ELSEVIER Journal of Computational and Applied Mathematics 76 (1996) 31–53

JOURNAL OF
COMPUTATIONAL AND
APPLIED MATHEMATICS

Recursive interpolation algorithm: A formalism for solving systems of linear equations – II Iterative methods

A. Messaoudi

Laboratoire d'Analyse Numérique et d'Optimisation, UFR IEEA-M3, Université des Sciences et Technologies de Lille, F-59655 Villeneuve d'Ascq Cedex, France

Received 5 July 1995; revised 2 February 1996

Abstract

This paper presents a simple unifying algorithm for solving systems of linear equations. Solving a system of linear equations will be interpreted as an interpolation problem. This new approach led us to a general algorithm called the recursive interpolation algorithm RIA. In Part I we gave the connection between this algorithm and known direct methods; in this part the truncated and the restarted versions of the RIA will be given. We will also show how to choose two free sets of parameters in the RIA for recovering some iterative methods.

Keywords: Recursive interpolation algorithm; Iterative methods

AMS classification: 65F10

1. Introduction

The first contribution is to present a unified approach to the majority of the existing algorithms for solving systems of linear equations. They are embedded in a general algorithm, the RIA where they correspond to particular choices of two free sets of parameters. The RIA contains essentially all possible algorithms with the following property: they can solve, in exact arithmetic, a linear system starting from an arbitrary point and in a number of iterations no greater than the number of equations. The majority of the direct and iterative methods proposed in the literature have this property and fall therefore into the RIA class. The second contribution of this paper lies in new formulations of classical algorithms that may be computationally attractive and compete with classical formulations. In Part I the RIA and some of its properties have been studied, the connection between this algorithm and known direct methods has been given. The paper is organized as follows: we recall the RIA and some of its properties studied in Part I. We give also the restarted and the truncated versions. Section 3 is concerned with the application of the RIA to systems of linear equations. We show how

PII S0377-0427(96)00071-4

to choose the free sets of parameters in the RIA for recovering some iterative methods. We also consider the use of preconditioning in conjunction with some of the iterative methods.

2. Recursive interpolation algorithm: RIA

Let us consider the linear system

$$Ax = b, \tag{2.1}$$

where $A = (a_{i,j})$ is $n \times n$ real nonsingular matrix and $b = (b_1, \dots, b_n)^{\mathrm{T}}$ is a given vector of $\mathbb{R}^n$. We will recall the formulation of the RIA and give its restarted and truncated versions.

2.1. Formulation of the RIA

Let $u_1, \dots, u_n$, $z_1, \dots, z_n$ be elements of $\mathbb{R}^n$. We denote by $\langle \cdot, \cdot \rangle$ the usual inner product in $\mathbb{R}^n$ and by $\| \cdot \|$ the corresponding norm. We assume that the vectors $u_1, \dots, u_n$ are linearly independent. The vector interpolation problem [6, 7, 17, 23, 24] is defined as follows:

Let x_0 be an arbitrary vector of $\mathbb{R}^n$. For $k \leqslant n$ find the vector x_k of $\mathbb{R}^n$ such that

$$x_k = x_0 + \sum_{i=1}^{k} a_i u_i \tag{2.2}$$

and for $j = 1, \dots, k$

$$\langle z_j, x_k \rangle = \langle z_j, x \rangle. \tag{2.3}$$

This problem is solved by the recursive interpolation algorithm (RIA) which is described as follows [24].

Algorithm 1: RIA

```
$x_0$ is an arbitrary vector, $g_{0,1} = u_1$,
for $m = 1, \dots, k$,
    $x_m = x_{m-1} + \frac{\langle z_m, x - x_{m-1} \rangle}{\langle z_m, g_{m-1,m} \rangle} g_{m-1,m}$,
    $g_{0,m+1} = u_{m+1}$,
    for $i = 1, \dots, m$,
        $g_{i,m+1} = g_{i-1,m+1} - \frac{\langle z_i, g_{i-1,m+1} \rangle}{\langle z_i, g_{i-1,i} \rangle} g_{i-1,i}$,
    end $i$,
end $m$.
```

Remark 2.1. We see that

$$g_{i,m} = u_m - \sum_{j=1}^{i} \frac{\langle z_j, g_{j-1,m} \rangle}{\langle z_j, g_{j-1,j} \rangle} g_{j-1,j}. \tag{2.4}$$

The process used in the RIA for computing the vectors $g_{m-1,m}$ for $m=1,\ldots,n$ can be interpreted as a process for constructing a new basis of $\mathbb{R}^n$ from the old basis u_m. The vectors $g_{m-1,m}$ generated by the RIA can be normalized to have length 1.

In the formulation of the RIA we have assumed that $D_k=[\langle z_i,u_j\rangle]_{1\leqslant i,j\leqslant k}$ is a strongly nonsingular matrix [24] (i.e. no break-down), in general this condition is not satisfied and the RIA will fail. For avoiding this difficulty we can use a pivoting strategy [24]. Some important properties of the RIA have been studied in [23, 24]. We set for $m=1,\ldots,k$,

$$U_m=[u_1,\ldots,u_m], \tag{2.5}$$

$$Z_m=[z_1,\ldots,z_m], \tag{2.6}$$

$$G_m=[g_{0,1},\ldots,g_{m-1,m}], \tag{2.7}$$

$$Q_m=U_m[Z_m^{\mathrm{T}}U_m]^{-1}Z_m^{\mathrm{T}} \tag{2.8}$$

$$Q_m'=G_m[Z_m^{\mathrm{T}}G_m]^{-1}Z_m^{\mathrm{T}}. \tag{2.9}$$

where Z^{T} is the adjoint of the matrix Z. It is easy to see that

$$x_m=x_0+Q_m(x-x_0), \tag{2.10}$$

$$g_{m,i}=(I-Q_m)u_i \quad \text{for } i>m. \tag{2.11}$$

It has been proved that [24]

$$x_m=x_0+Q_m'(x-x_0), \tag{2.12}$$

$$g_{m,i}=(I-Q_m')u_i \quad \text{for } i>m. \tag{2.13}$$

Many iterative methods for solving systems of linear equations are particular cases of the RIA. An important particular case of the RIA is defined by choosing $z_m=Hg_{m-1,m}$, where H is assumed to be a symmetric matrix. Let $u_1,\ldots,u_n$ be linearly independent vectors of $\mathbb{R}^n$ and Q_m' be the matrix defined by (2.9). Then for this particular case we have the following result [24].

Proposition 2.2. *If for $m=1,\ldots,k$, we choose $z_m=Hg_{m-1,m}$, where H is assumed to be a symmetric matrix, and if D_k is a strongly nonsingular matrix, then we have*

(1) $Q_m'^{\mathrm{T}}H=HQ_m'$.
(2) $\langle z_m,x-x_{m-1}\rangle=\langle z_m,x-x_0\rangle=\langle Hu_m,x-x_{m-1}\rangle$.
(3) $\langle z_m,g_{m-1,i}\rangle=\langle Hu_m,g_{m-1,i}\rangle=\langle z_m,u_i\rangle$, *for* $i\geqslant m$.
(4) $Z_m^{\mathrm{T}}U_m$ *is an upper triangular matrix.*
(5) $Z_m^{\mathrm{T}}G_m$ *is a diagonal matrix.*

Remark 2.3. We see that the choice $z_m=Hg_{m-1,m}$ or $z_m=Hu_m$ leads to mathematically equivalent algorithms.

2.2. The restarted and the truncated versions of the RIA

A difficulty with the RIA is that, in general, it becomes increasingly expensive as the step number k increases. There are two distinct ways of avoiding this difficulty. The first is simply to restart the algorithm every s steps where s is some integer parameter no greater than k. The second is to truncate the process, by insisting that the new vectors $g_{i,m}$ will be expressed only by the previous s vectors. This leads to the restarted and the truncated versions of the RIA. The first will be denoted by RRIA(s), and the second by TRIA(s). The RRIA(s) is described as follows.

