[80240603 Advanced Machine Learning, Fall, 2012]

# Latent Feature Learning

**Jun Zhu**
dcszj@mail.tsinghua.edu.cn
State Key Lab of Intelligent Technology & Systems
Tsinghua University

November 29, 2011

# Outline

- Recap of what we have learned on nonparametric Bayesian methods
  - DP mixtures, Gaussain processes, Monte Carlo methods
- Finite latent feature models
  - PCA as the particular example
- Infinite latent feature models
  - Indian buffet process

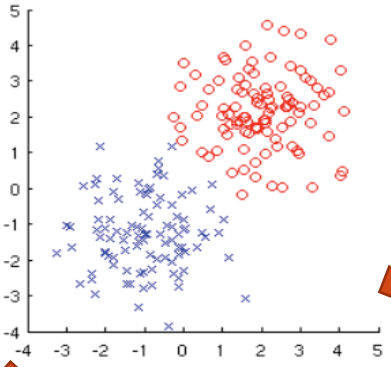
# Overview of what we have learn …

- What nonparametric Bayesian methods are?
- Dirichlet Process
  - Mixtures
  - Hierarchical Dirichlet process mixture models
- Gaussian Process
  - Regression
  - Classification
  - Laplace approximation methods
- Monte Carlo methods
  - Rejection sampling
  - Importance sampling
  - Markov chain Monte Carlo methods

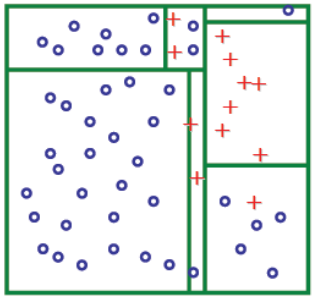
# Recap. of Nonparametric Bayesian

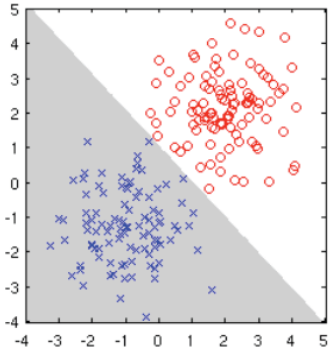
- What should we expect from nonparametric Bayesian methods?
  - Complexity of our model should be allowed to grow as we get more data
  - Place a prior on an unbounded number of parameters

# Example: Classification

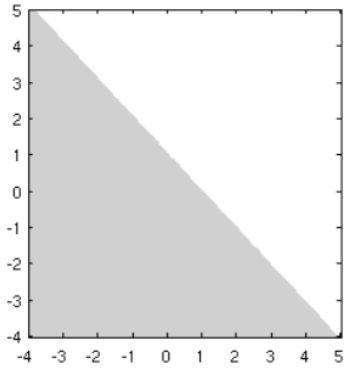


Data



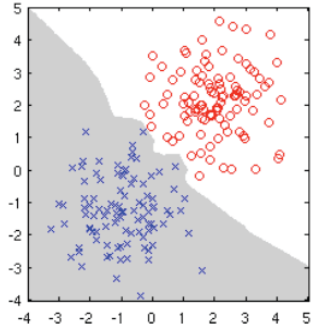


Build model



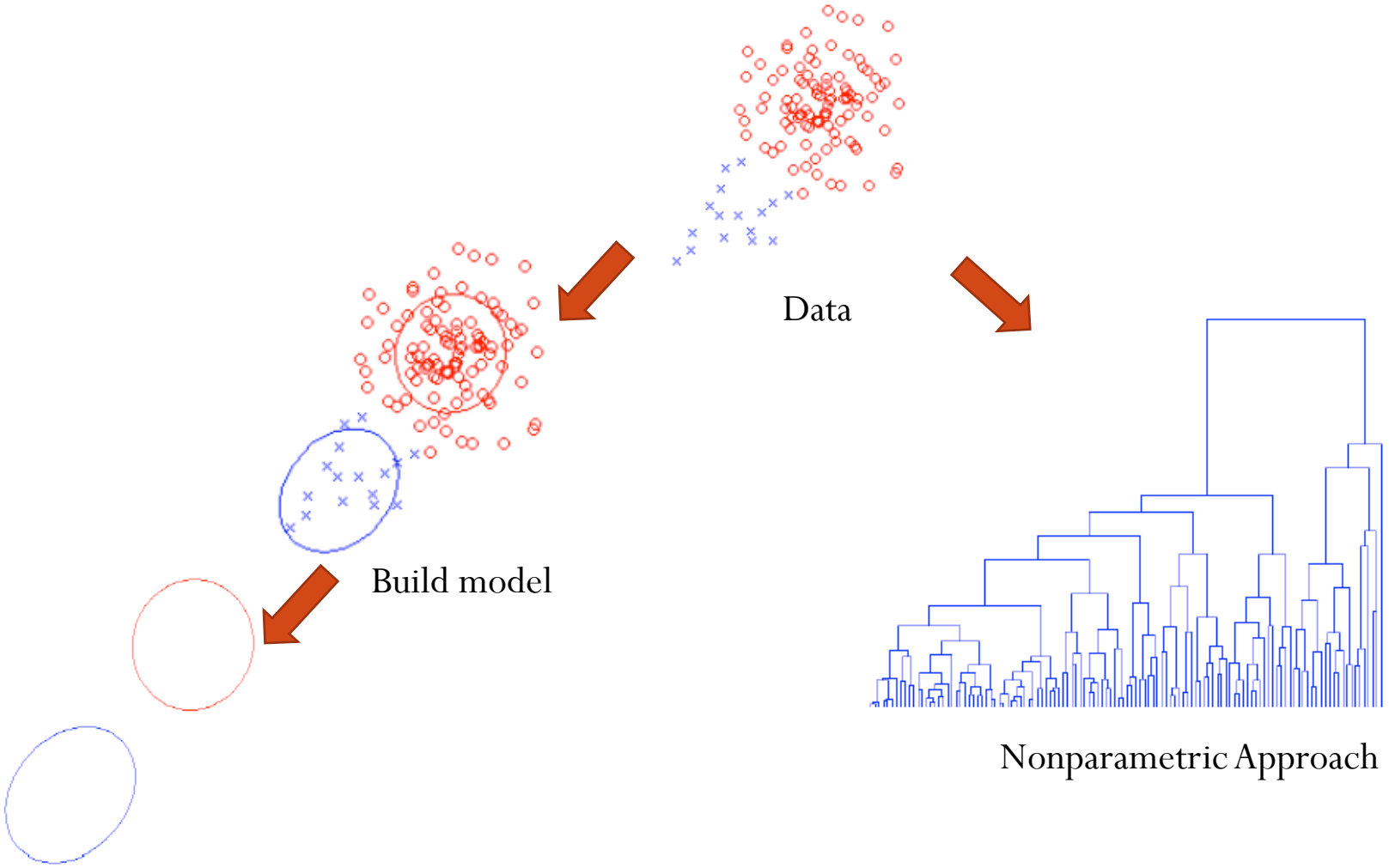
Predict using model

Parametric Approach

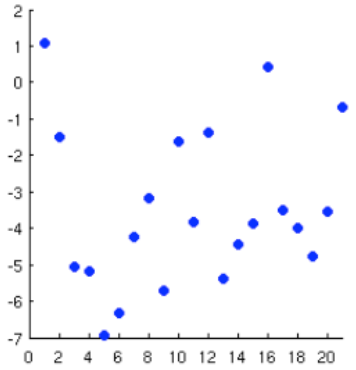


Nonparametric Approach

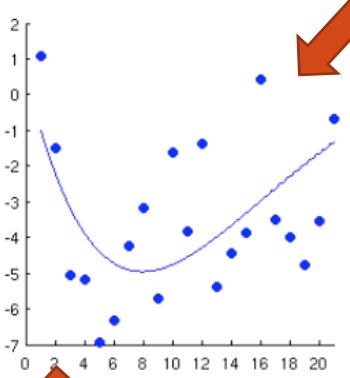
# Example: Clustering



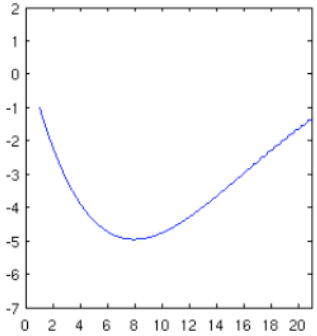
# Example: Regression




Data

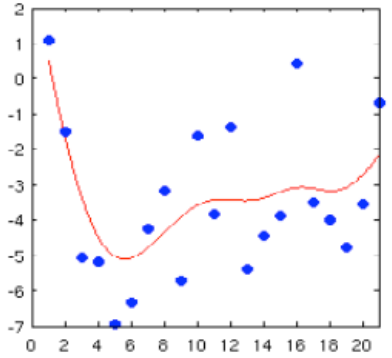



Build model




Predict using model

Parametric Approach




Nonparametric Approach

# A Nonparametric Bayesian Approach to Clustering

- We must again specify two things:
  - The likelihood function (how data is affected by the parameters):

$$p(\mathcal{D}|\theta)$$

    Identical to the parametric case.

  - The prior (the prior distribution on the parameters):

$$p(\theta)$$

    The Dirichlet Process!

- Exact posterior inference is still intractable. But we have can derive the Gibbs update equations!

# What is Dirichlet Process?

$$(G(A_1), \ldots, G(A_m)) \sim \text{Dirichlet}(\alpha G_0(A_1), \ldots, \alpha G_0(A_m))$$

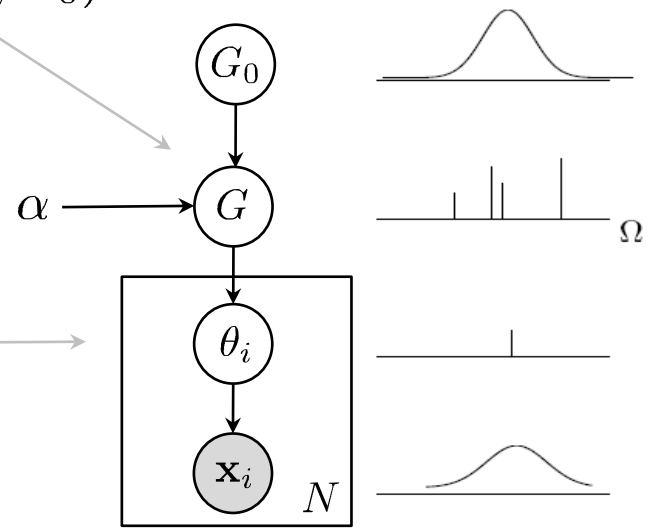



[http://www.nature.com/nsmb/journal/v7/n6/fig_tab/nsb0600_443_F1.html]

# The DP, CRP and Stick-Breaking Process

- Three birds on the same stone

$$G \sim DP(\alpha, G_0)$$




Stick-breaking Process (just the weights)

The CRP describes a partition of $\theta$ when $G$ is marginalized out

# Inference for DP Mixtures – Gibbs sampler

- We introduce the indicators $z_i$ and use the CRP representation.
- Randomly initialize $Z, \theta$ . Repeat:
  - sample each $z_i$ from

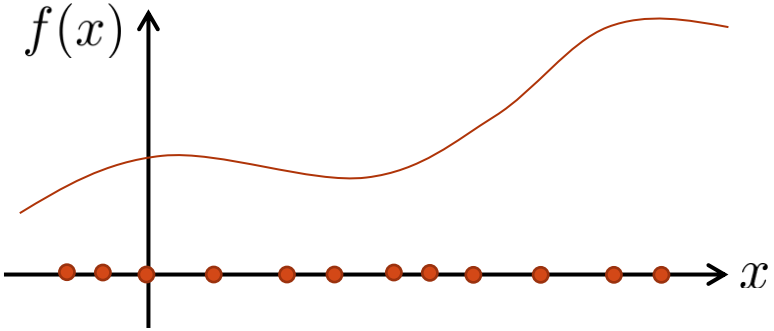
$$z_i | Z_{-i}, \theta, X \propto \sum_{k=1}^{K} n_{-i}^{k} p(\mathbf{x}_i | \theta_k) \delta_{z_i,k} + \alpha f(\mathbf{x}_i | G_0) \delta_{z_i,K+1}$$

  - Sample each $\theta_k$ based on $Z$ and $X$ only for occupied clusters

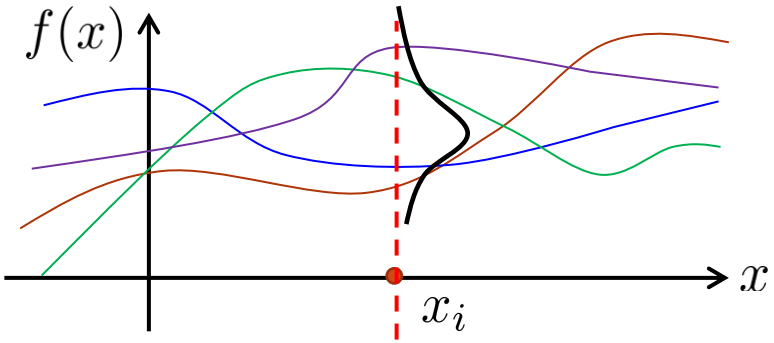
- This is the sampler we saw earlier, but now with some theoretical basis.

