Bayesian Nonparametrics

some research vignettes

Zoubin Ghahramani

Department of Engineering
University of Cambridge, UK

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

MLSS Tübingen Lectures
2013

Probabilistic Modelling

- A model describes data that one could observe from a system
- If we use the mathematics of probability theory to express all forms of uncertainty and noise associated with our model...
- ...then *inverse probability* (i.e. Bayes rule) allows us to infer unknown quantities, adapt our models, make predictions and learn from data.

Why Bayesian Nonparametrics...?

- **Why Bayesian?**

  Simplicity (of the framework)

- **Why nonparametrics?**

  Complexity (of real world phenomena)

Parametric vs Nonparametric Models

- *Parametric models* assume some finite set of parameters θ. Given the parameters, future predictions, x, are independent of the observed data, $\mathcal{D}$:

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

  therefore θ capture everything there is to know about the data.

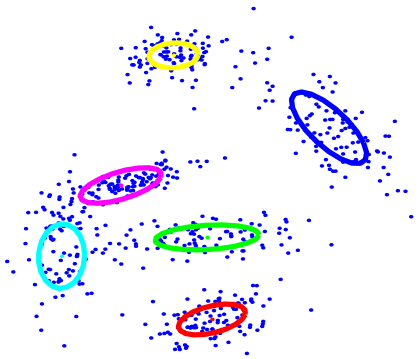
- So the complexity of the model is bounded even if the amount of data is unbounded. This makes them not very flexible.

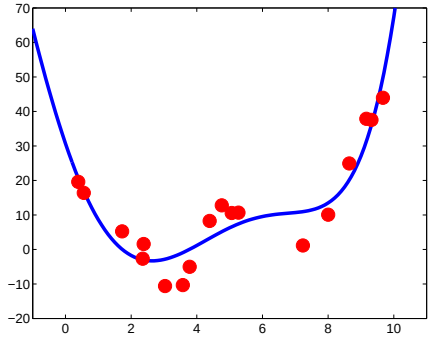
- *Non-parametric models* assume that the data distribution cannot be defined in terms of such a finite set of parameters. But they can often be defined by assuming an *infinite dimensional* θ. Usually we think of θ as a *function*.

- The amount of information that θ can capture about the data $\mathcal{D}$ can grow as the amount of data grows. This makes them more flexible.

Why nonparametrics?

- flexibility
- better predictive performance
- more realistic





Almost all successful methods in machine learning are essentially nonparametric[1]:

- kernel methods / SVM / GP
- deep networks / large neural networks
- k-nearest neighbors, ...

[1] or highly scalable!

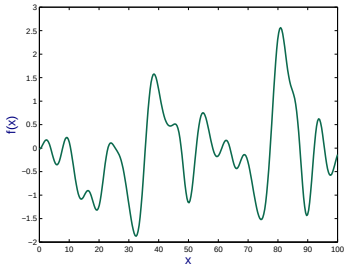
Examples of non-parametric models

Bayesian nonparametrics has many uses.

Parametric	Non-parametric	Process	Application
polynomial regression	Gaussian processes	GP	function approx.
logistic regression	Gaussian process classifiers	GP	classification
mixture models, k-means	Dirichlet process mixtures	DP / CRP	clustering
hidden Markov models	infinite HMMs	HDP	time series
factor analysis/pPCA/PMF	infinite latent factor models	BP / IBP	feature discovery
...			

Gaussian and Dirichlet Processes

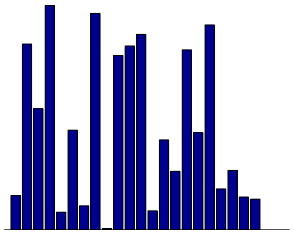
- Gaussian processes define a distribution on functions



$$f \sim \mathrm{GP}(\cdot|\mu, c)$$

where μ is the mean function and c is the covariance function.
We can think of GPs as "infinite-dimensional" Gaussians

- Dirichlet processes define a distribution on distributions

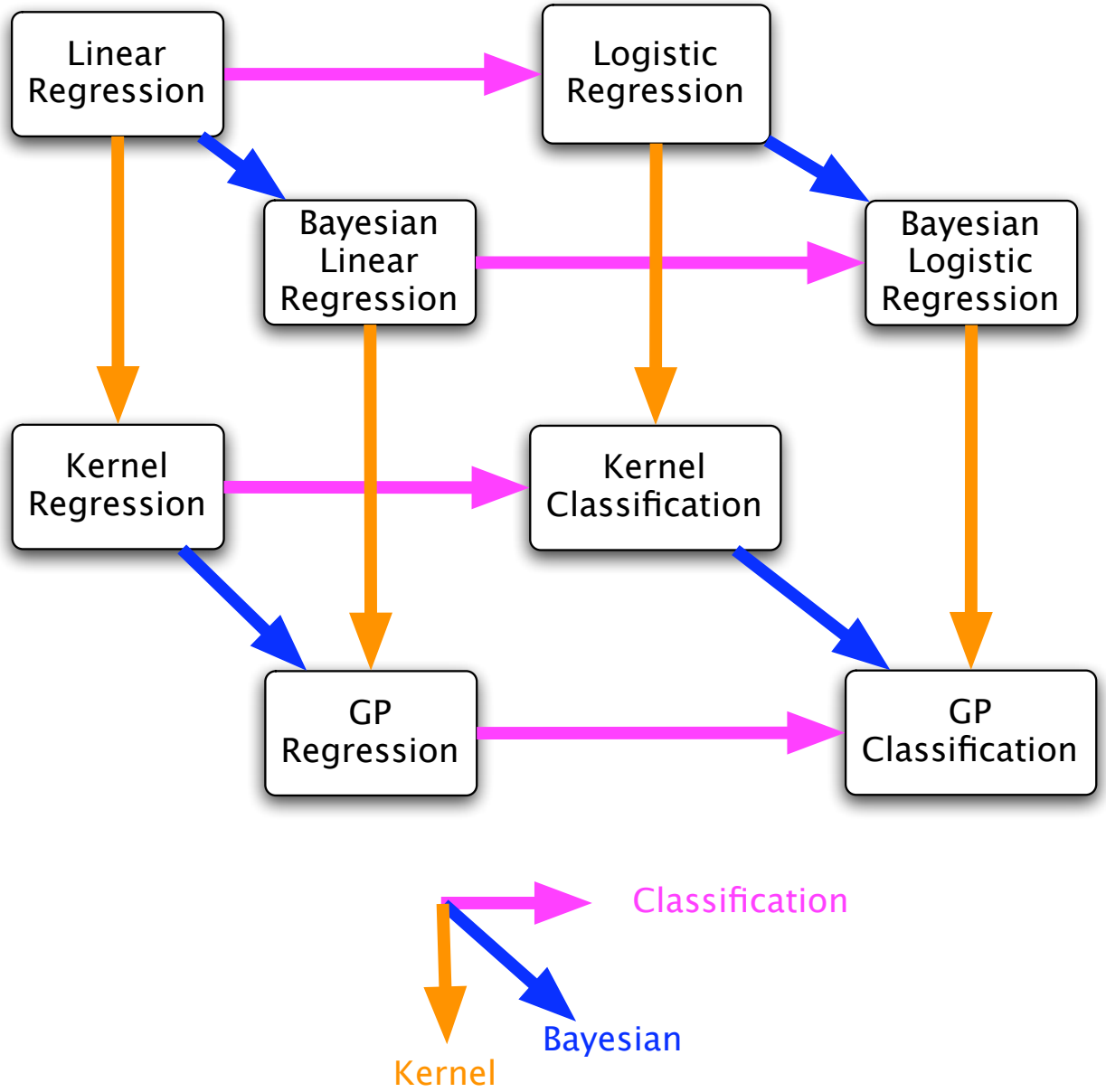


$$G \sim \mathrm{DP}(\cdot|G_0, \alpha)$$

where $\alpha > 0$ is a concentration parameter, and G_0 is the base measure.
We can think of DPs as "infinite-dimensional" Dirichlet distributions.

Note that both f and G are infinite dimensional objects.

A picture


Linear Regression
Logistic Regression
Bayesian Linear Regression
Bayesian Logistic Regression
Kernel Regression
Kernel Classification
GP Regression
GP Classification
Classification
Bayesian
Kernel

Dirichlet Distribution

The Dirichlet distribution is a distribution on the K-dim probability simplex.

Let $\mathbf{p}$ be a K-dimensional vector s.t. $\forall j : p_j \geq 0$ and $\sum_{j=1}^{K} p_j = 1$

$$P(\mathbf{p}|\boldsymbol{\alpha}) = \text{Dir}(\alpha_1, \ldots, \alpha_K) \stackrel{\text{def}}{=} \frac{\Gamma(\sum_j \alpha_j)}{\prod_j \Gamma(\alpha_j)} \prod_{j=1}^{K} p_j^{\alpha_j - 1}$$

where the first term is a normalization constant[2] and $E(p_j) = \alpha_j / (\sum_k \alpha_k)$

The Dirichlet is conjugate to the multinomial distribution. Let

$$c|\mathbf{p} \sim \text{Multinomial}(\cdot|\mathbf{p})$$

That is, $P(c = j|\mathbf{p}) = p_j$. Then the posterior is also Dirichlet:

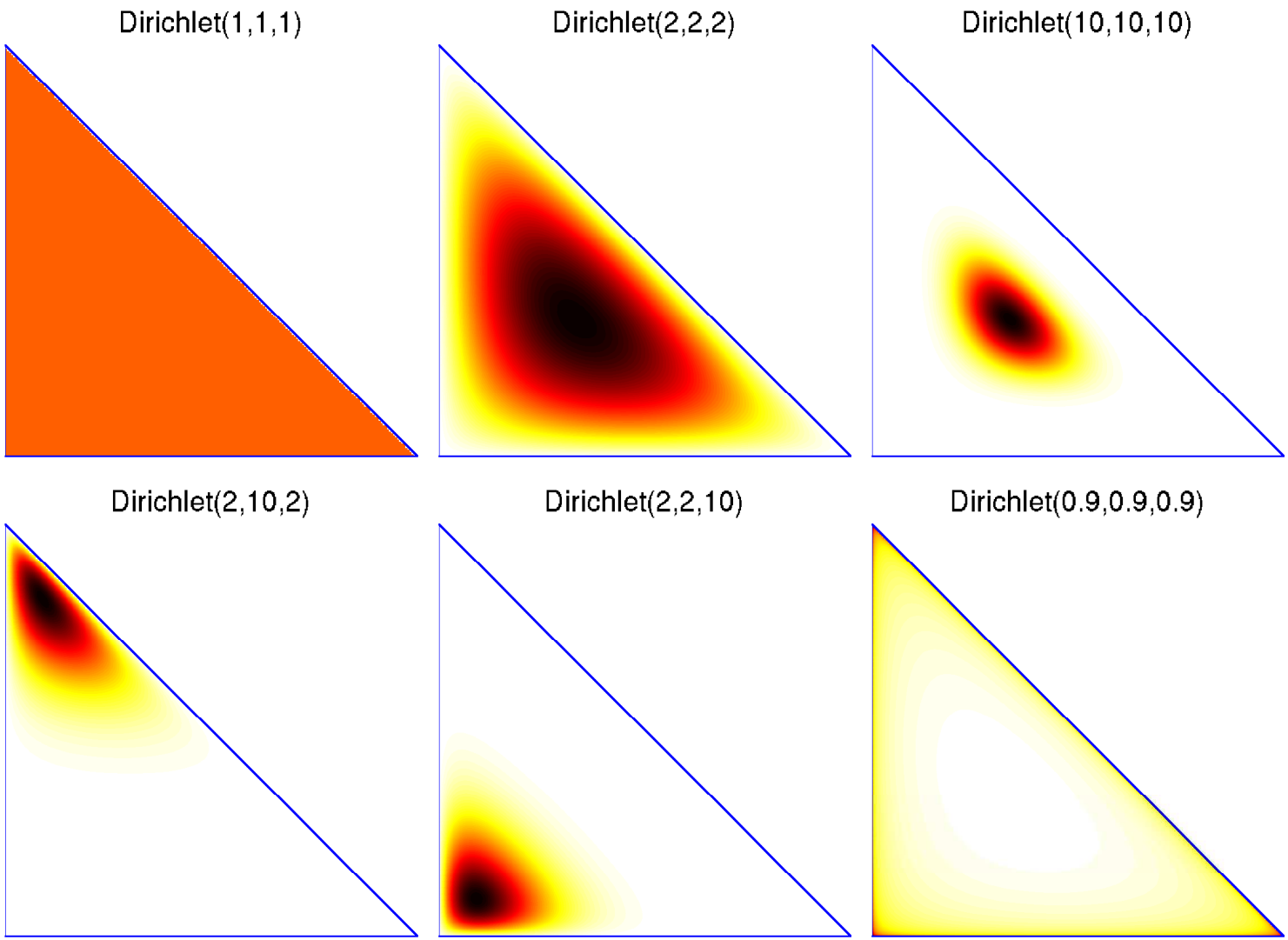
$$P(\mathbf{p}|c = j, \boldsymbol{\alpha}) = \frac{P(c = j|\mathbf{p}) P(\mathbf{p}|\boldsymbol{\alpha})}{P(c = j|\boldsymbol{\alpha})} = \text{Dir}(\boldsymbol{\alpha}')$$

where $\alpha'_j = \alpha_j + 1$, and $\forall \ell \neq j : \alpha'_\ell = \alpha_\ell$

[2] $\Gamma(x) = (x-1)\Gamma(x-1) = \int_0^\infty t^{x-1} e^{-t} dt$. For integer n, $\Gamma(n) = (n-1)!$

Dirichlet Distributions

Examples of Dirichlet distributions over $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3)$ which can be plotted in 2D since $\theta_3 = 1 - \theta_1 - \theta_2$:



Dirichlet Process

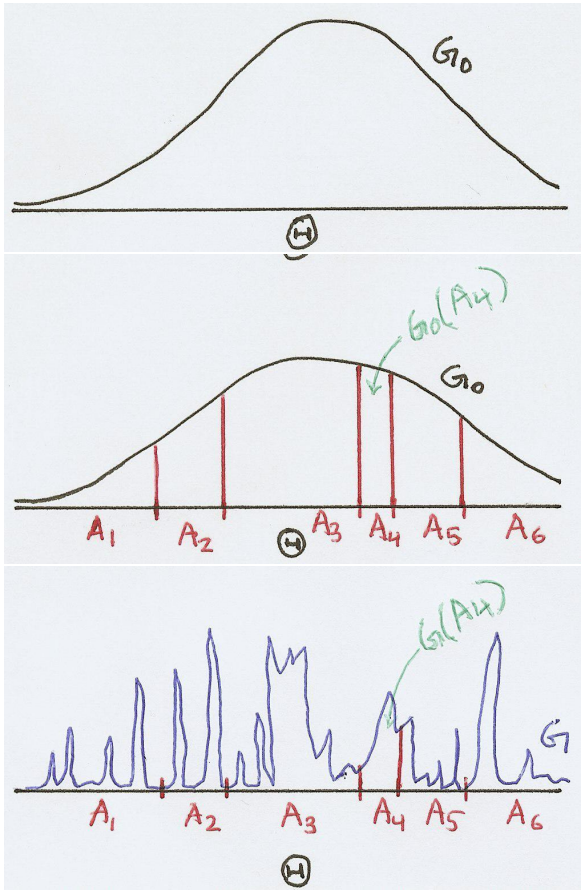
Let Θ be a measurable space, G_0 be a probability measure on Θ, and α a positive real number.

