Pattern Recognition Letters 31 (2010) 1919–1923

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

On the regularization path of the support vector domain description

Michael Sass Hansen [*], Karl Sjöstrand, Rasmus Larsen

Informatics and Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark

ARTICLE INFO

Article history:
Received 3 August 2009
Available online 4 July 2010
Communicated by Y. Ma

Keywords:
Support vector domain description (SVDD)
Regularization path
One-class classifier
Novelty detection

ABSTRACT

The internet and a growing number of increasingly sophisticated measuring devices make vast amounts of data available in many applications. However, the dimensionality is often high, and the time available for manual labelling scarce. Methods for unsupervised novelty detection are a great step towards meeting these challenges, and the support vector domain description has already shown its worth in this field. The method has recently received more attention, since it has been shown that the regularization path is piece-wise linear, and can be calculated efficiently. The presented work restates the new findings in a manner which permits the calculation with $O(n \cdot n_{\mathcal{B}})$ complexity in each iteration step instead of $O(n^2 + n_{\mathcal{B}}^3)$, where n is the number of data points and $n_{\mathcal{B}}$ is the number of boundary points. This is achieved by updating and downdating the system matrix to avoid redundant calculations. We believe this will further promote the use of this method.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

We are often faced with data of high-dimensionality. Imaging devices with an intrinsic high number of variables are emerging for more and more applications, and in order to deal with this class of data, a whole series of data analysis tools have emerged. Many of these use the kernel trick to create efficient algorithms dealing seamlessly with the high number of dimensions through inner products, while keeping flexibility for modelling distributions (Vapnik, 1995). The support vector domain description (SVDD), introduced by Tax and Duin (1999), is a method for one-class labelling, which also falls into the aforementioned category. SVDD may be used for novelty detection, clustering or outlier detection (Zhang et al., 2006; Ben-Hur et al., 2001; Guo et al., 2009). The data is classified as either inliers or outliers through the introduction of a minimal containing sphere. The description has strong ties to the one-class version of the two-class method support vector machines (SVM) (Schölkopf et al., 2001).

The basic goal of SVDD is to find a minimal sphere containing inliers while minimizing the distance from the boundary to the outliers. More formally it can be stated as the following optimization problem

$$\min_{R^2,\boldsymbol{a},\xi_i} \sum_i \xi_i + \lambda R^2 \quad \text{where } (\boldsymbol{x}_i - \boldsymbol{a})(\boldsymbol{x}_i - \boldsymbol{a})^T \leqslant R^2 + \xi_i, \quad \xi_i \geqslant 0 \quad \forall i, \qquad (1)$$

where $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]$ is the data matrix with each point $\boldsymbol{x}_i \in R^p$, $\boldsymbol{a}$ is the center and R is the radius of the sphere, and ξ_i are the slack variables, allowing some points, the *outliers*, to lie outside the sphere, while still satisfying the constraints, i.e. (1) states that the squared distance from the center of the hypersphere should be no more than the squared radius plus slack. The regularization is governed by the parameter λ. This formulation is equivalent to the one by Tax and Duin (1999), but we use $\lambda = 1/C$ for regularization for simplicity of presentation. A large value of λ puts a high penalty on the radius and results in a small sphere, whereas a small λ lets the radius grow to include more points as *inliers*.

Originally, the optimization problem as posed in Section 2, is transformed into the dual problem using the Lagrange multipliers with the Karush–Kuhn–Tucker conditions, and is solved as a quadratic optimization problem. Recently, it was shown by Sjöstrand et al. (2007) that the regularization path of the parameter λ is piece-wise linear, and can be calculated with an $O(n_{\mathcal{B}}^3 + n^2)$ complexity for each iteration step, where $n_{\mathcal{B}} \ll n$ is the number of points on the boundary of the sphere and n is the total number of points. This result has been used to construct a generalized distance by Hansen et al. (2007). In Section 3, a more efficient approach reducing the complexity to $O(n \cdot n_{\mathcal{B}})$ in each iteration step is derived.

2. The support vector domain description

A Lagrangian operator can be used to solve the problem of finding the optimum sphere, posed in (1). The Lagrangian is given by

$$L_p : \sum_i \alpha_i((\boldsymbol{x}_i - \boldsymbol{a})(\boldsymbol{x}_i - \boldsymbol{a})^T - R^2 - \xi_i) + \sum_i \xi_i + \lambda R^2 - \sum_i \gamma_i \xi_i, \qquad (2)$$

where α_i and γ_i are the Lagrange multipliers. The Karush–Kuhn–Tucker complimentary conditions hold since the optimization problem is convex, and they are given by

[*] Corresponding author. Fax: +45 45882673.
E-mail address: msh@imm.dtu.dk (M.S. Hansen).

