Projet Krylov - Rapport

BOUKRAICHI Hamza - 2INH

May 22, 2018

Contents

1 Introduction

Le but de ce projet est d'implanter en Matlab les variantes redémarrée et à fenêtre glissante de la méthode GMRES préconditionnée. LE critère d'arrêt sera basé sur une erreur inverse en veillant à ne pas recalculer explicitement le résidu associé à chaque itéré. Le but de ce rapport est alors de rappeler les principes des différentes variantes de la méthode mais aussi présenter différents résultats empiriques et leur analyse ainsi que les choix d'implémentation pour lesquels j'ai opté.

2 Rappel des méthodes

Le but de la méthode GMRES est de résoudre le système linéaire $Ax = b$ de manière itérative. Son principe repose sur la construction d'une base orthornormale de l'espace de Krylov $\mathbf{K_m} = Vect(b, Ab, ..A^{m-1}b)$ qui est effectuée selon la procédure d'Arnoldi.

2.1 pGMRES

Rappelons le pseudo-code de la variante préconditionnée à gauche. :

Algorithm 1 Left Preconditionned GMRES

Set the initial guess x_0
Compute $r_0 = M^{-1}(b - Ax_0)$
$\beta = \|r_0\|$
$v_1 = r_0/\beta$;
normR $= \beta$
normRHS $= \|M^{-1}b\|$
$j = 0$
convergence = false
while (not convergence) **and** ($j <$ max_it) **do**
  $j = j + 1$
  $w = M^{-1}Av_j$
  for $i = 1, \ldots, j$ **do**
    $h_{i,j} = v_i^T w$
    $w = w - h_{i,j}v_i$
  end for
  $h_{j+1,j} = \|w\|$
  $v_{j+1} = w/h_{j+1,j}$
  Solve the least-squares problem $y_j = \arg\min \|\beta e_1 - \bar{H}_j y\|$
  $x_j = x_0 + V_j y_j$
  Compute $normR = \|M^{-1}(b - Ax_j)\| = \|\beta e_1 - \bar{H}_j y_j\|$
  convergence = (normR / normRHS) $\leq \epsilon$
end while

Cette première version, bien qu'elle garantie la convergence, elle ne tient pas assez compte du coût de calcul et du coût mémoire. Néanmoins le principe de préconditionnement est introduit ce qui permet de réduire le coût de calcul à condition du choix de préconditionneur adéquat.

2.2 Restarted-GMRES

L'idée de cette version est de fixer la taille maximum de l'espace de Krylov m. Ainsi la méthode GMRES est lancée plusieurs fois jusqu'à atteindre la précision souhaitée, elle est interrompu quand la taille de l'espace a atteint la taille maximum tout en gardant l'information générée, puisque l'itéré initial de l'exécution suivante et l'itéré final de l'exécution précédente. Garantissant ainsi la convergence de la méthode tout en réduisant la taille de l'espace manipulé.

2.3 dq-GMRES

Le but d'introduire cette variante est le fait de maintenir le caractère orthogonale des vecteurs de la base de l'espace de Krylov uniquement sur les m derniers vecteurs, mais aussi exploiter le caratère creux de la matrice issu de la procèdure d'Arnoldi en appliquant des rotations de Givens pour la résolution du système linéaire.

Rappelons alors le pseudo-code de la variante à fenêtre glissante. :

Algorithm 2 DQGMRES(m)