For all $(A_1, \ldots A_K)$ finite partitions of Θ,

$$G \sim \mathsf{DP}(\cdot | G_0, \alpha)$$

means that

$$(G(A_1), \ldots, G(A_K)) \sim \mathsf{Dir}(\alpha G_0(A_1), \ldots, \alpha G_0(A_K))$$



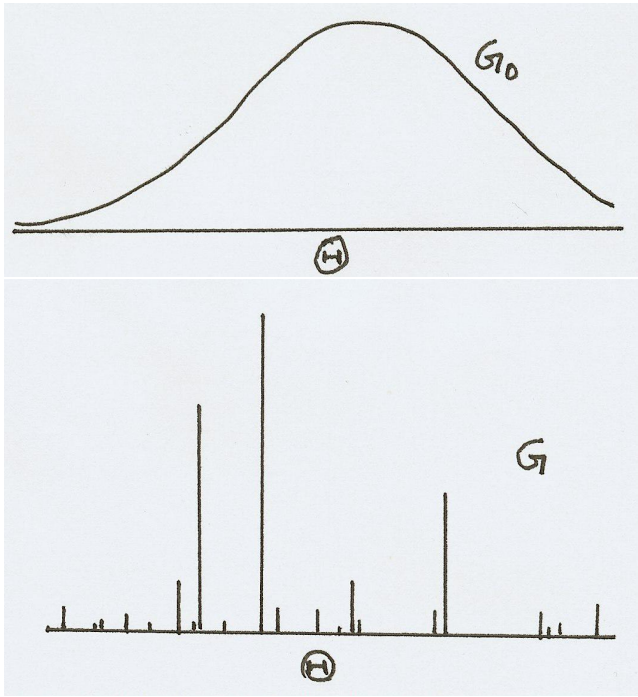
(Ferguson, 1973)

Dirichlet Process

$G \sim \text{DP}(\cdot|G_0, \alpha)$ OK, but what does it look like?

Samples from a DP are discrete with probability one:

$$G(\theta) = \sum_{k=1}^{\infty} \pi_k \delta_{\theta_k}(\theta)$$



where $\delta_{\theta_k}(\cdot)$ is a Dirac delta at θ_k, and $\theta_k \sim G_0(\cdot)$.

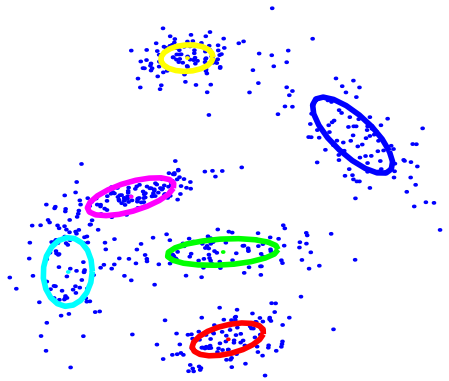
Note: $E(G) = G_0$

As $\alpha \to \infty$, G looks more "like" G_0.

Clustering

Infinite mixture models

(e.g. Dirichlet Process Mixtures)



Why?

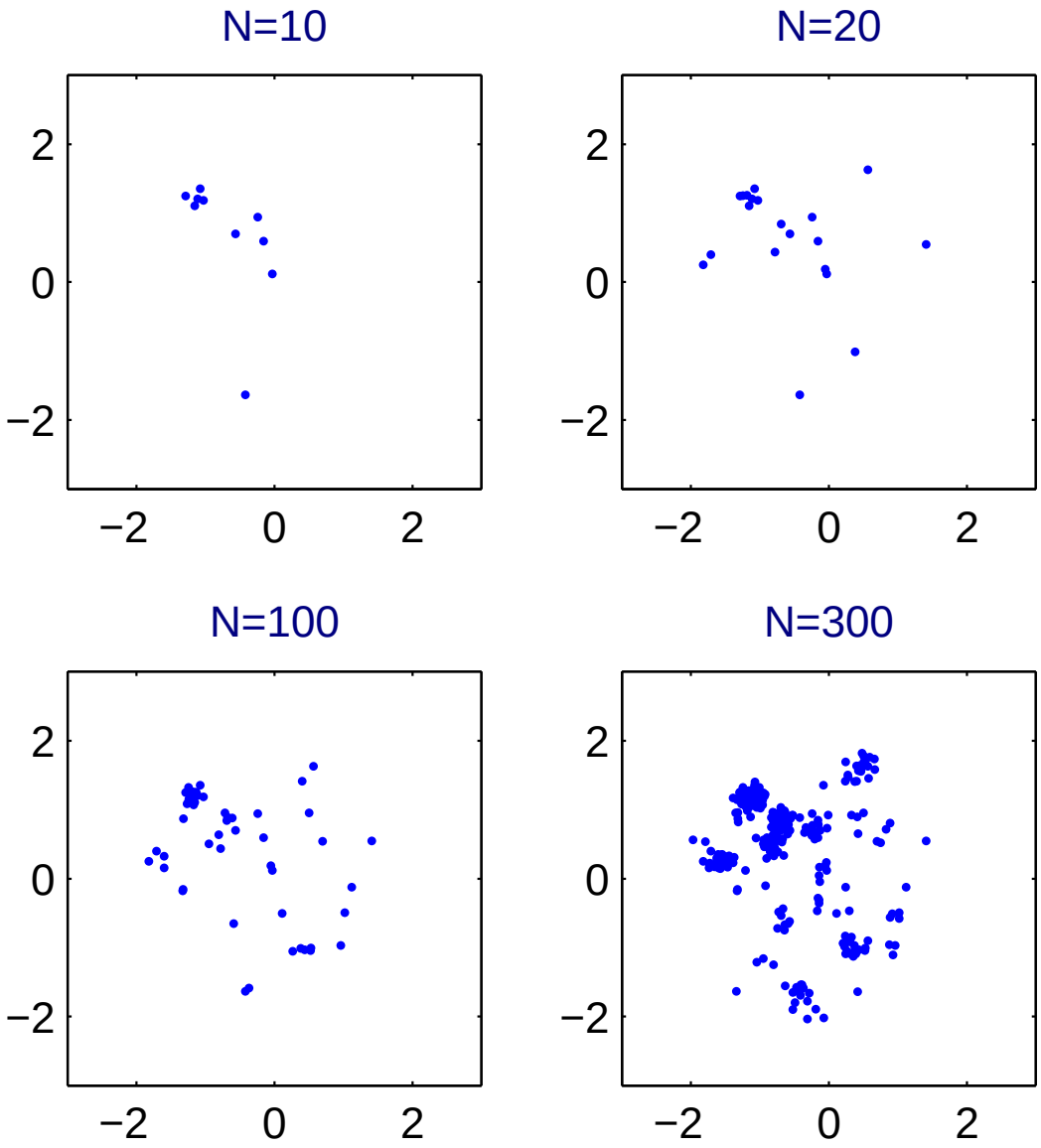
- You might not believe a priori that your data comes from a finite number of mixture components (e.g. strangely shaped clusters; heavy tails; structure at many resolutions)

- 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 as number of components $K \to \infty$
- If you use symmetric Dirichlet priors on the $\{\pi_k\}$ you get a Dirichlet Process Mixture; the distribution over partitions is given by a Chinese Restaurant Process.
- 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)

Relationship between DPs and CRPs

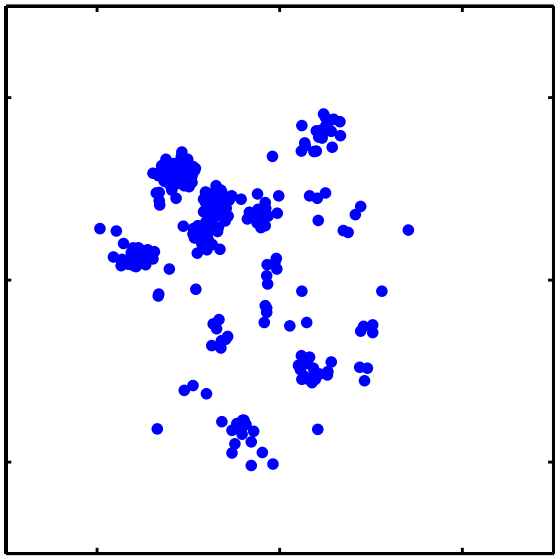
- DP is a distribution on distributions
- DP results in discrete distributions, so if you draw n points you are likely to get repeated values
- A DP induces a partitioning of the n points
  e.g. $(1\,3\,4)\,(2\,5) \Leftrightarrow \theta_1 = \theta_3 = \theta_4 \neq \theta_2 = \theta_5$
- Chinese Restaurant Process (CRP) defines the corresponding distribution on partitions
- Although the CRP is a sequential process, the distribution on $\theta_1, \ldots, \theta_n$ is exchangeable (i.e. invariant to permuting the indices of the θs): e.g.

$$P(\theta_1, \theta_2, \theta_3, \theta_4) = P(\theta_2, \theta_4, \theta_3, \theta_1)$$

Dirichlet Processes: Big Picture

There are many ways to derive the Dirichlet Process:

- Dirichlet distribution
- Urn model
- Chinese restaurant process
- Stick breaking
- Gamma process



DP: distribution on distributions

Dirichlet process mixture (DPM): a mixture model with infinitely many components where parameters of each component are drawn from a DP. Useful for clustering; assignments of points to clusters follows a CRP.

Bayesian nonparametrics for structured data

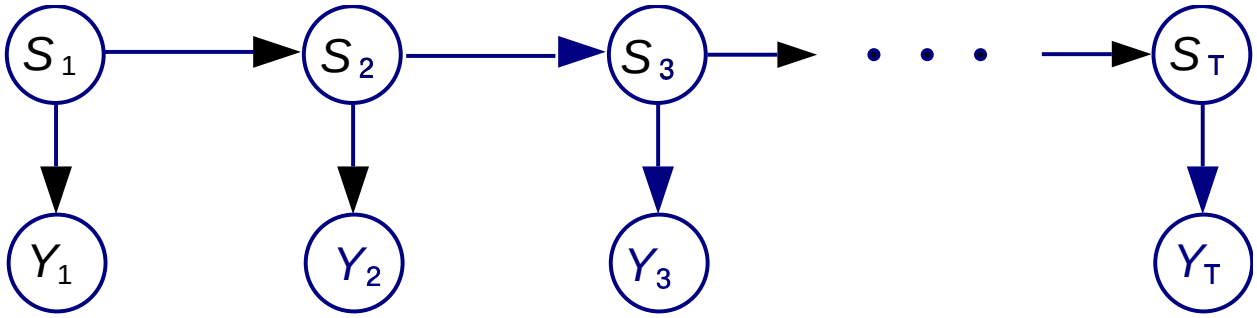
Bayesian nonparametrics applied to models of other structured objects:

- Clusters [and hierarchies]
- Time Series
- Sparse Matrices and Feature Allocation Models
- Networks

Times Series

Hidden Markov Models

Hidden Markov models (HMMs) are widely used sequence models for speech recognition, bioinformatics, biophysics, text modelling, video monitoring, etc.

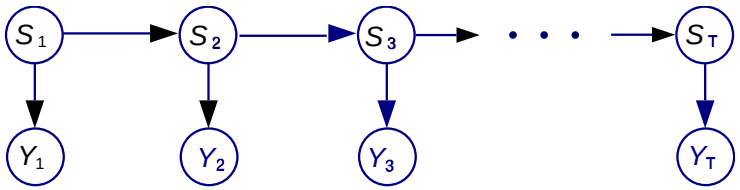


In an HMM, the sequence of observations $\mathbf{y}_1, \ldots, \mathbf{y}_T$ is modelled by assuming that it was generated by a sequence of discrete hidden states $s_1, \ldots, s_T$ with Markovian dynamics.

If the HMM has K states ($s_t \in \{1, \ldots K\}$) the transition matrix has $K \times K$ elements.

HMMs can be thought of as *time-dependent mixture models*.

Infinite hidden Markov models (iHMMs)

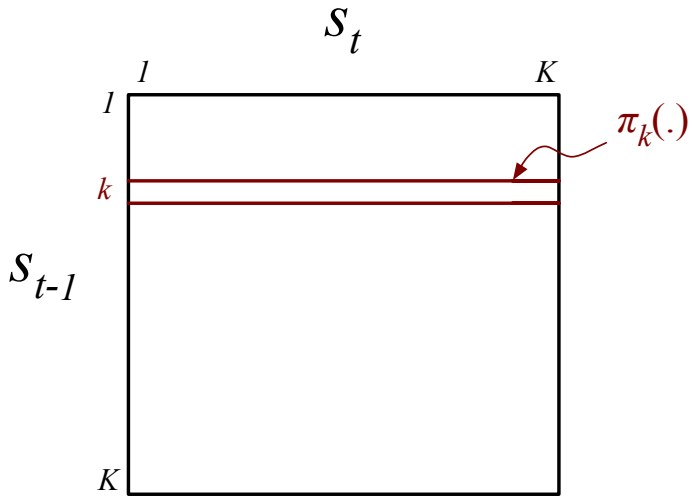


HMMs can be thought of as *time-dependent mixture models*.

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

We want to let $K \to \infty$

Infinite HMMs can be derived from the HDP.



