

# Couplage implicite/explicite en présence de grands déplacements. Application aux structures tournantes.

T. Chantrait<sup>1</sup>, J. Rannou<sup>2</sup>

<sup>1</sup> Safran Tech, Modelling & Simulation, Rue des Jeunes Bois, Châteaufort, 78114 Magny-Les-Hameaux, FRANCE, teddy.chantrait@safrangroup.com

<sup>2</sup> Onera-MAS, Université Paris-Saclay, 92322 Châtillon, FRANCE, johann.rannou@onera.fr

**Résumé** — Une modification de l’algorithme GC, implémenté dans [1], est proposée ici pour étendre le périmètre d’applicabilité de la stratégie de cosimulation implicite/explicite en présence de grands déplacements. L’algorithme proposé est éprouvé sur une structure tournante académique, assimilable à un disque aubagé impacté. Il est montré que les déplacements obtenus avec cette nouvelle stratégie sont bien corrélés à la solution de référence “tout explicite” et qu’un gain de temps de calcul peut être obtenu sous certaines conditions. Les limitations de la stratégie sont également mises en lumière.

**Mots clés** — cosimulation, implicite, explicite, méthode GC, structure tournante.

## 1 Introduction

Les techniques de couplage implicite/explicite ont très tôt été imaginées comme en témoignent certains travaux de la fin des années 70 tels que ceux de Belytschko [2]. Mais ce n’est qu’au début des années 2000 que les travaux de Gravouil et Combescure [3] permettent véritablement d’élaborer des algorithmes de couplage, dits de co-simulation. Ces derniers permettent d’allier les atouts d’au moins deux codes de calcul, caractérisés par un schéma d’intégration temporelle différent et/ou une discrétisation temporelle différente, pour résoudre un problème de structure.

Les apports essentiels de ces travaux sont d’avoir proposé un cadre d’analyse de tels couplages et d’en avoir dégagé une condition de stabilité pour les intégrateurs en temps de Newmark [4]. L’algorithme initial (appelé Gravouil-Combescure ou GC [5]) a été par la suite décliné en différentes versions et étendu à d’autres intégrateurs en temps [6–9]. C’est cependant cette version initiale qui a été utilisée dans [10]. L’application visée était l’impact basse vitesse et faible énergie d’un panneau composite stratifié. L’intérêt de la co-simulation était de traiter une partie localisée de la zone impactée par un code explicite (en l’occurrence *Europlexus*<sup>1</sup>). La solution dans cette région est non-linéaire et non-régulière, un solveur explicite y est donc tout à fait adapté. Le reste de la structure est traité avec un code implicite (*Z-set*<sup>2</sup>) qui a des pas de temps beaucoup plus grands que le pas de temps critique du solveur explicite. Un rapport de pas de temps de plus de 1000 est utilisé ce qui a permis un gain de temps CPU sur cette application. De plus, il a été montré que la frontière entre ces deux zones peut se déplacer en fonction de l’évolution des endommagements par un mécanisme appelé *bascule*. Ce mécanisme de bascule permet d’augmenter le gain en temps CPU en optimisant la taille de la zone explicite durant la résolution. Néanmoins, une restriction faite dans ces travaux est que le problème implicite doit être élastique linéaire.

Il s’agit ici, d’évaluer la possibilité d’extension de la stratégie de couplage développée dans [1] à des structures tournantes. Au terme des développements proposés, une simulation d’un impact sur aube *fan* ou d’une perte d’aube de turobréacteur est envisagée, sachant qu’une partie au moins du système est en rotation. Il est alors proposé qu’une partie de l’aube et de l’impacteur soit traitée dans un code explicite et que le reste de la structure, c’est à dire la partie complémentaire, soit traitée dans un code implicite. A la différence de ce qui a déjà été fait, la difficulté ici, est liée à la rotation de composants qui doit être intégrée au modèle et qui, même en l’absence de non-linéarités matériaux, implique une non-linéarité géométrique qu’il faut traiter dans la partie implicite (partie complémentaire de l’aube et de l’impacteur).

1. *Europlexus* est un code de dynamique explicite développé par le Commissariat à l’énergie atomique et aux énergies alternatives (CEA) en France et le Joint Research Centre (JRC) à Ispra en Italie.

