Bayesian Factor Analysis for Real-Valued Data

Mingyuan Zhou

IROM Department, McCombs School of Business
The University of Texas at Austin

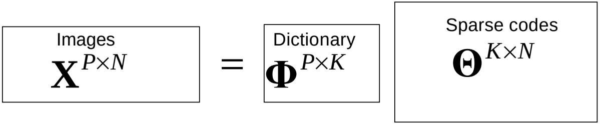
Duke-Tshinghua Machine Learning Summer School
Duke-Kushan University, Kunshan, China
August 02, 2016

Outline

- Preliminaries
  - Bayes' rule
  - likelihood, prior, posterior
  - hierarchical models
  - Markov chain Monte Carlo
  - Variational Bayes
- Factor analysis for real-valued data
  - Gibbs sampling
  - Variational Bayes
  - Collaborative filtering (matrix completion)

Outline

- Bayesian sparse factor analysis
  - Dictionary learning and sparse coding
  - Sparse priors on the factor scores
    - Spike-and-slab sparse prior
    - Beta-Bernoulli process, Indian buffet process
    - Bayesian Lasso shrinkage prior
  - Bayesian dictionary learning
    - Image denoising and inpainting
    - Introduce covariate dependence
    - Matrix completion



Bayes' rule

- In equation:

$$P(\boldsymbol{\theta} \,|\, X) = \frac{P(X \,|\, \boldsymbol{\theta})P(\boldsymbol{\theta})}{P(X)} = \frac{P(X \,|\, \boldsymbol{\theta})P(\boldsymbol{\theta})}{\int P(X \,|\, \boldsymbol{\theta})P(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

If $\boldsymbol{\theta}$ is discrete, then $\int f(\boldsymbol{\theta})d\boldsymbol{\theta}$ is replaced with $\sum f(\boldsymbol{\theta})$.

- In words:

$$\text{Posterior of } \boldsymbol{\theta} \text{ given } X = \frac{\text{Conditional Likelihood} \times \text{Prior}}{\text{Marginal Likelihood}}$$

The *i.i.d.* assumption

- Usually $X = \{x_1, \ldots, x_n\}$ represents the data and $\boldsymbol{\theta}$ represents the model parameters.
- One usually assumes that $\{x_i\}_i$ are independent and identically distributed (*i.i.d*) conditioning on $\boldsymbol{\theta}$.
- Under the conditional *i.i.d.* assumption:
  - $P(X \mid \boldsymbol{\theta}) = \prod_{i=1}^{n} P(x_i \mid \boldsymbol{\theta})$.
  - The data in X are exchangeable, which means that $P(x_1, \ldots, x_n) = P(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$ for any random permutation σ of the data indices $1, 2, \ldots, n$.

Marginal likelihood and predictive distribution

- Marginal likelihood:

$$P(X) = \int P(X, \boldsymbol{\theta}) d\boldsymbol{\theta} = \int P(X \,|\, \boldsymbol{\theta}) P(\boldsymbol{\theta}) d\boldsymbol{\theta}$$

- Predictive distribution of a new data point x_{n+1}:

$$P(x_{n+1} \,|\, X) = \int P(x_{n+1} \,|\, \boldsymbol{\theta}) P(\boldsymbol{\theta} \,|\, X) d\boldsymbol{\theta} \quad \text{(assuming } \textit{i.i.d.}\text{)}$$

- The integrals are usually difficult to calculate. A popular approach is using Monte Carlo integration.
  - If possible, directly simulate S random samples $\{\boldsymbol{\theta}^{(s)}\}_{1,S}$ from $P(\boldsymbol{\theta} \,|\, X)$, otherwise, construct a Markov chain to draw $\{\boldsymbol{\theta}^{(s)}\}_{1,S}$ from $P(\boldsymbol{\theta} \,|\, X)$.
  - Approximate the integral as

$$P(x_{n+1} \,|\, X) \approx \sum_{s=1}^{S} \frac{P(x_{n+1} \,|\, \boldsymbol{\theta}^{(s)})}{S}.$$

Selecting an appropriate data likelihood $P(X \mid \boldsymbol{\theta})$

Selecting an appropriate conditional likelihood $P(X \mid \boldsymbol{\theta})$ to describe your data. Some common choices:

- Real-valued: normal distribution $x \sim \mathcal{N}(\mu, \sigma^2)$

$$P(x \mid \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

- Real-valued vector: multivariate normal distribution $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is the covariance matrix
- Gaussian maximum likelihood and least squares: finding the μ that minimizes the least squares objective function

$$\sum_{i=1}^{n} (x_i - \mu)^2$$

is the same as finding the μ that maximizes the Gaussian likelihood

$$\prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(x_i-\mu)^2}{2\sigma^2}\right]$$

- Binary data: Bernoulli distribution $x \sim \text{Bernoulli}(p)$

$$P(x \mid p) = p^x(1-p)^{1-x}, \quad x \in \{0, 1\}$$

- Count data: non-negative integers
  - Poisson distribution $x \sim \text{Pois}(\lambda)$

$$P(x \mid \lambda) = \frac{\lambda^x e^{-\lambda}}{x!}, \quad x \in \{0, 1, \ldots\}$$

  - Negative binomial distribution $x \sim \text{NB}(r, p)$

$$P(x \mid r, p) = \frac{\Gamma(n+r)}{n!\Gamma(r)} p^n (1-p)^r, \quad x \in \{0, 1, \ldots\}$$

- Positive real-valued:
  - Gamma distribution
    - $x \sim \text{Gamma}(k, \theta)$, where k is the shape parameter and θ is the scale parameter:

$$P(x \mid k, \theta) = \frac{\theta^{-k}}{\Gamma(k)} x^{k-1} e^{-\frac{x}{\theta}}, \quad x \in (0, \infty)$$

    - Or $x \sim \text{Gamma}(\alpha, \beta)$, where $\alpha = k$ is the shape parameter and $\beta = \theta^{-1}$ is the rate parameter:

$$P(x \mid \alpha, \beta) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}, \quad x \in (0, \infty)$$

    - $\mathbb{E}[x] = k\theta = \alpha/\beta$, $\text{var}[X] = k\theta^2 = \alpha/\beta^2$.
  - Truncated normal distribution

- Categorical: $(n_1, \ldots, n_k) \sim \text{Multinomial}(n, p_1, \ldots, p_k)$

$$P(n_1, \ldots, n_k \,|\, n, p_1, \ldots, p_k) = \frac{n!}{\prod_{i=1}^k n_i!} p_1^{n_1} \ldots p_k^{n_k}$$

where $n_i \in \{0, \ldots, n\}$ and $\sum_{i=1}^k n_i = n$.
- Ordinal, ranking
- Vector, matrix, tensor
- Time series
- Tree, graph, network, etc

Constructing an appropriate prior $P(\boldsymbol{\theta})$

- Construct an appropriate prior $P(\boldsymbol{\theta})$ to impose prior information, regularize the joint likelihood, and help derive efficient inference.
- Informative and non-informative priors:
  one may set the hyper-parameters of the prior distribution to reflect different levels of prior beliefs.
- Conjugate priors
- Hierarchical priors

Conjugate priors

If the prior $P(\boldsymbol{\theta})$ is conjugate to the likelihood $P(X \mid \boldsymbol{\theta})$, then the posterior $P(\boldsymbol{\theta} \mid X)$ and the prior $P(\boldsymbol{\theta})$ are in the same family.

- Conjugate priors are widely used to construct hierarchical Bayesian models.
- Although conjugacy is not required for MCMC/variational Bayes inference, it helps develop closed-form Gibbs sampling/variational Bayes update equations.

- Example (i): beta is conjugate to Bernoulli.

$$x_i \,|\, p \sim \text{Bernoulli}(p), \;\; p \sim \text{Beta}(\beta_0, \beta_1)$$

  - Conditional likelihood:

$$P(x_1, \ldots, x_n \,|\, p) = \prod_{i=1}^{n} p^{x_i}(1-p)^{1-x_i}$$

  - Prior: $P(p \,|\, \beta_0, \beta_1) = \frac{\Gamma(\beta_0+\beta_1)}{\Gamma(\beta_0)\Gamma(\beta_1)} p^{\beta_0-1}(1-p)^{\beta_1-1}$
  - Posterior:

$$P(p \,|\, X, \beta_0, \beta_1) \propto \left\{\prod_{i=1}^{n} p^{x_i}(1-p)^{1-x_i}\right\}\left\{p^{\beta_0-1}(1-p)^{\beta_1-1}\right\}$$

$$(p \,|\, x_1, \ldots, x_n, \beta_0, \beta_1) \sim \text{Beta}\left(\beta_0 + \sum_{i=1}^{n} x_i, \;\; \beta_1 + n - \sum_{i=1}^{n} x_i\right)$$

  - Both the prior and posterior of p are beta distributed.

