Présentation MPNA
Méthode des itérations simultanées

Matthias Beaupère & Pierre Granger

M2 CHPS

20 février 2019

Plan

Introduction

Position du problème

- Calcul de vp de grandes matrices creuses → matrice de Google.
- Seulement quelques vp dominantes.
- Algorithmes robustes.
- Algorithmes adaptés aux architectures massivement parallèles.

La méthode des itérations simultanées

Méthode de la puissance

- Extraction de la vp dominante d'une matrice A.
- Multiplication répétée d'un vecteur initial par A.
- Convergence en $\left(\frac{\lambda_2}{\lambda_1}\right)^N$

Méthode des itérations simultanées

- Espace invariant par A de dim $k > 1$.
- Multiplication répétée du sous-espace par A.
- Orthonormalisation du sous-espace.

Données d'entrée

- M : taille du sous-espace de Krylov
- k : nombre de vecteurs propres demandé
- p : précision demandé
- A : matrice de taille $N \times N$ donnée en entrée
- N_{iter} : nombre d'itérations

Description de l'algorithme

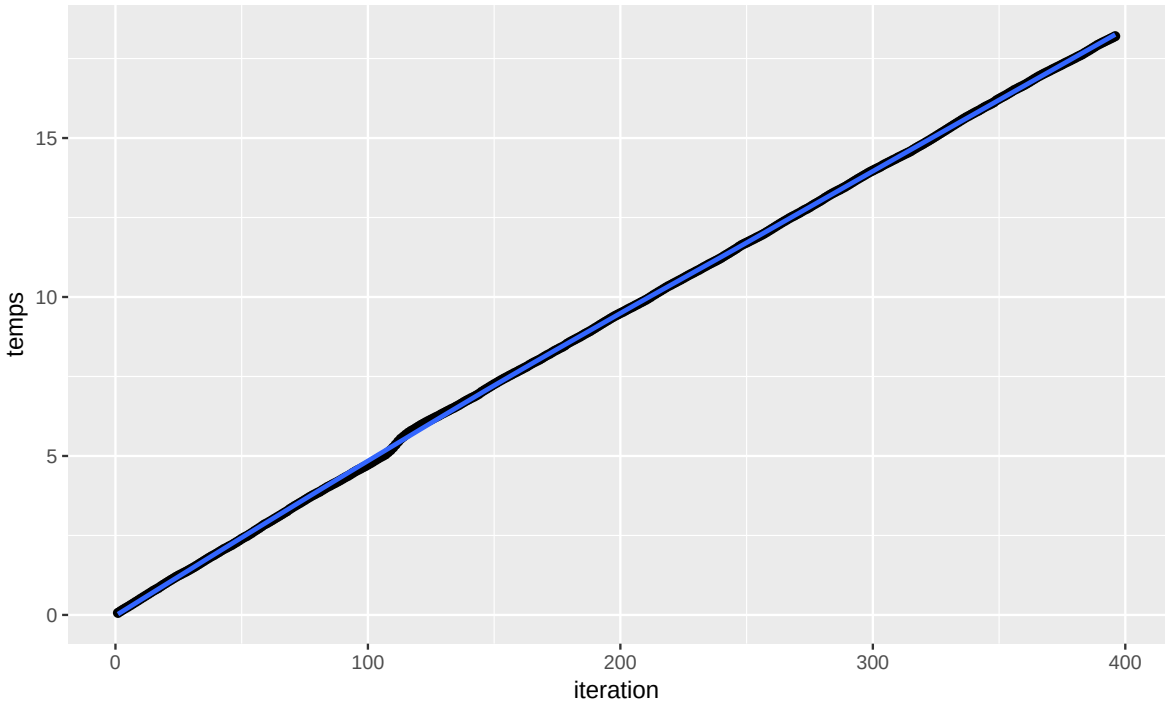
$Q \leftarrow rand()$
while $i = 0..N_{iter} - 1$ OU max(precisions) $<$ p **do**
    $Z = AQ$
    Gram-Schmidt Q
    Projection $B = Z^t AZ$
    Décomposition de Schur $B = Y^t RY$
    Retour dans l'espace d'origine $Q = ZY$
    Calcul de la précision des vecteurs de Q
    Sélection des k vecteurs propres
end while

