Applied Mathematics and Computational Intelligence
Volume 6, 2017 [83-98]

Solving Large System of Linear Equation Using Successive Over-Relaxation, Conjugate Gradient and Preconditioned Conjugate Gradient

Nurhanani Abu Bakar[1,a] and Mohd Rivaie[1,b]

[1]*Department of Computer Science and Mathematics, MARA University of Technology (UiTM) Terengganu, Campus Kuala Terengganu, Malaysia*

ABSTRACT

System of linear equations is widely used especially in industries. Industrial mathematical applications usually involve large problem. System of linear equation can be solved using direct and indirect method. Direct method requires the use of inverse matrix to solve the problem. However, for large system of linear equation, finding an inverse could be difficult and time consuming. Therefore, indirect methods in the form of numerical calculation such as Successive Over-Relaxation, Conjugate Gradient and Preconditioned Conjugate Gradient are used. The aim of this research is to compare the performance of those three methods to solve variety of system of linear equation from small scale to large scale in term of number of iteration and CPU time. Numerical results show that the Conjugate Gradient method is the best method for solving system of linear equation in terms of both number of iteration and CPU time. Above all, those three methods could be used to solve system of linear equations.

Keywords: System of linear equation, Iterative method, Successive Over-Relaxation, Conjugate Gradient, Preconditioned Conjugate Gradient

1. INTRODUCTION

Matrix form of type $Ax = b$ is one of the topics that are commonly applied in many areas, like structural design, principal component analysis, exploration and remote sensing, biology, electricity, solid mechanics, molecular spectroscopy, structural dynamics, automatics control theory, vibration theory, and others (Konghua et al., 2007). Linear systems arise in real-world for various applications. A system of linear equations is a collection of two or more linear equations involving the same set of unknowns (Gharib et al., 2015). The word "system" indicates that the equations are to be considered collectively rather than individually.

A linear equation in n unknowns is an equation of the form:

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n = b$$

where $a_1, a_2, \ldots, a_n$ and b are real numbers and $x_1, x_2, \ldots, x_n$ are variable (Leon, 2010). A linear system of m equations in n unknowns is then a system in the form:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned}$$

[a] nurhanani143@yahoo.com, [b] rivaie75@tganu.uitm.edu.my

where the a_{mn}'s and the b_m's are all real numbers. Commonly, system of linear equation is converted to matrix form of type $Ax = b$ (Shastri et al., 2015). Considering a system of linear equations of three variables

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= b_3 \end{aligned}$$

The matrix form of type $Ax = b$ will be as below:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

From the above matrix form, the left-hand side matrix or the coefficient matrix refers to the matrix A. Whereas, the right-hand side matrix refer to the matrix b. Then, the value of x_1, x_2 and x_3 represent the solution of the linear system.

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, b = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

In fact, many industries are facing with problem of high dimensional system. Despite that, the problem can be solved by using iterative methods as it may take less time and lower cost compare to direct solver. System of linear equations is widely used especially in industries. System of linear equation can be solved using direct and indirect method. Some researchers such as Kacamarga et al., (2014), Jamil (2012) and Shastri et al., (2015) studied the comparisons between direct and indirect methods. They mentioned that the indirect methods become more efficient than the direct method. In this research, Successive Over-Relaxation (SOR), Conjugate Gradient (CG) and Preconditioned Conjugate Gradient (PCG) method are chosen to solve the system of linear equations.

2. DIRECT METHODS

The direct methods of solving linear equations are known to have their difficulties when using manual and computer calculation. For instance, the problem with Gauss elimination is sensitive to rounding error (Kalambi, 2008). Generally, the solution is obtained after a single application of Gaussian elimination. This can be a problem because once a solution is obtained, there is no improvement can be done to the solution. Besides, solving large system of equation is a burden to the computer in term of computational time due to the high memory storage. This make a direct method too difficult to be used as it need lots of step to be carried out. More importantly, the operations cost of Gaussian elimination is too high for most large systems due to the high computational memory requirement.

3. LITERATURE REVIEW

System of linear equations are collections of linear equations that consist of many linear equations with the same variables (Jamil, 2012). The system might consist many unknown variables which must be solved simultaneously in the form of $Ax = b$. There are many methods proposed by other researcher that studied about this system. Rodríguez-Alvarez et al. (2010) have applied this system in image reconstruction and proposed the QR decomposition to

solve the linear system. Refer to Liu (2011), the direct methods such as Gauss-Jordan and LU decomposition are slower than Jacobi, Richardson and Gauss-Seidel which are known as iterative methods. On the other hand, Homotropy Perturbation method showed the best results in term of errors and computational times.

According to Atasoy et al., (2012), the system of linear equations can be applied in many scientific and engineering problems. The solution of Linear Circuit Equation System (LCES) is solved by using Gauss-Jordan Elimination Method and examined in terms of CPU time. Besides, the study about classical iterative refinement (IR) and k-fold iterative refinement (RIR) for solving linear equation in the form of $Ax = b$ has shown that RIR is very stable and robust compared with IR which is effective but not stable enough (Smoktunowicz & Smoktunowicz, 2013). Linear system $Ax = b$ with matrix A is a circulant matrix normally used to find numerical solution for elliptic and parabolic partial differential equation. Fuyong (2014) stated that the circulant matrices are commonly used to solve problem in many areas for examples physics, electromagnetic, signal processing, statistics and applied mathematics.

In contrast, Dimov et al., (2015) have proposed a new Walk on Equation (WE) Monte Carlo algorithm to solve system of linear algebraic (LA) equations. The comparison of WE Monte Carlo and Preconditioned Conjugate (PCG) method showed that convergence for WE Monte Carlo is better. Furthermore, research about ill conditioned linear system is presented by Douglas et al., (2016). They have used Krylov subspace method to solve sparse matrix which has a large condition number. Afterwards, Yeung et al., (2017) present a preconditioner to improve the convergence of solution to compute ill conditioned system. These problems are solved by using Generalized Minimal Residual (GMRES) method and modified Biconjugate Gradient Stabilized (MBiCG) method.

Based on this literature review, it can be seen that, most researchers interested to solve system of linear equations using direct and indirect methods. Therefore, this research focused on solving the system of linear equations with various type of matrix using three different iterative methods.

