Analyse des tolérances de systèmes mécaniques hyperstatiques - techniques de résolution

Sommaire

3.1 Introduction 76

3.2 Analyse des tolérances de systèmes mécaniques hyperstatiques - méthode classique 76

3.2.1 Condition d'assemblage – formalisation de son problème d'optimisation et evaluation de la probabilité de non-assemblage 77

3.2.2 Condition de fonctionnalité – formalisation de son problème d'optimisation et évaluation de la probabilité de non-fonctionnalité 80

3.2.3 Synthèse 83

3.3 Choix de l'algorithme d'optimisation 84

3.3.1 Démarche 85

3.3.2 Résultats – Comparaisons 88

3.3.3 Synthèse 93

3.4 Nouvelle approche d'analyse des tolérances basée sur le modèle probabiliste des jeux 94

3.4.1 Construction de la densité par l'estimation par noyaux 95

3.4.2 Génération de nouvelles variables basés sur l'estimation par l'estimation par noyaux 97

3.4.3 Intégration des nouvelles variables dans la démarche de l'analyse des tolérances 100

3.4.4 Justification de la proposition d'un modèle probabiliste pour la caractérisation du jeu 102

3.5 Conclusion 104

3.1 Introduction

L'objectif de l'analyse des tolérances par l'approche statistique est de déterminer les indicateurs : les probabilités d'assemblage et de fonctionnalité des systèmes mécaniques hyperstatiques. Il existe plusieurs techniques qui permettent de réaliser l'analyse des tolérances de systèmes mécaniques comme évoqué dans le chapitre 1. La simulation de Monte Carlo est la méthode de référence qui a été utilisée pour l'analyse des tolérances des systèmes mécaniques hyperstatiques. La simulation de Monte Carlo couplée à une technique d'optimisation proposée dans les travaux de Qureshi (2011), est adaptée dans le contexte de cette thèse pour répondre à la problématique de l'analyse des tolérances. Cette méthode est aussi adoptée comme méthode de référence dans ce manuscrit. Les objectifs de ce chapitre sont de montrer l'adaptation de cette méthode de référence pour tenir compte des développements du chapitre précédent à savoir l'intégration des défauts de forme et la distinction entre les différents types de contact. Ensuite, une nouvelle méthode est proposée afin de réduire le temps de calcul de la simulation de Monte Carlo : la section 3.3 détaille la méthode proposée basée sur la méthode de l'estimation par noyaux (en Anglais *Kernel Density Estimation* (KDE)) afin de déterminer des modèles probabilistes des composantes des jeux des contacts fixes et glissants qui sont intégrés par la suite dans l'optimisation et la simulation de Monte Carlo.

3.2 Analyse des tolérances de systèmes mécaniques hyperstatiques - méthode classique

La méthode de Monte Carlo (MC) est une méthode classique d'analyse statistique des tolérances. Pour un système mécanique hyperstatique, la fonction réponse dans l'analyse des tolérances n'est pas toujours explicite comme pour les systèmes mécaniques isostatiques (Dumas et al., 2015a). Dans ce contexte, la MC est très appropriée dans le cas d'une fonction réponse implicite des systèmes mécaniques hyperstatiques.

L'objectif de l'analyse des tolérances statistique utilisée dans ce manuscrit est de calculer les probabilités de défaillance d'assemblage et de fonctionnalité : la probabilité de défaillance d'assemblage ($\tilde{P}_{na}$) et la probabilité de défaillance de fonctionnalité ($\tilde{P}_{nf}$). La méthode développée dans ce chapitre prend en compte (i) l'introduction d'un ensemble de défauts de forme dans les différentes contraintes modélisant le comportement géométrique d'un système mécanique hyperstatique, (ii) une optimisation pour l'étude de chaque contact fixe et glissant. Cette optimisation basée sur la notion de distances signées, permet de déterminer les

configurations possibles des jeux d'assemblage des contacts fixes et glissants du système mécanique. Une optimisation est ainsi menée sur chaque contact fixe et glissant à chaque itération de la simulation de MC.

Afin de calculer les deux probabilités de défaillance, un grand nombre N_{mc} d'échantillons des variations géométriques est généré. A chaque itération de Monte Carlo, un échantillon des ensembles **X** et **F** est tiré aléatoirement suivant des lois de probabilité définies et les conditions d'assemblage et de fonctionnalité sont vérifiées.

3.2.1 Condition d'assemblage – formalisation de son problème d'optimisation et evaluation de la probabilité de non-assemblage

La condition d'assemblage est formalisée mathématiquement en se basant sur la notion du quantificateur « $\exists$ » tirée des travaux de Dantan *et al.* (2005), Qureshi *et al.* (2012). Toutefois, cette formalisation est modifiée dans les travaux de thèse de ce manuscrit afin de prendre en compte les défauts de forme. La condition d'assemblage est alors traduite par : « *il existe une configuration admissible des jeux du système mécanique telle que les équations de compatibilité, les contraintes d'interface soient respectées* ». Sa traduction mathématique est donnée par :

$$\exists \mathbf{G} \in \left\{ \mathbf{G} \in \mathbf{R}^m : C_c\left(\mathbf{X},\mathbf{G}\right) = 0 \cap C_i\left(\mathbf{X},\mathbf{F},\mathbf{G}^{'}\right) \le 0 \cap C_i\left(\mathbf{X},\mathbf{F},\mathbf{G}^{*}\right) = 0 \right\} \tag{3.1}$$

Le but de la méthode d'optimisation pour la vérification de la condition d'assemblage est de minimiser les composantes des torseurs jeux des contacts flottants et les composantes des déplacements cinématiques des torseurs glissants sous la contrainte du respect des équations de compatibilité et des contraintes d'interface. Cette optimisation permet de trouver, lorsqu'elle converge les meilleures composantes des torseurs jeux qui permettent l'assemblage du système mécanique. La fonction objectif de l'optimisation est alors une fonction des composantes du vecteur $\tilde{\mathbf{G}}^{'}$. Pour cette optimisation la fonction objectif et les contraintes sont linéaires. Le problème d'optimisation peut alors être formulé mathématiquement par :

$$R_a\left(\mathbf{X},\mathbf{F}\right) = \begin{cases} \underset{\mathbf{G}^{'}}{\text{Min}}\ \mathrm{f}\left(\tilde{\mathbf{G}}^{'}\right) \\ \text{sous} \quad C_c\left(\mathbf{X}^{(k)},\tilde{\mathbf{G}}^{'},\mathbf{G}^{*(k)}\right) = 0 \qquad \forall\ k = \left\{1,...,N_{mc}\right\} \\ \qquad\quad C_i\left(\mathbf{X}^{(k)},\tilde{\mathbf{G}}^{'},\mathbf{F}^{(k)}\right) \le 0 \end{cases} \tag{3.2}$$

où $C_c\left(\mathbf{X}^{(k)},\tilde{\mathbf{G}}',\mathbf{G}^{*(k)}\right)$ représente l'ensemble des équations de compatibilité des k^{ième} échantillons de $\mathbf{X}$ et $\mathbf{G}^*$. $C_i\left(\mathbf{X}^{(k)},\tilde{\mathbf{G}}',\mathbf{F}^{(k)}\right)$ représente l'ensemble des contraintes d'interface des contacts flottants exprimées en fonction des k^{ième} échantillons de $\mathbf{X}$ et $\mathbf{G}^*$. $\tilde{\mathbf{G}}'$ représente le vecteur rassemblant toutes les composantes des torseurs jeux des contacts flottants et les composantes des déplacements cinématiques des contacts glissants ; les composantes de $\tilde{\mathbf{G}}'$ sont les variables qui doivent être déterminées par cette optimisation ; les composantes de $\tilde{\mathbf{G}}'$ définissent la pire configuration des jeux assurant l'assemblage du système mécanique. $\mathbf{G}^{*(k)}$ représente le vecteur regroupant les composantes des torseurs jeux des contacts fixes et glissants obtenues par d'autres optimisations (les simulations locales, chapitre 2, sous-section 2.3.1) en fonction des k^{ième} échantillons de $\mathbf{X}$ et $\mathbf{F}$, et $\mathbf{G}$ est le vecteur regroupant toutes les composantes des torseurs jeux i.e. $\mathbf{G}=\left\{\mathbf{G}^*,\mathbf{G}'\right\}$. N_{mc} représente le nombre total d'échantillons générés par la simulation de Monte Carlo.

Au cours du processus de résolution du problème d'assemblage, l'Eq. (3.2) est mise sous forme matricielle comme le montre la formalisation suivante :

$$R_a\left(\mathbf{X},\mathbf{F}\right)=\begin{cases}\underset{\tilde{\mathbf{G}}'}{\text{Min}} \quad \text{f}\left(\tilde{\mathbf{G}}'\right) \\ \textit{sous} \quad \left[C_{omp}\right]\left[\tilde{\mathbf{G}}'\right]=\left[\mathbf{X}^{(k)}+\mathbf{G}^{*(k)}\right] \\ \qquad \left[C_{\text{Interface}}\right]\left[\tilde{\mathbf{G}}'\right]\le\left[\mathbf{X}^{(k)}_{\text{Intrinsèque}}+\mathbf{F}^{(k)}\right]\end{cases} \qquad \forall\ k=\left\{1,...,N_{mc}\right\} \qquad (3.3)$$

où $\left[C_{omp}\right]$ représente la matrice contenant les coefficients des composantes des torseurs jeux des contacts flottants et des composantes des déplacements cinématiques des contacts glissants ; $\left[C_{\text{Interface}}\right]$ représente la matrice contenant les coefficients des composantes des contacts flottants ; $\mathbf{X}^{(k)}_{\text{Intrinsèque}}$ est un vecteur contenant les composantes des écarts intrinsèques du k^{ième} échantillon du vecteur $\mathbf{X}$.

Exemple : *Formalisation d'un problème d'assemblage entre deux pièces*

La figure montre l'assemblage entre deux pièces composé d'un contact fixe (1b/2b) et d'un contact flottant (1a/2a).

