


Comparaison de solveurs couplés pour la résolution de systèmes linéaires FEM/BEM résultant de la discrétisation de problèmes aéroacoustiques

Conférence francophone d'informatique en Parallélisme, Architecture et Système
COMPAS'21

Emmanuel Agullo, Marek Felšöci, Guillaume Sylvand

du 6 au 9 juillet 2021 à Lyon (en virtuel)

Inria Bordeaux Sud-Ouest
Équipe-projet HiePACS

Solveurs rapides pour l'aéroacoustique haute-fréquence

- thèse co-financée par Airbus et la région Nouvelle-Aquitaine
  - encadrée par Guillaume Sylvand [1] et Emmanuel Agullo [2]
- contexte industriel
  - étude de la propagation des ondes sonores générées par des avions
    - réduction de la pollution sonore, certification d'avions, ...

Figure 1 – **Flux d'air généré par un Airbus A319-112 au décolage (Sebaso, Wikimedia Commons, CC BY-SA 4.0)**

1. Airbus Central R&T / Inria Bordeaux Sud-Ouest
2. Inria Bordeaux Sud-Ouest

Modélisation

modélisation de la surface de l'avion et du flux d'air

modèle physique continu $\rightarrow$ *discrétisation* $\rightarrow$ modèle numérique discret

- couplage de deux méthodes de discrétisation
  - FEM [3] : parties volumiques (flux d'air)
  - BEM [4] : parties surfaciques (surface d'avion et du domaine volumique)

Figure 2 – Exemple d'un modèle discret FEM/BEM. Le maillage rouge résulte de la discrétisation BEM et le maillage vert résulte de la discrétisation FEM.

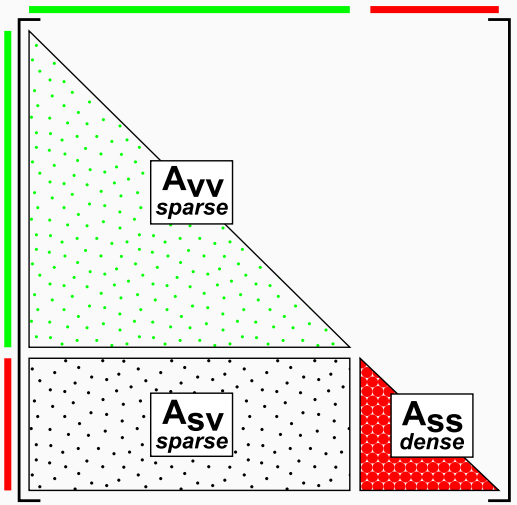
3. méthode des éléments finis (Finite Elements Method)
4. méthode des éléments finis de frontière (Boundary Elements Method)

Couplage FEM/BEM

vers un système linéaire couplé

- matrice de coefficients symétrique composée des parties
  - creuses : discrétisation des parties volumiques avec FEM (A_{vv}), interactions FEM/BEM (A_{vs}, A_{sv})
  - denses : discrétisation des parties surfaciques avec BEM (A_{ss})

$$\begin{bmatrix} A_{vv} & A_{vs} \\ A_{sv} & A_{ss} \end{bmatrix} \times \begin{bmatrix} x_v \\ x_s \end{bmatrix} = \begin{bmatrix} b_v \\ b_s \end{bmatrix}$$



- méthode de résolution directe à l'aide du complément de Schur [4]

Solution directe

réduction du problème aux frontières $\rightarrow$ simplification du système

$$\begin{matrix} R_1 \\ R_2 \end{matrix} \begin{bmatrix} A_{vv} & A_{vs} \\ A_{sv} & A_{ss} \end{bmatrix} \times \begin{bmatrix} x_v \\ x_s \end{bmatrix} = \begin{bmatrix} b_v \\ b_s \end{bmatrix}$$

Principales étapes de résolution

1. éliminer x_v de la seconde équation $\rightarrow$ complément de Schur S

$$\begin{matrix} R_1 \\ R_2 \leftarrow R_2 - A_{sv}A_{vv}^{-1} \times R_1 \end{matrix} \begin{bmatrix} A_{vv} & A_{vs} \\ 0 & \underbrace{A_{ss} - A_{sv}A_{vv}^{-1}A_{vs}}_{S} \end{bmatrix} \times \begin{bmatrix} x_v \\ x_s \end{bmatrix} = \begin{bmatrix} b_v \\ b_s - A_{sv}A_{vv}^{-1}b_v \end{bmatrix}$$

