Recent directions in nonparametric Bayesian machine learning

Zoubin Ghahramani

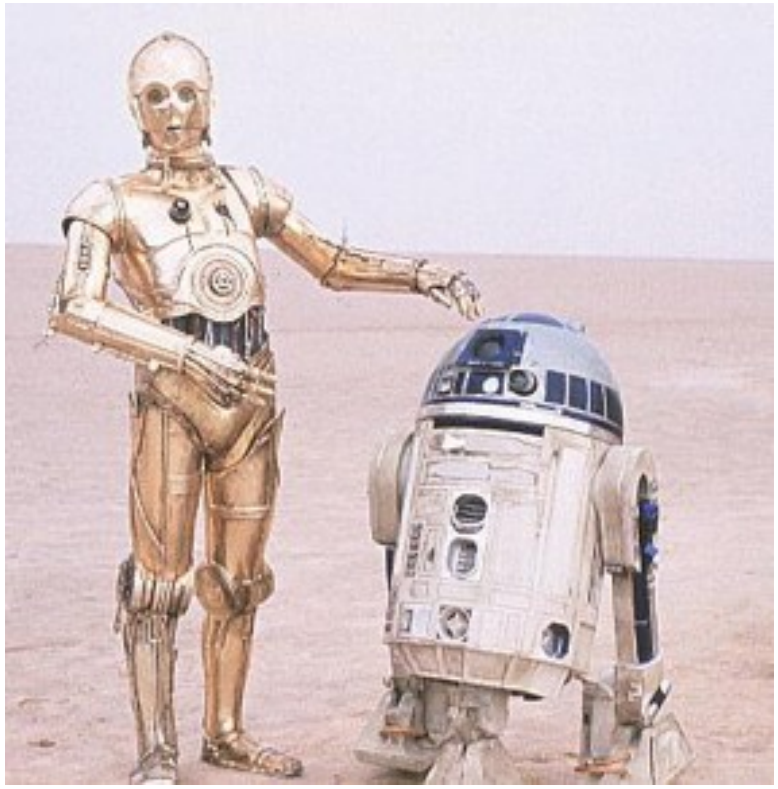
Department of Engineering
University of Cambridge, UK

Machine Learning Department
Carnegie Mellon University, USA

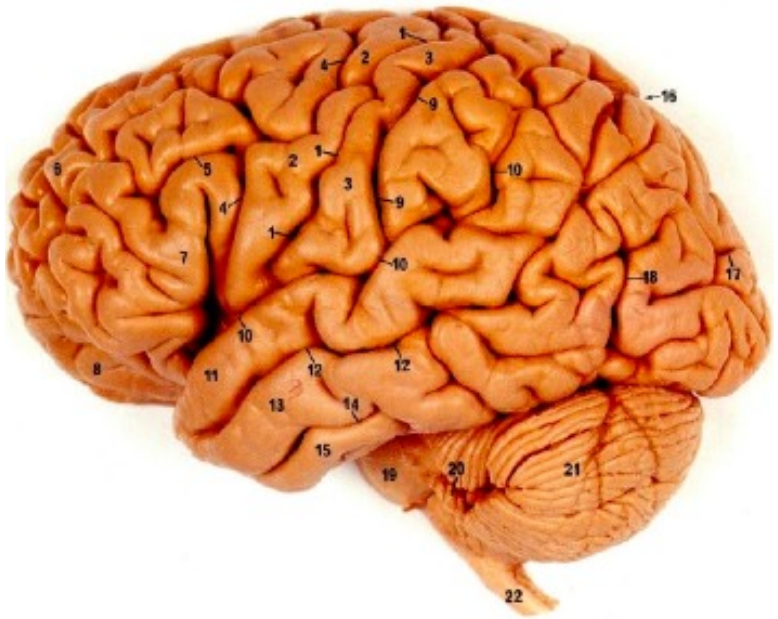
zoubin@eng.cam.ac.uk
http://learning.eng.cam.ac.uk/zoubin/

March 2008

How do we build thinking machines?



How do we understand biological thinking machines?



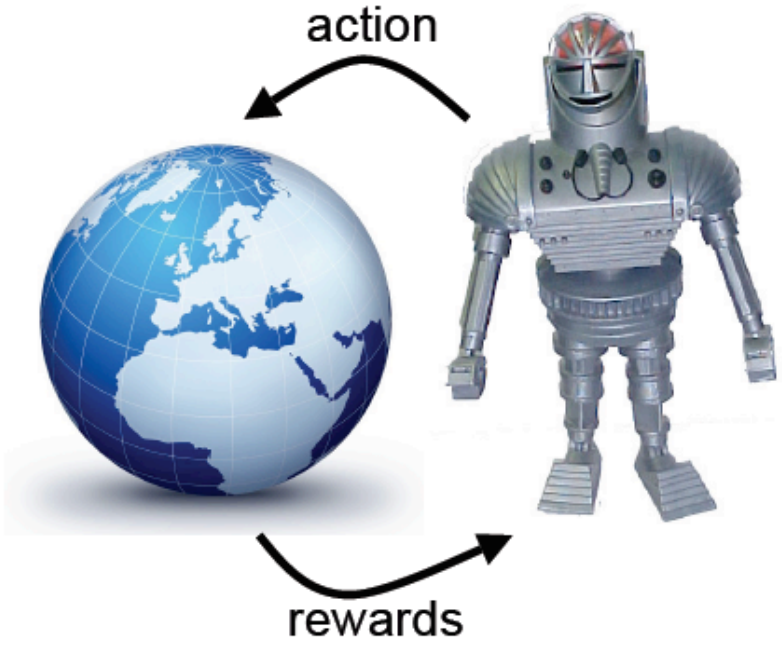
Machine Learning

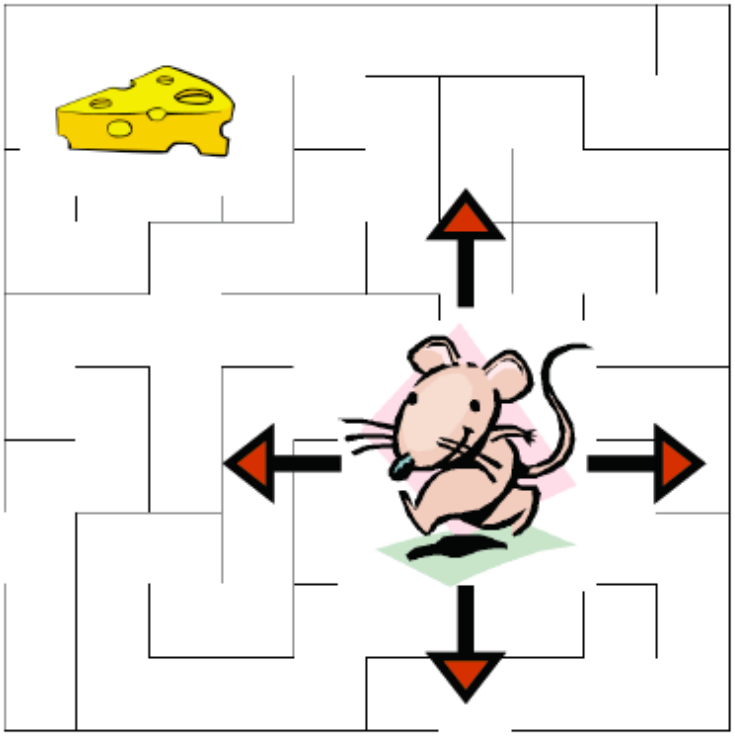
- *Machine learning is an interdisciplinary field studying both the mathematical foundations and practical applications of systems that learn, reason and act.*

- Other related terms: Pattern Recognition, Neural Networks, Data Mining, Statistical Modelling ...

- Using ideas from: Statistics, Computer Science, Engineering, Applied Mathematics, Cognitive Science, Psychology, Computational Neuroscience, Economics

A framework





Bellman's Optimality Equation

$$V^*(s) = R(s) + \max_a \gamma \sum_{s'} P(s'|s,a) V^*(s')$$

s, s'	states
a	an action
$R(s)$	the immediate reward in state s
$P(s'\|s,a)$	prob. of transitioning to s' from s after action a
γ	discount factor on future rewards
$V^*(s)$	optimal *value* of state s

Richard E. Bellman (1920-1984)

- Bellman's equation tells us how to use a model of the environment to convert immediate rewards into values.

- Using those values, we can solve the sequential decision making problem.

Bayes Rule

$$P(\text{hypothesis}|\text{data}) = \frac{P(\text{data}|\text{hypothesis})P(\text{hypothesis})}{P(\text{data})}$$

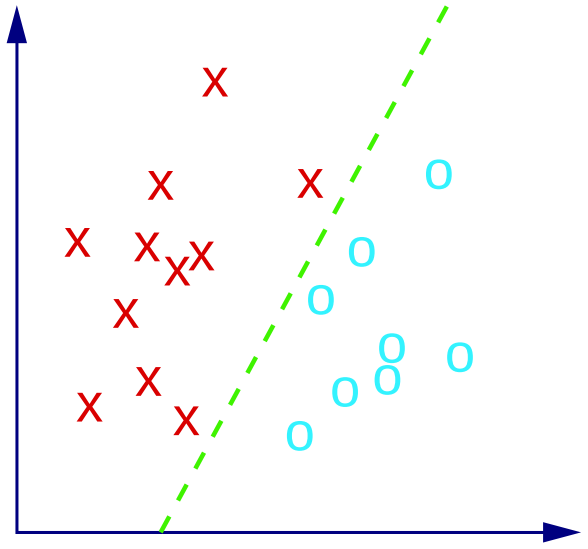
Rev'd Thomas Bayes (1702–1761)

- Bayes rule tells us how to do inference about hypotheses from data.
- Learning and prediction can be seen as forms of inference.

Linear Classification

Data: $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}$ for $n = 1, \ldots, N$ data points

$$\begin{aligned} \mathbf{x}^{(n)} &\in \Re^D \\ y^{(n)} &\in \{+1, -1\} \end{aligned}$$



Parameters: $\boldsymbol{\theta} \in \Re^{D+1}$

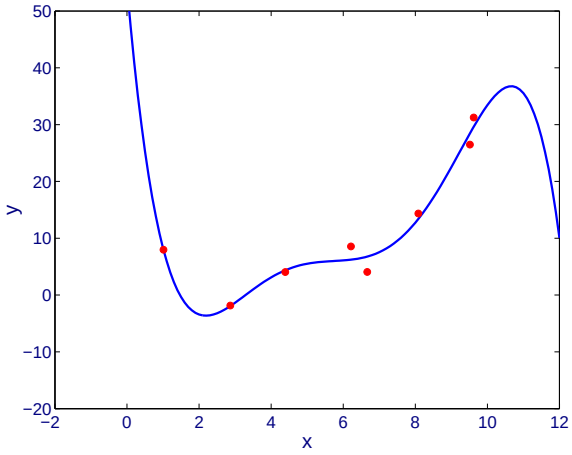
$$P(y^{(n)} = +1 | \boldsymbol{\theta}, \mathbf{x}^{(n)}) = \begin{cases} 1 & \text{if } \sum_{d=1}^{D} \theta_d \, x_d^{(n)} + \theta_0 \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Goal: To infer $\boldsymbol{\theta}$ from the data and to predict future labels $P(y|\mathcal{D}, \mathbf{x})$

Polynomial Regression

Data: $\mathcal{D} = \{(x^{(n)}, y^{(n)})\}$ for $n = 1, \ldots, N$

$$\begin{aligned} x^{(n)} &\in \Re \\ y^{(n)} &\in \Re \end{aligned}$$



Parameters: $\boldsymbol{\theta} = (a_0, \ldots, a_m, \sigma)$

Model:

$$y^{(n)} = a_0 + a_1 x^{(n)} + a_2 {x^{(n)}}^2 \ldots + a_m {x^{(n)}}^m + \epsilon$$

where

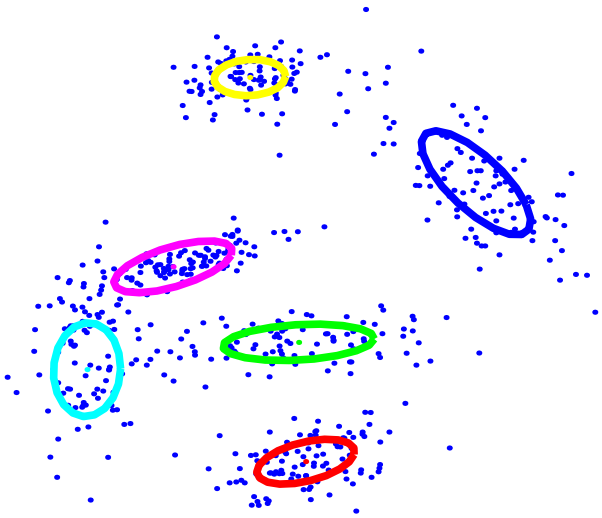
$$\epsilon \sim \mathcal{N}(0, \sigma^2)$$

Goal: To infer $\boldsymbol{\theta}$ from the data and to predict future outputs $P(y|\mathcal{D}, x, m)$

