# Lecture 6

1. Primal/dual optimization problem (KKT)
2. SVM dual
3. Kernels
4. Soft margin
5. SMO algorithm

## 1 Primal/dual optimization problem

To introduce the Support Vector Machine algorithm for the problems with nonlinearly separable classes we should recall the notion of primal and dual optimization problems. The method of Lagrange multipliers in the Multidimensional Calculus helps to solve the problem of optimization with additional constraints:

$$\min_w f(w)$$

subject to $h_i(w) = 0, i = 1 \ldots l$, or $h(w) = \begin{bmatrix} h_1(w) \\ h_2(w) \\ \vdots \\ h_l(w) \end{bmatrix} = \vec{0}$.

The Lagrangian is defined as

$$L(w, \beta) = f(w) + \sum_i \beta_i h_i(w),$$

where $\beta$ are Lagrange multipliers. The solution of the original optimization problem can be found by solving the system of equations

$$\frac{\partial L}{\partial w} = 0, \ \frac{\partial L}{\partial \beta_i} = 0$$

with respect to $w$ and $\beta$.

The primal problem by tradition is formulated in more general form:

$$\min_w f(w)$$

subject to

$$g_i \leqslant 0,\ i = 1, \dots, k,$$
$$h_i(w) = 0,\ i = 1, \dots, l,$$

or with vector notations:

$$g(w) \leqslant \vec{0},$$
$$h(w) = \vec{0}$$

For this problem the Lagrangian is defined by

$$L(w, \alpha, \beta) = f(w) + \sum_{i=1}^{k} \alpha_i g_i(w) + \sum_{i=1}^{l} \beta_i h_i(w)$$

and by definition

$$\theta_P(w) = \max_{\alpha, \beta, \alpha_i \geqslant 0} L(w, \alpha, \beta) = \begin{cases} f(w), & \text{if conditions for } g, h \text{ satisfies} \\ \infty, & \text{otherwise.} \end{cases}$$

Notice that if $g_i(w) > 0$, then $\theta_P(w) = \infty$; if $h_i(w) \neq 0$, then $\theta_P(w) = \infty$; otherwise, $\theta_P(w) = f(w)$.

With this definition the original problem is transformed to the **primal problem**:

$$p^* = \min_w \theta_P(w) = \min_w \max_{\alpha, \beta, \alpha_i \geqslant 0} L(w, \alpha, \beta).$$

The natural way to modify this problem is to switch max and min and formulate the **dual problem**:

$$d^* = \max_{\alpha \geqslant 0, \beta} \theta_D(\alpha, \beta) = \max_{\alpha \geqslant 0, \beta} \min_w L(w, \alpha, \beta),$$

where by definition

$$\theta_D(\alpha, \beta) = \min_w L(w, \alpha, \beta).$$

It is easy to show that $d^* < p^*$, because $\max \min \leqslant \min \max$.

**Example.**

$$\max_{y \in \{0,1\}} \min_{x \in \{0,1\}} \mathbb{1}\{x = y\} \leqslant \min_{x \in \{0,1\}} \max_{y \in \{0,1\}} \mathbb{1}\{x = y\}$$

Notice that $\min\limits_{x \in \{0,1\}} \mathbb{1}\{x = y\} = 0$ and $\max\limits_{y \in \{0,1\}} \mathbb{1}\{x = y\} = 1$.

The important theorem from optimization theory tells that under certain conditions: $d^* = p^*$ and we can solve dual problem instead of primal problem.

**Theorem.** Let

1) $f$ is convex (hessian $H \geqslant 0$);
2) $h_i$ is affine ($h_i(w) = a_i^T w + b_i$);
3) constraints $g_i$ are strictly feasible (there exist $w$ such that for any $i$ $g_i(w) < 0$).

Then

1) there exists $w^*$, $\alpha^*$ and $\beta^*$ such that $w^*$ solves primal problem and $\alpha^*$, $\beta^*$ solve the dual problem and $p^* = d^* = L(w^*, \alpha^*, \beta^*)$;
2) $\dfrac{\partial L}{\partial w}(w^*, \alpha^*, \beta^*) = 0,\ \dfrac{\partial L}{\partial \beta}(w^*, \alpha^*, \beta^*) = 0;$
3) $\alpha_i^* g_i(w^*) = 0$ (Karush-Kuhn-Tucker (KKT) complementarity condition).

Moreover, by definition $\alpha_i^* \geqslant 0$ and from the initial conditions $g_i(w^*) \leqslant 0$, which means that if $\alpha_i^* > 0$, then KKT condition implies $g_i(w^*) = 0$. In most cases,

$$\alpha_i^* > 0 \Leftrightarrow g_i(w^*) = 0$$

($g_i(w)$ is an **active constraint**).