Algorithm 2: RRIA(s)

```
1. $x_0$ is an arbitrary vector, choose or compute $z_i$, $u_i$ for $i = 1,\ldots,s$,
2. $g_{0,1} = u_1$,
   for $m = 1,\ldots,s$,
      $x_m = x_{m-1} + \frac{\langle z_m, x - x_{m-1}\rangle}{\langle z_m, g_{m-1,m}\rangle} g_{m-1,m}$,
      $g_{0,m+1} = u_{m+1}$,
      for $i = 1,\ldots,m$,
         $g_{i,m+1} = g_{i-1,m+1} - \frac{\langle z_i, g_{i-1,m+1}\rangle}{\langle z_i, g_{i-1,i}\rangle} g_{i-1,i}$,
      end $i$,
   end $m$,
3. $x_0 = x_s$, choose or compute $z_i$, $u_i$ for $i = 1,\ldots,s$, go to 2.
```

For the truncated version of the RIA, we truncate $g_{i,m}$ given by the relation (2.4). We denote it by $\bar{g}_{i,m}$. Then for $i \leqslant s$ we have $\bar{g}_{i,m} = g_{i,m}$ and for $i > s$ we have

$$\begin{aligned}\bar{g}_{i,m} &= \bar{g}_{0,m} - \sum_{j=i-s+1}^{i} \frac{\langle z_j, \bar{g}_{j-1,m}\rangle}{\langle z_j, \bar{g}_{j-1,j}\rangle} \bar{g}_{j-1,j}\\ &= \bar{g}_{0,m} - \sum_{j=i-s}^{i-1} \frac{\langle z_j, \bar{g}_{j-1,m}\rangle}{\langle z_j, \bar{g}_{j-1,j}\rangle} \bar{g}_{j-1,j} + \frac{\langle z_{i-s}, \bar{g}_{i-s-1,m}\rangle}{\langle z_{i-s}, \bar{g}_{i-s-1,i-s}\rangle} \bar{g}_{i-s-1,i-s} - \frac{\langle z_i, \bar{g}_{i-1,m}\rangle}{\langle z_i, \bar{g}_{i-1,i}\rangle} \bar{g}_{i-1,i}\\ &= \bar{g}_{i-1,m} + \frac{\langle z_{i-s}, \bar{g}_{i-s-1,m}\rangle}{\langle z_{i-s}, \bar{g}_{i-s-1,i-s}\rangle} \bar{g}_{i-s-1,i-s} - \frac{\langle z_i, \bar{g}_{i-1,m}\rangle}{\langle z_i, \bar{g}_{i-1,i}\rangle} \bar{g}_{i-1,i}.\end{aligned}$$

We thus obtain the following truncated RIA.

Algorithm 3: TRIA(s)

```
$x_0$ is an arbitrary vector, $\bar{g}_{0,1} = u_1$,
for $m = 1,\ldots,k$,
   $x_m = x_{m-1} + \frac{\langle z_m, x - x_{m-1}\rangle}{\langle z_m, \bar{g}_{m-1,m}\rangle} \bar{g}_{m-1,m}$,
   $\bar{g}_{0,m+1} = u_{m+1}$,
 if m ≤ s do
   for i = 1,...,m,
     $\bar{g}_{i,m+1} = \bar{g}_{i-1,m+1} - \frac{\langle z_i, \bar{g}_{i-1,m+1}\rangle}{\langle z_i, \bar{g}_{i-1,i}\rangle}\bar{g}_{i-1,i},$
   end i,
if m > s do
  for i = 1,...,s,
    $\bar{g}_{i,m+1} = \bar{g}_{i-1,m+1} - \frac{\langle z_i, \bar{g}_{i-1,m+1}\rangle}{\langle z_i, \bar{g}_{i-1,i}\rangle}\bar{g}_{i-1,i},$
  end i,
  for i = s + 1,...,m,
    $\bar{g}_{i,m+1} = \bar{g}_{i-1,m+1} + \frac{\langle z_{i-s}, \bar{g}_{i-s-1,m+1}\rangle}{\langle z_{i-s}, \bar{g}_{i-s-1,i-s}\rangle}\bar{g}_{i-s-1,i-s} - \frac{\langle z_i, \bar{g}_{i-1,m+1}\rangle}{\langle z_i, \bar{g}_{i-1,i}\rangle}\bar{g}_{i-1,i},$
  end i,
end m.
```

3. Application: iterative methods

If we choose, in the RIA, z_m and u_m such that $\langle z_m, x - x_{m-1}\rangle$ is known, then from Proposition 2.1(7) in Part I [24], we know that there exists $k_0 \leqslant n$ such that $x_{k_0} = x$. For the mth iterate vector x_m we define the associate residual vector, $r_m = b - Ax_m$, given by

$$r_m = r_{m-1} - \frac{\langle z_m, x - x_{m-1}\rangle}{\langle z_m, g_{m-1,m}\rangle} A g_{m-1,m}. \tag{3.1}$$

For applying the RIA we have to solve two problems:

P1. The RIA must be well defined (i.e., D_k is a strongly nonsingular matrix).

P2. z_m must be choosen such that $\langle z_m, x - x_{m-1}\rangle$ is known.

The majority of the methods proposed in the literature, for solving (2.1), have the following property: they can solve, in exact arithmetic, the linear system (2.1) in a number of iterations no greater than n. There are two classes of these methods: direct methods and iterative methods. The direct methods have been studied in Part I [24]. The iterative methods will be obtained from the RIA by choosing z_m depending on the preceding iterations. We find in this class the conjugate gradient (CG) method [16], the conjugate residual (CR) method [13], the generalized conjugate residual (GCR) method [13], the conjugate gradient method applied to the normal equation with minimum residual (CGNR) [13], the conjugate gradient method applied to the normal equation with minimum error (CGNE) [13], the Daniel method [10], the generalized Daniel method studied in [19, 22], the Axelsson methods [3, 4], the Axelsson and Vassilevski methods [5], the Orthodir, Orthomin and Orthores methods of Young and Jea [30], the generalized conjugate directions methods [11], the orthogonal errors methods [15], a class of the scaled ABS algorithms [1]. Some other iterative methods can also be obtained by the RIA. We will also study the full orthogonalization method and the GMRES algorithm

[26], which are special cases of the RIA. Now we will give the choice of u_m and z_m to obtain these methods and we will also give a new formulation for some of them.

3.1. The generalized conjugate direction methods

The prototype for the class of conjugate direction algorithms was described by Fox, Huskey and Wilkinson; see [28]. They take A to be symmetric and define the linearly independent vectors $v_1,\dots,v_n$ to be A-conjugate (i.e., $\langle Av_i,v_j\rangle = 0$ for $i \neq j$ and $\langle Av_i,v_i\rangle \neq 0$). Starting with x_0, the solution x_n of (2.1) may be constructed by the following algorithm:

```
For $m = 1,\dots,n$,
    $r_{m-1} = b - Ax_{m-1}$,
    $x_m = x_{m-1} + \frac{\langle v_m, r_{m-1}\rangle}{\langle v_m, Av_m\rangle} v_m$,
end m.
```

Note that this algorithm is a particular case of the RIA with the choice $z_m = A^{\mathrm{T}}v_m$ and $g_{m-1,m} = v_m$; the vectors $g_{m-1,m}$ are not computed by the RIA. The point to be noted is that the vectors $v_1,\dots,v_n$ serve two purposes: first they provide directions along which the approximate solutions x_m are to be altered, and second they delineate the subspaces in which the residuals r_m are forced to lie. The essential part of the first generalization of the notion of conjugacy is to provide a second set of vectors to serve the second purpose.

Definition 3.1. Let A, V, W be nonsingular matrices, with $V=[v_1,\dots,v_n]$ and $W=[w_1,\dots,w_n]$. Then (V,W) is an A-conjugate pair if $W^{\mathrm{T}}AV$ is lower triangular matrix.

The first generalization of the conjugate directions method, due to Stewart [28], is described as follows:

```
For $m = 1,\dots,n$,
    $r_{m-1} = b - Ax_{m-1}$,
    $x_m = x_{m-1} + \frac{\langle w_m, r_{m-1}\rangle}{\langle w_m, Av_m\rangle} v_m$,
end m.
```

Note that this algorithm is also a particular case of the RIA with the choice $z_m = A^{\mathrm{T}}w_m$ and $g_{m-1,m} = v_m$. The second generalization is due to Dennis and Turner [11]. They present an unifying work for a class of conjugate direction algorithms for solving the system (2.1). The class under consideration consists of algorithms that minimize an error functional over a subspace at each step. This method was called the generalized conjugate directions (GCD) algorithm. This algorithm is stated as a means of finding the minimizer of the quadratic

$$q(x) = \tfrac{1}{2}\langle x, Hx\rangle - \langle h, x\rangle,$$

where H is assumed to be an SPD matrix. The GCD algorithm produces a sequence $p_1, p_2, \dots,$ of linearly independent, mutually H-conjugate directions (i.e., $\langle p_i, Hp_j\rangle = 0$, $i \neq j$). The GCD algorithm is described as follows:

```
$\bar{x}_0$ is an arbitrary vector, $\bar{r}_0 = h - H\bar{x}_0$,
for $m = 1, \ldots,$ until convergence do
    If $\bar{r}_{m-1} \neq 0$, get $d_m$ such that $\langle d_m, \bar{r}_{m-1} \rangle \neq 0$,
        $\bar{x}_m = \{\mathrm{argmin}(q(x)), x \in \mathrm{span}\{p_1, \ldots, p_{m-1}, d_m\}\}$,
        $p_m = \bar{x}_m - \bar{x}_{m-1}$,
        $\bar{r}_m = h - H\bar{x}_m = \bar{r}_{m-1} - Hp_m$,
end m.
```

It has been shown in [11] that this algorithm includes the standard and the preconditioned conjugate gradient algorithms [13, 16], the conjugate residual algorithm [13], Craig's method [13], Elman's generalized conjugate residual algorithm [13], Vinsome's Orthomin algorithm [29], the Orthodir algorithm of Young and Jea [30] and the GMRES algorithm of Saad and Schultz [26]. Now we will give the explicit algorithm [11].