2. résoudre le système réduit avec le complément de Schur $\rightarrow x_s$

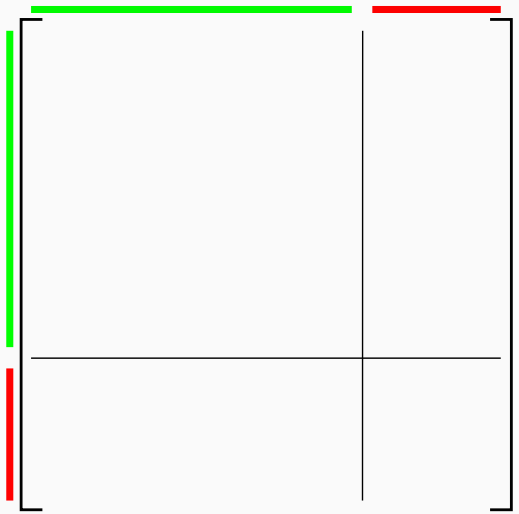
$$\underbrace{(A_{ss} - A_{sv}A_{vv}^{-1}A_{vs})}_{S} x_s = b_s - A_{sv}A_{vv}^{-1}b_v$$

3. calculer $x_v = A_{vv}^{-1}(b_v - A_{vs}x_s)$

Problème

Si tout tient en mémoire...

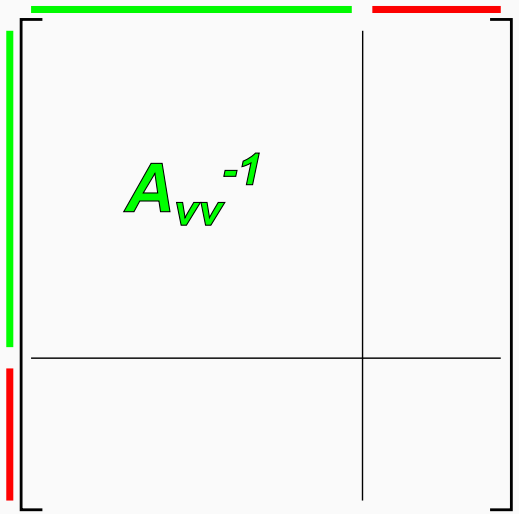
- schéma à deux étapes (two-stage scheme)
  - couplage d'un solveur creux (MUMPS) et dense (SPIDO, HMAT)
- avantage des solveurs bien optimisés de la communauté
- API complément de Schur $\rightarrow S = A_{ss} - A_{sv} A_{vv}^{-1} A_{sv}^T$



Problème

Si tout tient en mémoire...

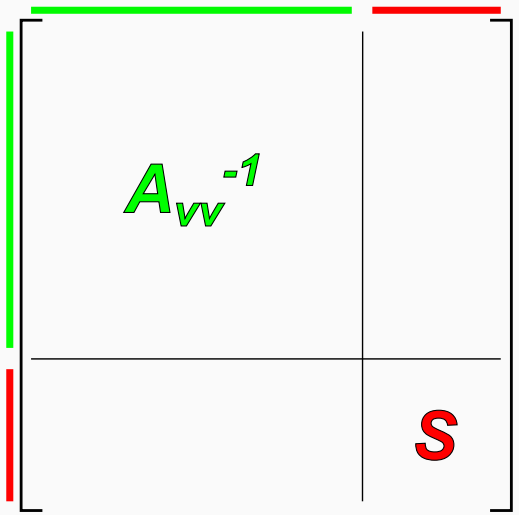
- schéma à deux étapes (two-stage scheme)
  - couplage d'un solveur creux (MUMPS) et dense (SPIDO, HMAT)
- avantage des solveurs bien optimisés de la communauté
- API complément de Schur $\rightarrow S = A_{ss} - A_{sv} A_{vv}^{-1} A_{sv}^T$



Problème

Si tout tient en mémoire...

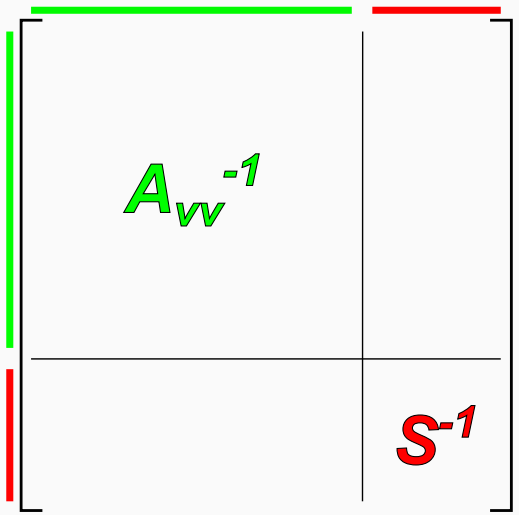
- schéma à deux étapes (two-stage scheme)
  - couplage d'un solveur creux (MUMPS) et dense (SPIDO, HMAT)
- avantage des solveurs bien optimisés de la communauté
- API complément de Schur $\rightarrow S = A_{ss} - A_{sv} A_{vv}^{-1} A_{sv}^T$



Problème

Si tout tient en mémoire...

