Department of Electrical
and
Computer Systems Engineering

Technical Report
MECSE-7-2003

Matrix Formulation for the Support Vector Machine Classifier

D. Lai and N. Mani

Matrix Formulation for the Support Vector Machine Classifier

D. LAI , N.MANI
Dept. of Electrical and Computer Systems Engineering
Monash University, Clayton, Vic. 3168, Australia.
23/7/2003

Abstract: In this paper, we investigate solving the constrained quadratic program for Support Vector Machine classification formulation by solving a constrained set of linear equations written in matrix form. We have applied this form in our previous work and have observed its validity through empirical results. Several other researchers have proposed optimization algorithms, which solved a linear minimization form similar in nature. We first derive this linear form through the use of convex mathematical programming and rewrite it in matrix form. We attempt to reconcile our form with that which has been popularly used.

I. INTRODUCTION

The Support Vector Machine(SVM) formulation is a supervised learning formulation introduced by Vapnik[1] for pattern recognition. This formulation embodies the Structural Risk Minimization principle proposed by Vapnik in his pioneering of statistical learning theory where the performance of the SVM binary classifier is determined by controlling its capacity and minimizing the training error on the data set. The SVM classifier problem is generally treated using mathematical programming methods which results in a constrained quadratic program obtained from Lagrange Theory. The constrained Lagrange for the SVM classifier is a quadratic program that generally requires the use of interior point methods, activesets or some form of chunking[2] which are somewhat complex to implement for first time users of Support Vector Machines. Furthermore, the standard Lagrange Dual results in an objective function subject to an equality constraint and bounded Lagrange Multipliers which require enforcement at each stage of the iteration.

Several researchers have proposed optimization algorithms[3-5] that solve a linear program instead. They have partioned their data into subsets and iterated on individual subsets. This amounts to solving a decomposed problem, hence the apt name of decomposition methods. The convergence of the decomposition methods have been examined by[6]. For now, this method seems to be popular and easy to implement with comparable optimization times.

In this paper, we proceed further by showing that this linear progam can be written as a set of linear equations which surprisingly results in minimizing the training error on a data set. Futhermore we show this is equivalent to obtaining the solution to SVM classification which also enforces Structural Risk Minimization. We first attempt to explore the duality properties of the SVM quadratic program. We then show that solving a constrained minimization of training errors directly satisfies the optimal solution requirements for the dual Lagrange program. The resulting minimization program is a linear program with implicit maximization of the margin of the hyperplane. Our linear program constitutes solving a set of linear equations which can be written in matrix form and allows simple iterative algorithms to obtain the solution. We give a geometrical interpretation of the feasible region of solutions resulting from the mathematical programs in the hope of providing a better understanding of the underlying mechanics of optimization algorithms. The geometrical view is specific to the set of solutions and is different from previous work [7] which examines the construction of the classifier based on the geometric distribution of data.

This paper is divided in to sections as follows; section 2 will review the Support Vector Machine classifier from a mathematical programming perspective and draw attention to the geometric properties resulting from the constraints on the resulting objective functions. Section 3 is devoted to the derivation of the linear program and the matrix form while Section 4 will investigate the solution set further. The

remainder of the paper will be devoted to a discussion on the possibilities of the matrix form and future directions for research.

II. Support Vector Machine Classification

A. *Overview of the Support Vector Classification Formulation*

In binary classification, we are given a set of training data which have been labelled according to two classes. For historical reasons and the ease of modelling, the labels are choosen to be +1 and -1. The data to be classfied is collected in a training set defined here as,

$$\begin{aligned} &\Theta=\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2)...(\mathbf{x}_n, y_n)\} \\ &\mathbf{x}_i \in \mathbb{R}^N \\ &y_i = \{1,-1\} \end{aligned} \tag{1}$$

The task is then to train a machine to learn the relationship between the data and their respective labels. This amounts to learning the geometrical structure of the space in which the two classes of data lie. In Support Vector Machines, the idea is to define a boundary separating these two classes as much as possible. This can be interpreted as a linear hyperplane in data space where the distance between the boundaries of the two classes and the hyperplane is known as the margin of the hyperplane. Maximizing the margin of the hyperplane is then equivalent to maximizing the distance between the class boundaries. Vapnik suggests that the form of the hyperplane be chosen from family of functions with sufficient capacity[1]. In particular, F contains functions for the linearly and non-linearly separable hyperplanes;

$$f(x) = \sum_{i=1}^{l} w_i x_i + b \tag{2}$$

$$f(x) = \sum_{i=1}^{l} w_i \phi(x_i) + b \tag{3}$$

The weight vector, **w** in (3) is no longer the same expansion as in the linearly separable case (2). In fact, the non-linear mapping $\phi : \mathbf{x} \subset \Re^n \to \Re^m$ and $n, m \in [1, \infty)$ defines the mapping from data space to feature space. Hence the weights in feature space will have a one to one correspondence with the elements of $\phi(\mathbf{x})$. Now for separation in feature space, we would like to obtain the hyperplane with the following properties;

$$\begin{aligned} &f(\mathbf{x}) = \sum_{i=1}^{l} w_i \varphi(\mathbf{x}_i) + b \\ &f(\mathbf{x}) > 0 \quad \forall \quad i : y_i = +1 \\ &f(\mathbf{x}) < 0 \quad \forall \quad i : y_i = -1 \end{aligned} \tag{4}$$

The conditions in (4) can be described by a linear discriminant function, so that for each element pair in Θ, we have;

$$y_i \left(\sum_{i=1}^{l} w_i \varphi(\mathbf{x}_i) + b \right) \geq 1 - \xi_i \tag{5}$$

The size of the soft-margin is govern by the positive slack variables, ξ_i which relax the conditions in (4). The distance from the hyperplane to a support vector is $\frac{1}{\|\mathbf{w}\|}$ and the distance between the support vectors of one class to the other class is simply $\frac{2}{\|\mathbf{w}\|}$ by geometry. The SVM margin maximization problem is then formulated as;

$$\text{P.1}\begin{cases}\text{minimize} \quad \Im(\mathbf{w})=\frac{1}{2}\sum_{i=1}^{l} w_i^2 + C\sum_{i=1}^{l}\xi_i \\ \text{subject to} \begin{cases} y_i \ (\sum_{i=1}^{l} w_i\varphi(\mathbf{x}_i)+ b \ ) \ \geq \ 1\text{-}\xi_i \\ \xi_i > 0, \forall i = 1..l \end{cases}\end{cases} \tag{6}$$

The parameter C can be interpreted as a regularization parameter which controls the tradeoff between generalization accuracy and the number of training errors. A larger C generally results in lower training errors but poorer prediction accuracy.

