Introduction to Optimization

16th April 2018

Adapted from slides provided by Prof. Jihun Hamm.

Outline

What is optimization?

- Finding (one or more) minimizer of a function subject to constraints

$$
\begin{aligned}
\underset{x}{\operatorname{argmin}} \quad & f_0(x) \\
\text{s.t.} \quad & f_i(x) \leq 0, i = \{1, ..., k\} \\
& h_j(x) = 0, j = \{1, ..., l\}
\end{aligned}
\tag{1}
$$

What is optimization?

- Finding (one or more) minimizer of a function subject to constraints

$$
\begin{aligned}
\underset{x}{\operatorname{argmin}} \quad & f_0(x) \\
\text{s.t.} \quad & f_i(x) \leq 0, i = \{1, ..., k\} \\
& h_j(x) = 0, j = \{1, ..., l\}
\end{aligned}
\tag{1}
$$

- Most of the machine learning problems are, in the end, optimization problems.

Examples

- (Soft) Linear SVM

$$\begin{aligned} \underset{w}{\operatorname{argmin}} \quad & \sum_{i=1}^{n} \|w\|^2 + C \sum_{i=1} n\epsilon_i \\ \text{s.t.} \quad & 1 - y_i x_i^T w \leq \epsilon_i \\ & \epsilon_i \geq 0 \end{aligned} \tag{2}$$

- Maximum Likelihood

$$\underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log p_\theta(x_i) \tag{3}$$

- K-means

$$\underset{\mu_1, \mu_2, \ldots, \mu_k}{\operatorname{argmin}} J(\mu) = \sum_{j=1}^{k} \sum_{i \in C_j} \|x_i - \mu_j\|^2 \tag{4}$$

Outline

Convex sets

Definition

A set $C \subseteq \mathbb{R}^n$ is convex if for $x, y \in C$ and any $\alpha \in [0,1]$, $\alpha x + (1-\alpha)y \in C$.

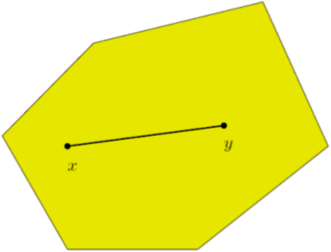


Figure: Convex Set

Convex sets

Example

- All of $\mathbb{R}^n$
- Non-negative orthant, $\mathbb{R}^n_+$: let $x \geq 0, y \geq 0$, clearly $\alpha x + (1-\alpha)y \geq 0$.
- Affine subspaces: $Ax = b, Ay = b$, then

$$A(\alpha x + (1-\alpha)y) = \alpha Ax + (1-\alpha)Ay = b$$

.

- Arbitrary intersections of convex sets: let C_i be convex for $i \in \mathcal{I}, C = \bigcap_i C_i$, then

$$x \in C, y \in C \Rightarrow \alpha x + (1 - \alpha y) \in C_i \subseteq C, \forall i \in \mathcal{I}$$

.

Convex functions

Definition

A function $f : \mathbb{R}^n \to \mathbb{R}$ *is convex if for* $x, y \in$ *dom* f *and any* $a \in [0, 1]$,

$$f(ax + (1-a)y) \leq af(x) + (1-a)f(y)$$

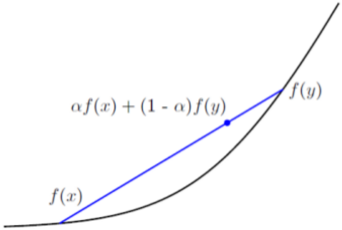


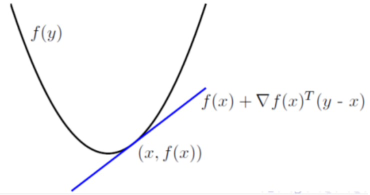
Figure: Convex Function

Convexity condition 1

Theorem

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable. Then f is convex if and only if for all $x, y \in$ dom f.

$$f(y) \geq f(x) + \nabla f(x)^T (y - x)$$



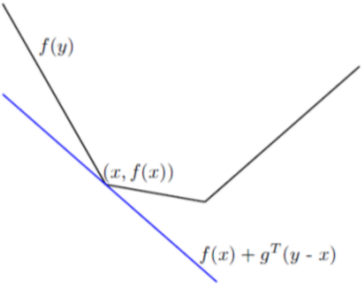
Subgradient

Definition

The subgradient set, or subdifferential set, $\partial f(x)$ of f at x is

$$\partial f(x) = \{g : f(y) \geq f(x) + g^T(y - x) \quad \forall y\}$$

.