# 2 SVM dual problem

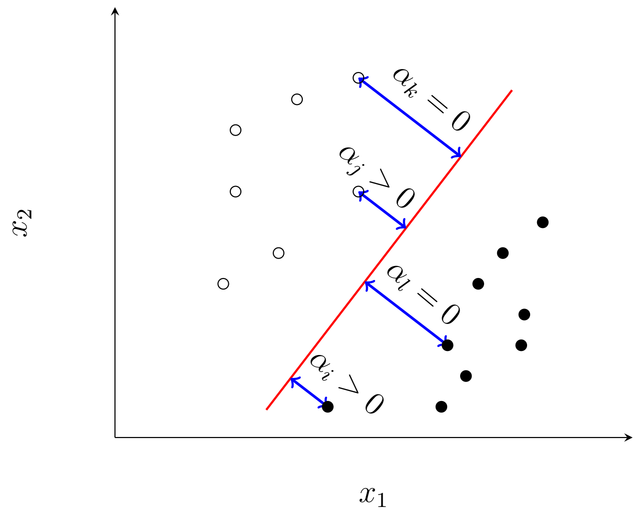
In this section we apply the idea of Lagrange multipliers to the SVM optimization problem and formulate a SVM dual problem. The SVM optimization problem has been formulated in the previous lecture as

$$\min_{w,b} \frac{||w||^2}{2},$$

subject to

$$y^{(i)}(w^T x^{(i)} + b) \geqslant 1,\ i = 1, \ldots, m.$$

We define $g_i(w, b) = -y^{(i)}(w^T x^{(i)} + b) + 1 \leqslant 0$. Notice that we do not have coefficients $\beta$ as there are no constraints for $h$. If $\alpha_i > 0$, then $g_i(w, b) = 0$ (active constraint) and implies that the training example $(x^{(i)}, y^{(i)})$ has a functional margin equals to 1.




The Lagrangian has the form

$$L(w,b,\alpha) = \frac{||w||^2}{2} - \sum_{i=1}^{m} \alpha_i (y^{(i)}(w^T x^{(i)} + b) - 1)$$

and dual problem is

$$\theta_D(\alpha) = \min_{w,b} L(w,b,\alpha).$$

In order to minimize the Lagrangian we find derivatives and set them to zero:

$$\nabla_w L(w,b,\alpha) = w - \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} = 0 \Rightarrow w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \tag{1}$$

and

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{m} y^{(i)} \alpha_i = 0.$$

Substitute these conditions back to the Lagrangian:

$$\begin{aligned}
&L(w,b,\alpha) = \frac{1}{2} w^T w - \sum_{i=1}^{m} \alpha_i (y^{(i)}(w^T x^{(i)} + b) - 1) = \\
&= \frac{1}{2} \left( \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \right)^T \left( \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \right) - \sum_{i=1}^{m} \alpha_i (y^{(i)}(w^T x^{(i)} + b) - 1) = \\
&= \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left\langle x^{(i)}, x^{(j)} \right\rangle - \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left\langle x^{(i)}, x^{(j)} \right\rangle + \sum_{i=1}^{m} \alpha_i = \\
&= \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left\langle x^{(i)}, x^{(j)} \right\rangle = W(\alpha),
\end{aligned}$$

where $\langle \cdot, \cdot \rangle$ is a notation for the dot product of two vectors.

Finally, the **SVM dual problem** is to find

$$\max_{\alpha} W(\alpha)$$

subject to

$$\alpha_i \geqslant 0,$$
$$\sum_i y_i \alpha_i = 0.$$

Notice that if $\sum_i y_i \alpha_i \neq 0$, then $\theta_D(\alpha) = -\infty$, otherwise, $\theta_D(\alpha) = W(\alpha)$.

After we find the solution $\alpha^*$, the coefficients can be found as

$$w = \sum_{i=1}^{m} \alpha_i^* y^{(i)} x^{(i)} \tag{2}$$

and we use the worst positive and negative training examples to find $b$:

$$b = \frac{\max\limits_{i:y^{(i)}=-1} w^T x^{(i)} + \min\limits_{i:y^{(i)}=1} w^T x^{(i)}}{2}.$$

With the equation (1) we can express the entire algorithm in terms of dot products:

$$h_{w,b} = g(w^T x + b) = g\left( \sum_{i=1}^{m} \alpha_i y^{(i)} \left\langle x^{(i)}, x \right\rangle + b \right) \tag{3}$$

# 3 Kernels

The idea of kernels is that often features are high dimensional ($x^{(i)} \in \mathbb{R}^m$) but instead of feature representations it is enough to find dot products.

**Example.** Assuming we have the problem with one feature $x \in \mathbb{R}$ only, the polynomial regression of the fourth order can be represented as a linear regression with the following list of features:

$$\varphi(x) : x \to \begin{bmatrix} x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix}$$

For the SVM optimization problem the hypotheis (3) is replaced by

$$h_{w,b} = g(w^T\varphi(x) + b) = g\left(\sum_{i=1}^{m} \alpha_i y^{(i)} \left\langle \varphi(x)^{(i)}, \varphi(x)\right\rangle + b\right),$$

and in all following calculations we should replace the dot product $\left\langle x^{(i)}, x^{(j)}\right\rangle$ by $\left\langle \varphi(x^{(i)}), \varphi(x^{(j)})\right\rangle$.

There are no any restrictions for the mapping $\varphi(x)$, in fact it is possible to have infinite dimensional $\varphi(x) \in \mathbb{R}^\infty$. Fortunately, for many different $\varphi$ we can specify the function (**kernel**) that defines the dot product:

$$K(x^{(i)}, x^{(j)}) = \left\langle \varphi(x^{(i)}), \varphi(x^{(j)})\right\rangle.$$

In such situations we do not need to compute $\varphi(x)$ explicitly, but we should compute the kernel $K(x,z)$ (which is less computationally expensive than computing $\varphi(x)$).

