# Optimisation multi-niveaux d'assemblages en contact frottant

Bruno Soulier, Pierre-Alain Boucard

CSMA 2011
10e Colloque National en Calcul des Structures
9-13 Mai 2011, Presqu'île de Giens (Var)

# Optimisation multi-niveaux d'assemblages en contact frottant

B. Soulier, P-A. Boucard[1]

[1] *LMT-Cachan, ENS Cachan/CNRS/UPMC/PRES UniverSud Paris, France, {soulier,boucard}@lmt.ens-cachan.fr*

**Résumé** — Le temps de calcul lié aux simulations numériques constitue le principal verrou des stratégies d'optimisation structurale. Une méthode d'optimisation multi-niveaux de modèles basée sur l'optimisation successive d'un métamodèle et du modèle mécanique complet est proposée. Le couplage à une stratégie multiparamétrique basée sur une méthode non incrémentale permet de réduire les coûts de calcul inhérents à la génération du métamodèle ainsi que ceux relatifs à la phase d'optimisation sur le modèle complet. La méthode est illustrée sur des cas d'optimisation d'assemblages en contact frottant.
**Mots clés** — optimisation, multi-niveaux de modèles, métamodèle, sous-structuration, méthode *LATIN*, multiparamétrique.

## 1 Introduction

La conception optimale et robuste de structures fait appel de façon massive à la simulation numérique où les phénomènes simulés sont de plus en plus détaillés. Pour aider l'ingénieur concepteur dans ses démarches d'optimisation qui deviennent de plus en plus complexes par leur caractère multi-niveaux et multi-disciplinaires, des méthodes numériques d'exploration de l'espace de conception par approximation globale sont développées [13]. De manière générale, les fonctions objectif et contraintes d'un problème d'optimisation structurale sont implicites par rapport aux variables de conception : leurs évaluations sont synonymes d'analyses éléments finis et constituent, de loin, l'opération la plus coûteuse de l'algorithme de minimisation. Les travaux présentés ici ont pour objectif de limiter les coûts de calculs, véritable verrou aux problèmes d'optimisation, en couplant une démarche d'optimisation multi-niveaux [3] et une stratégie de calcul multiparamétrique [1] [2] basée sur l'algorithme itératif *LATIN* développé par Ladevèze [9]. La stratégie d'optimisation multi-niveaux est présentée dans la première partie, ensuite nous rappelons le principe de la méthode *LATIN* appliquée aux problèmes de contacts ainsi que les idées principales de la stratégie multiparamétrique. Les gains en temps de calcul sont illustrés sur un cas d'optimisation d'assemblages en contact frottant.

## 2 Optimisation multi-niveaux

L'optimisation structurale fait intervenir deux domaines de compétence bien différents : la simulation numérique et l'optimisation. A partir d'un ensemble de variables de conception $x$ fourni par l'optimiseur, le simulateur retourne la (les) fonctions objectives correspondante(s) $f(x)$ et éventuellement les fonctions contraintes $g(x)$ (1).

$$\begin{cases} \min_x f(x) \\ g(x) \leq 0 \\ \underline{x_i} \leq x_i \leq \overline{x_i} \end{cases} \tag{1}$$

Les simulations numériques constituent l'opération la plus coûteuse de l'algorithme de minimisation plus particulièrement dans les problèmes d'optimisation fortement non-linéaire comme la modélisation du contact avec frottement dans les assemblages. De plus ces démarches d'optimisation nécessitent l'emploi d'optimiseur globaux pour atteindre l'optimum global. Afin de limiter les coûts de calcul une stratégie d'optimisation multi-niveaux est mise en place.

## 2.1 Optimisation multi-niveaux de modèle

les méthodes d'optimisation multi-niveaux peuvent se classifier en trois catégories. On distingue *l'optimisation parallèle de modèle* basée sur des méthodes de décomposition de domaines [4] [16], *l'optimisation multi-niveaux de paramètres* qui consiste à remplacer un problème d'optimisation par plusieurs sous-problèmes avec chacun un jeu réduit de paramètres [8] [10], et *l'optimisation multi-niveaux de modèles* [5] [12] qui introduit plusieurs niveaux de modélisation. La méthode d'optimisation multi-niveaux de modèles utilisée ici (figure 1) est basée sur l'optimisation successive d'un métamodèle constituant le premier niveau de modélisation et du modèle mécanique complet constituant le second niveau.

