1 Duality revisited

In this section, we will give a slightly different perspective on duality. Consider a constrained optimization program:

$$\begin{aligned} &\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \\ &\text{s.t } \mathbf{x} \in K \end{aligned}$$

where K is the feasible set of the program. Then it is clear that this problem is equivalent to:

$$\min_{\mathbf{x}\in\mathbb{R}^n} g(\mathbf{x})$$

where g is a function such that:

$$g(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{if } x \in K \\ +\infty & \text{otherwise} \end{cases} \tag{1}$$

Indeed, when $\mathbf{x} \notin K$, $g(\mathbf{x}) = +\infty$ and the minimum of g will not be attained at this point. When $\mathbf{x} \in K$, then f and g coincide and have the same minimum. So buy "encoding" the feasible set K into the function g, we transformed a constrained program into an unconstrained one.

More specifically, consider the convex optimization program:

$$\begin{aligned} &\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \\ &\text{s.t } g(\mathbf{x}) \leq a \\ &\quad\ h(\mathbf{x}) = b \end{aligned} \tag{2}$$

where f, g and h are convex. The following discussion can easily be adapted to an arbitrary number of constraints. Let us denote by K the (convex) feasible set of this program, then the function:

$$g(\mathbf{x}) = \max_{\lambda\geq 0,\, \mu\in\mathbb{R}} f(\mathbf{x}) + \lambda\big(g(\mathbf{x}) - a\big) + \mu\big(h(\mathbf{x}) - b\big)$$

satisfies the condition of Equation (1). Indeed, consider the case where $g(\mathbf{x}) > a$, then by taking $\lambda \to \infty$ (λ arbitrarily large), we see that $g(\mathbf{x}) = +\infty$. Similarly, if $h(\mathbf{x}) \neq b$, then either $h(\mathbf{x}) > b$, in which case we obtain $g(\mathbf{x}) = +\infty$ by taking $\mu \to \infty$, or $h(\mathbf{x}) < b$ and we obtain the same conclusion by taking $\mu \to -\infty$.

Let us denote by $\mathcal{L}(\mathbf{x}, \lambda, \mu)$ the function:

$$\mathcal{L}(\mathbf{x}, \lambda, \mu) = f(\mathbf{x}) + \lambda\big(g(\mathbf{x}) - a\big) + \mu\big(h(\mathbf{x}) - b\big)$$

Then we have just shown that the primal problem (2) is equivalent to:

$$\min_{\mathbf{x}\in\mathbb{R}^n} \max_{\lambda\geq 0,\, \mu\in\mathbb{R}} \mathcal{L}(\mathbf{x}, \lambda, \mu) \tag{3}$$

But you should remember from class, that we defined the Lagrgange dual function as:

$$g(\lambda, \mu) = \min_{\mathbf{x} \in \mathbb{R}^n} \mathcal{L}(\mathbf{x}, \lambda, \mu)$$

and the dual problem as the following maximization of the Lagrange dual function:

$$\max_{\lambda \geq 0,\, \mu \in \mathbb{R}} g(\lambda, \mu) = \max_{\lambda \geq 0,\, \mu \in \mathbb{R}} \min_{\mathbf{x} \in \mathbb{R}^n} \mathcal{L}(\mathbf{x}, \lambda, \mu)$$

Observe the nice symmetry with Equation 3 where the only difference is the order of the max and min operations: *the primal and dual problems are both optimizing the Lagrangian of the problem in different orders.*

The weak duality theorem can be re-interpreted in this context as the following inequality:

$$\max_{\lambda \geq 0,\, \mu \in \mathbb{R}} \min_{\mathbf{x} \in \mathbb{R}^n} \mathcal{L}(\mathbf{x}, \lambda, \mu) \leq \min_{\mathbf{x} \in \mathbb{R}^n} \max_{\lambda \geq 0,\, \mu \in \mathbb{R}} \mathcal{L}(\mathbf{x}, \lambda, \mu)$$

This inequality is true without any assumption on $\mathcal{L}$ (hence on f, g and h). The strong duality theorem states that the inequality is in fact an equality:

$$\max_{\lambda \geq 0,\, \mu \in \mathbb{R}} \min_{\mathbf{x} \in \mathbb{R}^n} \mathcal{L}(\mathbf{x}, \lambda, \mu) = \min_{\mathbf{x} \in \mathbb{R}^n} \max_{\lambda \geq 0,\, \mu \in \mathbb{R}} \mathcal{L}(\mathbf{x}, \lambda, \mu)$$

In other words, the order in which the parameters are optimized over does not matter (swapping the min and max operators is allowed). So Slater's condition seen in class can be interpreted as a sufficient condition under which the swapping of min and max operators is valid.

