

Duality, Gradient-free, in Application

Axel Böhm

January 24, 2022

- 1 Duality
- 2 Derivative free
- 3 Methods in practice

Duality

Establishes some relation between two classes of objects.

Definition (“Legendre transform” or “Fenchel conjugate”)

Given a function $f : \mathbb{R}^d \rightarrow \mathbb{R} \cup \{+\infty\}$, we define its **conjugate** $f^* : \mathbb{R}^d \rightarrow \mathbb{R} \cup \{+\infty\}$ by

$$f^*(y) = \sup_x \{\langle y, x \rangle - f(x)\}$$

Convex conjugate

$$f^*(y) = \sup_x \{\langle y, x \rangle - f(x)\}$$

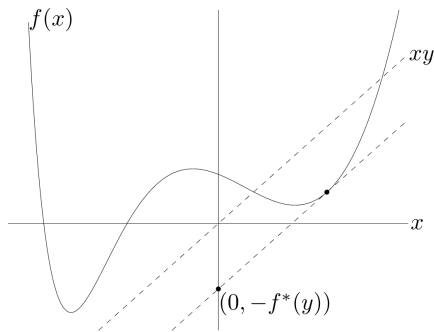


Figure: maximum gap between linear function $x \mapsto \langle y, x \rangle$ and f

Properties

- ◇ f^* is always convex. point-wise max of affine function
- ◇ Fenchel's inequality:

$$f(x) + f^*(y) \geq \langle x, y \rangle.$$

- ◇ Hence the biconjugate $f^{**} := (f^*)^*$ satisfies $f^{**} \leq f$.
- ◇ If f is convex and lsc. then $f^{**} = f$.
- ◇ etc.

Examples

◇ Norm: If $f(x) = \|x\|$, then

$$f^*(y) = \mathbb{1}(\|y\|_* \leq 1),$$

i.e. the indicator of the dual norm ball. Recall the definition of the dual norm:

$$\|y\|_* := \max_{\|x\| \leq 1} \{\langle y, x \rangle\}.$$

In particular: $\|\cdot\|_1 \leftrightarrow \|\cdot\|_\infty$

More examples

Generalized linear models: $\min_{x \in \mathbb{R}^d} f(Ax) + g(x).$

Two approaches to reformulate:

$$\min_x \max_y \underbrace{\langle y, Ax \rangle - f^*(y)}_{=f(Ax)} + g(x).$$

Switch min and max

$$\max_y \min_x -f^*(y) + \langle y, Ax \rangle + g(x).$$

Change sign to go from max to min

$$\max_y -f^*(y) - \max_x \underbrace{-\langle y, Ax \rangle - g(x)}_{=g^*(-A^T y)}.$$

Generalized linear models continued

Or reformulate

$$\min_{x \in \mathbb{R}^d} f(Ax) + g(x)$$

as

$$\min_{x \in \mathbb{R}^d, w \in \mathbb{R}^m} f(w) + g(x) \quad \text{s.t. } w = Ax$$

Use Lagrange function

$$\mathcal{L}(x, w, u) := f(w) + g(x) + \langle u, w - Ax \rangle,$$

then the dual function is given by

$$\varphi(u) = \min_{x \in \mathbb{R}^d, w \in \mathbb{R}^m} \mathcal{L}(x, w, u).$$

Dual problem

$$\max_{u \in \mathbb{R}^m} [\varphi(u) = -f^*(-u) - g^*(A^T u)].$$

Example: Lasso

ℓ_1 regularized regression

$$\min_{x \in \mathbb{R}^d} \frac{1}{2} \|Ax - b\|^2 + \lambda \|x\|_1$$

fits this template with

$$f(w) = \frac{1}{2} \|w - b\|^2 \quad \text{and} \quad g(x) = \lambda \|x\|_1$$

Computation gives

$$f^*(u) = \frac{1}{2} \|b\|^2 - \frac{1}{2} \|b - u\|^2 \quad \text{and} \quad g^*(v) = \mathbb{1}(\|v/\lambda\|_\infty \leq 1).$$

Dual of Lasso

We had

$$f^*(u) = \frac{1}{2}\|b\|^2 - \frac{1}{2}\|b - u\|^2 \quad \text{and} \quad g^*(v) = \mathbb{1}(\|v/\lambda\|_\infty \leq 1).$$

So the dual is

$$\begin{aligned} & \max_{u \in \mathbb{R}^m} -f^*(-u) - g^*(A^T u) \\ & \Leftrightarrow \min_{u \in \mathbb{R}^m} \|b + u\|^2 \quad \text{s.t.} \quad \|A^T u\|_\infty \leq \lambda \end{aligned}$$

Similarly, for least squares, ridge/logistic regression, SVM, etc.

But why?

- ◇ Duality gap gives a **certificate** of current optimization quality

$$\begin{aligned} f(A\bar{x}) + g(\bar{x}) &\geq \min_{x \in \mathbb{R}^d} f(Ax) + g(x) \\ &\geq \max_{u \in \mathbb{R}^m} -f^*(-u) - g^*(A^T u) \\ &\geq -f^*(-\bar{u}) - g^*(A^T \bar{u}) \end{aligned}$$

- ◇ Stopping criterion
- ◇ Dual problem is sometimes easier to solve

Zero-order Optimization

\Leftrightarrow Derivative-Free

\Leftrightarrow Blackbox

Look, no gradients!