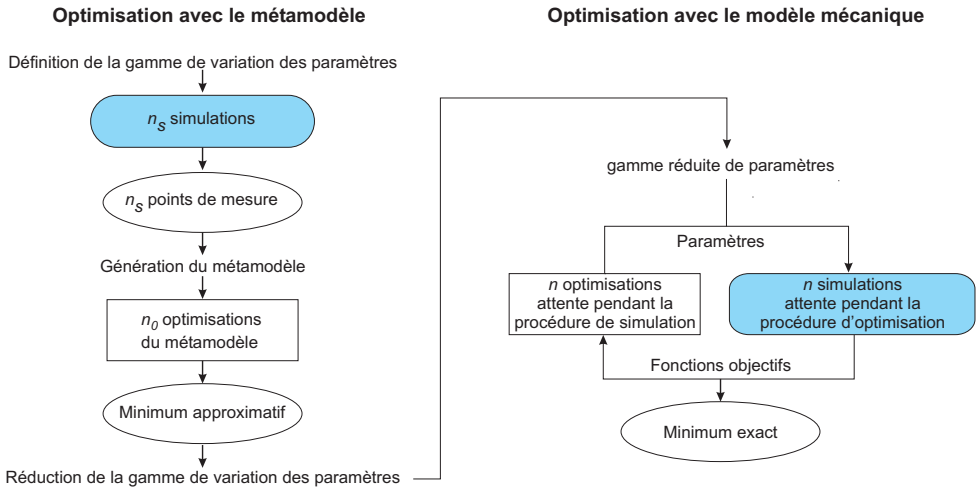


FIGURE 1 – Schéma d'optimisation multi-niveaux

Cette démarche d'optimisation multi-niveaux de modèles est initialisée par la génération du métamodèle qui se nourrit de $n_s$ simumations numériques. Ensuite, la première étape a pour objectif de localiser la vallée où se situe l'optimum global par une stratégie d'optimisation globale réalisée à moindre coûts sur le métamodèle avec un algorithme génétique [7]. Partant de cette vallée, la recherche de l'optimum global est réalisée dans une seconde étape par une stratégie d'optimisation locale sur le modèle mécanique complet via un algorithme de descente de type gradient. L'optimisation locale sur le modèle mécanique complet conduit à $n$ simulations complémentaires. La réduction des coûts de calcul dans l'ensemble de la démarche d'optimisation nécessite la diminution des coûts de calcul inhérents à la génération du métamodèle qui nécessite $n_s$ simulations sur le modèle mécanique complet, mais aussi la diminution dans la phase d'optimisation sur le modèle complet qui conduit à réaliser $n$ nouvelles simulations. Pour réduire ces coûts nous couplons une stratégie dite « multiparamétrique » basée sur une méthode non incrémentale et un métamodèle construit par interpolation diffuse.

## 2.2 Construction du métamodèle

Nous utilisons dans ces travaux un métamodèle d'interpolation cumulative [14] qui s'inspire des méthodes SPH [6] [11]. Ce métamodèle (2) permet de restituer les non-linéarités des fonctions objectifs et contraintes d'optimisation et d'en calculer explicitement les gradients.

$$\tilde{y}(x) = \frac{\sum_i \phi_i(x) a_i}{\sum_i \phi_i(x)} \quad avec \quad \phi_i(x) = e^{-k||x-x_i||^2} \tag{2}$$

Les fonctions de forme $\phi_i$ sont choisies de type gaussiennes. Les coefficients $a_i$ sont obtenus par une méthode de collocation.

De plus, la flexibilité de ce métamodèle facilite la remise à jour des surfaces d'interpolation au cours du processus d'optimisation, ce qui permet d'améliorer localement la qualité des approximations.

# 3 Stratégie multiparamétrique avec la méthode *LATIN*

L'optimisation engendrant un nombre important de calculs, il est crucial d'introduire une stratégie de résolution adaptée permettant de réduire les temps de calcul lorsque les paramètres évoluent. Pour cela nous utilisons l'algorithme itératif *LATIN* développé par Ladevèze [9] associé à une stratégie multiparamétrique.

