2008 International Symposium on Nonlinear Theory and its Applications
NOLTA'08, Budapest, Hungary, September 7-10, 2008

A New Decomposition Algorithm for Solving Convex Quadratic Programming Problems

Norikazu Takahashi†, Yuta Kobayashi‡ and Bo Chen‡

†Faculty of Information Science and Electrical Engineering, Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395 Japan
‡Department of Electrical Engineering and Computer Science, Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395 Japan
Email: norikazu@csce.kyushu-u.ac.jp

Abstract—A new decomposition method for solving general convex quadratic programming problems is proposed in this paper. Experimental results show that the proposed method is effective when the number of variables is large and the number of equality constraints is small.

1. Introduction

Quadratic programming (QP) problem is one of the most fundamental nonlinear optimization problems. Many problems arising in engineering and science can be formulated as the QP problem. For example, the training of support vector machines (SVMs), a pattern classification technique which has attracted considerable attention in recent years, is formulated as a special type of QP problems [1].

In a QP problem, one has to minimize a quadratic function of some variables subject to a set of linear equality constrains and a set of linear inequality constraints. If the objective function is convex, the problem is said to be a convex QP problem. It is often said that the computation time of a convex QP problem with n variables is proportional to n^3. In case of SVMs, the number of variables in the QP problem is equal to the number of training samples, which is often very large, for example, more than ten thousands. Therefore, the reduction of the computation time for large scale convex QP problems is very important for the training of SVMs.

Decomposition method was first proposed by Osuna *et. al* [2] in order to solve large-scale convex QP problems arising in the training of SVMs efficiently. A basic strategy of the decomposition method is to execute two operations repeatedly until some optimality condition is satisfied: one is to select a set of q variables, which is called the working set, among all variables; the other is to minimize the objective function by updating only the selected q variables. Since then many authors have developed different types of decomposition methods for SVMs [3, 4], and some of them are widely used as the standard methods.

In this paper, we propose a new decomposition method which is applicable to general convex QP problems. The conventional decomposition methods are restricted to problems with one equality constraint, and the working set selection method depends heavily on this fact. In the proposed method, on the other hand, the working set selection is done based on the optimal solution of a linear programming problem which is solved to check the stopping condition. We also evaluate the effectiveness of the proposed method through many experiments.

2. Quadratic Programming Problem

A general QP problem is formulated as follows:

$$\begin{array}{ll} \text{Minimize} & f(x) = \frac{1}{2}x^T Qx + c^T x \\ \text{Subject to} & Ax = b \\ & Ex \le d \end{array} \tag{1}$$

where $x \in \mathbb{R}^n$ is a variable; $Q \in \mathbb{R}^{n\times n}$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, $E \in \mathbb{R}^{l\times n}$ and $d \in \mathbb{R}^l$ are constants; $Ex \le d$ stands for componentwise inequality. If Q is positive semi-definite, the objective function $f(x)$ is convex and hence (1) is called a convex QP problem.

In this paper, we will consider

$$\begin{array}{ll} \text{Minimize} & f(x) = \frac{1}{2}x^T Qx + c^T x \\ \text{Subject to} & Ax = b \\ & x \ge 0 \end{array} \tag{2}$$

instead of (1), where $x \in \mathbb{R}^n$ is a variable; $Q \in \mathbb{R}^{n\times n}$, $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$ are constants. Problem (2) can be obtained by setting $l = n$, $E = -I$ and $d = 0$ in (1) where I is the identity matrix. Thus (2) seems to be a special case of (1), but, as shown in Appendix, any QP problem can be transformed into the form of (2). In this sense, (2) is regarded as a standard form of QP problems.