Performances théoriques

Produit AQ	$O(N^2M)$
Gram-Schmidt	$O(NMlog(M))$
Projection	$O(N^2M)$
Décomposition de Schur	$O(1)$
Précision	$O(NM^2)$
Sélection	$O(1)$

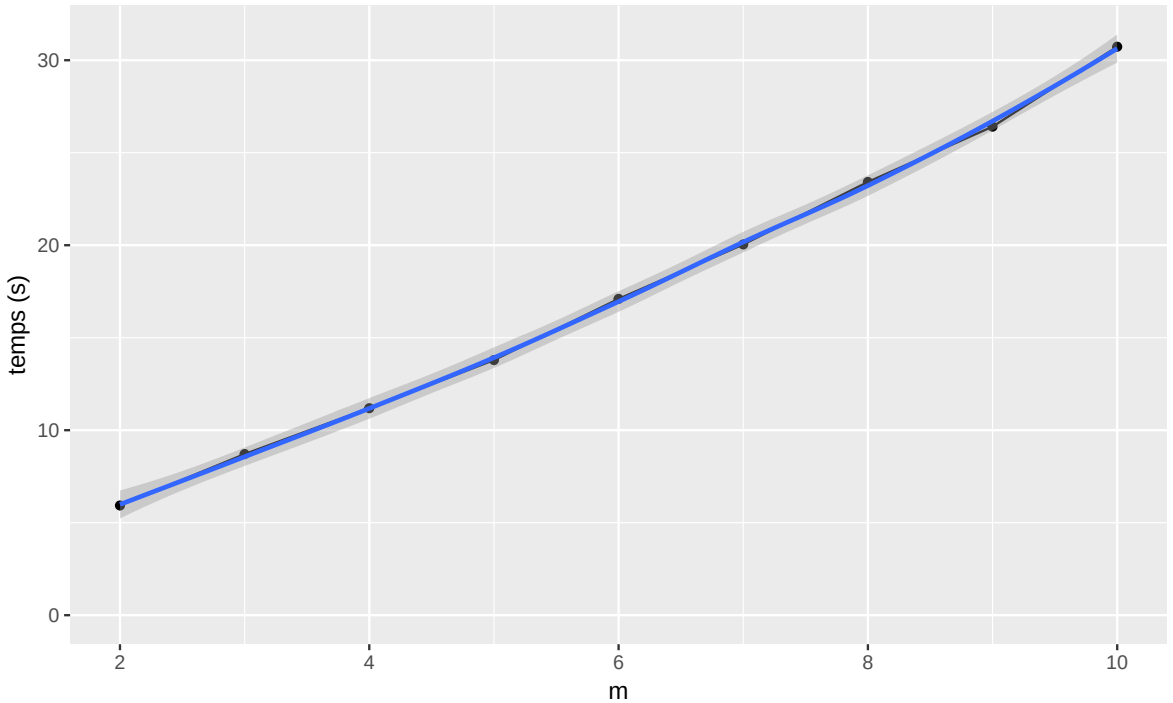
$$\boxed{C^{tot} = O(N_{iter}N^2M)}$$

Nombre d'itérations



