Duality in Optimization

Lena Birschitzky, Patrick Pollek
Uni Wien

***Abstract*—This project explores the concept of duality in optimization, i.e. the approach of solving optimization problems by considering their dual version. First we give a short introduction to the theory. Then we compare primal and dual solution methods for two standard optimization problems: the Support Vector Machine (SVM) problem and the Minimum Enclosing Ball (MEB) problem.**

I. Introduction

An optimization problem is usually given in its primal form, from which we can also derive a dual formulation. The optimal value of the dual problem is a lower bound for that of the primal problem, and under some conditions they are equal. We can use this fact, as well as the changed dimensions in the dual problem, to more efficiently solve certain primal problems.

In this project we consulted the following literature: [1] [2] [3] [4] [5] [6]

II. The Structure of our Paper

Introduction
Theory on Duality
Support Vector Machine Problem
Minimal Enclosing Ball Problem
Implementation for SVM
Conclusion

III. Theory on Duality

We will consider an optimation problem in the standard form

$$\begin{aligned} &\text{minimize} && f_0(x) && \text{(1a)}\\ &\text{subject to} && f_i(x) \leq 0, \quad i = 1, \dots, m && \text{(1b)}\\ &&& h_i(x) = 0, \quad i = 1, \dots, p && \text{(1c)} \end{aligned}$$

where $x \in \mathbb{R}^n$ is our variable. We will call this a primal problem, and we will call the optimal value of this problem p^*. In general this poses a non convex problem.

A. The Lagrange dual function

The key idea of the Lagrange duality is to include the constrains of our problem into a new function. We define the Lagrangian $L : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ as

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x). \tag{2}$$

We call λ_i the Lagrange multiplier associated with the i-th inequality constraint $f_i(x)$ and analogous for ν_i. The vectors $\lambda = (\lambda_1, \dots, \lambda_m)$ and $\nu = (\nu_1, \dots, \nu_p)$ are called dual variables associated with the problem.

We now can define the Lagrange dual function $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ in the following way, for some $\lambda \in \mathbb{R}^m, \nu \in \mathbb{R}^p$:

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) = \inf_{x \in \mathcal{D}} \left(f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x) \right) \tag{3}$$

and we will also refer to this as the dual problem correspoding to (1).

B. Using the dual as a lower bound

Solving this yields a lower bound for the solution p^* of the primal problem: For any $\lambda \succeq 0$ and any ν we have

$$g(\lambda, \nu) \leq p^*. \tag{4}$$

Since we are interested in getting as close as possible to p^*, the so called Lagrange dual problem naturally arises:

$$\begin{aligned} &\text{maximise} && g(\lambda, \nu) && \text{(5a)}\\ &\text{subject to} && \lambda \succeq 0. && \text{(5b)} \end{aligned}$$

We will refer to the solution of the dual problem (5) as d^*, and we immediately see that

$$d^* \leq p^*. \tag{6}$$

In some settings this inequality can be shown to be an equality, then we also say that the optimal duality gap $p^* - d^*$ is zero, and that strong duality holds. For our purposes the following result is very useful:

C. The KKT conditions

Given that the primal problem is convex and satisfies the Karush-Kuhn-Tucker (KKT) conditions, we have $d^* = p^*$. The KKT conditions are as follows:

$$\begin{aligned} f_i(x^*) &\leq 0, \ i = 1, \dots, m && \text{(7a)}\\ h_i(x^*) &= 0, \ i = 1, \dots, p && \text{(7b)}\\ \lambda_i^* &\geq 0, \ i = 1, \dots, m && \text{(7c)}\\ \lambda_i^* f_i(x^*) &= 0, \ i = 1, \dots, m && \text{(7d)}\\ \nabla f_0(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla f_i(x^*) + \sum_{i=1}^{p} \nu_i^* \nabla h_i(x^*) &= 0. && \text{(7e)} \end{aligned}$$

IV. Support Vector Machine Problem

The SVM optimization problem is defined as follows:

$$\begin{aligned} &\text{minimize} && \gamma && (8a)\\ &\text{subject to} && y^{(i)}(w^T x^{(i)} + b) \geq \gamma, \quad i = 1, \dots, n && (8b)\\ & && \|w\| = 1. && (8c) \end{aligned}$$

Here, $X \in \mathbb{R}^{n \times d}$, $w \in \mathbb{R}^d$, meaning the data consists of n instances in d-dimesional space, and we are optimizing a vector in $\mathbb{R}^d$.

