International Journal of Mathematics and
Computer Science, **16**(2021), no. 2, 837–854

High Performances of Stabilized Lanczos-Types for Solving High Dimension Problems: A Survey

**Maharani Abu Bakar, Alifhar Juliansyah,
Rehana Thalib, Danang Adi Pratama**

Faculty of Ocean Engineering Technology and Informatics
Universiti Malaysia Terengganu
Kuala Terengganu, Malaysia

email: maharani@umt.edu.my

(Received October 11, 2020, Revised December 26, 2020,
Accepted December 29, 2020)

Abstract

Lanczos-type algorithms are iterative methods for solving symmetric and unsymmetric systems of linear equations (SLEs) which particularly involve large numbers of variables and equations. One big issue with these algorithms is known as breakdown; it causes the algorithms to fail before obtaining (converging to) a good solution. This obviously reduces the performances of the algorithms. Several strategies have been suggested and investigated. Some recent ones include restarting the algorithms using quality points, switching between variants of these algorithms and embedding interpolation and extrapolation processes within these Lanzcos-type algorithms. The Embedding of Interpolation and Extrapolation Methods within Lanczos-type Algorithms or EIEMLA approach and its modification (or MEIEMLA), as well as the hybrid restarting-MEIEMLA, are some of the latest strategies introduced to combat breakdown and improve the convergence rates of Lanczos-type algorithms. All of those methods are reviewed in this paper and put in context. We highlight their advantages and disadvantages to direct the interested reader to potential further investigations. Besides, we introduce the use of support vector machine

Key words and phrases: Lanczos-type algorithms, high dimensions of SLEs, restarting, switching, interpolation and extrapolation model.
AMS (MOS) Subject Classifications: 65F10, 65Y10.
ISSN 1814-0432, 2021, http://ijmcs.future-in-tech.net

(SVR) in similarly purposes as EIEMLA and MEIEMLA. The comparison of the methods is on efficiency and robustness. Numerical results are also included.

1 Introduction

Lanczos-type methods [1, 2] for solving symmetric and non-symmetric systems, are projection methods on Krylov subspaces, [3, 4], which are mainly built through the theory of formal orthogonal polynomials (FOPs) [5, 6]. The theory allows us to create any combinations of the three-term-recurrence relationships of a family of orthogonal polynomials, P_k, and their adjacent ${P_k}^{(1)}$. The way to compute the coefficients of the recurrence formula by using the theory of FOPs leads to form various Lanczos-type algorithms [7].

Several types of Lanczos-based algorithms have been investigated in [8] and in [9, 10]. We call them Baheux-types and Farooq-types, respectively, refering to their founders. Both types differ by the degree of orthogonal polynomials used in the recurrence formula, where the latter ones used the higher degree of polynomials. In this case, the difference in the degrees of polynomials of Baheux-types is at least two, whereas in those of Farooq-types the difference in degrees is three. Consequently, more computation of coefficients need to be done to form Farooq-types algorithms. However, Farooq-types are more robust than Baheux-types for solving high dimensions of SLEs as was claimed in [9].

Apart from their effectiveness to solve high dimensions of SLEs, there is one big issue which always accompanies the algorithms every time they solve the SLEs. It is known as *breakdown* and causes a failure to terminate at a good solution [13, 14]. The breakdown phenomena and strategies to deal with it have been interesting topics of discussions in recent decades. Let's list them from the pioneer strategy to the recent ones.

Through the method of recursive zoom (MRZ), Brezinski [5], [16], proposed a strategy to jump over such breakdown by jumping over the non-existing orthogonal polynomials. This is due to the absence of orthogonal polynomials in the recurrence formula which causes a breakdown. Therefore, this method is also known as a look-ahead strategy [17]. Similar works, called look-around strategy, have been investigated by selecting well-conditioned pairs of neighboring polynomials which are associated with the Pade table [19]. More recently, interesting techniques to cure the breakdown have been investigated by Farooq and Salhi [11, 12] involving restarting and switching strategies. Basically, in a restarting strategy, when a Lanczos algorithm runs for a fixed number of iterations, or before it faced breakdown, we pick up one iterate to be used as the initialization point for the next run and using the same algorithm. This is different from switching where for the next run we use other types of Lanczos algorithms.

The most recent investigation of a technique to cure the breakdown is the interpolation and extrapolation model embedded in Lanczos-type algorithms. The so-called embedding interpolation and extrapolation model within Lanczos-type algorithms, or EIEMLA, exploits patterns which persistently exist in every iteration [20, 29]. Note that these patterns consist of some good approximate solutions generated by a Lanczos-type algorithm. Thus, the interpolant PCHIP is used to capture the pattern and realize it as a model function. The modification of EIEMLA has also been considered in [23] to interpolate the good iterates more efficiently.

This survey sketches strategies such as restarting, switching, EIEMLA, and MEIEMLA, to briefly present their comparisons of robustness and efficiency. Moreover, we discuss their limitations in order to give the reader a broader picture.

In section 2, a developed restarting method is given which considers a quality point as a restarting point. This strategy led to algorithms RLMinRes and RLMedVal, [20] which, numerically, have proven better than the traditional restarting by Farooq [11]. In section 3 we present Farooq's switching strategy verssu the new switching [23]. In section 4, we discuss, in more details, the derivation of EIEMLA and MEIEMLA including their comparisons. In section 5, we present a brief analysis of the four methods. Finally, in section 6, we conclude by presenting some possibilities for further work.