4. FUNDAMENTAL OF ITERATIVE METHODS

Iterative method can be used to approximate solutions that converge rapidly or slowly. Iterative method have become a famous method among scientific computing field to compute large number of unknown equations (Sickel et al., 2005). Iterative method is a mathematical solver that improves approximate results for some problems. According to Collignon (2011), there are many types of iterative methods. Jacobi, Gauss Seidel and Successive Over-relaxation Method (SOR) method are the basic iterative techniques for solving linear systems (Saad, 2003). Some of the iterative methods such as Jacobi and Gauss-Seidel could be used with some success but mainly because of the limited amount of memory that is required (Saad & Van Der Vost, 2000). In practical computations, it is shown that Gauss-Seidel iteration converges slowly. Consequently, SOR is used to accelerate convergence by taking a bigger step at each iteration.

Theoretically, Jacobi and Gauss-Seidel method are slower than SOR to solve large linear system with the best chosen value of extrapolation factor (Kalambi, 2008). The SOR method is devised by applying extrapolation value to the Gauss-Seidel method. In order to accelerate the rate of convergence, a suitable value of extrapolation factor, ω is selected. However, there are still disadvantages of SOR method since the convergence of the solution depend on the value of extrapolation factor denoted as ω.

The Conjugate Gradient (CG) is efficient for solving systems of linear equations (Li et al., 2008). The CG method also can be used for minimization and it is related to a minimization method

known as steepest descent. For problems $Ax = b$ of large dimension, and the matrix is ''ill-conditioned'', it is often possible to improve the performance of the conjugate gradient method by using a technique known as preconditioning. A matrix is ill-conditioned if its determinant approaches zero and may suffer with numerical errors. In some case, where the eigenvalues are close to the origin, the iterative methods tend to have many iteration before the stopping criteria is met. This may cause the unstable system does not converge to the solution point (Douglas et al., 2016).

Besides, an ill-conditioned system can be determined based on its condition number. The condition number for a system can be expressed as $K = \frac{\lambda_{max}}{\lambda_{min}}$, where λ_{max} and λ_{min} are the largest and smallest eigenvalues of A. The high value of condition number is considered as ill-conditioned (Burden & Faires, 2011). The Preconditioned Conjugate Gradient (PCG) is one of the most efficient methods for solving systems with symmetric positive definite coefficient matrices (Arany, 1999).

5. SUCCESSIVE OVER-RELAXATION (SOR) METHOD

SOR method is a generalization of and improvement on the Gauss-Seidel (GS) Method. This method is an extrapolation of the GS method which calculate the GS method's iterates to accelerate its convergence (Mittal, 2014). The source of the SOR method is the observation that GS converges monotonically. The approximations x^k stay on the same side of the solution x as iteration number increases. SOR method introduced the acceleration parameter or extrapolation factor denoted as ω. Listed below is the formula for SOR method:

$$x_i^{(k)} = (1-\omega)x_i^{(k-1)} + \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right]$$

where,
x^k : k-th iteration of x
ω: Extrapolation factor
a : refer to the element of matrix A based in row, i and column, j
b : refer to the element of matrix b based in row, i and column, j

The algorithm below shows the calculation for steps in SOR method.

Input N: No. of unknowns and equation
a_{ij}: Entries of $A, i, j = 1 \dots N$
b_{ij}: Entries of $b, i = 1 \dots N$
Error Tolerance = 10^{-6}, ω

Output $x_i^{(k)}$: Entries of $x_{(k)}$

Algorithm 1. Step in SOR method

Step 1 Choose an initial guess $x^{(0)} = (0)$ to the solution x. For k = 1, 2, 3,..., k_{max} and For i = 1, 2,..., N

Step 2 $$x_1^{(1)} = (1-\omega)x_1^{(0)} + \frac{\omega}{a_{11}}\left(b_1 - a_{12}x_2^{(0)} - a_{13}x_3^{(0)}\right)$$

Step 3 $$x_2^{(1)} = (1-\omega)x_2^{(0)} + \frac{\omega}{a_{22}}\left(b_2 - a_{21}x_1^{(1)} - a_{23}x_3^{(0)}\right)$$

Step 4 $x_3^{(1)} = (1-\omega)x_3^{(0)} + \frac{\omega}{a_{33}}\left(b_3 - a_{31}x_1^{(1)} - a_{32}x_2^{(1)}\right)$

Step 5 Repeat for $x_i^{(k)}$ which are k_{th} iteration of x and i, number of unknowns.

Step 6 Repeat the calculations until tolerance factor are achieve

Extrapolation factor

Extrapolation factor is the process of estimating, beyond the original observation range, the value of a variable on the basis of its relationship with another variable. The SOR is devised by applying extrapolation to GS. In this extrapolation, the form of a weighted average is taken between the previous iterate and the current GS iterate successively for each component (Chen, 2014).

$$x_i^{(k)} = \omega\bar{x}_i^{(k)} + (1-\omega)x_i^{(k)}$$

where $\bar{x}_i^{(k)}$ denotes a GS iterates for i^{th} component of $\bar{x}^{(k)} = (\bar{x}_1^{(k)}, \bar{x}_2^{(k)}, \dots, x_N^{(k)})^T$ and ω is the extrapolation factor. To accelerate the rate of convergence, a suitable value of ω is selected. There are five cases of extrapolation factor (Barrett et al., 1994). Table 1 shows the conditions of ω for each cases.

Table 1 Conditions of extrapolation factor

Cases	Extrapolation factor	Condition
I	$\omega = 1$	Simplifies to the Gauss-Seidel
II	$\omega = 0$	No iteration
III	$0 < \omega < 1$	Under-relaxation
IV	$1 < \omega < 2$	Over-relaxation
V	$\omega \geq 2$	Divergent

Regrettably, there is no general rule for selecting the optimal value of the relaxation factor ω. Besides, some researcher such as Gismalla (2014) use 1.25 for value of over-relaxation parameter which is the same with Burden & Faires (2011). In contrast, according to Eiermann & Varga (1993), $\omega = 1.88401$ was used as an extrapolation factor in order to get a better results. However, author like Yang et al., (2005) and Barrett et al., (1994) showed that $\omega = 1.20$ is the best extrapolation parameter. Therefore, in this project, the value of extrapolations factor are varied which are set as 0.50, 1.25 and 1.90.

