ADDIS ABABA UNIVERSITY
School of Graduate Studies
Department of Mathematics

Graduate Seminar Report

ON

WOLFE AND LEMKE ALGORITHMS FOR QUADRATIC PROGRAMMING

(Submitted in partial fulfillment of M.Sc. Degree in Mathematics)

Compiled by: Hailemariam Abebe

Advisor: Dr. Berhanu Guta

June, 2010

Addis Ababa

Acknowledgement

I would like to express my sincere gratitude to my advisor Dr. Berhanu Guta for his genuine advice, constructive comments and for provision of material in preparing this seminar report.

I also extend my deep appreciation to the Department of Mathematics for its material support and giving me the chance to type the manuscript of this paper by myself. Again many thanks to all friends who gave me moral support and wish my progress.

Finally, my thanks go to Gondar University for the financial support in attending the master degree program.

Table of Contents

CHAPTER 1

INTRODUCTION

This seminar report focuses on Wolfe and Lemke algorithms which are used to solve quadratic programming problems. Solving quadratic programming problems using only these methods is difficult. So they work in existence with Beale algorithm to reach an optimal solution with in a finite iteration.

This chapter introduces the basic concepts of quadratic programming problems, its optimality conditions and solution methods as well as theorems which are needed later for a better understanding of the next chapters.

1.1 Quadratic programming

Quadratic programming (QP) is a special type of mathematical optimization problem. It is the problem of optimizing (minimizing or maximizing) a quadratic function of several variables subject to linear constraints on these variables. Quadratic programming problems may be stated in many equivalent forms. We define the standard QP problem in the form

$$\begin{aligned} &\textit{Minimize} \quad f(x) = c^T x + \frac{1}{2} x^T Q x \\ &\textit{subject to}: \quad Ax \le b \\ &\qquad\qquad x \ge 0 \end{aligned} \tag{1.1}$$

where x and c are $n \times 1$ vectors, A is $m \times n$ matrix, and Q is $n \times n$ symmetric matrix. The scalar term $\frac{1}{2} x^T Q x$ is a quadratic form.

Quadratic programming problems are typically classified as being convex or non-convex, depending on whether Q is positive semi- definite or indefinite. In addition, we shall say that a quadratic programming problem is strictly convex if Q is positive definite. A quadratic objective function can be maximized by minimizing its negative. Inequality constraints can be converted to equations by the addition of nonnegative slack variables.

If Q is a positive semidefinite matrix, the quadratic program has a global minimizer if there exists at least one vector x satisfying the constraints and objective function which is bounded below on the feasible region. If the matrix Q is positive definite then this global minimizer is unique. If Q is zero, then the problem becomes a linear program.

1.2 Optimality Conditions

In this section we discuss on results which gives emphasis on convexity of quadratic programming and their optimality conditions for quadratic programming. The Karush–Kuhn–Tucker(KKT) point is essential to determine optimality of a given QP problem. From optimization theory, a necessary condition for a point x_0 to be a global minimizer is for it to satisfy the KKT conditions. These conditions are also sufficient when objective function is convex. Before this, we define terms which are useful for the discussion.

Definition 1.2.1: Let $f : S \to R$ be a function where S is nonempty convex in R^n.

(a) The function f is said to be convex if for each $x_1, x_2 \in S$ $f(\lambda x_1 + (1-\lambda)x_2) \le \lambda f(x) + (1-\lambda) f(x_2)$ *for each* $\lambda \in (0,1)$. f is said to be concave if $-f$ is convex.

(b) The function f is said to be pseudoconvex if for each $x_1, x_2 \in S$ (S is open) with $\nabla f(x_1)^T (x_2 - x_1) \ge 0$ we have $f(x_2) \ge f(x_1)$.

Definition 1.2.2: Let $f : R^n \to R$, and consider the problem to minimize $f(x)$ subject to $x \in S$. If $x_0 \in S$ and, $f(x_0) \le f(x)$, then x_0 is called optimal solution or global minimum. If there existsan $\in$ *–neighborhood* $N_\epsilon(x_0)$ around x_0 such that $f(x_0) \le f(x)$ for all x in $N_\epsilon(x_0) \cap S$, then x_0 is called a local minimum.

Definition 1.2.3: Consider the QP problem (1.1). If x_0 is an optimal solution of (1.1) then, for some Lagrange multipliers $\lambda \in \Re_+^m$ the following KKT conditions hold:

$$c + Qx_0 + A^T\lambda \geq 0 \qquad \textit{dual feasiblity}$$
$$Ax_0 - b \leq 0 \qquad \textit{primal feasiblity}$$
$$x_0^T(c + Qx_0 + A^T\lambda) = 0 \qquad \textit{complementary slackness}$$
$$\lambda^T(Ax_0 - b) = 0. \qquad \textit{complementary slackness}$$

If x_0 together with $\lambda \in \Re_+^m$ satisfy the KKT conditions, then x_0 is called a KKT point for (1.1).

Theorem 1.2.4: The necessary and sufficient conditions for an x_0 to be an optimal solution to the quadratic programming problem (1.1), where $f(x)$ is convex, is that satisfies the KKT conditions.

Proof: Since the constraints of the problem (1.1) are linear, the KKT constraint qualification is automatically satisfied. To show x_0 is optimal solution, let x_0 be KKT point to (1.1) with corresponding Lagrange multiplier λ^* and since $f(x)$ is convex, then the Lagrange function

$$L(x_0, \lambda^*) = c^T x_0 + \tfrac{1}{2} x_0^T Q x_0 + \lambda^{*T}(Ax_0 - b), \quad (x_0, \lambda^*) \in \Re_+^n \times \Re_+^m$$

is convex and $L_x(x_0, \lambda^*) = 0$. Then by Lagrange lemma [1], we have x_0 is an optimal solution.

Theorem 1.2.5: Let S be a nonempty open convex set in R^n, and let $f : S \to R$ be twice differentiable on S. Then, f is convex if and only if the Hessian matrix Q is positive semidefinite at each point in S.

Proof: Suppose that f is convex and let $x_0 \in S$. We need to show that $x^T Q(x_0)x \geq 0$ for each $x \in R$. Since S is open, then for any given $x \in R^n$, $x_0 + \lambda x \in S$ *for* $|\lambda| \neq 0$ and

sufficiently small, we get the two expression from its convexity and by second order Taylor series.

$$f(x_o + \lambda x) \geq f(x_0) + \lambda \nabla f(x_0)^T x \tag{1.2}$$

$$f(x_o + \lambda x) = f(x_0) + \lambda \nabla f(x_0)^T x + \tfrac{1}{2}\lambda^2 x^T Q(x_0) x + \lambda^2 \|x\|^2 \alpha(x_0; \lambda x). \tag{1.3}$$

Subtracting (1.3) from (1.2) we get

$$\tfrac{1}{2}\lambda^2 x^T Q(x_0) x + \lambda^2 \|x\|^2 \alpha(x_0; \lambda x).$$

Dividing by $\lambda^2 \succ 0$ and letting $\lambda \to 0$, it follows that

$$x^T Q(x_0) x \geq 0$$

Therefore, Q is positive semidefinite.

1.3 Solution Methods

There are several approaches of solving quadratic programming. Among these, the methods of Wolfe and Lemke algorithms are the best known and mostly widely used quadratic programming algorithms in existence with Beale algorithm.

