Méthode d'Arnoldi hybride sur réseau de machines hétérogènes

Serge PETITON

Laboratoire d'Informatique Fondamentale de Lille (LIFL), URA CNRS 369
Université des Sciences et Technologies de Lille (Lille I)
petiton@lifl.fr

Travail réalisé avec Nahid Emad (PRiSM, Université de Versailles) et Guy Edjlali (DCS, University of Maryland at College Park); en partie à l'ETCA, à Yale (CT, USA) et à l'AHPCRC (MN, USA).

ASCI, Jeudi 20 Juin 1996

Méthode d'Arnoldi hybride

sur réseau de machines hétérogènes

PLAN

I Introduction

II Les méthodes d'Arnoldi itératives

III Une méthode d'Arnoldi data parallèle

III La méthode d'Arnoldi hybride parallèle

IV Expérimentations sur réseau de machines hétérogènes comportant deux Connection Machines

V La méthode d'Arnoldi hybride entrelacée

VI Conclusion et recherches futures

Introduction

De nombreuses applications scientifique demandent le calcul des éléments propres de très grandes matrices creuses ($n \geq 10^6$).

(Collaboration avec l'Aérospatiale, centre de recherche de Suresne ; et plusieurs matrices obtenues au SEH avant 1995)

Quelle solution pour problème de valeur propre de grande taille? :

- `méthodes numériques adaptées,`
- `structure de données adéquate,`
- `calculs en parallèle,`
- `bonne répartition des calculs parallèles,`
- `accélération de la convergence`

Mélange de ces techniques pour résoudre une application.

Introduction (suite)

Le calcul scientifique parallèle demande de nombreuses études dans plusieurs domaines ; par exemple pour la méthode d'Arnoldi hybride :

Arnoldi sur machines data parallèles,

Matrices creuses sur machines data parallèles,

Communication entre machines parallèles hétérogènes,

Méthodes hybrides,

Tests de convergence,

Vecteurs de départ et redémarrage,

....

Problème étudié :

Calculs de valeurs et vecteurs propres de très **grandes** matrices **creuses** (non régulières) **non Hermitiennes**.

Hypothèses :

Convergence **lente** et difficile (matrices non normales et à grands défauts de normalité)

Les méthodes d'Arnoldi itératives

Grands problèmes de valeur propre

Soient A une matrice $n \times n$, $\mathcal{K}_m$ un sous-espace de $\mathcal{C}^n$ avec $dim(\mathcal{K}_m) = \nu << n$. La méthode de projection orthogonale sur le sous-espace $\mathcal{K}_m$ consiste à approcher une paire d'éléments propres (λ_i, w_i) de A par $(\lambda_i^{(m)}, w_i^{(m)})$ satisfaisant:

$$\begin{cases} \lambda_i^{(m)} \in \mathcal{C}, w_i^{(m)} \in \mathcal{K}_m \\ \mathcal{P}_m(A - \lambda_i^{(m)} I) w_i^{(m)} = 0 \end{cases} \tag{1}$$

où $\mathcal{P}_m$ est la matrice de projection orthogonale sur $\mathcal{K}_m$.
La méthode consiste à construire une base orthonormale V_m dans $\mathcal{K}_m$ et à résoudre (1) dans cette base:

$$\begin{cases} \lambda_i^{(m)} \in \mathcal{C}, y_i^{(m)} \in \mathcal{C}^\nu \\ (\hat{C}_m - \lambda_i^{(m)}) y_i^{(m)} = 0 \end{cases} \tag{2}$$

avec $w_i^{(m)} = V_m y_i^{(m)}$ et où la matrice $\hat{C}_m = V_m^H A V_m$ d'ordre ν représente l'application

$$\mathcal{A}_m = \mathcal{P}_m A_{|\mathcal{K}_m} : \mathcal{K}_m \longrightarrow \mathcal{K}_m \text{ dans la base } V_m.$$

Méthode basée sur la convergence des sous-espaces de Krylov vers le sous-espace propre lié aux vecteurs propres associés aux r valeurs propres dominantes.

Choix du sous-espace de projection

Soit $S = lin(u_1, \cdots, u_\mu)$ avec $1 \le \mu < n$.

$\mathcal{K}_m = lin(S, AS, \cdots, A^{m-1}S)$ pour $m = 1, \cdots \nu < n$

$dim(\mathcal{K}_m) = \mu m = \nu$.

A Hermitienne: Lanczos ($\mu = 1$) et Lanczos par bloc ($\mu > 1$)

A non-Hermitienne: Arnoldi ($\mu = 1$) et Arnoldi par bloc ($\mu > 1$)

"Convergence"

$(\lambda_i^{(m)}, w_i^{(m)}) \xrightarrow{?} (\lambda_i, w_i)$ lorsque m croit.

Soit $\alpha_m = dist(w_i, \mathcal{K}_m) = \| (I - \mathcal{P}_m)w_i \|_2 = \sin \Theta(w_i, \mathcal{K}_m)$

(a) On peut borner $| \lambda_i - \lambda_i^{(m)} |$ et $\| w_i - w_i^{(m)} \|_2$ en fonction de α_m

(b) on peut montrer que pour le choix ci-dessus de $\mathcal{K}_m$, il existe un ou plusieurs $w_i^{(m)}$ t.q. α_m soit petit.

$$\Downarrow$$

1- les vps situées aux extrémités du spectre convergent plus vite,

2- la convergence est plus rapide si ces vps sont bien séparées,

3- la distance diminue si m croit mais pour m grand le problème du stockage et du coût du calcul se pose

La méthode d'Arnoldi itérative (m fixé, $r = 1$)

(i)- Choix d'un vecteur initial v,

(ii)- Projection d'Arnoldi sur $\mathcal{K}_{m,v} = lin(v, Av, \cdots, A^{m-1}v)$,

(iii)- Calculer la plus grande valeur de Ritz $\lambda_1^{(m)}$ et son vecteur de Ritz associé $w_1^{(m)}$.

Si la précision est suffisante, stop.

Sinon, $v = w_1^{(m)}$ et aller à *(ii)*.

`Recherche de` r `valeurs propres dominantes` :

$v = \sum_{i=1}^{r} \rho_i w_i^{(m)}$ avec $(\rho_i = 1)_{i=1,r}$ ou choisi selon d'autres critères.

Convergence d'Arnoldi itératif

aucune preuve

`Accélération de la convergence`

- Méthode hybride d'Arnoldi-Tchébycheff [Saad],
- Méthode d'Arnoldi hybride [Petiton,Emad,Edjlali],
- ...

La Première méthode d'Arnoldi hybride

(i)- Choix d'un vecteur initial v et d'une dimension m,

(ii)- Projection d'Arnoldi sur $\mathcal{K}_{m,v}$,

(iii)- Calculer les r valeurs dominantes de Ritz et les vecteurs de Ritz associés. Si la précision est suffisante, stop. Sinon calculer un nouveau v, choisir un nouveau paramètre m et aller à *(ii)*.

Hypothèse

$m \in \{m_1, \cdots, m_\ell\}$ avec $m_1 < \cdots < m_\ell << n$

Algorithme (a) d'Arnoldi hybride

- choix du vecteur initial v
- faire jusqu'à convergence
  pour $i = 1$ à ℓ faire

  (i)- projection d'Arnoldi sur

  $$\mathcal{K}_{m_i,v} = lin(v, Av, \cdots, A^{m_i-1}v)$$

  (ii)- calcul des éléments de Ritz

  $$(\lambda_1^{(m_i)}, w_1^{(m_i)}), \cdots, (\lambda_r^{(m_i)}, w_r^{(m_i)})$$

  (iii)- choix du nouveau vecteur initial v

  $$v = f(w_1^{(m_i)}, \cdots, w_r^{(m_i)})$$

  fin pour i

Algorithme `(b)` **d'Arnoldi hybride**

choix des vecteurs initiaux $v_1, \cdots, v_\ell$

faire jusqu'à convergence

pour $i = 1$ à ℓ faire

(i)- projection d'Arnoldi sur

$$\mathcal{K}_{m_i,v_i} = lin(v_i, Av_i, \cdots, A^{m_i-1}v_i)$$

(ii)- calcul des éléments de Ritz

$$(\lambda_1^{(m_i)}, w_1^{(m_i)}), \cdots, (\lambda_r^{(m_i)}, w_r^{(m_i)})$$

fin pour i

pour $i = 1$ à ℓ faire

$$\textit{(iii)-}\ v_i = f_i(w_1^{(k_1)}, \cdots, w_r^{(k_i)})$$

fin pour i

fin faire

Arnoldi data parallèle

Calculer les valeurs propres dominantes (et leur vecteurs propres associés) de très grandes matrices creuses A non Hermitiennes

Trois parties : projection, calculs de Ritz, test d'arrêt et redémarrage

La phase de projection est composée d'une boucle sur j séquentielle comme suit, (avec v_0 donné et m fixé) :

$$v_j = Av_{j-1} - \sum_{i=0}^{j-1} h_{i,j-1} v_i, \tag{3}$$

$$\text{où } h_{i,j-1} = (Av_{j-1}, v_i); i = 0, j-1, \tag{4}$$

$$h_{j,j-1} = \|v_j\|_2, \tag{5}$$

$$v_j = v_j / \|v_j\|_2. \tag{6}$$

Les vecteurs de la base de Krylov sont donc orthonormalisés.

La matrice $h_{i,j}$ est sous forme d'Hessenberg sup.

Stratégies de redémarrage : combinaison linéaire de r vecteur de Ritz :

$$w_1^{(m)}, w_2^{(m)}, ..., w_r^{(m)};$$

<u>La méthode d'Arnoldi itérative</u> (suite)

La phase de projection et la phase de calcul du nouveau vecteur initial sont fortement data parallèles ; si ce n'est que le calcul de :

$$v_j = Av_{j-1} - \sum_{i=1}^{i=j-1} h_{i,j} v_i, \tag{7}$$

$$avec\; h_{i,j} = (Av_{j-1}, v_i)\ ;\ i = 1, j-1, \tag{8}$$

engendre des communications générales (calcul du type $A(Ax + g(x)) + h(x))$; A denses ou creuses)

Les opérations data parallèles :

$w_j = Av_{j-1},\quad$ *communications générales engendrées*

$(w_j, v_i) = h_{i,j}$ pour $i = 1, j-1,$

$z_j = \sum_{i=1}^{j-1} h_{i,j} v_i,$

$v_j = w_j - z_j,$

Ordre de la complexité data parallèle, par itération, de la méthode d'Arnoldi :

Dense : $\mathcal{O}(mlog_2(n)) + \mathcal{O}(m^2)$, avec n^2 processeurs
Creux : $\mathcal{O}(mlog_2(C)) + \mathcal{O}(m^2)$, avec nC processeurs.

Matrices creuses *data parallèles*

$$A = \begin{pmatrix} & & 13 & & & & 17 & 18 \\ & & & 24 & & 26 & & \\ 31 & 32 & & & 35 & & & \\ 41 & & & & 45 & 46 & & \\ & & & 54 & & 56 & & \\ & 62 & & & 65 & & 67 & \\ 71 & & & 74 & & & & 78 \\ & 82 & 83 & & & & 87 & \end{pmatrix}, \quad Acc = \begin{bmatrix} 13 & 17 & 18 \\ 24 & 26 & - \\ 31 & 32 & 35 \\ 41 & 45 & 46 \\ 54 & 56 & - \\ 62 & 65 & 67 \\ 71 & 74 & 78 \\ 82 & 83 & 87 \end{bmatrix}$$

$$Acr = \begin{bmatrix} 31 & 32 & 13 & 24 & 35 & 26 & 17 & 18 \\ 41 & 62 & 83 & 54 & 45 & 46 & 67 & 78 \\ 71 & 82 & - & 74 & 65 & 56 & 87 & - \end{bmatrix}$$

$$Ai = \begin{bmatrix} 3 & 3 & 1 & 2 & 3 & 2 & 1 & 1 \\ 4 & 6 & 8 & 5 & 4 & 4 & 6 & 7 \\ 7 & 8 & - & 7 & 6 & 5 & 8 & - \end{bmatrix}, \qquad Aj = \begin{bmatrix} 3 & 7 & 8 \\ 4 & 6 & - \\ 1 & 2 & 5 \\ 1 & 5 & 6 \\ 4 & 6 & - \\ 2 & 5 & 7 \\ 1 & 4 & 8 \\ 2 & 3 & 7 \end{bmatrix}$$

$$Jc = \begin{bmatrix} 1 & 2 & 1 & 1 & 3 & 2 & 2 & 3 \\ 1 & 1 & 2 & 1 & 2 & 3 & 3 & 3 \\ 1 & 1 & - & 2 & 2 & 2 & 3 & - \end{bmatrix}, \qquad Ic = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & - \\ 1 & 1 & 1 \\ 2 & 2 & 2 \\ 2 & 3 & - \\ 2 & 3 & 2 \\ 3 & 3 & 2 \\ 3 & 2 & 3 \end{bmatrix}$$

FIG. 1 - *Placement des données; $n = 8$ and $C = 3$.*

$$
A = \begin{pmatrix}
 & & 13 & & & & 17 & 18 \\
 & & & 24 & & 26 & & \\
31 & 32 & & & 35 & & & \\
41 & & & & 45 & 46 & & \\
 & & & 54 & & 56 & & \\
 & 62 & & & 65 & & 67 & \\
71 & & & 74 & & & & 78 \\
 & 82 & 83 & & & & 87 &
\end{pmatrix}, \qquad x = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \end{pmatrix},
$$

$$
X = \left(\begin{array}{c|c|c|c|c|c|c|c}
x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\ \hline
x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\ \hline
x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8
\end{array}\right), \qquad Acr = \left[\begin{array}{c|c|c|c|c|c|c|c}
31 & 32 & 13 & 24 & 35 & 26 & 17 & 18 \\ \hline
41 & 62 & 83 & 54 & 45 & 46 & 67 & 78 \\ \hline
71 & 82 & - & 74 & 65 & 56 & 87 & -
\end{array}\right],
$$

$$
Ai = \left[\begin{array}{c|c|c|c|c|c|c|c}
3 & 3 & 1 & 2 & 3 & 2 & 1 & 1 \\ \hline
4 & 6 & 8 & 5 & 4 & 4 & 6 & 7 \\ \hline
7 & 8 & - & 7 & 6 & 5 & 8 & -
\end{array}\right], \qquad Jc = \left[\begin{array}{c|c|c|c|c|c|c|c}
1 & 2 & 1 & 1 & 3 & 2 & 2 & 3 \\ \hline
1 & 1 & 2 & 1 & 2 & 3 & 3 & 3 \\ \hline
1 & 1 & - & 2 & 2 & 2 & 3 & -
\end{array}\right],
$$

$$
G_1 = \left[\begin{array}{c|c|c|c|c|c|c|c}
31 & 32 & 13 & 24 & 35 & 26 & 17 & 18 \\
x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\
1 & 2 & 1 & 1 & 3 & 2 & 2 & 3 \\
3 & 3 & 1 & 2 & 3 & 2 & 1 & 1 \\ \hline
41 & 62 & 83 & 54 & 45 & 46 & 67 & 78 \\
x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\
1 & 1 & 2 & 1 & 2 & 3 & 3 & 3 \\
4 & 6 & 8 & 5 & 4 & 4 & 6 & 7 \\ \hline
71 & 82 & - & 74 & 65 & 56 & 87 & - \\
x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\
1 & 1 & - & 2 & 2 & 2 & 3 & - \\
7 & 8 & - & 7 & 6 & 5 & 8 & -
\end{array}\right], \qquad G_2 = \left[\begin{array}{c|c|c|c|c|c|c|c}
t_{1,1} & t_{1,2} & t_{1,3} & t_{1,4} & t_{1,5} & t_{1,6} & t_{1,7} & t_{1,8} \\
1 & 2 & 1 & 1 & 3 & 2 & 2 & 3 \\
3 & 3 & 1 & 2 & 3 & 2 & 1 & 1 \\ \hline
t_{2,1} & t_{2,2} & t_{2,3} & t_{2,4} & t_{2,5} & t_{2,6} & t_{2,7} & t_{2,8} \\
1 & 1 & 2 & 1 & 2 & 3 & 3 & 3 \\
4 & 6 & 8 & 5 & 4 & 4 & 6 & 7 \\ \hline
t_{3,1} & t_{3,2} & - & t_{3,4} & t_{3,5} & t_{3,6} & t_{3,7} & - \\
1 & 1 & - & 2 & 2 & 2 & 3 & - \\
7 & 8 & - & 7 & 6 & 5 & 8 & -
\end{array}\right],
$$

$$
G_3 = \left[\begin{array}{c|c|c|c|c|c|c|c}
t_{1,3} & t_{1,4} & t_{1,1} & t_{2,1} & t_{2,4} & t_{2,2} & t_{3,1} & t_{3,2} \\ \hline
t_{1,7} & t_{1,6} & t_{1,2} & t_{2,5} & t_{3,6} & t_{3,5} & t_{3,4} & t_{2,3} \\ \hline
t_{1,8} & - & t_{1,5} & t_{2,6} & - & t_{2,7} & t_{2,8} & t_{3,7}
\end{array}\right], \quad G_4 = \left[\begin{array}{c|c|c|c|c|c|c|c}
w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 & w_8 \\ \hline
w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 & w_8 \\ \hline
w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 & w_8
\end{array}\right].
$$

FIG. 2 - *Calcul de $w = Ax$, puis de $A(Ax + f(x)) + g(x)$ et/ou $A^m(x)$*

Placement initial: Sur chaque processeur virtuel $pv(i_c, j); i_c = 1, C; j = 1, N$, placer sur la grille G :

- Le $j^{ème}$ élément du vecteur V . Soit !!v la variable contenant cet élément v_j.
- i, tel que l'élément contenu dans le processeur virtuel est $(a_{i,j})$.
- Le $i_c^{ième}$ élément non nul de la colonne compressée j de la matrice A (i.e. l'élément $a_{i,j}$). Soit !!A cette variable massivement parallèle.
- j_c, tel que l'élément non nul contenu dans le processeur virtuel soit le $j_c^{ième}$ élément non nul de la ligne compressée.

