Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación

Lecturas en Ciencias de la Computación
ISSN 1316-6239

Residual iterative schemes for large-scale linear systems

William La Cruz y Marcos Raydan

RT 2006-05

Centro de Cálculo Científico y Tecnológico de la UCV

CCCT-UCV

Caracas, Mayo, 2006.

(Revisado Diciembre 2006)

Residual iterative scheme for large-scale linear systems

William La Cruz[*] Marcos Raydan[†]

December 14, 2006

Abstract

A new iterative scheme that use the residual vector as search direction is proposed and analyzed for solving large-scale nonsymmetric linear systems. It is closely related to Richardson's method, although the stepsize and some other new features are inspired by the success of recently proposed residual methods for nonlinear systems. The convergence is analyzed for general matrices, and strong convergence is established when the symmetric part of the coefficient matrix is positive (negative) definite. A preliminary numerical experimentation is included to show that the proposed scheme outperforms some recently proposed variations on Richardson's method, and also to show that it is competitive with well-known and well-established Krylov subspace methods: GMRES and BiCGSTAB, with and without preconditioning.

Keywords: Linear systems, Richardson's method, Krylov subspace methods, Barzilai-Borwein stpsize.

1 Introduction

We are interested in solving linear systems of equations

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

where $A \in I\!R^{n\times n}$ is not symmetric, $b \in I\!R^n$, and n is large.

[*]Dpto. de Electrónica, Computación y Control, Escuela de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad Central de Venezuela, Caracas 1051-DF, Venezuela. E-mail: lacruzw@ucv.ve. This author was supported by CDCH project PI-08-14-5463-2004 and Fonacit UCV-PROJECT 97-003769.

[†]Dpto. de Computación, Facultad de Ciencias, Universidad Central de Venezuela, Apartado 47002, Caracas 1041-A, Venezuela. E-mail: mraydan@kuaimare.ciens.ucv.ve. This author was supported by Fonacit UCV-PROJECT 97-003769.

The well-known Richardson's method (also known as Chebyshev method) and its variations are characterized by using the residual vector, $r(x) = b - Ax$, as search direction to solve (1) iteratively (see, e.g., [6, 10, 11, 22, 24]). In general, these variations of Richardson's method are not competitive with Krylov subspace methods, that offer nowadays the best potential for solving (1) when combined with suitable preconditioning strategies. For a review on Richardson's method and variations see [7, 27, 28].

Nevertheless, from a different perspective, solving linear systems of equations can be seen as a particular (although very special) case of solving nonlinear systems of equations. For nonlinear systems, some new iterative schemes have recently been presented that use in a systematic way the residual vectors as search directions [17, 18]. These ideas become effective, and competitive with Newton-Krylov ([2, 8, 9, 16]) schemes for large-scale nonlinear systems, when the step lengths are chosen in a suitable way.

In this work we combine and adapt, for linear systems, the ideas introduced in [17, 18] for the nonlinear case. To be precise, we present in Section 2 a scheme that takes advantage of the method presented in [17] for choosing the direction, plus or minus the residual, depending on the sign of a Rayleigh quotient closely related to the step length. It also takes advantage of the new globalization strategy proposed and analyzed in [18] that allows the norm of the residual to decrease non monotonically and still guarantees convergence.

It is worth noticing that since our proposal uses plus or minus the residual as search direction, then it can be viewed as a new variant of the well-known Richardson's method. However, there are significant new features. The most important is the use of a new steplength based on the Barzilai-Borwein choice ([1, 12, 15, 20]) that has proved to yield fast local convergence for the solution of nonlinear optimization problems ([3, 4, 5, 14, 21]). However, this special choice of step size cannot guarantee global convergence by itself, as it usually happens with other variations (see, e.g., [6, 10, 11, 22, 24]). For that, we combine its use with a tolerant globalization strategy, that represents the second new feature. In section 3 we present a preliminary numerical experimentation to compare the proposed scheme with some variations on Richardson's method, and also with well-known and well-established Krylov subspace methods: GMRES and BiCGSTAB, with and without preconditioning

2 General algorithm and convergence

Let the functions $g : I\!R^n \to I\!R^n$ and $f : I\!R^n \to I\!R$ be given as

$$g(x) = Ax - b, \tag{2}$$

and

$$f(x) = \|g(x)\|^2, \tag{3}$$

where $\| \cdot \|$ denotes, throughout this work, the Euclidian norm. We now present our general algorithm that generates the iterates using plus or minus the residual vector as search direction, and a spectral steplength closely related to the Barzilai-Borwein choice of steplength [1], as follows

$$x_{k+1} = x_k + \text{sgn}(\beta_k)(1/\beta_{k-1})r_k,$$

where $\beta_k = (r_k^t A r_k)/(r_k^t r_k)$, and r_k is the residual vector at x_k. The use of the Barzilai-Borwein steplength is inspired by the success obtained recently for solving nonlinear systems of equations [17, 18]. To guarantee convergence, from any initial guess x_0 and any positive initial steplength $1/\beta_0$, a nonmonotone line search strategy needs to be incorporated. This globalization strategy is inspired by the proposition presented in [18], and requires some fixed parameters: $\{\eta_k\}$, γ, and $\sigma_{min} < \sigma_{max}$. Let us assume that $\{\eta_k\}$ is a given sequence such that $\eta_k > 0$ for all $k \in I\!N$ (the set of natural numbers) and

$$\sum_{k=0}^{\infty} \eta_k = \eta < \infty. \tag{4}$$

Let us also assume that $\gamma \in (0, 1)$ and $0 < \sigma_{min} < \sigma_{max} < 1$.

Algorithm 2.1. *Residual Algorithm 1 (RA1)*

```
Given: $x_0 \in I\!R^n$, $\alpha_0 > 0$, $\gamma \in (0,1)$, $0 < \sigma_{min} < \sigma_{max} < 1$, $\{\eta_k\}_{k \in I\!N}$ such that (4)
    holds. Set $r_0 = b - Ax_0$, and $k = 0$;

Step 1. If $r_k = 0$, stop the process (successfully);

Step 2. Set $\beta_k = (r_k^t A r_k)/(r_k^t r_k)$;

Step 3. If $\beta_k = 0$, stop the process (unsuccessfully);

Step 4. (Backtracking process) Set $\lambda \leftarrow 1$;

Step 5. If $\|r_k - \text{sgn}(\beta_k)(\lambda/\alpha_k)Ar_k\|^2 \leq \|r_k\|^2 + \eta_k - \gamma\lambda^2\|r_k\|^2$ go to Step 7;

Step 6. Choose $\sigma \in [\sigma_{min}, \sigma_{max}]$, set $\lambda \leftarrow \sigma\lambda$, and go to Step 5;

Step 7. Set $\lambda_k = \lambda$, $x_{k+1} = x_k + \text{sgn}(\beta_k)(\lambda/\alpha_k)r_k$, and $r_{k+1} = r_k - \text{sgn}(\beta_k)(\lambda/\alpha_k)Ar_k$;

Step 8. Set $\alpha_{k+1} = |\beta_k|$, $k = k + 1$ and go to Step 1.
```

Remark 2.1. The vector r_k is built recursively in the algorithm, for the sake of efficiency, but it is mathematically equivalent to $r_k = b - Ax_k$ for all $k \in I\!N$.

