IIT KHARAGPUR

IIT GANDHINAGAR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Scalable Data Science

Lecture 4: Background on Optimization

Sourangshu Bhattacharya

Computer Science and Engineering

IIT KHARAGPUR

In this review

- Outline:
  - Definition of an optimization problem
  - Properties of optima
  - Algorithm for differentiable objectives
  - Convex functions – subgradients
  - Subgradient descent.
  - Stochastic gradient descent.

Sourangshu Bhattacharya
Computer Science and Engg.

What is Optimization?

Find the minimum or maximum value of an objective function (f_0) w.r.t. arguments x.

The arguments must satisfy given a set of inequality and equality constraints.

General form:

$$\arg\min_x f_0(x)$$
$$\text{s.t.} f_i(x) \leq 0, i = \{1, \ldots, k\}$$
$$h_j(x) = 0, j = \{1, \ldots l\}$$

Example:

$$\min f(x,y) = x^2 + 2y^2$$
$$x > 0$$

or

$$\min f(x,y) = x^2 + 2y^2$$
$$-2 < x < 5, y \geq 1$$

or

$$\min f(x,y) = x^2 + 2y^2$$
$$x + y = 2$$

Why Do We Care?

Linear Classification

$$\arg\min_{w} \sum_{i=1}^{n} ||w||^2 + C \sum_{i=1}^{n} \xi_i$$

$$\text{s.t. } \; 1 - y_i x_i^T w \leq \xi_i$$

$$\xi_i \geq 0$$

Maximum Likelihood

$$\arg\max_{\theta} \sum_{i=1}^{n} \log \, p_\theta(x_i)$$

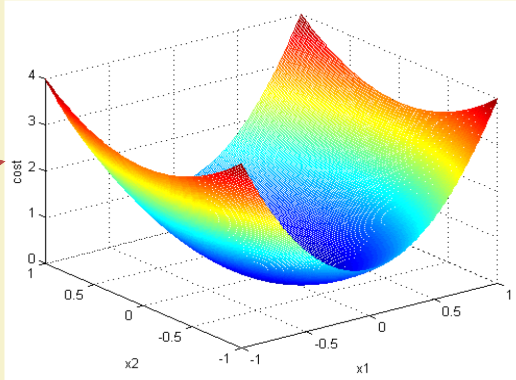
Machine Learning is Optimization !!

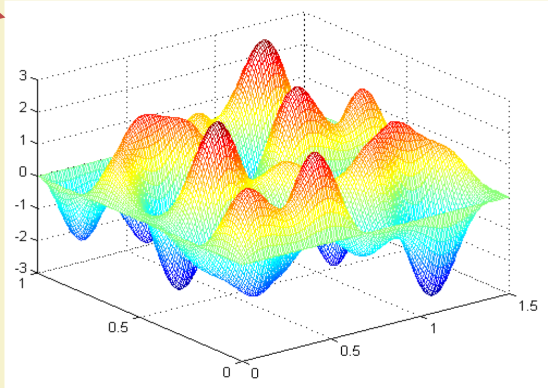
K-Means

$$\arg \min_{\mu_1, \mu_2, \ldots, \mu_k} J(\mu) = \sum_{j=1}^{k} \sum_{i \in C_j} ||x_i - \mu_j||^2$$

Types of objective functions

1. Objective functions may be unimodal or multimodal.
   a. Unimodal – only one optimum
   b. Multimodal – more than one optimum
2. Most algorithms work on unimodal functions. The optimum determined in such cases is called a local optimum.
3. The global optimum is the best of all local optimum designs.





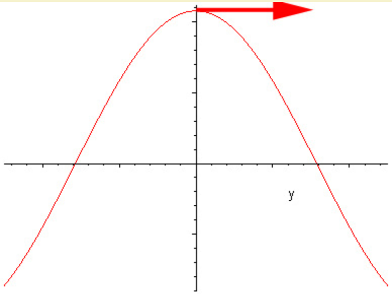
Types of optimization algorithms

- Derivative-based optimization (gradient based)
  - Objective function should be differentiable.
  - Capable of determining "search directions" according to an objective function's derivative.
    - Steepest descent method (Gradient descent);
    - Newton's method;
    - Conjugate gradient, etc.
- Derivative-free optimization
  - Searches over the feasible set in a systematic manner.
    - random search method;
    - genetic algorithm;
    - simulated annealing; etc.