0167-8655/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2010.06.019

$$\alpha_i\left(\boldsymbol{x}_i\boldsymbol{x}_i^T - 2\boldsymbol{a}\boldsymbol{x}_i^T + \boldsymbol{a}\boldsymbol{a}^T - R^2 - \xi_i\right) = 0, \tag{3}$$

$$\gamma_i\xi_i = 0. \tag{4}$$

The optimum is given where the derivatives of the variables are zero

$$\frac{\delta L_p}{\delta R^2} = 0 \quad \Longleftrightarrow \quad \lambda = \sum_i \alpha_i, \tag{5}$$

$$\frac{\delta L_P}{\delta \boldsymbol{a}} = 0 \quad \Longleftrightarrow \quad \boldsymbol{a} = \frac{\sum_i \alpha_i \boldsymbol{x}_i}{\sum_i \alpha_i}, \tag{6}$$

$$\frac{\delta L_P}{\delta \xi_i} = 0 \quad \Longleftrightarrow \quad \lambda_i = 1 - \alpha_i. \tag{7}$$

From Eqs. (7), (3) and (4), it is seen that $\alpha_i = 1$ for outliers (since $\gamma_i = 0$) and $\alpha_i = 0$ for inliers. On the boundary, α_i can take any value in [0; 1]. Inserting Eqs. (5)–(7) in (2), the minimization problem is transformed to the problem of maximizing the Wolfe dual form

$$\max_{\boldsymbol{\alpha}} \sum_i \alpha_i \boldsymbol{x}_i \boldsymbol{x}_i^T - \frac{1}{\lambda}\sum_i\sum_j \alpha_i\alpha_j \boldsymbol{x}_i\boldsymbol{x}_j^T, \quad 0 \leqslant \alpha_i \leqslant 1, \ \sum_i \alpha_i = \lambda,$$

The dimensionality of the input vectors $\boldsymbol{x}_i$ can be increased using a basis expansion and the dot-product substituted by an inner product. The inner products can then be replaced by $K_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$, where K is a positive definite kernel function satisfying Mercer's theorem. The Gaussian kernel $K_{i,j} = K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp - \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{\gamma}$ is a popular example of such a kernel function. The optimization problem may then be stated as

$$W_d = \max_{\boldsymbol{\alpha}} \sum_i \alpha_i K_{i,i} - \frac{1}{\lambda}\sum_o\sum_j \alpha_i\alpha_j K_{i,j}, \tag{8}$$

$$0 \leqslant \alpha_i \leqslant 1, \quad \sum_i \alpha_i = \lambda. \tag{9}$$

For a given λ, the squared distance from the center of the sphere to a point $\boldsymbol{x}$ is

$$f(\boldsymbol{x};\lambda) = K(\boldsymbol{x},\boldsymbol{x}) - \frac{2}{\lambda}\sum_i \alpha_i K(\boldsymbol{x},\boldsymbol{x}_i) + \frac{1}{\lambda^2}\sum_i\sum_j \alpha_i\alpha_j K_{i,j}, \tag{10}$$

where the decision boundary is not necessarily a sphere in the space of the input points, although it is, in the space of the basis of the kernel function used. For the derivation the following sets are defined; the set $\mathcal{A}$ contains all the input points, $\mathcal{B}$ denotes the set of points on the boundary, $\mathcal{O}$ is the set of outliers, and let $\mathcal{I}$ be the set of inliers.

3. Calculating the regularization path of the SVDD

This derivation is the main contribution of the current work, and differs from the derivation by Sjöstrand et al. to provide the basis for a more efficient calculation of the parameters using updating and downdating of a matrix inverse. Two well known theorems, showing that the Lagrange multipliers are continuous for a convex problem, are stated in Appendix A. In Section 3.1 an expression for the piece-wise linear relation between $\boldsymbol{\alpha}$ and λ is derived along with a scheme for fast calculation. Finally the algorithm is outlined in Section 3.2.

3.1. Piece-wise linear regularization path

Let the generalized radius be denoted by R, then a boundary point $\boldsymbol{x}_h$, where $h \in \mathcal{B}$ must satisfy

$$f(\boldsymbol{x}_h;\lambda) = K_{h,h} - \frac{2}{\lambda}\sum_i \alpha_i K_{h,i} + \frac{1}{\lambda^2}\sum_i\sum_j \alpha_i\alpha_j K_{i,j} = R^2, \quad h \in \mathcal{B}. \tag{11}$$