The program in P.1 is difficult to solve using practical optimization methods. However, duality theory allows us to transform a program to an easier and more practical version to solve. Duality in optimization theory was mainly developed by Fenchel, Kuhn and Tucker, Rockafeller, Dorn, Wolfe and many others[8-11]. Their theorems mainly contributed to the understanding of the popular and important Lagrange Theory which describes how to obtain an equivalent dual program of a mathematical program. In summary, suppose we have a mathematical program of the following form;

$$\text{A.I}\begin{cases} \text{minimize} \quad \Im(\mathrm{w}) \\ \text{subject to constraints} \begin{cases} \mathrm{H(w)} \leq 0 \\ \mathrm{G(w)}=0 \end{cases} \\ \text{where H(w)}=\begin{pmatrix} \mathrm{h}_1(w) \\ \vdots \\ \mathrm{h}_m(w) \end{pmatrix} \qquad \mathrm{G(w)}=\begin{pmatrix} \mathrm{g}_1(w) \\ \vdots \\ \mathrm{g}_n(w) \end{pmatrix} \end{cases} \tag{7}$$

If the objective function $\Im(\mathbf{w})$, equality constraints $g_i(w)$ and inequality constraints $h_i(w)$ all have smooth first partial derivatives which are linearly independent, there exists scalars $\alpha, \beta \geq 0$ known as Lagrange Multipliers so that we can write;

$$\text{A.II}\begin{cases} \text{minimize} \quad L(w) = f(\mathrm{w}) + \sum_{\mathrm{i}=1}^{n} \beta_i g_i(w) + \sum_{\mathrm{j}=1}^{m} \alpha_j h_j(w) \\ \text{subject to} \begin{cases} \nabla f(\mathrm{w}) + \sum_{\mathrm{i}=1}^{n} \beta_i \nabla g_i(w) + \sum_{\mathrm{j}=1}^{m} \alpha_j \nabla h_j(w) = 0 \\ \alpha, \beta > 0 \end{cases} \end{cases} \tag{8}$$

The program A.II is the *dual* to the program A.I and the solution to it is also a solution to A.I. Kuhn and Tucker later generalized this duality to include non-differentiable functions by introducing a saddle point theorem. It turns out that when the objective functions are non-differentiable, any saddle point of A.II is also a solution to A.I. We now apply Lagrange Theory to solve (6) giving us the Lagrange Primal problem of the following form;

$$\text{P.2}\begin{cases}\text{minimize } \Im(\mathrm{w},\alpha)=\dfrac{1}{2}\sum_{i=1}^{l} w_i^2 + C\sum_{i=1}^{l}\xi_i + \sum_{i=1}^{l}\alpha_i(y_i(\sum_{j=1}^{l} w_j\varphi(x_j)+b)-1+\xi_i) \\ \qquad\qquad + \sum_{i=1}^{l}\eta_i\xi_i \\ \text{subject to}\begin{cases}\nabla\left(\dfrac{1}{2}\sum_{i=1}^{l} w_i^2 + C\sum_{i=1}^{l}\xi_i\right)+\nabla\left(\begin{array}{l}\sum_{i=1}^{l}\alpha_i(y_i(\sum_{j=1}^{l} w_j\varphi(x_j)+b)-1+\xi_i) \\ +\sum_{i=1}^{l}\eta_i\xi_i\end{array}\right)=0 \\ \alpha_i \geq 0, \forall \mathrm{i}=1..l\end{cases}\end{cases} \tag{9}$$

The program P.2 is popularly solved through the use of its dual representation which is found by incorporating the gradient condition in P.2 into the objective function and eliminating the primal variables. The stationary partial gradients of P.2 are given by;

$$\begin{aligned}\nabla\Im(w,\alpha,\xi,b) &= \mathbf{w}+\sum_{i=1}^{l}\alpha_i y_i\varphi(\mathbf{x}_i)=\mathbf{0} \\ \nabla\Im(w,\alpha,\xi,b) &= \mathbf{y}(\sum_{i=1}^{l} w_i\varphi_i(\mathbf{x})+b)-1+\boldsymbol{\xi}=0 \\ \nabla\Im(w,\alpha,\xi,b) &= \sum_{i=1}^{l}\alpha_i y_i = 0 \\ \nabla\Im(w,\alpha,\xi,b) &= C\boldsymbol{\xi}-\boldsymbol{\alpha}=\mathbf{0}\end{aligned} \tag{10}$$

We then eliminate the primal variables through back substitution, giving us the Lagrange Dual;

$$\text{P.3}\begin{cases}\text{minimize } \Im(\boldsymbol{\alpha})=-\sum_{i=1}^{l}\alpha_i+\dfrac{1}{2}\sum_{i,j=1}^{l}\alpha_i\alpha_j y_i y_j\left\langle\varphi(\mathbf{x}_i),\varphi(\mathbf{x}_j)\right\rangle \\ \text{subject to } \begin{cases}0\leq\alpha_{\mathrm{i}}\leq C \\ \sum_{i=1}^{l}\alpha_{\mathrm{i}} y_i = 0\end{cases}\end{cases} \tag{11}$$

In P.3 we find that the feature vectors exist as dot products and can be represented by a kernel function using Mercer's Theorem. The formulation avoids the task of explicitly specifying the feature space mapping and we simply have to choose a kernel function;

$$K(\mathbf{x}_i,\mathbf{x}_j)=\left\langle\varphi(\mathbf{x}_i),\varphi(\mathbf{x}_j)\right\rangle \tag{12}$$

When solving the dual programs P.3, we can derive the following dual hyperplane form expressed solely in terms of the Lagrange Multipliers by using the gradient condition in (10) to substitute for (4);

$$f(x)=\sum_{i=1}^{l}\alpha_i y_i K(x_i,x)+b \tag{13}$$

The SVM solution is the vectors w^*,α^* which satisfy P.1-P.3 giving the trained Support Vector Classifier defined by the decision functions;

$$f(x)=\mathrm{sign}(\sum_{i=1}^{l} w_i^* x_i + b) \tag{14}$$

$$f(x) = \text{sign}(\sum_{i=1}^{l} \alpha_i^* y_i K(x_i, x) + b) \tag{15}$$

The asterisk denotes the optimal values of the variables and we should state that the form of (14) is only practical for linearly separable Support Vector Machine classifiers. It should be noted that the solution is sparse, meaning that a lot of $\alpha_i = 0$ and the decision function, $f(x)$ could be represented solely by the Support Vectors, (i.e. $\alpha_i \neq 0$).

III. A PRACTICAL LINEAR PROGRAM: MINIMIZATION OF TRAINING ERRORS

The dual form of P.3 is in essence a constrained quadratic program which requires optimizers that use activesets, interior point methods and so on. However, simpler programs have been devised to solve a different problem to P.3 choosing instead to solve the following program;

$$\text{P.4} \begin{cases} \text{minimize } \nabla \Im(\boldsymbol{\alpha})^T d \\ \text{where } \begin{cases} 0 \le \alpha_i + d_i \le C \\ \sum_{i=1}^{l} y_i d_i = 0 \end{cases} \end{cases} \tag{16}$$

Their algorithms terminate using a reformulated Karush-Kuhn-Tucker (KKT) conditions[12];

$$\begin{aligned} \nabla \Im(\alpha^*)_i + \text{by}_i \ge 0 \quad & \alpha_i = 0 \\ \nabla \Im(\alpha^*)_i + \text{by}_i \le 0 \quad & \alpha_i = C \\ \nabla \Im(\alpha^*)_i + \text{by}_i = 0 \quad & 0 < \alpha_i < C \end{aligned} \tag{17}$$

Most of the successful SVM optimization algorithms mentioned before, including the popular SMO and SVMlight further use a decomposition method, first proposed by Osuna[13] to tackle the problem of memory with large datasets. However [12]have mentioned that decomposition methods tend to be slower than Newtonian methods. Unfortunately, Newtonian methods require that the entire kernel matrix be kept in memory which may not be possible for datasets that have 10000 points or more. Thus a compromise on which algorithm to use when solving a SVM problem has to be made. Therefore, the next logical thing which follows is to investigate a possible design of a hybrid optimization algorithm that possibly uses some Newtonian or linear iterative update method combined with a form of space decomposition to solve P.4. The problem with P.4 is that it is not immediately clear how to apply linear iterative methods which are specifically designed to solve a set of linear equations; in particular partial differential equations. The motive for this paper is then clear, for to apply these iterative methods we have to first establish a system of linear equations closely related to P.4. In the following, we show how this is done by deriving (16) from basic principles first and then forming the set of linear equations in matrix form.