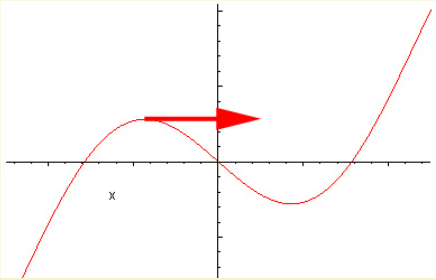
Gradient

- <u>Definition</u>: The gradient of f: $R^n \to R$ is a function ∇f: $R^n \to R^n$ given by

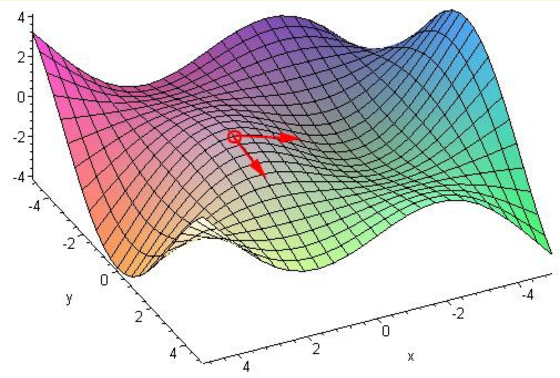
$$\nabla f(x_1, \ldots, x_n) := \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n} \right)^T$$



$$\frac{\partial f(x,y)}{\partial y}$$



$$\frac{\partial f(x,y)}{\partial x}$$

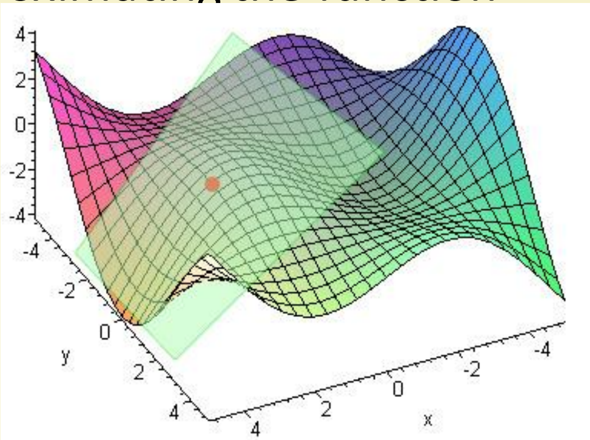


Gradient: $\left[\frac{\partial f(x,y)}{\partial x}, \frac{\partial f(x,y)}{\partial y} \right]$

Gradient

- The gradient defines (hyper) plane approximating the function infinitesimally

$$\Delta z = \frac{\partial f}{\partial x} \cdot \Delta x + \frac{\partial f}{\partial y} \cdot \Delta y$$



- For all directions v, $|v| = 1$:

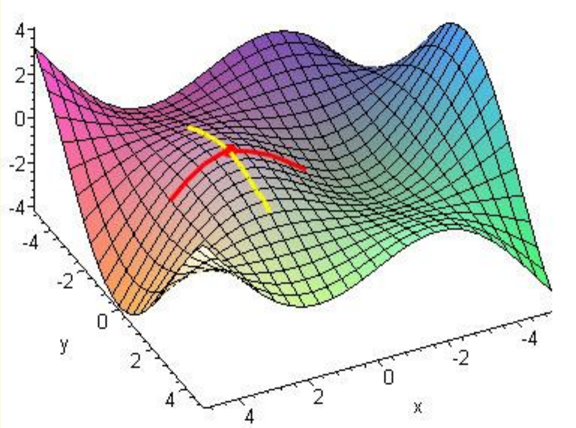
$$\frac{\partial f}{\partial v}(p) = \langle \nabla f_p, v \rangle$$

- Magnitude is highest when v and ∇f_p point to the same direction.
- Gradient points to direction of steepest descent