Set the initial guess x_0
$r_0 = b - Ax_0$; $\gamma_1 = \|r_0\|$
$v_1 = r_0/\gamma_1$;
for $j = 1, 2, \ldots, itmax$ **do**
  % Step 1 : Gram-Schmidt with orthogonality maintened among the last m vectors
  $w_j = Av_j$
  for $i = max(1, j - m + 1), \ldots, j$ **do**
    $h_{i,j} = v_i^T w_j$
    $w_j = w_j - h_{i,j} v_i$
  end for
  $h_{j+1,j} = \|w_j\|$
  $v_{j+1} = w_j / h_{j+1,j}$
  % Step 2 : Update the QR factorization of $\bar{H}_j$
  for $i = max(1, j - m), \ldots, j - 1$ **do**
    Apply $\mathcal{Q}_i$ to the last colum of $\bar{H}_j$
  end for
  % Step 3 : Apply $\mathcal{Q}_j$ to $\bar{H}_j$
  Compute c_j and s_j
  Compute $\gamma_{j+1} = -s_j \gamma_j, \gamma_j = c_j \gamma_j$
  Compute $h_{j,j} = c_j h_{j,j} + s_j h_{j+1,j}$
  % Step 4 : Update the iterate
  $p_j = h_{j,j}^{-1}(v_j - \sum_{i=max(1,j-m)}^{j-1} h_{i,j} p_i)$
  $x_j = x_{j-1} + \gamma_j p_j$
  if Converged **then**
    Stop
  end if
end for

3 Implémentation et choix de programmation

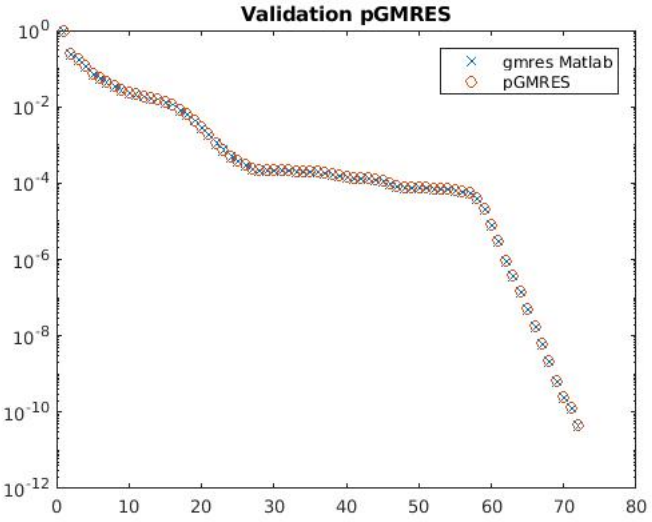
Les choix de programmation les plus pertinents se situent autour de la question de la gestion de l'espace mémoire :
Sur la variante de dqGMRES seuls les 2 derniers éléments de γ ont été stocké, mais aussi au niveau des matrices de rotations où il suffisait de stocker les vecteurs c et s au lieu des matrices. Pour être plus précis le stockage de l'un des vecteurs étant suffisant étant donné que $c^2 + s^2 = 1$, mais par soucis de clarté du code j'ai choisi de stocker les 2 vecteurs.

Des choix pertinents ont aussi été fait au niveau de l'optimisation du temps de calcul afin de suivre le principe énoncé en introduction à propos du calcul de la norme du résidu. Ainsi cette norme a été calculée selon les formules vues et démontrées en cours.