A. THE GRADIENT OF THE UNCONSTRAINED QUADRATIC PROGRAM

In this section, we first investigate the **unconstrained** QP problem of P.3 derived by dropping the constraints on the objective function. We can easily rewrite the function $\psi(\boldsymbol{\alpha})$ in matrix form which we now define as;

$$\psi(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$= \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{G} \boldsymbol{\alpha} \tag{18}$$

where $\begin{cases} \boldsymbol{\alpha}, \mathbf{1} \in \Re^l \\ \mathbf{G} \in \Re^l \times \Re^l \\ \mathrm{G}_{ij} = y_i y_j K(x_i, x_j) \\ y_i \in (1, -1) \quad \forall i = 1..l \end{cases}$

The function $\psi(\boldsymbol{\alpha})$ is quadratic in terms of the variable α (Lagrange Multiplier) and has a unique maximum which we denote as α^*. As before the asterisk defines the optimal values of the α vector, which in this case refers to the unique maximizer. The maximum occurs when all partial derivatives vanish or simply the gradients with respect to the vector $\boldsymbol{\alpha}$ become stationary, i.e

$$\nabla \psi(\boldsymbol{\alpha}^*) = 0 \tag{19}$$

For a quadratic function, this condition is necessary and sufficient (ref kaplan) in order to obtain the optimal vector $\boldsymbol{\alpha}^*$ which is the unique maximizer to $\psi(\alpha)$.The gradients of (18) with respect to the Lagrange Multipliers, α_i can be found by taking the partial derivatives of (18) which is Gateux-differentiable giving;

$$\nabla_{\alpha_i} \psi(\alpha) = 1 - y_i \sum_{j=1}^{l} \alpha_j y_j K(x_i, x_j) \qquad \forall i = 1..l \tag{20}$$

We verify the differentiability in the following simple example for $\boldsymbol{\alpha} \in \Re^2$. Let us denote $K_{ij} = K(x_i, x_j)$ and expanding the function $\psi(\boldsymbol{\alpha})$, we get;

$$\psi(\boldsymbol{\alpha}) = \alpha_1 + \alpha_2 - \frac{1}{2}\left(\alpha_1^2 y_1^2 K_{11} + \alpha_1 \alpha_2 y_1 y_2 K_{12} + \alpha_2 \alpha_1 y_2 y_1 K_{21} + \alpha_2^2 y_2^2 + K_{22}\right)$$

$$= \alpha_1 + \alpha_2 - \frac{1}{2}\left(\alpha_1^2 y_1^2 K_{11} + 2\alpha_1 \alpha_2 y_1 y_2 K_{12} + \alpha_2^2 y_2^2 K_{22}\right)$$

$$\nabla_{\alpha_1} \psi(\boldsymbol{\alpha}) = \frac{\partial \psi(\boldsymbol{\alpha})}{\partial \alpha_1} = 1 - \frac{1}{2}\left(2\alpha_1 y_1^2 K_{11} + 2\alpha_2 y_1 y_2 K_{12}\right)$$

$$= 1 - y_1\left(\alpha_1 y_1 K_{11} + \alpha_2 y_2 K_{12}\right) \tag{21}$$

$$= 1 - y_1 \sum_{j=1}^{2} \alpha_j y_j K_{1j}$$

We can clearly obtain (21) from (20) if we set $i = 1$ and we can quickly check that this is true also for $i = 2$. We can now generalize this to the entire vector $\boldsymbol{\alpha}$ and compute the maximizer $\boldsymbol{\alpha}^*$ directly by applying (19);

$$\psi(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{G} \boldsymbol{\alpha}$$

$$\nabla_{\boldsymbol{\alpha}} \psi(\boldsymbol{\alpha}) = \mathbf{1}^T \mathbf{1} - \frac{1}{2} \mathbf{1}^T \mathbf{G} \boldsymbol{\alpha} = 0 \tag{22}$$

The unique maximizer is then found by;

$$\rightarrow \mathbf{1}^T \mathbf{2} = \mathbf{1}^T \mathbf{G} \boldsymbol{\alpha}$$

$$\therefore \boldsymbol{\alpha}^* = \mathbf{G}^{-1} \mathbf{2}$$

Resubstituting this back into (18), provided **G** is non-singular, we get the maximum value of $\psi(\boldsymbol{\alpha})$ as;

$$\begin{aligned}\psi(\boldsymbol{\alpha}^*) &= \left(\mathbf{G}^{-1}\mathbf{2}\right)^T \mathbf{1} - \frac{1}{2}\left(\mathbf{G}^{-1}\mathbf{2}\right)^T \mathbf{G}\mathbf{G}^{-1}\mathbf{2} \\ &= \left(\mathbf{G}^{-1}\mathbf{2}\right)^T \mathbf{1} - \left(\mathbf{G}^{-1}\mathbf{2}\right)^T \mathbf{1} = 0\end{aligned} \tag{23}$$

This is somewhat interesting because we can conclude that no matter what the size of the training data set, the maximum value possible for P.3 is zero provided the matrix G is nonsingular. The constraints in P.3 actually restrict the feasible region of solutions which results in P.3 having a value larger than zero. A non-singular G is guranteed by using a positive definite kernel, e.g gaussian kernel and provided there are no duplicate examples with opposite labels. We will elaborate on the possiblities further arising from this observation in the next section.

It can be shown quickly that if minimize the negative of $\psi(\boldsymbol{\alpha})$, we will end up with the same unique solution provided the condition of $\det \mathbf{G} \neq 0$ on the matrix **G** holds. This fact is well known and, we have a trivial dual of the form;

$$\max_{\boldsymbol{\alpha}} \psi(\boldsymbol{\alpha}) = -\min_{\boldsymbol{\alpha}} \psi(\boldsymbol{\alpha}) \tag{24}$$

In fact the negative maximization of P.3 is known as the Wolfe dual in the literature (ref smola). We now proceed further with a minor observation. Scalar multiples of the function $\psi(\boldsymbol{\alpha})$ gives the same unique maximizer too. We show this in the following technical lemma.

Lemma III.1: Let the quadratic function $\psi(\boldsymbol{\alpha})$ where $\boldsymbol{\alpha} \in \Re^l$, have a unique maximum denoted by $\boldsymbol{\alpha}^*$. Then for any nonzero scalar $k, k \in \Re$ the function $k\psi(\boldsymbol{\alpha})$ possesses the same unique maximizer solution $\boldsymbol{\alpha}^*$.