## 3.1 Kernel examples

1. $K(x,z) = (x^T z)^2$, where $x, z \in \mathbb{R}^n$. We try to transform this kernel to the exact form of dot product:

$$K(x,z) = (x^T z)^2 = \left(\sum_{i=1}^{n} x_i z_i\right)\left(\sum_{j=1}^{n} x_j z_j\right) = \sum_{i=1}^{n}\sum_{j=1}^{n} (x_i x_j)(z_i z_j),$$

that can be interpreted as a dot product of vectors that contains all possible combinations of $x$ and $z$ components. For example, if $n = 3$, then

$$\varphi(x) : x \to \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_1x_3 \\ x_2x_1 \\ x_2x_2 \\ x_2x_3 \\ x_3x_1 \\ x_3x_2 \\ x_3x_3 \end{bmatrix}$$

To compute the dot product $\langle \varphi(x), \varphi(z)\rangle$ for two training examples we need $O(2n^2 + n)$ operations. If we use kernel for that we need $O(n)$

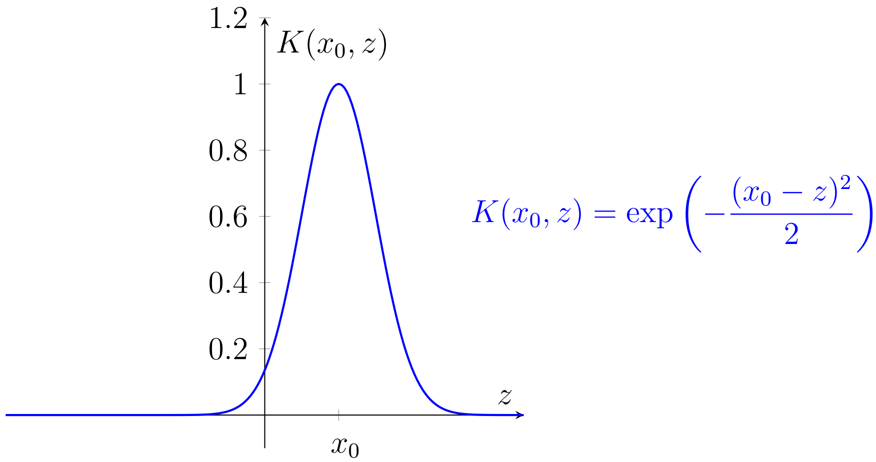
operations only (because we just calculate dot product of two vectors $x^T z$ and take square of it).

2. $K(x,z) = (x^T z + c)^2$ corresponds to

$$\varphi(x) : x \rightarrow \begin{bmatrix} x_1x_1 \\ x_1x_2 \\ x_1x_3 \\ x_2x_1 \\ x_2x_2 \\ x_2x_3 \\ x_3x_1 \\ x_3x_2 \\ x_3x_3 \\ \sqrt{2c}\cdot x_1 \\ \sqrt{2c}\cdot x_2 \\ \sqrt{2c}\cdot x_3 \\ c \end{bmatrix}$$

3. $K(x,z) = (x^T z + c)^d$ corresponds to $\begin{pmatrix} n+d \\ d \end{pmatrix}$ features of all monomials up to degree $d$.

4. $K(x,z) = \exp\left(-\frac{||x-z||^2}{2\sigma^2}\right)$ (**radial basis function (RBF) kernel**) corresponds to the transformation of feature space into an infinite dimensional Hilbert space. The intuition of this kernel is that if $x$ and $z$ are very similar than they will be pointing to the same direction and dot product should be large. In contrast if $x$ and $z$ are very different, then the dot product should be very small. If we fix $x$ and consider $K(x,z)$ as a function of $z$ the graph of this function is a bell shaped function:




## 3.2 Kernel testing

Assuming that we have chosen some function $K(x, z)$ as a kernel. The main question is: does there exist some $\varphi(x)$ such that $K(x, z) = \langle\varphi(x), \varphi(z)\rangle$?

**Definition**. For the given set of points $\{x^{(i)}, \ldots, x^{(m)}\}$ a **kernel matrix** $\mathbf{K} \in \mathbb{R}^{m\times m}$ is defined by

$$\mathbf{K}_{ij} = K(x^{(i)}, x^{(j)}), \tag{4}$$

where $K$ is a kernel function.

**Theorem (Mercer)**. Let $K(x, z)$ be given. Then $K$ is a valid (Mercer) kernel (i.e. there exists $\varphi$ such that $K(x, z) = \langle\varphi(x), \varphi(z)\rangle$) if and only if for all $\{x^{(i)}, \ldots, x^{(m)}\}$ the kernel matrix $\mathbf{K} \in \mathbb{R}^{m\times m}$ is symmetric positive semi-definite.

Indeed, for any vectors $x, z \in \mathbb{R}^n$

$$\begin{aligned}
&z^T\mathbf{K}z = \sum_i\sum_j z_i\mathbf{K}_{ij}z_j = \sum_i\sum_j z_i\varphi(x^{(i)})^T\varphi(x^{(j)})z_j = \\
&= \sum_i\sum_j z_i \sum_k(\varphi(x^{(i)}))_k(\varphi(x^{(j)}))_k\, z_j = \sum_k\sum_i\sum_j z_i\, (\varphi(x^{(i)}))_k(\varphi(x^{(j)}))_k\, z_j = \\
&= \sum_k\left(\sum_i z_i\varphi(x^{(i)})_k\right)^2 \geqslant 0.
\end{aligned}$$

Here we used a fact that $a^Tb = \sum\limits_k a_kb_k$.