Theorem

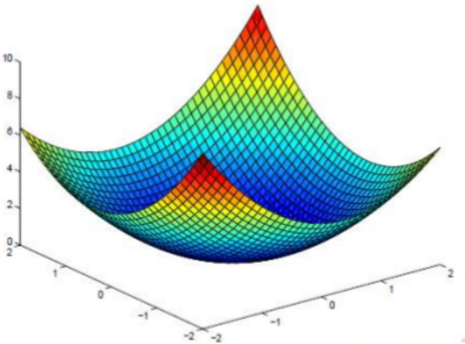
$f : \mathbb{R}^n \to \mathbb{R}$ is convex iff it has ono-empty subdifferential set everywhere.

Convexity condition 2

Theorem

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable, Then f is convex iff for all $x \in$ dom f,

$$\nabla^2 f(x) \succeq 0.$$



Examples of convex functions

- Linear/affine functions: $f(x) = b^T x + c$
- Quadratic function: $f(x) = \frac{1}{2} x^T A x + b^T x + c$, for $A \succeq 0$. *e.g.*, for regression:

$$\frac{1}{2}\|\mathbf{X}w - y\|^2 = \frac{1}{2} w^T \mathbf{X}^T \mathbf{X} w - y^T \mathbf{X} w + \frac{1}{2} y^T y$$

.

Examples of convex functions

- Norms (like l_l or l_2 for regularization):

$$\|ax + (1-a)y\| \leq \|ax\| + \|(1-a)y\| = a\|x\| + (1-a)\|y\|$$

.

- Composition with an affine function $f(Ax + b)$:

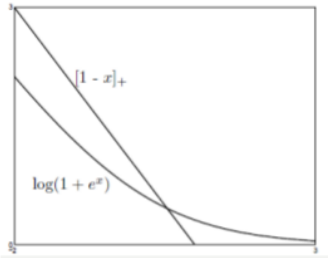
$$\begin{aligned} f(A(ax + (1-a)y) + b) &= f(a(Ax + b) + (1-a)(Ay + b)) \\ &\leq af(Ax + b) + (1-a)f(Ay + b) \end{aligned}$$

- Log-sum-exp (via $\nabla^2 f(x)$ PSD):

$$f(x) = \log\left(\sum_{i=1}^{n} \exp(x_i)\right)$$

Examples in machine learning

- SVM loss: $f(w) = [1 - y_i x_i^T w]_+$
- Binary logistic loss: $f(w) = \log(1 + \exp(-y_i x_i^T w))$



Convex optimization

Definition

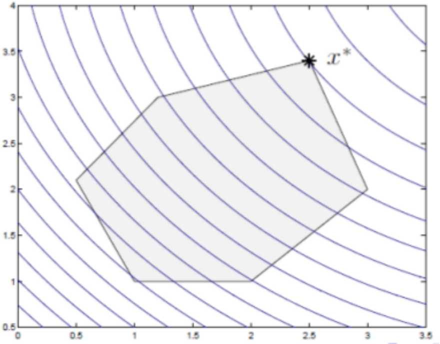
An optimization problem is convex if its objective is a convex function. The inequality constrains f_j are convex, and the equality constraints h_j are affine.

$$
\begin{aligned}
\min_{x} \quad & f_0(x) && \text{(Convex function)} \\
\text{s.t.} \quad & f_i(x) \leq 0, i = \{1, ..., k\} && \text{(Convex sets)} \\
& h_j(x) = 0, j = \{1, ..., l\} && \text{(Affine)}
\end{aligned}
\tag{5}
$$

Convex Problems are nice ...

Theorem

If $\hat{x}$ is a local minimizer of a convex optimization problem, it is a global minimizer.