ALGORITHM 4: GCD ALGORITHM

```
$\bar{x}_0$ is an arbitrary vector,   $\bar{r}_0 = h - H\bar{x}_0$,   $\beta_{1,0} = 0$,
for $m = 1, \ldots,$ until convergence do
    if $\bar{r}_{m-1} \neq 0$, get $d_m$ such that $\langle d_m, \bar{r}_{m-1} \rangle \neq 0$,
        $\beta_{m,j} = \dfrac{\langle d_m, Hp_j \rangle}{\langle Hp_j, p_j \rangle}$, for $m \geqslant 2$ and $j = 1, \ldots, m-1$,
        $\alpha_m = \dfrac{\langle d_m, \bar{r}_{m-1} \rangle}{\langle Hd_m, d_m - \sum_{j=1}^{m-1} \beta_{m,j} p_j \rangle}$,
        $\bar{x}_m = \bar{x}_{m-1} + \alpha_m \left( d_m - \sum_{j=1}^{m-1} \beta_{m,j} p_j \right)$,
        $p_m = \bar{x}_m - \bar{x}_{m-1}$,
        $\bar{r}_m = h - H\bar{x}_m = \bar{r}_{m-1} - Hp_m$,
end m.
```

Let us set $P_m = [p_1, \ldots, p_{m-1}, d_m]$. The following result has been proved in [11].

Proposition 3.2. (1) $\langle \bar{r}_m, p_j \rangle = 0, \ 1 \leqslant j \leqslant m$.

(2) $\langle p_i, Hp_j \rangle = 0, \ i \neq j$.

(3) $\langle p_i, Hp_i \rangle = \langle p_i, \bar{r}_0 \rangle$, *for all* i.

(4) $\bar{x}_m = \bar{x}_0 + P_m [P_m^{\mathrm{T}} H P_m]^{-1} P_m^{\mathrm{T}} \bar{r}_0$.

We have the following result.

Theorem 3.3. *If we choose, in the* RIA, $x_0 = \bar{x}_0$, $z_m = H(d_m - \sum_{j=1}^{m-1} \beta_{m,j} p_j)$ *and* $u_m = d_m$, *then we get the* GCD METHOD.

Proof. First let us remark that from Proposition 3.2 we have

$$\left\langle H\left(d_m-\sum_{j=1}^{m-1}\beta_{m,j}p_j\right), x-\bar{x}_{m-1}\right\rangle = \langle d_m, \bar{r}_{m-1}\rangle$$

and

$$\left\langle H\left(d_m-\sum_{j=1}^{m-1}\beta_{m,j}p_j\right), d_m-\sum_{j=1}^{m-1}\beta_{m,j}p_j\right\rangle = \left\langle Hd_m, d_m-\sum_{j=1}^{m-1}\beta_{m,j}p_j\right\rangle.$$

Then from this discussion it is sufficient to prove that $g_{m-1,m}=d_m-\sum_{j=1}^{m-1}\beta_{m,j}p_j$. For $m=1$ we have $g_{0,1}=u_1=d_1$. Assume that it is true for $m-1$. Then from (2.4), we have

$$\begin{aligned} g_{m-1,m} &= u_m-\sum_{j=1}^{m-1}\frac{\langle z_j, g_{j-1,m}\rangle}{\langle z_j, g_{j-1,j}\rangle}g_{j-1,j} \\ &= u_m-\sum_{j=1}^{m-1}\frac{\langle Hg_{j-1,j}, u_m\rangle}{\langle Hg_{j-1,j}, g_{j-1,j}\rangle}g_{j-1,j}. \end{aligned}$$

Note also that from the definition of p_j we have $g_{j-1,j}=p_j/\alpha_j$. So we get

$$\begin{aligned} g_{m-1,m} &= u_m-\sum_{j=1}^{m-1}\frac{\langle Hg_{j-1,j}, u_m\rangle}{\langle Hg_{j-1,j}, g_{j-1,j}\rangle}g_{j-1,j} \\ &= d_m-\sum_{j=1}^{m-1}\frac{\langle Hp_j/\alpha_j, d_m\rangle}{\langle Hp_j/\alpha_j, p_j/\alpha_j\rangle}p_j/\alpha_j \\ &= d_m-\sum_{j=1}^{m-1}\frac{\langle Hp_j, d_m\rangle}{\langle Hp_j, p_j\rangle}p_j. \qquad \square \end{aligned}$$

Note that $z_m=H(d_m-\sum_{j=1}^{m-1}\beta_{m,j}p_j)=Hg_{m-1,m}$, then we get a new formulation of the GCD algorithm which will be called the modified generalized conjugate directions (MGCD) algorithm.

Algorithm 5: MGCD algorithm

```
$x_0$ is an arbitrary vector, $\bar{r}_0 = h - Hx_0$,
for $m = 1, \ldots,$ until convergence do
    if $\bar{r}_{m-1} \neq 0$, get $g_{0,m}$ such that $\langle g_{0,m}, \bar{r}_{m-1}\rangle \neq 0$,
    for $i = 1, \ldots, m-1$,
        $g_{i,m} = g_{i-1,m} - \frac{\langle Hg_{i-1,i}, g_{i-1,m}\rangle}{\langle Hg_{i-1,i}, g_{i-1,i}\rangle} g_{i-1,i}$,
    end i,
```

Table 1
The particular cases of the GCD method

Algorithm	Assumptions	H	h	d_m
CG [16]	A SPD	A	b	r_{m-1}
CR [13]	A SPD	$A^{\mathrm{T}}A$	$A^{\mathrm{T}}b$	r_{m-1}
CGNE[a] [13]	A nonsingular	I	x	$A^{\mathrm{T}}r_{m-1}$
CGNR [13]	A nonsingular	$A^{\mathrm{T}}A$	$A^{\mathrm{T}}b$	$A^{\mathrm{T}}r_{m-1}$
GCR [12, 13]	A PD	$A^{\mathrm{T}}A$	$A^{\mathrm{T}}b$	r_{m-1}
PGCR [19, 22]	A nonsingular, H' SPD and K PD	$A^{\mathrm{T}}H'A$	$A^{\mathrm{T}}H'b$	$KA^{\mathrm{T}}H'r_{m-1}$
Daniel [10]	A nonsingular, H' and K SPD	$A^{\mathrm{T}}H'A$	$A^{\mathrm{T}}H'b$	$KA^{\mathrm{T}}H'r_{m-1}$
ORTHODIR [30]	A nonsingular, ZA SPD	ZA	Zb	$d_1 = q_0$, $d_m = Aq_{m-2}, m \geqslant 2$
ORTHOMIN [30]	A nonsingular, Z PD and ZA SPD	ZA	Zb	$d_m = r_{m-1}$,
GMRES[b] [26]	A nonsingular	$A^{\mathrm{T}}A$	$A^{\mathrm{T}}b$	$d_m = v_m$

[a] We have other equivalent formulation of this algorithm which is obtained from the RIA by choosing $z_m = u_m = A^{\mathrm{T}}r_{m-1}$, see Table 4.

[b] This identification will be proved in Secrion 3.5.

$$x_m = x_{m-1} + \frac{\langle g_{m-1,m}, \bar{r}_{m-1}\rangle}{\langle Hg_{m-1,m}, g_{m-1,m}\rangle} g_{m-1,m},$$

$$\bar{r}_m = h - Hx_m = \bar{r}_{m-1} - \frac{\langle g_{m-1,m}, \bar{r}_{m-1}\rangle}{\langle Hg_{m-1,m}, g_{m-1,m}\rangle} Hg_{m-1,m},$$

end m.

Note that from Proposition 2.2 and Remark 2.3 we get other equivalent formulations of this algorithm. The truncated version of the GCD algorithm [11] is given by the TRIA. Choosing H, h and d_m in the GCD method we get some well-known algorithms as the conjugate gradient (CG) algorithm of Hestenes and Stiefel [16], the version of conjugate gradient proposed by Daniel [10], the conjugate residual (CR) algorithm [13], the conjugate gradient algorithm applied to the normal equation with minimum residual (CGNR) [13], the conjugate gradient algorithm applied to the normal equation with minimum error (CGNE) or the Craig method [13]. The generalized conjugate residual (GCR) algorithm [12, 13], the Vinsome algorithm [29] and the preconditioned generalized conjugate residual (PGCR) algorithm proposed in [19, 22] are also particular cases of this method. In Table 1 we summarize the choices of H, h and d_m in the GCD method for obtaining these algorithms.

Now we consider the use of preconditioning techniques in conjunction with iterative methods. Let $M = M_1M_2$ denote a nonsingular matrix. The solution to (2.1) can be obtained by solving any of the following problems.

Left: $M^{-1}Ax = M^{-1}b$,

Right: $(AM^{-1})Mx = b$,

Split: $(M_1^{-1}AM_2^{-1})M_2x = M_1^{-1}b$.

