A Survey of Krylov Subspace Methods Solving Non-Symmetric Systems of Linear Equations

Phuong Dong Le

Abstract

Numerical linear algebra is a discipline that powers various algorithms and applications to data science. One of the core problems in this field is to concern with finding the solution to the linear system $Ax = b$, where $A \in \mathbb{R}^{n\times n}$ is a non-singular and non-symmetric matrix and $b \in \mathbb{R}^n$. In this report, we present generalized minimal residual (GMRES), biconjugate gradient (Bi-CG) and induced dimension reduction (IDR(s)) methods which are in the family of Krylov-subspace to solve the large systems of linear equations. We would describe each algorithm and the theory. In addition, we compare the performance benchmarks to illustrate with problems arising from mathematical model.

Keywords: Krylov-subspace methods, GMRES, biconjugate gradient (Bi-CG), induced dimension reduction (IDR(s)), iterative methods.

1 Introduction

Solving a large system of linear equations is a fundamental problem in numerical linear algebra. There have been different approaches such as splitting matrices methods, conjugate gradient for positive-symmetric definite matrix etc. Those methods require the matrix of system of linear equations to satisfy sufficient condition to converge; therefore, iterative methods have been developed to solve the problem for better efficiency [1, 2, 3].

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

where $A \in \mathbb{R}^{n\times n}$ is non-symmetric, $b \in \mathbb{R}^n$ and $x \in \mathbb{R}^n$ is unknown.

The Krylov subspace methods are developed to seek for approximations to the solution (1), which is known as the Krylov-subspace:

$$\mathcal{K}_k(A; r_0) = span\{r_0, Ar_0, A^2r_0, \ldots, A^{k-1}r_0\} \tag{2}$$

where k is the number of iterations, $r_0 = b - Ax_0$ is defined to be the initial residual, and x_0 is some initial guess to the solution.

There exists different Krylov subspace methods. One of the most well-known method is generalized minimal residual (GMRES) proposed by Saad and Schultz [3]. However, there are drawbacks in the cost of memory and computations to store the new orthogonal basis vectors at each iteration in GMRES [1, 2]. In the search for technical efficient and approach of Krylov subspace methods, Biconjugate gradient and induced dimension reduction have been developed. The biconjugate gradient method (BiCG) takes the approach of relaxing the idea of orthogonality by constructing two sets that are biorthogonal which are bases for $\mathcal{K}_k(A; r_0)$ and $\mathcal{K}_k(A^T; \hat{r}_0)$ respectively. Meanwhile, induced dimension reduction (IDR(s)) generates the residuals which are in the subspace of decreasing dimension [4]. The report is organized as follows. In the section 2, we shall introduce three algorithms in the family of Krylov subspace methods and a brief discussion of computational cost. In the section 3, we describe the numerical experiments and apply the methods to test the performance. We present both simple and complex experiments to solve linear system (1). Although image-processing experiment can be found at `https://github.com/phuongle0701/CS675_Project`, we shall not include the result on this report. In the section 4, we make some concluding remarks on Krylov subspace methods.

2 Krylov Subspace Methods

2.1 Generalized minimal residual

GMRES is one of Krylov methods that attempts to minimal norm residual approach [1, 2]. In general, at the iteration step k, the algorithm identifies the approximate solution x_k in which the norm of the residual $\|b - Ax_k\|_2$ is minimal over the Krylov subspace (2). GMRES computes an orthonormal basis $\{q_0, ..., q_k\}$ for $\mathcal{K}_{k+1}(A; r_0)$:

$$Q_{k+1} = \left[q_0 \middle| q_1 \middle| \dots \middle| q_k \right] \tag{3}$$

where the matrix Q_{k+1} satisfies $Q_{k+1}^T Q_{k+1} = I_{k+1}$.

We generate the orthonormal basis for the Krylov subspace by first setting:

$$q_0 = \frac{r_0}{\|r_0\|_2}$$

The modified Gram-Schmidt algorithm is applied to orthogonalize the vectors [1, 2, 3]:

$$\{q_0, Aq_0, ..., Aq_{k-1}\} \tag{4}$$

By Gram-Schmidt, we generate the new vector v_{k+1} such that:

$$v_{k+1} = Aq_k - h_{0,k} - h_{1,k}q_1 - \ldots - h_{k,k}q_k \tag{5}$$

where the projection coefficients $h_{i,k}$ are determined by the relation $h_{i,k} = q_i^T A q_k$.