The set of optimal solutions of a convex QP problem is completely characterized by the Karush-Kuhn-Tucker (KKT) conditions [5]. In case of (2), a feasible solution x is an optimal solution if and only if there exist $\lambda = [\lambda_1, \lambda_2, \ldots, \lambda_m]^T$ and $\mu = [\mu_1, \mu_2, \ldots, \mu_n]^T$ such that the KKT conditions:

$$Qx + c + A^T\lambda - \mu = 0 \tag{3}$$

$$\mu \ge 0 \tag{4}$$

$$\mu_i x_i = 0, \quad i = 1, 2, \ldots, n \tag{5}$$

are satisfied. If $x_i > 0$ then μ_i must be zero from (5). In this case, the i-th component of (4) is necessarily satisfied, and the i-th component of (3) is written as $[Qx]_i + c_i + [A^T\lambda]_i = 0$, where $[Qx]_i$ ($[A^T\lambda]_i$, resp.) represents the i-th component of the vector Qx ($A^T\lambda$, resp.). If $x_i = 0$ then (5) apparently holds, and the conditions (3) and (4) for the i-th component are rewritten as $[Qx]_i + c_i + [A^T\lambda]_i \geq 0$. Therefore the KKT conditions (3)–(5) can be rewritten as

$$[Qx]_i + c_i + [A^T\lambda]_i \begin{cases} = 0, & \text{if } x_i > 0 \\ \geq 0, & \text{if } x_i = 0 \end{cases}, \quad i = 1, 2, \ldots, n \quad (6)$$

3. Proposed Algorithm

3.1. Algorithm

A new decomposition algorithm we propose in this paper is as follows:

---

1. Find a feasible solution of Problem (2). Let this feasible solution be $x^{(0)}$. Set $k := 0$.
2. Set $g^{(k)} := Qx^{(k)} + c$.
3. Solve the following LP problem:

$$\begin{array}{ll} \text{Minimize} & \delta \\ \text{Subject to} & \left|g_i^{(k)} + [A^T\lambda]_i\right| \leq \delta, \quad \forall i \in I_+^{(k)} \\ & g_i^{(k)} + [A^T\lambda]_i \geq -\delta, \quad \forall i \in I_0^{(k)} \end{array} \quad (7)$$

   where $I_+^{(k)} = \{i \,|\, x_i^{(k)} > 0\}$ and $I_0^{(k)} = \{i \,|\, x_i^{(k)} = 0\}$. If the optimal value of δ is less than ϵ, then stop. Otherwise go to Step 4.
4. For $i = 1, 2, \ldots, n$, set

$$v_i^{(k)} := \begin{cases} \left|g_i^{(k)} + [A^T\lambda^*]_i\right|, & \text{if } i \in I_+^{(k)} \\ -\min(0, g_i^{(k)} + [A^T\lambda^*]_i), & \text{if } i \in I_0^{(k)} \end{cases} \quad (8)$$

   where (λ^*, δ^*) is the optimal solution of (7) obtained in Step 3. Sort these values in decreasing order as

$$v_{i_1}^{(k)} \geq v_{i_2}^{(k)} \geq \cdots \geq v_{i_n}^{(k)}.$$

   Set $I_B^{(k)} := \{i_s\}_{s=1}^q$ and $I_N^{(k)} := \{i_s\}_{s=q+1}^n$.
5. Solve (2) under the additional constraints

$$x_{i_s} = x_{i_s}^{(k)}, \quad \forall i \in I_N^{(k)},$$

   and set $x^{(k+1)}$ to the obtained optimal solution.
6. Add 1 to k and go to Step 2.

In Step 1, a feasible solution of (2) is found by solving the LP problem:

$$\begin{array}{ll} \text{Minimize} & f(x) = 0 \\ \text{Subject to} & Ax = b \\ & x \geq 0 \end{array} \quad (9)$$

which has n variables, m equality constraints, and n inequality constraints.

Both the stopping condition and the working set selection in the proposed algorithm are based on the KKT condition (6). In order to determine whether there exists λ satisfying the KKT condition, the LP problem (7) is solved, which has $m + 1$ variables and at most $2n$ inequality constraints. It is apparent that (7) has an optimal solution (δ^*, λ^*) such that $\delta^* = 0$ if and only if the KKT condition is satisfied. However, we employ an approximate optimality condition $\delta^* < \epsilon$ for the stopping condition, due to the rounding error in numerical computations. The strategy of the working set selection is to choose q variables which most violate the optimality condition. This is used in most decomposition methods for the training of SVMs. In the proposed algorithm, the degree of violation for each variable $x_i^{(k)}$ is represented by $v_i^{(k)}$ and calculated as (8).