Remark 2.2. The spectral step length $\alpha_{k+1} = (r_k^t r_k)/(r_k^t A r_k)$ for minimization was introduced in the Barzilai-Borwein paper [1]. The properties of their method for convex quadratic functions were established in [20], and further analyzed in [12]. For a review containing the more recent advances on spectral choices of the steplength, see [15].

Remark 2.3. In practice, the parameters associated with the line search strategy are chosen to reduce the number of backtrackings as much as possible while keeping the convergence properties of the method. For example, the parameter $\gamma > 0$ is chosen as a very small number ($\gamma \approx 1.D-4$), and η_k is chosen as a large number when $k = 0$ and then it is reduced as slow as possible making sure that (4) holds (e.g., $\eta_k = 10^4\,(1-10^{-6})^k$). Finally $\sigma_{min} < \sigma_{max}$ are chosen as classical safeguard parameters in the line search process (e.g., $\sigma_{min} = 0.1$ and $\sigma_{max} = 0.5$).

In order to present some convergence analysis for Algorithm 2.1, we first need some technical results. For the sake on clarity we introduce the notation $\text{sgn}_k = \text{sgn}(\beta_k)$ that will be used throughout the rest of the paper.

Proposition 2.1. *Let the sequences $\{x_k\}_{k\in\mathbb{N}}$, $\{r_k\}_{k\in\mathbb{N}}$, $\{\beta_k\}_{k\in\mathbb{N}}$ and $\{\lambda_k\}_{k\in\mathbb{N}}$ be generated by Algorithm 2.1. Then, the following properties hold.*

(a) For all $k \in \mathbb{N}$:

$$g(x_k) = -r_k, \tag{5}$$

$$g(x_{k+1}) = g(x_k) - \text{sgn}_k(\lambda_k/\alpha_k)Ag(x_k). \tag{6}$$

(b) The vector

$$d_k = \text{sgn}_k(1/\alpha_k)r_k = -\text{sgn}_k(1/\alpha_k)g(x_k)$$

is a descent direction for the function f, for all $k \in \mathbb{N}$. Moreover,

$$f(x_k + \lambda d_k) = \|r_k - \text{sgn}_k(\lambda/\alpha_k)Ar_k\|^2, \text{ for all } k \in \mathbb{N}.$$

(c) The iterates $x_{k+1} = x_k + \lambda_k d_k$ and λ_k satisfy the following conditions

$$f(x_{k+1}) \le f(x_k) + \eta_k - \gamma\lambda_k^2\|g(x_k)\|^2, \tag{7}$$

for $k \ge 0$.

Proof. (a). Since $r_k = b - Ax_k$, for $k \in \mathbb{N}$, then by the definition of g, the equations (5) and (6) holds.

Proof of (b). Since $\nabla f(x) = 2A^t g(x)$, then

$$\begin{aligned}\nabla f(x_k)^t d_k &= 2(g(x_k)^t A)(-\text{sgn}_k(1/\alpha_k)g(x_k))\\ &= -2(g(x_k)^t Ag(x_k))\text{sgn}_k(1/\alpha_k)\\ &= -2(\beta_k(g(x_k)^t g(x_k)))\text{sgn}_k(1/\alpha_k)\\ &= -2(|\beta_k|/\alpha_k)\|g(x_k)\|^2 < 0, \text{ for } k \in \mathbb{N}. \end{aligned} \tag{8}$$

Hence, d_k is a descent direction for the function f for all $k \in \mathbb{N}$. Using now *(a)* and the definition of g, f and d_k, we obtain

$$\begin{aligned} f(x_k + \lambda d_k) &= \|g(x_k + \lambda d_k)\|^2\\ &= \|A(x_k + \lambda d_k) - b\|^2\\ &= \|Ax_k + (\lambda/\alpha_k)\text{sgn}_k Ar_k - b\|^2\\ &= \|g(r_k) + (\lambda/\alpha_k)\text{sgn}_k Ar_k\|^2\\ &= \|r_k - (\lambda/\alpha_k)\text{sgn}_k Ar_k\|^2. \end{aligned}$$

Therefore, *(b)* holds.

Proof of (c). Using *(a)* and *(b)* it follows that

$$\|r_k - (\lambda/\alpha_k)\mathrm{sgn}_k A r_k\|^2 \leq \|r_k\|^2 + \eta_k - \gamma\lambda^2\|r_k\|^2$$

is equivalent to

$$f(x_k + \lambda d_k) \leq f(x_k) + \eta_k - \gamma\lambda^2\|g(x_k)\|^2.$$

Consequently, the iterates $x_{k+1} = x_k + \lambda_k d_k$ and λ_k satisfy (7), and *(c)* holds. $\square$

By Proposition 2.1 it is clear that Algorithm 2.1 can be viewed as an iterative process for finding stationary points of f, that coincide with solutions of $Ax = b$. In that sense, the convergence analysis for Algorithm 2.1 consists in proving that the sequence of iterates $\{x_k\}_{k\in I\!N}$ is such that $\lim_{k\to\infty} \nabla f(x_k) = 0$.

First we need to establish that Algorithm 2.1 is well defined.

Proposition 2.2. *Algorithm 2.1 is well defined.*

Proof. Since $\eta_k > 0$, then by the continuity of $f(x)$, the condition

$$f(x_k + \lambda d_k) \leq f(x_k) + \eta_k - \gamma\lambda^2\|g(x_k)\|^2,$$

is equivalent to

$$\|r_k - (\lambda/\alpha_k)\mathrm{sgn}_k A r_k\|^2 \leq \|r_k\|^2 + \eta_k - \gamma\lambda^2\|r_k\|^2,$$

that holds for $\lambda > 0$ sufficiently small. $\square$

Our next result guarantees that the whole sequence of iterates generated by Algorithm 2.1 is contained in a subset of $I\!R^n$.

Proposition 2.3. *The sequence $\{x_k\}_{k\in I\!N}$ generated by Algorithm 2.1 is contained in the set*

$$\Phi_0 = \{x \in I\!R^n : 0 \leq f(x) \leq f(x_0) + \eta\}. \tag{9}$$

Proof. Clearly $f(x_k) \geq 0$ for $k \in I\!N$. Hence, it suffices to prove that $f(x_k) \leq f(x_0) + \eta$ for $k \in I\!N$. For that we first prove by induction that

$$f(x_k) \leq f(x_0) + \sum_{i=0}^{k-1} \eta_i. \tag{10}$$

Equation (10) holds for $k = 1$. Indeed, since λ_1 satisfies (7) then

$$f(x_1) \leq f(x_0) + \eta_0.$$

Let us suppose that (10) holds for $k-1$ where $k \geq 2$. We will show that (10) holds for k. Using (7) and (10) we obtain

$$f(x_k) \leq f(x_{k-1}) + \eta_{k-1} \leq f(x_0) + \sum_{i=0}^{k-2} \eta_i + \eta_{k-1} = f(x_0) + \sum_{i=0}^{k-1} \eta_i,$$

which proves that (10) holds for $k \geq 2$. Finally, using (4) and (10) it follows that, for $k \geq 0$:

$$f(x_{k+1}) \leq f(x_0) + \sum_{i=0}^{k} \eta_i \leq f(x_0) + \eta,$$

and the result is established. $\square$

For our convergence results, we need the following proposition that is presented and established in [13] as Lemma 3.3. We include it here for the sake of completeness.