For smooth functions

Theorem

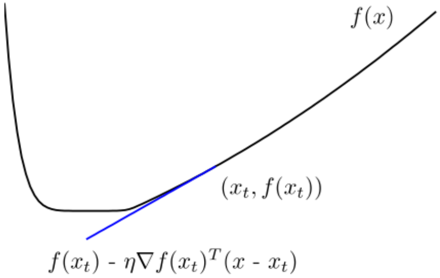
- $\nabla f(x) = 0$. *We have* $f(y) \geq f(x) + \nabla f(x)^T(y - x) = f(x)$.
- $\nabla f(x) \neq 0$. *There is a direction of descent.*

Outline

Gradient descent

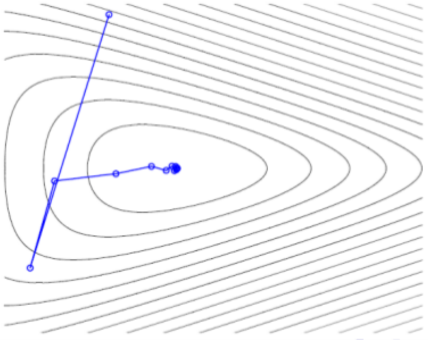
- Consider convex and unconstrained optimization.
- Solve $\min_x f(x)$.
- One of the simplest approach:
  - For $t = 1, ..., T$, $x_{t+1} \leftarrow x_t - \eta_t \nabla f(x_t)$
  - Until convergence
  - η_t is called step-size of learning rate.

Single step in gradient descent



Full gradient descent

$$f(x) = \log(\exp(x_1 + 3x_2 - .1) + \exp(x_1 - 3x_2 - .1) + \exp(-x_1 - .1))$$



How to choose step size?

- Idea 1: exact line search

$$\eta_t = \operatorname*{argmin}_{\eta} f(x - \eta \nabla f(x))$$

Too expensive to be practical.

- Idea 2: backtracking (Armijo) line search. Let $\alpha \in (0, 1/2), \beta \in (0, 1)$. Multiply $\eta = \beta\eta$ until

$$f(x - \eta \nabla f(x)) \leq f(x) - \alpha\eta \|\nabla f(x)\|^2$$

Works well in practice.

Newton's method

Idea: use a second-order approximation to function.

$$f(x + \Delta x) \approx f(x) + \nabla f(x)^T \Delta x + 1/2 \Delta x^T \nabla^2 f(x) \Delta x$$

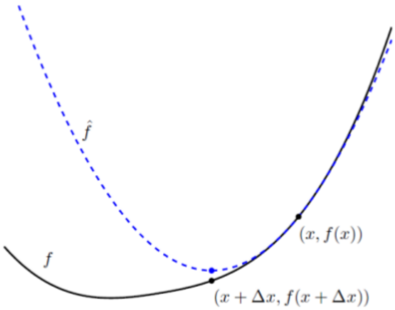
Choose Δx to minimize above:

$$\Delta x = -[\nabla^2 f(x)]^{-1} \nabla f(x)$$

This is descent direction:

$$\nabla f(x)^T \Delta x = -\nabla f(x)^T [\nabla^2 f(x)]^{-1} \nabla f(x) \leq 0$$

Single step in Newton's method



$\hat{f}$ is 2[nd]-order approximation, f is true function.

Convergence rate

- Strongly convex case: $\nabla^2 f(x) \succeq mI$, then "Linear convergence". For some $\gamma \in (0,1), f(x_t) - f(x^*) \leq \gamma^t, \gamma \leq 1$.

$$f(x_t) - f(x^*) \leq \gamma^t, t \geq \frac{1}{\gamma} \log \frac{1}{\epsilon} \Rightarrow f(x_t) - f(x^*) \leq \epsilon$$

  .

- Smooth case: $\|\nabla f(x) - \nabla f(y)\| \leq C\|x - y\|$.

$$f(x_t) - f(x^*) \leq \frac{K}{t^2}$$

- Newton's method often is faster, especially when f has "long valleys".