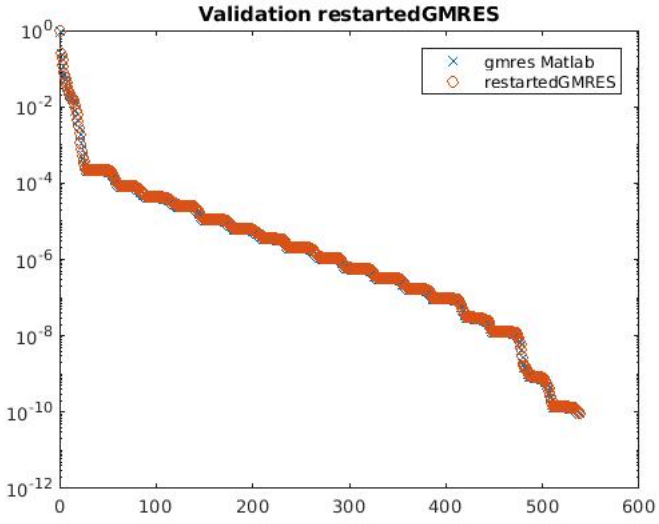
4 Tests et analyses des résultats

4.1 Validation des solveurs

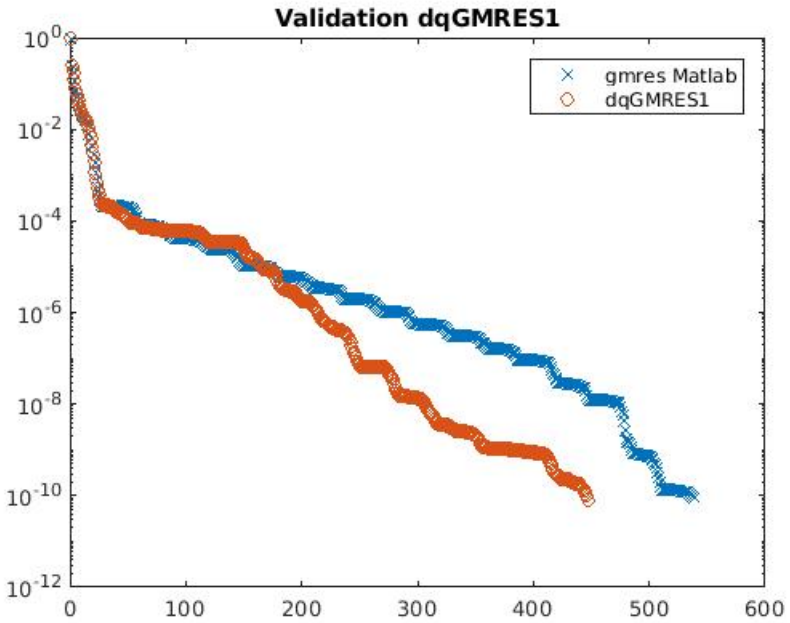
Voici les différentes courbes générées par matlab pour valider les différents solveurs en utilisant la matrice mat1.

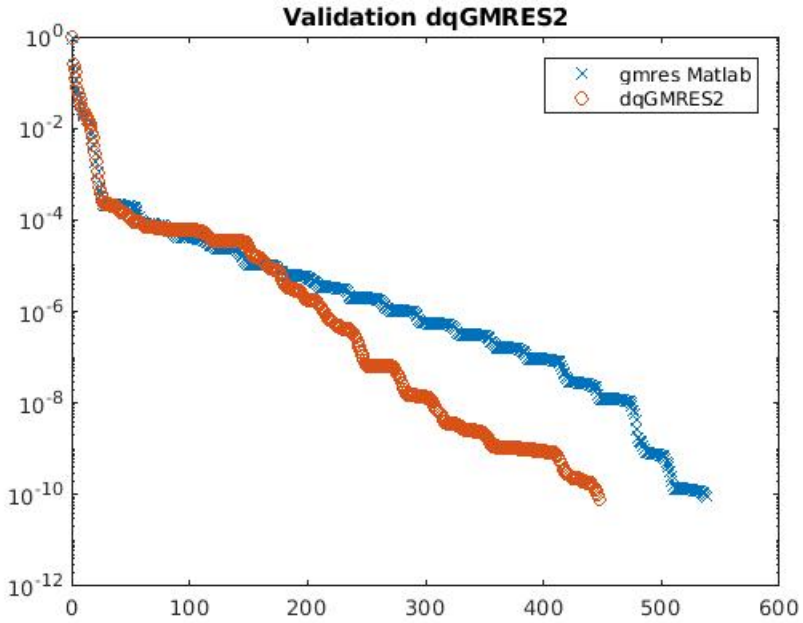


On remarque bien que les courbes de pGMRES et gmres matlab se superposent.