Proposition 2.4. *Let $\{a_k\}_{k\in \mathbb{N}}$ and $\{b_k\}_{k\in \mathbb{N}}$ be sequences of positive numbers satisfying*

$$a_{k+1} \leq (1+b_k)a_k + b_k \text{ and } \sum_{k=0}^{\infty} b_k < \infty.$$

Then, $\{a_k\}_{k\in \mathbb{N}}$ converges.

Proposition 2.5. *If Φ_0 is bounded and $\{x_k\}_{k\in \mathbb{N}}$ is generated by Algorithm 2.1, then*

$$\sum_{k=0}^{\infty} \|x_{k+1} - x_k\|^2 < \infty, \tag{11}$$

and,

$$\lim_{k\to\infty} \lambda_k \|g(x_k)\| = 0. \tag{12}$$

Proof. Using Proposition 2.3 and the fact that Φ_0 is bounded, $\{\|g(x_k)\|\}_{k\in \mathbb{N}}$ is also bounded. Since $\|x_{k+1} - x_k\| = \lambda_k \|g(x_k)\|$, then using (7) we have that

$$\|x_{k+1} - x_k\|^2 = \lambda_k^2 \|g(x_k)\|^2 \leq \frac{\eta_k}{\gamma} + \frac{1}{\gamma}(f(x_k) - f(x_{k+1})). \tag{13}$$

Since η_k satisfies (4) and Φ_0 is bounded, adding in both sides of (13) it follows that

$$\begin{aligned} \sum_{k=0}^{\infty} \|x_{k+1} - x_k\|^2 &\leq \frac{1}{\gamma}\sum_{k=0}^{\infty} \eta_k + \frac{1}{\gamma}\sum_{k=0}^{\infty}(f(x_k) - f(x_{k+1})) \\ &\leq \frac{\eta + f(x_0)}{\gamma} < \infty, \end{aligned}$$

which implies that

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0,$$

and so

$$\lim_{k\to\infty} \lambda_k \|g(x_k)\| = 0.$$

Hence, the proof is complete. $\square$

Proposition 2.6. *If Φ_0 is bounded and $\{x_k\}_{k\in I\!N}$ is generated by Algorithm 2.1, then the sequence $\{\|g(x_k)\|\}_{k\in I\!N}$ converges.*

Proof. Since $f(x_k) \geq 0$ and $(1+\eta_k) \geq 1$, for all $k \in I\!N$, then using (7) we have that

$$f(x_{k+1}) \leq f(x_k) + \eta_k \leq (1+\eta_k)f(x_k) + \eta_k.$$

Setting $a_k = f(x_k)$ and $b_k = \eta_k$, then it can also be written as

$$a_{k+1} \leq (1+b_k)a_k + b_k,$$

and $\sum_{k=0}^{\infty} b_k < \eta < \infty$. Therefore, by Proposition 2.4, the sequence $\{a_k\}_{k\in I\!N}$ converges, i.e., the sequence $\{f(x_k)\}_{k\in I\!N}$ converges. Finally, since $f(x) = \|g(x)\|^2$ and $\|g(x_k)\| \geq 0$, then the sequence $\{\|g(x_k)\|\}_{k\in I\!N}$ converges. □

We now present the main convergence result of this section. Theorem 2.1 shows that either the process terminates at a solution or it produces a sequence $\{r_k\}_{k\in I\!N}$ for which $\lim_{k\to\infty} r_k^t A r_k = 0$.

Theorem 2.1. *If Φ_0 is bounded, then Algorithm 2.1 terminates at a finite iteration i where $r_i = 0$, or it generates a sequence $\{r_k\}_{k\in I\!N}$ such that*

$$\lim_{k\to\infty} r_k^t A r_k = 0.$$

Proof. Let us assume that Algorithm 2.1 does not terminate at a finite iteration. By continuity, it suffices to show that all accumulation point $\bar{x}$ of the sequence $\{x_k\}_{k\in I\!N}$ it satisfies that $g(\bar{x})^t A g(\bar{x}) = 0$. Be $\bar{x}$ a accumulation point of $\{x_k\}_{k\in I\!N}$. Then, there exists an infinite set of indices $R \subset I\!N$ such that $\lim_{k\to\infty, k\in R} x_k = \bar{x}$.

From Proposition 2.5 we have that

$$\lim_{k\to\infty} \lambda_k \|g(x_k)\| = 0$$

that holds if

$$\lim_{k\to\infty} \|g(x_k)\| = 0, \tag{14}$$

or if

$$\liminf_{k\to\infty} \lambda_k = 0. \tag{15}$$

If (14) holds, the result follows immediately.

Let us assume that (15) holds. Then, there exists an infinite set of indices $K = \{k_1, k_2, k_3, \ldots\} \subseteq I\!N$ such that

$$\lim_{j\to\infty} \lambda_{k_j} = 0.$$

If $R \cap K = \emptyset$, then by the Proposition 2.5

$$\lim_{k\to\infty, k\in R} \|g(x_k)\| = 0.$$

Therefore, the thesis of the theorem is established.

Without loss of generality, we can assume that $K \subseteq R$. By the way λ_{k_j} is chosen in Algorithm 2.1, there exists an index $\bar{j}$ sufficiently large such that for all $j \geq \bar{j}$, there exists ρ_{k_j} $(0 < \sigma_{min} \leq \rho_{k_j} \leq \sigma_{max})$ for which $\lambda = \lambda_{k_j}/\rho_{k_j}$ does not satisfy condition (7), i.e.,

$$\begin{aligned} f\left(x_{k_j} + \frac{\lambda_{k_j}}{\rho_{k_j}} d_{k_j}\right) &> f(x_{k_j}) + \eta_{k_j} - \gamma \frac{\lambda_{k_j}^2}{\rho_{k_j}^2} \|g(x_{k_j})\|^2 \\ &\geq f(x_{k_j}) - \gamma \frac{\lambda_{k_j}^2}{\rho_{k_j}^2} \|g(x_{k_j})\|^2. \end{aligned}$$

Hence,

$$\frac{f(x_{k_j} + \frac{\lambda_{k_j}}{\rho_{k_j}} d_{k_j}) - f(x_{k_j})}{\lambda_{k_j}/\rho_{k_j}} > -\gamma \frac{\lambda_{k_j}}{\rho_{k_j}} \|g(x_{k_j})\|^2 \geq -\gamma \frac{\lambda_{k_j}}{\sigma_{min}} \|g(x_{k_j})\|^2.$$

