Artificial Intelligence

CS 165A

Jan 31, 2018

Instructor: Prof. Yu-Xiang Wang

Today

→ Finish unsupservised learning

→ Continuous optimization

Announcements

- Homework 2 will be released before midnight today.
  - It will be tougher than Homework 1
  - It will be overlapping with things that you need to write the code for MP1.

- Definitely start early!

Today

- Finish “unsupervised learning”
- Continuous optimization
  - Why everything is an optimization problem
  - Gradient Descent
  - Stochastic Gradient Descen

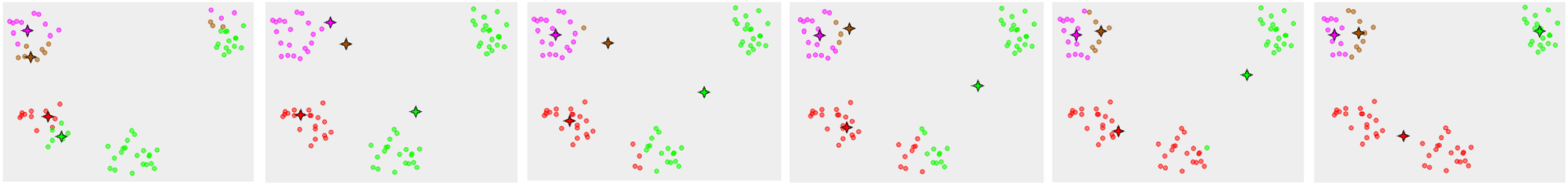
Recap: Unsupervised Learning so far

- x = f(x) with a compact/structured f
- Can be based on a generative model or discriminative model.
- Agents trying to summarize the essence about the observed data.

K-Means clustering

- Objective function $\min_{c_1,\ldots,c_k} \sum_i \min_{j\in[k]} \|x_i - c_j\|^2$
- Algorithm: Lloyd's algorithm.
  - Randomly initialize the centers at data points.
  - Assign data points to closest center
  - Update centers to be the mean of the data points assigned to it

- A typical run of Lloyd's algorithm:

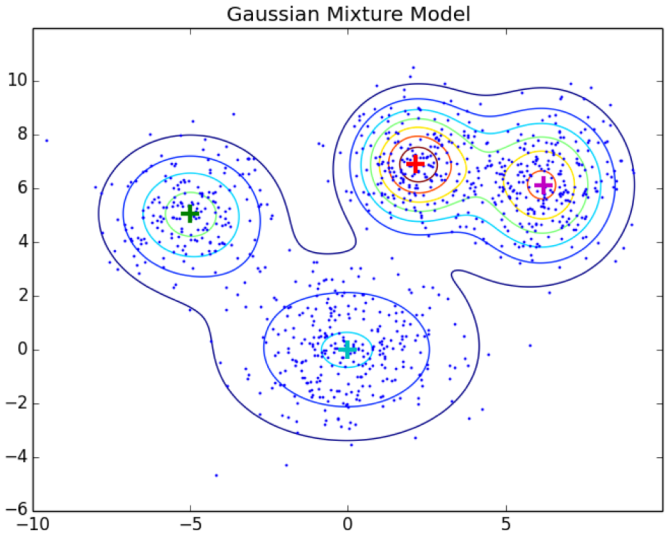


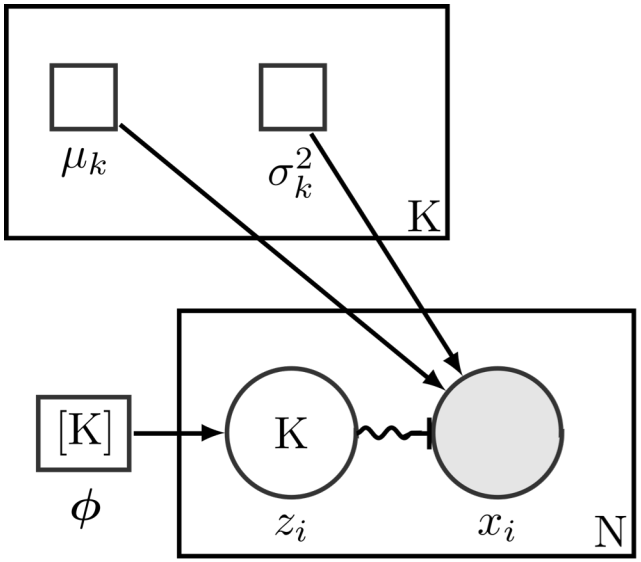
Gaussian Mixture Model

- Generative process:

$$
\begin{aligned}
\theta_{i=1\ldots K} &= \{\mu_{i=1\ldots K}, \sigma^2_{i=1\ldots K}\} \\
\mu_{i=1\ldots K} &= \text{mean of component } i \\
\sigma^2_{i=1\ldots K} &= \text{variance of component } i \\
z_{i=1\ldots N} &\sim \text{Categorical}(\boldsymbol{\phi}) \\
x_{i=1\ldots N} &\sim \mathcal{N}(\mu_{z_i}, \sigma^2_{z_i})
\end{aligned}
$$

