Punjab University
Journal of Mathematics (ISSN 1016-2526)
Vol. 45 (2013) pp. 39- 49

An alternative derivation of a new Lanczos-type algorithm for systems of linear equations

Saifullah
Department of Mathematics
University of Peshawar
Khyber Pakhtunkhwa, 25120, Pakistan
Email: saifullah.maths@gmail.com

Muhammad Farooq
Department of Mathematics
University of Peshawar
Khyber Pakhtunkhwa, 25120, Pakistan
Email: mfarooq@upesh.edu.pk

Abdellah Salhi
Department of Mathematical Sciences
University of Essex, Wivenhoe Park
Colchester, CO4 3SQ, UK
Email: as@essex.ac.uk

Abstract. Various recurrence relations between formal orthogonal polynomials can be used to derive Lanczos-type algorithms. In this paper, we consider recurrence relation A_{12} for the choice $U_i(x) = P_i(x)$, where U_i is an auxiliary family of polynomials of exact degree i. It leads to a Lanczos-type algorithm that shows superior stability when compared to existing Lanczos-type algorithms. The new algorithm is derived and described. It is then computationally compared to the most robust algorithms of this type, namely A_{12}, A_5/B_{10} and A_8/B_{10}, on the same test problems. Numerical results are included.

AMS (MOS) Subject Classification Codes: 65F10
Key Words: Lanczos Algorithm; Systems of Linear Equations; Formal Orthogonal Polynomials.

1. Introduction

The Lanczos algorithm, [26, 1], has been designed to find the eigenvalues of a matrix. However, it has found application in the area of Systems of Linear Equations (SLE's) where it is now well established. It is an iterative process which, in exact arithmetic, finds the exact solution in at most n number of steps [27], where n is the dimension of the problem. Several Lanczos-type algorithms have been designed and among them, the famous conjugate gradient algorithm of Hestenes and Stiefel [25], when the matrix is Hermitian and the bi-conjugate gradient algorithm of Fletcher [22], in the general case. In the last few decades, Lanczos-type algorithms have evolved and different variants have been derived, which can be found in [2, 4, 6, 7, 8, 9, 10, 12, 15, 23, 24, 28, 29, 30, 31, 33, 14, 18].

Lanczos-type algorithms are commonly derived using Formal Orthogonal Polynomials (FOP's), [6]. The connection between the Lanczos algorithm, [27] and orthogonal polynomials, [32] has been studied extensively in [2, 6, 9, 10, 12, 5, 11, 13, 17].

1.1. **Notation.** The notation introduced by Baheux, in [2, 3], for recurrence relations with three terms is adopted here. It puts recurrence relations involving FOP's $P_k(x)$ (the polynomials of degree at most k with regard to the linear functional c) and/or FOP's $P_k^{(1)}(x)$ (the monic polynomials of degree at most k with regard to linear functional $c^{(1)}$, [13]) into two groups: A_i and B_j. Although relations A_i, when they exist, rarely lead to Lanczos-type algorithms on their own (the exceptions being A_4, [2, 3], and A_{12}, [18]), relations B_j never lead to such algorithms for obvious reasons. It is the combination of recurrence relations A_i and B_j, denoted as A_i/B_j, when both exist, that lead to Lanczos-type algorithms. In the following we will refer to algorithms by the relation(s) that lead to them. Hence, we will have, potentially, algorithms A_i and algorithms A_i/B_j, for some $i = 1, 2, \dots$ and some $j = 1, 2, \dots$.

The paper is organized as follows. In the next section, the background to the Lanczos process is presented. Section 3 is on FOP's. Section 4 is on algorithm A_{12}, [18] and the estimation of the coefficients of the recurrence relations A_{12} used to derive it. Section 5 is the estimation of the coefficients of recurrence relation A_{12}, [18], used to derive the new algorithm of the same name $i.e$ $A_{12}(new)$. Section 6 describes the test problems and reports the numerical results. Section 7 is the conclusion and further work.

2. The Lanczos Process

Consider the system of linear equations,

$$A\mathbf{x} = \mathbf{b}, \tag{2.1}$$

where A is $n \times n$ real matrix, $\mathbf{b}$ and $\mathbf{x}$ are vectors in R^n.

Choose $\mathbf{x}_0$ and $\mathbf{y}$, two arbitrary vectors in R^n, such that $\mathbf{y} \neq 0$. Then, Lanczos process [27] consists in generating a sequence of vectors $\mathbf{x}_k \in R^n$, such that

$$\mathbf{x}_k - \mathbf{x}_0 \in F_k(A, \mathbf{r}_0) = \mathtt{span}(\mathbf{r}_0, A\mathbf{r}_0, \dots, A^{k-1}\mathbf{r}_0), \tag{2.2}$$

and

$$\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k \perp G_k(A^T, \mathbf{y}) = \mathtt{span}(\mathbf{y}, A^T\mathbf{y}, \dots, (A^T)^{k-1}\mathbf{y}), \tag{2.3}$$

where A^T is the transpose of matrix A.