One of the earliest and simplest methods for solving quadratic program was that proposed by Philip Wolfe in 1952; despite its rather venerable history, it is still quite commonly used today. In 1955 E.M.L. Beale published a paper in a British journal describing a method for solving a quadratic programs that was based on arguments from classical calculus rather than on the KKT conditions. The paper initially attracted little attention, especially in the United States, so an amplified version of it was published in its country in 1959. This second effort was more favorably recieved, and Beale's method has since become quite well known, although certainly not so widely used as that of Wolfe.

Another quadratic programming algorithm, that of Lemke appeared originally in 1962, which is rather more sophisticated and elegant than either of the two.

CHAPTER 2

WOLFE ALGORITHM

In this section we discuss on the solution strategy called Wolfe Method which is able to modify the simplex method in a way that enables it to solve quadratic programs by finding feasible points that satisfy KKT conditions.

2.1 Basic Concepts and Algorithm

Consider the following QP problem

$$\begin{aligned} &\textit{Minimize} \quad f(x) = c^T x + \frac{1}{2} x^T Q x \\ &\textit{subject to}: \quad Ax \le b \\ &\qquad\qquad \textit{and} \quad x \ge 0 \end{aligned} \tag{2.1}$$

where x and c are n component column vectors, A is m×n matrix, b is m×1, and Q is n×n symmetric matrix. The basic approach of the algorithm is to generate, by means of a modified simplex pivoting procedure, a sequence of feasible points that terminates at a solution point x_0 where the KKT conditions are satisfied.

Then we have Lagrange function

$$L(x,\lambda) = c^T x + \tfrac{1}{2} x^T Q x + \lambda^T (Ax - b), \quad (x,\lambda) \in \Re_+^{\,n} \times \Re_+^{\,m}.$$

Now we get the KKT conditions as

$$L_x(x_0,\lambda) = c + Qx_0 + A^T\lambda \ge 0 \tag{2.2}$$

$$Ax_0 - b \le 0 \tag{2.3}$$

$$x_0^{\,T}(c + Qx_0 + A^T\lambda) = 0 \tag{2.4}$$

$$\lambda^T(Ax_0 - b) = 0 \tag{2.5}$$

$$x_0 \ge 0, \lambda \ge 0 \tag{2.6}$$

By introducing of slack variables u and v, we get

$$c + Qx_0 + A^T\lambda - u = 0 \tag{2.7}$$

$$Ax_0 - b + v = 0 \tag{2.8}$$

$$x_0{}^T(c + Qx_0 + A^T\lambda) = 0 \tag{2.9}$$

$$\lambda^T(Ax_0 - b) = 0 \tag{2.10}$$

$$x_0 \geq 0, \lambda \geq 0, u \geq 0, v \geq 0 \tag{2.11}$$

The equations (2.7) and (2.8) form a system of linear equations for the variable x_0 and λ which can be solved by the simplex algorithm subject to the conditions (2.9) and (2.10).

From (2.7) we get

$$u = c + Qx_0 + A^T\lambda \ .$$

Then we get from (2.9)

$$x_0{}^T u = \sum_{i=1}^{n} x_i^0 u_i^0 = 0 \text{ where } x_0{}^T = (x_1^0, x_2^0, ..., x_n^0), u^T = (u_1, u_2, ..., u_n) .$$

Then we get

$$x_i^0 u_i = 0, \quad i \in \{1, 2, ..., n\} \ \ by\ x_0 \geq 0\ and\ u \geq 0 .$$

Similarly from (2.8) and (2.10) we get

$$\lambda_i v_i = 0, \qquad i \in \{1, 2, ..., m\}\ by\ \lambda \geq 0\ and\ \ v \geq 0 .$$

If x_i is a basic variable, i.e $x_i>0$, then $u_i=0$ by $x_iu_i=0$. But this means u_i cannot be taken as basic variable. Analogously, if u_i is a basic variable, i.e. $u_i>0$, then $x_i=0$ cannot be taken as basic variable. The same consideration we have for λ_i and v_i.

These are now the important conditions which we have to consider inorder to solve the equations (2.7) and (2.8) by the simplex algorithm.

Now consider the following simplex method for Ay=b where $y=(y_B,y_N)$ is variable containing basic and non basic. The steps are as follows

1. *Locate smallest negative* $\Delta z_j = -d_j + \sum_{i=1}^{m} d_{Bi} a_{ij}$, *where i is* i^{th} *row and j is* j^{th} column and d is associated values , excluding last column of b.
2. From $\theta = \min\left\{\frac{b_i}{a_{ij}}\right\}$, where i is i^{th} row and j is j^{th} column, excluding last column of b. Designate the pivot element i.e. element which is found at rows of θ and column of rows of Δz. If no element in θ column is positive, the program has no solution.
3. Use elementary row operation to convert the pivot element to (1) and then to reduce all other elements.
4. Repeat step (1) through (3) until there are no negative numbers in the last row i.e. $\Delta z_j = -d_j + \sum_{i=1}^{m} d_{Bi} a_{ij} \geq 0$ *for all j*. Optimal solution at the last row and last column will be achieved.

Steps of Wolfe Algorithm:

1. Formulate KKT conditions (2.2) to (2.6) for the given problem.
2. Introduce slack variables in (2.2) and (2.3) inorder to have equations. Check, whether there is a feasible basis. If there is a feasible basis, then the corresponding basic solution is optimal. If there is no a feasible basis, then go to step 3.
3. Introduce artificial variables inorder to get a feasible basis. Go to step 4.
4. Iterating by means of the big-M simplex method until a feasible basic solution is found or until verifying that there is no feasible solution. For changing of variables we have restrictions:
   (i) A nonbasic variable x_j may become new basic variable only,if the corresponding slack variable u_j is not already a basic variable and conversely.

(ii) A nonbasic variable λ_i may become new basic variable only, if the corresponding slack variable v_i is not already a basic variable and conversely.

Example: Consider the following QP problem:

$$\begin{aligned} &\textit{Minimize} \quad f(x) = x_1^2 + 2x_2^2 - 2x_1x_2 - 3x_1 - 4x_2 \\ &\textit{subject to}: \quad x_1 + 2x_2 \le 7 \\ &\qquad\qquad -x_1 + 2x_2 \le 4 \\ &\qquad\qquad x_1 \ge 0, x_2 \ge 0. \end{aligned}$$

Obviously we have here

$$c = \begin{bmatrix} -3 \\ -4 \end{bmatrix}, b = \begin{bmatrix} 7 \\ 4 \end{bmatrix}, Q = \begin{bmatrix} 2 & -2 \\ -2 & 4 \end{bmatrix} \text{ and } A = \begin{bmatrix} 1 & 2 \\ -1 & 2 \end{bmatrix}.$$

Then the Lagrange function is given by

$$L(x,\lambda) = f(x) + \lambda_1(x_1 + 2x_2 - 7) + \lambda_2(-x_1 + 2x_2 - 4), \quad (x,\lambda) \in \Re_+^2 \times \Re_+^2.$$

Step 1: As KKT conditions we get

$$\begin{aligned} 2x_1 - 2x_2 - 3 + \lambda_1 - \lambda_2 &\ge 0 \\ -2x_1 + 4x_2 - 4 + 2\lambda_1 + 2\lambda_2 &\ge 0 \\ x_1 + 2x_2 &\le 7 \\ -x_1 + 2x_2 &\le 4 \\ x_1(2x_1 - 2x_2 - 3 + \lambda_1 - \lambda_2) &= 0 \\ x_2(-2x_1 + 4x_2 - 4 + 2\lambda_1 + 2\lambda_2) &= 0 \\ \lambda_1(x_1 + 2x_2 - 7) &= 0 \\ \lambda_2(-x_1 + 2x_2 - 4) &= 0 \\ \lambda_1 \ge 0, \lambda_2 \ge 0 \quad x_1 \ge 0, x_2 &\ge 0. \end{aligned}$$

Step 2: By introducing of slack variables we get

$$\begin{aligned} 2x_1 - 2x_2 - 3 + \lambda_1 - \lambda_2 - u_1 &= 0 \\ -2x_1 + 4x_2 - 4 + 2\lambda_1 + 2\lambda_2 - u_2 &= 0 \\ x_1 + 2x_2 + v_1 &= 7 \\ -x_1 + 2x_2 + v_2 &= 4. \end{aligned}$$

Step 3: Inorder to get a feasible basis we have to introduce artificial variables h_1 and h_2 for the first and the second row.