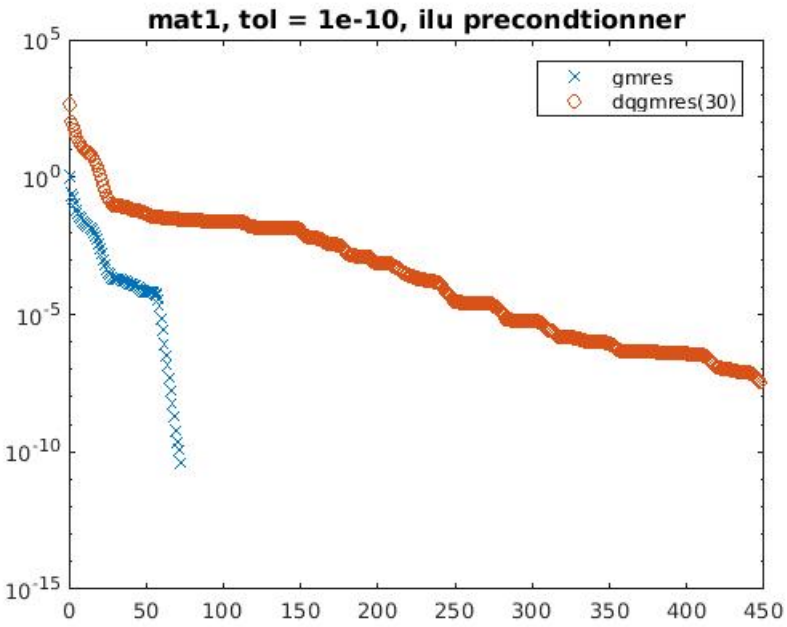


On remarque bien que les courbes de restartedGMRES et gmres matlab se superposent pour les bons choix de paramètres m et maxiter.