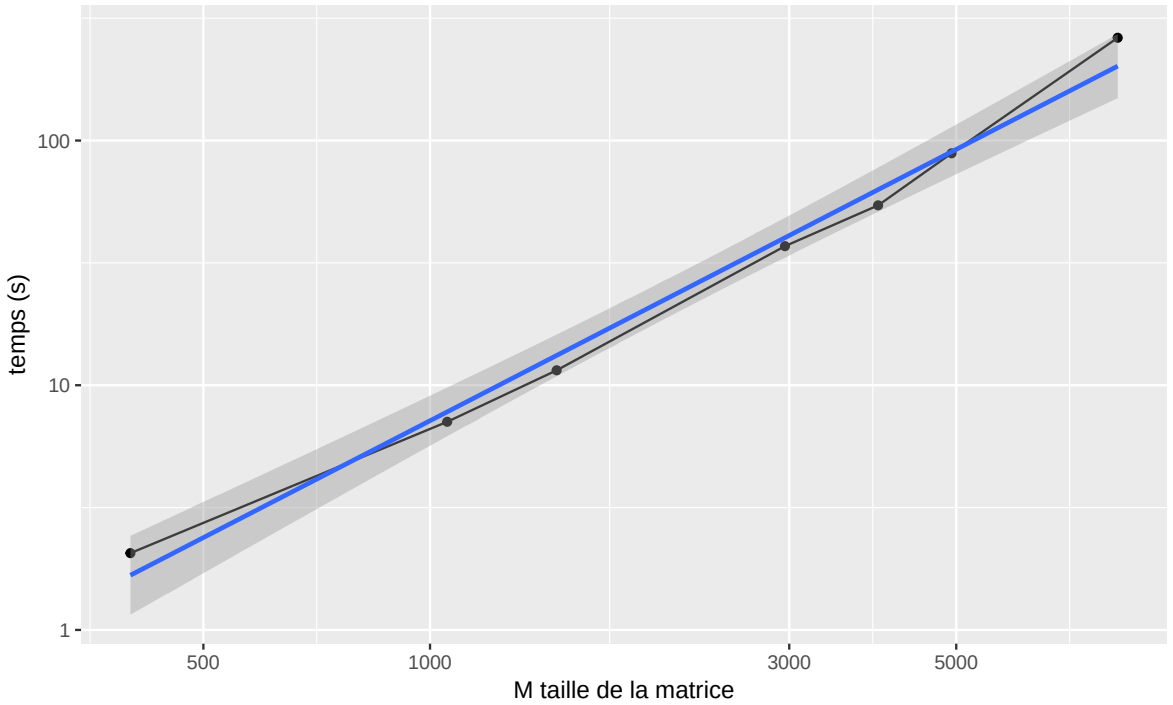
Evolution du temps de calcul en fonction du nombre d'itérations.

Taille du sous-espace de Krylov m



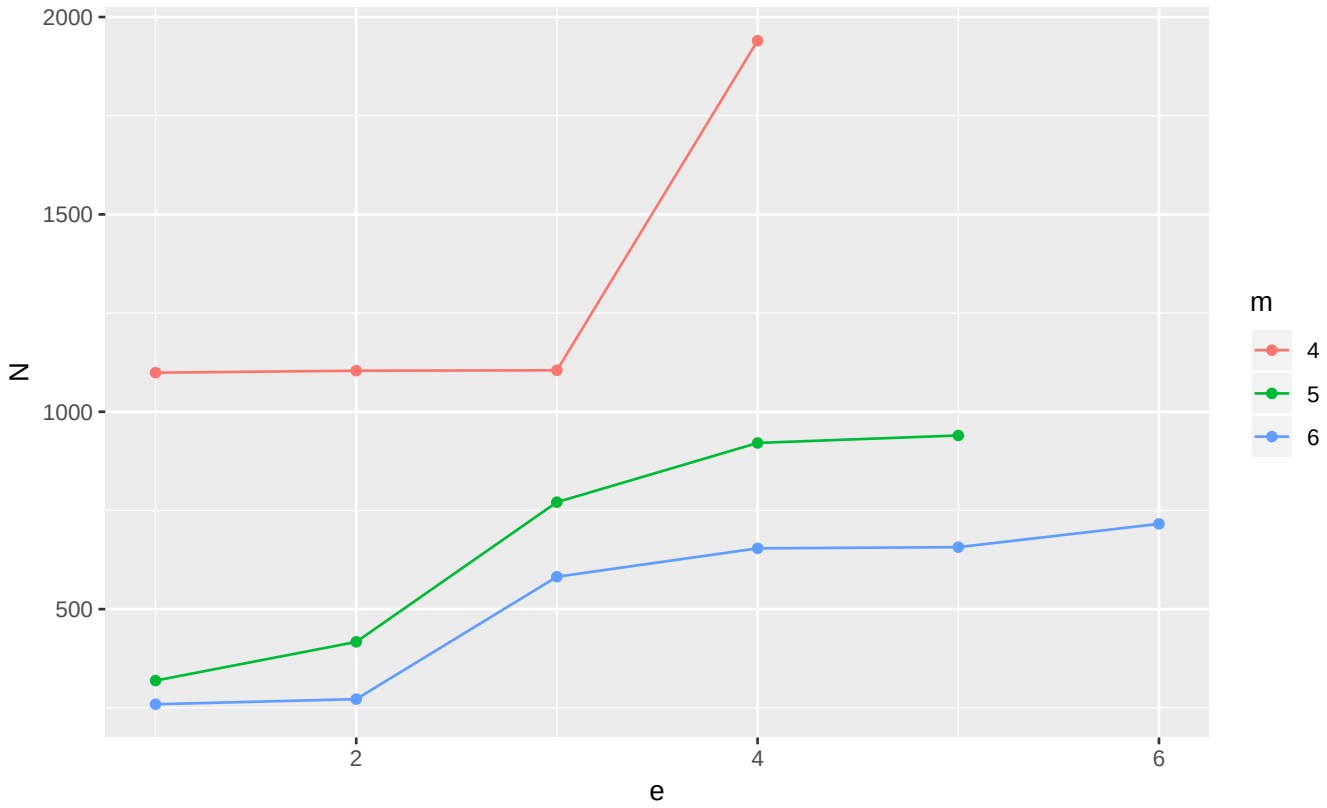
Evolution du temps de calcul en fonction de la taille du sous-espace de Krylov m.

