Vietnam Journal of Mathematics **37**:1(2009) 91-96

Vietnam Journal
of
MATHEMATICS

An Algorithm for Solving the Nearest Point Problem in an Affine Subspace

Pham Canh Duong[1] **and Le Thanh Hue**[2]

[1] *Institute of Mathematics, 18 Hoang Quoc Viet Road, 10307 Hanoi, Vietnam*

[2] *Faculty of Informatic Technology, University of Mining and Geology, Hanoi, Vietnam*

Received October 13, 2008
Revised December 17, 2008

Tóm tắt nội dung. We present a new algorithm for finding the projection of a vector into the intersection of a given set of hyperplanes in $\mathbb{R}^n$. This algorithm is simple to implement and is shown to be more efficient than the well-known original conjugate gradient algorithm.

2000 Mathematics Subject Classification: 90C05, 90C59, 65K05, 65F10.

Key words: Linear program, successive orthogonal projections algorithm, linear convergence, convex feasibility problem.

1. Introduction

Let's denote by $L(A, b)$ the m-dimensional affine subspace of $\mathbb{R}^n$, $m \leq n$, defined by

$$L(A, b) := \{x \in \mathbb{R}^n | Ax = b\}, \tag{1}$$

where A is a full rank $m \times n$ - real matrix and $b \in \mathbb{R}^n$.

In this paper $\mathbb{R}^n$ is considered as a n-dimensional euclidean space equipped with the usual scalar product and euclidean distance.

Given an arbitrary point $x \in \mathbb{R}^n$ and affine subspace $L(A, b)$, the nearest point problem in $L(A, b)$ is formulated as follows.

$$\text{Find } x^* \in L(A,b) \text{ such that } (\forall y \in L)\ \|x - x^*\| \leq \|x - y\|. \tag{2}$$

If we denote by $P_L : \mathbb{R}^n \to L(A, b)$ the projection operator from $\mathbb{R}^n$ onto $L(A, b)$ then (2) can be rewritten as $x^* = P_L x$. This problem is of great practical interest because it appears in the computational core of several other algorithms (see [3, 5]). Preferred method for its solution is usually chosen among variants of the well-known conjugate gradient method [4, 5, 6].

In this paper, we present a new algorithm for solving (2) having better complexity estimate than the one obtained for the conjugate gradient method. This is based on simple results presented in Sec. 2. The description of this new algorithm and its complexity analysis is given in Sec. 3. In Sec. 4, some preliminary computational results are discussed.

2. Preliminaries

Denote by A_i the i-th row of A, $i = 1, 2, \ldots, m$. Suppose that $m \leq n$ and Rank$A = m$. Let x_0 be an arbitrary point in $\mathbb{R}^n$. Consider the following set $\{x_i\}_{i=0}^m$ which is defined as

$$\begin{cases} x_0 = x_0, i = 0, \\ x_i = P_{H_i} x_0, 0 < i \leq m, \end{cases} \tag{3}$$

where $P_{H_i} x_0$ denote the projections of x_0 onto hyperplanes

$$H_i := \{x \in \mathbb{R}^n \,|\langle A_i, x\rangle = b_i\}.$$

The following result establishes the relationship between set (3) and the solution x^* to problem (2).

Proposition 2.1. *Suppose that x^* is the solution to problem* (2) *and let*

$$0 = \frac{x^* + x_0}{2}. \tag{4}$$

Then

$$(\forall i \in N, 0 < i \leq m),\ \|x_i - 0\| = \frac{1}{2}\|x^* - x_0\|. \tag{5}$$

Remark 2.2. Proposition 2.1 is a generalization of Theorem 1 in [2] by the authors and could be proved by the arguments similar to those used in that paper. Here, we give the proof for the sake of completeness.