2 A Review : Lanczos-type Algorithms

Lanczos-type algorithms for solving the non-symmetric system

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

where $A \in R^{nxn}$, $\mathbf{x}, \mathbf{b} \in R^n$, combine the projection method and Krylov subspace method. For the earlier one, the updated iterate $\mathbf{x}_k/$ should satisfy

1. $$\mathbf{x}_k - \mathbf{x}_0 \in K_k(A, \mathbf{r}_0), \tag{2.2}$$

   where $\mathbf{x}_0$ is the initial vector solution, K_k is the Krylov subspace, and $\mathbf{r}_0 = \mathbf{b} - A\mathbf{x}_0$ is the initial residual vector.

2.
$$\mathbf{r}_k \perp K_k(A^T, \mathbf{y}), \tag{2.3}$$
for any non-zero vector $\mathbf{y}$.

As in (2.2), the Krylov subspace method plays an important role in generating the approximate solutions $\mathbf{x}_k$ which are linearly combined with the basis of $A^i\mathbf{r}_0$, for $i = 0, 1, \ldots, k$,

$$\mathbf{x}_k = \mathbf{x}_0 + \alpha_0\mathbf{r}_0 + \alpha_1 A\mathbf{r}_0 + \ldots + \alpha_{k-1}A^k\mathbf{r}_0. \tag{2.4}$$

$$\mathbf{r}_k = \mathbf{x}_0 + \mathbf{r}_0 + \alpha_1 A\mathbf{r}_0 + \ldots + \alpha_k A^k\mathbf{r}_0 = P_k(A)\mathbf{r}_0, \tag{2.5}$$

where $P_k(A)$ is an orthogonal polynomial. As in (2.3), the orthogonality of the residual vectors $\mathbf{r}_k$ with respect to the Krylov subspace leads to the relation

$$\langle \mathbf{r}_k, A^{T^i}\mathbf{y}\rangle = 0, \quad \text{for } i = 0, 1, \ldots, k. \tag{2.6}$$

By the induced property of an Euclidean norm, equation (2.6) becomes

$$\langle \mathbf{y}, A^i P_k(A)\mathbf{r}_0\rangle = 0, \quad \text{for } i = 0, 1, \ldots, k. \tag{2.7}$$

Set

$$c_i = \langle \mathbf{y}, A^i\mathbf{r}_0\rangle, \tag{2.8}$$

for $i = 0, 1, \ldots, k-1$, where c is a linear function defined by $c_i = c(t^i)$ [5]. Multiply (2.8) with $P_k(t)$ and substitute it into (2.8) we get

$$c(t^i P_k(t)) = \langle \mathbf{y}, A^i P_k(A)\mathbf{r}_0\rangle = 0, \quad \text{for } i = 0, 1, \ldots, k-1. \tag{2.9}$$

Thus a Lanczos-type algorithm is derived by constructing the orthogonal polynomials P_k, computing $\mathbf{r}_k = P_k(A)\mathbf{r}_0$ recursively and updating the approximate iterate $\mathbf{x}_k$ from formula $\mathbf{b} - A\mathbf{r}_k$.

There are at least twenty formulas in the form of A_i/B_j of Lanczos basis which have been investigated by Baheux, read Baheux-types, [8]. Some of them have been implemented by algorithms such as A_4 (Orthores), A_5/B_5, A_6/B_8 (Orthodir), etc. There are other formulas, named A_{12}, A_{13}/B_{13}, A_{19}/B_6, etc, found by Farooq, read Farooq-types, [9, 10]. Farooq-types have been developed by increasing the degree of orthogonal polynomials which results in more computation of coefficients, however they are more robust. It is still open to investigate other types of Lanczos algorithms.

Apart from the flexibility of forming the orthogonal polynomials however, a Lanczos algorithm cannot converge accurately when solving high dimensions of SLEs, due to a breakdown [15] which is a latent drawback occurring due to division by zero or cumulative computational errors. The next sections discusses some recent strategies to deal with the breakdown.

3 The New Restarting and Switching

Popular strategies to treat the breakdown are restarting and switching which have been developed by Farooq and Salhi [11, 12]. In this section, this new version of restarting and switching is discussed.

Consider Lanczos Orthodir, [17], to solve system $A\mathbf{x} = \mathbf{b}$. Let's assume that breakdown occurs after k iterations. Then we stop Orthodir at the k^{th} iteration. Technically, restarting means that we pick up an iterate of the k iterates and use it as the initialization guess for the next around (cycle). In this case, we can choose either the last iterate before breakdown occurs, or other quality iterates. Traditional restarting usually picks the k^{th} iterate for restarting point. The developed restarting, such as the one discussed in [20], selects two quality iterates which are the iterates with the minimum residual norm and with median values, to re-cycle the Lanczos algorithm. These lead to algorithms RLMinRes and RLMedVal which have been implemented and successfully beat the traditional restarting. We present RLMinRes algorithm here as it is the best device compared with RLMedVal and traditional restarting. In fact, we compare it with other stabilized Lanczos algorithms in this paper.

Algorithm 1 The RLMinRes algorithm, [20]

1: Run LMinRes algorithm for k iterations and obtain sol_{min} and $norm_{min}$.
2: **while** $norm_{min} \geq \epsilon$ **do**
3: Initialize the algorithm with

$$\mathbf{x} = sol_{min},$$
$$\mathbf{y} = \mathbf{b} - A\mathbf{x}.$$

4: Run LMinRes algorithm for k iterations.
5: **end while**
6: Take sol_{min} as the approximate solution.
7: Stop.