Clustering with Gaussian Mixtures (Density Estimation)

Data: $\mathcal{D} = \{\mathbf{x}^{(n)}\}$ for $n = 1, \ldots, N$

$$\mathbf{x}^{(n)} \in \Re^D$$



Parameters: $\boldsymbol{\theta} = \left((\boldsymbol{\mu}^{(1)}, \Sigma^{(1)}) \ldots, (\boldsymbol{\mu}^{(m)}, \Sigma^{(m)}), \boldsymbol{\pi}\right)$

Model:

$$\mathbf{x}^{(n)} \sim \sum_{i=1}^{m} \pi_i \, p_i(\mathbf{x}^{(n)})$$

where

$$p_i(\mathbf{x}^{(n)}) = \mathcal{N}(\boldsymbol{\mu}^{(i)}, \Sigma^{(i)})$$

Goal: To infer $\boldsymbol{\theta}$ from the data and predict the density $p(\mathbf{x}|\mathcal{D}, m)$

Bayesian Machine Learning

$$P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})}$$

$P(\mathcal{D}\|\theta)$	likelihood of θ
$P(\theta)$	prior probability of θ
$P(\theta\|\mathcal{D})$	posterior of θ given $\mathcal{D}$

Model Comparison:

$$\begin{aligned} P(m|\mathcal{D}) &= \frac{P(\mathcal{D}|m)P(m)}{P(\mathcal{D})} \\ P(\mathcal{D}|m) &= \int P(\mathcal{D}|\theta, m)P(\theta|m)\, d\theta \end{aligned}$$

Prediction:

$$P(x|\mathcal{D}, m) = \int P(x|\theta, \mathcal{D}, m)P(\theta|\mathcal{D}, m)d\theta$$

Bayesian Occam's Razor and Model Comparison

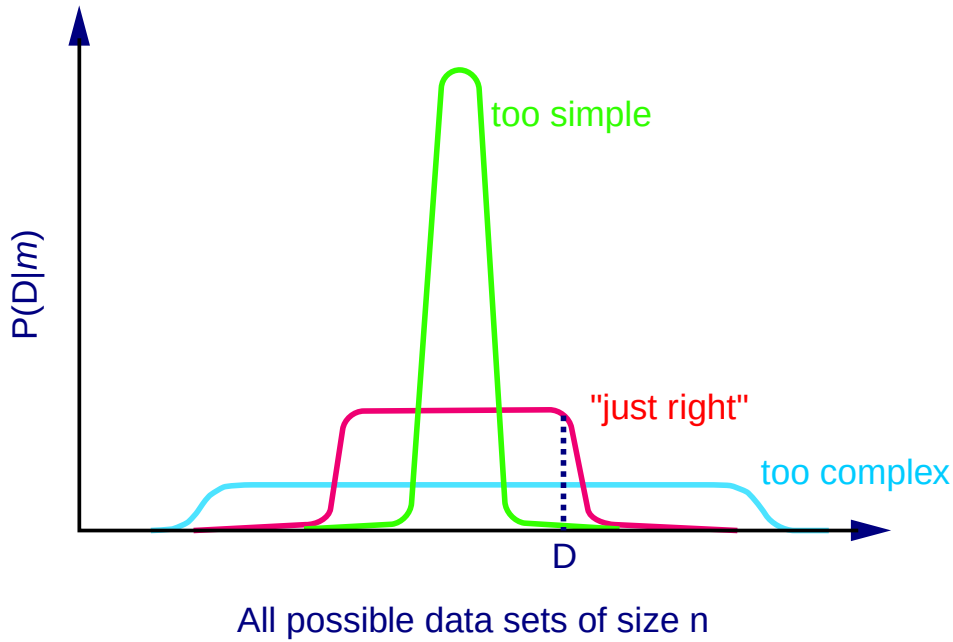
Compare model classes, e.g. m and m', using posterior probabilities given $\mathcal{D}$:

$$p(m|\mathcal{D}) = \frac{p(\mathcal{D}|m)\,p(m)}{p(\mathcal{D})}, \qquad p(\mathcal{D}|m) = \int p(\mathcal{D}|\boldsymbol{\theta}, m)\; p(\boldsymbol{\theta}|m)\; d\boldsymbol{\theta}$$

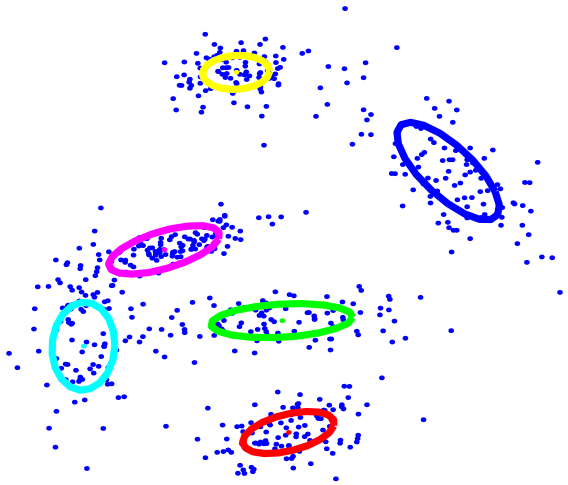
Interpretation of the Marginal Likelihood ("evidence"): The probability that *randomly selected* parameters from the prior would generate $\mathcal{D}$.

Model classes that are too simple are unlikely to generate the data set.

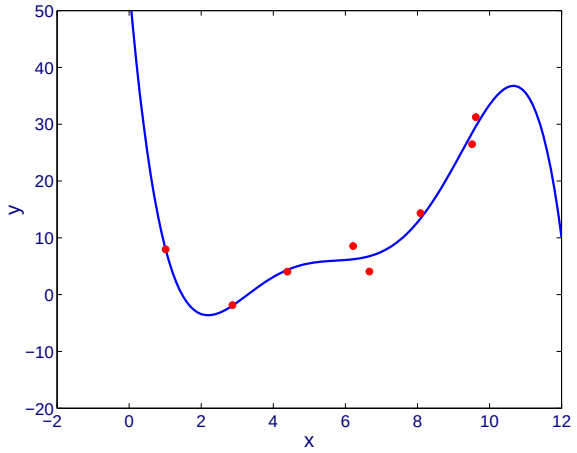
Model classes that are too complex can generate many possible data sets, so again, they are unlikely to generate that particular data set at random.



Model Comparison: two examples



e.g. selecting m, the number of Gaussians in a mixture model



e.g. selecting m the order of a polynomial in a nonlinear regression model

$$P(m|\mathcal{D}) = \frac{P(\mathcal{D}|m)P(m)}{P(\mathcal{D})}, \qquad P(\mathcal{D}|m) = \int P(\mathcal{D}|\theta, m)P(\theta|m)\, d\theta$$

A possible procedure:

1. place a prior on m, $P(m)$
2. given data, use Bayes rule to infer $P(m|\mathcal{D})$

What is the problem with this procedure?

Real data are complicated

Example 1:

You are trying to model people's patterns of movie preferences. You believe there are "clusters" of people, so you use a mixture model...

- How should you pick $P(m)$, your prior over how many clusters there are? teenagers, people who like action movies, people who like romantic comedies, people who like horror movies, people who like movies with Marlon Brando, people who like action movies but not science fiction, etc etc...

- Even if there are a few well defined clusters, they are unlikely to be Gaussian in the variables you measure. To model complicated distributions you might need many Gaussians for each cluster.

- Conclusion: any small finite number m seems unreasonable

Real data are complicated

Example 2:

You are trying to model crop yield as a function of rainfall, amount of sunshine, amount of fertilizer, etc. You believe this relationship is nonlinear, so you decide to model it with a polynomial.

- How should you pick $P(m)$, your prior over the order of the polynomial?
- Do you believe the relationship could be linear? quadratic? cubic? What about possible interactions between input variabes?
- Conclusion: any order polynomial seems unreasonable.

How do we adequately capture our beliefs?

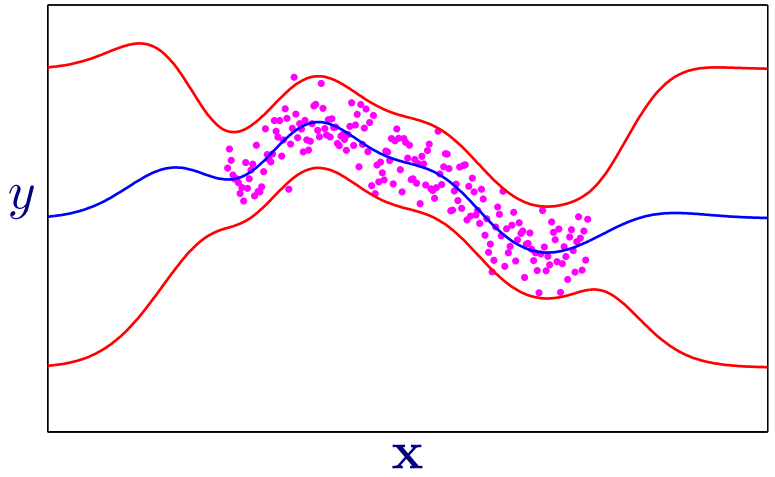
Non-parametric Bayesian Models

- Bayesian methods are most powerful when your prior adequately captures your beliefs.
- Inflexible models (e.g. mixture of 5 Gaussians, 4th order polynomial) yield unreasonable inferences.
- Non-parametric models are a way of getting very flexible models.
- Many can be derived by starting with a finite parametric model and taking the limit as number of parameters $\rightarrow \infty$
- Non-parametric models can automatically infer an adequate model size/complexity from the data, without needing to explicitly do Bayesian model comparison.[1]

[1] Even if you believe there are infinitely many possible clusters, you can still infer how many clusters are *represented* in a finite set of n data points.

Nonlinear regression and Gaussian processes

Consider the problem of nonlinear regression:
You want to learn a function f with error bars from data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$



A Gaussian process defines a distribution over functions $p(f)$ which can be used for Bayesian regression:

$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})}$$

Let $\mathbf{f} = (f(x_1), f(x_2), \ldots, f(x_n))$ be an n-dimensional vector of function values evaluated at n points $x_i \in \mathcal{X}$. Note, $\mathbf{f}$ is a random variable.

Definition: $p(f)$ is a Gaussian process if for *any* finite subset $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the marginal distribution over that subset $p(\mathbf{f})$ is multivariate Gaussian.

Gaussian process covariance functions

$p(f)$ is a Gaussian process if for *any* finite subset $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the marginal distribution over that finite subset $p(\mathbf{f})$ has a multivariate Gaussian distribution.

Gaussian processes (GPs) are parameterized by a mean function, $\mu(x)$, and a covariance function, $c(x, x')$.

$$P(f(x), f(x')) = \mathsf{N}(\boldsymbol{\mu}, \Sigma)$$

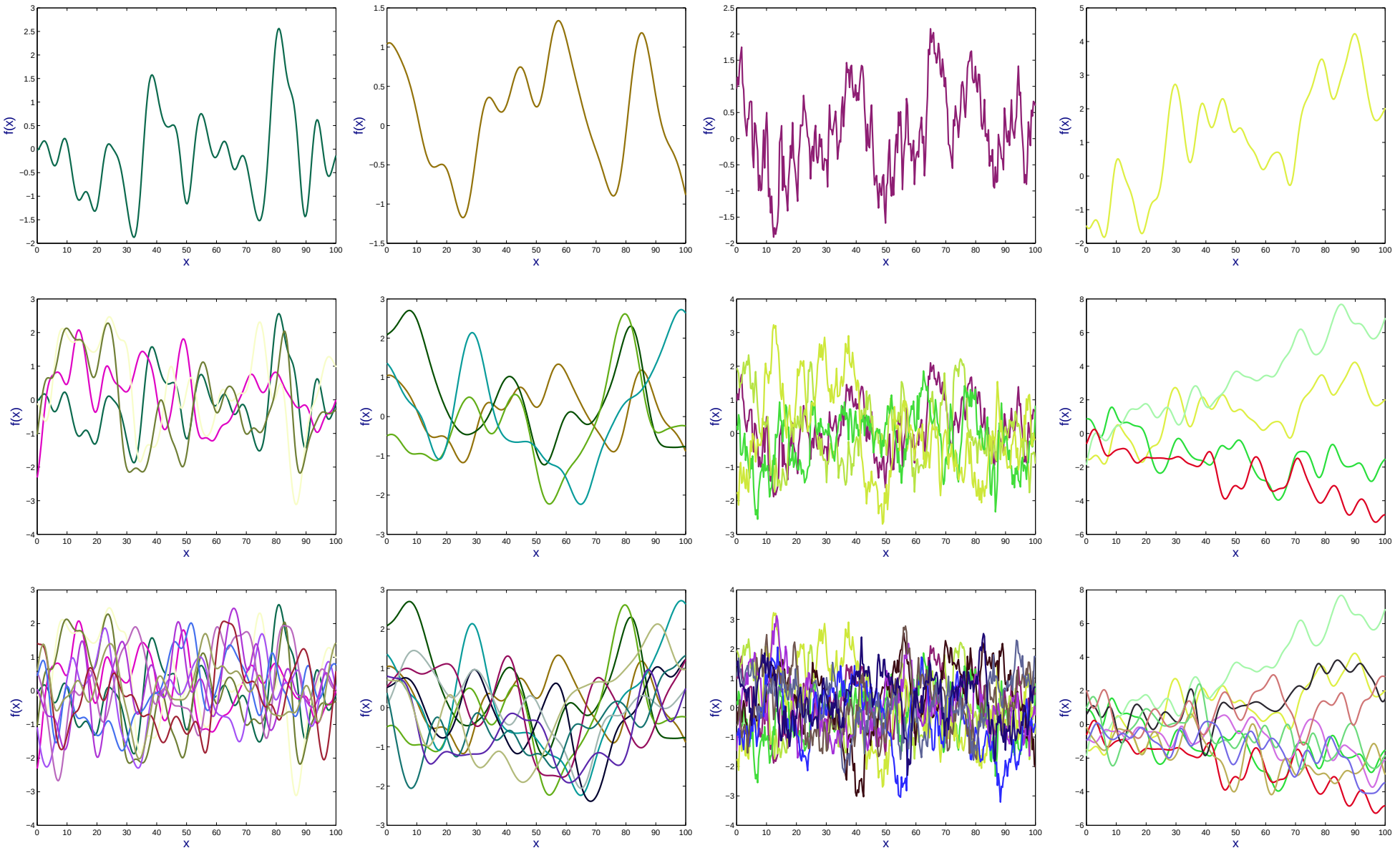
where $$\boldsymbol{\mu} = \begin{bmatrix} \mu(x) \\ \mu(x') \end{bmatrix} \quad \Sigma = \begin{bmatrix} c(x,x) & c(x,x') \\ c(x',x) & c(x',x') \end{bmatrix}$$