**Example**. $K(x, z) = -1$ is not a valid kernel function.

### 3.3 SVM with kernels

We can reformulate the SVM dual problem from the previous section as follows:

$$\max_{\alpha} W(\alpha) = \max_{\alpha} \left( \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j K\left(x^{(i)}, x^{(j)}\right) \right)$$

subject to

$$\begin{aligned} &\alpha_i \geqslant 0, \\ &\sum_i y_i \alpha_i = 0, \end{aligned}$$

with the hypothesis

$$h_{w,b} = g\left( \sum_{i=1}^{m} \alpha_i y^{(i)} K(x^{(i)}, x) + b \right), \tag{5}$$

where $K$ is a chosen kernel function.

The last remark is that the kernel idea is more general than SVM and we can formulate many algorithms in terms of dot products.

## 4 Soft margin

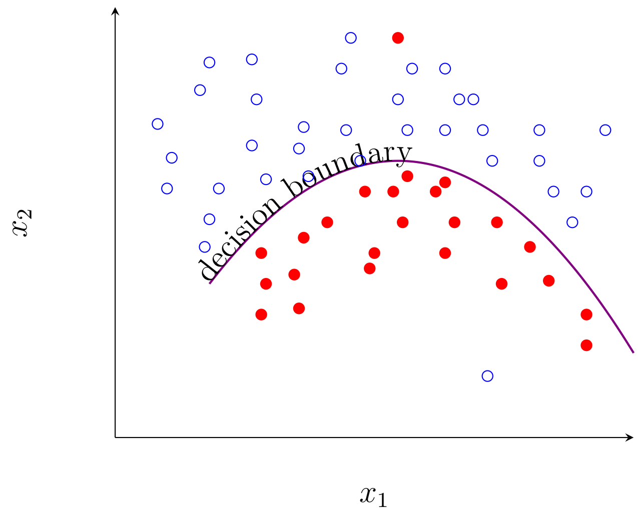
In case of non linear decision boundaries the SVM algorithm is called $L_1$ **norm soft margin SVM** and formulated as follows:

$$\min_{w} \frac{||w||^2}{2} + C \sum_{i=1}^{m} \xi_i$$

subject to

$$y^{(i)}(w^T x^{(i)} + b) \geqslant 1 - \xi_i,\ \xi_i \geqslant 0,\ i = 1, \ldots, m.$$

Such formulation is useful for non linear separable datasets, for example, in the next picture we cannot find the hyperplane that separates two classes.




Remember that if $y^{(i)}(w^T x^{(i)} + b) > 0$, then the example is classified correctly. With the above formulation we allow the algorithm to misclassify something (because of the term $1 - \xi_i$), but we encourage the algorithm not to do it, because it will increase the objective function by $\sum_{i=1}^{m} \xi_i$. Notice that this is also convex optimization problem.

As before we find the derivatives of Lagrangian

$$L(w, b, \xi, \alpha, r) = \frac{1}{2}||w||^2 + C\sum_i \xi_i - \sum_{i=1}^{m} \alpha_i(y^{(i)}(w^T x^{(i)} + b) - 1 + \xi_i) - \sum_{i=1}^{m} r_i \xi_i$$

and equate them to zero:

$$\nabla_w L(w, b, \xi, \alpha, r) = w - \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} = 0 \Rightarrow w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)},$$
$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{m} \alpha_i y^{(i)} = 0,$$
$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - r_i = 0.$$

We can also add the KKT conditions:

$$\alpha_i(y^{(i)}(w^T x^{(i)} + b) - 1 + \xi_i) = 0,$$
$$r_i \xi_i = 0.$$

Taking into consideration all these conditions we derive

$$\alpha_i = 0 \Rightarrow r_i = C > 0 \Rightarrow \xi_i = 0 \Rightarrow y^{(i)}(w^T x^{(i)} + b) \geqslant 1 \tag{6}$$

$$\alpha_i = C \Rightarrow y^{(i)}(w^T x^{(i)} + b) = 1 - \xi_i \Rightarrow y^{(i)}(w^T x^{(i)} + b) \leqslant 1 \tag{7}$$
$$0 < \alpha_i < C \Rightarrow r_i > 0 \Rightarrow \xi_i = 0 \Rightarrow y^{(i)}(w^T x^{(i)} + b) = 1 \tag{8}$$

To obtain the dual problem we substitute all these conditions to the Lagrangian:

$$\begin{aligned}
&L(w,b,\xi,\alpha,r) = \frac{1}{2}w^T w + C\sum_i \xi_i - \sum_{i=1}^m \alpha_i(y^{(i)}(w^T x^{(i)}+b) - 1 + \xi_i) - \sum_{i=1}^m r_i \xi_i = \\
&= \frac{1}{2}\left(\sum_{i=1}^m \alpha_i y^{(i)} x^{(i)}\right)^T \left(\sum_{i=1}^m \alpha_i y^{(i)} x^{(i)}\right) - \sum_{i=1}^m \alpha_i (y^{(i)}(w^T x^{(i)} + b) - 1) - \sum_i (C - \alpha_i - r_i)\xi_i = \\
&= \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j \left\langle x^{(i)}, x^{(j)} \right\rangle - \sum_{i=1}^m \sum_{j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j \left\langle x^{(i)}, x^{(j)} \right\rangle + \sum_{i=1}^m \alpha_i = \\
&= \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j \left\langle x^{(i)}, x^{(j)} \right\rangle = W(\alpha).
\end{aligned}$$