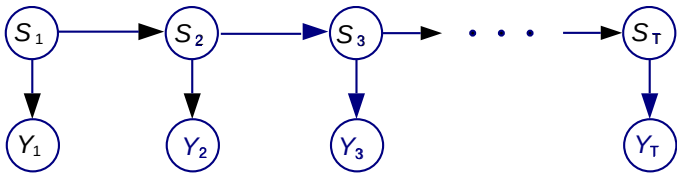
$$
\begin{array}{rcll}
\beta|\gamma & \sim & \text{Stick}(\cdot|\gamma) & \text{(base distribution over states)} \\
\pi_k|\alpha,\beta & \sim & \text{DP}(\cdot|\alpha,\beta) & \text{(transition parameters for state } k = 1,\dots\text{)} \\
\theta_k|H & \sim & H(\cdot) & \text{(emission parameters for state } k = 1,\dots\text{)} \\
s_t|s_{t-1},(\pi_k)_{k=1}^{\infty} & \sim & \pi_{s_{t-1}}(\cdot) & \text{(transition)} \\
y_t|s_t,(\theta_k)_{k=1}^{\infty} & \sim & p(\cdot|\theta_{s_t}) & \text{(emission)}
\end{array}
$$

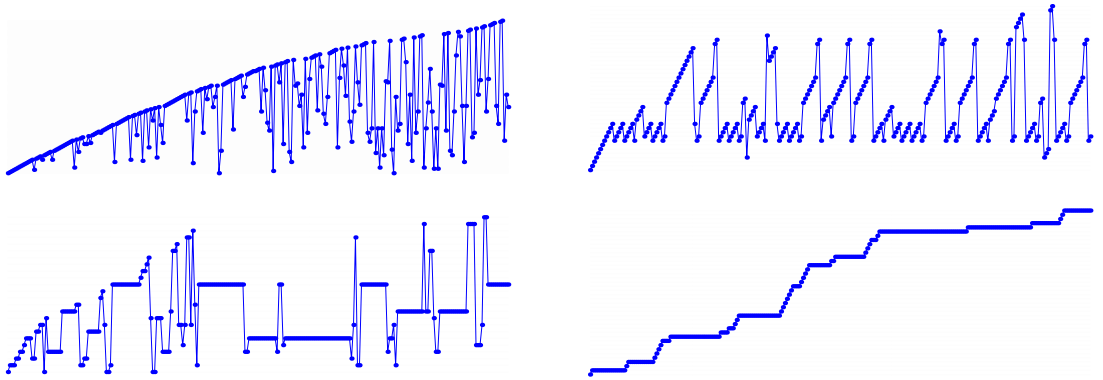
(Beal, Ghahramani, and Rasmussen, 2002) (Teh et al. 2005)

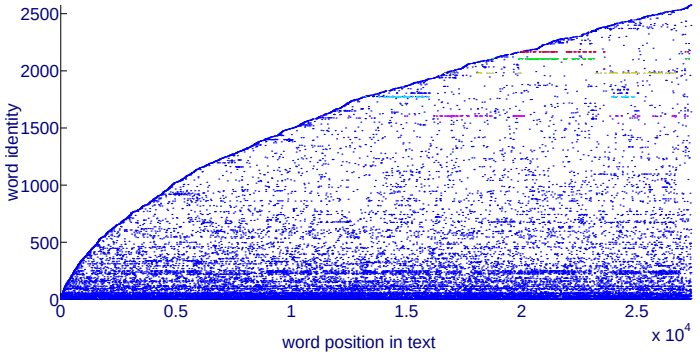
Infinite hidden Markov models (iHMMs)

Let the number of hidden states $K \to \infty$.

Here are some typical state trajectories for an iHMM. Note that the number of states visited grows with T.

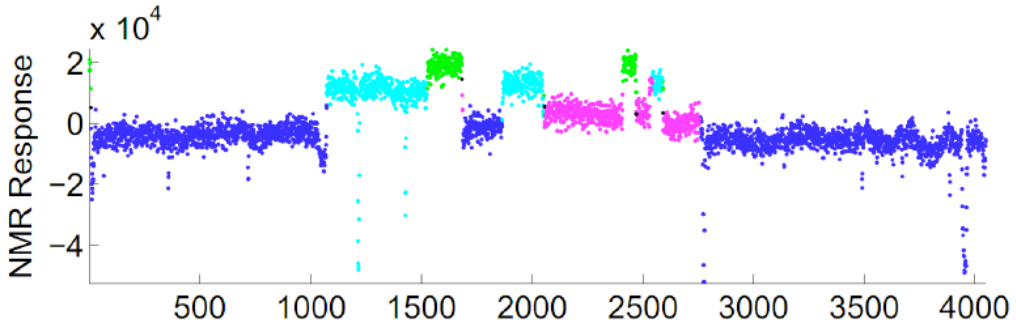


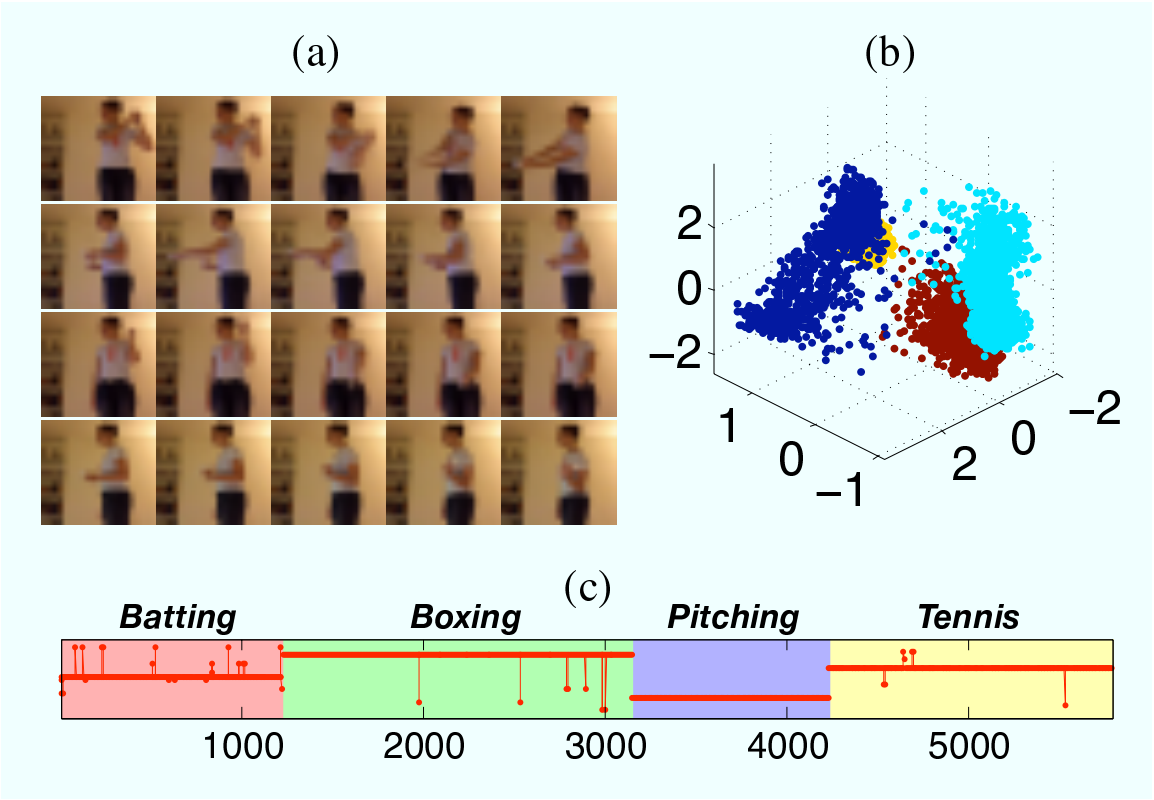




- Introduced in (Beal, Ghahramani and Rasmussen, 2002).
- Teh, Jordan, Beal and Blei (2005) showed that iHMMs can be derived from hierarchical Dirichlet processes, and provided a more efficient Gibbs sampler.
- We have recently derived a much more efficient sampler based on Dynamic Programming (Van Gael, Saatci, Teh, and Ghahramani, 2008). `http://mloss.org/software/view/205/`
- And we have parallel (.NET) and distributed (Hadoop) implementations (Bratieres, Van Gael, Vlachos and Ghahramani, 2010).

Infinite HMM: Changepoint detection and video segmentation

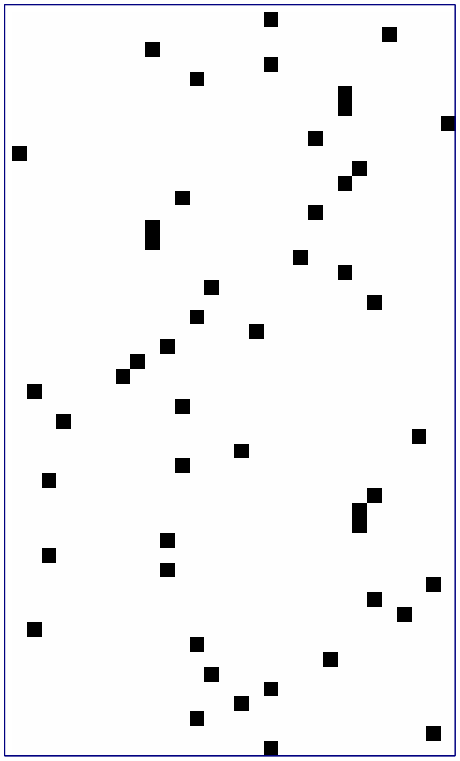




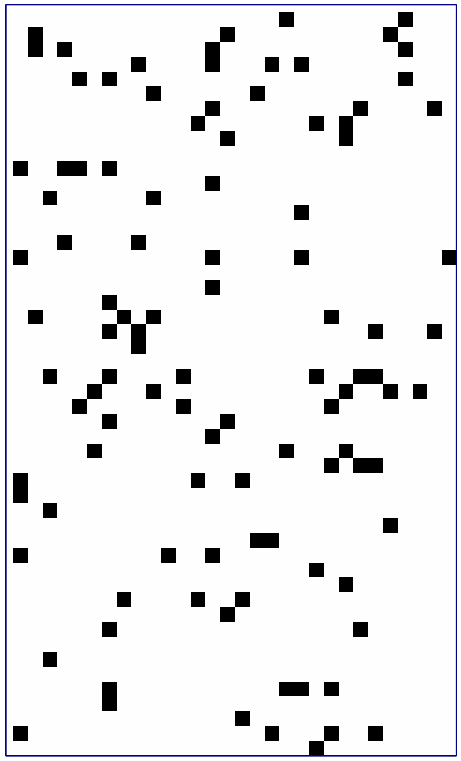
(w/ Tom Stepleton, 2009)

Sparse Matrices and Feature Models

Latent Cluster vs Feature Models



clustering

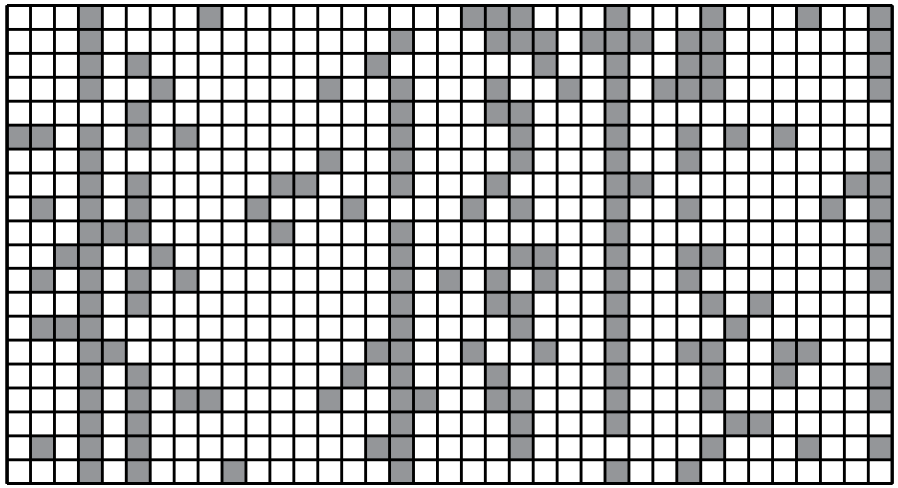


feature allocation

Chinese Restaurant Process (CRP)
Dirichlet process

Indian Buffet Process (IBP)
Beta process

Sparse binary matrices and latent feature models



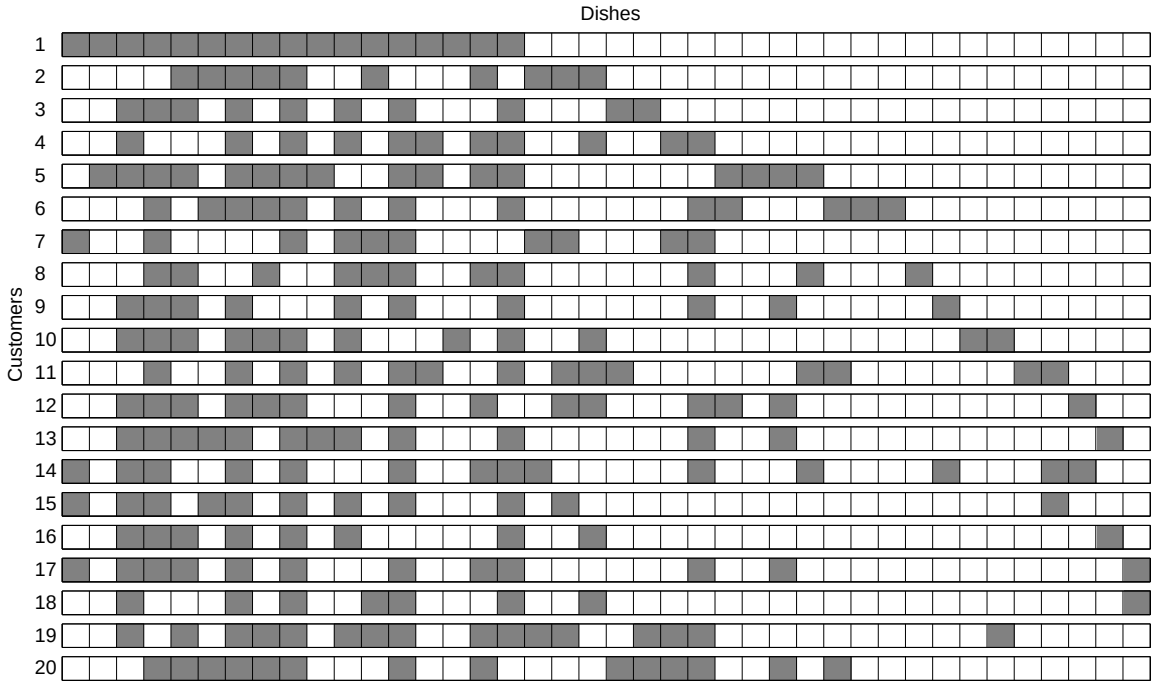
$z_{nk} = 1$ means object n has feature k:

