Experiments with The Triangle Algorithm for Linear Systems*

Thomas H. Gibson† and Bahman Kalantari ‡

Abstract

We present some computational experimentation for solving a linear system $Ax = b$ by applying the *Triangle Algorithm* described in [3], based on converting the system to a convex hull problem as described in [4]. A comparison of the performance with classical iterative methods such as Jacobi, Gauss-Seidel, and Successive Over-Relaxation method for small size problems suggests the Triangle Algorithm is competitive, requiring no restrictions on the input matrix.

1 Introduction

Solving linear systems of equations is undoubtedly one of the most common and practical problems in numerous aspects of scientific computing and applied mathematics. Iterative methods offer very important alternatives to direct methods and find applications in problems that require a solution to very large or sparse linear systems of equations. For example, discretization of partial differential equations via finite difference or finite elements methods. Classical iterative methods for solving linear systems of equations include the Jacobi, Gauss-Seidel, and successive over-relaxation (SOR) methods, see [6, 1, 2]. These methods will be compared with the Triangle Algorithm.

2 The Triangle Algorithm

Before we discuss the system solving performance of the *Triangle Algorithm*, we first introduce the algorithm which is designed in the context of the convex hull problem. Given a set $S = \{v_1, v_2, \ldots, v_n\} \subset \mathbb{R}^m$ and a distinguished point $p \in \mathbb{R}^m$, the *convex hull decision problem* (or just the convex hull problem) is to test if p is in $conv(S)$, the convex hull of S. The *Triangle Algorithm* uses the following:

Theorem 1. (Distance Duality [3]) *$p \in conv(S)$ if and only if given any $p' \in conv(S)$, there exists a v_j, called pivot, such that $\|p' - v_j\| \geq \|p - v_j\|$.* □

If no pivot exists for p', then $\|p' - v_j\| < \|p - v_i\|$ $\forall i = 1, \ldots, n$. We call such p' a *witness*. Given a desired tolerance $\epsilon \in (0, 1)$, and an iterate $p' = \sum_{i=1}^{n} \alpha_i v_i \in conv(S)$, as the name suggests, the Triangle Algorithm searches for a triangle $\triangle pp'v_j$ where $v_j \in S$ is a pivot. If such a triangle exists, the algorithm uses v_j to compute the point $p'' = \sum_{i=1}^{n} \alpha_i' v_i \in conv(S)$ as the closet point to p on the line segment $p'v_j$. The algorithm continues until a desired approximate solution in absolute or relative error is obtained, or a witness is found. In the case of the latter, a witness found implies that our target point p is not in the convex hull. We may refer to the above algorithm as the *Naive Triangle Algorithm*. As shown in [3], each iteration takes at most $O(mn)$ arithmetic operations to compute a pivot. Given the coordinates of p' and α_i's that give the iterate's representation as a convex combination of the v_i's, it takes $O(m)$ operations to compute p'' and $O(n)$ operations to compute the α_i''s. Alternative iteration complexity bounds are derived in [3] in order to compute $p' \in conv(S)$ so that $\|p - p'\| \leq \epsilon \|p - v_i\|$, for some v_i. One such a bound is $O(1/\epsilon^2)$. The pivot selected in the naive implementation may not be the best possible choice, however it is computed efficiently. A *blindfold* version of Triangle Algorithm is described in [5].

3 Pivot Angle Optimization

The Triangle Algorithm is a geometrical algorithm. During each iteration, the algorithm searches for a triangle $\triangle pp'v$ that uses v as a pivot to compute the next iterate p''. However for the iterate to make the best possible approximation of p, we need to minimize the gap $\|p - p''\|$. The trigonometric relationship between the gap and *pivot angle* $\theta' = \angle pp'v$ is related by the sine of θ', that is $\delta' = \delta \sin(\theta')$, where $\delta = \|p - p'\|$

*This research was carried out with the first author as DIMACS 2013 Summer REU student, supervised by the second.

†Department of Mathematics. Baylor University, thomas_gibson@baylor.edu

‡Department of Computer Science. Rutgers University, kalantari@cs.rutgers.edu

and $\delta' = \|p - p''\|$. If the gap is reduced as much as possible, then we can make the best possible approximation p'' to p in that iteration. We thus may choose the pivot that has the smallest pivot angle.