Proof:

Let $\psi(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$ (25)

Then

$$k\psi(\boldsymbol{\alpha}) = k\sum_{i=1}^{l} \alpha_i - \frac{k}{2}\sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \tag{26}$$

The solution to (26) is found as before when all gradients with respect to the variables are stationary. In particular, we require that $\forall i = 1..l$;

$$\begin{aligned}\nabla_{\alpha_i} k\psi(\boldsymbol{\alpha}) &= \nabla\left(k\sum_{i=1}^{l} \alpha_i - \frac{k}{2}\sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j)\right) \\ &= k - \frac{k y_i}{2}\sum_{i,j=1}^{l} \alpha_j y_j K(x_i, x_j) \\ &= k\left(1 - \frac{y_i}{2}\sum_{i,j=1}^{l} \alpha_j y_j K(x_i, x_j)\right) \\ &= k\nabla_{\alpha_i}\psi(\boldsymbol{\alpha})\end{aligned}$$

At the maximum point, $k\nabla\psi(\boldsymbol{\alpha}) = 0$ which holds if and only if $\nabla\psi(\boldsymbol{\alpha}) = 0, \forall \alpha_i \in \boldsymbol{\alpha}$ and hence the solution is also $\boldsymbol{\alpha}^*$ □

The general element gradient function, (20) by itself seems rather uninteresting, yet it is explicitly used as the objective function for the program (16). We know that they have to be zero at the maximum

or minimum points of the quadratic function and this is what optimization methods such as gradient descent methods iterate on. However, keeping in mind that we are examining the objective function of P.3 alone minus the constraints, our previous observation gives us the idea that we could manipulate scalar multiples of the gradients of a quadratic function to a form which we could associate easily with. We now use the result in **Lemma III.1** to tie in with a well-known quantity in the SVM classification problem by constructing the following theorem.

Theorem III.1: Let $\psi(\boldsymbol{\alpha})$ be an unconstrained SVM quadratic function having the form,

$$\psi(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2}\boldsymbol{\alpha}^T \mathbf{G}\boldsymbol{\alpha}$$

where the elements of **G** are defined as in (18). Let the unique maximum be denoted by $\boldsymbol{\alpha}^*$. Then for $\boldsymbol{\alpha} \neq \boldsymbol{\alpha}^*$ there exists a scalar, δ_i i.e $\delta_i \geq 0$ such that;

$$y_i - f_E(x_i, \boldsymbol{\alpha}) \leq |\delta_i| \quad \forall i = 1..l$$

where $f_E(x_i, \boldsymbol{\alpha}) = \sum_{j=1}^{l} \alpha_j y_j K(x_i, x_j) + b$ and $\lim_{\delta_i \to 0} \boldsymbol{\alpha} = \boldsymbol{\alpha}^*, \forall i = 1..l$ holds whenever **G** is nonsingular.

Proof:
Let

$$\underset{\alpha_i}{\upsilon}(\alpha) = y_i - \sum_{j=1}^{l} \alpha_j y_j K(x_i, x_j) \tag{27}$$

By construction, the vector $\boldsymbol{\alpha}$ lies in the convex set $\Re^l$. Then by the properties of a convex set, for any maximal $\boldsymbol{\alpha}^*$, there exists a vector, α in $\Re^l$ such that the vector $z = (1-t)\boldsymbol{\alpha}^* + t\alpha$ also lies in $\Re^l$ for some $0 < t \leq 1$. We then have;

$$\begin{aligned}
\underset{\alpha_i}{\upsilon}(z) &= \underset{\alpha_i}{\upsilon}\left((1-t)\boldsymbol{\alpha}^* + t\alpha\right) \\
&= y_i - \sum_{i,j=1}^{l} \left((1-t)\alpha_j^* + t\alpha_j\right) y_j K(x_i, x_j) \\
&= y_i - \sum_{i,j=1}^{l} \alpha_j^* y_j K(x_i, x_j) + t\sum_{i,j=1}^{l} \alpha_j^* y_j K(x_i, x_j) - t\sum_{i,j=1}^{l} \alpha_j y_j K(x_i, x_j) \\
&= \underset{\alpha_i}{\upsilon}(\alpha^*) + t\sum_{i,j=1}^{l} (\alpha_j^* - \alpha_j) y_j K(x_i, x_j) \\
&\leq \underset{\alpha_i}{\upsilon}(\alpha^*) + t\sum_{i,j=1}^{l} \alpha_j^* y_j K(x_i, x_j)
\end{aligned} \tag{28}$$

Let $t = 1$, we can see that the right hand most term is a constant which we can denote by ε and we now rewrite the inequality as ;

$$\underset{\alpha_i}{\upsilon}(\alpha) - \underset{\alpha_i}{\upsilon}(\alpha^*) \leq |\varepsilon| \tag{29}$$

Now, we can see from (27) and (20) that by construction;

$$\underset{\alpha_i}{\upsilon}(\alpha) = y_i \nabla \underset{\alpha_i}{\psi}(\alpha)$$

Then by applying **Lemma III.1**, we then have $\underset{\alpha_i}{\upsilon}(\alpha^*) = 0$ and using (13) and (19), we have;

$$\begin{aligned}
\underset{\alpha_i}{\upsilon}(\alpha) - \underset{\alpha_i}{\upsilon}(\alpha^*) &= y_i - \sum_{j=1}^{l} \alpha_j y_j K(x_i, x_j) \\
&= y_i - \left(f_E(x_i, \boldsymbol{\alpha}) - b\right) \\
&= y_i - f_E(x_i, \boldsymbol{\alpha}) + b \leq |\ \varepsilon_i\ |
\end{aligned}$$

$$\rightarrow y_i - f_E(x_i, \boldsymbol{\alpha}) \leq \left| \varepsilon_i - b \right| \tag{30}$$

We can then set $|\delta_i| = |\varepsilon_i - b|$ and see from (28) that $\delta_i \rightarrow 0, \forall i = 1..l$ implies that $\alpha \rightarrow \alpha^*$ and thus the limit is proved. If **G** is singular, then $\psi(\boldsymbol{\alpha})$ has a critical point which is non-unique. Hence the theorem holds only if **G** is nonsingular. □

The quantity $y_i - f_E(x_i, \boldsymbol{\alpha})$ is better known as the error on a particular training example, or simply the training error usually denoted by E_i. We refer to this as the dual form of the training error, because incidentally if we examine our discriminant function (5) from the original problem closely, we recover the following;

$$y_i\left(\sum_{i=1}^{n} w_i \varphi(\mathbf{x}_i) + b\right) \geq 1$$

$$\rightarrow y_i\left(\sum_{i=1}^{n} w_i \varphi(\mathbf{x}_i) + b\right) - 1 = \left|\sum_{i=1}^{n} w_i \varphi(\mathbf{x}_i) + b - y_i\right| \geq 0 = \left|y_i - \sum_{i=1}^{n} w_i \varphi(\mathbf{x}_i) + b\right| \leq 0$$

This quantity happens to be the training error expressed in terms of the original hyperplane, which we believe could be refered to fondly as the primal training error form. The results in the **Theorem III.1** show that if we set the dual training error for each example to zero, we would obtain the unique maximizer, $\boldsymbol{\alpha}^*$ to the quadratic function $\psi(\boldsymbol{\alpha})$. In fact, ensuring zero training error using the dual hyperplane form (13) is now shown to be implicitly equivalent to ensuring that the partial gradients of $\psi(\alpha)$ are stationary. Recall that the dual hyperplane form also ensures indirectly that the margin's are maximized in the weight space, **w.**

B. The Linear Program

We can now construct a constrained linear program by applying the same set of constraints and minimizing the sum of the dual training errors instead. We postulate that this is different from direct Empirical Risk Minimization because our use of the dual hyperplane form implicitly enforces capacity control through minimization of the hyperplane margins. We now have the following mathematical program;

$$\text{P.5}\begin{cases} \text{minimize} \quad \Im(\boldsymbol{\alpha}) = \sum_{i=1}^{l}\left|y_i - \left(\sum_{j=1}^{l}\alpha_i y_i K(x_i, x_j) + b\right)\right| \\ \text{subject to} \begin{cases} 0 \leq \alpha_i \leq C \\ \sum_{i=1}^{l}\alpha_i y_i = 0 \end{cases} \end{cases} \tag{31}$$

In fact, (30) in **Theorem III.1** allows us to reconcile (31) with the previous linear program described in Joachims etc.