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

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

- 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.
- $K \to \infty$ results in an Indian buffet process (IBP)

Indian buffet process



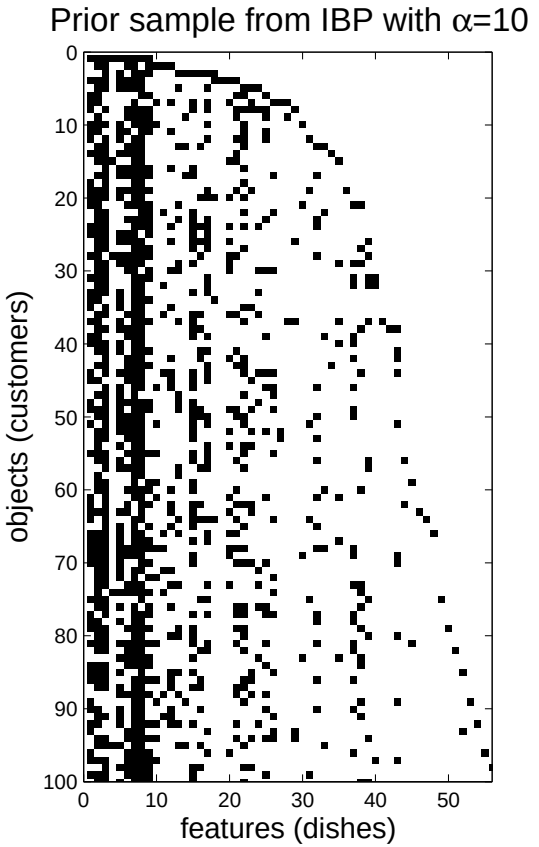
"Many Indian restaurants offer lunchtime buffets with an apparently infinite number of dishes"

- 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 his plate becomes overburdened.
- The n^{th} customer moves along the buffet, sampling dishes in proportion to their popularity, serving himself dish k with probability m_k/n, and trying a Poisson(α/n) number of new dishes.
- The customer-dish matrix, $\mathbf{Z}$, is a draw from the IBP.

(w/ Tom Griffiths 2006; 2011)

Properties of the Indian buffet process

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



Shown in (Griffiths and Ghahramani 2006, 2011):

- It is infinitely exchangeable.
- The number of ones in each row is Poisson(α)
- The expected total number of ones is αN.
- The number of nonzero columns grows as $O(\alpha \log N)$.

Additional properties:

- Has a stick-breaking representation (Teh, et al 2007)
- Has as its de Finetti mixing distribution the Beta process (Thibaux and Jordan 2007)
- More flexible two and three parameter versions exist (w/ Griffiths & Sollich 2007; Teh and Görür 2010)

Modelling Data with Indian Buffet Processes

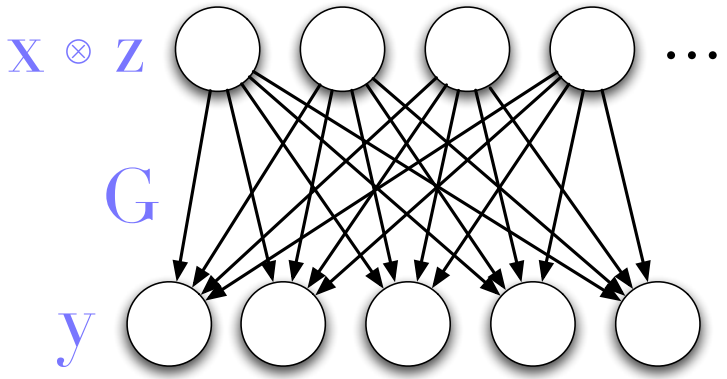
Latent variable model: let $\mathbf{X}$ be the $N \times D$ matrix of observed data, and $\mathbf{Z}$ be the $N \times K$ matrix of sparse binary latent features

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

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

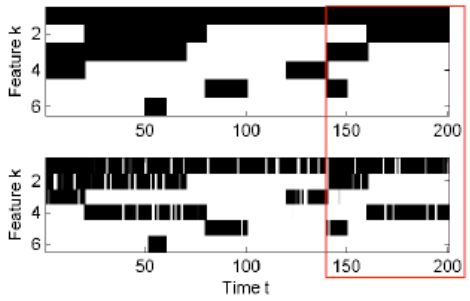
- Models for graph structures (w/ Wood, Griffiths, 2006; w/ Adams and Wallach, 2010)
- Models for protein complexes (w/ Chu, Wild, 2006)
- Models for choice behaviour (Görür & Rasmussen, 2006)
- Models for users in collaborative filtering (w/ Meeds, Roweis, Neal, 2007)
- Sparse latent trait, pPCA and ICA models (w/ Knowles, 2007, 2011)
- Models for overlapping clusters (w/ Heller, 2007)

Infinite Independent Components Analysis

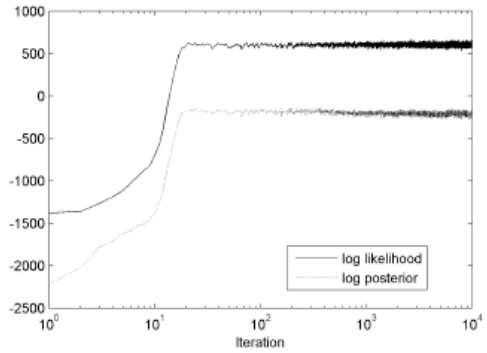


Model: $\mathbf{Y} = \mathbf{G}(\mathbf{Z} \otimes \mathbf{X}) + \mathbf{E}$

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, 2011)

Nonparametric Binary Matrix Factorization

genes $\times$ patients

users $\times$ movies

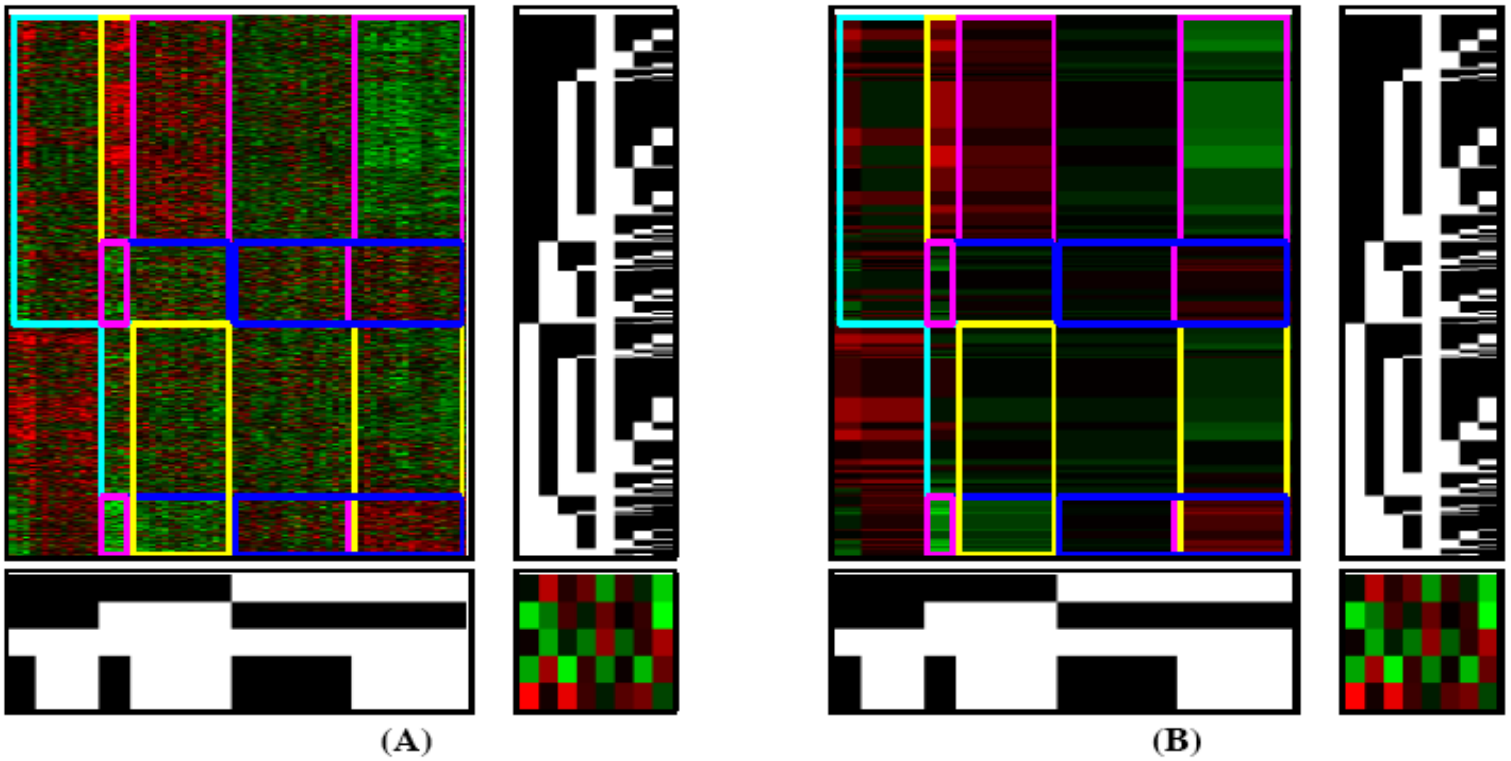
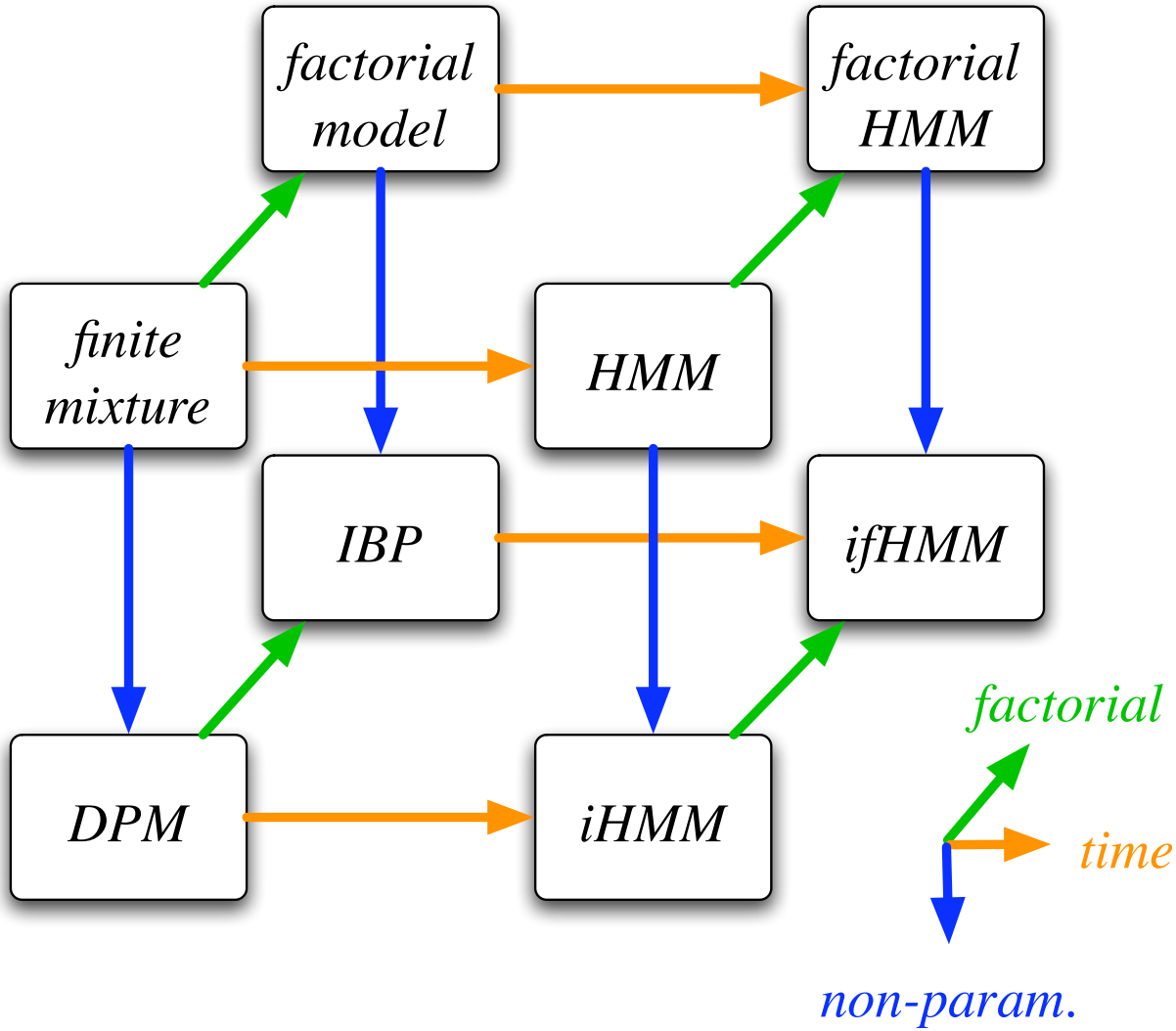


Figure 5: Gene expression results. **(A)** The top-left is $\mathbf{X}$ sorted according to contiguous features in the final $\mathbf{U}$ and $\mathbf{V}$ in the Markov chain. The bottom-left is $\mathbf{V}^\top$ and the top-right is $\mathbf{U}$. The bottom-right is $\mathbf{W}$. **(B)** The same as **(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.

Meeds et al (2007) Modeling Dyadic Data with Binary Latent Factors.

A Picture:
Relations between some models



Posterior Inference in IBP Models

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

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.