Different from restarting, when breakdown occur in a Lanczos algorithm, switching allows us to choose some Lanczos-type algorithms, rather than one algorithm, to go for the next around. New switching adopts the proposed restarting in the way of using the quality points [22]. However, only the iterate with the lowest residual norm is considered in the paper. In addition, the new switching applied the unlimited number of iterations rather than a fixed number which allows us to select more quality iterates. This leads to design SLUMinRes algorithm as shown in [22].

4 EIEMLA vs MEIEMLA on a Restarting Framework

The methods of EIEMLA, [29], and MEIEMLA, [23], can be called new variants of Lanczos-types and are based on the interpolation and extrapolation process. The interpolation process involves capturing some patterns persistently during the Lanczos iteration. The extrapolation process is then used to predict some iterates which are out of the iteration process. EIEMLA and MEIEMLA differ by the interpolation process.

4.1 EIEMLA vs MEIEMLA

Following [21], again consider a Lanczos algorithm to solve SLE by using k iterations, where $k \leq n$. We assume the algorithm breakdown after k iterations. Let $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \ldots, \mathbf{x}_k\}$ be the sequence of all approximate solutions including the one with the lowest residual norm $\mathbf{x}_m$. Our aim here is to find other elements that replace $\mathbf{x}_{k+1}, \mathbf{x}_{k+2}, \ldots$ if the breakdown didn't occur. To do that, we define a sequence $\{v_1, v_2, \ldots, v_n\}$ as follows

$$\begin{aligned} v_1 &= \left\{x_1{}^{(1)}, x_2{}^{(1)}, \ldots, x_k{}^{(1)}\right\} \\ v_2 &= \left\{x_1{}^{(2)}, x_2{}^{(2)}, \ldots, x_k{}^{(2)}\right\} \\ &\vdots \\ v_n &= \left\{x_1{}^{(n)}, x_2{}^{(n)}, \ldots, x_k{}^{(n)}\right\} \end{aligned} \tag{4.10}$$

where each v_i contains all of the i^{th} entries of elements in S. Moreover, we define w_i as follows

$$\begin{aligned} w_1 &= \left\{\left(t_1, x_1{}^{(1)}\right), \left(t_2, x_2{}^{(1)}\right), \ldots, \left(t_k, x_k{}^{(1)}\right)\right\} \\ w_2 &= \left\{\left(t_1, x_1{}^{(2)}\right), \left(t_2, x_2{}^{(2)}\right), \ldots, \left(t_k, x_k{}^{(2)}\right)\right\} \\ &\vdots \\ w_n &= \left\{\left(t_1, x_1{}^{(n)}\right), \left(t_2, x_2{}^{(n)}\right), \ldots, \left(t_k, x_k{}^{(n)}\right)\right\}, \end{aligned} \tag{4.11}$$

where $t \in R$. We then employ the interpolation approach, PCHIP, [24, 25] to interpolate each w_i to obtain a function f_i and thus compute it at a certain

t

$$f_i(t^*) \approx x_r^{(i)} \qquad \text{for } i = 1, 2, \ldots, n, \tag{4.12}$$

where $t^* \in [k+1, s] \subset R$, and $s \geq k+1$. In other words, the extrapolation is used to produce vector $\mathbf{x}_r$. Note that, due to the weaknesses of the extrapolation method, the values of t cannot go so far from k. In fact, we maintain selecting the new approximate solution with the minimum residual norm. The algorithm of EIEMLA is given in [21].

Some basic theorems related to the monotonicity of the new generations resulted by EIEMLA and the convergence rate can be seen in [26]. For the use of the restarting framework to speed up the convergence of EIEMLA, we refer the reader to [29]. The resulting algorithm is called REIEMLA. In addition, the use of parallel computing of REIEMLA and how to run it on the cloud computing we refer the reader to [29].

The revised version of EIEMLA (MEIEMLA) was proposed to improve the efficiency of EIEMLA in solving high dimensional problems. In this case, MEIEMLA interpolates $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \ldots, \mathbf{x}_k\}$ without re-arranging it as in (4.10). Thus, for $t \in [1, k] \subset R$, we have

$$W = \{(t_1, \mathbf{x}_1), (t_2, \mathbf{x}_2), \ldots, (t_k, \mathbf{x}_k)\}, \tag{4.13}$$

used for the interpolation process. The same results given in (4.12) would be obtained.

4.2 Restarting MEIEMLA : A Way to Speed Up The Convergence of MEIEMLA

In this section, our focus is on MEIEMLA and the speed of its convergence. In this case, our attention goes to a restarting framework. The restarting strategy has been proven in the previous investigations and is used to improve the residual norms of the approximate solutions. Similarly, when MEIEMLA algorithm is run once, it doesn't give us a small residual norm. Thus, we need to put it in the restarting framework. This approach is called hybrid restarting-MEIEMLA, [23]. The algorithm of hybrid restarting-MEIEMLA can be seen in Algorithm 2.

Algorithm 2 RMEIEMLA, [23]

1: Fix the number of iterations to, say k, and the tolerance, ϵ, to $1E - 13$.
2: Run MEIEMLA for k iterations. Obtain the residual norm $\mathbf{r}_{model}$ which is related to index m, as well as the approximate solution $\mathbf{x}_{model}$.
3: **if** $\|\mathbf{r}_{model}\| \leq \epsilon$ **then**
4: The solution is obtained; i.e., the iterate which is associated with this residual norm, $\mathbf{x}_{model}$.
5: Stop.
6: **else**
7: Initialize the algorithm with

$$\begin{aligned} \mathbf{x}_0 &= \mathbf{x}_{model} \\ \mathbf{y} &= \mathbf{b} - A\mathbf{x}_0 \end{aligned}$$

8: Go to 2.
9: **end if**
10: Take $\mathbf{x}_{model}$ as the approximate solution.
11: Stop.