Finally, the dual optimization problem with the kernel idea is stated as

$$\max_\alpha W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m y^{(i)} y^{(j)} \alpha_i \alpha_j K(x^{(i)}, x^{(j)}) \tag{9}$$

subject to

$$\begin{aligned}
&\sum_{i=1}^m y^{(i)} \alpha_i = 0, \\
&0 \leqslant \alpha_i \leqslant C,\ i = 1, \ldots, m.
\end{aligned} \tag{10}$$

# 5 SMO algorithm

In this section we come up with an efficient algorithm that solves the SVM optimization problem. First, consider the problem

$$\max_\alpha W(\alpha_1, \ldots, \alpha_m)$$

without constraint on $\alpha$'s.

**Algorithm 1** Coordinate ascent algorithm

1: **repeat**
2:     **for** $i = 1$ **to** $m$ **do**
3:         $\alpha_i = \arg\max_{\hat{\alpha}_i} W(\alpha_1, \ldots, \alpha_{i-1}, \hat{\alpha}_i, \alpha_{i+1}, \ldots, \alpha_m)$ (freeze all variables except $\alpha_i$)
4: **until** convergence

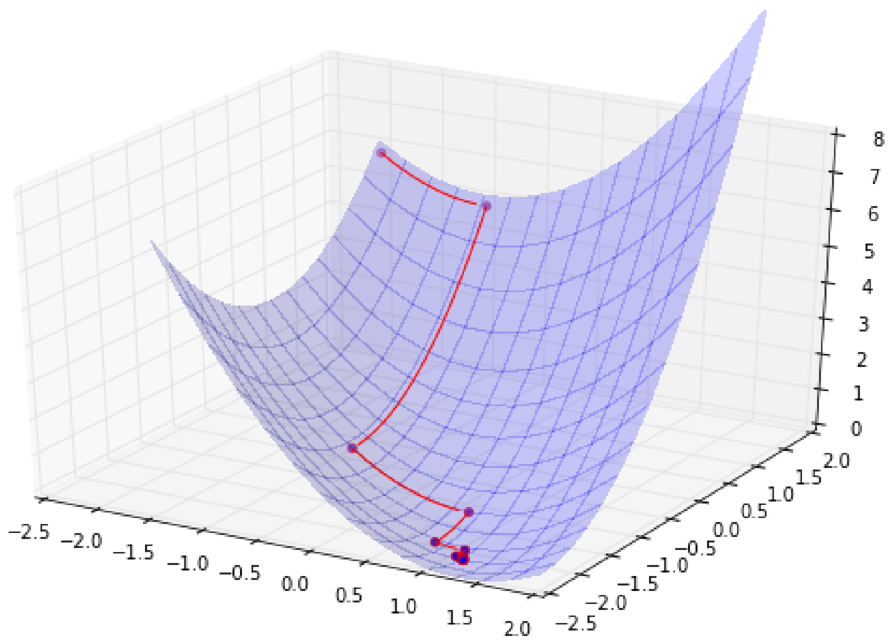



Figure 1: Coordinate ascent visualization

Compared to the gradient descent this algorithm takes much more steps, but for many optimization problems it is very easy to make step by one parameter.

We apply it for our SVM dual optimization problem. Unfortunately, this algorithm does not work in a straight way, because of the condition $\sum_{i=1}^{m} y^{(i)}\alpha_i = 0$ (if we fix all $\alpha$'s except one, then we can find the last $\alpha$ explicitly). That's why we try to optimize two $\alpha$'s per step.

**Algorithm 2** Sequential Minimal Optimization (SMO) algorithm

1: **repeat**
2:     **for** pairs $\alpha_i$ and $\alpha_j$ **do**
3:         $\alpha_i, \alpha_j = \arg\max_{\hat{\alpha}_i, \hat{\alpha}_j} W(\alpha_1, \ldots, \hat{\alpha}_i, \ldots, \hat{\alpha}_j, \ldots, \alpha_m)$ (freeze all variables except $\alpha_i$ and $\alpha_j$)
4: **until** convergence

We elaborate more about the implementation of this algorithm. Without loss of generality we update $\alpha_1$ and $\alpha_2$. The general case could be obtained in the same manner by replacing $\alpha_1$ by $\alpha_i$ and $\alpha_2$ by $\alpha_j$. We assume that we have $\alpha_i^{old}$ from the previous step of the SMO algorithm, for which conditions (10) hold:

$$\begin{aligned}
&\sum_{i=1}^{m} y^{(i)}\alpha_i^{old} = 0,\\
&0 \leqslant \alpha_i^{old} \leqslant C.
\end{aligned}$$

The first equation can be transformed to

$$\alpha_1^{old}y^{(1)} + \alpha_2^{old}y^{(2)} = -\sum_{i=3}^{m}\alpha_i^{old}y^{(i)} \Rightarrow \alpha_1^{old} + \alpha_2^{old}y^{(2)}y^{(1)} = -y^{(1)}\sum_{i=3}^{m}\alpha_i^{old}y^{(i)},$$

and the right-hand side of the last equation will be denoted by $\zeta$, then

$$\begin{aligned}
&\alpha_1^{old} + s\alpha_2^{old} = \alpha_1 + s\alpha_2 = \zeta,\\
&\text{where } s = y^{(1)}y^{(2)}.
\end{aligned}$$

Notice that $\zeta$ does not change after one step of the SMO algorithm.