# Random Function vs. Random Variable

- A function is represented as an infinite vector with a index set




- For a particular point $x_i$, $f(x_i)$ is a random variable



# Gaussian Process

- A Gaussian process (GP) is a generalization of a multivariate Gaussian distribution to infinitely many variables; thus functions

- **Def**: A stochastic process is Gaussian if and only if for every finite set of indices $x_1, \ldots, x_n$ in the index set

$$(f(\mathbf{x}_1), \cdots, f(\mathbf{x}_n))$$

  is a vector-valued Gaussian random variable

- A Gaussian distribution is fully specified by the mean vector and covariance matrix

$$\mathrm{f} = (f_1, \cdots, f_n)^\top \sim \mathcal{N}(\mu, \Sigma)$$

- A Gaussian process is fully specified by a mean function and covariance function

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), \kappa(\mathbf{x}, \mathbf{x}'))$$

  - Mean function $m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$
  - Covariance function

$$\kappa(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}' - m(\mathbf{x}'))]$$

# Prediction with Noise-free Observations

- For noise-free observations, we know the true function value

$$\{(\mathbf{x}_i, f_i)\}_{i=1}^n$$

- The joint distribution of training output $\mathbf{f}$ and test outputs $\mathbf{f}_*$

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right)$$

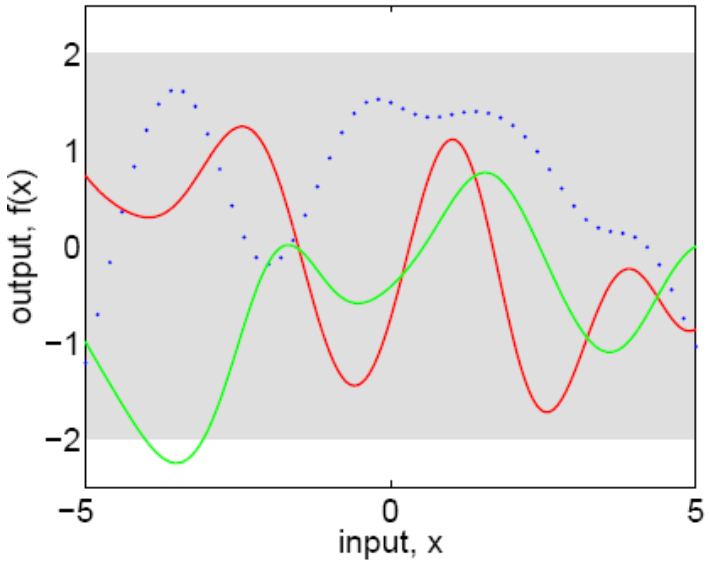
$$\mathbf{f}_* | X_*, X, \mathbf{f} \sim \mathcal{N}\Big( K(X_*,X)K(X,X)^{-1}\mathbf{f}, \\ K(X_*,X_*) - K(X_*,X)K(X,X)^{-1}K(X,X_*)\Big)$$

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} A & C \\ C^\top & B \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} \tilde{A} & \tilde{C} \\ \tilde{C}^\top & \tilde{B} \end{bmatrix}^{-1}\right)$$

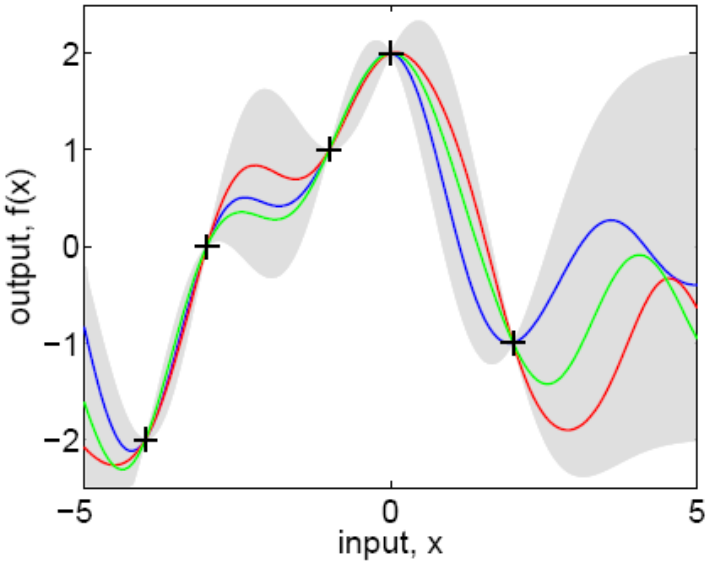
$$\mathbf{x}|\mathbf{y} \sim \mathcal{N}(\mu_x + CB^{-1}(\mathbf{y} - \mu_y),\ A - CB^{-1}C^\top)$$

# Posterior GP

- Samples from the prior and the posterior after observing "+"



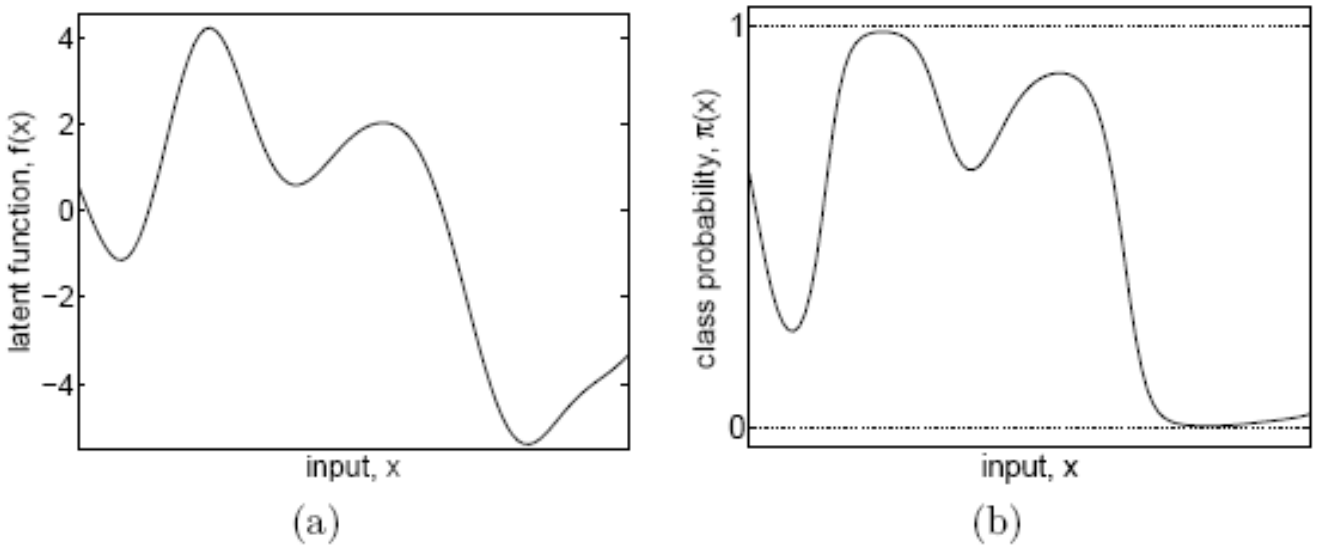

(a), prior




(b), posterior

  - shaded region denotes twice the standard deviation at each input
- Why the variance at the training points is zero?

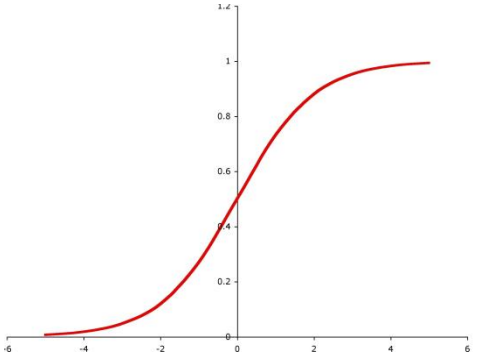
# Gaussian Process Classification




- Latent function f(x)

$$f(x) \sim GP(\mathrm{m}(x), K(x, x'))$$

$$\pi(x) \triangleq p(y = +1|x) = \sigma(f(x))$$

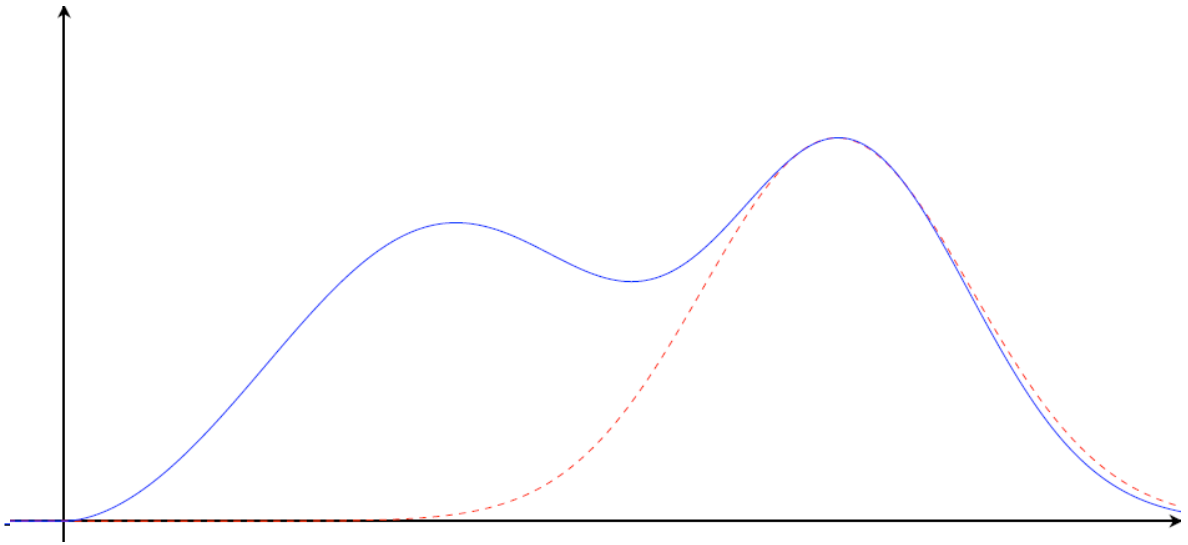


- Observations are independent given the latent function

$$p(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^{n} p(y_i|f_i)$$

# Laplace Approximation Methods

- Approximating a hard distribution with a "nicer" one



- Laplace approximation is a method using a Gaussian distribution as the approximation
- What Gaussian distribution?

# Laplace Approximation Methods

- Deviations by Taylor series expansion

$$f(x) = f(x_0) + \nabla f(x)|_{x=x_0}(x - x_0) + \frac{1}{2}\nabla^2 f(x)|_{x=x_0}(x - x_0)^2 + h.o.t\ldots$$

  - assume that the high-order terms are negligible
  - since $f(x_0)$ is a local maxima, $\nabla f(x)|_{x=x_0} = 0$

- Then, take the first three terms of the Taylor series at $x_0$

$$f(x) \approx f(x_0) + \frac{1}{2}\nabla^2 f(x)|_{x=x_0}(x - x_0)^2$$

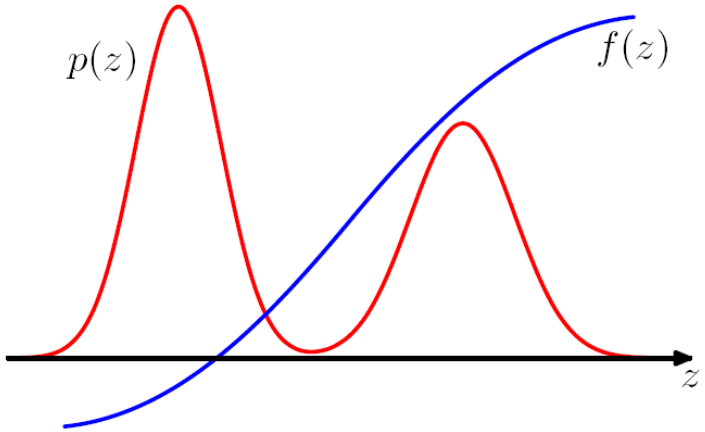
$$\int_a^b e^{Mf(x)}dx = e^{Mf(x_0)} \int_a^b \exp\left(\frac{1}{2}M\nabla^2 f(x)|_{x=x_0}(x - x_0)^2\right)dx$$

$$\text{Let } \sigma^2 = -\frac{1}{M\nabla^2 f(x)|_{x=x_0}}$$

$$\int_a^b e^{Mf(x)}dx = e^{Mf(x_0)} \int_a^b \exp\left(-\frac{1}{2\sigma^2}(x - x_0)^2\right)dx = e^{Mf(x_0}\sqrt{2\pi\sigma^2}$$