and similarly for $P(f(x_1), \ldots, f(x_n))$ where now $\boldsymbol{\mu}$ is an $n \times 1$ vector and Σ is an $n \times n$ matrix.

E.g.: $c(x_i, x_j) = v_0 \exp\left\{ -\left(\frac{|x_i - x_j|}{\lambda} \right)^{\alpha} \right\} + v_1 + v_2\, \delta_{ij}$ with params $(v_0, v_1, v_2, \lambda, \alpha)$

Once the mean and covariance functions are defined, everything else about GPs follows from the basic rules of probability applied to mutivariate Gaussians.

Samples from Gaussian processes with different $c(x, x')$



Using Gaussian processes for nonlinear regression

Imagine observing a data set $\mathcal{D} = \{(x_i, y_i)_{i=1}^n\} = (\mathbf{x}, \mathbf{y})$.

Model:

$$\begin{aligned} y_i &= f(x_i) + \epsilon_i \\ f &\sim \mathsf{GP}(\cdot|0, c) \\ \epsilon_i &\sim \mathsf{N}(\cdot|0, \sigma^2) \end{aligned}$$

Prior on f is a GP, likelihood is Gaussian, therefore posterior on f is also a GP.

We can use this to make predictions

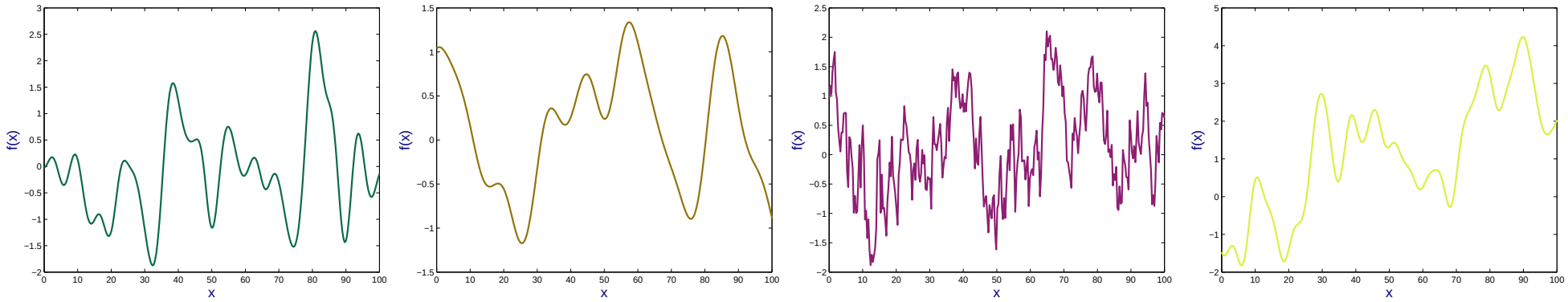
$$P(y'|x', \mathcal{D}) = \int df P(y'|x', f, \mathcal{D}) P(f|\mathcal{D})$$

We can also compute the marginal likelihood (evidence) and use this to compare or tune covariance functions

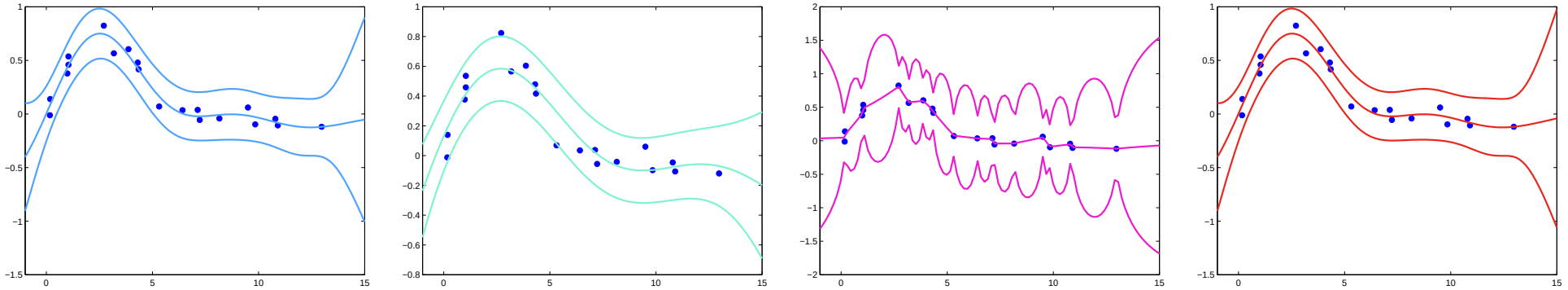
$$P(\mathbf{y}|\mathbf{x}) = \int df P(\mathbf{y}|f, \mathbf{x}) P(f)$$

Prediction using GPs with different $c(x, x')$

A sample from the prior for each covariance function:



Corresponding predictions, mean with two standard deviations:



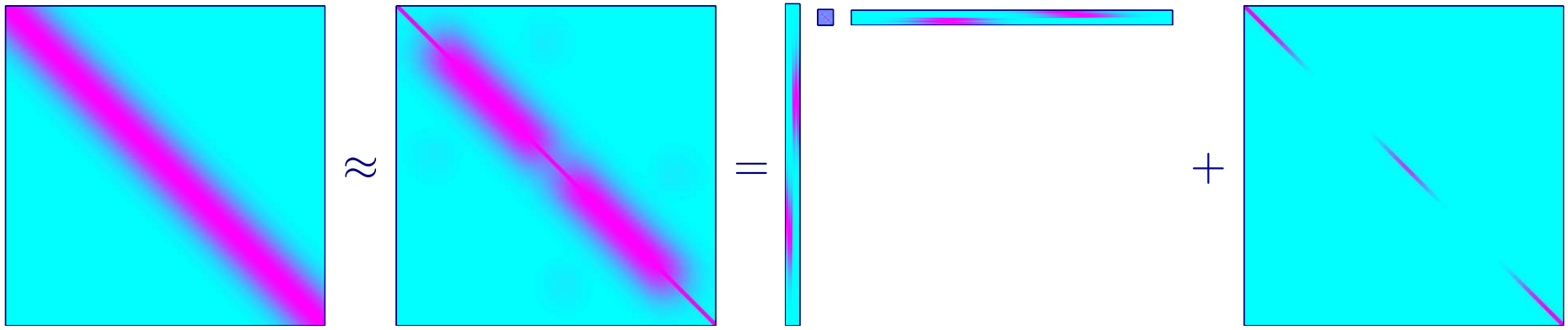
Sparse Approximations to Gaussian processes

(Snelson and Ghahramani, 2005, 2006, 2007)

Problem: $\mathcal{O}(N^3)$ computations for learning and prediction.
We can approximate GP through $M < N$ inducing points $\bar{\mathbf{f}}$ to obtain:

$$p(\mathbf{f}) = \int \mathrm{d}\bar{\mathbf{f}} \prod_n p(f_n|\bar{\mathbf{f}})\, p(\bar{\mathbf{f}})$$

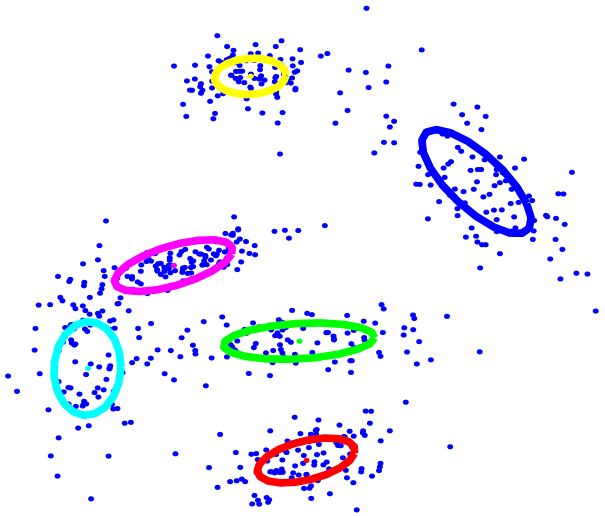
GP covariance $\mathcal{N}(\mathbf{0}, \mathbf{K}_N)$ $\approx$ $p(\mathbf{f})$ $=$ Approximation $\mathcal{N}(\mathbf{0}, \mathbf{K}_{NM}\mathbf{K}_M^{-1}\mathbf{K}_{MN} + \mathbf{\Lambda})$



- Approximate covariance inverted in $\mathcal{O}(M^2N)$
- Given data $\{\mathbf{X}, \mathbf{y}\}$ with noise σ^2, predictive mean and variance can be computed in $\mathcal{O}(M)$ and $\mathcal{O}(M^2)$ per test case respectively

Clustering

Basic idea: each data point belongs to a cluster

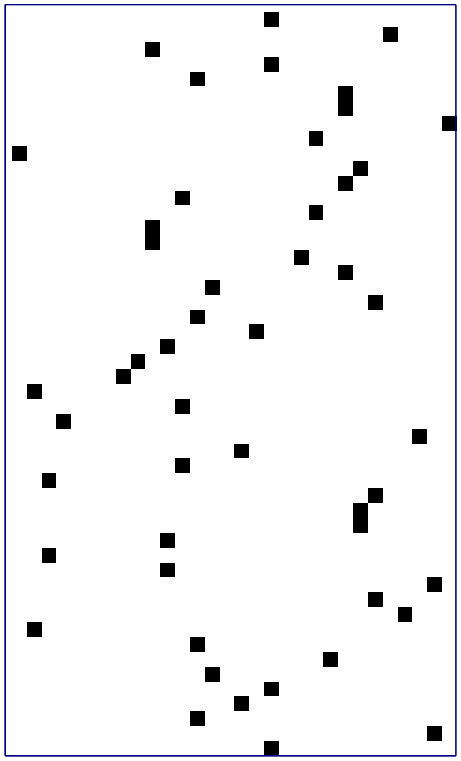


Many clustering methods exist:

- mixture models
- hierarchical clustering
- spectral clustering

Goal: to partition data into groups in an unsupervised manner

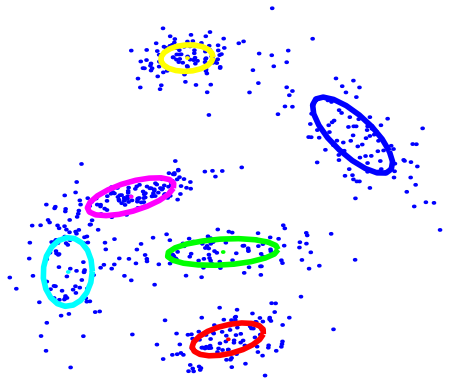
A binary matrix representation for clustering



- Rows are data points
- Columns are clusters
- Since each data point is assigned to one and only one cluster, the rows sum to one.
- Finite mixture models: number of columns is finite
- Infinite mixture models: number of columns is countably infinite

Infinite mixture models

(e.g. Dirichlet Process Mixtures)



Why?