Rewriting (31), we obtain

$$\Im(\alpha) = \sum_{i=1}^{l}\left|y_i - \left(\sum_{j=1}^{l}\alpha_i y_i K(x_i, x_j) + b\right)\right|$$

$$= \sum_{i=1}^{l} y_i \left|1 - \left(\sum_{j=1}^{l}\alpha_i y_i K(x_i, x_j)\right)\right| + b$$

Each element then satisfies;

$$
\begin{aligned}
y_i \Im(\alpha) &\to \sum_{i=1}^{l} y_i \nabla \psi_i(\boldsymbol{\alpha}) - y_i b \le y_i \left| \varepsilon_i - b \right| \\
&\to \sum_{i=1}^{l} y_i \nabla \psi_i(\boldsymbol{\alpha}) \le y_i \left| \varepsilon_i \right| \\
&\to \sum_{i=1}^{l} y_i \nabla \psi_i(\boldsymbol{\alpha}) - y_i \left| \varepsilon_i \right| \le 0 \\
&\to \sum_{i=1}^{l} d_i \nabla \psi_i(\boldsymbol{\alpha}) \le 0 \\
&\to \nabla \psi(\boldsymbol{\alpha})^T d \le 0
\end{aligned}
$$

since ε_i is the constant from **Theorem III.1** and equality is achieved at optimal α. We can now write the problem as a system of linear equations solved on a compact domain;

$$
\begin{aligned}
& Hu = F \\
& \text{subject to: } u \in Q \\
& H = \begin{pmatrix} K_{11} & \cdots & K_{1l} \\ \vdots & \ddots & \vdots \\ K_{l1} & \cdots & K_{ll} \end{pmatrix} \quad \mathrm{u} = \begin{pmatrix} \mathrm{y}_1 \alpha_1 \\ \vdots \\ \mathrm{y}_l \alpha_l \end{pmatrix} \quad \mathrm{F} = \begin{pmatrix} \mathrm{y}_1 - b \\ \vdots \\ \mathrm{y}_l - b \end{pmatrix}
\end{aligned}
\tag{32}
$$

The matrix H is better known as the Gram matrix, which is symmetrical, and positive definite provided we use a positive definite kernel. The vector of variables **u** could be written solely, in terms of the vector α but this would result in a non-symmetrical matrix H which would be unsuitable for acceleration methods suggested in [14]. The region Q is the feasible region of solutions, which will be discussed further in the next section. It is appropriate at this point to note that the iterates are bounded by this region, and hence optimization speeds using gradient methods or step length searches should have different rates of convergence.

We can further incorporate the equality constraint into the objective function through methods of elimination. This will be useful for writing the problem in matrix form without the step of computing the threshold,b. We note, that the problem formulation commonly used in [5, 15] treats the threshold as a Lagrange Multiplier which allows the problem to be formed as a slightly different set of linear equations. We first apply some elimination methods in the following sections.

1) Direct Elimination

This is intuitively the simplest method which stems from solving equations with multiple variables in linear algebra. Consider the equality constraint expanded as a single equation. We now have the following;

$$
\alpha_1 y_1 + \alpha_2 y_2 ..... \alpha_l y_l = 0 \tag{33}
$$

where the elements $\alpha_j \in \boldsymbol{\alpha}, y_j \in \mathbf{y}, \forall j = 1...l$. Let us rewrite each element in terms of the others, we obtain;

$$
\begin{aligned}
\alpha_1 &= \frac{1}{y_1}\left(\alpha_2 y_2 + \alpha_3 y_3 + ... \alpha_l y_l\right) \\
\alpha_2 &= \frac{1}{y_2}\left(\alpha_1 y_1 + \alpha_3 y_3 + ... \alpha_l y_l\right) \\
&\vdots \\
\alpha_l &= \frac{1}{y_l}\left(\alpha_1 y_1 + \alpha_2 y_2 + ... \alpha_{l-1} y_{l-1}\right)
\end{aligned}
$$

Then a particular Lagrange Multiplier can be expressed in the following relationship;

$$\alpha_j = \frac{1}{y_j}\sum_{\substack{k=1 \\ k\neq j}}^{l}\alpha_k y_k \tag{34}$$

We can now substitute (34) back into the equation of the objective function P.5 to give;

$$\begin{aligned}
\Im(\boldsymbol{\alpha}) &= \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l}\alpha_j y_j K\left(x_i, x_j\right) + b \right)\right] \\
&\rightarrow \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l}\frac{1}{y_j}\sum_{\substack{k=1 \\ k\neq j}}^{l}\alpha_k y_k y_j K\left(x_i, x_j\right) + b \right)\right] \\
&\rightarrow \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l}\sum_{\substack{k=1 \\ k\neq j}}^{l}\alpha_k y_k K\left(x_i, x_j\right) + b \right)\right] \\
&\rightarrow \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l}\left(\sum_{k=1}^{l}\alpha_k y_k - \alpha_j y_j \right) K\left(x_i, x_j\right) + b \right)\right]
\end{aligned}$$

We now obtain the following linear program.

$$\text{P.6}\begin{cases} \text{minimize} \quad \Im(\boldsymbol{\alpha}) = \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l}\left(\sum_{k=1}^{l}\alpha_k y_k - \alpha_j y_j \right) K\left(x_i, x_j\right) + b \right)\right] \\ \text{subject to} \ \{ \ 0 \le \alpha_i \le C \end{cases} \tag{35}$$

2) Generalized Elimination Method

The direct elimination method might not be the best method or even practical to implement. A generalized elimination method[8] exists which uses a linear transformation on the variable, in this case the Lagrange Multiplier to reduce a problem with equality constraints to an unconstrained problem. In our case, we show that this method reduces to the direct elimination method. We first note that the equality constraint can be written in vector form as;

$$\mathbf{y}^T\boldsymbol{\alpha} = 0 \tag{36}$$

Let **A** and **B** be a $l\times 1$ vector and a $l\times(l-1)$ matrix respectively such that $\left[A:B\right]$ is a non-singular matrix ;

$$\left[A:B\right] = \begin{bmatrix} a_1 & b_{11} & \cdots & b_{1,l-1} \\ \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ a_l & b_{1l} & \cdots & b_{l,l-1} \end{bmatrix}$$

Furthermore, let us choose **A** and **B** so that $\mathbf{Y}^T\mathbf{A} = 1$ and $\mathbf{Y}^T\mathbf{B} = \mathbf{0}$. **A** is generally a $l\times k$ matrix if there happens to be k-equality constraints, but in our case it is a vector due to the single equality constraint. In any case, let us treat **A** as a generalized left inverse for $\mathbf{Y}$ so that we can say that;

$$\begin{aligned}
&\mathbf{Y}^T\boldsymbol{\alpha} = 0 \\
&\rightarrow \boldsymbol{\alpha} = \mathbf{A}^T\mathbf{0} = \mathbf{0}
\end{aligned}$$