6. CONJUGATE GRADIENT METHOD

The Conjugate Gradient method (CG) is the method designed for solving symmetric positive definite (SPD) linear system (Modersitzki, 1994). The conjugate gradient method can be observed theoretically as a direct method, as it produces the exact solution after a finite number of iterations with several conditions (Ryaben'kii & Tsynkov, 2006). Amazingly, CG method can be used as an iterative method as it provides monotonically improving approximations value of x to the exact solution (O'Leary, 1979). Basically, monotone can be divided into two which are increasing and decreasing. Monotone increasing is also known as a strictly increasing that always increasing and never remains constant. Meanwhile, monotone decreasing is also known as a strictly decreasing that always decreasing and never remain constant (Weisstein, 2002). Monotonic convergence can be related with performance that improves each iteration. Also, the rate of convergence show the effectiveness of solution point to converge (O'Leary, 1979). Then,

CG method is known to overcome this problem without needing any parameter. Research had shown that CG method is more effective due to the CPU times which is faster than Jacobi (Kacamarga et al., 2014). In addition, Hestenes and Stiefel (1952) stated that CG method has the advantages where an improvement can be made by taking one additional step. This concluded that CG method is better than Jacobi, Gauss Seidel and SOR method.

The algorithm below shows the calculation for steps in CG method.

Algorithm 2. Step in CG method

Step 1 Let $x_{(0)} = 0, r_{(0)} = b - Ax_{(0)}, p_{(0)} = r_{(0)}$. For $k = 1, 2, 3, \ldots$

Step 2 $\alpha_{(0)} = \frac{r_{(0)}^T r_{(0)}}{p_{(0)}^T A p_{(0)}}$, when $k = 1$

Step 3 $x_{(1)} = x_{(0)} + \alpha_{(0)} p_{(0)}$

Step 4 $r_{(1)} = r_{(0)} - \alpha_{(0)} A p_{(0)}$

Step 5 $\beta_{(1)} = \frac{r_{(1)}^T r_{(1)}}{r_{(0)}^T r_{(0)}}$

Step 6 $p_{(1)} = r_{(1)} + \beta_{(0)} p_{(0)}$

Step 7 Repeat step 2 to step 6 for k_{th} iteration of x

Step 8 Repeat the calculations until tolerance factor are achieve

7. PRECONDITIONED CONJUGATE GRADIENT METHOD

The CG method is an excellent algorithm, being both extremely fast but also very easy to implement. However, through numerical experiments, it turns out that a lot of times the CG method simply does not converge as fast as it should (Bai & Wang, 2006). If the matrix A is ill conditioned, the CG method may suffer from numerical errors (Benhamadou, 2007). Consequently, Preconditioning Conjugate Gradient method will be used to overcome the sensitivity of rounding off.

Additionally, slow convergence problem because of the system is ill-conditioned. System of linear system that has small condition number is called as well-conditioned. Whereas, a high condition number indicates ill-conditioning system (Kreyszig, 2010). An ill-conditioned system can be determined based on high condition number. The condition number for a system can be expressed as $K = \frac{\lambda_{max}}{\lambda_{min}}$. It can be shown that for the conjugate gradient method, the number of iterations required to reach convergence is proportional to the condition number. Preconditioning is a technique of reducing the condition number in order to improve the rate of convergence of an iterative process (Benzi, 2002).

The algorithm below shows the calculation for steps in PCG method.

Algorithm 3. Step in PCG method

Step 1 $r_{(0)} = b - Ax_{(0)}, p_{(0)} = z_{(0)}$. For $k = 0,1,2,3, \ldots$

Step 2 $z_{(0)} = M^{-1} r_{(0)}$

Step 3 $\alpha_{(0)} = \frac{r_{(0)}^T z_{(0)}}{p_{(0)}^T A p_{(0)}}$

Step 4 $x_{(1)} = x_{(0)} + \alpha_{(0)} p_{(0)}$

Step 5 $r_{(1)} = r_{(0)} + \alpha_{(0)} A p_{(0)}$

Step 6 $z_{(1)} = M^{-1} r_{(1)}$

Step 7 $\beta_{(0)} = \frac{z_{(1)}^T r_{(1)}}{z_{(0)}^T r_{(0)}}$

Step 8 $p_{(1)} = z_{(1)} + \beta_{(0)} p_{(0)}$

Step 9 Repeat step 2 to step 9 for k_{th} iteration of x

Step 10 Repeat the calculations until tolerance factor are achieve

8. STOPPING CRITERIA

The stopping criteria for each method are the same. The stopping criterion of an iterative method is to iterate until:

$$\frac{\left\|x^{(k)} - x^{(k-1)}\right\|_\infty}{\|x^{(k)}\|_\infty} < \varepsilon, x^{(k)} \neq 0$$

This means that, if $x^{(k-1)}$ is an approximation to value of $x^{(k)}$, then the absolute error is $\left\|x^{(k)} - x^{(k-1)}\right\|_\infty$, and relative error is $\frac{\|x^{(k)} - x^{(k-1)}\|_\infty}{\|x^{(k)}\|_\infty} < \varepsilon$ provided that $x^{(k)} \neq 0$. Besides, some researcher such as Gismalla (2014) use 5×10^{-5} for tolerance factor. While Burden & Faires (2011) use 10^{-3} for their calculation that involve tolerance factor. In contrast, according to Aparecido et al., (2014), they use 10^{-14} as tolerance factor in order to get a better results. Authors like Yang et al., (2005) and Barrett et al., (1994) showed that $\frac{\|x^{(k)} - x^{(k-1)}\|_\infty}{\|x^{(k)}\|_\infty} < 10^{-6}$ is the best stopping criteria. Therefore, in this research, the value of tolerance factor are varied where ε is set as $10^{-3}, 10^{-6}$ and 10^{-9}. The following stopping criterion is used:

$$\frac{\left\|x^{(k)} - x^{(k-1)}\right\|_\infty}{\|x^{(k)}\|_\infty} < \varepsilon$$

9. DATA AND INITIAL VALUES

Matrix A is created in five different dimension sizes which are $2 \times 2, 3 \times 3, 5 \times 5, 10 \times 10$ and 20×20. Similarly, matrix b are generated for five different dimension sizes which are $2 \times 1, 3 \times 1, 5 \times 1, 10 \times 1$ and 20×1. For non-symmetric matrix of matrix A, only three different dimension sizes involved which are $2 \times 2, 3 \times 3$ and 5×5. Also, for matrix b, three different dimension sizes which are $2 \times 1, 3 \times 1$ and 5×1. Matrix A is generated for both symmetric and non-symmetric. Both positive definite and negative definite of matrix are used in order to test the ability of methods to find the approximate solution. The two type of matrix are chosen due to the existence of real number of eigenvalues.