- schéma à deux étapes (two-stage scheme)
  - couplage d'un solveur creux (MUMPS) et dense (SPIDO, HMAT)
- avantage des solveurs bien optimisés de la communauté
- API complément de Schur $\rightarrow S = A_{ss} - A_{sv} A_{vv}^{-1} A_{sv}^T$



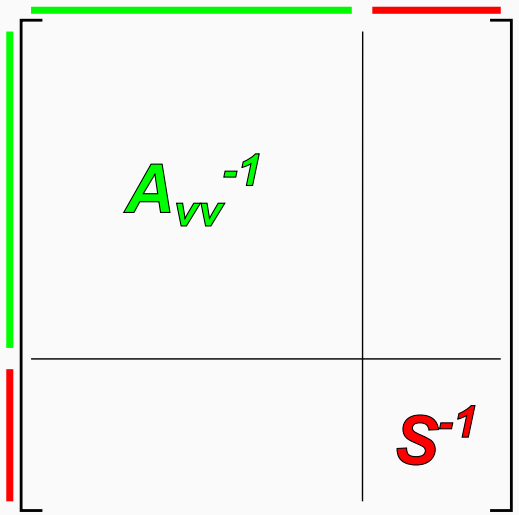
Problème

Si tout tient en mémoire...

- schéma à deux étapes (two-stage scheme)
  - couplage d'un solveur creux (MUMPS) et dense (SPIDO, HMAT)
- avantage des solveurs bien optimisés de la communauté
- API complément de Schur $\rightarrow S = A_{ss} - A_{sv} A_{vv}^{-1} A_{sv}^T$

Compression, systèmes plus grands ?

- stockage de S dense très coûteux en mémoire
  - compression de $S \rightarrow \mathcal{H}$-matrice (HMAT)
- adaptation du schéma à deux étapes
  - continuer à utiliser les solveurs de la communauté

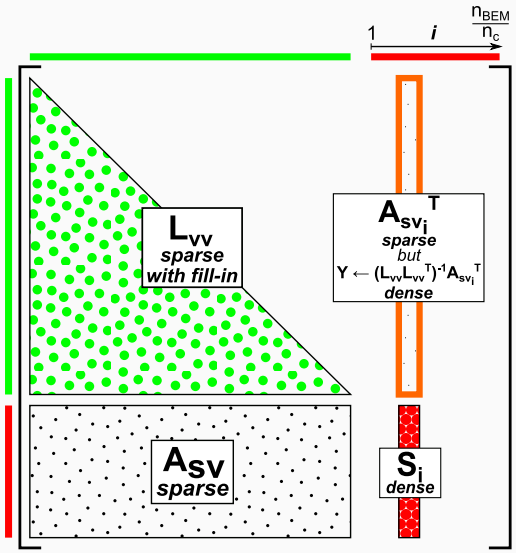


Schémas à deux étapes sans compression

Multi-solve

$$S_i = A_{ss_i} - A_{sv}(L_{vv}L_{vv}^T)^{-1}A_{sv_i}^T$$

- 1 factorisation de la matrice verte (symétrique)
- plusieurs *solve* impliquant les blocs oranges (résultat dense)

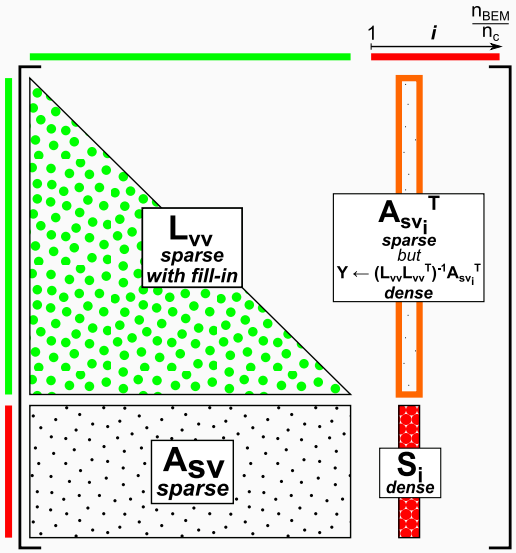


Schémas à deux étapes sans compression

Multi-solve

$$S_i = A_{ss_i} - A_{sv}(L_{vv}L_{vv}^T)^{-1}A_{sv_i}^T$$

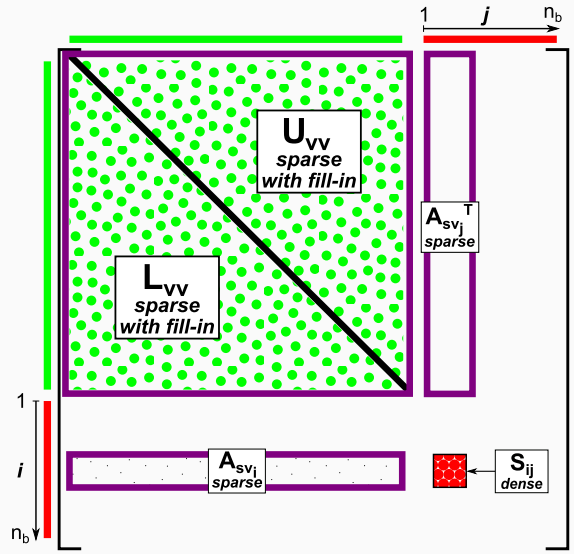
- 1 factorisation de la matrice verte (symétrique)
- plusieurs *solve* impliquant les blocs oranges (résultat dense)