Let us assume without loss of generality that $I_B = \{1, 2, \ldots, q\}$. Then the optimization problem in Step 5 is expressed as

$$\begin{array}{ll} \text{Minimize} & \frac{1}{2}x_B^T Q_{BB} x_B + (Q_{BN}x_N^{(k)} + c_B)^T x_B \\ \text{Subject to} & A_B x_B = b - A_N x_N^{(k)} \\ & x_B \geq 0 \end{array} \quad (10)$$

where x_B, $x_B^{(k)}$ and c_B are the vectors composed of the first q components of x, $x^{(k)}$ and c, respectively; Q_{BB} is the $q \times q$ matrix composed of the first q rows and the first q columns of Q; Q_{BN} is the $q \times (n - q)$ matrix composed of the first q rows and the last $n - q$ columns of Q; A_B and A_N are the first q columns and the last $n - q$ columns of A, respectively. Since (10) is a QP problem with q variables, it can be solved much faster than the original problem (2).

3.2. Convergence Issue

It is easily seen from the constraints of (10) that $x^{(k)}$ satisfies the constraints of (2) for all k. Also, it is apparent that $f(x^{(k)})$ is monotone decreasing with k. However, these facts are not sufficient to guarantee the convergence or the finite termination of the proposed algorithm. Nevertheless, the authors conjecture that the proposed algorithm always stops within a finite number of iterations after finding an optimal solution, because no counterexample has been found in a number of experiments carried out by the authors. As for the decomposition methods for the training of SVMs, the convergence properties have been well studied and some sufficient conditions for ensuring the convergence have been derived [6, 7]. Analytical techniques used in these studies may also be useful for the convergence analysis of the proposed method.

4. Experiments

In order to evaluate the effectiveness of our decomposition algorithm, we compare the computation time of the

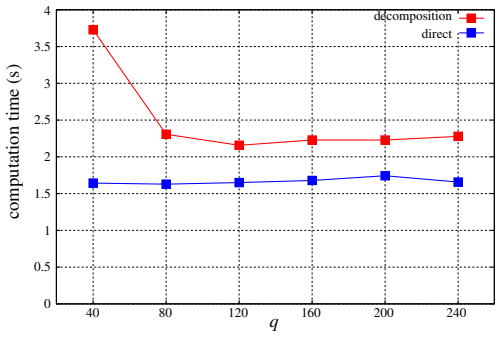

Figure 1: Computation time of the direct method and the proposed method for $n = 400$ and $m = 10$.

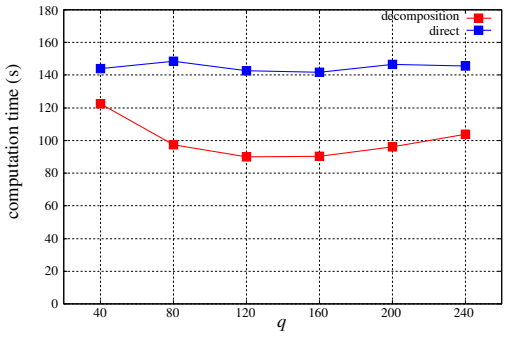

Figure 2: Computation time of the direct method and the proposed method for $n = 1400$ and $m = 10$.