## 3.1 Rappel de la méthode *LATIN* pour les problèmes de contact avec frottement

Le principe de base de la méthode *LATIN* est de séparer les difficultés en résolvant tour à tour deux groupes d'équations : les équations locales et éventuellement non-linéaires et les équations linéaires éventuellement globales.

Un assemblage est composé d'un ensemble de sous-structures (typiquement les pièces de l'assemblage) qui communiquent au travers d'interfaces (voir figure 2). Chaque interface est une entité mécanique à part entière qui possède ses propres inconnues et relations de comportement. Pour simplifier la présentation, nous ne considérons que deux sous-structures $\Omega_E$ et $\Omega_{E'}$ reliées par une interface $\Gamma^{EE'}$. Les variables d'interface sont deux champs de forces $\underline{F}^E$ et $\underline{F}^{E'}$ et deux champs de vitesses duaux $\underline{\dot{W}}^E$ et $\underline{\dot{W}}^{E'}$ (figure 2). Par convention, $\underline{F}^E$ et $\underline{F}^{E'}$ sont les actions des interfaces sur les sous-structures et $\underline{\dot{W}}^E$ et $\underline{\dot{W}}^{E'}$ sont les vitesses des sous structures vues par l'interface.

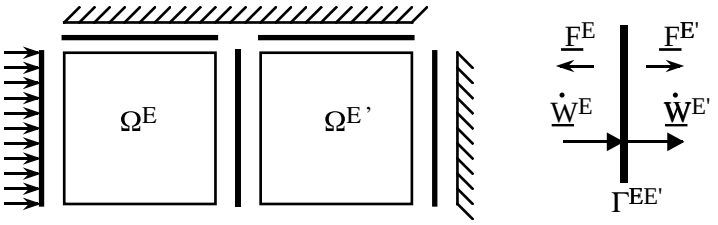


FIGURE 2 – Décomposition d'un assemblage et variables d'interface

Le champ de déplacement en tout point $M$ de $\Omega_E$ et à tout instant $t$ de $[0,T]$ est $\underline{U}^E(M,t)$. Le problème à résoudre sur chaque sous-structure est alors de trouver l'histoire du champ de déplacement $\underline{U}^E(M,t)$ et du champ de contrainte $\sigma^E(M,t)$ vérifiant :

– l'admissibilité cinématique :

$$\varepsilon = \varepsilon(\underline{U}^E); \quad \underline{U}^E(M,t)_{|\partial\Omega_E} = \underline{W}^E(M,t); \quad \underline{U}^E \in \mathcal{U}_{ad}^{[0,T]} \tag{3}$$

– l'admissibilité statique : $\forall \underline{\dot{U}}^\star \in \mathcal{U}$

$$\int_{\Omega_E} Tr(\sigma^E \varepsilon(\underline{\dot{U}}^\star))d\Omega_E - \int_{\Omega_E} \underline{f}_d.\underline{\dot{U}}^\star d\Omega - \int_{\partial\Omega_E} \underline{F}^E.\underline{\dot{U}}^\star dS = 0 \tag{4}$$

– le comportement élastique : $\forall M \in \Omega_E$ et $\forall t \in [0,T]$,

$$\sigma^E(M,t) = \mathcal{K}_e \varepsilon(\underline{U}^E(M,t)) \tag{5}$$

($\mathcal{K}_e$ est l'opérateur de Hooke)

Le problème à résoudre sur chaque interface est de trouver l'histoire des champs de force ($\underline{F}^E(M,t)$ et $\underline{F}^{E'}(M,t)$) et de vitesse ($\underline{\dot{W}}^E(M,t)$ et $\underline{\dot{W}}^{E'}(M,t)$) vérifiant :

– l'équilibre : $\forall M \in \Gamma^{EE'}$ et $\forall t \in [0,T]$,

$$\underline{F}^E(M,t) + \underline{F}^{E'}(M,t) = 0 \tag{6}$$

– le comportement décrit sous la forme d'une loi d'évolution non-linéaire $\mathcal{R}$ entre les forces et le taux de saut de déplacement $\underline{\dot{W}}^{EE'}$ au travers de l'interface (typiquement : contact, frottement, etc) :
$\forall M \in \Gamma^{EE'}$ et $\forall t \in [0,T]$,

$$\underline{F}^E(M,t) = \mathcal{R}(\underline{\dot{W}}^{EE'}(M,\tau), \tau \in [0,t]) \tag{7}$$