Multi-factorization

$$S_{ij} = A_{ss_{ij}} - A_{sv_i}(L_{vv}U_{vv})^{-1}A_{sv_j}^T$$

- plusieurs factorisations de la matrice violette (non-symétrique)
- calcul des blocs de complément de Schur via l'API



Envrionnement expérimental

Cas test

- tube court (longueur : 2 m, diamètre : 4 m)
- systèmes linéaires suffisamment proches des cas réels
  - exemple reproductible pour la communauté [1]
- maillage volumique v discrétisé avec FEM
- maillage surfacique s discrétisé avec BEM

Configuration

- PlaFRIM, un seul nœud *miriel* (126 Gio de RAM) à la fois
- 24 fils d'exécution OpenMP, MKL et StarPU
- paramètre de précision ϵ fixé à 10^{-3} le cas échéant

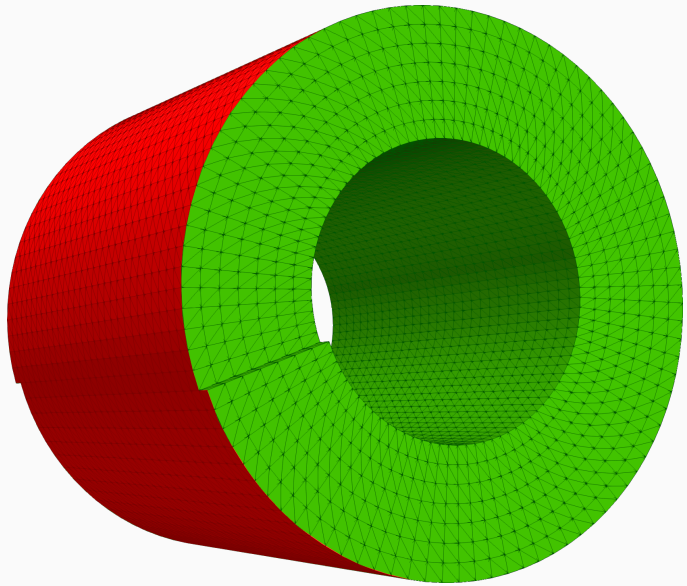


Figure 3 – **Un cas test de tube court à 20,000 inconnues**

Multi-solve vs. multi-factorization sans compression

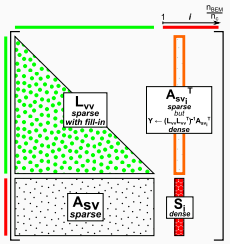


Figure 4 – Multi-solve

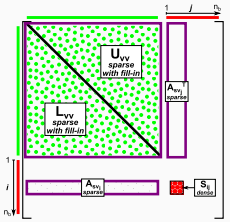


Figure 5 – Multi-factorization

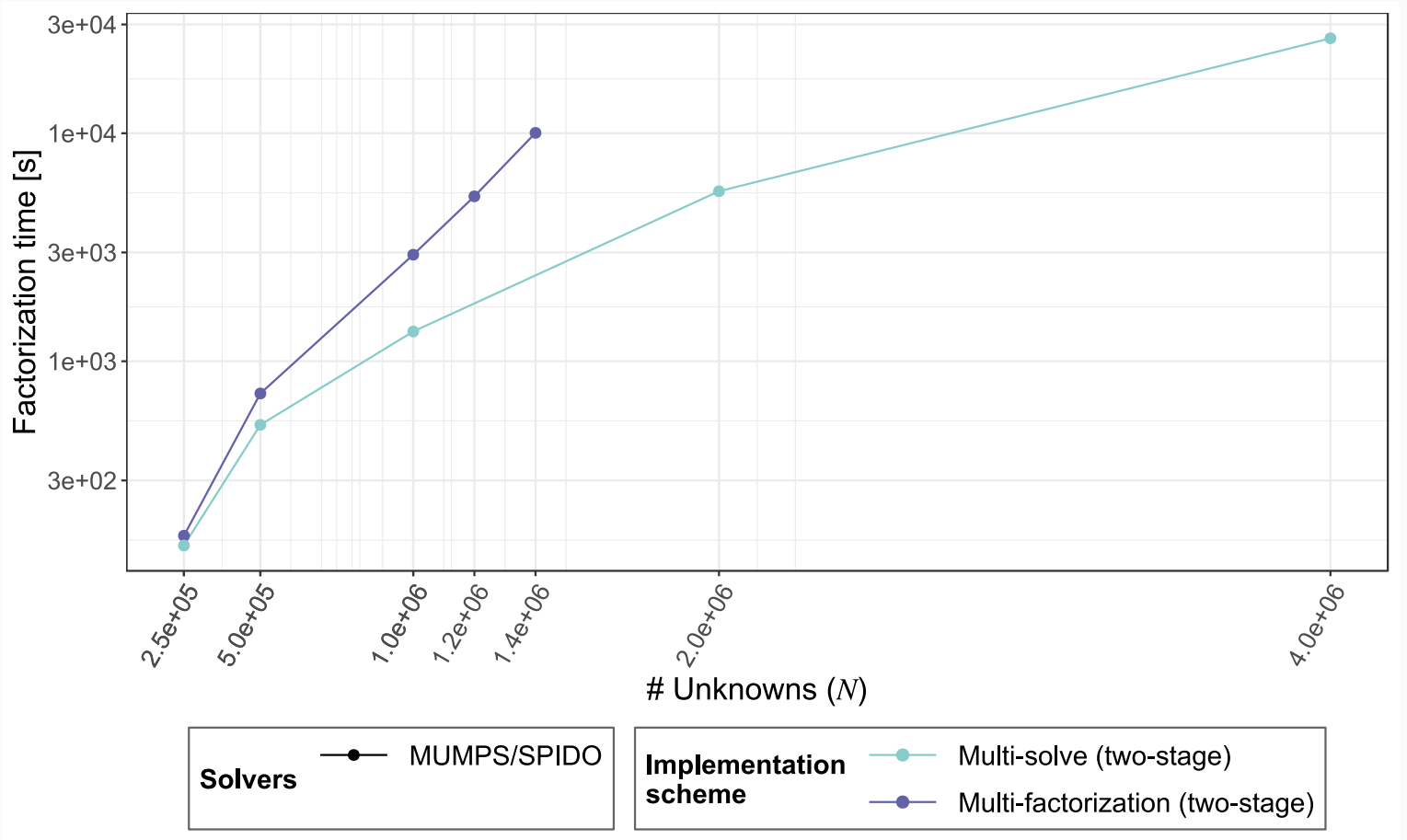


Figure 6 – Comparaison des meilleurs temps d'exécution de multi-solve et multi-factorization pour le couplage MUMPS/SPIDO (sans compression) sur des systèmes couplés FEM/BEM comptant jusqu'à 4 000 000 d'inconnues au total.

Multi-solve (sans compression)