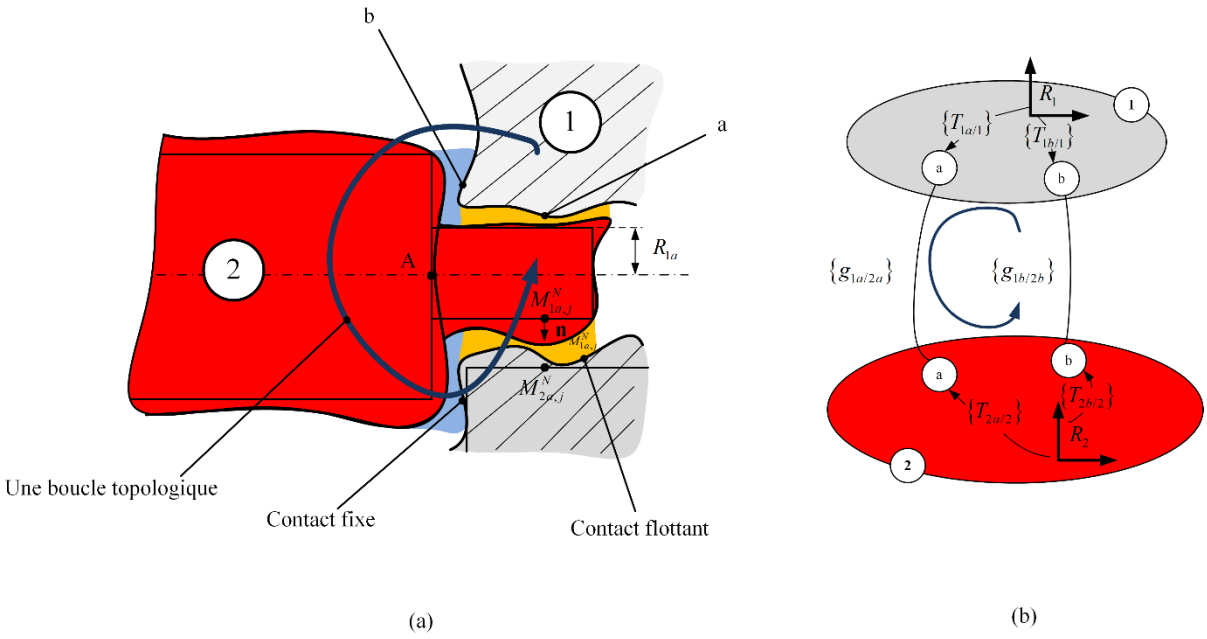

Figure 3.1 : (a) Assemblage entre deux pièces ; (b) Graphe des liaisons de l'assemblage

La formalisation mathématique sous forme matricielle du problème d'assemblage de ce contact est obtenue en combinant les Eqs. (2.9) et (2.21) *du chapitre 2 :*

$$
\begin{cases}
\text{Min} \quad \mathrm{f}\left(\alpha_{1a2a},\beta_{1a2a},\gamma_{1a2a},u_{1a2a,A},v_{1a2a,A},w_{1a2a,A}\right) \\
\textit{sous} \\
\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_{1a2a} \\ \beta_{1a2a} \\ \gamma_{1a2a} \\ u_{1a2a,A} \\ v_{1a2a,A} \\ w_{1a2a,A} \end{bmatrix} = \begin{bmatrix} \alpha_{1a1}-\alpha_{2a2}+\alpha_{2b2}+\alpha^*_{1b2b}-\alpha_{1b1} \\ \beta_{1a1}-\beta_{2a2}+\beta_{2b2}+\beta^*_{1b2b}-\beta_{1b1} \\ \gamma^*_{1b2b} \\ u_{1a1,A}-u_{2a2,A}+u^*_{1b2b} \\ v_{1a1,A}-v_{2a2,A}+v^*_{1b2b} \\ w^*_{1b2b}+w_{2b2,A}+w_{1b1,A} \end{bmatrix} \\
\begin{bmatrix} -l_{1a,j}.\sin\theta & l_{1a,j}.\cos\theta & 0 & \cos\theta & \sin\theta & 0 \end{bmatrix} \begin{bmatrix} \alpha_{1a2a} \\ \beta_{1a2a} \\ \gamma_{1a2a} \\ u_{1a2a,A} \\ v_{1a2a,A} \\ w_{1a2a,A} \end{bmatrix} \le \left[\left(\frac{d_{1a}-d_{2a}}{2} \right) + \left\| f_{1a,M^N_{1a,j}} \right\| + \left\| f_{2a,M^N_{b,j}} \right\| \right]
\end{cases}
\tag{3.4}
$$

Les composantes du torseur jeu $\{g_{1b/2b}\}_A$ *sont considérées comme connues car elles sont obtenues par la simulation locale décrite dans la partie 2.3.2.2 du chapitre 2.*

Apres avoir défini le problème d'assemblage et la technique d'optimisation permettant de déterminer le vecteur $\tilde{\mathbf{G}}'$, la simulation de Monte Carlo est lancée pour un nombre N_{mc} d'échantillons des vecteurs **X** et **F**. Les composantes du vecteur $\mathbf{G}^*$ sont déterminées par d'autres optimisations sur les torseurs jeux des contacts fixes et glissants, à chaque itération de la simulation de Monte Carlo. La combinaison de la simulation de Monte Carlo et des techniques d'optimisations permet de calculer la probabilité de non-assemblage qui représente une indication sur la qualité des produits fabriqués. Cette probabilité de non-assemblage est obtenue en utilisant la relation :

$$\tilde{\mathrm{P}}_{\mathrm{na}} = \frac{1}{N_{mc}} \sum_{i=1}^{N_{mc}} I_{Dfa}\left(\mathbf{X}^{(i)}, \mathbf{F}^{(i)}\right) \tag{3.5}$$

où I_{Dfa} représente la fonction indicatrice définie par :

$$I_{Dfa}\left(\mathbf{X},\mathbf{F}\right) = \begin{cases} 1 \text{ si } R_a\left(\mathbf{X},\mathbf{F}\right) \text{ n'a pas de solution} \\ 0 \text{ si } R_a\left(\mathbf{X},\mathbf{F}\right) \text{ a une solution} \end{cases} \tag{3.6}$$

3.2.2 Condition de fonctionnalité – formalisation de son problème d'optimisation et évaluation de la probabilité de non-fonctionnalité

Les conditions de fonctionnalité sont aussi développées en se basant sur la notion du quantificateur « ∀ » tirée des travaux de Dantan *et al.* (2005), Qureshi *et al.* (2012). La définition de la condition de fonctionnalité a déjà été donnée dans le chapitre 1. Toutefois, une modification est faite à sa formalisation mathématique afin de prendre en compte les défauts de forme :

$$\forall \mathbf{G} \in \left\{ \mathbf{G} \in \mathbf{R}^m : C_c\left(\mathbf{X},\mathbf{G}\right) = 0 \cap C_i\left(\mathbf{X},\mathbf{F},\mathbf{G}'\right) \le 0 \cap C_i\left(\mathbf{X},\mathbf{F},\mathbf{G}^*\right) = 0 \right\}, C_f\left(\mathbf{X},\mathbf{F},\mathbf{G}\right) \ge 0 \tag{3.7}$$

La formalisation mathématique du problème fonctionnel est transformée en un problème d'optimisation. Cette optimisation permet de déterminer les jeux fonctionnels de la pire des configurations des jeux. La fonction objectif de l'optimisation est alors une fonction des composantes du vecteur contenant les composantes du torseur jeu fonctionnel ($\tilde{\mathbf{G}}'_{\text{fonctionnel}}$). Cette fonction objectif est minimisée telle que les équations de compatibilité, les contraintes d'interface et les conditions fonctionnelles soient vérifiées. Le problème d'optimisation du problème fonctionnel est alors formalisé mathématiquement par :

$$R_f\left(\mathbf{X},\mathbf{F}\right)=\begin{cases}\underset{\mathbf{G}'_{\text{fonctionnel}}}{\text{Min}} & C_f\left(\mathbf{X}^{(k)}_{\text{Intrinsèque}},\mathbf{F}^{(k)},\tilde{\mathbf{G}}'_{\text{fonctionnel}}\right)\\ \text{sous} & C_c\left(\mathbf{X}^{(k)},\tilde{\mathbf{G}}',\mathbf{G}^{*(k)}\right)=0 \qquad \forall\ k=\{1,...,N_{mc}\}\\ & C_i\left(\mathbf{X}^{(k)},\tilde{\mathbf{G}}',\mathbf{F}^{(k)}\right)\le 0\end{cases} \tag{3.8}$$

où $\tilde{\mathbf{G}}'_{\text{fonctionnel}}$ représente le vecteur contenant les composantes du torseur jeu fonctionnel qui sont évaluées par cette optimisation.