Remark. The Lagrange dual function g being (pointwise) a minimum of affine functions is a concave function. Hence the dual problem of a convex minimization problem is a concave maximization problem.

Remark. The interpretation of the strong duality theorem as swapping a min and max operator is reminiscent of Von Neumann's minimax theorem that we used to define the value of two-players, zero-sum games. In fact, the strong duality theorem under Slater's condition can be seen as a generalization of Von Neumann's minimax theorem.

2 Taking duals

Let us consider the following convex optimization program:

$$\begin{aligned} \min_{\mathbf{x} \in \mathbb{R}^n} \; & -\sum_{i=1}^{d} \log(b_i - \mathbf{a}_i^\intercal \mathbf{x}) \\ \text{s.t } & \mathbf{a}_i^\intercal \mathbf{x} - b_i \leq 0, \; 1 \leq i \leq d \end{aligned}$$

The motivation is to find a feasible point $\mathbf{x}$ of the set of linear inequalities $\mathbf{a}_i^\intercal \mathbf{x} - b_i$ (defining a polytope) which is far from the boundary of the polytope. Note that as $\mathbf{a}_i^\intercal \mathbf{x}$ gets close to b_i, the value of the objective function converges to $+\infty$.

The goal of this section is to derive the dual of the above problem. Simplifying the primal is a good thing to do before computing the dual. Here, we can introduce extra variables $y_i = b_i - \mathbf{a}_i^\intercal \mathbf{x}$, such

that the original problem can be rewritten:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n,\,\mathbf{y}\in\mathbb{R}^d} \quad & -\sum_{i=1}^d \log(y_i) \\ \text{s.t } & y_i \geq 0,\ 1 \leq i \leq d \\ & y_i = b_i - \mathbf{a}_i^\intercal \mathbf{x},\ 1 \leq i \leq d \end{aligned}$$

the advantage is that the arguments of the logs are now simpler. We then rewrite the problem in "minmax" form by introduction a dual variable λ_i for each inequality constraint, and a dual variable μ_i for each equality constraint:

$$\min_{\mathbf{x}\in\mathbb{R}^n,\,\mathbf{y}\in\mathbb{R}^d} \max_{\lambda\geq 0,\,\mu\in\mathbb{R}^d} -\sum_{i=1}^d \log(y_i) - \sum_{i=1}^d \lambda_i y_i + \sum_{i=1}^d \mu_i(\mathbf{a}_i^\intercal \mathbf{x} - b_i + y_i)$$

The dual problem is obtained by swapping the max and the min:

$$\max_{\lambda\geq 0,\,\mu\in\mathbb{R}^d} \min_{\mathbf{x}\in\mathbb{R}^n,\,\mathbf{y}\in\mathbb{R}^d} -\sum_{i=1}^d \log(y_i) - \sum_{i=1}^d \lambda_i y_i + \sum_{i=1}^d \mu_i(\mathbf{a}_i^\intercal \mathbf{x} - b_i + y_i)$$

We now need to understand the Lagrange dual function:

$$g(\lambda,\mu) = \min_{\mathbf{x}\in\mathbb{R}^n,\,\mathbf{y}\in\mathbb{R}^d} -\sum_{i=1}^d \log(y_i) - \sum_{i=1}^d \lambda_i y_i + \sum_{i=1}^d \mu_i(\mathbf{a}_i^\intercal \mathbf{x} - b_i + y_i)$$

First thing to note is that if $\sum_{i=1}^d \mu_i \mathbf{a}_i \neq 0$, then the minimum is $-\infty$ (we can always make x_j arbitrarily large or small for coordinates j such that $\sum_{i=1}^d \mu_i \mathbf{a}_{ij} \neq 0$). When $\sum_{i=1}^d \mu_i \mathbf{a}_i = 0$, then the function no longer depends on $\mathbf{x}$ and we are left with optimizing over $\mathbf{y} \in \mathbb{R}^d$. Singular points are given by:

$$\frac{1}{y_i} = \mu_i - \lambda_i,\ 1 \leq i \leq d$$

In summary, we have:

$$g(\lambda,\mu) = \begin{cases} \sum_{i=1}^d \log(\mu_i - \lambda_i) - \mu^\intercal \mathbf{b} + n & \text{if } \sum_{i=1}^d \mu_i \mathbf{a}_i = 0 \\ -\infty & \text{otherwise} \end{cases}$$