Newton's method

- Inverting a Hessian is very expensive: $O(d^3)$
- Approximate inverse Hessian: BFGS, Limited-memory BFGS
- Or use Conjugate Gradient Descent.
- For unconstrained problems, you can use these off-the-shelf optimization methods
- For unconstrained non-convex problems, these methods will find local optima

Optimization for machine learning

- Goal of machine learning
    - Minimize expected loss $L(h) = \mathbf{E}[\mathrm{loss}(h(x), y)]$ given samples $(x_i, y_i), i = 1, 2, ..., m$
    - But we don't know $P(x, y)$, nor can we estimate it well.
- Empirical risk minimization
    - Substitute sample mean for expectation.
    - Minimize empirical loss: $L(h) = 1/n \sum_i \mathrm{loss}(h(x_i), y_i)$
    - *a.k.a.* Sample Average Approximation.

Batch gradient descent

Minimize empirical loss, assuming it's convex and unconstrained

- Gradient descent on the empirical loss:
- At each step,

$$w^{k+1} \leftarrow w^k - \eta_t \left(\frac{1}{n} \sum_{i=1}^{n} \frac{\partial L(w, x_i, y_i)}{\partial w} \right)$$

- Note: at each step, gradient is the average of the gradient for all samples $(i = 1, ..., n)$.
- Very slow when n is very large.

Stochastic Gradient Descent

- Alternative: compute gradient from just one (or a few samples)
- Known as SGD: At each step,

$$w^{k+1} \leftarrow w^k - \eta_t \frac{\partial L(w, x_i, y_i)}{\partial w}$$

  (choose one sample i and compute gradient for that sample only)
- the gradient of one random sample is not the gradient of the objective function.
- Q1: Would this work at all?
- Q2: How good is it?

Stochastic Gradient Descent

- Alternative: compute gradient from just one (or a few samples)
- Known as SGD: At each step,

$$w^{k+1} \leftarrow w^k - \eta_t \frac{\partial L(w, x_i, y_i)}{\partial w}$$

  (choose one sample i and compute gradient for that sample only)
- the gradient of one random sample is not the gradient of the objective function.
- Q1: Would this work at all?
- Q2: How good is it?
- A1: SGD converges to not only thy empirical loss minimum, but also to the expected loss minimum!
- A2: Convergence (to expected loss) is slow: $f(w_t) - E[f(w^*)] \leq O(1/t)$ or $O(1/\sqrt{t})$

Practically speaking ...

- If the training set is small, we should use batch learning using quasi-Newton or conjugate gradient descent.
- If the training set is large, we should use SGD.
- If the size of training set is somewhere in between, we use mini-batch SGD.
- Convergence is very sensitive to learning rate, which needs to be determined by trial-and-error (model selection or cross-validation)

Outline

Lagrangian function

Start with optimization Problem:

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0, i = \{1, ..., k\} \\ & h_j(x) = 0, j = \{1, ..., l\} \end{aligned} \tag{6}$$

From *Lagrangian* using Lagrange multipliers $\lambda_i \geq 0, \nu_i \in \mathbb{R}$

$$\mathcal{L}(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{k} \lambda_i f_i(x) + \sum_{j=1}^{l} \nu_j h_j(x) \tag{7}$$

Lagrangian function

Original/primal problem:

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0, i = \{1, ..., k\} \\ & h_j(x) = 0, j = \{1, ..., l\} \end{aligned}$$

is equivalent to min-max optimization:

$$\min_x [\sup_{\lambda \succeq 0, \nu} \mathcal{L}(x, \lambda, \nu)]$$

Why?

Lagrangian function

Original/primal problem:

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0, i = \{1, ..., k\} \\ & h_j(x) = 0, j = \{1, ..., l\} \end{aligned}$$

is equivalent to min-max optimization:

$$\min_x [\sup_{\lambda \succeq 0, \nu} \mathcal{L}(x, \lambda, \nu)]$$

Why?

- consider a two-player game, if player 1 chooses x that violates a constraint $f_1(x) > 0$, player 2 chooses $\lambda_1 \to \infty$ so that $\mathcal{L}(x, \lambda, \nu) = ... + \lambda_1 f_1(x) + ... \to \infty$
- Therefore, player 1 is forced to satisfy constraints.