We state that this is simply the trivial solution of $\boldsymbol{\alpha}$ and is of course not unique because we can define other feasible points by taking any feasible direction from the trivial solution, i.e

$$\boldsymbol{\alpha} = \mathbf{A}^T\mathbf{0} + \boldsymbol{\kappa} \tag{37}$$

provided of course that $\boldsymbol{\kappa}$ is a direction in the linear space $\left\{\boldsymbol{\kappa}\middle|\mathbf{Y}^T\boldsymbol{\kappa}=0\right\}$ with dimension $l-1$. Now, let the matrix **B**, contain linearly independent coloumns, $\left\{\mathbf{b}_1,\mathbf{b}_2...\mathbf{b}_{l-1}\right\}$ which can be viewed as reduced coordinate directions and the scalars λ_i form the reduced variables of these coordinate directions. Then any feasible direction, $\boldsymbol{\kappa}$ is given by;

$$\boldsymbol{\kappa}=\mathbf{B}\boldsymbol{\lambda}=\sum_{i=1}^{l-1}\mathbf{b}_i\lambda_i \tag{38}$$

Fig III.1 shows an example of the feasible direction in terms of reduced coordinates when $\boldsymbol{\alpha}\in\Re^3$.

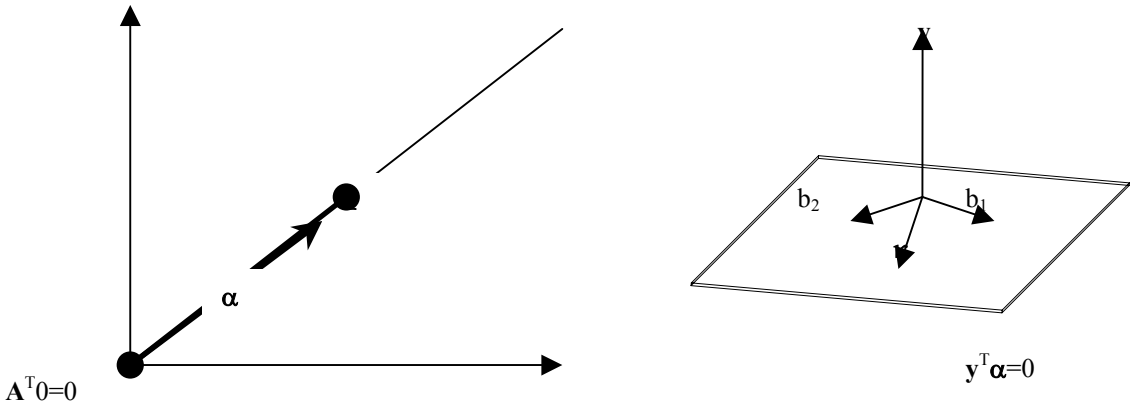

Fig III.1: The feasible points of $\boldsymbol{\alpha}$ can be found by taking a step from the trivial point in the feasible direction $\boldsymbol{\kappa}$. The reduced coordinate vectors b_1 and b_2 determine the feasible direction, $\boldsymbol{\kappa}$.

The problem that remains now is to choose a suitable matrix B which has linearly independent coloumns. We propose one such matrix **B** derived from the condition $\mathbf{Y}^T\mathbf{B}=\mathbf{0}$ having the following form;

$$\mathbf{B}=\begin{bmatrix}\alpha_2 & \alpha_3 & \cdots & \alpha_l\\ \alpha_3 & \alpha_4 & \cdots & \alpha_1\\ \vdots & \vdots & \cdots & \vdots\\ \alpha_1 & \alpha_2 & \cdots & \alpha_{l-1}\end{bmatrix} \tag{39}$$

Simplyfing (37), we now have the following $\boldsymbol{\alpha}$ vector expressed in reduced coordinate directions as;

$$\boldsymbol{\alpha}=\mathbf{B}\boldsymbol{\lambda} \tag{40}$$

We now can substitute this relationship into (31) to give us the following program;

$$\text{P.7}\begin{cases}\text{minimize}\quad \Im(\boldsymbol{\alpha})=\sum_{i=1}^{l}\left[y_i-\left(\sum_{j=1}^{l}\left(\sum_{k=1,k\neq j}^{l-1}\lambda_k\alpha_k\right)y_jK\left(x_i,x_j\right)+b\right)\right]\\ \text{subject to}\quad \left\{\ 0\le\alpha_k\le\dfrac{C(l-1)}{\lambda_k}\right.\end{cases} \tag{41}$$

We note that $\boldsymbol{\kappa}$ is sometimes known as a feasible correction.

3) Geometrical Elimination

Our equality constraint is unique because we can also use simple plane geometry to incorporate it into the objective function. In our previous discussion, we treated the equality constraint as a hyperplane which had a section lying in a convex polyhedra. We now expand this hyperplane idea by rewriting (36) in the general hyperplane equation form;

$$\mathbf{y}.(\boldsymbol{\alpha}-\mathbf{p})=0 \tag{42}$$

The normal to the hyperplane is the vector, **y** and any point on the hyperplane satisfies equation (42). The point, **p** is any point lying on the hyperplane surface. Since the hyperplane passes through the origin, then $\mathbf{p}=\mathbf{0}$ is one such point. All we have to do now is to find the general parametric equation of the feasible alpha points lying on the plane and substitute it back into the objective function. We would have then incorporated the linear equality constraint into the objective function.

First, let us recall that a hyperplane in the space $\Re^l$ is a $l-1$-dimensional affine subspace of $\Re^l$. We expand (42) to give ,

$$y_1(\alpha_1-p_1)+y_2(\alpha_2-p_2).....+y_l(\alpha_l-p_l)=0 \tag{43}$$

Now, we can choose any nonzero scalar, y_i to solve for the corrosponding point (α_i-p_i). Since this is classification and the scalars are all either 1 or -1, we could choose any point. For the sake of this discussion, let us solve for the point α_l. We get;

$$\alpha_{l=}-\frac{y_1}{y_l}(\alpha_1-p_1)-\frac{y_2}{y_l}(\alpha_2-p_2).....-\frac{y_{l-1}}{y_l}(\alpha_{l-1}-p_{l-1})$$

If we set , $\{\alpha_1,\alpha_2....\alpha_{l-1}\}=\{\lambda_1,\lambda_2....\lambda_{l-1}\}$ where $-\infty<\lambda_i<\infty,\forall i=1..l$, we get the following;

$$\begin{aligned}\boldsymbol{\alpha}&=\{\alpha_1,\alpha_2....\alpha_l\}\\&=\left\{\lambda_1,\lambda_2....\lambda_{l-1},-\frac{y_1}{y_l}(\lambda_1-p_1)-\frac{y_2}{y_l}(\lambda_2-p_2).....-\frac{y_{l-1}}{y_l}(\lambda_{l-1}-p_{l-1})\right\}\\&=\lambda_1\begin{pmatrix}1\\0\\\vdots\\-\frac{y_1}{y_l}\left(\lambda_1-\frac{p_1}{\lambda_1}\right)\end{pmatrix}+\lambda_2\begin{pmatrix}0\\1\\\vdots\\-\frac{y_2}{y_l}\left(\lambda_2-\frac{p_2}{\lambda_2}\right)\end{pmatrix}\cdots+\lambda_{l-1}\begin{pmatrix}0\\\vdots\\1\\-\frac{y_{l-1}}{y_l}\left(\lambda_{l-1}-\frac{p_{l-1}}{\lambda_{l-1}}\right)\end{pmatrix}\end{aligned} \tag{44}$$