We can reformulate this to a nicer (i.e. convex) problem:

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\|w\|^2 && (9a)\\ &\text{subject to} && y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad i = 1, \dots, n. && (9b) \end{aligned}$$

Let us also derive the dual version of this problem.

The Lagrangian of the Problem in (9) is given by

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i [y^{(i)}(w^T x^{(i)} + b) - 1] \quad (10)$$

We now have to find the Lagrange Dual Function (see 3). This will be done by taking the derivative with respect to w and b and setting it to zero. We have

$$\nabla_w \mathcal{L}(w, b, \alpha) = w - \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)} = 0 \quad (11)$$

which implies

$$w = \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)} \quad (12)$$

and

$$\nabla_b \mathcal{L}(w, b, \alpha) = \sum_{i=1}^{n} \alpha_i y^{(i)} = 0. \quad (13)$$

Plugging (12) into Equation (10) we get

$$\begin{aligned} \mathcal{L}(b, \alpha) = \frac{1}{2}(\sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)})^T \sum_{j=1}^{n} \alpha_j y^{(j)} x^{(j)} \\ + \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i [y^{(i)} (\sum_{j=1}^{n} \alpha_j y^{(j)} x^{(j)T} x^{(i)} + b)], \end{aligned} \quad (14)$$

which simplifies to

$$\begin{aligned} \mathcal{L}(b, \alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle \\ - b \sum_{i}^{n} \alpha_i y^i. \end{aligned} \quad (15)$$

The last term is zero due to (13) so that we arrive at the dual problem:

$$\begin{aligned} &\text{maximize} && \mathcal{L}(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} y^{(i)} y^{(j)} \alpha_i \alpha_j \langle x^{(i)}, x^{(j)} \rangle && (16a)\\ &\text{subject to} && \alpha_i \geq 0, \ i = 1, \dots, n && (16b)\\ & && \sum_{i=1}^{n} \alpha_i y^{(i)} = 0, && (16c) \end{aligned}$$

where the first constrained is due to (5) and the second one due to (13).

Notice that we are now optimizing a parameter $\alpha \in \mathbb{R}^n$, so we have changed the dimensions of the problem.

One can easily check that the SVM problem satisfies the KKT conditions, so the duality gap is 0 and we could solve the primal problem by solving this dual problem instead.

V. Minimal Enclosing Ball Problem

Given a set of points $A := \{a_1, \dots, a_m\} \subset \mathbb{R}^n$, a minimum enclosing ball is a ball with minimal radius for which all points of A lie on the inside. It can be described by the following (primal) optimization problem:

$$\begin{aligned} &\min_{c,r} && r \\ &\text{s.t.} && ||a_i - c|| \leq r \quad i = 1, \dots, m \end{aligned} \quad (17)$$

The Problem (17) can be reformulated by squaring the constraints and substituting $\gamma := r^2$:

$$\begin{aligned} &\min_{c,\gamma} && \gamma \\ &\text{s.t.} && (a^i)^T a^i - 2(a^i)^T c + c^T c \leq \gamma \quad i = 1, \dots, m \end{aligned} \quad (18)$$

We can then take the Lagrangian of (18) to arrive at the dual problem:

$$\begin{aligned} &\max_{u} && \Phi(u) \\ &\text{s.t.} && \sum_{i=1}^{m} u_i = 1 \\ & && u_i \geq 0 \quad i = 1, \dots, m \end{aligned} \quad (19)$$

where

$$\Phi(u) := \sum_{i=1}^{m} u_i (a^i)^T a^i - \Big(\sum_{i=1}^{m} u_i a^i \Big)^T \Big(\sum_{i=1}^{m} u_i a^i \Big)$$