The new orthonormal vector q_{k+1} is determined by normalizing the vector v_{k+1} such that $q_{k+1} = \frac{v_{k+1}}{h_{k+1,k}}$ where $h_{k+1,k} = \|v_{k+1}\|_2$.

It leads to the basis vectors in (4) satisfy the relation:

$$h_{k+1,k}q_{k+1} = Aq_k - h_{0,k}q_0 - h_{1,k}q_1 - \ldots - h_{k,k}q_k \tag{6}$$

The process to generate the orthonormal basis for the Krylov subspace is known as the *Arnold* procedure [2]. The equation (6) can be written in matrix form as:

$$AQ_{k+1} = Q_{k+2}\tilde{H}_{k+1} \tag{7}$$

where $Q_{k+1} \in \mathbb{R}^{n\times(k+1)}, Q_{k+2} \in \mathbb{R}^{n\times(k+2)}$, and $\tilde{H}_{k+1} \in \mathbb{R}^{(k+2)\times(k+1)}$ is an upper Hessenberg matrix of the following form [1, 2]:

$$\tilde{H}_{k+1} = \begin{bmatrix} h_{0,0} & h_{0,1} & h_{0,2} & \ldots\ldots & h_{0,k} \\ h_{1,0} & h_{1,1} & h_{1,2} & \ldots\ldots & \vdots \\ & h_{2,1} & h_{2,2} & \vdots\vdots & \vdots \\ & & \ddots & \ddots\ddots & \vdots \\ & & & h_{k,k-1} & h_{k,k} \\ & & & & h_{k+1,k} \end{bmatrix} \tag{8}$$

Since the columns of Q_{k+1} form an orthonormal basis for the Krylov subspace $\mathcal{K}_k(A; r_0)$, the algorithm chooses the optimal $y \in \mathbb{R}^{k+1}$ to minimise $\|r_{k+1}\|_2$ over the space:

$$x_{k+1} = x_0 + Q_{k+1}y \tag{9}$$

We use the relation from Arnoldi procedure (7), we can write:

$$\|r_{k+1}\|_2 = \|r_0 - AQ_{k+1}y\|_2 = \left\| r_0 - Q_{k+2}\tilde{H}_{k+1}y \right\|_2 \tag{10}$$

Since $q_0 = \frac{r_0}{\|r_0\|_2}$ forms the first column of Q_{k+2}, we can write $r_0 = \|r_0\|_2 Q_{k+2} e_1$ where $e_1 = (1, 0, 0, \ldots, 0)^T \in \mathbb{R}^{k+2}$ is the canonical basis. Therefore, we get the following relation:

$$\begin{aligned}
\|r_{k+1}\|_2^2 &= \Big\| Q_{k+2}\Big(\|r_0\|_2 e_1 - \tilde{H}_{k+1} y\Big)\Big\|_2^2 \\
&= \bigg(Q_{k+2}\Big(\|r_0\|_2 e_1 - \tilde{H}_{k+1} y\Big)\bigg)^T \bigg(Q_{k+2}\Big(\|r_0\|_2 e_1 - \tilde{H}_{k+1} y\Big)\bigg) \\
&= \Big(\|r_0\|_2 e_1 - \tilde{H}_{k+1} y\Big)^T Q_{k+2}^T Q_{k+2}\Big(\|r_0\|_2 e_1 - \tilde{H}_{k+1} y\Big) \\
&= \Big\| \Big(\|r_0\|_2 e_1 - \tilde{H}_{k+1} y\Big)\Big\|_2^2
\end{aligned} \tag{11}$$

Thus, the relation (11) leads to solve a least-squares problem with the Hessenberg matrix $\tilde{H}_{k+1} \in \mathbb{R}^{(k+1)\times(k+2)}$ (8) to obtain the vector y. Shortly, pseudocode of GMRES algorithm is summarized in appendix A.