Taille de la matrice M



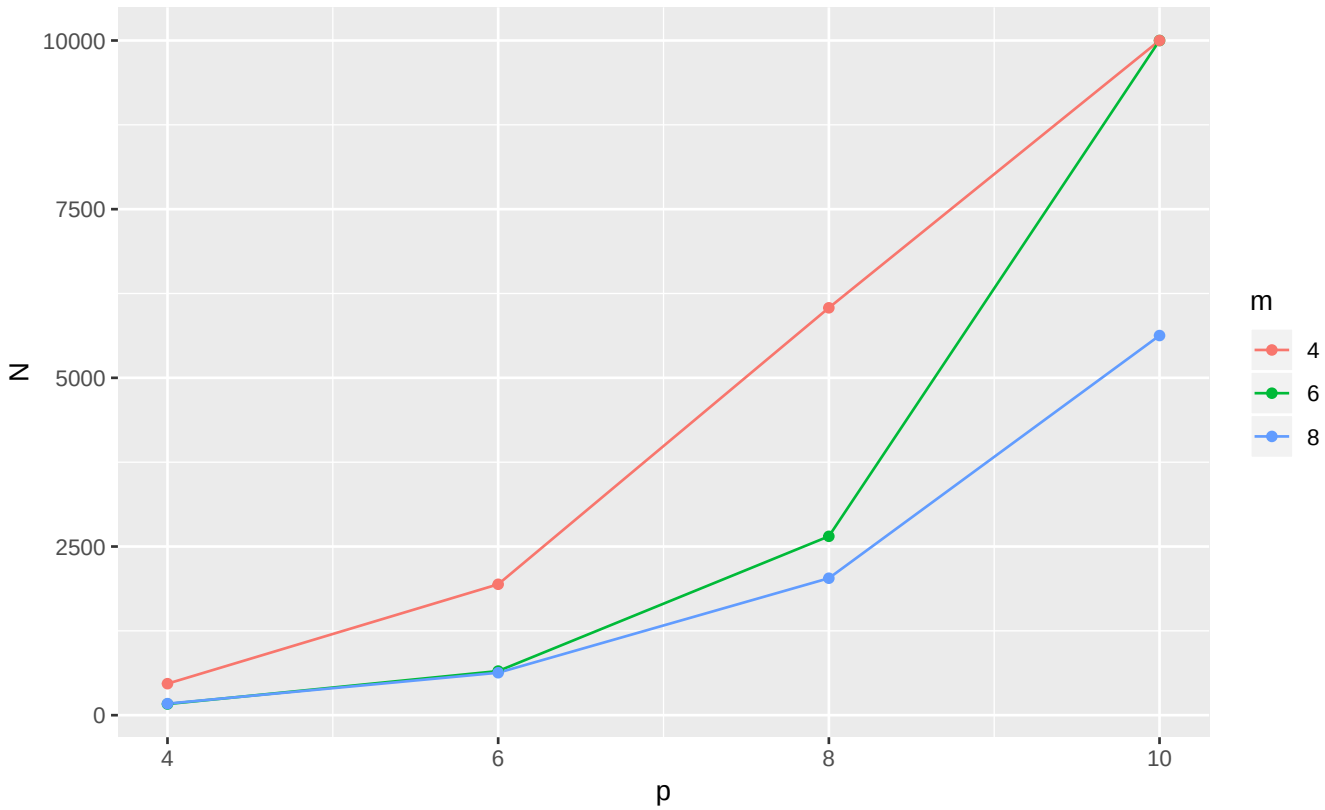
Evolution du temps de calcul en fonction de la taille de la matrice M.