Table 2
The preconditioned algorithms obtained from the GCD method

Algorithm	Assumptions	H	h	d_m
CGNE[a] with left preconditioning [13]	A and M nonsingular	I	x	$A^TM^{-T}M^{-1}r_{m-1}$
CGNE[a] with SPD split [13]	A nonsingular and M SPD	M	Mx	$M^{-1}A^TM^{-1}r_{m-1}$
CGNR with right preconditioning [13]	A and M nonsingular	A^TA	A^Tb	$M^{-1}M^{-T}A^Tr_{m-1}$
CGNR with SPD split [13]	A nonsingular and M SPD	$A^TM^{-1}A$	$A^TM^{-1}b$	$M^{-1}A^TM^{-1}r_{m-1}$
Concus et al. [9]	A and M SPD	A	b	$M^{-1}r_{m-1}$
Meijerink and Van Der Vorst [21]	A SPD	A	b	$[LL^T]^{-1}r_{m-1}$

[a] We have other equivalent formulations of these algorithms obtained from the RIA, see Table 4.

The use of such an auxiliary matrix is known as preconditioning. Table 2 gives us the choices of H, h and d_m in the GCD method for obtaining some known preconditioned algorithms.

Remark 3.4. The GCR(s) and the ORTHOMIN(s) algorithms as described in [12, 13] correspond, respectively, to the RRIA(s) and the TRIA(s) with the choice $H = A^TA$, $h = A^Tb$ and $d_m = r_{m-1}$. The preconditioned GCR(s) and ORTHOMIN(s) as described in [13] correspond, respectively, to the RRIA(s) and the TRIA(s) with the choice $H = A^TM_1^{-T}M_1^{-1}A$, $h = A^TM_1^{-T}M_1^{-1}b$ and $d_m = M_2^{-1}M_1^{-1}r_{m-1}$, and the GCR(s) and ORTHOMIN(s) with symmetric positive-definite split preconditioning as described in [13] correspond, respectively, to the RRIA(s) and the TRIA(s) with the choice $H = A^TM^{-1}A$, $h = A^TM^{-1}b$ and $d_m = M^{-1}r_{m-1}$.

3.2. The Axelsson and Vassilevski methods

For generalizing the conjugate gradient method, Axelsson proposed an algorithm in [3], which in [4] he generalized further to the generalized conjugate gradient least square algorithm. The Axelsson and Vassilevski method [5] is a general version of the Axelsson methods. It is described as follows

ALGORITHM 6: THE AXEL AND VASSIL METHOD

```
$\bar{x}_0$ is an arbitrary vector, $r_0 = b - A\bar{x}_0$, $p_0 = Mr_0$,
for $m = 1, \ldots,$ until convergence do
    $\bar{x}_m = \bar{x}_{m-1} + \sum_{j=0}^{m-1} \alpha_j^{(m-1)} p_j$,
    $p_m = Mr_m - \sum_{j=0}^{m-1} \beta_{m,j} p_j$,
end m,
```

where M is an auxiliary preconditioning matrix, $\alpha_j^{(m-1)}$ $(j=0,\dots,m-1)$ are choosen such that $\bar{x}_m$ minimizes $\langle r_m, Br_m\rangle$ and $\beta_{m,j}$ $(j=0,\dots,m-1)$ are choosen such that $\langle p_m, B'p_j\rangle = 0$ with B and B' SPD matrices. The following properties are proved in [5].

Proposition 3.5. (1) $\langle BAp_j, r_m\rangle = 0$, *for* $j=0,\dots,m-1$.

(2) $\beta_{m,j} = \langle B'p_j, Mr_m\rangle / \langle B'p_j, p_j\rangle$, *for* $j=0,\dots,m-1$.

(3) $\alpha_j^{(m-1)}$ *for* $j=0,\dots,m-1$ *are solution of the following system*:

$$\begin{bmatrix} \langle BAp_0, Ap_0\rangle & \cdots & \langle BAp_0, Ap_{m-1}\rangle \\ \cdots & \cdots & \cdots \\ \langle BAp_{m-1}, Ap_0\rangle & \cdots & \langle BAp_{m-1}, Ap_{m-1}\rangle \end{bmatrix} \begin{bmatrix} \alpha_0^{(m-1)} \\ \cdots \\ \alpha_{m-1}^{(m-1)} \end{bmatrix} = \begin{bmatrix} 0 \\ \cdots \\ \langle BAp_{m-1}, r_{m-1}\rangle \end{bmatrix}.$$

Theorem 3.6. *If we choose, in the* RIA, $x_0 = \bar{x}_0, z_m = A^{\mathrm{T}}BAp_{m-1}$ *and* $u_m = p_{m-1}$, *then we get the Axelsson and Vassilevski method.*

Proof. It is sufficient to show that

$$\frac{\langle z_m, x - x_{m-1}\rangle}{\langle z_m, g_{m-1,m}\rangle} g_{m-1,m} = \sum_{j=0}^{m-1} \alpha_j^{(m-1)} p_j.$$

We will prove it by induction; for $m=1$ we have

$$\frac{\langle z_1, x - x_0\rangle}{\langle z_1, g_{0,1}\rangle} g_{0,1} = \frac{\langle A^{\mathrm{T}}BAp_0, x - \bar{x}_0\rangle}{\langle A^{\mathrm{T}}BAp_0, p_0\rangle} p_0 = \frac{\langle BAp_0, r_0\rangle}{\langle BAp_0, Ap_0\rangle} p_0 = \alpha_0^{(0)} p_0.$$

Now we assume that it is true for $m-1$ and we prove it for m. Let H_m be the matrix given in Proposition 3.5; it is easy to see that $H_m = Z_m^{\mathrm{T}} U_m$ and

$$H_m = \begin{bmatrix} H_{m-1} & Z_{m-1}^{\mathrm{T}} u_m \\ z_m^{\mathrm{T}} U_{m-1} & \langle z_m, u_m\rangle \end{bmatrix},$$

then we get

$$H_m^{-1} = \begin{bmatrix} H_{m-1}^{-1} + H_{m-1}^{-1} Z_{m-1}^{\mathrm{T}} u_m (H_m/H_{m-1})^{-1} z_m^{\mathrm{T}} U_{m-1} H_{m-1}^{-1} & -H_{m-1}^{-1} Z_{m-1}^{\mathrm{T}} u_m (H_m/H_{m-1})^{-1} \\ -(H_m/H_{m-1})^{-1} z_m^{\mathrm{T}} U_{m-1} H_{m-1}^{-1} & (H_m/H_{m-1})^{-1} \end{bmatrix},$$

with

$$\begin{aligned} (H_m/H_{m-1}) &= \langle z_m, u_m\rangle - z_m^{\mathrm{T}} U_{m-1} H_{m-1}^{-1} Z_{m-1}^{\mathrm{T}} u_m \\ &= \langle z_m, u_m - U_{m-1}[Z_{m-1}^{\mathrm{T}} U_{m-1}]^{-1} Z_{m-1}^{\mathrm{T}} u_m\rangle \\ &= \langle z_m, g_{m-1,m}\rangle. \end{aligned}$$

Applying Proposition 3.5 and the fact that $\langle z_m, x - x_{m-1}\rangle = \langle BAp_{m-1}, r_{m-1}\rangle$ we get

$$\begin{bmatrix} \alpha_0^{(m-1)} \\ \cdots \\ \alpha_{m-1}^{(m-1)} \end{bmatrix} = H_m^{-1} \begin{bmatrix} 0 \\ \cdots \\ \langle BAp_{m-1}, r_{m-1}\rangle \end{bmatrix}$$

$$= \begin{bmatrix} -\langle z_m, x - x_{m-1}\rangle (H_m/H_{m-1})^{-1} H_{m-1}^{-1} Z_{m-1}^{\mathrm{T}} u_m \\ \langle z_m, x - x_{m-1}\rangle (H_m/H_{m-1})^{-1} \end{bmatrix},$$

and

$$\sum_{j=0}^{m-1} \alpha_j^{(m-1)} p_j = \langle z_m, x - x_{m-1}\rangle (H_m/H_{m-1})^{-1} (p_{m-1} - [p_0, \ldots, p_{m-2}][Z_{m-1}^{\mathrm{T}} U_{m-1}]^{-1} Z_{m-1}^{\mathrm{T}} u_m)$$

$$= \frac{\langle z_m, x - x_{m-1}\rangle}{\langle z_m, g_{m-1,m}\rangle} (u_m - U_{m-1}[Z_{m-1}^{\mathrm{T}} U_{m-1}]^{-1} Z_{m-1}^{\mathrm{T}} u_m)$$

$$= \frac{\langle z_m, x - x_{m-1}\rangle}{\langle z_m, g_{m-1,m}\rangle} g_{m-1,m}. \qquad \square$$

Remark 3.7. In the Axelsson and Vassilevski method the vectors p_m can be computed from the auxiliary vectors $g_{j,m}$ used in the RIA with the choice $z_m = B'p_{m-1}$ and $u_m = Mr_{m-1}$.

Then we have a new formulation of the Axelsson and Vassilevski method, which will be called the modified Axelsson and Vassilevski method.