Step 4: Now applying the Big-M simplex technique to this example, yields the sequence of simplex iterations given in the following Table.

Iteration	Basic variables	Solution	Objective value	Entering variable	Leaving variable
1	(h_1, h_2, v_1, v_2)	(3, 4, 7, 4)	0	x_2	h_2
2	(h_1, x_2, v_1, v_2)	(5, 1, 5, 2)	-2	x_1	v_1
3	(h_1, x_2, x_1, v_2)	(5/2,9/4,5/2,2)	-11.375	λ_1	h_1
4	$(\lambda_1, x_2, x_1, v_2)$	(1, 2, 3, 3)	-12	—	—

Hence, from the last simplex iteration we have a solution $(x_1,x_2)=(3,2)$, $(\lambda_1,\lambda_2)=(1,0)$, $(u_1,u_2)=(0,0)$, $(v_1,v_2)=(0,3)$. With these values since all the KKT conditions are satisfied and since the optimization problem is convex we have that $(x_1,x_2)=(3,2)$ is an optimal solution.

2.2 Convergence of Wolfe Algorithm

Before attempting to prove that whenever Q is positive semidefinite Wolfe's algorithm will eventually find a feasible point satisfying the KKT conditions, we must first demonstrate that such a point actually exists. To obtain this result we need only show that when Q is positive semidefinite the problem (2.1) cannot have an unbounded minimum. It will then follow that some feasible point must be a global minimum and therefore a constrained local minimum at which the KKT conditions necessarily hold.

When Wolfe's algorithm is applied to a QP problem of the form (2.1), with Q is positive definite or positive semidefinite and c=0, it will obtain with in a finite number of iterations(barring degeneracy) a feasible solution at which KKT conditions are satisfied. Because if Q is is positive semidefinite, then (2.1) is a convex program, and sufficiently of the KKT conditions thus guarantees that x_0 is an optimal solution.

When Q is indefinite or negative semidefinite, convergence may fail entrely, or if it occurs, the solution obtained may not even be a local minimum. Hence with some difficulties when this condition happens, we may apply another solution method that of Beale.

2.3 Beale Method

Beale has developed a method for solving quadratic programming problems which is based on the basic principless of classical calculus rather than the KKT conditions.Beale algorithm is directly applicable to any QP of the following form:

$$\begin{aligned} &\textit{Minimize} \quad f(x) = c^T x + \frac{1}{2} x^T Q x \\ &\textit{subject to}: \quad Ax = b \\ &\qquad\qquad x \geq 0 \end{aligned} \tag{2.12}$$

where A is m×n matrix , b is m×1, c is n×1 , x is n×1 and Q is n×n symmetric. We do not require Q to be positive semidefinite i.e. we don't require the objective function to be convex. The algorithm generally yield a local minimum of a nonconvex quadratic function, ofcourse if the objective function is convex, the local solution obtained will be global. It is assumed that the problem is non-degenerate.

Beale's method is an iterative procedure and begins with any basic feasible solution of the problem; this can be done via the phase I simplex procedure, if necessary. Let A be partitioned as A=(B,N), where B is the basis matrix which for the convenienc is assumed to consist of the first m-columns of A and N is the matrix consisting of the remaining n-m columns of A. Also let x_B and x_N be the vectors of basic and non basic variables. The constraints Ax=b, can then be written as $Bx_B+Nx_N=b$ and $x_B=B^{-1}b-B^{-1}Nx_N$. The basic variable can thus be written intermas of nonbasic variables $z_q = x_{m+q}$, q=1,2,...,n-m as

$$x_h = \alpha_{h0} + \sum_{q=1}^{n-m} \alpha_{hq} z_q .$$

Steps of Beale Algorithm:

1. Obtain a basic feasible solution of the problem i.e. any basic feasible solution.
2. Express the basic variables x_h, $h = 1,2,...,m$ interms of nonbasic variables $z_q = x_{m+q}$, q=1,2,...n-m. Then we will have ,

$$x_h = \alpha_{h0} + \sum_{q=1}^{n-m} \alpha_{hq} z_q$$

3. Express the objective function $f(x)$ in terms of the non basic variables in the symmetric form

$$f(x) = \sum_{k=0}^{n-m}\sum_{q=0}^{n-m} \gamma_{kq} z_k z_q$$

$$= \gamma_{00} + 2\sum_{k=1}^{n-m} \gamma_{kq} z_k + \sum_{k=1}^{n-m}\sum_{q=1}^{n-m} \gamma_{kq} z_k z_q,$$

$$where\ z_0 = 1\ and\ \gamma_{kq} = \gamma_{qk}.$$

4. Consider the partial derivative of $f(x)$ with respect to any one of nonbasic variables say z_k.

$$\frac{1}{2}\frac{\partial f}{\partial z_k} = \gamma_{k0} + \sum_{q=1}^{n-m} \gamma_{kq} z_q$$

with all other non basic variables held at zero.

(a) If $\frac{\partial f}{\partial z_k} \geq 0$ for all k, the current solution is optimal.

(b) If $\frac{\partial f}{\partial z_k} < 0$ for at least one k, increase z_k until either

(i) Some basic variables becomes zero, or

(ii) $\frac{\partial f}{\partial z_k}$ vanishes and is about to become positive.

5. To determine the change in the basis, calculate

$$Min\left[\frac{\alpha_{h0}}{|\alpha_{hk}|}, \frac{|\gamma_{k0}|}{\gamma_{kk}}, h = 1,2,...,m\right] \text{ for } a_{hk} < 0, \text{ and } \gamma_{kk} > 0.$$

(i) If the minimum occurs for some h=v, then x_v becomes nonbasic.

(ii) If the minimum occurs for the second term, introduce a new non basic variable u_t, defined by

$$u_t = \frac{1}{2}\frac{\partial f}{\partial z_k} = \gamma_{k0} + \sum_{q=1}^{n-m} \gamma_{kq} z_q$$

u_t is unrestricted and is called a free variable. This will lead to one equation and one more basic variable than before.

6. Go to step 2 and repeat the process until $\frac{\partial f}{\partial z_k} \geq 0$ for all k and $\frac{\partial f}{\partial u_t} = 0$ for all free variables.
7. Obtain the local (global if f is convex) optimal solution and the value of min f by setting non basic variables equal to zero in their expressions.

