Middle-East Journal of Scientific Research 8 (2): 536-540, 2011
ISSN 1990-9233
© IDOSI Publications, 2011

A Numerical Comparison of Two Different Implementations of GMRES Method

[1]*H. Saberi Najafi and* [2]*H. Zareamoghaddam*

[1]Lahijan Branch, Islamic Azad University, Lahijan, Iran
[2]Kashmar Branch, Islamic Azad University, Kashmar, Iran

Abstract: In this article, we compare two different implementations of GMRES, namely [1, 2]. These methods do not use Given's rotations. Some numerical examples are presented to show the difference of the processes.

Key words: Arnoldi process . AGMRES . SGMRES . linear systems . Krylov subspaces

INTRODUCTION

One of the important computational method for solving a linear system of equations

$$Ax = b \tag{1}$$

where A is a nonsymmetric, $n\times n$, matrix, is the GMRES method. This method has been developed in 1986 by Saad and Schultz [3]. In this method by using the Arnoldi process an orthogonal basis $V = \{v_1, v_2, \ldots v_k\}$ is made for krylov subspace

$$K_k(r_0, A) \equiv \text{span}\{r_0, Ar_0, \ldots A^{k-1}r_0\}$$

where x_0 is an initial vector and $r_0 = b - Ax_0$. The approximation solution in kth step is $x_k = x_0 + V_k y$ where y minimizes the least square problem.

For the rest of article we use the two following algorithms:

Algorithm 1: Arnoldi-modified Algorithm

```
0.  Choose a vector $v_1$ such that $\|v_1\| = 1$.
1.  for l = 1,…,m do
        $v_{k+1} = Av_k$,
        for i = 1,…,k do
            $h_{i,k} = \langle v_{k+1}, v_i \rangle$,
            $v_{k+1} = v_{k+1} - h_{i,k}v_i$,
        End.
        $h_{k+1,k} = \|v_{k+1}\|$,
        $v_{k+1} = v_{k+1}/h_{k+1,k}$,
    End.
```

In this algorithm by using a unit vector v_1, the k orthonormal vectors generates for $K_k(r_0,A)$ and therefore we have $AV_k = V_{k+1}\bar{H}_k$ which is the base of GMRES method. $\bar{H}_k$ is the $(k+1)\times k$ upper Hessenberg matrix obtained by $h_{i,k}$ and in GMRES method we try to minimize $\|e_1 - \bar{H}_k z\|$ where $z\in R^k$ and $e_1 = (1,0,\ldots,0)^T \in R^{k+1}$.
Now the algorithm is:

Algorithm 2: Restarted GMRES method (GMRES(k))

```
0.  Choose a starting point $x_0$ and compute
        $r_0 = b - Ax_0$ and $v_1 = \frac{r_0}{\|r_0\|}$
1.  Construct the orthonormal vectors $v_1, v_2, \ldots, v_k$, by Algorithm 1.
2.  Find $\bar{z} \in R^k$ which minimizes $\|e_1 - \bar{H}_k z\|$.
3.  $x_k = x_0 + \|r_0\| V_k \bar{z}$, if $x_k$ salves (1) exit, otherwise set
    $x_0 = x_k$, $r_0 = r_k$, $v_1 = r_0/\|r_0\|$ and go to 1.
```

IMPLEMENTATIONS OF SGMRES AND AGMRES

After developing the GMRES method in [3], different implementations have been introduced such as [1, 2, 4] and more research works have been done [5-8]. Brown [9] considers the GMRES as an analytical method and compares with FOM by some examples. Rozloznik [10] shows that GMRES is a stable numerical method and is a powerful method for parallel computing. The other ideas in [6, 11] describe that how we can increase the speed of convergence of GMRES. As we already described this method by using the Arnoldi projection process instead of solving a linear system with dimension n, uses Given's rotations for solving a least square problem $(k+1)\times k$. Since using Given's rotations has a high cost, so we are interested in

Corresponding Author: H. Zareamoghaddam, Islamic Azad University, Kashmar Branch, Kashmar, Iran

not using those rotations, therefore we present two different implementations of GMRES method without using Given's rotations, they are called SGMRES and AGMRES.