Dans la procédure de l'analyse des tolérances, l'Eq. (3.8) est mise sous forme matricielle :

$$R_f\left(\mathbf{X},\mathbf{F}\right)=\begin{cases}\underset{\tilde{\mathbf{G}}'_{\text{fonctionnel}}}{\text{Min}} & C_f\left(\mathbf{X}^{(k)}_{\text{Intrinsèque}},\mathbf{F}^{(k)},\tilde{\mathbf{G}}'_{\text{fonctionnel}}\right)\\ \textit{sous} & \left[C_{omp}\right]\left[\tilde{\mathbf{G}}'\right]=\left[\mathbf{X}^{(i)}+\mathbf{G}^{*(k)}\right] \qquad \forall\ k=\{1,...,N_{mc}\}\\ & \left[C_{\text{Interface}}\right]\left[\tilde{\mathbf{G}}'\right]\le\left[\mathbf{X}^{(k)}_{\text{Intrinsèque}}+\mathbf{F}^{(k)}\right]\end{cases} \tag{3.9}$$

Toutes ces formalisations mathématiques permettent d'évaluer lors de l'analyse des tolérances la probabilité de non-fonctionnalité d'un système mécanique, en utilisant les relations suivantes :

$$\tilde{\mathrm{P}}_{\mathrm{nf}}=\frac{1}{N_{mc}}\sum_{i=1}^{N_{mc}} I_{Df}\left(\mathbf{X}^{(i)},\mathbf{F}^{(i)}\right) \tag{3.10}$$

où I_{Df} représente la fonction indicatrice définie par :

$$I_{Df}\left(\mathbf{X}\right)=\begin{cases}1 \text{ si } R_f\left(\mathbf{X},\mathbf{F}\right)\le 0\\ 0 \text{ si } R_f\left(\mathbf{X},\mathbf{F}\right)>0\end{cases} \tag{3.11}$$

Le processus de l'analyse des tolérances d'un système mécanique hyperstatique en considérant les différents types de contact est constitué de plusieurs étapes. Les premiers problèmes rencontrés sont les problèmes d'assemblage et fonctionnel au regard de la vérification de certaines tolérances attribuées aux différentes entités des pièces mécaniques. Ces problèmes d'assemblage et fonctionnel sont résolus à l'aide des formalisations mathématiques utilisant les notions de quantificateurs « $\exists$ » et « $\forall$ ». Ces formalisations mathématiques des problèmes d'assemblage et fonctionnel sont ensuite transformées en deux problèmes d'optimisation. La première optimisation qui concerne le problème d'assemblage, a pour objectif de déterminer les composantes du vecteur $\tilde{\mathbf{G}}'$ telles que les équations de compatibilité et les contraintes d'interface soient respectées. La deuxième optimisation concernant le problème fonctionnel, a pour objectif de déterminer les composantes du vecteur $\tilde{\mathbf{G}}'_{\text{fonctionnel}}$ et $\tilde{\mathbf{G}}'$ telles que les équations de compatibilité, les contraintes d'interface et les conditions fonctionnelles soient respectées. Toutefois, avant la formalisation des différentes contraintes, les composantes du vecteur $\mathbf{G}^*$ doivent être déterminées par des optimisations sur les torseurs jeux des contacts fixes et glissants. Les composantes de $\mathbf{G}^*$ sont considérées comme connues avant de lancer les optimisations sur les problèmes d'assemblage et fonctionnel. Toutes ces techniques d'optimisations sont couplées à une simulation de Monte Carlo qui génère aléatoirement N_{mc} échantillons des vecteurs $\mathbf{X}$ et $\mathbf{F}$. La détermination de $\mathbf{G}^*$ par optimisation à chaque itération de MC, constitue la différence par rapport à la méthode précédente proposée par Qureshi *et al.* (2012). L'analyse des tolérances permettent de déterminer la probabilité de non-assemblage et la probabilité de non-fonctionnalité à l'aide des Eqs. (3.5) et (3.10). La Figure 3.2 illustre les étapes de la méthode proposée.

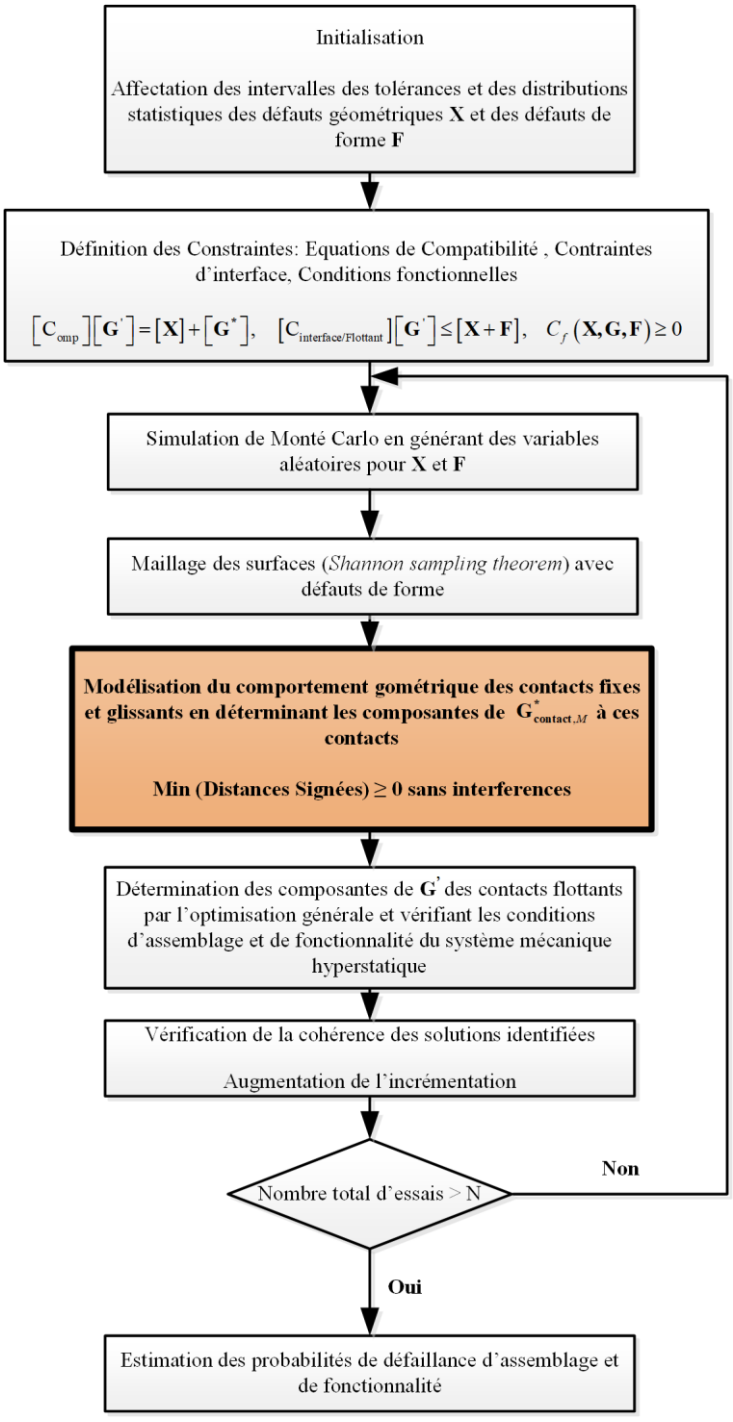

Figure 3.2 : Procédure de l'analyse des tolérances

3.2.3 Synthèse

Pour synthétiser, l'objectif de cette section est de proposer une nouvelle méthode d'analyse des tolérances qui prend en compte les défauts de forme et la modélisation du comportement géométrique des différents types de contacts. Ainsi, la méthode précédente d'analyse des

tolérances basée sur la simulation MC et une technique d'optimisation a été modifiée afin de prendre en compte les défauts de forme. La nouvelle méthode d'analyse des tolérances des systèmes mécaniques hyperstatiques, est composée de trois différents types de problèmes d'optimisation. Ces problèmes d'optimisation nécessitent un algorithme de résolution efficace et juste. De ce fait, des simulations et comparaisons sont menées sur quelques algorithmes de résolution de problèmes d'optimisation afin de choisir le meilleur. Ces comparaisons sont présentées dans la section suivante et sont réalisées sur un système mécanique qui est présenté en détail dans le chapitre 4.

3.3 Choix de l'algorithme d'optimisation

Cette section propose de déterminer le meilleur algorithme permettant de réaliser à la fois les optimisations sur les composantes des torseurs jeux des contacts fixes et glissants et aussi l'optimisation accompagnant la simulation de MC. En effet, plusieurs algorithmes existent dans la littérature pour résoudre les problèmes d'optimisation. Un choix est donc nécessaire pour déterminer le plus adéquat avec les problèmes d'optimisation rencontrés dans les techniques d'analyse des tolérances. Pour atteindre cet objectif, des simulations sont réalisées en considérant le cas du moteur électrique comme cas d'étude. Ce moteur électrique est décrit plus en détail dans le chapitre suivant. Ainsi les simulations réalisées à la fois sur la simulation locale et sur la méthode générale de l'analyse des tolérances, ont pour but de déterminer l'algorithme possédant les caractéristiques suivantes :

- La justesse : les composantes du torseur jeu déterminées sont analysées pour vérifier la similarité des résultats suivant les algorithmes et les types de maillages. De plus, les contraintes du problème d'optimisation sont analysées pour vérifier des violations ou non. Enfin, les probabilités estimées sont analysées pour vérifier la similitude des résultats pour les mêmes échantillons suivant les algorithmes.
- L'efficacité : les temps de calcul des algorithmes sont analysés afin de déterminer celui qui permet la résolution des problèmes d'optimisation avec un temps faible.

La démarche suivie et les algorithmes étudiés sont présentés dans la sous-section 3.3.1. Les résultats et les comparaisons entre les algorithmes pour déterminer le meilleur, sont proposés dans la sous-section 3.3.2.

3.3.1 Démarche

Les simulations employées pour choisir le meilleur algorithme ont été réalisées à deux niveaux.

Le premier niveau concerne les simulations locales sur les composantes du torseur jeu du contact *1a/2a* (voir Figure 4.2, section 4.2). Cinq échantillons d'amplitude des défauts de forme ont été tirés et appliqués aux surfaces du contact. Ces tirages d'amplitude sont les mêmes pour tous les algorithmes choisis. Les optimisations sont aussi réalisées en considérant différents types de maillage : 100, 625, 2500 et 10000 points de discrétisation. Le but de ces simulations est d'analyser les variations des composantes du vecteur $\mathbf{G}^{*}_{1a/2a,M}$ déterminées pour chaque tirage, en fonction des algorithmes et du type de maillage. De plus, le taux de violation des contraintes pour chaque algorithme est calculé. Le vecteur $\mathbf{G}^{*}_{1a/2a,M}$ est un vecteur constitué des composantes des déplacements non-cinématiques du torseur jeu $\left\{g^{*}_{1a/2a}\right\}_{M}$ du contact glissant entre les surfaces *1a* et *2a* et exprimé au point *M* situé au centre de la liaison. Le vecteur $\mathbf{G}^{*}_{1a/2a,M}$ peut s'écrire sous la forme :

$$\mathbf{G}^{*}_{1a/2a,M} = \begin{Bmatrix} w_{1a2a,M} \\ \alpha_{1a2a} \\ \beta_{1a2a} \end{Bmatrix} \tag{3.12}$$

La fonction objectif de chaque algorithme est en fonction des composantes $w_{1a2a,M}$, α_{1a2a}, β_{1a2a}. Un exemple des composantes du vecteur $\mathbf{G}^{*}_{1a/2a,M}$ dérivé d'un torseur jeu $\left\{g^{*}_{1a/2a}\right\}_{M}$ d'un contact plan-plan est montré sur la Figure 3.3.