La solution $s$ est écrite comme un ensemble de champs dépendants du temps, relatifs à la fois aux interfaces et aux sous-structures :

$$s = \sum_E s^E \quad ; \quad s^E = \left\{ \underline{U}^E(M,t), \boldsymbol{\sigma}^E(M,t), \underline{\dot{W}}^E(M,t), \underline{F}^E(M,t) \right\} \quad t \in [0,T]$$

On partage les équations en deux groupes où les sous-structures sont considérées comme élastiques, et où toutes les non-linéarités sont concentrées sur les interfaces.

- L'ensemble $\mathcal{A}_d$ des solutions $s^E$ qui vérifient les équations **linéaires** relatives aux sous-structures ;
- L'ensemble $\Gamma$ des solutions $s^E$ qui vérifient les équations **locales** (éventuellement non-linéaires) relatives aux interfaces.

La détermination de la solution du problème se fait alors itérativement (figure 3) par la recherche d'approximations successives $s$ qui vérifient alternativement les deux groupes d'équations en utilisant des directions de recherche $E^+$ et $E^-$.

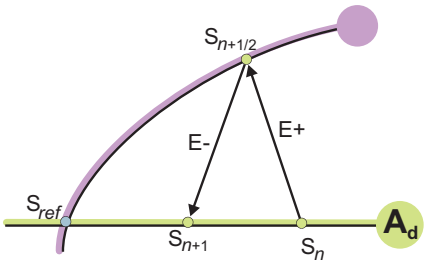


FIGURE 3 – Schéma d'une itération de méthode Latin

## 3.2 Stratégie multiparamétrique

La stratégie multiparamétrique [1] [2] est basée sur le fait que l'algorithme *LATIN* peut être initialisé à partir de n'importe quelle solution vérifiant les conditions d'admissibilité. Dans le cas d'une étude paramétrique, pour un jeu de paramètres donné, on réinitialise la boucle *LATIN* avec la solution convergée correspondant à un autre jeu de paramètres (figure 4). Lorsqu'un paramètre évolue légèrement, la solution globale du problème change peu, ainsi par la stratégie multiparamétrique, la convergence est atteinte plus rapidement en un nombre réduit d'itérations.

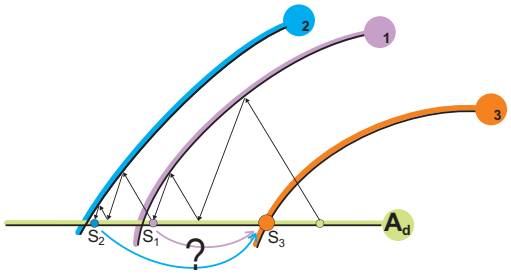


FIGURE 4 – Réinitialisation de la *LATIN*

Les paramètres pris en compte ici seront typiquement des efforts/déplacements imposés, les paramètres de frottement, les jeux, les pré-charges/serrages, soit tous les paramètres qui décrivent le comportement non-linéaire de l'interface. Le point essentiel qui permet d'obtenir une stratégie efficace est que dans le cas particulier de sous-structures élastiques, les interfaces jouent un rôle fondamental. Elles permettent en effet d'initialiser le calcul sur le problème associé à un nouveau jeu de paramètres sans avoir à sauvegarder toutes les informations sur les sous-structures. De plus, les informations sur les interfaces issues du calcul de référence, permettent d'initialiser la méthode avec une première approximation à fort contenu mécanique bien adaptée au problème à résoudre. Ainsi, si la solution du problème de référence n'est pas trop éloignée de la solution du nouveau problème cible, on peut espérer trouver cette dernière à

moindre coût. Parmi les différentes stratégies de réinitialisation étudiées [15] on choisit dans cette étude d'initialiser chaque nouveau calcul à partir de la solution convergée obtenue pour le jeu de paramètres le plus proche au sens d'une distance dans l'espace des paramètres.