5 Comparisons Between Restarting, Switching, and Hybrid Restarting-MEIEMLA

In this section, we discuss the advantages and disadvantages of the three recent strategies to stabilize the Lanczos-type algorithms. In particular, we compare algorithms RLMinRes, SLUMinRes, and RMEIEMLA which respectively are the implementation of restarting from the iterate with the minimum residual norm, switching from the iterate with the minimum residual norm, and restarting from the iterate generated by MEIEMLA. For the comparison, we implement the three methods to solve SLEs ranging from 10000 to 300000 dimensions. The built systems are generated through the discretization process of the PDE heat equation $-\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} + \gamma \frac{\partial u}{\partial x} = 0$, [31], where the tridiagonal matrix has the main diagonal elements of the form $[-1 + \lambda 4 - 1 - \lambda]$, where λ's can take the values of 0.2 ; 0.5 ; 0.8 ; 5 ; 8. These values vary by the way we compute the matrix numbers which leads to the illness of a system. The bigger λ would result in an ill-conditioned system.

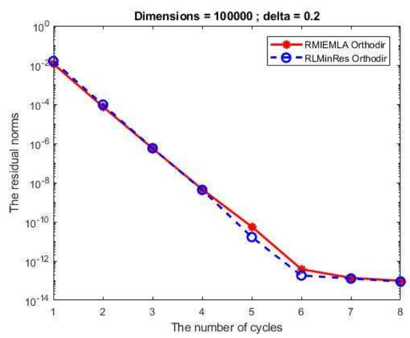
The numerical results are presented in Tables (1) and (3). In addition, we observe the behavior of each method graphically and compute time consumption.

Table 1: Performance of Some Stabilized Lanczos Algorithms in Solving PDE Heat Equation with $\delta = 0.2$.

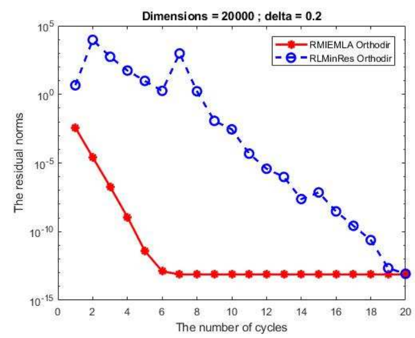
Dim	RLMinRes Orthodir			RMEIEMLA Orthodir			SLMinRes Orthodir		
n	$\|r_{restart}\|$	$T(s)$	cycles*	$\|r_{model}\|$	$T(s)$	cycles*	$\|r_{switch}\|$	$T(s)$	cycles*
10000	$8.4428E-14$	0.4353	8	$9.4653E-14$	6.0224	7	$9.3815E-14$	1.1471	16
20000	$6.6341E-14$	0.9481	9	$7.2280E-14$	11.9899	8	$9.9039E-14$	2.1426	16
30000	$9.0837E-14$	1.2707	8	$5.4071E-14$	17.1562	8	$9.7830E-14$	3.0299	18
40000	$8.0851E-14$	1.5624	8	$6.8725E-14$	24.2096	8	$7.6845E-14$	3.5286	16
50000	$6.4022E-14$	2.2081	9	$7.5717E-14$	28.2761	8	$7.0601E-14$	5.7705	18
60000	$7.1777E-14$	2.7165	9	$8.2005E-14$	34.7594	8	$9.3471E-14$	6.5348	18
70000	$8.6367E-14$	2.5954	8	$6.8601E-14$	49.3165	9	$7.4162E-14$	6.9142	18
80000	$6.8722E-14$	3.2773	9	$8.8157E-14$	48.220	8	$8.2355E-14$	7.8532	18
90000	$6.9133E-14$	4.4366	9	$7.8552E-14$	58.424	9	$8.7809E-14$	9.8722	18
100000	$8.9519E-14$	4.3797	9	$9.4484E-14$	58.5675	8	$8.0343E-14$	9.7309	16
200000	$7.9076E-14$	14.5185	9	$9.6699E-14$	$1.4E+02$	9	$8.2958E-14$	34.7571	18
300000	$8.865E-14$	41.4729	9	$1.2163E-13$	$1.19E+02$	10	$9.4945E-14$	50.7832	17

Table 2: Performance of Some Stabilized Lanczos Algorithms in Solving PDE Heat Equation with $\delta = 0.8$.