By the Mean Value Theorem it follows that

$$\nabla f(x_{k_j} + t_{k_j} d_{k_j})^t d_{k_j} > -\gamma \frac{\lambda_{k_j}}{\sigma_{min}} \|g(x_{k_j})\|^2, \text{ for } j \geq \bar{j}, \tag{16}$$

where $t_{k_j} \in [0, \lambda_{k_j}/\rho_{k_j}]$ tends to zero when $j \to \infty$. By continuity and the definitions of β_k and d_k, we obtain

$$\lim_{j\to\infty} d_{k_j} = -\text{sgn}\left(\frac{g(\bar{x})^t A g(\bar{x})}{g(\bar{x})^t g(\bar{x})}\right)(1/\bar{\alpha}) g(\bar{x}), \tag{17}$$

where $\bar{\alpha} = \lim_{k\to\infty, k\in K} \alpha_k$. We can assume that $\bar{\alpha} > 0$. If $\bar{\alpha} = 0$, then by the definition of the α_k, the thesis of the theorem is established. Setting $\bar{d} = \lim_{j\to\infty} d_{k_j}$ and noticing that $(x_{k_j} + t_{k_j} d_{k_j}) \to \bar{x}$ when $j \to \infty$, then taking limits in (16) we have that

$$\nabla f(\bar{x})^t \bar{d} \geq 0. \tag{18}$$

Since $2(A^t g(\bar{x}))^t \bar{d} = \nabla f(\bar{x})^t \bar{d} < 0$, then $2g(\bar{x})^t A \bar{d} = 0$. However from (17)

$$\nabla f(\bar{x})^t \bar{d} = -(2/\bar{\alpha}) g(\bar{x})^t A g(\bar{x}) = 0.$$

Therefore, $g(\bar{x})^t A g(\bar{x}) = 0$, with that which the thesis of the theorem is established. $\square$

Theorem 2.1 guarantees that Algorithm 2.1 converges to a solution of (1) whenever the Rayleigh quotient of A,

$$c(x) = \frac{x^t A x}{x^t x}, \quad x \neq 0, \tag{19}$$

satisfies that $|c(r_k)| > 0$, for $k \geq 1$. If the matrix A is indefinite, then it could happen that Algorithm 2.1 generates a sequence $\{r_k\}_{k \in I\!N}$ that converges to the residual $\bar{r}$ so that $\bar{r}^t A \bar{r} = 0$ and $\bar{r} \neq 0$.

In our next result, we show the convergence of the algorithm when the symmetric part of A, $A_s = (A^t + A)/2$, is positive definite, which appears in several different applications. Of course, similar properties will hold when A_s is negative definite.

Theorem 2.2. *If Φ_0 is bounded and the matrix A_s is positive definite, then Algorithm 2.1 terminates at a finite iteration i where $r_i = 0$, or it generates a sequence $\{r_k\}_{k \in I\!N}$ such that*

$$\lim_{k \to \infty} r_k = 0.$$

Proof. Since A_s is positive definite, $r_k^t A r_k = r_k^t A_S r_k > 0$, $r_k \neq 0$, for all $k \in I\!N$. Then, by the Theorem 2.1 the Algorithm 2.1 terminates at a finite iteration i where $r_i = 0$, or it generates a sequence $\{r_k\}_{k \in I\!N}$ such that $\lim_{k \to \infty} r_k = 0$. □

To be precise, the next proposition shows that if A_s is positive definite, then in Algorithm 2.1 it holds that $\beta_k > 0$ and $d_k = (1/\alpha_k) r_k$, for all $k \in I\!N$.

Proposition 2.7. *Let the matrix A_s be positive definite, and let α_{min} and α_{max} be the smallest and the largest eigenvalues of A_s, respectively. Then the sequences $\{\beta_k\}_{k \in I\!N}$ and $\{d_k\}_{k \in I\!N}$, generated by Algorithm 2.1 satisfy that $d_k = (1/\alpha_k) r_k$, for $k \in I\!N$.*

Proof. It is well-known that the Rayleigh quotient of A satisfies, for any $x \neq 0$,

$$0 < \alpha_{min} \leq c(x) \leq \alpha_{max}. \tag{20}$$

By the definition of β_k we have that

$$\beta_k = c(r_k) \geq \alpha_{min} > 0, \text{ for } k \geq 0.$$

Moreover, since $\beta_k > 0$ for $k \geq 0$, then

$$d_k = \text{sgn}(\beta_k)(1/\alpha_k) r_k = (1/\alpha_k) r_k, \text{ for } k \geq 0.$$

□

Proposition 2.7 guarantees that the choice $d_k = (1/\alpha_k) r_k$ is a descent direction, when A_s is positive definite. This yields a simplified version of the algorithm for solving linear systems when the matrix has positive (or negative) definite symmetric part.

Algorithm 2.2. *Residual Algorithm 2 (RA2)*

Given: $x_0 \in I\!R^n$, $\alpha_0 > 0$, $\gamma \in (0, 1)$, $0 < \sigma_{min} < \sigma_{max} < 1$, $\{\eta_k\}_{k \in I\!N}$ such that (4) holds. Set $r_0 = b - A x_0$, and $k = 0$.

Step 1. If $r_k = 0$, stop the process;

Step 2. Set $\lambda \leftarrow 1$;

Step 3. If $\|r_k - (\lambda/\alpha_k)Ar_k\|^2 \leq \|r_k\|^2 + \eta_k - \gamma\lambda^2\|r_k\|^2$ go to Step 5;

Step 4. Choose $\sigma \in [\sigma_{min}, \sigma_{max}]$, set $\lambda \leftarrow \sigma\lambda$, and go to Step 3;

Step 5. Set $\lambda_k = \lambda$, $x_{k+1} = x_k + (\lambda_k/\alpha_k)r_k$, y $r_{k+1} = r_k - (\lambda_k/\alpha_k)_k Ar_k$;

Step 6. Set $\alpha_{k+1} = (r_k^t Ar_k)/(r_k^t r_k)$, $k = k+1$ and go to Step 1.

Remark 2.4. (i) Algorithm 2.2 is well defined.

(ii) The sequence $\{x_k\}_{k\in I\!N}$ generated by Algorithm 2.2 is contained in Φ_0.

(iii) Since $\alpha_{k+1} = c(r_k)$ and $c(x)$ satisfies (20), then

$$0 < \alpha_{min} \leq \alpha_k \leq \alpha_{max}, \tag{21}$$

for $k \geq 1$. Moreover, $0 < (\lambda_k/\alpha_k) \leq \sigma_{max}\alpha_{min}^{-1}$, for $k \geq 1$.

(iv) Since Algorithm 2.2 is a simplified version of the Algorithm 2.1 when A_s is positive definite, then its convergence is established by Theorem 2.2.

Proposition 2.8. *Let g and f be given by (2) and (3), respectively. Let us assume that $\{x_k\}_{k\in I\!N}$, $\{r_k\}_{k\in I\!N}$ and $\{\lambda_k\}_{k\in I\!N}$, are generated by Algorithm 2.2. Then the following properties hold*

(a) For each $k \in I\!N$:

$$\begin{aligned} g(x_k) &= -r_k, \\ g(x_{k+1}) &= g(x_k) - (\lambda_k/\alpha_k)Ag(x_k). \end{aligned}$$

(b) The vector

$$d_k = (1/\alpha_k)r_k = -(1/\alpha_k)g(x_k)$$

is a descent direction for f. Moreover,

$$f(x_k + \lambda d_k) = \|r_k - (\lambda/\alpha_k)Ar_k\|^2.$$

(c) The iterates $x_{k+1} = x_k + \lambda_k d_k$ and λ_k satisfy conditions (7), for $k \geq 0$.