We introduce the following notations (using (4)):

$$\begin{aligned}
&h(x) = \sum_{i=1}^{m}\alpha_i^{old}y^{(i)}K(x^{(i)},x) + b,\\
&v_j = \sum_{i=3}^{m} y^{(i)}\alpha_i\mathbf{K}_{ij} = h(x^{(j)}) - b - \alpha_1^{old}y^{(1)}\mathbf{K}_{1j} - \alpha_2^{old}y^{(2)}\mathbf{K}_{2j}.
\end{aligned}$$

Then

$$\begin{aligned}
&W(\alpha_1,\alpha_2,\ldots,\ldots) = \alpha_1 + \alpha_2 - \frac{1}{2}\sum_{i=1}^{m} y^{(i)}y^{(1)}\alpha_i\alpha_1\mathbf{K}_{i1}\\
&-\frac{1}{2}\sum_{i=1}^{m} y^{(i)}y^{(2)}\alpha_i\alpha_2\mathbf{K}_{i2} + V(\alpha_3,\ldots,\alpha_m) =
\end{aligned}$$

separate first two terms for each sum:

$$\begin{aligned}
&= \alpha_1 + \alpha_2 - y^{(1)}y^{(2)}\alpha_1\alpha_2\mathbf{K}_{12} - \frac{1}{2}(\alpha_1)^2\mathbf{K}_{11} - \alpha_1 y^{(1)}\sum_{i=3}^{m} y^{(i)}\alpha_i\mathbf{K}_{i1}\\
&-\frac{1}{2}(\alpha_2)^2\mathbf{K}_{22} - \alpha_2 y^{(2)}\sum_{i=3}^{m} y^{(i)}\alpha_i\mathbf{K}_{i2} + V(\alpha_3,\ldots,\alpha_m) =\\
&= \alpha_1 + \alpha_2 - s\alpha_1\alpha_2\mathbf{K}_{12} - \frac{1}{2}(\alpha_1)^2\mathbf{K}_{11} - \frac{1}{2}(\alpha_2)^2\mathbf{K}_{22}\\
&-\alpha_1 y^{(1)}v_1 - \alpha_2 y^{(2)}v_2 + V(\alpha_3,\ldots,\alpha_m) =
\end{aligned}$$

substitute the expression $\alpha_1 = \zeta - s\alpha_2$:

$$\begin{aligned}
&= \zeta - s\alpha_2 + \alpha_2 - s(\zeta - s\alpha_2)\alpha_2\mathbf{K}_{12} - \frac{1}{2}(\zeta - s\alpha_2)^2\mathbf{K}_{11} - \frac{1}{2}(\alpha_2)^2\mathbf{K}_{22} \\
&-(\zeta - s\alpha_2)y^{(1)}v_1 - \alpha_2 y^{(2)}v_2 + V(\alpha_3, \ldots, \alpha_m).
\end{aligned}$$

We have obtained the quadratic function with respect to $\alpha_2$. To find the maximum we find the derivative and equate it to zero:

$$\begin{aligned}
&W'_{\alpha_2} = -s + 1 - s\zeta\mathbf{K}_{12} + 2\alpha_2\mathbf{K}_{12} \\
&+\zeta s\mathbf{K}_{11} - \alpha_2\mathbf{K}_{11} - \alpha_2\mathbf{K}_{22} + y^{(2)}v_1 - y^{(2)}v_2 = 0.
\end{aligned}$$

Then

$$2\alpha_2\mathbf{K}_{12} - \alpha_2\mathbf{K}_{11} - \alpha_2\mathbf{K}_{22} = s - 1 + s\zeta\mathbf{K}_{12} - \zeta s\mathbf{K}_{11} - y^{(2)}v_1 + y^{(2)}v_2.$$

Substitute the expressions for $v_1$, $v_2$ and $\zeta = \alpha_1^{old} + s\alpha_2^{old}$:

$$\begin{aligned}
&\alpha_2(2\mathbf{K}_{12} - \mathbf{K}_{11} - \mathbf{K}_{22}) = s - 1 + s(\alpha_1^{old} + s\alpha_2^{old})(\mathbf{K}_{12} - \mathbf{K}_{11}) \\
&-y^{(2)}(h(x^{(1)}) - b - \alpha_1^{old}y^{(1)}\mathbf{K}_{11} - \alpha_2^{old}y^{(2)}\mathbf{K}_{12}) \\
&+y^{(2)}(h(x^{(2)}) - b - \alpha_1^{old}y^{(1)}\mathbf{K}_{12} - \alpha_2^{old}y^{(2)}\mathbf{K}_{22}) = \\
&= s - 1 + \alpha_2^{old}(2\mathbf{K}_{12} - \mathbf{K}_{11} - \mathbf{K}_{22}) - y^{(2)}h(x^{(1)}) + y^{(2)}h(x^{(2)})
\end{aligned}$$

and finally, using $s - 1 = y^{(2)}(y^{(1)} - y^{(2)})$:

$$\alpha_2 = \alpha_2^{old} - y^{(2)}\frac{(h(x^{(1)}) - y^{(1)}) - (h(x^{(2)}) - y^{(2)})}{2\mathbf{K}_{12} - \mathbf{K}_{11} - \mathbf{K}_{22}} \tag{11}$$