# 4 Optimisation structurale d'un assemblage en contact frottant

Les performances de la stratégie multiparamétrique dans le contexte de l'optimisation multi-niveaux de modèle sont illustrées sur un exemple technologique d'assemblage en contact frottant au niveau des liaisons pignon/arbre réalisées par frette. Une frette (figure 5) est un composant technologique composé d'une bague biconique intérieure ajustée sur le pignon, de deux bagues coniques extérieures dont l'une est taraudée et entre lesquelles on exerce un effort via une série de vis distribuées suivant une couronne.

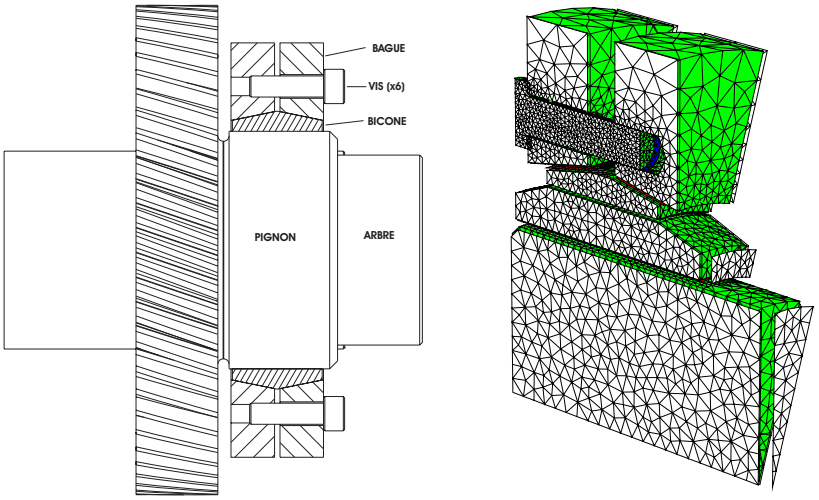


FIGURE 5 – Assemblage par frette

L'optimisation porte ici sur les paramètres de liaison soit les jeux, les serrages et les coefficients de frottement entre les différents éléments de l'assemblage. Les fonctions objectifs étudiées sont le couple et l'effort axial transmissibles entre le pignon (ou roue) et l'arbre. Les contraintes de bornes entre les paramètres sont données dans le tableau 1.

TABLE 1 – Bornes sur les paramètres de conception

| paramètre | min | max |
|---|---|---|
| coefficients de frottement $f_{arbre/roue}$ , $f_{roue/cone}$ , $f_{cone/bague}$ | 0,05 | 0,5 |
| serrage vis $F_{vis}$ $(N)$ | 1500 | 2500 |
| jeu arbre/roue $j_{arbre/roue}$ $(\mu m)$ | 0 | 49 |
| jeu roue/cone $j_{roue/cone}$ $(\mu m)$ | 36 | 106 |

On maximise l'effort axial transmissible entre l'arbre et le pignon via une stratégie d'optimisation dite *multi-niveaux* qui consiste dans un premier temps à réaliser une optimisation globale sur le métamodèle et dans un deuxième temps de réaliser, à partir de ce premier optimum, une optimisation locale sur le modèle EF complet avec une stratégie multiparamétrique.

Différents tirages de 8, 32 et 64 jeux de paramètres (notés $n_{pts}$) suivant un plan d'expériences orthogonal combinant 6 facteurs à deux niveaux sont utilisés lors de la première phase de construction du métamodèle. Le tableau 2 regroupe les temps de calcul et nombre d'itérations réalisés lors de cette première phase.

On réalise dans un premier temps une optimisation globale à moindre coût à partir du métamodèle via un algorithme génétique (fonction *ga* de matlab) qui génère $n_{eval\_ga}$ évaluations de la fonction objectif. Cette première étape permet de localiser la vallée où se situe l'optimum global de la fonction objectif. On réalise ensuite une optimisation locale sur le modèle mécanique complet qui génère $n_{eval\_gradient}$

TABLE 2 – construction du métamodèle

| nombre tirages $n_{pts}$ | nombre d'itérations | tps CPU par jeu de paramètres ($s$) | | | tps CPU total ($s$) |
|---|---|---|---|---|---|
| | | maximal $t_{max}$ | minimal | moyen $t_{moy}$ | $t_{gen\_meta}$ |
| 8 | 1036 | 91 | 60 | 71 | 568 |
| 32 | 2491 | 91 | 3 | 45 | 1426 |
| 64 | 1856 | 95 | 3 | 17 | 1077 |

évaluations de la fonction objectif. Les optima de la fonction objectif (effort axial transmissible par la liaison) lors des deux phases d'optimisation ainsi que les valeurs finales des paramètres détaillés dans le tableau 3 montrent qu'un métamodèle plus riche permet de s'approcher de l'optimum global.

