A Comparison of Some Iterative Methods in Scientific Computing

Shawn Sickel

Dr. Man-Chung Yeung, Mr. Jon Held
Department of Mathematics
University of Wyoming
July 29, 2005
Summer Research Apprentice Program

Abstract

Linear systems $\mathbf{Ax} = \mathbf{b}$ arose from industrial applications are usually large and sparse. It is very common that those systems involve millions of unknowns. Some basic solution methods that we learned in high-school classes, for example, Gaussian Elimination, do not work in finding the solutions of linear systems of such large size since they are too slower and require huge storage spaces. The most popular methods nowadays used in industry are iterative methods. In this paper, we selected and studied some iterative methods. Our experimental results show that Krylov subspace iterative methods are generally faster than the basic iterative methods. Among Krylov subspace iterative methods, Conjugate Gradient method is the best if $\mathbf{A}$ is symmetric and positive definite(SPD). When $\mathbf{A}$ is not SPD, but $\mathbf{A}^H$ is available, Bi-Conjugate Gradient method is then the best choice for the solution of the linear system. In the case where $\mathbf{A}^H$ is not available, Bi-Conjugate Gradient Stabilized method should be used.

Acknowledgement

For the success in my paper, I would like to thank my fellow SRAPers and the SRAP staff for encouragement and support. I would not have been able to complete this research paper without the help and guidance of Man-Chung Yeung and Jon Held. They taught me how to use Matlab computer programming, they helped edit my paper, and they showed me how the different iterative methods work.

Contents

1 Introduction

Iterative methods for solving general, large, sparse linear systems have been gaining popularity in many areas of scientific computing. This is due in great part to the increased complexity and size of the new generation of linear and nonlinear systems that arise from typical applications. At the same time, parallel computing has penetrated the same application areas, as inexpensive computer power has become broadly available and standard communication languages such as MPI have proved a much needed standardization. This has created an incentive to utilize iterative rather than direct solvers, e.g., Gauss Elimination, because the problems solved are typically from three or more dimensional models for which direct solvers often become ineffective due to the huge sizes of the resulting linear systems. Another incentive is that iterative methods are far easier to implement on parallel computers (see [4]).

Iterative methods nowadays can be divided into three groups: 1. basic iterative methods, 2. Krylov subspace methods, 3. Multigrid methods. In this paper, we will compare the performances through numerical experiments of the first two groups of methods. The numerical experiments will be done based on the test data from industry — we will download the test data from the government website *http://math.nist.gov/MatrixMarket/data* where industrial companies often post their test data.

The outline of the paper is as follows. In §2, we select some representative iterative methods and compare them through experiments. More precisely, we consider two famous basic iterative methods, Jacobi and Gauss-Seidel, in §2.1 and we think about three Keylov subspace methods: CG, BiCG and BiCGSTAB in §2.2. We end the paper with some conclusions based on our experimental results in §3.

2 Iterative methods and experiments

Consider the solution of

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{1}$$

where $\mathbf{A} \in \mathcal{C}^{N\times N}$ is a large, sparse matrix. In practice, when the coefficient matrix $\mathbf{A}$ is sparse, people always use iterative methods rather than direct methods, say, Gaussian Elimination(GE), for the solution of (1). The reason is that, GE needs to perform two steps to solve (1):

1. decompose $\mathbf{A}$ as $\mathbf{A} = \mathbf{L}\mathbf{U}$ where $\mathbf{L}$ is lower triangular and $\mathbf{U}$ is upper triangular;
2. solve $\mathbf{L}\mathbf{y} = \mathbf{b}$ and $\mathbf{U}\mathbf{x} = \mathbf{y}$ by forward and backward substitutions, respectively.

However, even though $\mathbf{A}$ is sparse, the resulting $\mathbf{L}$ and $\mathbf{U}$ are usually dense. If one uses GE to solve (1), one needs to store the $\mathbf{L}$ and $\mathbf{U}$ for the forward and backward substitutions in step #2. Therefore, the storage requirement of GE is $\frac{1}{2}N^2 + \frac{1}{2}N^2 = N^2$ of floating point numbers. This is a huge memory requirement since it is not unusual in industry that $N = 10^9$, and hence $N^2 = 10^{18}$ which is out of the storage capacity of all the computers nowadays.