The first sum can be split in terms depending on λ and constant terms (always 1 or 0 for points on the outside and inside). This gives $\sum_i \alpha_i K_{h,i} = \sum_{i\in\mathcal{B}} \alpha_i K_{h,i} + \sum_{i\in\mathcal{O}} \alpha_i K_{h,i}$. Only the first term depends on λ while the boundary set, $\mathcal{B}$, stays fixed, since α_i is always 1 on the outside. Let $k_i = K_{i,i}$ and define

$$R' = R^2 - \frac{1}{\lambda^2}\sum_i\sum_j \alpha_i\alpha_j K_{i,j},$$

and notice that R' takes the same value for all $h \in \mathcal{B}$. Let $\boldsymbol{K}_{\mathcal{B},\mathcal{B}}$ denote the matrix containing the inner products of the boundary points, $\boldsymbol{K}_{\mathcal{B},\mathcal{O}}$ denote the matrix with inner products of the boundary points and outliers, and let $\boldsymbol{k}_{\mathcal{B}}$ be a vector with elements $k_i, i \in \mathcal{B}$. Let $\boldsymbol{\alpha}_{\mathcal{B}}$ be a vector with the Lagrange multipliers α_i on the boundary, and let $\mathbf{1}_j$ be a column vector of length j, with all elements equal to 1. Let $n_{\mathcal{B}}$ denote the number of points in $\mathcal{B}$, then the set of Eq. (11) can be rewritten in matrix form as

$$\begin{bmatrix} \frac{2}{\lambda}\boldsymbol{K}_{\mathcal{B},\mathcal{B}} \mathbf{1}_{n_{\mathcal{B}}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_{\mathcal{B}} \\ R' \end{bmatrix} = \boldsymbol{k}_{\mathcal{B}} - \frac{2}{\lambda}\boldsymbol{K}_{\mathcal{B},\mathcal{O}} \mathbf{1}_{n_{\mathcal{O}}}. \tag{12}$$

This system of equations consists of $n_{\mathcal{B}}$ equations and $n_{\mathcal{B}} + 1$ unknown variables. The constraint from (5) is included in the linear system, and $\sum_i \alpha_i = \sum_{i\in\mathcal{B}} \alpha_i + n_{\mathcal{O}}$, where $n_{\mathcal{O}}$ is the number of outliers

$$\begin{aligned} \begin{bmatrix} \frac{2}{\lambda}\boldsymbol{K}_{\mathcal{B},\mathcal{B}} & \mathbf{1}_{n_{\mathcal{B}}} \\ \mathbf{1}^T_{n_{\mathcal{B}}} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_{\mathcal{B}} \\ R' \end{bmatrix} &= \begin{bmatrix} \boldsymbol{k}_{\mathcal{B}} - \frac{2}{\lambda}\boldsymbol{K}_{\mathcal{B},\mathcal{O}} \mathbf{1}_{n_{\mathcal{O}}} \\ \lambda - n_{\mathcal{O}} \end{bmatrix} \\ &= \begin{bmatrix} \frac{1}{\lambda}\boldsymbol{I}_{n_{\mathcal{B}}\times n_{\mathcal{B}}} & \mathbf{0}_{n_{\mathcal{B}}} \\ \mathbf{0}^T_{n_{\mathcal{B}}} & 1 \end{bmatrix} \left(\lambda \begin{bmatrix} \boldsymbol{k}_{\mathcal{B}} \\ 1 \end{bmatrix} + \begin{bmatrix} -2\boldsymbol{K}_{\mathcal{B},\mathcal{O}} \mathbf{1}_{n_{\mathcal{O}}} \\ -n_{\mathcal{O}} \end{bmatrix} \right). \end{aligned}$$

This may be rewritten

$$\begin{bmatrix} 2\boldsymbol{K}_{\mathcal{B},\mathcal{B}} & \mathbf{1}_{n_{\mathcal{B}}} \\ \mathbf{1}^T_{n_{\mathcal{B}}} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_{n_{\mathcal{B}}\times n_{\mathcal{B}}} & \mathbf{0}_{n_{\mathcal{B}}} \\ \mathbf{0}^T_{n_{\mathcal{B}}} & \lambda \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_{\mathcal{B}} \\ R' \end{bmatrix} = \lambda \begin{bmatrix} \boldsymbol{k}_{\mathcal{B}} \\ 1 \end{bmatrix} + \begin{bmatrix} -2\boldsymbol{K}_{\mathcal{B},\mathcal{O}} \mathbf{1}_{n_{\mathcal{O}}} \\ -n_{\mathcal{O}} \end{bmatrix}.$$

Define

$$\boldsymbol{K}' = \begin{bmatrix} 2\boldsymbol{K}_{\mathcal{B},\mathcal{B}} & \mathbf{1}_{n_{\mathcal{B}}} \\ \mathbf{1}^T_{n_{\mathcal{B}}} & 0 \end{bmatrix}.$$

