

Résolutions performantes de problèmes industriels de grande taille en électromagnétisme sur machines parallèles.

Emeric MARTIN

CERFACS, 42 Avenue G. Coriolis, 31057 Toulouse Cedex

Iain S. DUFF

CERFACS

Luc GIRAUD

CERFACS

Julien LANGOU

CERFACS

1 Introduction

La modélisation numérique de situations physiques aboutit souvent à la résolution d'une série d'équations dont les inconnues représentent des grandeurs physiques que l'on souhaite déterminer. On cherche donc à résoudre un système de n équations linéaires regroupées sous la forme:

$$A \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix}$$

avec A la matrice des coefficients de taille $n \times n$, x_i les inconnues qui constituent le vecteur solution x et enfin b_i les valeurs connues qui constituent le second membre b .

Le problème s'écrit finalement:

$$\text{Trouver } x \in \mathbb{R}^n \text{ tel que } Ax = b. \quad (1)$$

Pour cela on peut utiliser une méthode directe qui cherche à décomposer A en un produit de matrices de structures simples, facilement inversibles pour nous donner la solution à moindre coût. Cependant dès que la taille de la matrice devient importante, ce procédé devient très cher en mémoire et en temps de calcul. Pour passer outre, on utilise une méthode itérative, en l'occurrence une méthode de type Krylov qui va construire une suite de sous-espaces emboîtés de taille croissante qui vont contenir à la limite la solution cherchée. La méthode choisie est celle du GMRES (cf [8] pour la théorie et [4] pour l'implantation), les sous-espaces ont la forme suivante:

$$\mathbb{K}_m(A, v) = \langle v, Av, \dots, A^{m-1}v \rangle$$

avec $v \in \mathbb{R}^n$ et $m \leq n$.

Le but de cette méthode est de minimiser, sur chaque sous espace, la norme de la quantité suivante appelée résidu: $r = b - Ax$. A l'itération m , et donc sur le m -ième sous-espace, le problème devient maintenant le suivant:

$$\begin{aligned} & \text{Trouver } x_m \in \mathbb{K}_m \text{ tel que:} \\ & \min_{x \in \mathbb{K}_m} \|Ax - b\| = \|Ax_m - b\|. \end{aligned}$$

Comme les espaces sont emboîtés et de taille croissante, le résidu ne peut que décroître au sens large au fur et à mesure de la construction des espaces:

$$\begin{aligned} \mathbb{K}_{m-1} &\subseteq \mathbb{K}_m \\ \min_{x \in \mathbb{K}_{m-1}} \|b - Ax\| &\geq \min_{x \in \mathbb{K}_m} \|b - Ax\| \\ \text{soit } \|r_{m-1}\| &\geq \|r_m\| \end{aligned}$$

Plaçons nous dans le cas où la convergence n'est pas encore atteinte. S'il existe un espace \mathbb{K}_m avec $m < n$ tel que $\mathbb{K}_m = \mathbb{K}_{m-1}$ alors la solution va être obtenue dans \mathbb{K}_m et le procédé itératif prend fin. Sinon on obtient au final $m = n$, soit $K_m = \mathbb{R}^n$, on est donc sûr de trouver x tel que $Ax = b$ (c'est l'intitulé du problème (1) de départ). La convergence est ainsi assurée au pire en n itérations. Plus le nombre d'itérations est grand, plus la norme du résidu baisse et plus on se rapproche de la solution.

En se plaçant dans le cas où la convergence ne s'effectue pas rapidement et où la matrice possède une taille importante, cette méthode peut s'avérer très chère à mettre en oeuvre. En effet il s'agit de stocker en mémoire une base orthonormale de l'espace courant, espace dont la dimension grossit à chaque itération. Comme les ressources en mémoire ne sont pas infinies, un paramètre appelé "restart" est introduit dans GMRES et correspond à la taille maximale autorisée pour un sous-espace. Si la convergence n'est toujours pas atteinte à ce niveau là, on relance le même procédé en réactualisant le point de départ (r_0) par le dernier résidu trouvé ($r_{restart}$). Ce qui contribue à recréer une nouvelle série de sous-espaces $\mathbb{K}_m(A, r_{restart})$.