Proof. Setting $\text{sgn}(\beta_k) = 1$ in the proof of Proposition 2.1 all the claims in *(a)*, *(b)* and *(c)* hold. $\square$

Proposition 2.9. *If $\{x_k\}_{k\in I\!N}$ is generated by Algorithm 2.2, then*

$$\nabla f(x_k)^t d_k \leq (-2\alpha_{min}/\alpha_{max})\|g(x_k)\|^2, \tag{22}$$

for $k \geq 1$.

Proof. From Algorithm 2.2 we have that

$$\begin{aligned} \nabla f(x_k)^t d_k &= (-2/\alpha_k) g(x_k)^t A g(x_k) \\ &\leq (-2/\alpha_{max}) g(x_k)^t A g(x_k) \\ &\leq (-2\alpha_{min}/\alpha_{max}) g(x_k)^t g(x_k) \\ &= (-2\alpha_{min}/\alpha_{max})\|g(x_k)\|^2, \end{aligned}$$

for $k \geq 1$. $\square$

3 Numerical experiments

We report on some numerical experiments that illustrate the performance of algorithm *RA2*, presented and analyzed previously, for solving non symmetric and positive (or negative) definite linear systems. In all experiments, computing was done on a Pentium IV at 3.0 GHz with MATLAB 6.0, and we stop the iterations when

$$\frac{\|r_k\|}{\|b\|} \leq \varepsilon, \tag{23}$$

where $0 < \varepsilon \ll 1$.

As we mentioned in the introduction, our proposal can be viewed as a new variant of the well-known Richardson's method, with some new features. First, we will compare the behavior of algorithm *RA2* with two different variations of Richardson's method, whose general iterative step from a given x_0 is given by

$$x_{k+1} = x_k + \lambda_k r_k,$$

where the residual vectors can be obtained recursively as

$$r_{k+1} = r_k - \lambda A r_k.$$

It is well-known that if we choose the steplengths as the inverse of the eigenvalues of A (i.e., $\lambda_k = \lambda_i(A)$ for $1 \leq i \leq n$), then in exact arithmetic the process terminates at the solution in at most n iterations. Due to roundoff errors, in practice, this process is repeated cyclically (i.e., $\lambda_k = \lambda_{i\, mod\, n}(A)$ for all $k \geq n$). In our results, this cyclic scheme will be reported as the *ideal Richardson's method.*

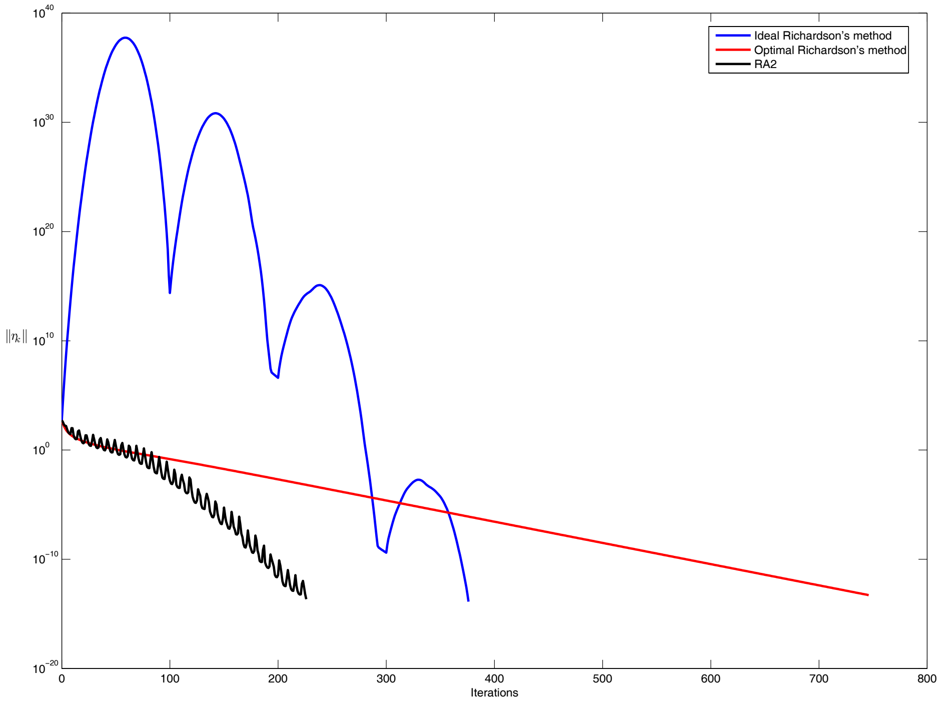

Figure 1: Behavior of the ideal and the optimal Richadson's method when compared with the *RA2* algorithm for $A = -Gallery('lesp', 100)$.

A recent variation on Richardson's method that has optimal properties concerning the norm of the residual, is discussed by Brezinski [6] and chooses the steplength as follows:

$$\lambda_k = \frac{r_k^T w_k}{w_k^T w_k},$$

where $w_k = Ar_k$. This option will be referred in our results as the *optimal Richardson's method.*

For our first experiment, we set $n = 100$, $b = \text{rand}(n, 1)$, $\varepsilon = 1.0D - 16$, and the matrix

$$A = -Gallery('lesp', n).$$

The results for this experiment are shown in Figure 1. We observe that the ideal Richardson's method is numerically unstable and requires several cycles to terminate the process. The optimal Richardson's method has a monotone behavior and it is numerically stable, but requires significantly more iterations than the *RA2* algorithm to reach the same accuracy. It is also worth noticing the nonmonotone behavior of the *RA2* algorithm that accounts for the fast convergence. Similar behavior was observed for the three methods on several positive definite test matrices.

We now present a comparison on several problems with well-known Krylov subspace methods. GMRES [23] and BiCGSTAB [26] are among the best-known Krylov iterative methods for solving large-scale non symmetric linear systems (see, e.g., [25, 27]). Therefore, we compare the performance of algorithm *RA2* with these two methods, without preconditioning and also taking advantage of two classical preconditioning strategies of general use: Incomplete LU (ILU) and SSOR.