Equation (2.2) implies,

$$\mathbf{x}_k - \mathbf{x}_0 = -\beta_1\mathbf{r}_0 - \beta_2 A\mathbf{r}_0 - \cdots - \beta_k A^{k-1}\mathbf{r}_0. \tag{2.4}$$

Multiplying both sides by A and adding and subtracting $\mathbf{b}$ on the left hand side of (2.4) gives

$$\mathbf{r}_k = \mathbf{r}_0 + \beta_1 A\mathbf{r}_0 + \beta_2 A^2\mathbf{r}_0 + \cdots + \beta_k A^k\mathbf{r}_0. \tag{2.5}$$

If we set

$$P_k(x) = 1 + \beta_1 x + \cdots + \beta_k x^k,$$

then we can write from (2.5)

$$\mathbf{r}_k = P_k(A)\mathbf{r}_0. \tag{2.6}$$

From (2.3), since $(A^T)^i\mathbf{y}$ and $\mathbf{r_k}$ are each in orthogonal subspaces, we can write,

$$((A^T)^i\mathbf{y}, \mathbf{r_k}) = (\mathbf{y}, A^i\mathbf{r_k}) = (\mathbf{y}, A^iP_k(A)\mathbf{r_0}) = \mathbf{0}, \text{ for } i = 0, \ldots, k-1.$$

Thus, the coefficients $\beta_1, \ldots, \beta_k$ form a solution of system of linear equations,

$$\beta_1(\mathbf{y}, A^{i+1}\mathbf{r}_0) + \cdots + \beta_k(\mathbf{y}, A^{i+k}\mathbf{r}_0) = -(\mathbf{y}, A^i\mathbf{r}_0), \text{ for } i = 0, \ldots, k-1. \quad (2.7)$$

If the determinant of the above system is not zero then its solution exists and allows to obtain $\mathbf{x}_k$ and $\mathbf{r}_k$. Obviously, in practice, solving the above system directly for increasing values of k is not feasible; k is the order of the iterate in the solution process. We shall see now how to solve this system for increasing values of k recursively, that is, if polynomials P_k can be computed recursively. Such computation is feasible as the polynomials P_k form a family of FOP's and will now be explained. In exact arithmetic, k should not exceed n, where n is the dimension of the problem.

3. Formal Orthogonal Polynomials

Let c be a linear functional on the space of complex polynomials defined by

$$c(x^i) = c_i \text{ for } i = 0, 1, \ldots$$

where

$$c_i = ((A^T)^i\mathbf{y}, \mathbf{r}_k) = (\mathbf{y}, A^i\mathbf{r}_k) \text{ for } i = 0, 1, \ldots$$

Again, because of (2.3) above, an orthogonality condition can be written as,

$$c(x^iP_k) = 0 \text{ for } i = 0, \ldots, k-1. \quad (3.1)$$

This condition shows that P_k is the polynomial of degree at most k which is a FOP with respect to the functional c, [5].

Given the expression of $P_k(x)$ above, $P_k(0) = 1, \forall k$ is a normalization condition for these polynomials; P_k exists and is unique if the following Hankel determinant

$$H_k^{(1)} = \begin{vmatrix} c_1 & c_2 & \cdots & c_k \\ c_2 & c_3 & \cdots & c_{k+1} \\ \vdots & \vdots & & \vdots \\ c_k & c_{k+1} & \cdots & c_{2k-1} \end{vmatrix}$$

is not zero, in which case we can write $P_k(x)$ as

$$P_k(x) = \frac{\begin{vmatrix} 1 & x & \cdots & x^k \\ c_0 & c_1 & \cdots & c_k \\ \vdots & \vdots & & \vdots \\ c_{k-1} & c_k & \cdots & c_{2k-1} \end{vmatrix}}{\begin{vmatrix} c_1 & \cdots & c_k \\ \vdots & & \vdots \\ c_k & \cdots & c_{2k-1} \end{vmatrix}}, \quad (3.2)$$

where the denominator of this polynomial is $H_k^{(1)}$, the determinant of the system (2.7). We assume that $\forall\, k, H_k^{(1)} \neq 0$ and therefore all the polynomials P_k exist for all k. If for some k, $H_k^{(1)} = 0$, then P_k does not exist and breakdown occurs in the algorithm, [6, 9, 10, 12, 11].

A Lanczos-type method consists in computing P_k recursively, then $\mathbf{r}_k$ and finally $\mathbf{x}_k$ such that $\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k$, without inverting A. This gives the solution of the system (2.1) in at most n steps, where n is the dimension of the SLE. For more details, see [6, 8].

FOPs can be put together into recurrence relations. Such relations give rise to various procedures for the recursive computation of P_k and hence we get different Lanczos-type algorithms for computing $\mathbf{r}_k$ and, therefore, $\mathbf{x}_k$. These algorithms have been studied in [2, 6, 8, 9, 10, 12, 11, 3]. They differ by the recurrence relationships used to express the polynomials $P_k, k = 2, 3, ...$

4. Recurrence relation A_{12} based algorithm

Algorithms A_5/B_{10}, A_8/B_{10} and A_{12} are the most robust algorithms as found in [2, 18, 20, 21, 3], on the same problems considered here. We, therefore compare our results with these algorithms. Since the algorithm we introduce here is also based on the recurrence relation A_{12} [18, 19], according to the notation of [2], it is really a modification of algorithm A_{12} that can be found in [18]. Indeed, A_{12} is derived using the auxiliary polynomial $U_i(x) = x^i$, of exact degree i, while here we derive a new algorithm A_{12} but for $U_i(x) = P_i(x)$. For completeness, we recall algorithm A_{12} here but leave out its derivation which can be found by the interested reader in [18].

4.1. **Algorithm A_{12}.** Algorithm A_{12} [18] can be described as follows.