- Graphical model:



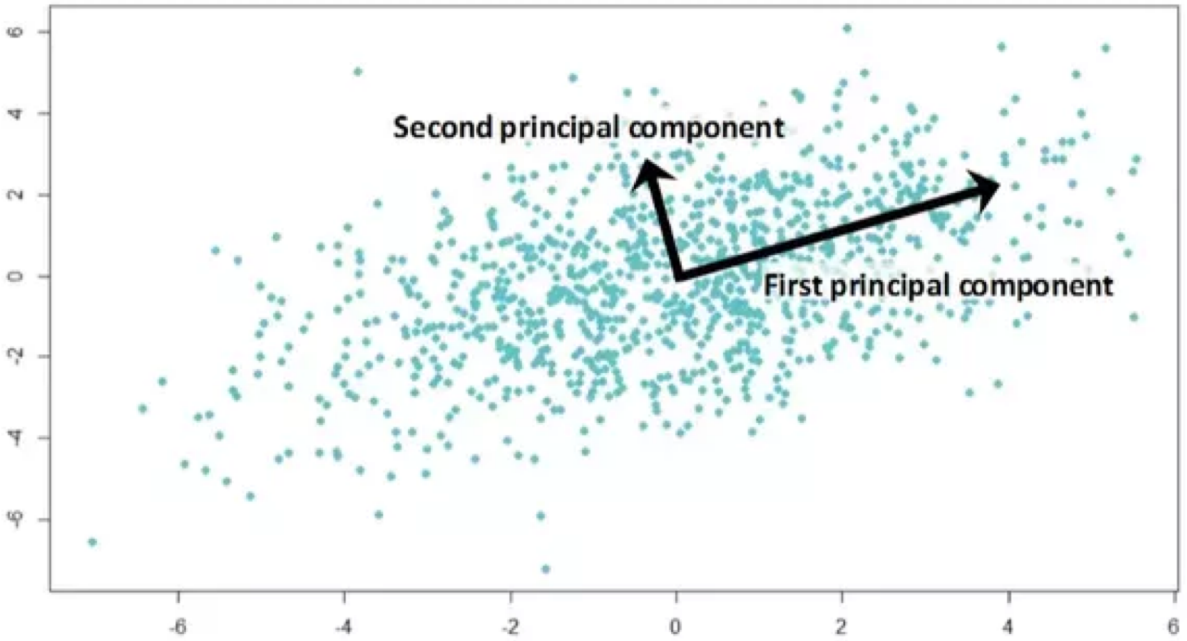


Principle Components Analysis

- Given data x in R^d, find best k-dimensional approximations that explains the variance of the data.

$$\min_{W \in \mathbb{R}^{d \times k}} \|WW^T - \mathrm{Cov}(X)\|_F^2$$

$$\mathrm{Cov}(X) = \frac{1}{n}(x_i - \bar{x})(x_i - \bar{x})^T$$



Probabilistic Principle Component Analysis

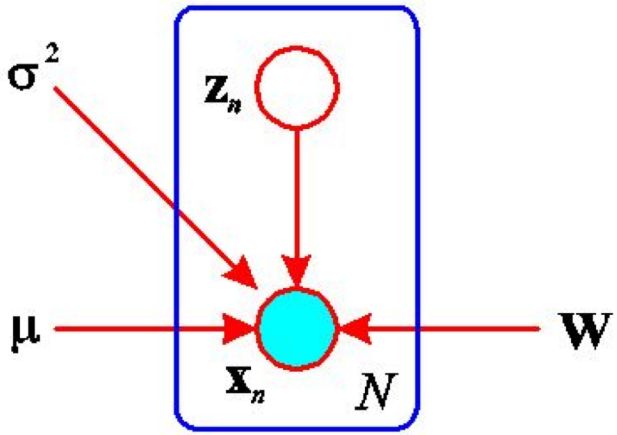
- Generative process:

$$z_i \sim N(0, I_k)$$
$$x_i \sim N(\mu + W z_i, \sigma^2 I_d)$$

Or equivalently:

$$x_i \sim N(\mu, WW^T + \sigma^2 I_d)$$

- BayesNet of PPCA: (from Chris Bishop)