2.2 Biconjugate gradient

Biconjugate gradient takes the approach of constructing two biorthogonal sets $\{q_1, ..., q_k\}$ and $\{\hat{q}_1, ..., \hat{q}_k\}$ which are respectively the bases to:

$$\mathcal{K}_k(A; r_0) = span\Big\{r_0, Ar_0, \ldots, A^{k-1} r_0\Big\} \tag{12}$$

$$\mathcal{K}_k(A^T; \hat{r}_0) = span\Big\{\hat{r}_0, A^T \hat{r}_0, \ldots, (A^T)^{k-1} \hat{r}_0\Big\} \tag{13}$$

where the first Krylov subspace is for the linear system (1) and the second Krylov subspace is for the dual system $A^T \hat{x} = \hat{b}$.

To construct the biorthogonal basis vectors, we apply the unsymmetric Lanczos procedure with $q_1 = \frac{r_0}{\|r_0\|_2}$ such that $r_0^T \hat{r}_0 \neq 0$ [1, 2]:

$$AQ_k = Q_k T_k + r_k e_k^T, \quad \hat{Q}_k^T r_k = 0 \tag{14}$$

$$A^T \hat{Q}_k = \hat{Q}_k T_k^T + \hat{r}_k e_k^T, \quad Q_k^T \hat{r}_k = 0 \tag{15}$$

where $Q_k = \Big[q_1 \Big| q_2 \Big| \ldots \Big| q_k\Big]$ and $Q_k^{-T} = \hat{Q}_k = \Big[\hat{q}_1 \Big| \hat{q}_2 \Big| \ldots \Big| \hat{q}_k\Big]$ such that $\hat{Q}_k^T Q_k = I_k$.

The unsymmetric Lanczos algorithm decomposes the matrix A to a tridiagonal form using a similarity transformation such that [1]:

$$Q_k^{-1} A Q_k = T_k \tag{16}$$

where $T_k \in \mathbb{R}^{k\times k}$ is a tridiagonal matrix such that:

$$T_k = \begin{bmatrix}
\alpha_1 & \gamma_1 & & & \\
\beta_1 & \alpha_2 & \gamma_2 & & \\
& \ddots & \ddots & \ddots & \\
& & \ddots & \ddots & \gamma_{k-1} \\
& & & \beta_{k-1} & \alpha_k
\end{bmatrix} \tag{17}$$

Similarly, since biconjugate gradient is one of Krylov subspace methods, we seek the optimal $y \in \mathbb{R}^k$ in (9). We use the relation in (14) to derive as follows:

$$\hat{Q}_k^T\Big(b - Ax_k\Big) = \hat{Q}_k^T\Big(b - A(x_0 + Q_k y)\Big) = \hat{Q}_k^T\Big(b - Ax_0 - AQ_k y\Big) = \hat{Q}_k^T r_0 - T_k y = 0$$

Therefore, the relation $T_k y = \hat{Q}_k r_0$ leads to solving a tridiagonal system to obtain the optimal vector y in the iterate formula (9). In summary, the pseudocode for BiCG (biconjugate gradient) algorithm is outlined in appendix B.

2.3 Induced dimension reduction

IDR(s) method is also a family of Krylov subspace method with short-recurrence for solving the nonsymmetric systems of linear equations (1) [4]. The algorithm introduces a completely different approach for solving the linear system compared to other Krylov subspace methods. IDR(s) method is based on the following theorem [4].

Theorem 1. *Let A be any matrix in $\mathbb{R}^{n\times n}$, let $u_0 \in \mathbb{R}^n$ be any nonzero vector, and let $\mathcal{G}_0$ be the full Krylov subspace $\mathcal{K}^n(A; u_0)$. Let S be the proper subspace of $\mathbb{R}^n$ such that S and $\mathcal{G}_0$ do not share a non-trivial invariant subspace of A, and define the sequence $\mathcal{G}_i$ for $i = 1, 2, ...$ as follows:*

$$\mathcal{G}_i = (I - \omega_i A)(\mathcal{G}_{i-1} \cap S) \tag{18}$$

where ω_i's are non-zero scalars. Then the following hold:

1. $\mathcal{G}_i \subset \mathcal{G}_{i-1} \quad \forall i > 0.$
2. $\mathcal{G}_i = \{\mathbf{0}\}$ *for some* $i \leq N$.