Algorithm 1 : Lanczos-type algorithm A_{12}

Choose $\mathbf{x}_0$ and $\mathbf{y}$ such that $\mathbf{y} \neq 0$,
Set $r_0 = b - Ax_0$, $y_0 = y$, $p = Ar_0$, $p_1 = Ap$, $c_0 = (y, r_0)$,
$c_1 = (y, p)$, $c_2 = (y, p_1)$, $c_3 = (y, Ap_1)$, $\delta = c_1c_3 - c_2^2$,
$\alpha = \frac{c_0c_3 - c_1c_2}{\delta}$, $\beta = \frac{c_0c_2 - c_1^2}{\delta}$,
$r_1 = r_0 - \frac{c_0}{c_1}p$, $x_1 = x_0 + \frac{c_0}{c_1}r_0$,
$r_2 = r_0 - \alpha p + \beta p_1$, $x_2 = x_0 + \alpha r_0 - \beta p$,
$y_1 = A^Ty_0$, $y_2 = A^Ty_1$, $y_3 = A^Ty_2$.
for k = 3, 4,..., **do**
$y_{k+1} = A^Ty_k$, $q_1 = Ar_{k-1}$, $q_2 = Aq_1$, $q_3 = Ar_{k-2}$,
$a_{11} = (y_{k-2}, r_{k-2})$, $a_{13} = (y_{k-3}, r_{k-3})$, $a_{21} = (y_{k-1}, r_{k-2})$, $a_{22} = a_{11}$,
$a_{23} = (y_{k-2}, r_{k-3})$, $a_{31} = (y_k, r_{k-2})$, $a_{32} = a_{21}$, $a_{33} = (y_{k-1}, r_{k-3})$,
$s = (y_{k+1}, r_{k-2})$, $t = (y_k, r_{k-3})$, $F_k = -\frac{a_{11}}{a_{13}}$,
$b_1 = -a_{21} - a_{23}F_k$, $b_2 = -a_{31} - a_{33}F_k$, $b_3 = -s - tF_k$,
$\Delta_k = a_{11}(a_{22}a_{33} - a_{32}a_{23}) + a_{13}(a_{21}a_{32} - a_{31}a_{22})$,
$B_k = \frac{b_1(a_{22}a_{33} - a_{32}a_{23}) + a_{13}(b_2a_{32} - b_3a_{22})}{\Delta_k}$,
$G_k = \frac{b_1 - a_{11}B_k}{a_{13}}$,
$C_k = \frac{b_2 - a_{21}B_k - a_{23}G_k}{a_{22}}$,
$A_k = \frac{1}{C_k + G_k}$,
$r_k = A_k\{q_2 + B_kq_1 + C_kr_{k-2} + F_kq_3 + G_kr_{k-3}\}$,
$x_k = A_k\{C_kx_{k-2} + G_kx_{k-3} - (q_1 + B_kr_{k-2} + F_kr_{k-3})\}$;
If $||r_k|| \leq \epsilon$, then $x = x_k$, **Stop**.
end for

5. THE NEW ALGORITHM A_{12} AND ITS DERIVATION

As said above, in [18], relation A_{12} is derived using the auxiliary polynomial $U_i(x) = x^i$, of exact degree i. Here, we discuss the same relation, but for $U_i(x) = P_i(x)$. Coefficients are estimated for the new case. The Lanczos-type algorithm based on A_{12} for the new choice of U_i, is called $A_{12}(new)$. This new algorithm is described below. Before deriving and discussing it, we recall the definition of an orthogonal polynomials sequence, [18].

Definition 1. A sequence P_m is called an orthogonal polynomial sequence, [16] with respect to the linear functional c if, for all nonnegative integers n and m,
(i) P_m is a polynomial of degree m,
(ii) $c(x^n P_m) = 0$, for $m \neq n$,
(iii) $c(x^m P_m) \neq 0$.

5.1. Relation A_{12} for the choice $U_i(x) = P_i(x)$.

Consider the following recurrence relationship, [18],

$$P_k(x) = A_k\{(x^2 + B_k x + C_k)P_{k-2}(x) + (D_k x^3 + E_k x^2 + F_k x + G_k)P_{k-3}(x)\} \quad (5.1)$$

where P_k, P_{k-2}, and P_{k-3} are polynomials of degree k, $k-2$, and $k-3$, respectively. A_k, B_k, C_k, D_k, E_k, F_k and G_k are the coefficients to be determined using the normality and the orthogonality conditions given in Section 3. Let, again, c be a linear functional defined by $c(x^i) = c_i$. The orthogonality condition gives

$$c(U_i P_k) = 0, i = 0, 1 \cdots, k-1.$$

For $x = 0$, and applying the normality condition, (5.1) becomes

$$1 = A_k\{C_k + G_k\}. \quad (5.2)$$

Now multiply (5.1) by U_i. Applying 'c' on both sides and using the orthogonality condition, we get

$$\begin{aligned} c(x^2 U_i P_{k-2}) + B_k c(x U_i P_{k-2}) + C_k c(U_i P_{k-2}) + D_k c(x^3 U_i P_{k-3}) \\ + E_k c(x^2 U_i P_{k-3}) + F_k c(x U_i P_{k-3}) + G_k c(U_i P_{k-3}) = 0. \end{aligned} \quad (5.3)$$

The orthogonality condition holds for $i = 0, 1, 2, \cdots, k-7$.
For $i = k-6$, equation (5.3) gives

$$D_k c(x^3 U_{k-6} P_{k-3}) = 0,$$

which implies that $D_k = 0$, since $c(x^3 U_{k-6} P_{k-3}) \neq 0$.
For $i = k-5$, (5.3) becomes $E_k c(x^2 U_{k-5} P_{k-3}) = 0$.
Since $c(x^2 U_{k-5} P_{k-3}) \neq 0$, $E_k = 0$.
For $i = k-4$, we get
$c(x^2 U_{k-4} P_{k-2}) + F_k c(x U_{k-4} P_{k-3}) = 0$, which gives

$$F_k = -\frac{c(x^2 U_{k-4} P_{k-2})}{c(x U_{k-4} P_{k-3})}. \quad (5.4)$$

For $i = k-3$, $i = k-2$ and $i = k-1$ equation (5.3) can be respectively written as,

$$\begin{aligned} B_k c(x U_{k-3} P_{k-2}) + G_k c(U_{k-3} P_{k-3}) = \\ -c(x^2 U_{k-3} P_{k-2}) - F_k c(x U_{k-3} P_{k-3}), \end{aligned} \quad (5.5)$$

$$\begin{aligned} B_k c(x U_{k-2} P_{k-2}) + C_k c(U_{k-2} P_{k-2}) + G_k c(U_{k-2} P_{k-3}) = \\ -c(x^2 U_{k-2} P_{k-2}) - F_k c(x U_{k-2} P_{k-3}), \end{aligned} \quad (5.6)$$

$$B_k c(xU_{k-1}P_{k-2}) + C_k c(U_{k-1}P_{k-2}) + G_k c(U_{k-1}P_{k-3}) = -c(x^2U_{k-1}P_{k-2}) - F_k c(xU_{k-1}P_{k-3}). \tag{5.7}$$

