A Proof of Sparseness, Optimality, and Convergence of an LP-SVR

Pablo Rivas, *Senior, IEEE*, and Korn Sooksatra
School of Engineering and Computer Science
Department of Computer Science, Baylor University
Email: Pablo_Rivas@Baylor.edu

***Abstract*—This article presents a proof of convergence and sparsity of a linear programming support vector machine for regression. First, the Support Vector Regression (SVR) problem is posed as a linear programming problem modeled on a primal and dual fashion leading to the definitions of optimality. Second, we describe a sequential optimization method based on variables decomposition, constraints decomposition, and primal-dual interior point methods for solving large-scale regression/classification problems. Third, based on the methodology, we present proof of convergence and optimality conditions of the sequential optimization and its ability to produce sparse solutions.**

***Index Terms*—machine learning theory, support vector machines, support vector regression, interior point methods, linear programming**

I. Introduction

Machine Learning is a tool which has been applied to solve many problems nowadays and includes various interesting algorithms (e.g. Neural Network, Support Vector Machine (SVM), Deep Learning and Genetic Algorithm). The algorithm which is focused in this paper is SVM because it is one of traditional algorithms in the field of machine learning and has been widely used in several applications from the past until the present (e.g. Face detection [1], Text and hypertext categorization [2], Classification of images [3], Bioinformatics [4] and Handwriting recognition [5]).

In addition to classification problems, SVM can also be applied to address regression problems, and this is called Support Vector Regression (SVR). Recently, researchers have used SVR to find a solution for practical problems. In 2016, Errami and Rziza [6] proposed a method using SVR to improve a pedestrian detection after performing features extraction. In 2018, Hou et.al. [7] utilized SVR optimized by improved fruit fly optimization algorithm in order to predict the stock prices, and Azad et.al. [8] discovered a solution to forecast electricity peak load using SVR for urban development. In 2019, Wu et.al [9] applied Twin SVR to find a pair of non-parallel lower and upper bound functions for feature selection problems. In 2020, Li et.al. [10] proposed a clustering-based SVR approach for forecasting bus passenger flows. They applied Affinity Propagation(AP) to cluster the input data including training data and test data, and then in each cluster, the optimized least square SVR predicts the results for the test data.

Although SVR is significantly popular for being applied to solve some problems or improve a number of solutions as mentioned earlier, it still needs to be developed in term of a few factors (e.g. time complexity with a large-scale data) and be proved that it can have some principal properties (e.g. sparseness).

This paper formulates a non-linear SVR into a linear programming SVR (LP-SVR) by applying primal, dual and KKT conditions and also shows an algorithm consisting of several methods to optimize LP-SVR (e.g. variable decomposition, constraints decomposition and interior point methods). At last the theoretical proofs which show that the algorithm can meet sparseness, optimality and convergence are provided. Therefore, the contributions of this paper can be concluded as the following:

- LP-SVR is formulated from a non-linear SVR.
- The algorithm for large-scale LP-SVR is proposed.
- The theoretical proofs of sparseness and convergence of the proposed algorithm are shown.

The remainder of this paper is organized as followed. LP-SVR is formulated in Section II, and Section III shows the proof of sparseness of LP-SVR. In Section IV, the proposed algorithm and the proof of convergence with respect to variable and constraints decomposition are explained, and the proof of convergence of interior point method is provided in Section V. Finally, everything is concluded in Section VI.

II. Large-Scale LP-SVR Formulation

Let us assume we have training samples $\{(\mathbf{x}_i, d_i)\}_{i=1}^N$, where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^M$ is a regressor and d is the desired output. Then, one can define a non-linear SVR prediction function:

$$d_j \equiv f(\mathbf{x}_j) = \sum_{i=1}^{N} (\alpha_i^+ - \alpha_i^-) k(\mathbf{x}_i, \mathbf{x}_j) + (b^+ - b^-), \quad (1)$$

where $\boldsymbol{\alpha}^+, \boldsymbol{\alpha}^- \in \mathbb{R}_+^N$; $b^+, b^- \in \mathbb{R}_+$; $k(\cdot,\cdot)$ is a valid kernel function [11], [12], e.g. a polynomial kernel; $\boldsymbol{\alpha}^+ = \max\{\boldsymbol{\alpha}, 0\}$, $\boldsymbol{\alpha}^- = \max\{-\boldsymbol{\alpha}, 0\}$; $b^+ = \max\{b, 0\}$, $b^- = \max\{-b, 0\}$; $\boldsymbol{\alpha} \in \mathbb{R}^N$; and $b \in \mathbb{R}$. Kernel functions map the input feature vectors to the kernel-induced feature space denoted $\mathcal{H}$ since these kernel functions follow the properties of Hilbert spaces [11]. The kernel-induced feature space for non-linear SVR can be defined as $\mathcal{H} = \left\{ f(\mathbf{x}_j) : d_j = \sum_{i=1}^N (\alpha_i^+ - \alpha_i^-) k(\mathbf{x}_i, \mathbf{x}_j) + (b^+ - b^-) \right\}$, for all $\mathbf{x}_j \in \mathbb{R}^M$ and $j = \{1, 2, \ldots, N\}$. The objective is to find the set of parameters $\boldsymbol{\alpha}$ and b. One can find these parameters via constrained optimization.

A. Primal, Dual, and KKT Conditions

First, let us assume the mapping

$$k(\mathbf{x}_i, \mathbf{x}_j) : \mathcal{X}^{(N\times M)\times(M\times N)} \mapsto \mathcal{H}^{N\times N}.$$

Then, assume that the slack variables ξ_i, ξ_i^* can be expressed as simply $2\xi_i$ (since $\xi_i\xi_i^* = 0$). Then, let us introduce a slack variable $\mathbf{u}$ to get rid of the inequalities of the traditional SVR problem. Based on the above, the following optimization problem can be proposed:

$$\begin{array}{ll} \min\limits_{\boldsymbol{\alpha}^{\pm},b^{\pm},\boldsymbol{\xi},\mathbf{u}} & \sum_{i=1}^{N}\left(\alpha_i^+ + \alpha_i^- + 2C\xi_i\right) \\ \text{s.t.} & \left\{\begin{array}{rcl} -\sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)k(\mathbf{x}_j,\mathbf{x}_i) & & \\ \quad -b^+ + b^- - \xi_j + u_j & = & \epsilon - d_j \\ \sum_{i=1}^{N}(\alpha_i^+ - \alpha_i^-)k(\mathbf{x}_j,\mathbf{x}_i) & & \\ \quad +b^+ - b^- - \xi_j + u_j & = & \epsilon + d_j \\ \alpha_j^+, \alpha_j^-, b^+, b^-, \xi_j, u_j & \geq & 0 \end{array}\right. \\ \text{for all} & j = 1, 2, \ldots, N. \end{array} \tag{2}$$