PCA vs. MLE of Prob. PCA

- Sample covariance matrix:

$$\mathrm{Cov}(X) = \frac{1}{n}(x_i - \bar{x})(x_i - \bar{x})^T$$

- Singular Value decomposition of Cov(X) = U Λ U'

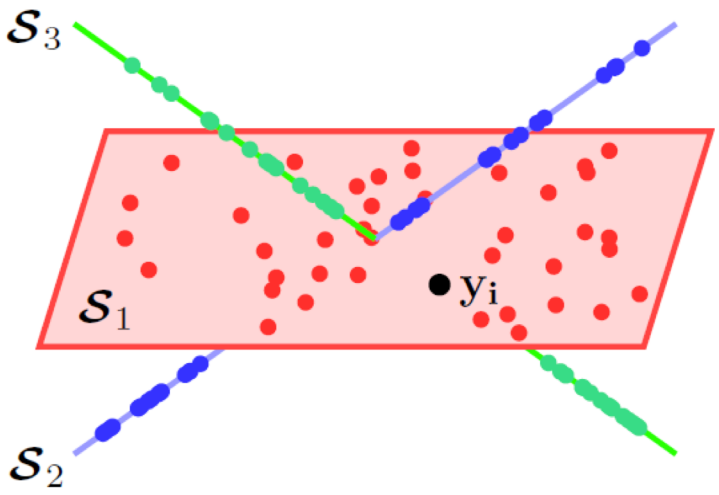
- PCA: Output U[:, 1:k] Λ[:, 1:k]$^{1/2}$

- PPCA's MLE: $W_{\mathrm{MLE}} = U[:, 1:k](\Lambda[:, 1:k] - \sigma^2 I_k)^{1/2}$

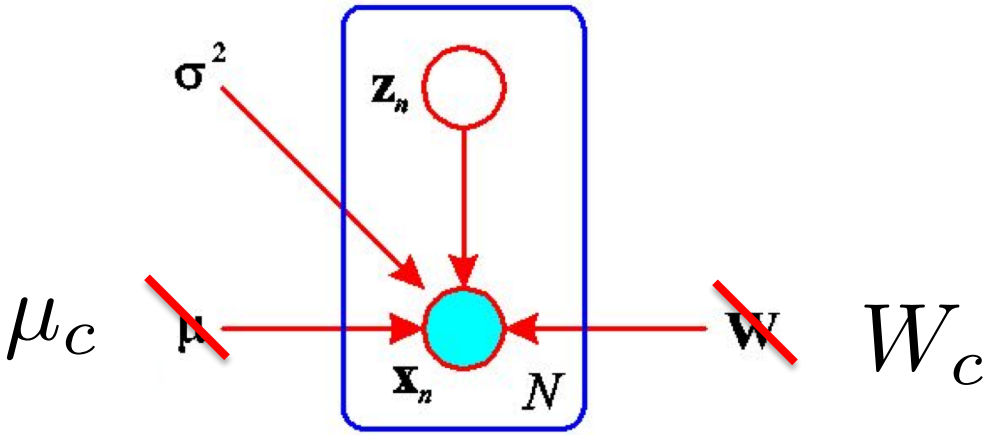
$$\sigma^2_{\mathrm{MLE}} = \frac{1}{d-k}\sum_{j=k+1}^{d} \lambda_j$$

Clustering and Dimension Reduction at the Same time? Subspace Clustering problems.

- Model-based: Assume data are close to Union-of-subspaces --- Sparse Subspace Clustering and etc…
- Model-free: Find best k-subspace approximation.
  - Q-flats, k-planes. Studied mostly in CS theory.