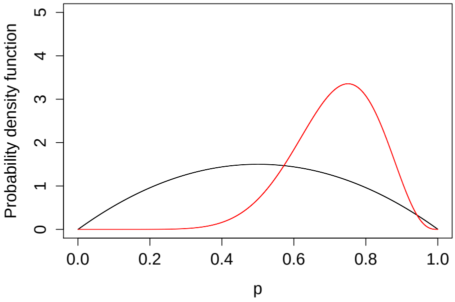
Flip a coin 10 times, observe 8 heads and 2 tails. Is this a fair coin?

- Model 1: $x_i \mid p \sim \text{Bernoulli}(p)$, $p \sim \text{Beta}(2, 2)$
  - Black is the prior probability density function:

$$p \sim \text{Beta}(2, 2)$$

  - Red is the posterior probability density function:

$$(p \mid x_1, \ldots, x_{10}) \sim \text{Beta}(10, 4)$$



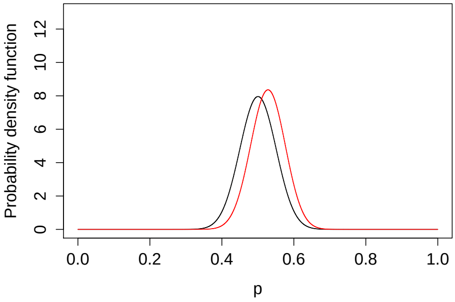
Flip a coin 10 times, observe 8 heads and 2 tails. Is this a fair coin?

- Model 2: $x_i \,|\, p \sim \text{Bernoulli}(p)$, $p \sim \text{Beta}(50, 50)$
  - Black is the prior probability density function:

$$p \sim \text{Beta}(50, 50)$$

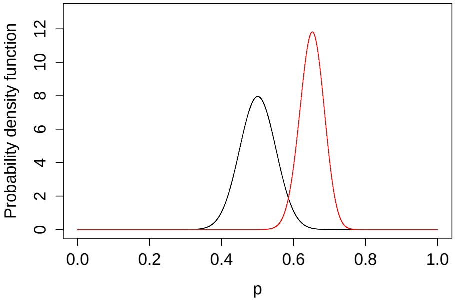
  - Red is the posterior probability density function:

$$(p \,|\, x_1, \ldots, x_{10}) \sim \text{Beta}(58, 52)$$



Flip 100 times, observe 80 heads and 20 tails. Is this a fair coin?

- Model 2: $x_i \,|\, p \sim \text{Bernoulli}(p)$, $p \sim \text{Beta}(50, 50)$
  - Black is the prior probability density function:
  $$p \sim \text{Beta}(50, 50)$$
  - Red is the posterior probability density function:
  $$(p \,|\, x_1, \ldots, x_{100}) \sim \text{Beta}(130, 70)$$



Data, prior, and posterior

- Suppose the data is the same:
  - The data would have a stronger influence on the posterior if the prior is weaker.
- Suppose the prior is the same:
  - More observations usually reduce the uncertainty in the posterior.

- Example (ii): the gamma distribution is the conjugate prior for the precision parameter of the normal distribution.

$$x_i \,|\, \mu, \varphi \sim \mathcal{N}(\mu, \varphi^{-1}), \;\; \varphi \sim \text{Gamma}(\alpha, \beta)$$

  - Conditional likelihood:

$$P(x_1, \ldots, x_n \,|\, \mu, \varphi) \propto \varphi^{n/2} \exp\left[-\varphi \sum_{i=1}^{n} (x_i - \mu)^2/2\right]$$

  - Prior: $P(\varphi \,|\, \alpha, \beta) \propto \varphi^{\alpha-1} e^{-\beta\varphi}$
  - Posterior:

$$P(\varphi \,|\, -) \propto \left\{\varphi^{n/2} e^{-\varphi \sum_{i=1}^{n}(x_i-\mu)^2/2}\right\}\left\{\varphi^{\alpha-1} e^{-\beta\varphi}\right\}$$

$$(\varphi|-) \sim \text{Gamma}\left(\alpha + \frac{n}{2},\; \beta + \sum_{i=1}^{n} \frac{(x_i-\mu)^2}{2}\right)$$

  - Both the prior and posterior of φ are gamma distributed.

- Example (iii): $x_i \sim \mathcal{N}(\mu, \varphi^{-1}),\ \mu \sim \mathcal{N}(\mu_0, \varphi_0^{-1})$
- Example (iv): $x_i \sim \text{Poisson}(\lambda),\ \lambda \sim \text{Gamma}(\alpha, \beta)$
- Example (v): $x_i \sim \text{NegBino}(r, p),\ p \sim \text{Beta}(\alpha_0, \alpha_1)$
- Example (vi): $x_i \sim \text{Gamma}(\alpha, \beta),\ \beta \sim \text{Gamma}(\alpha_0, \beta_0)$
- Example (vii):

$$(x_{i1}, \ldots, x_{ik}) \sim \text{Multinomial}(n_i, p_1, \ldots, p_k),$$

$$(p_1, \ldots, p_k) \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_k) = \frac{\Gamma(\sum_{j=1}^k \alpha_j)}{\prod_{j=1}^k \Gamma(\alpha_j)} \prod_{j=1}^k p_j^{\alpha_j - 1}$$

Hierarchical priors

- One may construct a complex prior distribution using a hierarchy of simple distributions as

$$P(\boldsymbol{\theta}) = \int \dots \int P(\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}_t) P(\boldsymbol{\alpha}_t \,|\, \boldsymbol{\alpha}_{t-1}) \dots P(\boldsymbol{\alpha}_1) d\boldsymbol{\alpha}_1 \dots d\boldsymbol{\alpha}_t$$

- Draw $\boldsymbol{\theta}$ from $P(\boldsymbol{\theta})$ using a hierarchical model:

$$\begin{aligned}
\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}_t, \dots, \boldsymbol{\alpha}_1 &\sim P(\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}_t) \\
\boldsymbol{\alpha}_t \,|\, \boldsymbol{\alpha}_{t-1}, \dots, \boldsymbol{\alpha}_1 &\sim P(\boldsymbol{\alpha}_t \,|\, \boldsymbol{\alpha}_{t-1}) \\
&\dots \\
\boldsymbol{\alpha}_1 &\sim P(\boldsymbol{\alpha}_1)
\end{aligned}$$

- Example (i): beta-negative binomial distribution[1]

$$n \mid \lambda \sim \mathrm{Pois}(\lambda), \;\; \lambda \mid r, p \sim \mathrm{Gamma}\left(r, \frac{p}{1-p}\right), \;\; p \sim \mathrm{Beta}(\alpha, \beta)$$

$$P(n \mid r, \alpha, \beta) = \iint \mathrm{Pois}(n; \lambda)\mathrm{Gamma}\left(\lambda; r, \frac{p}{1-p}\right)\mathrm{Beta}(p; \alpha, \beta) d\lambda dp$$

$$P(n \mid r, \alpha, \beta) = \frac{\Gamma(r+n)}{n!\Gamma(r)} \frac{\Gamma(\beta+r)\Gamma(\alpha+n)\Gamma(\alpha+\beta)}{\Gamma(\alpha+\beta+r+n)\Gamma(\alpha)\Gamma(\beta)}, \quad n \in \{0, 1, \ldots\}$$

  - A complicated probability mass function for a discrete random variable arises from a simple beta-gamma-Poisson mixture.

[1] Here $p/(1-p)$ represents the scale parameter of the gamma distribution

- Example (ii): Student's t-distribution

$$x \,|\, \varphi \sim \mathcal{N}(0, \varphi^{-1}), \;\; \varphi \sim \text{Gamma}(\alpha, \beta)$$

$$P(x) = \int \mathcal{N}(x; 0, \varphi^{-1}) \text{Gamma}(\varphi; \alpha, \beta) d\varphi$$
$$= \frac{\Gamma(\alpha + \frac{1}{2})}{\sqrt{2\beta\pi}\,\Gamma(\alpha)} \left(1 + \frac{x^2}{2\beta}\right)^{-\alpha - \frac{1}{2}}$$

If $\alpha = \beta = \nu/2$, then $P(x) = t_\nu(x)$ is the Student's t-distribution with ν degrees of freedom

- Homework: derive the probability density function shown above.