Influence de m



Nombre d'itérations N nécessaires pour faire converger e valeurs propres pour différentes tailles de sous-espace de Krylov m et une précision $p = 10^{-6}$

Influence de p



Nombre d'itérations N nécessaires pour faire converger $e = 4$ valeurs propres pour différentes tailles de sous-espace de Krylov m et une précision p

Principe du locking

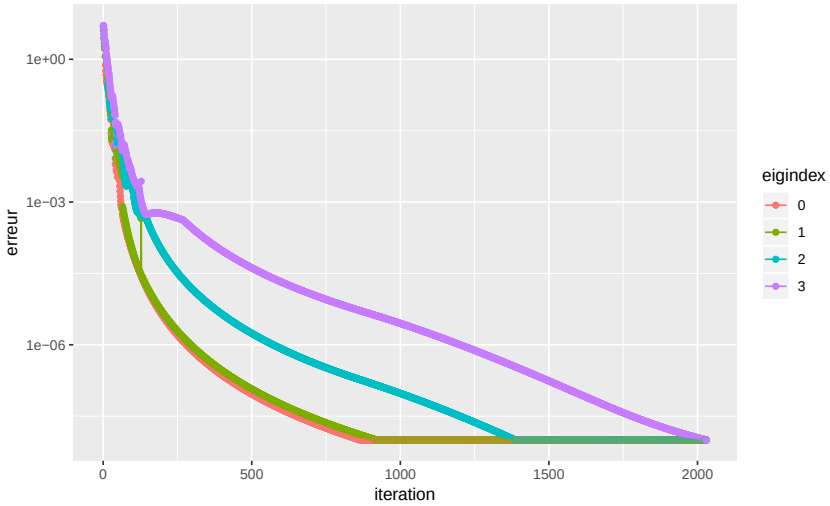
Justificatons

- Vitesses de convergence différentes des vp.
- Perte de temps.
- Instabilités numériques.

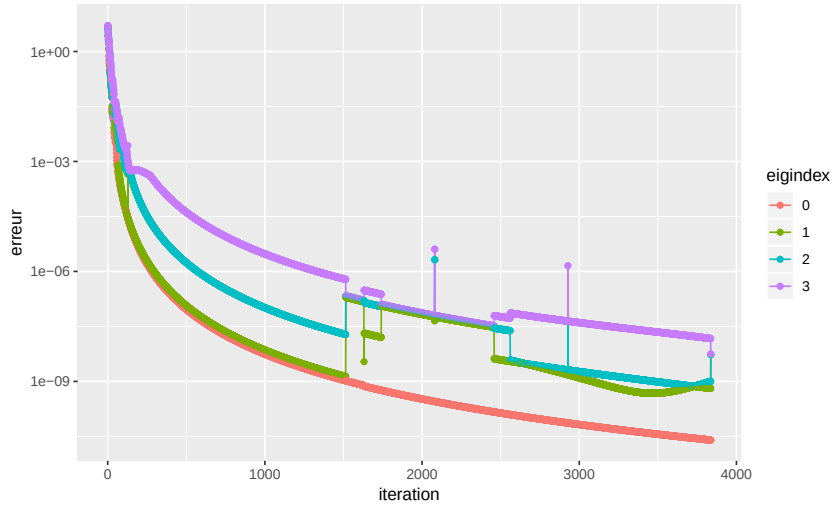
Le locking

- On verrouille les vp lorsqu'ils ont convergé.
- On ne le multiplie par A.
- On diminue m.
- On l'utilise pour l'orthonormalisation.