Problem (2) can be posed as the linear programming problem in its *canonical* for, which is the following:

$$\begin{array}{ll} \min\limits_{\mathbf{z}\in\mathbb{R}^n} & \mathbf{c}^T\mathbf{z} \\ \text{s.t.} & \left\{\begin{array}{rcl} \mathbf{A}\mathbf{z} & = & \mathbf{b} \\ \mathbf{z} & \geq & \mathbf{0}, \end{array}\right. \end{array} \tag{3}$$

where $\mathbf{z} \in \mathbb{R}^n$ is a vector containing the unknowns, $\mathbf{c} \in \mathbb{R}^n$ and $\mathbf{b} \in \mathbb{R}^m$ are vectors of known parameters, and $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a matrix of known coefficients associated to $\mathbf{z}$ in a linear relationship.

Thus, Problem (2) can be posed as a linear programming problem by defining the following equalities:

$$\mathbf{A} = \begin{pmatrix} -\mathbf{K} & \mathbf{K} & -1 & 1 & -\mathbf{I} & \mathbf{I} \\ \mathbf{K} & -\mathbf{K} & 1 & -1 & -\mathbf{I} & \mathbf{I} \end{pmatrix}, \tag{4a}$$

$$\mathbf{b} = \begin{pmatrix} \mathbf{1}\epsilon - \mathbf{d} \\ \mathbf{1}\epsilon + \mathbf{d} \end{pmatrix}, \tag{4b}$$

$$\mathbf{z} = \begin{pmatrix} \boldsymbol{\alpha}^+ & \boldsymbol{\alpha}^- & b^+ & b^- & \boldsymbol{\xi} & \mathbf{u} \end{pmatrix}^T, \tag{4c}$$

$$\mathbf{c} = \begin{pmatrix} \mathbf{1} & \mathbf{1} & 0 & 0 & \mathbf{2C} & \mathbf{0} \end{pmatrix}^T, \tag{4d}$$

where $\mathbf{A} \in \mathbb{R}^{(2N)\times(4N+2)}$, $\mathbf{b} \in \mathbb{R}^{2N}$, $\mathbf{z}, \mathbf{c} \in \mathbb{R}^{4N+2}$. If we use the above equalities, then problems (3) and (2) are identical, and we can claim that the problem has been posed as an LP problem.

Similarly, using equalities (4a)-(4d), we can obtain the dual problem of (2) as follows:

$$\begin{array}{ll} \max\limits_{\boldsymbol{\lambda}} & \mathbf{b}^T\boldsymbol{\lambda} \\ \text{s.t.} & \left\{\begin{array}{rcl} \mathbf{A}^T\boldsymbol{\lambda} + \mathbf{s} & = & \mathbf{c} \\ \mathbf{s} & \geq & \mathbf{0}, \end{array}\right. \end{array} \tag{5}$$

where $\boldsymbol{\lambda}$ is a vector of dual variables defined over $\mathbb{R}^{2N}$, and $\mathbf{s}$ is a slack vector variable in $\mathbb{R}^{4N+2}$.

Similarly, for the primal (2) and dual (5), the KKT conditions are defined as follows:

$$\mathbf{A}^T\boldsymbol{\lambda} + \mathbf{s} = \mathbf{c}, \tag{6a}$$

$$\mathbf{A}\mathbf{z} = \mathbf{b}, \tag{6b}$$

$$z_i s_i = 0, \tag{6c}$$

$$(\mathbf{z}, \mathbf{s}) \geq \mathbf{0}, \tag{6d}$$

$$\text{for } \; i = 1, 2, \ldots, n,$$

where the equality $z_i s_i$ implies that one of both variables must be zero. This equality will be referred to as the *complementarity condition*. Note that the KKT conditions depend on the variables $(\mathbf{z}, \boldsymbol{\lambda}, \mathbf{s})$, and if the set of solutions $(\mathbf{z}^*, \boldsymbol{\lambda}^*, \mathbf{s}^*)$ satisfy all the conditions, the problem is said to be solved. The set $(\mathbf{z}^*, \boldsymbol{\lambda}^*, \mathbf{s}^*)$ is known as a *primal-dual solution*.

III. Optimality and Proof of Sparseness

Let $\mathbf{z}^*$ be the solution to the primal problem (2), and let $(\boldsymbol{\lambda}^*, \mathbf{s}^*)$ be the solution to the dual problem (5). The proposed LP-SVR exhibits two important properties. First, that it has a *global solution*. That is, if $\mathbf{z}^*$ is a minimum for problem (2), then $\mathbf{z}^*$ is a global minimum since problem (2) is a convex problem (*i.e.*, a linear programming problem) [13]–[16].

Second, its *optimality conditions* are well defined. That is, for problem (2), the KKT conditions (6a)-(6d) are necessary and sufficient for optimality since $(\mathbf{z}^*, \boldsymbol{\lambda}^*, \mathbf{s}^*)$ is a solution to the primal (2) and dual (5), then it follows that the KKT conditions (6a)-(6d) are necessary and sufficient for optimality [13], [14], [16], [17].

One concern of the work presented here is to demonstrate that the solution of the proposed LP-SVR is better than that of SVRs in the sense of solution sparseness. Sparseness in a solution is desired because any SV-based model relies on actual feature vectors $\mathbf{x}_i$ to define the optimal set of model parameters (α_i, b) for all $i \;:\; \alpha_i \neq 0$. Especially since the feature vectors $\mathbf{x}_i$ are required for kernel distances as shown in (1).