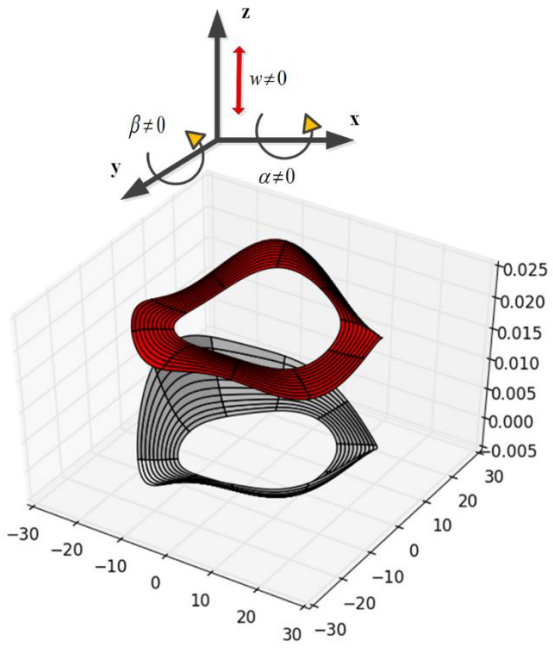

Figure 3.3 : Composantes du vecteur $\mathbf{G}^*_{kj/ij}$ *d'un contact glissant plan-plan à contour circulaire*

Les contraintes de ce problème d'optimisation sont les distances signées entre les deux surfaces du contact. Le problème d'optimisation à résoudre à ce niveau est exactement le même que celui qui est décrit dans l'équation (2.14) du Chapitre 2 sur les contacts fixes et glissants. Dans le cas actuel, il s'agit d'un contact glissant plan-plan.

Le deuxième niveau des simulations concerne l'analyse des tolérances menée sur le système mécanique. Il s'agit de la technique de l'analyse des tolérances décrite dans la section 3.2. Pour réaliser ces simulations, dix tirages de MC sont générés. Ces tirages sont les mêmes pour tous les algorithmes. De même, les amplitudes des défauts de forme des contraintes d'interface des contacts cylindre-cylindre (contacts *2c/3c*, *1e/3e*, *1b2b*) du système mécanique sont les mêmes pour tous les algorithmes. Cinq réalisations des composantes du vecteur $\mathbf{G}^*_{1a/2a,M}$ ont été générées pour chaque algorithme choisi et pour deux types de maillage des surfaces *1a* et *2a* (100 et 10000 points de discrétisation).

Les différents algorithmes ou solveurs choisis pour réaliser les simulations et qui sont disponibles dans le langage de programmation Python sont décrits ci-après. Tous ces algorithmes de comparaison traitent de problèmes d'optimisation linéaire sous contraintes. Les fonctions objectifs et les contraintes des différents problèmes d'optimisation sont linéaires.

Algorithme du simplexe est un algorithme très performant de résolution des problèmes d'optimisation linéaire. Il a été développé en 1947 par George Dantzig (Meyer, 2011). Les problèmes d'optimisation sont résolus en se déplaçant sur les frontières d'un sommet (point extrême) vers un autre prochain sommet. L'algorithme du simplexe trouve la solution optimale en utilisant une séquence d'étapes de pivot sur le système d'équations original. De plus, l'utilisation de cet algorithme suppose la connaissance d'une solution initiale au problème d'optimisation linéaire. Plus amples informations sur cet algorithme standard du simplexe peuvent être trouvées dans (Dantzig et Thapa, 1997; Nocedal et Wright, 1999) ; ils sont résumés brièvement dans l'Annexe C. Cet algorithme du simplexe est implémenté dans le langage Python sous différentes fonctions : Linprog qui est basé sur la méthode standard du simplexe et Lpsolve qui est basé sur la méthode modifiée du simplexe. La description des deux implémentations est exposée comme suit.

- **Linprog** est une fonction qui a pour objectif de résoudre les problèmes d'optimisation linéaire. La fonction objectif et les contraints du problème d'optimisation doivent être linéaires. C'est une fonction qui est basée sur la méthode standard du simplexe.
- **Lpsolve** est un solveur de programmation linéaire basé sur la méthode modifiée du simplexe (*revised simplex method* en Anglais) et la méthode de séparation et évaluation (*Branch-and-bound method* en Anglais). La méthode modifiée du simplexe a le même principe comme la méthode standard du simplexe mais le « tableau » (*dictionnary* en Anglais) n'est plus utilisé. Il a été développé par Michel Berkelaar. Lpsolve peut résoudre des problèmes purement linéaires, ou des problèmes avec des variables mixtes (entier/binaire), des variables semi-continues ou des problèmes de modèles d'ensembles spéciaux ordonnés. La fonction objectif et les contraintes doivent être linéaires dans la formalisation du problème dans Lpsolve. Plus amples informations sur la méthode modifiée du simplexe peuvent être trouvées dans les travaux de Meyer (2011).

L'algorithme des moindres carrés séquentiels est un algorithme développé en se basant sur l'algorithme SQP (*Sequential Quadratic Programming* en Anglais) ou l'algorithme de l'optimisation quadratique successive. C'est un algorithme utilisé pour résoudre les problèmes d'optimisation non-linéaire tels que les problèmes d'optimisation avec des fonctions objectif quadratiques et des contraintes sous la forme d'un ensemble d'équations et d'inéquations.

Cependant, il peut être utilisé pour résoudre des problèmes linéaires (le cas des problèmes d'optimisation de l'analyse des tolérances de ce manuscrit). L'algorithme des moindres carrés séquentiels est basé sur l'utilisation de la matrice hessienne de la fonction objectif et le gradient des contraintes (équations et inéquations) pour effectuer l'optimisation. Il remplace le problème initial par une séquence de sous-problèmes quadratiques avec des contraintes linéarisées qui sont beaucoup plus simples à résoudre (Gill et Wong). Plus d'informations sur l'algorithme SQP peuvent être trouvées dans (Boggs et Tolle; Gill et Wong; Nocedal et Wright, 1999).

CVXPY est un langage de domaine spécifique intégré à Python et qui permet de résoudre les problèmes d'optimisation convexe. L'optimisation convexe est une optimisation dans laquelle le critère à évaluer et l'ensemble admissible sont convexes. CVXPY permet d'exprimer les problèmes d'optimisation convexes dans une syntaxe naturelle mathématique beaucoup plus simple que la forme restrictive standard d'autres solveurs (Diamond et Boyd). Les problèmes d'optimisation dans CVXPY sont convertis sous la forme conique i.e. sous une forme généralisée de la programmation linéaire. Cette conversion est réalisée en appliquant la méthode de l'implémentation des graphes des fonctions convexes proposée par (Grant et Boyd, 2008). Le problème sous la forme conique est l'équivalent au problème original et en résolvant le problème sous la forme conique, une solution est trouvée au problème original. La forme conique est une large classe qui regroupe les programmes linéaires, les programmes coniques du second ordre et des programmes semi définis. Plusieurs solveurs coniques existent dans le CVXPY : CVXOPT (Vandenberghe, 2010), ECOS (Domahidi et al., 2013), SCS (O'Donoghue et al., 2013), etc. Le solveur de résolution est choisi en fonction du type de problème d'optimisation. Le solveur qui a été choisi dans les études est le solveur ECOS. C'est un solveur de problème d'optimisation sous forme conique intégré à Python et développé par Domahidi *et al.* (2013). Il est basé sur la méthode des points intérieurs.

3.3.2 Résultats – Comparaisons

Le premier critère étudié est la justesse dans le cas des simulations locales des contacts fixes et glissants. Les résultats du Tableau 3.2 montrent que les variations des tirages de résolution et du maillage n'ont pas d'impact sur les résultats de Lpsolve et de Linprog car les composantes du torseur jeu en fonction des différents tirages sont les mêmes pour ces deux solveurs. Cependant, on note de petites différences ou divergences dans les résultats des algorithmes de

SLSQP et de CVXPY. Les algorithmes de Lpsolve et de Linprog sont les meilleurs pour la résolution des optimisations des simulations locales au regard du critère de la justesse.

De plus, des fonctions indicatrices ont été placées dans les algorithmes de résolutions des optimisations au niveau local i.e. sur les composantes du vecteur $\mathbf{G}^{*}_{1a/2a,M}$ afin de déterminer le taux des violations des contraintes pour chaque algorithme en fonction des différents types de maillages. Les résultats sont présentés dans le Tableau 3.1 où on peut remarquer que le taux de violation des contraintes varie entre 0 et 4%. Les violations sont plus importantes pour l'algorithme CVXPY.

La première interprétation qui peut en découler de ces résultats est : Lpsolve et Linprog sont les plus performants pour réaliser les optimisations des simulations locales. Cependant, il est à noter que les temps de calcul de Linprog sont un peu plus élevés que ceux de Lpsolve.