In the theorem (1), the first condition implies that it is possible to generate the residuals that are in the nested subspaces $\mathcal{G}_i$ where the smallest possible subspace is $\{\mathbf{0}\}$ in the second condition. We shall apply this theorem to solve the problem by dimension reduction. We know that the residual $r_k \in \mathcal{K}_k(A; r_0)$ can be expressed as a polynomial of the form [2, 4]:

$$r_k = \phi_k(A) r_0 \tag{19}$$

where $\phi_k(x)$ is a polynomial of degree k.

Suppose we calculate the residuals and approximate solution up to k-th step, then we can have the following relation to calculate approximate solution at $(k+1)$-th step:

$$A(x_{k+1} - x_k) = -(r_{k+1} - r_k) = \Big[\phi_k(A) - \phi_{k+1}(A)\Big] r_0$$

We calculate the solution at $(k+1)$-th step if the polynomial difference above is divisible i.e $\phi_{k+1}(v) = \phi_k(v) + v p_k(v)$ for $v \in \mathbb{P}^k \setminus \mathbb{P}^{k-1}$ for some vector v. Thus, the general Krylov iterative solver is as follows [4]:

$$x_{k+1} = x_k + \alpha v_k - \sum_{i=1}^{\hat{l}} \gamma_i \Delta x_{k-i} \tag{20}$$

$$r_{k+1} = r_k - \alpha A v_k - \sum_{i=1}^{\hat{l}} \gamma_i \Delta r_{n-i} \tag{21}$$

where $v_k \in \mathcal{K}_k(A; r_0) \setminus \mathcal{K}_{k-1}(A; r_0)$, Δx_k is the forward difference operator i.e $\Delta x_k = x_{k+1} - x_k$, and $\hat{l}$ is the depth of recursion which is a small integer compared to N.

By theorem (1), we know that the residual r_{k+1} will be in the subspace $\mathcal{G}_{i+1}$ [4]:

$$r_{k+1} = (I - \omega_{i+1}A)v_k$$

where $v_k \in \mathcal{G}_i \cap S$.

Using this restriction on v_k, we can choose

$$v_k = r_k - \sum_{i=1}^{s} \gamma_i \Delta r_{k-i}$$

We can express the general Krylov iterative formulas (20) (21) in the following:

$$x_{k+1} = x_k + \omega_{i+1}v_k - \sum_{i=1}^{s} \gamma_i \Delta x_{k-i} \tag{22}$$

$$r_{k+1} = r_k - \omega_{i+1}v_k - \sum_{i=1}^{s} \gamma_i \Delta r_{k-i} \tag{23}$$

We see that s defines the depth of recursion which is a positive integer chosen to be smaller than N. According to P. Sonneveld and M. Gijzen, the choice of $s<<N$ which is usually ranging from 1 to 16 defines the short-recurrence algorithm in terms of memory efficient and computational cost [4]. In short, the pseudocode for induced dimension reduction algorithm is summarized in Appendix C.

2.4 Computational cost

One of important results in GMRES method is that it solves the system at most n iterations [2, 3]. However, GMRES method suffers from two problems: computational cost and memory storage. First, computation increases from iteration to iteration since the cost is $\mathcal{O}(k^2)$ at k-th iteration [2]. Also, as k increases, the memory requirement grows. From Arnoldi procedure, we need to store $v_1, ..., v_k \in \mathbb{R}^n$; therefore, it takes $\mathcal{O}(nk)$ space [2]. Although the BiCG may not achieve the convergence in fewer iterations than GMRES, it requires less memory and computations than GMRES. The BiCG method has three-term recurrences with a fixed amount of storage and vector-matrix operations [2]. For symmetric system, BiCG would require twice computational cost of conjugate gradient because of (12) (13) [2]. Lastly, because IDR(s) method requires the choice of s so the computational cost and memory storage should increase for larger s. IDR(s) algorithm in Appendix C can be divided into three parts. First part requires $(s + 1)$ matrix-vector products, and then it needs a $(s^2 + s + 2)$ inner products [4]. The last part requires $2s^2 + \frac{7}{2}s + \frac{5}{2}$ updates [4]. So IDR(s) has the same cost per iteration; moreover, the method would guarantee convergence at most $n + \frac{n}{s}$ iterations [4].