SGMRES: The SGMRES is known as simpler GMRES and developed by Walker and Zhou in 1994 [2]. They suggested that if the vector Ar_0 instead of r_0 uses for generating the orthonormal basis V_k we have an upper triangular system to solve which is much easier then upper Hessenberg problem. In this method the orthonormal vectors $v_1, v_2,\ldots, v_k$ generates by Algorithm 1 and r_0 make an orthogonal set with $v_1, v_2,\ldots, v_{k-1}$ by using Gram-Schmidt process. Finally the approximate solutions $x_k = x_0+d$ obtain.

Algorithm 3 shows how vector d produces. For more information [2].

Algorithm 3: Simpler GMRES (SGMRES)

```
0.  Given x, set r = b-Ax and $\rho_0 \equiv \|r\|$. If $\rho_0$<eps, accept x and exit; otherwise, update $r = \frac{r}{\rho_0}$ and
    Set $\rho = 1$.

1.  For k = 1,2,…,m do
    a. Evaluate $v_k \equiv Av_{k-1}$ ($v_1 \equiv A_r$).
    b. If k>1, then for j = 1,…, k-1 do
        i. Set $h_{j,k} = \langle v_j, v_k\rangle$.
        ii. update $v_k = v_k - h_{jk} v_j$.
    c. Set $h_{kk} = \|v_k\|_2$ update $v_k = \frac{v_k}{h_{kk}}$.
    d. Set $H_k = \begin{pmatrix} H_{k-1} & h_{1k} \\ & \\ 0\cdots 0 & h_{kk} \end{pmatrix} (H_1 = (h_{11}))$.
    e. Set $\xi_k = \langle r, v_k\rangle$; update $\rho = \rho\sin\left(\cos^{-1}\left(\frac{\xi_k}{\rho}\right)\right)$; If $\rho,\rho_0 \le$eps, go to 2.
    f. update $r = r-\xi_k v_k$.

2.  Let k be the final iteration number from 1.
    a. Solve $H_k z = (\xi_1,\ldots,\xi_k)^T$ for $z = (\eta_1,\ldots,\eta_k)^T$.
    b. Form $y = \begin{cases} \eta_1 r & , \text{ if } k = 1 \\ \eta_1 r + \sum_{i=1}^{k-1} (\eta_{i+1} + \eta_1\xi_i) v_i & , \text{ if } k > 1 \end{cases}$
    c. update $x = x+\rho_0 y$, If $\rho,\rho_0 \le$eps, accept x and exit ;otherwise, update
       $r = \frac{(r-\xi_k v_k)}{\rho}$, $\rho_0 = \rho.\rho_0$ , $\rho = 1$
       and return to 1.
```

AGMRES: This method has been developed by Ayachour in 2003 [1]. In this method we use the steps of Algorithm2, but Ayachour for computing $\bar{z}$ in step2 uses some differentiable functions. He introduces two differentiable functions f_k or g_k for $\|e_1 - \bar{H}_k z\|$ depending the situations of $h_{k+1,k}$.

Finally the combinations of the two optimized solutions will appear in a theorem for computing the approximate solution as $x_k = x_0+d$ without using the Given's rotations [1].

The following algorithm is an optimized implementation of the above mentioned theorem. In this algorithm if we assume

$$\bar{H}_k = \begin{pmatrix} w \\ H_k \end{pmatrix}$$

where

$$w = (h_{1,1},\ldots,h_{1,k})$$

and

$$H_k = \begin{pmatrix} h_{2,1} & \cdots & h_{2,k} \\ & \ddots & \vdots \\ & & h_{k+,k} \end{pmatrix}$$

then we consider $H'_k = H_k + e_k e_k^T$. In that theorem the matrix H'^{-1}_k uses many times, therefore we consider R_k as H'^{-1}_k which can be computed from R_{k-1} and the first (k-1) components of the last column of H'_k (here it is g). Now the algorithm for computing x_k is:

Algorithm 4: Implementation of Ayachour for GMRES (AGMRES)

```
0.  Choose x then r = b-Ax and $\rho_0 \equiv \|r\|$. If $\rho_0 \le$eps then x is a solving of (1), Stop, else set $v_1 = \frac{r}{\|r\|}, \alpha_0 = 1$

1.  for k = 1,…,m do
    a. Evaluate $v_{k+1} = Av_k$.
    b. Set $w_k = \langle v_{k+1}, v_1\rangle$, $v_{k+1} = v_{k+1} - w_k v_1$.
    c. for j = 1,…,k do
        $\lambda_{j-1} = \langle v_{k+1}, v_j\rangle$, $v_{k+1} = v_{k+1} - \lambda_{j-1} v_j$.
    d. $\beta = \|v_{k+1}\|, v_{k+1} = v_{k+1}/\beta$, set $g = (\lambda_1,\ldots,\lambda_{k-1})^T$.
    e. Evaluate $R_k = \begin{pmatrix} R_{k-1} & -R_{k-1}g \\ & 1 \end{pmatrix}$, $u_k = \langle R_k(:,k), w^T\rangle$.
    f. Set $\gamma_k = \frac{1}{\sqrt{\beta^2 + (u_k\alpha_0)^2}}$, $\sin\theta_k = \beta\gamma_k$, $\alpha_1 = \alpha_0 \sin\theta_k$.
    g. $\|r_k\| = \rho_0\alpha_1$ , if $\|r_k\| <$ eps or $|u_k| <$ eps go to 2.
    h. Update $u_k = \frac{u_k}{\beta}$, $R_k(:,k) = \frac{1}{\beta} R_k(:,k)$, $\alpha_0 = \alpha_1$.
```

2. Let k be the final iteration number from 1.

   a. Set $y=\left(\sin^2\theta_k u_1,\ldots,\sin^2\theta_k u_{k-1},\gamma_k^2 u_k\right)^T$, $\bar{z}=H_k'^{-1}y$.

   b. $x=x+\rho_0\alpha_0^2V_k\bar{z}$, $r=b-Ax$.

   c. If $\|r\|<eps$ accept x otherwise $\rho_0=\|r\|$, $v_1=\frac{r}{\|r\|}$

   and return to 1.

NUMERICAL TESTS FOR COMPARISON OF THE METHODS

In this section we compare the two methods by numerical examples using MATLAB software. In the presented figures the residual norm is in the basis of $\log_{10}$. In the following tables the Error is the norm of the residuals of the computed solutions and Time is the total consuming time. Also note that in this section we consider $x_0=(0,0,\ldots0)^T$.

Example 1: In this example the matrix A is almost upper triangular. The results show that both methods AGMRES and SGMRES converges fast. The matrix A is

$$A=\begin{pmatrix} 1 & 0 & 0.5 & 0 & & 1 \\ & 1 & 0 & 0.5 & \ddots & \\ & & 1 & \ddots & \ddots & 0 \\ & & & \ddots & 0 & 0.5 \\ & & & & 1 & 0 \\ 1 & & & & & 1 \end{pmatrix}$$

In this example n = 100 and k = 10 have been chosen. The vector b has been selected in the way that $x=(1,2,\ldots,100)^T$ be a solution of (1).

As Fig. 1 shows both methods have been reached to the desired solution after 6 iterations.

It is interesting to note that both methods for an almost upper triangular matrix converge very fast (Table 1).