Dual function and dual problem

- Dual function:

$$g(\lambda, \nu) = \inf_x \mathcal{L}(x, \lambda, \nu)$$
$$= \inf_x \left\{ f_0(x) + \sum_{i=1}^{k} \lambda_i f_i(x) + \sum_{j=1}^{l} \nu_j h_j(x) \right\}$$

- Dual problem:

$$\max_{\lambda \succeq 0, \nu} [\inf_x \mathcal{L}(x, \lambda, \nu)]$$

- Primal problem:

$$\min_x [\sup_{\lambda \succeq 0, \nu} \mathcal{L}(x, \lambda, \nu)]$$

- Q: How are primal and dual solutions related?

Weak duality

Dual function lower-bounds the primal optimal value!

Lemma (Weak Duality)

If $\lambda \succeq 0$, then

$$g(\lambda, \nu) \leq f_0(x^*)$$

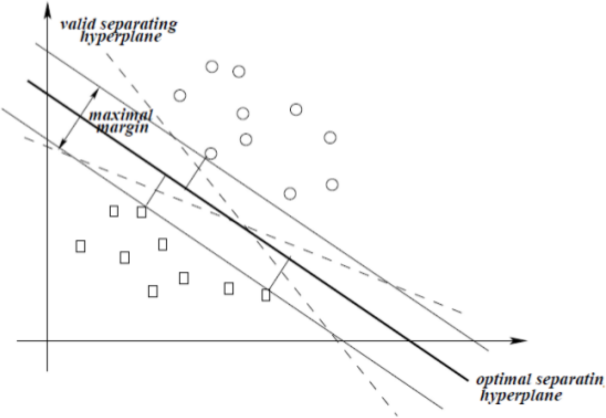
Proof.

$$\begin{aligned} g(\lambda, \nu) &= \inf_x \mathcal{L}(x, \lambda, \nu) \leq \mathcal{L}(x^*, \lambda, \nu) \\ &= f_0(x^*) + \sum_{i=1}^{k} \lambda_i f_i(x^*) + \sum_{j=1}^{l} \nu_j h_j(x^*) \leq f_0(x^*). \end{aligned}$$

Strong duality

- For convex problems, primal and dual solutions are equivalent! $\sup_{\lambda \succeq 0, \nu} g(\lambda, \nu) = f_0(x^*)$
- Equivalently, $\max \min \mathcal{L}(x, \lambda, \nu) = \min \max \mathcal{L}(x, \lambda, \nu)$
- What does the theorem mean in practice?
- When you have a primal constrained minimization problem, which may be hard to solve, you may solve the dual problem, which may be easier to solve (simpler constrains), it yields the same solution!

SVM Recap


valid separating
hyperplane
maximal
margin
optimal separatin
hyperplane

SVM in primal form

Primal SVM:

$$
\begin{aligned}
\min \quad & 1/2\|w\|^2 \\
\text{s.t.} \quad & y_i(wx_i + w_0) \geq 1 \text{ for } i = 1, \dots, m
\end{aligned}
$$

- for linearly separable cases.
- It is a linearly constrained QP, and therefore a convex problem.

SVM in dual form

The *Lagrangean function* associated to the primal form of the given QP is

$$L_P(w, w_0, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{m} \alpha_i(y_i(wx_i + w_0) - 1)$$

with $\alpha_i \geq 0, i = 1, ..., m$. Finding the minimum of L_P implies

$$\frac{\partial L_P}{\partial w_0} = -\sum_{i=1}^{m} y_i\alpha_i = 0$$

$$\frac{\partial L_P}{\partial w} = w - \sum_{i=1}^{m} y_i\alpha_i x_i = 0 \Rightarrow w = \sum_{i=1}^{m} y_i\alpha_i x_i$$

where $\frac{\partial L_P}{\partial w} = (\frac{\partial L_P}{\partial w_1}, ..., \frac{\partial L_P}{\partial w_d})$

By substituting these constraints into L_P we get its dual form
$L_D(\alpha) = \sum_{i=1}^{m} \alpha_i - 1/2 \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i\alpha_j y_i y_j x_i x_j$

Constrained optimization methods

- Log barrier method
- Projected (sub)gradient
- Interior point method
- Specialized methods
    - SVM: Sequential Minimal Optimization
    - Structured-output SVM: cutting-plane method
- Other optimization not covered in this lecture:
    - Bayesian models: EM, variational methods
    - Discreet optimization
    - Graph optimization

Thank you!