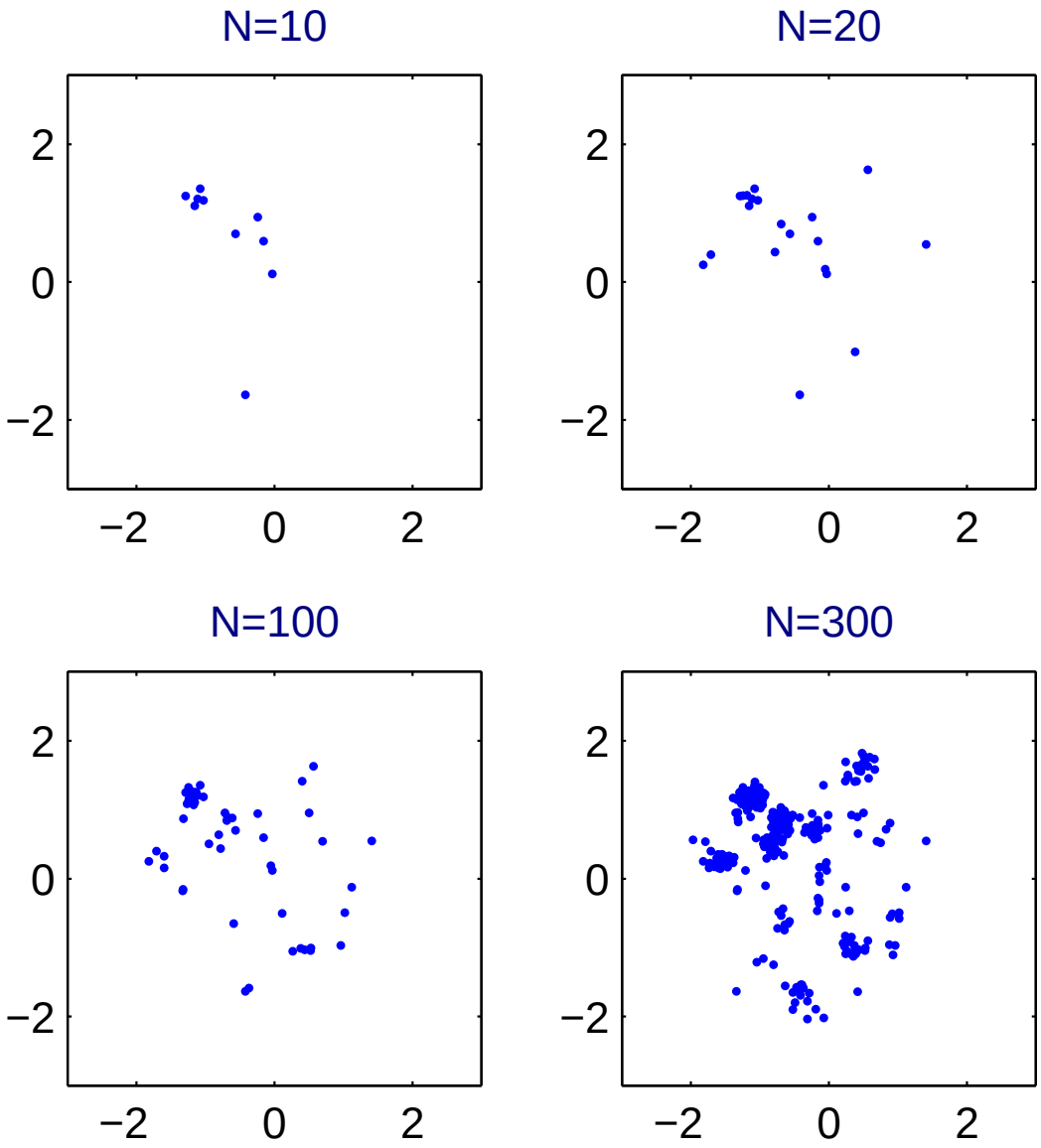
- You might not believe a priori that your data comes from a finite number of mixture components (e.g. strangely shaped clusters; heavy tails; structure at many resolutions)

- Inflexible models (e.g. a mixture of 6 Gaussians) can yield unreasonable inferences and predictions.

- For many kinds of data, the number of clusters might grow over time: clusters of news stories or emails, classes of objects, etc.

- You might want your method to automatically infer the number of clusters in the data.

Samples from a Dirichlet Process Mixture of Gaussians



Notice that more structure (clusters) appear as you draw more points.

(figure inspired by Neal)

Infinite mixture models

$$p(x) = \sum_{k=1}^{K} \pi_k \, p_k(x)$$

How?

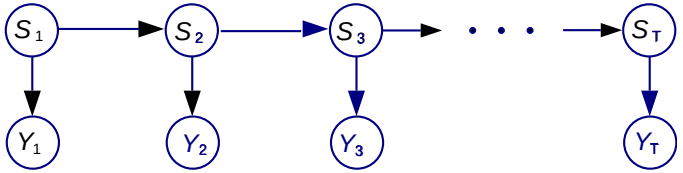
- Start from a finite mixture model with K components and take the limit[2] as number of components $K \to \infty$
- But you have infinitely many parameters!
- Rather than optimize the parameters (ML, MAP), you integrate them out (Bayes) using, e.g:
  - MCMC sampling (Escobar & West 1995; Neal 2000; Rasmussen 2000)
  - expectation propagation (EP; Minka and Ghahramani, 2003)
  - variational methods (Blei and Jordan, 2005)
  - Bayesian hierarchical clustering (Heller and Ghahramani, 2005)

[2] Dirichlet Process Mixtures; Chinese Restaurant Processes

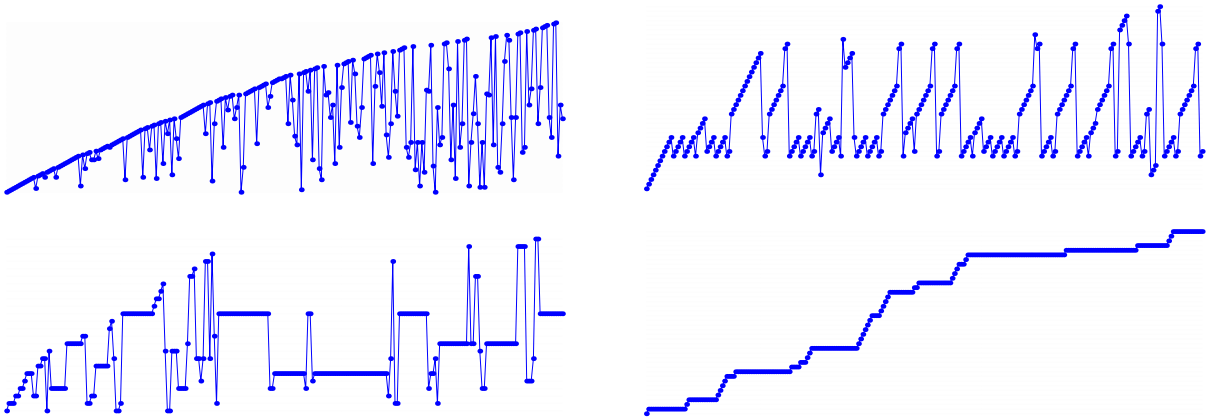
Infinite hidden Markov models (iHMMs)

Hidden Markov models (HMMs) can be thought of as time-dependent mixtures.

In an HMM with K states, the transition matrix has $K \times K$ elements.

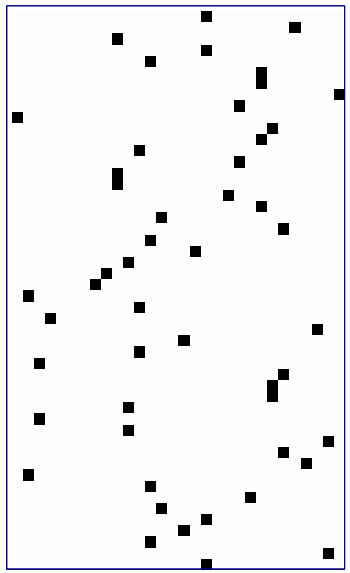


We let $K \to \infty$, this results in an iHMM.



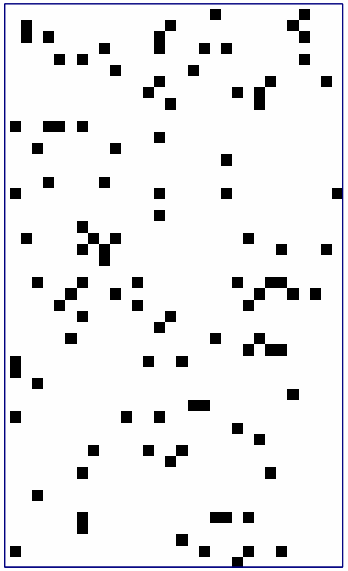
- Introduced in (Beal, Ghahramani and Rasmussen, 2002).
- Teh, Jordan, Beal and Blei (2005) showed that iHMMs can be derived from hierarchical Dirichlet processes, and provided a more efficient Gibbs sampler.
- We have recently derived a much more efficient sampler based on Dynamic Programming (Van Gael, Saatci, Teh, and Ghahramani, 2008).

A binary matrix representation for clustering



- Rows are data points
- Columns are clusters
- Since each data point is assigned to one and only one cluster...
- ...the rows sum to one.

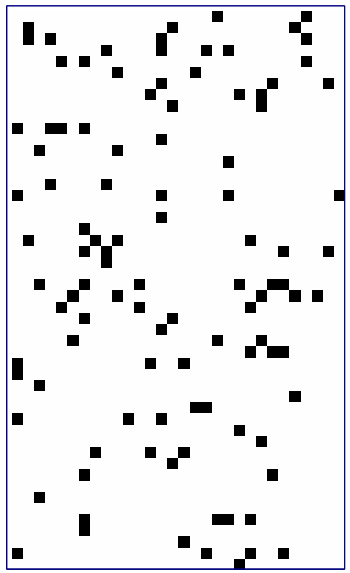
More General Priors on Binary Matrices



- Rows are data points
- Columns are latent features
- We can think of **infinite** binary matrices...
  ...where each data point can now have *multiple* features, so...
  ...the rows can sum to more than one.

(Griffiths and Ghahramani, 2005)

More General Priors on Binary Matrices



Another way of thinking about this:

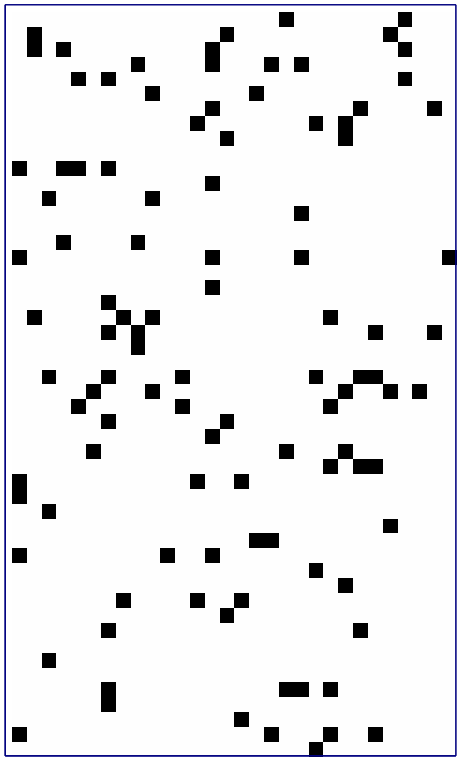
- there are multiple overlapping clusters
- each data point can belong to several clusters simultaneously.

If there are K features, then there are 2^K possible binary latent representations for each data point.

Why?

- Many statistical models can be thought of as modelling 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, they do not have distributed representations.
- Consider modelling 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.

From finite to infinite binary matrices



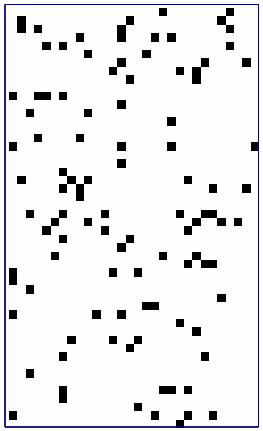
$z_{nk} = 1$ means object n has feature k:

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

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

- Note that $P(z_{nk} = 1|\alpha) = E(\theta_k) = \frac{\alpha/K}{\alpha/K+1}$, so as K grows larger the matrix gets sparser.
- So if $\mathbf{Z}$ is $N \times K$, the expected number of nonzero entries is $N\alpha/(1 + \alpha/K) < N\alpha$.
- Even in the $K \to \infty$ limit, the matrix is expected to have a finite number of non-zero entries.