- Example (iii): Laplace distribution (e.g., Park and Casella, JASA 2008)

$$x \,|\, \eta \sim \mathcal{N}(0, \eta), \;\; \eta \sim \text{Exponential}(\gamma^2/2), \quad \gamma > 0$$

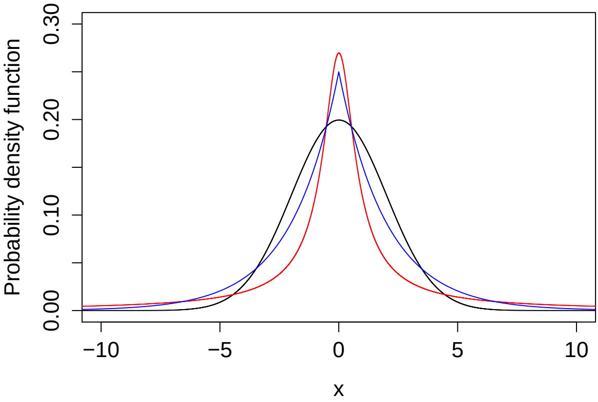
$$P(x) = \int \mathcal{N}(x; 0, \eta)\text{Exponential}(\eta; \gamma^2/2) d\eta = \frac{\gamma}{2} e^{-\gamma\,|\,x\,|}$$

$P(x)$ is the probability density function of the Laplace distribution, and hence

$$x \sim \text{Laplace}(0, \gamma^{-1})$$

- Homework (optional): derive the probability density function shown above (hint: check the inverse Gaussian distribution for help).
- The Student's t and Laplace distributions are two widely used sparsity-promoting priors.

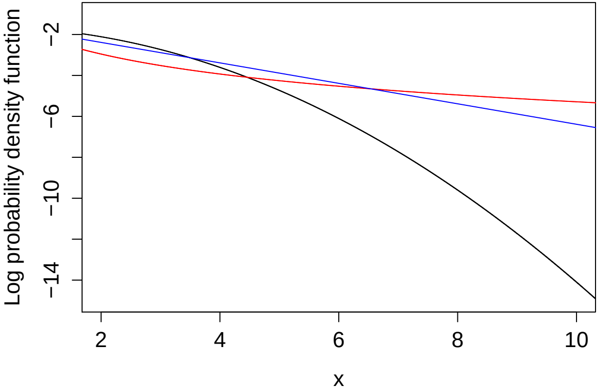
Black: $x \sim \mathcal{N}[0, (\sqrt{2})^2]$
Red: $x \sim t_{0.5}$
Blue: $x \sim \text{Laplace}(0, 2)$



Black: $x \sim \mathcal{N}[0, (\sqrt{2})^2]$

Red: $x \sim t_{0.5}$

Blue: $x \sim \text{Laplace}(0, 2)$



Priors and regularizations

- Different priors can be matched to different regularizations as

$$-\ln P(\boldsymbol{\theta} \,|\, X) = -\ln P(X \,|\, \boldsymbol{\theta}) - \ln P(\boldsymbol{\theta}) + C,$$

  where C is a term that is not related to $\boldsymbol{\theta}$.
- Assume that the data are generated as $x_i \sim \mathcal{N}(\mu, 1)$ and the goal is to find a maximum a posteriori probability (MAP) estimate of μ.
  - If $\mu \sim \mathcal{N}(0, \varphi^{-1})$, then the MAP estimate is the same as

$$\operatorname*{argmin}_{\mu} \sum_{i=1}^{n} (x_i - \mu)^2 + \varphi\mu^2$$

  - If $\mu \sim t_\nu$, then the MAP estimate is the same as

$$\operatorname*{argmin}_{\mu} \sum_{i=1}^{n} (x_i - \mu)^2 + (\nu + 1)\ln(1 + \nu^{-1}\mu^2)$$

  - If $\mu \sim \text{Laplace}(0, \gamma^{-1})$, then the MAP estimate is the same as

$$\operatorname*{argmin}_{\mu} \sum_{i=1}^{n} (x_i - \mu)^2 + \gamma \,|\, \mu \,|$$

A typical advantage of solving a hierarchical Bayesian model over solving a related regularized objective function:

- The regularization parameters, such as φ, ν and γ in the previous slide, often have to be cross-validated.
- In a hierarchical Bayesian model, we usually impose (possibly conjugate) priors on these parameters and infer their posteriors given the data.
- If we impose non-informative priors, then we let the data speak for themselves.

Inference via Gibbs sampling

- Gibbs sampling:
  - One of the simplest Markov chain Monte Carlo (MCMC) algorithm for multivariate distributions.
  - Widely used for statistical inference.
- For a multivariate distribution $P(x_1, \ldots, x_n)$ that is difficult to sample from, if it is simpler to sample each of its variables conditioning on all the others, then we may use Gibbs sampling to obtain samples from this distribution as
  - Initialize $(x_1, \ldots, x_n)$ at some values.
  - For $s = 1 : S$
    For $i = 1 : n$
    Sample x_i conditioning on the others from
    $$P(x_i \,|\, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$
    End
    End

- A complicated multivariate distribution (Zhou and Walker, 2014):

$$p(z_1, \ldots, z_n \,|\, n, \gamma_0, a, p) = \frac{\gamma_0^l p^{-al}}{\sum_{\ell=0}^{n} \gamma_0^\ell p^{-a\ell} S_a(n, \ell)} \prod_{k=1}^{l} \frac{\Gamma(n_k - a)}{\Gamma(1-a)},$$

  where z_i are categorical random variables, l is the number of distinct values in $\{z_1, \ldots, z_n\}$, $n_k = \sum_{i=1}^{n} \delta(z_i = k)$, and $S_a(n, \ell)$ are generalized Stirling numbers of the first kind.

- Gibbs sampling is easy:
  - Initialize $(z_1, \ldots, z_n)$ at some values.
  - For $s = 1 : S$

    For $i = 1 : n$

    Sample z_i from

$$P(z_i = k \,|\, z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n, n, \gamma_0, a, p) \propto \begin{cases} n_k^{-i} - a, & \text{for } k = 1, \ldots, l^{-i}; \\ \gamma_0 p^{-a}, & \text{if } k = l^{-i} + 1. \end{cases}$$

    End

    End

Gibbs sampling in a hierarchal Bayesian model

- Full joint likelihood of the hierarchical Bayesian model:

$$P(X, \boldsymbol{\theta}, \boldsymbol{\alpha}_t, \ldots, \boldsymbol{\alpha}_1) = P(X \,|\, \boldsymbol{\theta}) P(\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}_t) P(\boldsymbol{\alpha}_t \,|\, \boldsymbol{\alpha}_{t-1}) \ldots P(\boldsymbol{\alpha}_1)$$

- Exact posterior inference is often intractable. We use Gibbs sampling for approximate inference.
- Assume in the hierarchical Bayesian model that:
  - $P(\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}_t)$ is conjugate to $P(X \,|\, \boldsymbol{\theta})$;
  - $P(\boldsymbol{\alpha}_t \,|\, \boldsymbol{\alpha}_{t-1})$ is conjugate to $P(\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}_t)$;
  - $P(\boldsymbol{\alpha}_j \,|\, \boldsymbol{\alpha}_{j-1})$ is conjugate to $P(\boldsymbol{\alpha}_{j+1} \,|\, \boldsymbol{\alpha}_j)$ for $j \in \{1, \ldots, t-1\}$.

- In each MCMC iteration, Gibbs sampling proceeds as
  - Sample $\boldsymbol{\theta}$ from
    $$P(\boldsymbol{\theta} \,|\, X, \boldsymbol{\alpha}_t) \propto P(X \,|\, \boldsymbol{\theta}) P(\boldsymbol{\theta} \,|\, \boldsymbol{\alpha}_t);$$
  - For $j \in \{1, \dots, t-1\}$, sample $\boldsymbol{\alpha}_j$ from
    $$P(\boldsymbol{\alpha}_j \,|\, \boldsymbol{\alpha}_{j+1}, \boldsymbol{\alpha}_{j-1}) \propto P(\boldsymbol{\alpha}_{j+1} \,|\, \boldsymbol{\alpha}_j) P(\boldsymbol{\alpha}_j \,|\, \boldsymbol{\alpha}_{j-1}).$$
- If $\boldsymbol{\theta} = (\theta_1, \dots, \theta_V)$ is a vector and $P(\boldsymbol{\theta} \,|\, X, \boldsymbol{\alpha}_t)$ is difficult to sample from, then one may further consider sampling $\boldsymbol{\theta}$ as
  - for $v \in \{1, \dots, V\}$, sample θ_v from
    $$P(\theta_v \,|\, \boldsymbol{\theta}^{-v}, X, \boldsymbol{\alpha}_t) \propto P(X \,|\, \boldsymbol{\theta}^{-v}, \theta_v) P(\theta_v \,|\, \boldsymbol{\theta}^{-v}, \boldsymbol{\alpha}_t),$$
    where $\boldsymbol{\theta}^{-v} = (\theta_1, \dots, \theta_{v-1}, \theta_{v+1}, \dots, \theta_V)$.