Mixture of Probabilistic PCA



μ_c W_c

$$c_i \sim \text{Categorical}(\tau)$$

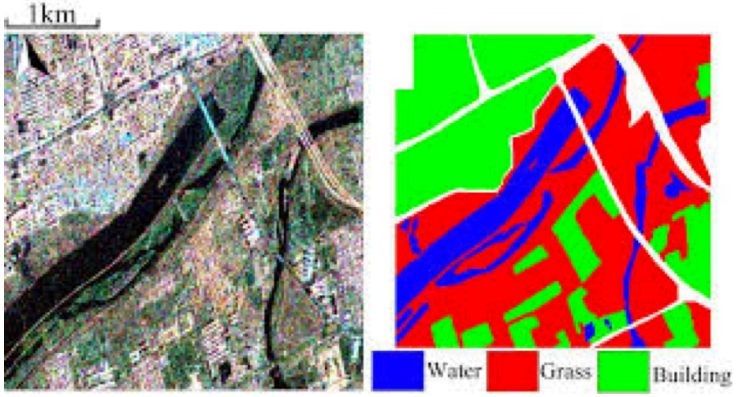
Algorithms for subspace clustering

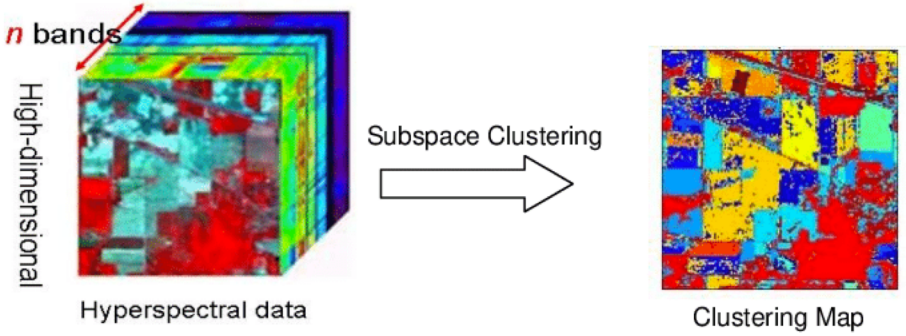
- EM for MPPCA.
  - Statistically sound but easily stuck at local minima.
- Gibbs sampling for MPPCA
  - Not a polynomial time algorithm.
  - Sampling in high-dimension is generically hard.
- Sparse Subspace Clustering (Elhamifar and Vidal, 2009)
  - Works amazingly well for the "model-based" subspace clustering
  - Robust theoretical guarantee well-understood:
    - Soltanolkotabi and Candes (2012), W. and Xu (2013) and many more…

- Initializing EM of MPPCA with SSC?
  - Possibly a low-hanging fruit…

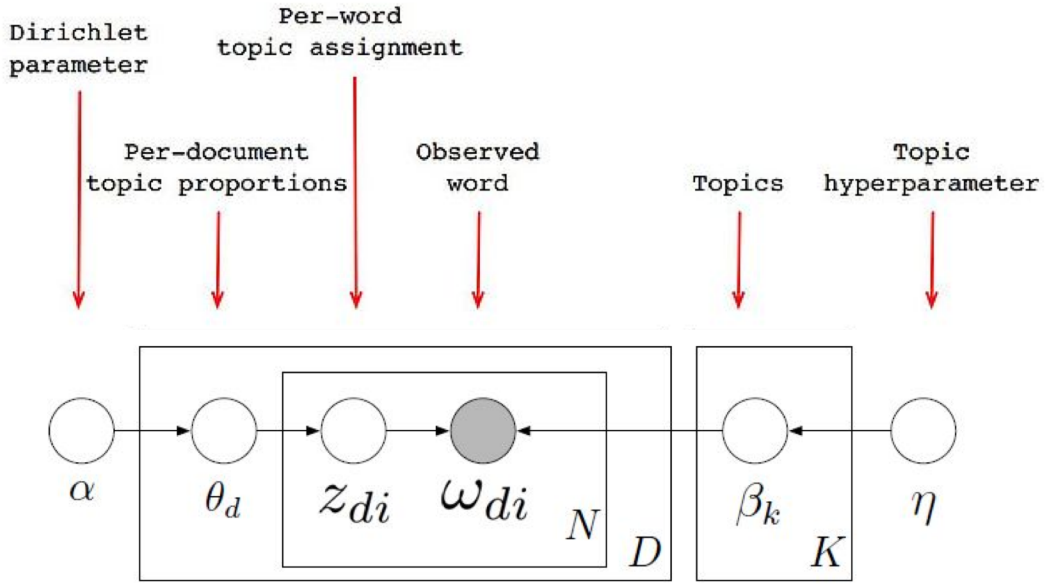
Applications of subspace clustering

- Self-driving car
- Hyperspectral imaging
- Face clustering
- And more!





