Optimization for Machine Learning (in a Nutshell)

Jihun Hamm
What is optimization?
  - Examples in machine learning
  - Unconstrained vs constrained

Convex optimization
  - Convex sets
  - Convex functions
  - Convex optimization
Contents (cont’d)

- Unconstrained optimization
  - Gradient descent
  - Newton’s method
  - Batch vs online learning
  - Stochastic Gradient Descent

- Constrained optimization
  - Lagrange duality
  - SVM in primal and dual forms
  - Constrained methods
What is optimization?

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

\[ \arg \min_x f_0(x) \]

s.t. \[ f_i(x) \leq 0, \ i = \{1, \ldots, k\} \]

\[ h_j(x) = 0, \ j = \{1, \ldots l\} \]

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

- **(Soft) Linear SVM**
  \[
  \arg\min_w \sum_{i=1}^{n} ||w||^2 + C \sum_{i=1}^{n} \xi_i \\
  \text{s.t. } 1 - y_i x_i^T w \leq \xi_i \\
  \xi_i \geq 0
  \]

- **Maximum Likelihood**
  \[
  \arg\max_\theta \sum_{i=1}^{n} \log p_\theta(x_i)
  \]

- **K-means**
  \[
  \arg\min_{\mu_1, \mu_2, \ldots, \mu_k} J(\mu) = \sum_{j=1}^{k} \sum_{i \in C_j} ||x_i - \mu_j||^2
  \]
Optimization is difficult in general

- Minimize $f(x)$
Contents

- What is optimization?
  - Examples in machine learning
  - Unconstrained vs constrained

- Convex optimization
  - Convex sets
  - Convex functions
  - Convex optimization
Convex sets

- **Def:** A set $C \subseteq \mathbb{R}^n$ is convex if for $x, y \in C$ and any $a \in [0, 1]$,\[ ax + (1 - a)y \in C \]
Examples of convex set

- All of $\mathbb{R}^n$ (obvious)

- 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

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

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

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

so $\alpha x + (1 - \alpha)y \in C$. 

Convex functions

Def:
A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if for $x, y \in \text{dom } f$ and any $a \in [0, 1],
\[ f(ax + (1-a)y) \leq af(x) + (1-a)f(y) \]
Suppose \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) is differentiable. Then \( f \) is convex if and only if for all \( x, y \in \text{dom } f \)

\[
f(y) \geq f(x) + \nabla f(x)^T (y - x)
\]
**Definition**

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

$$\partial f(x) = \{ g : f(y) \geq f(x) + g^T(y - x) \text{ for all } y \}.$$

**Theorem**

$f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if it has non-empty subdifferential set everywhere.
Theorem
Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is twice differentiable. Then $f$ is convex if and only if for all $x \in \text{dom } f$,

$$\nabla^2 f(x) \geq 0.$$
Examples of convex functions

- **Linear/affine functions:**
  
  \[ f(x) = b^T x + c. \]

- **Quadratic functions:**

  \[ f(x) = \frac{1}{2} x^T A x + b^T x + c \]

  for \( A \succeq 0 \). For regression:

  \[
  \frac{1}{2} \| X w - y \|^2 = \frac{1}{2} w^T X^T X w - y^T X w + \frac{1}{2} y^T y.
  \]
More examples

- Norms (like $\ell_1$ or $\ell_2$ for regularization):
  \[
  \|\alpha x + (1 - \alpha)y\| \leq \|\alpha x\| + \|(1 - \alpha)y\| = \alpha \|x\| + (1 - \alpha) \|y\|.
  \]

- Composition with an affine function $f(Ax + b)$:
  \[
  f(A(\alpha x + (1 - \alpha)y) + b) = f(\alpha(Ax + b) + (1 - \alpha)(Ay + b)) \\
  \leq \alpha f(Ax + b) + (1 - \alpha)f(Ay + b)
  \]

- 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

- Def:

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

\[
\underset{x}{\text{minimize}} \quad f_0(x) \quad \text{(Convex function)}
\]

\[
\text{s.t.} \quad f_i(x) \leq 0 \quad \text{(Convex sets)}
\]

\[
h_j(x) = 0 \quad \text{(Affine)}
\]
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**
\[ \nabla f(x) = 0 \text{ if and only if } x \text{ is a global minimizer of } f(x). \]

**Proof.**

- \( \nabla f(x) = 0 \). 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.
Contents (cont’d)

- Unconstrained optimization
  - Gradient descent
  - Newton’s method
  - Batch vs online learning
  - Stochastic Gradient Descent

- Constrained optimization
  - Lagrange duality
  - SVM in primal and dual forms
  - Constrained methods
Consider convex and unconstrained optimization.

Solve $\min_x f(x)$

One of the simplest approach:

- For $t = 1, \ldots, T$
  - $x_{t+1} \leftarrow x_t - \eta_t \nabla f(x_t)$
- Until convergence
- $\eta_t$ is called step-size or learning rate.
Single step in gradient descent

\[ f(x_t) - \eta \nabla f(x_t)^T (x - x_t) \]
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

\[ \eta_t = \underset{\eta}{\text{argmin}} \ f(x - \eta \nabla f(x)) \]

Too expensive to be practical.