# Problems to be Solved by Monte Carlo Methods

- **Sampling**
  - to generate a set of samples $\{\mathbf{z}_l\}_{l=1}^{L}$ from a given probability distribution $p(\mathbf{z})$
  - the distribution is called target distribution
  - can be from statistical physics or data modeling
- **Integral**
  - To estimate expectations of functions under this distribution




$$\mathbb{E}[f] = \int f(\mathbf{z})p(\mathbf{z})d\mathbf{z}$$

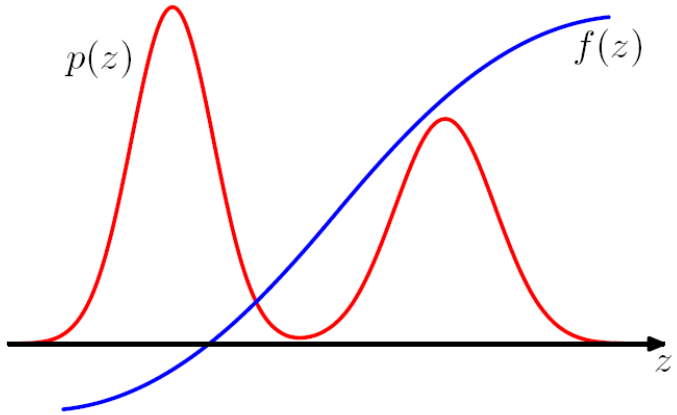
# Basic Procedure of Monte Carlo Methods

- Draw a set of independent samples

$$\forall 1 \leq l \leq L, \quad \mathbf{z}^{(l)} \sim p(\mathbf{z})$$

- Approximate the expectation with

$$\hat{f} = \frac{1}{L} \sum_{l=1}^{L} f(\mathbf{z}^{(l)})$$




  - where is the distribution p? $p(\mathbf{z}) \approx \frac{1}{L} \sum_{l=1}^{L} \delta_{\mathbf{z},\mathbf{z}^{(l)}}$ Histogram with Unique Points as the Bins
  - why this is good?

$$\mathbb{E}[\hat{f}] = \mathbb{E}[f] \qquad \mathrm{var}[\hat{f}] = \frac{1}{L}\mathbb{E}[(f - \mathbb{E}[f])^2]$$

  - Accuracy of estimator does not depend on dimensionality of z
  - High accuracy with few (10-20 independent) samples
  - However, obtaining independent samples is often not easy!

# Basic Sampling Algorithms

- If z is uniformly distributed over (0, 1), then $y = f(z)$ has the distribution

$$p(y) = p(z)\left|\frac{dz}{dy}\right|$$

  - where $p(z) = 1$
- Normally, we know *p(y)* and infer *f*. This can be done via

$$z = h(y) = \int_{-\infty}^{y} p(\hat{y})d\hat{y}$$

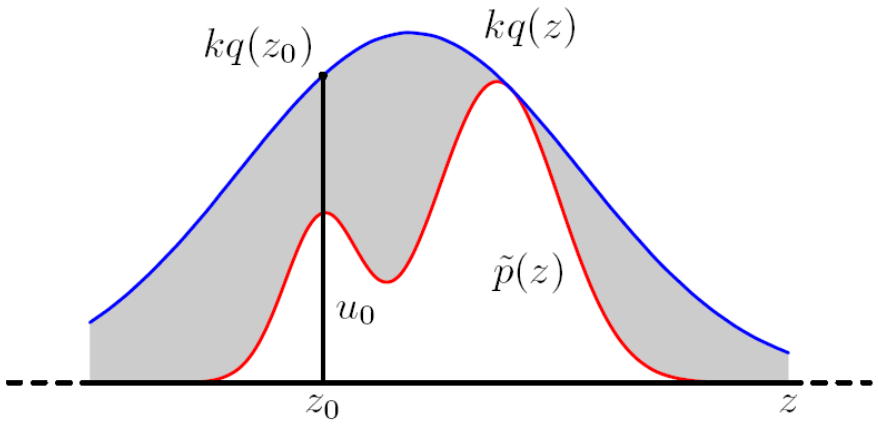
$$y = h^{-1}(z)$$

- So we have to transform uniformly distributed random numbers
  - using a function which is the inverse of the indefinite integral of the distribution

# Rejection Sampling

- Samples are drawn from simple distribution $q(z)$
- Rejected if they fall in grey area between $\tilde{p}(z)$ and $kq(z)$

$$p(z) = \frac{\tilde{p}(z)}{Z_p}$$




- Resulting samples are distributed according to *p(z)* which is the normalized version of $\tilde{p}(z)$

# Importance Sampling

- The expectation can be computed as

$$\mathbb{E}[f] = \int f(\mathbf{z})p(\mathbf{z})d\mathbf{z} = \int f(\mathbf{z})\frac{p(\mathbf{z})}{q(\mathbf{z})}q(\mathbf{z})d\mathbf{z}$$

- Use Monte Carlo methods

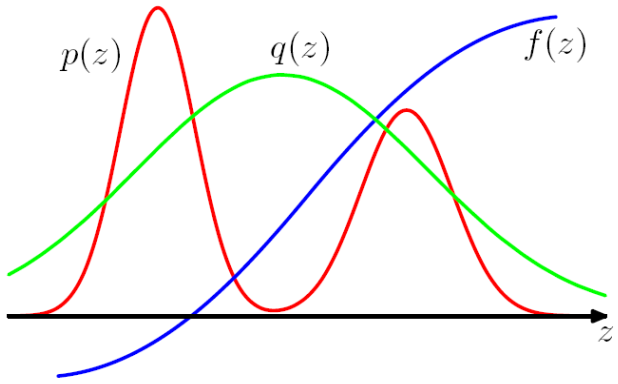
$$\mathbb{E}[f] \approx \frac{1}{L}\sum_{l=1}^{L} r_l f(\mathbf{z}^{(l)})$$

  - where the importance weights are

$$r_l = \frac{p(\mathbf{z}^{(l)})}{q(\mathbf{z}^{(l)})}$$

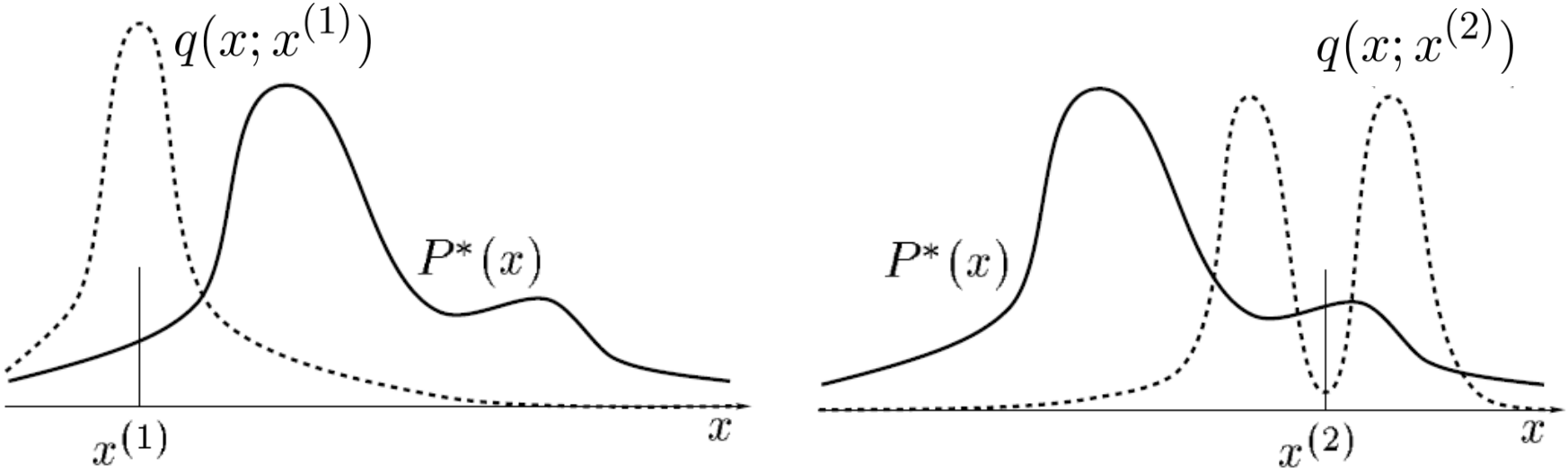
  - and the samples are

$$\mathbf{z}^{(l)} \sim q(\mathbf{z})$$



# Geometry of MCMC

- Proposal depends on current state
- Not necessarily similar to the target
- Can evaluate the un-normalized target



# Metropolis Algorithm

- Proposal distribution is symmetric

$$q(\mathbf{z}|\mathbf{z}') = q(\mathbf{z}'|\mathbf{z})$$

- The candidate sample $\mathbf{z}^*$ is accepted with probability

$$A(\mathbf{z}^*, \mathbf{z}^{(\tau)}) = \min\left(1, \frac{\tilde{p}(\mathbf{z}^*)}{\tilde{p}(\mathbf{z}^{(\tau)})}\right)$$

  - The acceptance can be done by
    - draw a random $u \sim \text{Uniform}(0, 1)$
    - accepting the sample if $A(\mathbf{z}^*, \mathbf{z}^{(\tau)}) > u$

- If sample is accepted, set $\mathbf{z}^{(\tau+1)} = \mathbf{z}^*$ ; otherwise $\mathbf{z}^{(\tau+1)} = \mathbf{z}^{(\tau)}$

- **Note:** $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots$ is not a set of independent samples

# Gibbs Sampling

- A special case of Metropolis-Hastings algorithm
- Consider the distribution $p(\mathbf{z}) = p(z_1, \ldots, z_M)$

- Gibbs sampling performs the follows
  - Initialize $\{z_i : i = 1, \ldots, M\}$
  - For $\tau = 1, \ldots, T$
    - Sample $z_1^{(\tau+1)} \sim p(z_1 | z_2^{(\tau)}, z_3^{(\tau)}, \ldots, z_M^{(\tau)})$

      $\vdots$
    - Sample $z_j^{(\tau+1)} \sim p(z_j | z_1^{(\tau+1)}, \ldots, z_{j-1}^{(\tau+1)}, z_{j+1}^{(\tau)}, \ldots, z_M^{(\tau)})$

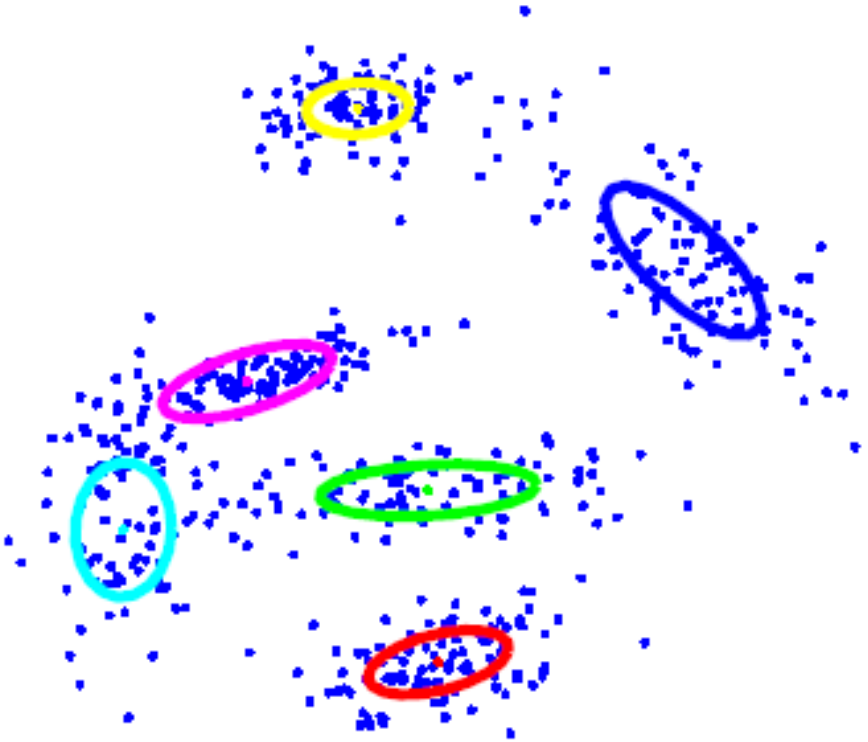
      $\vdots$
    - Sample $z_M^{(\tau+1)} \sim p(z_j | z_1^{(\tau+1)}, z_2^{(\tau+1)}, \ldots, z_{M-1}^{(\tau+1)})$

# Today, we learn

- Latent feature models
- Indian buffet process

# Clustering

- **Basic idea**: each data point belongs to a cluster



# Why latent features?

- Many statistical models can be thought of as modeling data in terms of hidden or latent variables.
- Clustering algorithms (e.g. using mixture models) represent data in terms of which cluster each data point belongs to.
- But clustering models are restrictive…
- Consider modeling people's movie preferences (the "Netflix" problem). A movie might be described using features such as "is science fiction", "has Charlton Heston", "was made in the US", "was made in 1970s", "has apes in it"… these features may be unobserved (latent).
- The number of potential latent features for describing a movie (or person, news story, image, gene, speech waveform, etc) is unlimited.