Gradient

- Let f: $R^n \to R$ be a smooth function around p, if f has local **minimum** (maximum) at p then:

$$\nabla f_p = \bar{0}$$

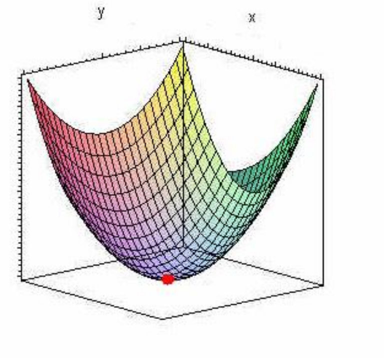


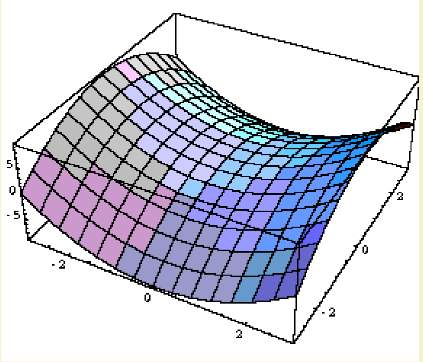
Intuitive: necessary for local min(max)

Hessian matrix

- If the derivative of ∇f exists, we say that f is twice differentiable.
  - Write the second derivative as $D^2 f$ (or $\boldsymbol{F}$), and call it the *Hessian* of f.

$$\boldsymbol{F} = D^2 f = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2}(\boldsymbol{x}) & \frac{\partial^2 f}{\partial x_2 \partial x_1}(\boldsymbol{x}) & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_1}(\boldsymbol{x}) \\ \frac{\partial^2 f}{\partial x_1 \partial x_2}(\boldsymbol{x}) & \frac{\partial^2 f}{\partial x_2^2}(\boldsymbol{x}) & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_2}(\boldsymbol{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_1 \partial x_n}(\boldsymbol{x}) & \frac{\partial^2 f}{\partial x_2 \partial x_n}(\boldsymbol{x}) & \cdots & \frac{\partial^2 f}{\partial x_n^2}(\boldsymbol{x}) \end{bmatrix}$$





- The local optimum is mínimum (máximum) if the Hessian matrix is positive (negative) definite.
- Else it is a saddle point.

Sourangshu Bhattacharya
Computer Science and Engg.

Constrained Optimization

$$
\begin{aligned}
&\text{Minimize} && f(x) && \\
&\text{Subject to} && g_j(x) \geqslant 0 && \text{for } j = 1, 2, \ldots, J \\
& && h_k(x) = 0 && \text{for } k = 1, 2, \ldots, K \\
& && x = (x_1, x_2, \ldots, x_N) &&
\end{aligned}
$$

Lagrangian: $L(x, u, v) = f(x) - \sum_{j=1}^{J} u_j g_j(x) - \sum_{k=1}^{K} v_k h_k(x)$

Sourangshu Bhattacharya
Computer Science and Engg.

Kuhn-Tucker conditions

Find vectors $x_{(N\times 1)}$, $u_{(1\times J)}$, and $v_{(1\times K)}$ that satisfy

$$\nabla f(x) - \sum_{j=1}^{J} u_j \nabla g_j(x) - \sum_{k=1}^{K} v_k \nabla h_k(x) = 0$$

$$g_j(x) \geqslant 0 \qquad \text{for } j = 1, 2, \ldots, J$$

$$h_k(x) = 0 \qquad \text{for } k = 1, 2, \ldots, K$$

$$u_j g_j(x) = 0 \qquad \text{for } j = 1, 2, \ldots, J$$

$$u_j \geqslant 0 \qquad \text{for } j = 1, 2, \ldots, J$$

Sourangshu Bhattacharya
Computer Science and Engg.

Algorithms

Gradient Descent

- An algorithm for: $\min_{x} f(x)$

Input $x_0 \in R^n$

Step 0: set $i = 0$

Step 1: if $\nabla f(x_i) = 0$ **stop**,

else, compute ***search direction*** $h_i \in R^n$

Negative Gradient direction

Step 2: compute the ***step-size*** $\lambda_i \in \arg\min_{\lambda \geq 0} f(x_i + \lambda \cdot h_i)$