Now for simplicity let us denote the right sides of equations (5.5), (5.6) and (5.7) by b_1 ,b_2 and b_3 respectively then we get the following system of equations,

$$B_k c(xU_{k-3}P_{k-2}) + G_k c(U_{k-3}P_{k-3}) = b_1, \tag{5.8}$$
$$B_k c(xU_{k-2}P_{k-2}) + C_k c(U_{k-2}P_{k-2}) + G_k c(U_{k-2}P_{k-3}) = b_2, \tag{5.9}$$
$$B_k c(xU_{k-1}P_{k-2}) + C_k c(U_{k-1}P_{k-2}) + G_k c(U_{k-1}P_{k-3}) = b_3. \tag{5.10}$$

If Δ_k denotes the determinant of the coefficient matrix of the above system of equations then,

$$\begin{aligned}\Delta_k = c(xU_{k-3}P_{k-2})\{&c(U_{k-2}P_{k-2})c(U_{k-1}P_{k-3}) \\ &-c(U_{k-2}P_{k-3})c(U_{k-1}P_{k-2})\} \\ +c(U_{k-3}P_{k-3})\{&c(xU_{k-2}P_{k-2})c(U_{k-1}P_{k-2}) \\ &-c(U_{k-2}P_{k-2})c(xU_{k-1}P_{k-2})\}.\end{aligned} \tag{5.11}$$

If $\Delta_k \neq 0$ then,

$$B_k = \frac{1}{\Delta_k}\{b_1\{c(U_{k-2}P_{k-2})c(U_{k-1}P_{k-3}) - c(U_{k-2}P_{k-3})c(U_{k-1}P_{k-2})\} + c(U_{k-3}P_{k-3})\{b_2 c(U_{k-1}P_{k-2}) - b_3 c(U_{k-2}P_{k-2})\}\},$$

$$G_k = \frac{b_1 - c(xU_{k-3}P_{k-2})B_k}{c(U_{k-3}P_{k-3})},$$

$$C_k = \frac{b_2 - c(xU_{k-2}P_{k-2})B_k - c(U_{k-2}P_{k-3})G_k}{c(U_{k-2}P_{k-2})}.$$

With the above new estimated coefficients, the expression of polynomials $P_k(x)$ can be written as,

$$P_k(x) = A_k\{(x^2 + B_k x + C_k)P_{k-2}(x) + (F_k x + G_k)P_{k-3}(x)\}. \tag{5.12}$$

Now, for $U_i(x) = P_k(x)$, and from equation (5.4), F_k becomes

$$F_k = -\frac{c(x^2P_{k-4}P_{k-2})}{c(xP_{k-4}P_{k-3})}.$$

Similarly, from equation (5.11) Δ_k becomes,
$\Delta_k = c(xP_{k-3}P_{k-2})\{c(P_{k-2}^2)c(P_{k-1}P_{k-3}) - c(P_{k-2}P_{k-3})c(P_{k-1}P_{k-2})\} + c(P_{k-3}^2)\{c(xP_{k-2}^2)c(P_{k-1}P_{k-2}) - c(P_{k-2}^2)c(xP_{k-1}P_{k-2})\}$.

Using (definition 1), [18], Δ_k simplifies to

$$\Delta_k = -c(P_{k-3}^2)c(P_{k-2}^2)c(xP_{k-1}P_{k-2}).$$

Using again definition 1 and $U_i(x) = P_k(x)$, the rest of the coefficients can be determined as follows. Let

$$\begin{aligned}b_1 &= -c(x^2P_{k-3}P_{k-2}) - F_k c(xP_{k-3}^2), \\ b_2 &= -c(x^2P_{k-2}^2) - F_k c(xP_{k-2}P_{k-3}), \\ b_3 &= -c(x^2P_{k-1}P_{k-2}) - F_k c(xP_{k-1}P_{k-3}),\end{aligned}$$

then

$$B_k = \frac{1}{\Delta_k}\{b_1\{c(P_{k-2}P_{k-2})c(P_{k-1}P_{k-3}) - c(P_{k-2}P_{k-3})c(P_{k-1}P_{k-2})\} \\ + c(P_{k-3}P_{k-3})\{b_2 c(P_{k-1}P_{k-2}) - b_3 c(P_{k-2}P_{k-2})\}\},$$

or,

$$B_k = -\frac{b_3 c(P_{k-3}^2)c(P_{k-2}^2)}{\Delta_k} = \frac{b_3}{c(xP_{k-1}P_{k-2})},$$

$$G_k = \frac{b_1 - c(xP_{k-3}P_{k-2})B_k}{c(P_{k-3}^2)},$$

$$C_k = \frac{b_2 - c(xP_{k-2}P_{k-2})B_k - c(P_{k-2}P_{k-3})G_k}{c(P_{k-2}P_{k-2})} = \frac{b_2 - c(xP_{k-2}^2)B_k}{c(P_{k-2}^2)},$$

and

$$A_k = \frac{1}{C_k + G_k}.$$

As in [18], we can write,

$$\mathbf{r}_k = A_k\{A^2\mathbf{r}_{k-2} + B_k A\mathbf{r}_{k-2} + C_k\mathbf{r}_{k-2} + F_k A\mathbf{r}_{k-3} + G_k\mathbf{r}_{k-3}\}, \tag{5.13}$$

$$\mathbf{x}_k = A_k\{C_k\mathbf{x}_{k-2} + G_k\mathbf{x}_{k-3} - (A\mathbf{r}_{k-2} + B_k\mathbf{r}_{k-2} + F_k\mathbf{r}_{k-3})\}. \tag{5.14}$$