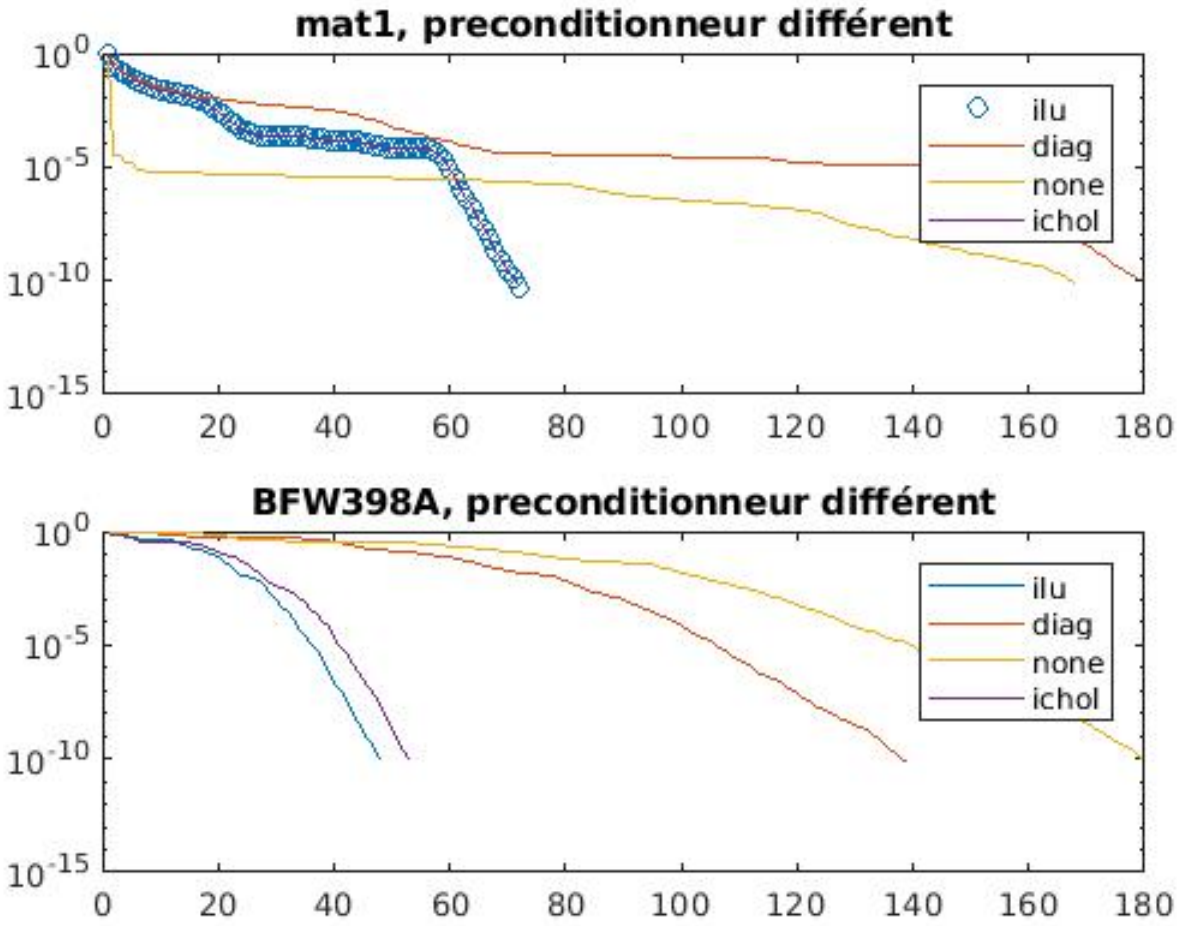


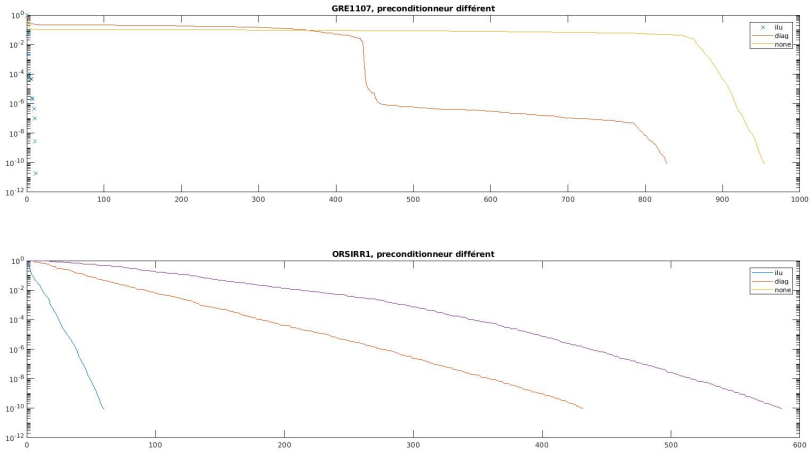
Pour dqGMRES il est tout à fait normal que les courbes ne se superposent qu'au début, étant donné que l'orthogonalisation des vecteurs n'est plus effectuée pour les m derniers vecteurs de l'espace de Krylov associé.



En comparant dqGMRES et gmresMatlab sans restart on retrouve la figure fournie, ce qui permet de valider l'implantation de la variante dqGMRES.