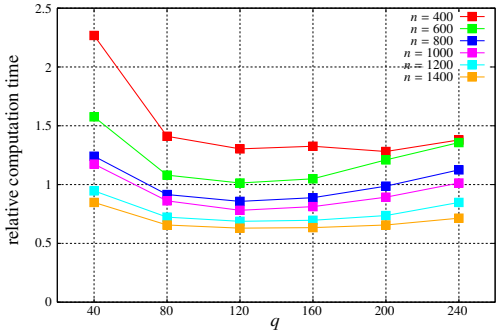

Figure 3: Relative computation time of the proposed method for $m = 10$.

proposed method with that of the direct method for various kinds of QP problems of the form (2). By the direct method we mean that (2) is solved with a conventional QP problem solver. We have implemented both the proposed method and the direct method in Scilab 4.1.2, which has the function "linpro" for solving LP problems and the function "quapro" for solving QP problems. Programs were run on a PC with Intel Celeron 2.66GHz and 512MB RAM.

QP problems were randomly generated as follows. Components of the matrix A and the vectors c and b were set to random numbers between -1 and 1. The matrix Q was generated by $Q = P^T P$ so that Q becomes positive semi-definite, where components of the matrix P were set to random numbers between -1 and 1.

First, we measured the computation time of the direct method and the decomposition method for QP problems with $n = 400$, $m = 10$ and $q \in \{40, 80, 120, 160, 200, 240\}$. Results are shown in Fig.1 The direct method is faster than the decomposition method for all q.

Next, we measured the computation time of the direct method and the decomposition method for QP problems with $n = 1400$, $m = 10$ and $q \in \{40, 80, 120, 160, 200, 240\}$. Results are shown in Fig.2 This time, the decomposition method is faster than the direct method for all q. In particular, the computation time is reduced to about 64% by using the decomposition method when $q = 160$. These results show that the decomposition method in fact works effectively.

It is seen from Figs.1 and 2 that the effectiveness of the proposed method depends heavily on n, the number of variables. So we next measured the relative computation time, which is defined as the computation time of the proposed method divided by that of the direct method, for $n = 400, 600, 800, 1000, 1200, 1400$. Results are shown in Fig.3. It is clearly seen that the relative computation time decreases as n increases. When $n = 400$ or $n = 600$, the relative computation time is greater than 1 for any q, which means that the direct method is faster than the proposed method. When $n = 800$ or $n = 1000$, the relative computation time is less than 1 for some values of q. When $n = 1200$ and $n = 1400$, the relative computation time is less than 1 for any q.

So far, we have fixed the value of m to 10. However, m is also an important factor that determines the computation time of the proposed method, because our decomposition algorithm has to solve an LP problem with $m + 1$ variables in each iteration. In order to investigate how the value of m affects the computation time of the decomposition algorithm, we measured the computation time for $n = 1000$, $q = 160$ and $m \in \{10, 20, 30, 40\}$. Results are shown in Fig.4. The decomposition method is faster than the direct method for $m = 10$ and $m = 20$, but slower for $m = 30$ and $m = 40$. From this observation, we can say that the the proposed algorithm is less effective for large m.

Finally, in order to investigate which part of the algorithm is the most time consuming, we have measured the computation time of Step 1, where an feasible solution is found by solving (9), for $n = 1400$, $m = 10$ and $q \in \{40, 80, 120, 160, 200, 240\}$. Results are shown in Fig.5. The computation time of Step 1 is much longer than that of the remaining part. Therefore, if we solve (9) by the decomposition method, the total computation time may be reduced further.

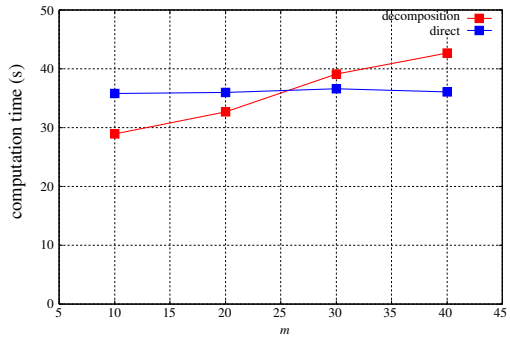

Figure 4: Relationship between computation time of the proposed method and m for $n = 1000$ and $q = 160$.