- Idea 2: backtracking (Armijo) line search. Let \( \alpha \in (0, \frac{1}{2}) \), \( \beta \in (0, 1) \). 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.

\[ f(x + \Delta x) \approx f(x) + \nabla f(x)^T \Delta x + \frac{1}{2} \Delta x^T \nabla^2 f(x) \Delta x \]

Choose \( \Delta x \) to minimize above:

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

This is descent direction:

\[ \nabla f(x)^T \Delta x = -\nabla f(x)^T \left[ \nabla^2 f(x) \right]^{-1} \nabla f(x) < 0. \]
Single step in Newton’s method

\[
\hat{f} \text{ is 2}^{\text{nd}}\text{-order approximation, } f \text{ is true function.}
\]
Convergence rate

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

$$f(x_t) - f(x^*) \leq \gamma^t \quad \text{or} \quad t \geq \frac{1}{\gamma} \log \frac{1}{\varepsilon} \Rightarrow f(x_t) - f(x^*) \leq \varepsilon.$$

- 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 $L(h) = \mathbb{E}[\text{loss}(h(x), y)]$
  given samples $(x_i, y_i) \ i = 1, 2 \ldots m$
- 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: $L(h) = 1/n \sum_i \text{loss}(h(x_i), y_i)$
- A.K.A. Sample Average Approximation
Batch gradient descent

- Let’s put our knowledge into use
- 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 \((i = 1, \ldots, n)\)
  - Very slow when \(n\) is very large
Stochastic gradient descent

Alternative: compute gradient from just one (or a few samples)

Known as stochastic gradient descent:

- At each step,

\[ w^{(k+1)} \leftarrow w^{(k)} - \eta \frac{\partial L(w, x_i, y_i)}{\partial w} \]

(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 the empirical loss minimum, but also to the expected loss minimum!

A2: Convergence (to expected loss) is slow

\[ f(w_t) - E[f(w^*)] \leq O(1/t) \text{ or } O(1/\sqrt{t}) \]
Practically speaking....

- If the training set is small:
  - batch learning using quasi-Newton or conjugate gradient descent
- If the training set is large:
  - stochastic gradient descent
- Somewhere in between
  - mini-batch
- Convergence is very sensitive to learning rate
  - Basically, it needs to be determined by trial-and-error (model selection or cross-validation)
Constrained optimization

- Unconstrained optimization
  - Gradient descent
  - Newton’s method
  - Batch vs online learning
  - Stochastic Gradient Descent

- Constrained optimization
  - Lagrange duality
  - SVM in primal and dual forms
  - Constrained methods
Lagrangian function

Start with optimization problem:

\[
\begin{align*}
\text{minimize} \quad & f_0(x) \\
\text{subject to} \quad & f_i(x) \leq 0, \quad i = \{1, \ldots, k\} \\
& h_j(x) = 0, \quad j = \{1, \ldots, l\}
\end{align*}
\]

Form Lagrangian using Lagrange multipliers \( \lambda_i \geq 0, \nu_i \in \mathbb{R} \)

\[
\mathcal{L}(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{k} \lambda_i f_i(x) + \sum_{j=1}^{l} \nu_j h_j(x)
\]
Original/primal problem

is equivalent to min-max optimization

Why?
- Consider a two-player game
- If player 1 chooses $x$ that violates a constraint $f_1(x) > 0$, player 2 choose $\lambda_1 \to \infty$ so that $L(x, \lambda, \nu) = \ldots + \lambda_1 f_1(x) + \ldots \to \infty$
- 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** (cf: Primal problem)
  
  \[
  \begin{align*}
  \text{maximize} & \quad \left[ \inf_{\lambda \geq 0, \nu} \mathcal{L}(x, \lambda, \nu) \right] \\
  \text{minimize} & \quad \left[ \sup_{\lambda \geq 0, \nu} \mathcal{L}(x, \lambda, \nu) \right]
  \end{align*}
  \]

- **Q:** How are primal and dual solutions related?
Weak duality

- Dual function lower-bounds the primal optimal value!

**Lemma (Weak Duality)**

*If* \( \lambda \geq 0 \), then

\[
g(\lambda, \nu) \leq f_0(x^*).
\]

**Proof.**

We have

\[
g(\lambda, \nu) = \inf_x \mathcal{L}(x, \lambda, \nu) \leq \mathcal{L}(x^*, \lambda, \nu)
\]

\[
= f_0(x^*) + \sum_{i=1}^{k} \lambda_i f_i(x^*) + \sum_{j=1}^{l} \nu_j h_j(x^*) \leq f_0(x^*).
\]
For convex problems, primal and dual solutions are equivalent!

Equivalently, \( \max \min L(x,\lambda,\nu) = \min \max L(x,\lambda,\nu) \)

What does the theorem mean in practice?
- You had a constrained minimization problem, which may be hard to solve
- Dual problem may be easier to solve (simpler constrains)
- When you solve the dual problem, it also gives the solution for the primal problem!
SVM Recap
SVM in primal form

- **Primal SVM**

  \[
  \begin{align*}
  \text{minimize} & \quad \frac{1}{2}||w||^2 \\
  \text{subject to} & \quad y_i(w \cdot x_i + w_0) \geq 1 \text{ for } i = 1, \ldots, m
  \end{align*}
  \]

  - 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

\[ L_P(w, w_0, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{m} \alpha_i (y_i (w \cdot x_i + w_0) - 1) \]

with \( \alpha_i \geq 0, i = 1, \ldots, m \). 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 \( \frac{\partial L_P}{\partial w} = (\frac{\partial L_P}{\partial w_1}, \ldots, \frac{\partial L_P}{\partial w_d}) \)

By substituting these constraints into \( L_P \) we get its dual form

\[ L_D(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i \cdot x_j \]
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
  - Discrete optimization
  - Graph optimization