As we know from [2, 3],

$$\begin{cases} \text{setting } U_k(x) = P_k(x) \text{ and } \mathbf{z}_k = P_k(A^T)\mathbf{y}, \text{ we get} \\ c(U_kP_k) = (y, U_k(A)P_(A)\mathbf{r}_0) = (U_k(A^T)\mathbf{y}, P_k(A)\mathbf{r}_0) = (\mathbf{z}_k, \mathbf{r}_k). \end{cases} \tag{5.15}$$

So, from relation (5.12), after replacing **x** by A^T, multiplying by **y** on both sides and using (5.15) we can write,

$$\mathbf{z}_k = A_k\{(A^T)^2\mathbf{z}_{k-2} + B_k A^T\mathbf{z}_{k-2} + C_k\mathbf{z}_{k-2} + F_k A^T\mathbf{z}_{k-3} + G_k\mathbf{z}_{k-3}\}. \tag{5.16}$$

Similarly using (5.15) all coefficients become,

$F_k = -\frac{c(x^2P_{k-4}P_{k-2})}{c(xP_{k-4}P_{k-3})} = -\frac{(A^T\mathbf{z}_{k-2}, A\mathbf{r}_{k-4})}{(\mathbf{z}_{k-3}, A\mathbf{r}_{k-4})}$,

$\Delta_k = -c(P_{k-3}^2)c(P_{k-2}^2)c(xP_{k-1}P_{k-2}) = -(\mathbf{z}_{k-3}, \mathbf{r}_{k-3})(\mathbf{z}_{k-2}, \mathbf{r}_{k-2})(\mathbf{z}_{k-1}, A\mathbf{r}_{k-2})$.

$b_1 = -(A^T\mathbf{z}_{k-3}, A\mathbf{r}_{k-2}) - F_k(\mathbf{z}_{k-3}, A\mathbf{r}_{k-3})$,

$b_2 = -(A^T\mathbf{z}_{k-2}, A\mathbf{r}_{k-2}) - F_k(\mathbf{z}_{k-2}, A\mathbf{r}_{k-3})$,

$b_3 = -(A^T\mathbf{z}_{k-1}, A\mathbf{r}_{k-2}) - F_k(\mathbf{z}_{k-1}, A\mathbf{r}_{k-3})$,

$B_k = \frac{b_3}{c(xP_{k-1}P_{k-2})} = \frac{b_3}{(\mathbf{z}_{k-1}, A\mathbf{r}_{k-2})}$,

$G_k = \frac{b_1 - c(xP_{k-3}P_{k-2})B_k}{c(P_{k-3}^2)} = \frac{b_1 - (\mathbf{z}_{k-3}, A\mathbf{r}_{k-2})B_k}{(\mathbf{z}_{k-3}, \mathbf{r}_{k-3})}$,

$C_k = \frac{b_2 - c(xP_{k-2}^2)B_k}{c(P_{k-2}^2)} = \frac{b_2 - (\mathbf{z}_{k-2}, A\mathbf{r}_{k-2})B_k}{(\mathbf{z}_{k-2}, \mathbf{r}_{k-2})}$,

$A_k = \frac{1}{C_k + G_k}$.

All previous formulae are valid for $k \geq 4$. So we need $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$ and $\mathbf{z}_1$, $\mathbf{z}_2$, $\mathbf{z}_3$ to calculate $\mathbf{r}_k$ and $\mathbf{z}_k$ recursively. $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{z}_1$, $\mathbf{z}_2$ are found differently in [18], while $\mathbf{r}_3$ and $\mathbf{z}_3$ can be determined in a similar way giving,

$\mathbf{r}_3 = \mathbf{r}_0 - \frac{\acute{\alpha}}{\Delta}\mathbf{p} + \frac{\acute{\beta}}{\Delta}\mathbf{p}_1 - \frac{\acute{\gamma}}{\Delta}\mathbf{p}_2$,

$\mathbf{z}_3 = \mathbf{z}_0 - \frac{\acute{\alpha}}{\Delta}\mathbf{y}_1 + \frac{\acute{\beta}}{\Delta}\mathbf{y}_2 - \frac{\acute{\gamma}}{\Delta}\mathbf{y}_3$.
Using $\mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k$, we get from $\mathbf{r}_3$,
$\mathbf{x}_3 = \mathbf{x}_0 + \frac{\acute{\alpha}}{\Delta}\mathbf{r}_0 - \frac{\acute{\beta}}{\Delta}\mathbf{p} + \frac{\acute{\gamma}}{\Delta}\mathbf{p}_1$, where $\Delta = c_1(c_3c_5 - c_4^2) - c_2(c_2c_5 - c_3c_4) + c_3(c_2c_4 - c_3^2)$,
$\acute{\alpha} = c_0(c_3c_5 - c_4^2) - c_2(c_1c_5 - c_2c_4) + c_3(c_1c_4 - c_3c_2)$,
$\acute{\beta} = c_0(c_2c_5 - c_4c_3) - c_1(c_1c_5 - c_2c_4) + c_3(c_1c_3 - c_2^2)$,
$\acute{\gamma} = c_0(c_2c_4 - c_3^2) - c_1(c_1c_4 - c_2c_3) + c_2(c_1c_3 - c_2^2)$.
Note that parameters $\mathbf{p}, \mathbf{p}_1, \mathbf{p}_2, \mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \Delta, \alpha, \beta$, and γ are temporary and defined in the algorithm below.

5.2. **Algorithm $A_{12}(new)$.** We can now describe the new variant of algorithm $A_{12}(new)$ as follows.

