Residual Algorithm with Preconditioner for Linear System of Equations

J. Izadian

Department of Mathematics, Faculty of Sciences, Mashhad Branch, Islamic Azad University, Mashhad, Iran.
Jalal_Izadian@yahoo.com

M. Jalili

Department of Mathematics, Neyshabur Branch, Islamic Azad University, Neyshabur, Iran.
Jalili.maryam@yahoo.com

Abstract

One of the most powerful tools for solving large and sparse systems of linear equation is iterative methods. Their significant advantages like low memory requirements and good approximation properties make them very popular, and they are widely used in applications throughout science and engineering. Residual Algorithm for solving large-scale nonsymmetric linear system of equation which symmetric part is positive (or negative) definite, is evaluated. It uses in a systematic way the residual vector as a search direction, and a spectral steplength. The global convergence is analyzed. A preliminary numerical experimentation is included for showing that the new algorithm is a robust method for solving nonsymmetric linear system and it is competitive with the well-known GMRES and BICGSTAB in number of computed residual and CPU time. The new method for sparse matrix with $10^{12}$ entries has been successfully examined with we use the two preconditioning strategies ILU and SSOR.

Keywords: linear systems, nonsymmetric matrices, preconditioned iterative methods
1. Introduction

We introduce an iterative method for solving the linear system of equation
\[ Ax = b \] (1)
When \( A \in \mathbb{R}^{n \times n} \) is nonsymmetric and the symmetric part of \( A \), i.e., \( A_s = \frac{A^T + A}{2} \)
is positive (or negative) definite, \( b \in \mathbb{R}^n \) and \( n \) is large enough.

Different iterative methods have been developed for solving (1). To solve a very large linear system of the form given in (1), especially those derived from 3D soil-structure interaction problems, direct solution methods such as sparse LU factorization are impractical from the perspective of computational time and memory requirement. Krylov subspace iterative methods on the other hand, may overcome these difficulties and they are commonly used for solving large-scale linear systems. Three of the most popular Krylov subspace iterative methods for solving the system (1) are SYMMLQ, MINRES and symmetric QMR (SQMR) method. The SYMMLQ and MINRES methods, developed by Paige and Saunders in 1975 [6], could only be used in conjunction with symmetric positive definite preconditioners.

Unlike the optimal SYMMLQ and MINRES methods, the SQMR method proposed by Freund and Nachtigal [2] can be used in conjunction with a symmetric indefinite preconditioner. Because of this flexibility, we choose to use RA throughout this paper. There are also numerical results showing that RA combined with a symmetric indefinite preconditioner is more effective than a positive definite preconditioner.

This paper is organized as follows. In Section 2, we give the model algorithm RA. In Section 3, we give Extensions of the method which we combine RA with a preconditioning method in order to speed up convergence. In Section 4, we consider the application of the RA algorithm for solving large-scale nonsymmetric linear system of equations whose symmetric part is positive (or negative) definite. In last section, we give a short conclusion.

2. Model Algorithm

Let the functions \( f \) and \( g \) are given as
\[
\begin{align*}
g : & \mathbb{R}^n \rightarrow \mathbb{R}^n \\
g(x) & = Ax - b \\
\end{align*}
\] (2)
\[
\begin{align*}
f : & \mathbb{R}^n \rightarrow \mathbb{R} \\
f(x) & = \| g(x) \|_2^2 \\
\end{align*}
\] (3)
where \( \| \cdot \|_b \) denotes the Euclidian norm. Now assume that \( \{ \eta_k \} \) is a given sequence such that \( \eta_k > 0 \) for \( k \in N \) (the set of natural numbers) and

\[
\sum_{k=0}^{\infty} \eta_k = \eta < \infty.
\]

Suppose \( \gamma \in (0, 1) \), \( 0 < \sigma_{\min} < \sigma_{\max} < 1 \), \( \alpha > 0 \) and \( x_0 \in \mathbb{R}^n \) is a given arbitrary initial point. Combining the systematic use of the search direction, the spectral choice of step length, and the Armijo test, we obtain the following RA (Residual Algorithm) algorithm.