In 2010, Zhang, *et al.* performed a comprehensive study in regard to SVM sparseness [18]. The authors explain that sparseness of a learning machine depends on the problem and the precision of the solution. Then, the authors prove (see [18] Theorems 1 and 2) that, for their proposed LP-SVR, the solution is always sparser than regular SVRs. Therefore, based on Zhang's theorems we prove that our formulation is also sparser than regular SVRs. Before the proof, the following definitions are necessary:

Definition 1 (Support Vectors)**.** *Let $\mathcal{T} = \{\mathbf{x}_i, d_i\}_{i=1}^N$ be a training set; let $\mathbf{z}$ be a solution to problem (2); and let (1) be the regression function for problem (2). Then,*

1) $\mathcal{V}_S = \{\mathbf{x}_i \;:\; d_i - \epsilon < f(\mathbf{x}_i) < d_i + \epsilon\}$ *defines the set of Saturated Support Vectors (SSVs).*
2) $\mathcal{V}_E = \{\mathbf{x}_i \;:\; f(\mathbf{x}_i) = d_i + \epsilon, \text{ or } f(\mathbf{x}_i) = d_i - \epsilon\}$ *defines the set of Exact Support Vectors (ESVs).*
3) $\mathcal{V}_N = \{\mathbf{x}_i \;:\; f(\mathbf{x}_i) < d_i + \epsilon, \text{ or } f(\mathbf{x}_i) > d_i - \epsilon\}$ *defines the set of Non-Support Vectors (NSVs).*

4) $\mathcal{V}_\alpha = \{\mathbf{x}_i \ : \ \alpha_i \neq 0\}$ *defines the set of Sparse Vectors (SPVs).*
5) $N = |\mathcal{V}_S| + |\mathcal{V}_E| + |\mathcal{V}_N|$.
6) $\mathcal{S} = \mathcal{V}_S \cup \mathcal{V}_E$*, means that the union of the SSVs and the ESVs is the set of Support Vectors (SVs).*
7) $\mathcal{A} = \{\alpha_i \ : \ \alpha_i \neq 0\}$ *denotes the set of Non-zero Coefficients of the decision function (1) and of problem (2).*

Theorem 1 (Based on Zhang, *et al.* [18]: ESV Bound)**.** *Given an optimal solution* $\mathbf{z}^*$ *to (2), the number of nonzero* α_i *coefficients of (2) has the following upper bound:*

$$|\mathcal{A}| \leq |\mathcal{V}_E|. \tag{7}$$

for all $i \ : \ \alpha_i \neq 0$.

Proof. Consider the LP problem

$$\min_{\mathbf{z}^*} \quad \mathbf{c}^T\mathbf{z}^* \tag{8a}$$
$$\text{s.t. } \mathbf{A}\mathbf{z}^* = \mathbf{b} \tag{8b}$$
$$\mathbf{z}^* \geq \mathbf{0}, \tag{8c}$$

which is equivalent to (2), where $\mathbf{A} \in \mathbb{R}^{m\times n}$ and $\mathbf{z}^* \in \mathbb{R}^n$. Then, for $\boldsymbol{\alpha}^*$ we can define the following equality

$$\begin{aligned} |\mathcal{A}| =& |\{\alpha_j^* \ : \ \alpha_j^* \neq 0, \ j = 1, 2, \ldots, N\}| \\ &+ |\{\alpha_j^{+*} \ : \ \alpha_j^{+*} \neq 0, \ j = 1, 2, \ldots, N\}| \\ &+ |\{\alpha_j^{-*} \ : \ \alpha_j^{-*} \neq 0, \ j = 1, 2, \ldots, N\}|. \end{aligned} \tag{9}$$

Now, let

$$NPE(\mathbf{z}^*) \leq m, \tag{10}$$

denote an upper bound to the number of positive elements in $\mathbf{z}^*$. By (10), there are at most $2N$ basic variables in $\mathbf{z}^*$ that can take nonzero values; the other non-basic variables take zeros. Among these basic variables, there are $|\mathcal{V}_S|$ of $\boldsymbol{\xi}^* > 0$ and $|\mathcal{V}_S|$ of $\mathbf{u}^* > 0$, $|\mathcal{V}_N|$ of $\mathbf{u}^* > 0$, and $|\mathcal{V}_E|$ of $\mathbf{u}^* = 2\epsilon$ apart from the nonzero coefficients in $\alpha_j^{+*}, \alpha_j^{-*}, \ j = \{1, 2, \ldots, N\}$. As a result, the number of nonzero coefficients is

$$|\mathcal{A}| = |\{\alpha_j^* \ : \ \alpha_j^* \neq 0, \ j = 1, 2, \ldots, N\}| \tag{11}$$
$$\leq 2N - 2|\mathcal{V}_S| - |\mathcal{V}_N| - |\mathcal{V}_E|. \tag{12}$$

Since $N = |\mathcal{V}_S| + |\mathcal{V}_E| + |\mathcal{V}_N|$, we have that

$$|\mathcal{A}| \leq |\mathcal{V}_E| + |\mathcal{V}_N|. \tag{13}$$

This completes the proof. □

Therefore, we can say that given an optimal solution $\mathbf{z}^*$ to (2), the number of nonzero α_i coefficients of (2) has the following upper bound:

$$|\mathcal{A}| \leq |\mathcal{V}_E|,$$

for all $i : \alpha_i \neq 0$. This property states that ESVs characterize the sparseness of problem (2) just as SVs characterize the sparseness of any SVR formulation. The property proved by the theorem points out that the proposed LP-SVR problem (2) possess better sparseness than that of standard SVRs, since there are always several SSVs in standard SVRs, especially for practical noisy datasets used in recognition or regression problems [18]. We can further have an upper bound with respect to the rank of kernel matrix as follows:

Theorem 2 (Based on Zhang, *et al.* [18]: Rank Bound)**.** *Given an optimal solution* $\mathbf{z}^*$ *to (2), the number of nonzero coefficients of (2) has the following upper bound:*

$$|\mathcal{A}| \leq \mathit{rank}(\mathbf{K}), \tag{14}$$

and the column vectors $k(\mathbf{x}_j, \mathbf{x}_1), k(\mathbf{x}_j, \mathbf{x}_2), \ldots, k(\mathbf{x}_j, \mathbf{x}_i)$*, are linearly independent for all* $j \in \mathcal{A}$*.*

Proof. For the LP-SVR problem (2), that is equivalent to (8), let us denote the column vector matrix associated with the variables $\alpha_j^{+*}, \alpha_j^{-*}, \ j = \{1, 2, \ldots, N\}$ by

$$\mathbf{B}_{\boldsymbol{\alpha}} = \begin{pmatrix} -\mathbf{K} & \mathbf{K} \\ \mathbf{K} & -\mathbf{K} \end{pmatrix}. \tag{15}$$