Initial point is also known as starting point which indicates the starting point of calculation. Normally, initial point will be started at zero. However, in this research, the initial point is set at five different points as stated in the table of numerical results. The initial point are selected from point close to the solution point to the one furthest away.

10. RESULT ANALYSIS

In this research, the result will be computed by using norm wise absolute error (Ipsen, 2009). Error will be defined by using the following formula:

$$\text{Absolute error} = \|x - \tilde{x}\|_\infty$$
where, $x \neq 0$ or $\tilde{x} \neq 0$
x: Exact value
$\tilde{x}$: Approximate value

The solution from direct method represents the exact value while the results obtained from iterative methods which are SOR, CG and PCG method become the approximate value. Then, the relative errors for each method are computed to determine which method has the smallest error. The number of repeating step in each method are counted as number of iteration. The number of iteration is counted until the tolerance factor is achieved. The most efficient method that has less iteration within the same stopping criteria had been identified and concluded. In contrast to the highest iteration, this method is concluded as less efficient.

The related methods are tested using the same computer with the same processor that is Intel® Core™ i5-7200U CPU @ 2.50GHz 2.71GHz with RAM 4.00 GB. The same computer with the same processor is used because different processor will lead to the changeable results of CPU times.

The table in the following section shows the result for symmetric matrix in two aspects: number of iterations and CPU times. Each aspect consists of two types of matrix which is positive and negative definite matrix. The positive and negative definite involves three different value of tolerance factor. However, the table shown only considers numerical results of tolerance factor of 10^{-9}. This is because, 10^{-9} is the best stopping criteria which give small number of errors. For detailed result, thesis Abu Bakar (2017) can be referred. Thus, the best types of matrix can be determined with the help of performance profile except for results of error. The SOR 1, SOR 2 and SOR 3 represent the SOR : $\omega = 0.5$, $\omega = 1.25$ and $\omega = 1.9$.

10.1 Numerical Results Based on Errors

The numerical results of error for symmetric positive definite (SPD) matrix and symmetric negative definite (SND) matrix are shown in Table 2 and Table 3. The word 'F' refer to Fail, indicating that the method is unable to converge to the solution points.

Table 2 Numerical results of error for SPD

MATRIX	INITIAL POINT	SOR 1	SOR 2	SOR 3	CG	PCG
2x2	(0,0)	2.2858E-09	9.0463E-11	7.6583E-10	0.0000E+00	0.0000E+00
	(10,10)	2.8930E-09	1.0232E-10	5.6666E-10	9.9920E-16	F
	(100, 100)	2.3170E-09	3.0499E-10	5.9934E-10	5.1167E-14	F
	(-10,-10)	2.4081E-09	9.4273E-11	7.4621E-10	1.0968E-14	F
	(-100,-100)	2.6317E-09	3.0868E-10	6.1541E-10	2.4394E-14	F

3x3	(0,0,0)	6.1717E-09	1.5961E-10	1.2690E-09	0.0000E+00	0.0000E+00
	(10,10,10)	6.3259E-09	3.5915E-11	9.1498E-10	1.0408E-14	F
	(100,100,100)	5.9009E-09	8.0817E-11	1.5100E-09	3.0602E-14	F
	(-10,-10,-10)	6.1662E-09	4.1962E-11	1.0559E-09	1.4753E-14	F
	(-100,-100,-100)	6.0622E-09	7.8599E-11	1.5371E-09	5.7001E-14	F
5x5	(0,..,0)	1.9699E-09	5.5282E-10	1.0266E-09	8.5657E-15	8.5657E-15
	(10,..,10)	2.4291E-09	8.2008E-10	9.9461E-10	9.4934E-15	F
	(100,..,100)	2.4488E-09	5.0525E-10	1.0441E-09	2.2374E-14	F
	(-10,,..,-10)	2.1285E-09	7.2564E-10	9.7795E-10	1.1782E-14	F
	(-100,..,-100)	2.2598E-09	4.9736E-10	1.0411E-09	1.6325E-13	F
10x10	(0,..,0)	2.9464E-09	2.3906E-10	4.7908E-10	8.8998E-15	8.8998E-15
	(10,..,10)	2.8818E-09	1.5589E-10	4.3769E-10	8.9928E-15	F
	(100,..,100)	2.6968E-09	2.1506E-10	4.7197E-10	1.3591E-14	F
	(-10,,..,-10)	3.1894E-09	1.7464E-10	4.7918E-10	1.5973E-15	F
	(-100,..,-100)	2.7243E-09	2.1669E-10	4.8020E-10	1.4233E-14	F
20x20	(0,..,0)	3.4008E-09	1.6830E-10	4.2795E-10	7.2643E-16	7.2643E-16
	(10,..,10)	3.2825E-09	2.2698E-10	4.1166E-10	1.0404E-15	F
	(100,..,100)	3.4196E-09	1.4390E-10	4.3698E-10	3.6069E-14	F
	(-10,,..,-10)	3.4341E-09	9.3534E-11	4.7395E-10	3.6066E-15	F
	(-100,..,-100)	3.4351E-09	1.4574E-10	4.4668E-10	1.1124E-14	F
	TOTAL	1.6972E-08	7.7846E-10	2.1972E-09	5.2566E-14	7.2643E-16