Iterative methods, on the other hand, are cheap in storage. The typical operation in an iterative method is a matrix times a vector, $\mathbf{G}\mathbf{v}$. The $\mathbf{G}$ is usually constructed from $\mathbf{A}$ and is sparse. Hence $\mathbf{G}$ occupies only a little memory in a computer. Therefore, the memory required

to implement an iterative method is a very limited and that is why this kind of methods is so popular in practice when solving large, sparse linear systems.

Iterative methods nowadays can be divided into three groups:

i) basic iterative methods;

ii) Krylov subspace methods;

iii) Multigrid methods.

In this paper, we will only discuss and compare some of the methods from the first two groups through numerical experiments.

2.1 Basic iterative methods

There exist many basic iterative methods, see, for instance, [2] and [4]. In this section, we study two of them: Jacobi method and Gauss-Seidel method.

2.1.1 Jacobi method and Gauss-Seidel method

To solve the linear system (1), split $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{D} - \mathbf{E} - \mathbf{F}$$

where $\mathbf{D}$, $\mathbf{E}$ and $\mathbf{F}$ are the diagonal, strictly lower triangular and strictly upper triangular parts of $\mathbf{A}$. Then (1) becomes

$$(\mathbf{D} - \mathbf{E} - \mathbf{F})\mathbf{x} = \mathbf{b}. \tag{2}$$

From (2), we then have

$$\mathbf{x} = \mathbf{D}^{-1}(\mathbf{E} + \mathbf{F})\mathbf{x} + \mathbf{D}^{-1}\mathbf{b}.$$

Replace the $\mathbf{x}$ on the right by $\mathbf{x}^{(k)}$, the approximate solution at iteration step k, and the $\mathbf{x}$ on the left by $\mathbf{x}^{(k+1)}$, the approximate solution at iteration step $k+1$, we then have the Jacobi method

$$\mathbf{x}^{(k+1)} = \mathbf{D}^{-1}(\mathbf{E} + \mathbf{F})\mathbf{x}^{(k)} + \mathbf{D}^{-1}\mathbf{b}.$$

To obtain Gauss-Seidel method, write (2) as

$$(\mathbf{D} - \mathbf{F})\mathbf{x} = \mathbf{E}\mathbf{x} + \mathbf{b}.$$

Replace the $\mathbf{x}$ on the right with $\mathbf{x}^{(k)}$ and the $\mathbf{x}$ on the left with $\mathbf{x}^{(k+1)}$:

$$(\mathbf{D} - \mathbf{F})\mathbf{x}^{(k+1)} = \mathbf{E}\mathbf{x}^{(k)} + \mathbf{b}.$$

So,

$$\mathbf{x}^{(k+1)} = (\mathbf{D} - \mathbf{F})^{-1}\mathbf{E}\mathbf{x}^{(k)} + (\mathbf{D} - \mathbf{F})^{-1}\mathbf{b}$$

and this is called Gauss-Seidel method.

2.1.2 Convergence of a basic iterative method

In general, a basic iterative method always has the form

$$\mathbf{x}^{(k+1)} = \mathbf{G}\mathbf{x}^{(k)} + \mathbf{f} \tag{3}$$

where $\mathbf{G}$ is a matrix and $\mathbf{f}$ is a vector. Compared to (3), the $\mathbf{G}$ for Jacobi method is

$$\mathbf{G} = \mathbf{D}^{-1}(\mathbf{E} + \mathbf{F})$$

and the $\mathbf{G}$ for Gauss-Seidel method is

$$\mathbf{G} = (\mathbf{D} - \mathbf{F})^{-1}\mathbf{E}.$$

Theoretically, it can be proved that a basic iterative method (3) will approach the exact solution $\mathbf{x}^*$ of (1) if $\rho(\mathbf{G}) < 1$, namely, $\mathbf{x}^{(k)}$ will tend to $\mathbf{x}^*$ as k becomes larger and larger if $\rho(\mathbf{G}) < 1$. This is a result stated by the following theorem from Saad's book[4, p.112].