Step 3: set $x_{i+1} = x_i + \lambda_i \cdot h_i$ go to step 1

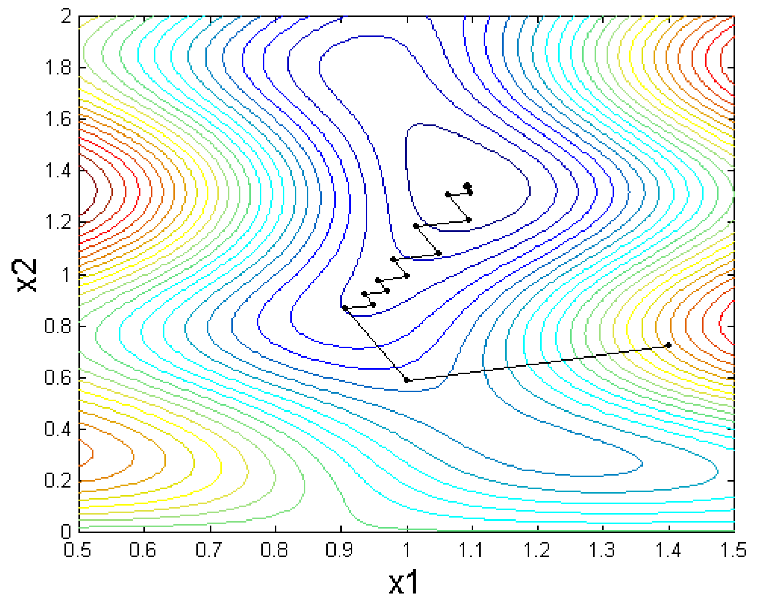
Sourangshu Bhattacharya
Computer Science and Engg.

Gradient Descent

Given:

$$f(x_1, x_2) = 2\sin(1.47x_1)\sin(0.34x_2) + \sin(x_1)\sin(1.9x_2)$$

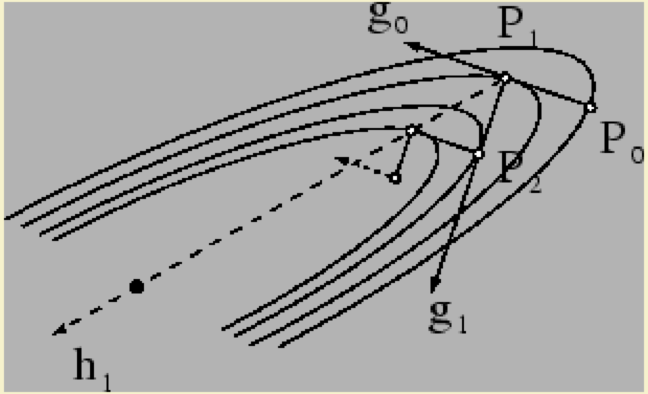
Find the minimum when x_1 is allowed to vary from 0.5 to 1.5 and x_2 is allowed to vary from 0 to 2.



Sourangshu Bhattacharya
Computer Science and Engg.

Gradient Descent

- What is the problem with steepest descent?



  - We can repeat the same directions over and over…
- Wouldn't it be better if, every time we took a step, we got it right the first time?

Newton's Method

Idea: use a second-order approximation to function.

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

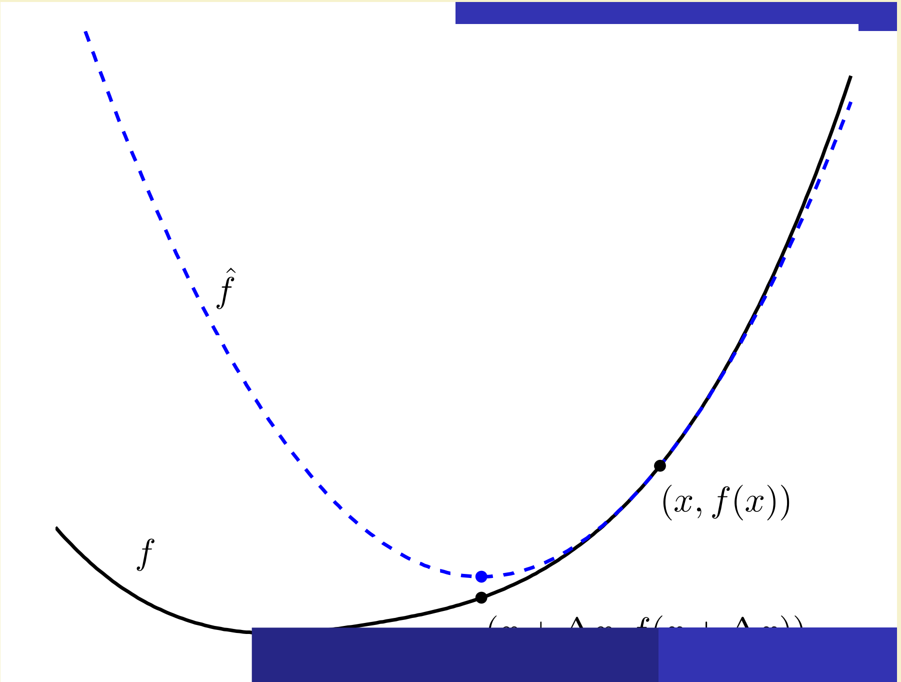
Choose Δx to minimize above:

$$\Delta x = -\underbrace{\left[\nabla^2 f(x)\right]^{-1}}_{\text{Inverse Hessian}} \underbrace{\nabla f(x)}_{\text{Gradient}}$$

This is descent direction:

$$\nabla f(x)^T \Delta x = -\nabla f(x)^T \left[\nabla^2 f(x)\right]^{-1} \nabla f(x) < 0.$$

Newton's Method Picture

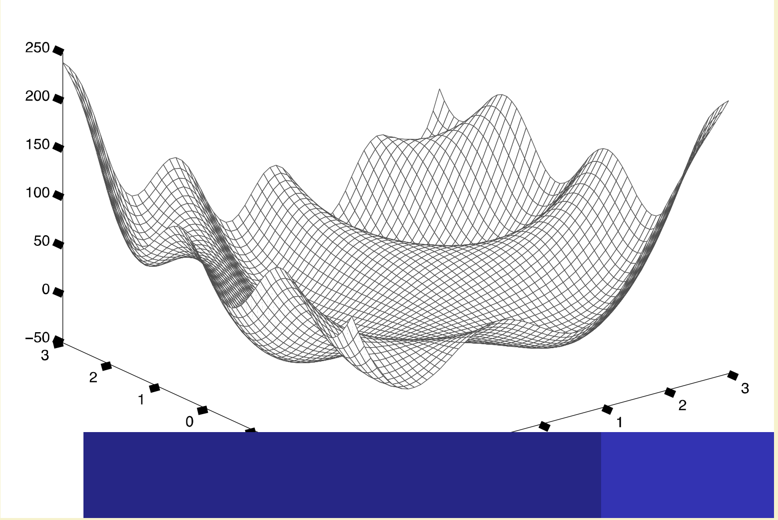


$\hat{f}$ is 2^{nd}-or[redacted]

Sourangshu Bhattacharya
Computer Science and Engg.

Prefer Convex Problems

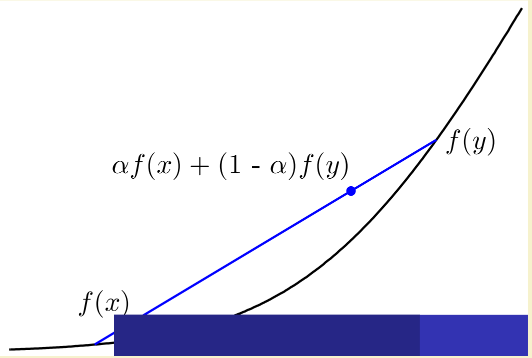
Local (non global) minima and maxima:

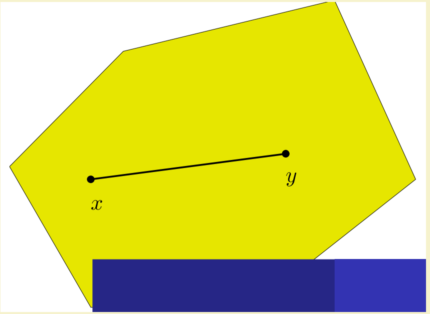


Convex Functions and Sets

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

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





A set $C \subseteq \mathbb{R}^n$ is convex if for $x, y \in C$ and any $a \in [0, 1]$,