Assuming the points are in general position in the expanded basis, such that the circle center is determined by at most the expanded plus one points, then $\boldsymbol{K}'$ can be inverted to obtain an expression for $\boldsymbol{\alpha}_{\mathcal{B}}$

$$\begin{bmatrix} \boldsymbol{I}_{n_{\mathcal{B}}\times n_{\mathcal{B}}} & \mathbf{0}_{n_{\mathcal{B}}} \\ \mathbf{0}^T_{n_{\mathcal{B}}} & \lambda \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_{\mathcal{B}} \\ R' \end{bmatrix} = \boldsymbol{K}'^{-1} \left(\lambda \begin{bmatrix} \boldsymbol{k}_{\mathcal{B}} \\ 1 \end{bmatrix} + \begin{bmatrix} -2\boldsymbol{K}_{\mathcal{B},\mathcal{O}} \mathbf{1}_{n_{\mathcal{O}}} \\ -n_{\mathcal{O}} \end{bmatrix} \right). \tag{13}$$

From this we learn that $\boldsymbol{\alpha}_{\mathcal{B}}$ is piece-wise linear in λ, while none of the constraints given in (9) are violated.

3.2. The algorithm

Since α_i, by Theorem 2, is continuous as a function of λ, this may be applied in finding the regularization path. Notice that if $\lambda = n$, it is easily seen that $\alpha_i = 1$, $i = 1, \ldots, n$. Therefore the algorithm is started in a state, where $\lambda = n$, and from this starting point λ can be decreased, and the two events that happen while decreasing λ are:

- A point from either the inside or the outside enters the boundary.
- A point exits the boundary to either the inside or the outside.

In between any of these events, the regularization path is piece-wise linear, as shown in Section 3.1, and the parameters can be calculated from (13).

In the following, let l be the last event that occurred and $l+1$ be the next event, so that λ^l was the previous and bigger value of the regularization parameter. Let $\boldsymbol{\alpha}_l$ be the value of all α_i at the event l and $\boldsymbol{\alpha}_{l+1}$ at the following event $l+1$. Then using that $\boldsymbol{\alpha}$ is continuous

$$\boldsymbol{\alpha}_{l+1} = \boldsymbol{\alpha}_l + (\lambda^{l+1} - \lambda^l)\boldsymbol{p}_l, \tag{14}$$

where only the points α_i on the boundary need to be updated. Let $\boldsymbol{x}_e \in \mathcal{A}$ be any point, and λ_e be the value of λ for which the event following event l would happen, if everything except λ was fixed. In Section 3.2.1 λ_e is found for all points outside the boundary, i.e. $\mathcal{I} \cup \mathcal{O}$, and in Section 3.2.2 λ_e is found for points on the boundary $\mathcal{B}$.

3.2.1. Boundary entry event

This event happens at a point where the distance to one of the non-boundary points equals the radius of the (generalized) sphere. This condition can be formulated as

$$f(\boldsymbol{x}_e;\lambda) - R^2 = K_{e,e} - \frac{2}{\lambda_e}\sum_i \alpha_i K_{e,i} + \frac{1}{\lambda_e^2}\sum_i\sum_j \alpha_i\alpha_j K_{i,j} - R^2 = 0.$$

Using that R^2 is given by Eq. (11) we find that

$$\begin{aligned} 0 &= K_{e,e} - \frac{2}{\lambda_e}\sum_i \alpha_i K_{e,i} - K_{h,h} + \frac{2}{\lambda_e}\sum_i \alpha_i K_{h,i} \\ &= K_{e,e} - K_{h,h} + \frac{2}{\lambda_e}(\boldsymbol{K}_{h,\mathcal{A}} - \boldsymbol{K}_{e,\mathcal{A}})(\boldsymbol{\alpha}_l + (\lambda_e - \lambda^l)\boldsymbol{p}_l) \\ \iff \quad \lambda_e - \lambda^l &= -\frac{(\boldsymbol{K}_{h,\mathcal{A}} - \boldsymbol{K}_{e,\mathcal{A}})\boldsymbol{\alpha}_l + \frac{\lambda^l}{2}(K_{e,e} - K_{h,h})}{(\boldsymbol{K}_{h,\mathcal{A}} - \boldsymbol{K}_{e,\mathcal{A}})\boldsymbol{p}_l + \frac{1}{2}(K_{e,e} - K_{h,h})}, \end{aligned} \tag{15}$$

where the sums have been replaced by matrix products, and $\boldsymbol{\alpha}$ has been substituted using (14). As we are decreasing the value of λ, we are only interested in values of $\lambda_e - \lambda^l$ smaller than 0. The biggest value, smaller than zero, of $\lambda_e - \lambda^l$ therefore marks the first entry event to occur. Since the complexity of calculating $\boldsymbol{K}_{\mathcal{A},\mathcal{A}}\boldsymbol{\alpha}_l$ is $O(n^2)$, this calculation should be done iteratively, updating $\boldsymbol{K}_{\mathcal{A},\mathcal{A}}\boldsymbol{\alpha}_l$ in each step, by noting $\boldsymbol{K}_{\mathcal{A},\mathcal{A}}\boldsymbol{\alpha}_{l+1} = \boldsymbol{K}_{\mathcal{A},\mathcal{A}}\boldsymbol{\alpha}_l + (\lambda^{l+1} - \lambda^l)\boldsymbol{K}_{\mathcal{A},\mathcal{B}}\boldsymbol{p}$, it can be calculated with complexity $O(n \cdot n_{\mathcal{B}})$.