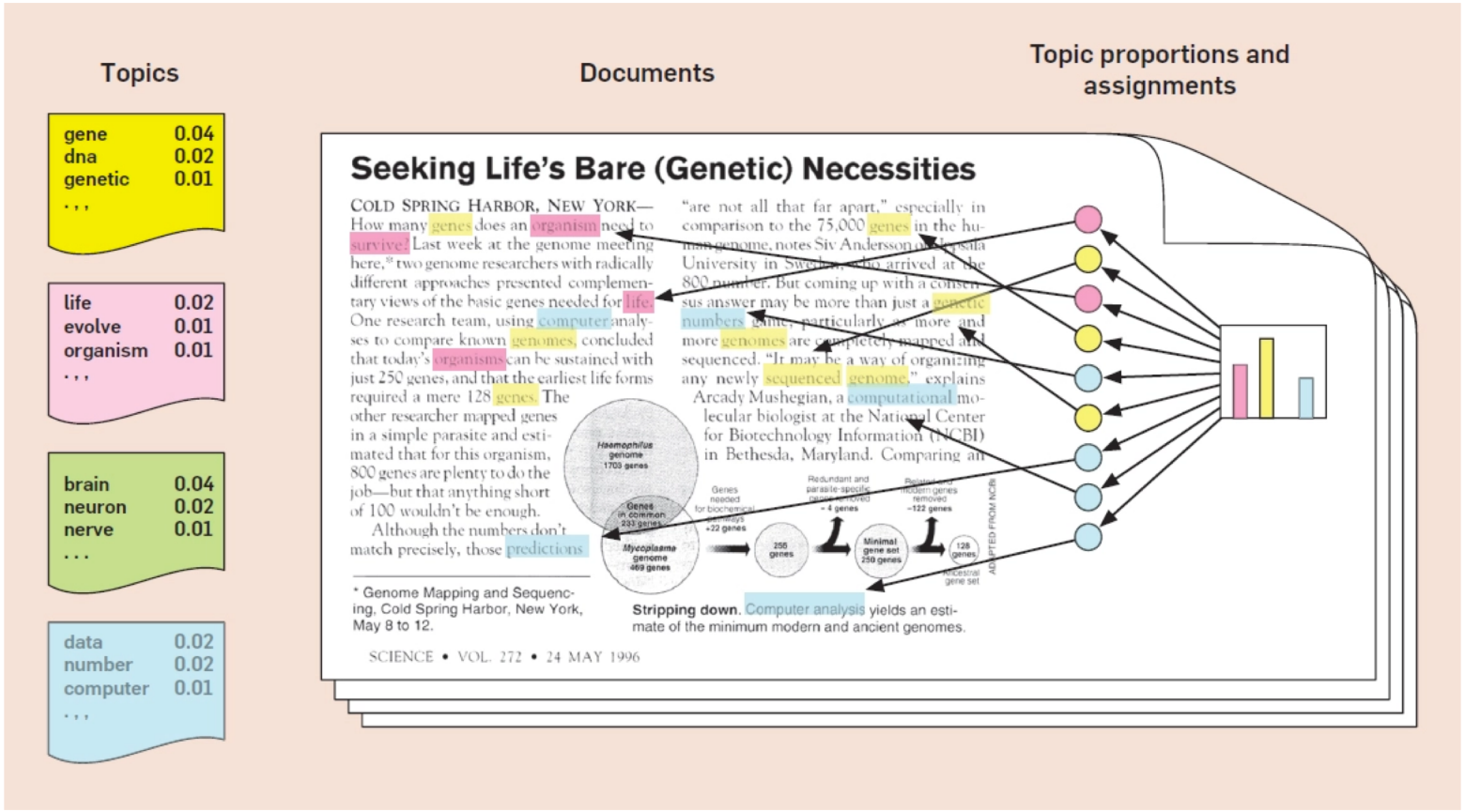
Topic modelling: Latent Dirichlet Allocation (A Dirichlet Admixture Model)



1) Draw each topic $\beta_k \sim \text{Dirichlet}(\eta)$
2) For each document:
   1) Draw topic proportions $\theta_d \sim \text{Dirichlet}(\alpha)$
   2) For each word:
      1) Draw $z_{di} \sim \text{Mult}(\theta_d)$
      2) Draw $\omega_{di} \sim \text{Mult}(\boldsymbol{\beta}_{z_{di}})$

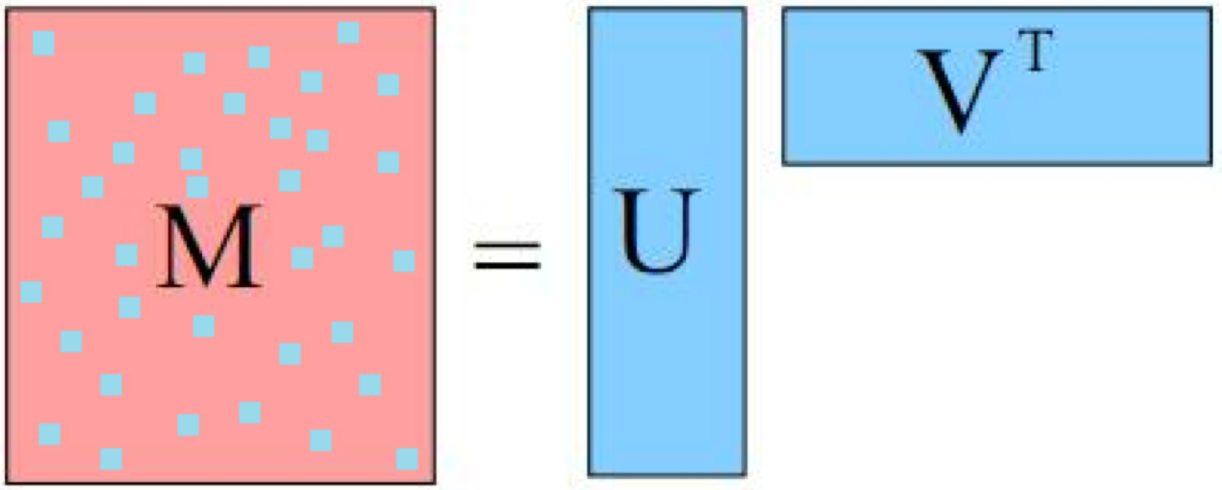
Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." *Journal of machine Learning research*3.Jan (2003): 993-1022.