$$ax + (1-a)y \in C$$

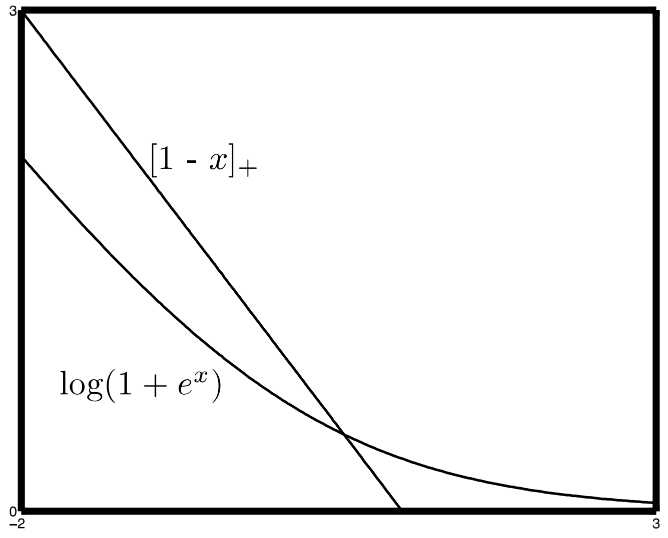
Important Convex Functions

SVM loss:

$$f(w) = \left[1 - y_i x_i^T w\right]_+$$

Binary logistic loss:

$$f(w) = \log\left(1 + \exp(-y_i x_i^T w)\right)$$

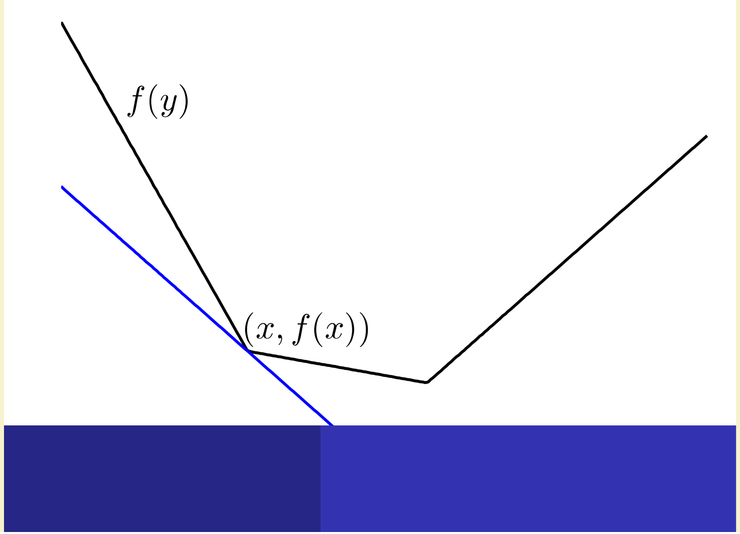


Convex Optimization Problem

$$\begin{aligned} \underset{x}{\text{minimize}} \;\; & f_0(x) && \text{(Convex function)} \\ \text{s.t.} \;\; & f_i(x) \leq 0 && \text{(Convex sets)} \\ & h_j(x) = 0 && \text{(Affine)} \end{aligned}$$

Subgradient Descent Motivation

Lots of non-differentiable convex functions used in machine learning:



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

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

Subgradient Descent – Algorithm

Really, the simplest algorithm in the world. Goal:

$$\underset{x}{\text{minimize}} \quad f(x)$$

Just iterate

$$x_{t+1} = x_t - \eta_t g_t$$

where η_t is a stepsize, $g_t \in \partial f(x_t)$.

Online learning and optimization

- Goal of machine learning :
  - Minimize expected loss

$$\min_h L(h) = \mathbf{E}\left[\mathrm{loss}(h(x), y)\right]$$

given samples

$$(x_i, y_i) \; i = 1, 2 \ldots m$$

- This is Stochastic Optimization
  - Assume loss function is convex

Batch (sub)gradient descent for ML

- Process all examples together in each step

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

where L is the regularized loss function

- Entire training set examined at each step
- Very slow when *n* is very large

Stochastic (sub)gradient descent

- "Optimize" one example at a time
- Choose examples randomly (or reorder and choose in order)
  - Learning representative of example distribution

for $i = 1$ to n:

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

where L is the regularized loss function

Stochastic (sub)gradient descent

for $i = 1$ to n:

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

where L is the regularized loss function

- Equivalent to online learning (the weight vector *w* changes with every example)
- Convergence guaranteed for convex functions (to local minimum)

References:

- R. Fletcher **Practical Methods of Optimization**, 2nd Edition. *John Wiley & Sons, Inc.* July 2000.
- Stephen Boyd and Lieven Vandenberghe. **Convex Optimization** *Cambridge University Press 2009*.
- Wikipedia.

Thank You!!