\section*{2-1 RA algorithm}

Given: \( \alpha > 0 \), \( \gamma \in (0, 1) \), \( 0 < \sigma_{\min} < \sigma_{\max} < 1 \), \( \{ \eta_k \} \) such that (4) holds, and \( x_0 \in \mathbb{R}^n \) be a sufficiently good initial guess.

Set \( r_0 = b - Ax_0 \), and \( k = 0 \).

\begin{itemize}
  \item Step1: If \( r_k = 0 \), stop the process.
  \item Step2: Set \( \lambda = 1 \).
  \item Step3: If

\[
\left\| r_k - \left( \frac{\lambda}{\alpha_k} \right) Ar_k \right\|^2 \leq \left\| r_k \right\|^2 + \lambda^2 \left( \gamma \alpha \right)^2 \left\| r_k \right\|^2
\]

\begin{align*}
\lambda_k &= \lambda, \\
x_{k+1} &= x_k + \left( \frac{\lambda}{\alpha_k} \right) r_k, \\
r_{k+1} &= r_k - \left( \frac{\lambda}{\alpha_k} \right) Ar_k
\end{align*}

\item Step4: Choose \( \sigma \in [\sigma_{\min}, \sigma_{\max}] \), set \( \lambda \leftarrow \sigma \lambda \) and go to step 3.

\item Step5: Set

\[
\lambda_k = \lambda, \\
x_{k+1} = x_k + \left( \frac{\lambda}{\alpha_k} \right) r_k, \\
r_{k+1} = r_k - \left( \frac{\lambda}{\alpha_k} \right) Ar_k
\]

\item Step6: Set

\[
\alpha_{k+1} = \frac{r_{k+1}^T Ar_{k+1}}{r_{k+1}^T r_k}, \\
k = k + 1
\]

go to step 1.

A global convergence theorem is presented in [4], that guarantee convergence of the solution of problem (1).
3 Extensions of the method

Now, we combine RA with a preconditioning method in order to speed up convergence.

3-1 The preconditioned RA algorithm

We suppose that $M$ be a preconditioning matrix. In stead of solving (1) one may as well solve

$$M A x = M b$$

(5)

or

$$A M y = b \quad \text{with} \quad x = M y.$$  

(6)

Therefore by replacing $A$ by $M A$ and $r_s = b - A x_s$ by $r_s = M (b - A x_s)$ we have an algorithm to solve (5) iteratively. In this case the computed residuals $r_k$ are not the real residuals, but $r_k = M (b - A x_k)$. By replacing $A$ by $A M$ and $X_s = x_s$ by $X_s = M^{-1} x_s$, we have an algorithm that solves iteratively (6).

The computed residuals are the real ones, that is, $r_k = b - A M x_k$, but now $x_k$ is not the approximated value that we are interested in, we would like to have $M x_k$ instead. If we do not want to monitor the approximations of the exact solution $x$, it suffices to compute $M x_k$ only after termination. In both variants, the modified RA algorithm may converges faster, due to the preconditioning.

3-2 SSOR preconditioning

The SSOR preconditioner can be derived straight forward from matrix coefficient. The SSOR preconditioner strategies is based on decomposition $A$ as

$$A = L + D + U$$

(7)

where $L$ and $U$, respectively, are the lower and upper triangular part of $A$ and $D$ is the diagonal part of $A$. The SSOR matrix is defined as:

$$M = (D + L) D^{-1} (D + U)$$

(8)

3-3 ILU preconditioning

The Incompelete LU factorization [1,3], ILU, is probably the best known general purpose preconditioner. We define:

$$S = \left\{ (i, j) \mid a_{ij} \neq 0 \right\}$$

(9)
such that $A = (a_{ij})_{n \times n}$. The ILU matrix denote $M = \tilde{L}\tilde{U}$. The ILU factorization, yields:

$$A = \tilde{L}\tilde{U} + R$$

where $R$ is the residual matrix representation the difference between $A$ and $\tilde{L}\tilde{U}$.