Afin d'accélérer la convergence de cette méthode itérative on va introduire un préconditionneur: c'est une matrice qui permet de transformer le système (1) en un système équivalent mais ayant de meilleures propriétés de convergence:

$$M_1 Ax = M_1 b. \quad (2)$$

La matrice M_1 doit être à la fois facile à construire, à appliquer et peu coûteuse en mémoire. La motivation de ce travail est partie de la constatation d'un fait: la convergence du GMRES est souvent pénalisée dès que $M_1 A$ possède des valeurs propres proches de zéro. On va donc proposer une correction à ajouter au système (2) à partir de l'information spectrale de $M_1 A$.

2 Présentation de la méthode

Reprenons le système (2). Comme hypothèse on suppose que la matrice $M_1 A$ est diagonalisable, c'est à dire qu'on peut l'écrire sous la forme:

$$M_1 A = V \Lambda V^{-1} \quad (3)$$

avec Λ la matrice diagonale formée par les valeurs propres λ_i , classées par module croissant et V les vecteurs propres à droite. On va chercher à décaler vers 1 les k plus petites valeurs propres, associées aux vecteurs propres V_ε , telles que: $|\lambda_i| \leq \varepsilon$. La proposition suivante a été démontrée dans [1]:

Proposition 1 Soit W tel que $\tilde{A}_c = W^H A M_1 V_\varepsilon$ soit de rang plein, $\tilde{M}_c = M_1 V_\varepsilon \tilde{A}_c^{-1} W^H$ et $\tilde{M} = M_1 + \tilde{M}_c$. Alors $A \tilde{M}$ est similaire à une matrice dont les valeurs propres sont:

$$\begin{cases} \eta_i = \lambda_i & \text{if } |\lambda_i| > \varepsilon, \\ \eta_i = 1 + \lambda_i & \text{if } |\lambda_i| \leq \varepsilon. \end{cases}$$

La matrice \tilde{M}_c est la correction qui va assurer que le nouveau système qu'on va résoudre en fin de compte:

$$\tilde{M}Ax = \tilde{M}b. \quad (4)$$

ne possède plus de valeurs propres dans un voisinage proche de zéro (dans la boule de rayon ε centré en zéro). Les valeurs propres gênantes ont été translatées vers 1 selon l'axe des abscisses. La matrice W est une matrice de taille $n \times k$. Par ce moyen, on cherche juste à récupérer la contribution sur la matrice des coefficients A , des k plus petites valeurs propres afin de s'en affranchir, par construction, à l'aide de \tilde{M}_c . On choisira pour la suite et par simplicité $W = V_\varepsilon$. Cette méthode est implantée dans un code industriel (développé par G. Alléon (EADS-CCR) et G. Sylvand (CERMICS-INRIA), cf [6]) afin de quantifier en terme d'itérations et de temps ses performances.

3 Le contexte de travail

3.1 Aspect physique

Le code sur lequel on va s'appuyer est un code qui cherche à calculer l'écho radar renvoyé par un objet. Le principe consiste à envoyer une onde plane électromagnétique sur cet objet pour récupérer le champ électromagnétique rayonné en retour et en déduire la signature radar associée à cet angle d'incidence. Afin de reconstituer l'image radar complète de l'objet, il va falloir utiliser une série d'angles consécutifs. Pour chaque angle d'incidence, on va devoir résoudre un système linéaire dont seul le second membre va varier d'un angle à l'autre.

3.2 Aspect numérique

La matrice des systèmes envisagés est dense complexe, symétrique mais non-hermitienne. Le coût d'un produit matrice-vecteur est de l'ordre de n^2 avec n le nombre d'inconnues du problème. La méthode multipôle rapide (FMM: Fast Multipole Method [7]) implantée permet d'abaisser ce coût à $n \log(n)$. Ce qui permet d'envisager l'étude de cas de très grandes tailles. Le code est parallèle pour des machines à mémoire distribuée. Le préconditionneur choisi est basé sur la minimisation de la norme de Frobenius où la norme de Frobenius d'une matrice est :

$$\|A\|_F = \left(\sum_{i=1}^n \sum_{j=1}^n |a_{ij}|^2 \right)^{1/2}.$$

On cherche donc à construire M_{Frob} tel que:

$$\min_{M \in S} \|I - MA\|_F = \|I - M_{Frob}A\|_F$$

S étant l'ensemble des matrices de $\mathbb{R}^{n \times n}$ ayant une structure creuse prédéfinie. C'est ce préconditionneur que l'on va corriger ($M_1 = M_{Frob}$). Le calcul des plus petites valeurs propres de la nouvelle matrice ($M_{Frob}A$) sera effectué par le logiciel ARPACK (cf [5]) en séquentiel, dans une phase préparatoire au calcul. Pour calculer l'information spectrale qui nous sera utile, ARPACK a un coût en temps et en nombre de produits matrice-vecteur qu'il faudra prendre en compte au moment de dresser le bilan global.