In all the experiments described here we implemented GMRES with the restart parameters $m = 20$ (GMRES(20)) and $m = 40$ (GMRES(40)), and for all considered methods we use the vector $x_0 = 0$ as the initial guess. For algorithm *RA2* we use the following parameters: $\alpha_0 = \|b\|$, $\gamma = 10^{-4}$, $\sigma_{min} = 0.1$, $\sigma_{max} = 0.5$, and $\eta_k = 10^4\left(1 - 10^{-6}\right)^k$. For choosing a new λ at Step 4, we use the following procedure, described in [18]: given the current $\lambda_c > 0$, we set the new $\lambda > 0$ as

$$\lambda = \begin{cases} \sigma_{min}\lambda_c & \text{if } \lambda_t < \sigma_{min}\lambda_c, \\ \sigma_{max}\lambda_c & \text{if } \lambda_t > \sigma_{max}\lambda_c, \\ \lambda_t & \text{otherwise,} \end{cases}$$

where

$$\lambda_t = \frac{\lambda_c^2 f(x_k)}{f(x_k + \lambda_c d) + (2\lambda_c - 1) f(x_k)}.$$

In the following tables, the process is stopped when (23) is attained, but it can also be stopped prematurely for different reasons. We report the different possible failures observed with different symbols as follows:

* : The method reaches the maximum (20000) number of iterations.

** : The method stagnates (three consecutive iterates are exactly the same).

*** : Overflow is observed while computing one of the internal scalars.

For our second experiment we consider a set of 10 test matrices, described in Table 1, and we set the right hand side vector $b = (1, 1, \ldots, 1)^t \in I\!R^n$. In Table 1 we report the problem number (M), a brief description of the matrix, and the MATLAB commands to generate it.

We summarize on Table 2 the behavior of GMRES(20), GMRES(40), BICGSTAB and *RA2* without preconditioning. We have chosen $\varepsilon = 10^{-10}$ in (23) for stopping the iterations. We report the matrix number (M) from Table 1, the dimension of the problem (n); the number of computed residuals (CR), and the CPU time in seconds until convergence (T).

In Tables 3 and 4 we report the results for the matrices 4, 5 ,6, 7 ,8 and 9, when we use the following two preconditioning strategies.

(A) *Incomplete LU factorization with drop tolerance*
The preconditioning matrix is obtained, in MATLAB, with the command $[L, U]$ = luinc(A,0.5).

Table 1: First set of test matrices

M	Description	MATLAB Commands
1	Sparse adjacency matrix from NASA airfoil.	MATLAB demo: airfoil
2	Singular Toeplitz lower Hessenberg matrix	A=gallery('chow',n,1,1)
3	Circulant matrix	A=gallery('circul',v), where $v \in \mathbb{R}^n$ is such that $$v_i = \begin{cases} 10^{-6}, & i = 1, \\ 1, & i = n/2, \\ -1, & i = n, \\ 0, & \text{otherwise.} \end{cases}$$
4	Diagonally dominant, ill-conditioned, tridiagonal matrix	A=gallery('dorr',n,1)
5	Perturbed Jordan block	A=gallery('forsythe',n,-1,2)
6	Matrix whose eigenvalues lie on a vertical line in the complex plane	A=gallery('hanowa',n,n)
7	Jordan block	A=gallery('jordbloc',n,2)
8	Tridiagonal matrix with real sensitive eigenvalues	A = -gallery('lesp',n)
9	Pentadiagonal Toeplitz matrix	A=gallery('toeppen',n,1,10,n,-10,-1)
10	Upper triangular matrix discussed by Wilkinson and others	A=gallery('triw',n,-0.5,2)

(B) *The SSOR preconditioning strategy*
The preconditioning matrix is given by

$$(D - \omega E)D^{-1}(D - \omega F),$$

where $-E$ is the strict lower triangular part of A, $-F$ is the strict upper triangular part of A, and D is the diagonal part of A. We take $\omega = 1$.

In this case, we set $\varepsilon = 5 \times 10^{-15}$ in (23) for stopping the iterations. In Figures 2 and 3 we show the behavior of all considered methods when using preconditioning strategies (A) and (B), respectively, for problems 4-9.

Table 2: GMRES(20), GMRES(40), BICGSTAB and *RA2* without preconditioning

		GMRES(20)		GMRES(40)		BICGSTAB		*RA2*	
M	n	CR	T	CR	T	CR	T	CR	T
1	4253	60	0.406	60	0.375	**	**	64	0.047
2	1000	229	3.047	229	3.000	423	9.266	538	5.594
3	5000	*	*	*	*	1	0.016	2	0.000
4	500	20674	32.375	20674	32.594	549	0.188	19449	4.969
5	5000	28	0.313	28	0.234	77	0.141	29	0.016
6	5000	17	0.188	17	0.109	21	0.047	31	0.000
7	5000	27	0.297	27	0.188	62	0.125	28	0.000
8	5000	2562	22.203	2562	21.266	4740	13.688	10943	10.297
9	5000	4	0.094	4	0.031	4	0.031	4	0.000
10	5000	3067	26.172	3067	24.313	***	***	3408	3.203

Table 3: GMRES(20), GMRES(40), BICGSTAB and *RA2* with preconditioning (A)

		GMRES(20)		GMRES(40)		BICGSTAB		*RA2*	
M	n	CR	T	CR	T	CR	T	CR	T
4	50000	*	*	*	*	*	*	3	0.078
5	500000	38	61.438	48	100.906	81	38.484	20	4.875
6	500000	1	2.125	1	2.641	1	0.844	2	0.703
7	500000	37	59.391	38	87.656	72	34.031	19	4.656
8	500000	21	35.688	21	36.813	46	23.234	11	2.938
9	500000	2	3.266	2	3.813	3	2.250	2	0.875

For our third test problem, we consider the second order centered-differences discretization of

$$-\nabla^2 u + \gamma(xu_x + yu_y) + \beta u = f, \tag{24}$$

on the unit square, with homogeneous Dirichlet boundary conditions, $u = 0$, on the border of the region. We set the parameters $\gamma = 7100$ and $\beta = 100$ to guarantee that

Table 4: GMRES(20), GMRES(40), BICGSTAB and *RA2* with preconditioning (B)

		GMRES(20)		GMRES(40)		BICGSTAB		*RA2*	
M	n	CR	T	CR	T	CR	T	CR	T
4	50000	*	*	*	*	*	*	3	0.109
5	500000	25	185.953	25	186.609	46	263.625	10	60.703
6	500000	1	1.688	1	2.188	1	0.641	2	0.469
7	500000	21	37.156	21	38.203	26	16.922	10	4.828
8	500000	11	15.203	11	15.344	23	12.313	7	2.297
9	500000	2	5.516	2	6.594	3	4.203	2	2.734