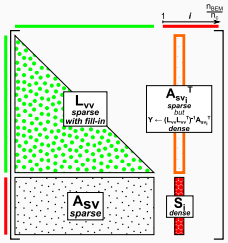


Figure 7 – MUMPS/SPIDO

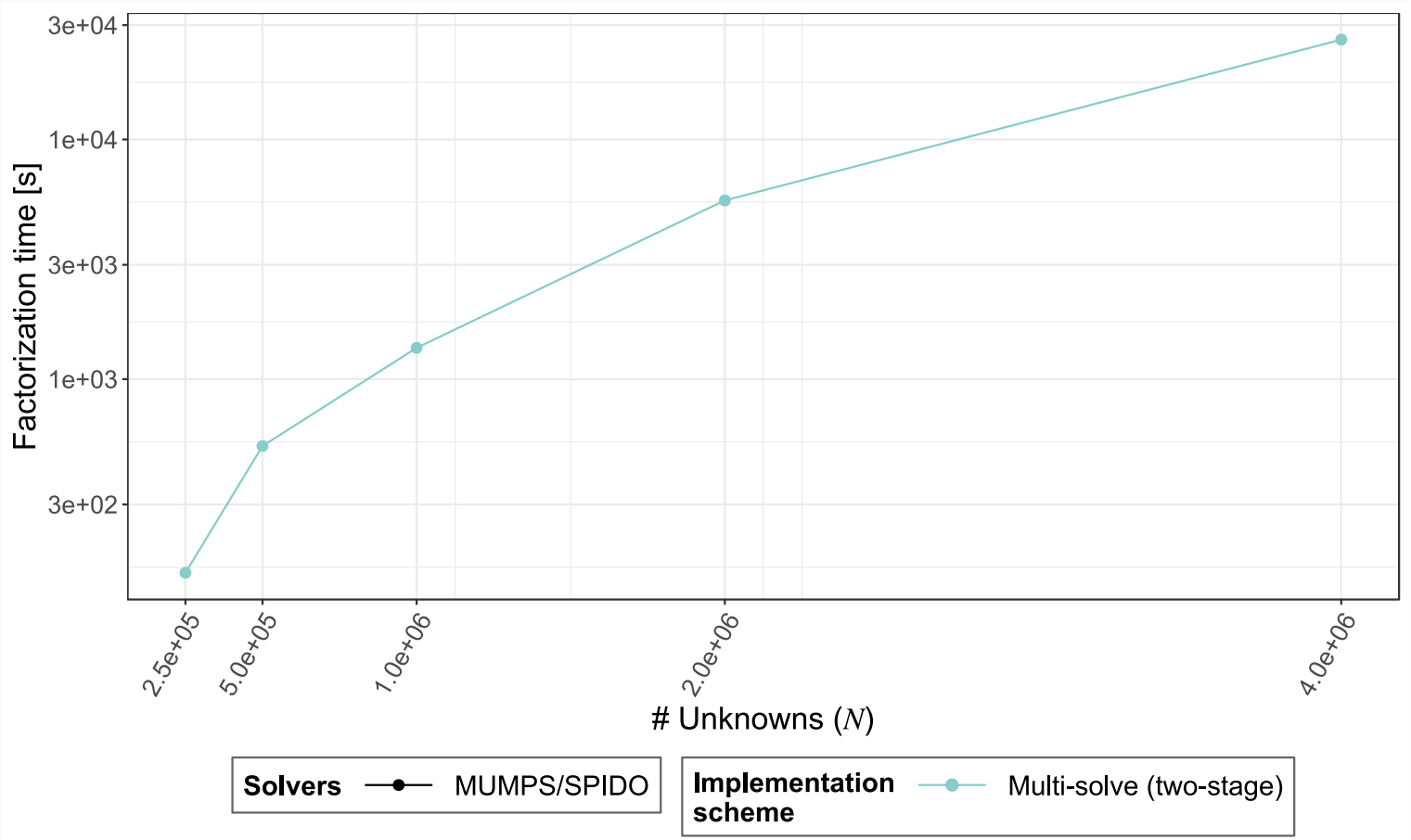


Figure 8 – **Meilleurs temps d'exécution de multi-solve pour le couplage MUMPS/SPIDO (sans compression) sur des systèmes couplés FEM/BEM comptant jusqu'à 4 000 000 d'inconnues au total.**

Compression

Multi-solve

$$S_i = A_{ss_i} - A_{sv}(L_{vv}L_{vv}^T)^{-1}A_{sv_i}^T$$