3.2.2. Boundary exit event

Though Eq. (13) gives an explicit expression for α_i, this is only the case, when i denotes a point on the boundary. Otherwise α_i is limited by the constraints $0 \leqslant \alpha \leqslant 1$. As α_i, for i on the boundary, increases or decreases monotonically, only one of the two constraints comes into effect. Let the effective constraint be given by

$$C_{exit,e} = \begin{cases} 0, & \text{if } p_e \geqslant 0, \\ 1, & \text{if } p_e < 0, \end{cases}$$

then the boundary exit value for the eth point is given by

$$\lambda_e - \lambda^l = \frac{C_{exit,e} - \alpha_{l,e}}{p_e}.$$

3.2.3. Finding the next event $l+1$

Having calculated the first entry event and the first exit event, the only thing left is to choose which of the two events happens first and let

$$\lambda^{l+1} = \lambda^l + \max_{\boldsymbol{x}_e \in \mathcal{A}, \lambda_e - \lambda^l < 0}\{\lambda_e - \lambda^l\}.$$

An issue that has to be dealt with is how to propagate $\boldsymbol{\alpha}$ if the boundary set is the empty set. This is done simply by adding the closest outlier to the boundary set, which corresponds to making a discontinuous change in R^2, but not in $f(\boldsymbol{x})$ or $\boldsymbol{\alpha}$.

4. Complexity

The slope of $\boldsymbol{\alpha}$ with respect to λ, given by $\boldsymbol{p} = \boldsymbol{K}'^{-1}[\boldsymbol{k}_{\mathcal{B}}1]^T$ in (13) can be calculated using simple matrix multiplications of complexity $O(n_{\mathcal{B}}^2)$. $\boldsymbol{K}'^{-1}$ can be calculated using updating and downdating, also with complexity $O(n_{\mathcal{B}}^2)$, as is shown in Section 4.1. The complexity of calculating $\boldsymbol{p}_{\mathcal{B}}$ is $O(n_{\mathcal{B}}^2)$, while the complexity of evaluating the boundary entry conditions is $O(n \cdot n_{\mathcal{B}})$, which means that the overall complexity in each iteration step is of the order of $O(n \cdot n_{\mathcal{B}})$, as $n \geqslant n_{\mathcal{B}}$. The regularization path of the SVM could be found with the same complexity (Hastie et al., 2004), and the problems also show strong resemblance. Fig. 1 shows a graph of the calculation time of the previous algorithm and the presented implementation. Note that the computation time follows the theoretical complexity. For a population of 1000 points, the current implementation can be more than 100-times faster, and for our testing purposes this has been the limit for the length of the calculations we set up for the implementation presented in (Sjöstrand et al., 2007). The stability of the calculations has also set a natural limit, as discussed in Section 4.2. The method works on the covariance matrix (the kernel values) alone, and dimensions and distributions obviously have an impact on this matrix. However, the algorithm only requires the matrix to result from a Mercer kernel. The important data-dependent factors in the calculation workload are the number of iteration steps, which in the real world experiments we have made ranges between three and five times the number of points, n, and the number of boundary points $n_{\mathcal{B}}$, which usually is considerably smaller than the total number of points.

4.1. Calculation of $\boldsymbol{K}'^{-1}$

The two events that may occur to the boundary set either reduce or augment $\mathcal{B}$ by one point. This allows for an efficient calculation of $\boldsymbol{K}'^{-1}$, which is the purpose of the current section. Using the following result by Strassen (1969), the updates and downdates of the inverse can be calculated efficiently

$$\begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{A}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{S}_A\boldsymbol{C}\boldsymbol{A}^{-1} & -\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{S}_A \\ -\boldsymbol{S}_A\boldsymbol{C}\boldsymbol{A}^{-1} & \boldsymbol{S}_A \end{bmatrix}, \tag{16}$$

where the Schur complement of $\boldsymbol{A}$ is denoted $\boldsymbol{S}_A = (\boldsymbol{D} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}$. The efficient calculation of $\boldsymbol{K}'^{-1}_{l+1}$, the inverse of matrix $\boldsymbol{K}'$ after event l, will be presented in the following two paragraphs.