Example of LDA: Topics and Assignments of Documents to those Topics.



The matrix factorization view

- Your data is a matrix M
- Most unsupervised learning models aim at finding some low-rank factorization of M into:



- PCA: U is principle components, V are coefficients.
- LDA: U is topics, V are topic mixture probabilities
- K-Means: U is cluster centers, V is cluster assignments.

Plans for subsequent lectures

- The last two lectures
  - Modelling the world
  - Coming up with objective functions to optimize
- Today
  - Scalable algorithm to optimize these objective functions
- Next Tuesday
  - Data is not generated from our model
  - Future data are not in our training data set
  - Why does this work at all?

Optimization perspective of AI

- A rational agent $\max_{a_1,\ldots,a_T} \text{Utility}(a_1, \ldots, a_T)$
- Discrete optimization $\min_{p \in \text{Paths}} \text{Distance}(p)$
  - **Algorithmic tool**: Dynamic programming
- Continuous optimization $\min_{\theta \in \mathbb{R}^d} \text{TrainingError}(\theta)$
  - **Algorithmic tool**: Gradient descent / Stochastic gradient descent

Different objectives to optimize

- Statistical estimation:
  - MLE: Maximize the log-likelihood function
  - MAP: log-likelihood function + log-prior
- Machine Learning
  - Empirical Risk Minimization (ERM):

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta, (x_i, y_i))$$

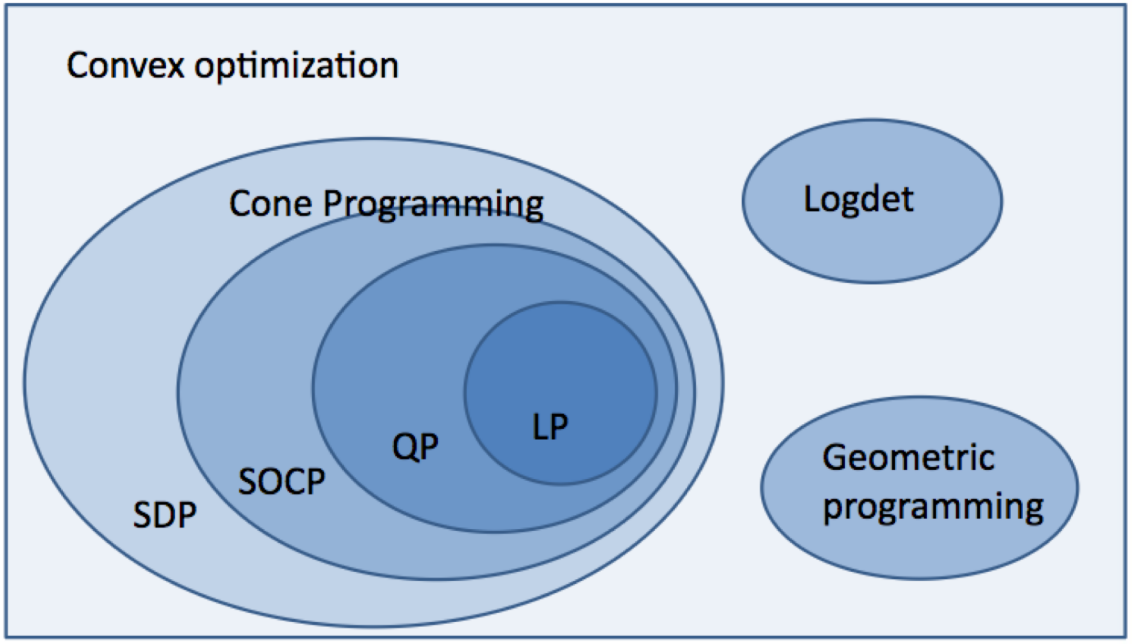
  - Regularized ERM:

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta, (x_i, y_i)) + \lambda r(\theta)$$

Hierarchies of optimization

- Traditionally:
  - Linear Programming vs. Nonlinear Programming

- Modern view:
  - Convex optimization vs. Nonconvex optimization



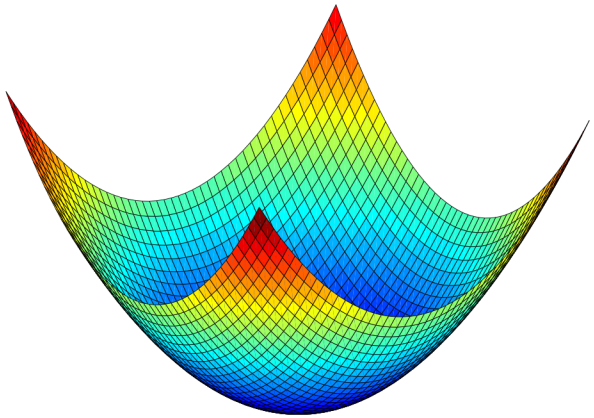
Convex vs. Non-convex optimization

Progress is only tip of the iceberg..

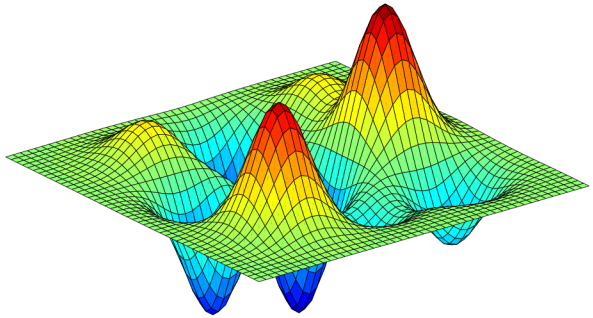
Real world is mostly non-convex!

Images taken from `https://www.facebook.com/nonconvex`

Convex vs Nonconvex optimization



- Unique optimum: global/local.

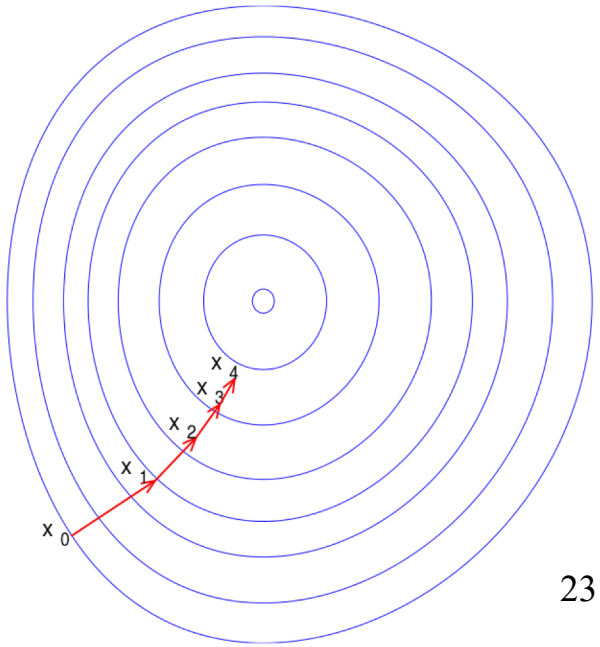


- Multiple local optima
- In high dimensions possibly exponential local optima

The machine learning take of this problem

- The problem: $\min_{\theta} f(\theta)$
- Let's just optimize it anyway!
  - With gradient descent.
- Assumption: The objective function is differentiable almost everywhere.

$$\theta_{t+1} = \theta_t - \eta_t \nabla f(\theta_t)$$



What do we consider to be successful

- For convex optimization

$$f(\theta_t) - \min_{\theta} f(\theta) \leq \epsilon_t$$

- For nonconvex optimization
  - First order optimality condition: $\left\|\nabla f(\theta_t)\right\|^2 \leq \epsilon_t$

Convergence analysis of Gradient Descent

- The problem: min_θ f(θ)
- Assumption: f is L-smooth, f* > - ∞
- The algorithm:

$$\theta_{t+1} = \theta_t - \eta_t \nabla f(\theta_t)$$

- Derivation on the blackboard.

Potential mid-term question:

Bounding the suboptimality by additionally assuming that f is convex!

Time complexity

- In Machine Learning

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta, (x_i, y_i))$$

- In every iteration, evaluate the gradient

$$\frac{1}{n} \sum_{i} \nabla_\theta \ell(\theta, (x_i, y_i))$$

Can we do better?