According to [18], the number $|\mathcal{A}|$ of nonzero variables in $\alpha_j^{+*}, \alpha_j^{-*}, \ j = \{1, 2, \ldots, N\}$ is at most equal to the number of columns in the corresponding basic column vector matrix $\mathbf{B}_{\boldsymbol{\alpha}}^*$ which are shared by $\mathbf{B}_{\boldsymbol{\alpha}}$ and the optimal basic matrix $\mathbf{B}^*$ corresponding to the optimal solution $\mathbf{z}^*$. Hence,

$$|\mathcal{A}| \leq \text{rank}(\mathbf{B}_{\boldsymbol{\alpha}}^*) \leq \text{rank}(\mathbf{B}_{\boldsymbol{\alpha}}) = \text{rank}(\mathbf{K}), \tag{16}$$

thus,

$$|\mathcal{A}| \leq \text{rank}(\mathbf{K}). \tag{17}$$

Since the optimal basic matrix $\mathbf{B}^*$ is linearly independent, so is $\mathbf{B}_{\boldsymbol{\alpha}}^*$. Now the column vectors $\{(\phi_j(\mathbf{x}_1).\phi_j(\mathbf{x}_2), \ldots, \phi_j(\mathbf{x}_i) \ : \ \alpha_j^* \neq 0, \ i, j = 1, 2, \ldots, N\}$ are linearly independent, since $\alpha_j^{+*} \geq 0$ (or $\alpha_j^{-*} \geq 0$) associated with $\alpha_j^* \neq 0$ must be the basic variables. □

In summary, we can say that the number of nonzero coefficients of (2) has the following upper bound:

$$|\mathcal{A}| \leq \text{rank}(\mathbf{K}),$$

and the column vectors $k(\mathbf{x}_j, \mathbf{x}_1), k(\mathbf{x}_j, \mathbf{x}_2), \ldots, k(\mathbf{x}_j, \mathbf{x}_i)$, are linearly independent for all $j \in \mathcal{A}$. This means that the LP-SVR regression function (1) can be exactly reproduced using only those samples that are SVs, without affecting performance. This property also indicates that vectors $k(\mathbf{x}_j, \mathbf{x}_1), k(\mathbf{x}_j, \mathbf{x}_2)$, ..., $k(\mathbf{x}_j, \mathbf{x}_i)$, for all $j \in \mathcal{A}$, in the decision function $f(\mathbf{x}) = \sum_{i\in\mathcal{A}}(\alpha_i^+ - \alpha_i^-)k(\mathbf{x}_i, \mathbf{x}) + (b^+ - b^-)$, are linearly independent. This means one cannot further reduce the number of basis functions in the regression function, and also suggests that the proposed LP-SVR (2) will lead to the sparsest model representation.

Let us remark that problem (2) was designed to maximize computational efficiency without sacrificing accuracy, which is achieved by not introducing unnecessary parameters into the problem and by posing a problem that minimizes the number of SVs without affecting performance. Nonetheless, the training phase (*i.e.*, learning process) still may be computationally expensive for applications with N larger than a few thousands.

In the following section, the learning process for the case when N is very large is explained.

IV. Proof of Convergence and Optimality Conditions

Algorithm 1 shows an algorithm that combines both variable and constraint decomposition to solve a large-scale LP-SVR training problem.

As Algorithm 1 shows, the process of solving problem (2) involves the iterative call to the variable decomposition and constraints decomposition algorithms described in [19]–[21], until the KKT conditions (6a)-(6d) or a stopping criteria are satisfied. Clearly, if the KKT conditions are satisfied, the solution is guaranteed to be the optimal solution. Therefore, as long as the variable and constraint decomposition parts of Algorithm 1 *terminate* at an optimal solution, the complete process also *converges to an optimal solution*.

To explain convergence and optimality of the decomposition algorithms, first, we discuss how the constraints decomposition process, *i.e.* line 6 through 17 of Algorithm 1, terminates in a finite number of iterations; second, it follows to discuss how the variables decomposition process, *i.e.* line 3 through 28 of Algorithm 1, terminates in a finite number of iterations; and third, we discuss global convergence.

A. Finite Number of Iterations for Constraints Decomposition Algorithm

Theorem 3 (Based on Bradley, *et al.* [22]: Finite Termination of Algorithm 1)**.** *The sequence $\mathbf{z}^{(t)}$ generated in line 13 of Algorithm 1 has the following properties:*

1) *The sequence $\mathbf{c}^T\mathbf{z}^{(t)}$ of the objective function values is non-increasing and is bounded by the global minimum of $\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z}=\mathbf{b},\ \mathbf{z}\geq\mathbf{0}\}$.*
2) *The sequence $\mathbf{c}^T\mathbf{z}^{(t)}$ of the objective function values becomes constant: $\mathbf{c}^T\mathbf{z}^{(t)}=\mathbf{c}^T\mathbf{z}^{t+1}$ for all $t\geq j$ for some $j\geq 1$.*
3) *For $t\geq j$, the active constraints in line 13 of Algorithm 1 at $\mathbf{z}^{(t)}$ with positive multipliers remain active for iteration $t+1$.*
4) *For all $t\geq t_{max}$, for some $t_{max}\geq j$, $\mathbf{z}^{(t)}$ is a solution of the linear program $\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z}=\mathbf{b},\ \mathbf{z}\geq\mathbf{0}\}$, provided all active constraints at $\mathbf{z}^{(t)}$ have positive multipliers remain for $t\geq j$.*

This theorem is extremely important, because it allows us to establish that the algorithm terminates in finite time. Here, the theorem assumes that the linear program $\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z}=\mathbf{b},\ \mathbf{z}\geq\mathbf{0}\}$ has a solution and, hence, that all the sub-problems in line 13 of Algorithm 1 also have solutions. Second, assume they satisfy the KKT conditions. Then, under these assumptions, we can prove that the theorem holds true. However, we must first establish and prove the following lemma, which consequently will allow us to prove the theorem.

Lemma 1 (Based on Bradley, *et al.* [22]: Decomposition KKT Conditions)**.** *If $\mathbf{z}$ solves the linear program $\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z}=\mathbf{b},\ \mathbf{z}\geq\mathbf{0}\}$, and $(\mathbf{z},\boldsymbol{\lambda})\in\mathbb{R}^{n+m}$ is a primal-dual optimal pair, such that $\boldsymbol{\lambda}_{\mathcal{I}}>0$, where $\mathcal{I}\subset\{1,2,\ldots,m\}$ and $\boldsymbol{\lambda}_{\mathcal{J}}=0$, where $\mathcal{J}\subset\{1,2,\ldots,m\}$, $\mathcal{I}\cup\mathcal{J}=\{1,2,\ldots,m\}$, then*

$$\mathbf{z}\in\arg\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}\quad s.t\ \{\mathbf{A}_{\mathcal{I}}\mathbf{z}=\mathbf{b}_{\mathcal{I}},\ \ \mathbf{z}\geq\mathbf{0}\},\tag{18}$$

where $\mathbf{A}_{\mathcal{I}}$ has rows of $\mathbf{A}_i$, for all $i\in\mathcal{I}$, and $\mathbf{b}_{\mathcal{I}}$ has elements b_i, for all $i\in\mathcal{I}$.