Theorem 2.1 *If $\rho(\mathbf{G}) < 1$ where $\rho(\mathbf{G})$ denotes the largest eigenvalues of $\mathbf{G}$ in absolute value, then the iterative method (3) converges for any $\mathbf{f}$ and any $\mathbf{x}_0$. Moreover, the smaller the $\rho(\mathbf{G})$ is, the faster the iterative method (3) will converge.*

2.1.3 Numerical experiments

Consider the problem of solving the following linear system:

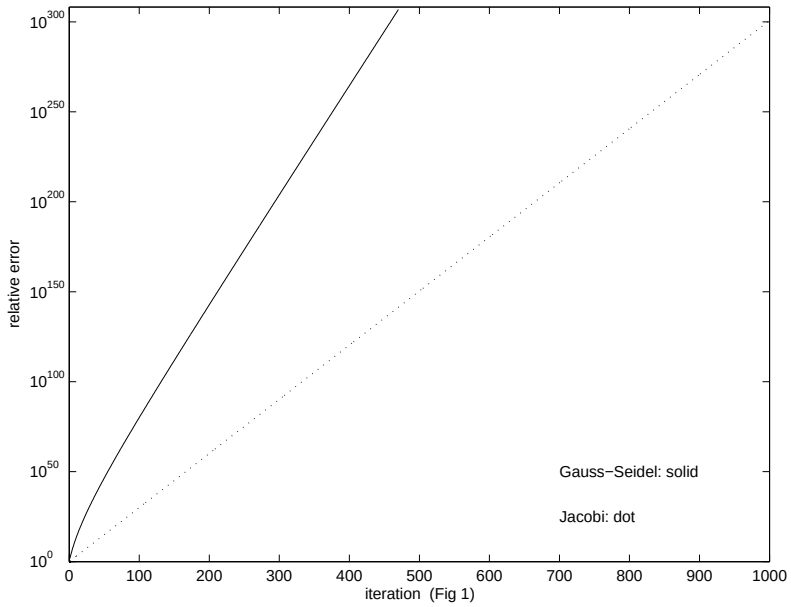
$$\begin{bmatrix} \alpha & -1 & & & & \\ -1 & \alpha & -1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & \alpha & -1 \\ & & & & -1 & \alpha \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_N \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ \vdots \\ 1 \end{bmatrix}.$$

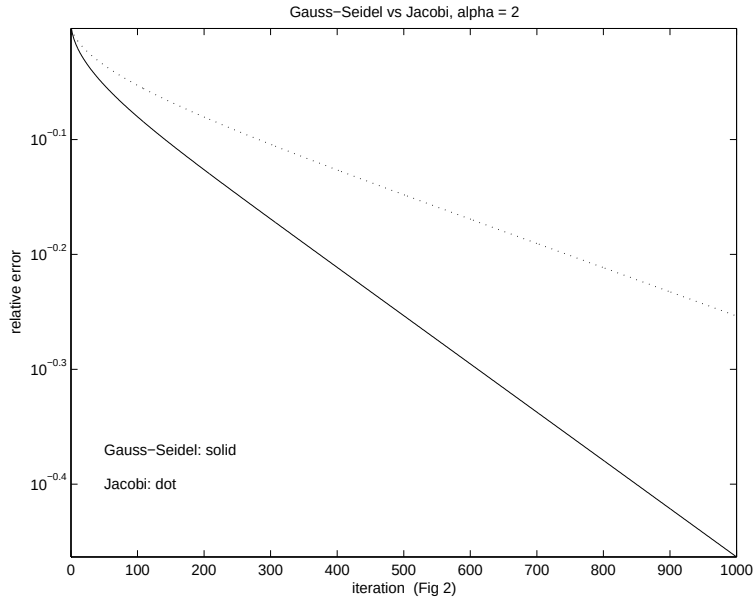
for $\alpha = 1, 2$ and 3. The corresponding $\rho(\mathbf{G})$ for Jacobi and Gauss-Seidel methods are as follows:

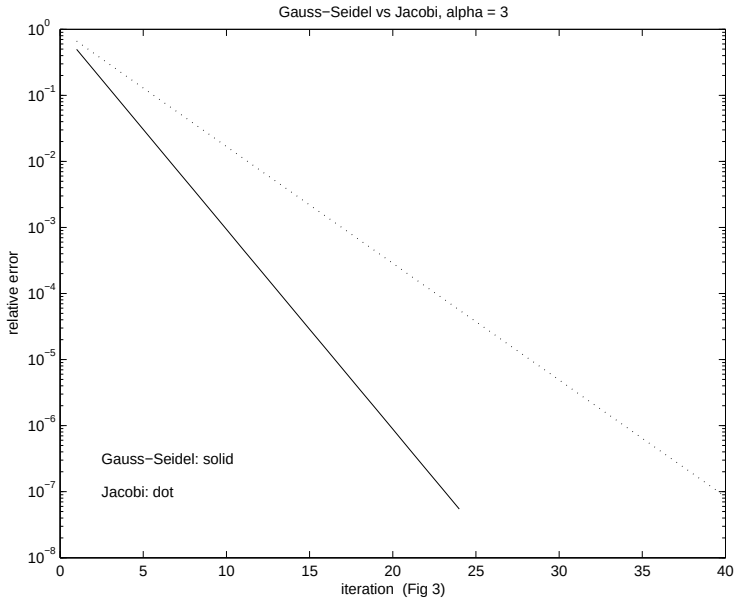
α	$\rho(\mathbf{G})$ of Jacobi	$\rho(\mathbf{G})$of Gauss-Seidel
1	1.9990	3.9961
2	0.9995	0.9990
3	0.6663	0.4440

Using this 100×100 matrix, we have computed the separate largest eigenvalues in absolute for each of the basic iterative methods that we are studying and listed them in the above table. Having looked at the corresponding eigenvalues between the Jacobi method and the Gauss-Seidel method, we found that $\rho(\mathbf{G})$ is smaller for the Gauss-Seidel method when $\alpha = 2, 3$, proving that Gauss-Seidel converges faster according to Theorem 2.1 (see Figures #2 and #3). Another reason why the Gauss Seidel method is faster is because it uses more updated

convergence values to find better guesses than the Jacobi method. For $\alpha = 1$, both methods diverge since their $\rho(\mathbf{G})$ are greater than 1 (see Fig #1).







2.2 Krylov subspace methods

Basic iterative methods can not solve all the linear systems. In fact, if $\rho(\mathbf{G}) < 1$, a basic iterative method will fail. In this section, we will study a second group of iterative methods, *Krylov Subspace methods*. The idea behind a Krylov subpace method is that, at iteration step k, the method searches a "good" approximate solution to the linear system (1) from the following subspace

$$span\{\mathbf{b}, \mathbf{Ab}, \mathbf{A}^2\mathbf{b}, \cdots, \mathbf{A}^{k-1}\mathbf{b}\}.$$

In mathematics, people call the above subspace *Krylov subspace*, and therefore, the corresponding methods are called *Krylov subspace methods*.

2.2.1 Conjugate gradient method (CG)

Conjugate gradient method[3] is the first Krylov subspace method in the history which is designed only for solving *symmetric positive definite (SPD)* linear system $\mathbf{Ax} = \mathbf{b}$, namely, the coefficient matrix $\mathbf{A}$ satisfies i) $\mathbf{A}^H = \mathbf{A}$; ii) $\mathbf{x}^H\mathbf{Ax} > 0$ for any $\mathbf{x} \neq \mathbf{0}$. If the linear system is not SPD, the CG method will fail to converge.

Two professors from UCLA created this method in 1952. Here is the algorithm[4, p.190]:

Conjugate Gradient Iterative Method

Set $\mathbf{r}_0 = \mathbf{b} - \mathbf{Ax}_0$, $\mathbf{p}_0 = \mathbf{r}_0$
For $k = 1, 2, \cdots$
$\alpha_{k-1} = \mathbf{r}_{k-1}^H\mathbf{r}_{k-1}/\mathbf{p}_{k-1}^H\mathbf{Ap}_{k-1}$;
$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_{k-1}\mathbf{p}_{k-1}$;
$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_{k-1}\mathbf{Ap}_{k-1}$;
$\beta_{k-1} = \mathbf{r}_k^H\mathbf{r}_k/\mathbf{r}_{k-1}^H\mathbf{r}_{k-1}$;
$\mathbf{p}_k = \mathbf{r}_k + \beta_{k-1}\mathbf{p}_{k-1}$;
End

2.2.2 Bi-conjugate gradient method (BiCG)

Then, a natural question that you may ask is that, if we are given a non-SPD system, is there any *Krylov subspace method* to find the solution to the linear system? The answer is *Yes*. The following Krylov iterative method is called *Bi-Conjugate Gradient Method*, which can solve any linear system and was created by a mathematician named Fletcher[1] in 1974. The following algorithm was copied from Saad's book[4, p.223].