Stochastic Gradient Descent

- Gradient descent

$$\theta_{t+1} = \theta_t - \eta_t \nabla f(\theta_t)$$

- Stochastic gradient descent

$$\theta_{t+1} = \theta_t - \eta_t \hat{\nabla} f(\theta_t)$$

- Using a stochastic approximation of the gradient:

$$\mathbb{E}[\hat{\nabla} f(\theta_t) | \theta_t] = \nabla f(\theta_t)$$

$$\mathbf{Var}[\hat{\nabla} f(\theta_t) | \theta_t] \leq \sigma^2$$

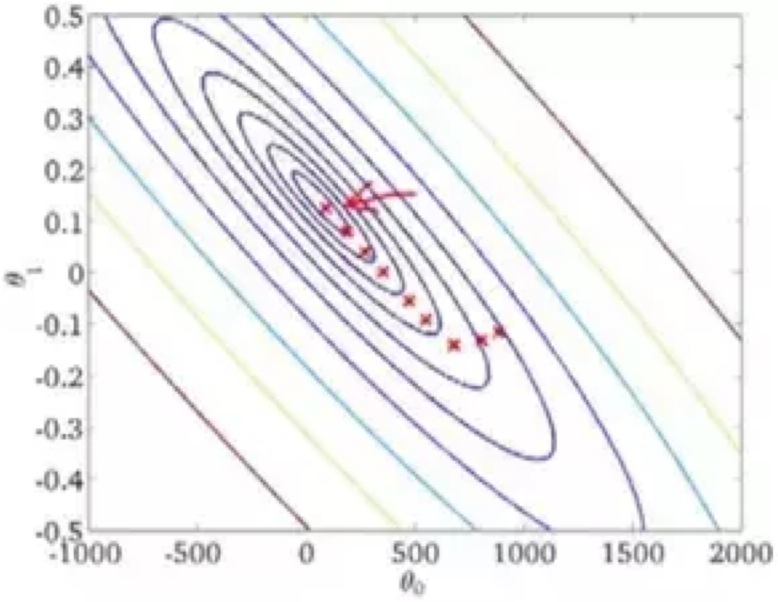
One natural stochastic gradient to consider in machine learning

- Recall that

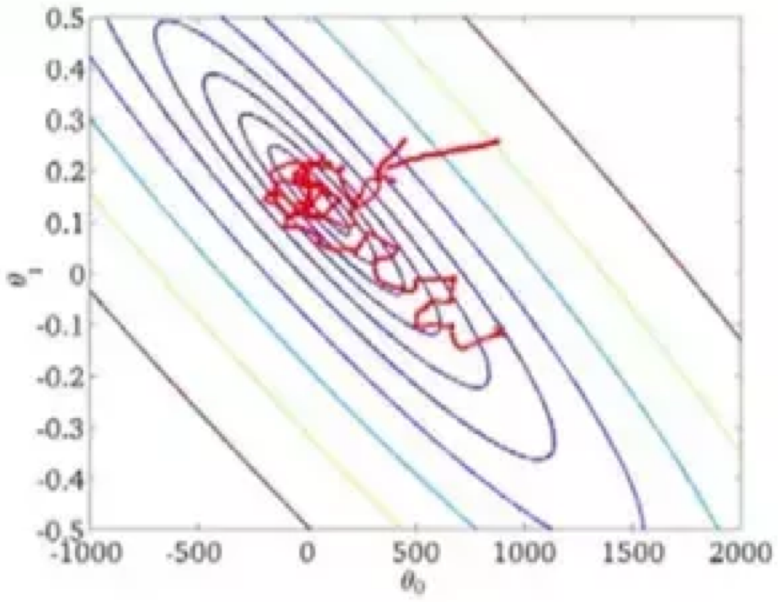
$$\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta, (x_i, y_i))$$

- Pick a **single** data point i uniformly at random
  - Use $\nabla_\theta \ell(\theta, (x_i, y_i))$
  - Show that this is an unbiased estimator!
  - Show that under Lipschitz condition this has bounded variance!

Illustration of GD vs SGD



Batch Gradient Descent



Stochastic Gradient Descent

Convergence of SGD

- $\min_\theta f(\theta)$
- Assumption: f is L-smooth, $f^* > -\infty$

- Work on the blackboard.

- What's the time complexity of SGD? In terms of number of stochastic gradient?

We are optimizing the stochastic objective

- Empirical risk minimization

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \ell(\theta, (x_i, y_i))$$

- What we really want to solve:

$$\min_{\theta \in \mathbb{R}^d} \mathbb{E}_{((x,y)\sim\mathcal{D})} \Big[\ell(\theta, (x, y)) \Big]$$

- The 1/sqrt{t} convergence rate is optimal up to a logarithmic factor!