3 Numerical Experiments

In this section, we shall test and compare the benchmark performance of three Krylov subspace methods: GMRES, BiCG and IDR(s). Since the computational cost and memory storage IDR(s) algorithm varies on the choice of integer s, we choose to test the method on $s = \{4, 8, 100\}$ in which the first two are safe choices while we impose $s = 100$ to verify the result.

3.1 Nonsymmetric linear system

We consider testing our methods on the nonsymmetric linear system which is from Example 35.1 of Trefenthen and Bau [5] such that:

$$A_n x = b \tag{24}$$

where $A_n = nI + P \in \mathbb{R}^{n\times n}$ and $P \in \mathbb{R}^{n\times n}$ is a matrix with entries from a random normal distribution with mean 0 and standard deviation $1/(2\sqrt{n})$.

We choose $b = (1, 2, \ldots, n)^T \in \mathbb{R}^n$, $x_0 = \mathbf{0}$ and tolerance to be 10^{-10} as the criterion for relative residual with 1000 maximum number of iterations to set up experiment [5].

Problem Size	GMRES	BiCG	IDR(4)	IDR(8)	IDR(100)
$n = 200$	5	7	6	6	6
$n = 400$	5	6	5	5	5
$n = 800$	5	6	5	5	5
$n = 1600$	4	6	4	4	4

Table 1: The number of converging iterations for each Krylov subspace methods.

Problem Size	GMRES	BiCG	IDR(4)	IDR(8)	IDR(100)
$n = 200$	0.007(sec.)	0.014(sec.)	0.045(sec.)	0.017(sec.)	0.006(sec.)
$n = 400$	0.008(sec.)	0.002(sec.)	0.003(sec.)	0.001(sec.)	0.006(sec.)
$n = 800$	0.011(sec.)	0.004(sec.)	0.005(sec.)	0.003(sec.)	0.009(sec.)
$n = 1600$	0.038(sec.)	0.010(sec.)	0.005(sec.)	0.005(sec.)	0.015(sec.)

Table 2: The elapsed-time of each Krylov subspace method with respect to the problem size of nonsymmetric linear system.

For the nonsymmetric linear system (24), it is suggested that the GMRES and IDR(s) converge faster in terms of number of iterations than BiCG algorithm. However, we see that for GMRES and IDR(s) methods there is a trade-off between run-time and number of iterations. In particular, as the problem size increases, GMRES takes more time to execute than other methods since there is an increase of size vector-matrices in the Arnoldi step. Meanwhile, there is an increase of matrix-products of building $\mathcal{G}_i$ space for IDR(s) algorithm. In the next part, we would test each method on solving Poisson equation in comparison of performance.

3.2 Poisson equation

In this part, we consider the experiment on solving two- and three-dimensional Poisson partial differential equation with zero boundary conditions as follows:

$$-\Big(\sum_{i=1}^{d} \frac{\partial^2 T}{\partial x_i^2}\Big) = f, \quad \text{on } (0,1)^d \tag{25}$$

where d is the dimension of Poisson equation, and f is the source term.

We discretize the Poisson equation (25) using finite difference and with Kronecker product approach [1] so we obtain the linear system form $Tu = f$ where $T \in \mathbb{R}^{N\times N}$ such that $N = m^d$ (the number of grid points) and $u \in [0,1]^d$. We choose tolerance to be 10^{-12} to be the convergence criterion for relative residual, 5000 maximum number of iterations and $u_0 = \mathbf{0}$.