4 Auxiliary Pivots

Given $S \in \mathbb{R}^m$, a finite subset $S' \in conv(S)$, distinct from S is called *auxiliary pivots*. An iteration of the Triangle Algorithm that searches for pivots in $S \cup S'$ may result in a more optimal choice. Many strategies are possible for generating S'. A simple strategy is this: Once it is apparent that cycling is occurring, i.e. the same sequence of points are being selected as pivots, we introduce their mid-point as new points. Subsequent iterations then may use such auxiliary points as pivot. The *mid-point method* is one out of numerous other heuristic possibilities. It greatly improves the performance, however other auxiliary pivot strategies are described in detail in [3].

5 Solving Linear Systems

Consider solving a linear system $Ax = b$ in dimension n. In many real world applications iterative methods are often the only practical way to solve the linear system due to the fact that direct methods would be prohibitively expensive and impossible in some cases. This is usually the case for systems which are extremely large. Suppose $x = A^{-1}b$ is nonnegative. Then it suffices to approximate $0 \in conv(\{a_1, \ldots, a_n, -b\})$. The required accuracy is proven in [3] (see Sensitivity Theorem). For small examples the performance is typical of the following.

$$A = \begin{pmatrix} 4 & 2 & -2 \\ 3 & 6 & 1 \\ 0 & 2 & 3 \end{pmatrix}, \quad b = \begin{pmatrix} 8 \\ 13 \\ 5 \end{pmatrix}, \quad x = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}$$

Our initial iterate is the centroid of the system, $p' = (-1, -\frac{3}{4}, 0)^T$. For comparison, we use the Jacobi, Gauss-Seidel, and SOR methods for solving the same system, with the same initial guess x_0. For SOR, we use an optimal relaxation factor $\omega = 1.21$. The criteria for termination was $\|b - Ax_0\| < 10^{-8}$.

Comparisons	
Method	Iterations
Triangle Algorithm	1
Jacobi	59
Gauss-Seidel	24
Successive Over-Relaxation (SOR)	15

Suppose that it is not known if $x = A^{-1}b$ is nonnegative. Then for some scalar $t > 0$ the solution of $Ax = b + tAe$, where e is the vector of ones, will be nonnegative. If we know the value of such t and solve the new system, its solution x_* will satisfy $A(x_* - te) = b$ so that $x_* - te$ is the desired solution to $Ax = b$. Thus we can apply the Triangle Algorithm to solve $Ax = b + tAe$. Rather than guessing a value for t, in [4] the *Incremental Triangle Algorithm* is described: As in the nonnegative case we apply the Triangle Algorithm to solve $Ax = b$. If the solution is not nonnegative, the algorithm will produce a witness. Using this we increase t from zero to a positive number and repeat the process. The computational performance is as good.

6 Concluding Remarks

The implementation of two proposed versions of the Triangle Algorithm in [4] for computing an approximate solution to a linear system in several examples has proven to be quite successful in solving a small linear system, outperforming the other methods. However, there are examples where the performance of the Triangle Algorithm could be slow. This is still subject to further experimentation. Large scale and more experiments are required and there are still many other iterative methods, such as the Krylov methods, that the Triangle Algorithm has yet to be compared with. In future work we plan to investigate the incorporations of auxiliary methods to the Triangle Algorithm to improve its performance. While the algorithm is still in early stages of experimentation, it is a promising method that could potentially become another viable tool for a range of problems from system solving to optimization and linear programming.

References

[1] Cullen, Charles G. An Introduction to Numerical Linear Algebra. Boston: PWS Pub., 1994. Print.

[2] Kahan, W. Gauss-Seidel Methods of Solving Large Systems of Linear Equations. Ph.D. thesis. Toronto, Canada, University of Toronto, 1958.

[3] B. Kalantari, A characterization theorem and an algorithm for a convex hull problem, arxiv.org/pdf/1204.1873v3.pdf, 2013.

[4] B. Kalantari, Solving linear system of equations via a convex hull algorithm, arxiv.org/pdf/1210.7858v1.pdf, 2012.

[5] B. Kalantari, Finding a lost treasure in convex hull of points from known distances. In the Proceedings of the 24th Canadian Conference on Computational Geometry (2012), 271 - 276.

[6] Saad, Y. "Basic Iterative Methods." Iterative Methods for Sparse Linear Systems. 2nd ed. Philadelphia: SIAM, 2003. N. pag. Print.