Example: Consider the following QP problem:

$$\begin{aligned} \textit{Minimize} \quad & f(x) = -x_1 - 2x_2 + x_2^2 \\ \textit{subject to}: \quad & x_1 + 2x_2 \leq 4 \\ & 3x_1 + 2x_2 \leq 6 \\ & x_1 \geq 0, x_2 \geq 0 \end{aligned}$$

By introducing slack variables, we have the problem

$$\begin{aligned} \textit{Minimize} \quad & f(x) = -x_1 - 2x_2 + x_2^2 \\ \textit{subject to}: \quad & x_1 + 2x_2 + x_3 = 4 \\ & 3x_1 + 2x_2 + x_4 = 6 \\ & x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0. \end{aligned}$$

Now to solve this problem using Beale method:

Step 1: Take x_3 and x_4 as basic variables.

Step 2: Express them in terms of nonbasic variables as

$$x_3 = 4 - x_1 - 2x_2 \quad \text{and} \quad x_4 = 6 - 3x_1 - 2x_2$$

Step 3: $f(x) = -x_1 - 2x_2 + x_2^{\,2}$

Step 4: $\frac{1}{2}\frac{\partial f}{\partial x_1} = -\frac{1}{2}$ *and* $\frac{1}{2}\frac{\partial f}{\partial x_2} = -1$

Step 5: Hence it is profitable to increase x_2 and we calculate

$$Min\left[\frac{a_{30}}{|a_{32}|}, \frac{a_{40}}{|a_{42}|}, \frac{|\gamma_{20}|}{\gamma_{22}}\right] = Min\{\frac{4}{2}, \frac{6}{2}, \frac{1}{1}\} = Min\{2,3,1\} = 1.$$

We therefore introduce a free nonbasic variable

$$u_1 = \frac{1}{2}\frac{\partial f}{\partial x_2} = -1 + x_2$$

We again express basic variables in terms of nonbasic variables

$$\begin{aligned} x_2 &= 1 + u_1 \\ x_3 &= 4 - x_1 - 2(1 + u_1) = 2 - x_1 - 2u_1 \\ x_4 &= 6 - 3x_1 - 2(1 + u_1) = 4 - 3x_1 - 2u_1 \\ f(x) &= -x_1 - 2(1 + u_1) + (1 + u_1)^2 \\ &= -1 - x_1 + u_1^{\,2}. \end{aligned}$$

Now, $\frac{1}{2}\frac{\partial f}{\partial u_1} = 0$, $\frac{1}{2}\frac{\partial f}{\partial x_1} = -\frac{1}{2}$. Hence x_1 is therefore increased and we calculate:

$$Min\left\{\frac{a_{30}}{|a_{31}|}, \frac{a_{40}}{|a_{41}|}, \frac{|a_{20}|}{\gamma_{21}}, \frac{|\gamma_{10}|}{\gamma_{11}}\right\} = Min\{\frac{2}{1}, \frac{4}{3}\} = \frac{4}{3}, (a_{21} = 0, \gamma_{11} = 0).$$

Thus x_4 leaves the basis and x_1 enters. Then,

$$x_1 = \frac{4}{3} - \frac{2}{3}u_1 - \frac{1}{3}x_4$$

$$x_2 = 1 + u_1$$

$$x_3 = \frac{2}{3} - \frac{4}{3}u_1 + \frac{1}{3}x_4.$$

Step 6: Now from $f(x) = -\frac{7}{3} + \frac{2}{3}u_1 + \frac{1}{3}x_4 + u_1^2$, we get

$$\frac{1}{2}\frac{\partial f}{\partial u_1} = \frac{1}{3} \text{ and } \frac{1}{2}\frac{\partial f}{\partial x_4} = \frac{1}{6}.$$

Hence it is profitable to decrease u_1. We therefore introduce another free variable u_2, $u_2 = \frac{1}{3} + u_1$ and thus,

$$u_1 = -\frac{1}{3} + u_2$$

$$x_1 = \frac{14}{19} - \frac{2}{3}u_2 - \frac{1}{3}x_4$$

$$x_2 = \frac{2}{3} + u_2$$

$$x_3 = \frac{10}{9} - \frac{4}{3}u_2 + \frac{1}{3}x_4$$

$$f(x) = -\frac{22}{9} + \frac{1}{3}x_4 + u_2^2.$$

Now, $\frac{1}{2}\frac{\partial f}{\partial u_2} = 0$ *and* $\frac{1}{2}\frac{\partial f}{\partial x_4} = \frac{1}{6} > 0$.

Step 7: Here, the minimum solution is achieved:

$$x_1 = \frac{14}{19}, x_2 = \frac{2}{3} \text{ and } Min\, f = \frac{22}{9}.$$

2.4 Programming in MATLAB using Wolfe Algorithm

This matlab program is capable of solving a convex quadratic programming problem by Wolf's method. For the convergence of the algorithm it is necessary that either Hessian of the objective function be positive definite or positive semidefinite Hessian with linear term zero. From the following program we will obtain an output result after supplying the inputs and by calling wolf(Q,c,b,A,minimize). The function wolf solves a QP problem using Wolfe or restricted entry simplex method. The modified simplex Algorithm for solving the LP problem:

minimize c'*x+0.5*x'*Q*x
Subject to A*x <= b
x >= 0.

```
function [x,fval]=wolf(Q,c,b,A,minimize)

% Input:-
% Q : The Hessian of the objective function. It is a required parameter.
% c : The (column) vector of coefficient of linear terms in the objective function. It is a
%     required parameter.
% b : The (column) vector containing right hand side constant of the constraints. It is a
%     required parameter.
% A : The coefficient  matrix of the left hand side of the constraints. it is a required
%     parameter.
% minimize : This parameter indicates whether the objective function is to be minimized.
%            minimize = 1 indicates a mimization problem and minimize = 0 stands for
%           a maximization problem. It is an optional parameter.

n = length(c);
m = length(b);

if ~isequal(size(A,1),m) || ~isequal(size(Q,1),...
      size(Q,2)) || ~isequal(size(Q,1),n) || ~isequal(size(A,2),n)
   fprintf('\nError: Dimension mismatch!\n');
   return
end

if nargin < 3 || nargin > 5
   fprintf('\nError:Number of input arguments are inappropriate!\n');
   return
end
```

```
if nargin < 5
   minimize = 0;
end

if minimize == 1
   c = -c;
   Q = -Q;
end

if min(eig(-Q)) < 0 % Checking convexity of Hessian
   fprintf('\nError: Wolf method may not converge to global optimum!\n');
   return
elseif (min(eig(-Q)) == 0) && ~isempty(find(l,1))
   fprintf('\nError: Wolf method may not converge to global optimum!\n');
   return
end

count = n;
a = [-1*Q A' -eye(count,count) zeros(count,m);A zeros(m,m + count) eye(m,m)];
d = [c;b];

for i = 1 : count + m
   if(d(i) < 0)
      d(i) = -d(i);
      a(i,:) = -a(i,:);
   end
end

cb = zeros(1,count + m);
bv = zeros(1,count + m);
nbv = (1 : 2 * (count + m));
c = zeros(1,2 * (count + m));
rem = zeros(1,count + m);

for i = 1 : count + m
   if(a(i,count + m + i) == -1)
      bv(i) = 2 * (count + m) + i;
      cb(i) = -1;
   elseif(a(i,count + m + i) == 1)
      rem(i)=count + m + i;
      bv(i) = count + m + i;
      cb(i) = 0;
   end
end
```

```
[h,j,k] = find(rem);
a(:,k) = [];
c(k) = [];
nbv(k) = [];
r = cb * a - c;
exitflg = 0;
iter = 0;
z = cb * d;
[w,y] = size(a);
opt = 0;

while(exitflg == 0)

  iter = iter + 1;
  fprintf('\n\n %d th tableau:\n',iter );% If you don't need
  fprintf('\n\t\t\tBV\t');disp(nbv);%intermedite tableaus be
  disp([bv' d a ;0 z r]);
  r_new = r;
  found = 0;

  while found == 0

    [u,v] = min(r_new);
    leave = 0;

    if ~(u < 0)
      if abs(z) > 10^-6
        fprintf('\nError: Wolf method fails to find optimum!\n');
        exitflg = 1;
        found = 1;
      else
        fprintf('\nThe optimum has achieved!\n');
        exitflg = 1; opt = 1;
        found = 1;
      end
    else

      ratio = bitshift(1,30);
      check = 0;

      for i = 1 : w
        if bv(i) <= 2 * (count +  m) && abs(bv(i) - nbv(v)) == count + m
          check = 1;
        end
```

```
  end

  if check == 0
    for i = 1 : w
      if a(i,v) > 0 && (d(i) / a(i,v)) < ratio
        ratio = d(i) / a(i,v);
        leave = i;
      end
    end
    fprintf('\nEntering Variable:'); disp(nbv(v));
    fprintf('\nLeaving Variable:'); disp(bv(leave));

      for i = 1 : w
      for j = 1 : y
        if i ~= leave && j ~= v
          a(i,j) = a(i,j) - a(i,v) * a(leave,j) / a(leave,v);
        end
      end
    end

    z = z - d(leave) * r(v) / a(leave,v);
    for j = 1 : y
      if j ~= v
        r(j) = r(j) - r(v) * a(leave,j) / a(leave,v);
        a(leave,j) = a(leave,j) / a(leave,v);
      end
   end
      for i = 1 : w
      if i ~= leave
        d(i) = d(i) - a(i,v) * d(leave) / a(leave,v);
        a(i,v) = -a(i,v) / a(leave,v);
      end
    end

    d(leave) = d(leave) / a(leave,v);
    a(leave,v) = 1 / a(leave,v);
    r(v) = -r(v) / a(leave,v);
    temp = nbv(v);
    nbv(v) = bv(leave);
    bv(leave) = temp;
    found = 1;
  elseif check == 1
    r_new(v) = 1;
  end
end
```

```
  end
end
if opt == 1
  x = zeros(n,1);
  for i = 1 : w
    if bv(i) <= n
      x(bv(i)) = d(i);
    end
  end
end
```

CHAPTER 3

LEMKE ALGORITHM

In 1968, Lemke proposed a complementary pivoting algorithm for solving linear complementary problems since the KKT conditions for QP can be written as a linear complementary problems.

3.1 Definitions and Basic Concepts

This section introduces some basic definitions of the linear complementary problems and KKT conditions for QP.

Definition 3.1.1: Let M be given n×n matrix and let q be agiven n vector. The linear complementary peoblem (LCP) is to find vectors w and z such that

$$w - Mz = q \tag{3.1}$$

$$w_j \geq 0, z_j \geq 0 \quad \text{for } j = 1,\ldots,n \tag{3.2}$$

$$w_j z_j = 0 \quad \text{for } j = 1,\ldots,n \tag{3.3}$$

or to conclude that no such solution exist. Here, (w_j,z_j) is apair of complementary variables. A solution (w,z) to the above system is called complementary feasible solution. Moreover, such a solution is complementary basic feasible solution if (w,z) is a basic feasible solution to (3.1) and (3.2) with one variable of the pair (w_j,z_j) basic for each j=1,...,n.

Definition 3.1.2: Consider the system defined below

$$w - Mz - ez_o = q \tag{3.4}$$

$$w_j \geq 0, z_j \geq 0, z_o \geq 0 \quad \text{for } j = 1,\ldots,n \tag{3.5}$$

$$w_j z_j = 0 \quad \text{for } j = 1,\ldots,n \tag{3.6}$$

where e is an n ones vector, $z_o=\max\{-q_i : 1\leq i\leq n\}$.

A feasible solution (w,z,z_o) to this system is called an almost complementary basic feasible solution if

(1) (w,z,z_o) is a basic feasible solution to (3.4) and (3.5).

(2) neither w_s nor z_s are basic for some s in $\{1,....,n\}$.

(3) z_o is basic, and exactly one variable from each complementary pair (w_j,z_j) is basic for $j=1,...,n$ and $j \neq s$.

Definition 3.1.3: Given an almost complementary basic feasible solution (w,z,z_o), where w_s and z_s are both non-basic, an adjacent complementary basic feasible solution is obtained by introducing either w_s or z_s in the basis replacing a variable other than z_o from the basis.

Consider the following QP problem:

$$\begin{aligned} &\textit{Minimize} \quad f(x) = c^T x + \frac{1}{2} x^T Q x \\ &\textit{subject to}: \quad Ax \leq b \\ &\qquad\qquad x \geq 0 \end{aligned}$$

where A is $m \times n$ matrix , b is $m \times 1$, c is $n \times 1$, x is $n \times 1$ and Q is nxn symmetric matrix.

Denoting the Lagrangian multiplier vectors of the constraints $Ax \leq b$ and $x \geq 0$ by u and v, respectively, and denoting the vector slack variables by y, the KKT condition can be written as

$$\begin{aligned} Ax + y &= b \\ -Qx - A^T u + v &= c \\ {}^T v = 0, u^T y &= 0 \\ x, y, u, v &\geq 0 \end{aligned}$$

Now, letting

$$M=\begin{bmatrix} 0 & -A \\ A' & Q \end{bmatrix}, \quad q=\begin{bmatrix} b \\ d \end{bmatrix}, w=\begin{bmatrix} y \\ v \end{bmatrix} \text{ and } z=\begin{bmatrix} u \\ x \end{bmatrix}.$$

Thus, we can write the KKT condition as a linear complementary problem w-Mz=q, $w^T z=0$ and (w,z) ≥ 0. (w, z) is a complementary pair. Therefore the complementary pivoting algorithm can be used to find a KKT point of the quadratic problem which stops in a finite number of iterations. The method has been shown to work well when the objective function is positive definite, and it requires computational effort with n+m constraints.

Difficulties may arise when the Hessian matrix Q is positive semi definite. The simplest practical approach to overcome this difficulty is to add a small constant to the diagonal elements of Q in such a way that the modified matrix Q becomes positive definite. Although the resultant solution will not be exact, the difference will be insignificant if the alterations are kept small enough.

Introducing the artificial variable z_0, the former algorithm moves among adjacent almost complementary basic feasible solutions until either a complementary basic solution is obtained or a direction indicating unboundedness of the region defined by (3.4) through (3.6) is found.

Steps of Lemke Algorithm:

Initialization Step: Introduce an artificial variable z_0 and consider the system (3.4) to (3.6). If $q \geq 0$, stop; (w,z)=(q,0) is a complementary basic feasible solution. Otherwise, express the system (3.4) and (3.5) in a table format as in the simplex table. Let $-q_s=\max\{-q_i : 1 \leq i \leq n\}$, and update the table by pivoting at row s and the z_0 column. Thus, the basic variables z_0 and w_j for j=1,2,...,n, and $j \neq s$ are non negative. Let $y_s = z_s$ and go to the main step.