sans compression

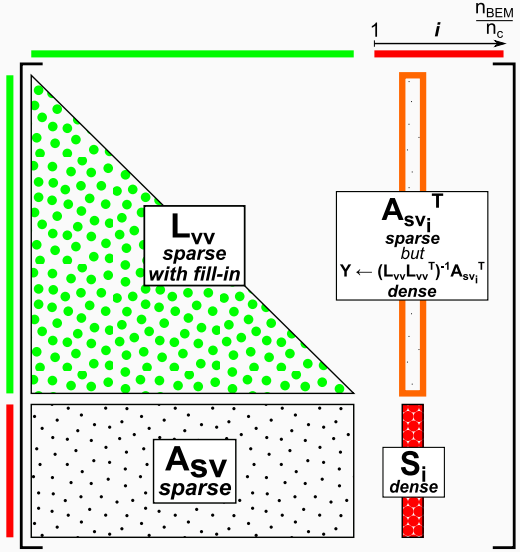


Figure 9 – **MUMPS/SPIDO**

Multi-solve

$$S_i = A_{ss_i} - A_{sv}(L_{vv}L_{vv}^T)^{-1}A_{sv_i}^T$$

sans compression → avec compression

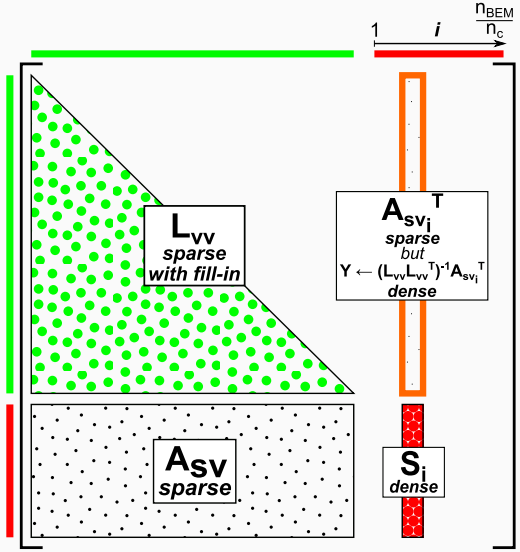


Figure 9 – MUMPS/SPIDO

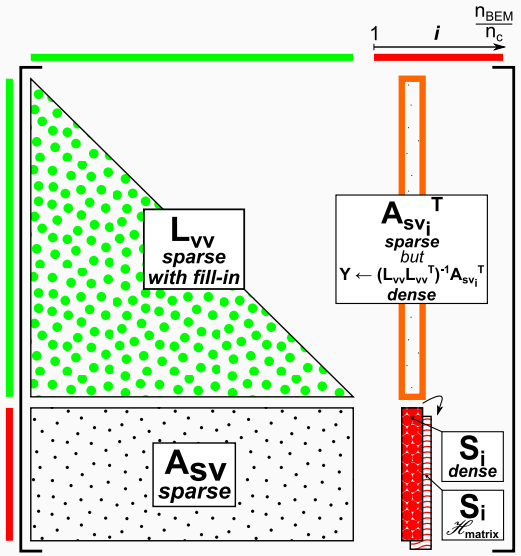


Figure 10 – MUMPS/HMAT (v1)