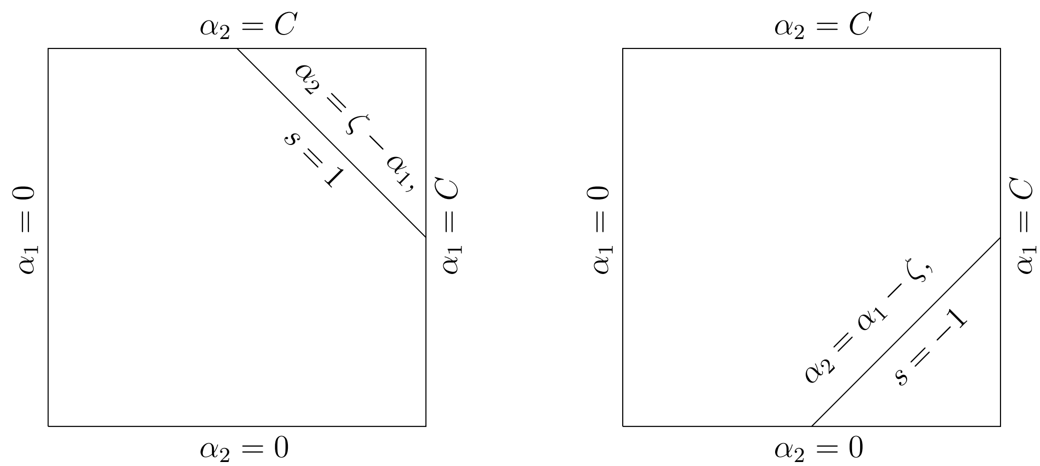
and

$$\alpha_1 = \zeta - s\alpha_2 = \alpha_1^{old} + s(\alpha_2^{old} - \alpha_2) \tag{12}$$

We have not used the conditions $0 \leqslant \alpha_i \leqslant C$ yet. Remember that $\zeta$ is a constant during each step of the SMO algorithm, which means that we can consider the equation

$$\alpha_1 + s\alpha_2 = \zeta \Rightarrow \alpha_2 = s\zeta - s\alpha_1$$

as an equation of the straight line.




There are two possible cases (see the figures):

- If $s = 1$, then $\alpha_2 = \zeta - \alpha_1$. Then we will have the following chain of implications:

$$0 \leqslant \alpha_1 \leqslant C \Rightarrow \zeta - C \leqslant \zeta - \alpha_1 \leqslant \zeta \Rightarrow 0 \leqslant \zeta - C \leqslant \alpha_2 \leqslant \zeta \leqslant C,$$

  which means that $\max(0, \zeta - C) \leqslant \alpha_2 \leqslant \min(\zeta, C)$, or using $\zeta = \alpha_1^{old} + s\alpha_2^{old}$ ($s = 1$):

$$\max(0, \alpha_1^{old} + \alpha_2^{old} - C) \leqslant \alpha_2 \leqslant \min(\alpha_1^{old} + \alpha_2^{old}, C) \qquad (13)$$

- If $s = -1$, then $\alpha_2 = \alpha_1 - \zeta$. In this case:

$$0 \leqslant \alpha_1 \leqslant C \Rightarrow -\zeta \leqslant \alpha_1 - \zeta \leqslant C - \zeta \Rightarrow 0 \leqslant -\zeta \leqslant \alpha_2 \leqslant C - \zeta \leqslant C,$$

  which means that $\max(0, -\zeta) \leqslant \alpha_2 \leqslant \min(C - \zeta, C)$, or using $\zeta = \alpha_1^{old} + s\alpha_2^{old}$ ($s = -1$):

$$\max(0, \alpha_2^{old} - \alpha_1^{old}) \leqslant \alpha_2 \leqslant \min(C + \alpha_2^{old} - \alpha_1^{old}, C) \qquad (14)$$

The KKT conditions also give the formula to calculate $b$. Assuming that after one step of the SMO algorithm we got $0 < \alpha_2 < C$, then

$$y^{(2)}(w^T x^{(2)} + b) = 1 \Rightarrow w^T x^{(2)} + b = y^{(2)},$$

implies

$$\begin{aligned}
&b_2 = y^{(2)} - w^T x^{(2)} = y^{(2)} - \sum_{i=1}^{m} y^{(i)} \alpha_i \mathbf{K}_{i2} = \\
&= y^{(2)} - y^{(1)}\alpha_1 \mathbf{K}_{12} - y^{(2)}\alpha_2 \mathbf{K}_{22} - \sum_{i=3}^{m} y^{(i)} \alpha_i \mathbf{K}_{i2} = \\
&= y^{(2)} - y^{(1)}\alpha_1 \mathbf{K}_{12} - y^{(2)}\alpha_2 \mathbf{K}_{22} - (h(x^{(2)}) - b - \alpha_1^{old} y^{(1)} \mathbf{K}_{12} - \alpha_2^{old} y^{(2)} \mathbf{K}_{22}) = \\
&= b^{old} - (h(x^{(2)}) - y^{(2)}) - y^{(2)} \mathbf{K}_{22}(\alpha_2 - \alpha_2^{old}) - y^{(1)} \mathbf{K}_{12}(\alpha_1 - \alpha_1^{old})
\end{aligned} \tag{15}$$