3-3-1 **ILU algorithm for** $A = (a_{ij})_{n \times n}$

Step 1: Do $r = 1 : n - 1$

Step 2: $d = \frac{1}{a_{rr}}$

Step 3: Do $i = r + 1 : n$

Step 4: If $(i, r) \in S$ then

Step 5: $e = da_i$

Step 6: $a_i = e$

Step 7: Do $j = r + 1, n$

Step 8: If $(i, j) \in S$ and $(r, j) \in S$ then

Step 9: $a_{ij} = a_{ij} - ea_j$

Step 10: End

Step 11: End

Step 12: End

Step 13: End

Step 14: End

4. **Numerical result**

In this section we consider the application of the RA algorithm for solving large-scale nonsymmetric linear system of equations whose symmetric part is positive (or negative) definite. For RA algorithm we use the following parameters:

$$\alpha = \|b\|, \gamma = 10^{-4}, \sigma_{\min} = 0.1, \sigma_{\max} = 0.9, \eta_k = 10^{-3}(1 - 10^{-7})^k$$

We choose a new $\tilde{\lambda}$ at step 4 by following procedure, given the current $\lambda_c > 0$ we set the new $\tilde{\lambda} > 0$

$$\tilde{\lambda} = \begin{cases} 
\sigma_{\min}\lambda_c & \text{if} \quad \lambda_i < \sigma_{\min}\lambda_c \\
\sigma_{\max}\lambda_c & \text{if} \quad \lambda_i < \sigma_{\max}\lambda_c \\
\tilde{\lambda}_i & \text{otherwise}
\end{cases}$$
where

\[ \lambda_i = \frac{\lambda_c^2 f(x_k)}{f(x_k + \lambda_c d) + (2\lambda_c - 1)f(x_k)}. \]

The iteration terminated if:

\[ \frac{\|r_k\|}{\|b\|} \leq \varepsilon, \quad 0 < \varepsilon << 1, \quad \varepsilon = 5 \times 10^{-15}. \]

In the numerical result given below, the well-known and well-established Krylov subspace methods: BICGSTAB and GMRES [5,7], that restarts every 20 steps \((m = 20)\), namely GMRES(20) and \(m = 40\), namely GMRES(40) have been used to solve the given system of linear equations and is compared with RA.

We have used a PC-Pentium(R)4 ,CPU 2.4 GHz, 448 MB of RAM and MATLAB R2006a for computations. In all of experiences we have used \(x_0 = 0\), on the initial guess. In the following numerical experiments, \(n\) and "nnz" denote the order of matrix, and the number of nonzero elements of matrix, respectively. On the other hand iter and CPUtime, refer to number of iteration and elapsed time, respectively. In Table 1 we report, the dimension of the problem \((n)\); the number of computed residual \((\text{iter})\), and elapsed time (sec) until convergence occurred \((\text{CPUtime})\).

**Notation:** RESALG denote RA in legend figure.

### 4-1 Example

We consider the matrix by MATLAB code \(A = \text{gallery('hanowa', }n,n)\). We assume that \(n = 7000, b = (1,1,...,1)\) and \(\text{nnz} = 14000\). The matrix for \(n = 10\) is presented in Figure 1. In Figures 2 and 3 we show the behavior of all considered methods when using preconditioning strategies ILU and SSOR.