4 Présentation des résultats.

La méthode présentée précédemment a déjà été testée avec succès sur des problèmes de petites tailles (cf [1]). On souhaiterait maintenant vérifier, par l'intermédiaire de ce code, si c'est toujours le cas sur des problèmes de plus grandes tailles et quelquesoït la géométrie de départ. Quatre maillages d'objets bien distincts vont être étudiés: un avion (23 676 inconnues), une aile percée d'un trou à sa base (communément appelée "Cetaf", 5391 inconnues), une canalisation ondulée (surnommée Cobra, 60 695 inconnues), et enfin une forme d'Amande (104 793 inconnues).

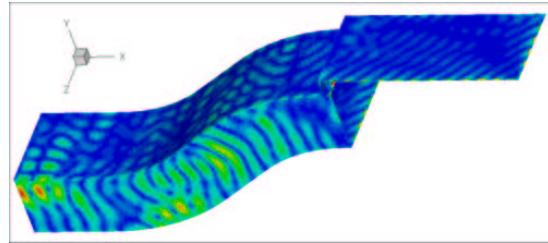
La plateforme de tests est un cluster de SMP (compaq Alpha server). Concernant les paramètres du GMRES, la tolérance est fixée à 10^{-3} , c'est à dire qu'on stoppe la méthode itérative dès que l'on trouve un $x_m \in \mathbb{K}_m$ qui vérifie:

$$\|(\tilde{M}A)x_m - \tilde{M}b\| < \text{tolérance} \cdot \|\tilde{M}b\|. \quad (5)$$

Dans un premier temps, le paramètre de "restart" est fixé à une valeur très grande de telle sorte qu'on reste sur la même suite de sous-espaces \mathbb{K}_m sans chercher à en reconstruire une deuxième si la solution n'a pas été trouvée dans la première (on parle de "restart infini"). On balaye en incidence chaque objet en tenant compte de sa ou ses symétrie(s).



(a) Cas de l'Airbus



(b) Cas du Cobra

Figure 1: Quelques géométries.

4.1 Choix du nombre de valeurs propres.

On s'intéresse au choix optimal du nombre de valeurs propres à décaler. Plutôt que de choisir ce nombre arbitrairement, on va le déterminer expérimentalement. On va se placer sur la plage d'angles d'incidence où le GMRES a le plus besoin d'itérations pour converger; on va se contenter d'une dizaine d'angles consécutifs. Afin que cela soit plus représentatif, on va effectuer pour chaque nouvelle valeur propre traitée une moyenne des nouvelles itérations obtenues sur la plage entière.

Dans la Figure 2 est représenté le nombre moyen d'itérations du GMRES comme une fonction du nombre de valeurs propres décalées. La quantité 0 en abscisse concernant le nombre de valeurs propres, correspond au cas du système préconditionné sans correction. Le nombre 20 semble être le nombre le plus intéressant pour l'Airbus contre 15 pour le Cobra. En effet, au delà on continue à obtenir des gains mais ils sont moins significatifs. On procède de la même façon pour les géométries restantes: sur le Cetaf la méthode itérative se comporte mieux avec 20 valeurs propres traitées et sur l'amande 60. Nous pouvons également observer l'influence sur la vitesse de convergence du nombre de valeurs propres, afin de confirmer ce choix.