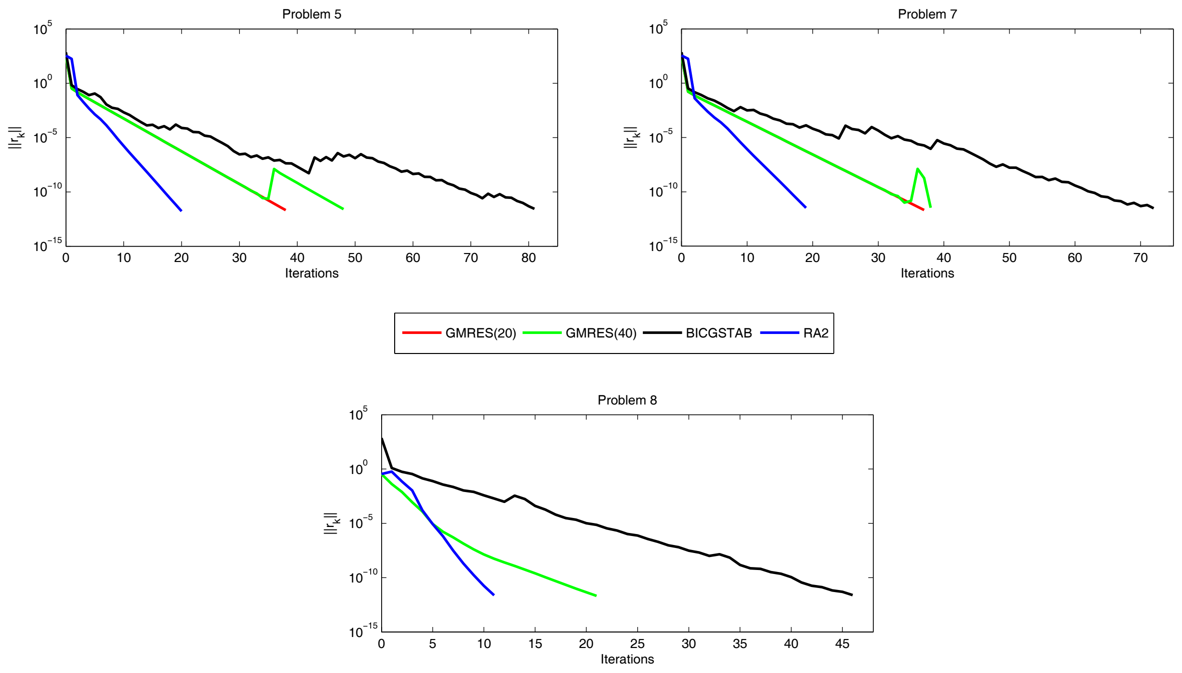

Figure 2: Behavior of all methods when using preconditioning techniques (A)

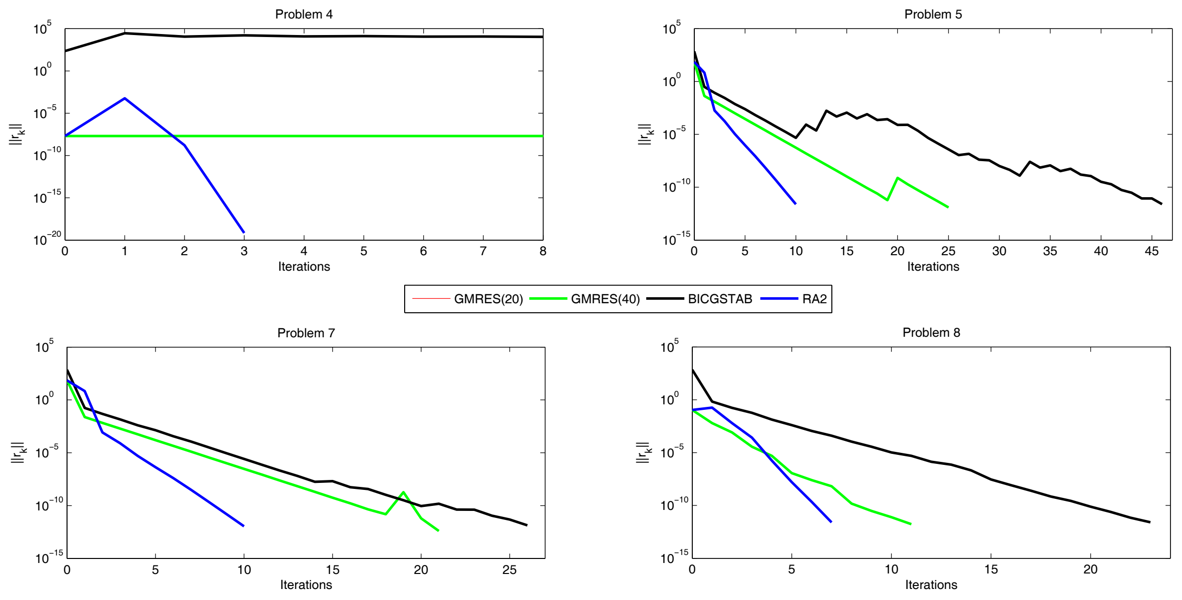

Figure 3: Behavior of all methods when using preconditioning techniques (B)

the symmetric part of the matrix is positive definite. The discretization grid has 71 internal nodes per axis producing an $n \times n$ matrix where $n = 5041$. The right hand side vector is chosen such that the solution vector is $x = (1, 1 \ldots, 1)^t$. In all the experiments we choose the initial guess as $x_0 = (0, 0, \ldots, 0)^t$. Once again we compare GMRES(20), GMRES(40), BICGSTAB and *RA2* with the preconditioning strategies (A) and (B).

In Table 5 we report the results obtained with GMRES, BICGSTAB and *RA2* for solving problem (24), when using the preconditioning strategies described in (A) and (B). We set $\varepsilon = 10^{-13}$ in (23) for stopping the iterations. In Figure 4 we show the behavior of all methods when using preconditioning techniques (A) and (B) for solving (24).

Table 5: GMRES(20), GMRES(40), BICGSTAB and *RA2* for solving (24)

	GMRES(20)		GMRES(40)		BICGSTAB		*RA2*	
Preconditioning strategy	CR	T	CR	T	CR	T	CR	T
(A)	*	*	730	12.984	*	*	123	1.094
(B)	*	*	195	25.016	*	*	19	2.266

We observe that, in general, *RA2* is a robust method for solving non symmetric linear systems whose symmetric part is positive (negative) definite. It is competitive with the well-known GMRES and BICGSTAB in number of computed residuals and CPU time, without preconditioning. We also observe that *RA2* outperforms GMRES and BICGSTAB when preconditioning strategies that reduce the number of cluster of eigenvalues are incorporated.

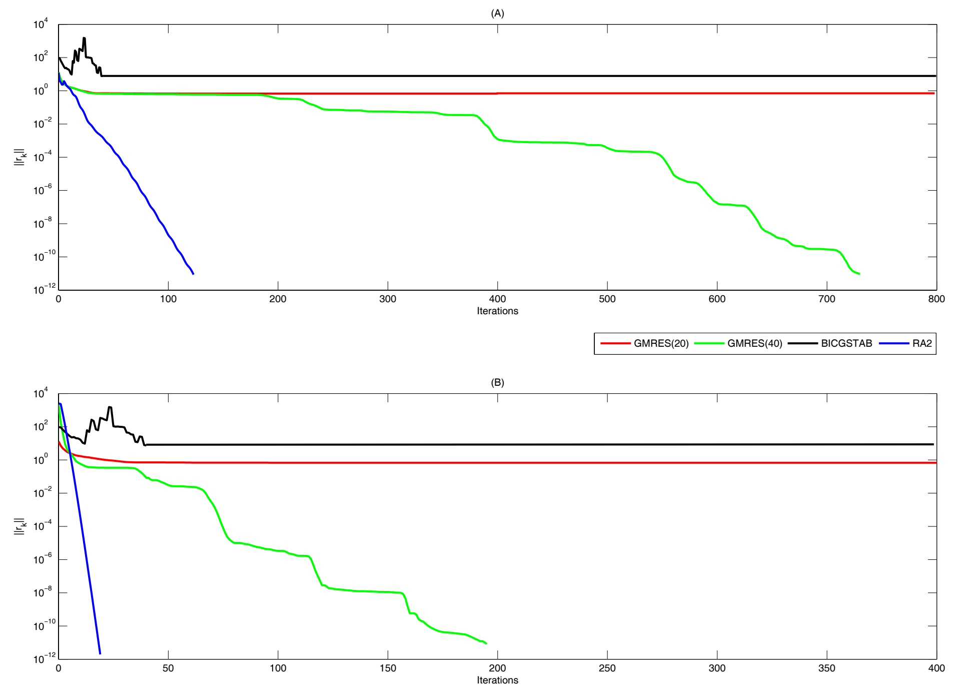

Figure 4: Behavior of all methods when using preconditioning techniques (A) and (B) for solving (24)