<table>
<thead>
<tr>
<th>(n = 7000)</th>
<th>(RA)</th>
<th>BICGSTAB</th>
<th>GMRES(20)</th>
<th>GMRES(40)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPUtime</td>
<td>iter</td>
<td>CPUtime</td>
<td>iter</td>
</tr>
<tr>
<td><strong>ILU</strong></td>
<td>1.48547</td>
<td>3</td>
<td>1.85422</td>
<td>15</td>
</tr>
<tr>
<td><strong>SSOR</strong></td>
<td>0.98623</td>
<td>3</td>
<td>1.01651</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 1
Residual algorithm with preconditioner

Figure 1: Behavior of all methods when using ILU preconditioner

Figure 2: Behavior of all methods when using SSOR preconditioner
4-2 Example
Consider the matrix by MATLAB code $A = \text{gallery ('toeppen ', } n, 1, 10, n, -10, -1)$.
Let assume that $n = 10^6$, $b = (1, 1, \ldots, 1)$ and $nnz = 4999994$. The matrix in the case $n = 10$ has been presented in Figure 4, and in Figure 5 we show the behavior of all considered methods when using preconditioning strategies $ILU$.

<table>
<thead>
<tr>
<th>n = $10^6$</th>
<th>RA</th>
<th>BICGSTAB</th>
<th>GMRES(20)</th>
<th>GMRES(40)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPUtime</td>
<td>iter</td>
<td>CPUtime</td>
<td>iter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ILU</td>
<td>14.04</td>
<td>2</td>
<td>186.379</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>402.538</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>SSOR</td>
<td>6.0553</td>
<td>2</td>
<td>269.234</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>388.271</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 2

Figure 4: Pentadiagonal Toeplitz matrix

Figure 5: behavior of all method when use ILU preconditioner
4-3 Example

We consider the elliptic problem $u_{xx} + u_{yy} = e^{x^2 - y^2} \sin(x + y)$ on the unit square, with homogeneous Dirichlet boundary conditions, $u = 0$, on the border of the region. The discretization grid has 100 internal nodes per axis producing an $n \times n$ matrix where $n = 10^4$. In Figures 6,7 we show the behavior of all considered methods when using preconditioning strategies $ILU, SSOR$; Figure 8 shows $(x_k, y_k, u(x_k, y_k))$.

<table>
<thead>
<tr>
<th>$n = 10^4$</th>
<th>RA</th>
<th>BICGSTAB</th>
<th>GMRES(20)</th>
<th>GMRES(40)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPUtime</td>
<td>iter</td>
<td>CPUtime</td>
<td>iter</td>
</tr>
<tr>
<td>$ILU$</td>
<td>0.02297</td>
<td>15</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>$SSOR$</td>
<td>5.59485</td>
<td>46</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 3
4-3 Example

We consider the following simple boundary-value problem of second order. We subdivide [-1,1] into $n=10^4$ equal subintervals, the difference method give n $n \times n$ matrix where $n=10^4$ and matrix is nonsymmetric positive definite that arise from the discretization of the boundary value problem. In Figure 9 we show the behavior of all considered methods when using preconditioning strategies ILU; Figure 10 shows $(x_k,y_k,u(x_k,y_k))$

$$\begin{cases} 
  y''(x) = \frac{1}{(2-x)} y'(x) + \frac{1}{(2(2-x)^2)} y(x) + x & -1 \leq x \leq 1 \\
  y(-1) = 0 \\
  y(1) = 0 
\end{cases}$$

<table>
<thead>
<tr>
<th>$n = 10^4$</th>
<th>RA</th>
<th>BICGSTAB</th>
<th>GMRES(20)</th>
<th>GMRES(40)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPUtime</td>
<td>iter</td>
<td>CPUtime</td>
<td>iter</td>
</tr>
<tr>
<td>ILU</td>
<td>0.1305</td>
<td>43</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>SSOR</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 4
5. Conclusions

We have interpreted in this paper the RA algorithm, and compare with well-known Krylo subspace methods, with using two preconditioning strategies ILU and SSOR. We observe that using the residual and spectral step length decrease nonmonotonically norm of the residual and also guarantees convergence. Finally, the numerical results confirm that the RA algorithm presented in this article is worth further application and apply for nonlinear system of equation.

References


Received: October, 2011