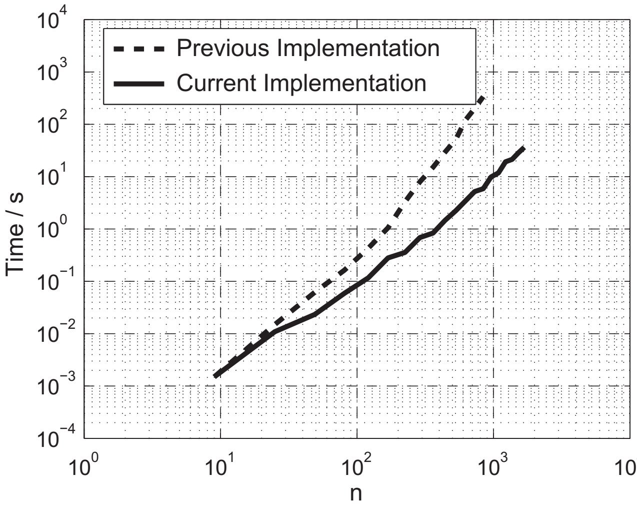


Fig. 1. Logarithmic plot of the complexities of the two different implementations.

Updating: Suppose that the point b^* has been added to $\mathcal{B}_l$ to form $\mathcal{B}_{l+1}$, then $\boldsymbol{K}'_{l+1}$ can be written as

$$\boldsymbol{K}'_{l+1} = \begin{bmatrix} \boldsymbol{K}'_l & \boldsymbol{K}_{\mathcal{B}_l,b^*} \\ \boldsymbol{K}_{b^*,\mathcal{B}_l} & K_{b^*,b^*} \end{bmatrix}. \tag{17}$$

Here $S_A = \left(K_{b^*,b^*} - \boldsymbol{K}_{b^*,\mathcal{B}_l}\boldsymbol{K}_l'^{-1}\boldsymbol{K}_{\mathcal{B}_l,b^*}\right)^{-1}$ and define $\boldsymbol{S}_C = \boldsymbol{K}_l'^{-1}\boldsymbol{K}_{\mathcal{B}_l,b^*}$, then the inverse can be calculated from

$$\begin{bmatrix} \boldsymbol{K}'_l & \boldsymbol{K}_{\mathcal{B}_l,b^*} \\ \boldsymbol{K}_{b^*,\mathcal{B}_l} & K_{b^*,b^*} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{K}_l'^{-1} + \boldsymbol{S}_C S_A \boldsymbol{S}_C^T & -\boldsymbol{S}_C S_A \\ -S_A \boldsymbol{S}_C^T & S_A \end{bmatrix}, \tag{18}$$

which only requires a multiplication of a vector with a matrix of size $n_{\mathcal{B}}$, and this multiplication has complexity $O(n_{\mathcal{B}}^2)$.

Downdating: Suppose that the point b^* has been removed from $\mathcal{B}_l$ to form $\mathcal{B}_{l+1}$. Then $\boldsymbol{K}_l'^{-1}$ can be written using Eq. (18), only here b^* is the point that was removed from the boundary

$$\boldsymbol{K}_l'^{-1} = \begin{bmatrix} \boldsymbol{A}_{n_{\mathcal{B}_{l+1}} \times n_{\mathcal{B}_{l+1}}} & \boldsymbol{B}_{n_{\mathcal{B}_{l+1}} \times 1} \\ \boldsymbol{C}_{1 \times n_{\mathcal{B}_{l+1}}} & D_{1 \times 1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{K}'_{l+1} & \boldsymbol{K}_{\mathcal{B}_{l+1},b^*} \\ \boldsymbol{K}_{b^*,\mathcal{B}_{l+1}} & K_{b^*,b^*} \end{bmatrix}^{-1}$$
$$\Rightarrow \boldsymbol{K}_{l+1}'^{-1} = \boldsymbol{A} - \boldsymbol{BC}D^{-1}. \tag{19}$$

4.2. Stability

As the currently derived fast path algorithm updates the parameters rather than recalculating them, as do all path algorithms, the result will drift due to numeric instability caused by the double precision used in the current implementation. This issue is investigated by running the implementation on different data sets, while testing the results for given values of the regularization parameter λ using an implementation of quadratic programming. The stability was tested on data sets of dimension 2 and 3 with three clusters of 1000 points. Each cluster is having a Gaussian independent and identical distribution with standard deviations sampled from a uniform distribution on [.4; 1.2] and with centers sampled uniformly from a cube of side length 8. In Fig. 3 the 2D data sets are seen with similarly constructed distributions, but fewer points, and the result of the stability test can be seen in Fig. 2. The result can be seen to differ by no more than 0.5% even for 25,000 updates and downdates of the inverse and the value of λ. The relationship appears to be linear, which can be attributed to the cumulative effect of rounding errors through the updates.