2. *Z-set* est un code aux éléments finis développé par les Mines ParisTech, l’Onera et Safran Tech, <http://zset-software.com>.

Un algorithme intégrant les grands déplacements dans la partie implicite, et sous certaines hypothèses, est ainsi mis en place puis éprouvé sur une structure tournante académique assimilable à un disque aubagé.

## 2 Algorithme GC adapté à la prise en compte de grands déplacements

Pour mettre en place l'algorithme proposé, il est supposé que la discrétisation temporelle du problème implicite soit choisie de telle sorte qu'une seule itération de Newton soit nécessaire à chaque incrément implicite. Ceci peut paraître restrictif à ce stade, mais il est espéré que les pas de temps implicite respectant ces conditions, soient tout de même significativement plus grands que le pas de temps critique du solveur explicite qui est lui, gouverné par la condition de stabilité du schéma en temps explicite. De plus, dans les applications traitées dans [1], le coût de la résolution implicite était négligeable devant celui de la résolution explicite. Donc même si le coût de la résolution implicite devait être plus important du fait de cette contrainte, et qu'il reste tout de même petit devant celui de la résolution explicite, cette hypothèse peut s'avérer acceptable vis-à-vis du temps de calcul.

La prise en compte de grands déplacements implique que l'opérateur de raideur de la partie implicite n'est plus constant contrairement à ce qui a été fait dans les travaux précédents. Il en résulte que le complément de Schur du domaine implicite ne l'est plus non plus. Or cet opérateur est utilisé dans l'algorithme de couplage pour résoudre le problème d'interface entre les deux domaines. La mise en œuvre de l'algorithme de couplage est ainsi repensée en conséquence.

Les algorithmes 1 et 2 représentent formellement le nouveau processus de résolution proposé dans chacun des deux codes entre les instants  $t_n$  et  $t_n + dt_{imp} = t_n + m dt_{exp}$  avec  $dt_{imp}$  le pas de temps utilisé dans le code implicite et  $dt_{exp}$  celui utilisé dans le code explicite et  $m$  le rapport entre les deux.

Par rapport à l'algorithme classique, il faut noter ici que l'opérateur global implicite linéarisé et son complément de Schur sont recalculés à chaque début d'incrément. Par contre ces opérateurs sont strictement les mêmes entre la résolution du problème sans liaison (PBSL) et la résolution du problème avec liaison (PBAL). Les prédicteurs utilisés dans le calcul des seconds membres de la résolution du PBSL et du PBAL sont également les mêmes.

---

### Algorithme 1 : Algorithme de résolution d'un incrément implicite

---

**Données :** conditions initiales et état du sous-domaine en début d'incrément à  $t = t_n$

**début**

```

calcul des prédicteurs de Newmark
calcul de l'opérateur linéarisé
calcul du complément de Schur de l'opérateur tangent
→ envoi du complément de Schur de l'opérateur tangent au code explicite
force à l'interface nulle
résolution du système linéaire "sans liaison" (PBSL)
détermination de la vitesse interfaciale "sans liaison" à  $t = t_n + dt_{imp}$ 
→ envoi de la vitesse interfaciale "sans liaison" à  $t = t_n + dt_{imp}$  au code explicite
← réception en provenance du code explicite de l'effort d'interface à  $t = t_n + dt_{imp}$ 
résolution du système linéaire "avec liaison" (PBAL)
calcul du résidu
si critère sur le résidu d'équilibre validé alors
| continuer
sinon
| cas non géré → fin du calcul
fin
calcul du nouvel état du sous-domaine
 $t_n \leftarrow t_n + dt_{imp}$ 

```

**fin**

---

---

**Algorithme 2 :** Algorithme de résolution de  $m$  incréments explicites (entre  $t_n$  et  $t_n + dt_{imp} = t_n + mdt_{exp}$ )

---

**Données :** conditions initiales et état du sous-domaine en début d'incrément à  $t = t_n$

**début**

**pour**  $j \leftarrow 1$  à  $m$  **faire**

    effort de liaison à l'interface nul

    mise à jour des quantités cinématiques

**si**  $j=1$  **alors**

$\leftarrow$  réception en provenance du code implicite de la vitesse interfaciale "sans liaison"