Multi-solve

$$S_i = A_{ss_i} - A_{sv}(L_{vv}L_{vv}^T)^{-1}A_{sv_i}^T$$

sans compression → avec compression

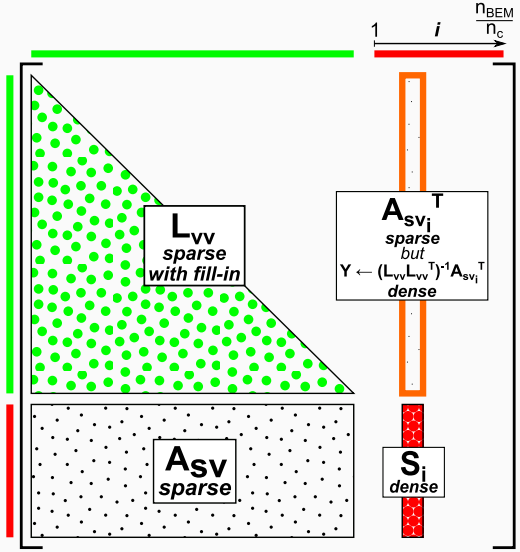


Figure 9 – MUMPS/SPIDO

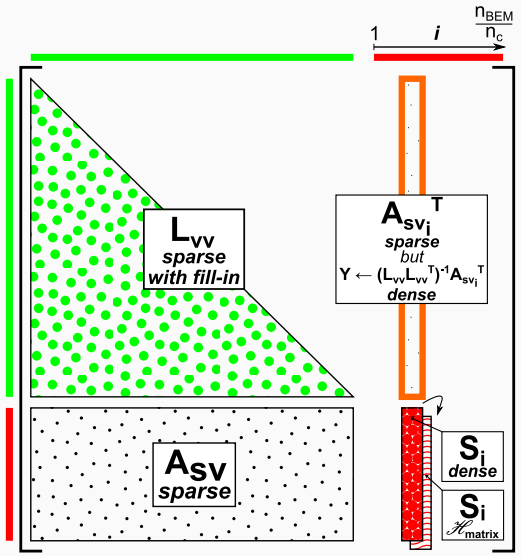


Figure 10 – MUMPS/HMAT (v1)

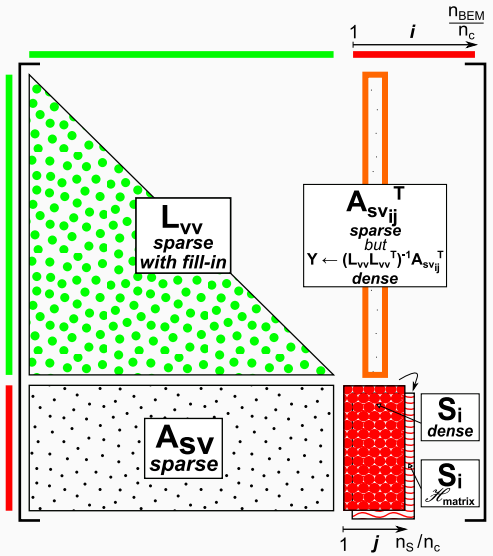


Figure 11 – MUMPS/HMAT (v2)

Multi-solve (sans vs. avec compression)

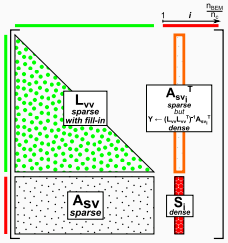


Figure 12 – MUMPS/SPIDO

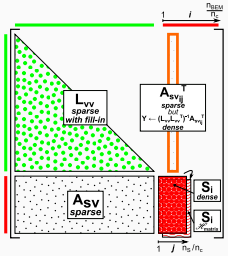


Figure 13 – MUMPS/HMAT (v2)