Data augmentation and marginalization

What if $P(\boldsymbol{\alpha}_j \,|\, \boldsymbol{\alpha}_{j-1})$ is not conjugate to $P(\boldsymbol{\alpha}_{j+1} \,|\, \boldsymbol{\alpha}_j)$?

- Use other MCMC algorithms such as the Metropolis-Hastings algorithm.
- Marginalization: suppose $P(\boldsymbol{\alpha}_j \,|\, \boldsymbol{\alpha}_{j-1})$ is conjugate to $P(\boldsymbol{\alpha}_{j+2} \,|\, \boldsymbol{\alpha}_j)$, then one may sample $\boldsymbol{\alpha}_j$ in closed form conditioning on $\boldsymbol{\alpha}_{j+2}$ and $\boldsymbol{\alpha}_{j-1}$.
- Augmentation: suppose ℓ is an auxiliary variable such that

  $$P(\ell, \boldsymbol{\alpha}_{j+1} \,|\, \boldsymbol{\alpha}_j) = P(\ell \,|\, \boldsymbol{\alpha}_{j+1}, \boldsymbol{\alpha}_j) P(\boldsymbol{\alpha}_{j+1} \,|\, \boldsymbol{\alpha}_j) = P(\boldsymbol{\alpha}_{j+1} \,|\, \ell, \boldsymbol{\alpha}_j) P(\ell \,|\, \boldsymbol{\alpha}_j),$$

  and $P(\boldsymbol{\alpha}_j \,|\, \boldsymbol{\alpha}_{j-1})$ is conjugate to $P(\ell \,|\, \boldsymbol{\alpha}_j)$, then one can sample ℓ from $P(\ell \,|\, \boldsymbol{\alpha}_{j+1}, \boldsymbol{\alpha}_j)$ and then sample $\boldsymbol{\alpha}_j$ in closed form conditioning on ℓ and $\boldsymbol{\alpha}_{j-1}$.
- We will provide an example on how to use marginalization and augmentation to derive closed-form Gibbs sampling update equations when discussing count data.

Posterior representation with MCMC samples

- In MCMC algorithms, the posteriors of model parameters are represented using collected posterior samples.
- To collect S posterior samples, one often consider $(S_{Burnin} + g * S)$ Gibbs sampling iterations:
  - Discard the first S_{Burnin} samples;
  - Collect a sample per $g \geq 1$ iterations after the burn-in period.

  One may also consider multiple independent Markov chains, collecting one or multiple samples from each chain.
- MCMC Diagnostics:
  - Inspecting the traceplots of important model parameters
  - Convergence
  - Mixing
  - Autocorrelation
  - Effective sample size
  - ...

- With S posterior samples of $\boldsymbol{\theta}$, one can approximately
  - calculate the posterior mean of $\boldsymbol{\theta}$ using

$$\sum_{s=1}^{S} \frac{\theta^{(s)}}{S}$$

  - calculate $\int f(\boldsymbol{\theta})P(\boldsymbol{\theta} \mid X)$ using

$$\sum_{s=1}^{S} \frac{f(\boldsymbol{\theta}^{(s)})}{S}$$

  - calculate $P(x_{n+1} \mid X) = \int P(x_{n+1} \mid \boldsymbol{\theta})P(\boldsymbol{\theta} \mid X)d\boldsymbol{\theta}$ using

$$\sum_{s=1}^{S} \frac{P(x_{n+1}|\boldsymbol{\theta}^{(s)})}{S}$$

  - the error of Monte Carlo integration with S independent samples decreases with $\sqrt{S}$.

Variational Bayes inference

- Since $\ln P(X) = \ln P(X, \boldsymbol{\theta}) - \ln P(\boldsymbol{\theta} \,|\, X) = \ln \frac{P(X,\boldsymbol{\theta})}{Q(\boldsymbol{\theta})} - \ln \frac{P(\boldsymbol{\theta}\,|\,X)}{Q(\boldsymbol{\theta})}$ and $\ln P(X) = \int Q(\boldsymbol{\theta}) \ln P(X) d\boldsymbol{\theta}$, we have

$$\begin{aligned}\ln P(X) &= \int Q(\boldsymbol{\theta}) \ln \frac{P(X,\boldsymbol{\theta})}{Q(\boldsymbol{\theta})} d\boldsymbol{\theta} + \int Q(\boldsymbol{\theta}) \ln \frac{Q(\boldsymbol{\theta})}{P(\boldsymbol{\theta}\,|\,X)} d\boldsymbol{\theta} \\ &= \mathcal{L}(Q) + \mathrm{KL}(Q||P).\end{aligned}$$

- Since $\mathrm{KL}(Q||P) \geq 0$, minimizing the Kullback-Leibler (KL) divergence of $P(\boldsymbol{\theta}\,|\,X)$ from $Q(\boldsymbol{\theta})$ is the same as maximizing the lower bound

$$\mathcal{L}(Q) = \mathbb{E}_Q[\ln P(X, \boldsymbol{\theta})] - \mathbb{E}_Q[\ln Q(\boldsymbol{\theta})].$$

- For tractable inference, one typically assumes that $Q(\boldsymbol{\theta})$ can be factorized as $Q(\boldsymbol{\theta}) = \prod_i Q_i(\theta_i)$.
- Under this factorized form, the lower bound is maximized with

$$Q(\theta_i) = \frac{\exp\{\mathbb{E}_{\{q_j\}_{j\neq i}}[\ln P(X, \boldsymbol{\theta})]\}}{\int \exp\{\mathbb{E}_{\{q_j\}_{j\neq i}}[\ln P(X, \boldsymbol{\theta})]\} d\theta_i}.$$

Factor analysis

- Denote $\mathbf{D} = (\boldsymbol{d}_1, \ldots, \boldsymbol{d}_K) \in \mathbb{R}^{P \times K}$ as a factor loading matrix.
- Denote $\mathbf{S} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N) \in \mathbb{R}^{K \times N}$ as a factor score matrix.
- If $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{s}_i + \boldsymbol{\mu} + \boldsymbol{\epsilon}_i, \ \boldsymbol{s}_i \sim \mathcal{N}(0, \boldsymbol{\Lambda}), \ \boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \boldsymbol{\Psi})$, then marginalizing out $\boldsymbol{s}_i$ leads to

$$\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{D}\boldsymbol{\Lambda}\mathbf{D}^T + \boldsymbol{\Psi}),$$

  where $\boldsymbol{\Lambda}$ is typically defined as an identity or diagonal matrix.
- For simplicity, let's construct a hierarchical Bayesian model with $\boldsymbol{\mu} = 0$, $\boldsymbol{\Lambda} = \mathrm{diag}\{\gamma_{s1}^{-1}, \ldots, \gamma_{sK}^{-1}\}$, and $\boldsymbol{\Psi} = \gamma_\epsilon^{-1}\mathbf{I}_P$.

- Hierarchical model for Bayesian factor analysis:

$$\boldsymbol{x}_i = \mathbf{D}\boldsymbol{s}_i + \boldsymbol{\epsilon}_i, \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \gamma_\epsilon^{-1}\mathbf{I}_P)$$
$$\boldsymbol{d}_k \sim \mathcal{N}(0, P^{-1}\mathbf{I}_P), \qquad s_{ik} \sim \mathcal{N}(0, \gamma_{sk}^{-1})$$
$$\gamma_s \sim \text{Gamma}(c_0, d_0), \quad \gamma_\epsilon \sim \text{Gamma}(e_0, f_0)$$

- The number of factors K is a tuning parameter.
- Other variations can also be considered, such as letting $\boldsymbol{s}_i \sim \mathcal{N}\left(0, \gamma_s^{-1}\mathbf{I}_K\right)$ and restricting $\boldsymbol{d}_k$ and/or $\boldsymbol{s}_i$ to be nonnegative.
- Data are partially observed (missing data problem):

$$\boldsymbol{y}_i = \boldsymbol{\Sigma}_i \boldsymbol{x}_i$$

where $\boldsymbol{\Sigma}_i$ is a projection matrix on $\boldsymbol{x}_i$, which is constructed by removing the rows of the identity matrix that correspond to the indices of the missing values in $\boldsymbol{x}_i$, with $\boldsymbol{\Sigma}_i\boldsymbol{\Sigma}_i^T = \mathbf{I}_{||\boldsymbol{\Sigma}_i||_0}$

- Full joint likelihood:

$$
\begin{aligned}
&P(\mathbf{Y}, \boldsymbol{\Sigma}, \mathbf{D}, \mathbf{S}, \gamma_s, \gamma_\epsilon) \\
=&\prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i; \boldsymbol{\Sigma}_i \mathbf{D} \boldsymbol{s}_i, \gamma_\epsilon^{-1} \mathbf{I}_{||\boldsymbol{\Sigma}||_0}) \mathcal{N}(\boldsymbol{s}_i; 0, \mathrm{diag}\{\gamma_{s1}^{-1}, \ldots, \gamma_{sK}^{-1}\}) \\
&\prod_{k=1}^{K} \mathcal{N}(\boldsymbol{d}_k; 0, P^{-1} \mathbf{I}_P) \\
&\mathrm{Gamma}(\gamma_s; c_0, d_0), \mathrm{Gamma}(\gamma_\epsilon; e_0, f_0)
\end{aligned}
$$

- Gibbs sampling (Similar to Zhou et al., IEEE TIP 2012)
  - For $k = 1, 2, \dots, K$
    - Sample s_{ik} from Normal for $i = 1, 2, \dots, N$
    - Sample $\boldsymbol{d}_k$ from Multivariate Normal (with a diagonal covariance matrix)
    - Sample γ_{sk} from Gamma
  - Sample γ_ϵ from Gamma
- Note that one may also sample $\boldsymbol{s}_i = (s_{i1}, \dots, s_{iK})^T$ from a multivariate normal distribution, which is more computationally expensive to sample from since the $K \times K$ covariance matrix is generally not diagonal.
- We illustrate in the next slides on how to find $P(\boldsymbol{d}_k \,|\, -)$, the conditional posterior of $\boldsymbol{d}_k$.