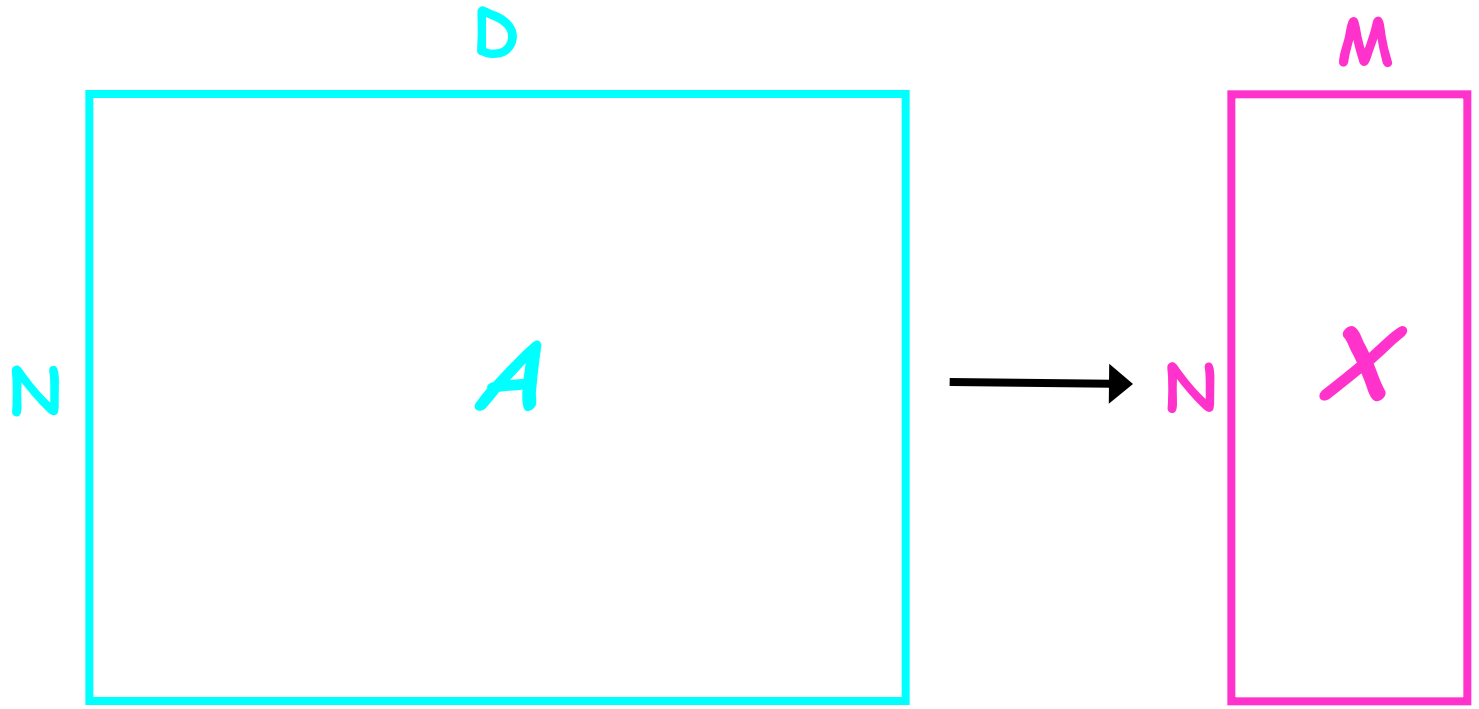
# Latent Feature Models are not New …

- PCA
- ICA
- LDA (latent discriminant analysis)
- LSI
- Neural networks

- Topic models
  - A special case with some constraints (e.g., conservation of belief constraint)

# Data Reduction

- summarization of data with many (D) variables by a smaller set of (M) derived (synthetic, composite) variables.



# PCA: Principal Component Analysis

- probably the most widely-used and well-known of the "standard" multivariate methods
- invented by Pearson (1901) and Hotelling (1933)
- first applied in ecology by Goodall (1954) under the name "factor analysis" ("principal factor analysis" is a synonym of PCA).

# PCA

- a way to describe multivariate data by maximizing variances
- a useful tool for data compression and information extraction
- to find combinations of variables (factors) that describe major trends in the data
- an eigenvalue decomposition process of the covariance (correlation) matrix of the variables

# Steps of PCA after Seeing a Set of Points

- Minus the empirical mean to get centered data
- Compute the covariance $$S = \frac{1}{N} \sum_n \mathbf{x}_n \mathbf{x}_n^\top$$
- Doing eigenvalue decomposition
  - Let $U$ be the eigenvectors of $S$ corresponding to the top d eigenvalues
- Encode data $Y = U^\top X$
- Reconstruct data $\hat{X} = UY = UU^\top X$

# Theory of PCA

- There are three types of interpretation

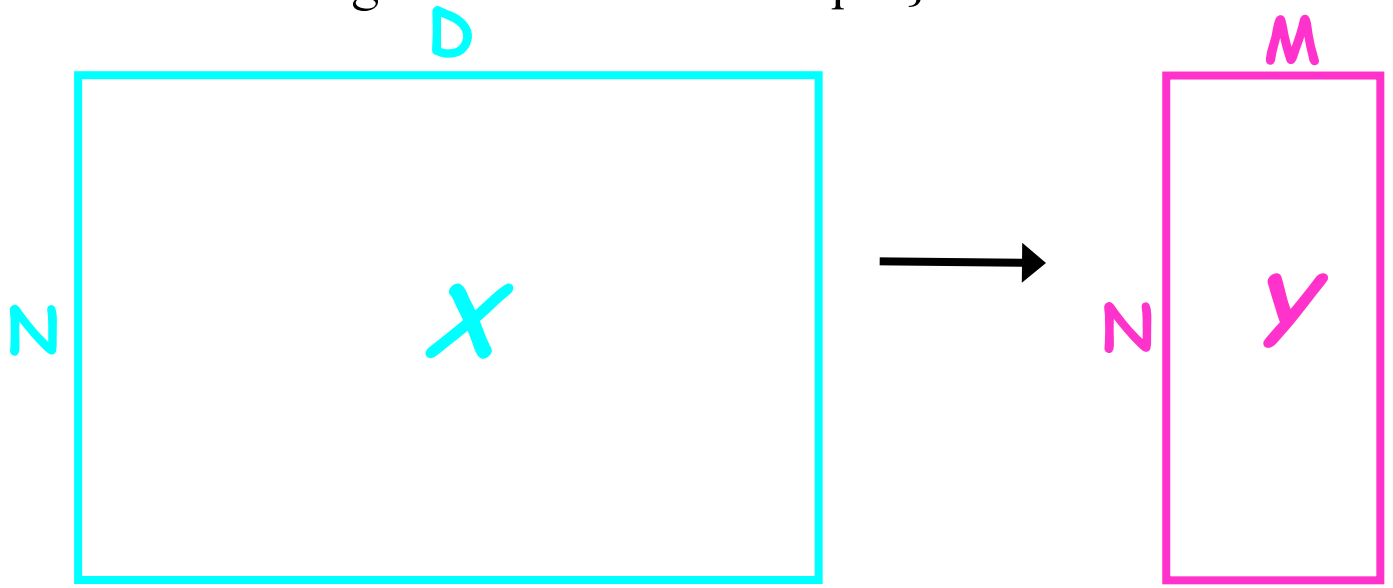
- Minimum variance
- Least reconstruct error
- Probabilistic model

# Maximum Variance Formulation

- Given a set of data points $\{\mathbf{x}_n\},\ n = 1, \ldots, N$

$$\mathbf{x}_n \in \mathbb{R}^D$$

- **Goal**:
  - Project the data into an *M*-dimensional (M < D) space while maximizing the variance of the projected data



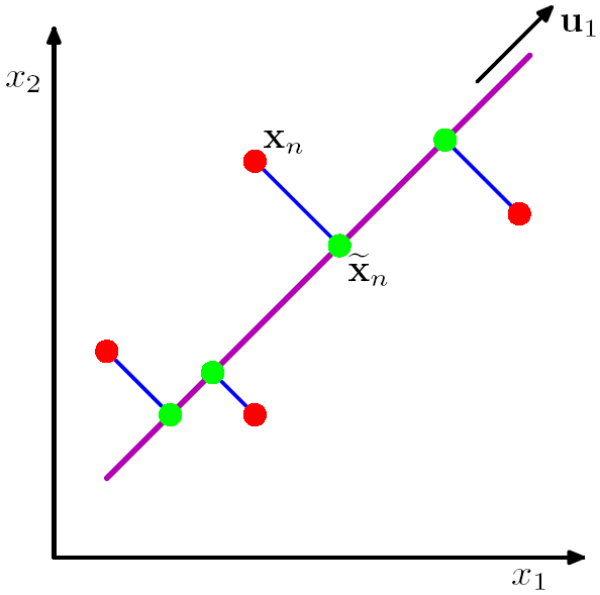
# Maximum Variance Formulation

- Let's start with the 1-dimensional projection, i.e., M = 1
- We only care about the projection direction, not the scale, so we assume

$$\mu_1^\top \mu_1 = 1$$

- The projection is

$$y_n = \mu_1^\top \mathbf{x}_n$$




- Mean and variance of projected data:

$$\bar{y} = \mu_1^\top \bar{\mathbf{x}}, \quad \text{where } \bar{\mathbf{x}} = \frac{1}{N}\sum_n \mathbf{x}_n$$

$$\text{var}(y) = \frac{1}{N}\sum_n (\mu_1^\top \mathbf{x}_n - \mu_1^\top \bar{\mathbf{x}})^2 = \mu_1^\top S \mu_1$$

  - sample covariance $S = \frac{1}{N}\sum_n (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top$

# Maximum Variance Formulation

- Now, we get a constrained optimization problem

$$\max_{\mu_1} \quad \text{var}(y) = \frac{1}{N}\sum_n (\mu_1^\top \mathbf{x}_n - \mu_1^\top \bar{\mathbf{x}})^2 = \mu_1^\top S \mu_1$$

  - where $\mu_1^\top \mu_1 = 1$

- Solve it using Lagrangian methods, we get
  - The eigenvector problem

$$S\mu_1 = \lambda_1 \mu_1$$

  - The lagrange multiplier is the eigenvalue

$$\mu_1^\top S \mu_1 = \lambda_1$$

  - The eigenvector corresponds to largest eigenvalue is 1st PC.

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_n \mathbf{x}_n \qquad S = \frac{1}{N}\sum_n (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top$$

# Maximum Variance Formulation

- Additional components can be incrementally found

$$\max_{\mu_2} \quad \text{var}(y) = \frac{1}{N}\sum_n (\mu_2^\top \mathbf{x}_n - \mu_2^\top \bar{\mathbf{x}})^2 = \mu_2^\top S \mu_2$$

  - where $\mu_2^\top \mu_2 = 1$ and $\mu_1^\top \mu_2 = 0$

- Solve this problem with Lagrangian method, we have

$$2S\mu_2 - 2\lambda_2\mu_2 + \gamma\mu_1 = 0$$

  - which leads to

$$S\mu_2 - \lambda_2\mu_2 - \gamma\mu_1 = 0$$

  - Left multiplying $\mu_1^\top$, we get (remember $\mu_1$ is eigenvector)

$$\gamma = \mu_1^\top S\mu_2 = \lambda_1\mu_1^\top\mu_2 = 0$$

  - Thus, $S\mu_2 = \lambda_2\mu_2$ $\quad$ $\mu_2^\top S\mu_2 = \lambda_2$

# Minimum Error Formulation

- A set of complete orthonormal basis

$$\{\mu_i\}, \ i = 1, \ldots, D$$

$$\mu_i^\top \mu_j = \delta_{ij}$$

- Each data point can be represented as

$$\mathbf{x}_n = \sum_i \alpha_{ni} \mu_i$$

  - Due to the orthonormal property, we can get

$$\alpha_{ni} = \mathbf{x}_n^\top \mu_i$$

$$\mathbf{x}_n = \sum_i (\mathbf{x}_n^\top \mu_i) \mu_i$$

# Minimum Error Formulation

- A set of complete orthonormal basis

$$\{\mu_i\}, \ i = 1, \ldots, D$$

$$\mu_i^\top \mu_j = \delta_{ij}$$

- We consider a low-dimensional approximation

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni}\mu_i + \sum_{i=M+1}^{D} b_i \mu_i$$