Bi-Conjugate Gradient Iterative Method

Set $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$. Choose $\mathbf{r}_0^*$ such that $(\mathbf{r}_0^*)^H \mathbf{r}_0 \neq 0$.
Set $\mathbf{p}_0 = \mathbf{r}_0, \mathbf{p}_0^* = \mathbf{r}_0^*$.
For $k = 1, 2, \cdots$
$\alpha_{k-1} = \mathbf{r}_{k-1}^H \mathbf{r}_{k-1}^* / \mathbf{p}_{k-1}^H \mathbf{A}^H \mathbf{p}_{k-1}^*$;
$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_{k-1}\mathbf{p}_{k-1}$;
$\mathbf{r}_k = \mathbf{r}_{k-1} - \alpha_{k-1}\mathbf{A}\mathbf{p}_{k-1}$;
$\mathbf{r}_k^* = \mathbf{r}_{k-1}^* - \alpha_{k-1}\mathbf{A}^H \mathbf{p}_{k-1}^*$;
$\beta_{k-1} = \mathbf{r}_k^H \mathbf{r}_k^* / \mathbf{r}_{k-1}^H \mathbf{r}_{k-1}^*$;
$\mathbf{p}_k = \mathbf{r}_k + \beta_{k-1}\mathbf{p}_{k-1}$;
$\mathbf{p}_k^* = \mathbf{r}_k^* + \beta_{k-1}\mathbf{p}_{k-1}^*$;
End

2.2.3 Bi-conjugate gradient stabilized (BiCGSTAB)

In some applications, we want to solve $\mathbf{A}\mathbf{x} = \mathbf{b}$, but we do not know the matrix $\mathbf{A}$. What we know is that, given us a vector $\mathbf{v}$, we can have $\mathbf{A}\mathbf{v}$ through some other methods. Hence we can know the result $\mathbf{A}\mathbf{v}$, but have no way to know $\mathbf{A}^H\mathbf{v}$. Therefore, BiCG is not applicable in such situations.

The following algorithm can solve any linear system, but not involves $\mathbf{A}^H$ in its implementation. The algorithem is called *Bi-Conjugate Gradient Stabilized method* and was created by van der Vorst[5] in 1992. We copied the algorithm from Saad's book[4, p.234].

BiCGSTAB

Set $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$. Choose $\mathbf{r}_0^*$.
Set $\mathbf{p}_0 = \mathbf{r}_0$.
For $k = 1, 2, \cdots$
$\alpha_{k-1} = \mathbf{r}_{k-1}^H \mathbf{r}_0^* / (\mathbf{A}\mathbf{p}_{k-1})^H \mathbf{r}_0^*$;
$\mathbf{s}_{k-1} = \mathbf{r}_{k-1} - \alpha_{k-1}\mathbf{A}\mathbf{p}_{k-1}$;
$\omega_{k-1} = (\mathbf{A}\mathbf{s}_{k-1})^H \mathbf{s}_{k-1} / (\mathbf{A}\mathbf{s}_{k-1})^H \mathbf{A}\mathbf{s}_{k-1}$;
$\mathbf{x}_k = \mathbf{x}_{k-1} + \alpha_{k-1}\mathbf{p}_{k-1} + \omega_{k-1}\mathbf{s}_{k-1}$;
$\mathbf{r}_k = \mathbf{s}_{k-1} - \omega_{k-1}\mathbf{A}\mathbf{s}_{k-1}$;
$\beta_{k-1} = \frac{\mathbf{r}_k^H \mathbf{r}_0^*}{\mathbf{r}_{k-1}^H \mathbf{r}_0^*} \times \frac{\alpha_{k-1}}{\omega_{k-1}}$;
$\mathbf{p}_k = \mathbf{r}_k + \beta_{k-1}(\mathbf{p}_{k-1} - \omega_{k-1}\mathbf{A}\mathbf{p}_{k-1})$;
End