- Denoting

$$\boldsymbol{y}_i^{-k} = \boldsymbol{y}_i - \boldsymbol{\Sigma}_i \mathbf{D}\boldsymbol{s}_i + \boldsymbol{\Sigma}_i \boldsymbol{d}_k s_{ik} = \boldsymbol{y}_i - \boldsymbol{\Sigma}_i \sum_{k' \neq k} \boldsymbol{d}_k s_{ik'},$$

since $\boldsymbol{y}_i \sim \mathcal{N}(\boldsymbol{\Sigma}_i \mathbf{D}\boldsymbol{s}_i, \gamma_\epsilon^{-1}\mathbf{I}_{||\boldsymbol{\Sigma}||_0})$, we have

$$\boldsymbol{y}_i^{-k} \sim \mathcal{N}(\boldsymbol{\Sigma}_i \boldsymbol{d}_k s_{ik}, \gamma_\epsilon^{-1}\mathbf{I}_{||\boldsymbol{\Sigma}||_0})$$

in the prior and hence in the posterior

$$\begin{aligned} P(\boldsymbol{d}_k \,|\, -) &\propto e^{-\frac{P}{2}\boldsymbol{d}_k^T\boldsymbol{d}_k} e^{-\frac{1}{2}\sum_i \gamma_\epsilon (\boldsymbol{y}_i^{-k} - \boldsymbol{\Sigma}_i \boldsymbol{d}_k s_{ik})^T(\boldsymbol{y}_i^{-k} - \boldsymbol{\Sigma}_i \boldsymbol{d}_k s_{ik})} \\ &\propto e^{-\frac{1}{2}\boldsymbol{d}_k^T(P\mathbf{I}_P + \gamma_\epsilon \sum_i s_{ik}^2 \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i)\boldsymbol{d}_k + \boldsymbol{d}_k^T \gamma_\epsilon \sum_i s_{ik} \boldsymbol{\Sigma}_i^T \boldsymbol{y}_i^{-k}} \\ &\propto e^{-\frac{1}{2}(\boldsymbol{d}_k - \boldsymbol{\mu}_{\boldsymbol{d}_k})^T \boldsymbol{\Sigma}_{\boldsymbol{d}_k}^{-1} (\boldsymbol{d}_k^T - \boldsymbol{\mu}_{\boldsymbol{d}_k})}. \end{aligned}$$

Therefore, we can sample $\boldsymbol{d}_k$ from its conditional posterior as

$$(\boldsymbol{d}_k \,|\, -) \sim \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{d}_k}, \boldsymbol{\Sigma}_{\boldsymbol{d}_k}),$$

where $\boldsymbol{\Sigma}_{\boldsymbol{d}_k} = (P\mathbf{I}_P + \gamma_\epsilon \sum_i s_{ik}^2 \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i)^{-1}$, $\boldsymbol{\mu}_{\boldsymbol{d}_k} = \gamma_\epsilon \boldsymbol{\Sigma}_{\boldsymbol{d}_k} \sum_i s_{ik} \boldsymbol{\Sigma}_i^T \boldsymbol{y}_i^{-k}$.

- Homework 1: derive Gibbs sampling update equations.
- Homework 2: Evaluate the model and Gibbs sampler on the MovieLens 100K dataset.
  - 943 users and 1946 movies
  - 80,000 ratings as training and 20,000 ratings as testing
  - For a given K, consider 1500 Gibbs sampling iterations. Discard the first 1000 samples and collect the remaining 500 samples to compute the predicted ratings on held-out user-movie pairs.
  - Set $K = 5, 10, 20, 40, 80, 160$ and examine how the performance, measured by root-mean-square error (RMSE) on the heldout ratings, changes with K.
  - Is the above procedure of choosing K practical. Why? If not, how to choose the best K?

Variational Bayes inference

- Choose $Q = Q(\gamma_\epsilon)\prod_k Q(\gamma_{sk})Q(\boldsymbol{d}_k)\prod_i Q(s_{ik})$, where

$$Q(\gamma_{sk}) = \text{Gamma}(\tilde{c}_{\gamma_{sk}}, \tilde{d}_{\gamma_{sk}}), \quad Q(\gamma_\epsilon) = \text{Gamma}(\tilde{e}_0, \tilde{f}_0)$$

$$Q(\boldsymbol{d}_k) = \mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{d}_k}, \boldsymbol{\Sigma}_{\boldsymbol{d}_k}), \quad Q(s_{ik}) = \mathcal{N}(\nu_{ik}, \Omega_{ik})$$

  One may also replace $\prod_k\prod_i Q(s_{ik})$ with $\prod_i Q(\boldsymbol{s}_i)$.
- Find Q to minimize $\mathcal{L}(Q) = \mathbb{E}_Q[\ln P(X,\boldsymbol{\theta})] - \mathbb{E}_Q[\ln Q(\boldsymbol{\theta})]$, where the joint likelihood $P(X,\boldsymbol{\theta})$ is shown in slide 39.
- For $Q(\boldsymbol{d}_k)$, we have

$$\boldsymbol{\Sigma}_{\boldsymbol{d}_k} = (P\mathbf{I}_P + \langle\gamma_\epsilon\rangle \textstyle\sum_i \langle s_{ik}^2\rangle \boldsymbol{\Sigma}_i^T\boldsymbol{\Sigma}_i)^{-1}$$
$$\boldsymbol{\mu}_{\boldsymbol{d}_k} = \langle\gamma_\epsilon\rangle \boldsymbol{\Sigma}_{\boldsymbol{d}_k} \textstyle\sum_i \langle s_{ik}\rangle \boldsymbol{\Sigma}_i^T \langle \boldsymbol{y}_i^{-k}\rangle,$$

  where $\langle\gamma_\epsilon\rangle = \tilde{e}_0/\tilde{f}_0$, $\langle s_{ik}\rangle = \nu_{ik}$, $\langle s_{ik}^2\rangle = \nu_{ik}^2 + \Omega_{ik}$, and $\langle \boldsymbol{y}_i^{-k}\rangle = \boldsymbol{y}_i - \boldsymbol{\Sigma}_i \sum_{k'\neq k}\langle \boldsymbol{d}_k\rangle\langle s_{ik'}\rangle$.

Variational Bayes inference

- Homework 3 (Optional): find the update equations for the other parameters, including ν_{ik}, Ω_{ik}, $\tilde{c}_{\gamma_{sk}}$, $\tilde{d}_{\gamma_{sk}}$, $\tilde{e}_0$, and $\tilde{f}_0$.
- Homework 4 (Optional): Code the variational Bayes algorithm and compare its performance with that of Gibbs sampling on the MovieLens 100K dataset.

Introduction to dictionary learning and sparse coding

- The input is a data matrix $\mathbf{X} \in \mathbb{R}^{P \times N} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, each column of which is a P dimensional data vector.
- Typical examples:
  - A movie rating matrix, with P movies and N users.
  - A matrix constructed from 8×8 image patches, with $P = 64$ pixels and N patches.
- The data matrix is usually incomplete and corrupted by noises.
- A common task is to recover the original complete and noise-free data matrix.