We can now simply substitute this relationship into (31) to get the objective function as;

$$\begin{aligned}\Im(\boldsymbol{\alpha}) &= \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l} \alpha_j y_j K(x_i, x_j) + b \right) \right] \\ &\rightarrow \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l-1} \lambda_j y_j K(x_i, x_j) - \sum_{k=1}^{k-1} \frac{y_k}{y_l} \left(\lambda_k - \frac{p_k}{\lambda_k} \right) y_l K(x_i, x_l) + b \right) \right] \\ &\rightarrow \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l-1} \lambda_j y_j K(x_i, x_j) - \sum_{k=1}^{k-1} y_k \left(\lambda_k - \frac{p_k}{\lambda_k} \right) K(x_i, x_l) + b \right) \right] \\ &\rightarrow \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l-1} \lambda_j y_j K(x_i, x_j) - \sum_{k=1}^{k-1} \lambda_k y_k K(x_i, x_l) + b \right) \right]\end{aligned}$$

We obtain the last step by using the origin as a known point, **p** on the hyperplane. We then obtain the linear program as;

$$\text{P.8}\begin{cases} \text{minimize} \quad \Im(\boldsymbol{\alpha}) = \sum_{i=1}^{l}\left[y_i - \left(\sum_{j=1}^{l-1} \lambda_j y_j K(x_i, x_j) - \sum_{k=1}^{k-1} \lambda_k y_k K(x_i, x_l) + b \right) \right] \\ \text{subject to} \ \{ \ 0 \le \lambda_k \le C \end{cases} \tag{45}$$

Let us make a note that this is closely similar in nature to the results of the direct elimination and the generalized elimination method. In fact, P6-P8 are the same linear program obtained through different methods of elimination. The geometrical method gives us a more general form of the linear program in terms of known points **p** which are also feasible solutions $\boldsymbol{\alpha}$. This will be particularly useful for the design of a geometrical algorithm which solves the SVM problem based on search directions on the hyperplane enforced by the equality constraint.

IV. Geometrical View of SVM solutions

A. *Feasible Region of Solutions*

The solution to the P.3-P.8, must lie in the feasible region, $\mathbb{Q}$ defined by the intersections of the constraint set. We denote the region algebraically as follows;

$$\begin{aligned}&\mathbb{Q} = H(\alpha) \cap \bigcap_{i=1}^{l} g(\alpha_i) \\ &g(\alpha) = \sum_{i=1}^{l} \alpha_i y_i, H = \left(\alpha_j \middle| (\alpha_j - C) \le 0 \right) \\ &\text{where } i, j, \alpha_j \ge 0, C > 0\end{aligned} \tag{46}$$

The set H is of size l, and forms a l-sided convex polyhedra lying entirely in the positive quadrant of the $\Re^l$ space with each element h_j forming a boundary plane. Recall that l is the number of training examples and hence directly influences the dimension of the polyhedra. The region dictated by the equality constraint g is a linear manifold or hyperplane intersecting the polyhedra passing through the origin. Our optimal solution, $\boldsymbol{\alpha}^*$ must lie in the region $\mathbb{Q}$ or in other words the optimal vector of Lagrange Multipliers is a convex subset of $\mathbb{Q}$, i.e $\boldsymbol{\alpha}^* \in \mathbb{Q}$.

We illustrate this in the following example, for $l = 3$, we have the following 3-dimension polyhedra having the form of a cube in Fig IV.1 below. The linear manifold is now the plane

$g(\alpha)=\sum_{i=1}^{3}\alpha_i y_i$ which intersects the cube and the feasible region, $\mathbb{Q}$ of solutions is now restricted to the surface of the plane contained within the region of the cube.

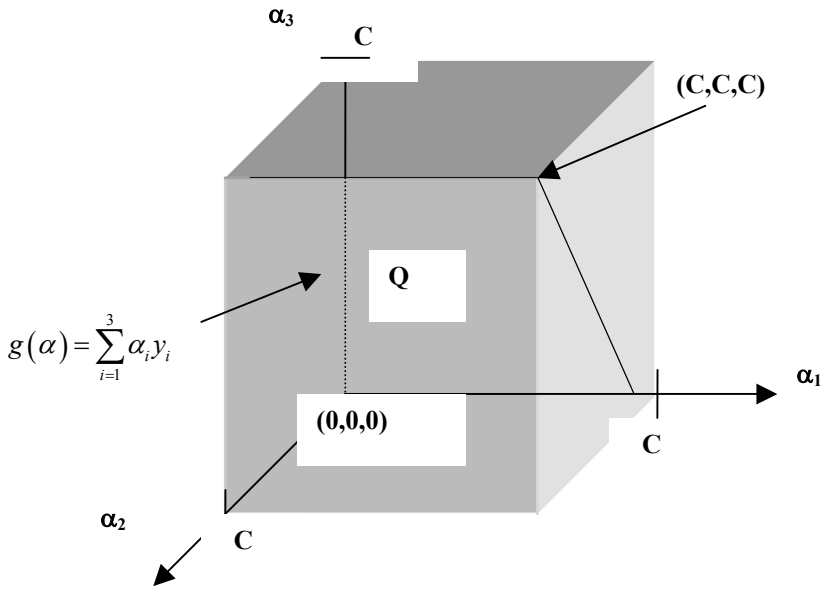

Fig IV.1: The feasible region **Q**, for a classification problem with just 3 Lagrange Multipliers.

The problem with our now bounded region of solutions is that there could exist many linear manifolds (possibly infinite) all parallel to each other which gives a local minimum for P.3 and P.4. Optimization programs may then have slower rates of convergence if the solution set oscillates between these manifolds. In fact, Keerthi[16] has indirectly noted this problem with the Sequential Minimal Optimization(SMO) algorithm which he attributes to the pessimistic updates of the threshold, **b** in the original SMO. If we look at the derivations of P.3 and P.4, we see that the linear restriction arises directly from treating the threshold **b**, as a variable and is further evidence of our assertion that many possible linear manifolds exist which give equally good feasible regions. Adjusting the threshold **b** during optimization, is then geometrically interpreted as alternating between the possible linear manifolds whilst looking for the optimal solution. Note that the position of the final seperating hyperplane is also determined by the threshold **b**.

There are a number of ways to avoid this problem. Firstly, we could remove the linear restriction completely from the quadratic program and end up with a feasible region which is a convex polyhedra due to H alone. We have discovered one way through the use of a special class of semiparametric functions which we have previously proposed in [14], but work on the generalization of this model has not yet been attempted. Others chose to incorporate the threshold as a Lagrange Multiplier instead[17] but this still requires explicit computation of b in a different form. We have proposed several ways to eliminate the equality constraint. We may obtain a local solution instead but end up with an easier implementation with possibly better rates of convergence. However, further investigation is need to assertain the effect of elimination on the generalization of the trained model.