Performances du locking



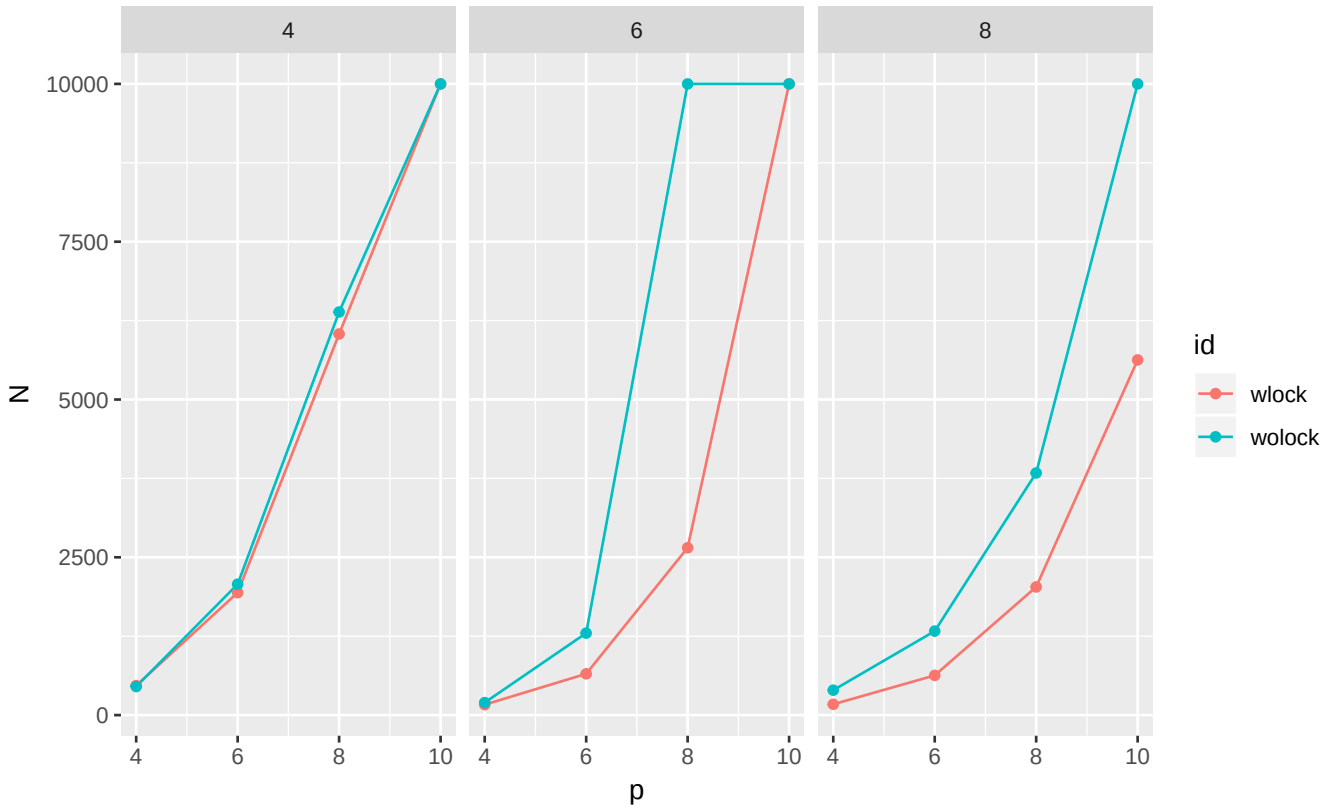
Avec locking



Sans locking

Précision au cours des itérations N pour $e = 4$ valeurs propres pour une taille de sous-espace de Krylov $m = 8$