Main Step:

1. In the updated table, let d_s be the column under the variable y_s and $\bar{q}$ be the right hand side column of constants denoting the values of the basic variables. If $d_s \leq 0$, go to step 4. Othewise, determine a row index r by the minimum ratio test

$$\frac{\bar{q}_r}{d_{rs}} = \min\left\{\frac{\bar{q}_i}{d_{is}} : d_{is} \succ 0, 1 \leq i \leq n\right\}$$

If the basic variable at the r is z_0, go to step 3. Otherwise, go to step 2.

2. The basic variable at row r is either w_l or z_l for some $l \neq s$ which are different. The variable y_s enters the basis and the table is updated by pivoting at row r and the y_s column, where

$$y_s = \begin{cases} z_l & \text{if } w_l \text{ leaves the basis} \\ w_l & \text{if } z_l \text{ leaves the basis} \end{cases}$$

Return to step 1.

3. Here y_s enters the basis, and z_0 leaves the basis. Update the current table by pivoting at the y_s column and the z_0 row. Stop, because a complementary basic feasible solution is obtained.

4. Stop with ray termination. A ray $R=\{(w,z,z_0)+\lambda d: \lambda \geq 0\}$ is found such that every point in R satisfies (3.4),(3.5) and (3.6). Here (w,z,z_0) is almost complementary basic feasible solution associated with the last table, and d is an extreme direction of the set defined by (3.4) and (3.5), having 1 in the row corresponding to y_s, $-d_s$ in the rows of the current basic variable and zero everywhere else. Stopping the algorithm at this step is termed ray termination.

Example: Consider the following QP problem:

$$\begin{aligned} Minimize\ & -18x_1 + x_1^2 - x_1x_2 + x_2^2 \\ subject\ to:\ & x_1 + x_2 \le 12 \\ & -x_1 + x_2 \le 6 \\ & x_1 \ge 0, x_2 \ge 0. \end{aligned}$$

Here $c=\begin{bmatrix} -18 \\ 0 \end{bmatrix}$, $Q=\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$, $A=\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$ *and* $b=\begin{bmatrix} 12 \\ 6 \end{bmatrix}$.

Denoting the vector slacks by y and the Lagrangian multiplier vectors for the constraints $Ax \le b$ and $x \ge 0$ by u and v respectively.

Then KKT condition is given as follows

$$\begin{aligned} Ax+y &= b \\ -Qx-A^Tu+v &= c \\ x^Tv=0,\ u^Ty &= 0 \\ x, y, u, v &\ge 0. \end{aligned}$$

We have the following system of equations

$$\begin{aligned} x1+x2+y1 &= 12 \\ -x1+\ \ y2 &= 6 \\ 2x1-x2-u1+u2+v1 &= -18 \\ -x1\ -2x2-u1-u2+v2 &= 0. \end{aligned}$$

Now $M=\begin{bmatrix} 0 & -A \\ A' & Q \end{bmatrix}=\begin{bmatrix} 0 & 0 & -1 & -1 \\ 0 & 0 & 1 & -1 \\ 1 & -1 & 2 & -1 \\ 1 & 1 & -1 & 2 \end{bmatrix}$, $w=\begin{bmatrix} y_1 \\ y_2 \\ v_1 \\ v_2 \end{bmatrix}$, $z=\begin{bmatrix} u_1 \\ u_2 \\ x_1 \\ x_2 \end{bmatrix}$, *and* $q=\begin{bmatrix} 12 \\ 6 \\ -18 \\ 0 \end{bmatrix}$.

Let $w=(w_1,w_2,w_3,w_4)^T$ and $z=(z_1,z_2,z_3,z_4)^T$, where $w_1=y_1, w_2=y_2, w_3=v_1, w_4=v_4$ and $z_1=u_1, z_2=u_2, z_3=x_1, z_4=x_2$.

Then the LCP can be written as

$$w\text{-}Mz=q \text{ or } \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} - \begin{bmatrix} 0 & 0 & -1 & -1 \\ 0 & 0 & 1 & -1 \\ 1 & -1 & 2 & -1 \\ 1 & 1 & -1 & 2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \begin{bmatrix} 12 \\ 6 \\ -18 \\ 0 \end{bmatrix}.$$

Now applying Lemke pivoting simplex type to this LCP, it yields the sequence of simplex iterations given in the following Table.

Iteration	Basic variables	Solution	Objective value	Entering variable	Leaving variable
1	(w_1, w_2, z_0, w_4)	(30,24,18,18)	0	z_3	w_4
2	(w_1, w_2, z_0, z_3)	(12,18,6,6)	-72	z_4	w_1
3	(z_4, w_2, z_0, z_3)	(4,14,2,10)	-84	z_1	z_0
4	(z_4, w_2, z_1, z_3)	(3,12,3,9)	-99	—	—

Hence, from the last simplex iteration table we have the KKT point $(x_1,x_2)=(z_3,z_4)=(9,3)$ and $(w_1,w_2,w_3,w_4,z_1,z_2,z_3,z_4)=(0,12,0,0,3,0,9,3)$. Since Q is positive semidefinite, the program is convex. Therefore the point (9,3) will be an optimal solution.

3.2 Convergence of Lemke Algorithm

In this section we discuss under certain conditions on the matrix M, the algorithm stop in a finite number of steps either with a complementary basic feasible solution or with ray termination, and it produces a KKT point and ray termination is only possible if the QP problem has unbounded optimal solution.

Definition 3.2.1: Let M be $n \times n$ matrix. Then, M is said to be copositive if $z^TMz \geq 0$ for each $z \geq 0$. Furthermore, M is said to be copositive-plus if it is copositive and if $z \geq 0$ and $z^TMz=0$ imply that $(M+M^T)z=0$.

Theorem 3.2.2: Suppose that each almost complementary basic feasible solution to the system defined by (3.4) to (3.6) is non degenerate; i.e. each basic variable is positive and suppose that M is copositive-plus. Then, the complementary pivoting algorithm stops in a finite number of steps. Further, if M has nonnegative entries with diagonal positive elements, then the complementary pivoting algorithm stops in a finite number of steps with complementary basic feasible solution.

Proof: Let (w,z,z_0) be an almost complementary basic feasible solution, where w_s and z_s are both nonbasic. Then, (w,z,z_0) has , at most two adjacent almost complementary basic feasible solutions, one obtained by introducing w_s in the basis and the other obtained by introducing z_s in the basis. By the non degeneracy assumption, each of this solutions is distinct from (w,z,z_0) because the column under w_s or z_s is ≤ 0, or else introducing w_s or z_s in the basis derives z_0 out of the basis, thus producing complementary basic feasible solution.

We now show that none of the almost complementary basic feasible solution generated by the algorithm is repeated. Let (w,z,z_0) be the point generated at a general iteration v. By contradiction , suppose that $(w,z,z_0)_{k+\alpha} = (w,z,z_0)_k$ for some positive integers k and α, where $k+\alpha$ is the smallest index for which a repetition is observed. By the non degeneracy assumption, $\alpha > 1$. Further more, by the rules of the algorithm , $\alpha > 2$. But since $(w,z,z_0)_{k+\alpha-1}$ is adjacent to $(w,z,z_0)_{k+\alpha}$, it is adjacent to $(w,z,z_0)_k$. If k=1, and since $(w,z,z_0)_k$ has exactly one adjacent almost complementary basic feasible solution, $(w,z,z_0)_{k+\alpha-1} = (w,z,z_0)_{k+1}$, and, hence , a repetition occurs at iteration $k+\alpha-1$, contradicting our assumption that the first repetition occurs at iteration $k+\alpha$. If $k \geq 2$, then $(w,z,z_0)_{k+\alpha-1}$ is adjacent to $(w,z,z_0)_k$ and, hence must be equal to $(w,z,z_0)_{k+1}$. In either case, a repetition occurs at iteration $(w,z,z_0)_{k+\alpha-1}$, which contradicts our assumption. Thus,the point generated by the algorithm are distinct.