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) see also (Adams et al 2010).

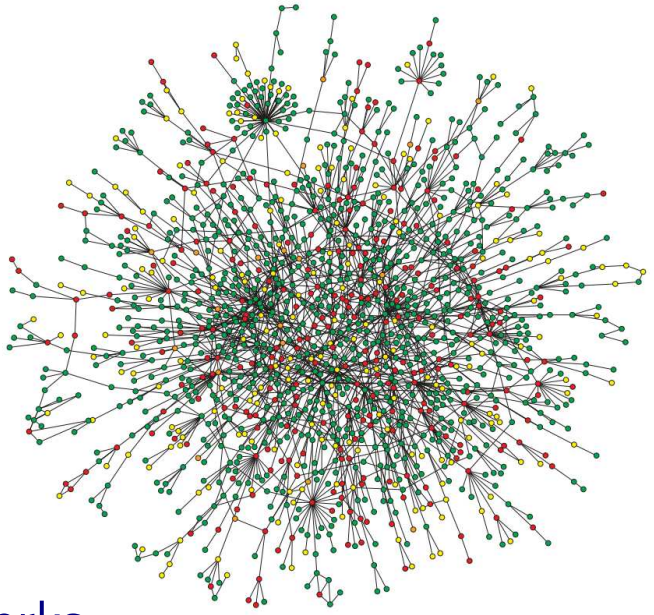
Deterministic Inference: variational inference (Doshi et al 2009a) parallel inference (Doshi et al 2009b), beam-search MAP (Rai and Daume 2011), power-EP (Ding et al 2010), submodular MAP (w/ Reed, 2013)

Some Case Studies

- Networks and relational data
- Scaling
- Discovering structure in Gaussian process kernels

Modelling Networks

We are interested in modelling networks.



Biological networks: protein-protein interaction networks

Social networks: friendship networks; co-authorship networks

We wish to have models that will be able to

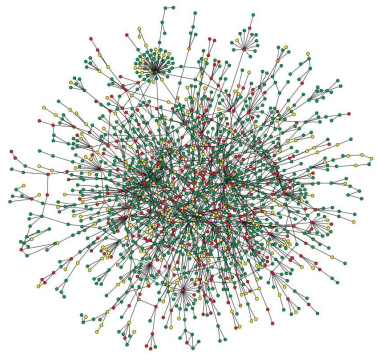
- predict missing links,
- infer latent properties or classes of the objects,
- generalise learned properties from smaller observed networks to larger networks.

Figure from Barabasi and Oltvai 2004: A protein-protein interaction network of budding yeast

Networks and Relational Data

Networks are just a way of representing certain kinds of *relational data*:

- `friend(John,Mary)`
- `buy(Jack,iPhone)`
- `rate(Fred,Titanic,5)`
- `cite(PaperA,PaperB)`
- `author(PaperA,John)`
- `regulate(TranscriptionFactorA,GeneB)` ...

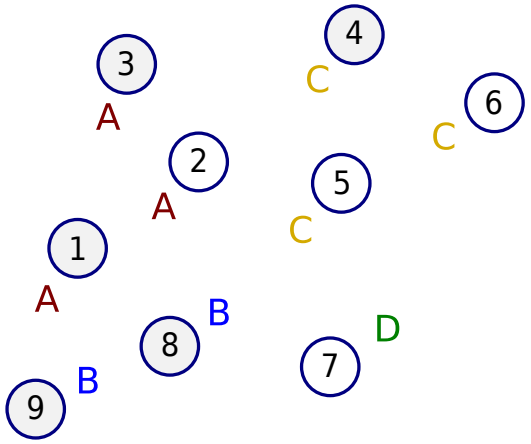


Relational data are ubiquitous; we need general models for such data.

There are deep and interesting connections between network modelling $\leftrightarrow$ matrix factorization $\leftrightarrow$ exchangeable arrays $\leftrightarrow$ relational data.

Nonparametric Latent Class Models

Infinite Relational Model (Kemp et al 2006)



Each node v_i has a hidden class $c_i \in \{1, \ldots, \infty\}$

For all i:

$$c_i | c_1, \ldots, c_{i-1} \sim \mathrm{CRP}(\alpha)$$

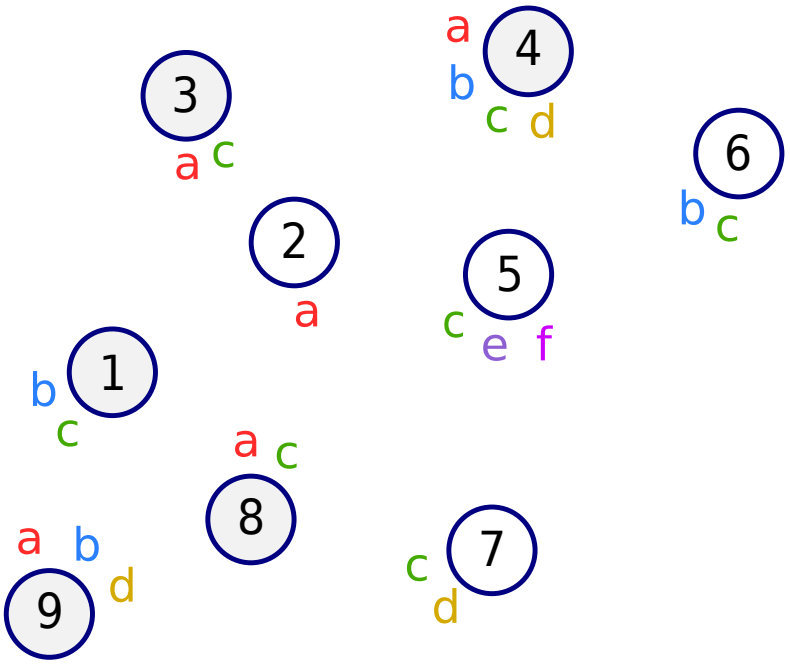
Probability of a link between two nodes v_i and v_j depends on their classes:

$$P(y_{ij} = 1 | c_i = k, c_j = \ell) = \rho_{k\ell}$$

Note that $\boldsymbol{\rho}$ is an infinitely large matrix, but if we give each element a beta prior we can integrate it out.

Inference done via MCMC. Fairly straightforward to implement.

Latent Feature Models



- Each node posses some number of latent features.
- Alternatively we can think of this model as capturing *overlapping clusters or communities*
- The link probability depends on the latent features of the two nodes.
- The model should be able to accommodate a potentially unbounded (infinite) number of latent features (e.g. (Miller, Griffiths and Jordan 2010) use an IBP).

Exchangeable Sequences

Exchangeable sequence:
A sequence is exchangeable if its joint distribution is invariant under arbitrary permutation of the indices:

$$(X_1, X_2, ...) \stackrel{d}{=} (X_{\pi(1)}, X_{\pi(2)}, ...) \ \ \forall \pi \in S_\infty.$$

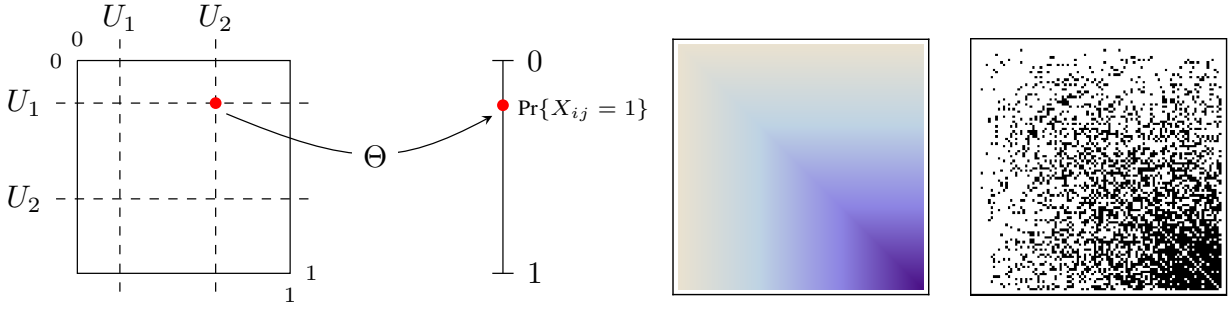
de Finetti's Theorem:
$(X_i)_{i \in N}$ is exchangeable if and only if there exists a random probability measure Θ on X such that $X_1, X_2, \ldots | \Theta \ \sim \ \text{iid} \ \Theta$

Interpretation:
Any probabilistic model of data which assumes that the order of the data does not matter, can be expressed as a Bayesian mixture of iid models. Note that Θ may in general need to be infinite dimensional (i.e. *nonparameteric*).

Exchangeable Arrays

Exchangeable arrays: An array $X = (X_{ij})_{i,j\in\mathbb{N}}$ is called an exchangeable array if $(X_{ij}) \stackrel{d}{=} (X_{\pi(i)\pi(j)})$ for every $\pi \in S_\infty$.

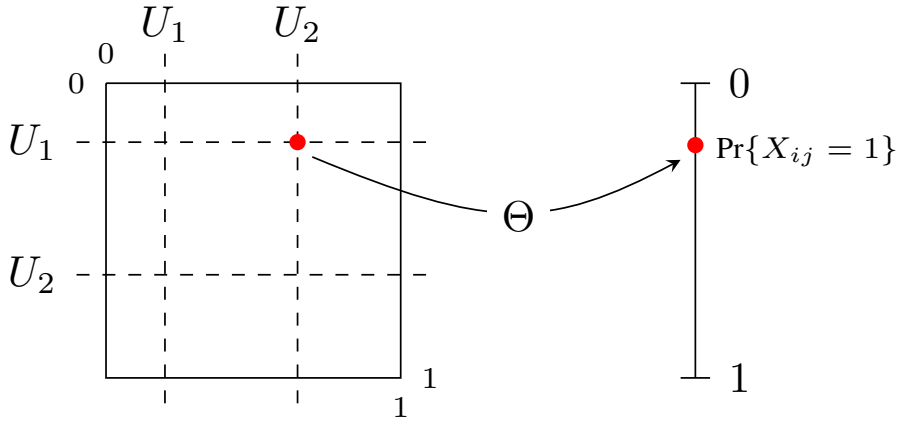
Aldous-Hoover Theorem:
A random matrix (X_{ij}) is exchangeable if and only if there is a random (measurable) function $F : [0,1]^3 \to X$ such that $(X_{ij}) \stackrel{d}{=} (F(U_i, U_j, U_{ij}))$ for every collection $(U_i)_{i\in\mathbb{N}}$ and $(U_{ij})_{i\leq j\in\mathbb{N}}$ of i.i.d. Uniform$[0,1]$ random variables, where $U_{ji} = U_{ij}$ for $j < i \in \mathbb{N}$.

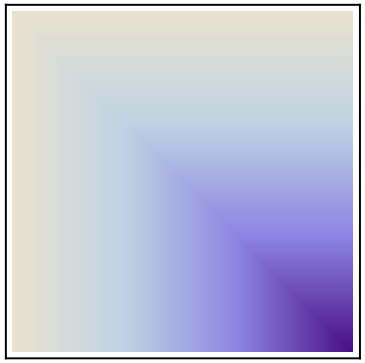


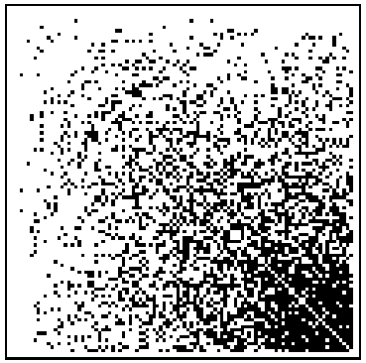
Interpretation:
Any model of matrices, arrays (or graphs) where the order of rows and columns (nodes) is irrelevant can be expressed by assuming *latent variables* associated with each row and column, and a *random function* mapping these latent variables to the observations.

Random Function Model

We develop a nonparametric probabilistic model for simple arrays and graphs that makes explicit the Aldous Hoover representation:







$$
\begin{aligned}
\Theta &\sim \mathrm{GP}(0, \kappa) \\
U_1, U_2, \ldots &\overset{\text{iid}}{\sim} \mathrm{Uniform}[0, 1] \\
W_{ij} &= \Theta(U_i, U_j) \\
X_{ij} &\sim P[\cdot | W_{ij}]
\end{aligned}
$$

(w/ James Lloyd, Dan Roy, Peter Orbanz, NIPS 2012)

Random Function Model

The random function model can be related to a number of existing models for matrices, arrays/tensors, and graphs.