Algorithm 2 : Lanczos-type Algorithm $A_{12}(new)$.

Choose $\mathbf{x}_0$ and $\mathbf{y}$ such that $\mathbf{y} \neq 0$.
Set $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$, $\mathbf{z}_0 = \mathbf{y}$, $\mathbf{p} = A\mathbf{r}_0$, $\mathbf{p}_1 = A\mathbf{p}$, $\mathbf{p}_2 = A\mathbf{p}_1$, $\mathbf{p}_3 = A\mathbf{p}_2$, $\mathbf{p}_4 = A\mathbf{p}_3$,
$c_0 = (\mathbf{y}, \mathbf{r}_0)$, $c_1 = (\mathbf{y}, \mathbf{p})$, $c_2 = (\mathbf{y}, \mathbf{p}_1)$, $c_3 = (\mathbf{y}, \mathbf{p}_2)$, $c_4 = (\mathbf{y}, \mathbf{p}_3)$, $c_5 = (\mathbf{y}, \mathbf{p}_4)$,
$\delta = c_1c_3 - c_2^2$, $\alpha = \frac{c_0c_3 - c_1c_2}{\delta}$, $\beta = \frac{c_0c_2 - c_1^2}{\delta}$,
$\mathbf{r}_1 = \mathbf{r}_0 - (\frac{c_0}{c_1})\mathbf{p}$, $\mathbf{x}_1 = \mathbf{x}_0 + (\frac{c_0}{c_1})\mathbf{r}_0$,
$\mathbf{r}_2 = \mathbf{r}_0 - \alpha\mathbf{p} + \beta\mathbf{p}_1$, $\mathbf{x}_2 = \mathbf{x}_0 + \alpha\mathbf{r}_0 - \beta\mathbf{p}$,
$\mathbf{y}_1 = A^T\mathbf{y}$, $\mathbf{y}_2 = A^T\mathbf{y}_1$, $\mathbf{y}_3 = A^T\mathbf{y}_2$,
$\mathbf{z}_1 = \mathbf{z}_0 - (\frac{c_0}{c_1})\mathbf{y}_1$, $\mathbf{z}_2 = \mathbf{z}_0 - \alpha\mathbf{y}_1 + \beta\mathbf{y}_2$,
$\Delta = c_1(c_3c_5 - c_4^2) - c_2(c_2c_5 - c_3c_4) + c_3(c_2c_4 - c_3^2)$,
$\acute{\alpha} = c_0(c_3c_5 - c_4^2) - c_2(c_1c_5 - c_2c_4) + c_3(c_1c_4 - c_3c_2)$,
$\acute{\beta} = c_0(c_2c_5 - c_4c_3) - c_1(c_1c_5 - c_2c_4) + c_3(c_1c_3 - c_2^2)$,
$\acute{\gamma} = c_0(c_2c_4 - c_3^2) - c_1(c_1c_4 - c_2c_3) + c_2(c_1c_3 - c_2^2)$,
$\mathbf{r}_3 = \mathbf{r}_0 - \frac{\acute{\alpha}}{\Delta}\mathbf{p} + \frac{\acute{\beta}}{\Delta}\mathbf{p}_1 - \frac{\acute{\gamma}}{\Delta}\mathbf{p}_2$,
$\mathbf{z}_3 = \mathbf{z}_0 - \frac{\acute{\alpha}}{\Delta}\mathbf{y}_1 + \frac{\acute{\beta}}{\Delta}\mathbf{y}_2 - \frac{\acute{\gamma}}{\Delta}\mathbf{y}_3$,
$\mathbf{x}_3 = \mathbf{x}_0 + \frac{\acute{\alpha}}{\Delta}\mathbf{r}_0 - \frac{\acute{\beta}}{\Delta}\mathbf{p} + \frac{\acute{\gamma}}{\Delta}\mathbf{p}_1$.
for k = 4,5…, **do**
  $q_1 = A\mathbf{r}_{k-2}$, $q_2 = A\mathbf{q}_1$, $q_3 = A\mathbf{r}_{k-3}$,
  $\mathbf{s}_1 = A^T\mathbf{z}_{k-2}$, $\mathbf{s}_2 = A^T\mathbf{s}_1$, $\mathbf{s}_3 = A^T\mathbf{z}_{k-3}$,
  $\Delta_k = -(\mathbf{z}_{k-3}, \mathbf{r}_{k-3})(\mathbf{z}_{k-2}, \mathbf{r}_{k-2})(\mathbf{z}_{k-1}, A\mathbf{r}_{k-2})$,
  $F_k = -\frac{(A^T\mathbf{z}_{k-2}, A\mathbf{r}_{k-4})}{(\mathbf{z}_{k-3}, A\mathbf{r}_{k-4})}$,
  $b_1 = -(A^T\mathbf{z}_{k-3}, A\mathbf{r}_{k-2}) - F_k(\mathbf{z}_{k-3}, A\mathbf{r}_{k-3})$,
  $b_2 = -(A^T\mathbf{z}_{k-2}, A\mathbf{r}_{k-2}) - F_k(\mathbf{z}_{k-2}, A\mathbf{r}_{k-3})$,
  $b_3 = -(A^T\mathbf{z}_{k-1}, A\mathbf{r}_{k-2}) - F_k(\mathbf{z}_{k-1}, A\mathbf{r}_{k-3})$,
  $B_k = \frac{b_3}{(\mathbf{z}_{k-1}, A\mathbf{r}_{k-2})}$,
  $G_k = \frac{b_1 - (\mathbf{z}_{k-3}, A\mathbf{r}_{k-2})B_k}{(\mathbf{z}_{k-3}, \mathbf{r}_{k-3})}$,
  $C_k = \frac{b_2 - (\mathbf{z}_{k-2}, A\mathbf{r}_{k-2})B_k}{(\mathbf{z}_{k-2}, \mathbf{r}_{k-2})}$,
  $A_k = \frac{1}{C_k + G_k}$,
  $\mathbf{r}_k = A_k\{\mathbf{q}_2 + B_k\mathbf{q}_1 + C_k\mathbf{r}_{k-2} + F_k\mathbf{q}_3 + G_k\mathbf{r}_{k-3}\}$,
  $x_k = A_k\{C_k\mathbf{x}_{k-2} + G_k\mathbf{x}_{k-3} - (\mathbf{q}_1 + B_k\mathbf{r}_{k-2} + F_k\mathbf{r}_{k-3})\}$,
  $\mathbf{z}_k = A_k\{\mathbf{s}_2 + B_k\mathbf{s}_1 + C_k\mathbf{z}_{k-2} + F_k\mathbf{s}_3 + G_k\mathbf{z}_{k-3}\}$.
  If $||r_k|| \leq \epsilon$, then $x = x_k$, **Stop**.
end for