- A powerful approach is to learn a dictionary $\mathbf{D} \in \mathbb{R}^{P \times K}$ from the corrupted $\mathbf{X}$, with the constraint that a data vector is sparsely represented under the dictionary.
- The number of columns K of the dictionary could be larger than P, which means that the dictionary could be over-complete.
- A learned dictionary could provide a much better performance than an "off-the-shelf" or handcrafted dictionary.
- The original complete and noise-free data matrix is recovered with the product of the learned dictionary and sparse representations.

$$\underset{\text{Images}}{\mathbf{X}^{P\times N}} = \underset{\text{Dictionary}}{\mathbf{\Phi}^{P\times K}}\ \underset{\text{Sparse codes}}{\mathbf{\Theta}^{K\times N}}$$

Optimization based methods

- $\mathbf{X} \in \mathbb{R}^{P \times N}$ is the data matrix, $\mathbf{D} \in \mathbb{R}^{P \times K}$ is the dictionary, and $\mathbf{W} \in \mathbb{R}^{K \times N}$ is the sparse-code matrix.
- Objective function:

  $\min_{\mathbf{D},\mathbf{W}}\{||\mathbf{X} - \mathbf{DW}||_F\}$ subject to $\forall i,\ ||\boldsymbol{w}_i||_0 \leq T_0$

- A common approach to solve this objective function:
  - Sparse coding state: update sparse codes $\mathbf{W}$ while fixing the dictionary $\mathbf{D}$;
  - Dictionary learning state: update the dictionary $\mathbf{D}$ while fixing the sparse codes $\mathbf{W}$;
  - Iterate until convergence.

- Sparse coding stage: Fix dictionary $\mathbf{D}$, update sparse codes $\mathbf{W}$.
    - $\min_{\boldsymbol{w}_i} ||\boldsymbol{w}_i||_0$ subject to $||\boldsymbol{x}_i - \mathbf{D}\boldsymbol{w}_i||_2^2 \leq C\sigma^2$
    - or $\min_{\boldsymbol{w}_i} ||\boldsymbol{x}_i - \mathbf{D}\boldsymbol{w}_i||_2^2$ subject to $||\boldsymbol{w}_i||_0 \leq T_0$
- Dictionary update stage: Fix sparse codes $\mathbf{W}$ (or sparsity patterns), update dictionary $\mathbf{D}$.
    - Method of optimal direction (MOD) (fix the sparse codes):

$$\mathbf{D} = \mathbf{X}\mathbf{W}^T(\mathbf{W}\mathbf{W}^T)^{-1}$$

    - K-SVD (fix the sparsity pattern, rank-1 approximation):

$$\boldsymbol{d}_k \boldsymbol{w}_{k:} \approx \mathbf{X} - \sum_{m \neq k} \boldsymbol{d}_m \boldsymbol{w}_{m:}$$

- Restrictions of optimization based dictionary learning algorithms:
  - Have to assume a prior knowledge of noise variance, sparsity level or regularization parameters;
  - Nontrivial to handle data anomalies such as missing data;
  - May require sufficient noise free training data to pretrain the dictionary;
  - Only point estimates are provided.
  - Have to tune the number of dictionary atoms.
- We will solve all restrictions except for the last one using a parametric Bayesian model.
- The last restriction could be solved by making the model be nonparametric, which will be briefly discussed.

Sparse factor analysis (spike-and-slab sparse prior)

- Hierarchical Bayesian model (Zhou et al, 2009, 2012):

$$\begin{aligned} \boldsymbol{x}_i &= \mathbf{D}(\boldsymbol{z}_i \odot \boldsymbol{s}_i) + \boldsymbol{\epsilon}_i, & \boldsymbol{\epsilon}_i &\sim \mathcal{N}(0, \gamma_\epsilon^{-1}\mathbf{I}_P) \\ \boldsymbol{d}_k &\sim \mathcal{N}(0, P^{-1}\mathbf{I}_P), & \boldsymbol{s}_i &\sim \mathcal{N}(0, \gamma_s^{-1}\mathbf{I}_K) \\ z_{ik} &\sim \text{Bernoulli}(\pi_k), & \pi_k &\sim \text{Beta}(c/K, c(1-1/K)) \\ \gamma_s &\sim \text{Gamma}(c_0, d_0), & \gamma_\epsilon &\sim \text{Gamma}(e_0, f_0) \end{aligned}$$

where $\boldsymbol{z}_i \odot \boldsymbol{s}_i = (z_{i1}s_{i1}, \ldots, z_{iK}s_{iK})^T$.
Note if $z_{ik} = 0$, then the sparse code $z_{ik}s_{ik}$ is exactly zero.

- Data are partially observed:

$$\boldsymbol{y}_i = \boldsymbol{\Sigma}_i \boldsymbol{x}_i$$

where $\boldsymbol{\Sigma}_i$ is the projection matrix on the data, with $\boldsymbol{\Sigma}_i\boldsymbol{\Sigma}_i^T = \mathbf{I}_{||\boldsymbol{\Sigma}_i||_0}$

- Full joint likelihood:

$$P(\mathbf{Y}, \mathbf{\Sigma}, \mathbf{D}, \mathbf{Z}, \mathbf{S}, \boldsymbol{\pi}, \gamma_s, \gamma_\epsilon)$$

$$= \prod_{i=1}^{N} \mathcal{N}(\boldsymbol{y}_i; \mathbf{\Sigma}_i \mathbf{D}(\boldsymbol{z}_i \odot \boldsymbol{s}_i), \gamma_\epsilon^{-1} \mathbf{I}_{||\mathbf{\Sigma}||_0}) \mathcal{N}(\boldsymbol{s}_i; 0, \gamma_s^{-1} \mathbf{I}_K)$$

$$\prod_{k=1}^{K} \mathcal{N}(\boldsymbol{d}_k; 0, P^{-1} \mathbf{I}_P) \mathrm{Beta}(\pi_k; c/K, c(1-1/K))$$

$$\prod_{i=1}^{N} \prod_{k=1}^{K} \mathrm{Bernoulli}(z_{ik}; \pi_k)$$

$$\mathrm{Gamma}(\gamma_s; c_0, d_0), \mathrm{Gamma}(\gamma_\epsilon; e_0, f_0)$$

- Gibbs sampling (details can be found in Zhou et al., IEEE TIP 2012)
  - Sample z_{ik} from Bernoulli
  - Sample s_{ik} from Normal
  - Sample π_k from Beta
  - Sample $\boldsymbol{d}_k$ from Multivariate Normal
  - Sample γ_s from Gamma
  - Sample γ_ϵ from Gamma
- Homework 5 (Optional): Modify the model by letting $\boldsymbol{s}_i \sim \mathcal{N}(0, \mathrm{diag}\{\gamma_{s1}^{-1}, \ldots, \gamma_{sK}^{-1}\})$.
  - Derive and code the Gibbs sampling algorithm.
  - Test the algorithm on MovieLens 100K. Set $K = 160$ and mimick the same testing procedure used in Homework 2.
  - Examine the update equations and explain whether imposing sparsity brings computational savings.
  - Plot the posterior distribution (using the collected MCMC samples) of the inferred number of "active" factors for $K = 160$.

- Logarithm of the posterior

$$
\begin{aligned}
-\log\ p(\boldsymbol{\Theta}\,|\,\mathbf{X},\mathcal{H}) = \ & \frac{\gamma_\epsilon}{2}\sum_{i=1}^{N}\|\boldsymbol{x}_i - \mathbf{D}(\boldsymbol{s}_i\odot\boldsymbol{z}_i)\|_2^2 \\
& + \frac{P}{2}\sum_{k=1}^{K}\|\boldsymbol{d}_k\|_2^2 + \frac{\gamma_s}{2}\sum_{i=1}^{N}\|\boldsymbol{s}_i\|_2^2 \\
& - \log f_{Beta-Bern}(\{\boldsymbol{z}_i\}_{i=1}^{N};\mathcal{H}) \\
& - \log \mathrm{Gamma}(\gamma_\epsilon|\mathcal{H}) - \log \mathrm{Gamma}(\gamma_s|\mathcal{H}) \\
& + Const.
\end{aligned}
$$

  where $\boldsymbol{\Theta}$ represent the set of model parameters and $\mathcal{H}$ represents the set of hyper-parameters.

- The sparse factor model tries to minimize the least squares of the data fitting errors while encouraging the representations of the data under the learned dictionary to be sparse.