Table 3 Numerical results of error for SND

MATRIX	INITIAL POINT	SOR 1	SOR 2	SOR 3	CG	PCG
2x2	(0,0)	2.5716E-09	3.2702E-10	7.7981E-10	0.0000E+00	0.0000E+00
	(10,10)	2.5173E-09	9.1126E-11	5.3844E-10	1.4445E-14	F
	(100, 100)	2.6270E-09	6.1599E-11	7.3848E-10	1.4445E-14	F
	(-10,-10)	2.3054E-09	6.7224E-11	5.3902E-10	0.0000E+00	F
	(-100,-100)	2.4641E-09	5.8372E-11	6.9563E-10	1.0214E-14	F
3x3	(0,0,0)	2.4190E-09	8.7807E-11	6.4794E-10	5.4097E-15	5.4097E-15
	(10,10,10)	2.5760E-09	1.4570E-10	5.0380E-10	5.4097E-15	F
	(100,100,100)	2.3463E-09	4.0106E-11	7.0243E-10	1.5568E-14	F
	(-10,-10,-10)	2.7608E-09	1.4565E-10	4.0919E-10	5.4506E-15	F
	(-100,-100,-100)	2.1687E-09	3.9620E-11	4.7693E-10	1.3879E-14	F
5x5	(0,..,0)	1.9933E-08	2.9626E-09	4.5414E-10	1.5384E-14	1.5384E-14
	(10,..,10)	2.0167E-08	2.9540E-09	4.3824E-10	1.0228E-13	F
	(100,..,100)	2.0440E-08	2.8419E-09	4.5012E-10	1.1855E-13	F
	(-10,,..,-10)	2.0444E-08	2.8361E-09	4.5672E-10	3.0916E-09	F
	(-100,..,-100)	1.9800E-08	3.0916E-09	4.5989E-10	4.5358E-14	F

10x10	(0,..,0)	2.9464E-09	2.3906E-10	4.7908E-10	8.8998E-15	8.8998E-15
	(10,..,10)	3.1894E-09	1.7464E-10	4.7918E-10	1.5973E-15	F
	(100,..,100)	2.7243E-09	2.1669E-10	4.8020E-10	1.4233E-14	F
	(-10,,..,-10)	2.8818E-09	1.5589E-10	4.3769E-10	8.9928E-15	F
	(-100,..,-100)	2.6968E-09	2.1506E-10	4.7197E-10	1.3591E-14	F
20x20	(0,..,0)	2.9707E-09	2.2483E-10	4.6021E-10	6.9335E-16	6.9335E-16
	(10,..,10)	2.7086E-09	2.0588E-10	4.3893E-10	4.5602E-15	F
	(100,..,100)	3.2268E-09	8.2483E-11	4.0955E-10	8.0627E-14	F
	(-10,,..,-10)	2.8833E-09	2.0462E-10	4.3618E-10	3.8011E-15	F
	(-100,..,-100)	3.3295E-09	8.2440E-11	4.0931E-10	6.1506E-14	F
TOTAL		1.5119E-08	8.0025E-10	2.1542E-09	1.5119E-13	6.9335E-16

Based on Table 2 and Table 3, CG FR has the smallest errors compare to other methods. Additionally, PCG method have similar error values with CG FR for initial points of (0,0). For SOR method, the $\omega = 1.25$ possessed the smallest errors compared to other value of extrapolation factor.

The numerical results of error for non-symmetric positive definite (NSPD) matrix and non-symmetric negative definite (NSND) matrix are shown in Table 4 and Table 5.

Table 4 Numerical results of error for NSPD

MATRIX	INITIAL POINT	SOR 1	SOR 2	SOR 3	CG	PCG
2x2	(0,0)	1.4936E-09	9.8669E-11	3.9007E-10	F	F
	(10,10)	1.4936E-09	8.8377E-11	4.3939E-10	F	F
	(100,100)	1.4936E-09	4.1678E-11	6.0994E-10	F	F
	(-10,-10)	1.4936E-09	9.7252E-11	4.4738E-10	F	F
	(-100,-100)	1.4936E-09	4.2548E-11	6.3104E-10	F	F
3x3	(0,0,0)	4.0746E-09	5.1627E-10	8.8059E-10	F	F
	(10,10,10)	3.8163E-09	1.8833E-10	4.6485E-10	F	F
	(100,100,100)	4.2922E-09	6.0455E-11	3.8182E-10	F	F
	(-10,-10,-10)	3.7473E-09	1.8494E-10	5.0155E-10	F	F
	(-100,-100,-100)	4.2758E-09	5.9983E-11	3.8450E-10	F	F
5x5	(0,..,0)	1.8234E-09	3.6766E-10	7.0221E-10	F	F
	(10,..,10)	2.0800E-09	4.0141E-10	8.4437E-10	F	F
	(100,..,100)	1.9109E-09	5.0083E-10	7.3925E-10	F	F
	(-10,,..,-10)	2.0602E-09	4.1224E-10	8.7374E-10	F	F
	(-100,..,-100)	1.9087E-09	5.0216E-10	7.4184E-10	F	F
TOTAL		9.7831E-09	2.1843E-09	3.9014E-09	0.00E+00	0.00E+00

Table 5 Numerical results of error for NSND

MATRIX	INITIAL POINT	SOR 1	SOR 2	SOR 3	CG	PCG
2x2	(0,0)	1.6596E-09	2.0130E-10	5.1517E-10	F	F
	(10,10)	1.6391E-09	3.9167E-11	5.5630E-10	F	F
	(100, 100)	1.7700E-09	1.3459E-10	3.7940E-10	F	F
	(-10,-10)	1.4752E-09	1.2988E-10	4.4512E-10	F	F
	(-100,-100)	1.6836E-09	1.2821E-10	3.6070E-10	F	F
3x3	(0,0,0)	4.4369E-09	5.7482E-10	5.0125E-10	F	F
	(10,10,10)	3.7423E-09	1.3617E-10	7.9696E-10	F	F
	(100,100,100)	4.0291E-09	8.8626E-11	4.3437E-10	F	F
	(-10,-10,-10)	4.2534E-09	1.3327E-10	4.2671E-10	F	F
	(-100,-100,-100)	3.7787E-09	8.5723E-11	4.1570E-10	F	F
5x5	(0,..,0)	2.3168E-08	4.4296E-09	F	F	F
	(10,..,10)	2.3574E-08	4.8688E-09	F	F	F
	(100,..,100)	2.3723E-08	4.4886E-09	F	F	F
	(-10,..,-10)	2.3427E-08	4.4553E-09	F	F	F
	(-100,..,-100)	2.3015E-08	4.7767E-09	F	F	F
TOTAL		1.1691E-07	2.3019E-08	0.0000E+00	0.0000E+00	0.0000E+00

Referring to Table 4 and Table 5, CG FR and PCG method does not able to solve the given problems. Among SOR method, SOR : $\omega = 0.5$ have the biggest error compared to other values of extrapolation factor. In contrast, SOR : $\omega = 1.25$ have the smallest error for some of the initial point similar to SOR : $\omega = 1.9$.