VI. Runtime of Primal and Dual SVM for Different Dimensions

We conducted a Monte Carlo simulation to compare the runtime for fitting primal and dual models. We used the Python library *scikit − learn*, specifically the class *sklearn.svm.LinearSVC* [1] to fit models to artificially created problems. These Problems were created using

[1] This method uses the library LIBLINEAR, which uses a coordinate descent method for the primal and a trust region Newton method for the dual. https://www.jmlr.org/papers/volume9/fan08a/fan08a.pdf

the $sklearn.datasets.make\_classification$ method with different parameters which we will specify. Non-specified parameters are left on default values. We set the maximum number of iterations for each classifier to 200000.

The results are summarised in Table 1.

To further visualize these results, we are plotting the time to fit for each method with fixed number of features and varying number of samples (datapoints).

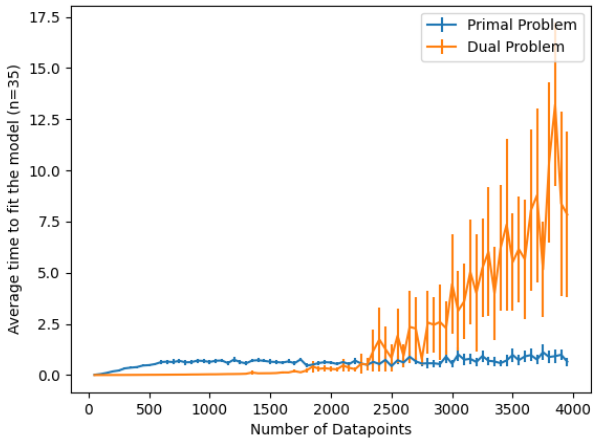

Fig. 1. Average time to fit the Model [lower means better] with fixed number of features (600). On the x-axis we see the number of Samples. We can see that the Dual is better when we only have few data points and gets much slower when dealing with a lot of data points. The Primal is at the beginning slower and rises continuously but after a certain point of data points, the time to fit the model becomes more ore less constant (in this setting).The error bars represent the 95 % confidence interval. (Parameters where: Number of features : 600, Number of informative features: 20, Number of clusters per class: 1,Number of Monte-Carlo iterations : 35

This confirms our observations in IV: Since the primal solves a problem in d = #features and the dual in n = #datapoints, we exect the dual to be faster when $d >> n$.

All experiments where conducted on a Windows 10 machine with an AMD Ryzen 7 3800X 8-Core Processor [3.90 GHz] and 16 GB of Ram (DDR4) on a single thread.

VII. Implementation of SGD for Primal and Dual SVM

We wrote our own implementation of Stochastic Gradient Descent (based on what we had already done for the lecture), in order to generate learning curves for the primal and dual problem.

We found it easiest to work with a slightly different formulation of the primal problem which makes the constraints implicit:

$$\min_{w \in R^d} \sum_{i=1}^{n} \ell(y_i X_i^\top w) + \frac{1}{2}|w\|^2 \tag{20}$$

where $\ell : R \to R$, $\ell(z) := \max\{0, 1 - z\}$ is the hinge loss function.

We also used matrix-vector notation for the dual:

$$\max_{\boldsymbol{\alpha} \in R^n} \alpha^\top \mathbf{1} - \tfrac{1}{2}\alpha^\top Y X X^\top Y \alpha \text{ such that } 0 \le \alpha_i \le 1 \ \forall i, \tag{21}$$

where $Y := diag(y)$.

The gradient of 21 can be expressed as the sum of the entries of the following vector:

$$diag(\alpha * y) \cdot X \cdot x_k^T, \tag{22}$$

where $*$ is pointwise multiplication and x_k is the k-th row of X.