Algorithme: Sur chaque processeur virtuel $pv(i_c, j); i_c = 1, C; j = 1, N$ faire:

1. !!v * !!A = !!w
2. *Envoyer* !!w au processeur $pv(i, j_c)$
3. Réduction avec addition le long de la plus petite dimension de la géométrie de tous les messages reçus sur ces processeurs $pv(i_c, j), i_c = 1, C; \forall j$. Les résultats sont redondants sur chaque processeur $pv(i_c, j)$, $\forall i_c = 1, C$.

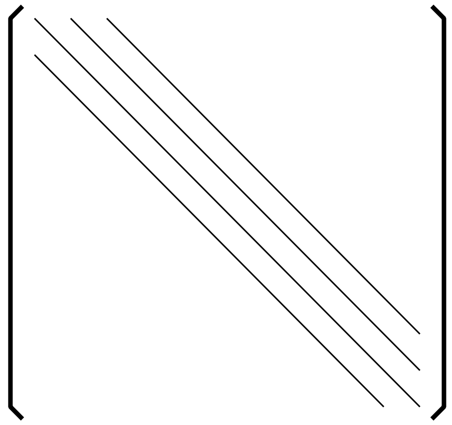
Placement final: Sur chaque processeur virtuel $pv(i_c, j); i_c = 1, C; j = 1, N$, placer sur la grille G :

- !!v et !!A comme initialement placés.
- Le $j^{ième}$ élément du vecteur $z = Av$.

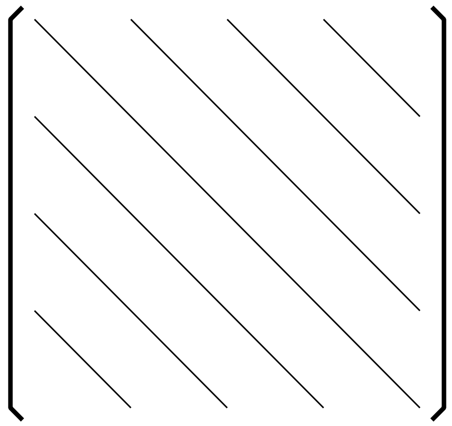
La complexité est donc de $1 + log_2(C)$ opérations massivement

<u>Patrons de matrices creuses de test</u>

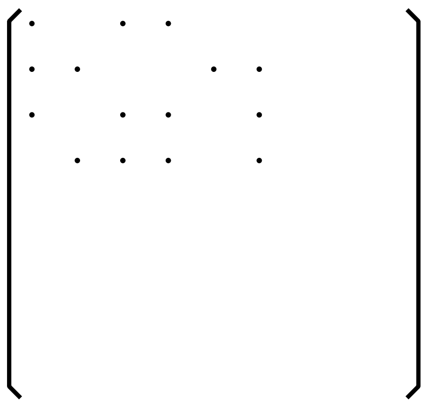
<u>C-diagonales</u>



<u>C-distribuées</u>

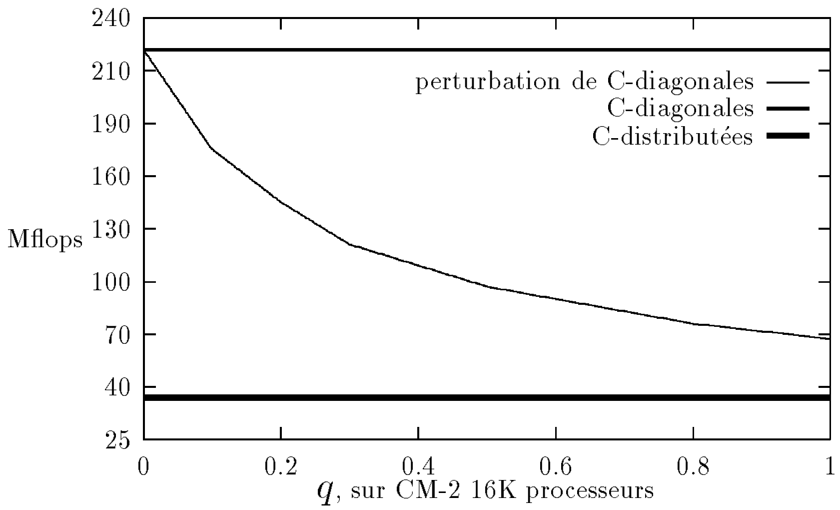


<u>C-perturbées, exemple q=4 :</u>



Multiplication matrice creuse × vecteur

Avec compilation ou vectorisation des communication.



CM-2(200) : Importance de la distance des éléments non nuls à la diagonale.

CM-5 : importance du nombre de messages, patrons dynamiques ; pas de compilation des communications.

Les patrons C-diagonales et C-distribuées engendrent respectivement les meilleures et les pires performances sur les CM-200 et CM-5 (sauf peut être des patrons très particuliers).

Nous limitons donc notre étude des performances à nombre d'itérations fixées à ces deux patrons.

Performances sur une Connection Machines

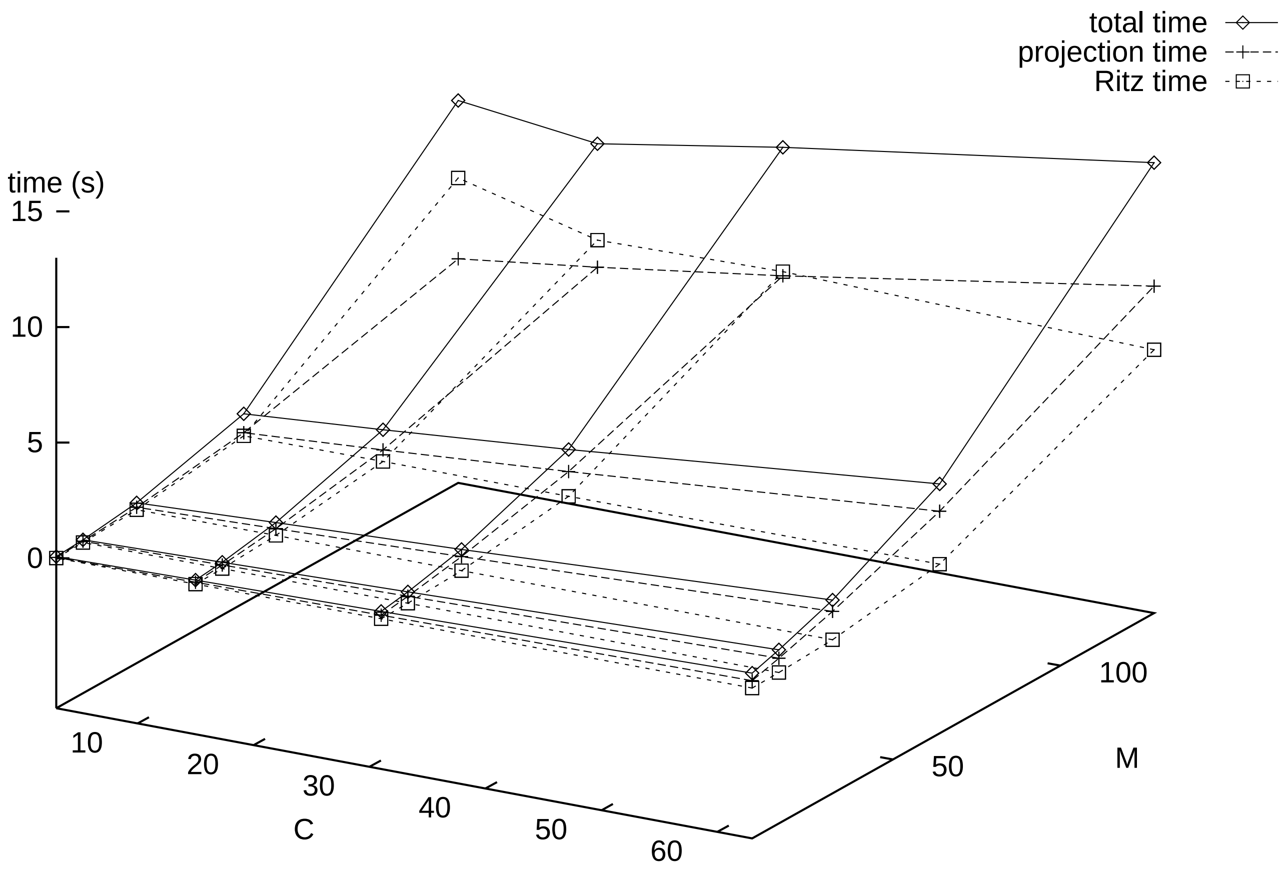


FIG. 3 - *Comparaison entre les parties de la méthode d'Arnoldi, pour une matrice C-diagonale, avec N=16384*

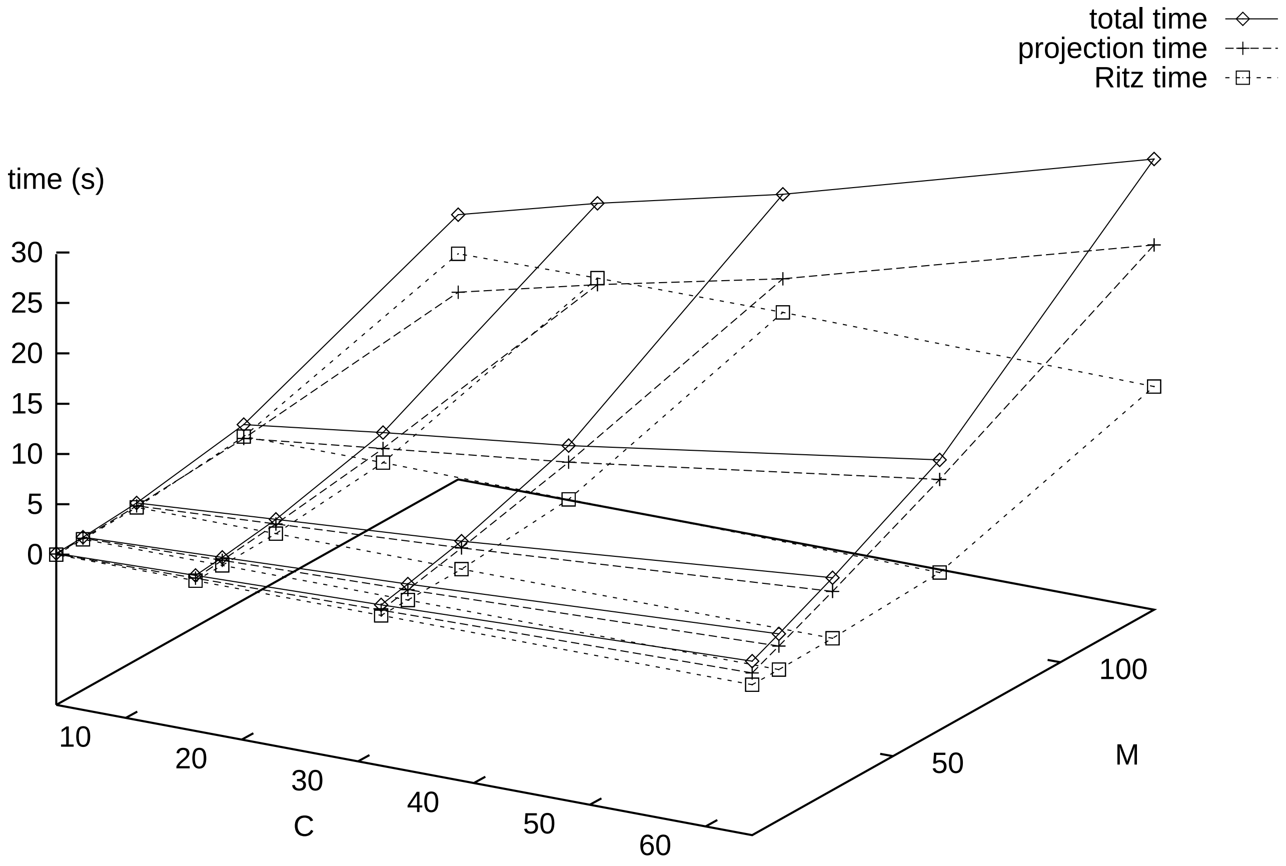


FIG. 4 - *Comparaison entre les parties de la méthode d'Arnodli, pour une matrice C-distributée, avec N=16384*

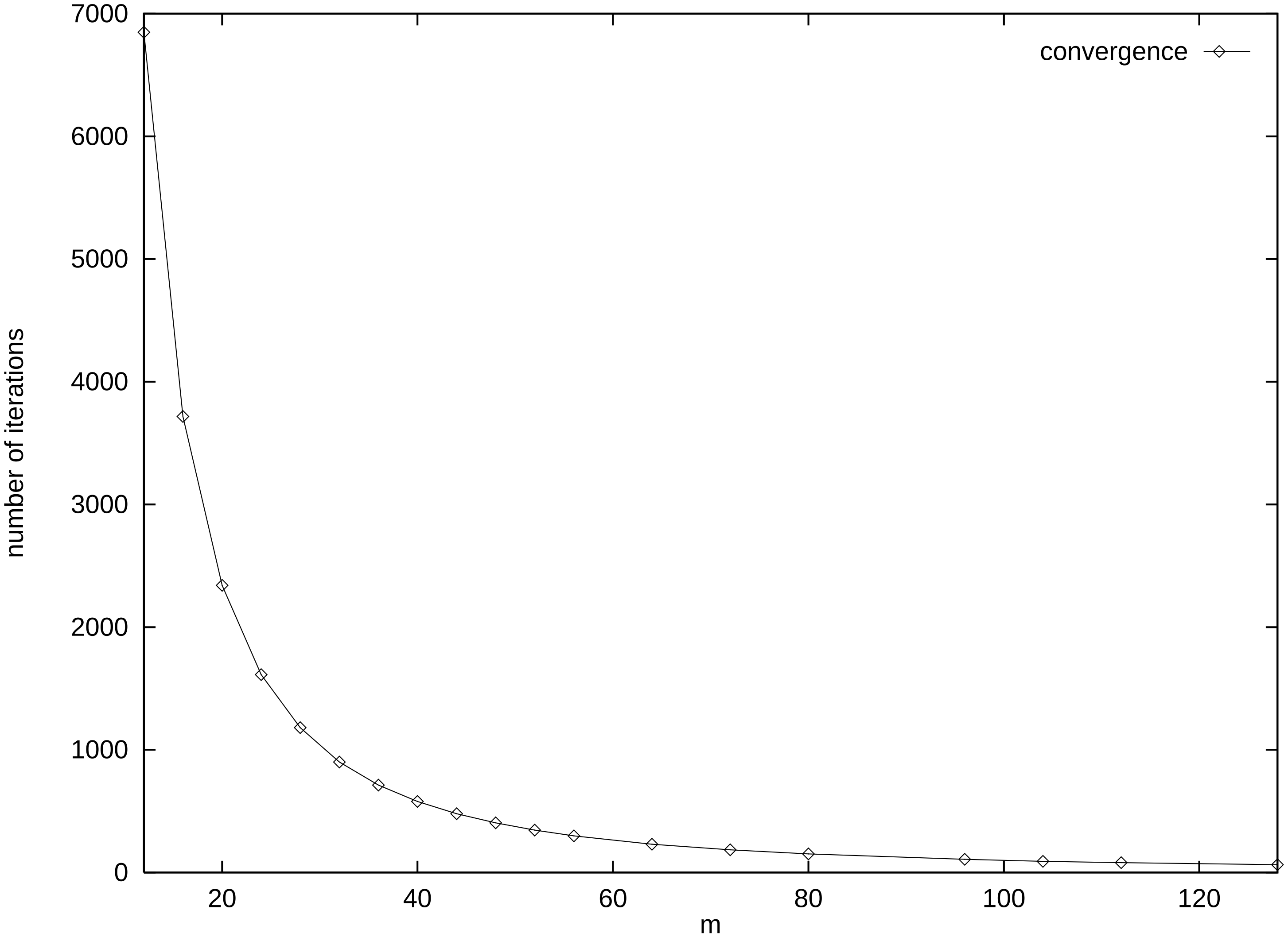


FIG. 5 - *Nombre d'itération en fonction de la taille du sous espace, avec* $r = 1$

Les calculs de Ritz, i.e. dans le sous espace, sont d'importants goulots d'étranglement.