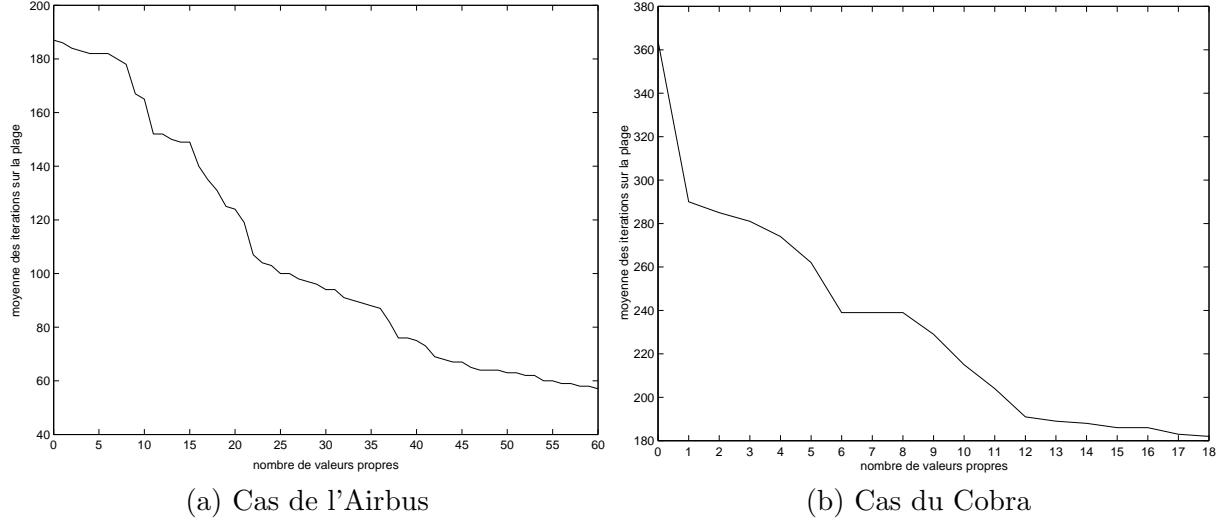


Figure 2: Influence du nombre de valeurs propres traitées sur le nombre d’itérations.

La Figure 3 affiche des historiques de convergence en représentant l’évolution de l’erreur inverse au fur et à mesure des itérations. L’erreur inverse correspond à la quantité suivante:

$$\frac{\|M_1 b - M_1 A x_m\|}{\|M_1 b\|} \quad (6)$$

et la méthode itérative s’arrête dès que l’erreur inverse devient plus petite que la tolérance que l’on s’est fixée (cf (5)). Pour le Cetaf le fait de prendre déjà une valeur au delà de 15 ne semble pas accélérer davantage la convergence, pour l’amande une valeur entre 50 et 60 semble déjà très bien convenir.

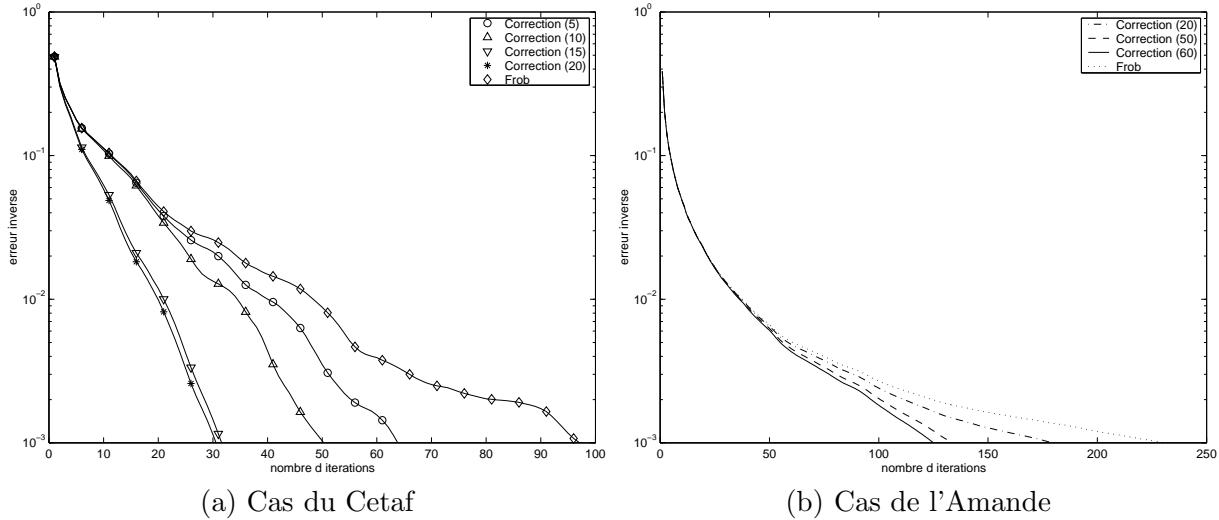


Figure 3: historique de convergence pour différents choix de valeurs propres.