- The best approximation is to minimize the error

$$J = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$

# Minimum Error Formulation

- A set of complete orthonormal basis

$$\{\mu_i\},\ i = 1, \ldots, D \qquad \mu_i^\top \mu_j = \delta_{ij}$$

- The best approximation is to minimize the error

$$J = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2 \qquad \tilde{\mathbf{x}}_n = \sum_{i=1}^{M} z_{ni}\mu_i + \sum_{i=M+1}^{D} b_i \mu_i$$

  - we get

$$z_{ni} = \mathbf{x}_n^\top \mu_i,\ i = 1, \ldots, M \qquad b_i = \bar{\mathbf{x}}^\top \mu_i,\ i = M+1, \ldots, D$$

  - use the general representation $\mathbf{x}_n = \sum_i (\mathbf{x}_n^\top \mu_i)\mu_i$
  - we get that the displacement lines in the orthogonal space

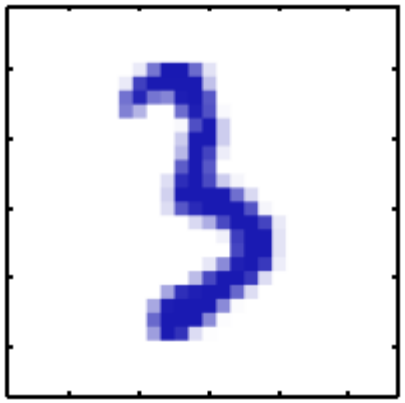
$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=M+1}^{D} \{(\mathbf{x}_n - \bar{\mathbf{x}})^\top \mu_i\}\mu_i$$

# Minimum Error Formulation

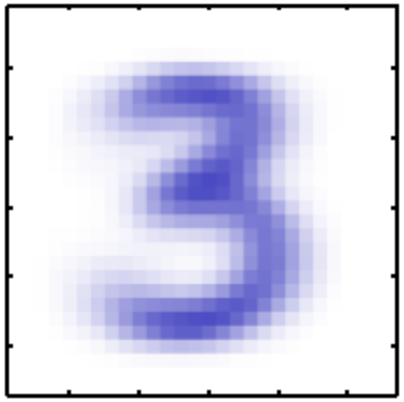
- With the result $$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=M+1}^{D} \{(\mathbf{x}_n - \bar{\mathbf{x}})^\top \mu_i\}\mu_i$$
- We get the error
$$J = \frac{1}{N}\sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2 = \frac{1}{N}\sum_{n}\sum_{i=M+1}^{D} (\mathbf{x}_n^\top \mu_i - \bar{\mathbf{x}}^\top \mu_i)^2$$
$$= \sum_{i=M+1}^{D} \mu_i^\top S \mu_i$$
- The optimization problem
$$\min_{\mu_i} J$$
  - where $\mu_i^\top \mu_i = 1$
  - Solutions are eigenvectors and $J = \sum_{i=M+1}^{D} \lambda_i$

# An Example

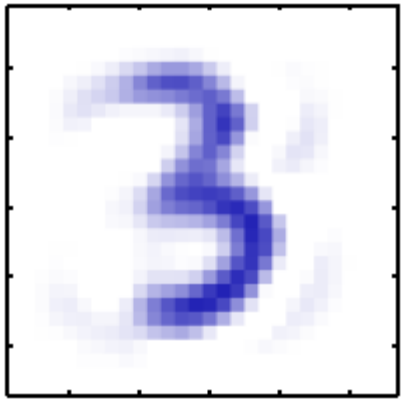
Original



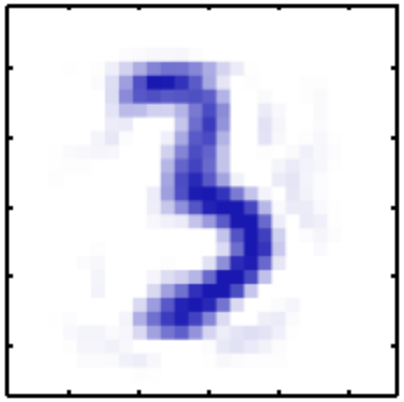
$M = 1$



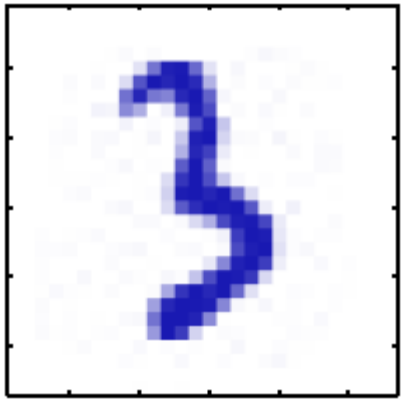
$M = 10$



$M = 50$



$M = 250$



# High-dimensional Data

- When $D$ is very large, the number of data points $N$ may be smaller than the dimension. It is better to use the $N \times N$ Gram matrix instead of the $D \times D$ covariance matrix.
- For centered data, we have
  - $S = \frac{1}{N} X X^\top$ with eigenvectors and eigenvalues $(\lambda_i, \mu_i)$
  - $G = X^\top X$ with eigenvectors and eigenvalues $(\gamma_i, \nu_i)$

- By left-multiplying $X^\top$ to $X X^\top \mu_i = N \lambda_i \mu_i$ , we get

$$X^\top X (X^\top \mu_i) = N \lambda_i (X^\top \mu_i) \quad , \quad \nu_i = X^\top \mu_i \quad \text{and} \quad \gamma_i = N \lambda_i$$

- Thus,

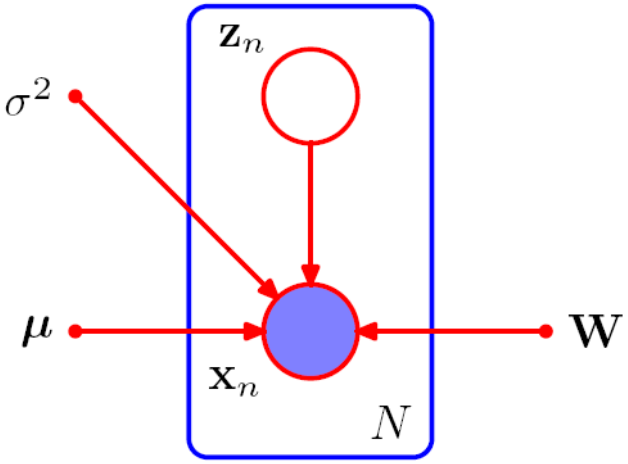
$$X \nu_i = X X^\top \mu_i = N \lambda_i \mu_i = \gamma_i \mu_i$$

$$\mu_i = \frac{1}{\gamma_i} X \nu_i$$

# Probabilistic PCA

- A simple linear-Gaussian model
- Let z be a latent feature vector $\mathbf{z} \in \mathbb{R}^M$
  - In Bayesian, we assume it's prior $\mathbf{z} \sim \mathcal{N}(0, I)$
- A linear-Gaussian model

$$\mathbf{x} = W\mathbf{z} + \mu + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \sigma^2 I)$$




  - this gives the likelihood

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|W\mathbf{z} + \mu, \sigma^2 I)$$

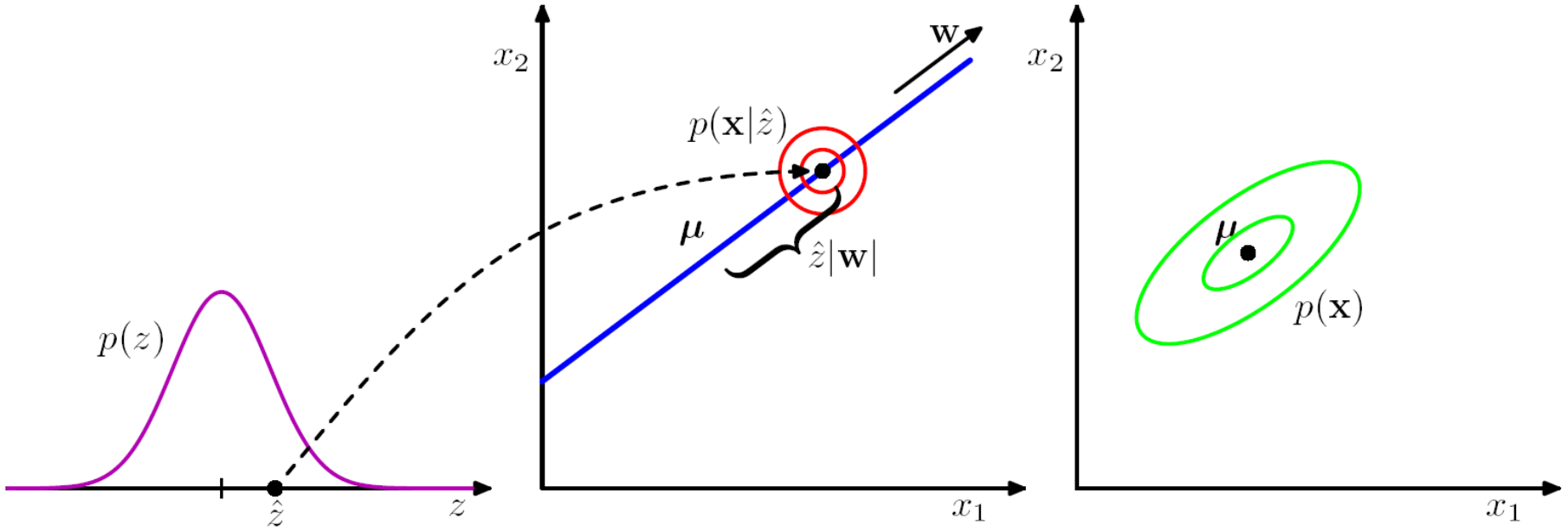
  - the columns of W span a linear subspace

# Probabilistic PCA

- By the properties of Gaussian, we can get the marginal

$$p(\mathbf{x}) = \int p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z} = \mathcal{N}(\mathbf{x}|\mu, C)$$

$$C = WW^{\top} + \sigma^2 I$$



# Probabilistic PCA

- By the properties of Gaussian, we can get the conditional

$$p(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|M^{-1}W^{\top}(\mathbf{x}-\mu), \sigma^{-2}M)$$

$$M = W^{\top}W + \sigma^{2}I$$

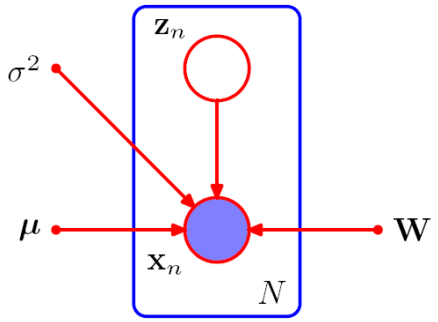
  - The posterior mean depends on x
  - Posterior covariance is independent of x

# Maximum Likelihood PCA

- Given a set of observations $X = \{\mathbf{x}_n\}$, the log-likelihood is

$$\log p(X|\mu, W, \sigma^2) = \sum_n \log p(\mathbf{x}_n|W, \mu, \sigma^2)$$

$$= -\frac{ND}{2}\log(2\pi) - \frac{N}{2}\log|C| - \frac{1}{2}\sum_n (\mathbf{x}_n - \mu)^\top C^{-1}(\mathbf{x}_n - \mu)$$




- We get the MLE: $\hat{\mu} = \bar{\mathbf{x}}$ and

$$\log p(X|\mu, W, \sigma^2) = -\frac{N}{2}\left(D\log(2\pi) + \log|C| + \mathrm{Tr}(C^{-1}S)\right)$$

  - The stationary points can be written as (Tipping & Bishop, 1999)

$$\hat{W} = U_M(L_M - \sigma^2 I)^{1/2}R$$

  - $L_M$ is diagonal with eigenvalues $\lambda_i$ ; $R$ is MxM orthogonal matrix; $U_M$ is DxM matrix whose columns are eigenvectors

# Maximum Likelihood PCA

- Recover the conventional PCA
  - Take the limit $\sigma^2 \to 0$ , we get the posterior mean

$$M^{-1}W^{\top}(\mathbf{x}-\mu) = (\hat{W}^{\top}\hat{W})^{-1}\hat{W}^{\top}(\mathbf{x}-\bar{\mathbf{x}})$$

  - which is an orthogonal projection of the data point into the latent space
  - So we recover the standard PCA

# EM Algorithm for PPCA

- **E-step**: evaluate expectation of complete likelihood

$$\mathbb{E}[\log p(X, Z|\Theta)] = -\sum_n \left\{ \frac{D}{2}\log(2\pi\sigma^2) + \frac{1}{2}\mathrm{Tr}(\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^\top]) \right.$$

$$\left. + \frac{1}{2\sigma^2}\|\mathbf{x}_n - \mu\|^2 - \frac{1}{\sigma^2}\mathbb{E}[\mathbf{z}_n]^\top W^\top(\mathbf{x}_n - \mu) + \frac{1}{2\sigma^2}\mathrm{Tr}(\mathbb{E}[\mathbf{x}_n\mathbf{z}_n^\top]W^\top W) \right\}$$