Plus m croit, à n fixé, et plus ce goulot d'étranglement est prédominant sur le temps de calcul global d'une itération d'Arnoldi.

Comme la méthode converge d'autant plus vite que la taille du sous espace m est grande, il nous faut trouver un compromis entre peu d'itérations lentes ou plus d'itérations rapides.

Ce choix est de plus en plus fréquents en calculs parallèles scientifiques.

Les calculs de Ritz seront effectués sur les hôtes ou sur les processeurs de contrôle.

Accélérer la convergence est donc primordiale.

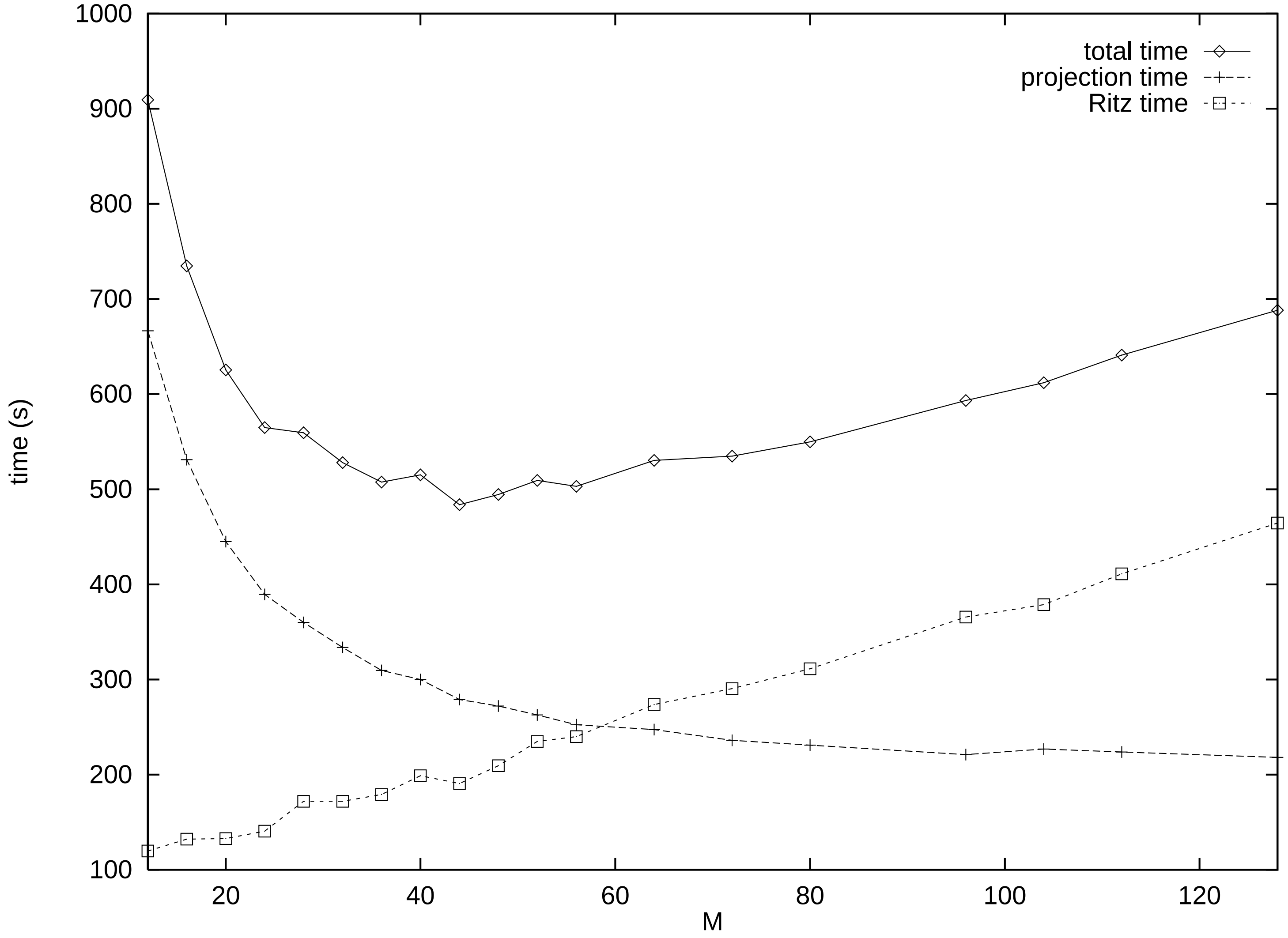


FIG. 6 - *Temps de calcul pour les diverses parties de la méthode d'Arnoldi, avec $r = 1$*

Il existe une taille optimale pour une matrice et une machine fixées.

La trouver?

III <u>Arnoldi hybride</u>

Initialization

set

a number l of different Arnoldi process,
a set of subspace size m_i pour $i = 1, l$, with $m = (m_1, \cdots, m_\ell)^*$,
a set of initial guess $v_1, \cdots, v_\ell$ with $V = (v_1, \cdots, v_l)$.

Iteration

do until convergence

for $i = 1$ to ℓ do

(i)- Arnoldi projection onto $\mathcal{K}_{m_i, v_i} = lin(v_i, Av_i, \cdots, A^{m_i - 1} v_i)$

(ii)- Ritz element computation $\mathcal{R}(i) = ((\lambda_1^{(m_i)}, w_1^{(m_i)}), \cdots, (\lambda_r^{(m_i)}, w_r^{(m_i)}))$

done for i

if no convergence then

for $i = 1$ to ℓ do

(iii)- <u>Generalized Restart</u> : $v_i = f_i(\mathcal{R}(1), \cdots, \mathcal{R}(l))$

done for i

end if

end until

Do in parallel for $i = 1, l$

Arnoldi projection i
Ritz element computation $\mathcal{R}(i)$
Computation of residual $\epsilon_k = \|Aw_k^{m_i} - \lambda_k^{m_i} w_k^{m_i}\|, k = 1, r$
If no convergence then

Do (*Computation process*)

do until convergence

if necessary informations are received then

Hybrid Restart

else

Simple Restart

endif
Arnoldi projection i
Ritz element computation
Computation of $\epsilon_k = \|Aw_k^{m_i} - \lambda_k^{m_i} w_k^{m_i}\|, k = 1, r$

end until

in parallel with (*Communication process*)

do until convergence

Do

send elements of $\mathcal{R}(i)$ to process $k, k = 1, l$ and $k \neq i$ with respect to the function f_k

in parallel with

received elements of $\mathcal{R}(k)$ from process $k, k = 1, l$ and $k \neq i$ with respect to the function f_i

End parallel Do

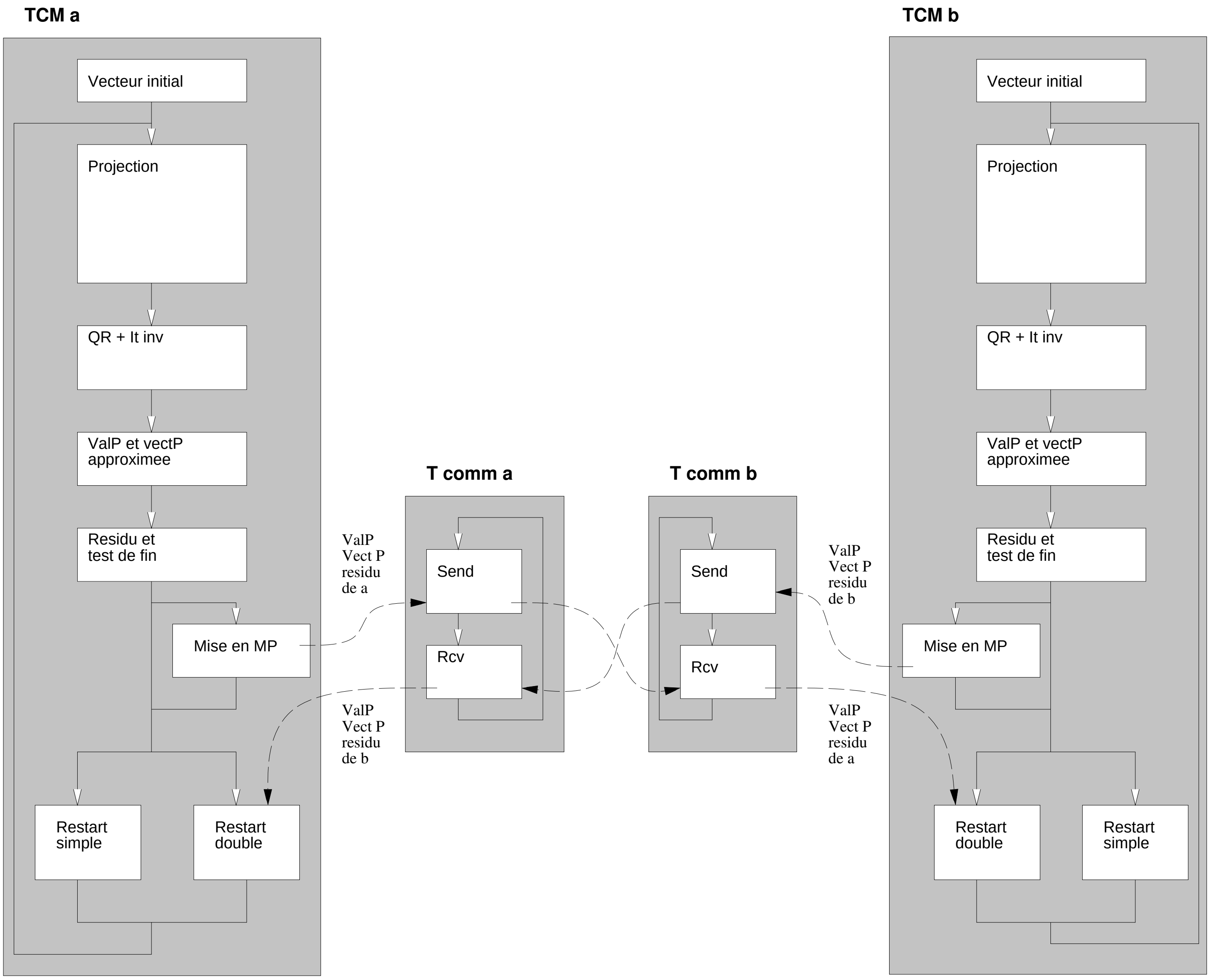
end do until

Done in parallel for i

end if

end do in parallel

Algorithme d'Arnoldi(m, m') Parallèle



La méthode d'Arnoldi Hybride présentée utilise deux machines parallèles : une CM5 à 32 processeurs et CM200 à 4K processeurs (chacune à un total de 128 processeurs vectoriels).

La CM200 connaît les valeurs de Ritz, les vecteurs de Ritz et les résidus calculés sur la CM5 pour l'itération en cours ou pour une itération précédente.

Notons $\lambda_{i,k}^{CM5}$ la i^{eme} valeur de Ritz et $u_{i,k}^{CM5}$ le i^{eme} vecteur de Ritz, calculés par la CM5 durant l'itération k.

Le choix du vecteur initial, de la taille du sous-espace et de la phase de redémarrage de la méthode seront différents sur les deux machines.

La CM200 trouvera des valeurs Ritz $\lambda_{i,k}^{CM200}$, des vecteurs de Ritz $u_{i,k}^{CM200}$ et des résidus. Elle pourra redémarrer l'algorithme en employant :

λ_{i,k_1}^{CM5}, u_{i,k_1}^{CM5} et,

λ_{i,k_2}^{CM200}, u_{i,k_2}^{CM200} et,

leurs résidus associés.

Algorithme d'Arnoldi(m, m') Parallèle (suite)

Les λ_{i,k_1}^{CM5}, u_{i,k_1}^{CM5} sont calculés par la CM5 à l'itération k_1, alors que les λ_{i,k_2}^{CM200}, u_{i,k_2}^{CM200} sont déterminés par la CM200 à l'itération k_2 (Dans notre cas $k_1 > k_2$ car la CM5 est plus rapide que la CM200).

Pour redémarrer l'algorithme avec les données des deux machines, la CM5 devra recevoir les valeurs de Ritz, les vecteurs de Ritz et les résidus calculés par la CM200.

Si les communications et les calculs se chevauchent et si les communications sont rapides, il est alors possible de disposer sur les deux machines, à chaque itération, au moment du restart, des valeurs de Ritz, vecteurs de Ritz et des résidus calculés sur les deux machines.

Il faut pour cela déterminer la taille des sous-espaces en fonction de la rapidité des diverses machines et de la vitesse du réseau.

La phase de redémarrage de l'algorithme est modifiée, elle est divisée en deux parties: restart simple et restart généralisé (ou double si m et m').

Si les valeurs de Ritz, vecteurs de Ritz et résidus calculés par `TCMb` (resp. `TCMa`) sont reçus par `TCMa` (resp. `TCMb`), alors le redémarrage utilise ces données supplémentaires ainsi que les valeurs de Ritz, vecteurs de Ritz et résidus qu'il a calculé.

Stratégies de redémarrage pour Arnoldi(m, m')

Le calcul des résidus permet de sélectionner ces vecteurs.

CM-5 : convergence vers le sous-espace propre de taille m_1

CM-200 : convergence vers le vecteur propre prédominant, ou le suivant si celui ci est déjà calculé. Permet d'accélérer la convergence sur l'autre machine vers les r vecteurs propres recherchés.

Le fait de prendre deux vecteurs de départ différents joue aussi un rôle important lors des premières `itérations`.

D'autres stratégies sont aussi testées.

On peut généraliser au cas de plus de deux machines en parallèle.