From finite to infinite binary matrices



We can integrate out θ, leaving:

$$
\begin{aligned}
P(\mathbf{Z}|\alpha) &= \int P(\mathbf{Z}|\theta)P(\theta|\alpha)d\theta \\
&= \prod_k \frac{\Gamma(m_k + \frac{\alpha}{K})\Gamma(N - m_k + 1)}{\Gamma(\frac{\alpha}{K})} \frac{\Gamma(1 + \frac{\alpha}{K})}{\Gamma(N + 1 + \frac{\alpha}{K})}
\end{aligned}
$$

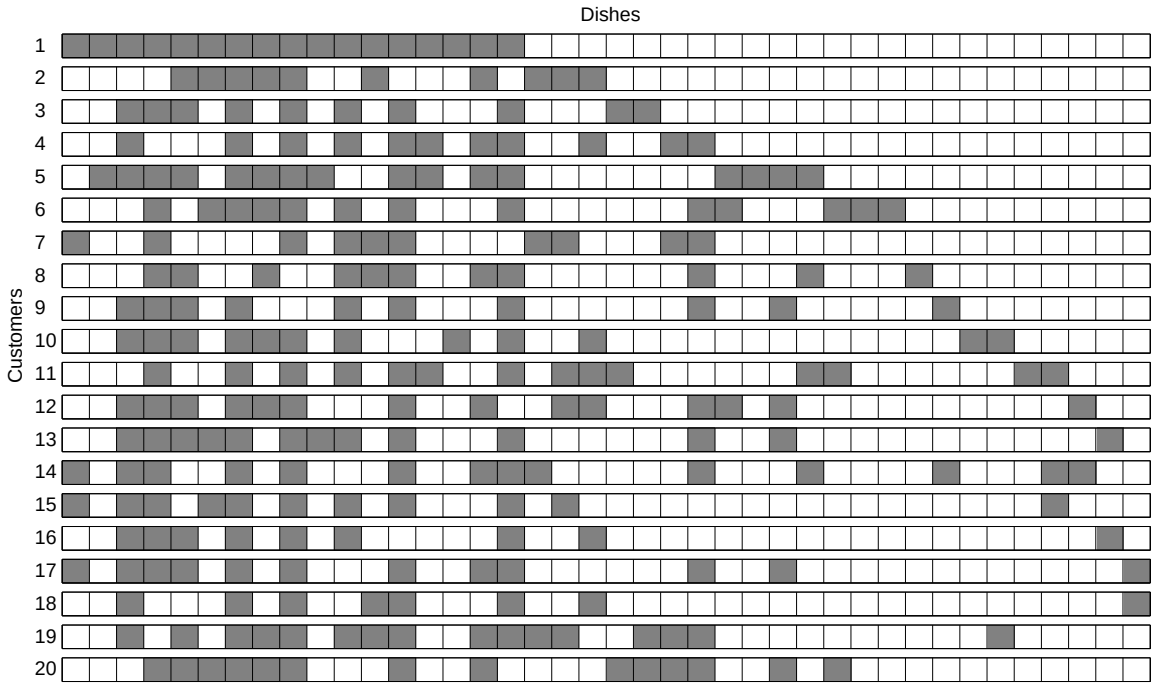
The conditional feature assignments are:

$$
\begin{aligned}
P(z_{nk} = 1|\mathbf{z}_{-n,k}) &= \int_0^1 P(z_{nk}|\theta_k)p(\theta_k|\mathbf{z}_{-n,k})\ d\theta_k \\
&= \frac{m_{-n,k} + \frac{\alpha}{K}}{N + \frac{\alpha}{K}},
\end{aligned}
$$

where $\mathbf{z}_{-n,k}$ is the set of assignments of all objects, not including n, for feature k, and $m_{-n,k}$ is the number of objects having feature k, not including n.
We can take limit as $K \to \infty$.

"Rich get richer", like in Chinese Restaurant Processes.

Indian buffet process



"Many Indian restaurants in London offer lunchtime buffets with an apparently infinite number of dishes"

- First customer starts at the left of the buffet, and takes a serving from each dish, stopping after a Poisson(α) number of dishes as her plate becomes overburdened.
- The nth customer moves along the buffet, sampling dishes in proportion to their popularity, serving himself with probability m_k/n, and trying a Poisson(α/n) number of new dishes.
- The customer-dish matrix is our feature matrix, $\mathbf{Z}$.

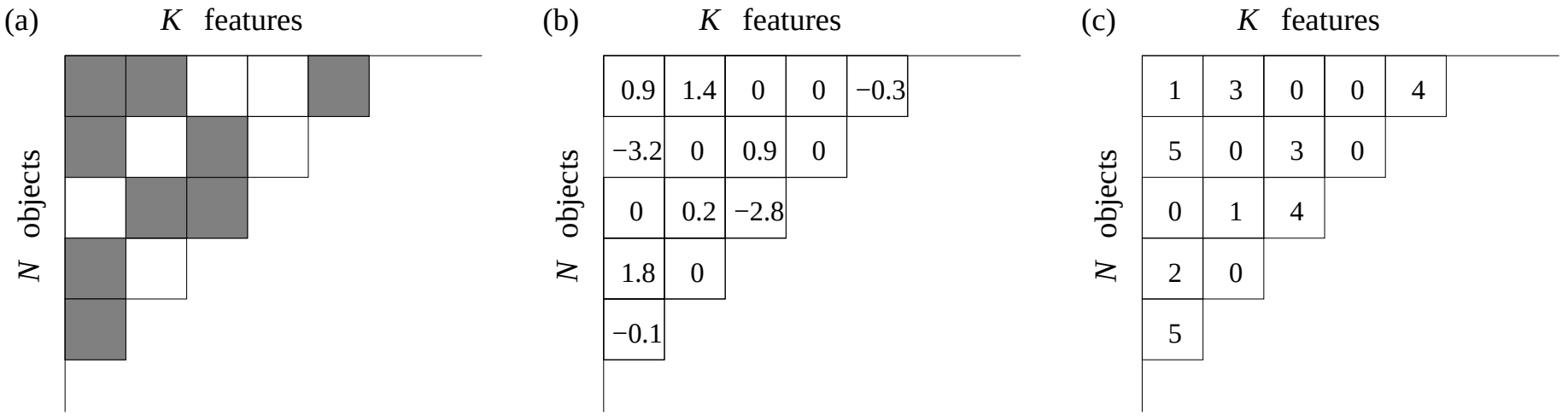
From binary to non-binary latent features

In many models we might want non-binary latent features.

A simple way to generate non-binary latent feature matrices from $\mathbf{Z}$:

$$\mathbf{F} = \mathbf{Z} \otimes \mathbf{V}$$

where $\otimes$ is the elementwise (Hadamard) product of two matrices, and $\mathbf{V}$ is a matrix of independent random variables (e.g. Gaussian, Poisson, Discrete, ...).



What do we do with $\mathbf{Z}$?

Model data.

Modelling Data

Latent variable model: let $\mathbf{X}$ be the $N \times D$ matrix of observed data, and $\mathbf{Z}$ be the $N \times K$ matrix of binary latent features

$$P(\mathbf{X}, \mathbf{Z}|\alpha, \beta) = P(\mathbf{X}|\mathbf{Z})P(\mathbf{Z}|\alpha, \beta)$$

By combining the IBP with different likelihood functions we can get different kinds of models:

- Models for graph structures (w/ Wood, Griffiths, 2006)
- Models for protein complexes (w/ Chu, Wild, 2006)
- Models for choice behaviour (Görür & Rasmussen, 2006)
- Models for users in collaborative filtering (w/ Meeds, Roweis, Neal, 2006)
- Sparse latent trait, pPCA and ICA models (w/ Knowles, 2007)
- Models for overlapping clusters (w/ Heller, 2007)

Posterior Inference in IBPs

$$P(\mathbf{Z}, \alpha|\mathbf{X}) \propto P(\mathbf{X}|\mathbf{Z})P(\mathbf{Z}|\alpha)P(\alpha)$$

Gibbs sampling: $P(z_{nk} = 1|\mathbf{Z}_{-(nk)}, \mathbf{X}, \alpha) \propto P(z_{nk} = 1|\mathbf{Z}_{-(nk)}, \alpha)P(\mathbf{X}|\mathbf{Z})$

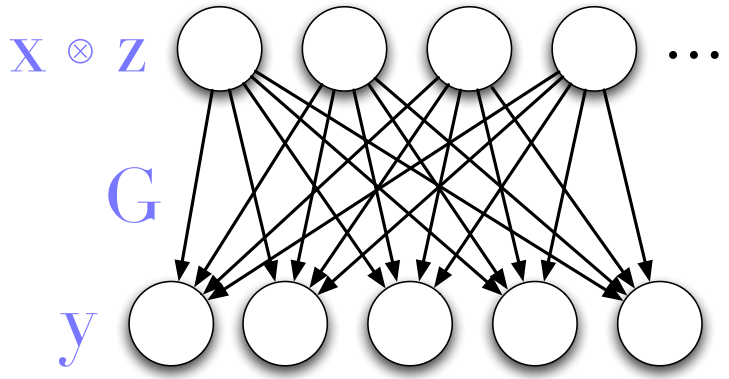
- If $m_{-n,k} > 0$, $P(z_{nk} = 1|\mathbf{z}_{-n,k}) = \frac{m_{-n,k}}{N}$
- For infinitely many k such that $m_{-n,k} = 0$: Metropolis steps with truncation* to sample from the number of new features for each object.
- If α has a Gamma prior then the posterior is also Gamma $\rightarrow$ Gibbs sample.

Conjugate sampler: assumes that $P(\mathbf{X}|\mathbf{Z})$ can be computed.

Non-conjugate sampler: $P(\mathbf{X}|\mathbf{Z}) = \int P(\mathbf{X}|\mathbf{Z}, \theta)P(\theta)d\theta$ cannot be computed, requires sampling latent θ as well (e.g. approximate samplers based on (Neal 2000) non-conjugate DPM samplers).

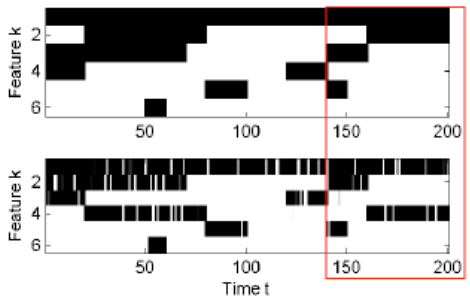
*__Slice sampler:__ works for non-conjugate case, is not approximate, and has an adaptive truncation level using an IBP stick-breaking construction (Teh, et al 2007).

Infinite Independent Components Analysis

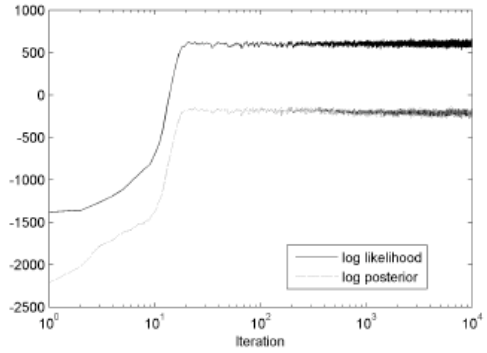


Model: $\mathbf{Y} = \mathbf{G}(\mathbf{Z} \otimes \mathbf{X}) + \mathbf{E}$

where $\mathbf{Y}$ is the data matrix, $\mathbf{G}$ is the mixing matrix $\mathbf{Z} \sim \text{IBP}(\alpha, \beta)$ is a mask matrix, $\mathbf{X}$ is heavy tailed sources and $\mathbf{E}$ is Gaussian noise.



(a) *Top:* True $\mathbf{Z}$. *Bottom:* Inferred $\mathbf{Z}$. Red box denotes test data.



(b) Plot of the log likelihood and posterior for the duration of the iICA$_2$ run.

Fig. 1. True and inferred $\mathbf{Z}$ and algorithm convergence.