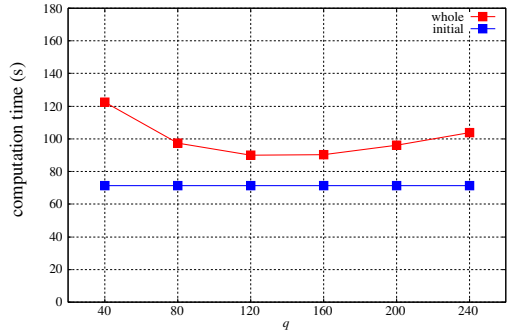

Figure 5: Total computation time and the computation time for finding an initial feasible solution for $n = 1400$ and $m = 10$.

5. Conclusions

A new decomposition method that can be applied to general convex QP problems was proposed. According to experimental results, the proposed method is faster than the standard QP problem solver when the number of variables is large and the number of equality constraints is small. The reduction of the computation time of Step 1 is a future problem.

Acknowledgments

This work was supported in part by Kyushu Industrial Technology Center and Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology, Japan.

References

[1] V. Vapnik, *Statistical Learning Theory*, Wiley, New York, 1998.

[2] E. Osuna, R. Freund, and F. Girosi, "An improved training algorithm for support vector machines," in *Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing*, 1997, pp. 511–519.

[3] J. C. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges, and A. Smola, Eds., Cambridge, MA: MIT Press, 1998.

[4] T. Joachims, "Making large-scale support vector machine learning practical," in *Advances in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges, and A. Smola, Eds., Cambridge, MA: MIT Press, 1998.

[5] M. S. Bazaraa, H. D. Sherali and C. M. Shetty, *Nonlinear Programming: Theory and Algorithms*, John Wiley & Sons, Hoboken, NJ, 1993.

[6] S. S. Keerthi and E. G. Gilbert, "Convergence of a generalized SMO algorithm for SVM classifier design," *Machine Learning*, vol. 46, pp.351–360, 2002.

[7] N. Takahashi and T. Nishi, "Global convergence of decomposition learning methods for support vector machines," *IEEE Trans. Neural Networks*, vol. 17, no. 6, pp. 1362–1369, Nov. 2006.

A. Transformation of QP problems

We will show that any QP problem of the form (1) can be transformed into the form of (2). First, by introducing the variable $y \in \mathbb{R}^l$, the inequality constraint in (1) is transformed as

$$Ex + y = d, \quad y \geq 0\,.$$

Next, we express the variable $x \in \mathbb{R}^n$ with two nonnegative variables $u \in \mathbb{R}^n$ and $v \in \mathbb{R}^n$ as

$$x = u - v, \quad u \geq 0, \quad v \geq 0\,.$$

Finally, by defining $\tilde{x} = [u^T, v^T, y^T]^T \in \mathbb{R}^{2n+l}$, we can rewrite (1) as

$$\begin{array}{ll} \text{Minimize} & \tilde{f}(\tilde{x}) = \frac{1}{2}\tilde{x}^T\tilde{Q}\tilde{x} + \tilde{c}^T\tilde{x} \\ \text{Subject to} & \tilde{A}\tilde{x} = \tilde{b} \\ & \tilde{x} \geq 0 \end{array}$$

which has the same form as (2), where $\tilde{Q} \in \mathbb{R}^{(2n+l)\times(2n+l)}$, $\tilde{c} \in \mathbb{R}^{2n+l}$, $\tilde{A} \in \mathbb{R}^{m+l}$ and $\tilde{b} \in \mathbb{R}^{m+l}$ are given by

$$\tilde{Q} = \begin{bmatrix} Q & -Q & O \\ -Q & Q & O \\ O & O & O \end{bmatrix}, \quad \tilde{c} = \begin{bmatrix} c \\ -c \\ 0 \end{bmatrix},$$

$$\tilde{A} = \begin{bmatrix} A & -A & O \\ E & -E & I \end{bmatrix}, \quad \tilde{b} = \begin{bmatrix} b \\ d \end{bmatrix}$$