6. Numerical Tests

$A_{12}(new)$ has been tested against A_{12}, A_5/B_{10} and A_8/B_{10}, the best Lanczos-type algorithms according to [2, 18, 3]. The test problems arise in the 5-point discretisation of the operator $\frac{-\partial^2}{\partial x^2} - \frac{\partial^2}{\partial y^2} + \gamma \frac{\partial}{\partial x}$ on a rectangular region [3]. Comparative results on instances of the following problem ranging from dimension 10 to 100 for parameter δ taking values 0.0 and for the tolerance $eps = 1.0e-05$, are recorded in Table 1.

$$A = \begin{pmatrix} B & -I & \cdots & \cdots & 0 \\ -I & B & -I & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & -I & B & -I \\ 0 & \cdots & \cdots & -I & B \end{pmatrix},$$

with

$$B = \begin{pmatrix} 4 & \alpha & \cdots & \cdots & 0 \\ \beta & 4 & \alpha & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \beta & 4 & \alpha \\ 0 & \cdots & & \beta & 4 \end{pmatrix},$$

and $\alpha = -1 + \delta$, $\beta = -1 - \delta$. The right hand side **b** is taken to be $\mathbf{b} = A\mathbf{X}$, where $\mathbf{X} = (1, 1, \ldots, 1)^T$, is the solution of the system. The dimension of *B* is 10.

TABLE 1. Experimental results for problems when $\delta = 0$

	A_5/B_{10}		A_8/B_{10}		A_{12}		$A_{12}new$	
n	$\|\|r_k\|\|$	t(sec)	$\|\|r_k\|\|$	t(sec)	$\|\|r_k\|\|$	t(sec)	$\|\|r_k\|\|$	t(sec)
10	$2.0866e-013$	0.002628	$3.5775e-013$	0.008440	$1.0252e-013$	0.042433	$2.7146e-015$	0.018560
20	$2.5278e-014$	0.002619	$1.6765e-013$	0.008624	$1.8456e-013$	0.042880	$2.4416e-015$	0.017902
30	$2.4011e-009$	0.003139	$6.9352e-009$	0.009134	$1.6272e-007$	0.043438	$2.0829e-010$	0.019099
40	$1.5539e-009$	0.003344	$1.5680e-009$	0.009113	$2.0343e-010$	0.043924	$2.7946e-011$	0.019164
50	$1.8730e-006$	0.003810	$1.4671e-006$	0.009634	$4.7570e-005$	0.044461	$1.2734e-006$	0.020314
60	$5.9083e-006$	0.003747	$6.6800e-006$	0.009599	$2.8615e-005$	0.044002	$2.3608e-006$	0.020202
70	$9.3260e-006$	0.004658	$4.6961e-006$	0.010246	$8.5638e-005$	0.044369	$5.3790e-007$	0.020988
80	$4.5674e-006$	0.005496	$4.6144e-006$	0.011470	$6.8618e-005$	0.046109	$3.5468e-006$	0.022625
90	NaN		NaN		$7.2121e-005$	0.047276	$4.3695e-006$	0.021556
100	$9.0038e-006$	0.004284	$8.4881e-007$	0.010383	$3.1098e-005$	0.044758	$2.0040e-008$	0.020606

Table 1 records the computational results obtained with algorithms A_{12}(new), A_{12}, A_5/B_{10} and A_8/B_{10}. Clearly, A_{12}(new) is an improvement on A_{12} on both robustness/stability and efficiency accounts. Compared to the well established A_5/B_{10} and A_8/B_{10}, it is definitely more robust/stable; indeed, all problems have been solved to the required accuracy by A_{12}(new), and the other two algorithms failed to do so in one case as evidenced by the "NaN" outputs which point to breakdown or lack of robustness and stability, on the problem of dimension n=90. On efficiency, however, as expected, algorithms A_5/B_{10} and A_8/B_{10} are faster since they rely on recurrence relations involving lower order FOP's requiring few coefficients to estimate; unlike A_{12} and A_{12}(new).

7. CONCLUSION

In this paper we have shown that, if the recurrence relation A_{12} [18], is determined for the choice of $U_i(x) = P_i(x)$, other than x^i which is discussed in [18], then a more robust algorithm $A_{12}(new)$ can be derived. The numerical performance of this algorithm compares well to that of three existing Lanczos-type algorithms, which were found to be the best among a number of such algorithms, [2, 18, 3], on the same set of problems as considered here. Another achievement of $A_{12}(new)$ is that it can solve the above problem when its dimension is up to 500, while the rest of algorithms give results for problems with dimensions less or equal to 100.

REFERENCES

[1] C. G. Broyden and M. T. Vespucci. *Krylov Solvers For Linear Algebraic Systems*, Elsevier, Amsterdam, The Netherlands, 2004.