Tableau 3.1 : Résultats sur les violations de contraintes des simulations locales

Algorithme	**Nb de points**	**Nombre total de contraintes**	**Contraintes respectées**
Lpsolve	100	100	100
	625	625	625
	2500	2500	2499
	10000	10000	10000
SLSQP	100	100	100
	625	625	624
	2500	2500	2500
	10000	10000	10000
CVXPY	100	100	96
	625	625	624
	2500	2500	2496
	10000	10000	9998
Linprog	100	100	100
	625	625	625
	2500	2500	2500
	10000	10000	10000

Tableau 3.2 : Résultats des composantes de $\mathbf{G}^{*}_{1a/2a}$ en fonction des algorithmes et des types de maillage (en gris les résultats divergents)

Lpsolve																
Tirages	**1er tirage**			**2e tirage**			**3e tirage**			**4e tirage**			**5e tirage**			**Temps**
Torseur jeu	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	
100	2,79E-02	3,61E-03	1,50E-05	9,53E-02	2,81E-03	5,67E-06	8,57E-02	1,90E-03	1,19E-06	5,16E-02	0	1,66E-05	1,41E-02	0	1,02E-05	2,00E-02
625	1,65E-02	3,39E-03	0	6,64E-02	2,62E-03	1,04E-05	9,82E-02	1,60E-03	2,83E-06	-1,56E-02	0	0	4,10E-02	0	2,27E-05	0,07
2500	7,65E-03	3,87E-03	0	5,23E-02	2,88E-03	2,67E-05	7,99E-02	1,90E-03	0	-1,62E-02	0	0	1,52E-02	0	7,99E-07	0,25
10000	8,05E-03	3,87E-03	1,30E-06	5,12E-02	2,88E-03	8,73E-06	7,88E-02	1,90E-03	3,04E-06	-1,56E-02	0	0	1,38E-02	0	8,34E-06	1,3

SLSQP																
Tirages	**1er tirage**			**2e tirage**			**3e tirage**			**4e tirage**			**5e tirage**			**Temps**
Torseur jeu	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	
100	2,79E-02	3,61E-03	1,50E-05	9,53E-02	2,81E-03	5,67E-04	8,57E-02	1,90E-03	1,19E-06	5,38E-02	-3,49E-04	1,69E-05	2,22E-02	-3,80E-04	9,98E-06	4,00E-02
625	1,21E-02	3,39E-03	-0,00000428	6,64E-02	2,61E-03	1,03E-05	1,02E-01	1,60E-03	2,83E-06	-1,56E-02	0	0	4,10E-02	-0,0002165	2,27E-05	0,08
2500	1,14E-02	3,87E-03	-3,27E-03	5,23E-02	2,88E-03	2,66E-05	7,99E-05	1,90E-03	-2,09E-05	-1,62E-02	0	0	2,17E-02	-3,81E-04	-1,76E-06	0,24
10000	8,05E-03	3,87E-03	1,30E-06	5,12E-02	2,88E-03	8,73E-06	7,88E-02	1,90E-03	3,03E-06	-1,56E-02	0	0	2,15E-02	-3,81E-04	9,95E-06	7,70E-01

CVXPY																
Tirages	**1er tirage**			**2e tirage**			**3e tirage**			**4e tirage**			**5e tirage**			**Temps**
Torseur jeu	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	
100	2,79E-02	3,61E-03	1,50E-05	9,53E-02	2,81E-03	5,66E-06	8,57E-02	1,90E-03	1,19E-06	5,21E-02	-3,49E-04	1,69E-05	1,41E-02	-3,80E-04	9,98E-06	1,20E-01
625	1.65E-02	3,87E-03	-4,28E-06	6,64E-02	2,89E-03	1,04E-05	9,82E-02	1,60E-03	2,83E-06	-1,56E-02	0	0	4,10E-02	-3,81E-04	2,26E-05	0,2
2500	1,14E-02	3,87E-03	-3,27E-05	5,23E-02	2,88E-03	2,67E-05	7,99E-02	1,90E-03	-2,08E-05	-1,62E-02	0	0	1,62E-02	-3,81E-04	-1,75E-06	0,42
10000	8,05E-03	3,87E-03	1,30E-06	5,12E-02	2,88E-03	8,73E-06	7,88E-02	1,90E-03	3,03E-06	-1,56E-02	0	0	1,45E-02	-3,81E-04	9,95E-06	9,80E-01

Linprog																
Tirages	**1er tirage**			**2e tirage**			**3e tirage**			**4e tirage**			**5e tirage**			**Temps**
Torseur jeu	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	***w***	***α***	***β***	
100	2,79E-02	3,61E-03	1,50E-05	9,53E-02	2,81E-03	5,67E-06	8,57E-02	1,90E-03	1,19E-06	5,16E-02	0,00E+00	1,66E-05	1,41E-02	0	1,02E-05	1,30E-01
625	1,65E-02	3,39E-03	0,00E+00	6,64E-02	2,62E-03	1,04E-05	9,82E-02	1,60E-03	2,83E-06	-1,56E-02	0	0	4,10E-02	0	2,27E-05	0,52
2500	7,65E-03	3,87E-03	0,00E+00	5,23E-02	2,88E-03	2,67E-05	7,99E-02	1,90E-03	0,00E+00	-1,62E-02	0	0	1,52E-02	0	7,99E-07	2,71
10000	8,05E-03	3,87E-03	1,30E-06	5,12E-02	2,88E-03	8,73E-06	7,88E-02	1,90E-03	3,04E-06	-1,56E-02	0	0	1,38E-02	0,00E+00	8,34E-06	6,16E+01

La seconde série de simulations porte sur l'analyse des tolérances i.e. des simulations globales afin de déterminer les probabilités de défaillance. Les simulations ont été réalisées en faisant varier la densité du maillage et l'algorithme des optimisations des simulations locales, afin d'évaluer les probabilités d'assemblage et fonctionnelle. Ces simulations ont été réalisées pour le même nombre de tirage de MC $(N_{mc} = 10)$ et pour deux types de maillage : 100 et 10000 points de discrétisation. Le Tableau 3.3 montre uniquement les résultats pour $N_{mc} = 10$ et 100 points de discrétisation pour les simulations locales. Plus d'informations sur ces simulations peuvent être trouvées dans l'Annexe C.

Les résultats montrent les mêmes probabilités de défaillance pour les algorithmes Lpsolve, Linprog, et CVXPY. Les résultats de l'algorithme SLSQP montrent des divergences pour certains résultats de tirages. De plus, les simulations sur les violations de contraintes lors de l'analyse des tolérances sont réalisées en fonction des algorithmes pour les simulations locales et globales, et aussi en fonction des différents types de maillage. Les résultats sur ces violations sont montrés dans le Tableau 3.4. On peut remarquer une faible proportion de violation des contraintes lorsque Lpsolve, Linprog et CVXPY sont utilisés pour réaliser l'analyse des tolérances. Leur taux de violations varient entre 0 et 0.8% pour les inéquations.

Toutefois, pour l'algorithme SLSQP, une forte proportion de violation a été constaté, sauf dans le cas où les algorithmes Lpsolve et Linprog sont utilisés au niveau de la simulation locale. Cette forte proportion de violation des contraintes de l'algorithme SLSQP varie entre 39.11% et 42.67% pour les inéquations et est de 14% pour les équations.

Tableau 3.3 : Résultats des probabilités de défaillance en fonction des algorithmes des simulations locales et globales (en gris les résultats divergents)

méthode simulation globale	**Lpsolve**														
Tirages	1er tirage			2e tirage			3e tirage			4e tirage			5e tirage		
méthodes simulation locale	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time
Lpsolve	0	1	0,15600109	0	1	0,31200194	0	1	0,42120314	0	0	0,57720399	0	0	0,70200396
SLSQP	0	1	0,1404	0	1	0,327	0	1	0,468	0	1	0,592	0	0,8	0,7488
CVXPY	0	1	0,14040112	0	1	0,265	0	1	0,39	0	1	0,546	0	1	0,7176
Linprog	0	0	0,1716	0	1	0,312	0	1	0,4368	0	0	0,5777	0	0	0,702

méthode simulation globale	**SLSQP**														
Tirages	1er tirage			2e tirage			3e tirage			4e tirage			5e tirage		
méthodes simulation locale	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time
Lpsolve	0,3	0,1	0,409	0,2	0,2	1,023	0,2	0,2	1,41	0	0	1,6	0	0	1,79
SLSQP	0,3	0,1	0,4	0,5	0	0,789	0,2	0,2	1,189	0	0	1,67	0	0	2,08
CVXPY	0,4	0,1	0,456	0,1	0,1	0,89	0,1	0,1	1,29	0	0	1,71	0	0	2,09
Linprog	0	0	0,195	0,3	0,2	0,57	0,2	0,2	0,95	0	0	1,156	0	0	1,358

méthode simulation globale	**CVXPY**														
Tirages	1er tirage			2e tirage			3e tirage			4e tirage			5e tirage		
méthodes simulation locale	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time
Lpsolve	0	1	0,3588	0	1	0,7176	0	1	1,0296	0	0	1,326	0	0	1,59121013
SLSQP	0	1	0,281	0	1	0,546	0	1	0,827	0	1	1,12132	0	0,8	1,42
CVXPY	0	1	0,28080201	0	1	0,53040409	0	1	0,7950518	0	1	1,10760713	0	1	1,35720921
Linprog	0	0	0,312	0	1	0,655	0	1	0,9204	0	0	1,295	0	0	1,638

méthode simulation globale	**Linprog**														
Tirages	1er tirage			2e tirage			3e tirage			4e tirage			5e tirage		
méthodes simulation locale	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time	Pnf	Pna	Time
Lpsolve	0	1	3,65	0	1	6,895	0	1	8,97	0	0	11,6	0	0	14,24
SLSQP	0	1	3,49	0	1	8,03	0	1	9,86	0	1	12,34	0	0,8	14,38
CVXPY	0	1	5,1012	0	1	10,249	0	1	15,397	0	1	20,389	0	1	25,6465
Linprog	0	0	2,49601603	0	1	5,66283584	0	1	7,65964489	0	0	10,2648659	0	0	12,9480829

Tableau 3.4 : Résultats sur les violations de contraintes pour les simulations globales (N_{mc} = 10)

Algorithme (simulation globale)	Algorithme (Simulation locale)	Nombre total d'équations	Nombre d'équations respecté	Nombre total d'inéquations	Nombre d'inéquations respecté
Lpsolve	**Lpsolve**	14	14	450	447
	SLSQP	14	14	450	448
	CVXPY	14	14	450	448
	Linprog	14	14	450	447
SLSQP	**Lpsolve**	14	14	450	446
	SLSQP	14	12	450	274
	CVXPY	14	12	450	258
	Linprog	14	14	450	446
CVXPY	**Lpsolve**	14	14	450	448
	SLSQP	14	14	450	450
	CVXPY	14	14	450	450
	Linprog	14	14	450	448
Linprog	**Lpsolve**	12	12	450	446
	SLSQP	12	12	450	448
	CVXPY	12	12	450	448
	Linprog	12	12	450	446

3.3.3 Synthèse

Au regard de toutes les interprétations faites pour les différentes simulations (voir Tableau 3.5), Lpsolve est le mieux adapté afin de réaliser d'abord les simulations locales sur les composantes du vecteur $\mathbf{G}^{*}_{1a/2a,M}$ et ensuite pour mener la simulation globale i.e. l'analyse des tolérances du système mécanique combinée à la simulation de MC. Ainsi l'algorithme de la méthode modifiée du simplexe implémenté dans Lpsolve est utilisé pour les optimisations des analyses des tolérances des systèmes mécaniques présentés dans le chapitre 4.