Algorithm 7: the modified Axelsson and Vassilevski method

```
$x_0$ is an arbitrary vector,  $r_0 = b - Ax_0$,  $g_{0,1} = \bar{g}_{0,1} = Mr_0$,
for $m = 1, \ldots,$ until convergence do
    $x_m = x_{m-1} + \frac{\langle BA\bar{g}_{m-1,m}, r_{m-1}\rangle}{\langle BA\bar{g}_{m-1,m}, Ag_{m-1,m}\rangle} g_{m-1,m}$,
    $r_m = r_{m-1} - \frac{\langle BA\bar{g}_{m-1,m}, r_{m-1}\rangle}{\langle BA\bar{g}_{m-1,m}, Ag_{m-1,m}\rangle} Ag_{m-1,m}$,
    $\bar{g}_{0,m+1} = Mr_m$,
    for $i = 1, \ldots, m$,
        $\bar{g}_{i,m+1} = \bar{g}_{i-1,m+1} - \frac{\langle B'\bar{g}_{i-1,i}, \bar{g}_{i-1,m+1}\rangle}{\langle B'\bar{g}_{i-1,i}, \bar{g}_{i-1,i}\rangle} \bar{g}_{i-1,i}$,
    end i,
    $g_{0,m+1} = \bar{g}_{m,m+1}$,
    for $i = 1, \ldots, m$,
        $g_{i,m+1} = g_{i-1,m+1} - \frac{\langle A^{\mathrm{T}}BA\bar{g}_{i-1,i}, g_{i-1,m+1}\rangle}{\langle A^{\mathrm{T}}BA\bar{g}_{i-1,i}, g_{i-1,i}\rangle} g_{i-1,i}$,
    end $i$,
end $m$.
```

Remark that the truncated version of the Axelsson and Vassilevski method is given by the RIA with this choice and by truncating the sums for $\bar{x}_m$ and p_m. The methods proposed by Axelsson in [3, 4] are particular cases of the Axelsson and Vassilevski method.

3.3. The Orthodir and Orthomin methods

For solving an unsymmetric system by generalized CG methods, Young and Jea [30] distinguished between Orthodir, Orthomin and Orthores algorithms. The Orthodir algorithm is described as follows:

ALGORITHM 8: ORTHODIR ALGORITHM

```
$\bar{x}_0$ is an arbitrary vector,  $r_0 = b - A\bar{x}_0$,   $q_0 = r_0$,
for $m = 1, \ldots,$ until convergence do
    $\bar{x}_m = \bar{x}_{m-1} + \frac{\langle q_{m-1}, Zr_{m-1}\rangle}{\langle q_{m-1}, ZAq_{m-1}\rangle} q_{m-1}$,
    $r_m = r_{m-1} - \frac{\langle q_{m-1}, Zr_{m-1}\rangle}{\langle q_{m-1}, ZAq_{m-1}\rangle} Aq_{m-1}$,
    $q_m = Aq_{m-1} - \sum_{i=0}^{m-1} \beta_{m,i} q_i$,
    with $\beta_{m,i} = \frac{\langle q_i, ZA^2 q_{m-1}\rangle - \sum_{j=0}^{i-1} \beta_{m,j}\langle q_i, ZAq_j\rangle}{\langle q_i, ZAq_i\rangle}$
end $m$,
```

where Z is an auxiliary matrix such that ZA is PD matrix. If ZA is SPD then we get the algorithm proposed by Ashby et al. [2]. The orthogonal error method [15] which is an extension of the work in [14] is also a particular case of the ORTHODIR algorithm. It is described as follows. Let B be a nonsingular matrix and $F(B) = \{\langle Bx, x\rangle / x \in / R^n\}$ be the field of values of B. We assume that $0 \notin F(B)$ then the orthogonal error method is given by the ORTHODIR algorithm with $ZA = B$. For the ORTHODIR algorithm we have the following result.

Theorem 3.8. *If we choose, in the* RIA, $x_0 = \bar{x}_0, Z_m = A^{\mathrm{T}}Z^{\mathrm{T}}g_{m-1}, u_1 = g_0$ *and* $u_m = Ag_{m-2}$ *for* $m \geqslant 2$, *then we get the* ORTODIR ALGORITHM.

Proof. It is sufficient to show that $g_{m-1,m} = q_{m-1}$ for $m \geqslant 1$. We proceed by induction, for $m = 1$ we have $g_{0,1} = u_1 = q_0$. Assume now that the property is true for $m \geqslant 1$. From the Proposition 2.3 in Part I [24], we have $Z_m^{\mathrm{T}}G_m$ is a lower triangular matrix. Then if we set $t = (t_{m,0}, t_{m,1}, \ldots, t_{m,m-1})^{\mathrm{T}} \in \mathbb{R}^m$ and $t = [Z_m^{\mathrm{T}}G_m]^{-1}Z_m^{\mathrm{T}}u_{m+1}$ we get for $i = 0, \ldots, m-1$

$$\begin{aligned} t_{m,i} &= \frac{\langle z_{i+1}, u_{m+1}\rangle - \sum_{j=0}^{i-1} t_{m,j}\langle z_{i+1}, g_{j,j+1}\rangle}{\langle z_{i+1}, g_{i,i+1}\rangle} \\ &= \frac{\langle q_i, ZA^2 q_{m-1}\rangle - \sum_{j=0}^{i-1} t_{m,j}\langle q_i, ZAq_j\rangle}{\langle q_i, ZAq_i\rangle} \\ &= \beta_{m,i}, \end{aligned}$$

and from (2.13) we have

$$\begin{aligned} g_{m,m+1} &= u_{m+1} - G_m[Z_m^{\mathrm{T}} G_m]^{-1} Z_m^{\mathrm{T}} u_{m+1} \\ &= Aq_{m-1} - \sum_{i=0}^{m-1} \beta_{m,i} q_i \\ &= q_m. \quad \square \end{aligned}$$

Then we have a new formulation of the Orthodir algorithm, which will be called the modified Orthodir algorithm. It is given as follows.

Algorithm 9: modified Orthodir algorithm

```
$x_0$ is an arbitrary vector,   $r_0 = b - Ax_0$,   $g_{0,1} = r_0$,
for $m = 1, \ldots,$ until convergence do
    $x_m = x_{m-1} + \frac{\langle g_{m-1,m}, Zr_{m-1}\rangle}{\langle g_{m-1,m}, ZAg_{m-1,m}\rangle} g_{m-1,m}$,
    $r_m = r_{m-1} - \frac{\langle g_{m-1,m}, Zr_{m-1}\rangle}{\langle g_{m-1,m}, ZAg_{m-1,m}\rangle} Ag_{m-1,m}$,
    $g_{0,m+1} = Ag_{m-1,m}$,
    for $i = 1, \ldots, m$,
        $g_{i,m+1} = g_{i-1,m+1} - \frac{\langle g_{i-1,i}, ZAg_{i-1,m+1}\rangle}{\langle g_{i-1,i}, ZAg_{i-1,i}\rangle} g_{i-1,i}$,
    end i,
end m.
```

Young and Jea [30] proposed another algorithm for solving the linear system (2.1). This algorithm was called the Orthomin algorithm, it is described as follows.

Algorithm 10: Orthomin algorithm

```
$\bar{x}_0$ is an arbitrary vector,   $r_0 = b - A\bar{x}_0$,   $p_0 = r_0$,
for $m = 1, \ldots,$ until convergence do
    $\bar{x}_m = \bar{x}_{m-1} + \frac{\langle p_{m-1}, Zr_{m-1}\rangle}{\langle p_{m-1}, ZAp_{m-1}\rangle} p_{m-1}$,
    $r_m = r_{m-1} - \dfrac{\langle p_{m-1}, Zr_{m-1}\rangle}{\langle p_{m-1}, ZAp_{m-1}\rangle} Ap_{m-1},$

    $p_m = r_m - \sum_{i=0}^{m-1} \beta_{m,i} p_i,$

    with $\beta_{m,i} = \dfrac{\langle p_i, ZAr_m\rangle - \sum_{j=0}^{i-1} \beta_{m,j}\langle p_i, ZAp_j\rangle}{\langle p_i, ZA\, p_i\rangle},$

  end m,
```

where Z and ZA are assumed to be PD matrices. We have the following result.

Theorem 3.9. *If we choose, in the* RIA, $x_0 = \bar{x}_0$, $z_m = A^{\mathrm{T}}Z^{\mathrm{T}}p_{m-1}$ *and* $u_m = r_{m-1}$, *then we get the* ORTHOMIN ALGORITHM.

Proof. The same proof as for Theorem 3.8.

Note that $z_m = A^{\mathrm{T}}Z^{\mathrm{T}}g_{m-1,m}$; then we have a new formulation of the ORTHOMIN algorithm which will be called the MODIFIED ORTHOMIN ALGORITHM.