      à  $t = t_n + mdt_{exp}$

$\leftarrow$  réception en provenance du code implicite du complément de Schur à

$t = t_n + mdt_{exp}$

**fin**

    calcul du saut interfacial de vitesse "sans liaison" à  $t = t_n + jdt_{exp}$

    calcul de l'effort de liaison à l'interface

**si**  $j=m$  **alors**

$\rightarrow$  envoi de la force de liaison à l'interface à  $t = t_n + mdt_{exp}$

**fin**

    mise à jour de l'état du sous-domaine avec le nouvel effort de liaison à l'interface

$t \leftarrow t_n + jdt_{exp}$

**fin**

$t_n \leftarrow t_n + mdt_{exp}$

**fin**

---

### 3 Description du cas test de validation

Le cas test académique visible sur la Figure 1 est utilisé pour la mise en place du couplage implicite/explicite entre *Europlexus* et *Z-set* avec l'algorithme proposé. Il s'agit d'une structure simplifiée assimilable à un disque comportant six aubes. Dans ce cas test, le disque implicite traité dans *Z-set* est mis en rotation autour de l'axe  $z$ . Aucune condition limite n'est appliquée dans le code *Europlexus* où l'impacteur et le secteur indiqué sur la Figure 1 sont traités. Ainsi, il est attendu que le secteur explicite soit entraîné en rotation via les efforts d'interface résultants du couplage implicite/explicite entre ces deux codes. Par conséquent, l'impacteur et le secteur entrent en contact l'un avec l'autre. Le contact apporte ainsi une source de non-linéarité dans le système et la rotation imposée dans le code *Z-set* conduit à se placer dans un formalisme en grand déplacement. Il est à noter que l'interface implicite/explicite comporte uniquement 32 nœuds soit 96 degrés de liberté.

### 4 Résultats

L'évolution du déplacement suivant la direction  $x$  du point de contrôle situé en haut de l'aube près de la zone de contact, et visible sur la Figure 1, est reportée sur la Figure 2 pour différentes simulations. Il y a en particulier la solution issue de la simulation tout explicite (full-explicit) ainsi que celles obtenues par les cosimulations implicite/explicite avec les rapports de pas de temps ( $m$ ) de 10, 100 et 1000. Il peut être observé que l'ensemble des solutions sont bien corrélées entre elles, même s'il ne faut pas chercher à avoir une correspondance exacte entre les solutions couplées et celle issue de la solution de référence. En effet, les schémas explicites et implicites ne convergent pas de la même manière et même si le problème était complètement convergé en temps, la discrétisation de la masse n'est pas la même entre les deux codes. Si dans un code explicite la masse est lumpée, elle ne l'est pas en implicite. Pour avoir une stricte étude de convergence il faudrait donc également faire une convergence en espace, ce qui n'est pas proposé ici.

Les temps de calcul correspondant aux différentes étapes de l'algorithme de couplage pour  $m = 10$ ,  $m = 100$  et  $m = 1000$  sont reportées sur la Figure 3. Il peut être constaté, comme attendu, qu'il y a systématiquement un facteur 10 entre chaque cosimulation sur le temps de résolution implicite puisque