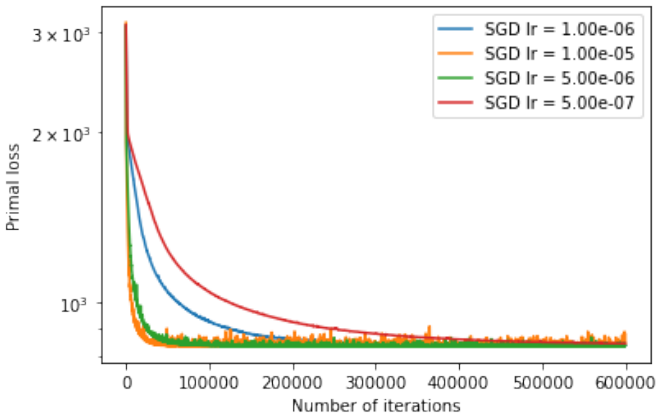

Fig. 2. Learning curve for the primal method

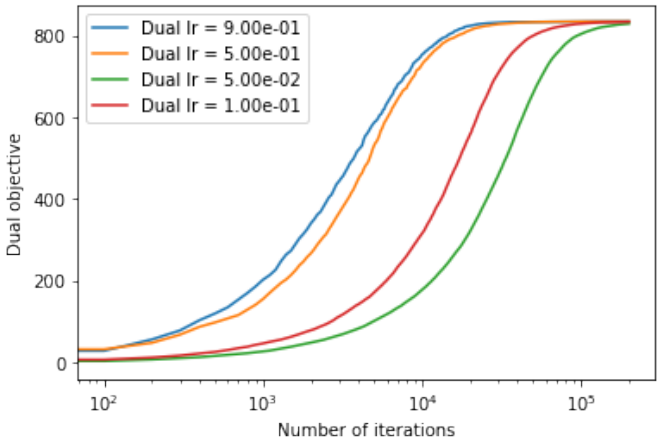

Fig. 3. Learning curve for the dual method

We tested different learning rates for the primal and dual problem, and plotted how the loss and accuracy evolve over the number of iterations. We observed that with suitable

learning rates, both methods came close to the optimal value quite quickly. Both methods took about the same number of iterations to achieve the same accuracy. However the time needed for each single iteration was different for primal and dual, depending on the dimensions of the problem (cf. VI).

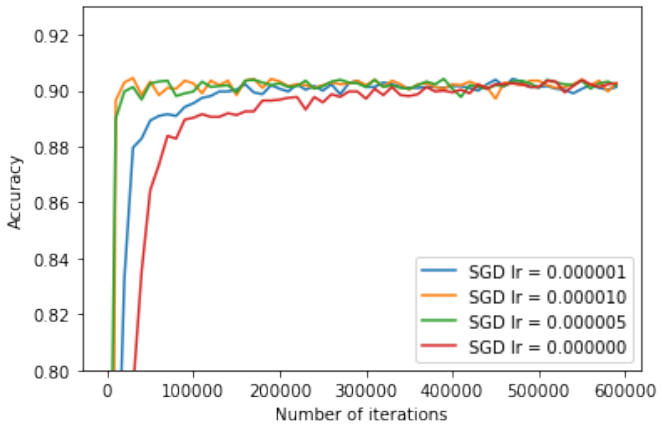

Fig. 4. Accuracy curve for the primal method

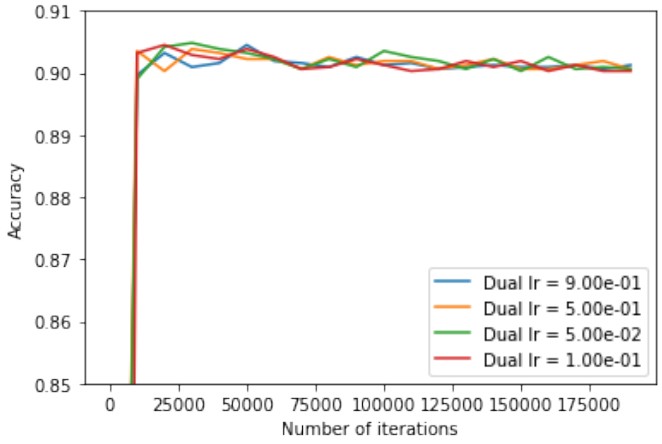

Fig. 5. Accuracy curve for the dual method

We also observed that since the duality gap is zero for this problem, using both methods for the same problem gives a guarantee for how close we already are to the optimal objective value, because we are approximating it from above and below.