Proof. The KKT conditions for a primal-dual optimal pair $(\mathbf{z},\boldsymbol{\lambda})$ are:

$$\begin{aligned}\mathbf{c}&=\mathbf{A}^T\boldsymbol{\lambda},\\ \boldsymbol{\lambda}^T(\mathbf{A}\mathbf{z}-\mathbf{b})&=0\\ \mathbf{A}\mathbf{z}-\mathbf{b}&\geq 0,\\ \mathbf{z},\boldsymbol{\lambda}&\geq\mathbf{0},\end{aligned}$$

which under the condition $\boldsymbol{\lambda}_{\mathcal{I}}>0$ imply that

$$\begin{aligned}\mathbf{A}_{\mathcal{I}}\mathbf{z}&=\mathbf{b}_{\mathcal{I}},\\ \boldsymbol{\lambda}_{\mathcal{J}}&=0,\\ \mathbf{A}_{\mathcal{J}}\mathbf{z}&\geq\mathbf{b}_{\mathcal{J}}.\end{aligned}$$

It can be claimed [22] that $\mathbf{z}$ is also a solution for (18) because the primal-dual optimal pair $(\mathbf{z},\boldsymbol{\lambda})$ satisfies the KKT conditions:

$$\begin{aligned}\mathbf{c}&=\mathbf{A}_{\mathcal{I}}^T\boldsymbol{\lambda}_{\mathcal{I}},\\ \boldsymbol{\lambda}_{\mathcal{I}}&\geq\mathbf{0},\\ \mathbf{A}_{\mathcal{I}}\mathbf{z}&=\mathbf{b}_{\mathcal{I}},\end{aligned}$$

which are necessary and sufficient. □

*Proof of **Theorem 3**.* The proof is as follows:

1) By Lemma 1 $\mathbf{c}^T\mathbf{z}^{(t)}$ is a lower bound for $\mathbf{c}^T\mathbf{z}^{(t+1)}$. Therefore, of the objective function values is non-increasing. Since the constraints in line 13 of Algorithm 1 form a subset of the constraints of $\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z}=\mathbf{b},\ \mathbf{z}\geq\mathbf{0}\}$, then, it follows that $\mathbf{c}^T\mathbf{z}^{(t+1)}\leq\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z}=\mathbf{b},\ \mathbf{z}\geq\mathbf{0}\}$.
2) Since there is a finite number of vertices to the linear program $\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z}=\mathbf{b},\ \mathbf{z}\geq\mathbf{0}\}$, as well as of the subproblems in line 13 of Algorithm 1, it follows that from a certain $\bar{t}$ onward, a finite subset of such vertices will repeat infinitely often. Since a repeated vertex gives the same value for $\mathbf{c}^T\mathbf{z}$, it follows, by the nondecreasing property of $\mathbf{c}^T\mathbf{z}^{(t)}$ established above, that all vertices between repeated vertices also have the same objective value $\mathbf{c}^T\mathbf{z}$ and thus: $\mathbf{c}^T\mathbf{z}^{(t)}=\mathbf{c}^T\mathbf{z}^{(t+1)}\leq\min_{\mathbf{z}}\{\mathbf{c}^T\mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z}=\mathbf{b},\ \mathbf{z}\geq\mathbf{0}\}$, $\forall t\geq\bar{t}$.
3) Let $\bar{t}$ be as defined in the theorem. Let the index $j\in\{1,2,\ldots,m\}$ be that of some active constraint at iteration $\bar{t}$ with positive multiplier ($\mathbf{A}_j\mathbf{z}^{(\bar{t})}=\mathbf{b}_j,\boldsymbol{\lambda}_j^{(\bar{t})}>\mathbf{0}$), which has become inactive in the next step, that is:

Algorithm 1 Variables and Constraints Decomposition Strategy for a LS LP-SVR

Require: a) B_0, initial working set size. b) $\mathcal{T} = \{\mathbf{x}_i, d_i\}_{i=1}^{N}$, a training set with N samples. c) τ, number of blocks for block decomposition. d) l, maximum in-out.

$\mathcal{B} \leftarrow$ randomly selected B_0 indices as the initial wk. set.
$\mathcal{M} \leftarrow$ indices not in $\mathcal{B}$, that denotes the initial fixed set.
Begin ▷ **Variables Decomposition**.
  $\mathcal{W} \leftarrow \{\mathbf{x}_i, d_i\}_{i \in \mathcal{B}}$
  Fix $\alpha_j = 0$ for all $j \in \mathcal{M}$. ▷ Variables in (2) ignored.
  Begin ▷ **Constraints Decomposition**.
    Define $\mathbf{A}, \mathbf{b}, \mathbf{c}$ with (4a)-(4d).
    $\mathbf{A}_{\mathcal{B}}, \mathbf{b}_{\mathcal{B}}, \mathbf{c}_{\mathcal{B}} \leftarrow \mathbf{A}, \mathbf{b}, \mathbf{c}$
    $\begin{pmatrix} \mathbf{A}_{\mathcal{R}}^1 & \mathbf{b}_{\mathcal{R}}^1 \\ \mathbf{A}_{\mathcal{R}}^2 & \mathbf{b}_{\mathcal{R}}^2 \\ \vdots & \vdots \\ \mathbf{A}_{\mathcal{R}}^\tau & \mathbf{b}_{\mathcal{R}}^\tau \end{pmatrix} \leftarrow$ BLOCKPARTITION(τ,$\mathbf{A}_{\mathcal{B}}$,$\mathbf{b}_{\mathcal{B}}$)
    $t = 0$ ▷ Iterations counter.
    repeat
      $t = t + 1$
      $\mathbf{z}_{\mathcal{R}}^{(t)} \leftarrow$ IPMSOLVELP($\mathbf{c}_{\mathcal{R}}$,$\mathbf{A}_{\mathcal{R}}$,$\mathbf{b}_{\mathcal{R}}$) ▷ Solver.
      $z_{i,\mathcal{B}}^{(t)} = \begin{cases} z_{j,\mathcal{R}}^{(t)} & \text{if } j = i, \quad \text{for all } j \in \mathcal{R} \\ 0 & \text{otherwise,} \end{cases}$
    until $\left(\mathbf{c}_{\mathcal{B}}^T \mathbf{z}_{\mathcal{B}}^{(t)} = \mathbf{c}_{\mathcal{B}}^T \mathbf{z}_{\mathcal{B}}^{(t+4)}\right)$ ▷ Stops after 4 iterations.
    $\mathbf{z}_{\mathcal{B}} \leftarrow \mathbf{z}_{\mathcal{B}}^{(t)}$ ▷ Problem is solved for $\mathcal{W}$.
  End ▷ End **Constraint Decomposition**.
  for all $j \in \mathcal{M}$ **do** ▷ Verify problem solved for $\mathcal{M}$.
    Reconstruct u_j, ξ_j, verify primal LP.
    Fix $\lambda_j = 0$, reconstruct s_j, verify dual LP.
    $\widetilde{\mathcal{B}} \leftarrow$ VERIFYCOMPLEMENTARITY(z_j, s_j, B_0)
  end for
  if $z_j s_j \neq 0$ **then** ▷ If problem (2) is not solved then, choose new working set.
    $\mathcal{B} \leftarrow$ CREATENEWWORKINGSET($\mathcal{B}$,B_0,$\widetilde{\mathcal{B}}$)†
  else
    Stop Training
  end if
End ▷ End **Variable Decomposition**.
$\begin{pmatrix} \boldsymbol{\alpha}^+ & \boldsymbol{\alpha}^- & b^+ & b^- & \boldsymbol{\xi} & \mathbf{u} \end{pmatrix}^T \leftarrow \mathbf{z}$
return $(\boldsymbol{\alpha}^+, \boldsymbol{\alpha}^-, b^+, b^-)$