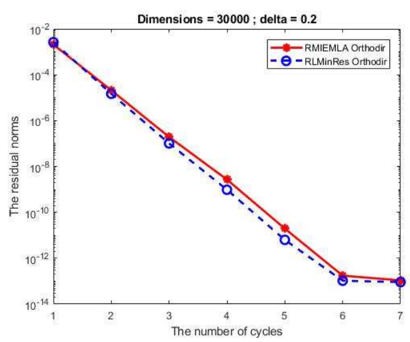
Dim	RLMinRes Orthodir			RMEIEMLA Orthodir			SLMinRes Orthodir		
n	$\|r_{restart}\|$	$T(s)$	cycles*	$\|r_{model}\|$	$T(s)$	cycles*	$\|r_{switch}\|$	$T(s)$	cycles*
10000	$1.5049E-13$	0.3504	4	$9.4653E-14$	6.0224	7	$9.3815E-14$	1.1471	16
20000	$2.6369E-13$	0.8487	5	$7.2280E-14$	11.9899	8	$9.9039E-14$	2.1426	16
30000	$1.0262E-13$	1.0369	12	$5.4071E-14$	17.1562	8	$9.7830E-14$	3.0299	18
40000	$5.9800E-13$	2.4820	6	$6.8725E-14$	24.2096	8	$7.6845E-14$	3.5286	16
50000	$5.1085E-13$	4.4005	8	$7.5717E-14$	28.2761	8	$7.0601E-14$	5.7705	18
60000	$6.2498E-13$	3.2488	5	$8.2005E-14$	34.7594	8	$9.3471E-14$	6.5348	18
70000	$5.2539E-12$	3.6561	5	$6.8601E-14$	49.3165	9	$7.4162E-14$	6.9142	18
80000	$5.8252E-12$	3.3097	4	$8.8157E-14$	48.220	8	$8.2355E-14$	7.8532	18
90000	$6.1577E-12$	4.0325	4	$7.8552E-14$	58.424	9	$8.7809E-14$	9.8722	18
100000	$6.1524E-12$	5.4372	5	$9.4484E-14$	58.5675	8	$8.0343E-14$	9.7309	16
200000	$7.9076E-14$	14.5185	9	$9.6699E-14$	$1.4E+02$	9	$8.2958E-14$	34.7571	18
300000	$8.865E-14$	41.4729	9	$1.2163E-13$	$1.19E+02$	10	$9.4945E-14$	50.7832	17



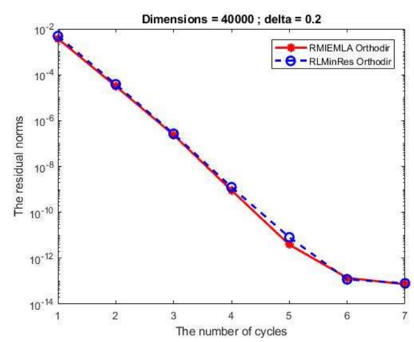
(a) Dim 10000



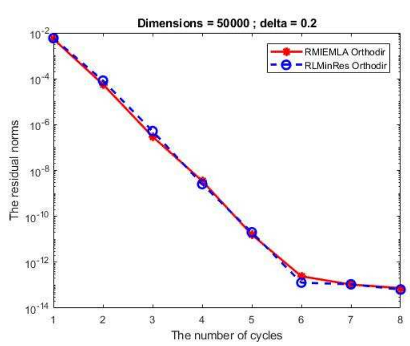
(b) Dim 20000



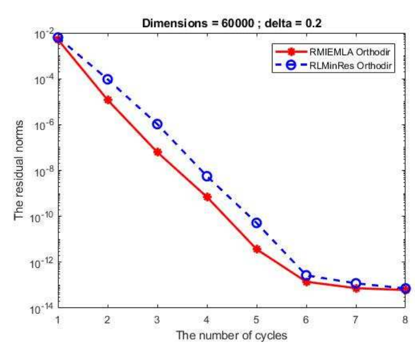
(c) Dim 30000



(d) Dim 40000

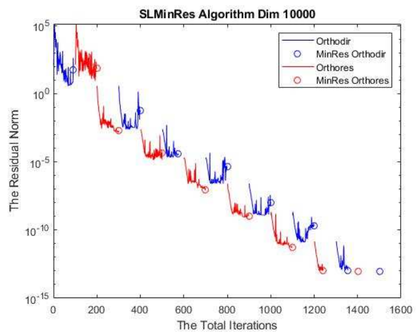


(e) Dim 50000

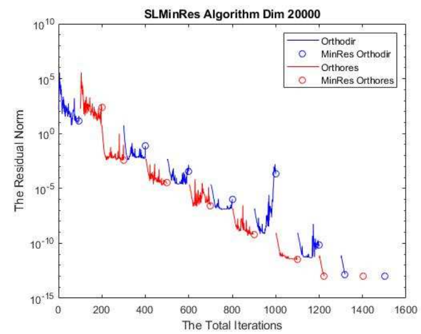


(f) Dim 60000

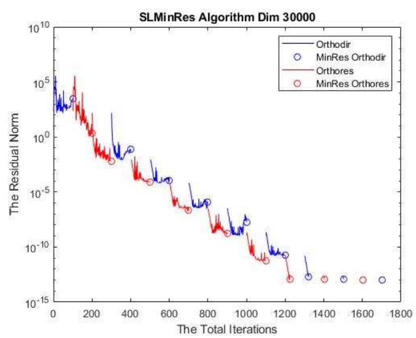
Figure 1: The Performance of RMEIEMLA Orthodir and RLMinRes Orthodir for Solving the Variety of SLE's.