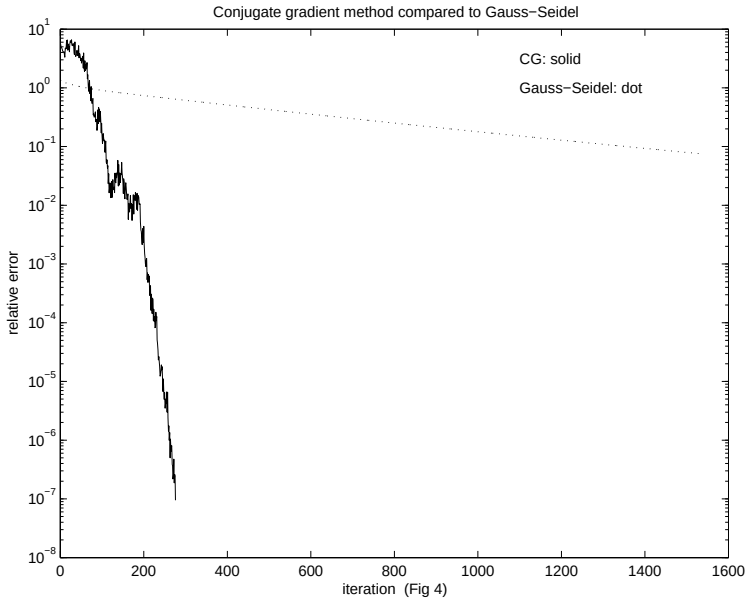
From the algorithm, we can that it only involves the computation $\mathbf{A}\mathbf{v}$, no $\mathbf{A}^H\mathbf{v}$.

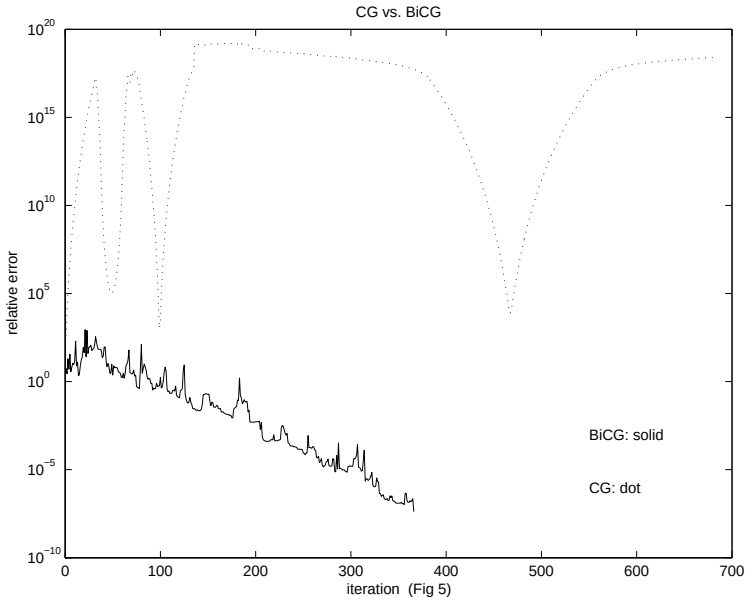
2.2.4 Numerical experiments

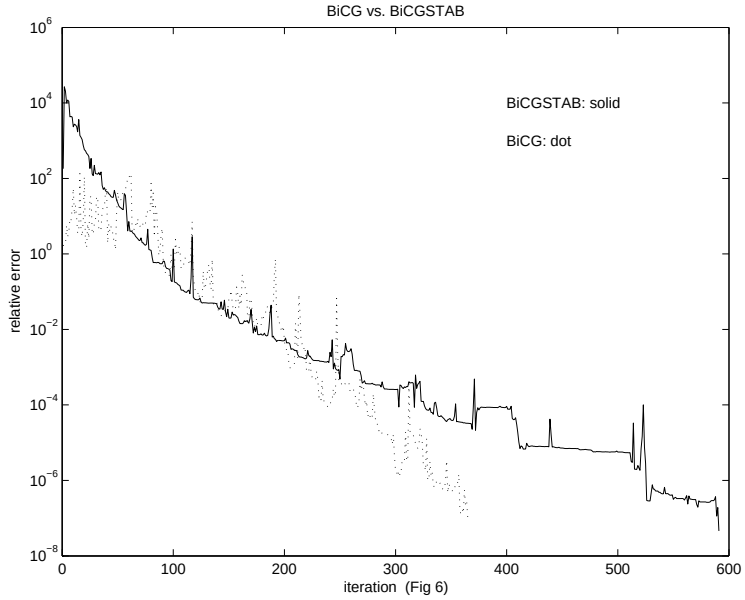
The GS method only converges when $\rho(\mathbf{G}) < 1$. For a large matrix, GS has many more iterations than the CG method does. Figure 4 shows the convergence of both methods, where the test matrix is a 2003×2003 structural engineering matrix labeled *bcsstk05* from the group BCSSTRUC1, Harwell-Boeing collection. We can see that, when solved by the CG method, it converged after about 300 iterations, whereas the GS method has yet to converge after 1600 iterations. GS is slow to converge because $\rho(\mathbf{G}) = 0.9986$, very close to 1. As long as the matrix is SPD, the CG method will be faster than the GS method, because the CG method is approximating the solution using the span of the subspace, and must converge within N steps, where N is the size of the matrix.