4.2 Résultats sur un balayage complet

Essayons maintenant d'effectuer un balayage complet à partir du système corrigé. Les Figures 4 et 5 présentent le nombre d'itérations nécessaire pour atteindre la convergence à la tolérance prescrite pour chaque angle d'incidence, et ceci de deux façons: en utilisant uniquement le préconditionneur de Frobenius (Frob, en trait plein) d'une part puis en se servant de la mise à jour énoncée dans la proposition (1) d'autre part (Correction, en pointillé). Le nombre choisi de valeurs propres pour chaque géométrie est rappelé dans la légende.

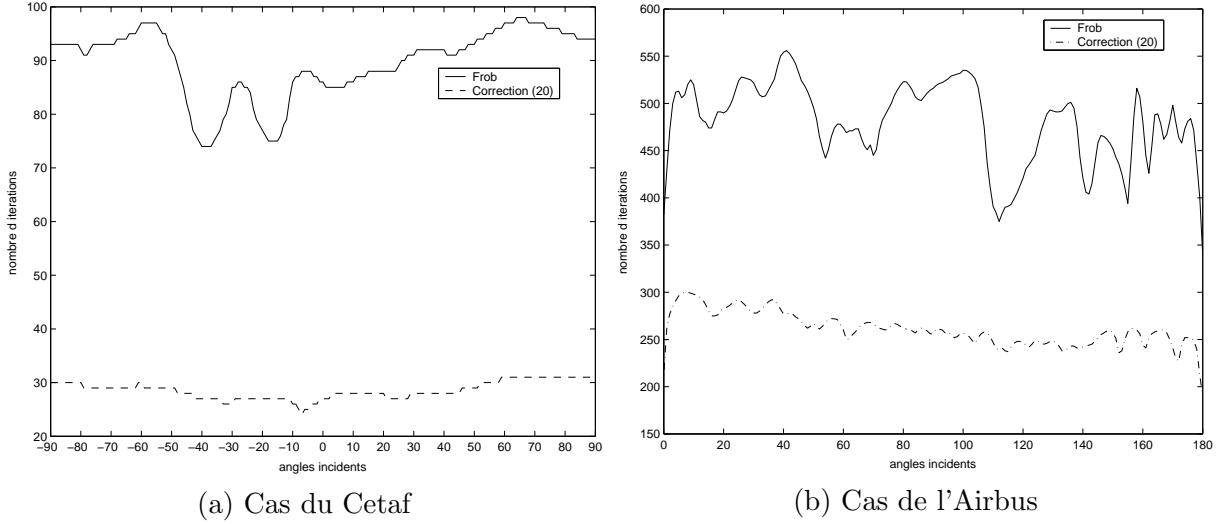


Figure 4: Tolérance 10^{-3} , restart “infini”.

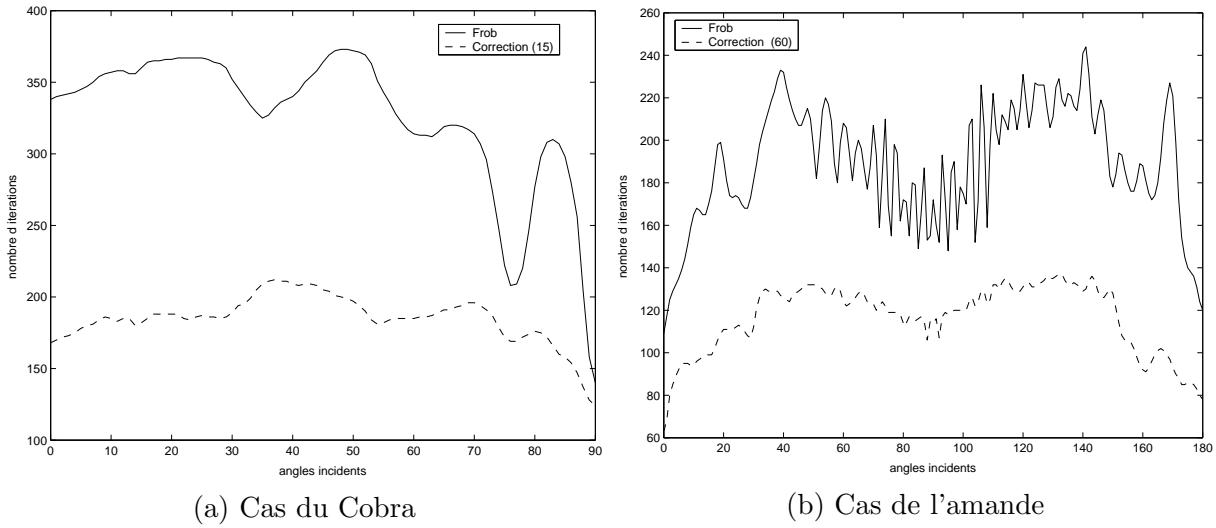


Figure 5: Tolérance 10^{-3} , restart “infini”.