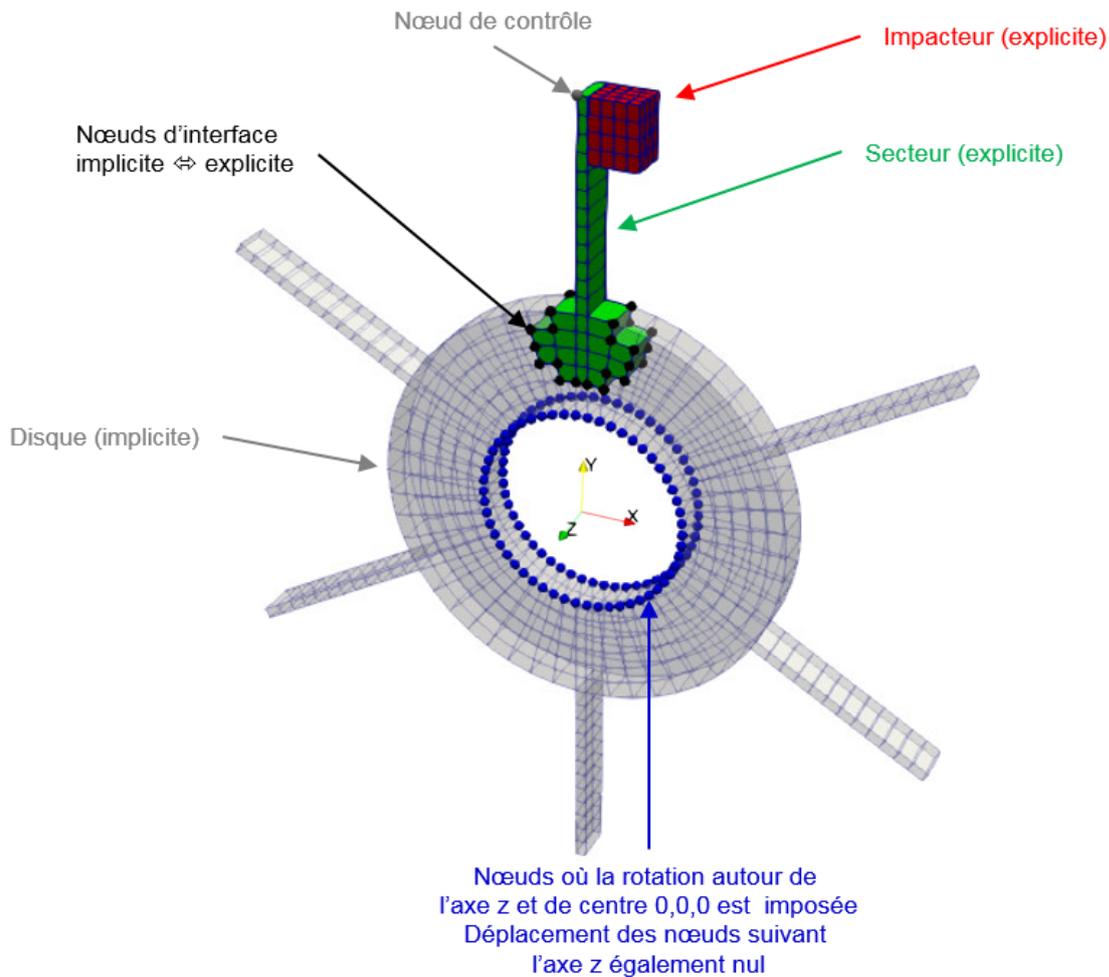


FIGURE 1 – Description du cas test.

cela correspond à la définition du paramètre  $m$  qui évolue avec ce même facteur pour les tests réalisés. Le temps dévolu au calcul explicite est constant et le temps dévolu aux échanges MPI est négligeable. La partie qui peut coûter cher ici est le temps alloué aux calculs des compléments de Schur. Il augmente d'un facteur 10 également car il est refait à chaque début d'incrément. Le problème est qu'il consomme un temps non négligeable puisque cela correspond pour chaque cosimulation à environ 120% du temps de calcul implicite comprenant les résolutions avec et sans liaison.

Ce qui n'apparaît pas ici est la complexité de ces calculs en fonction de la taille des problèmes. Pour l'évaluer, les notations suivantes sont utilisées :

- $N_i$  : le nombre de degrés de liberté du problème implicite
- $N_e$  : le nombre de degrés de liberté du problème explicite
- $N_{i/e}$  : le nombre de degrés de liberté du problème d'interface

À pas de temps explicite et rapport  $m$  fixés, les complexités des différentes opérations peuvent être écrites pour aller de l'instant  $t_n$  à l'instant  $t_n + mdt_{exp}$  :

- résolution du problème implicite :

$$O(N_i^2)$$

- calcul du complément de Schur implicite à partir de la factorisation précédente :

$$O(N_i^2 N_{i/e})$$

- factorisation de l'opérateur d'interface dense :

