de Newton-Raphson. Pour observer l'une et l'autre de ces méthodes, nous les avons appliquées à un cas-test par le principe du *blind test*. Les différents algorithmes ont été testés pour différents jeux de valeurs initiales. La première remarque faite est que les résultats sont dépendants des valeurs initiales prises pour les deux algorithmes testés. Quel que soit l'algorithme utilisé pour l'identification des paramètres matériaux des tissus mous de la jambe, il sera préférable d'utiliser des valeurs initiales les plus proches possibles des valeurs minimisant la fonction coût, nous serons donc amenés à déterminer l'ordre de grandeur des paramètres à identifier.

L'étude de la sensibilité aux paramètres permet de constater que le paramètre c_1 est le paramètre le plus sensible. Les paramètres c pouvant être assimilés à des modules d'élasticité, la relation $c_1 < c_2$ indique que les déformations dans le matériau 1 sont plus importantes que dans le matériau 2. La différence d'amplitude des déformations permet en partie d'expliquer les sensibilités différentes aux paramètres c_1 et c_2. Plus les déformations sont faibles, plus la fonction coût sera minimisée par plusieurs valeurs. Ceci est une conséquence d'une fonction coût définie au pixel près. Néanmoins, celle-ci nous permet de définir des intervalles (pour le matériau 1) et des bornes inférieures (pour le matériau 2), dans lesquels se situent les valeurs cibles. La méthode d'identification utilisée permet alors de borner les valeurs minimisant la fonction coût.