Since there is only a finite number of almost complementary basic feasible solutions, and since non of them is repeated , the algorithm stops in a finite number of steps with a complementary basic feasible solution or with ray termination.

To show the algorithm stops with complementary basic feasible solution: First note that by the stated assumption on M, the system w-Mz=q, (w,z)≥0 has a solution ,say, by choosing z sufficiently large so that w=Mz+q≥0. The result then follows from above by noting that M is copositive-plus. This completes the proof.

Theorem 3.2.3: Let A be m×n matrix and let Q be an nxn symmetric matrix. If $y^TQy\geq 0$ for each y≥0, then the matrix

$$M=\begin{bmatrix} 0 & -A \\ A^T & Q \end{bmatrix}$$

is copositive. In addition, if y≥0 and $y^TQy=0$ implies that Qy=0, then M is copositive-plus.

Proof: First, we show that M is copositive. Let $z^T=(x^T, y^T)\geq 0$. Then,

$$z^TMz=(x^T,y^T)\begin{bmatrix} 0 & -A \\ A^T & Q \end{bmatrix}\begin{pmatrix} x \\ y \end{pmatrix}=y^TQy \qquad (3.7)$$

By assumption , $y^TQy\geq 0$, and, hence, Q is copositive. To show that M is copositive-plus, suppose that z≥0 and $z^TMz=0$. It suffices to show that $(M+M^T)z=0$. But

$$M+M^T=\begin{bmatrix} 0 & 0 \\ 0 & 2Q \end{bmatrix}.$$

And hence,

$$(M+M^T)z=\begin{bmatrix} 0 \\ 2Qy \end{bmatrix}.$$

Since $z^TMz=0$, then $y^TQy=0$ by (3.7). By assumption, since y≥0 and $y^TQy=0$, then Qy=0, and, hence $(M+M^T)z=0$, so that M is copositive-plus.

Corollary 3.2.4: If Q is positive semidefinite, then $y^TQy=0$ implies that Qy=0, so M is copositive plus. Furthermore, If Q has non negative elements with positive diagonal elements, then M is copositive-plus.

Proof: It is suffice to show that $y^TQy=0$ implies that Qy=0. Let Qy=d, and noting that Q is positive semidefinite, we get

$$0 \le (y^T + \lambda d^T)Q(y - \lambda d) = y^T Qy + \lambda^2 d^T Qd - 2\lambda \|d\|^2$$

Since $y^TQy=0$, then dividing the above inequality by:

λ and $\lambda \to 0^+$, it follows that 0=d=Qy.

It is obvious for non negative entries M is copositive , To show with positive diagonal elements: If $y \ge 0$ and $y^TQy=0$, then y=0, and, hence, Qy=0. By the theorem, M is copositive-plus.

Theorem 3.2.5: Consider the problem to minimize $f(x) = c^T x + \frac{1}{2} x^T Qx$ subject to $Ax \le b$, $x \ge 0$. Suppose that the feasible region is not empty. Further, suppose that the complimentary pivoting algorithm is used in an attempt to find a solution to the KKT system $w-Mz=q$, $(w,z) \ge 0$, $w^Tz=0$, where

$$M=\begin{bmatrix} 0 & -A \\ A^t & Q \end{bmatrix}, \quad q=\begin{bmatrix} b \\ d \end{bmatrix}, \quad w=\begin{bmatrix} y \\ v \end{bmatrix} \text{ and } z=\begin{bmatrix} u \\ x \end{bmatrix}.$$

where y is the vector of slack variables, and u and v are the Lagrangian multiplier vectors associated with the constraints $Ax \le b$ and $x \ge 0$, respectively. In the absence of degeneracy, under any of the following conditions, the algorithm stops in a finite number of iterations with a KKT point if:

1. Q is positive semidefinite and c=0
2. Q is positive definite
3. Q has nonnegative elements with positive diagonal elements.

Furthermore, if Q is positive semidefinite, then ray termination implies that optimal solution is unbounded.

Proof: Assume that Q is symmetric. By theorem 3.2.2 , the algorithm stops in a finite steps of iterations with either a KKT point or a ray termination. If Q is positive semidefinite or positive definite, or has non negative elements with positive diagonal elements , then by corollary 3.2.4 , M is copositve-plus.

Now suppose that ray termination occurs . Since M is copositive-plus, ray termination is possible only if the following system has no solution:

$$\begin{aligned} Ax + y &= b \\ -Qx - A^T u + v &= c \\ x, y, u, v &\geq 0. \end{aligned}$$

By Farakas' theorem , the following system must have a solution (d,f):

$$Ad \leq 0 \tag{3.8}$$
$$A^T f - Qd \geq 0 \tag{3.9}$$
$$f \geq 0 \tag{3.10}$$
$$d \geq 0 \tag{3.11}$$
$$b^T f + c^T d < 0. \tag{3.12}$$

Multiplying (3.9) by $d^T \geq 0$, and noting that $f \geq 0$ and $Ad \leq 0$, it follows that

$$0 \leq d^T Af - d^T Qd \leq 0 - d^T Qd = -d^T Qd. \tag{3.13}$$

By assumption, there exist x^* and y^* such that $Ax^* + y^* = b, (x^*, y^*) \geq 0$. Substituting for b in (3.12) and noting (3.9) and that $f(x^*, y^*) \geq 0$, we get

$$0 > c^T d - b^T f = c^T d + (y^* + Ax^*)^T f \geq c^T d + x^{*T} A^T f \geq c^T d + x^* Qd \tag{3.14}$$

Now suppose Q is positive semidefinite. By (3.13), it follows that $d^T Qd = 0$ and by corollary 3.2.4, it follows that Qd=0. By (3.14), we have $c^T d < 0$. Since $Ad \leq 0$ and $d \geq 0$, d is a direction of the feasible region, so that $x^* + \lambda d$ is feasible for all $\lambda \geq 0$. Now consider $f(x^* + \lambda d)$, where $f(x) = c^T x + \frac{1}{2} x^T Qx$. Since Qd=0, we get

$$f(x^* + \lambda d) = f(x^*) + \lambda(c^T + x^{*T} Q)d + \tfrac{1}{2}\lambda^2 d^T Qd = f(x^*) + \lambda c^T d.$$

Since $c^T d < 0$, $f(x^* + \lambda d)$ approaches $-\infty$ by choosing λ arbitrarily large; thus we have an unbounded optimal solution.

We now show that any termination is not possible under conditions 1, 2, or 3 of the theorem. Now suppose, by contradiction to the conclusion of the theorem that ray termination occurs under any of these conditions. From (3.13), $d^T Qd \leq 0$. Under conditions 2 or 3, d=0, which is impossible in view of (3.14). If condition 1 holds, on the other hand, then Qd=0. This together with the assumption that c=0, contradicts (3.14). This completes the proof.