Handling data anomalies

- Missing data
  - full data: $\boldsymbol{x}_i$, observed: $\boldsymbol{y}_i = \Sigma_i \boldsymbol{x}_i$, missing: $\bar{\Sigma}_i \boldsymbol{x}_i$

$$\begin{aligned}\mathcal{N}(\boldsymbol{x}_i; \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i), \gamma_\epsilon^{-1}\mathbf{I}_P) &= \mathcal{N}(\boldsymbol{\Sigma}_i^T \boldsymbol{y}_i; \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i), \boldsymbol{\Sigma}_i^T \boldsymbol{\Sigma}_i \gamma_\epsilon^{-1}\mathbf{I}_P) \\ &\quad \mathcal{N}(\bar{\boldsymbol{\Sigma}}_i^T \bar{\boldsymbol{\Sigma}}_i \boldsymbol{x}_i; \bar{\boldsymbol{\Sigma}}_i^T \bar{\boldsymbol{\Sigma}}_i \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i), \bar{\boldsymbol{\Sigma}}_i^T \bar{\boldsymbol{\Sigma}}_i \gamma_\epsilon^{-1}\mathbf{I}_P)\end{aligned}$$

- Spiky noise (outliers)

$$\begin{aligned}\boldsymbol{x}_i &= \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i) + \boldsymbol{\epsilon}_i + \boldsymbol{v}_i \odot \boldsymbol{m}_i \\ \boldsymbol{v}_i &\sim \mathcal{N}(0, \gamma_v^{-1}\mathbf{I}_P), \; m_{ip} \sim \text{Bernoulli}(\pi'_{ip}), \; \pi'_{ip} \sim \text{Beta}(a_0, b_0)\end{aligned}$$

- Recovered data

$$\hat{\boldsymbol{x}}_i = \mathbf{D}(\boldsymbol{s}_i \odot \boldsymbol{z}_i)$$

How to select K?

- As $K \to \infty$, one can show that the parametric sparse factor analysis model using the spike-and-slab prior becomes a nonparametric Bayesian model governed by the beta-Bernoulli process, or the Indian buffet process if the beta process is marginalized out. This point will not be further discussed in this lecture.
- We set K to be large enough, making the parametric model be a truncated version of the beta process factor analysis model. As long as K is large enough, the obtained results would be similar.

Sparse factor analysis (Bayesian Lasso shrinkage prior)

- Hierarchical Bayesian model (Xing et al., SIIMS 2012):

$$
\begin{aligned}
\boldsymbol{x}_i &\sim \mathcal{N}(\mathbf{D}\boldsymbol{s}_i, \alpha^{-1}\mathbf{I}_P), & s_{ik} &\sim \mathcal{N}(0, \alpha^{-1}\eta_{ik}) \\
\boldsymbol{d}_k &\sim \mathcal{N}(0, P^{-1}\mathbf{I}_P), & \eta_{ik} &\sim \text{Exp}(\gamma_{ik}/2) \\
\alpha &\sim \text{Gamma}(a_0, b_0), & \gamma_{ik} &\sim \text{Gamma}(a_1, b_1)
\end{aligned}
$$

- Marginalizing out η_{ik} leads to

$$
P(s_{ik} \,|\, \alpha, \gamma_{ik}) = \frac{\sqrt{\alpha\gamma_{ik}}}{2} \exp(-\sqrt{\alpha\gamma_{ik}}\,|\,s_{ik}|)
$$

- This Bayesian Lasso shrinkage prior based sparse factor model does not correspond to a nonparametric Bayesian model as $K \to \infty$. Thus the number of dictionary atoms K needs to be carefully set.

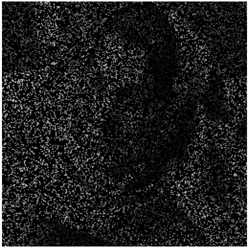
- Logarithm of the posterior

$$
\begin{aligned}
-\log\ p(\boldsymbol{\Theta}\,|\,\mathbf{X},\mathcal{H}) = \ & \frac{\alpha}{2}\sum_{i=1}^{N}\|\boldsymbol{x}_i - \mathbf{D}\boldsymbol{s}_i\|_2^2 \\
& + \frac{P}{2}\sum_{k=1}^{K}\|\boldsymbol{d}_k\|_2^2 \\
& + \sum_{i=1}^{N}\sum_{k=1}^{K}\sqrt{\alpha\gamma_{ik}}|s_{ik}| \\
& - \log f(\alpha, \{\gamma_{ik}\}_{i,k}; \mathcal{H})
\end{aligned}
$$

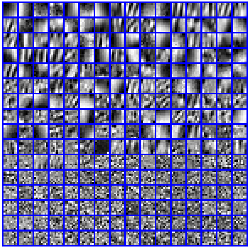
- This model tries to minimize the least squares of the data fitting errors while encouraging the representations $\boldsymbol{s}_i$ to be sparse using L_1 penalties.

Nonparametric Bayesian dictionary learning

- Automatically decide the dictionary size K.
- Automatically decide the sparsity level for each image patch.
- Automatically decide the noise variance.
- Simple to handle data anomalies.
- Insensitive to initialization, does not requires a pertained dictionary.
- Assumption: image patches are fully exchangeable.



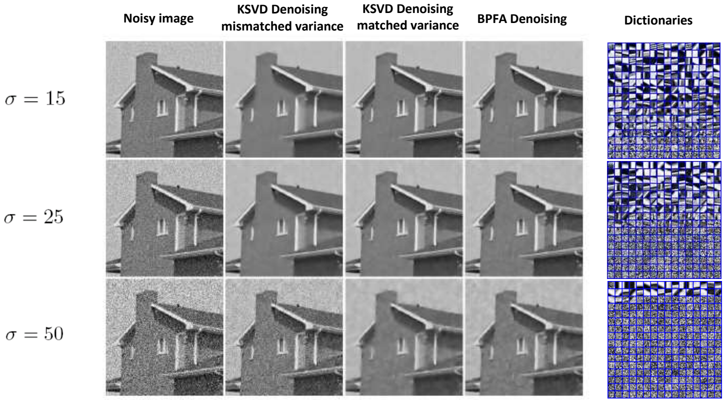
80% pixels missing at random



Learned dictionary

Recovered image (26.90 dB)

Image denoising



Original Noisy Image (dB)	K-SVD Denoising mismatched variance (dB)	K-SVD Denoising matched variance (dB)	Beta Process Denoising (dB)
24.58	30.67	34.32	34.52
20.19	31.52	32.15	32.19
14.56	19.60	27.95	27.95

Image denoising

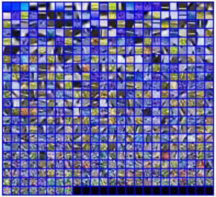
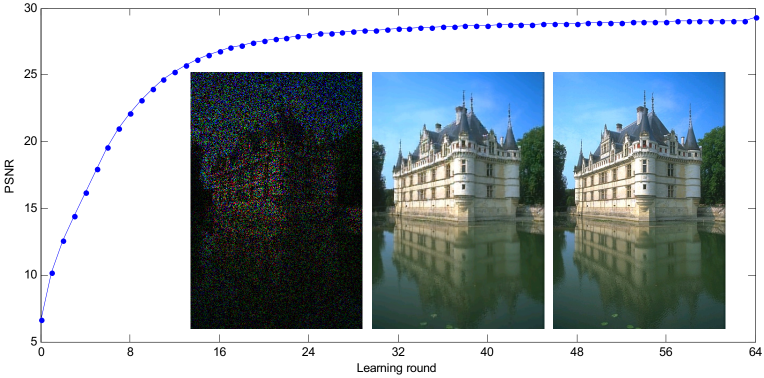


Image inpainting

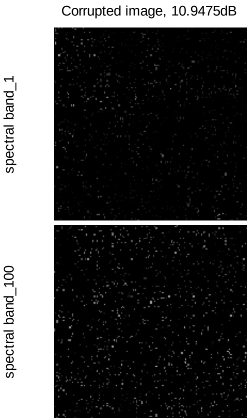
Left to right: corrupted image (80% pixels missing at random), restored image, original image