IV Expérimentations

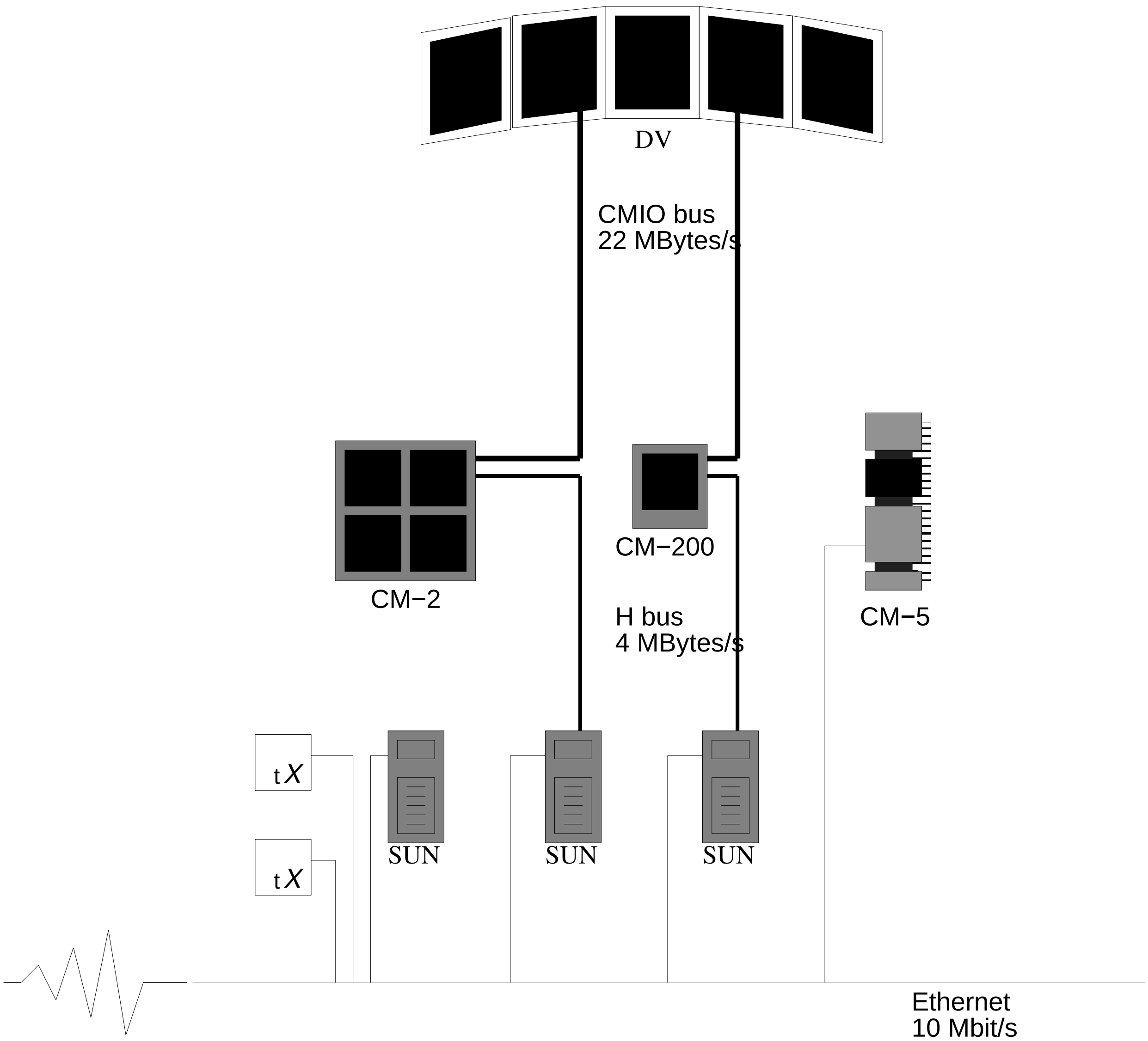


Fig. 7 - *Platforme d'experimentation au SEH, 1992-94*

Actuellement :

LIFL : MP-1 de 16K processeurs + ferme de 16 alpha DEC

IDRIS : Cray T3D + *C98 + Hippi*

Edinburg : idem IDRIS

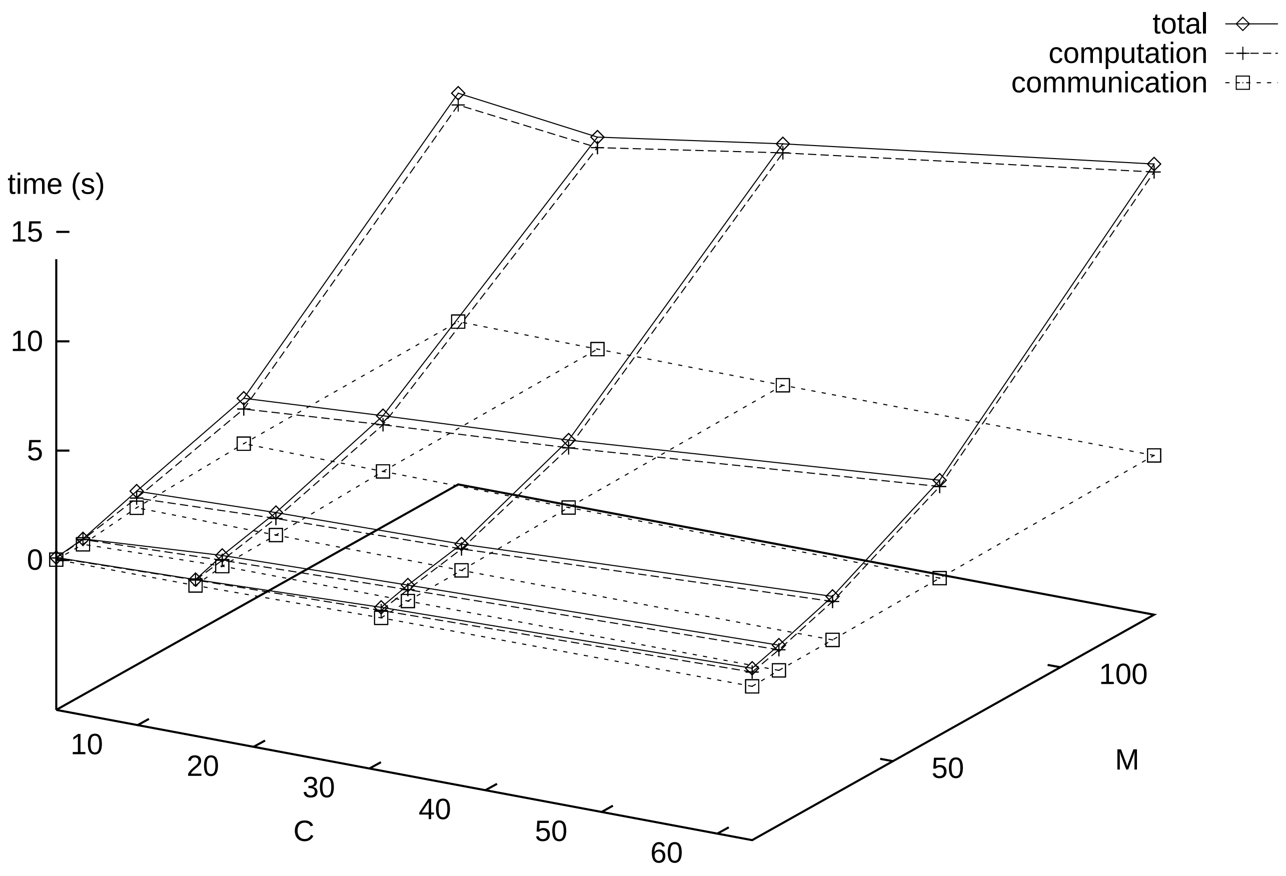


FIG. 8 - *Comparaizon between the total duration, the computation time and the communication time for one Arnoldi(r,m,l,V) iteration on the CM-5, for a C-diagonal pattern matrix.*

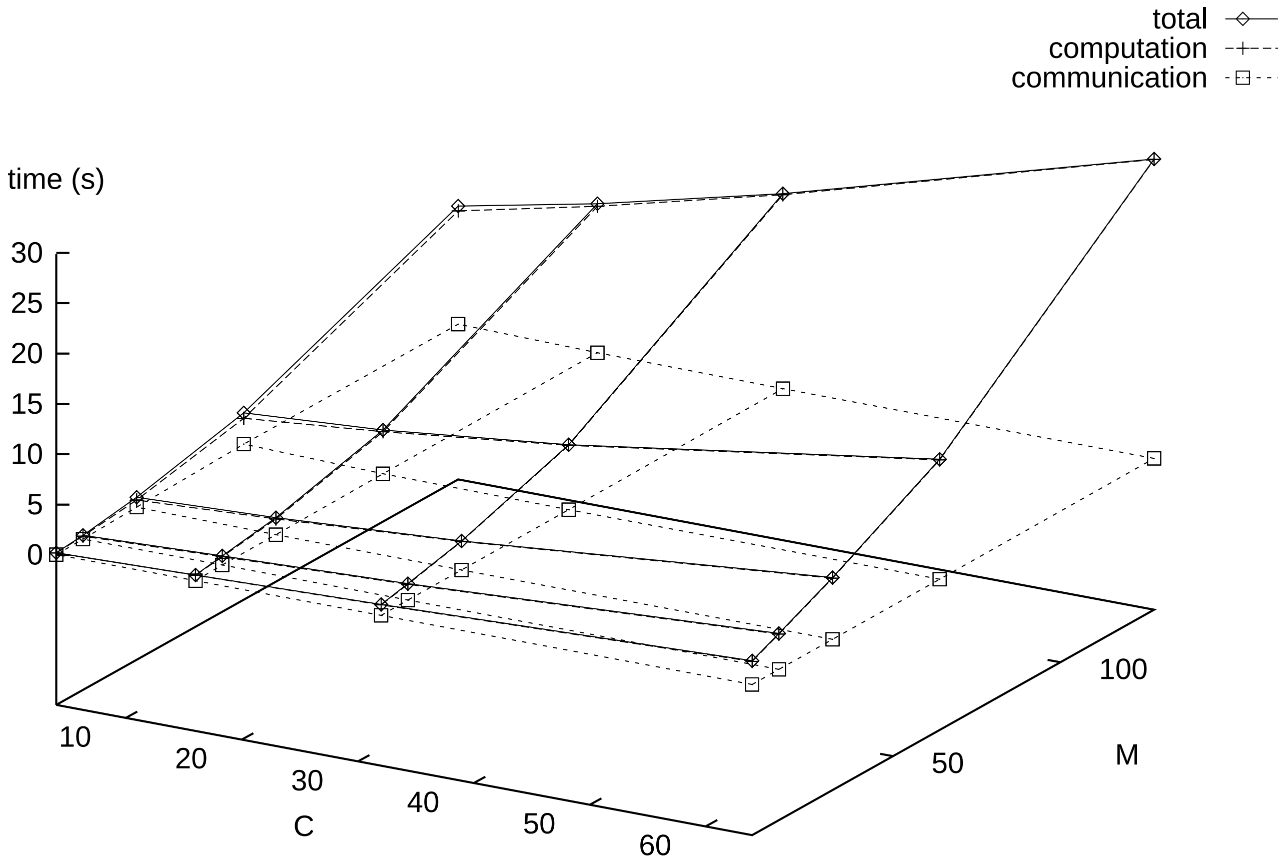
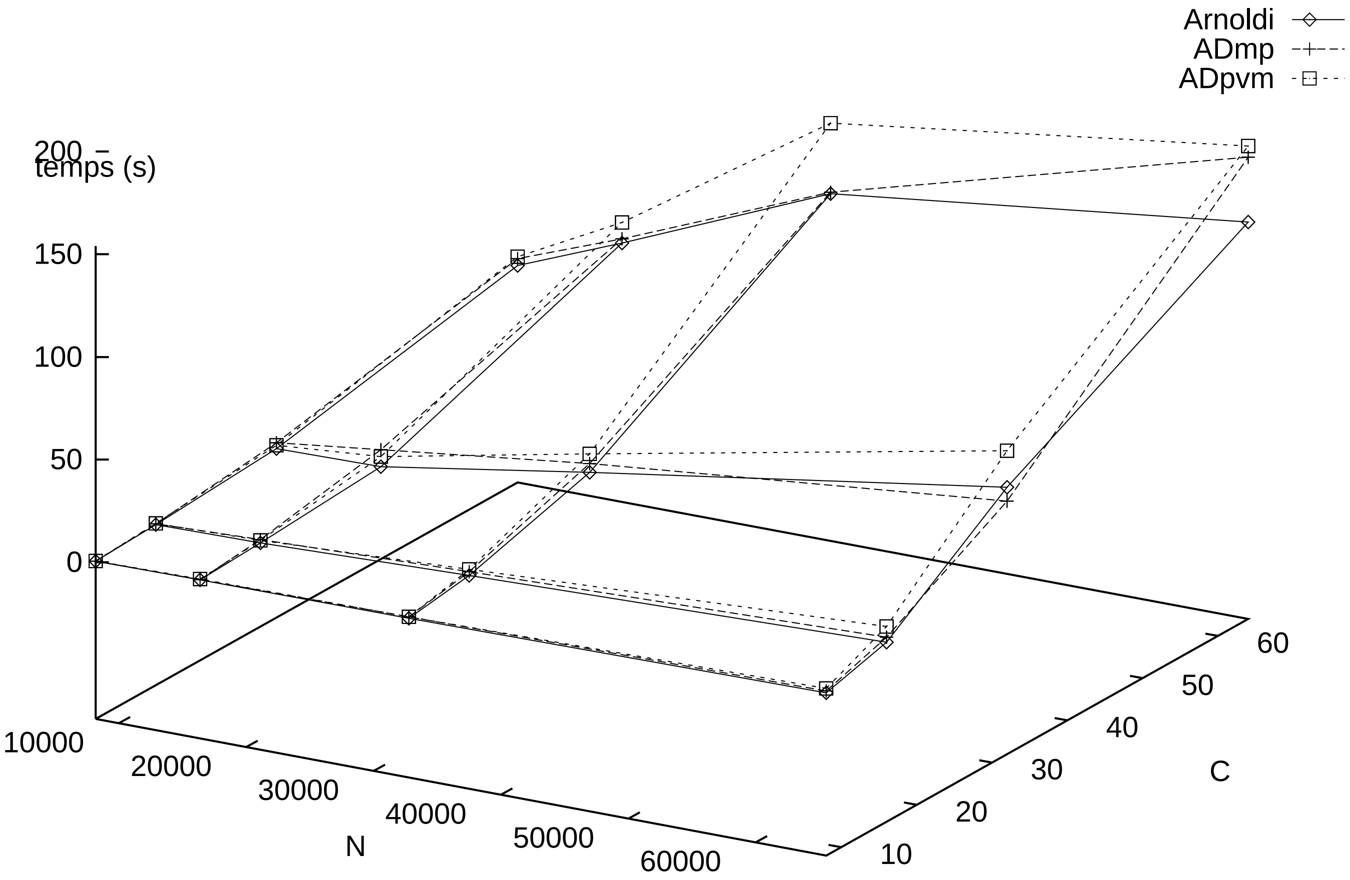


FIG. 9 - *Comparaizon between the total duration, the computation time and the communication time for one Arnoldi(r,m,l,V) iteration on the CM-5, for a C-distributed pattern matrix.*