ALGORITHM 11: MODIFIED ORTHOMIN ALGORITHM

```
  $x_0$ is an arbitrary vector,  $r_0 = b - Ax_0$,  $g_{0,1} = r_0$,
  for $m = 1, \ldots,$ until convergence do

    $x_m = x_{m-1} + \dfrac{\langle g_{m-1,m}, Zr_{m-1}\rangle}{\langle g_{m-1,m}, ZAg_{m-1,m}\rangle} g_{m-1,m},$

    $r_m = r_{m-1} - \dfrac{\langle g_{m-1,m}, Zr_{m-1}\rangle}{\langle g_{m-1,m}, ZAg_{m-1,m}\rangle} Ag_{m-1,m},$

    $g_{0,m+1} = r_m,$

    for $i = 1, \ldots, m,$

      $g_{i,m+1} = g_{i-1,m+1} - \dfrac{\langle g_{i-1,i}, ZAg_{i-1,m+1}\rangle}{\langle g_{i-1,i}, ZAg_{i-1,i}\rangle} g_{i-1,i},$

    end i,

  end m.
```

Note that the ORTHORES algorithm [30] is an other version of the generalized CG method. There are two important contributions specifying the cases where these algorithms are really useful in [18, 20]. The truncated versions of them are given by the TRIA.

3.4. The scaled ABS Algorithms

We will recall a particular class of the scaled ABS algorithms [1]. Let us consider the following system:

$$V^{\mathrm{T}}Ax = V^{\mathrm{T}}b, \tag{3.2}$$

where $V=[v_1,\ldots,v_n]$ is assumed to be a nonsingular matrix. The scaled ABS algorithms, defined for solving (3.2), are described as follows.

Algorithm 12: scaled ABS algorithms

```
  $\bar{x}_0$ is an arbitrary vector, $H_0$ is an arbitrary nonsingular matrix,
  for $m = 1,\ldots,$ until convergence do
    $p_m = H^{\mathrm{T}}_{m-1}q_m$, $q_m$ is chosen such that $\langle p_m, A^{\mathrm{T}}v_m\rangle \neq 0$,
    $\bar{x}_m = \bar{x}_{m-1} + \frac{\langle A^{\mathrm{T}}v_m, x - \bar{x}_{m-1}\rangle}{\langle A^{\mathrm{T}}v_m, p_m\rangle} p_m$,
    $H_m = H_{m-1} - H_{m-1}A^{\mathrm{T}}v_m w_m^{\mathrm{T}} H_{m-1}$,
    $w_m$ is chosen such that $\langle A^{\mathrm{T}}v_m, H^{\mathrm{T}}_{m-1}w_m\rangle = 1$,
  end m.
```

We are interested by the following particular choice of w_m:

$$w_m = \frac{q_m}{\langle A^{\mathrm{T}}v_m, H^{\mathrm{T}}_{m-1}q_m\rangle}. \tag{3.3}$$

This particular choice of w_m satisfies the condition $\langle A^{\mathrm{T}}v_m, H^{\mathrm{T}}_{m-1}w_m\rangle = 1$. Now we will give the choice of z_m and u_m in the RIA in the goal to obtain the scaled ABS algorithm with w_m defined by (3.3). Let us remark that

$$H_{m-1} = H_0 - \sum_{j=1}^{m-1} H_{j-1}\frac{A^{\mathrm{T}}v_j q_j^{\mathrm{T}}}{\langle A^{\mathrm{T}}v_j, H^{\mathrm{T}}_{j-1}q_j\rangle}H_{j-1},$$

and

$$\begin{aligned} p_m &= H^{\mathrm{T}}_{m-1}q_m \\ &= H_0^{\mathrm{T}}q_m - \sum_{j=1}^{m-1}\frac{\langle A^{\mathrm{T}}v_j, H^{\mathrm{T}}_{j-1}q_m\rangle}{\langle A^{\mathrm{T}}v_j, H^{\mathrm{T}}_{j-1}q_j\rangle}H^{\mathrm{T}}_{j-1}q_m \\ &= H_0^{\mathrm{T}}q_m - \sum_{j=1}^{m-1}\frac{\langle A^{\mathrm{T}}v_j, H^{\mathrm{T}}_{j-1}q_m\rangle}{\langle A^{\mathrm{T}}v_j, p_j\rangle}p_j. \end{aligned}$$

Then we have the following result.

Theorem 3.10. *Let w_m be given by (3.3). If we choose, in the* RIA, $x=\bar{x}_0$, $z_m=A^{\mathrm{T}}v_m$ *and* $u_m=H_0^{\mathrm{T}}q_m$, *then we get the corresponding class of the* SCALED ABS ALGORITHM.

Proof. From the above discussion it is sufficient to prove that $g_{j,m}=H_j^{\mathrm{T}}q_m$. We proceed by induction, for $j=1$ we have $g_{0,m}=u_m=H_0^{\mathrm{T}}q_m$. Now we assume that $g_{j-1,m}=H_{j-1}^{\mathrm{T}}q_m$ and we will prove it for j we have

$$\begin{aligned} g_{j,m} &= g_{j-1,m} - \frac{\langle z_j, g_{j-1,m}\rangle}{\langle z_j, g_{j-1,j}\rangle} g_{j-1,j} \\ &= H_{j-1}^{\mathrm{T}}q_m - \frac{\langle A^{\mathrm{T}}v_j, H_{j-1}^{\mathrm{T}}q_m\rangle}{\langle A^{\mathrm{T}}v_j, H_{j-1}^{\mathrm{T}}q_j\rangle} H_{j-1}^{\mathrm{T}}q_j \\ &= \left(H_{j-1} - H_{j-1}\frac{A^{\mathrm{T}}v_j q_j^{\mathrm{T}}}{\langle A^{\mathrm{T}}v_j, H_{j-1}^{\mathrm{T}}q_j\rangle} H_{j-1}\right)^{\mathrm{T}} q_m \\ &= H_j^{\mathrm{T}}q_m. \qquad \square \end{aligned}$$

Then we get a new formulation of ALGORITHM 12, which will be called the modified scaled ABS algorithm.

ALGORITHM **13:** MODIFIED SCALED ABS ALGORITHM

```
$x_0$ is an arbitrary vector,   $r_0 = b - Ax_0$,   $H_0$ is an arbitrary nonsingular matrix,
$g_{0,1} = H_0^{\mathrm{T}}q_1, q_1$ is chosen such that $\langle Ag_{0,1}, v_1\rangle \neq 0$, $m = 0$,
1. $m = m + 1$ until convergence do
    $x_m = x_{m-1} + \frac{\langle v_m, r_{m-1}\rangle}{\langle v_m, Ag_{m-1,m}\rangle} g_{m-1,m}$,
    $r_m = r_{m-1} - \frac{\langle v_m, r_{m-1}\rangle}{\langle v_m, Ag_{m-1,m}\rangle} Ag_{m-1,m}$,
    $q_{m+1}$ is chosen,
2. $g_{0,m+1} = H_0^{\mathrm{T}}q_{m+1}$,
    for $i = 1,\ldots,m$,
        $g_{i,m+1} = g_{i-1,m+1} - \frac{\langle v_i, Ag_{i-1,m+1}\rangle}{\langle v_i, Ag_{i-1,i}\rangle} g_{i-1,i}$,
    end i,
    if $\langle Ag_{m,m+1}, v_{m+1}\rangle \neq 0$ go to 1,
    otherwise change $q_{m+1}$ and go to 2.
```

Remark that the Stewart algorithm [28], which is also the Broyden algorithm [8], is obtained from ALGORITHM 13 with p_m arbitrary such that $\langle A^{\mathrm{T}}v_m, p_m\rangle \neq 0$.

3.5. Special case

In Section 2, $x_k - x_0$ has been expressed in an arbitrary system $\{u_1, \ldots, u_k\}$ by the formula (2.10). $x_k - x_0$ can be also expressed in any linearly independent system. This section is concerned with a special case, we construct a linearly independent system and we form $x_k - x_0$ in this system. The system to construct is a basis of a Krylov subspace $K_k = \{v_1, \ldots, v_k\}$ and $x_k - x_0$ will be expressed in this subspace as follows, $x_k = x_0 + V_k[V_k^{\mathrm{T}} A V_k]^{-1} V_k^{\mathrm{T}} A(x - x_0)$, with $V_k = [v_1, \ldots, v_k]$. We use the Arnoldi method for constructing this system and the Full Orthogalization method [26] for forming x_k in this system. The generalized minimal residual (GMRES) algorithm of Saad and Schultz [26] can be also given by using the RIA.

3.5.1. The Arnoldi method

Arnoldi's method [26] which uses the Gram–Schmidt method for computing an orthonormal basis $\{v_1, v_2, \ldots, v_k\}$ of the Krylov subspace $K_k = \mathrm{span}\{v_1, Av_1, \ldots, A^{k-1}v_1\}$ can be described as follows.

```
ALGORITHM 14: ARNOLDI'S METHOD
  1. Choose an initial vector $v_1$ with $\|v_1\| = 1$.
  2. For $m = 1, \ldots, k$, do
       $h_{j,m} = \langle v_j, Av_m \rangle, \; j = 1, \ldots, m,$
       $\tilde{v}_{m+1} = Av_m - \sum_{j=1}^{m} h_{j,m} v_j,$
       $h_{m+1,m} = \|\tilde{v}_{m+1}\|,$
       $v_{m+1} = \tilde{v}_{m+1} / h_{m+1,m}.$
```

For a symmetric matrix this method uses only a recurrence of tree terms and becomes the Lanczos method [25]. In practical implementation it is more suitable to replace the Gram–Schmidt algorithm of step 2 by the modified Gram–Schmidt; see [24, 27]. If V_k is the $n \times k$ matrix whose columns are $\{v_1, v_2, \ldots, v_k\}$, then $H_k = V_k^{\mathrm{T}} A V_k$, is the upper $k \times k$ Hessenberg matrix whose entries are the scalars $h_{j,m}$. Now we will show that the Arnoldi method can be given by the auxiliary vectors $g_{m-1,m}$ used in the RIA.