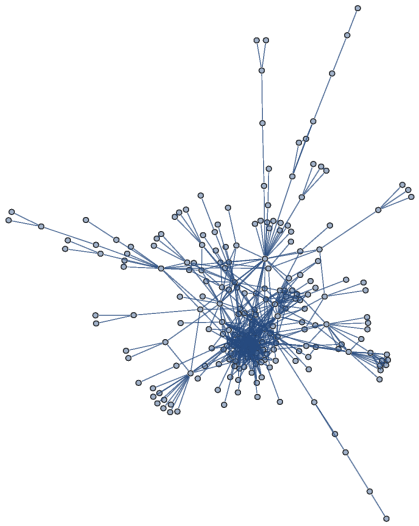
Graph data

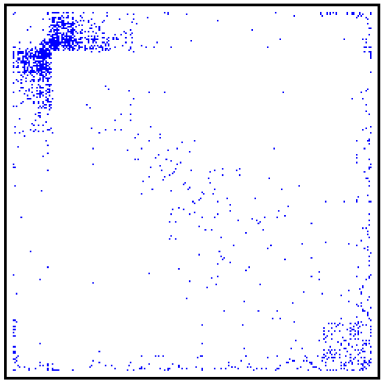
Random function model	Θ	$\sim$	$\mathcal{GP}(0, \kappa)$
Latent class	W_{ij}	$=$	$m_{U_i U_j}$ where $U_i \in \{1, \ldots, K\}$
IRM	W_{ij}	$=$	$m_{U_i U_j}$ where $U_i \in \{1, \ldots, \infty\}$
Latent distance	W_{ij}	$=$	$-\lvert U_i - U_j \rvert$
Eigenmodel	W_{ij}	$=$	$U_i' \Lambda U_j$
LFRM	W_{ij}	$=$	$U_i' \Lambda U_j$ where $U_i \in \{0, 1\}^\infty$
ILA	W_{ij}	$=$	$\sum_d \mathbb{I}_{U_{id}} \mathbb{I}_{U_{jd}} \Lambda^{(d)}_{U_{id} U_{jd}}$ where $U_i \in \{0, \ldots, \infty\}^\infty$
SMGB	Θ	$\sim$	$\mathcal{GP}(0, \kappa_1 \otimes \kappa_2)$

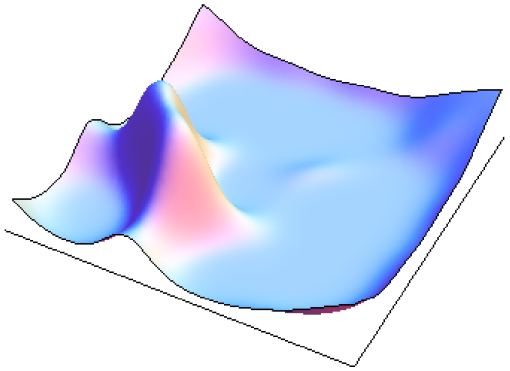
Real-valued array data

Random function model	Θ	$\sim$	$\mathcal{GP}(0, \kappa)$
Mondrian process based	Θ	$=$	piece-wise constant random function
PMF	W_{ij}	$=$	$U_i' V_j$
GPLVM	Θ	$\sim$	$\mathcal{GP}(0, \kappa \otimes \delta)$

Random Function Model: Results







AUC results

Data set	High school			NIPS			Protein		
Latent dimensions	1	2	3	1	2	3	1	2	3
PMF	0.747	0.792	0.792	0.729	0.789	0.820	0.787	0.810	0.841
Eigenmodel	0.742	0.806	0.806	0.789	0.818	0.845	0.805	0.866	0.882
GPLVM	0.744	0.775	0.782	0.888	0.876	0.883	0.877	0.883	0.873
RFM	**0.815**	**0.827**	**0.820**	**0.907**	**0.914**	**0.919**	**0.903**	**0.910**	**0.912**

Scalable approximate inference

Scalable approximate inference

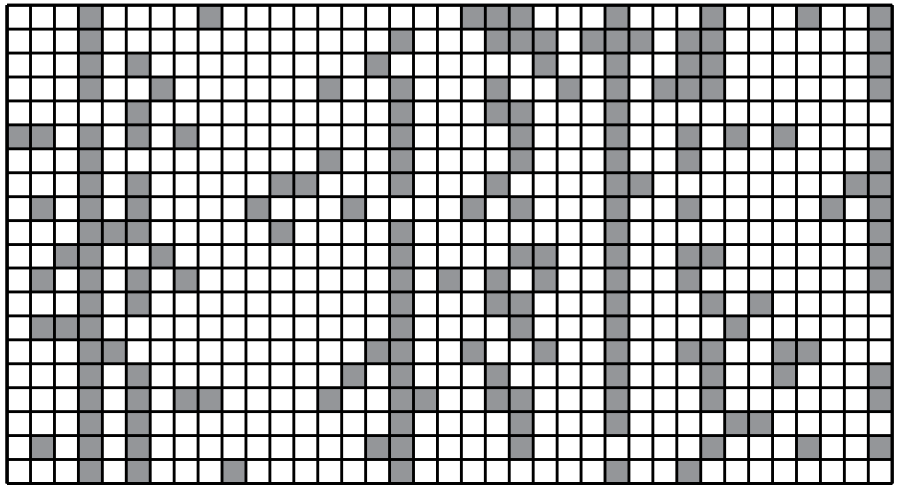
$$P(x|\mathcal{D},m) = \int P(x|\theta,\mathcal{D},m)P(\theta|\mathcal{D},m)d\theta$$

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

How do we compute these integrals in practice?

- Laplace Approximation
- Bayesian Information Criterion (BIC)
- Variational Bayesian approximations
- Expectation Propagation (and loopy belief propagation)
- Markov chain Monte Carlo
- Sequential Monte Carlo
- ...

Review of IBPs:
Sparse binary matrices and feature allocation models



$z_{nk} = 1$ 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.
- $K \to \infty$ results in an Indian buffet process (IBP)

Submodular MAP inference in IBPs

Approach: maximise over Z while maintaining a variational approximation to the integral over the other model parameters

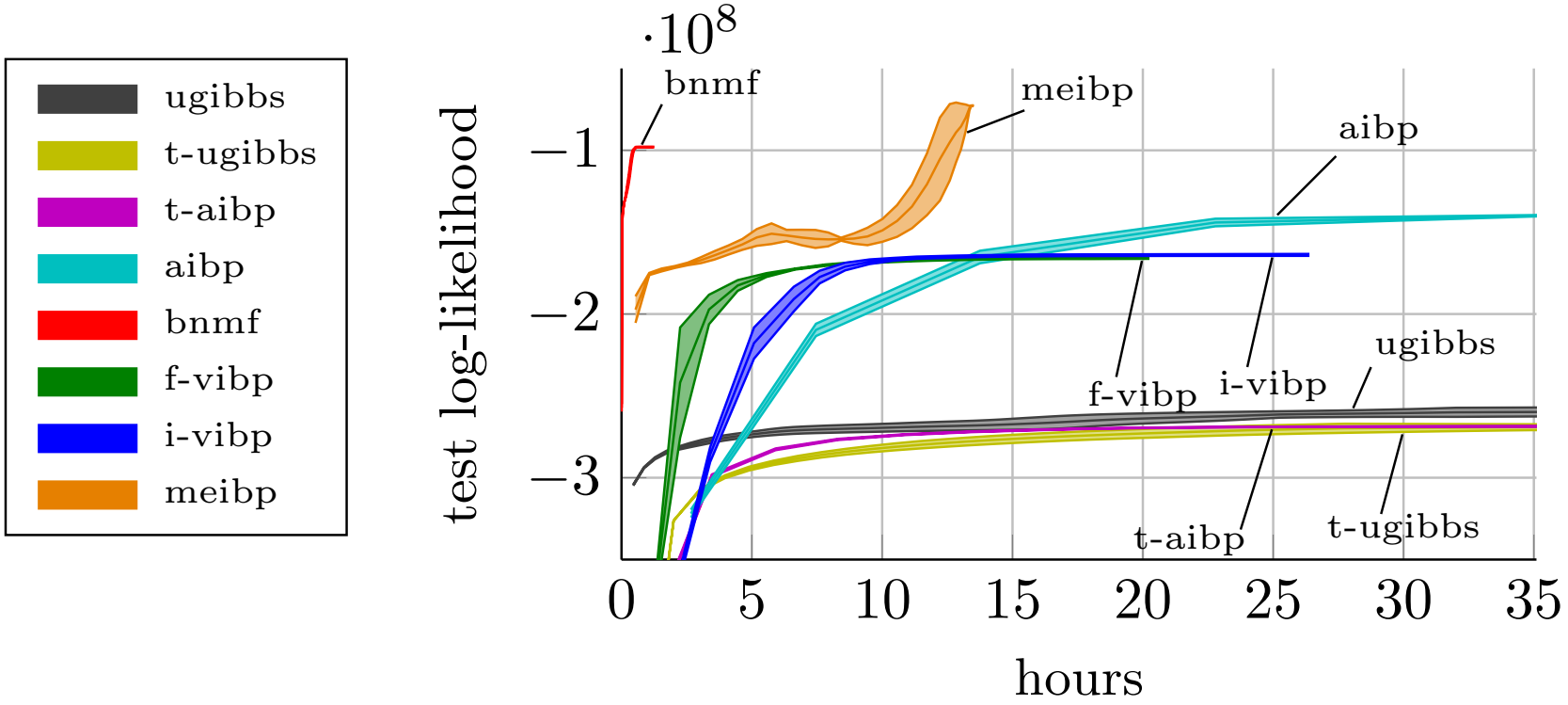
Yields a *submodular maximisation* algorithm that can be maximized with a simple greedy algorithm with a (1/3)-approximability guarantee

Submodularity: (diminishing returns) for $A \subseteq B$ and f a set function:

$$f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B)$$

(w/ Colorado Reed, ICML 2013)

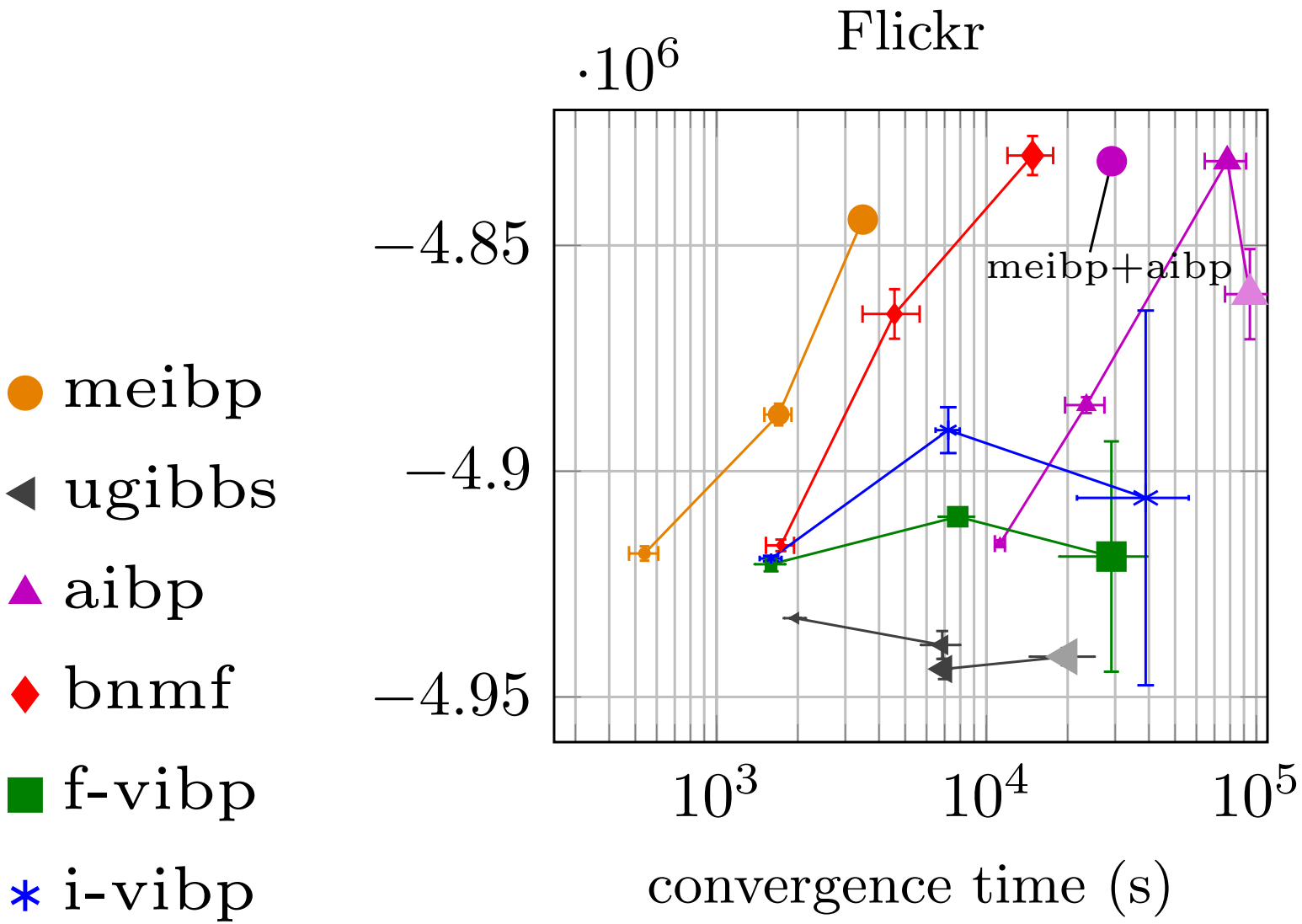
Submodular MAP inference in IBPs: Results



Synthetic datasets: $N = 10^5$ data points, $D = 10^3$ dimensions, $K_{\max} = 50$ features.

meibp:	our new method based on submodular maximisation (w/ Reed, 2013)
aibp:	accelerated IBP (w/ Doshi-Velez, 2009)
ugibbs:	uncollapsed Gibbs sampling (w/ Doshi-Velez, 2009)
vibp:	variational IBP (Doshi-Velez et al, 2009)
bnmf:	iterated conditional modes for NMF (Schmidt et al 2009)
bs-ibp:	beam-search (didn't complete one iteration): (Rai and Daume, 2011)
inmf:	power-EP for non-negative IBP (didn't complete one iteration) (Ding et al, 2010)

Submodular MAP inference in IBPs: Results



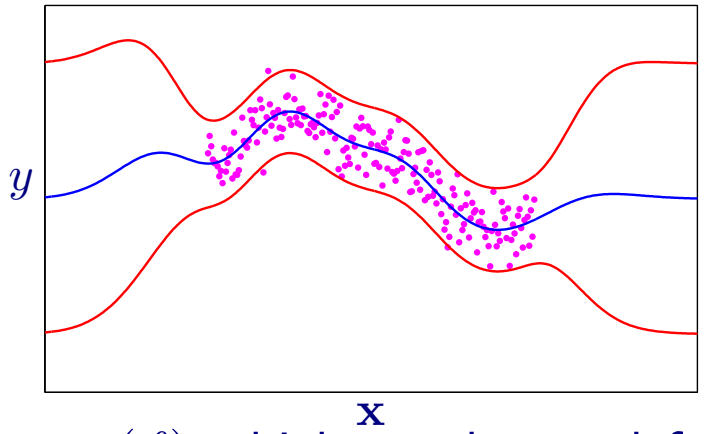
binary image-tag indicators from Flickr - dataset of 25000 images by 1500 indicators.