†**Note**: Indices having been at least l times in and out of the working set $\mathcal{B}$ are moved permanently into $\mathcal{B}$.

$\mathbf{A}_j \mathbf{z}^{(\bar{t}+1)} > \mathbf{b}_j$. We then obtain the following contradiction by the previous item and the KKT saddlepoint contradiction:

$$\begin{aligned} 0 &= \mathbf{c}^T \mathbf{z}^{(\bar{t}+1)} - \mathbf{c}^T \mathbf{z}^{(\bar{t})} \\ &\geq \left(\boldsymbol{\lambda}^{(\bar{t})}\right)^T \left(\bar{\mathbf{A}}^{(\bar{t})} \mathbf{z}^{(\bar{t}+1)} - \bar{\mathbf{b}}^{(\bar{t})}\right) \\ &\geq \boldsymbol{\lambda}_j^{(\bar{t})} \left(\mathbf{A}_j \mathbf{z}^{(\bar{t}+1)} - \mathbf{b}_j\right) \geq 0. \end{aligned} \tag{19}$$

4) By the second item above, a finite number of vertices repeat infinitely for $t \geq \bar{t}$ all with constant $\mathbf{c}^T \mathbf{z}^{(t)}$. Since active constraints with positive multipliers at iteration t remain active at iteration $t+1$ by the third item above and hence have a positive multiplier by assumption of the fourth item, the set of active constraints with positive multipliers will remain constant for $t \geq \bar{t}$, for some $\tilde{t} \geq \bar{t}$, because there is a finite number of constraints, and hence $\mathbf{z}^{(t)}$ will remain a fixed vertex $\bar{\mathbf{z}}$ for $t \geq \tilde{t}$. The point $\bar{\mathbf{z}}$ will satisfy all the constraints of the linear problem $\min_{\mathbf{z}} \{\mathbf{c}^T \mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z} = \mathbf{b}, \ \mathbf{z} \geq \mathbf{0}\}$ because all constraints are eventually imposed on the infinitely repeated vertex $\bar{\mathbf{z}}$. Hence, $\mathbf{c}^T \bar{\mathbf{z}}$ which lower-bounds the minimum of $\min_{\mathbf{z}} \{\mathbf{c}^T \mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z} = \mathbf{b}, \ \mathbf{z} \geq \mathbf{0}\}$ is also a minimum of $\min_{\mathbf{z}} \{\mathbf{c}^T \mathbf{z}\}$ subject to $\{\mathbf{A}\mathbf{z} = \mathbf{b}, \ \mathbf{z} \geq \mathbf{0}\}$ because $\bar{\mathbf{z}}$ is feasible. Hence the algorithm can be terminated at $t = \tilde{t}$.

□

This in summary proves that the constraint decomposition strategy has a finite number of iterations and will terminate in finite time.

B. Finite Number of Iterations for Variables Decomposition Algorithm

Let $\mathcal{T} = \{\mathbf{x}_i, d_i\}_{i=1}^{N}$ define a training with N samples. Then the number of variables and constraints in problem (2) is $4N + 2$ and $2N$ respectively. Now, following Torii, *et al.* algorithm in [23], we can define $\mathcal{B}^{(t)}$ as the working set at iteration t. Then, we let the following sequence

$$\begin{aligned}\mathcal{B}^{(t)} &= \mathcal{B}^{(t+k+1)},\\ \mathcal{B}^{(t+1)} &= \mathcal{B}^{(t+k+2)}, \ \dots \ ,\\ \mathcal{B}^{(t+k)} &= \mathcal{B}^{(t+2k+1)},\end{aligned}$$

denote an infinite loop, since the same working set it is being repeated every k iterations. Now, suppose we use line 24 of Algorithm 1 to permanently add to the working set those constraints entering and leaving the working set for a number of iterations. Then, if for any reason, *e.g.*, the properties of the dataset samples, an infinite loop exists, it would be prevented at line 24 of Algorithm 1, since the constraints entering and exiting the working set by at least l times are added permanently to the working set, which implies that the infinite loop is broken. That is, the infinite loop will not occur at current (t, $t + k$) or further iterations ($t + 1$, $t + k + 1$). Moreover, since the number of both constraints and variables is finite, the number of infinite loops is also finite. Therefore, infinite loops are handled in finite steps. Hence, Algorithm 1 terminates in a finite number of iterations.

V. Interior Points Convergence and Optimality

Part of the computational robustness of the proposed decomposition methods rely on the usage of interior point methods (IPM) for linear programming (see [24] for a comprehensive review on IPM). Here we want to discuss shortly how the linear programs are being solved in line 13 of Algorithm 1,.