$$O(N_{i/e}^3)$$

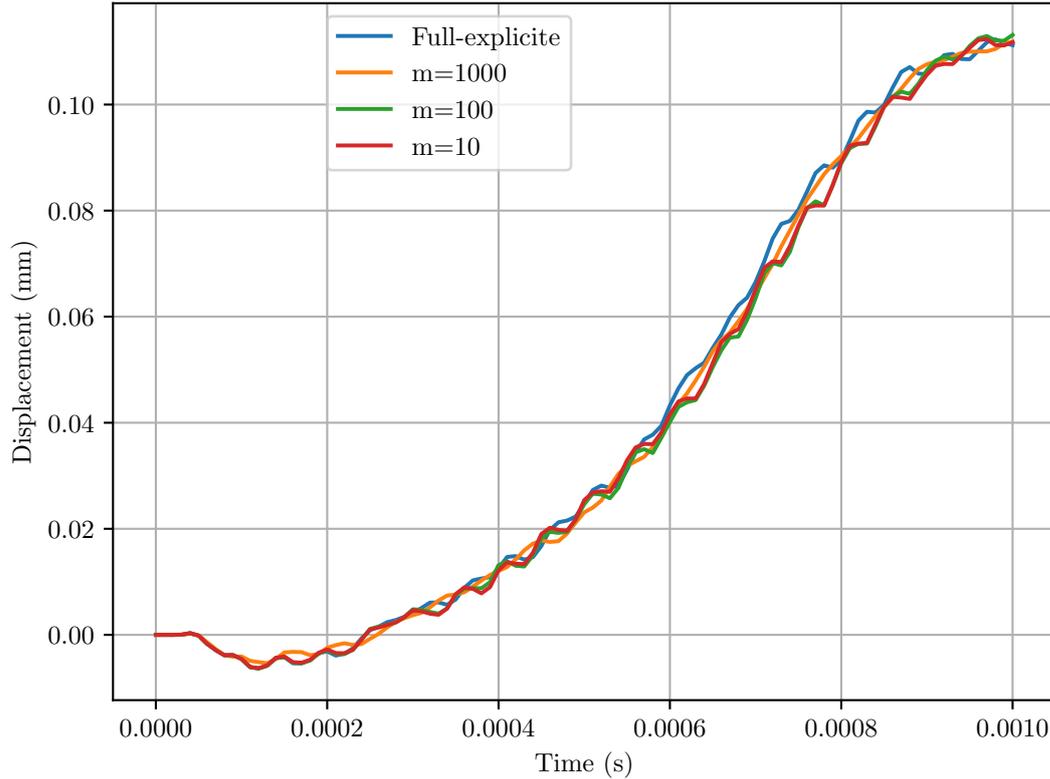


FIGURE 2 – Déplacement du point de contrôle pour différents rapports  $m$  et pour la solution de référence.

— résolution du problème explicite :

$$O(mN_e)$$

Ainsi, si la Figure 3 présente un gain pour une discrétisation spatiale donnée et  $m = 1000$ , il peut être souligné que le calcul du complément de Schur peut potentiellement devenir très cher. Si on raffine la discrétisation et sans y prendre garde, son coût de calcul en  $O(N_i^2 N_{i/e})$  peut potentiellement devenir plus important que celui de la factorisation du problème implicite en  $O(N_i^2)$ . Le mécanisme de bascule qui avait été introduit dans [1] permettrait certes de limiter la valeur de  $N_{i/e}$  mais il faudrait également compter sur la parallélisation du code implicite. Malgré cela, lorsqu'un maillage plus fin est désiré, la multiplication du nombre de cœurs de calcul ne sera pas suffisante pour compenser l'évolution en  $O(N_{i/e}^3)$  de la factorisation du complément de Schur. Une réflexion sur la stratégie de réactualisation du calcul du complément de Schur est donc en cours. Il faut noter par ailleurs que les temps de calcul présentés sur la Figure 3 peuvent très certainement être optimisés, mais cela ne change en rien cette analyse de complexité algorithmique.

## 5 Conclusion et perspectives

L'algorithme GC a été modifié afin d'être utilisé dans un problème non-linéaire en présence de grands déplacements. Un cas test de taille modeste mais représentatif des applications visées a été proposé pour servir de support à des études de convergence en temps. Les résultats obtenus semblent satisfaisants dans la mesure où les calculs couplés fournissent des résultats tout à fait comparables au calcul de référence. Certaines limites peuvent cependant être mises en évidence :

— le problème implicite non-linéaire ne doit pas itérer : chaque incrément doit satisfaire le critère de convergence sur le résidu en une itération. Ceci implique une contrainte sur le pas de temps maximal. Si des itérations devaient être nécessaires, il faudrait utiliser dans *Europlexus* un méca-