Structure Discovery for Gaussian Process Kernels

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 on functions $p(f)$ which can be used for Bayesian regression:

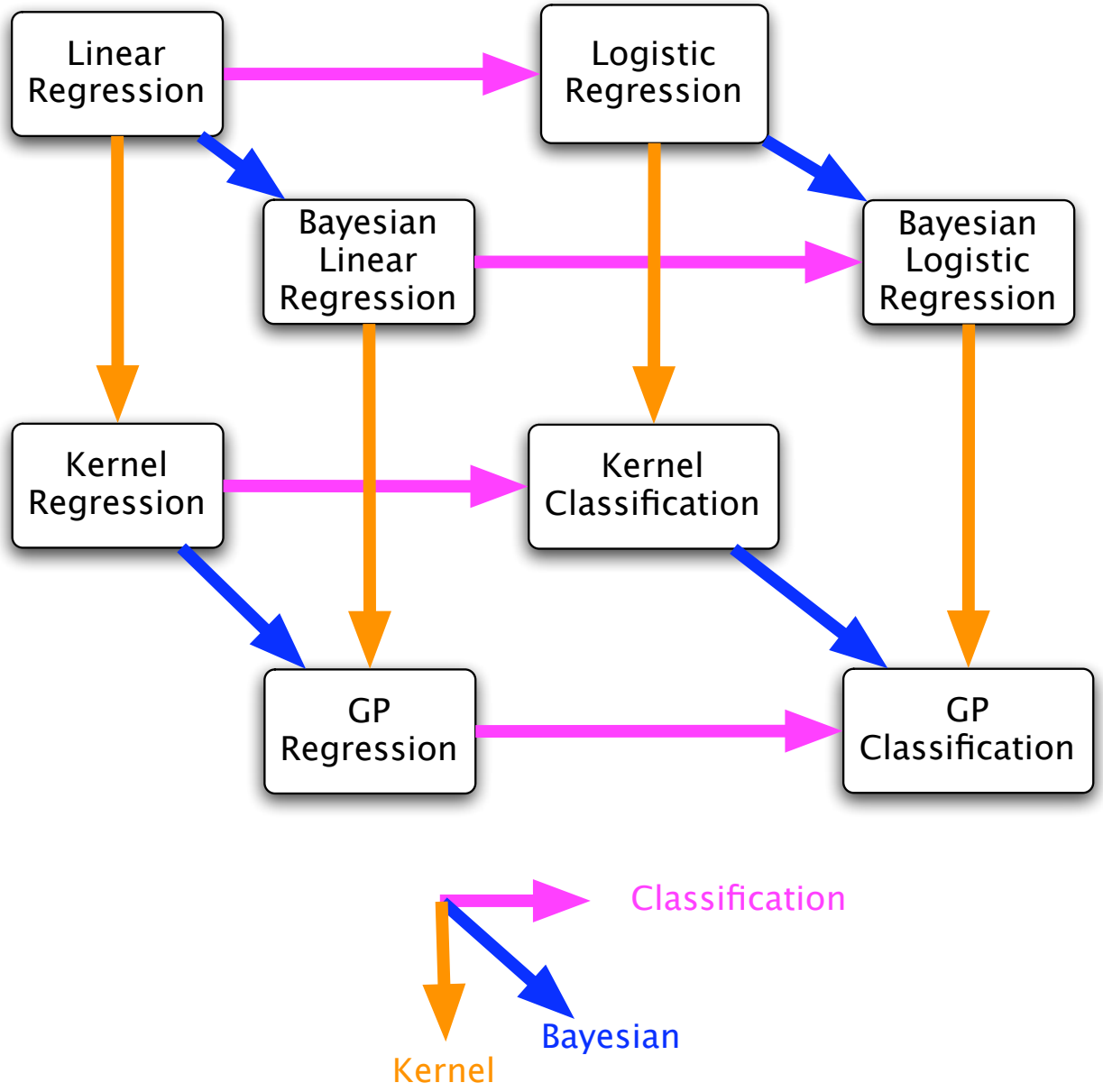
$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})}$$

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.

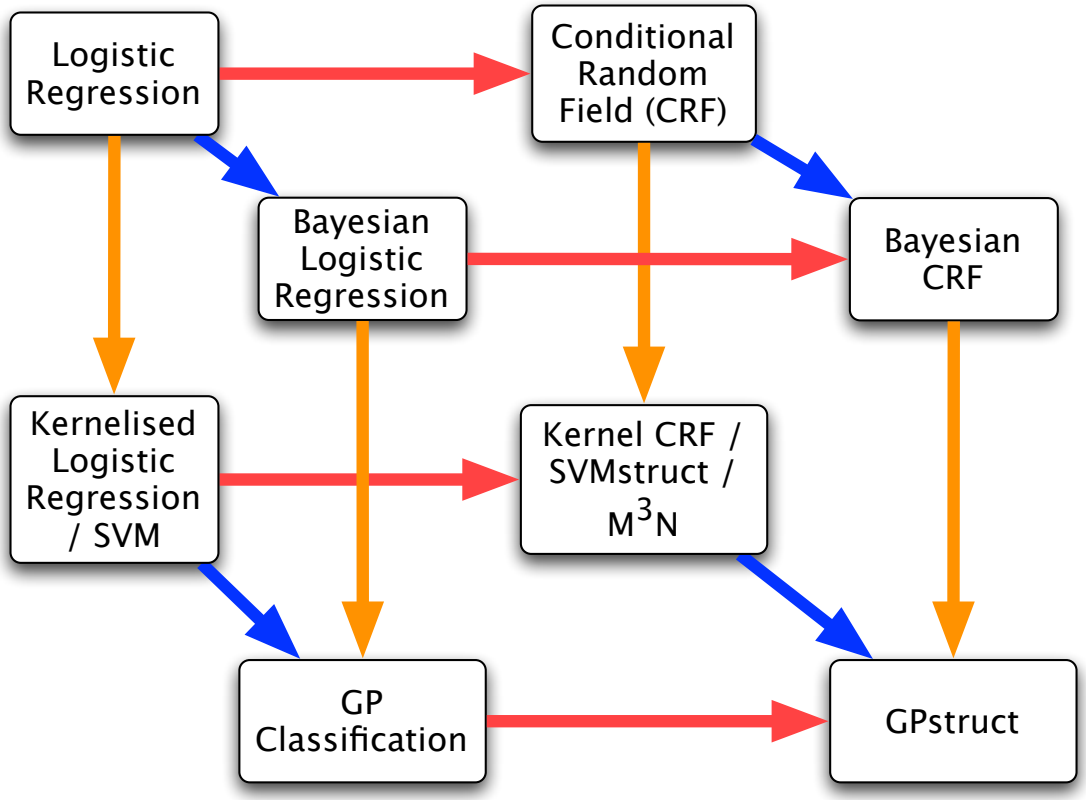
Definition: $p(f)$ is a Gaussian process if for any finite subset $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the marginal distribution over that subset $p(\mathbf{f})$ is multivariate Gaussian.

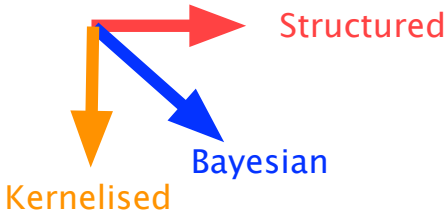
$$f \sim \mathrm{GP}(\mu, K)$$

Bayesian kernelised regression and classification

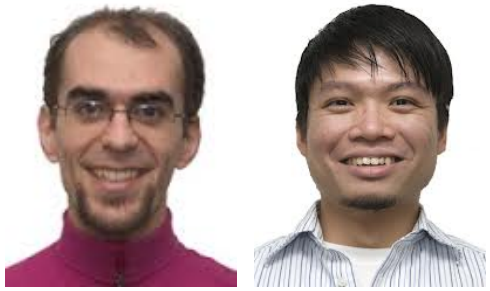


Bayesian kernelised structured prediction





(w/ Bratieres and Quadrianto, arXiv 2013)



How do we learn the kernel (covariance function)?

$$f \sim \mathrm{GP}(\mu, K)$$

- Usual approach: parametrise the kernel with a few hyperparameters and optimise or infer these. An example covariance function:

$$K(x_i, x_j) = v_0 \exp\left\{-\left(\frac{|x_i - x_j|}{r}\right)^{\alpha}\right\} + v_1 + v_2\,\delta_{ij}$$

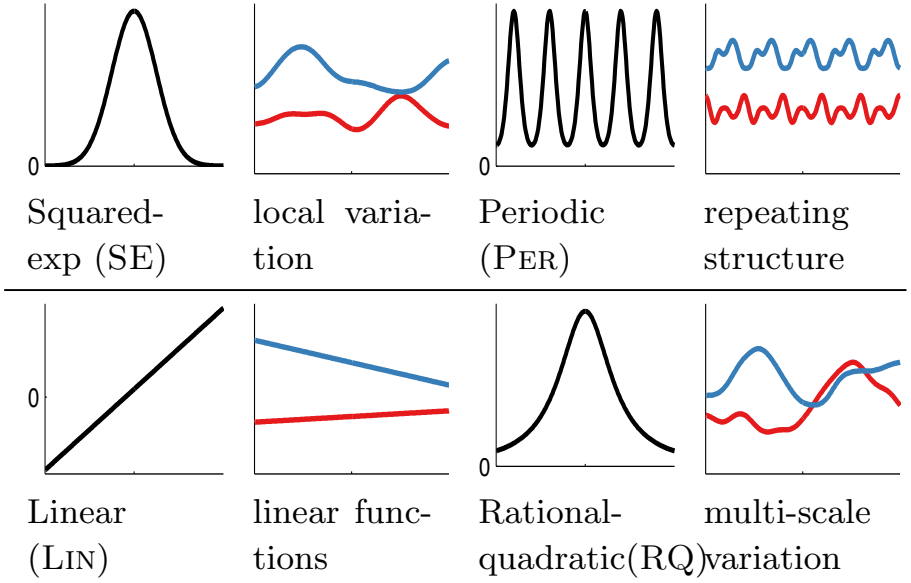
  with parameters $(v_0, v_1, v_2, r, \alpha)$. These kernel parameters are interpretable and can be learned from data:

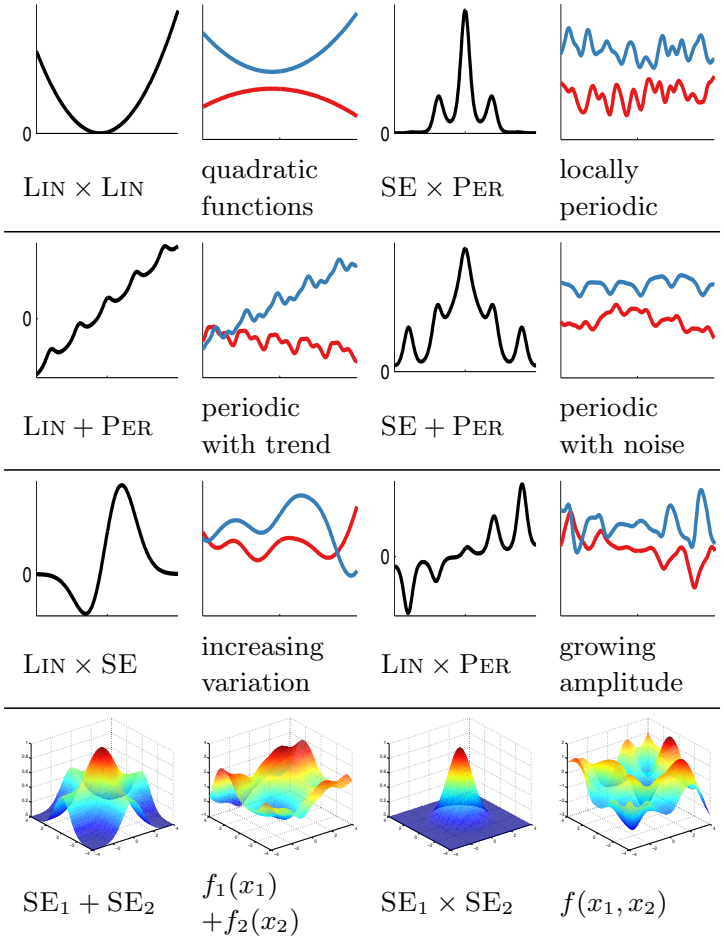
v_0	signal variance
v_1	variance of bias
v_2	noise variance
r	lengthscale
α	roughness

- Structure discovery for the kernel by searching over a grammar of kernels

Kernel Composition

By taking a few simple **base kernels** and two **composition rules**, kernel addition and multiplication, we can span a rich space of structured kernels.