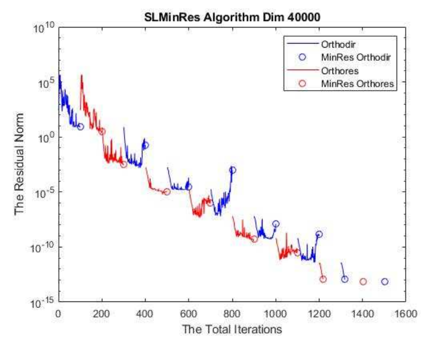
(a) Dim 10000



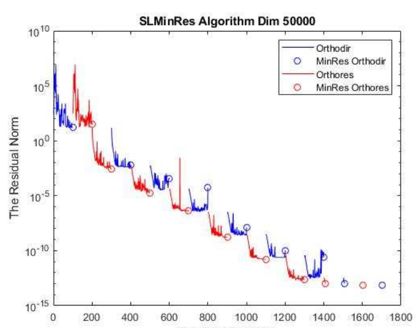
(b) Dim 20000



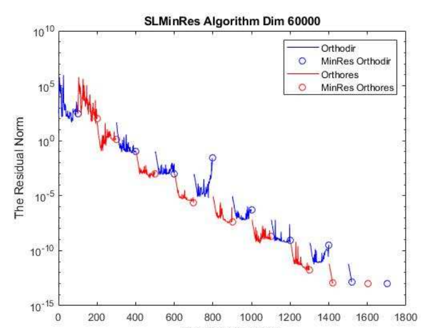
(c) Dim 30000



(d) Dim 40000



(e) Dim 50000



(f) Dim 60000

Figure 2: The Performance of SLMinRes Orthodir/Orthores for Solving the Variety of SLE's.

5.1 Advantages of Algorithm RLMinRes, SLUMinRes, and RMEIEMLA

In general, the strategy of restarting from the iterate with the minimum residual norm (RLMinRes) consumed the least computational time compared with the hybrid restarting-MEIEMLA (RMEIEMLA) and switching from the iterate with the lowest residual norm (SLUMinRes). It can be seen from Table 1 that RLMinRes used only 0.4 seconds to solve 1000 dimensions of SLEs, whereas SLUMinRes and RMEIMLA used 1.15 seconds and 6.02 seconds, respectively. These numbers increased as the dimensions increase. For instance, to solve SLEs dimensions 30000, RMEIMLA used $1.19E^2$ seconds, which is the highest one, while SLUMinRes and RLMinRes spent 50.78 and 41.47 seconds, respectively.

Our interpretation for this condition is that RMEIEMLA consumed the highest time to solve the SLEs compared with the other two methods, due to the interpolation process. However, the number of iterations used in RMEIEMLA is slightly smaller than RLMinRes and SLUMinRes, which is indicated by the number of cycles used. Furthermore, RMEIEMLA has less risk of breakdown occurrence compared with the other two methods. This is due to RMEIEMLA using a smaller number of iterations. Thus RMEIEMLA is more robust compared with RLMinRes and SLMinres.

5.2 Introduction to Support Vector Regression for the Solution Prediction of SLEs

In this section, we introduce the use of support vector regression (SVR) [32] to predict a solution of SLEs. It is, however, different from the nature of SVR as a prediction tool under the machine learning which involves the data set of the observation process. Here, the data set is generated by the Lanczos algorithms for solving SLEs. It is similar to EIEMLA and MEIEMLA in predicting the new solution of Lanczos solutions. In fact, we combine it with the restarting strategy to get better performance. The details of the new approach are as follows

According to Figure 3, once some iterates were generated by Lanczos Orthodir, we collect them as the data set of $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$. Hence, we used them as training data and observed data. For the observed data, in particular, we choose some iterates with small residual norms, including the one with the minimum residual norm, $\mathbf{x}_m$. The idea is that we would get the new solutions as a result of the prediction process which has a similar property

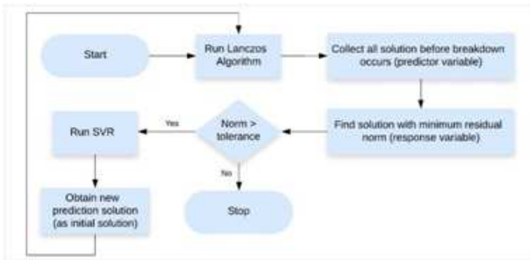


Figure 3: The procedure of hybrid SVR-Lanczos Orthodir for solving SLE's

Table 3: Performance of Some Stabilized Lanczos Algorithms in Solving PDE Heat Equation with $\delta = 0.8$.

Dim(n)	$\|r_{pred}\|$	$T(s)$	cycles*	MAE	MSE	RMSE
1000	$9.5881E-13$	68.408	110	$3.7659E-16$	$6.5482E-29$	$8.0921E-15$
2000	$8.9427E-13$	129.643	109	$5.9686E-16$	$6.3008E-29$	$7.9377E-15$
3000	$8.9585E-13$	171.013	101	$5.0212E-16$	$4.0659E-29$	$6.3764E-15$
4000	$3.5619E-13$	333.881	145	$8.5709E-17$	$4.4436E-30$	$2.1079E-15$
5000	$9.6981E-13$	387.296	128	$5.3379E-16$	$3.9950E-29$	$6.3206E-15$
6000	$6.5599E-13$	362.021	96	$4.3036E-16$	$2.2075E-29$	$4.6984E-15$
7000	$5.1537E-13$	535.780	113	$8.9529E-16$	$7.4105E-29$	$8.6084E-15$
8000	$0.0000E+00$	483.891	90	$0.0000E+00$	$0.0000E+00$	$0.0000E+00$
9000	$1.9863E-12$	721.707	111	$7.8554E-16$	$1.1608E-28$	$1.0774E-14$
10000	$0.0000E+00$	1291.575	192	$1.1102E-20$	$1.2326E-36$	$1.1102E-18$
20000	$6.6319E-13$	1685.536	147	$1.2168E-17$	$1.2964E-30$	$1.1386E-15$
30000	$3.0929E-13$	2029.554	121	$1.8511E-17$	$5.4108E-31$	$7.3558E-16$
40000	$0.0000E+00$	4714.904	197	$5.5511E-21$	$1.2325E-36$	$1.1102E-18$
50000	$1.2535E-12$	2916.554	105	$1.1181E-16$	$1.4026E-29$	$13.7451E-15$

as $\mathbf{x}_m$ but different from Krylov basis. All the procedures formulated under the hybrid SVR-Lanczos (SVR-L) algorithm are presented in Algorithm 3.