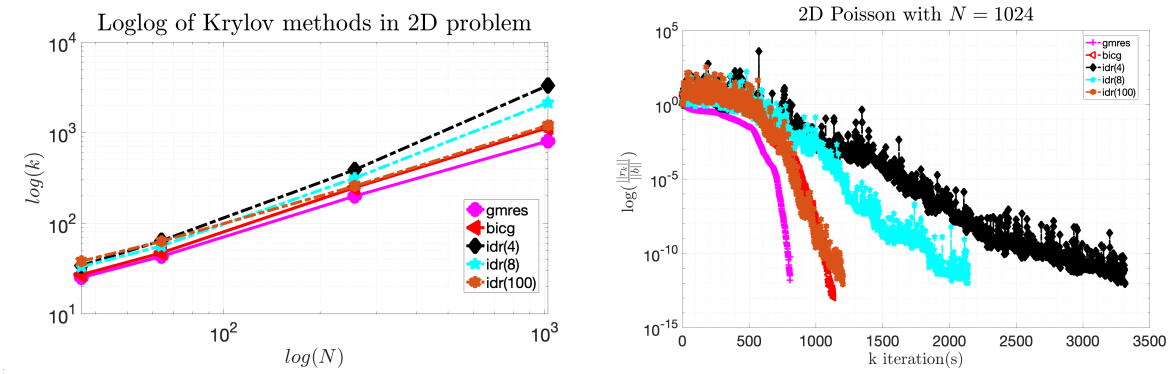
3.2.1 2D Poisson equation

We choose the grid size to be $m = \{6, 8, 16, 32\}$ which corresponds to problem size $N = \{36, 64, 256, 1024\}$. We use the three-source term function as follows:

$$f(x_1, x_2) = \begin{cases} 1 & \|(x_1, x_2) - (0.3, 0.3)\|_2 \leq 0.1 \\ 1 & \|(x_1, x_2) - (0.6, 0.6)\|_2 \leq 0.1 \\ 0 & \text{otherwise} \end{cases}$$

Problem Size	GMRES	BiCG	IDR(4)	IDR(8)	IDR(100)
$N = 36$	0.015(sec.)	0.007(sec.)	0.005(sec.)	0.010(sec.)	0.020(sec.)
$N = 64$	0.030(sec.)	0.005(sec.)	0.003(sec.)	0.014(sec.)	0.017(sec.)
$N = 256$	0.033(sec.)	0.003(sec.)	0.013(sec.)	0.013(sec.)	0.059(sec.)
$N = 1024$	0.7(sec.)	0.021(sec.)	0.088(sec.)	0.094(sec.)	0.645(sec.)

Table 3: The elapsed-time of each Krylov subspace iterative methods on 2D Poisson problem.



(a) (b)

Figure 1: (a). Problem size versus iterations in *log* scale. (b). Convergence in 2D Poisson equation with problem size $N = 1024$.

According to table (4), as the problem size increases BiCG method performs to be the fastest in run time while the runner-up would be IDR(4). Meanwhile, GMRES method slows down as the problem size increases due to the drawback in computational cost and memory storage. However, in the figure (1)(a), GMRES has the least converging iterations. IDR(100) and BiCG have close in the converging iterations, but IDR(4) and IDR(8) have more converging iterations as in the figure (1)(b). Overall, in this experiment, GMRES and IDR(4) should be the best methods in terms of run-time and number of iterations.

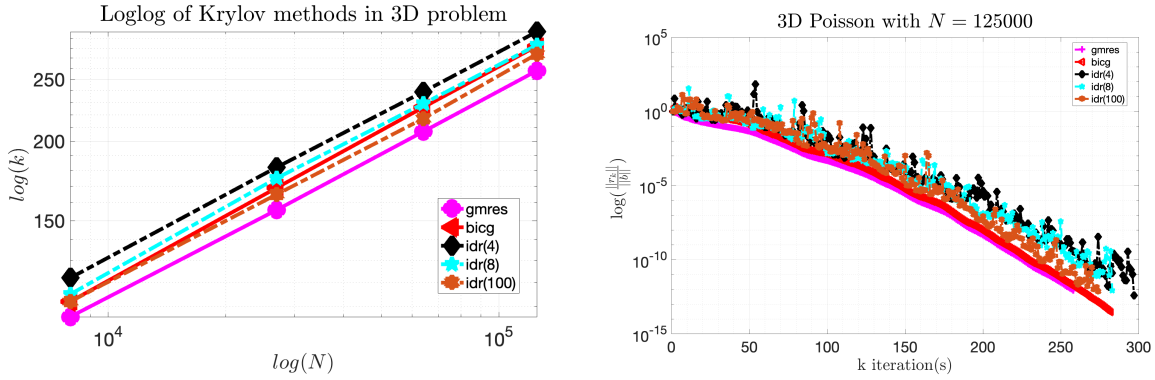
3.2.2 3D Poisson equation

Similarly, we choose the grid sizes to be $m = \{20, 30, 40, 50\}$ corresponding to problem sizes $N = \{8000, 27000, 64000, 125000\}$. We use the same heat-source term as in 2D Poisson equation. The following plots are the results of 3D Poisson equation.