Tableau 3.5 : Conclusion sur le choix du meilleur algorithme

Méthodes	Méthode globale	Méthode locale	Temps de calcul
Lpsolve	✓	✓	✓
SLSQP	✗	✓	✓
CVXPY	✓	✗	✓
Linprog	✓	✓	✗

La procédure de la méthode d'analyse des tolérances proposée peut parfois demander des efforts numériques importants au regard du nombre de tirage de Monte Carlo à effectuer et du nombre de contraintes qui doivent être vérifiées. Ainsi dans l'optique de réduire ces efforts numériques importants, une nouvelle méthode d'analyse des tolérances est proposée en utilisant la méthode de l'estimation par noyaux.

3.4 Nouvelle approche d'analyse des tolérances basée sur le modèle probabiliste des jeux

La méthode proposée dans cette section a pour objet de générer de nouvelles variables des composantes du torseur jeu $\left\{g^*_{\text{contact}}\right\}_M$ modélisant un contact fixe ou glissant dans l'optique de réaliser moins de simulations locales et de réduire ainsi les temps de calcul. Pour des raisons de lisibilité, nous utilisons dans toute cette section voire dans le reste du manuscrit, le vecteur $\mathbf{G}^*_{\mathbf{contact},M}$ au lieu du torseur jeu $\left\{g^*_{\text{contact}}\right\}_M$. $\mathbf{G}^*_{\mathbf{contact},M}$ correspond au vecteur $\mathbf{G}^*_{kj/ij,M}$ défini dans le chapitre 2, partie 2.3.2.2.

Dans la méthode classique de l'analyse des tolérances proposée précédemment, l'optimisation sur l'assemblage des contacts fixes et glissants est réalisée à chaque itération de MC. Dans l'optique de réduire d'éventuels efforts numériques importants, une première simulation de Monte Carlo est appliquée à chaque contact fixe ou glissant afin de déterminer les réalisations des composantes de son torseur jeu permettant de le modéliser. Nous appellerons les réalisations issues de cette simulation de MC préliminaire, les données originales. Ensuite, la densité de probabilité jointe associée aux composantes des torseurs jeux est identifiée et est par la suite utilisées pour générer de nouvelles variables des composantes des torseurs jeux. C'est la méthode de l'estimation par noyaux (en Anglais *Kernel Density Estimation*) qui est utilisée afin de déterminer la fonction de densité de probabilité jointe des composantes des torseurs jeux.

La définition mathématique de la méthode de l'estimation par noyau est donnée dans la partie 3.4.1, ensuite, la méthode permettant de générer de nouvelles variables de composantes des torseurs jeux en se basant sur la méthode du noyau est décrite dans la sous-section 3.4.2. Enfin, l'intégration des nouvelles variables générées dans la méthode de l'analyse des tolérances est décrite dans la sous-section 3.4.3.

3.4.1 Construction de la densité par l'estimation par noyaux

L'estimation par noyaux est une méthode statistique non-paramétrique qui permet d'estimer la fonction de densité de probabilité d'une variable aléatoire (Tai et Uhlen, 2015). C'est une méthode utilisée à la fois pour les variables continues unidimensionnelles et multidimensionnelles. Avant l'application de cette méthode, une simulation de Monte Carlo est lancée afin de constituer un ensemble de réalisation des composantes des torseurs jeux des contacts fixes et glissants. L'objectif est alors de déterminer la fonction de densité de probabilité de ces réalisations en appliquant l'estimation par noyaux. Il existe plusieurs autres méthodes permettant d'estimer la fonction densité de probabilité des variables aléatoires (Silverman, 1986) : les histogrammes, l'estimateur naïf, la méthode des plus proches voisins, l'estimateur par noyau variable, les estimateurs de séries variables et les estimateurs du maximum de vraisemblance pénalisée. Cependant, dans le cadre des travaux de thèse, l'estimation par noyau est la méthode que nous avons choisie. Un algorithme est employé par la suite pour générer de nouveaux échantillons pour l'analyse des tolérances. La densité de probabilité obtenue par l'estimation par noyau pour une variable aléatoire continue unidimensionnelle *y* est définie par :

$$f\left(y\right)=\frac{1}{n}\sum_{i=1}^{n}K\left(\frac{y-y^{(i)}}{h}\right) \tag{3.13}$$

où $K(.)$ représente la fonction du noyau centrée sur l'observation $y^{(i)}$, h représente est un paramètre indiquant le degré de lissage de l'estimation, *n* le nombre total de variable. Dans la littérature, plusieurs types de noyaux existent (Silverman, 1986) : le noyau de Epanechnikov, le noyau biweight, le noyau triangulaire, le noyau gaussien et le noyau rectangulaire. Le noyau gaussien a été choisi dans ces travaux de thèse compte tenu de sa simplicité et aussi de sa capacité à approximer les distributions des variables des composantes des torseurs jeux $\left\{g^{*}_{\text{contact}}\right\}$. Le noyau gaussien est défini comme suit :

$$K\left(u\right)=\frac{1}{h\sqrt{2\pi}}\exp\left(-\frac{1}{2}u^{2}\right) \tag{3.14}$$

où $u=\left(y-y^{(i)}\right)/h$.

Dans le cadre des travaux menés de cette thèse, les travaux portent sur les composantes des vecteurs $\mathbf{G}^{*}_{\mathbf{contact},M}$ des contacts fixes et glissants. Les études sont confrontées dans ce cas à des

variables aléatoires continues multivariées qui peuvent être corrélées. L'estimation de la densité par la méthode multivariée de l'estimation par noyau est alors utilisée. Ainsi, $K(.)$ représente la fonction du noyau gaussien multivarié et la variable aléatoire *y* redevient un ensemble de plusieurs variables unidimensionnelles ($\mathbf{y} = [Y_1, Y_2, ..., Y_d]$ où *d* est la dimension du paramètre ***y***). La fonction du noyau gaussien multivarié peut s'écrire sous la forme (Rajagopalan et al., 1997) :

$$K(u) = \frac{1}{(2\pi)^{d/2} \det(\mathbf{S})^{1/2} h^d} \exp(-u/2) \tag{3.15}$$

Où $u = \frac{\left(\mathbf{y} - \mathbf{Y}^{(i)}\right)^T \mathbf{S}^{-1} \left(\mathbf{y} - \mathbf{Y}^{(i)}\right)}{h^2}$, $\mathbf{Y}^{(i)} = \left[Y_1^{(i)}, Y_2^{(i)}, ..., Y_d^{(i)}\right]^T$ *i*=1,…, *n*, *n* représente le nombre total d'échantillon de variables aléatoires généré grâce à la méthode du noyau, **S** représente la matrice de covariance des variables. C'est cette fonction du noyau gaussien multivarié qui est utilisée dans la suite des travaux de thèse.

Un des paramètres important de cette approche et à prendre avec précaution est la fenêtre de lissage *h*. Ce paramètre permet de contrôler le degré de lissage de la densité estimée des variables (Rajagopalan et al., 1997). Il existe plusieurs techniques permettant de déterminer la meilleure valeur pour ce paramètre *h* : la méthode de la validation croisée, la méthode du plug-in, la règle de Scott, la règle Silverman (Heidenreich et al., 2013; Scott, 1992). La méthode de la validation croisée et celle du plug-in sont des méthodes basées sur des techniques d'optimisation afin de déterminer la meilleure valeur du paramètre *h* et cela est couteux en temps de calcul (Rajagopalan et al., 1997). Ainsi, la règle de Silverman a été choisie dans ces travaux de thèse pour estimer le paramètre *h*. En se basant sur la règle de Silverman, l'expression du calcul du paramètre *h* est donnée par :

$$h = \left(\left(4/(2d+1)\right)^{1/(d+4)} \right) . n^{-1/(d+4)} \tag{3.16}$$

où *n* représente le nombre total de données, *d* le nombre total de variables aléatoires (ici, le nombre des composantes du torseur jeu).

La méthode d'estimation de la densité basée sur l'estimation par noyau est par la suite utilisée afin de générer de nouvelles variables ayant les mêmes caractéristiques que celles des données originales. Cette méthode est décrite ci-après.

3.4.2 Génération de nouvelles variables basées sur l'estimation par l'estimation par noyaux

L'estimation par noyaux est tout d'abord utilisée afin de déterminer la fonction densité de la probabilité jointe des variables des composantes du vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$ d'un contact fixe ou glissant. Ensuite en se basant sur la densité de probabilité jointe des variables, de nouvelles variables sont générées et leur utilisation permet de remplacer la simulation locale menée à chaque itération de Monte Carlo afin de déterminer les meilleures composantes du torseur jeu permettant à un contact fixe ou glissant de s'assembler.