For CG to work, the matrix $\mathbf{A}$ must be a SPD matrix. Figure 5, a chemical kinetics RCHEM radiation study matrix labeled *fs_680_1* from the group FACSIMILE of Harwell-Boeing collection, is a 680×680 matrix. BiCG is used to solve the asymmetrical linear systems, while CG can only solve SPD matrices and this is why CG can not converge in this experiment and is going up instead of down.

BiCGSTAB is basically BiCG, but does not require the A-transpose $\mathbf{A}^H$ in its implementation, so it maybe somewhat slower, but can still solve linear systems that BiCG can not — in some applications, we only know $\mathbf{A}$ without knowing $\mathbf{A}^H$. Figure 6, using the same matrix from Figure 5, shows the slight difference in BiCG and BiCGSTAB's convergence rates.







3 Conclusion

In this paper, we have looked over basic iterative methods and krylov subspace methods. Among these, our goal was to find out which was the most efficient under certain conditions. Basic iterative methods, such as the Gauss-Seidel method and the Jacobi method represent the first steps into solving linear systems with iterations.

The test data was taken from Matrix Market, at http://math.nist.gov/MatrixMarket/. *Bcsstk05* is the page with the data used. The Jacobi and Gauss-Seidel methods both have the same properties as to what makes it converge faster or indefinable. If $\rho(\mathbf{G}) > 1$, the basic iterative methods will not converge. The close the p(G) is to 1, the slower it will converge.

The Gauss-Seidel method has a lower value of p(G) than the Jacobi method does, so that the Gauss-Seidel method converges faster.

The CG method requires the matrix to be a SPD, a Symmetrical Positive Definite matrix. This method uses the span not just the guesses of the previous guess step. It is considerably faster than the Gauss-Seidel method, and has a maximum of N steps to converge for an $N \times N$ matrix. Gauss-Seidel on the other hand, will have many more iterations at larger dimensions.

CG and BiCG are two different methods for two different situations. Depending on the givens, the one of the equations is useless. For example, If the matrix is not an SPD matrix, then CG will not converge, so BiCG must be used. If the matrix is SPD, though, CG must be used since CG is cheaper in computational cost per iteration than BiCG.

BiCGSTAB is needed because in real life situations, The A-Transpose might not always be a given component in solving the linear systems. BiCGSTAB does not converge as fast as BiCG, but BiCGSTAB will converge without the transpose.

References

[1] R. Fletcher, *IConjugate gradient methods for indefinite systems*, in Proceedings of the Dundee Biennial Conference on Numerical Analysis, 1974, G. A. Watson, ed., Springer-Verlag, New York, 1975, pp. 73-89.

[2] W. Hackbusch, *Iterative solution of large sparse systems of equations*, Springer-Verlag, New York, 1994.

[3] M. R. Hestenes and E. L. Stiefel, *Methods of conjugate gradients for solving linear systems*, Journal of Research of the National Bureau of Standards, Section B, 49(1952), pp. 409-436.

[4] Y. Saad, *Iterative methods for sparse linear systems*, 2nd edition, SIAM, Philadelphia, PA, 2003.

[5] H. A. van der Vorst, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems,* SIAM J. Sci. Statist. Comput., 12 (1992), pp. 631–644.