  - where

$$\mathbb{E}[\mathbf{z}_n] = M^{-1}W^\top(\mathbf{x}_n - \bar{\mathbf{x}})$$

$$\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^\top] = \sigma^2 M^{-1} + \mathbb{E}[\mathbf{z}_n]\mathbb{E}[\mathbf{z}_n]^\top$$

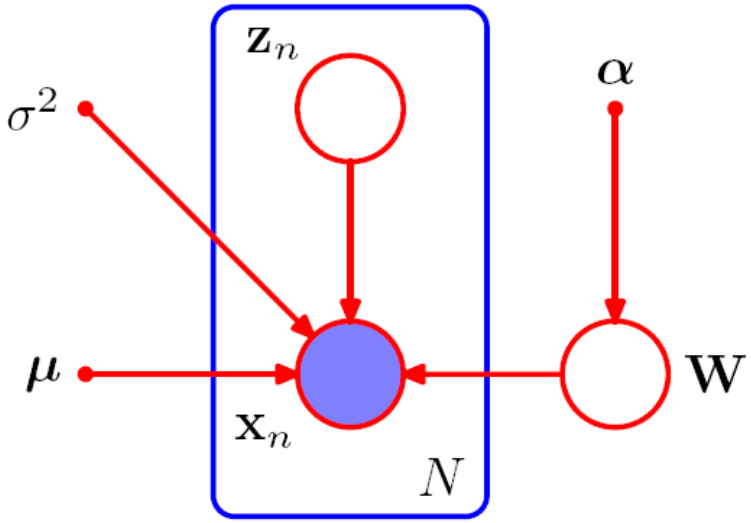
- **M-step**: optimizes over parameters

$$W = \left[\sum_n (\mathbf{x}_n - \bar{\mathbf{x}})\mathbb{E}[\mathbf{z}_n]\right]^\top \left[\sum_n \mathbb{E}[\mathbf{z}_n\mathbf{z}_n^\top]\right]^{-1}$$

$$\sigma^2 = \frac{1}{ND}\sum_n \left\{ \|\mathbf{x}_n - \bar{\mathbf{x}}\|^2 - 2\mathbb{E}[\mathbf{z}_n]^\top W^\top(\mathbf{x}_n - \bar{\mathbf{x}}) + \mathrm{Tr}(\mathbb{E}[\mathbf{z}_n\mathbf{z}_n^\top]W^\top W) \right\}$$

# Bayesian PCA

- A prior is assumed on the parameters W



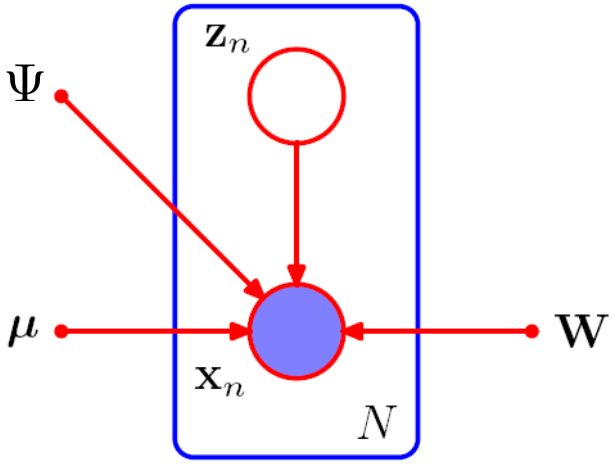

$$p(W|\alpha) = \prod_{i=1}^{M} \left(\frac{\alpha_i}{2}\right)^{D/2} \exp\left\{-\frac{1}{2}\alpha_i \mathbf{w}_i^\top \mathbf{w}_i\right\}$$

- Inference can be done in closed-form, as in GP regression
- Fully Bayesian treatment put priors on $\mu, \sigma^2, \alpha$

# Factor Analysis

- Another simple linear-Gaussian model
- Let z be a latent feature vector $\mathbf{z} \in \mathbb{R}^M$
  - In Bayesian, we assume it's prior $\mathbf{z} \sim \mathcal{N}(0, I)$
- A linear-Gaussian model

$$\mathbf{x} = W\mathbf{z} + \mu + \epsilon \qquad \epsilon \sim \mathcal{N}(0, \Psi)$$




  - $\Psi$ is a diagonal matrix
  - this gives the likelihood

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|W\mathbf{z} + \mu, \Psi)$$

  - the columns of W span a linear subspace

# Factor Analysis

- We can the inference tasks almost the same as in PCA
- The predictive distribution is Gaussian
- EM algorithm can be applied to maximum likelihood estimation

# Kernel PCA

- PCA can be generalized to non-linear principal components
- Read the Chapter 12 of Bishop's book

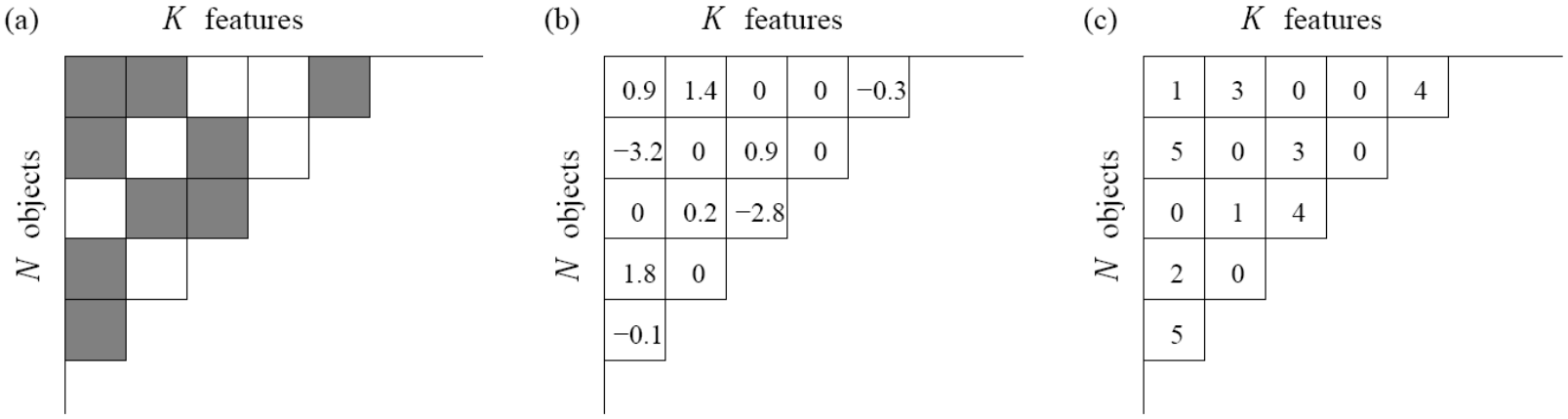
# Model Selection Issue

- How to decide the latent dimension?

- We will present a non-parametric technique to automatically infer the latent dimension

# Latent Feature Models

- Consider $N$ objects, the latent features form a matrix
- The feature matrix can be decomposed into two components
  - A binary matrix $Z$ indicating which features possessed by each object
  - A matrix V indicating the value of each feature for each object



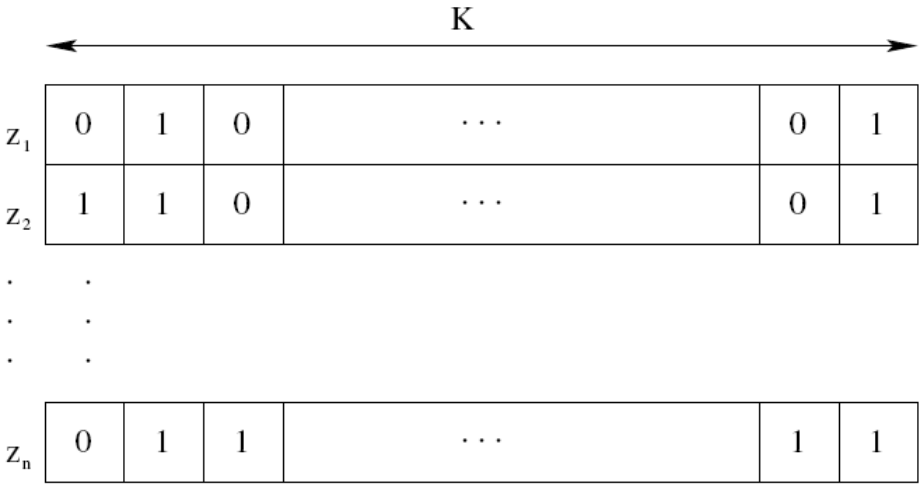

- Sparsity is imposed on the binary matrix $Z$
- For Bayesian, the prior can be imposed as $p(F) = p(Z)p(V)$
- We will focus on $p(Z)$, which determines the effective dimensionality of latent features

# A Finite Latent Feature Model

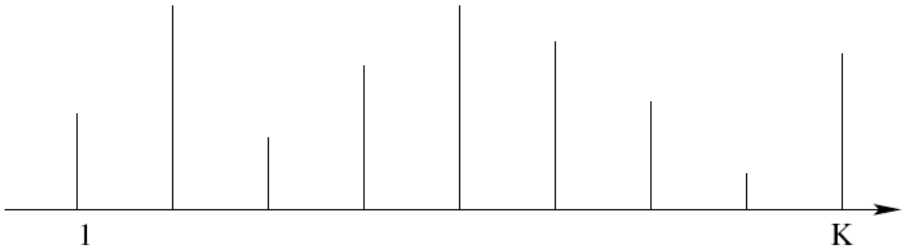
- A random finite binary latent feature model

$$\pi_k|\alpha \sim \text{Beta}(\frac{\alpha}{K}, 1)$$

$$z_{ik}|\pi_k \sim \text{Bernoulli}(\pi_k)$$




- $\pi_k$ is the relative probability of each feature being on, e.g.,



# A Finite Latent Feature Model

- The marginal probability of a binary matrix Z is

$$p(Z) = \prod_{k=1}^{K} \int \Big( \prod_{i=1}^{N} p(z_{ik}|\pi_k) \Big) p(\pi_k) d\pi_k$$

$$= \prod_{k=1}^{K} \int \Big( \prod_{i=1}^{N} \pi_k^{z_{ik}} (1-\pi_k)^{1-z_{ik}} \Big) p(\pi_k) d\pi_k$$

$$m_k = \sum_{i=1}^{N} z_{ik}$$

$$= \prod_{k=1}^{K} \int \pi_k^{m_k} (1-\pi_k)^{N-m_k} p(\pi_k) d\pi_k$$

$$= \prod_{k=1}^{K} \frac{\frac{\alpha}{K}\Gamma(m_k + \frac{\alpha}{K})\Gamma(N - m_k + 1)}{\Gamma(N + 1 + \frac{\alpha}{K})}$$

Features are independent!

$$\int_0^1 \pi^{r-1}(1-\pi)^{s-1} d\pi = \frac{\Gamma(r)\Gamma(s)}{\Gamma(r+s)}$$

# A Finite Latent Feature Model

- The conditional probability of each feature assignment

$$p(z_{ik}=1|Z_{-(i,k)}) = p(z_{ik}=1|\mathbf{z}_{-(i,k)}) = \frac{p(z_{ik}=1,\mathbf{z}_{-(i,k)})}{p(\mathbf{z}_{-(i,k)})}$$

$$p(z_{ik}=1,\mathbf{z}_{-(i,k)}) = \frac{\Gamma\left((m_{-(i,k)}+1)+\frac{\alpha}{K}\right)\Gamma\left(N-(m_{-(i,k)}+1)+1\right)}{\Gamma(N+\frac{\alpha}{K}+1)}$$

$$p(\mathbf{z}_{-(i,k)}) = \frac{\Gamma(m_{-(i,k)}+\frac{\alpha}{K})\Gamma\left((N-1)-m_{-(i,k)+1}\right)}{\Gamma\left((N-1)+\frac{\alpha}{K}+1\right)}$$

$$p(z_{ik}=1|Z_{-(i,k)}) = \frac{m_{-(i,k)}+\frac{\alpha}{K}}{N+\frac{\alpha}{N}}$$

$$p(\mathbf{z}_k) = \frac{\frac{\alpha}{K}\Gamma(m_k+\frac{\alpha}{K})\Gamma(N-m_k+1)}{\Gamma(N+1+\frac{\alpha}{K})}$$

$$m_k = \sum_{i=1}^{N} z_{ik} \qquad \Gamma(x+1) = x\Gamma(x)$$

# A Finite Latent Feature Model

- Expectation of the number of non-zero features

$$\mathbb{E}[1^\top Z 1] = \mathbb{E}\Big[\sum_{ik} z_{ik}\Big] = K\mathbb{E}[1^\top \mathbf{z}_k]$$