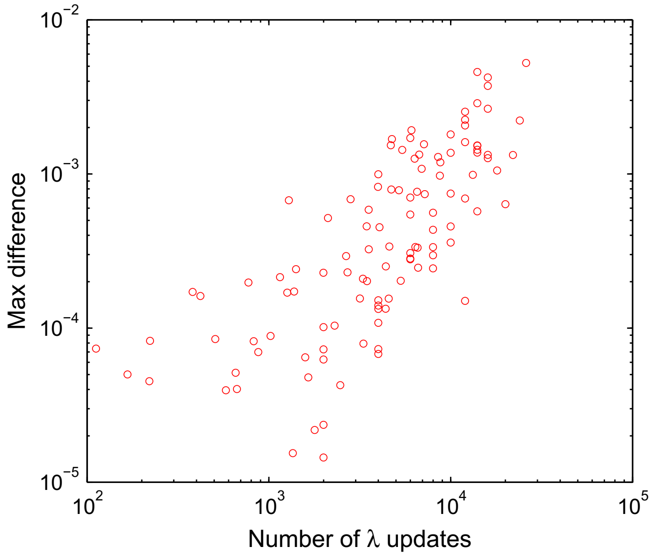
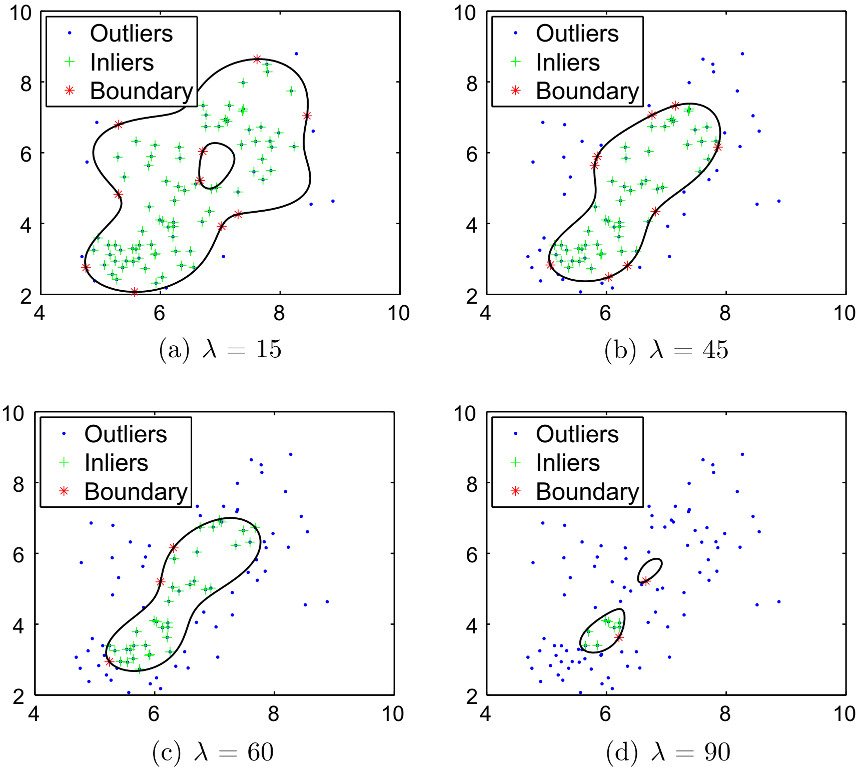


Fig. 2. Logarithmic plot of the error of the implementation compared to the result obtained using quadratic programming.



(a) $\lambda = 15$

(b) $\lambda = 45$

(c) $\lambda = 60$

(d) $\lambda = 90$

Fig. 3. Decision boundaries for different values of λ.

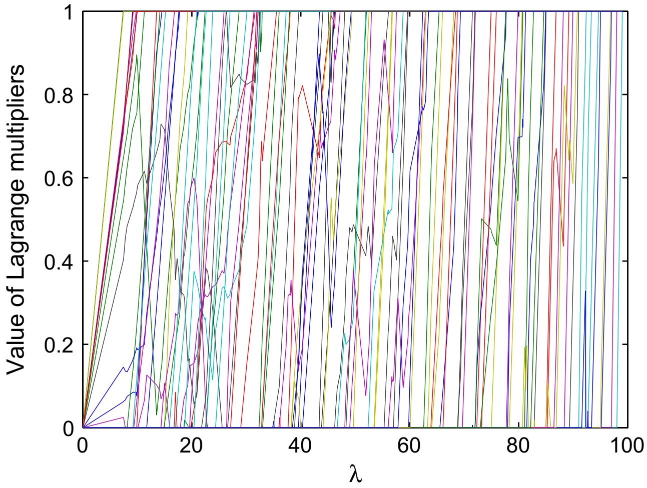


Fig. 4. The paths of the loading coefficients α_i. For $\lambda = 0$ the α_i, each represented by a curve, are all 0 (all inliers for an unconstrained sphere), and as λ is increased more and more α_i become 1, and for $\lambda = n = 100$ all α_i are 1. By some measure, the least typical point corresponds to the curve that first becomes 1 (and stays 1), and the most typical point corresponds to the last α_i to become 1.