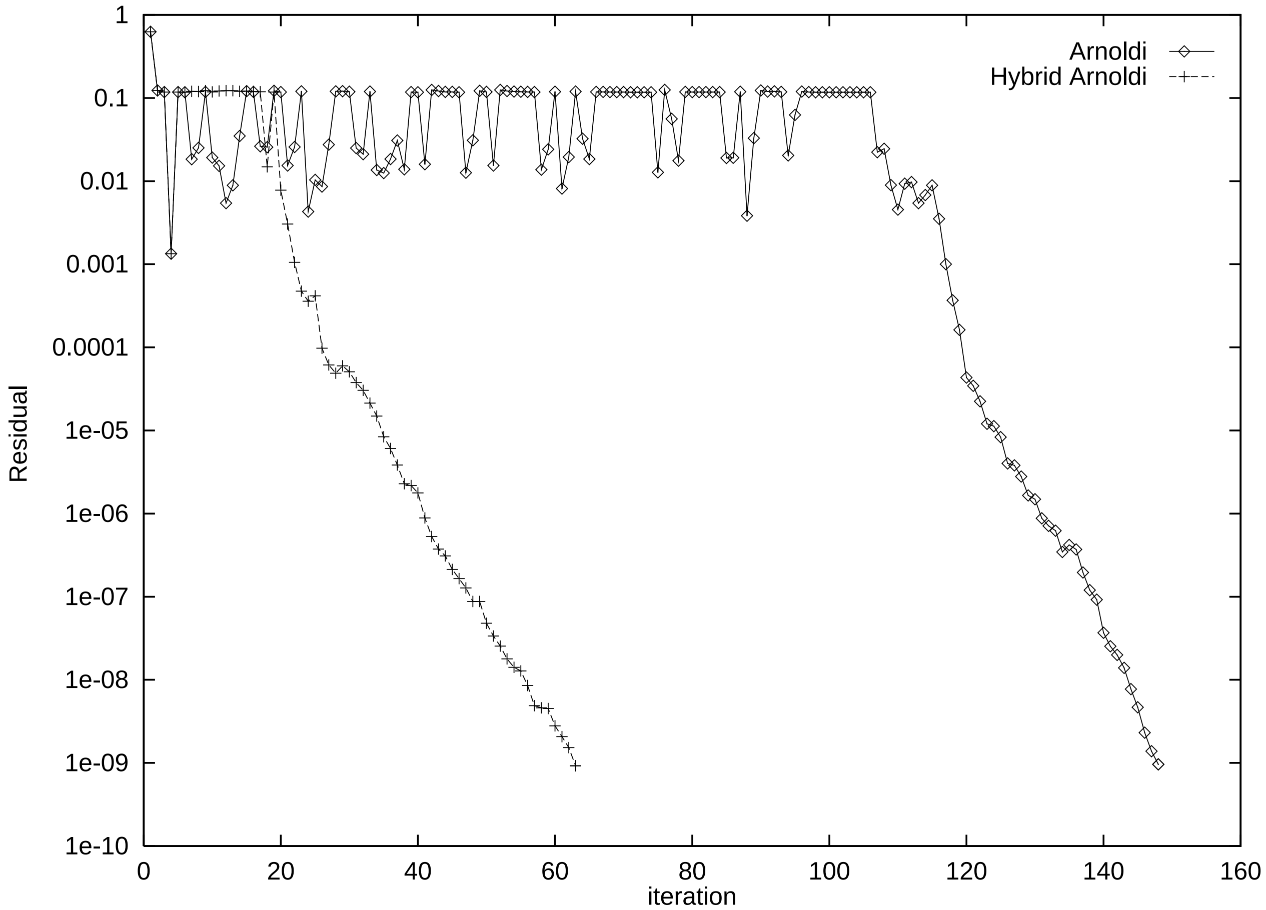
Temps CM-5 versus CM200 et CM-5 en parallèle

Le temps obtenu sur la CM-5 seule est légèrement inférieur de celui obtenu lorsque les deux Connection Machines travaillent en parallèle. Les deux temps sont très comparables.

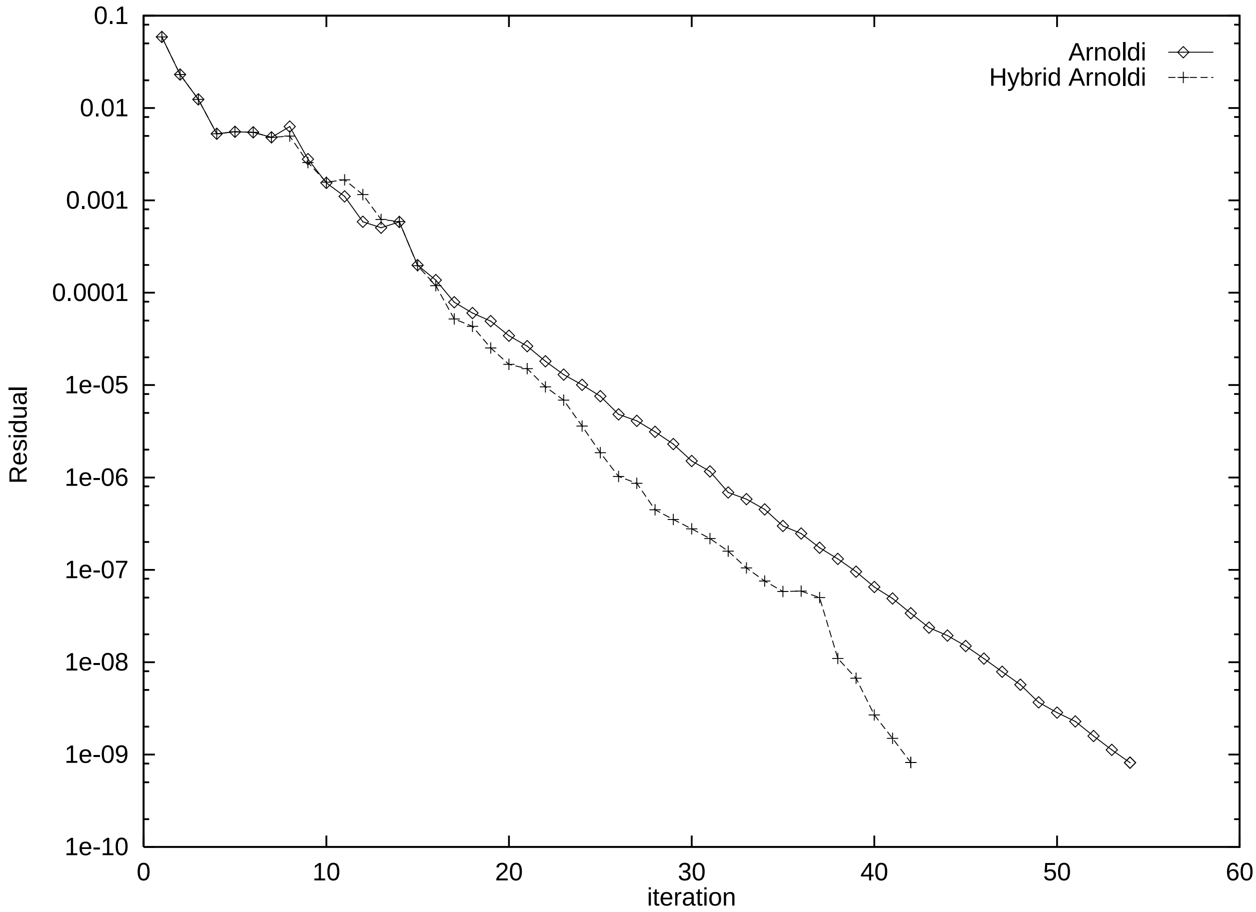


Temps d'une itération de la méthode d'Arnoldi sur la CM5, d'Arnoldi Hybride sur CM5 et CM200, avec deux stratégies de synchronisation différentes.

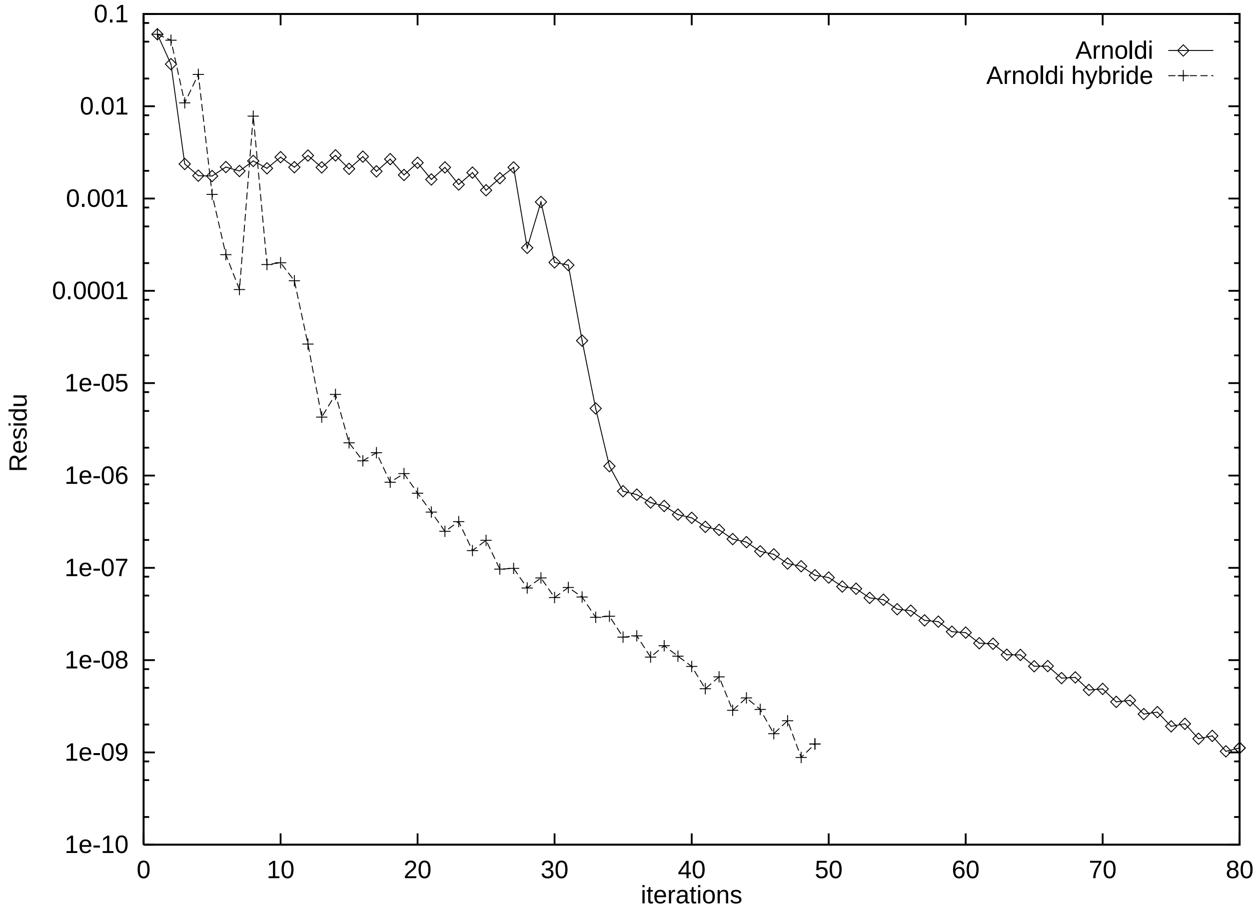
Résultats expérimentaux (convergence)



La matrice de test est `C-diagonal`, avec $N = 1024$ et $C = 21$. Quatre valeurs propres sont calculées. La taille du sous-espace est de 40. Le nombre d'itérations nécessaires pour la méthode d'Arnoldi (respectivement pour la méthode d'Arnoldi Hybride) est de 148 (respectivement 63). Pour la méthode d'Arnoldi Hybride introduite, le nombre d'itérations calculées sur la CM-200 est de 33, et le nombre d'étapes de communication entre la CM-200 et la CM-5 est de 32.



La matrice utilisée est `C-diagonal`, avec $N = 1024$ et $C = 63$. Quatre valeurs propres sont calculées. La taille du sous-espace est de 26. Le nombre d'itération nécessaire pour la méthode d'Arnoldi (respectivement la méthode d'Arnodli Hybride) est de 54 (respectivement de 42). Pour la méthode d'Arnoldi Hybride, le nombre d'itérations calculées sur la CM-200 est de 15, et le nombre d'étapes de communication entre la CM-200 et la CM-5 sont de 14.

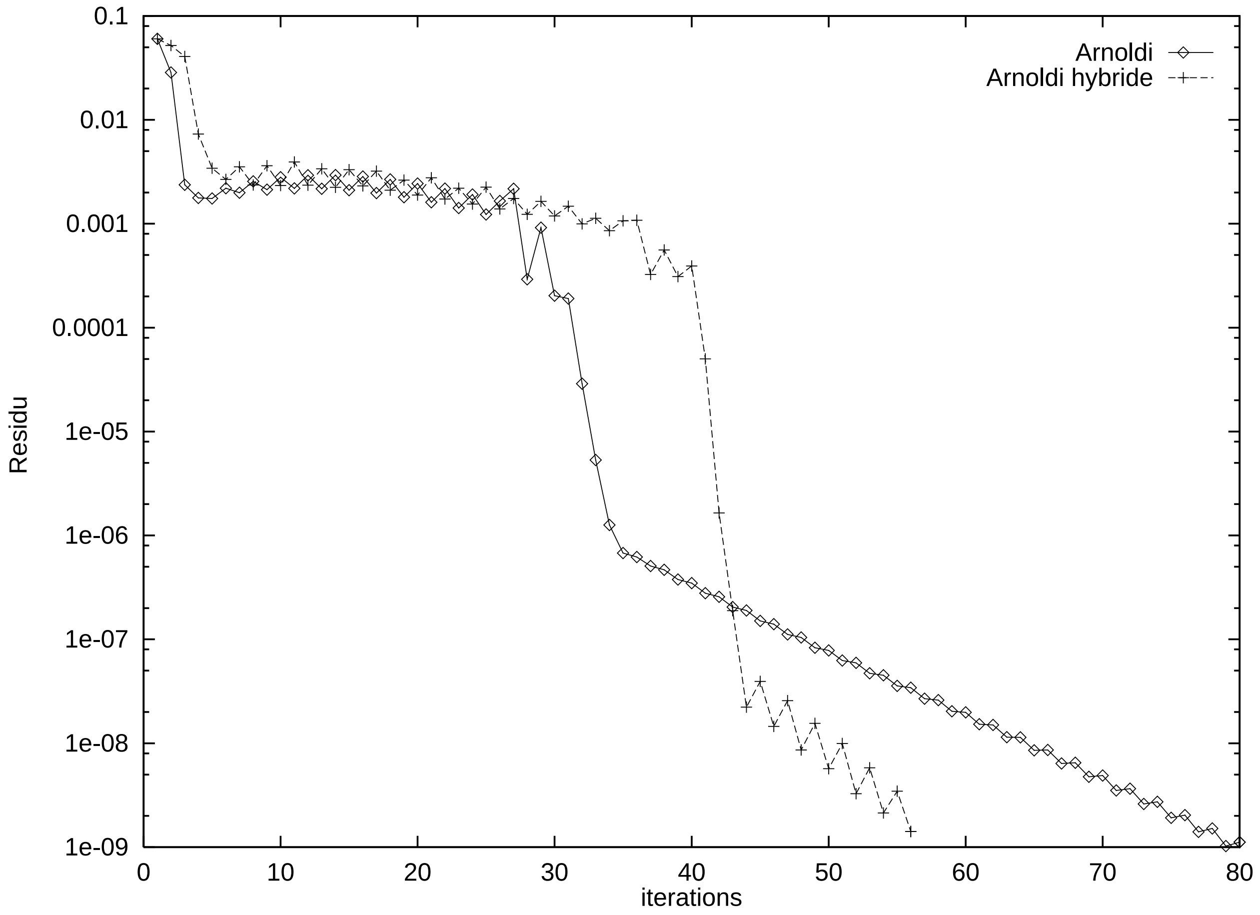


Résidu de la méthode d'Arnoldi versus celui de la méthode d'Arnoldi Hybride introduite; avec N=1024, C=21, M1=32, M2=32 et

$x_0^{CM5} = \frac{1}{N^{\frac{1}{2}}}(1, .., 1)^t$,

x_0^{CM200}=vecteur aléatoire normé,

Restart R1.