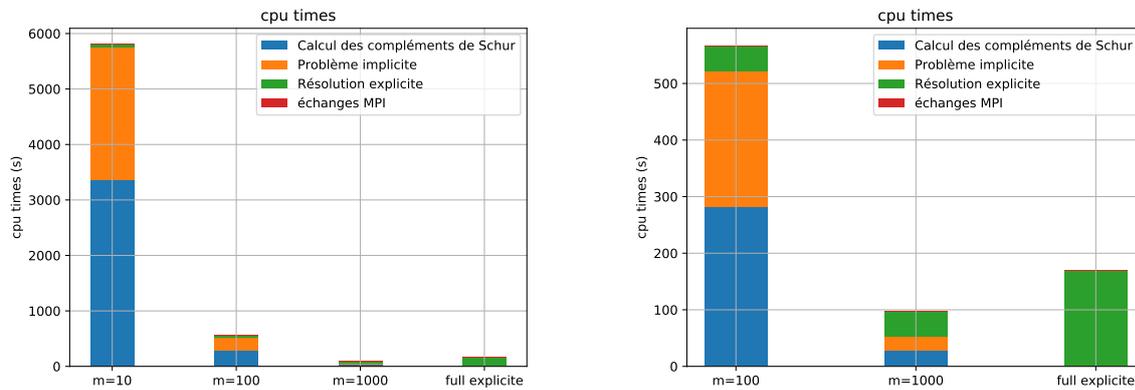


FIGURE 3 – Temps de calcul pour différentes valeurs de  $m$ . La Figure de droite est un zoom.

nisme permettant de reprendre le calcul plusieurs incréments en arrière (de manière similaire à un redémarrage).

- le coût de calcul du complément de Schur implicite et de factorisation de l'opérateur d'interface peut être important car sa complexité algorithmique est élevée. Cela peut être acceptable si les rapports DDL d'interface/DLL total et DDL explicite/DLL implicite sont faibles mais peut devenir très pénalisant autrement.

Ce dernier item constitue un véritable point dur. Des pistes de réflexion pour y remédier sont en cours.

## Références

- [1] Teddy Chantry. *Approche multi-échelle en espace et en temps pour la prévision des endommagements dans les structures composites soumises à un impact de faible énergie*. PhD thesis, Onera / LaMCoS - UMR CNRS 5259 - INSA de Lyon, décembre 2014.
- [2] Ted Belytschko and Robert Mullen. Stability of explicit-implicit mesh partitions in time integration. *International Journal for Numerical Methods in Engineering*, 12(10) :1575–1586, 1978.
- [3] Anthony Gravouil. *Méthode multi-échelles en temps et en espace avec décomposition de domaines pour la dynamique non-linéaire des structures*. PhD thesis, ENS Cachan / LMT, 2000.
- [4] NM Newmark. A method of computation for structural dynamics. In *Proc. ASCE*, volume 85, pages 67–94, 1959.
- [5] A. Combescure and A. Gravouil. A numerical scheme to couple subdomains with different time-steps for predominantly linear transient analysis. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12) :1129–1157, 2002.
- [6] Arun Prakash, Ertugrul Taciroglu, and Keith D. Hjelmstad. Computationally efficient multi-time-step method for partitioned time integration of highly nonlinear structural dynamics. *Computers & Structures*, 133(0) :51–63, 2014.
- [7] N. Mahjoubi. *Méthode générale de couplage de schéma d'intégration multi-échelles en temps en dynamique des structures*. 2010.
- [8] N. Mahjoubi, A. Gravouil, A. Combescure, and N. Greffet. A monolithic energy conserving method to couple heterogeneous time integrators with incompatible time steps in structural dynamics. *Computer Methods in Applied Mechanics and Engineering*, 2010.
- [9] A Gravouil, A Combescure, and M Brun. Heterogeneous asynchronous time integrators for computational structural dynamics. *International Journal for Numerical Methods in Engineering*, 102(3-4) :202–232, 2015.
- [10] Teddy Chantry, Johann Rannou, and Anthony Gravouil. Low intrusive coupling of implicit and explicit time integration schemes for structural dynamics : Application to low energy impacts on composite structures. *Finite Elements in Analysis and Design*, 86 :23–33, 2014.