VIII. A Small Remark About Kernels

We saw in equation 16 that we have to compute a dot product between datapoints. This allows us to use the kernel trick to map implicitly to a higher dimensional space by replacing this dot product. By doing so we are able solve problems where the data is not linear separable in the original space but indeed is linear separable in the higher dimensional space. We could use a wide variety of Kernels such as polynomial Kernels or radial basis function kernel (RBF).

IX. Conclusion

We introduced the theory behind duality in optimization. We applied this to the SVM problem (which is convex and satisfies the KKT conditions) by solving some examples with both primal and dual SGD. We were able to verify for our examples that the duality gap is zero, and that both methods come close to the optimum. We can therefore confirm that one can freely choose between the primal and dual method, and one should decide based on the dimensions of the problem in order to optimize the runtime.

References

[1] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, Mar. 2004. [Online]. Available: https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf

[2] "Lecture 3: Svm dual, kernels and regression by a. zisserman," https://www.robots.ox.ac.uk/~az/lectures/ml/lect3.pdf, accessed: 30.01.2022.

[3] "A tutorial for support vector machine by wei-lun chao," http://disp.ee.ntu.edu.tw/~pujols/Support%20Vector%20Machine.pdf, accessed: 30.01.2022.

[4] "Cs229 lecture notes by tengyu ma and andrew ng," http://cs229.stanford.edu/notes2020fall/notes2020fall/cs229-notes3.pdf, accessed: 30.01.2022.

[5] "A tutorial on support vector machines for pattern recognition by christopher j.c. burges," https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/svmtutorial.pdf, accessed: 30.01.2022.

[6] "Support vector machines for beginners – training algorithms by abhisek jana," http://www.adeveloperdiary.com/data-science/machine-learning/support-vector-machines-for-beginners-training-algorithms/#training-dual-svm-using-kernel, accessed: 30.01.2022.

Table 1: All experiments where averaged over 100 runs

Problem	**Samples** *n_samples*	**features** *n_features*	**informative features** *n_informative*	**clusters per class** *n_clusters_per_class*	**average accuracy**	**average time to fit** in seconds	**variance**
Primal	100	2	2	1	0.9354	0.0002502	0.0004334
Dual	100	2	2	1	0.9456	0.0003003	0.0004587
Primal	1000	2	2	1	0.9471	0.0006105	0.0004882
Dual	1000	2	2	1	0.9446	0.0025823	0.0015387
Primal	10000	2	2	1	0.9456	0.0036432	0.0004804
Dual	10000	2	2	1	0.9502	0.0334867	0.0208917
Primal	100	400	20	2	1.0000	0.0498167	0.0142599
Dual	100	400	20	2	1.0000	0.0017543	0.0005143
Primal	200	400	20	2	1.0000	0.1926680	0.0380189
Dual	200	400	20	2	1.0000	0.0040832	0.0006742
Primal	300	400	20	2	1.0000	0.3278076	0.0628115
Dual	300	400	20	2	1.0000	0.0075969	0.0013282
Primal	500	400	20	2	1.0000	0.4866396	0.1065668
Dual	500	400	20	2	1.0000	0.0221303	0.0085774
Primal	100	3000	20	1	1.0000	0.1208909	0.0235087
Dual	100	3000	20	1	1.0000	0.0067761	0.0006307
Primal	500	3000	20	1	1.0000	0.5706844	0.4220353
Dual	500	3000	20	1	1.0000	0.0517344	0.0029324
Primal	1000	3000	20	1	1.0000	0.6942377	0.8116526
Dual	1000	3000	20	1	1.0000	0.1173785	0.0083099
Primal	2000	3000	20	1	1.0000	1.1702216	1.5569004
Dual	2000	3000	20	1	1.0000	0.2846513	0.0397023