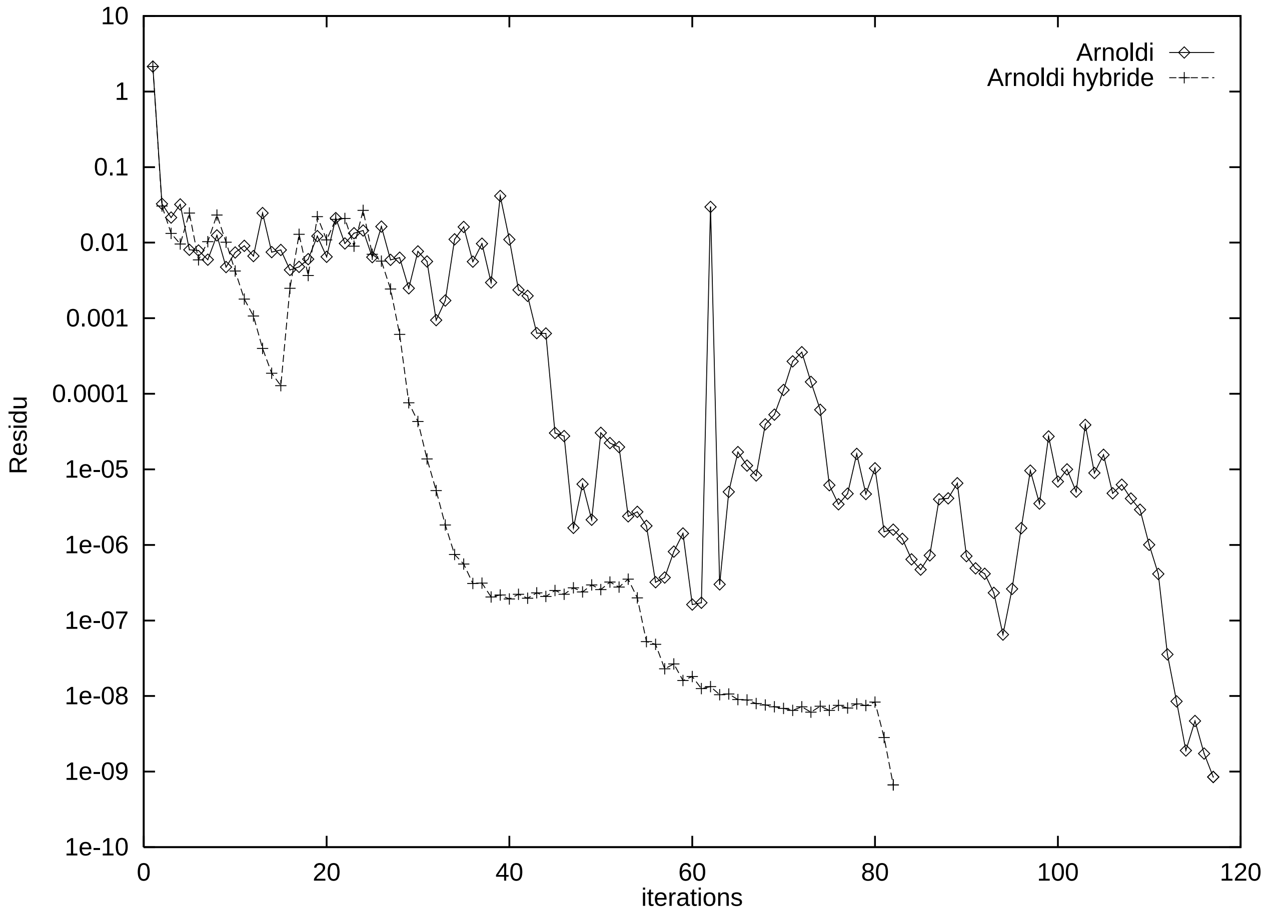


Résidu de la méthode d'Arnoldi versus celui de la méthode d'Arnoldi Hybride introduite; avec N=1024, C=21, M1=32, M2=20 et :

$x_0^{CM5} = \frac{1}{N^{\frac{1}{2}}}(1, .., 1)^t,$

$x_0^{CM200} = \frac{1}{N^{\frac{1}{2}}}(1, .., 1)^t,$

Restart R1.

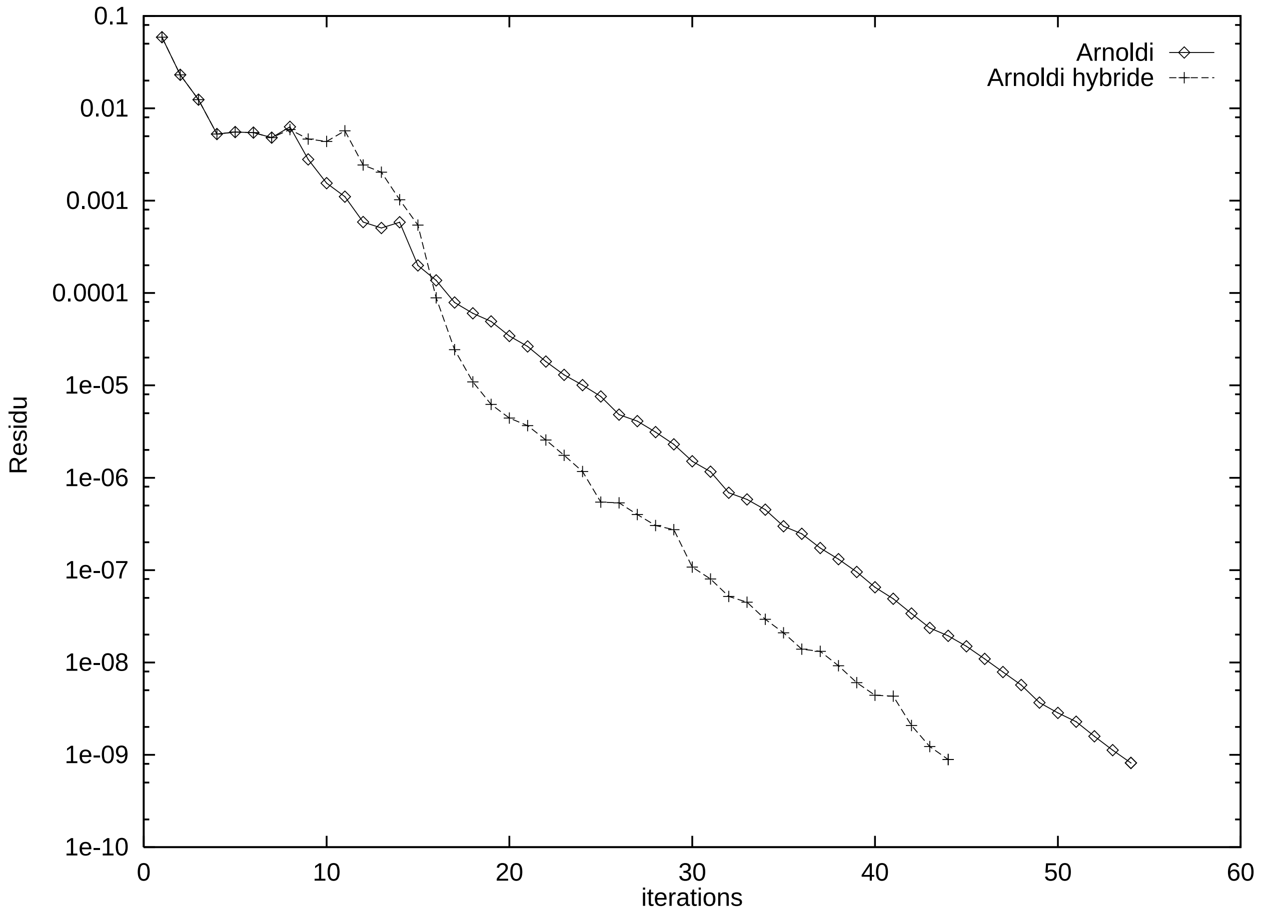


Résidu de la méthode d'Arnoldi versus celui de la méthode d'Arnoldi Hybride introduite; N=1024, C=21, M1=28, M2=15 et :

$x_0^{CM5} = \frac{1}{N^{\frac{1}{2}}}(1, .., 1)^t$,

x_0^{CM200}=vecteur aléatoire normé,

Restart R1.



Résidu de la méthode d'Arnoldi versus celui de la méthode d'Arnoldi Hybride introduite; N=1024, C=63, M1=26, M2=26 et :

$x_0^{CM5} = \frac{1}{N^{\frac{1}{2}}}(1, .., 1)^t$,

x_0^{CM200}=vecteur aléatoire normé,

Restart R1.

V Arnoldi Hybride entrelacée

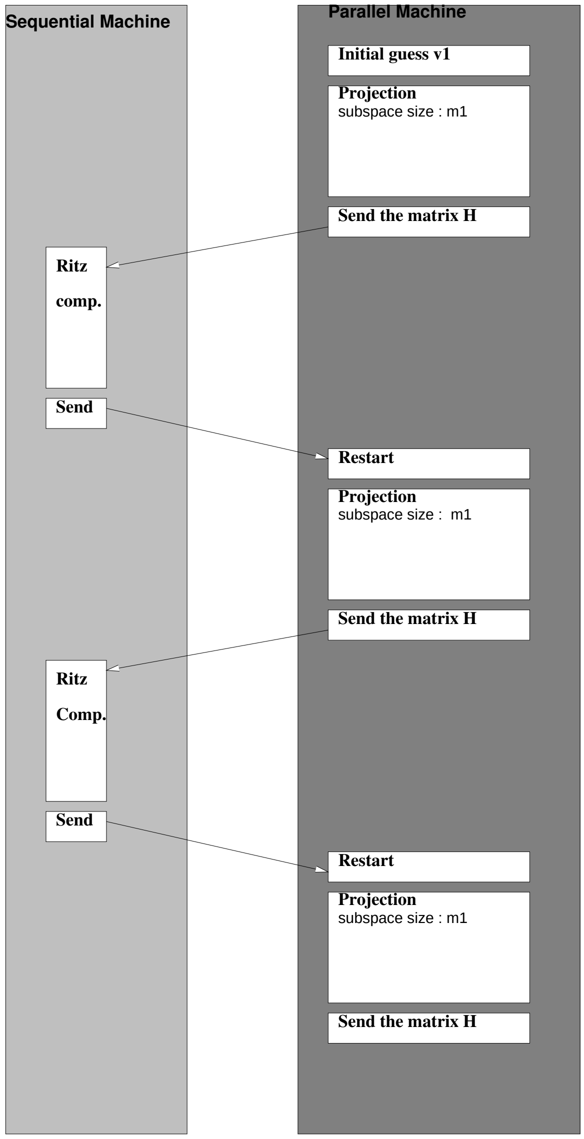


FIG. 10 - *Heterogeneous parallel hybrid Arnoldi using one sequential machine and one super-computer.*

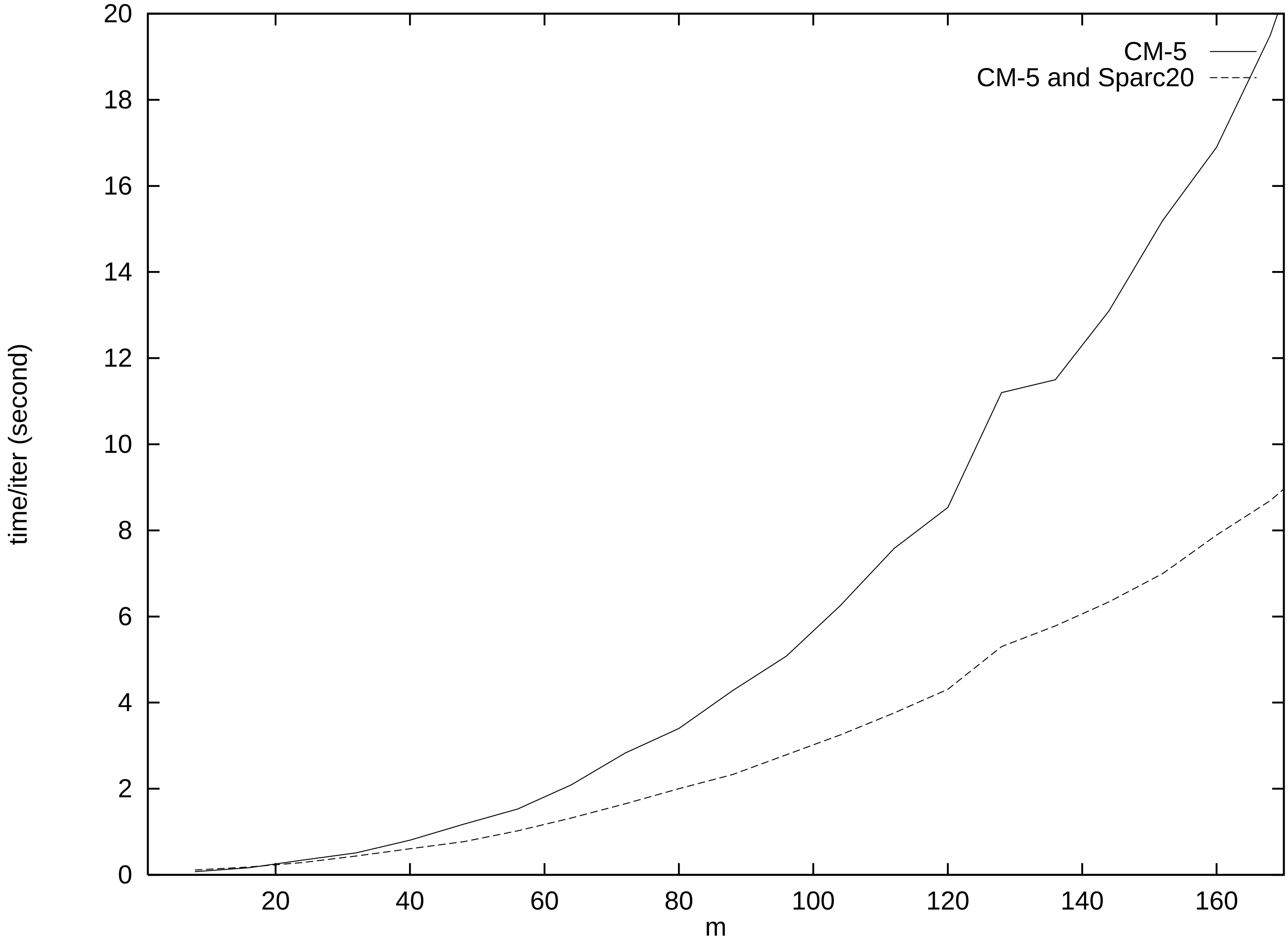


Fig. 11 - *Time in seconds for one iteration of the Parallel Arnoldi and the Heterogeneous Arnoldi for a C-diagonal matrix,* $C = 3$

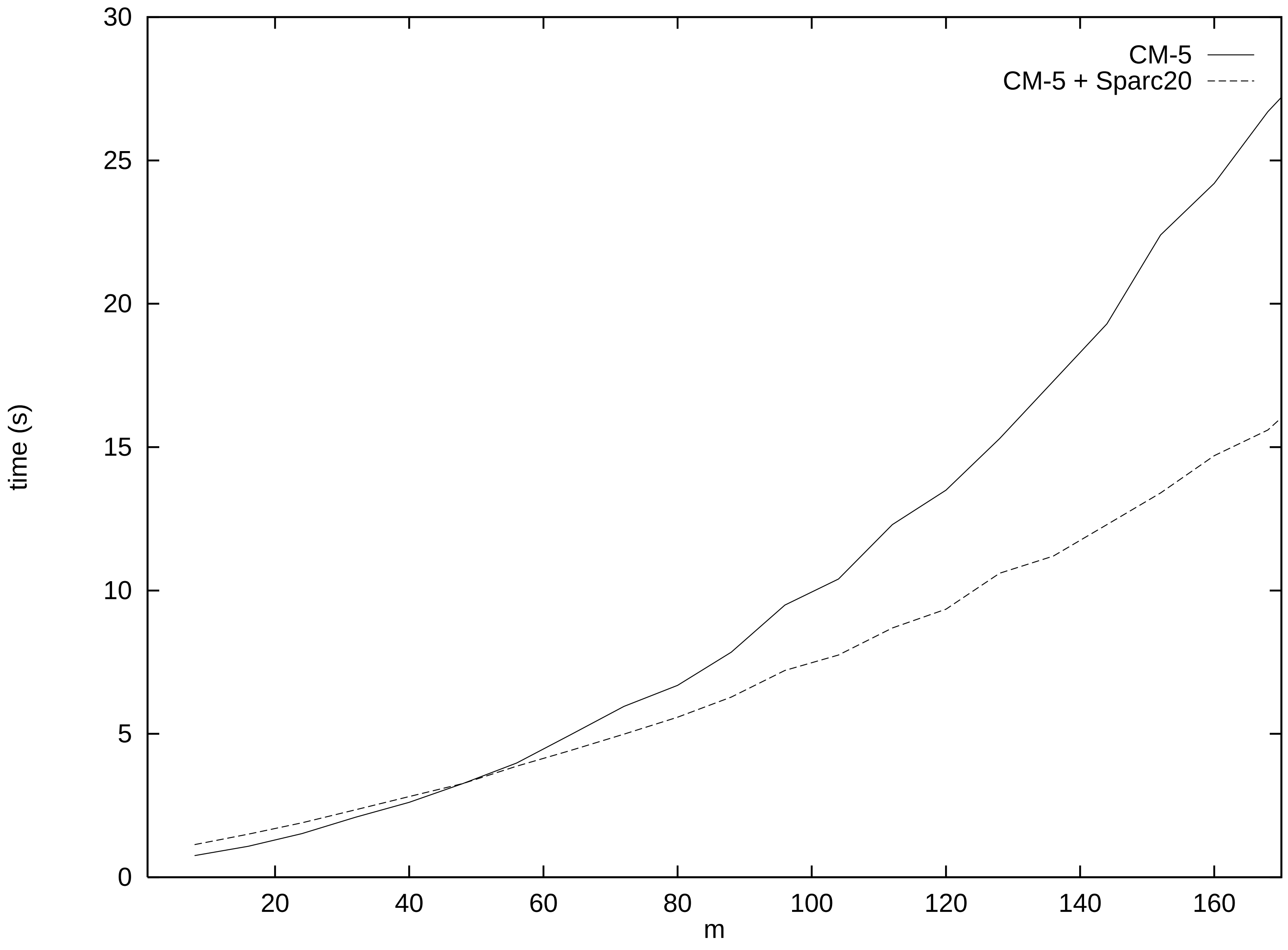


FIG. 12 - *Time in seconds for one iteration of the Parallel Arnoldi and the Heterogeneous Arnoldi for a C-diagonal matrix, $C = 63$*