Performances du locking



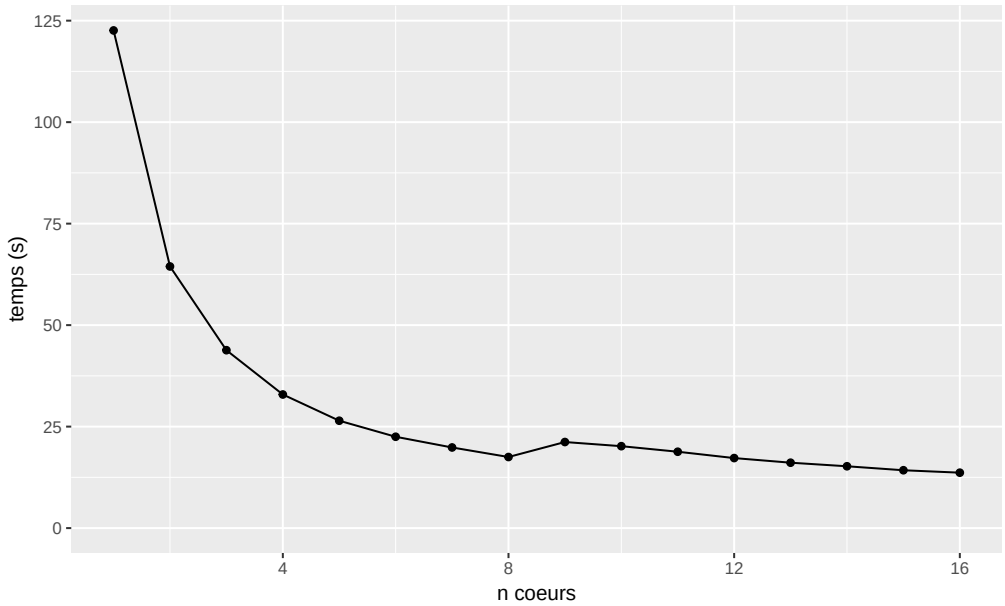
Nombre d'itérations N nécessaires pour faire converger $e = 4$ valeurs propres pour différentes tailles de sous-espace de Krylov m et une précision p avec et sans utilisation du locking

Multicœurs

Parallélisation OpenMP

- On parallélise les produits matriciels dans l'espace d'origine
- Pragmas devant les boucles parallélisables
- Pas de pénalité de communication
- Performance : $C \to \frac{C}{N_{\text{cores}}}$

Multicœurs : performances pratiques



Nœud avec 8 cœurs hyperthreadés pour un calcul de 4 valeurs propres avec $m = 8$ à $p = 10^{-8}$

Accélération x7 avec 8 coeurs physiques !

Multinœuds

Parallélisation du calcul matriciel $C = AB$

- **Rank 0** partage les matrices A et B aux autres processus
- **Tous** calculent une sous-matrice et renvoient le résultat
- **Rank 0** assemble C

Performances théoriques

On pose N le nombre de processus

Temps de calcul processeur

Chaque cœur calcul $\frac{1}{N}$ de la matrice C

$$\text{Accélération} \times \text{N}$$

Communication pour un calcul $C = AB$

- **Aller** : Toute la matrice B en broadcast
- **Aller** : $\frac{1}{\sqrt{N}}$ de la matrice A pour chaque processus
- **Retour** : $\frac{1}{N}$ de la matrice C

Multinœuds : performances pratiques

Sur le supercalculateur poincare

taches	noeuds	N	threads	M	temps(s)
1	1	1473	16	10	4.899005
4	4	1473	16	10	9.844218
6	1	1473	16	10	66.033455
1	1	4929	16	10	29.481719
4	4	4929	16	10	82.517647

Conclusion

Algorithme

- Complexité pratique proche de la théorie.
- Importance du choix de m en fonction des autres paramètres.
- Meilleure convergence et efficacité avec locking.

Parallélisation

- Très bonne parallélisation intra-nœud.
- Mauvaise scalabilité sur inter-nœud $\rightarrow$ communications trop couteuses.
- Méthodes hybrides probablement mieux adaptées à l'inter-nœud.