(w/ David Knowles, 2007)

Modelling Dyadic Data

genes $\times$ patients

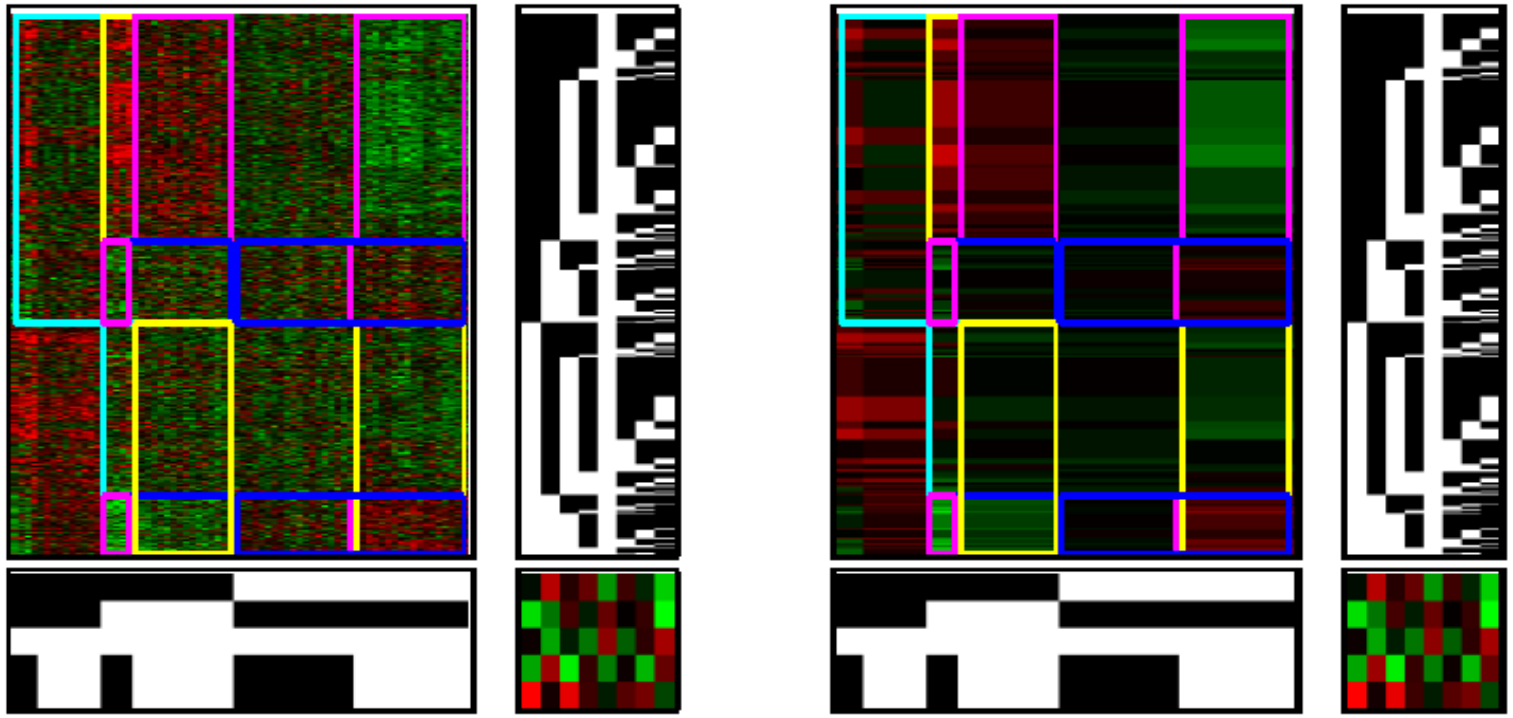


Figure 5: Gene expression results. **(A)** The top-left is $\mathbf{X}$ sorted according to contiguous features in the final $\mathbf{U}$ and $\mathbf{V}$ in the Markov chain. The bottom-left is $\mathbf{V}^\top$ and the top-right is $\mathbf{U}$. The bottom-right is $\mathbf{W}$. **(B)** The same as **(A)**, but the expected value of $\mathbf{X}$, $\hat{\mathbf{X}} = \mathbf{U}\mathbf{W}\mathbf{V}^\top$. We have hilighted regions that have both u_{ik} and v_{jl} on. For clarity, we have only shown the (at most) two largest contiguous regions for each feature pair.

users $\times$ movies

(w/ Meeds, Roweis, Neal, 2006)

Summary

- Bayesian machine learning treats learning as an probabilistic inference problem.

- Bayesian methods work well when the models are flexible enough to capture relevant properties of the data.

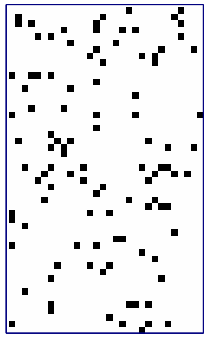
- This motivates non-parametric Bayesian methods, e.g.:

  - Gaussian processes for regression.
  - Dirichlet process mixtures for clustering.
  - Indian buffet processes for latent feature modelling.

`http://learning.eng.cam.ac.uk/zoubin`
`zoubin@eng.cam.ac.uk`

Appendix

From finite to infinite binary matrices



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

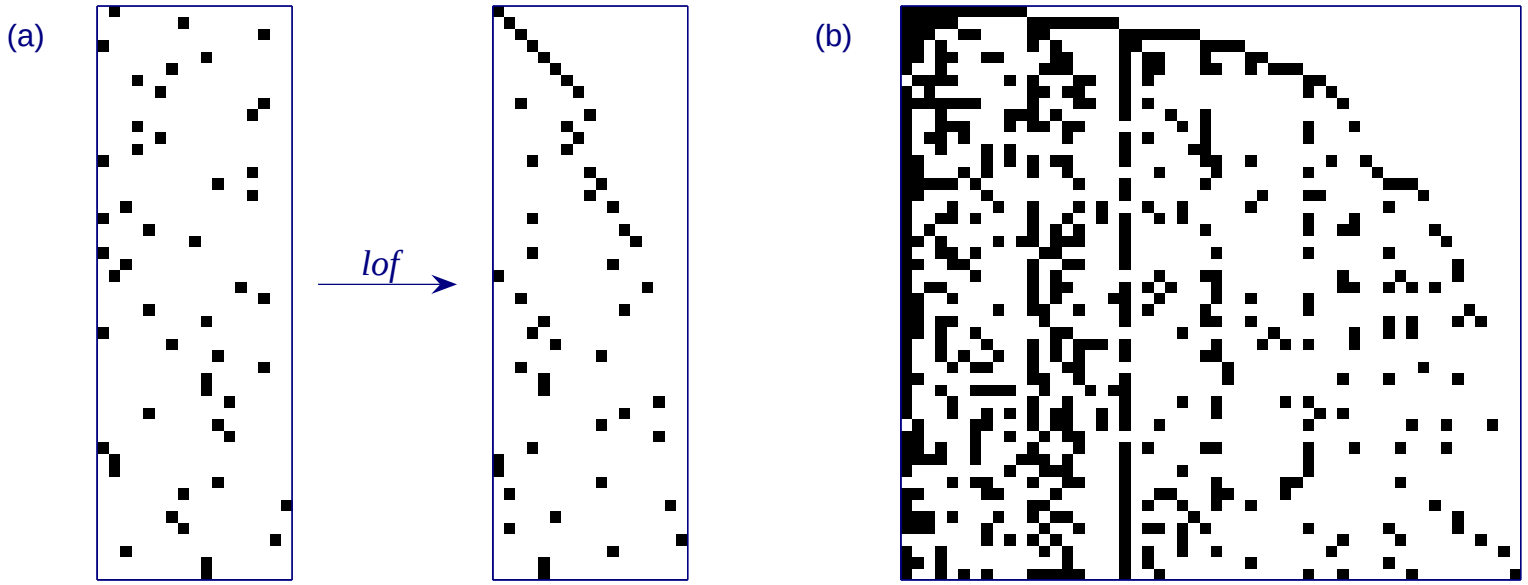
$$\lim_{K\to\infty} P(\mathbf{Z}|\alpha) \quad = \quad 0$$

However, if we consider equivalence classes of matrices in left-ordered form obtained by reordering the columns: $[\mathbf{Z}] = lof(\mathbf{Z})$ we get:

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

- 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).
- This distribution is **infinitely exchangeable**, i.e. it is not affected by the ordering on objects. This is important for its use as a prior in settings where the objects have no natural ordering.

Binary matrices in left-ordered form



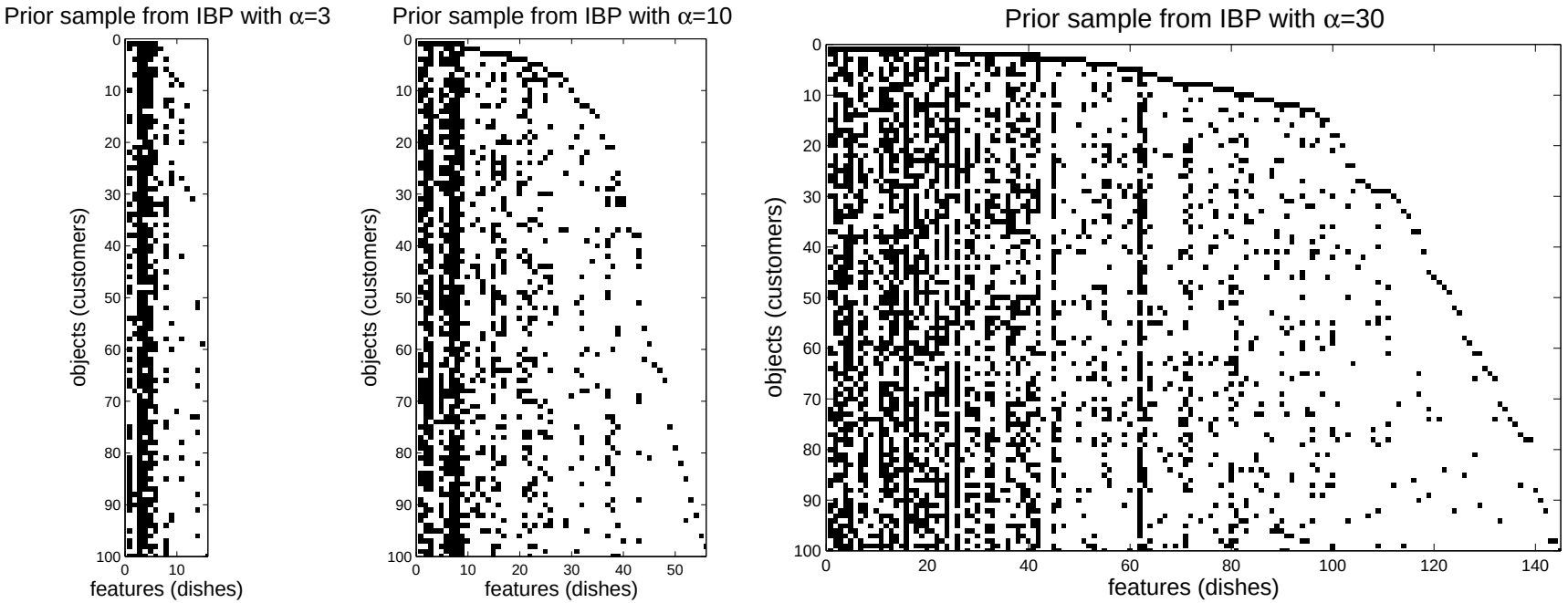
(a) The class matrix on the left is transformed into the class matrix on the right by the function $lof()$. The resulting left-ordered matrix was generated from a Chinese restaurant process (CRP) with $\alpha = 10$.

(b) A left-ordered feature matrix. This matrix was generated from the prior on infinite binary matrices with $\alpha = 10$.

Two generalizations

- a two-parameter generalization of the Indian Buffet Process
- from binary to non-binary latent features

I. A two-parameter generalization of the IBP?



Limitation:

- The hyperparameter α controls the number of features per object $K_n \stackrel{\text{def}}{=} \sum_k z_{nk} \sim \text{Poisson}(\alpha)$
- But α also controls the total number of features possessed by a set of N objects, i.e. the variability across rows of $\mathbf{Z}$.
- This seems limited—we really want independent control over the mean number of features and the variability across rows.

I. A two-parameter generalization of the IBP

$z_{nk} = 1$ means object n has feature k

One-parameter IBP

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

Two-parameter IBP

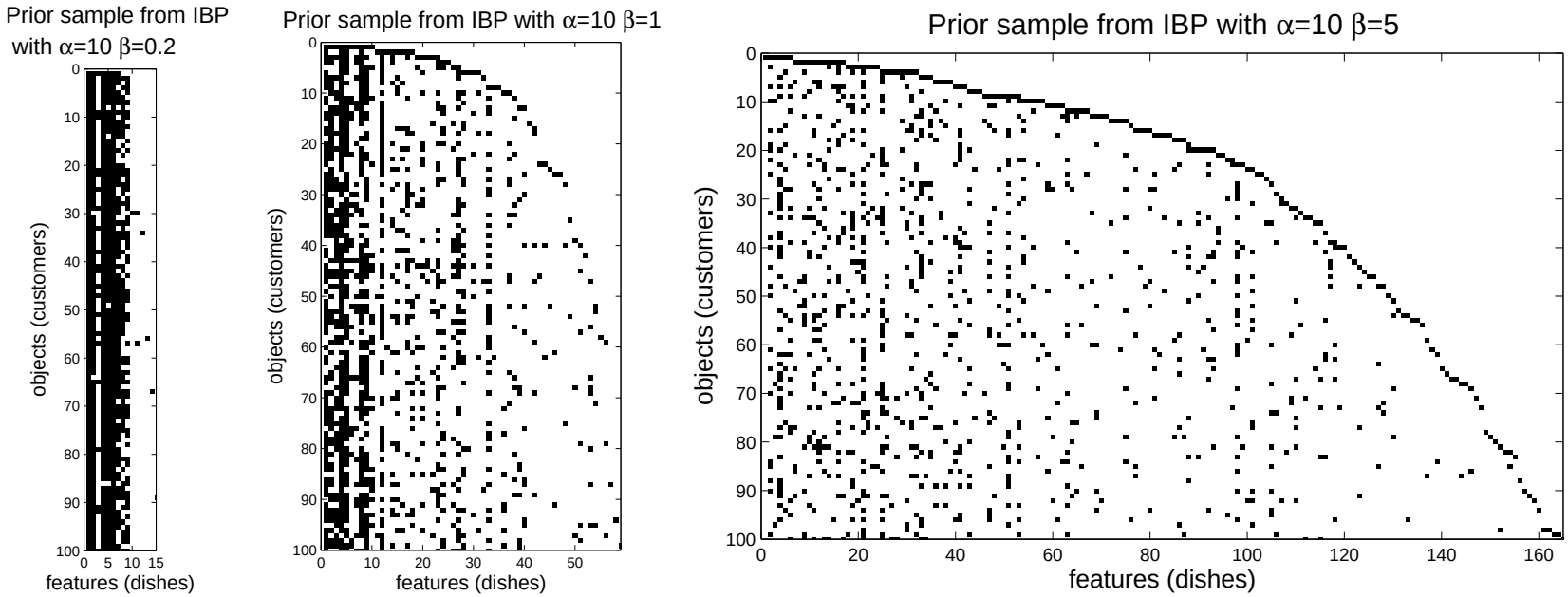
$$z_{nk} \sim \text{Bernoulli}(\theta_k)$$
$$\theta_k \sim \text{Beta}(\alpha\beta/K, \beta)$$