La Table 1 dresse le bilan en nombre de produits matrice-vecteur (noté M.V) et en temps en secondes (noté Tps) sur un balayage complet (180 systèmes) avec et sans correction. Le coût d'ARPACK a été placé entre parenthèses. Les tests ont été réalisés sur 8 et 32 processeurs (Proc.)

Géométrie	Taille	Proc.	Frobenius		Correction	
			M.V	Tps	M.V	Tps
Cetaf	5 391	8	16 391	1h40	(170) 5 349	(43s) 30mn
Airbus	23 676	32	123 639	67h	(1 000) 47 197	(34mn) 24h15
Cobra	60 695	32	29 777	21h10	(1 000) 16 921	(23mn) 10h40
Almond	104 793	8	34 375	40h40	(4 900) 21 273	(3h50) 25h

Table 1: Bilan récapitulatif.

En partant d'une grande tolérance, on gagne ainsi, selon la géométrie, un facteur entre 2 et 3 en temps, et entre 1.5 et 3 en itérations. La mise en oeuvre de cette correction ralentit le produit matrice-vecteur. En effet, on effectue moins d'itérations, mais elles nous coûtent plus cher. Cependant le rapport entre les temps moyens d'un produit avec et sans modifications met en évidence un facteur 1.2, ce qui reste relativement peu coûteux en comparaison des gains obtenus. De plus ces gains en temps peuvent être beaucoup plus intéressants encore en faisant varier le paramètre “restart”.

4.3 Influence du “restart”

On s'intéresse au rôle que peut jouer le paramètre “restart” sur le comportement des itérations du système corrigé, par comparaison avec celles du système non corrigé. Comme on peut le voir dans le tableau suivant, plus ce paramètre est petit, plus l'écart entre le cas modifié (Cor.) et le cas non modifié (Frob) est important en terme d'itérations. Ceci est d'autant plus intéressant, que dans le cadre de la résolution de problèmes de grandes tailles, on se trouve dans l'incapacité matérielle de stocker des espaces de très grandes tailles à cause de leur coût prohibitif en mémoire. On cherche donc à travailler habituellement avec une valeur de “restart” modérée.

Restart	Cetaf		Airbus		Cobra		Almond	
	Frob	Cor.(20)	Frob	Cor.(20)	Frob	Cor.(15)	Frob	Cor.(60)
10	669	37	-	1200	2624	481	-	1497
30	275	31	-	688	1031	232	429	163
50	197	31	-	608	760	207	334	144
∞	98	31	490	283	367	187	232	126

Table 2: Conséquences du “restart” sur les itérations.

La mention “-” signifie que la convergence à la tolérance souhaitée n'a pas été obtenue après 5000 itérations. On constate de suite qu'à faible “restart”, le préconditionneur M_{Frob} pour une précision correcte n'arrive jamais à faire converger le GMRES, alors qu'une fois que les plus petites valeurs propres du système préconditionné ont pu être traitées, il devient possible de converger avec un nombre acceptable d'itérations. Il est également intéressant de noter que le taux de convergence dans ce cas là, s'améliore au fur et à mesure que le restart augmente. Très rapidement la courbe tend à se rapprocher de celle correspondant au restart infini, alors que dans le cas sans correction, ce phénomène tarde à prendre forme. C'est ce qu'illustre la Figure 6, obtenue en déplaçant 20 valeurs propres, et en prenant différentes valeurs pour le “restart” (10, 30, 50, et ∞).