First, let us consider the KKT conditions (6a)-(6d) established for our problem (2). Let us recall that the problem (3) is equivalent to (2) and that the KKT conditions (6a)-(6d) are also equivalent to (6a)-(6d). IPM considers the KKT conditions as the following function:

$$\mathbf{F}(\mathbf{z}, \boldsymbol{\lambda}, \mathbf{s}) = \begin{pmatrix} \mathbf{A}^T\boldsymbol{\lambda} + \mathbf{s} - \mathbf{c} \\ \mathbf{A}\mathbf{z} - \mathbf{b} \\ \mathbf{XS1} \end{pmatrix} = \mathbf{0}, \tag{20a}$$

$$\mathbf{z}, \mathbf{s} \geq \mathbf{0} \tag{20b}$$

where $\mathbf{X} = \text{diag}(z_1, z_2, ..., z_n)$, and $\mathbf{S} = \text{diag}(s_1, s_2, ..., s_n)$. The IPM generates a set of solutions $\mathcal{F}^{(t)} = (\mathbf{z}^{(t)}, \boldsymbol{\lambda}^{(t)}, \mathbf{s}^{(t)})$ at each iteration t. The key idea is to find solutions $(\mathbf{z}^{(t)}, \boldsymbol{\lambda}^{(t)}, \mathbf{s}^{(t)})$ that satisfy $\mathbf{F}(\mathbf{z}^{(t)}, \boldsymbol{\lambda}^{(t)}, \mathbf{s}^{(t)}) = 0$ and more importantly $\mathbf{z}^{(t)}, \mathbf{s}^{(t)}$ being strictly positive, except at the solution where $\mathbf{z}$ or $\mathbf{s}$ may be equal to zero.

IPM surrounds the current point in a linear model in order to obtain the step direction $(\Delta\mathbf{z}, \Delta\boldsymbol{\lambda}, \Delta\mathbf{s})$ as follows:

$$\mathbf{J}(\mathbf{z}, \boldsymbol{\lambda}, \mathbf{s}) \begin{pmatrix} \Delta\mathbf{z} \\ \Delta\boldsymbol{\lambda} \\ \Delta\mathbf{s} \end{pmatrix} = -\mathbf{F}(\mathbf{z}, \boldsymbol{\lambda}, \mathbf{s}), \tag{21}$$

where $\mathbf{J}(\mathbf{z}, \boldsymbol{\lambda}, \mathbf{s})$ is the Jacobian of $\mathbf{F}(\mathbf{z}, \boldsymbol{\lambda}, \mathbf{s})$. Then the step direction (using a predictor-corrector strategy) becomes

$$\begin{pmatrix} \mathbf{0} & \mathbf{A}^T & \mathbf{I} \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{S} & \mathbf{0} & \mathbf{X} \end{pmatrix} \begin{pmatrix} \Delta\mathbf{z} \\ \Delta\boldsymbol{\lambda} \\ \Delta\mathbf{s} \end{pmatrix} = \begin{pmatrix} -\mathbf{r}_c \\ -\mathbf{r}_b \\ -\mathbf{XS1} - \Delta\mathbf{X}^{\text{aff}}\Delta\mathbf{S}^{\text{aff}}\mathbf{1} + \sigma\mu\mathbf{1} \end{pmatrix}, \tag{22}$$

where $\mathbf{r}_c = \mathbf{A}^T\boldsymbol{\lambda} + \mathbf{s} - \mathbf{c}$ and $\mathbf{r}_b = \mathbf{A}\mathbf{z} - \mathbf{b}$ are residuals, $\Delta\mathbf{X}^{\text{aff}}, \Delta\mathbf{S}^{\text{aff}}$ are the affine-scaling direction, μ is the duality gap, and σ is an adaptive line-search parameter depending on μ. The new iterate is therefore

$$(\mathbf{z}, \boldsymbol{\lambda}, \mathbf{s}) + \alpha(\Delta\mathbf{z}, \Delta\boldsymbol{\lambda}, \Delta\mathbf{s}), \tag{23}$$

where $\alpha \in (0, 1]$ is appropriately chosen in order to maintain $(\mathbf{z}, \mathbf{s})$ strictly positive.

A. On The Convergence of IPM for LP

Let $\mathcal{F}^{(t)} = (\mathbf{z}^{(t)}, \boldsymbol{\lambda}^{(t)}, \mathbf{s}^{(t)})$ be the set of feasible solutions generated inside the function call IPMSolveLP($\cdot$) at line 13 of Algorithm 1 at iteration t. Under this definition, Zhang, Tapia, *et al.* [25], as well as [26]–[29], demonstrate that IPM for LP exhibits the following properties:

- $\mathcal{F}^{(t)}$ converges to $\mathcal{F}^*$,
- the duality gap converges to zero ${\mathbf{z}^{(t)}}^T\boldsymbol{\lambda}^{(t)} \to 0$ with q**-quadratic** behavior if all solutions are non-degenerate,
- the duality gap converges to zero ${\mathbf{z}^{(t)}}^T\boldsymbol{\lambda}^{(t)} \to 0$ with q**-superlinear** behavior if there is any degenerate solution.

The above properties demonstrate that LP-IPM is *q-quadratically* convergent to a feasible solution, *i.e.*, it is equivalent to the Newton method. Even in case of degeneracy, IPM is *q-superlinearly* convergent. In contrast, the simplex method is of exponential complexity. In spite of this, the simplex is typically used in most decomposition strategies for large-scale SVM.

Figure 1 shows the behavior of Primal, Dual, and Complementarity Condition using IPM for an arbitrarily three-class non-separable classification problem. The solution can easily be found in very few iterations.

VI. Conclusion

This paper has presented proofs of convergence and sparsity of a particular kind of Support Vector Machine for regression known as Linear Programming Support Vector Machine for Regression (LP-SVR).

We began by posing a Support Vector Regression (SVR) problem as a linear programming problem, defining its primal

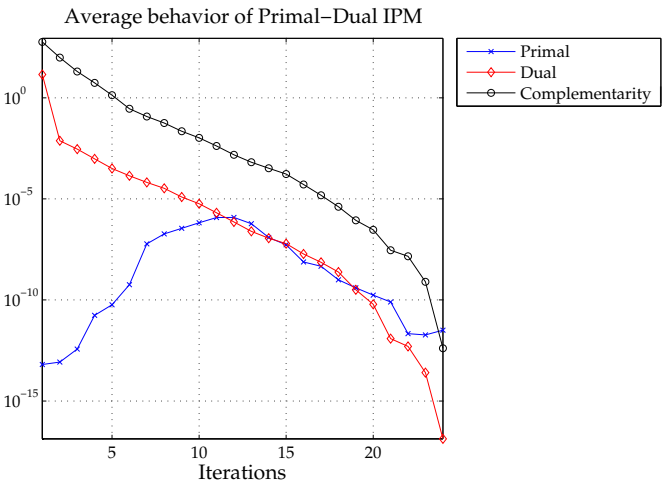

Fig. 1. Behavior of the KKT conditions as the number of iterations progress. The primal, dual, and complementarity condition must converge to zero. The results shown represent the average value over several experiments on arbitrarily three-class non-separable classification problems.

and dual. Then we used these definitions to build to the proof of optimality using traditional KKT conditions analysis.