10.2 Numerical Results Based on Iteration Numbers

This section provides summarization of the numerical results focusing on the comparison of iteration numbers. The results for positive definite and negative definite with tolerance factor of 10^{-9} are combined together and compared based on number of iteration. The comparison is shown by considering numerical results of tolerance factor of 10^{-9} only. This is because 10^{-9} is the best stopping criteria which gives small number of errors. The performance profile of all methods based on the number of iterations for symmetric matrix is shown in Figure 1.

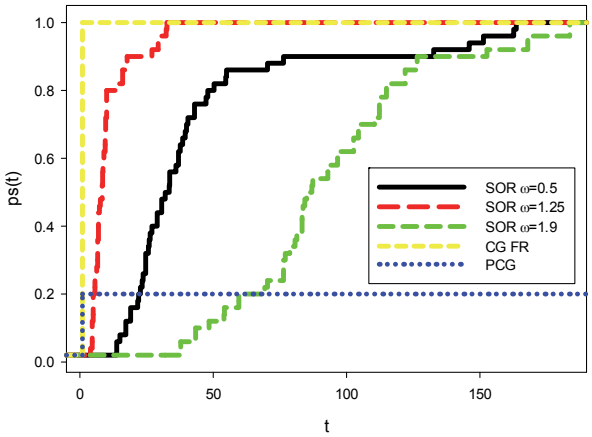

Figure 1. Performance profile of iteration number for symmetric matrix

Referring to Figure 1, from the right side (t >150), it shown that all methods except PCG method are capable on solving system of linear equation by reaching $p_s(t) = 1$. Furthermore, the left side (t < 50) revealed that the best method in term of number of iteration is CG FR method. According to the rule, the performance profile on the top left is the best method compared to the others solver used. The worst method is SOR with $\omega = 1.9$, SOR with $\omega = 0.5$, SOR with $\omega = 1.25$ and PCG which does not successfully converge. This shows that CG FR method has the best performance in number of iterations. Among SOR method, extrapolation factor of $\omega = 1.25$ is the best compared to other value.

The performance profile of all methods based on the number of iterations for non-symmetric matrix is shown in Figure 2.

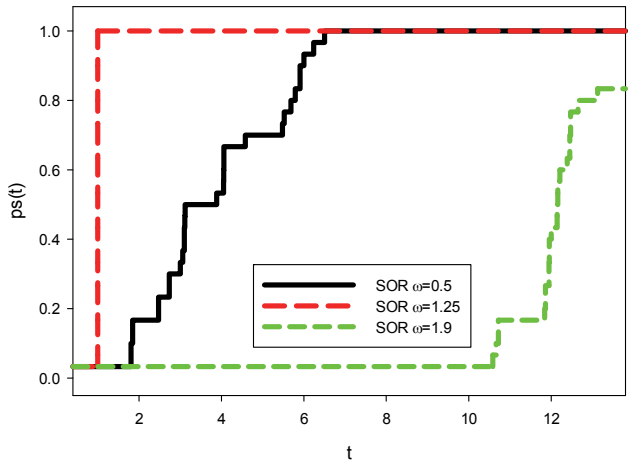

Figure 2. Performance profile of iteration number for non-symmetric matrix

Referring to Figure 2, from the right side (t>12) shown that SOR methods with $\omega = 0.5$ and SOR with $\omega = 1.25$ are capable on solving system of linear equation by reaching $p_s(t) = 1$. Oppositely, SOR with $\omega = 1.9$ is unable to achieve $p_s(t) = 1$. For non-symmetric matrix, CG FR and PCG method failed to solve the system of linear equation. Furthermore, the left side revealed that the best method in term of number of iteration is SOR with $\omega = 1.25$. According to the rule, the performance profile on the top left is the best method compared to the other solver. The worst method is SOR with $\omega = 1.9$ and SOR with $\omega = 0.5$, while CG FR and PCG failed to solve all the given problems. This show that SOR : $\omega = 1.25$ has the best performance in number of iterations.

10.3 Numerical Results Based on CPU times

This section provides summarization of the numerical results focuses on the comparison of CPU time. The results for positive definite and negative definite with tolerance factor of 10^{-9} are combined together and compared based on CPU times. The comparison is shown by considering numerical results of tolerance factor of 10^{-9} only. This is because, 10^{-9} is the best stopping criteria which give small number of errors. The performance profile of all methods based on the CPU times for symmetric matrix is shown in Figure 3.

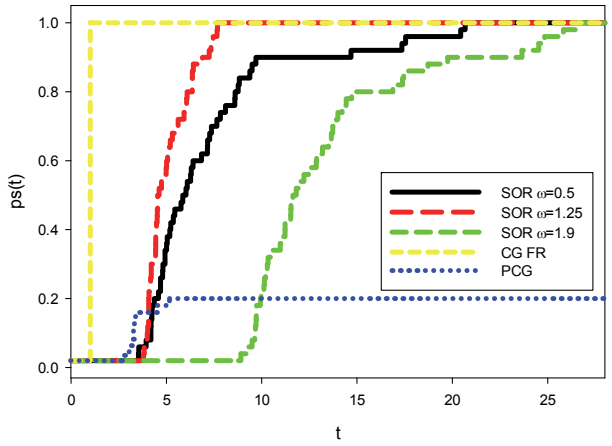

Figure 3. Performance profile of CPU times for symmetric matrix

Referring to Figure 3, from the right side, it shown that all methods except PCG method are capable on solving system of linear equation by reaching $p_s(t) = 1$. Furthermore, the left side revealed that the best method in term of CPU time is CG FR method. According to the rule, the performance profile on the top left is the best method compared to the others solver used. In the opposite, the worst method is SOR : $\omega = 1.9$, SOR : $\omega = 0.5$, SOR : $\omega = 1.25$ and PCG which does not successfully converge. This shows that CG FR method has the best performance in CPU time. Among the SOR method, extrapolation factor of $\omega = 1.25$ is the best compared to other value.