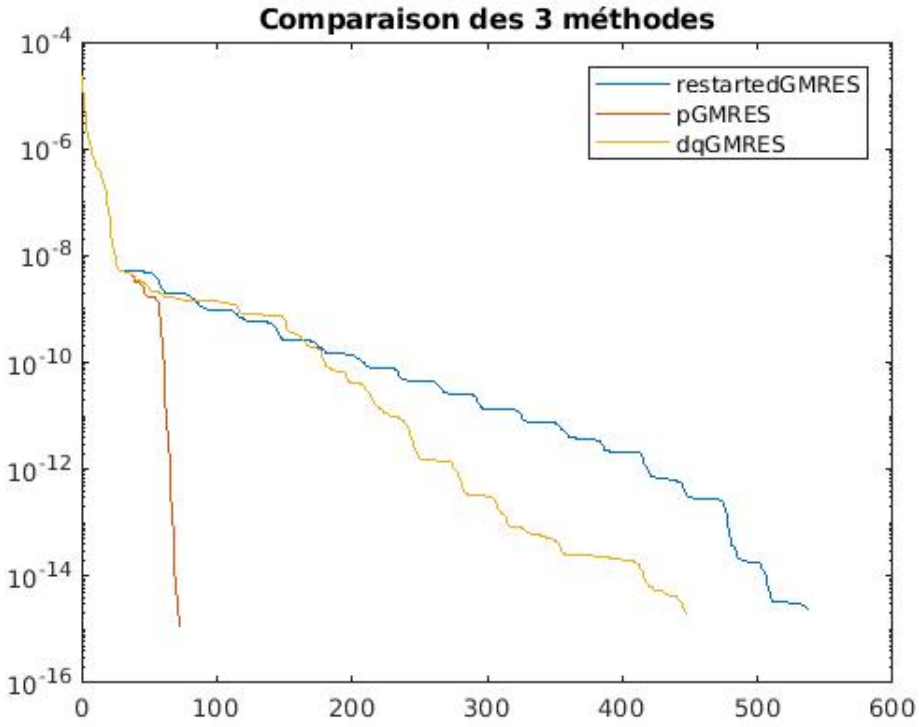
4.2 Comparaison des préconditioneurs





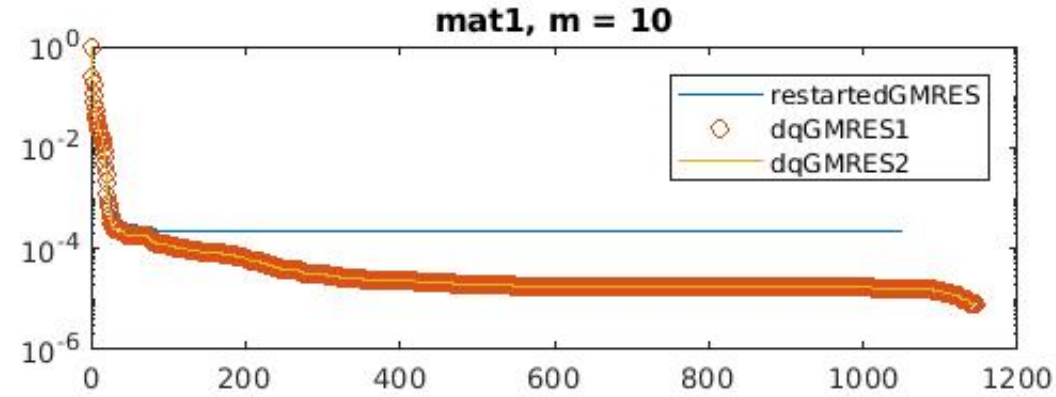
On remarque que pour mat1 la courbe pour ichol et ilu se superposent et que il vaut mieux pour cette matrice ne pas utiliser de préconditionneur plutot que le préconditionneur diagonale.
On en déduit donc que le choix du préconditionneur adéquat est fortement lié à la structure et la nature de la matrice. On remarque aussi que l'effet du préconditionneur est plus apparent au niveau des matrices de plus grandes tailles, néanmoins il est assez clair que dans la plupart des cas le préconditionneur ilu est plus performant et permet de réduire considérablement le coût de calcul des méthodes.