Properties of the two-parameter IBP

- Number of features per object is $\text{Poisson}(\alpha)$
- Setting $\beta = 1$ reduces to IBP.
- Parameter β is feature repulsion, $1/\beta$ is feature stickiness.
- Total expected number of features is $\bar{K}_+ = \alpha \sum_{n=1}^{N} \frac{\beta}{\beta + n - 1} \longrightarrow \alpha\beta \log N$
- $\lim_{\beta \to 0} \bar{K}_+ = \alpha$
- $\lim_{\beta \to \infty} \bar{K}_+ = N\alpha$

Joint work with Peter Sollich

I. A two-parameter generalization of the IBP

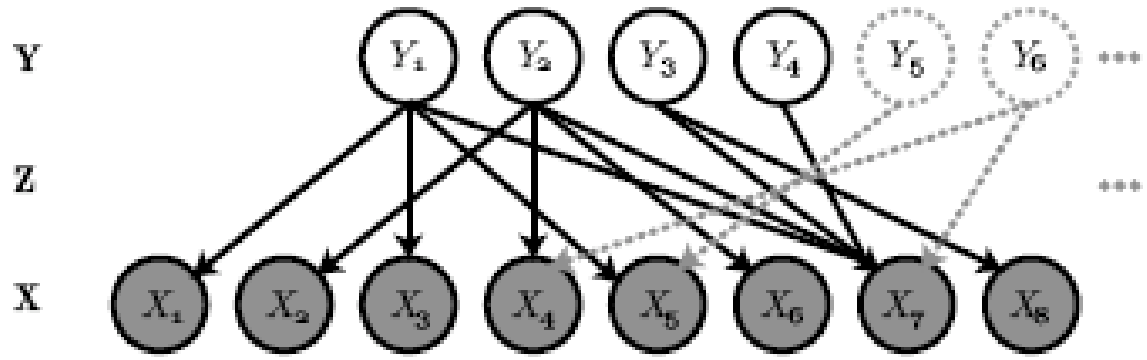


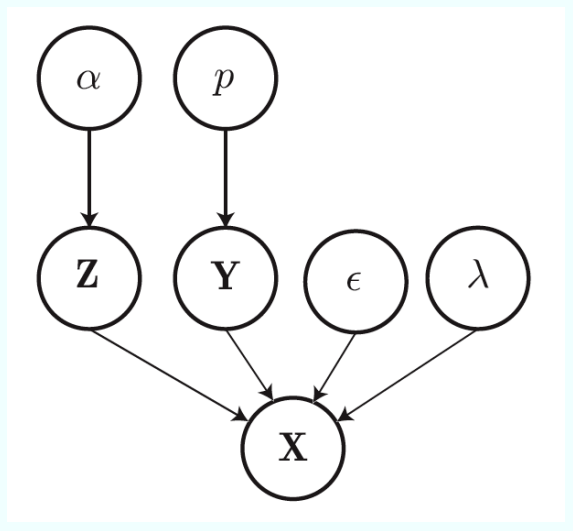
- First customer starts at the left of the buffet, and takes a serving from each dish, stopping after a Poisson(α) number of dishes as her plate becomes overburdened.

- The nth customer moves along the buffet, sampling dishes in proportion to their popularity, serving himself with probability $m_k/(\beta - 1 + n)$, and trying a Poisson($\alpha\beta/(\beta - 1 + n)$) number of new dishes.

An application of IBPs

"A Non-Parametric Bayesian Method for Inferring Hidden Causes"
(w/ Frank Wood, Tom Griffiths, 2006)





$\mathbf{Y}$ - binary latent factors (diseases)
$\mathbf{Z}$ - graph structure ($\sim$ IBP)
$\mathbf{X}$ - observed binary features (symptoms)

"Noisy-or" observations: $P(x_{nt} = 1|\mathbf{Z}, \mathbf{Y}, \lambda, \epsilon) = 1 - (1 - \lambda)^{\sum_k z_{nk} y_{kt}}(1 - \epsilon)$

An application of IBPs

Gibbs sampling traces

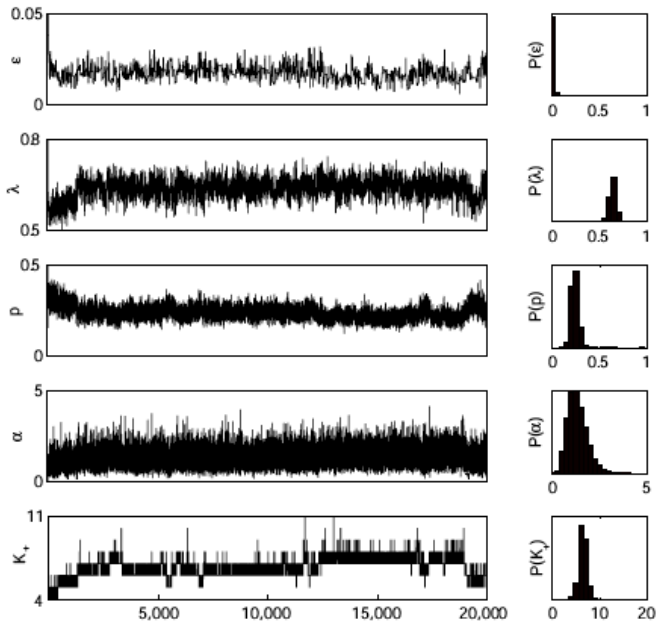


Figure 5: Trace plots and histograms for the Gibbs sampler applied to the signs exhibited by 50 stroke patients. The left column shows the current value of $\epsilon, \lambda, p, \alpha$, and K_+ as the sampler progressed, where K_+ is obtained by examining the current $\mathbf{Z}$ sample. The right column shows histograms of the same variables computed over the samples.

Comparison to RJMCMC

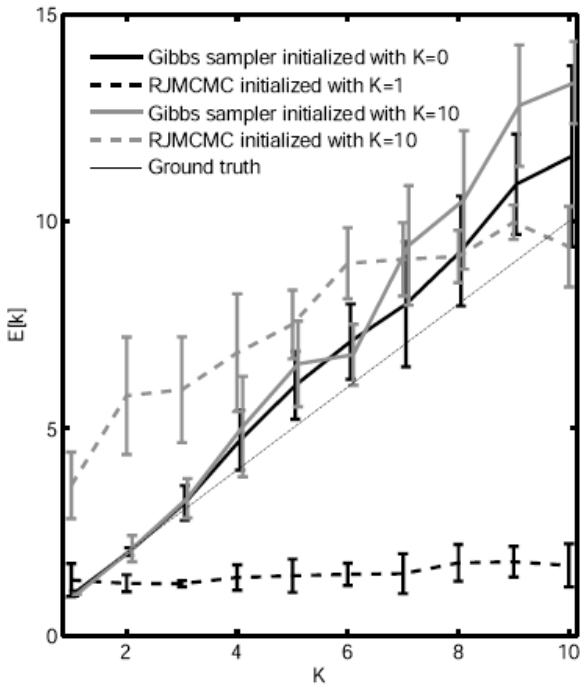


Figure 3: Learning the number of hidden causes using both RJMCMC and Gibbs sampling. Each line show the mean and standard deviation of the expected value of the dimensionality of the model (K for RJMCMC, and K_+ for Gibbs) taken over 500 iterations of sampling for each of 10 datasets.

An application of IBPs

Inferring stroke localization from patient symptoms:

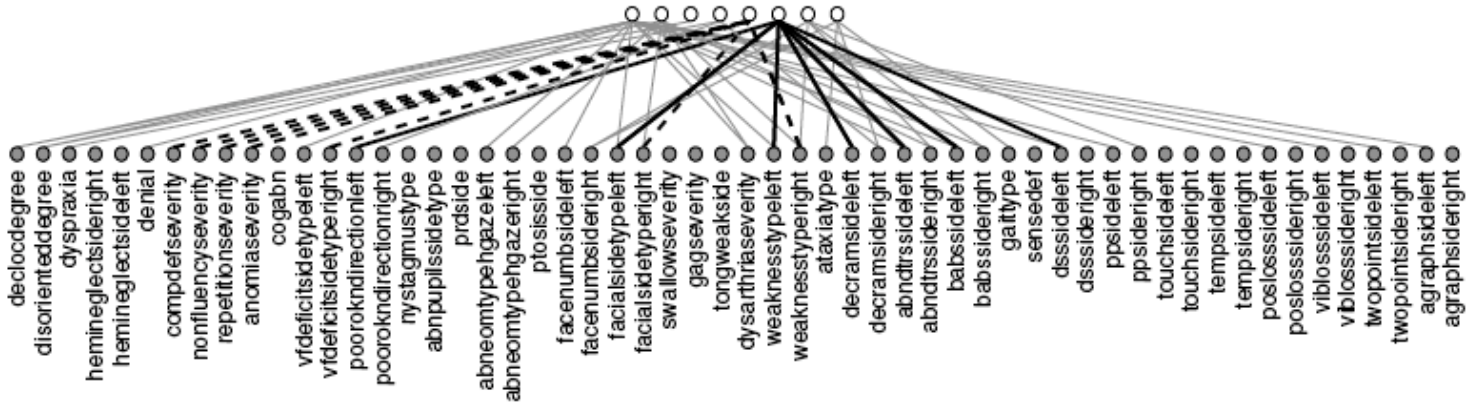


Figure 6: Causal structure with highest posterior probability. Two grouping of signs are highlighted. In solid black, we find a grouping of poor optokinetic nystagmus, lack of facial control, weakness, decreased rapid alternating movements, abnormal deep tendon reflexes, Babinski sign, and double simultaneous stimulation neglect, all on the left side, consistent with a right frontal/parietal infarct. In dashed black, we find a grouping of comprehension deficit, non-fluency, repetition, anomia, visual field deficit, facial weakness, and general weakness, with the latter three on the right side, generally consistent with a left temporal infarct.

(50 stroke patients, 56 symptoms/signs)

An application of IBPs - II

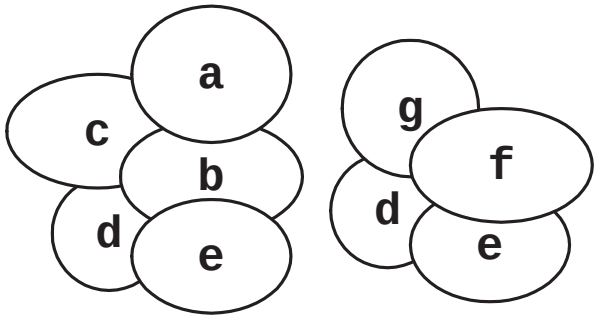
"Identifying protein complexes in high-throughput protein interaction screens using an infinite latent feature model" (with Wei Chu, David L Wild, and Roland Krause)

- Data: protein-protein interaction screens.
- Proteins belong to complexes.
- A protein can belong to multiple complexes.
- We don't know how many complexes there are.
- We want to infer the complexes from the interaction screen data.