Theorem 3.11. *If we choose $z_m = v_m$ for $m=1, \ldots, k$, $u_1 = v_1$ and $u_m = Av_{m-1}$ for $m = 2, \ldots, k$, and if we use the process given in Remark* 2.1, *then we get the* ARNOLDI *method.*

Proof. It is sufficient to prove that $g_{m-1,m} = v_m$. For $m = 1$ we have $g_{0,1} = u_1 = v_1$. Now we assume that this property is true for $m \geqslant 1$ and we will prove it for $m + 1$. Invoking (2.13) we have $g_{m,m+1} = (I - Q'_m)u_{m+1}$, where $Q'_m = G_m[Z_m^{\mathrm{T}} G_m]^{-1} Z_m^{\mathrm{T}}$ and $Z_m^{\mathrm{T}} G_m = V_m^{\mathrm{T}} V_m = I_m$, then we get

$$\begin{aligned} g_{m,m+1} &= u_{m+1} - V_m V_m^{\mathrm{T}} u_{m+1} \\ &= Av_m - V_m V_m^{\mathrm{T}} Av_m \\ &= Av_m - [v_1, v_2, \ldots, v_m](h_{1,m}, h_{2,m}, \ldots, h_{m,m})^{\mathrm{T}} \end{aligned}$$

$$= Av_m - \sum_{j=1}^{m} h_{j,m} v_j$$

$$= \tilde{v}_{m+1},$$

and by normalizing $g_{m,m+1}$ we obtain the result. □

Note that for the ARNOLDI METHOD $z_m = g_{m-1,m} = v_m$ and from Proposition 2.2 and Remark 2.3 we can get other equivalent formulations. The process given by the RIA and corresponding to this choice is the MODIFIED ARNOLDI METHOD. It is described as follows.

ALGORITHM 15: THE MODIFIED ARNOLDI METHOD

```
1. Choose an initial vector $g_{0,1}$ with $\|g_{0,1}\| = 1$.
2. For $m = 1, \ldots, k$, do
     $g_{0,m+1} = Ag_{m-1,m}$,
     for $i = 1, \ldots, m$,
       $g_{i,m+1} = g_{i-1,m+1} - \langle g_{i-1,i}, g_{0,m+1} \rangle g_{i-1,i}$,
     end i,
     $g_{m,m+1} = g_{m,m+1} / \|g_{m,m+1}\|$.
```

3.5.2. *The full orthogonalization method*

In order to solve the linear system (2.1) by the Galerkin method using the orthonormal basis V_k, we seek an approximate solution x_k of the form $x_k = x_0 + t_k$, where x_0 is some initial guess to the solution x, and t_k is a vector of the Krylov subspace $K_k = \text{span}\{r_0, Ar_0, \ldots, A^{k-1}r_0\}$, with $r_0 = b - Ax_0$. Suppose that k steps of the ARNOLDI METHOD are carried out starting with $v_1 = r_0/\|r_0\|$. Then it is easily seen that the vector $t_k = V_k[V_k^{\mathrm{T}} A V_k]^{-1} V_k^{\mathrm{T}} A(x - x_0) = \|r_0\| V_k H_k^{-1} e_1$, where $e_1 = (1, 0, \ldots, 0) \in \mathbb{R}^k$. Hence, we can define the full orthogonalization method (FOM) [26] as follows.

ALGORITHM 16: FOM

```
1. Choose $x_0$ and compute $r_0 = b - Ax_0$ and $v_1 = r_0/\|r_0\|$.
2. For $m = 1, \ldots, k$, do
     $h_{j,m} = \langle v_j, Av_m \rangle$, $j = 1, \ldots, m$,
     $\tilde{v}_{m+1} = Av_m - \sum_{j=1}^{m} h_{j,m} v_j$,
     $h_{m+1,m} = \|\tilde{v}_{m+1}\|$,
     $v_{m+1} = \tilde{v}_{m+1} / h_{m+1,m}$.
3. $x_k = x_0 + V_k[V_k^{\mathrm{T}} A V_k]^{-1} V_k^{\mathrm{T}} A(x - x_0)$.
```

Invoking (2.10) we remark that for solving 3 of the full orthogonalization method we can use the RIA with the choice $z_m = A^{\mathrm{T}} v_m$ and $u_m = v_m$, for this choice the RIA can be fail down (i.e., break-down). To avoid this difficulty we use the PRIA. x_k can be also solved by using any direct method reviewed in Part I or any iterative method reviewed above with a choice of z_m and u_m such that the method obtained may be well defined (i.e., $Z_k^{\mathrm{T}}U_k$ is a strongly nonsingular matrix).

3.5.3. The generalized minimum residual (Gmres) *algorithm*

The generalized minimum residual (Gmres) method [26] is a projection method which minimizes the residual norm over all vectors in the affine subspace $x_0 + K_k$. To describe this algorithm we start by noticing that after k steps of Arnoldi's method we have an orthonormal system V_{k+1} and a $(k+1) \times k$ matrix $\bar{H}_k$ whose only nonzero entries are the elements $h_{j,m}$ generated by the method. Thus, $\bar{H}_k$ is the same as H_k except for an additional row whose only nonzero element is $h_{k+1,k}$ in the $(k+1,k)$ position. Writing any vector x in x_0+K_k in the form $x=x_0+z$, where z is an k-vector of K_k we define the least-squares problem

$$\min_{z \in K_k} \|b - A(x_0 + z)\| = \min_{z \in K_k} \|r_0 - Az\|. \tag{3.4}$$

If we set $z = V_k y$, we can view the norm to be minimized as the following function of y:

$$J(y) = \|\beta v_1 - A V_k y\|, \tag{3.5}$$

where we have let $\beta = \|r_0\|$. Using the fact that $AV_k = V_{k+1}\bar{H}_k$ and the fact that V_{k+1} is orthonormal we obtain

$$J(y) = \|\beta e_1 - \bar{H}_k y\|. \tag{3.6}$$

The vector e_1 is the first column of the $(k+1) \times (k+1)$ identity matrix. Hence, the solution of the least-squares problem (3.4) is given by $x_k = x_0 + V_k y_k$, where y_k minimizes $J(y)$ defined by (3.5), over $y \in \mathbb{R}^k$. For implementing x_k, Saad and Schultz [26] used the QR-factorization of $\bar{H}_k$ by using plane rotations of Givens, and solved an upper triangular system obtained from this factorization. Using the normalized QR-factorization studied in Part I [24], a new implementation of Gmres algorithm can be obtained, this question is under investigation. The restarted version of the Gmres algorithm [26] can be given by the restarted version of the ria. Now we will show that the Gmres algorithm can be also obtained by using the ria. For that let x_k be the kth iterate vector obtained by the Gmres algorithm. As $x_k - x_0$ is a vector of K_k and the corresponding residual vector r_k is minimum in norm, then we get

$$x_k = x_0 + \sum_{i=1}^{k} \alpha_{k,i} v_i, \tag{3.7}$$

where $(\alpha_{k,i})$ for $i = 1, \ldots, k$ are solution of the following linear system:

$$\begin{bmatrix} \|Av_1\|^2 & \langle Av_1, Av_2\rangle & \ldots & \langle Av_1, Av_k\rangle \\ \langle Av_2, Av_1\rangle & \|Av_2\|^2 & \ldots & \langle Av_2, Av_k\rangle \\ \ldots & \ldots & \ldots & \ldots \\ \langle Av_k, Av_1\rangle & \langle Av_k, Av_2\rangle & \ldots & \|Av_k\|^2 \end{bmatrix} \begin{bmatrix} \alpha_{k,1} \\ \alpha_{k,2} \\ \ldots \\ \alpha_{k,k} \end{bmatrix} = \begin{bmatrix} \langle Av_1, r_0\rangle \\ \langle Av_2, r_0\rangle \\ \ldots \\ \langle Av_k, r_0\rangle \end{bmatrix}. \tag{3.8}$$

The linear system (3.8) can be expressed in a matrix-form as follows:

$$[V_k^{\mathrm{T}} A^{\mathrm{T}} A V_k](\alpha_{k,.}) = V_k^{\mathrm{T}} A^{\mathrm{T}} A(x - x_0), \tag{3.9}$$