The performance profile of all methods based on the CPU times for non-symmetric matrix is shown in Figure 4.

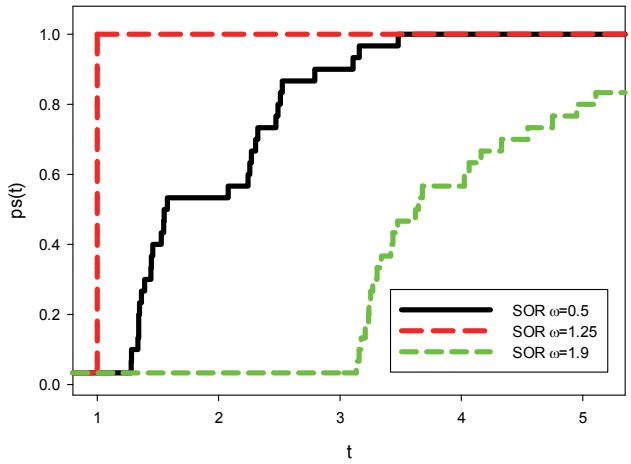

Figure 4. Performance profile of CPU times for non-symmetric matrix

Referring to Figure 4, from the right side, it shown that SOR methods with $\omega = 0.5$ and SOR : $\omega = 1.25$ are capable on solving system of linear equation by reaching $p_s(t) = 1$. However, SOR : $\omega = 1.9$ is unable to achieve $p_s(t) = 1$. For non-symmetric matrix, CG FR and PCG method failed to solve the system of linear equation. Furthermore, the left side revealed that the best method in term of number of iteration is SOR with $\omega = 1.25$. According to the rule, the performance profile on the top left is the best method compared to the others solver. The worst method is SOR with $\omega = 1.9$, SOR with $\omega = 0.5$, while CG FR and PCG failed to solve all the given problems. This show that SOR with $\omega = 1.25$ has the best performance in number of iterations.

11. DISCUSSION

Findings from this study show that the CPU time depends on the number of iterations. Methods with lower number of iterations take less CPU times to converge to the solution points. Therefore, CG FR is the best method that suites with both criteria for symmetric matrix. In contrast, for non-symmetric matrix, CG FR and PCG failed for all initial points. Researcher like Kaasschieter (1988) stated that PCG method can be used for solving system of linear equation of symmetric matrix only. Besides, Omolehin et al. (2008) stated that CG method relies on SPD matrix. Some of the techniques for CG method frequently failed for non-symmetric matrix as mentioned by Saad (1988). Eisenstat et al. (1983) have identified a class of CG like Descent method as a suitable method for solving non-symmetric matrix. Moreover, Musschoot (1999) and Concus & Golub (1976) proposed Lanczos' method and Generalized Conjugate Gradient (GCG) respectively for non-symmetric systems. There are some other methods that can be used to solve non-symmetric systems of linear equations. Therefore, based on the supporting fact and the numerical result, it is concluded that system of linear equation of non-symmetric matrix cannot be solved by using CG FR and PCG method. The SOR method with $\omega = 1.25$ is best method which suite with both criteria for non-symmetric matrix.

12. CONCLUSION

System of linear equations is solved using three iterative methods which are SOR, CG FR and PCG. For SOR method, the extrapolation factor are varies in order to identify the best value in finding the solution points. The PCG method is also used since there are some cases where matrix A in system of linear equation has small eigenvalues which might be approaching to zero. This system is called as ill-conditioned and the condition can be determined based on the condition number that can be calculated based on its eigenvalues. In this research, the calculation based on PCG is calculated without considering its condition number. Therefore, all methods are used to solve large and small system of linear equation.

In this research, the efficiency of CG FR, PCG, SOR with $\omega = 0.5$, SOR with $\omega = 1.25$ and SOR with $\omega = 1.9$ are analyzed based on number of iterations and CPU time. From the previous section on result analysis, all methods except PCG method succeed in solving symmetric matrix while the SOR method only succeed in solving non-symmetric matrix. Based on performance profile (Figure 1), the CG FR possess the lowest number of iterations for symmetric matrix while referring to Figure 2, SOR method with $\omega = 1.25$ possess the lowest number of iterations for non-symmetric matrix. Based on performance profile (refer Figure 3) the CG FR possess the lowest CPU time for symmetric matrix while (refer Figure 4) SOR method with $\omega = 1.25$ possess the lowest CPU time for non-symmetric matrix. In other word, the lower the number of iteration, the faster the CPU time.

The tolerance factor in this research consists of three different value which are 10^{-3}, 10^{-6} and 10^{-9}. Calculation for 10^{-9} as stopping criteria have the smallest error compare to others value. This are varied in order to determine the accuracy of approximate points. Besides, errors of each method are calculated using absolute error formula. The CG FR method have the smallest error compare to others method for symmetric matrix and SOR method with $\omega = 1.25$ have the smallest error compare to others method for non-symmetric matrix.
In conclusion, CG FR is the best method for solving system with symmetric matrix and SOR method with $\omega = 1.25$ is the best method for solving system with non-symmetric matrix. This can be referred to the number of iteration, CPU time and error calculation.

13. FUTURE WORK

In this research, the PCG method is tested without considering its condition number. The ill-conditioned system can be determined based on the high condition number. Thus, the PCG can be used to overcome the ill-conditioned system and improves the rate of convergence. Moreover, it is suggested to compare the matrix with higher dimension and applied in real world mathematical cases.

14. ACKNOWLEDGEMENTS

This work is reviewed and examined by a lecturer from MARA University of Technology, Siti Musliha Nor-Al-Din. Authors would like to thank lecturers and a panel, Noor Erni Fazliana Mohd Akhir that involved in Final Year Project Viva-Voce 2017.