As the results, we implement the Algorithm 3 to solve several SLEs problems that are generated by a discretization process of PDE heat equation as explained in the previous section. These are presented in the following table

We can see, from Table 3, that the residual norms seem small (although they are slightly larger than the results of RMEIEMLA) with the significant errors of MAE, MSE, and RMSE. This means that the prediction solution using SVR performed well.

5.3 Limitations and Future Works

Besides the advantages, the three algorithms have some weaknesses. RLMinRes and SLUMinRes, for instance, cannot be more accurate than $1E^{-14}$. This is because we involve the minimum residual norm as the base of the updated

Algorithm 3 Hybrid Restarting SVR-Lanczos Algorithm

Fix the number of iterations to, say k, and the tolerance, ϵ, to $1E-13$.
while $norm_{pred} \geq \epsilon$ **do**
    Run a Lanczos-type algorithm (say Orthodir) for k iterations, and collect

$$S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m, \ldots, \mathbf{x}_k\}, \tag{5.14}$$
$$R = \{\|\mathbf{r}_1\|, \|\mathbf{r}_2\|, \ldots, \|\mathbf{r}_m\|, \ldots, \|\mathbf{r}_k\|\}, \tag{5.15}$$
$$sol_{min} = \mathbf{x}_m. \tag{5.16}$$
$$norm_{min} = \|\mathbf{r}_m\|. \tag{5.17}$$

    for $S_i = \{\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \ldots, \mathbf{x}_{i,m}, \ldots, \mathbf{x}_{i,k}\}$ **do**
        Set all values in S_i as predictor variables and sol_{min} as response variable
        Predict a new value in each row $\hat{\mathbf{y}}_i$
    end forSTATE

$$\begin{aligned} sol_{pred} &= \hat{\mathbf{y}}_i \\ norm_{pred} &= \|\mathbf{b} - A\hat{\mathbf{y}}_i\| \end{aligned}$$

end while
Take $\mathbf{x}_{ppred}$ as the approximate solution.
Stop.

iterate. When we run the algorithm RLMinRes or SLUMinRes, after sometime the algorithm reaches $index = 1$ (the index is the position number of the lowest residual norm). Consequently, if we increased the tolerant say to $1E - 16$, then the looping inside of the algorithm cannot stop because there is only one residual norm remaining and that is still bigger than the tolerant. Similar to RMEIEMLA, it also involves the computation of the minimum residual norm, which causes the same problem as the RLMinRes and SLMinRes. Therefore, we suggest the investigation of these three methods in order to advance the stabilization of Lanczos algorithms. Moreover, our suggestion goes for implementation of such approaches of restarting, switching, and exploiting patterns in non-linear systems. For further study, one may look at some problems such as elliptic, hyperbolic, and parabolic equations which are included in partial differential equations problems.

6 Conclusion

We have reviewed some recent strategies to stabilize the Lanczos-type algorithms for solving high dimensions of SLEs. These are restarting and switching from quality points and an embedding interpolation function in Lanczos algorithms. As a result of these strategies, the new Lanczos variants such as RLLastIt, RLMedVal, RLMinRes, SLMinRes, SLUMinRes, REIEMLA, and RMEIEMLA are generated. All of them have been established empirically and we have presented three of them, RLMinRes, SLMinRes, and RMEIEMLA. We compared them in terms of robustness and efficiency to solve several SLEs, ranging from 10000 to 300000 dimensions and concluded that RLMinRes is the best method compared with SLMinRes and RMEIEMLA in term of efficiency, while RMEIEMLA is the best in terms of robustness. However, further investigations still await for some of reasons that we explained in the previous section.

Acknowledgment

This paper was supported by the Research intensified Grant Scheme (RIGS) Fasa 1/2019, Vot Number 55192/7, Universiti Malaysia Terengganu. We would like to thank Prof. Dr. Abdellah Salhi for ideas of EIEMLA and the switching strategy in this paper.

References

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

[2] C. Lanczos, , Solution of Systems of Linear Equations by Minimized Iteration, Journal of Research of the National Bureau of Standards, **49**, (1952), 33–53.

[3] Y. Saad, Krylov Subspace Method for Solving Large Unsymmetric Linear Systems, Mathematics of Computations, **37,** no. 155, (1981), 105–126.

[4] Y. Saad, Iterative Methods for Sparse Linear Systems, 3^{nd} ed., Philadelphia: Society for Industrial and Applied Mathematics, (2003) , 5–10.

[5] C. Brezinski, R. H. Zaglia, A New Presentation of Orthogonal Polynomials with Applications to their Computation, Numerical Algorithm, **1**, (1991), 207–221.

[6] C. Brezinski, H. Sadok, Lanczos-type Algorthms for Solving Systems of Linear Equation, Applied Numerical Mathematics, **11**, (1993), 443–473.