Next, we described a sequential optimization method based on variables decomposition, constraints decomposition, and primal-dual interior point methods for solving large-scale regression/classification problems.

Finally, based on the entire methodology, we present proof of convergence and optimality conditions of the sequential optimization as well as its ability to produce sparse solutions.

Further work includes extending these proofs to other SVM formulations that fit the general structure of linear programming problems.

Acknowledgment

The author would like to thank the support of the Department of Computer Science at Baylor University during this research study.

References

[1] S. Kumar, A. Kar, and M. Chandra, "Svm based adaptive median filter design for face detection in noisy images," in *2014 International Conference on Signal Processing and Integrated Networks (SPIN)*, 2014, pp. 695–698.

[2] S. Liu, C. Jia, and P. Chen, "Hypertext classification using weighted transductive support vector machines," in *2006 International Conference on Machine Learning and Cybernetics*, 2006, pp. 3535–3540.

[3] S. Manthira Moorthi, I. Misra, R. Kaur, N. P. Darji, and R. Ramakrishnan, "Kernel based learning approach for satellite image classification using support vector machine," in *2011 IEEE Recent Advances in Intelligent Computational Systems*, 2011, pp. 107–110.

[4] B. Satpute and R. Yadav, "Support vector machines for protein family identification using surface invariant coordinates," in *2018 3rd International Conference for Convergence in Technology (I2CT)*, 2018, pp. 1–4.

[5] D. Nasien, H. Haron, and S. S. Yuhaniz, "Support vector machine (svm) for english handwritten character recognition," in *2010 Second International Conference on Computer Engineering and Applications*, vol. 1, 2010, pp. 249–252.

[6] M. Errami and M. Rziza, "Improving pedestrian detection using support vector regression," in *2016 13th International Conference on Computer Graphics, Imaging and Visualization (CGiV)*, 2016, pp. 156–160.

[7] X. Hou, S. Zhu, L. Xia, and G. Wu, "Stock price prediction based on grey relational analysis and support vector regression," in *2018 Chinese Control And Decision Conference (CCDC)*, 2018, pp. 2509–2513.

[8] M. K. Azad, S. Uddin, and M. Takruri, "Support vector regression based electricity peak load forecasting," in *2018 11th International Symposium on Mechatronics and its Applications (ISMA)*, 2018, pp. 1–5.

[9] Q. Wu, H. Zhang, R. Jing, and Y. Li, "Feature selection based on twin support vector regression," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 2903–2907.

[10] C. Li, X. Wang, Z. Cheng, and Y. Bai, "Forecasting bus passenger flows by using a clustering-based support vector regression approach," *IEEE Access*, vol. 8, pp. 19 717–19 725, 2020.

[11] J. Mercer, "Functions of positive and negative type, and their connection with the theory of integral equations," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 209, pp. 415–446, 1909.

[12] R. Courant and D. Hilbert, *Methods of Mathematical Physics: Partial Differential Equations*. John Wiley & Sons, 2008.

[13] G. B. Dantzig and M. N. Thapa, *Linear programming 1: introduction*. Springer Science & Business Media, 2006.

[14] G. Dantzig, *Linear programming and extensions*. Princeton Univ Pr, 1998.

[15] J. E. Dennis Jr and R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*. SIAM, 1996.

[16] M. C. Ferris, O. L. Mangasarian, and S. J. Wright, *Linear programming with MATLAB*. SIAM, 2007.

[17] J. Von Neumann, "A model of general economic equilibrium," in *Readings in the Theory of Growth*. Springer, 1971, pp. 1–9.

[18] L. Zhang and W. Zhou, "On the sparseness of 1-norm support vector machines," *Neural Networks*, vol. 23, no. 3, pp. 373–385, 2010.

[19] P. Rivas-Perea and J. Cota-Ruiz, "An algorithm for training a large scale support vector machine for regression based on linear programming and decomposition methods," *Pattern Recognition Letters*, vol. 34, no. 4, pp. 439–451, 2013.

[20] P. Rivas-Perea, J. Cota-Ruiz, and J.-G. Rosiles, "A nonlinear least squares quasi-newton strategy for lp-svr hyper-parameters selection," *International Journal of Machine Learning and Cybernetics*, vol. 5, no. 4, pp. 579–597, 2014.

[21] P. Rivas-Perea, J. Cota-Ruiz, J. A. Perez V., D. Garcia C., and J. G. Rosiles, "Lp-svr model selection using an inexact globalized quasi-newton strategy," *Journal of Intelligent Learning Systems and Applications*, vol. 5, no. 1, pp. 19–28, February 2013.

[22] P. Bradley and O. Mangasarian, "Massive data discrimination via linear support vector machines," *Optimization methods and software*, vol. 13, no. 1, pp. 1–10, 2000.

[23] Y. Torii and S. Abe, "Decomposition techniques for training linear programming support vector machines," *Neurocomputing*, vol. 72, no. 4-6, pp. 973–984, 2009.

[24] S. Wright, *Primal-dual interior-point methods*. Society for Industrial Mathematics, 1997.

[25] Y. Zhang, R. Tapia, and F. Potra, "On the superlinear convergence of interior-point algorithms for a general class of problems," *SIAM Journal on Optimization*, vol. 3, no. 2, pp. 413–422, 1993.

[26] Y. Zhang and R. A. Tapia, "Superlinear and quadratic convergence of primal-dual interior-point methods for linear programming revisited," *Journal of Optimization Theory and Applications*, vol. 73, no. 2, pp. 229–242, 1992.

[27] R. A. Tapia, Y. Zhang, and Y. Ye, "On the convergence of the iteration sequence in primal-dual interior-point methods," *Mathematical programming*, vol. 68, no. 1-3, pp. 141–154, 1995.

[28] J. Ji, F. Potra, R. Tapia, and Y. Zhang, "An interior-point method with polynomial complexity and superlinear convergence for linear complementarity problems," Rice Univ Houston TX Dept of Computational and Applied Mathematics, Tech. Rep., 1991.

[29] M. Argáez and L. Velázquez, "A new infeasible interior-point algorithm for linear programming," in *Proceedings of the 2003 Conference on Diversity in Computing*, 2003, pp. 12–14.