Example 2: In this example the matrix A has the form

$$A=\begin{pmatrix} 1 & 0 & 1.1 & 4.6 & & & \\ 2.1 & 2 & 0 & 1.1 & 4.6 & & \\ 3.5 & 2.1 & 3 & 0 & & \ddots & \\ 0 & 3.5 & \ddots & \ddots & \ddots & \ddots & 4.6 \\ & & \ddots & \ddots & \ddots & 0 & 1.1 \\ & & & 3.5 & 2.1 & n-1 & 0 \\ & & & & 3.5 & 2.1 & n \end{pmatrix}$$

Table 1:

	Iterate	Error	Time
SGMRES	5	9.4436 e^{-012}	1.2791 e^{-002}
AGMRES	6	9.4156 e^{-012}	1.5028 e^{-002}

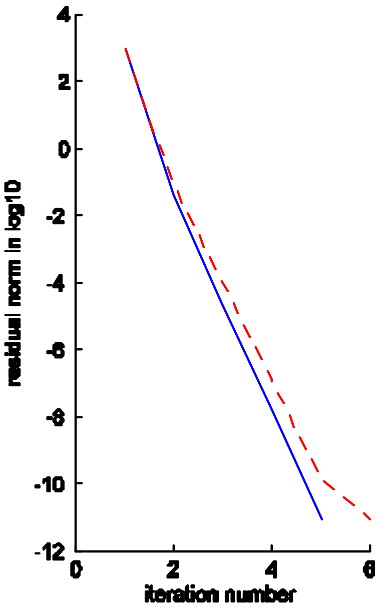

Fig. 1: With n = 100, k = 10

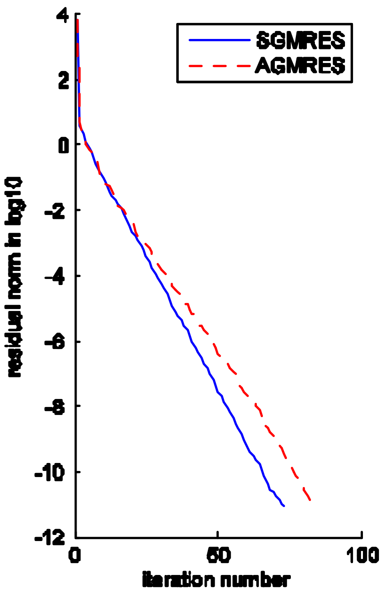

Fig. 2: With n = 500, k = 15

Consider n = 500, k = 15 and $x=(1,1,\ldots,1)^T$ be the solution. The results again show that the SGMRES works faster.

The numerical results are shown in Table 2.

Table 2:

	Iterate	Error	Time
SGMRES	73	$9.7698\ e^{-012}$	$8.2592\ e^{-001}$
AGMRES	83	$9.9305\ e^{-012}$	$9.3438\ e^{-001}$

Table 3:

	Iterate	Error	Time
SGMRES	14	$9.2374\ e^{-012}$	$4.5576\ e^{-001}$
AGMRES	19	$9.6620\ e^{-012}$	$4.4398\ e^{-001}$

Example 3: In this exa mple we have used a block matrix, this matrix has been taken from [11]. Let p be an even integer and denote by I and 0, respectively, the $\frac{p}{2}\times\frac{p}{2}$ identity and zero matrices. Define also the $\frac{p}{2}\times\frac{p}{2}$ matrices T_1 and T_2 as in

$$T_1=\begin{pmatrix} -2 & 0 & \dots & \dots & 0 \\ -1 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & -1 & 0 \\ 0 & \dots & \dots & -1 & -1 \end{pmatrix},\ T_2=\begin{pmatrix} -1 & 0 & \dots & \dots & 0 \\ -1 & -1 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & -1 & 0 \\ 0 & \dots & 0 & -1 & -2 \end{pmatrix}$$