Table 3
The identification of some iterative methods with RIA

Algorithm	Assumptions	Choice of z_m	Choice of u_m
GCD	A nonsingular and H SPD	Hd_m or $H(d_m - \sum_{j=1}^{m-1} \beta_{m,j} p_j)$	d_m
Axel. and Vassil.	A nonsingular and B SPD	A^TBAp_{m-1}	p_{m-1}
Orthodir	A nonsingular and ZA PD	$A^TZ^Tq_{m-1}$	$u_1 = q_0$, $u_m = Aq_{m-2},\ m \geqslant 2$
Orthomin	A nonsingular, Z and ZA PD	$A^TZ^Tp_{m-1}$	r_{m-1}
Scaled ABS	A and H_0 nonsingular	A^Tv_m	$H_0^Tq_m$
Arnoldi	A nonsingular	v_m	$u_1 = v_1$, $u_m = Av_{m-1},\ m \geqslant 2$
FOM	A nonsingular	A^Tv_m	v_m
GMRES	A nonsingular	A^TAv_m or $A^TAg_{m-1,m}$	v_m

Table 4
The identification of the algorithms obtained by the GCD method with the RIA

Algorithm	Assumptions	Choice of z_m	Choice of u_m
CG	A SPD	Ap_{m-1} or Ar_{m-1}	r_{m-1}
CR	A SPD	A^TAp_{m-1} or A^TAr_{m-1}	r_{m-1}
CGNE	A nonsingular	p_{m-1} or A^Tr_{m-1}	A^Tr_{m-1}
CGNR	A nonsingular	A^TAp_{m-1} or $A^TAA^Tr_{m-1}$	A^Tr_{m-1}
GCR	A PD	A^TAp_{m-1} or A^TAr_{m-1}	r_{m-1}
Daniel	A nonsingular, H and K SPD	A^THAp_{m-1} or $A^THAKA^THr_{m-1}$	KA^THr_{m-1}
PGCR	A nonsingular, H SPD and K PD	A^THAp_{m-1} or $A^THAKA^THr_{m-1}$	KA^THr_{m-1}
CGNE with left preconditioning	A and M nonsingular	p_{m-1} or $A^TM^{-T}M^{-1}r_{m-1}$	$A^TM^{-T}M^{-1}r_{m-1}$
CGNE with SPD split	A nonsingular and M SPD	Mp_{m-1} or $A^TM^{-1}r_{m-1}$	$M^{-1}A^TM^{-1}r_{m-1}$
CGNR with right preconditioning	A and M nonsingular	A^TAp_{m-1} or $A^TAM^{-1}M^{-T}A^Tr_{m-1}$	$M^{-1}M^{-T}A^Tr_{m-1}$
CGNR with SPD split	A nonsingular and M SPD	$A^TM^{-1}Ap_{m-1}$ or $A^TM^{-1}AM^{-1}A^TM^{-1}r_{m-1}$	$M^{-1}A^TM^{-1}r_{m-1}$
Concus. et al.	A and M SPD	Ap_{m-1} or $AM^{-1}r_{m-1}$	$M^{-1}r_{m-1}$
Meijerink and Van Der Vorst	A SPD	Ap_{m-1} or $A[LL^T]^{-1}r_{m-1}$	$[LL^T]^{-1}r_{m-1}$

where $V_k^TA^TAV_k$ is a SPD matrix, and (3.7) can be written in a matrix form as

$$x_k = x_0 + V_k[V_k^TA^TAV_k]^{-1}V_k^TA^TA(x - x_0). \tag{3.10}$$

Then from (2.10) and (3.10) we can see immediately that x_k can be obtained from the RIA with the choice $z_m = A^TAv_m$ and $u_m = v_m$. From this discussion we have the following result.

Theorem 3.12. *If we choose, in the* RIA, $z_m = A^{\mathrm{T}}Av_m$ *and* $u_m = v_m$ *for* $m = 1,\ldots,k$, *then we get the* GMRES *algorithm.*

In Tables 3 and 4 we summarize the results about the identification of the RIA with the various iterative methods discussed in this paper.

A comparative study of the costs and the storage of these algorithms and their modified versions will be included in a forcoming work. This formalism can be applied in $\mathbb{R}^{2n}$ for giving so-called bi-recursive interpolation algorithm, which can be applied for giving some iterative methods like Lanczos methods. This problem is under investigation.

Acknowledgements

I am grateful to Professors C. Brezinski, K. Jbilou and H. Sadok for their helpful comments. I also would like to thank the referees for their helpful comments and valuable suggestions.

References

[1] J. Abaffy and E. Spedicato, *ABS Projection Algorithms: Mathematical Techniques for Linear and Nonlinear Equations* (Halsted Press, Chichester, 1989).

[2] S.F. Ashby, T.A. Manteuffel and P.E. Saylor, A taxonomy for conjugate gradient methods, *SIAM J. Numer. Anal.* **27** (1990) 1542–1568.

[3] O. Axelsson, Conjugate gradient type methods for unsymmetric and inconsistent systems of linear equations, *Linear Algebra Appl.* **29** (1980) 1–16.

[4] O. Axelsson, A generalized conjugate gradient, least square method, *Numer. Math.* **51** (1987) 209–227.

[5] O. Axelsson and P.S. Vasselevskil, A block box generalized conjugate gradient solver with inner iterations and variable-step preconditioning, *SIAM J. Matrix Anal. Appl.* **12** (1991) 625–644.

[6] C. Brezinski, Recursive interpolation, extrapolation and projection, *J. Comput. Appl. Math.* **9** (1983) 369–376.

[7] C. Brezinski, Other manifestations of the Schur complement, *Linear Algebra Appl.* **111** (1988) 231–247.

[8] C.G. Broyden, A class of methods for solving nonlinear simultaneous equations, *Math. Comput.* **19** (1965) 577–593.

[9] P. Concus, G.H. Golub and D.P. O'Leary, A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations, in: Eds., J.R. Bunch and D.J. Rose, *Sparse Matrix Computations* (Academic Press, New York, (1976).

[10] J.W. Daniel, The conjugate gradient method for linear and nonlinear equations, *SIAM J. Numer. Anal.* **4** (1967) 10–26.

[11] J.E. Dennis Jr. and K. Turner, Generalized conjugate directions, *Linear Algebra Appl.* **88/89** (1987) 187–209.

[12] S.C. Eisenstat, H.C. Elman and M.H. Schultz, Variational iterative methods for nonsymmetric systems of linear equations, *SIAM J. Numer. Anal.* **20** (1983) 345–357.

[13] H.C. Elman, Iterative methods for large sparse nonsymmetric systems of linear equations, Ph.D. Thesis, Computer Science Dept., Yale University, 1982.

[14] V. Faber and T.A. Manteuffel, Necessary and sufficient conditions for the existence of a conjugate gradient methods, *SIAM J. Numer. Anal.* **21** (1984) 352–362.

[15] V. Faber and T.A. Manteuffel, Orthogonal error methods, *SIAM J. Numer. Anal.* **24** (1987) 170–187.

[16] M. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. National Bur. Stand.* **49** (1952) 409–435.

[17] K. Jbilou, Recursive projection and interpolation algorithms. Application for solving linear systems, Publication ANO 303, Université de Lille 1, 1993.

[18] K.C. Jea and D.M. Young, On the simplification of generalized conjugate-gradient methods for nonsymmetrizable linear systems, *Linear Algebra Appl.* **52/53** (1983) 399–417.

[19] P. Joly and G. Meurant, Complex conjugate gradient methods, *Numer. Algorithms* **4** (1993) 379–406.

[20] W.D. Joubert and D.M. Young, Necessary and sufficient conditions for the simplification of generalized conjugate-gradient algorithms, *Linear Algebra Appl.* **88/89** (1987) 449–485.

[21] J.A. Meijerink and H.A. Van Der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, *Math. Comput.* **31** (1977) 148–162.

[22] A. Messaoudi, *Sur la décomposition incomplète de certaines classes de matrices; algorithmes itératifs associés*, Thèse Univ. de Franche Comté, 1983.

[23] A. Messaoudi, Some properties of the recursive projection and interpolation algorithms, *IMA J. Numer. Anal.* **15** (1995) 307–318.

[24] A. Messaoudi, Recursive interpolation algorithm: a formalism for solving systems of linear equations. Part I: direct methods, *J. Comput. Appl. Math.*, submitted.

[25] C.C. Paige and M.A. Sunders, Solution of sparse indefinite systems of linear equations, *SIAM J. Numer. Anal.* **12** (1975) 617–629.

[26] Y. Saad and M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear equations, *SIAM J. Sci. Statist. Comput.* **7** (1986) 856–869.

[27] H. Sadok, Méthodes de projection pour les systèmes linéaires et non linéaires, Thèse Université de Lille 1, 1994.

[28] G.W. Stewart, Conjugate direction methods for solving systems of linear equations, *Numer. Math.* **21** (1973) 285–297.

[29] P.K.W. Vinsome, Orthomin, an iterative method for solving sparse sets of simultaneous linear equations, in: *Society of Petroleum Engineers of AIME, Proc. 4th Symp. on Reservoir Simulation* (1976) 149–159.

[30] D.M. Young and D.M. Jea, Generalized conjugate-gradient acceleration of nonsymmetric iterative methods, *Linear Algebra Appl.* **34** (1980) 159–194.