For large numbers of samples, and thus updates, the method presented in (Sjöstrand et al., 2007) suffers from imprecision in calculating the next value λ^{l+1} because the value was estimated explicitly rather than estimating the difference $\lambda^{l+1} - \lambda^{l}$, as in the current work.

5. Demonstration

To demonstrate the method a small example is analyzed using the implemented algorithm. From two sources with 2-dimensional Gaussian distributions 100 points are sampled and they are analyzed with a Gaussian kernel function with a width of 1. The result can be seen in Fig. 3. Note that this value of the kernel parameter leaves room for a rather flexible decision boundary. In the Figure it can be seen that some of the points, the support vectors, are outside and some are inside, corresponding to a α_i of 0. In Fig. 4 the entire regularization path of $\boldsymbol{\alpha}$, that is the α_i corresponding to each point, can be observed.

The calculation is performed in a fraction of a second for this rather small sample size.

6. Conclusion

The support vector domain description (SVDD) is a new and popular method. Recent work by Sjöstrand et al. (2007) demonstrated that the regularization path of the weight coefficients depends piece-wise linearly on λ. This allows for an efficient calculation of the regularization path. The current work restates new findings in a manner that permits the calculation with a complexity of $O(n \cdot n_{\mathcal{B}})$ instead of $O(n^2 + n_{\mathcal{B}}^3)$ in each iteration step. It has been demonstrated that for $n = 800$ points, the calculation of the regularization path could be performed up to 100-times faster. The algorithm keeps the numeric error small for sample sizes up to 3000 points, smaller than 0.5% in the analyzed cases. We believe that this contribution will allow for even more applications of the method, either for choosing robust estimates of the distance, or possibly in the area of support vector clustering.

Appendix A. Continuity of the Lagrange multipliers

Theorem 1. *The Wolfe dual form W_d given by (8) is continuous with respect to the regularization parameter, λ.*

Proof. Let α_1 be a solution for a given set of points and regularization parameter λ_1, and $\boldsymbol{\alpha}_2$ a solution for regularization parameter λ_2. It is seen that for any $0 \leqslant s \leqslant 1$, $\boldsymbol{\alpha} = s\boldsymbol{\alpha}_1 + (1 - s)\boldsymbol{\alpha}_2$, satisfies the conditions on $\boldsymbol{\alpha}$, and due to the polynomial form of W_d it can be concluded that W_d is continuous. □

Theorem 2. *The Lagrange multipliers α are continuous with respect to λ.*

Proof. Follows directly from the fact that W_d is continuous and the solution to a convex problem is unique. □

References

Ben-Hur, A., Horn, D., Siegelmann, H.T., Vapnik, V., 2001. Support vector clustering. J. Machine Learn. 2, 125–137.

Guo, S.M., Chen, L.C., Tsai, J.S.H., 2009. A boundary method for outlier detection based on support vector domain description. Pattern Recognition 42 (1), 77–83.

Hansen, M.S., Sjöstrand, K., Olafsdóttir, H., Larsson, H.B.W., Stegmann, M.B., Larsen, R., 2007. Robust pseudo-hierarchical support vector clustering. In: Proc. Scandinavian Conf. on Image Analysis.

Hastie, T., Rosset, S., Tibshirani, R., Zhu, J., 2004. The entire regularization path for the support vector machine. J. Machine Learn. Res. 5, 1391–1415.

Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., Williamson, R., 2001. Estimating the support of a high-dimensional distribution. Neural Comput. 13, 1443–1471.

Sjöstrand, K., Hansen, M.S., Larsson, H.B., Larsen, R., 2007. A path algorithm for the support vector domain description and its application to medical imaging. Med. Image Anal.

Strassen, V., 1969. Gaussian elimination is not optimal. Numer. Math. 13, 354–356.

Tax, D.M., Duin, R.P., 1999. Support vector domain description. Pattern Recognition Lett. 20 (11–13), 1191–1199.

Vapnik, V., 1995. The Nature of Statistical Learning Theory. Springer, New York.

Zhang, J., Yan, Q., Zhang, Y., Huang, Z., 2006. Novel fault class detection based on novelty detection methods. Intell. Comput. Signal Proc. Pattern Recognition 345, 982–987.