TABLE 3 – Optima sur l'effort axial transmissible par la liaison $P_{ax}$

| | Optimum $P_{ax}$ ($N$) | | Paramètres | | | | | |
|---|---|---|---|---|---|---|---|---|
| $n_{pts}$ | phase 1 | phase 2 | $f_{arbre/roue}$ | $f_{roue/cone}$ | $f_{cone/bague}$ | $F_{vis}$ ($kN$) | $j_{arbre/roue}$ | $j_{roue/cone}$ |
| 8 | 99315 | 103769 | 0,500 | 0,050 | 0,050 | 2500 | 0,050 | 0,068 |
| 32 | 100530 | 103624 | 0,500 | 0,489 | 0,056 | 2500 | 0,050 | 0,106 |
| 64 | 100760 | 103483 | 0,499 | 0,050 | 0,053 | 2500 | 0,050 | 0,103 |

On définit le temps total d'optimisation $t_{multiniv+multipar}$ associé à la stratégie d'optimisation multi-niveaux avec stratégie multiparamétrique (8) comme la somme du temps :

- de génération du métamodèle avec stratégie multiparamétrique $t_{gen\_meta}$,
- du temps d'optimisation sur le métamodèle $t_{opt\_meta}$,
- du temps d'optimisation sur le modèle EF complet avec la Latin multiparamétrique $t_{opt\_EF\_multipar}$.

$$t_{multiniv+multipar} = t_{gen\_meta} + t_{opt\_meta} + t_{opt\_EF\_multipar} \tag{8}$$

Les temps CPU et nombre d'itérations nécessaires à la conduite de ces deux phases d'optimisation sont regroupés dans le tableau 4.

TABLE 4 – Optimisation globale sur métamodèle et locale sur modèle complet

| nombre tirages $n_{pts}$ | Optimisation sur métamodèle | | | Optimisation sur modèle EF | | tps CPU total ($s$) |
|---|---|---|---|---|---|---|
| | évaluation fonction $n_{eval\_ga}$ | tps CPU ($s$) | | évaluation fonction $n_{eval\_gradient}$ | tps CPU ($s$) $t_{opt\_EF\_multipar}$ | $t_{multiniv+multipar}$ |
| | | generation $t_{gen\_meta}$ | optimisation $t_{opt\_meta}$ | | | |
| 8 | 1080 | 568 | 0,7 | 237 | 1204 | 1773 |
| 32 | 1240 | 1426 | 1,2 | 269 | 1385 | 2812 |
| 64 | 1080 | 1077 | 1,3 | 193 | 1240 | 2318 |

Nous comparons cette méthode d'optimisation multi-niveaux basée sur une stratégie de calculs multiparamétrique avec la même méthode d'optimisation mais sans l'aspect multiparamétrique, puis avec une méthode d'optimisation globale dite directe sur le modèle EF complet avec et sans stratégie multiparamétrique (Tableau 5). Pour celà, le temps CPU correspondant au premier calcul est pris comme référence, et on considère que le temps nécessaire pour chaque variation de paramètre est identique au temps du premier calcul.