Dans la méthode classique de l'analyse des tolérances décrite dans la section 3.2, un échantillon de l'ensemble **X** (ensemble des composantes des défauts géométriques et dimensionnels) est généré à chaque itération de la simulation de MC, ainsi qu'un échantillon de l'ensemble **F** des défauts de forme. Les déviations découlant de ces deux ensembles sont appliquées à chacune des surfaces des contacts fixes et glissants afin de déterminer les composantes du vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$ associé à ces contacts. La simulation locale consiste à optimiser les composantes du vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$ (par exemple α, β, w pour un contact plan-plan) tel que les distances signées entre les deux surfaces de la liaison soit soient supérieures ou égales à zéro. Afin d'utiliser la méthode probabiliste développée dans cette sous-section, il est indispensable d'appliquer une simulation de MC à cette simulation locale i.e. réaliser cette simulation un certain nombre de fois afin d'établir des réalisations des composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$. Ces réalisations sont considérées comme les données originales et sont représentées par la suite par la matrice **D**. L'objectif après est de déterminer la densité de probabilité jointe de ces réalisations de $\mathbf{G}^{*}_{\mathbf{contact},M}$ et de générer de nouvelles variables permettant de réaliser l'analyse des tolérances d'un système mécanique hyperstatique. L'objectif à terme est de réaliser un nombre réduit de simulation MC préliminaire pour déterminer les données originales (i.e. $N_{KDE} < N_{mc}$) et ensuite de les utiliser en employant la méthode du noyau pour générer un nombre d'échantillons plus grand pour l'analyse des tolérances, afin de réduire le temps de simulation.

Plusieurs expériences réalisées sur l'optimisation des composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$ des contacts fixes et glissants révèlent qu'il y a une importante variabilité des distributions marginales de ces composantes. Aussi, la structure de dépendance entre les réalisations de ces composantes est complexe. Pour déterminer les lois marginales de chaque composante d'un vecteur

$\mathbf{G}^{*}_{\mathbf{contact},M}$, il est envisageable par exemple d'utiliser un ensemble de lois prédéfinies (loi normale, loi uniforme, loi bêta, loi gamma etc.) combinées avec des méthodes basées sur les copules (copules Elliptiques, copules Archimédiens, copules Marshall-Olkin etc.) (Embrechts et al., 2003) pour déterminer la structure de dépendance. Toutefois, ces méthodes classiques ne sont assez flexibles pour décrire avec précision la distribution marginale et la structure de dépendance entre les réalisations des composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$. L'estimation par noyaux a donc été choisie à cause de sa simplicité, sa flexibilité et aussi sa précision de générer de nouvelles variables respectant les distributions marginales originales et aussi respectant la structure de dépendance.

Un algorithme est utilisé afin de générer les nouvelles variables des composantes des torseurs jeux qui sont utilisées dans la méthode d'analyse des tolérances (voir Algorithme 1). Les nouvelles variables basée sur la méthode d'estimation par noyau ont été obtenues en utilisant les réalisations originales de $\mathbf{G}^{*}_{\mathbf{contact},M}$ (les données originales). Comme expliqué ci-dessous, une première simulation de MC a été lancée sur un contact fixe ou glissant afin de récolter les différentes réalisations des composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$ qui formeront la matrice **D**, matrice des données originales. Certains paramètres doivent être définis pour décrire l'algorithme.

ALGORITHME 1: Générer de nouvelles variables basées sur l'estimation par noyaux

Etape 1: Considérons $\mathbf{D} = \left[\mathbf{G}^{*}_{\mathbf{contact},M,1}, \mathbf{G}^{*}_{\mathbf{contact},M,2}, \ldots, \mathbf{G}^{*}_{\mathbf{contact},M,N_{KDE}}\right]$ où $\mathbf{G}^{*}_{\mathbf{contact},M,\mathrm{i}}$ est de taille *1* x *k, i=1 à* N_{KDE}, obtenu en réalisant N_{KDE} fois l'optimisation des paramètres du torseur jeu d'un contact fixe ou glissant. Tirer uniformément $\mathbf{G}^{*}_{\mathbf{contact},M,\mathrm{i}}$ de $\left[\mathbf{G}^{*}_{\mathbf{contact},M,1}, \mathbf{G}^{*}_{\mathbf{contact},M,2}, \ldots, \mathbf{G}^{*}_{\mathbf{contact},M,N_{KDE}}\right]$.

Etape 2: Simuler les nouvelles variables du noyau i.e. des variables normales multidimensionnelles centrées sur $\mathbf{G}^{*}_{\mathbf{contact},M,\mathrm{i}}$ et ayant comme écart-type $h^2 * \mathbf{S}$. Cette simulation $\mathcal{N}(\mathbf{G}^{*}_{\mathbf{contact},M,\mathrm{i}}, h^2 * \mathbf{S})$ donne une matrice de dimension N_{KDE} x *k*. *h* représente le paramètre de lissage et $\mathbf{S}$ représente la matrice de covariance des variables.

Etape 3: Les nouvelles variables obtenues représentent le modèle probabiliste des composantes du torseur jeu qui sont ensuite introduites dans les contraintes à développer (équations de compatibilité, conditions fonctionnelles, contraintes d'interface).

Prenons par exemple N_{KDE} le nombre total de la première simulation de MC, *k* le nombre de composantes du torseur jeu du contact fixe ou glissant, et **D** la matrice représentant les données originales et de dimension N_{KDE} x k. Pour déterminer le modèle probabiliste associé à

l'estimation par noyaux, les données originales sont uniformément tirées et de nouvelles variables normales multidimensionnelles centrées sur le précédent tirage sont simulées. La nouvelle matrice qui est obtenue de l'application de cette loi normale représente le modèle probabiliste des composantes du torseur jeu du contact fixe ou glissant. Un algorithme a été utilisé (Rajagopalan et al., 1997) pour la détermination du modèle probabiliste et est détaillé dans l'algorithme 2 :

ALGORITHME 2: Modèle probabiliste caractérisant le jeu d'un contact fixe ou glissant

Entrée : Le nombre total des échantillons de Monte Carlo **(** N_{KDE} **)**

Entrée : Les surfaces nominales d'un contact fixe ou glissant

Entrée : Une matrice vide **D** de dimension N_{KDE} x k

Assurer : le nombre des itérations d'optimisation à un contact

Etapes//

1: Générer les amplitudes des défauts de forme pour chaque surface du contact, $\mathbf{F} = \{\mathbf{F}^{(1)}, \mathbf{F}^{(2)}, \ldots, \mathbf{F}^{(N_{KDE})}\}$ où $\mathbf{F}^{(i)} = [f_1, f_2, \ldots, f_n]$

n =nombre des points de maillage sur chaque surface

2: Appliquer les amplitudes des défauts de forme aux surfaces discrétisées du contact

3: Pour chaque i $\in$ N_{KDE} **faire**

4: **Calculer min** $f\left(\mathbf{G}^{*}_{\mathbf{contact},M}\right)$ telle que **Distances Signées (S_1, S_2) ≥ 0** (S_1, S_2: les surfaces non-idéales potentiellement en contact)

5: // Changer la i^{ième} ligne de **D** par les composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$

6: fin

7: Tirer uniformément $\mathbf{G}^{*}_{\mathbf{contact},M,\mathrm{i}}$ de $\mathbf{D}=\left[\mathbf{G}^{*}_{\mathbf{contact},M,1}, \mathbf{G}^{*}_{\mathbf{contact},M,2}, \ldots, \mathbf{G}^{*}_{\mathbf{contact},M,N_{mc}}\right]$**.** Ceci permet d'obtenir la matrice **M**.

8: Générer les variables normales multivariées **N** centrées sur 0 et ayant h^2x **S** comme écart-type, **N** est une matrice de dimension N_{KDE} x k .

Sortie :

Calculer **P = M + N**. Chaque colonne de **P** représente le modèle probabiliste de chaque composante du torseur jeu.

Le processus de détermination du modèle probabiliste associé au vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$, combine à la fois la procédure de l'algorithme 1 et la méthode d'optimisation basée sur l'introduction des distances signées comme évoqué dans le chapitre 2. Les variables du modèle probabiliste caractérisant le jeu d'un contact fixe ou glissant, sont par la suite, introduites dans le processus global de l'analyse des tolérances d'un système mécanique hyperstatique.

3.4.3 Intégration des nouvelles variables dans la démarche de l'analyse des tolérances

La méthode classique d'analyse des tolérances d'un système mécanique hyperstatique proposée dans la section 4.2, est basée sur deux méthodes d'optimisation : (i) une première optimisation concerne le vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$ d'un contact fixe ou glissant, effectuée à chaque itération de MC, (ii) une deuxième optimisation est combinée à une simulation de MC et permet de vérifier si le système mécanique est globalement assemblable et fonctionnel. La méthode de détermination du modèle probabiliste des composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$ est principalement appliquée au niveau de la première optimisation afin d'éviter une seconde optimisation imbriquée dans l'optimisation générale.

Ainsi, une méthode permettant de déterminer les modèles probabilistes des composantes d'un vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$ a été proposée dans la section 3.3.2 afin de pallier les inconvénients liés à ces efforts numériques. Le processus d'établissement de ces modèles probabilistes est détaillé précédemment. La matrice **P** contient les composantes du modèle probabiliste des composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$ qui sont intégrées dans le processus général de l'analyse des tolérances. Pour réaliser l'analyse des tolérances, il y a certaines contraintes qui sont développées afin de modéliser le comportement géométrique d'un système mécanique hyperstatique : les équations de compatibilité, les conditions fonctionnelles et les contraintes d'interfaces. Les composantes de la matrice **P** sont introduites principalement dans les équations de compatibilité et les conditions fonctionnelles. Ainsi, à chaque itération de MC, les composantes d'une ligne de la matrice **P** sont introduites dans les équations de compatibilité voire de conditions fonctionnelles comme variables connues. Il est important de noter que dans le cas des contacts glissants, il y a certaines composantes des torseurs jeux qui ne sont déterminées en considérant uniquement les composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$. Il s'agit des composantes associées aux déplacements cinématiques de ces contacts glissants. Ils sont ajoutées comme des variables inconnues à la seconde méthode d'optimisation où sont déterminées les composantes des torseurs des contacts flottants ($\mathbf{G}^{'}$) et qui servent ensuite à calculer les probabilités de défaillance d'assemblage et de fonctionnalité d'un système mécanique. La Figure 3.4 permet de synthétiser la nouvelle méthode d'analyse des tolérances basée sur le développement d'un modèle probabiliste caractérisant le jeu des contacts fixes et glissants.