  - the last equality is due to the independence of the features

- For feature k, we have

$$\mathbb{E}[1^\top \mathbf{z}_k] = \sum_{i=1}^{N} \mathbb{E}[z_{ik}] = \sum_{i=1}^{N} \int_0^1 \pi_k p(\pi_k) d\pi_k = N \frac{\frac{\alpha}{K}}{1+\frac{\alpha}{K}}$$

  - The last equality is due to the fact that expectation of Beta(r, s) is $\frac{r}{r+s}$

  - Thus, $\mathbb{E}[1^\top Z 1] = \frac{N\alpha}{1+\frac{\alpha}{K}} < N\alpha$

# From Finite to Infinite

- A technical difficulty: the probability for any particular matrix goes to zero as $K \to \infty$

$$\lim_{K\to\infty} p(Z|\alpha) = 0$$

- However, if we consider equivalence classes of matrices in left-ordered form obtained by reordering the columns:

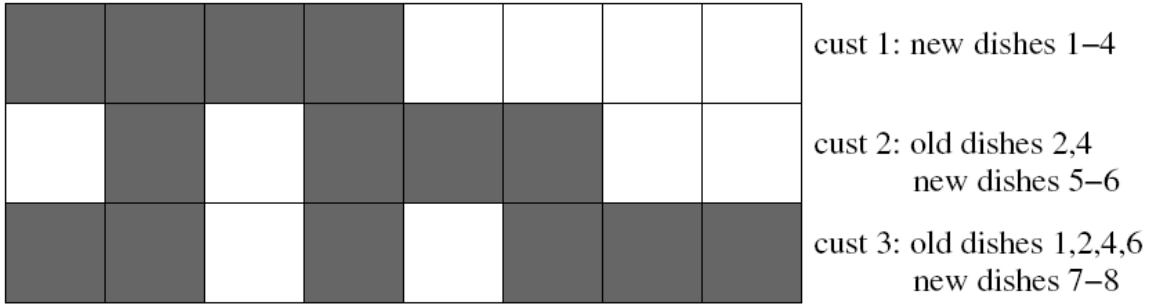
$$\lim_{K\to\infty} p([Z]|\alpha) = \exp\{-\alpha H_N\} \frac{\alpha^{K_+}}{\prod_{h>0} K_h!} \prod_{k\le K_+} \frac{(N-m_k)!(m_k-1)!}{N!}$$

  - $K_+$ is the number of features assigned (i.e. non-zero columns).
  - $H_N = \sum_{n=1}^{N} \frac{1}{n}$ is the Nth harmonic number.
  - $K_h$ are the number of features with history h (a technicality).



# Indian Buffet Process

- A stochastic process on infinite binary feature matrices
- Generative procedure:
  - Customer 1 chooses the first $K_1$ dishes: $K_1 \sim \text{Poisson}(\alpha)$
  - Customer *i* chooses:
    - Each of the existing dishes with probability $\frac{m_k}{i}$
    - $K_i$ additional dishes, where $K_i \sim \text{Poisson}(\frac{\alpha}{i})$




$$Z_{i.} \sim \mathcal{IBP}(\alpha)$$

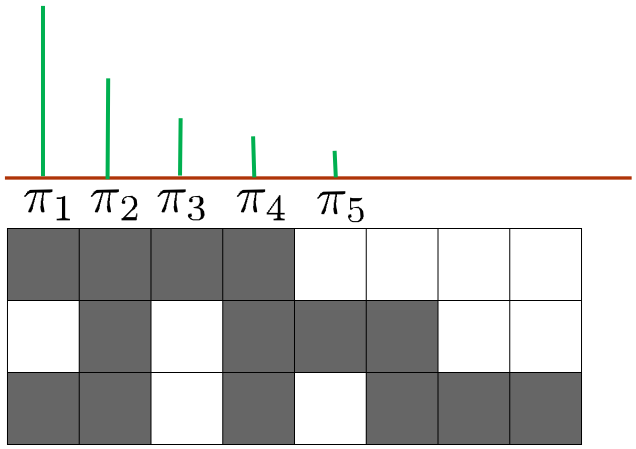
# Indian Buffet Process

- A stochastic process on infinite binary feature matrices
- Stick-breaking construction: $Z_i \sim \mathcal{IBP}(\alpha)$

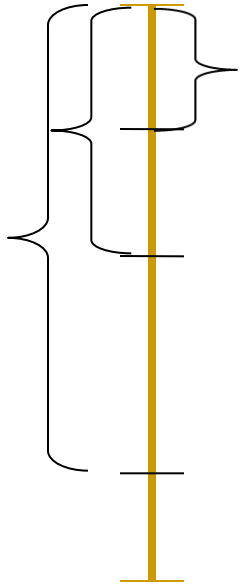
$$z_{nk} \sim \text{Bernoulli}(\pi_k)$$

$$\pi_i(\mathbf{v}) = v_i \pi_{i-1}(\mathbf{v}) = \prod_{j=1}^{i} v_j$$

$$v_i \sim \text{Beta}(\alpha, 1)$$




| $\prod_{j=1}^{i-1} v_j$ | $v_i$ | $\pi_i$ |
|---|---|---|
| 0 | 0.8 | 0.8 |
| 0.8 | 0.5 | 0.4 |
| 0.4 | 0.4 | 0.16 |



# Inference by Gibbs Sampling

- In the finite Beta-Bernoulli model, we have

$$p(z_{ik} = 1 | Z_{-(i,k)}) = \frac{m_{-(i,k)} + \frac{\alpha}{K}}{N + \frac{\alpha}{N}}$$

- Set limit $K \to \infty$ , we have the conditional for infinite model

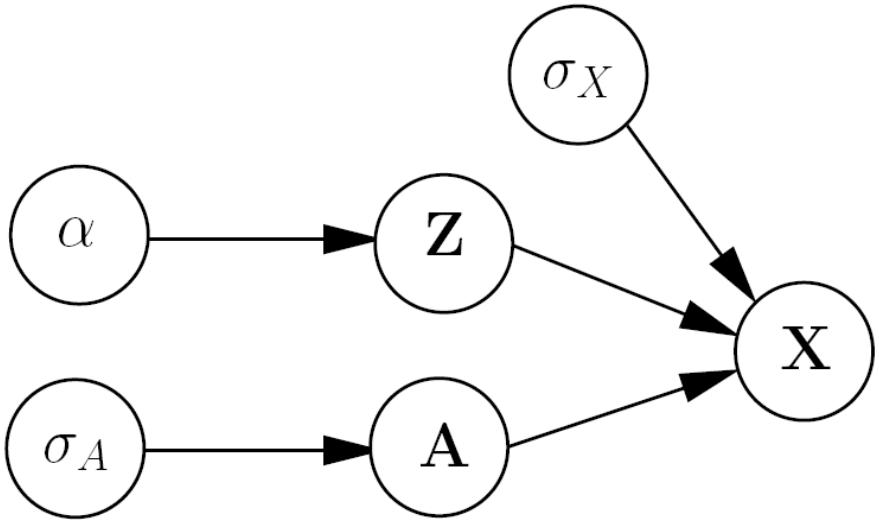
$$p(z_{ik} = 1 | Z_{-(i,k)}) = \frac{m_{-(i,k)}}{N}$$

  - for any k such that $m_{-(i,k)} > 0$
  - The number of new features should be drawn from

$$\text{Poisson}\left(\frac{\alpha}{K}\right)$$

# Use with Data

- A linear-Gaussian model with binary features




  - Gaussian likelihood $p(X|Z, A, \sigma_X) = \mathcal{N}\left(ZA, \sigma_X^2 I\right)$
  - Gaussian prior $p(A|\sigma_A) = \mathcal{N}\left(0, \sigma_A^2 I\right)$

# Inference with Gibbs Sampling

- The posterior is

$$p(Z, A|X, \alpha) \propto p(X|Z)p(Z|\alpha)$$

- The conditional for each feature assignment

$$p(z_{nk} = 1|Z_{-(n,k)}, X, \alpha) \propto p(z_{nk} = 1|Z_{-(n,k)}, \alpha)p(X|Z)$$

  - If $m_{-(i,k)} > 0$ , $p(z_{ik} = 1|Z_{-(i,k)}) = \frac{m_{-(i,k)}}{N}$
  - For infinitely many k such that $m_{-(i,k)} = 0$ : Metropolis steps with truncation to sample from the number of new features for each object

- For linear-Gaussian model, $p(X|Z)$ can be computed

# Other Issues

- Sampling methods for non-conjugate models
- Variational inference with the stick-breaking representation of IBP

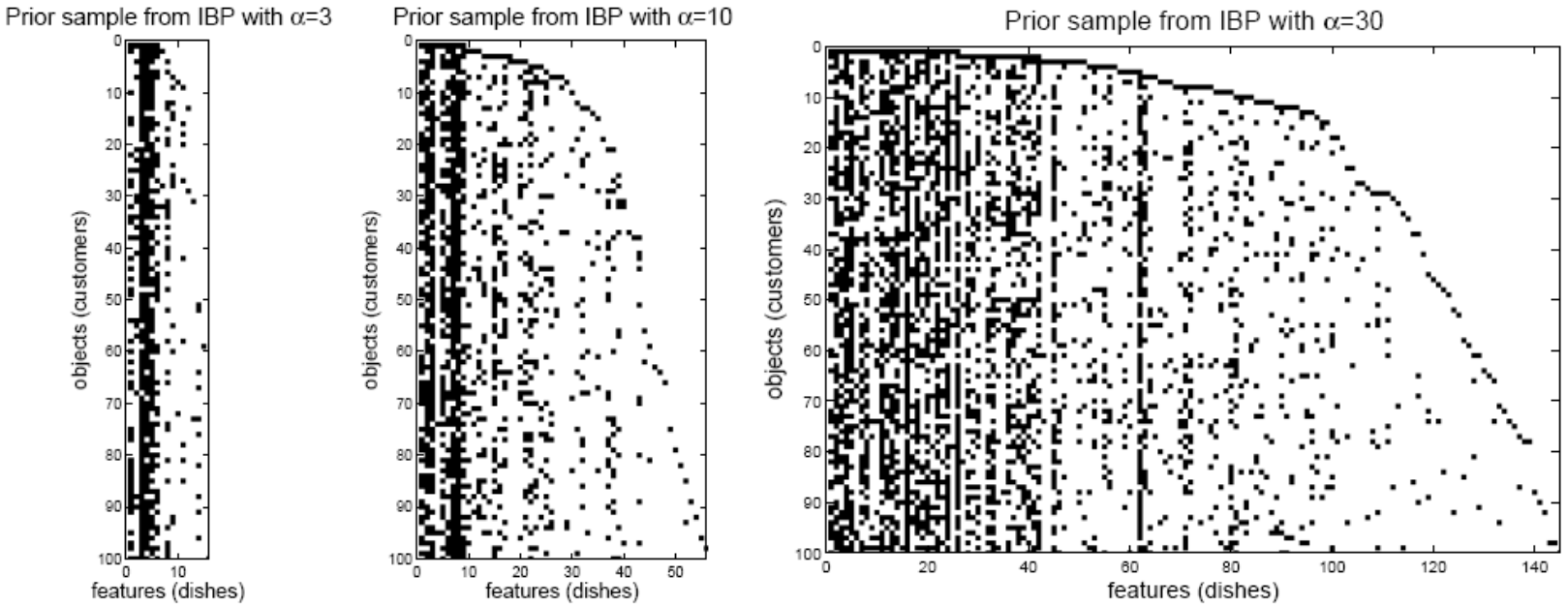
$$z_{nk} \sim \text{Bernoulli}(\pi_k)$$

$$\pi_i(\mathbf{v}) = v_i \pi_{i-1}(\mathbf{v}) = \prod_{j=1}^{i} v_j$$

$$v_i \sim \text{Beta}(\alpha, 1)$$

- Applications to various types of data
  - Graph structures, overlapping clusters, time series models

# A Two Parameter Generalization of IBP




- **Limitation**:
  - Hyperparameter $\alpha$ controls the # of features per object $K_n = \sum_k z_{nk} \sim \text{Poisson}(\alpha)$
  - But, it also controls the total # of features for a set with $N$ objects, i.e., the variability across rows of Z (limited variability)
  - This seems limited – we really want independent control over the mean number of features and the variability across rows

# A Two Parameter Generalization of IBP

**One-parameter IBP**

$$z_{nk} \sim \text{Bernoulli}(\theta_k)$$
$$\theta_k \sim \text{Beta}(\alpha/K, 1)$$

**Two-parameter IBP**

$$z_{nk} \sim \text{Bernoulli}(\theta_k)$$
$$\theta_k \sim \text{Beta}(\alpha\beta/K, \beta)$$

- Properties of the two-parameter IBP:
  - Number of features per object is $K_n = \sum_k z_{nk} \sim \text{Poisson}(\alpha)$
  - Setting $\beta = 1$ reduces to IBP
  - Parameter $\beta$ is feature repulsion, $1/\beta$ is feature stickiness
  - Total expected number of features is