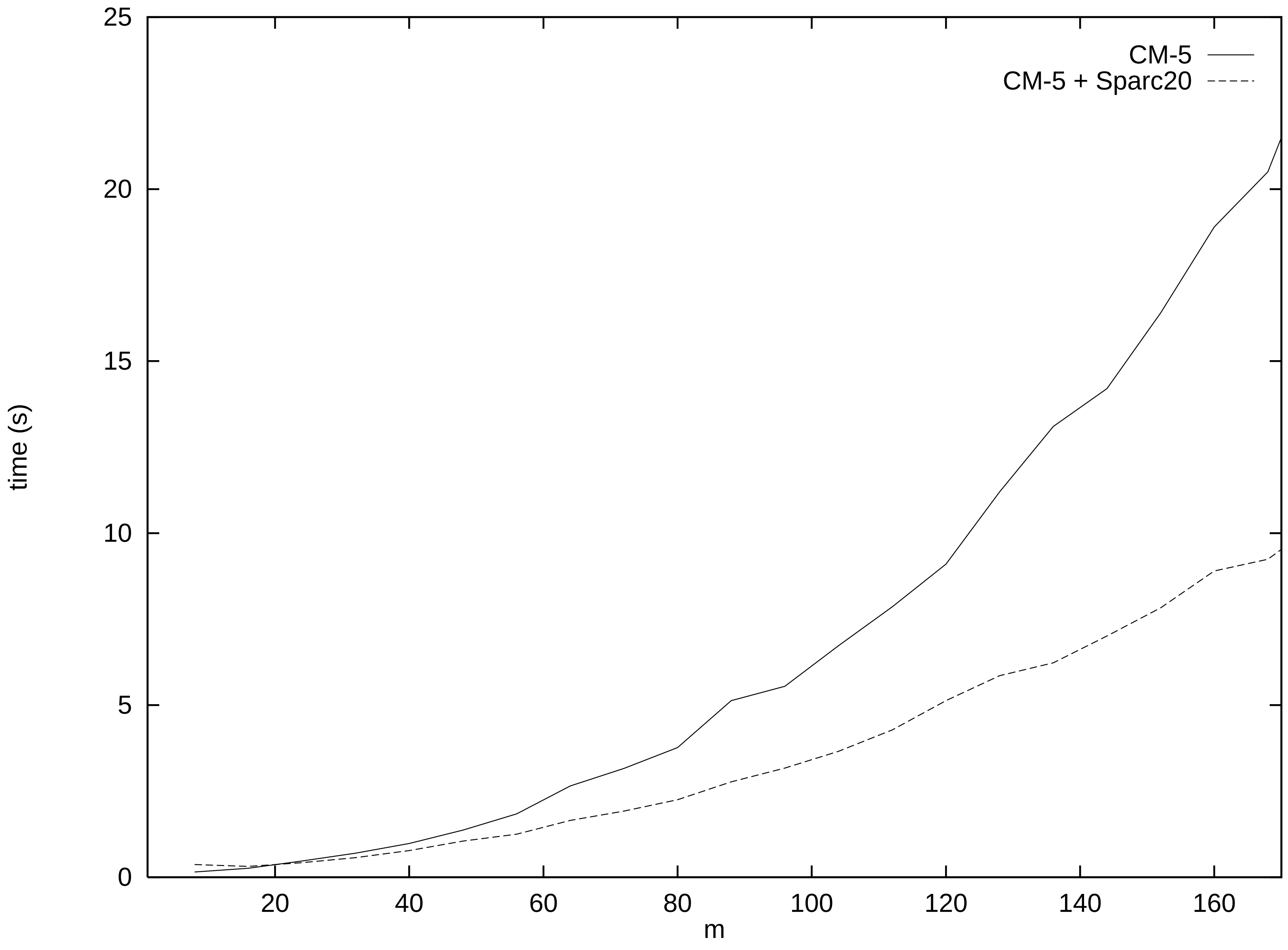


FIG. 13 - *Time in seconds for one iteration of the Parallel Arnoldi and the Heterogeneous Arnoldi for a C-distributed matrix; C=4*

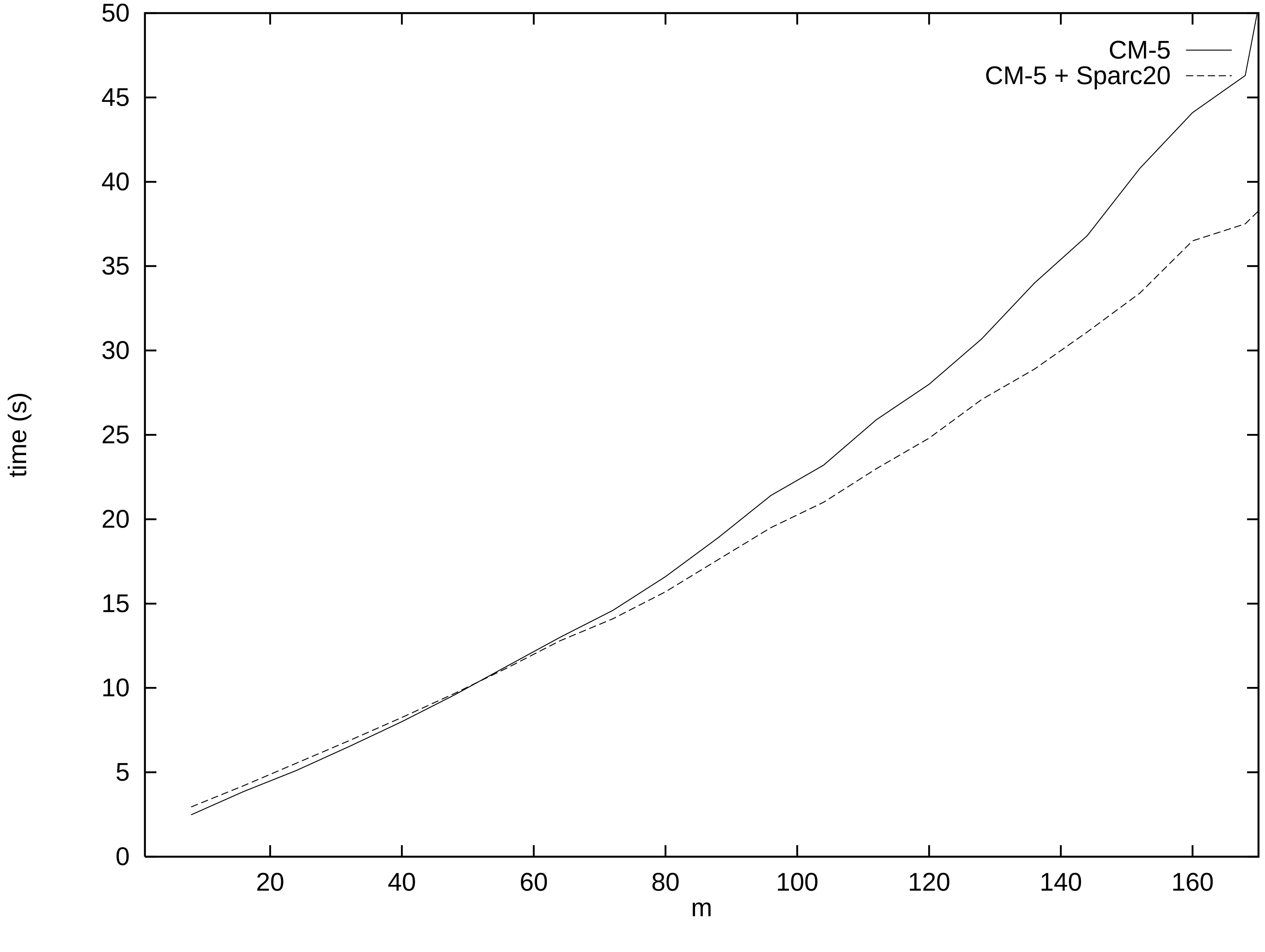


FIG. 14 - *Time in seconds for one iteration of the Parallel Arnoldi and the Heterogeneous Arnoldi for a C-distributed matrix; C=64*

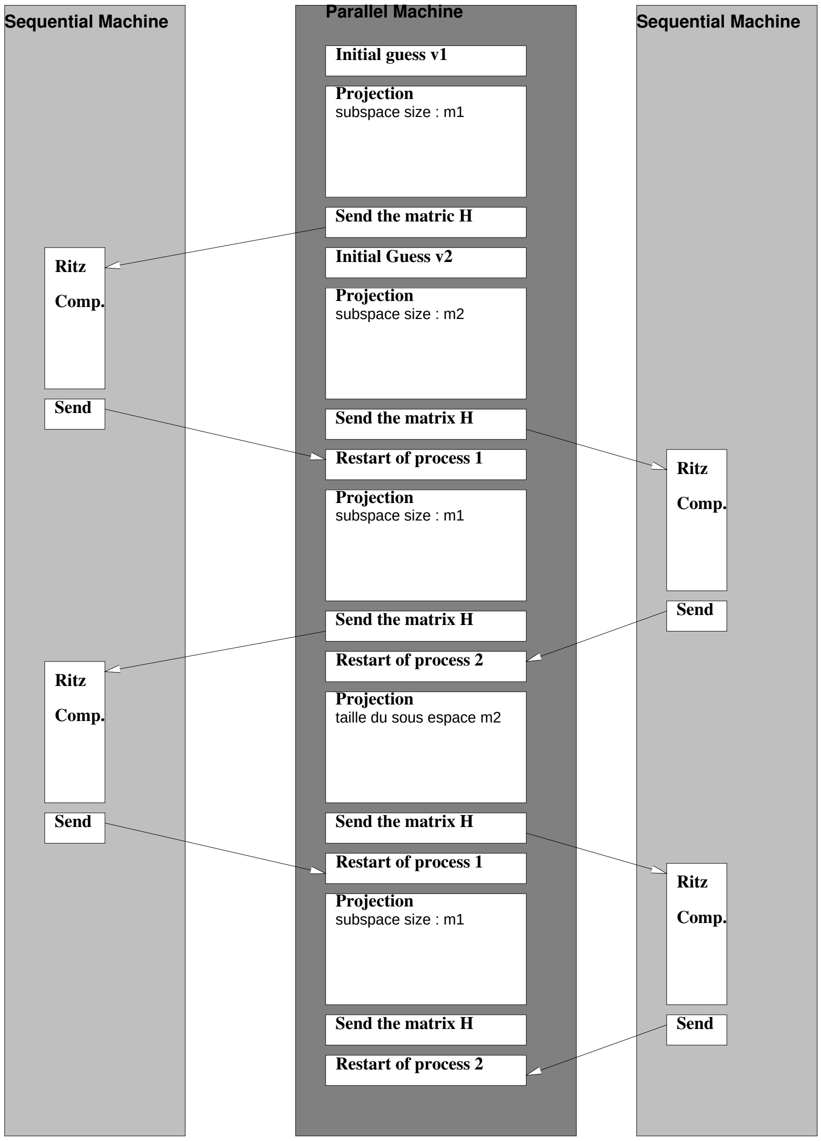


FIG. 15 - *Heterogeneous parallel hybrid Arnoldi using two sequential machines and a supercomputer.*

Choix de m_1 et de m_2

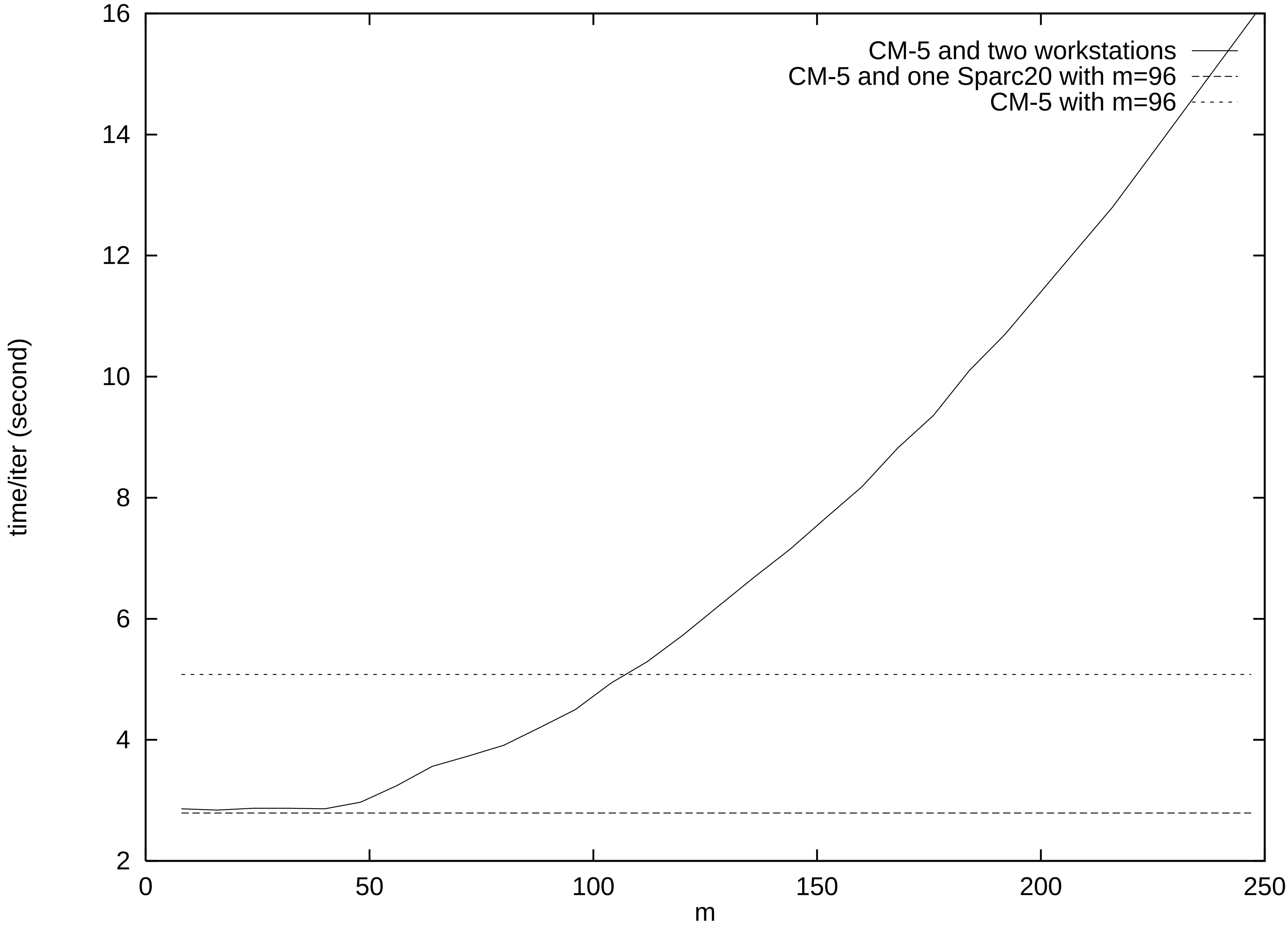


Fig. 16 - *Comparison between the Parallel Arnoldi on the CM5, the Heterogeneous Arnoldi using the CM5 and one and the Interleaved Arnoldi using the CM5 two Sparc 20 computers, for C-diagonal matrices with* $C = 3$*.*