On définit ainsi le temps total d'optimisation $t_{multiniv}$ associé à la stratégie d'optimisation multi-niveaux sans stratégie multiparamétrique (9) comme la somme du temps :

– de génération du métamodèle sans stratégie multiparamétrique correspondant au produit du nombre de jeu de paramètres destiné à la construction du métamodèle $n_{pts}$ par le temps CPU associé au premier calcul $t_{max}$,
– d'optimisation sur le métamodèle $t_{opt\_meta}$ ,
– d'optimisation sur le modèle EF complet sans stratégie multiparamétrique correspondant au nombre d'évaluations de la fonction objectif $n_{eval\_gradient}$ par l'algorithme d'optimisation locale multiplié par le temps CPU associé au premier calcul $t_{max}$.

$$t_{multiniv} = n_{pts}.t_{max} + t_{opt\_meta} + n_{eval\_gradient}.t_{max} \tag{9}$$

Dans le cadre d'une optimisation directe sur le modèle EF complet, le nombre d'évaluation de la fonction objectif $n_{eval\_ga}$ est beaucoup plus important que le nombre $n_{pts}$ d'évaluations du modèle EF réalisées pour la construction du métamodèle. De plus, le temps CPU moyen par jeu de paramètres $t_{moy}$ converge vers une valeur minimale lorsque le nombre de tirages augmente. On approxime ce minimum par la valeur prise lors du plan d'expériences en 64 essais soit $t_{moy}(n_{pts} = 64)$. Le temps total d'optimisation $t_{direct+multipar}$ associé à la stratégie d'optimisation directe sur le modèle EF complet avec stratégie multiparamétrique (10) est défini comme le produit entre le nombre d'évaluation $n_{eval\_ga}$ de la fonction objectif par l'algorithme génétique lors de la phase d'optimisation globale et la valeur minimale du temps CPU moyen lors des différents appels au solveur EF $t_{moy}(n_{pts} = 64)$.

$$t_{direct+multipar} = n_{eval\_ga}.t_{moy}(n_{pts} = 64) \tag{10}$$

De la même manière on estime le temps total d'optimisation $t_{direct}$ associé à la stratégie d'optimisation directe sur le modèle EF complet sans stratégie multiparamétrique (11) comme le produit entre le nombre d'évaluation $n_{eval\_ga}$ de la fonction objectif par l'algorithme génétique lors de la phase d'optimisation globale et le temps CPU associé au premier calcul du plan d'expériences en 64 essais $t_{max}(n_{pts} = 64)$.

$$t_{direct} = n_{eval\_ga}.t_{max}(n_{pts} = 64) \tag{11}$$

TABLE 5 – Gain avec la méthode multi-niveaux couplée à une stratégie multiparamétrique

| nombre tirage $n_{pts}$ | Optimisation multi-niveaux sans multiparamétrique | | Optimisation directe multiparamétrique | | Optimisation directe sans multiparamétrique | |
|---|---|---|---|---|---|---|
| | temps CPU ($s$) $t_{multiniv}$ | gain 1 $\frac{t_{multiniv}}{t_{multiniv+multipar}}$ | temps CPU ($s$) $t_{direct+multipar}$ | gain 2 $\frac{t_{direct+multipar}}{t_{multiniv+multipar}}$ | temps CPU ($s$) $t_{direct}$ | gain 3 $\frac{t_{direct}}{t_{multiniv+multipar}}$ |
| 8 | 22389 | 12,6 | 18176 | 10,3 | 102665 | 57,9 |
| 32 | 27669 | 9,8 | 18176 | 6,5 | 102665 | 36,5 |
| 64 | 24432 | 10,5 | 18176 | 7,8 | 102665 | 44,3 |

Le *gain*1 défini comme le ratio entre le temps de calcul de la stratégie multi-niveaux sans multiparamétrique $t_{multiniv}$ et du temps de calcul de notre stratégie multi-niveaux couplée au multiparamétrique $t_{multiniv+multipar}$ montre que dans une stratégie multi-niveaux le couplage de la stratégie multiparamétrique divise le temps CPU par un facteur de l'ordre de 10. Dans le cas d'une optimisation directe sur le modèle EF complet, le *gain*2 défini comme le ratio entre le temps de calculs de la stratégie directe avec multiparamétrique et du temps de calcul de notre stratégie multi-niveaux couplée au multiparamétrique montre que la stratégie multi-niveaux divise dans cet exemple le temps CPU par un facteur compris entre 6 et 11. Le *gain*3 défini comme le ratio entre le temps de calcul de la stratégie directe sans multiparamétrique et du temps de calcul de notre stratégie multi-niveaux couplée au multiparamétrique montre que notre méthode divise dans cet exemple le temps CPU par un facteur de l'ordre de 40.