[2] C. Baheux, *Algorithmes d'implementation de la méthode de Lanczos*, PhD thesis, University of Lille 1, France, 1994.

[3] C. Baheux, New Implementations of Lanczos Method. *Journal of Computational and Applied Mathematics*, **57**, (1995), 3–15.

[4] A. Bjôrck, T. Elfving, and Z. Strakos, Stability of Conjugate Gradient and Lanczos Methods for Linear Least Squares Problems, *SIAM Journal of Matrix Analysis and Application*, **19**, (1998), 720–736.

[5] C. Brezinski, *Padé-Type Approximation and General Orthogonal Polynomials*, Internat. Ser. Nuner. Math. 50. Birkhäuser, Basel, 1980.

[6] C. Brezinski and H. Sadok, Lanczos-type algorithms for solving systems of linear equations, *Applied Numerical Mathematics*, **11**, (1993), 443–473.

[7] C. Brezinski and M. R. Zaglia, Hybird procedures for solving linear systems, *Numerische Mathematik*, **67**, 1994, 1–19.

[8] C. Brezinski, M. R. Zaglia, and H. Sadok. New look-ahead Lanczos-type algorithms for linear systems, *Numerische Mathematik*, **83**, (1999), 53–85.

[9] C. Brezinski, M. R. Zaglia, and H. Sadok. The matrix and polynomial approaches to Lanczos-type algorithms, *Journal of Computational and Applied Mathematics*, **123**, (2000), 241–260.

[10] C. Brezinski, M. R. Zaglia, and H. Sadok. A review of formal orthogonality in Lanczos-based methods. *Journal of Computational and Applied Mathematics*, **140**, (2002), 81–98.

[11] C. Brezinski and M. R. Zaglia, A new presentation of orthogonal polynomials with applications to their computation, *Numerical Algorithms*, **1**, (1991), 207–222.

[12] C. Brezinski, M. R. Zaglia, and H. Sadok. A Breakdown-free Lanczos type algorithm for solving linear systems, *Numerische Mathematik*, **63**, (1992), 29–38.

[13] C. Brezinski, M. R. Zaglia, and H. Sadok. Avoiding breakdown and nearbreakdown in Lanczos type algorithms, *Numerical Algorithms*, **1**, (1991), 261–284.

[14] Q. Ye, A Breakdown-Free Variation of the Nonsymmetric Lanczos Algorithms, *Mathematics of Computation*, **62**, (1994), 179–207.

[15] D. Calvetti, L. Reichel, F. Sgallari, and G. Spaletta, A Regularizing Lanczos iteration method for underdetermined linear systems, *Jouranl of Computational and Applied Mathematics*, **115**, (2000) 101–120.

[16] T. S. Chihara, *An Introduction to Orthogonal Polynomials*, Gordon and Breach, New York, London, Paris, 1984.

[17] A. Draux, *Polynômes Orthogonaux Formels*, Application, LNM 974. Springer-Verlag, Berlin, 1983.

[18] M. Farooq. *New Lanczos-type Algorithms and their Implementation*, PhD thesis, University of Essex, UK, 2011.

[19] M. Farooq and A. Salhi. New Recurrence Relationships Between Orthogonal Polynomials Which Lead to New Lanczos-type Algorithms, *Journal of Prime Research in Mathematics*, **8**, (2012), 61-75.

[20] M. Farooq and A. Salhi. A Switching Approach to Avoid Breakdown in Lanczos-type Algorithms, *to appear in Applied Mathematics and Information Sciences in 2014.*

[21] M. Farooq and A. Salhi. A Pre-emptive Restarting Approach to Beating the Inherent Instability of Lanczos-type Algorithms, *to appear in Iranian Journal of Science and Technology, Transaction-A, Science, in 2013.*

[22] R. Fletcher, Conjugate gradient methods for indefinite systems. in: G.A. Watson, (Ed), *Numerical Analysis, Dundee 1975*, Lecture Notes in Mathematics,, volume 506. Springer, Berlin, 1976.

[23] A. Greenbaum. *Iterative Methods for Solving Linear System*, Societ for Industrial and Applied Mathematics, Philadelphia, 1997.

[24] A. El Guennouni. A unified approach to some strategies for the treatment of breakdown in Lanczos-type algorithms. *Applicationes Mathematicae*, **26**, (1999), 477–488.

[25] M.R. Hestenes and E. Stiefel. Mehtods of conjugate gradients for solving linear systems. *Journal of the National Bureau of Standards*, **49**, (1952), 409–436.

[26] C. Lanczos. An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integeral Operators. *Journal of Research of the National Bureau of Standards*, **45**, (1950), 255–282.

[27] C. Lanczos. Solution of systems of linear equations by minimized iteration. *Journal of the National Bureau of Standards*, **49**, (1952), 33–53.

[28] G. Meurant. *The Lanczos and conjugate gradient algorithms, From Theory to Finite Precision Computations*. SIAM, Philadelphia, 2006.

[29] B. N. Parlett and D. S. Scott. The Lanczos Algorithm With Selective Orthogonaliztion. *Mathematics of Computation*, **33**, (1979), 217–238.

[30] B. N. Parlett, D. R. Taylor, and Z. A. Liu. A Look-Ahead Lanczos Algorithm for Unsymmetric Matrices. *Mathematics of Computation*, **44**, (1985), 105–124.

[31] Y. Saad. On the Lanczos method for solving linear system with several right-hand sides. *Mathematics of Computation*, **48**, (1987), 651–662.

[32] G. Szegö. Orthogonal Polynomials, *American Mathematical Society*, Providence, Rhode Island, 1939.

[33] H. A. Van der Vorst, An iterative solution method for solving $f(A)\mathbf{x}=\mathbf{b}$, using Krylov subspace information obtained for the symmetric positive definite matrix A, *Journal of Computational and Applied Mathematics*, **18(2)**, (1987), 249–263.