[7] C. Brezinski, R. Zaglia, H. Sadok, A Review of Formal Orthogonality in Lanczos-based Method, Journal Of Computational and Applied Mathematics, **140**, (2002), 81–98.

[8] C. Baheux, New Implementations of Lanczos Method, Journal Of Computational and Applied Mathematics, **57**, (1995), 3–155.

[9] M. Farooq, 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.

[10] Zakir Ullah, Muhammad Farooq, Abdellah Salhi, A19/B6: A new Lanczos-type algorithm and its implementation, Journal of Prime Research in Mathematics, **11,** (2015), 106–122.

[11] M. Farooq, A. Salhi, A Preemptive Restarting Approach to Beating Inherent Instability, Iranian Journal of Science and Technology Transaction a Science, **37**, (2013), 349–358.

[12] M. Farooq, A. Salhi, A Switching Approach to Beating the Inherent Instability of Lanczos-type Algorithms, Journal of Applied Mathematics and Information Systems, **8**, no. 5, (2014), 2161–2169.

[13] C. Brezinski, R. Zaglia, Breakdowns in the Computation of Orthogonal Polynomials, Springer, Dordrecht, **296,** (1994), 49–59.

[14] C. Brezinski, R. Zaglia, Breakdowns in the Implementation of Lanczos Method for Solving Linear Systems, Comput. Math. Appl., **33**, (1997), 31–44.

[15] C. Brezinski, R. Zaglia, H. Sadok, Avoiding Breakdown and Near-Breakdown in Lanczos-type Algorithms, Numerical Algorithm, **1**, no. 2, 1991, 261–284.

[16] C. Brezinski, R. Zaglia, H. Sadok, A Breakdown-Free Lanczos-type Algorithm for Solving Linear Systems, Numerical Mathematics, **63**, no. 1, (1992), 9–38.

[17] C. Brezinski, R. Zaglia, Look-Ahead in Bi-CGSTAB and Other Methods for Linear Systems, BIT, **35**, no. 2, (1995), 169–201.

[18] Ronald B. Morgan, On Restarting the Arnoldi Method for Large Nonsymmetric Eigenvalue Problems, Mathematics of Computation, **65**, (1996), 1213–1230.

[19] P. Grave-Morris, A Look-Around Lanczos Algorithm for Solving a System of Linear Equations, Numerical Algorithm, **15**, (1997), 247–274.

[20] Maharani Abu Bakar, Abdel Salhi, Restarting from Specific Points to Cure Breakdown in Lanczos-type Algorithms, Journal of Mathematical and Fundamental Sciences, **2**, no. 47, (2015), 167–184.

[21] Maharani Abu Bakar, Abdel Salhi, Wali Khan Mashwani, Enhancing the Stability of Lanczos-type Algorithms by Restarting The Point Generated by EIEMLA for the Solution of Systems of Linear Equations, International Science Journal, Lahore, **28**, no. 4, (2016), 3325–3335.

[22] Maharani Abu Bakar, Abdel Salhi, Alifhar Juliansyah, New Switching Strategy from the Quality points for Unfixed Iterations to Tackle Breakdown in Lanczos-type Algorithms, *Preprint*, DOI : 10.13140/RG.2.3.10407.16809, (2018).

[23] Maharani Abu Bakar, Abdel Salhi, Wali Khan Mashwani, Lanczos-type Algorithms with Embedded Interpolation and Extrapolation Models for Solving Large Scale Systems of Linear Equations, International Journal of Computing Science and Mathematics, **10**, no. 5, (2019), 429–442.

[24] F. N. Fritsch, R. E. Carlson, Monotone Piecewise Cubic Interpolation, SIAM Journal Numerical Analysis, **17**, no. 2, (1980), 239–248.

[25] F. N. Fritsch, J. Butland, A Method for Constructing Local Monotone Piecewise Cubic Interpolants, SIAM J. Sci. Stat. Comput., **5**, no. 2, (1984), 300–304.

[26] Maharani Abu Bakar, Abdel Salhi, Wali Khan Mashwani, Yeniay Özgür, Niken Larasati, Tryani Triyani, Solving Large Scale Systems of Linear Equations with a Stabilized Lanczos-type Algorithms Running on a Cloud Computing Platform, Hacettepe Journal of Mathematics and Statistics, **47**, no. 6, (2018), 1730–1741.

[27] A. Sidi, Vector Extrapolation Method with Applications to Solution of Large Systems of Equations and to PageRank Computations, Technical report, Computer Science Department Technion, Israel Institute of Technology, 2006.

[28] A. Sidi, Review of Two Vector Extrapolation Methods of Polynomial Type with Applications to Large-Scale Problems, Journal of Computational Science, **3**, (2012), 92–101.

[29] Maharani Abu Bakar, Abdel Salhi, Introduction of Interpolation and Extrapolation Models in Lanczos-type Algorithms A13B6 and A13B10 to Enhance their Stability, Journal of Mathematics and Fundamental Sciences, **50**, no. 2, (2018), 148–165.

[30] Maharani Abu Bakar, Abdel Salhi, RMEIEMLA: The Advance in Improving Robustness of Lanczos-type Algorithms, AIP Conference Proceedings: The 4th Innovation and Analytics Conference and Exhibition, AIP Conference Proceedings, 2138(1):030009, (2019).

[31] Richard L. Burden, J. Douglas Faires, Numerical Analysis, 9th edition, Brooks/Cole, 2019.

[32] M. Khun, K. Johnson, Applied Predictive Modeling, Springer, 2013.