4 Conclusions

We present a residual algorithm ($RA2$) for solving large-scale nonsymmetric linear systems when the symmetric part of the coefficient matrix is positive (or negative) definite. Due to its simplicity, the method is very easy to implement, memory requirements are minimal and, so, its use for solving large-scale problems is attractive (MATLAB codes written by the authors are available by request).

We have compared the performance of the new residual method with GMRES and BICGSTAB, on some test problems, without preconditioning and also using two classical preconditioning strategies (ILU and SSOR). Our preliminary numerical results indicate that using the residual direction with a suitable step length can be competitive for solving large-scale problems, and preferable when the eigenvalues are clustered by a preconditioning strategy. Many new preconditioning techniques have been recently developed (see e. g., [25, 27] and references therein) that possess the clustering property when dealing with nonsymmetric matrices. In general, any preconditioning strategy that reduces the number of cluster of eigenvalues of the coefficient matrix (suitable for Krylov-subspace methods) should accelerate the convergence of the residual scheme introduced in this work. In that sense, the new residual method can be viewed as an extension of the preconditioned residual method, based on the Barzilai-Borwein choice of step length, introduced in [19].

For nonsymmetric systems with an indefinite symmetric part, the proposed general scheme $RA1$ can not guarantee convergence to solutions of the linear system, and usually convergence to points that satisfy $\lim_{k\to\infty} r_k^t A r_k = 0$, as predicted by Theorem 2.1, but such that $r_k \neq 0$, is observed in practice.

References

[1] J. Barzilai and J. M. Borwein (1988). Two-point step size gradient methods, *IMA Journal of Numerical Analysis 8*, 141–148.

[2] S. Bellavia, and B. Morini (2001). A globally convergent newton-gmres subspace method for systems of nonlinear equations. *SIAM J. Sci. Comput. 23*, 940–960.

[3] E. G. Birgin, and J. M. Martínez (2001). A spectral conjugate gradient method for unconstrained optimization. *Applied Mathematics and Optimization* 43, pp. 117–128.

[4] E. G. Birgin, and Y.G. Evtushenko (1998). Automatic differentiation and spectral projected gradient methods for optimal control problems. *Optimization Methods and Software* 10, pp. 125–146.

[5] E. G. Birgin, J. M. Martínez and M. Raydan (2000). Nonmonotone spectral projected gradient methods on convex sets. *SIAM Journal on Optimization* 10, pp. 1196–1211.

[6] C. Brezinski (1996). Variations on Richardson's method and acceleration. *Bull. Soc. Math. Belg.*, 33–44.

[7] C. Brezinski (1997). *Projection Methods for Systems of Equations.* North Holland, Amstrerdam.

[8] P. Brown, and Y. Saad (1990). Hybrid Krylov methods for nonlinear systems of equations. *SIAM J. Sci. Comp. 11*, 450–481.

[9] P. Brown, and Y. Saad (1994) Convergence theory of nonlinear Newton-Krylov algorithms. *SIAM Journal on Optimization 4*, 297–330.

[10] D. Calvetti and L. Reichel (1996). Adaptive Richardson iteration based on Leja points. *J. Comp. Appl. Math.*, 71, 267–286.

[11] D. Calvetti and L. Reichel (1996). An adaptive Richardson iteration method for indefinite linear systems. *Numer. Algorithms*, 12, 125–149.

[12] Y. H. Dai and L. Z. Liao (2002). R-linear convergence of the Barzilai and Borwein gradient method. *IMA Journal on Numerical Analysis* 22, pp. 1–10.

[13] J. E. Jr. Dennis, and J. J. Moré (1974). A characterization of superlinear convergence and its applications to quasi-Newton methods. *Math. of Comp.*, 28, 549–560.

[14] M. A. Diniz-Ehrhardt, M. A. Gomes-Ruggiero, J. M. Martínez and S. A. Santos (2004). Augmented Lagrangian algorithms based on the spectral gradient for solving nonlinear programming problems. *Journal of Optimization Theory and Applications* 123, pp. 497–517.

[15] R. Fletcher (2005). On the Barzilai-Borwein method. In: *Optimization and Control with Applications* (L.Qi, K. L. Teo, X. Q. Yang, eds.) Springer, 235–256.

[16] C. T. Kelley (1995). *Iterative Methods for Linear and Nonlinear Equations.* SIAM, Philadelphia.

[17] W. La Cruz and M. Raydan (2003). Nonmonotone spectral methods for large-scale nonlinear systems. *Optimization Methods and Software*, 18, 583–599.

[18] W. La Cruz, J. M. Martínez and M. Raydan (2006). Spectral residual method without gradient information for solving large-scale nonlinear systems. *Math. of Comp.*, 75, 1449–1466.

[19] B. Molina and M. Raydan (1996). Preconditioned Barzilai-Borwein method for the numerical solution of partial differential equations. *Numerical Algorithms*, 13, pp. 45–60.

[20] M. Raydan (1993). On the Barzilai and Borwein choice of the steplength for the gradient method. *IMA Journal on Numerical Analysis 13*, 321–326.

[21] M. Raydan (1997). The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM Journal on Optimization* 7, pp. 26–33.

[22] L. Reichel (1991). The application of Leja points to Richardson iteration and polynomial preconditioning. *Linear Algebra Appl.*, 154-156, 389–414.

[23] Y. Saad and M. H. Shultz (1986). GMRES: generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7, 856–869.

[24] D. C. Smolarski and P. E. Saylor (1988). An optimum semi-iterative method for solving any linear system with a square matrix. *BIT*, 28, 163–178.

[25] Y. Saad (2003). *Iterative Methods for Sparse Linear Systems.* SIAM, Philadelphia.

[26] H. A. van der Vorst (1992). Bi-CGSTAB: a fast and smoothly convergent variant Bi-CG for the solution of non-symmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 13, 631–644.

[27] H. A. van der Vorst (2003). *Iterative Krylov Methods for Large Linear Systems.* Cambridge University Press.

[28] D. M. Young (1990). A historical review of iterative methods. In: *A History of Scientific Computing* (S. G. Nash, Eds.) Addison-Wesley Reading, MA, 180–194.