B. Dual Hyperplane Form

It is interesting to note that P.3 simplifies P.2 greatly by reducing the objective function to just one variable, that is the Lagrange Multiplier. In fact, what is happening is that we are now looking for the solution to $\Im(\mathbf{w}^*,\boldsymbol{\alpha})$ where the weight vector, $\mathbf{w}^*$ is optimal provided $\Im(\mathbf{w}^*,\boldsymbol{\alpha}^*)$ is a saddle point of $\Im$. This is guranteed anyway since $\mathbf{w}^*$ is a saddle point obtained from the stationary gradient of P.2 in (10). Then by Kuhn and Tucker's saddle point theorem[18], we are assured that there exists an

optimal value for $\boldsymbol{\alpha}_0 \geq 0$ so that $\left(\mathbf{w}^*, \boldsymbol{\alpha}_0\right)$ is a saddle point for P.3 and hence also a solution to P.2. This allows us to assert that the dual hyperplane form (13) is derived only when $\mathbf{w}$ is a saddle point for P.2 and hence implicitly embodies the minimization of the hyperplane margin. We note though that a modfication of (13) has been used by Osuna[19] to approximate the decision surface, but his form can be shown to reduce to the original form by setting $\beta_i = \alpha_i + \delta_i$ where δ_i is some slack or tolerance variable. In our case, we argue that using this form in any objective function, $\Im(w, \alpha)$ implies searching for the saddle point of $\boldsymbol{\alpha}$, given that we have found the saddle point of $\mathbf{w}$ or in short, maximized the margin of the hyperplane**.** Hence we conclude that the linear minimization programs solved by Joachims et. al. and also our proposed system of equations indirectly solve the quadratic Support Vector Machine program for classification.

V. Discussion

The derivation of programs P6-P8 is also indirectly related to solving a *linear complementarity problem[8]* which has been applied in some game theory and boundary value problems. Dantzig-Wolfe have used a principle pivoting method[9] which is an iterative method to find the solution to these kind of problems but in general all methods are closely related to solving a Linear Program and involves row operations on the equations in the objective function. Mangasarian[20, 21] further examines the solutions to linear complementarity programs if the objective function is concave. Concavity gurantees a solution which is difficult to prove for functions otherwise i.e nonconcave. However, the feasible region for our linear programs is bounded and it is known that there is a solution with every minimizer a relative boundary point and at least one minimizer an extreme point[22]. Geometrically, extreme points are vertices of the convex polyhedra which is the feasible region of solutions and if the unique minimizer to the unconstrained problem lies outside the polyhedra, the constrained minimizer lies on a boundary plane with at least one such minimizer being the vertex of the polyhedra.

The method of geometrical elimination which resulted in P8 interestingly gives rise to a number of possibilities for incremental SVM methods. Incremental methods which investigate different C parameters actually look at convex polyhedra of different sizes but the region of feasible solutions remain bounded to the surface of the hyperplane (42). The solution vector, $\boldsymbol{\alpha}$ of one C is now the known point $\mathbf{p}$ for the next value of C. Based on this, it is possible to calculate notions of distances between optimal solutions using the distances between the points of the surface or even angles between different hyperplanes resulting from different normal vectors $\mathbf{y}$ which arise due to additional input data**.**

The matrix equations of the linear program allows us to apply modified linear iterative methods and other methods that operate on sets of linear equations. This was meant to demonstrate a fast and simple way of implementing Support Vector Machines for classification. The single iterate updates gives us the possiblity of using heuristics to choose the next point of update. We can cycle through the subsets of Support Vectors, arrange them in order of magnitudes or size of change and so on. Different datasets work well with different heuristics and we have yet to gain a full comprehension of the reasons for this observation. The convergence of this method has yet to be proved but we conjecture that it is similar to that of decomposition algorithms and activeset methods. The analysis of the rates of convergence of such methods will be the direction of our future work. In particular, we would be investigating the effect of ordering the elements in the working subsets on the convergence speeds.

VI. Conclusion

We have transformed the quadratic program SVM classification problem to a set of linear equations which can be represented by a matrix equation solved on a compact set. Solving the system indirectly maximizes the margin of the hyperplane by using the dual hyperplane form. Future work would involve investigating this program applied to the Support Vector Regression model. Another interesting direction would be to investigate the effect of fixing the linear manifold on generalization accuracy. In fact, our previous work on semiparametric SVM classifiers can be interpreted as one example of fixing the linear manifold. The geometrical interpretation of the region of feasible solutions provides interesting ideas for the construction of optimization algorithms based on gradient flows in this bounded region.

REFERENCES

[1] V. N. Vapnik, *The nature of statistical learning theory*, 2nd ed. New York: Springer, 2000.

[2] B. Schölkopf and A. J. Smola, *Learning with kernels : support vector machines, regularization, optimization, and beyond*. Cambridge, Mass.: MIT Press, 2002.

[3] C.-J. Lin, "LIBSVM," http://www.csie.ntu.edu.tw/~cjlin/.

[4] J. Platt, "Fast Training of Support Vector Machines Using Sequential Minimal Optimization," in *Advances in Kernel Methods-Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds.: Cambridge MIT Press, 1998, pp. 185-208.

[5] T. Joachims, "Making Large Scale Support Vector Machine Learning Practical," in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds.: Cambridge , MIT Press, 1998, pp. 169-184.

[6] C.-C. Chang, C.-W. Hsu, and C.-J. Lin, "The analysis of decomposition methods for support vector machines," *Neural Networks, IEEE Transactions on*, vol. 11, pp. 1003-1008, 2000.

[7] B. Kristin and B. E., "Duality and Geometry in SVM Classifiers," presented at International Conference on Machine Learning, San Francisco, 2000.

[8] R. Fletcher, *Practical methods of optimization*. Chichester [Eng.] ; New York: J. Wiley, 1981.

[9] P. E. Gill, W. Murray, and M. H. Wright, *Practical optimization*. London ; New York: Academic Press, 1981.

[10] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex analysis and minimization algorithms*. Berlin ; New York: Springer-Verlag, 1993.

[11] R. T. Rockafellar, *Convex analysis*. Princeton, N.J.,: Princeton University Press, 1970.

[12] C.-J. Lin, "A formal analysis of stopping criteria of decomposition methods for support vector machines," *Neural Networks, IEEE Transactions on*, vol. 13, pp. 1045-1052, 2002.

[13] E. F. Osuna, R.; Girosit, F., "Training support vector machines: an application to face detection," presented at Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on, 1997.

[14] D. Lai, M. Palaniswami, and N. Mani, "Fast Linear Optimization with Automatically Biased Support Vector Machines," Monash University MECE-4-2003, 2003.

[15] C.-J. Lin, "On the convergence of the decomposition method for support vector machines," *Neural Networks, IEEE Transactions on*, vol. 12, pp. 1288-1298, 2001.

[16] S. K. S. S.S. Keerthi, C. Bhattacharyya and K.R.K. Murthy, "Improvements to Platt's SMO algorithm for SVM classifier design,," Control Division, Dept. of Mechanical Engineering,National University of Singapore CD-99-14, 1999.

[17] C.-J. Lin, "Linear convergence for a decomposition method for Support Vector Machines," November 2001.

[18] J. Stoer and C. Witzgall, *Convexity and optimization in finite dimensions*. Berlin, New York,: Springer-Verlag, 1970.

[19] B. Schölkopf, C. J. C. Burges, and A. J. Smola, *Advances in kernel methods : support vector learning*. Cambridge, Mass.: MIT Press, 1999.

[20] O. L. Mangasarian, "Machine Learning via Polyhedral Concave Minimization," 95-20, 1995.

[21] O. L. Mangasarian, "Solution of General Linear Complementarity Problems via Nondifferentiable Concave Minimization," 96-10, 1996.

[22] W. Kaplan, *Maxima and minima with applications : practical optimization and duality*. New York: Wiley, 1999.