REFERENCES

[1] S. Gharib, S.R. Ali, R. Khan, N. Munir, and M. Khanam, Global Journal of Computer Science and Technology 15, (2015).
[2] S.J. Leon, *Linear algebra with applications*, 8th ed. (Pearson Prentice Hall, Upper Saddle River, New Jersey, 2010).
[3] K.D. Shastri, R. Biswas, and P. Kumari, International Journal for Scientific Research & Development **3**, 3287 (2015).
[4] G. Konghua, X. Hu, and L. Zhang, Applied Mathematics and Computation **187**, 1434 (2007).
[5] M.F. Kacamarga, B. Pardamean, and J. Baurley, Applied Mathematical Sciences **8**, 837 (2014).
[6] N. Jamil, International Journal of Emerging Sciences **2**, 310 (2012).
[7] I. Kalambi, Journal of Applied Sciences and Environmental Management **12**, 53 (2008).
[8] M.J. Rodríguez-Alvarez, F. Sánchez, A. Soriano, and A. Iborra, Mathematical and Computer Modelling **52**, 1258 (2010).
[9] H.K. Liu, Applied Mathematics and Computation **217**, 5259 (2011).
[10] N.A. Atasoy, B. Sen, and B. Selcuk, Procedia Technology **1**, 31 (2012).
[11] A. Smoktunowicz and A. Smoktunowicz, Applied Numerical Mathematics **67**, 220 (2013).
[12] L. Fuyong, Applied Mathematics and Computation **232**, 1269 (2014).
[13] I. Dimov, S. Maire, and J.M. Sellier, Applied Mathematical Modelling **39**, 4494 (2015).
[14] C.C. Douglas, L. Lee, and M.C. Yeung, Procedia Computer Science **80**, 941 (2016).
[15] M.C. Yeung, C.C. Douglas, and L. Lee, Journal of Computational Science **20**, 177 (2017).
[16] S. Sickel, M.C. Yeung, and J. Held, in *Summer Research Apprentice Program Symposium* (University of Wyoming, Laramie , Wyoming, 2005).
[17] T.P. Collignon, "Efficient Iterative Solution of Large Linear Systems on Heterogeneous Computing Systems," Ph. D. thesis, Delft University of Technology, 2011
[18] Y. Saad, *Iterative methods for sparse linear systems*, 2nd ed. (Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 2003).
[19] Y. Saad and H.A. Van Der Vorst, Journal of Computational and Applied Mathematics **123**, 1 (2000).
[20] D.H. Li, Y.Y. Nie, J.P. Zeng, and Q.N. Li, Mathematical and Computer Modelling **48**, 918 (2008).
[21] R.L. Burden and J.D. Faires, *Numerical analysis* (Cengage learning, Boston, 2011).
[22] I. Arany, Computers & Mathematics with Applications **38**, 125 (1999).
[23] S. Mittal, International Journal of High Performance Computing and Networking **7**, 292 (2014).
[24] S. Chen, International Journal of Numerical Analysis and Modeling **11**, 117 (2014).
[25] R. Barrett et al., *Templates for the solution of linear systems: building blocks for iterative*

methods (Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania, 1994).
[26] D.A. Gismalla, International Journal of Engineering and Technical Research (IJETR) **2**, 7 (2014).
[27] M. Eiermann and R.S. Varga, Linear Algebra and its Applications **182**, 257 (1993).
[28] W.Y. Yang, W. Cao, T.S. Chung, and J. Morris, *Applied numerical methods using MATLAB*, 1st ed. (John Wiley & Sons, Hoboken, New Jersey, 2005).
[29] J. Modersitzki (1994). *Conjugate gradient type methods for solving symmetric, indefinite linear systems.* Report no. 868, Department of Mathe,atics, University of Urecht.
[30] V.S. Ryabenkii and S.V. Tsynkov, *A theoretical introduction to numerical analysis* (Chapman & Hall/CRC, Boca Raton, Florida, 2006).
[31] D.P. O'leary, Computing **22**, 59 (1979).
[32] E.W. Weisstein, *CRC concise encyclopedia of mathematics*, 2nd ed. (Chapman & Hall/CRC, Boca Raton, Florida, 2002).
[33] M.R. Hestenes and E. Stiefel, Journal of Research of the National Bureau of Standards **49**, 409 (1952).
[34] Z.Z. Bai and Z.Q. Wang, Journal of Computational and Applied Mathematics **187**, 202 (2006).
[35] M. Benhamadou, Applied Mathematics and Computation **189**, 927 (2007).
[36] E. Kreyszig, *Advanced Engineering Mathematics*, 10th ed. (John Wiley & Sons, 2010).
[37] M. Benzi, Journal of Computational Physics **182**, 418 (2002).
[38] J.B. Aparecido, N.Z. Souza and J.B. Campos-Silva, "Conjugate Gradient Method for Solving Large Sparse Linear Systems on Multi-Core Processors," in *10th World Congress on Computational Mechanics* (São Paulo, 2014)
[39] I.C.F. Ipsen, *Numerical matrix analysis: linear systems and least squares* (Society for Industrial and Applied Mathematics, Philadelphia, 2009).
[40] N. Abu Bakar, "Solving Large System of Linear Equation Using Iterative Methods (Successive Over-Relaxation, Conjugate Gradient and Preconditioned Conjugate Gradient)," B.Sc. thesis, MARA University of Technology, 2017.
[41] E.F. Kaasschieter, Journal of Computational and Applied Mathematics **24**, 265 (1988).
[42] J.O. Omolehin, M.K.A. Abdulrahman, K. Rauf, and P.J. Udoh, African Journal of Mathematics and Computer Science Research **1**, 28 (2008).
[43] Y. Saad, Journal of Computational and Applied Mathematics **24**, 89 (1988).
[44] S.C. Eisenstat, H.C. Elman, and M.H. Schultz, SIAM Journal on Numerical Analysis **20**, 345 (1983).
[45] C. Musschoot, Journal of Computational and Applied Mathematics **101**, 61 (1999).
[46] P. Concus and G.H. Golub, in *Computing Methods in Applied Sciences and Engineering*, 1st ed. (Springer-Verlag Berlin Heidelberg, 1976), pp. 56–65.

APPENDIX

If any, the appendix should appear directly after the references without numbering, and on a new page.