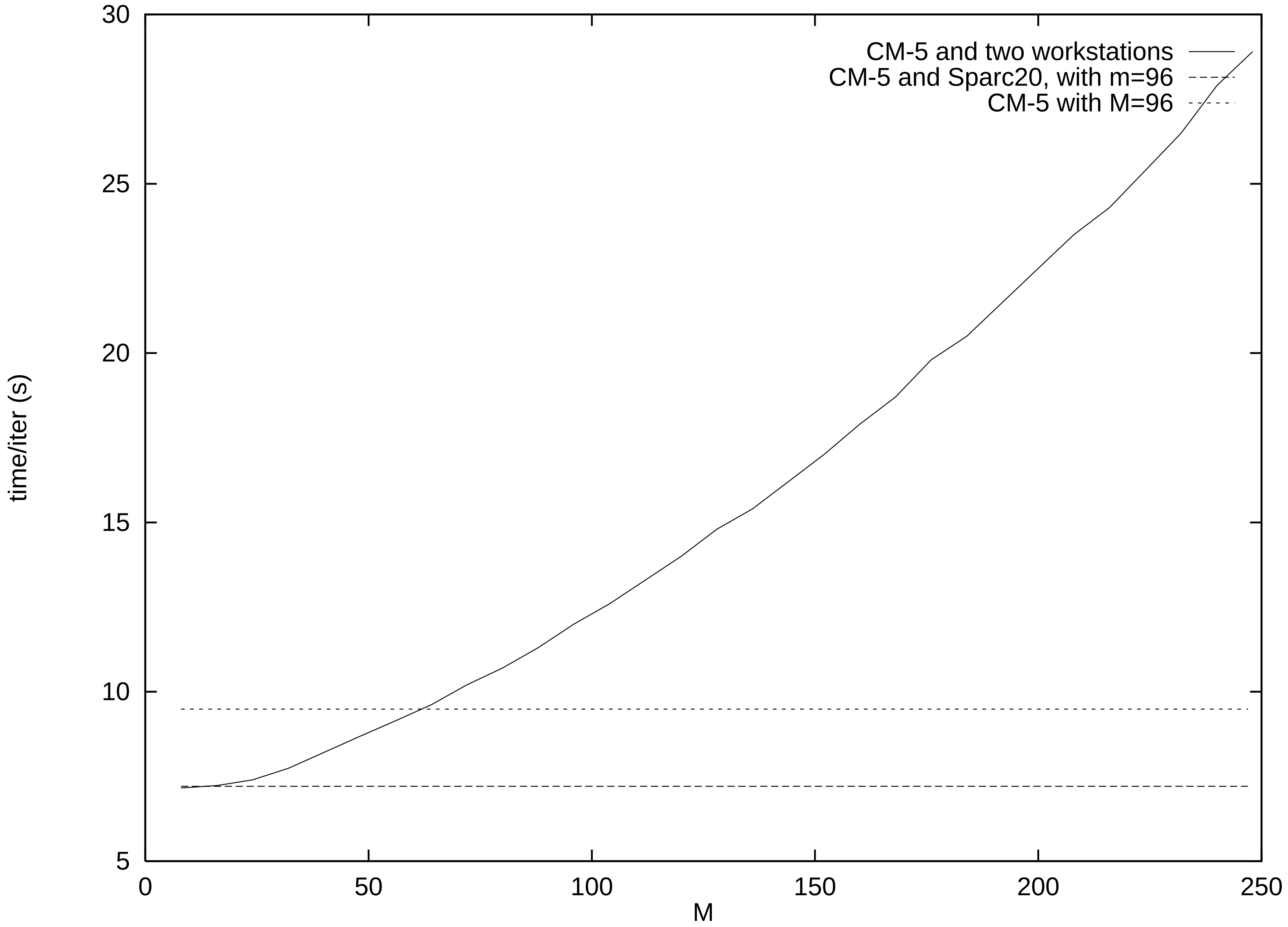


FIG. 17 - *Comparison between the Parallel Arnoldi on the CM5, the Heterogeneous Arnoldi using the CM5 and one and the Interleaved Arnoldi using the CM5 two Sparc 20 computers, for C-diagonal matrices with $C = 63$.*

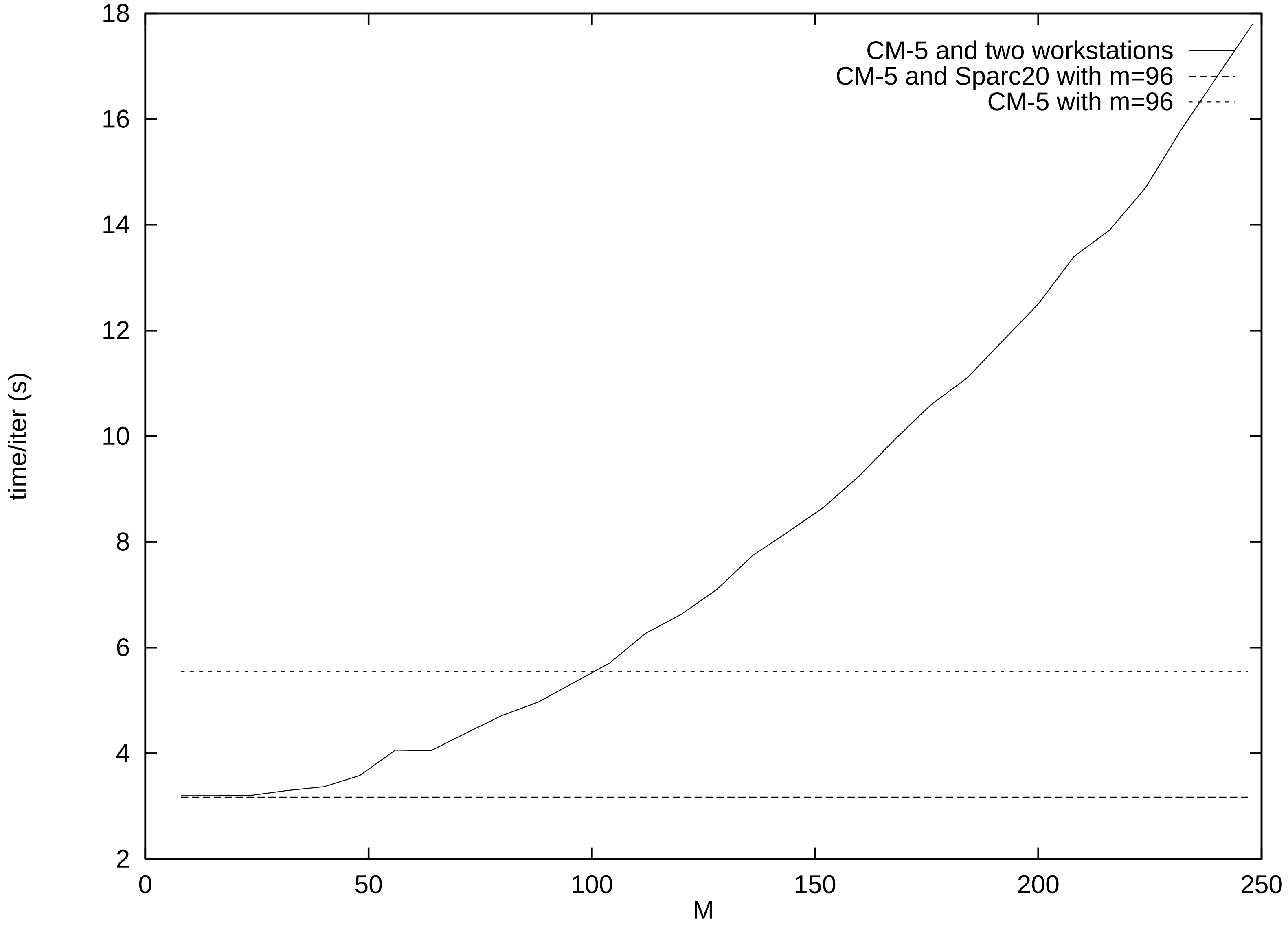


Fig. 18 - *Comparison between the Parallel Arnoldi on the CM5, the Heterogeneous Arnoldi using the CM5 and one and the Interleaved Arnoldi using the CM5 two Sparc 20 computers, for C-distributed matrices with $C = 4$.*

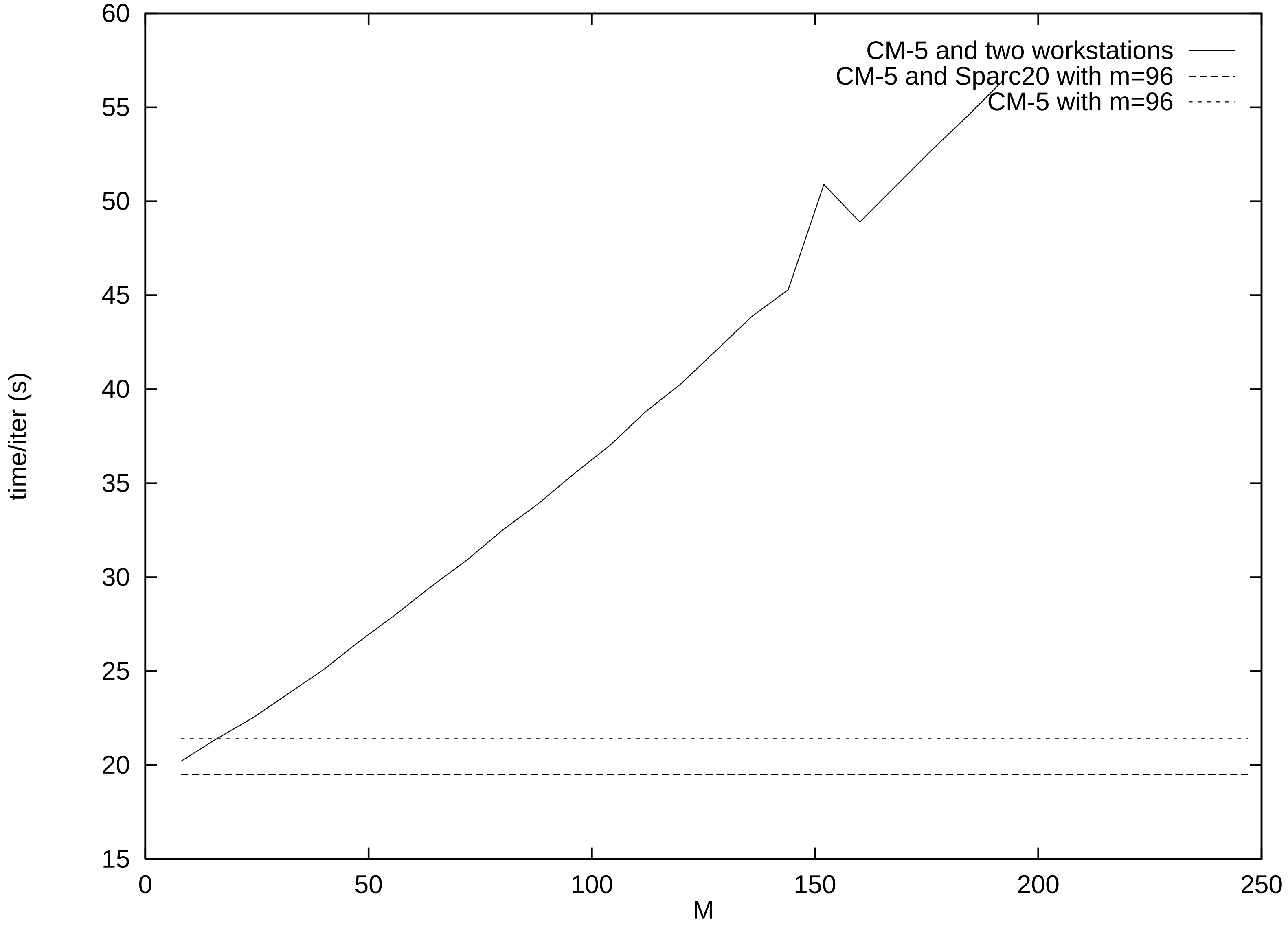


FIG. 19 - *Comparison between the Parallel Arnoldi on the CM5, the Heterogeneous Arnoldi using the CM5 and one and the Interleaved Arnoldi using the CM5 two Sparc 20 computers, for C-distributed matrices with $C = 64$.*

Il convient donc de choisir m_1 pour la convergence vers le sous espace propre recherché et m_2 pour accélérer la convergence. Mais d'autres stratégies sont à l'étude.

V Conclusion et perspectives

Il est nécessaire d'avoir des Processeurs de Contrôle bi-processeurs et des liaisons Hippis pour connecter les machines parallèles concernées.

Nous avons une bonne efficacité des calculs sur chaque machine ; i.e. une bon taux de calcul et un débit important sur chacune des machines utilisées (en fonction du type de parallélisme de chaque partie de l'algorithme).

Les calculs sont réalisés sur les machines les mieux adaptées (ex : calculs de Ritz sur les processeurs de contrôle et projections sur les machines massivement parallèles).

Nous pensons obtenir une accélération de la convergence liée aux calculs hybrides ; cela dépend des vecteurs initiaux et des stratégies de redémarrage. Beaucoup de tests sont encore nécessaires; à partir de plusieurs matrices et applications

Il est souvent possible d'utiliser les ressources de calcul, parallèles ou non, disponibles sur un réseau, pas forcement local, pour accélérer la convergence. Après avoir bien réparti les calculs selon leurs meilleurs modes de calcul.

Études en cours :

FGMRES+Arnoldi-hybride Avec G. Bergère et A. Essai à Lille.

BiCGStab hybride avec restart Avec C. Lecalvez à Lille, l'ETCA, Edinburgh et l'IDRIS.

Arnoldi Hybride Suite de l'étude, avec N. Emad et G.Edjlali : expérimentation et théorie.

Une réflexion plus générale sur les méthodes hybrides pour les environnements de machines parallèles hétérogènes est en cours.

Références

[1] C. Brezinski et M. R. Zaglia, *A hybrid procedure for solving linear systems*, Tech. Rep. ANO-283, USTL, Septembre 1992.

[2] G. Edjlali, N. Emad, and S. Petiton, *Hybrid Method on Network of Heterogeneous Parallel Computers*, Proceedings of the 14th IMACS World Congress, Atlanta, July 11-15, 1994.

[3] G. Edjlali, S. Petiton, and N. Emad, *Interleaved Parallel Hybrid Arnoldi Method for a Parallel Machine and a Network of Workstations*, Proceedings ISAS'96 Conference, Orlando, FL, July 1996.

[4] N. Emad, S. Petiton, and G. Edjlali, *The Iterative Arnoldi Hybrid Method*, To appear.

[5] S. Petiton, *Parallel Subspace Method for non-Hermitian Eigenproblems on the Connection Machine*, Applied Numerical Mathematics Journal 10, North Holland, 1992.

[6] S. Petiton and G. Edjlali, *Data parallel structures and algorithms for sparse matrix computation*, in Advances in Parallel Computing, North-Holland, 1993.

[7] S.Petiton, G.Edjlali, *Krylov Subspace Methods on Multi-Connection Machines Environnement*, Proceedings of the Second European CM Users Meeting, pp 17-28, Jean-Michel Alimi et al. Editors, Meudon, France, 1993.

[8] S. Petiton, Y. Saad, K. Wu, and W. Ferng, *Basic sparse matrix computations on the cm-5*, International Journal of Modern Physics C, 4, 1993.

[9] J. Saltz, S. Petiton, H. Berryman, and A. Rifkin, *Performance effects of irregular communications patterns on massively parallel multiprocessors*, Journal of Parallel and Distributed Computing, Academic Press, 1990.