The matrix A is a nonsymmetric $p^2\times p^2$ matrix as in the following

$$A=\begin{pmatrix}
4I & 0 & \dots & \dots & \dots & \dots & \cdots & 0 & T_1 & -2I & 0 & \dots & \dots & \dots & \dots & 0 \\
0 & 4I & & & & & & \vdots & -I & T_2 & -I & 0 & & & & \vdots \\
\vdots & & \ddots & & & & & \vdots & 0 & -I & T_1 & -I & 0 & & & \vdots \\
\vdots & & & \ddots & & & & \vdots & \vdots & 0 & -I & T_2 & -I & 0 & & \vdots \\
\vdots & & & & \ddots & & & \vdots & \vdots & & & & \ddots & & & \vdots \\
\vdots & & & & & \ddots & & \vdots & \vdots & & & & & \ddots & & \vdots \\
\vdots & & & & & & 4I & 0 & \vdots & & & & 0 & -I & T_1 & -I \\
0 & \dots & \dots & \dots & \dots & \dots & 0 & 4I & 0 & \dots & \dots & \dots & \dots & 0 & -2I & T_2 \\
T_2 & -2I & 0 & \dots & \dots & \dots & \dots & 0 & 4I & 0 & \dots & \dots & \dots & \dots & \dots & 0 \\
-I & T_1 & -I & 0 & & & & \vdots & 0 & 4I & & & & & & \vdots \\
0 & -I & T_2 & -I & & & & \vdots & \vdots & & \ddots & & & & & \vdots \\
\vdots & 0 & -I & T_1 & -I & 0 & & \vdots & \vdots & & & \ddots & & & & \vdots \\
\vdots & & & & \ddots & & & \vdots & \vdots & & & & \ddots & & & \vdots \\
\vdots & & & & & \ddots & & \vdots & \vdots & & & & & \ddots & & \vdots \\
\vdots & & & & 0 & -I & T_2 & -I & \vdots & & & & & & 4I & 0 \\
0 & \dots & \dots & \dots & \dots & 0 & -2I & T_1 & 0 & \dots & \dots & \dots & \dots & \dots & 0 & 4I
\end{pmatrix}$$

Now suppose $p = 30$, i.e. $n = 900$, $k = 20$, $x = (1,2,\dots,n)^T$, $x = (1,2,\dots,n)^T$ be the exact solution. The results show the behavior of the methods.

As the results show in Table 3 both method works very fast. Note that if the dimension increases then the SGMRES does not work as good as before. For example we have tested for $p = 64 (n = 4096)$, $k = 40$ and $x = (1,1,\dots,1)^T$.

The results have been plotted in Fig. 4 and show that AGMRES works faster, i.e. the AGMRES works better when we increase the dimension. In this case although SGMRES works a bite slower than AGMRES but both of them solve a problem with n = 4096, by at most 13 iterations, when k = 40.

This example emphasizes that the orthogonal projection method is really useful for solving the sparse linear system of equations.

The numerical results for this case are in Table 4 as follows

Table 4:

	Iterate	Error	Time
SGMRES	13	$9.8744\ e^{-012}$	4.5271
AGMRES	11	$9.4749\ e^{-012}$	3.6161

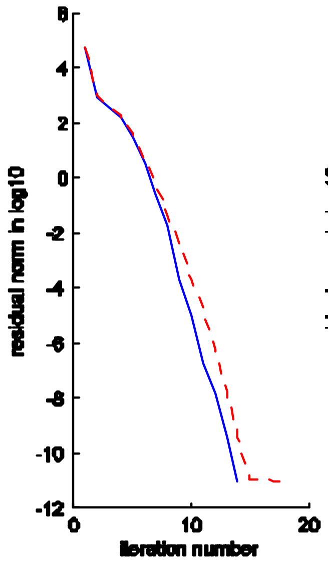

Fig. 3: With n = 900, k = 20

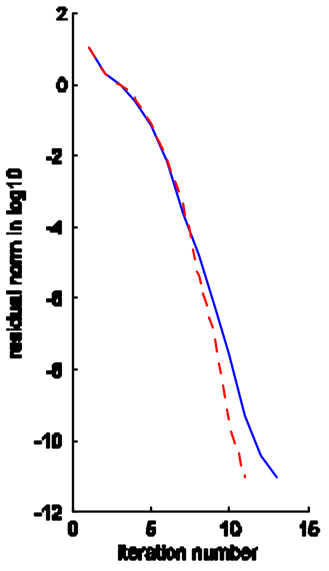

Fig. 4: With n = 4096, k = 40

Table 5:

	Iterate	Error	Time
SGMRES	70	9.1819 e^{-012}	5.9216 e^{+001}
AGMRES	8	8.3457 e^{-012}	3.2909 e^{+000}

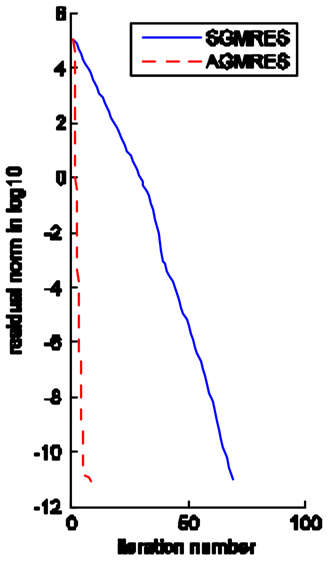

Fig. 5: With n = 1000, k = 25

Example 4: In this example we have tested the methods for complex matrix, e.g.

$$A = \begin{pmatrix} 4 & 0 & 1 & 0.7 & & & \\ 2i & 4 & 0 & 1 & \ddots & & \\ & 2i & 4 & \ddots & \ddots & 0.7 \\ & & \ddots & \ddots & 0 & 1 \\ & & & 2i & 4 & 0 \\ & & & & 2i & 4 \end{pmatrix}.$$

This matrix has been taken from [1].

We have selected n = 1000, k = 25 and $x = (1,2,\ldots,n)^T$. The results are plotted in Fig. 5 and shows that the SGMRES does not work good any more and even it is not as good as AGMRES.

The numerical results are in Table 5 and show that the speed of convergence for SGMRES is low.

CONCLUSIONS

s we already described the GMRES is a powerful method for solving the linear systems of equations. Two implementations AGMRES and SGMRES for solving their least square problem do not use Given's rotations. In SGMRES a solution of an upper triangular system requires, but in AGMRES the introduced differentiable functions will be used. Both methods are powerful for solving the linear systems, but they show different behavior when we come to numerical examples. As the numerical examples in the last sections show when the dimension of the matrix increases, e.g. greater than 2000.

he AGMRES works better and if the matrix A is a complex matrix the SGMRES has the low performance of convergence while AGMRES converges fast. Finally if A is a real matrix having not very large dimension, the SGMRES works very fast so we suggest of using this method in these cases.

REFERENCES

1. Ayachour, E.H., 2003. A fast implementation for GMRES method. J. Comput App. Math., 159: 269-283.
2. Walker, W.F. and L. Zhou, 1994. A simpler GMRES, Linear Algebra Appl., 1: 571- 581.
3. Saad, Y. and M.H. Schultz, 1986. GMRES: Ageneralized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Statist. Comput., 7: 856-869.
4. Walker, W.F., 1988, Implementation of the GMRES method, using Householder transformation. SIAM J. Sci. Stat. Comput., 9: 152-163.
5. Saad, Y., 1993, A flexible inner-outer preconditioned GMRES algorithm. SIAM S. Sci. Comput., 14: 461-469.
6. Essai, 1998. Weighted FOM and GMRES for solving nonsymmetric linaer systems. Numer. Algorithm, 18: 277-299.
7. Nachtigal, N.M. *et al.*, 1992. A hybrid GMRES algorithm for nonsymmetric linear systems. SIAM J. Matrix Anal. Appl., 13: 796-825.
8. Turner, K. and H.F. Walker, 1992. Efficient high accuracy solution with GMRES (m). SIAM J. Sci. Stat. Comput., 13: 815-825.
9. Brown, P.N., 1991. A theoretical comparison of the Arnoldi and the GMRES algorithms. SIAM J. Sci. Stat. Comput., 12: 58-78.
10. Rozloznik, M., 1996. Numerical stability of the GMRES method. Ph.D. Thesis, Czech Technical University.
11. Saberi Najafi, H. and H. Zareamoghaddam, 2008. A new computational GMRES method. Applied Mathematics and Computation, 199 (2): 527-534.