# 5 Conclusion

L'originalité de ces travaux basée sur le couplage d'une stratégie d'optimisation multi-niveaux à une stratégie multiparamétrique montre que les gains en temps CPU peuvent être conséquents voire redoutables dans les démarches d'optimisation de problèmes non-linéaires tels que les assemblages en contact frottant. La diminution du temps CPU, véritable verrou des stratégies d'optimisation structurale, d'un facteur compris entre 40 et 50 sur cet exemple, doit permettre de traiter des problèmes de plus grande taille en terme de nombre de paramètres. L'étude de l'influence de la taille et du type de l'échantillon nécessaire à la construction du métamodèle doit permettre d'accroître ce gain. Les temps de calcul doivent pouvoir encore être diminués d'une part par un couplage plus fort entre les deux phases d'optimisation avec remise à jour du métamodèle, et d'autre part par une recherche intelligente du meilleur point de réinitialisation.

# 6 Remerciements

Ces travaux ont été effectués dans le cadre du projet ANR-08-COSI-007-10, OMD2 financé par l'Agence Nationale de la Recherche.

## Références

[1] P.A. Boucard, L. Champaney. *A suitable computational strategy for the parametric analysis of problems with multiple contact* International Journal for Numerical Methods in Engineering, 57,1259-1282, 2003.

[2] P.A. Boucard, L. Champaney. *Approche multirsolution pour l'étude paramétrique d'assemblages par contact et frottement*, Revue Européenne des Eléments Finis, 13(5/7), 437-448, 2004.

[3] P.A. Boucard, S. Buytet, B. Soulier, P. Chandrashekarappa, R. Duvigneau. *Multidisciplinary Design Optimization in Computational Mechanics . Section Multilevel Modeling*, Publisher Hermes, 199-260, 2010.

[4] M.E.M. El-Sayed, C.K.Hsiung. *Optimum structural design with parallel finite element analysis*, Computers & Structures, 40, 1469-1474, 1991.

[5] H. Engels, W. Becker, A. Morris. *Implementation of a multi-level optimisation methodology within the e-design of a blended wing body*, Aerospace Science and technology, 8, 145-153, 2004.

[6] R.A. Gingold, J.J. Monaghan. *Smooth particle hydrodynamics : theory and application to non-spherical stars*, Monthly Notices Roy. Astronm. Soc., 181, 375-189, 1977.

[7] D.E. Goldberg. *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley, 1989.

[8] S. Kravanja, A. Sorsak, Z.Kravanja. *Efficient multilevel MINLP strategies for solving large combinatorial problems in engineering*, Optimization and engineering, 4, 97-151, 2003.

[9] P. Ladevèze. *Nonlinear Computational Structural Mechanics - New Approaches and non-Incremental Methods of Calculation*, Springer Verlag, 1999.

[10] B. Liu, R.T. Haftka, L.T. Watson. *Global-local structural optimization using response surfaces of local optimization margins*, Structural and Multidisciplinary Optimization, 27, 352-359, 2004.

[11] B. Nayrolles, G. Touzot, P. Villon. *Generalizing the Finite Element Method : Diffuse approximation and diffuse elements*, Computational Mechanics, 10, 307-318, 1992.

[12] G.M. Robinson, A.J. Keane. *A case for multi-level optimisation in aeronautical design*, Aeronautical Journal, 103, 481-485, 1999.

[13] W.J. Roux, N. Stander, R.T. Haftka. *Response surface approximations for structural optimization*, International Journal for Numerical Methods in Engineering, 42, 517-534, 1998.

[14] B. Soulier, L. Richard, B. Hazet, V. Braibant. *Crashworthiness optimization using a surrogate approach by stochastic response surface*, Recent Advances in Integrated Design and Manufacturing in Mechanical Engineering, Springer, 159-168, 2003.

[15] B. Soulier, P.A. Boucard. *A multiparametric strategy for the large-scale multilevel optimization of structural assemblies*, 8th World Congress on Structural and Multidisciplinary Optimization - WCSMO-8, 2009.

[16] P.K. Umesha, M.T. Venuraju, D. Hartmann, K.R. Leimbach. *Optimal design of truss structures using parallel computing*, Structural and Multidisciplinary Optimization, 29, 285-297, 2005.