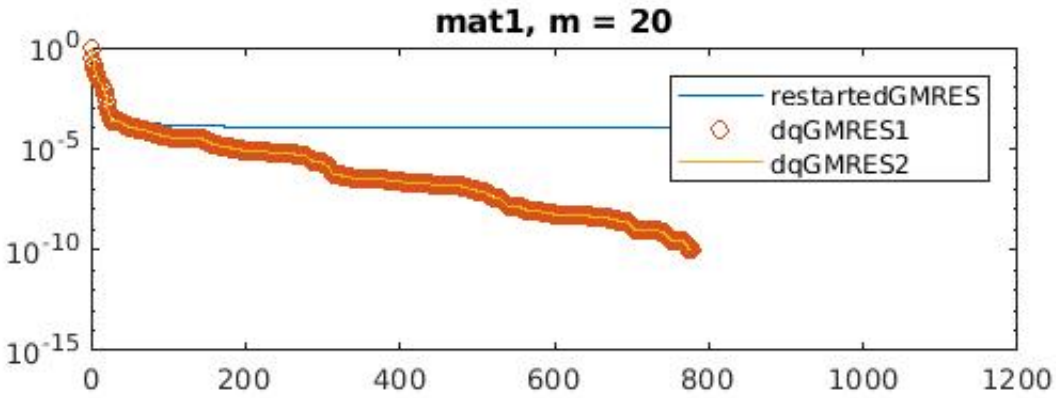
4.3 Comparaion des solveurs

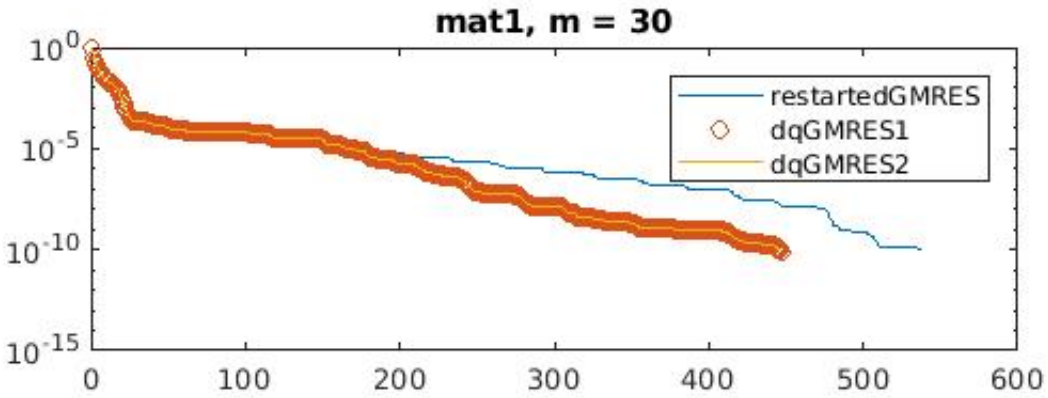


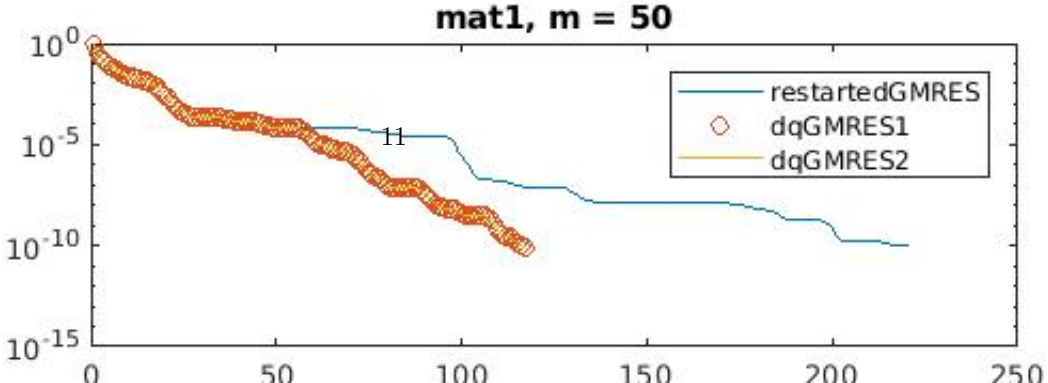
On remarque que les variantes redémarrée et à fenêtre glissante prennent plus d'itérations à converger. Ce qui est tout à fait normal étant donnée que considérer uniquement certains vecteurs de l'espace de Krylov réduit la précision du résultat issu de chaque itération.

4.4 Effet de la taille de la fenêtre









On remarque que le fait d'augmenter la taille de la fenêtre permet une convergence plus rapide des variantes dqGMRES et restartedGMRES, ce qui est un résultat assez attendu étant donné qu'augmenter la taille de la fenêtre revient à considérer plus de vecteur de l'espace de Krylov ce qui implique à son tour une meilleur précision. Néanmoins augmenter la taille de la fenêtre augmente aussi le coût mémoire de la méthode, alors il faudra trouver une taille de fenêtre optimal permettant de minimiser le coût du calcul et le coût mémoire tout en gardant une bonne précision.

5 Conclusion

Après analyse des résultats empiriques on peut remarquer les variantes de GMRES codées, permettent évidemment de réduire le coût mémoire de la résolution du système linéaire, néanmoins le coût temporel quant à lui augmente. Il faudra donc trouver un compromis afin d'éviter que l'un des 2 coûts n'explose. Le choix de la taille de la fenêtre dépendra du système (taille de la matrice et sa structure) mais aussi de la machine sur laquelle les méthodes seront exécutées. Un autre aspect à ne pas négliger est le choix adéquat du préconditionneur, qui dépend également de la taille et la structure de la matrice, étant donnée qu'un bon choix permet de réduire considérablement le temps de calcul.