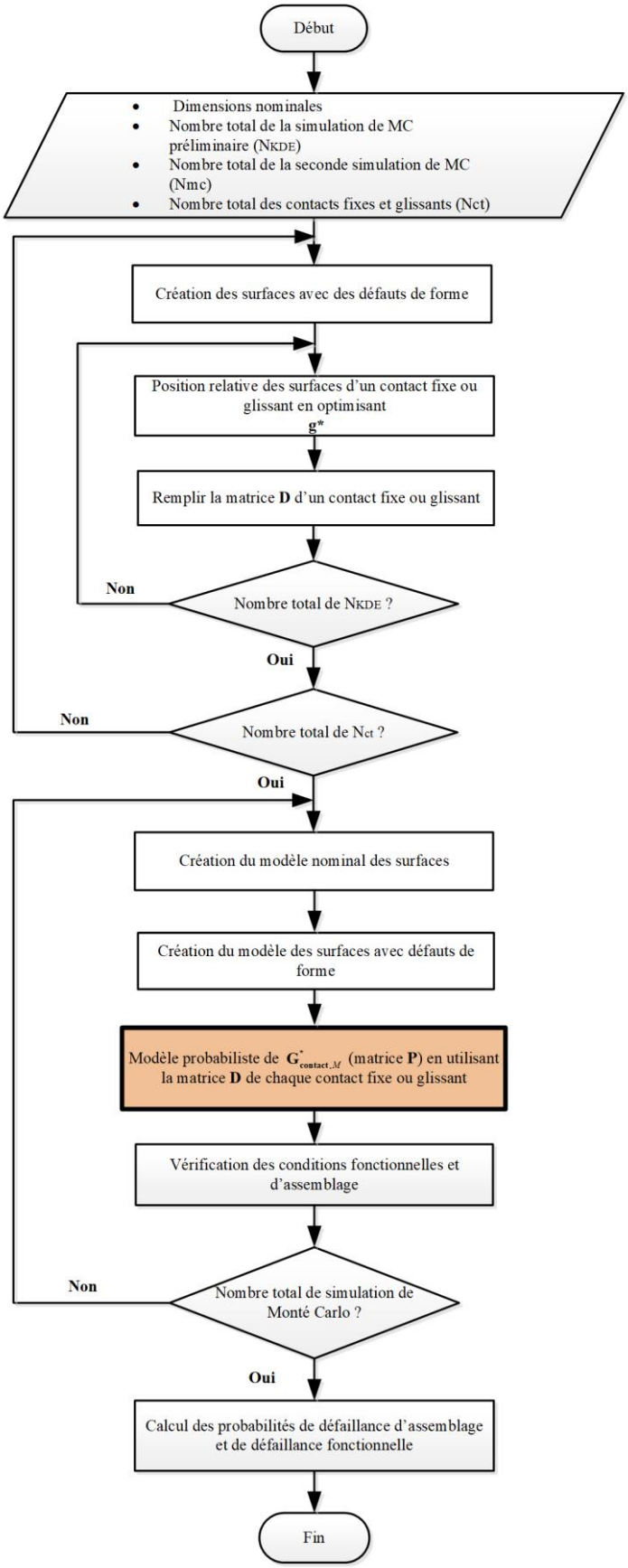

Figure 3.4 : Approche globale améliorée de l'analyse des tolérances

3.4.4 Justification de la proposition d'un modèle probabiliste pour la caractérisation du jeu

Cette partie synthétise les avantages de l'utilisation du modèle probabiliste caractérisant le jeu d'un contact fixe ou glissant.

- Certains systèmes mécaniques sont composés de plusieurs contacts fixes ou glissants identiques. Dans ce cas par exemple, l'introduction de modèles probabilistes dans la démarche globale de l'analyse des tolérances permet de réaliser juste une seule fois la simulation préliminaire de MC permettant d'obtenir la matrice des données originales (**D**). Une fois obtenue pour un contact, elle pourra être utilisée pour évaluer la fonction densité de probabilité conjointe de son contact et aussi des autres contacts identiques. En effet les contacts identiques ont les mêmes défauts élémentaires et les mêmes caractéristiques dimensionnelles. D'où les composantes du vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$ qui sont évaluées sont identiques. En prenant la matrice des données originales d'un seul contact, les modèles probabilistes des composantes de son torseur et ainsi que ceux des autres contacts identiques peuvent être déterminés. Il n'est donc plus nécessaire de refaire la simulation de MC préliminaire pour les autres contacts identiques, ce qui permet de réduire le temps de calcul.
- La méthode de l'analyse des tolérances basée sur le modèle probabiliste des composantes de $\mathbf{G}^{*}_{\mathbf{contact},M}$, permet d'étudier séparément les contacts fixes et glissants. Ainsi, un large spectre de ces contacts fixes et glissants peut être étudié en avance pour déterminer leur matrice **D** respective. Ces simulations préliminaires peuvent être gourmandes en termes de temps de calcul mais une fois que la conception des systèmes mécaniques est lancée, les modèles probabilistes sont rapidement établis et introduits dans la méthode générale de l'analyse des tolérances. Ce qui permet de gagner en temps de calcul et de conception.
- Pour réaliser l'analyse des tolérances en utilisant le modèle probabiliste des réalisations des composantes d'un vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$, deux simulations de MC sont utilisées. La première simulation de MC permet de déterminer les données originales qui sont utilisées pour développer le modèle probabiliste des composantes d'un vecteur $\mathbf{G}^{*}_{\mathbf{contact},M}$ en utilisant l'estimation par noyaux. La seconde simulation de MC est associée à une technique d'optimisation et l'ensemble permet de réaliser l'analyse des tolérances du

système mécanique hyperstatique. Supposons que le paramètre N_{KDE} soit le nombre d'échantillons de la première simulation de MC et le paramètre N_{mc} soit le nombre d'échantillons de la seconde simulation de MC. Une autre utilité de l'utilisation du modèle probabiliste pour l'analyse des tolérances est de choisir un nombre N_{KDE} inférieur à N_{mc} ($N_{mc} > N_{KDE}$ ou $N_{mc} >> N_{KDE}$). En effet N_{KDE} peut être choisi de telle manière que les échantillons des données originales soient suffisants pour déterminer une densité de probabilité avec une précision acceptable et aussi obtenir des probabilités de défaillance suffisamment précises. Il n'est pas nécessaire que N_{KDE} soit égal à N_{mc}, ce qui permet de réduire le temps de calcul. Pour montrer l'utilité de la démarche proposée à ce point, une série de simulations a été lancée pour différents nombres N_{KDE}, pour un nombre N_{mc} fixe égal à 100000, pour un nombre de points de discrétisation fixe d'un contact glissant égal à 2500 et pour les amplitudes des défauts de forme suivant la loi uniforme $\mathcal{U}(1.5\text{x}10^{-3}, 3.5\text{x}10^{-3})$. Les résultats sont montrés dans le Tableau 3.6 où on peut remarquer que $N_{KDE} = 10000$ est déjà suffisant pour développer le modèle probabiliste puis réaliser l'analyse des tolérances du système mécanique car les probabilités calculées sont stables.

Tableau 3.6 : Résultats en fonction des différents nombres N_{KDE}

Simulations	**Résultats**				
			2500		
N_{KDE}	**10000**	**20000**	**30000**	**50000**	**100000**
Probabilités	P_{nf} = 1.63x10^{-2} P_{na} = 3.00x10^{-5}	P_{nf} = 1.64x10^{-2} P_{na} = 3.00x10^{-5}	P_{nf} = 1.63x10^{-2} P_{na} = 3.00x10^{-5}	P_{nf} = 1.63x10^{-2} P_{na} = 3.00x10^{-5}	P_{nf} = 1.63x10^{-2} P_{na} = 3.00x10^{-5}
T_1	10min	13min	19min	24min	50min
T_2	9h43min	9h44min	9h42min	9h45min	9h43min
Temps de calcul (T_1 + T_2)	9h53min	9h57min	10h01min	10h09min	10h33min

- Il existe d'autres méthodes que la simulation de MC pour réaliser l'analyse des tolérances d'un système mécanique hyperstatique. La méthode FORM (Dumas et al., 2015a) peut être par exemple utilisée pour résoudre les problèmes fonctionnels des systèmes mécaniques hyperstatiques. La proposition d'un modèle probabiliste des composantes du torseur jeu pour réaliser l'analyse des tolérances est une démarche qui peut être appliquée comme celle qui est proposée dans cette partie mais aussi être appliquée pour la méthode FORM.

3.5 Conclusion

Dans ce chapitre deux approches d'analyse des tolérances des systèmes mécaniques hyperstatiques ont été présentées. Les deux méthodes sont basées sur des simulations de MC associées à des techniques d'optimisation. Des simulations préliminaires ont été réalisées afin de déterminer l'algorithme d'optimisation le plus adapté aux études. Plusieurs algorithmes d'optimisation ont été comparés et l'algorithme basé sur la méthode modifiée du simplexe a été choisi en raison de ses performances supérieures.

Dans la première méthode d'analyse des tolérances nommée méthode classique, des simulations locales sont appliquées sur les contacts fixes et glissants pour déterminer les configurations optimales des jeux permettant de satisfaire les contraintes de non-interpénétrations entre les surfaces potentiellement en contact. Cependant, ces simulations locales répétées à chaque itération de la méthode de MC nécessitent des efforts numériques importants. Pour pallier ce problème, des modèles probabilistes des réalisations des composantes des vecteurs $\mathbf{G}^{*}_{\mathbf{contact},M}$ des contacts fixes et glissants sont par la suite proposés. Ce modèle probabiliste est obtenu en faisant appel à l'estimation par noyaux. L'utilisation du modèle probabiliste dans la démarche d'analyse des tolérances offre plusieurs avantages dont l'utilisation d'un petit nombre d'échantillons pour déterminer les réalisations desquelles est appliquée à l'estimation par noyaux dans le but de minimiser le temps de calcul. Les différentes méthodes développées dans ce chapitre ont été appliquées à de différents systèmes mécaniques.