And the dual problem can be rewritten as:

$$\begin{aligned} \max_{\lambda\geq 0,\,\mu\in\mathbb{R}^d} \quad & \sum_{i=1}^d \log(\mu_i - \lambda_i) - \mu^\intercal \mathbf{b} + n \\ \text{s.t } & \sum_{i=1}^d \mu_i \mathbf{a}_i = 0 \end{aligned}$$

Finally, we observe that the objective function is decreasing in λ_i, so we should always take λ_i as small as possible, *i.e.* equal to zero. The dual thus simplifies:

$$\begin{aligned} \max_{\mu\in\mathbb{R}^d} \quad & \sum_{i=1}^d \log \mu_i - \mu^\intercal \mathbf{b} + n \\ \text{s.t } & \sum_{i=1}^d \mu_i \mathbf{a}_i = 0 \end{aligned}$$

3 Stochastic gradient descent

The stochastic gradient descent algorithm is a variant of the gradient descent algorithm where the gradient of the function $\nabla f(\mathbf{x}_t)$ at iteration t is replaced by a random variable g_t such that:

$$\mathbb{E}[g_t] = \nabla f(\mathbf{x}_t) \quad \text{and} \quad \mathbb{E}[\|g_t\|^2] \leq G^2$$

In other words, g_t is equal to the true gradient in expectation and has bounded expected norm. The algorithm is described in Algorithm 1.

Algorithm 1 Stochastic gradient descent

Require: $\mathbf{x}_1$

1: **for** $t = 1$ to T **do**
2: $\quad \mathbf{x}_{t+1} \leftarrow \mathbf{x}_t - \eta_t g_t$
3: **end for**
4: **return** $\bar{\mathbf{x}}_T = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_t$

Note that as opposed to standard gradient descent, the gradient is now stochastic and the algorithm is no longer strictly descending (the value of the current solution is not necessarily decreasing). This explains the last line of the algorithm where instead of returning the solution at the lest step, we return the average of all the solutions seen so far to "smooth out" the noise.

Theorem 1. *With step size $\eta_t = \frac{D}{G\sqrt{t}}$, the solution $\bar{\mathbf{x}}_T$ returned by Algorithm 1 is such that:*

$$\mathbb{E}[f(\bar{\mathbf{x}}_T)] - \min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) \leq \frac{3GD}{2\sqrt{T}}$$

Proof. The proof I didn't finish in section was taken from the excellent book *Online Convex Optimization* by Elad Hazan whose draft is available at http://ocobook.cs.princeton.edu/. What is interesting is that the proof is a direct application of the theorem we saw in class given regret bounds for gradient descent in online convex optimization. □

Application to machine learning. At first glance, stochastic gradient descent might not seem very useful: how to find a g_t equal to the true gradient in expectation? is it much simpler than computing the true gradient? It turns out that optimization problems coming from machine learning are very amenable to stochastic gradient descent.

A standard problem in machine learning is the following: given a data set $\{(x_i, y_i),\ 1 \leq i \leq n\}$, the goal is to find a model f_w parametrized by w which fits the data well. The fit of the model at data point i is captured by a loss function $\ell(f_w(x_i), y_i)$. The optimization problem thus takes the following form:

$$\min_w \frac{1}{n}\sum_{i=1}^{n} \ell(f_w(x_i), y_i)$$

The true gradient of the objective function is given by:

$$\frac{1}{n}\sum_{i=1}^{n} \nabla_w \ell(f_w(x_i), y_i) \tag{4}$$

and requires iterating over the n data points, which is prohibitive for very large datasets.

Instead, consider g_w computed by the following algorithm:

Algorithm 2 Stochastic gradient

Require: w

1: $i \leftarrow$ pick uniformly at random between 1 and n

2: **return** $g_w = \nabla_w \ell(f_w(x_i), y_i)$

g_w is now a random vector, each i is chosen with probability $\frac{1}{n}$, hence the expected value of g_w is given by:

$$\mathbb{E}[g_w] = \frac{1}{n} \sum_{i=1}^{n} \nabla_w \ell(f_w(x_i), y_i)$$

which is exactly the gradient computed in Equation 4. So g_w can be used as a "stochastic" gradient in the stochastic gradient algorithm. Note that now, each iteration of the gradient descent only requires accessing one data point of the dataset, which induces huge performance gains!