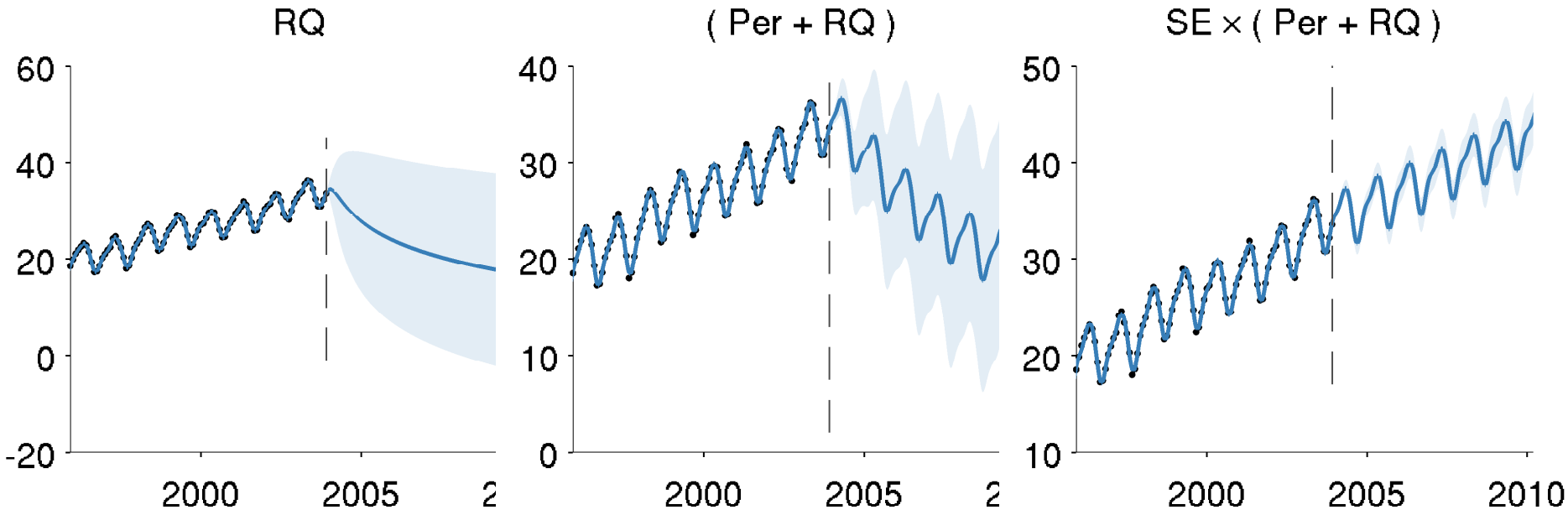




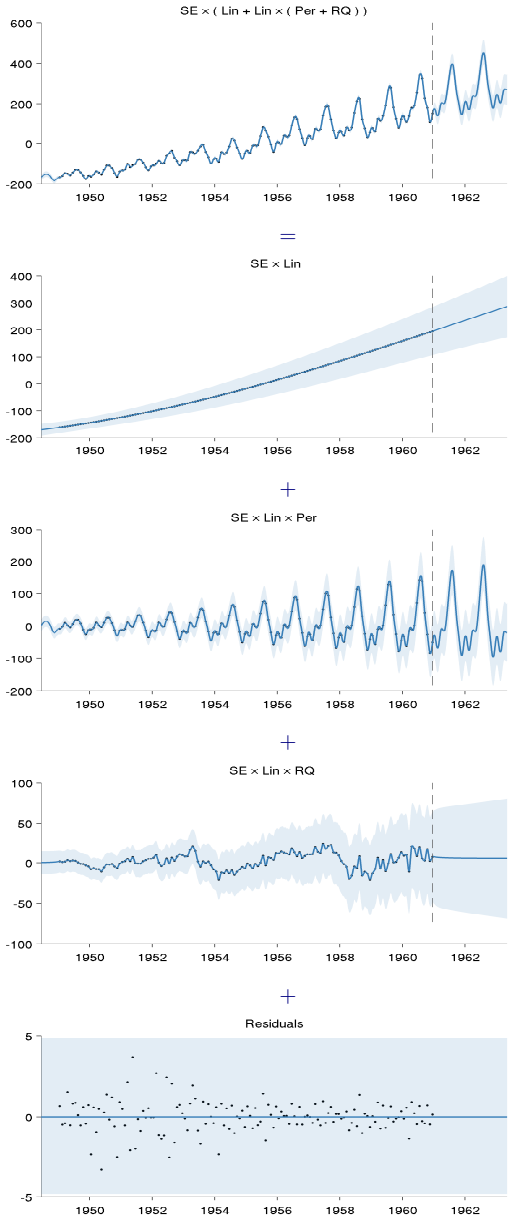
(w/ Duvenaud, Lloyd, Grosse, and Tenenbaum, ICML 2013)

(Wilson and Adams, ICML 2013)

Kernel Composition: Mauna Loa CO_2 Keeling Curve



Kernel Composition: Airline passenger prediction



Kernel Composition: results

	Mean Squared Error (MSE)					Negative Log-Likelihood				
Method	bach	concrete	puma	servo	housing	bach	concrete	puma	servo	housing
Linear Regression	1.031	0.404	0.641	0.523	0.289	2.430	1.403	1.881	1.678	1.052
GAM	1.259	0.149	0.598	0.281	0.161	1.708	0.467	1.195	0.800	0.457
HKL	**0.199**	0.147	0.346	0.199	0.151	-	-	-	-	-
GP SE-ARD	**0.045**	0.157	0.317	0.126	**0.092**	**−0.131**	0.398	0.843	0.429	0.207
GP Additive	**0.045**	**0.089**	**0.316**	**0.110**	0.102	**−0.131**	**0.114**	**0.841**	**0.309**	0.194
Structure Search	**0.044**	**0.087**	**0.315**	**0.102**	**0.082**	**−0.141**	**0.065**	**0.840**	**0.265**	**0.059**

Summary of kernel structure discovery

Structure learning on kernels can be useful both for good automated prediction, and for interpretable results.

...towards and **automated statistician**.

Summary

- Bayesian machine learning is simply the application of probability theory to learning from data.
- Nonparametrics is needed to capture complexity of real data.

- **New models for complex structured data**:
  - Random Function Model for exchangeable arrays and relations
- **Fast scalable inference**:
  - submodular maximisation in IBPs
- **Automatic structure discovery**:
  - kernel structure discovery in Gaussian processes

Thanks to


Konstantina Palla


David Knowles


Colorado Reed


Andrew Wilson


James Lloyd


Peter Orbanz


Dan Roy


David Duvenaud

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

`zoubin@eng.cam.ac.uk`

postdocs available! to be advertised on `ml-news@googlegroups.com`

Some References

- Adams, R.P., Wallach, H., Ghahramani, Z. (2010) Learning the Structure of Deep Sparse Graphical Models. AISTATS 2010.
- Beal, M. J., Ghahramani, Z. and Rasmussen, C. E. (2002) The infinite hidden Markov model. NIPS **14**:577–585.
- Bratieres, S., van Gael, J., Vlachos, A., and Ghahramani, Z. (2010) Scaling the iHMM: Parallelization versus Hadoop. International Workshop on Scalable Machine Learning and Applications (SMLA-10), 1235–1240.
- Bratieres, S., Quadrianto, N, and Ghahramani, Z. (2013) Bayesian Structured Prediction Using Gaussian Processes. http://arxiv.org/abs/1307.3846
- Bru, M. (1991). Wishart processes. *Journal of Theoretical Probability* 4(4):725751.
- Doshi-Velez, F. and Z. Ghahramani. (2009) Accelerated Sampling for the Indian Buffet Process. In *International Conference on Machine Learning (ICML 2009)*.
- Doshi-Velez, F., K.T. Miller, J. Van Gael, and Y.W. Teh. (2009) Variational Inference for the Indian Buffet Process. In *Artificial Intelligence and Statistics Conference (AISTATS 2009)*.
- Duvenaud, D., Lloyd, J. R., Grosse, R., Tenenbaum, J. B. and Ghahramani, Z. (2013) Structure Discovery in Nonparametric Regression through Compositional Kernel Search. ICML 2013.
- Ghahramani, Z. (2013) Bayesian nonparametrics and the probabilistic approach to modelling. *Phil. Trans. R. Soc. A* 371: 20110553.
- Griffiths, T.L., and Ghahramani, Z. (2006) Infinite Latent Feature Models and the Indian Buffet Process. NIPS **18**:475–482.
- Griffiths, T.L., and Ghahramani, Z. (2011) The Indian buffet process: An introduction and review. *Journal of Machine Learning Research* **12**(Apr):1185–1224.

- Kemp, C., J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. (2006) Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st National Conference on Artificial Intelligence*.
- Lloyd, J., Orbanz, P., Ghahramani, Z. and Roy, D. (2012) Random function priors for exchangeable graphs and arrays. NIPS 2012.
- Meeds, E., Ghahramani, Z., Neal, R. and Roweis, S.T. (2007) Modeling Dyadic Data with Binary Latent Factors. NIPS **19**:978–983.
- Miller, K.T., T. L. Griffiths, and M. I. Jordan. (2010) Nonparametric latent feature models for link predictions. In *Advances in Neural Information Processing Systems 22*.
- Neal, R.M. (2000) Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9:249–265.
- Nowicki, K. and Snijders, T. A. B. (2001) Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96:1077–1087.
- Orbanz, P. (2010) Construction of nonparametric Bayesian models from parametric Bayes equations. In *Advances in Neural Information Processing Systems 22*, 2010.
- Rasmussen, C.E. and Williams, C.K.I. (2006) *Gaussian processes for Machine Learning*. The MIT Press.
- Reed, C. and Ghahramani, Z. (2013) Scaling the Indian Buffet Process via Submodular Maximization. ICML 2013.
- Stepleton, T., Ghahramani, Z., Gordon, G., Lee, T.-S. (2009) The Block Diagonal Infinite Hidden Markov Model. AISTATS 2009, 552–559.
- Teh, Y.W., Jordan, M.I, Beal, M. and Blei, D. (2004) Hierarchical Dirichlet processes. In *Advances in Neural Information Processing Systems 17*. MIT Press, Cambridge, MA.

- Teh, Y.W., D. Görür, and Z. Ghahramani (2007) Stick-breaking construction for the Indian buffet process. In *Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, San Juan, Puerto Rico.
- Teh, Y.W. and Görür, D. (2010) Indian Buffet Processes with Power-law Behavior. In NIPS 2010.
- Thibaux, R. and M. I. Jordan. (2007) Hierarchical Beta processes and the Indian buffet process. In *Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*.
- Wilson, A.G. and Adams, R.P. (2013) Gaussian process covariance kernels for pattern discovery and extrapolation. International Conference on Machine Learning (ICML).
- Wood, F. and T. L. Griffiths. (2007) Particle filtering for nonparametric Bayesian matrix factorization. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1513–1520. MIT Press, Cambridge, MA, 2007.
- Wood, F., T. L. Griffiths, and Z. Ghahramani. (2006) A non-parametric Bayesian method for inferring hidden causes. In *Proceedings of the 22nd Conference in Uncertainty in Artificial Intelligence (UAI '06)*.
- van Gael, J., Saatci, Y., Teh, Y.-W., and Ghahramani, Z. (2008) Beam sampling for the infinite Hidden Markov Model. ICML 2008, 1088-1095.
- van Gael, J and Ghahramani, Z. (2010) Nonparametric Hidden Markov Models. In Barber, D., Cemgil, A.T. and Chiappa, S. *Inference and Learning in Dynamic Models*. CUP.

Appendix

Dirichlet Process: Conjugacy

$$G \sim \mathrm{DP}(\cdot|G_0, \alpha)$$

If the prior on G is a DP:

$$P(G) = \mathrm{DP}(G|G_0, \alpha)$$

...and you observe θ...

$$P(\theta|G) = G(\theta)$$

...then the posterior is also a DP:

$$P(G|\theta) = \frac{P(\theta|G)P(G)}{P(\theta)} = \mathrm{DP}\left(\frac{\alpha}{\alpha+1}G_0 + \frac{1}{\alpha+1}\delta_\theta,\ \alpha+1\right)$$

Generalization for n observations:

$$P(G|\theta_1, \ldots, \theta_n) = \mathrm{DP}\left(\frac{\alpha}{\alpha+n}G_0 + \frac{1}{\alpha+n}\sum_{i=1}^{n}\delta_{\theta_i},\ \alpha+n\right)$$

Analogous to Dirichlet being conjugate to multinomial observations.

Dirichlet Process

Blackwell and MacQueen's (1973) urn representation

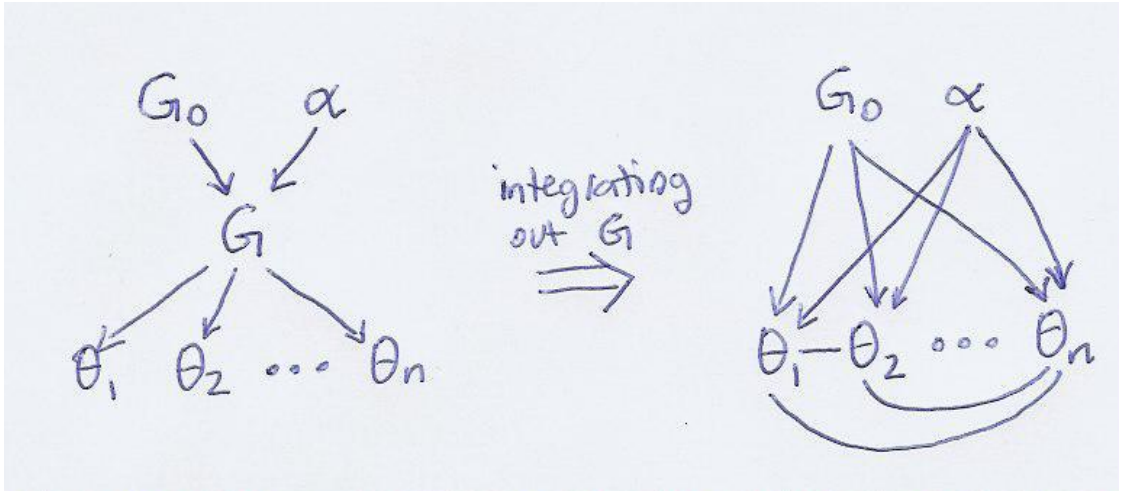
$$G \sim \mathrm{DP}(\cdot|G_0, \alpha) \quad \text{and} \quad \theta|G \sim G(\cdot)$$

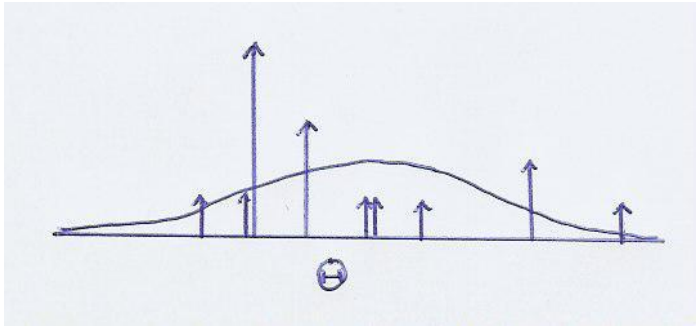
Then

$$\theta_n|\theta_1, \ldots \theta_{n-1}, G_0, \alpha \sim \frac{\alpha}{n-1+\alpha} G_0(\cdot) + \frac{1}{n-1+\alpha} \sum_{j=1}^{n-1} \delta_{\theta_j}(\cdot)$$

$$P(\theta_n|\theta_1, \ldots \theta_{n-1}, G_0, \alpha) \propto \int dG \prod_{j=1}^{n} P(\theta_j|G) P(G|G_0, \alpha)$$

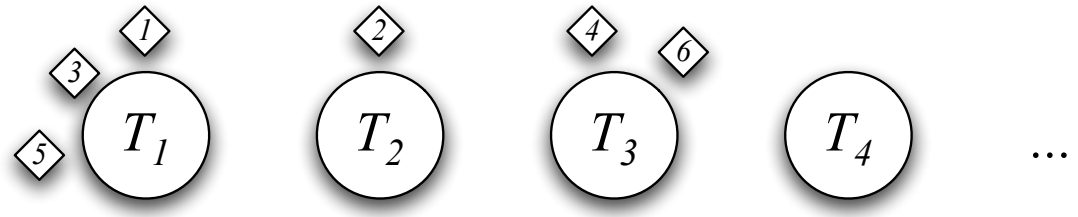
The model exhibits a "clustering effect".





Chinese Restaurant Process

The CRP generates samples from the distribution on 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)