Similarly, if $0 < \alpha_1 < C$:

$$b_1 = b^{old} - (h(x^{(1)}) - y^{(1)}) - y^{(2)} \mathbf{K}_{12}(\alpha_2 - \alpha_2^{old}) - y^{(1)} \mathbf{K}_{11}(\alpha_1 - \alpha_1^{old}) \tag{16}$$

If none of the conditions $0 < \alpha_1 < C$ and $0 < \alpha_2 < C$ is true, then we can take the average $\dfrac{b_1 + b_2}{2}$ (any $b$ between $b_1$ and $b_2$ satisfies to the KKT conditions).

When we switch to the general case and take any pair of $\alpha_i$, $\alpha_j$, first we choose $\alpha_j$ such that it does not satisfy the KKT condition (8) (with some tolerance $\gamma$):

$$0 < \alpha_j < C \Rightarrow y^{(j)}(w^T x^{(j)} + b) = 1 \Rightarrow y^{(j)}(h(x^{(j)}) - y^{(j)}) = 0 \tag{17}$$

Also notice that if $\alpha_j = C$, then we could have $y^{(j)}(h(x^{(j)}) - y^{(j)}) < 0$ and if $\alpha_j = 0$, then we could have $y^{(j)}(h(x^{(j)}) - y^{(j)}) > 0$. The following algorithm summarizes all our calculations with references to the formulas.

# References

[1] J. Platt. "Fast Training of Support Vector Machines using Sequential Minimal Optimization", *in Advances in Kernel Methods - Support Vector Learning, B. Scholkopf, C. Burges, A. Smola, eds., MIT Press*, 1998.

 *REFERENCES*

**Algorithm 3** SMO algorithm for Support Vector Machine

**set** $C$, $\gamma$, initial values $\alpha_i = 0$, $b = 0$<br>
**repeat**<br>
  **for** $j = 1$ **to** $m$ **do**<br>
    evaluate $E_j = h(x^{(j)}) - y^{(j)}$<br>
    **if** $(y^{(j)}E_j < -\gamma$ and $\alpha_j < C)$ or $(y^{(j)}E_j > \gamma$ and $\alpha_j > 0)$ **then** ▷ (17)<br>
      **repeat**<br>
        choose $\alpha_i, i \neq j$, randomly<br>
        evaluate $E_i = h(x^{(i)}) - y^{(i)}$<br>
        **if** $y^{(i)} \cdot y^{(j)} > 0$ **then**<br>
          $L = \max(0, \alpha_i + \alpha_j - C)$ ▷ (13)<br>
          $H = \min(\alpha_i + \alpha_j, C)$ ▷ (13)<br>
        **else**<br>
          $L = \max(0, \alpha_j - \alpha_i)$ ▷ (14)<br>
          $H = \min(C + \alpha_j - \alpha_i, C)$ ▷ (14)<br>
        **if** $L == H$ **then continue**<br>
        evaluate $\eta = 2\mathbf{K}_{ij} - \mathbf{K}_{ii} - \mathbf{K}_{jj}$<br>
        **if** $\eta == 0$ **then continue**<br>
        evaluate $\alpha_j^{new} = \min(\max(\alpha_j - y^{(j)}\dfrac{E_i - E_j}{\eta}, L), H)$ ▷ (11)<br>
        **if** $|\alpha_j^{new} - \alpha_j| < 10^{-5}$ **then continue**<br>
        evaluate $\alpha_i^{new} = \alpha_i + y^{(j)}y^{(i)}(\alpha_j - \alpha_j^{new})$ ▷ (12)<br>
        **if** $(\alpha_j^{new} > 0$ and $\alpha_j^{new} < C)$ **then**<br>
          $b = b - E_j - y^{(j)}\mathbf{K}_{jj}(\alpha_j^{new} - \alpha_j) - y^{(i)}\mathbf{K}_{ij}(\alpha_i^{new} - \alpha_i)$ ▷ (15)<br>
        **else**<br>
          **if** $(\alpha_i^{new} > 0$ and $\alpha_i^{new} < C)$ **then**<br>
            $b = b - E_i - y^{(j)}\mathbf{K}_{ij}(\alpha_j^{new} - \alpha_j) - y^{(i)}\mathbf{K}_{ii}(\alpha_i^{new} - \alpha_i)$ ▷ (16)<br>
          **else**

$$
\begin{aligned}
b = b - 0.5 \cdot (&E_i + E_j \\
&+ y^{(j)}(\mathbf{K}_{jj} + \mathbf{K}_{ij})(\alpha_j^{new} - \alpha_j) \\
&+ y^{(i)}(\mathbf{K}_{ij} + \mathbf{K}_{ii})(\alpha_i^{new} - \alpha_i))
\end{aligned}
$$

▷ (15), (16)

      **until** False<br>
**until** convergence<br>
**return** $\alpha$, $b$