3.4 Programming in MATLAB using Lemke Algorithm

This matlab program is capable of solving a convex quadratic programming problem changing to linear complementary problems by Lemke method. For the convergence of the algorithm it is necessary that either Hessian of the objective function be positive definite or positive semidefinite Hessian with linear term zero. From the following programme we will obtain an output result after supplying the inputs and by calling Lemke(A,Q,b,c). Lemke.m Quadratic Programming using Lemke's Complementary Pivoting Algorithm solves quadratic program in standard form:

minimize c'*x + 1/2*x'*Q*x
subject to A*x <= b
x >= 0.

```
function [x,obj,st,i_tab,f_tab,f_bas,d] = Lemke(A,Q,b,c)
```

```
% inputs:
%
% A - constraint left-hand matrix
% Q - symmetric objective matrix.
% b - constraint right-hand column vector
% c - objective column vector
%
% Returns:
%
% x - (local) optimal solution
% obj - optimal value of objective function
% st - solution type 1 - optimal
% 2 - unbounded
% 3 - infeasible
% i_tab - initial tableau
% f_tab - final tableau
% f_bas - final basis
% d - extreme direction
%
stype={'Optimal';'Unbounded';'Infeasible'};
m=length(c);
m1=length(b);

M=[zeros(m1,m1) -A;A' Q];
q=[b;c];

[w,z,st,i_tab,f_tab,f_bas,d] = lemke(M,q);

if st~=3
  x=z(m1+1:m+m1);
end

if ~isempty(d)
  dx=d(m+2*m1+1:2*m+2*m1);
  if sum(dx==0)
    st=1;
  end
end

if st~=3
  obj = c'*x+.5*x'*Q*x;
  if any(A*x>b+.001) % infeasible solution
    st=3;
  end
```

```
  end

  if st~=1
    obj=NaN;
    x=[];
  end

  st=stype{st};

  return

  % ==================
  % local functions
  % ==================

  % ------------------------------------------------
    function [w,z,st,i_tab,f_tab,f_bas,d] = lemke(M,q)
  % ------------------------------------------------

  d=[];
  m = length(q);
  tableau = [eye(m) -M -ones(m,1) q];
  i_tab=tableau;

  if all(q>=0) % Solution already found
    w=q;
    z=zeros(m,1);
    f_bas=1:m;
    f_tab=i_tab;
    st=1;
  else % Initialisation step
    tm = 2*m+2;
    zin = tm-1;
    rowin=1:m;
    [y,s]=min(q);
    pcol=tm-1;
    tableau=pivot(tableau,s,pcol,m);
    rowin(s)=zin;
    pcol=m+s;
    ds=tableau(:,pcol);
    while any(ds>0) % Main step 1 - select exit variable and pivot
      s=finds(tableau,m,tm,pcol);
      if s==0
        w=[];
        st=3;
```

```
        f_tab=[];
        f_bas=[];
        z=[];
        return
      end
      tableau=pivot(tableau,s,pcol,m);
      if rowin(s)~=zin % Main step 2 - update index
        ptemp=pcol;
        if rowin(s)>m
          pcol=rowin(s)-m;
        else
          pcol=rowin(s)+m;
        end
        rowin(s)=ptemp;
        ds=tableau(:,pcol);
      else % Main step 3 - stop with feasible solution
        w=tableau(:,tm);
        rowin(s)=pcol;
        a=zeros(2*m,1);
        a(rowin)=w;
        w=a(1:m);
        z=a(m+1:2*m);
        st=1;
        f_bas=rowin;
        f_tab=tableau;
        return;
      end
    end
    w=tableau(:,tm); % Main step 4 - stop with ray termination
    a=zeros(2*m,1);
    a(rowin)=w;
    w=a(1:m);
    z=a(m+1:2*m);
    f_bas=rowin;
    f_tab=tableau;
    d=zeros(1,tm-1);
    d(f_bas)=-ds;
    d(pcol)=1;
    st=2;
  end

  return

  % ------------------------------------------
    function tableau=pivot(tableau,prow,pcol,m)
```

```
% ----------------------------------------

tableau(prow,:)=tableau(prow,:)./tableau(prow,pcol);
for i=1:m
  if i~=prow

tableau(i,:)=tableau(i,:)-tableau(prow,:)*tableau(i,pcol)/tableau(prow,pcol);
  end
end

return

% ------------------------------------
  function s = finds(tableau,m,tm,pcol)
% ------------------------------------

if max(tableau(:,pcol))<.00001
  s=0;
else
  ic=0;
  for i = 1:m
    if tableau(i,pcol)>0
      ic=ic+1;
      cv=tableau(i,tm)./tableau(i,pcol);
      if ic==1
        mv=cv;
        s=1;
      elseif cv==mv & rand>.5
        s=i;
      elseif cv<mv
        mv=cv;
        s=i;
      end
    end
  end
end

return
```

CHAPTER 4

CONCLUSIONS

For any given quadratic programming problem the major factor limiting the choice among the solution methods is the character of Q in the objective function. If Q is or might be indefinite or positive semidefinite, convergence cannot be guaranteed for either Wolfe's or Lemke's method, and it becomes necessary to use some type of constrained hill-climbing approach to find the various local optima that may exist. This strategy is embodied in the quadratic programming algorithm of Beale, as well as in many other more general methods that can be applied to all linearly constrained problems, regardless of the nature of their objective functions.

In choosing among the algorithms, there are a number of different considerations to bear in mind. For example, suppose analyst wishes to solve quadratic programming problem with a convex objective function as quickly or as cheaply readily available, so that he is forced to modify an existing linear programming code. In such a case his primary consideration will be that of expedience , and it is clear that he should plan to use Wolfe's algorithm: Conversion from the simplex to the "Wolfe-simplex" method requires little more than modifying the logic by which the entering variable is chosen, so that the complementary slackness conditions $x_j u_j=0$, $j=1,\ldots,n$, are preserved. This may be a decisive advantage for users with limited budgets.

It should be noted that the Lemke algorithm has the advantage with respect to Wolfe and Beale, of not requiring the precomputation of a starting basic feasible solution to $Ax=b$. On the other hand, it also has the marginal disadvantage of not operating with in a feasible region: If computation interrupted for any reason before the problem has been

completely solved, not feasible solution whatever will have been guaranteed by the algorithm. This is because in almost every real problem solving environment the computer is allowed to run until the analyst can conclude with reasonable confidence that convergence will not occur.

References

[1] Bapi Chatterjee: *Quadratic Programming by Wolf method*, http://www. Mathworks.com, 28 April 2010

[2] Bazara M.S., Sherall H.D., Shetty C.M.: *Nonlinear Programming Theory and Algorithms*, (2nd edition) 1993

[3] Deumlich R.: *Optimization Theory II*, Undergraduate text, Addis Ababa 1996.

[4] Donald M. Simmons: *Nonlinear Programming for Operation Research.*

[5] John Burkardt: *Lemke solver for quadratic programming*, http://people.sc.fsu. edu/~burkardt/m_src/lemke/lemke.html, 05 December 2007

[6] Jorge Nocedal, Stephen J.Wright: *Numerical Optimization,* Springer operation research.

[7] Paul A. Jensen and Jonathan F: *Operations Research Models and Methods,* http://www.me.utexas.edu/~jensen/ORMM/supplements/methods/ nlpmethod/S2_quadratic.pdf

[8] Sinha S.M.: *Mathematical Programming Theory and Algorithms*, (First edition) 2006