Proof. By (4) we have

$$(0 - x_i) = \frac{1}{2}\left((x^* - x_i) + (x_0 - x_i)\right).$$

This implies

$$\|(0 - x_i)\|^2 = \frac{1}{4}\left(\|x^* - x_i\|^2 + \|x_0 - x_i\|^2 + 2\langle x^* - x_i, x_0 - x_i\rangle\right).$$

Since $(x^* - x_i) \in H_i$ and $(x_0 - x_i)$ is orthogonal to H_i, we have $\langle x^* - x_i, x_0 - x_i\rangle = 0$.
Hence,

$$\|x_i - 0\|^2 = \frac{1}{4}\left(\|x^* - x_i\|^2 + \|x_0 - x_i\|^2\right). \tag{6}$$

On the other hand, from the obvious equality:

$$(x^* - x_0) = (x_i - x_0) + (x^* - x_i),$$

we have

$$(x^* - x_0)^2 = (x_i - x_0)^2 + (x^* - x_i)^2 + 2\langle x_i - x_0, x^* - x_i\rangle.$$

Since $\langle x_i - x_0, x^* - x_i\rangle = 0$, this implies

$$\|x^* - x_0\|^2 = \|x_i - x_0\|^2 + \|x^* - x_i\|^2.$$

Thus, the equality (6) can be written as

$$\|0 - x_i\| = \frac{1}{2}\|x^* - x_0\|.$$

The proof is complete ■

This result has a simple geometrical interpretation. Suppose that the points $\{x_i\}_{i=0}^m$ are affinely independent and let's denote by V_m their affine span, and by S_m the m-dimensional sphere in V_m passing through all the points $\{x_i\}_{i=0}^m$. Then Proposition 2.1 says that the solution x^* to (2) must lie on the sphere S_m. Furthermore, x^* must be symmetric to x_0 with respect to the center O of S_m. So, (2) will be solved if one knows a way to find the center O. To this end, first, consider the following simple problem.

Auxiliary Problem. Let a, b and c be three distinct points in $\mathbb{R}^n$ and v be some vector in $\mathbb{R}^n$ not orthogonal to $(a - b)$, i.e. $\langle a - b, v\rangle \neq 0$.

Let $l(\lambda)$ be the straight line defined as

$$l(\lambda) := c + \lambda v, \lambda \in \mathbb{R}. \tag{7}$$

Consider the following auxiliary problem:

Find $\lambda^* \in \mathbb{R}$ such that

$$\|a - l(\lambda^*)\| = \|b - l(\lambda^*)\| . \tag{8}$$

To solve this problem let's set

$$z = \frac{1}{2}(a+b); w = \frac{a-b}{\|a-b\|} \tag{9}$$

and P be the hyperplane defined by

$$P := \{x \in R^n \,|\langle x, w\rangle = \langle z, w\rangle\} .$$

It is clear that a and b are symmetric with respect to P. Thus, the point to be found $l(\lambda^*)$, is just the intersection of $l(\lambda)$ with the plane P.
Consequently, we must have

$$\langle l(\lambda^*), w\rangle = \langle z, w\rangle \text{, i.e.}$$

$$\langle c + \lambda^* v, w\rangle = \langle z, w\rangle .$$

Hence, we obtain λ^* which solves (8)

$$\lambda^* = \frac{\langle z - c, w\rangle}{\langle v, w\rangle}. \tag{10}$$

3. Description of the Algorithm

As mentioned earlier, to solve the nearest point problem (2), it is sufficient to find the center O of the smallest sphere determined by points (3). This can be done as follows.

Algorithm 1. **(1) Initialization:** Generate $x_i = P_{H_i} x_0, i = 1, 2, 3, ..., m$.

(2) Step 1. Set

$$0_1 = \frac{1}{2}(x_1 + x_0);$$
$$e_1 = \frac{x_1 - x_0}{\|x_1 - x_0\|}.$$

(3) Step k. Calculate

$$\begin{cases} d = x_k - \sum\limits_{i=1}^{k-1} \langle x_k - x_0, e_i\rangle . e_i, \\ e_k = \frac{d}{\|d_k\|}. \end{cases} \tag{11}$$

Set

$$a = x_0; \quad b = x_k; \quad c = 0_{k-1}; \quad v = e_k.$$

Solve problem (8) using (10) to determine

$$\lambda = \frac{\langle z - c, w \rangle}{\langle v, w \rangle},$$

where

$$z = \frac{1}{2}(a + b) \; ; \quad w = \frac{(a - b)}{\|a - b\|}.$$

Calculate $0_k = c + \lambda v$.
Set $k \leftarrow k + 1$. if $(k > m)$ then goto Stop;
else goto Step k.