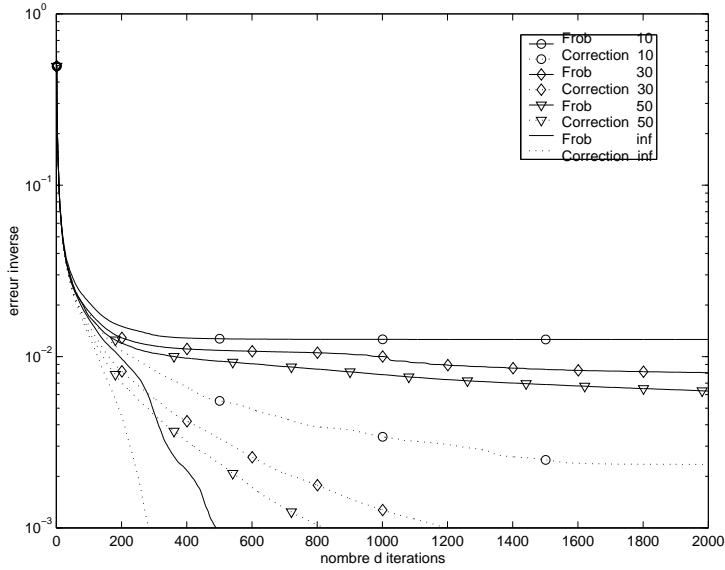


Figure 6: Sensibilité du “restart” sur la vitesse de convergence.

5 Conclusion

La présence de valeurs propres au voisinage de zéro pour la matrice d'un système linéaire ralentit souvent sa résolution. On vient de présenter une méthode originale qui permet de passer outre ces effets en créant un nouveau système équivalent qui ne possède plus les mêmes propriétés spectrales. Le préconditionneur de Frobenius déjà implanté dans le code est performant et intrinsèquement parallèle. Cependant la combinaison avec cette technique de raffinement permet de le rendre encore plus efficace et robuste. Cette dernière a déjà été testée sur d'autres types de préconditionneur sur des géométries réelles industrielles avec succès comme c'est déjà le cas ici. Par la suite nous n'allons plus calculer a priori l'information spectrale à partir d'une boîte à outils comme ARPACK, mais directement l'extraire au fur et à mesure des itérations du GMRES. Ceci évitera le surcoût d'ARPACK qui peut devenir très rapidement lourd quand la taille du problème commence à devenir importante.

L'intégralité des résultats seront présentés dans [9]. On y trouvera également une étude complète et détaillée sur la qualité du préconditionneur de départ, son influence sur le calcul ARPACK et sur le nombre de valeurs propres à déplacer; ainsi que des tests utilisant une autre méthode itérative voisine: le SEED GMRES, qui se sont avérés assez prometteurs.

Références

- [1] B. CARPENTIERI AND I. S. DUFF AND L. GIRAUD, *A class of spectral two-level preconditioners*, Technical Report CERFACS TR/PA/02/55, 2002.
- [2] B. CARPENTIERI, *Sparse preconditioners for dense complex linear systems in electromagnetic applications.*, Thèse INPT, 2002.
- [3] G. ALLÉON AND M. BENZI AND L. GIRAUD, *Sparse Approximate Inverse Preconditioning.*, Numerical Algorithms, Vol. 16, pp 1-15, 1997.

- [4] V. FRAYSSÉ AND L. GIRAUD AND S. GRATTON AND J. LANGOU, *A set of GMRES routines for real and complex arithmetics on high performance computers.*, Technical Report CERFACS TR/PA/03/03, 2003.
Logiciel du domaine public: <http://www.cerfacs.fr/algos/Softs/GMRES/index.html>.
- [5] R. B. LEHOUCQ AND D. C. SORENSEN AND C. YANG, *ARPACK User's guide: Solution of large-scale problem with implicitly restarted Arnoldi methods*, SIAM, <http://www.caam.rice.edu/software/ARPACK/>, 1998.
- [6] G. SYLVAND, *La Méthode Multipôle Rapide en Electromagnétisme : Performances, Parallelisation, Applications*, Thèse Ecole Nationale des Ponts et Chaussées, 2002.
- [7] E. DARVE, *The Fast Multipole Method: Numerical Implementation*, JCP, Vol. 160, nber 1, pp 195-240, 2002.
- [8] Y. SAAD AND M. H. SCHULTZ,
GMRES: A Generalized minimal residual algorithm for solving nonsymmetric linear systems., SISSC, Vol. 7, pp 856-869, 1986.
- [9] I. .S. DUFF AND L. GIRAUD AND J. LANGOU AND E. MARTIN,
Using spectral information for accelerating the convergence of GMRES for large electromagnetics calculations Technical Report CERFACS, 2003. En préparation.
http://www.cerfacs.fr/algos/reports/algo_reports_2003.html