Can we solve $\min_{x \in \mathbb{R}^d} f(x)$ without access to gradients?

Algorithm Random search

- 1: **for** $k = 1, 2, \dots$ **do**
 - 2: **for** pick a random direction $d_k \in \mathbb{R}^d$ **do**
 - 3: $\gamma_k := \arg \min_{\gamma \in \mathbb{R}} f(x_k + \gamma d_k)$
 - 4: $x_{k+1} := x_k + \gamma_k d_k$
-

Convergence rate for derivative-free random search

Converges same as gradient descent - up to a slow-down factor d .

Proof.

$$f(x_k + \gamma d_k) \leq f(x_k) + \gamma \langle d_k, \nabla f(x_k) \rangle + \frac{\gamma^2 L}{2} \|d_k\|^2$$

Minimizing the upper bound (RHS), there exists a step $\bar{\gamma}$ for which

$$f(x_k + \bar{\gamma} d_k) \leq f(x_k) - \frac{1}{L} \left\langle \frac{d_k}{\|d_k\|^2}, \nabla f(x_k) \right\rangle.$$

So our (exact line-search) step can only be better

$$f(x_k + \gamma_k d_k) \leq f(x_k + \bar{\gamma} d_k)$$

Taking expectation and using $\mathbb{E}_r \langle r, g \rangle^2 = 1/d \|g\|^2$ for $r \sim \text{sphere}$, gives

$$\mathbb{E}[f(x_k + \gamma_k d_k)] \leq E[f(x_k)] - \frac{1}{Ld} \mathbb{E} [\|\nabla f(x_k)\|^2].$$

Convergence rate for derivative-free random search

Same as what we obtained for **gradient descent**, now with an **extra factor of d** . d can be huge!!!

Similarly for other function classes

- ◇ For convex functions, we get a rate of $\mathcal{O}(\frac{dL}{\epsilon})$.
- ◇ For μ -strongly convex functions, we get a rate of $\mathcal{O}(d\kappa \log(1/\epsilon))$.

Always d times the complexity of gradient descent on the function class.

But assumed differentiability. Can also approximate the gradient.

Applications for derivative-free random search

Applications

- ◇ competitive method for reinforcement learning
- ◇ No need to store a gradient
- ◇ hyperparameter optimization, and other difficult e.g. discrete optimization problems, black-box, noisy

Reinforcement learning

$$s_{k+1} = f(s_k, a_k, e_k),$$

where s_k is the **state** of the system, a_k is the control **action**, and e_k is some random **noise**. We assume existence of f , but it is unknown.

We search for a “policy”

$$a_k := \pi(a_1, \dots, a_{k-1}, s_0, \dots, s_k)$$

Goal: Maximize reward

$$\begin{aligned} \max_{a_k} \mathbb{E}_{e_k} \left[\sum_{k=1}^N R_k(s_k, a_k) \right] \\ \text{s.t. } s_{k+1} = f(s_k, a_k, e_k) \end{aligned}$$

Adaptive & other SGD methods

Adagrad

An adaptive variant of SGD

Algorithm Adagrad

- 1: **for** $k = 1, \dots$ **do**
 - 2: pick stochastic gradient g_k
 - 3: $[G_k]_i = \sum_{l=1}^k ([g_l]_i)^2, \quad \forall i$
 - 4: $[x_{k+1}]_i = [x_k]_i - \frac{\gamma}{\sqrt{[G_k]_i}} [g_k]_i, \quad \forall i$
-

- ◇ chooses an **adaptive, coordinate-wise** learning rate
- ◇ strong performance in practice
- ◇ many variants: Adadelata, Adam, RMSprop

Adam

A **momentum** variant of **Adagrad**

Algorithm Adam

- 1: **for** $k = 1, \dots$ **do**
 - 2: pick stochastic gradient g_k
 - 3: $m_k = \beta_1 m_{k-1} + (1 - \beta_1) g_k$ momentum
 - 4: $[v_k]_i = \beta_2 [v_{k-1}]_i + (1 - \beta_2) ([g_k]_i)^2$ 2nd-order statistics
 - 5: $[x_{k+1}]_i = [x_k]_i - \frac{\gamma}{\sqrt{[v_k]_i}} [m_k]_i, \quad \forall i$
-

- ◇ forgetting of old weights
- ◇ momentum (see lecture on acceleration)
- ◇ strong performance for certain DL problems

SignSGD

Only use the sign (one bit) of each gradient entry:
Yields a **communication efficient** variant of SGD.

Algorithm SignSGD

- 1: **for** $k = 1, \dots$ **do**
 - 2: pick stochastic gradient g_k
 - 3: $[x_{k+1}]_i = [x_k]_i - \gamma_k \text{sign}([m_k]_i), \quad \forall i$
-

(possible rescaling of γ_k by $\|g_k\|_1$)

- ◇ combats communication bottleneck for distributed training
- ◇ convergence issues