$$f(x_1, x_2, x_3) = \begin{cases} 1 & \|(x_1, x_2, x_3) - (0.3, 0.3, 0.3)\|_2 \leq 0.1 \\ 1 & \|(x_1, x_2, x_3) - (0.6, 0.6, 0.6)\|_2 \leq 0.1 \\ 0 & \text{otherwise} \end{cases}$$

Problem Size	GMRES	BiCG	IDR(4)	IDR(8)	IDR(100)
$N = 8,000$	0.462(sec.)	0.021(sec.)	0.067(sec.)	0.076(sec.)	0.506(sec.)
$N = 27,000$	2.230(sec.)	0.108(sec.)	0.177(sec.)	0.237(sec.)	1.629(sec.)
$N = 64,000$	4.577(sec.)	0.232(sec.)	0.343(sec.)	0.456(sec.)	3.964(sec.)
$N = 125,000$	9.942(sec.)	0.497(sec.)	0.738(sec.)	0.987(sec.)	8.792(sec.)

Table 4: The elapsed-time of each Krylov subspace iterative methods on 3D Poisson problem.



(a) (b)

Figure 2: (a). Problem size versus converging iterations in *log* scale. (b). Convergence in 3D Poisson equation with problem size $N = 125,000$.

For 3D Poisson equation, table (4) suggests that BiCG and IDR(4) are among the fastest methods in run-time. GMRES and IDR(100) are both the slowest. For IDR(100), the reason comes from building more nested space $\mathcal{G}_i$ so it requires larger matrix-products and memory storage. For the number of iterations, GMRES still takes the first place; however, BiCG exceeds IDR(8) and IDR(100) as the problem size increases according to figure (2)(a). For instance, in figure (2)(b), it is clear that IDR(8) and IDR(100) takes less iterations than BiCG method. In short, BiCG and IDR(8) are two methods that could balance the trade-off performance in run-time and number of iterations in this case.

4 Concluding Remarks

In this report, we have investigated Krylov subspace methods which are GMRES, BiCG and IDR(s) when using them to solve systems in the form (1). Through a series of experiments in discretizing a mathematical model in partial differential equation, the results lead us to some concluding remarks.

IDR(s) method is a short-recurrence algorithm that requires an amount of operations and low memory storage per iteration compared with GMRES method. BiCG seems to be the best method in terms of computational time and number of iterations due to its three-recurrence. However, BiCG algorithm could not guarantee the convergence to the solution for the system. However, both GMRES and IDR(s) can guarantee to solve at most n and $n + \frac{n}{s}$ iterations respectively. In these experiments, we have seen that IDR(s) algorithm depends on the choice of s which should be chosen to be small to preserve the low memory storage. The report is not however exhaustive, and more future studies should be conducted to identify stability conditions for IDR(s) and explore the preconditioning technique for Krylov subspace methods. If these are considered, then IDR(s) algorithm should be a better choice to solve the non-symmetric linear system.

References

[1] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.

[2] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, USA, 2nd edition, 2003.

[3] Y. Saad and M. Schultz. Gmres: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *Siam Journal on Scientific and Statistical Computing*, 7:856–869, 1986.

[4] P. Sonneveld and M. Gijzen. Idr(s): A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations. *SIAM J. Scientific Computing*, 31:1035–1062, 01 2008.

[5] L. N. Trefethen and D. Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

5 Appendix

The pseudocodes for GMRES, BiConjugate and IDR(s) algorithms are implemented in MATLAB at `https://github.com/phuongle0701/CS675_Project`.

5.1 Appendix A

Algorithm 1: GMRES algorithm

Input: $A \in \mathbb{R}^{n\times n}$; $b \in \mathbb{R}^n$; $x_0 \in \mathbb{R}^n$ an initial guess of solution

$\mathrm{r}_0 = b - Ax_0$

$\mathrm{q}_0 = r_0/\|r_0\|$

Repeat until convergence criterion is reached.