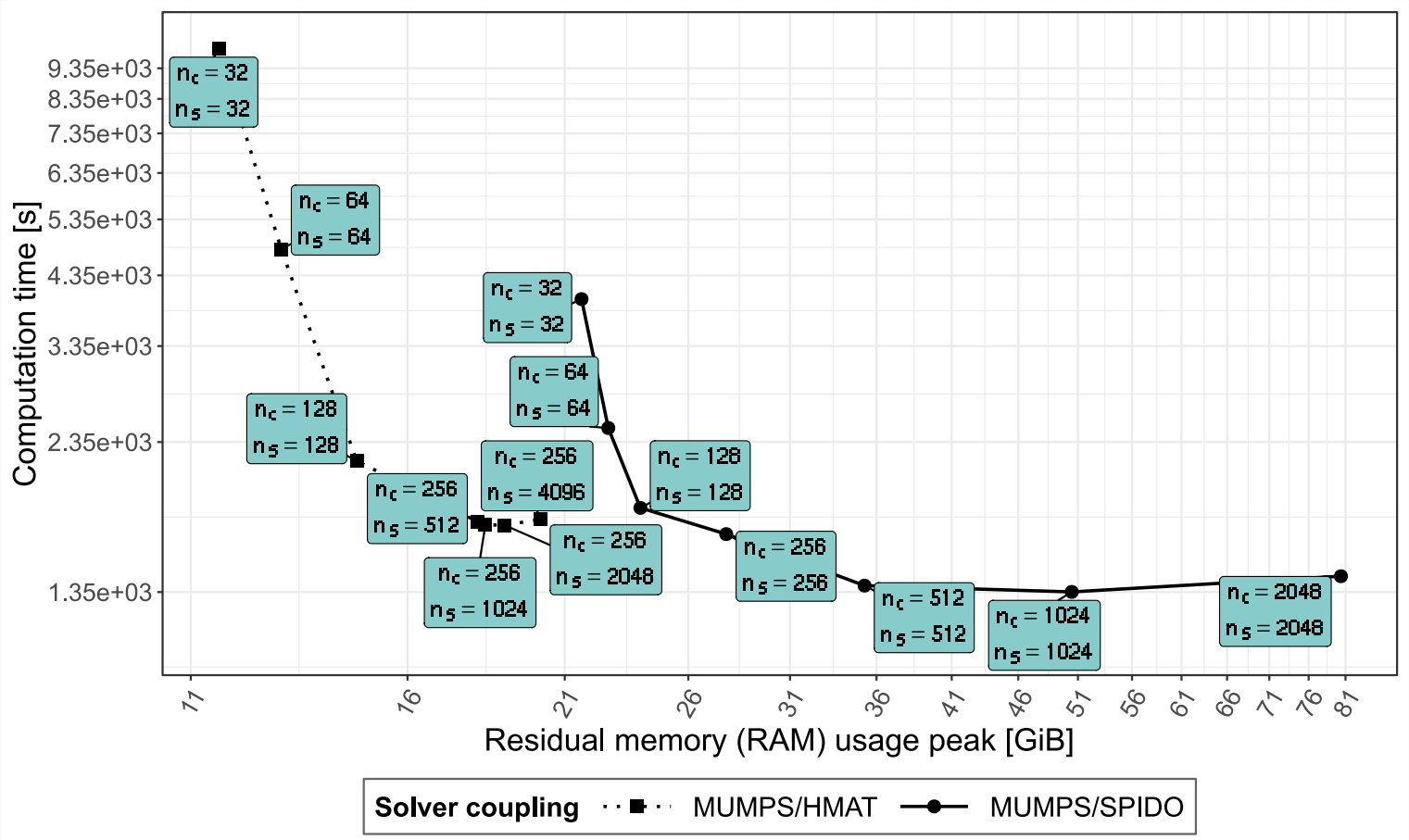


Figure 14 – Comparaison mutuelle des implémentations multi-solve pour les couplages MUMPS/SPIDO (sans compression) et MUMPS/HMAT (avec compression, v2) sur des systèmes couplés FEM/BEM comptant 1 000 000 d'inconnues (FEM : 962 831/BEM : 37 169) et en considérant différentes valeurs de n_C et n_S.

Récapitulatif

Schémas à deux étapes (two-stage)

- vers la compression
- multi-solve
  - pas d'utilisation de l'API complément de Schur
  - traitement des problèmes plus grands qu'avec multi-factorization
- multi-factorization
  - plusieurs appels à l'API complément de Schur
  - coût de multiples factorisations très élevé
    - compression ?

Récapitulatif

Multi-solve

- MUMPS/SPIDO (sans compression) vs. MUMPS/HMAT (avec compression)
- capacité de mémoire vive suffisante → exécution plus rapide
- compression de rang faible dans HMAT → empreinte mémoire plus faible
  - traitement de problèmes plus grands qu'avec MUMPS/SPIDO ?
  - plus vite ?

Pour la suite...

- étendre l'étude centrée sur la consommation mémoire vers multi-factorization
- évaluer l'impact du calcul *out-of-core*
- passer les tests de performance à l'échelle → plusieurs nœuds de calcul

Récapitulatif

Nos contributions récentes dans le domaine

- une comparaison des implémentations existantes de solveurs pour la résolution des systèmes couplés FEM/BEM
  - rapport de recherche montrant les résultats expérimentaux [2]
  - rapport technique décrivant l'envionnement expérimental et le processus pour reproduire toutes les expériences [3]

Merci pour votre attention !

Avez-vous des questions ?

[1] *test_FEMBEM, a simple application for testing dense and sparse solvers with pseudo-FEM or pseudo-BEM matrices*.
`https://gitlab.inria.fr/solverstack/test_fembem`.

[2] E. Agullo, M. Felšöci, and G. Sylvand, *A comparison of selected solvers for coupled FEM/BEM linear systems arising from discretization of aeroacoustic problems*, Research Report RR-9412, Inria Bordeaux Sud-Ouest, June 2021.

[3] ——, *A comparison of selected solvers for coupled FEM/BEM linear systems arising from discretization of aeroacoustic problems : literate and reproducible environment*, Technical Report RT-0513, Inria Bordeaux Sud-Ouest, June 2021.

[4] G. Golub and C. Van Loan, *Matrix computations*, vol. 3, Johns Hopkins University Press, 1996.