Hyperspectral image inpainting

$150 \times 150 \times 210$ hyperspectral urban image

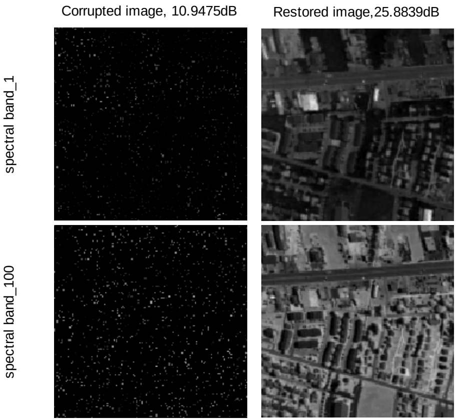
95% voxels missing at random



Hyperspectral image inpainting

$150 \times 150 \times 210$ hyperspectral urban image

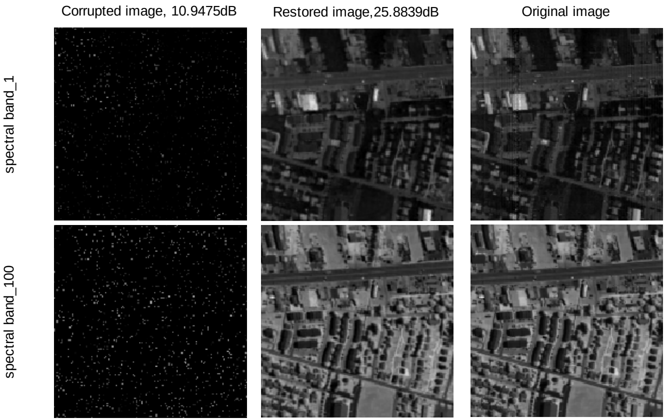
95% voxels missing at random



Hyperspectral image inpainting

$150 \times 150 \times 210$ hyperspectral urban image

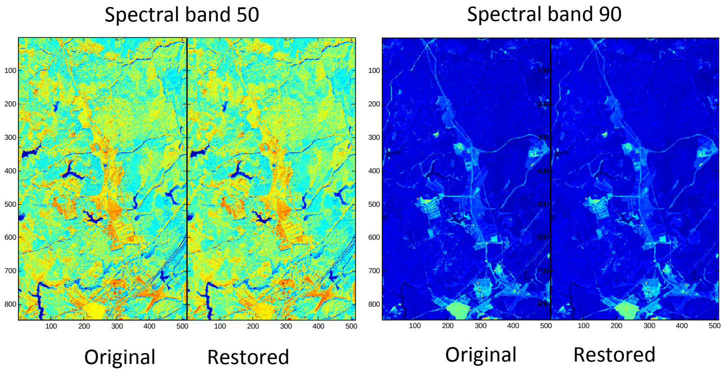
95% voxels missing at random



Hyperspectral image inpainting

$845 \times 512 \times 106$ hyperspectral image

98% voxels missing at random

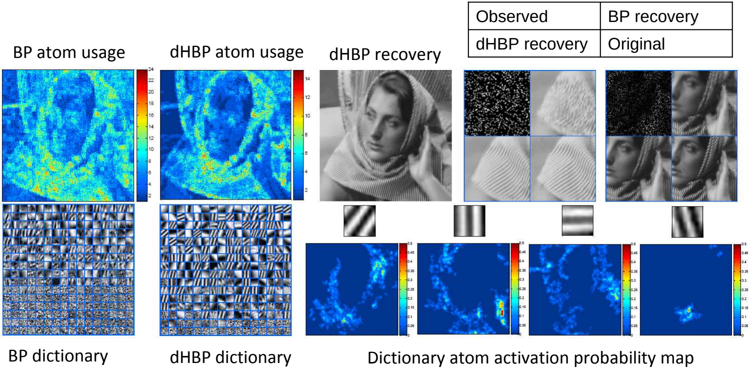


Exchangeable assumption is often not true

- Image patches spatially nearby tend to share similar features
- Left: patches are treated as exchangeable.
  Right: spatial covariate dependence is considered

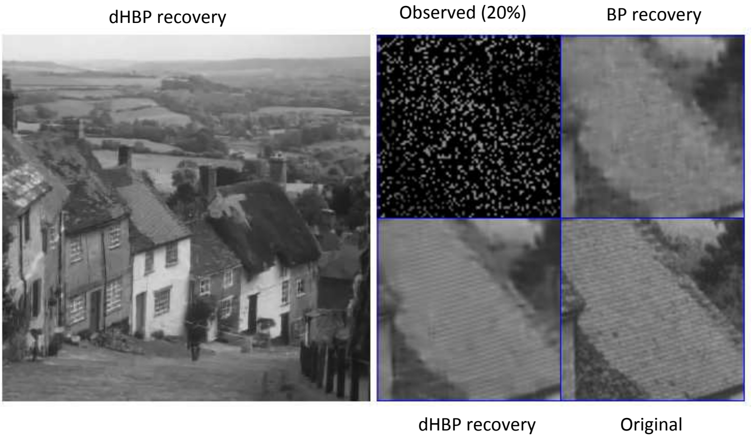
Covariate dependent dictionary learning (Zhou et al., 2011)

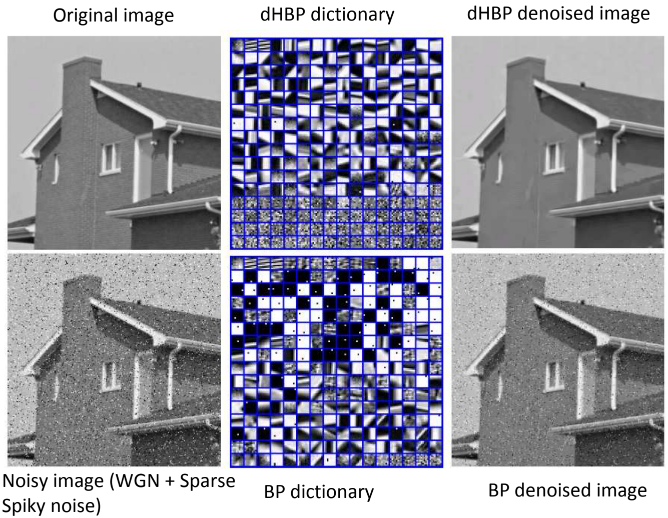
Idea: encouraging data nearby in the covariate space to share similar features.

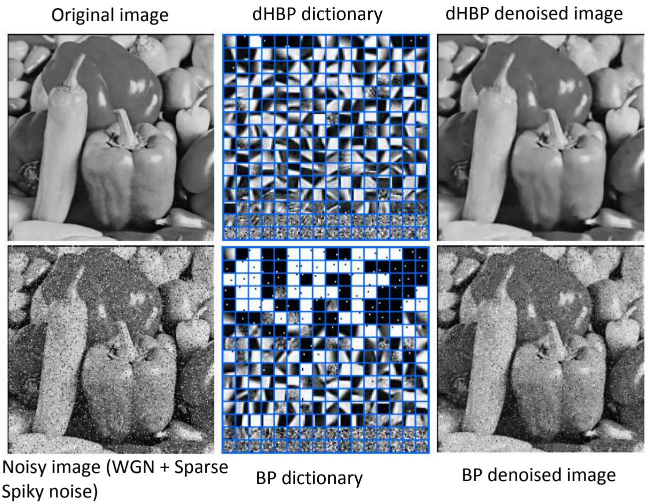


dHBP recovery Observed (20%) BP recovery

dHBP recovery Original


dHBP recovery
Observed (20%)
BP recovery
dHBP recovery
Original


Original image
dHBP dictionary
dHBP denoised image
Noisy image (WGN + Sparse Spiky noise)
BP dictionary
BP denoised image


Original image
dHBP dictionary
dHBP denoised image
Noisy image (WGN + Sparse Spiky noise)
BP dictionary
BP denoised image

Summary for Bayesian dictionary learning

- A generative approach for data recovery from redundant noisy and incomplete observations.
- A single baseline model applicable for all: gray-scale, RGB, and hyperspectral image denoising and inpainting.
- Automatically inferred noise variance and sparsity level and dictionary size.
- Dictionary learning and reconstruction on the data under test.
- Incorporate covariate dependence.
- Code available online for reproducible research.
- In a sampling based algorithm, the spike-and-slab sparse prior allows the representations to be exactly zero, whereas a shrinkage prior would not permit exactly zeros; for dictionary learning, the sparse-and-slab prior is often found to be more robust, be easier to compute, and performs better.

- Understand your data
- Define data likelihood
- Construct prior
- Derive inference using MCMC or Variational Bayes
- Implement in Matlab, R, Python, C/C++, ...
- Interpret model output

M. Aharon, M. Elad, and A. M. Bruckstein.
K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation.
IEEE Trans. Signal Processing, 2006.

M. Elad and M. Aharon.
Image denoising via sparse and redundant representations over learned dictionaries.
IEEE Trans. Image Processing, 2006.

T.L. Griffiths and Z. Ghahramani.
Infinite latent feature models and the Indian buffet process.
In *Proc. Advances in Neural Information Processing Systems*, pages 475–482, 2005.

R. Thibaux and M. I. Jordan.
Hierarchical beta processes and the Indian buffet process.
In *Proc. International Conference on Artificial Intelligence and Statistics*, 2007.

P. Trevor and G. Casella.
The Bayesian lasso.
Journal of the American Statistical Association, 2008.

Z. Xing, M. Zhou, A. Castrodad, G. Sapiro and L. Carin.
Dictionary learning for noisy and incomplete hyperspectral images.
SIAM Journal on Imaging Sciences, 2012

M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin.
Non-parametric Bayesian dictionary learning for sparse image representations.
In *NIPS*, 2009.

M. Zhou, H. Chen, J. Paisley, L. Ren, L. Li, Z. Xing, D. Dunson, G. Sapiro, and L. Carin.
Nonparametric Bayesian dictionary learning for analysis of noisy and incomplete images.
IEEE TIP, 2012.

M. Zhou, H. Yang, G. Sapiro, D. Dunson, and L. Carin.
Dependent hierarchical beta process for image interpolation and denoising.
In *AISTATS*, 2011.