$v = Aq_k$ `// Do Anordi procedure`

for $i \leftarrow 0$ **to** k **do**

  $h_{i,k} = q_i^T v$

  $v = v - h_{i,k} q_i$

$h_{k+1,k} = \|v\|$

$q_{k+1} = \frac{v}{h_{k+1,k}}$

We find y to minimize $\left\|\|r_0\|e_1 - \tilde{H}_{k+1}y\right\|$ `// least-squares problem` (11)

$x_{k+1} = x_0 + Q_{k+1}y$ `// iterative formula`

$\|r_{k+1}\| = \left\|\|r_0\|e_1 - \tilde{H}_{k+1}y\right\|$ `// update the norm of residual`

5.2 Appendix B

Algorithm 2: BiCG (biconjugate gradient) algorithm

Input: $A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n$ and $x_0 \in \mathbb{R}^n$ an initial guess of solution

Set $r_0 := q_0 = b - Ax_0$

Choose $\hat{q}_0 = \hat{r}_0$ such that $r_0^T \hat{r}_0 \neq 0$

Repeat until the convergence criterion is reached.

$\alpha_k = \frac{r_k^T \hat{r}_k}{q_k^T A^T \hat{q}_k}$

$x_{k+1} = x_k + \alpha_k q_k$ `// update iteration`

$r_{k+1} = r_k - \alpha_k A q_k$ `// update the residual`

$\hat{r}_{k+1} = \hat{r}_k - \alpha_k A^T \hat{q}_k$

$\beta_{k+1} = \frac{r_{k+1}^T \hat{r}_{k+1}}{r_k^T \hat{r}_k}$ `// unsymmetric Lanczos procedure`

$q_{k+1} = r_{k+1} + \beta_{k+1} q_k$

$\hat{q}_{k+1} = \hat{r}_{k+1} + \beta_{k+1} \hat{q}_k$

5.3 Appendix C

Algorithm 3: IDR(s) algorithm

Input: $A \in \mathbb{R}^{n \times n}, b \in \mathbb{R}^n$ and $x_0 \in \mathbb{R}^n$ an initial guess of solution, s minimum norm steps (positive integer)

Set $r_0 = b - Ax_0$

`// Build the vectors in` $\mathcal{G}_0$

for $k \leftarrow 0$ **to** $(s-1)$ **do**

  $v = Ar_k$; $\omega = \frac{v^T r_k}{v^T v}$

  $\Delta x_k = \omega r_k$; $\Delta r_k = -\omega v$

  $r_{k+1} = r_k + \Delta r_k$; $x_{k+1} = x_k + \Delta x_k$

$\Delta R_{k+1} = \left[\Delta r_k \middle| \ldots \middle| \Delta r_0\right]$; $\Delta X_{k+1} = \left[\Delta x_k \middle| \ldots \middle| \Delta x_0\right]$ `// forward differences`

Build $\mathcal{G}_i$ spaces for $i = 1, 2, 3, \ldots$ by first setting $k = s$

Repeat until the convergence criterion is reached. `// Loop over` $\mathcal{G}_i$ `spaces`

for $i \leftarrow 0$ **to** s **do**

  `// Loop` i `inside each` $\mathcal{G}_i$ `space`

  Solve y from $P^T \Delta R_k y = P^T r_k$

  $v = r_k - \Delta R_k y$

  if $i = 0$ **then**

    $q = Av$; $\omega = \frac{q^T v}{q^T q}$

    $\Delta r_k = -\Delta R_k y - \omega q$

    $\Delta x_k = -\Delta X_k y + \omega v$

  else

    $\Delta x_k = -\Delta X_k y + \omega v$

    $\Delta r_k = -A \Delta x_k$

  $r_{k+1} = r_k + \Delta r_k$ `// update the residuals`

  $x_{k+1} = x_k + \Delta x_k$ `// iterative solution`

  $k = k + 1$

  $\Delta R_k = \left[\Delta r_{k-1} \middle| \ldots \middle| \Delta r_{k-s}\right]$; $\Delta X_k = \left[\Delta x_{k-1} \middle| \ldots \middle| \Delta x_{k-s}\right]$