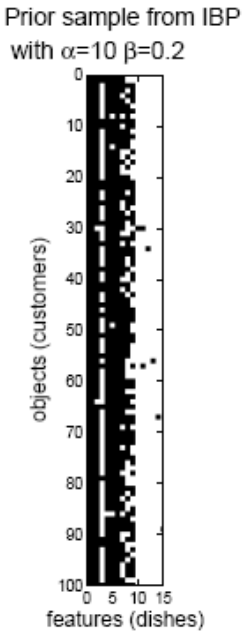
$$K_+ = \alpha \sum_{n=1}^{N} \frac{\beta}{\beta + n - 1} \to \alpha\beta \log N$$

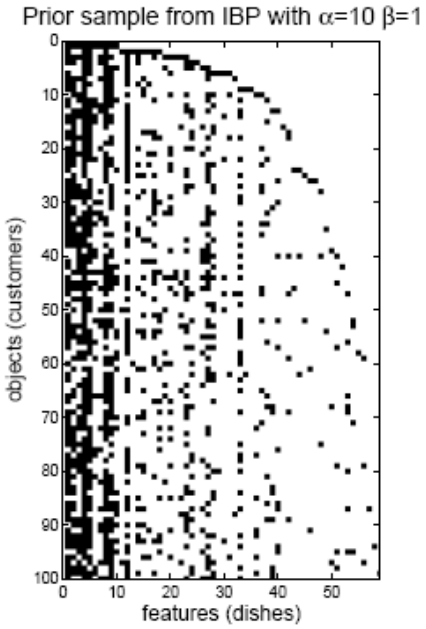
$$\lim_{\beta \to 0} K_+ = \alpha$$

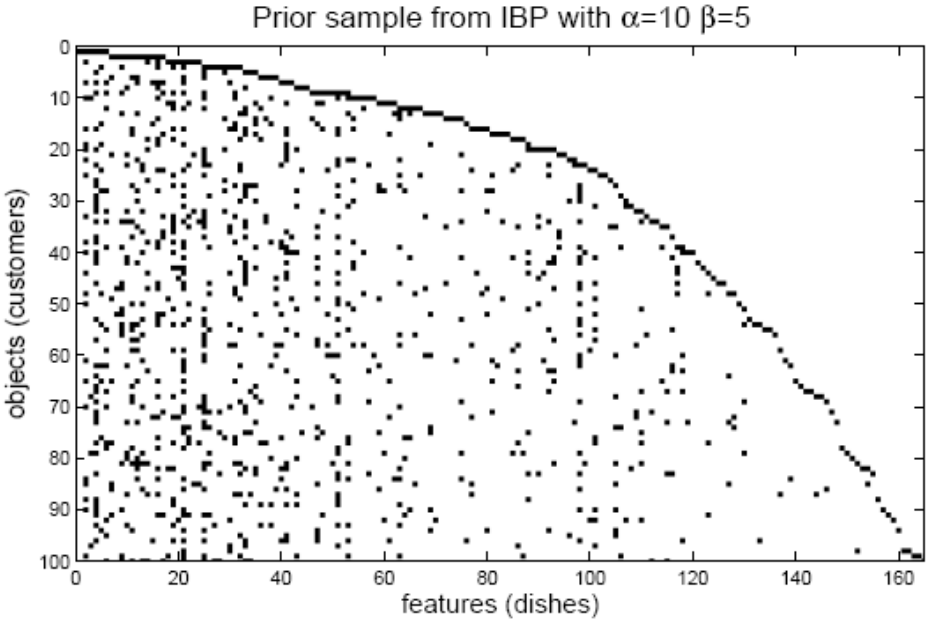
$$\lim_{\beta \to \infty} K_+ = N\alpha$$

# A Two Parameter Generalization of IBP

- Samples from the priors



# Markov Indian Buffet Process

- Let the $Z_{nk}$ have a Markov structure: e.g.,

$$p(Z_{nk} = 1 | Z_{n-1,k} = 0) = \theta^k_{0,1}$$

- This is useful for times series data, where we want the latent factors to turn on and off in a manner that depends on time

- Markov IBP (MIBP) defines such a process, which has IBP marginals (var Gael, Teh, Ghahramani, 2009)
  - MIBP can be used to generalize the Hidden Markov models

- More generally, we can have the IBP be dependent on covariates (dIBP) (Williamson, Ghahramani, 2010)

# Markov IBP and Time Series

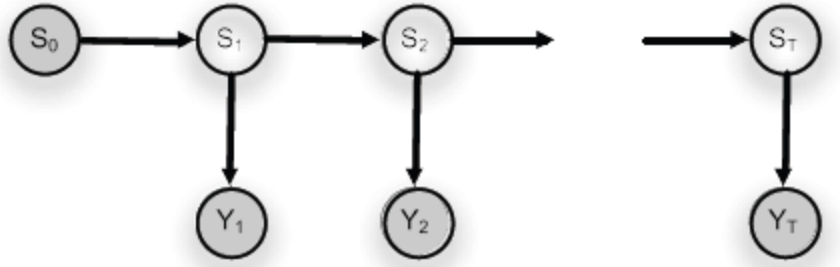



Figure 1: The Hidden Markov Model

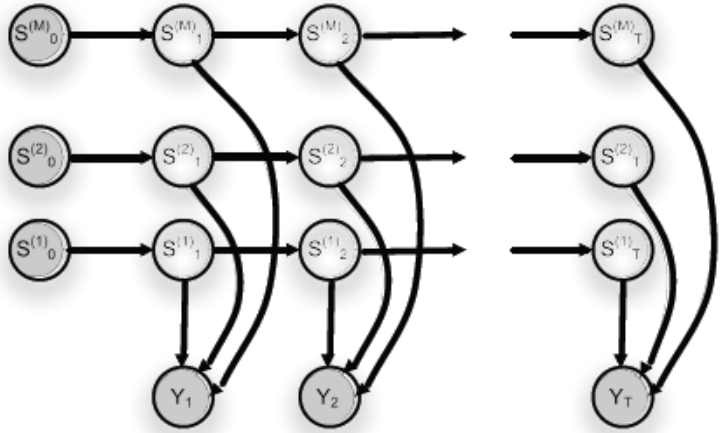



Figure 2: The Factorial Hidden Markov Model

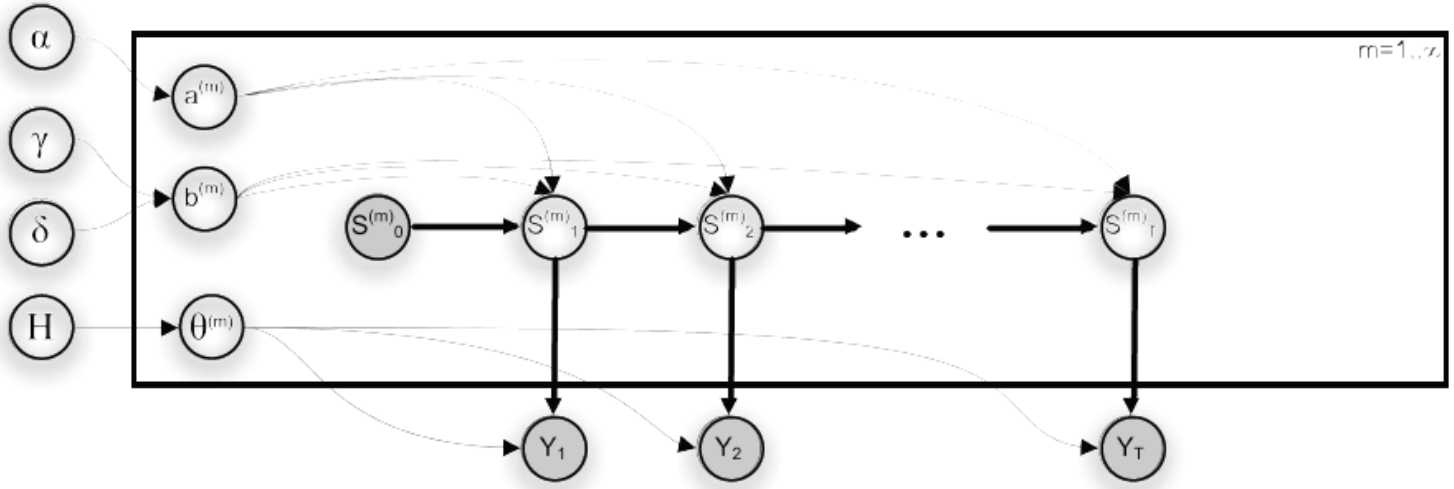
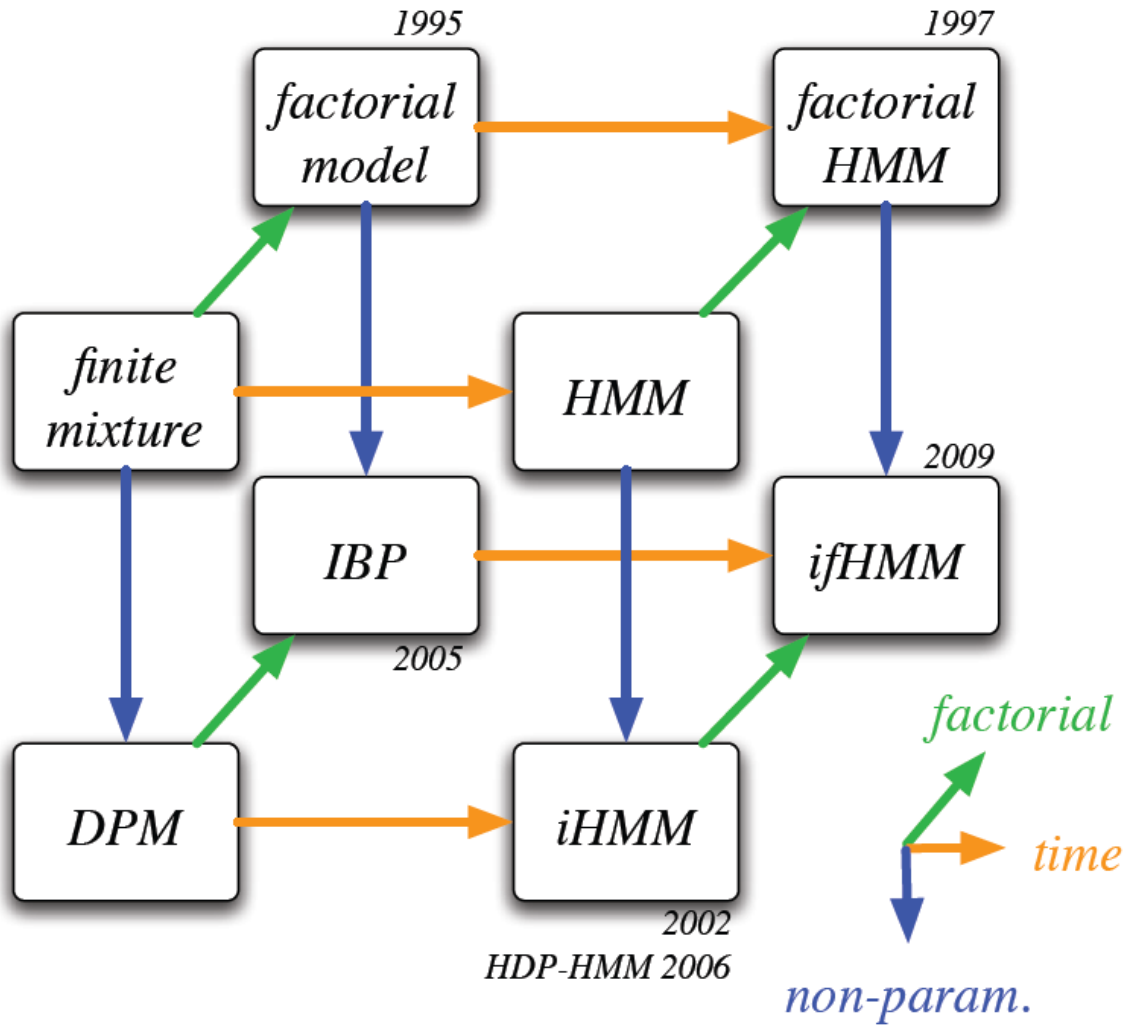



Figure 3: The Infinite Factorial Hidden Markov Model

# Big Picture


1995
factorial model
1997
factorial HMM
finite mixture
HMM
2009
IBP
ifHMM
2005
DPM
iHMM
2002
HDP-HMM 2006
factorial
time
non-param.

# References


- Chap. 13 of  Pattern Recognition and Machine Learning, Bishop, 2006
- Probabilistic Principal Component Analysis. M. Tipping, C. Bishop, 1999.
- Infinite Latent Feature Models and the Indian Buffet Process. T. Griffiths, Z. Ghahramani, 2005.
- The Indian Buffet Process: Scalable Inference and Extensions. Finale Doshi-Velez, 2009.