An application of IBPs, continued

We use data from affinity purification/mass spectrometry (APMS) experiments:

Individual proteins are tagged and used as baits to form physiological complexes with other proteins in the cell



(i) Native Protein Complexes

Protein	a	b	c	d	e	f	g
Bait a	1	1	1	1	*0*	0	0
Bait d	*0*	1	1	1	1	1	1
Bait f	0	0	0	1	1	1	1

(ii) Purification Matrix, B

	a	b	c	d	e	f	g
a	1	1	1	1	0	0	0
b	1	2	2	2	1	1	1
c	1	2	2	2	1	1	1
d	1	2	2	3	2	2	2
e	0	1	1	2	2	2	2
f	0	1	1	2	2	2	2
g	0	1	1	2	2	2	2

(iii) Adjacency Matrix, A

Complex	a	b	c	d	e	f	g
1	1	1	1	1	1	0	0
2	0	0	0	1	1	1	1
3	0	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

(iv) Complex Membership, Z

An application of IBPs, continued

Adjacency matrix:s $A = B^\top B$

Von Neumann Diffusion Kernel: $K := \sum_{\ell=1}^{\infty} \gamma^{\ell-1} A^\ell = A\,(1-\gamma A)^{-1}$
where γ is the diffusion factor.
The normalized kernel is an appropriate measure of similarity:

$$D_{ij} = \frac{K_{ij}}{\sqrt{K_{ii} K_{jj}}}$$

The prior:

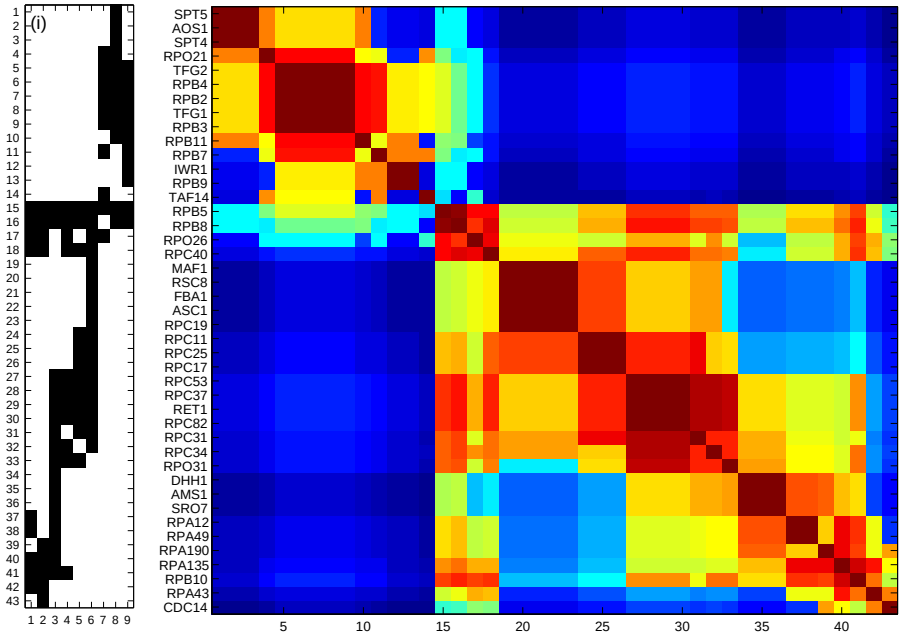
$$Z \sim \mathsf{IBP}(\alpha)$$

where $z_{ci} = 1$ means protein n belongs to complex c.
The likelihood:

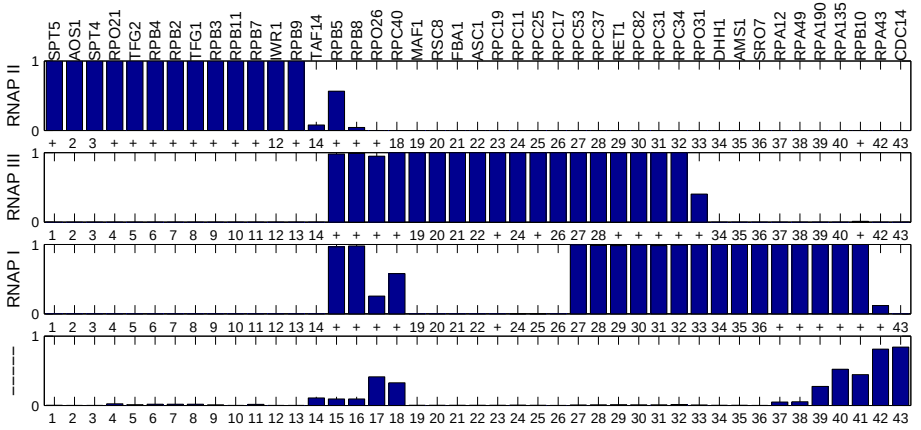
$$P(D|Z) = \prod_{\{ij:\mathbf{z}_i^\top \mathbf{z}_j > 0\}} (D_{ij})^{\mathbf{z}_i^\top \mathbf{z}_j} \prod_{\{ij:\mathbf{z}_i^\top \mathbf{z}_j = 0\}} (1 - D_{ij})$$

Inference via Gibbs sampling.

Results



The RNA Polymerase complexes. (i) presents the purification results using 9 bait proteins. (ii) presents the corresponding normalized von Neumann kernel matrix, where the gray scale indicates the probability of pairwise membership defined by D. The proteins are sorted according to the inferred complex membership

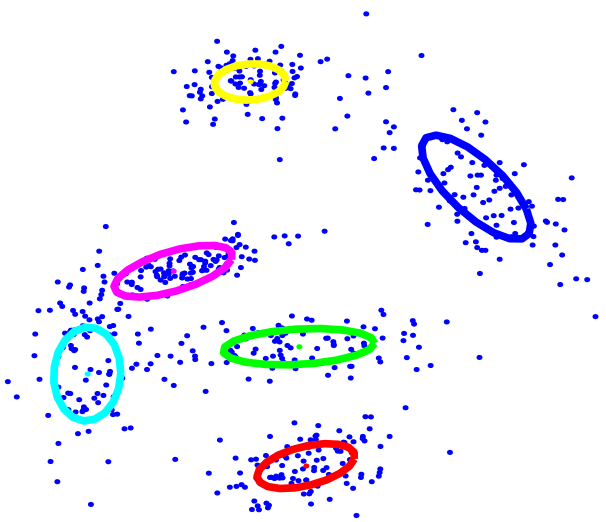


The four largest complexes identified by our algorithm. The bars indicate the probability of membership of the proteins. The top 3 complexes correspond to RNAP II, RNAP III and RNAP I respectively. The cross indices indicate the members of the three RNA polymerase complexes according to the MIPS protein complex database.

Dirichlet Process Mixtures (Infinite Mixtures)

Start with a finite mixture of K components to model data $\{\mathbf{x}^{(1)}, \ldots \mathbf{x}^{(N)}\}$:

$$p(\mathbf{x}^{(n)}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \, p_k(\mathbf{x}^{(n)}|\boldsymbol{\theta}_k)$$



Represent using latent indicator variables:

$$p(\mathbf{x}^{(n)}|\boldsymbol{\theta}) = \sum_{k=1}^{K} P(s^{(n)} = k|\boldsymbol{\pi}) \, p_k(\mathbf{x}^{(n)}|\boldsymbol{\theta}_k, s^{(n)} = k)$$

where $s^{(n)} = k$ means $\mathbf{x}^{(n)}$ came from component k.
Distribution of indicators $\mathbf{s} = (s^{(1)}, \ldots, s^{(N)})$ given $\boldsymbol{\pi}$ is **multinomial**

$$P(s^{(1)}, \ldots, s^{(N)}|\boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{N_k}, \qquad N_k \stackrel{\text{def}}{=} \sum_{n=1}^{N} \delta(s^{(n)}, k) \; .$$

Give the mixing proportions $\boldsymbol{\pi}$ a symmetric **Dirichlet prior**

$$p(\boldsymbol{\pi}|\alpha) = \frac{\Gamma(\alpha)}{\Gamma(\alpha/K)^K} \prod_{k=1}^{K} \pi_k^{\alpha/K-1}$$

Dirichlet Process Mixtures (Infinite Mixtures) - II

Distribution of indicators $\mathbf{s} = (s^{(1)}, \ldots, s^{(N)})$ given $\boldsymbol{\pi}$ is **multinomial**

$$P(s^{(1)}, \ldots, s^{(N)}|\boldsymbol{\pi}) = \prod_{k=1}^{K} \pi_k^{N_k}, \qquad N_k \stackrel{\text{def}}{=} \sum_{n=1}^{N} \delta(s^{(n)}, k) \ .$$

Give the mixing proportions $\boldsymbol{\pi}$ a symmetric **Dirichlet prior**

$$p(\boldsymbol{\pi}|\alpha) = \frac{\Gamma(\alpha)}{\Gamma(\alpha/K)^K} \prod_{k=1}^{K} \pi_k^{\alpha/K-1}$$

Integrate out the mixing proportions, $\boldsymbol{\pi}$, to obtain:

$$P(s^{(1)}, \ldots, s^{(N)}|\alpha) = \int d\boldsymbol{\pi} \ P(\mathbf{s}|\boldsymbol{\pi})P(\boldsymbol{\pi}|\alpha) = \frac{\Gamma(\alpha)}{\Gamma(N+\alpha)} \prod_{k=1}^{K} \frac{\Gamma(N_k + \alpha/K)}{\Gamma(\alpha/K)}$$

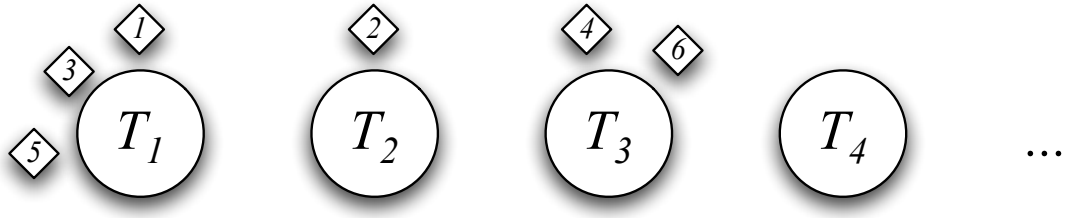
Take limit as $K \to \infty$, but instead of looking at distribution over indicator variables, $s^{(n)}$, consider the corresponding distribution over partitions of n data points.[3]

E.g.: (1 2 3) (4) (5) or (1 3) (2 5) (4) or (1 2 3 4 5) or (1) (2 5) (3) (4)...

[3] The number of such partitions is the nth Bell number

Chinese Restaurant Process

The CRP generates samples from the distribution over partitions induced by a DPM.



Generating from a CRP:

customer 1 enters the restaurant and sits at table 1.
$K = 1$, $N = 1$, $N_1 = 1$
for $N = 2, \ldots,$

customer N sits at table $\begin{cases} k & \text{with prob } \frac{N_k}{N-1+\alpha} & \text{for } k = 1 \ldots K \\ K+1 & \text{with prob } \frac{\alpha}{N-1+\alpha} & \text{(new table)} \end{cases}$

if new table was chosen **then** $K \leftarrow K + 1$ **endif**
endfor

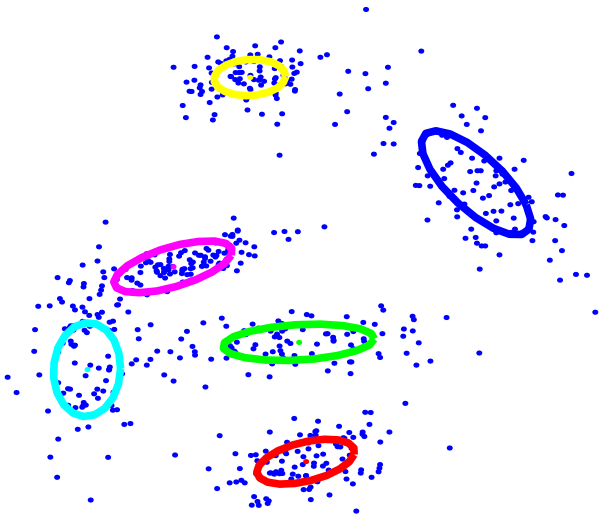
"Rich get richer" property.

(Aldous 1985; Pitman 2002)

Clustering with Gaussian Mixtures (Density Estimation)

Data: $\mathcal{D} = \{\mathbf{x}^{(n)}\}$ for $n = 1, \ldots, N$

$$\mathbf{x}^{(n)} \in \Re^D$$



Parameters: $\boldsymbol{\theta} = \left((\boldsymbol{\mu}^{(1)}, \Sigma^{(1)}) \ldots, (\boldsymbol{\mu}^{(m)}, \Sigma^{(m)}), \boldsymbol{\pi}\right)$

Model:

$$\mathbf{x}^{(n)} \sim \sum_{n=1}^{m} \pi_n \, p_i(\mathbf{x}^{(n)})$$

where

$$p_i(\mathbf{x}^{(n)}) = \mathcal{N}(\boldsymbol{\mu}^{(i)}, \Sigma^{(i)})$$

Goal: To infer $\boldsymbol{\theta}$ from the data and predict the density $p(\mathbf{x}|\mathcal{D}, m)$

Dirichlet Process Mixtures (Infinite Mixtures) - III

Starting from $$P(\mathbf{s}|\alpha) = \frac{\Gamma(\alpha)}{\Gamma(n+\alpha)} \prod_{j=1}^{K} \frac{\Gamma(n_j + \alpha/K)}{\Gamma(\alpha/K)}$$

Conditional Probabilities: Finite K

$$P(s^{(i)} = j|\mathbf{s}_{-n}, \alpha) = \frac{n_{-n,j} + \alpha/K}{n - 1 + \alpha}$$

where $\mathbf{s}_{-n}$ denotes all indices except n, and $n_{-n,j} \stackrel{\text{def}}{=} \sum_{\ell \neq i} \delta(s^{(\ell)}, j)$

DP: more populous classes are more more likely to be joined

Conditional Probabilities: Infinite K

Taking the limit as $K \to \infty$ yields the conditionals

$$P(s^{(i)} = j|\mathbf{s}_{-n}, \alpha) = \begin{cases} \frac{n_{-n,j}}{n-1+\alpha} & j \text{ represented} \\ \\ \frac{\alpha}{n-1+\alpha} & \text{all } j \text{ not represented} \end{cases}$$

Left over mass, α, $\Rightarrow$ **countably infinite** number of indicator settings.