(4) Stop $O = O_m$
The solution of the nearest point problem (2) is

$$x^* = 2O - x_0. \tag{12}$$

We note that the formula (11) is similar to the one used in the well-known Gramm-schmidt orthogonalization process. So, the vector $\{e_k\}_{k=1}^m$ are orthonormal. The use of auxiliary problem (8) in each step k guarantee $\|O_k - x_k\| = \|O_k - x_j\|$ for all $j < k$. Therefore, O_m is the center of the m-dimensional sphere passing through points (3).

Complexity analysis. We will estimate the computational complexity expressed in the numbered of required vector-to-vector multiplications. It is easily seen that most of the computational cost in the above algorithm arises from the need to compute the vector e_k in (11). At step k, the number of vector-to-vector multiplications required to compute e_k is $(k - 1)$. So, the total number of vector-to-vector multiplications required by the algorithm is

$$(1 + 2 + ... + (m + 1)) \approx \frac{m^2}{2}. \tag{13}$$

If the matrix A in (1) is square, i.e. $m = n$, then the subspace $L(A, b)$ is reduced to a single point which is the solution to the linear system $Ax = b$. Therefore, the proposed algorithm can also be used to solve general linear systems. The estimate (13) shows that it is more efficient than the original conjugate gradient method which requires, in exact real arithmetic, about n^2 vector-to-vector multiplications to solve a regular linear system of n variables (see, e.g. [4, 6, 1] for more details).

4. Numerical Experiments

The algorithm was tested using MatLab on a set of matrices of size ranging from $n = 50$ to $n = 1200$, taken from the netlib.org website. The exact solution vector x^* was chosen to be $x^* = (1, 2, \ldots, n)$ and vector $\mathbf{b}$ then was computed accordingly as $\mathbf{b} = \mathbf{A}x^*$. With the starting point $x_0 = (0, 0, \ldots, 0)$, we then applied the algorithm to the resulting linear system $\mathbf{A}x = \mathbf{b}$ to obtain the actual solution $\bar{x}$. The computational accuracy was estimated as

$$Error \ = \ \sum_{i=1}^{n} |\bar{x_i} - x_i^*|^2.$$

The proposed algorithm is computationally stable. In all of our tests, it always found, in one round, the solution with the average accuracy between 10^{-8} to 10^{-15} depending on the value of the condition number of the input systems.

Tài liệu

1. H. M. Brucker, Iteratively Solving Large Sparse Linear Systems on Parallel Computers, in *Quantum Simulations of Complex Many-Body Systems: From Theory to Algorithms*, Lecture Notes, J. Grotendorst, D. Marx, A. Muramatsu (Eds.), John von Neumann Institute for Computing, Julich, NIC Series, Vol. 10, ISBN 3-00-009057-6, 2002, pp. 521–548.
2. P. C. Dương, L. T. Huệ, and N. H. Vũ, Một phương pháp lặp giải hệ phương trình tuyến tính cỡ lớn, *Tạp chí ứng dụng Toán học* (III) (2005), 29–44.
3. G. H. Golub and C. F. VanLoan, *Matrix Computations*, Published by the Johns Hopkins Press, 1996.
4. M. R. Hestenes and E. Stiefel, Method of Conjugate Gradients for solving Linear Systems, *J. Res. Nat. Bur. Stand.* **49** (1952), 409-436 .
5. Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing Company, Boston, 1996.
6. Y. Saad and H. A. Vander Vorst, Iterative solution of linear systems in the 20-th century, *J. Comput. Appl. Math.* **123** (2000).